Crises as Temporality

A Critical Reimagination of the Networked Music Ensemble

via Live Coding Experimentation

CRISES AS TEMPORALITY

A CRITICAL REIMAGINATION OF THE NETWORKED MUSIC ENSEMBLE

VIA LIVE CODING EXPERIMENTATION

BY ALEJANDRO FRANCO BRIONES

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment

of the Requirements for the Degree Doctor of Philosophy

McMaster University DOCTOR OF PHILOSOPHY (2025) Hamilton, Ontario (Communication Studies and Media Arts)

TITLE: Crises as Temporality: A Critical Reimagination of the Networked Music Ensemble via Live Coding Experimentation

AUTHOR: Alejandro Franco Briones

SUPERVISOR: Dr. David Ogborn

NUMBER OF PAGES: ix, 296

Abstract

This research seeks to understand the role of music-making in environments heavily mediated by digital networked technology. I argue that music can be understood as a practice capable of anticipating shifts in the current mode of production and regimes of representation. Throughout the project, I unravel a form of subjectivity capable of overcoming the convergence of the crises of care, ecology, representational politics, and economy provoked by a capitalist class that consumes the means for its own reproduction and, with it, the means to reproduce life. This project does so by reimagining the arena of networked music as a collective non-commodified place for care and mutual aid. For this work, I have developed a research creation project that includes artwork, a piece of software for music exploration, and a written thesis, with all three components exploring the themes of temporality and crisis as mediated by networked computation and digital technologies. The first artwork discussed is *Temazcal 2*: a live-coded documentary co-created with Rolando Hernández. In this work we explore ideas about subjectivity and crises connected to the *temazcal*: both a sweat-lodge of pre-Hispanic origin common in southern and central Mexico as well as a canonic electroacoustic music work. The second artwork is *TimekNot*: a Domain Specific (Programming) Language designed to express polytemporal musical ideas and instantiate them as triggered audio samples. The third work is *La Fábrica Colapsada*: a cybernetic opera exploring the relationship between crisis and time as revealed in the stories of the 2017 earthquake of Mexico City. In observing the earthquake from a disaster studies perspective, I argue that the current music creation context can be seen as a disaster, engulfed in crisis, as well. From this perspective, I argue that within the algorithmic networked ensemble, new ways of framing social relations can allow us to imagine a world where many worlds can fit.

Acknowledgement

No one is alone. Doctoral projects can only happen with others. This project is no exception. I want to thank so many people that made this work possible. Only some of them are on this page.

Thanks to my supervisor David Ogborn for all the crucial and committed support, including very meaningful meetings to nerd out on music, books and code. Thanks to my committee members, Sara Bannerman and Andrea Zeffiro, for all their encouragement to keep going and with whom I had hard (in a good way!), meaningful and caring conversations that made this dissertation a strong text. I could not ask for a better supervisory committee. I am deeply grateful.

Thanks to Rolando Hernández, with whom I failed to build *temazcales*. Thanks to Iván López, with whom I hallucinated *Pirarán* for the first time. Thanks to Diego Villaseñor for all the philosophy, programming and music, as well as the incredible back and forth of theories, code, listening sessions and support.

I want to express my love and gratitude to my family: my father Guillermo Franco Valencia, my mother María Isabel Briones Castañeda, and my brother Guillermo Franco Briones. Memo, Mamá y Papá.

Muchas gracias por todo su enorme apoyo.

Thanks to Cassandra Weimann and Lorraine Bell for providing such excellent support for my work and studies. I would also like to thank the Factory Media Centre and the Art Council of Windsor and Region for their support in the development of my art practice. As well I would like to thank Chris Myhr for his great feedback on concerts and artworks.

Thanks to these weirdos: Sharmeen Khan, seldom heard DJ and exemplar comrade; Geordie Dent, time-hacker, pizza-salad wizard and eviction warrior; Rossana Lara, gurú of music scholarship; Aida Khorsandi, outstanding anarchic cable modulator; Behrang Takhayori, new year's traveler companion; Mehrdad Jafari Rad, Dastgah navigator; Sarah Imrisek, amazing dream-to-Hydra translator; Clementine Oberst and Gil Niessen, the best peers I could ask for; Alejandro Tamayo, whose sound poem still resonates on the wall of my house; Michael Palumbo, excellent community operator; Susie Kim and Freddie Villarete, friends of many brain conversations; and César Juárez tlasokamati miek compa!

Especially many thanks to my partner Sara Swerdlyk, brilliant scholar and furious knitter without whom this PhD would not be possible.

Content

Introduction

This research seeks to understand the role of music-making in environments heavily mediated by digital networked technology. Throughout this work, I have come to understand music as a practice capable of anticipating shifts in the current mode of production and regimes of representation. This project – Crises as Temporality – aims to unravel a form of subjectivity capable of overcoming what Nancy Fraser describes as the multidimensional crisis of cannibal capitalism (2022): the convergence of the crises of care, ecology, representational politics and economy provoked by a capitalist class that consumes the means for its own reproduction and, with it, the means to reproduce life. This project does so by reimagining the arena of networked music as a collective non-commodified place for care and mutual aid. As part of this dissertation, I supplement Fraser's framework by appending to this multitude of crises the crisis of the imagination – through which it is easier to imagine the end of the world than the end of capitalism – explored by scholars of the British cultural studies tradition such as Frederic Jameson and Mark Fisher.

The crucial questions posed in this project are: How can the networked music ensemble become a site of mutual aid and social reproduction pushing back against Eurocentric, androcentric, and anthropocentric music institutions susceptible to market capture? How can the lived experiences of the performers and the abstractions acquired by programming skills, scholarly research, and music work be synthesised into something reminiscent of Massumi's (2013) 'lived abstraction' ? By music work, I refer to the work that produces music forms, algorithms, and techniques for new music: prominent in this research is polytemporality as a set of algorithms that generate a novel musical texture. Finally, how can this lived

abstraction – the embodied and sensorial experience of ideas and concepts – be organically bound to the internet, instantiated as software, and yet accessible for broad collective knowledge and artistic production?

This research has led me to develop a research creation project that includes artwork, a piece of software for music exploration, and a written thesis, with all three components exploring the themes of temporality and crisis as mediated by networked computation and digital technologies. I aim to critically intervene in the fields of internet studies, data cultures, time studies, sound studies, artistic practice (specifically algorithmic music and live coding), and computational science. In the written section of my dissertation, I employ Marxist-feminist theory, assemblage theory, and decolonial frameworks to contextualise, describe, and expand conceptually upon the interventions made by the artefacts created.

The main contributions of this research are threefold. Firstly, I have created a series of documented networked digital performance pieces and interactive digital archives that employ novel conceptions of time, software, and musical materials. The performances are: the on-the-fly documentary *Temazcal 2* presented at the Networked Imagination Laboratory in March 2022;[1] and the live-coded cyber-opera *La Fábrica Colapsada* also performed at McMaster's Networked Imagination Lab in May 2023.[2] These artefacts push forward a new understanding of the unfolding temporality of crises and the care networks

[1] *Temazcal 2* has video documentation uploaded to YouTube appended as an URL to the dissertation as appendix A.1.1. The performance also has a repository of audio, videos, images, texts, among many other materials that can be consulted as appendix A.1.2.

[2] *La Fábrica Colapsada* was first performed at Arraymusic in April 2023, this has video documentation uploaded to YouTube appended as an URL to the dissertation as appendix A.2.1. The most recent performance at the Networked Imagination Lab on May 2023 has audio documentation and it is hosted in a website that displays the story-telling mechanisms of the opera, the URL for it is appended to this dissertation as appendix A.2.2. The work also has three repositories of audio, videos, images, texts, code, among many other materials that can be consulted as appendix A.2.3, A.2.4 and A.2.5.

they inspire. Secondly, I have designed and produced TimekNot, a domain-specific programming language for networked improvisational music creation, as software in a public internet-bound repository capable of triggering audio samples within a polytemporal musical framework I developed.[3] This software is available for public use under a GNU General Public License, contributing a valuable tool for audiovisual creation to anyone interested in polytemporality, algorithmic music, and the live coding artistic movement. Additionally, I have created software for earthquake data sonification and spatialisation in the context of the networked music found in the repositories for *La Fábrica Colapsada*. Thirdly, I wrote this dissertation in which I weave together ideas and practice, arguing for a reconceptualisation of the electronic, networked music ensemble and media arts in an era of converging crises. My main theoretical contribution within my written doctoral work has been to reframe sonic media art practice as, primarily, the locus of new social relations of care and, secondly, a niche for forms of art-making that supersede gatekeeping by elite cultural expressions and the limits of industry and market.

La Fábrica Colapsada, the aforementioned final artwork for my PhD project, includes a documented performance of a hybrid – in-person and web-based – algorithmic opera based on the earthquake that catastrophically struck Mexico City in September 2017. The opera is a networked music and live coding act – where coding in front of an audience becomes performance art – and draws extensively from existing social and geological data, relevant scientific literature, and my personal experiences, both as a programmer-artist and as someone who experienced first-hand the earthquake. Multiple art-research methods based on data analysis and processing, like sonification, are fundamental aspects of the

[3] Video demonstrations hosted in YouTube of TimekNot's earliest experiments is appended as appendix A.3. TimekNot's source code is appended to this dissertation as appendix A.4.

artwork. Similarly, I have constituted multiple repositories of audiovisual digital materials that form a series of archives deployed and publicly accessible on the internet to be reconfigured with software as a live performance. The artwork highlights a specific conception of temporality and crisis revealed by the seismic event and the emerging social relations. I have drawn from Peer Illner, Tithi Bhattacharya, Gabor Maté, David Harvey, Nick Dyer-Witheford, Atle Mikkola Kjøsen, James Steinhoff, and other relevant critical theory literature to reconceptualise the site of the performance – in this case, the networked environments – and to critically approach the crisis engendered by the earthquake.

The earthquake theme of the opera allows me to expand on temporality and crisis and, additionally, to apply specific social reproduction theory ideas on disaster to the nature-culture divide that often obscures the effects of cannibal capitalism: namely, the naturalisation of human suffering – usually understood as an effect of earthquakes – without a proper understanding of failing institutions and markets. I propose that the real disaster is the retreating institutions whose purpose has become to liberate markets from human friction – an inversion of our common sense. I embody such inversion through the stories told as part of the artwork.

The circumstances surrounding the earthquake – particularly the fact that 32 years prior, an even more devastating earthquake hit Mexico City on the same date – enables me to trace the impact of neoliberalism as an ideology that has shaped reality in the period between the two earthquakes. The narrative I am piecing together interrogates the response to the catastrophe under state-managed capitalism and, now, under financialisation. What characteristics reappear? What is different among them? More relevant for this project is whether the texture of time, the way people experience time in these critical periods, affects normative temporality. What is possible when the people of Mexico City are

exposed to such a texture of time? What relations of care and social reproduction are constructed in the face of this immense destruction? And what can we learn from Mexico City's relationship with earthquakes in a post-pandemic, ever-in-crisis world? In exploring these questions with live coding and data sonification, my PhD research bridges disciplinary discussions within data studies, media arts, and internet studies on the role of networks and data beyond their disciplinary and commodifying roles.

This artistic project involved small but profound networked collaborations with a specific set of people, namely: Rolando Hernández Guzmán, an excellent sound artist and curator from Mexico City; Iván López Pineda, a brilliant composer, producer, and percussionist from Morelia; and Diego Villaseñor de Cortina, a philosopher, programmer, musician, and long-standing collaborator of mine from Mexico City as well.

I met the challenge of collaborations in networked environments with know-how developed as part of my PhD of live coding systems, workflows and software. I particularly relied on Estuary, TidalCycles, SuperCollider, Flok, Reaper, and Audiomovers – mostly open-source software. Perhaps central to my networked, collaborative expertise is Estuary, "a browser-based platform for live coding that allows a heterogeneous collection of live coding languages to be used together in collaborative 'ensembles'" (Ogborn et al., 2022, p. 1). As a past member of the research team and someone closely involved with the project, I see Estuary as both an influence on my understanding of networks and a set of possibilities to explore in developing performances and software.

I also developed code snippets, programs, and software, such as spatialisation software, interfaces, and custom-functions for MIDI communication, among many other things that emerged as requirements for our experiments. At first, the idea was to work with ensembles in a more structured manner, but as the COVID crisis and lockdown conditions deepened, it became necessary to pay more attention to aspects

of care and social reproduction around ensemble-formation than aspects of networked music production. Thus, this doctoral work ended forming two small-but-intimate ensembles preoccupied perhaps more on life than music and technology.

The artwork's infrastructure is a series of time-keeping, spatialisation, and datafication software developed as deliverables for this project. This software will be available in various repositories online most of which will be appended to this dissertation. This software stands as data and text that illuminates specific aspects of this research. However, the software – more or less minimal pieces of code designed to help users to accomplish specific tasks for audio and artistic work – is intended to aid performers to use as art-making tools and programmers to use as algorithms to integrate into their software. The most relevant component of this infrastructure is the programming language I am currently developing: TimekNot. This language enables performers to think about music-making in a radical polyphony style, where every monophonic line has its own rhythmic and/or metric rationality. By including time expressions as part of the program rather than assuming a 'master' clock, TimekNot makes it possible to think of points of convergence that communicate otherwise different musical materials. This computer language is a tool I am developing specifically for live coding, a growing art practice and movement characterised by the explicit use of code as performance art, in which on-the-fly generated code creates side-effects, like sound or visuals. In this context, the programming language is revealed as part of the live-coding show by projecting the performer's computer screen to the audience. Thus, the language I designed for my PhD research is made with meaningful and expressive syntax and grammar. This language aims to reflect upon the conventional understanding of the internet, often framed as (cyber)space, and focus on the temporalities afforded by networks.

I have drawn extensively from methods and theories associated with what Chapman and Sawchuk (2015) term 'research-creation', adopting a substantial trans-disciplinary approach where I combine digital humanities, social sciences, computer science, and art practice. I rely on a practice and movement that envisions programming as a cultural activity. If programming for scientific or industrial research is oriented towards the questions of how and why – where the interest lies in describing an object or defining a series of instructions for the computer to complete a task in the most efficient way possible – the programming I draw from, as expressed in the *Handbook for Live Coding* (Blackwell et al., 2022, p. 140), relies on the question: what if? Thus, the outcome of programming is not determined by a series of predefined desired objects or processes but by exploring unknown data, novel side-effects (such as music) and, especially, unforeseen social arrangements. The kind of programming embodied by live coding is thus a way to continuously develop, test, and share new theories on different relationships between data structures, usability in a community context, and art production. As a research methodology, live coding can produce a vast number of materials to analyse and engage with in a critical manner.

To complement the research creation aspects of my PhD project based on programming and music-making, I have engaged in theoretical explorations that attempt to make a partition between, on the one hand, artworks driven by marketization, and, on the other hand, anti-market art practices using techno-scientific frameworks while also embracing emancipation. This analysis is conceptually grounded in Nancy Fraser's triple movement (Fraser, 2020), which is an in-depth intervention into the social theory developed by the sociologist Karl Polanyi (2001 [1944]). Fraser's framework of the triple movement seeks to explain the relationship between marketization, social protection, and emancipation through, on the

one hand, recognizing the market's co-optation power by integrating a political-economy critique of capitalism into emancipatory struggles and, on the other hand, re-embedding the economy into society mediated by emancipatory movements.

My doctoral research has been shaped by cycles of art-creation, theoretical reflection, and software production; these cycles exist recursively and not necessarily in a linear sequence. However, a significant trend can be traced: art-creation leads to theoretical insights that can be instantiated as software for art creation. This methodology has emerged from the research material and my research interests, which is how I have organised the written dissertation. As explained in this dissertation, an artwork starts the cycle, theory follows, software is designed, and, finally, an artwork, and its theoretical aftermath, are created again.

I hesitate to participate in matters of positionality for a few reasons. Firstly, as a racialised man from the global south who has struggled with matters of immigration while finishing a PhD in a reactionary nation-state committed to politically scapegoating international students, I fear being fixed and reduced – under a positive or a negative light – into an identity rather than taken seriously as a committed scholar. Secondly, according to Srivastava (2024), DEI frameworks and dialogues within university settings have failed in the context of anti-racist oppression. Despite the presence of anti-racist politics in institutions like universities, racial conflict has increased in the past years. Furthermore, it is my personal experience that positionality, often supported through intersectionality, reveals more of what a scholar does not care about rather than what they do. It is no surprise to me that higher education institutions – and some of their members – show so little care about matters of class in contrast to gender, sexual orientation, or race.

However, I am grateful to be involved with people that take diversity, equity, and inclusion seriously: pushing the institutional limits further towards a space of social justice capable of dialogue and mindful of the limits and openings it entails. I believe the unfortunate historic moment in which we currently transit is an opportunity for people open to new ideas to reconstitute DEI frameworks perhaps closer to a social reproduction perspective where class and matters of identity converge more closely and better.

Reflecting upon this dissertation, I have thought of the artefact discussed in Chapter 1 as functioning as a positionality question rather than a statement. As will become evident while reading Chapter 1, the mestizaje framework is used by the Mexican state to identify me racially, as with the racial profile of most Mexicans. Not so evident are the questions of masculinity and male privilege – particularly in the context of music-making – that I have attempted to tackle via the ensemble practice I have developed. Lastly, matters of representation are key for the tuning capacities implemented as part of TimekNot.

Precursor and Chapter 1

As a precursor to my final project and the software TimekNot, as well as a key deliverable of my PhD project work, I have co-created with Rolando Hernández the live-coded documentary *Temazcal 2*. In the context of this dissertation a live-coded documentary stands as a montage of audiovisual materials emerging from the clash between conventional cinematic documentaries, live coding practice, and various tense positionalities and natural, as well as computational, languages. This artwork has allowed me to experiment with the affordances of networked environments, storytelling on the web, crisis as temporality, live coding strategies and techniques, and the online ensemble as mutual aid and social reproduction in the context of the global pandemic. *Temazcal 2*, as extensively described in my first

chapter, emerges in the context of the early pandemic, when musicians were confined to the internet and denied normal employment opportunities that would allow them to live from art practice. This artwork exists as a collection of multiple media items (videos, audio samples, images, code, text, etc.) that, when activated, allow my collaborator Rolando (a sound artist from Mexico City), Diego Villaseñor (a close collaborator of mine that would recur throughout my art practice), and I to perform a live-coded documentary exploring two notions of the *temazcal*: as a trope in specific Mexican modernist music environments and as a sweat lodge of prehispanic origin that is an important part of the family history of my collaborator. From the creation of this artwork, a set of ideas became evident for Rolando and me: Firstly, the *temazcal* technology cannot be rebuilt (neither metaphorically nor literally) by us without forcing relationships of extraction into others since the knowledge required for such an operation is lost from Rolando's family core. Secondly, considering the advance towards integrated global capitalism that has been occurring since the eighties – the time from which various acts of appropriation converged with electroacoustic music development in Mexico – the hegemonic arrangement that enabled appropriation has been put into question not only by emancipatory forces but also by market forces altogether. Third, by acknowledging the first and second points, we avoid *Temazcal 2* falling back into forms of subjectivity and narratives that unwillingly update the power of an oppressive nation-state. At the same time, we attempt to suppress any forms of capture by the forces that Fraser terms 'progressive neoliberalism.' A form of subjectivity emerges that is incommensurable to common figures related to colonialism, like the Indigenous, the settler, and the *mestizo*.

As observed, the theory is enriched by the research creation project and vice versa. Academic conversations regarding Indigenous studies, the settler-colonial complex, decolonial thought, Marxism,

and feminism assist in the creation of programming code, ephemeral programming languages, digital archives, and audiovisual documents, which simultaneously instantiate theoretical insights that allow the authors and audiences to have a rich experience beyond texts or art.

A 'double consciousness' emerges from the analysis of this artwork. On one hand, a sharp critique of the appropriation of Indigenous identity tied to the Mexican state and its modernist artists forms the core of *Temazcal 2*. In this instance, I refer to a particular English-educated Mexican composer and how some of their artworks respond to an ideological state mandate to assimilate Indigenous identity. On the other hand, a relationship with non-Western, non-scientific forms of knowledge is emphasised, which cannot be reduced to the global north's (or global north-aligned) critical studies drive to deny any form of situated knowledge and positive relationship with the land created by those who do not identify as Indigenous. Rolando's familial relationship with the land does not arise from Indigenous identity or past cultural experiences, but from a desire to resist the void subjectivity tied to forced migration, cultural assimilation, and proletarianisation.

I will refer to the concept of subjectivity presented in this chapter as 'double consciousness,' which captures the interplay and feedback between the 'progressive' modernity, where "production appears as the aim of mankind [sic] and wealth as the aim of production" (Marx, 1973 [1857–61], p. 488), and the local social forms, where "the human being appears as the end of production" (Marx, as quoted in Harvey, 2023, p. 223). This artwork reveals two distinct states of double consciousness. This double consciousness indicates a dynamic subject capable of creating tactical and necessary inversions of themselves by articulating class consciousness alongside cultural identity.

Precursor and Chapter 2

The other major precursor and deliverable of my doctoral research is the first version of the computer language TimekNot, which is currently available as a repository and software deployed in the collaborative, live coding environment Estuary. TimekNot, the artifact explored in the second chapter, is a modest programming language that allows live coders to create heterogeneous, music-oriented temporal relationships on the fly and instantiate them as triggered audio samples. In other words, it allows live coders to create music structures with audio samples in an improvisatory manner. TimekNot's core is a robust systematisation of time relations between relatively autonomous musical layers. Hence, it can be understood as a polytemporal language. The root of such an unconventional musical style and rationality is partly based on the explorations of Mexican experimental musicians and composers, but it is also the effect of living in a city like Mexico, where the soundscape is often occupied by multiple sound-systems blasting different music and sounds at the same time. Thus, polytemporality is an abstraction that originates in material and lived experience. What is interesting about the kind of polytemporality I am invoking is that difference is not an operation that needs to be solved toward identity. Neither does this polytemporality dismiss the effects of the multiple timelines on each other. Time does not need to be collapsed into universality, but polytemporality maintains the tension of difference, and different speeds, as a valuable cultural experience.

The final artwork, the opera mentioned previously, builds upon many aspects of the precursors discussed here. More precisely, the 'double consciousness' unveiled in Chapter 1 is a theoretical steppingstone, while the Domain Specific Language described in Chapter 2 is a primary tool and environment to reshape music relations with a networked ensemble.

Final Artwork and Chapter 3

In Chapter 3, I analyse – and theoretically expand upon – *La Fábrica Colapsada*, the opera I created as part of my doctoral work that re-envisions disaster through the lens of social reproduction, cybernetics, and cultural critique. Closely observing the circumstances and responses around the Mexico City earthquakes of 1985 and 2017, the opera allows me to elaborate on concepts such as 'toxic resilience' and 'generative' and 'reactive resistance' to explore how crises reshape time and introduce novel ways of sensing the world. The concepts mentioned here may further express the 'double consciousness' explored within *Temazcal 2*. The 'ways of sensing' I have envisioned through the analysis of the earthquakes has become a pattern to understand further the ways in which music creation operates as well, in a similar crisis to the one provoked by the earthquakes. Through the networked ensemble in which I participate, Pirarán, I argue for the rejection of 'toxic resilience,' instead rethinking technology and music-making away from profit-hoarding and cultural domination – and closer towards a framework that I came to understand as 'algorithmic acid communism.' Music-making finds its minimal material conditions within TimekNot and the technical ecology surrounding it.

Crises are often understood as ruptures in the flow of normal time, eventually reabsorbed into the empty temporality of capitalism once they subside. However, under cannibal capitalism – with its constant, overlapping crises – this framing is not adequate. The structure we inhabit, the social totality itself, *is the crisis*. Each rupture examined in this research unveils the imagination required to end the ongoing nightmare. Rather than viewing them as an array of crises, I trace critical periods in which historical, psychosocial, infrastructural, cultural, and lived experiences converge into technologies, art practices, and cultural expressions that point toward a world in which many other worlds fit.

Chapter 1 - Temazcal 2: Subjectivity, Crisis and the Internet

In this chapter, I discuss *Temazcal 2*, an artwork that is a collection of multiple media items such as videos, audio samples, images, code, text, etc. When activated, these assets assist my collaborator, Rolando, and me in performing a live-coded documentary exploring two notions of the *temazcal*: as a renowned piece by an electroacoustic music composer and as a type of sweat lodge of pre-Hispanic origin used in traditional healing practices in Mesoamerica. The latter technology serves as inspiration for the former. I begin the chapter by contextualizing my collaboration with Rolando Hernández, a sound and conceptual artist and curator from Mexico City, with whom I worked between 2017 and 2023 and who was my main collaborator for *Temazcal 2*. In executing this project, Roland and I also worked with Diego Villaseñor de Cortina, a philosopher, musician, and programmer from Mexico City, with whom I often collaborate. Throughout the chapter, I also refer to Rolando's and my earlier collaboration with César Juárez Joyner, a musician and film scholar also from Mexico City, as well as the organisations and institutions connected to *Temazcal 2*, like the Networked Imagination Lab and the Factory Media Centre. I start this section by aiming to contextualise the reception and creation process of this artwork. I introduce key conversations, precedents, and collaborations, while also preparing the reader for the challenging theoretical discussion in the second section.

In the second part of this chapter, I present a theoretical discussion that stands at the centre of the artwork: the tense relationship between the figures of the Indigenous, the settler, and the *mestizo* that emerged from my position as an immigrant from Mexico in a doctoral program in Canada and the material and historic conditions for the performance of *Temazcal 2* in 2022. The theoretical connections

presented here can be traced back to the creative process and the reception of *Temazcal 2*. This section reveals the affordances of art creation as a form of knowledge production, but more interestingly, it proposes a form of subjectivity that negates the settler, the Indigenous, and the *mestizo*: the three figures related to colonial societies in the context explored. Similarly, it proposes a double consciousness capable of articulating questions of land and labour.

In the third part of this chapter, I describe in detail the development of the artwork as a musical work, its storytelling aspects, and the research involved in its creation. Three main takeaways emerge as I guide the reader through the artwork's rationale. First, the *temazcal* technology, once a crucial component of Rolando's family life, cannot be recuperated by us without exercising relations of extraction onto others. Second, the necessary and relentless critique of artistic practices, such as electroacoustic music, and its impetus to appropriate and reshape technologies like the *temazcal* to advance the colonial project of *mestizaje* can be misconstrued in the absence of more generative narratives on the subject. Finally, artworks such as *Temazcal 2* describe and perform a form of incommensurable subjectivity that refuses to be explained as Indigenous or settler within the settler-colonial paradigm and through *mestizaje* as the Mexican state's colonial project. This form of subjectivity can be described as double consciousness: a consciousness that articulates the situated knowledge of Rolando's family alongside techniques of appropriation that transform a global project towards whiteness – like *mestizaje* – into a possibility of interconnected self-determination and a way of being in this world that fosters social relations beyond capitalism and nationalism.

Materiality and History

The collection of digital items referred to here takes the material form of an online repository containing materials created and curated by Rolando and me. The repository is primarily organised into indexed scenes, which imply a sequence that should be understood as the order in which the scenes are to be performed. There are no explicit instructions on how to play the scenes or how to arrange the specific materials for each scene, allowing performers to discuss and devise their own interpretations of the materials.

The number of performers needed for *Temazcal 2* can vary from two to any number of people; so far, the performance has been carried out by three people: Rolando, myself, and Diego, who acts as the 'interpreter' of the music by modifying on-the-fly code snippets composed by me. The repository is bound to (but not limited by) the software Estuary, which is the environment where the performance is executed. In Estuary, programming code and digital items are articulated and displayed as audiovisual signals that are mixed and composed together.

The live-coded documentary premiered simultaneously at the Networked Imagination Laboratory[4] at McMaster University, streamed online via the Factory Media Centre[5] on YouTube, and showcased in the *Intercuraduría* gallery[6] in Mexico City. It was received by two live audiences and the delocalized, deferred audience of the internet. Additionally, there is a recorded performance that can be presented as an

[4] The Networked Imagination Laboratory is in McMaster University. It is a collaborative space to conduct research on networked art-practices, live coding, spatial audio, video game development where a lot of my research took place as a doctoral student.
[5] Factory Media Centre is a not-for-profit artist-driven resource centre for film, video, new media, installation, sound art, and other multimedia art forms based in Hamilton.
[6] Intercuraduria was a project of *curadoras* (female curators) that attempt to rethink art curation in Latin-American.

experimental documentary. The register narrates in a very specific manner, employing an audiovisual algorithmic montage to depict the relationships that Rolando and I have with the Indigenous technology related to Rolando's family, which has inspired artworks like *Temazcal* (1984) composed by Javier Álvarez.

Originally, the *temazcal* (derived from the *Macehualcopa* word *temazcalli*, meaning steam house) is a ceremonial structure, a round, womb-like, ground-level sweat lodge, where ritual and medicinal ceremonies, often related to birth, take place. The *temazcal* is a traditional artifact connected to healing practices in the valley of Mexico and the southern part of the country sometimes bound to Nahua cultures. This medical practice has been displaced by Western medicine as people using *temazcales* for these purposes are uprooted by violent processes of dispossession. In the context of this artwork, *Temazcal* (capitalised, *Temazcal 1* from now on) is an iconic electroacoustic music piece for maracas and fixed media by Javier Álvarez, which premiered in the UK in 1987, that appropriates and re-signifies the *temazcal*. I argue that Álvarez's artwork simultaneously builds nationalist narratives of the Mexican state – where the assimilation of Indigenous knowledge is performed as a predecessor of the figure of the *mestizo*, a mix of Indigenous and white cultures, ultimately a whitening project – while also reifies Eurocentrism and tropes of progress by adhering closely to classical music traditions, as well as scientific understandings of sound and electroacoustic techniques and technologies. The artwork is composed using a counterpoint logic: while I analyse the electroacoustic piece *Temazcal 1*, Rolando offers the story of his family and their movements from the *Mixteca Alta* to Mexico City and back, along with their relationship with forms of knowledge and social reproduction strategies, some of which have been lost and others transformed in the face of capitalist destruction. The migration processes of both Rolando and me are, in many ways, fundamental to understanding this artwork.

Temazcal 2 was developed conceptually by Rolando Hernández and me from 2017 until the summer of 2022. In 2017, Rolando and I coincided in a course on *Macehualcopa* (the language of the Nahua peoples of the central part of Mexico), where we encountered a remarkably familiar yet profoundly distant culture compared to our everyday reality. Earlier that year, Rolando invited me to participate in the *Ensamble Negro*, a music formation consisting of Mexico City-based musicians and sound artists assembled for the occasion by the Umbral music sessions to perform the work of Peter Ablinger, an Austrian composer often associated with neo-conceptualism, as part of his visit to Mexico City. The conversations inspired by these two experiences slowly evolved into plans to develop artistic projects together that would explore time and Mesoamerican philosophies in depth. In 2018, we produced our first collaboration in which Rolando acted as a curator, and I programed, composed, and performed (along with 17 other musicians with unconventional practices) an interactive score for polytemporal music. In 2019, we finally co-created an opera performed by the free improvisation ensemble *Ruido Trece* and other guest instrumentalists and sound artists, including Rolando and me. The opera and the networked ensemble piece mentioned here explored notions of time like polytemporality but did not yet approach explicitly any notion of Mesoamerican philosophy.

In the spring of 2020, the COVID-19 crisis erupted with well-known global consequences. One consequence was the widespread suppression of performance arts in their conventional spaces, which was catastrophic for artists who depended on concert venues, museums, theatres, nightclubs, and all kinds of infrastructure for them to earn a living through their art practice or make life worth living by art practice. The early COVID-19 lockdown was an opportunity to articulate experimental technology in the service of artists that would quickly require the means to keep making art in the new living conditions.

This aspect of my research, amidst the unfolding catastrophe in which it operates, will be addressed in Chapter 3 a little bit further.

In this spirit, Rolando and I planned two networked performances for the summer of 2020: *Dos Sures*, a series of live-coded radio performances using the platform *Campo Sónico* (designed by Diego primarily to crowdsource sounds from Archive.org) for Radio Alhara; and *In Xiuh Ce Amatl,* an ambitious live-coded documentary that intended to problematise the renewed interest in Mexican identity that the current political crisis had inspired in certain Mexican intellectual circles: namely, the crisis of hegemony that challenged the (neo)liberal 'democratic' Mexican period and the strong comeback of nationalist sentiment that re-ignited questions of *mestizaje*, indigeneity, whiteness, and more. This crisis is also a fundamental aspect of *Temazcal 2,* and a thorough explanation of it will unfold in the next section of this chapter.

For *In Xiuh Ce Amatl* we collaborated with César Juárez Joyner, a composer, film director and performer of folk music, and a speaker of *Macehualcopa* that would introduce notions of Mesoamerican philosophies to shape the performance. The work was presented at the Networked Music Festival, which happened transnationally but was organised by people located in Newcastle, UK; we also performed the piece for the *Campamento Extendido Cyberpunk Posternura* online concert series, organised in Valdivia, Chile. This collaboration with César, and the connections made between Canada, Mexico, the UK, and Chile during the process, constituted a key exchange of ideas and conversations. This exchange allowed César to significantly advance his documentary filmmaking research while also enabling Rolando and me to envision a longer form that addresses a recurring trope we identified as central to our networked collaboration, which I will describe in the following section. Before delving into the analysis of *Temazcal*

2, I will elaborate on my collaboration with César, which offers essential insights for understanding the potency of the work discussed.

César Juárez Joyner positions himself in his master's dissertation (2021) as a 'functional speaker' of an Indigenous language (*Macehualcopa*); a language that he has, "since childhood, heard extinguishing in the voice of [his] grandfather (Juárez Joyner, 2021, p. 9)." This life experience has allowed him to explore the processes and limitations through which documentary film practice represents minoritised languages and peoples. According to this research, language functions as a repository of social relations. Thus, if documentary film practice is understood as a formal consequence of the transmutation between technique, media and language, this language must be considered part of a thought-process that transcends film tradition when engaging with the social relations embedded in Indigenous languages. If documentary film practice is not transformed by the Indigenous visions it attempts to represent, it risks becoming visual anthropology or ethnographic cinema, which risks fixing and solidifying the western-northern gaze. Juárez Joyner reveals through his research the need to explore other ways of telling the history of minoritised people that do not adapt language and cultures to the documentary form but transform documentary form into something that can adequately express the cultures that interpellate it.

In Xiuh Ce Amatl is a *difrasismo* – a grammatical construction in which two words or concepts are combined to form a third one as a metaphor – that can be literally translated as 'the fire and one paper;' a better translation is 'burning paper,' or perhaps more precisely, living memory. Via live coding, the work is a radical exploration of the way Nahua philosophies trans-mutate both coding and documentary film practice. This process is described as follows:

> *The notion of an archive does not refer to a physical space with specific records but rather to complex cultural and philosophical entanglements that our bodies receive and may later respond to through reasoning and stimuli. In this sense, collective memory is proposed as a space for triggering individual reflection processes, where the viewer reconstructs their own cultural history through unfamiliar narratives and speculative historiographic lines (Juárez Joyner, 2021, p. 101).*[7]

Juárez Joyner further describes the compositional logic applied to the audiovisual components of the documentary as follows:

> *A narrative axis was developed based on the Mexica notion of the guerra florida, in which war was seen as a necessary exercise for the circulation of time. This concept gave rise to a system of dualities (red/black, huehuetl/ teponaxtli) that permeates the entire work. It influenced elements such as the color palette and the approach to sound design—where, despite both operating in real-time, the algorithms shaping their behavior adhered to this philosophical system (Juárez Joyner, 2021, p. 103).*[8]

The work came to be understood by Juárez Joyner as an algorithmic montage that creates a juxtaposition of imagery and (natural) languages, as well as a juxtaposition of algorithms and computational languages. Furthermore, in the tradition of Sergei Eisenstein and others related to the Soviet Avant-Garde

[7] Automatic translation, original: La noción de archivo no corresponde a un espacio físico con registros puntuales sino a complejos entreveramientos culturales y filosóficos que nuestro cuerpo recibe y a los que posteriormente, tal vez reaccione a través de razonamientos y estímulos. En dicho sentido se propone la memoria colectiva como un espacio para detonar procesos de reflexión individual en los que el espectador recree su propia historia cultural bajo la forma de narrativas poco familiares y líneas historiográficas especuladas.

[8] Automatic translation, original: Se desarrolló un eje narrativo a través de la noción mexica de la guerra florida, en la que la guerra era un ejercicio necesario para la circulación del tiempo. Con ello se creó un sistema de dualidades (rojo / negro, huehuetl/teponaxtli) presente en toda la obra. Ello determinó elementos como la paleta de colores o el tipo de tratamiento sonoro que, aún cuando ambos fueran en tiempo real, los algoritmos que acotaban dicho comportamiento atendían a dicho sistema filosófico.

and Soviet Montage Theory cinema movement (Nemchenko, 2017), these juxtapositions are not merely contemplative. Instead they attempt to foreground the dialectical tensions, as reflected in the quote above and, pragmatically, infect life with the experience of the artwork. In the case of *In Xiuh Ce Amatl*, the audience was invited to conceive of memory as living, as a practice rather than a static repository of fixed images and ideas. Finally, this work was the first exploration of the enunciation of *Macehualcopa* in the context of live coding and an experiment of *Macehualcopa* as a meaningful medium to communicate with computers. More research is required to fully grasp the implications of using this language as a base for future human-computer relations. It is especially interesting to think of a live coding language that makes use of *difrasismos* as is a common practice in *Macehualcopa*.

With all these lessons in mind, Rolando and I started working on *Temazcal 2* once the Factory Media Centre notified us that we were recipients of their artist residency program &Now.[9] The infrastructure provided by McMaster University as part of my PhD program gave us sufficient resources to focus almost exclusively on the development of the artwork.

The process that led to the creation of *Temazcal 2* is filled with strange resonances between our ideas, intentions, national processes of, both, reconciliation and assimilation put in place in Canada and Mexico, and the different reception mechanisms put in place in a networked environment. Present in all these conversations are Indigenous Peoples from the Abya Yala (but in the case if this artwork: Mexico and Canada) and the historical and incommensurable debts that colonial societies owe these groups. Indigenous Studies and Indigenous knowledges are highly contextual and often rest on the concrete

[9] &Now is a production residency and scholarship that helps artists to produce new works or finish pending ones.

relations between the nation-state and the Indigenous Nations encapsulated within them. Glen Sean Coulthard, a Dene scholar from Yellowknife offers his fifth thesis, *Beyond the State*, as a conclusion for his work 'Red Skin White Masks' that adumbrates this position:

> *I would venture to suggest that over the last forty years Indigenous peoples have become incredibly skilled at participating in the Canadian legal and political practices[.] In the wake of the 1969 White Paper, these practices emerged as the hegemonic approach to forging change in our political relationship with the Canadian state. We have also seen, however, that our efforts to engage these discursive and institutional spaces to secure recognition of our rights have not only failed, but have instead served to subtly reproduce the forms of racist, sexist, economic, and political configurations of power that we initially sought, through our engagements and negotiations with the state, to challenge (2014, p. 179).*

The critical theory in this chapter, while only partially engaged with Indigenous knowledge and struggles, seeks to supplement Coulthard's fifth thesis and Yásnaya Gil's notion of an 'us without state' (2018) by breaking the dialectics between Indigenous Nationhood and the state. Unfortunately, this needs to be interpreted in a perhaps bleaker light as presented by Gil. A guiding question for me to write this chapter and for the reader to orient themselves has been: What will happen when structures of power and processes of dispossession engendered as neo-feudal, transnational stacks finally absorb sufficient functions of the state and become capable of coordinating an offensive against all Indigenous people without the viscosity of the states encapsulating them?

Under this light I am imagining a broad communicative space incommensurable to paradigms such as *mestizaje* or settler colonialism, both locked into state logics. This artwork, rather than focusing

exclusively on Indigenous people and their knowledge, is precisely about the necessary subjects emerging from processes of coevalness (Loingsigh, 2019) as is immigration, as well as transnational ensemble-making and networked music allowed by the internet. The historical debts owed to Indigenous Peoples by colonial societies will not be resolved in a collegial conversation between whites and Indigenous in institutional settings. Nor by the strong dialectics of *mestizaje* promoted by the state, which ultimately favour Western rationality. As I will argue in the next section, the key lies in the continuities between land and labour struggles, or perhaps generally, in the way people can face the current *multidimensional* crisis.

The Conversations to Mind

Temazcal 2 reveals, particularly through Rolando's family history, a form of subjectivity that, while interplaying with them, cannot be properly described by the framework of *mestizaje* common to Mexican intellectual discourse, nor by the binary opposition of Indigenous and settler figures that dominate Anglo-American intellectual conversations. These conversations have made their way into certain Mexican intellectual niches through various means, including the imperial relations between Anglo-America and Latin America, the academic hegemonic power of Anglo-American universities and their global influence, or the communication processes enabled by networked environments (like the one presented in this chapter), or any combination of these factors.

Temazcal 2 falls short of grasping the gendered dimensions of *temazcal* technology. Federici's work highlights the role of women's work and the way in which capitalism changed the gendered relationships

of family, housework and social reproduction (Federici, 2004, 2018). This is relevant to this artwork given the gendered nature of the *temazcal* technology. As Federici demonstrates:

> *With the persecution of the folk healer, women were expropriated from a patrimony of empirical knowledge, regarding herbs and healing remedies, that they had accumulated and transmitted from generation to generation, its loss paving the way for a new form of enclosure. This was the rise of professional medicine, which erected in front of the "lower classes" a wall of unchallengeable scientific knowledge, unaffordable and alien, despite its curative pretenses (Federici, 2004, p. 201).*

With Federici's Marxist-feminist critique in mind, an effort is in place in *Temazcal 2* to foreground Rolando's female family members. Nevertheless, my analysis remains to fall short on such dimension. Moreover, beyond the gendered dimension here signaled, the epistemic one appears also as an interesting theme to explore. Art practices that remain in the intersection of art and science relate uncomfortably with forms of knowledge as the one here presented. On the one hand, there is a process of appropriation and re-signification; on the other, there is a profound dismissal.

The figures of the Indigenous, the settler, and the *mestizo* mentioned here come into play when rethinking the significance and scope of *Temazcal 2* in the specific context where it was performed: a networked environment localised simultaneously in Mexico City and Hamilton (Canada). I will argue how the subjectivity anticipated in *Temazcal 2* emerges from conversations where these three concepts, relevant to the context of the work, express an unresolvable tension.

Tense Figures: Indigenous and Settler

According to Vizenor (1999), the representation of the 'Indian' in the Anglosphere celebrates an absence rather than marking the presence of Indigenous subjects by perpetually producing simulated copies of 'the tribal.' In the media, representations of Indigenous Peoples are overdetermined by stereotypes and negative portrayals of culture often bound to white people's appropriation of Indigenous expressions. In Vizenor's *Manifest Manners*, there can be no 'authentic' or essential truth of what it means to be 'Indian' but misreadings, misrepresentations, mistranslations and misconstructions that can either deepen exploitation and expropriation of Indigenous Peoples or are capable of building Indigenous Sovereignty and self-determination. So, for Vizenor, Indigeneity is an extremely dynamic and de-essentialised category.

For Yásnaya Gil the category of Indigenous is political (A. Gil, 2018). This means that beyond any cultural differences that might exist between the Mexican *mestizo* population and Indigenous Peoples (or, in any case, differences between different Indigenous nations) are secondary to what is most relevant: the self-determination of people. For example, the first (and only) Indigenous president of Mexico cannot be understood as an Indigenous political agent since the ideological reality he built was a form of positivist, capitalist liberalism in detriment of most Indigenous Peoples and the start of the narrative of *mestizaje* as a whitening project.

Both perspectives are contrasting; however, both share a de-essentialised conception of the category of Indigenous. On the one hand, Vizenor argues that Indigenous Peoples are free to understand the term Indigenous in any way they choose. On the other hand, Gil makes it clear that the political project of self-determination and community building (and preserving) is what makes such a category relevant. I want to

clarify once more that the concept of Indigenous as understood by Vizenor and Gil are not what is being contested by *Temazcal 2* but the relationship that people like Rolando and me have with the Indigenous category that cannot be reduced to *mestizaje* or settler colonialism.

Contrastingly, the settler is an identity that "mirrors the construction of 'Indigenous' in contemporary terms (Lowman & Barker, 2015, p. 2)." These two concepts, settler and Indigenous, have in common a connection to the land. However, this connection, for settlers, has been forged "through violence and displacement (Lowman & Barker, 2015, p. 3)." The term settler (and its relationship with the term Indigenous) is strategically significant as it enables people to highlight the dynamics of dispossession and power in Canada, a nation-state understood within the framework of settler colonialism. Settler colonialism should be viewed as a form of colonialism based on the occupation of land by settlers displacing Indigenous populations.

There have been attempts to apply the framework described here to Latin America (Wolfe, 2006); however, it remains unclear how it can be translated to that context. According to Castellanos (2017, p. 777), settler colonialism "is a slippery concept to apply to Latin America where nation-building projects have framed criollización/creolization as 'an indigenizing process.' We are left with the quandary of debating who is a settler." The term has met resistance, partly due to a translation problem – settler colonialism as "*colonialismo de colonos*" appears as a redundant concept. However, such redundancy goes beyond a translation issue and highlights a series of difficult dichotomies.

In this section, I argue that the binary understanding of dispossession, whether through land or labour, cannot be clearly partitioned to develop distinct theoretical frameworks. Furthermore, "[t]he emphasis on binaries risks reproducing a monolithic, self-contained theory of settler colonialism lacking historical

and relational specificity, the very project initially challenged by Patrick Wolfe (Castellanos, 2017, p. 778)". The question here is, in what ways does partitioning colonialism in such specific manners advance a more complete understanding of the field?

The answer given by Speed (2017) appears as a critique to Wolfe's tendencies towards binaries and an amendment of Latin-American blind spots regarding structures of dominance and dispossession identified as settler-colonial. Speed also identifies a gap in Indigenous studies that could be compensated by 'southern' scholarship:

> *Above, I argued that there is a dual theoretical gap that coincides with a North–South divide in indigenous studies, one in which northern theorizations of the settler state have not grappled fully with neoliberal capitalism, and southern theories of the neoliberal state fail to recognize the significance of settler logics that structure the conditions of state formation, including in its current neoliberal iteration. In this essay, there was space only to argue the problematics of one side of that divide, in an effort to open a path for considering the settler structures of Latin America. The significance of neoliberalism in the Native north will have to remain for a later essay (Speed, 2017, pp. 788-789).*

When 'northern' intellectuals find the space and time to discuss capitalism, I believe they might find other redundancies between Latin-American and Anglosphere frameworks that often form a close articulation between Marxism and decolonial (or Indigenous) thought. Furthermore, in 'southern' settings, it would not be uncommon to find Indigenous intellectuals entirely rejecting the framework of *mestizaje* while upholding epistemologies closer to (but not quite like) settler colonialism. All partitions, either Wolfe's land/labour or Speed's North/South should be contested.

There is a necessity to integrate and seriously consider settler colonial critiques of Latin American frameworks. Specifically, the positive construction of the Indigenous figure cannot be reduced to a matter of identity for Latin Americans. Indeed, emerging scholarship in Latin America already systematically produces this critique, so there is a need to incorporate settler colonialism into these conversations. Nevertheless, the image of the settler risks creating a new form of dispossession for those who cannot be considered Indigenous but have seldom benefited from (settler) colonialism. The settler colonial framework may benefit white people (the closest concept to settler in this context) in places like Mexico by erasing a majoritarian portion of the population that self-identifies within *mestizaje* and exclusively discussing indigeneity and, by extension, settlers. In the south, the Indigenous/settler binary cannot stand as it is usually presented.

I have the impression that the binary grammar of 'settler' and 'Indigenous' is more readily and easily applied in the north than in the south, even if there are complications with its application on both sides of that divide. So, in a networked and globalised world, where Anglo-American and Latin American contexts can be juxtaposed as we have done for *Temazcal 2*, what would be the implications of adopting a grammar that fits with the global north but remains 'slippery' for the global south? Drawing from Chakraborty (2000), is there any possibility that the Mexican context remains 'in the waiting room of history,' while the conceptually whole 'Canadian' modernity is something to be aspired by the incomplete 'Mexican' modernity? Is it problematic to frame Indigenous Peoples in the geographic north as 'global north' and mixed populations in the south as settlers? As noted by Castellanos, there is a need for a communicative space between scholars from the south and the north to discuss such matters; whether this conceptual space engages with the settler colonial complex or another kind of conceptual construct

(just to name another one: Zapatista philosophy) remains to be explored. *Temazcal 2* precisely attempts to override forms of transnational communication that appear rooted in domination and control.

Whititude, *Ethos Barroco*, and *Mestizaje*

In *Temazcal 2* I associate the script I wrote as a pseudo-essay to Temazcal 1's reification of the Mexican state, and its intellectuals, ideological figure of *mestizaje*. I invoke Bolívar Echeverría and Federico Navarrete's conceptions of race to critique one of Mexico's most performed electroacoustic music pieces.

Bolívar Echeverría has developed the concept of *blanquitud* (a word composed by *blanco* (white) and the sufix *-tud* (which implies abundancy of) as a form of capitalist orientation. I will re-constitute this concept into the English language as a portmanteau of whiteness and attitude: whititude. Such an orientation favours certain (Protestant) capitalist rationality and forms of knowledge:

> *Whititude is all the set of visible features that come with productivity, from the clean and ordered body's physical appearance and its environment to the quality of language, the discrete positivity of the attitude and the gaze and the restrain and composure of the gestures and movements (Echeverría, 2010, p. 59)*[10].

Thus, whititude should not be understood (only) as whiteness in a racial or phenotypic sense but as a realisation of capitalist ethics, where people willingly assimilate into the capitalist way of life no matter their pigmentation. Thus, whititude allows us to associate processes of racial oppression with that of

[10] My translation, original in Spanish: [La blanquitud] es todo el conjunto de rasgos visibles que acompañan a la productividad, desde la apariencia física de su cuerpo y su entorno, limpia y ordenada, hasta la propiedad de su lenguaje, la positividad discreta de su actitud y su mirada y la mesura y compostura de sus gestos y movimientos

capitalist accumulation, which is a more precise way of understanding racialisation. As racial capitalism (Robinson, 2019) illuminates all capitalism as a process of racialisation, Echeverría's concept would imply how resilience under capitalism can be observed as a whitening process. Furthermore, Echeverría's conception of race is profoundly bound to class. Similarly, land is tied to labour, and decolonial thinking is inseparable to class struggle.

The '*ethos barroco'* is a concept proposed by Echeverría (1998) as a mechanism of resistance, allowing Indigenous urban populations of the XVI century to appropriate European cultural codes as survival strategies following the emergence of the colonial order in places like Mexico. Within *ethos barroco* lies a modernity that differs from capitalism. The *ethos barroco* can be contrasted with the *mestizaje* project.

Navarrete understands whititude from a different perspective; he extends the concept to all forms of rationalistic and progressive European modernity projects. He associates the concept of whititude with the state project of *mestizaje* quite convincingly. The *mestizo* subject is associated with eugenic-oriented *raza cósmica* (Vasconcelos, 1948) and has been a key figure in understanding the Mexican post-revolutionary state. *Mestizaje* describes a subject that is a mix of white and Indigenous Peoples, and it has been imposed as dominant in the post-revolutionary hegemonic arrangement bound to the Mexican state.

In *México Racista* (2016), Federico Navarrete describes the various ways whiteness is perceived and weaponised in Mexico to maintain a racialised structure of power, with its roots in the caste system of the *era virreinal*. Similarly, he disarticulates mestizaje, revealing it as a modern invention originating in 19th century political processes, functioning more as an ideological figure than a social reality in the country. This book debunks the persistent myth of the Mexican state as non-racist; after all, the first Mexican

president was Afro-American, one of his most relevant reformists was Indigenous and the Mexican revolution pushed upwards an immense amount of people of colour into positions of power and privilege. We could interpret these examples as manifestations of whititude in Mexican history.

However, Navarrete's denunciation of racism, more than building a radical (other) modernity as Echeverría has done, seems to be a call to reform our liberal landscape: (a) reinvent media representations: advertisements should reflect the demography of the country and comedy should not be made at the expense of minorities; and (b) reinvent citizenship and democratic representation capable of fulfilling the 'broken promises' of liberalism. These calls are rooted in the USA's liberal strategies, as explicitly stated by the author, which confirms that Mexico indeed belongs to the waiting room of history, aspiring to a complete modernity similar to that of the American empire before Trump.

This racial ambiguity, echoing Echeverría's whititude, present in Navarrete's denunciation of Mexican racism is useful for understanding how different projects of *mestizaje* have evolved in Mexico. Navarrete has constructed the project of *mestizaje* as a whitening project by racially and culturally propagating a becoming-white of Indigenous and Afro-Mexican people. This is more than a project towards capitalist rationality where whiteness acquires a different connotation. Considering the critique based on whititude and Navarrete's elaboration on racism via American liberalism, Navarrete's categories appear to disarticulate the relationships between whiteness and capitalist rationality by focusing on representation without considering any forms of critique of the political economy and matters of distribution.

For example, when discussing the forced disappearance and murder of the 43 teachers from Ayotzinapa that initiated Mexico's change in hegemony around 2014, Navarrete provides a complete picture of

representation, taking into account factors such as pigmentation, income, geography, and Indigeneity but fails to mention the political orientation of the militant teachers:

> *Indeed, the 43 victims of this crime against humanity belonged to the most marginalized and ignored sectors of Mexican society. They were excluded because of their regional origin in Guerrero, one of the country's poorest and most violent states; because of their poverty; because at least 11 of them spoke a native language other than Spanish (as Mixe linguist Yásnaya Aguilar, who provided this information, would put it); because of their physical appearance and skin color—traits rarely represented in the media, advertising, or dominant social narratives, and almost always associated, due to ingrained racism, with poverty and so-called illegitimate forms of political and social behavior. Finally, they were excluded because they were engaged in political activities that many commentators deemed subversive and therefore illegal and worthy of persecution (Navarrete Linares, 2016, p. 181).*[11]

He then proceeds to trace the associations between these teachers and other relevant historical figures in Mexican history. As he weaves his narrative, he transforms the meaning of these historic figures. He reconstitutes them in the framework of 'political participation and struggle for citizen's rights' rather than explicitly invoking them as what they are: militant Marxist revolutionaries. Navarrete, using the grammar

[11] Automatic translation from: En efecto, las 43 víctimas de ese crimen de lesa humanidad pertenecen a los sectores más marginados e ignorados de la sociedad mexicana: son excluidos por su origen regional, en Guerrero, uno de los estados más pobres y violentos del país; lo son por su pobreza; lo son por el hecho de que al menos 11 de ellos son hablantes de una lengua materna distinta al español (como diría la lingüista mixe Yásnaya Aguilar, quien me dio este dato); lo son también por su aspecto físico y su color de piel, casi nunca presentes en los medios de comunicación, la publicidad y las representaciones dominantes de nuestra sociedad, casi siempre asociados por nuestro racismo a la miseria y a las formas no válidas de comportamiento político y social; lo son, finalmente, porque estaban asociados a actividades políticas que muchos comentaristas califican de subversivas y, por lo tanto, de ilegales y dignas de persecución.

of liberal tradition, unwillingly or not, builds a neoliberal progressive scholarship by purifying Mexican history and Echeverría's framework from its radical components.

Yásnaya Gil and Federico Navarrete are exceptionally critical of the *mestizo* paradigm. These scholars point to individuals or state apparatuses that appropriate and weaponise Indigenous imagery and identity for their own benefit and/or people outside Indigenous communities; to build narratives of national identity; and to build subjectivities useful for capitalism and the nation-state. However, as explicitly mentioned by Gil (2020, p. 126) and suggested by Navarrete's more recent research interests, they acknowledge a continuity between Indigenous Peoples, Afro-Mexicans and others that are yet to be theorised. Gil accompanies her critique of *mestizaje* with a positive construction of the concept of Indigenous subjectivity and a form of social protection predicated on *Mixe* political structures incommensurable with European modernity. Since non-*Mixe* people cannot be included in such political structures, the question for people who do not identify as Indigenous or participate in Indigenous communities remains untouched.

Navarrete's anti-racist critique of *mestizaje* is supported by a form of liberalism that resonates with the social context of the USA. While *México Racista* may not be an academic text, it considers issues of readability and accessibility for more general audiences. However, in this book, his critique of Mexican racism hinges on purifying Mexican history and Latin American intellectual traditions of their most radical aspects, such as Marxism and the work of Echeverría. This is the pattern that intellectuals with a foot in the south and a foot in the north need to recognise: the specific pattern that *Temazcal 2* is intentionally avoiding is critiquing *mestizaje* from a settler-colonial perspective, which is generally practiced in the

north and sanitised of the most radical aspects of critical theory so it may be commensurable with progressive neoliberalism.

Triple Movements and Double Consciousness

As I have been building so far, a conversation between decolonial theory, Indigenous studies, Marxism, and Marxist-feminism frameworks might illuminate some ways in which the process of capitalism, on a global scale, interplays with the singularities described by Indigenous peoples, *mestizos,* and settlers.

Before sharply turning to Marxist analysis, it is important to understand that class is neither an identity nor a vector in an identity grid that intersects with other identities. Class analysis does not reduce different struggles or forms of oppression into a single kind that predetermines all forms of struggle. However, it has a status that should be acknowledged:

> *We need to be reminded why Marxism ascribes a determinative primacy to class struggle. It is not because class is the only form of oppression or even the most frequent, consistent, or violent source of social conflict, but rather because its terrain is the social organization of production which creates the material conditions of existence itself. The first principle of historical materialism is not class or class struggle, but the organization of material life and social reproduction. Class enters the picture when access to the conditions of existence and to the means of appropriation are organized in class ways, that is, when some people are systematically compelled by differential access to the means of production or appropriation to transfer surplus labour to others (Meiksins Wood, 1995, p. 108).*

Thus, class is necessary to understand forms of distribution of wealth that create the conditions of 'existence itself' under capitalism. To make commensurable issues of distribution and political representation and agency, I will fundament my writings in the theory of the triple movement (Fraser, 2020, p. 320). According to Fraser, the triple movement is shaped as follows:

1) People resist extreme forms of marketisation that, on the one hand, seek to determine all aspects of life and, on the other, deteriorate all forms of social protection as they prioritise financial gains over the needs for people to live their lives.
2) As people resist marketisation, they attempt to re-embed the economy (everything that supports life) into society. This may take the form of institutional support or networks of mutual aid. However, the process of re-embedding the economy within society is not intrinsically good.
3) To avoid oppressive forms of social protection people put in place emancipatory mediations capable of filtering-out returns to marketisation and pathways to oppressive social protection.

For instance, second wave feminist struggles to recognise unwaged housework are transformed into equal access to the labour market for women under neoliberalism, missing the underlying point of emancipatory movements: the labour market, that relies indirectly in unpaid housework, is intrinsically exploitative. This form of exploitation erodes social reproduction by, instead of simply allowing women in the workplace, allowing market forces to undermine the family wage. This misidentification, the belief that inclusion of women in the labour market is liberating, produces a new misidentification: women's inclusion in the work force provokes wage stagnation. Such situation allows oppressive political movements to reclaim the nuclear family, and the role of woman solely as care-givers, as a solution to social problems engendered by neoliberal austerity. The triple movement helps us trace back the chain of

misidentifications to correctly identify both marketisation – the commodification and exploitation of women via waged labour – and oppressive social protection – by limiting women to the household and unpaid care work – as the double-issue to be tackled.

A movement away from class determinism requires a theorisation and serious consideration of emancipatory frameworks like decolonial studies, Indigenous studies, cultural studies, feminism, queer theory, antifascism, etc. Marxism without such forces risks becoming a form of oppressive social protection; inversely perhaps, emancipatory forces risk capture from marketisation without a critique of political economy, which ultimately leads to radical forms of oppression. The decolonial critique of *mestizaje* and the settler colonial complex by either Mexican or northern Indigenous Peoples, as already discussed, remains ambiguously indifferent to issues of capital accumulation; similarly, the framework of *mestizaje* tends to authoritarian regimes as the one party's 'perfect dictatorship' of the Mexican twentieth century shows. With this in mind, it is necessary to offer a brief Marxist analysis that can supplement the categories here discussed.

The category of Indigenous in Mexico can be explained to resist proletarianisation of a surplus population. Indigenous people, from a Marxist analysis, can be considered a reserve army of workers for capital that keeps the price of labour low (Forssell Méndez, 2020). This, of course, does not mean that Indigenous identity is reducible only to its relationship to capitalism (as a surplus population), but it is useful to deploy such understanding strategically. In other words, Indigenous populations in Mexico remain outside the productive processes associated with capitalism (selling labour in the market) but not passively or outside the scope of market forces as often is the narrative. They are actively kept outside such processes to keep wages down and pressure workers into low paying jobs. In that way, the well-

known Mexican cheap and overworked labour force was produced. The process of *mestizaje* is precisely the transformation of this form of surplus population into productive labour for capitalism. The capture of Indigenous people into cycles of exploitation, is never just a process of proletarianisation but also that of accumulation by dispossession (Harvey, 2004) and/or ongoing original accumulation (Luxemburg, 2003; Whitener, 2021). This kind of dispossession is described by David Harvey as a process which "entailed taking land, say, enclosing it, and expelling a resident population to create a landless proletariat, and then releasing the land into the privatized mainstream of capital accumulation" (Harvey, 2005, p. 149).

Coulthard argues:

> *it is reasonable to conclude that disciplining Indigenous life to the cold rationality of market principles will remain on state and industry's agenda for some time to follow. In this respect Marx's thesis still stands. What I want to point out, rather, is that when related back to the primitive accumulation thesis it appears that the history and experience of dispossession, not proletarianization, has been the dominant background structure shaping the character of the historical relationship between Indigenous peoples and the Canadian state (2014b, p. 13)*

What are we to make of the tension between the Canadian and the Mexican context? It appears that colonialism responds to concrete local conditions that makes it very difficult to draw parallels. From such an observation, it also appears relevant to overcome the nation-state as an analytic limit. For this reason, the Marxist analysis of our current social totality becomes key. As it will become clear in the last part of this section, the current historic moment – moving away from neoliberalism – is characterised by an exalted disarticulation of working conditions and labour movements, and market forces striving to dispossess people beyond exploitation schemes. This is why it is fundamental to reintegrate the

struggles of land and labour. As narrated in *Temazcal 2*, Rolando's family is displaced from the *Mixteca Alta* into Mexico City. This process of proletarianisation attempts to turn them into cheap labour and dispossesses them of land and the means for social reproduction. This happened in the era of state-managed capitalism, when cheap labour and land were needed to build capitalist extraction in the country. Two generations later, a different form of capitalism triggers a new kind of movement discussed in *Temazcal 2*: from Mexico City back to the *Mixteca Alta*.

Mexico City is a tense place in the era of crises provoked by uncontrolled financialisation; a site of turbulence in which settlers, *mestizaje* and Indigenous realities clash. It is also one major market for real estate; this is especially true in relationships with Anglo-America and Europe. There are figures that can be related to the settler that could be theorised here: the whitexican, the digital nomad, the creative class, the expat, the tourist, etc. These people, regardless of their pigmentation, nationality, or culture build a form of whititude: an orientation towards capitalist lives, a capitalist ethos proper of Mexico City. More exactly, their presence and habits of consumption in a city like Mexico have made life unlivable for anybody else. Housing is unaffordable, public spaces and institutions exist now to serve the wealthy, and culture is changing (becoming further 'cosmopolitan', AKA white and European), and the Indigenous Peoples have been transformed into an object of admiration but also a subject of contempt. As the global north becomes unaffordable for their disappearing middle classes, places like Mexico City are seen as a viable alternative that allows them not to alter their lifestyle and class status. It is also important to acknowledge that many of the people moving from the global north to Mexico City are themselves dispossessed in their own countries by financialisation and, more recently, by fascism.

Temazcal 2 has allowed me to radically transform the concept of settler-colonialism and understand Mexico City's colonisation within its scope. First, the people that were pushed out, through marketisation, of the global north and relocated to Mexico City, are building a form of colonialism that can be explained as settler-colonialism. Second, the influx of people from the global north to Mexico City is building a progressive-neoliberal block – and a North American fascist block – that is profoundly transforming the political landscape of the city and, with it, its history and class relations. This means that forms of oppressive social protection may emerge from the erosion provoked by marketisation. The availability of cheap housing for Anglo Americans and Europeans is already creating tensions between these settlers and all kind of Mexican citizens. How can the government of the city respond to any hostility between the new settlers and old citizens? I can imagine the state, upholding the *mestizaje* narrative, might respond to the situation by furthering the ongoing whitening process. In doing so, a destructive marriage between settler-colonialism and *mestizaje* becomes imminent. Third, with this wave of whititude and by this form of settler-colonialism, new emancipatory forces are spread and relocated as well. Within the people expelled by the global north it is possible to trace relevant forms of solidarity capable of making Mexico City's experience global. From this knowledge, questions emerge: what is the position of the people displaced from rural Oaxaca 40 years ago by state-managed capitalist forces and now displaced from Mexico City by financialised capitalism? What is the role of graduate students in the global north that still call Mexico City their home?

As I mentioned, a figure is brought to the foreground in *Temazcal 2* that remains elusive and nebulous still, but some of its characteristics have been sketched throughout this section of the chapter. Like the figure of the Indigenous as defined by Vizenor and Gil, this figure should be constructed as a set of

misreadings, misrepresentations, and mistranslations of *mestizaje* capable of building a political emancipatory project. A political project built with complex coevalness properties, capable of tracing solidarities among different times and spaces. Here, it is relevant to try to understand networked environments as a crucial component to reveal the world-creating processes of capitalism, where the Canadian context can only exist insofar as the Mexican context does and vice versa. The most vital aspect here revealed is to acknowledge Indigenous Peoples' right to self-determination from all geographies. It remains unclear what the precise language to describe the figure revealed in *Temazcal 2* is, but the notion of a double consciousness can be helpful when put into words.

Double consciousness is a concept that originates from the black radical tradition. Du Bois (1997) coined this term to understand the lives of African Americans as both Black people and citizens of the USA. The term refers to an inner conflict derived from this fragmentation. Similarly, Fanon (2008) invokes a white mask/black skin dichotomy to understand the psychic condition of Black people in Martinique engaged within colonial relations with France. Fanon attempts to describe different ways Black people engage with white, Western civilization. Often, Black people would mis-recognise themselves as Black until meeting a situation where their racialisation is salient. All of this resonates with the condition of *mestizaje*. Thus, mestizaje is a process of double consciousness reminiscent to what is described in the black radical tradition. However, the double consciousness required in this context has a slightly different connotation that builds on top of Du Bois's concept. The meaning of double consciousness I propose is an articulation of class consciousness with the generative forms of situated knowledge and racialisation, as shown as alive in Rolando's family. This kind of double articulation has the capacity to uphold the triple movement's advantageous critical distance from marketisation and oppressive social

protection. Hence, when I describe a 'double consciousness' in relation to *mestizaje*, I mean to signal a subjectivity that is capable of embodying something very specific: social protection mediated by emancipation. Namely, this concept does not only allow racialised and proletarianised people to deploy different modes of social relations whenever necessary, but also to recognise the tensions and contradiction emerging from the different ways of being in the world.

So far, the *chic* critique of Marxist teleology has remained unmentioned in this chapter. To develop my double consciousness concept, I will need to tackle it now. Marx, in several moments of his work shows admiration towards the technical progress of capitalist society and the bourgeoisie. As the argument goes, capitalism clears the way from 'pre-modern' conditions of production to create the basis for socialism. In 'pre-capitalist' modes of production, 'human being appears as the aim of production [in contrast with the] modern world, where production appears as the aim of mankind [sic] and wealth as the aim of production (Marx & trans. Nicolaus, 1993, p. 488)." Capitalism's destruction of modes of production other than itself is celebrated by Marx as a necessary step towards socialism. This productive destruction is characterised by an inherent emptying out of subjectivity. For example, artistic movements orbiting around their love for technology can be understood as machines that output void subjects.

Nevertheless, Marx is a product of a fragmented and double social moment: romanticism and enlightenment. As such, contradictory positions become clear in his work:

> *In bourgeois economics – and in the epoch of production to which it corresponds – this complete working-out of the human content appears as a complete emptying-out, this universal objectification as total alienation, and the tearing-down of all limited, one-sided aims as sacrifice of the human end-in-itself to an entirely external end. This is why the childish world of antiquity*

> *appears on one side as loftier. On the other side, it really is loftier in all matters were closed shapes, forms and given limits are sought for. It is satisfaction from a limited standpoint; while the modern gives no satisfaction; or where it appears satisfied with itself, it is vulgar (Harvey, 2023, pp. 223-24; Marx & trans. Nicolaus, 1993, p. 488).*

In this fragment, Marx contradicts the teleological point described earlier by denoting the void subject from the movement towards socialism as vulgar. This is a useful way to understand *mestizaje* and to perhaps further understand the liberal critiques of *mestizaje* as well.

Emerging from these ideas, it is possible to state that my role in the process of *Temazcal 2* is to creatively destroy *Temazcal 1* and to establish the basis for a baroque form of communism that articulates the racialised and Marxist consciousness. From this process, a subject appears in Rolando's narration. The interstitial subject traced in *Temazcal 2* is capable of multi-perspective thinking and acting, capable of contradictions, and adjusted to navigate crises that have often been their origin.

Rolando's mother moved back to where her family came from by buying land and slowly building a retirement/family home. This family home is neither a return to the 'primordial home' nor a movement outside of capitalism, but it acknowledges the effects in people that these historic transformations had. These changes involve both processes of whititude and baroqueness that build a contradictory position, always tense and unsettled. The memory of a life that is not fully captured by systems of oppression is ignited but constantly struggling to remain present. As seen at the end of *Temazcal 2*: the memory of the ancestors is always lit in the night by a fragile flame on the verge of darkness but refusing to fade out.

In the following section, I will describe the artwork as standing one year and a half after its completion and presentation to the audience. The ideas expressed in this section will now describe the artistic practice that formed them. The dialectic relationship between these ideas and the practice allowed me to claim the baroque subjectivity (that is not empty at all) suggested here.

Temazcal 2: Archive and Documentary Practice

Preliminary Notes

The description of the work relies on two sources: the online repository that contains all the assets of the artwork and the video documentation I recorded on the day of its premiere, both attached to this dissertation. I argue that the artwork can be understood as a non-ephemeral interactive networked art piece, which one of its main consequences is an on-the-fly documentary. This means that people can use the repository to reconstruct it, manipulate it, reuse the material, and more. Furthermore, the precise meaning of the densely signified documentary we assemble with live coding can only be accessed through manipulation of the materials in the internet repository. Perhaps Estuary serves as a favorable interface for this artwork, but many others can be used as well.

There are two main reasons for presenting the documentary in this way. Firstly, *Temazcal 2* explores a speculated return to orality, which translates in our minds to a non-fixed form of knowledge production. This idea is inspired by César's elaboration on archive: "The notion of an archive does not refer to a physical space with specific records but rather to complex cultural and philosophical entanglements that our bodies receive and may later respond to through reasoning and stimuli" (Juárez Joyner, 2021, p. 101). In other words, the knowledge and experiences raised by this documentary should inhabit and traverse

bodies, rather than being stockpiled in some digital medium. At least, this was our initial intention, which has proven very difficult to realise in a world that relies on stockpiled information as evidence of existence. Secondly, *Temazcal 2* responds to Malovich's critique of interactive art (Manovich, 2009), where ephemeral interaction is interpreted as manipulation of audiences by creating the illusion that insignificant choices constitute freedom. By opening various pathways for people to access the materials of the artwork without providing a specific tailor-made interface for interaction, we are reasserting an interactive art aligned with a substantial form of agency: use the materials, explore them, create meaning with them, undo it, unfold it, erase it, etc. The core of these ideas can be found in the last chapter of the recently published live coding manual (2022), which defines its relevance as "[...] not so much [as] a mode d'emploi or user's manual on or about live coding [but] as an attempt to explore live coding as a user's manual or guide" (Blackwell et al., 2022, p. 240).

This layer of the artwork presents many challenging conditions of availability. Thus, I believe that this layer will probably be more accessible and advantageous to other live coding practitioners and other people involved in programming and networked art. However, the broader audience will likely not take full advantage of the freedom provided by this documentary. Nevertheless, it will still be available as video documentation on YouTube, recorded on the day of its first public performance. Therefore, I will refer primarily to the video documentation and, to a lesser extent, the repository to describe the artwork in this chapter.

This performance was an arrangement of the repository established between Rolando and me with ongoing feedback and input from Diego (the music code performer). A good metaphor for understanding the relationship between the repository and the performance is the night sky: the night sky and its visible

stars represent the repository, while the traced constellations symbolise the performance. The repository's materials can be categorised into various ideas we aim to convey: some images refer to Rolando's family; others relate to the site of a form of knowledge (the *temazcal* artifact) explored in the documentary; others include carved images on musical instruments that have been digitally reconstructed, along with schemes of constellations. A specific category of materials consists of maraca samples recorded to serve as the main instrument of this artwork but can also be utilised generally as an instrument for any type of performance or music session in Estuary, along with many other digital instruments that process samples. There is a collection of code snippets that form a composition of repository material into programs that can be evaluated in Estuary. Everything related to the creation of *Temazcal 2* exists within this repository. Additionally, the materials generated and used in *In Xiuh Ce Amatl* are also contained in it. In the performance referenced here, we extensively use the *huehuetl* samples recorded by César Juárez. Another set of items in the repository consists of the JSoLangs, which enable us to have this layer of meaning written in *Macehualcopa,* where some code snippets in general languages used in Estuary (specifically MiniTidal, Punctual, TimeNot, and CineCer0) are incorporated within commands in the Mesoamerican language. The repository also serves as a sort of 'historical' record of the project's development. One can observe a snippet of code that is later transformed into a language, which is transformed into a configuration on Estuary. Two scripts are the most significant materials for articulating the performance. One script is textual, developed by me; it constitutes a pseudo-academic analysis of the work *Temazcal 1*. The other script is aural, consisting of a series of voice-over recordings made by Rolando, narrating the family's relationship with the *temazcal*.

Temazcal 2 is a networked artwork because it exists materially on the internet and because the performances and its reception do not rely on local audiences. The premiere happened simultaneously in three locations: at the Networked Imagination Laboratory (NIL), McMaster University in Hamilton, Ontario, at the *Intercuraduria* Gallery in Mexico City, and it was also transmitted on the internet by the Factory Media Centre. This is a key aspect of the work as it forces us to consider concretely an audience that is not limited or determined by local conditions. Thus, we had to speak to very different audiences simultaneously, and, therefore, we could understand the impact of our work with a global totality in mind. This aspect of the work will be analysed in-depth at later stages of this chapter. For now, let me share some particularities regarding the conditions of this triple event premiere. The event in Mexico City was marked by a profound lack of care and disorganization from the curatorial staff of the space that did not support the event, as no profit was anticipated from such support. In general, the conditions did not allow for a broad reception. Nevertheless, people close to Rolando, Diego, and I attended, and we personally received feedback on the work. Rolando's mother was present at the performance in Mexico City, which I consider a very special reception. The NIL had better conditions; fundamentally, the global north's privileged access to technology like good internet connections, screens, computers, or speakers provided almost ideal conditions for the performance, quite contrastingly with Mexico City's experience. FMC, a non-for-profit center and artist-run new media community arts organization that provided an almost ideal platform to socialise the work.

One very interesting moment of this shared event was the moment for the land acknowledgment, which has become customary in Canadian institutional spaces in the context of Truth and Reconciliation and as a way to understand aspects of Canadian history like broken treatise and the horror of residential

schools. The impression shared amongst Rolando, Diego, and I, is that land acknowledgments generate a lot of cognitive dissonance when the context is removed from the reception of it. Some fundamental questions emerged from this gesture: what does it mean to acknowledge the land where an event is happening when half the audience has no relationship with such land and, more importantly, is unable to differentiate between the institutional context in which the land acknowledgment takes place, and the knowledge that Canadian institutions work together with the Mexican government and organised crime to dispossess people in Mexico in the most violent forms imaginable? How is land acknowledgment in Canada related to land dispossession in Mexico? This is further complicated by the themes explored in the documentary that refer to ongoing land and labor dispossession in the context of globalised capitalism. In many ways, the artwork is a response to the land acknowledgment as an institutional mechanism for Indigenous recognition. Our ensemble proposes to recognise land and labour, not only the different indigenous territories encapsulated in so-called Canada, but that of all peoples marked by Canadian (and all forms of) imperialism.

Before I continue with the thorough description of the documentary as inscribed in the online repository and as unfolding as performance, one more aspect of the work needs to be addressed. This work is an instance of class struggle that takes the form of appropriation. *Temazcal 2* appropriates upwards and seeks to undo an act of misrepresentation. I am analysing the music piece of Javier Álvarez not to undermine his character or public *persona* but with the hope I can shed some light on a cultural artifact that participates in the construction of social protection that benefits a minority at the expense of the oppression of targeted minoritised people.

Lastly, *Temazcal 2* presents a world beyond the crises of care, the environment, representational politics, and the economic crisis that the current social totality we call capitalism engenders (Fraser, 2023). Additionally, I believe that *Temazcal 2*, by rigorously pushing to its last consequences the patterns, ideas, and desires identified as part of this artwork, has achieved substantial originality as an artwork that tackles the pervasive crisis of imagination bound to neoliberal subjectivity (Fisher, 2009). What follows is a thorough description of the ideas and experiences that unfold and are contained in *Temazcal 2*.

Temazcal 2

Temazcal 2 begins with a time of crisis, more precisely, a convergence of times that marks many aspects explored in the work. The first and most immediate time marker that can be observed is the dawn in the *Mixteca Alta* (a region in southern Mexico, the exact location will remain unmentioned throughout this text), received with a fire that implies a vigil and signals an act of attentive observation of the night sky. The *Mixteca Alta* is also the locus of situated knowledge central to the documentary's narrative, as I will elaborate throughout this section of the chapter. In the background, a schematic representation can be observed of constellations found on Mesoamerican musical instruments, revealing a close relationship between celestial movements, calendar notions, and the performance with percussion instruments like the *sonajas* (an instrument related to the maracas) and *huehuetl*. These instruments are a central sound and symbol explored in *Temazcal 2*. In one of the constellations traced, it is possible to read the words:

> *"Temazcal is an electroacoustic music piece for maracas and fixed media which have received a lot of attention in the anglosphere. The work is inspired in the traditional 'temazcalli', particularly the one used in the Anahuac valley in the centre of Mexico."*

These 'text-shaped' constellations are a condensed version of the script I narrate throughout the piece. However, each section of the script has been assigned a glyph, which I will explain later. Due to the dense nature of this first scene, the glyph for it does not appear. Furthermore, not all the text here is visible, and it is also possible to see that the other constellations are 'made of text' as well, but due to the font size, the meaning is impossible to decipher. This text is a script that mimics the style of stale academic writing, and its topic is a critical analysis of the work *Temazcal 1*. The first part of the script provides the context of the work by *Temazcal 1*:

> *"Temazcal (maraca soloist and electronic sounds)), 1984. London, Institute of Contemporary Arts. Work by Javier Alvarez. Original performance by Julio Toro. It has become an iconic work for the electroacoustic music tradition. Performed regularly, the work is perhaps one of the most well known pieces of music for instrumentalist and electronic sounds ever made by a Mexican composer."*

The layering of 'texts' (the juxtaposition of the constellation scheme, the glyphs, the text that analyzes *Temazcal 1*, and the digital mediation) resembles a palimpsest, where multiple texts are stacked upon one another, obfuscating the previously recorded meaning. In this case, the symbols carved into musical instruments give meaning to celestial bodies, later interpreted by scientists, which are then re-signified by musicians like Javier Álvarez and, ultimately, re-appropriated by me, a media artist. The clash between layers of meaning displayed here is intentionally obscure, as the surface of the performed documentary acts as an entry point rather than a finished work. The implication is that interested audience members can access all the materials 'abstracted' from performance conditions, either as a repository of files or as instantly available sources to be played and manipulated in Estuary – as previously noted.

The sound of the maracas marking a 13-beat metric count fades in. These sounds anticipate an important aspect of the music composition of this documentary: a reverse count from major density to minor, using numbers with a particular symbolic value in Mesoamerican culture—the '13 heavens' narrated as part of the creation myth in many *Nahua* understanding of the cosmos. Lastly, and more importantly, Rolando starts to narrate a particular time of crisis for his family that involves the passing away of his grandmother (2017), a couple of (male and female) cousins, and a close nephew (2019). Rolando finishes his intervention with the words: 'My grandmother was born in *Tlaxiaco*, on the 5th of February, 1921, and my nephew passed away on the 5th of February, 2019, in *Tultepec*, close to Mexico City.' This juxtaposition of events, the end and the beginning of lives in a difficult chronology, breaks temporal linearity and opens a space of reflection and contemplation; the space explored in this artwork.

Perhaps already evident in this densely coded scene, two main narratives will unfold throughout the documentary in their particular ways: Rolando working through the time of crisis already mentioned by activating an embodied memory, and my pseudo-essay on appropriation amid a major global and national hegemony crisis. *Temazcal 2* is highly reminiscent of polyphonic music, as the two narrative lines create a 'counterpoint' where they are apparently disconnected but somehow complementary. One could say that the only convergence point between them is the figure of the *temazcal* and the radically different meaning it acquires depending on the two representations we created.

Taking lessons from previous works by Rolando and me (*In Xiuh Ce Amatl*), we decided to differentiate the two narrative lines to reflect their differences. Rolando's line relies on pictures, voices, landscapes, family portraits, gestures, family archives, and interviews that best fit with the more embodied and

mnemonic ideas (referenced by Rolando in his script) constituting the narrative. In the context in which *Temazcal* 2 was premiered, Rolando's narrative is illegible for people who do not speak Spanish, thereby sheltering certain forms of knowledge and information that intentionally remain difficult to access.

My part relies on code, schemes, academic research, musicological analysis, music critique, score writing, composition, and other mechanisms that may appear as abstract and historic (rather than concrete and mnemonic). It also uses English, a language more transparent for any audience member regardless of their native tongue. The industrial-academic complex relies on English as a lingua franca, and as such, it naturally aligns with the pseudo-academic style I have developed for this script.

Two further layers of signification are noticeable related to language interplay: the commands given to the computer that animate and instantiate the work's components are JSoLangs (smaller, on-the-fly created languages enabled by Estuary) designed based on *Macehualcopa*, a language spoken in central Mexico, or *Macehualcopa* mixed with English and a few elements in Spanish (like the name of files, etc.). The infrastructure expressed in *Macehualcopa* is a modest continuation of the research done for *In Xiuh Ce Amatl* by Rolando and me in collaboration with César Juárez Joyner, as previously mentioned.

As the first scene fades out and a second scene visually presents Rolando's family members to the audience, we can see the cursor of the music performer (central panel, using the *Kakilistli* language) change the pattern kakillistli-1 to kakilistli-2, which changes the meter of the music from 13 beats per cycle to 12. It is also possible to see a second glyph originally found on the soon-to-be-explained Mesoamerican percussion instrument that represents a constellation. Again, we see it marked by the pseudo-academic essay in an (almost) illegible manner. The script in this scene explains what a *temazcalli* is:

> *"Temazcalli is the name given by Nahua speakers to a steam house in which ritual and medicinal ceremonies take place. Often these rituals have to do with birth and pregnancy."*

In this segment of the script, the first piece of information will later resonate with Rolando's narrative and the subsequent representation of Álvarez's music: the *temazcal* artifact is associated with birthing and medicine. Thus, it represents a gendered technology and form of knowledge.

The central aspect explored in this scene is the memories narrated by Rolando's mother, aunts, and uncle regarding their mother, Petrona (Rolando's grandmother) and her mother and their relationship with the *temazcal*. The family's recollections can be summarized in three main ideas: (1) when they were young, they witnessed how their grandmothers used to provide *'temazcal* baths' to women, who had just given birth, to fortify their bodies; (2) they remember their mother healing family and community members with her situated knowledge of the plants and herbs of the region; and (3) they remember a variety of experiences related to the *temazcal*, such as a bathing technique with blankets, steam, and herbs. The *temazcal* (apparently) concretely existed in *Tlaxiaco* (where the family lived when they were young). The austere maraca sound plays as an accompaniment to the voices telling their family's story. The metric count moves from 12 to 11, and the documentary transitions to the next scene.

The third scene is a presentation of Rolando's family nucleus. A family portrait can be seen in the background held by Clara, Rolando's mother. Simultaneously, the third glyph can be seen holding the text:

> *"In Javier Alvarez' temazcal, we can also see the central figure of the performance, the soloist evoking a shaman with its sacred sonaja (in this piece transformed into the more musical maraca).*

We can hear the cavernous womb-evoking space, the rocks crackling in the fire, the water evaporating, the smoke and steam rising… All of the elements that are often found in these kinds of ceremonies are captured by the electroacoustic sonic imagination. Quite a spectacle! (for Europeans)"

One of the reasons the script is a pseudo-academic essay is the same reason that impedes me from verifying the veracity of the analysis above: there are no commentaries by the author about the representations of the piece settings, at least not readily available online or written down in the score of the piece. The performer's role as a meaning-making figure remains explicitly undefined. I do not believe this is to obfuscate any meaning of the piece related to Mesoamerican culture. The role of this set of symbols will be explored later in this analysis. The actual meaning of the piece is the research on electroacoustic music that is so prominent in Álvarez's career.

Rolando's narration now focuses on the site of the family's *temazcal, which once stood* on a lateral terrace of the central church of the town where Rolando's mother established her retirement home. The visual components of this second section of the scene present the church and the location where this artifact used to be, according to Clara's memories. As implied, there are no traces of the *temazcal* anymore. A new aural component emerges: an arpeggio moves slowly, refreshing the ear while marking an important turn in the documentary.

The metre turns to 10, and the visual narrative focuses on the site for the new *temazcal* that Rolando and I planned to build as the original idea for this documentary. This requires further explanation. Originally, the main goal of *Temazcal 2* was to build a new *temazcal* and document the process of its construction. I travelled to Mexico, to the site we established as the place for this project and planned for a couple of

days to build this structure. The idea was to build it without relying on anybody's knowledge or labour except our own. The only indications that we had were Rolando's family memory, YouTube tutorials, and conversations with some people who described the process to us as they understood it. We gathered the materials and started processing them with some tools we borrowed. We tried to put up the bamboo stick structure and failed until finally, we had something to work with: a flimsy and sort-of inadequate structure. Then we tried to cook some clay for the bricks. After a day of trying, we finally realized that the knowledge to rebuild the *temazcal* was lost and that we would have to either rely on other people's knowledge or accept the fact that the information required is lost from Rolando's family core. In the documentary, the site of the new *temazcal* can be observed with a couple of videos attempting to build the structure. Rolando advances his narrative with an audio clip of his mother reminiscing about her mother's knowledge of healing with plants, which should be grouped as the same kind of knowledge as that of the *temazcal* artifact.

The glyph for this scene contains the text:

> *"Alvarez, on the one hand, utilises the exotic and orientalised perception of what Mexico is in the European imagination and, on the other hand becomes a key figure that brings whiteness and European progress to Mexico by consolidating the electroacoustic tradition with local flavours. In this way, Alvarez feeds the predatory and colonial culture in which academic art in the UK (and Western Europe) sustains itself and, at the same time, he advances tropes of progress pervasive in Mexican modernity. He appears as foreign and exotic to Europeans at the same time he instrumentalises his own whiteness as a sign of authority and prestige in Mexico."*

This text describes a well-known pattern of behavior related to academically sanctioned artists who study abroad. They simultaneously adopt knowledge from Eurocentric universities as canonical while misrepresenting (and capitalising on) situated forms of knowledge. The side effect of this is the reification of progress and what Chakraborty calls the “waiting room of history” (2000) in which Europeans must experience modernity first, and the developing world can only adopt it after and as fast as possible. The dependence on knowledge production from places like Mexico to places like the UK or Canada has been analysed within the theoretical framework of dependency theory and the international distribution of labour where it is argued that “under-development as experienced in Latin America and elsewhere is the direct result of capital intervention, rather than a condition of 'lacking' development or investment (Schmidt, 2018)”.

Scene 5 starts silently and with very few elements. We hear the voices of Rolando's family members as they focus on the memories of the use of plants by their mother and grandmother to heal people. For example, Uncle Tino would tell an anecdote about a big trip his mother took to heal a newborn member of the family in Mexico City and all the effort and sacrifice it entailed. He talks about the effectiveness of medicinal plants. In the background, pictures of the family members circulate as a way to emphasise their voices.

From scene 6 to scene 10, there is a rupture in the formal sequence of the piece. The metric count and the sequence of glyphs are now broken. In this segment, the audience sees the condensed script in the shape of constellations, and the consistency of the maraca’s timbre feels unstable and sparse. Instead of the maracas, the foreground is occupied by a *huehuetl* drum pattern. Here, we concentrate a lot of the performance elements developing the narrative against the electroacoustic music piece. What we try to

create is a buildup: transitioning from very quiet to very loud, from very slow to very fast, from a few sporadic events to a more dense and compact musical structure. This section of the piece suggests a movement, similar to that proposed by *Temazcal 1*, from the representation of the "pre-Hispanic" to the "modern" world. The *huehuetl* and the 7/8 rhythmic figure we used for the build-up evoke the pre-Hispanic imagined past and the generic drum n bass combined with the ukulele (reminiscent of the harp of certain Mexican folk music) refers to an imagined modernity. This should be read in parallel with the images of a percussionist performing *Temazcal 1* and me dancing pretending to play the maracas. A close-up of my instruments reveals a *sonaja* that can be brought at any generic touristic market in Mexico. The gestures of this shamanic *personae* I am performing are exaggerated, mysterious and, at the same time, humorous.

Rolando advances his narrative by talking about his aunt Flor, the one closest to herbal knowledge and traditional medicine from that generation. Flor lived on the site of this situated knowledge, along with Rolando's grandmother, where she witnessed how people from the community would rely on her mother to heal children from digestive problems, evil eye, fear, and other ailments. Rolando's main idea in this intervention is the body's autonomy from "professional" medical knowledge granted by the knowledge of these medicinal plants.

In scene 6, a substantial part of the script is missing, perhaps the most theory-heavy part of it. Here it is:

> *"The Mexican modernist project envisions a capitalist modernity that excludes Indigenous Peopless as political subjects but assimilates their cultures via a social order reminiscent of the Spanish caste system. This is often understood as mestizaje.*

Mestizaje have allowed state managed capitalism and financialised capitalism to expropriate land and labour from Indigenous living communities, turning them into cheap labour. Mestizaje is the process of proletarianisation of Indigenous Peopless and the formation of what Marx describes as the worker's reserve army (people that can be introduced to the labour market in case of shortage or to keep the price of labour low). Furthermore, mestizaje has been key to enabling the originary accumulation necessary for capitalism's boost towards imperialism and the accumulation by dispossession common in late capitalism."

Scene 7, with a metric count of 7 as well, is considered by us as the vertiginous centre of the piece. At this point, we are trying to introduce as much instability as possible to the aural aspects of this work. In the background, the constellations appear, sometimes more readable than before. It is possible to read:

"Mestizaje is expressed as a cultural process in which European ontologies and epistemologies capture Indigenous cognitive territories, transforming them. In other words, mestizaje allows the metabolisation of anything external to whiteness by eurocentric frameworks. A mestizo speaks Spanish but their vocabulary is filled with words from originary languages. Mestizaje tames and depoliticises Indigenous subjectivity (reduces it to folklore and culture) and it attempts to make indigeneity accessible without renouncing whiteness."

The segment of the script above and the previous one trace a relationship between the analysed cultural artifact and the ideological project it reflects: the appropriation and re-signification of Indigenous knowledge. For example, *Temazcal 1* tends to have a male figure at its center. To clarify, there is no written instruction to have a male performer for the piece, but a quick survey on YouTube shows that multiple performances of the piece are almost always performed by a male percussionist. Additionally, the

premiere of the piece was dedicated to Julio Toro, and it was not uncommon at Mexico City's contemporary music festivals to see the composer perform the piece himself.

The visual aspects of the work become hyperactive as the aural crescendo and accelerando advance. The glyphs are revealed as components that, when composed together, form another layer of meaning that I will talk about soon. Moreover, a series of flashes, accompanied by images of pre-Hispanic instruments and schemes tracing constellations zooming in, overtake the work. As the visuals and audio bloat the ensemble coordinates a sudden change of scene that is in many ways precise and effective as an audiovisual movement.

Before I continue describing the artwork, let me take a moment to acknowledge certain performative and technical aspects of this moment of the piece. I want to remind the reader that this documentary is being performed live by three coders in two locations and with three very different computational and networked conditions. This means that each of us, the performers, is experiencing the piece in three very different ways. Furthermore, the three performers have very different relationships to coding.

The ensemble communicates mainly through the chat function provided in Estuary. Nevertheless, the three performers have different interfaces that vary in the way they show the movements of the other performers. In other words, each performer sees very different things as their function in the ensemble requires. The performers also interact differently with the code: from automated generation and evaluation of snippets to almost from-scratch editing of code; from overseeing solely aural or visual components to combining both. Also, each performer uses a different textual interface, JSoLang, with a different underlying language underneath (CineCer0, Punctual, MiniTidal, and TimeNot).

For all the aforementioned reasons, it is necessary to understand the immense ability required by the player to develop the intuition to understand Estuary's behaviors and, perhaps more impressive, the immanent dynamics of the performance. This ensemble-making instinct is unrelated to virtuosity or technical proficiency with computers or technology. At the same time, it differs from other ensembles where the openness and improvisatory ethos allow different performance styles and abilities to interplay productively. Something different drives the ensemble gathered around *Temazcal 2*. In this case, I argue that the time gained by the automation that algorithms and code provide was spent making sense of our experiences related to the *temazcal* rather than spending time becoming 'strong coders.' Thus, the three of us were familiarized with the archive of images and sounds, and we became fluent in its affordances. We developed a form of *mnemotechnia* that was crucial for the performance.

Back to the description of the work, as the build-up concludes, the background freezes, and a clip of composer Gabriela Ortiz, an established composer from International and Mexican environments, appears in the foreground speaking. As Ortiz speaks, a ukulele sample that evokes traditional folkloric music from the south of Mexico appears loudly. A drum'n bass pattern is suddenly introduced, first as code and then as sound, to the ukulele melody, and the audience hears and reads the English subtitles of Ortiz saying:

> *"where is the consideration for voices of composers that nourish from vernacular melodies?... Javier Álvarez... he transcends ... it is not black or white, it is something deeper, it is how to work them and how to appropriate them to generate something very personal that, as a creator, represents me."*

Let me provide some context for this clip (Rodolfo Acosta et al., 2021). On the 21st of September 2021, as part of the dialogue table 'Colonialism and decolonization in concert contemporary music' held online by UNAM as part of the International Encounters of Extraordinary Lectures, five Latin-American and Spanish composers and conductors gathered to discuss the intersection of concert music-making and colonialism. A friend of mine described this talk as something like a gathering of bankers discussing capitalism. Most of the discussion revolved around tropes of representation, statistics of inclusion, and the usual institutionalised discourse. Someone, mimicking whether willingly or not Spanish far-right forums on the internet, claimed that Spanish people prevailed over the 'colonization of Spain' by 'the Muslims' and the effects of that were beneficial for Spanish culture.

There were two particularly interesting interventions. Morales-Ossio argued that the appropriation of 'aboriginal' music represents an act of usurpation on which concert music in Latin America often relies to engage with tropes of identity. He continues to illustrate his point by describing how the Mapuche people in Chile are dispossessed of their land for resource extraction, including wood, and outlining the processes of original accumulation upon which Latin American nations are built. He claims that the appropriation of folkloric musical elements is reductionist and tied to European forms of colonialism, which is merely another manifestation of the usurpation he described earlier. Morales-Ossio's argument is met with an unapologetic response from Ortiz, who defends the right of creators (specifically Westernized artists) to appropriate vernacular cultural expressions.

This is what I have sampled and portrayed in the documentary. This referenced intervention allows me to trace a direct line between *Temazcal 1* and the current stance of influential voices in the composition environment in the face of decolonialism. Interestingly, in previous interventions, Ortiz expressed

concern that (white) Latin-American women are often misrepresented in Anglo-America using stereotypical framing for their concerts or grouping them always only with other Latin-Americans. At the same time, the composer shows a strong conviction to defend appropriation as an inherent right of art creators. I have reflected a lot about this dialogue between established and influential composers. I realized that this is a recurring pattern I have observed before in my experience as an 'emerging composer.'

It is evident at this point that this is a crucial point for *Temazcal 2*. The glass ceiling of progressive neoliberal expressions should be interrogated relentlessly. *Temazcal 2* inverses the relationship of appropriation that Ortiz adamantly defends as we appropriate contemporary composers like Ortiz and Álvarez to talk about the meaning of the *temazcal* obscured by *Temazcal 1*. A second effect is introduced to *Temazcal 2*, a second inversion. Ortiz's complains about the Latin-American label and her impetus to appropriate vernacular culture does not happen within the limits of the Mexican nation-state. Ultimately, classical music consumption in the global north, clearly categorised into different genres based on colonial preconceptions, is organised around the expectation of people like Ortiz and Álvarez to extract novel sounds from people 'outside civilisation.' Gabriela Ortiz will extract culture if centres of power are willing to consume it. Thus, critiquing Ortiz or Álvarez is an incomplete strategy that can eventually lead to misconceptions about development and progress. Considering the strength of the nation-state in the imagination of people and the incapacity to observe the whole machinery of capitalism as a unit, the task is to understand the loss of the *temazcal* as a global problem as much as a local one.

Again, we proceed with the script:

"Mestizaje then should be understood as the alienation from Indigenous Peoples's traditional livelihood in order to integrate them into the globalised capitalist frameworks as servants. As people adopted the clock and empty time over the cosmic cycles to mark the passing of their lives, they became alienated from life and unable to see beyond the cold reality of capitalism.

The migration from rural areas to cities and its catastrophic consequences is a good example of this alienation: Indigenous Peoples leaving behind their lands, languages and culture to become cheap labour. Temazcal (the music piece) is a hegemonic cultural artifact that enables such alienation, using electroacoustic music as a technological and ideological framework."

This part of the script reveals the intention to inscribe the artwork not only as a critique of nationalistic ideology and its art but also to place its relevance as a cultural artifact within the broader context of the social totality we call capitalism.

In the next section, we aim to provide a brief pause for the audience to 'catch their breath.' Sporadic arrhythmic maraca hits can be heard, accompanied by various noises and glitchy sounds. Visually, my script predominates with the following text:

"Alvarez music intervenes electroacoustic ideology in many ways: from timbric to rhythmic imagination, from parametric notation to representation of rhythmic gesturing and cognition at the centre of techno-scientific and artistic music research. In his thesis there is no discussion of Temazcal. However, the piece composed after Temazcal, called Papalotl (insisting on the use of Indigenous tropes) is discussed in technical terms. No mention of the poetic reality it evokes with

the title. It would be safe to assume that Temazcal and the symbolic reality it portrays are not engaged as (traditional or ancestral) knowledge but as an inspiration or even an object of study."

In the background of this scene, we can see the eight glyphs used to show the script up to this point. The separation of both the script and the glyphs is meaningful. At the center of the glyphs, it is possible to see a representation of an *ozomatli* (a monkey figure with earpieces often associated with the deity of dance and music and the constellation Ursa Minor in *Nahua* culture). The arrangement of the eight glyphs and the *ozomatli* at the core (surrounded by a perimeter of 13 petals that are missing from this representation) was originally found in an archaeological artifact referred to as *Huehuetl Ozomatli*, located to the north of Mexico City.

What is known about this artifact has been theorized by Daniel Castañeda and Vicente T. Mendoza in the book *'Instrumental Precortesiano,'* (1933), supported by information gathered by Sahagún in the 7th book of the '*Historia General de las cosas de la Nueva España* (1829).*'* The general thesis of the images analyzed here, developed in their chapter '*Teoría de las Constelaciones Circumpolares en las Culturas Precortesinas,'* consists of explaining how the representation of the *ozomatli* and the octant of glyphs represent real stars and constellations that were fundamental in the pre-Hispanic world to "map" the passage of time. The authors emphasize that this representation is both aesthetically imaginative and free and functions as an objective observation of the stars that trace cosmic rhythms, merging forms of knowledge that can be perceived as scientific and artistic. This scheme, found within a musical instrument, emerges as a constellation connecting music, dance, design, instrument-making, calendar precision, and cosmic rhythms.

Rolando, who brought this text and these images to my attention, and I discussed the relevance and meaning that this section might bring to *Temazcal 2*. Based on these conversations, I am attempting to present, without offering favorable or unfavorable interpretations, how Indigenous Peoples have developed a different conception of time-keeping that can be evoked without being appropriated in a way that relies on dispossession.

The 11th segment of *Temazcal 2* reintroduces the metric structure, with 3 beats per metre accompanied by the sounds of birds. There is also a dramatic shift in color and texture in the visual field. The background features pictures from Rolando's family archive, depicting the site of the knowledge held and transmitted by his grandmothers to their aunts and mother. Similarly, Rolando's voice narrates the relationship between the knowledge his family possesses, the land where it is situated, and the plants. Rolando characterises these plants as animated entities that can possess a voice while remaining silent.

My script, now without constellations or glyphs shaping it, continues to reaffirm some previously discussed ideas:

> *"However, the hegemonic narration of the Mexican state allows us to speculate on a fictional link between old indigenous temporal conceptions and Alvarez's research on cognition, gesture, time and movement. In this way, Alvarez appropriates, assimilates, supplants and re-tells forms of knowledge developed by a community in which he does not participate nor belong. This pattern has been observed before: a nostalgic re-telling of an idealised past followed by a future envisioned as an identical repetition of such ideal past."*

In the next scene, we hear a double-beat metre accompanied by the soundscape of the *Mixteca Alta*. The visuals feature the plants, which were described as active family members in the previous scene. Rolando advances his narrative by reminding us that the family *temazcal* and the knowledge of the plants used in its health rituals are lost. However, he recalls some plants used for his own healing processes when he was younger. Following Rolando's voice, we can hear his mother talking about how her mother healed her and her siblings by applying the same plants.

Juxtaposed with this mourning for lost knowledge, my script points towards another form of dispossession: the erasure of contemporary indigenous lives.

> *"Temazcal's score is a commodity that can be purchased on the composer's website: https://temazcal.co.uk/store/*
>
> *Only 55 USD.*
>
> *He has built a career and a reputation feeding from the prestige of his English education and assimilation as well as the mysticism of his Mexican identity. Simultaneously, broadening the scope of the colonial European project and advancing a Mexican modernity in which Indigenous Peoples are a relic of the past."*

My last intervention is a hopeless image:

> *"The last section of Temazcal 1 is interesting. A traditional folkloric piece for harp, making the maraca player improvise an accompaniment, suddenly appears. The mood of the piece shifts dramatically: from a shamanic intense experience to a festive modern environment. Perhaps implying that the Mexican identity has its roots in rituals like the Temazcal.*

I imagine a restaurant in a nice tourist location. Packed with gringo, euro-gringo and whitexican tourists with pink cheeks. Tired and satisfied, in the middle of their holidays. Eating a feast. Very cheap too! A table being served by the locals; filled with all kinds of food and drinks. The European descendants devouring everything, people with darker skin trying to smile, to move fast. Trying to keep the drinks flowing. The hungry tourists laugh and enjoy the background music. The hungry listeners demand another tune so the feast can continue, joyful and endless."

The last section in Rolando's narrative is very interesting. Ramón, Rolando's uncle, remembers the birth of his siblings (Rolando's aunts and uncles). He recalls that his grandmother (Rolando's great-grandmother) assisted in the birth of all of them. In Spanish, the word for giving birth also means 'to illuminate.' This memory is complemented by the faint light of a candle illuminating Rolando's grandparents. The backward movement of the metric units, mapping cosmic rhythms in the maracas, resonates with the movement of the memories traced: from *Tultepec*, close to Mexico City, far into the past.

Revelations from the Loop of Practice and Theory

In conclusion, I will explain the three major points emerging from the creative process behind *Temazcal 2,* which can be observed in the previous section. I will also explain how these three points connect to the three figures of the Indigenous, the settler, and the *mestizo*; and I will articulate these three points via the triple movement and the already described form of double consciousness. These points are: (a) the irreversible erosion of the *temazcal*; (b) the consequences of a critique of *Temazcal 1* in times of

progressive neoliberalism; and (c) the necessity for new forms of subjectivity such as the one emerging from *Temazcal 2*.

Firstly, the *temazcal* technology cannot be rebuilt – neither metaphorically nor literally – by us without imposing relationships of extraction onto others, since the knowledge required for such an operation has been lost from Rolando's family core. Scene 3, as shown in the video documentation and online repository, is dedicated to the unsuccessful search for the old *temazcal*. The location where the *temazcal* used to be is now an extension of the town's church. Similarly, the family members (all except Rolando's mother) have lost track of its presence in their memories. Scene 4 illustrates our unsuccessful attempt to rebuild the *temazcal*. The repository contains several videos that serve as testimony to this effort. To admit the impossibility of rebuilding the *temazcal* is to acknowledge the necessity of self-determination for Indigenous Peoples and the need to move away from the tropes of appropriation common in *mestizaje* narratives.

Secondly, considering the global advance towards financialisation that has occurred between *Temazcal 1* (1987) and *Temazcal 2* (2022), the hegemonic arrangement that sustained Álvarez's work has been questioned. This idea is the afterthought emerging from Scenes 6 through 10, where the thickest critique of the electroacoustic music piece is presented sonically and visually. Here it is key to emphasise the role of the referenced conversation on decolonialism where Ortiz defends the right of artists to appropriate vernacular culture inherent to contemporary Western art creation.

Rolando and I radicalise this gesture in *Temazcal 2* by relentlessly appropriating tropes, ideas, and images from *Temazcal 1*. As we do so, and considering how this artwork is going to be received and disseminated, we understand that Ortiz's misreading of decolonial frameworks can give us a clue to the

real dimension of the problem that goes beyond any nation-state and its relationship with culture: namely, the market of Europeans and Anglo-Americans hungry for exotic sounds without engaging in extractive practices by recurring to exchange with people claiming ownership of both cultures: Indigenous and white.

Here it is important to understand the critique of *Temazcal 1* through the triple movement described above. On one hand, understanding the incompleteness of a relentless critique of the Mexican institutional environment is key to seeing how marketisation plays a role in this conversation. On the other hand, the negative portrayal of the Mexican environment can serve as a positive conception of the Canadian context in the minds of Canadian (and global north) audiences. Instead, *Temazcal 2* requires articulating the two distinct forms of colonialism discussed here and understanding how they interact within a globally interconnected context. The risk identified in the networked art context – and this artwork – is to, willingly or not, impose the Canadian-centric (or perhaps Anglo-American-centric) settler-colonial paradigm as the sole explanation for ongoing colonial projects in Mexico and to fill a perceived conceptual void. Settler-colonialism, as a critique of *mestizaje,* acts as a form of oppressive social protection.

Third, by acknowledging the first and second points, to avoid *Temazcal 2* from falling back into forms of subjectivity and narratives that unwillingly renew the power of an oppressive nation-state while also striving to suppress any forms of capture by forces I identify here as progressive neoliberalism, we delineate a figure that is neither Indigenous, nor settler, nor a product of *mestizaje*. This subjectivity represents a consciousness that articulates the situated knowledge of Rolando's family alongside techniques of appropriation that transform a global project towards whiteness – like *mestizaje* – into a

possibility of interconnected self-determination and a way of being in this world that builds social relations beyond capitalism and nationalism.

This point is illustrated by, on the one hand, the productive destruction of *Temazcal 1* at the hands of *Temazcal 2,* along with the former's relation to exploitation and dispossession critiqued in the last scenes of the latter. On the other hand, the overarching theme of Rolando's grandmother's relationship with plants and healing that often accompanies the erosion of the *temazcal*. It is crucial to understand this knowledge of plants as a dynamic and living memory that changes with the movements of Rolando's family. It is important to note that the relationship with the land portrayed here cannot be described within the tension between the settler and Indigenous.

As I have insisted throughout this chapter, I have not found the specific language to define the form of subjectivity emerging from *Temazcal 2*. However, I have attempted to describe some of its attributes and affordances in the more theoretical section. I conclude with the encounter of practice and theory, hoping to emphasise that, methodologically, this is where this artwork is strongest.

Chapter 2 - TimekNot: displacing drones and beats with radical polyphony

TimekNot is a computer language that allows live coders to program heterogeneous, music-oriented temporal relationships on-the-fly and instantiate them as triggered audio samples. TimekNot's core is a robust systematisation of time relations between relatively autonomous musical layers or, perhaps more telling, timelines. Hence, it can be understood as a polytemporal language, for the etymology-scrupulous: multi-temporal or poly-chronical, why not. For the tradition-oriented and vanguardist-at-heart: the music this language thinks about is more or less a multi-tempo (or poly-tempo) music. Or, why not again, *tempi* music. In this chapter, I will attempt to explain software as a cultural project and artwork that seeks to intervene in the algorithmic music landscape and live coding ecology. I will try to explain the alliances traced by the grammar utilised in the language and the experiences and observations that have led to its current form. I will also describe its functionalities and technical affordances, but I won't go into a lot of details – a Read-Me document will be provided as part of this artefact that will explain thoroughly its use.

Context

TimekNot emerges from multiple experiences and desires. Walking downtown in any big city will appear as a chaos of sounds from various places whose overall texture is identifiable as a soundscape. However, in Mexico City, you can walk through a *tianguis* (a market that is installed precariously and ephemerally in

the street) and hear 4 or 5 different sound systems blasting music at full volume at any given moment, appearing to compete for the airwaves. This is not a phenomenon emerging from electronic devices; one can walk to Mexico City's Garibaldi Square and listen to five simultaneous traditional music bands playing different pieces with different instruments and with different noisy audiences around you. Differently from *tianguis's* sound systems, the music bands cannot avoid listening to each other and, from time to time, 'converge' into each other's rhythmic flow or at least, the listener will perceive in a pronounced way that this happens. Thus, polytemporality is an abstraction that originates in material and lived experience. What is interesting about the kind of polytemporality I am invoking is that difference is not an operation that needs to be solved toward identity. Neither does this polytemporality dismiss the effects of the multiple timelines on each other. Time does not need to be collapsed into universality, but polytemporality maintains the tension of difference as a valuable cultural experience.

TimekNot finds a direct precedent in my master's major research project: TimeNot, a language to live code tempo canons *a la* Nancarrow (Franco Briones & Villaseñor, 2020; Franco Briones, 2019). Tempo canons are a series of identical melodic structures transposed in tempo and pitch played together and share a point of convergence. The point of convergence is the instant of the music work where the transposed melodic structures and the chronological time coincide. The tempo canon was a way for Nancarrow to allow listeners (himself mostly) to differentiate the aural effects of the proposed tempo transposition. The difference between the transposed melodies, when they are not at the convergence point, is called echoic distance. This concept, echoic distance, can be defined as the chronological interval between the same structural point of two different transpositions. As can be deduced, at the convergence point, the echoic distance is zero. From my experience with TimeNot (notice the lack of -k), I

have been able to generalise the concept of polytemporality by introducing concepts that emerge from the specific conditions of live coding practice in particular, digital music-making in general and my artistic research experience. Perhaps TimekNot is proposing to focus on an outsider's temporality.

TimeNot was, in many ways, a "fancy delay" that allowed players to explore a very specific effect felt on top of either a melodic idea or a sequence of samples (where the index of the sample was transposed rather than its pitch). It relied, more or less strictly, on the concepts of convergence point, echoic distance and transposition as conceptualised by Kyle Gann and Conlon Nancarrow (1995). TimeNot could create tempo canons with a determined duration, and the transpositions were always smaller than the intended total duration, allowing these programs to loop identically. While TimeNot explored nested musical ideas and repetition, TimekNot focuses on difference and openness.

Defining TimekNot

TimekNot has taken a more radical approach to polytemporality. As will be evident in the notation, TimekNot separates completely the temporal aspects of its programs from the aural ones, and by doing so, it allows users to create purely temporal structures and relationships. Like any clock or metronome, these temporal structures discretise (or grammatise) the flow of time. This time, grammar can be flat and regular, like the seconds of a stopwatch, or it can be structured and irregular, like musical form or cinematic temporal experience. These time structures serve as a base for building musical material and act as expressive clocks that function as reference points to create other clocks serving that same function and so forth.

Unlike TimeNot, TimekNot does not specialise in tempo canons. So, two musical structures that converge do not have to be transposed, identical repetitions. The two converging ideas can be heterogeneous and radically different from each other. This implies that spontaneous and non-synchronic relationships are possible; I am not interested in mathematically complex temporal relations, loose synchronicity that favours ethereal music textures, or chance-based or stochastic randomness; I want to prioritise intentional and expressive temporal relations chosen by the human player for, perhaps, no explicit reason. If the converging music structures are not identical, the convergence's morphology is two-fold: the point of the structure where the convergence happens and the point of the other structure where the current one will converge. Thus, convergence points in TimekNot signal the possibility of a convergence between different musical ideas rather than the certainty of echoic repetition. This means that the concept of echoic distance is also redefined; rather than emerging from the tempo canon, as was the case in TimeNot, it becomes a decision made by the player. It can intentionally involve a transposition of tempo and pitch/timbre; it can be a transposition of pitch/timbre with a different temporal structure altogether or something completely different. An instance of this last case may take many forms; one could be a program where the moment a temporal structure converges with another showcases salient parameter values (like pitch, gain, etc.) that indicate that the convergence point, acting as an attractor, is structurally meaningful. For example, imagine a melodic figure in which pitch intervals only ascend towards the convergence point and only descend afterward. Now, imagine that this structure converges with another melodic figure that moves in the opposite direction: it only descends towards the convergence point and ascends after it.

TimekNot relies not only on transposition of ideas, but it also tends towards non-identical repetition by considering a convergence point as external to a loop. If two temporal structures are looped, each new iteration of the structure is always different because it relates to other structures uniquely. The convergence points act as anchors from which it is possible to compare different structures. Non-looped musical ideas have an inferred timeline useful to allow them to converge with broader structures. The ability to infer infinite timelines from a finite structure – as well as the understanding of these rhythmic structures as 'clocks' for other rhythmic structures – has been one of the major arguments for me to consider additive rhythm rather than divisive, as I will explain in subsequent sections.

I have referred to temporal structure in a general way; however, these structures are in themselves divided into two: polytemporal and rhythmic aspects of the temporal structure. Previously, I have referred to the temporal structures as expressive clocks; the polytemporal and rhythmic aspects here mentioned respond to this description: polytemporality creates (multiple) clocks while the rhythmic aspects make them expressive. The polytemporal aspects help players to manage the alignment of the timeline with other temporal structures and generate the clocks to synchronise sounds and the rhythmic aspects will produce the expressive formation of onsets/offsets and blocks of information that respond to the rationale of additive rhythm.

Time: Polyrhythmic? Polymetric? Polytemporal?

To better understand the temporal structures described here and the possibilities of TimekNot as a polyphonic music technology, we need to contrast another set of concepts: polyrhythm, polymetre, and polytempo. I define polyrhythm as the juxtaposition of different repetition speeds of musical events by subdivision and polymetre as the juxtaposition of different repetition speeds of musical events by

addition. Unlike polyrhythm or polymetre, polytempo relies on convergence points that do not depend on the (shared or different) structural characteristics of the various musical ideas at hand. Thus, polytemporal music, as I frame it, bypasses normative ideas of 'harmonic rhythm,' which are prevalent in popular modernism, postmodern eclecticism, and vanguardist/experimental modernism, opening a field of possibility for polyphony. In other words, polytemporality does not necessarily create complex rhythmic relationships and structures; instead, it captures certain attributes of such complex relations. Polytemporality, as I have expressed more generally regarding live coding music, focuses on breaking and questioning patterns rather than harnessing them. Ultimately, polyrhythm, polymetre, and polytempo are different ways of conceptualising the duration between events or the speed at which events occur. An example of a polyrhythm could be a basic triplet against quarter notes in Western academic music, juxtaposing four notes against three. An example of polymetric rhythm is found in Karnatic music. Tisra in Chatusra is a technique that juxtaposes a metre of three within a metre of four, producing a sound very similar to the polyrhythm previously mentioned, but it is conceptualised through the addition of rhythmic units rather than subdivision. Thus, as a purely listening experience, there would be no difference at all between a polymetric and a polyrhythmic musical idea; what changes is the cultural context, which can be reflected in its notation or mnemonic mechanisms for the transmission of knowledge relative to its own cultural context, and one might even say in its political economy. It is possible to express this polyrhythm and polymetre in a polytemporal way. For example, the musical idea mentioned above could be notated with one voice at a tempo of 240 bpm for the quarter figure, played against another voice at 180 bpm for the quarter. Let me reiterate: the perceived sonic outcome from these three different techniques can be quite similar; nevertheless, notating a sonic phenomenon with three different

organisational schemes in mind is a relevant part of live coding, where notation – visible to the audience as a performance – plays a foundational role.

I draw from concepts of African polyphony (Arom, 1991), Karnatic music (Reina, 2015) and ideas on metre and groove (Abel, 2014) to differentiate and negotiate notions of additive and divisive rhythms as well as ideas of polyrhythm and poly-metre, allowing me to form a coherent and cohesive understanding of rhythm, musical time and musical form. I am drawing extensively from Pätzold's (2014) and Toop's (Toop & Ferneyhough, 1995, p. 285) analysis of temporal musical aspects found in Brian Ferneyhough's music, where various notions of metric and rhythmic complexity are explored thoroughly. Perhaps more importantly, I am carefully considering the insights of Henry Cowell on his seminal work New Musical Resources (Nicholls, 1996) and Gann's analysis of the oeuvre of Conlon Nancarrow as well as analysis on the work of Charles Ives as a basis for polytempic music (Thoegersen, 2022). I am drawing from Spiegel's pattern transformations (1981) to extend the basic organisation schemes I have implemented for this language. I am also drawing extensively from the musical and intellectual works of Germán Romero, Iván Naranjo (Naranjo, 2017) and Samuel Cedillo, who have a particular way of understanding polytemporality in a structural, post-structural and post-phenomenological framework, respectively. Beyond musical style and form, Lauren Redhead has allowed me to think of the web of rhythmic, metric and temporal concepts as a language as they resonate with her concepts of heterochronicity and non-linear time that rely on 'the presence of some organising principles, some macrostructure and syntax that permits categorical understanding of a work's signification and its semantics (Redhead, 2022, p. 154)". In other words, Redhead points to notational principles that do not sit outside time but are iterations of events in-time, which intertwine *aesthesis* and *poiesis*. In Redhead approach, this takes the form of sampling

performances of the piece that later become part of the electroacoustic components; in my case, I register all this temporal and rhythmic knowledge as a repository of code where these ideas are instantiated, and later modified or reinforced, as notation, syntax and grammar that can be invoked to create music rather than determine a musical pathway.

Coding Paradigms

I have been inspired extensively by concepts of Functional Reactive Programming (Krouse, 2018) to implement certain aspects of TimekNot. While with different connotations, I draw extensively from the concepts of behaviour, event and dynamic. At the heart of TimekNot there is the notion of a continuous flow of time and the cultural (and scientific) understanding that, to change our cognition of time we need to discretise it, taking samples of it in ways that reflect our intentions. Nevertheless, these concepts, associated with computer science, convey only partially the situation in which TimekNot operates.

Algorithms and programming are central to our current historic moment, a moment when the general conditions of production rely on software that needs to be coded by humans. Programming is at the centre of capitalist accumulation processes. Nevertheless, with AI framework enforcement and the automation of software creation via generative Artificial Intelligence, coding is becoming a (sort of obsolete, post-industrial) cultural activity resonating broadly with the ideas presented by Noble as post-modern programming (2004). Culture-oriented coding, like post-modern programing, concentrates on the vocality of code, its double articulation and its ephemerality (Cox & McLean, 2012). Culture-oriented coding relies on meaningful surfaces and stands in a contradictory position with culture emerging from the dev operations of programming jobs. In Chapter 1, I elaborated on the double consciousness emerging from the processes that *Temazcal 2* reveals: the *temazcal* technology, on the one hand,

captured by *Temazcal 1* becomes bound to perform a vulgar emptying-out of what might be considered "pre-modern" subjectivity; *Temazcal 2* captures *Temazcal 1* and empties it out from its whitening function, at the same time it proposes a subjectivity capable of escaping capitalist rationality. A similar double consciousness emerges from culture-oriented coding as well: coding activates a culture distant from cognitive white-collar work capable of escaping capitalist rationality.

Live Coding

Live coding, as a practice, offers a rich set of examples of how to conceptualise musical time that have certainly informed the process of TimekNot. The time-dependent variables in FoxDot (Kirkbride, 2016) allow me to think of program block iteration as meaningful for parametric changes in the running program; the notion of recursive time in Ex-tempore (Sorensen, 2018) makes use of asynchronous time scheduling in a way that TimekNot will take advantage in future implementations; cyclical time in TidalCycles (McLean & Wiggins, 2010) and SuperCollider's pattern library (Harkins, 2009) allowed me to understand pattern-oriented composition as an affordance of computation; and the Just-In-Time SuperCollider library (Rohrhuber et al., 2005) as well as Punctual's development (Queralt Molina, 2023) of such idea have allowed me to think of programming gestures as expressive within the context of music-making. Many notational ideas have been thought through explorations on these languages and idioms. There are theoretical discussions that are extremely relevant for this software like Rohrhuber's concepts of passage and encounter as metaphors for time conceptions derived from algorithmic music where the passage of time (from past to future) is contrasted with the encounter of events in a time that resembles a "location without place" (Rohrhuber, 2018, p. 21). Similarly, live coding is a thinking-act that relies on Kairotic instants – which are both, a "cut" and a "will to invent" (Cocker, 2018). Such cuts are an

expression, a performance even, of human agency vis-a-vis modern computation that ultimately signals that time in live coding is a multiplicity of possibilities that are kept from happening simultaneously by an aesthetisised thinking-act, a time that requires agency from the subject engaged with it (Franco Briones, 2022).

One difficult binary that live coding navigates is the oral/written transmission and register of knowledge. Taking this into consideration, instead of a literature review on the state of the art of live coding, let me offer reflections on my participation in conversations around live coding in an Anglo-American and transnational context.

The International Conference on Live Coding (ICLC) has been taking place since 2015, providing a relevant intellectual arena to discuss programming, or coding, as a cultural act. It is an unusual academic space since it usually stands between software demonstrations, theoretical (either scientific or artistic) research, and performance art. Usually, an ICLC day would start with a series of panel discussions, followed by workshops and in the evenings, there would be concerts and performances. Novel software would pace the whole conference, followed by the performances. The theoretical expressions that would be salient are those that integrate computational science with artistic practice. Nevertheless, this ICLC seems to have moved away from such a scheme.

In the 2023 ICLC plenty of systems have emerged that recombine numerous ideas present in earlier systems. I believe that TimekNot can be understood as part of these systems: not as the edge of technical power but as a re-interpretation of a well-established culture. Aligned with this set of systems, several questions arose about who has access to live coding and who participates in it. Saliently, the notion of dissolving the audience vis-a-vis an active community that collapses the exchange, production

and consumption of art was put forward. Perhaps we are witnessing a partial fulfillment of the old promise of live coding: a language per person to express their individual artistic idiosyncrasies. Live coding is at a point where computational systems, artistic practices and communal experiences are being confused, problematised, redubbed and thought over. The keynotes at this conference have an underlying theme: liveness/life/aliveness. First, the reinvigoration of notation without computational side-effects as proposed by Sichio; second, Baalman's retrospective analysis of what live coding – "as an act of rebellion against the fixed idea software as immutable, impenetrable, but yet advertised as neutral, systems that are humanity's future" (2023) – has achieved in the last 20 years. Lastly, the reincarnation of Click Nilson, a *personae* invented by Nick Collins, was first intended to differentiate himself from the American experimentalist composer Nic Collins; but secondly, what I interpret as a critique of the reverence for the 'excentric genius and creator' that the communal ethos of live coding also rebels against.

In an era that Nancy Fraser (2019) characterises with the Gramscian "old is dying and new cannot be born," the openness of live coding's transnational community to leap into the future, allowing the new to speak, is notable. It is also notable that, with the new, the morbid symptoms, like genius excentric composers, are also re-born. The discussion on subjectivity earlier in this section and in Chapter 1 appears as particularly relevant: the live coder risks becoming a hollow, shamanic genius composer, or they hold a key for a movement of programmers, artists and scholars capable of articulating their double consciousness.

More recently, Estuary's 8th birthday is marked by the implementation of the ExoLang pathway. Exolangs are a mechanism in Estuary that enables people to modularly integrate their own computer languages

into Estuary's multilingual, collaborative environment aligning with the ethos of the ICLC. If the promise of live coding is one language per-person, the promise of Estuary is an anti-Babel cyber-space-time where all languages are understandable amongst themselves in their aural or visual effects. TimekNot emerges from this context and proposes a notation that responds to its challenges and opportunities.

TimekNot's Core Components and Intervention

TimekNot's conception of time pays particular attention to some concepts that emerge from live coding practice. Saliently, the instant in which the player communicates a new program to the computational system. This instant is called evaluation time. So, additionally to polyrhythm, polymetre and programs that are indifferent to evaluation time, TimekNot can produce polytemporal relationships anchored in evaluation times to facilitate the computation of musical ideas that mix infinite with finite, and ideas that have an explicit, intentional starting point and others that do not. Evaluation time, in TimekNot, enables players to orient their performances towards the convergence points or any other relevant instant in their intentions. This might seem subtle, but the proximity of intentional convergence points in the immediate horizon of many of TimekNot's evaluated expressions affects the sonic outcome and offers new affordances to the performer engaged in the listening-editing-evaluating loop.

TimekNot's aural expressions – designed to craft the sound of each individua event – are simple but flexible; they are based on an indexing system that uniquely identifies an instant in the timelines created by the temporal expressions. These indexes are assigned to a sound parameter depending on the intentions of the performer. The values assigned can be manually written by the player, they can be replicated from another voice, they can be transposed, or they can be changed depending on their relationship with the convergence points. With the temporal notation and this modest aural notation, a

surprisingly broad variety of sonic structures can be formed very easily. The notational independence between the temporal and aural aspects of this software may appear counterintuitive and complicated but it tends towards slow coding and to think of musical structures that have a longer breath. At least that is what I have experienced in my personal practice.

TimekNot was conceived within the context of networked collaborative, multilingual live coding. It is deployed both as a standalone and as a language in the software Estuary, meaning that special attributes to explore collaborative and multilingual live coding are essential to its core functionalities. Even though musical ideas favoured in TimekNot can be, to a certain extent, autonomous in terms of tempo to external time-keeping mechanisms (it does not necessarily synchronise to external or 'master' tempo sources), it has a baked-in mechanism to synchronise external tempo using the Tempi library. Using this ability to synchronise multiple tempi (internal to TimekNot) with external tempi (external, in principle, coming from Estuary) it will be possible to interplay easily with MiniTidal, Punctual, CinerCer0, Seis8s or any other language that is part of Estuary's ecology. Secondly, due to the zone system that makes the interactions in Estuary's ensembles, it will be possible to access information from other TimekNot players or editors. This feature will have a particular status in the language's development and will be briefly explained later in this paper.

In this way I have constituted TimekNot as a fragile music-making tool quite capable of making the performer contradict themselves logically at any given moment, where each coding embodiment manifests its own time and time relations. TimekNot should be understood as a music technology where there is no abstraction that functions as a master grid, disciplining and homogenising music making, but concrete temporal sites all of which can act as a reference to any other player or musical idea.

The Temporal Notation and the Computation of Time

In this section, I will describe the program's structure and the purpose of each component. I will start with the more general characteristics and will also go into finer detail as I move along. I will first describe the top layer of the program where the different temporal expressions are identified with a keyword, then I will describe the polytemporal aspects of the notation, followed by the rhythmic aspects described in detail. Later, I will describe the loop mechanisms and comment on external sources of time-keeping assistance. Finally, I will refer to a particular implementation: acceleration, which adds a substantial layer of complexity to the temporal aspects of this notation.

A Map

At the top of each expression, there is a way to identify the temporal idea with a string. Since the most interesting interactions in TimekNot rely on creating temporal relationships between autonomous musical structures the identifier is very relevant. Similarly, attaching audio side-effects is a separate process from the more or less eccentric temporal notation that makes use of the identifier. The notation for identifying a musical idea is:

`identifier <-` or perhaps clearer: `myClock <-`

The left of the arrow is where the identifier should be written. To the right of the arrow the time structure is described. At the top of the program is a map (2-3 (balanced) search tree); these structures let me identify different, same-type expressions and group them together without losing track of their inner arrangement. At this layer of the program an interesting problem emerges as an effect of a polytemporal conception of time where a musical idea is dependent on others. This map of temporal expressions

might lead to infinite recursive loops where, for example, structure-3 converges with structure-2, which converges with structure-1, which converges with structure-3, etc. This points to an interesting development pathway for this software: a notation in which a form of causal loop can be notated and instantiated as a musical idea. Casual loops are a common trope in science fiction or speculative narratives; for example, in the science fiction Netflix show Dark, the main character travels in time to become his grandfather. To explain this, I need to talk further about convergence points and how they differentiate one musical idea from another. Let me clarify, the causal loop idea here presented would not mean that a causal loop can be implemented but can be represented in a similar way that science fiction would. It is possible to understand these causal loops as speculative exercises to explore time as a concept. What cultural and musical knowledge could emerge from such structures?

Polytemporal Notation

Polytemporality, as defined above, is a music technique where autonomous musical ideas are juxtaposed via convergence points (an instant of the temporal structure that converges with another instant of another temporal structure). Since I am implementing finite structures that are not exactly ephemeral, I must implement an on-the-fly anchoring mechanism where convergence points are calculated depending on intentional time points. So, to synchronise temporal structures together I have developed three main mechanisms: a) evaluation time, b) referring to the external voice (the voice emerging from Estuary's tempo), or c) converging with another voice defined by the player. There is an additional function to save convergence points in the cache of the program that will be explained in detail later in this chapter.

The different anchoring mechanisms are differentiated by the number of arguments next to the identifier's arrow. One argument binds the voice to evaluation time, where the argument represents an onset time from evaluation time. The case with two arguments will imply that the voice is bound to the external voice, and the case with three arguments, where the first one is an identifier, means that the voice converges with another voice made by the player. The first case is very straightforward: start of the voice happens at evaluation time or any given seconds after evaluation time.

```
v0 ← atEval 240cpm
```

The above snippet implies that the beginning of a voice is exactly at evaluation time; the tempo of this voice is 120 cycles per minute, and the identifier is v0.

```
v1 ← 3 secsAfterEval ((1/4) = 100bpm)
```

The program above is like the previous one, but the voice starts 3 seconds after evaluation time and has a tempo of 100 bpm the quarter note. Note that these programs will not produce any sound but they enable a clock.

Let me explain the other two cases in more detail. The second case (external convergence from now on) re-conceptualises external tempo sources (Estuary's tempo in this case) as a silent, one-event voice that elapses forever. The Tempi library provides functions to transform evaluation time into a beat count, which is understood in TimekNot as a looped voice silently elapsing.

```
v2 ← (4 afterEval) (10) 270cpm
```

The program above has a tempo of 270 cpm (cycles per minute). Since it has two arguments in the convergence section, this implies that its convergence points are calculated with the external voice. The

current voice converges to the external voice four events after the evaluation time. If the external voice is near event index 2666, that means the convergence point will be at event index 2670. The current voice (`v2`) will be converging from event 10, meaning that `v2` will be at event 10 while the external voice will be at event 2670 at the same time.

```
v3 ← (mod 4 2 roundEval) (5-0.3) (1/4 = 90bpm)
```

The voice `v3` will have a convergence point by adding 2 indexes to the nearest multiple of 4 in the convergenceTo and converge from the position marked by the structure index 5-0.3.

The third case (convergence from now on) is identical to the second except it creates a convergence between two voices defined by the player. External convergence converges with an external voice and (internal) convergence converges with an internal voice.

```
v4 ← v3 (17) (13) (v2 5:7)
```

The voice v4 will converge with voice v3. It will converge to index event 17 and converge from index event 13. The tempo will be defined by taking the tempo v2 and getting the 5:7 ratio.

The two arguments in external convergence and the last two of convergence represent very similar data structures that have a key difference: a point of the converged previously/externally defined structure (named convergeTo) where the currently defined structure will converge and a point of the presently defined structure (convergeFrom) that will converge with the previously/externally defined one. This might sound too complicated (and certainly is), but it is the core of the language, and it is a good idea to proceed slowly and patiently to understand these two concepts. These two arguments require data structure and notation to calculate the convergence point according to desired instructions from the

players. The ways to calculate them are percentage, structure and process, all of which describe a position relative to a block or block series. It is a coordinate system that prioritises the precise position of a block, the flow of time snapping to meaningful points indifferent to blocks or the flow of time based on a precise description of the temporal structure.

The percentage represents a percentage of the elapsing structure where a convergence point will happen. It is notated with a number between 0 and 100 (or more) followed by a %. The percentage is a way of organising the temporal structure without considering the points of the structure where a sample is triggered. Process is a series of indexes that moves forward from the start of the voice. If it is looped, it will traverse each new iteration of it. Structure is a way of calculating convergence points according to a coordinate system that can distinguish between structure iterations, position within the structure and subdivision levels. An example of the syntax is: 2-0.3. This would mean the player refers to the third iteration, the first event with a layer of subdivisions, and we are pointing to the fourth event within this subdivision. Process and Structure snap into the rhythmic structure and might align perfectly with a triggered sample or a silent event. One of these three data structures will suffice for a convergeFrom. However, the convergeTo argument, most of the time, has to be merged with evaluation time since it is implied that an already running timeline is being referenced and the player needs to have some information about its current state. There is a data structure that will manage the alignment between a convergence point and an evaluation time. There are three ways of producing this alignment: origin, snap and modulo.

Origin will ignore evaluation time and align the new voice with moment 0 of the converged voice. Origin constructor can be invoked by giving only a percentage, process or structure. For example, `(4-2.0)` will

create a convergence point of 5 iterations and the second event after the beginning of the voice. Snap will start a count, either around or near the event, after the evaluation time. The type of the count will be defined using percentage, process or structure data constructors. So, the expression: (13% afterEval) will make the convergence point the next beat of the converged voice after evaluation time plus 13%. The last aligner is called ‘mod’ and has an integer as an argument. Mod will start a count at the closest to the event (can be either before or after) that is multiple of the integer given. The expression: (mod 4 7 roundEval) will make the convergence point when the converged voice is at a multiple of 4 near evaluation time.

If the polytemporal aspects here described are not of interest to the player, they can be bypassed by substituting the convergence points notation with the word diverge. Diverge, in addition to ignoring evaluation time, will align the moment 0 of all involved voices, which means that the start of the voice being defined is going to converge with the start of the external voice. A divergent behavior is the best strategy to align fully with other live coding environments and music-making tools that rely on sample triggering.

```
v5 ← diverge 120cpm
```

Assuming that Estuary’s tempo is 120cpm, v5 will be perfectly synchronised with any other language in Estuary that relies on its tempo. The only difference here would be that Estuary’s full tempo mark is 120 bpm for the quarter note, while TimekNot’s tempo mark is 120 bpm for the whole cycle (what I call cpm). This means that every 4th sample triggered in Estuary’s Minitidal will trigger one event in this language.

Conceptually, this is an array of ideas that are difficult to communicate and immerse oneself as a performer. It would be naïve to think that this software is being produced to be consumed *en masse* or even to be used in the little niche corners of the live coding ecology as a standard like SonicPi, TidalCycles or FoxDot. However, and with a certain layer of irony, it responds to the notion, well established in vanguardist composition circles, of 'creating your own language' for music-making and the notion of 'art without audience to programs without users' that marked specific desirable ideas on live coding scene: each live coder should develop its own set of tools according to their idiosyncrasies, and the proliferation of such tools marks a healthy scene rather than the dependence and proliferation of users of another given environment or language. Once that has been clarified, I also believe that TimekNot, as a non-polytemporal music tool, is very easy to understand and use when starting to play music. Furthermore, it is a language that eventually will allow openings for players to create their own functions and idioms as part of their relationship with the language. The language is ready to be used, contemplated, discussed, destroyed, hacked, pirated, copied, misinterpreted, or even a source of profit (but hardly).

I believe the major conceptual contribution to the field of live coding that this part of the language offers is the synthesis of evaluation time and polytemporality that allow a kind of synchronisation that merges embodiment, finitude, encounter and difference, as has been explained so far. Investing attention in the moment of evaluation without losing track of infinite loops and finite structures relevant to the audio composition can illuminate new ways of engaging in live coding. Thus, this software moves against the Bebop principle: 'You are either on or off the train'. With this program, you can be on and off the train simultaneously.

The last component of the polytemporal notation – and first of the rhythmic notation perhaps – is the tempo argument for each voice that can be notated as a cpm (cycles per minute) mark, a cps (cycles per second) mark, a bpm (beats per minute), a ratio interval with either the external or another voice as reference, an acceleration pattern or a duration. The first three options are absolute measures that do not rely on any other voice or information. The cpm and bpm tempo marks require further explanation. BPM is a common concept in academic music-making; on the top of a score often you would find a tempo mark that will equalise a rhythmic figure with a bpm number. This is a 19th century convention that appeared when the metronome started to be used to describe the tempo in a music work instead of an arbitrary approximate description. Nowadays, the rhythmic figure is often omitted and assumed to be the quarter note the default. In TimekNot, given its attempts to extend the ways in which the player describes the tempo, it requires the whole explicit tempo mark. The cpm is useful when a player attempts to use the rhythmic notation (that will be explained in the next section) as the main expressive vehicle. cpm is also the only possible tempo mark that can be drawn from an external source.

The tempo marks are notated: 120cpm, 3cps, (1/4) = 120bpm respectively. The ratio option creates another relationship to any other existing voice, including the external one, where the tempo of the current voice is defined proportionally with another. For example, to use the famous dissonant tempo relationship from Nancarrow's Study 33, the player would have to notate: 41:29. Without an identifier, it will relate to the metric voice. There might be an identifier before the proportion: v0 41:29. In this case, if v0 has a tempo of 120, then the current voice tempo will be determined as follows: 120 * (41/29).

The notation for acceleration patterns is provisional and only supports lineal and sinusoidal movements within the rhythmic block. The notation would look like this `: (lin 300cpm 1200cpm)` or `(~ 1 << 0 range:`

`300 cpm, 2000 cpm)`. The last one, describing a sinusoidal pattern, has a first argument being frequency (where 1 is one cycle per rhythmic block), phase and range.

The duration notation will allow performers to think about the overall duration of the blocks rather than the frequency of their components. This is useful to fit extremely different programs into the same temporal segment.

The tempo argument ambiguously stands between the construction of the rhythmic structure and the polytemporal structure of the musical idea. It is the argument that gives character to each voice and determines the duration and density of the rhythmic structure. Since it can be determined using proportions taken from other voices, I have decided to group it as part of the polytemporal sub-notation of the language.

Rhythmic Notation

Once the tempo is established, as well as the starting point of each voice and its convergence point, the player must describe a rhythmic structure. The conceptualisation of rhythm I am applying here consists of a binary of opposed behaviours; in the first functional version of TimekNot, these behaviours are whether to trigger a sound or not and the notation appears as follows:

```
xooxooxoxoxo
```

This binary of triggered sounds and rests is organised as a sequence of isometric durations. This means that the time interval between one and another onset/offset is always the same, except for the acceleration tempo mark or a particular instance that will be analysed later in this section. This isometric distance is established in the notation's tempo mark. So, if the tempo is 60 cpm, the distance between

any onset/offset will be 1 second. If the tempo is 120 cpm, the distance will be 0.5 seconds between each onset/offset. The notation I have chosen to express this idea is ‘x’ for triggered sounds and ‘o’ for untriggered ones. I have chosen this notation (XO from now on) for multiple reasons. Firstly, it is reminiscent of drum tablature notation often used in popular music and protest marching bands, which invokes a very intuitive understanding of rhythm. Secondly, computer keyboards have the ‘x’ and ‘o’ keys in a very natural position for the hands, reminiscent of a ‘drumming position’. As a performer, this allows me to ‘feel’ the notated rhythms with my body before listening to their computarised instantiation. There are four other variations of the rhythmic notation I would like to elaborate on in this section. The four variations here mentioned are, in a way, shorthand notations for something that can be expressed with standard XO notation.

Repeat Notation

The repeat notation allows players to repeat a pattern any number of times without doing it manually. This is particularly useful in cases where a player would need to re-state a rhythmic pattern a limited number of times rather than the choice to play it once or looped ad infinitum, as would be the more standard option explored later in this section. This notation looks like this:

```
!oxoxoxxox#3
```

The pattern above could be re-written as:

```
oxoxoxxox oxoxoxxox oxoxoxxox
```

Bjorklund Notation

The Bjorklund notation applies an Euclidean algorithm to one or two rhythmic patterns. This algorithm will try to spread a series of impulses as evenly as possible in a finite Euclidean space. For example, distributing 3 impulses in an eight-point space would look like this using the simplest rhythmic notation here developed:

xooxooxo

This rhythm could be interpreted as a Cuban tresillo, but depending on the cultural framework of the listener, it can also be interpreted in many ways. In Sub-Saharan music, this is known differently. The Bjorklund algorithm may appear as a generator of popular rhythms, as Toussaint (2005)claims. Nevertheless, two factors need to be considered here. Firstly, a popular music rhythmic pattern is a form of situated knowledge integrated to multiple aspects of music creation that cannot be reduced to an onset/offset pattern. For example, the *tresillo* figure can be heard in the left hand of the piano parts in Cuban *Habaneras*. But if we change the context to *Son Cubano*, we will find the *tresillo* as part of the *clave* pattern, which is often bound to a timbre (or series of timbres). So, the rhythmic figure of the *tresillo* is related to certain timbres depending on cultural and social contexts as well as broader 'rhythmic phrases' and functions. Secondly, the pattern (3,8) - sometimes identified equivalently as *tresillo* – does not hold the same status as (7,13). This is because (3,8) is ubiquitous on a planetary scale as a popular rhythm, and (7,13) is a rather eccentric pattern with no clear cultural significance as a popular rhythm. Thus, Bjorklund algorithms do not generate rhythmic patterns; they generate arrays of Boolean values that sometimes resemble or are useful to generate popular or unpopular rhythms.

The additive rhythm prioritised in TimekNot proposes reorientating the Bjorklund algorithms as music-generating tools. This variation of the notion of Bjorklund algorithms stems from experiments I have made using these algorithms as an analytic tool for popular music, specifically the music of Damaso Pérez Prado. This Cuban musician developed key ideas around mambo music and became a fundamental part of Mexico City's popular music scene in the 40s, 50s and 60s. The principle I am basing this slight change of orientation relies on a continuous variation of thematic material common in many forms of folk and popular music that is present in the instrumental solos of Pérez Prado's music.

TimekNot allows four different notations for Bjorklund algorithms:

Simple. This notation has three arguments. The impulse value (k), the number of intervals value (n) and a rotation value (r). It looks like this: `(3,8,0)`, which will generate in simple XO notation: `xoo xoo xo`. It is possible to omit the rotation value, and 0 will be passed by default. From now on, I will omit the rotation value.

K. This notation has the same values as the simple one, but it has an added pattern that will be played in the (k). It looks like this: `(xx,3,8)` and will generate in simple XO notation this: `xxoooo xxoooo xxoo`. Notice that the spaces without impulses will have the same number of offsets as the pattern given as K.

Inverse K. This notation will allow players to assign a pattern to the spaces not occupied by (k). The notation looks like this: `'(xx,3,8)` and as XO would look like this: `ooxxxx ooxxxx ooxx`.

Full. This notation requires two patterns to be passed to the algorithm. The notation would look like this: `(xxo,xo,3,8)`. As XO would look like this: `xxoxoxo xxoxoxo xxoxo`. Something to notice in this notation is

that, because of the additive rhythmic paradigm, the two patterns have different durations (three units and two units).

As it should be evident about the rhythmic notation so far, the responsibility to manage the duration and alignment of two rhythmic ideas is on the side of the player. Thus, assuming two rhythmic structures have the same tempo, a sense of polymetric relationships is favoured. In other words, each onset or offset will tend to be the same duration. So, each group of onsets/offsets will not be adjusted to the same metric unit creating, implicitly, longer, more long-term-oriented, cycles. However, the duration of each rhythmic structure will always be determined by the polytemporal aspects of the temporal notation; it is important to keep this in mind.

Subdivision Notation

The notable exception regarding the duration of each onset/offset is the subdivision notation that allows players to create events embedded in a full-duration unit event. This notation is recursive, so it is capable of providing any kind of rhythmic feeling required by the player. This is the aspect of the rhythmic notation that opens the possibility of having a divisive rhythm, where the unit can be subdivided into any number of events. The subdivisions can be subdivided in themselves, making it possible to create specific rhythmic relations. The notation appears like this: `[xxxx]`

So, everything between square brackets will have a total duration of one tempo unit. So, in this case, each X will have a duration of 0.25 tempo units. It is possible to do something more elaborated:

```
[xx[ox][xx[xx]]]
```

This will divide one tempo unit into four units with a duration of 0.25. Subsequently the third of this will be divided into two. So, the `ox` of the third beat will have a duration of 0.125 tempo units. The last beat is divided into three beats, each with a duration of 0.083, with the last beat divided into two events of 0.0416.

The philosophical understanding of rhythm I am trying to maintain in this software tries to minimise the equivalency of duration with a rhythm that often persists in music theory since duration should be mostly changed with the polytemporal notation explained previously. Additionally, the rhythmic pattern produced via this notation is not directly bound to sound but to a 'clock' that does not necessarily interact with sound directly.

Numeric Notation

The numeric notation relies on additive rhythm but is not exactly based on the binary XO logic. It allows players to declare a duration between one or another onset using integers. For example: `3,3,2`

Would produce something equivalent to this: `xooxooxo`

This notation has a rotation value that can be invoked as follows: `3,3,2 <<5`

That in standard XO looks like this: `oxoxooxo`

In future instances of TimekNot, the XO notation will not mean precisely onset/offset. The player can assign something to either X or O. This means that both X and O could be onsets of two different sounds or audio parameters. What would this mean for the numeric notation? I believe that this notation will only be useful to generate onset values at the signaled moments of the rhythmic sequence. The 'negative

space' between the signaled moments will have no meaning. Hence, it does not precisely fit the logic of the XO rhythmic notation developed so far. The numbers that this notation takes as input can only be integers; so, in my mind, the reasoning behind it does not fully fit to that of (free) durations neither, which would open a far too removed conception of rhythm that I do not want to explore with TimekNot. The rhythmic notation responds to an interesting challenge: a form of reconfiguring the texture of time (the discretisation of that which is continuous) oriented towards sound creation but not entirely surrendered to it.

Loop / Unloop notation

The start of the rhythmic notation is separated from the polytemporal notation by a pipeline: `|`. The end of the rhythmic notation – that stands as the end of a temporal expression – is marked by either a `||` or `:|` symbol, which is reminiscent of a musical double bar or repetition bar notation. These symbols allow a program to be unlooped or looped accordingly. This loop/unloop possibility has a philosophical implication that requires some explanation. Any temporal expression written by a player in TimekNot becomes (at least conceptually) an infinite timeline; however, the possibility of instantiating it as a computational side effect (in particular, as audio) is finite. If a timeline is not looped it will play only one block of sound: the one at the head. What is relevant from this operation is to understand that the head is not exactly the 'first' block of events. It is the one that establishes the proper alignment for this timeline. Before the head, there is (at least conceptually) a series of event blocks that can be used as a reference clock for other timelines to align. The same with event blocks after the head, they exist conceptually to allow other timelines to align.

Unix Time / Embodied Time as External References and beyond

I have developed certain ideas about time-keeping in TimekNot that favour expressions as references to other expressions. In other words, I am mostly interested in the ability of the time expressions as other expressions reference clocks. However, there is always the need to have a reference that is not contained within the program; an external time reference. In the mesh of clocks I favour in TimekNot, ultimately there must be a reference to a frequency of something beyond the language's own markers.

For TimekNot, I have developed two different systems to provide such references. The first one is evaluation time, which uses the player's interaction with the software to determine a program's alignment. In the case of the language instance available in Estuary, the second one acquires its tempo from Estuary's tempo data structure. Suffice it to point out that this tempo is ultimately used as a reference to midnight of January 1st, 1970. This is a convention for Unix Time, which computers rely on to keep track of time in, among other things, networked computation. The former external time reference is the body and the player's interactions with the computer; the latter is a widely spread standard that allows synchronisation amongst players in Estuary and/or potentially other networked music/art systems.

Briefly on Acceleration

Acceleration, in musical idiom, is the term I use for accelerandi and rallentandi. However, (negative or positive) acceleration does not evoke Western/academic uses of gradual changes in speed. It is an implementation of tempo experiments prominent in Nancarrow's oeuvre. Differently from the acceleration tempo mark already implemented – that relies on the finitude of the block to homogenise

duration – Acceleration applied to looped (infinite) voices can very quickly become unmanageable in computational terms, reaching a speed tending to infinity sooner than later. This concept presents challenges to map it within a stateless computational paradigm – as the one utilised to program TimekNot – since every iteration of a block would have a different duration relying on the state of the previous block. The challenges are not only regarding how to calculate such complexity but also how to notate it in a consistent manner.

I have implemented a form of acceleration as an in-block feature – where the duration between events in a rhythmic expression accelerates). This use of acceleration implies that the convergence point is not the organising factor for acceleration but the rhythmic structure. This kind of acceleration, when looped, falls into cyclic thinking. This kind of unlooped acceleration has many limits that will be tackled in later iterations of TimekNot. How acceleration is currently implemented in TimekNot is notated as follows:

```
acc0 <- (10 afterEval) (10) (~ 1 << 0 range 400cpm, 100cpm) | xxxxxxxxxx :|
```

It is difficult to predict how these functions could affect the overall texture of the software and how this kind of 'complexity' would enable avenues towards a techno-scientific and/or vanguardist orientation of the program. So, I am advancing slowly.

The Aural Notation

The other main component of this software is the aural notation that I will describe in detail. The aural notation is meant to sculpt the actual sound of the program. If the temporal notation is meant to create the composition of the events, the aural notation is meant to produce the parameters to shape each individual sound event. Thus, whether a sound is low or high pitch, if they have low pass filters, what

sample to use, what panning, what gain, or many other characteristics allowed by TimekNot. I will follow closely the syntax enabled by the program to structure the section. First, I will explain how to attach a temporal expression to a set of sound attributes. I will explain the notions of Voice, Event, Onset and Index thoroughly to clarify this process. I will explain the selection of sound parameters I have enabled in TimekNot and the span grammar.

The aural notation has a very simple rationale. It starts with an identifier, and using the period as an operator, it will add a parameter as follows:

```
v1.sound
```

The identifier is not an arbitrary name; it should refer to a temporal expression that was previously written. For example, the previous snippet would have to be preceded by something like this:

```
v1 <- diverge | xx[xx] :|
```

Of course, the `v1.sound` example is incomplete, a computable, manually written, aural expression requires: (a) an identifier, (b) a parameter, (c) a span, (d) a list of values. This would look as follows:

```
v1.sound = _-_ “bass marimba harp cello”;
```

The list of sounds is a particular case that requires double quotation marks; all other lists do not need them. After the equal sign, there is a very mysterious symbol: _-_. This symbol will communicate to the software what kind of distribution it should apply to the sounds that need to be assigned to events derived from the rhythmic structure; from now on, this will be referred to as the span. The end of the expression requires a semicolon in all cases. Before I explain parameters, span and the list, I need to explain how the temporal expression generates useful information to process the aural expression.

Voices, Event, Onsets and Indexes

I compound aural with temporal information via a data structure called voice. A voice is a term taken from vocal music practice, where each different monophonic line is a different person (hence, a different voice). In this software, a voice is not a silent manifestation of an ongoing clock: it is a block or sequence of blocks (a layer) of monophonic sounds (or more precisely, sound samples) organised in time by a temporal expression and arranged in audio parameters by an aural expression. Thus, a voice is the intersection of a temporal and aural expression where the temporal arrangement provides a grid to arrange the aural aspects.

To arrange the aural aspects, I first need to process in a particular way the temporal expression; for this I have created the Event structure, which represents an onset and an index. The onset holds the timestamp with the instant (in absolute terms) of when is the current event supposed to be triggered as well as a Boolean that represents if this is an X or an O. The onset structure will allow me in future versions of TimekNot to do interesting arrangements like assigning one set of sound parameters to Xs or Os or create a flow of sounds to the overall structure without distinguishing singular voices. For now, I will focus on the index structure.

The index gives a position of an event in the inner structure of the voice or in the overall 'flow' of events. So, it is possible to describe the event's position in two ways: By each individual event or by a specific position in the rhythmic structure. An example perhaps could clarify this:

```
v0 <- diverge | xxx[xx] :|
```

This will generate a series of indexes for each event that is represented as a process and as a structure as follows, respectively:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, etc.

0-0, 0-1, 0-2, 0-3.0, 0-3.1, 1-0, 1-1, 1-2, 1-3.0, 1-3.1, etc.

Above, you can see the representation of two blocks of sound in a looped voice. If these events were not looped, the ‘structured index’ would stop before the first number changes to 1, and the ‘process index’ will stop at 4. Since it is looped, these counts will go on forever. This notation should look familiar as it was explained in the convergeTo and convergeFrom explanations of the polytemporal aspects of the temporal notation.

I have called the first set of indexes ‘process’ because I emphasise continuity regardless of the rhythmic structure. Thus, these indexes describe an ongoing process. The other indexes are called ‘structured’ because they describe with precision the rhythmic structure passed from the temporal expression additionally to the count of iterations of a block. The structured index starts with a number representing the number of iterations of a block elapsed; after the slash, it will describe the events in the first layer of subdivisions (the top of the rhythmic structure), after a period, the second layer, etc. This indexing system is fundamental to assigning a sound to a timed event.

Potentially, this will allow players to have one-occurrence aural events or recurring in cycles proposed intentionally and not assigned automatically by the temporal expression. In the current version of TimekNot, this indexing system is mostly underutilised.

Sound Parameters

The sound parameters are derived from Dirt, software to play audio samples with some level of accuracy, created by Alex McLean, direct precedent to SuperDirt created by Julian Rohrhuber and others, and WebDirt created by Jamie Beverley and David Ogborn. SuperDirt and WebDirt are sampling software for SuperCollider and the Browser used for TidalCycles (McLean & Wiggins, 2010) and Estuary (Ogborn et al., 2022) respectively. As TimekNot is compatible with both WebDirt and SuperDirt it is a natural fit to manipulate sound parameters in these ways. This is a list of all sound parameters that can be changed in this way:

Sound. Has ‘s’ as a synonym. It requires a list of strings. This parameter changes the folder of samples in the style of TidalCycles. This is necessary to instantiate any sound at all in TimekNot.

N. This is the sample’s index within the sample folder invoked with sound. It is a list of integers.

Vowel will add a formant filter to the sample.

Gain will modify the volume of the sample. From here onwards, all parameters take a list of numbers (either integers or decimals).

Speed will modify the speed of reproduction for the sample.

Pan will modify the sample’s panning. Mostly, it will change the position in the stereo field of the output.

Begin will change the start of playback of the sample.

End will change the end of playback of the sample.

Cutoff will add a low pass filter to the sample.

More sound parameters will be added as the software develops. There is a lack of ways to modify pitch in this list (only speed will change pitch). This is because pitch has a category that will be explored in later sections of this chapter. For now, I will point out that pitch will be enabled by a local system based on Iranian and other Middle Eastern music systems, Wendy Carlos's tuning systems, and will also have a notation for tuning systems.

Span

An interesting aspect of this notation that needs to be explained is the span notation. As I already stated, the span notation communicates how to distribute the sound values throughout the temporal structure produced by the time expression bound to the current identifier. Imagine eight events, and imagine three samples assigned to such program. The span will define how to distribute the three samples onto the eight events.

For example, one way is via cycling: 1 2 3 1 2 3 1 2 3 1. Another is by spreading: 1 1 1 1 2 2 2 3 3 3.

This uses an indexing system previously explained. Each event is uniquely identified, meaning there is a cyclical or repetitive representation of the events and an infinite sequence of them.

Currently there are six ways to distribute the values:

By **spreading** them in a block of events. Given the rhythmic structure, this will spread the sound values as evenly as possible. If a rhythmic structure has 10 onsets/offsets and is assigned two sounds, it will assign the first sound to the first five events and the second to events from five to 10.

By **cycling through events** regardless of the rhythmic structure. This will cycle the events ignoring the rhythmic structure.

By **cycling through blocks**. This will ignore the process index and everything to the right of the ``-`` in the structured index.

By **cycling through the events within the rhythmic structure**. This will ignore the process index and everything to the left of the ``-`` in the structured index. For now, this software can parse up to the first layer of subdivisions. Further experiments are required to determine the proper way to include the potentially infinite number of subdivisions the rhythmic structure allows.

By subdivision layers of a structured rhythm. This will assign a single value to a given nested subdivision of the rhythmic structure.

By proximity to any existing convergence point. This span is a particular case that exceeds the logic of the index used so far to assign sound parameters to time events. By definition, it is non-repetitive and can only work if the time expression used to create this voice is convergent (either converges with the external voice or to any other voice).

The notation of span appears difficult to follow. However, it has a simple explanation. This notation is an eccentric interpretation of a Haskell notation. In Haskell _ means that an argument of a function should be ignored as it is just a placeholder. In my proposed notation, I want to express that the player does not care about the specific iteration of an event or a block of events but what kind of index or part of an indexing system the player should consider. _-_ implies that the structured index should be minded, this notation is assigned to (1) above. _ implies that the process index should be minded, this is the notation

for (2) above. `_-` implies that the number of iterations (block) should be considered (3). `-_` is the notation used to express (4) when the sound values should cycle through the internal indexes of a rhythmic structure. The notation of (5) is `-`, where we care about the nested subdivisions of a rhythmic structure. Easier to listen to than to explain with words. The notation of (6) is simply `.` . This is more of a mnemonic device to re-orient the attention of the player to the convergence point.

If a player does care about a concrete index, they will be able to create one-time-occurrences of sound events by directly assigning them to the index as follows:

```
v0.sound = 10-2.1 “cp”;
```

Or, evaluation time can be considered:

```
v0.sound = (10-2.1 afterEval) “cp”;
```

You can also assign a specific segment using the right-oriented arrow to spread and the curvy-right-oriented arrow to cycle:

```
v0.sound = (10-2.1 -> 14-0) “clap bass harp”

v0.sound = (10-2.1 ~> 14-0) “clap bass harp”
```

(Non-echoic) Transpositions

There are other ways to assign sounding values without having to manually input them, specifically two other ways:

By simple transposition. This allows players to directly replicate an arrangement of sound parameters from another voice. The notation looks like this: `v4.sound = v0;`

By transposition with an operation. This allows players to transpose parameters from another voice with the partial application of a function. The notation is still in progress.

The transposition of individual sound parameters is interesting because it implies that the temporal structure is not necessarily identical in two voices that might use the same sound values. If a tempo canon allows the listener to enjoy tempo differences, this type of transposition invites players to experiment with the effects of all kinds of time transformations in a single sound arrangement. It is not very subtle or elegant but full of possibilities.

Higher Order Idioms and Computations

Once the temporal and aural aspects of this language are established, it is possible to think of more compact notations to refer to patterns of programming that are already common (in my brief practice with TimekNot) or are highly likely to become common. This is the layer of the program that can better describe more directly what TimekNot makes possible as a music-making technology. For now I will not describe explicitly these notations since some of them are not yet implemented or their grammar and syntax are unstable. However, TimekNot's future work would focus on instantiating these components.

Canonise

An idiom to describe tempo canons as the ones favoured in Nanc-In-A-Can Super Collider Library (Franco Briones & Villaseñor de Cortina, 2019) and TimeNot will be opened. Tempo canons can converge with other temporal expressions by convergeTo and convergeFrom. The convergeFrom will have to indicate the canonic layer to be used as a reference. The main difference between tempo canons and

other more regular temporal expressions is the tempo value that, instead of receiving a tempo, it will receive a list of tempi.

Concatenated Loop

The concatenate operator will allow players to concatenate together two or more unlooped temporal expressions and create a loop of the new structure comprised of the two. Concatenating voices provide a very different way to understand musical cycles. It reinforces the additive rhythmic rationale over the sub-divisive one. Like this idea, it is possible to concatenate a temporal expression to an algorithmically generated version of itself. The notion of a modulo should be considered as part of the possible transformations to temporal expressions.

Razgado

The *razgado* (from the right hand's guitar technique to play several strings at the same time in Spanish popular music) is a gesture that is perceived as something between an arpeggio and a chord. Nancarrow and Gann conceptualised this in a very different manner than I am doing for this program. These are fast bursts of notes that are a very fast instance of a polytemporal structure. So, the most common *razgado* can be felt as the onset just before the convergence point of a tempo canon, where echoic distance is closest to 0. Thus, *razgado* might be imagined via echoic distance from the convergence point. The *razgado* is a transversal kind of interesting structure; as I imagine it, pitch and other aural components form a list that does not happen sequentially but across temporal structures.

Angels

There is a data structure I have called Novus inspired by the poetic figure that Walter Benjamin described in his famous ninth thesis of history (1989). This function ‘linearises’ a program by keeping a projected point in time (usually after evaluation time) in the cache of the program, which can potentially create sharp inflexions. Radical changes in texture or any other form of making an instant in time charged with meaning. The notation requires a time-point expression to determine where this point will be established and then invoke in a temporal expression as part of its polytemporal aspects.

The time-point expression has three modes (lift, move, remove) and has three measuring units (date, seconds and beats (taken from the external tempo)). Date is a dramatic unit to use for a live coding performance. An arena of experimentation for this expressive mechanism is the annual Euler Room Equinox performances that make calendar dates and times meaningful enough, given the automated stage-management tools they use. By introducing date-time formatted into the performance, history makes its way to the stage.

The Case for Outsider’s Pitch

Computers have been used to experiment with various domains of music creation: experiments on spatialisation and timbre have been dominant historically. Synthesis and recording methods have been developed to compensate with (and explore beyond) the richness of instrument and voice timbre. Creating generative and algorithmic methods to formally organise sound is also salient. Rhythm has also been a major preoccupation of electronically/automatically produced music. Nancarrow is perhaps the

prime example of experimentation with automatisation and rhythm. Pitch (perhaps more correctly, melody and tuning) has seldom been explored methodically and substantially until recently.

Historically, one clear exception is Wendy Carlos with the α, β and γ tunning systems implemented to perform The Beauty and the Beast. This paragraph from her might shed some light on the issue:

> The arena of musical scales and tuning has certainly not been a quiet place to be for the past three hundred years. But it might just as well have been if we judge by the results: the same 12 $\sqrt{2}$ equally Tempered scale established then as the best available tuning compromise, by J. S. Bach and many others (Helmholtz, 1954, Apel 1972), remains to this day essentially the only scale heard in Western music. That monopoly crosses all musical styles, from the most contemporary of jazz and avant-garde classical, and musical master pieces from the past, to the latest technopop rock with fancy synthesizers, and everywhere in between. Instruments of the symphony orchestra attempt with varying degrees of success to live up to the 100-cent semitone, even though many would find it inherently far easier to do otherwise: the strings to "lapse" into Pythagorean tuning, the brass into several keys of just intonation *(Carlos, 1987)*.

This is a crucial matter, given the characteristics acquired by Western electronic music. Electronic music tends to be culturally specific to European academic or market-oriented music. A telling manifestation is the lack of representation of tuning systems – other than the 12 equal temperament chromatic scale and its derived modes – in electronic music instruments and production spaces. I have chosen to base the pitch manipulation in TimekNot on Middle Eastern tuning and pitch organisation systems as default and standard. I will support and develop a series of ideas on tuning and pitch creation based on Erv Wilson's work. I will start this exploration with Dastgahs, which I have become familiar with through conversations

with Mehrdad Jafari Rad and Aida Khorsandi and Combinatorial Product Sets (CPS) based on the library developed by Diego Villasenor de Cortina. I will implement a repository of scales that can possibly be invoked as the Dastgahs are. Additionally, I have implemented the α and β scales developed by Wendy Carlos as part of this software.

I will implement a notation that accurately represents the Dastgahs and allows players to play a synthesis of the style of Iranian traditional music and the affordances of TimekNot. CPSs have various advantages that work well with polytemporality: a CPS is a pitch set resulting from the product of factors in a set. In simpler words, the player must produce the number of factors that can be multiplied at once and then take the product of all these multiplications and adjust it into a period (usually the 1:2 ratio (the octave)). For example, having a size 2 and the factors 1, 3, 5 and 7 will use the products of 1*3, 1*5, 1*7, 3*5, 3*7 and 5*7 adjusted into an octave. So, this scale will have the fifth (1*3), the third (1*5), the seventh (1*7), another more complex seventh (3*5), a complex fourth (3*7) and a complex second (5*7). A CPS is a pitch set that can be applied in different layers as subsets. So, it should not be understood exactly as a scale. For example, just using the products derived from factor 3. Or the union of 3 and 5, their intersection or any other operation related to mapping. Using subgroups, as referred to in the last sentence, has quite an organically polyphonic logic that works very well with the temporal aspects of TimekNot. CPS can be harmonised easily with a diatonic scale of 12 ET (in case relating with other languages), or, in any case with many other Dastgahs.

I am experimenting with different interesting concepts I have found either in the tuning experiments inspired by Erv Wilson and based on Diego's research or in the Dastgah system. (1) The concept of Moteghayyer (or 'changeable' note) where a note is tuned differently depending on the melodic direction

it might take. (2) The idea of different octaves having different interval arrangements based on the same tuning system. (3) Subsets of tuning systems and how they may interact in canonic situations: the index of the canonic voice determining which subset to express as pitch. My comprehension of these systems is still unstable so I will adjust the code, syntax, notation, and functionalities according to my ongoing research.

The CPS notation is a bit complicated. It requires a pitch expression to be written and then invoked in the aural expressions. The pitch expression's syntax mimics the temporal expressions, but it is surrounded by curly brackets. These curly brackets are not necessary, but the player needs some way to differentiate cognitively between expressions. A pitch expression could look like this:

```
{ myPitch <- cps 2 (1 3 5 7 11) }
```

This expression will allow the player to invoke `myPitch` in an aural expression as follows:

```
v0.myPitch = _-_ 0 2 4 6 5 8 9;
```

I hope to represent the cultural disparities expressed by the multiplicity of pitch systems in the notation of TimekNot as productive and open a space where traditional tuning systems bypass Western classical tradition to foster the possibility of change without the mediation of the market or dominant cultural institutions.

The Standalone and the Score Widget

Due to the complexity of the concepts explored in this software, I have decided to create a stand-alone that relies on a piano-roll visualisation. This visualiser will also operate as an animated score widget

capable of offering a map for an acoustic instrumental ensemble to synchronise with an Estuary (or, more generally, cybernetic) ensemble.

The goal of the standalone is three-fold: (1) to act as a pedagogical tool for people to understand the idea of general polytemporality I am proposing; (2) for this software to communicate a set of philosophical principles regarding time and musical creation that can be discussed as embodied theory; and (3) to attempt to bring down the first wall of digital music: the ability to coordinate with acoustic instruments in a post-disciplinary and non-hierarchical manner.

The visualisation of the standalone and the score widget is an easy graphical way to understand how time-keeping happens in TimekNot by representing time events (without any aural attributes, unless the user wants to map them into the visualisation).

The first two purposes are simple: I want to create conditions for others to take advantage of the software by understanding its rationality, and I want to discuss how digital music-makers conceive time relations in a productive manner aided by an interactive representation of musical time. The third one is an attempt to overcome a characteristic of live coding that is difficult to understand: the reliance of multiple binaries that live coding relates with in a difficult manner: elite/vernacular, audio/visual, presence/representation, abstract/embodied, instrumentalist/composer, artist/musician, acoustic/electronic, digital/analog, etc. Drawing extensively from Sichio's work on scoring systems for choreographic live coding (Sicchio, 2024) and other experiments on non-computarised coding (Torres Núñez del Prado, 2022) I am trying to generalise TimekNot to produce time relationships that are not exclusively for the generation of audio but also for the generation of (real, human) movements on the existing world as well as visual signals. I want to eventually have a TimekNot that can be completely mute – a TimekNot modality that describes a time

structure that function as a score for guiding compositions or improvisations – so the fracture between the human and the computarised can be amended by enabling the additional possibility of computers only computing while humans make (all of the) art when necessary or when desired.

Conclusion:: TimeNotation -> TimeNot -> TimekNot -> TimeKnit

TimekNot, as a networked language in Estuary's ecology, standalone and score widget implicitly move against the *zombiefication* of electronic music that comes with AI frameworks without rejecting the challenges and benefits of automation. I will talk further about this in the last chapter of this dissertation. It is an incomplete programming project that assumes that others and other cognitive territories play a role in its wholeness. It does not require artificial agents to be completed, but people, their ideas and desires. In other words, TimekNot understands its collaborative and multilingual impetus and the (human) joy of creating. This language is not in competition with other languages but within a relationship of mutualism. The expectation is for almost every TimekNot performance to be a performance with Tidal, Punctual, Seis8vos, CinerCer0, Hydra or any other language in Estuary with which it shares a niche. TimekNot can develop into incompleteness beyond the ecology of Estuary to propose languages with composition capabilities, languages that can afford concrete computational relations with other languages. Additionally, the language acts as an instantiation of a musical time theory and as a scoring device, it extends its life towards the realm of 'acoustic' music and cultural reflection without relying on the dominion of Musicology, Music Composition and Philosophy.

TimekNot attempts to feel like a knot for the performer: tense and complicated, like a crisis perhaps. It is supposed to feel like a suspension of normal temporality and into an anomalous time of possibilities. A

space where the players can perform the undoing of a stream of information that is supposed to be disorienting, and with no easy resolution. Thus, TimekNot, instead of facilitating the musical process, is supposed to complicate it to thrust the players into a state where new forms of music creation can be experimented, explored, imagined, performed, and/or rejected. TimekNot is a notation, a negation of common time and a tense knot of temporalities.

Introducing TimeKnit

A feature of this language will be implemented that further complicates players' relationship with music and experiments with the affordances of networked computation and Estuary's performance space quite broadly (this feature will have no consequences in the standalone version). A mode will be enabled in the language, allowing players to access specific data structures built by other players in the context of Estuary's editor zones. So, if a performer creates a clock identified as `mainClock`, another performer can create another clock that converges with this one or even directly use it to create an aural expression. A key question is, can this idea be extended to support other languages to concatenate to TimekNot in a similar way?

The ability to access other people's expressions – and make them their own – on-the-fly will change the dynamics of a live coding networked ensemble: from a set of individuals playing together in a positive manner to a tense negotiation of the musical event where the new is the result of accords between players. This means that to advance and change as a group, every member must reach tacit agreements on when, how and where to go. Hopefully, an intuition of commonality might emerge from such experiments that adumbrates a music-making moving towards mutual aid and communism as cultural

practice and political economy. To enable the mode here described, the players will have to write at the top of their program:

```
###free
```

Chapter 3 - The Collapsed Factory: Converging Crises as Critical Periods

In this chapter, I explore in theoretical and technical terms the audiovisual work *La Fábrica Colapsada*. This artistic piece I composed and performed is the major artwork of my doctoral research. It is a 30-minute-long operatic, networked piece made with visuals, multichannel audio, microtonal tuning, polytemporal rhythmic structures, fixed media, reactive algorithmic music in the form of improvisational semi-analogue synthesisers, and live coding accompanied by a website where the storytelling happens asynchronously. The artwork is also constituted by multiple online repositories where the multimedia materials of the opera can be accessed and various pieces of software can be repurposed for geologic seismic data processing of the 2017 earthquake, wavetable synthesis instruments creation, and spatialization in networked environments.

More importantly, *La Fábrica Colapsada* is an opera that has been created as an artefact capable of producing new forms of knowledge and sensibility regarding crisis and time by observing – through the lens of research creation – the earthquakes that occurred in Mexico City on September 19, 2017, as well as on the same date but thirty-two years prior, in 1985. I claim that through this artwork, I have been able to re-envision 'disaster' from a social reproduction perspective supplemented by cybernetics and cultural critique. Emerging from this analysis as guiding concepts to navigate crisis and re-imagine what time can be, I have coined the terms 'toxic resilience' as well as 'generative' and 'reactive' forms of resistance.

Building upon this re-conceptualisation of disaster, I have constituted, along with my collaborators Iván López and Diego Villaseñor, *Pirarán*: a networked music ensemble rooted in care and mutual aid. A political economy analysis of current music creation shows that many ideas on disaster around the earthquake presented here can be extrapolated to consider the global music scene a disaster. Namely, I propose concepts like toxic resilience and reactive and generative resistance. Nevertheless, our proposal of a networked music ensemble demonstrates that it is possible to rearrange technological means – re-shaping them as technologies of the non-self – to move away from disaster towards communism, as observed during the earthquake. We have done so by rejecting 'toxic resilience' and embracing generative and reactive forms of resistance found and explored as the creative process and research behind this opera. Similarly, I have developed a concept of music that escapes vanguardism and popular modernism by signalling towards a culture of 'algorithmic acid communism', a term inspired by Mark Fisher (2017) and characterised here by its orientation towards the profound, substantial transformation potential of people and collectives. From this analysis, it is possible to imagine network music and live coding tools as technologies of the 'non-self' proposed by Gilbert (2014).

Context

This artwork draws extensively from *Temazcal 2*, the live-coded documentary described in Chapter 1. Both works utilise Estuary – the networked software for collaborative, multilingual live coding – as an interface and stage. Both works have produced small but profound and meaningful collaboration spaces. Both works are transnational in nature. In many ways, I have based *La Fábrica Colapsada* (LFC from now on) on many techniques, concepts and notions built, explored and invented for *Temazcal 2*. The two works are forms of appropriation and re-signification of well-established, hegemonic art forms and

works. The most relevant aspect that characterises both works is the development of the networked ensemble as a site of mutual aid and care, an attribute that I consciously developed for LFC. As this chapter traces multiple converging crises, the settler-colonial impulse, and the housing crisis, confronted by the people of Mexico City described in the theoretical section of Chapter 1 converges with the earthquake and the disaster described in this chapter. Furthermore, the concepts of triple movement and double consciousness developed in Chapter 1 are the subjectivities and processes that inform the storyline of this opera.

There are various key differences as well: Temazcal 2 was a co-creation with the sound artist and curator Rolando Hernández and it was performed by the two of us and Diego Villaseñor – musician, philosopher, programmer, and an old and constant collaborator of mine – in a more interpretative role while LFC is solely my composition that relies substantially on the imaginative and creative musical improvisation skills of Diego Villaseñor and Iván López, a brilliant synthesiser performer, percussionist, composer, scholar, and audio engineer living in *Morelia*, Mexico. The broader projects involved in both works have major differences: UHM – Rolando and my networked art project – was a research unit on time and cosmovisions and *Pirarán* – Iván, Diego and I – form an algorithmic acid music networked ensemble. *Temazcal 2* is a documentary where interviews with key people are the core of the work. At the same time, LFC – even though it uses real stories drawn from secondary sources – is an opera that opens a sensual and embodied contemplative space that supersedes realism. LFC is a music work (even in its non-sonic components, aurality is key) while *Temazcal 2* was always understood as a cinematic work whose narrative is profoundly influenced by counterpoint techniques.

La Fábrica Colapsada is the site where the most intense explorations for TimekNot – the software described in Chapter 2 – took place. Such explorations included more than 20 two-hour-long sessions with the other parts of the ensemble playing either other live coding languages, all of us or two of us playing TimekNot, or – more regularly – Iván and Diego playing the synthesisers with me playing TimekNot. TimekNot afforded the ensemble a certain freedom from regularity and precision that other live coding languages would not favour, particularly in latency-rich contexts like the networked ensemble. The texture I could create with TimekNot was often described as an irregular three-dimensional sonic space where synthesiser performers could freely focus on any melodic line and shift their focus to another, re-orienting their performance direction. In this way, TimekNot became the basis for the ensemble's improvisations in a way that motivated new forms of articulation and rhythmic plasticity. In other words, Iván and Diego developed a synthesiser counterpoint practice that mimics the affordances of TimekNot, and TimekNot has substantially developed its most simple core possibilities as a two, three or four-voice radical counterpoint (or multi-monophonic) sonic machine.

The infrastructure of the opera can be broken down into multiple parts, and its vastness will be thoroughly explained later in this chapter. Iván and Diego's technical setup is beyond this dissertation's scope. However, suffice to mention that both programmed multiple synthesisers and enabled complex audio environments based in DAWs and all kinds of synthesiser software. The political economy of their sound is partially open-source and free software. It is important to state that the software used to create stream-in their synthesiser improvisations is a paid service for professional audio production, popular in the market. It is a neo/techno-feudal technology, as it is rented, opaque, and with mostly good reviews but for the needs of *Pirarán* it appears as glitchy and sometimes unreliable, at least for the price we

monthly pay for it. Yet, it simplifies an otherwise difficult and uncertain task, especially for people without in-depth programming knowledge. In the future, we aspire to utilise networking technology that will allow us to overpass this issue.

For this work, I developed two major sections: the overture and the acts. The overture is highly structured and unusually prescribed. The three acts are musical free improvisations that, taking advantage of the concept of polytemporality, flow continually and without specific differentiation. Only in the interactive (the website) and visual components do the acts appear linearly and can be followed rationally.

I conceived this artwork in the early COVID-19 pandemic. I attempt to observe Mexico City's experience with earthquakes as a preamble and contrast to the lockdown suffered globally, and personally observing as a PhD student in Canada. Similarly, it is possible to observe that the earthquake aligns with the COVID-19 crisis in the opera's storyline. The COVID-19 lockdown is also the starting point of *Pirarán,* as the crisis pushed musicians into an unanticipated situation of unemployment and risk. In some ways, this crisis marked my doctoral process irrevocably, and it should be understood as a period where past and future crises converge.

It is also important to position myself as a precariously employed, male, not-Indigenous-nor-white (as described in Chapter 1) subject at the time of the earthquake, who used some of my time caring and participating in mutual aid initiatives in the aftermath of the earthquake. I witnessed first-hand the psychological, emotional, economic and physical pain and anxiety endured by people in the city. In many ways, this dissertation has allowed me to make sense of what I experienced personally in such a crisis.

I conceived this work 15 years after the collapse of the global economy during the financial crisis of 2008-2009 that fractured progressive neoliberal hegemony (Fraser, 2019), inaugurating a major crisis; more personally, as a result of the global crisis of 2008-2009, I experienced the loss of my family home and the shattering and dissolution of my family core, thrusting the four of us into three different countries and four different cities. Since then, I have been reading freely on the internet and in libraries in Mannheim, Newcastle, Mexico City, Toronto and Hamilton trying to understand why my mother lost the home she built with so much love, why my father had to confront the police protesting the loss of his job and pension by financial speculation, why my brother's children have a native language other than his father and mother. Why am I here? How can I heal what needs to be healed, and what is it that I have already healed and allowed to flourish?

However, this artwork's representation mechanisms are precisely not only about me, my skills, or my cultural context; it centres people who have endured extraordinary forms of dispossession, sharply contrasting with their unique ability to challenge the forces that attempt to destroy life. The people that inspire this work are immigrants, women, Indigenous Peoples, queer folks, and workers. Let it be clear that these vectors of identity are not a generic list of left-leaning figures, but the actual concrete subjects responding to an earthquake in ways that are described in the opera as emancipatory. With this work, I attempt to observe the ways in which others have faced dispossession and learn how to face it instead of solely enduring or enabling it in my most immediate environments: academia and art institutions. As it will be clear by the end of this chapter, the experiences and teachings of the earthquake can affect music creation in unforeseen ways. In a more or less confusing manner – in the era of selfies – I am attempting to erase my presence and the presence of *Pirarán* as a participant of a profit machine (the music

industry) and hierarchies of domination (music institutions) ultimately abolishing both of this spaces so a music practice that collapses exchange, consumption and production may flourish, turning art practice into life abolishing work altogether.

Earthquake Facts and Ideas

There are some facts and ideas regarding the September 2017 (19-S II) earthquake that need to be understood before developing the analysis of the artwork. On September 19th,1985 (19-S I), an earthquake of 8.1 degrees on the Richter scale shook Mexico City for 120 seconds (Poniatowska, 1995). The aftermath was overwhelming: 412 buildings collapsed, and 5728 were significantly damaged. The death count remains uncertain: according to the Mexican Secretary for National Defence, 2000 people died in the earthquake; the National Health Institute claims these numbers are more likely between 3000 and 6000, whereas some journalists count over 20,000 fatalities. The magnitude of the tragedy surpassed the Mexican State, yet it rejected help (intervention) from the international community, except for a new International Monetary Fund loan (Ramírez de Garay, 2023), of course. The inability of the government to act in the aftermath of the earthquake activated a network of solidarity among citizens that became the basis for a civil society and would become key for future political changes in the city and the country (Leal Martínez, 2014).

Around 1 pm on September 19th, 2017, two hours after the ceremonial evacuation drill that commemorates the 1985 earthquake, an earthquake of less intensity (7.1 degrees Richter) but with an epicentre closer to the capital of the country shook Mexico City's people again (SHCP, 2017). Around 50 buildings collapsed, and more than 300 people lost their lives. The earthquake was not as deadly or intense as 19-S I, but it was certainly the most destructive since then. Immediately, people started

gathering around what was perceived to be the most affected areas of the city to help: bringing in useful items like first response medical equipment, tools for debris removal, water and food for people working at the sites. Facing the communication breakdown, collectives of bike riders organised to transport information and resources from one location to another; hotels facilitated rooms for people that lost their homes; architects designed manuals for people to check if their homes were in danger of collapsing; audio engineers used their equipment to listen to possible survivors underneath the debris of some buildings. From all the interesting and heartbreaking stories one can read about the earthquake, I have omitted the ones that captured the people's interest the most, at least some weeks or months after the earthquake. Most of these stories are characterised by an old familiar pattern: the market and state failing to ameliorate the crisis while an emerging civil society succeeds at disaster relief relying solely on resilience. In these narratives, civil society is an unexpected agent that, out of nowhere, saves the day.

The origin of such disaster relief storytelling in Mexico City's context is 19-S I, when the institutions of state-managed capitalism were giving way to the neoliberal order. Historic conditions were diametrically different from what we have nowadays. However, 19-S II occurred in the context of neoliberalism's winding down and something else emerging in its place. Hence, new patterns are needed to understand what occurred in September 2017.

I have prioritised the stories that shed light on the earthquake's transformative potential, considering the interplay of three major social forces: the market, the state, and civil society. Considering these three social forces, a more precise vision of autonomy in an era of retreating states and advancing corporations becomes evident. The role of civil society in this triangular motion can either align with the state efforts to concentrate wealth and power into the hands of the owners of the market by taking functions away from

the state furthering its existential crisis or re-imagine their role all together so civil society can rule over the state and the market effectively re-embedding the economy into society (away from financialisation) and transforming society's mediations via emancipation (away from state oppressive structures). According to the research presented, the stories I tackled in the opera appear as examples of the latter.

The first act traces the formation of the 'Casa de los Pueblos y Comunidades Indígenas 'Samir Flores Soberanes'' as an aftermath of the earthquake. This house was formerly known as the Instituto Nacional de los Pueblos Indígenas (INPI), a state institution involved in the development and progress of Indigenous Peoples. In an act of revelry and challenging the state and the expanding housing market of the city, the Otomí migrant community resident in Mexico City took and transformed the building into a space for political organising against dispossession, particularly housing-related. These organising efforts are rooted in the earthquakes observed in this artwork (Vilenica et al., 2023).

The second act narrates the efforts to rebuild San Gregorio Atlapulco after the earthquake and all the issues that emerged from it. Specifically, the land ownership regimes and water supply are currently contested in the area. The act follows the earthquake's aftermath and later role in the organisation for the defence of water and territory against state's attempt to change the land's status and re-route water supplies from the town to wealthier parts of the city (Catrip et al., 2018; Castañeda Gutiérrez et al., 2018; Saffon Sanín et al., 2019).

The last act revolves around a collapsed building within the textile district of Mexico City that appears quite reminiscent of a 19-S I known story (Borzacchiello, 2017). The building had a textile factory and offices owned by Argentinian-Israeli and Taiwanese-Paraguayan business owners. Both lived off the exploitation of immigrant women from Asia, Central America and other parts of Mexico (Turati, 2017;

Lagunes Huerta, 2017; Ramos & Guerrero, 2017). Under the direction of business owners, the state discouraged survivors' rescue at this site to protect the interests of transnational capital (Villeda, 2018). The effect of this attempt to suppress disaster relief mobilised feminist collectives that – fueled by the memory of a similar incident in 19-S I – slowed down, and made visible, the state's attempt to cover up the factory's ill-management and transnational capitalism's inhuman practices (Satizábal & Melo Zurita, 2021).

Artwork and Chapter Sections and Ideas

The artwork is divided into two major sections: overture and acts. The overture is a series of strata functioning as different lenses to observe the earthquake. A layer that affects the body by exposing it into vibrations with a particular sense of danger, another layer can be understood as a form of social energy that affect our sensibility of location and belonging, another layer functions as a way to gaze into the instruments of objectivity in an attempt to distinguish the earthquake event from its social repercussions, the next layer is the rupturing event that traverses various layers of meaning of this work, and finally, after the rupture event, three juxtaposed acts appear where the ensemble performs music via a set of new social relations: a desired and necessary mode of production yet-to-exist on our timeline.

Similarly to the earthquake's disaster, music industry's current disaster – that will be analysed better in the last section of this chapter – has unleashed a crisis for music making that presents a similar pattern: the synthesis of old cultures combined with new technologies alter profoundly art-making processes so artists respond to it through resilience. However, resilience (sometimes named mutual aid) without class struggle deepens the corporation's power and strips the state of its capacity to mediate in favour of its citizens. The mode of production that we are proposing as an ensemble emerges from the earthquake

experience: a music that seeks to abolish art markets and the institution of art altogether by collapsing production, consumption and exchange within our practice.

The stage, then, becomes something more than entertainment. It is a space where the affordances of our proposed mode of production can be envisioned and negotiated. Against a recurring mis-conception of Attali's notion of noise as political economy that has made its way into live coding, where music style and artistic thought shape production and reproduction, I will argue that a soft dialectic interplay is formed between the stage and life where life and concrete experience is always at the starting point of any process. Thus, traditional idealistic notions in the arts – such as vanguardism versus popular modernism, rock versus pop, and high versus low culture – become irrelevant as long as everyone participates equally in the market and institutions. Similarly, on-stage expressions of queerness, Blackness, decolonial impulses, cyborg feminism, non-human agencies, and similar themes become meaningless if, off-stage, the environment is dominated by (toxic) resilience, competition, self-promotion, individualism, bullying, and other alienating behaviors inherent to capitalism. As established in Chapter 1 through Temazcal 2, whiteness and imperialism (alongside heteronormativity and patriarchy) are intrinsically tied to the rationality of capitalism. To be non-white, in some way, is to engage with a life beyond capital's scope.

From this perspective, our ensemble's transition from groove-based music to what might be considered avant-garde styles remains an act of agency, provided it remains faithful to the social relations grounded in care and mutual aid that we advocate. Similarly, this work is an opera very loosely, such reference is meant to de-stabilise the reception of this artwork, keeping-it at odds with commodification and institutionalisation by undermining and mining (appropriating) the term.

LFC operates at two different layers: as a theatre and as a factory, as live coding often does (Franco Briones, 2022). It is a theatre because it displays an archive of images and narrative lines that tell a story and induce emotional and intellectual dialogues with and within the 'spectators.' It is a factory because it is the site of a particular set of music tools, infrastructure, skills and styles meant to be shared with the 'spectators' so they can create their own music. The ideas shaping the opera are received as representation and performance. The schizoanalytic model – where desire is machinic production and circulation rather than repression and lack – is useful to understand the basic operation that I use to connect materials: concatenation. Concatenation allows me to put together various heterogeneous components: social reproduction & disaster studies & cybernetics & research creation & cultural critique & […]

The structure of the rest of the chapter transitions from the theoretical aspects found by the research and performance of the opera to the description of the work and its ties with theory. This is done twice: first for the overture and the representational aspects of the work, and then for the performative aspects of the work, mostly the three-act section. The 'Critical Period' section is a theoretical discussion on disaster that stands as the interpretation of the opera's narrative. More precisely, this first section stands as a translation of the opera as an artwork to critical theory: both, the section and the artwork are reiterating the same ideas but in different 'languages.' The section 'La Fábrica Colapsada' describes some of its more relevant technical details, the opera's narrative mechanisms, and how these are bound to the previous theoretical section. The section 'Art(work) in the Net(work)' traces the way in which the theoretical insights foregrounded in the 'Critical Period' section – combined with the mirroring of the

performance and research of the opera described in 'La Fábrica Colapsada' – have shaped the practice of *Pirarán* vis-à-vis the current state of the music industry and institutions.

Critical Period: Disaster as (Temporarily-)Situated Knowledge

War and colonialism are at the root of our understanding of natural disasters. Perhaps this reflects the schism between nature and culture pervasive in Western forms of knowledge, or it appears as a commentary regarding the alleged origins of social organisation, where social conflict is marked by humanity trying to escape the over-determination of life by nature. But war and colonialism have created the conditions for events like earthquakes to be excessively lethal, and the response to them remains ineffective, at the very least.

After World War II, the approach to natural and human-made disasters was largely shaped by strategies derived from the knowledge acquired by the US military during their missions to bomb cities such as Dresden, Hiroshima, and Nagasaki, among others. From an American perspective, *"[...] the experience of war became the template for our perception of the most diverse kinds of natural and man-made disaster* (Illner, 2021)*."*

The cybernetic principles developed for the war efforts became the paradigm for understanding system responses to disruptions of meta-stability. Through the principles of command and control, the state sought to direct and predict the behaviour of people and other components involved in rupturing events. According to this perspective, a disaster is a sudden and transient disruption of an otherwise calm, stable social existence. This paradigm thus emphasises a particular sense of time: normality interrupted by a salient event.

Early literature on natural disasters (Quarantelli, 1978) , while integrating a more nuanced view of disasters beyond the command and control framework, still oversimplifies the role of the social fabric by focusing primarily on the 'nature' that produces the salient event. As Illner (2021) notes, “[t]he effort to denaturalise nature, in other words, gives rise to the naturalisation of society” (p. 4). Subsequent approaches, relying on the concept of vulnerability, aim to challenge the centrality of the event itself and incorporate the systemic conditions that enable it (Hewitt, 1983). Since the 1980s, a more materialistic understanding of disaster has emerged, framing disasters as results of social actions and processes. This perspective introduces a longer sense of time into disaster studies. The shift towards explaining disasters through social factors has significant political implications. Approximately 75% of disasters occur in the global south (Bankoff et al., 2004; Scarlett, 2022), regions that have been underdeveloped and overly exploited. Class, gender, and race are central to the vulnerability approach to disaster. This perspective profoundly challenges the naturalisation of earthquakes, floods, superstorms, hurricanes, and other phenomena, and this challenge is a fundamental aspect of the artwork described in the next section.

In the case of disasters in Mexican territory, the origins can be traced back to the imposition of the colonial order, which “[…] disrupted the balance with the 'generous land' (Alcántara-Ayala, 2019)” as pre-Hispanic relations with the land, water, and resources gave way to European rationality. The first recorded natural disaster after the imposition of the colonial order was the flood of Mexico City in 1555. This flood was caused by the city's construction atop a lake, radiating outward from the island where the *Mexica* city of *Mexico-Tenochtitlán* once stood before being buried under cathedrals, banks, universities, mental asylums, and military fortifications. Where there had once been floating, human-made islands for harvesting the city’s food supply, there were now baroque buildings, farmland, and pasture. Seventy-five

years later, the 1629 flood saw water levels rise 2 metres above ground after 40 hours of rain. For days, survivors in Mexico City held mass atop buildings, praying for the rain to stop. This disaster led authorities to intensify their efforts to drain the lake, which, after centuries of continuous effort, altered the soil's properties: it became porous and gelatinous, amplifying rather than absorbing vibrations.

Gradually, with its vertical growth, the city became vulnerable to earthquakes and ongoing flooding. Despite significant damage from a 1950s earthquake, it wasn't until the earthquake of September 19th, 1985, that the Mexican state intervened in disaster prevention by establishing the Sistema Nacional de Protección Civil (National System for Civil Protection, SNPC) in 1986. This system aligned with the shift in disaster studies to understand the exposure of different populations to systemic vulnerabilities, as proposed by Hewitt. Following the September 19th, 2017, earthquake, there have been serious attempts to reform the SNPC, renaming it the Sistema de Gestión Integral del Riesgo de Desastres (GIRD, Integral Administration of Disaster Risk). The reformed system aims to enhance efficiency, equity, comprehensiveness, transversality, co-responsibility, and accountability (transparency and oversight). It is intended to"constitute the axis of a transformation that allows addressing the root causes and conditioning factors of disaster risk, with the aim that institutional efforts are not merely directed at responding to emergencies or promoting fragmented reconstruction actions that do not contribute to reducing vulnerability."

Vulnerability studies are broadly influential in approaching disaster, both in Mexico and elsewhere. Despite the significance of shifting towards a social conception of these events, the field remains constrained by a philosophical understanding of time in which the rupturing event remains central.

> *"If vulnerability scholars have conducted in-depth analysis of unequal social, political, ecological and economic conditions, they have paradoxically limited the impact of their rich, structural studies by always relating them back to the exposure to a momentary disaster, now called hazard. They have thus held on to the normative idea of a more or less stable everyday state that is impacted by a sudden disruption" (Illner, 2021, pp. 15).*

Recent literature offers a post-materialist perspective on understanding disasters, drawing interesting insights from this viewpoint. For example, Quarantelli's distinction between emergency, disaster, and catastrophe – each with different magnitudes of disruption (Quarantelli, 2006) – is re-evaluated. Zhang's reconceptualisation (2023) differentiates catastrophe, an interstice where structural change is negotiated through varying degrees of violence and chaos, from disaster, an event that disrupts normality but eventually dissipates after a period of urgency and crisis. This distinction is particularly compelling.

Speculative imagination provides a rich approach to disasters. For instance, exercises in downward counterfactuals – imagining past events occurring differently or evolving into something else – serve as a valuable philosophical experiment that offers a new perspective on disaster (Woo, 2023). Emerging from financial imagination and a fascination with speculative futurity, this perspective creates a specific temporal arrangement: one that partitions concrete human experiences from abstract time scales bound to geology or other non-human phenomena (Bauer & Malik, 2023).

The following fragment summerises very well some post-materialist thoughts on disaster:

> *"The 9/11 commission report concluded that the most important failure on 9/11 was a failure of imagination. One way of avoiding imagination failure is for strategic analysis to be more holistic*

> *and inter-disciplinary, connecting right brain with left brain; linking humanities with sciences. Reimagining history through exploring downward counterfactuals is an inter-disciplinary research agenda, blending psychology with physics. This agenda promotes risk awareness, and so contributes to risk preparedness, and enhancing societal disaster resilience" (Woo, 2023, p. 113).*

Nevertheless, the opera has allowed me to think differently from the perspectives often favoured by scholarship at the 'intersection of art and science' or that claims an inter-disciplinary status, typically spanning the 'humanities and sciences.' The opera aligns more closely with a social reproduction perspective, integrating art research, cultural critique, and cybernetics. Instead of relying on inter-disciplinary frameworks, LFC is committed to schizoanalysis, unorthodox dialectics, and computer science theory.

Post-material research has apparent limitations. It is worth questioning the partition between catastrophe and disaster in a world where these concepts are closely enmeshed, as observed in the opera's analysis. The assumption that the humanities or art automatically produce critical perspectives on technocratic blind spots is inadequate; often, the humanities act as the handmaid of capitalism by focusing critique on banal or irrelevant aspects, while art can produce misleading visions of technology that obscure the social relations it engenders. Art and humanities are often instrumentalised to humanise capitalist inhuman practices. Moreover, the reification of disaster as an event that breaks normality and should be met with resilience serves to naturalise capital's violence on people's everyday lives. I argue that, under the current mode of production, suffering is systemic, and ruptures in 'normality' present opportunities for humans to experience agency. Therefore, while everyday life may be catastrophic, earthquake disasters can reveal pathways to overcoming these conditions.

Social Reproduction and Disaster

Social reproduction theory seeks to understand the conditions in which the worker is reproduced under capitalism; it "displays an analytical irreverence to "visible facts" and privileges "process" instead" (Bhattacharya, 2017, p. 2; Ferguson, 2020). One particularity of labour power as a commodity is its non-capitalist form of reproduction. Labour power reproduces itself by non-capitalist means. If Marx's Capital concentrates on understanding how labour produces value, then social reproduction theory attempts to understand how labour is reproduced. As understood by Fraser (2022), the site of analysis of social reproduction theory points to the background conditions contrasting what is deemed central in Marxist analysis: the family, unpaid work by women, or immigrants, institutions, and many forms of state support like pensions, childcare, etc. Concentrating on the background conditions of capitalism will also lead us to understand current forms of alienation beyond the alienation from labour that is the focus of Marx's Capital: alienation from our human peers and alienation from nature. Illner proposes to tackle disaster studies from a social reproduction perspective that integrates the work of civil society as a fundamental aspect of social reproduction for the reproduction of labour.

Aligning with Illner's social reproduction analysis, I am posing some fundamental questions around the idea of disaster: How can we distinguish the effects of an earthquake from the disaster of everyday life under capitalism? Consequently, questions emerge around meta-stability as an applicable cybernetic principle for understanding networked governance and contemporary forms of capitalism. How are networked music ensembles capable of subverting the current recursive disaster hell we are forced to inhabit? For now, the analysis of the earthquake induced by the composition and performance of the opera continues.

To supplement the perspective of disaster studies, it is necessary to consider disaster capitalism, a term coined by Naomi Klein that describes transforming a crisis into a profit opportunity (Klein, 2007). This often involves private contractors handling reconstruction efforts, altering territorial distribution or land ownership after reconstruction, or leveraging the disorientation and shock provoked by a disaster to impose or deepen a neoliberal order without pushback from the affected people. However, disaster-caused destruction cannot always be directly linked to profit, as privatised reconstruction efforts are costly and difficult to administer and manage. In other words, direct capital is not often involved in disaster relief because it cannot benefit sufficiently; instead, the state, always acting in the interest of capital, is more closely related to disaster relief. On a different scale, disaster is often commodified as a form of tourism (Martini & Sharma, 2022). This form of commodification is probably not an issue in the context of earthquakes in Mexico. However, it signals a significant problem with disaster: the inclination to banalise or romanticise it. Artworks like the one discussed here face such a risk.

The 'shock' proposed by Klein resembles the 'state of exception' described by Agamben (2005). The state of exception describes a suspension of normality in the face of unprecedented situations, granting governments unparalleled power and freedom to act, liberated from the limits of legality. Within this exceptional time, it is often the case that the state provides, in the name of capital, the means to safeguard and expand infrastructure, variable capital (people) and fixed capital (means of production), not necessarily in any order of importance. Agamben's ideas on exception have taken an extreme form in the face of the COVID-19 pandemic:

> *"Professors who agree — as they are doing en masse — to submit to the new dictatorship of telematics and to hold their courses only online are the perfect equivalent of the university*

> *teachers who in 1931 swore allegiance to the Fascist regime. As happened then, it is likely that only fifteen out of a thousand will refuse, but their names will surely be remembered alongside those of the fifteen who did not take the oath" (Agamben, 2020).*

As a form of disaster relief, telematics is understood by Agamben in the context of universities, undermining the pandemic's biological reality, to disarm critical thinking in the face of the most pervasive capitalist rationality. In contrast, Bratton's defence of positive biopolitics and global government, which suspiciously echoes of nation-statehood on a different scale, stands fiercely critical of Agamben:

> *"Agamben's pandemic outbursts are extreme but also exemplary of this wider failure. Philosophy and the Humanities failed the pandemic because they are bound too tightly to an untenable set of formulas, reflexively suspicious of purposeful quantification, and unable to account for the epidemiological reality of mutual contagion or to articulate an ethics of an immunological commons. Why? Partially because the available language of ethics is monopolized by emphasis on subjective moral intentionality and a self-regarding protagonism for which "I" am the piloting moral agent of outcomes" (Bratton, 2021).*

At the COVID crossroad, intellectuals were more or less aligned with Bratton or at least rejected Agamben's ideas. In any case, the expansion or diminishing state was at the centre of the analysis. Beyond old philosophers in despair and grant-recipient scholars of global government, I would argue that the state's role in disaster relief effectiveness needs to integrate the emergence of autonomously organised civil society. Klein focuses on corporative interest and market forces while Agamben and Bratton critique the state from negative and positive points of view.

These positions, similarly to the vulnerability studies and disaster studies perspective, fall back into the conundrum earlier described: precedence is given to rupture events, limiting profound critiques to the systemic inequalities at the root of the issue. Although these frameworks are useful in the case explored in the opera and in many other disasters that become the starting point of new forms of dispossession, there is a need to expand beyond production, profit, command and control into the domain of social reproduction.

The binary of state and market, mirrored by the frameworks of disaster capitalism and state of exception, is missing a third attribute to properly understand the situation: civil society, mutual aid, and the theoretical framework of social reproduction. Illner claims that we need to critically incorporate civil society into the framework of disaster studies in the 21st century: As people are trapped between a retreating state and advancing forces of marketisation, the figures of mutual aid and civil society are assimilated to fulfil a necessary reproductive function. An example explored by Illner is the disaster relief efforts of the Occupy movement in the wake of Hurricane Sandy (Occupy Sandy), which struck vulnerable populations in New York. Occupy Sandy was praised by the American Department of Homeland Security as more effective than state-led efforts. Praise, of course, accompanied by an immense funding cut to disaster relief and a modest set of grants and NGO funding used to co-opt Occupy activists. Not to mention that Occupy Sandy operated on a volunteer basis and charity that amassed 1.5 million dollars. The state was 'off the hook,' and private capital was excluded from disaster relief altogether. In 19-S II, the political landscape did not permit for the state to be 'off the hook,' but it was clear that political agents capable of capitalising on narratives of solidarity and mutual aid

positioned themselves to win the elections and, a couple of months later, to impose a 'republican austerity' into the country.

Illner claims that civil society – with tools like mutual aid, and (situated) local knowledge – in the case of Occupy Sandy, appear as a useful figure for reducing the significance of the state as a social institution – allowing it to focus on surveillance and law enforcement – and to further liberate the market from necessary labour for its reproduction. In Mexico, perhaps more familiar with disaster, a more general political change is taking place where the state relies on narratives around mutual aid initiatives. In other words, capitalism, via mutual-aid-based organising, is acquiring the ability to operate under profound stress conditions by the development of resilience by its most exploited population. Subcomandante Marco's foresight cannot appear more sobering as he claimed that "[...] the market can get used to that reality; it is possible for it to operate in a context of destabilisation or civil war and still be quoted on the stock exchange" (2001)[12].

From Toxic Resilience to Reactive and Generative Resistance

The movements towards resilience and co-optation of mutual aid by neoliberalism by no means indicate they should be rejected as emancipatory strategies; they only require to be observed through a perspective that integrates structures with processes. Similarly, we need to expand the philosophical understanding of time that pervades early cybernetic perspectives on disaster as well as the vulnerability

[12] My translation, original: "el mercado sí puede acostumbrarse a esa realidad; es posible que opere en un escenario de desestabilización o de guerra civil y cotice en la bolsa de valores"

perspective. For now, I will put in place a profound critique of the concept of resilience, re-dubbing it as toxic resilience, and contrast it with resistance.

Resilience implies flexibility and adaptability in the face of damaging environments as it "[...] designates the capacity of a system to withstand strain and pressure without incurring fatal damage (Illner, 2021, p. 17)." Resilience is a concept originating from ecology; here is a first clue on how vulnerability perspectives, even though advancing towards a more anthropogenic understanding of disaster, fail to avoid the naturalisation of human suffering. The notion of resilience is now applied in a multitude of contexts: from psychology to war, from computational science to art practice, from financialised markets to disaster studies. While working on LFC, I came to understand that when resilience is not shaped by class consciousness, it becomes toxic resilience, contrasting resilience as an ecological concept. Whenever it is mediated by class struggle, it is understood as resistance.

In university settings, toxic resilience is praised above intellectual curiosity. In art practice, it is observed as more valuable than aesthetic exploration. The first condition to be a scholar or an artist is to be capable of enduring capitalist destruction similarly to people guarding themselves from a super storm without (state) support. Choosing intellectual curiosity above toxic resilience is punished by academic death. Choosing artistic courage in the face of extreme austerity will lead you to the torture room of misrepresentation or indifference. Such is the paradox of culture in our current condition: by surviving we die, and what survives is un-dead, not actually living. Technocratic art and technology studies are a site in which toxic resilience replicates; when an artist or scholar liberates time, by technical means, they often choose to become more resilient to capitalist cannibalism, which only means that capital has multiplied once more its resource supply. I will elaborate on this point in the last section of this chapter by analysing

the political economy of music in relation to the networked music ensemble that performed the opera. The solution is not to cultivate a dead-drive via anti-resilience or dismiss its usefulness (O'Brien, 2017). The idea here is to develop a concept that allows us to think dynamically about social reproduction, production and refusal. This is why it is fundamental to revisit the concept of resistance to amend resilience.

Thus resistance, in contrast with resilience, is a dynamic process between forces of domination and defiance, where subaltern groups not only survive but, with them, other forms of life persist as a challenge to hegemony. Illner's contrasts the victory of Occupy Sandy with the rise of Black Lives Matter (BLM) in the aftermath of the George Floyd's murder and amid the COVID-19 pandemic:

> *"[W]hat is remarkable about the [BLM's] uprising is that protesters consistently connected police brutality to Coronavirus vulnerability, as part of the same systemic exposure to premature death that characterises everyday life for black Americans" (Illner, 2021, p. 127).*

While Occupy Sandy was easily co-opted by state power, the experiences endured by black communities during the pandemic led to an organisation effort that manifested strongly with powerful political implications at the heart of fascism and empire. The three acts of the opera reinforce Illner's argument. Thus, Black Lives Matter's uprising is comparable to that of the movements related to the House of the People 'Samir Flores Soberanes', San Gregorio Atlapulco and the activists of *Punto La Gozadera* and organisers around the collapsed factories that transformed their earthquake's involvement into resistance avoiding altogether toxic resilience. There are plenty of examples around the earthquake that fit the pattern of toxic resilience I am drawing here, mostly stereotypical stories that need a different kind of analysis: from state propaganda making up soup-opera-like rescue operations, passing through the

strange oxymoron of nationalistic exaltation of mutual aid to solipsistic techno-scientific contemplations of humanless nature.

Drawing from Caygill (2013), there are two opposing ways to understand resistance: a negative resistance from domination and a positive resistance for the reproduction of life. I will refer to the former as reactive resistance, and the latter as generative resistance. Reactive resistance intrinsically de-naturalises catastrophe by revealing and opposing a specific source of oppression. While Illner does not discuss in-depth resistance as a concept, it is safe to infer that he proposes a reactive resistance, and perhaps generative resistance is more in line with Occupy movements that he critiques. Faramelli (2020) proposes that only generative resistance can produce an 'ontological transformation of the world.' Faramelli views Zapatismo as a prime example of positive, generative resistance: a philosophy capable of producing forms of subjectivity that escape fixed (Indigenous) identity and ossified class consciousness. According to this position, Zapatismo – as well as the movements foregrounded in LFC – focus on the transformation of the experience of humanity to grow strength out of difference and produce 'a world where many worlds can fit.'

I argue, based on research on earthquake events and the performance of cyber operas, that Zapatismo – and the movements described in the opera's story – operate in both ways: through reactive and generative resistance. The ability to dynamically transit freely from reactive to generative moments of resistance is perhaps a third category that the opera maps effectively. Faramelli's position cannot anticipate the actions described in the opera's storylines: three movements in opposition to capital's expansion that attempt to take territory from the market and the state explicitly and directly. These movements are represented in LFC as the three acts of the opera, implying that this artwork signals

towards reactive resistance solely, which is not the case, but this will be explored in the following sections of this chapter.

To understand how the movements articulated by the earthquake event induced also generative forms of resistance I will need to produce a dialectical inversion of Illner's argument: now that it is clear that the disaster is capitalism, then it is necessary to understand what is left of the earthquake event beyond the capitalist-induced destruction and pain misconstrued as the outcome of the earthquake. It is necessary to, again, turn solely to the critical rupturing event.

Ways of Sensing: Cybernetics, the Internet Brain and Trauma/trauma

To move the argument forward it is necessary to invoke cybernetics once more. However, not the cybernetics originating in military command and control, but the cybernetics stemming from psychiatry and democratic socialism. Based on psychiatry, Stafford Beer's cybernetics, especially around the Cybersin infrastructure for Allende's socialist national project of cybernetic management of state production, uses the brain as a model. Beer's cybernetics are performative, and its role is adaptation. However, these cybernetics:

> *"[…] prioritized the long-term survival of the company over the short-term goals of any one department. This attention to overall survival reinforced the importance of holistic management and of Beer's conviction that effective management functioned like the human nervous system. Most companies of his time divided their operations into department that oversaw the company's activities in assigned areas and dealt with the problems that arose in these areas. Beer believed that this fragmented, reductionist approach could result in decisions that benefited a particular*

department in the short term, but that moved the company towards a greater instability in the long term" (Medina, 2014, p. 25).

Such conception of adaptability moves Beer's cybernetics away from toxic resilience. The main feature of Beer's conception of cybernetics has to do with self-regulation and down-up as well as top-down control mechanisms. This is how communication technologies worked in the aftermath of the earthquake that allowed the reaction of civil society and allowed it to feedback with/against state control. The brain metaphor, as well as ideas on adaptability and self-regulation will be key to translating the earthquake's experience (from my position especially) into the mechanisms that govern the opera's systems of creation, communication, management, and control. Similarly, the metaphor of the 'cybernetic brain' is key to understanding the cultural affordances of the ensemble discussed later, the music conception that it engenders, and the internet as a space of possibility. However, I am falling back into systemic processes and structure rather than tackling the seismic event as a site of analysis.

What is left after subtracting the capitalist disaster from the earthquake is also the subtraction of the reality of capitalism that mediates our relations as human beings. In other words, the reality imposed by capitalism crumbles like a pile of rocks and shatters like glass leaving room for something else. The earthquake opens a psychosocial critical time-space where people can act freely. Free from work, free from social conventions, free from alienation. Such temporality is also affected in its conventional emptiness, speed, and linearity. Mexico City is a city prone to disaster, and as such, transgenerational and personal trauma around it emerges and is negotiated in the public space – even more spectacularly in the earthquake of 2017, which date coincided with the 1985 earthquake. The earthquake left us with visions of mutual aid, care, and class struggle that have shown that life beyond capitalism is possible

where people can freely choose to struggle for life and to help and to heal each other. These visions are fundamental for the long-term organisation strategies that are the focus of the opera's narrative.

As a scholar who openly challenges academia's either dismissal or instrumentalisation of art as research, I will give myself plenty of poetic licence to elaborate a complex and useful metaphor around the texture and possibilities revealed by the earthquake. If a brain is a proper metaphor for social management based on Beer's cybernetics, then we can understand better the earthquake drawing from neurosciences:

> *"[Studies] suggest that MDMA may exert its therapeutic effects through a well-conserved mechanism of amygdalar serotonergic function that regulates fear-based behaviors and contributes to the maintenance of PTSD. Perhaps by reopening an oxytocin-dependent critical period of neuroplasticity that typically closes after adolescence15, MDMA may facilitate the processing and release of particularly intractable, potentially developmental, fear-related memories" (Mitchell et al., 2021, p. 1031).*

What this paragraph suggests is that MDMA is useful in the treatment of PTSD by opening a critical (mental) space where fixed, solidified habits and thought patterns can become momentarily malleable. Thus, the patient can 'let go' of fixations and fear-related memories. Even though the social value implied by this research – global north soldiers will be able to go to war, brutalise black and brown bodies and come back and 'let go' their Trauma – seems extremely problematic and is out of scope from this dissertation, the mechanism by which this drug works seems quite interesting. The key concept here is that of the critical period: "[…] a developmental epoch during which the nervous system is expressly

sensitive to specific environmental stimuli that are required for proper circuit organization and learning (Nardou et al., 2019, p. 116)."

I draw from this little fragment of neuroscience that the earthquake, similarly, opens a critical period that allows psychosocial neuroplasticity. People experience reality from a different perspective, develop bonds and relationships that are not mediated by profit or domination, and—ultimately—develop a 'way of seeing' that gets internalised, invades capitalist reality, and resists it. In this way, reactive resistance can only operate in conjunction with generative resistance. The experience of people's care induces long-term courage for struggle.

Amid a disaster like an earthquake, people acquire new ways of sensing the world that reveal two fundamental premises: a world beyond capitalism is possible, and the reality we suffer under capitalism is a disaster. What I am exploring in the opera is the way in which the momentary lucidity here described manages to manifest as a prolonged social force in the form of active resistance against housing insecurity, exploitation of workers, territorial expropriation, predation of resources, and as a form of imagination that adumbrates new forms of culture sharply in contrast with what we currently have.

To understand the long-term effects of the earthquake's visions, it is necessary to interrogate the following: what is the difference between a pile of debris in Mexico City and an unaffordable empty condo downtown Toronto? Or even more, what is the difference between a collapsed building in 2017 in Mexico City and a gentrified neighbourhood in 2025 in Mexico City? What is the difference between someone in shock after witnessing a building shattering into pieces and a person's shattered body from police brutality? What is the difference between homelessness provoked by financialisation and homelessness provoked by seismic activity? Gabor and Daniel Maté (2022) explore trauma as an underlying determinant

of current forms of culture they deem toxic. However, trauma should not be considered solely salient catastrophic events or radically inhuman forms of personal abuse. In turn, trauma is a great spectrum of psychic and/or physical wounds remaining after a great variety of circumstances that can go from cataclysmic events to everyday life. At one end of the spectrum there is Trauma – PTSD would be an example – and at the other end there is trauma – for example, enduring bullying at school. A minimal working definition of trauma can be synthesised as follows: "Trauma is when we are not seen and known (Maté & Maté, 2022, p. 23; van der Kolk, 2014, p. 43)"

First thing would be to acknowledge the continuities between a collapsed building and an unaffordable condo-as-financial-instrument: both are expressions of the catastrophe that capitalism is and are sources of trauma. What is different between these two is that a collapsed building can be sensed drastically. At the same time, the condo remains subtly in the background, hiding the multiple forms of dispossession it engenders. When a building falls, the veil of ideology is lifted, revealing the connection between trauma, state abandonment and capitalist hoarding. First world people with their carefully crafted reality, cannot feel their oppression, while overwhelmed with subtle forms of trauma. In an earthquake event we look back at what refuses to see us and recognise us, revealing the continuity between Trauma and trauma. In the earthquake I experienced, an inversion took place where power appeared as obscenely unnatural and then, the naturalisation of human suffering became unthinkable. The sound of glass shattering of condos going down echoes whenever I see a condo coming up in Toronto or Hamilton's destroyed downtown cores.

In the course of this dissertation – specifically in chapters one and three – I have discussed in depth these mirroring forms of disaster as two sides of the same coin: the dispossession by 'natural' disaster, on the

one hand, and the dispossession by the novel form of capitalist settler-colonialism pervading Mexico City currently, on the other. I have also discussed how this two-sided disaster has been met with whititude and toxic resilience – both forms of false consciousness that reproduce capitalist resources and rationality. It is important to understand the continuities between what is narrated in *Temazcal 2* and LFC. The double consciousness framework that I developed in Chapter 1 can also be applied to the experiences of the people at the heart of LFC: the residents of San Gregorio Atlapulco, the immigrant Otomí community, as well as the working-class, queer and feminist collectives responding to the earthquake. The pattern of the triple movement that I used in Chapter 1 also appears when understanding resistance as generative and reactive. This fold consciously traced in my dissertation signals towards a conception of capitalism as a social totality: that partitions demarcating national territories, academic disciplines like economics or politics, and dichotomies between social justice or class struggle are not useful unless we make the necessary connections to explain integrated global capitalism. The broader methodological point that can be learned from this dissertation is that patterns applied to these two different, and apparently contrasting, crises become useful when faced by a sequence and overlapping crises of distinctive and unique nature. The converging nature of crises is in itself a theoretical consideration that needs to be central for tackling issues in a world shaped by ecological, social, economic collapse as well as the collapse of care and human imagination.

The perspective of social reproduction on disaster argued by Illner – that shifts the attention away from the transient event and into the structural attributes of our historic moment – is supplemented with this psychosocial perspective: capitalism is the disaster that traumatises the social body, the earthquake is then medicine capable of opening a critical period for it to see, sense, understand and heal its trauma.

What I understand as an important contribution from this chapter section to disaster studies is the proposition of a cybernetic science that reproduces this inversion: A science that aims to move us away from meta-stability into the unfamiliar, like MDMA opening a state of neuroplasticity where we suspend old habits to reconfigure the way we relate to the world. This insurgent cybernetic science would correctly assume that our everyday life is a catastrophe that needs to be reshaped by radical ways of relating with each other, our work, and the environment.

LFC sits close to the media conception of *pharmakon* (Stiegler, 2012), where technical means open ways to see the world differently. However, this media artwork articulates media and arts with concrete social struggles perhaps inclining it towards what Stiegler frames as the medicinal effects rather than the poisonous forms of alienation it may become. In the absence of concrete social relations, media may become the territory of the realm of hungry ghosts, where the appetite for pharmacology's short-term relief dominates the lives of people trapped in our trauma-fueled culture.

This opera, by re-orienting knowledge around the earthquake experienced in 2017, is a cybernetic system that promises to deliver neuroplasticity to the Internet's brain making it possible to exercise certain agency in the way we interact with our digital environments. It is also a storytelling system that has reshaped certain narratives of what the 2017 Mexico City earthquake represents (in the opera's narrative is reactive resistance), and it is the most substantial attempt at algorithmic acid music, a form of generative resistance vis-à-vis the music institutions and industry that dominates our cultural landscape mirroring the earthquake's experiences. The section that follows will thoroughly describe the opera, emphasising its symbolic dimensions and the last section will discuss how the lessons learned in the earthquake's experience and research have allowed me to envision the music ensemble as a site of

mutual aid and care capable of anticipates modes of production and regimes of representation that we might want in our time.

La Fábrica Colapsada

Like *Temazcal 2*, this title indulges in a game of appropriation that de-stabilises its meaning. It is presented as an opera in the same way that *Temazcal 2* is dubbed a documentary. Lately, the titles of my artworks signal towards a contradictory position, as did my music education and the style flourishing from it: too un-European before the cultural turn towards progressive neoliberal de-colonialism and too European for the context after it.

It is also an un-ironic reference to the work *La Fabbrica Illuminata* by the Italian composer Luigi Nono. Nono's intellectual and political work – perhaps strongly present in *La Fabbrica Illuminata* – is profoundly marked by Gramscian Marxism (Velasco-Pufleau, 2018) and Operaismo (Wilkins, 2023) in the same way that my current work is influenced by Marxist-Feminism, Zapatismo and Fisher's cultural critique: three theoretical domains that resonate loudly with Gramsci's ideas. Moreover, with this reference I am identifying an underlying current of thought that is urgently needed in the face of our current historic moment and urgently requires perspectives like the one offered by the present artwork in the same way the New Left moment and the Italian 'hot summer' engendered Nono's intervention.

The title of the opera points to a building that fell and another one that was heavily damaged in the earthquake of 19-S I and finally fell in 19-S II (Turati, 2017). The repetition of two very similar situations is too poetic to be left un-mentioned. At the time of 19-S II, such a coincidence enabled a confusing situation where people were spreading a rumour about 19-S I as unfolding. Here is the root of my

argument: the civil society acting on 19-S II was also acting to remedy what happened in 19-S I. There seems to be no distinct disasters but only one that prolongs through time and re-emerges when the earth shakes.

The Overture

In this section I will describe the overture (first half of the artwork) as a composition and the conditions in which the three acts of the opera (second half) rest. The first moment of the overture produces a strong sensorial reaction in the audience. It is meant to concentrate on the fact that earthquakes are vibrational events. I use low frequencies, acoustic beats and loud volumes to affect the audience in their bodies with something that attempts to mimic earthquake waves but in an acoustic medium rather than seismic movements. The acoustic beats appear pre-musical and purely sensorial already creating a clash between the symbolic and the affective dimensions of the piece. On the screen the libretto's performance takes place:

> *"On September 19th, 1985, a factory full of textile workers collapsed as an 8.1 magnitude earthquake hit downtown Mexico City. They said it was the earth, a divine act, its purpose only knowable by god, its mechanisms explained solely through science. The majority of the people working in the factory were women. Their evacuation was delayed because they were obliged to protect the factory materials from being stolen. They were not rescued when trapped under the collapsed factory because cheap labour is disposable. Their bodies were crushed as debris so the industrial textile machines could be saved. After the factory collapsed, everything else did as well..."*

What is being described here are the facts on the 19-S I earthquake in the Topeka Factory south of downtown Mexico City. Numerous documentaries and papers have been written around this tragedy and its aftermath: a union of textile workers that was key to govern labour laws and movements in Mexico City and the rest of the country. In historic terms, this earthquake – and its aftermath – are some of the ways in which a culture of state protectionism and state managed capitalism appeared naked to a civil society ready to transform life in Mexico. What happened in the following 32 years is a series of co-optation or suppression of social mobilisations: first, the words democracy and solidarity were used to accept market competition and bribe active groups to accept land and resource dispossession. Following this, the rise of Zapatismo was betrayed and criminalised. The epidemic of femicide was covered-up. Movement after movement build-up pressure to end neoliberal rule while governments and market forces attempted to diminish it.

Such a pattern becomes the idea embedded in the second moment of the overture. Where 2017 acts as an attractor to which social movements are pulled towards, some beforehand as the Ayotzinapa protests in 2015-16, and some afterwards like the protests against global north's genocide of Palestinians. A spiralling, zoom-in movement forms an overwhelming turbulence that captures the attention of the audience anticipating a pivotal moment of the work.

This part is predominantly live coded with Estuary's MiniTidal language. Diego and I play with the samples while Iván explores noise and harmonic-rich synthesiser timbres. The samples are secondary sources retrieved from Archive.org and recorded by members of the ensemble. The samples are: a) The radio transmission of the First Declaration of the Lacandon Jungle from 1st of January,1991 to mark the military rise of the EZLN against the Mexican and American government, b) the protest in Mexico's City Zócalo

square over the murder and forced disappearance of 43 communist rural teachers and students of Ayotzinapa communal university, c) a mix of feminist protests on Madrid and Morelia recorded by Iván, d) recordings of the Plaza Dignidad on the 2019-20 protests in Chile over transit fare, e) May day celebration protests in Berlin on 2023, and (d) anti-genocide protests in Hamilton in 2024. The granulation technique used to create the disorienting effect is supplemented with a spatial composition that spreads the sound grains around the 24 channel system at the NIL and spreads them in the stereo field or the quadraphonic field in what I have attempted to be analogous ways.

At this point it is worth noting that the three sites of the performance (McMaster's NIL and Array Music in Toronto, as well as Diego's rooftop apartment in Mexico City and Iván's studio in Morelia) all present different conditions to transform information into audio signals. For example, the concert that took place in the NIL in early June had 3 different versions for its reception: the stereo version in Mexico City, the stereo version in Morelia (in theory identical to the Mexico City one), and the 24.4 version at the NIL. It cannot be said that the version where I am – with the status of composer – is the definitive and 'original' one. Certain mechanisms for spatial transformation were used between versions that had signified the piece in different ways.

As I write this chapter, I am working collaboratively with the ensemble in the version that will be released for an Internet repository and platform, and we are understanding how sound affects the listener differently depending on which version was experienced in the performances. I am invested in the diffused spatial character of what I have experienced while Iván and Diego prioritise the clarity and specific detailed 'image' favoured in their stereo version. Often, these two orientations on what the sound signifies clash. What seems to be the way forward is the juxtaposition and merging of Ambisonic binaural

techniques of the audio made by me with a clean stereo image of their synthesisers. Sometimes we incorporate their binaural signal to favour a diffuse rather than clear image of their sound. Regardless, this artwork has produced an imaginary space lacking realistic attributes nor linear temporality that, regardless of everybody's distinct experience of it, can produce the same conceptual and aesthetic goal. In one of the rehearsals, the ensemble was overcome by the chaotic, disorienting and immersive nature of this section of the overture. As expressed at some point: "all the revolutions of the world converging" in this imaginary space, no matter how spatialisation was operating for each of us.

At the end of this section the overture's libretto moves forward:

> *"People, survivors, kept rising. Searching for the ones trapped beneath the debris. Asking for silence to listen for movement, a sign, the minimal hope... They, the un-affected, still blame it on nature. They say that the numbers, the measurements, do not lie. The plaques moved. (Our) pain is only natural. No need to be affected, they claim, we just need better computers, better models, better predictions... Now we are supposed to be a geologic force. The so-called anthropocene, humans shaping the earth. I wonder, how are the people trapped under the debris capable of enforcing their will onto the planet? Why would they want to? The question around which forces shape the planet is one of wilful ignorance. Our times are marked by one predicament: extinction or communism."*

This second intervention stands between 19-S I and 19-S II as what it described can be assigned to any of both events. This paragraph maps certain kinds of discourse around the earthquakes: the mutual aid, the accent on the natural aspects of these disasters, the ideological problems around the figure of the

Anthropocene. Finally, it quotes Berardi's diagnostic of 21st century circumstances invoking Luxemburg's famous 'socialism or barbarism' slogan (Luxemburg, 2004).

At the same time, a drone sound, sharply in contrast with the chaotic energy of the second section emerges statically and characterised by a digital synthesis texture. The gaze presented by the opera changed once more. In this section I am dealing with questions of situated knowledge and the scientific objective gaze. I attempted to do so through a data sonification of seismic geological data retrieved from seismic stations around Mexico City at the time of 19-S II.

This was perhaps the most challenging aspect of the artwork as it involved a lot of unfamiliar knowledge to be acquired. It relied in correspondence with the Servicio Sismológico Nacional (National Seismic Service, SSN) and scientists regarding the best approach to process the data in a manner that could be meaningful as art and as a media representation valuable for scientific research. Victor Hugo Espíndola Castro, the scientist in charge of our communications, provided firstly, a SAC file that I attempted to process using the Seismic Analysis Code Software. After being unable to produce the wanted results with such a niche software, Dr. Espíndola provided a CSV file I was able to process in SuperCollider, being much more familiar with that programming environment. A lot of the meta-data of the files was imprecise. However, I was able to retrieve the main information: 3 channels of the seismic station's coil moving in three different axes: North-South, East-West, and Vertical movement. I relied on the name of the station (its geographic coordinates), and these three channels of information. I produced, for each seismic station, three files that were later transformed using Wavelet algorithms, that then produced 4 time-series representing different resolutions of the same data. Giving a total of 12 time-series per

station and an overall total of 120. The code that processes all this information can be consulted in the repository attached to this dissertation.

There were many issues that needed to be considered at this point of the research: seismic activity's frequency range oscillates around 0.001 and 20 Hertz and the data points interval between samples was that of ten facts per second (100 hertz). If these are the frequency ranges I get as data, how can I express them as audio since the spectrum of human hearing goes from 20 to 22000 Hertz? I received information from ten different seismic stations and a big part of the work required me to anticipate issues with phase and noise pollution present in the data. I explored various forms of data analysis for this work, and I was ultimately drawn to wavelet transform and discrete wavelet decomposition functions since it offers better time-frequency localisation, provides multiresolution analysis, and it is a robust method to identify and isolate particular features of an earthquake (Jun-Wei et al., 2014). I used a Ricker wavelet for this project and produced four different resolution scales. I used a module found online that allowed me to produce wavelet transform operations including decomposition (Schatz, n.d.). I had to write the function for the Ricker wavelet. I believe that the Haskell software I hacked is still not robust enough to be used for scientific purposes. Nevertheless, a whole workflow was put in place to decompose seismic data.

The most relevant insight acquired in the sonification process is its validity as a scientific tool. Given my experience with sonification I, would say that rather than trying to understand sonification as a scientific methodology we should understand it as a composition technique capable of producing sonic material that may assist composers and artists, rather than providing scientists with observation tools capable of an objective gaze. The dangers artists run while sonifying are those of invigorating positivist and neo-positivist ideologies that privilege scientific knowledge above any other. Moreover, sonification as a

composition technique might re-orient scientific gaze towards a more holistic perspective on life and natural phenomena.

As this work relies on spatial information – the sound sources are located around the audience – a significant portion of my work had to do with locating the sounds representing the information of each station as an analogy of the geographic relations between the seismic stations and three representative locations in the city that are meaningful on the opera's story. The listener of this sonification study is metaphorically positioned simultaneously in San Gregorio Atlapulco, The House of the People 'Samir Flores Soberanes,' and the collapsed building of colonia Obrera. As the listener stands in these three places simultaneously, they hear the seismic stations emitting sound as speakers rather than 'microphones' capturing nature. The listener is observing the instruments of observation put in place to anticipate earthquakes. The gaze is on scientific knowledge, nature remains elusive to sonification and visualisation techniques. The visuals here are more abstract and reactive to sound, they invoke a form of turbulence that works very well with the kind of synthesis used in the sonification and with the last section of the overture that follows.

The sonification is interrupted by sound that I describe as an irrational interruption of an otherwise exceptionally planned artwork. This section is noisy, arbitrary, and the most personal moment of the composition. It is a work of audio-synthesis that attempts to communicate my personal experience of the 2017 earthquake. I do so by creating four synthesis processes that attempt to replicate the material and psychic experience of the earthquake: a) shattering glass, b) debris rumble, c) buildings falling, and d) panic. This section is, in a way, programmatic; it describes a sort of theatrical imaginary action. I attempt to describe the sequence of events that I experienced at the moment of the earthquake. The

audience does not have to understand the programmatic aspects of this section, the main idea I attempt to communicate is that of a rupturing event: what I have previously described as the reality imposed by capitalism, crumbling like a pile of rocks and shattering like glass, leaving room for something else.

Again, this rupture is not what it seems. What shatters are not the buildings in Mexico City, it is capitalist realism and its common sense. But also, at the level of the music work presented, this rupture event opens a critical period where the work shifts to a general texture of radical freedom to improvise. This critical period mirrors the one explained in the last section of this chapter: the earthquake, when removed from the catastrophe that is capitalism, opens a similar critical period. In a way, the inferred continuity of the work, rational and schematic up to a point, is undermined by this unexpected turn. Such a gesture is not a simple formal pivot to another style of music-making, but what I have successfully attempted is to thrust the performance towards something that conceptually is rigorously consistent but unexpected by any listener. Perhaps is better described by the last libretto's intervention:

> *"On September 19th, 2017, an earthquake catastrophically hit Mexico City once again, just two hours after the commemorative drill of the 1985 earthquake held annually in all public buildings and announced through speakers all over the city. More than 40 buildings fell. Hundreds of lives lost. Among many: a house, a piece of land and a factory collapsed not only with catastrophic consequences but also shattering the reality that keeps us on the way towards disaster. After the glass stopped shattering, the smell of gas dissipated, and the dust settled, a critical period opened up, when the clock gave way to calmed, slow and multiple temporalities. The site of production gave way to a space of reproduction: for mutual aid and class struggle."*

The Three Acts

The section that comes next are the three acts of the opera. As polytemporality would allow me, the acts are not linear but are performed continuously. The visuals that maintain the storytelling do appear one after another, linearly. The visuals have been programmed with Punctual language taking advantage mainly of timed functions, audio analysis functions and polar coordinates combined with an archive of images related to the earthquake. The way I have modified and animated these images respond to certain concepts that I relate to the storyline. The first act appears to me as a network of houses and people spread in certain city neighbourhoods. Thus, I have tried to present images of the *Otomí* community as a network. The images of act 2 are animated first, reminiscent of water and then of fire, tropes present in the storyline of *San Gregorio Atlapulco*. The last moment of the third act is an image that gets distorted, resembling a flower blooming from the debris of the collapsed building in the *Obrera* neighbourhood. To understand these images better, the audience is encouraged to read the opera's story deployed as a website asynchronously from the opera's performance.

I have already talked about networked and spatial conditions in relation to the second section of the overture; the three acts are the section where this becomes a crucial aspect of the work. The acts are where network latency, different audio and computer equipment, and acoustic circumstances produce distinctive experiences and meaning of the artwork according to the location of the ensemble members or the audience. Rather than thinking of this as inconsistencies that need to be corrected, I understand this as the possibility of understanding reality itself as multiple: multiple times and multiple spaces co-exist without collapsing into each other. More importantly, this multiplicity of time-spaces is not metaphorical but a technical condition for networked algorithmic music-making.

Paradoxically for these differences to be meaningful, there should be a process to harmonise the software being used to produce audio. For example, one substantial issue was how to have consistent levels and be able to have different spatial configurations. This particular issue became the focus of rehearsals for the NIL performance: how to be able to understand Iván and Diego's perspective on levels and how could they understand mine? We surmounted this problem by relying on audio analysis software, constantly recording audio from different perspectives, and talking a lot about our audio experiences. Similarly, we produced a process to aesthetically create an imaginary space for exploration where we agreed upon a conceptual goal that would delimit and stabilise our horizon of possibilities.

In terms of tuning and polytemporal systems, these two are already an intrinsic possibility of TimekNot, particularly the ability of creating music ideas in different tempi and articulating them organically. Tuning research is one of the most relevant guiding concepts of Diego's research. The conversations around tuning systems and polytemporal textures have resulted in a mutual transformation of Diego's and my own practice. Nevertheless, for LFC we created a unique 'middle ground' for aesthetic exploration, where the output is never quite what anybody would desire. Diego dialectically meshed my conversations on Middle Eastern music with his research on just intonation creating an approximate version of a Shur Dastgah that we dubbed ShurNot. I approached his research on (haptic) synthesiser's performance of just intonation scales into a fixed structure that inherently defined specific patterns for using a scale. In this structure, the higher the octave, the higher the density of notes (from 3 notes in the first octave, to 12 notes in the fourth). Some notes would be tuned differently if they were to move in ascending or descending melodic motion. The structure could be rotated, changing the 'mode' from ShurNot to something more reminiscent of a minor scale. In terms of polytemporality as rhythmic principle, I had to

adapt TimekNot's research to the synthesiser performance that, for this project, was favoured by both Iván and Diego. Iván and Diego developed a counterpoint technique that I understood as a response to TimekNot. Beyond timbre, rhythm, and harmony, we became an ensemble preoccupied actively on melody.

Beyond the coded infrastructure in which this work stands, the three acts described are an expression of reactive algorithmic music, where a series of instruments have been programmed to react to the performer's interactions, and live coding, where code is being manipulated live. Iván has developed a guiding practice of embodiment in performing environments as the one here discussed. Such a technique shifts live coding's speed, it requires calmer and more spaced interventions and a more thoughtful listening.

This distinction matters because the different relationships to the algorithm allow performers to produce different textures and fulfil different functions. Synthesisers can change very drastically and very fast the type of interventions they have amongst themselves and with the live-coded part. Live coding is a slower process that maintains a certain distance with the material produced. In our setup, the live coded components became a kind of formal vessel, while the synthesiser's performance inhabited it.

In terms of audio and music, what appears in the three acts is the convergence of the disaster that is capitalism revealed by the earthquake and the disaster that is capitalism that can be observed by a basic political economy analysis of current forms of music production. What has been described in the storytelling of the work is now performed by the ensemble. This convergence of critical periods is a rich space-time that connects the concrete social circumstances presented in this opera with the crisis of

music-making that *Pirarán* faces as an ensemble. The music presented here is an attempt to transform a crisis into a culture that rejects toxic resilience and embraces generative and reactive resistance.

In this section I have described the components of the artwork that function as infrastructure or explicitly convey meaning regarding the earthquake storytelling. The meaning I draw from the earthquake's analysis has been an important event that has shaped the way in which I view the world. This is for me now evident as I observe the path that *Pirarán* has taken as an ensemble. In a way, all the components described here attempt to explain and prepare the audience to confront the radical freedom that the acts of the opera present. A freedom even from the narrative mechanisms that the opera relied upon thus far. This radical freedom can only be explained by understanding the political economy and the background conditions that shaped it, which will be the task of the next section.

Art(work) in the Net(work): The Algorithmic Networked Ensemble as a Site of Care and Mutual Aid

Music Disaster

In this subsection I will paint a persuasive picture of the music industry – and by extension music institutionality – as a disaster that does not allow new forms of music to emerge or musicians to live in a dignified manner. In this subsection, it is possible to understand disruption not as a rupturing event but as an advance of a systemic dispossession towards culture workers. In the next subsection it becomes clear that what technology disarrays is capitalist realism when approached critically.

Daniel Ek, the CEO of Spotify recently claimed that the cost of music-making – 'content creation' in his words – is close to zero (Ek, 2024). Spotify is a music distribution software for hosting music online and making music recommendations to people. Thus, the major asset this company has generated is the algorithm that sorts music and binds it to an individual user's music taste. The company made more than 1 billion dollars of profit last year while also de-monetising any music listing streamed less than 1000 times. Additionally, Spotify is infamous for the low pay per streaming to the artists creating their 'content.' Of course, music influencers – who wish their work was monetised better – have relentlessly critiqued the Swedish tech-billionaire. Such a situation encapsulates perfectly the current cultural crisis pervasive to music-making. Now let us imagine something much worse.

The algorithms deployed by Spotify are clever and interesting technology capable of overcoming an old issue of music markets: the stockpiling of music without any enjoyment as use-value. Algorithmic technology has transformed the way we exchange and consume music by connecting potential listeners to music with unprecedented precision. However, Spotify ultimately relies on music made by humans which creates limits to its exploitation potential: the infinite consumption machine does not have an infinite production machine. So, how can Ek, the billionaire, become Ek the trillionaire by tackling the problem of dependency on slow and sticky human production?

The answer lies in the emerging generative Artificial Intelligence software technology that is in the process of pervading every industry imaginable to eventually become basic infrastructure for everyday life. As the impact of AI technology permeates all art forms, it becomes evident that this form of automation vastly differs from previous ones. All previous forms of automation have aimed to minimise variable capital (AKA human, waged labour required for a task) and maximise fixed capital (AKA the

means of production: machines and technical knowledge). According to Harvey's reading of the Grundrisse (Harvey, 2023), the current AI situation is yet another manifestation of class struggle as capitalist seek to minimise their dependence on workers. Machines cannot create value, but they are "a source of relative surplus value (Harvey, 2023, p. 324)." However, the automation of tasks and the formation of an externally objectified intellect that can be instrumentalised by workers does not imply a suppression of capitalist relations by scientific command of work (Marx & trans. Nicolaus, 1993). According to Marx, machines are optimised to extract surplus value from the worker's abstract labour, not to substitute it. Thus, capitalist accumulation ultimately requires (human) labour.

According to Dyer-Witheford, Kjøsen and Steinhoff (2019), Artificial Intelligence is not merely a method for automation; instead, it represents a substantial attempt to render labour a non-human activity, making humans redundant for the generation of surplus. In such a scenario, the horizon of the arts is one of zombified art markets: a complete circuit of production, consumption, and exchange of art where humans witness passively from the fringes the glory of capitalist incessant output. In this scenario, human imagination is transformed into raw materials: the original accumulation for such mode of production.

Limits on computational power and copyright issues within the music industry are impeding the full development of AI for music. However, Daniel Ek is anticipating the future; this is what he is referring to when he speaks about the 'zero cost' of 'content creation:' generative AI software expropriating the creativity of music-makers as data to create instruments, interfaces, mastering workflows, harmony, melody, rhythmic patterns, or quite plainly whole songs prompted by a user's desires. This will be perfected by extreme forms of data colonialism – a framework useful for understanding software that

users rent for a service while designed to extract data from them (Couldry & Mejias, 2019). Social media (Web 2.0) is the most paradigmatic space for data colonialism as it has effectively transformed life itself into a process of data extraction. While Spotify is already appropriating large volumes of user's interactions data and it relies mostly on non-AI algorithms, the appropriation of immense volumes of data of music producers and consumers could eventually feed AI machines not only to anticipate the music options that better fit a certain user, but to create music just-in-time to maximise user's consumption and appropriate revenue from production. This is already in-process as Spotify already creates music for it's own consumption to re-absorb revenue (Castle, 2017; Klawans, 2024). Spotify's full potential is to reduce the music industry to an infinite music production machine plugged into an infinite music consumption machine. If billions of dollars can be made with the impediment of human intervention, imagine what can be achieved when we remove it from the equation.

Anticipating such an automatised economy of music, one now can understand the crisis of the imagination and the slow cancellation of the future described by Mark Fisher as not a bug but a feature. The general disarticulation of attention provoked by postmodern cultural hegemony, and communication technology like social media, has turned consumers and producers of culture towards a radical form of conservatism: characterised as easily deliverable and immediate reward prone. Artists' precarity in the form of lack of access to dignified housing – thus, a lack of space and time to distance from pre-existing cultural forms and imagine new ones – as well as the dismantling of cultural institutions providing financial support has turned art towards safety and risk-aversion. Thus, social reproduction perspective is fundamental to understand the crisis of music making and this perspective is key to find a way out of this crisis.

Twentieth-first century music is, according to Fisher, no more than pastiche and retromania. Cultural artefacts everywhere are remakes of a previously successful art-piece. This perfectly fits the way in which AI can intervene in cultural industries. An infinite recombination of data that gives the illusion of novelty but cannot, by definition, create something new. Hence, technology is not essentially problematic, but it is entwined with capitalist culture. Such an argument, often misconstrued as technophobia or fear of novelty, is difficult to communicate in a cultural context that naturalises technology's obsolescence rationality and conceives of time as a straight accelerating line moving forward. However, when we acknowledge pre-existing cultural and social circumstances that shape technological and scientific research it becomes clear that the problem is not technological. The scope of this critique is to dialectically transform both technology and culture through the desire of escaping the nightmare of cannibal capitalism.

Frederic Jameson (1985) claims that Jaques Attali's groundbreaking work on the political economy of music shows that "the music of today stands both as a promise of a new, liberating mode of production, and as the menace of a dystopian possibility which is that mode of production's baleful mirror image (Jameson, 1985, p. xi)." This subsection describes the timeless disaster that music-making has become under neoliberalism. Not only its past but also its possible nightmarish future.

Let's move into the promise of new liberatory modes – incarnated, I propose, by the networked, algorithmic music ensemble. A promise and a provocation that attempts to change equally our past and our future to something away from algorithms as control and extraction mechanisms.

Promise and Provocation

The crisis previously described seems inevitable and natural. However, as understood by Jameson, it is only the dark mirror of a liberating mode of production that is equally real and possible as a global algorithmic culture. As I have claimed earlier in this chapter, *Pirarán* is an algorithmic networked ensemble capable of adumbrating such a mode of production. *Pirarán* – from the future conjugation of the Spanish verb *pirar*, which originally means to escape, to fugue, but in current Mexican colloquial language often means 'to go crazy' – has been conceived in the midst of an existential crisis bound also to a music-making crisis and with its members experiencing at some moment of their lives the crisis-prone existence in Mexico City, specially the earthquake that is tackled in the first section of this chapter. Drawing extensively from such crises experience, we have developed an intuitive form of resistance reminiscent of the one I have described earlier in this chapter and I will attempt to map it as part of our musical practice.

Networks, Algorithms and Live Coding

This subsection requires to understand extensively the political implications of networked music and algorithmic music. Networked music – which is music made with others in real-time via an internet connection – has a rich history. However, after the COVID-19 lockdown it entered the everyday live since all musicians had to adapt to the internet as a medium to experiment and deliver music. As I have explained already in Chapter 1, my work in Estuary (Ogborn et al., 2022) and my research converged with the unfolding situation allowing me a privileged point of view regarding networked art collaboration and technical comprehension.

Algorithmic music refers to music governed by predetermined rules, following specific algorithms and performed automatically - historically by mechanised means and now via digital computation. Interactive algorithmic music allows real-time modification to the algorithm by a performer. This flexibility enables musicians to change their mind mid-performance, breaking the rules as part of the process. This concept is often known as live coding, where writing and erasing code becomes the performance itself. Live coding is often understood by its artists as a performance of human agency in the face of a computerised system (Franco Briones, 2022). “Live coders” are a huge and growing community around the world, with connections to both artistic and academic communities, and institutionalised in conventions like the annual International Conference on Living Coding.

The human ability to make, embrace, and amend mistakes is central to live coding. No wonder, this attribute has fostered one of the most inclusive and egalitarian electronic music movement, rooted in the anti-capitalist principles of copyleft software and communal property. Within the live coding ethos lies a kind of politics that are impressively dynamic and reprogrammable: imagine a political system that allows anarchism in the summer, Marxism in the winter, and a liberal fall.

Live coding emerges as algorithmic technologies shift the main site of surplus production from large operations (like massive record labels) to platforms like Spotify. Spotify addresses the issue of stockpiling music by increasing the mediations between production and consumption. However, live coding explores algorithms as tools for musical creation, capable of diminishing the distance between consumption, exchange and production.

In the 1970s, Attali (1985) described a future music practice that would overcome contradictions like the stockpiling of music without time to consume it. This closely resembles live coding. Attali envisioned

music capable of collapsing exchange, consumption, and production by distributing musical instruments and interfaces, allowing people to bypass traditional music institutions and industry.

Live coding allows people to create unique musical instruments and sounds, sharing them online seamlessly. It removes the need for extensive formal training or industry contracts. Furthermore, using techniques of networked music, it is possible to invite others into your own site for autonomous music-making and listening. Attali's vision of a music practice dissolving commodity fetishism and surplus accumulation aligns with live coding, enhanced by networked music elements.

Live coding, though resistant to co-optation, remains vulnerable to rapid technological advances and cultural slowdowns as described by Fisher. Currently, live coding is seen as a technical advantage in productivity, enabling immense output with minimal resources as described in Marx's foundations to political economy as the concept of relative surplus value (Harvey, 2023; Marx & trans. Nicolaus, 1993). If every musician is a tiny CEO and their music a small corporation, live coding offers relative surplus value – a temporary advantage due to its unique technical edge over industry competitors. So, a question that seems relevant now is: what have live coders done with the time gained by automation? Have we collapsed exchange, production, and consumption, and finally overcome surplus accumulation? Or have we, in a *petit-bourgeois* manner, amassed some form of relative surplus to advance our individualised musical enterprises that will evaporate once programming languages cease to be a technical advantage? Have we developed a form of toxic resilience? Coders faithful to the compulsion of perpetual production are susceptible to new technologies that promise a new source of relative surplus value and further ways of automation that substitute even more variable with fixed capital. This would be AI frameworks. So, in a way, the real life of live coding starts after its death as a novelty. AI as automation will run the same fate.

However, AI as a non-human labour power participating in the market and integrated to our everyday lives as infrastructure is uncharted territory. As such, it is difficult to anticipate what it means to have 'intelligence' in the environment at the service of capital.

Pirarán: A Flight Away from Toxic Resilience and Towards Resistance

This is the background that shapes *Pirarán*. The disaster in music propelled even further by the COVID-19 pandemic was the starting point of our creative process. Given such conditions, our project felt for me as a social experiment away from the artistic and technological domains that often over-determine and over-code the music-making context. A suspension of normality shaped *Pirarán's* possibilities, so to speak.

Suddenly we had time to reflect upon our professional careers as musicians, often a lonely path that forces us into competition rather than collaboration. The lockdown, the vulnerabilities revealed by it, and the perspective it provided in the face of the crisis of music circuits reshaped the way I valued music-making. Rather soon in our process of ensemble-formation, I found myself witnessing the unfolding of the creative process of my peers and becoming invested in it. My responses to their interests and research functioned as that of a committed audience trying hard to empathise with the vision of Iván and Diego. All of these responses, of course, felt reciprocal and all of these was also experienced for the creation process of *Temazcal 2*. What I want to emphasise here is the capacity to listen that opened avenues of development for us not necessarily for our collaborations, but for our more general 'professional interests' as musicians. This is also true for our collective improvisation: we become the commentators of our collective work as well. *Pirarán's* members, in this way, found a first way to socialise artistic creation that relieved some pressure to be a successful music enterprise. This

ensemble claims to abolish audiences by absorbing such function within the ensemble We are our own audience for our individual music-making practices, which is an idea similar to how the live coding community functions, as discussed at the ICLC 2023 (McLean et al., 2023). If the point was to be heard as a musician, this was already happening in our rehearsals. It is worth noting that our positionality as three heterosexual cis-men was not particularly inclined for such a dynamic: men are not socialised to engage in emotional and caring relations with each other. In some ways, the care we put in the process of ensemble-making draws from queer and feminist traditions of mutual aid that were so effective at forming networks of support among people that would need them.

From this point forward, *Pirarán*'s development relied on forms of mutual aid – that is, unconditional direct help and circulation of resources – such as equitable redistribution of financial support especially from the global north to the global south; knowledge-building activities in the form of tutorials, resource-sharing and explanations; shared skill-based labour for grant-writing, coding, music production, among other things that need to remain illegible to academia in order to be preserved as mutual aid.

While mutual aid is a driving concept for *Pirarán*, LFC needs to be understood in the context of my doctoral work. Most of the work for LFC required being executed as lonely-authored composition, programming and research for two main reasons. First, the parameters of doctoral work require this research to be conducted individually. Second, and as a consequence of the first point, the benefits of such work cannot be socialised as all other work conducted by the ensemble. After all, only one of us is getting a PhD out if this ensemble[13]. It is key to point towards the tensions already mentioned especially

[13] As of yet. However, it is in the horizon of Iván and Diego to start graduate work.

around resilience in academic environments that would need to be developed further to, as well, admit and understand the limits of mutual aid within this project.

Pirarán is also an effort to degrow presence of individuals in music circuits. I will discuss this point later in this chapter more, but the first thing we understand is that capturing people's attention in social media should be minimised if not avoid at all costs. Moreover, algorithmic, networked technology and immense technical flexibility allows three musicians to present their work with any budget of time and resources and *Pirarán's* setup is streamlined and centralised (often most of the setup happens in a studio while on stage we only need to connect to the mixer). Our ensemble usually liberates allocation of sets in concerts for more people to participate. On our last live performance *Pirarán* collaborated with Sarah Imrisek, Iván and Diego were on stage while Sarah and I were at the NIL. In this concert, four people performed in 10 minutes while most of the other sets were individuals presentations and often spent time beyond the recommended per set. Third, the three research streams of Iván, Diego and I are never sacrificed in any of our works. This means that the half hour of LFC showcases deep research on tuning systems, synthesis creation, live coding, spatialisation, among many other things, which is a framework that often clashes with the academic rationality where you can only present one research topic at the time. All materials – software, instructions, code snippets, tuning files, synth patches, etc. – are free and available for others to use. All of *Pirarán*'s production potentially multiplies production for whoever might need it and know how to operate it.

Pirarán understands capitalism as a social totality. I have attempted to argue in this dissertation that issues of representation and strategies for redistribution – like free open source software – cannot by themselves adumbrate a world beyond capitalism. As such, the ensemble's practice is rooted in the

impossibility to stand alone, capable of escaping capital's social relations. However, it stands as an incomplete and out-of-phase project almost as an archeological artefact not from the past, or even the future, but another timeline. In this sense, the ensemble is an anticapitalist project and it mobilises art creation towards the abolition of art institutions and markets.

We encountered various challenges in securing satisfactory technical conditions for our sessions, but perhaps the most complicated was communication. Intra-ensemble communication during performances in non-networked environments typically relies on visual cues, body language, and a shared acoustic space, which was not available to us. The form of communication we developed combines on-screen activity shared by the ensemble – such as code edits or custom time-managing automatic conductors, facilitated by Estuary – and purely auditory elements. In our performances, we have crafted a sound that best serves as a metaphor for a tactile, non-rational, noisy form of communication. By minimising visual forms of communication, our music has been infused with a unique way of hearing that can be described as psychedelic communication that undermines individualism and favours collective agency.

All of these have become infrastructure that requires significant technical knowledge of computers, audio, and programming. However, this knowledge is not gate-kept in higher education institutions or in secretive corporate labs; it tends to be almost entirely free and open source.

Our rehearsals are a combination of collective software debugging, music playing, and talking/listening. The loop of debugging, playing, listening, and talking allows us to help each other and practise a lot of care and attention. We learn how to listen to each other's needs, desires and anxieties, and be able to create a space for everybody to be capable of bringing to life their (audio)visions and to be committed to

making each other's visions come to life. This is one of the most relevant aspects of our ensemble that operates under the principle of care and mutual aid.

The extra-ensemble communication, what is usually considered as gathering an audience, has been the most challenging issue for our ensemble. This aspect of the labour of music-making requires us to shift our attention from music creation and creative work towards the work of marketing strategies and publicity. This work eroded the care and attention already built at the base of the ensemble. At some point, our time was getting lost in reels, stories, selfies, hashtags, etc. After this unsuccessful attempt of due diligence as a music enterprise, we made the decision to refuse most of our activity in social media, not only Spotify. It is difficult to put in words the anxiety and stress that refusing to participate in online distribution platforms and social media has provoked in the ensemble since we do think this music is worth listening to. However, we discovered that our radical commitment to caring for each other and the people supporting us was more important and ultimately productive than the attention of ‘the scene’ or the algorithmic mechanisms of control, surveillance and uneven distribution of attention and resources.

The relationship many artists develop with social media and distribution platforms can be understood as toxic resilience, which aims to free the market from the necessary labour for its own reproduction. By using time liberated by technology to promote their individual careers, artists relieve the market and the state of the responsibility to provide dignified living conditions for cultural workers. Toxic resilience in artists' environments also fosters transactional social relations. Social media and Spotify encourage a commodified relationship with our peers while further humanising commodities, deepening our alienation from others and from our own labour. In such circumstances, what artists consider valuable is heavily influenced by their reliance on toxic resilience, rendering art sterile and meaningless.

Refusing to participate in social media and platforms like Spotify, as a rejection of toxic resilience, aligns with our vision of what music can be. After all, collapsing consumption, exchange, and production – as Attali envisioned and as the live coding movement encourages – requires letting go of passive audiences (McLean et al., 2023), which in turn seeks to abolish the institution of art and to overthrow the music industry. If trauma is "[...] when we are not seen and known," then social media (another infinite consumption machine) is fueled by trauma. So, the social reproduction work guiding our ensemble has enabled us to address trauma manifested as the 'genocide of authenticity' that social media entails (Maté, 2022).

On the horizon of our possibilities is a project for an AI agent that will create 'content' by gathering data from our rehearsals, formatting it, and posting it automatically on our social media. This approach aims to address our ongoing marketing problem while undermining the relevance of social media content by rendering it as pointless, non-human labour. While the AI agent will have an apparent mandate to promote our ensemble, its long-term goal is to encourage people to move away from social media. We want to invite them to reimagine the internet as a space for creation and enjoyment, rather than a fragmented domain of production, exchange, and consumption. These are some of *Pirarán's* reactive resistance mechanisms embedded and imagined in our workflow: humans creating art while machines handle the work, allowing us to focus more on care and mutual aid. From such priorities, art that leads us towards new paradigms and a novel culture, free from capitalism, can be envisioned and experienced. Such vision can be understood as the ensemble's form of generative resistance.

Algorithmic Acid Music and Algorithmic Acid Communism

The ensemble's generative resistance is a conception of music that changes direction depending on the creative ideas we are exploring. Thus, it is not a style, genre, or identity that we have developed as such. *Pirarán* is influenced by communities that explore various aspects of tuning and time beyond conventional Western traditions, popular Latin American modernism such as Pérez Prado, Cumbia chichadélica and MicoRex, contemporary and experimental musicians, as well as musical genres including free jazz, hard bop, vaporwave, hippie synth music, techno, glitch, industrial, ambient, and noise. In *Pirarán*, networked and algorithmic environments converge with psychedelia and psychedelic music to produce what we call algorithmic acid music. LFC is the work that best reflects this psychedelic orientation.

With the renewed interest in global north countries in psychoactive drugs that threaten to commodify psychedelic culture, there has been a novel interest in psychedelic music (Farrell, 2023). *Pirarán's* sound is psychedelic in multiple ways: It demands a reorientation of the listener (Wanke, 2021) towards a delocalised sound capable of creating inner spaces that we navigate as performers but are clearly communicable to the audience. The multiple ways in which we create immersive sounds for LFC – multichannel spatialisation, tuning and tempo variations, synthesis exploration – not only explore geographic and literal space but open a space of possibility where aesthetics remain transformative rather than fixed.

This music is psychedelic because it emerges from multiple crises that rupture common sense and reality, leaving the listener and us in a liminal space where things can change very quickly and into anything imaginable. In LFC, the rupture event and critical period described in the second and first

section of this chapter allows the acts to turn towards the unforeseen. For the ensemble, the crises in the political economy of music and the COVID-19 pandemic preceded our collaboration. In my personal experience, 19-S II ruptured psychosocial stability, allowing the unthinkable. LFC's acts, *Pirarán*, and Mexico City can only exist in a space where hegemony has fallen like a pile of rocks and glass.

But more importantly, the term psychedelics traces a historical fold between the current moment and a cultural period from the late 1960s to the early 1970s, when psychedelic music thrived as a result of counterculture tied to many liberation struggles: the Black Panthers, gay rights movements, second wave feminism, decolonial struggles around the globe, movements against the Vietnam War, the rise of the New Left, among many others. In a nutshell, it was a historic moment that revealed what could have been instead of neoliberalism. Perhaps it also overlaps with the 1920s in Mexico City, where heterogeneous cultural expressions reflected the rich political landscape of a country full of possibilities after an intense social revolution (Madrid, 2008).

I am not advocating a retreat to 'the good old days' when 'music was better.' Nor should this be misunderstood as 'left melancholia,' failing to imagine adaptive new forms of radical transformation. Indeed, algorithmic acid music challenges the progressive neoliberal castle that attempts to misconstrue leftist self-critique to purify feminism, decolonialism, and other emancipatory movements from global class struggle.

LFC abducts its listeners and performers into a present time that is not ours but could be. The acts of LFC come from a timeline where we have Beer and Allende's cybernetics rather than Bezos's Amazon, where there is a thriving, multicultural, plurinational Palestine, or where neoliberalism was defeated by acid communism.

Acid communism, a term popularised by Mark Fisher (2017), is often associated with research and political thought from the UK-based radical group Plan C, the historiography of the 1970s by John Medhurst (2014) and Andy Beckett (2009), and the cultural studies work of Jeremy Gilbert (2014). The dark mirror of acid communism is capitalist and socialist realism (Fisher, 2009), both trapped in a dialectic where 'reality' undermines any opportunity for meaningful transformation. Often bound to the counterculture, acid communism is a politically robust cultural project capable of avoiding co-optation by capitalism and totalitarianism.

> *The concept of acid communism is a provocation and a promise. It is a joke of sorts, but one with very serious purpose. It points to something that, at one point, seemed inevitable, but which now appears impossible: the convergence of class consciousness, socialist-feminist consciousness-raising and psychedelic consciousness, the fusion of new social movements with a communist project, an unprecedented aestheticisation of everyday life (Fisher, 2017).*

According to Fisher, music plays a particular role in acid communism by being able to puncture common sense (Fisher, 2010). By refusing to merely entertain, music abducts the listener and exposes them to meaningful differences. Music becomes the "struggle over the means of perception[, f]ought out in the nervous system" (Fisher, 2010, p. 559). Beyond political economy analysis and the social reproduction perspective, acid communism enables *Pirarán* to transform the psycho-social field. Thus, music opens a critical period conducive to neuroplasticity.

This strong psychedelic impetus is mediated by technologies of the non-self. These technologies are meant to disrupt the individual as a minimal social unit and challenge the notion of private property at its core. They are not fixed; LSD, MDMA, or psilocybin are technologies that have been re-imagined as

commodities or in the service of warfare. Practices such as meditation and yoga are already associated with consumerism and Silicon Valley's toxic culture. Hence, there is always a level of risk in identifying what would constitute a technology capable of dissolving the sense of self that alienates people from communal existence. Through sound art and music, and aligning with various strategies of live coding, algorithmic, and networked music, *Pirarán* has transformed cybernetics and algorithms into technology and media of the non-self. This algorithmic acid communism offers both a promise and a provocation in response to current forms of culture and communication.

Conclusion

Throughout this thesis, I have analysed the works that I created as part of my larger PhD project. These works include: *Temazcal 2*, TimekNot, and *La Fábrica Colapsada*. Each of these works explores the intersecting relations of temporality and crisis, as well as the possibilities and emerging conditions for care and subversive subjectivities. I have argued that the algorithmic networked music ensemble contains within its possibilities the basis for new regimes of representation and modes of production that re-embed people at the centre of production, reproduction, and exchange. The dimensions by which I have described the three artworks—the two performances and the software—tend to be multiple: a perspective on the technologies used, a reflection on their meaning as cultural artifacts and their role in knowledge production, and a discussion of the artistic practices they engender. Similarly, these artworks trace the flow of the proposed methodology in which hacking produces art that produces meaning that produces technology to be hacked once more, etc.

In the first chapter, I described and expanded on *Temazcal 2*, a live-coded documentary artwork developed in collaboration with Rolando Hernández, with additional support from Diego Villaseñor de Cortina. The piece is a collection of multimedia elements – like images, audio samples, videos, programming languages, software, among other things – meant to be activated in performance to examine two interpretations of the *temazcal*: as both an electroacoustic composition and a Mesoamerican technology used in birthing and healing practices. As the input and output of this artwork – I engaged in a theoretical conversation regarding the tensions between indigeneity, mestizaje, and settler colonialism as they emerge in *Temazcal 2's* performance and my position as a Mexican immigrant

navigating a doctoral pathway in Canada. I have argued that *Temazcal 2* critiques how electroacoustic music appropriates Indigenous technologies and proposes a different subjectivity that rejects settler, Indigenous, and mestizaje figures as structured within colonial and capitalist histories. This refusal generates a form of double consciousness, linking situated knowledge – such as Rolando's family history – with critical artistic appropriation.

The second chapter described the affordances and limits of TimekNot: a computer language designed for live coders to program heterogeneous, music-oriented temporal relationships on-the-fly, enabling the triggering of audio samples across relatively autonomous musical layers. As a polytemporal language, TimekNot conceptualizes music through multiple, coexisting tempi, offering a novel approach to structuring rhythmic and temporal interactions. This chapter presented TimekNot as both a cultural project and an artistic intervention in algorithmic music and live coding. It explored the language's grammar and syntax, the experiences that shaped its development, and its role within the broader live coding ecology. There have been plenty of lessons regarding TimekNot's affordances as a live coding language that were learned in *Temazcal 2's* execution as an artwork; the double consciousness developed as part of it became a guiding intuition in the software writing and language designing process. The aim of TimekNot's affordances has been to provide a tool with which live coders could be reconstituted as collective subjects rather than individuals.In this way, this software has refused to reify the cultural neutrality of music-making technology.

The last chapter provided an in-depth exploration of *La Fábrica Colapsada*, a 30-minute operatic, networked artwork that serves as the central piece of my doctoral research. Incorporating visuals, multichannel audio, microtonal tuning, polytemporal rhythms, reactive algorithmic music, and live

coding, the piece unfolds as a performance piece, also asynchronously on a website. It is supported by online repositories containing multimedia materials and software reusable for seismic data processing, wavetable synthesis, and spatialization in networked environments. In this chapter, I analysed how, more than an artistic work, *La Fábrica Colapsada* is an artifact that generates new knowledge and sensibility about crisis and time, particularly through the lens of Mexico City's earthquakes on September 19, 1985, and 2017. Through this lens, I re-envisioned 'disaster' by integrating perspectives from social reproduction theory, cybernetics, and cultural critique, coining concepts such as toxic resilience, generative resistance, and reactive resistance as key frameworks for understanding and navigating crisis.

Building on this theoretical foundation, I additionally reflected on the formation of *Pirarán*, a networked music ensemble created with my collaborators Iván López and Diego Villaseñor, which embodies a baroque ethos of care and mutual aid. Extending the analysis of disaster to the global music scene, I argued that artistic production, mirroring broader social crises, is trapped in a crisis of its own making. Music-making requires to be reformulated and, while doing so, it responds to the crises it reflects. *Pirarán* rejects toxic resilience in favor of generative and reactive resistance, refusing to participate in content creation for technological enclosures like social media and Spotify and repurposing technological means as technologies of the non-self to move beyond disaster towards communal forms of music-making. Finally, I introduced algorithmic acid communism as the mode of production envisioned within our practice of networked music and live coding.

Importantly, the ideas explored in Chapter 3 find many resonances with those of Chapter 1. The double consciousness, transformed into an impetus towards communism in the TimekNot explorations, manifests as the people protagonising the stories within the opera's acts. While this is developed further

through the concepts of reactive and generative resistance, it is anticipated in Chapter One's section on double consciousness and triple movements and its description of Mexico City's relationship with whititude and baroque ethos. TimekNot from Chapter 2 became the infrastructure for Pirarán's exploration, leading to the execution of *La Fábrica Colapsada*, from which a style of music improvisation emerged that I have called radical polyphony: a music style rooted in the conceptual principles of the language and the networked conditions in which we operate. Ultimately, *Pirarán* embodies the central ideas explored throughout this dissertation: the social relations within a musical practice mediated by networked, live-coding technology based on care and mutual aid capable of adumbrating unforeseen modes of production and regimes of representation.

Faced by crises, the networked ensembles here discussed – both my collaborations with Rolando, Iván, and Diego – opened a small space-time to suspend empty, accelerating time and to propose a slow, constant, multiple, non-linear, texture of time. This texture of time is the outcome of focusing on social relations above technological research and artistic creation, assuming that within such relations new forms of art and technology can emerge. Thus, these ensembles transform crises into a critical period where new connections can be imagined and created. In a world where crises are becoming more and more the most frequent life experience, I have found it useful to inverse the discourse around them: the crises are the norm, and niches and moments where redistribution of wealth and social justice emerge are rare and valuable as well as real and concrete. The challenge related to the crisis of the imagination mentioned at the introduction of this work has been to lift these critical periods into artistic visions and networked infrastructure capable of harnessing crises to come into more critical periods.

This dissertation reveals some interesting insights regarding live coding's thinking-in-action. According to Cocker, live coding's "[...] mode of thought is less concerned with the development of theoretical knowledge (theoria), nor solely with a mode of making or production (poiēsis), but rather its thinking-in-action is inherently related to the enactment and exercise of a politics of action, moreover, of ethical-political action" (Cocker, 2016, p. 106). Similarly, understanding performance as a theatre of agency – the stage as a space where the human and the computational perform/negotiate/interrogate agency and power (Franco Briones, 2022) – requires expanding its scope to domains outside the fields of technology and art.

Throughout the artworks explored, I have come to understand thinking-in-action as an exercise inexorably bound to a broader sense of life. Hence, the performative thinking-in-action and theatre of agency of live coding is a site of tension immensely capable of theory and production rather than compulsive actionable thinking. Throughout the three chapters of this dissertation, critical theory has acted as the context and output of the artworks. In contrast, the artwork becomes the site of thinking-in-action and an expanded theatre of agency. This is why it has been so relevant to talk about subjectivity, crises, earthquakes, disasters, cities, tuning systems, polytemporality, convergence, mestizaje, resistance, resilience, trauma, acid communism, colonialism, capitalism, Spotify, fascism, double consciousness, technologies of the non-self, political economy, care, mutual aid, and so on.

The networked music ensembles involved in these artworks and the software developed for this project both function as sites of politico-economic analysis and critique and as a vision of a mode of production yet to be experienced in broader social settings. This is why it has been relevant to publicly share – with copyleft licenses – all infrastructure related to these artworks. It has also been important to establish the

ensembles and collaborations beyond programming expertise or (social-) networking opportunities - based on conversations mediated by care and mutual aid. This is why the collective, networked live coding discussed here anticipates shifts in the current mode of production and regimes of representation.

A common misinterpretation of the thinking-in-action of coding-as-performance appears when Attali's argument on noise as a political economy is sanitised from its historic-materialist components. I call stage fetishism the magical notion that the stage creates, ex-nihilo, conditions for the emergence of the unforeseen, the new and the experimental: that, somehow, the actions on stage, thus artistic and computational thought, inherently transform reality. However, performing agency and thought process remain locked in a feed-back-and-forth dialectic loop with theory and, as Meiksins would claim, the organisation of material life and social reproduction. This means that whatever appears on stage is a manifestation of existing modes of production, its background conditions and regimes of representation. These appearances act as either a critique of what we have, a reification of the status quo, a proposal of what could be instead, or something else, perhaps.

Beyond live coding, this dissertation demonstrates that art practices, especially those bound to technological development, are not inherently liberatory or social. Rather, art and music-making that aspire to be emancipatory require an explicit connection to the world given by a way of thinking beyond techno-scientific and artistic scholarship. Throughout this project, I have explored how and argued why the arts and sciences require a mediation that can be characterised as emancipatory. Such emancipatory mediation is not an afterthought nor a superficial layer to be appended to what is often considered serious research, but it is the core by which such research becomes relevant to the world.

This scholarship has opened several pathways for future research. The most immediate area to expand is the development of TimekNot and its infrastructure. TimekNot remains functional at its most basic level, demonstrating what can be done with polytemporality and radical polyphony. However, the widgets to visualise its activity (its standalone version) and its higher order functionalities that multiply its effects and economise its syntax are still to be implemented. In the not-so-distant future, Pirarán plans to play a set with the three of us using only TimekNot once the references and the more complex ideas can be played.

The infrastructure explored and created for *Temazcal 2* and *La Fábrica Colapsada* remain as prototypes. Concrete projects should be tackled to socialise software that is useful for people interested in networked music and live coding. The concepts of latency-native ensembles should be expanded within the paradigm of polytemporality utilising the concept of echoic distance and implemented as software. Software designed to adapt multichannel audio to networked music should be created by tapping more profoundly into insights on Ambisonics. Software that preserves and manages the dynamic relationship between audio signals in different local conditions and setups would advance networked music. *Macehualcopa* as a natural language target for a MiniTidal translation is still a project that could substantially impact the inclusion of Indigenous Peoples into live coding. Better and more profound explorations of networked instrument creation should be developed to keep entangling live coding with haptic music creation. Intra-ensemble communication must be re-imagined through the concept of psychedelic communication, and forms of interplay and ensemble coordination – beyond hard synchronisation – need to be further investigated. In general, a whole framework for post-pandemic networked music should be fully conceptualised.

The networked conditions of *Pirarán* and my collaboration with Rolando for *Temazcal 2* shed some light on the authorship of an artwork. It is clear that an artwork that intends to be performed live in more than one location at the same time becomes effectively unique in terms of authorship: not quite like different artworks, not entirely different versions of the same artwork. In this case, the fundamental difference is the diverging aural experiences of Iván, Diego, Rolando, and me. For example, while the version of *La Fábrica Colapsada* at the Networked Imagination Lab (NIL) is mine since it is shaped around my listening experience in the NIL and its technical, social and acoustic conditions, the versions at Iván's studio and *Estudio Piracantos* belong to them, respectively. This mostly impacts the way documentation and post-production are managed. However, as discussed and agreed upon, the benefits of any of these versions will always be distributed equally among us. Once established that each version of a networked performance is bound to an aural experience, consideration must be given to transmitting and exchanging data – in the form of digital audio or code. And finally, how the data transmitted from different locations is embodied as a performing human presence to avoid the instrumentalisation of the other. Perhaps there needs to be further discussion around avoiding confusing human performers with artificial agents, an unfortunate sign of the times. Questions about surplus enjoyment of performances arise as well: while a person at the live location might receive immediate feedback - praise, criticism, as well as the fun and frivolity of the in-person performance - from a live audience, ensemble participants who are in distant studio conditions at the same time might be alienated from it.

Moreover, questions on transnational networked music ensembles and border regimes amidst a rising era of retrieving globalisation, resurgent nationalism, and re-entrenching isolationism need to be explored. Less relevant, questions of concert etiquette around mobile devices might also be important to

tackle. These issues require attention as research and should be registered as specialised literature on network music, labour studies, and international relations. Indeed, the intersection of labour studies, networked technology, programming, and art creation is a relevant site to interrogate further matters of proletarianisation, colonialism, and capitalist expansion described in this dissertation. Even further, this intersection can reveal an artistic consciousness within unions capable of envisioning both the abolition of work and (institutionalised) art, as well as how to generalise the notion of the strike beyond the union, with perhaps a goal to enable a conscious suppression of the imagination of non-unionised, "flexible," creative labour so management in the service of the industry faces the horror of a realism without mediations.

Based on the analysis of the current moment of music-making, Spotify, and AI, a broad cultural analysis beyond the scope of this dissertation is yet to be realised. Such an analysis may portray a historical moment in which a third culture, like the non-aligned movement of the twentieth century, emerges globally on the horizon of what is possible. Contrastingly, an inverse renaissance describes our current pathway, where the intersection of the arts and sciences without emancipation leads us into a neo-feudal, techno-fascist society: not exactly a dark age, but a blinding, neon-lite age, like how I imagine the torture rooms at Guantanamo Bay, resonating with loud music systems running all the time for all time – a brighter world to be avoided.

Bibliography

A. Gil, Y. E. (2018). *Un Nosotros Sin Estado*. Ediciones ONA.

Abel, M. (2014). *Groove: An aesthetic of measured time*. Brill.

Agamben, G. (2005). *State of Exception*. University of Chicago Press.

Agamben, G., & . (2020, May 23). Requiem for the Students. *Diario Della Crisi*. https://d-dean.medium.com/requiem-for-the-students-giorgio-agamben-866670c11642

Aguilar Gil, Y. E. (2020). *Ää Manifiestos Sobre La Diversidad Lingüística*. Almadía Editoria.

Alcántara-Ayala, I. (2019). Desastres en México: Mapas y apuntes sobre una historia inconclusa. *Investigaciones Geográficas*, *100*. https://doi.org/10.14350/rig.60025

Arom, S. (1991). *African Polyphony and Polyrhythm: Musical Structure and Methodology* (M. Thom, B. Tuckett, & R. Boyd, Trans.). Cambridge University Press; Cambridge Core. https://doi.org/10.1017/CBO9780511518317

Attali, J., foreword Jameson, F., afterword McClary, S., & trans. Massumi, B. (1985). *Noise: The political economy of music*. University of Minnesota Press.

Bankoff, G., Frerks, G., & Hilhorst, D. J. M. (2004). Mapping Vulnerability, Disasters, Development and People. *Mapping Vulnerability: Disasters, Development and People*. https://doi.org/10.4324/9781849771924.

Bauer, D., & Malik, S. (2023). Xenotemporality and Time-Complexes. In *Catastrophe Time!* (pp. 210–225). Strange Attractor Press.

Becket, A. (2009). *When the lights went out: Britain in the seventies*. Faber and Faber.

Benjamin, W. (1989). *Theses on the Philosophy of History*. https://api.semanticscholar.org/CorpusID:141079626

Bhattacharya, T., & Vogel, L. (2017). Introduction. In T. Bhattacharya (Ed.), *Social Reproduction Theory* (pp. 1–20). Pluto Press; JSTOR. https://doi.org/10.2307/j.ctt1vz494j.5

Blackwell, A. F., Cocker, E., Cox, G., McLean, A., & Magnusson, T. (Eds.). (2022a). Notation. In *Live Coding: A User's Manual* (pp. 125–158). The MIT Press. https://doi.org/10.7551/mitpress/13770.003.0012

Blackwell, A. F., Cocker, E., Cox, G., McLean, A., & Magnusson, T. (Eds.). (2022b). What Does Live Coding Want? In *Live Coding: A User's Manual* (p. 0). The MIT Press. https://doi.org/10.7551/mitpress/13770.003.0012

Borzacchiello, E. (2017). MEMORIA A PARTIR DE UN SISMO: LA FÁBRICA DE CHIMALPOPOCA. *Memoria: Revista de Crítica Militante.* https://revistamemoria.mx/?p=1839

Bratton, B. (2021). *The Revenge of the Real: Politics for a Post-Pandemic World*. Verso Books.

Carlos, W. (1987). Tuning: At the Crossroads. *Computer Music Journal*, *11*(1), 29–43. JSTOR. https://doi.org/10.2307/3680176

Castañeda, D., & Mendoza, V. T. (1933). *Instrumental precortesiano: Instrumentos de percusión*. Imprenta del Museo Nacional de Arqueología, Historia y Etnografía. https://books.google.ca/books?id=5DVXmwEACAAJ

Castañeda Gutiérrez, I. G., Mendoza Cruz, E. I., Carrillo Valderrama, S. L., Portillo López, G. L., Gutiérrez Gutiérrez, D., Harriss Clare, C. J., & González Muñiz, E. (2018). Los daños de un edificio de interés histórico-comunitario en San Gregorio Atlapulco, Xochimilco. *Rutas de Campo, SEGUNDA ÉPOCA*, *NÚM. 3*(Enero-Junio), 114–121.

Castellanos, M. B. (2017). Introduction. *American Quarterly*, *69*(4), 777–781. JSTOR.

Castle, C. (2017). Spotify's "Fake Artist" Issue and Other Problems at Scale. *Music Tech Solutions*. https://musictech.solutions/2017/07/09/spotifys-fake-artist-issue-and-other-problems-at-scale/

Catrip, K., De Aguinaga Padilla, F., Breña, F., & Santiago, R. (2018). *LOS EFECTOS DEL SISMO DE 19 DE SEPTIEMBRE EN UNA COLONIA PERIFÉRICA POBRE DE LA CIUDAD DE MÉXICO: EL CASO DE SAN GREGORIO ATLAPULCO.*

Caygill, H. (2013). *On Resistance: A Philosophy of Defiance*. Bloomsbury Academic.

Chakrabarty, D. (2000). INTRODUCTION: In *Provincializing Europe* (pp. 3–23). Princeton University Press; JSTOR. http://www.jstor.org/stable/j.ctt7rsx9.5

Chapman, O., & Sawchuk, K. (2015). Creation-as-Research: Critical Making in Complex Environments. *RACAR: Revue d'art Canadienne / Canadian Art Review*, *40*(1), 49–52. JSTOR.

Cocker, E. (2016). Performing thinking in action: The meletē of live coding. *International Journal of Performance Arts and Digital Media*, *12*(2), 102–116. https://doi.org/10.1080/14794713.2016.1227597

Cocker, E. (2018). What now, what next—Kairotic coding and the unfolding future seized. *Digital Creativity*, *29*(1), 82–95. https://doi.org/10.1080/14626268.2017.1419978

Couldry, N., & Mejias, U. A. (2019). Data Colonialism: Rethinking Big Data's Relation to the Contemporary Subject. *Television & New Media*, *20*(4), 336–349. https://doi.org/10.1177/1527476418796632

Coulthard, G. S. (2014a). Conclusion. Lessons from Iddle No More: The Future of Indigenous Activism. In *Red Skin White Masks: Reject the Colonial Politics of Recognition* (p. 220). University of Minnesota Press.

Coulthard, G. S. (2014b). Introduction: Subjects of Empire. In *Red Skin White Masks: Reject the Colonial Politics of Recognition* (p. 220). University of Minnesota Press.

Cox, G., & McLean, A. (2012). Vocable Code. In *Speaking Code: Coding as Aesthetic and Political Expression*. MIT Press. https://doi.org/10.7551/mitpress/8193.003.0006

Disasters: Theory and research (Quarantelli, E. L.). (1978). Sage.

Du Bois, W. E. B. (1997). *The Souls of Black Folk* (David W. Blight and Robert Gooding-Williams). Bedford Books.

Dyer-Witheford, N., Kjøsen, A. M., & Steinhoff, J. (2019). *Inhuman Power*. Pluto Press; JSTOR. https://doi.org/10.2307/j.ctvj4sxc6

Echeverría, B. (1998). *La Modernidad de lo Barroco*. Ediciones Era.

Ek, D. (2024). [X post]. https://x.com/eldsjal/status/1795871513293320204?lang=es

Fanon, F., 1925-1961, author. (2008). *Black skin, white masks*. First edition, New edition. New York : Grove Press, 2008. ©2008. https://search.library.wisc.edu/catalog/9910388221802121

Faramelli, A. (2020). *Resistance, Revolution and Fascism: Zapatismo and Assemblage Politics*. Bloomsbury Publishing.

Farrell, G. L. (2023). Introduction. In *Musical Psychedelia: Research at the Intersection of Music and Psychedelic Experience* (Farrell, G.L. (Ed.)). Routledge.

Federici, S. (2004). *Caliban and the Witch: Women, The Body, and Primitive Accumulation*. Autonomedia.

Federici, S. (2018). *Witches, Witch-Hunting, and Women*. BTL.

Ferguson, S. (2020). *Women and Work*. Between The Lines; JSTOR. https://doi.org/10.2307/j.ctvs09qm0

Fisher, M. (2009). *Capitalist Realism: Is There No Alternative?* Zero Books.

Fisher, M. (2010). Militant Tendencies Feed Music. In *K-Punk: The collected and unpublished writings of Mark Fisher (2004-2016)* (ed. Ambrose, Daniel, pp. 555–561). Zero Books.

Fisher, M. (2014). The Slow Cancellation of the Future. In *Ghosts of my life: Writings on depression, hauntology and lost futures*. Zero books.

Fisher, M. (2017). *Acid Communism* [Book Introduction]. https://my-blackout.com/2019/04/25/mark-fisher-acid-communism-unfinished-introduction/

Forssell Méndez, A. (2020). Una interpretación materialista del mestizaje. *Común*. https://revistacomun.com/blog/una-interpretacion-materialista-del-mestizaje/

Franco Briones, A. (2019). *TIMENOT: A Computational Notation for Time-Oriented Live Coding* [Major Research Project]. McMaster University.

Franco Briones, A. (2022). Live Coding as a Theatre of Agency and a Factory of Time. In *Music and Time: Psychology, Philosophy, Practice* (M. Phillips&M. Sergeant, pp. 112–126). Boydell & Brewer.

Franco Briones, A., & Villaseñor, D. (2020). Poly-temporality: Towards an ecology of time-oriented live coding. *Proceedings of the Fifth International Conference on Live Coding*. International Conference on Live Coding, Limerick, Ireland. https://doi.org/10.5281/zenodo.3939527

Franco Briones, A., & Villaseñor de Cortina, D. (2019). Nanc-in-a-Can Canon Generator. SuperCollider code capable of generating and visualizing temporal canons critically and algorithmically. *Proceedings of the Fourth International Conference on Live Coding*. Proceedings of the Fourth International Conference on Live Coding, Madrid. https://doi.org/10.5281/zenodo.3946192

Fraser, N. (2019). *The Old Is Dying and the New Cannot Be Born*. Verso.

Fraser, N. (2020). *Fortunes of Feminism: From State-Managed Capitalism to Neoliberal Crisis*. Verso Books. https://books.google.ca/books?id=LACREAAAQBAJ

Fraser, N. (2022). *Cannibal Capitalism: How our System is Devouring Democracy, Care, and the Planet and What We Can Do A bout It*. Verso Books.

Gann, K. (1995). *The Music of Conlon Nancarrow*. Cambridge University Press.

Gilbert, J. (2014). *Common Ground: Democracy and Collectivity in an Age of Individualism*. Pluto Press. https://doi.org/10.2307/j.ctt183p7m6

Harkins, J. (2009). *A Practical Guide to Patterns*. chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://opencourses.ionio.gr/modules/document/file.php/DAVA257/Harkins_A_Practical_Guide_to_Patterns.pdf

Harvey, D. (2004). The “New” Imperialism: Accumulation by Dispossession. *Socialist Register*, *Vol. 40: Socialist Register 2004: The New Imperial Challenge*.

Harvey, D. (2005). ch. 4 Accumulation by Dispossession. In *The New Imperialism*. Oxford University Press.

Harvey, D. (2023). *A Companion to Marx’s Grundrisse*. Verso Books.

Hewitt, K. (1983). The Idea of Calamity in a Technocratic Age. In *Interpretation of Calamity From the Viewpoint of Human Ecology*. Geographical Review.

Illner, P. (2021). *Disasters and Social Reproduction: Crisis Response between the State and Community*. Pluto Press. https://doi.org/10.2307/j.ctv19m6198

Juárez Joyner, C. I. (2021). *Los límites narrativos y formales en la práctica del cine documental como posibilidad de representación en la creación de un documental con el pueblo pa ipai* [MAESTRO EN CINE DOCUMENTAL]. UNAM.

Jun-Wei, B., Zuo-Ju, W., Zhi-Jia, W., Fang, O., & Yuan, C. (2014). Wavelet Transform and Its Application in Earthquake Engineering. *Fifth International Conference on Intelligent Systems Design and Engineering Applications, Hunan, China*, 1126–1128. https://doi.org/10.1109/ISDEA.2014.250.

Kirkbride, R. (2016). FoxDot: Live Coding with Python and SuperCollider. *International Conference on Live Interfaces*. International Conference on Live Interfaces, Sussex.

Klawans, J. (2024). Spotify has an issue with “fake artists.” *The Week*. https://theweek.com/tech/spotify-fake-bands

Klein, N. (2007). *The Shock Doctrine: The rise of disaster capitalism*. Metropolitan Books/Henry Holt and Company.

Krouse, S. (2018). Explicitly Comprehensible Functional Reactive Programming. *REBLS*.

Lagunes Huerta, L. (2017, October 5). Continúan contradicciones en información sobre predio de Chimalpopoca. *Cimac Noticias*. http://www.cimacnoticias.com.mx/etiqueta/predio-chimalpopoca

Leal Martínez, A. (2014). De pueblo a sociedad civil: El discurso político después del sismo de 1985. *Revista mexicana de sociología*, *76*(3), 441–469.

Loingsigh, A. N. (2019). Coevalness. In C. Forsdick, Z. Kinsley, & K. Walchester (Eds.), *Keywords for Travel Writing Studies: A Critical Glossary* (pp. 45–47). Anthem Press; Cambridge Core. https://www.cambridge.org/core/product/C7D1DB64D3A52CA005BD3F26F0582D6E

Lowman, E. B., & Barker, A. J. (2015). *Settler: Identity and Colonialism in 21st Century Canada*. Fernwood Publishing.

Luxemburg, R. (2003). *The Accumulation of Capital*. Routledge.

Luxemburg, R. (2004). The Junius Pamphlet. In *The Rosa Luxemburg Reader* (Hudis, Peter; Anderson, Kevin B., pp. 312–341). Monthly Review Press.

Madrid, A. L. (2008). *Sounds of a Modern Nation: Music, Culture, and Ideas in Post-revolutionary Mexico*. Temple University Press.

Manovich, L. (2009). *On Totalitarian Interactivity (Notes from the Enemy of the People)*. https://manovich.net/index.php/projects/on-totalitarian-interactivity

Martini, A., & Sharma, N. (2022). Framing the sublime as affect in post-disaster tourism. *Annals of Tourism Research*, *97*. https://doi.org/10.1016/j.annals.2022.103473

Marx, K., & trans. Nicolaus, M. (1993). *Grundrisse: Foundations of the Critique of Political Economy*. Penguin.

Maté, G. (Director). (2022). DAHLIA YoutTube Channel [Broadcast]. In *Dr. Gabor Maté tells Dahlia social media rewards people for being fake: A Genocide of Authenticity*. https://www.youtube.com/watch?v=PIdgaSX7ZEg

Maté, G., & Maté, D. (2022). *The Myth of Normal: Trauma, Illness, & Healing in a Toxic Culture*. Avery.

McLean, A., Rohrhuber, J., & Wieser, R. (2023). The Meaning of Live: From Art Without Audience to Programs Without Users. *International Conference on Live Coding*. International Conference on Live Coding (ICLC2023), Utrecht, Netherlands. https://doi.org/10.5281/zenodo.7843567

McLean, A., & Wiggins, G. (2010). Tidal—Pattern Language for the Live Coding of Music. *Proceedings of the 7th Sound and Music Computing Conference 2010*. Sound and Music Computing conference 2010, Barcelona. https://doi.org/10.5281/zenodo.849841

Medhurst, J. (2014). *That Option No Longer Exists: Britain 1974-76*. Zero Books. https://books.google.ca/books?id=tbG-oAEACAAJ

Medina, E. (2014). *Cybernetic Revolutionaries: Technology and Politics in Allende's Chile*. MIT Press.

Meiksins Wood, E. (1995). History or Technological Determinism. In *Democracy Against Capitalism*. Cambridge University Press.

Mitchell, J. M., Bogenshutz, M., Lilienstein, A., Harrison, C., Kleiman, S., Parker-Guilbert, K., Ot'alora G., M., Garas, W., Paleos, C., Gorman, I., Nicholas, C., Mithoefer, M., Carlin, S., Poulter, B., Mithoefer, A., Quevedo, S., Wells, G., Klaire, S. S., van der Kolk, B., … Doblin, R. (2021). MDMA-assisted therapy for severe PTSD: a randomized, double-blind, placebo-controlled phase 3 study. *Nature Medicine*, *27*, 1025–1033. https://doi.org/10.1038/s41591-021-01336-3

Naranjo, I. (2017). *ASSEMBLAGE, RECURSION, AND FLEXIBLE STRUCTURES IN THREE RECENT PIECES* [DOCTOR OF MUSICAL ARTS]. Stanford University.

Nardou, R., Eastman M., L., Rothhaas, R., Xu, R., Yang, A., Boyden, E., & Dölen, G. (2019). Oxytocin-dependent reopening of a social reward learning critical period with MDMA. *Nature*, *569*, 116–120. https://doi.org/10.1038/s41586-019-1075-9

Navarrete Linares, F. (2016). *México racista: Una denuncia* (Primera edición). Grijalbo; WorldCat.

Nemchenko, L. (2017). Montage as the Meaning-generative Principle of Avant-garde: From Montage in Cinema to Montage in Theatre (Soviet and Post-Soviet Theatre and Cinema). *Convention 2017 "Modernization and Multiple Modernities" (ISPS Convention 2017)*. ISPS Convention 2017, Dubai.

Nicholls, D. (1996). Henry Cowell's "New Musical Resources." In H. Cowell (Ed.), *New Musical Resources* (pp. 153–174). Cambridge University Press; Cambridge Core. https://doi.org/10.1017/CBO9780511597329.018

Noble, J., & Biddle, R. (2004). Notes on notes on postmodern programming. *SIGPLAN Not.*, *39*(12), 40–56. https://doi.org/10.1145/1052883.1052890

O'Brien, S. (2017). Resilience Stories: Narratives of Adaptation, Refusal, and Compromise. *Resilience: A Journal of the Environmental Humanities*, *4*(2–3), 43–65. https://doi.org/10.5250/resilience.4.2-3.0043

Ogborn, D., Beverley, J., Brown-Hernandez, N., Franco Briones, A., Gray, B., MacLean, A., N. del Angel, L., Oduro, K., Park, S., Roberts, A., Rodríguez, J., Sicchio, K., Stewart, D. A., Testa, C., & Tsabary, E. (2022). Estuary 0.3: Collaborative audio-visual live coding with a multilingual browser-based platform. *Web Audio Conference 2022 (WAC2022)*. Web Audio Conference 2022, Cannes, France. https://doi.org/10.5281/zenodo.6767377

Pätzold, C. (2014). Aspects of Temporal Organization in Brian Ferneyhough's 'Carceri d'Invenzione III. *Journal for New Music and Culture*, *8*. http://www.searchnewmusic.org/paetzold.pdf

Polanyi, K. (2001). *The Great Transformation: The Political and Economic Origins of Our Time*. Beacon Press.

Queralt Molina, J. (2023). The complete guide to live-coding visuals in Punctual. *Algorithmic Pattern Salon*. Algorithmic Pattern Salon. https://doi.org/10.21428/108765d1.397b6e6b

Ramírez de Garay, I. (2023). El sismo de 1985 y la deuda externa. Economía política y moral de un desastre. *Historia mexicana*, *73*(2), 831–877. https://doi.org/10.24201/hm.v73i2.4683

Ramos, A. C., & Guerrero, C. (2017, September 25). ¿Quiénes son los muertos de Chimalpopoca? *Pie de Página*. http://piedepagina.mx/quienes-son-los-muertos-de-chimalpopoca.php

Redhead, L. (2022). Nothing Really Changes': Material Processes in and as Timein hearmleoþ-gieddunga. In *Music and Time: Psychology, Philosophy, Practice* (M. Phillips&M. Sergeant). Boydell & Brewer.

Reina, R. (2015). *Applying Karnatic Rhythmical Techniques to Western Music*. Routledge. https://doi.org/10.4324/9781315567402

Robinson, C. J. (2019). *On racial capitalism, Black internationalism, and cultures of resistance*. Pluto Press.

Rodolfo Acosta, José Luis Castillo, Gabriela Ortiz, & Cristian Morales-Ossio (Directors). (2021). *Cátedra Márquez | Mesa de diálogo Colonialismo y descolonización en la música contemporánea* [Video recording]. https://www.youtube.com/watch?v=N5CL-zAuM34

Rohrhuber, J. (2018). Algorithmic Music and the Philosophy of Time. In A. McLean & R. T. Dean (Eds.), *The Oxford Handbook of Algorithmic Music*. Oxford University Press.

Rohrhuber, J., Campo, A. de, & Wieser, R. (2005). Algorithms Today: Notes on Language Design for Just in Time Programming. *International Conference on Mathematics and Computing*. https://api.semanticscholar.org/CorpusID:559371

Saffon Sanín, M. P., Vera, J., Gómez, P., Mora, M., Ortiz, M., & Félix, A. P. (2019). Capítulo quinto: AFECTACIONES EN SAN GREGORIO ATLAPULCO, XOCHIMILCO. In *Contra el desamparo del Estado: Violaciones a los derechos de las personas damnificadas por el sismo 19S*. UNAM, IIJ.

Sahagún, Bernardino de. (1829). *Historia general de las cosas de Nueva España. Tomo Segundo: Libros V - IX* (Carlos María de Bustamante). Impr. del ciudadano A. Valdés.

Satizábal, P., & Melo Zurita, M. de L. (2021). Bodies-holding-bodies: The trembling of women's territorio-cuerpo-tierra and the feminist responses to the earthquakes in Mexico City. *Third World Thematics: A TWQ Journal*, *6*(4–6), 267–289. https://doi.org/10.1080/23802014.2022.2123953

Scarlett, J. P. (2022). The harmful legacy of colonialism in natural hazard risk. *Nat Commun*, *13*(6945). https://doi.org/10.1038/s41467-022-34792-7

Schatz, V. (n.d.). *Wavelet transform in Haskell* [Computer software]. https://www.volkerschatz.com/science/haswavelet.html

Schmidt, S. (2018). Latin American Dependency Theory. *Global South Studies*.

Sicchio, K. (2024). Live Notation for Patterns of Movement. *The Drama Review*, *68*(1), 104–116.

Sorensen, A. (2018). *Extempore: The design, implementation and application of a cyber-physical programming language* [PHD]. 10.25911/5d67b75c3aaf0

Speed, S. (2017). Structures of Settler Capitalism in Abya Yala. *American Quarterly*, *69*(4), 783–790. JSTOR.

Spiegel, L. (1981). *Manipulations of Musical Patterns* (p. 22).

Stiegler, B. (2012). RELATIONAL ECOLOGY AND THE DIGITAL PHARMAKON. *Culture Machine*, *13*. www.culturemachine.net

Subcomandante Marcos. (2001, March 10). *Subcomandante Marcos, entrevista con Julio Scherer* (J. Scherer, Interviewer) [TV]. https://enlacezapatista.ezln.org.mx/2001/03/10/subcamandante-marcos-entrevista-con-julio-scherer/

Thoegersen, P. A. (2022). The Progenitor: Charles Ives's Universe Symphony and Its Legacy: Polytempic Polymicrotonal Art Music. In *Polytempic Polymicrotonal Music*. Routledge.

Toop, R., & Ferneyhough, B. (1995). Ferneyhough, interview with Richard Toop. In *Collected Writings*. Harwood Academic Publishers.

Torres Núñez del Prado, P. (2022). The Sonified Textiles within the Text(il)ura Performance: Cross-cultural Tangible Interfaces as Phenomenological Artifacts VIS. *Nordic Journal for Artistic Research*.

Toussaint, G. (2005). The Euclidean Algorithm Generates Traditional Musical Rhythms. *Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*. Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science, Banff, Canada.

Turati, M. (2017, September 26). La fábrica caída en Chimalpopoca, vieja conocida del gobierno. *Proceso*. https://www.proceso.com.mx/reportajes/2017/9/26/la-fabrica-caida-en-chimalpopoca-vieja-conocida-del-gobierno-192099.html

van der Kolk, B. (2014). *The Body Keeps the Score: Brain, Mind and Body in the Healing of Trauma*. Penguin.

Vasconcelos, J. (1948). *La raza cósmica*. https://api.semanticscholar.org/CorpusID:60479621

Velasco-Pufleau, L. (2018). On Luigi Nono's Political Thought: Emancipation Struggles, Socialist Hegemony and the Ethic Behind the Composition of Für Paul Dessau. *Music & Politics*, *XII*(2). https://doi.org/10.3998/mp.9460447.0012.205

Vilenica, A., Guerra Arjona, F., & Otomí community occupying the offices of the INPI in Mexico City. (2023). This house belongs to everyone: Otomí community occupation of the National Indigenous Peoples' Institute (INPI) in Mexico City as a struggle for dignified housing and the right to the city. *Radical Housing Journal*, *5*(1), 251–263. https://doi.org/10.54825/CSWW4265

Villeda, K. (2018). Tu nombre es el mío. In *Pies en la tierra. Crónicas de septiembre. México*. Huidobro S., Silva Graciela S., y Sánchez Jiménez Raquel.

Vizenor, G. R. (1999). *Manifest Manners: Narratives on Postindian Survivance*. https://api.semanticscholar.org/CorpusID:142065962

Wanke, R. D. (2021). *Sound in the Ecstatic-Materialist Perspective on Experimental Music*. Routledge.

Whitener, B. (2021). Rosa Luxemburg in Mexico: On the Return of Primitive Accumulation. *Critical Sociology*, *48*, 089692052199244. https://doi.org/10.1177/0896920521992444

Wilkins, J. (2023). The Italian Communist Composer Who Wrote Revolutionary Music for the Working Class. *Jacobin*. https://jacobin.com/2023/06/luigi-nono-italy-fabbrica-illuminata-workers-music

Wolfe, P. (2006). Settler colonialism and the elimination of the native. *Journal of Genocide Research*, *8*(4), 387–409. https://doi.org/10.1080/14623520601056240

Woo, G. (2023). If Things Have Turned for the Worst: A Catastrophist's Diary. In *Catastrophe Time!* (pp. 92–113). Strange Attractor Press.

Zhang, G. Z. (2023). Catastrophe Time! In *Catastrophe Time!* (pp. 92–113). Strange Attractor Press.

Appendices

A.1 *Temazcal 2* Support Materials

A.1.1. Temazcal 2 video documentation online: https://www.youtube.com/watch?v=RhIQPzgPZhw&t=6s

A.1.2. Temazcal 2 repository online: https://github.com/afrancob/temazcal2

A.2 *La Fábrica Colapsada* Support Materials

A.2.1. Arraymusic presentation 2023: https://youtu.be/RQ8adQk9y7Y?t=3563

A.2.2. Website with the audio documentation and storytelling: https://lfc.piraran.com/

A.2.3. Repository 1: https://github.com/AFrancoB/2017

A.2.4. Repository 2: https://github.com/AFrancoB/la-fabrica-colapsada-visuales

A.2.5. Repository 3: https://github.com/AFrancoB/la-fabrica-colapsada

A.3 TimekNot Support Materials

First performance documented with TimekNot: https://www.youtube.com/watch?v=EgWf3Qql-lo

A.4 TimekNot Source Code

The source code presented here is the first version, the one in which the Chapter 2 is based. More recent versions can be explored in the following repo (especially branch 'shedingCode' where I am currently working). The repository can be found here: https://github.com/afrancob/timeknot

A.4.1 The Main Module

```
module Main where

import Prelude
import Data.Either
import Data.Maybe
import Effect
import Effect.Now
import Effect.Ref
import Effect.Class
import Effect.Console (log)
import Data.Tempo
import Effect.Ref (new,write)
import Data.Traversable

import Data.Rational
import Data.List.Lazy hiding (many,Pattern)
import Data.Array as A

import Data.List as L

import Data.Map as M
import Data.Tuple

import Foreign

import Partial.Unsafe
import Data.Enum
```

```
import Data.DateTime (DateTime(..))

import Data.Newtype

-- import Halogen as H
-- import Halogen.Aff as HA
-- import Halogen.HTML as HH
-- import Halogen.HTML.Properties as HP
-- import Halogen.HTML.Events as HE
-- import Halogen.VDom.Driver (runUI)

-- import Visualisation
-- import Svg.Parser

import AST
import TimePacketOps
import Parser
import Parser
import Novus

import Parsing


launch:: {} -> Effect TimekNot
launch _ = do
  log "timekNot: launch"
  ast <- new $ L.fromFoldable [TimeExpression  M.empty]
  tempo <- newTempo (1 % 1) >>= new
  eval <- nowDateTime >>= new
  vantageMap <- new $ (M.empty)
  pure { ast, tempo, eval, vantageMap}

-- { zone :: Int, time :: Number, text :: String }
define :: TimekNot -> { zone :: Int, time :: Number, text :: String } -> Effect {
success :: Boolean, error :: String }
define tk args = do
  log "timekNot: evaluate"
  -- program <- read tk.ast -- this does not do anything, can be erased...?
  currentVM <- read tk.vantageMap
  log $ "currentVM" <> show currentVM
  tempo <- read tk.tempo
  eval <- nowDateTime
  let pr = check' currentVM $ runParser args.text parseProgram
  case pr of
    Left error -> pure $ { success: false, error }
    Right p -> do
      write eval tk.eval
      write p tk.ast
      write (processVantage (getVantageMap p) currentVM eval tempo) $ tk.vantageMap
      pure $ { success: true, error: "bad syntax" }
```

```
check':: VantageMap -> Either ParseError Program -> Either String Program
check' vm (Left error) = Left $ parseErrorMessage error
check' vm (Right aProgram) = case check vm aProgram of
                               true -> Right aProgram
                               false -> Left "failed the check, time bites it's own
tail"

-- { zone :: Int, windowStartTime :: Number, windowEndTime :: Number }
render:: TimekNot -> {zone :: Int, windowStartTime :: Number, windowEndTime ::
Number} -> forall opts. Effect (Array Foreign)
render tk args = do
    let ws = numToDateTime (args.windowStartTime * 1000.0000) -- haskell comes in
milliseconds, purescript needs seconds
    let we = numToDateTime (args.windowEndTime * 1000.0000)
    program <- read tk.ast
    vantageMap <- read tk.vantageMap
    -- log $ "vm: " <> show vantageMap
    t <- read tk.tempo
    eval <- read tk.eval
    let tp = assambleTimePacket ws we eval t vantageMap
    -- log $ show program
    -- log $ show ws
    -- log $ show ws
    -- log $ show t
    programToForeign program tp

    -- events <- programToWaste program tp
    -- log $ show events
    -- pure $ map unsafeToForeign events

setTempo :: TimekNot -> ForeignTempo -> Effect Unit
setTempo tk t = do
  -- log $ "setTempo is called" <> show (fromForeignTempo t)
  write (fromForeignTempo t) tk.tempo
```

A.4.2 Voices

```
module Voices (programToForeign) where

import Preludeimport Effect (Effect)
import Effect.Console
import Data.Map as M
import Data.Array (filter,fromFoldable,(!!), zipWith, replicate, concat, (..), (:),
init, tail, last,head,reverse,zip, cons, snoc, length, singleton, splitAt)
import Data.TraversableWithIndex
import Foreign
import Data.Tempo

import AST
```

```
import Parser -- getTemporalMap
import Aural  -- getPitchXMap  --- this two functions do not seem to belong in there
import TestOpsAndDefs
import TemporalSpecs
import AuralSpecs

-- glosario:
---- Voice is an ongoing or finished or yet-to-be-played musical idea enabled by
---- Block is a rhythmic pattern of onsets
---- Onset a moment in time when a sound is instantiated
---- An Event is an Onset with an Index
---- Index is a mark that allows me to identify the position of the onset so sounds
and sound characteristics can be attached to it
---- process-oriented index: an int identifier for each onset on a flow of onsets.
---- eventIndex is the way I will refer to process oriented indexes
---- structure-oriented index: an int identifier for each segment on a voice and an
array to identifier internal events in a voice: The head is the 'natural'
subdivisions of the voice, each new element in the array is a new subdivision
---- a structure oriented index has a voice index and a structure index. A voice
index is an Int while the Structure Index is an Array Int. The notation I have made
for the structure oriented index is: 3-0.2.4  to the left of the (-) is the voice
index and to the right of it is the event position in the rhythmic idea. The head of
the array is the top level of the nested subdivisions and the last is the deepest
level of the subdivisions.

programToForeign:: Program -> TimePacket -> Effect (Array Foreign)
programToForeign program timePacket = concat <$> calculatedVoices -- waste
  where voices' = programToVoice program -- Voices
        xenoPitches = getXPitchMap program -- Map String XenoPitch
        calculatedVoices = fromFoldable <$> M.values <$> calculateVoices
(getTemporalMap program) voices' xenoPitches timePacket

programToVoice:: Program -> Voices
programToVoice program = M.intersectionWith (\x y -> Voice x y) tempoMap auralMap
  where tempoMap = getTemporalMap program
        auralMap = getAuralMap program

calculateVoices:: M.Map String Temporal -> Voices -> M.Map String XenoPitch ->
TimePacket -> Effect (M.Map String (Array Foreign)) -- (M.Map String (Array
AlmostWaste))
calculateVoices tempoMap voiceMap xenopitches tp = traverseWithIndex (calculateVoice
tempoMap voiceMap xenopitches tp) voiceMap  -- to get rid of Effect, change
traverseWithIndex to mapWithIndex

calculateVoice:: M.Map String Temporal -> Voices -> M.Map String XenoPitch->
TimePacket -> String -> Voice -> Effect (Array Foreign)-- (Array AlmostWaste)
calculateVoice tempoMap voiceMap xenopitches tp aKey (Voice temporal aurals) = do
    let events = calculateTemporal tempoMap tp aKey temporal -- Array Event
    let rhythmic = getRhythmic tempoMap temporal
    events >>= (auralSpecs voiceMap rhythmic aurals xenopitches)
```

A.4.3 TimePacketOps

```
module TimePacketOps (assambleTimePacket,secsFromOriginAtVantage, secsFromOriginAtWS,
secsFromOriginAtWE, secsFromOriginAtEval,metricFromOriginAtWS, metricFromOriginAtWE,
metricFromOriginAtEval, voiceFromOriginToEval, fromDateTimeToPosix,
fromDateTimeToPosixMaybe, numToDateTime) where

import Prelude
import Data.Maybe
import Data.Newtype
import Data.Tempo
import AST
import DurationAndIndex
import Data.Rational (Rational(..), (%), fromInt, toNumber)
import Data.DateTime
import Data.DateTime.Instant
import Data.Time.Duration
import Data.Map (Map, lookup)

assambleTimePacket:: DateTime -> DateTime -> DateTime -> Tempo -> VantageMap ->
TimePacket
assambleTimePacket ws we eval t v = {ws: ws, we: we, eval: eval, origin: origin t,
tempo: t, vantageMap: v}

numToDateTime:: Number -> DateTime
numToDateTime x =
      let asMaybeInstant = instant $ Milliseconds x -- Maybe Instant
          asInstant = unsafeMaybeMilliseconds asMaybeInstant
      in toDateTime asInstant

unsafeMaybeMilliseconds:: Maybe Instant -> Instant
unsafeMaybeMilliseconds (Just x) = x
unsafeMaybeMilliseconds Nothing = unsafeMaybeMilliseconds $ instant $ Milliseconds
0.0

secsFromOriginAtVantage:: TimePacket -> String -> Number
secsFromOriginAtVantage tp k = vPosix - oPosix
    where oPosix = fromDateTimeToPosix tp.origin
          v = fromMaybe tp.tempo.time $ lookup k tp.vantageMap
          vPosix = fromDateTimeToPosix v

          vPosix = fromDateTimeToPosix v
secsFromOriginAtWS tp = ws - oPosix
    where oPosix = fromDateTimeToPosix tp.origin
          ws = fromDateTimeToPosix tp.ws

          ws = fromDateTimeToPosix tp.ws
secsFromOriginAtWE tp = we - oPosix
    where oPosix = fromDateTimeToPosix tp.origin
          we = fromDateTimeToPosix tp.we
```

```
secsFromOriginAtEval:: TimePacket -> Number
secsFromOriginAtEval tp = eval - oPosix
    where oPosix = fromDateTimeToPosix tp.origin
          eval = fromDateTimeToPosix tp.eval

metricFromOriginAtWS:: TimePacket -> Number  -- is this needed anyway?
metricFromOriginAtWS tp = originSecsAtWS / voiceDur
    where originSecsAtWS = secsFromOriginAtWS tp -- :: Number
          vTempo = toNumber $ tp.tempo.freq * (60%1) -- hz to bpm
          voiceDur = durInSecs 1.0 vTempo

metricFromOriginAtWE:: TimePacket -> Number
metricFromOriginAtWE tp = originSecsAtWE / voiceDur
    where originSecsAtWE = secsFromOriginAtWE tp -- :: Number
          vTempo = toNumber $ tp.tempo.freq * (60%1) -- htz to bpm
          voiceDur = durInSecs 1.0 vTempo

metricFromOriginAtEval:: TimePacket -> Number -- equivalent (in theory) to
timeToCount ???
metricFromOriginAtEval tp = originSecsAtEval / voiceDur
    where originSecsAtEval = secsFromOriginAtEval tp -- :: Number
          vTempo = toNumber $ tp.tempo.freq * (60%1) -- htz to bpm :: Number
          voiceDur = durInSecs 1.0 vTempo

voiceFromOriginToEval:: TimePacket -> Number -> Number -> Number
voiceFromOriginToEval tp vTempo vUnits = originSecsAtEval / voiceDur
    where originSecsAtEval = secsFromOriginAtEval tp -- :: Number
          voiceDur = durInSecs vUnits vTempo


fromDateTimeToPosix:: DateTime -> Number
fromDateTimeToPosix x = (unwrap $ unInstant $ fromDateTime x)/1000.0000

fromDateTimeToPosixMaybe:: Maybe DateTime -> Maybe Number
fromDateTimeToPosixMaybe (Just x) = Just $ (unwrap $ unInstant $ fromDateTime
x)/1000.0000
fromDateTimeToPosixMaybe Nothing = Nothing
```

A.4.4 Parser

```
module Parser(temporal, check, parseProgram, replica, getTemporalMap, getAuralMap,
test, testP,xPitchExpression, expression, getVantageMap, parseDate, utcA) where

import Prelude

import Data.Identity
import Data.List (List(..), head, tail, elem, (:), concat, (..), range)
import Data.List (fromFoldable, filter) as L
import Data.Array (fromFoldable) as A
```

```
import Data.Either
import Data.Int
import Data.String (take, length)
import Data.Tuple (Tuple(..), fst, snd)
import Data.Map (Map(..), filter, lookup, keys, singleton, fromFoldable,
toUnfoldable, member, unions, empty)
import Data.Maybe (Maybe(..), fromMaybe)
import Data.Set as Set
import Data.Maybe
import Data.Rational (Rational(..), toRational, fromInt, (%))

import Data.DateTime (exactDate, Year(..), Month(..), Day(..))

import Data.FunctorWithIndex (mapWithIndex)

import Data.String.CodeUnits (fromCharArray)
import Data.String (split, Pattern)

import Data.Formatter.DateTime (Formatter, parseFormatString, unformat)
import Data.Formatter.Number (Formatter, parseFormatString, unformat) as N

import Data.DateTime
import Data.DateTime.Instant
import Data.Time.Duration


import Parsing
import Parsing.String
import Parsing.String.Basic
import Parsing.Combinators
import Parsing.Combinators.Array (many)
import Parsing.Language (haskellStyle)
import Parsing.Token (makeTokenParser)

import AST
import Rhythm
import Aural

type P = ParserT String Identity

testP str = runParser str parseProgram

-- ISSUES
---- range of Numbers is absolutely broken. DO NOT USE
---- Make all tests: start testing all the checks: tempoCheck, idCheck
replicaCheck!!!

-- TO DO LIST October 17th:
---- refactor Aural and Value
---- finish refactor of transposeWith

---- implement keyword last DONE
```

```
---- implement copy of temporals: v1 <- v0 DONE
---- implement many aurals for one temporal DONE
---- implement weight

-----
-- minor goals:

-- for now: pitch from the middle east DONE partially:structParser
-- refactor show of my data types. They mostly suck.

-- implementaciones siguientes:
  -- xenopitch -- PATHWAY OPENED. CPS ARE IMPLEMENTED

  -- refer to the most recent version of tempi-purs DONE (instead of pulling it from
the internet I copied the code from the repo) DONE

  -- events specific to concrete indexes: 2-0.1 "cp" -- should generate a cp sound
only at 2-0.1

  -- implement unleash parser

  -- razgado  -- doable after concat
-- r <- razgado 0.2 5 -- donde 0.2 es dur en secs y 5 es numero de notas
-- r.sound = "grandpiano" .speed = _-_ 1 1.1 1.2 1.3 1.4 1.5;

---- doable after razgado
  -- canonise (tms) cp rhythmic <<- vantage (id index (_-4 means every fourth event
in block))
-- v0 <- diverge | xxxx :|
-- c <- canonise cpm(100,200,300,400,500) cp: 5 | xxxxx || <<- v0 _-4

  -- concat temporals  -- Doable soon
-- v0 <- (2 afterEval) (3)              | xxox ||
-- v1 <- (2 afterEval) (1) (v0 3:5) | xxx[xx]ox ||
-- w <- v0 <> v1 :|

  -- refactor auralSpecs!!!!!!

  -- acceleration in unlooped events (how to represent this? and calculate the
durations of the events??)

  -- Start with post-evaluation CPstry utcA,
-- Vantage.build = "first" (100 secsFromEval)
-- Vantage.build = "second" (100 xBeatsFromEval)
-- Vantage.move = "first" (100 secsFromEval)
-- Vantage.move = "second" (100 xBeatsFromEval)
-- Vantage.move = "first" (3 fromCurrentPosition)
-- Vantage.remove = "first"

-- v <- first cFrom tm | xxxx :|
```

```
  -- grand project:
  -- Monoid programs:: Map ZoneIndex Voices
  ----- each zone has its eval time. Every zone accesses temporals and aurals
  ----- trans-zone relationships:
        --  two zones cannot name equally a temporal
        --  priority given to referencing (rather than referenced) zones
        -- what are the implications of this in an ensemble?


parseProgram:: P Program
parseProgram = do
  whitespace
  xs <- many expression
  eof
  pure $ L.fromFoldable xs

expression:: P Expression
expression = do
  _ <- pure 1
  choice [try timeExpression, try aural, try vantageExpression, xPitchExpression]

xPitchExpression:: P Expression
xPitchExpression = do
  _ <- pure 1
  x <- braces $ many $ xPitch
  pure $ XenoPitchExpression $ unions x

xPitch:: P (Map String XenoPitch)
xPitch = do
  _ <- pure 1
  id <- identifier
  _ <- reserved "<-"
  x <- choice [try cpSet] --, try mos, try edo]
  _ <- reserved ";"
  pure $ singleton id x

cpSet:: P XenoPitch
cpSet = do
  _ <- pure 1
  _ <- reserved "cps"
  sz <- natural
  factors <- parens $ many natural
  subsets' <- subsets <|> pure Nothing
  pure $ CPSet sz factors subsets'

subsets:: P (Maybe (Array Subset))
subsets = do
  _ <- pure 1
  _ <- reserved "|"
  xs <- chooseSubset `sepBy` comma
  pure (Just $ A.fromFoldable xs)
```

```
chooseSubset:: P Subset
chooseSubset = choice [try intersection, try difference, try union, includes]

includes:: P Subset
includes = do
  _ <- pure 1
  n <- natural
  pure $ Subset n

union:: P Subset
union = do
  _ <- pure 1
  ns <- natural `sepBy` (reservedOp "u")
  pure $ Unions $ A.fromFoldable ns

intersection:: P Subset
intersection = do
  _ <- pure 1
  a <- natural
  _ <- reservedOp "n"
  b <- natural
  pure $ Intersection a b

difference:: P Subset
difference = do
  _ <- pure 1
  a <- natural
  _ <- reservedOp "c"
  b <- natural
  pure $ Difference a b

--
vantageExpression:: P Expression
vantageExpression = do
  _ <- pure 1
  x <- vantage
  pure $ VantagePointExpression x

vantage:: P (Map String Vantage)
vantage = do
  _ <- pure 1
  id <- identifier
  x <- choice [buildA, moveA, removeA]
  _ <- charWS ';'
  pure $ singleton id x

-- MyID.lift = 20 beats from eval;
buildA:: P Vantage
buildA = do
  _ <- pure 1
  _ <- reserved ".lift"
```

```
  _ <- reservedOp "="
  x <- choice [try beatA, try secsA, utcA]
  pure $ Build x

utcA:: P TimePoint
utcA = do
  _ <- pure 1
  d <- date
  t <- choice [parens timeOfDay, timeOfDay]
  local <- (parens $ local) <|> (reserved "so-called utc") *> pure 0
  tiempo <- liftEither $ parseDate (d <> " " <> t)
  result <- liftMaybe (\_ -> "Not a local time") $ adjust (Hours $ toNumber local)
tiempo
  pure $ UTC result

date:: P String
date = do
  _ <- pure 1
  y <- natural
  m <- identifier
  d <- natural
  pure (show y <> "-" <> m <> "-" <>  show d)

timeOfDay:: P String
timeOfDay = do
  _ <- pure 1
  h <- natural
  _ <- reservedOp ":"
  m <- natural
  _ <- reservedOp ":"
  s <- natural
  pure ((padHour h) <> ":" <> show m <> ":" <>  show s)

local:: P Int
local = do
  _ <- pure 1
  _ <- reserved "so-called utc"
  op <- choice [reservedOp "-" *> pure 1, reservedOp "+" *> pure (-1)]
  n <- natural
  pure (n * op)

padHour:: Int -> String
padHour n = if (length iAsStr) > 1 then iAsStr else "0" <> iAsStr
  where iAsStr = show n

parseFormatter:: Either String Formatter
parseFormatter = parseFormatString "YYYY-MMMM-DD HH:m:s"

parseDate:: String -> Either String DateTime
parseDate s = case parseFormatter of
                Left x -> Left x
                Right x -> unformat x s
```

```
beatA:: P TimePoint
beatA = do
  num <- naturalOrFloat
  _ <- reserved "beats from eval"
  pure $ Beat $ toRat (toNumber' num)

secsA:: P TimePoint
secsA = do
  num <- naturalOrFloat
  _ <- reserved "secs from eval"
  pure $ Secs $ toRat (toNumber' num)

-- MyID.move = 20 beats from eval;
moveA:: P Vantage
moveA = do
  _ <- pure 1
  _ <- reserved ".move"
  _ <- reservedOp "="
  x <- choice [try beatMoveA, secsMoveA]
  pure $ Move x

beatMoveA:: P (Either Rational Rational)
beatMoveA = do
  num <- naturalOrFloat
  _ <- reserved "beats from current"
  pure $ Left $ toRat (toNumber' num)

secsMoveA:: P (Either Rational Rational)
secsMoveA = do
  num <- naturalOrFloat
  _ <- reserved "secs from current"
  pure $ Right $ toRat (toNumber' num)

removeA:: P Vantage
removeA = do
  _ <- pure 1
  _ <- reserved ".remove"
  pure $ Remove

timeExpression:: P Expression
timeExpression = do
  _ <- pure 1
  x <- temporal
  pure $ TimeExpression x

temporal:: P (Map String Temporal)
temporal = do
  _ <- pure 1
  choice [try replica, try polytemporalRelation]

-- inACan:: P (Map String Temporal)
```

```
-- inACan = do
--    _ <- pure 1
--    id <- voiceId
--    _ <- reserved "<-"
--    voice <- voiceId
--    cTo <- choice [try cToLast, try $ parens parsePercenTo, try $ parens
parseProcessTo, parens parseStructureTo]
--    cFrom <- choice [try cFromLast, try $ parens cFromPercen, try $ parens
cFromProcess, parens cFromStructure]
--    tm <- brackets $ many tempoMark <|> pure XTempo -- the alternative should be
same as estuary tempo
--    _ <- charWS '|'
--    r <- rhythmic
--    l <- choice [(strWS "||" *> pure false), (strWS ":|" *> pure true)]
  -- pure $ singleton (fst p) $ Temporal (snd p) r l

-- polytemporalList:: String -> ConvergeTo -> ConvergeFrom -> List TempoMark -> Map
String Polytemporal
-- polytemporalList id cTo cFrom tms =
--    let newIDs = map (\index -> id <> "-" <> (show index)) $ range 0 $ length tms
--    in newIDs


replica:: P (Map String Temporal)
replica = do
  _ <- pure 1
  id <- voiceId
  _ <- reserved "<-"
  id2 <- voiceId
  _ <- semi
  pure $ singleton id $ Replica id2

polytemporalRelation:: P (Map String Temporal)
polytemporalRelation = do
  _ <- pure 1
  p <- choice [try kairos, try metric, try converge]
  _ <- charWS '|'
  r <- rhythmic
  l <- choice [(strWS "||" *> pure false), (strWS ":|" *> pure true)]
  pure $ singleton (fst p) $ Temporal (snd p) r l

kairos:: P (Tuple String Polytemporal)
kairos = do
  _ <- pure 1
  id <- voiceId
  _ <- reserved "<-"
  n <- choice [secsFromEval, atEval]
  tm <- parens tempoMark <|> pure XTempo
  pure $ Tuple id $ Kairos n tm

secsFromEval:: P Number
secsFromEval = do
```

```
  _ <- pure 1
  n <- naturalOrFloat
  _ <- reserved "secsAfterEval"
  pure $ toNumber' n

atEval:: P Number
atEval = do
  _ <- pure 1
  _ <- reserved "atEval"
  pure 0.01

metric:: P (Tuple String Polytemporal)
metric = do
  _ <- pure 1
  id <- voiceId
  _ <- reserved "<-"
  polytemporal <- choice [try divergingMetric, convergingMetric]
  pure $ Tuple id polytemporal

divergingMetric:: P Polytemporal
divergingMetric = do
  _ <- pure 1
  _ <- reserved "diverge"
  tm <- parens tempoMark <|> pure XTempo -- the alternative should be same as estuary
tempo
  pure $ Metric (ProcessTo 0 Origin) (Process 0) tm

convergingMetric:: P Polytemporal
convergingMetric = do
  _ <- pure 1
  cTo <- choice [try $ parens parsePercenTo, try $ parens parseProcessTo, parens
parseStructureTo] <|> (pure $ ProcessTo 0 Snap)
  cFrom <- choice [try $ parens cFromPercen, try $ parens cFromProcess, parens
cFromStructure]
  tm <- parens tempoMark <|> pure XTempo -- the alternative should be same as estuary
tempo
  pure $ Metric cTo cFrom tm

converge:: P (Tuple String Polytemporal)
converge = do
  _ <- pure 1
  id <- voiceId
  _ <- reserved "<-"
  polytemporal <- choice [try diverging, try converging, novus]
  pure $ Tuple id polytemporal

diverging:: P Polytemporal
diverging = do
  _ <- pure 1
  _ <- whitespace
  voice <- voiceId
  _ <- reserved "diverge"
```

```
  tm <- parens tempoMark <|> pure XTempo -- the alternative should be same as estuary
tempo
  pure $ Converge voice (ProcessTo 0 Origin) (Process 0) tm

converging:: P Polytemporal
converging = do
  _ <- pure 1
  _ <- whitespace
  voice <- voiceId -- choice between metricVoice or arbitrary name of a voice
  cTo <- choice [try cToLast, try $ parens parsePercenTo, try $ parens
parseProcessTo, parens parseStructureTo]
  cFrom <- choice [try cFromLast, try $ parens cFromPercen, try $ parens
cFromProcess, parens cFromStructure]
  tm <- parens tempoMark <|> pure XTempo -- the alternative should be same as estuary
tempo
  pure $ Converge voice cTo cFrom tm

novus:: P Polytemporal
novus = do
  _ <- pure 1
  _ <- whitespace
  _ <- reserved "Novus"
  _ <- reservedOp "."
  vantID <- voiceId
  cFrom <- choice [try cFromLast, try $ parens $ cFromPercen, try $ parens $
cFromProcess, parens cFromStructure]
  tm <- parens tempoMark <|> pure XTempo
  pure $ Novus vantID cFrom tm

cFromLast:: P ConvergeFrom
cFromLast = do
  _ <- pure 1
  _ <- strWS "last"
  pure $ Last

cFromPercen:: P ConvergeFrom
cFromPercen = do
  _ <- pure 1
  p <- naturalOrFloat
  _ <- charWS '%'
  pure $ Percen (toNumber' p)

cFromProcess:: P ConvergeFrom
cFromProcess = do
  _ <- pure 1
  e <- natural
  pure $ Process e

cFromStructure:: P ConvergeFrom
cFromStructure = do
  _ <- pure 1
  v <- natural
```

```
  _ <- string "-"
  st <- structParser
  pure $ Structure v st

--
cToLast:: P ConvergeTo
cToLast = do
  _ <- pure 1
  last <- choice [try lastMod, lastSnap, lastOrigin]
  pure last

lastOrigin:: P ConvergeTo
lastOrigin = do
  _ <- pure 1
  _ <- strWS "last"
  pure $ LastTo Origin

lastSnap:: P ConvergeTo
lastSnap = do
  _ <- pure 1
  _ <- strWS "last"
  _ <- reserved "afterEval"
  pure $ LastTo Snap

lastMod:: P ConvergeTo
lastMod = do
  _ <- pure 1
  _ <- reserved "mod"
  m <- natural
  _ <- strWS "last"
  _ <- reserved "afterEval"
  pure $ LastTo (Mod m)

parsePercenTo:: P ConvergeTo
parsePercenTo = do
  _ <- pure 1
  p <- choice [try percenMod, try percenSnap, percenOrigin]
  pure p

parseProcessTo:: P ConvergeTo
parseProcessTo = do
  _ <- pure 1
  c <- choice [try processMod, try processSnap, processOrigin]
  pure c

parseStructureTo:: P ConvergeTo
parseStructureTo = do
  _ <- pure 1
  c <- choice [try structureMod, try structureSnap, structureOrigin]
  pure c

percenOrigin:: P ConvergeTo
```

```
percenOrigin = do
  _ <- pure 1
  n <- naturalOrFloat
  _ <- charWS '%'
  pure $ PercenTo (toNumber' n) Origin

percenSnap:: P ConvergeTo
percenSnap = do
  _ <- pure 1
  n <- naturalOrFloat
  _ <- charWS '%'
  _ <- reserved "afterEval"
  pure $ PercenTo (toNumber' n) Snap

percenMod:: P ConvergeTo
percenMod = do
  _ <- pure 1
  _ <- reserved "mod"
  m <- natural
  n <- naturalOrFloat
  _ <- charWS '%'
  _ <- reserved "afterEval"
  pure $ PercenTo (toNumber' n) (Mod m)

processOrigin:: P ConvergeTo
processOrigin = do
  _ <- pure 1
  n <- natural
  pure $ ProcessTo n Origin

processSnap:: P ConvergeTo
processSnap = do
  _ <- pure 1
  n <- natural
  _ <- reserved "afterEval"
  pure $ ProcessTo n Snap

processMod:: P ConvergeTo
processMod = do
  _ <- pure 1
  _ <- reserved "mod"
  m <- natural
  n <- natural
  _ <- reserved "afterEval"
  pure $ ProcessTo n (Mod m)

structureOrigin:: P ConvergeTo
structureOrigin = do
  _ <- pure 1
  st <- forStructure
  pure $ StructureTo (fst st) (snd st) Origin
```

```
structureSnap:: P ConvergeTo
structureSnap = do
  _ <- pure 1
  st <- forStructure
  _ <- reserved "afterEval"
  pure $ StructureTo (fst st) (snd st) Snap

structureMod:: P ConvergeTo
structureMod = do
  _ <- pure 1
  _ <- reserved "mod"
  m <- natural
  st <- forStructure
  _ <- reserved "afterEval"
  pure $ StructureTo (fst st) (snd st) (Mod m)

forStructure:: P (Tuple Int (Array Int))
forStructure = do
  _ <- pure 1
  v <- natural
  _ <- string "-"
  st <- structParser
  pure $ Tuple v st

structParser:: P (Array Int)
structParser = do
  _ <- pure 1
  xs <- natural `sepBy` string "."
  pure $ A.fromFoldable xs

voiceId:: P String
voiceId = do
    _ <- pure 1
    x <- identifier -- many $ noneOf ['\\','<',' ']
    pure x

tempoMark:: P TempoMark
tempoMark = do
  _ <- pure 1
  x <- choice [try cpm, try bpm, try cps, try ratio, acceleration]
  pure x

acceleration:: P TempoMark -- (~ 1 << 0 range 100cpm, 1000cpm)
acceleration = do
  _ <- pure 1
  _ <- reserved "~"
  freq <- toNumber' <$> naturalOrFloat
  _ <- reserved "<<"
  ph <- toNumber' <$> naturalOrFloat
  _ <- reservedOp "range"
  max <- choice [try cpm, try bpm, try cps, try ratio]
  _ <- reservedOp ","
```

```
  min <- choice [try cpm, try bpm, try cps, try ratio]
  pure $ Sin {osc: toRat freq, min: min, max: max, phase: toRat ph}

cpm:: P TempoMark
cpm = do
  _ <- pure 1
  x <- toNumber' <$> naturalOrFloat
  _ <- reserved "cpm"
  pure $ CPM (toRat x)

bpm:: P TempoMark
bpm = do
  _ <- pure 1
  fig <- figure
  _ <- charWS '='
  x <- toNumber' <$> naturalOrFloat
  _ <- reserved "bpm"

  pure $ BPM (toRat x) fig

figure:: P Rational
figure = do
  n <- natural
  _ <- charWS '/'
  d <- natural
  pure $ toRational n d

cps:: P TempoMark
cps = do
  _ <- pure 1
  x <- toNumber' <$> naturalOrFloat
  _ <- reserved "cps"
  pure $ CPS (toRat x)

ratio:: P TempoMark
ratio = do
  _ <- pure 1
  id <- voiceId
  x <- natural
  _ <- reservedOp ":"
  y <- natural
  pure $ Prop id x y

--
test :: VantageMap -> String -> Either String Program
test vm x =
  case runParser x parseProgram of
    Left (ParseError err _) -> Left err
    Right prog -> case check vm prog of
                    true -> Right prog
                    false -> Left "failed the check"
```

```
getTemporalMap:: Program -> Map String Temporal
getTemporalMap program = unions $ map unexpressTempo $ L.filter (\ expression ->
isTemporal expression) program
  where isTemporal (TimeExpression _) = true
        isTemporal _ = false

unexpressTempo:: Expression -> Map String Temporal
unexpressTempo (TimeExpression x) = x
unexpressTempo _ = empty

getAuralMap:: Program -> Map String (List Aural)
getAuralMap program = toListAurals $ map unexpressAural $ L.filter (\ expression ->
isAural expression) program
  where isAural (AuralExpression _) = true
        isAural _ = false

toListAurals:: List (Map String Aural) -> Map String (List Aural)
toListAurals mapas = unions $ map (\k -> toAurals k vals) $ map fst vals
  where vals = concat $ map toUnfoldable mapas
        toAurals key vals = singleton key $ map snd $ L.filter (\v -> (fst v) == key)
vals

unexpressAural:: Expression -> Map String Aural
unexpressAural (AuralExpression x) = x
unexpressAural _ = empty

--
getVantageMap:: Program -> Map String Vantage
getVantageMap program = unions $ map unexpressVantage $ L.filter (\ expression ->
isVantage expression) program
  where isVantage (VantagePointExpression _) = true
        isVantage _ = false

unexpressVantage:: Expression -> Map String Vantage
unexpressVantage (VantagePointExpression x) = x
unexpressVantage _ = empty

-- test' :: String -> Either String (Map String Temporal)
-- test' x =
--   case getTemporalMap <$> runParser x parseProgram of
--     Left (ParseError err _) -> Left err
--     Right aMap -> Right $ check aMap

check :: VantageMap -> Program -> Boolean
check vm program = checkedTempoAspects && checkedPitch
  where checkedTempoAspects = checkT vm (getVantageMap program) $ getTemporalMap
program
        checkedPitch = checkXPitch program


checkT :: VantageMap -> Map String Vantage -> Map String Temporal -> Boolean
checkT vm vMNew aMap' = checkID && checkTempoMark
```

```
  where aReplicaMap = filter isReplica aMap'
        aMap = filter (not isReplica) aMap'
        checkID = not $ elem false $ mapWithIndex (check2 vm vMNew aMap' Nil) aMap'
        checkTempoMark = not $ elem false $ mapWithIndex (checkTempi aMap Nil) aMap

isReplica:: Temporal -> Boolean
isReplica (Replica _) = true
isReplica _ = false

getReplicaKey:: Temporal -> String
getReplicaKey (Replica id) = id
getReplicaKey _ = "2666"

check2 :: VantageMap -> Map String Vantage -> Map String Temporal -> List String ->
String -> Temporal -> Boolean
check2 _ _ aMap alreadyRefd aKey (Temporal (Kairos _ _) _ _) = true
check2 _ _ aMap alreadyRefd aKey (Temporal (Metric _ _ _) _ _) = true
check2 vm vMNew aMap alreadyRefd aKey (Temporal (Converge anotherKey _ _ _) _ _) =
  case lookup anotherKey aMap of
    Nothing -> false
    Just anotherValue -> case elem aKey alreadyRefd of
                           true -> false
                           false -> check2 vm vMNew aMap (aKey : alreadyRefd)
anotherKey anotherValue

check2 vm vMNew aMap alreadyRefd aKey (Temporal (Novus vantageKey _ _) _ _) =
  case lookup vantageKey vMNew of
    (Just x) -> if isRemove x then false else true
    Nothing -> case lookup vantageKey vm of
                 (Just _) -> true
                 Nothing -> false

-- in case of replica
check2 vm vMNew aMap alreadyRefd aKey (Replica id)
  | aKey == id = false
  | otherwise =
    case lookup id aMap of
         Nothing -> false
         Just nVal -> case elem id alreadyRefd of
                        true -> false
                        false -> check2 vm vMNew aMap (aKey : alreadyRefd) id nVal

isRemove Remove = true
isRemove _ = false

checkTempi:: Map String Temporal -> List String -> String -> Temporal -> Boolean
checkTempi aMap alreadyRefd aKey temporal =
    if (getTempoRef temporal) == Nothing then true
    else case lookup anotherKey aMap of
           Nothing -> false
           Just anotherValue -> case elem aKey alreadyRefd of
                      true -> false
```

```
                    false -> checkTempi aMap (aKey : alreadyRefd) anotherKey
anotherValue
    where anotherKey = fromMaybe "" $ getTempoRef temporal

getTempoRef:: Temporal -> Maybe String
getTempoRef (Temporal (Kairos _ tm) _ _) = isTempoRefd tm
getTempoRef (Temporal (Metric _ _ tm) _ _) = isTempoRefd tm
getTempoRef (Temporal (Converge _ _ _ tm) _ _) = isTempoRefd tm
getTempoRef (Temporal (Novus _ _ tm) _ _) = isTempoRefd tm
getTempoRef (Replica _) = Nothing

isTempoRefd:: TempoMark -> Maybe String
isTempoRefd (Prop id _ _) = Just id
isTempoRefd _ = Nothing

--- negative Numbers
parseNumber:: P Number
parseNumber = choice [
  try $ parens (toNumber' <$> naturalOrFloat),
  try (toNumber' <$> naturalOrFloat),
  negNum
]

negNum:: P Number
negNum = do
  _ <- charWS '-'
  x <- naturalOrFloat
  pure ((-1.0) * toNumber' x)

tokenParser = makeTokenParser haskellStyle
parens      = tokenParser.parens
braces      = tokenParser.braces
identifier  = tokenParser.identifier
reserved    = tokenParser.reserved
naturalOrFloat = tokenParser.naturalOrFloat
natural = tokenParser.natural
float = tokenParser.float
whitespace = tokenParser.whiteSpace
colon = tokenParser.colon
brackets = tokenParser.brackets
comma = tokenParser.comma
semi = tokenParser.semi
integer = tokenParser.integer
stringLiteral = tokenParser.stringLiteral
reservedOp = tokenParser.reservedOp

toNumber':: Either Int Number -> Number
toNumber' (Left x) = toNumber x
toNumber' (Right x) = x

charWS:: Char -> P Char
charWS x = do
```

```
  _ <- pure 1
  x <- char x
  whitespace
  pure x

strWS:: String -> P String
strWS x = do
  _ <- pure 1
  x <- string x
  whitespace
  pure x

toRat:: Number -> Rational
toRat x =
    let pFact = 1000000
        floored = floor x -- 12
        fract = x - (toNumber floored) -- 12.5 - 12.0 = 0.5
        fract' = round $ fract * (toNumber pFact) -- 500000
    in (floored%1) + (fract'%pFact) -- 12 + (500000%1000000)




------ this is an attempt to create a Number range using Formatter
-- getProperDigits:: String -> String -> Either String N.Formatter
-- getProperDigits a b =
--   case (length a' <= 2) && (length b' <= 2) of
--     false -> "not really a number"
--     true -> if a'!0 > b'!0 then
--   where a' = split (Pattern ".") a
--         b' = split (Pattern ".") b

-- compareAB:: Maybe Int -> Maybe Int -> String
-- compareAB (Just a) (Just b) = if a>=b then a "0" else b
-- compareAB Nothing (Just b) = b
-- compareAB (Just a) Nothing = a
-- compareAB Nothing Nothing = 0

-- parseNumFormatter:: Either String N.Formatter
-- parseNumFormatter = N.parseFormatString "0.000"

-- parseNum:: String -> Either String Number
-- parseNum s = case parseNumFormatter of
--                 Left x -> Left x
--                 Right x -> N.unformat x s
```

A.4.5 Rhythm

```
module Rhythm(rhythmic) where
```

```
import Prelude

import Data.Either
import Data.Identity
import Data.List hiding (many,take)
import Data.List.Lazy (replicate,repeat,take)
import Data.Foldable (foldl)
import Data.Int
import Data.Tuple
import Data.String (singleton, joinWith)
import Data.Maybe hiding (optional)
import Data.Functor
import Control.Monad
import Data.List.NonEmpty (toList)
import Data.Map as M
import Data.String as Str

import Effect (Effect)
import Effect.Console (log)

import Parsing
import Parsing.String
import Parsing.String.Basic
import Parsing.Combinators
import Parsing.Language (haskellStyle)
import Parsing.Token (makeTokenParser)


import AST

type P = ParserT String Identity

-- to do: implement int parser as follows:
-- | <4 4 3> :| -- this should generate: xoooxoooxoo
-- | <4 4 3 , 0> :| -- this should generate: xoooxoooxoo , separates the pattern from
a rotation value
-- | <4 4 3 , 0, xx> :| -- this should generate: xxooooooxxooooooxxoooo, instead of
assuming a pattern (X) the player gives one to the program
-- this should be added to Rhythmic as a constructor:
-- Numeric Rhythmic (Array Int) Int

rhythmic:: P Rhythmic
rhythmic = do
  _ <- pure 1
  x <- choice [try parseRhythmList, try parseSD, try parseRepeat, try parseBjorklund,
parseXO]
  pure x

parseRhythms:: P Rhythmic
parseRhythms = do
  _ <- pure 1
```

```
  choice [try parseRhythmList, try parseSD, try parseBjorklund, try parseRepeat,
parseXO]

parseRhythmList:: P Rhythmic
parseRhythmList = do
  _ <- pure 1
  x <- parseXOorSDorReporBjork
  xs <- toList <$> many1 parseXOorSDorReporBjork
  pure $ Rhythmics $ x:xs

parseXOorSDorReporBjork:: P Rhythmic
parseXOorSDorReporBjork = do
  _ <- pure 1
  choice [try parseSD, try parseRepeat, try parseBjorklund, parseXO]

parseSD:: P Rhythmic
parseSD = do
  _ <- pure 1
  _ <- charWS '['
  x <- parseRhythms
  _ <- charWS ']'
  pure $ Sd x

parseRepeat:: P Rhythmic
parseRepeat = do
  _ <- pure 1
  _ <- charWS '!'
  x <- parseRhythms
  _ <- charWS '#'
  y <- integer
  pure $ Repeat x y

parseBjorklund:: P Rhythmic
parseBjorklund = do
    _ <- pure 1
    x <- choice [try $ parens parseFull, try $ parens parseK, try $ parens
parseSimpleBl, parseInv]
    pure x

parseFull:: P Rhythmic
parseFull = do
  _ <- pure 1
  kPatt <- parseRhythms
  _ <- comma
  invPatt <- parseRhythms
  _ <- comma
  k <- natural
  _ <- comma
  n <- natural
  _ <- optional comma
  o <- natural <|> pure 0
  pure $ Bjorklund (Full kPatt invPatt) k n o
```

```
parseSimpleBl:: P Rhythmic
parseSimpleBl = do
  _ <- pure 1
  k <- natural
  _ <- comma
  n <- natural
  _ <- optional comma
  o <- natural <|> pure 0
  pure $ Bjorklund Simple k n o

parseK:: P Rhythmic
parseK = do
  _ <- pure 1
  p <- bPattern
  pure $ Bjorklund (K p.patt) p.k p.n p.rotate

parseInv:: P Rhythmic
parseInv = do
  _ <- pure 1
  _ <- string "'("
  p <- bPattern
  _ <- string ")"
  pure $ Bjorklund (InvK p.patt) p.k p.n p.rotate

bPattern:: P {patt:: Rhythmic, k:: Int, n:: Int, rotate:: Int}
bPattern = do
  _ <- pure 1
  patt <- parseRhythms
  _ <- comma
  k <- natural
  _ <- comma
  n <- natural
  _ <- optional comma
  o <- natural <|> pure 0
  pure {patt: patt, k: k, n: n, rotate: o}

parseXO:: P Rhythmic
parseXO = do
  _ <- pure 1
  x <- choice [charWS 'x' *> pure X, charWS 'o' *> pure O]
  pure x


charWS:: Char -> P Char
charWS x = do
  _ <- pure 1
  x <- char x
  whitespace
  pure x

strWS:: String -> P String
```

```
strWS x = do
  _ <- pure 1
  x <- string x
  whitespace
  pure x


tokenParser = makeTokenParser haskellStyle
parens      = tokenParser.parens
braces      = tokenParser.braces
identifier  = tokenParser.identifier
reserved    = tokenParser.reserved
naturalOrFloat = tokenParser.naturalOrFloat
natural = tokenParser.natural
float = tokenParser.float
whitespace = tokenParser.whiteSpace
colon = tokenParser.colon
brackets = tokenParser.brackets
comma = tokenParser.comma
semi = tokenParser.semi
integer = tokenParser.integer
stringLit = tokenParser.stringLiteral
```

A.4.6 Aural

```
module Aural(aural,variationsStr, checkXPitch, getXPitchMap, prog) where

import Prelude

import Data.Identity
import Data.List (List(..), head, tail, elem, (:), filter, fromFoldable, (..),
length, zip, concat, mapMaybe)
import Data.Array (fromFoldable, length) as A
import Data.Either
import Data.Int
import Data.Tuple (Tuple(..), fst, snd)
import Data.Map (Map(..), lookup, keys, singleton, toUnfoldable, member, values,
unions)
import Data.Map (fromFoldable) as M
import Data.Maybe (Maybe(..), fromMaybe)
-- import Data.Set as Set
import Data.String as Str

import Data.FunctorWithIndex (mapWithIndex)

import Data.String.CodeUnits (fromCharArray)

import Parsing
import Parsing.String
```

```
import Parsing.String.Basic
import Parsing.Combinators
import Parsing.Combinators.Array (many)
import Parsing.Language (haskellStyle)
import Parsing.Token (makeTokenParser)

import AST
import Rhythm

type P = ParserT String Identity

aural:: P Expression
aural = do
    _ <- pure 1
    x <- parseValues
    _ <- reserved ";" -- this need to be fixed!
    pure $ AuralExpression x -- (Map Strg Aural)

parseValues:: P (Map String Aural)
parseValues = do
    _ <- pure 1
    id <- voiceId
    xs <- many value
    pure $ singleton id (fromFoldable xs)

value:: P Value
value = do
    _ <- pure 1
    _ <- reservedOp "."
    valType <- choice [try sound,try n, try gain, try pan, try speed, try begin, try
end, try vowel, try cutoff, try cutoffh, try inter, try maxw, try minw, try legato,
try orbit, try mayeh, try prog, try xeNotes, xeno]
    pure valType

prog:: P Value
prog = do
    _ <- pure 1
    _ <- choice [reserved "prog"]
    _ <- reservedOp "="
    sp <- parseSpan
    xs <- many idOfPitch
    pure $ Prog sp $ fromFoldable xs

idOfPitch:: P (Tuple String (Maybe Int))
idOfPitch = do
    id <- identifier
    n <- (Just <$> brackets natural) <|> pure Nothing
    pure $ Tuple id n

xeNotes:: P Value
xeNotes = do
    _ <- pure 1
```

```
    _ <- choice [reserved "xnotes"]
    _ <- reservedOp "="
    sp <- parseSpan
    l <- choice [try (fromFoldable <$> parseRangeInt), fromFoldable <$> many natural]
    vars <- variationsInt <|> pure Nil
    pure $ XNotes sp l vars

xeno:: P Value
xeno = do
    _ <- pure 1
    xID <- choice [try shurNot, try shurNot8, xeno']
    _ <- reservedOp "="
    sp <- parseSpan
    xnL <- choice [try (A.fromFoldable <$> parseRangeInt), many natural]
    pure $ Xeno xID sp $ fromFoldable xnL

shurNot8:: P (Tuple String (Maybe Int))
shurNot8 = do
    _ <- pure 1
    _ <- reserved "shurNot8"
    pure $ Tuple "shurNot8" Nothing

shurNot:: P (Tuple String (Maybe Int))
shurNot = do
    _ <- pure 1
    _ <- reserved "shurNot"
    pure $ Tuple "shurNot" Nothing

xeno':: P (Tuple String (Maybe Int))
xeno' = do
    _ <- pure 1
    id <- identifier
    n <- (Just <$> brackets natural) <|> pure Nothing
    pure $ Tuple id n

-- Dastgah

-- Intervals: Bozorg 182; Kuchak 114; Tanini 204; Baghie 90.

-- shur: D Eqb F G Aqb Bb C
-- normalised to 24ET: 0   150      300        500         650        800
1000        1200
-- using intervals:    0   182      114 (296)  204 (500) - 182 (682)- 114 (796) - 204
(1000) - 1200
-- name of interfvals: 0 - Bozorg - Kuchak   -  Tanini   - Bozorg   - Kuchak    -
Tanini  ??? how to get to the Octave?

-- corrected by Mehdad: 0 1.82 2.96 5.0 7.04 7.94 9.98 12.0

mayeh:: P Value
mayeh = do
    _ <- pure 1
```

```
    choice [try shur]

shur:: P Value
shur = do
    _ <- pure 1
    _ <- choice [reserved "shur"]
    _ <- reservedOp "="
    shur <- makeShur
    pure shur

-- Dastgah Span Dastgah

makeShur:: P Value
makeShur = do
    _ <- pure 1
    sp <- parseSpan
    shurList <- choice [try (A.fromFoldable <$> parseRangeInt), many natural]
    pure $ Dastgah sp (Shur $ fromFoldable shurList)


orbit:: P Value
orbit = do
    _ <- pure 1
    _ <- choice [reserved "orbit"]
    _ <- reservedOp "="
    m <- choice [try makeOrbit, transposeOrbit]
    pure m

transposeOrbit:: P Value
transposeOrbit = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedOrbit id n

makeOrbit:: P Value
makeOrbit = do
    _ <- pure 1
    sp <- parseSpan
    nList <- choice [try (A.fromFoldable <$> parseRangeInt), many natural]
    vars <- variationsInt <|> pure Nil
    pure $ Orbit sp (fromFoldable nList) vars

legato:: P Value
legato = do
    _ <- pure 1
    _ <- choice [reserved "legato"]
    _ <- reservedOp "="
    m <- choice [try makeLegato, transposeLegato]
    pure m

transposeLegato:: P Value
transposeLegato = do
```

```
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedLegato id n

makeLegato:: P Value
makeLegato = do
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Legato sp (fromFoldable coLs) vars

--
inter:: P Value
inter = do
    _ <- pure 1
    _ <- choice [reserved "inter"]
    _ <- reservedOp "="
    m <- choice [try makeInter, transposeInter]
    pure m

transposeInter:: P Value
transposeInter = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedInter id n

makeInter:: P Value
makeInter = do
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Inter sp (fromFoldable coLs) vars

--
minw:: P Value
minw = do
    _ <- pure 1
    _ <- choice [reserved "minw"]
    _ <- reservedOp "="
    m <- choice [try makeMinw, transposeMinw]
    pure m

transposeMinw:: P Value
transposeMinw = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedMinW id n

makeMinw:: P Value
makeMinw = do
```

```
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ MinW sp (fromFoldable coLs) vars

--
maxw:: P Value
maxw = do
    _ <- pure 1
    _ <- choice [reserved "maxw"]
    _ <- reservedOp "="
    m <- choice [try makeMaxw, transposeMaxw]
    pure m

transposeMaxw:: P Value
transposeMaxw = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedMaxW id n

makeMaxw:: P Value
makeMaxw = do
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ MaxW sp (fromFoldable coLs) vars

--
cutoffh:: P Value
cutoffh = do
    _ <- pure 1
    _ <- choice [reserved "hcutoff"]
    _ <- reservedOp "="
    cutoffh <- choice [try makeCutOffH, transposeCutOffH]
    pure cutoffh

transposeCutOffH:: P Value
transposeCutOffH = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedCutOffH id n

makeCutOffH:: P Value
makeCutOffH = do
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ CutOffH sp (fromFoldable coLs) vars
```

```
cutoff:: P Value
cutoff = do
    _ <- pure 1
    _ <- choice [reserved "cutoff"]
    _ <- reservedOp "="
    cutoff <- choice [try makeCutOff, transposeCutOff]
    pure cutoff

transposeCutOff:: P Value
transposeCutOff = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedCutOff id n

makeCutOff:: P Value
makeCutOff = do
    _ <- pure 1
    sp <- parseSpan
    coLs <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ CutOff sp (fromFoldable coLs) vars

vowel:: P Value
vowel = do
    _ <- pure 1
    _ <- choice [reserved "vowel"]
    _ <- reservedOp "="
    vowel <- choice [try makeVowel, transposeVowel]
    pure vowel

transposeVowel:: P Value
transposeVowel = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedVowel id n

makeVowel:: P Value
makeVowel = do
    _ <- pure 1
    sp <- parseSpan
    vLs <- choice [many parseVowel]
    vars <- variationsVow <|> pure Nil
    pure $ Vowel sp (fromFoldable vLs) vars

variationsVow:: P (List (Variation String))
variationsVow = do
    _ <- pure 1
    _ <- reserved "&"
    xs <- everyVow `sepBy` (reserved "&")
    pure xs

everyVow:: P (Variation String)
```

```
everyVow = do
    _ <- pure 1
    _ <- reserved "every"
    n <- integer
    sp <- parseSpan
    xs <- choice [many parseVowel]
    pure $ Every n sp $ fromFoldable xs

parseVowel:: P String
parseVowel = do
    _ <- pure 1
    x <- choice [charWS 'a' *> pure "a", charWS 'e' *> pure "e", charWS 'i' *> pure
"i", charWS 'o' *> pure "o", charWS 'u' *> pure "u"]
    pure x

end:: P Value
end = do
    _ <- pure 1
    _ <- choice [reserved "end"]
    _ <- reservedOp "="
    end <- choice [try makeEnd, transposeEnd]
    pure end

-- transposeEndWith:: P Value
-- transposeEndWith = do
--     id <- voiceId
--     with <- parens transNumVal
--     pure $ TransposedEndWith id with

transposeEnd:: P Value
transposeEnd = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedEnd id n

makeEnd:: P Value
makeEnd = do
    _ <- pure 1
    sp <- parseSpan
    spdList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ End sp (fromFoldable spdList) vars

begin:: P Value
begin = do
    _ <- pure 1
    _ <- choice [try $ reserved "begin",reserved "begin"]
    _ <- reservedOp "="
    b <- choice [try makeBegin, transposeBegin]
    pure b

-- transposeBeginWith:: P Value
```

```
-- transposeBeginWith = do
--     id <- voiceId
--     with <- parens transNumVal
--     pure $ TransposedBeginWith id with

transposeBegin:: P Value
transposeBegin = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedBegin id n

makeBegin:: P Value
makeBegin = do
    _ <- pure 1
    sp <- parseSpan
    panList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Begin sp (fromFoldable panList) vars

speed:: P Value
speed = do
    _ <- pure 1
    _ <- choice [reserved "speed"]
    _ <- reservedOp "="
    n <- choice [try makeSpeed, transposeSpeed]
    pure n

-- transposeSpeedWith:: P Value
-- transposeSpeedWith = do
--     id <- voiceId
--     with <- parens transNumVal
--     pure $ TransposedSpeedWith id with

transposeSpeed:: P Value
transposeSpeed = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedSpeed id n

makeSpeed:: P Value
makeSpeed = do
    _ <- pure 1
    sp <- parseSpan
    spdList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Speed sp (fromFoldable spdList) vars

pan:: P Value
pan = do
    _ <- pure 1
    _ <- choice [try $ reserved "pan",reserved "p"]
    _ <- reservedOp "="
```

```
    p <- choice [try makePan, transposePan]
    pure p

-- transposePanWith:: P Value
-- transposePanWith = do
--     id <- voiceId
--     with <- parens transNumVal
--     pure $ TransposedPanWith id with

transposePan:: P Value
transposePan = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedPan id n

makePan:: P Value
makePan = do
    _ <- pure 1
    sp <- parseSpan
    panList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Pan sp (fromFoldable panList) vars

gain:: P Value
gain = do
    _ <- pure 1
    _ <- choice [reserved "gain"]
    _ <- reservedOp "="
    g <- choice [try makeGain, transposeGain]
    pure g

-- transposeGainWith:: P Value
-- transposeGainWith = do
--     id <- voiceId
--     with <- parens transNumVal
--     pure $ TransposedGainWith id with

transNumVal:: P (List (Number -> Number))
transNumVal = do
    _ <- pure 1
    op <- choice [reservedOp "+" *> pure add, reservedOp "*" *> pure mul]
    numList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    pure $ map op $ fromFoldable numList

transposeGain:: P Value
transposeGain = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedGain id n

makeGain:: P Value
makeGain = do
```

```
    _ <- pure 1
    sp <- parseSpan
    gainList <- choice [try (A.fromFoldable <$> parseRangeNum), many parseNumber]
    vars <- variationsNum <|> pure Nil
    pure $ Gain sp (fromFoldable gainList) vars

n:: P Value
n = do
    _ <- pure 1
    _ <- choice [reserved "n"]
    _ <- reservedOp "="
    n <- choice [try makeN, {-try transposeNWith,-} transposeN]
    pure n

-- transposeNWith:: P Value
-- transposeNWith = do
--     id <- voiceId
--     n <- brackets natural <|> pure 0
--     with <- transIntVal
--     pure $ TransposedNWith id n with

transposeN:: P Value
transposeN = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedN id n

makeN:: P Value
makeN = do
    _ <- pure 1
    sp <- parseSpan
    nList <- choice [try (A.fromFoldable <$> parseRangeInt), many natural]
    vars <- variationsInt <|> pure Nil
    pure $ N sp (fromFoldable nList) vars

sound:: P Value
sound = do
    _ <- pure 1
    _ <- choice [try $ reserved "sound",reserved "s"]
    _ <- reservedOp "="
    sound <- choice [try makeSound, transposeSound]
    pure sound

transposeSound:: P Value
transposeSound = do
    id <- voiceId
    n <- brackets natural <|> pure 0
    pure $ TransposedSound id n

makeSound:: P Value
makeSound = do
    _ <- pure 1
```

```
    sp <- parseSpan
    strList <- sampleParser
    vars <- variationsStr <|> pure Nil
    pure $ Sound sp strList vars

--
variationsStr:: P (List (Variation String))
variationsStr = do
    _ <- pure 1
    _ <- reserved "&"
    xs <- everyStr `sepBy` (reserved "&")
    pure xs

everyStr:: P (Variation String)
everyStr = do
    _ <- pure 1
    _ <- reserved "every"
    n <- integer
    sp <- parseSpan
    xs <- sampleParser
    pure $ Every n sp xs

variationsInt:: P (List (Variation Int))
variationsInt = do
    _ <- pure 1
    _ <- reserved "&"
    xs <- everyInt `sepBy` (reserved "&")
    pure xs

everyInt:: P (Variation Int)
everyInt = do
    _ <- pure 1
    _ <- reserved "every"
    n <- integer
    sp <- parseSpan
    xs <- choice [try parseRangeInt, fromFoldable <$> many natural]
    pure $ Every n sp xs

variationsNum:: P (List (Variation Number))
variationsNum = do
    _ <- pure 1
    _ <- reserved "&"
    xs <- everyNum `sepBy` (reserved "&")
    pure xs

everyNum:: P (Variation Number)
everyNum = do
    _ <- pure 1
    _ <- reserved "every"
    n <- integer
    sp <- parseSpan
    xs <- choice [try parseRangeNum, fromFoldable <$> many parseNumber]
```

```
    pure $ Every n sp xs

--
parseSpan:: P Span
parseSpan = do
    _ <- pure 1
    x <- choice [
                   reserved "-_" *>  pure CycleInBlock
                 , try $ reserved "_-" *>  pure CycleBlock
                 , try $ reserved "_-_" *> pure SpreadBlock
                 , reserved "_" *>   pure  CycleEvent
                ]
    pure x
    ---

sampleParser:: P (List String)
sampleParser = do
    sampleNames <- stringLit
    pure $ stringToSamples sampleNames

stringToSamples:: String -> List String -- what to do with commas??
stringToSamples s = fromFoldable $ Str.split (Str.Pattern " ") $ Str.trim s

voiceId:: P String
voiceId = do
    _ <- pure 1
    x <- identifier
    pure x

--

toNumber':: Either Int Number -> Number
toNumber' (Left x) = toNumber x
toNumber' (Right x) = x


charWS:: Char -> P Char
charWS x = do
  _ <- pure 1
  x <- char x
  whitespace
  pure x

strWS:: String -> P String
strWS x = do
  _ <- pure 1
  x <- string x
  whitespace
  pure x

--
parseRangeInt:: P (List Int)
```

```
parseRangeInt = do
    x <- natural
    _ <- reservedOp ".."
    y <- natural
    pure (x..y)

parseRangeNum:: P (List Number)
parseRangeNum = do
    x <- parseSpecialNum
    _ <- reservedOp ".."
    y <- parseSpecialNum
    pure $ specialRange x y

specialRange:: Tuple Int Int -> Tuple Int Int -> List Number
specialRange (Tuple i1 d1) (Tuple i2 d2) = map (\rangeInt -> (toNumber rangeInt) /
10.0) rangeInts
    where n1 = ((toNumber i1) * 10.0) + (toNumber d1) -- Tuple 2 3 will become 23
          n2 = ((toNumber i2) * 10.0) + (toNumber d2) -- Tuple 3 6 will become 36
          rangeInts = ((floor n1)..(floor n2))


parseSpecialNum:: P (Tuple Int Int)
parseSpecialNum = choice [try parseSpecialNum', toSpecial <$> natural]

toSpecial:: Int -> Tuple Int Int
toSpecial n = Tuple n 0

parseSpecialNum':: P (Tuple Int Int)
parseSpecialNum' = do
    x <- natural
    _ <- charWS '.'
    y <- natural
    pure $ Tuple x y

--- negative Numbers
parseNumber:: P Number
parseNumber = choice [
  try $ parens (toNumber' <$> naturalOrFloat),
  try (toNumber' <$> naturalOrFloat),
  negNum
]

negNum:: P Number
negNum = do
  _ <- charWS '-'
  x <- naturalOrFloat
  pure ((-1.0) * toNumber' x)


-- tests

-- test' x =
```

```
--   case runParser x parseProgram of
--     Left (ParseError err _) -> Left err
--     Right aMap -> Right $ check aMap

checkXPitch:: List Expression -> Boolean
checkXPitch exs = (checkXPitch' exs) && (checkProg exs)

checkProg :: List Expression -> Boolean
checkProg expressions = not $ elem false $ map (\kn -> func aXenoPitchMap kn)
listOfPitchID
  where aXenoPitchMap = getXPitchMap expressions
        listOfPitchID = getProgIDs $ getAuralMap expressions

getProgIDs:: Map String (List Aural) -> List (Tuple String (Maybe Int))
getProgIDs aurals = noteIDs
    where noteIDs = concat $ mapMaybe keepProg $ concat $ concat $ values aurals

keepProg:: Value -> Maybe (List (Tuple String (Maybe Int)))
keepProg (Prog _ lista) = Just lista
keepProg _ = Nothing

checkXPitch' :: List Expression -> Boolean
checkXPitch' expressions = not $ elem false $ map (\kn -> func aXenoPitchMap kn)
listOfPitchID
  where aXenoPitchMap = getXPitchMap expressions
        listOfPitchID = getXenoIDs $ getAuralMap expressions

func:: Map String XenoPitch -> Tuple String (Maybe Int) -> Boolean
func mapa (Tuple "shurNot8" Nothing) = true
func mapa (Tuple "shurNot" Nothing) = true
func mapa (Tuple k Nothing) = case lookup k mapa of
                                Nothing -> false
                                Just xn -> true
func mapa (Tuple k (Just n)) = case lookup k mapa of
                                Nothing -> false
                                Just xn -> f xn n

f:: XenoPitch -> Int -> Boolean
f (CPSet s f (Just subs)) indx = indx <= A.length subs
f ShurNot8 _ = true
f ShurNot  _ = true
f _ _ = false

getXenoIDs:: Map String (List Aural) -> List (Tuple String (Maybe Int))
getXenoIDs aurals = noteIDs
    where noteIDs = mapMaybe keepXeno $ concat $ concat $ values aurals

keepXeno:: Value -> Maybe (Tuple String (Maybe Int))
keepXeno (Xeno id _ _) = Just id
keepXeno _ = Nothing

getAuralMap:: Program -> Map String (List Aural)
```

```
getAuralMap program = toListAurals $ map unexpressAural $ filter (\ expression ->
isAural expression) program
  where isAural (AuralExpression _) = true
        isAural _ = false

toListAurals:: List (Map String Aural) -> Map String (List Aural)
toListAurals mapas = unions $ map (\k -> toAurals k vals) $ map fst vals
  where vals = concat $ map toUnfoldable mapas
        toAurals key vals = singleton key $ map snd $ filter (\v -> (fst v) == key)
vals

unexpressAural:: Expression -> Map String Aural
unexpressAural (AuralExpression x) = x
unexpressAural _ = empty

getXPitchMap:: Program -> Map String XenoPitch
getXPitchMap program = unions $ map unexpressPitch $ filter (\ expression -> isXPitch
expression) program
  where isXPitch (XenoPitchExpression _) = true
        isXPitch _ = false

unexpressPitch:: Expression -> Map String XenoPitch
unexpressPitch (XenoPitchExpression x) = x
unexpressPitch _ = empty

-- there are three layers that need to be identified: the string that identifies the
bounded temporal, the Int that identifies the index of the aural, and then I need a
way to identify its type of Value:if it is a sound, gain, speed, etc. For this I
could use the constructor of Value...?

-- checkTransposition:: Voices -> Boolean
-- checkTransposition aMap = not $ elem false $ mapWithIndex (checkTransposition1
aMap Nil) anAuralMap
--     where anAuralMap = mapMaybe (\(Voice _ a) -> Just a) aMap


-- checkTransposition1:: Voices -> List (Tuple String Int) -> String -> List Aural ->
Boolean
-- checkTransposition1 aMap refd id aurals = not $ elem false $ mapped
--     where zipped = zip aurals (0..(length aurals))
--           mapped = map (checkTransposition2 aMap refd id) zipped

-- checkTransposition2 :: Voices -> List (Tuple String Int) -> String -> Tuple (List
Value) Int -> Boolean
-- checkTransposition2 aMap refd id aurals =




tokenParser = makeTokenParser haskellStyle
```

```
parens      = tokenParser.parens
braces      = tokenParser.braces
identifier  = tokenParser.identifier
reserved    = tokenParser.reserved
naturalOrFloat = tokenParser.naturalOrFloat
natural = tokenParser.natural
float = tokenParser.float
whitespace = tokenParser.whiteSpace
colon = tokenParser.colon
brackets = tokenParser.brackets
comma = tokenParser.comma
semi = tokenParser.semi
integer = tokenParser.integer
stringLiteral = tokenParser.stringLiteral
reservedOp = tokenParser.reservedOp
stringLit = tokenParser.stringLiteral
```

A.4.7 Novus

```
module Novus(processVantage) where

import Prelude
import Data.Maybe
import Data.Maybe
import Data.Map
import Data.Rational

import Data.Rational
import Data.DateTime.Instant
import Data.Time.Duration

import Data.Tempo

import AST

processVantage:: Map String Vantage -> VantageMap -> DateTime -> Tempo -> VantageMap
processVantage novus vm eval t = difference (unions
[processed,unprocessed,remainFromBuild]) remove
  where unprocessed = difference vm novus -- remain the ones that are not altered
        isBuild (Build _) = true
        isBuild _ = false
        builds = filter (\v -> isBuild v) novus
        builds = filter (\v -> isBuild v) novus
        toBuild = difference builds vm -- List String
        isMove (Move _) = true
        isMove _ = false
        moves = filter (\v -> isMove v) novus
        toMove1 = intersection moves vm
        toMove2 = intersection moves toBuild
        isRemove Remove = true
```

```
        isRemove _ = false
        remove = intersection (filter (\v -> isRemove v) novus) vm
        processed = mapMaybeWithKey (\k v -> transformVtoMaybeDT k v eval t vm) $
unions [toBuild, toMove1, toMove2]

transformVtoMaybeDT:: String -> Vantage -> DateTime -> Tempo -> VantageMap -> Maybe
DateTime
transformVtoMaybeDT _ (Build x) eval t _ = result
  where result = case x of
                Secs secs -> adjust (Seconds $ toNumber secs) eval
                Beat beat -> adjust (Seconds $ toNumber (beat * t.freq)) eval
                UTC dt -> Just dt
transformVtoMaybeDT k (Move x) eval t vm = current >>= adjust (Seconds x')
  where x' = case x of
                Right secs -> toNumber secs
                Left beat -> toNumber (beat * t.freq)
        current = lookup k vm -- Maybe DateTime
transformVtoMaybeDT _ Remove _ _ _ = Nothing
```

A.4.8 Duration Ad Index

```
module
DurationAndIndex(onsetDurations,durFromRhythmic,rhythmicToVoiceDuration,rhythmicToOns
ets, getIndexes, rhythmicStructIndex, getVoiceIndex, getBlocks, durInSecs,
onsetsFromBlocks, bjorklund) where

import Prelude
import Effect (Effect)
import Effect.Console

import Data.Tuple
import Data.Tuple
import Data.Either
import Data.Foldable (sum)
import Data.Int
-- import Data.FunctorWithIndex (mapWithIndex)
import Data.Array (filter,fromFoldable,(!!), zipWith, replicate, concat, (..), (:),
init, tail, last,head,reverse,zip, cons, snoc, length, singleton, splitAt)

import Data.List
import Data.Traversable (scanl)
import Data.List (fromFoldable,concat,zip,zipWith,length,init) as L
import Data.Map (Map(..)) as M

import Control.Applicative

import Data.Newtype

import Data.Tempo
```

```
import AST
import Parser
import Parser

import Data.Rational (Rational(..), (%), fromInt)
import Data.Rational (toNumber) as R
import Data.DateTime
import Data.DateTime.Instant
import Data.Time.Duration

import Data.TraversableWithIndex

-- for testin

import Data.Enum
import Partial.Unsafe

-- get durations

durFromRhythmic:: Rhythmic -> Number -> Number
durFromRhythmic:: Rhythmic -> Number -> Number
durFromRhythmic O tempo = durInSecs 1.0 tempo
durFromRhythmic (Sd rhy) tempo = durInSecs 1.0 tempo
durFromRhythmic (Repeat rhy n) tempo = (durFromRhythmic rhy tempo) * (toNumber n)
durFromRhythmic (Rhythmics xs) tempo = sum $ map (\x -> durFromRhythmic x tempo) xs
durFromRhythmic (Bjorklund eu k n r) tempo = durFromRhythmic (simplifyBjorklund eu k
n r) tempo

rhythmicToVoiceDuration:: Rhythmic -> Number -- does not need Tempo...?
rhythmicToVoiceDuration X = 1.0
rhythmicToVoiceDuration X = 1.0
rhythmicToVoiceDuration (Sd xs) = 1.0
rhythmicToVoiceDuration (Repeat xs n) = foldl (+) 0.0 x
    where x = replicate n $ rhythmicToVoiceDuration xs
rhythmicToVoiceDuration (Bjorklund eu k n r) = rhythmicToVoiceDuration $
simplifyBjorklund eu k n r
rhythmicToVoiceDuration (Rhythmics xs) = foldl (+) 0.0 x
    where x = map (\x -> rhythmicToVoiceDuration x) xs

rhythmicToOnsets:: Rhythmic -> List Onset
rhythmicToOnsets rhy =
    let voiceDur = rhythmicToVoiceDuration rhy
        rhythmicSegments = onsetDurations 1.0 rhy
        durInPercentOfEvents = Cons 0.0 $ (fromMaybe (L.fromFoldable []) $ L.init $
scanl (+) 0.0 $ map (\x -> x/voiceDur) $ getDur <$> rhythmicSegments) -- List Number
    in L.zipWith (\x y -> Onset x y) (getBool <$> rhythmicSegments)
durInPercentOfEvents -- we need to keep the XO -- THIS gives percentage position
within voice,

onsetDurations:: Number -> Rhythmic -> List Onset
onsetDurations dur X =  L.fromFoldable [Onset true dur]
```

```
onsetDurations dur O =  L.fromFoldable [Onset false dur]
onsetDurations dur (Sd xs) = onsetDurations' dur xs
onsetDurations dur (Repeat xs n) = L.concat $ map (\x -> onsetDurations dur x) $
L.fromFoldable $ replicate n xs
onsetDurations dur (Bjorklund eu k n r) = onsetDurations dur (simplifyBjorklund eu k
n r)
onsetDurations dur (Rhythmics xs) = L.concat $ map (\x-> onsetDurations dur x) xs

onsetDurations':: Number -> Rhythmic -> List Onset
onsetDurations' dur X = L.fromFoldable [Onset true dur]
onsetDurations' dur O = L.fromFoldable [Onset false dur]
onsetDurations' dur (Sd xs) = onsetDurations' dur xs
onsetDurations' dur (Repeat xs n) = L.concat $ map (\x -> onsetDurations' newDur x) $
L.fromFoldable $ replicate n xs
    where newDur = dur / (toNumber n)
onsetDurations' dur (Bjorklund eu k n r) = onsetDurations dur (simplifyBjorklund eu k
n r)
onsetDurations' dur (Rhythmics xs) = L.concat $ map (\x-> onsetDurations' newDur x)
xs
    where newDur = dur / (toNumber $ L.length xs)

getDur:: Onset -> Number
getDur (Onset _ x) = x

getBool:: Onset -> Boolean
getBool (Onset x _) = x

----- index calculations -----

getIndexes:: Rhythmic -> Number -> Number -> Number -> Number -> Array Index
getIndexes rhythmic xws we x1 dur =
  let lenOnset = L.length $ rhythmicToOnsets rhythmic
      voiceIndexes = getVoiceIndex xws we x1 dur
      structIndexes = rhythmicStructIndex rhythmic [0]
      eventIndexesPerVoice = (0..(lenOnset-1))
      eventIndexes = funquilla voiceIndexes eventIndexesPerVoice lenOnset -- Array
(Array Int)
  in assambleIndex voiceIndexes structIndexes eventIndexes

assambleIndex:: Array Int -> Array (Array Int) -> Array (Array Int) -> Array Index
assambleIndex vs st es = concat $ zipWith f vs xs
  where xs = map (\e -> zip st e) es
        f:: Int -> Array (Tuple (Array Int) Int) -> Array Index
        f v xs = map (\x -> Index v (fst x) (snd x)) xs

funquilla:: Array Int -> Array Int -> Int -> Array (Array Int)
funquilla voicesIndexes onsetIndexes lenOnsets = map (\voiceIndex -> funquilla'
onsetIndexes lenOnsets voiceIndex) voicesIndexes
  where funquilla' onsetIndexes lenOnsets voiceIndex = map (\onsetIndex ->
(voiceIndex*lenOnsets)+onsetIndex) onsetIndexes
```

```
rhythmicStructIndex:: Rhythmic -> Array Int -> Array (Array Int)
rhythmicStructIndex X i = [i]
rhythmicStructIndex O i = [i]
rhythmicStructIndex (Rhythmics xs) i = concat $ map (\(Tuple x i') ->
rhythmicStructIndex x [i']) zipped
  where zipped = zip (fromFoldable xs) (0..((L.length xs)-1))
rhythmicStructIndex (Repeat rhy n) i = rhythmicStructIndex (simplifyRepeat rhy n) i
rhythmicStructIndex (Sd rhy) i = rhythmicStructIndex' rhy i
rhythmicStructIndex (Bjorklund eu k n rot) i = rhythmicStructIndex (simplifyBjorklund
eu k n rot) i

rhythmicStructIndex':: Rhythmic -> Array Int -> Array (Array Int)
rhythmicStructIndex' X i = [i]
rhythmicStructIndex' X i = [i]
rhythmicStructIndex' (Rhythmics xs) i = concat $ map (\(Tuple x i') ->
rhythmicStructIndex' x (snoc i i')) zipped
  where zipped = zip (fromFoldable xs) (0..((L.length xs)-1))
rhythmicStructIndex' (Sd rhy) i = rhythmicStructIndex' rhy i
rhythmicStructIndex' (Repeat rhy n) i = rhythmicStructIndex' (simplifyRepeat rhy n) i
rhythmicStructIndex' (Bjorklund eu k n rot) i = rhythmicStructIndex'
(simplifyBjorklund eu k n rot) i

-- simplifyRhythmic:: Rhythmic -> Rhythmic
-- simplifyRhythmic (Repeat rhy x) = replicateRhythmic rhy x
-- simplifyRhythmic (Bjorklund eu k n rot) = bjorklundRhythmic eu k n rot
-- simplifyRhythmic rhy = rhy

type STEP a = Tuple (Tuple Int Int) (Tuple (Array (Array a)) (Array (Array a)))

simplifyBjorklund:: Euclidean -> Int -> Int -> Int -> Rhythmic
simplifyBjorklund (Simple) k n rot = Rhythmics xs
  where xs = L.fromFoldable $ map (\r -> if r == true then X else O) $ blRotated rot
$ bjorklund (Tuple k n)
simplifyBjorklund (K patt) k n rot = Rhythmics xs
  where xs = L.fromFoldable $ map (\r -> if r == true then patt else patto) $
blRotated rot $ bjorklund (Tuple k n)
        patto = Rhythmics $ L.fromFoldable $ replicate (L.length $ rhythmicToOnsets
patt) O
simplifyBjorklund (InvK patt) k n rot = Rhythmics xs
  where xs = L.fromFoldable $ map (\r -> if r == true then patto else patt) $
blRotated rot $ bjorklund (Tuple k n)
        patto = Rhythmics $ L.fromFoldable $ replicate (L.length $ rhythmicToOnsets
patt) O
simplifyBjorklund (Full pK pN) k n rot = Rhythmics xs
  where xs = L.fromFoldable $ map (\r -> if r == true then pK else pN) $ blRotated
rot $ bjorklund (Tuple k n)

blRotated:: Int -> Array Boolean -> Array Boolean
blRotated rot patt = x.after <> x.before
  where x = splitAt rot patt

bjorklund:: Tuple Int Int -> Array Boolean
```

```
bjorklund (Tuple i j') = (concat x') <> (concat y')
  where j = j' - i
        x = replicate i [true]
        y = replicate j [false]
        (Tuple _ (Tuple x' y')) = bjorklund' (Tuple (Tuple i j) (Tuple x y))

bjorklund':: forall a. STEP a -> STEP a
bjorklund' (Tuple n x) =
    let (Tuple i j) = n
    in if min i j <= 1
       then Tuple n x
       else bjorklund' (if i > j then left (Tuple n x) else right (Tuple n x))

right:: forall a. STEP a -> STEP a
right (Tuple (Tuple i j) (Tuple xs ys)) = Tuple (Tuple i (j-i)) (Tuple (zipWith (<>)
xs ys') ys'')
  where splitted = splitAt i ys
        ys' = splitted.before
        ys' = splitted.before

left:: forall a. STEP a -> STEP a
left (Tuple (Tuple i j) (Tuple xs ys)) = Tuple (Tuple j (i-j)) (Tuple (zipWith (<>)
xs' ys) xs'')
    where splitted = splitAt j xs
          xs' = splitted.before
          xs'' = splitted.after

-- the output is always Rhythmics constructor
simplifyRepeat:: Rhythmic -> Int -> Rhythmic
simplifyRepeat (X) n = Rhythmics $ L.fromFoldable $ replicate n X
simplifyRepeat (X) n = Rhythmics $ L.fromFoldable $ replicate n X
simplifyRepeat (Sd rhy) n = Rhythmics $ map (\r -> Sd r) $ L.fromFoldable $ replicate
n rhy
simplifyRepeat (Repeat rhy n2) n1 = simplifyRepeat rhy n
  where n = round ((toNumber n1) * (toNumber n2))
simplifyRepeat (Bjorklund eu k n' rot) n = simplifyRepeat (simplifyBjorklund eu k n'
rot) n
simplifyRepeat (Rhythmics xs) n = Rhythmics $ L.fromFoldable $ concat $ replicate n $
fromFoldable xs


getVoiceIndex:: Number -> Number -> Number -> Number -> Array Int -- Index for Voice
getVoiceIndex xws we x1 dur =
  let nOfFstBlock = nFirstBlock xws x1 dur  -- :: Int
      nOfLstBlock = nLastBlock we x1 dur  -- Maybe Int
      nOfBlocks = case nOfLstBlock of
                    Nothing -> []
                    (Just n) -> (nOfFstBlock..n) -- [Int]
  in nOfBlocks

-- start times of blocks in expanded window, in seconds since origin
getBlocks:: Number -> Number -> Number -> Number -> Array Number
```

```
getBlocks xws we x1 dur =
  let nOfFstBlock = nFirstBlock xws x1 dur  -- :: Int
      nOfLstBlock = nLastBlock we x1 dur  -- Maybe Int
      nsOfBlocks = case nOfLstBlock of
                    Nothing -> [] --(nOfFstBlock..(nOfFstBlock + 1))
                    (Just n) -> (nOfFstBlock..n) -- [Int]
  in (blockToPos nsOfBlocks x1 dur)

blockToPos:: Array Int -> Number -> Number -> Array Number
blockToPos is x1 dur = map (\i -> x1 + ((toNumber i) * dur)) is

-- n of first block at or after ws, regardless of how far in the future
nFirstBlock:: Number -> Number -> Number -> Int
nFirstBlock _ _ 0.0 = 0
nFirstBlock ws x1 dur = nOfNxBlock
  where xwsB = (ws - x1)/dur -- number of block elapsed at xws
        nOfNxBlock
          | xwsB <= 0.0 = 0
          | otherwise = ceil xwsB

nLastBlock:: Number -> Number -> Number -> Maybe Int
nLastBlock we x1 dur = nOfLastBlock
  where wEndBlocks = (we - x1)/dur -- number of blocks elapsed at we
        nOfLastBlock
          | wEndBlocks < 0.0 = Nothing
          | otherwise = Just $ floor wEndBlocks

-- posix needs to be removed
onsetsFromBlocks:: Array Number -> Array Onset -> Number -> Array Onset
onsetsFromBlocks blocks onsets dur = concat $ map (\block -> onsetsFromBlock onsets
block dur) blocks

onsetsFromBlock:: Array Onset -> Number -> Number -> Array Onset
onsetsFromBlock onsets block dur = map (\(Onset bool pos) -> Onset bool (block +
(pos*dur))) onsets

-----
durInSecs:: Number -> Number -> Number
durInSecs dur bpm = dur * (bpmToDur bpm)

bpmToFreq bpm = bpm / 60.0 -- bpmToCPS

freqToDur freq = 1.0 / freq

bpmToDur bpm = 1.0 / bpmToFreq bpm

countInFreqToSecs:: Rational -> Rational -> Rational
countInFreqToSecs freq x = x / freq

toRat:: Number -> Rational
toRat x =
    let pFact = 1000000
```

```
        floored = floor x -- 12
        fract = x - (toNumber floored) -- 12.5 - 12.0 = 0.5
        fract = x - (toNumber floored) -- 12.5 - 12.0 = 0.5
    in (floored%1) + (fract'%pFact) -- 12 + (500000%1000000)
```

A.4.9 Temporal Specifications

```
module TemporalSpecs (calculateTemporal) where

import Preludeimport Effect (Effect)
import Effect.Console
import Data.Tuple
import Data.Tuple
import Data.Either
import Data.Map as M
import Data.Foldable (sum)
import Data.Int
import Data.Array (filter,fromFoldable,(!!), zipWith, replicate, concat, (..), (:),
init, tail, last,head,reverse,zip, cons, snoc, length, singleton, splitAt)
import Data.List (List(..))

import Data.Tempo

import AST
import Acceleration
import TestOpsAndDefs
import DurationAndIndex
import TimePacketOps

import Data.Rational (Rational(..), (%), fromInt)
import Data.Rational (toNumber) as R -- still need to convert all Number calcs into
Rational!!


calculateTemporal:: M.Map String Temporal -> TimePacket -> String -> Temporal ->
Effect (Array Event)
calculateTemporal mapa tp aKey (Replica id) = do
  let replicatedTemporal = fromMaybe defTemporal $ M.lookup id mapa
  result <- calculateTemporal mapa tp aKey replicatedTemporal
  pure result

calculateTemporal m tp aKey (Temporal (Kairos asap tm) rhythmic loop) = do
  let dur = establishDur tm tp.tempo m rhythmic
      -- tempo = processTempoMark tempoMark tp.tempo mapa
      posixAtOrigin = fromDateTimeToPosix (origin tp.tempo)
      eval = secsFromOriginAtEval tp
      ws = secsFromOriginAtWS tp
      ws = secsFromOriginAtWS tp
      -- dur = durFromRhythmic rhythmic tempo -- number
      -- dur = durInSecs (sum $ rhythmicToLinDur rhythmic) tempo -- acc experiment
```

```
      x1 = eval + asap -- always the start of the program
      blocks = getBlocks (ws - dur) we x1 dur -- Array Number
      -- onsets = onsetsFromBlocks blocks (fromFoldable $ rhythmicToOnsetsAcc
rhythmic) dur -- Array Onset --- absolute position
      onsets = onsetsFromBlocks blocks (fromFoldable $ rhythmicToOnsets' tm tp.tempo
m rhythmic) dur -- Array Onset --- absolute position
      indexes = getIndexes rhythmic (ws - dur) we x1 dur -- Array Index
      events = zipWith Event onsets indexes
      posFromEvent (Event (Onset _ p) _) = p
      looped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) events
      unlooped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) <= we) $ unloopEvents events
  pure if loop then looped else unlooped

calculateTemporal mapa tp aKey (Temporal (Metric cTo' cFrom' tm) rhythmic loop) = do
  let dur = establishDur tm tp.tempo mapa rhythmic
  -- let dur = durFromRhythmic rhythmic $ processTempoMark tm tp.tempo mapa --
correct (change tempo naming to other name)
  -- log ("durCalcTempoMetricTemporal " <> show dur)
  let lengthRhythm = (length $ fromFoldable $ rhythmicToOnsets rhythmic)-1
  let simCTo = simplifyCTo lengthRhythm cTo'
  let simCFrom = simplifyCFrom lengthRhythm cFrom'
  x1 <- x1MetricVoice tp tm simCTo simCFrom rhythmic mapa
  -- log ("x1 IN metricTemporal " <> show x1)
  let posixAtOrigin = fromDateTimeToPosix (origin tp.tempo)
  -- let eval = secsFromOriginAtEval tp
  let ws = secsFromOriginAtWS tp
  let ws = secsFromOriginAtWS tp
  let blocks = getBlocks (ws - dur) we x1 dur -- to check
  -- log ("blocksMetricTemporal: " <> show blocks)
  let onsetPercent = fromFoldable $ rhythmicToOnsets' tm tp.tempo mapa rhythmic --
Array Onsets --- Position in Percentage
  let onsets = onsetsFromBlocks blocks onsetPercent dur -- Array Onset --- absolute
position
  let indexes = getIndexes rhythmic (ws - dur) we x1 dur -- Array Index
  let events = zipWith Event onsets indexes
  let posFromEvent:: Event -> Number
      posFromEvent (Event (Onset _ p) _) = p
  let looped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) events
  let unlooped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) $ unloopEvents events
  pure $ if loop then looped else unlooped
                                  -- v2            --v1
calculateTemporal mapa tp aKey (Temporal (Converge cKey cTo' cFrom' tm) rhythmic
loop) = do
  -- let dur = durFromRhythmic rhythmic $ processTempoMark tm tp.tempo mapa
  let dur = establishDur tm tp.tempo mapa rhythmic
  let lengthRhythm = (length $ fromFoldable $ rhythmicToOnsets rhythmic)-1
  let simCTo = simplifyCTo lengthRhythm cTo'
  let simCFrom = simplifyCFrom lengthRhythm cFrom'
```

```
  x1 <- x1ConvergeVoice tp tm cKey simCTo simCFrom rhythmic mapa -- v1
  let posixAtOrigin = fromDateTimeToPosix (origin tp.tempo)
  let ws = secsFromOriginAtWS tp
  let ws = secsFromOriginAtWS tp
  let blocks = getBlocks (ws - dur) we x1 dur -- to check
  let onsetPercent = fromFoldable $ rhythmicToOnsets' tm tp.tempo mapa rhythmic --
[Onsets] Pos in Percentage
  let onsets = onsetsFromBlocks blocks onsetPercent dur --[Onsets] absolute position
  let indexes = getIndexes rhythmic (ws - dur) we x1 dur -- Array Index
  let events = zipWith Event onsets indexes
  let posFromEvent:: Event -> Number
      posFromEvent (Event (Onset _ p) _) = p
  let looped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) events
  let unlooped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) $ unloopEvents events
  pure $ if loop then looped else unlooped
----- CALCULATE NOVUS!!!!!!!!!!!!!
calculateTemporal mapa tp aKey (Temporal (Novus vKey cFrom' tm) rhythmic loop) = do
  let dur = establishDur tm tp.tempo mapa rhythmic
  -- let dur = durFromRhythmic rhythmic $ processTempoMark tm tp.tempo mapa
  let lengthRhythm = (length $ fromFoldable $ rhythmicToOnsets' tm tp.tempo mapa
rhythmic)-1
  let simCFrom = simplifyCFrom lengthRhythm cFrom'
  let cp = secsFromOriginAtVantage tp vKey
  -- log ("cp novus: " <> show cp)
  x1 <- x1NovusVoice tp tm cp simCFrom rhythmic mapa -- v1

  let posixAtOrigin = fromDateTimeToPosix (origin tp.tempo)
  let ws = secsFromOriginAtWS tp
  let ws = secsFromOriginAtWS tp
  let blocks = getBlocks (ws - dur) we x1 dur -- to check
  let onsetPercent = fromFoldable $ rhythmicToOnsets' tm tp.tempo mapa rhythmic
  let onsets = onsetsFromBlocks blocks onsetPercent dur --[Onsets] absolute position
  let indexes = getIndexes rhythmic (ws - dur) we x1 dur -- Array Index
  let events = zipWith Event onsets indexes
  let posFromEvent:: Event -> Number
      posFromEvent (Event (Onset _ p) _) = p
  let looped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) events
  let unlooped = addPosixOriginToCalculation posixAtOrigin $ filter (\e ->
(posFromEvent e) >= ws && (posFromEvent e) < we) $ unloopEvents events
  pure $ if loop then looped else unlooped

x1NovusVoice:: TimePacket -> TempoMark -> Number -> ConvergeFrom -> Rhythmic -> M.Map
String Temporal -> Effect Number
x1NovusVoice tp tm cp cFrom' rhythmic mapa = do
  -- let tempo = processTempoMark tm tp.tempo mapa
  let cFrom = calculateCFrom cFrom' rhythmic
  let dur = establishDur tm tp.tempo mapa rhythmic
  -- let dur = durFromRhythmic rhythmic tempo
  let x1 = cp - (cFrom * dur)
```

```
  pure x1

unloopEvents:: Array Event -> Array Event
unloopEvents es = filter (\(Event _ (Index b _ _)) -> b == 0) es

addPosixOriginToCalculation:: Number -> Array Event -> Array Event
addPosixOriginToCalculation posix es = map (\(Event (Onset bool pos) i) -> Event
(Onset bool (pos + posix)) i) es

simplifyCTo:: Int -> ConvergeTo -> ConvergeTo
simplifyCTo n (LastTo a) = ProcessTo n a
simplifyCTo n cTo = cTo

simplifyCFrom:: Int -> ConvergeFrom -> ConvergeFrom
simplifyCFrom n Last = Process n
simplifyCFrom n cfrom = cfrom

-- find x1 and dur of referenceVoice for convergent temporal
x1ConvergeVoice:: TimePacket -> TempoMark -> String -> ConvergeTo -> ConvergeFrom ->
Rhythmic -> M.Map String Temporal -> Effect Number
x1ConvergeVoice  tp tm cKey cTo' cFrom' rhythmic mapa = do
  let refTemporal = fromMaybe defTemporal $ M.lookup cKey mapa
  let refRhythmic = getRhythmic mapa refTemporal
  -- let refTempo = processTempoMark (tempoMark mapa refTemporal) tp.tempo mapa --
::Number -- cpm
  let refDur = establishDur (tempoMark mapa refTemporal) tp.tempo mapa refRhythmic
  -- let refDur = durFromRhythmic refRhythmic refTempo
  -- let refDur = (\(Temporal p rhy _) -> durFromRhythmic rhy $ processTempoMark
(getTempoMark p) tp.tempo mapa) refTemporal
  refX1 <- findReferencedX1 tp refTemporal mapa
  refVoiceAtEval <- elapsedVoiceAtEval tp refX1 refDur -- not secs but cycles
  -- log ("refVoiceAtEval top " <> show refVoiceAtEval)
  let innerPos = innerPosCTo refRhythmic cTo'
  let cTo = calculateCToNEW innerPos refVoiceAtEval cTo'
  -- let processedTempoMark = processTempoMark tm tp.tempo mapa
  let cFrom = calculateCFrom cFrom' rhythmic
  let dur = establishDur tm tp.tempo mapa rhythmic
  -- let dur = durFromRhythmic rhythmic processedTempoMark
  let x1 = calculateStartConvergent refDur cTo dur cFrom  -- result in secs
  -- log ("x1 converge voice top " <> show (refX1 + x1))
  -- cuando empieza la voz en secs, cuanto dura cada bloque en secs, donde esta la
voz en eval
  pure (refX1 + x1)

findReferencedX1::TimePacket -> Temporal -> M.Map String Temporal -> Effect Number
findReferencedX1 tp (Replica id) mapa = do
  let replicatedTemporal = fromMaybe defTemporal $ M.lookup id mapa
  result <- findReferencedX1 tp replicatedTemporal mapa
  pure result
findReferencedX1 tp (Temporal (Kairos asap tm) rhy _) mapa = do
  let eval = secsFromOriginAtEval tp
  let x1 = eval + asap
```

```
  -- log ("x1 kairos voice " <> show x1)
  pure x1
findReferencedX1 tp (Temporal (Metric cTo cFrom tm) rhy _) mapa = do
  x1 <- x1MetricVoice tp tm cTo cFrom rhy mapa
  -- log ("x1 metric voice " <> show x1)
  pure x1            -- v1            --v0
findReferencedX1 tp (Temporal (Converge cKey cTo cFrom tm) rhy l) mapa = do
  let way = keysForReferencePath cKey mapa (Nil) -- Array String
  -- log ("key " <> show cKey)
  -- log ("way: " <> show way)      -- v1               --v0
  recursiveX1 <- recursiveRefX1 tp (Temporal (Converge cKey cTo cFrom tm) rhy l) mapa
Nothing way -- v1's x1
  -- log ("x1 converge voice" <> show recursiveX1)
  pure recursiveX1
  ---- calculate NOVUS!!!!!!!!!!!!!!!!!!!!!!
findReferencedX1 tp (Temporal (Novus vKey cFrom tm) rhy l) mapa = pure 0.0


recursiveRefX1:: TimePacket -> Temporal -> M.Map String Temporal -> Maybe (Tuple
String Number) -> List String -> Effect Number      -- v2
-- incoming with [] is v1
recursiveRefX1 tp temporal mapa incomingKeyX1' (Nil) = do
  let cKey = getKey mapa temporal
  let (Tuple cTo' cFrom') = convergences mapa temporal
  -- let (Tuple cTo' cFrom') = (\(Temporal p _ _) -> getConvergences p) temporal
  let rhythmic = getRhythmic mapa temporal

--- HERE WORK ON implementing establishDur function for acceleration and function

  -- let processedTM = processTempoMark (tempoMark mapa temporal) tp.tempo mapa
  -- let processedTM = (\(Temporal p _ _) -> processTempoMark (getTempoMark p)
tp.tempo mapa) temporal
  -- let dur = durFromRhythmic rhythmic processedTM
  let dur = establishDur (tempoMark mapa temporal) tp.tempo mapa rhythmic
  let cFrom = calculateCFrom cFrom' rhythmic

  let temporalHack = fromMaybe defTemporal $ M.lookup cKey mapa
  incomingKeyX1 <- if incomingKeyX1' == Nothing then
           Just <$> (Tuple cKey <$> (findReferencedX1 tp temporalHack mapa)) else
pure incomingKeyX1'

  -- log ("incomingKeyX1 " <> show incomingKeyX1)

  let (Tuple refKey refX1) = fromMaybe (Tuple "error" 2.666) incomingKeyX1
  -- log ("refX1 " <> show refX1)
  let refTemporal = fromMaybe defTemporal $ M.lookup refKey mapa

  let refRhythmic = getRhythmic mapa refTemporal
  -- let refTM = processTempoMark (tempoMark mapa refTemporal) tp.tempo mapa
  -- let refTM = (\(Temporal p _ _) -> processTempoMark (getTempoMark p) tp.tempo
mapa) refTemporal
  -- let refDur = durFromRhythmic refRhythmic refTM
```

```
  let refDur = establishDur (tempoMark mapa refTemporal) tp.tempo mapa refRhythmic
  refVoiceAtEval <- elapsedVoiceAtEval tp refX1 refDur -- not secs but cycles
  let innerPos = innerPosCTo refRhythmic cTo'
  let cTo = calculateCToNEW innerPos refVoiceAtEval cTo'

  let x1 = calculateStartConvergent refDur cTo dur cFrom
  let x1 = calculateStartConvergent refDur cTo dur cFrom
  pure (refX1 + x1)

recursiveRefX1 tp temporal mapa incomingKeyX1 (Cons x xs) = do
  let refTemporal = fromMaybe defTemporal $ M.lookup x mapa  -- v0
  let (Tuple cTo' cFrom') = convergences mapa refTemporal
  let refRhythmic = getRhythmic mapa refTemporal
  -- let refProcessedTM = processTempoMark (tempoMark mapa refTemporal) tp.tempo mapa
  -- let refRhythmic = (\(Temporal _ r _) -> r) refTemporal
  -- let refProcessedTM = (\(Temporal p _ _) -> processTempoMark (getTempoMark p)
tp.tempo mapa) refTemporal
  let refDur = establishDur (tempoMark mapa refTemporal) tp.tempo mapa refRhythmic
  -- let refDur = durFromRhythmic refRhythmic refProcessedTM
  let cFrom = calculateCFrom cFrom' refRhythmic
  refX1 <- case incomingKeyX1 of
          Nothing -> findReferencedX1 tp refTemporal mapa -- result: v0's block1's
start point (x1)
          Just prevKeyX1 -> do
                let prevTemporal = fromMaybe defTemporal $ M.lookup (fst prevKeyX1)
mapa
                let prevRhythmic = getRhythmic mapa prevTemporal
                -- let prevTM = processTempoMark (tempoMark mapa prevTemporal)
tp.tempo mapa
                -- let prevDur = durFromRhythmic prevRhythmic prevTM
                let prevDur = establishDur (tempoMark mapa prevTemporal) tp.tempo
mapa prevRhythmic
                prevVoiceAtEval <- elapsedVoiceAtEval tp (snd prevKeyX1) prevDur --
not secs but cycles
                let innerPos = innerPosCTo prevRhythmic cTo'
                let cTo = calculateCToNEW innerPos prevVoiceAtEval cTo'
                let refX1 = calculateStartConvergent prevDur cTo refDur cFrom
                pure ((snd prevKeyX1) + refX1)
  result <- recursiveRefX1 tp temporal mapa (Just (Tuple x refX1)) xs
  -- log ("result recursiveRef " <> show result)
  pure result

convergences:: M.Map String Temporal -> Temporal -> Tuple ConvergeTo ConvergeFrom
convergences _ (Temporal p _ _) = getConvergences p
convergences _ (Temporal p _ _) = getConvergences p
                                Nothing -> Tuple defConvergeTo defConvergeFrom
                                Just t -> convergences m t

getConvergences:: Polytemporal -> Tuple ConvergeTo ConvergeFrom
getConvergences (Converge _ cTo cFrom _) = Tuple cTo cFrom
getConvergences _ = Tuple defConvergeTo defConvergeFrom
```

```
keysForReferencePath:: String -> M.Map String Temporal -> List String -> List String
keysForReferencePath aKey {-v0-} mapa listOfReferences
  | isNotConvergent aKey mapa = (Nil)
  | otherwise =  -- v0
      if (isNotConvergent nextCheck mapa)
        then (Cons nextCheck listOfReferences)
          else keysForReferencePath nextCheck mapa (Cons nextCheck listOfReferences)
    where nextCheck = getKey mapa $ fromMaybe defTemporal $ M.lookup aKey mapa

getKey _ (Temporal (Converge aKey _ _ _) _ _) = aKey
getKey _ (Temporal _ _ _) = "2666"
getKey m (Replica id) =
getKey m (Replica id) =
    Nothing -> "2666"
    Just t -> getKey m t

isNotConvergent aKey mapa = f' mapa $ fromMaybe defTemporal $ M.lookup aKey mapa

f' m (Temporal (Converge _ _ _ _) _ _) = false
f' m (Temporal _ _ _) = true
f' m (Replica id) = case M.lookup id m of
f' m (Replica id) = case M.lookup id m of
                       Just t -> f' m t

elapsedVoiceAtEval:: TimePacket -> Number -> Number -> Effect Number
elapsedVoiceAtEval tp x1 dur = do
  let eval = secsFromOriginAtEval tp
      atEval = (eval - x1) / dur
  pure atEval

---- finding x1 for Metric
x1MetricVoice:: TimePacket -> TempoMark -> ConvergeTo -> ConvergeFrom -> Rhythmic ->
M.Map String Temporal -> Effect Number
x1MetricVoice tp tm cTo' cFrom' rhythmic mapa = do
x1MetricVoice tp tm cTo' cFrom' rhythmic mapa = do
  -- log ("tempo-X1Metric " <> show tempo)
  let eval = secsFromOriginAtEval tp
  let externalVoiceSecs = 1.0 / (R.toNumber tp.tempo.freq)
  -- log ("externalVoiceSecs-X1Metric " <> show externalVoiceSecs)
  let cyclesAtEval = R.toNumber $ timeToCount tp.tempo tp.eval
  -- log ("cyclesAtEva-X1Metric " <> show cyclesAtEval)
  let cTo = calculateCToMetric cyclesAtEval cTo' -- cycles of compared voice
  -- log ("cTo-X1Metric " <> show cTo)
  let cFrom = calculateCFrom cFrom' rhythmic -- ignore for now, test with 0
  let dur = establishDur tm tp.tempo mapa rhythmic
  -- let dur = durFromRhythmic rhythmic tempo -- correct (change tempo naming to
other name)
  -- log ("dur-X1Metric " <> show dur)
  let x1 = calculateStartConvergent externalVoiceSecs cTo dur cFrom  -- result in
secs
  -- log ("x1 (result of X1 Metric)" <> show x1)
  pure x1
```

```
----- this funca!! <3  <3  <3
calculateStartConvergent:: Number -> Number -> Number -> Number -> Number
calculateStartConvergent durConverged convergeTo durVoice convergeFrom =
startOfVoiceInSecs
  where cTo = convergeTo * durConverged
  where cTo = convergeTo * durConverged
        startOfVoiceInSecs = cTo - cFrom

-- calculating convergence points
calculateCToMetric:: Number -> ConvergeTo -> Number
calculateCToMetric cyclesAtEval (StructureTo b st a) = (toNumber b) + aligned
  where aligned = aligner cyclesAtEval a
calculateCToMetric cyclesAtEval (ProcessTo i a) = (toNumber i) + aligned
  where aligned = aligner cyclesAtEval a
calculateCToMetric cyclesAtEval (PercenTo p a) =  (p / 100.0) + aligned
  where aligned = aligner cyclesAtEval a
calculateCToMetric cyclesAtEval _ = 0.0

calculateCToNEW:: Number -> Number -> ConvergeTo -> Number
calculateCToNEW innerPos cyclesAtEval (StructureTo b st a) = innerPos + aligned
  where aligned = aligner cyclesAtEval a
calculateCToNEW innerPos cyclesAtEval (ProcessTo i a) = innerPos + aligned
  where aligned = aligner cyclesAtEval a
calculateCToNEW innerPos cyclesAtEval (PercenTo p a) =  (p / 100.0) + aligned
  where aligned = aligner cyclesAtEval a
calculateCToNEW innerPos cyclesAtEval _ = 0.0

aligner:: Number -> CPAlign -> Number -- in cycles of external metre
aligner cyclesAtEval Origin = 0.0
aligner cyclesAtEval Snap = (toNumber $ ceil cyclesAtEval)
aligner cyclesAtEval (Mod m) = ceiledModInMetre * (toNumber m)
  where modInMetre = cyclesAtEval / (toNumber m)
        ceiledModInMetre = toNumber $ ceil modInMetre

innerPosCTo:: Rhythmic -> ConvergeTo -> Number
innerPosCTo rhythmic cTo = percentPos
  where onsetPercent = fromFoldable $ rhythmicToOnsets rhythmic -- Array Onsets ---
Position in Percentage
        lenOnset = length onsetPercent
        structIndexes = rhythmicStructIndex rhythmic [0] -- Array (Array Int)
        eventIndexesPerVoice = (0..(lenOnset-1)) -- Array Int
        structAndPos = zip structIndexes $ map (\(Onset b p) -> p) onsetPercent
        eventsAndPos = zip eventIndexesPerVoice $ map (\(Onset b p) -> p)
onsetPercent
        percentPos = filterEventToPosTo cTo structAndPos eventsAndPos lenOnset

calculateCFrom:: ConvergeFrom -> Rhythmic -> Number
calculateCFrom cp rhythmic = percentPos
  where onsetPercent = fromFoldable $ rhythmicToOnsets rhythmic -- Array Onsets ---
Position in Percentage
        lenOnset = length onsetPercent
```

```
        structIndexes = rhythmicStructIndex rhythmic [0] -- Array (Array Int)
        eventIndexesPerVoice = (0..(lenOnset-1)) -- Array Int
        structAndPos = zip structIndexes $ map (\(Onset b p) -> p) onsetPercent
        eventsAndPos = zip eventIndexesPerVoice $ map (\(Onset b p) -> p)
onsetPercent
        percentPos = filterEventToPosFrom cp structAndPos eventsAndPos lenOnset

filterEventToPosTo:: ConvergeTo -> Array (Tuple (Array Int) Number) -> Array (Tuple
Int Number) -> Int -> Number
filterEventToPosTo cp structAndPos eventsAndPos lenOnset = result
  where result = case cp of
                    (StructureTo v st a) -> fromMaybe 0.0 $ head $ map (\x -> cpPos
(Left v) (snd x) lenOnset) $ filter (\x -> fst x == st) structAndPos
                    (ProcessTo e a) -> fromMaybe 0.0 $ head $ map (\x -> cpPos (Right
e) (snd x) lenOnset) $ filter (\x -> fst x == (e`mod`lenOnset)) eventsAndPos
                    (PercenTo p a) ->  p / 100.0
                    _ -> 0.0

filterEventToPosFrom:: ConvergeFrom -> Array (Tuple (Array Int) Number) -> Array
(Tuple Int Number) -> Int -> Number
filterEventToPosFrom cp structAndPos eventsAndPos lenOnset = result
  where result = case cp of
                    (Structure v st) -> fromMaybe 0.0 $ head $ map (\x -> cpPos (Left
v) (snd x) lenOnset) $ filter (\x -> fst x == st) structAndPos
                    (Process e) -> fromMaybe 0.0 $ head $ map (\x -> cpPos (Right e)
(snd x) lenOnset) $ filter (\x -> fst x == (e`mod`lenOnset)) eventsAndPos
                    (Percen p) ->  p / 100.0
                    _ -> 0.0

cpPos:: Either Int Int -> Number -> Int -> Number
cpPos (Left v) x lenOnset = v' + x
  where v' = (toNumber v)
cpPos (Right n) x lenOnset = (toNumber $ floor n') + x
  where n' = (toNumber n)/(toNumber lenOnset)

-- dur
establishDur:: TempoMark -> Tempo -> M.Map String Temporal -> Rhythmic -> Number
establishDur (Dur n) xT m rhy = R.toNumber n
establishDur (Sin sin) xT m rhy = durInSecs (sum $ rhythmicToSinDur rhy (R.toNumber
sin.osc) min max (R.toNumber sin.phase)) min
  where min = processTempoMark sin.min xT m
  where min = processTempoMark sin.min xT m
establishDur (Prop id x y) xT m rhy = durProp m xT otherTempoMark otherRhy prop
  where prop = (toNumber x / toNumber y)
        otherTemporal = fromMaybe defTemporal $ M.lookup id m
        otherTempoMark = tempoMark m otherTemporal
        otherRhy = getRhythmic m otherTemporal
establishDur tm xT m rhy = durFromRhythmic rhy $ processTempoMark tm xT m

durProp:: M.Map String Temporal -> Tempo -> TempoMark -> Rhythmic -> Number -> Number
durProp m xT (Sin sin) r prop = durOther / prop
  where min = processTempoMark sin.min xT m
```

```
  where min = processTempoMark sin.min xT m
        durOther = durInSecs (sum $ rhythmicToSinDur r (R.toNumber sin.osc) min max
(R.toNumber sin.phase)) min

durProp m xT (Dur d) r prop = (R.toNumber d) / prop
durProp m xT (Prop id x y) _ prop = durProp m xT otherTM otherRhy otherProp
  where otherProp = (toNumber x / toNumber y) * prop
        otherTemporal = fromMaybe defTemporal $ M.lookup id m
        otherTM = tempoMark m otherTemporal
        otherRhy = getRhythmic m otherTemporal
durProp m xT another r prop = (durFromRhythmic r $ processTempoMark another xT m) /
prop

rhythmicToOnsets':: TempoMark -> Tempo -> M.Map String Temporal -> Rhythmic -> List
Onset
rhythmicToOnsets' (Sin s) xT m rhy = rhythmicToOnsetsSin rhy (R.toNumber s.osc) min
max (R.toNumber s.phase)
  where min = processTempoMark s.min xT m
  where min = processTempoMark s.min xT m
rhythmicToOnsets' tm xT m rhy = case tm of
                              (Prop id x y) -> rhythmicToOnsets' otherTM xT m rhy
                                where otherTemporal = fromMaybe defTemporal $
M.lookup id m
                                      otherTM = tempoMark m otherTemporal
                                      otherRhy = getRhythmic m otherTemporal
                              _ -> rhythmicToOnsets rhy

processTempoMark:: TempoMark -> Tempo -> M.Map String Temporal -> Number
processTempoMark (CPM cpm) _ _ = R.toNumber (cpm / (4%1))
processTempoMark (BPM bpm figure) _ _ = R.toNumber ((bpm / (4%1)) / figure)
processTempoMark (CPS cps) _ _ = R.toNumber (cps * (60%1))
processTempoMark XTempo t _ = (R.toNumber (t.freq * (60%1) * (4%1)))
processTempoMark (Prop id x y) t mapa = fromMaybe 120.0 otherTempo
  where prop = (toNumber x / toNumber y)
        otherTempo = (\temporal -> calculateRTempo mapa t (tempoMark mapa temporal)
prop) <$> M.lookup id mapa
processTempoMark other t mapa = 0.0

calculateRTempo:: M.Map String Temporal -> Tempo -> TempoMark -> Number -> Number
calculateRTempo m t (CPM cpm) prop = (R.toNumber (cpm / (4%1))) * prop
calculateRTempo m t (BPM bpm figure) prop = (R.toNumber ((bpm / (4%1)) / figure)) *
prop
calculateRTempo m t (CPS cps) prop = R.toNumber (cps * (60%1)) * prop
calculateRTempo m t XTempo prop = (R.toNumber (t.freq * (60%1) * (4%1))) * prop
calculateRTempo m t (Prop id x y) prop = calculateRTempo m t newTM newProp
  where newProp = (toNumber x / toNumber y) * prop
        newTM = fromMaybe (CPM (fromInt 120)) $ (\temporal -> tempoMark m temporal)
<$> M.lookup id m
calculateRTempo m t other prop = 0.0
```

A.4.10 Acceleration

```
module Acceleration (rhythmicToSinDur,sinusoidalAcceleration,rhythmicToOnsetsSin)
where

import Prelude
import Effect (Effect)
import Effect.Console (log)

import Data.Int (toNumber)
import Data.Number
import Data.Array
import Data.Array (fromFoldable)
import Data.List (List(..))
import Data.List (fromFoldable, (:)) as L
import Data.Foldable (sum)
import Data.Tuple
import Data.Maybe (fromMaybe)

import AST
import DurationAndIndex

rhythmicToOnsetsSin:: Rhythmic -> Number -> Number -> Number -> Number -> List Onset
rhythmicToOnsetsSin rhy osc min max ph = L.fromFoldable $ zipWith Onset onsets pos
    where onsets = map (\(Onset b p) -> b) $ fromFoldable $ rhythmicToOnsets rhy
          rhyDur = rhythmicToSinDur rhy osc min max ph
          folded' = fromMaybe {init: [], last: 2.666} $ unsnoc $ (scanl (+) 0.0
rhyDur)
          pos = map (\fo -> fo / (sum rhyDur)) $ (0.0 : folded'.init)

acceleration :: Number -> Number -> Number -> Number -> Number -> Number
acceleration startTime finalTime startSpeed endSpeed currentTime =
  let
    deltaTime = finalTime - startTime
    deltaSpeed = endSpeed - startSpeed
    acceleration = deltaSpeed / deltaTime
    initialSpeed = startSpeed
  in
    initialSpeed + acceleration * currentTime

rhythmicToLinDur:: Rhythmic -> Array Number
rhythmicToLinDur rhythmic = map (\(Tuple dur acc) -> (dur / (0.5 + acc))) zipped
    where onsetDur = map (\(Onset b p) -> p) $ fromFoldable $ onsetDurations 1.0
rhythmic
          onsetPos = map (\(Onset b p) -> p) $ rhythmicToOnsets rhythmic
          onsetAcc = fromFoldable $ map (\pos -> acceleration 0.0 1.0 1.0 2.0 pos)
onsetPos
          zipped = zip onsetDur onsetAcc

sinusoidalAcceleration :: Number -> Number -> Number -> Number -> Number
sinusoidalAcceleration frequency currentTime amplitude phase =
```

```
  amplitude * sin (2.0 * pi * frequency * currentTime + phase)
  -- amplitude * sin (2.0 * pi * frequency * currentTime + phase)

-- freq is actually cycles per block. 1 means one whole oscilation per block
-- amplitud will determine the range. If amplitude 1 it will go from 0 to 1 and -1

rhythmicToSinDur:: Rhythmic -> Number -> Number -> Number -> Number -> Array Number
rhythmicToSinDur rhythmic freq min' max' ph = map (\(Tuple dur acc) -> dur / (min +
((amp) + acc))) zipped
    where min = 1.0
          max = max' / min'
          amp = (max - min) / 2.0
          phase = if ph == 0.0 then 0.0 else pi / ph
          onsetDur = map (\(Onset b p) -> p) $ fromFoldable $ onsetDurations 1.0
rhythmic
          onsetPos = map (\(Onset b p) -> p) $ rhythmicToOnsets rhythmic
          onsetAcc = fromFoldable $ map (\pos -> sinusoidalAcceleration freq pos amp
phase) onsetPos
          zipped = zip onsetDur onsetAcc



-- trapezoidalRule:: (Number -> Number) -> Number -> Number -> Number
-- trapezoidalRule f a b =
--   let
--     n = 1000
--     h = (b - a) / toNumber n
--     sum = foldl (\acc i -> acc + f (a + toNumber i * h)) 0.0 $ 1..(n-1)
--   in
--     h / 2.0 * (f a + 2.0 * sum + f b)

-- areaUnderCurveSin:: Number -> Number -> Number
-- areaUnderCurveSin start end = trapezoidalRule sin ((start*2.0) * pi) ((end*2.0) *
pi)

-- areaUnderCurveLineal:: Number -> Number -> Number
-- areaUnderCurveLineal start end = trapezoidalRule linearAcc start end
--     where
--             linearAcc:: Number -> Number
--             linearAcc t = 2.0 * t

-- -- rhythmicToSinDur:: Rhythmic -> Array Number
-- rhythmicToSinDur rhythmic = map (\(Tuple dur acc) -> 1.0/(dur * (1.0 + acc)))
zipped
--     where onsetDur = map (\(Onset b p) -> p) $ fromFoldable $ onsetDurations 1.0
rhythmic
--           onsetPos = map (\(Onset b p) -> p) $ rhythmicToOnsets rhythmic
--           onsetAcc = fromFoldable $ map (\pos -> areaUnderCurveSin 0.0 pos)
onsetPos
--           zipped = zip onsetDur onsetAcc
```

```
-- rhythmicToLinDur rhythmic = map (\(Tuple dur acc) -> 1.0/(dur * (1.0 + acc)))
zipped
--     where onsetDur = map (\(Onset b p) -> p) $ fromFoldable $ onsetDurations 1.0
rhythmic
--           onsetPos = map (\(Onset b p) -> p) $ rhythmicToOnsets rhythmic
--           onsetAcc = fromFoldable $ map (\pos -> areaUnderCurveLineal 0.0 pos)
onsetPos
--           zipped = zip onsetDur onsetAcc
```

A.4.11 Aural Specifications

```
module AuralSpecs (auralSpecs) where

import Prelude

import Effect (Effect)
import Effect.Console (log)

import Data.Tuple
import Data.Maybe
import Data.Maybe.First
import Data.Either
import Data.Map as M
import Data.Foldable (sum)
import Data.Int
import Data.FunctorWithIndex (mapWithIndex)
import Data.Array (filter,fromFoldable,(!!), zipWith, replicate, concat, (..), (:),
init, tail, last,head,reverse,zip, cons, uncons, snoc, length, singleton)
import Data.List
import Data.List (fromFoldable,concat,zip,zipWith,length,init) as L
import Data.Traversable (scanl,traverseDefault,sequence)

import Data.Newtype
import Foreign

import Data.Tempo

import AST
import DurationAndIndex
import Parser
import Rhythm
import TestOpsAndDefs
import AssambleWebdirt
import XenoPitch
import Dastgah

import Data.Rational (Rational(..), (%), fromInt)
import Data.Rational (toNumber) as R
import Data.DateTime
```

```
import Data.DateTime.Instant
import Data.Time.Duration

-- aural specs maps on the list of aurals. One aural attribute at the time to process
auralSpecs:: Voices -> Rhythmic -> List Aural -> M.Map String XenoPitch -> Array
Event -> Effect (Array Foreign)
auralSpecs v r aurals x es' = map concat <$> traverseDefault (\a -> auralSpecs' v r a
x es) $ fromFoldable aurals
    where es = filter checkOnset es' -- here O get removed!

auralSpecs':: Voices -> Rhythmic -> Aural -> M.Map String XenoPitch -> Array Event ->
Effect (Array Foreign)
auralSpecs' voices rhy aural xenopitch events
   | (checkForSound aural) = pure []
   | otherwise = traverseDefault (processEvent voices rhy aural xenopitch) events

checkForSound:: List Value -> Boolean
checkForSound aural = not $ elem true $ map isSound aural

checkOnset:: Event -> Boolean
checkOnset (Event o i) = (\(Onset b p) -> b) o

processEvent:: Voices -> Rhythmic -> List Value -> M.Map String XenoPitch -> Event ->
Effect Foreign
processEvent v r vals xp ev = do
  let when = processWhen ev
  let s = processSound v r (getS vals) ev
  let n = processN v r (getN vals) ev
  let gain = processGain v r (getG vals) ev
  let pan = processPan v r (getP vals) ev
  let speed = processSpeed v r (getSpeed vals) ev
  let begin = processBegin v r (getBegin vals) ev
  let end = processEnd v r (getEnd vals) ev
  let vowel = processVowel v r (getVowel vals) ev
  let cutoff = processCutOff v r (getCutOff vals) ev
  let cutoffh = processCutOffH v r (getCutOffH vals) ev
  let maxw = processMaxW v r (getMaxW vals) ev
  let minw = processMinW v r (getMinW vals) ev
  let inter = processInter v r (getInter vals) ev
  let legato = processLegato v r (getLegato vals) ev
  let orbit = processOrbit v r (getOrbit vals) ev
  let note = processNote v xp r (getNote vals) (getXNote vals) ev
  makeWebDirtEvent when s n gain pan speed begin end vowel cutoff cutoffh maxw minw
inter legato orbit note

makeWebDirtEvent:: Number -> String -> Int -> Maybe Number -> Maybe Number -> Maybe
Number -> Maybe Number -> Maybe Number -> Maybe String -> Maybe Number -> Maybe
Number -> Maybe Number -> Maybe Number -> Maybe Number -> Maybe Number -> Maybe Int -
> Maybe Number -> Effect Foreign
makeWebDirtEvent when s n gain pan speed begin end vowel cutoff cutoffh maxw minw
inter legato orbit note = do
  oEvent <- objectWithWhenSN when s n
```

```
  oG <- optVNum oEvent gain addGain
  oP <- optVNum oG pan addPan
  oSp <- optVNum oP speed addSpeed
  oB <- optVNum oSp begin addBegin
  oE <- optVNum oB end addEnd
  oCOff <- optVNum oE cutoff addCutOff
  oCOffH <- optVNum oCOff cutoffh addCutOffH
  oMax <- optVNum oCOffH maxw addMaxW
  oMin <- optVNum oMax minw addMinW
  oInter <- optVNum oMin inter addInter
  oLeg <- optVNum oInter legato addLegato
  oOrbit <- optVInt oLeg orbit addOrbit
  oV <- optVStr oOrbit vowel addVowel
  oN <- optVNum oV note addNote
  pure oN

optVNum:: Foreign -> Maybe Number -> (Foreign -> Number -> Effect Foreign) -> Effect
Foreign
optVNum o Nothing _ = pure o
optVNum o (Just x) f = f o x

optVStr:: Foreign -> Maybe String -> (Foreign -> String -> Effect Foreign) -> Effect
Foreign
optVStr o Nothing _ = pure o
optVStr o (Just x) f = f o x

optVInt:: Foreign -> Maybe Int -> (Foreign -> Int -> Effect Foreign) -> Effect
Foreign
optVInt o Nothing _ = pure o
optVInt o (Just x) f = f o x

processNote:: Voices -> M.Map String XenoPitch -> Rhythmic -> Maybe Value -> Maybe
Value -> Event -> Maybe Number
processNote _ xp r Nothing xNotes e = Nothing
processNote m xp r (Just (TransposedPitch id n)) xn e = findRefdNote m xp r e (Tuple
id n) xn
processNote _ xp r (Just (Prog span lista)) xNotes e = mergeProgWithNote xp r (Prog
span lista) xNotes e
processNote _ xp r (Just (Dastgah span d)) _ e = spanMaybe span newList e r
  where newList = getMIDIInterval $ analysisDastgahPattern span r $ fromFoldable
(getDastgahList d)
processNote _ xp r (Just (Xeno id span lista)) xn e = spanMaybe span (fromFoldable
midiIntervals) e r
  where target = getXPTarget (fst id) xp
        -- target = fromMaybe (EDO 0.0 0) $ M.lookup (fst id) xp
        midiIntervals = xenoPitchAsAuralPattern (Tuple target (snd id)) (fromFoldable
lista) span r
processNote _ _ _ _ _ _ = Nothing


getXPTarget:: String -> M.Map String XenoPitch -> XenoPitch
getXPTarget "shurNot8" _ = ShurNot8
```

```
getXPTarget "shurNot" _ = ShurNot
getXPTarget id xp = fromMaybe (EDO 0.0 0) $ M.lookup id xp

findRefdNote:: Voices -> M.Map String XenoPitch -> Rhythmic -> Event -> Tuple String
Int -> Maybe Value -> Maybe Number
findRefdNote m xp r e (Tuple id n) xn = processNote m xp r newVal xn e
    where newVal = cycleAurals n (M.lookup id m) getNote

mergeProgWithNote:: M.Map String XenoPitch -> Rhythmic -> Value -> Maybe Value ->
Event -> Maybe Number
mergeProgWithNote xp r prog xnote ev = pitchSystemNoteToMIDI xp prog' <$> xnote'
    where prog' = processProg r prog ev
          xnote' = processXNotes r xnote ev

pitchSystemNoteToMIDI:: M.Map String XenoPitch -> (Tuple String (Maybe Int)) -> Int -
> Number
pitchSystemNoteToMIDI mapa (Tuple id subset) nota = xenoPitchAsMIDINum (Tuple xn
subset) nota
  where xn = fromMaybe (EDO 0.0 0) $ M.lookup id mapa

processProg:: Rhythmic -> Value ->  Event -> Tuple String (Maybe Int)
processProg r (Prog span xs) ev = fromMaybe (Tuple "error" Nothing) $ spanMaybe span
(fromFoldable xs) ev r
processProg r _ ev = Tuple "error" Nothing

processXNotes:: Rhythmic -> Maybe Value ->  Event -> Maybe Int
processXNotes r (Just (XNotes sp xs vars)) ev = processVarsMaybe vars sp xs ev r
processXNotes r _ ev = Nothing

--
processOrbit:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Int
processOrbit vs r Nothing ev = Nothing
processOrbit vs r (Just (TransposedOrbit id n)) ev = findRefdOrbit r ev (Tuple id n)
vs
processOrbit _  r (Just (Orbit sp xList vars)) ev = processVarsMaybe vars sp xList ev
r
processOrbit _ _ _ _ = Nothing

findRefdOrbit:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Int
findRefdOrbit r ws (Tuple id n) mapa = processOrbit mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getOrbit

--
processLegato:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processLegato vs r Nothing ev = Nothing
processLegato vs r (Just (TransposedLegato id n)) ev = findRefdLegato r ev (Tuple id
n) vs
processLegato _  r (Just (Legato sp xList vars)) ev = processVarsMaybe vars sp xList
ev r
processLegato _ _ _ _ = Nothing

findRefdLegato:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
```

```
findRefdLegato r ws (Tuple id n) mapa = processLegato mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getLegato

--
processInter:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processInter vs r Nothing ev = Nothing
processInter vs r (Just (TransposedInter id n)) ev = findRefdInter r ev (Tuple id n)
vs
processInter _  r (Just (Inter sp xList vars)) ev = processVarsMaybe vars sp xList ev
r
processInter _ _ _ _ = Nothing

findRefdInter:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdInter r ws (Tuple id n) mapa = processInter mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getInter

--
processMinW:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processMinW vs r Nothing ev = Nothing
processMinW vs r (Just (TransposedMinW id n)) ev = findRefdMinW r ev (Tuple id n) vs
processMinW _  r (Just (MinW sp xList vars)) ev = processVarsMaybe vars sp xList ev r
processMinW _ _ _ _ = Nothing

findRefdMinW:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdMinW r ws (Tuple id n) mapa = processMinW mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getMinW

--
processMaxW:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processMaxW vs r Nothing ev = Nothing
processMaxW vs r (Just (TransposedMaxW id n)) ev = findRefdMaxW r ev (Tuple id n) vs
processMaxW _  r (Just (MaxW sp xList vars)) ev = processVarsMaybe vars sp xList ev r
processMaxW _ _ _ _ = Nothing

findRefdMaxW:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdMaxW r ws (Tuple id n) mapa = processMaxW mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getMaxW

--
processCutOffH:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processCutOffH vs r Nothing ev = Nothing
processCutOffH vs r (Just (TransposedCutOffH id n)) ev = findRefdCutOffH r ev (Tuple
id n) vs
processCutOffH _  r (Just (CutOffH sp xList vars)) ev = processVarsMaybe vars sp
xList ev r
processCutOffH _ _ _ _ = Nothing

findRefdCutOffH:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdCutOffH r ws (Tuple id n) mapa = processCutOffH mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getCutOffH

--
```

```
processCutOff:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processCutOff vs r Nothing ev = Nothing
processCutOff vs r (Just (TransposedCutOff id n)) ev = findRefdCutOff r ev (Tuple id
n) vs
processCutOff _  r (Just (CutOff sp xList vars)) ev = processVarsMaybe vars sp xList
ev r
processCutOff _ _ _ _ = Nothing

findRefdCutOff:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdCutOff r ws (Tuple id n) mapa = processCutOff mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getCutOff

--
processVowel:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe String
processVowel vs r Nothing ev = Nothing
processVowel vs r (Just (TransposedVowel id n)) ev = findRefdVowel r ev (Tuple id n)
vs
processVowel _  r (Just (Vowel sp xList vars)) ev = processVarsMaybe vars sp xList ev
r
processVowel _ _ _ _ = Nothing

findRefdVowel:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe String
findRefdVowel r ws (Tuple id n) mapa = processVowel mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getVowel

--
processEnd:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processEnd vs r Nothing ev = Nothing
processEnd vs r (Just (TransposedEnd id n)) ev = findRefdEnd r ev (Tuple id n) vs
processEnd _  r (Just (End sp xList vars)) ev = processVarsMaybe vars sp xList ev r
processEnd _ _ _ _ = Nothing

findRefdEnd:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdEnd r ws (Tuple id n) mapa = processEnd mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getEnd

--
processBegin:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processBegin vs r Nothing ev = Nothing
processBegin vs r (Just (TransposedBegin id n)) ev = findRefdBegin r ev (Tuple id n)
vs
processBegin _  r (Just (Begin sp xList vars)) ev = processVarsMaybe vars sp xList ev
r
processBegin _ _ _ _ = Nothing

findRefdBegin:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdBegin r ws (Tuple id n) mapa = processBegin mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getBegin

--
processSpeed:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processSpeed vs r Nothing ev = Nothing
```

```
processSpeed vs r (Just (TransposedSpeed id n)) ev = findRefdSpeed r ev (Tuple id n)
vs
processSpeed _  r (Just (Speed sp xList vars)) ev = processVarsMaybe vars sp xList ev
r
processSpeed _ _ _ _ = Nothing

findRefdSpeed:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdSpeed r ws (Tuple id n) mapa = processSpeed mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getSpeed

--
processPan:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processPan vs r Nothing ev = Nothing
processPan vs r (Just (TransposedPan id n)) ev = findRefdP r ev (Tuple id n) vs
processPan _  r (Just (Pan sp xList vars)) ev = processVarsMaybe vars sp xList ev r
processPan _ _ _ _ = Nothing

findRefdP:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdP r ws (Tuple id n) mapa = processPan mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getP
--
processGain:: Voices -> Rhythmic -> Maybe Value -> Event -> Maybe Number
processGain vs r Nothing ev = Nothing
processGain vs r (Just (TransposedGain id n)) ev = findRefdG r ev (Tuple id n) vs
processGain _  r (Just (Gain sp xList vars)) ev = processVarsMaybe vars sp xList ev r
processGain _ _ _ _ = Nothing

findRefdG:: Rhythmic -> Event -> Tuple String Int -> Voices -> Maybe Number
findRefdG r ws (Tuple id n) mapa = processGain mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getG
--
processN:: Voices -> Rhythmic -> Maybe Value -> Event -> Int
processN vs r Nothing ev = 0
processN vs r (Just (TransposedN id n)) ev = findRefdN r ev (Tuple id n) vs
processN _  r (Just (N span nList vars)) ev = processVarsInt vars span nList ev r
processN _ _ _ _ = 2666

findRefdN:: Rhythmic -> Event -> Tuple String Int -> Voices -> Int
findRefdN r ws (Tuple id n) mapa = processN mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getN
--
processSound:: Voices -> Rhythmic -> Maybe Value -> Event -> String
processSound vs r Nothing ev = "no sound value even with check!"
processSound vs r (Just (TransposedSound id n)) ev = findRefdSound r ev (Tuple id n)
vs
processSound _  r (Just (Sound span sList vars)) ev = processVarsStr vars span sList
ev r
processSound _ _ _ _ = "processSound failed at pattern matching"

--
processVarsStr:: List (Variation String) -> Span -> List String -> Event -> Rhythmic
-> String
```

```
processVarsStr Nil sp xs ev r = spanStr sp (fromFoldable xs) ev r
processVarsStr (Cons v vs) sp xs ev r =
    if isVar v ev
    then spanStr vSpan (fromFoldable vList) ev r
    else processVarsStr vs sp xs ev r
        where vSpan = getVSpan v
              vList = getVListStr v

processVarsInt:: List (Variation Int) -> Span -> List Int -> Event -> Rhythmic -> Int
processVarsInt Nil sp xs ev r = spanInt sp (fromFoldable xs) ev r
processVarsInt (Cons v vs) sp xs ev r =
    if isVar v ev
    then spanInt vSpan (fromFoldable vList) ev r
    else processVarsInt vs sp xs ev r
        where vSpan = getVSpan v
              vList = getVListInt v

---- work here with the maybes!!!!
processVarsMaybe:: forall a. List (Variation a) -> Span -> List a -> Event ->
Rhythmic -> Maybe a
processVarsMaybe Nil sp xs ev r = spanMaybe sp (fromFoldable xs) ev r
processVarsMaybe (Cons v vs) sp xs ev r =
    if isVar v ev
    then spanMaybe vSpan (fromFoldable vList) ev r
    else processVarsMaybe vs sp xs ev r
        where vSpan = getVSpan v
              vList = getVList v

getVSpan:: forall a. Variation a -> Span
getVSpan (Every _ sp _) = sp

getVListStr:: Variation String -> List String
getVListStr (Every _ _ xs) = xs

getVListInt:: Variation Int -> List Int
getVListInt (Every _ _ ns) = ns

getVList:: forall a. Variation a -> List a
getVList (Every _ _ xs) = xs

isVar (Every n _ _) ev = ((getBlockIndex ev)`mod`n) == 0
isVar _ _ = false

findRefdSound:: Rhythmic -> Event -> Tuple String Int -> Voices -> String
findRefdSound r ws (Tuple id n) mapa = processSound mapa r newVal ws
    where newVal = cycleAurals n (M.lookup id mapa) getS

-- Potential here for crazyness: delay, anticipate, swing, contratiempo, microrhythm,
snap...
processWhen:: Event -> Number
processWhen (Event o i) = (\(Onset b p) -> p) o
```

```
---- span functions, three flavours: String, Int, Num
spanMaybe:: forall a. Span -> Array a -> Event -> Rhythmic -> Maybe a
spanMaybe CycleEvent xs event _ = xs !! (getEventIndex event `mod` length xs)
spanMaybe CycleBlock xs event _ = xs !! (getBlockIndex event `mod` length xs)
spanMaybe CycleInBlock xs event _ = xs !! (getStructureIndex event `mod` length xs)
spanMaybe SpreadBlock xs event rhythmic = spreadInBlock xs event rhythmic
spanMaybe _ _ _ _ = Nothing

spanStr:: Span -> Array String -> Event -> Rhythmic -> String
spanStr CycleEvent xs event _ = strMaybe $ xs !! (getEventIndex event `mod` length
xs)
spanStr CycleBlock xs event _ = strMaybe $ xs !! (getBlockIndex event `mod` length
xs)
spanStr CycleInBlock xs event _ = strMaybe $ xs !! (getStructureIndex event `mod`
length xs)
spanStr SpreadBlock xs event rhythmic = strMaybe $ spreadInBlock xs event rhythmic
spanStr _ _ _ _ = "error at spanStr, invalid span constructor"

spanInt:: Span -> Array Int -> Event -> Rhythmic -> Int
spanInt CycleEvent xs event _ = intMaybe $ xs !! (getEventIndex event `mod` length
xs)
spanInt CycleBlock xs event _ = intMaybe $ xs !! (getBlockIndex event `mod` length
xs)
spanInt CycleInBlock xs event _ = intMaybe $ xs !! (getStructureIndex event `mod`
length xs)
spanInt SpreadBlock xs event rhythmic = intMaybe $ spreadInBlock xs event rhythmic
spanInt _ _ _ _ = 2666

------
-- spread functions are now general for all values!!! Bliss
spreadInBlock:: forall a. Array a -> Event -> Rhythmic -> Maybe a
spreadInBlock xs event rhythmic = spreadWrap percenPos xsLimits
  where percenPositions = map (\(Onset b p) -> p) $ rhythmicToOnsets rhythmic
        modIndex = (getEventIndex event) `mod` (length $ fromFoldable
percenPositions)
        percenPos = fromMaybe 0.0 $ (fromFoldable percenPositions) !! modIndex
        segment = 1.0 / toNumber (length xs)
        limitsFst = cons 0.0 (scanl (+) 0.0 $ replicate ((length xs) - 1) segment)
        limitsSnd = snoc (scanl (+) 0.0 $ replicate ((length xs) - 1) segment) 1.0
        xsLimits = zip xs $ zip limitsFst limitsSnd

spreadWrap:: forall a. Number -> Array (Tuple a (Tuple Number Number)) -> Maybe a
spreadWrap percenPos asWithlimits = fromMaybe Nothing $ head $ filter isJust $ map
(\(Tuple as limits) -> spread percenPos as limits) asWithlimits

spread:: forall a. Number -> a -> (Tuple Number Number) -> Maybe a
spread percenPos a limits = if (percenPos >= fst limits) && (percenPos < snd limits)
then (Just a) else Nothing

---- helpers
strMaybe:: Maybe String -> String
strMaybe x = fromMaybe "error" x
```

```
intMaybe:: Maybe Int -> Int
intMaybe x = fromMaybe 2666 x

numMaybe:: Maybe Number -> Number
numMaybe x = fromMaybe 2.666 x

cycleAurals:: Int -> Maybe Voice -> (List Value -> Maybe Value) -> Maybe Value
cycleAurals n mVoice f = do
  voice <- mVoice
  let aurals = (\(Voice t aurals) -> aurals) voice
  let len = L.length aurals
  newVal <- (fromFoldable aurals) !! (n`mod`len)
  f newVal

-- getters
----- structure index is weird, think of nested levels
getStructureIndex:: Event -> Int
getStructureIndex (Event _ (Index _ xs _)) = fromMaybe 0 $ head $ xs

getBlockIndex:: Event -> Int
getBlockIndex (Event _ (Index n _ _)) = n

getEventIndex:: Event -> Int
getEventIndex (Event _ (Index _ _ n)) = n

getXNote:: List Value -> Maybe Value
getXNote aural = head $ filter isXNote $ fromFoldable aural

isXNote:: Value -> Boolean
isXNote (XNotes _ _ _) = true
isXNote _ = false

getNote:: List Value -> Maybe Value
getNote aural = head $ filter isNote $ fromFoldable aural

isNote:: Value -> Boolean
isNote (Dastgah _ _) = true
isNote (Xeno _ _ _) = true
isNote (Prog _ _) = true
isNote _ = false

getDastgahList:: Dastgah -> List Int
getDastgahList (Shur ns) = ns
getDastgahList _ = Nil


getMaxW:: List Value -> Maybe Value
getMaxW aural = head $ filter isMaxW $ fromFoldable aural

isMaxW:: Value -> Boolean
isMaxW (MaxW _ _ _) = true
```

```
isMaxW (TransposedMaxW _ _) = true
isMaxW _ = false

getMinW:: List Value -> Maybe Value
getMinW aural = head $ filter isMinW $ fromFoldable aural

isMinW:: Value -> Boolean
isMinW (MinW _ _ _) = true
isMinW (TransposedMinW _ _) = true
isMinW _ = false

getOrbit:: List Value -> Maybe Value
getOrbit aural = head $ filter isOrbit $ fromFoldable aural

isOrbit:: Value -> Boolean
isOrbit (Orbit _ _ _) = true
isOrbit (TransposedOrbit _ _) = true
isOrbit _ = false

getLegato:: List Value -> Maybe Value
getLegato aural = head $ filter isLegato $ fromFoldable aural

isLegato:: Value -> Boolean
isLegato (Legato _ _ _) = true
isLegato (TransposedLegato _ _) = true
isLegato _ = false

getInter:: List Value -> Maybe Value
getInter aural = head $ filter isInter $ fromFoldable aural

isInter:: Value -> Boolean
isInter (Inter _ _ _) = true
isInter (TransposedInter _ _) = true
isInter _ = false

getCutOffH:: List Value -> Maybe Value
getCutOffH aural = head $ filter isCutOffH $ fromFoldable aural

isCutOffH:: Value -> Boolean
isCutOffH (CutOffH _ _ _) = true
isCutOffH (TransposedCutOffH _ _) = true
isCutOffH _ = false

getCutOff:: List Value -> Maybe Value
getCutOff aural = head $ filter isCutOff $ fromFoldable aural

isCutOff:: Value -> Boolean
isCutOff (CutOff _ _ _) = true
isCutOff (TransposedCutOff _ _) = true
isCutOff _ = false

getVowel:: List Value -> Maybe Value
```

```
getVowel aural = head $ filter isVowel $ fromFoldable aural

isVowel:: Value -> Boolean
isVowel (Vowel _ _ _) = true
isVowel (TransposedVowel _ _) = true
isVowel _ = false

getEnd:: List Value -> Maybe Value
getEnd aural = head $ filter isEnd $ fromFoldable aural

isEnd:: Value -> Boolean
isEnd (End _ _ _) = true
isEnd (TransposedEnd _ _) = true
isEnd _ = false

getBegin:: List Value -> Maybe Value
getBegin aural = head $ filter isBegin $ fromFoldable aural

isBegin:: Value -> Boolean
isBegin (Begin _ _ _) = true
isBegin (TransposedBegin _ _) = true
isBegin _ = false

getSpeed:: List Value -> Maybe Value
getSpeed aural = head $ filter isSpeed $ fromFoldable aural

isSpeed:: Value -> Boolean
isSpeed (Speed _ _ _) = true
isSpeed (TransposedSpeed _ _) = true
isSpeed _ = false

getP:: List Value -> Maybe Value
getP aural = head $ filter isP $ fromFoldable aural

isP:: Value -> Boolean
isP (Pan _ _ _) = true
isP (TransposedPan _ _) = true
isP _ = false

getG:: List Value -> Maybe Value
getG aural = head $ filter isG $ fromFoldable aural

isG:: Value -> Boolean
isG (Gain _ _ _) = true
isG (TransposedGain _ _) = true
isG _ = false

getN:: List Value -> Maybe Value
getN aural = head $ filter isN $ fromFoldable aural

isN:: Value -> Boolean
isN (N _ _ _) = true
```

```
isN (TransposedN _ _) = true
isN _ = false

getS:: List Value -> Maybe Value
getS aural = head $ filter isSound $ fromFoldable aural

isSound:: Value -> Boolean
isSound (Sound _ _ _) = true
isSound (TransposedSound _ _) = true
isSound _ = false
```

A.4.12 Assemble WebDirt

```
module AssambleWebdirt (objectWithWhenSN, addGain, addPan, addSpeed, addBegin,
addEnd, addVowel, addMaxW, addMinW, addInter, addLegato, addOrbit, addCutOff,
addCutOffH, addNote) where

import Prelude
import Effect (Effect)
import Foreign

-- foreign
foreign import objectWithWhenSN :: Number -> String -> Int -> Effect Foreign
-- export objectWithWhenSN = when => s => n => () => { return { when: when, s: s, n:
n }; }

foreign import addGain :: Foreign -> Number -> Effect Foreign
-- export addGain = o => gain => () => { o.gain = gain; return o; }

foreign import addPan :: Foreign -> Number -> Effect Foreign
-- export addPan = o => pan => () => { o.pan = pan; return o; }

foreign import addSpeed :: Foreign -> Number -> Effect Foreign
-- export addSpeed = o => speed => () => { o.speed = speed; return o; }

foreign import addBegin :: Foreign -> Number -> Effect Foreign
-- export addBegin = o => begin => () => { o.begin = begin; return o; }

foreign import addEnd :: Foreign -> Number -> Effect Foreign
-- export addEnd = o => end => () => { o.end = end; return o; }

foreign import addVowel :: Foreign -> String -> Effect Foreign
-- export addVowel = o => vowel => () => { o.vowel = vowel; return o; }

foreign import addCutOff :: Foreign -> Number -> Effect Foreign
-- export addCutoff = o => cutoff => () => { o.cutoff = cutoff; return o; }

foreign import addCutOffH :: Foreign -> Number -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }
```

```
foreign import addMaxW :: Foreign -> Number -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

foreign import addMinW :: Foreign -> Number -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

foreign import addInter :: Foreign -> Number -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

foreign import addLegato :: Foreign -> Number -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

foreign import addOrbit :: Foreign -> Int -> Effect Foreign
-- export addCutoffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

foreign import addNote :: Foreign -> Number -> Effect Foreign
-- export const addNote = o => note => () => { o.note = note; return o; }
```

A.4.13 Assemble WebDirt JS

```
"use strict";

export const objectWithWhenSN = when => s => n => () => { return { whenPosix: when,
s: s, n: n }; }

export const addGain = o => gain => () => { o.gain = gain; return o; }

export const addPan = o => pan => () => { o.pan = pan; return o; }

export const addSpeed = o => speed => () => { o.speed = speed; return o; }

export const addBegin = o => begin => () => { o.begin = begin; return o; }

export const addEnd = o => end => () => { o.end = end; return o; }

export const addVowel = o => vowel => () => { o.vowel = vowel; return o; }

export const addCutOff = o => cutoff => () => { o.cutoff = cutoff; return o; }

export const addCutOffH = o => cutoffh => () => { o.hcutoff = cutoffh; return o; }

export const addMaxW = o => maxw => () => { o.maxw = maxw; return o; }

export const addMinW = o => minw => () => { o.minw = minw; return o; }

export const addInter = o => inter => () => { o.inter = inter; return o; }

export const addLegato = o => legato => () => { o.legato = legato; return o; }
```

```
export const addOrbit = o => orbit => () => { o.orbit = orbit; return o; }

export const addNote = o => note => () => { o.note = note; return o; }
```

A.4.14 Erv

```
module Erv where

import AST (XenoNote)

foreign import ratioToCents :: Number -> Number

foreign import makeCPSScale :: Int -> Array Int -> Array XenoNote




-- foreign import pruebilla :: Int -> Int -> Int


--- objeto resultante de cps.make , typear..?
-- {
--   meta: {
--     scale: 'cps',
--     period: 2,
--     size: 2,
--     factors: [ 1, 3, 5, 7 ],
--     'normalized-by': 1,
--     type: '2)4'
--   },
--   scale: [
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 35,
--       'bounded-ratio': 1.09375,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 5,
--       'bounded-ratio': 1.25,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
```

```
--       ratio: 21,
--       'bounded-ratio': 1.3125,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 3,
--       'bounded-ratio': 1.5,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 7,
--       'bounded-ratio': 1.75,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 15,
--       'bounded-ratio': 1.875,
--       'bounding-period': 2
--     }
--   ],
--   nodes: [
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 3,
--       'bounded-ratio': 1.5,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 5,
--       'bounded-ratio': 1.25,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 7,
--       'bounded-ratio': 1.75,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 15,
```

```
--       'bounded-ratio': 1.875,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 21,
--       'bounded-ratio': 1.3125,
--       'bounding-period': 2
--     },
--     {
--       set: [Array],
--       'archi-set': [Array],
--       ratio: 35,
--       'bounded-ratio': 1.09375,
--       'bounding-period': 2
--     }
--   ],
--   graphs: {
--     full: {
--       '{:set #{1 3}, :archi-set #{:a :b}, :ratio 3, :bounded-ratio 1.5, :bounding-
period 2}': [Array],
--       '{:set #{1 5}, :archi-set #{:a :c}, :ratio 5, :bounded-ratio 1.25,
:bounding-period 2}': [Array],
--       '{:set #{1 7}, :archi-set #{:a :d}, :ratio 7, :bounded-ratio 1.75,
:bounding-period 2}': [Array],
--       '{:set #{3 5}, :archi-set #{:b :c}, :ratio 15, :bounded-ratio 1.875,
:bounding-period 2}': [Array],
--       '{:set #{3 7}, :archi-set #{:b :d}, :ratio 21, :bounded-ratio 1.3125,
:bounding-period 2}': [Array],
--       '{:set #{5 7}, :archi-set #{:c :d}, :ratio 35, :bounded-ratio 1.09375,
:bounding-period 2}': [Array]
--     },
--     simple: {
--       '#{1 3}': [Array],
--       '#{1 5}': [Array],
--       '#{1 7}': [Array],
--       '#{3 5}': [Array],
--       '#{3 7}': [Array],
--       '#{5 7}': [Array]
--     }
--   }
-- }
```

A.4.15 Erv JS

```
import * as Erv from "@diegovdc/erv";

export const ratioToCents = Erv.default.utils.ratioToCents;
```

```
export const makeCPSScale = (size) => (factor) =>
Erv.default.cps.make(size,factor).scale;
```

A.4.16 Dastgah

```
module Dastgah (analysisDastgahPattern, getMIDIInterval) where

import Prelude
import Data.Int (toNumber)
import Data.Array (filter,fromFoldable,(!!), zipWith, replicate, concat, (..), (:),
init, tail, last,head,reverse,zip, cons, uncons, snoc, length, singleton)
import Data.List
import Data.List (fromFoldable,concat,zip,zipWith,length,init,uncons) as L
import Data.Maybe
import Data.Tuple


import AST
import DurationAndIndex

getMIDIInterval:: Array DastgahNote -> Array Number
getMIDIInterval xs = map (\x -> x.midiInterval) xs

analysisDastgahPattern::Span -> Rhythmic -> Array Int -> Array DastgahNote
analysisDastgahPattern CycleEvent _ ns = map assambleDastgahNote zipped
  where first = ns
        s = fromMaybe {head: 0, tail: []} $ uncons ns
        second = snoc s.tail s.head
        zipped = zip first second
analysisDastgahPattern CycleBlock _ ns = map assambleDastgahNote zipped
  where first = ns
        s = fromMaybe {head: 0, tail: []} $ uncons ns
        second = snoc s.tail s.head
        zipped = zip first second
analysisDastgahPattern CycleInBlock r ns = map assambleDastgahNote zipped
  where structure = map (\x -> (x `mod` (length ns))) $ map (\x -> fromMaybe 0 $ head
x) $ rhythmicStructIndex r [0]
        seque = map (\x -> fromMaybe 0 (ns !! x)) structure
        first = seque
        s = fromMaybe {head: 0, tail: []} $ uncons seque
        second = snoc s.tail s.head
        zipped = zip first second
analysisDastgahPattern SpreadBlock r ns = map assambleDastgahNote zipped
  where percenPositions = map (\(Onset b p) -> p) $ rhythmicToOnsets r -- xxx[xx] : 0
0.25 0.5 0.75 0.875
        segment = 1.0 / toNumber (length ns)  -- 1 3 5 : 0.333333
        limitsFst = cons 0.0 (scanl (+) 0.0 $ replicate ((length ns) - 1) segment) --
[0, 0.333, 0.666]
```

```
        limitsSnd = snoc (scanl (+) 0.0 $ replicate ((length ns) - 1) segment) 1.0 --
[0.333, 0.666, 1]
        limits = zip limitsFst limitsSnd  -- [(0,0.333), (0.333,0.666), (0.666,1)]
        noteLimits = zip ns limits -- [(1,(0,0.333)),(3,(0.333,0.666)),(5,(0.666,1))]
        funka:: Array (Tuple Int (Tuple Number Number)) -> Number -> Array Int
        funka noteLimits percenPos = map fst $ filter (\(Tuple _ limit) -> (percenPos
>= (fst limit)) && (percenPos < (snd limit))) noteLimits
        realNS = concat $ map (\percenPos -> funka noteLimits percenPos) $
fromFoldable percenPositions
        first = realNS
        s = fromMaybe {head: 0, tail: []} $ uncons realNS
        second = snoc s.tail s.head
        zipped = zip first second

assambleDastgahNote:: Tuple Int Int -> DastgahNote
assambleDastgahNote (Tuple x y) = {function: fu, movement: mov, midiInterval:
checkedMidiInt}
  where (Tuple midiInter fu) = shurIntToFuncAndMIDIInt x
        mov = getMovement x y
        checkedMidiInt = if x == 6 then checkSixth mov midiInter else midiInter

checkSixth:: Interval -> Number -> Number
checkSixth UpJump midiInter = midiInter + 0.92
checkSixth UpNext midiInter = midiInter + 0.92
checkSixth _ midiInter = midiInter

getMovement:: Int -> Int -> Interval
getMovement note target
  | (note < target) && (target == (note+1)) = UpNext
  | (note < target) && (target /= (note+1)) = UpJump
  | (note > target) && (target == (note-1)) = DownNext
  | (note > target) && (target /= (note-1)) = DownJump
  | note == target = Unison
  | otherwise = Unison


---- need octaves!!! -- start from 0 to 6
shurIntToFuncAndMIDIInt:: Int -> Tuple Number String
shurIntToFuncAndMIDIInt n = case (n-1)`mod`8 of
                          0 -> Tuple 0.0 "unknown"
                          1 -> Tuple 1.82 "unknown"
                          2 -> Tuple 2.96 "unknown"
                          3 -> Tuple 5.0 "unknown"
                          4 -> Tuple 7.04 "unknown"
                          5 -> Tuple 7.94 "unknown" -- upwards move 8.86
                          6 -> Tuple 9.98 "unknown"
                          7 -> Tuple 12.0 "unknown"
                          _ -> Tuple 0.0 "unknown"


-- 0 1.82 2.96 5.0 7.04 7.94 9.98 12.0
-- data Dastgah = Shur (List Int)
```

```
dToList:: Dastgah -> List Number
dToList (Shur ns) = map f ns
  where f n = case (n-1)`mod`8 of
                0 -> 0.0
                1 -> 1.82
                2 -> 2.96
                3 -> 5.0
                4 -> 7.04
                5 -> 7.94
                6 -> 9.98
                7 -> 12.0
                _ -> 0.0
dToList _ = Nil
```

A.4.17 Tuning Systems

```
module XenoPitch (xenoPitchAsAuralPattern, xenoPitchToMIDIInterval, testXN,
xenoPitchAsMIDINum) where

import Prelude

import Partial.Unsafe

import Data.Array ((:), elem, filter,unsafeIndex, length, sortWith, zip, (!!),
fromFoldable, cons, uncons, snoc, init, tail, last,head,reverse, replicate, concat)

import Data.List (scanl)

import Data.Tuple
import Data.Int (toNumber, floor)
import Data.Number (floor) as N
import Data.Maybe

import Data.Set (Set(..))
import Data.Set as Set

import Erv (makeCPSScale,ratioToCents)
import AST
import DurationAndIndex

---- this top is the list processed in AuralSpecs along with span and Value
-- top:: XenoPitch -> Array Number

-- lista: [0,1,2,3]
-- xnAsMIDI: [0, 2.5, 3.3, 5.5, 7.7, 8.8, 10.1]

-- data XenoPitch = CPSet Int (Array Int) (Array Int) | MOS Int Int | EDO Number Int
testXP = CPSet 2 [1,3,5,7] (Just $ [Unions [1,3]])
testXN = xenoPitchAsAuralPattern (Tuple testXP $ Just 1) [0,1,2,3]
```

```
--- new fucntion that receives tuning system / note and index and can produce the
midiInterval required

xenoPitchAsMIDINum:: Tuple XenoPitch (Maybe Int) -> Int -> Number
xenoPitchAsMIDINum (Tuple xn (Just i)) nota = asMIDI
    where   scaleAsMIDISubsets = xenoPitchToMIDIInterval xn -- Array Array Num
            subset = fromMaybe [0.0] $ scaleAsMIDISubsets !! i
            lengthOfSet = length subset
            (Tuple index octave) = cycleAndOctavesOfPatternInSet' nota lengthOfSet
            asMIDI = (fromMaybe (0.0) $ subset !! index) + octave
xenoPitchAsMIDINum (Tuple xn Nothing) nota = asMIDI
    where   scaleAsMIDISubsets = xenoPitchToMIDIInterval xn -- Array Array Num
            subset = fromMaybe [2.666] $ scaleAsMIDISubsets !! 0
            lengthOfSet = length subset
            (Tuple index octave) = cycleAndOctavesOfPatternInSet' nota lengthOfSet
            asMIDI = (fromMaybe (0.0) $ subset !! index) + octave

cycleAndOctavesOfPatternInSet':: Int -> Int -> Tuple Int Number
cycleAndOctavesOfPatternInSet' n setLen = Tuple cycledNote isOctave
    where cycledNote = n `mod` setLen
          isOctave = toNumber $ (floor $ (toNumber n) / (toNumber setLen))*12

xenoPitchAsAuralPattern:: Tuple XenoPitch (Maybe Int) -> Array Int -> Span ->
Rhythmic -> Array Number
xenoPitchAsAuralPattern (Tuple ShurNot Nothing) lista sp r = map (\n-> n.midiInterval
+ (386.3137138648348*0.01)) $ map assambleShurNot shurNot
    where shurNot = analysisShurNotPattern sp r lista
xenoPitchAsAuralPattern (Tuple ShurNot8 Nothing) lista sp r = map (\n->
n.midiInterval) $ map assambleShurNot8 shurNot8
    where shurNot8 = analysisShurNotPattern sp r lista
xenoPitchAsAuralPattern (Tuple xn (Just i)) lista _ _ = asMIDI
    where   scaleAsMIDISubsets = xenoPitchToMIDIInterval xn -- Array Array Num
            subset = fromMaybe [0.0] $ scaleAsMIDISubsets !! i
            lengthOfSet = length subset
            cyclesAndOctave = cycleAndOctavesOfPatternInSet lista lengthOfSet
            asMIDI = map (\(Tuple index octave) -> (fromMaybe (0.0) $ subset !!
index) + octave) cyclesAndOctave
xenoPitchAsAuralPattern (Tuple xn Nothing) lista _ _ = asMIDI
    where   scaleAsMIDISubsets = xenoPitchToMIDIInterval xn -- Array Array Num
            subset = fromMaybe [2.666] $ scaleAsMIDISubsets !! 0
            lengthOfSet = length subset
            cyclesAndOctave = cycleAndOctavesOfPatternInSet lista lengthOfSet
            asMIDI = map (\(Tuple index octave) -> (fromMaybe (0.0) $ subset !!
index) + octave) cyclesAndOctave

cycleAndOctavesOfPatternInSet:: Array Int -> Int -> Array (Tuple Int Number)
cycleAndOctavesOfPatternInSet ns setLen = zip cycledList isOctave
    where cycledList = map (\n -> n `mod` setLen) ns
          isOctave = map (\n -> toNumber $ (floor $ (toNumber n) / (toNumber
setLen))*12 ) ns
```

```
addOctave:: Int -> Number
addOctave n = 12.0 * (N.floor $ (toNumber n) / 12.0)

centaura:: Int -> Number
centaura n = case n`mod`12 of
                0 -> 0.0
                1 -> 53.27294323014412*0.01
                2 -> 203.91000173077484*0.01
                3 -> 266.8709056037379*0.01
                4 -> 386.3137138648348*0.01
                5 -> 498.04499913461217*0.01
                6 -> 551.3179423647567*0.01
                7 -> 701.9550008653874*0.01
                8 -> 764.9159047383506*0.01
                9 -> 884.3587129994477*0.01
                10 -> 968.8259064691249*0.01
                11 -> 1088.2687147302222*0.01
                _ -> 0.0


analysisShurNotPattern:: Span -> Rhythmic -> Array Int -> Array (Tuple Int Int)
analysisShurNotPattern CycleEvent _ ns = zipped
  where first = ns
        s = fromMaybe {head: 0, tail: []} $ uncons ns
        second = snoc s.tail s.head
        zipped = zip first second
analysisShurNotPattern CycleBlock _ ns = zipped
  where first = ns
        s = fromMaybe {head: 0, tail: []} $ uncons ns
        second = snoc s.tail s.head
        zipped = zip first second
analysisShurNotPattern CycleInBlock r ns = zipped
  where structure = map (\x -> (x `mod` (length ns))) $ map (\x -> fromMaybe 0 $ head
x) $ rhythmicStructIndex r [0]
        seque = map (\x -> fromMaybe 0 (ns !! x)) structure
        first = seque
        s = fromMaybe {head: 0, tail: []} $ uncons seque
        second = snoc s.tail s.head
        zipped = zip first second
analysisShurNotPattern SpreadBlock r ns = zipped
  where percenPositions = map (\(Onset b p) -> p) $ rhythmicToOnsets r -- xxx[xx] : 0
0.25 0.5 0.75 0.875
        segment = 1.0 / toNumber (length ns)  -- 1 3 5 : 0.333333
        limitsFst = cons 0.0 (scanl (+) 0.0 $ replicate ((length ns) - 1) segment) --
[0, 0.333, 0.666]
        limitsSnd = snoc (scanl (+) 0.0 $ replicate ((length ns) - 1) segment) 1.0 --
[0.333, 0.666, 1]
        limits = zip limitsFst limitsSnd  -- [(0,0.333), (0.333,0.666), (0.666,1)]
        noteLimits = zip ns limits -- [(1,(0,0.333)),(3,(0.333,0.666)),(5,(0.666,1))]
        funka:: Array (Tuple Int (Tuple Number Number)) -> Number -> Array Int
        funka noteLimits percenPos = map fst $ filter (\(Tuple _ limit) -> (percenPos
>= (fst limit)) && (percenPos < (snd limit))) noteLimits
```

```
        realNS = concat $ map (\percenPos -> funka noteLimits percenPos) $
fromFoldable percenPositions
        first = realNS
        s = fromMaybe {head: 0, tail: []} $ uncons realNS
        second = snoc s.tail s.head
        zipped = zip first second

assambleShurNot8:: Tuple Int Int -> ShurNot
assambleShurNot8 (Tuple x y) = {movement: mov, midiInterval: checkedMidiInt}
  where midiInter = shur8IntToMIDIInt x
        mov = getMovement x y
        checkedMidiInt = midiInter

 --- this tuple is the interval to analyse
assambleShurNot:: Tuple Int Int -> ShurNot
assambleShurNot (Tuple x y) = {movement: mov, midiInterval: checkedMidiInt}
  where midiInter = shurIntToMIDIInt x
        mov = getMovement x y
        secondChecked = if x == 8 then checkSec mov midiInter else midiInter
        sixthChecked = if x == 12 then checkSixth mov midiInter else secondChecked
        checkedMidiInt = sixthChecked

checkSec:: Interval -> Number -> Number
checkSec DownJump midiInter = midiInter + ((165.00422849992202 * 0.01) -
(111.73128526977847 * 0.01) - midiInter)
checkSec DownNext midiInter = midiInter + ((165.00422849992202 * 0.01) -
(111.73128526977847 * 0.01) - midiInter)
checkSec _ midiInter = midiInter

checkSixth:: Interval -> Number -> Number
checkSixth UpJump midiInter = midiInter + (866.9592293653092 * 0.01) -
(813.6862861351652 * 0.01)
checkSixth UpNext midiInter = midiInter + (866.9592293653092 * 0.01) -
(813.6862861351652 * 0.01)
checkSixth _ midiInter = midiInter


getMovement:: Int -> Int -> Interval
getMovement note target
  | (note < target) && (target == (note+1)) = UpNext
  | (note < target) && (target /= (note+1)) = UpJump
  | (note > target) && (target == (note-1)) = DownNext
  | (note > target) && (target /= (note-1)) = DownJump
  | note == target = Unison
  | otherwise = Unison


shur8IntToMIDIInt:: Int -> Number
shur8IntToMIDIInt n = case n`mod`26 of
                          0 -> 0.0 - 24.0
                          1 -> (701.955000865387 * 0.01) - 24.0
                          2 -> 0.0 - 12.0
```

```
                          3 -> (266.8709056037379*0.01) - 12.0
                          4 -> (498.04499913461217*0.01) - 12.0
                          5 -> (701.9550008653874*0.01) - 12.0
                          6 -> (968.8259064691249*0.01) - 12.0
                          7 -> 0.0
                          8 -> 203.91000173077484 * 0.01 -- desciende con
111.73128526977847
                          9 -> 266.8709056037379 * 0.01
                          10 -> 498.04499913461217 * 0.01
                          11 -> 701.9550008653874 * 0.01
                          12 -> 884.3587129994477 * 0.01  -- asciende con
866.9592293653092
                          13 -> 968.8259064691249 * 0.01
                          14 -> 12.0
                          15 -> 12.0 + (53.27294323014412 * 0.01)
                          16 -> 12.0 + (203.91000173077484 * 0.01)
                          17 -> 12.0 + (266.8709056037379 * 0.01)
                          18 -> 12.0 + (386.3137138648348 * 0.01)
                          19 -> 12.0 + (498.04499913461217 * 0.01)
                          20 -> 12.0 + (551.3179423647567 * 0.01)
                          21 -> 12.0 + (701.9550008653874 * 0.01)
                          22 -> 12.0 + (764.9159047383506  * 0.01)
                          23 -> 12.0 + (884.358712999447 * 0.01)
                          24 -> 12.0 + (968.8259064691249 * 0.01)
                          25 -> 12.0 + (1088.2687147302222 * 0.01)
                          _ -> 0.0

-- [0
--   (0.0 *
--    53.27294323014412
--    203.91000173077484 *
--    266.8709056037379 *
--    386.3137138648348
--    498.04499913461217 *
--    551.3179423647567
--    701.9550008653874 *
--    764.9159047383506
--    884.3587129994477 *
--    968.8259064691249 *
--    1088.2687147302222)]


shurIntToMIDIInt:: Int -> Number
shurIntToMIDIInt n = case n`mod`26 of
                          0 -> 0.0 - 24.0
                          1 -> (701.9550008653874 * 0.01) - 24.0
                          2 -> 0.0 - 12.0
                          3 -> (165.00422849992202*0.01) - 12.0
                          4 -> (315.64128700055255*0.01) - 12.0
                          5 -> (582.51219260429*0.01) - 12.0
                          6 -> (813.6862861351652*0.01) - 12.0
                          7 -> 0.0
```

```
                          8 -> 165.00422849992202 * 0.01 -- desciende con
111.73128526977847
                          9 -> 315.64128700055255 * 0.01
                          10 -> 498.04499913461217 * 0.01
                          11 -> 701.9550008653874 * 0.01
                          12 -> 813.6862861351652 * 0.01  -- asciende con
866.9592293653092
                          13 -> 1017.5962878659401 * 0.01
                          14 -> 12.0
                          15 -> 12.0 + (111.73128526977847 * 0.01)
                          16 -> 12.0 + (165.00422849992202 * 0.01)
                          17 -> 12.0 + (315.64128700055255 * 0.01)
                          18 -> 12.0 + (378.6021908735147 * 0.01)
                          19 -> 12.0 + (498.04499913461217 * 0.01)
                          20 -> 12.0 + (582.51219260429 * 0.01)
                          21 -> 12.0 + (701.9550008653874 * 0.01)
                          22 -> 12.0 + (813.6862861351652 * 0.01)
                          23 -> 12.0 + (866.9592293653092 * 0.01)
                          24 -> 12.0 + (1017.5962878659401 * 0.01)
                          25 -> 12.0 + (1080.557191738903 * 0.01)
                          _ -> 0.0

type ShurNot = {
  movement:: Interval,
  midiInterval:: Number
}

--  [4
--   (0.0  *  e
--    111.73128526977847
--    165.00422849992202  * ft
--    315.64128700055255  *  g
--    378.6021908735147
--    498.04499913461217  *
--    582.51219260429       bb
--    701.9550008653874   *
--    813.6862861351652  *  c   descendente
--    866.9592293653092  *   ascendente
--    1017.5962878659401  *
--    1080.557191738903)]

---- the ordering of subsets is still buggy, figure it out!! Jan 2024
xenoPitchToMIDIInterval:: XenoPitch -> Array (Array Number)
xenoPitchToMIDIInterval (CPSet size factors Nothing) = map (addSampleRoot <<<
toMIDIInterval) [scale]
    where scale = makeCPSScale size factors -- Array XenoNote
xenoPitchToMIDIInterval (CPSet size factors (Just subsets)) = map (addSampleRoot <<<
toMIDIInterval) (scale : subs)
    where scale = makeCPSScale size factors -- Array XenoNote
          subs = map (orderSetofXNotes <<< getSubSet scale) subsets
xenoPitchToMIDIInterval _ = []
```

```
getSubSet:: Array XenoNote -> Subset -> Array XenoNote
getSubSet xn subset = fromFoldable $ getSubset' xn subset

------ this function might not be working properly.... also check the ordering func
above
getSubset':: Array XenoNote -> Subset -> Set XenoNote
getSubset' xn (Subset isInSet) = Set.fromFoldable $ filter (\x -> elem isInSet x.set
) xn
getSubset' xn (Unions ns) = Set.unions $ map f ns
    where f isInSet = Set.fromFoldable $ filter (\x -> elem isInSet x.set ) xn
getSubset' xn (Intersection a b) = Set.intersection a' b'
    where a' = Set.fromFoldable $ filter (\x -> elem a x.set ) xn
          b' = Set.fromFoldable $ filter (\x -> elem b x.set ) xn
getSubset' xn (Difference a b) = Set.difference a' b'
    where a' = Set.fromFoldable $ filter (\x -> elem a x.set ) xn
          b' = Set.fromFoldable $ filter (\x -> elem b x.set ) xn
getSubset' _ _ = Set.fromFoldable []

orderSetofXNotes:: Array XenoNote -> Array XenoNote
orderSetofXNotes s = sortWith (_."bounded-ratio") s

-- sortWith (_.age) [{name: "Alice", age: 42}, {name: "Bob", age: 21}]
--    = [{name: "Bob", age: 21}, {name: "Alice", age: 42}]


toMIDIInterval:: Array XenoNote -> Array Number
toMIDIInterval xns = map toMIDIInterval' xns

toMIDIInterval':: XenoNote -> Number
toMIDIInterval' xn = (ratioToCents xn."bounded-ratio") / 100.0

addSampleRoot:: Array Number -> Array Number
addSampleRoot xs = 0.0 : xs


-- parseo:
-- v0.myCPS[0] = _ 0 1 2 3 4 5 6;

-- {
-- myCPS <- cps 2 (1,3,5,7) | setsWith 3, setsWith 5, setsWith 7;

-- myMOS <- mos 12 7 0;

-- }

-- where myCPS is an array and index 0 is the set, index 1 is setsWith 3, index 2 is
setsWith 5, etc...
-- third arg of mos is rotation
```

A.4.18 Abstract Syntax Tree

```
module AST(TimekNot(..),Vantage(..), TimePoint(..), VantageMap(..), Voices(..),
Voice(..),Program(..),Expression(..),Aural(..),Value(..),
Variation(..),Dastgah(..),Span(..),Temporal(..),Polytemporal(..),Rhythmic(..),
Euclidean(..), Event(..), TimePacket(..), Onset(..), Index(..), TempoMark(..),
Sinusoidal(..), ConvergeTo(..), ConvergeFrom(..), CPAlign(..), XenoPitch(..),
XenoNote(..), DastgahNote(..), Interval(..), Subset(..), showEventIndex,
showStructureIndex) where

import Prelude
import Effect.Ref
import Data.List
import Data.String as Str
import Data.Tempo
import Data.DateTime
import Data.Rational
import Data.Map
import Data.Tuple
import Data.Either
import Data.Maybe

type TimekNot = {
  ast :: Ref Program,
  tempo :: Ref Tempo,
  eval :: Ref DateTime,
  vantageMap :: Ref (Map String DateTime)
  }

type Program = List Expression

data Expression = TimeExpression (Map String Temporal) | AuralExpression (Map String
Aural) | VantagePointExpression  (Map String Vantage) | XenoPitchExpression (Map
String XenoPitch)

instance expressionShow :: Show Expression where
  show (TimeExpression x) = show x
  show (AuralExpression x) = show x
  show (XenoPitchExpression x) = show x
  show (VantagePointExpression x) = show x

-- Temporal values is short for TemporalRelationship and Aural is short for Aural
Values. Polytemporal stands for TempoRelationship, Rhythmic stands shor for Rhythmic
values

type Voices = Map String Voice

-- aural:: List Value // Temporal, which  type has:: Polytemporal Rhythmic Loop
data Voice = Voice Temporal (List Aural)

instance voiceShow :: Show Voice where
```

```
    show (Voice t a) = show t <> " " <> show a

type Aural = List Value -- aural is a list of aural attributes for a given time
layer. tend to be 1 sound, 1 n, 1 gain, 1 pan, etc.

-- :: List Aural is a non-monophonic time layer. Each event will trigger multiple
samples with different aural attributes

type VantageMap = Map String DateTime

data Vantage = Build TimePoint | Move (Either Rational Rational) | Remove

instance vantageShow :: Show Vantage where
  show (Build x) = "established " <> show x
  show (Move num) = " moved by " <> show x -- fornow secs but enable xBeats
      where x = case num of
                  (Left beat) -> show beat <> " beats"
                  (Right secs) -> show secs <> " secs"
  show Remove = " removed"

data TimePoint = Beat Rational | Secs Rational | UTC DateTime

instance timePoint :: Show TimePoint where
  show (Beat beat) = show beat <> " beats from eval"
  show (Secs secs) = show secs <> " secs from eval"
  show (UTC utc) = show utc

-- future additions to Value: OSound | OTransposedSound | Full Sound OSound
-- for now only X generates sounds, O should be allowed to invoke sound as well. Full
will allow to invoke sound for X and O as pairs

data Variation a = Every Int Span (List a)

instance showVariation :: Show a => Show (Variation a) where
  show :: Variation a -> String
  show (Every n sp xs) = "every " <> show n <> " " <> show sp <> " " <> show xs

data Value =
  Sound Span (List String) (List (Variation String)) | TransposedSound String Int |
  N Span (List Int) (List (Variation Int)) | TransposedN String Int |
  Gain Span (List Number) (List (Variation Number)) | TransposedGain String Int |
  Pan Span (List Number) (List (Variation Number)) | TransposedPan String Int |
  Speed Span (List Number) (List (Variation Number)) | TransposedSpeed String Int |
  Begin Span (List Number) (List (Variation Number)) | TransposedBegin String Int |
  End Span (List Number) (List (Variation Number)) | TransposedEnd String Int |
  Vowel Span (List String) (List (Variation String)) | TransposedVowel String Int |
  CutOff Span (List Number) (List (Variation Number)) | TransposedCutOff String Int |
  CutOffH Span (List Number) (List (Variation Number)) | TransposedCutOffH String Int
|
  MaxW Span (List Number) (List (Variation Number)) | TransposedMaxW String Int |
  MinW Span (List Number) (List (Variation Number)) | TransposedMinW String Int |
  Inter Span (List Number) (List (Variation Number)) | TransposedInter String Int |
```

```
  Legato Span (List Number) (List (Variation Number)) | TransposedLegato String Int |
  Orbit Span (List Int) (List (Variation Int)) | TransposedOrbit String Int |
  Dastgah Span Dastgah | Xeno (Tuple String (Maybe Int)) Span (List Int) |
  Prog Span (List (Tuple String (Maybe Int))) | XNotes Span (List Int) (List
(Variation Int)) | TransposedPitch String Int

instance valueShow :: Show Value where
  -- show (Soundy sp xs every) = show sp <> " " <> show xs <> show " " <> show every
  show (Sound x l v) = show x <> " " <> show l <> " " <> show v
  show (TransposedSound voice n) = "s transposed from " <> voice
  show (N x l v) = show x <> " " <> show l <> " " <> show v
  show (TransposedN voice n) = "n transposed from " <> voice
  -- show (TransposedNWith voice n l) = show l <> "n transposedWith from " <> voice
  show (Gain x l v) = show x <> " " <> show l
  show (TransposedGain voice n) = "gain transposed from " <> voice
  -- show (TransposedGainWith voice n l) = "gain transposedWith from " <> voice
  show (Pan x l v) = show x <> " " <> show l
  show (TransposedPan voice n) = "pan transposed from " <> voice
  -- show (TransposedPanWith voice n l) = "pan transposedWith from " <> voice
  show (Speed x l v) = show x <> " " <> show l
  show (TransposedSpeed voice n) = "speed transposed from " <> voice
  -- show (TransposedSpeedWith voice n l) = "speed transposedWith from " <> voice
  show (Begin x l v) = show x <> " " <> show l
  show (TransposedBegin voice n) = "begin transposed from " <> voice
  -- show (TransposedBeginWith voice n l) = "begin transposedWith from " <> voice
  show (End x l v) = show x <> " " <> show l
  show (TransposedEnd voice n) = "end transposed from " <> voice
  -- show (TransposedEndWith voice n l) = "end transposedWith from " <> voice
  show (Vowel x l v) = show x <> " " <> show l
  show (TransposedVowel voice n) = "vowel transposed from " <> voice
  show (CutOff x l v) = show x <> " " <> show l
  show (TransposedCutOff voice n) = "cutoff transposed from " <> voice
  show (CutOffH x l v) = show x <> " " <> show l
  show (TransposedCutOffH voice n) = "hcutoff transposed from " <> voice
  show (MaxW x l v) = show x <> " " <> show l
  show (TransposedMaxW voice n) = "maxw transposed from " <> voice
  show (MinW x l v) = show x <> " " <> show l
  show (TransposedMinW voice n) = "minw transposed from " <> voice
  show (Inter x l v) = show x <> " " <> show l
  show (TransposedInter voice n) = "w interpolation transposed from " <> voice
  show (Legato x l v) = show x <> " " <> show l
  show (TransposedLegato voice n) = "legato transposed from " <> voice
  show (Orbit x l v) = show x <> " " <> show l
  show (TransposedOrbit voice n) = "orbit transposed from " <> voice
  show (Dastgah span d) = show d
  show (Xeno id span l) = show l
  show (Prog span l) = "prog" <> show l
  show (XNotes span l v) = "xnotes " <> show l
  show (TransposedPitch voice n) = "pitch transposed from " <> voice

data Span = CycleEvent | CycleBlock | CycleInBlock | SpreadBlock -- | Weight
```

```
instance spanShow :: Show Span where
  show CycleEvent =    "_"
  show CycleBlock =    "_-"
  show CycleInBlock =  "-_"
  show SpreadBlock =   "_-_"
  -- show BySubdivision = "-"
  -- show Weight = "-_-"

data Dastgah = Shur (List Int) -- 1 to 8 then it cycles back

instance showDatsgah :: Show Dastgah where
  show (Shur l) = "shur " <> show l

data Temporal = Temporal Polytemporal Rhythmic Boolean | Replica String -- this will
require a check and the recursive implementation now very familiar

instance temporalShow :: Show Temporal where
    show (Temporal x y z) = show x <> " " <> show y <> (if z then " looped" else "
unlooped")
    show (Replica id) = "replicated from " <> show id

data Polytemporal =
  Kairos Number TempoMark | -- last arg is tempo -- Arg: universal time unit
(miliseconds and datetime in purs)
  -- Kairos starts a program at evaluation time (or as soon as possible), no
underlying grid
  Metric ConvergeTo ConvergeFrom TempoMark | -- starts a program attached to a
default underlying voice (a tempo grid basically) first number is the point to where
the new voice will converge, second number is the point from which it converges.
  Converge String ConvergeTo ConvergeFrom TempoMark | -- Args: String is the voice
identifier, convergAt (where this voice converges with the identified voice) and
convergedFrom (the point of this voice that converges with the identified voice)  --
Converge starts a program in relationship with another voice
  Novus String ConvergeFrom TempoMark -- |
  -- InACan (List Polytemporal)

instance polytemporalShowInstance :: Show Polytemporal where
  show (Kairos asap t) = "kairos: " <> show asap <> " tempo mark: " <> show t
  show (Metric cTo cFrom t) = "(cTo "<>show cTo<>") (cFrom "<>show cFrom <> ") (tempo
mark: " <> show t <> ")"
  show (Converge voice cTo cFrom t) = "toVoice "<>show voice<>" (cTo "<>show cTo<>")
(cFrom "<>show cFrom <> ") (tempo mark: " <> show t <> ")"
  show (Novus vantageId cFrom t) = "vantagePoint "<>show vantageId<>" (cFrom "<>show
cFrom <> ") (tempo mark: " <> show t <> ")"
  -- show (InACan xs) = "InACan "<> show xs

data Rhythmic =  -- whenPosix, thats it
  X | -- x
  O |
  Sd Rhythmic | -- [x]
  Repeat Rhythmic Int |
  Bjorklund Euclidean Int Int Int |
```

```
  Rhythmics (List Rhythmic) -- xoxo

instance Show Rhythmic where
  show X = "x"
  show O = "o"
  show (Sd xs) = "[" <> show xs <> "]"
  show (Repeat xs n) = "!" <> show xs <> "#" <> show n
  show (Bjorklund eu k n r) = "("<>show k<>","<>show n<>") "<>show eu
  show (Rhythmics xs) = show xs

data Euclidean = Full Rhythmic Rhythmic | K Rhythmic | InvK Rhythmic | Simple -- add
simple inverse

instance euclideanShowInstance :: Show Euclidean where
  show (Full x y) = "full: " <> (show x) <> " " <> (show y)
  show (K x) = show x
  show (InvK x) = show x
  show (Simple) = "simple"

-- CPAlign will provide a convergence point in relation to a part of the program.
-- mod 4 will align a cp with the next voice start multiple of 4. The convergenceTo
value with 'mod 4' will converge to the other voice at the next voice muliplte of 4.
If this would be the convergenceFrom, the voice will align to the other voice from
its next voice multiple of 4.

-- data Align = Mod Number Number | Mod' Number | Snap Number | Snap' Number | Origin
Number  -- this is the goal

data CPAlign = Mod Int | Snap | Origin  -- this is the first stage

instance Show CPAlign where
  show (Mod m) = "cp after first multiple of " <> show m <> " ahead"
  show  Snap = "closest to eval"
  show  Origin = "diverge at origin"

-- Aligners:
---- Mod Multiple Offset (next start of voice/event multiple of N with an offset
number becomes voice 0)
---- Mod' Multiple Offset (closest multiple, can be in the past already)
---- Snap cp happens at closest voice or event.
---- Origin will align the cp at 0 (1st of January, 1970: 12:00 am)

data ConvergeFrom = Structure Int (Array Int) | Process Int | Percen Number | Last

instance Show ConvergeFrom where
  show (Structure x xs) = show x <>"-"<> result <> " "
      where subdivisions = foldl (<>) "" $ map (\x -> show x <> ".") xs
            result = Str.take (Str.length subdivisions - 1) subdivisions
  show (Process e) = show e
  show (Percen p) = show p <> "%"
  show Last = "last"
```

```
data ConvergeTo = StructureTo Int (Array Int) CPAlign | ProcessTo Int CPAlign |
PercenTo Number CPAlign | LastTo CPAlign

instance Show ConvergeTo where
  show (StructureTo x xs a) = show x <>"-"<> result <> " " <> show a
      where subdivisions = foldl (<>) "" $ map (\x -> show x <> ".") xs
            result = Str.take (Str.length subdivisions - 1) subdivisions
  show (ProcessTo e a) = show e <> " " <> show a
  show (PercenTo p a) = show p <> "% " <> show a
  show (LastTo a) = "last"

-- perhaps this is the output of processTempoMark, this will allow users to declare a
total duration of a block (reverting more or less the additive logic to divisive)

-- data TimeSignature = Duration Rational | TM TempoMark | Sin Sinusoidal
-- type Sinusoidal = {tempoMark:: TempoMark, freq:: Rational, amp:: Rational}

data TempoMark = XTempo | CPM Rational | BPM Rational Rational | CPS Rational | Prop
String Int Int | Sin Sinusoidal | Dur Rational

instance Show TempoMark where
  show XTempo = "external"
  show (CPM cpm) = show cpm <> "cpm"
  show (BPM bpm figure) = show bpm <> "bpm the " <> show figure
  show (CPS cps) = show cps <> "cps"
  show (Prop id x y) = "from voice: " <> id <> " " <> show x <> ":" <> show y
  show (Sin acc) = show acc
  show (Dur n) = "dur " <> show n

type Sinusoidal = {
  min:: TempoMark,
  max:: TempoMark,
  osc:: Rational,
  phase:: Rational
}

type TimePacket = {
  ws:: DateTime,
  we:: DateTime,
  eval:: DateTime,
  origin:: DateTime,
  tempo:: Tempo,
  vantageMap:: VantageMap
}

data Event = Event Onset Index

instance Show Event where
    show (Event o i) =  show o <> " " <> show i

showEventIndex (Index _ _ n) = show n
```

```
showStructureIndex (Index x xs _) = show x <>"-"<> result
      where subdivisions = foldl (<>) "" $ map (\x -> show x <> ".") xs
            result = Str.take (Str.length subdivisions - 1) subdivisions

data Onset = Onset Boolean Number

instance Show Onset where
    show (Onset true n) =  "(X" <> " psx:" <>(Str.drop 0 $ show n) <>")"
    show (Onset false n) = "(O" <> " psx:" <>(Str.drop 0 $ show n) <>")"

instance Ord Onset where
    compare (Onset bool1 pos1) (Onset bool2 pos2) = pos1 `compare` pos2

instance Eq Onset where
    eq (Onset bool1 pos1) (Onset bool2 pos2) = pos1 == pos2

data Index = Index Int (Array Int) Int

instance indexShow :: Show Index where
    show (Index x xs n) = show x <>"-"<> result <> " (" <> (Str.take 8 $ show n) <>
")"
      where subdivisions = foldl (<>) "" $ map (\x -> show x <> ".") xs
            result = Str.take (Str.length subdivisions - 1) subdivisions

-- Dastgah

type DastgahNote = {
  function:: String,
  movement:: Interval,
  midiInterval:: Number
}


-- xenopPitch

data Subset = Subset Int | Unions (Array Int) | Intersection Int Int | Difference Int
Int | Nested Subset

instance subsetShow :: Show Subset where
  show _ = "subset"

data XenoPitch = CPSet Int (Array Int) (Maybe (Array Subset)) | MOS Int Int | EDO
Number Int | ShurNot8 | ShurNot

instance xenoShow :: Show XenoPitch where
    show (CPSet s f subs) = "cps " <> show s <> " " <> show f <> " " <> show subs
    show (MOS k n) = "mos " <> show k <> " " <> show n
    show (EDO p d) = "edo " <> show p <> " " <> show d
    show ShurNot8 = "ShurNot8"
    show ShurNot = "ShurNot"

type XenoNote = {
```

```
    set:: Array Int,
    "archi-set":: Array String,
    ratio:: Int,
    "bounded-ratio":: Number,
    "bounding-period":: Int
}

data Interval = UpJump | UpNext | DownJump | DownNext | Unison

instance intervalShow :: Show Interval where
  show UpJump = "UpJump"
  show UpNext = "UpNext"
  show DownJump = "DownJump"
  show DownNext = "DownNext"
  show Unison = "Unison"
```