

Towards Networked Hyperorgan Performance: A Software Architecture for Remote Musical Interaction

Hannes Ritschel 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
hannes.ritschel@uni-a.de

Thomas Kiderle 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
thomas.kiderle@uni-a.de

Johanna Kuch 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
johanna.kuch@uni-a.de

Anan Schütt 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
anan.schuett@uni-a.de

Elisabeth André 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
elisabeth.andre@uni-a.de

Ruben Schlagowski 

Human-Centered Artificial Intelligence
University of Augsburg
Augsburg, Germany
ruben.schlagowski@uni-a.de

Abstract—Hyperorgans combine pipe-organ traditions with digital capabilities such as MIDI and OSC, but their heterogeneous interfaces do not inherently support networked, real-time collaborative performance of organists. This paper proposes a software architecture and browser-based user interface that abstract instrument-specific controls to support interoperable, geographically distributed performance within the Global Hyperorgan vision. We describe the design requirements and a hybrid client-server and peer-to-peer architecture using MIDI for instrument control, WebSockets for coordination, and WebRTC for musical events and audio monitoring.

Index Terms—pipe organ, hyperorgan, MIDI, remote musical interaction, networked music performance

I. INTRODUCTION

The pipe organ is an instrument with a long-standing tradition. Throughout its history, numerous technical innovations have continuously expanded its musical and technical capabilities. Since the late twentieth century, some organs have additionally been equipped with digital interfaces such as MIDI (Musical Instrument Digital Interface), allowing them to be played not only manually from the console but also through digital control. In 2011, the term “hyperorgan” was first defined as an “organ with extended capabilities that seamlessly blend the electronic and acoustic worlds” [1, p. 3], including support for MIDI and OSC (Open Sound Control) [1]. Later, the concept of the “Global Hyperorgan” was formulated as an “intercontinental, creative space for acoustic musicking”, involving pipe organs that are “networked for real-time, geographically distant performance” [2].

This vision is closely related to the research fields of Networked Music Performance (NMP) [3], [4] and the Internet of Musical Things (IoMusT) [5], [6]. NMP investigates low-latency communication, synchronization, and distributed musi-

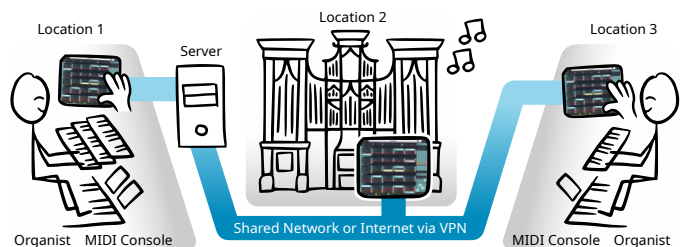


Fig. 1. Idea of the proposed system, enabling organists to collaboratively perform on hyperorgans across geographically distributed locations. Organists perform from local MIDI organ consoles or keyboards. Remote organ controls are available via the local console and/or an organ-agnostic touch interface.

cal interaction, while IoMusT extends these concepts towards interconnected musical devices, performers, and audiences.

Important technologies include the encoding and transport of music data in networks and over the internet. RTP-MIDI [7] standardizes the transport of MIDI command streams over IP networks and provides mechanisms for timestamps and recovery from packet loss. ipMIDI¹ instead distributes MIDI data over multicast UDP to applications listening on the corresponding virtual port within the same local subnet; it requires no client-server setup and does not route packets over the Internet. Established NMP systems such as JackTrip [8], LoLa², SoundJack³, and Jamulus⁴ focus primarily on low-latency audio or audiovisual transmission [4]. Browser-based NMP systems use WebRTC to reduce setup effort and support low-latency musical interaction [9], [10].

¹<https://www.nerds.de/en/ipmidi.html>

²<https://lola.conts.it/>

³<https://www.soundjack.eu/>

⁴<https://jamulus.app/>

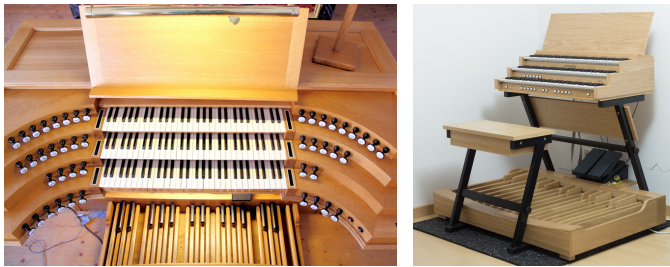


Fig. 2. Left: Traditional pipe-organ console with three manuals, pedalboard, stops, and couplers. Right: Digital MIDI console with three manuals, pedalboard, and two expression pedals.

In the context of hyperorgans the software Organ Sync⁵ routes MIDI messages from local input devices or music software to networked organs. However, the software does not expose organ-specific registration controls such as stops and couplers through its interface. Consequently, it is primarily suited to transmitting note events.

Existing Global Hyperorgan performances have often relied on software environments such as SuperCollider, prepared or algorithmically generated MIDI sequences, or application-specific control interfaces. While these approaches successfully demonstrate remote control of pipe organs, they rarely support the interaction model in which organists perform manually from a traditional organ console.

To the best of our knowledge, no existing system yet enables organists to perform on heterogeneous hyperorgans through a common web-based interface while abstracting instrument-specific MIDI implementations. As illustrated in Figure 1, we present a work-in-progress software architecture through which organists can connect to remote hyperorgans, perform from MIDI organ consoles or keyboards, and control remote instruments with unified controls.

II. PIPE ORGANS AND MIDI

Most digital interfaces in pipe organs use messages defined in the MIDI 1.0 standard from 1983. For each of 16 channels, MIDI, among other message types, defines *Note On/Off* messages (for pressing and releasing keys), *Control Change* messages (for setting parameters), *Program Change* messages (for changing the instrument of a channel), *System Exclusive* messages (for arbitrary, non-standardized data) and more. Since the MIDI standard primarily addresses electronic music production, there is no direct mapping from pipe-organ-specific features to MIDI messages, which is why their implementation varies from instrument to instrument. Only in recent years has a MIDI standard for typical pipe-organ features been proposed by a German electronics supplier⁶ and has since been implemented by several organ builders in new instruments. In the following, we outline typical features of pipe organs (see also Figure 2) together with message types that we found in real-world organ MIDI interfaces.

⁵<https://github.com/woutersnoei/organ-sync>

⁶<https://orgelpunkt.com/>



Fig. 3. Thumb pistons on the digital MIDI console from Figure 2 for saving, recalling, cycling through, and clearing combinations.

A. Manuals and Pedals

An organ has one or more manuals (keyboards) that are played with the hands and pedals that are played with the feet. Both manuals and pedals are used to play notes. As with keyboards and synthesizers, Note On/Off messages are the standard for playing notes via MIDI, the MIDI channel identifies the individual manual/pedals.

B. Stops and Couplers

While the number of keys on an organ is smaller than the number of keys on a piano, the organ can produce different timbres (called stops or registers) with different characteristics (for example, resembling flutes, trumpets, strings, or other instrumental families) and pitch levels (indicated by footage designations, such as 8' for normal pitch, 4' sounds one octave higher, 16' sounds one octave lower). Manuals and pedals represent different divisions each, to which a unique set of stops is assigned. With couplers, one can simultaneously play multiple divisions from a single manual/pedals to combine their sounds. The process of selecting and combining stops is called “registration”. Organists do this as part of their concert preparations. Organs with a MIDI interface, for example, use Note On/Off, Control Change, or Program Change messages for turning on/off/toggling the state of stops and couplers.

C. Expressions

Most organs have one (or more) swell divisions. All stops of such a division are enclosed in a box with swell shades that can be opened or closed with an expression pedal in order to increase or decrease the loudness of the corresponding division. Organ MIDI interfaces use for example Control Change messages to open and close the swell shades.

D. Combination Action

The organist can save combinations to memory. Such presets of stops/couplers can be recalled later either in sequential order or by index. Typically, one operates the combination action with thumb pistons below the first manual of the organ console (see Figure 3) and/or with foot pistons above the pedals. In some instruments with MIDI interface, the combination action can be controlled, for example, using Note On/Off messages.

III. SYSTEM ARCHITECTURE

A. Design Requirements

Pipe organs are highly individual instruments. Besides differences in size, technical construction, tonal concept, and available features, their digital control interfaces also vary considerably. While MIDI provides a standardized communication

TABLE I
COMPARISON WITH RELATED NETWORKED MUSIC AND ORGAN-CONTROL TECHNOLOGIES AND SOFTWARE.

Approach	Primary focus	Browser-based	Organ profiles	Organ-oriented UI
RTP-MIDI	Transport of MIDI over IP	No	No	No
ipMIDI	Multicast MIDI within a local subnet	No	No	No
JackTrip, LoLa, SoundJack, Jamulus	Low-latency audio or audiovisual NMP	Varies	No	No
Organ Sync	Networked organ MIDI routing	No	No	No registration
Proposed architecture	Interoperable hyperorgan control and monitoring	Yes	Yes	Yes

protocol, there is no common convention for representing organ-specific functionality. Consequently, the MIDI interfaces of different organs often expose only a subset of the functionality described in section II and employ instrument-specific message mappings, sometimes including proprietary System Exclusive (SysEx) messages. Furthermore, some advanced organ functions cannot be represented using MIDI alone. For this reason, certain systems additionally provide OSC interfaces to expose such features.⁷

From these considerations, we derive the following requirements. *R1: Instrument abstraction* separates common organ functions from instrument-specific MIDI messages. *R2: Input abstraction* maps heterogeneous MIDI keyboards and consoles to the manuals, pedals, and expressions of remote instruments. *R3: Low-latency transmission* avoids unnecessary server traversal for time-critical performance events. *R4: Shared registration state* coordinates stops, couplers, and combinations across participants and instruments. *R5: Browser accessibility* minimizes platform-specific installation and configuration. *R6: Controlled deployment* assumes that known participants, the server, and instruments share a trusted local network or VPN; operation over an untrusted public network is outside the scope of this work.

B. Server, Clients and Features

The proposed architecture for networked hyperorgan performance (see Figure 4) follows a hybrid client-server and peer-to-peer model. It supports a common subset of features found on many pipe organs: playing the manuals/pedals, controlling stops/couplers, expressions, and combination action. While a central server coordinates the session and maintains the shared application state, clients exchange time-critical musical data directly via peer-to-peer connections. Table II summarizes the responsibilities and intended communication paths.

1) *Server*: The server hosts the HTML5-based user interface that participants open in a web browser to join a networked session. Communication between the server and clients is realized using TCP-based WebSocket connections, which are used for state synchronization, configuration management, and WebRTC signaling.

The server stores all instrument and MIDI input device definitions. These include the individual organs, such as available stops, couplers, expressions, and the corresponding

⁷Although OSC is a standardized communication protocol, there is likewise no common convention for representing organ-specific functionality. Since most organs offering OSC also provide a MIDI interface for their core functionality, this work focuses on MIDI.

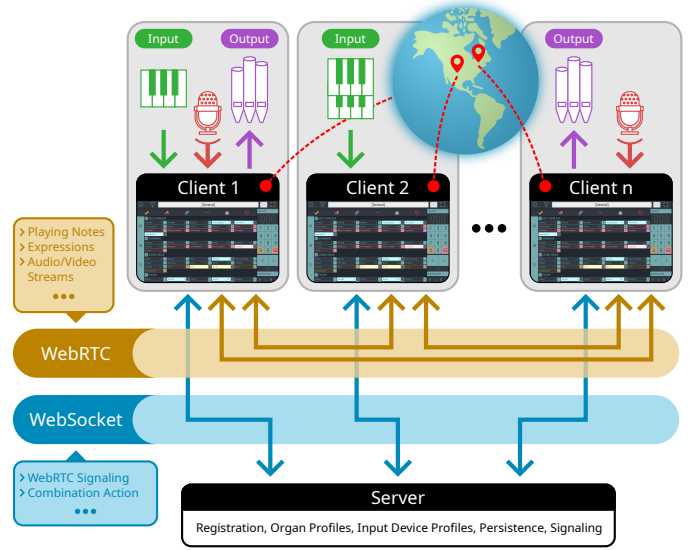


Fig. 4. Hybrid architecture using peer-to-peer WebRTC for time-critical events and monitoring audio, and WebSockets for server-coordinated communication. All components share a trusted network or VPN.

TABLE II
COMMUNICATION IN THE PROPOSED ARCHITECTURE.

Data	Communication Path	Requirement
Notes	WebRTC DataChannel	Low latency; state recovery
Expressions	WebRTC DataChannel	Low latency; latest value
Stops/couplers	WebSocket via server	Reliable shared state
Combinations	WebSocket via server	Reliable, coordinated change
Configuration	WebSocket via server	Reliable and ordered
Audio monitoring	WebRTC media	Low latency; loss tolerant

MIDI messages required to operate them. Likewise, MIDI input devices are described by their manuals, MIDI channel assignments, and device-specific control mappings, including combination action buttons. These definitions only have to be created once and can subsequently be reused by the clients.

2) *Clients*: Each participant runs a browser-based client that can simultaneously act as a MIDI input endpoint and an instrument output endpoint. MIDI input devices, such as keyboards or digital organ consoles, can be connected to the client to generate musical events. Likewise, pipe organs may be connected via MIDI output, allowing remote performers to play them using MIDI devices connected to their local clients.

To minimize latency, our architecture uses direct peer-to-peer WebRTC connections between participants. WebRTC DataChannels are intended for the transmission of time-critical musical data, such as note events and expression states.

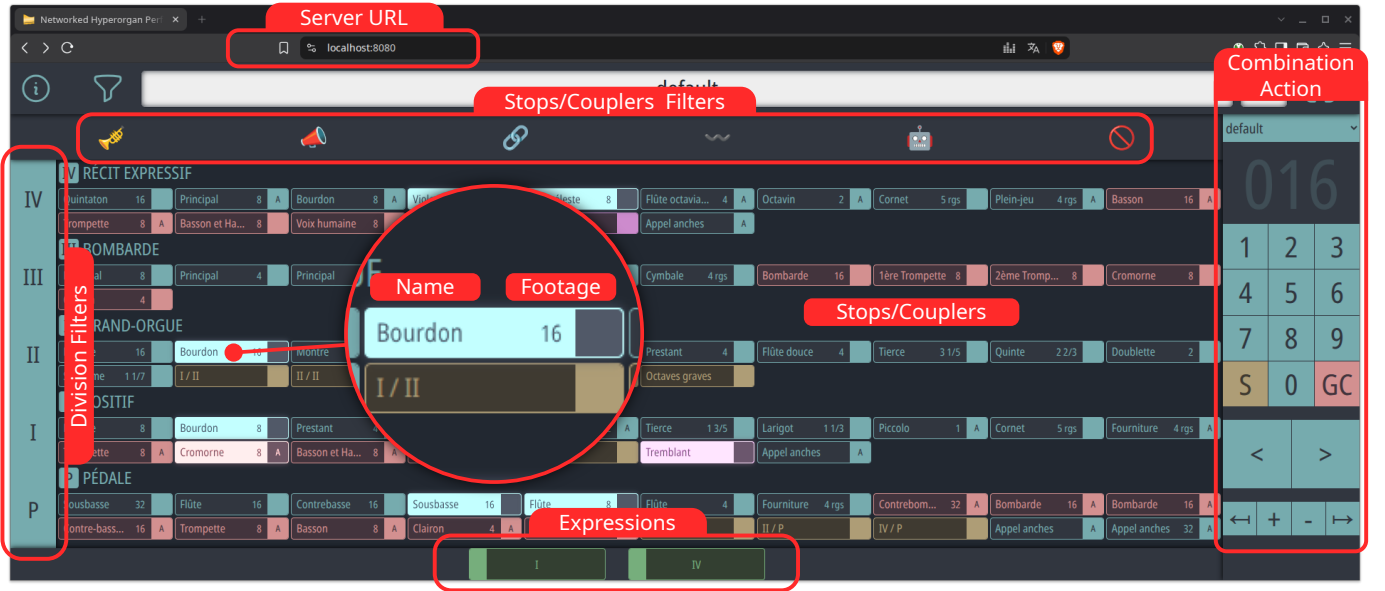


Fig. 5. HTML5 interface for registration, expression control, combination actions, and MIDI input mapping (MIDI mapping is not visible in this screenshot).

Because the envisaged Global Hyperorgan performances are expected to involve only a small number of clients, direct communication avoids routing these messages through the central server while reducing latency and server load.

In parallel, each client maintains a persistent, TCP-based WebSocket connection to the server for server-coordinated communication, including application state synchronization, combination action, and session management.

3) *Combination Action*: Since many organs do not expose their internal combination action via MIDI, we implement a server-side combination action. It maintains the canonical state of all stops and couplers. When a stored combination is recalled, the server computes the difference between the current and requested registration and determines the required stop and coupler changes for each connected instrument. Only these incremental updates are transmitted to the corresponding clients, which send the necessary messages to the instruments. This approach enables synchronized registration changes across multiple organs and clients.

4) *Audio Monitoring*: Participants must also hear the instruments played at the remote locations. Our architecture establishes WebRTC audio streams between all clients. These streams allow participants to monitor the audio produced by every other location. These streams are not intended to provide high-fidelity audio, but rather a quick monitoring solution based on the low-latency Opus codec. An approach for higher-quality streaming via WebRTC has been investigated in [10].

C. User Interface

The HTML5 interface in Figure 5 follows the mental model of a traditional organ console, grouping controls by division. Browser-based delivery provides cross-platform access with minimal local setup to the essential features from section II:

1) *Stops and Couplers*: For registration, the main area in the center lists all stops, grouped by division (manuals, pedals). Users can filter different stop types using the controls at the top and show or hide individual divisions on the left. Each button displays the stop name and footage. Button colors distinguish ordinary stops (blue), reeds (red), couplers (yellow), and tremulants (magenta) for better orientation.

2) *Expressions*: The state of the swell divisions is visualized and can be controlled with sliders below the stop list.

3) *Combination Action*: Users can insert, delete, save and recall presets of stops and couplers sequentially or by index. The MIDI mapping is part of the preset.

4) *Input Mapping*: When performing with a MIDI keyboard or console (see Figure 2), users can assign the local keyboards and expression pedals to the remote instrument's manuals, pedals and expressions. This assignment is particularly useful when the number of keyboards or expression pedals does not match that of the remote instrument.

IV. CONCLUSION

This paper proposed a software architecture for interoperable networked hyperorgan performance. It combines an organ-agnostic browser interface with centralized coordination and peer-to-peer paths for musical events and audio monitoring. Future work will implement and evaluate the concept and investigate cumulative network, instrument, acoustic, and monitoring delays, as well as human feedback with regards to usability and practical, musical use.

REFERENCES

- [1] R. Harlow. "Recent organ design innovations and the 21st-century "Hyperorgan"." (May 2011), [Online]. Available: [https://huygens-fokker.org/docs/Harlow%20-%20Recent%20Organ%20Design%20Innovations%](https://huygens-fokker.org/docs/Harlow%20-%20Recent%20Organ%20Design%20Innovations%20-%202011.pdf)

20and%20the%2021st%20Century%20Hyperorgan.pdf (visited on 09/24/2026).

- [2] R. Harlow, M. Petersson, R. Ek, F. Visi, and S. Östersjö, "Global Hyperorgan: A platform for telematic musicking and research," *International Conference on New Interfaces for Musical Expression*, Apr. 29, 2021. DOI: 10.21428/92fbeb44.d4146b2d.
- [3] A. B. Renaud, A. Carôt, and P. Rebelo, "Networked music performance: State of the art," in *Proceedings of the AES 30th International Conference*, Finland Saariselkä, vol. 4, 2007, pp. 1–7.
- [4] C. Rottondi, C. Chafe, C. Allocchio, and A. Sarti, "An Overview on Networked Music Performance Technologies," *IEEE Access*, vol. 4, pp. 8823–8843, 2016, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2016.2628440.
- [5] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet, "Internet of Musical Things: Vision and Challenges," *IEEE Access*, vol. 6, pp. 61994–62017, 2018, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2872625.
- [6] L. Turchet and C. Rottondi, "On the relation between the fields of Networked Music Performances, Ubiquitous Music, and Internet of Musical Things," *Personal and Ubiquitous Computing*, vol. 27, no. 5, pp. 1783–1792, Oct. 1, 2023, ISSN: 1617-4917. DOI: 10.1007/s00779-022-01691-z.
- [7] J. Lazzaro and J. Wawrzynek, "RTP Payload Format for MIDI," Internet Engineering Task Force, RFC 6295, Jun. 2011. DOI: 10.17487/RFC6295. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc6295.html> (visited on 09/24/2026).
- [8] J.-P. Cáceres and C. Chafe, "Jacktrip: Under the hood of an engine for network audio," *Journal of New Music Research*, vol. 39, no. 3, pp. 183–187, 2010.
- [9] M. Sacchetto, P. Gastaldi, C. Chafe, C. Rottondi, and A. Servetti, "Web-Based Networked Music Performances via WebRTC: A Low-Latency PCM Audio Solution," *Journal of the Audio Engineering Society*, vol. 70, no. 11, pp. 926–937, Dec. 12, 2022, ISSN: 15494950. DOI: 10.17743/jaes.2022.0021.
- [10] M. Sacchetto, L. Severi, C. Rottondi, *et al.*, "Pristine Quality Networked Music Performance System for the Web," in *Proceedings of the 10th Convention of the European Acoustics Association Forum Acusticum 2023*, Turin, Italy: European Acoustics Association, Jan. 17, 2024, pp. 2957–2964, ISBN: 978-88-88942-67-4. DOI: 10.61782/fa.2023.0556.