Hear-Here

a choreographed peer-to-peer network for live participation on the radio

August Black
ATLAS Institute, University of Colorado Boulder
Boulder, Colorado 80309-0320
mail@august.black

ABSTRACT

This paper describes a software system and live audio event called "Hear-Here" that links live microphone input from multiple users together in an FM radio broadcast. The system connects users in a browser-based peer-to-peer network using WebRTC whereby each user, taking turns, is able to contribute 2 seconds of audio at a time, ad infinitum. The paper provides a description and evaluation of the system and radio event along with background motivation and related work.

CCS CONCEPTS

• **Human-centered computing** → **Participatory design**; **Collaborative content creation**; *Mobile devices*;

KEYWORDS

peer-to-peer audio, WebRTC, participatory radio

ACM Reference format:
August Black. 2017. Hear-Here. In *Proceedings of ACM Audio Mostly conference, London, UK, August 2017 (Audio Mostly'17),* 6 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Drawing from and contributing to a long tradition of radio and transmission art[7][9], "Hear-Here" creates a radiophonic space for direct interaction on FM radio among participants.

The peer-to-peer network of the "Hear-Here" system links the microphone input from multiple users online into a single sonic event that is choreographed according to a predefined game-like sequence. The system activates the microphone for each participant in turn at 2 second intervals. Each participant is free to use their mic as they choose, playing instruments, singing, speaking, feeding back the signal, etc. A master server collects the input from participants to broadcast it live on FM radio and stream it online where each user, near and far, can hear the results within some threshold of latency.

The system operates over WebRTC[1] and consists of three parts: a client side that is run by each participant, a single server side that collects the transmitted audio in a central location for broadcast, and a signal server that manages the handshake process and other control signals. Both the client and the server run in a standard web browser (Firefox, Chrome).

The initial broadcast of "Hear-Here" took place on WGXC[2] in Acra and Hudson, New York on June 27th, 2015.

2 BACKGROUND

Radio is arguably the first and ultimate futuristic medium. It not only gives volume and voice to an imaginary space devoid of air, it connects disparate geographies, cultures, individuals, and machines over distance. The birth of radio inspired a generation of artists and engineers alike at the turn of the 20th century. The utopian sentiment that radio initially delivered is best indicated in Khlebnikov's 1921 "Radio of the Future"[10], where radio is fetishized as "...the central tree of our consciousness" where it "...will inaugurate new ways to cope with our endless undertakings and will unite all mankind."

The internet, and more precisely the WWW, also had an initial wave of utopian and anarchistic fervor that has now, like with radio, subsided.

Radio is ultimately defined in hardware where the transmitter and receiver are physical devices. The internet is mostly an abstraction that can communicate over differing physical transport. Radio is, in fact, just one physical layer of internet. Where the internet might provide multi-cast range and on-demand programming, radio provides uni-cast concentration and channel-based programming. The salient

Audio Mostly'17, London, UK

DOI: 10.475/123_4

[1] https://webrtc.org/
[2] https://wavefarm.org/wgxc

difference is that AM/FM radio is an immediate synchronous medium with little to no latency.

"Hear-Here" is an attempt to tease out a new kind of futuristic inspired transmission space from the overlap of these two media. It emphasizes experimentalism, open networks, and the joy of live interconnectedness - whereby individuals connect with one another through some system simultaneously.

3 RELATED WORK

The "Hear-Here" performance and radio event draws from multiple overlapping histories of participatory and networked audio performance, telecommunication art, and radio art. It also touches upon a number of research areas such as browser-based audio production, peer-to-peer control networks, audio streaming, mobile platforms for music making, and systems for audience participation.

Some examples of prior work include collaborative software environments. GroupLoop[12] is a similar browser-based WebRTC application that posits advanced users in a peer-to-peer network for the purpose of exploring microphone feedback. Cabrera describes an OSC protocol for sharing control UI and data in a serverless peer-to-peer network.[4] Roberts et al. describe a system for distributed control over networks via mobile devices[13] as well as interfaces for in-browser audio production and collaboration.[14] Wakefield et al. have produced browser-based infrastructure for distributed live coding environments[15]

Other influential publications emphasize audience participation, musical interaction, and/or mobile interactions. The Daisyphone[3] allows players to create and edit short shared loops of music for group music improvisation. Dialtones[11] is a project that triggers the audience's phones via a custom software dialer. Public Sound Objects[1] is a web-based shared musical space, which has been an experimental framework to implement and test different approaches for on-line music communication. JamSpace [8] is a hardware and software interface for real-time rhythmic collaboration from isolated locations. GraphTheory[5] created a browser interface that allows users to navigate among short, looping musical fragments for composing open-form works.

Beyond related academic conferences, this project finds influence in tele-communication projects such as Max Neuhaus's "Public Supply I"[3], Robert Adrian's The World in 24 Hours[6] , Horizontal Radio[4], the many projects by ORF Kunstradio[5] as well as early net-streaming experiments on the XCHANGE network[6]. Userradio[2] is another software-based streaming audio instrument for the radio that provides precursory development.

"Hear-Here" is different from prior research in a number of subtle, but important ways. First and foremost is the focus of producing "space" rather than the creation of acoustic phenomena. The output is on FM radio and emphasizes the simultaneous connectivity over distance rather than co-located musical expression. Secondly, the intended audience and participation of "Hear-Here" is more general and direct than works cited above. Radio is a real gallery without walls. People listen to radio while driving cars, making dinner, playing with kids, etc. Part of the fun of the system is its simple microphone-driven engagement and low barrier to entry. As such, the importance of musical structure and rhythm, signal processing techniques are mostly absent, while direct user input with voice and available impromptu instrumentation is foregrounded.

4 SYSTEM DESIGN

"Hear-Here" uses WebRTC, WebAudioApi, and websockets to connect users into one broadcast event. It has 3 main components: the client, the signal server, and the master control.

Both the client and the central master control visit a web site with their browsers. The socket server signals to each participant's browser with RTC connection information to link the client's audio output directly to the master control's audio input. Once the RTC connection is established between peers, the socket signal server is used only to transmit the 2 second pulse so that the system state (ie. which participant is currently live and which are waiting their turn) is synced among participants. All audio is routed from participant directly to the master control, browser to browser.

Participants can join and leave the event without disrupting one another. The master controller can also join and leave the event. When the master disconnects, audio is no longer streamed or broadcast to the participants. However, when the master control reconnects, the signal server renegotiates the RTC information and the system is able to recover within seconds.

The signal server

The signal server is a node.js application using the express.js and socket.io libraries. Essentially, it handles the initial handshake between multiple participants and the single master control.

It has two HTTP end points, /index.html and /master for the clients and master control, respectively. The javascript included in both will open a websocket back to the signal

[3] http://www.medienkunstnetz.de/works/public-supply-i/
[4] http://www.kunstradio.at/HORRAD/horrad.html
[5] http://kunstradio.at
[6] http://www.net-art.org/xchange

server. Via the websockets, the signal server manages the Interactive Connectivity Establishment (ICE) for the webRTC communication. Once the ICE is handed off between the client and server its only job is to reflect the system state (a simple JSON data object) driven by the master to the clients every 2 seconds.

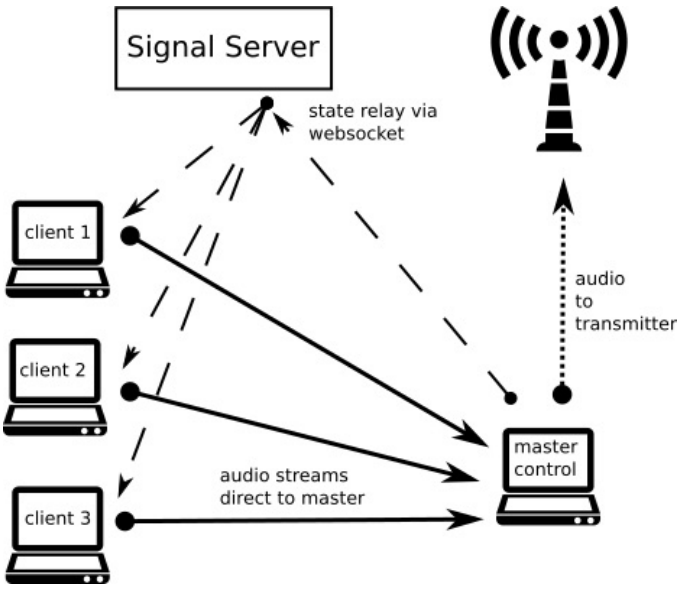

Figure 1: Audio streams directly from client browser to master control browser.

In this case, the peer-to-peer network that the signal server manages has a directionality - multiple clients establish a connection with master control and stream audio directly to it, not to each other.

The client

The client app is a single HTML and javascript webpage that does two main things. It sends audio to the master control and presents the user with a visual interface that describes event status. The interface shows the user the number of other participants currently involved (and as they come and go), their position in the line-up, as well as network outages.

Each participant views the interface where they are represented as the center circle on screen. The other participants are represented as the other smaller external circles. When it is your turn as a participant, the large center circle fades from red to green (Figure 2). It then fades back to red when your mic is turned off (Figure 3). The same goes for the representation of the other users on screen. If you go offline, the large central circle will turn grey (Figure 4). If another user disconnects, their circle will first become grey and will eventually disappear from screen if disconnected long enough.

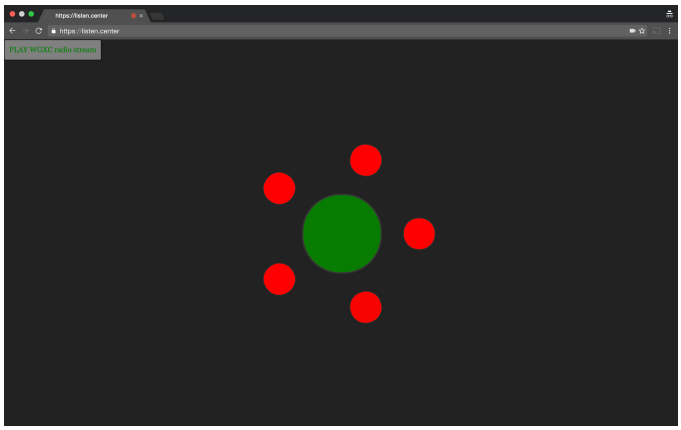

Figure 2: When the center circle is green it's your turn on the mic. Have a go.

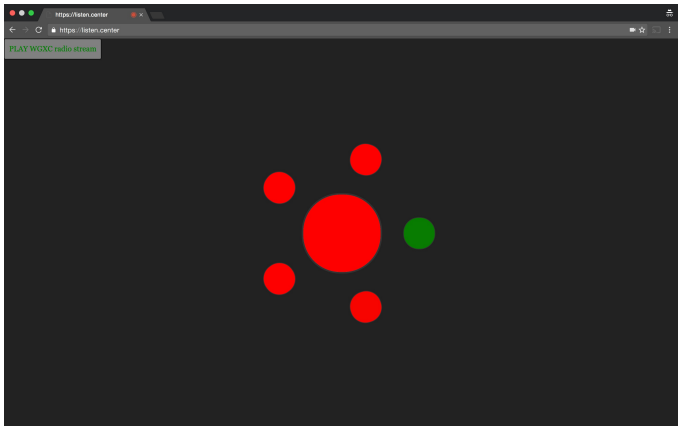

Figure 3: When the center circle is red it's another persons turn on the mic. The green circle shows who is next.

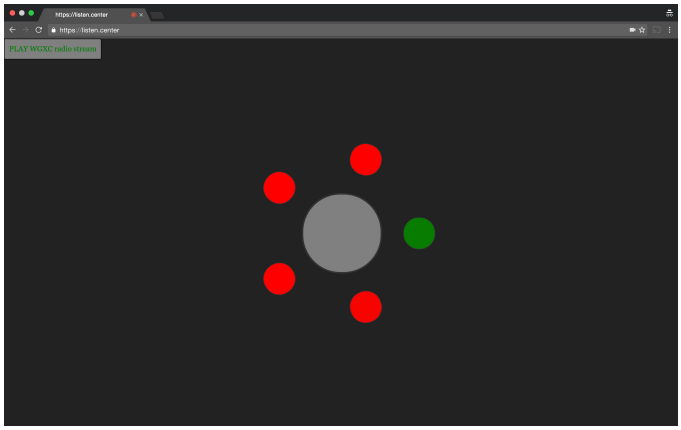

Figure 4: When the center circle is grey, it means you lost your connection.

The design of the client interface is meant to be clear and simple, something that a user could understand by watching for a few seconds and without a textual description.

The master control

The master application is a single HTML and javascript webpage that does two main things. It collects the audio input via WebRTC from the multiple clients and drives the 2 second pulse to synchronize the system.

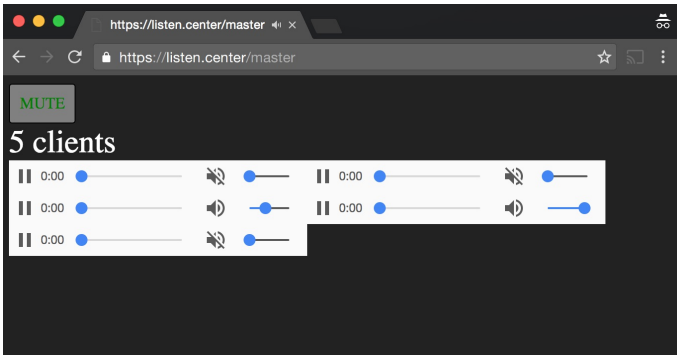

Figure 5: The master interface showing 5 participants

Figure 5 shows the master interface with 5 connected clients. At 2 second intervals, it will a) update it's internal state designating which participant is currently live; in this case 4 users are off, 1 is live b) reflect the state information to all users through the signal server, and c) fade out the last user's audio input while fading in the new users' audio input. Additionally, it has a convenience mute button to fade out all inputs and stop the 2 second interval driver.

In an actual setup of the system, the user would just pull up the master interface in their laptop and connect the audio from the laptop to the radio's audio mixer console.

5 THE BROADCAST

The live broadcast of this work happened over a 2 hour period on June 27, 2015. Tom Roe and I sat in the WaveFarm studio for WGXC in upstate New York with the master control hooked up the radio mixing console. We perceived an average of 20-25 simultaneous participants, with participants coming and going as time drew on.

We also participated as clients on our mobile phones, feeding the system with our voices and other musical and non-musical impulses. Occasionally, we'd talk over the running audio to inform users about the online interface and how to interact.

The result of the FM broadcast happened somewhat serendipitously, unfolding without any planning for content. While mostly full of cacophony and perceivable system feedback, there were occasions of intentional communication between participants. At one point users tried to count with each other, the next participant incrementing the count. At another time the participants listed off the names of cheeses. Some participants tried to initiate songs or read poetry.

Figure 6: An inquisitive participant with her kitchen sounds

The 2 second intervals of user input was also a simple and effective constraint. It gave the broadcast a regular cadence and pulse but was far from monotonic. Even if one user was playing a single track of pre-recorded material, it could be 20-30 seconds (with just 10-15 participants) before the system picked up that user's input again. Not unlike the Kuleshov effect in film[7], listeners can derive more meaning from the interaction of the sequential audio segments than from any individual input. At the same time, the 2 second limitation made it easy for users to participate without worry of filling a longer time period.

Overall, I was somewhat surprised and intrigued by the variety of sounds and by the fact that so many children participated. The entire two hours can be heard at https://wavefarm.org/archive/xxjc5w.

6 EVALUATION AND FUTURE WORK

There are a couple engineering challenges and situational circumstances that make this work somewhat particular and unique. I'll outline the circumstances first and challenges second.

The ubiquity of mobile phones and browsers with microphone input coupled with an almost universal network accessibly make this system very frictionless and responsive. Users only need to load a web page and start making noise.

[7] http://www.elementsofcinema.com/editing/kuleshov-effect.html

The system is also fairly fast and robust. The page load is fast due to the fact that most of the functionality is provided by the browser (WebRTC networking, audio encoding and streaming, socket connectivity etc). Users need not login or configure system or application parameters. They only need to click "okay" to allow the browser to access the device's microphone. The socket connectivity also provides a robustness and fluidity not seen in former web-based applications that might have needed to poll servers for ongoing connection information.

This is not to say the system has no challenges. One particular challenge is the inherent feedback that happens when the audio input and output are so closely situated on a users laptop or mobile phone. On one hand, this can be counted as a stylistic attribute that gives color to the system. Additionally, for quiet users it provides a way to contribute to the system without making sound themselves. On the other hand, it can also confuse users that might be encountering the system for the first time and expect a clear signal to align themselves with how the system "should" function.

Another challenge is the clear communication of the system state. I believe the interface I developed does a decent job, but could be fine tuned to better indicate latency, network congestion, and the labeling of other users.

Yet another challenge is the inherent latency. For users within the terrestrial FM broadcast range, the latency is fairly minimal and practically imperceptible. For users that are not within FM broadcast range and listen to the online streamed version, they must deal with the additional buffering of an encoded mp3 return audio. Once understood as a part of the system, however, the latency is no longer an issue. So long as the user actuates their microphone when their circle turns green, they can then listen for it in the mix. In some ways the latency is even advantageous since it can keep the feedback from growing too fast too soon. I believe I can improve on this in future versions by delivering a low-quality WebRTC audio stream for online participants instead of the mp3.

The biggest challenge for this system is one that I unfortunately cannot solve. That is, all Apple mobile devices do not support the getUserMedia API that allows the user to access their microphone from the browser. The system will only work in Chrome or Firefox browsers on desktops (Apple included) and non-Apple mobile phones. Since Apple is unlikely to allow its mobile devices access to the microphone through the browser, one future development I might pursue is the creation of an iphone application for user with these disabled devices.

In future iterations I would like to improve upon the interface and think of ways to compensate for - or better represent - latency and for the lack of support on iPhones. I would also like to introduce a slider to the master interface so that driver of this system can change the user input interval to be as little as a few milliseconds or as much as 10 seconds, essentially varying the perceived audio from the scale of granular synthesis into the order of narration. I would also like to test the system in a larger radio context with more participants, perhaps setting the system live for shorter segments at scheduled times so that users may develop some experience with how the it behaves. With the correct interface, I believe the system can scale to multiple hundreds of participants. One interesting aspect to explore regarding scalability would be varying scheduling routines - instead of round-robin, perhaps queuing or overlapping users would be appropriate for user numbers in the hundreds or thousands.

7 CONCLUSION

"Hear-Here" is a robust and flexible system for allowing direct user input via browser-to-browser audio streaming on FM radio. Its novelty lies in its simplicity and its direct overlapping of radio and internet methodologies. I developed the initial prototype for "Hear-Here" in 2015 at a 10 day Wave Farm residency. The code can be found at https://github.com/augustblack/hearhere

ACKNOWLEDGMENTS

The initial development of "Hear-Here" was generously supported by Wave Farm in Acra, NY. I also thank Laura Devendorf for reading drafts of this paper.

REFERENCES

[1] Alvaro Barbosa, Jorge Cardoso, and Günter Geiger. 2005. Network Latency Adaptive Tempo in the Public Sound Objects System. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Vancouver, BC, Canada, 184–187. http://www.nime.org/proceedings/2005/nime2005_184.pdf

[2] August Black. 2004. Userradio. In *Proceedings of the 12th annual ACM international conference on Multimedia*. Association for Computing Machinery, 186–187.

[3] Nick Bryan-Kinns and Patrick G. Healey. 2004. Daisyphone: Support for Remote Music Collaboration. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Hamamatsu, Japan, 27–30. http://www.nime.org/proceedings/2004/nime2004_027.pdf

[4] Andres Cabrera. 2015. Serverless and Peer-to-peer distributed interfaces for musical control. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 355–358. http://www.nime.org/proceedings/2015/nime2015_351.pdf

[5] Jason Freeman. 2007. Graph Theory : Interfacing Audiences Into the Compositional Process. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. New York City, NY, United States, 260–263. http://www.nime.org/proceedings/2007/nime2007_260.pdf

[6] Heidi Grundmann (Ed.). 1984. *Art Telecommunicaton*. Western Front Publication.

[7] Heidi Grundmann, Elisabeth Zimmermann, Reinhard Braun, Dieter Daniels, Andreas Hirsch, and Anne Thurmann-Jajes (Eds.). 2008. *Re-Inventing Radio: Aspects of Radio as Art*. Revolver, Frankfurt, Germany.

[8] Michael Gurevich. 2006. JamSpace: Designing A Collaborative Networked Music Space for Novices. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Paris, France, 118–123. http://www.nime.org/proceedings/2006/nime2006_118.pdf

[9] Galen Joseph-Hunter, Penny Duff, and Maria Papadomanolaki (Eds.). 2011. *Transmission Arts: Artists and Airwaves*. PAJ Publications.

[10] Velimir Khlebnikov. 1987. *Collected Works of Velimir Khlebnikov Volume I.* Harvard University Press, Boston, Massachusetts. 392 pages.

[11] Golan Levin. 2005. A Personal Chronology of Audiovisual Systems Research. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Vancouver, BC, Canada, 2–3. http://www.nime.org/proceedings/2005/nime2005_002.pdf

[12] David Ramsay and Joseph Paradiso. 2015. GroupLoop: A Collaborative, Network-Enabled Audio Feedback Instrument. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 251–254. http://www.nime.org/proceedings/2015/nime2015_119.pdf

[13] Charles Roberts, Graham Wakefield, and Matt Wright. 2012. Mobile Controls On-The-Fly: An Abstraction for Distributed NIMEs. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan, Ann Arbor, Michigan. http://www.nime.org/proceedings/2012/nime2012_303.pdf

[14] Charles Roberts, Graham Wakefield, and Matthew Wright. 2013. The Web Browser As Synthesizer And Interface. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Graduate School of Culture Technology, KAIST, Daejeon, Republic of Korea, 313–318. http://nime.org/proceedings/2013/nime2013_282.pdf

[15] Graham Wakefield, Charlie Roberts, Matthew Wright, Timothy Wood, and Karl Yerkes. 2014. Collaborative Live-Coding with an Immersive Instrument. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, London, United Kingdom, 505–508. http://www.nime.org/proceedings/2014/nime2014_328.pdf