MASTER THESIS

—

Real-Time Procedural Risers in Games: Mapping Riser Audio Effects to Game Data

—

Author:
Stijn de Koning

Supervisor:
Mata Haggis-Burridge

This thesis is submitted in fulfilment of the requirements
for the degree of Master Game Technology

In the

International Games Architecture and Design
Academy for Digital Entertainment

June 25, 2021

Declaration of Authorship

I, Stijn de Koning, declare that this thesis titled, “Real-Time Procedural Risers in Games: Mapping Riser Audio Effects to Game Data” and the work presented are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: *June 24, 2021*

Breda University of Applied Sciences

Abstract

International Games Architecture and Design
Academy for Digital Entertainment

Master of Game Technology

Real-Time Procedural Risers in Games:
Mapping Riser Audio Effects to Game Data

By Stijn de Koning

The application of audio to a nonlinear context causes several issues in the game audio design workflow. Mainly, industry standard software is not optimised for creating nonlinear audio, and a disconnect exists between the game's timeline and the audio timeline or grid. Standard middleware solutions improve the audio development workflow, but also cause new issues, such as requiring users to switch back and forth between different software before being able to test a sound in its context. This thesis applied procedural riser audio effects to a walking simulator horror game to explore an alternative to game audio development. The research investigated which are the essential parameters of the riser, and how they can be implemented in Unreal Engine based on the game's state and events. Interviews have been done with game audio designers, programmers, and leads/directors. The interviewees were shown a prototype, developed for this research, that allows the user to design a riser, control how it should adapt, and implement it in Unreal Engine. Analysis of the interviews indicates that modulating the volume and pitch of the riser over multiple layers are the most essential parameters. These parameters can best be modulated based on the player's current position, the elapsed time, and the amount of action over time. Furthermore, the data indicates that the approach put forward in the research is a viable and relevant solution to the stated issues and worth further investigation.

Acknowledgements

I would like to express my deepest appreciation for my project supervisor, Mata-Haggis Burridge, who guided me throughout the entire project and donated much of his time and knowledge. Furthermore, I would also like to thank my second reader, Thomas Buijtenweg, who was always available for feedback or questions I had.

In addition, I would like to thank all industry professionals who took time out of their busy schedules to do an interview: Will Davis, Clément Duquesne, Nicolas Fournel, Florian Fuesslin, Twan de Graaf, Tom Hays, Charlie Huguenard, Tom de Smit, Bogdan Vera, and Jonathan van den Wijngaarden. They provided countless valuable insights on the topic, as well as the industry as a whole, which has been of insurmountable value.

Finally, I would like to thank to everybody else who supported me throughout the project, including my family and friends. Special thanks goes out to Ciska Vriezenga who shared her advice and brainstormed with me throughout the project.

Contents

List of Figures

List of Abbreviations

AMP Amplifier

CPU Central Processing Unit

DAW Digital Audio Workstation

GDC Game Developers Conference

LFO Low Frequency Oscillator

MIDI Musical Instrument Digital Interface

MVP Minimum viable product

UE4 Unreal Engine 4

Chapter 1: Adapting Risers in Real-Time

Adapting parameters of audio in real-time, instead of playing cutup pieces of the audio based on game state and events, could be a practical and efficient game audio design method. This research revolves around adapting riser audio effects for a specific game test case, using the research question: What are the essential parameters to control a riser audio effect in a real-time video game, and how can they be implemented into Unreal Engine based on the game's state and events? A prototype tool for audio designers is developed and tested with industry professionals. Time and energy can be saved by using adaptive risers instead of other currently standard nonlinear audio development techniques. The research aims to contribute to further emancipating audio within game development by researching how to efficiently adapt a linear sound's parameters in real-time.

1.1 Linear Audio for Nonlinear Games

Colin Walder, Code Lead Audio & Localisation at CD PROJEKT RED, states: "By making a connection between the music and the game action we can effectively tell our audience what to feel"[1]. There is a discrepancy in audio being inherently linear and games being nonlinear. Audio is inherently linear because it only exists over time, it cannot be paused. Because of this discrepancy, certain issues arise in audio design and the audio design workflow. Some examples of these issues are industry standard software that is not optimised for creating nonlinear audio, visual and interactive context not being available while developing the audio, the limitations for a composer using standard nonlinear audio design methods, and a disconnect between the game's timeline and the musical timeline or grid. To tackle some of these issues, different middleware solutions exist. However, on top of not solving all the stated issues, middleware programs cause some issues of their own, such as requiring its users to switch back and forth between a DAW, a middleware solution, and a game engine before being able to test a sound within its context. Standard adaptive composition techniques divide music into different horizontal and vertical slices, to play based on game data. To become more adaptive, these slices need to be shorter, on top of which they need to be able to layer and transition with even more slices. Therefore, to create a highly adaptive system, the composer and their music will face escalating restrictions. There also is a disconnect

[1] Walder, C. (2019). Synchronizing Action-Based Gameplay to Music, in G. Somberg (Ed.) Game Audio Programming 2 (332 - 347). Taylor & Francis Group

between the musical grid and the game's timeline based on event and state. The slices of audio follow a musical grid based on tempo and measure and audio can only adapt at a point that is musically viable. Non-musical audio is also cut up into numerous different elements when required to be more adaptive. These challenges are inherent to the linear nature of sample-based audio being connected to real-time gameplay and will influence a player's experience of the game. Furthermore, a long-lasting dynamic building up sound can be difficult to implement. This study uses a specific case for the rising audio in a real-time game situation to test the implementation of a dynamic riser effect. The is to solve some of the issues that emerge from sample-based audio, by creating a system for dynamic audio, and to demonstrate this approach's validity with a specific test case.

1.2 Risers and a Horror Game Test Case

The type of audio examined are riser audio effects. In an online course, composer Mikael Baggström defines risers as:

> 'Riser Effects' is the term for the kind of sounds that rise in pitch over time. Sounds that continuously increases the energy and tension build up. However, even though the general direction is going upwards, you can add rhythm or internal movement into the upwards slide to make it more unique. You can also shape the overall curve of the riser over time.[2]

Risers are an interesting type of sound to investigate because their most important timing is at the impact point at the end of the sound's build-up. More standard sounds, such as piano notes or gunshots, have a very immediate impact at their start. Risers build up to a certain point, which is the point that should be synchronised with a certain event. This also makes them difficult to use in an adaptive manner, as it is difficult to start a riser without knowing when the impact is going to be. Furthermore, risers can last for a longer amount of time and therefore there is room to adapt parameters based on game events and state. When sampling and CD technology came along, high-quality samples started being used in games and game music started to move to Hollywood-

[2] Baggström, M. (2019). *Sound Design – Create Riser FX for Transitions*. Retrieved November 2020, from https://www.skillshare.com/classes/Sound-Design-Create-Riser-FX-for-Transitions/1618082930

type orchestral music[3]. Currently, riser audio effects are prevalent in Hollywood audio, which may make them relevant for games as well. Furthermore, risers are an effective low CPU and memory cost solution to get a lot of auditory impact and are applicable to both sound design and music. Riser's broad set of use-cases makes them an interesting and relevant effect to research.

To further scope the research and limit external factors influencing the results, the adaptive risers are tested on a specific test level. The design of this level is discussed in Chapter 3.3.4. The level is a horror environment in which the player moves around freely. At a certain point in the game an event happens, which the riser must build up to. This test level has a limited number of external influences, is relatively simple to make, and the horror genre fits with the focus on tension of riser audio effects.

1.3 Layout

Review of the subject's theoretical background is discussed in Chapter 2 to investigate standard game audio developments practises and their challenges and obstructions, what has already been done to attempt to tackle these issues, and what riser audio effects generally consist of. Qualitative interviews with industry professionals about a prototype that has been developed to apply the method in practise are the research's method of data gathering. The methodology in Chapter 3 discusses this method. The results are summarised and discussed in Chapter 4. Finally, Chapter 5 provides a conclusion to the research question and lists relevant further research topics to be explored in the future.

[3] Böttcher, N., Serafin, S. (2014). *A Review of Interactive Sound in Computer Games: Can Sound Affect the Motoric Behaviour of a Player?* In K. Collins (Ed.), B. Kapralos (Ed.), H. Tessler (Ed.) The Oxford Handbook of Interactive Audio. Oxford University Press.

Chapter 2: Related work

An overview of the current standard approaches to game audio design and existing solutions serves to further map out the stated issues and form a foundation for the construction of a potential solution. Firstly, the common game audio design workflow and its integration of middleware is discussed. Information on middleware, as well as industry examples, serve to provide insight in standard methods and tools used. An inhouse software solution developed by Guerrilla Games that aims to improve the emancipation of audio in game development is highlighted. Hereafter, common dynamic composition techniques are examined. Following this, procedural game audio is discussed. Highlights of well received procedural game audio systems as well as less successful systems are analysed. Common audio tools used to produce risers are then examined to map out the different elements they can consist of, their different uses, and to help further define the sound effect. Hereafter, the auditory illusion called the Shepard scale is examined. The Shepard scale is a potential solution to making risers go on for an indefinite amount of time. Finally, tools usability and UX testing is discussed to provide insight in a method of developing and testing a tool to answer a research question. As little sources are available on testing the exact type of tool this research aims to develop, general examples are analysed. The discussed related work is applied to form a methodology in Chapter 3.

There are a few ethical concerns to be addressed. As not much scientific research has been done in the field, most information has been gathered from presentations and articles by industry professionals that do not scientifically substantiate their statements. To contain the literature research, sources are selected based on overall quality and on how much they differ from each other. Industry examples from different backgrounds are analysed. However, there is a risk that the selection of discussed sources is one-sided. In Chapter 2.4, audio plugins used to design riser type of effects are discussed. This is a handpicked selection of software and every audio designer's approach to designing these effects can differ. Furthermore, I am an audio designer and programmer myself and therefore have a coloured opinion on the subject matter in general.

2.1 The Standard Game Audio Workflow and Its Use of Middleware Solutions

Böttcher and Serafin[4] describe the use of middleware:

> Today, most companies use audio middleware solutions to deal with the sound in the game. Audio middleware is an interface between software, whereby the interface design is intended to reduce programming requirements for the sound designers, while allowing for connectivity with the game engine. By using audio middleware, the sound designer can link sound to different game objects, such as specific areas, variables, or event triggers.[5]

They state middleware is often developed with an easy-to-use graphical interface optimised for the sound designer. This allows sound designers to implement more of the audio themselves and therefore saves a programmer's time. They list two standard middleware solutions: FMOD[6] and Wwise[7]. Another widely used middleware that is not mentioned is CRI Middleware[8]. It should be noted that most audio middleware programs allow for third-party plugins. Therefore, their functionalities can be extended on and adapted based on what a game requires. Lead audio programmer Stephane Beauchemin[9], explains about middleware: "While a customized audio engine certainly can provide value for the right game, audio middleware engines are great tools for audio programmers, and prevent us having to reinvent the wheel each time we start on a new project."[10] . They state that often a programmer is required to implement the middleware and add to the middleware's features. Middleware provides a time-efficient solution to implementing game audio for sound designers as well as audio programmers.

---

[4] Böttcher, N., Serafin, S. (2014). *A Review of Interactive Sound in Computer Games: Can Sound Affect the Motoric Behaviour of a Player?* In K. Collins (Ed.), B. Kapralos (Ed.), H. Tessler (Ed.) The Oxford Handbook of Interactive Audio. Oxford University Press.
[5] Böttcher, Serafin (2014), para. 26.3
[6] FMOD (middleware), Firelight Technologies (1995)
[7] Wwise (middleware), Audiokinetic (2006)
[8] CRI Middleware (middleware), CRI Middleware Co (1983)
[9] Beauchemin, S. (2019). *A Rare Breed*. In G. Somberg (Ed.) Game Audio Programming 2 (33 - 41). Taylor & Francis Group
[10] Beauchemin (2019), para. 2.1

In their talk about the real-time audio processing in NieR: Automata[11], Shuji Kohata[12] discusses developed audio tools and how they improved the audio experience. Kohata used Wwise to develop plugins and implement the game audio. They state real-time audio has a powerful influence on a game's interactivity, because immediately receiving audio feedback can be paramount for the immersion of a game. Audio expressivity can be maximised by having sound designers and developers venture into each other's territories. Providing sound designers with real-time tools, improves their ability to communicate audio to the listener in a more direct way.

In a talk on the audio system of Killzone: Shadow Fall[13], lead sound designer Anton Woldhek and audio programmer Andreas Varga discuss their motivations for building an integrated audio system for Guerrilla's Decima engine[14]. Some of their initial goals for this system where to emancipate sound, instantly be able to hear work in game, have an extendable system with reusable components, and to have a high run-time performance. Guerrilla developed a system that allows for audio to be tested within a game in a matter of seconds, during run-time, which reduces idea to game time and removes the programmer from the creative loop. This addresses a stated issue in the time and effort it takes to implement audio in a game using the common middleware-based approach. Emancipation of audio could translate to a larger community of users and therefore more feedback and understanding of the role of audio across more disciplines. Another motivation for developing this system was risk aversion. There is more room to fail because a lot of time is saved, which allows for more experimentation and more time to fine tune the audio. A node-based system provides an easy-to-use graphical interface and can easily be extended or modified. This modular approach is also resistant to quick advances in gaming technology. The speakers argue they were able to develop this system fairly quickly, can easily add to it, that it improves the quality as well as quantity of the audio, and that it saves time and energy in the long run.

[11] NieR: Automata (video game), Platinum Games (2017)
[12] Kohata, Shuji (2014). *An Interactive Sound Dystopia: Real-Time Audio Processing in NieR: Automata*. Retrieved October 2020, from https://www.youtube.com/watch?v=BrUQdd96qzk&t=28s
[13] Killzone Shadow Fall (video game), Guerrilla Games (2013)
[14] Varga, A., Woldhek, A. (2014). *The Next-Gen Dynamic Sound System of Killzone Shadow Fall*. Retrieved October 2020, from https://www.gdcvault.com/play/1020559/The-Next-Gen-Dynamic-Sound

2.2 Common Dynamic Audio Design Techniques

A common method of subdividing music used in game audio design, is to divide music into vertical and horizontal elements (Figure 2.1) with reorchestration and resequencing. Charlie Huguenard, Audio Software Engineer at Facebook, states:

> Currently, the most popular methods of composing music for games are the two provided by many audio middleware engines right out of the box: 'horizontal' and 'vertical' composition. […] Horizontally composed music plays one piece of music at a time. When the state of the game changes, a new piece of music is swapped in, usually with a transition stinger in between. Vertically composed music fades in and out a collection of stems—groups of instruments mixed to a single audio file—in response to game state changes. Both rely on pre-recorded pieces of music and offer handy ways to vary a game's score without sacrificing much of the composer's control over the mix and arrangement.[15]

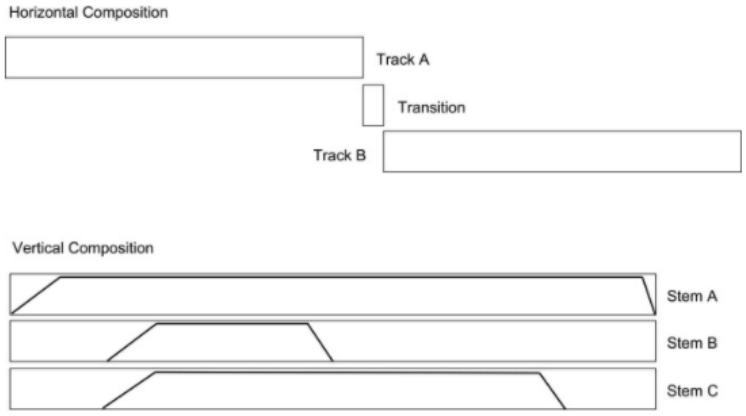


Figure 2.1 An overview of horizontal and vertical composition[16]

By applying different methods of reorchestration and resequencing, audio systems can be designed based on their required adaptiveness. New possibilities are generated by dividing the audio in stems and/or layers and applying a ruleset based on game data to what needs to be played and when. This reduces the music's repetitiveness, on top of which fewer audio assets are required, and storage space can be saved. Using these adaptive composition techniques, audio can be linked to game events while still complying to an audio-based grid. A downside, as discussed in the introduction, is that

[15] Huguenard, C. (2019). *Note Based Music Systems*. In G. Somberg (Ed.) Game Audio Programming 2 (307 - 331). Taylor & Francis Group. P. 308
[16] Huguenard (2019), p. 309

this audio grid adheres to a separate timeline from the game action because the audio must adhere to tempo and measure.

Adapting audio based on game state and events can help to immerse the audio within its interactive world as well as allow for the use of music as a gameplay element. Chris Christodoulou, composer for the Risk of Rain[17] franchise, implements this technique[18]. In Risk of Rain 2, the player travels through levels by finding and activating a teleporter. This teleporter emits a vertical audio layer to the music that increases in amplitude when the player gets closer to it. The vertical layer adds to the immersion by being used as a gameplay mechanic, as it can be used to help accomplish an objective.

The shorter the audio fragments, the faster the audio can adapt. When using reorchestration and resequencing, increasing audio's adaptability also limits the composer as musical fragments need to be shorter. Similarly, the more different combinations or transitions required, the more the composer's options are limited. In a talk about the soundtrack for DOOM[19], Mick Gordon argues dynamic music can get in the way of the flow of a game:

> Pretty much every battle in DOOM is a linear path. You enter the arena, the demons come out, your role is to kill all the demons and then you move forward. [...] We chose not to do too much crazy dynamic music stuff within that structure, because it just seemed to get in the way of the groove. [20]

It could be argued that the dynamic adaption could get in the way of 'the groove', because it can be difficult to make dynamic music responsive enough to adapt to the fast-paced shooter that is DOOM, without limiting the musical possibilities too much. Kubatko, a composer for mobile games such as Reaper: Tale of a Pale Swordsman[21], does not apply adaptive systems. He states people do not play mobile games for as long as other games and that music is often turned off or other music is played over the game. Standard adaptive audio systems can limit the composer and decrease player experience.

---

[17] Risk of Rain (video game franchise), Hopoo Games (2013 - 2020)
[18] Koning, S. de (2019). *Composing for Games as a Film Composer*. Retrieved October 2020, from https:// sdkoning.com/PF/ComposingforGamesasaFilmComposer.html
[19] Doom (video game), id Software, Bethesda Game Studios Austin (2016)
[20] Gordon, M. (2017). *DOOM: Behind the Music*. Retrieved October 2020, from https://www.youtube.com/watch?v=U4FNBMZsqrY. (49:24)
[21] Reaper: Tale of a Pale Swordsman (video game), Hexage (2013)

Game music composer Michael Sweet mentions a Stinger-Based approach[22]. This approach uses audio stingers that are directly linked to game events and therefore do not have to account for an audio-based grid. A difficulty with this approach is that stingers need to be able to layer over one another when triggered in quick succession, as well as the audio going silent when the stingers are not triggered by player action.

2.3 Procedural Audio in Games

The term 'procedural audio' is often used to refer to hyper adaptive audio systems that generate audio on a note-by-note basis in real-time. However, the term can be used for any audio that adapts based on external factors: "Instances of procedural music in games are usually only to modify or adapt the composed music to avoid sounding repetitive when listened to many times."[23]. A distinction can be made between audio with a limited set of possibilities that cannot adapt as quickly, and audio that can adapt almost instantaneously because it is being generated in real-time. In this paper, the more standard adaptive audio techniques are addressed as adaptive or nonlinear and hyper adaptive audio systems are referred to as real-time or procedural. This chapter discusses the hyper adaptive procedural systems.

By applying procedural audio techniques, memory can be saved, and sound designers can obtain new tools and ideas for applying les linear and more dynamic sound design[24]. In a talk about the sound of No Man's Sky[25], Paul Weird discusses his problem solving in developing an audio system for a large-scale game with a lot of procedurally generated environment[26]. "No Mans sky is a procedurally driven science fiction action-adventure game. It crosses several different genres. [...] There is no real story structure to it, it is very open ended."[27]. Instead of receiving specific events from the game to trigger audio,

[22] Sweet, M. (2016). *Top 6 Adaptive Music Techniques in Games – Pros and Cons*. Retrieved April 2020, from https://www.designingmusicnow.com/2016/06/13/advantages-disadvantages-common-interactive-music-techniques-used-video-games/
[23] Adam, T. (2014). Procedural Music Generation and Adaptation Based on Game State. The Faculty of California Polytechnic State University
[24] Böttcher, N., Serafin, S. (2014). *A Review of Interactive Sound in Computer Games: Can Sound Affect the Motoric Behaviour of a Player?* In K. Collins (Ed.), B. Kapralos (Ed.), H. Tessler (Ed.) The Oxford Handbook of Interactive Audio. Oxford University Press
[25] No Man's Sky (video game), Hello Games (2016)
[26] Weir, P. (2017). *The Sound of No Man's Sky*. Retrieved October 2020, from https://www.youtube.com/watch?v=zKJ_XuQjjiw
[27] Weir (2017), 01:27

states and variables such as the amount and type of trees surrounding the player are used to influence the audio.

Weird defines procedural generation as: "Algorithmically created content or gameplay, including graphics or sound"[28]. Procedural audio is defined by Weird as: "The creation of sound in real-time, using synthesis techniques such as physical modelling, with deep links into game systems."[29] They deem procedural audio useful when there is an issue of scale and to avoid repetition. Difficulties with procedural content are also stated:

> Some of the difficulties of procedural content, is to give the sense of this kind of meaningfulness. Like it feels like it is handcrafted. […] one of my big issues is that if you have procedural audio, the perception of it has to be as good as traditional audio. It is no good if you compromise. 'Well, its procedural, but it sounds a bit shit, but it does not matter because it is procedural.' That is not acceptable. As a sound designer I cannot accept that.[30]

Another issue mentioned is that variety does not get perceived as much when everything constantly changes. A solution for this Weird provides is to control what can be generated and what cannot. More difficulties with procedural audio are that it takes time and money to make and that an audio designer needs to be able to communicate well with a programmer.

Weird also touches on a problem they find with currently common procedural audio generation: "It is almost like, it is treating sounds like a function, rather than a creative emotive element. […] To have a wind that is essentially filtered white noise. I am doing them a disservice, but you get my point, it is like saying: 'okay there is wind. We have solved that.'" [31]. They talk about conveying emotion with sound design and that procedural audio can often lack emotion. Böttcher and Serafin[32] also mention sound design often having an 'added value'. For example, footsteps of a villain should sound evil. For generating the voices of aliens, the team created a Wwise plugin synthesis tool

[28] Weir (2017), 18:21
[29] Weir (2017), 28:18
[30] Weir (2017), 19:20
[31] Weir (2017), 25:58
[32] Böttcher, N., Serafin, S. (2014). *A Review of Interactive Sound in Computer Games: Can Sound Affect the Motoric Behaviour of a Player?* In K. Collins (Ed.), B. Kapralos (Ed.), H. Tessler (Ed.) The Oxford Handbook of Interactive Audio. Oxford University Press

that analyses the audio designer's performances. The 'added value' or 'human element' is added by analysing the performance of Weird on the inhouse made instrument.

Kent Jolly discusses why and how real-time procedural audio techniques are applied to SPORE[33], in Procedural Music in SPORE[34]. SPORE is a game with a lot of procedural generation over different elements of the game. However, initially the use of procedural audio was dismissed because it was thought that it would be too CPU expensive. Jolly also states that the sampled audio often used for these techniques can sound unemotive: "Procedural music can sound very MIDI"[35]. Because the procedural music would only need to be implemented in less CPU expensive parts of the game the first problem was solved. The 'MIDI sound' of procedural music was addressed by using very short types of sounds. The system links the visual programming language Pure Data[36] to the game engine and generates seeds for random patterns that reseed every eight counts. Basic composition techniques such as adding or removing octaves or adding a drop after a set number of measures, are used to make sure the audio stays interesting over a longer amount of time. Different sound palettes were implemented for different parameters. For example, when a more predatory part is added to a creature, the music shifts to more aggressive types of sounds. The procedural music is also used as a sound design element. For example, when adding a part to the creature, a short melody plays. The audio is meant to get the player in a creative state. To prevent the audio from becoming distracting, the audio will get 'simpler' and less in your face after a certain amount of time has passed. Even though the game and its audio system were well received, the system was not used as much after SPORE. It could be argued this is due to the innovation in game and sampling technology to allow for more orchestral-based music.

In Mini Metro[37], audio elements are generated in real-time based on how the player designs their metro lines (Figure 2.3). The game's internal clock is used to sequence the audio[38]. Freeland states a presentation on this system at GDC to be about: "How we avoided using looping music tracks completely, by using sequential sets of data to

[33] Spore (Video Game), Electronic Arts (2008)
[34] Jolly, K., McLeran, A. (2008). *Procedural Music in Spore*. Retrieved October 2020, from https://www.gdcvault.com/play/323/Procedural-Music-in
[35] Jolly, McLeran (2008), 06:05
[36] Pure Data (Programming Language), Modified BSD (1996)
[37] Mini Metro (Video Game), Dinosaur Polo Club (2014)
[38] Freeland, R. (2014). *Serialism & Sonification in Mini Metro*. Retrieved October 2020, from https://www.youtube.com/watch?v=FgV4hSfsl00&t=37s

generate music and sound."[39]. Freeland also states: "Immediate feedback is often reserved for sound design and not music. [...] Immediate feedback in music can feel forced. If not handled correctly, it can feel ham-fisted."[40]

Figure 2.2 Screenshot of Mini Metro

The game applies Serialism, a composition technique that uses sequential sets of data on different parameters working together to generate music. Game data is combined with externally authored data, which consists of a ruleset that Freeland applies on their designed sounds. This addresses the risk of procedural music lacking the feeling of being 'handcrafted'. The motivation for using this real-time system was:

> Because all these game objects have sounds that trigger in a musical way, by using a shared rhythmic language that is cognisant of the game clock and using game data to further tie them to what is actually happening in the game, things start to feel communal and unified.[41]

Just as with SPORE, shorter responsive sounds were used. Because the musical elements are also used as the sound effects, it could be stated that the procedural music system in this game is made by using music as sound design. Musical elements are derived from different game elements. Figure 2.3 depicts how the map of Mini Metro can be translated into a musical grid. The clock of the game and objects within the game are used as a grid to trigger sounds that is sequenced through by moving objects within the game.

[39] Freeland (2014), 0:18
[40] Freeland (2014), 11:19
[41] Freeland (2014), 12:58

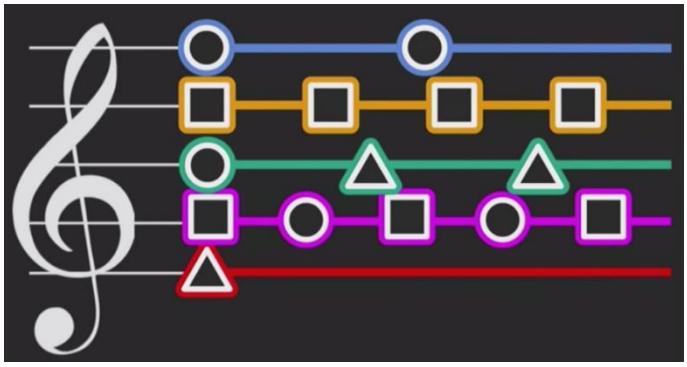


Figure 2.3 The metro lines of mini metro visualised as sheet music[42]

One of the most recent implementations of procedural music in a AAA game is the real-time procedural percussion system developed by Intelligent Music Systems[43] applied to Rise of the Tomb Raider[44], described in Real-time Procedural Percussion Scoring in Tomb Raider's Stealth Combat[45]. Speaker Tahouri states that the complex nonlinear combat system for this game demanded highly dynamic and adaptive music. The percussion system applies a data driven system which analyses composed MIDI tracks to generate variations in real-time. By analysing externally authored MIDI data, the music can be provided with a certain character and again the issue of procedural music not sounding 'handcrafted' is addressed. Game parameters can be linked to what audio needs to be played and its required intensity. On their website, Intelligent Music Systems state that[46]: "Procedural scoring opens up a world of new musical possibilities. Game music can change as quickly as needed to keep up with the most complex game action."[47]. They explain about their system that:

> The Dynamic Percussion System runs as a DSP unit in your audio engine (supported by FMOD and Wwise) with negligible CPU footprint. Instead of stacking up audio percussion tracks, the system only needs one channel. It stays synchronized with traditional looped audio in your audio engine. So, you can record your other stems fand loop them in your system while the percussion plays along procedurally.[48]

---

[42] Freeland (2014), 03:25
[43] Intelligent Music Systems (software developer) 2016
[44] Rise of the Tomb Raider (video game), Crystal Dynamics (2015)
[45] Lamperski, P., Tahouri, B. (2016). *Real-time Procedural Percussion Scoring in 'Tomb Raider's' Stealth Combat*. Retrieved October 2020, from https://www.gdcvault.com/play/1023215/Real-time-Procedural-Percussion-Scoring
[46] Intelligent Music Systems (2016). *Introducing the Dynamic Percussion System Powering Music for Rise of the Tomb Raider*. Retrieved November 2020, from http://www.intelligentmusicsystems.com/
[47] Intelligent Music Systems (2016), para. 3
[48] Intelligent Music Systems (2016), para. 4

AUD.js[49] is a Java-based application for web games that can quickly be implemented to generate music based on the game generation and adaption. The system uses standardised western composition techniques such as counter point to generate and adapt 8-bit style music in real-time. The writer states that the adaptiveness did not necessarily improve the player experience.

> We conducted A/B tests comparing static music, both composed and computer-generated, to dynamically adapting music. We find that AUD.js provides reasonably effective music for games, but that adaptiveness of the music does not necessarily improve player experience over composed music.[50]

A reason for the experience not necessarily improving could be that no externally authored data was used, such as was the case with the other discussed examples.

Lopes, Liapis and Yannakakis investigated the automated generation of levels and soundscapes with the development of Sonancia, by comparing two types of 'tension curves'[51]. The player's objective in Sonancia is to find a certain object by walking through generated rooms. Tension is created by monsters located in some of the rooms.

> Tension is visualized as a tension curve, allowing designers to define the intended rise and fall of tension in the level (DTC).[52]

The system uses two types of 'tension curves' to measure the intended tension and the perceived tension.

> Sonancia allows designers to define the flow of relaxation and tension through the generated level. To do this, the system requires two tension curves: the desired (designer- specified) tension curve (DTC) and the actual (level-based) tension curve (LTC).[53]

[49] Adam, T. (2014). *Procedural Music Generation and Adaptation Based on Game State.* The Faculty of California Polytechnic State University
[50] Adam (2014), Abstract
[51] Lopes, P., Liapis, A., Yannakakis, G. (2015). *Targeting Horror via Level and Soundscape Generation.* Institute of Digital Games, University of Malta. Received November 2020, from https://www.semanticscholar.org/paper/Targeting-Horror-via-Level-and-Soundscape-Lopes-Liapis/8131268f835035850cc997df45949c9e940327e0
[52] Lopes, Liapis, Yannakakis (2015), p.39
[53] Lopes, Liapis, Yannakakis (2015), p.37

Instruments are selected semi-randomly. Sounds are then placed throughout the levels based on the suspense curve and the provided tension value per sound. Comparing a desired tension with the measured tension provided insight in how well the system performs in matching the desired tension.

The more well received discussed procedural audio systems all use relatively short sounds, to address the risk of longer samples sounding to 'MIDI', on top of which longer samples would have to be modified in real-time. To prevent audio from getting distracting or disrupting, SPORE decreases audio intensity over time. Decreasing intensity over time could be a method for making riser effects go on for longer amounts of time. A 'lack of meaningfulness', or the lack of a 'human element' or 'added value' is generally seen as a risk when using procedural audio. To address this, externally authored data is often implemented. Mini Metro maps music directly to game action by using the internal clock and objects within the game. Procedural audio can be used effectively to save memory, create a feeling unified feeling, address scale issues, and avoid repetition.

2.4 Riser Audio Tools

To map out elements risers can consist of, their different uses, and to help define them, common audio tools used to produce them are examined and reoccurring features and notable differences are highlighted. Risers generally consist of two parts: the build-up and builddown, or more well known as the attack and release. In between the attack and release is the impact, which is the sound that the attack builds up to. Most of the discussed tools provide methods of designing a sound for the impact as well. The discussed tools have been selected based on their popularity and if they offer certain features their alternatives do not.

Figure 2.4 Screenshot of Whoosh Designer

Whoosh Designer[54] (Figure 2.5) divides whooshes into three dynamic elements: the attack, the peak, and the decay. The peak is another term for the previously described impact. Every element consists of two samples that can be selected from a list of provided options. Arrow buttons provide different panning modulation options. The sound's duration can either be an 8th note, a 4th note, an entire measure, or two measures. Whoosh Designer offers a quick and relatively simple way to design whooshes, without offering as much control over the output as other discussed tools.

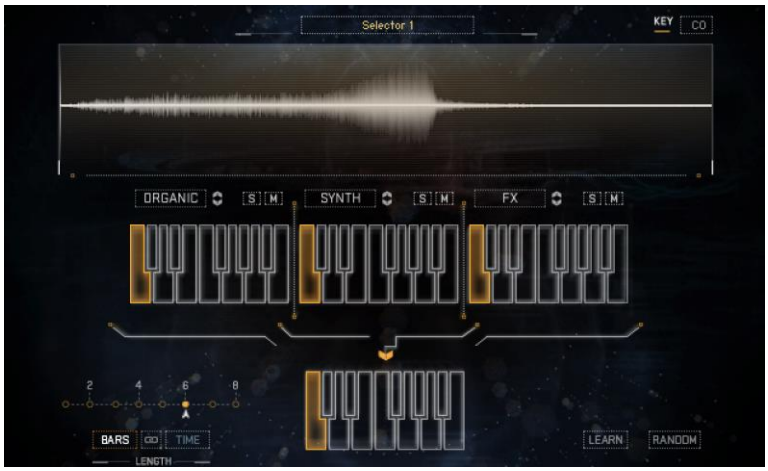


Figure 2.5 Screenshot of Gravity's Riser window

Gravity[55] is a virtual instrument with a dynamic collection of modern scoring tools. The plugin distinguishes rising type audio effects into risers and whooshes. Whooshes have a

[54] Whoosh Designer (audio plugin), Zero-G (2014)
[55] Gravity (audio plugin), Heavyocity (2016)

relatively shorter build-up (up to one measure) and are often a-tonal and noise-based. Risers are longer lasting (up to 30 seconds), subdivided into three different elements (Figure 2.6): Synthetic, FX and Organic Hybrid. Synthetic risers consist of recordings of synthesised sounds, FX risers are edited audio recordings, and Organic Hybrids are edited organic sounds such as recordings of a selection of string instruments. All types of effects can be synchronised with the audio grid by linking the start of the sound to a full beat or measure. When this option is selected and a note is pressed, the riser only starts playing at the start of the next beat or measure. The duration of a riser can be set in either time in seconds (up to 30 seconds) or bars (up to eight bars). A step sequencer (Figure 2.7) is provided to modulate volume, panning, and pitch. This step sequencer can also be linked to the DAW's grid so that the modulation's rhythm can be synchronised with the rest of the audio.

Figure 2.6 Screenshot of Gravity's Modulation tab

In Tonsturm's Whoosh[56], four layers of samples can be selected. Every layer has its own gain slider, lowpass-filter, and highpass-filter. A fifth sample can be selected for the impact. The plugin can synchronise in time in seconds, as well as beats or measures. A Doppler effect with extensive controls over the speed and amount of the Doppler effect has also been implemented. The samples can be edited together as well as separately. By slightly offsetting the impact point for each sample randomly, variations can be generated every time a note is played.

[56] Whoosh (audio plugin), Tonsturm (2014)

Figure 2.7 Screenshot of Whoosh

Rise & Hit[57] applies an interface in which the dynamics of the element that the user is currently editing are always on screen. A sample, or for example the length of an effect, is visualised in the centre of the screen. At the top of the screen, the settings for the sound's duration are also always visible. The sound is divided into four elements. Every element has its own sample and mixing panel for effects and modulation. The duration of the sound can be edited in beats as well as time in seconds.

Figure 2.8 Screenshots of Rise & Hit

[57] Rise & Hit (audio plugin), Native instruments (2014)

Air Music's the Riser[58] is the only prevalent synthesis-based example. Instead of selecting samples, the user is provided with wavetable oscillators to synthesise the sound. The plugin is divided into seven main components: generators containing the oscillators, a filter, an AMP, an effects panel, an LFO modulation panel, decay, and overall stereo-width and volume. There are three generators. One for 'sweep', which is the main tonal sound, one for noise, and one for chord, which is a harmonic layer on top of the sweep. The plugin uses two-dimensional sliders consisting of a start and end point, in between which a line is drawn visualising the element's modulation. The separate decay component handles the release of the sound. Notable is the high amount of control that has been provided over the attack compared to the three sliders provided for the release.

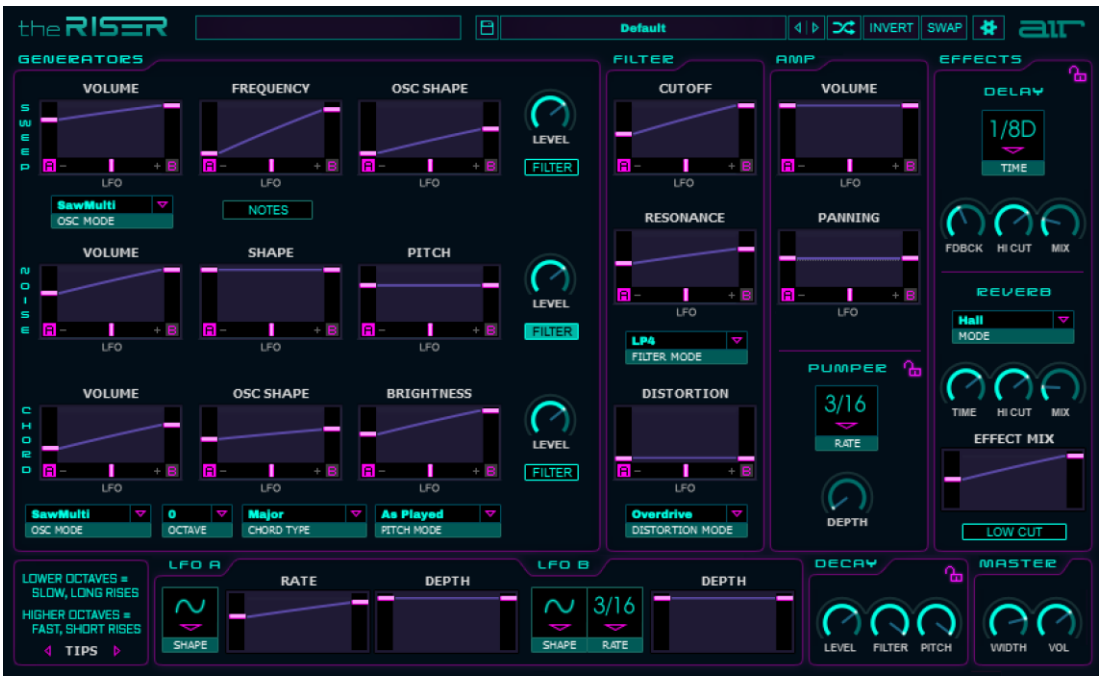


Figure 2.9 Screenshot of The Riser

The discussed plugins show a lot of similarities. Most of them provide the ability to design an output consisting of an attack and release, as well as a sound for the impact. The only plugin that did not provide any form of impact sound is The Riser, which also is the only synthesis-based plugin discussed. All the plugins allowed for the sound to consist out of multiple layers. Every analysed plugin's description or manual mentions the plugin to be created for designing cinematic type sounds. This supports the

[58] The Riser (audio plugin), Air (2014)

statement in the introduction that risers are prevalent in Hollywood's film industry. Most extensive control is provided over the volume and pitch of the risers. Furthermore, panning and filtering are often emphasised. Other reoccurring functionalities are allowing the user to link the timing to the audio grid, multiple options for setting the duration, and a lot of modulation functionality.

2.5 The Shepard Scale

In a paper on the use of Shepard tones in recent films[59], Eleonora Rapan examines Shepard tones and their use in linear visual media and pop music. Important to note is that this paper does not mention the distinction between a Shepard tone and the auditory illusion created by modulating the amplitude of layered Shepard tones, called the Shepard scale[60]. The writer uses the term Shepard tone to refer to both the auditory illusion created in a Shepard scale and singular Shepard tones. The Shepard Scale is a prime example of risers and their utility, as well as a potential solution for keeping risers going indefinitely. "In December 1964, Roger Shepard published his article, 'Circularity in Judgments of Relative Pitch' in the Journal of the Acoustical Society of America. This idea lead to the development of the so-called 'Shepard scale' or 'Shepard tone'."[61]. It has been used in popular music such as in Meddle[62] by Pink Floyd and in the sound design and/or soundtracks of films such The Prestige[63], The Dark Knight[64], and most recently Dunkirk[65]. Notable, is that the script of Dunkirk was written for the use of the Shepard scale. Comparable with the way a Shepard scale works, the three timelines of the story have been written in a way to provoke a continual feeling of increasing intensity.

> By managing the amplitudes of each octave in a Gaussian bell-shaped envelope, the frequencies in the extremes are not perceived: frequencies at the lower end enter softly, and the upper ones disappear in a similar way. This means that the

[59] Rapan, E. (2018). *Shepard Tones and Production of Meaning in Recent Films: Lucrecia Martel's Zama and Christopher Nolan's Dunkirk*. In D. Power (Ed.), S. Deutsch (Ed.), LK. Sider (Ed.) The New Soundtrack (135 - 144). Edinburgh University Press. Retrieved November 2020, from https://www.researchgate.net/publication/327410211_Shepard_Tones_and_Production_of_Meaning_in_Recent_Films_Lucrecia_Martel's_Zama_and_Christopher_Nolan's_Dunkirk
[60] Vernooij, E., Orcalli, A., Fabbro, F., Crescentini, C., (2016). *Listening to the Shepard-Risset Glissando: the Relationship between Emotional Response, Disruption of Equilibrium, and Personality.* Frontiers in Psychology. Retrieved November 2020, from https://www.frontiersin.org/article/10.3389/fpsyg.2016.00300
[61] Rapan (2018), p.135
[62] Meddle (song), Pink Floyd, Harvest Records (1971)
[63] The Prestige (movie), Christopher Nolan (2007)
[64] The Dark Knight (movie), Christopher Nolan (2008)
[65] Dunkirk (movie), Christopher Nolan (2017)

> middle frequencies are heard fully. These frequencies are very clear and always moving up or down, but the ones at the last octave slowly disappear. This is done in order to prevent our ears being aware of arriving at a limit, but they do perceive the resulting endless rise or fall.[66]

About the scale's use in Zama[67], Rapan states that: "It is likely that the idea of working with the Shepard tone as a sonic element carries us and the character we are watching into the Shepard tone's timing, a time where we don't perceive a beginning or end." This implies the use of a Shepard scale not only to increase intensity, but also to create a perception of time without beginning nor end.

2.6 Tools Development and Testing

Dzmitry Aliakseyeu[68] discussed different forms of research based on behavioural and attitudinal methods, and qualitative and quantitative methods during an online lecture. Qualitative research can be used to build a theory or model, but not to validate a theory. Quantitative research provides concrete data but does not provide an explanation as to why a specific outcome has been reached. Attitudinal methods analyse what a person says, while behavioural methods analyse the behaviour of subjects. Behavioural methods can provide more concrete data as a person's subjectivity does not have to be considered, but it is more difficult to find the reasoning behind the behaviour. Aliakseyeu also argues that when designing product nowadays, the product usually fits into a certain ecosystem. For example, when designing an audio plugin, existing audio software and common workflows should be considered. When designing a research method, it is important to keep this ecosystem in mind as well as to decide if concrete data should be collected or a theory should be build.

Dmitry Gaiduk[69] states: "Usability testing helps to improve CRO by finding users' intents and wishes, making a decision on fixing some badly performing parts of a website or app and the developing of additional functionality" [70]. Not only should it be tested if the tool is something its target audience would want to use, but also if the developed tool itself is

[66] Rapan (2018), p.137
[67] Zama (movie), Lucrecia Martel (2017)
[68] Aliakseyeu, D. (November 2020). *User Experience Evaluation in Industry*. Guest lecture presented in Breda University of Applied Sciences, Breda
[69] Gaiduk, D. (2019). *How to Do Usability and UX Testing for Mobile App*. Extracted October 2020, from https://medium.com/uxreality-blog/how-to-do-usability-and-ux-testing-for-mobile-app-211f92f3cd6d
[70] Gaiduk (2019), para. 2

easy enough to use. Gaiduk states the difference between Usability and User Experience to be:

> Usability is the way of how a product can be used by users to reach specified goals. Usability testing is aimed to uncover how much the product (app, website) is easy to use, understandable, is it able to satisfy the user's needs effectively.
>
> User Experience is a user's perceptions of the product (app, website). User Experience testing includes measuring users' emotions, gaze movements, preferences, and all key details of behaviour during and after use the product.[71]

Test tasks should have clear objectives to obtain measurable results. The testers should be asked to perform certain tasks with the application. "A task, set within a usability test, should be precise, actionable and shouldn't contain any hints, which can facilitate its performance (and thus distort the results of the test by imposing certain logic of behaviour on users)." Using clear and measurable tasks, usability testing and user research can provide insight into the users' perception of a tool as well as how well the tool can be used to reach specified goals.

[71] Gaiduk (2019), para. 5-6

Chapter 3: Methodology

In the previous chapters, issues have been stated on the limitations off current game audio design methods and their software workflow, caused by the discrepancy between the linearity of audio and the nonlinearity of games. To attempt to address these issues and answer the research question, professional game audio designers are interviewed on a prototyped tool that attempts to address these issues for riser audio effects. The tool allows its users to design and adapt risers in real-time based on game data. This chapter firstly discusses the method of gathering data from industry professionals by providing them with a prototype. Aspects such as the construction of the interview, the data analysis method, expected results, potential risks, and ethical concerns are also discussed. Following this, an interview with a professional procedural tools developer at Ubisoft Paris serves to provide insight in common traps and good practices in procedural tools development. Then the tool's design and approach to adapting risers in real-time is discussed. The tool is first mocked using existing middleware to gain a better understanding of potential issues and required features, and to showcase the current difficulties when trying this research's approach to game audio using the currently standard available tools. After this, the development of the tool is discussed. Finally, a pilot study is done to test the interview's structure and questions, and collect initial data on the prototype to see if any features need to be revised or added before doing the rest of the interviews.

3.1 Research Method

Qualitative data will be gathered from interviews to answer the research question: What are the essential parameters to control a riser audio effect in a real-time video game, and how can they be implemented based on the game's state and events? This question can be subdivided in two sections:

- The first section is: 'What are the essential parameters to control a Riser audio effect in a real-time video game?' This covers what audio parameters need to be adapted and how they should be modulated.
- The second section is: 'and how can they be implemented based on the game's state and events?' This is about what game data should be used and how this data should be mapped to audio data in real-time.

Furthermore, the overall viability of the tool and method is of interest. The tool's advantages and disadvantages should provide insight in the viability of the proposed

approach. There may also be results that indicate if this procedural approach could be applied to other audio effects.

As stated by Aliakseyeu in Chapter 2.6, when developing a new tool, the surrounding ecosystem should be kept in mind. It is important to examine if the prototype fits into the current workflow and pipeline of audio designers. In the same chapter, Gaiduk mentions the importance of usability testing. Usability questions will be asked in the interviews to gather feedback on how well the tool facilitates the audio designer in the performance of their tasks, such as if the controls are clear in their function.

3.1.1 Qualitative Interviews with Game Audio Developers

Data will be gathered by interviewing professional audio developers and showcasing a prototyped tool that implements real-time procedural risers. Testing a proof of concept with experienced industry professionals reduces speculation and provides practical insights from professionals in the field[72]. Quantitative data gathering is waived because the perception and knowledge of audio of the general game audience can vary widely. Professional audio designers, unlike players, have a general understanding of audio and its practical uses in games. Audio designers should be aware of the audio's role in game development. Furthermore, proving that the method can enhance the experience for an audience without providing a method of applying real-time risers, would reduce the direct industry relevance of this research. Developing a tool allows the interviewees to experience the real-time audio design method instead of having to speculate on the approach. As the research is about exploration and understanding, in contrary to attempting to validate a theory, the results should provide an indication based on the opinions of professionals.

3.1.2 Interview Structure and Questions

Every interviewee will be questioned on the same set of topics. To contextualise the questions to the work and experience of interviewees, certain nuances of questions might differ. To be able to interview audio designers from around the world, the interviews will be held digitally. Because of this, it is not feasible to have the interviewees try out the software themselves. Furthermore, having interviewees try out

[72] Burls, A., Gray, D. Kogan, M. (2014). *Salutogenisis and coaching: Testing a proof of concept to develop a model for practitioners*. International Journal of Evidence Based Coaching and Mentoring, Oxford Brookes University. Retrieved June 2021, from https://search.informit.org/doi/abs/10.3316/informit.705249145757477

the software themselves, could shift their focus to the user experience of the specific software instead of the proposed method's viability. Therefore, the software will be showcased. Having interviewees test the tool themselves also could take considerably longer. The interview in its entirety should take about one hour. Asking for more time might result in less participants as well as too much data to analyse within the scope of this research.

After touching on privacy and consent, the participant will be asked background questions on their experience and approach to game audio. This serves to put their answers in perspective. The participant will also be made aware they are free to ask questions or make remarks at any moment. Then the research will be explained by describing the stated issues and the research's approach to addressing them, the workflow of the prototype, and the specific context of risers. Hereafter, an initial set of questions will be asked. It is relevant to note that the order of the questions might differ as it can be the case that the audio designer already touches on a topic naturally. As this is a qualitative and exploratory research, the order is not of importance to the results. The interviewee will be asked about their first impression of the research based on the explanation, and how they would currently approach a build-up type sound for the explained game test case. Also, their experience and opinions on procedural audio will be examined to again provide perspective on their professional approach, as well as to map out advantages and disadvantages of the procedural method. After this round of questions, the tool will be showcased, followed by a second round of questions. Again, the interviewee's first impression will be asked for. The interviewee will also be asked if there are any cases in the past where they would have liked to have been able to use the tool, if there are any changes they would make or features they would like to add, if the tool would fit in their current workflow, and if the tool's interface and workflow could be improved. The full interview script can be found in Appendix A.

3.1.3 Data Analysis

The gathered data will first be transcribed to then be analysed using thematic analysis. The analysis will be mostly inductive to not let it be influenced by expected results. Transcription software Otter [73] will be used to get initial automated transcriptions that will then be manually improved. After the interviews have been transcribed they will be analysed, and notable parts will be highlighted and assigned a code. The data coding tool

[73] Otter (speech to text tool), Otter.ai (2016)

NVivo[74] will be used for the coding. The codes will be analysed and combined into themes. An overview of these themes and their codes will be created and a code handbook that explains the themes and their codes will be made and held up to date as the groupings are iterated on. The connections between the themes will be analysed, stated, and visualised. Grouping the data based on different themes serves to provide a clear overview of the interviewee's answers and their relations to one another, to be interpreted to answer the research question.

3.1.4 Industry Relevance

The tool aims to improve the audio design workflow by not requiring its user to switch back and forth between different software. This could save time, and therefore improve both quality and creativity. Creating a way to implement risers in nonlinear games also brings the general advantages of risers. Designing a riser can take less time than for example using an orchestra to achieve the same auditory impact. Furthermore, time can be saved by having the tool handle certain things automatically without the user having to program it manually. Risers are a low-cost audio effect with intense impact and could therefore also save memory space. Also, risers are currently popular in Hollywood type movies and therefore there might be an interest in using and hearing them in games. Real-time procedural risers could also be a unique feature for a game. Finally, addressing existing obstructions in the audio design process for this specific sound, can lead to insight in tackling the stated issues for more types of audio and cases in the future.

3.1.5 Expected Results

It can be concluded from Chapter 2 that there is room for improvement in the game audio pipeline for the specific issues in adaptivity and workflow. A real-time procedural solution could be viable and there is a general interest in these types of tools. However, current methods are very established, and as there often is little time available in game development, interviewees might question if they would have the resources to implement it. Furthermore, the prototype is optimised for a specific sound. There might be concerns if it could be used for other types of games and sound effects as well. Chapter 2.4 concludes that risers often provide layering functionalities and extensive control over gain and pitch. The same parameters are expected to be prevalent for this tool as well. The most prevalent translation to game data is that the riser can go on for a

[74] NVivo (qualitative data analysis tool), QSR International (2020)

longer amount of time when the player is slow, or even walks the wrong way, while still sounding naturally. This can potentially be done by measuring the amount of action over time and decreasing parts of audio that depend on modulation more. Furthermore, as with the example of Spore in Chapter 2.3, the amount of time spend should also have an impact.

3.1.6 Risk Analysis

Risk	**Estimated Likelihood**	**Estimated Severity**	**Likelihood mitigation**	**Severity mitigation**
Not enough interested testers	Medium	High	Start reaching out early and to a high number of people	Hold interviews to a high-quality standard
Risers are too specific	Medium	Medium	Underline what is specific to risers and what can be applied to other types of audio as well	Underline what is specific to risers and what can be applied to other types of audio as well
The tool does not work	Low	High	Start development early and test thoroughly	Create a mocked version of the tool as a back-up
The exact or a similar tool is released	Medium	Medium	Keep the research somewhat private	Focus on the elements that make this tool unique
Data loss	Low	High	Back up often and in multiple locations	n/a
Personal circumstances	Medium	Medium	n/a	Taking potential unforeseen circumstances into account in the planning
Testers cancel or do not show up	Medium	Medium	Send a reminder message some time before the scheduled interview	Have a high number of scheduled interviews with a high-quality standard

3.1.7 Ethical Considerations

On top of the ethical concerns mentioned in the literature review, the method has a few ethical considerations to be mentioned. Interviewees should be made aware of their privacy rights, given the option to do the interview anonymously, and asked for their consent to record the interview, publish it, and use the gathered data in the thesis. The test scenario is in the horror genre, which deliberately raises the levels of tension in a player/developer. This could impact the stress level or make the interviewee feel uncomfortable in some way. I, as the creator of the software, am also the person doing the interview, and therefore potentially introducing bias into the responses and results. The research and the tool will be explained and showcased before asking certain questions, which might influence the answers of interviewees. The selection of testers should include people from different professional backgrounds, as well as cultural backgrounds, to improve inclusivity and get a wider perspective on the tool. The term 'master' is a common term used in audio production for multiple aspects. This can be a sensitive term and will be avoided where possible during development, in the documentation, and in the interview by the interviewer. The term 'main' will be used instead. Finally, this study is not requiring new hardware, it is using existing assets, so does not have a hardware carbon production footprint.

3.2 Interview on Procedural Tools Development

Twan de Graaf provided insight in his experience as technical artist and procedural tools developer at Ubisoft Paris during an online interview. An example of a project that De Graaf worked on is a tool used to generate an audio map. When updating large parts of the terrain, the audio had to be updated as well, which would take a lot of time to do manually. The tool translates game data, such as the distance to a certain object, to audio data. The audio designer takes the audio map and creates a rule system to trigger and adapt audio based on it.

On top of saving time by being able to generate output more quickly, de Graaf also mentions procedural tools can play a significant role in avoiding human error. This is an additional industry advantage of this thesis's tool in addition to the industry relevance mentioned in 3.1.5. Furthermore, De Graaf states that for a procedural tool's developer, optimising 'boring tasks' is just as interesting as optimising 'interesting tasks'.

It can be challenging to communicate about the advantages of procedural generation. A way to improve this, is to work closely with a single 'dedicated expert user'. When the dedicated user is satisfied with the tool, the tool can be provided to others that might benefit from its functionality. A risk here is that others are not always immediately as happy with it because they did not have the same 'growing up experience with the tool'. In contrary to making a tool for a single person, there are also advantages of making a tool for as many people as possible, as stated in Chapter 2.2. De Graaf states there is something to say for both approaches. Creating a tool for everyone restricts the tool' s complexity but reaching a larger audience at once can save time.

De Graaf discusses a few common traps in tools development. Firstly, he states to avoid developing all encapsulating tools: "Fixing one thing can break 3 other things." He also explains that it makes finding bugs more difficult. De Graaf's team is not against feature creep, but it should always be considered if the feature should not have its own tool. De Graaf also mentions that he generally keeps parameters in the background first: "It often turns out that you are not going to use 80% of them." If it turns out to be something that needs to be tweaked a lot, he exposes it by either creating an interface element or by creating a setting in a data file. When asked when a tool is finished, De Graaf responds: "Basically never. […] They are done until somebody asks for a new feature." Finally, when working with dedicated users, often not as much documentation is required. This can be a trap when a tool is exposed to more users later: "When more people start asking the same question, or even asking the same question twice. Then it is really necessary to start making documentation."

3.3 Tool Design

Based on the literature review, a design for the minimum viable product can be constructed. The potential uses for the tool are to build audio, implement audio, make audio a gameplay element, and the sonification of physical elements or events. Preferably, the tool should be made as modular as possible so that it can more easily be applied to other types of audio. In Chapter 2.3 the use of externally authored data when using procedural generation is often mentioned. For this tool, control over the sound and adaption of the riser should suffice as externally authored data. Below is an overview of the stated issues in game audio design and the tool's approach to tackling them.

Issue in game audio design	Approach to addressing the issue
The disconnect between the audio grid and the game action.	Adapting the samples themselves, instead of using separate grids or triggering audio on a note-by-note basis.
Designing adaptive longer more dynamic, or emotive sounds is difficult without knowing the desired course of tension beforehand. Composing longer pieces of music using standard nonlinear composition methods or longer notes using procedural music generation requires a trade-off between the amount of restriction for the adaptiveness and the amount of restriction for the composer.	Adapting a dynamic sound in real time using an algorithm based on audio parameters and game data.
The time straining process of switching back and forward between multiple software to design and implement audio, and the lack of direct context because of this process.	Design and edit the sound and quickly be able to test it within the game with the least number of steps and export/import time required as possible.

3.3.1 Risers

As concluded in Chapter 2.4, risers generally consist of three dynamic elements: the attack, the impact, and the release. Most emphasis is put on the attack as risers are often used to build up to something. The release serves to build the audio down again. The impact generally consists of static samples that do not require any modulation. The blue line in Figure 3.1 highlights the attack of the riser of which any parameter such as the amplitude, pitch, and panning can be adapted. The start and impact points should be triggered by the game. By modulating the audio data in between the start and impact, the riser can go on for a longer or shorter amount of time.

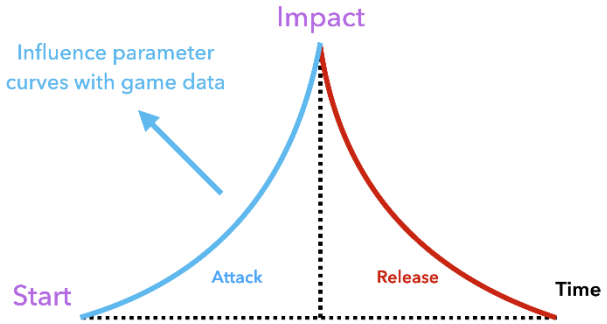


Figure 3.1 An overview of a riser and its adaption

3.3.2 Modulation

Risers are made by modulating parameters of audio to first build the audio up in intensity during its attack and then drop in intensity during its release. There are three main factors that are significant to control the modulation. First, there is the modulation curve (or envelope shape), which controls how the intensity of an element needs to be increased over time (Figure 3.2). Secondly, there is the seek speed which sets how fast a parameter can modulate. Finally, there is the range. The range of a pitch envelope can for example be one octave, which is a doubling in frequency. These three factors should be set based on the sound and the audio parameter that is modulated. For example, modulating the pitch of noise does not have much effect and a noise layer might need to build up more slowly than another layer.

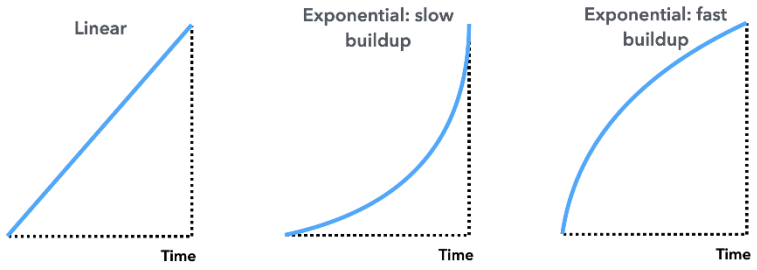


Figure 3.2 Examples of standard modulation curves

The most important parameters to modulate based on their auditory impact as well as on how much they occurred in the analysis of existing riser plugins in Chapter 2.4 are the amplitude and pitch. Examples of secondary parameters to modulate are panning, filter cut-off, and stereo-width.

3.3.3 Sound Handling

The plugins discussed in Chapter 2.4 either apply synthesis or use premade samples. Using live synthesis or granular synthesis is out of scope for this research as it generally is more work to develop and requires more processing. Premade samples can also not be used as the samples have a maximum length and the length of the adaptive risers is not predefined. The sound for the prototype will therefore be handled by using a looping system. Riser samples are deconstructed to static loop-able samples without any modulation (Figure 3.3). They are then modulated by the tool during playback to have them rise in intensity. The process of producing these sounds for the tool is discussed in Chapter 3.4.2.

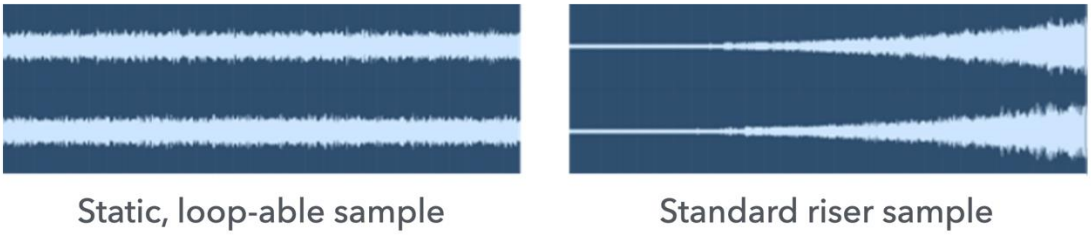


Figure 3.3 The waveform of a static sample (left) and a sample that rises in amplitude (right)

3.3.4 The Test Level

Because testing the tool with any type of gameplay mechanic is out of scope, the tool shall initially be tested and showcased using a specific test case or test level. To decide on the initial test level, different test levels have been designed and one has been selected. An overview of the designed levels can be found in Appendix B.

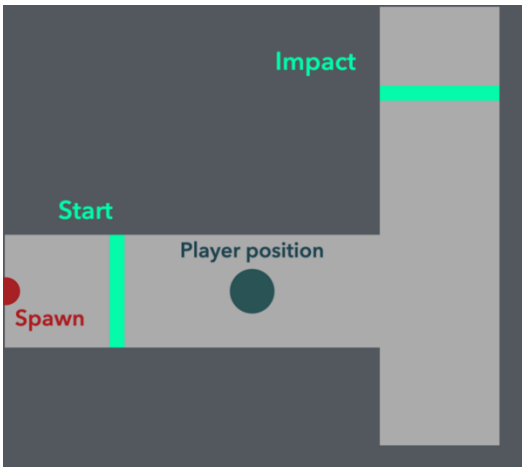


Figure 3.4 Overview of the designed test level

The case that was decided on is a walking simulator type horror game in which the player can walk around freely. At a certain point, the player shall pass a trigger that starts the riser, after which at a point further in the map, the impact will be triggered (Figure 3.4). The horror genre was selected because of its emphasis on tension which matches the emphasis on tension of riser audio effects. This case is relatively simple to make and has very little external factors that could influence the data such as enemy AI or a combat system.

As not putting any other sounds in the context game could feel unnatural and would not accurately represent how well the riser works within context, the other sounds in the level can have an influence on the experience. This should be taken into consideration

when measuring the results. Other factors in the game such as the environment and gameplay can also influence the results in a similar way. However, professional audio designers can be expected to have an idea of how much they are influenced by the other sounds and elements.

3.3.5 Game Adaption

To adapt parameters of the audio live based on game data, the riser will be subdivided into different layers that can be modulated separately. Three types of modulation, that are combined to reach the eventual output, have been designed. The first is called position modulation and is modulated by the position the riser needs to get to. For the designed test level this is the players distance to the impact point relative to the distance from the start to the impact point. The second is called action modulation and decreases the riser's intensity based on the amount of game action that is happening. For the test level this translates to the action modulation decreasing the riser's intensity when the player's position is not changing a lot. On top of decreasing the intensity the seek speed can also be decreased. The action modulation can be divided into two aspects: having the riser fall in intensity and reaching the lowest point of intensity where no more modulation is possible. During the lowest point of intensity, only minimal static sounds shall be played. A Shepard scale can be used here to still have the riser feel like it is building up. The third type of modulation is time modulation. In the example of SPORE in Chapter 2.3, the music gets less intense after a certain amount of time. As the test case does not provide a time frame, the player could walk back and forth endlessly. To prevent the riser from staying intense when this is the case, the amount of time spent should also decrease the riser's intensity.

3.4 Tool Development

The development of a prototype tool solution has been split in multiple sections. First the test level has been developed and the static sounds to use in the tool have been produced. Using these sounds the tool has been mocked FMOD Studio to start testing the sound and adaption early, experiment with features, and to gain insight in the current difficulties that arise when making procedural risers with existing tools. Using insights from the mock's development, the tool has been prototyped.

3.4.1 Test Level Case

Three potentially feasible approaches to developing the test level have been examined. The first is to use the open-source game Amnesia: The Dark Descent[75], the second is to use an asset pack, and the third is to use a game that I have previously worked on called Project Rookery[76] (Figure 3.5). A disadvantage of using an already existing game is that the interviewee might have set expectations because they have played or seen the game before. Another disadvantage is that Amnesia uses an engine that is no longer being used. Advantages are that there already is audio available for context, that it could save time, and that an already established game is used and therefore the game design has been proven and will not influence the results. The advantage of using an asset pack is that it provides complete control. However, asset packs often do not have audio integrated, context audio would have to be integrated manually. The main advantages of using Project Rookery are that it already contains sound design and music, and that I know my way around the project as I have worked on it before.

Figure 3.5 Screenshot of *Project Rookery*

Based on the listed advantages and disadvantages, the approach of using Amnesia was initially selected. However, an unsupported library would require a workaround that would take too much time to implement. Because of this, the second most preferred option, Project Rookery has been selected. Project Rookery is a first-person horror walking simulator set in Victorian London. Near the end of the game, a jump scare happens when suddenly a flock of birds breaks through a window. A riser can be linked to reach its impact on this event.

[75] Amnesia: The Dark Descent (video game), Frictional Games (2010)
[76] Dijk, J. van, Koning, S. de (2020). Project Rookery Trailer. Retrieved January 2021, from http://sdkoning.com/PF/RookeryTrailer.html

3.4.2 Sound Design

In Chapter 3.2.2 the method of handling source audio by deconstructing risers to static loops is explained. Based on the discussed riser tools in Chapter 2.4, the following classification of layers has been made: noise, pads, impact, Shepard scale, wavetable, and FX. FX is an extra layer consisting of static sound effects such as rumbling. These types of sounds can help to keep the riser going on for a longer amount of time as they can be played without modulation. The DAW Logic Pro X[77] and the synthesiser Massive X[78] have been used to create the sounds.

Figure 3.6 A modulation curve with as little attack as possible (left) and a more standard curve (right)

To make the sounds loop-able, they should not have any attack or release. Clicking sounds can occur when audio does not perfectly loop. A clicking sound is a brief audible tick sound caused by a speaker having to quickly jump to a certain point because the audio does not align. Figure 3.6 depicts the default amplitude envelope of Massive X to the right, and to the left a sound with as little attack and release as possible. Because Massive X's minimum attack and release are slightly audible, the output does not automatically loop perfectly. Furthermore, some of the designed sounds are not completely static and therefore do not loop seamlessly either. To address this, a simple sound design technique has been applied (Figure 3.7). The start of the samples has been cut off and pasted at their end using a crossfade that makes sure the two samples transition seamlessly. Because the new end of the sample has been taken from the start, the waveforms of the new end of the samples and their new start now perfectly align.

[77] Logic Pro X (digital audio workstation), Apple (2013)
[78] Massive X (digital synthesiser), Native Instruments (2020)

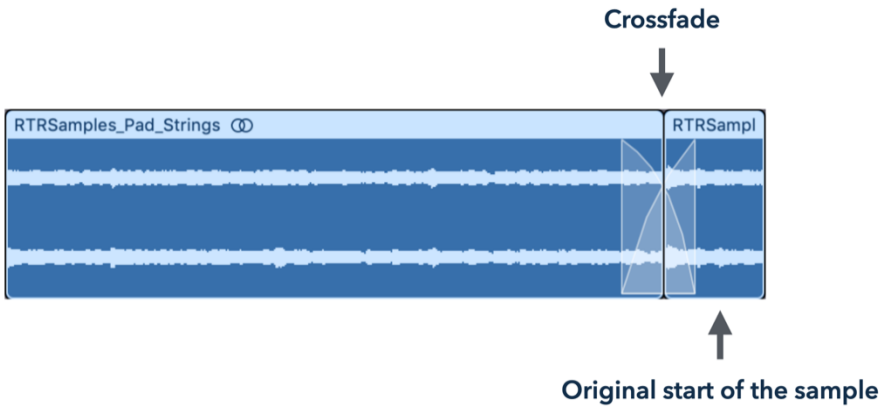


Figure 3.7 Depiction of the technique used to make sounds loop-able

Figure 3.8 depicts a Shepard scale using MIDI notes. Three layers are separated by an octave, and all slowly rise an octave in pitch. The highest tone slowly fades out in volume, while the lowest fades in. This way the same number of tones is always audible. To help mask the fact that the sound is looping, the audio has been layered as much as possible and more complex sounds have been used. At a certain threshold (that varies per person), very high or very low frequencies become less audible to the human ear. This can be compensated for by slightly increasing the volume of the very low and high layers.

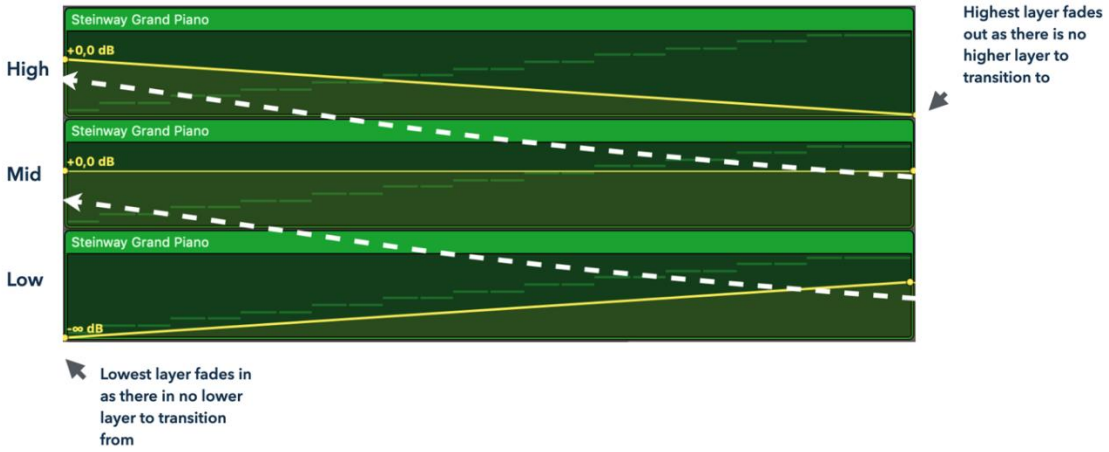


Figure 3.8 A Shepard scale using separate notes

3.4.3 Mock in FMOD Studio

To first test the design, a mock in FMOD Studio has been made[79]. FMOD events hold sequencers on which sounds, or sub-events can be placed. A main event has been divided into layers (Figure 3.9). Every layer's event holds and loops the sounds for that specific layer. Every layer uses amplitude modulation for the position modulation and action modulation. A few layers also have pitch modulation implemented. Every layer's modulation factors (curve, range, and seek speed) have been set depending on the type of sound and intended effect, as described in 3.2.1. For example, the noise only uses amplitude modulation to slowly fade in, while a pad rises more quickly in both amplitude and pitch. When the stop parameter is set to one, the impact is immediately triggered, and the riser stops. The impact consists of a random selection of three hits (higher frequency impact samples) and three subs (lower frequency impact samples).

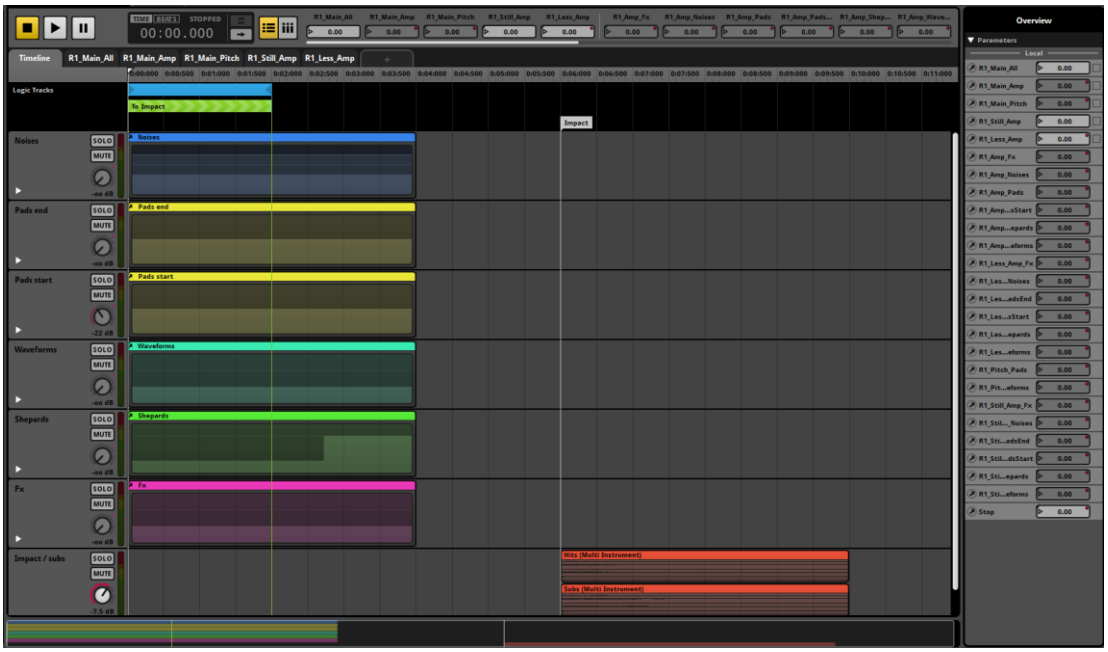


Figure 3.9 Screenshot of the FMOD main event and its parameters (right)

Creating this mock provided substantial insight in the current difficulties that arise when applying the real-time method using existing standard middleware. To have full control over the risers, every layer needs to be able to be modulated separately. To achieve this in FMOD, every layer and modulation type must be linked to a parameter by hand. FMOD Studio also does not provide a default modulation curve. Every curve needs to be drawn by hand, even though often the same or similar curve shapes are used.

[79] Koning, S. de (2021). *RTR Mock Showcase*. Retrieved January 2021, from http://sdkoning.com/PF/RTRMock.html

Another related issue is that every parameter needs to be set by hand using sliders. To be able to test the modulation of multiple parameters as once, macro parameters that modulated all these parameters had to be created. Finally, the overview FMOD provides of the parameters is a list without any information on what parameters influence what aspects (right side of Figure 3.9), and parameters can only be viewed and edited separately.

To integrate the audio in the test level, standard FMOD Unreal blueprint integration has been used (Figure 3.10). Because the mock only serves to investigate, this implementation does not include every feature as designed in 3.3. For this test, the macro parameters were used, as setting every parameter manually using blueprints would take too much time. Triggers for the start and the impact were placed in the level and used to start and stop the FMOD main event. The players distance to the impact has been scaled to the amount of distance between the start and the goal and used to influence the macro control of the mock's position modulation. The result within the test level was satisfactory, indicating the method of adapting parameters to suffice.

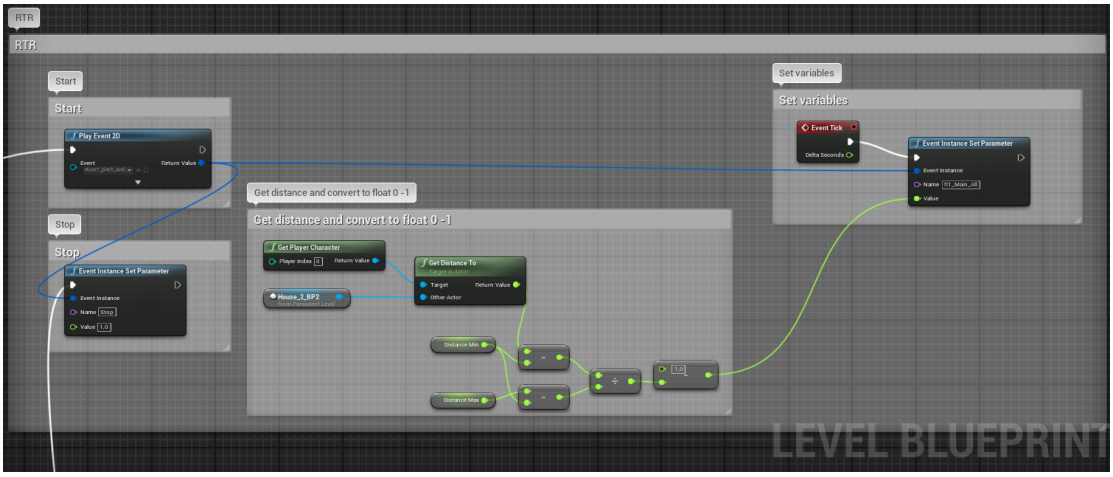


Figure 3.10 Screenshot of the level blueprint used to implement the mock FMOD project

3.4.4 Audio Software Development Tools

To decide on an approach to developing the tool's prototype, common approaches to game audio tools have been examined and their main advantages and disadvantages have been listed (Appendix C). The examined options include middleware or game engine plugins, game engine editor tools, and a middleware-based approach. The middleware-based approach consists of a separate editor used by the audio designer to design the audio and its adaption, similar to the way standard middleware work (Figure

3.11). This editor exports data about the audio and adaption to the engine, which is then read and interpreted by a game engine integration plugin.

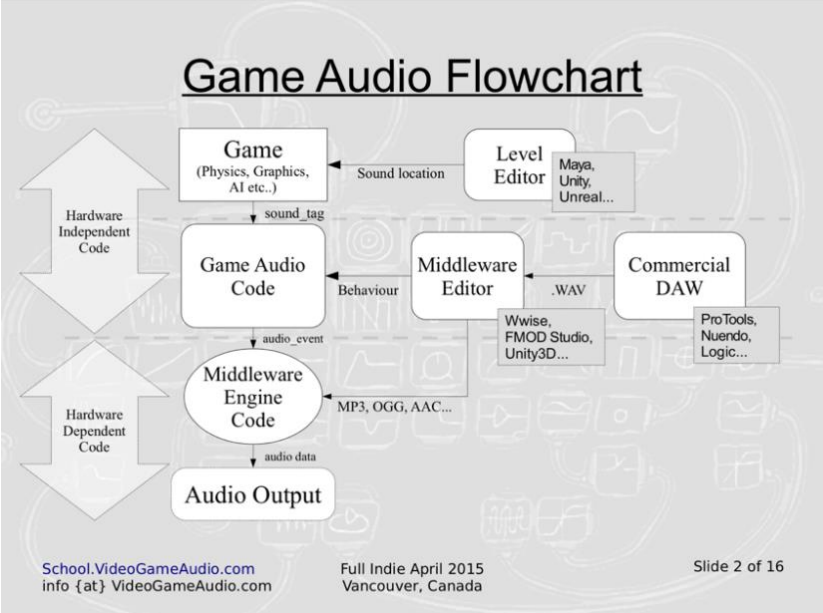


Figure 3.11 Dataflow of the game audio approach using middleware[80]

Because of this research's exploratory nature, there is a high likelihood that features will have to be adapted or extended upon. As stated in Chapter 2.2, a modular approach is preferable as it is more resistant to quick advances in the industry and can be more easily adapted or extended on. This statement is supported in 3.2 by De Graaf. The tool should also be easy to fit into the current audio workflow to mediate the risk, described in Chapter 3.1.4, of the tool not being viable because it takes too much time to implement in the workflow of an audio designer.

Based on the examination of different approaches, the middleware-based approach has been selected. Middleware or game engine plugins are optimised for audio processing but provide limited control on the way adaption is handled. An editor plugin would be a relatively quick option that directly addresses the issue of having to switch between different software, as everything will be done within the engine. However, it would not provide complete freedom and would be limited to one engine. The middleware-based approach requires more work, but because it is based on current middleware solutions, it is an approach that audio designers are used to and can be implemented into any game engine.

[80] Huguenard, C. (2019). *Note Based Music Systems*. In G. Somberg (Ed.) Game Audio Programming 2 (307 - 331). Taylor & Francis Group. P. 308

3.4.5 The Prototype

The developed tool consists of four elements (Figure 3.12). The audiosystem is used to play and modulate the audio. As with standard middleware, the editor tool[81] and the engine plugin[82] use the same low level audiosystem. In the editor tool the audio designer can design the riser, set settings on how the riser should adapt to game data, and play the riser by mocking game data. The editor tool exports the user's settings to a .json file that is imported by the UE4 plugin. In the UE4 plugin, blueprints can be used to integrate the audio. An overview video that demonstrates this process is available at http://sdkoning.com/PF/RTRShowcase.html.

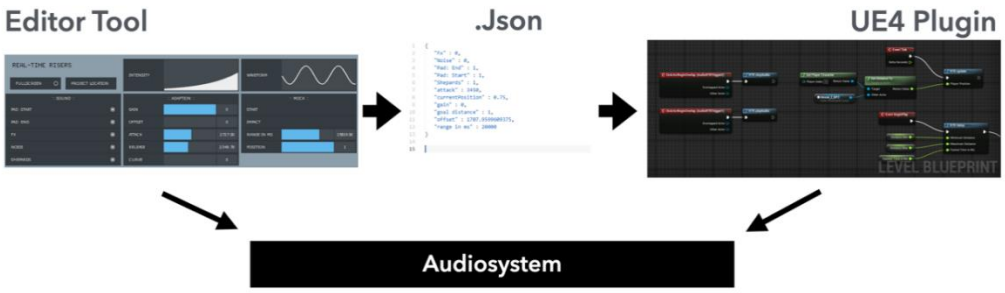


Figure 3.12 The waveform of a static sample (left) and a sample that rises in amplitude

To be able to create the low-level audiosystem quickly, FMOD's low level API has been used. For the editor tool's visual interface, the library OpenFrameworks[83] has been used. The tool automatically loads and parses the audio into different layers, using specific filenames. Figure 3.13 highlights different sections of the tool.

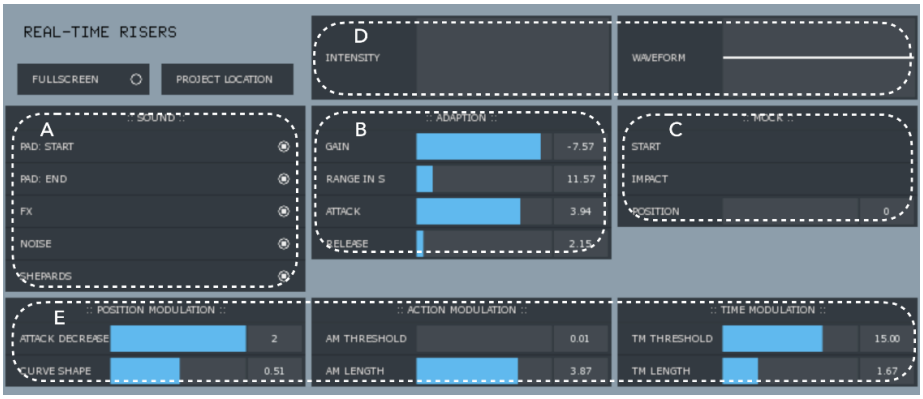


Figure 3.13 MVP of the editor tool

[81] Koning, S. de (2021). GitHub editor tool project repository. Retrieved January 2021, from https://github.com/StijndeK/RTR
[82] Koning, S. de (2021). GitHub UE4 plugin project repository. Retrieved January 2021, from https://github.com/StijndeK/RTR_UE4Integration
[83] OpenFrameworks (open-source C++ toolkit)

A. In the SOUND tab, the user can select which layers should play. In potential future iterations, the user should be provided with more extensive control over the sound.
B. In the ADAPTION tab all settings on how to adapt the audio can be controlled. This now includes the maximum gain value, more precise control over the start of the riser, the length of the release, and the range of the slider. The range is the minimal amount of time that the riser needs to build up from minimum to maximum intensity.
C. The MOCK tab can be used to play the riser in the editor tool. For the current version this includes manual start and impact buttons and a slider to set the players position during playback.
D. Visual feedback is provided with an intensity curve and a waveform visualiser.
E. Control over the three different modulation types is provided at the bottom of the screen. The POSITION MODULATION includes an offset to the range set in the ADAPTION tab and the curve shape. Both the ACTION MODULATION and the POSITION MODULATION have threshold and length (range) input sliders.

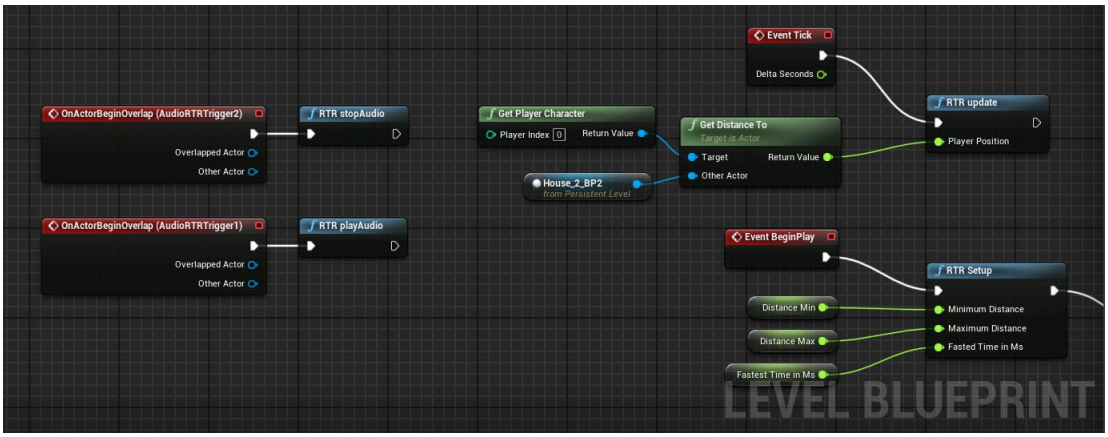


Figure 3.14 Screenshot of the blueprint functions of the UE4 plugin[84]

For the integration in UE4, two different methods have been developed. First, two actors have been developed, one for the start and one for the impact position of the riser. These actors can be dragged onto their desired locations and the start actor needs to be provided with a reference to the placed impact actor in the start actor's inspector. For the second provided method, four blueprint nodes have been developed (Figure 3.14). A start and a stop node serve to trigger the start and impact of the riser. The setup node is

[84] Koning, S. de (2021). *RTR Tool Showcase*. Retrieved January 2021, from http://sdkoning.com/PF/RTRShowcase.html

used to initialise the tool and pass values on the minimum and maximum distance for the player. The setup node also asks for the fastest time the player can reach the impact from the start position to set the initial main seek speed. Finally, the update node needs to be called every 60th second to provide the players position and update the modulation accordingly.

3.4.6 Future Developments

This prototype contains the minimum functionality necessary to be able to sufficiently test the procedural game audio approach. There are a lot of relevant future developments, some of which already have been mention previously. For example, the user can be provided with more control over the design and modulation of the sound, for example being able to select certain sounds per layer. Furthermore, randomisation is often used in game audio to generate material and keep audio from sounding repetitive. More can be randomised in the tool as it currently only randomises the impact. By doing the interviews at this early stage in the tool's development, a lot of feedback can be gathered early on for potential future development.

3.5 Pilot Study

To test if the designed research method works as intended, a pilot study has been carried out. An initial test interview has been done with audio director and composer Jonathan van den Wijngaarden. It is worth noting that this interview was done while the prototype was still missing some features and therefore the tool was not showcased to its fullest extent. The interview has been used to test the interview questions and layout, test the approach to data coding, and to gain initial feedback on the tool.

The interview went fluently without any hiccups. The interviewee was positive about the tool and provided a lot of initial insights. A lot of subjects came up naturally before a question was asked about it. The data analysis using the described tools in 3.1.3 went well, but the interpretation of the data analysis of this interview was limited because it was the only interview done so far. Based on these results, it seems likely that the data gathered will suffice.

3.5.1 Modifications Made to the Interview

The interview took about 10 minutes more than an hour, so a question on if the audio designer experiences other related issues in the audio design process has been removed. This question was initially added for future research but is not of relevance to the current research. Furthermore, the data coding showed that which audio parameters to modulate was discussed a relatively small amount of time. Because of this, a more specific question on what elements of the audio to modulate and how to modulate them based on game data has been added to the script.

Chapter 4: Results and Discussion

Data has been gathered from interviews with professionals in the field and analysed to answer the research question: What are the essential parameters to control a riser audio effect in a real-time video game, and how can they be implemented into Unreal Engine based on the game's state and events? The interviews also served to gain insight in the viability of the method and how it fits in current standard game audio development practise. Based on the analysed data it seems likely that pitch and volume over multiple layers are the essential parameters to control a riser in a real-time video game. However, a lot of additional parameters can be used, and it might differ for other games than the test level that was used. These parameters can be effectively linked to game data based on action, time passed, and game action over time. The real-time riser effect can be efficiently implemented using a middleware-based approach that works side-by-side with current standard tools, although integrating the tool into middleware or a game engine is often preferred.

This chapter lays out and discusses the results of the interviews. Typically, the results and discussion chapter would be separate. Because of this research's qualitative and exploratory nature, this would involve extensive recapitulation, which would be nonoptimal to the development of the thesis. In nine interviews with audio developers, the research has been explained, the prototype has been showcased, and questions to help answer the research question have been asked. The length of the interviews ranged from 52 minutes to 65 minutes. This chapter first discusses adjustments made to the methodology during the process of taking and analysing the interviews. Then the coding structure is highlighted. After this, the participants and their backgrounds are discussed. Then the results gathered from analysing the interviews are stated and discussed. Because of the research's exploratory nature, every interview had a different emphasis, and often new ideas and concepts were brought up by interviewees. Relevant data outside of the immediate scope of the research will be discussed in Chapter 5. Then the validity, replicability, and ethical concerns are discussed. Finally, the results and discussion are summarised.

4.1 Alterations Made to the Methodology

Multiple changes have been made to the methodology during data collection and analysis. The criteria for interviewees have been broadened, an additional video has

been made for the showcase part of the interview, the script has been altered, and the method of analysing data has been modified.

4.1.1 Expanding the Interviewee Experience Criteria

The criteria for interviewees have been expanded to any type of game audio developer instead of solely sound designers. Firstly, because there is no standard naming convention for roles within game audio development. The exact role of someone who works under a specific title can differ widely, which makes filtering by job title less relevant. Also, the research proposes a method to audio design that does not necessarily need to be applied by traditional game sound designers. As will be discussed in Chapter 4.4.2, the proposed method requires an alternative type of audio design. Furthermore, a broader collective experience provides a wider perspective on the tool. This fits the research's exploratory nature as it provides more diverse feedback on the viability and therefore improves the relevance of the research. For example, audio programmers could also provide insight in the technical obstructions and requirements of the method. A broader range of experience over the interviewees does make it more difficult to draw specific conclusions as it results in small subsets of people in specific roles. However, the different backgrounds can also be used to compare certain answers and discuss if certain results are based on the experience and field of work of the interviewee.

4.1.2 Sending a Video for the In-Game Showcase

When showing the tool in-game, with all the game sounds, the quality of audio sharing over an online video call turned out to be too low for participants to be able to provide sufficient feedback. The first interviewee Clément Duquesne mentioned that he could not make a proper judgement because of it and pulled up a video of the tool that was sent to him earlier instead. Because of this, a one-minute video showing the tool in-game[85] was made to be send to the participants at this point in the interview. After the third interview with Tom de Smit, after showing this video the interviewee was also asked whether they wanted to also see it demonstrated live, as De Smit stated he would like to see the tool in a less controlled environment. It is important to note that the second interviewee, Nicolas Fournel, is the only participant that has only seen the video and was not asked if he would have wanted a live demonstration as well, which may have had an influence on his answers.

[85] Koning, S. de (2021). RTR Showcase in-game. Retrieved April 2021, from https://www.youtube.com/watch?v=pbYCn5AFTOo

4.1.3 Script Alterations

Some alterations have been made to the interview script (Appendix D). As not only sound designers have been interviewed some questions have been adjusted to the background of the interviewee. For example, audio designers were asked if the tool would fit in their workflow while audio programmers were asked if they thought the tool would fit in the audio design workflow. Because of this research's focus on exploring the subject, interviewees were encouraged to mention and ask anything they might think of. As interviewees generally mentioned a lot of valuable insights not directly related to the research question, some difficulties arose fitting all the questions in the timeslot. Because of this, three questions were established to be less relevant to the research question and have not been asked to everyone. The question on the interviewee's experience with the horror genre was not asked to every interviewee as the research's test level is a horror game, but the focus of the research question is not on horror games specifically. The question on how the interviewee would approach a build-up type of goal for audio was also deemed les relevant, as not all participants were audio designers. The question on the understandability of the prototype's user interface has not been asked to every participant because the user interface of the prototype is not directly relevant to the research question. Because only a few people were asked these questions, too little data was available to conclude anything on these specific subjects. Furthermore, every participant immediately agreed with the stated issues. Therefore, the question about whether they still agreed with the issues after the showcasing of the tool was removed. Also, some of the interviewees defined there to not be a difference between procedural audio and adaptive audio. Because of this, the question on whether they have experience with procedural audio was not asked to them, as this question intended to ask the participant about their experience with procedural audio as a specific type of adaptive audio and not about their experience with adaptive audio in general. After the interview with Duquesne, it was also added to the script to make specifically clear that the prototype is a proof of concept. Duquesne initially was not completely sure of how to assess the prototype.

4.1.4 Data analysis

The interviews have been transcribed using Otter[86] and coded using MAXQDA[87]. Originally, NVivo[88] was mentioned as the coding tool this research would use, being also

[86] Otter.ai (speech to text tool), Otter.ai (2016)
[87] NVivo (qualitative data analysis tool), QSR International (2020)
[88] MAXQDA (qualitative data analysis tool), VERBI Software (2020)

the tool originally used for the pilot study and the first two interviews. However, because of NVivo's pricing, MAXQDA was tested as an alternative. Due to MAXQDA's modular interface and its visual representation of the coding, MAXQDA was then considered more optimal for efficient coding. Because of this the previous codes were manually transferred to MAXQDA, and all further coding was done in MAXQDA. Furthermore, contrary to what was stated in the methodology, keyword analysis has not been used. The data coding provided sufficient data for the scope of this research, and because of the research's qualitative nature, keyword analysis provided a limited amount of insight.

4.2 Data Coding

Based on the gathered data from nine interviews, 48 codes and subcodes were created and divided into six topics. A code handbook can be found in Appendix G. First, codes were created for every interview. Then these codes were combined into top-level codes based on the similarity per discussed subject. Figure 4.1 shows an overview of the main subjects in which the codes have been divided and how many times they have been referenced. 'Parameters and Modulation' are all codes that answer the research question on what parameters to modulate and how to adapt them to game data. 'Viability' discusses the value and industry relevance of the proposed method. 'Procedural Audio' and 'Background Information' both serve to gain insight into the specific interviewee. 'Pipeline and Workflow' includes every code on how to integrate the method in an engine. 'Other Relevant Data' collects other codes that do not fit in any of the above-mentioned subjects.

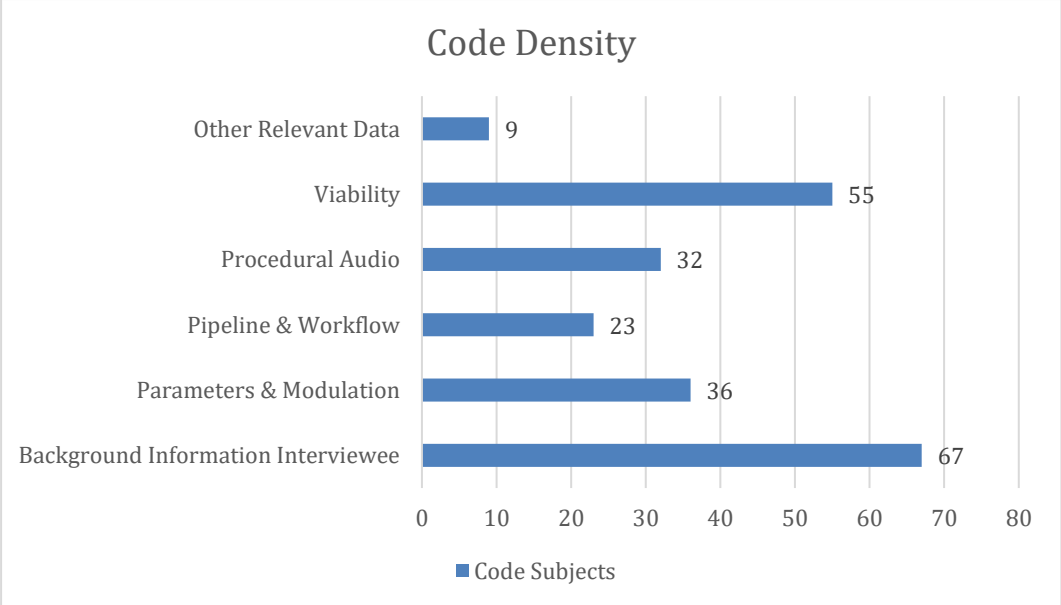


Figure 4.1 Code density of main subjects

4.3 Participants

Participants based in East Asia, Western Europe, and North America have been interviewed. Questions on their experience and views served to establish a profile of the test group and to put answers in context of the specific interviewee's experience and interests. Figure 4.2 show an overview of the participants' roles and their years of experience in the industry, based on the data summarised in Appendix E.1. The years of experience have been divided in groups of five years. The figure shows that the amount of experience varies widely and that there is not a specific amount of experience that has been analysed substantially more than others. The role titles have been divided in four categories: audio design, audio programming, audio direction/lead programming, and audio direction/lead design. The term audio designer is used for people who describe themselves mainly as being a sound designer or composer, which are the people who develop the audio assets and possibly implement them as well. The title audio programmer is used for people whose focus is on any game audio related programming. The category audio direction/lead is used for interviewees who work in managing and leadership type roles. Because audio directors or leads often have extensive experience in audio design or programming, they have been subdivided based on whether they are more experienced in design or in programming. Every interviewee was assigned one of these labels based on their own description of their expertise. An overview of these descriptions and every interviewee's interest in the field can be found in Appendix E.2. For example, Duquesne stated: 'I am a game audio designer and I do a bit of programming also.'[89]. Because Duquesne first stated to be an audio designer and secondly mentioned to do a bit of programming, Duquesne has been classified as an audio designer. This classification has been made to provide a clear overview. However, due to differing career trajectories, the exact experience of participants within each category can vary widely. Overall, the figure shows the interviewees to have diverse experience, both in the amount of time in the industry as in their types of roles.

[89] Duquesne, Appendix F.2.1

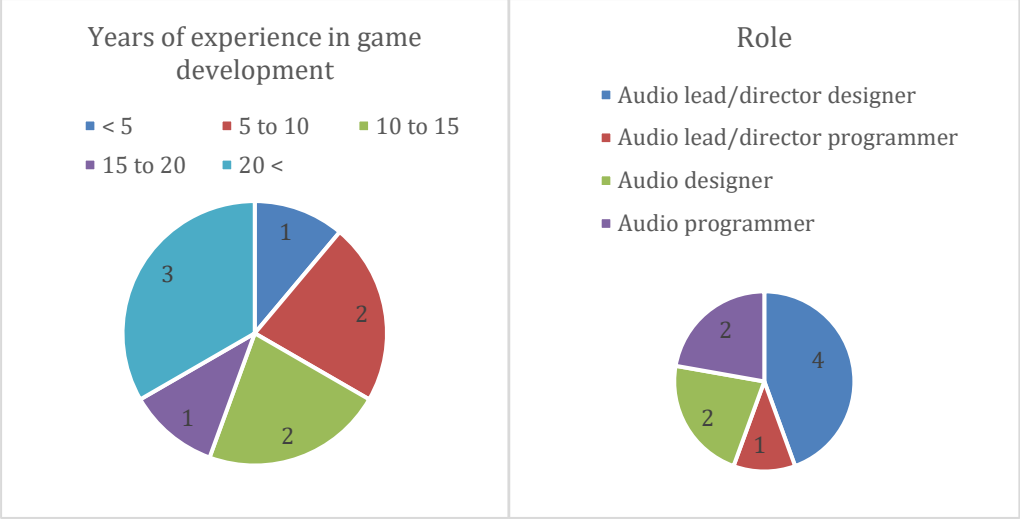


Figure 4.2 Overview of participants' roles and experience in game development.[90]

4.3.1 View on Experimenting and Recognition of the Stated Issues in Game Audio Design

Participants were also asked about their view on experimentation in the field. All participants stated to enjoy experimenting and agreed that the space for it is mostly based on available time and budget. De Smit said: "I like to try new things. Provided there is time. There might not always be time."[91]. And Duquesne similarly mentioned: "Well, it really depends on the context of the developments, because sometimes you just do not have the time to experiment."[92]. As every participant mentioned they like to experiment in one way or another and that it often depends on the available resources, it can be stated that they all see the value of an experimental approach but are also aware the resources for it might not always be available.

Chapter 2 indicated there being room for improvement in the game audio pipeline for the specific issues in adaptivity and workflow. All the participants recognised the issues related to game audio as stated in this research. However, Florian Fuesslin did also mention that he would not call them problems: "I do not think it is a problem, it is just the reality we work in. Without the players we do not have a job. And the player is the disturbing factor in our world."[93].

[90] Interview Experience Table, Appendix E.1
[91] Smit, Appendix F.6.1
[92] Duquesne, Appendix F.2.2
[93] Fuesslin, Appendix F.4.1

4.3.2 Procedural Audio View and Definition

In Chapter 2.3 it was stated that the term procedural audio does not have a clear definition and that this research defines procedural audio as hyper adaptive real-time audio. At the start of the interviews, participants were asked a few questions about procedural audio to obtain more background information to relate their answers to, and to make sure that it is clear what is meant when the term procedural audio is mentioned during the interview.

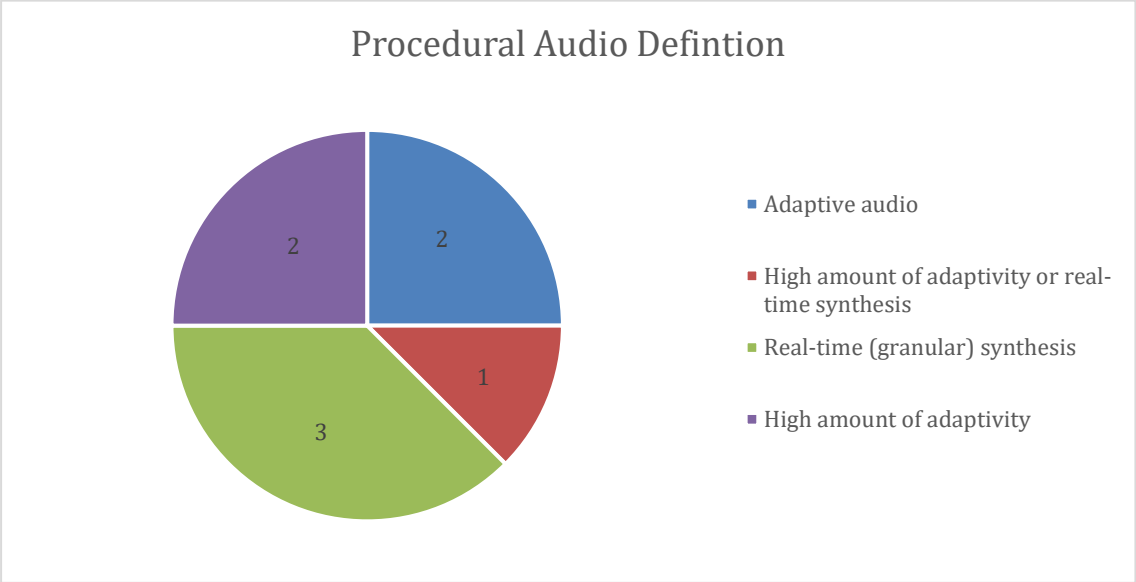


Figure 4.3 Overview of participants' definitions of procedural audio.[94]

Figure 4.3 shows the division of the participants' definitions of procedural audio over four categories: adaptive audio, high amount of adaptivity, real-time generation (synthesis), high amount of adaptivity or real-time synthesis. It is important to keep in mind that half of the interviewees mentioned not to be completely satisfied with the definition they gave or stated that it is a difficult question. Tom Hays defined the adaptiveness of audio as a continuum:

> I do not think there is one or the other. I think that well executed triggering of individual, one shots and stereo sounds and music streaming, it is procedural. It is almost like granularity is a continuum where on the one end you might have a single sound for a gunshot or for a rock fall, and then at the other end you might

[94] Interviewee Procedural Audio Definition Table, Appendix E.3

> have sounds that tie into every little, tiny interaction of the rock as it rolls down the hill.[95]

Audio programmers Fournel and Bogdan Vera specifically mentioned the distinction between procedural audio as a high amount of adaptivity and procedural audio as real-time synthesis. Fournel defined procedural audio as generative audio but also indicated to be aware that others see it as a threshold of adaptivity[96]. When asked about the definition of procedural audio, Vera responded by asking: "Do you mean procedural audio synthesis or procedural audio arrangement?". Vera then continued to explain:

> I mean, arrangement would mean generative composition, sort of. You put together segments of music or sounds in some way, procedurally. As in according to some rules automatically. Procedural synthesis would be, instead of going to record a sound file, you generate it from some algorithm. I think those things are very different.[97]

When asked about the advantages and disadvantages of procedural techniques, all participants agreed they can be technically demanding and are not always required. Huguenard also stated:

> The downside is that they blow up easily. There are edge cases to deal with and stuff like that. And going with sort of like linear production techniques, and then applying some interactivity to those resulting sounds can be a lot more stable and predictable.[98]

The advantages mentioned are also comparable for all interviewees. Most of them indicated procedural audio allows for quicker reactivity and that it can automate a lot of the design process. A few interviewees also brought up that by being able to react more quickly, game audio design can move more towards movie-type sound design. Finally, Fournel stated that often both should be used:

> Well, I think in a lot of cases, you might want to use both. First its limited, again, by the CPU usage you can have. So, in most cases, you probably want to use procedural audio, maybe if you want more interactivity for your sound. Let us say you have a fight between dragons, and your dragon is just in front of the

[95] Hays, Appendix F.5.1
[96] Fournel, Appendix F.3.1
[97] Bogdan, Appendix F.8.1
[98] Huguenard, Appendix F.6.1

> screen, in the middle of the screen, just in front of the player. Obviously, you want the roar to follow the movement of the throat of the dragon and everything. You will want that to have like, full control, full procedural audio. If there is another player's dragon very far away, you do not need procedural audio for that, you do not even see the thing very well. You can play a sample or a couple of fundamental samples.[99]

4.4 Gathered Results Overview and Discussion

The data gathered has been divided into multiple subjects based on how they relate to the research question: What are the essential parameters to control a riser audio effect in a real-time video game, and how can they be implemented into Unreal Engine based on the game's state and events? The first topic discussed is the riser parameters to adapt and how to adapt them based on game data. How to integrate the proposed method in the pipeline and workflow is then reviewed. Finally, the tool's viability and its industry relevance are examined. For every topic, the results are laid out and the meaning of the results are evaluated to make an argument about how the data could address the research question.

4.4.1 Riser Parameters and Modulation Based on Game Data

All interviewees were asked what they thought about the chosen parameters in the prototype and how these parameters are modulated based on the game data. This directly addresses the first part of the research question on what the essential parameters are to control a riser audio effect in a real-time video game, as well as how they can be implemented based on the game's state and events.

Every interviewee, except Duquesne, stated that the parameters used in the prototype suffice, either stating that they make sense or that they work well. The prototype divides the riser in five layers and modulates each layer's volume and optionally pitch with a seek-speed and curve. One of the interviewed audio directors also noted:

> Sound design is sometimes pretty simple. It is time, frequency is very important. And if you design sound, people always say 'more plugins'. But in the end, it is like 'I use pitch, volume, maybe reverb'. The standard feature set EQ, compression, that is way more than you usually need. So, I agree let us not make

[99] Fournel, Appendix F.3.2

> it overly complicated. Especially as it seems to work, and you achieve the feeling or the emotional point you wanted to make.[100]

The interview with audio designer Duquesne did not contain sufficient data on the parameters to base a clear conclusion on his opinion of this subject. However, Duquesne did state about the prototype's controls that: "It does only one kind of riser. When you are a sound designer and you are looking for ways to do something, you mostly want to control how it will sound, what sounds you will use."[101]. This could indicate Duquesne finds that the prototype's parameters are not enough to suffice as the essential parameters. However, this could also solely be directed at the tool's user interface. All participants did name parameters to add, and four of the interviewees specifically indicated that there is a high number of parameters to add. Also, both Vera and Fuesslin said that a riser does not have to be a new sound but can also be the adapting of already playing sound. Vera stated: "Think of it like the riser itself does not have to be just the sound that is rising, the sound effect that is applied. It is the gesture of the riser that can be applied to many things. It can be applied to the lighting in the scene"[102].

All participants except Duquesne agreed with how parameters are modulated based on game data in the prototype. Duquesne found that the riser did not rise enough at the end.[103] Charlie Huguenard said: "The abstraction of position in motion or action, that makes sense."[104]. And Fuesslin stated: "I think it is a good start."[105]. Huguenard, De Smit, and Davis brought up the issue of what would happen when the player starts behaving very unpredictably. Davis stated: "The biggest problem in the game […] is 'do not do what you want to do'. You start the trigger, and then they go 'oh look, shiny squirrel' and wander off in a completely different direction."[106] Fuesslin argued that players tend not to behave unpredictably in this test case:

> This is a psychology thing, but I guess you would not stop pushing forward with the controller. Because you will continue. The building is interesting, how it is burning, it triggers all these questions. And then the riser is basically supporting your approach to the impact moment. And that is perfect.[107].

---

[100] Fuesslin, Appendix F.4.2
[101] Duquesne, Appendix F.2.3
[102] Vera, Appendix F.8.2
[103] Duquesne, Appendix F.2.4
[104] Huguenard, Appendix F.6.2
[105] Fuesslin, Appendix F.4.3
[106] Davis, Appendix F.1.1
[107] Fuesslin, Appendix F.4.4

About the issue of what would happen when the player behaves unpredictably, Vera stated:

> The thing is players do not really do that. It is like, if they are trying hard to break your riser, it is their fault. It is a video game. The average person really will not, because it is a video game anyway. I can go really close to a texture and look at it and it is going to look like it is low res, because it is a video game. So, it is like 'do not look at it so close'. Same with audio, the experience for you is, even though it is dynamic, it is still going to be a little bit railroaded because there is a planned event. That is fine.[108]

Based on these results, it can be stated that the riser's parameters in the prototype of volume and pitch modulation are the basic parameters of a riser to adapt to game data, at least for the specific test level used in this research. The essential parameters to control the modulation are the seek speed, and curve. Eight out of the nine interviewees agreed with the parameters and the modulation used in the prototype. This supports the analysis of existing riser plugins in the literature review (in Chapter 2.4), where these were also the most prevalent parameters. However, most participants also stated that a lot more is possible and that what to add would depend on the specific game or level. Modulation based on action, time, and action over time can be effective game data to control the riser's parameters. All participants except Duquesne agreed with the prototype's approach to handling the adaption. Duquesne found the riser did not rise enough at the end. Some interviewees viewed unpredictable player behaviour as a potential risk that the prototype's adaption might struggle with, but Vera and Fuesslin gave multiple reasons for this not to be an issue.

4.4.2 Pipeline, Workflow, and User Control

The participants were asked about the way the prototype is integrated in Unreal Engine and how this relates to the current standard game audio design workflow. This topic addresses the part of the research question on the possibilities of implementation of the real-time riser into Unreal Engine. In chapter 3.4 the prototype's middleware-based approach is described as having a standalone tool that exports data to a game engine plugin, similar to standard game audio middleware. Identified advantages of this approach include it being a method audio designers are used to and that it allows for a

[108] Vera, Appendix F.8.3

modular and unrestricted approach to the development. Four of the interviewees specifically stated this approach works well. Davis said it works well as it is simple enough to be alongside standard tools, but also mentioned that an AAA studio would probably want it to be integrated in their engine[109]. Fournel emphasised it works well as long as it saves time[110]. Fuesslin noted that it works well but he also said it would be useful to implement it in middleware[111]. Hays noted that the prototype's approach works well and there might be difficulties integrating the tool in an existing middleware: "I think you could come close, but its underlying intent is different."[112].The other interviewees thought integrating the method within an existing standard middleware or engine would be a better approach. Notable is that two of the four interviewees that did find the prototype's approach sufficient, also mentioned integrating the tool in current standard solutions, which increases the relevance of integrating the method in an existing toolset. All four interviewees that stated the prototype's approach works well are categorised as audio lead/director. A possible explanation for this might be that people in leadership positions are less focused on the practical workflow, as they will generally not be the ones to apply the tools as much as the audio designers. However, it could also be argued that as they generally have been in the industry longer, they may more experience with adapting their workflow.

Four of the interviewees mentioned the tool's integration in current standard middleware as a further improvement. Two of the four indicated this as a possible further improvement, while the other two stated it as a necessity for it to be viable. Audio designer De Smit said: "what I prefer most is to have most things centralised. I would probably ask if this could be converted into a Wwise plugin."[113]. All four interviewees that mentioned the integration in middleware are audio designers or leads/directors with a background in audio design, which may be explained by the fact that they are the people to use the tool and integrate it in their workflow.

Generally audio programmers placed more emphasis on the constraints and work required to do it for existing engine or middleware but are also aware that audio designers would want it to be integrated. Audio programmers Vera and Huguenard as well as audio director Davis stated it could work as a middleware plugin, but also

[109] Davis, Appendix F.1.2
[110] Fournel, Appendix F.3.3
[111] Fuesslin, Appendix F.4.5
[112] Hays, Appendix F.5.2
[113] Smit, Appendix F.7.2

mentioned integrating the tool in a game engine. When asked about the trade-off between different approaches Huguenard explained:

> It depends. Sometimes if a project that is using your thing is specialised enough and specialised in favour of the thing that your tool uniquely does, then it can be a conversation about using that instead of Wwise or using that instead of FMOD. Where it starts to get a little difficult to sell, is when the project calls for a lot of things that one of the more widely used tools does really well.[114]

Another discussed topic about the controls the prototype provides relates to how to implement the riser efficiently, but also to what riser parameters to modulate and how to modulate them. The prototype mainly provides control over how the audio is to be modulated. For sound design the user can solely turn specific layers of the audio on or off. The control offered has been explained in more detail in Chapter 3.4.5. Audio designers generally wanted more control over the sound design. Audio designer Duquesne said that as a sound designer he mostly wants control over the way it will sound[115]. Fournel made a distinction between the implementation and the creative element of the tool: "I think where you want to give options is more on the creative side and less on the implementation side."[116]. Fournel also mentioned that this type of tool provides a different type of control requiring a different type of audio design than most audio designers are used to. Vera supported this by saying:

> There is a tension when it comes to sound design people not being in control of the same parameters they are used to. They know how to go record the sound they want. You have to teach them a whole new technical skill when it comes to procedural. It is almost like a procedural sound designer is a different role than a sound designer.[117]

If the research's method were to fit in the standard pipeline and be used by audio designers, some form of centralisation seems essential. The data collected on the workflow and user control suggests that while the middleware-based approach of the prototype has certain advantages, it is preferable to be integrated to a game engine or to middleware. Seven interviewees mentioned integrating it in an engine or middleware.

---

[114] Huguenard, Appendix F.6.3
[115] Duquesne, Appendix F.2.3
[116] Fournel, Appendix F.3.4
[117] Vera, Appendix F.8.4

Audio designers prefer this centralisation to be done in the middleware, not the engine. Audio programmers recognise this, but also argue for the limitations of integrating the method. Similarly, audio designers also prefer complete control over the sound design in a way that they are used to. However, it could also be argued that this different type of method may require a different type of audio designer.

4.4.3 Viability and Industry Relevance

The viability of the research's approach indicates how well the riser's parameters are adapted based on game data, as well as the relevance of the research to the industry. Seven out of the eight interviewees stated that the method has potential. Some interviewees were very convinced. Huguenard said: "It has potential? Yeah of course. Is that a trick question?"[118]. Others agreed that it has potential, but also mentioned some conditions. Jonathan van den Wijngaarden stated: "I think it has potential. It is an interesting idea. It might have to tackle a few hurdles, but I think you are already thinking in the right direction."[119]. Five interviewees said they are curious how well it performs for edge cases, indicating that although they believe the approach to be viable, there are certain potential issues that have not yet been addressed in the developed prototype. Interviewees who mention potential use cases or think of use cases where they would have liked to or could have applied it in the past, speaks to research's viability. Six interviewees mentioned a use case for which they would have liked to apply the method in the past. The other interviewees said the type of projects they worked on would not have required it. Furthermore, seven interviewees put forward use cases for which they would find the tool interesting or for which they thought it might work well.

Duquesne was the only interviewee who was not positive towards the viability, as he stated:

> I am so familiar with Wwise that I think I would still go to Wwise for something like that. Also, I did not have many risers in my games. So, I am not thinking, 'Oh I could have used this at this moment or this moment'. In my personal experience I would not have used it, I think.[120]

[118] Huguenard, Appendix F.6.4
[119] Wijngaarden, Appendix F.9.1
[120] Duquesne, Appendix F.2.5

Here Duquesne already mentioned the fact that he is familiar with Wwise as one of the reasons he would not want to use it, as well as that he did not have risers in his games before. His programming experience was another reason for not considering the prototype as viable for his own use: "one of the reasons that I am not very attracted to specific audio middleware, is that I have the competence to do audio tools that answer to specific needs. I am tempted to do this myself if I have to."[121]. Finally, Duquesne also said: "It is a thing I often find with small procedural tools like this, is that if it is not important for your game, then you do it in another simpler way or you just do not do it if it is not that important."[122]. Notably, Duquesne was also the only interviewee that did not agree with the way the riser is modulated, which could be another reason why Duquesne would not want to use the prototype.

It can be argued that the research is relevant and viable as eight of the nine interviewees stated the research to be viable. Their naming of potential future use cases or cases where they could or would have liked to use it in the past, supports this statement. Duquesne did not find it interesting, which can be explained by his ability to program himself, not finding the modulation sufficient, being used to a workflow with Wwise, not having used risers much before for games, and wanting full control over the riser. Others also indicate that they are curious about how it performs in edge cases, which could be valuable further research.

4.5 Validity, Replicability, Relevance, and Ethical Concerns

The meaning, importance, and relevance of the results discussed in this chapter have been reflected on. The validity and replicability is discussed to indicate what has been done improve the validity and replicability of the research and to note what could be improved with further research. Then the relevance of the research is discussed. Finally, some new and recurring ethical concerns are addressed.

4.5.1 Validity and Replicability

A couple of decisions have been made to improve the validity of the research. Firstly, interviews with nine audio developers with different types and amounts of experience from different parts of the world provide a broad perspective on the research question. Also, the method and all adjustments made along the way have been documented in the

[121] Duquesne, Appendix F.2.6
[122] Duquesne, Appendix F.2.7

thesis, improving the research's replicability. Information on the interviewees and their backgrounds is provided and all used material, including the codes, interview scripts, and the prototype tool, has been made available. Furthermore, this research's approach to asking questions based on a proof-of-concept prototype, reduces the amount of speculation, and applies the research to an example that is directly relevant to the industry. The research's exploratory approach and the script's open-ended questions also reduce the risk of overlooking certain concerns regarding the method of addressing issues in the field and its viability.

There are also a couple of concerns regarding the validity of the research and its capability to answer the research question that are important to note. Because the interviewer was also the one who created the interview and prototype, the participants may have been more willing. Furthermore, since interviewees were asked about their general impression, encouraged to mention anything they might think of, and questions were contextualised to the background and experience of each interviewee, not every question was formulated in the same way or asked in the same order. This is consistent with the qualitative nature of the research, but it should be noted that the order and way the questions were asked may have influenced the interviewees' answers. The next chapter discusses how other research techniques and quantitative research can provide more concrete data for the topic. Also, because of the research's scope, the participants were all shown the tool in a predesigned test level that had been used to develop and test the tool and for which the tool had been optimised. This could affect their opinion of how it will perform for other scenarios. However, most participants did note this themselves. Another concern for the validity is that because participant's from very different backgrounds in game audio development have been interviewed, this reduces the sample size per specific experience. Finally, some of the questions are based on an online showcase, in an interviewer-controlled environment, which may have influenced how the interviewees perceived the research. However, this was addressed by asking interviewees if they wanted to see something specific showcased. It could also be argued that letting people test the tool themselves would also affect their opinion and likely cause the conversation to shift more towards the usability of the tool, which is less relevant to the research.

4.5.2 Relevance

The results indicate an interest in the proposed method as well as that it can be a viable industry application, which supports the research's industry relevance. Eight of the nine

participants indicated that the proposed method is viable. Furthermore, although not all participants have had to deal with the stated issues in game audio development that this research seeks to address, all participants recognised the stated problems. The viability and replicability of the research also support the relevance of the research.

4.5.3 Ethical Concerns

The methodology mentioned the importance of interviewing people from different cultural backgrounds and genders. Even though people from three different continents have been interviewed, there is a lack of diversity among the interviewed professionals, as they are predominantly white and male. This can firstly be caused by the general lack of diversity in the industry, but this is also a reflection of wider issues of access to technology for women and minorities, and this is rooted in deeper systemic issues of sexism, racism, and capitalism in society. The games industry will need to take action to diversify its workforce, but this will be most effective when aligned with wider societal change. Other potential explanations are that minorities in the industry might not be confident to do interviews, they might be frequently asked for these types of interviews because they are part of a minority, or they might not want to be asked based on their gender or other characteristics. Another concern was raised by one of the interviewees when discussing the user interface which is that the tool's font is pixelated and unclear. The font is a default font that has not been altered for the prototype. If this type of tool were to be released it would have to be changed to make it easier to use for people with dyslexia or visual impairments.

4.6 Summary

The collected and analysed data indicate that the parameters and modulation used in the prototype are the essential parameters to control a riser, and that they can be adapted to game data using modulation based on action, time, and action over time. The parameters that the prototype uses are volume and pitch modulation over five layers, with control over the modulation's seek speed and curve. The real-time riser effect can be efficiently implemented using a middleware-based approach, although integrating into middleware or an engine is often preferred, especially by audio designers. Furthermore, the results indicate that the method is a viable approach, which improves the relevance and validity of the research. The diverse backgrounds of the interviewees, the contextualisation of the research question using a prototype, and the replicability of the research also improve the validity of the research.

Chapter 5: Conclusion and Future Directions

The goal of this thesis is to explore potential improvements to the standard game audio workflow by tackling various reoccurring obstructions, using the scoped case of riser audio effects for a walking simulator horror game. The used research question is: What are the essential parameters to control a riser audio effect in a real-time video game, and how can they be implemented into Unreal Engine based on the game's state and events? Researching the common game audio workflow and what has already been done confirmed there to be room for optimisation in the game audio workflow, highlighted some successful solutions in the past, provided insight in the most important features of existing riser audio effects, and provided insight in a potential approach to keeping risers going for a long time by using Shepard scales. The research proposes a method of integrating audio based on adapting parameters of the audio itself, instead of triggering different pieces or layers of audio based on what is happening in the game, with a quick and easy to use integration into Unreal Engine. Professional audio developers with different roles in the field have been interviewed based on a prototype that applies this method to reduce speculation.

The analysed data from nine interviews indicates the essential parameters to control a riser in a real-time video game are the riser's gain and pitch over a multitude of layers. However, a lot of other parameters were named that sound designers would prefer to also control, such as panning and filtering the frequency spectrum. These parameters can be efficiently modulated based on action, time, and action over time, where every modulation type has a separate range and threshold, and every layer has a separate modulation curve and range. The real-time riser effect can be efficiently designed and implemented using a middleware-based approach consisting of a standalone tool with a user interface that extracts a datafile, which is read by a game engine plugin that can be used to link the riser to game data. Using this approach, the tool is not restricted by the limitations of common middleware. However, further research could explore different approaches to the pipeline of the tool, as a high number of interviewees mentioned they would prefer the tool to be integrated in standard middleware or the game engine. The data also indicates the prototype's approach to implementation in Unreal Engine 4 works well, i.e. using two game objects, or with the alternative approach using blueprints. Furthermore, the data gathered showed overall positivity towards the viability of the method, also for other game cases and audio sound effects. Further research should be done to explore applying the method to other audio effects and

games, as well as research what happens when the tool is applied to a game that is to be released to the public. The diverse backgrounds of the interviewees, the contextualisation of the research question using a prototype, and the replicability of the research speak to the validity of the research. However, user testing the tool with audio developers and user testing the resulting game audio with players could also be valuable future research to gain more insight in the proposed method's viability and how to apply it. Also, four of the interviewees mentioned the proposed method allows for making audio less reactive and therefore allows closer alignment with the techniques used in film and other linear media, which is another field of interest to explore in further research. Overall, this research indicates that adapting the pitch and gain of multiple layers of a riser based on time, action, and action over time, implemented with an automated pipeline, could be viable solution to current game audio design obstructions and is worth further investigation.

Appendix A: Interview Script

A.1 Privacy and consent

For the purpose of this research, this interview will be recorded (both video and audio) and transcribed. No sensitive personal data will be recorded. Participation is voluntary, and you are free to refuse to participate or withdraw from the research at any time. Recording and/or transcriptions will potentially be put online to support the research. Do you consent to this?

A.2 Introduction

- Introduce guest (if not anonymous) and thank them for doing the interview.
- *I will start by briefly explaining the structure of this interview.*
    - Introduction
    - Background information on myself
    - Background information on the guest
    - Explanation of the research
    - First round of questions
    - Showcase
    - Second round of questions
- Introduction of the interviewer
- Background information on the guest.
    - *Could you provide a brief description of who you are and what you have worked on?*
    - *What is your interest in game audio?*
    - *Do you generally prefer experimenting with new methods or using existing proven techniques? And how do you go about deciding which methods to use?*
    - *Have you ever worked with horror games or other quick suspense type of scenarios, and if so, can you speak to this?*

A.3 About the Research

- General explanation of the research, using figures to help demonstrate the subject.
    - Research question
    - Hypothesis
    - The tool

A.4 Questions #1

- *Have you ever come across any of the problems described in the explanation of the research (in general, not just for risers)? Or is it something you have thought about before? And do you even see it as a problem?*
    - *If you have, how did you address it?*
- *How would you approach a build-up type of goal for audio right now? For the specific context and in general.*
- Procedural audio
    - *How would you define procedural audio and what makes it different from other nonlinear audio design techniques?*
    - *Do you have experience using any form of procedural techniques?*

    - *If so, what type? And why was it used?*
  - *When should and when shouldn't procedural methods be used and why? What are some of the limitations of procedural audio?*
- *What is your initial opinion on the research's approach to addressing the stated issues?*

A.5 Showcase

- Demonstration of the current prototype. Most of it is subject to change. Please feel free to ask any questions.
- *Do you have any experience with UE4?*
- Showcase the tool.

A.6 Questions #2

- *What is your first impression?*
- *Do you still think the stated issues are a problem or not a problem?*
- *Do you think this method has potential? If it were more worked out, would you want to use it over existing techniques? Could making more types of audio with this technique be used more in the future?*
- *Are there any cases in the past where you would have used this?*
- *What do you think about the chosen audio parameters to modulate and how they are modulated based on the game data? How would you improve or adapt the current system?*
- *Are there any changes you would make? Future developments*
- *Would this tool fit into your current pipeline and/or workflow?*
- *Is the tool's interface easy (enough to) understand?*

A.7 Follow up

- *Would you be interested in me sending you the final paper or an update at the end of the project?*

Appendix B: Test Cases

Case Chase/Escape

Parameters

- Player speed
- Movement direction
- Proximity to goal
- Time passed
- Start and stop
- (Optional for finite level)Distance start and stop

Case Task in Time

Parameters

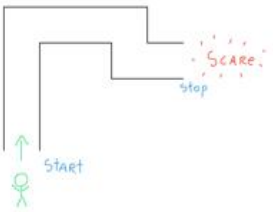
- Progress to goal (amount of enemies)
- Health (enemy progress)
- Time passed
- Start and stop

Case Health

Parameters

- current position
- (Optional) amount of risk (if there are no enemies, no need for heavy tension)
- (Optional) speed
- (Optional) Movement direction
- Time passed
- Start and stop
- Distance start and stop

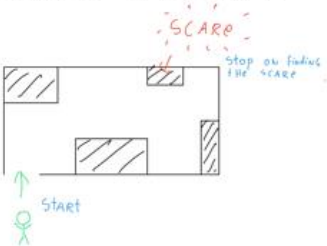
Case Horror 1



Parameters

- Player speed
- Player position / proximity to goal
- Movement direction
- Time passed
- Start and stop
- Distance start and stop

Case Horror 2



Parameters

- Player position / proximiy to goal
- Movement direction
- Time passed
- Start and stop
- Distance start and stop

Appendix C: Tools Development Approaches Overview

Approach	Advantages	Disadvantages
Audio middleware plugin	• Common approach • Integrated in most engines • Well documented	• Plugins are mostly focussed on DSP, not on audio adaption • Limited by middleware's limitations • Still obstructed by having to switch between software
Middleware-based	• Able to integrate in any engine. • Complete control and freedom • Same workflow as middleware, and therefore easier to learn • Modular	• Costs more time and effort
Custom UE4 editor	• Everything in 1 location • Quick • GUI corresponds with engine interface	• Specific to one engine • Limited control
JUCE audio framework Unity plugin	• Easily convertible code	• Specific to one engine • Limited control • Plugins are mostly focussed on DSP, not on audio adaption
Using node-based audio languages Max MSP/Pure Data and OSC	• Quickest prototyping option	• Limiting • Not optimised • Programs always needs to run simultaneously

Appendix D: Edited Interview Script

Edited interview script of adaptions made after the first interview with Clément Duquesne.

D.1 Privacy and consent

For the purpose of this research, this interview will be recorded (both video and audio) and transcribed. No sensitive personal data will be recorded. Participation is voluntary, and you are free to refuse to participate or withdraw from the research at any time. Do you consent to this?

D.2 Introduction

- Introduce guest (if not anonymous) and thank them for doing the interview.
- *I will start by briefly explaining the structure of this interview.*
    - Introduction
    - Background information on myself
    - Background information on the guest
    - Explanation of the research
    - First round of questions
    - Showcase
    - Second round of questions
- Introduction of the interviewer
- Background information on the guest.
    - *Could you provide a brief description of who you are and what you have worked on?*
    - *What is your interest in game audio?*
    - *Do you generally prefer experimenting with new methods or using existing proven techniques? And how do you go about deciding which methods to use?*
    - **Optional:** *Have you ever worked with horror games or other quick suspense type of scenarios, and if so, can you speak to this?*

D.3 About the Research

- General explanation of the research, using figures to help demonstrate the subject.
    - Research question
    - Hypothesis
    - The tool

D.4 Questions #1

- *Have you ever come across any of the problems described in the explanation of the research (in general, not just for risers)? Or is it something you have thought about before? And do you even see it as a problem?*
    - *If you have, how did you address it?*
- **Optional:** *How would you approach a build-up type of goal for audio right now? For the specific context and in general.*
- Procedural audio

    - *How would you define procedural audio and what makes it different from other nonlinear audio design techniques?*
    - **Optional:** *Do you have experience using any form of procedural techniques?*
        - *If so, what type? And why was it used?*
    - *When should and when shouldn't procedural methods be used and why? What are some of the limitations of procedural audio?*
- *What is your initial opinion on the research's approach to addressing the stated issues?*

D.5 Showcase

- Demonstration of the current prototype. Most of it is subject to change. Please feel free to ask any questions.
- *The prototype serves as a proof of concept.*
- *Do you have any experience with UE4?*
- Showcase the tool.

D.6 Questions #2

- *What is your first impression?*
- *Do you think this method has potential? If it were more worked out, would you want to use it over existing techniques? Could making more types of audio with this technique be used more in the future?*
- *Are there any cases in the past where you would have used this?*
- *What do you think about the chosen audio parameters to modulate and how they are modulated based on the game data? How would you improve or adapt the current system?*
- *Are there any changes you would make? Future developments*
- **For audio designers:** *Would this tool fit into your current pipeline and/or workflow?*
- **For audio programmers or leads/directors:** *Would this tool fit into the general audio design pipeline and/or workflow?*
- **Optional:** *Is the tool's interface easy (enough to) understand?*

D.7 Follow up

- *Would you be interested in me sending you the final paper or an update at the end of the project?*

Appendix E: Interviewees Background Information

E.1 Interviewee Experience Table

Interviewee	Time in game development	Role
Davis, Will	20 <	Audio lead/director designer
Duquesne, Clément	5 - 10	Audio designer
Fournel, Nicolas	20 <	Audio lead / director programmer
Fuesslin, Florian	15 - 20	Audio lead/director designer
Hays, Tom	20 <	Audio lead/director designer
Huguenard, Charlie	10 -15	Audio programmer
Smit, Tom de	> 5	Audio designer
Vera, Bogdan	5- 10	Audio programmer
Wijngaarden, Jonathan van den	10 -15	Audio lead/director designer

E.2 Interviewee Descriptions and Interest in the Field

Interviewee	Description	Interest in the field
Davis, Will	*Okay. I started doing audio for games with a bit of a programming background. Because back then you had to write your own drivers. To do that old chip stuff. Ended up head of audio for Codemasters. Over at EA. A senior audio director for Ubisoft at a head office in Paris. I have done, I stopped counting at 498 games. So, I have done a few. Most recently, I was head of production for NeoBards. […] Currently working on real-time audio implementation for movie work using Unreal.*	*I suppose I like that interaction between the various elements that make a final result. It is incredibly easy to get good audio. For a game, a film, it does not matter. You just give it to some people who have got a basic talent and you will have a good result with some time and some effort. But so, hiring good audio people does not actually mean you will get good audio, particularly in games or 'nonlinear' workflows. Because you have got to have the rest of the team also wanting it. It is far more important that everybody else wants good audio, or good visuals, or good gameplay, or whatever it is. The end result is the combination of everything else combined.*
Duquesne, Clément	*I am Clément Duquesne. I am a game audio designer and do a bit of programming also. And a bit of music also, but some. And I studied musicology at the beginning but with an interest in programming. Then I did a video game school in France called ENJMIN. I entered Ubisoft where I worked on the game audio systems of Ghost Recon Wildlands with Twan (referring to Twan de Graaf). What did I do other than that? Oh, yeah, I joined a small indie studio called Swing Swing Submarine for a while, then another one called the Pixel Reef, where I worked on the VR game Paper Beasts. And then I worked with friends at a small studio La Poule Noire, where I am still there. And now I do some freelance jobs on top of that.*	*Game audio is really the thing that makes the games alive. And that is what I love. And it is really the best of feelings when there is a game that exists for a few months, people are playing the prototype, it is fun, it works well and then you add audio. And then it is magic, and everything comes to life. And it is so great. So, that is what I love about this. And I also love the encounter between the creative artistic side and the technical constraints and possibilities of programming and runtime real-time environments. That does something really unique and captivating.*

Fournel, Nicolas	*My name is Nicolas Fournel, and I am mostly kind of audio programmer, or researcher, or something like that. I have worked pretty much all my life in the audio software industry. Before more for music, at the beginning of my career. Well actually, that is not even true. Mostly for sound effects, actually. And mostly for the gaming industry. So, I worked a little bit in France, where I am from, in a small company. Then I moved to California to work for Factor 5, which was a company working for Nintendo GameCube. So, I worked on video system of GameCube console. Then I moved to Konami, and then Electronic Arts, then Sony. And at all that time, I was working more or less on what could be called procedural audio. At different levels, but I was doing a lot of other things like AI, UI, whatever. And then, about nine years ago, I moved to Japan and I created Tsugi, which is a company which is really dedicated to procedural audio. So, until now, we were focusing mostly on the gaming industry. And now we are trying to derive that a little bit for all industries.*	*I think I have an interest in generative technologies, in general. But also in audio. That is why the two combined together. When I was a working at Sony, I was working on quite a few things. But the two main things I was working on were, in addition to the regular audio engine, where procedural audio and audio feature extraction. Which I found, can combine themselves pretty nicely if you want to do some synthesis, but you want to have some kind of feeling which is rooted in reality of more interesting sounds.*
Fuesslin, Florian	*Yes, I am Florian Fuesslin. I am audio director at Ubisoft Blue Byte in Dusseldorf. I started my career like 15 plus years ago, as an audio intern at the company Crytek. Before that, I had a 10-year background in producing and making music. So, piano and drums are my instruments, and I went to the school of audio engineering to get a diploma degree there. And on the way to the bachelor, I decided to do that internship, which turned out to be/ Yeah, I never went back to university. And then I went through all the ranks, like different design positions, then became lead and eventually director. And that was all at Crytek in Frankfurt. And then after this 15 years, I decided to 'new challenge new opportunity'. And to join Ubisoft. I think the cycle closes as well, when I am teaching at various universities and of course writing this chapter (referring to chapter written for a book on game audio programming).*	*I think it was really making music and sound effects and all that was a really big time of my youth that I spend on that. And yeah, producing various bands and playing in bands. And the other part was always playing computer games. Being born in the 80s means you had both worlds still right? The analogue telephone and then suddenly, digital stuff coming around, or the first home consoles. […] And part of my youth was really playing a lot of games as well. And then I always was interested in the sounds as well, I think with Warcraft 3, there was a map editor. And that is where I started to record and implement my first sound. It was a pretty simple like area trigger, like if the player passes that gate, then play like 'death to all humans', or whatever. […] You start with nothing. And you have an idea in your head, how the world could look like and how it could sound as well. And then you have to build everything from scratch. […] We start with nothing. We really have to imagine all these things and come up with all the details.*
Hays, Tom	*I was a musician as a kid and then got into turning knobs as soon as we got effects boxes. […] And went to audio engineering school then detoured off and got a degree in history at UC Berkeley because I cannot just think about one thing all the time. […] I got a job in […] it turned out that this job was for a dialogue editor. And we were editing dialogue on a super low level to fit it into data compression. [„„] And then this is mostly for talking watches and toys. But we wound up doing some work for Disney and Sega on some video games. So, I kind of got into that world and got hooked. And after a few detours, I*	*I definitely fell into it. But the thing is, I was developing an interest in computers. I came kind of late. I was not the guy like Will Davis, people like that, where I did everything I could to get my hands on whatever computer existed when I was 12. I was way more interested in playing guitar. But when I was in, in college later in my 20s, and when I got to an Atari 1040 and a Dr. T's KCS sequencer. Yeah, it has probably went up the market before you are born. So, that that really hooked me,*

	started freelancing and started a small business in 92. I got to do a lot of work at Broderbund. I did the sound editing for the PC port of Mist back in 93. And then worked on a bunch of stuff for a company called Rocket Science Games in San Francisco wound up as their audio director of 95-97. They went out of business, I got the first job I could find, which was audio director at Novalogic. Moved down to Southern California and worked on a bunch of war games and learned how to record weapons and tanks and stuff like that, which is a lot of fun. […] And beginning of 92 got an offer at Treyarch and went over there. Worked on the end of Spider Man, and then some less known games, shall we say. […] And yeah, then did Spider Man 2, Call of Duty two, Ultimate Spider Man. […] And then I got an offer to work the vendor side, building up the department at Technicolor. […] I did that until 2013, at which point I basically took our core team and started Rockets Sound, the company that I am running now. Like 90% of our work is, is voice stuff	*and I really enjoyed it. And so when, when I started to see these crossovers between, essentially computer science and audio and music, that that really sparked my interest. That is kind of what drew me into it. And the thing is, I love music, I love playing music, but I also kind of like tinkering. And I do have, I am not necessarily a math guy, but I have an engineering kind of mindset. Like fixing things and figuring out how to make things happen. And so, game audio just covered a lot of different areas in terms of things that I find interesting.*
Huguenard, Charlie	*So, I am Charlie Huguenard. My background is in computer music but I work in video games, or have worked in video games, and recently VR, for about 13 years. I started off as a note tracker, and a sort of audio designer for music games. And then, through that, kind of learned a bit more about game design, and a little bit more programming, and started working as a gameplay programmer or a game designer sometimes. And I spent about half of my career working as a gameplay programmer. But because my background was in computer music and sound, eventually I started finding opportunities to do more audio programming. And in the process as on, the chapters I have written for the Game Audio Programming books, I kind of found that I can combine the computer music background with the audio programming with the gameplay programming. And the sort of like the game design aspects of things to push forward a little bit more of the procedural sound and generative interactive music, nonlinear music, composition, sort of areas of things. So, that has my main kind of area of interest nowadays.*	*My background in music is actually jazz music. And then also, some punk and ska music and that kind of thing. But mostly jazz music, which is very improvisational. So, it was a lot of thinking on your feet, a lot of rewriting things. Or writing things and rewriting them, and then changing your mind and playing the wrong note and playing it again. All of that. And so, that is really what I bring to the sort of computer music area and the computer sound or interactive sound areas as well.*
Smit, Tom de	*I am Tom, I graduated from the […] University of the Arts Utrecht. I had a bachelor in sound design and received a masters that focused on nonlinear audio. And games as well, but specifically horror games and providing tension and in what ways you could go about that. After that, I have been freelancing, or actually, during my program at School of Fine Arts, I was already freelancing. Shortly after my graduation, I started working at Bohemia Interactive, for almost four years in Amsterdam as a senior sound designer on the Arma 3 Contact expansion. And as a sound designer on some DLCs, and the main product line of Arma 3 and all the trailers and advertisements that concern the production of a game. And after that, I started at PUBG, which I am now little*	*I did not play game one day and was like 'wow, this sounds so cool, I want to be a game audio designer'. No, I probably, at first, I noticed that the music industry was terrible in the Netherlands. So, I moved to media. Then I noticed that television media and the film industry is also, well a film industry is nearly non-existent in the Netherlands, and the media world is very difficult to get a foot in the door. Because many agencies have their people already. So, that takes a long time. So, what kind of was left at that point was game audio, games basically. And when I started looking into that, I noticed that that really interested me.*

	less than a year, in three weeks, I will be working a year, again as a senior sound designer. And for me, game audio, I have quite a specific view on that, which might not be shared in the same way by everyone. But I think it is very important and it is something that I took from the academy with me. And that is that game audio is really, the purpose of it is to support a storyline or a narrative, or the enforcement of emotions in players.	*The way how something creatively has so many connections into all these other disciplines. And the nonlinearity of it, obviously, is very important to my interest in the field.*
Vera, Bogdan	*My name is Bogdan Vera, and I did a couple of degrees in music tech at Bournemouth in the University of New York. Then I tried doing a PhD in electronic engineering at Queen Mary University. But I left that two years in, because I was really interested more in the industry stuff and less so in research. So, I have like a little bit of a background in machine learning from there as well, which I never really get to use. But then I ended up working at Media Molecule to basically develop a music and audio creation system for the game Dreams that we released last year. Which is basically this game where you can, inside the game, create other games, and also make music sculpt things, animations, make characters effectively. It is like a game engine within the game. And as part of that, I developed the sound engine, and the music and audio creation and editing tools. Mostly on the UI side.*	*I like game audio in general. Obviously, I have a have an interest in UX and tools, and interactivity. So, I am really big into like worlds where the player has an impact on the sound experience in some way.*
Wijngaarden, Jonathan van den	*I have been working in the games industry for 17 years, starting out as a composer. Then moving from only composition to also doing composition and sound design. And then being an audio lead and audio director. So, I have sort of had all of these roles in the past over the past decades. […] Right now, I am working with the Arizona team on After the Fall, which is going to be the spiritual successor to Arizona sunshine. And I have also been doing a lot of smaller projects. Doing a couple of mobile games as well as doing several national things for stuff like Rabobank. So, really a diversity of all kinds of projects. And as of recent also, because of working very closely with Vertigo, I have been branching more and more into VR audio and doing a lot more VR games.*	*It started with just being a gamer. I grew up on video games very early on. We would always have PCs in our household from a very early age. I remember being one of the only persons who actually had a dedicated PC in the household. So, I would start playing arcade rip-offs, like Pac Man on PC and Dig Dug and all those early games. And that sort of grew as a hobby and later on passion. […] I was listening to the soundtrack to Command and Conquer, which had, hip hop, heavy metal, orchestral music, all sorts of cool music. And I thought, wow, this is this is really cool. […] And then I started an educational course towards this direction.*

E.3 Interviewee Procedural Audio Definition Table

Definition of procedural audio	**Interviewees**
Adaptive audio	Will Davis, Florian Fuesslin, Tom Hays
High amount of adaptivity	Charlie Huguenard, Tom de Smit
Real-time (granular) synthesis	Clément Duquesne, Nicolas Fournel, Jonathan van den Wijngaarden
High amount of adaptivity or real-time synthesis	Bogdan Vera

Appendix F: Exerts from the Interviews

Exerts from the interviews quoted or mentioned in Chapter 4.

F.1 Davis, Will

F.1.1 Player predictability

The biggest problem in the game, and you mentioned it, is 'do not do what you want to do'. You start the trigger, and then they go 'oh look, shiny squirrel' and wander off in a completely different direction.

F.1.2 Tool integration

That is a that is a tricky one. Because if you were at Ubisoft, the first thing that you would do would be rebuild that tool inside of Anvil, or whichever game engine they wanted to use, and make it part of the larger toolset. Or actually, probably, you would be putting it on the list the tools group to do it. And then you would be a liaison from the game team and the tools group. Every company is completely different. Can I see individual sound designers using it? Absolutely. Because it can speed it up. It is simple. And actually, it does not require any effort to learn if you have already got a basic understanding of these things anyway. And most sound designers do. One would hope. Saying that, your tool is simple enough that you can just whack it on 'oh, I got some layers, whack in some sounds, change these things'. In five minutes of just playing with it, you would have a basic understanding of how it works. So, that is really nice. It is slightly faster than using blueprints or code or, a lot of work from a lot of sound designers would still be in their DAW. That is a quick way of hearing sounds together and how it is going to work. That is cool.

F.2 Duquesne, Clément

F.2.1 Description

I am Clément Duquesne. I am a game audio designer and I do a bit of programming also. And a bit of music also, but some. I studied musicology at the beginning but with an interest in programming.

F.2.2 Experimentation

Well, it really depends on the context of the developments, because sometimes you just do not have the time to experiment. And you have to find things that are already there. And you have to make with what exists. But I had the luck to work on several projects that had the time to find new solutions, and the resources to find them. So, on Ghost Recon and Paper Beasts it was the case. And so, I had the time to search for stuff, to test things. And yeah, I guess when I have the time, I try to find new solutions that work well with what we were trying to do. Because we are always trying to do new things.

F.2.3 Audio control

Just something else that I was thinking about is that as a sound designer, what would maybe prevent me from wanting to use this solution is, but it is maybe just something that is to come, it is that it is not open enough. It does only one kind of riser. When you are a sound designer and you are looking for ways to do something, you mostly want to control how it will sound, what sounds you will use. That is one of one of the things I would change.

F.2.4 Riser modulation

For a riser I find that in the last seconds it does not rise enough. […] Because I guess the moment where your risers stops is really entirely distance base

F.2.5 Viability

I am so familiar with Wwise that I think I would still go to Wwise for something like that. And also, I did not have many risers in my games. So, I am not thinking, 'Oh I could have used this at this moment or this moment'. In my personal experience, I would not have used it, I think.

F.2.6 Programming competence

It was done by me (referring to the audio integration in code for Paper Beasts). I had the responsibility of the programming as well. That is also one of the reasons that I am not very attracted to specific audio middleware, is that I have the competence to do audio tools that answer to specific needs. I am tempted to do this myself if I have to.

F.2.7 Viability

…I am not sure risers, I mean apart from horror games, I am not sure it is a thing that will happen a lot in terms of quantity. I think you can still find the time when you need to. It is a

thing I often find with small procedural tools like this, is that if it is not important for your game, then you do it in another more simple way or you just do not do it if it is not that important. And if it is really important, you want to do it your way.

F.3 Fournel, Nicolas

F.3.1 Procedural Audio

Well, I guess that is real-time synthesis. Real-time sound synthesis based on input from the player or the game. So, something that reacts in real-time and generate audio from that. […] In my work, it is always synthesis. Then, my definition of procedural audio is really straight and limited to generating something from scratch. Most of the time. I totally understand that other people call procedural audio some kind of rules that you follow to create interactive music, or some kind of scripting and they call procedural techniques. I am totally fine with that (laughs). I do not mind. But my definition in my work and for when I talk about procedural audio, that is sound synthesis. And now I am a bit less strict. But even before, I really use no samples at all and totally generated.

F.3.2 When should procedural audio be used?

Well, I think in a lot of cases, you might want to use both. First its limited, again, by the CPU usage you can have. So, in most cases, you probably want to use procedural audio, maybe if you want more interactivity for your sound. Let us say you have a fight between dragons, and your dragon is just in front of the screen, in the middle of the screen, just in front of the player. Obviously, you want the roar to follow the movement of the throat of the dragon and everything. You will want that to have like, full control, full procedural audio. If there is another player's dragon very far away, you do not need procedural audio for that, you do not even see very well the thing. You can play a sample or a couple of fundamental samples. So, that is one thing. I think there are cases where you do not necessarily need procedural audio, which are/ I am not even sure how to say this in English. Citations, like when you refer to something which is very well known. A very typical sound for example, you have a fighting game, and you have a very well known, I am really bad at weapons, very well know AK or something. So, in that case, usually players really like to observe the exact sound of that weapon. And in that case, it is a short sound, there is not so much control. You really want that particular sound. So let us go with samples right. Again, I am talking about sound effects right. If you cannot obviously, because of CPU reasons, then you have to choose your battles. Pick up where you can use it. And if you really need some kind

of station effect something like really/ Well some people will say also, for something realistic, but I do not necessarily/ Yeah, depends on what the level of the procedural audio modal you have, right of what levels they have. Of course, if you need something very realistic and you do not have access to very good models, then obviously you should use samples.

F.3.3 Tool integration

Interviewer: *So, this type of tool, do you think it could fit in the current pipeline and workflow of audio design?*
Fournel: *Yeah, I think so. If it saves time to people. People do not have a lot of time. After that, if you decide to make several, then you kind of need to send something over that, so you do not ask people 'okay, this is this one and this one does that' As long as there is a uniform user experience, way to set things up. Yeah, I think that is interesting.*

F.3.4 User control

I think where you want to give options is more on the creative side and less on the implementation side. If you can have the implementation go very quickly, easily like, just dropping a box, attaching something. That is great. And nobody will ever complain, because they have more creative options.

F.4 Fuesslin, Florian

F.4.1 Issues regarding game audio development

I do not think it is a problem, it is just the reality we work in. Without the players we do not have a job. And the player is the disturbing factor in our world. There is nothing we can do about it, rather than in the end, it is a lot of psychology. Using old tricks to steer players' attention where we want it to have it. To design the levels that the player will most likely use that path. And then maybe using that, let us say golden path as it is usually mentioned in games, to use that golden path, and design around that. Of course, if there is the most important emotional cutscene, where your mom dies, and the player decides to look at the feet or totally away from the explosion. Cool guys never look at explosions, right? So, in the end, we cannot do anything about it. But we can do the setup that it is very, very likely that the player will not miss out that scene. Or you have to do a cutscene. But again, if you are in the interactive medium, the last thing you want to do is taking away the controls or the camera, especially if you talk virtual reality games and with a headset.

F.4.2 Simplicity of sound design

Sound design is sometimes pretty simple. It is time, frequency is very important. And if you design sound, people always say 'more plugins'. But in the end, it is like 'I use pitch, volume, maybe reverb'. The standard feature set EQ compression that is way more than you usually need. So, I agree let us not make it overly complicated. Especially as it seems to work and you achieve the feeling or the emotional point you wanted to make.

F.4.3 Parameters & modulation

I think it is a good start. Same like, I send you the link, they also use a lot of distance and that works fine. Because what you know, you know the max speed of the player, you know the angle the player's turning. And by this you always know 'oh, okay, it will definitely, I do not know, 10 meters per second is max speed'. The player will always take two seconds from this position to hit the impact moment. And that is cool, because you can well, you can even choose the right piece then. If you want to play a different audio based on this. But I think distance already works perfectly fine.

F.4.4 Player predictability

I think it is project based. But in this situation, it works really well. Like you basically come around the corner and you see that point of interest That is guiding. Well, okay, there is nowhere else to really go right. You could turn around now and walk to the end of the street, but there is an invisible wall, that is blocked. You will eventually turn and go towards the church. But as you do, the riser will give you direct feedback about 'something is about to happen'. This is a psychology thing, but I guess you would not stop pushing forward with the controller. Because you will continue. The building is interesting, how it is burning, it triggers all these questions. And then the riser is basically supporting your approach to the impact moment. And that is perfect.

F.4.5 Tool integration

It is a standalone thing, right? It connects with the different engines. And I think this is where the beauty is, this is where the middleware have really proven that they worth getting. And they all have a plugin pipeline and architecture. So, you could easily make a Wwise plugin out of that, right? It is not a big deal. Then it is just a matter of, let us say, good software design.

F.5 Hays, Tom

F.5.1 Procedural audio

I actually see it as a continuum. I do not think there is one or the other, I think that well executed triggering of individual, one shots and stereo sounds and music streaming, it is procedural. It is almost like granularity is a continuum where on the one end you might have a single sound for a gunshot or for a rock fall, and then at the other end you might have sounds that tie into every little, tiny interaction of the rock as it rolls down the hill. Because it is one of those things where you can drive yourself crazy thinking about how many different sounds are being made when rock rolls down the hill. And so, it is all procedural in that way. Now, of course, if you are synthesising the sounds, and the end result, or the hundreds or 1000s of sound grains that go into the rock rolling down the hill, then/ I guess that there is a line that you are crossing when you are going from sample based into truly synthesised, but I do not know. I tend to see the whole thing as a continuum. And it may be that there is some kind of point where there is just a paradigm shift, and it is completely unlike anything you have seen before, but I guess I do not know where that is. I have been surprised before.

F.5.2 Tool integration

There is a sort of philosophy and mindset behind it, it has a little different from the tools that you might have right now. Say we are trying to exactly what you are doing, but using say existing Wwise, or whatever else. I think you could come close, but its underlying intent is different.

F.6 Huguenard, Charlie

F.6.1 When should procedural audio be used?

Well, one answer I think I agree with is whenever you want. Do whichever one you feel like doing. Because some people do enjoy working in sort of linear production techniques more than then in designing systems or designing procedural systems and vice versa. And that can have a huge impact on the outcome. Because if somebody is really into their method, and they are good at their method, then it is going to sound better. So, that is one answer. The other answer, though, is that I think that for me, I go for procedural solutions first. And the reason is that they are always more flexible. The downside is that they blow up easily. There are edge cases to deal with and stuff like that. And going with sort of like linear

production techniques, and then applying some interactivity to those resulting sounds can be a lot more stable and predictable.

F.6.2 Parameters & modulation

They all make sense to me. The abstraction of position and in motion or action, that makes sense. As opposed to tying it to like, the 3d position, for instance. Because I could also see some instances where maybe you want to use this system for something that is not actually about player motion. Say you have a relationship meter or a relationship mechanic, and you have some sort of relationship value with another character, and you want that to build. You can apply that to this. So, makes sense to me.

F.6.3 Tool integration

It depends. Sometimes if a project that is using your thing is specialised enough and specialised in favour of the thing that your tool uniquely does, then it can be a conversation about using that instead of Wwise or using that instead of FMOD. Where it starts to get a little difficult to sell, is when the project calls for a lot of things that one of the more widely used tools does really well. Like say, dialog localization, or being able to batch process large numbers of sound effects, or that kind of thing. That is when it gets a little trickier. And running side by side if possible, becomes a good option. Or if it is possible, plugging your tool into that system. And I have definitely considered doing the third option, like plugging one of my music system tools into Wwise, instead of trying to get people to use it instead of Wwise. But you know it is definitely a consideration. And it is not the most interesting part of the work, but it is a huge part of the work.

F.6.4 Viability

It has potential? Yeah of course. Is that a trick question? No, I mean, as you know by now I am a big proponent of procedural sound design and system thinking and sound design. To me this is just a thing that we definitely should do. And we should do more of it. And have more tools like this. Because, I think I kind of alluded to it earlier, the sound designers, the purely design side of the people that I have worked with, they already want this stuff. And they already know how cool it could be. But they do not spend a lot of time thinking about it, or asking for it, because they think that it has too hard. So, if more people are making tools that prove that it has not too hard, then they are going to get used.

F.7 Smit, Tom de

F.7.1 Experimentation

I like to try new things. Provided there is time. There might not always be time. My experience until now is that introducing new technology usually comes with the necessity for extra support. So, this is something that you would want to time in the onset of development. For instance, during preproduction or if there is an allocated period within production that you have discussed with management 'we will take this time just to investigate this specific piece of technology'.

F.7.2 Centralisation to Wwise

Say my workflow currently is Unreal Engine and Wwise. If I were to use this tool, I would have to in Unreal is what I understand. Is that correct? […] I am asking because, what I prefer most is to have most things centralised. I would probably ask if this could be converted into a Wwise plugin. Because I have Unreal on one screen, Wwise on the other, and then my loudness meters on the third one. And I do not know if I would have space for yet another application that would kind of battle with the other applications for focus. Because Windows is really annoying like that. That would be maybe a reason that I would say lets put it either in Unreal, but preferably in Wwise as a plugin. Because that is already compatible with Unreal

F.8 Vera, Bogdan

F.8.1 Procedural Audio

Do you mean procedural audio synthesis or procedural audio arrangement? […] I mean, arrangement would mean generative composition, sort of. You put together segments of music or sounds in some way, procedurally. As in like, according to some rules automatically. Procedural synthesis would be, instead of going to record a sound file, you generate it from some algorithm. I think those things are very different. When it comes to procedural synthesis. I think it is very interesting. And I have tried in the past to propose it to people at Media Molecule for example. And we did we did try for a while. And to some degree, you kind off can do that in Dreams now. But there is a tension when it comes to sound design people not being in control of the same parameters they are used to. They know how to go record the sound they want. You have to teach them a whole new technical

skill when it comes to procedural. It is almost like a procedural sound designer is a different role than a sound designer.

F.8.2 Risers

Think of it like the riser itself does not have to be just the sound that is rising, the sound effect that is applied. It is the gesture of the riser that can be applied to many things. It can be applied to the lighting in the scene. Another team might decide that fog comes in at the same time, you could tie that in the riser. It is the same kind of parameter. If you make it work together with the rest of the game design. Maybe the lights get darker in anticipation. In Dreams, we have often implemented this linkage between sound and the rest of the rest of the tools.

F.8.3 Player predictability

The thing is, players do not really do that (referring to unpredictable player behaviour). It is like, if they are trying hard to break your riser, it is their fault. It is a video game. The average person really will not. Because it is a video game anyway. I can go really close to a texture and look at it and it is going to look like it is low res, because it is a video game. So, it is like 'do not look at it so close'. Same with audio, the experience for you is, even though it is dynamic, it is still going to be a little bit railroaded because there is a planned event. That is fine.

F.8.4 Procedural audio design

Arrangement would mean generative composition, sort of. You put together segments of music or sounds in some way, procedurally. As in like, according to some rules automatically. Procedural synthesis would be, instead of going to record a sound file, you generate it from some algorithm. I think those things are very different. When it comes to procedural synthesis. I think it is very interesting. And I have tried in the past to propose it to people at Media Molecule for example. And we did we did try for a while. And to some degree, you kind off can do that in Dreams now. There is a tension when it comes to sound design people not being in control of the same parameters they are used to. They know how to go record the sound they want. You have to teach them a whole new technical skill when it comes to procedural. It is almost like a procedural sound designer is a different role than a sound designer.

F.9 Wijngaarden, Jonathan van den

F.9.1 Viability

I think it has potential. It is an interesting idea. It might have to tackle a few hurdles, but I think you are already thinking in the right direction. There are a couple of initial issues that are there that you seem to be aware of.

Appendix G: Coding Handbook

An overview of the coding used to analyse the data, as described in Chapter 4.2.

Code	**Subcode**	**Definition**
Background Information Interviewee		
Career length in games		Amount of time interviewee has been in the game industry (divided in groups of 5)
General description		Description of interviewee of themselves as professional
How to address the stated issues		The interviewee's take on how to address the issues the research states on game audio development
View on stated issues	Agree	Interviewee relates to the stated issues the research states on game audio development
	Would not call it a problem	Interviewee states they would not call the stated issues a problem
View on experimenting		Interviewee's view on experimenting
Interest in the field		Interviewee's interest in the field
Role	Audio programmer	Interviewee is mainly experienced as audio programmer
	Audio designer	Interviewee is mainly experienced as audio designer
	Audio lead/director programmer	Interviewee currently has a leadership role and before this mainly has experience in audio programming
	Audio lead/director designer	Interviewee currently has a leadership role and before this mainly has experience in audio design
Parameters & Modulation		
Modulation	Player predictability	Does and should the method address extremely unpredictable player behavior
	Prototype's modulation does not suffice	The prototype's modulation based on the game's state and events does not suffice
	Prototype's modulation suffices	The prototype's modulation based on the game's state and events suffices
	Pathfinding	The prototype calculates a straight line, and does not account for corners
Parameters	Prototype's parameters suffice	The prototype's parameters are indicated to work well and suffice as the basic parameters.
	Prototype's parameters are not enough	There are more parameters to add for it to suffice.

	High number of possibilities	Interviewee stated that there is a high number of potential parameters to add to the prototype
	Riser can also be adapting other audio	A riser can also be the adaption of already playing sound
Pipeline & Workflow		
Integration approach	Centralise to middleware	Interviewee mentions it would be preferred to have the method integrated in a currently standard middleware.
	Centralise (to middleware or engine)	Interviewee mentions it would be preferred to have the method integrated in a currently standard middleware or game engine.
	Prototype's approach works well	The prototype's approach to integration works well (does not need to necessarily be centralised or adapted in another way)
	Advantages and disadvantages with different approaches	Advantages and disadvantages of different approaches to integrating the prototyped system in the workflow discussed or compared.
Provided controls	Different type of control	The tool provides a different type of approach than sound designers are used to.
	Sound designers prefer traditional control	Full control over the sound design of the audio is preferred
Other future developments		Other future optimalisations to the prototype's approach to workflow.
Procedural Audio		
Advantages and Disadvantages		Advantages and Disadvantages of procedural audio, using the participants definition.
Experience		Experience of participant with procedural audio, using their own definition.
Definition	Adaptive audio	Procedural audio is audio that adapts to the game.
	High amount of adaptivity	Procedural audio is a certain threshold of adaptivity.
	Real-time (granular) synthesis	Procedural audio is real-time generative audio.
	High amount of adaptivity or real-time synthesis	There are two types of procedural audio: a threshold in adaptivity and real-time synthesis.

	Unsatisfied with definition	Interviewee is unsatisfied or unsure of their definition.
Industry Application		Are procedural techniques underused or overused, and why?
User Interface		
Understandability		The understandability of the interface of the prototype.
Improvements		Potential improvements to the UI mentioned.
Viability		
Method potential	Method has potential	The proposed method is stated to have potential to be of practical use in the field.
	Potential is project dependent	The interviewee states that its potential might differ per project.
	Addressing reactiveness allows for more linear-type audio	The interviewee states that by addressing reactiveness the method can allow for more linear audio design type techniques.
	Method does not provide enough to be used over middleware	The proposed method does not provide enough to be used over middleware
	Method gives up too much control	The proposed method gives up too much control
Use cases	Potential future use cases	Interviewee provides example(s) of potential future use cases for the tool
	Use cases in past projects	Interviewee provides example(s) of past use cases for the tool
Requirements		Requirements for the method to be viable
Application of method on other audio effects		Interviewee discusses if the method is applicable to other types of audio effects
Other Relevant Data		
Industry relevance		Industry relevance is discussed
Meaning of 'nonlinear'		The definition of nonlinearity is discussed
Horror genre		The horror genre is discussed and what makes it different, if it is different, from other genres.

Bibliography

Adam, T. (2014). Procedural Music Generation and Adaptation Based on Game State. The Faculty of California Polytechnic State University

Aliakseyeu, D. (November 2020). User Experience Evaluation in Industry. Guest lecture presented in Breda University of Applied Sciences, Breda

Baggström, M. (2019). Sound Design – Create Riser FX for Transitions. Retrieved November 2020, from https://www.skillshare.com/classes/Sound-Design-Create-Riser-FX-for-Transitions/1618082930

Beauchemin, S. (2019). A Rare Breed. In G. Somberg (Ed.) Game Audio Programming 2 (33 - 41). Taylor & Francis Group

Böttcher, N., Serafin, S. (2014). A Review of Interactive Sound in Computer Games: Can Sound Affect the Motoric Behaviour of a Player? In K. Collins (Ed.), B. Kapralos (Ed.), H. Tessler (Ed.) The Oxford Handbook of Interactive Audio. Oxford University Press.

Burls, A., Gray, D. Kogan, M. (2014). Salutogenisis and coaching: Testing a proof of concept to develop a model for practitioners. International Journal of Evidence Based Coaching and Mentoring, Oxford Brookes University. Retrieved June 2021, from https://search.informit.org/doi/abs/10.3316/informit.705249145757477

Dijk, J. van, Koning, S. de (2020). Project Rookery Trailer. Retrieved January 2021, from http://sdkoning.com/PF/RookeryTrailer.html

Freeland, R. (2014). Serialism & Sonification in Mini Metro. Retrieved October 2020, from https://www.youtube.com/watch?v=FgV4hSfsl00&t=37s

Gaiduk, D. (2019). How to Do Usability and UX Testing for Mobile App. Extracted October 2020, from https://medium.com/uxreality-blog/how-to-do-usability-and-ux-testing-for-mobile-app-211f92f3cd6d

Gordon, M. (2017). DOOM: Behind the Music. Retrieved October 2020, from https://www.youtube.com/watch?v=U4FNBMZsqrY.

Huguenard, C. (2019). Note Based Music Systems. In G. Somberg (Ed.) Game Audio Programming 2 (307 - 331). Taylor & Francis Group. P. 308

Intelligent Music Systems (2016). Introducing the Dynamic Percussion System Powering Music for Rise of the Tomb Raider. Retrieved November 2020, from

Jolly, K., McLeran, A. (2008). Procedural Music in Spore. Retrieved October 2020, from https://www.gdcvault.com/play/323/Procedural-Music-in

Kohata, Shuji (2014). An interactive Sound Dystopia: Real-Time Audio Processing in NieR: Automata. Retrieved October 2020, from https://www.youtube.com/watch?v=BrUQdd96qzk&t=28s

Lopes, P., Liapis, A., Yannakakis, G. (2015). Targeting Horror via Level and Soundscape Generation. Institute of Digital Games, University of Malta. Received November 2020, from https://www.semanticscholar.org/paper/Targeting-Horror-via-Level-and-Soundscape-Lopes-Liapis/8131268f835035850cc997df45949c9e940327e0

Koning, S. de (2019). Composing for Games as a Film Composer. Retrieved October 2020, from https:// sdkoning.com/PF/ComposingforGamesasaFilmComposer.html

Koning, S. de (2021). GitHub editor tool project repository. Retrieved January 2021, from https://github.com/StijndeK/RTR

Koning, S. de (2021). GitHub UE4 plugin project repository. Retrieved January 2021, from https://github.com/StijndeK/RTR_UE4Integration

Koning, S. de (2021). RTR Mock Showcase. Retrieved January 2021, from http://sdkoning.com/PF/RTRMock.html

Koning, S. de (2021). RTR Tool Showcase. Retrieved January 2021, from http://sdkoning.com/PF/RTRTool.html

Koning, S. de (2021). RTR Showcase in-game. Retrieved April 2021, from https://www.youtube.com/watch?v=pbYCn5AFTOo

Lamperski, P., Tahouri, B. (2016). Real-time Procedural Percussion Scoring in 'Tomb Raider's' Stealth Combat. Retrieved October 2020, from https://www.gdcvault.com/play/1023215/Real-time-Procedural-Percussion-Scoring

Rapan, E. (2018). Shepard Tones and Production of Meaning in Recent Films: Lucrecia Martel's Zama and Christopher Nolan's Dunkirk. In D. Power (Ed.), S. Deutsch (Ed.), LK. Sider (Ed.) The New Soundtrack (135 - 144). Edinburgh University Press. Retrieved November 2020, from https://www.researchgate.net/publication/327410211_Shepard_Tones_and_Production_of_Meaning_in_Recent_Films_Lucrecia_Martel's_Zama_and_Christopher_Nolan's_Dunkirk

Sweet, M. (2016). Top 6 Adaptive Music Techniques in Games – Pros and Cons. Retrieved April 2020, from https://www.designingmusicnow.com/2016/06/13/advantages-disadvantages-common-interactive-music-techniques-used-video-games/

Varga, A., Woldhek, A. (2014). The Next-Gen Dynamic Sound System of Killzone Shadow Fall. Retrieved October 2020, from https://www.gdcvault.com/play/1020559/The-Next-Gen-Dynamic-Sound

Vernooij, E., Orcalli, A., Fabbro, F., Crescentini, C., (2016). Listening to the Shepard-Risset Glissando: The Relationship between Emotional Response, Disruption of Equilibrium, and Personality. Frontiers in Psychology. Retrieved November 2020, from https://www.frontiersin.org/article/10.3389/fpsyg.2016.00300

Walder, C. (2019). Synchronizing Action-Based Gameplay to Music, in G. Somberg (Ed.) Game Audio Programming 2 (332 - 347). Taylor & Francis Group

Weir, P. (2017). The Sound of No Man's Sky. Retrieved October 2020, from https://www.youtube.com/watch?v=zKJ_XuQjjiw

This template is based on a template by:
Steve Gunn (http://users.ecs.soton.ac.uk/srg/softwaretools/document/templates/)
Sunil Patel (http://www.sunilpatel.co.uk/thesis-template/)