



## **IMPLEMENTATIONSVERKTYGENS PÅVERKAN PÅ KOMPONERANDET**

Hur påverkas komponerandet av dataspelsmusik av implementationsverktyg för dataspelsmusik?

## **HOW IMPLEMENTATION TOOLS AFFECT COMPOSITION**

How is the act of composing for video games affected by the video game implementation tools used?

Examensarbete i medier, estetik och berättande  
Grundnivå 15 högskolepoäng  
Vårtermin 2026

Johannes Rönndahl  
Lily Sandberg

Handledare: Lars Bröndum  
Examinator: Anders Sjölin

# Sammanfattning

I denna studie undersöks hur digitala implementationsverktyg påverkar kompositionsprocessen och motivering av kompositörens verktygsval när de skapar musik för spel. Ludomusikologi är ett ungt fält där frågor om hur det kan studeras ställs. Fältets befintliga forskning har sällan varit utifrån ett produktionsperspektiv, vilket denna undersökning ämnat ha. I undersökningen intervjuades tre yrkesverksamma kompositörer med erfarenhet av olika implementationsverktyg. Intervjuerna genomfördes semi-strukturerat och spelades in i transkriberingssyfte. Transkriberingarna bearbetades med en "Gioia-inspirerad" tematisk tolkning. Tolkningen av materialet pekar bland annat på att implementationsverktyg förbättrar arbetsflödet, iterationsmöjligheter och att den musikaliska kvalitén kan öka till följd av detta. För framtida forskning hade fler respondenter behövts för att möjliggöra en ökad generaliserbarhet för studiens upptäckter.

**Nyckelord:** Dataspelsmusik, Komposition, Implementation, Ludomusikologi, Mjukvara, Implementationsverktyg.

# Innehållsförteckning

<b>1</b>	<b>Introduktion .....</b>	<b>1</b>
<b>2</b>	<b>Bakgrund .....</b>	<b>2</b>
2.1	Tidigare forskning på dataspelsmusik .....	2
2.2	Implementationsverktyg .....	3
<b>3</b>	<b>Problemformulering.....</b>	<b>6</b>
3.1	Metodbeskrivning.....	6
3.1.1	Transkribering och kodning .....	7
3.1.2	Forskningsetik.....	8
<b>4</b>	<b>Genomförande och Analys .....</b>	<b>9</b>
4.1	Förberedande arbete .....	9
4.2	Pilotintervju .....	9
4.3	Intervjuer.....	9
4.3.1	Tobias Kalliokulju .....	10
4.3.2	Gustaf Blix.....	13
4.3.3	Jacob Westberg .....	17
4.4	Analys av intervjusvaren .....	23
<b>5</b>	<b>Sammanfattning och diskussion .....</b>	<b>27</b>
5.1	Sammanfattning.....	27
5.2	Diskussion .....	27
5.2.1	Samhälleliga och etiska aspekter .....	29
5.3	Framtida arbete.....	30
	<b>Referenslista .....</b>	<b>31</b>

# 1 Introduktion

En kompositör för dataspelsmusik har unika möjligheter i sättet hen implementerar musiken jämfört med andra medier på grund av dataspels potential att vara interaktiva och adaptiva. En dataspelsmusikkompositör kan skapa musik som följer vad spelaren gör och eller det som händer på skärmen. För att få musiken att fungera som tilltänkt kan dataspelsutvecklare använda sig av mjukvara som underlättar steget att implementera musiken i spelet. En relevant fråga är då: Hur påverkar dessa verktyg komponerandet av dataspelsmusik?

För att besvara frågan intervjuades yrkesverksamma dataspelskompositörer. Detta kan ge insikt i hur dessa verktyg påverkar komponerandet och därmed fördjupa förståelsen av implementationsverktygens roll för kompositören. Genom respondenternas egna erfarenheter och upplevelser av användandet av implementationsverktyg kan ett samband mellan verktyg och komposition av musik undersökas. Urvalet av respondenter gjordes genom subjektivt urval kombinerat med snöbollsurval.

Forskning inom ludomusikologi (läran om dataspelsmusik) har varit utifrån ett analytiskt eller historiskt perspektiv men sällan utifrån ett utvecklarperspektiv (Engström 2020). Utöver detta är ludomusikologi ett ungt vetenskapligt fält där forskare försöker definiera fältets forskningsmöjligheter (Fernández-Cortés & Cook 2021; Kamp, Summers, & Sweeney, M. 2016).

Bakgrundskapitlet ger kontext för studien och förklarar bland annat ludomusikologi och dess historia som forskningsfält, vilka begränsningar spelmusik hade på 80- och 90-talet, vad implementationsverktyg är och definieras som i den här studien samt flera tekniker som används inom implementationsverktygen som looping och randomisering.

I problemformuleringskapitlet förklaras studiens syfte, dess frågeställning, och metod samt en problematisering av den metoden. Vidare i kapitlet beskrivs metodens detaljer samt en överblick på studiens struktur. Fortsättningsvis beskrivs det hur datan samlats in, bearbetats och tolkats. Därefter redogörs forskningsetiska perspektiv på metoden.

I genomförande och analyskapitlet beskrivs studiens utförande i detalj med förberedning, pilotintervju, samt hur intervjuerna genomförts och analyserats. Genomförandet av pilotintervjun beskrivs utifrån vad som hände och vad som behövde förändras innan intervjuerna. Intervjuerna beskrivs sedan kronologiskt med tillhörande tolkningsdiagram och avslutas med ett större tolkningsdiagram som representerar alla respondenternas svar med tillhörande tolkningar.

I sammanfattning och diskussionskapitlet diskuteras den insamlade datan, vad studien kommit fram till, eventuella slutsatser och svaret på frågeställningen. Det diskuteras även utifrån intervjuerna och egna tankar vad som kan hända i framtiden, hur spelmusik och implementation kan påverkas av implementationsverktyg och i vilken utsträckning arbetet hade kunnat fortsätta.

## 2 Bakgrund

Nedan följer en kort redogörelse för ludomusikologi som vetenskapligt fält och dess historia, hur icke-linjär musik fungerar i dataspel, dataspelsmusikens historiska begränsningar, vad implementationsverktyg är samt flera tekniker som används inom dem – däribland generativ musik. Utöver detta förklaras även teknikernas roll inom implementationsverktygen och hur verktygen underlättar för implementeringen.

### 2.1 Tidigare forskning på dataspelsmusik

Engström (2020) skriver i en litteraturöversikt att det är förvånansvärt lite forskning som gjorts på ljuden i dataspel från ett utvecklarperspektiv. Han menar att studierna som gjorts inom ämnet i fråga är främst analyser på etablerade spel. Vidare förklarar han att det inte heller finns mycket forskning på spelutveckling från mediaproduktionens perspektiv. Enligt Fernández-Cortés och Cook (2021) och Kamp, Summers och Sweeney (2016) är ludomusikologi ett ungt vetenskapligt fält som forskare försöker definiera forskningsmöjligheter inom. Fernández-Cortés och Cook (2021) menar även att dataspelsmusik som forskningsfält har uppfattats som ett komplement till ett ”enkelt barnligt intresse” och för att forskningsmöjligheterna ska förbättras krävs det ett erkännande från forskningsvärlden genom att denna uppfattning förändras. Fernández-Cortés och Cook (2021) tar också upp att forskare inom ludomusikologi alltid har behövt motivera fältets relevans, utifrån det ekonomiska värdet, demografin och dess kulturella genomslag.

Hårdvara till dataspel hade till en början många fler begränsningar än idag när det kommer till musik och ljud. Ljudchippen som fanns i 80- och 90-talets spelkonsoler saknade förmågan att spela upp mer än ett par ljud åt gången vilket begränsade musikskapandet till slagverk, melodi, undermelodi och bas (Burke 2024). Detta innebar att vissa ljud fick flera funktioner, till exempel att en musik-stinger transponerades upp och återanvändes vid flera tillfällen, för att spara minne (Horowitz & Looney 2014). Burke (2024) tar också upp Sega Genesis och dess begränsning med åtta kilobyte dedikerad ljud-RAM, samt att hela spelet skulle få plats inom fyra megabyte, jämfört med idag då begränsningen på hårdvaran inte är lika extrem.

Medina-Grey (2016) beskriver hur musiken i dataspel kan betraktas som ”modulär”. Hon menar att musiken kan organiseras med hjälp av regler som bestämmer vilken ordning olika delar av den kommer att spela upp ljud. I en dataspelskontext kan dessa regler innebära exempelvis var spelaren befinner sig, vad spelaren gör och vad som händer på skärmen samt element av slump, menar hon.

Spelstudion House House (2019) försökte efterlikna en kompositionsteknik som kallas ”Mickey Mousing” när de implementerade Claude Debussys Prelude (Golding 2021). Tekniken beskrivs kort som ett försök att synkronisera musiken till animationen i film, ofta för komisk effekt. I *Untitled Goose Game* (House House 2019) har utvecklarna delat upp musikstycken av Debussy i mindre delar som spelas upp bit för bit när spelaren utför en handling som för progressionen framåt (Golding 2021; Bhogal 2024). Nico Disseldorp som utvecklade dataspels musiksystem tilldelade en ”nivå av uppmärksamhet till gäsen” till spelets bybor vilket korresponderar till tre musikaliska tillstånd: ”tystnad”, ”låg intensitet” och ”hög intensitet”. För att uppnå känslan av att spelarens handlingar kommenteras av pianot, har varje musikdel en låg och hög intensitetsnivå som byts i relation till bybornas ”uppmärksamhetsnivå”.

Ertan (2025) undersöker hur känslan av "flow" spelaren kan uppfatta under speltid kan uppnås på liknande sätt i film. För att spelaren ska kunna uppnå känslan av "flow" behövs en balans av utmaning och spelarfärdighet. I texten ges exempel på sätt att få spelaren att uppleva en ökad självskattad tillfredsställelse, vilka ligger till grund för möjligheten att uppnå känslan av "flow" enligt honom. Ett av dessa är "dynamic music" som syftar på hur musiken kan förändras beroende på vad som händer på skärmen och/eller med spelaren, till exempel att vara i aktiv strid jämfört med att vara i ett säkert område. Vidare menar Ertan (2025) att musiken agerar som en "osynlig tråd" som syr samman narrativ, spelarens uppfattning och interaktiviteten till en sammanhängande upplevelse. Musik är inte bara utsmyckning, utan driver aktivt engagemang som formar publikens känsla och uppfattade narrativ, menar han.

Hutchings och McCormack (2020) beskriver i en artikel hur de skapat ett adaptivt musiksysteem (AMS) och implementerat det i två distinkt olika spel för att testa musikens påverkan av spelarens immersion. I studien skapades ett musiksysteem som genererade musik med hjälp av algoritmer som analyserat musikaliska mönster i olika databaser. Den genererade musiken kunde sedan dynamiskt förändras baserat på vad spelarna gjorde och/eller befann sig i spelet. Vidare beskrivs det att algoritmisk musikkomposition har blivit mer avancerad men ofta saknar kontroll över det musikaliska uttrycket som är önskat i dataspel. I artikeln redogörs svagheter och styrkor som denna typ av musikskapande innebär, vilka är: hur konsekvent musiken är, transparens och flexibilitet. I studien kartlade forskarna känslassociationer till olika musikstilar som låg till grund för musiksysteem för att bättre kunna generera musik som representerar känslan som efterfrågas. AMS implementerades i två olika redan existerande spel där systemet gjorde tre saker: tog reda på spelets tillstånd, matchade spelets tillstånd med kartläggningen av känslorna och genererade musiken med hjälp av musikaliska fragment som organiseras av systemet. Det som uppdagades av denna studie var att spelare med det adaptiva musiksysteem självskattade en ökad immersion av spelupplevelsen jämfört med musiken som var i spelet från början.

Lopez Duarte (2020) beskriver i en tidskriftsartikel flera adaptiva musiksysteem som genererar musik kontinuerligt. Utmaningen som ställs i texten är skillnaden mellan förinspelad musik och hur den görs adaptiv, kontra musiksysteem som skapar ny musik i stunden. Han menar att kompositörer för dataspelsmusik alltid har brottats med sätt att få musik att bli mer adaptiv men ändå behålla sin narrativa funktion. Ett av musiksysteem, istället för att spela upp korta förinspelade musikklipp, har ett slumpmässigt "startvärde" (seed) som matas in i en algoritm vilket genererar en form som noterna följer. Detta startvärde används sedan för att återskapa samma form och rytm i en annan harmonisk kontext, som systemet då korrigerar till att passa in i kontexten. Som exempel visas en melodi som behåller samma form och rytm, men när ackordharmoniken förändras rättas formen så att noterna som spelas passar in i ackordet (Lopez Duarte 2020). Detta resulterar i ett musiksysteem som spelar ett helt nytt stycke varje gång det startar. Enligt honom kan mindre adaptiva musiksysteem betraktas som "unintentional filling of auditory space" (Lopez Duarte 2020, s. 39) om musiken är för repetitiv. Vidare förklarar Lopez Duarte (2020) att konst är för det mesta skapat av människor för människor. Det är oförutsägbart vilket emotionellt uttryck musiken kommer ha i förtid. Han menar också att en avsaknad av kreativ riktning kan betraktas som ett kreativt val.

## 2.2 Implementationsverktyg

Implementationsverktyg är hjälpmedel som underlättar implementationen av ljud och musik till dataspel. "Mellanprogram", också känt som "middleware", är ett program eller mjukvara

som används som en brygga mellan två olika program. I fallet av dataspelsljud finns till exempel *FMOD Studio* (Firelight Technologies 2013). Det marknadsförs av företaget som ett program som effektiviserar implementationen och gör att musikern eller ljudläggaren kan fokusera på ljudläggningen i stället för programmeringen av ljudsystemen. Samtidigt menar de att deras API (Application Programming Interface) är lätt att förstå och data-driven, vilket minskar gapet mellan ljudläggare eller musiker och programmerare.

Det är möjligt att implementera dataspelsmusik utan användandet av mellanprogram. Dessa verktyg finns till för att underlätta implementationen. Utöver två vanliga exempel som *FMOD Studio* (Firelight Technologies 2013) och *Wwise* (Audiokinetic 2006) finns det mindre kända mellanprogram som *Miles' Sound System* (Epic Games Tools 1991) vilket har använts i över 7200 spel (Rad Games Tools 2024).

En av de vanligaste teknikerna inom implementationsverktyg för att göra musik i dataspel adaptiv är genom "vertical layering" i kombination med "horizontal resequencing" (Mc Glynn 2023). Vertical layering innebär att det skapas musikstycken som kan delas upp i individuella spår (Raybould 2011) och sedan beroende på vad som händer i spelet kan dessa spår mixas upp eller ner i ljudvolym för att exempelvis öka intensiteten vid ett givet tillfälle. Horizontal resequencing innebär att övergångarna i musiken sker i relation till spelets tillstånd (Mc Glynn 2023; Raybould 2011; Horowitz & Looney 2014), exempelvis om spelaren är i aktiv strid eller inte. Då kan musiken byta mellan dessa delar "horisontellt", framåt eller bakåt på musikens tidslinje. Horowitz & Looney (2014) förklarar också hur horisontell logik kan gömma övergångar mellan musikstycken genom att till exempel spela en stinger över dem.

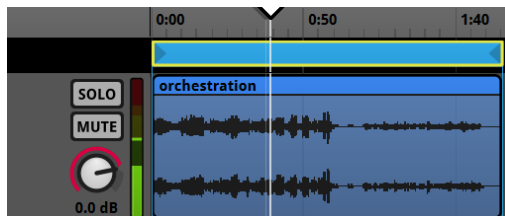
Transitions (Övergångar) är ett sätt att gå från en del av musiken till en annan. Detta är viktigt i en spelmusikalisk kontext eftersom musiken ofta ändras beroende på speltillstånd (game state). Det går inte att förutspå när speltillståndet kommer ändras och därför måste spelets musik kanske ändras på olämpliga platser (Raybould 2011). Till exempel kan speltillståndet ändras när spelaren gör någonting, och det är någonting som för kompositören är oförutsägbart när det kommer hända.

Looping är en teknik som vanligen används vid komponerandet av dataspelsmusik. Det används för att få ut mer av samma stycke musik genom att börja om från början av samma stycke utan att spelaren nödvändigtvis märker det, det vill säga att musiken omärkbart börjar om (Marks 2017). Marks (2017) hävdar att det är en av de bästa tekniker för att spara prestanda, och att det är en kompromiss mellan spelutvecklare och kompositör för att spelutvecklaren får mer musik ut från samma stycke om det upprepas. En loop kan vara lagrad i musikmotorn utan att ta vidare resurser från processorn eftersom mer musik inte behöver laddas in.

Looping är även en produkt av speldesign eftersom utvecklaren och kompositören inte vet när musiken kommer gå vidare (Marks 2017). Detta betyder att musiken måste ha ett sätt att bryta sig ur loopen eller bygga vidare på den. Marks (2017) menar då att spelsektioner, menyer, med flera är därför perfekta placeringar för loops. Looping är inte heller svårt, menar Marks. Loopande musik i sin råaste form är att musiken är klippt på precis rätt ställe efter en takt för att sedan börja den direkt igen så att det inte är märkbart (Marks 2017).

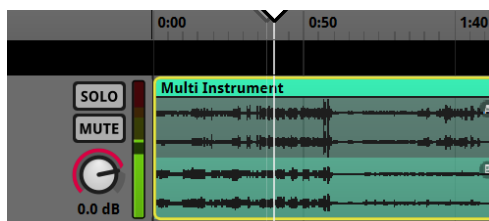
Implementationsverktygen, och mer specifikt mellanprogram, underlättar för looping på flera sätt. Ett exempel är att *FMOD Studio* (Firelight Technologies 2013) har "Loop regions" (Firelight Technologies 2026) (Se figur 1.) där två punkter på tidslinjen definieras. När ljudet går in i regionen loopar den tills att ljudläggaren eller kompositören väljer att ändra musiken.

Detta kan göras på flera sätt men ett exempel är att ljudläggaren definierar en "Transition Marker" (Övergångsmarkör) (Firelight Technologies 2026) som "aktiveras" när en bestämd del av spelets logik har uppfyllts. När musiken sedan kommer fram till den aktiverade markören i loopen bryter den sig ur och fortsätter spela annan musik eller ett bra avslut på den som kompositören har definierat som destination för markören. På så sätt kan utvecklaren kontrollera vad som måste uppfyllas för att musiken ska fortsätta och förändras.



**Figur 1:** En Loop Region i FMOD Studio (markerad i gult).

Randomisering betyder flera olika saker i implementeringskontext. Det kan betyda att ljudmotorn väljer ett ljudspår att spela slumpmässigt med till exempel ett Multi Instrument i *FMOD Studio* (Se figur 2), eller att ljudmotorn transponerar upp eller ner ljud med en parameter. Det används för att få variation i ljud och musik genom att motorn slumpmässigt väljer vad som ska spelas och hur eftersom looping utan variation riskerar att bli enformigt.



**Figur 2:** Ett Multi Instrument i FMOD Studio. Det är inställt på att välja ljudspår slumpmässigt (shuffle).

Mycket av den här bakgrunden har gått igenom mellanprogram (FMOD, Wwise) men det är även viktigt att ta upp spelmotorernas inbyggda lösningar för implementerandet av ljud och musik, vilka funktioner som finns och hur dessa skiljer sig från mellanprogram. Som ett exempel har Unity enkla supportfunktioner för spelljud. Den har "Audio Sources" och "Audio Listeners", där den förstnämnda är ljudkällan (var den finns i spelvärlden) och sistnämnda är ljudlyssnaren (var lyssnaren finns i spelvärlden) (Unity Software Inc. 2026a). Relationen mellan dessa två är vad som skapar tredimensionella ljud för spelaren. Unity har även stöd för reverb med "Reverb Zones" (platser i spelvärlden där det ekar) och flera audio filter för att modifiera ljud i realtid (Unity Software Inc. 2026a). Audio Sources går också att ställa in så de loopar när ljudet tar slut (Unity Software Inc. 2026b). Utöver dessa inbyggda lösningar kan egna lösningar för ljud programmeras.

## 3 Problemformulering

Engström (2020) skriver att det inte har gjorts mycket forskning på ljuden i spel från ett utvecklarperspektiv. Mycket av forskningen som gjorts är från ett analysperspektiv. Därmed behöver forskningen utökning eftersom det inte har undersökts mycket på ljudimplementationen i spelforskning.

Syftet med studien är att fördjupa förståelsen för implementeringsverktygens roll i komponerandet av dataspelsmusik. Eftersom implementationsverktyg för dataspelsmusik används i hög utsträckning inom spelindustrin finns det ett värde i att undersöka hur dessa påverkar komponerandet. Därför kan undersökningen leda till meningsfulla insikter som kan vara till gagn för dataspelsindustrin och dataspelsvetenskapen som helhet.

I litteraturen som presenterades i bakgrundskapitlet har implementeringen av dataspelsmusik redogjorts för utifrån ett historiskt och analytiskt perspektiv samt hur implementationsverktygen påverkar *kompositionen*, men inte hur verktygen påverkar *komponerandet* av musiken. I denna studie ämnas det att undersökas: Hur påverkar digitala implementationsverktyg kompositionprocessen och hur motiverar kompositörer deras val av verktyg när de komponerar musik för spel?

### 3.1 Metodbeskrivning

För att undersöka frågeställningen har tre yrkesverksamma dataspelsmusikkompositörer intervjuats via Zoom (Zoom Communications, Inc 2026). Kvalitativ forskningsmetod valdes eftersom komponeringen av musik är subjektiv. Varje kompositör ser komponering på ett annorlunda sätt. I studien undersöks hur några kompositörer tänker. Intervjuformatet möjliggör att gå in på djupet med vilken musik som komponerats och implementerats och varför. Det gör även att kompositören kan förklara alla sätt som de använt implementationsverktyg i en öppen diskussion.

För urvalet av respondenter har subjektivt urval och snöbollsurval använts. Respondenterna valdes ut baserat på deras erfarenhet i spelindustrin som kompositör samt att de har använt implementeringsverktyg vid skapandet av musiken till ett eller flera dataspelsprojekt. Ett mejl skickades ut till samtliga respondenter med information om vad undersökningen innebär, vilken tidsram som är aktuell samt länk till mötet. I slutet av varje intervju frågades respondenterna om de känner till någon annan dataspelsmusikkompositör som de tror kan ställa upp på en likadan intervju, därmed snöbollsurvalet. Det var inte säkert att snöbollsurvalet skulle leda till fler lämpliga respondenter vilket är varför urvalsprocessen kombinerades med subjektivt urval.

Som förberedelse inför första intervjun genomfördes en pilotintervju för att upptäcka potentiella brister i frågor och format. Intervjuerna genomfördes med en respondent i taget av en intervjuare enligt Ejvegårds (2009) rekommendationer. Intervjuerna spelades in med hjälp av Zooms (Zoom Communications, Inc 2026) inbyggda inspelningsverktyg. Efter intervjuerna hade transkriberats skickades dessa ut till berörd respondent för att säkerställa att denne är nöjd med innehållet.

Enligt Denscombe (2018) är den kvalitativa forskningsansatsen ofta förknippad med småskaliga studier eftersom man inte behöver samla in lika stor mängd data för analys, jämfört med kvantitativa studier. Därför var det lämpligt att denna forskningsansats användes för

studien eftersom tidsomfånget var begränsat till tio veckor och att databearbetningen gjordes av två personer. Kvalitativa forskningsprojekt handlar ofta om få människor eller händelser vilket gör att dessa fenomen undersöks mer ingående. Den kvalitativa forskningsansatsen använder även text och bilder som analysenhet, vilket detta projekt har gjort.

Intervjufrågorna genomfördes semistrukturerat: vissa frågor var samma för alla respondenter och vissa frågor talades det olika mycket om. Detta underlättade för respondenten att fritt förklara tankar utan att bli styrd åt ett visst håll. För att underlätta jämförelsen mellan respondenterna har vissa frågor ställts i samma ordning.

En svaghet med intervjumetoden är att svaren inte bygger på sakliga fakta utan bygger på respektive persons upplevelser, vilka inte är mätbara. Detta innebär att datan riskerar att bli svårtolkad vilket ytterligare understryker vikten av granskningen av materialet och tolkandet av datan. En utmaning är att förbli opartisk vid analysen av data och en garanti på detta kan aldrig ges. Ejvegård (2009) belyser också vikten av att vara förberedd inför intervjun och inte vara nervös eftersom detta kan smitta av sig på respondenten. Han pålyser även att intervjuer kan ta tid att sammanställa och tolka vilket gör att valet av respondenterna behöver göras med omsorg.

För att hålla intervjuerna semi-strukturerade gjordes en "intervju-guide" med frågor till respondenterna samt potentiella fördjupningsfrågor. Denna guide reviderades efter pilottestet där det uppdagades vilka frågor som fungerade bättre än andra och i vilken ordning de kom i. Detta ökade översikten något men var fortfarande problematiskt att kunna hoppa mellan frågor och ämnen flytande. Därför reviderades guiden igen till ett digitalt dokument där varje fråga kunde strykas och flödet av konversationen kunde lättare upprätthållas.

Exempel på frågor som ställts i intervjun är: "Vilken bakgrund har du? Vad har du arbetat på tidigare i din karriär? Hur mycket tänkte du på musikens tekniska funktion i spelet? Hur är musiken tänkt att interagera med spelaren? I vilket skede börjar man tänka på kompositionen i termer som implementering? Vilken roll spelar implementeringsverktygen när man arbetar iterativt?". Det förekom även frågor av bredare natur som att kort beskriva en aspekt av implementationen, exempelvis "hur gör man för att effektivt testa hur musiken fungerar i ett dataspel?". Det fanns även frågor som gick in på någonting mer specifikt, "Hur går det till när man ska implementera musiken som komponerats?". Frågornas syfte är att ge en överblick på respondentens erfarenhet med dataspelsmusik, undersöka hur respondenten gått till väga för att skapa musiken samt hur implementationsverktygen påverkat deras sätt att skapa denna musik.

### **3.1.1 Transkribering och kodning**

Efter intervjuerna transkriberades materialet med hjälp av intervjuinspelningen och ett transkriberingsverktyg som underlättade det tidskrävande arbetet genom att det skapade en text som korrekturlästes istället för att transkriberingen gjordes manuellt. Efter att transkriberingen av alla intervjuer genomförts har datan sammanställts och tolkats. Detta gjordes genom att leta efter nyckelord och teman i innehållet hos respektive respondent. Dessa utgör underlaget för slutsatsen. Före, under och efter intervjun har respondenterna informerats om att alla svar hanteras konfidentiellt och inte kommer publiceras offentligt utan dennes samtycke.

När pilottestet var genomfört transkriberades denna för hand vilket tog längre tid än planerat. Därför användes den inbyggda transkriberingsfunktionen i appen *Samsung Voice Recorder*

(Samsung Electronics 2026) för att underlätta och snabba på denna process genom att lokalt transkribera intervjun med hjälp av AI-teknologi, vilket gav en större struktur av intervjun. Transkriptionen var inte felfri och granskades ord för ord.

För att analysera svaren har varje svar kategoriserats in i en tabell med intervjufråga och intervjusvar med respektive respondent. Frågorna tilldelades varsin färg som användes för att markera respondenternas svar i den fullständiga transkriptionen. Detta möjliggjorde att enklare kunna hänvisa till vad som sades när i intervjun samt i vilken kontext.

De fullständiga transkriberingarna av intervjuerna bearbetades till en tabell med samtliga frågor med tillhörande svar från respektive respondent i tre separata tabeller. Tabellen innehåller direkta citat från intervjun samt korta sammanfattningar av svaren. Vissa frågor ställdes inte uttryckligen till respektive respondent, men kategoriserades ändå under den tillhörande frågan. Då en annan fråga formulerades inkluderades denna bland svaren. Denna sammanställningstabell skickades ut till respektive respondent för att kontrollera behovet för eventuella korrigeringar. Endast det transkriberade innehållet som inkluderats i rapporten skickades ut. Detta material godkändes av respektive respondent efter att eventuella korrigeringar genomförts.

Därefter tolkades svaren och sammanställdes i ett ”Gioia-inspirerat” tolkningsdiagram (Gioia, Corley & Hamilton 2013). Detta består av tre kolumner: första ordningens koncept, andra ordningens teman och övergripande dimensioner. Första ordningens koncept är väldigt nära deltagarnas egna svar på frågorna i intervjun. Andra ordningens teman är en tolkning av flera sammantagna första ordningens koncept. Övergripande dimensioner är en tolkning av flera andra ordningens teman till en större abstraktion av innehållet. Detta ökar studiens transparens och spårbarhet då man ser flödet från data till abstraktion.

### **3.1.2 Forskningsetik**

Eftersom denna studie involverar människor är det extra viktigt att god forskningsetik följs. Vetenskapsrådet (2024) presenterade en ny rapport om god forskningssed där det etableras riktlinjer, vilka efterföljs i den här studien. Detta innebär att respektera och upprätthålla deltagarnas autonomi, att deras autonomi är fullständig och informerad samt att de hålls informerade under arbetets gång om någonting relaterat till hanteringen av deras data förändras. Deltagarna har även rätt att återta sitt samtycke när som helst, och därmed få all deras tillhörande data raderad. De kan också när som helst återta sitt samtycke till att deras namn används i studien, i vilket fall de anonymiseras eller att informationen inte används. Namnens inkludering i rapporten bryter mot forskningsetiken, men detta motiveras genom att datan ska ha kredibilitet. Deltagarna har samtyckt till att namnen är inkluderade och har godkänt all data som har samlats in efter korrigeringar från dem.

## 4 Genomförande och Analys

### 4.1 Förberedande arbete

Projektet planerades utifrån uppskattade tidsramar som de olika momenten skulle behöva, kontakta och boka tid med respondenter, göra en intervjuguide med frågor och följdfrågor, genomföra pilotintervjun, genomföra intervjuer med respondenterna, transkribera intervjuerna, koda och tolka datan, skicka ut och revidera transkriptionen till respondenterna samt sammanställa all data när denna är på plats. Valet av respondenter gjordes utifrån deras erfarenhet med att skapa dataspelsmusik och deras kontaktuppgifter var tillgängliga.

Efter att ha undersökt hur lång tid en transkription tar av en timmes intervju uppdagades det att det kan ta ungefär åtta timmar per timme av intervjumaterial. Detta var utifrån att inget AI-transkriberingsverktyg tänkte används.

När det bestämts att det endast kommer genomföras en intervju om dagen gjorde det att planeringen kunde göras mer omfattande och därmed underlätta för tidsupplägget utmed projektet.

Först behövde respondenter kontaktas. Detta gjordes med ett mejl där nödvändig information angående undersökningen formulerades. Därefter formulerades frågorna till intervjudelen och komprimerades till en lista. Samtidigt som listan utvecklades kontaktades pilotintervjurespondenten med relevant information gällande tid, syfte och innehåll av intervjun.

### 4.2 Pilotintervju

Pilotintervjun har genomförts för att testa hur väl frågorna och strukturen på intervjun fungerade. Detta gjordes över *Zoom* (Zoom Communications, Inc. 2026) och deltagaren var en lärare i musikimplementation på Högskolan i Skövde. Intervjun spelades in med hjälp av Zooms inbyggda inspelningsverktyg för att sedan transkriberas.

Intervjufrågorna formades efter kontext och vad som talats om tidigare i intervjun. De var ett verktyg för att styra samtalet med, snarare än att vara fastställda. Detta för att få en mer naturlig konversation som respondenten kunde använda som diskussionsunderlag.

Under intervjuns gång reflekterade intervjuaren över konversationen och försökte styra den mot andra frågor. Detta betydde att det blev en naturlig konversation som underlättade för fria tankar. Efter pilottestet transkriberades innehållet för hand. Detta är i kontrast till de följande intervjuerna eftersom transkriberandet tog lång tid.

I pilottestet uppdagades det frågor som upplevdes ledande. Dessa frågor reviderades inför nästkommande intervjuer. I slutet av intervjun frågades deltagaren även om det fanns någon annan som hade kunnat tänka sig ställa upp på denna intervju. Därigenom kom tipset till vår första respondent efter pilotintervjun.

### 4.3 Intervjuer

Intervjuerna genomfördes alla via *Zoom* (Zoom Communications, Inc. 2026) med intervjuare, antecknare och respondent närvarande. Inspelningen gjordes på både intervjuarens och antecknarens dator. Antecknaren presenterade sig kort inför intervjun och lämnade sedan över ordet till intervjuaren för i. Kameran på respondent och intervjuare var på utmed hela

intervjun. I samband med introduktionen gavs en sammanfattning av vilka som genomför studien och vad som undersöks utan att färga respondenternas förväntningar om vad de borde säga eller inte. Respondenterna fick även kort information om hur innehållet skulle hanteras. Det hölls aldrig mer än en intervju per dag.

Alla frågor inför intervjuerna var utformade så att respondenterna fritt kunde berätta om hur de bemöter olika problem som uppstår vid komponerandet av dataspelsmusik. Detta resulterade i att frågorna som ställdes till respektive respondent skiljer sig åt något i omfattning, innehåll samt ordning.

Första frågan som alla respondenter besvarade var att berätta om sig själva och sin bakgrund som kompositör. Detta gav en överblick på vilka respondenterna är och varför de är relevanta för att närmare undersöka frågeställningen.

Respondenternas bakgrund är av intresse eftersom komponering av musik samt sättet man bemöter och löser problem i dataspelsmusik är högst individuellt. Med, i teorin, oändliga möjligheter och lösningar på dessa problem ansågs det relevant att få en överblick på respondenternas bakgrund som kompositörer. Inspelningarna av intervjuerna har endast använts till att kunna transkribera intervjuerna i efterhand.

Respondenterna är alla verksamma inom spelutveckling och har arbetat på spelprojekt i olika omfattning. Efter att respondenterna berättat lite om sig själva och vilka projekt de jobbat på frågades det mer ingående om ett specifikt projekt och hur de där löst eventuella problem. Därefter ställdes frågor om något mellanprogram använts för detta projekt och hur arbetet har sett ut.

### **4.3.1 Tobias Kalliokulju**

Den första respondenten var kompositören Tobias Kalliokulju (Appendix A). Han har arbetat med olika dataspelsprojekt både i samband med sin utbildning, professionellt och på fritiden. Sedan tidig ålder har Tobias haft dataspelsintresse, musikintresse och spelat instrument. Tobias har arbetat med olika spelmotorer och ljud-mellanprogram samt undervisar i komposition för dataspelsmusik. Tobias har studerat jazz på folkhögskola, kandiderat och läst master på Kungliga Musikhögskolan där han tog examen 2025 och därefter varit yrkesverksam i några månader som spelmusikkompositör. Tobias har studerat kompositionsprocessen i dataspel kopplat till implementering utifrån ett konstnärligt forskningsperspektiv. Musiken i spelet som diskuteras ingående i "fråga två" (Appendix A) komponerades som en del av hans masterarbete. Utöver dataspel har Tobias även komponerat musik till andra medium som exempelvis teater.

Efter introduktionen i intervjun ställdes frågor om ett specifikt projekt som Tobias arbetat på. Frågorna berörde vilken typ av musik som skapats till projektet samt hur det arbetats med att implementera musiken i dataspellet. Projektet använde spelmotorn Unity och använde även Wwise men bytte till FMOD senare i projektet för att implementera musiken i projektet.

Tobias spelade in musiken till projektet i en studiomiljö tillsammans med andra musiker. I kombination med förberett musikaliskt material som Tobias tagit fram improviserade musikerna i studiomiljön till dataspellet som kördes i bakgrunden på en skärm i studion. Det förberedda materialet inkluderade ett huvudtema, ett seglingstema och två stingers. För detta projekt ville de inte bestämma för mycket på förhand hur musiken skulle låta eller exakt hur alla instrument skulle spela. Detta improviserades till stor del fram i studion tillsammans med

musikerna. Det inspelade materialet bearbetades med hjälp av ett verktyg som heter ”≡ AMAPP”. Denna app underlättade för implementeringen av musiken som spelats in för att sedan snabbare kunna testas i dataspelet. Programmen som användes vid inspelningen av musiken för projektet möjliggjorde en kompositionsprocess i en studiomiljö som hade kortare intervall mellan varje iteration jämfört med projekt som inte använder sig av dessa program. Den snabba iterationsprocessen gjorde att de snabbt kunde utvärdera resultatet genom hela produktionslinjen: från inspelning av instrument till implementerat i spelet. Till följd av att musikerna i studion improviserade till spelet uppkom det ett nytt musikaliskt tema, vilket möjliggjordes av projektets arbetsflöde i studion (Se figur 3).

Fråga:	Första ordnings koncept:	Andra ordningens tema:
Hur arbetade du/ni med att få in musiken i spelet?	Uppkom ett nytt musikaliskt tema till resultat av arbetssättet.	Improvisation Mjukvarusamhörighet Snabba iterationer

**Figur 3:** Tematisk tolkning av Tobias svar.

Vidare ställdes frågan om vilken roll implementationsverktyg har när man arbetar iterativt. Tobias svarade att det kan ta väldigt lång tid att utvärdera hur musiken fungerar och känns i ett spel om man inte själv är involverad med implementationen av musiken. Med projektet i föregående stycke som exempel, pålyste Tobias hur mycket snabbare den iterativa processen var i det projektet när möjligheten att testa hur musiken låter och känns i spelet tack vare programmen som användes vid inspelningen och implementationen av musiken till projektet. Om man som kompositör inte är involverad med implementationen av musiken i ett dataspelsprojekt kan tiden mellan att man gjort musiken och att den testas och utvärderas i dataspelet ta veckor innan man får återkoppling. Det kan dessutom då visa sig att musiken inte riktigt lyckas uppnå det den är tänkt att göra vilket innebär att det kan ta några veckor till innan nästa återkoppling. Tobias menar även att denna process inte uppmuntrar spelföretag att spela in live-musiker i en studio. Oftast vid inspelning av live-musiker i en studio har materialet förberetts i detalj för att hålla kostnaden nere och effektivisera inspelningen.

Tobias säger även att en kortare iterativ process ger kompositören fler möjligheter att iterera på musiken vilket kan leda till en ökad kvalitet av den men också ge mer tid till att skapa musik. Han tycker även att mellanprogram samt kontrollen man har över implementeringen spelar en viktig roll i kontrollen över det kreativa slutresultatet (Se figur 4).

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vilken roll spelar implementationsverktygen när man arbetar iterativt?	Möjliggör snabba iterationer av musiken. Ökad iteration kan öka musikkvalitén Förenklar delaktigheten för kompositören i implementationen	Snabba iterationer Ökad musikkvalité Förenklad delaktighet

**Figur 4:** Tematisk tolkning av Tobias svar.

Vidare i intervjun ställdes frågan: hur mycket tänker du på hur musiken ska fungera i spelet redan när du komponerar? Tobias svarade ”[...] jag tycker inte att det går att skriva musik utan att veta hur den kommer att fungera i spelet.” Han fortsätter ”[...] både ur ett kreativt perspektiv och hur den behöver vara strukturerad i exporten, eller kommer det vara ett intro

som sen går över till en loop? [...] Det är många variabler som vänder upp och ner på hur man borde skriva om det är det ena eller det andra”.

Han beskriver även att han brukar ställa väldigt många frågor till ansvariga i projektet som exempel vad musiken fyller för funktion och vad kompositören vill att spelaren ska känna. Finns det inte svar på någon av de frågorna borde det inte vara någon musik vid det givna tillfället säger Tobias. Han säger även ”Jag brukar så mycket jag kan, innan jag ens skriver en ton, försöka förstå vad det är jag skriver till”.

Han belyser även vikten vid att kommunicera över disciplinränser eftersom det lätt kan uppstå missförstånd när man förklarar saker med termer andra inte förstår. I ett exempel från ett tidigare Game Jam spel beskriver Tobias när han gjorde musik till projektet utan att vara inblandad i spelmotorn och hur det skiljde sig åt från att implementera själv. Där beskriver han att det då kändes som om att man var låst och utanför och hade svårt att förstå hur dels koden i projektet var strukturerad och vilket stadie spelet är i och om det borde vara en musikdel med variationer i sig eller två helt olika musikdelar på två olika tillfällen. Detta påminde Tobias hur mycket det underlättar att själv vara involverad i spelmotorn och implementeringen av musiken. Med mellanprogram kan man bli mer självständig och arbeta i sitt ”eget ekosystem” menar Tobias, trots detta är det väldigt viktigt att ha kontinuerlig kontakt med utvecklaren pålyser han (se figur 5). I detta projekt blev det missförstånd på vilken musik som skulle spelas var i spelet, men det upplevdes som väldigt positivt av utvecklarna i projektet att musiken var adaptiv på sättet som Tobias insisterade på att få den bli.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Hur mycket tänkte du på hur musiken interagerar med spelet och/eller spelmotorn när du komponerar?	Medveten om musikens funktion och roll innan komponering äger rum. Musikens musikaliska struktur Fråga många frågor till utvecklare. Vara involverad med implementeringen personligen.	Musikens roll Musikens funktion Musikaliska struktur Frågeställande Kommunikation

**Figur 5:** Tematisk tolkning av Tobias svar

Vidare diskuterades frågor angående uppfattningen hos andra hur musik i spel bör vara och graden av interaktivitet. Tobias säger att tidig projektinvolvering kan möjliggöra idédiskussion med övriga i projektet angående musikens interaktivitet, vilket kan ge dem en bättre förståelse för musikens värde. Han tycker även att om man som kompositör tycker om interaktiv musik borde man också vara involverad med implementeringen av den. Att använda mjukvara som kräver en viss typ av kompetens kan innebära en risk för mindre spelutvecklare i form av att åsidosätta resurser till att realisera systemen som krävs, men detta kan undvikas om kompositören implementerar och behärskar programmet själv.

Återigen påtalades vikten av iterationer av musiken; ”Ju fler iterationer man hinner med för att man själv är involverad i systemen själv, desto mer tid kan man lägga på att skriva musiken”. Dessutom kan detta göra att det blir färre omfattande missförstånd mellan kompositör och utvecklare, menar Tobias.

Vidare ställdes frågan vilka skillnader det finns med att jobba med kontra utan mellanprogram när man gör musik. Påverkar det hur man komponerar och i så fall hur? Där säger Tobias att i projekt utan mellanprogram behöver tid läggas på att bygga system och funktionalitet som

redan finns i mellanprogram. Det gör att mer tid kan ägnas åt att skriva musik när man använder mellanprogram eftersom tiden ägnad på att bygga system som redan finns inbyggt i mellanprogrammet minskas. Tobias använder mest FMOD och säger att det är "DAW"-liknande (Digital Audio Workstation) vilket är likt andra program han är van vid att arbeta med. Även underlättningen av att arbeta självständigt i ett mellanprogram utan att störa den större versionshanteringen i ett spelprojekt är uppskattad. Vidare beskrivs olika projekt Tobias arbetat på som använder olika spelmotorer och implementationsverktyg och att han upplevde att processerna skilde sig mycket åt. Exempelvis kunde man snabbt få ut fungerande prototyper med FMOD och Unity medan med Godots inbyggda ljudfunktioner tog det lång tid att bara bygga rätt system. " [...] tiden man kan lägga på att skriva musik och faktiskt testa det är ju jättestor skillnad". Tobias understryker också svårigheten med att jämföra tre olika konstnärliga processer.

Ett påstående ställdes efter detta för att förtydliga om det är tiden som man generellt sätt tjänar mest på och ifall Tobias höll med om det. Som svar betonade Tobias iterationsmöjligheternas vikt ytterligare, där varje iteration som i teorin gör saker bättre. Vidare sade han att man inte alltid vet hur musiken ska låta eller fungera i ett sammanhang. Tobias liknar komposition med en viss "långsam improvisation". Kompositören vet inte alltid i förväg hur musiken ska vara i detalj. " [...] men varje test ger information som leder en vidare". Han belyser även att man som kompositör sällan träffar rätt med musiken på första försöket och behöver testa och utvärdera i ett cykliskt förhållande.

I samband med denna diskussion påtalade Tobias att olika mellanprogram kan generera nya idéer när man försöker förstå sig på olika funktioner som programmet har: "Att verktygen vi använder är ju också oscillatorer för den konstnärliga processen" (Se figur 6).

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vad är din uppfattning om andras uppfattning av musikens roll i spel?	Tidig involvering underlättar för att den kreativa visionen för musiken förstås av andra. Involverad i implementeringen som kompositör.	Tidig involvering Komponera & implementera
Varför har du använt dig av dem och vad är det du uppskattar med dem?	Mer tid åt att komponera musik. Underlättad versionshantering. Underlättar självständig iterering. Snabbt prototypa idéer. Liknande interface som "DAW". Inbyggd funktionalitet i middleware:n.	Mer kompositionstid Snabb prototyp Självständig iterering Programutformning Färre missförstånd Inbyggd funktionalitet
Hur får verktygen dig att tänka på komponerandet av musik?	Olika processer i olika program. Verktygens funktionalitet ger inspiration till nya idéer.	Idéoscillator Tillvägagångssätt varierar.
Vilken roll spelar implementationsverktygen när man arbetar iterativt?	Ju fler iterationer desto mer tid åt att skapa musik. Minskar missförstånd mellan kompositör och utvecklare. Lättare att träffa rätt med musiken med fler iterationer.	Mer kompositionstid Bättre visionskommunikation Träffsäkrare slutresultat

**Figur 6:** Tematisk tolkning av Tobias svar.

### 4.3.2 Gustaf Blix

Den andra respondenten var kompositören Gustaf Blix (Appendix B). Han har länge haft både datorspels- och musikintresse och spelat instrument i grupper som tonåring då han också började skriva musik på datorn. Gustaf har studerat klassisk komposition på universitetsnivå. Han har arbetat med akustiska symfoniorkestrar med både originalkompositioner och arrangering av andras verk. Utöver detta har han även frilansat som kompositör och för videoproduktion, arbetat som lärare och arbetat med olika spelprojekt. Vissa projekt med och vissa utan finansiering.

Intervjun fortsatte efter introduktionen att beröra hur musiken gjordes för ett särskilt projekt, likt i första intervjun. Här ställdes samma fråga om vad för typ av musik som gjorts till det spelet samt hur det implementerades i spelet. Där uppdagades det att redan i början av komponerandet hade programmeraren för projektet en tanke på vilken funktion musiken kunde ha för spelet. Idén var att gestalta olika karaktärers karaktärsdrag i spelet genom olika instrument. Gustaf tycker att det var en bra idé på modulär musik som programmeraren gav som förslag.

Vidare berättar Gustaf att han komponerat musiken för spelet genom att ha olika lager med instrument som alla kan kombineras i vilken kombination som helst vertikalt. Musiken i spelet reagerar på kunden spelaren betjänar genom att ändra instrumenteringen. Det är samma låt hela tiden, förutom när man möter bossen, men instrumenten matchas med kunden. Eftersom det är samma låt hela tiden nämner Gustaf att det är viktigt att musiken av den anledningen inte blir ”enformig” eller ”tråkig”. Det är 39 olika spår av musik som alla fungerar tillsammans i vilken kombination som helst i samma tempo. I fasen där omeletter lagas finns det ”lite stressig jazzgitarr”. Gustaf har endast komponerat musiken den för detta projekt och inte implementerat den.

För projektet har musiken implementerats direkt i spelmotorn utan mellanprogram. Gustaf beskriver hur det varit mycket arbete med att få musiken att fungera så som den var tilltänkt. Det var svårt att få en unison förståelse för hur musiken skulle bete sig i spelet mellan programmerare och kompositör. Gustaf själv är inte programmerare och har därför inte lika omfattande insyn på programmeringsrelaterade problem som projektets programmerare. Eftersom projektet inte använde ett mellanprogram frågade intervjuaren om Gustaf fick några direktiv till hur han skulle skapa musiken. Utöver att musiken skulle fungera i olika lager i vilken kombination som helst var där inga andra specifika instruktioner eller önskemål från utvecklaren. Gustaf berättade sedan om när han förklarade för programmeraren att när man loopar musiken så klipps reverb-svansen av abrupt. Gustaf förklarade hur man kan lösa problemet med ett mellanprogram och ville då veta om man kunde göra en liknande lösning direkt i Unity (se figur 7). Detta slutade med att Gustaf behövde göra om alla stems genom att klippa av reverb-svansen och lägga den i början av respektive musikspår och balansera volymen så att den inte är för uppenbar första gången man lyssnar på den, beskriver han.

Gustaf beskriver att när man inte implementerar musiken själv så blir det inte alltid som man har tänkt sig. Det kan vara svårt att få personer som inte är musiker att förstå vad man menar samtidigt som man lägger mer arbete på dem att hitta en lösning på ett problem som de inte fullt ut förstår. Då är det bättre att man som musiker försöker hitta en lösning som fungerar för projektet, vilket är schystare mot dem säger Gustaf.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Hur arbetade ni/du med att få in musiken i spelet?	Kommunikation mellan programmerare och musiker. Svårt att förstå varandra. Inte använt middleware. Kommunicerat med hjälp av middlewares ramverk. Inte implementerar musiken själv kan bli besvärligt.	Kommunikation Missförstånd Merarbete

**Figur 7:** Tematisk tolkning av Gustafs svar.

Vidare i intervjun ställs frågan vilka mellanprogram som Gustaf har erfarenhet av, vilket han svarar att han har erfarenhet av både FMOD och Wwise. Han säger även under intervjun att han arbetar på ett projekt som använder Wwise nu. Följdfråga som ställdes till Gustaf var varför han har använt mellanprogram när han gjort det. Då svarar Gustaf med att i ett projekt användes det för att kunna bygga ett musiksysteem som ändras beroende på om spelaren är i närheten av fiender.

Därefter ställdes frågan hur FMOD och/eller Wwise får honom att tänka på komponerandet och hur musiken ska läggas upp. Där svarade han att "I stället för att tänka linjärt så kan man tänka modulärt" (se figur 8). Han fortsätter: "Då kan man ha som pusselbitar som kan komma i vilken ordning som helst, kanske flera lager som både horisontellt och vertikalt. Det sättet gör man inte musik på så ofta annars". Därefter fick Gustaf frågan om han upplevde att det är stor skillnad att skapa musik för spel jämfört med den typen av musik han har arbetat på tidigare. Han tyckte att det är väldigt stor skillnad.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Varför har du använt middlewares och vad uppskattar du med dem?	Möjligheten att använda deras inbyggda funktionalitet för adaptiv musik.	Inbyggd funktionalitet
Hur får verktygen dig att tänka på komponerandet av musik?	Tänka modulärt, snarare än linjärt. Stor skillnad att göra musik från andra medier.	Konceptualisering Komponeringsskillnader

**Figur 8:** Tematisk tolkning av Gustafs svar.

Vidare i intervjun diskuterades frågan: "hur mycket tänker du på hur musiken interagerar med spelaren/spelet när du komponerar?". Där svarade Gustaf att han tänker på det varje gång han jobbar (se figur 9). Han gav även ett exempel på ett annat spelprojekt han arbetat på där han hade spelet i gång på en av två skärmar där han komponerade musiken samtidigt på andra skärmen för att få en känsla för spelet och vad som fungerar och inte.

Senare diskuterades hur en typisk planering ser ut för Gustaf för att komponera ett musikstycke för ett spel. Där börjar Gustaf med att hitta musikaliska huvudteman för spelet, därefter vilka typer av ljud som ska användas eller musikstil. Han påtalar även att det är bra att bestämma ifall det ska vara interaktiv/dynamisk musik eller inte. Detta kan vara avgörande för ifall man kommer behöva använda ett mellanprogram eller inte, säger han. Därefter ställdes frågan om han skulle vilja vara med i början av planeringen av ett spelprojekt eller ifall man får ett uppdrag att göra musiken längre in i projektets gång. Han tycker att båda delar fungerar bra men lyfter att han tycker det vore roligt att vara delaktig redan från början, vilket innebär att man får ett ökat inflytande och tycker det är roligt att vara delaktig i de sakerna.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Hur mycket tänkte du på hur musiken interagerar med spelare/spelmotorn när du komponerar?	Medveten om musikens roll i spelet konstant i komponerandet. Komponerar parallellt som spelet är i gång på skärm 2.	Syftesmedveten
Beskrivning av en typisk planering för ett musikstycke för dataspel.	Tidigt delaktigt i projekt vore positivt och roligt. Etablera musikalisk riktning för projektet. Bestämma hur interaktiv musiken ska vara. Kan avgöra användandet av middleware.	Tidig delaktighet Musikriktning Interaktivitetsgrad

**Figur 9:** Tematisk tolkning av Gustafs svar.

Nästa fråga som ställdes var vilken roll han tycker implementeringsverktyg spelar när man arbetar iterativt. Där svarade Gustaf med att vad det gäller mellanprogram så spelar det mindre roll vilket mellanprogram man använder, det som är mest avgörande är *om* man använder ett mellanprogram eller inte. Det avgör hur man kan utforma musiken, säger Gustaf. Ett exempel han nämner är just loop-funktionaliteten utan att klippa av reverb-svansen som mellanprogram har och Unitys inbyggda ljudsystem saknar. En annan aspekt som Gustaf lyfter är att för mindre spelutvecklingsföretag kan licenserna för att använda mellanprogramvara vara ett finansiellt hinder till att använda dem i sitt projekt. Därför tycker Gustaf att det är viktigt att kunna arbeta i spelprojekt utan att vara beroende av ett mellanprogram.

”Det är viktigt att kunna [använda mellanprogram], men det är också viktigt att kunna avstå. [...] det är bättre att vara så enkel att jobba med som möjligt än att jag kunna göra en massa andra coola avancerade tekniska grejer”.  
(Appendix B)

Vidare i intervjun ställdes frågan hur brukar du testa att musiken fungerar i spelet när den är implementerad, hur gör du för att testa att musiken gör vad den ska? Där svarade Gustaf att han brukar testa hur musiken låter direkt i sin ”DAW” på ena skärmen samtidigt som han har spelet på den andra skärmen och testar så att musiken känns och låter bra oavsett var man är i spelet. Sedan påtalar han att det är kritiskt att man faktiskt testar hur musiken låter och beter sig i spelet när den är implementerad via ett mellanprogram. Där beskriver han ett exempel på när han trodde att han gjorde rätt med implementeringen men när han lyssnade på det i spelmotorn visade det sig att några lager som hördes när de inte skulle det. Detta hade inte upptäckts utan att ha testat hur musiken fungerade i spelmotorn.

Vidare påtalas det av Gustaf att det är fördelaktigt om man själv kan implementera musiken så att man får den att låta som tilltänkt (se figur 10). I fall man som kompositör helt eller delvis överlåter implementationen till programmerare finns det risk att det inte kommer att bli eller låta som man har tänkt. Han liknar det med att om programmerare skulle överlåta delar av sin process till en kompositör som inte är programmerare eller förstår det språket kommer det högst troligen bli någonting fel. Där är situationen liknande om man som kompositör överlåter implementeringen till programmerare som kanske inte förstår musik eller hur det fungerar då kan det bli svårt att förklara hur musiken är tänkt att fungera i spelet.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vilken roll spelar implementationsverktyg när man arbetar iterativt?	Största påtagliga rollen är om man har middleware eller inte. Loopfunktionalitet i middlewares som spelmotorer kan sakna. Viktigt att behärska implementering med och utan middleware, vara flexibel	Användandet av middleware Inbyggd funktionalitet Kompositörens flexibilitet
Hur testas man att musiken fungerar som den är tilltänkt?	Bra ifall kompositören är involverad i implementeringen. Risken ökar musiken inte fungerar som tilltänkt vid överlåtning till andra instanser.	Involvering Överlåtning Kreativ vision

**Figur 10:** Tematisk tolkning av Gustaf svar.

Nästa ämne som diskuterades i intervjun var vilka utmaningar man som kompositör för dataspelsmusik står inför. Där beskriver han att spelmusik innefattar ibland helt andra stilar än vad film och teatermusik gör. I spel kan musiken oftare ges mer utrymme i förgrunden, jämfört med film och teater. Om den gör det måste man försöka göra så att det inte blir irriterande att lyssna på hela tiden, man behöver hitta en balans.

En följdfråga som ställdes till Gustaf var hur man undviker att trötta ut spelaren med musiken i spelet. Där svarade Gustaf med att lyssna mycket på det själv och se ifall man tröttnar. Utöver det sade han att "less is more". Hans kompositionsprofessor rekommenderade att lägga till så mycket som möjligt för att sedan plocka bort bit för bit. Sedan nämndes även den interaktiva aspekten som dataspel kommer med. Där sade Gustaf att man som kompositör behöver tänka mindre linjärt och tänka mer i lager snarare än en enda lång tidslinje (se figur 11). Detta menar han inbjuder kompositören till ett nytt sätt att tänka gällande musikkomposition. Gustaf lyfte även fram ytterligare en aspekt han anser är viktigt vilket är att musiken ska vara välproducerad.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vilka utmaningar har man som kompositör när man komponerar musik för dataspel?	Behöver tänka mindre linjärt och mer modulärt. Inbjuder till ett nytt sätt att tänka. Spel innefattar fler musikstilar jämfört med exempelvis film. Lyssnarens irritation i beaktning.	Musikstilvariation Konceptualisering Lyssnarirritation

**Figur 11:** Tematisk tolkning av Gustafs svar.

### 4.3.3 Jacob Westberg

Den tredje respondenten är spelmusikkompositören Jacob Westberg (Appendix C). Han har under sin uppväxt haft ett starkt teknikintresse och tyckte om programmering redan från grundskolan. Jacob har gått på kulturskola där han lärt sig spela instrument. Under gymnasiet tog han extrakurser i musikproduktion. Jacobs akademiska bakgrund är en masterexamen i musikproduktion på Kungliga Musikhögskolan i Stockholm, studerat musikvetenskap vid Uppsala Universitet. Han har även studerat ljudproduktion vid Högskolan i Blekinge där det formats spelutvecklingsteams från parallella program, likt Högskolan i Skövde. Utöver detta

har Jacob studerat programmering, utvecklat appar och har jobbat med spelprojekt i anslutning till programmeringsstudierna. Jacob har frilansat efter studierna där han arbetat på olika spelprojekt. Han jobbar för tillfället för en stor svensk spelutvecklare.

I introduktionen beskriver Jacob att han kände sig låst i sin professionella yrkesroll när han var beroende av programmerare att implementera sina verk in i spelen. Han upplevde att hans musikaliska idéer inte kunde förverkligas när han var tvungen att överlåta implementeringen till någon annan.

Jacob har utvecklat appen "≡ AMAPP" som underlättar kompositionen av interaktiv musik till dataspel. Detta beskriver han som ett verktyg som hjälper kompositören att jobba adaptivt redan i kompositionsprocessen, snarare än att det kommer som ett efterarbete. Han uttrycker även ett intresse av vad som händer med den kreativa processen om man börjar i den änden redan i kompositionsfasen. Vidare förklarar han att det är vanligt för honom att först när han fått ut idén så att han kan lyssna på den som han kan göra bedömningen om den är bra eller inte. Där fortsätter han med att förklara hur adaptivitetaspekten gör denna process ytterligare komplex eftersom man behöver göra musik som låter bra när den rör sig mellan olika moment, variationer eller delar.

Jacob understryker att "≡ AMAPP" inte är ett mellanprogram, utan ett verktyg för att underlätta för komponerandet av adaptiv musik som finns inuti en "DAW". Han säger att man vinner väldigt mycket på att ha en "pipeline" som är snabb att jobba i. "≡ AMAPP" samlar upp, taggar upp och renderar ut filerna till att bli en "asset" som kallas för "kluster". Detta gör att man exempelvis i kompositionsfasen kan lyssna på olika interaktiva lager på ett smidigt sätt i sin "DAW". Han berättar att när man jobbar i en "tidslinje" måste man rendera ut alla olika items, stems och ljudfiler och sedan hålla isär alla dessa, vilket kan bli väldigt besvärligt. När man renderar ut olika intensiteter av ett vertikalt lagrat musikaliskt verk kan det lätt bli fel som exempelvis att något spår var tystat när det inte skulle vara det. "≡ AMAPP" underlättar med att hålla isär och strömlinjeforma den processen. En sak som Jacob menar då blir lite konstig är att jobba med icke-linjär musik i ett verktyg som är anpassat efter linjär musik. Han påtalar också att detta fenomen är någonting han finner väldigt intressant. Han fortsätter med att beskriva varför han tror att en "DAW" är utformad med en tidslinje som grund. Han menar att den härstammar från hur noter skrivna på papper också är efter en tidslinje, vilket är en aspekt av hur musik har skapats. Då ställer han frågan vad som händer ifall de linjära verktygen man arbetar med försöker möta den icke-linjära aspekten genom att anpassa verktygen till att bli mer anpassade efter icke-linjära miljöer.

Därefter frågades vilka mellanprogram som Jacob har erfarenhet av. Där svarade Jacob med att han är mest bekväm med FMOD och Wwise. Därefter säger han att det börjat dyka upp många olika mellanprogram som är bra på olika saker.

Efter detta diskuterades om "≡ AMAPP" förändrat hur Jacob komponerar och i så fall hur. Det diskuterades även om det fanns någon likhet mellan hur ett mellanprogram skulle potentiellt format komponerandet. Där beskriver Jacob att mellanprogram kan abstrahera vad ett "sound-event" är. Han förklarar att om man förstår sitt mellanprogram så kan det ge unika och specifika idéer till kreativa lösningar. Han beskriver även hur FMOD och Wwise skiljer sig i sättet de abstraherar "sound-events". FMOD är mer "DAW"-liknande där man arbetar på en tidslinje, medan i Wwise arbetar man i en form av "trädhierarki". Detta menar Jacob skapar en annan typ av förståelse för hur man kan ta fram musik där han åter pålyser hur ett gränssnitt kan göra stor skillnad i hur man tar sig an ett problem.

Fortsättningsvis beskriver Jacob hur "≡ AMAPP" gör att han slipper lägga mental energi på organiseringen av filer, vilket gör att han kan vara mer spontan i de idéerna han vill testa. Någoting som utforskades av Jacob och tillsammans med några andra var hur "≡ AMAPP" kan användas i kombination med spelmotor och mellanprogram i en studiomiljö. Han beskriver att normalt sätt när man spelar in i en studiomiljö gör man ett antal tagningar med en uppskattning på hur det kommer att låta och fungera implementerat i spelet i åtanke samtidigt som man spelar in. Det finns en risk för att man kan behöva göra efterarbete en lång tid efter inspelningen för att det först då uppdagats att det inte blev bra, efter implementeringen är gjord. Detta ville Jacob testa, hur man kan implementera musiken direkt efter inspelningstagningarna i en studiomiljö för att omedelbart kunna bedöma hur musiken fungerar i sitt tilltänkta sammanhang (se figur 12). Detta möjliggjorde att musikerna i studion kunde improvisera fram idéer till hur musiken skulle låta och bete sig i en studiomiljö samtidigt som spelet kördes på en skärm i studion. Då hade de projektet som referens snarare en "vision" av musiken som referens.

Vidare jämför han med filmmusik där han säger det är vanligt att man har filmen på plats med tilltänkta tillfällen som musiken ska vara med när man spelar in musiken. "Det kan komma kreativa lösningar från det. Man kan ta beslut som visar sig vara väldigt viktiga i sådana lägen", säger Jacob.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Hur arbetade du/ni med att få in musiken i spelet?	Förarbete och noga förberedelse. Improvisation i en studiomiljö. Arbetsflöde med mjukvara som möjliggör snabba iterationer i en studiomiljö. Komponerar till spelet och implementerar det på plats.	Snabb iteration Förberedelse Improvisation
Hur får verktygen dig att tänka på komponerandet av dataspelsmusik?	Abstraherar problem olika med olika verktyg. FMOD är "DAW" likt, använder tidslinje.	Konceptualisering

**Figur 12:** Tematisk tolkning av Jacobs svar.

Vidare i intervjun diskuterades frågan hur resultatet påverkats av att kunna iterera mycket snabbare när verktygen användes i projektet som beskrivits. Där förklarar han att den största tidsboven i ett projekt är den iterativa processen. Han säger att om man inte kan implementera musiken själv så är den iterativa processen alldeles för lång. Där beskriver han att efter musiken har producerats ska den renderas ut på ett bra sätt. Om kompositören inte implementerar själv måste materialet överlämnas till en programmerare. Förhoppningsvis förstår programmeraren hur det är tänkt att bli implementerat med relevanta musiksystem för att sedan implementera det i spelet. Det är först då det slutgiltiga resultatet kan utvärderas. Då kanske det visar sig att musiken inte blivit "bra" i spelet och andra lösningar måste då hittas. Jacob fortsätter: "[...] ju kortare iterativ process man har desto snabbare kan det tas kreativa beslut". Han understryker även att en kortare iterativ process från början till slut är viktig för att kunna vara effektiv och kreativ.

Vidare förklarar Jacob att i hans erfarenhet är det första iterationen som tar längst tid. Det är många system som ska samverka och lösningar som ska byggas innan musiksystemet går att lyssna på i spelet. Efter att musiken, struktureringen av events i sitt mellanprogram och

implementationslösningarna gjorts och finns i spelmotorn så att spelet kan byggas, först då kan bedömning av hur musiken blivit i spelet göras. Förutsatt att det inte görs omfattande förändringar i systemen tar nästkommande iterationer inte lika lång tid att göra, fortsätter Jacob. Detta förberedes i projektet på förhand där systemen redan var kopplade och strukturerade så att implementeringen kunde ske snabbt i inspelningsstudion.

Genom att fortsatt vara närvarande i den kreativa processen av musikskapande, utan att behöva ägna sig åt andra problem, underlättas utvärderingen av det kreativa resultatet, menar Jacob. Då kan utvärderingen göras redan när musiken komponeras för att undvika eventuella överraskningar på saker som uppdagas först efter implementeringen är gjord (se figur 13). Är implementeringen långt efter inspelningen kanske man behöver spela in saker på nytt eller kompromissa med sin konstnärliga vision på andra sätt.

En följdfråga som ställdes var om det kan sägas ju snabbare man når utvärderingsfasen desto bättre, vilket Jacob svarade på med att han inte lägger en värdering i om det är bra eller dåligt utifrån ett konstnärligt perspektiv. Däremot utifrån ett pengar- och resurseffektivt perspektiv är det definitivt att föredra. Han understryker även att:

” [...] konst är konst, saker ska få kunna ta lång tid att göra. Man vrider och vänder på någonting, man måste hitta de här tekniska problemen man står inför. Det kan vara mycket viktigare än att det går snabbt att göra någonting.” (Appendix C)

Han tog upp generativ artificiell intelligens (hädanefter AI) som exempel på att material kan produceras snabbt men ställde också frågan ifall materialet som producerats av AI:n är vad som är intressant för projektet.

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vilken roll spelar implementationsverktyg när man arbetar iterativt?	Ju kortare iterationsprocess, desto snabbare kan man ta kreativa beslut. Första iterationen mest omfattande och krävande. Närvarande i det ett kreativt sammanhang utan längre avbrott. Överlämning av implementering till andra kan innebära längre tid till utvärdering och otillfredsställande resultat.	Iterationsprocessens tid Ökad effektivitet Ökad kreativitet Personligen implementera. Kommunikation

**Figur 13:** Tematisk tolkning av Jacobs svar.

Vidare i intervjun diskuterades det hur musiken effektivt kan testas för att säkerställa att den fungerar som den är tilltänkt, både med och utan mellanprogram och hur dessa skiljer sig åt. Då berättade Jacob att när saker skapas utifrån ett koncept, när det inte finns mycket av ett spel i början, har det sällan blivit rätt från början. Det är först när grafik och en spelmiljö har kommit in i bilden som mer utförliga skraddarsydda lösningar och musikstycken kan göras. Utöver detta förklarade Jacob att han försöker realisera implementationen, den adaptiva lösningen i Wwise eller FMOD, och sedan lyssna på hur exempelvis övergångarna låter mellan två tillfällen. Då kanske mängden trumfill behöver ändras på, var de spelas men också hur kadensen i musiken är strukturerad. Det hjälper honom att förstå hur musiken och systemet ska fungera i slutändan. Han tillägger även att med hjälp av ”≡ AMAPP” kan det testas hur

musiken kommer låta enkelt, direkt i sin "DAW".

Därefter diskuterades det i vilket skede kompositören börjar tänka på implementationen av musiken till ett dataspel. Där säger Jacob att "det inte bara är olika från projekt till projekt utan även från kontext till kontext" (Appendix C). Vidare förklarar Jacob att han är väldigt intresserad av speldesign och har speldesignen i åtanke när han komponerar. Han ställer sig många frågor om exempelvis vilken kontext musiken kommer vara i, vad det är för typ av spel och vad spelarens mål är. Jacob menar också att han tror de flesta som jobbar med adaptiv musik har liknande tankar i huvudet när de komponerar och ger exempel på musik som ändras när spelaren smyger och blir jagad. När Jacob undervisar om adaptiv musik ger han exempel på hur musikens relation till speldesign kan illustreras genom tumbrottnings. Spelet inleds med en ramsa som sedan övergår till en strid. Efter detta finns det en vinnare och en förlorare. Genom dessa enkla regler skapas ett intressant spel och om musik ska göras till den kontexten, vad vill kompositören då säga med musiken, menar Jacob. Vilka känslor vill hen illustrera, hur lång tid är det från början till slut, när spelet är slut, hur låter det då beroende på vem som vann eller förlorade? Detta är frågor som Jacob ställde till denna övning. Han menar att det finns vissa viktiga moment i speldesignen som kompositören har i åtanke när hen komponerar.

För att skriva musik till ett stridsmoment i ett spel brukar han skriva ett musikstycke med olika instrument, delar och övergångar. Han har en ungefärlig uppfattning på hur det kommer låta när musiken börjar och slutar. Sedan delar han upp detta musikstycke i "intro, main loop, outro". Utifrån detta kan han sedan extrahera ut olika variationer av den musiken vilket blir ett större musiksysteem i slutändan. Han påtalar även att verktygen som används påverkar det slutgiltiga resultatet mycket (Se figur 14).

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Hur gör man för att effektivt testa hur musiken fungerar i ett dataspel?	Realisera implementationen och lyssnar på hur det blir i middleware-program. AMAPP möjliggör en mer träffsäker representation av slutresultatet i spelet.	Lyssna i lämplig mjukvara
I vilket skede börjar man tänka på kompositionen i termer som implementeringen?	Bundet till vilken typ av spel och situation i spelet. När man vet kontext kan man börja tänka på hur musiken ska fungera i spelet.	Kontextbundet Medveten konstant

**Figur 14:** Tematisk tolkning av Jacobs svar.

Nästa ämne som diskuterades var vilka utmaningar spelmusikkompositörer har. Där svarade Jacob med att om man har tur blir man inblandad i projektet tidigt i produktionsfasen. Detta gör att man kan förvalta och bygga sina idéer väldigt tidigt. Det gör att övriga i projektet får en ökad förståelse för musiksysteмен och förutsättningarna för att skapa musiken förbättras. Det underlättar att jobba konstnärligt med vad adaptiv musik är, säger Jacob. Han fortsätter och berättar att det inte är ovanligt att kompositörer kommer in i produktionsprocessen väldigt sent vilket gör att det inte finns lika mycket utrymme att bygga och förvalta idéer. Han säger även att komma in tidigt i produktionsprocessen bygger upp förväntningar hos utvecklingsteamet då man kan ta sig an fler och fler idéer och musikstycken till olika bossar exempelvis. Men ju fler saker man tar på sig och större projektet blir desto högre blir

förväntningen av att det blir bra i slutändan. Han beskriver även att med erfarenhet har han blivit bättre på att prioritera de viktigaste sakerna först och inte lägga tid på komplicerade lösningar som är oviktiga. Längre fram i intervjun nämnde Jacob även att bygga upp komplexa musiksystem som gör musiken spännande att lyssna på kan innebära ett jättejobb att förverkliga. Det är vad någon som skriver adaptiv musik blir expert på bland annat, säger Jacob.

Utöver detta beskriver Jacob att det generellt sätt finns en oförståelse i spelbranschen kring hur mycket arbete som går in i att skapa musik till spel. Det är många som inte bryr sig särskilt mycket om den konstform som man själv håller på med. Jacob beskriver med hjälp av ett exempel: någon lyssnar på tre minuter musik och tänker att det borde gå väldigt snabbt att producera, men de tre minuterna kanske har arbetats fram över ett år. Det gör det komplicerat att navigera runt den oförståelsen, tycker Jacob. Utöver detta säger Jacob att kunna kommunicera vad som kan göras med förutsättningarna ett projekt har är värdefullt.

Vidare i intervjun diskuterades vilka tekniker Jacob brukar använda sig av när han komponerar där vertical layering förekom bland dessa. Utöver detta uppdagades det även någonting Jacob uppskattar med användandet av mellanprogram. Enkla adaptiva system kan med hjälp av ett mellanprogram lättare utvecklas (se figur 15). Han säger: "Ett middleware hjälper dig att bygga nya delar till musik eller bygga vidare på lösningar väldigt väl".

Fråga:	Första ordningens koncept:	Andra ordningens tema:
Vilka utmaningar har man som kompositör när man komponerar musik för dataspel?	Fördelaktigt att vara involverad tidigt i produktionsfasen, möjliggör förvaltning av idéer. Ökar förväntningarna på kompositören Oförståelse för hur mycket arbete som dataspelsmusik kräver. Prioritera viktigaste sakerna först Kommunicera vad man som kompositör kan åstadkomma med resurserna som finns till hands.	Kommunikation Förväntningar Tidigt delaktig Omfång och prioritering
Vilka tekniker använder du vid komponerandet av dataspelsmusik?	Vertical layering Middlewares är bra på att bygga vidare på redan existerande lösningar.	Extrapolera tekniklösningar

**Figur 15:** Tematisk tolkning av Jacobs svar.

## 4.4 Analys av intervjusvaren

Någonting som uttalades av samtliga respondenter var vikten av kommunikation mellan kompositör och utvecklare. För att öka chansen för att den kreativa visionen realiserar behöver där finnas en gemensam förståelse mellan kompositör och utvecklare. Att kommunicera intentioner tar tid ifrån att komponera musik. Behovet av att förklara hur musiken är tänkt att fungera ökar när ett spelprojekt inte använder en mellanmjukvara. Behovet ökar ytterligare om man som kompositör inte är involverad i implementationen av musiken. Risken för merarbete på grund av missförstånd mellan kompositör och utvecklare ökar också.

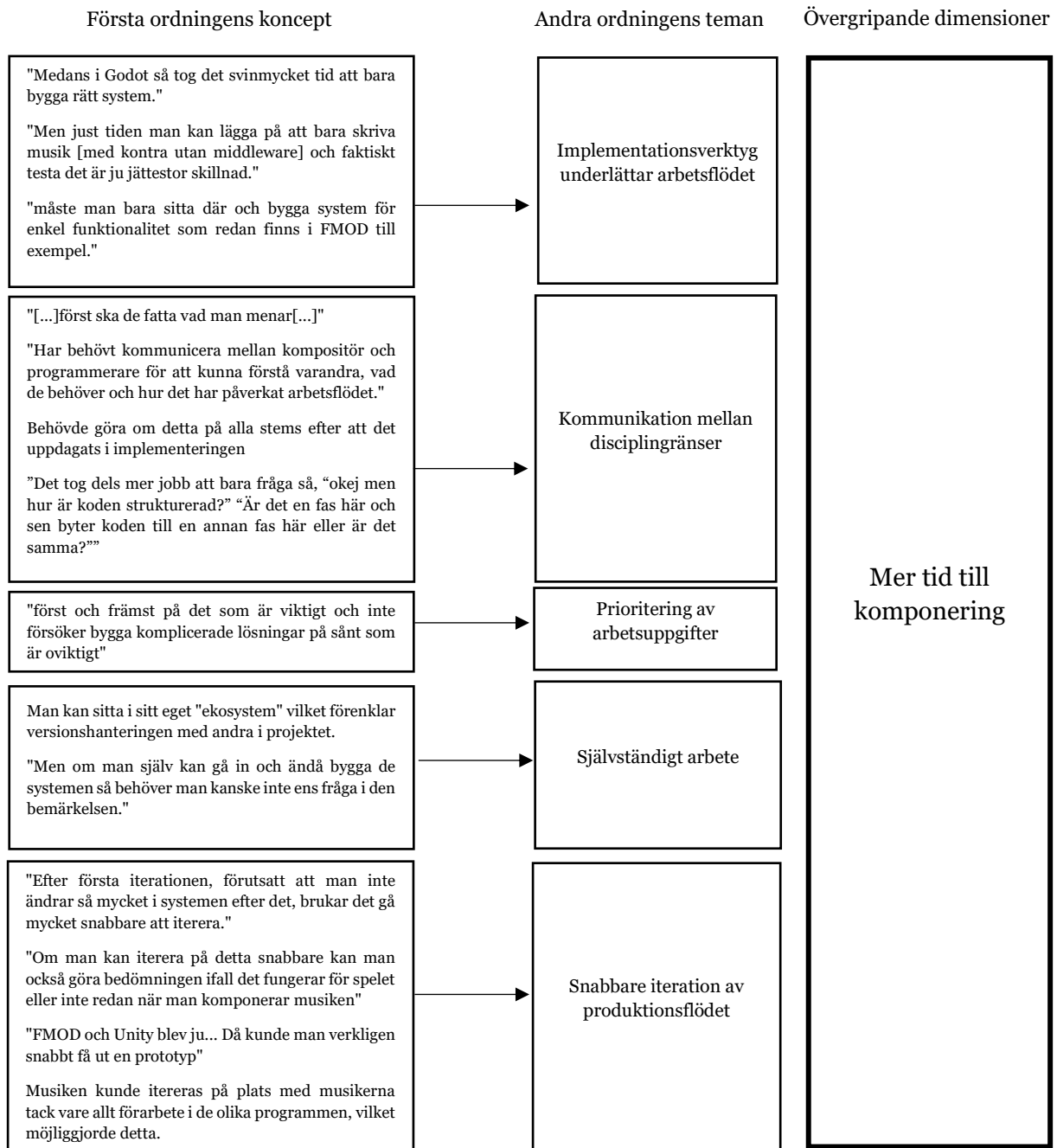
Vidare framgår det att implementationsverktyg underlättar för arbetsprocessen genom att snabbare kunna iterera på musiken när den är i sitt tilltänkta sammanhang, alltså i spelet. Den omfattande arbetsprocessen av komponering och implementering, vilka i sig innefattar flera program och processer, påverkar tiden för utvärdering av den kreativa slutprodukten. I intervjuerna framgår det alltså att: där iterationsmöjligheterna ökar gör också realiseringen av den kreativa visionen och kvalitén av slutprodukten.

Filhantering är också någonting som underlättas vid användandet av implementationsverktyg enligt intervjuerna, vilket ytterligare bidrog till att mer tid kunde ägnas åt att skapa musik. Det påtalades av respondenterna att mellanmjukvaror ökade möjligheten till självständigt arbete och underlättade även versionshantering i spelprojektet. Det beskrevs av en respondent som att kunna jobba i ett eget "ekosystem" fristående spelprojektet när man arbetar med implementeringen personligen. I samtliga intervjuer förekom det att en ökad delaktighet i implementeringen från kompositören innebär ett större inflytande och kontroll över den kreativa slutprodukten.

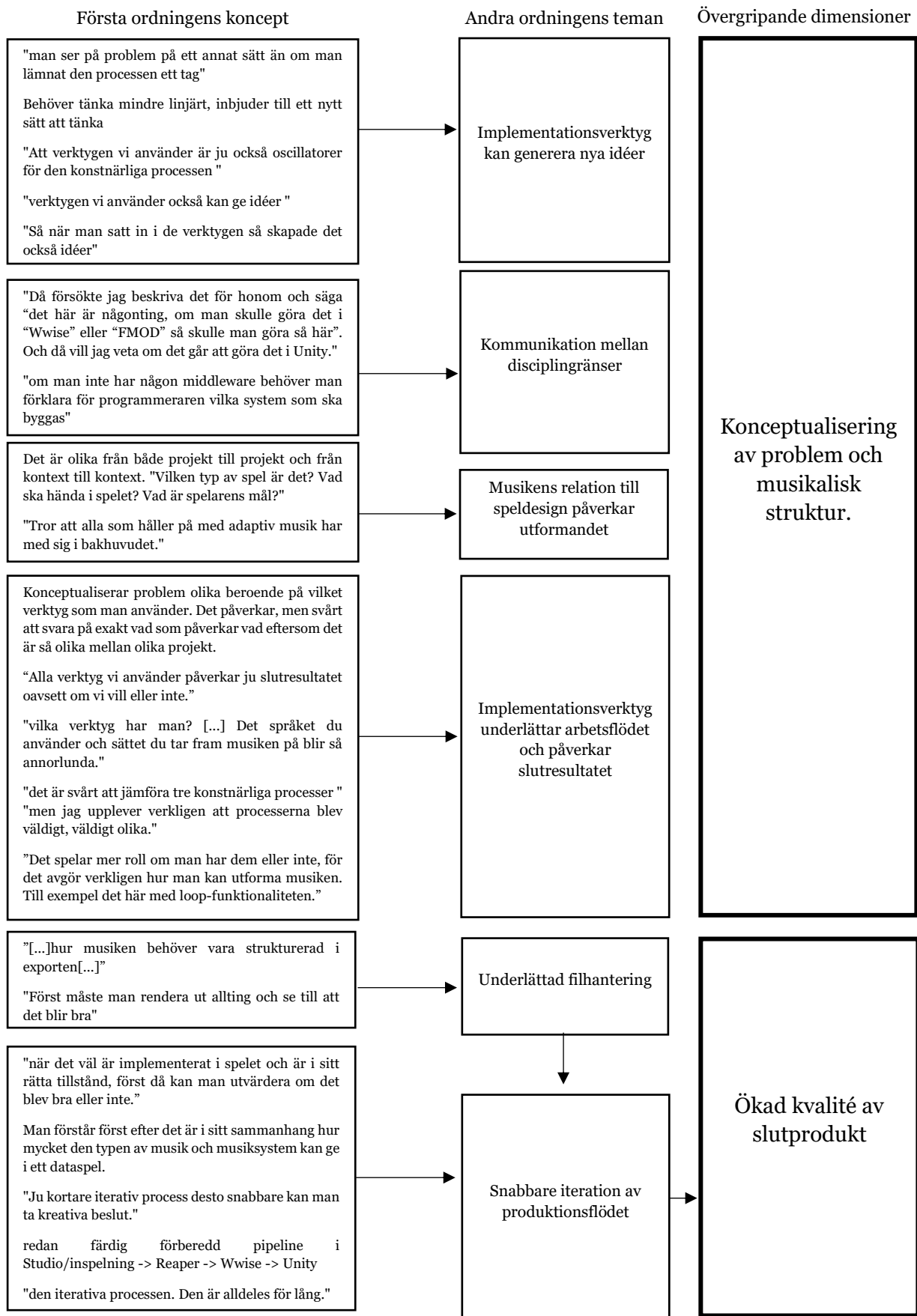
Vertical layering är en teknik som kom på tal i samtliga intervjuer och som har använts av respektive respondent på olika sätt i projekten de arbetat på. Detta var inte oväntat eftersom det är en av de vanligaste teknikerna som används för adaptiv musik för dataspel (Mc Glynn 2023). Samtliga respondenter beskrev även musikens musikaliska struktur likt hur Medina-Gray (2016) beskriver modulär musik i dataspel. Spelets tillstånd styr vilken ordning som olika musikdelar spelas i. Det beskrevs i intervjun med Gustaf att musiken i dataspel kan liknas vid olika pusselbitar som kan läggas i vilken ordning som helst beroende på vad som händer i spelet. Hur detta hanteras i spelet kan underlättas med hjälp av implementationsverktyg enligt samtliga respondenter.

Musiken kan även ha en narrativ funktion genom att gestalta karaktärers sinnestillstånd (Ertan 2025) likt det samtliga respondenter beskrivit med att olika karaktärer gestaltats genom olika instrument. Utöver detta kan musiken återspegla viktiga faser i spelet, exempelvis stridsfas och återhämtningsfas, likt tumbrottningsexemplet Jacob nämnde i intervjun. Detta påminner om vad Ertan (2025) menar med att musik kan sy ihop narrativ och spelmekaniska funktioner.

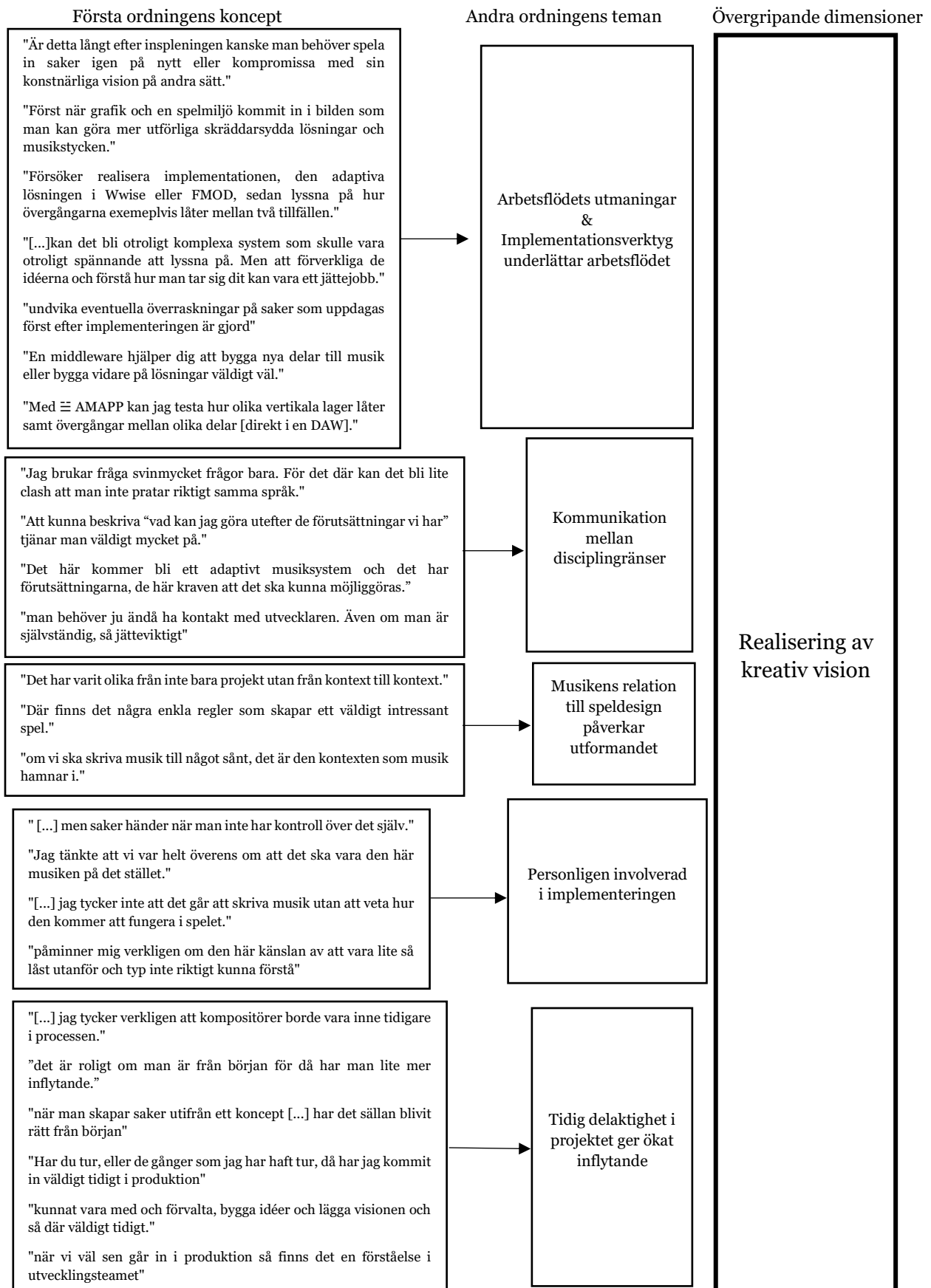
Intervjuerna har tolkats och kategoriserats i följande fyra övergripande dimensioner: "Konceptualisering av problem och musikalisk struktur", "Mer tid till komponering", "Realisering av kreativ vision" och "Ökad kvalitét av slutprodukt" (Se figur 16 - 18). Av samtliga respondenter förekom det olika exempel på hur implementationsverktyg har genom dess inbyggda funktionalitet inspirerats till nya idéer för musikens utformning i ett spelprojekt. Med hjälp av den inbyggda funktionaliteten i mellanmjukvaror har problem abstraherats och lösts. Det påtalades även hos samtliga att det fanns skillnader i arbetsprocesserna när en mellanmjukvara används kontra inte.



**Figur 16:** "Gioia-inspirerat" tolkningsdiagram, Mer tid till komponering som övergripande dimension.



**Figur 17:** "Gioia-inspirerat" tolkningsdiagram, Konceptualisering av problem och musikalisk struktur samt ökad kvalitet av slutprodukt som övergripande dimensioner.



**Figur 18:** "Gioia-inspirerat" tolkningsdiagram, Realisering av kreativ vision som övergripande dimensioner.

## 5 Sammanfattning och diskussion

### 5.1 Sammanfattning

Ludomusikologi är ett ungt fält där frågor om hur det kan studeras ställs (Fernández-Cortés & Cook 2021; Kamp, Summers, & Sweeney 2016). Detta kan vara en förklaring till den begränsade mängden forskning som gjorts på ämnet. Fältets befintliga forskning har sällan varit utifrån ett produktionsperspektiv.

Studien, som har ägt rum under loppet av åtta veckor, bestod av tre faser: förberedelsefasen, intervjufasen, och analysfasen. Under den första faser planerades studien, intervjumaterialet förbereddes och respondenter kontaktades. Den andra faser hölls intervjuerna vilka därefter bearbetades. I tredje faser analyserades och diskuterades den insamlade datan.

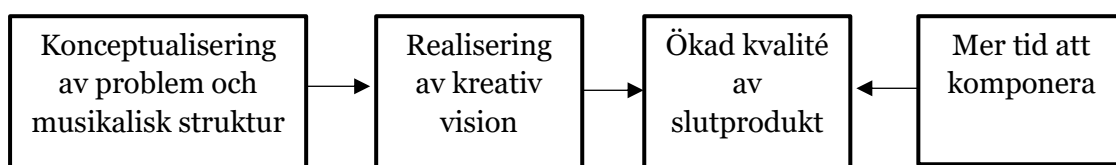
Studiens frågeställning är "Hur påverkas komponerandet för dataspelsmusik av implementationsverktygen för dataspelsmusik?". För att undersöka frågeställningen användes en kvalitativ forskningsansats med subjektivt urval. Deltagare valdes utifrån att de är etablerade dataspelsmusiker i dataspelsindustrin. I slutändan fick studien tre deltagare; Tobias Kalliokulju, Gustaf Blix och Jacob Westberg.

Intervjuerna transkriberades och bearbetades varefter de kodades och tolkades med hjälp av "Gioia-inspirerade" tolkningsdiagram. Samtliga respondenter tycker implementationsverktyg såsom *FMOD Studio* (Firelight Technologies 2013) och *Wwise* (Audiokinetic 2006) förändrar arbetsflödet och eventuellt kan inspirera till nya idéer, trots att den kausaliteten är svår att bevisa.

Respondenterna lyfte även att kommunikation mellan musiker och programmerare är viktigt, att det behövs kommuniceras mer när mellanprogram inte användes och än mer när musikern inte är involverad i implementationen. De tyckte även att iterationsprocessen blir lättare och snabbare med implementationsverktyg.

### 5.2 Diskussion

De övergripande dimensionerna som tolkats fram från datan relaterar till varandra. Exempelvis kan "mer tid att komponera" innebära att kompositören kommer närmare sin "realisering av kreativ vision" som i sin tur kan "öka kvalitén av slutprodukten". "Konceptualisering av problem och musikalisk struktur" kan också i sin tur påverka till vilken grad kompositören kan komma närmare sin "realisering av kreativ vision" vilket sedan kan leda till "ökad kvalitét av slutprodukten" (se figur 19).



**Figur 19:** De olika övergripande dimensionerna och deras relation till varandra.

Viktigt att ha i åtanke är att samtliga tolkningsdiagram är forskarnas *tolkningar* vilket lägger ansvar på forskarna att de behandlat och tolkat datan på ett sätt som korrekt återspeglar

respondenternas svar. Även om en opartisk tolkning av datan aldrig kan garanteras ska det eftersträvas i den mån det går. En styrka med att använda Gioia-inspirerade tolkningsdiagram är att det ökar transparensen och spårbarheten till var tolkningarna kommer ifrån, vilket inte eliminerar eventuell bias från forskarna men gör denna transparent.

En svaghet med intervjumetoden och frågornas utformning är att datan är subjektiv och svårtolkad. Detta har resulterat i att mycket tid har ägnats åt att tolka intervjuvaren vilket har begränsat mängden respondenter. Detta har i sin tur begränsat snöbollsurvalets önskade effekt, dessutom var det för ont om tid att hitta nya respondenter under intervjufasen. Därför var det viktigt att det subjektiva urvalet täckte upp den potentiella bristen i respondentantalet som ett misslyckat snöbollsurval kunde innebära.

En intressant punkt som uppdagats i intervjuvaren är frågan om hur tidigt kompositören bör komma in i ett spelprojekt. Kompositörerna påtalar att det är positivt och i vissa fall fördelaktigt att inkludera tidigt i arbetsprocessen, samtidigt som de också säger hur svårt det är att träffa rätt med musiken i ett tidigt skede. Detta kan uppfattas som motsägelsefullt men en mer nyanserad tolkning av situationen vore följande. Å ena sidan, att bli involverad tidigt i ett spelprojekt kan innebära en utökad frihet att utforska, motivera och förvalta idéer vilket också kan underlätta för att hitta en gemensam vision av musikens syfte för samtliga i projektet. Samtidigt påtalades det att det är svårt att träffa rätt när många delar av projektet inte är färdigställda vilket innebär att musiken behöver omarbetas. Å andra sidan, blir kompositören involverad senare i projektets gång kan fler färdigställda komponenter bidra till en ökad träffsäkerhet för musikens syfte, men på bekostnad av kreativ frihet. Detta kan då leda till en flaskhals i projekten där musikern väntar på att kunna testa sin musik, vilket också förlänger iterationsprocessen och påverkar projektet i sin helhet. Detta kan vara anledningen till att musiker som norm är involverade senare i projekten. Resultatet är kanske att förväntningarna är mer realistiska på musikern men det kanske uppstår ett annat problem i arbetsprocessen som behöver åtgärdas.

Likt det beskrivet i bakgrundskapitlet var ett av intervjuvaren att musiken i spelet är väldigt nära kopplat till speldesign. I fallet av looping är spelets design det som skapar begränsningarna, men som Jacob beskrev i intervjun är musiken i spel helt och hållet kopplat till speldesign, vilket i sin tur ger en insikt i hur spelmusik fungerar. Till exempel är Jacobs process för att skriva musik intimt kopplat till hur spelets logik är utformad. Hans exempel med tumbrottnings visar tydligt hur en kompositör tänker när processen börjar. Några frågor som kan ställas är då: "Vilka delar behöver musiken ha?", "Hur kopplar delarna till varandra?", "Vilken roll har musiken i sammanhanget?" och "Vilka sätt kan och borde musiken förändras på?". Spelets design och logik förändrar svaret på alla de här frågorna. Detta kan liknas med det Medina-Gray (2016) beskriver i sin artikel om modularitet i dataspelsmusik. Utöver detta kan Ertans (2025) artikel, som berör hur dynamisk musik förändras beroende på speltillståndet, liknas med det Jacob beskriver i intervjun.

Från intervjuerna kan tolkningen göras att en minskad delaktighet i implementationen från kompositörens håll ökar risken för att musiken inte kommer fungera som den är tilltänkt. Implementeringen av musiken är en viktig del av den kreativa visionen för musiken. Det är svårt att utvärdera olika konstnärliga processer, men det är tydligt att det finns en uttalad uppfattning hos deltagarna att implementationsverktygen påverkat arbetsprocessen. Sättet ett kompositionsproblem bemöts påverkas av hur olika program är utformade. Även om frågan angränsar mot gränssnittens design påverkar detta fortfarande valen som utvecklaren gör och kan därför vara värt att analysera närmre i framtida studier. Värt att ha i åtanke är att

respondenterna har arbetat olika mycket med olika program vilket också kan påverka hur de använder implementationsverktygen. Erfarenhet och vana vid mjukvaran kan göra att sättet man använder dem förändras och utvecklas på utmed användandet.

Musik som upprepar sig kan uppfattas som störande för spelaren (Lopez Duarte 2020). Det finns tekniker som med hjälp av algoritmiskt generad musik kan göra den mindre repetitiv, men detta kan vara på bekostnad av kontrollen över det musikaliska uttrycket och den musikaliska kvalitén (Hutchings & McCormack 2020). Även om musiken kan bli mer varierad av denna typ av musik innebär inte det nödvändigtvis att musiken får den tilltänkta effekten (Lopez Duarte 2020; Hutchings & McCormack 2020).

Musikimplementationen idag ser väldigt annorlunda ut jämfört med ett par decennier sedan. Begränsningar i hårdvara finns självklart fortfarande idag men de är inte alls lika extrema (Burke 2024). I dag begränsas inte musiken i dataspel av hårdvaran utan det är snarare hur idéerna förvaltas och formas i arbetsprocessen samt hur de utförs som är mer komplexa. Till exempel om kommunikationsproblem uppstår under processen blir det svårare att komma närmare den kreativa visionen för både musiken och spelet eftersom tid, energi och resurser går till det istället.

Hur mycket av våra upptäckter kan det konstateras är på grund av *implementationsverktyg* och vad är på grund av *dataspelets dynamiska natur*? Detta går att likna med frågan ”vad kom först, hönan eller ägget?” vilket illustrerar komplexiteten i frågeställningen. Även om det är svårt att dra en generell slutsats kring vilken roll implementationsverktygen spelar i formandet av kompositionsprocessen, indikerar respondenternas svar att fler idéer kan utforskas utmed ett projektets gång och därmed öka träffsäkerheten på vad kompositören ämnar uppnå med musiken genom användandet av verktygen.

## 5.2.1 Samhälleliga och etiska aspekter

Spelutvecklingsbranchen och därmed spelmusiksbranchen är mansdominerad. Detta reflekteras även i studien där exklusivt män intervjuats. Dock är det också svårt att få en mångfaldsrepresentation när mängden deltagare som intervjuats i studien är så få. I vidare studier hade detta kunnat åtgärdas genom mer mångfaldsrepresentation och fler respondenter.

Att använda en mellanmjukvara kostar, i de flesta fall, pengar och mindre spelföretag har inte alltid möjligheten att spendera tid och resurser på användandet av dem. Men denna kostnad kan i sin tur betala av sig själv i form av mer effektivt arbete och en bättre slutprodukt. Denna undersökning kan bidra till en mer nyanserad bild av vinsterna och förlusterna med användandet av olika typer av implementationsverktyg utifrån ett konstnärligt- och effektivitetsperspektiv.

Även om effektivitet är en viktig del av ett komplicerat spelprojekt är det inte allt. Ofta förknippat med effektivitet är hur lång tid någonting tar att göra. Likt det som Jacob beskriver i intervjun ” [...] konst är konst, saker ska få kunna ta lång tid att göra.” (Appendix C) måste den konstnärliga processen få ha sin gång. Detta innebär att tiden det tar att producera ett konstnärligt verk inte alltid går att värdera på samma sätt mellan konstnärliga verk. Som Jacob nämnde i intervjun kan AI producera ett ”konstnärligt” verk mycket snabbare än vad en människa hade klarat av, men det den producerar kanske inte är intressant för sammanhanget. Generativ musik kommer även med upphovsrättsliga problem med vem som kan anses vara innehållets skapare (Lopez Duarte 2020). Det är inte alltid lätt att utreda hur den generativa

musiken skapats och vilka delar AI:n har tränats på för att kunna generera musiken.

### 5.3 Framtida arbete

För att få en större överblick över dataspelsbranschen hade fler respondenter kunnat inkluderas vilket också hade ökat generaliserbarheten. Med ett ökat antal respondenter kan även större mångfald uppnås.

Ifrån den datan som har samlats in är det tydligt vad åsikterna för just dessa tre musiker är. Fler respondenter hade kunnat etablera ett större mönster i spelmusikbranschen som helhet. En annan vinkel hade kunnat vara att samla in data från de andra delarna inom spelutveckling, till exempel grafik, programmering, eller den angränsande delen, ljud. De nya perspektiven hade kunnat göra det lättare att navigera ludomusikologi.

AI kan påverka implementationsprocessen i framtiden. Det är fortfarande osäkert i vilken mån AI kommer förändra dataspelsmusik och dess implementation men risken finns att datan som samlats in här inte kommer vara aktuell i framtiden. Generativ musik kommer högst sannolikt att utvecklas och bli bättre med tiden och kanske förändra sättet kompositörer arbetar med musik för dataspel. Dessa implikationer kan vara värda att utforska vidare på.

En annan fråga som uppdragats av detta arbete är ifall personer som inte har använt mellanprogram, som bara har förlitat sig på de inbyggda implementationsverktygen i spelmotorer, har samma uppfattning som respondenterna i denna studie. Har kompositörer med erfarenhet av mellanprogram annan uppfattning om hur dataspelsmusik konceptualiseras jämfört med kompositörer utan erfarenhet av mellanprogram? Detta belyser återigen möjligheterna för mer forskning inom området.

I intervjuerna diskuterades hur olika mellanprogram är utformade olika, vilket kan påverka hur man tar sig an problem. Exempelvis hur *FMOD Studio* (Firelight Technologies 2013) använder ett tidslinjeformat gränssnitt jämfört med hur *Wwise* (Audiokinetic 2006) använder ett trädhierarkiskt gränssnitt påverkar konceptualiseringen av musiken i dataspel. Därför vore det intressant att undersöka hur ett programs utseende och utformande kan påverka kompositionsprocessen.

En annan aspekt som uppdragats i intervjuerna var när man som kompositör ska bli involverad i ett spelprojekt. Det framgick att samtliga respondenter gärna är tidigt delaktiga i processen men att musiken inte träffar rätt från början. När är då den mest fördelaktiga tidpunkten för kompositör och spelutvecklare att inkludera kompositören i processen? Detta kan utforskas vidare.

## Referenslista

- Audiokinetic. (2006). *Wwise* (Första utgåva) [Mjukvara]. Montreal: Audiokinetic Inc. <https://www.audiokinetic.com/en/wwise/overview/>
- Bhagal, G. K. (2024). Claude Debussy's Piano Music and the Evocation of Atmosphere and Humor in Video Games. I Gibbons, W. & Grimshaw-Aagaard, M. (red.) *The Oxford handbook of video game music and sound*. New York: Oxford University Press. s. 556-575.
- Burke, R. K. (2024). Hard Limitations and Soft Possibilities: A "Synthetic" History of Early Video Game Sound Technology. I Gibbons, W. & Grimshaw-Aagaard, M. (red.) *The Oxford handbook of video game music and sound*. New York: Oxford University Press. s. 159-185.
- Denscombe, M. (2018). *Forskningshandboken – för småskaliga forskningsprojekt inom samhällsvetenskaperna*. 4. uppl. Studentlitteratur AB, Lund.
- Engström, H. (2020). *Game Development Research*. Skövde: Högskolan i Skövde. <https://his.diva-portal.org/smash/get/diva2:1501250/FULLTEXT01.pdf>
- Ejvegård, R (2009). *Vetenskaplig metod*. 4. uppl. Studentlitteratur AB, Lund.
- Epic Games Tools. (1991). *Miles Sound System* (Första utgåva) [Mjukvara].
- Ertan, E. (2025). Applying Video Game Satisfaction Elements to Cinematic Audio-Visuals: A Cross-Disciplinary Analysis through Ludomusicology, Diegesis and Flow Theory. *İletişim Kuram Ve Araştırma Dergisi*. 71, s. 23-44. <https://doi.org/10.47998/ikad.1669165>
- Fernández-Cortés, J. P. (2021). Ludomusicology: Normalizing the Study of Video Game Music, övers. Cook, K. M. *Journal of Sound and Music in Games*. 2(4). s. 13–35. doi:10.1525/jsmg.2021.2.4.13.
- Firelight Technologies. (2013). *FMOD Studio* (Första utgåva) [Mjukvara]. Melbourne: Firelight Technologies Pty Ltd. <https://www.fmod.com/>
- Firelight Technologies. (2026). *FMOD Studio User Manual 2.03.12* [Manual]. Melbourne: Firelight Technologies Pty Ltd. <https://www.fmod.com/docs/2.03/studio/glossary.html>
- Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2013). Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. *Organizational Research Methods*. 16(1). 15-31.
- Golding, D. (2021). Finding Untitled Goose Game's Dynamic Music in the World of Silent Cinema. *Journal of Sound and Music in Games*. 2(1). s.1–16. doi:10.1525/jsmg.2021.2.1.1.
- Horowitz, S. & Looney, S. (2014). *The essential guide to game audio : the theory and practice of sound for games*. Burlington, MA: Focal Press.
- House House. (2019). *Untitled Goose Game* [Dataspel]. Panic Inc..
- Hutchings, P. E. & McCormack, J. (2020). Adaptive Music Composition for Games. *IEEE Transactions on Games, Games, IEEE Transactions on, IEEE Trans. Games*. 12(3). s. 270–280. doi:10.1109/TG.2019.2921979.
- Kamp, M., Summers, T. & Sweeney, M. (2016). Introduction. I Kamp, M. Summers, T. & Sweeney, M. (red.) *Ludomusicology : approaches to video game music*. Equinox Publishing. s. 1-7.

- Lopez Duarte, A. E. (2020). Algorithmic interactive music generation in videogames: A modular design for adaptive automatic music scoring. *SoundEffects - An Interdisciplinary Journal of Sound and Sound Experience*. 9(1). s. 38–59. <https://doi.org/10.7146/se.v9i1.118245>
- Marks, A. (2017). *Aaron Marks' Complete Guide to Game Audio*. 3. uppl. A K Peters/CRC Press. <https://www.oreilly.com/library/view/-/9781317636182/>
- Mc Glynn, J.D. (2023). The “Cinematic Promise” of Video Game Music: Stylistic Convergence, Current-Generation Remakes, and the Case of Final Fantasy VII. *Journal of Sound and Music in Games*. 4(4). s. 108–138. doi: <https://doi.org/10.1525/jsmg.2023.4.4.108>
- Medina-Gray, E. (2016). Modularity in Video Game Music. I Kamp, M. Summers, T. & Sweeney, M. (red.) *Ludomusicology : approaches to video game music*. Equinox Publishing. s. 53-72.
- Rad Games Tools. (2024). *The Miles Sound System*. <https://www.radgametools.com/miles.htm> [Hämtad: 2026-03-26]
- Raybould, D. (2011). *The Game Audio Tutorial: a practical guide to crafting and implementing sound and music for interactive games*. Amsterdam: Focal Press.
- Samsung Electronics (2026). *Samsung Voice Recorder* (version 21.5.81.43) [Mjukvara]. Android.
- Unity Software Inc. (2026a). Audio Overview. *Unity Manual* [Manual]. San Francisco: Unity Technologies. <https://docs.unity3d.com/6000.3/Documentation/Manual/AudioOverview.html>
- Unity Software Inc. (2026b). Audio Source component reference. *Unity Manual* [Manual]. San Francisco: Unity Technologies. <https://docs.unity3d.com/6000.3/Documentation/Manual/AudioSource-reference.html>
- Vetenskapsrådet. (2024). *God forskningssed*. <https://www.vr.se/analys/rapporter/vara-rapporter/2024-10-02-god-forskningssed-2024.html>
- Zoom Communications, Inc. (2026). *Zoom Workplace* (version 6.7.8 (32670)) [Mjukvara]. <https://zoom.us/client/6.7.8.32670/ZoomInstallerFull.exe?archType=x64>

## Appendix A – Tobias Kalliokulju Intervjuutdrag

Fråga	Svar
Berätta lite om dig själv och om din kompositionskarriär. (vilka projekt har du jobbat på? etc.)	<p>Tillbringat mycket tid i kulturskolan i sin uppväxt. Spelat olika instrument i sin uppväxt.</p> <p>Spelat dataspel från en tidig ålder.</p> <p>Studerat Jazz på folkhögskola samt kandidat och master på Högskola i musik och media. Hade Audio och spelmusik som inriktning på mastern, gick ut i våras. Yrkesverksam i några månader som spelmusikkompositör.</p> <p>Utifrån ett konstnärligt forskningsperspektiv har Tobias utforskat kompositionsprocessen i dataspel kopplat till implementering.</p> <p>Little wing deliveries var en del av fyra delstudier i Tobias masterarbete.</p> <p>Arbetat i olika spelmotorer och audio-middlewares för dataspel.</p> <p>Några spelprojekt som är på gång men inte släppts än.</p> <p>Undervisat i komposition för dataspel.</p>
Fråga om ett projekt mer i detalj. (Berätta om vilken musik som gjordes för det projektet.)	<p><u>Little Wing Deliveries</u></p> <p>Gjordes i Unity, använder FMOD i slutprodukten</p> <p><i>“Jag som skrev musiken och skötte liksom det musikaliska arbetet i studion. Och det var väldigt så, rent kreativt, så en stor referens till projektet generellt var “A Short Hike”</i></p> <p><i>“Jag försökte preppa så lite som möjligt, För grejen är att vi gick in i studion och ville typ inte bestämma så mycket innan.”</i></p>
Vad är musikens syfte i spelet?	<p><u>Little Wing Deliveries</u></p> <p>Gestalta olika karaktärers karaktärsdrag genom olika instrument.</p> <p>Få lite olika känslor på de olika öarna.</p>
Har musiken någon tilltänkt narrativ, spelmekansik funktion eller väcka en viss känsla? Hur arbetade du för att uppnå det?	<p><i>“när man lyckats hjälpa de här karaktärerna och kommer till huvudön igen, då läggs deras instrument till i loopen av du-har-klarat-ditt-uppdrag-musiken”</i></p> <p><b>[Svaret nedan sades i en annan fråga angående ett Game Jam spel]</b></p> <p><i>“[...] om det ska finnas ett historieberättarperspektiv på musiken och om man vill säga någonting om storyn med musiken, då ska man förstå vad det är man försöker berätta, så att man också kan ta beslut över att “okej, men då kanske vi ska återknyta med ett instrumentval här eller ta tillbaka temat” till och med.”</i></p> <p><i>“Det finns massa aspekter som påverkar och jag tycker man ska ha en ganska god diskussion om det.”</i></p>

<p>Hur arbetade ni/du med att få in musiken i spelet?</p>	<p>Jacob som skötte själva implementations-biten i det projektet</p> <p><i>"Det blir ofta när man går in i studio med spel så blir det så himla mycket att man måste förpreppa allt"</i></p> <p><i>"Men kan det finnas en process där man i studion kan liksom implementera och testa hur det funkar i spelet och kunna utvärdera i studion för att kunna ta konstnärliga beslut."</i></p> <p>Det var lite utforska lite av både Wwise och FMOD.</p> <p>[Game Jam spelet]</p> <p>Var inte inblandad direkt i motorn. Skapade musik till spelet men implementerade inte den själv.</p> <p><i>"Återigen det här game jammet, så var det fel musik på andra stället och det var bara så... Jag tänkte att vi var helt överens om att det ska vara den här musiken på det stället och den här musiken på det stället, men saker händer när man inte har kontroll över det själv."</i></p> <p>Övriga i Game Jam Projektet uppskattade musiksyttemen och sättet den var adaptiv som Tobias uppmuntrade till att göra.</p> <p>Man förstår först efter det är i sitt sammanhang hur mycket den typen av musik och musiksyttem kan ge i ett dataspel.</p>
<p>Har du arbetat med middlewares? Vilka då?</p>	<p>FMOD, Wwise, Metasounds (Unreal built in) [inte middleware], Godot (built in audio features) [inte middleware]</p>
<p>Varför har du använt dig av dem [middlewares] och vad är det du uppskattar med dem?</p>	<p><b>[Frågan ställdes inte uttryckligen i intervjun]</b></p> <p><i>"FMOD och Unity blev ju... Då kunde man verkligen snabbt få ut en prototyp"</i></p> <p><i>"Medans i Godot så tog det svinmycket tid att bara bygga rätt system."</i></p> <p><i>"måste man bara sitta där och bygga system för enkel funktionalitet som redan finns i FMOD till exempel."</i></p> <p>Man kan sitta i sitt eget "ekosystem" vilket förenklar versionshanteringen med andra i projektet.</p> <p><i>"Men just tiden man kan lägga på att bara skriva musik och faktiskt testa det är ju jättestor skillnad." [Med kontra utan Middleware].</i></p>
<p>Hur får verktygen dig att tänka på komponerandet av musiken?</p>	<p><i>"Men i mitt masterarbete så utvecklar vi ett spel, musik i Unreal, med MetaSounds, ett med Unity och FMOD och ett med Godot med built in Godot audio features. Och det blev ju väldigt olika processer."</i></p> <p><i>"det är svårt att jämföra tre konstnärliga processer " "men jag upplever verkligen att processerna blev väldigt, väldigt olika."</i></p> <p><i>"FMOD och Unity blev ju... Då kunde man verkligen snabbt få ut en prototyp"</i></p> <p><i>"man får lägga en jävla massa tid på att bygga system som</i></p>

	<p><i>egentligen redan finns i ett program om man bara hade kunnat använda det programmet."</i></p> <p><i>"Så när man satt in i de verktygen så skapade det också idéer"</i></p> <p><i>"Att verktygen vi använder är ju också oscillatorer för den konstnärliga processen "</i></p> <p><i>"verktygen vi använder också kan ge idéer "</i></p>
<p>Hur mycket tänkte du på hur musiken interagerar med spelet och/eller spelmotorn när du komponerar?</p>	<p><i>"[...] jag tycker inte att det går att skriva musik utan att veta hur den kommer att fungera i spelet."</i></p> <p><i>"Jag tycker att det är både ur ett kreativt perspektiv men också hur musiken behöver vara strukturerad i exporten typ, eller kommer det vara ett intro som sen går in i en loop?"</i></p> <p><i>"Jag brukar fråga svinmycket frågor bara. För det där kan det bli lite clash att man inte pratar riktigt samma språk. Jag tycker att implementeringsbiten blir ett sätt att prata lite mer samma språk men ändå kunna ha den här kreativa representationen av. "Vad vill vi uppnå med att det ens finns musik här?" "Vad vill vi liksom säga till spelaren?" Klyschor, men "Vill att spelarna ska känna sig lugn, är det ingen fara?" "Vill vi att det ska kännas som att någonting är fel?". Så, vad vill vi säga, och framförallt också "Vad fyller den för funktion?" För att om det inte finns någon svar på den frågan så tycker jag inte att det borde vara musik."</i></p> <p><i>"Jag brukar så mycket jag kan, innan jag ens skriver en ton, försöka förstå vad det är jag skriver till."</i></p>

Beskriv en typisk planering för ett musikstycke för ett spel.

**[Frågan ställdes inte uttryckligen i intervjun, detta svarades till frågan; Hur mycket tänker du på hur musiken interagerar med spelet eller spelmotorn?]**

*"Jag gjorde kort projekt nu till global game jam, där jag inte var inblandad i motorn. [...] jag ville göra något lite remotely och lite casually. Typ skriva lite musik och skicka iväg den bara."*

*"Det påminner mig verkligen om den här känslan av att vara lite så låst utanför och typ inte riktigt kunna förstå. Det tog dels mer jobb att bara fråga så, "okej men hur är koden strukturerad?" "Är det en fas här och sen byter koden till en annan fas här eller är det samma?" Det finns massor av variabler som gör att "då borde det vara samma musik, men med olika detaljer i båda tracksen." Eller "är det två helt olika musikgrejer?" Vad vill vi säga med den första och den andra, en massa sådana grejer som bara "just det, fan vad skönt det är att vara inne i motorn"."*

*"[...] jag tycker verkligen att kompositörer borde vara inne tidigare i processen. Just för att kunna ställa de frågorna och också inte hamna i ett läge där det är så... Jag är på ett sätt trött på "aa vi behöver bara ett vibe track till vårt spel." Ja, jo, det är väl fett... men vad vill ni säga med det då? Jag tror att det finns mer att säga här med musiken om vi har tid att ställa de frågorna."*

*"Jag är ju inte en programmerare. Det tycker jag också är en viktig aspekt. [...] Jag kan ändå mina if-satser och mina enums, och jag vet vad jag ska leta efter. [...] man behöver ju ändå ha kontakt med utvecklaren. Även om man är självständig, så jätteviktigt, tycker jag."*

<p>Vilken roll spelar implementeringsverktygen när man arbetar iterativt?</p>	<p>"[...] iterationsprocessen är ju en av de stora bitarna där man kan liksom inte testa spelmusik utanför motorn eller inte i spelet och säga med säkerhet att inte funkar eller funkar."</p> <p>"[...] att förkorta den processen och snabbt kunna testa det i spelet och utvärdera, det sparar ju extremt mycket tid."</p> <p>"Vilket faktiskt kan leda till mera iterationer av musiken [...] någonstans blir ju varje iteration i bästa fall, bättre."</p> <p>"Skickar iväg ett track, utan att vara inblandad själv i motorn eller utan att implementera själv så kanske det tar tre fyra veckor innan man får återkoppling och sen "Nej men det var fel, vi behöver den här grejen istället." så gör man det och så skickar man iväg och så tar det två veckor till."</p> <p>"[...] det blir en så otroligt lång process för någonting som man hade kunnat gå in och testa själv på 10 minuter."</p> <p>"Det uppmuntrar inte direkt hela den här processen att ha livemusiker som kostar pengar i en studio om man inte vet exakt vad man ska spela in."</p> <p>"Då tycker jag att både middlewares och att ha kontroll över sin egen implementering är en stor del i att kunna också ha kontroll över det kreativa slutresultatet."</p> <p><b>[Frågan som ställdes här var även: Det är tiden som man tjänar mest på, skulle du hålla med?]</b></p> <p>"Iterationsmöjligheter. För som sagt, varje iteration i teorin gör saker bättre."</p> <p>"Det är klart att det finns säkert gånger man over-engineerar musik också, men jag tänker ändå att varje iteration kommer från någonting. En känsla av att: Det här funkar inte. "Okej, men vad är det som inte funkar?" Då kan man ha de där diskussionerna ganska snabbt. "Det funkar inte." "Okej, men varför och vad är det vi vill uppnå med en förändring?" Okej men cool. Då kommer vi hela tiden lite närmare känslan."</p> <p>"För man kanske inte heller vet vad det är från början, det är ju inte alltid man vet det."</p> <p>"Komposition är ju någonstans en långsam improvisation. Det är ju inte alltid man vet exakt, men varje test ger en information som leder en vidare."</p> <p>"Men jag tycker verkligen att det borde vara liksom: skriva någonting, implementera det, Skriv nånting, implementera det. Det borde vara ett mycket mer cykliskt förhållande, för annars vet man inte vad man håller på med."</p> <p>"Det är säkert de som har en fantastisk känsla av att hitta rätt exakt på första gången. Men 99 procent av oss testar ju saker. För att man har en känsla av vad som ska och kan funka."</p> <p>"Ju fler iterationer man hinner med för att man är involverad</p>
---	--

	<p><i>i systemen själv, desto mer tid kan man lägga på att skriva musiken."</i></p> <p><i>"Och desto mindre tid kanske man kan lägga på att diskutera och bråka över vem som sa fel eller var otydlig i beställningen och så har man skickat någonting som inte funkade för att någon har misstolkat."</i></p> <p><i>([...] det blir bara så mycket tid man lägger på att bara diskutera. Jag säger inte att det händer så ofta, men det går åt mycket tid att komma fram till vilken musik som ska vara på vilket ställe.)</i></p>
<p>Hur gör man för att effektivt testa hur musiken fungerar i ett dataspel?</p>	<p><b>[Vilka skillnader tycker du det finns mellan att jobba med kontra utan middlewares när man gör musik? Påverkar det någonting hur du komponerar din musik, i så fall hur ser det ut?]</b></p> <p><i>"Det första som comes to mind är att man får lägga en jävla massa tid på att bygga system som egentligen redan finns i ett program om man bara hade kunnat använda det programmet."</i></p> <p><i>"[...] Istället får jag bara skriva mer musik."</i></p> <p><i>"Och med middleware så kanske man lägger undan mer tid i början. Framför allt om det är utvecklare som inte har jobbat med det tidigare att bara få det att funka. Men det är ju förhållandevis en väldigt kort period av en utvecklingsprocess."</i></p> <p><i>"Och just att man också kan vara i ett ekosystem man också är bekväm och bekant med. Jag sitter mest i FMOD och det är ju... Det är ju väldigt DAW-liktsom. Det är ju en stor fördel med att jobba i ett ekosystem som man är ganska van att jobba med. Jag uppskattar också egentligen den här uppdelningen som kan vara med att man kan jobba i FMOD utan att nödvändigtvis vara inne i versionshanterings-messen med att jag vill sätta in en rad kod i något script."</i></p>

<p>Vilka utmaningar har man som kompositör när man komponerar musik för dataspel?</p>	<p><b>[Frågan ställdes inte uttryckligen i intervjun, sades i samband med frågan; Det kanske finns en viss uppfattning att musiken bara ska vara i bakgrunden att sätta den här känslan och inte vara jätteadaptiv. Hur ser du på den uppfattningen kring vilken funktion musiken ska ha?]</b></p> <p><i>"Jag upplever ju att det finns ett stort glapp i hur mycket folk förstår sig på ljud generellt, men i förlängningen också musik."</i></p> <p><i>"[...] jag tror jättemycket på att om man tycker att det är fett med interaktiv musik som kompositör, då tycker jag också att man ska vara involverad i implementeringen."</i></p> <p>Att behöva fråga om tid för att kunna utveckla system för interaktiv musik i spelet om man inte själv kan göra det. Kan vara svårt för övriga i projektet att se ett värde i interaktiva musiksystem och tiden som läggs på det.</p> <p><i>"Men om man själv kan gå in och ändå bygga de systemen så behöver man kanske inte ens fråga i den bemärkelsen."</i></p> <p>Tid ägnad åt att diskutera med medarbetare vilken musik som ska vara var.</p> <p><b>[Övriga punkter som uppdagats utmed intervjun]</b></p> <p>Viktigaste är att skapa bra musik.</p> <p>Tekniska aspekter av dataspelsutveckling. (exempelvis programmering)</p>
---	---

## Appendix B – Gustaf Blix Intervjuutdrag

Fråga	Svar
Berätta lite om dig själv och om din kompositionskarriär. (vilka projekt har du jobbat på? etc.)	<p>Spelat gitarr i rockband och började skriva musik till datorn i tonåren. Studerat komposition, klassisk. Grundläggande mediakomposition.</p> <p>Sjävlärd spelmusikkompositör.</p> <p>Frilansat med musikuppdrag och videoproduktion, jobbat som lärare.</p> <p>Haft stort intresse för dataspel sedan länge. Jobbat på Spelen "Sulfur", första jobbet, och "omelet you cook". Sjävlärd med middleware program. Involverad med många spelprojekt som inte haft finansiering eller är pågående varav några lades ner.</p> <p>Arbetat med symfoniorkestrar, både orginalkompositioner och arrangemang.</p> <p>Haft en relativt kort karriär inom spelbranchen hittills.</p> <p>"försöka samla på mig influenser från så många olika håll som möjligt"</p>
Fråga om ett projekt mer i detalj. (Berätta om vilken musik som gjordes för det projektet.)	<p>Omelet you cook. Spel gjort av två personer.</p> <p>"Jazzig" musik.</p> <p>Idén av hur musiken ska fungera och implementeras i spelet kom från programmeraren.</p> <p>Matchar musiken till spelets olika karaktärers karaktärsdrag.</p> <p>Musiken har samma tempo genom hela spelet.</p> <p>39 olika spår [av musik] som kan kombineras på vilket sätt som helst.</p>
Vad är musikens syfte i spelet?	<p>[Omelete you cook]</p> <p>Gestalta olika karaktärers karaktärsdrag.</p> <p>Sätta "känslan" för spelet.</p> <p>[Fick du något direktiv på hur musiken behövde vara eftersom ni inte använde en middleware för det projektet?]</p> <p><i>"[...] bara att de [musikstämmorna] ska funka ihop oavsett var det är för kombination."</i></p> <p><i>"Men det är exakt samma låt, samma tempo hela tiden. Just nu är det 39 olika tracks, det kanske blir fler, och alla dem funkar ihop. Så det kan bli vilka kombinationer som helst."</i></p>
Har musiken någon tilltänkt narrativ, spelmekansik funktion eller väcka en viss känsla? Hur arbetade du för att uppnå det?	<p>"AABA"-form på musiken, jazz brukar vara uppstrukturerat så.</p> <p>Gestalta karaktärernas olika karaktärsdrag.</p> <p>"Det är ju samma låt hela tiden, förutom när man möter bossen, vilket man gör efter ganska många timmar. Och då var det ju viktigt att inte bli enformigt och tråkigt."</p> <p>Använde olika kompositionstekniker för att kunna variera</p>

	<p>musiken utan att ändra på tidsomfonget av den färdiga musiken. Ändrade tonart, vilka instrument som spelades samt vilka delar av musiken som spelades.</p> <p>Har ett huvudtema som är lätt att känna igen.</p>
Hur arbetade ni/du med att få in musiken i spelet?	<p>[Omelete you cook]</p> <p>Arbetat med att få musiken att fungera som den var tilltänkt.</p> <p>Har behövt kommunicera mellan kompositör och programmerare för att kunna förstå varandra, vad de behöver och hur det har påverkat arbetsflödet.</p> <p>Inte använt middleware för det projektet.</p> <p>Behövde korta av reverb-svansen för att få den klinga ut i början av loopen, så att det inte blir ett abrupt avklippt ljud. Behövde göra om detta på alla stems efter att det uppdagats i implementeringen att det är bästa lösningen för projektet då.</p>
Har du arbetat med middlewares? Vilka då?	Wwise, FMOD
Varför har du använt dig av dem [middlewares] och vad är det du uppskattar med dem?	<p>[Frågan ställdes inte uttryckligen i intervjun]</p> <p>Loopfunktionaliteten</p> <p>"då använde jag den här "scatter" funktionen för att få en unik ljudeffekt varje gång."</p>
Hur får verktygen dig att tänka på komponerandet av musiken?	<p>"Istället för att tänka linjärt så kan man tänka modulärt."</p> <p>"Det funkar bäst för väldigt atmosfärisk musik, där det inte händer så mycket, där det inte behövs någon dramaturgi."</p> <p>"Då kan man ha som pusselbitar som kan komma i vilken ordning som helst, kanske flera lager som både horisontellt och vertikalt. Det sättet gör man inte musik på så ofta annars."</p>
Hur mycket tänkte du på hur musiken interagerar med spelet och/eller spelmotorn när du komponerar?	<p>[Sulfur]</p> <p>"jag faktiskt spelet i gång samtidigt som jag satt och försökte hitta på idéer för att komma in i atmosfären och sådär."</p>
Hur är musiken tänkt att interagera med spelaren?	<p>"I båda de spelen [Omelete you cook och Sulfur] så ändras musiken egentligen bara mellan banor eller mellan nivåer."</p> <p>"Men däremot med det här andra [dataspelet] som jag gör"</p> <p>"man kommer inom en radie, fiendens radie, och då ändras musiken lite grann. Sen när fienden dyker upp så ändras den väldigt mycket."</p>

<p>Beskriv en typisk planering för ett musikstycke för ett spel.</p>	<p>"Det är att hitta huvudteman och "spikar" dem typ. Det underlättar så mycket sen."</p> <p>"bestämna sig för vilken typ av, antingen musikstil eller vilka sorters ljud man vill ha. för det kan vara så otroligt olika beroende på vad det är för spel"</p> <p>"det är ju bra att bestämma om det ska vara interaktiv musik eller inte, det avgör om vi kommer behöva middleware"</p> <p>"det är roligt om man är från början för då har man lite mer inflytande, det är roligt att vara delaktig i de sakerna också."</p>
<p>I vilket skede börjar man tänka på kompositionen i termer som implementering?</p>	<p>Så fort han vet om det behöver göras många interaktiva komponenter. Har mest sysslat med linjär musik.</p>
<p>Vilken roll spelar implementeringsverktygen när man arbetar iterativt?</p>	<p>"De [implementationsverktygen, FMOD, Wwise, osv.] kan ju ungefär samma sak så man behöver egentligen inte veta vilket man ska använda när man gör musiken."</p> <p>"Det spelar mer roll om man har dem eller inte, för det avgör verkligen hur man kan utforma musiken. Till exempel det här med loop-funktionaliteten."</p> <p>"Har man inte det och om man använder Unity då, hädanefter kommer jag att använda det här tricket som jag använder nu."</p> <p>"Det är viktigt att kunna [använda middlewares], men det är också viktigt att kunna avstå."</p> <p>"det är bättre att vara så enkel att jobba med som möjligt än att jag kunna göra en massa andra coola avancerade tekniska grejer."</p> <p>[Ställdes i frågan "Hur arbetade nu för att få in musiken i spelet?]</p> <p>Behövde korta av reverb-svansen för att få den klinga ut i början av loopen, så att det inte blir ett abrupt avklippt ljud. Behövde göra om detta på alla stems efter att det uppdagats i implementeringen att det är bästa lösningen för projektet då.</p> <p>"programmerare har jäkligt mycket att göra ofta. Så om man lägger en sån grej på dem att först ska de fatta vad man menar, om de inte själva kan musik ska man först försöka beskriva det och sen är det schystare mot dem och mot en själv om man bara försöker hitta den nya lösningen"</p>
<p>Hur gör man för att effektivt testa hur musiken fungerar i ett dataspel?</p>	<p>Kommunicera mycket med programmeraren.</p> <p>Har musiken/DAW på ena skärmen och Spelet på andra skärmen. Musiken är inte kopplad till spelet live, utan bara för att få känslan av spelet samtidigt som det komponeras musik för spelet.</p>

<p>Vilka utmaningar har man som kompositör när man komponerar musik för dataspel?</p>	<p>Spelmusik innefattar många fler stilar än vad film och teatermusik gör. I spel kan man ge musiken mycket mer plats, den måste inte vara i bakgrunden hela tiden, jämfört med film och teater.</p> <p>Den kan också ta mycket plats, och om den gör det måste man försöka göra så att det inte blir irriterande att lyssna på hela tiden. Behöver hitta en balans.</p> <p>Behöver tänka mindre linärt, inbjuder till ett nytt sätt att tänka. Tänka i lager snarare än en enda lång tidslinje.</p> <p>"Det är ju samma låt hela tiden, förutom när man möter bossen, vilket man gör efter ganska många timmar. Och då var det ju viktigt att inte bli enformigt och tråkigt." (Detta sades i frågan "Hur arbetade du för att få ett narrativ i musiken?")</p>
<p>Hur undviker man att trötta ut lyssnaren med musiken?</p>	<p>Lyssna på musiken själv och se ifall man tröttnar.</p> <p>Proppa fullt med olika saker, man kan alltid plocka bort i efterhand. Lättare att ta bort än att lägga till.</p> <p>Ser det som ett marmorblock som man sakta formar när man skär bort delar av blocket.</p>

## Appendix C – Jacob Westberg Intervjuutdrag

Fråga	Svar
<p>Berätta lite om dig själv och om din kompositionsriktiga karriär. (vilka projekt har du jobbat på? etc.)</p>	<p>Arbetar som spelkompositör. Gått på kulturskola, spelat instrument. Läst extrakurser på Gymnasiet, musikproduktion. Har arbetat med musik på andra sätt än med dataspel. Även undervisat i dataspelsmusik.</p> <p>Akademiskt: Har en masterexamen i musikproduktion från Kungliga Musikhögskolan i Stockholm. Har även studerat musikvetenskap vid Uppsala Universitet. Har även studerat ljudproduktion vid Högskolan i Blekinge. Liknande situation på Högskolan i Blekinge som Högskolan i Skövde. Parallella program som bildar spelprojektsteams.</p> <p>Studerat programmering, utvecklat appar. Jobbat med spelprojekt i kombination med dessa studier.</p> <p>Frilansat efter studierna, jobbat på olika spelprojekt.</p> <p>Jobbar för en stor svensk spelutvecklare för tillfället.</p> <p>Tagit fram ett kompositionsverktyg: ≡ AMAPP.</p>
<p>Fråga om ett projekt mer i detalj. (Berätta om vilken musik som gjordes för det projektet.)</p>	<p><b>[Frågan ställdes inte uttryckligen i intervjun, men det berättades om olika tillfällen som man löst problem för olika spelprojekt.]</b></p> <p>Ett projekt: "DJ-spel", vandrar i menyn och då spelas en "ackordvända".</p> <p>≡ AMAPP: Hur fungerar denna app? I en DAW kan du börja planera upplägg och rendera ut adaptiva musiklager. Den underlättar att arbeta i ett linjärt verktyg som en DAW och gör den mer anpassad för ett icke-linjärt arbetsflöde.</p> <p>Little wing deliveries. Ett spel som ≡ AMAPP testades att användas i en Studiomiljö. Iterationsprocessen var mycket kortare och därmed kunde man prova hur slutresultatet blir mycket snabbare.</p>

<p>Hur arbetade ni/du med att få in musiken i spelet?</p>	<p>Little Wing Deliveries:          Spelade in i en studiomiljö med förberätt musikaliskt material. Därefter spelade musiker i studion till spelet som kör i bakgrunden samtidigt som de spelar in det som improviseras fram.          Detta gick via en redan färdig förberedd pipeline i Studio/inspelning -&gt; Reaper -&gt; Wwise -&gt; Unity. Detta kunde göras på plats samtidigt som det spelades in i en studiomiljö.</p>
<p>Har du arbetat med middlewares? Vilka då?</p>	<p>FMOD, Wwise.</p>
<p>Hur får verktygen dig att tänka på komponerandet av musiken?</p>	<p>Konceptualiserar problem olika beroende på vilket verktyg som man använder. Det påverkar, men svårt att svara på exakt vad som påverkar vad eftersom det är så olika mellan olika projekt.</p>
<p>Hur är musiken tänkt att interagera med spelaren?</p>	<p><b>[Frågan ställdes inte uttryckligen i intervjun]</b></p> <p>Ett projekt: "DJ-spel", vandrar i menyn och då spelas en "ackordvända".</p> <p><i>Det har varit olika från inte bara projekt utan från kontext till kontext. Liksom "Ah nu ska vi göra en bossfight här. Den är strukturerad på det här sättet, den ser ut så här." Till att "Nu är det ett helt nytt sammanhang här som vi kanske utforskar, som man inte riktigt vet vad det kommer bli i slutändan. Men designen är det här." Någonting som jag brukar försöka återkoppla till som funkar för mig är att jag är otroligt intresserad av speldesign överlag, och vad är det för kontext som musiken kommer att vara i?</i></p> <p>I Jacobs undervisning ges exempel utifrån tumbrottning, hur man kan tänka om man ska skapa musik till det. Hur musiken relaterar till speldesign.</p> <p><i>Av de enkla reglerna som tumbrottning är, alltså, man börjar med en liten ramsa och så sätter man igång. Så fajtas man lite grann och sen så har någon vunnit. Där finns det några enkla regler som skapar ett väldigt intressant spel.</i></p> <p><i>Och det är den kontexten som musik, om vi ska skriva musik till något sånt, det är den kontexten som musik hamnar i. Vad är det för känslor vi illustrerar? Hur lång tid det är mellan att man sätter igång tills man är klar? Hur löser vi då när man ska vinna eller förlora? Hur låter det då?</i></p> <p><i>Det finns några moment här som är viktiga för just speldesignens skull. Men när jag väl sätter mig och kanske skriver någonting för att "nu ska vi ha en liten fight här". Då kan jag börja med att jag skriver ett musikverk så att det känns som ett, som man tar fram ett källmaterial som kanske består av lite olika instrument och lite övergångar och så där och så tänker jag "ungefär så här kommer det sluta, ungefär</i></p>

	<i>så här kommer börja</i> ".
I vilket skede börjar man tänka på kompositionen i termer som implementering?	<p>Det är olika från både projekt till projekt och från kontext till kontext. "Vilken typ av spel är det? Vad ska hända i spelet? Vad är spelarens mål?"</p> <p>Tror att alla som håller på med adaptiv musik har med sig i bakhuvudet: "Om man smyger här ska det finnas en viss typ av musik här, om man blir jagad så byter man till någonting annat". Det är den kontexten som avgör vilken typ av musik som skrivs till ett spelprojekt. "Vilka känslor vill man illustrera? Hur lång tid är det mellan att man sätter igång och att man är klar med ett spelmoment?"</p> <p>I sitt program där han lär ut spelmusik tar han upp ett exempel. Om man ska skriva till ett stridsmoment i ett spel brukar han tänka att man skriver ett musikstycke med olika instrument och delar/övergångar. Det kommer låta "ungefär såhär" när det börjar och slutar. Sedan delar han upp detta musikstycke i "intro, main loop, outro". Utifrån det kan man sedan extrahera ut olika variationer av det. Blir ett större system i slutändan men att grundidén är att jobba som att vara "musiker". [jobba och komponera linjärt/med intro, vers, outro]. Vill skriva musik som har rörelse hela tiden.</p> <p>Sättet man tar fram musiken på, vilka verktyg man använder, gör också att resultatet blir annorlunda.</p>

<p>Vilken roll spelar implementeringsverktygen när man arbetar iterativt?</p>	<p>Den största tidsboven för någon som gör musik till spel (om man inte kan implementera musiken själv) är den iterativa processen. Den är alldeles för lång. Först måste man rendera ut allting och se till att det blir bra, sedan om man inte har någon middleware behöver man förklara för programmeraren vilka system som ska byggas, hur ska musiken fungera, sedan när det väl är implementerat i spelet och är i sitt rätta tillstånd, först då kan man utvärdera om det blev bra eller inte. Då kan det visa sig att det inte var bra alls och måste då hitta en annan lösning.</p> <p>Ju kortare iterativ process desto snabbare kan man ta kreativa beslut.</p> <p>Efter första iterationen, förutsatt att man inte ändrar så mycket i systemen efter det, brukar det gå mycket snabbare att iterera. Vägen dit dock kan vara väldigt mödosam när man sitter i ett komplicerat spelprojekt.</p> <p>Med dessa olika verktyg (Reaper, AMAPP, Wwise och Unity) kunde Jacob förbereda implementeringen i alla programmen för att sedan kunna snabbt implementera det som producerades i en studiomiljö och fortsätta vara i den kreativa processen, utan att behöva vänta på flera steg däremellan. Musiken kunde itereras på plats med musikerna tack vare allt förarbete i de olika programmen, vilket möjliggjorde detta.</p> <p>Detta gör att man ser på problem på ett annat sätt än om man lämnat den processen ett tag, gjort andra saker däremellan och syslat med andra lösningar till andra problem för att sedan gå tillbaka till projektet och kunna bedöma resultatet först efter allting är inne i spelet som det är tänkt eller inte. Om man kan iterera på detta snabbare kan man också göra bedömningen ifall det fungerar för spelet eller inte redan när man komponerar musiken för att undvika eventuella överaskningar på saker som uppdagas först efter implementeringen är gjord. Är detta långt efter inspleningen kanske man behöver spela in saker igen på nytt eller kompromissa med sin konstnärliga vision på andra sätt.</p> <p>Lägger ingen värdering i ju snabbare iteration desto bättre utifrån ett konstnärligt perspektiv, men utifrån ett tid-och-pengar perspektiv är det att föredra. Men konst är konst och måste få lov att ta tid. Man kan behöva sitta med vissa saker länge och vända och vrida på koncept eller idéer för att se vad som bäst passar projektet. Tar man AI som jämförelse, den kan producera mycket mer på kortare tid, men är det den producerar vad som är intressant för projektet?</p>
---	--

<p>Hur gör man för att effektivt testa hur musiken fungerar i ett dataspel?</p>	<p>[Finns det exempel på hur du gjort när du använt en middleware respektive inte använt en middleware? Hur har de skiljt sig åt i så fall?]</p> <p>Har inget konkret exempel.</p> <p>Berättar att när man skapar saker utifrån ett koncept, när det inte finns mycket av ett spel i början, har det sällan blivit rätt från början. Först när grafik och en spelmiljö kommit in i bilden som man kan göra mer utförliga skraddarsydda lösningar och musikstycken.</p> <p>Försöker realisera implementationen, den adaptiva lösningen i Wwise eller FMOD, sedan lyssna på hur övergångarna exempelvis låter mellan två tillfällen. Då kanske mängden trumfill ändras på, var de spelas men också hur kadensen i musiken är strukturerad. Det hjälper Jacob att förstå hur musiken och systemet ska fungera i slutändan.</p> <p>Med AMAPP kan jag testa hur olika vertikala lager låter samt övergångar mellan olika delar.</p>
---	--

Vilka utmaningar har man som kompositör när man komponerar musik för dataspel?

*Har du tur, eller de gånger som jag har haft tur, då har jag kommit in väldigt tidigt i produktion. Och kunnat vara med och förvalta, bygga idéer och lägga visionen och så där väldigt tidigt. Så att när vi väl sen går in i produktion så finns det en förståelse i utvecklingsteamet om att "sen så ska vi göra... Det här kommer bli ett adaptivt musiksysteem och det har förutsättningarna, de här kraven att det ska kunna möjliggöras."*

*Men ju fler grejer man tar på sig, ju större projektet blir så står man också med den förväntningen av att "då ska det här bli någonting sen".*

*Hur scopear man det på ett sätt så att man vet när man väl går in och börjar jobba med projektet, "vad är det som är viktigt" så att man lägger tid, först och främst på det som är viktigt och inte försöker bygga komplicerade lösningar på sånt som är oviktigt.*

*Det finns en ouppnåelig förväntan över vad gör man med... "Hur snabbt går att ta fram ett musikverk ungefär? Hur snabbt kan man jobba?" till exempel. Bara den oförståelsen bland folk som man jobbar med i spelbranschen gör att det blir väldigt komplicerat att navigera kring sånt.*

*Att kunna beskriva "vad kan jag göra utefter de förutsättningar vi har" tjänar man väldigt mycket på, att bli duktig på det.*

**[Fråga om vilka tekniker som används vid komponerandet av dataspelsmusik.]**

*Jag tycker att vertical layering är ett otroligt kraftfullt- ett väldigt enkelt typ av system som kan göra väldigt stor skillnad.*

**[Övriga saker som uppdagats i intervjun]**

*Det är lätt att börja fantisera om när man har all den här möjligheten att påverka musiken utefter spelets tillstånd, då kan det bli otroligt komplexa system som skulle vara otroligt spännande att lyssna på. Men att förverkliga de idéerna och förstå hur man tar sig dit kan vara ett jättejobb.*

*Vi [som jobbar med dataspelsmusik] blir experter på [...] "hur tar vi ett tema eller ett motiv och förvaltar det genom spelet i olika sammanhang beroende på vad som har hänt med hjälp av adaptiva tankesätt?"*

*Det är bra att tänka "enkelt" som ett första steg, att tänka "vad är det mest enkla adaptiva systemet med musik som låter bra", och sen så tänka på att "det här kan man börja extrahera ut, göra fler lager eller bygga till en till sektion på*

	<p><i>det". Om systemen som sedan ska spela upp där är så pass flexibla, så att du kan lägga in saker.</i></p> <p><i><u>En middleware hjälper dig att bygga nya delar till musik eller bygga vidare på lösningar väldigt väl.</u> Det är någonting som jag har jobbat på och varit intresserad av under en längre tid och som jag har fått en känsla kring.</i></p> <p><i>Hur skriver jag adaptiv musik utefter mina förutsättningar, och min stil som jag är intresserad av att skriva? Bara för att jag har skrivit musik i en viss genre och stil så betyder inte att de idéer och det adaptiva musiksysteem som blir av det går att applicera på andra genrer. Det känns inte heller självklart för mig att det är så.</i></p>
--	--