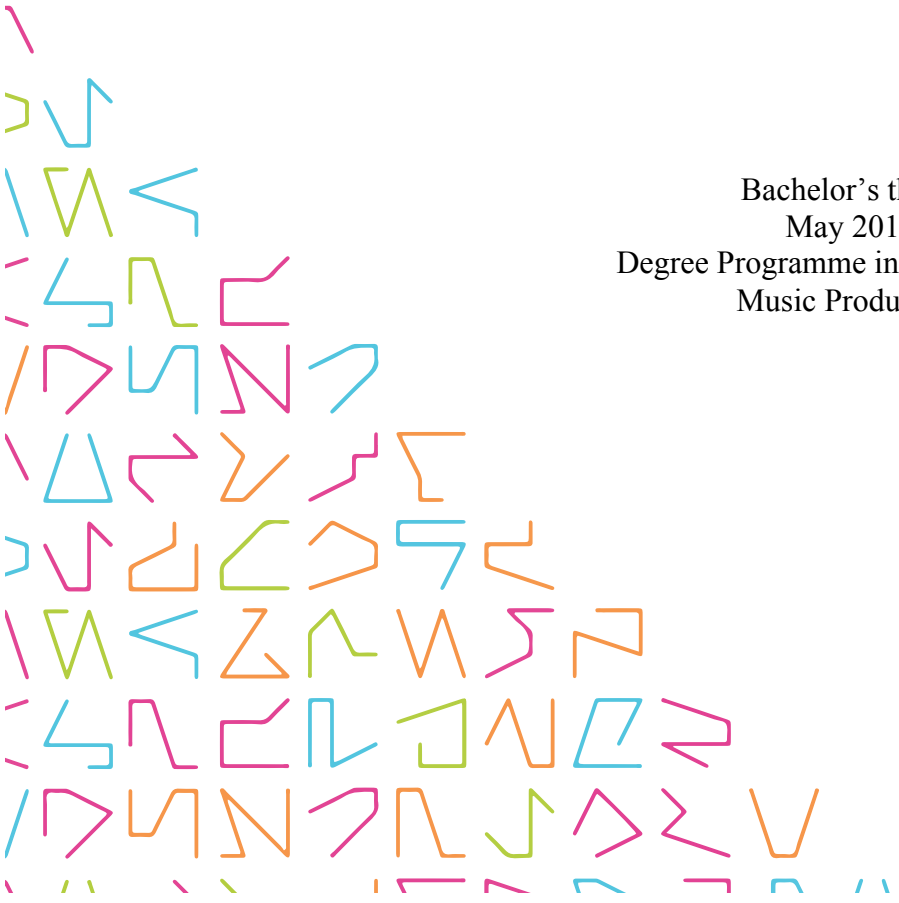
TAMPERE UNIVERSITY
OF APPLIED SCIENCES

PRODUCING ADAPTIVE MUSIC FOR NON-LINEAR MEDIA

Lassi Kähärä



Bachelor's thesis
May 2018
Degree Programme in Media & Arts
Music Production

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Media & Arts
Music Production

Lassi Kähärä:
Producing Adaptive Music for Non-Linear Media

Bachelor's thesis 42 pages
May 2018

As non-linear media such as videogames and interactive applications are getting more popular each year and interactivity is being implemented into other media, a need for non-linear adaptive music is growing. Traditionally a linear medium, music, needs to adapt to the actions the user decides to take in these interactive experiences. Adaptivity has been a vital part of videogame scores for years, but the techniques are shifting as new ways to compose and implement music for videogames are being developed constantly.

In the thesis history of adaptive music, techniques non-linear music composers, adaptive music techniques and means of implementation were discussed and compared with examples. A comprehensive look into the various possibilities of composition and implementation of score music for videogames was given.

A score for a virtual reality puzzle game was composed and implemented utilizing Ableton Live and Audiokinetic Wwise software to create an immersive and adaptive score. The combination of Live and Wwise worked well together, supporting the adaptive fashion of the game score. Using Wwise was a learning experience and through experimentation and testing an interactive music system suitable for the project was created.

Key words: adaptive music, videogame music, interactive music, audiokinetic wwise, ableton live

CONTENTS

GLOSSARY

DAW	Digital Audio Workstation
VR	Virtual Reality
EQ	Equalization
SFX	Sound Effect
Low Pass Filter	An audio processing method where the higher frequencies are filtered out and lower frequencies are passed through the filter.
RPG	Role playing game, a popular genre of videogames
BPM	Beats per minute

1 INTRODUCTION

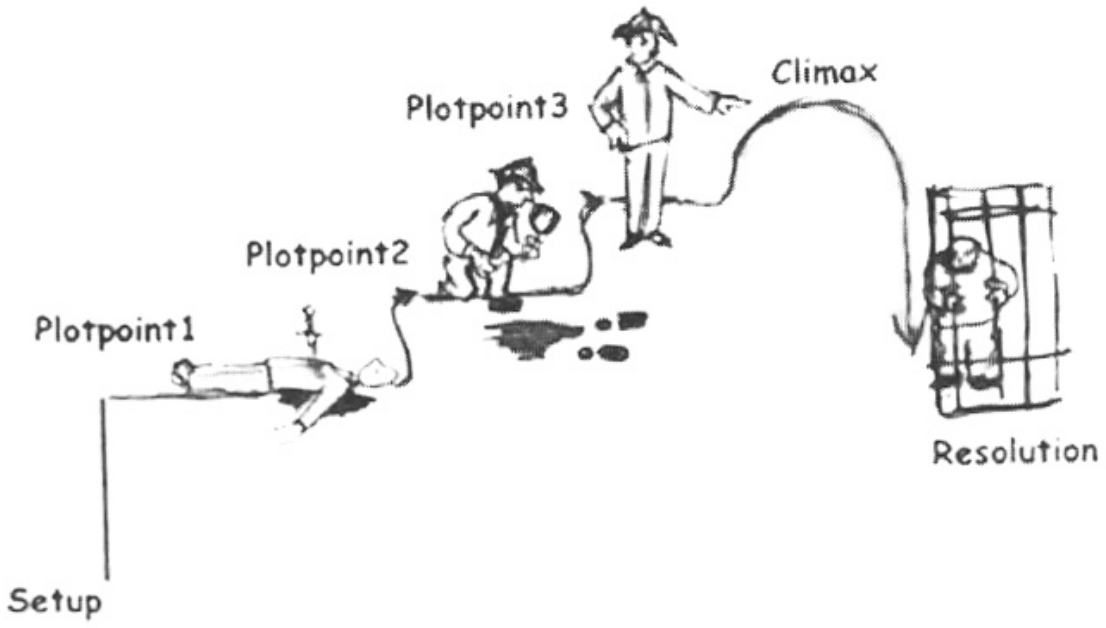
In addition to traditional linear media such as movies and TV-shows, non-linear media such as videogames and virtual reality applications are evolving at a fast pace. These non-linear media experiences need music that interacts with the user and the experience. The composition process of the soundtracks for these non-linear experiences differs vastly from linear musical experiences and requires new techniques and applications to work well. Videogames have already long been developed with a score that adapts to the actions in the game, but new techniques and possibilities are constantly being developed. The interactivity of a videogame soundtrack is a big part of the experience and can even be considered the most important part for some games.

The objective of this thesis is to cover various adaptive music techniques, adaptive music history and ways to compose and implement music for modern videogames. Different adaptive music techniques and their uses are discussed and compared with examples. The advantages as well as the problems concerning adaptive music composition and implementation are discussed.

An interactive music system and a score made for a virtual reality puzzle game is used as the practical part of the thesis. The planning of the project is explained and reflected upon. Original score music was composed inside Ableton Live utilizing various virtual instruments and later implemented into the game utilizing Audiokinetic Wwise software. The various ways to approach the interactive music system inside Wwise are explored and discussed in the thesis.

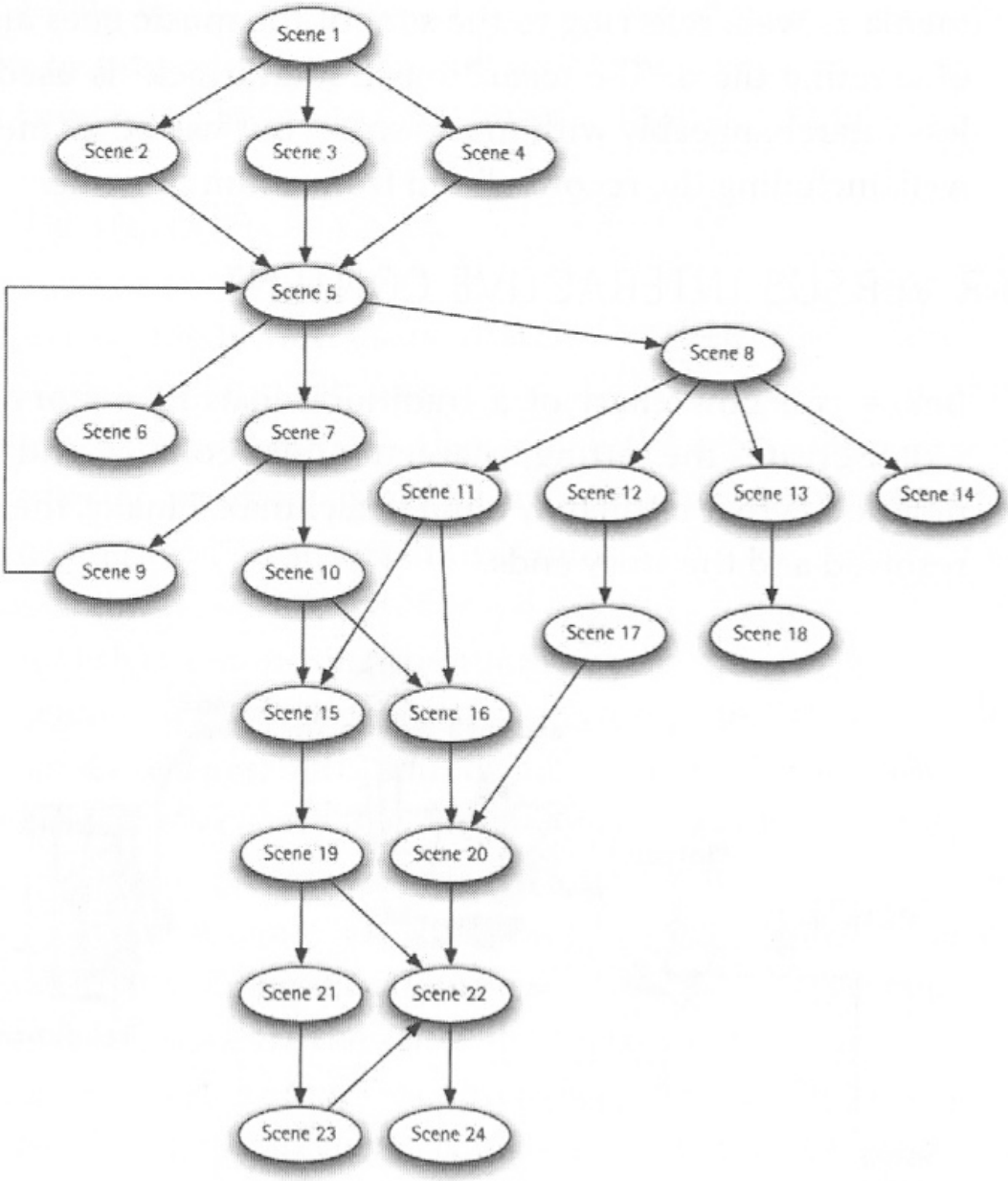
2 COMPOSING MUSIC FOR NON-LINEAR MEDIA

Music albums, songs and movie soundtracks are all linear musical experiences. This means when you start to listen to a song or watch a movie, the audio starts and plays until the end or until you decide stop it. You enjoy the music as it is and it is not interactive in any way. It also sounds the same every time it is played. A song by Depeche Mode, a Mozart symphony and the score for Apocalypse Now are all examples of linear musical experiences.



Example of a linear storyline. (Hoffert 2007, 9.)

Producing music for video games and other interactive experiences differs vastly from writing linear musical scores. When composing score music for a movie the composer knows exactly how the story is going to develop and at what pace. In most modern video games the storyline is chosen by the player from what happens and when to the pace of things happening in the story. As Paul Hoffert writes: “Videogame designers often speak of each player experiencing a unique movie, and that means each user must also experience a unique music soundtrack”. (Hoffert 2007, 8.)



Non-linear stories such as videogames can have multiple branching storylines and scenes. (Hoffert 2007, 10.)

2.1 Non-linear composition techniques

In movie scores the music moves forward seamlessly with the picture, what happens one time will never happen again and the story moves on. In videogames a player could explore the same area for hours before moving on to a different area or part of the story. After that the player could go back and move to a different part of the story and so on. This unpredictability creates obvious obstacles for composers. How to compose re-memberable, striking music that always fits the situation and mood without sounding too repetitive and still interesting throughout a 60-hour experience?

In addition to obvious differences in musical styles in videogames, there are also big differences in the way the music is handled in videogames. When scoring for a videogame it is essential to plan carefully beforehand. It is hugely important to know how the music needs to act in the game, as there is infinite ways music can adapt to action in a videogame. A puzzle mobile game soundtrack is usually very different to an open world RPG (role playing game). As Louis-Xavier Buffoni writes in his 2018 article composers need to mainly keep two things in mind when composing for videogames:

1. What elements of game play have an influence on music?
2. How music needs to adapt to these elements?

(Buffoni 2018.)

2.1.1 Looping

Maybe the most obvious obstacle in non-linear music composition is the need of looping. Most game soundtracks require looping as a player could be within the same area or part of the game for as long as they desire to. Because of the need for looping musical cues composers need to plan and take that limitation in consideration already at the beginning of composition. (Martins 2012.)

Creating interesting loops that can be played almost endlessly is a difficult task. Musical phrases such as melodies and chord progressions can get repetitive and boring very quickly. The loops composed for games need to be either long enough to not get repetitive quickly, or monotonous and subtle enough to not get irritating. If you compose short musical cues, these must be followed quickly by another cue or blended and layered with other cues. (Martins 2012.) I will talk about adaptive music techniques to overcome these obstacles.

When working with loops the composer needs to make sure that the decays and reverb tails that might occur at the end of the loop are not abruptly cut off. In modern DAW's such as Ableton Live an option to render as a loop is provided. The DAW moves the tails of the reverb into the beginning of the loop to avoid choppiness. (Sweet 2015,

136.) In audio middleware such as Wwise it is also possible to play these tails only when transitioning to a new musical cue, which makes the transition more seamless.

Loops usually end and begin on downbeats, because it makes counting and synchronizing simple. However a composer could want their loop to begin with a pickup note or a small phrase at the end of the loop. This is also possible to configure in an audio engine such as Wwise. A loop could begin with a one beat pickup but when it loops again, it would start on the first beat of the loop, not the pickup. (Sweet 2015, 135.)

One technique of looping to avoid repetition is creating asynchronous loops. Think of two synth loops playing on top of each other. If these loops would both be in 4/4 time signature they would quickly start to sound monotonous and worn out. By changing the other loops time signature to 5/4 or 3/4 polyrhythms are created. These rhythms can go on for very long before starting to sound repetitive. By triggering multiple loops in different time signatures at different times, interesting results could be achieved. This type of polyrhythmic variation can easily lead to a crowded soundscape, but with careful use can act as a powerful tool for variation. (DeSantis 2018.)

2.1.2 Tempo & rhythm

The tempo and rhythm are an important factor to take in consideration when composing music that will be transitioning and shifting all the time. If all the music is composed in the same tempo, it makes it easy to transition from a musical cue to another seamlessly. However this might not be necessary or suitable for all situations. A composer could go with a more ambient approach, which is not necessarily tied to a tempo at all, and doesn't need to be either. (Rohrmann 2016.) Even silence can work as a powerful transitional tool. In Hearthstone developed by Blizzard the music, which plays during a match simply ends when the song ends and for a while there is only silence. In most cases it would not necessarily work, but in Hearthstone it fits right in as the gameplay is slowly paced and other audio such as SFX and dialogue fills the soundfield. (Hearthstone 2018.)

2.1.3 Harmonics

In addition to rhythm and tempo, yet another thing composers need to be aware about is the harmonic framework of the score. In linear music composers can go from key to another easily and change the harmonic framework of the score around according to the changes in the experience they are writing music for. In games the composer needs to be aware of the transitions and way the music is going to be played, so that there is no abrupt changes in keys or the music doesn't begin to sound too monotonic. (Sweet 2015. 130.)

Before starting to compose the music for the game there needs to be spotting and planning of what kind of music is needed and where, and how often the score is going to be changed around and how. All this depends on the type of the game and how the music should be fitting in and enhancing the gaming experience. If the game simply transitions horizontally from a musical cue to another, a key change can be a good choice. Changing a key from minor to major in a suitable spot can act as a powerful tool to switch the mood of the player. This way the score doesn't get repetitive or too monotonous too quickly. However if the game needs quick changes in music, a vertical remixing approach could be more suitable. In this case all of the music needs to be in the same key or harmonic framework to layer well on top of each other. (Sweet 2015, 151.) I will talk more about horizontal and vertical adaptive music techniques later.

3 HISTORY OF ADAPTIVE MUSIC

3.1 What is adaptive music?

In the previous chapter I talked about the differences of linear and non-linear music. In most modern videogames music is non-linear and adaptive. In video games music is an interactive and dynamic element, as the music adapts to user inputs in the game. (Buffoni 2018.) This technique has many names; adaptive, interactive, non-linear or dynamic music. I chose to use the word adaptive, because it describes well how the music adapts when the game changes according to the story and the actions the player takes. (Velardo 2017.)

A common example of adaptive music in modern video games can be found in many action games. When a player is exploring the map, an exploration theme plays. This can be a fairly calm, ambient track that blends well into the environmental sounds and is not too distractive to the player. However when a player engages in combat the music will change according to this cue. This transition can be done in many ways, which I will talk about in great detail later.

3.2 History of adaptive music

Adaptive music is nothing new. Already for many years in the history video games the music has reacted to user inputs. While Atari's arcade classic Pong released in 1972 is considered to be one of the first arcade games and while it did have audio SFX when the ball hit the player's block, it didn't have any music. (Computing History 2018.) Another arcade classic Space Invaders released in 1978 however is considered one of the first games to have an adaptive soundtrack. When the player is fighting the aliens and they get closer to the player, the simple few notes get faster and faster. (Elmsley 2017.) Another example from the same era of gaming is the 1982 arcade game Frogger, where a new musical cue would play after you cross a certain point in the level.

These audio systems seem very simple by today's standards, but it is important to know the history of video game music to know how much game audio has developed over the years. The limitations of early audio systems led to experimentation and innovations that act as an important base for today's music systems. These limitations included a limited number of voices, limited synthesis, limited sample memory and the difficulty of programming audio inside the early audio systems. Some of these limitations like memory and voice limitations still are a challenge to composers. Many find these limitations to be more of a blessing than a curse, as working around them can fuel new innovations and ideas. (Sweet 2015, 86.)

3.3 iMUSE

Lucas Arts, previously known as Lucasfilm Games became famous for their adventure games in the late 1980's and the 90's. Lucas Arts became known for their games based on their SCUMM game engine, which enabled the famous point & click style interface. Many of the Lucas Arts adventure games were based on George Lucas' films, which meant the music was heavily connected with movie soundtracks.

After the first Monkey Island game, Secret of Monkey Island was released by Lucas Arts to huge success, composer Michael Land wanted a more interactive and flexible music system for the next Monkey Island title. In Secret of Monkey Island the music was fairly simple. There was source music, couple of different themes for characters, intro and outro music and a few interludes. The source music was connected to the place the music was "performed" in, so for example if the character entered a house, the music for the house would play. (Moormann 2012, 82.)

The arrival of the MIDI based iMUSE interactive music system allowed for two main upgrades to the music system inside the game engine: 1. The music was varied based on a variety of cues, which allowed a musical continuity between the different parts and themes. 2. iMUSE made it possible to re-orchestrate the score seamlessly based on MIDI. When the character would enter a new area, new instruments could be added to the mix without interrupting the base of the score. These two techniques are now known as horizontal re-sequencing and vertical re-orchestration. (Moormann 2012, 82.) Still today most of the modern games use these principles to create an adaptive soundtrack.

With the help of iMUSE the transitions between themes and moods in Monkey Island 2 were seamless. When the main character Guybrush Threepwood first arrives to the island he is robbed of his treasure by a character named Largo LaGrande. Like Threepwood, Largo LaGrande has his own unique theme, which is played when he is shown in the game. When later the main character Threepwood is moving around the Woodtick village entering various shops and houses, new elements are seamlessly added to the Woodtick village theme music. (Elmsley 2017.) Because the iMuse system was based on MIDI information, the transitions and instrumentation changes were totally seamless, similar to if one were to add or change around MIDI information in a modern DAW, like Ableton Live. iMuse sent out MIDI data which the dedicated soundcard then would play back. Because of this the soundcard of the computer you were playing with would affect how the instruments would sound (Silk 2014.) Modern soundcards don't work this way and this is the main reason for MIDI not being a reliable way to handle adaptive music in most modern games.

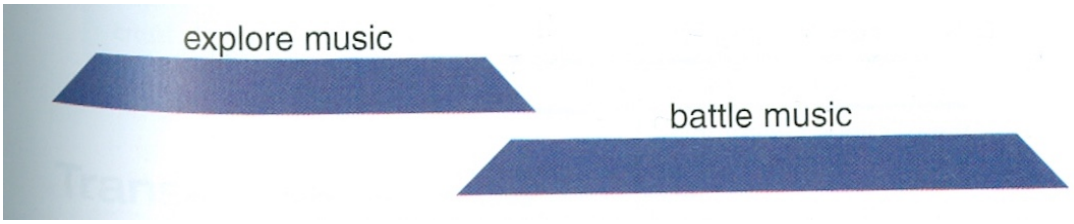
4 ADAPTIVE MUSIC TECHNIQUES

There are a variety of techniques available to create an interactive adaptive music system to your game. The choice of what adaptive music techniques to use in the game you are composing for depends completely on the type of the game. Slow paced, atmospheric puzzle games require different solutions than fast paced first person shooters. (Sweet 2016.) Usually games use a variety of the techniques that I am going to talk about later in this part. Michael Sweet describes adaptive music techniques excellently in his 2015 book "Writing Interactive Music for Video Games" which I use as the main reference for this part.

Maybe the most common adaptive music techniques are horizontal re-sequencing and vertical remixing. These techniques can be broken down into sub-categories of different ways to use them. (Sweet 2015.)

4.1 Horizontal re-sequencing

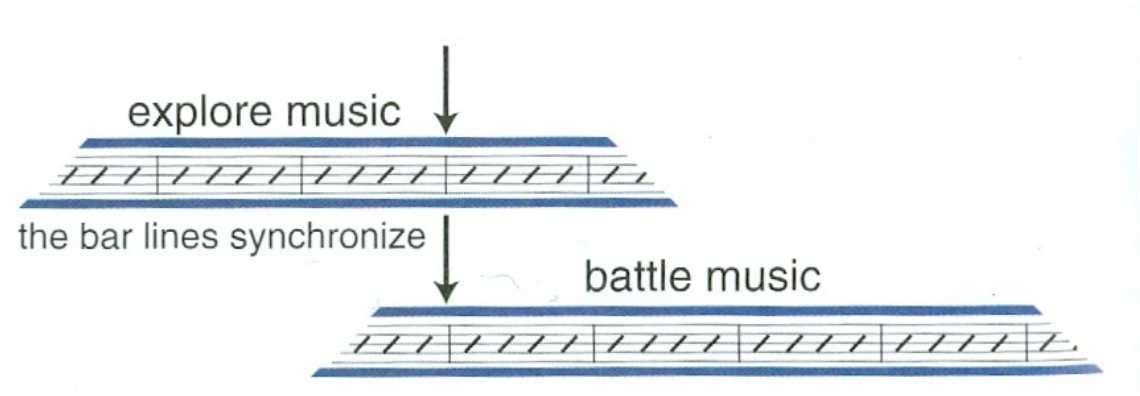
In horizontal re-sequencing different music cues trigger based on the players actions or other changes in the game. There are various subcategories of horizontal re-sequencing techniques. Crossfading between different musical cues is maybe the simplest example of horizontal re-sequencing. When a player moves between two areas of the game, the music of the previous are would fade out as a new musical cue of the new area fades in. Another common example of horizontal re-sequencing utilizing crossfading found in many games is the change of music when entering a fight. (Sweet 2015, 145.) For example a player could be walking around the area and calm music would play. When an enemy approaches the player the music would change to a stealthier mood. When the player engages in combat, new elements are added to the music. After the fight would be over, the music would transition back to a calm mood. (Deus Ex: Human Revolution 2011.)



An example of a exploration theme crossfading into battle music (Sweet 2015, 145).

4.1.1 Synchronized & non-synchronized re-sequencing

Crossfading tracks in the score can be either synchronized or non-synchronized to tempo (Sweet 2015, 145). When the score is synchronized the transition between tracks sounds usually smoother and more musical. This can be easily done in middleware such as Wwise, where you can set a tempo which the music will follow. This global tempo keeps both of the tracks fading in and out in the same rhythm. You can also determine when you want the transition to happen. In Wwise you can set the transition to happen immediately, at the next bar, next beat, end of the current musical cue and so on. These transition rules are helpful if you would like the previous music track to finish a certain phrase or a melody before the next track will fade in. (Audiokinetic 2018.)



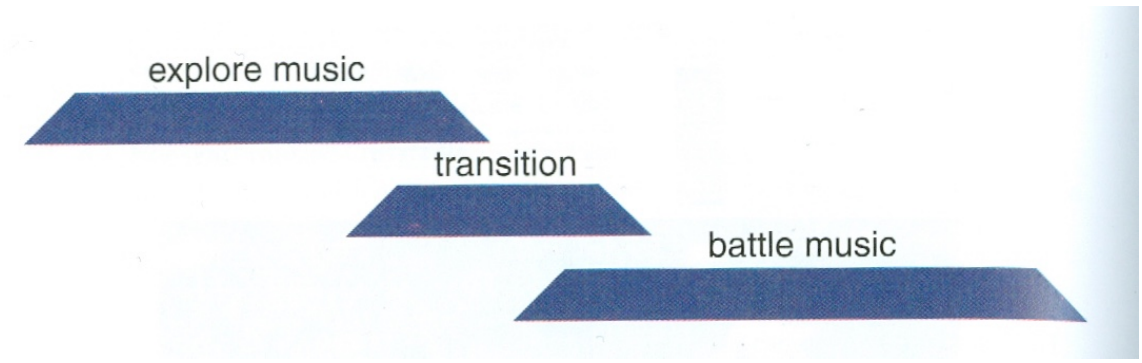
Two musical cues synchronized by tempo (Sweet 2015, 146).

With a non-synchronized score the crossfading of musical tracks can be very sudden and sound unnatural. If the previous track is played at 125 bpm (beats per minute) and a 107 bpm track is faded in it can sound unpleasing and disruptive. With music that is not tied to a tempo this method can be useful, and create a powerful, sudden impact. This technique can be very startling to the player if not executed properly. (Sweet 2015, 146.)

4.1.2 Transitional score

In a transitional score two musical cues are tied together by a musical transition track. Lets take an example where the player is exploring an area and chooses to walk towards the nearby village. The exploration has a subtle, quiet theme, but when the player crosses a certain point entering the village, a transition cue is triggered. This transition cue acts as a build up to the villages theme that will be played after the transition. Games like the Uncharted series by Naughty Dog use transitions in between cues to tie the score together (Brown 2014).

Similar to other re-sequencing techniques, using immediate crossfades the transition can occur at an unwanted time and interrupt an important melody or a cadence in the track that was playing. It is also important to have a long enough transition between the two musical cues, especially if the cues are very different. In the village theme example I gave earlier the village theme can be very different in character to the regular exploration theme, so a long enough transition is required to bridge the two.



A transition can be used to tie two musical cues together (Sweet 2015, 148).

4.1.3 Phrase branching

As I mentioned earlier in audio middleware such as Wwise a transition between two musical cues can be programmed to occur more naturally after a certain beat or a bar. As an example when transitioning from an exploration theme into the combat theme, the combat music wouldn't play immediately, but with a little delay on the next beat or bar. (Audiokinetic 2018.) This is called a branching score or phrase branching. Contrary

to the previously explained horizontal re-sequencing techniques phrase branching does not involve crossfading, but the musical cues are transformed instantly on the next suitable spot. (Sweet 2015, 149.)

Because the audio engine knows to wait until a certain part of the musical cue is played before transitioning into a new one, phrase branching can sound very musical and make the score develop well. If an important melody is being played and a new musical cue is triggered, the melody can be played until the end and the transition to a new musical cue will not interrupt it. However because of this delay in the transition the phrase branching technique can be too slow for certain situations. For example if a player was involved in combat but is already done with all the enemies a combat theme could still be playing until the end of a certain phrase in the music. While this can sound musically pleasing and natural, the player can get confused why the combat theme is still playing after defeating the last enemy. This type of confusion is not desirable as it can affect the gameplay in a bad way. (Sweet 2015, 149.)

4.2 Vertical remixing

Vertical remixing, sometimes called re-orchestration or layering is a technique, where a musical cue or a track is broken up to two or more layers. When an event is triggered in the game, a new layer could be brought into the mix on top of the "base" layer rather than switching to a new musical cue as in the horizontal re-sequencing techniques. This way even a short clip of music can be broken up to multiple layers and create more complex variations on the same track to avoid repetition. (Sweet 2015, 157.)

Benefits of this approach are the immediate and seamless variations if so desired. When a new layer is being introduced, it can fade in instantly without sounding crude and choppy. When new instruments are added to the mix, the transitions and changes can sound very subtle and if so desired, unnoticeable. These subtle changes can work as very powerful tools for building the mood of the game. Think of how an added timpani track can bring intensity to a boss fight. The player doesn't necessarily even notice the difference instantly, but it could help the player feel more connected and immersed in the game. (Michael Sweet 2015, 157.)

When a musical cue is broken down into multiple layers a whole new aspect of variation is introduced compared to horizontal techniques. If two musical cues are crossfaded into each other it can sound way more musical with for example percussion fading out first and starting the new cue with only an ambient pad sound. The composer could create one long track that plays when the player is exploring the game. This exploration cue acts as a base and new layers, such as various melodies and themes in the same key can be brought in very quickly and easily. (Velardo 2017.) The soundtrack of Rockstar Games Red Dead Redemption is a prime example of vertical remixing. The whole soundtrack of the game was composed in 130 bpm and in the key of A minor, which allows for new elements to come in and react quickly to the actions of the player or environment. When the character gets on a horse, a rolling bass layer is added to the mix to inform the player about the change and make them feel like something new is happening. (Stuart 2010.) This really adds to the immersion and the good feel of the gaming experience.

There are also certain disadvantages to the vertical remixing method. When using the layering technique, all of the layers need to be in the same key and tempo. This results in the inability to create big, sudden changes. Another disadvantage of the vertical remixing technique can be the not-so-musical way of fading in and out parts of the music in the middle of lead melodies or other important phrases of the score. If the base layer would be in the middle of playing an important, catchy melody and suddenly, according to a change in the game horns and percussion are added to the mix, it can sound rather disruptive and even dissonant. In linear music these changes are obviously planned beforehand and thus sound intended. (Sweet 2015, 164).

4.3 Music based games

Commonly music is based around the game, but in some games music comes first, and the game is based around the music. A pioneering title of the rhythm based game genre is Dance Dance Revolution, a Japanese game by Konami, first released in 1998. In Dance Dance Revolution players hit blocks with arrows pointing into every direction with their foot according to what's happening on the screen. When an arrow pointing left is shown on the screen, players need to step on the left arrow block with an accurate timing. (Nintendo 2010.) Another extremely popular videogame of this genre is the Guitar Hero franchise, where players push buttons on a guitar shaped controller according to notes appearing on the screen. Depending on how well the player performs, the music changes accordingly. In Guitar Hero players can choose from various popular music tracks, which creates an immersive feeling of getting to play your favourite songs on a "guitar". (Guitar Hero 2005.) Both of these games require rhythmic precision and the whole gameplay is based around what music is playing.

Another pioneering example from 1996 is PaRappa The Rapper for Playstation 1. The game was developed by NanaOn-Sha, a company focused on rhythm games. The composer Masaya Matsuura is considered to be a pioneer in rhythm games, as PaRappa The Rapper is considered to be the first rhythm game ever created. In the game you play as the character Parappa, a paper-thin dog who wants to become a superhero by rapping. The game is based on the now familiar technique of pushing buttons as they appear on the screen. When the buttons are pressed on the correct rhythm PaRappa will rap the lyrics sharply on beat. If the player is inaccurate with the timing, the rapping will sound off beat and eventually you will lose the game. (PaRappa the Rapper 1996.)

A more modern example of the music based game genre is the Crypt of the Necrodancer, released by Brace Yourself Entertainment in 2015. In Crypt of the Necrodancer you explore dungeons, kill monsters and collect loot, but the character only moves on the beat of the score. The player must input the movement button presses according to the four by four beat, otherwise the character doesn't move. (Crypt of the Necrodancer 2015.)The game is considered to be very difficult because of the accuracy needed to progress in the game. The score composed by Danny Baranowski has achieved a huge popularity streaming millions of times on Spotify and Bandcamp. (Spotify 2018.) This is a great example of how important a videogame score can be.

4.4 Procedural music

The usual way of handling adaptive music is to write short or long pieces of predetermined musical cues and tie them together with adaptive techniques discussed before. Usually adaptive scores have randomization to some degree, however some game composers have chosen a procedural way to create a score that won't necessarily ever sound the same. The good side of procedural music is the infinite variety of the soundtrack, as it doesn't repeat itself. The difficulty of procedural music systems is to get the logic and control working so that it will sound musical and intended while still being generative and procedural. Random sounds tied together won't usually sound too pleasing. (Sweet 2015, 239.)

Spore, developed by Maxis and released by Electronic Arts in 2008 is a game where players become gods, creating their own universe from cells to space age (Spore 2008). The gameplay itself is largely procedural with the planets environment generating randomly through an algorithm. The players could also "dream up" a creature, and the game engine would animate it based on the information. The composer and producer Brian Eno created a soundtrack for Spore that utilizes procedural music techniques. Eno used a software called Puredata, which is used to create audio devices such as synthesizers and samplers with a node based programming interface. In Spore however the audio itself is handled in the game engine and Puredata handles the procedural system. In Puredata the programmer can set certain scales, which the played synth notes will then follow, or loop eight randomly selected drum samples and so on. In Spores Puredata system Brian Eno and the programmers set a group of more complex rules and logic so that the music would sound musical, yet ever changing. (Kosak 2008.)

Another more recent example of procedural content in gameplay and in the score is No Man's Sky released in 2016 by Hello Games (No Man's Sky 2016). In the game players can explore an endless universe of procedurally generated planets with their spacecrafts. The soundtrack for No Man's Sky is also partially procedural, reacting to what happens in the game. A british electronic rock band 65daysofstatic was hired to do the score for the game, but instead of just recording a group of tracks they split the tracks up into small layers and pieces of music that can be paired together procedurally. The Hello Games audio director Paul Weir created an audio system called Pulse, which is a way of

“organizing lot’s of small, granular bits of sound [and] music components, and imposing a bit of structure and logic and control on top of them”. This way the soundtrack never sounds quite the same and acts according to the actions of the player. (Epstein 2016.)

5 ABLETON LIVE AS A TOOL FOR COMPOSING

5.1 Why Ableton Live compared to other DAW's?

As stated before composing for non-linear media differs greatly from composing a linear soundtrack. In most modern DAW's (Digital Audio Workstations) you can sequence, edit and process audio in almost any way imaginable. That means every composer can achieve every result they want with their DAW of choice. However I think Ableton Live has great features for non-linear music composition. What makes Ableton Live different in my opinion is the non-linear session view, where the composer can play around with clips and loops endlessly. Ableton Live is divided into two different views, a linear timeline arrangement view that you would find in most DAW's like Logic Pro or Pro Tools, and a session view. In this mode Ableton shows a grid of tracks lined up horizontally, and on every track a list of clip slots vertically. These tracks are either audio, MIDI, or effect return tracks. The session view compiles clips of audio or MIDI into a grid, which is tied into the global tempo if so desired. This makes it very easy and intuitive to play around with compositional ideas. (Ableton 2018.)

Ableton Live layout. (Kähärä 2018.)

5.2 Ableton Live composition techniques

I come from a background of electronic music, where sampling, looping and sequencing is much more familiar than recording instruments and playing live. When composing for videogames I feel like my background has helped me quite a lot in understanding how loops and sequences work, and how to compose music suitable for media that requires these techniques. Ableton Live is a great tool for this, as it was at first intended as a live tool for electronic musicians (Henke 2016).

When composing for games it is important to acknowledge the way the music is going to work inside the game, and how it is going to be implemented. Certain progressions or even styles of music wouldn't necessarily work the same way in a non-linear format as in a linear score. In Ableton's session view the music is broken down into loops and clips and that already takes the composer out of the linear way of making music. For example if I have five different drum loops on one track, two bass loops on another and three MIDI melodies loaded up on the session view, I can play around with different combinations and transitions endlessly. This resembles very much the horizontal resequencing and vertical remixing techniques discussed earlier. (Rohrmann 2016.) When playing around with loops in the session view, it is also very easy and intuitive to record new ideas, and come up with good layers quickly. When you launch clips on top of each other, take out tracks and add layers it resembles a combination of horizontal resequencing and vertical remixing. In other DAW's like Logic or Pro Tools the composer would need to arrange everything manually into the arrangement view. That takes more time and effort, and can kill the inspiration quickly. For me Live's session view acts as a sketchbook of ideas.

Another great feature in Live is to jump between the session and arrangement modes. This enables to record what you did in session view into an arrangement to see how a transition or a progression would end up sounding in the game. After that you could also break that arrangement down into new clips and play around with them even more. This provides a great workflow for creating complex scores that work well in a non-linear fashion. (Ableton 2018.)

When dealing with loops in Wwise or other middleware, it's important to think of the decays and reverb tails that go over the loop. In Ableton Live, you can render as a loop, so that Live bounces the leftover tails to the beginning of the loop. This is a good technique in some cases, but sometimes it is not desired to have that tail in the beginning of the loop. It's possible to loop a clip in Wwise for any time desired, and when transitioning from that loop to another cue, the leftover tail plays over the new cue creating a smooth transition. That's why it is a good practice to bounce loops with a couple extra bars after the loop, and only loop the desired part in Wwise. (Audiokinetic 2018.)

6 UTILIZING AUDIOKINETIC WWISE AS A MUSIC SYSTEM

6.1 What is Audiokinetic Wwise

Wwise is a middleware program designed to make it easier for composers and sound designers to implement audio into games. Wwise provides a convenient workflow for all audio needs and has features that common game engines like Unity or Unreal Engine wouldn't necessarily have. It also doesn't require any coding or scripting which makes it more accessible for audio designers and composers to use. The composer can create the whole audio system within Wwise with all of the required logic, and the programmers can then call the events such as triggers, states and switches from the game engine. (Audiokinetic 2018.) The Audiokinetic Wwise official websites courses and tutorials work as the main reference for this chapter.

6.2 Wwise Hierarchy

Wwise is split into three main hierarchy sections, the Master-Mixer Hierarchy, Actor-Mixer Hierarchy and the Interactive Music Hierarchy. These folders act as parents to various work units, virtual folders, actor-mixers, containers and sounds. Even in a smaller game project a lot of assets pile up easily, so it is very important to group and manage all of the assets in a clear and convenient way. These hierarchies help to organize your Wwise project so that every asset can be accessed quickly. Inside these hierarchies you can organize audio inside different containers and work units. By choosing the right type of container for your audio, you get access to more suitable options for the kind of audio you are working with. (Audiokinetic 2018.)

6.2.1 Containers

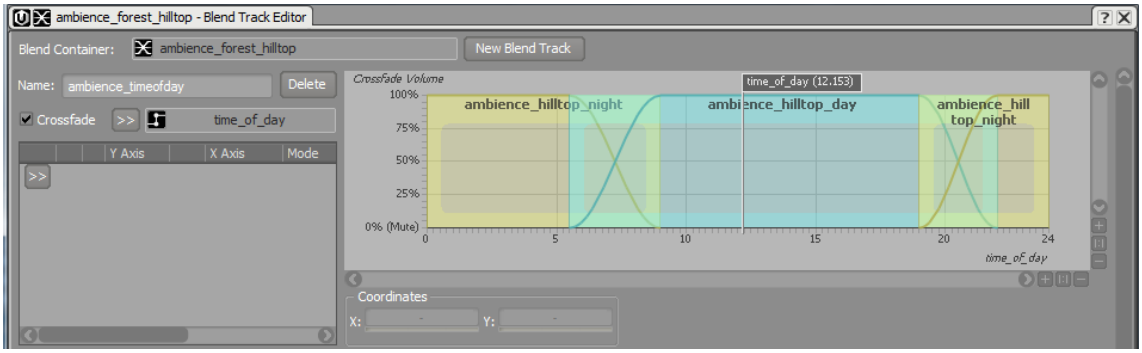
A Random Container is a group of objects or containers that that play back the objects in random order. Random containers are useful for sounds like footsteps or ricochet sounds, where variation is created through randomization. With a random container even as little as five footstep sounds can make up an illusion of multiple sounds and a varied soundscape. (Audiokinetic 2018.)

A sequence container is a container, where the objects inside are played after one another in a specific playlist style order. Sequence containers can be useful in for example gun sounds, where a mechanical layer is played first, then the body and a sub layer are added and a tail layer is played last. This wouldn't work inside a random container, as the sounds need to be played back in a specific order. (Audiokinetic 2018.)

Switch containers are containers, where objects or containers are mapped to correspond to different switches in the game. The object inside the switch container are then played according to what switch is currently active in the game. Footstep sounds work as a great example again. In a game there might be various materials where the game character can walk on, such as grass, concrete or snow. As there is different footstep sounds for each material, they can be grouped in different containers inside a switch container. (Audiokinetic 2018.) When the character is walking on snow, a snow switch is switched on and the corresponding container of snow footstep sounds is played. When the character moves to walk on concrete, the concrete footstep sound container is switched on.

A blend container is used to blend together objects inside a blend container. These objects can be then played back simultaneously on blend tracks and RTPC's (Real Time Game Parameters) can be used to control different object properties. Crossfading can be used between objects inside a blend track. (Audiokinetic 2018.) Blend containers are very useful in ambience tracks, such as wind ambiences. In Witcher 3 Wwise is mapped to track the weather inside the game. This is done with an RTPC that has corresponding values for clear and calm weather into stormy and windy weather. The blend containers have ambience tracks of different wind intensities, which are then blended and crossfaded into one another according to the RTPC controlled by the game engine. (CD Pro-

jekt Red 2016.) This way the audio never drags behind and is tied together with the game.



In this demonstrative picture, the two hilltop ambience tracks are blended and crossfaded according to the time of the day. This is controlled by an RTPC with 24 values, one for each hour. (Cryengine 2018.)

6.2.2 Hierarchy sections

The Master-Mixer Hierarchy is for audio buses, where you can route audio inside the Wwise. This is a familiar procedure from every common DAW, where audio is routed through buses to group and process things differently. These buses can have their own effects and settings, so that all of the music can be grouped together separately from all ambient sounds for example. This allows for various changes inside buses, such as low pass filter or volume variation. (Audiokinetic 2018.)

The Actor-Mixer Hierarchy is used for all of the sound effect assets in the game. This includes all of the sound effects and ambiences in the game. Inside this hierarchy it is possible to group the sound effects into multiple different containers and folders discussed before. (Audiokinetic 2018.)

The Interactive Music Hierarchy is used for all the music in the game. Considering the nature of the thesis, this is the hierarchy we'll be dealing with the most. The Interactive Music Hierarchy handles all the music files used inside the game and offers various different containers with different properties and purposes. Similar to the actor mixer hierarchy, these containers are used depending on the type of logic and action wanted in the music system. (Audiokinetic 2018.)

The containers inside the interactive music system are Music Switch Containers, Music Playlist Containers and Music Segments. (Audiokinetic 2018.) I will talk about these more in my practical part explanation.

6.3 Wwise Events & Game Syncs

Audio in Wwise is driven by various different events and game syncs. These controls make it possible for the programmer to call the events and game syncs from the engine, and Wwise does the rest of the work. (Audiokinetic 2018.) In the following chapters events and game syncs are explained.

6.3.1 Events

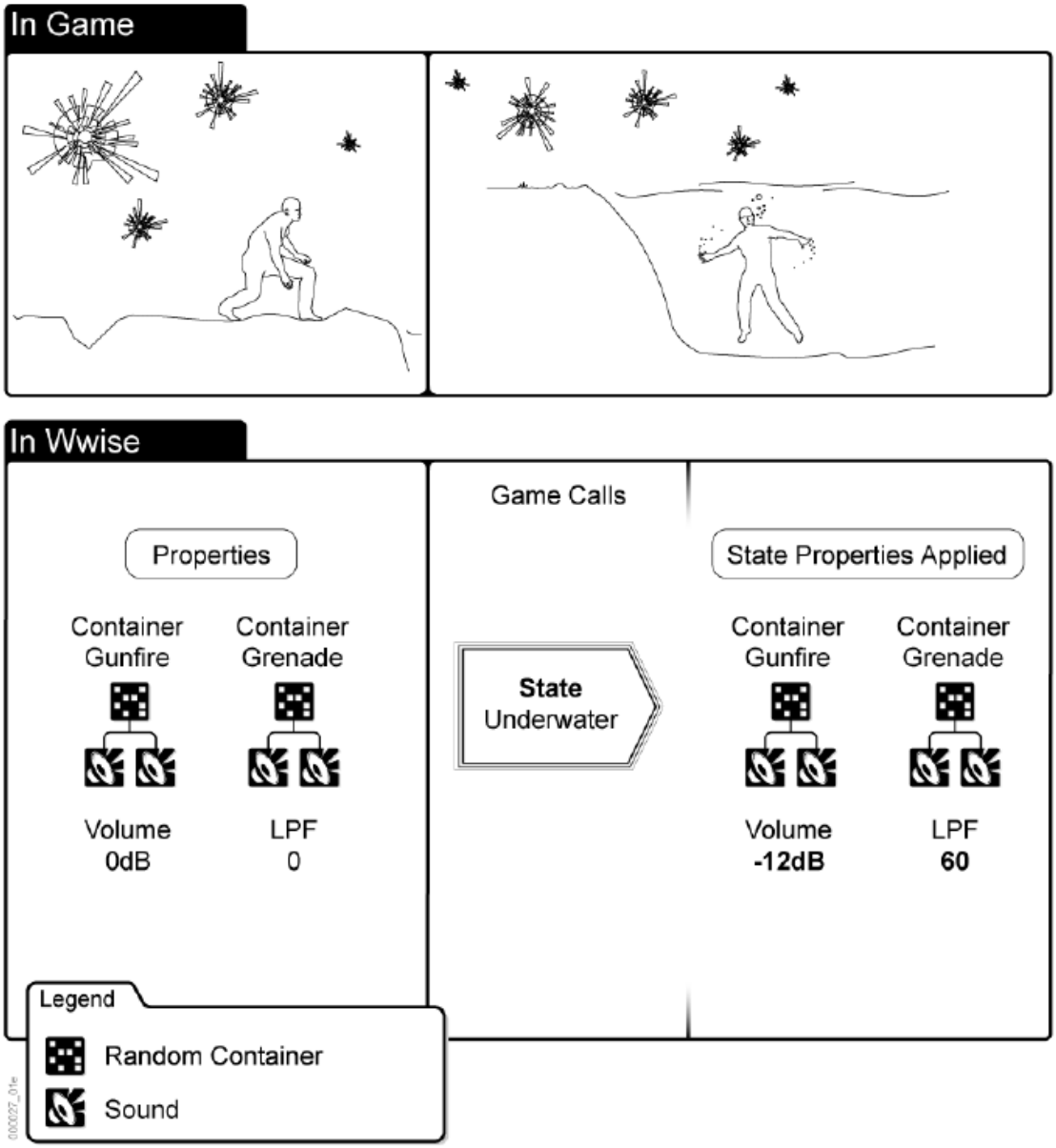
In Wwise events are used to control and drive the sound effects, music and other changes. An event can be assigned to do a particular job inside Wwise, and then it can be launched withing the game engine. The most simple example of this would be a gun sound effect. When a gun is fired in the game, the game engine launches an event that plays the gun sound inside Wwise. This event would be a simple play event for the gun sound. Other common events include stop, which stops the audio playing and break, which allows the current audio cue to finish before stopping it. You can also set voice volume, pitch, low pass and high pass filters and mutes with events. (Audiokinetic 2018.)

6.3.2 Game Syncs

When a change happens in the game, Wwise manages these changes in the audio side with game syncs. In the game syncs tab in Wwise you can find subcategories such as states, switches, game parameters and triggers. (Audiokinetic 2018.)

States can be thought of as mixer setting snapshots, where values are changed according to which state is active in the game. An easy way to describe this is when a character is underwater. Usually when underwater the volume of the voices is a bit lower and a fair amount of low pass filtering is applied. When the character goes underwater in game, Wwise receives a call from the game engine to switch the active state to “underwater”.

This state then changes the volume and filtering settings for the needed containers and objects as seen on the demonstrative picture below. States work great in changing multiple values at the same time, as any amount of values can be associated with a state. States can also be used to switch between musical cues. (Audiokinetic 2018.)

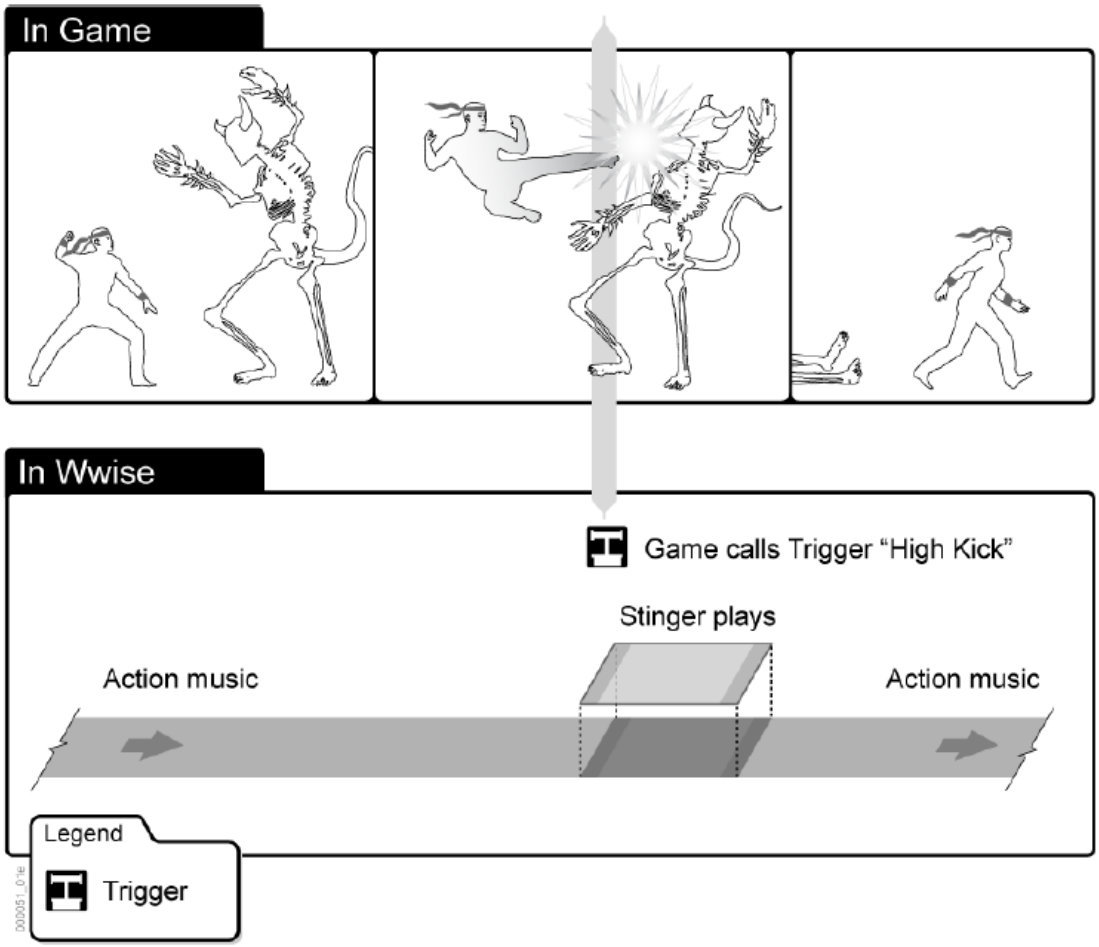


The character goes underwater, and an “underwater” state is called by the game, changing volume and filtering parameters in the audio containers. (Audiokinetic 2018.)

Switches can be thought as light switches, you turn them on and it stays on until you turn it off. In Wwise these switches can be assigned to switch containers to switch between for example footstep SFX material types, as I told before in the switch container paragraph. (Audiokinetic 2018.)

Game Parameters are elements in Wwise that are used to change objects values based on an RTPC (Real Time Parameter Control). A common way to use this is a game parameter assigned to blend tracks in a blend container. The blending is done by varying the RTPC value, which can be assigned to a value inside the game engine. It is also possible to use RTPC's to control vertical layers of music inside Wwise. As an example an RTPC with values from 1-5 could represent five intensities of music during a battle. These five points of the RTPC could be mapped to layer volumes, so that when the value gets bigger, more layers are added to the mix. (Audiokinetic 2018.)

Triggers can be used to trigger stingers on top of the game score. Sometimes a short, musical audio cue can be an effective way to bring intensity or information to the player. Stingers are great for this purpose, as they can be tied to the grid and the tempo of the score. A regular sound SFX would trigger instantly, but a musical stinger can be set to play on the next beat or bar. In the picture below a trigger is being called by the game engine, and the according stinger is played when the character finishes the enemy. This stinger works as an impactful and informative cue that works well tied with the scores tempo. (Audiokinetic 2018.)



Game calls the trigger for a musical stinger (Audiokinetic 2018.)

7 PRACTICAL PART

7.1 Project info

For the practical part of my thesis I created adaptive interactive music for a puzzle virtual reality game developed by students in TAMK TiKo game development department. The game carries a working title Cold Potato, but is to be changed later. In this game the character takes control of the subconscious of puppets placed around a steampunk style environment. The character aims to the puppets with a teleporter placed on the characters hand, and moves around the environment by teleporting to these puppets. The puppets are placed in tricky places, so that the player needs to solve puzzles in order to progress. The game is being developed in Unreal Engine 4, and the audio is done with Audiokinetic Wwise middleware. As it is a virtual reality experience, the game can be played with either the Oculus Rift or HTC Vive VR headsets. I am going to create sound design for the project in the future, but for my thesis project I focused on the interactive music.

7.2 Planning the project

I first heard about the project through a teacher in the TiKo department, when I was searching for projects to do audio for. When I heard more about the game I became instantly interested in creating music and sound design for the game. The steampunk and sci-fi feel of the game seemed very interesting and suitable for my skills.

For about two months I did some demos of the possible music for the game and started to think about the adaptive qualities of the music. As the game felt very atmospheric and moody, I wanted the music to play a big role in creating an immersive virtual reality experience. I started to collect some references from other games and music in general. I did some case studies and studied how adaptive music was handled in other games with a heavy focus on atmosphere. One of the inspirations was Playdead's Inside, which features a very minimalistic but powerful adaptive soundtrack composed by Martin Stig Andersen and Søs Gunver Ryberg. The minimalism leaves room for the sound design

and the other elements in the game, while still creating a beautiful atmosphere. I also received references of other game soundtracks they liked from the other developers.

7.3 Composing the score

When I was creating demos for the game I wanted the tracks to be in the same tempo, the same key and a similar musical style. This was essential considering the vertical and horizontal adaptive music techniques I wanted to utilize in the score. I decided on the tempo of 125 bpm, and the key of F minor. This way any piece of music can blend with another seamlessly at any point and it will sound pleasing harmonically and rhythmically. In Ableton Live it is easy to see how a new musical idea fits in with the existing ones, so the composing process was a fairly painless task in this matter.

I wanted the score to be dark, floaty and atmospheric, but with memorable melodies and hooks. This was maybe the most difficult task to achieve, as it is easy to overpopulate the tracks with different instruments and phrases. I also wanted all of the tracks to have a base layer, that would sound good on their own. When new elements would be added to the mix, they would compliment the most important first layer and bring in new nuances and charachteristics. Every track was also written and composed with looping in mind, so that they would sound good even after many repeats.

As the game has a gloomy steampunk aesthetic, I wanted the music to have a futuristic feel, but sound organic at the same time. All of the music is electronic and synth based, but I created sounds that would still sound organic and have certain grit to them. This was also achieved by processing with saturation, equalizing and filtering. Many of the songs use delays and reverbs in experimental ways, creating a spacey yet organic soundscape.

7.4 Implementing the adaptive music in Wwise

When I decided on doing adaptive music for our game project, Wwise seemed like the best fit because of its capabilities with adaptive music. The interactive music system in Wwise is capable of handling a large variety of changes and settings when it comes to adaptive music in a game. Things like a set tempo and grid are difficult to achieve with the game engines own audio system.

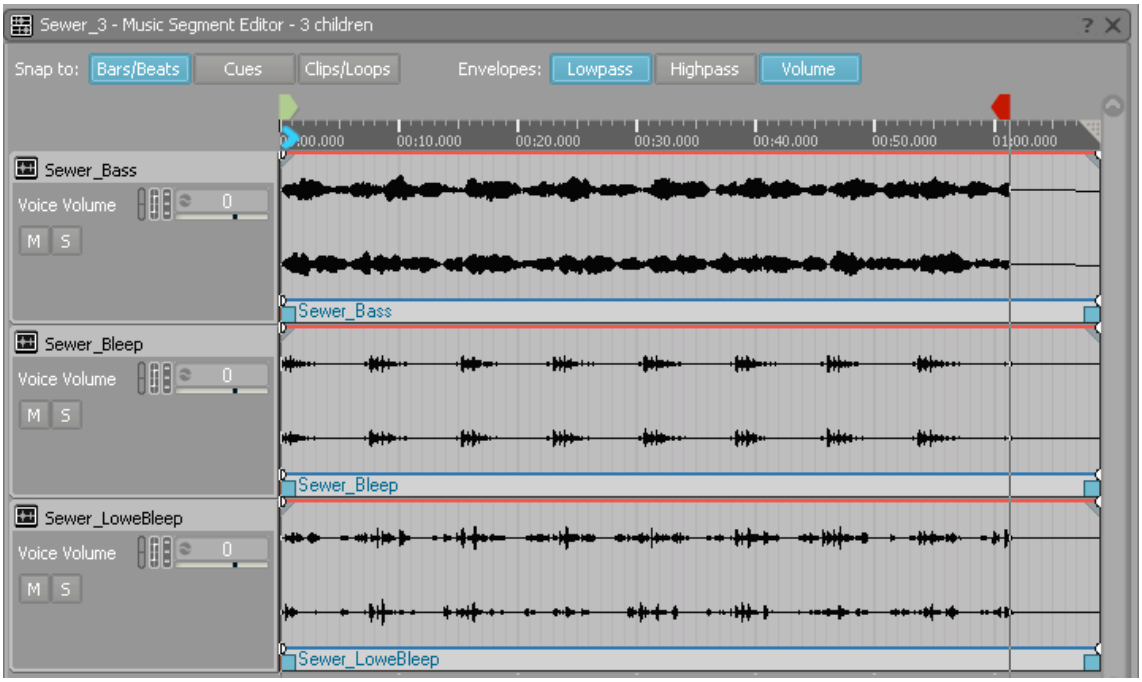
When I started to play around with the music implementation, I came across many difficulties and problems I had to figure out. This was a good thing, as in the process I learned how to use the Wwise interactive music system properly. In the end I settled on a way to handle the music that fitted best for our game.

I started with creating a parent switch container for all the music in the game. The music switch container allows me to do changes on the whole group and handle logic such as transitions and playback. In this container I also set the tempo to be 125 bpm for all of the objects in the container.

Under the switch container I created music playlist containers for all of the different layers of music I had in the game. There is various ways how to do layering and vertical remixing inside Wwise, but for this game this was the most efficient one. Every level had three to six layers depending on the level. For example the “circle hall” level in the game has six layers, all on their own playlist containers. These playlist containers are able to have music segments playing back in ways chosen by the designer. It is possible to have a complicated logic of playback so that a certain track plays after the first one has played three times and so on. For my project every playlist container just plays one music segment on an infinite loop, except for the “Jail” playlist where there is a short intro segment.

Inside every playlist container is a music segment with one or more music tracks. These music segments work similar to sequencers in modern DAW’s, where tracks are stacked vertically and they have a grid tied to tempo. Every track is loopable, with pre-entries and exit cues. These cues are handy when dealing with reverb tails or pickup notes on a loop. A loop could have a one beat pickup, and then loop four times. When transitioning to a new loop, the tail of the loop would still play after the new loop begins, creating a

smooth transition. In my project the different playlist containers for different layers are filled with according layers of music tracks. The first layer of a level has only one base layer in a music segment, but the fourth one has four or five layers of music inside a music segment.



A picture from my Wwise project demonstrating the music segment editor. The green marker marks the entry cue, the blue arrow is the playback position marker and the red marker is the exit cue. Notice how after the exit cue a tail is left to play. (Kähärä 2018.)

Inside the music switch container these playlists are coupled with states that represent the players progress. Every playlist container is named according to the state that it is associated with, such as “Jail_3”. This way when a state changes, the music changes accordingly. The states change whenever the player teleports to a new puppet in the game and progresses through a level. Every level has a different number of puppets, and sometimes player needs to go through three puppets for the music to evolve. These new layers may sound unnoticeable at first, but it creates a feeling of progress and achievement as the music progresses with the player.

I chose to associate the music with states because it allows for smooth transitions. Whenever a new layer of music is added to the mix, the switch container waits until the next bar to trigger it. The transition also has a 1.4 second fadeout and fade in with equal

curves to crossfade between states. Every transition is the same except for the transition from the “Between” playlist to the “Jail_1” playlist. The “Jail_1” playlist has a short intro segment which I want the engine to play when transitioning from “Between” to “Jail_1”. This is done by adding a “jump to playlist item: Jail_1” command in the transition settings.

Some of the states also change the parameters in the playlist containers. When the player finishes the puzzle successfully on the circle hall level and teleports to the final puppet, a “Circlehall_Final” state is turned on and the according playlist plays. This playlist has four layers, but some of the voices are pitched down an octave and low pass filtering is applied. This is done because I wanted the player to feel the sense of achievement and a greater change in the gameplay, like something is finished. States are good for these type of changes because they can handle multiple setting variations and changes.

I also composed musical stingers for the circle hall level. These trigger when the player progresses to a new puppet in the right order. These six stingers in total play a melody when the player progresses through the puppets in the right order. Every stinger is a short chord played by a piano and a pad, which blends in harmonically with the track playing. The stinger waits until the next beat in the tempo grid to play. Next beat is not immediate, but quick enough to feel like it happens when the player teleports. When it’s tied to the tempo, it feels more like a musical addition than a regular SFX.

8 DISCUSSION / CONCLUSION

During this project I learned a lot about adaptive music, especially about composing and implementing it. Before the game project I had made music for several small videogame projects, but I had never implemented the music myself, and it certainly wasn't adaptive to this degree. Before this project I had used Wwise previously for a bit, but creating and playing around with the adaptive music system really taught me a good amount of things you can achieve with Wwise, especially in the interactive music side. The project also taught how to compose music for interactive media and videogames in general. All the considerations with tempo, harmonical structure and looping needed to be planned beforehand, but always needed some fixing after implementing them into Wwise. This was a great lesson about the importance of implementation and doing it by myself. I really think composers should learn Wwise or some other audio middleware just to learn how to write better adaptive music. Working with our project team was a pleasant experience as everyone motivated each other and music was treated as an equally important part of the game.

I think in general there is going to be a growing need for adaptive music in media. A lot of linear media is moving towards a more interactive approach and interactive music is a huge part of these experiences. By converting to non-linear, adaptive music the media experience can evolve into a new level of immersion and entertainment. If everything else is interactive, the music should be too. Virtual reality is going to be more affordable and easily approachable every year and immersive adaptive music for these applications and experiences is going to be a standard. Economically adaptive music composition looks healthier than many other sides of music business, as the videogame industry is constantly growing and the need for new games and applications is constantly rising. I think adaptive music is a must for future media and new techniques are going to be developed in an increasing speed.

REFERENCES

Audiokinetic. 2018. Wwise 101 Certification Course. Read on 7.5.2018.
https://www.audiokinetic.com/learn/certifications/

Buffoni, Louis-Xavier. 2018. Article. Viewed on 7.5.2018.
http://gamesounddesign.com/making-interactive-music-for-games-part-one.html

CD Projekt Red, Audiokinetic. 2016. Wwise Tour 2016 – CD Projekt Red Witcher (2 of 6) – Ambience. Released on 3.10.2016. Viewed on 7.5.2018.
https://www.youtube.com/watch?v=VJUuI_dw8Cc

Crypt of the Necrodancer. 2015. Brace Yourself Games. Videogame.

Computing History. 2018. Atari PONG. Article.
http://www.computinghistory.org.uk/det/4007/Atari-PONG/

Cryengine. 2018. Wwise & Time of Day. Released on 18.3.2018. Viewed on 7.5.2018.
http://docs.cryengine.com/pages/viewpage.action?pageId=29450727

DanceDanceRevolution. 2010. Konami. Videogame.

De Santis,Dennis. 2018. Making Music: Creative Strategies for Electronic Music Producers. Ableton.

Deus Ex: Human Revolution. 2011. Square Enix. Videogame.

Elmsley, Andy. 2017. Blog Post. 8 Outstanding Adaptive Music Soundtracks. Released on 21.5.2017. Viewed on 7.5.2018.
http://melodrive.com/blog/8-outstanding-adaptive-music-soundtracks/

Epstain, Mike. 2016. How 'No Man's Sky' Composes Completely Original Music for Every Player. Released on 8.9.2016. Viewed on 7.5.2018.
https://www.digitaltrends.com/gaming/no-mans-sky-music/

Guitar Hero. 2005. Harmonix Music Systems. Videogame.

Hayes, Lance. 2011. A Composers Prespective On Game Audio. Blog Post. Released on 18.5.2011. Viewed on 7.5.2018.
https://www.keyboardmag.com/kb-blog/847

Kosak, Dave. 2008. The Beat Goes On: Dynamic Music in Spore. Article. Released on 20.2.2008. Viewed on 7.5.2018.
http://pc.gamespy.com/pc/spore/853810p1.html

Martins, Marcelo. 2012. Rethinking the Audio Loop in Games. Blog post. Released on 4.2.2012. Read on 7.5.2018.
https://www.gamasutra.com/blogs/MarceloMartins/20120402/167786/Rethinking_the_audio_loop_in_games.php

Moormann, Peter. 2013. Music and Game; Perspectives on a Popular Alliance. Springer VS.

No Man's Sky. 2016. Hello Games. Videogame.

PaRappa the Rapper. 1996. Masaya Matsuura. Videogame.

Rohrmann, C. Andrew. 2016. Creating Galak Z's Hyper Adaptive Music On An Indie Budget. Video lecture. Uploaded on 13.1.2016. Viewed on 7.5.2018.
https://www.youtube.com/watch?v=kB4botTHi90

Silk, Peter. 2012. iMUSE and Interactive Game Soundtracks. Article. Released on 27.6.2014. Viewed on 7.5.2018.
http://www.kestrelpi.co.uk/blog/2014/6/27/imuse-and-interactive-game-soundtracks

Spore. 2008. Maxis. Videogame.

Spotify. 2018. Crypt of the Necrodancer Soundtrack.

Stuart, Keith. 2010. Redemption Songs: The Making of the Red Dead Redemption Soundtrack. Article Released on 26.5.2010. Viewed on 7.5.2018.
https://www.theguardian.com/technology/gamesblog/2010/may/26/red-dead-redemption-soundtrack

Sweet, Michael. 2015. Writing Interactive Music for Video Games. Addison Wesley.

Sweet, Michael. 2016. Top 6 Adaptive Music Techniques in Games – Pros and Cons. Article. Released on 13.6.2016. Read on 7.5.2018.
https://www.designingmusicnow.com/2016/06/13/advantages-disadvantages-common-interactive-music-techniques-used-video-games/

Velardo, Valerio. 2017. What Is Adaptive Music. Melodrive. Blog Post. Released on 22.1.2017. Viewed on 7.5.2018.
http://melodrive.com/blog/what-is-adaptive-music

9 APPENDICES

Appendix 1. Demonstrative video of the thesis' media part.

Part 1. Youtube.
https://youtu.be/591JWl82x6c

Part 2. Youtube.
https://youtu.be/QqAJNSMYM-4

Part 3. Youtube.
https://youtu.be/V2yWB_6BHVM