Distributed Music Systems

Designing Web-Based Tools for Artistic Research and Practice

Aliénor Golvet

Thèse de doctorat de Sorbonne Université

ED130 EDITE - École Doctorale Informatique Télécommunication Électronique

UMR9912 STMS - Sciences et Technologies de la Musique et du Son
IRCAM · CNRS · Sorbonne Université

Equipe ISMM - Interaction Son Musique Mouvement

Composition du jury

Michel BUFFA *Université Côte d'Azur*	**Rapporteur**
Anna XAMBÓ *Queen Mary University of London*	**Rapportrice**
Alain BONARDI *Université Paris 8*	**Examinateur**
Ariane STOLFI *Universidade Federal do Sul da Bahia*	**Examinatrice**
Frédéric BEVILACQUA *IRCAM*	**Directeur**
Benjamin MATUSZEWSKI *IRCAM*	**Co-encadrant**

Abstract

The main object of study of this thesis are Co-Located Distributed Music Systems which can be described as systems of heterogeneous devices interconnected via a network able to exchange data and information and to produce sound for a musical purpose. We specifically study co-located systems in which all interaction with such systems occurs in a shared physical space. As such, this thesis is placed in the direct legacy of a body of works that addressed the technical implementation and the artistic potential of computer networks for music creation. More especially, it builds on the conceptual and technical framework implemented in previous research projects on collective musical interaction mediated by web and mobile technologies.

First, on the technical side, we contributed to the integration of embedded devices into such distributed systems. Second, we considered the use of distributed systems by "expert users" such as researchers, music composers and performers. Our work employs an interdisciplinary and project-grounded design approach to investigate 1) the potentials and affordances provided by such distributed systems in the ecosystem of musical creation, 2) the material ecosystem of technologies that is needed to support it, 3) the methodologies usable for its design and 4) its appropriation by expert users and its integration into already existing musical practices with their established practices, including software and hardware.

These questions are explored through a variety of projects that employ different design methodologies and perspectives, and involve collaborations with researchers and artists at various points of the design process. These projects include A^3PM, an application designed for supporting an experimental methodology in empirical musicology and used in various research projects; *Koryphaîos*, a software for composing music for distributed ensembles of devices integrated in the *Max/MSP* environment that we conceived through a collaboration with composer Luciano L. Barbosa; *Simone* a distributed instrument for collective improvisation which we used in experimental workshops to study its appropriation by groups of users and the forms of networked interaction that emerged; *Simone Solo* an instrument that uses distributed sound sources and controlled by a single instrumentalist that we conceived through a long-term collaboration with artist Jean-Brice Godet; and the creation by the author of *Quasimodots*, a musical piece for 40 Raspberry Pi computers equipped with microphones. Through these projects we observed the emergence of novel work practices in research and artistic contexts, novel methods of composition or performance and novel types of collective interaction.

Résumé

L'objet d'étude principal de cette thèse sont les systèmes musicaux distribués co-localisés qui peuvent être décrits comme des systèmes composés d'appareils hétérogènes interconnectés via un réseau et capables d'échanger des données et des informations pour produire du son à des fins musicales. Nous étudions spécifiquement les systèmes dits co-localisés dans lesquels les interactions avec ces systèmes ont lieu dans un même espace physique. En tant que telle, cette thèse s'inscrit dans l'héritage direct d'un ensemble de travaux qui ont abordé la mise en œuvre technique et le potentiel artistique des réseaux informatiques pour la création musicale. Plus particulièrement, elle s'appuie sur le cadre conceptuel et technique mis en œuvre dans des projets de recherche antérieurs sur l'interaction musicale collective médiatisée par les technologies web et mobiles.

Tout d'abord, sur le plan technique, nous avons contribué à l'intégration de dispositifs embarqués dans de tels systèmes distribués. Deuxièmement, nous avons envisagé l'utilisation de systèmes distribués par des « utilisateurs experts » tels que des chercheurs, des compositeurs et des musiciens. Notre travail se base sur une approche de design de type recherche-projet et interdisciplinaire pour étudier 1) les affordances et les possibilités offertes par ces systèmes distribués dans l'écosystème de la création musicale, 2) l'écosystème matériel des technologies nécessaires pour le soutenir, 3) les méthodologies nécessaires pour leurs conception et 4) son appropriation par des utilisateurs experts et son intégration dans des pratiques musicales déjà existantes avec leurs pratiques (logiciels et matériel compris) déjà établies.

Ces questions sont explorées à travers une variété de projets qui utilisent différentes méthodologies et perspectives de design, et impliquent des collaborations avec des chercheurs et des artistes à différents stades du processus de conception. Ces projets comprennent A^3PM, une application conçue pour soutenir une méthodologie expérimentale en musicologie empirique et utilisée dans divers projets de recherche ; *Koryphaîos*, un logiciel de composition musicale pour ensembles distribués d'appareils intégré dans l'environnement *Max/MSP* que nous avons conçu en collaboration avec le compositeur Luciano L. Barbosa ; *Simone* un instrument distribué pour l'improvisation collective que nous avons utilisé dans des ateliers expérimentaux pour étudier son appropriation par des groupes d'utilisateur et les formes d'interaction en réseau qui en ont émergé; *Simone Solo* un instrument qui utilise des sources sonores distribuées et contrôlées par un seul instrumentiste que nous avons conçu dans le cadre d'une collaboration à long terme avec l'artiste Jean-Brice Godet ; et la création par l'autrice de *Quasimodots*, une pièce musicale pour 40 ordinateurs Raspberry Pi équipés de microphones. À travers ces projets, nous avons observé l'émergence de nouvelles pratiques de travail dans des contextes de recherche et artistiques, de nouvelles méthodes de composition ou de performance et de nouveaux types d'interaction collective.

Contents

List of Figures

List of Tables

Introduction

0.1 Context

The use of computer networks in a musical context, initiated in 1978 by the *League of Automatic Composers*, has given rise to numerous technical and artistic developments that have in their own way transformed the way music is created. These developments range from the simple use of communication protocols (such as Musical Instrument Digital Interface (MIDI)[1] or Open Source Control (OSC)[2]) and synchronization protocols (such as Ableton Link[1] [3]) between musical devices, to unique network-based arrangements such as laptop orchestras[4].

[1]: (1996), *The Complete MIDI 1.0 Detailed Spec*

[2]: Wright et al. (1997), 'Open SoundControl: A New Protocol for Communicating with Sound Synthesizers'

1: http://ableton.github.io/link/

[3]: Goltz (2018), 'Ableton Link – A Technology to Synchronize Music Software'

[4]: Gresham-Lancaster (2017), 'A Personal Reminiscence on the Roots of Computer Network Music'

Outside the music world, the introduction of computer networks into work processes, the development of new communication protocols and the Internet, and the proliferation of new types of devices such as smartphones and nanocomputers have led to the development of ubiquitous computing, first described by Weiser as the vanishing of computer technologies in our environment through networks of "hundreds of devices"[5].

More recently, the specification of the Web Audio API (Application Programming Interface) by the World Wide Web Consortium (W3C) in 2011[2] and its implementation in the majority of web browsers have made web technologies as a viable platform for developing musical applications and to benefit quite naturally from the integration of network technologies in these applications. In the field of research, communities interested in the technical, practical and artistic development of these technologies have gathered in conferences such as the Web Audio Conference since 2015 and the IEEE International Symposium on the Internet of Sounds since 2020. Various examples of applications belonging to this field are : remote musical performance via videoconferencing tools, virtual reality musical applications, audience participation in a musical performance, etc...

2: `https://www.w3.org/TR/webaudio/`

0.2 Motivation and Aims

In this general context, our work focuses on the particular case of what we call *Co-Located Distributed Music Systems*.

In the field of computer science, the term "distributed system" describes a decentralized system in which computing is performed on multiple computers connected via a network and able to communicate messages and data between each other. Following Van Steen and Tanenbaum[6], the different elements of the distributed system should "appear to its users as a single coherent system". Distributed Music Systems thus refers to networks of heterogeneous devices interconnected via a network and able to exchange data and information and to produce sound for a musical purpose (cf Figure 0.1). These devices may be autonomous or interacting with human actors or their surrounding environments. The term "co-located" (or situated) refers to the fact that all interactions occur in a shared physical space. We use the concept of Co-Located Distributed Music Systems to put the emphasis on the system from a holistic perspective including all its actors (users and devices) and their resulting social collective interaction, and the creative and artistic potentials it raises.

[6]: Van Steen et al. (2016), 'A Brief Introduction to Distributed Systems'

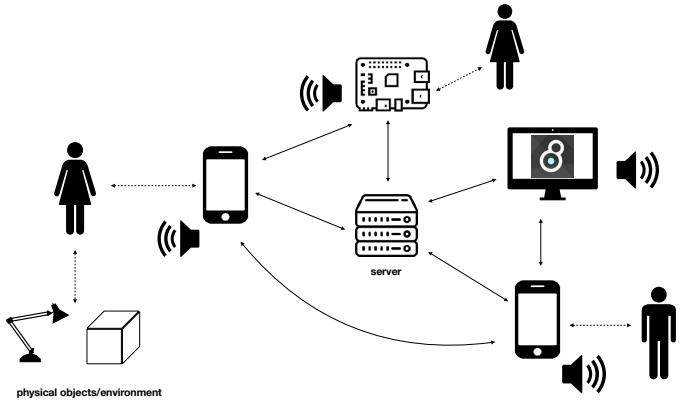


Figure 0.1: Distributed Music Systems.

This thesis takes place in the direct legacy of a previous research project called *CoSiMa* (Collective Situated Media) coordinated by Ircam which aimed to develop a platform for co-located collective interaction based on web and mobile technologies. Our approach draws largely from the technical infrastructure elaborated in this project and from its conceptual framework drawing from Latour's Actor-Network Theory (ANT)[7], which highlights the entanglement of interactions in networks composed of an "heterogeneous

[7]: Latour (1996), 'On Actor-Network Theory. A Few Clarifications, Plus More Than a Few Complications'

and evolving mix of persons, objects and technologies, instruments and mediation techniques"[8].

[8]: Bevilacqua et al. (2021), 'On Designing, Composing and Performing Networked Collective Interactions'

However, our approach brings several new contributions to this legacy by expanding the range of devices used in our distributed systems beyond smartphones (and thus increasing the concern for systems to support heterogeneity and interoperability). We also diverge by focusing less on collective interaction and audience participation and more on the use of distributed systems by expert users such as researchers, music composers and performers. Working with this category of users is beneficial as they provide unique insights to the design process as highlighted by Wendy Mackay: "Artists and scientists are my favorite research participants because they are endlessly creative. They push the limits of technology, discover interesting innovations, but also generate weird problems. They look at things from a different angle and challenge our traditional ways of thinking."[9].

[9]: Wanderley et al. (2019), 'HCI, Music and Art'

The main research question that this thesis seeks to answer is then the following: What does distributed music systems do to artistic research and musical practice ecologies?

Indeed, beyond the mere design of these distributed musical systems, our approach seeks to question how these kinds of apparatus can be integrated into already existing musical practices with their already well-established habits, software and hardware. Rodger *et al.*[10] have highlighted the problem of adopting a functional and problem-solving approach in the design of musical instruments. They argue that musical instruments should be considered as "a constellation of *processes* (affordances) which may be shared with other instruments, and which may change over time". They also advocate for an ecological approach that takes into account the socio-cultural environment in which musicians are acting.

[10]: Rodger et al. (2020), 'What Makes a Good Musical Instrument? A Matter of Processes, Ecologies and Specificities'

From then, in this thesis, the object of study and observation is not the "distributed music system" as a theoretical and immaterial concept but rather the study of the potentials and affordances it gives rise to in the ecosystem of musical creation; of the material ecosystem of technologies that is needed to support it; of the design methodologies it is preferable to use for its conception; of its appropriation by communities of users.

The question of appropriation, used by Dourish to "refer to the ways in which people adopt and adapt interactive technologies, fitting them into working practices and evolving those practices around them"[11] is of prime importance in this work. Throughout these thesis, we employ multiple design methodologies that aim to support user appropriation of our tools and we aim to document how users appropriate them through experimental setups and long-term observation of their use out of the laboratory.

[11]: Dourish (2003), 'The Appropriation of Interactive Technologies'

Indeed, by observing a number of related works, it appeared to us that most projects implementing distributed music systems were conceived with a restricted scope, even limited to personal or one-off use. On the contrary, to varying degrees, the projects we have implemented are conceived as authoring tools, leaving their users as much freedom as possible to create their own application, installation or performance and to integrate them within their work ecosystems.

As a secondary objective of this research project, we aimed to investigate environmental interactions in Co-Located Distributed Music Systems through the addition of microphones and sensors. This is a common thread that influenced the creation of several of the research projects that constitutes the latter part of this thesis.

0.3 Methods

This thesis presents a variety of projects that seek to shed light on this question from their own perspective. In that sense, our approach echoes Rohrhuber's remark on the structure of a musical network: "Including the observer as an active participant into the system, the relational structure of a social and musical network cannot be based on an absolute reference system, but results in multiple points of observation"[12].

[12]: Rohrhuber (2007), 'Network Music'

The general methodology of this thesis is therefore grounded in "practice-based design research"[13], sometimes also called "research through design" or "project-grounded design" [14] in which knowledge emerge through the design process of an artefact or system. We also base our approach on interdisciplinarity, following Findeli's remark that "an ideal design research question would thus be one that uncovers and emphasizes the complex

[13]: Gaver et al. (2022), 'Emergence as a Feature of Practice-based Design Research'

[14]: Findeli et al. (2008), 'Research through Design and Transciplinarity: A Tentative Contribution to the Methodology of Design Research'

interdisciplinarity of the specific anthropological experience that is at stake in a design question"[15].

[15]: Findeli (2012), 'Searching for Design Research Questions: Some Conceptual Clarifications'

Throughout this thesis we therefore engaged in collaborations with various "expert" users of our systems and we have chosen to focus on three fields of practice into which the systems we have designed have been integrated: empirical musicology, computer-assisted composition and improvised musical creation. The choice of these fields of application, or rather of materialization of this thesis stems naturally from the research and creative environment in which this thesis took place (Ircam), the various people we met during its development, and the personal affinities of the author of this thesis.

In the context of Human Computer Interaction (HCI) research for dance, Fdili Alaoui[16] has highlighted the need to move beyond problem-solving and techno-solutionist approaches that considers the design of novel technology that solves an identified problem as a contribution and that seeks to achieve a level of generalizability. Instead, she argues that "generalizability, beyond not being achievable, is actually not desirable" and encourages "alternative approaches allowing artists that are experimenting with technologies to embrace the messiness of their practice in order to contribute to the field with their own methods, insights and voices"[17]. We embrace this vision and we aim to present an account of our approach as thoroughly experimental, prototypal and in contact with artists and real-life practice. We also value the hybridization of roles and the artists and researchers' insights within the design process and aim to bring users of our systems to the role of co-developers.

[16]: Fdili Alaoui (2023), 'Dance-Led Research'

[17]: Fdili Alaoui (2019), 'Making an Interactive Dance Piece'

For each of the projects presented in this manuscript, we employ a different methodology and design perspective that we identified as better suited to the specific context of the project. This includes third-person perspectives and user-centered design, second-person perspectives with participatory design and long-term collaborations with artists, and first-person perspectives with research-creation methodologies.

From the plurality of methodologies and disciplines employed, this thesis can be considered in the epistemological framework of *experimental systems* described by Rheinberger[18]. Experimental systems describe the research process and environment as

[18]: Rheinberger (2011), 'Consistency from the Perspective of an Experimental Systems Approach to the Sciences and Their Epistemic Objects'

"dynamic, materially conditioned research bodies" that evolve and generate knowledge through unpredictable and uncontrolled movement. It also describes the research process as an assemblage of diverse and heterogeneous elements including the research objects themselves, the technical elements that exist to observe them or develop them, the theoretical frameworks that describe them and the general socio-cultural contexts in which research takes place. This thesis therefore aims to address the variety of these elements and the often unpredictable and non-linear fashion in which they interact to produce knowledge, observations and design artefacts.

0.4 Contributions

A summary of this thesis' contributions detailed in the following chapters is presented in Figure 0.2. These contributions, each represented by a specific Co-Located Distributed Music System we have developed (aside from the set of "core technologies" at the top") are arranged on two axes. The vertical axis represents the design perspective we have taken in the design process ranging from third-person perspective to first-person perspective. The horizontal axis arranges these projects depending on whether they were designed as experimental tools or part of experimental setups or as tools for artistic creation or performance. The contributions of this thesis are therefore of different types including technological development, empirical observations, methodological approaches, and musical creation.

In Chapter 1, we present a background on various elements that have been mobilized in this research work in the fields of art, design and computer science. First, we provide a non-exhaustive historical account on the development of "computer network music" and "ubiquitous computing". We also provide some background on epistemological and methodological concepts employed in this thesis.

Chapter 2 presents the ecosystem of low-level technological components that are used throughout this thesis including the *Web Audio API* and the *soundworks* framework. Our contributions include the implementation of new web components, the proposition of new use cases, the technical evaluation and improvement of

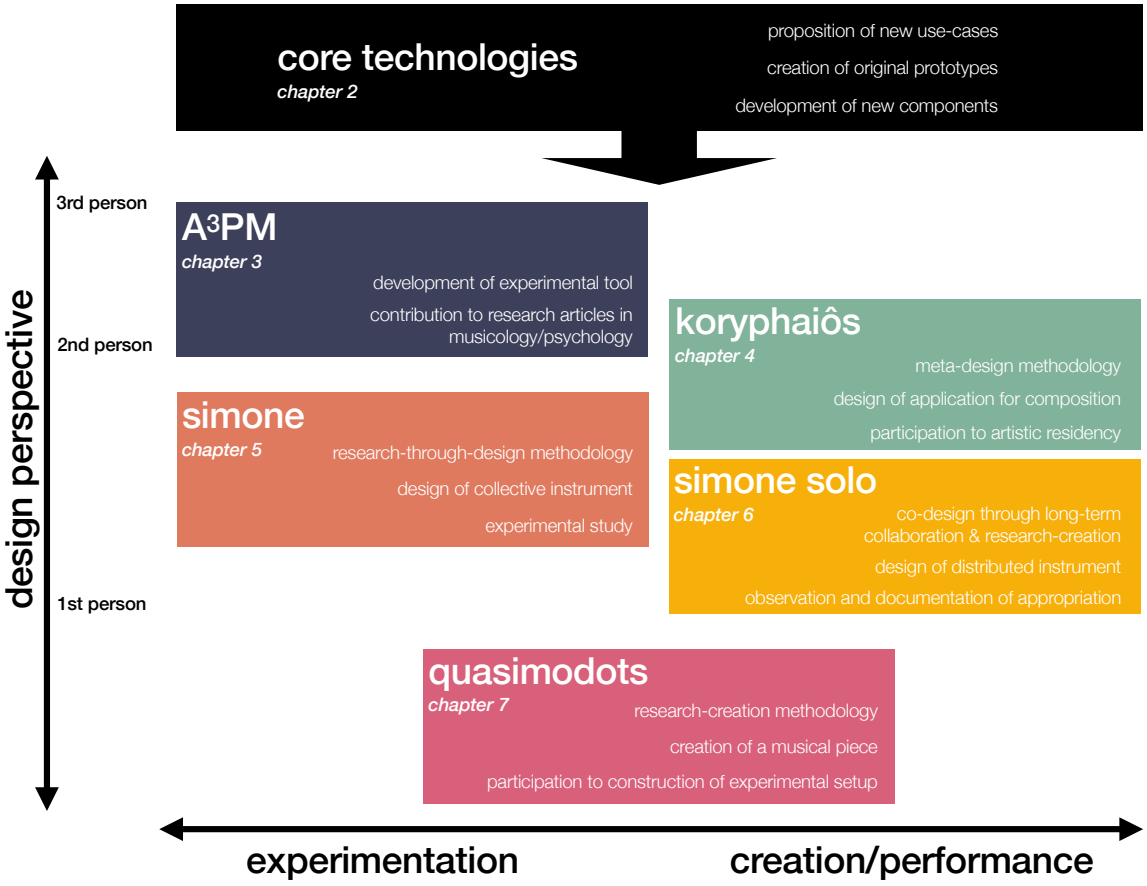


Figure 0.2: Diagram summarizing this thesis' contributions.

these technologies including the use of microphones. To provide an example, we also present a prototype of an application dedicated to the development of audio effects on a network of devices developed with Benjamin Matuszewski.

Chapter 3 presents our participation in the development of *A³PM*, a distributed application dedicated to the methodology of self-annotation of musical performance in empirical musicology. We detail the design and implementation of this application. We also present some of the use cases of *A³PM* in research projects in musicology and psychology to which we've contributed.

Chapter 4 presents *Koryphaîos* an application that enables musical composition for distributed ensembles of devices in the *Max/MSP* environment. We detail the design of *Koryphaîos* created through a Meta-Design methodology by a collaboration with composer Luciano Leite Barbosa.

Chapter 5 presents *Simone*, a distributed musical instrument

for collective improvisation created through a research-through-design methodology. First, the design and implementation of *Simone* is detailed. In a second part, we present experimental workshops in which we have invited groups of participants to improvise collectively with the system. Through the analysis of interviews with participants and quantitative results, results highlight the process of appropriation of the instrument and the nature of collective interaction with a network instrument. We use these results to reflect on the design of distributed instruments.

Chapter 6 presents *Simone Solo*. Originating from *Simone*, *Simone Solo* is an instrument consisting of distributed sound sources controlled by a single instrumentalist through a web interface. We first detail the design of *Simone Solo* and then present a long-term collaboration with artist Jean-Brice Godet through which we documented his process of appropriation of the system thus enabling us to confront our instrument to real-life musical practice.

Chapter 7 reports on the creation of *Quasimodots* a musical piece for an ensemble of 40 Raspberry Pi devices equipped with microphones and speakers inspired by Alvin Lucier. Through a research-creation methodology and a first-person perspective, we present the creation process of the piece and the experience of rehearsing and performing it.

Finally, we have published online a companion website to this manuscript. This companion website showcases various audiovisual elements that provides a better insight on elements discussed in this manuscript. The index of this companion website is available at the following address: `https://alienorgolvet.com/thesis`

0.5 Acknowledgments

The work described in this thesis has received partial support from the DOTS research project funded by the French National Research Agency (ANR-22-CE33-0013-01).

0.6 Publications

This thesis led to multiple publications and presentations in conferences.

0.6.1 Journal papers

- Aliénor Golvet, Louise Goupil, Pierre Saint-Germier, Benjamin Matuszewski, Gérard Assayag, Jérôme Nika, and Clément Canonne. 'With, against, or without? Familiarity and Copresence Increase Interactional Dissensus and Relational Plasticity in Freely Improvising Duos.' In: *Psychology of Aesthetics, Creativity, and the Arts* 18.2 (Sept. 2021). (Visited on 09/29/2022)
- Emanuelle Majeau-Bettez, Aliénor Golvet, and Clément Canonne. 'Tracking Auditory Attention in Group Performances: A Case Study on Éliane Radigue's *Occam Delta XV*'. In: *Musicae Scientiae* (Oct. 2023). (Visited on 10/26/2023)
- Aliénor Golvet, Benjamin Matuszewski, and Frédéric Bevilacqua. 'SIMONE : Un Instrument Distribué Pour l'improvisation Musicale'. In: *Revue Francophone Informatique et Musique* 10 (2024). (Visited on 09/24/2024)

0.6.2 Conference Papers

- Aliénor Golvet, Luciano L. Barbosa, Etienne Démoulin, and Benjamin Matuszewski. 'Koryphaîos A Patchworked Compositional Environment for Distributed Music Systems'. In: *Web Audio Conference*. Cannes, France, 2022. (Visited on 09/07/2022)
- Aliénor Golvet, Benjamin Matuszewski, and Frederic Bevilacqua. 'Simone : Un Instrument Distribué Pour l'étude Des Interactions Improvisées Collectives'. In: *Journées d'Informatique Musicale (JIM 2023)*. Saint-Denis, France, 2023 - Best Young Researcher Paper Award
- Benjamin Matuszewski and Aliénor Golvet. 'Rapid Prototyping of Distributed Musical Things Using Web Technologies'. In: *2023 4th International Symposium on the Internet of Sounds*. Pisa, Italy: IEEE, Oct. 2023. ISBN: 9798350382549. (Visited on 02/06/2024)

- Aliénor Golvet, Benjamin Matuszewski, and Frederic Bevilacqua. 'Designing a Distributed Musical Instrument for Collective Improvised Interaction'. In: *Audio Mostly 2024 - Explorations in Sonic Cultures*. Milan Italy: ACM, Sept. 2024. ISBN: 9798400709685. (Visited on 09/24/2024)

0.6.3 Other communications

Presentation at "1ère journée jeunes chercheur·e·s Recherches Culturelles Participatives", 2022, Paris, France.

0.6.4 Other

Member of the organising committee of the JJCAAS 2023 conference[3] (Journées Jeunes Chercheur·se·s en Audition, Acoustique musicale et Signal audio)

3: https://jjcaas2023.sciencesconf.org/

1 Background

This chapter presents a background on various elements that have been mobilized in this thesis. The first part focuses on a historical account of the development of "computer network music". The second part presents various epistemological and methodological concepts used in this thesis.

1.1 Networks and Music

In this section we present a selective historical account of the developments of computer network music in its various dimensions from its early steps as a successor to radio and telephone art of the early 20th century to the recent waves of collective experiences driven by the *Web Audio API*. Because of the sheer number of historical examples, we chose to focus on works with a particular historical relevance with respect to the rest of the field and on works that were influential to this thesis.

Throughout this section we refer to a companion web page containing audios and videos that may help you get a better perspective of the work mentioned. This page is accessible at the following address: `https://alienorgolvet.com/thesis/chapter-background.html`.

1.1.1 The Birth of Computer Network Music

In 1978, the foundation of the *League of Automatic Music Composers* by Jim Horton, Rich Gold, John Bischoff and David Behrman in California marked the inception of the field of "computer network music". The members of the group, each one playing with a computer program they had personally developed, were linked together as a way to exchange data and interconnect processes (cf Figure 1.1). *The League*. Through this experiment, they were one of the firsts to go against the prevailing practice of the computer music of the era to control every parameter of sound through programmation and left room for decentralization of sound production.

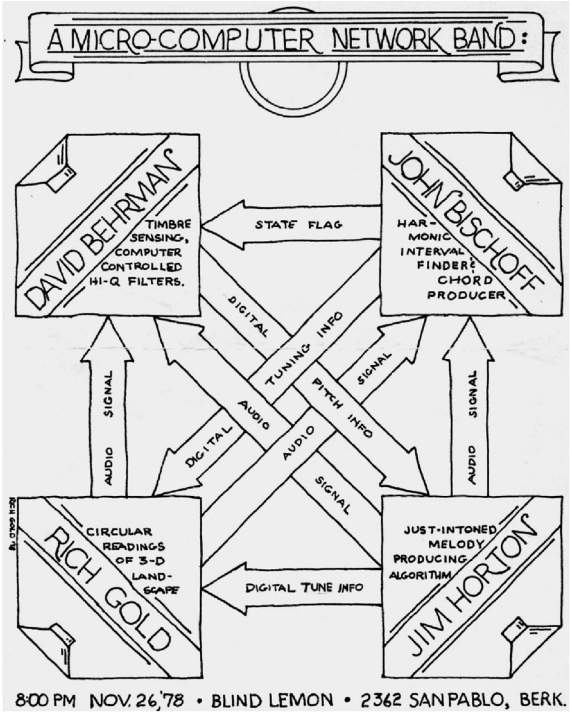


Figure 1.1: A flyer for a performance by the *League of Automatic Music Composers* that shows the network of data exchange between group members. Picture taken from [12]

In that fashion, the members of *The League* themselves were influenced by the American musical Avant-Garde from the 1960s represented by John Cage, David Tudor, Pauline Oliveros, Alvin Lucier, etc. . . [4]. These artists were decisive in the way they invented new forms of composition based on the electronic equipment available at the time (or designed specifically for the performance) and on the indeterminate and decentralized interaction between these equipments, performers and their surrounding environment [26].

[4]: Gresham-Lancaster (2017), 'A Personal Reminiscence on the Roots of Computer Network Music'

[26]: Gresham-Lancaster (1998), 'The Aesthetics and History of the Hub'

Of course, due to the social and cultural importance of "net-

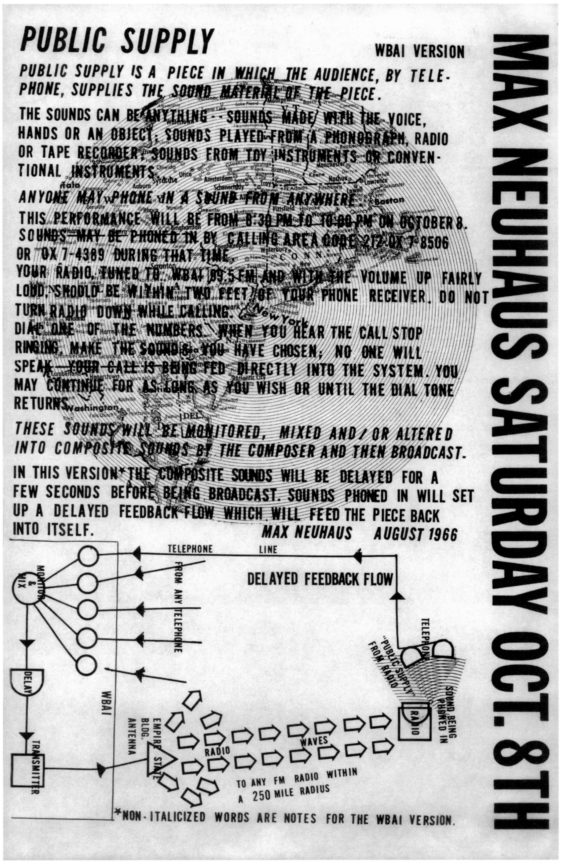


Figure 1.2: A poster for Max Neuhaus' *Public Supply*. Picture taken from https://jenniferstob.com/tag/max-neuhaus/

works" (e.g. transportation networks) in our modern society, the concept of network is pervasive throughout the history of art and network music and holds many precursory works that follow the history of the development of telecommunication technologies[12, 27]. In 1906, the *Telharmonium* system was used to transmit a piano performance over the telephone.

In 1966, Max Neuhaus' performance *Public Supply I* involved the artist mixing incoming telephone calls to a radio station[1] (cf Figure 1.2). In 1977, for the follow-up piece *Radio Net*[2], Neuhaus created a system to automatically mix and create loops out of the incoming phone calls. In 1987, the pioneering performance *Razionalnik*[3] involved the multilateral exchange of MIDI data through the telephone between 4 remote locations (cf Figure 1.3).

[12]: Rohrhuber (2007), 'Network Music'
[27]: Collins (2010), *Introduction to Computer Music*

1: http://www.medienkunstnetz.de/works/public-supply-i/
2: http://www.medienkunstnetz.de/works/radio-net/audio/2/
3: https://kunstradio.at/SPECIAL/XR/history_frameset.html

1.1.2 The Internet Age

At the turn of the millennium, the democratization of computer equipment and of the Internet lowered the technological barrier and provided new ways to use the network as part of artistic

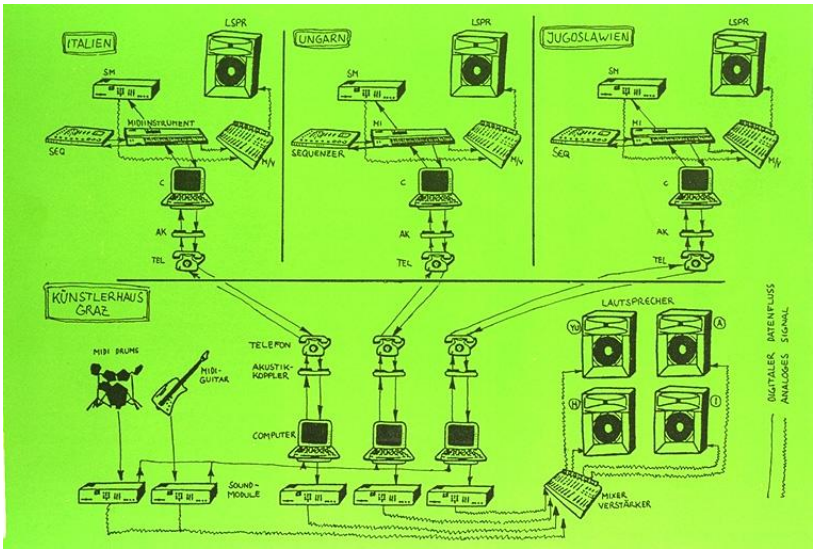


Figure 1.3: Diagram showing the transmission and exchange of data between 4 remote locations in the *Razionalnik* performance

endeavors [28, 29].

[28]: Föllmer (2005), 'Lines of Net Music'
[29]: Traub (2005), 'Sounding the Net'

A New Medium for Sound Propagation

Some artists started using the possibility to transmit data to create musical instruments and installations inspired by the aesthetic of a globalized network as a "new sound propagation medium" [30].

[30]: Chafe (2009), 'Tapping into the Internet as an Acoustical/Musical Medium'

The *Global String* (2000)[31] installation by Tanaka and Toeplitz consists of a physical steel cable and vibration sensors that transmit the vibration of the physical string to a metaphorical "network string" that connects two remote locations. On the other end of the "network string", a similar steel cable reacts to the incoming vibration data. *Global String* is described by its creators as "a musical instrument where the network is its resonating body". *Global String* is remarkable as a network music instrument that mixes heterogeneous mediums in the form of a tangible installation and transmission through a network space.

[31]: Tanaka et al. (2002), 'Global String: A Musical Instrument for Hybrid Space'

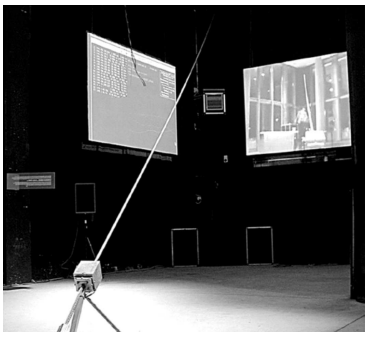


Figure 1.4: *Global String* by Tanaka and Toeplitz. Picture taken from [31]

Of course, with the globalization of sound transmission, the issue of latency starts to appear. While it raises a problem in the case of synchronized remote performances, some artists understood latency as a specific aesthetic feature of network music. For example, works by Chris Chafe and the SoundWIRE project such as *Network Harp* (2001)[32] use the latency delay created by the Internet medium as the basis of a plucked string physical model in which slower transmissions result in a lower pitched sound.

[32]: Chafe et al. (2002), 'Physical Model Synthesis with Application to Internet Acoustics'

Shared Creative Spaces

The Internet also provided an opportunity to create shared creative space and collaborative applications. "By removing the concept of source and site in general, and shattering it across a multitude of non-places"[29], the distributed nature of these applications also implied a blurring of the lines between what constitutes a composition and a performance, between the roles of composer, performer and audience [29]. For Föllmer, networked music "facilitates the creation of forms of music making that can quickly transform the listener at various levels of competence to become collaborators in networked musical projects"[28]. For him, the Internet radically challenges the form of the traditional art work by supporting more collective musical activity "Internet music is a musical form that does not have a work at its core, but rather exists as a collective communal activity similar to the musical cultures of antiquity"[28].

[29]: Traub (2005), 'Sounding the Net'

[28]: Föllmer (2005), 'Lines of Net Music'

An example of such an Internet-based collaborative instrument is *Auracle* (2004)[33] conceived by Max Neuhaus. In *Auracle*, users create music collectively by using a voice-controlled synthesizer interface (cf Figure 1.5). Their vocal input is analyzed and mapped to various control parameters to output synthetic sounds. *Auracle* was conceived as a communal space to encourage participation and dialogue between users "without musical training or technical expertise".

[33]: Freeman et al. (2005), 'Auracle'

Collaborative Composition

While *Auracle* and similar works are geared toward spontaneous improvisation ("jamming") and collective creation by users with little musical expertise, other projects focus on the development of collective composition environments for more advanced musicians.

FMOL (F@ust Music On-Line) (1997)[34] created by Sergi Jordà is composed of multiple authoring components. One of them is based on a server/client architecture that allows users to upload compositions made with the synthesis engine and the multiple GUI available. Users may also download other users' composition and edit them by adding more layers of sound or modifying existing ones. The modified composition will then be stored on the

[34]: Jordà (2002), 'FMOL'

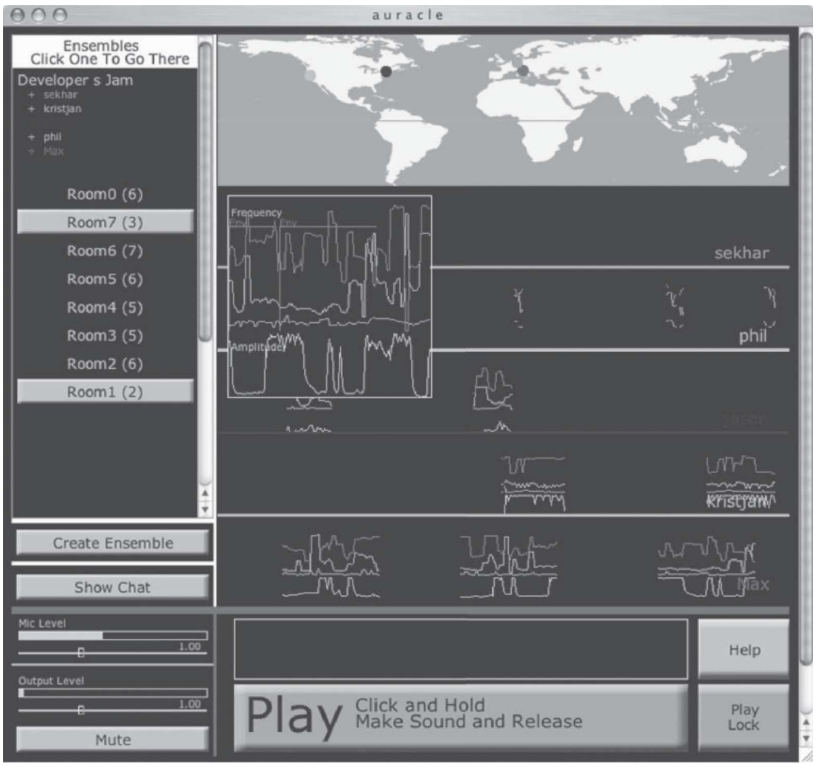


Figure 1.5: The interface for *Auracle*. Dots on the world map at the top display the position of current participants. Picture taken from [33]

server as a child of the original composition in a tree-like structure, allowing multiple users to participate in a single composition alternatively. *FMOL* has been used in multiple contexts including participatory composition for plays and operas as well as for performing improvisations with the *FMOL Trio*.

Quintet.net (1999)[35] developed by Georg Hajdu in the *Max/MSP* environment enables up to five performers to play music under the control of a "conductor". *Quintet.net* is intended to be a complete creation environment "that deals with virtually every aspect of multimedia content production". The environment's architecture is based on a server/client model. The server handles management of the clients and some signal processing tasks. Several clients are available: the main client that allows playing in real-time using MIDI controller and VST (Virtual Studio Technology) plugins (cf Figure 1.6). The conductor client is used to monitor the other clients, influence their performance and send textual directions through a chat functionality. Finally, a listener client allows a remote audience to listen to the mix of the five performers and contains a voting system that can be used by the conductor to influence the performance. While it can be used for real-time performance, *Quintet.net* is also adapted to a dedicated Composition Development Kit that contains a score editor and a bank editor. Files created by the

[35]: Hajdu (2005), 'Quintet.Net'

composition kit can then be opened in the conductor client and shared with the performers during performance.

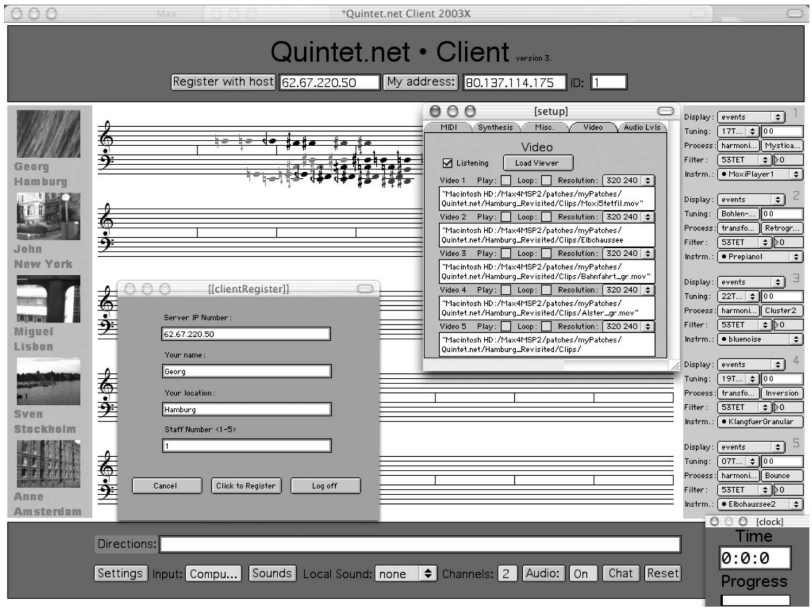


Figure 1.6: The main client interface of *Quintet.net*. Picture taken from [35]

The Internet as a Database

Some works engage with the Internet as a global database and file-sharing space as the basis of creative systems based on a sound collage aesthetic.

N.A.G. (Network Auralization for Gnutella) (2003)[36] by Jason Freeman uses the *Gnutella* Peer-to-Peer network as the basis for sound collages. Users may enter a search term and the application will then find mp3 files that match the term. The application may then choose to download some of those files and mix them together, change their playback speed and loop them[4].

[36]: Freeman (2004), 'N.A.G.'

In a similar fashion, *Playsound.space* (2017)[37] by Stolfi harnesses the *Freesound* online sound database to create a web-based music making tool. In *Playsound.space*, the user may perform search queries to the database and select sounds to play. The user may then choose to change the volume and speed of the sounds and loop them. *Playsound.space* also implements a multi-user chat function that can be used to share queries in the context of participatory improvisation performances [38].

4: an audio example by Freeman is available the following address https://freemusicarchive.org/music/Jason_Freeman/People_Doing_Strange_Things_With_Electricity_Too/csr054_pdstwetoo_2-03_jason-freeman_nag-network-auralization-for-gnutella-dorkbot-mix/

[37]: Stolfi et al. (2018), 'Participatory Musical Improvisations with Playsound.Space'

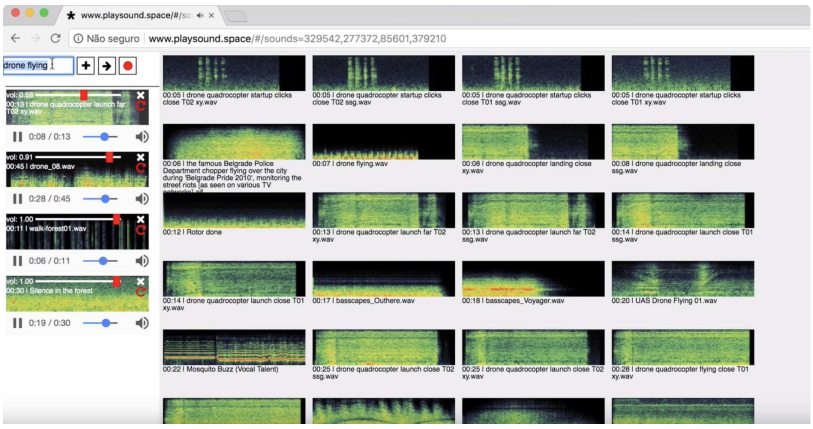


Figure 1.7: The *Playsound.space* interface.

1.1.3 The Mobile Phone Age

Following the development of the Internet, the democratization of the mobile phone (and later the smartphone with its increased computing power and access to the Internet), allowed artists to design distributed music systems in which the audience's portable devices are used for the diffusion of sound or for interaction. An historical review of this movement is available in [39].

[39]: Taylor (2017), 'A History of the Audience as a Speaker Array'

One of the first examples is *Dialtones: A Telesymphony* ('2001) by Golan Levin. Audience members were asked to download a specific ringtone on their mobile phone and were assigned to a specific seat in the audience. During performance, Levin and his partners literally performed the audience by calling specific audience members with a dedicated interface. *Dialtones* exemplifies many aspects of the genre of mobile music. While the sound quality of mobile speakers might be limited, it can create an interesting aesthetical effect when taken into account and also enables spatial distribution of sound within the audience with hundreds of readily available speakers.

Other works involve direct participation of the audience in the creative process through their mobile device rather than acting as passive elements in the diffusion of sound [40, 41]. Xambó and Roma[42] detailed some of the composition strategies available when creating this type of collective experience. Some of these dimensions include whether the audience's actions are independent or interdependent, whether the audience is interacting with performers on stage, the type of sensors used and the type of feedback that is provided, whether the diffusion of sound is performed through the audience's phone or through the room's sound system

[40]: Oh et al. (2011), 'Audience-Participation Techniques Based on Social Mobile Computing'

[41]: Hödl et al. (2020), 'Large-Scale Audience Participation in Live Music Using Smartphones'

[42]: Xambó et al. (2020), 'Performing Audiences'

etc. . .

1.1.4 Interconnected Instruments

As we saw from this non-exhaustive review of some of its main traditions, the field of network music is large and grew up to encompass many different endeavors with often very different aesthetical, cultural and technical concerns. Several taxonomies were proposed to make sense of the variety of works that can be encountered.

For instance Föllmer[43] identifies twelve types of Net music distributed in a 3 dimensional space along 3 criteria defined by: 1) the extent to which the network structure influences the music produced 2) the degree of interactivity offered to the listener 3) the degree of complexity of the system.

[43]: Föllmer (2005), 'Electronic, Aesthetic and Social Factors in Net Music'

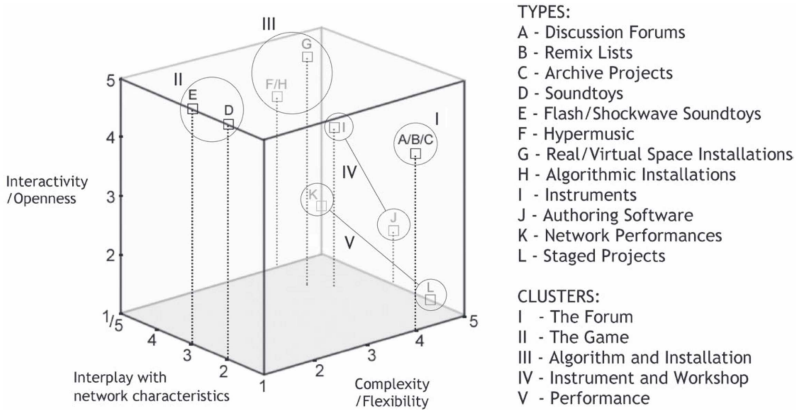


Figure 1.8: Föllmer's classification of Net music in twelve types along three dimensions. Picture taken from [43]

In another approach, Barbosa[44] classifies network music systems on a 2D space with a "location axis" that makes a distinction between remote and co-located systems and an "interaction" axis that separates synchronous and asynchronous systems.

[44]: Barbosa (2003), 'Displaced Soundscapes'

In this thesis, our approach focuses on the "co-located" and "synchronous" quadrant which encompasses what Barbosa calls "Local Inter-Connected Musical Networks" and which can be considered in the direct legacy of what Weinberg introduced as "Interconnected Musical Networks".

In an article, Weinberg, provides a "theoretical framework" of Interconnected Musical Network (IMN)[45] described as systems in which "electronic (or mechanical) communication channels among players" are used for participants to "take an active role in

[45]: Weinberg (2005), 'Interconnected Musical Networks: Toward a Theoretical Framework'

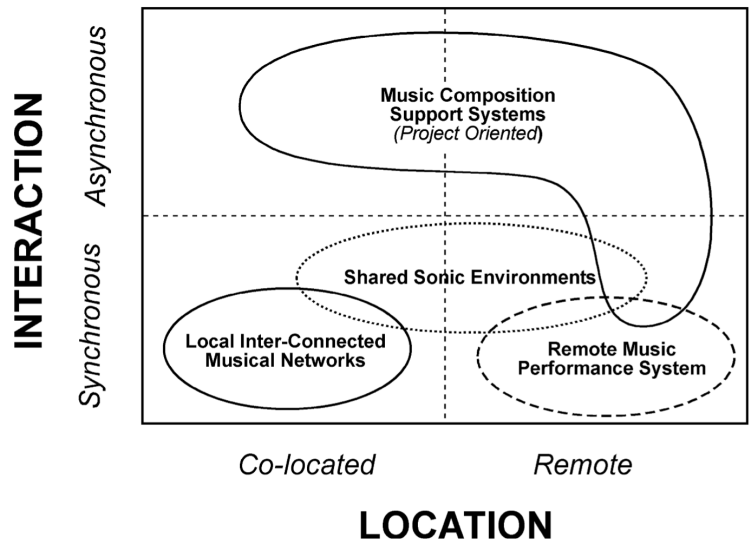


Figure 1.9: Barbosa's classification of network music systems along two dimensions. Picture taken from [44]

determining and influencing not only their own musical output but also that of their peers". In Weinberg's IMNs, the social interaction between players is seen as a key element.

An interesting example of an Interconnected Musical Network is *Soundnet* (1996) created by *Sensorband* (Zbigniew Karkowski, Edwin van der Heide, Atau Tanaka). *Soundnet* consists of a giant 11mx11m network of ropes. At the end of the ropes, sensors detect stretching and movement of the ropes and transmit it to a digital signal processing interface to create sounds. This multiuser instrument is then performed by the three performers by climbing on it (cf. Figure 1.10). While *Soundnet* does not use computer networks, it uses a physical network to exhibit the interconnection between all elements of a network architecture. Indeed, pulling on one rope influences the rest of the structure and the actions of all the performers at the same time.

Weinberg classifies these IMNs along multiple dimensions. A first dimension opposes "process-centered networks" to "structure-centered networks". The latter focuses on the musical outcome as defined by the composer, the former focuses on players' experience and interaction and on the exploration of novel ways of playing. Another dimension describes the "topology" of the network, meaning the organization of the nodes of the network, the path of communication and interaction that connects them and their hierarchical structure. Weinberg describes many different network topologies depending on whether the network is conceived as centralized or decentralized, synchronous or sequential (cf. Figure 1.11).

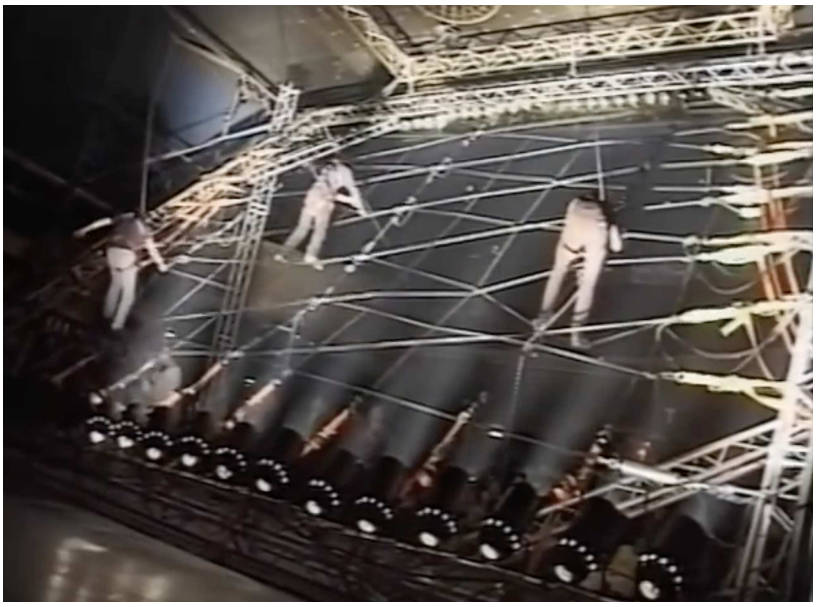


Figure 1.10: *Sensorband* performing with the *Soundnet* installation.

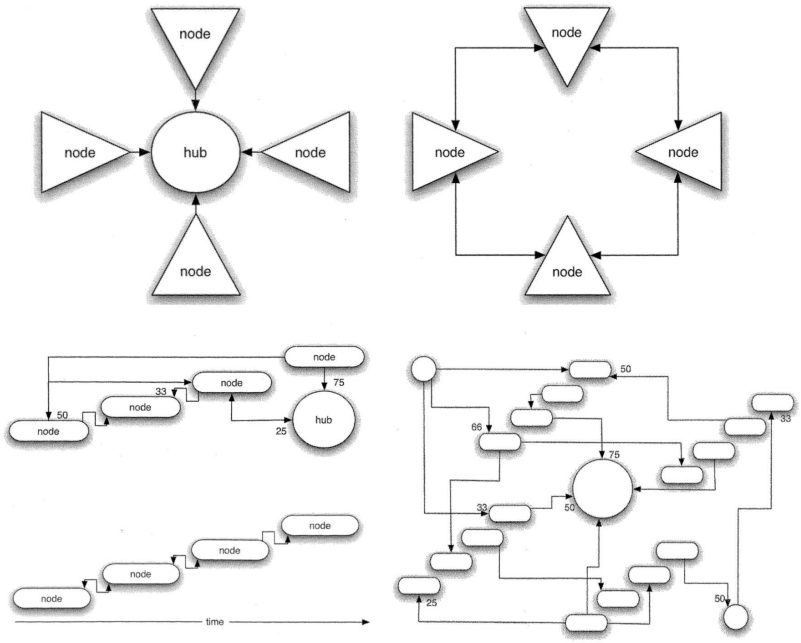


Figure 1.11: Some examples of network topologies as described by Weinberg. Picture taken from [45].

This conceptual framework of network topologies was later refined by Matuszewski *et al.* to describe what they call "Situated Networked Music Systems"[5]. For them, an important problem in the description of network topologies by Weinberg is that the high-level social organization of the network is determined by the lower-level technical implementation of the network. Instead, the conceptual framework of Matuszewski *et al.* aims at "consolidating current theoretical frameworks by decoupling the topological interaction description from its low-level technical implementation aspects". Furthermore, in an approach inspired by Latour's Actor-

5: The term "situated", that can be heard as a synonym of "co-located", refers to the fact that they focus on interactions happening in a shared physical space

Network Theory (ANT)[7], they propose to "consider any actor, human or not, performing or not, as a node in the network". Finally, they propose 6 different "interaction topologies" describing possible interactions between actors that can be implemented with the same technical network architecture (cf. Figure 1.12).

[7]: Latour (1996), 'On Actor-Network Theory. A Few Clarifications, Plus More Than a Few Complications'

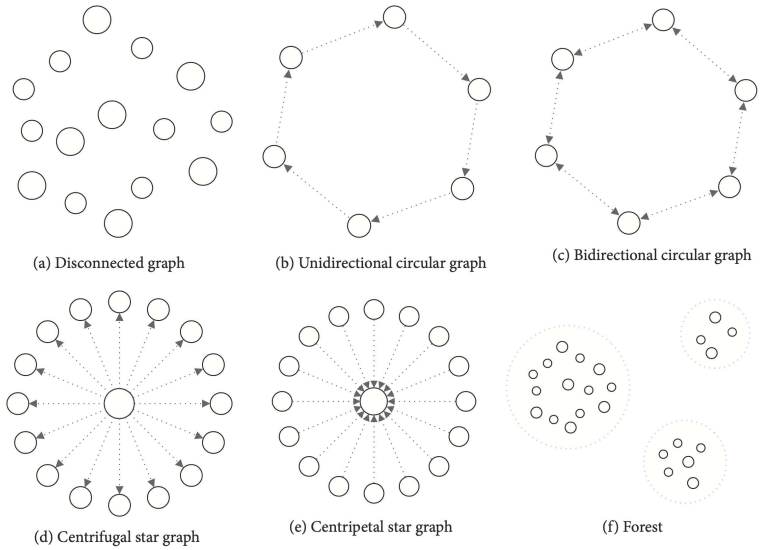


Figure 1.12: Interaction topologies as described by Matuszewski *et al.*. Picture taken from [46].

These different topologies were implemented in multiple applications and installations as part of the *CoSiMa*[6] (Collaborative Situated Media) research project at Ircam whose goal was to develop a platform for collective interaction based on web and mobile technologies. Reflecting on this research project, Bevilacqua *et al.*[8] advocates for a "holistic perspective" of Interconnected Musical Networks by considering the complex entanglement of interactions between human actors and devices and the "sociocultural assemblages" that results from it. A large part of this thesis is placed in the direct continuation of this research project by drawing from its conceptual framework as well as most of its technological components (cf. Chapter 2).

6: https://cosima.ircam.fr/

[8]: Bevilacqua et al. (2021), 'On Designing, Composing and Performing Networked Collective Interactions'

Some works developed during the *CoSiMa* project include *Drops* (2015), a collective experience where participants may play various "sound drops" by touching their smartphone's screen. The sound drop will then be echoed on two other participant's phones before coming back to the original player. Another work is *CoLoop* (2017), a physical speaker playing loop sequences that may be created collectively by participants using their smartphones.

1.1.5 From Ubiquitous Computing to the Internet of Musical Things/Ubimus

At the beginning of the 1990s, Mark Weiser coined the term *Ubiquitous Computing* to describe networked arrangements of devices seamlessly integrating into the background of our work environment that he and his collaborators were developing at the Xerox Palo Alto Research Center (PARC)[5]. In Weiser's vision, each electronic device in a room is equipped with a small computer and communicates with other computers present around it thus replacing the paradigm of the personal computer and laptop. Moreover, they developed computers of different sizes from tabs (the size of a post-it) to boards (the tactile equivalent to a chalkboard). Therefore, Weiser's ubiquitous environment required the development of a network infrastructure that seamlessly connects heterogeneous hardware components.

[5]: Weiser (1991), 'The Computer for the 21st Century'

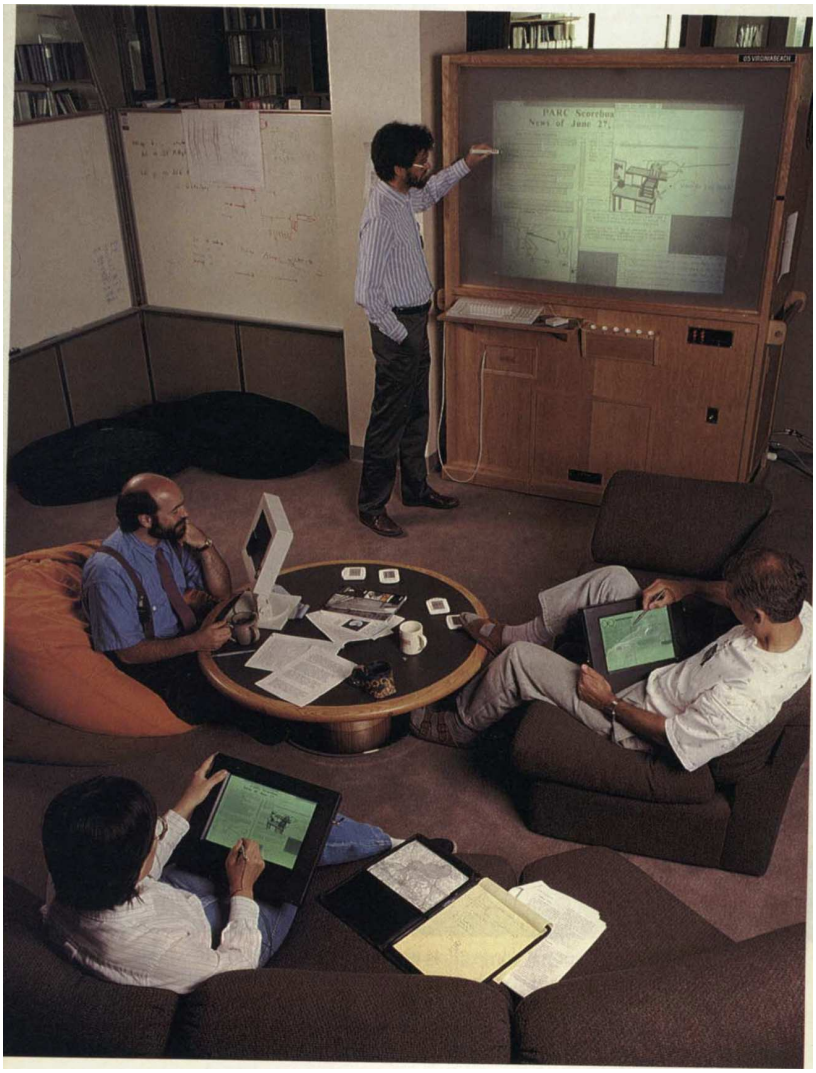


Figure 1.13: Weiser's utopian vision of "Ubiquitous Computing". Tactile boards replace the classic chalkboard and communicate with portable tablets carried around by users. Picture taken from [5].

Of course, Weiser's original predictions, imbued with the techno-utopianism zeitgeist of the late 20th century, did not stand the test of time and the technological utopianism of Weiser's vision was shattered by the capitalist development of technology over the last decades with its increased demand for advertising, nonstop notifications and attacks on privacy[47].

[47]: Aylett et al. (2015), 'The Broken Dream of Pervasive Sentient Ambient Calm Invisible Ubiquitous Computing'

Nevertheless, most ideas of Ubiquitous Computing persevered through the HCI and CSCW (Computer-Supported Cooperative Work) communities and led to the creation of specific research communities around the Ubicomp conference and the field of Internet of Things (IoT) among others.

In the music field, Keller and Lazzarini founded the "Ubiquitous Music Group" that aims to "investigate the creative potential of converging forms of social interaction, mobile and distributed technologies and materially grounded artistic practices"[48, 49] and coined the term "Ubiquitous Music" (or UbiMus). Likewise, influenced by the IoT field, Turchet *et al.* developed the Internet of Musical Things (IoMusT) paradigm[50] described as "wireless networks of smart devices dedicated to musical purposes, which allow for various forms of interconnection among musicians, audio engineers, audiences, and educationalists, in both co-located and remote". The IoMusT aims to "foster new opportunities for the musical industry, paving the way to new services and applications capable of exploiting the interconnection of the digital and physical realms".

[48]: Keller et al. (2018), 'Theoretical Approaches to Musical Creativity'
[49]: Schiavoni et al. (2019), 'Interaction and Collaboration in Computer Music Using Computer Networks'
[50]: Turchet et al. (2018), 'Internet of Musical Things'

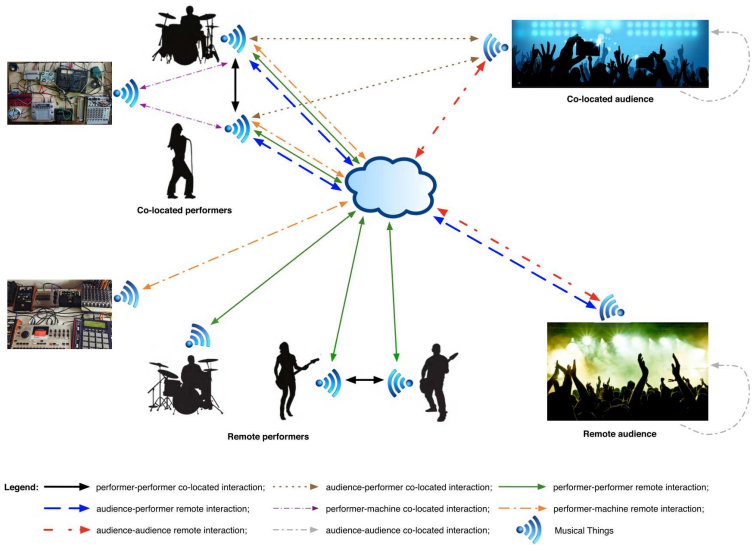


FIGURE 1. **Interaction possibilities between musicians, audience members, and machines.**

Figure 1.14: Diagram describing interactions within the IoMusT paradigm. Picture taken from [50].

Compared to this approach, our approach is focused on the high-level observation and integration of distributed applications within artistic research and musical practice, rather than on the low-level technological requirements and challenges. To do this we adopt various research methodologies that we describe in the next section.

1.2 Epistemology and Methods

In this section we present a review of some of the epistemological and methodological elements that we used throughout this thesis. We first discuss the notion of experimental systems as proposed by Rheinberger. We then review some examples of practice-based design research methodologies. After that we give some background on the concept of third- second- and first-person perspective in design. Finally we address the specific topic of appropriation.

1.2.1 Experimental Systems

The concept of experimental systems was proposed by Rheinberger in the context of "an ongoing replacement in history of science of a theory-dominated by a practice-dominated perspective"[18]. For him, experimental systems form the "privileged spaces in which the production of knowledge, that is, the generation of *new* knowledge takes place".

[18]: Rheinberger (2011), 'Consistency from the Perspective of an Experimental Systems Approach to the Sciences and Their Epistemic Objects'

The concept of experimental systems is beneficial to our work as it provides us with a holistic approach to the research context and environment of this thesis. In the words of Rheinberger, "the advantage of the concept of experimental system lies in its faculty to think and to bind together essential, but nevertheless very different and heterogeneous aspects of the scientific research process - such as instruments and measurement apparatus, preparation arrangements of different kinds, the necessary skills to use them in meaningful ways, the research objects, and not least the spaces in which these moments are brought to interact with each other in productive and creative arrangements"[18].

At the core of an experimental system, and its driving force over time, is what Rheinberger describes as a dialectic between

"epistemic things" (that is the object of research) and "technical objects" (the instruments and apparatus, the technical setup used for the observation of epistemic things and the condition for their existence). Furthermore, in experimental systems, objects' roles are never fixed in time. As observation and knowledge are refined, epistemic things can become technical objects of their own. Likewise, some elements of the technical setup can regain a status of epistemic things. Therefore, for Rheinberger "experimental systems are therefore to be seen as dynamic, materially conditioned research bodies; they bring scientific objects into existence, and at the same time, determine the boundaries of their conceptual apprehension"[18].

The concept of experimental systems was used by Schwab *et al.*[51] in the context of the context of artistic research to make sense of the specific way in which works of art may exhibit an epistemic value and to provide a better understanding of "artistic modes of investigation".

[51]: Schwab et al. (2013), *Experimental Systems: Future Knowledge in Artistic Research*

Schwab observes that the transposition of the experimental systems to the art is not without problematic consequences. For instance, he argues that the status of technical objects in context of experimentation and the characterization of experimental systems as "producers of technology" is at risk of imposing a "techno-solutionist" approach to art: "even if we interpret "technology" very broadly [...] the implication remains that contemporary artistic output can be "black-boxed" to operate functionally in a new experimental setting" and that "from the vantage point of contemporary art, the dialectic between epistemic thing and technical object may simply not be transferable to experimentation within artistic research; to transfer it raises expectations of utility that are regressive and potentially detrimental to artistic practice."

Nonetheless, Schwab *et al.* affirms that the sense of movement, often unpredictable, of experimental systems, and the "sense of potentiality" that guides the researcher is applicable to artistic research: "despite such difficulties, such discussions can serve to acknowledge that limited sets of materials and unique practices, brought together as part of longstanding engagements with meaning that has not yet been achieved, bring about occasional surprises and a sense of movement that is beyond one's control"[51].

Throughout this manuscript, we aim to provide a perspective

on the research projects that are described therein inspired by experimental systems. For instance, In Chapter 2, we try to affirm the material contexts in which this research takes place by describing the ecosystem of tools which we used. While these tools form the technical objects of our research project, they themselves used to hold an epistemic status when they were developed[52] and the experiments we performed through our different use cases called for frequent re-interrogations of their implementation and updates.

Likewise, at some times the focus of our research was on the design of distributed systems but we later used them as tools in experimental setups. At other times, conclusions we drew from the observation of our tools "in the wild" then served as the basis in the development of new tools.

Finally, throughout this thesis we adopt a interdisciplinary approach, in accordance with Rheinberger's observation that "The modern experimental sciences derive their dynamics less and less from drawing disciplinary boundaries and from cementing them socially, and more and more from the digressions and transgressions of smaller research units below the level of disciplines, in which knowledge has not yet become labeled and classified, and in which new forms of knowledge can take shape at any time"

1.2.2 Practice-Based Research in HCI

In the fields of HCI and design, the turn toward "practice-based perspective" has taken multiple forms and yielded different approaches[53].

[53]: Goodman et al. (2011), 'Understanding Interaction Design Practices'

Benford *et al.*[54] proposed the concept of *performance-led research in the wild* to describe situations in which artists "drive the creation of novel performances with the support of HCI researchers that are then deployed and studied at public performance". They argue that such an experimental framework is useful to describe how artists may leverage interactive technologies in real-life performance context and how artists may shape the research questions and design process of such technologies.

[54]: Benford et al. (2013), 'Performance-Led Research in the Wild'

First developed in the Francophone world, mainly in Quebec, *research-creation* is defined by Stévance and Lacasse as "a research

approach established from or through a process of creation fostering the diffusion of an artistic production and a theoretical discourse"[7][55].

In her "Ten Propositions for Research-Creation"[56], Manning has described the capacity of research-creation to not only answer how art generates new forms of knowledge but also how these new forms of knowledge are "extra-linguistic" and "may have no means of evaluation within current disciplinary models". For her, the issue with these current disciplinary-models resides in the way they tend to generate "accounts of experience that separates out the human subject from the ecologies of encounter"[57], thus resulting in forms of knowledge that are static and confined within the boundaries of pre-existing categories. Manning therefore encourages us to "find ways of activating thought that is experienced rather than known, that is material and affective, and where experience accounts for "more than human" encounters"[58] and argues with Deleuze that "making is a thinking in its own right".

The *research through design* approach has been introduced by Frayling[59] and later refined by Findeli as "project-grounded design"[14] is a methodology in which knowledge is drawn from the design process of artefacts or systems. Likewise, Redstrom has qualified research through design approaches as forms of knowledge "starting from the particular"[60] while noting disagreement about the implication of such a stance.

On one side, Gaver has stated how "it is the artefacts we create that are the definite facts of research through design"[61] and has argued that research through design approach should not strive for convergence and standardization as part of a "scientist" tendencies in HCI but should instead "take pride in its aptitude for exploring and speculating, particularising and diversifying, and - especially - its ability to manifest the results in the form of new, conceptually rich artefacts".

On the other side, Zimmerman and Forlizzi instead suggests that artefacts are the basis for more generalizable results and a bridge toward more theoretical accounts: "the artifact functions as a specific instantiation of a model - a theory - linking the current state to the proposed, preferred state"[62].

Beyond those disagreements, Redstrom highlight how research

7: "une démarche de recherche établie à partir de ou à travers un processus de création encourageant dans son sillon la diffusion d'une production artistique et d'un discours de nature théorique"

[55]: Stévance et al. (2014), *Les enjeux de la recherche-création en musique*

[56]: Manning (2016), 'Ten Propositions for Research-Creation'

[57]: Manning (2016), 'Against Method'

[58]: Springgay et al. (2015), 'How Do You Make a Classroom Operate like a Work of Art? Deleuzeguattarian Methodologies of Research-Creation'

[59]: Frayling (1993), 'Research in Art and Design'

[14]: Findeli et al. (2008), 'Research through Design and Transciplinarity: A Tentative Contribution to the Methodology of Design Research'

[60]: Redström (2021), 'Research through and through Design'

[61]: Gaver (2012), 'What Should We Expect from Research through Design?'

[62]: Zimmerman et al. (2008), 'The Role of Design Artifacts in Design Theory Construction'

through design approaches often display attention to the wide array of notions that surrounds the design practice: "it is not only the making as such that grounds the perspective, but also how objects, materials and more make certain experiences possible to both designers and others (intended users included)"[60]

In the field of computer music and NIME, Tahiroğlu *et al.* have shown how Digital Music Instruments as artefacts can be considered as "probes" to provide insights into the nature of music practice and instrumental design[63]. They state that "the design process is a cultural experiment where we give something out into the world, whether that be an instrument, installation, code library or the music itself, and in that process we observe how the work is received and gain invaluable feedback or return".

[63]: Tahıroğlu et al. (2020), 'Digital Musical Instruments as Probes'

1.2.3 Third, Second and First-Person Perspectives

Throughout this thesis we employ multiple methodological perspectives ranging from third-person perspective to first-person perspective[64, 65].

[64]: Tomico et al. (2012), 'Designing for, with or Within'
[65]: Smeenk et al. (2016), 'A Systematic Analysis of Mixed Perspectives in Empathic Design: Not One Perspective Encompasses All'

Third-person perspective represents the majority of design projects in which the design process is based on users' perspectives (often obtained through interviews) and observations of how they use the designed system while still maintaining a conceptual distance between designers and users. In such cases, the designer is often seen as a neutral and objective expert, not carrying a subjective experience nor imposing any personal influence on the design process. Works in the third person perspective often employ a User-Centered Design (UCD) approach based on assumptions such as "that designs should be based on empirical, average, or acceptable user needs, or that the hallmark of quality for a design concept is how well a designed object functions in an empirically circumscribed setting"[66].

[66]: Pierce et al. (2015), 'Expanding and Refining Design and Criticality in HCI'

Second person perspective includes works in which the users are directly involved in the design process along with the designer. Methodologies in the second-person perspective include Participatory Design or Co-Design in which users are directly involved throughout the design process[67, 68]. Second-person practice often involves forms of participation such as workshops and co-design sessions[69]. Mackay and Beaudouin-Lafon argue

[67]: Muller et al. (1993), 'Participatory Design'
[68]: Mackay et al. (2023), 'Participatory Design and Prototyping'
[69]: Lucero et al. (2012), 'The Dialogue-Labs Method'

that participatory design practices "helps designers develop a deeper understanding of the actual design problem and avoid faulty design paths"[68]. They also stress out the fact that in such cases "designers must always consider what users can and cannot contribute" and "designers and developers are responsible for considering the range of options and constraints and for balancing the trade-offs among them".

[68]: Mackay et al. (2023), 'Participatory Design and Prototyping'

On a similar line, second-person perspective can include works in which the designer's personal subjective experience and perspective is taken into account. For instance, Pierce[66] called for "authorial voices" and design that "explicitly and significantly embodies intentions or ideas arising from a concern or curiosity of the designer". Such a position enables "new relationships with 'users', not as potential consumers of utilitarian products to solve their needs, but as collaborators in discovering new meanings and values in the things they design".

Finally, works in the first-person perspective acknowledge the designer's positionality and use the designer's own lived experience as a resource for design and as a source of knowledge [70]. This includes methodology such as autoethnography[71], autobiographical design[72, 73] and micro-phenomenology[74].

[70]: Desjardins et al. (2021), 'Introduction to the Special Issue on First-Person Methods in HCI'

[71]: Bang et al. (2024), 'A Retrospective Autoethnography Documenting Dance Learning Through Data Physicalisations'

[72]: Neustaedter et al. (2012), 'Autobiographical Design in HCI Research'

[73]: Gaver (2006), 'The Video Window'

[74]: Prpa et al. (2020), 'Articulating Experience'

While first-person perspectives in a research context have raised questions regarding their generalizability[16], Zhang and Wakkary[75] have argued for recognizing the "legitimacy of designers' personal experiences in interaction design practice" . Furthermore, in providing a reflective account of one's own experience, first-person perspectives can generate new collective insights by "resonat[ing] critically with the reader's personal experience and understanding of the interaction with technology"[16].

[16]: Fdili Alaoui (2023), 'Dance-Led Research'

[75]: Zhang et al. (2014), 'Understanding the Role of Designers' Personal Experiences in Interaction Design Practice'

1.2.4 Designing for appropriation

One of our main considerations during this thesis concerns the notion of "user appropriation" of our tools. Appropriation is defined by Dourish as "the way in which technologies are adopted, adapted and incorporated into working practice. This might involve customisation in the traditional sense (that is, the explicit reconfiguration of the technology in order to suit local needs), but it might also simply involve making use of the technology for

purposes beyond those for which it was originally designed, or to serve new ends"[11].

[11]: Dourish (2003), 'The Appropriation of Interactive Technologies'

Appropriation is a frequent concern in HCI and design methodologies. Mackay described how customization happens as a collaborative activity within organizations through the sharing of customization files[76]. Gaver *et al.* suggests that "ambiguity" may be considered as a resource for design in HCI for the "ability it gives designers to suggest issues and perspectives for consideration without imposing solutions" and encourages users to "establish deeper and more personal relations with the meanings offered by systems"[77]. Caroll establishes a model of technology appropriation, arguing that "the design of a technology innovation is completed by users as they appropriate it. Therefore, understanding the influences on the appropriation of a specific technology or family of technologies is an essential part of the design process"[78].

[76]: Mackay (1990), 'Patterns of Sharing Customizable Software'

[77]: Gaver et al. (2003), 'Ambiguity as a Resource for Design'

[78]: Carroll (2004), 'Completing Design in Use: Closing the Appropriation Cycle'

Dix proposes some design guidelines and practical advice to design *for* appropriation[79]. Among them, he includes "support not control" ("Instead of designing a system to do the task you can instead design a system so that the task can be done), "encourage sharing", "learn from appropriation" (use observation of the way users appropriated the system in the design of a new system) and "pluggability and configuration" ("create systems where the parts can be plugged together in different ways by the user) which he considers a critical issue in ubiquitous computing. In the context of ubiquitous computing, Newman *et al.* argue that "systems should inherently support the ability of users to assemble available resources to accomplish their tasks" and value the design of "highly generic tools that allow end-user discovery, configuration, interconnection, and control of the devices around us"[80].

[79]: Dix (2007), 'Designing for Appropriation'

[80]: Newman et al. (2002), 'Designing for Serendipity'

Fischer and Giaccardi proposed a conceptual and methodological design framework that aims to foster user appropriation and emergent uses called *Meta-Design*[81]. Their framework is especially fit for situations (of which artistic practice belongs to) in which "future uses and problems cannot be completely anticipated at design time", arguing that "if systems cannot be modified to support new practices, users will be locked into old patterns of use".

[81]: Fischer et al. (2006), 'Meta-Design'

Meta-Design could be summarized by three important characteristics. First, as it can't be completely designed prior to use, the

application must be designed to evolve, and moreover to co-evolve with its users. Second, the application should support and provide a learning path from simple user to expert user and ultimately to co-designer of the application. Finally, this process takes place in a model called *Seed-Evolving Growth-Reseeding (SER)* (cf. Figure 1.15), in which software development is performed during the *Seed* and *Reseeding* phases, while the *Evolving Growth* phase is dedicated at observing and documenting how users adopt and appropriate the application.

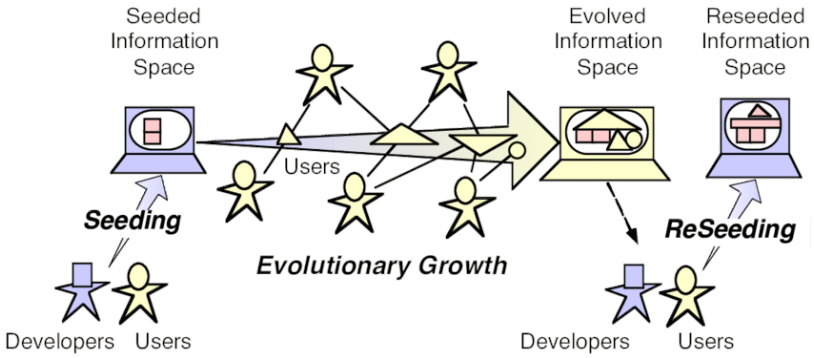


Figure 1.15: The Seed-Evolving Growth-Reseeding (SER) described by Fischer and Giaccardi. Picture taken from [81].

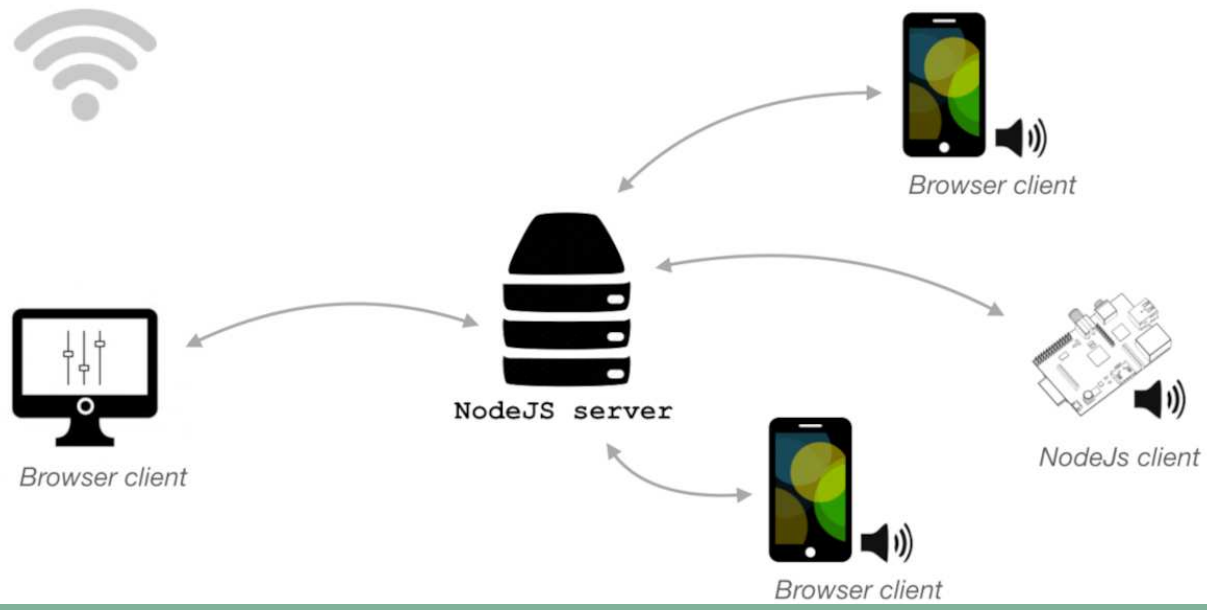
In this thesis, we use the *Meta-Design* in the design of *Koryphaîos* (cf. Chapter 4) and, to a smaller extent, in the design of *Simone Solo* (cf. Chapter 6.

In the context of the design of Digital Music Interfaces (DMI), some works have addressed the topic of appropriation. Zappi and McPherson[82] conducted a study in which participants were given a specifically built instrument with highly-constrained sonic capabilities. Through this experiment, they observe the development of diverse playing styles among participants. Through observation of some DMI systems, Magnusson[83] finds that it is an instrument's set of constraints that guide user appropriation. Finally, McPherson and Tahiroğlu[84] studied how music programming languages (such as *Max/MSP* or *SuperCollider*) possess a "latent influence" and impose "idiomatic patterns" on the design of DMI they are used for.

[82]: Zappi et al. (2014), 'Dimensionality and Appropriation in Digital Musical Instrument Design'

[83]: Magnusson (2010), 'Designing Constraints'

[84]: McPherson et al. (2020), 'Idiomatic Patterns and Aesthetic Influence in Computer Music Languages'



2 An Ecosystem of Web-Based Tools

As stated in the introduction chapter, our intent is to build distributed systems that match certain objectives: 1) The ability to run seamlessly on a fleet of many heterogeneous devices. 2) The ability to interact with tools used by artists in their usual workflow 3) The ability to rapidly prototype ideas in a creative context and to be versatile by making adaptation to new use cases feasible.

To do this, we build upon an ecosystem of tools, hardware and software, some of them standardized and largely adopted over many applications, some of them developed over the years by our team in projects related to this thesis and some of them developed over this thesis. By presenting this (partial) account of this ecosystem, we aim to show that while this thesis led to the development of specific and singular higher-level projects, it also takes place in a years-long team-wise research project on distributed music systems to develop the base tools necessary to develop these projects.

My personal contribution to this ecosystem lies mainly at 3 levels: 1) beta-testing new tools and features, 2) proposing new features guided by my userpi cases (e.g. microphones support on Raspberry Pi and in the `node-web-audio-api`, cf. 2.2.1), 3) developing some components for specific use cases (e.g. a web component for supporting MIDI controllers in our interfaces, cf. paragraph 10).

This ecosystem is mostly based on web technologies standard-

ized by the World Wide Web Consortium (W3C), such as the Web Audio API[1], the Web Midi API[2] and WebSockets[3], and on the *Node.js* JavaScript runtime, which, while not per se a standard, is backed by the Open JS Fundation[4].

1: https://www.w3.org/TR/webaudio/
2: https://www.w3.org/TR/webmidi/
3: https://websockets.spec.whatwg.org/
4: https://openjsf.org/

Using web technologies for the software ecosystem allows us to develop cross-platform applications which can be deployed on any available hardware that is able to run either a web browser or *Node.js*. Our systems can therefore naturally be integrated in many workflows and situations with minimal to none additional code.

As an example, we can consider a synthesizer controller interface that could simultaneously be opened on a laptop browser for playing while sitting at a desk and on a tablet for playing with tactile controls while standing and moving across the room. Similarly, it allows for creating synthesizer clients that can simultaneously run on a laptop and play sound from large dedicated speakers in a concert room, on the audience smartphones for a more diffuse sound around the room and on small nano-computers equipped with portable speakers.

2.1 Hardware

. For a device to be part of our systems, we only require the ability to connect to Wi-Fi networks, ability to run a JavaScript runtime (web browser or *Node.js*) and an audio output.

In some cases, we may benefit from devices with extended features such as an audio input, a screen, or diverse sensors and / or actuators. In particular, a major direction that this thesis followed was to use audio inputs and microphones in the context of distributed music systems.

We present the different types of devices used in this thesis (mainly laptop, smartphones and Raspberry Pi) and detail their advantages, drawbacks and for which situations they are more fitting.

2.1.1 Laptops

Laptops are of course the go-to platform for music creation nowadays and an essential part in the workflow of most people working in the ecosystem of music creation. For this reason, integrating laptops in our distributed systems will give us an opportunity to interact with other music creation software (most of the time using OSC communication) that are part of musicians' usual workflow such as *Ableton Live* or *Max/MSP*.

Additionally, laptops possess a high number of connectors which could allow us to expand the scope of our system and enhance their compatibility with users' practices and usual hardware equipment. For instance this includes higher grade sound cards for higher fidelity input/output or MIDI controllers that are often an essential element when creating digital music instruments.

For this reason, a laptop is a very useful device to create a controller interface for a distributed system. In this thesis we will often use them to give access to web interfaces used to monitor other clients (cf. Chapter 3), display an interface that controls a distributed digital music instrument (cf. Chapter 6 and 7) or provide an option for users to develop some customized elements that will be broadcasted to a distributed system (cf. Section 2.3 and Chapter 4). These controller interfaces will often be intended to be used by a single user, often with a particular status compared to the other human or non-human actors in the system (e.g. a musician controlling multiple clients that play sounds on the audience's smartphones).

In some cases, for the same reasons, laptops can also be useful to provide fully equipped playing stations for a distributed multiplayer instrument with a low number of players (cf. Chapter 5). However, due to their weight, cost and fragility, laptops will not be used in cases where we need numerous clients or portable setups.

2.1.2 Smartphones

Smartphones offer many advantages when used in the context of distributed music systems.

As they are now widely adopted by the general population, smartphones form a pool of devices that will almost always be available in most contexts. Smartphones are therefore a great choice of device when you want some level of participation from human actors (e.g. the audience of a concert or the visitors of an installation) or if you want a low-cost solution (since you'll exploit devices that the actors will bring).

Over the years, smartphones have become more and more versatile devices, offering many options to developers. In some contexts, the presence of a screen can be useful for providing guidance or feedback to the audience (for instance by prompting them to perform a specific action). Beyond possessing a microphone (particularly useful in some of the cases we will consider), smartphones are now equipped with many different sensors, providing new ways of interacting with the device: accelerometer, gyroscope, ambient light detector, etc.

Finally, while the sound of the smartphones' speakers could be described as "low fidelity", it can be remarked that in the presence of a minimal number of device (let's say a dozen), this is not a problem as the mass of sound will often blur the harshness of the individual device's sound, even offering a very interesting effect to consider when composing.

However, smartphones also possess their own shortcomings and drawbacks that makes it a sometimes frustrating platform to use in the contexts we consider.

Applications relying on the audience's smartphones will necessarily be confronted with the differences in technological literacy of their audience. From connecting to a local Wi-Fi network to connecting to a specific URL (Uniform Resource Locator) in the web browser and sometimes necessitating unusual ways of interacting with the device, each of these steps will not always be familiar to by audience members. This will almost always require some sort of mediation and in the worst case lead to a phase of live debugging, thus breaking the flow of a live event, even for the best prepared team.

Because of the lack of standardized characteristics and operating software between various brands and models of smartphones, developing on this platform will also lead to various problems

of compatibility (in particular with some web technologies only available in the latest versions of the OS which are sometimes not available on older models) and differences in latency. Because the output latency can largely differ on the model (sometimes more than hundreds of milliseconds), applications using the audience smartphones to play sound will necessarily need to adapt to the fact that precise synchronization is not reachable.

Moreover, smartphones' inner functioning and characteristics are sometimes rather opaque and unpredictable. For instance some superfluous processes can never be turned off, which may cause some performance issues or even disturb the current main process. For instance, a smartphone user in a distributed context may be disturbed by an incoming call or notification, or in applications where user interaction is not needed, the smartphone will unpredictably go into sleep mode. Finally, smartphones offer very little flexibility as a development platform and can generally not adapt to a variety of use cases by being augmented with more equipment. This is a problem in creative situations where a prototyping workflow is often favored.

In summary, smartphones are a great choice for devices in distributed music systems if you need some sort of interaction from the audience (as seen in Chapter 3). However, even in this case, they lack the reliability and predictability that is necessary in the context of music creation (cf. Chapter 4). Moreover, as we mentioned in the previous chapters, our objective was to move away from the perspective of "audience interaction" in the context of networked-based musical applications.

2.1.3 Raspberry Pi

Working with embedded devices offers many advantages and can mitigate some of the shortcomings experienced with using smartphones.

Among all the different solutions for embedded devices and nano-computers (e.g. the Bela platform [85]), we chose to focus on the Raspberry Pi computer (cf Figure 2.1). In its version 4, the Raspberry Pi offers the following features: a quad core Cortex-A72 (ARM v8) 64-bit processor at 1.8GHz; up to 8GB of LPDDR4-3200 SDRAM; two USB 2.0 and two USB 3.0 ports; 2.4 GHz and 5.0 GHz

[85]: McPherson et al. (2015), 'An Environment for Submillisecond- Latency Audio and Sensor Processing on BeagleBone Black'

IEEE 802.11ac wireless, Bluetooth 5.0 and BLE Gigabit Ethernet for networking and communications, and a 40 pin GPIO header to interface with external electronic devices.

Additionally, we chose to extend the platform with a Hi-FiBerry DAC+ ADC Pro sound card [5] which features stereo input and output, support up to 192kHz sample rate and communicate with the Raspberry Pi through the I2S bus.

5: https://www.hifiberry.com/shop/boards/hifiberry-dac-adc-pro/

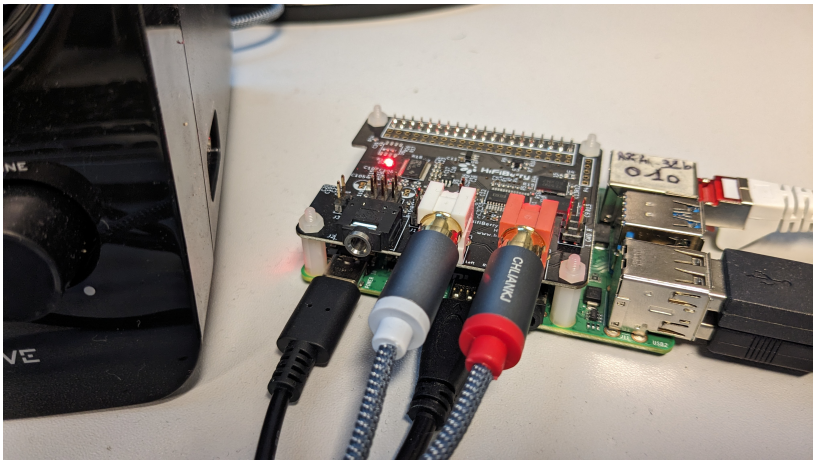


Figure 2.1: The standard Raspberry Pi hardware used in this thesis: a Raspberry Pi 4 and a Hi-FiBerry DAC+ ADC Pro sound card.

We use the default Raspberry Pi operating system and the installation and configuration of basic software components (e.g *Node.js* and the Jack audio server [86]) rely on a few configuration scripts.

[86]: Letz et al. (2005), 'Jack Audio Server for Multi-Processor Machines'

Compared to smartphones, embedded devices offer a more flexible platform that can be customized to fit one exact needs. In particular, this fixes the problem of smartphones going into sleep mode or being disturbed by other processes unpromptedly. Moreover, depending on the project, the "standard" minimal setup we described can be easily extended or adapted. For example, the Raspberry Pi can easily work together with microcontrollers such as the Arduino platform for the prototyping of more traditional digital musical instruments.

Moreover, a Raspberry Pi offers much better autonomy than a smartphone. Using batteries, they can provide a very portable system which facilitates deployment in unusual spaces (e.g. an art gallery or a forest).

Finally, our minimal setup is rather low cost which is of primary importance when building a fleet of devices composed of numerous devices.

Raspberry Pi's main drawbacks lies in the fact that setting up a fully working environment will therefore be a very demanding task in terms of development. As a consequence of the flexibility of the environment, there are a lot of building blocks to create and combine before coming up with a device and a basic software stack that makes it usable in our context. Fortunately, the Raspberry Pi benefits from a large community of users and of important documentation and support.

Once these building blocks have been developed, the Raspberry Pi therefore became the platform of choice in our projects. Using Raspberry Pi computers is especially useful when some of the elements in the networks need to behave as audio output without direct handling or interaction (cf. Chapter 6) or in the case where they need to react to the whole environment, not just interacting with a human actor (cf. Chapter 7).

Even in the case where interaction with human actors is necessary, a Raspberry Pi can easily be augmented with various sensors and input devices. The flexibility and versatility of the hardware, in stark contrast with the closedness of the smartphone design, offers some very interesting options for the design of hardware in distributed music systems.

2.2 Software Tools

We present the main software tools that are extensively used over this thesis. This include: the *Web Audio API* in javascript that we use for audio processing, the *soundworks* framework that is the base framework for all of our distributed applications, the *sc-components* library of custom web components for creating interfaces and the *dotpi-manager* application for managing and monitoring Raspberry Pi computers.

2.2.1 Web Audio API

The *Web Audio API* is a *JavaScript* API that can be used to process and synthesize audio in the web browser beyond the simple playback of audio files provided by the HTML `<audio>` element. The *Web Audio API* was first defined by a specification document in 2011

and has since been adopted as a recommendation by the World Wide Web Consortium (W3C). It is now supported by the wide majority of web browsers currently available[6].

6: https://caniuse.com/audio-api

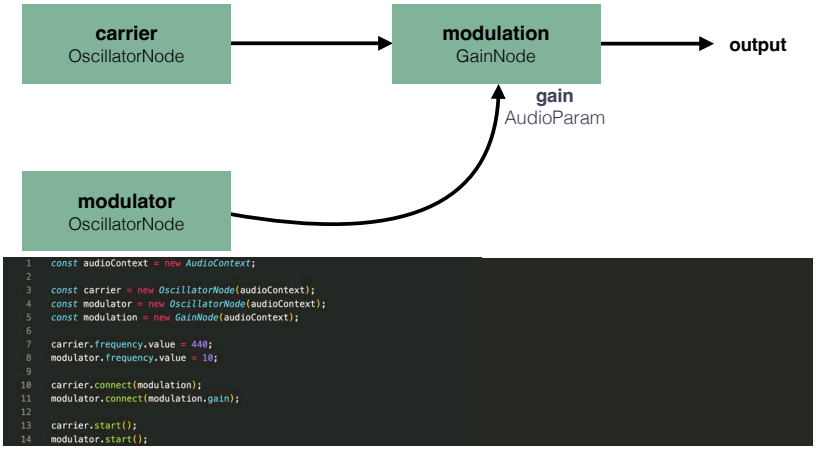
The syntax of the *Web Audio API* is based on a concept of audio routing. Various *audio nodes* can be created and connected to each other so as to form an *audio graph* (much like in visual programming languages such as *Max/MSP*). All these audio nodes belong to an *audio context* that manages the creation of nodes and the audio processing. While the audio processing is handled by audio nodes, direct scripting operation on audio data can also be performed.

Multiple audio nodes are available including the `GainNode` which multiplies the signal by a gain, the `OscillatorNode` which can generate a periodic (sinus, triangle, square or sawtooth) signal, the `BiquadFilterNode` which implements various types of low-order filters or the `DelayNode` which can be used to delay a signal.

Audio nodes can possess inner parameters, called *properties*. For instance, the `OscillatorNode` has three properties: `frequency` which determines the frequency of the oscillator in hertz, `detune` which determines the detuning of the frequency in cents, and `type` which determines the type of waveform used by the oscillator (either 'sine', 'square', 'triangle' or 'sawtooth').

Some of these properties representing audio-related parameters, such as `OscillatorNode.frequency`, are implemented as `AudioParam`. An `AudioParam` possesses its own set of methods that can be used to automate its value to make it follow a linear or exponential ramp or any determined piecewise linear curve. An `AudioParam` may also be controlled by another audio node. For instance one can connect the output of an `OscillatorNode` to the frequency property of a `BiquadFilterNode` to control it with a low-frequency oscillator (LFO). There are two types of behavior for `AudioParam`: *a-rate* `AudioParam` whose value is modified at audio rate (at each sample of the audio signal) and *k-rate* `AudioParam` whose value is kept constant for each block of audio data processed (i.e. 128 frames).

While multiple audio nodes are available to perform basic audio processing, it is also possible to perform direct processing on audio data. The first option, now deprecated, is to use a `ScriptProcessorNode` which provides the option to execute a




```
const audioContext = new AudioContext;

const carrier = new OscillatorNode(audioContext);
const modulator = new OscillatorNode(audioContext);
const modulation = new GainNode(audioContext);

carrier.frequency.value = 440;
modulator.frequency.value = 10;

carrier.connect(modulation);
modulator.connect(modulation.gain);

carrier.start();
modulator.start();
```

Figure 2.2: A simple example of amplitude modulation using the *Web Audio API*. A carrier oscillator's output is connected to a `GainNode`. The gain node's gain value is then modulated at audio rate by another `OscillatorNode` by connecting its output to the *gain* `AudioParam` of the gain node. The corresponding *JavaScript* code is displayed below.

script on each block of audio data. However, because the code runs in the main thread rather than in the audio thread, this method can create performance issues and audible discontinuities in the rendered audio signal. The other option is to create an `AudioWorkletNode`. This node can be used to exchange data with an `AudioWorkletProcessor` which performs audio processing in a dedicated high-priority thread.

The *Web Audio API* also handles the processing of audio files that can be stored in a `AudioBuffer` object and played using the `AudioBufferSourceNode`. It also provides the option to interact with multimedia input devices (for instance the internal microphone of the device or an external sound card) using the `MediaStreamAudioSourceNode` (for receiving input from a media stream).

The *Web Audio API* is largely used in all the projects of this thesis. Whenever we describe an audio processing task, such as playing an audio file, creating a synthesis engine, analyzing audio data, it is performed using the *Web Audio API*. The *Web Audio API* is unfortunately only available in the web browser. In the context of distributed music systems, this proved to be a major obstacle to the interoperability with devices using web technologies outside of the web browser such as Raspberry Pi computers.

To solve this, a *Rust* implementation of the *Web Audio API* has been developed along with a *Node.js* library named `node-web-audio-api` that provides *JavaScript* bindings of the *Rust* library[87]. The *Node.js* bindings provide an API that is completely similar to the *Web Audio API*. This allows developers to reuse large sections of code and to create Web Audio components that can run

[87]: Matuszewski et al. (2023), 'The Web Audio API as a Standardized Interface Beyond Web Browsers'

in both environments. This was an essential step in the adoption of nano-computers in this thesis (cf. Chapter 6 and 7).

2.2.2 Soundworks

soundworks[7] is a *JavaScript* framework for developing web-based distributed applications [88]. *soundworks* provides an API that handles the creation of both web browser and *Node.js* clients, communication between these clients and a central *Node.js* server through web sockets, creation of distributed states shared across the server with a subscription model and various plugins to extend the main functionalities (cf. Figure 2.3).

7: `soundworks.dev`

[88]: Matuszewski (2020), 'A Web-Based Framework for Distributed Music System Research and Creation'

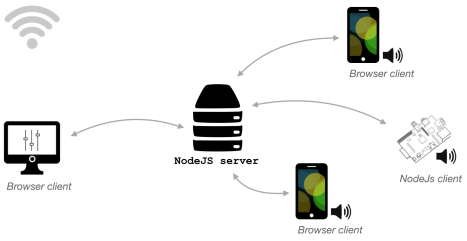


Figure 2.3: Standard architecture of a *soundworks* application.

While agnostic to the type of distributed application it is used for, *soundworks* is particularly geared towards the development of musical applications and fosters rapid prototyping practices that are well suited to the design of artistic applications. For these reasons, *soundworks* is extensively used in this thesis and is the backbone of all the projects presented in the second section of this manuscript. We present the main functionalities of *soundworks*.

Shared States

One of the main features of *soundworks* is to create *shared states*. Clients may attach to any shared state created on the server and therefore receive any update made to a shared state. Communication of updates is handled through WebSockets. The creation of a shared state can be performed either server-side or client-side. Creating a shared state first goes through the definition of a template *JavaScript* object called a *schema* that defines the name, type and update mechanism of data stored in the shared state (cf. Figure 2.4). Schemas are then registered to the server by providing an identifier and a shared state using this schema can then be created by using this identifier.

```
export default {
  name: {
    type: 'string',
    default: null,
    nullable: true,
  },
  color: {
    type: 'string',
    default: '#000000',
  },
  sourceState: {
    type: 'integer',
    default: null,
    nullable: true
  },
  playing: {
    type: 'boolean',
    default: false,
  },
  volume: {
    type: 'float',
    default: 0,
    min: -70,
    max: 0,
  },
  detune: {
    type: 'float',
    default: 0,
    min: -12,
    max: 12,
  },
  presets: {
    type: 'any',
    default: true,
    nullable: true,
  },
}
```

Figure 2.4: A *soundworks* schema that defines the template of data stored in a shared state. Each entry defines the identifier of

Clients can then attach to this shared state using the `client.stateManager.attach` function which returns a `SharedState` object. Value of a shared state attribute can then be accessed using the `sharedState.get('attribute-name')` command and updating attributes values can be performed with the `sharedState.set(newValues)` where `newValues` is an object containing name of attribute/value of attribute pairs. Callbacks to be executed upon updates can be registered using the `sharedState.onUpdate` function. See Figure 2.5 for an example.

Several shared states following the same schema can be created. Clients can then attach to a single shared state using the `attach` method by providing the state's ID or can attach to all the shared states following the same schema by using the `getCollection` method. In that case, a `SharedStateCollection` is returned which provides options to get and set values of attributes for all the states in the collection. An `onUpdate` method is also available and triggered upon any updates of any shared state in the collection. Such collections are especially useful when creating a control interface that aims to control and monitor multiple clients or devices.

In the following of this manuscript, all communication in-between clients and between clients and server is assumed to

Client A

```
// Creating a new shared state using the 'controller' schema registered on the server
const controller = await client.stateManager.create('controller');

// Setting value of the 'playing' and 'volume' attributes of this shared state
controller.set({
  playing: true,
  volume: -10,
});
```

Client B

```
// Attaching to the 'controller' shared state
const controller = client.stateManager.attach('controller');

// Accessing value of the 'volume' attribute
console.log('Volume (in dB)', controller.get('volume'));

// Reacting to updates of the 'volume' attribute to change a gainNode gain value
controller.onUpdate(updates => {
  gainNode.gain.value = updates.volume;
});
```

Figure 2.5: Examples of how to use a shared state in *soundworks* to exchange data between clients. A shared state is created in client A following the schema presented in Figure 2.4 and the values of some attributes are modified. Client B attaches to this shared state and is able to access values of the state's attributes and to react to updates.

happen through *soundworks* shared states.

In the following we detail some typical use cases for shared states. In these examples we assume the existence of two clients: client A which is mostly intended to be a controller interface accessible in a web browser typically by a musician or experimentalist and client B which is intended to be a player interface accessible either in a web browser to be used by a generic audience member/participant or on a raspberry pi computer for autonomous use.

Triggering an event remotely: Shared states can be used to trigger an event on a remote device. To do this, one can create a simple schema with a boolean attribute called 'trigger'. One then creates a shared state with this schema on client A. One can then attach to this shared state on client B and write a callback function reacting to updates of the 'trigger' attribute using the `sharedState.onUpdate` method to trigger the playing of a sound file using the *Web Audio API* for example. Back to client A, one can then render a simple interface with a button that when pressed, will change the value of the 'trigger' attribute using the `sharedState.set(trigger: true)` command, thus triggering the playing of the sound file on any device connected to client B.

Controlling a synthesizer: Shared states can be used to store the parameters of an audio synthesizer. On client A, one can create the interface of an audio synthesizer with various parameters (volume, pitch, type of waveform, envelope values, etc...). A schema corresponding to these parameters (including eventual min/max and default values) can be written to create a shared state storing values of the interface. These values will then be accessible

and updated on client B and used as parameters in an audio synthesizer created with the *Web Audio API*. Client A can then be accessed on a computer browser and client B on multiple devices simultaneously (smartphone, Raspberry Pi). A single interface can therefore control multiple synthesizers at the same time.

Note that the intended behavior will depend greatly on which client the shared state is created. If created on client A, all clients B will react to the same values and play the same sound, however opening another client A will result in conflicts as both client A possess independent shared states. If created on client B, all clients B will possess their own independent set of parameters and the control interface will necessitate a separate interface for each client. Finally, if the shared state is created on the server, all clients A will attach to the same shared state and thus share the same values. In such case, any change on a client A interface will automatically modify the other client A interface accordingly

Displaying a list of clients: Assume that each client B creates a shared state X that contains information and data such as the name of the user or their answers to a questionnaire. On client A, one can attach to the collection of all shared states X to display on an interface the list of currently connected clients and values of their information. The interface will be able to be updated automatically in cases of new connection or disconnection or update of a client's value.

Plugins

Several plugins are available as part of the *soundworks* API to extend the framework's core functionalities, especially in the case of audio applications. We present a few plugins that are extensively used in this thesis.

The Platform Init Plugin Several processes in the browser are secured and require a user interaction (e.g. a click/touch action) to be activated. In particular it is the case for resuming an audio context in the *Web Audio API* but also for accessing camera or microphone streams. The *platform init* plugin provides a way to simplify this process by displaying a splash screen upon connection

to a client's address. This screen requires a user interaction (a click) before initialization of the client.

The Sync Plugin In the case of distributed systems, as all nodes in the network are independent from each other, they naturally possess their own definition of time that differs from the other nodes [6]. Nonetheless, the existence of a shared clock among all the nodes in the system is an essential element in most distributed musical applications to trigger musical events at the same time or to share a common tempo or beat. Several libraries and tools are dedicated to generating a shared clock among a distributed system of devices such as Ableton Link[8].

[6]: Van Steen et al. (2016), 'A Brief Introduction to Distributed Systems'

8: `https://ableton.github.io/link/`

The *sync* plugin in *soundworks* is dedicated to this task[89]. Upon initialization by the server, it creates a master clock that returns the time since the start of the process. The link between this master clock and each client's local clock can then be performed using the plugin's API with the function `getLocalTime` which returns a time relative to the local clock from a "synchronized" time relative to the master clock and the function `getSyncTime` which performs the reverse operation. Most of the time in this thesis, the local clock on each client will be defined using the `audioContext.currentTime` property so as to synchronize the different audio clocks.

[89]: Lambert et al. (2016), 'Synchronisation for Distributed Audio Rendering over Heterogeneous Devices, in HTML5'

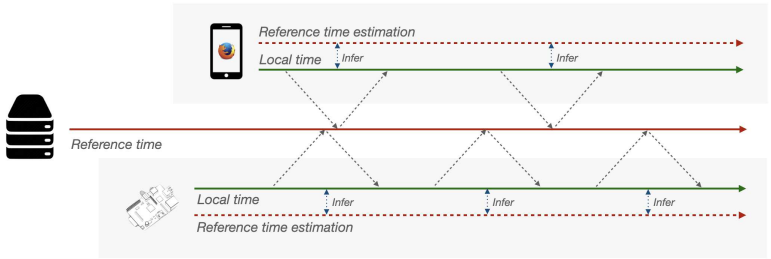


Figure 2.6: Link between the "Reference time" created by the server and the Local time on each client with the *soundworks* Sync plugin. Picture taken from `https://soundworks.dev/tutorials/plugin-sync.html`

This can be used to easily trigger a single musical event at the same time on all clients by sending a "synchronized" time to all clients, converting it to each client's local time using the `getLocalTime` function and feeding this local time to the *Web Audio API*. Synchronization is guaranteed up to an accuracy of 5-10ms[89] which is small enough to synchronize musical events. Unfortunately, this synchronization system does not compensate for the audio output latency of the client's sound card which may greatly vary depending on the device used.

The Filesystem Plugin The *filesystem* plugin is used to watch a directory on the server and to perform several operations on it from any client. The method `getTree` returns the tree structure of the directory with subdirectories and files. The plugin API provides other methods to create subdirectories, rename or delete files and directories and to write data in a file. A `onUpdate` method can be used to register a callback that is executed whenever a change happens in the watched directory's structure.

This plugin is particularly useful to watch over directories containing a soundbank of audio files or log files and to render these directories' structure in interfaces. It is used in this fashion in most of the projects of this thesis.

The Scripting Plugin The *scripting* plugin is used to distribute *JavaScript*scripts over the network. New scripts can be created using the `plugin.create` method and updated using the `plugin.setValue` method. This creates a `SharedScript` object to which clients can attach and receive updates. Shared Scripts are automatically transpiled and bundled any time they are updated. After attaching to the shared script, clients may use the `sharedScript.execute` method to dynamically import the script at runtime.

This plugin is especially useful to provide customizable elements in a digital music instrument. For instance a text editor might be available in a web page for a user to write scripts. These scripts will then be broadcasted to clients to be used as a block in an audio process or to customize a visual interface for instance. Because script updates are automatically applied at runtime, changes will be immediately applied which makes it especially useful for prototyping in creative workflows. Such examples of use cases are showcased in Section 2.3 to provide user defined audio effects or in Chapter 4 to provide customizable synthesizers or routing of messages.

2.2.3 sc-components

To support the development of graphical user interface in the browser, it became essential to develop a library of HTML components[9] adapted to musical control interfaces. This library called

9: https://www.webcomponents.org/specs

sc-components[10] provides a large variety of web components ranging from simple buttons to keyboards and waveform display. This library was enriched over the time of this thesis throughout new use cases and in the following I detail some personal contributions to this library.

10: `https://ircam-ismm.github.io/sc-components/`

Color Picker Component

I added the `sc-color-picker` that mimics the behavior of the default `<input type="color">` element in HTML but with a graphical display that is more coherent with the library's style (cf. Figure 2.7). Upon pressing the button, the color picker of the browser is displayed and a callback function is executed when changing colors. In this thesis this component is used in Chapter 6 to provide customization options in a user interface or to set the color of a LED connected to a *dotpi* unit.

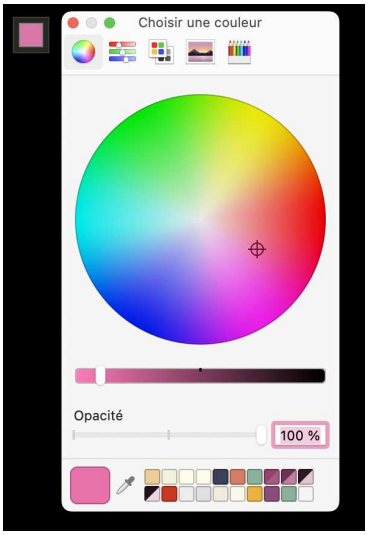


Figure 2.7: The `sc-color-picker` element with the Firefox color picker menu opened.

Loop, Record and Transport Buttons

I added a set of buttons for representing transport of audio files that includes : a play/pause/stop set of buttons (`sc-transport`), a record button (`sc-record`) and a loop button (`sc-loop`) (cf. Figure 2.8). A callback function is executed upon any change of state of the buttons. For instance, in this thesis, these buttons are used to start/stop a synthesis engine or to start a recording process.

Figure 2.8: From left to right: the `sc-transport` element in the "pause" state, the `sc-record` in the active state, the `sc-loop` component in the active state.

Status Component

The `sc-status` was developed to display the status of an operation between two states represented by the color red or green (cf. Figure 2.9). The color of the component can be set with the `active` attribute. For example, in this thesis, this component is used to indicate whether an audio file is loaded or not.

Figure 2.9: Two `sc-status` elements in two different states.

MIDI Component

When developing digital musical interfaces in the web browser, it became essential to develop a web component that enables users to operate the library's components using MIDI devices. The `sc-midi`

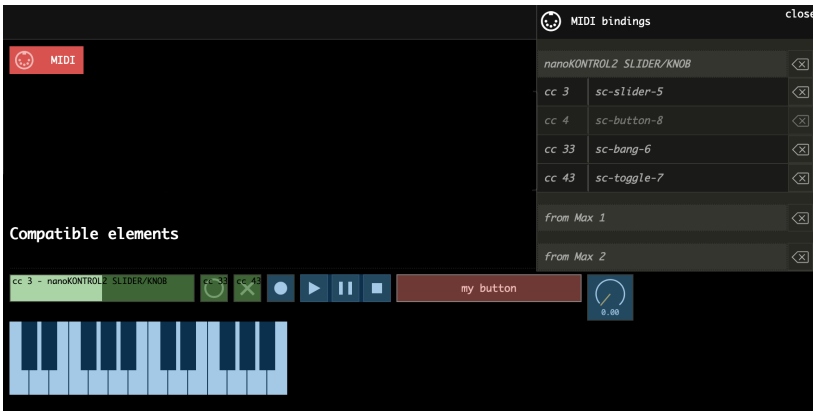


Figure 2.10: The `sc-midi` element (top left) activated. The list of MIDI bindings is displayed in the top right. Assignable components are displayed in blue. Components already assigned are displayed in green. The button in red is currently selected and can be assigned to a MIDI CC by sending a message.

component that I helped develop can be used to define MIDI bindings with the buttons, slider, dial and keyboard components of the library (cf. Figure 2.10).

This component makes use of the *Web MIDI API*[90] to request access to MIDI devices connected to the device and to receive their inputs. When activating the component, all assignable components on the page turn are colored in blue. The user can select a component they wish to assign to a MIDI CC message, in which case it will be colored in red. Sending a MIDI CC message when selecting a component will create a binding that will be displayed in the MIDI bindings list in the top right corner of the browser window. This list can also be used to delete existing bindings. Components that are already assigned to a CC message will be colored in green.

[90]: (), *WebMidi API Specification*

MIDI bindings are saved in the browser's `localStorage` and can therefore be retrieved between sessions.

In this thesis, the `sc-midi` component represented an important step in the development of the *Simone Solo* interface (cf. Chapter 6).

Waveform Component

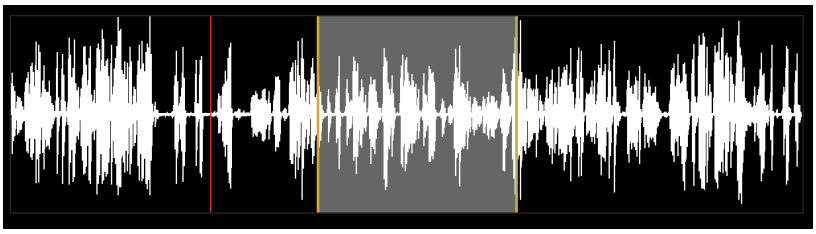


Figure 2.11: The `sc-waveform` displaying the waveform of an audio file. The red vertical line is the cursor. The grayed zone is the selection that can be moved over the waveform. Yellow vertical lines are handles that can be moved to shorten/lengthen the selection

The `sc-waveform` component was developed to display and interact with the waveform of a sound file (cf. Figure 2.11). This compo-

nent accepts an `AudioBuffer` as an input. The given `AudioBuffer` is first reduced to a single channel and then normalized. Then, for each displayed pixel, we compute the bounds of the waveform of the signal over the corresponding time segment.

The component also offers the option to display a cursor at a set position. A selection can also be drawn with the mouse over the waveform. This selection can then be moved over the waveform and lengthened/shortened using handles at its extremities. Changing the selection triggers a callback returning the limits of the selection.

This component has been developed as part of this thesis from earlier prototypes featured in the *Simone* application (cf. Chapter 5) and is used in the second version of the /simonesolo application (cf. Chapter 6).

2.2.4 Dotpi Manager

As the number of *dotpi* units used in our applications increased, it quickly became essential to develop a software solution to manage an ensemble of *dotpi*s and perform routine tasks such as executing commands or transferring programs.

The *dotpi manager* application was developed using *soundworks*. Upon starting the server, the manager interface is available in a web browser (cf. Figure 2.12). A daemon installed on every *dotpi* automatically connects to the manager server.

The interface is divided into 3 panels. In the bottom left panel, a list of *dotpi* connected is displayed by their hostname. On the same line, four `sc-status` display whether the *dotpi* is connected to the server and to the internet or not, whether data is being transferred to the *dotpi* or not, and whether a process is currently running or not. Next to it, two toggle buttons can be used to (de)activate the execution of command on this *dotpi* and the display of its command line printing. Finally, a bang button can be pressed to trigger the playing of a sound (either noise burst or sweep) on the *dotpi* for quickly checking that the audio is working. The list of *dotpi* clients can be filtered by their hostname using a text field. In the case of disconnection of a client, they are still displayed in the list.

In the top left panel, options are provided to execute commands

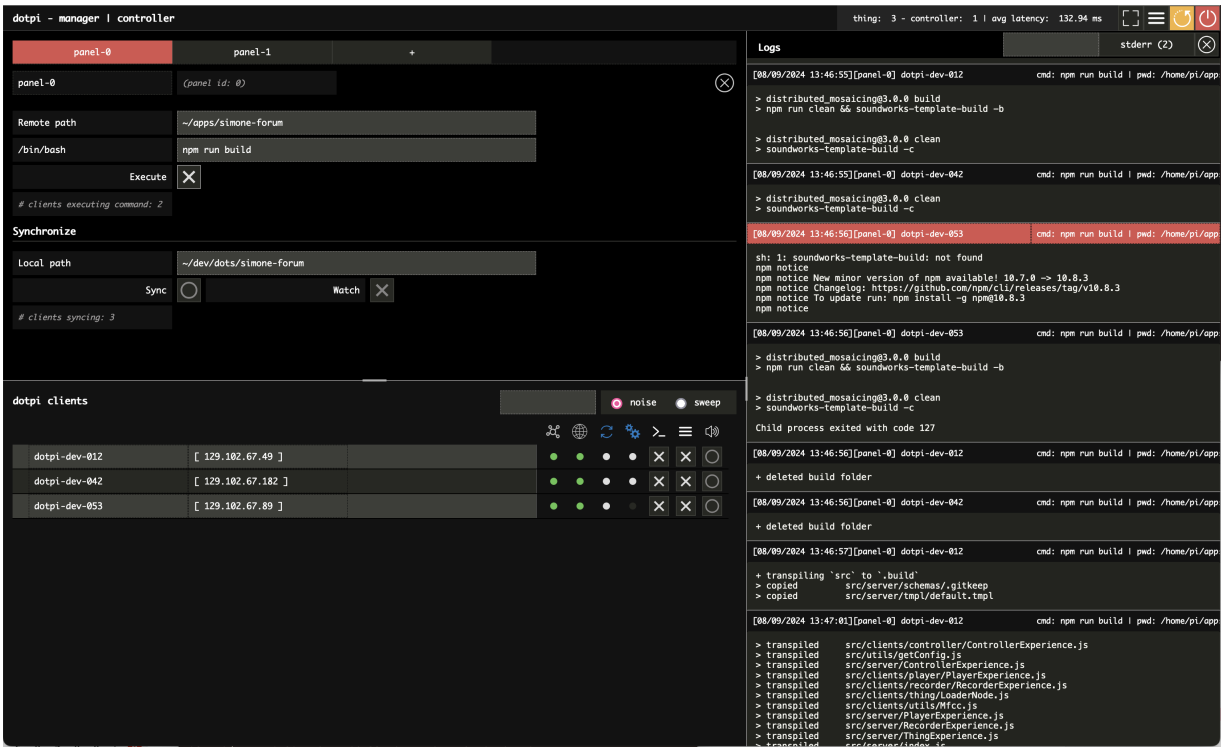


Figure 2.12: The *dotpi-manager* interface in a web browser.

remotely on the clients and to synchronize a local folder with the clients. The first text field is used to define the remote path to use when executing commands and for transferring data. The second text field is used to write a command to be executed on all clients (except for those whose command execution is untoggled). Command execution is triggered with a toggle button. The last text field in the "Synchronize" section is used to define the local path of a folder (on the server) to synchronize it on the remote path. A bang button below triggers the synchronization and transfer of data. Finally, a toggle button can be activated to watch any change on the local folder to synchronize it automatically on the clients. This is especially useful for prototyping applications as any change in the clients' code is automatically broadcasted to currently connected *dotpi*s.

The right panel displays command line logs of all *dotpi*s. Logs can be filtered by client name or by only displaying error messages. Finally, in the top right corner, two buttons can be used to reboot or shutdown all currently connected *dotpi*s.

The *dotpi-manager* is used in Chapters 6 and 7 whenever we had to use *dotpi*s. It is used both during development context for easily prototyping on *dotpi* units and in production context to

monitor and manage execution of commands on large ensembles of *dotpi*s.

2.3 An Example Prototype: Distributed Audio Effects

In this section we present a prototype co-developed with B. Matuszewski that showcases most of the tools presented in this chapter. This prototype was presented in [24].

[24]: Matuszewski et al. (2023), 'Rapid Prototyping of Distributed Musical Things Using Web Technologies'

This application developed within *soundworks* is dedicated to the prototyping and deployments of audio effects on a fleet of devices. Each device is equipped with a microphone and speakers (it could be a smartphone or a Raspberry Pi computer). Audio scripts can be written using the *Web Audio API* in an interface available in a web browser. Scripts are then automatically distributed to the network of devices in real time. Audio processing defined by the scripts is then executed locally on the devices.

The web interface is composed of two main panels (cf. Figure 2.13). On the left panel, a text editor (a `sc-editor` component from the *sc-components* library) allows the user to create and edit audio scripts using the *Web Audio API*. Users need to define the audio graph between the input and the output nodes of the device. Upon saving, the script is automatically shared on the network to be executed by the connected devices using the *scripting* plugin in *soundworks*. Scripts are saved on the server in a "scripts" folder. The file tree of this folder is displayed in the application using the `sc-filetree` component which can be used to rename and delete scripts directly from the interface.

On the right panel, for each device connected a line is displayed that enables the user to set which script is currently executed on the device, change the input and output gain of the device and monitor the input (dry) signal and output (wet) signal (using the `sc-signal` component).

Upon connection to the server, a client is attached to the *scripting* plugin to be able to access the remote scripts and receive their updates. A "shared state" is also created that stores data specific to this client such as the script to be used, the input or output gain

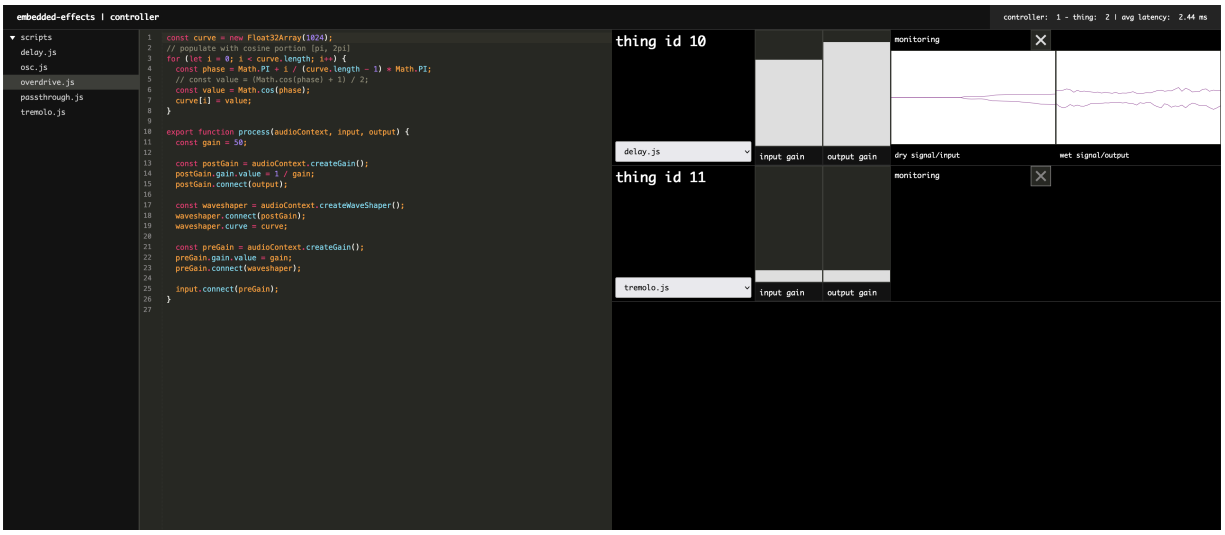


Figure 2.13: The web interface for the distributed audio effects application. Left panel shows a script written using the *Web Audio API* in an editor. The right panel shows that two clients are currently connected. The first client (id 10) is currently executing the "delay.js" and monitoring of the input/output signal is displayed.

or whether to activate monitoring on the interface. Each controller interface automatically attaches to any new client's shared state to receive or publish updates to this shared state. For instance, changing the script selected by a client on the interface triggers a callback function that changes the value of the `selectedScript` attribute in the shared state.

Using shared states and the *scripting* plugin means that the web interface can be simultaneously used on multiple devices at the same time without concurrency issues. For example, a `inputGain` float value is stored in a client's shared state. When a user changes the input gain of a client using the corresponding slider on the interface on their laptop, a callback is triggered that updates the *inputGain* value on the shared state of this client. Upon receiving the update, the client then executes a callback function to modify the gain value of its input `GainNode`. At the same time, the controller interface opened on the user's tablet also receives the updates and triggers a callback to update the display of the slider to match with the new updated value.

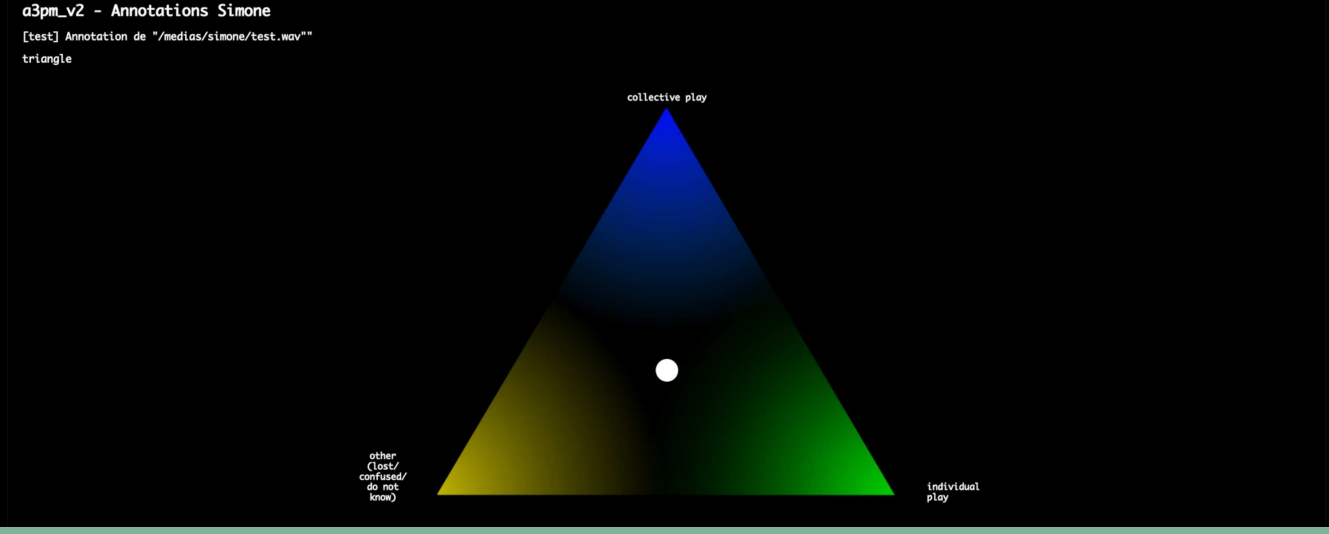
The audio graph on the client is entirely defined using the *Web Audio API*. Likewise, the user-defined audio scripts are written using the *Web Audio API*. Thanks to this, these scripts can either be executed by both web browser clients and on embedded devices using the `node-web-audio-api`. This allows users to use any heterogeneous configuration of devices including smartphones with internal speakers, Raspberry Pi with portable speakers or laptop

computers connected to larger speakers.

When a script is selected for a client, this client automatically fetches the script from the *scripting* plugin and the script is processed and executed by connecting the input node of the script to the input microphone of the device and the output node of the script to the audio destination of the device. In the case that a script currently executed on a client is updated, this client automatically transitions to the new version with a crossfade.

This prototype showcases most of the benefits of working with the technologies presented in this chapter and used in this thesis. First, it enables quick prototyping and fosters a trial-and-error workflow that is beneficial in most creative situations. One can simply change a delay value or add a filter to the audio graph and the result will be instantly propagated to the network of devices and automatically executed. Moreover, since the interface is simultaneously accessible on multiple devices, it is well adapted to various workflows. Second, it provides an interface that is accessible by users with different skills as all the internal processes are hidden. A musician unfamiliar with *JavaScript* could easily use the system after being provided with the basics of the *Web Audio API* syntax (that is not so different from other audio programming languages) or even, given some evolution of the application, use higher-level audio components such as Web Audio Modules [91].

[91]: Buffa et al. (2022), 'Web Audio Modules 2.0'



3 A^3PM: A Distributed Application for Self-Annotation of Musical Performance

This chapter introduces A^3PM, a distributed application that aims to provide the technical support for the deployment of the self-annotation methodology. A^3PM was developed using a third-person perspective and user-centered design methodology by identifying some of the key features and specifications needed for the use cases that were envisioned.

In this chapter, my contributions lie in the participation to the development of A^3PM, in various additions to this application, in the continuous monitoring of its use in various experimental contexts and to a port of the application to the latest version of the *soundworks* framework. Through this participation, I also participated in multiple research projects published in journals and conference of musicology and psychology [19, 20]

[19]: Golvet et al. (2021), 'With, against, or Without?'
[20]: Majeau-Bettez et al. (2023), 'Tracking Auditory Attention in Group Performances'

3.1 Self-Annotation of Musical Performance

Self annotation of musical performance is a methodology for the study of musical performance that involves having musicians perform and immediately after to ask them to continuously annotate the recording on a selected aspect of their performance.

The technique of self annotation has been developed over the past years at Ircam by Clément Canonne *et al*. (This chapter draws some elements from his text on self-annotations that will be

published in the forthcoming book [92]). It is mainly inspired by the method of "self confrontation interview" developed by Jacques Theureau [93] in which participants are confronted to recordings and material traces of their activity in order to reenact the act of performing and to access a verbal report of the pre-reflexive conscience from the participant. This method has been extensively used in the field of musicology and in the analysis of the act of composing a musical piece by Theureau and Donin [94].

[92]: Canonne et al. (Forthcoming), *New Methods and New Challenges in Empirical Musicology*

[93]: Theureau (2010), 'Les Entretiens d'autoconfrontation et de Remise En Situation Par Les Traces Matérielles et Le Programme de Recherche « Cours d'action »'

[94]: Theureau et al. (2006), 'Comprendre Une Activité de Composition Musicale'

Unlike self confrontation interviews, self annotations aim to provide access to non-verbal and continuous dimensions of a selective aspect of their experience of performing. When analyzing group performances, focusing on non-verbal and continuous reports has the benefit of allowing equal comparison between performers' reports as it eschews potential language biases and provides data over the whole performance, which constitutes an advantage of this methodology over more widespread ethnographic methodologies. One advantage of self-annotations over other experimental ("in the lab") methodologies is that it is non-intrusive and guarantees ecological validity. Indeed, before annotating, musicians only need to perform as they usually do without added equipment (such as pedals to press) or specific instructions.

3.2 Motivations and Design

To support the task of auto-annotation, we developed a distributed application called A^3PM (which stands for "Application pour l'Auto Annotation de la Performance Musicale" or, in English, "Application for self-annotation of musical performance). While the application was first designed as a tool that satisfies a minimal list of specifications, it evolved as a more modular and autonomous application, beyond its initial intended use. The addition of new features over time brought up the possibility of new use cases and extended the theory of self-annotation in a movement of co-evolution[95] (cf Section 3.4).

[95]: Dorst (2019), 'Co-Evolution and Emergence in Design'

Developing a distributed system for self-annotation was not an arbitrary decision. While a non-distributed application could have been imagined for this task, the distributed nature of our system provides multiple advantages.

First, by designing a distributed system in which a server broadcasts the recording to any number of participants connected to a web application, we minimize the time needed to setup the annotation system as the only step needed for the experimenter is to copy the audio files to be annotated on the server while all the participants are able to connect to the application and to annotate the files simultaneously. This guarantees that the task of self-annotation is performed by musicians as quickly as possible after the performance, which is a crucial point to avoid too much deterioration of their memories of the performance. This needs to be the case for any number of musicians involved, even for large groups (indeed one of the first applications of A^3PM was for a large group of 16 musicians [96]).

[96]: Goupil et al. (2020), 'Musical Coordination in a Large Group without Plans nor Leaders'

Second, developing a web-based interface for self-annotation accessible on any device with a regular web browser means that the task of self annotation can be performed anywhere in the field by bringing very lightweight equipment (a laptop and the equipment to create a local network) and by making musicians use their smartphones or tablets to annotate. This helps preserve the ecological validity of the self-annotation methodology (which, as we mentioned, is one of its main advantages) as the procedure can be carried in a place familiar to the musicians, for instance their rehearsal place.

To guarantee that participants are able to perform the self-annotation task smoothly and autonomously, the user experience of A^3PM is designed as a series of screens linked by a state machine that guides participants through the procedure. The different screens implemented contain a screen to configure their name or ID, different explanation screens with texts configurable by the experimenter and a way to provide a test interface for participants to familiarize themselves with the annotation interface.

Moreover, this architecture as a state machine allows us to define multiple types of annotation interface in order to adapt to different types of research questions and to develop new ones as new use cases emerge. At the time of writing, A^3PM provides 3 types of interface : a slider (for intensity-based questions or annotation between 2 poles), a triangle (for annotation between 3 poles) or a square (for annotation between 4 poles). Moreover, depending on the context and the research question, the triangle and the square interface are adapted to annotation on a continuous

space or on a discrete/categorical space.

Using A^3PM in the field, it gradually appeared that a visualization tool would be a valuable addition to overcome some of the application's shortcomings. First, some experimenters or users may lack the necessary skill to parse the annotation data files and to compute plots and statistics to visualize data. Second, it would provide a less abstract experience to participants by showing them immediate feedback on the task they performed and by fostering discussion after the annotation task.

3.3 The A^3PM Application

A^3PM is a *soundworks* application composed of: a *Node.js* server, a series of web interfaces for self annotation of performance, a controller interface to provide feedback on participants to the experimenter during the annotation task and a visualization interface that allows to plot graphs of data collected throughout experiments.

3.3.1 Projects and Configuration Files

Before launching the application, a JSON file must be defined to configure an annotation project (cf. Figure 3.1). This file details the different tasks that the participants will perform in the study. A `Project` is thus defined as a series of `Tasks`. Each task has a defined type and may contain multiple sound files to annotate. The configuration file must contain the following information :

- The name of the project
- The language of the project ('fr' for French or 'en' for English)
- A list of tasks configuration.

Each of the tasks configurations is an object that contains the following information:

- `annotationType`: The type of annotation task. This defines the interface used for annotation. At the time of writing, the application offers the choice between a slider interface (for annotation between two poles), a triangle (3 poles) and a square (4 poles).

- `mediaFolder`: The name of the subfolder in which to fetch audio files for this task. Multiple files can be annotated in the same task.
- `tags`: A list of arrays of tags that will be displayed on each pole of the annotation interface. The list can contain multiple arrays in case the tag placement needs to be randomized for experimental purposes.
- `testRecordings` (optional): the file name of an eventual test recording in the mediaFolder. In case a test recording is defined, participants will be brought to a test interface to familiarize with the annotation interface before starting the task.

```
{
  name: 'test project',
  language: 'en',
  tasks: [
    {
      annotationType: 'slider',
      mediaType: 'audio',
      mediaFolder: 'task1',
      tags: [
        ['quiet', 'loud'],
      ],
    },
    {
      annotationType: 'triangle',
      mediaType: 'audio',
      mediaFolder: 'task2',
      tags: [
        ['cookie', 'pie', 'ice cream'],
      ],
    },
    {
      annotationType: 'square',
      mediaType: 'audio',
      mediaFolder: 'task1',
      tags: [
        ['north', 'east', 'south', 'west'],
      ],
    },
  ],
}
```

Figure 3.1: A project configuration file in JSON.

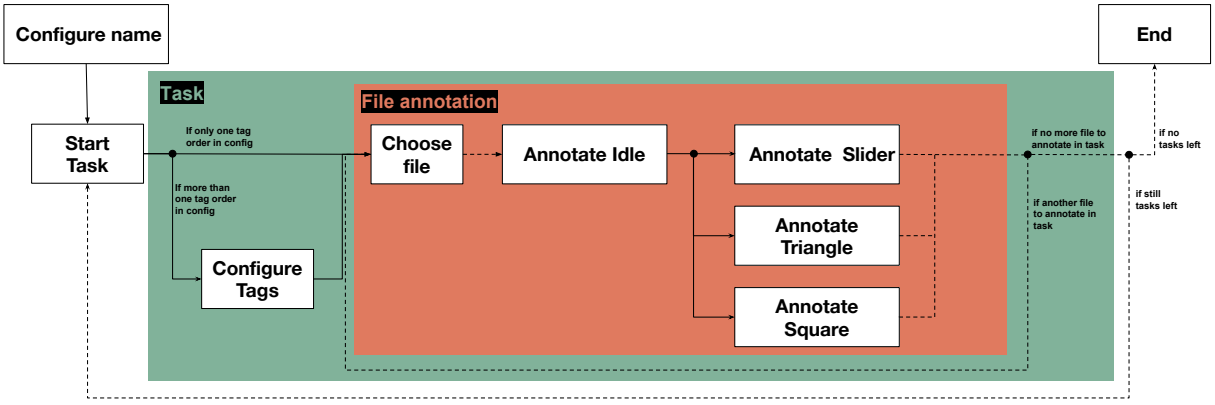


Figure 3.2: A diagram of the state machine of the web interface of A^3PM. Solid arrows indicate a user action, dashed arrows indicate a transition without user action.

3.3.2 The Web Interface for Self Annotation

From a participant point of view, A^3PM is experienced as a series of screens corresponding to different states and tasks of the current project. Each of these state is implemented as a *soundworks* `context` which allows to define `enter` and `exit` function which are executed both on the client and server side respectively upon entering and exiting the state. In our case, these functions are used to log data or to load assets. The application is accessible as a web page on any device that has a compatible web browser and connected to the local network of the server.

The state machine of the user experience is described as a diagram in Figure 3.10. First, participants are brought to a screen in which they can input their name. Upon submit, a folder is created on the server that will contain a log of the participant's actions and their annotation data.

Participants are then brought to the "Start Task" state of the first task. In the case there are multiple tag orders in the task configuration, participants may be brought to a "Configure Tags" state in which they need to select a tag order. On the next step, the application randomly selects and loads a file to annotate from the media folder given in the task configuration. In the case that a test file was selected in the task configuration, this file is selected first. The application then moves to the "Annotate Idle" state in which participants are shown a button to press whenever they feel ready to start annotating. Upon pressing the button, they are brought to the annotation screen corresponding to the annotation type given in the task configuration : either the slider screen, the

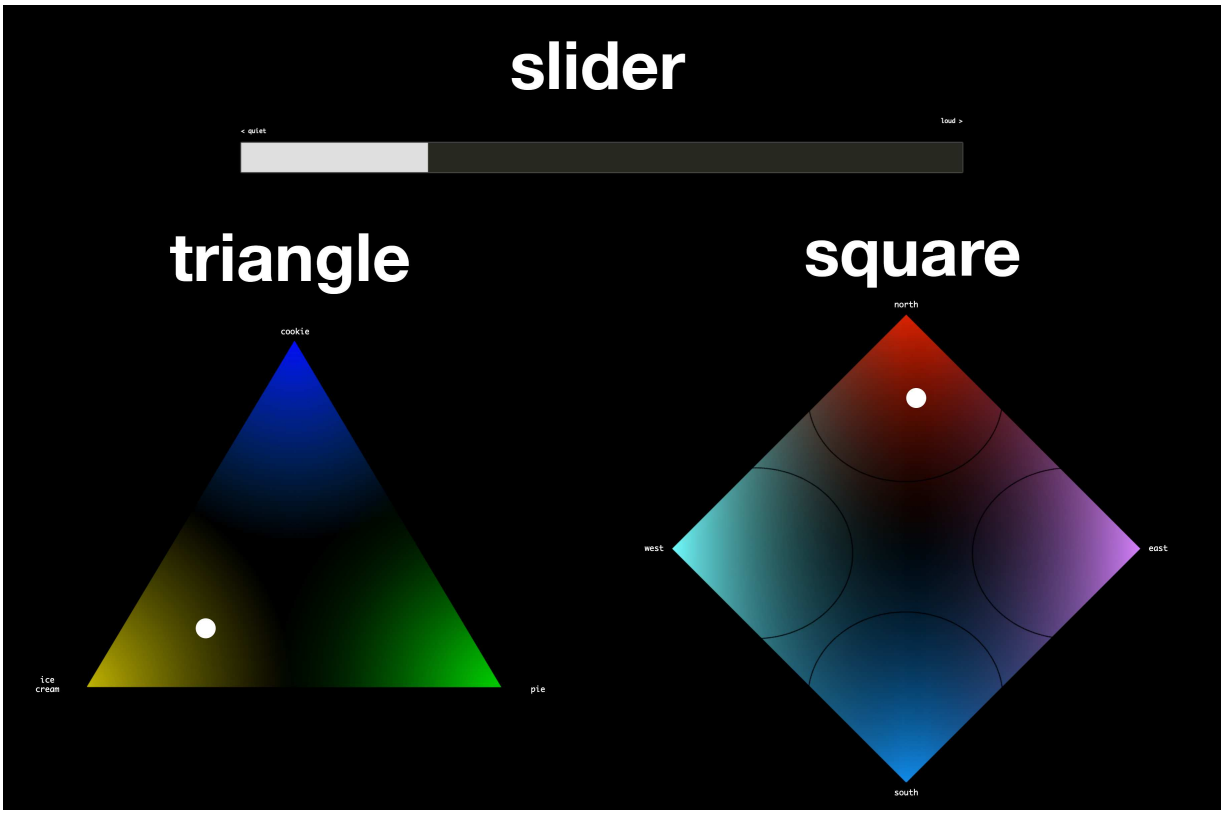


Figure 3.3: The three types of annotation interfaces available in A^3PM.

triangle screen or the square screen (cf. Figure 3.3).

On this screen, the audio file starts playing automatically and users may interact with the annotation interface. On the slider interface, participants may move the position of the slider from one extremity to another. On the triangle and the square interface, participants can move a white dot within the shape displayed. Colored zones and tags corresponding to each vertex of the figure are displayed but the dot's position is always saved as continuous position and not as categorical data. When annotating, the participant's position on the interface is periodically (every 50 ms) written in a file on the server.

At the end of the sound file, the application makes a series of verification to decide which path in the diagram to follow. If there are still files to annotate in this task, the application goes back to the "Choose file" state and participants proceed with annotating another file. Else, that means this task is finished. The application then moves on to the next task in the project configuration and goes back to the "Start Task" state. If there is no more task to perform, participants are brought to an end screen thanking them for participating.

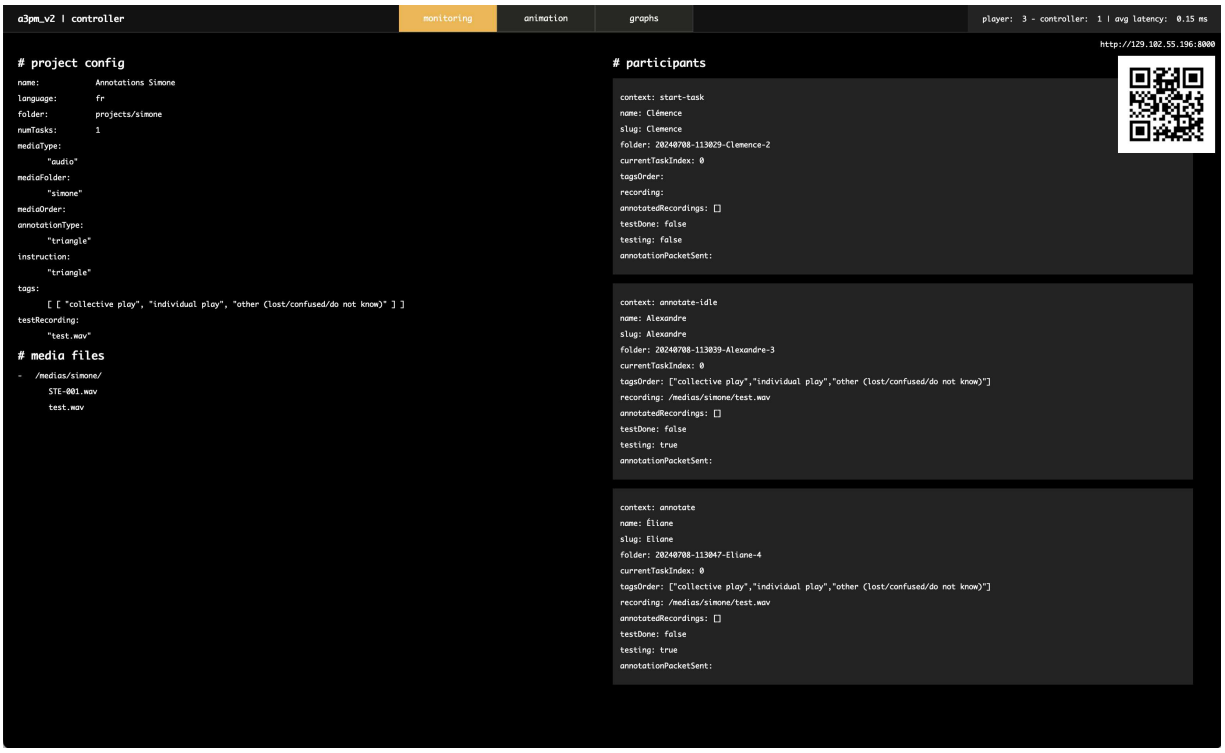


Figure 3.4: The monitoring interface in a web browser.

Between each of these steps, each participant's state is logged in the folder containing their annotation data on the server. In case of disconnection, the participant is able to reconnect to the application and restart from the step they were at.

3.3.3 The Monitoring Interface

The application provides a monitoring interface accessible in a web browser. As seen on Figure 3.4, the monitoring interface displays multiple information useful to the experimenter to monitor the conduct of the study:

The number of participants connected to the application and the average network latency (top right corner).The project configuration.The file tree of the audio files folder on the server.The current state of each participant, in particular their name, the task they are performing and the audio file they are annotating.A QR code leading to the address of the web annotation interface. This QR code can be scanned by the participants to quickly connect to the application.

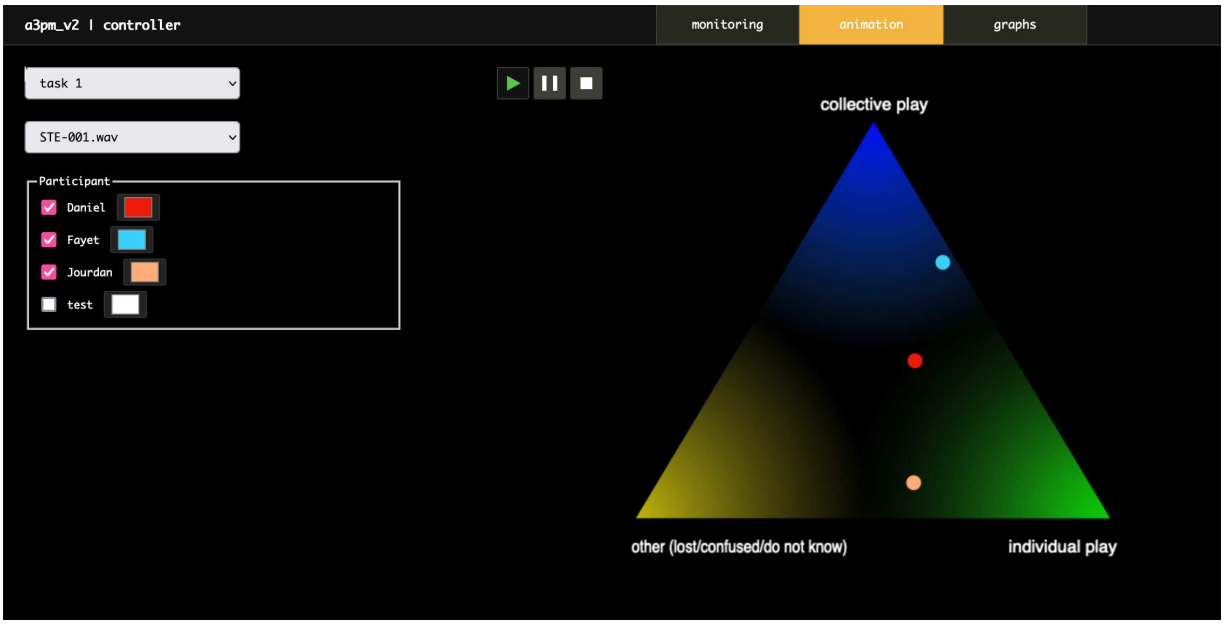


Figure 3.5: The animation tab in a web browser.

3.3.4 The Visualization Interface

Also accessible in a web browser, we developed a visualization interface for A^3PM. The visualization is divided in two tabs : First, an "animation" tab that allows to play back animations of participant's annotation over time; Second, a "graphs" tab that allows to display a timeline of the participants' annotation on a graph.

The Animation Tab

On the animation tab, a continuous animation of the participants' annotation over time can be played (cf Figure 3.5). Upon selecting the task and audio file that correspond to the data that needs to be animated, a subset of participants can be selected from the list of all participants. The annotation data of the selected participants will be superimposed and animated on the annotation interface as the sound file is playing. Next to each participant's name, a color picker allows changing the participant's dot color. The animation can be paused (and can be then resumed with the play button) or stopped (in that case it will restart from the beginning when pressing the play button).

The Graphs Tab

On the graphs tab, graphs of the timeline of participant's annotation can be plotted using the *Plotly.js* library[1] for quick visualization and comparison (cf. Figure 3.6). Using a menu, a specific participant annotation can be selected for plotting. For each data point (a timecode and a position on the interface), a color is computed depending on the zone of the interface that the point position is in (cf. the three zones of the triangle on Figure 3.3 and compare to the colors on Figure 3.6 for example). A vertical bar is then drawn at the corresponding timecode with the computed color. Graphs are displayed vertically allowing for quick comparison between multiple participants annotations of the same audio file.

1: https://plotly.com/javascript/

On the upper left of each graph, a button can be pressed to cycle between two color modes: a categorical mode (each vertical section of the graph is drawn in solid color that depends on the zone of the interface that the participant was in), or a continuous mode (each vertical section of the graph is drawn in a color that depends on the zone of the interface the participant was in but with the addition of a transparency ratio that is higher the more the participant was far from the extremity of the zone)[2]. While some experiments would ask their participants to annotate between multiple zones/categories and would thus use the first mode, others would ask participants to annotate continuously, using proximity to each extremity as a marker of nuance, thus benefiting from the second mode.

2: cf. Figure 3.6 where the first and third graphs are in categorical mode and the second graph is in continuous mode

Using available features from the *Plotly.js* library, users can zoom in and out some sections of the graphs and graphs can be downloaded as png files.

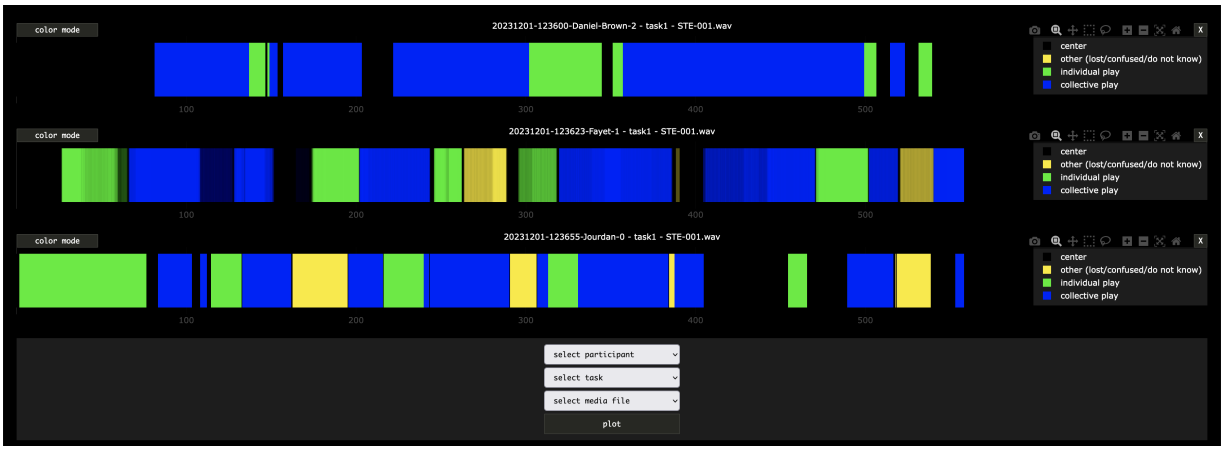


Figure 3.6: The graphs tab in a web browser.

3.4 Use Cases

The A^3PM application has been used in multiple contexts. While initially intended for musicological experiments and study of musical performances in controlled settings, other use cases emerged with the appearance of some features, especially the visualization features.

3.4.1 Use in Experimental Musicology

A^3PM has been used in numerous musicological experiments to study the psychology and phenomenology of musical performance. These uses are documented in various papers that we list below.

In a first experiment by Goupil *et al.* [96], a group of 16 musicians performing collective free improvisation from the *Orchestre des Nouvelles Créations, Expérimentations et Improvisations Musicales* (ONCEIM) ensemble used an early version of A^3PM to annotate their 20 minutes performance on a slider between one extreme manifesting the intent to "change the direction of the music collectively produced by the group" and the other extreme manifesting the intent to "support the direction of the music collectively produced by the group". Each musician was provided an iPod to perform the task.

[96]: Goupil et al. (2020), 'Musical Coordination in a Large Group without Plans nor Leaders'

In another study by Golvet *et al.*[19], duos of musicians from the Parisian free improvisation scene were asked to play 4 improvisations running from 5 to 10 minutes. Immediately after playing, they listened back to their improvisations in a randomized order

[19]: Golvet et al. (2021), 'With, against, or Without?'

and were tasked to annotate them on a triangle interface to indicate their "relational intents". Each vertex of the triangle corresponded to one of the three possible relational intents :

- "With" (Avec) : You intend to converge with what the other musician is doing.
- Against ("Contre"): You intend to diverge from what the other musician is doing.
- Without ("Sans"): You intend to ignore what the other musician is doing.

It was also mentioned that although the triangle displayed clearly identifies three zones (cf Figure 3.3), the annotations were continuous and thus participants could indicate the intensity to which they conformed to each intent by varying their proximity to the corresponding vertex. See Figure 3.7 for a graph showing the timeline of some participants' annotations. As we noticed that this type of graphs was useful for analysis and produced each time A^3PM was used, we decided to systematize their production and develop the visualization interface presented above.

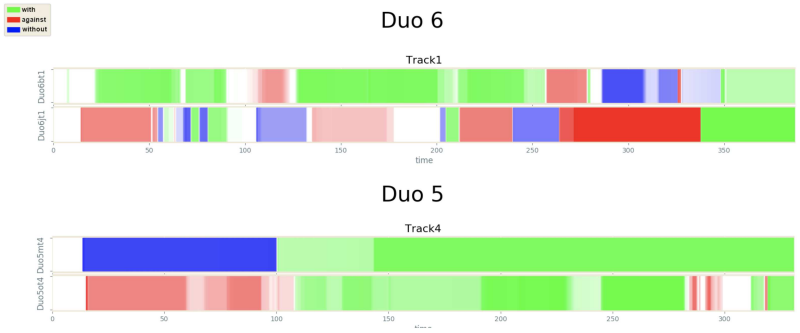


Figure 3.7: One of the first graphs produced in the study presented in [21] showing the timeline of some participants' annotations.

Using the same corpus of recording collected by Golvet *et al.*[19], Wolf *et al.* [97] recruited participants and asked them to listen to a selection of recordings from this corpus and to annotate them using a slider to rate the amount of perceived tension in the music, going from "No tension" to "Highly tensed".

[19]: Golvet et al. (2021), 'With, against, or Without?'

[97]: Wolf et al. (2023), 'Beyond Togetherness'

In a study by Majeau-Bettez *et al.*[20] on the focus of auditory attention in the group performance of Eliane Radigue's *Occam Delta XV* (2018), members of *Quatuor Bozzini* were asked to annotate their performance by indicating on a square interface where their focus of attention was placed. The square mimicked the disposition of the quartet, with the instruction given that the participant annotating was placed at the bottom vertex. This illustrates how the annotation

[20]: Majeau-Bettez et al. (2023), 'Tracking Auditory Attention in Group Performances'

interface is not necessarily an abstract space but can reflect a physical space, thus providing a direct phenomenological link between the performance and the annotation task, favoring immersion and helping to reenact the memory of the performance[3].

A similar procedure was employed in a study by Faraco *et al.*[98] in which trios of improvisers were also tasked to indicate their focus of attention. This time, a triangle interface mimicked the performance setup. Because musicians were playing individually in closed booth with headphones with the sound of one musician panned to the right and the sound of the other panned to the left, they could drag the dot toward the right or the left vertex of the triangle to indicate that their attention was focused on one musician or the other, or closer to the bottom vertex to indicate that they were listening to themselves. Using results from this experiment, videast Elsa Laurent [4] created multiple novel ways to visualize the focus of listening attention of musicians using blur or occultation effects (cf Figure. 3.8).

3: a video excerpt of the animation reconstructed from the Quatuor's annotations along with the music is visible on the companion website: https://alienorgolvet.com/thesis/chapter-a3pm.html

[98]: Faraco et al. (2024), 'Listening Behaviors and Musical Coordination in Collective Free Improvisation'

4: http://www.elsalaurent.com/

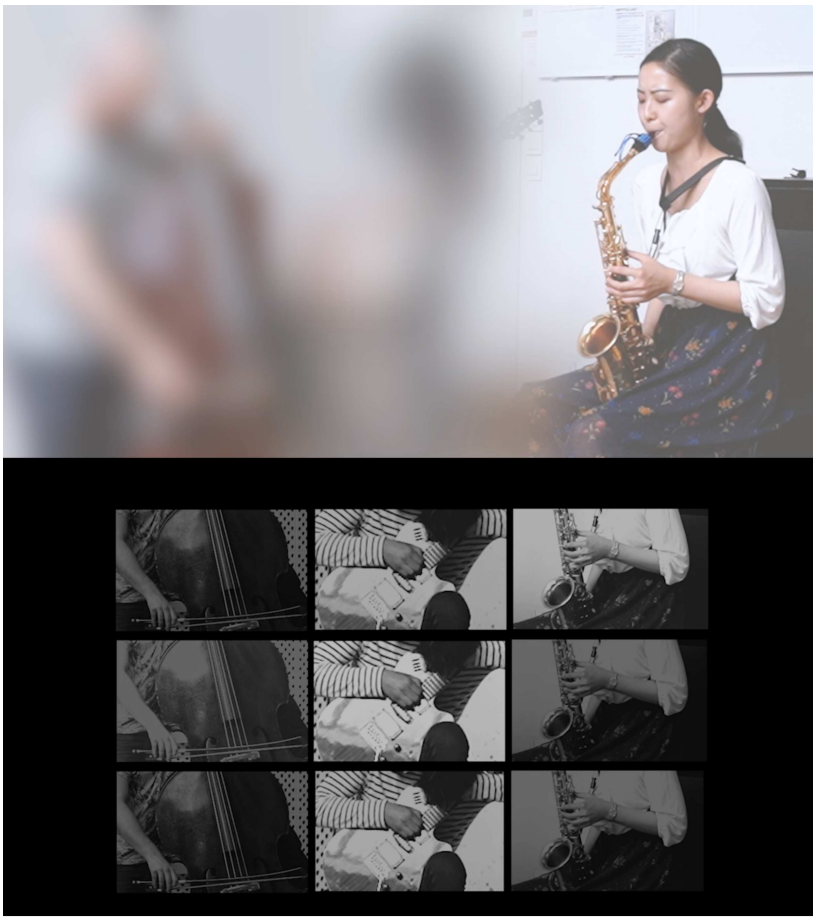


Figure 3.8: Some stills from videos made by videast Elsa Laurent using results from an experiment with A^3PM studying focus of listening attention of trios of improvisers. Blurry or faint musicians indicate that the focus of attention was away from them.

3.4.2 Use in Live Participatory Setting

More recently, A^3PM has been used in live participatory settings in which the annotation task is not performed by musicians but by the audience during a concert.

The first instance of this was organized at Ircam in November 2023 as part of the *Journées Perception Sonore* ("Sound Perception Days") of the *Société Française d'Acoustique* (SFA, French Acoustic Society). The event took the form of a "laboratory-concert" taking place in the *Espace de Projection* at Ircam and with an audience of more than 100 persons. Much more than a simple concert, the event was conceived as a large-scale experiment to study modes of listening and perception in a live concert setting. As the concert took place, audience members had to connect to a web page leading to the A^3PM application to perform live annotation.

The event was divided into two main sections. In the first one, two improvisers played a 20 minute improvisation. As they played, audience members were asked to move a slider to indicate which of the musicians they were listening to. Moving the slider to the right (resp. to the left) meaning that they were listening to the musician to the right (resp. to the left) and moving it to the center meaning that they were listening to both equally. Unbeknownst to the audience, a screen displayed instructions to the musicians to guide their playing thus providing different conditions for testing the audience locus of attention.

In the second section, audience members listened to a variety of spatialized electronic compositions composed by Sébastien Roux[5]. In each of these compositions, an audio cue had to be spotted and participants had to answer whether or not the audio cue was present in the track they were told to follow.

5: http://www.sebastienroux.net/

Due to the short production time and particular constraints imposed by the event (such as the need to synchronize annotations to an external audio stream), a specific version of A^3PM had to be developed. In particular, a questionnaire interface had to be implemented as it is not part of the regular version of A^3PM. Moreover, OSC communication with the production *Max/MSP* patch had to be established so as to synchronize the web application and the diffusion of sound (for instance to automatically pass to the next question in the second section). Finally, a specific visualization

interface was developed for the occasion.

Visualization of audience members' annotations and answers were presented live right after the event (cf Figure 3.9). This provided an immediate feedback and feeling of completion to the audience and elicited discussion between audience members and experimenters.

This experience was reiterated as part of an event in the Manifeste festival organized by Ircam in June 2024 [6]

6: https://www.centrepompidou.fr/fr/programme/agenda/evenement/d48AJ9Q

Figure 3.9: Photograph of the visualization of the results of the laboratory-concert being presented live at the Espace de Projection at Ircam. White curves represent the timeline of audience members annotation of the first section of the concert. The red curve represents the curve of the mean position at each time.

Similarly, in October 2024, A^3PM was used as part of a concert with the Splitter Orchestra and the Trondheim Jazz Orchestra. In this event, rather than collecting results on the audience perceptive experience, annotations are used as a way to interact directly with the music played by having instructions to the musicians being triggered depending on the audience positions on a slider.

3.4.3 Use by Artists in Situation of Play and Rehearsal

Another use of A^3PM which has yet to be thoroughly explored is by groups of musicians in situations of rehearsal. The reflexive point of view given by the task of self-annotation and the visualization of other group members being immediately accessible could provide ground for interesting discussion and reflection on group dynamics.

This has been explored during a research residency with the MilesDavisQuintetOrchestra conducted by Clément Canonne as part of the DSYNC project[7]. Among the different experiments, one of them asked group members to use A^3PM to indicate on a slider their level of immersion within the group ranging from "i had a clear feeling of my own individuality" to "i was one with the

7: A documentary film on the residency has been produced and can be seen at the following address : https://laplateforme-sonsdhiver.org/dphase-documentaire/

group" (cf Figure 3.9).

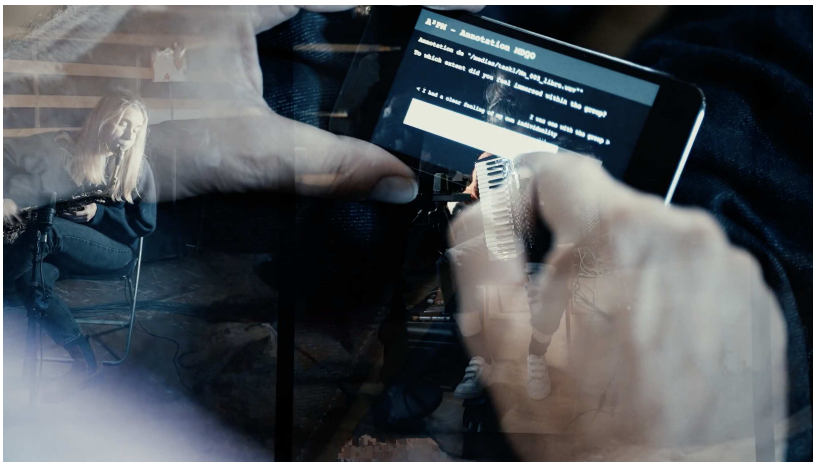


Figure 3.10: A still from the documentary film D_PHASE by Romain AL in which we see a member of the MilesDavisQuintetOrchestra using A^3PM to annotate their performance.

3.5 Discussion

The development of A^3PM demonstrates how a distributed framework can fit the specific demands of a particular problem.

In the case of self-annotations of musical performance, designing a distributed system allowed us to propose a technical solution that is : 1) lightweight, requiring little material apart from a laptop, a wifi router and devices such as smartphones and tablets sometimes already owned by participants. This makes our solution easy to deploy "in the field" thus guaranteeing one of the main theoretical strength of the methodology, that of ecological validity 2) available on heterogeneous devices, thus minimizing cost of deployment as participants' own devices can be used 3) quick to set up due to the minimal equipment required but also thanks to the networked architecture, hence satisfying one of the main constraints of the methodology employed, that of minimizing the setup time to avoid deterioration of participants' memories. 4) agnostic to the number of participants and enabling simultaneous use, thus making it viable for a large variety of situations ranging from single musician or quartets to large audiences.

Two main strengths can be identified in the implementation of A^3PM. First the modularity of the user experience provided by its implementation as a state machine. This enables us to easily create new elements in the user experience (different paths suited for each research project or new annotation types, for instance when we implemented a questionnaire for the laboratory concert)

without deep modifications to the application's architecture.

The second strength lies in the visualization interface that we developed. More than a mere practical tool, this is an element of design for different levels of expertise. A sufficiently skilled experimenter could still use the application in the first intended way, by parsing user annotation data and analyzing them using a programming language, but the visualization interface allows users that lack these skills to access a generic analysis of these data. By extending the range of potential users and by providing immediate feedback to participants, we observed that this element of design unintentionally gave rise to new use cases, outside of musicology experiments.

Use by artists in situations of rehearsal could help redefine the work process of a music band by providing information on the shared experience of playing together or could become a tool to support situations of playing with constraints on group dynamics.

Use in live situations with an audience of participants provides an interesting perspective on participatory methodologies in research. Presenting an immediate feedback on the performed task in the form of visualizations has been met with enthusiasm, making the experience less abstract for participants and making them realize the benefit of the research project. This immediate feedback on results provides ground for further discussion and reflection between participants and researcher, thus lowering the barrier that might separate these two categories, and thus improving the feeling of involvement of participants in the research. This may be seen as a reciprocal gift given to the participants, the initial gift being the participants' data collected by researchers. Some works on participatory science stress the importance of reciprocity to prevent participatory science from becoming extractive and to empower voluntary participants of these research programs [99, 100].

[99]: Hetland (2020), 'The Quest for Reciprocity: Citizen Science as a Form of Gift Exchange'
[100]: Wilmsen (2006), 'Participation, Reciprocity, and Empowerment in the Practice of Participatory Research'

3.6 Chapter Summary

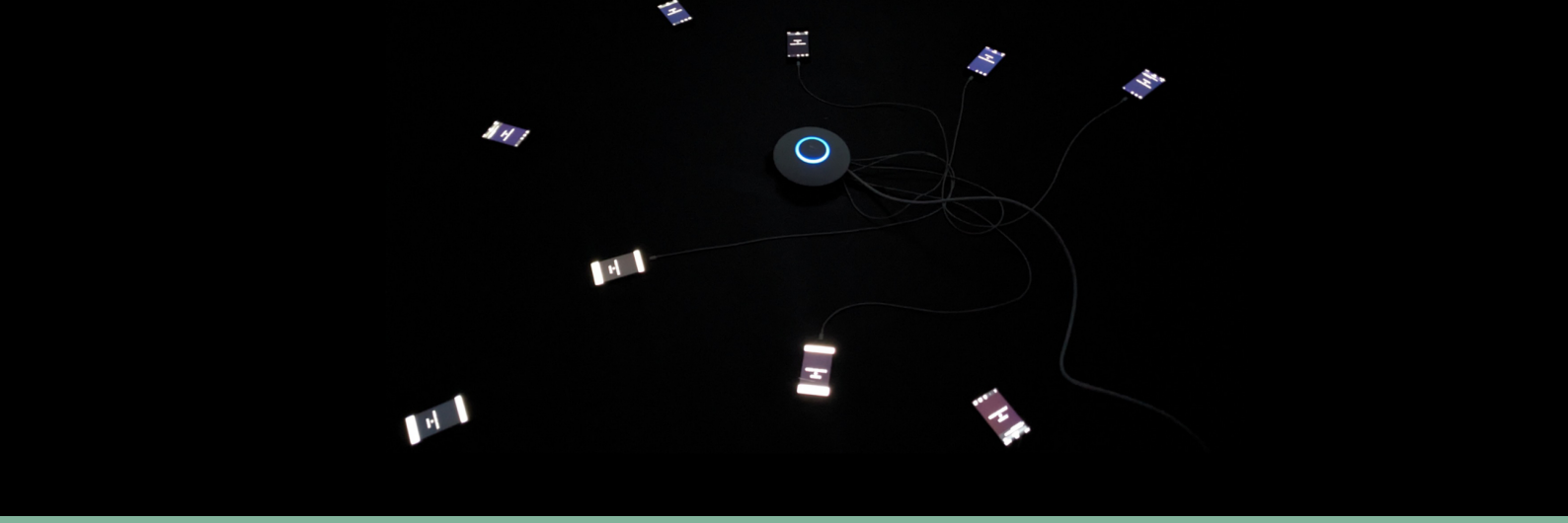
In this chapter, we introduced A^3PM, an application developed for the purpose of a new methodology of research : self-annotation

of musical performance. This methodology involves having musicians perform and immediately after to ask them to continuously annotate the recording on a selected aspect of their performance. The development of A^3PM followed a user-centered methodology by identifying the specifications needed to support the envisioned use cases. It also aimed to reach a degree of generalizability to support new use cases as the methodology of self-annotation was applied to more diverse research situations.

A^3PM is implemented as a *soundworks* application. It provides a web-based interface allowing musicians to annotate performances on their devices immediately after playing. The system supports various annotation interfaces, such as sliders and geometric shapes, tailored to different research needs. Thanks to the distributed architecture, the experimenter only needs to transfer the sound files to annotate on the server before it is broadcasted to the various participants while annotation data is written directly on the server. Moreover, thanks to web technologies the annotation task can be performed on a variety of devices, provided that it supports regular web browsers, and it can be undertaken simultaneously by any number of participants. Finally, because it necessitates little technical material, it is a lightweight, cost-effective and quick to set up solution which tackles the specific constraints of the self-annotation methodology.

A^3PM has been used in multiple use cases. While first intended for research in empirical musicology and being used in multiple published research projects, new use cases emerged over time, some of them driven by new features of the application. These new use cases include use by artists in situations of play or rehearsal and use in live situations by an audience of participants.

More use cases are already scheduled in a close future, thus providing ground for further developments of A^3PM and of its theoretical backbone. Future improvements of the application includes more visualization tools and the ability to compute some statistics on the user annotation data. We also consider developing a user interface to configure projects for users not familiar with writing JSON files.



4 *Koryphaîos*: A Patchworked Compositional Environment for Distributed Music Systems

4.1 Motivations and Methodology

Several research works and artistic projects have harnessed the audience's smartphone devices' ability to connect to networks and to produce sound by considering it as an array of speakers distributed among the public[39] (cf. Chapter 1). However, despite the growing number of artwork, concerts and installations that have been proposed along the years, it can be argued that such approaches and compositional techniques are far from reaching widespread adoption. We postulate that one of the possible reasons for this state of affairs is the lack of high-level and ready to use composition environments that 1) lean on existing composer's skills and practices, and 2) take into account the specificities (e.g. network, number of devices) of distributed systems. Indeed, while on the one hand we have frameworks dedicated to build distributed music systems[88, 101] that lack high-level tools oriented toward composition, on the other hand we have software and libraries dedicated to composition [102, 103] that are not primarily oriented toward the specificities of the Web platform and of network-based approaches.

Composing for distributed systems therefore remains a difficult task that generally ends up with ad-hoc systems and idiosyncratic solutions. The difficulties one must face are twofold: 1) tackle complex design, architectural and development questions and 2) reduce the unbounded creative possibilities afforded by the system

[39]: Taylor (2017), 'A History of the Audience as a Speaker Array'

[88]: Matuszewski (2020), 'A Web-Based Framework for Distributed Music System Research and Creation'

[101]: Allison et al. (2013), 'NEXUS: Collaborative Performance for the Masses, Handling Instrument Interface Distribution through the Web'

[102]: Bresson et al. (2011), 'OpenMusic'

[103]: Agostini et al. (2015), 'A Max Library for Musical Notation and Computer-Aided Composition'

to define a creative space that can be artistically manipulated. In this regard, we consider with Magnusson that 'In new musical instruments created with general and diverse building blocks, the rationale for creating high-level constraints is primarily to engender an identity, a musical world that is simple, intuitive, and direct'[83]. To put in other words, when dealing with computer music environments, creativity arises from a set of carefully designed constraints and affordances encoded in the software that maps a defined space for musical expression.

[83]: Magnusson (2010), 'Designing Constraints'

In this chapter, rather than proposing an integrated and monolithic solution, we propose to approach this problem through the creation of *Koryphaîos*, a bridge between existing tools (i.e. *Max/MSP* and *soundworks*) that aims to improve their interoperability and ease of use in a common patchworked workbench, Indeed, contemporary music composers being generally familiar with the *Max/MSP* environment, we decided to build upon their existing practices and skills to foster the possibilities of our Web-based distributed music frameworks. Additionally, such an approach aims to put back the tools of composition and creation in the hands of the composers rather than relying on a developer as an intermediary agent, therefore leaving more time and cognitive space for the creative process rather than on solving technical issues.

For the design process, we decided to inscribe our methodological approach in the framework of *Meta-Design*[81, 104]. As described in Chapter 1, Meta-Design is a methodology aiming to tackle situations (of which artistic practice belongs to) in which "future uses and problems cannot be completely anticipated at design time". It aims to guarantee that the designed solution is able to co-evolve with its users and users are empowered to the status of co-designer of the application over time. Finally, it alternates between phases of software development and phases of documentation of user appropriation.

[81]: Fischer et al. (2006), 'Meta-Design'
[104]: Fischer et al. (2017), 'Revisiting and Broadening the Meta-Design Framework for End-User Development'

In this project, we also considered it important to approach our question from different perspectives within a heterogeneous team composed of persons with multiple backgrounds, skills and activities. We therefore employed a second-person perspective by working from the start in close collaboration with composer Luciano Leite Barbosa who was already familiar with composition for distributed ensembles of devices. In this frame, one of our first objectives was to re-create and re-implement a piece composed

by Luciano in 2018, *Color Fields* for accordion, smartphones and electronics (cf. Fig. 4.1)[1], into a more interactive and versatile compositional workbench. We also asked him to develop a pool of examples with a variety of compositional techniques as well as to propose new features he thought he could need to further simplify his compositional process for distributed music performance. We iteratively developed the first version of *Koryphaîos* with the intent to make all these examples and use-cases fully working. Alongside these goals, we also asked Luciano to test every version of our application and to deliberately push it to the limits. Indeed, as Tahiroğlu et al. note, 'Musicians often use musical instruments in ways that the original designers never intended, probing for hidden affordances'[63]. We also regularly organized test sessions in the studio to test the application under more realistic conditions with a larger number of mobile phones. These test sessions were not only useful to detect technical issues but were also opportunities to discuss with Luciano about new features or modifications. It allowed us to readjust the course of development to incorporate unanticipated elements, which pushed us to design our software architecture in terms of modularity and flexibility to foster rapid testing and addition of new features.

1: a video of the performance of *Color Fields* is available on the companion website https://alienorgolvet.com/thesis/chapter-koryphaios.html

[63]: Tahıroğlu et al. (2020), 'Digital Musical Instruments as Probes'

Figure 4.1: Premiere of *Color Fields* by Jean-Étienne Sotty at the *CENTQUATRE-PARIS*, 2018.

Another design goal was to provide an environment that hides some low-level aspects (e.g. networking, message routing) to the users, but still provide several entry points at its domain level (e.g. audio synthesis, mapping). As such, a large part of *Koryphaîos* is conceived with the idea that it could provide a 'a background against which situated cases, coming up later, can be interpreted'[81], an application that is able to translate the creative endeavor of its users in the language of a network of mobile devices. This approach rep-

[81]: Fischer et al. (2006), 'Meta-Design'

resents an attempt to lower the technical wall that exists between composers and the network of sound-producing mobile devices, to facilitate the process of co-adaptivity between them. Objectives of modularity and openness guarantee that the network will adapt to the many idiosyncrasies of its users but in return, we hope that the relationship with the network (with its specific capacity to question traditional music boundaries [8]) created through *Koryphaîos* could influence composers to reinvent their practice.

[8]: Bevilacqua et al. (2021), 'On Designing, Composing and Performing Networked Collective Interactions'

Finally, while we recognize that the environment as described here, necessarily embodies some aesthetic and compositional perspectives of one single composer (i.e. Luciano L. Barbosa), we hope the genericity and extensibility of the proposed system could prove to be interesting for other composers and artists as well.

After a short review of the related works (Section 4.2), we will describe in Section 4.3 the design choices and overall architecture of our environment. Then, we will showcase in Section 4.4 some of the artistic and musical possibilities it unfolds. Finally, we reflect on this project and some of its shortcomings in 4.5.

The work presented in this chapter has been published in an article at the 2022 Web Audio Conference[22].

4.2 Related Works

In this section we present several tools dedicated to computer-assisted composition that have been proposed over the years. We then present the choice we made amongst these software for our own application.

A number of dedicated software and tools (e.g. *Bach* and *Cage* [103], MaxScore [105]) have been created, often with the help of, or by composers themselves, to manipulate symbolic musical data and scores. For example, *OpenMusic*[2][102] has been developed at IRCAM since the end of the 1990s. It uses a graphical interface and offers a large range of functionalities for algorithmic composition and usage of digital signal processing. [106].

[103]: Agostini et al. (2015), 'A Max Library for Musical Notation and Computer-Aided Composition'

[105]: Hajdu et al. (2018), 'Maxscore'

2: `https://openmusic-project.github.io/`

[102]: Bresson et al. (2011), 'OpenMusic'

[106]: Agon et al. (2006), *The OM Composer's Book. Volume1*

Another example is *ossia score*[3]. Born from the *i-score* software which has been developed since the late 1990s at LABRI [107]. The software focuses on the sequencing of multimedia events and

3: `https://ossia.io/`

[107]: Allombert et al. (2008), 'Iscore'

on the construction of interactive scenarios. It benefits from the embedding of multiple programming languages and its support for a large number of communication protocols (OSC, websocket, etc. . .) [108].

[108]: Celerier (2018), 'Authoring Interactive Media : A Logical & Temporal Approach'

More recently, the *Bach* package for *Max/MSP*[4] has been proposed by Agostini et. al [103]. *Bach* (and its brother package *Cage* [109]) provides various objects, including graphical interfaces, made for performing low and high-level operations on lists of musical data. *Bach* has been heavily inspired by *OpenMusic* and both environments share a lot of functionalities, but while the latter is more advanced and provides more possibilities and processing power in some contexts, *Bach* benefits from the ability to operate with other elements in the *Max/MSP* environment.

4: https://www.bachproject.net/

[103]: Agostini et al. (2015), 'A Max Library for Musical Notation and Computer-Aided Composition'

[109]: Agostini et al. (2014), 'Cage'

Amongst these options we decided to work with the *Bach* library. We wanted to create a tool as accessible as possible, and a large number of composers are already familiar with *Max/MSP* and use it in their works. Also, as one of our goals was to develop a more fluid and user-friendly communication interface between the two software, we think the architecture developed in our application could serve as an interesting model that could be declined to other *Max/MSP* packages (e.g. score following, MuBu [110], . . .), either in combination with *Bach* or not.

[110]: Schnell et al. (2009), 'MuBu & Friends - Assembling Tools for Content Based Real-Time Interactive Audio Processing in Max/MSP'

4.3 Design Overview

Guided by these objectives, we developed a *soundworks*-based application for composing distributed music pieces using the *Bach* library in *Max/MSP*. As an overview, *Koryphaîos* is built around a local network of devices, at the center of which lies a *Node.js* server to which *Max/MSP* and the mobile devices can connect. The *Node.js* server receives the score information from *Bach* and *Max/MSP* though OSC and dispatches this information to the connected Web client through WebSocket channels (cf Fig. 4.2).

In the following section we present the application from a design perspective. We start by presenting the composition interface available as a *Max/MSP* patch. We then present how the application produces music out of an array of mobile devices by detailing the communication process over the server and the custom audio

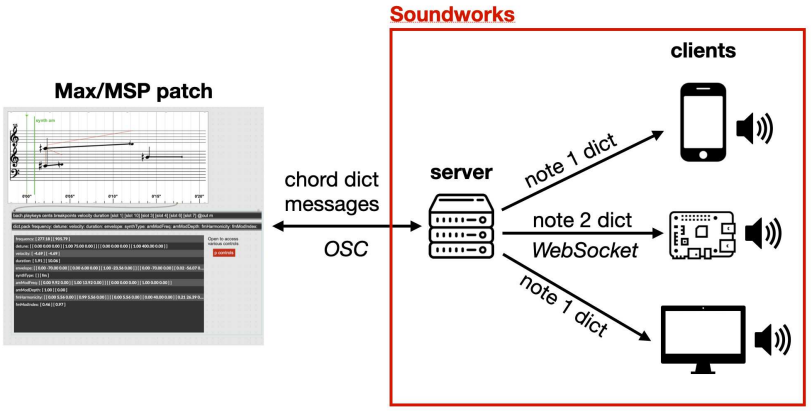


Figure 4.2: Diagram of the communication between the different parts of *Koryphaîos*.

engine we developed using the *Web Audio API*. Next, we present a variety of functionalities the application provides for monitoring and control to facilitate its usage both in studio and in concert situations. Finally, we detail several features aimed at fostering its appropriation and customization by users, potentially opening doors for future evolutions.

4.3.1 An Interface for Composition

From the point of view of the composer, our application is primarily seen as an interface for composing distributed music performances through a *Max/MSP* patch (Fig. 4.3). The core element of the patch is a `bach.roll` object provided by the *Bach* library. A `bach.roll` is presented as an interactive score sheet which you can edit to place musical notes. Each note can be associated with arbitrary metadata (i.e. using "slots") of different types (e.g. envelope breakpoints, modulation parameters, text, filenames). As the score contained in the `bach.roll` object is played in real-time, its output notes are collected by *Koryphaîos*' `kp.to_soundworks` object and formatted as a *Max/MSP* dictionary. The user can freely define which data is to be collected from the `bach.roll` object and to which parameters in *Koryphaîos* they are mapped to by sending a list of parameters to one of the outlet of the `kp.to_soundworks` object. The dictionary then created contains all the desired data for sound synthesis over different synthesizers (e.g. AM, FM) developed using the *Web Audio API*, for instance: synthesizer to use, frequency, velocity, duration, envelopes, synthesizer parameters, etc.

We designed *Koryphaîos*'s *Max/MSP* objects so that lower-lever coding from the composer can be avoided. Any note data sent out by the `bach.roll` object is automatically formatted to be consumed

by the rest of the application.

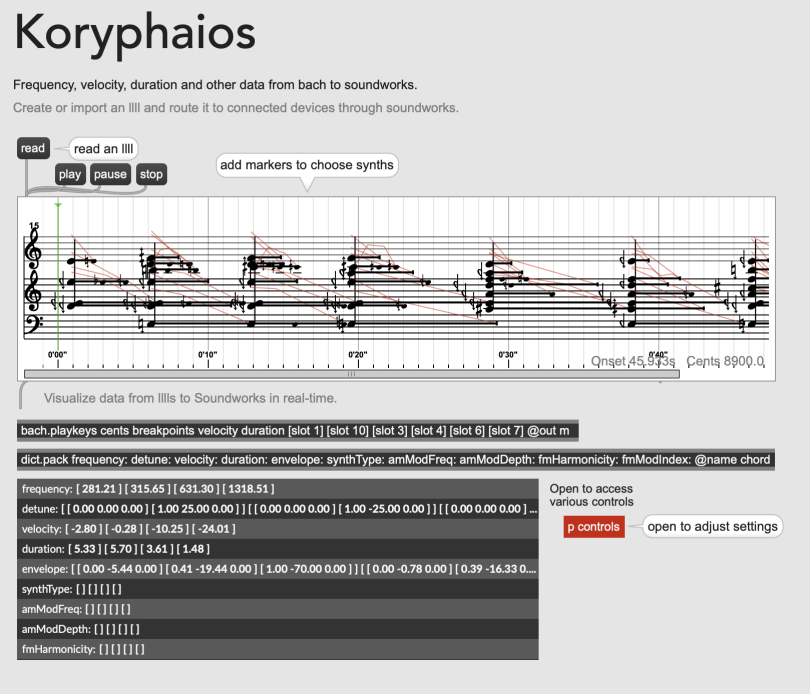


Figure 4.3: Example of the main composer interface in *Max/MSP* using the *Bach* library in *Koryphaîos*.

4.3.2 The Audience as a Speaker Array

Each note information is sent and parsed from the *Max/MSP* patch to the *Node.js* soundworks server through the `soundworks.shared-state` object built on top of OSC[5]. Upon reception on the server-side, the notes are tagged with a synchronized timestamp and dispatched to all connected clients for rendering using Web Audio synthesizers. By default, the application currently includes different dispatch strategies: **sendAll** (all notes received are sent to all connected clients at the same time), **randomSpread** (the n notes of a chord are split between n random groups of clients of the same size), **randomPoint** (any incoming chord is sent to a single randomly-chosen client).

5: a newer version of the object uses WebSockets to communicate with the server

Upon reception of the time-tagged note by the client, the latter creates an instance of the specified synthesizer (predefined or user-defined), and schedule its rendering using a synchronized scheduler created thanks to the `@ircam/sync`[6] library, which achieves clock synchronization up to 5 ms [89]. By default, the application proposes 5 types of generic synthesizers: a basic sine synth, an AM synth, a FM synth, an audio buffer player and a granular synth.

6: `https://github.com/ircam-ismm/sync`

[89]: Lambert et al. (2016), 'Synchronisation for Distributed Audio Rendering over Heterogeneous Devices, in HTML5'

The synthesizer instances are finally piped through master buses for balance and volume controls.

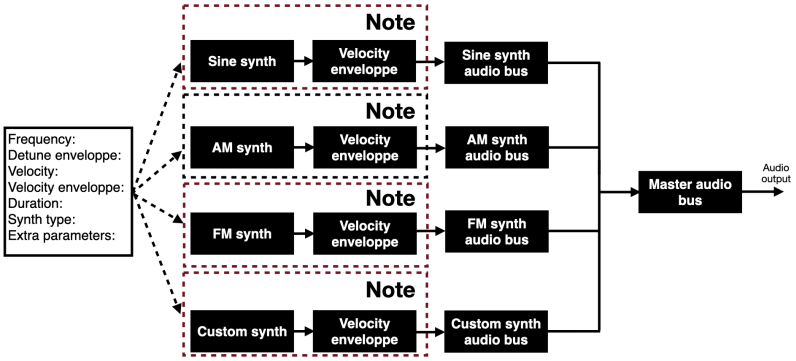


Figure 4.4: Graph of the audio path within the application. Upon reception of the score information, a `Note` object is instantiated, containing a synthesizer instance and a velocity envelope. The `Note` is connected to the corresponding synthesizer's bus which is connected to the master bus. Finally the output of the master bus is sent to the audioContext's destination.

4.3.3 Contexts, Control and Feedback

The application has been designed to be used both in the studio and in concert situations. To fulfill the multiple and sometimes contradictory requirements of these different contexts, we decided to provide multiple access to the same functionality as well as complementary information from different entry points.

For example, alongside the *Max/MSP* interface, we developed a browser-based *controller* interface. This interface is composed of different parts useful both for monitoring and control:

- A text box that logs any incoming note on the server, which is useful to monitor the proper functioning of the application both in working and in concert situations.
- Buttons to switch between available dispatch strategies for the incoming notes on the fly.
- Master and synthesizer-specific bus controls containing each a mute button and a volume slider (cf. Fig. 4.5). The Master also exposes two sliders for controlling the frequencies of a low-pass filter and a high-pass filter. All these controls are also available in the *Max/MSP* patch and their visual display is synchronized over the network. To simplify the control and use of these different interfaces in concert situation, we also implemented possibilities of control over a MIDI device either in the *Max/MSP* patch (using the built-in MIDI map assignment) or directly in the browser using a MIDI map assignment interface developed using the Web MIDI API[7].

7: this constituted the first steps in the development of the `sc-midi` component, cf Chapter 2

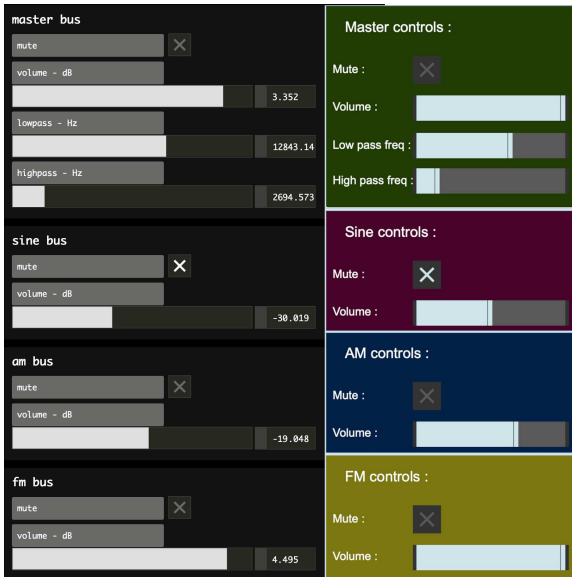


Figure 4.5: A part of the controller interface: audio bus controls in the browser (left) and in *Max/MSP* (right)

Finally, we implemented a *concert mode* that provides a series of interfaces that guide the public through the performance:

1. Upon connection to the web page of the applications, the participant is brought to a volume test page in which they are asked to set the volume of their phone to a comfortable level. While mainly technical, this step can also be considered and used by the composer as a real introductory part of the piece.
2. This step is followed by a waiting screen showing the names of the piece and of the composer, prompting them to wait for the performance to begin.
3. Upon starting the performance in the controller interface, participants are automatically brought to the main playing interface in which the audio engine is connected to their device's output and reception of notes from the server is activated. We also included a simple visualization that displays the current energy of the sound produced by the participant's device through a full-screen animated gray scale.
4. Once the performance has ended, participants are brought to the end page thanking them for their participation.

4.3.4 Appropriation and Evolutionary Growth

To support the evolution of the application in the hands of its users, *Koryphaîos* provides various possibilities for customization, inclusion of user-made components and sharing of information.

Koryphaîos allows advanced users familiar with *JavaScript* to

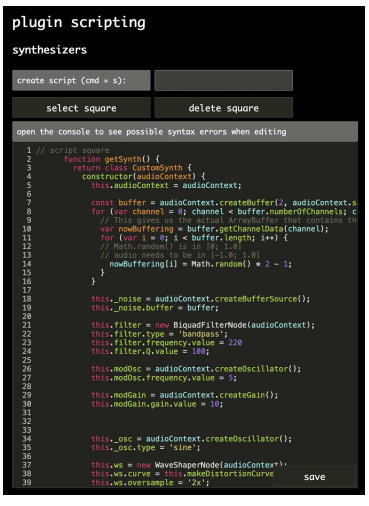


Figure 4.6: A user-made synthesizer in the scripting interface in the browser.

program custom implementations of several levels of the application. Thanks to the *soundworks' scripting* plugin[8] (cf. Chapter 2), user-made scripts can be created and modified on the fly, without having to restart the application or the network. Since user-made scripts are stored locally as a single file they can also be easily shared among users.

8: https://github.com/collective-soundworks/soundworks-plugin-scripting

Script creation and edition is made possible by using text editors available in the controller interface in the browser (cf. fig 4.6 and Chapter 2). Upon creation of the script, any user-made component is treated as any other component in the application. For instance, a user-made synth can be called up by its name in the `bach.roll` object and a dedicated audio bus with GUI in the controller interface is dynamically created. As explained before, the mappings between the `bach.roll` slots and the user-defined synths' parameters can also be defined at runtime in the *Max/MSP* patch by sending specific messages to the `kp.to_soundworks` object.

This process of appropriation by users also extends to more "social" aspects surrounding the application. To this end, we also created an information repository in the form of a wiki on the github repository of the application[9]. It already contains documentation and tutorials on several aspects of *Koryphaîos* as well as the description of the example patches developed by Luciano L. Barbosa (cf section 4.4). We hope this knowledge base will grow and develop as users may share their own components, musical examples and ideas.

9: https://github.com/ircam-ismm/koryphaios/wiki

4.4 Musical Examples

In this section, we present several musical examples created within *Bach* that showcase the compositional possibilities opened by the application. First, we present two simple case-studies created during the design and development of the application. Second, we describe the first sketch of a sound installation, *Refraction*. Third, we present *Color Fields*, a piece composed in 2018 and rewritten using *Koryphaîos*. Finally, we present *Trialogues*, a piece composed in 2023 during a residency at Grame[10] in Lyon. All these musical examples have been created by Luciano L. Barbosa, and explore distributed synthesis techniques that expand the familiar notion of additive synthesis by taking into account the spatialization of

10: https://www.grame.fr/

each frequency. As a sound can be created or recreated through several sources (e.g. smartphones or other connected devices), the possibilities of distribution of frequencies through devices are therefore numerous, including random distribution, single or multiple frequencies per device, organization of devices into groups, etc. The resulting sound has an intrinsic immersive quality, as a high number of sound sources are used and these sources can be easily spread in the concert space.

4.4.1 Case Studies

A first example of the possibilities of *Koryphaîos*, leveraging also on the *Cage* library for *Max/MSP*, is to read a sound analysis and resynthesize sounds directly with smartphones. In the following example (see Fig. 4.7), the object `bach.readsdif` is used to read an sdif file of a sound analysis, which is displayed in the `bach.roll`. A number of symbolic transformations, such as time stretching (using the `cage.timestretch` object) or frequency shifting (using the `cage.fshift` object) can be applied to this resynthesis. All these transformations can be tested, rendered and listened to in real-time through connected devices.

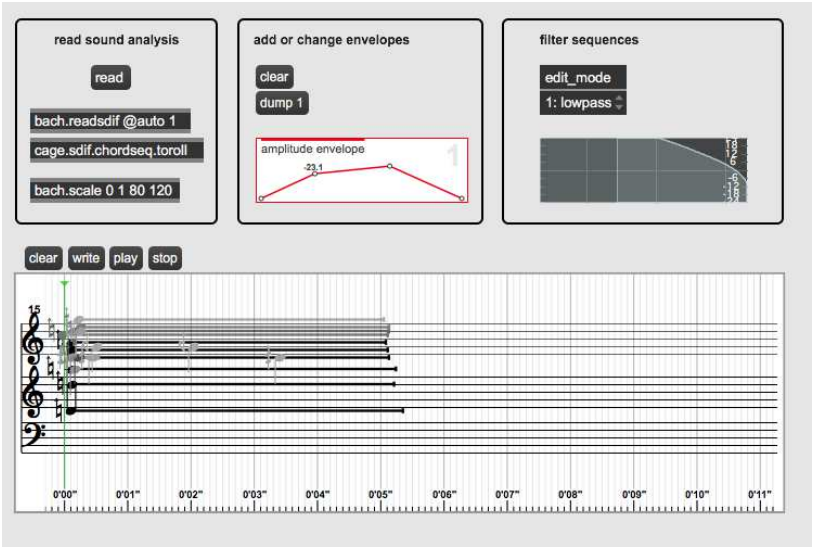


Figure 4.7: *Max/MSP* example patch of additive resynthesis of sound analysis using *Koryphaîos*.

More complex and generative processes can also be handled by *Koryphaîos*. The example shown in Fig. 4.8 shows the possibilities of using generative material in *Bach* and sending it to *soundworks* in real-time for audio rendering. This simple generative patch creates a new sequence when the cursor arrives at the marker `generate seq`, using random values to create a new harmonic sequence.

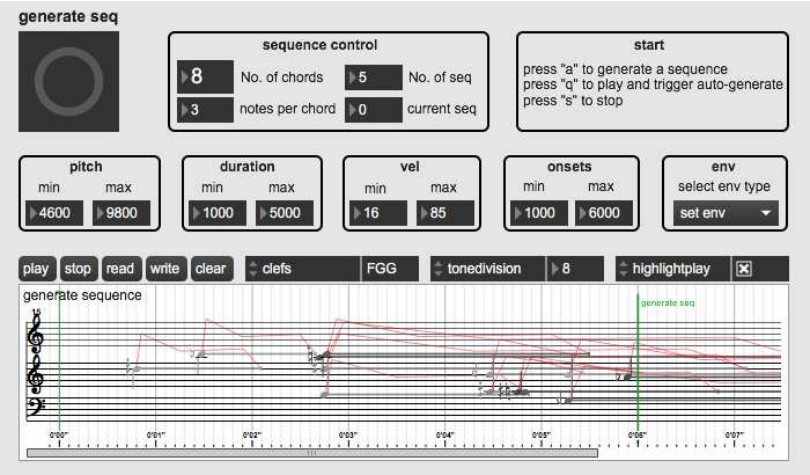


Figure 4.8: *Max/MSP* example patch of generative music using *Koryphaîos*.

4.4.2 Refraction

The sound installation *Refraction* is an example of a piece created directly using *Koryphaîos*. The piece is an installation for smartphones that are spread out in the performance room and that includes the participation of the audience. Its main compositional materials are additive synthesis and FM synthesis, with occasional use of AM synthesis, where the sonic result can be envisioned as a distributed synthesis technique in which the rendering of each component is distributed in space amongst devices.

The installation consists of three `bach.roll` objects that are linked to one another through markers, playing independently and overlapping at times. During the compositional process, the frequency materials were input freely on the `bach.roll` and *Koryphaîos* allowed immediate feedback of the resulting sound. Each note on the `bach.roll` (see Fig. 4.9) contains basic data such as pitch, velocity and duration. The *slots* are used to carry additional data including amplitude envelopes, amplitude modulation values for modulating frequency and tremolo depth, frequency modulation values for harmonicity and modulation indices, and type of synthesizer (e.g. sine wave, am or fm) to be used.

4.4.3 Color Fields

Color Fields, dedicated to Jean-Étienne Sotty, was composed in 2018 during the *Cursus* program at IRCAM[11]. It was written for XAMP microtonal accordion [111], smartphones, electronics and included audience participation through the use of mobile devices.

11: `https://youtu.be/4GuYtPejijI`

[111]: Sotty et al. (2017), 'L'accordéon Microtonal XAMP : Gestation, Fabrication et Évolution d'un Nouvel Instrument'

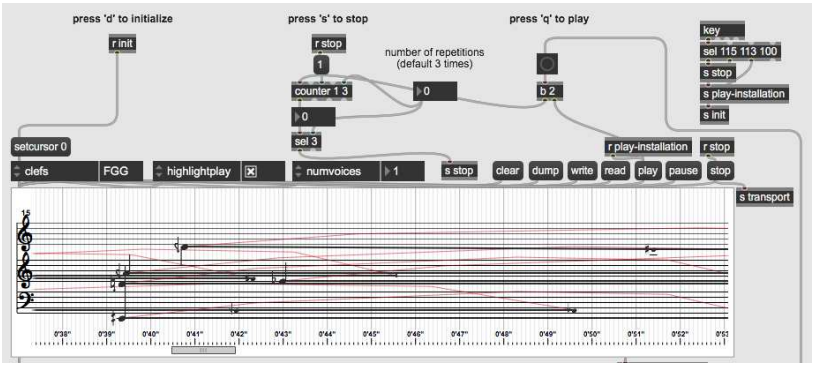


Figure 4.9: Patch of the *Refraction* installation.

The electronics were conceived to be diffused mainly through the audience's smartphones, with the aim of spreading the sound throughout the hall and having audience members participate in the sound production of the work. This feature allowed interesting possibilities of sound masses and harmonic blend between the soloist on stage and the sound coming from the devices of the audience.

The piece used additive synthesis as its main compositional material, and each frequency was assigned to a single device from the audience, randomly distributed among the smartphones of connected audience members. The frequency material was created through a number of processes carried out in the *OpenMusic* software, such as sound analysis and transformation of the resulting data, and exported to *Bach*. Other composition techniques used in the piece included free manipulation of harmonies directly within the `bach.roll` object.

In the first version of the piece, using ad-hoc OSC communications and protocol between *Bach* and *soundworks*, frequency and velocity values for each smartphone were hardcoded in advance in flat files directly read by the *soundworks* server. Each event in the main patch would trigger a *bach.roll* containing markers that controlled the start of each harmony stored in *soundworks*. Only the envelope values were sent from *Bach* to soundworks in real-time, handling the overall volume of the synthesis distributed through the devices of the audience. In order to organize the movement of different harmonic fields, the synthesis data was assigned to different groups of smartphones. Such architecture, with data spread between the *Max/MSP* patch and *soundworks*, was however difficult to test and change, making the compositional process slow, cumbersome and error prone.

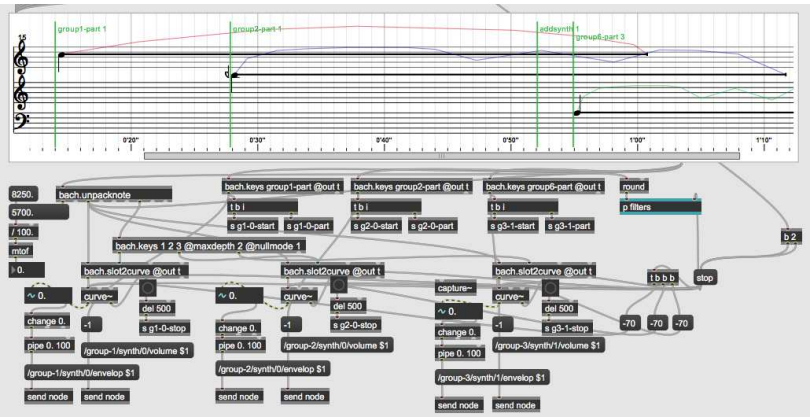


Figure 4.10: Concert patch of the first version of *Color Fields* in *Max/MSP*.

As discussed in Section 4.1, being able to recreate *Color Fields* (see the first patch version in Fig. 4.10) within *Koryphaîos* was one of our main goals from the beginning. As a result, we consider the new updated version of the piece rewritten using *Koryphaîos* to be both more efficient and versatile as the synthesis data and parameters are fully contained in the `bach.roll` and can therefore be manipulated in real-time and from a single place.

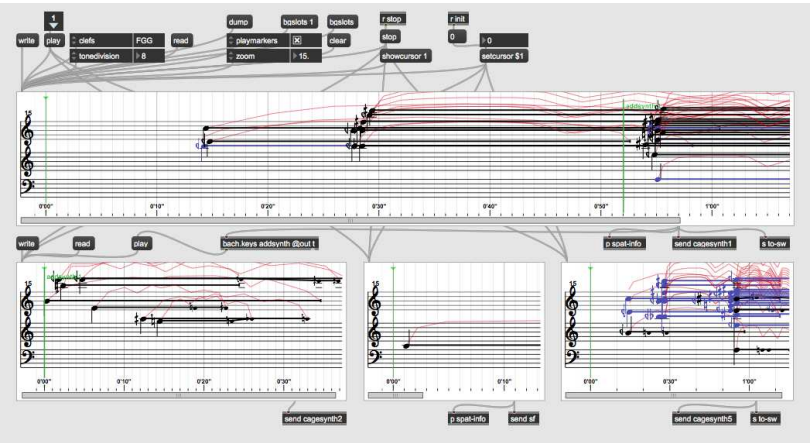


Figure 4.11: *Max/MSP* patch of the novel version of *Color Fields*, rewritten in *Koryphaîos*.

In this new version shown in Fig. 4.11, each *event* of the piece stores one or more `bach.rolls` that are in direct communication with *soundworks* through the `bach.roll`'s playout outlet. Each `bach.roll` connects to a `send` object that routes the data to a subpatch containing the *soundworks* component. Finally, *soundworks* receives values of frequency, velocity, duration and envelope and propagates them to its smartphone clients, which carry out the synthesis themselves. AM and FM synthesis parameters are included directly in the `bach.roll`, unlike the first version of the piece. The update both simplified the patch and increased its stability, validating also that *Koryphaîos* was able to handle the complex harmonic structures of *Color Fields*, such as dense chords and resynthesis.

4.4.4 Trialogues

Trialogues was composed in 2023. It is an audiovisual composition for flutes, electronics and smartphones. The performance lasts for approximately 45 minutes and is divided into 3 parts. Within all these three parts, the piece mixes the various sound sources accompanied by video projection. I participated in a residency at Grame in Lyon in September 2023 to assist with the technical part of the composition for smartphones created with *Koryphaîos* (cf Figure 4.12)[12]. *Trialogues* will be created in June 2025 in Paris.

12: a video excerpt is available on the companion website

My contributions to this residency consisted mainly in some marginal adaptations. For instance I developed a dispatch strategy that routed the low-frequency notes to the smartphones which were connected to portable speakers to avoid situations in which these notes would be played on smartphones' internal speakers not large enough to render them.

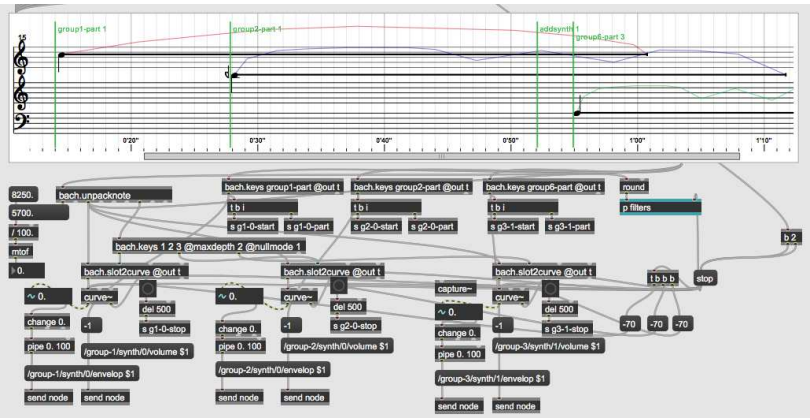


Figure 4.12: Pictures from the *Trialogues* residency at Grame in Lyon in September 2023. Right picture shows the video screen, flutist Samuel Casale, some smartphones around a WiFi router on the ground and the *Max/MSP* patch on a computer screen.

4.5 Post Mortem

We reflect on some of the shortcomings of our experience with *Koryphaîos* since its development.

First, while we stated our intent to follow a meta design process and laid ground for a phase of "evolving growth" and for appropriation by users by integrating customizable aspects in the application and a wiki, we ran into the problem that there was only one user of *Koryphaîos* (Luciano L. Barbosa). We can identify two main reasons for this situation. First, this is a common situation encountered in the field of research when developing new systems. As researchers, we often lack the time and resources to advertize

our work and support it, something even more true as a PhD student. Second, while efforts were made to create an application accessible to most and adapted to existing practices, *Koryphaîos* still demands the technical infrastructure, the knowledge and skills and the room to roll out a local network of multiple devices.

Maybe worse than letting *Koryphaîos* in a stale state with no evolution, we believe that this lack of users led the system to crystallize into a specific state and vision, the one used by Luciano. With no other user to challenge the limits of the application and to explore other ways to use it, the system eventually grew out to become more like a house of cards rather than a solid and stable structure: working for Luciano's cases but fragile and unstable in other situations and with the reasoning that if it's working sufficiently good for the only use case we have, we better not touch it by fear of making it collapse.

Secondly, in the development of *Koryphaîos* we sometimes encountered friction between our aspiration to develop a flexible and dynamic tool and our desire to rely on already existing software. This was most palpable when trying to implement custom elements and displaying them dynamically in *Max/MSP*, a strongly static language, making us rely on extensive and often unreliable workarounds.

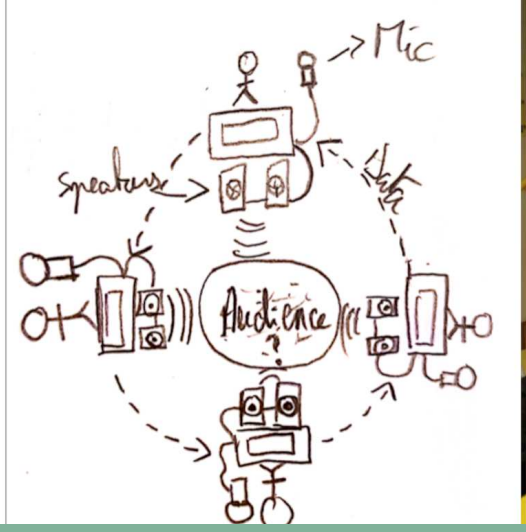
Finally, this experience highlighted the shortcomings of smartphones as devices in distributed music systems. When using *Koryphaîos* in live sessions, we encountered many instances of unprompted disconnections, devices going into sleep mode or being disturbed by the usual processes of the smartphone. Moreover, setting up the system and guiding the audience to connect to the correct web page demands pedagogical efforts that are very time consuming and often unadapted to a concert setting. Differences in equipment (with some smartphones too outdated to support the *Web Audio API*) or in technological literacy (with some people lacking knowledge about networks to follow the process autonomously) can be felt as frustrating by audience members who are left out of the experience. In fact, it appeared to us that the only good reason to use networks of smartphones as an array of speakers is that it creates a very specific sonic experience of the sound due to the spatial distribution of speakers. This effect can be reproduced by a network of nano-computers which, although more costly to set up, are not subject to the shortcomings mentioned

above and are much more flexible and stable. This experience was decisive in the decision to abandon smartphones as viable devices in our research and to turn towards networks of nano-computers (cf. Chapters 6 and 7).

4.6 Chapter summary

In this chapter, we introduced *Koryphaîos*, an application for networked music composition. *Koryphaîos* is built on top of and aims at creating a bridge between two existing libraries: the *Bach* packages for *Max/MSP* that provide tools for computer assisted composition and the *soundworks* framework which is dedicated to the development of distributed multimedia applications. In this work we followed a second-person perspective and a meta-design approach, emphasizing and promoting the heterogeneous nature of our team composed of researchers, developers, computer-music designers and composers.

Koryphaîos is designed around multiple connected parts. The first one is a composition interface in a *Max/MSP* patch, which at its core is a `bach.roll` object that sends out note data that is encoded and sent to the rest of the application via OSC communication. The second one is a *soundworks* application that, upon reception of the note data, dispatches them to the connected devices for Web Audio rendering. The application also provides functionalities for monitoring and control in a concert context available both in a *Max/MSP* patch and in a dedicated page in the browser. *Koryphaîos* was designed with appropriation and customization by its users in mind. It thus provides options for the development of user-made components, potentially opening for their sharing between users. We then presented different examples using a variety of compositional techniques (distributed additive synthesis, generative music, spectral analysis and resynthesis) that shows the flexibility of *Koryphaîos* for composing in a distributed context. Finally, we reflected on the whole design process and described why this project led us to abandon smartphones in favor of nano-computers.



5 *Simone*: Designing a Distributed Musical Instrument for Collective Improvised Interaction

Simone is a web-based system for co-located distributed improvisation controlled by audio inputs. The original idea for *Simone* was to explore the use of audio inputs in the control of distributed musical performance systems. The first prototypes started as a web interface for controlling an granular synthesis-based engine before evolving into a full fledged distributed system influenced by several historical examples. Eventually, during the course of its development, especially while experimenting with web audio development on Raspberry Pi computers, it gave birth to two different applications (*Simone* and *Simone Solo* presented in Chapter 6) which, although sharing similarities in their overall design and technical implementation, are intended for different situations of use and for different devices.

Simone, the first one is designed as a distributed instrument for collective improvisation. Using this version, groups of users may improvise using a web interface. These web clients are all connected to a server and thus can exchange information and data through the network. In this version of *Simone*, the type of data that can be shared and the path that it can follow depends on what we call "interaction scenarios".

Simone Solo, the second one, is conceived as a solo improvisation instrument in the form of a network of nano-computers and portable devices (i.e. Raspberry Pi) controlled by a single web interface. Each connected device then serves as a distinct sound source. The web interface provides controls ranging from general

control of all the sound sources at once to fine tuning of synthesis parameters of singular sound sources, thus allowing the creation of a large variety of different spatially distributed sonic spaces.

This chapter covers the collective version of *Simone*, which has been designed following a research-through design methodology as an experimental setup to study instrument learning processes and collective improvised interaction. We first present the motivations and objectives that guided its development. We then mention a few historical inspirations that served as an inspiration for its design. In section 5.3, we cover the design of the system itself and various aspects such as the interface or the "interaction scenarios". In a second time, we present our other contribution using this system which lies in an experimental study in which we invited groups of expert users to improvise with *Simone* collectively (Section 5.4). We present results from this study from which we collected qualitative data through semi-structured interviews with groups of users about their appropriation of the instrument and their collective interactions with the system and quantitative data in the form of self-annotations of their performance. Finally, we draw from these results to reflect on the design of *Simone* and more generally on the design of collective musical instruments (Section 5.6)

Parts of this chapter have been published in two research articles[21, 25].

[21]: Golvet et al. (2024), 'SIMONE'
[25]: Golvet et al. (2024), 'Designing a Distributed Musical Instrument for Collective Improvised Interaction'

5.1 Motivations, Early Prototypes and Research Objectives

The very basis of this project started as an attempt to explore audio inputs in the context of distributed musical systems. Rather than starting to design fully modular hardware and software for a creation environment, we decided to go for a performance system with a synthesis engine based on audio loops and audio mosaicing (a type of audio synthesis akin to granular synthesis already explored by members of our research team, cf Section 5.3.1).

An early simplified prototype of *Simone*, not as a collective instrument but as a solo web interface featuring audio loops and

audio mosaicing was presented during the Ircam open day in 2023. Using this interface, users could record an audio sample, hear it being transformed by audio mosaicing, play audio loops of the result and interact with synthesis parameters using sliders (cf. Figure 5.1).

Figure 5.1: Presentation of an early version of *Simone* during the Ircam open day 2023.

Early sketches of the system already mention our objective to create a "true" collective instrument. Being confronted with previous examples of distributed instrumental setups or distributed systems that used audio inputs was instrumental in this early prototyping period. We were mainly influenced by the technical setup of Weinberg's *Voice Networks* [45] and the way the band *The Hub* composed musical pieces by defining communication protocols between its members (cf. Section 5.2 and Figure 5.2 and 5.3).

[45]: Weinberg (2005), 'Interconnected Musical Networks: Toward a Theoretical Framework'

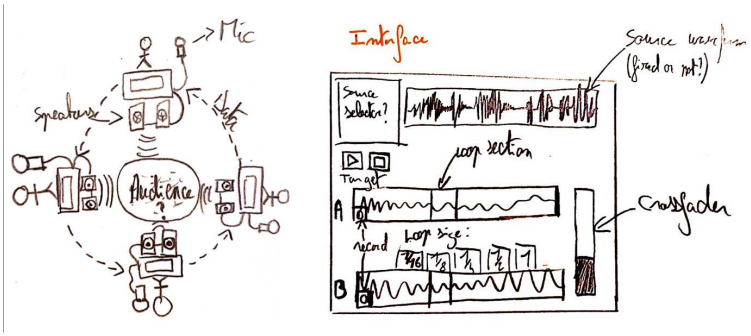


Figure 5.2: An early sketch of the technical setup and interface for *Simone*.

From the start, *Simone* has been conceived as an instrument geared toward Free Improvisation (also called "free music" or even "non-idiomatic improvisation"), with an aesthetic influenced by sound collage, *musique concrète* and noise music. Unlike other forms of improvisation that follow an idiom, for instance those of jazz

Figure 5.3: A group of people playing Voice Networks. Photo ©Gil Weinberg.

or Indian classical music, and often rely on a particular harmonic or modal framework, free improvisation has been described by musician Derek Bailey as having "no prescribed idiomatic sound". Bailey adds that "the characteristics of freely improvised music are established only by the sonic-musical identity of the person or persons playing it" [112].

[112]: Bailey (1993), *Improvisation*

While this aesthetic orientation reflects some personal preferences as a musician, we also see Free Improvisation as a fertile ground for research in the context of this work for multiple reasons: 1) We aim to observe what elements foster collective interaction when using a distributed instrument outside of the presence (in idiomatic musical practice) of a predetermined structure that works as "a constraint and a resource for intra-personal coordination (for the soloist) and inter-personal coordination (for all the members of the group)" [113]. 2) The open-ended nature of free improvisation leaves room for a plurality of strategies and musical outcome as Saint-Germier et al. note, "a same ensemble in the same context can generate extremely different musical contents on the basis of the same general collective intention of freely improvising music together" [113]. We expect that users of *Simone* might explore different strategies that could help us to gain a more general understanding of our system uses. 3) By not prescribing any type of interaction, it gave us more flexibility in the design of our instrument and on the forms of distributed interaction that could be imagined.

[113]: Saint-Germier et al. (2021), 'Joint Improvisation, Minimalism and Pluralism about Joint Action'

[113]: Saint-Germier et al. (2021), 'Joint Improvisation, Minimalism and Pluralism about Joint Action'

Ultimately, we identified *Simone* as a tool to explore the ap-

propriation of a distributed instrument by groups of users. While the NIME (New Interfaces for Musical Expression) community has for several years been interested in questions of instrument design and appropriation [82, 114], most of the works in this field concerns practices where musicians use their own interfaces without necessarily being collectively connected by a network. The question of instrumental design of networked musical systems in *co-located situations*, and the learning and collective appropriation of such *distributed* systems by groups of multiple performers playing simultaneously, still needs to be further explored.

[82]: Zappi et al. (2014), 'Dimensionality and Appropriation in Digital Musical Instrument Design'
[114]: Magnusson (2009), 'Of Epistemic Tools'

Thus, we consider musical creation and more particularly musical improvisation as a stimulating situation where decisions and changes made during the design of the artifact can produce effects that cannot be fully anticipated. Hence, we assume a personal approach to design that is informed by our experience as users and designers of musical interfaces as well as our personal aesthetic preferences. This implies that the design of our system carries a fair load of personal choices. In return, we employ a reflexive point of view in the analysis of the qualitative data we collected by observing how these personal choices interact with users' experience.

While the design and development of a musical instrument is in itself a contribution of our work, as already introduced in Section 1.2, there is debate in the design community on the epistemic value of design artifacts and whether or not we can devise general knowledge from it. For example Gaver asserts that "however valuable generalised theory may be, [...] it is the artefacts we create that are the definite facts of research through design"[61] and Zimmerman and Forlizzi instead suggest that "the artifact functions as a specific instantiation of a model - a theory - linking the current state to the proposed, preferred state" [62]. Following these remarks, we follow a research through design approach as "forms of making centred around knowledge and understanding that has formed in relation to a certain kind of judgment characterized [...] by starting from the particular" [60]. Hence, we aim that our system serves as a tool from which can emerge insights and questions on a wider scale on the design of a collective instrument and on distributed musical interactions. Moreover, following Findeli's statement that "the 'conception' part is only one of the two main moments or constituents of a design

[61]: Gaver (2012), 'What Should We Expect from Research through Design?'
[62]: Zimmerman et al. (2008), 'The Role of Design Artifacts in Design Theory Construction'

[60]: Redström (2021), 'Research through and through Design'

project, the 'reception' part being the other" [15], the instantiation of our prototype in actual situations is also one of our objectives.

[15]: Findeli (2012), 'Searching for Design Research Questions: Some Conceptual Clarifications'

Hence, the overall objectives of this work are: 1) to design of a distributed digital music instrument oriented toward musical improvisation and featuring collective interaction, 2) to observe the appropriation of this instrument by groups of expert users (i.e. improvising musicians with digital means) 3) to reflect on the design process of this instrument through the observation and interviews of users.

5.2 Background and Inspirations

In this section, we present a non-exhaustive list of some examples of networked music systems that served as direct inspirations in the design of *Simone*.

The pioneering and paradigmatic example of such use of the network is the band *The Hub*, founded in 1986 following the *League of Automatic Music Composers* [4, 115]. Although each musician in the group controlled a system of his own making, their performances were based on the exchange of information via a local network according to a communication protocol defined for each piece. For example, in the piece *Waxlips* (1991), each musician receives individual notes from the other band members via the network. This musician must first play the note received, then apply an arbitrary transformation before sending it to another member of the group. We reused this idea in *Simone* by implementing different interaction scenarios that define communication channels between players and the server. *The Hub*'s experiments served as a model for many music ensembles, so-called *laptop orchestras*, in which the network infrastructure had more or less aesthetic importance [27, 116, 117].

[4]: Gresham-Lancaster (2017), 'A Personal Reminiscence on the Roots of Computer Network Music'
[115]: Stone (2021), 'Non-Mathematical Musings on Information Theory and Networked Musical Practice'

[27]: Collins (2010), *Introduction to Computer Music*
[116]: Trueman et al. (2006), 'PLOrk: The Princeton Laptop Orchestra, Year 1'
[117]: Wang et al. (2008), 'Do Mobile Phones Dream of Electric Orchestras?'

Following a distinction made by Rohrhuber [12], we can distinguish two types of structures for the sharing of information within a networked music system: either simultaneous access to a common, shared state, or the sharing and circulation of objects and information within the network.

In the first category, we can mention *Emupo* [118], an interface for collective musical improvisation developed in Max/MSP.

[12]: Rohrhuber (2007), 'Network Music'
[118]: Canonne et al. (2011), 'L'EMUPO: Une Interface Logicielle Pour l'improvisation Collective'

Intended to be played autonomously or to complement other instrumentalists, *Emupo* can be controlled by several users at once, each taking control of different parameters in the production of the same sound, giving rise to "intra-instrumental" interaction.

In the second category, the *powerbooks unplugged* ensemble [119] offers group performances based on circulation of distributed states containing code extracts, musician comments and messages triggering the production of sound events on the loudspeakers of any computer on the network. The example of *powerbooks unplugged* also highlights another advantage offered by the decentralized aspect of sound production: the possibility of freely choosing the way in which sound is spatialized, and a flexible number of musicians.

[119]: Rohrhuber et al. (2007), 'Purloined Letters and Distributed Persons'

The design of *Simone* is more influenced by the decentralized approach of the second category although it could be argued that some interaction scenarios could be imagined to make it pertain to the first category.

Some other examples of distributed instruments were influential to us in the way they used vocal inputs to control sound synthesis. Gil Weinberg's *Voice Networks* (2003) [45] is a collaborative music installation that allows non-expert users to take part in a collective music-making experience, highlighting the social aspect of group play. The installation comprises four stations arranged in a square, facing each other, with a screen in the middle. Each station is equipped with a microphone, a touchpad controller and loudspeakers. Participants can record sound loops with their microphones and apply sound transformations to them using the touchpad which controls a Max/MSP patch. The network is used to exchange sound loops created between participants. The technical configuration of *Voice Networks* and the prominent use of vocal loops served as inspiration for the design of our system.

[45]: Weinberg (2005), 'Interconnected Musical Networks: Toward a Theoretical Framework'

Designed by Max Neuhaus et al., *Auracle* (2004) [33] is a collaborative synthesizer accessible on the Internet. The system analyzes the sound captured by each user's microphone and extracts several types of data at different levels (audio descriptors; classifications of vocal "gestures" by principal component analysis) to control a synthesizer. This idea was used as an inspiration in the design of the audio engine of *Simone*. *Auracle* is designed as a system that responds to user activity and seeks to encourage non-verbal com-

[33]: Freeman et al. (2005), 'Auracle'

munication and dialogue through a musical experience accessible via an interface aimed at a non-expert audience.

5.3 Design Overview

Inspired by some of the above-mentioned work, we have developed *Simone*, a distributed instrument for co-located collective improvisation.

Our main objective was to design a collective music creation system that encourages improvisation, that uses voice in the sound synthesis process, and that relies on the exchange of information via the network. The system is conceived as an experimental ground to study various paradigms of collective interaction and their approach by musicians. It must therefore be quickly accessible, yet possess the necessary depth and flexibility so that they don't feel restricted in a context of improvisation and creation.

Simone implements different interaction scenarios (cf. section 5.3.2), which propose variations in terms of their interfaces and of the interaction topologies[46] implemented. Nonetheless, each scenario shares a set of common features:

[46]: Matuszewski et al. (2019), 'Interaction Topologies in Mobile-Based Situated Networked Music Systems'

1. The microphone is used as a medium through which sound synthesis is controlled.
2. Sound synthesis is based on the principle of audio mosaicing (cf. section 5.3.1).
3. The system uses a local network to share information between agents (i.e. users and terminals).

The type of transmitted information (synchronized clock, audio files, analysis data, etc.) depends on the chosen scenario.

5.3.1 Sound Synthesis and Vocal Inputs

Sound synthesis in *Simone* is performed using a technique close to granular synthesis [120] and concatenative synthesis [121] known as *audio mosaicing*[122, 123]. This technique, which can be seen as the audio analog to the more widespread concept of photomosaics (cf. Fig. 5.4), consists in reconstructing an audio signal (thereafter named the *model* signal) with elements from another signal (the

[120]: Roads (2004), *Microsound*

[121]: Schwarz (2006), 'Concatenative Sound Synthesis'

[122]: Zils et al. (2001), 'Musical Mosaicing'

[123]: Janer et al. (2008), 'Extending Voice-Driven Synthesis to Audio Mosaicing'

generator signal).

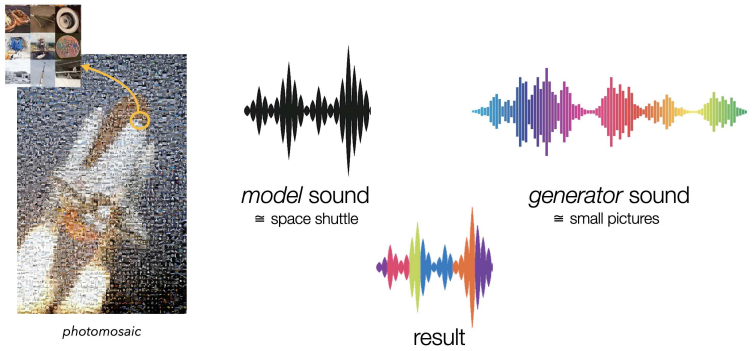


Figure 5.4: An example of a photomosaic and the analogy with audio mosaicing

To do this, the *model* signal is cut into small audio segments (called *grains*). For each grain, we look for the most similar grain in the *generator* file, according to a chosen metric and audio descriptors. The resulting sound is the concatenation of grains from the *generator* signal. The synthesized sound then follows the temporal evolution of the *model* signal but with the timbre of the *generator* signal.

In the case of *Simone*, the similarity between grains is computed according to the distance between their vectors of Mel Frequency Cepstral Coefficients (MFCC) which are multi-dimensional features used to describe an audio signal. MFCC are particularly well suited to the analysis of speech and are useful to capture timbral characteristics of sound [124]. Additionally, we compute the Root Mean Squared (RMS) energy of each grain of input signal which is mapped directly to the volume of the output signal to preserve the dynamics of the input signal. This was added after users' in a pilot study we conducted mentioned that they found it sometimes difficult to perceive a link between the *model* sound they recorded and the sound that was played by the synthesis engine.

[124]: Müller (2007), *Information Retrieval for Music and Motion*

We chose to rely on audio mosaicing in *Simone* for multiple reasons. First, while one of our objectives was to explore the use of vocal inputs to control a music instrument, we still wanted *Simone* to be accessible to users who may feel uncomfortable to use their voice in a collective improvisation context. Indeed, as Weinberg noted by observing how users would use *Voice Networks*, "regarding the choice of the voice as an intuitive and malleable gateway for creative and collaborative interaction, it was interesting to observe that although the voice is probably the most intuitive and prevalent means of communication in everyday life, some

participants were inhibited to use it in a public installation setting due to its "committing" or "revealing" nature or perhaps due to common "stage fright.""[45]. Because in *Simone* the voice is completely altered by the mosaicing process, we assumed this would mitigate this effect.

[45]: Weinberg (2005), 'Interconnected Musical Networks: Toward a Theoretical Framework'

Moreover, because audio mosaicing eschews most traditional musical concepts such as rhythm or harmony, we presumed it would be well adapted to the task of free improvisation in which the focus is put on concepts of timbre and textures [125, 126]. We also presumed it would foster a more exploratory stance when playing and provide a low barrier to entry and a sense of learning.

Finally, audio mosaicing is relatively easy to implement compared to other synthesis techniques, which allowed us to focus on interface design and made early prototyping quicker and more flexible. While similar in results to timbre transfer methods by machine learning [127], audio mosaicing requires much lower computing power and can be easily implemented using the Web Audio API making *Simone* readily available on a large variety of devices (computers, laptops, tablets or even mobile phones).

[127]: Mor et al. (2018), 'A Universal Music Translation Network'

5.3.2 Interaction Scenarios

In order to guide collective interactions and inspired by The Hub's definition of communication protocols as a method of composition, *Simone* is declined in three interaction scenarios. Each scenario corresponds to a different network arrangement that defines the type of data that is exchanged between users, the path that data follows during these exchanges and the roles adopted by users. The three interaction scenarios detailed below are called: *Drum Machine*, *Clone* and *Solar System*.

These scenarios were designed to showcase different levels of intensity in the network's influence and channels for mutual interaction between players. In the *Drum Machine* scenario, no direct mutual interaction happens between players as they only share an underlying rhythmic structure imposed by the network and cannot communicate information to each other. In the *Clone* scenario, mutual interaction only exists in a delayed manner as the sound recorded by a player before playing only influences the other player during the performance. In this case, communications channels

exist but are only open before performance. Mutual interaction is at its strongest in the *Solar System* scenario, where communication channels are permanently open during performance, as the player in the *sun* role has an active influence on the sound of the others.

Drum Machine

In this scenario, all users share a common musical clock. Users can record a *model* sound and choose a *generator* sound among a soundbank. They can also use sliders to change some synthesis parameters (cf. Section 5.3.3). A loop section is defined on the *model* sound. The loop section can be moved over the *model* sound signal. The length of the loop section on the *model* signal is restricted to a rhythmic grid based on this shared clock. The result can be seen as a sort of distributed drum machine. See Fig. 5.5 for a diagram showing the communication channels in this scenario.

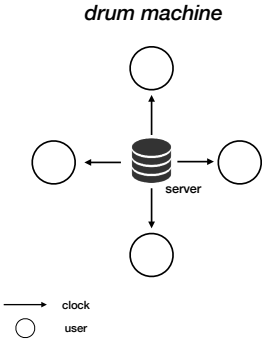


Figure 5.5: Diagram of interaction and data communication in the *Drum Machine* scenario.

Clone

At the start of this scenario, users are asked to record a sound using their microphone. This sound is then sent to another user to act as their *generator* sound. When playing, users will not be able to choose another *generator* sound. Hence, each user must learn to play with the voice of another user. In this scenario, users can record a *model* sound, define a loop section on the *model* sound and use sliders to change some synthesis parameters. See Fig. 5.6 for a diagram showing the communication channels in this scenario.

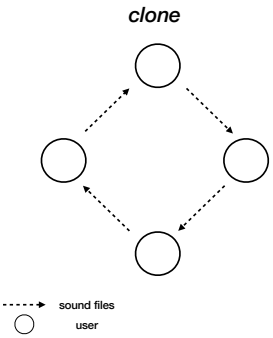


Figure 5.6: Diagram of interaction and data communication in the *Clone* scenario.

Solar System

Unlike in other scenarios, users are in an asymmetric configuration. One of them (called the *Sun*) is the only one able to record a *model* sound. The analysis data from this *model* sound is then simultaneously sent to all other users (called *Satellites*) to control their sound synthesis. On their end, Satellites may select a *generator* sound from a soundbank and act on various synthesis parameters. Hence, the same *model* sound recorded by the *Sun* is simultaneously reinterpreted by various *generator* sounds on the satellites. See Fig. 5.7 for a diagram showing the communication channels in this scenario.

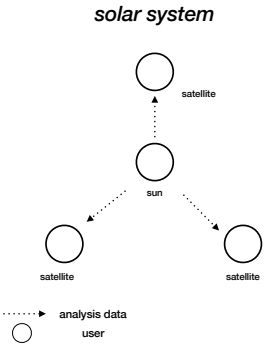


Figure 5.7: Diagram of interaction and data communication in the *Solar System* scenario.

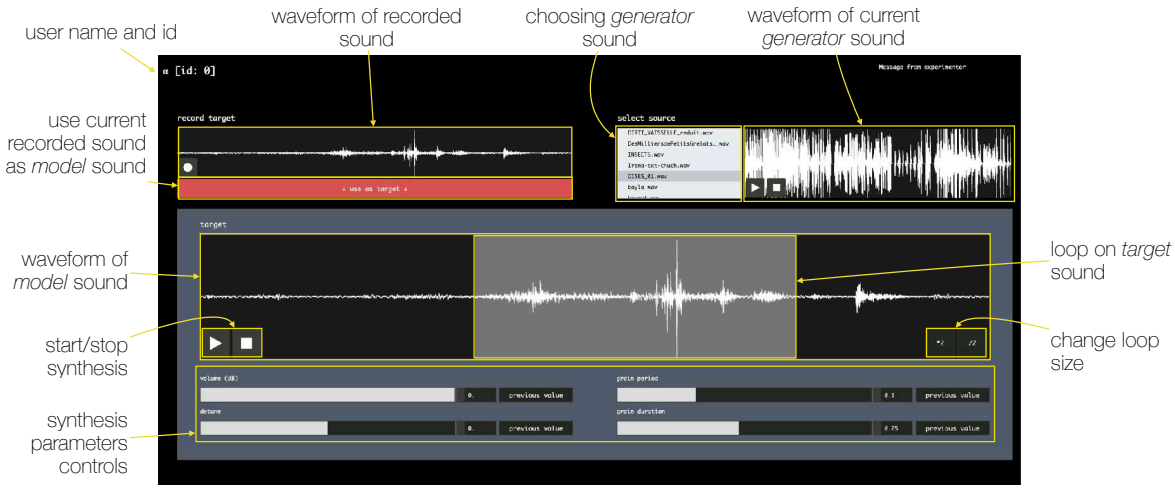


Figure 5.8: Interface of *Simone* in the *Drum Machine* scenario. Other scenarios' interfaces may differ and are adapted to each scenario's specific task

5.3.3 Interface Design

The interface underwent several iterations through testing and refinement. Particular attention was paid to limit its complexity by selecting carefully the number of interactive elements. This choice was made in order to facilitate the familiarization and appropriation of the system in a limited timeframe and to foster the agentivity of the participants in a context of collective improvisation. Several modifications were made to the interface after a pilot study. This includes adding color panels, making important elements bigger and by having their placement more coherent with the intended way to use the interface (cf. Figure 5.8 for a screenshot of an early version).

The interface may differ slightly according to the different scenarios of the system. Figure 5.8 shows the interface in the *Drum Machine* scenario (Interfaces of the other scenarios are shown in the appendix section of this chapter). Apart from these slight differences, the interface generally comprises the following elements:

- A recording area from which the user can trigger the recording with the microphone, visualize the waveform of the recorded sound and trigger the use of this recording as their *model* sound (top left area in the figure).
- An area dedicated to the *generator* sound with a menu allowing the user to select a sound file among a soundbank, visualize its waveform and play it back (top right area in the figure).

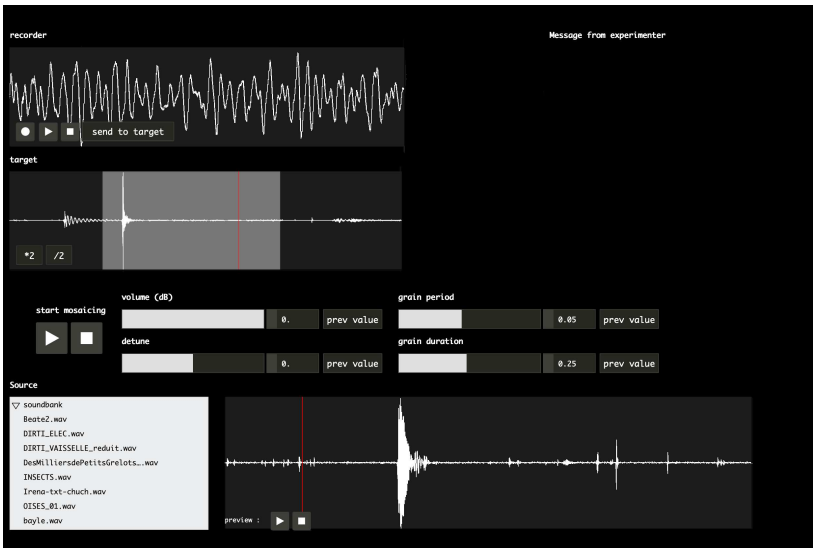


Figure 5.9: First version of the interface of *Simone* in the *Drum Machine* scenario.

- The main panel (at the center of the figure) shows the waveform of the *model* currently in use. The user can select a section to loop using the mouse. The selected loop can be of any length, or constrained by a time grid. The display of the *model* sound waveform was chosen to offer a balance between giving some information on the resulting sound and an invitation to explore and play. Indeed, while the waveform reflects the sound level of each section of the *model* sound and thus of the synthesized result, the audio engine also takes into account other dimensions of the sound, which means that the generated result cannot be fully anticipated, thus encouraging the user to explore.
- Finally, at the bottom of the interface, four sliders allow the user to modify various synthesis parameters: volume, pitch, period and duration of sound grains. Access to these few parameters, which are common to most granular synthesis-based instruments, enables a greater variety of instrumental play and opens up important possibilities for complementarity and dialogue between users.

5.4 User Study

5.4.1 Participants

We recruited 9 participants (aged 20 to 40) via personal and professional network. Participants were grouped in groups of 3 for a workshop session depending on the moment they were available.

They had no previous experience of playing with each other. We specifically sought participants with a practice in musical improvisation and with basic knowledge of electronic music instruments (i.e. having already used a Digital Audio Workstation and/or hardware synthesizers and familiarity with the vocabulary and concepts of digital audio synthesis), to ensure that the learning time of the interface would not take too long and that they would be comfortable to improvise with the other participants. Participants were not remunerated but were given snacks and drinks during the experiment.

5.4.2 Setup

Groups of participants were invited in our laboratory in a room where the experimental setup was installed.

The experimental setup consists of multiple stations (one for each participant) set up around a table (cf Figure 5.10). Each workstation is comprised of :

- the participant's computer (laptop) with a web browser connected to *Simone*,
- a Shure SM58 microphone mounted on a stand,
- an audio interface to connect the microphone to the participant's computer,
- a pair of loudspeakers (Creative Inspire T10) with one turned towards the center of the table and the other one turned toward the participant's workstation to provide audio feedback,
- a pair of closed headphones (used only in the first testing phase of the workshop (cf. Section 5.4.3).

The main experimenter's computer was used to create a local network and to launch the application's server. It was also used to send messages to participants whenever necessary during the improvisations using a dedicated web page. A camera and an audio recorder were installed to record the whole workshop.

The soundbank used by participants in the workshop included instrumental sounds (a piano piece from Bach, a vocal piece from Monteverdi), a drum loop, 2 vocal sounds (one of a person speaking and one of a person whispering), 2 field recordings of percussion

made by the first author, field recordings of insects and birds, and 2 files of electroacoustic music samples.

Figure 5.10: A picture from one of the workshop sessions showing the experimental setup.

5.4.3 Procedure

Before starting the workshop, participants are given an information notice detailing the context, goals and the experimental procedure. They are also asked to sign a consent form to allow us to record the audio and video of the workshop.

After that, the main experimenter explains the general structure of the workshop and introduces *Simone* and more especially the concept of audio mosaicing to the participants. Then, participants are brought to a solo version of *Simone*. During 10 minutes they can freely use this interface using headphones to familiarize themselves with the controls and the way audio mosaicing works. This first step also aims at allowing them to acquire a basic sonic vocabulary.

After these 10 minutes of discovery of the interface, the workshop consists of three sessions corresponding to each of the interaction scenarios described in section 5.3.2[1]. For each of these scenarios participants are asked to perform a collective improvisation for 7 to 8 minutes. Following the improvisation, the main experimenter engages a group discussion to record their immediate impression by asking "Do you have any remarks? What is your immediate reaction?". After these three sessions, the main experimenter starts a semi-structured group interview of around 30 minutes with the participants. Questions in this final interview focus on asking participants to detail how they used the instruments and which strategies they employed during the improvisations, especially in relation to the collective and network aspects of the sessions :

1: Two of the groups performed the session in the order 1) *Drum Machine* 2) *Clone* 3) *Solar System* and one of the group in the order 1) *Clone* 2) *Solar System* 3) *Drum Machine*

- How did you use the interface? Do you have any remarks about the interface and the system in general?
- From a musical point of view, can you describe what you did, the strategies you employed?
- What do you think of the musical result you produced?
- What did you think of the collective and networked aspects of the system compared to more traditional individual musical practice?

We decided to perform group interviews rather than individual interviews as we were interested in capturing how participants perceived interpersonal dynamics within the workshop. Moreover, group discussions allows participants to pick up on other participants' comments and impressions thus highlighting converging and diverging opinions and points of view, and providing insight on social relationships within the group [128].

[128]: Frey et al. (1991), 'The Group Interview in Social Research'

Finally, in the last phase of the workshop, participants are asked to collectively choose the improvisation they think is the best or the most convincing. They are then asked to perform a self-annotation of this improvisation using the A^3PM application (cf Chapter 3). Upon connection to A^3PM, participants fill in their ID. They are then brought to a test page to experiment with the interface and to make sure everything is working correctly. Then they start the self-annotation task. The interface takes the form of a triangle whose vertices are each associated to one of the following categories: collective play, individual play, lost/confused/cannot tell (cf. Figure 5.11). Within the triangle is a white dot that participants can move using the mouse in one of the three categories delineated by colored zones around the corresponding vertex. Position of the white dot marked with a timestamp relative to the start of the audio recording is then regularly (every 50ms) sent to the server to be written in a text file.

After finishing the self-annotation task, participants are asked for final remarks and thanked for their participation. In total the workshop lasts for approximately 2 hours.

We recorded audio and video using a camera and audio recorder. Recordings of the interviews were automatically transcribed and transcriptions were corrected by hand. Video excerpts and full audio recordings of the sessions are available on the companion website: `https://alienorgolvet.com/thesis/chapter`

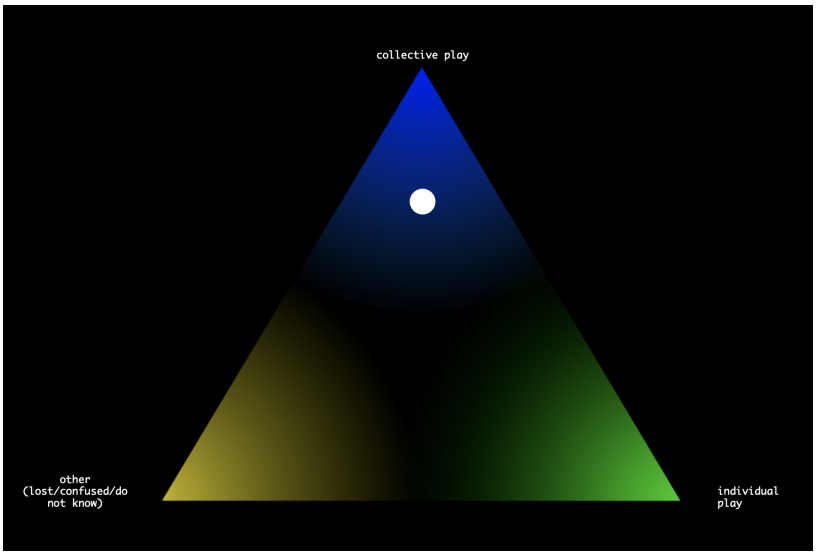


Figure 5.11: The triangle interface for self-annotation

`-simone-collective.html` Interviews were made in French and English depending on groups of participants. French interviews were translated to English for the writing of this paper.

5.5 Results

5.5.1 Method

In this section, we first present a quantitative analysis of self-annotations from which we computed various statistics about participants' proportions of collective play (section 5.5.2).

Then, in order to identify shared themes across the subjective experience of our participants, we analyzed our interview data using reflexive Thematic Analysis (TA) as described by Braun and Clarke [129]. TA is a flexible method for analyzing qualitative data that consists, after familiarization with the data, in a first step of coding (i.e. assigning a "code", a short label, to sections of the data) and a second step of theme development (i.e. identifying theme running across the codes). Our approach was inductive, meaning that codes were developed from the data itself and not from a predetermined coding scheme. We followed a critical realist framework, assuming an independent reality but acknowledging that this reality is not directly accessible due to the subjective and located nature of participants' experience and discourse in regard with their personal background and the broader socio-cultural context. In addition, reflexive TA specifically acknowledges

[129]: Terry et al. (2017), 'Thematic Analysis'

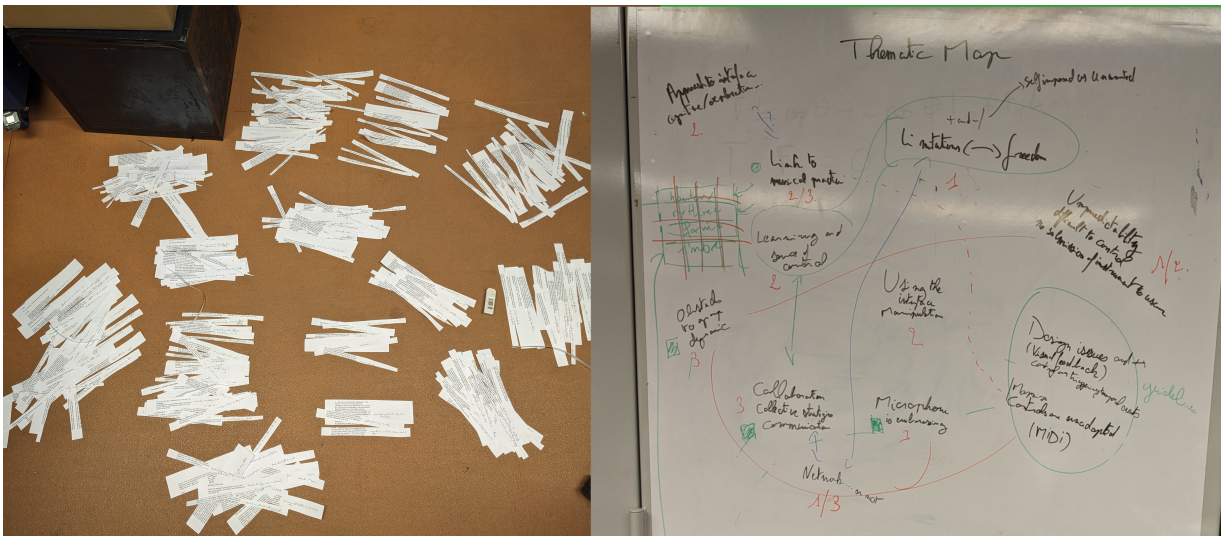


Figure 5.12: Picture documenting the analysis process. Left: regrouping codes into proto-themes. Right: the thematic map displaying proto-themes and relationships between them.

the subjectivity of the coding and the interpretative task by the researchers.

The process of coding, using both semantic and latent codes, was undertaken independently by the first two authors (cf Figure 5.12 for a picture documenting this process). These codes were then regrouped into proto-themes. Finally, we compared our proto-themes and discussed together to converge toward three main themes: (1) Participants' relationship to constraints and freedom (section 5.5.3) , (2) their description of the process of familiarization and learning of the instrument (section 5.5.4), and (3) their description of group dynamics with regard to the networked nature of the instrument (section 5.5.5).

In the following, participants are referred to as P1 to P9. The three groups were composed of : P1 to P3, P4 to P6 and P7 to P9.

5.5.2 Quantitative Analysis of Self-Annotations

We preprocessed the self-annotation data by linearly interpolating positions on the interface between time points with a resolution of 10Hz to guarantee that annotations from participants from the same workshop would be compared on the same time grid.

We first computed the proportion of time each participant indicated they were playing collectively, individually and "other"

	group 1	group 2	group 3	average
joint collective play	22.0%	13.8%	11.4%	15.7%
mutual collective play	51.7%	51.8%	67.6%	57.0%

Table 5.1: Proportions of joint and mutual collective play for each group and on average.

(meaning their annotation was in the "cannot tell" zone of the interface or in no particular zone). We also computed the average values over all participants. The results are presented in Figure 5.13. On average, participants indicated that they were playing collectively for 50.9% (sd: 5.9%) of the time. It is worth noting that "individual play" can not only indicate moments when participants were unable to engage in collective play but can also be a viable interaction strategy in musical improvisation

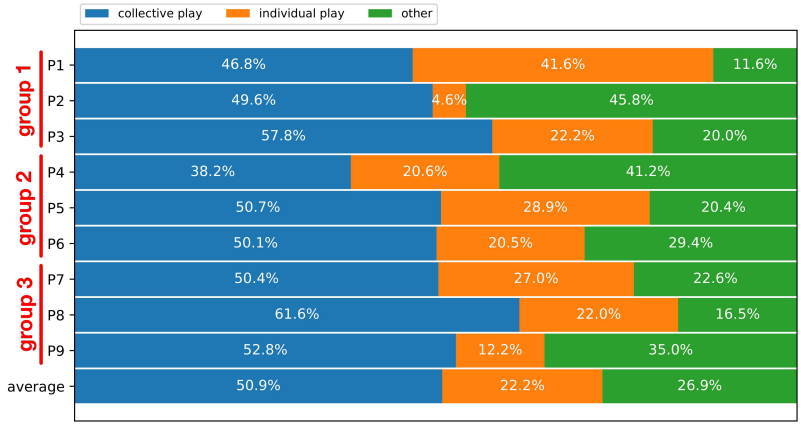


Figure 5.13: Proportions of time participants indicated playing collectively, individually and other. Last line is average values over all participants

To verify if these feelings of collective play were shared among participants, we computed the proportions of joint collective play and mutual collective play for each of the three groups of participants. We call "proportion of joint collective play" the proportion of time the three participants indicated playing collectively simultaneously and "proportion of mutual collective play" the proportion of time that at least two players indicated playing collectively at the same time. We also computed the average value over the three groups of these two variables. Results are presented in Table 5.1.

On average participants indicated playing collectively all three together simultaneously only 15.7% of the time. However, the proportion of mutual collective play shows that for 57.0% of the time, at least two of them indicated playing collectively at the same time. This suggests that while playing collectively as a group of three was a relatively rare phenomenon, participants were at least able to engage in collective play with another player for the majority

	third 1	third 2	third 3
proportion of collective play	28.7%	58.1%	65.9%

Table 5.2: Average (over participants) proportion of collective play over each third of the sessions.

of the time. This is corroborated by a comment from P5 in the interviews : "And I think like the number [of players] favors the one to one situation. Because a trio is always like [...] you engage with one, with one, with one, with one. . . [...] I think in three like it's really easy to just pair with one at a time."

Finally, we wanted to verify whether familiarity and learning over an improvisation improves ability to play collectively. To do that, we computed proportions of collective play over each third of the tracks and we averaged it over all participants. Results are presented in Table 5.2. On average, participants indicated playing collectively only 28,7% of the time in the first third of the annotated improvisation while this number rose to 58.1% in the second third and 65.9% in the last third. This shows that, at least during the annotated improvisations, participants increasingly felt comfortable playing collectively over time. As expected, the system requires a first period of exploration and learning at the beginning that tends to limit the collective interaction. Then, we observe that collective interaction is predominant in a second period.

5.5.3 Theme 1: The Relationship to Constraints and Freedom

Interviews with participants revealed a complex and ambiguous relationship to their (often contradictory) perception of constraints and freedom during the experiment. Most participants reported on the various types of constraints they perceived during the workshops. These constraints were of different kinds. Some of them were built-in the instrument's interface, some of them came from how sound synthesis works and some others were consequences of the different scenario's explicit constraints.

Regarding *scenario constraints*, some participants interestingly noted that the limitations did not restrain their agency. For example, when commenting on the *Solar System* session, P9 says that "there was still enough parameters to focus on". Similarly, after the same

scenario, P5 commented that they "thought it would be harder to make different stuff because we have the same target so but in fact the sources are very different so if we select different sources then we are... it's easier to complement each other I would say".

This comment points toward the fact that constraints were actually perceived as helpful. A limited set of parameters meant "less to think of, in a good way" (P4) and would reduce the overall cognitive load of the interface and guide exploration and understanding of the instrument's possibilities. This fact is highlighted by comments from P1: "with fewer parameters, we at least know what they do", P4: "I also like the limitations. I think it's easier to explore in a way and not be too focused on the possibilities of the interface" and P5: "in this case having like the *model* [sound] fixed makes you really think about what the *generator* [sounds] are, how to use them".

Constraints also proved to be fruitful for collaboration between participants: a reduced set of parameters increased P1's mutual attention to the group's actions and scenarios' constraints forced P4 to think about collaboration and to "figure things out together". On a musical level, scenarios' constraints also increased feelings of synchronization (P3, P7).

However these constraints were sometimes also perceived negatively. For example, in the *Drum Machine* scenario, P5 reports that they felt the rhythmic "grid was quite limiting" and that "the imposition was too strong". In the *Clone* scenario, P2 mentioned that they felt "stuck" because of the *generator* sound they were assigned and that "it was quite frustrating not to be able to have a sound that I liked but that also contributed to something [...] that was happening." Finally, in the *Solar System* scenario, P3 (who assumed the *sun* role) felt that they missed the ability to send different data to each of the *satellite* players.

Regarding the *interface*, many comments from the participants suggest that they perceived the instrument as rich and complex, offering a "lot of possibilities" (P6), "a lot of choices" to make (P3) or "quite of lot of things to do" (P3) when playing. This richness of the interface is reflected in the rich variety of sound the instrument is capable of producing or in the flexibility it offers to create different effects with a rather low number of parameters.

For instance, as P4 noticed that the two other participants used the same *generator* sound at some point, they remarked "I think it was still quite rich. Like there was still difference to it".

However, this complexity was not always perceived as positive. Some participants saw the instrument as being too complex or offering too many parameters to manipulate, even seeing this complexity as a "risk" (P6). This complexity would keep players in a permanent state of exploration and testing that would prevent them from focusing on the task of collective improvisation (P1, P5).

It is worth noting that what is perceived as a constraint is not absolute but sometimes heavily dependent on the participant's own experience and background. This fact is most visible when P2 and P3 addressed the topic of rhythm within the *Drum Machine* scenario. The rhythmic grid constraint was felt as enjoyable by P2 as they reported that they could "feel a pulse". We even noticed that this participant's foot was tapping along the beat during the session. They perceived this constraint as satisfying, linking it to their own musical practice: "maybe it's because I'm a percussionist, I liked it". On the other hand, P3 did not perceive the "rhythmical impact" of this constraint as the micro-temporal events of granular synthesis went against their conception of musical rhythm as "a hierarchy of representation" of "pulse, rhythm, tempo".

This very subjective reception of freedom and constraints is also reflected in the way some participants implemented personal strategies either by working around constraints they disliked or by self-imposing constraints. For instance, P5 and P9 described trying to find ways to go around the inherent rhythmical nature of loops in *Simone* to try to "do something more continuous or different" (P5) while P3 reported that they chose to focus only on "two or three *generator* sounds" (among 12) to avoid spending too much time exploring the possibilities of the whole soundbank.

To summarize, all these comments map out a fairly complex and ambiguous relationship to constraints within *Simone*. For participants, full satisfaction with the instrument and the conduct of collective music endeavors requires the design of, as P5 describes, "a nice balance between freedom and limitations", something rendered arduous by the fact that this relation to constraints is

very much subjective.

5.5.4 Theme 2: Learning and Taming the Instrument

Interviews revealed that over the course of the workshop participants had a sense of learning how to control an instrument that was often perceived at first as hard to control. This conscious process of learning was highlighted in the interviews:

> No, there was definitely like a learning [...] So it's more like ways of playing evolve over time because you try new stuff and suddenly you have a grip on what you're doing. So ok, this makes that so I'm going to do it again or doing slightly differently. (P6)

Moreover, getting this "grip" on the instrument proved to be a necessary condition before participants started collaborating. For example, P6 reported that "at some point I remember it was towards the end of a performance I was kind of, felt like, okay I kind of get what is happening so now let's try to interact" (P6). This requirement of a certain degree of appropriation as a condition for collaboration is described by several participants (P1, P3, P6) who originally engaged in a phase of testing and exploration, but quickly realized that this prevented them from engaging with other players:

> I had so much choice of *generator* sound [...] I think I didn't take enough time to settle on one sound and think ok now let's choose this and play with finer parameters and listen to what's going on with the other [participants]. (P1)

At first, participants indeed described *Simone* as unpredictable or reacting in unsuspected ways. For example, P1 reported it as "difficult to anticipate", and P5 described it as "stochastic, chaotic" and with the ability to "change completely the environment if you mess with [it]". It appears that this feeling of unpredictability has multiple roots. First, some participants had difficulties gauging their influence on the instrument's output (P1, P9): "What I recorded... to me, it didn't seem to have much effect on what I was making" (P9). Second, some participants felt that they could not rely on familiar combinations of parameters as two very close

states could yield vastly different results : "I'd change things and go back to a previous state and it wasn't the same. I felt like I couldn't repeat it." (P1).

To overcome these problems, participants elaborated different strategies for creating a vocabulary around the instrument. This is first reflected in their description of how they approached the instrument to try to understand how it works. While P5 and P7 described taking a more intuitive/inductive approach: "At first [...] I was kind of trying to get a feel of it [...] more like poking around and thinking "ok this does this, this does that". Not asking myself like, "what does it mean technically?"" (P7), P4 reported on the contrary that "there is something about the interface that makes me want to approach it in a cognitive way" and that they tried to understand "exactly what's happening" from a technical standpoint.

From another perspective, it seems that participants' process of making sense of the instrument happened through the prism of their own musical background and preconceptions. This is reflected for example in their reliance on traditional music concepts such as tonal harmony and rhythm when describing their approach but also as they compared *Simone* to traditional instrumental practice. While some participants' familiarity with granular synthesis (P9) or with working with sound collage (P7) seemed to ease their learning curve, others were startled by the way *Simone* would work very differently from a traditional instrument: " There's a way of triggering sound that's not a visible gesture like tapping on an instrument and it comes out. It's more based on the dynamics of the waveform, on granularity. (P3)"

To summarize, all these comments highlight that participants implemented different strategies to overcome what appeared at first to be unpredictable or complex features of the instrument and that they considered this learning phase as a condition for collaborating on the improvisation.

5.5.5 Theme 3: Networked Group Dynamic

Participants described the strategies they employed to play collectively with the other players. While this collaboration was

sometimes fragile and hard to establish due to various obstacles, it was nonetheless perceived as satisfying. Participants' discourse focused on human to human interaction but also on their specific relationship to the network that varied greatly depending on the scenario.

Although musical improvisation generally does not prescribe any specific type of interaction [113], participants' main strategy during the workshops seemed to be to complement each other's sounds (rather than disrupting other players' actions or simply playing independently from the others). For instance, as P4 was about to record their sound previous to the *Clone* sessions, they made the remark that they were influenced by what P5 recorded just before: "I'm thinking now of not adding vocals, because [P5] did". This complementarity was also present when participants were playing :

> At the end of the third session we were more into very environmental sounds. I mean, there were bird sounds. And then I thought, ok, I'll try to go in that direction too, I'll try to make sounds that are very. . . Yes more towards noises or things that are evocative of sounds that are a bit more natural. (P7)

This matter of complimenting others' sound was mostly influential on the choice of *generator* sound participants would choose, but they also sought out to complement musical structure at the micro level: "Sometimes some loops were coming back in a way. So I tried to adjust and say, you want to do loops, so I'm going to try to jump in and see whether I can add to it" (P6).

At the macro level, participants seemed concerned with the building of a mutual construction which was made easier by working with "gradual" (P5) changes. Participants' perception of the group dynamic is aligned with a common view of free improvised music as a mutual construction, perpetually moving, in danger of collapsing at any moment but whose collapse is essential to uphold a sense of creativity. But because participants' were also in a learning process regarding *Simone*, they favored a cautious approach where disturbing the current state was seen as creatively destructive more than productive:

[113]: Saint-Germier et al. (2021), 'Joint Improvisation, Minimalism and Pluralism about Joint Action'

> **P4**: It would be interesting to see if we have the same idea on which moments were good surprises. But it's very fleeting and it's like suddenly happening and then suddenly it's gone because someone did something.
>
> **P6**: Or sometimes it's nice for a while and then it becomes boring. [...] There's this difficulty of finding a zone where it seems to work and then it doesn't seem to work anymore and you have to move. I mean it's what free improvisation is about. But here because we're not very familiar with the instrument or the thing it's even harder I would say.

Yet, this challenge that necessitates quick adaptation was deemed interesting by other participants:

> **P7 (sun)**: I was afraid to go in directions that would not be satisfying to them you see. Or changing a lot of stuff. In fact it can be disturbing. You're trying to build something, for example and I'm going to modify... bring a bit of chaos in that
>
> **P8 (satellite)**: But you have to readjust. And that's interesting. As you say, it makes you change your plans. Sometimes, you make a little mechanism for 30, 40 seconds, thinking "I'm going to go towards here", and then in fact, [the *sun* player] shortens our loop, you see, and as a result, we no longer have access to the same... You have to readjust pretty quickly. I think it's interesting, actually.

This cautious approach was also reflected in the way participants would manage volume during the workshops, often playing very low leading to a prisoner's dilemma situation where everyone is waiting for the others to be more daring. Short interviews between sessions were also the place where the group would discuss readjustments to be made and group strategies to follow : "Maybe we should all just put the volume really high" (P4).

Participants also reported on several elements that were an obstacle to group dynamics and collaboration. This includes the fact that the interface demands too much focus on the screen hence

preventing interpersonal gaze which could be useful to transmit cues (P1, P2, P4, P6, P7), the difficulty to hear other participants due to loudspeakers disposition creating a "distance" and a "separation" (P9) between players (P1, P2, P3, P6, P8, P9), the difficulty to single out contributions because of the homogeneity between produced sound (P1, P4) or the process of recording which was perceived as as embarrassing because it disrupted the shared sonic space (P1).

Underlying all collective interaction in *Simone* is the network. But depending on the scenario, it was not always perceived. From the comments gathered, it appeared that the participants were mostly aware of the presence and actions of the network in the *Solar System* scenario in which an action from a participant had an influence on another participant was directly perceptible:

> It is in the [solar system scenario] that the network aspect appears clearly. It was the first time I had the impression that... a real feeling of synchronization with the others. [...] There is an obvious action from one computer to the other. (P1)

For the other scenarios where players cannot send data to each other, the network's action was barely noticed or was seen as too "simple" (P5) or as "primitive" (P3) which made some participants wish for more interactivity between players in the form of increased distribution of parameters:

> Maybe in [the *Drum Machine*] case giving... like multiplication, loop windows divisions, giving more distributed controls on rhythm may be interesting. Even giving subdivisions or beat modifications. It could change a lot. (P3)

The fact of distributing control of parameters not only increased awareness of the network but also awareness of the other players, giving a more acute sense of group interactions:

> Because I'm wondering whether there's really an action that's just been done by the other person that's affecting my sound or not. [...] In this interaction, I have the impression that I'm really thinking in terms of people's actions, and that I'm wondering what actions are the

> other people in this network producing? There's a point where it goes a bit beyond music, [...] it's something where there's really a question of network and... And *inter*-action between... the players that I find quite strong. (P1)

In some extreme cases, the interaction agency enabled through the network could also be perceived as unbalanced. For instance in the *Solar System* scenario, where the *sun* player can decide whether to mute the *satellite* players, some satellite players felt such action as "brutal" (P7) or even "infantilizing" (P9). Therefore, any action in the network can be seen either as positive or negative, and careful actions are required by all players to create a balanced and creative environment. This represents one of the challenges of such an approach. Globally, despite the few cases we just reported, participants reported a rather collaborative and constructive interaction between themselves.

5.6 Discussion

5.6.1 Shared Agencies in Collective Musical Interaction

In this chapter we have always referred to *Simone* as "an instrument", using the singular, which of course can raise the question of whether *Simone* constitutes a single instrument played by multiple players, or multiple separate instruments connected to each other. In his article on "multi-user instruments" [130], Jordà identifies interdependence as one of the main properties that characterizes such an instrument, arguing that "if no mutual interaction is allowed, the concept of multi-user instrument is definitely debatable". Hence, the question becomes how to define and design such mutual interaction, which Weinberg consider achievable (in what he calls Interconnected Musical Networks) "only by constructing electronic (or mechanical) communication channels among players", in which case "participants [can] take an active role in determining and influencing, not only their own musical output, but also their peers" [45].

[130]: Jordà (2005), 'Multi-User Instruments'

[45]: Weinberg (2005), 'Interconnected Musical Networks: Toward a Theoretical Framework'

At first, such a position seems to be confirmed by the fact that this perception of mutual interaction over the instrument was also shared by our participants as they felt the presence of the network

more strongly in the *Solar System* scenario while it was felt as weaker in the other scenarios. However we believe that *Simone* still qualifies as a single instrument even in cases where no direct connections are built between players (like in the *Drum Machine* scenario), as they are still connected over a shared structure (e.g. rhythmical grid) that has a strong influence on players. Indeed, this feeling of interdependency appearing from a shared structure is present in multiple musical traditions, an example of which is the Indonesian Gamelan often considered to be a single instrument played by many persons. We believe that an interesting avenue for developing further such an idea would be to provide the network with an even stronger role to the point that users of the instrument would perceive it as having its own agency.

Another interesting question lies in the way agency over different aspects of sound, or of musical parameters, can be established in such a collective interaction. This idea has already been identified by Jordà who describes the case of two persons performing on the same piano with one player playing the keyboard and another playing the pedals (thus affecting the timbre of the resulting sound) [130]. In many examples of multi-user instruments, like *Voice Networks* [131] or *88 fingers* [132], the design of the system gives an equal role over all aspects of the instrument to the players. In *Simone*, and in particular in the *Solar System* scenario, players are only able to control certain dimensions of sound synthesis, while being deprived of control on other dimensions and thus relying on other player's contribution (cf. Figure 5.14). Interestingly, such asymmetry was identified by one of our participants (P1, see section 5.5.5) as a concept that strengthened the feeling of interaction between players over a same distributed instrument. Such hierarchies and asymmetry need however to be carefully designed and thought of, as our results also show that losing control on some aspects of sound can lead to confusion.

[130]: Jordà (2005), 'Multi-User Instruments'
[131]: Weinberg (2005), 'Voice Networks'
[132]: Schnell et al. (2017), '88 Fingers'

5.6.2 Designing Constraints and Instrumental Agency

From the beginning, *Simone* has been designed towards what we called "expert users", defined as users familiar with digital music instruments and software. To that end, we decided to build a fully digital interface borrowing ideas from Digital Audio Stations

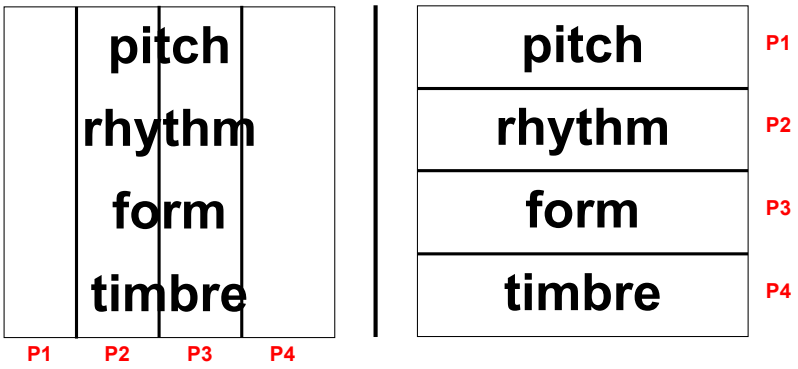


Figure 5.14: Two ways of designing the way of sharing control of parameters in a distributed instrument. On the left, each player has control on all dimensions of sound. On the right, each player can only control one dimension of sound, relying on the other player's contribution to act on the other ones.

(DAW) interfaces, with a limited number of parameters and with a straightforward mapping between available input/parameters and their effects (cf Fig. 5.15). For example, granular parameters in the interface like "detune" or "grain duration" are directly mapped to said detuning or duration of sonic grain, making it self-explanatory for any user familiar with granular synthesis.

Nonetheless, the different learning paths we observed during our experiments led us to reconsider the category of "expert user", as it appears difficult to gather users with vastly different baggage in a single category. Indeed, depending on their practice and experience, users brought varying expectations and different ways to make sense of the behavior of the technological artifact, which made us reflect on our initial assumptions and beliefs on the relative neutrality of some of our design choices.

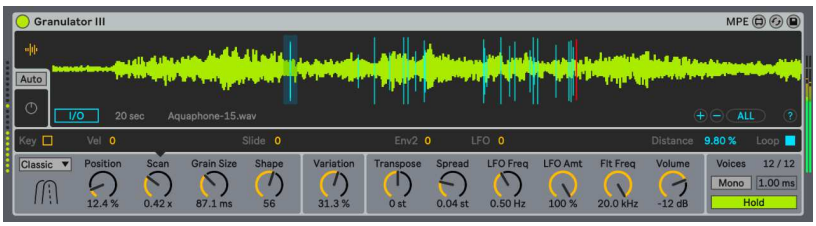


Figure 5.15: The interface of Granulator III by Robert Henke, a fairly popular granular synthesizer plugin for Ableton Live. Notice the similarity with the *Simone* interface: the display of the waveform, some of the parameters available (grain size, transpose)

Regarding design choices, our participants described the various constraints of the system as helpful for reducing the cognitive load of the interface and for guiding the process of familiarization and understanding of the instrument. The role of constraints in early stages of familiarization with a digital instrument has already been identified in earlier works with Magnusson [83] stating that "the main bulk of the time spent in learning the instrument involves building a habituated mental model of its constraints" and with Gurevich et al. [133] noting that the development of personal style "emerges both as a result of constraint [...] as well as in spite of constraint". As in our study, the latter also observes that users'

[83]: Magnusson (2010), 'Designing Constraints'

[133]: Gurevich et al. (2012), 'Playing with Constraints'

"interpretation" of constraints can heavily depend on context and users' background.

While these works focus on the case of individual instruments, our results also show that in a collective instrument, constraints are helpful to guide collective interaction. The specificity of *Simone* in that domain lies in the fact that beyond the instrument's inner sonic constraints, another layer of constraints is created by the interaction scenarios' rules.

Magnusson describes digital music instruments as "epistemic tools", noting that the act of designing such a system "entails the encapsulation of a specific musical outlook"[114]. *Simone* makes no exception and carries values of its designers as musicians, purposely avoiding the use of concepts from western music theory such as pitch or time signatures in favor of a more experimental approach. We may further state that *Simone* was in part created by us not from a top-down approach but with an experimental approach as musicians and users ourselves. The design process was "organic", borrowing elements from musical and technical influences we mentioned, testing the instrument at each new version and sometimes renouncing to have a total control on all aspects of the produced sound.

[114]: Magnusson (2009), 'Of Epistemic Tools'

There has been much discussion in the design and NIME community digital music systems having their own agency [134, 135], often drawing on Actor Network Theory [7] and Cybernetics [136], and leading for example to the development of actual instruments such as Davis' *Feral Cello* that seeks to "actively challenge notions of instrumental mastery and 'absolute control"' [137]. In fact, our design process might evoke the practice of "hacking" that is commonly found in the world of improvised music. In an ethnographic study on improvisers building-up their own instrumental devices [138], Canonne describes a practice that embraces devices that are inherently unpredictable and prone to accidents out of the control of its user. The fact that these instruments possess their own agency as he says, "allows the improviser to discover new uses" during the time of the performance.

[134]: Bown et al. (2009), 'Understanding Interaction in Contemporary Digital Music'

[135]: Borgo et al. (2010), 'Configurin(g) KaiBorg: Interactivity, Ideology, and Agency in Electro-Acoustic Improvised Music'

[7]: Latour (1996), 'On Actor-Network Theory. A Few Clarifications, Plus More Than a Few Complications'

[136]: Wiener (1950), *The Human Use of Human Beings*

[137]: Davis (2017), 'The Feral Cello'

[138]: Canonne (2019), 'Élaborer Son Dispositif d'improvisation'

Hence, while some unpredictable aspects of mosaicing synthesis in *Simone* has startled participants and as Magnusson argues, "where the digital instruments exhibit any chaotic or entropical

behaviour, it tends to be due to a failure in design, a bug in the code or loose wiring in the hardware" [114], we think of it as an integral part of the instrument and as well-suited to the task of improvisation *Simone* was intended for. Playing with *Simone* collectively is therefore intended not as a task of building a mutual construction with a tool that bends to the will of its users, but more as trying to collectively tame a system with its own proper agency and to accept its serendipitous nature. The collective nature of *Simone* therefore does not reside only in the time of performance but also in between sessions when users may exchange tips and devise strategies to work with the system.

[114]: Magnusson (2009), 'Of Epistemic Tools'

5.6.3 Instrumentality and Collective Improvisation

The analysis of participants' self-annotations of their performance shows that *Simone* successfully managed to provide them with the impression of playing collectively and that this feeling was shared between participants of the same group, although collective interaction appeared to have happened mostly on a one-by-one basis rather than with the whole group. These quantitative results must be completed by qualitative accounts of how the improvisations were experienced and perceived by the musicians.

The book *The Practice of Musical Improvisation* (*TPoMI* in what follows) by Bertrand Denzler and Jean-Luc Guionnet [139] provides a valuable account of the experience of collective improvisation from the point of view of improvisers. The way some of the comments found in these interviews echo comments from our participants and some of our observations allows us to reflect on some aspects of the design of *Simone* as an instrument for improvisation.

[139]: Denzler et al. (2020), *The Practice of Musical Improvisation*

In the absence of a predetermined shared plan or referent to follow, one can wonder which elements drive musicians' decision-making process and the temporal evolution of collective interaction in free improvisation. During our workshops, our participants described that they tried to complement the sound of other players but that these moments of coordination were sometimes disturbed by a player's decision to go in another direction, a vision reminiscent of Borgo's description of improvisation as a negotiation of freedoms and as "as a forum in which to explore various cooperative and conflicting interactive strategies" [140]. This dialectical

[140]: Borgo (2002), 'Negotiating Freedom'

process of construction and destruction as the creative core of free improvisation [141, 142] put our participants in a state of permanent awareness to adapt to other player's actions, as described in *TPoMI*: "When you're playing, you're fine-tuning all the time, you're constantly adapting to different factors which more or less generate the content." and "Sometimes, all of a sudden, there is a moment where it's going over the edge and falls into something totally different, and you need to be ready to go with that"

[141]: Borgo (2005), *Sync or Swarm*
[142]: Canonne et al. (2012), '"Cognition and Segmentation In Collective Free Improvisation'

One of our participants' commented that the distributed control of parameters in the *Solar System* scenario strengthened a sense of "inter-action" even suggesting that "it goes a bit beyond music". This is in line with the recognition that such extra-musical relationship is often seeked out by free improvisers in their musical practice:

> It's not only the sound. There's also the presence, the state of mind of the person, things that essentially you discover before playing, in a discussion, in a social relationship with the person. So it's a whole. In the end what I'm looking for isn't music, but rather a particular space for exchange. (*TPoMI*)

Perhaps more than a matter of interaction, one improviser in *TPoMI* mentions that "when I play with other musicians, lots of barriers fall. It's no longer me and them, but rather a kind of unity that forms, that's part of the listening". In *Simone* this feeling of unity is perhaps reinforced by the fact that participants use a similar interface, synthesizer and share the same soundbank, giving more cohesion to the overall sound produced and a better understanding of others' actions: "because we all have the same controls, the same parameters, I know what you have to do. It's not like you're playing the trumpet or something and I don't know exactly what you're doing. So I know he can do this, he can do that and it's the same things I can do. [...] It's easier to understand what they are doing" (P9). However, several comments from our participants suggest that this feeling of unity could have been improved by providing more visual feedback on the actions of other players and network communication between players, by improving the technical setup to give the impression of a shared sonic space instead of localized loudspeakers and by providing a way to use *Simone* with a tangible interface.

5.7 Chapter Summary

In this project, we employed a research through design approach to study the design and appropriation of *Simone*, a distributed musical instrument for collective improvisation. *Simone* was designed through a second-person perspective: while it embeds many personal aesthetic preferences, it was also conceived as an experimental setup to explore instrument learning processes and collective improvised interaction.

Our contributions are twofold: First, we presented the design and implementation of *Simone* whose originality lies in its implementation of different interaction scenarios that define the way users are interconnected through the network. Second, we studied the process of appropriation of the system by expert users. To that end, we organized workshops where groups of participants were asked to improvise collectively with *Simone* and where we collected qualitative data through semi-structured group interviews and quantitative data in the form of self-annotations of musical performances. These data allowed us to reflect both on the design of *Simone* as well as on the forms of learning paths and of collective improvised interaction that occurred during the workshops.

Our results suggest that the process of appropriation is guided by a complex perception of the constraints of the instrument and is strongly dependent on participants' musical background. It also shows that after this step of familiarization, participants were able to interact collectively to build a coherent musical discourse and that these interactions were influenced by the networked connections between participants and the different interaction scenarios within *Simone*.

For this initial design step of *Simone*, most decisions regarding the instrument design were made prior to user appropriation. In the next chapter, considering with Canonne that an interesting aspect of free improvised music practices lies in the process of coevolution between the improvisers' practice and their instruments in the form of instrument augmentation and *bricolage* [143], we will focus on observing how design decisions of such a distributed system can be left to its users and how the system can evolve over a longer period of time and use.

[143]: Canonne (2021), *La Lutherie Des Improvisateurs*

APPENDIX: Interface for each scenario

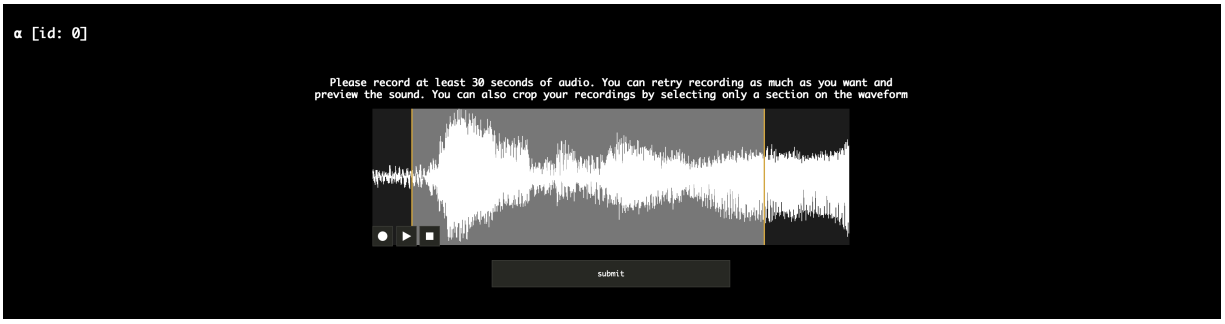


Figure 5.16: Before the *Clone* scenario users are brought to this interface to record a sound that will be sent to another user to become their *generator* sound.

Figure 5.17: Interface of *Simone* in the *Clone* scenario. Compared to the *Drum Machine* scenario, user's cannot choose the *generator* sound from the soundbank (it is assigned to them at the start) and loops are no longer restricted to fixed length.

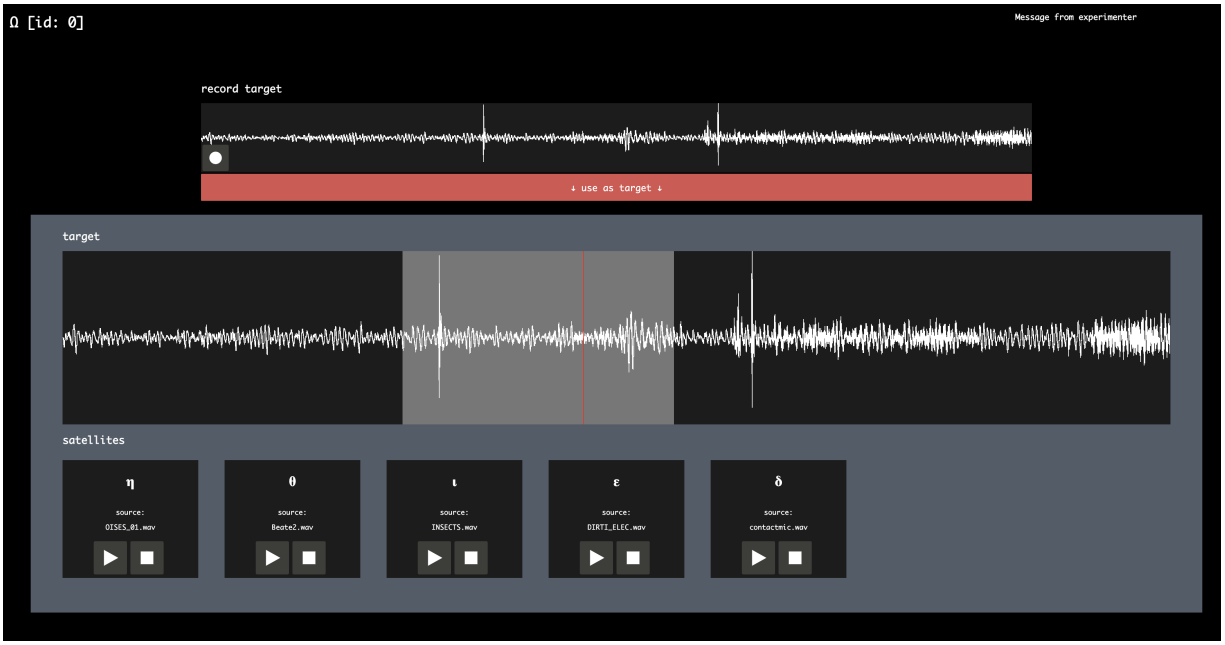


Figure 5.18: Interface of *Simone* in the *Solar System* scenario for the *sun* player. In this role, controls are restricted to recording a *model* sound and starting/stopping synthesis of the *satellites*.

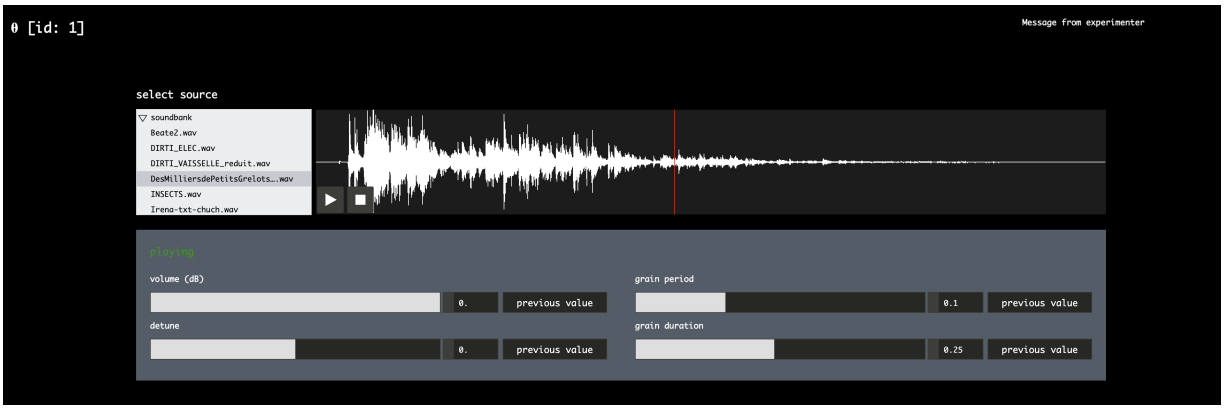


Figure 5.19: Interface of *Simone* in the *Solar System* scenario for a *satellite* player. In this role, controls are restricted to choosing a *generator* sound among a soundbank and changing synthesis parameters.

6 *Simone Solo*: Designing a Distributed Instrument Through Long-Term Research-Creation

6.1 Introduction

When designing the collective version of *Simone* (cf. Chapter 5) and reflecting on the different interaction scenarios I imagined, it quickly appeared that a (semi-)autonomous version of the system could be created. This version called "The Aviary" involved a number N of devices equipped with microphones scattered across a room. All these devices would use their microphone stream as the real time input of a mosaicing engine. The devices would then react to sounds played in the room or even to the sound of the other devices themselves to create a sort of autonomous dialogue. A controller interface would be available to a user to monitor devices, change their *generator* sound and control volumes and synthesis parameters. I saw this idea as fruitful in order to pursue development of a version of *Simone* that could run on Raspberry Pi computers.

Due to the difficulty to create a mosaicing engine that reacts in real time, this idea was postponed and we redirected our effort to another configuration of the system made by tweaking the *Solar System* scenario. In this configuration, the few controls left to the *satellite* clients (i.e. choice of source sound and control of synthesis parameters) could be transferred to the *sun* client, thus transforming *satellite* clients into simple reactive clients playing sound after receiving analysis data. The result then becomes a solo instrument in which a single user controls an ensemble of

distributed devices.

Parallel to the development of this version of *Simone*, we met clarinetist, composer and improviser Jean-Brice Godet[1] with the objective to pursue a long term collaboration on the design and appropriation of distributed musical systems.

1: https://www.jeanbricegodet.com/

For the past years, Jean-Brice had been developing an improvised musical practice with multiple dictaphones and tape players playing at the same time. This led him to participate in the creation of a street performance named "Espace de Gratuité" by stage director Ema Drouin[2] in which mobile phones are placed in closed objects and play sound files selected by Jean-Brice. Jean-Brice himself had designed a small controller interface that allowed him to start and stop the playback of those files.

2: https://deuxiemegroupe.org/repertoire/espace-s-de-gratuite

During our first meetings with Jean-Brice, we proposed some ideas of collaborations that could be developed to explore the co-design of distributed music systems informed by his musical practice. Since I was developing the first prototype of *Simone Solo* at the same time, we agreed to give Jean-Brice a first version of the system that he could learn and play with and to document its learning process. We regularly met with him to observe how he would use the system in studio sessions and through video recordings he would send us we discussed how the design of the system could be improved.

Therefore, this chapter presents work in the second-person perspective through our engagement in a co-design methodology informed by a research-creation approach over a period of time of more than 1 year and a half. A chronological timeline of the major events of this collaboration and the various evolution in the development of *Simone Solo* is shown on Figure 6.1. This chapter adopts a chronological structure recounting the different steps of our collaboration with Jean-Brice and the progression of the project. These different steps will be put in regard with an interview made with Jean-Brice more than 1 year and a half after starting this collaboration[3].

3: This interview was conducted at Ircam and was made in French and translated for the need of this manuscript

This chapter contains multiple accompanying videos. These are visible at the following address: https://alienorgolvet.com/thesis/chapter-simone-solo.html

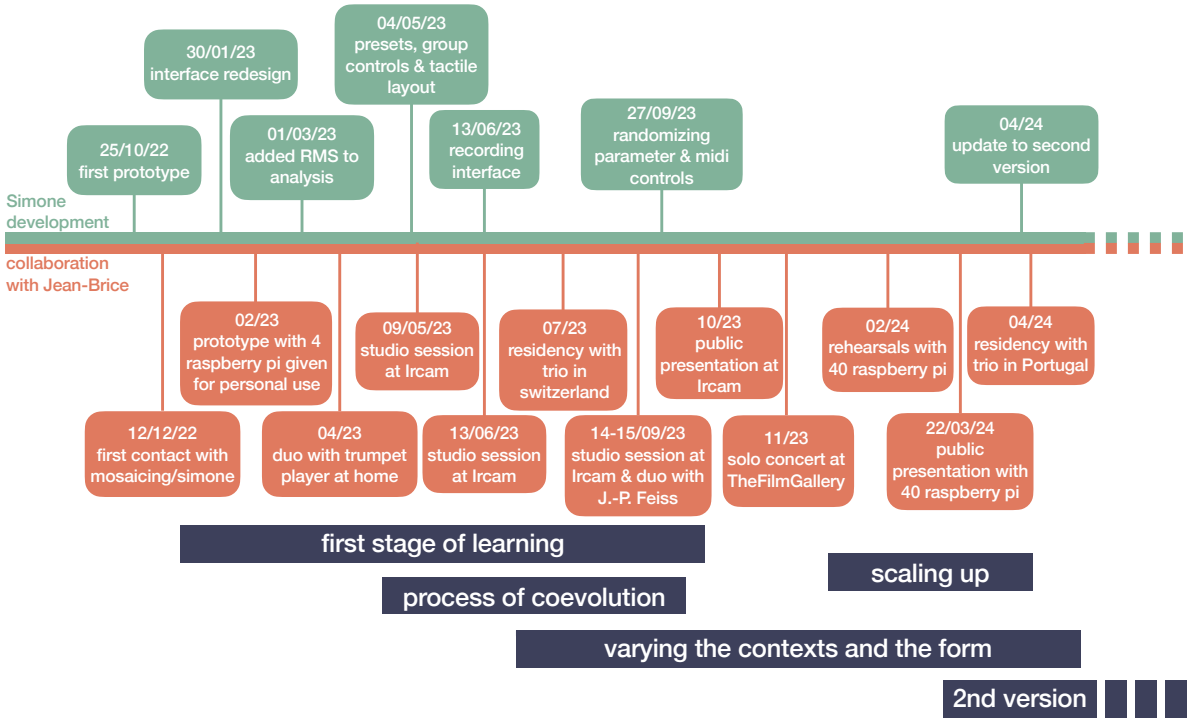


Figure 6.1: Timeline showing the different steps in the development of *Simone Solo* and in the collaboration with Jean-Brice. The bottom section shows how this chapter's sections and subsections relate to this timeline.

6.2 Design Overview and First Implementation

Simone Solo is a distributed instrument controlled through a web interface meant to be played by a single instrumentalist. The sound is produced by a network of devices (each called *satellite*) equipped with loudspeakers using a synthesis engine that uses audio mosaicing. Any device that can run *Node.js* or a web browser can be used as a *satellite* but in the scope of this work we only use Raspberry Pi nano computers. The instrument is operated through a web interface that offers the option to: 1) record a sound with the microphone and use it as the *model* sound for audio mosaicing on all *satellites* simultaneously 2) control various synthesis parameters on each *satellite* individually (cf. Figure 6.2).

In this section we present a first implementation of the system and the first steps undertaken by Jean-Brice to learn how to use the instrument.

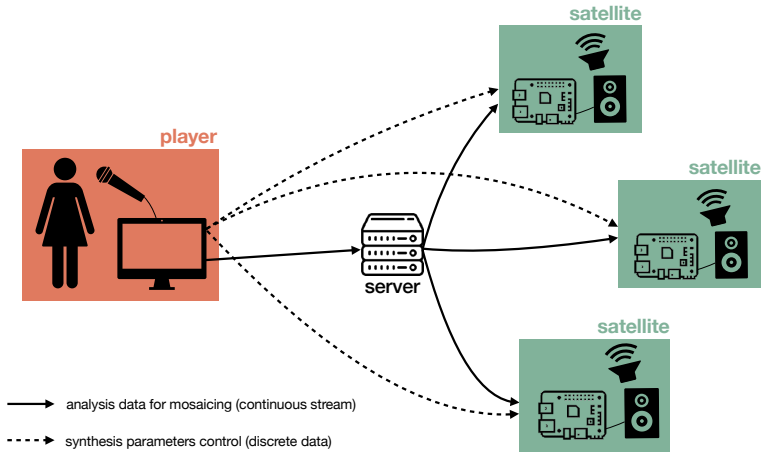


Figure 6.2: Diagram of the *Simone Solo* application.

6.2.1 The Controller Interface (First Version)

The *Simone Solo* interface is accessed through a web browser (cf Figure 6.3). The interface is divided into multiple sections.

The Record Section

At the top of the page, the "record target" section is used to record a sound with the microphone input selected upon entering the web page. After recording is complete, the waveform of the recorded sound will be displayed. If the user is satisfied with the recorded sound, they can press the "↓ use as target ↓" button to use this sound as the running *model* sound in the audio mosaicing process.

Upon pressing this button, the new *model* sound goes through a process of analysis. For each grain of the sound, the MFCC vector and RMS value are computed. This analysis process is performed in another thread using a Web Worker to avoid blocking the interface.

The *model* Sound Section

In this section, the waveform of the currently running *model* sound is displayed. By dragging the mouse over the waveform, the user can select a smaller segment of the waveform. The analysis engine will then loop over this segment of the *model* sound. The selected segment can be moved over the waveform by clicking on it and

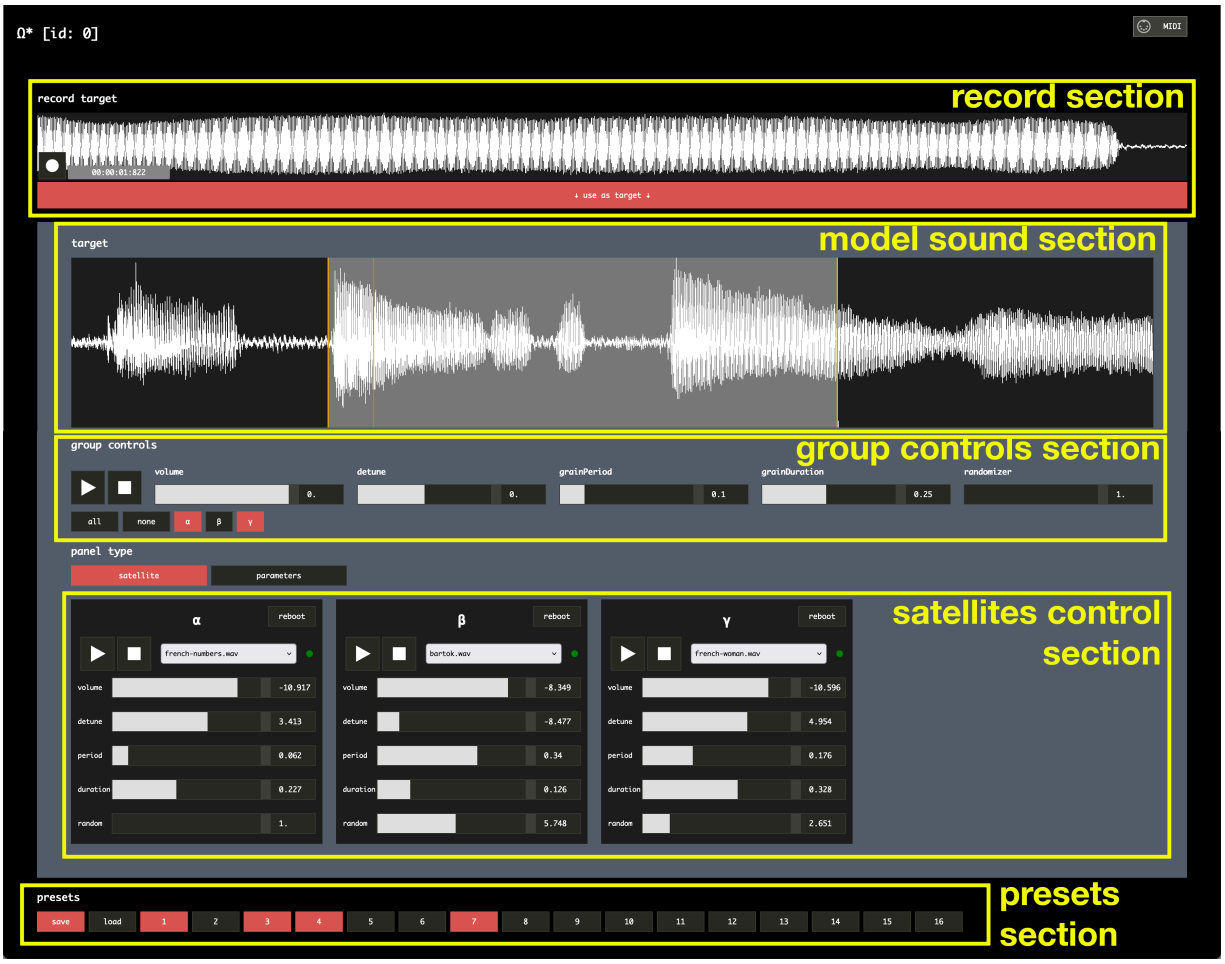


Figure 6.3: The full *Simone Solo* interface in a web browser and its different sections.

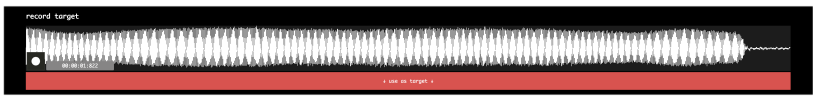


Figure 6.4: The record section in the *Simone Solo* interface.

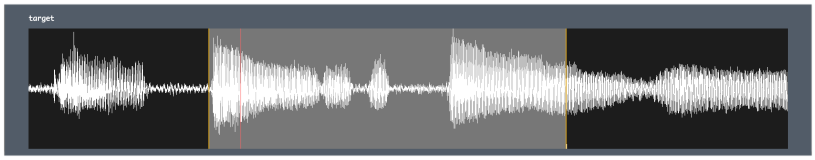


Figure 6.5: The *model* sound section in the *Simone Solo* interface.

dragging the mouse. Small yellow handles on each end of the segment can be used to lengthen or shorten it.

An "analysis engine" constantly runs in the background. With a fixed period, a new grain of the *model* sound is selected, and the grain MFCC and RMS values are fetched and normalized before being simultaneously sent to all the *satellites* though the server. The analysis engine linearly progresses through the selected segment of the *model* sound and loops as it reaches the end of the segment.

The Satellites Controls Section

Figure 6.6: The satellite controls section in the *Simone Solo* interface.

For each *satellite*, a control panel is displayed. This control panel allows to control the synthesis parameters specific to each satellite, namely: whether to start or stop synthesis, the *generator* sound to use for mosaicing (a small dot next to the menu turns green when the file is loaded on the *satellite*), sliders to control volume, detuning, period and duration of grains and a randomization in the selection of grains (instead of selecting the grain most similar to the target grain, we select it among the n most similar grains).

6.2.2 The Satellite Client

The *satellite* client is accessible via a web browser or through *Node.js*. Upon connection to the server, a *satellite* is given a name (a greek letter).

Upon receiving a command from the controller interface to load a new *generator* sound, this file is downloaded from the server. It is then immediately analyzed by computing the MFCC and RMS values of each grain of the sound. This analysis task is performed in another thread using a Web Worker[4] to avoid blocking the synthesis process. Finally, the MFCC vectors are arranged in a k-d tree structure[5] to allow for fast nearest neighbor search (see below).

The synthesis engine of the *satellite* clients was implemented using the *Web Audio API* (for web clients) and the *Node Web Audio API* (for *Node.js* clients). The *satellite* client periodically receives an update on the current *model* grain being sent by the controller interface managed by the instrumentalist. This update contains the MFCC vector of the *model* grain and its RMS value. We also compute the mean and standard deviation of the MFCC vectors for normalization purposes (necessary for comparison with the

4: using the `web-worker` library in *Node.js* `https://www.npmjs.com/package/web-worker`

5: using the `static-kdtree` library in *Node.js* `https://www.npmjs.com/package/static-kdtree`

model data).

At each time period (controlled by the "grain period" parameter), the *n* (with *n* being the value of the "random" parameter) most similar grains (in terms of the euclidean distance of their MFCC vectors) from the *generator* sound are fetched from the k-d tree containing all of the *generator* grains MFCC vectors using a k-NN (k-Nearest Neighbors) algorithm. A single grain is randomly selected from the *n* nearest grains. This grain is then multiplied by a triangle envelope for smoother granulation and then passes through a gain whose value is set by the RMS value of the *model* grain. Finally, this grain is played for a duration set by the "grain duration" parameter and with a detuning set by the "detune" parameter.

A video presenting the basic functioning of *Simone Solo* and the main features is visible on the companion website `https://alienorgolvet.com/thesis/chapter-simone-solo.html`

6.3 Process of Appropriation and Evolutions

We gave Jean-Brice a first version of *Simone Solo* as well as four[6] Raspberry Pi computers with portable loudspeakers and the equipment to create a local network. We gave him the instruction to use the instrument regularly and to document the different learning steps and use cases.

6: we gave him four more a few months later

6.3.1 A First Stage of Learning

Interestingly, one of the first steps in the process of appropriating the instrument for Jean-Brice was to constitute a personal soundbank of *generator* sounds. This was made in various steps with various types of sound[7].

7: A video showing one of the first steps of this process is available on the companion website

A first step was to start with what Jean-Brice describes as "simple musical elements" and could be qualified as "test sounds" to understand how the system would react to them. These sounds include: sinus, clicks, white noise, piano chords, clarinet sounds... These sounds were initially envisaged as building blocks for understanding how the instrument reacts and functions, as what Jean-Brice describes as a "base grammar", thus revealing a per-

ception of the system and of the synthesis engine as a sort of black box that provides a deterministic output to a given input. Therefore, Jean-Brice's process of familiarization takes an additive, combinatorial and inferential approach as he describes it as "starting from simple elements [...] and to see how I could combine them, how I could transform them to try to make sound pieces with them". Over time, these "simple musical elements" became an integral part of Jean-Brice's musical material with Simone, for example with frequent use of masses of sine waves slightly shifted in frequency.

A second part of this sound bank was built up from personal recordings of instrumental sounds (either solo or group). Jean-Brice's familiarity with the timbral content of these recordings certainly facilitated the appropriation process. The use of this kind of sound also allows us to see Jean-Brice's use of *Simone Solo* in the continuity of his work as an instrumentalist, as a kind of remix and deconstruction of his own practice.

Finally, in a later part, the sound bank was enriched with sounds gleaned from everyday life, thanks to a web page dedicated to the recording of sounds that we have developed (see below), in an aesthetic of "détournement" close to *musique concrète*. Jean-Brice was able to record sounds from a live TV broadcast of the French Open, and use them almost immediately in *Simone Solo*[8] (see Figure 6.7).

8: A video is visible on the companion website

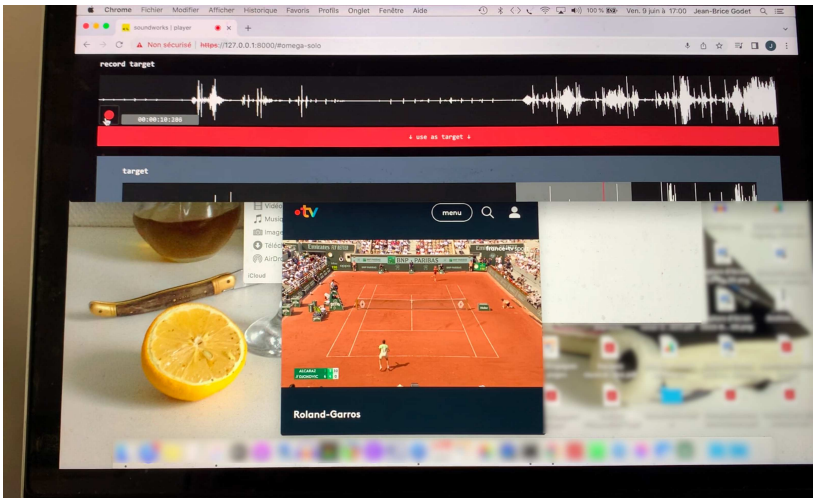


Figure 6.7: Simone's recording interface (top) opened at the same time as a live TV broadcast of the French Open on Jean-Brice's computer.

Jean-Brice describes his process of learning the instrument as a process of trial and error ("I proceeded by trial and error: success, failure, trial, success, failure, until I found musical places I liked.") and by playing and producing sound material as much as possible

("to produce, to produce sound material, to improve my ability to interact with the software, with the interface"). He followed this process for approximately 10 months by using the system from time to time in an irregular manner until reaching a stage where he felt "comfortable enough with the system, with the material [he could produce]" to play in front of an audience.

Jean-Brice's describes his method of evaluating his mastery of *Simone Solo* as similar to any other musical instrument:

> I think it means both understanding exactly what it does when I change the parameters in real time and if I imagine sound material, being able to imagine, if I touch this parameter, it's going to do this. Like a musical instrument. This is really a musical instrument thing. [...] Projecting yourself into a desire for a precise sound matter with respect to where I am currently, and thinking "if I move this parameter, I'll get there". [...] And if that's what's happening, that means I'm in control of the interface."

6.3.2 A Process of Co-Evolution

Jean-Brice's process of learning *Simone Solo* never happened with a fixed state of the system and the interface. Over the videos and comments he would send us and the studio session we had with him, we engaged in a constant process of coevolution of the system with him. Jean-Brice describes this process in the following way: "Whenever I'd reach a limitation, I'd call you, we'd find a way of pushing that limitation, one way or another, and then other limitations would appear". Each new feature would provide more options for Jean-Brice and would have an influence on the development of playing techniques. We detail a few of these modifications.

The amount of manipulation required to control a few *satellite* quickly demonstrated the importance of a means of controlling sound sources by group. Instead of an interface for creating different groups, which would have been too difficult to implement and cumbersome to handle in the first state of the interface (cf. Section 6.5), we preferred to implement sliders and buttons that control all the Raspberry Pis at the same time, while leaving the

possibility of leaving some of them unchanged (cf. Figure 6.8). For the same reason, we've also implemented a system of presets for saving parameter combinations. These presets are saved on the user's computer using JavaScript's *localStorage* function, allowing them to be retained between sessions (cf Figure 6.9).

Figure 6.8: The group controls section in the first version of *Simone Solo*. This section controls multiple *satellites* at the same time. At the bottom, a button for each *satellite* is displayed. When selected (button in red), the corresponding *satellite* will react to group controls.

Figure 6.9: The presets section in the first version of *Simone Solo*. 16 preset slots are available. Red buttons indicate that a preset is currently saved to this slot.

As it inherited from the design of *Simone*, the *Simone Solo* interface was initially developed to be played with a mouse or trackpad. To Jean-Brice, this quickly proved to be a major obstacle to the development of an instrumental practice and a prerequisite to consider *Simone Solo* as a musical instrument:

> To say that it's an instrument, we're going to have to develop the interface, with the touch screen, you know, and the MIDI interface. In any case, there comes a point where the trackpad really shows its limits. [...] in any case, yes, I think there's a possibility there for a real musical instrument with a dedicated control interface.

A major feature requested by Jean-Brice was the addition of MIDI support. We specifically developed a MIDI component for the textttsc-components package for this purpose (cf. Chapter 2) and added it to the *Simone Solo* controller interface. This component can be used to set up MIDI bindings to any slider and buttons in the interface (cf Figure 6.10). Support of MIDI devices to control *Simone Solo* was seen by Jean-Brice as a definite "step" that had a significant change in his way to play the instrument.

To provide more control options, Jean-Brice also requested another layout for satellite control parameters specifically tailored to tactile devices. While usable with tactile devices, the default interface with panels dedicated to each *satellite* (cf. Figure 6.6) made it difficult to move multiple sliders at the same time due to the

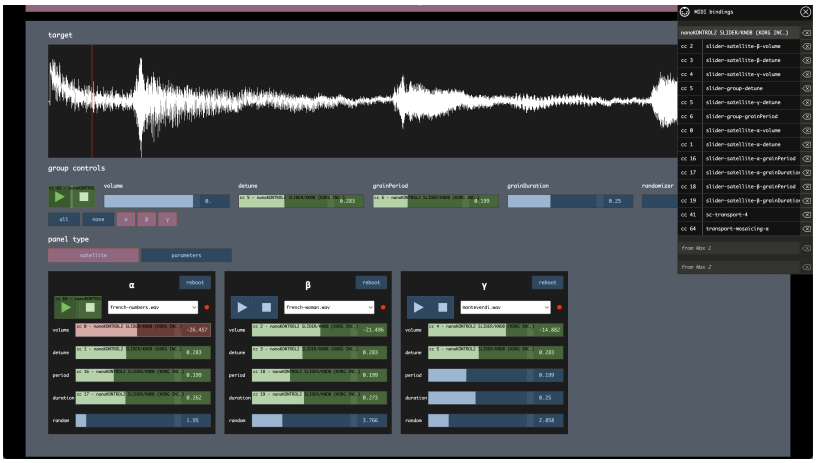


Figure 6.10: Setting up MIDI bindings in *Simone Solo*

disposition and orientation of sliders. We implemented another layout with vertical sliders for each synthesis parameter (cf. Figure 6.11).

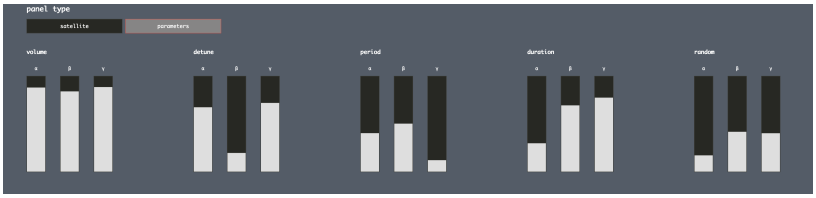


Figure 6.11: Another layout for *satellites*' synthesis parameters control for tactile devices.

The introduction of these new controllability options allowed Jean-Brice to manipulate multiple parameters at the same time, something which was not possible with a mouse or trackpad, which was seen as a radical change in his way to play the instrument, multiplying the number of possible effects. The ability to use MIDI controllers also improved the visual and gestural aspects of performances. Jean-Brice mentioned that an audience member once expressed their frustration of "not understanding what [he is] doing as [he is] behind the screen making things and [they] cannot see". For audience members, it may be important, as Jean-Brice postulates, to "have a visual perception of the relationship of what you're doing and what is produced".

As we observed that the constitution of a soundbank was a major part of Jean-Brice's process of appropriation, we developed a recording interface directly accessible in the web browser. This interface can be used to record a sound with the microphone input and to upload it to the server directly into the soundbank and immediately used on the *satellites* as a *generator* sound (cf. Figure 6.12 for a screenshot and Figure 6.7 for a picture of Jean-Brice using it in real conditions).

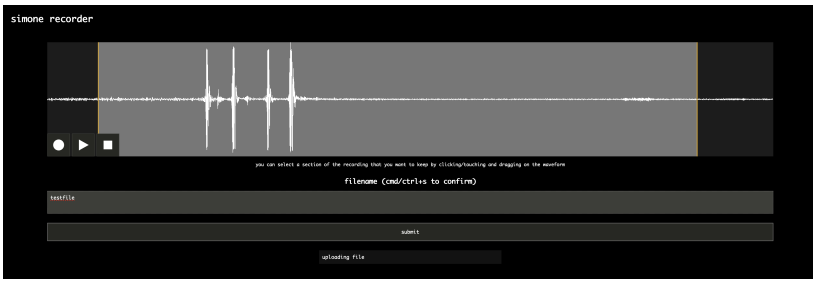


Figure 6.12: The recording interface for *Simone Solo* in the web browser. A sound can be recorded with the microphone input, cropped by selecting a segment with the mouse and uploaded to the server with the chosen filename.

On the other hand, a number of modifications suggested by Jean-Brice were not implemented, as we felt they were not consistent with the original design and intent of the instrument. For example, Jean-Brice suggested the ability to have a different loop segment on the *model* sound for each Raspberry Pi. This went against my desire to have a system whose sound sources are controlled by the same audio input, thus guaranteeing a certain acoustic coherence.

This negotiation between user/musician and designer demonstrates a specific complexity in the design of a digital musical instrument. During a playing session with another musician, the latter observed this difference with more traditional musical instruments:

> That's the limit of this kind of open-ended stuff, because if you take an instrument... any traditional instrument like cello or piano has its limits, and that's why when you play it, you know the limits, and when you write for it, you know the limits even if you think you're going to push them. The danger with something that can be modified ad infinitum is that in the end... why should you [stick to a definite state]?

A feeling of arbitrariness may therefore arise for the user in some of the design choices presented to him, as everything could be modified with a few lines of code. Leaving too many parameters or modifiable elements to the user can be a hindrance to the development of instrumental practice with the system. Over time, It therefore became important for us to set limits and declare which elements of the system were fixed and not open to re-discussion. Thus, we generally confined ourselves to quality-of-life type modification that did not alter the general aesthetics and intention of the instrument.

Reflecting on his role during the development of *Simone Solo*,

Jean-Brice describes alternating between three different roles depending on context: "At the same time beta tester, guinea pig and almost composer, with composer needs and therefore inclined to ask for certain modifications from you."

6.3.3 Building on Previous Experience to Develop a Specific Practice

From the start, the process of appropriation of *Simone Solo* by Jean-Brice has been guided by his own musical practice as an improviser with the clarinet and with dictaphones and tape recorders. The playing techniques he developed, the vocabulary he used and the learning path he took were all to some extent inspired by his previous practice.

For instance, he followed the same method of combining "simple elements" that we described above as a first learning step when he started playing with dictaphones and tape players. Jean-Brice draws a bridge between these two practices by speaking of his vision of playing with both systems as a "succession of sound matters" with superpositions and transitions between "vertical" matters (breaks or rhythms) and "horizontal" matters (non-rhythmic layers of sound).

Likewise, Jean-Brice describes his tendency to exploit the very limits of an instrument in the process of learning:

> I have a slight tendency... and it was the same with tapes, to push the faders all the way, you know, one way or the other, and see what happens. I have a tendency to do that on the clarinet, to see if I can go beyond these limits, physical limits or software limits, how far we can go into these things, what happens if we do it for a long time. I tend to like going to these places to see, to define my playground, the framework.

This familiarity with his previous practice largely defined Jean-Brice's process of appropriation of *Simone Solo* and his own practice of the instrument to a point where he would testify being able to express himself as a musician with *Simone Solo* and he would recognize the sound material he would usually produce. Towards the end, Jean-Brice started imagining integrating *Simone Solo* as

another tool in the rest of his practice, for instance by setting it up as a sound installation playing before a concert, seeing it as a direct "extension" of his work with tapes.

Over time, Jean-Brice also developed specific playing techniques for *Simone Solo*. One of the main ones consists in recording the sound produced by the system and to use it as the *model* sound. Doing it multiple times in a row creates a sort of inner feedback loop of the instrument and, according to Jean-Brice, provides a feeling of "coherence" to the overall sound. Other techniques imply using the same *generator* sound on all *satellites* but with slightly different sets of parameters to create spatial variations or pushing the software to its limits with extreme values of parameters that cause the sound synthesis process to slow down or freeze the sound synthesis.

As in this last example, the process of appropriation and the discovery of novel playing techniques by Jean-Brice often passes by a practice of *détournement*, or, as what Jean-Brice describes better as a practice of "bypassing" the system's own nature and limitations. In addition to the tendency to explore the limits of the instrument's capabilities, Jean-Brice explains, for instance, that he found a way to bypass the specific textural aspect of mosaicing by using very long *model* sounds.

This process of discovery of techniques is often entangled with the implementation of new features. For instance, Jean-Brice often mentioned the difficulty to create sudden "breaks" in the sound due to the nature of mosaicing. He later described that "we've made a few changes that make creating breaks easier. Among other things, being able to change *generator* sound on all the Raspberry at the same time. So in the end, I think we've adapted. We've created possibilities for breaks that weren't there at the start"

In the end, Jean-Brice was able to develop a specific musical practice with *Simone Solo* by drawing on familiar elements brought from his already existing musical practice. Outside of these specific techniques he developed, Jean-Brice also describes how, by nature of its design, *Simone Solo* is a unique instrument which makes it interesting to play.

For instance, Jean-Brice particularly insists on the fact that the system and the interface only provides a partial, "fuzzy" control

on sound: "You control a sound material, but in a rather imprecise manner, which makes the system interesting". He adds that, due to the rather unpredictable nature of mosaicing, "you can reach ultra satisfying sound materials without really knowing which path you took. [...] You take another path and you reach a more or less similar texture but not exactly the same. In an improvised music context, it's great". This "fuzziness" is counterbalanced by the overall coherence and unity of sound provided by the fact that all *satellites* are controlled by the same *model* sound.

Moreover, Jean-Brice explains reaching a state where he would be able to produce "unprecedented", "noisy" sound textures that he would not personally be able to produce with any other tools, also due to the spatialized nature of the sound production. "The fact that the speakers can be moved around", he says, "makes you want to add movement to it. There's plenty to do. It's a good playground." In a reverse movement of influence, these new elements would themselves feed the rest of Jean-Brice's musical practice: "I recycle everything that comes my way. Simone, that's part of it. And this multidiffusion loudspeaker system fueled other thoughts, other research, other imaginations".

6.3.4 Varying the Contexts, Varying the Form

Following the first months of solo sessions at home and in the studio with us, Jean-Brice felt comfortable enough with the instrument to start varying the contexts of playing and to play with other musicians.

The first attempts of playing with other musicians happened at home with a trumpet player and in the studio at Ircam with cellist Jean-Philippe Feiss[9][10] (cf 6.13).

This studio session provided us with fruitful observations about the interaction between *Simone Solo* and another musician in the context of an improvised performance. Already accustomed to playing with Jean-Brice, the cellist recognized Jean-Brice's general aesthetic and musical language with which he was familiar in Jean-Brice's playing with *Simone Solo*. Noting that *Simone Solo*'s sound aesthetic matched that of electroacoustic music close to tape music or sound collage, he remarked that this made him follow two main strategies of interaction, either following the textural

9: https://www.jeanphilippefeiss.com/

10: Two videos from these sessions are visible on the companion website https://alienorgolvet.com/thesis/chapter-simone-solo.html

Figure 6.13: Jean-Brice (right) playing *Simone Solo* in the studio at Ircam with cellist Jean-Philippe Feiss.

side of the music, or opposing it by playing more melodically. He mentioned regretting that *Simone Solo* didn't offer more melodic possibilities to be confronted with melodic styles of playing with the cello.

As we did not explain how *Simone Solo* works in the first playing sessions, Jean-Philippe revealed a feeling of confusion as he imagined that it was a simple system of diffusion of fixed sounds or an autonomous human-machine improvisation system that reacted to his sound. Revealing the nature of *Simone Solo* changed his way of interacting with the system.

Jean-Brice's progress with the instrument went a step further by "confronting himself with the construction of a form" which he did through a public presentation at Ircam in October 2023[11], with a first solo concert at TheFilmGallery in November 2023 and through a trio with Frantz Loriot (on viola and turntable) and Ben Gerstein (on trombone and field recordings) for a residency in Swiss in July 2023 and for another residency in Portugal in April 2024[12].

11: video available on the companion website

12: video excerpts from these two residencies available on the companion website

Figure 6.14: Jean-Brice (right) playing *Simone Solo* with Ben Gerstein (field recordings, trombone) and Frantz Loriot (turntable, viola).

By playing with different musicians, Jean-Brice observed depending on the nature of the other instruments. Jean-Brice noted that playing *Simone Solo* was easier with electroacoustic instruments, with aesthetic closer to *Simone Solo*, with whom the objective is often the co-construction of a collective sound texture. With acoustic instruments, where improvisational playing often requires greater reactivity and the ability to create sudden breaks in sound, playing with *Simone Solo* was sometimes made more challenging by the difficulty of fully anticipating the result of the mosaicing process, and by the low reactivity of the instrument.

While it could be feared that the multiplicity of sound sources might drown out the auditory space and render other instruments inaudible, Jean-Brice was able to note that this was not the case, thanks to the sense of coherence between sources mentioned above and the specific color of mosaicing synthesis.

Playing with other musicians was useful to Jean-Brice as he mentioned that "it's interesting to have another pair of ears, or an interaction with another musician to force yourself to displace". Playing with others worked as a destabilization that helped Jean-Brice to maintain his practice of *Simone Solo* in evolution. He describes the practice of improvisation as filled with accidents and unexpected events born from the interaction that constantly provides new elements to "rebound" from. Playing with others also forced him to be more precise and to develop a set of gestures to be able to interact and react to the others' sound.

From his own experience, Jean-Brice also mentions that the practice of a musical instrument is enriched by exchanging with other practitioner of said instrument: "Instrumental practice, it's still very much made up of that, of other instrumentalists with whom you exchange, and with whom you realize, ah, well, you've found this, you play like this, you do like this, you manage to play that piece, etc. . . ". While Jean-Brice is for the moment the only player of *Simone Solo*, future works could be to ask other musicians to play the instrument and to observe how their practice would develop through their exchanges.

6.4 Scaling Up the System : Playing *Simone Solo* for 40 Devices

At the beginning of 2024, we had set up a large system of 40 Raspberry Pi each equipped with a pair of loudspeakers in the *Espace de Projection* at Ircam (cf Figure 6.15)[13]. As we were tasked to create short musical pieces for this system, we asked Jean-Brice to create one by using *Simone Solo*. As we anticipated, working with such a number of *satellites* revealed previously unseen challenges and shortcomings in the design of the system and forced Jean-Brice to change his approach of playing with *Simone Solo*.

13: this system is also described in Chapter 7

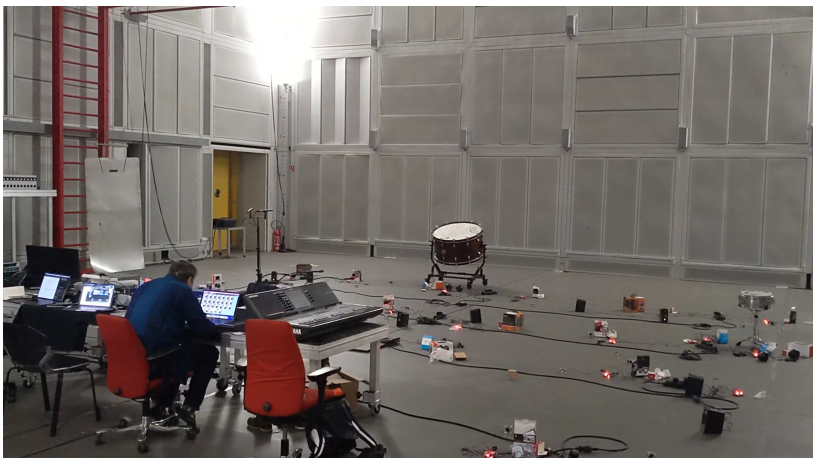


Figure 6.15: Jean-Brice playing *Simone Solo* with a network of 42 *satellites* in the *Espace de Projection* at Ircam.

From the start, it was obvious that the controller interface of *Simone Solo* was not tailored to control this many *satellites*. Indeed, a control panel was displayed for each single *satellite* thus making the screen filled with too many sliders and buttons to be handled by a single person (cf Figure 6.16). Jean-Brice had to circumvent the problem by creating groups of *satellites* by assigning multiple parameters to the same MIDI CC message. Even in this case, the process of creating such groups was a long and tedious process for Jean-Brice. Moreover, due to the number of displayed elements, using MIDI messages to control multiple elements would make the interface extremely slow and would create freezes in the production of sound and lagging in the change of sound parameters.

These difficulties are particularly visible in videos I filmed[14] in which Jean-Brice's reactions vary from laughter, anger and confusion due to the absurd reactions of the interface, and awe from hearing the sound produced by the system. Indeed, playing *Simone Solo* for such a large number of *satellites* produces an impression of sound that is strikingly different from playing with 4

14: video available on the companion website `https://alienorgolvet.com/thesis/chapter-simone-solo.html`

Figure 6.16: Jean-Brice playing *Simone Solo* with 42 *satellites*.

or 8 *satellites*. Whereas in the latter case, each sound source can be distinguished, in the former case, you can only hear an impression of being surrounded by a sound field without definite center or singular points.

This is not only due to the nature of the system but also to the fact that Jean-Brice was forced to adapt his composition process. For him, new questions arose and a completely different approach had to be taken: "How do you control 40 loudspeakers and what do you send into them and how do you play them? That's another question. There's a real threshold effect in terms of the number of speakers you can control, in real time." As we mentioned, groups of *satellites* had to be defined, thus limiting the possibilities of creating punctual sound sources. However, due to the limited time we had with the system, Jean-Brice describes creating these groups in a "completely random way".

Moreover, concerning the structure of the piece, Jean-Brice decided to reuse elements he had developed by playing with 8 *satellites* and "to condense them in a 4 minute piece"[15]. He focused on creating movements of large masses of noise made with sinuses around the whole system and randomly playing clarinet or vocal sounds.

15: a video of the performance of the piece called *Clarissime Iris* is available on the companion website

Overall, while the end result was definitely striking and stood out as a fleshed-out musical piece, Jean-Brice was hindered by the inadequacy of the controller interface and the fact that it did not scale up with the number of *satellites*. Jean-Brice testifies that "I had to find solutions and I found some. But they were not necessarily the best and not necessarily the most suitable for this piece."

6.5 A Second Version

After the experience in the *Espace de Projection*, it became obvious that a complete overhaul of *Simone Solo* was necessary to overcome some of its shortcomings and guarantee its sustainability. Hence, the application was completely redesigned to create a second version.

One of the main issues of the first version of *Simone Solo* was that it was conceived as a side version of the collective version of *Simone*, thus existing in the same code. This situation complicated maintenance as any changes could have an effect on the rest of *Simone*. Thus a requirement for a second version of *Simone Solo* was to develop a fully stand alone application.

To overcome the difficulty to operate the system with a large number of devices, the new version had to be designed to be agnostic to the number of *satellites*. To do this, we refined the idea of group controls that was sketched in the first version. In the new version, any number of groups can be defined and *satellites* can attach to these groups in any arrangement. Each group have their own set of synthesis parameters (the same as the *satellite* control panels in the first version: *generator* sound, volume, detune, etc. . .) and any change in these parameters are broadcasted to the *satellites* attached to this group.

Another problem of the first version was that we did not anticipate prolonged use over multiple sessions. Thus the state of the application would be reset at each new session and each *satellite* device would not keep the same ID which would make development of a consistent practice with the system cumbersome. In the new version, we implemented automatic saving of group arrangements and presets between sessions. This allows restarting from a fixed configuration of *satellites*. Moreover, *satellites* are not anymore assigned to a greek letter but are named by their OS hostname for better consistency.

A clear separation was made in the interface and in the architecture between the parts of the application that concerned the inputs (recording and processing of the *model* sound) and the outputs (synthesis engine). Benefiting from this, the new version offers multiple input modes. Apart from the already existing “loop record” mode in which the user is able to record a sound with the

microphone and use it as a *model* sound (with the option to define a loop segment on the waveform), the new version also offers a "loop load" mode in which a sound file from the soundbank can be loaded as a *model* file and a "realtime" mode in which the microphone input is analyzed in real time, each grain at a time.

Finally, we improved performances by simplifying the architecture of the application and by using Web Workers as often as possible for tasks with heavy workload.

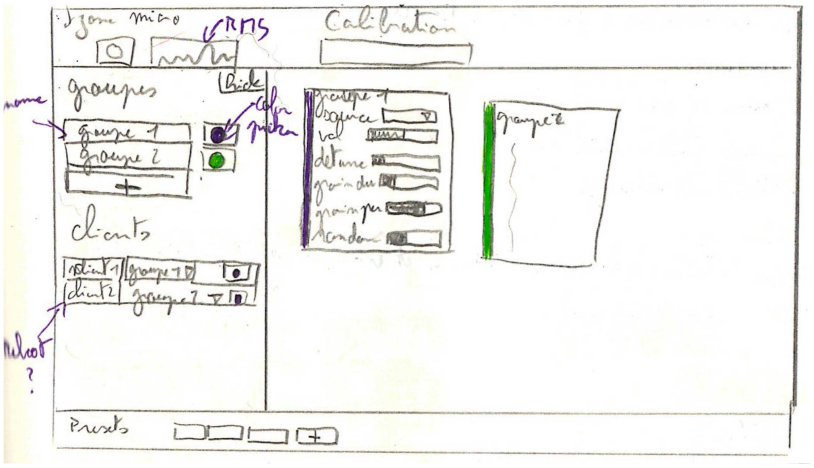


Figure 6.17: Sketching a new interface for *Simone Solo*.

6.5.1 The Interface

A fully new interface was designed for this new version of *Simone Solo* (cf. Figure 6.17 for an early sketch and Figure 6.18 for a screenshot of the full interface). Again, the interface is divided into multiple panels corresponding to different parts of the workflow. More efforts were made to optimize space and to provide customizable elements.

The Input Section

At the top of the interface, a panel is dedicated to interacting with the inputs of the systems (cf Figure 6.19). As mentioned above, three input modes are available. The user can switch between these modes thanks to the menu / tabs on the left.

The *loop record mode* is similar to the original version of *Simone Solo*. The user can record a sound using the input microphone. The waveform of the recorded sound is then displayed. A red button can be pressed to use this recorded sound as the current *model*

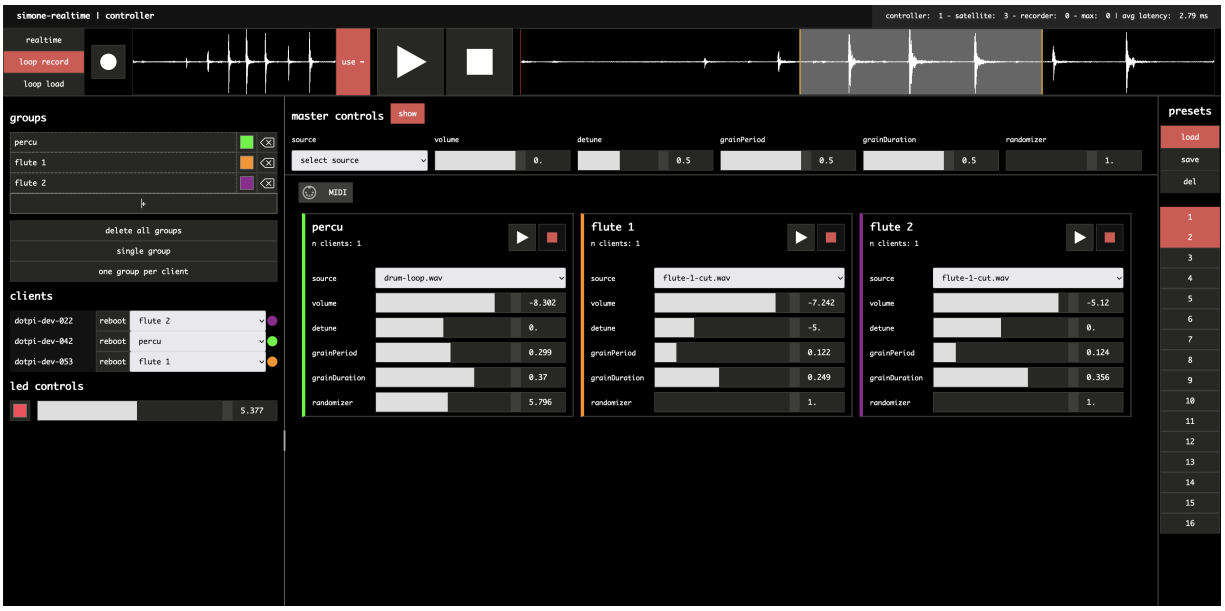


Figure 6.18: The interface of the second version of *Simone Solo* in a web browser.

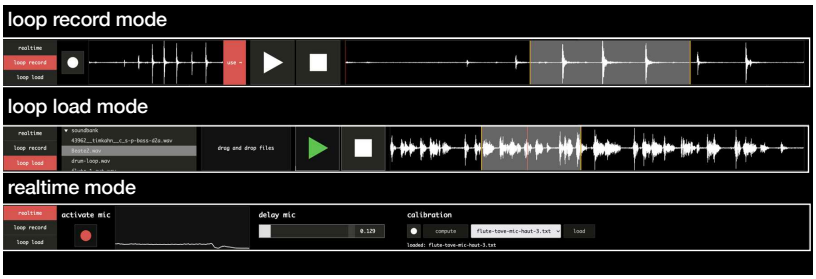


Figure 6.19: The control panels for the three input modes in the *Simone Solo* v2 interface.

sound. Upon pressing this button, an analysis process is performed in a Web Worker to compute the MFCC and RMS values of the new *model* sound. The current *model* sound is then updated at the end of this process. A loop segment can then be selected on the waveform of the current *model* sound. A large play/stop button can be used to start/stop the analysis engine and the sending of analysis data to the server.

The *loop load mode* functions very similarly but instead of recording a sound with the microphone, the user can load a sound file from the server to be used as a *model* sound. The option is also offered to directly drag and drop a sound file from the user's computer. Again, the waveform of the current *model* sound is displayed and a loop segment can be selected.

In the *realtime mode*, the input stream of the microphone is analyzed in real time and the result of the analysis is sent to the *satellites* in a continuous stream. The main obstacle to a real time mode in *Simone Solo* was that the step of normalization of MFCC

and RMS values is not feasible. Indeed, in the *loop record* and *loop load* modes, the mean and standard deviation of the MFCC and minimum and maximum values RMS over the whole signal can be computed as the sound file is loaded. In *realtime* mode however, these values cannot be anticipated. To solve this, we had to resort to a calibration process. The user can record a short excerpt of sound that is representative to the variety of sounds that will be played later. This short calibration sound is then analyzed to compute mean and standard deviation of MFCC and minimum and maximum values of RMS. These values are then stored in a file on the server and can be loaded in later sessions. On the interface, a record button is used to open/close the microphone input stream. Next to it, a signal displays the RMS value of the input signal in real time. Finally, a slider can be used to manipulate a delay between the input stream and the analysis process. This was introduced after observing that playing with no delay would disturb instrumentalists as the system was too reactive.

The Left Panel

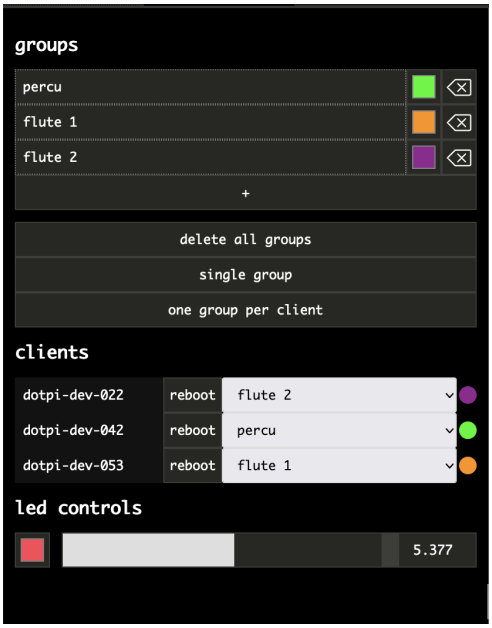


Figure 6.20: The left panel in the *Simone Solo* v2 interface.

The left panel of the interface contains multiple controls over the interface and the *satellites*. First, a section is used to define control groups. Groups can be created, deleted, renamed and a color can be selected for display purpose (see below).

Some shortcut buttons are available either to delete all groups, to create a single group for all *satellites*, or to create as many groups

as *satellites*.

Below, a list of *satellites* connected to the application is displayed. For each *satellite* (displayed by their OS hostname), a drop-down menu can be used to select which group this *satellite* is attached to. A small dot next to this menu is colored according to the selected group's color for quick visualization.

Finally, at the bottom of this panel, a small section is used to control the color and intensity multiplier of a LED connected to the Raspberry Pi computers. The LED's intensity depends on the output volume of the *satellite* and can provide a striking visual effect when playing with *Simone Solo*.

The Groups Controls Panel

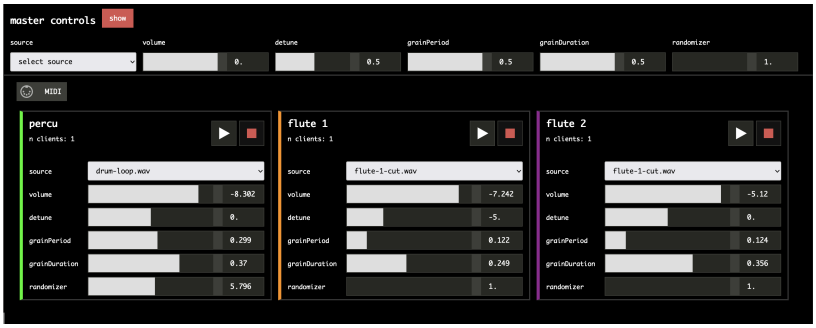


Figure 6.21: The groups controls panel in the *Simone Solo* v2 interface.

The main part of the interface is dedicated to the groups controls panel. For each group, a control panel is displayed. This panel contains the name of the group, a colored border on the left that corresponds to the selected color of the group and the same controls that were found in the *satellite* control panels in the first version of *Simone Solo*: a play/stop synthesis button, a menu for choosing the *generator* sound, sliders for controlling volume, detune, period and duration of grains and the randomizer parameter. Each change in these controls is immediately broadcasted to any *satellite* attached to this group. Whenever a new *satellite* is attached to a group, it adopts the current group's parameters values.

At the top is a "master controls" section. These controls affect all groups simultaneously. This section can be hidden if desired.

The Presets Panel

On the right of the interface, a small panel is dedicated to presets. Options are given to save and load presets of parameters configuration and to quickly switch between them. At the time of writing, presets only contain sets of parameters for the same groups configuration. Presets are saved on the server and can be retrieved between sessions.

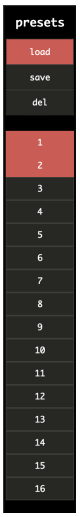


Figure 6.22: The presets panel in the *Simone Solo* v2 interface.

6.5.2 Adopting the New Version

In a rather interesting circular process, Jean-Brice discovered by himself that using the "realtime" input mode of the new version of the controller interface could be used to make the system function autonomously, similarly to the "Aviary" system which was the idea at the origin of the development of *Simone Solo*.

This effect can be produced by using the "realtime" input mode and by placing the microphone close to the loudspeakers of the *satellites*. The sound of the *satellites* will be picked up by the microphone and be instantly fed to the mosaicing system to control the *satellites*' output. This creates an interesting feedback of information throughout the system.

It is not surprise that Jean-Brice quickly discovered this new playing technique as it is the direct continuation of a technique he already developed in the first version of *Simone Solo* by recording the overall sound of the instrument over a short time to use it as a *model* sound. A video of this effect can be seen on the companion website `https://alienorgolvet.com/thesis/chapter-simone-solo.html`.

6.6 Discussion

6.6.1 Designing for Real Instrumental Practice

The long-term collaboration with Jean-Brice enabled us to highlight a number of observations by confronting our system to an instrumental practice influenced by a technical and cultural context, and

real-life conditions of use responding to production constraints. This confrontation also had a clear impact on the design of the device and its interface by highlighting a number of design elements essential to the development of an instrumental practice.

This includes the implementation of a number of features such as MIDI controller support, which not only makes the system easier to handle and increases the number of playing possibilities (e.g. the ability to act on multiple parameters simultaneously), but also enables Jean-Brice to organize the gestures he uses with *Simone Solo* within the spatial and material frame of a control interface reminiscent of already familiar gestures (unlike mouse manipulation, which is more idiosyncratic). The MIDI controller thus can be seen as a direct bridge between two musical practices.

We can also mention the implementation of presets, which not only vary live playing possibilities by enabling radical changes of parameter settings, but also allows Jean-Brice to save work steps and extend instrument learning over several sessions, thus anchoring instrument practice over a longer period of time.

It is worth mentioning how the interaction between the observation of Jean-Brice practice and the development of new features did not happen in a direct linear manner. Modifications in *Simone Solo* did not always occur because Jean-Brice asked for it. This process of evolution of the system rather occurred through the crossing of multiple timelines. For instance, the support of MIDI controllers was asked rather early by Jean-Brice but it was already planned to be developed as part of our web components ecosystem (cf Chapter 2) and took time to be implemented as we needed other basic components. While we could have implemented an ad-hoc solution for Jean-Brice in the meantime, we decided to postpone this to inscribe the development of this solution in a broader development strategy and to guarantee that the solution would be more generic, reusable and robust. This example shows how our practice-based methodology is concerned with the idea of reusing development and knowledge across multiple disciplines and applications.

Working with Jean-Brice also enabled us to take the instrument out of the laboratory and deploy it in a variety of environments. In particular, Jean-Brice took advantage of the device's portability to deploy it outdoors. In its current state, however, the device is not

fully adapted to this kind of use, and a next stage of the project could involve designing compact, waterproof cases in which all the elements of one of the device's sound sources (raspberry pi, batteries, speakers or audio output) would be integrated. This would also make the instrument more durable and stable through a form that would be easier to transport and pass on to other potential users.

Another perspective for the development of this collaboration lies in the integration of *Simone Solo* with the rest of the tools Jean-Brice uses in his musical practice. As things stand, the *Simone Solo* system remains relatively closed and is unable to communicate with other processes or software (such as other web interfaces or the *Max/MSP* environment, for instance). In particular, this prevents *Simone Solo* from being controlled by other processes, or from sharing a common temporality of musical events with other applications.

6.6.2 Digital Instruments and Appropriation

Looking back on this collaboration, I regret that Jean Brice only felt partially empowered to the role of "co-developer". Although we were in constant discussion about which elements to add or modify, Jean-Brice always relied on me, the developer, to implement these elements, since he didn't have the necessary skills to act on the system's computer program. Jean-Brice was thus deprived of that way of appropriating the instrument common to many improvisers, through material modification of the instrument (e.g. preparation of pianos) similar to a *hacking* approach as observed by Canonne[138].

[138]: Canonne (2019), 'Élaborer Son Dispositif d'improvisation'

However, it appeared complicated to implement a mechanism similar to that used in *Koryphaîos* (see Chapter 4) that would allow the user to access certain parts of the program. While this is well suited to user customization of certain individual aspects of an application, it seems complicated to implement a similar option when it comes to acting on the very structure of the instrument. The manner in which the elements of *Simone Solo* are interconnected made this impossible.

Jean-Brice did only partially appropriate the instrument. To go further and give him the ability to make it evolve himself (beyond

the relatively marginal act of choosing his sound bank and the controllers he uses) would then have needed him to learn the *JavaScript* language and the *soundworks* framework. Interestingly, in the latter part of this collaboration he expressed that he wanted to learn these skills, which shows how the role of a musician and user can shift in contact with Digital Music Instruments.

To give instrumentalists back more control and encourage an appropriation approach that is more faithful to their practice, it would seem more beneficial to design more "material" systems. Through the *Organium*[144] project, Magnusson *et al.* show how a "library" of material technical components that can be assembled together encourages an improvisatory approach to instrument design. They also show how the various projects developed through this infrastructure act as *boundary objects*[145] that "facilitate the exchange of ideas across disciplines, forming a shared platform for research that creates a common language while enabling the understanding of each other's disciplinary language".

[144]: Magnusson et al. (2024), 'The Organium'

[145]: Star et al. (1989), 'Institutional Ecology, 'Translations' and Boundary Objects'

6.7 Chapter Summary

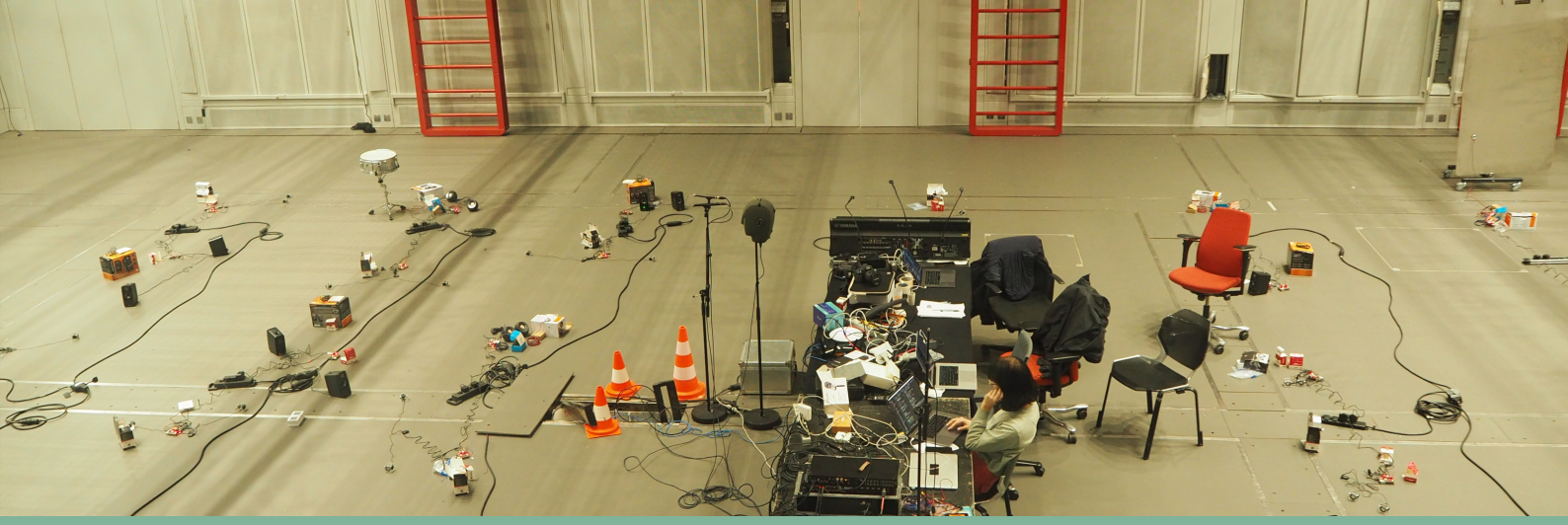
In this chapter I presented *Simone Solo*, an instrument consisting of distributed sound sources (called *satellites*) controlled by a single instrumentalist through a web interface. Using a second-person perspective, we pursued a long-term collaboration with the artist Jean-Brice Godet during which we gave him a prototype of the instrument and observed his process of appropriation.

We described the different steps of this process. A first step of learning lasted for several months during which Jean-Brice used a trial and error process guided by its own musical practice to make sense of the instrument. During this phase, I also proceeded to implement new features and modifications suggested by Jean-Brice and I described how this entailed a process of negotiation between the designer (me) and the user (Jean-Brice). Once Jean-Brice felt comfortable enough with *Simone Solo* he was able to use it in multiple contexts in concerts and with other musicians.

In the last step, Jean-Brice performed with *Simone Solo* with 40 *satellites*. This experience revealed some of the shortcomings in the design of the control interface and forced him to adapt his practice

of the instrument. To address these shortcomings and to better integrate the elements that were added to the instrument throughout the collaboration, I completely redesigned the interface.

Finally, I discussed how our long-term collaboration with Jean-Brice allowed us to confront the design of the instrument to real-life musical practice and revealed the specific constraints it entailed that could not be anticipated during the design phase. Moreover, reflecting on the form that the collaboration took offers a contrasting vision of the appropriation of the instrument by Jean-Brice.



7 Creating *Quasimodots*: A musical piece for 40 Raspberry Pi

7.1 Inspirations

The main inspiration for this piece is a composition by Alvin Lucier entitled *Quasimodo, The Great Lover* (1970). Inspired by long-distance communication of whales, Alvin Lucier described it as a piece "for any person who wished to send sounds over long distances through air, water, ice, metal, stone, or any other sound-carrying medium, using the sounds to capture and carry to listeners far away the acoustic characteristics of the environments through which they travel."

Written in prose, like most of his other scores of the time [146], the score starts by describing the technical equipment needed for the execution of the piece : "Use one or more microphone-amplifier-loudspeaker systems to lengthen the distance over which the sounds may be sent". For example a sound emitted at the top of the building could be transmitted to the ground floor by placing these microphone-amplifier-loudspeaker systems at the end of each room and in the stairwell. Of course, in-between each system, the sound is transformed by the medium it passes through, and the end result, heard at the end of the chain of systems, carries the influence of each of these mediums. In fact, the piece was specifically composed to highlight this effect as it is mentioned in the score that "extensions of modifications of the range, timbre, envelope, or duration of any sounds by electronic, mechanical, or any other means may be made at the performer's first stage

[146]: Lucier et al. (2005), *Reflections*

only. Further extensions or modifications should be made only by the environment or environment through which the sounds travel.". This also leaves room for sounds that are produced in the middle of the chain, in-between two systems as Lucier mentions in an interview: "I also stipulated in the score that, in some cases, the relay system may be opened up for persons to walk through, contributing their own sounds to the performance" [146].

[146]: Lucier et al. (2005), *Reflections*

As a precursor of network music, this piece had an influence in the field of networked music performances and multiple instances of telematic performances of the piece are documented [1].

1: See for example this recording of a performance organized by Matt Rogalsky and Laura Cameron in 2007 between 11 remote sites https://mattrogalsky.bandcamp.com/track/quasimodo-the-great-lover-alvin-lucier and this one dating from 2009 between two remote sites and led by Pauline Oliveros and Mark Dresser https://archive.org/details/QuasimodoTheGreatLover

This score strongly resonated with me during my reflections on using microphones in the context of distributed musical systems. Our team in the DOTS project was asked by Ircam to program a showcase of our developments with networks of Raspberry Pi computers by creating a few short musical pieces. My initial idea (that later evolved as described in this chapter) was to recreate *Quasimodo, The Great Lover* with the network of devices at our disposal. The goal was then to implement a system of microphone-equipped nano-computers that would amplify and transmit the captured sound through small loudspeakers. Hopefully, a sound emitted at one point in space would then travel from one small Raspberry Pi to its neighbors to reach a point farther in space. The result would then become a reactive installation that audience members could interact with to make sound travel over a whole room.

7.2 Technical implementation

7.2.1 Hardware

The piece was created and performed on a network of between 30 and 40 Raspberry Pi nano-computers equipped with microphone and loudspeakers. Each Raspberry Pi was equipped with a *HiFiBerry DAC+ ADC Pro* sound card and was powered using the official power supply during rehearsal sessions and using a portable battery during performances. In the following we refer to one of these units as a *dotpi* (plural *dotpi*s). See Figure 7.1 for an example of a *dotpi* unit.

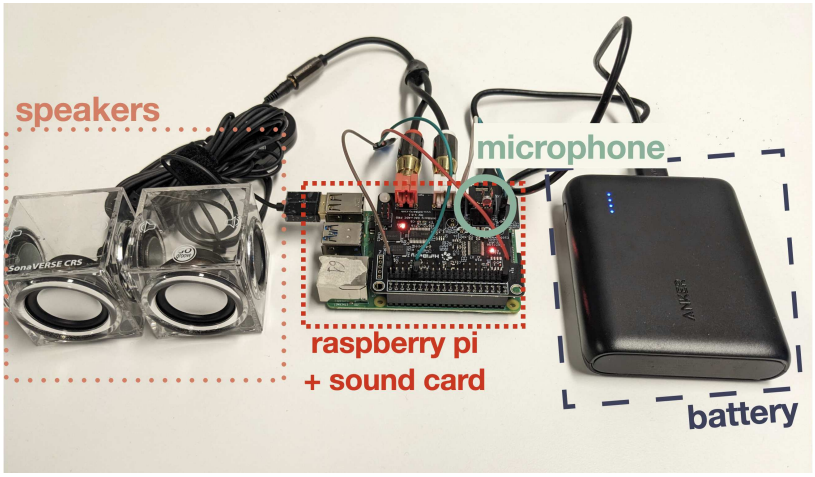


Figure 7.1: A fully working *dotpi* unit.

Small omni-directional electret microphones were used[2]. We started by connecting microphones to stereo jack plugs but quickly realized that the end result was not robust enough and that the soldering process was too long to meet production deadlines. We switched to soldering the microphones directly to headers to be plugged directly on the sound card's analogue output pins (cf Figure 7.2).

2: full specification available here: https://www.farnell.com/datasheets/2869767.pdf

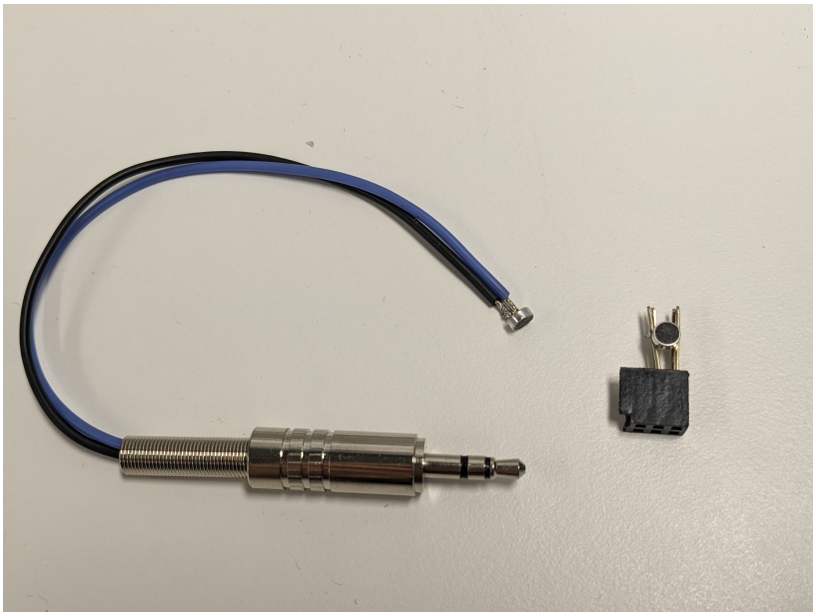


Figure 7.2: Two types of soldering techniques for the *dotpi*s' microphones: On the left, connected to a jack plug, on the right, soldered to a pin header.

Two types of loudspeakers were plugged to the RCA output of the *dotpi*s (cf. Figure 7.3 :

A pair of *GOgroove SonaVERSE CRS* speakers: small USB-powered loudspeakers. Although small and lacking in low-frequency sound rendition, they can be directly powered over the Raspberry Pi USB ports and offer sufficient volume.

A pair of *Creative Inspire T10* speakers: Larger loudspeakers delivering powerful sound with high-fidelity. They however need to be plugged to an AC outlet.

Figure 7.3: Two types of loudspeakers used. On the left: GOgroove SonaVERSE CRS speakers. On the right: Creative Inspire T10 speakers.

For some of the *dotpi*s, we replaced loudspeakers with transducers (cf Figure 7.4) that were fixed to musical instruments and resonating objects (drums, thunder sheets, cardboard boxes, etc. . .).

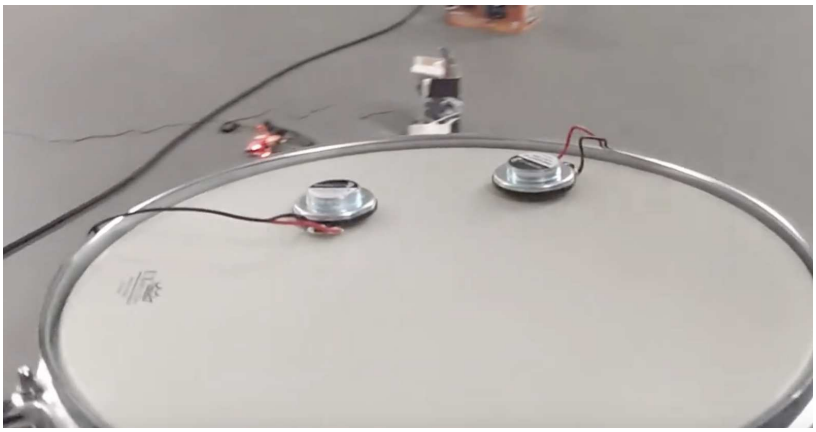


Figure 7.4: Two transducers connected to a *dotpi* and fixed to a drum.

40 *dotpi* units were installed in the *Espace de Projection* at Ircam. They were arranged in a grid as sketched in Figure 7.5. The desk with the Wi-Fi router and the server computer was installed at the center of the room during rehearsal but was moved to one of the corners of the room during performances. Figure 7.6 shows the full installation on the day of the performance.

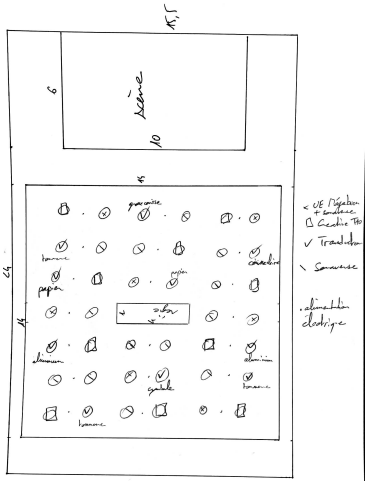


Figure 7.5: Floor plan of the *Espace de Projection* for the performance of *Quasimodots*. Each circle represents

Figure 7.6: The full installation in the *Espace de Projection* for the performance of *Quasimodots*. Each *dotpi* unit was placed in a cardboard box to facilitate logistics.

7.2.2 Software

A *soundworks* application was developed specifically for the piece. Two types of clients were implemented. A *Node.js* client for the *dotpi*s and a Web client for the control interface.

The *Node.js* client uses the *Node Web Audio API* to access the device's microphone. The microphone stream then goes through an audio path composed of different effects and is then connected to the audio output node to be played on the device's speakers. The audio path is described on Figure 7.7.

The microphone's stream first passes through a delay node. This was implemented to avoid having too much feedback as the device's microphone could capture the speakers' sound. The delayed sound is fed back to the delay node, providing a controllable feedback effect on each device The delayed sound then passes through a gain node for amplification.

Moreover, I decided to add a sound generator using granular synthesis. While first conceived as a quick way to stimulate the system of microphone-amplifier-loudspeaker, it also acquired its own aesthetic merit as the mass of granular sound blended well with the feedback sounds generated and created a mysterious atmosphere. Upon pressing a button on the interface, random grains of sounds are generated. The granular generator passes through a Attack-Decay-Sustain-Release envelope.

The amplified sound and the granular output are piped to a mixer to balance the volume of the two streams. The mix then passes through a 4-bands filter bank. Again, while implemented for the pragmatic purpose of cutting high frequency sounds and diminishing feedback over the system, the filter bank became an integral part of the instrumental practice of *Quasimodots*.

Finally, the audio stream goes through a gain node whose gain value is computed depending on a gaussian curve over the spatial disposition of devices: The further the device is from the center of the curve, the lower the gain. This was implemented to create spatial movement in the sound mass.

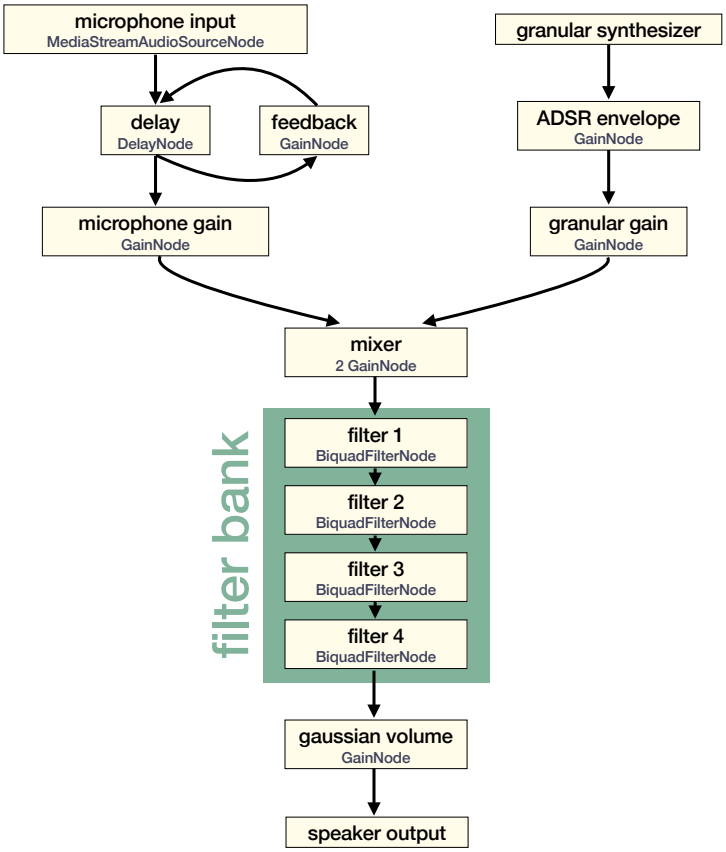


Figure 7.7: The audio path in *Quasimodots*

The control interface (cf Figure 7.8) provides controls to manipulate the parameters of the nodes in the audio path described above. Various sliders are used to control the various gains, the delay and the feedback, the ADSR envelope for the granular synthesis and the 4-bands filter bank. A bang button triggers granular synthesis over all the connected clients. A radio buttons menu enables selecting which sound will be used for granular synthesis. A map showing the spatial disposition of clients (displayed as

green dots)[3] is used to control the so-called "gaussian volume", that is the gain of each device computed on a gaussian 2-D curve. A yellow dot can be dragged to move the center of the curve. This gaussian volume can be toggled on and off. Finally, a list of all *dotpi*s is displayed, with the name of the sound file that is loaded for granular synthesis.

3: note that this spatial disposition in hardcoded in the application as we are not able to obtain the position of each client

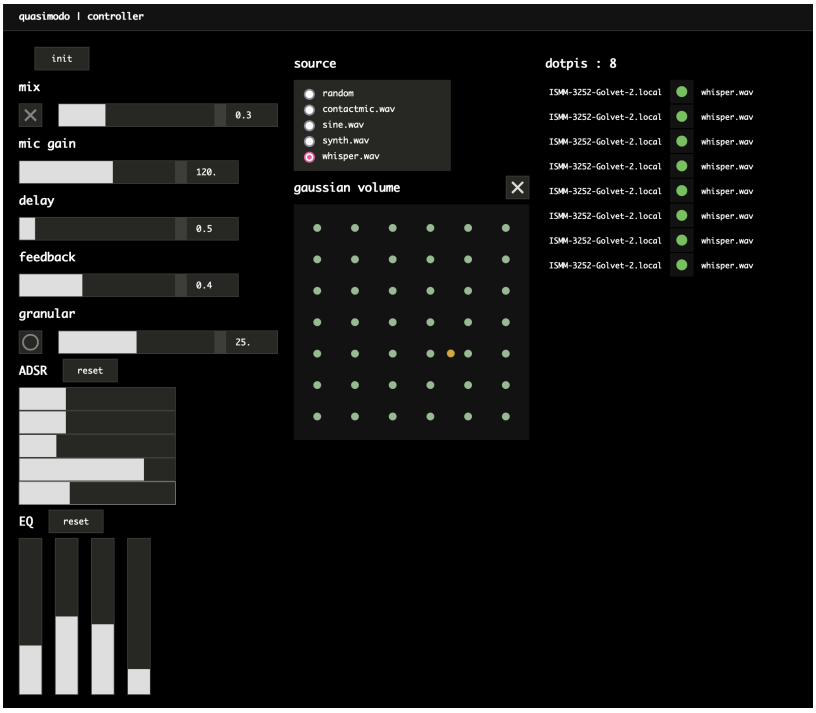


Figure 7.8: The controller interface of *Quasimodots* in a web page

7.3 Rehearsals and performance

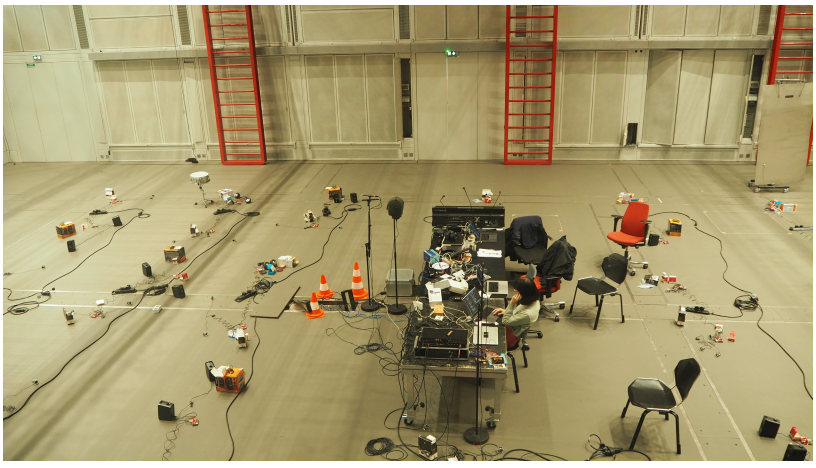


Figure 7.9: Rehearsing *Quasimodots* in the *Espace de Projection* at Ircam.

Rehearsals took place a few weeks before performance (cf Figure 7.9) and revealed the difficulty to reach the intended effect of the system. This was mainly caused by the precise balance in the amplifier gain that had to be found so as to make audio

transmission of sound from one *dotpi* to the next possible without causing too much internal feedback as one *dotpi*'s microphone would hear its own speaker's output. Speakers also had to be very carefully positioned and their volume precisely set so as not to create internal feedback.

This balance could be easily found in the setup described by Alvin Lucier in his piece as microphone-amplifier-speakers systems are placed linearly and are only interacting with the previous and the next system. In this case, amplifier gains can be set one after another. In our case, elements of our system were placed in the same room in a grid configuration. This resulted in much more interaction between every element, thus making the task of finding this balance exponentially more difficult.

Because setting up hardware elements proved to be long, tedious and difficult to reproduce, I implemented different software solutions to tame the volume of internal feedback generated by the system in the form of the delay and the filter bank.

During rehearsals, we tried performing with the system by placing some objects generating sounds (for instance a tape player) next to a *dotpi* or by playing an instrument or clapping/speaking into a *dotpi*'s microphone. This resulted in collective playing sessions faithful to the original artistic intent of the piece in which we could hear the sound of each player and sound source propagating over the system.

The piece was performed twice at Ircam in the *Espace de Projection* during the *Ateliers du Forum* in March 2024[4]. Each performance lasted around 5 minutes. Due to the event's configuration and the size of the audience, it quickly appeared that the accent had to be put on feedback and larsens rather than on the capacity of the system to propagate sounds. This resulted in a musical piece quite far from its original intent.

4: videos of rehearsals and the performance are visible at the following address `https://alienorgolvet.com/thesis/chapter-creation.html`

During performance, I realized that as most of the last evolutions of the system were implemented to tame the feedback, it could also be used to manipulate this feedback in a musical manner. I used mainly 3 techniques during performance : 1) Playing with the filter bank and especially with the lowest and highest frequency bands. The initial parameters of the filter bank were set on a very fragile equilibrium to limit feedback. Playing with the

filters would then intensify feedback and create a very dramatic change of sound from very high pitched sounds to a mass of low-frequency "growling" sounds washing over the audience. 2) Triggering granular synthesis over the whole system with a sound composed of a whispered voice that complemented the mysterious atmosphere set by the feedback sound and the darkness of the room. Due to the random triggering of granular synthesis on each *dotpi*, this would also create a feeling of spatialization. 3) Using the gaussian volume feature to create spatial movement over the room. Moving the center of the curve slowly would create very discernable movement while moving it quickly from one side to another would stop the feedback temporarily before making it reappear in another place.

During the second performance, fewer people were present in the room which allowed us to try playing with fewer feedback and leaving more room to sound transmission over the system. Some audience members even clapped next to some of the *dotpi*s's microphones. However it quickly appeared that feedback was still too strong and overpowered most of the sound transmission.

7.4 Discussion

7.4.1 Meeting the System Halfway

By reflecting on the whole process of composition of *Quasimodots*, we can observe that the end result was quite different from the original intention. While the original intent was to recreate a musical piece or at least to emulate its sonic effect based on the propagation of sound through different mediums, the performance of the piece was clearly focused on the manipulation of a mass of feedback. This stems directly from the approach we followed: experimental, prototypal, based on "hacking". To perform a faithful rendition of the score of *Quasimodo The Great Lover*, I would have had to use better microphones and loudspeakers and to carefully tune their placement.

Instead, I embraced the open-ended nature of the score and followed Lucier's own ethos when composing the piece as he mentioned in an interview on *Quasimodo The Great Lover* that "I'm adding things to the score that I didn't think of when we first

performed the piece. It was just a physical idea then, and the possibilities that came up were dependent on the actualities of the performances." [146].

[146]: Lucier et al. (2005), *Reflections*

Indeed, due to production constraints, I used what I had available to develop the system: small, quickly soldered microphones, a fixed placement of *dotpi*s and numerous elements were added to the system over rehearsals as a way to tame feedback. Over time, these elements changed status and became part of the panel of elements and gestures I used as I played the performance, even sometimes to amplify feedback.

Playing with *Quasimodots* required me to engage in a dialogue with the system and to acknowledge its fragility and sensibility that only appeared through its physical instantiation and could not be anticipated on paper. Always on the verge of collapsing, even sometimes becoming a frightening monster as a large mass of *dotpi*s coalescing into feedback, I learned to treat *Quasimodots* as another person with the care and caution it requires.

This recalls Burns and Burtner's comment in their paper on acoustic feedback in composition: "The complex, "messy" responses of feedback systems necessitate an intuitive approach in composition and performance; formal and sonic complexities result from the emergent properties of the system, interacting in the moment with the composer and performer. *Feedback systems will speak for themselves*."[5] they say. They add that "the potentials of the system at any moment and the range of influence of the controls can only be explored through improvisation" [147].

5: italics are mine

[147]: Burns et al. (2004), 'Recursive Audio Systems: Acoustic Feedback in Composition'

It also resonates with Elia and Overholt's conclusions on their distributed participatory installation based on audio feedback named *Squidback*[148] which shares some similarities with *Quasimodots*. In particular they say that "the lack of parametric control interfaces invites performative actions to undertake a *dialogic quality*, requiring receptivity for the system's own properties and development."[6].

[148]: Elia et al. (2021), 'Squidback: A Decentralized Generative Experience, Based on Audio Feedback from a Web Application Distributed to the Audience'

6: italics are mine

7.4.2 Ecosystemic Interactions

Elia and Overholt also mentions that their system "invites exploration of a control space that extends to the physical space where

the performance is situated, affected by acting physically on the devices and nearby objects."

Similarly, equipping a *dotpi* with a microphone (or with any other type of sensor) provides it with an awareness of its environment, it makes it able to react itself to its surroundings without being a slave to the performer's interaction, it has the power to make it an even more autonomous agent.

Such use of microphones can also bring the system into the field of *ecosystemic interactions* as described by Agostino Di Scipio[149]. Ecosystemic interactions aim to go beyond the common paradigm of interactive music systems in which the sound generating system is not able to directly change its internal state which is only determined by the performer's actions. An ecosystemic interactive system establishes a connection between the system and its environment thus taking the form of a "*machine/ambience* interrelationship" and a "triangular, ecosystemic connection, *man/ambience/machine*" (cf Figure 7.10). With such an approach, Di Scipio describes that "the computer *acts upon* the environment, observes the latter's response, and adapts itself, re-orienting the sequence of its internal states based on the data collected".

[149]: Di Scipio (2003), "Sound Is the Interface"

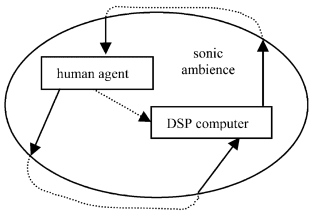


Figure 7.10: Ecosystemic interactions as described by Di Scipio. Figure taken from [149]

While in a strict sense *Quasimodots* does not qualify as an ecosystemic interface as the sound processing parameters are handled via the performer's interface, I noticed that equipping *dotpi*s with microphones provided them with an acute attention to their surroundings, amplifying every micro sounds produced and revealing the fragility of their arrangement and connection to the external world.

In a certain way, *Quasimodots* is a behavioral opposite of Jean-Luc Hervé's sound installation *Biotope* (2019)[7][52]. In *Biotope*, a network of Raspberry Pi computers recreate a living ecosystem with acousmatic sounds. Conceived as a "fearful acoustic system", the Raspberry Pi devices, equipped with presence sensors, react to human presence in the room by toning down the sound they produce. On the contrary, in *Quasimodots* the network of *dotpi*s reacts offensively to human presence and the inevitable noise they generate by screaming at them, by producing a mass of feedback that engulfs them and renders their presence undesirable.

7: *Biotope* was realized at Ircam and was created at Centre Pompidou, Paris. More info: `https://www.centrepompidou.fr/en/program/calendar/event/cbEzpqo`. Video here: `https://www.youtube.com/watch?v=RmSujqdT6L0`

[52]: Matuszewski (2020), 'Systèmes Musicaux Distribués et Technologies Web : Motivations, Design et Implémentation d'un Système Expérimental'

7.4.3 Towards a Modular, Ecosystemic, Distributed, Musical Creation System

Another situation also appeared during rehearsals: when feedback was limited, we started playing with the system as multiple players. Someone would clap next to a *dotpi*, someone else would play saxophone next to another one, a third person would place sound sources like tape players next to another *dotpi* and a fourth person would manipulate the control interface of the system. The system simultaneously took the form of a physical audio effect applied to our sound, another player alongside us and a spatial playground for collective interaction (cf video at `https://alienorgolvet.com/thesis/chapter-creation.html`).

This points towards the creation of a modular ecosystemic distributed musical creation system with *dotpi*s (cf. Figure 7.11). In such a system, *dotpi*s would be equipped with various sensors, providing them with an awareness of different dimensions of their physical environment: acoustic microphone, contact microphone, electromagnetic field microphone, accelerometer, presence sensor, light sensor, etc... For all these input signals, various information could be computed in real-time: signal intensity, frequency, acceleration etc... All these variables would be stored in the *dotpi*'s shared state to be distributed on the network and be used by other *dotpi*s.

Using another part of the interface, these incoming data (either from the *dotpi*'s sensors or from other *dotpi*s) could be connected to an audio path defined by the users with a code editor. For instance the frequency of the microphone of a *dotpi* could be connected to the frequency of a modulator. Or the acceleration from the accelerometer signal of another *dotpi* could be connected to the frequency of a filter.

Such a system would fit Di Scipio's definition of ecosystemic interaction as a paradigm in which one is not only directly interacting with the system but is "composing interactions".

Such a system would be adapted to different situations. Thanks to its portability, it could work as an autonomous outdoor installation, reacting to the natural environment for instance with a microphone transmitting the sound of a singing bird to an actuator stuck on a drum, or with a contact microphone transmitting the

vibration of a road to other dotpis' synthesizers. A similar idea has been explored in the *Legatus* project by Villicaña-Shaw et al. [150] described as an "environmentally-reactive soundscape augmentation artifact". This device however does not implement networking features.

[150]: Villicana-Shaw et al. (2023), 'Legatus: Design and Exhibition of Loudspeaker-Based, Environmentally-Reactive, Soundscape Augmentation Artifacts in Outdoor Natural Environments'

It could be also used by a group of musicians to augment collective interaction in the same manner as we did with *Quasimodots*. In that case, the group would benefit from the fact that the control interface that determines the path of communications within the network and the synthesis patches used would be simultaneously accessible by all members on their respective devices.

Hence, future works could consist in the implementation of such a system and in a long term study of how a group of musicians would appropriate such a system over various rehearsal sessions.

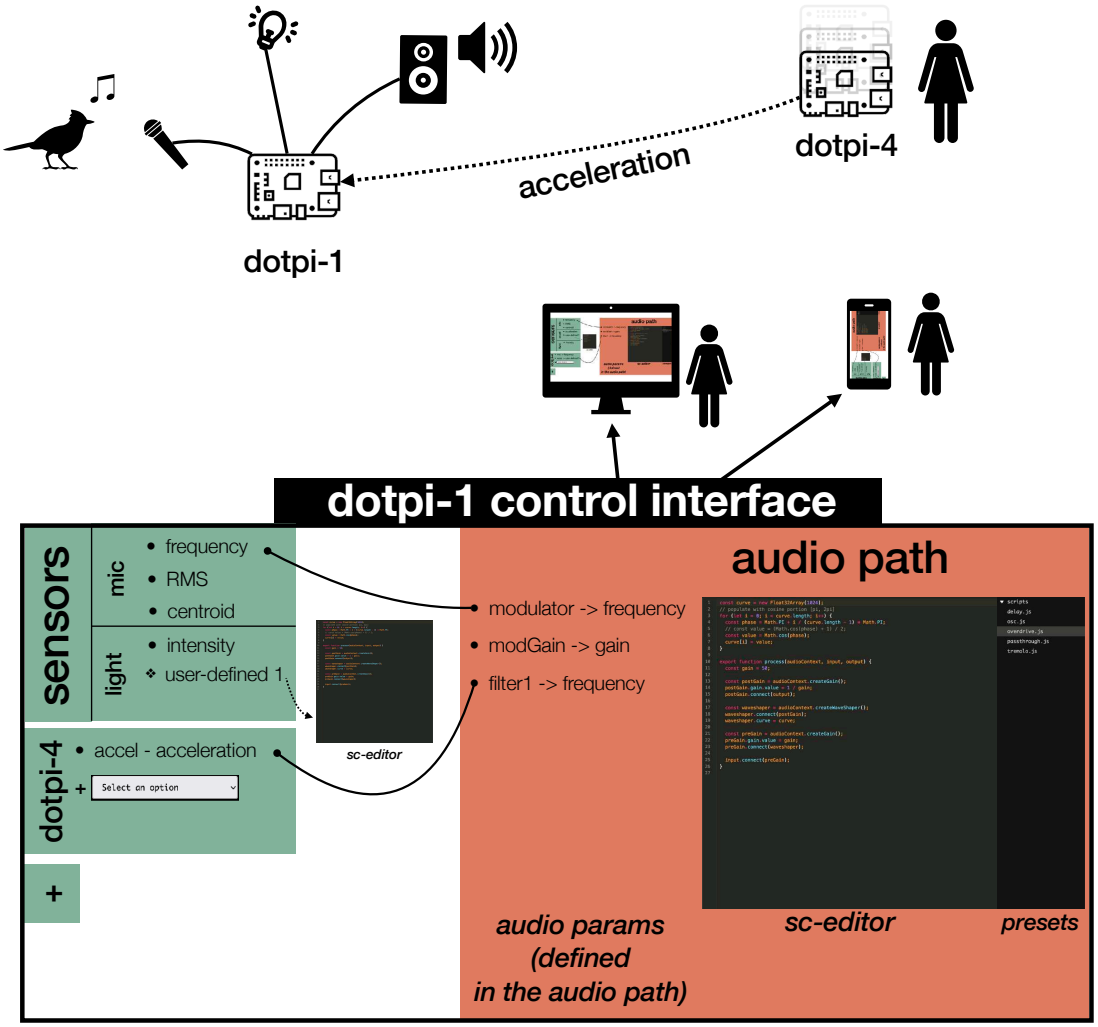


Figure 7.11: Diagram of a modular ecosystemic distributed creation system. Variables computed from the *dotpi*'s input signal or from other *dotpi*s are connected to audio parameters in an user-defined audio path.

7.5 Chapter summary

In this chapter, I documented the creation of *Quasimodots*, a musical piece for an ensemble of 40 Raspberry Pi devices equipped with microphones and speakers. This piece was originally inspired by a composition by Alvin Lucier based on the transmission of sound through different acoustic mediums by a chain of microphone-amplifier-speakers systems.

For the purpose of *Quasimodots*, we soldered 40 small microphones to plug on our Raspberry Pi and equipped them with portable speakers or actuators connected to various surfaces. I also developed a specific *soundworks* application in which the input sound of the Raspberry Pi's microphone is amplified and transmitted through the speakers. A control interface available in the web browser is used to control the parameters of the audio path.

The piece was rehearsed over multiple sessions and performed at the Ateliers du Forum 2024 in the Ircam's Espace de Projection. Rehearsal sessions revealed the difficulty in finding the right set of parameters and positioning of the *dotpi*s to avoid the appearance of internal feedback, thus complicating the realization of the original artistic intent. During performance, I realized that I had to embrace the existence of feedback and used the elements I implemented to tame this feedback to control it in a musical manner and even amplify it.

Finally, I discussed how the system's unpredictability and the experimental approach I had during the creation of the piece required me to engage in a dialogic approach to the system. In the end, I describe how equipping Raspberry Pi devices with microphones brings them into the field of Di Scipio's "ecosystemic interactions" and point toward the creation of a distributed ecosystemic creation system.

Conclusion

In this thesis we have investigated co-located distributed music systems, and how they can take place into musical practice and artistic research. To do this we employed a variety of design methodologies guided by a "project-grounded design" approach (cf. Chapter 1) rooted in practice and in collaborations with "expert users" (researchers and artists). We focused on the observation of emerging uses and user appropriation and tried to avoid a "techno-solutionist" approach that tends to solve problems solely identified through the development of novel technology. We were also interested in understanding how the systems we designed could operate within already existing habits, processes and technology.

In this PhD work, we developed multiple projects and software applications: *A³PM* , *Koryphaîos*, *Simone*, *Simone Solo*, and quasimodo. Each of them provided us with a different perspective on our research questions that appeared through long-term confrontation with real-life conditions of use and in experimental contexts. In accordance with the notion of experimental systems that was invoked at the beginning of this thesis, it is important to note the non-linearity of this research and the manner in which all these research projects themselves emerged and developed through interaction between each others, through technological innovations that were developed collectively and in the context of a specific socio-cultural environment and personal preferences of the author of this manuscript.

In the case of empirical musicology, we developed the *A³PM* tool where the distributed aspects allowed us to tackle constraints specific to the research methodology designed by the musicologist, which in turn allowed for the emergence of new use cases. In the case of musical composition, *Koryphaîos* enabled the exploration of novel compositional methods based on distributed sound diffusion. Our approach fostered user appropriation and was naturally integrated in software environments commonly used in contemporary composition. In the case of collective improvisation, networked arrangements in *Simone* enabled us to strengthen the sense of interaction between players and led to unique types of musical interaction. It also fostered discussion on how to distribute

agencies and interactions in collective networked systems. In the case of solo improvisation, novel performance and composition techniques appeared through Jean-Brice Godet's appropriation of *Simone Solo*. These techniques then infused the rest of his musical practice and fostered new creative approaches. In the case of the creation of *Quasimodots*, we observed the emergence of a specific form of agency and ecosystemic nature in the developed system. This forced us to radically adapt the manner in which we approached the system and to engage in a dialogue with it.

As a whole, we designed our systems as authoring tools and we believe all of our users were able to gain a sense of expertise and were inspired to learn the skills to reach a role beyond the one of a passive user.

As we based our design process on an ecosystem of web-based technologies developed by and with collaborators, it enabled us to fulfill our initial requirements during the design process: support of heterogeneous devices and interoperability between these devices and software components, easiness of deployment, approach oriented toward fast prototyping adapted to creative contexts.

As we choose to work on a variety of case studies, it might be difficult to guarantee that our results are generalizable. Nevertheless, as we argued in the introduction, generalization may not be desirable in such artistic contexts and we tried on purpose to value the contrasting, emerging and situated knowledge that each of our projects provided us.

Interestingly, it appears that a line could be drawn between all these projects: Distributed Music Systems, due to their inherent complexity, require an approach that is open to emergence and surprise. This is manifest in *Quasimodots*' feedbacks, in Jean-Brice reaction to playing with 40 *satellites*, in *Simone*'s users account of unpredictability, in the unpredicted use cases of A^3PM and the sonic effects of Luciano's compositions for *Koryphaîos*.

That is because of the often intractable number of interactions happening between all actors in a distributed system, because of the myriad of communication pathways that are drawn between them, because of the changing hierarchies and clusters that may arise and because of the often fragile and experimental nature of

technologies employed in this work. Networks require artists to let go of the idea of total control over all the elements of the system they use and this unpredictability and untamability needs to be taken into account in their practice. It requires artists to be attuned to the volatile behavior of the system.

The research project we have presented possesses some limitations.

First, we were confronted with the frequent difficulty to disseminate our tools and to build a community of users, which impeded the full implementation of design methodologies that generally require a large number of users. This problem, already common in academic research communities, is even more exacerbated in the case of a PhD thesis. Moreover, the few attempts I made to disseminate our software tools were also confronted with the fact that I lacked the time and resources needed to provide technical support or to develop "production-ready" versions of the software. Confronted with this fact, I found it simpler and more fruitful to follow methodologies that were not confronted to this problem such as the long-term involvement of a single artistic partner (Jean-Brice Godet with *Simone Solo*) or first-person research-creation methods (as part of the creation of *Quasimodots*). Nevertheless, the software remains open-source and available on GitHub, and could be used by the community.

Second, while all of our applications enable collaborative uses of distributed music networks, this potential of collaboration among users can be studied further. Such aspects have been studied in *Simone*, but also seen in very emerging use cases such as groups of musicians using A^3PM or playing collectively with *Quasimodots*. Further investigations in collaborative interaction, could be pursued in relation to collective and group-based musical practice.

Although with current limitations, our work opens up multiple perspectives. Future work may consist in pursuing the development of the various software and collaborations. In the case of A^3PM, this includes the development of a Graphical User Interface (GUI) enabling users to configure projects and the redaction of a more complete documentation. These will be necessary for the dissemination of the tool in the research community. As part of the collaboration with Jean-Brice, we plan to develop a "production" version of the *Simone Solo* system, including the design of watertight

boxes containing all the components of a sound source (Raspberry Pi, speakers, battery) for easy deployment and transport. This production version could also be entrusted to other artists. Finally, the creation of *Quasimodots* inspires other musical performances. In particular, I plan to use a reduced version (with 6 raspberry pi) of the system as part of my personal musical practice.

More generally, one of the objectives of the latter parts of this thesis, inspired by the author's practice of Field Recordings and Musique Concrète and the work of some music composers (such as Alvin Lucier to name the main one), was the investigation of environmental interactions within distributed music systems equipped with microphones and sensors. The creation of *Quasimodots* provided us with the experience of working with a system aware of its environment. Nonetheless, this question has yet to be investigated further. In Chapter 7 we sketched the concept of a modular, ecosystemic, distributed music system. The implementation of such a system and the observation of its long-term appropriation by groups of expert users or by the author herself remain a goal for the future.

Bibliography

Here are the references in citation order.

[1] *The Complete MIDI 1.0 Detailed Spec*. Los Angeles, California, United States, 1996 (cited on page 1).

[2] Matthew Wright and Adrian Freed. 'Open SoundControl: A New Protocol for Communicating with Sound Synthesizers'. In: *International Computer Music Conference (ICMC)*. Thessaloniki, Greece, 1997 (cited on page 1).

[3] Florian Goltz. 'Ableton Link – A Technology to Synchronize Music Software'. In: *Linux Audio Conference*. Berlin, Germany, 2018 (cited on page 1).

[4] Scot Gresham-Lancaster. 'A Personal Reminiscence on the Roots of Computer Network Music'. In: *Leonardo Music Journal* 27 (Dec. 2017), pp. 71–77. DOI: 10.1162/LMJ_a_01022. (Visited on 06/10/2022) (cited on pages 1, 12, 95).

[5] Mark Weiser. 'The Computer for the 21st Century'. In: *Scientific American* 265.3 (1991), p. 94. (Visited on 10/01/2021) (cited on pages 1, 23).

[6] Maarten Van Steen and Andrew S. Tanenbaum. 'A Brief Introduction to Distributed Systems'. In: *Computing* 98.10 (Oct. 2016), pp. 967–1009. DOI: 10.1007/s00607-016-0508-7. (Visited on 04/30/2024) (cited on pages 2, 46).

[7] Bruno Latour. 'On Actor-Network Theory. A Few Clarifications, Plus More Than a Few Complications'. In: *Soziale Welt* 47 (1996), pp. 369–381 (cited on pages 2, 22, 120).

[8] Frederic Bevilacqua et al. 'On Designing, Composing and Performing Networked Collective Interactions'. In: *Organised Sound* 26.3 (Dec. 2021), pp. 333–339. DOI: 10.1017/S135577182100042X. (Visited on 01/25/2022) (cited on pages 3, 22, 76).

[9] Marcelo M. Wanderley and Wendy E. Mackay. 'HCI, Music and Art: An Interview with Wendy Mackay'. In: *New Directions in Music and Human-Computer Interaction*. Ed. by Simon Holland et al. Cham: Springer International Publishing, 2019, pp. 115–120. DOI: 10.1007/978-3-319-92069-6_7. (Visited on 10/14/2024) (cited on page 3).

[10] Matthew Rodger et al. 'What Makes a Good Musical Instrument? A Matter of Processes, Ecologies and Specificities'. In: *International Conference on New Interfaces for Musical Expression (NIME)*. Birmingham, UK, 2020 (cited on page 3).

[11] Paul Dourish. 'The Appropriation of Interactive Technologies: Some Lessons from Placeless Documents'. In: *Computer Supported Cooperative Work (CSCW)* 12.4 (Dec. 2003), pp. 465–490. DOI: 10.1023/A:1026149119426. (Visited on 10/01/2021) (cited on pages 4, 31).

[12] Julian Rohrhuber. 'Network Music'. In: *The Cambridge Companion to Electronic Music*. Ed. by Nick Collins and Julio d'Escrivan. 1st ed. Cambridge University Press, Dec. 2007, pp. 140–155. DOI: 10.1017/CCOL9780521868617.010. (Visited on 01/23/2023) (cited on pages 4, 12, 13, 95).

[13] William Gaver et al. 'Emergence as a Feature of Practice-based Design Research'. In: *Designing Interactive Systems Conference*. Virtual Event Australia: ACM, June 2022, pp. 517–526. DOI: 10.1145/3532106.3533524. (Visited on 10/13/2024) (cited on page 4).

[14] Alain Findeli et al. 'Research through Design and Transciplinarity: A Tentative Contribution to the Methodology of Design Research'. In: *Swiss Design Network Symposium*. Bern, Switzerland, 2008 (cited on pages 4, 28).

[15] Alain Findeli. 'Searching for Design Research Questions: Some Conceptual Clarifications'. In: *Mapping Design Research*. Ed. by Simon Grand. Basel: Birkhäuser, 2012 (cited on pages 5, 95).

[16] Sarah Fdili Alaoui. 'Dance-Led Research'. PhD thesis. Université Paris Saclay, 2023 (cited on pages 5, 30).

[17] Sarah Fdili Alaoui. 'Making an Interactive Dance Piece: Tensions in Integrating Technology in Art'. In: *Proceedings of the 2019 on Designing Interactive Systems Conference*. San Diego CA USA: ACM, June 2019, pp. 1195–1208. DOI: 10.1145/3322276.3322289. (Visited on 10/13/2024) (cited on page 5).

[18] Hans-Jörg Rheinberger. 'Consistency from the Perspective of an Experimental Systems Approach to the Sciences and Their Epistemic Objects'. In: *Manuscrito* 34.1 (June 2011), pp. 307–321. DOI: 10.1590/S0100-60452011000100014. (Visited on 10/01/2021) (cited on pages 5, 25, 26).

[19] Aliénor Golvet et al. 'With, against, or without? Familiarity and Copresence Increase Interactional Dissensus and Relational Plasticity in Freely Improvising Duos.' In: *Psychology of Aesthetics, Creativity, and the Arts* 18.2 (Sept. 2021), pp. 182–195. DOI: 10.1037/aca0000422. (Visited on 09/29/2022) (cited on pages 9, 55, 65, 66).

[20] Emanuelle Majeau-Bettez, Aliénor Golvet, and Clément Canonne. 'Tracking Auditory Attention in Group Performances: A Case Study on Éliane Radigue's *Occam Delta XV*'. In: *Musicae Scientiae* (Oct. 2023). DOI: 10.1177/10298649231203641. (Visited on 10/26/2023) (cited on pages 9, 55, 66).

[21] Aliénor Golvet, Benjamin Matuszewski, and Frédéric Bevilacqua. 'SIMONE : Un Instrument Distribué Pour l'improvisation Musicale'. In: *Revue Francophone Informatique et Musique* 10 (2024). DOI: 10.56698/rfim.766. (Visited on 09/24/2024) (cited on pages 9, 66, 91).

[22] Aliénor Golvet et al. 'Koryphaîos A Patchworked Compositional Environment for Distributed Music Systems'. In: *Web Audio Conference*. Cannes, France, 2022. DOI: 10.5281/ZENODO.6767566. (Visited on 09/07/2022) (cited on pages 9, 76).

[23] Aliénor Golvet, Benjamin Matuszewski, and Frederic Bevilacqua. 'Simone : Un Instrument Distribué Pour l'étude Des Interactions Improvisées Collectives'. In: *Journées d'Informatique Musicale (JIM 2023)*. Saint-Denis, France, 2023 (cited on page 9).

[24] Benjamin Matuszewski and Aliénor Golvet. 'Rapid Prototyping of Distributed Musical Things Using Web Technologies'. In: *2023 4th International Symposium on the Internet of Sounds*. Pisa, Italy: IEEE, Oct. 2023, pp. 1–5. DOI: 10.1109/IEEECONF59510.2023.10335368. (Visited on 02/06/2024) (cited on pages 9, 52).

[25] Aliénor Golvet, Benjamin Matuszewski, and Frederic Bevilacqua. 'Designing a Distributed Musical Instrument for Collective Improvised Interaction'. In: *Audio Mostly 2024 - Explorations in Sonic Cultures*. Milan Italy: ACM, Sept. 2024, pp. 405–420. DOI: 10.1145/3678299.3678341. (Visited on 09/24/2024) (cited on pages 10, 91).

[26] Scot Gresham-Lancaster. 'The Aesthetics and History of the Hub: The Effects of Changing Technology on Network Computer Music'. In: *Leonardo Music Journal* 8 (1998), p. 39. DOI: 10.2307/1513398. (Visited on 06/10/2022) (cited on page 12).

[27] Nick Collins. *Introduction to Computer Music*. Hoboken: John Wiley & Sons, 2010 (cited on pages 13, 95).

[28] Golo Föllmer. 'Lines of Net Music'. In: *Contemporary Music Review* 24.6 (Dec. 2005), pp. 439–444. DOI: 10.1080/07494460500296102. (Visited on 01/23/2023) (cited on pages 14, 15).

[29] Peter Traub. 'Sounding the Net: Recent Sonic Works for the Internet and Computer Networks'. In: *Contemporary Music Review* 24.6 (Dec. 2005), pp. 459–481. DOI: 10.1080/07494460500296136. (Visited on 01/23/2023) (cited on pages 14, 15).

[30] Chris Chafe. 'Tapping into the Internet as an Acoustical/Musical Medium'. In: *Contemporary Music Review* 28.4-5 (Aug. 2009), pp. 413–420. DOI: 10.1080/07494460903422362. (Visited on 10/06/2024) (cited on page 14).

[31] Atau Tanaka and Bert Bongers. 'Global String: A Musical Instrument for Hybrid Space'. In: *International Computer Music Conference (ICMC)*. Gothenburg, Sweden, 2002 (cited on page 14).

[32] Chris Chafe, Scott Wilson, and Daniel Walling. 'Physical Model Synthesis with Application to Internet Acoustics'. In: *IEEE International Conference on Acoustics Speech and Signal Processing*. Orlando, FL, USA: IEEE, May 2002, pp. IV-4056-IV–4059. DOI: 10.1109/ICASSP.2002.5745548. (Visited on 10/06/2024) (cited on page 14).

[33] Jason Freeman et al. 'Auracle: A Voice-Controlled, Networked Sound Instrument'. In: *Organised Sound* 10.3 (Dec. 2005), pp. 221–231. DOI: 10.1017/S1355771805000968. (Visited on 06/15/2022) (cited on pages 15, 16, 96).

[34] Sergi Jordà. 'FMOL: Toward User-Friendly, Sophisticated New Musical Instruments'. In: *Computer Music Journal* 26.3 (Sept. 2002), pp. 23–39. DOI: 10.1162/014892602320582954. (Visited on 10/07/2024) (cited on page 15).

[35] Georg Hajdu. 'Quintet.Net: An Environment for Composing and Performing Music on the Internet'. In: *Leonardo* 38.1 (Feb. 2005), pp. 23–30. DOI: 10.1162/leon.2005.38.1.23. (Visited on 10/07/2024) (cited on pages 16, 17).

[36] Jason Freeman. 'N.A.G.: Network Auralization for Gnutella'. In: *Proceedings of the 12th Annual ACM International Conference on Multimedia*. New York NY USA: ACM, Oct. 2004, pp. 180–181. DOI: 10.1145/1027527.1027567. (Visited on 10/06/2024) (cited on page 17).

[37] Ariane Stolfi et al. 'Participatory Musical Improvisations with Playsound.Space'. In: *Web Audio Conference WAC-2018*. Berlin, Germany, 2018, p. 5 (cited on page 17).

[38] Ariane S Stolfi and Mathieu Barthet. 'Improvisation in Isolation: Quarentena Liv(r)e and Noise Symphony with the Playsound Online Music Making Tool'. In: *Web Audio Conference WAC-2021*. Barcelona, Spain, 2021, p. 6 (cited on page 17).

[39] Benjamin Taylor. 'A History of the Audience as a Speaker Array'. In: *Proceedings of the International Conference on New Interfaces for Musical Expression*. Copenhagen, Denmark, 2017 (cited on pages 18, 73).

[40] Jieun Oh and Ge Wang. 'Audience-Participation Techniques Based on Social Mobile Computing'. In: *Proceedings of the International Computer Music Conference 2011*. University of Huddersfield, UK, 2011 (cited on page 18).

[41] Oliver Hödl et al. 'Large-Scale Audience Participation in Live Music Using Smartphones'. In: *Journal of New Music Research* 49.2 (Mar. 2020), pp. 192–207. DOI: 10.1080/09298215.2020.1722181. (Visited on 01/24/2022) (cited on page 18).

[42] Anna Xambó and Gerard Roma. 'Performing Audiences: International Conference on New Interfaces for Musical Expression'. In: *Proceedings of the International Conference on New Interfaces for Musical Expression 2020*. Ed. by Romain Michon and Franziska Schroeder. Proceedings of the Conference on New Interface for Musical Expression (NIME). Birmingham City University, July 2020, pp. 55–60. (Visited on 06/07/2023) (cited on page 18).

[43] Golo Föllmer. 'Electronic, Aesthetic and Social Factors in Net Music'. In: *Organised Sound* 10.3 (Dec. 2005), pp. 185–192. DOI: 10.1017/S1355771805000920. (Visited on 10/07/2024) (cited on page 19).

[44] Álvaro Barbosa. 'Displaced Soundscapes: A Survey of Network Systems for Music and Sonic Art Creation'. In: *Leonardo Music Journal* 13 (Dec. 2003), pp. 53–59. DOI: 10.1162/096112104322750791. (Visited on 06/15/2022) (cited on pages 19, 20).

[45] Gil Weinberg. 'Interconnected Musical Networks: Toward a Theoretical Framework'. In: *Computer Music Journal* 29.2 (2005), pp. 23–39 (cited on pages 19, 21, 92, 96, 99, 117).

[46] Benjamin Matuszewski, Norbert Schnell, and Frederic Bevilacqua. 'Interaction Topologies in Mobile-Based Situated Networked Music Systems'. In: *Wireless Communications and Mobile Computing* 2019 (Mar. 2019), pp. 1–9. DOI: 10.1155/2019/9142490. (Visited on 10/28/2021) (cited on pages 22, 97).

[47] Matthew P. Aylett and Aaron J. Quigley. 'The Broken Dream of Pervasive Sentient Ambient Calm Invisible Ubiquitous Computing'. In: *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*. Seoul Republic of Korea: ACM, Apr. 2015, pp. 425–435. DOI: 10.1145/2702613.2732508. (Visited on 10/01/2021) (cited on page 24).

[48] Damián Keller and Victor Lazzarini. 'Theoretical Approaches to Musical Creativity: The Ubimus Perspective'. In: *Musica Theorica* 2.1 (Jan. 2018). DOI: 10.52930/mt.v2i1.33. (Visited on 10/08/2024) (cited on page 24).

[49] Flávio Luiz Schiavoni, Pedro H. De Faria, and Jônatas Manzolli. 'Interaction and Collaboration in Computer Music Using Computer Networks: An UbiMus Perspective'. In: *Journal of New Music Research* 48.4 (Aug. 2019), pp. 316–330. DOI: 10.1080/09298215.2019.1635626. (Visited on 05/04/2024) (cited on page 24).

[50] Luca Turchet et al. 'Internet of Musical Things: Vision and Challenges'. In: *IEEE Access* 6 (2018), pp. 61994–62017. DOI: 10.1109/ACCESS.2018.2872625. (Visited on 09/29/2022) (cited on page 24).

[51] Michael Schwab et al., eds. *Experimental Systems: Future Knowledge in Artistic Research*. Orpheus Institute Series. Leuven: Leuven University Press, 2013 (cited on page 26).

[52] Benjamin Matuszewski. 'Systèmes Musicaux Distribués et Technologies Web : Motivations, Design et Implémentation d'un Système Expérimental'. PhD thesis. Paris, France: Université Paris 8, 2020 (cited on pages 27, 165).

[53] Elizabeth Goodman, Erik Stolterman, and Ron Wakkary. 'Understanding Interaction Design Practices'. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Vancouver BC Canada: ACM, May 2011, pp. 1061–1070. DOI: 10.1145/1978942.1979100. (Visited on 10/13/2024) (cited on page 27).

[54] Steve Benford et al. 'Performance-Led Research in the Wild'. In: *ACM Transactions on Computer-Human Interaction* 20.3 (July 2013), pp. 1–22. DOI: 10.1145/2491500.2491502. (Visited on 10/13/2024) (cited on page 27).

[55] Sophie Stévance and Lacasse Serge. *Les enjeux de la recherche-création en musique: Institution, définition, formation*. Hors-Collection. Québec: Les Presses de l'Université Laval, 2014 (cited on page 28).

[56] Erin Manning. 'Ten Propositions for Research-Creation'. In: *Collaboration in Performance Practice*. Ed. by Noyale Colin and Stefanie Sachsenmaier. London: Palgrave Macmillan UK, 2016, pp. 133–141. DOI: 10.1057/9781137462466_7. (Visited on 10/13/2024) (cited on page 28).

[57] Erin Manning. 'Against Method'. In: *The Minor Gesture*. Duke University Press, 2016 (cited on page 28).

[58] Stephanie Springgay and Nikki Rotas. 'How Do You Make a Classroom Operate like a Work of Art? Deleuzeguattarian Methodologies of Research-Creation'. In: *International Journal of Qualitative Studies in Education* 28.5 (May 2015), pp. 552–572. DOI: 10.1080/09518398.2014.933913. (Visited on 10/13/2024) (cited on page 28).

[59] Christopher Frayling. 'Research in Art and Design'. In: *Royal College of Art, Research Papers* 1.1 (1993) (cited on page 28).

[60] Johan Redström. 'Research through and through Design'. In: *Artifact* 8.1 (Dec. 2021), pp. 16.1–16.19. DOI: 10.1386/art_00016_1. (Visited on 04/29/2024) (cited on pages 28, 29, 94).

[61] William Gaver. 'What Should We Expect from Research through Design?' In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Austin Texas USA: ACM, May 2012, pp. 937–946. DOI: 10.1145/2207676.2208538. (Visited on 04/29/2024) (cited on pages 28, 94).

[62] John Zimmerman and Jodi Forlizzi. 'The Role of Design Artifacts in Design Theory Construction'. In: *Artifact* 2.1 (Apr. 2008), pp. 41–45. DOI: 10.1080/17493460802276893. (Visited on 05/06/2024) (cited on pages 28, 94).

[63] Koray Tahıroğlu et al. 'Digital Musical Instruments as Probes: How Computation Changes the Mode-of-Being of Musical Instruments'. In: *Organised Sound* 25.1 (Apr. 2020), pp. 64–74. DOI: 10.1017/S1355771819000475. (Visited on 01/12/2022) (cited on pages 29, 75).

[64] O. Tomico, V. O. Winthagen, and M. M. G. Van Heist. 'Designing for, with or within: 1 st , 2 nd and 3 rd Person Points of View on Designing for Systems'. In: *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design*. Copenhagen Denmark: ACM, Oct. 2012, pp. 180–188. DOI: 10.1145/2399016.2399045. (Visited on 10/12/2024) (cited on page 29).

[65] Wina Smeenk, Oscar Tomico, and Koen van Turnhout. 'A Systematic Analysis of Mixed Perspectives in Empathic Design: Not One Perspective Encompasses All'. In: *International Journal of Design* 10.2 (2016), pp. 31–48 (cited on page 29).

[66] James Pierce et al. 'Expanding and Refining Design and Criticality in HCI'. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. Seoul Republic of Korea: ACM, Apr. 2015, pp. 2083–2092. DOI: 10.1145/2702123.2702438. (Visited on 10/12/2024) (cited on pages 29, 30).

[67] Michael J. Muller and Sarah Kuhn. 'Participatory Design'. In: *Communications of the ACM* 36.6 (June 1993), pp. 24–28. DOI: 10.1145/153571.255960. (Visited on 10/12/2024) (cited on page 29).

[68] Wendy E. Mackay and Michel Beaudoin-Lafon. 'Participatory Design and Prototyping'. In: *Handbook of Human Computer Interaction*. Springer International Publishing. 2023, pp. 1–33 (cited on pages 29, 30).

[69] Andrés Lucero, Kirsikka Vaajakallio, and Peter Dalsgaard. 'The Dialogue-Labs Method: Process, Space and Materials as Structuring Elements to Spark Dialogue in Co-Design Events'. In: *CoDesign* 8.1 (Mar. 2012), pp. 1–23. DOI: 10.1080/15710882.2011.609888. (Visited on 10/12/2024) (cited on page 29).

[70] Audrey Desjardins et al. 'Introduction to the Special Issue on First-Person Methods in HCI'. In: *ACM Transactions on Computer-Human Interaction* 28.6 (Dec. 2021), pp. 1–12. DOI: 10.1145/3492342. (Visited on 10/12/2024) (cited on page 30).

[71] Tove Grimstad Bang et al. 'A Retrospective Autoethnography Documenting Dance Learning Through Data Physicalisations'. In: *Designing Interactive Systems Conference*. IT University of Copenhagen Denmark: ACM, July 2024, pp. 2357–2373. DOI: 10.1145/3643834.3661607. (Visited on 10/12/2024) (cited on page 30).

[72] Carman Neustaedter and Phoebe Sengers. 'Autobiographical Design in HCI Research: Designing and Learning through Use-It-Yourself'. In: *Proceedings of the Designing Interactive Systems Conference*. Newcastle Upon Tyne United Kingdom: ACM, June 2012, pp. 514–523. DOI: 10.1145/2317956.2318034. (Visited on 10/12/2024) (cited on page 30).

[73] William W. Gaver. 'The Video Window: My Life with a Ludic System'. In: *Personal and Ubiquitous Computing* 10.2-3 (Apr. 2006), pp. 60–65. DOI: 10.1007/s00779-005-0002-2. (Visited on 10/12/2024) (cited on page 30).

[74] Mirjana Prpa et al. 'Articulating Experience: Reflections from Experts Applying Micro-Phenomenology to Design Research in HCI'. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Honolulu HI USA: ACM, Apr. 2020, pp. 1–14. DOI: 10.1145/3313831.3376664. (Visited on 10/12/2024) (cited on page 30).

[75] Xiao Zhang and Ron Wakkary. 'Understanding the Role of Designers' Personal Experiences in Interaction Design Practice'. In: *Proceedings of the 2014 Conference on Designing Interactive Systems*. Vancouver BC Canada: ACM, June 2014, pp. 895–904. DOI: 10.1145/2598510.2598556. (Visited on 10/12/2024) (cited on page 30).

[76] Wendy E. Mackay. 'Patterns of Sharing Customizable Software'. In: *Proceedings of the 1990 ACM Conference on Computer-supported Cooperative Work - CSCW '90*. Los Angeles, California, United States: ACM Press, 1990, pp. 209–221. DOI: 10.1145/99332.99356. (Visited on 10/02/2024) (cited on page 31).

[77] William W. Gaver, Jacob Beaver, and Steve Benford. 'Ambiguity as a Resource for Design'. In: *Proceedings of the Conference on Human Factors in Computing Systems - CHI '03*. Ft. Lauderdale, Florida, USA: ACM Press, 2003, p. 233. DOI: 10.1145/642611.642653. (Visited on 09/13/2022) (cited on page 31).

[78] Jennie Carroll. 'Completing Design in Use: Closing the Appropriation Cycle'. In: *ECIS 2004 Proceedings*. 2004 (cited on page 31).

[79] Alan Dix. 'Designing for Appropriation'. In: *Proceedings of the 21st BCS HCI Group Conference*. Lancaster University, UK: British Computer Society, 2007 (cited on page 31).

[80] Mark W. Newman et al. 'Designing for Serendipity: Supporting End-User Configuration of Ubiquitous Computing Environments'. In: *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*. London England: ACM, June 2002, pp. 147–156. DOI: 10.1145/778712.778736. (Visited on 10/02/2024) (cited on page 31).

[81] Gerhard Fischer and Elisa Giaccardi. 'Meta-Design: A Framework for the Future of End-User Development'. In: *End User Development*. Ed. by Henry Lieberman, Fabio Paternò, and Volker Wulf. Vol. 9. Dordrecht: Springer Netherlands, 2006, pp. 427–457. DOI: 10.1007/1-4020-5386-X_19. (Visited on 11/05/2021) (cited on pages 31, 32, 74, 75).

[82] Victor Zappi and Andrew P McPherson. 'Dimensionality and Appropriation in Digital Musical Instrument Design'. In: *International Conference on New Interfaces for Musical Expression (NIME '14)*. London, UK, 2014 (cited on pages 32, 94).

[83] Thor Magnusson. 'Designing Constraints: Composing and Performing with Digital Musical Systems'. In: *Computer Music Journal* 34.4 (Dec. 2010), pp. 62–73. DOI: 10.1162/COMJ_a_00026. (Visited on 01/14/2022) (cited on pages 32, 74, 119).

[84] Andrew McPherson and Koray Tahıroğlu. 'Idiomatic Patterns and Aesthetic Influence in Computer Music Languages'. In: *Organised Sound* 25.1 (Apr. 2020), pp. 53–63. DOI: 10.1017/S1355771819000463. (Visited on 02/21/2022) (cited on page 32).

[85] Andrew McPherson and Victor Zappi. 'An Environment for Submillisecond- Latency Audio and Sensor Processing on BeagleBone Black'. In: *Audio Engineering Society Convention*. Warsaw, Poland, 2015 (cited on page 37).

[86] Stéphane Letz, Yann Orlarey, and Dominique Fober. 'Jack Audio Server for Multi-Processor Machines'. In: *International Computer Music Conference*. Barcelona, Spain, 2005 (cited on page 38).

[87] Benjamin Matuszewski and Otto Rottier. 'The Web Audio API as a Standardized Interface Beyond Web Browsers'. In: *Journal of the Audio Engineering Society* 71.11 (Nov. 2023), pp. 790–801. DOI: 10.17743/jaes.2022.0114 (cited on page 41).

[88] Benjamin Matuszewski. 'A Web-Based Framework for Distributed Music System Research and Creation'. In: *Journal of the Audio Engineering Society* 68.10 (Dec. 2020), pp. 717–726. DOI: 10.17743/jaes.2020.0015. (Visited on 10/12/2021) (cited on pages 42, 73).

[89] Jean-Philippe Lambert, Sébastien Robaszkiewicz, and Norbert Schnell. 'Synchronisation for Distributed Audio Rendering over Heterogeneous Devices, in HTML5'. In: *Proceedings of the 2nd Web Audio Conference (WAC-2016)*. Atlanta, USA, 2016 (cited on pages 46, 79).

[90] *WebMidi API Specification*. https://www.w3.org/TR/webmidi/ (cited on page 49).

[91] Michel Buffa et al. 'Web Audio Modules 2.0: An Open Web Audio Plugin Standard'. In: *Companion Proceedings of the Web Conference 2022*. Virtual Event, Lyon France: ACM, Apr. 2022, pp. 364–369. DOI: 10.1145/3487553.3524225. (Visited on 10/18/2024) (cited on page 54).

[92] Clément Canonne and Fanny Gribenski, eds. *New Methods and New Challenges in Empirical Musicology*. Oxford University Press, Forthcoming (cited on page 56).

[93] Jacques Theureau. 'Les Entretiens d'autoconfrontation et de Remise En Situation Par Les Traces Matérielles et Le Programme de Recherche « Cours d'action »'. In: *Revue d'anthropologie des connaissances* 4.1 (2010). DOI: 10.3917/rac.010.0287. (Visited on 07/03/2024) (cited on page 56).

[94] Jacques Theureau and Nicolas Donin. 'Comprendre Une Activité de Composition Musicale : Les Relations Entre Sujet, Activité Créatrice, Environnement et Conscience Préréflexive:' in: *Sujets, Activités, Environnements*. Presses Universitaires de France, Feb. 2006, pp. 221–251. DOI: 10.3917/puf.barbi.2006.01.0221. (Visited on 07/03/2024) (cited on page 56).

[95] Kees Dorst. 'Co-Evolution and Emergence in Design'. In: *Design Studies* 65 (Nov. 2019), pp. 60–77. DOI: 10.1016/j.destud.2019.10.005. (Visited on 10/15/2024) (cited on page 56).

[96] Louise Goupil et al. 'Musical Coordination in a Large Group without Plans nor Leaders'. In: *Scientific Reports* 10.1 (Nov. 2020), p. 20377. DOI: 10.1038/s41598-020-77263-z. (Visited on 10/26/2023) (cited on pages 57, 65).

[97] Thomas Wolf, Louise Goupil, and Clément Canonne. 'Beyond Togetherness: Interactional Dissensus Fosters Creativity and Tension in Freely Improvised Musical Duos.' In: *Psychology of Aesthetics, Creativity, and the Arts* (June 2023). DOI: 10.1037/aca0000588. (Visited on 07/10/2024) (cited on page 66).

[98] Arthur Faraco et al. 'Listening Behaviors and Musical Coordination in Collective Free Improvisation'. In: *Music & Science* 7 (Jan. 2024), p. 20592043241257023. DOI: 10.1177/20592043241257023. (Visited on 07/10/2024) (cited on page 67).

[99] Per Hetland. 'The Quest for Reciprocity: Citizen Science as a Form of Gift Exchange'. In: *A History of Participation in Museums and Archives*. London: Routledge, 2020, pp. 257–277 (cited on page 71).

[100] Carl Wilmsen. 'Participation, Reciprocity, and Empowerment in the Practice of Participatory Research'. In: 2006 (cited on page 71).

[101] Jesse Allison, Yemin Oh, and Benjamin Taylor. 'NEXUS: Collaborative Performance for the Masses, Handling Instrument Interface Distribution through the Web'. In: *International Conference on New Interfaces for Musical Expression (NIME)*. Daejeon, Korea, 2013 (cited on page 73).

[102] Jean Bresson, Carlos Agon, and Gérard Assayag. 'OpenMusic: Visual Programming Environment for Music Composition, Analysis and Research'. In: *Proceedings of the 19th ACM International Conference on Multimedia - MM '11*. Scottsdale, Arizona, USA: ACM Press, 2011, p. 743. DOI: 10.1145/2072298.2072434. (Visited on 01/31/2022) (cited on pages 73, 76).

[103] Andrea Agostini and Daniele Ghisi. 'A Max Library for Musical Notation and Computer-Aided Composition'. In: *Computer Music Journal* 39.2 (June 2015), pp. 11–27. DOI: 10.1162/COMJ_a_00296. (Visited on 01/20/2022) (cited on pages 73, 76, 77).

[104] Gerhard Fischer, Daniela Fogli, and Antonio Piccinno. 'Revisiting and Broadening the Meta-Design Framework for End-User Development'. In: *New Perspectives in End-User Development*. Ed. by Fabio Paternò and Volker Wulf. Cham: Springer International Publishing, 2017, pp. 61–97. DOI: 10.1007/978-3-319-60291-2_4. (Visited on 03/07/2022) (cited on page 74).

[105] Georg Hajdu and Nick Didkovsky. 'Maxscore: Recent Developments'. In: *Proceedings of the International Conference on Technologies for Music Notation and Representation*. Montréal, Canada: Zenodo, 2018. DOI: 10.5281/ZENODO.1289703. (Visited on 01/21/2022) (cited on page 76).

[106] Carlos Agon et al., eds. *The OM Composer's Book. Volume1*. Collection Musique/Sciences. Paris: Ircam - Centre Pompidou, Éditions Delatour France, 2006 (cited on page 76).

[107] Antoine Allombert, Myriam Desainte-Catherine, and Gérard Assayag. 'Iscore: A System for Writing Interaction'. In: *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts - DIMEA '08*. Athens, Greece: ACM Press, 2008, p. 360. DOI: 10.1145/1413634.1413699. (Visited on 02/04/2022) (cited on page 76).

[108] Jean-Michaël Celerier. 'Authoring Interactive Media : A Logical & Temporal Approach'. PhD thesis. Université de Bordeaux, 2018 (cited on page 77).

[109] Andrea Agostini, Éric Daubresse, and Daniele Ghisi. 'Cage: A High-Level Library For Real-Time Computer-Aided Composition'. In: *Proceedings of the International Computer Music Conference (ICMC)*. Athens, Greece, 2014. DOI: 10.5281/ZENODO.850533. (Visited on 01/31/2022) (cited on page 77).

[110] Norbert Schnell et al. 'MuBu & Friends - Assembling Tools for Content Based Real-Time Interactive Audio Processing in Max/MSP'. In: *International Computer Music Conference (ICMC)*. Montréal, Canada, 2009 (cited on page 77).

[111] Jean-Étienne Sotty and Fanny Vicens. 'L'accordéon Microtonal XAMP : Gestation, Fabrication et Évolution d'un Nouvel Instrument'. In: *La revue du Conservatoire* 5 (2017) (cited on page 84).

[112] Derek Bailey. *Improvisation: Its Nature and Practice in Music*. New York: Da Capo Press, 1993 (cited on page 93).

[113] Pierre Saint-Germier, Cédric Paternotte, and Clément Canonne. 'Joint Improvisation, Minimalism and Pluralism about Joint Action'. In: *Journal of Social Ontology* 7.1 (July 2021), pp. 97–118. DOI: 10.1515/jso-2020-0068. (Visited on 02/05/2024) (cited on pages 93, 114).

[114] Thor Magnusson. 'Of Epistemic Tools: Musical Instruments as Cognitive Extensions'. In: *Organised Sound* 14.2 (Aug. 2009), pp. 168–176. DOI: 10.1017/S1355771809000272. (Visited on 09/13/2022) (cited on pages 94, 120, 121).

[115] Phil Stone. 'Non-Mathematical Musings on Information Theory and Networked Musical Practice'. In: *Organised Sound* 26.3 (Dec. 2021), pp. 327–332. DOI: 10.1017/S1355771821000418. (Visited on 06/10/2022) (cited on page 95).

[116] Daniel Trueman et al. 'PLOrk: The Princeton Laptop Orchestra, Year 1'. In: *International Computer Music Conference (ICMC '06)*. New Orleans LA USA, 2006 (cited on page 95).

[117] Ge Wang, Georg Essl, and Henri Penttinen. 'Do Mobile Phones Dream of Electric Orchestras?' In: *International Computer Music Conference (ICMC '08)*. Belfast, Ireland, 2008, p. 8 (cited on page 95).

[118] Clément Canonne, Roméo Monteiro, and Joris Rühl. 'L'EMUPO: Une Interface Logicielle Pour l'improvisation Collective'. In: *Journées d'Informatique Musicale*. Saint-Etienne, France, 2011 (cited on page 95).

[119] Julian Rohrhuber et al. 'Purloined Letters and Distributed Persons'. In: *Music in the Global Village Conference*. Műcsarnok Budapest, 2007 (cited on page 96).

[120] Curtis Roads. *Microsound*. Cambridge (Mass.) London: the MIT press, 2004 (cited on page 97).

[121] Diemo Schwarz. 'Concatenative Sound Synthesis: The Early Years'. In: *Journal of New Music Research* 35.1 (Mar. 2006), pp. 3–22. DOI: 10.1080/09298210600696857. (Visited on 05/01/2024) (cited on page 97).

[122] Aymeric Zils and François Pachet. 'Musical Mosaicing'. In: *COST G-6 Conference on Digital Audio Effects (DAFX-01)*. Limerick, Ireland, 2001 (cited on page 97).

[123] Jordi Janer and Maarten de Boer. 'Extending Voice-Driven Synthesis to Audio Mosaicing'. In: *Sound and Music Computing Conference (SMC '08)*. Berlin, Germany, 2008 (cited on page 97).

[124] Meinard Müller. *Information Retrieval for Music and Motion*. New York: Springer, 2007 (cited on page 98).

[125] Matthew Sansom. 'Imaging Music: Abstract Expressionism and Free Improvisation'. In: *Leonardo Music Journal* 11 (Dec. 2001), pp. 29–34. DOI: 10.1162/09611210152780647. (Visited on 07/24/2024) (cited on page 99).

[126] William Hsu. 'Using Timbre in a Computer-Based Improvisation System'. In: *International Computer Music Conference (ICMC)*. Barcelona, Spain, 2005 (cited on page 99).

[127] Noam Mor et al. 'A Universal Music Translation Network'. In: (2018). DOI: 10.48550/ARXIV.1805.07848. (Visited on 02/01/2024) (cited on page 99).

[128] James H. Frey and Andrea Fontana. 'The Group Interview in Social Research'. In: *The Social Science Journal* 28.2 (June 1991), pp. 175–187. DOI: 10.1016/0362-3319(91)90003-M. (Visited on 05/02/2024) (cited on page 105).

[129] Gareth Terry et al. 'Thematic Analysis'. In: *The SAGE Handbook of Qualitative Research in Psychology*. 55 City Road: SAGE Publications Ltd, 2017. DOI: 10.4135/9781526405555 (cited on page 106).

[130] Sergi Jordà. 'Multi-User Instruments: Models, Examples And Promises'. In: *International Conference on New Interfaces for Musical Expression (NIME 2005)*. Vancouver, BC, Canada: Zenodo, June 2005. DOI: 10.5281/ZENODO.1176760. (Visited on 06/27/2023) (cited on pages 117, 118).

[131] Gil Weinberg. 'Voice Networks: The Human Voice as a Creative Medium for Musical Collaboration'. In: *Leonardo Music Journal* 15 (Dec. 2005), pp. 23–26. DOI: 10.1162/lmj.2005.15.1.23. (Visited on 09/29/2022) (cited on page 118).

[132] Norbert Schnell and Benjamin Matuszewski. '88 Fingers'. In: *Web Audio Conference (WAC '17)*. London, UK, 2017 (cited on page 118).

[133] Michael Gurevich, Adnan Marquez-Borbon, and Paul Stapleton. 'Playing with Constraints: Stylistic Variation with a Simple Electronic Instrument'. In: *Computer Music Journal* 36.1 (Mar. 2012), pp. 23–41. DOI: 10.1162/COMJ_a_00103. (Visited on 09/13/2022) (cited on page 119).

[134] Oliver Bown, Alice Eldridge, and Jon McCormack. 'Understanding Interaction in Contemporary Digital Music: From Instruments to Behavioural Objects'. In: *Organised Sound* 14.2 (Aug. 2009), pp. 188–196. DOI: 10.1017/S1355771809000296. (Visited on 01/18/2022) (cited on page 120).

[135] David Borgo and Jeff Kaiser. 'Configurin(g) KaiBorg: Interactivity, Ideology, and Agency in Electro-Acoustic Improvised Music'. In: *Proceedings of the International Conference Beyond the Centres: Musical Avant-Gardes Since 1950*. Thessaloniki, Greece, 2010 (cited on page 120).

[136] Norbert Wiener. *The Human Use of Human Beings*. Houghton Mifflin, 1950 (cited on page 120).

[137] Tom Davis. 'The Feral Cello: A Philosophically Informed Approach to an Actuated Instrument'. In: *International Conference on New Interfaces for Musical Expression (NIME 2017)*. Copenhagen, Denmark, May 2017. (Visited on 09/13/2022) (cited on page 120).

[138] Clément Canonne. 'Élaborer Son Dispositif d'improvisation : Hacking et Lutherie Dans Les Pratiques de l'improvisation Libre'. In: *Volume !* 16 : 1 (Dec. 2019), pp. 61–79. DOI: 10.4000/volume.7266. (Visited on 01/30/2024) (cited on pages 120, 152).

[139] Bertrand Denzler and Jean-Luc Guionnet, eds. *The Practice of Musical Improvisation: Dialogues with Contemporary Musical Improvisers*. 1st. New York: Bloomsbury Academic, 2020 (cited on page 121).

[140] David Borgo. 'Negotiating Freedom: Values and Practices in Contemporary Improvised Music'. In: *Black Music Research Journal* 22.2 (2002), p. 165. DOI: 10.2307/1519955. (Visited on 02/07/2024) (cited on page 121).

[141] David Borgo. *Sync or Swarm: Improvising Music in a Complex Age*. New York: Continuum, 2005 (cited on page 122).

[142] Clément Canonne and Nicolas B Garnier. 'Cognition and Segmentation In Collective Free Improvisation: An Exploratory Study'. In: *International Conference on Music Perception and Cognition*. Thessaloniki, Greece: Unpublished, 2012. DOI: 10.13140/2.1.3534.9445. (Visited on 02/07/2024) (cited on page 122).

[143] Clément Canonne. *La Lutherie Des Improvisateurs*. [Unpublished manuscript], 2021 (cited on page 123).

[144] Thor Magnusson et al. 'The Organium: A Library of Technical Elements for Improvisatory Design Thinking'. In: *Audio Mostly 2024 - Explorations in Sonic Cultures*. Milan Italy: ACM, Sept. 2024, pp. 198–208. DOI: 10.1145/3678299.3678319. (Visited on 10/02/2024) (cited on page 153).

[145] Susan Leigh Star and James R. Griesemer. 'Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39'. In: *Social Studies of Science* 19.3 (Aug. 1989), pp. 387–420. DOI: 10.1177/030631289019003001. (Visited on 10/02/2024) (cited on page 153).

[146] Alvin Lucier, Gisela Gronemeyer, and Reinhard Oehlschlägel. *Reflections: interviews, scores, writings: Interviews, Notationen, Text 1965 - 1994 = Reflexionen*. 2., erw. und überarb. Aufl. Edition Musiktexte 003. Köln: MusikTexte, 2005 (cited on pages 155, 156, 164).

[147] Christopher Burns and Matthew Burtner. 'Recursive Audio Systems: Acoustic Feedback in Composition'. In: *Leonardo Electronic Almanac* 12.2 (2004) (cited on page 164).

[148] Gianluca Elia and Dan Overholt. 'Squidback: A Decentralized Generative Experience, Based on Audio Feedback from a Web Application Distributed to the Audience'. In: *Sound and Music Computing Conference*. 2021 (cited on page 164).

[149] Agostino Di Scipio. ''Sound Is the Interface': From *Interactive* to *Ecosystemic* Signal Processing'. In: *Organised Sound* 8.3 (Dec. 2003), pp. 269–277. DOI: 10.1017/S1355771803000244. (Visited on 11/25/2021) (cited on page 165).

[150] Nathan Villicana-Shaw et al. 'Legatus: Design and Exhibition of Loudspeaker-Based, Environmentally-Reactive, Soundscape Augmentation Artifacts in Outdoor Natural Environments'. In: *International Conference on New Interfaces for Musical Expression (NIME)*. Mexico City, Mexico, 2023 (cited on page 167).