Addressing Creativity in Network Communication for Computer Music Interaction

Flávio L. Schiavoni[1], Pedro H. de Faria[2], Jônatas Manzolli[2]

[1]Computer Science Department
Federal University of São João del-Rei
São João del-Rei - MG - Brazil

fls@ufsj.edu.br

[2]Interdisciplinary Nucleus for Sound Studies (NICS)
Department of Music
University of Campinas
Campinas - Brazil

pedro.faria@nics.unicamp.br, jonatas@nics.unicamp.br

Abstract. *This paper draws analogies between computer network communication and real time collaboration in music. Computer networks have different addressing methods (namely broadcast, unicast and multicast), these addressing methods are directly related to music performance interaction through existing network communication objects in Pure Data. These analogies and recommendations envisage assisting in the development of new collaborative music creation and performance which may explore the different possibilities that each model delineates. Specific artistic applications will be used as examples to benefit musicians and composers to apprehend network music performance potentialities.*

1. Introduction

Music is traditionally seen as a collaborative and cooperative art field. This applies for orchestras, chorals, bands, rock'n'roll groups, duos or even a guitar playing accompanied by voices. Most part of musical practices are not individualistic and represent and express human forms of interaction and communication. Since the development of the Internet, computer networks propagated new forms of human interaction communication. The computer is now one of the most powerful tools for music creation [Iazzetta 2010] and computer networks can be used to expand this instrument and create several new models for music interaction [Malloch et al. 2008]. Computer music can explore this concept and propose new approaches to enhance collaboration and cooperation in music. Computer network concepts may be associated with music interaction models, such as addressing methods, data flow and communication roles, leading to efficient results in computer music interaction.

These concepts will be discussed in a theoretical point of view and relate them with music interaction scenarios in a computer music environment tool called Pure Data. Pure Data[1] (aka Pd) is a free graphic programming environment widely used in real time

[1]Available on http://puredata.info

music performances. Pd supports network communication by means of an object collection that suits specific protocols, data flow and addressing tasks. A set of musical scenarios will be described in order to elucidate how these objects can be used to perform collaborative music. The related computer network theory will be mapped to music interaction models. Such scenarios consider decisions regarding fundamental aspects of an artistic project that may directly intervene in the choice of a specific tool and network configuration setup, namely the overall number of performers, their disposition in sections and hierarchy for guidance, interference and interaction of sonic material.

Therefore this paper provides a technical and compositional background of relevant topics related to computer music performance interaction in the context of network and ubiquitous music based on a collection of free and accessible technological tools. Such approach may help musicians to better understand how to create computer music groups and how to address it using computer networks in a creative and technically consistent way.

The remainder of this paper is structured as follows: section 2 discusses the theoretical approach to network communication; section 3 discusses Pure Data as an accessible tool for network music interaction; section 4 presents and discusses three projects by the authors that serve as technical and artistic implementations of the concepts and tools regarding computer music performance and computer network.

2. Network Communication

In computer networks, communication is initiated by one party and another party responds to it. Network communication uses two terms to define these roles: server and client. The client program initiates the conversation while the server responds. Together, server and client create a distributed application [Donahoo and Calvert 2009, p. 7].

Despite the terms, these roles are associated only with communication start up. Servers wait for a client connection while clients need to know the server address to connect to. Once the connection is done, these roles are replaced by an application protocol that defines the data flow and how resources will be shared through a network.

Although there are other communication protocols, UDP [Postel 1980] and TCP [Postel 1981] are the most used transport protocols in the TCP/IP stack[2]. Unlike UDP, TCP uses a "handshake" when the communication begins, confirming message reception (through ACK messages) and the client connection. Because of these details, UDP is known as a connectionless, unreliable protocol while TCP is a connection oriented reliable protocol.

Compared to existing forms of human communication, TCP is similar to a telephone call, where the communication is only possible if both sides are connected. UDP is similar to a mail service, where one can send information and not be sure it reached its destination. Using this mail analogy, the receiver is like the server and the sender is like the client because the sender needs to know previously the receiver address to initiate a communication. This feature allows UDP to use different addressing methods, presented in the following section.

[2]SCTP and DCCP are also transport protocols in the TCP/IP stack.

2.1. Addressing Methods

Addressing methods are the associations between the destination address and a network endpoint. In network communication, the addressing can be one-to-one or one-to-many association. A one-to-one association is called Unicast and is defined by a connection addressing form such that a sender can reach only one endpoint in a data transmission. Broadcast (sender sends to all connected clients) and Multicast (receivers choose if they want to receive from the sender) use a one-to-many association where in a single transmission one sender can reach multiple endpoints.

2.1.1. Unicast

Unicast communication is a peer to peer model. In a simple unicast connection, the server has one client only. Both protocols, UDP and TCP can use this addressing methodology.

In a unicast communication, the client needs to know the server's IP address and port to connect. The server can, if necessary, learn the client's addresses information when it receives the initial communication from the client.

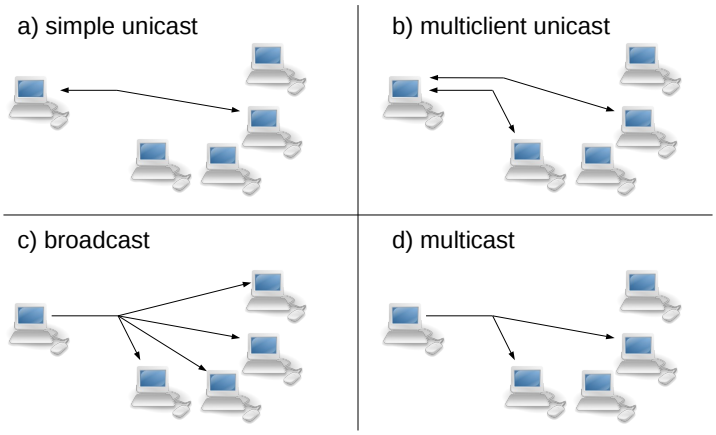

Figure 1. Network addressing methods

As shown in Fig.1.a, unicast is a one-to-one association and only two peers can be part of the communication.

2.1.2. Multiclient Unicast

The simple unicast communication does not support multiple clients. However, it is possible to implement a multiclient unicast server that accepts several clients. The server can send or receive different data to / from different clients, as presented in Fig.1.b. The server can also send the same information to every client looping the list. In this case, if a client is slower than others it can create an undesirable network bottleneck, slowing down the entire network. Despite the multiclient feature, multiclient unicast still a unicast communication and can be implemented with UDP or TCP.

2.1.3. Broadcast

In electrical engineering and communication, broadcast is a data transmission to a dispersed audience, such as in radio or TV. The sender transmits the data through a particular channel and several receivers can be connected on this channel and receive the transmitted data. In this way, broadcast is useful when a host needs to communicate without knowing the receiver addresses.

It is also possible to use broadcast on TCP/IP networks. In this case, the broadcast comes from a set of local area network (LAN) reserved addresses. The data sent to the broadcast reserved address (e.g. 255.255.255.255 to IPv4 LAN) is received by every computer in the same subnet [Mogul 1984]. TCP/IP broadcast can use different ports to identify a data channel. Thus, every server that wants to receive broadcast data creates a socket that listens to a broadcast port.

Similar to TV stations, a broadcast sender does not know who is receiving the data, how many receivers are listening to the associated port, or even if there is a receiver waiting for data. Unlike TV stations, it is acceptable to have several senders emitting broadcast data to the same port at the same time. Since broadcast is connectionless, it is not possible to implement this communication with a connection oriented protocol, namely TCP.

It is important to note, as presented in Fig.1.c, that every computer in the LAN will receive every broadcast message, even if they do not have a socket waiting for the data. In the latter case, all the received data is discarded locally. Instead of the large broadcast utilization, IPv6 has no broadcast addresses. Their function were superseded by multicasting addresses [Hinden and Deering 2006].

2.1.4. Multicast

Broadcast is a powerful communication paradigm but if most of the receivers are just discarding packets locally, it unnecessarily floods the network. Multicast is the solution to solve this conflict.

Similar to broadcast, network multicast communication is a set of reserved network addresses. Unlike broadcast, multicast uses an address range, from 224.0.0.0 to 239.255.255.255 in IPv4 [Cotton et al. 2010]. Also differently from broadcast, multicast clients will only receive a stream of packets if they have previously joined the specific multicast group address.

In order to receive multicast messages, every receiver has to subscribe in a multicast group. A network device, such as a server, router, switch or access point, has to be responsible for managing the multicast groups. It can also provide information about who is listening in the multicast group.

Unlike multiclient unicast, multicast is a one-to-many association. As presented in Fig.1.d, the same message reaches every receiver subscribed in a multicast group. For this reason multicast is very useful to network discovery services or to communicate with unknown devices [Malloch et al. 2008].

Since multicast avoids message retransmission for each client, it saves on network bandwidth and its usage is recommended for high bandwidth network services such as audio and video transmission.

3. Pure Data Network Objects

Pure Data provides a set of objects for network communication that are accessible and free to use. These are part of the implementation on the creative projects of the next section. In the current section Pd network objects will be introduced in order to illustrate how to implement general music communication using the addressing methods previously described. All of the above mentioned addressing methods are available for music interaction.

Table1 displays a set of Pd objects developed specifically for this goal. In addition to the library name we present server and client instances, the protocol, the data flow and the addressing methods. Assuming that the client must know the server address to start the communication, receivers are displayed as servers and senders as clients.

Table 1. Pure Data network objects

Library	Server	Client	Protocol	Data flow	Addressing
Vanilla	netreceive	netsend	UDP / TCP	client sends	multiclient unicast, broadcast[3]
Maxlib	netserver	netclient	TCP	bidirectional	multiclient unicast[4]
iemnet	udpserver	udpclient	UDP	bidirectional	multiclient unicast, broadcast[5]
iemnet	udpreceive	udpsend	UDP	client sends	unicast, broadcast[6]
mrpeach	udpreceive	udpsend	UDP	client sends	unicast, broadcast, multicast

Pure Data also has objects to transmit audio streams, video streams and some TCP specific objects. Because TCP objects are always limited to unicast or multiclient unicast addressing methods we didn't include them here.

4. Creativity and Network Communication in Computer Music Interaction

A desired music interaction scenario can lead musicians and composers to a particular protocol, data flow and addressing method choice. Even though any network connection could be used in music performance, issues such as role definition, connection ease, and network environment configuration are decisive factors in determining best practices for the interaction performance [Obici and Schiavoni 2011]. Another important aspect is the proper technology selection and design during the project creation and execution [Rottondi et al. 2016, Gabrielli and Squartini 2016].

The different solutions presented here can assist musicians in selecting the best network connection for a specific network music performance scenario. In network music it is possible to map various data flow types between musicians: control, symbolic, audio

[3]It is possible to address the connection to a broadcast address, e.g. 255.255.255.255.

[4]Netserver can send / receive different data to different clients and identify which client sent which message. The broadcast message documented in the help is only multiclient unicast sending.

[5]If connected to broadcast address, the data flow is not bidirectional and just the client sends information. The broadcast message documented in the help is only multiclient unicast sending.

[6]It is possible to achieve broadcast addressing the connection to 255.255.255.255.

signals etc. The implementation of this data flow leads one to decide who should send data and who should receive the data as well as how many participants should send and receive data simultaneously.

Another important aspect refers to the participation of the audience in such artistic projects. The interactivity can be highly enhanced and new musical results may emerge if the audience can collaborate in a musical performance or becomes an active user of a multimedia installation system. This is a very important discussion when dealing with creativity in contemporary art [Wu et al. 2017], multimedia installations and ubiquitous music [Keller and Lazzarini 2017].

Decisions concerning specific musical parameters are typically determined by the main goal of each artistic project. These decisions, and also aesthetic implications for each of the various musical setup scenarios described below, are not the focus of this discussion up to this point.

Research in network communication for music collaboration frequently explores music creation, performance and diffusion through the Internet [Jordà 1999]. Experimentation of models for local communication that enables or enhances music interaction between machines and humans in a local network context are not much explored, and generally laptop orchestras are the most frequent users of such models [Rebelo and Renaud 2006].

In order to improve the discussion, we provide a series of three recent implementations about collaborative works of the authors, all related to these artistic practices in different technical network and ubiquitous music contexts.

4.1. D.S.C.H.

D.S.C.H. (acronym for Dmitri Shostakovich and also related to the musical motive used by the composer in his works) consists of an immersive interactive sound installation, where up to eight users are connected through wi-fi network in order to intuitively control audio triggering, processing and diffusion on an octophonic system from their mobile devices. The implementation is based on two free software platforms: MobMuPlat[7] and Pure Data (Pd).

The developed mobile application provides the user with several interaction possibilities with the system through his/her personal device (either Android or iOS, phone or tablet). MobMuPlat provides network communication through a set of protocols, *D.S.C.H.* is solely based on a multiclient unicast, where all clients (mobile users) send their data only to the server computer, therefore clients do not communicate between themselves directly through data. This choice enhances musical immersion of the users by stimulating an accurate listening and reaction to other users choices of excerpts and processing.

The application is designed to allow an intuitive control of parameters that are sent to a server computer (where all processing and sound diffusion happens), therefore there is no technical restriction regarding the memory or processing capabilities of the user's device. Each user has an independent set of controls, associated to spatialization, audio sample triggering (an algorithmic selection of excerpts based on the musical character

[7]MobMuPlat is a free software available at: `http://danieliglesia.com/mobmuplat/`

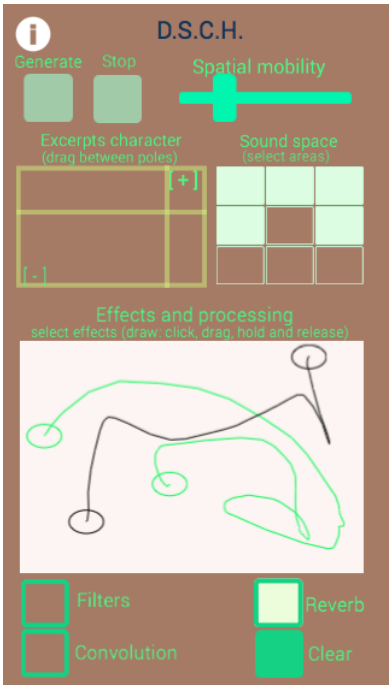

Figure 2. User control interface developed in MobMuPlat. The use of drawings and sliders enhance an intuitive control of parameters.

desired) and a conjoined set of three sound processing units divided between *reverb*[8], *filters*[9] and *convolutions*[10].

Furthermore, a new interaction model, which is currently in development, foresees adding eight ultrasonic sensors (built using Arduino) closely to each audio speaker. This set of sensors will provide the specific location of users in the interactive space. The localization data collected by the sensor will be the input of two algorithms that will allow the exploration of additional layers of interaction:

(a) a cellular automata algorithm based on a matrix that represents the spatial distribution of users. The platform sound will accompany the user as he/she moves in the installation space and a set of rules will guide the users as well. For example: the overpopulation (accumulation of several users) in a specific area of the installation will be responsible for the progressive ”death” of the audio in that particular region;

(b) a game theory (zero sum game) algorithm where users may gain direct access to control neighbors' sound processing and/or audio excerpt character, users scores in the game is based on their behavior in the space compared to other users. For example: users responsible for the ”death” of the audio in a region will receive a lower score and then may loose the control of their respective mobile devices, that is given to another user with higher scores.

In terms of network communication, this new implementation will require an ad-

[8]Implemented with the *freeverb* object in Pd, check the documentation for details.

[9]Three filters were used: high-pass, low-pass and a voltage control filter.

[10]Four types of convolutions based on FFT processing were used: phase, amplitude, cross and complex convolutions.

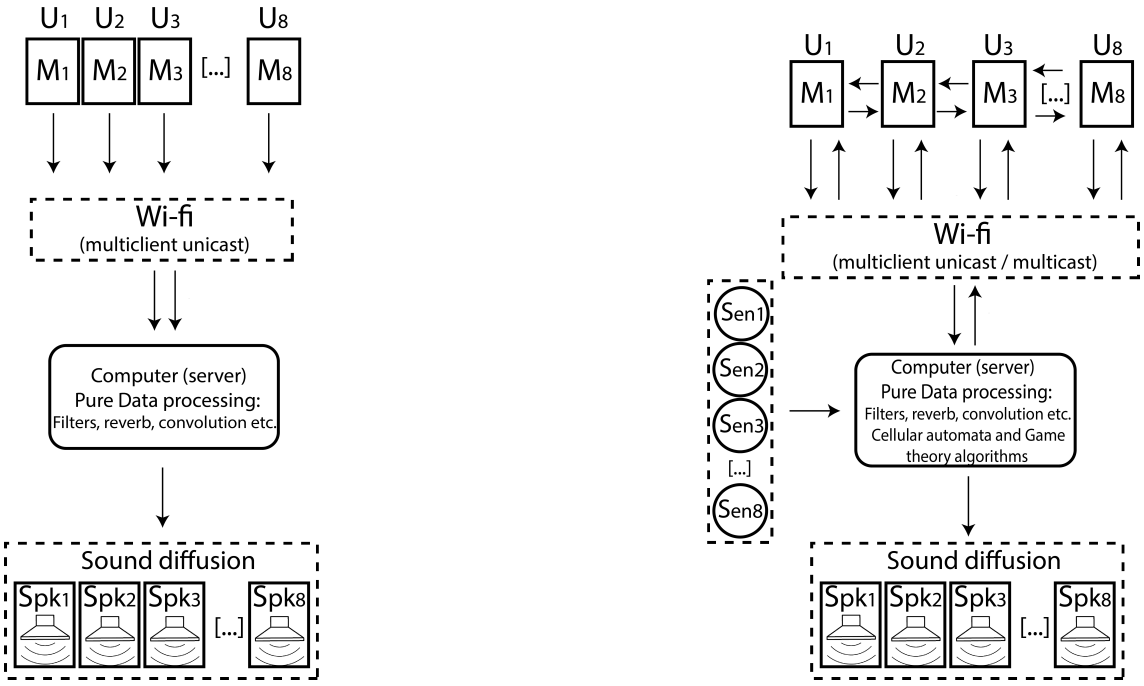

Figure 3. Current (left) and in development (right) implementations of *D.S.C.H.*

ditional many-to-many communication protocol in MobMuPlat called *Ping & Connect*, where "each client pings out its player number and IP address via multicast. Each client keeps track of other clients on the network, and may send unicast messages to all clients, or to individual clients by player number" [Iglesia]. Figure 2 presents both implementations discussed for *D.S.C.H.*, where *U* is an "user", *M* is a "mobile device", *Sen* is a "sensor" and *Spk* is an "audio speaker".

4.2. CromaCrono≈

In order to explore meaningful relationships between agents and environmental stimuli in a virtual space and understand their interaction with animations and sound-generative processes in real time, CromaCrono≈ was created with the notion that interactive media within mixed/virtual reality environments induces an agent coupling with the space, it is defined as the *sensing of Presence* [Wasserman et al. 2003, Sanchez-Vives and Slater 2005, Dubois et al. 2009].

CromaCrono≈ can be seen as an interactive system for audio-visual improvisation that produces digitally synthesized sounds and images in real time [Manzolli 2015]. Departing from observations on the way sensory processes are integrated with the environment, it exploits the interaction of space and time from the human agent perspective. Simple geometric shapes and computer-synthesized sounds support an audio-visual textural architecture (see figure 4). The Boids algorithm is used to control several parallel processes generating sounds and animating graphics in real time [Reynolds 1987, Reynolds 1988]. The Boids trajectories are used to control the display of hundreds primitive geometric shapes that vary in shape, color, speed and dispersion in space. All this variations produce the audio-visual texture that is coupled with generative rules for controlling sounds and interactions with local and remote agents. We also acquire that in

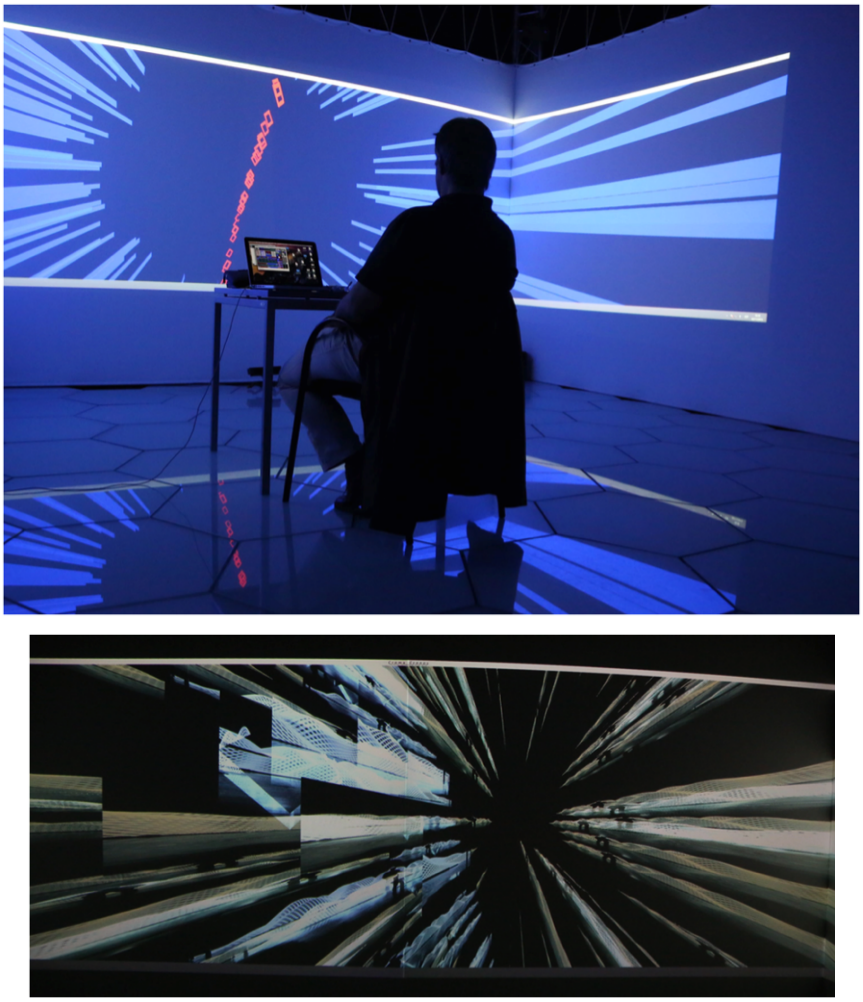

Figure 4. Two images generated by CromaCrono≈ environment illustrating the visual produced by the system in real time. It is a texture of interwoven processes which density increases as much as they are interconnected.

these processes the computer acts as an autonomous adaptive sentient guide that assists humans to explore creative spaces and discover novel patterns driven by the man-machine interaction. Moreover, the user's parametric control on the computer GUI is defined here as *explicit interactions*, complementary, and *ubiquitous interaction* is that of a remote agent interaction over the Internet.

The entire system works as an unified generative process that digitally synthesize sounds and images, receive GUI control parameters and remote control from the Internet, and in turning generate audio and image outputs (see figure 5). Despite of many processes controlled simultaneously, the system is designed to operate in a loop of 14 parameters. They are organized in a "Composition Curve" with 10 iterated-sections, 140 parameters in total (see figure 6). The economical representation allows fast broadcasting over the Internet. That is, a small set of compositional operations, makes it possible to evolve and share in real time compositions/improvisations via Internet. Local and remote agents control together the generative process of CromaCrono≈[11]

[11]The operation of CromaCrono≈ and the audiovisual texture generated in real time can be seen at

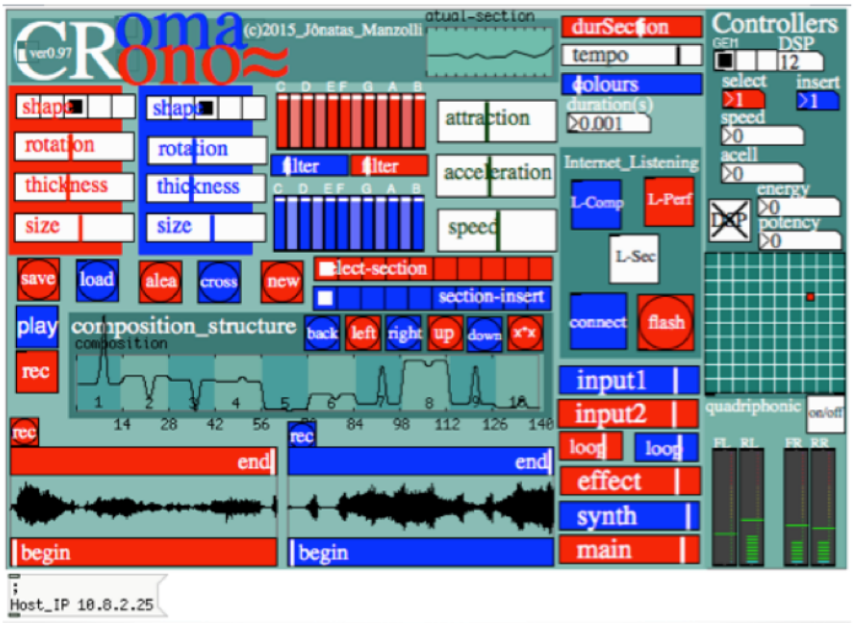

Figure 5. GUI of CromaCrono≈ showing all the integrated control parameters of the system.

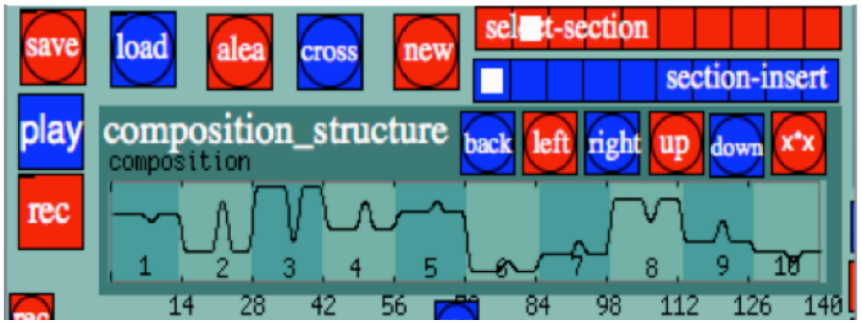

Figure 6. In detail the "Composition Curve" and the buttons to apply genetic and structural operations over the curve. A remote user can also "press" the buttons using a mirror program in a location via a VPN connection.

4.3. Concerto para Lanhouse

Concerto para Lanhouse (Lanhouse Concert) is an audiovisual installation for computers connected to a Local Area Network - LAN, presented in Figure 7. Lanhouse is how Internet Cafe is popularly called in Brazil, commercial venues provided with a LAN connection with Internet [Obici and Schiavoni 2011]. This creation used Pure Data to combine synthetic sounds and visual effects with GEM (Graphics Environment for Multimedia) creating an immersive environment of synchronized performance. The synchronization was implemented using Pd network objects and a control machine, apart from those in performance space, attains the role of a conductor, responsible to send commands to every machine regarding how to perform. These commands can be thought as a network score and has informations concerning sound synthesis and visual effects.

The instrument programming had two implementations, one using net-

<https://vimeo.com/145326063>

Figure 7. The Concerto para Lanhouse installation.

send/netreceive and other using netclient/netserver. Since the first approach used broadcast addressing, the same commands were sent to every machine by the conductor. Every machine was identified by a number and it was used to select the received commands and to determine if the machine should follow or ignore the instruction. The amount of network traffic was very high and it was not easy to really synchronize the machines using this approach. A second approach, using netclient/netserver was implemented using a multiclient unicast connection. Thus, the conductor could send independent commands to each machine, resulting in a higher processing demand but requiring a lower network traffic.

5. Conclusion

The discussion considered attributes related to connectivity, addressing methods that are crucial for the creation and implementation of interactive network music. These attributes are a necessary foregoing step for every network music interaction system proposal.

Three representative artistic works created using these theoretical concepts and computational tools were presented. These works used different addressing solutions which were specific to the creative and artistic demands of each project. Each of them explored different aspects within the context of network interaction and ubiquity. Table 2 presents the summarized technical information about the three artistic projects previously discussed.

Table 2. Details regarding projects implementation

Project	Server	Client	Protocol	Data flow	Addressing
D.S.C.H. (1)	netreceive	netsend	UDP	client sends	multiclient unicast
D.S.C.H.(2)	udpreceive	udpsend	UDP	client sends	multicast
CromaCrono≈	udpreceive	udpsend	UDP	client sends	multicast
Lanhouse Concert(1)	netreceive	netsend	UDP/TCP	server sends	broadcast
Lanhouse Concert (2)	netserver	netclient	TCP	server sends	multiclient unicast

As pointed along the article, the addressing methods and computational tools must be chosen according to a given aesthetic and practical necessity in the context of an artistic project. Using unicast, broadcast or multicast in a music collaboration scenario depends

on several aspects, including the fraction of network hosts interested in receiving the data, the knowledge of the communicating parties and how exclusive a network message is.

Comparing the technical summary for the three artistic applications in Table 2, some important observations emerge. Firstly, is possible to notice that different musical demands may use the same set of Pd objects to implement them, but with different addressing. For instance in the usage of netsend/receive object for multicasting in D.S.C.H. (version 1) and broadcasting in Concerto para Lanhouse (version 1). Secondly, same musical demands may use different sets of Pd objects given the system technical setup requirements, for instance the UDP multicasting in D.S.C.H. (version 2) is locally networked, while UDP multicasting in Cromacrono≈ communicates remotely through VPN over the internet, requiring another set of musical objects to implement a similar musical function for the users.

The technical setups approached to convene a specific computer music interaction scenario of an artistic project, generally emulates music interaction in standard acoustic music practice and also may expand the usual boundaries of the common practices. Therefore, the standard conceptions regarding creativity in music interaction in terms of performance and compositional demands are also expanded.

To illustrate the presented concepts, a set of Pure Data network objects were presented and classified based upon their protocols, addressing methods and data flow. Pure Data was chosen for this discussion given that it is a free software that provides a consistent straightforward programming vocabulary for musicians, and a large community of developers and users. Furthermore, this tool was applied in the creation of the three examples presented in this paper. Even though all examples for data flow operation were presented as Pd messages, it is important to state that the same procedures are valid to other types of data, such as OSC and audio/video streaming, among others.

Since Pure Data is an open source software, the source code of the listed objects is available, therefore attainable to read, explore its working structure and implement such features in other applications or platforms. The set of models presented are standard but not exhaustive, other possibilities may emerge depending on the necessity of specific artistic schemes.

References

Cotton, M., Vegoda, L., and Meyer, D. (2010). IANA Guidelines for IPv4 Multicast Address Assignments. RFC 5771 (Best Current Practice).

Donahoo, M. J. and Calvert, K. L. (2009). *TCP/IP Sockets in C Bundle: TCP/IP Sockets in C, Second Edition: Practical Guide for Programmers (Morgan Kaufmann Practical Guides)*. Morgan Kaufmann.

Dubois, E., Gray, P., and Nigay, L. (2009). *The engineering of mixed reality systems*. Springer Science & Business Media.

Gabrielli, L. and Squartini, S. (2016). *Wireless Networked Music Performance*. Springer Singapore.

Hinden, R. and Deering, S. (2006). IP Version 6 Addressing Architecture. RFC 4291 (Draft Standard). Updated by RFCs 5952, 6052.

Iazzetta, F. (2010). *Música e mediação tecnológica*. Perspectiva, São Paulo - Brazil.

Iglesia, D. MobMuPlat Rough Documentation. Project Website. Available at http://danieliglesia.com/mobmuplat/doc/index.htm.

Jordà, S. (1999). Faust music on line: An approach to real- time collective composition on the internet. *Leonardo Music Journal*, 9:5–12.

Keller, D. and Lazzarini, V. (2017). Theoretical approaches to musical creativity: The ubimus perspective. *Musica Theorica*.

Malloch, J., Sinclair, S., and Wanderley, M. M. (2008). A network-based framework for collaborative development and performance of digital musical instruments. In Kronland-Martinet, R., Ystad, S., and Jensen, K., editors, *Computer Music Modeling and Retrieval. Sense of Sounds*, pages 401–425. Springer-Verlag, Berlin, Heidelberg.

Manzolli, J. (2015). Multimodal generative installations and the creation of new art form based on interactivity narratives. In *Proceedings of 18th Generative Arts Conference*, Florence.

Mogul, J. (1984). Broadcasting Internet Datagrams. RFC 919 (INTERNET STANDARD).

Obici, G. L. and Schiavoni, F. L. (2011). Concerto para lanhouse. In *Proceedings of the Linux Audio Conference*, pages 170–174, Maynooth, Ireland.

Postel, J. (1980). User Datagram Protocol. RFC 768 (INTERNET STANDARD).

Postel, J. (1981). Transmission Control Protocol. RFC 793 (INTERNET STANDARD). Updated by RFCs 1122, 3168, 6093, 6528.

Rebelo, P. and Renaud, A. B. (2006). The frequencyliator: distributing structures for networked laptop improvisation. In *Proceedings of the 2006 NIME*, NIME '06, pages 53–56, Paris, France, France. IRCAM ; Centre Pompidou.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *ACM SIGGRAPH computer graphics*, volume 21, pages 25–34. ACM.

Reynolds, C. W. (1988). Not bumping into things. *Computer Graphics*, page G1.

Rottondi, C., Chafe, C., Allocchio, C., and Sarti, A. (2016). An overview on networked music performance technologies. *IEEE Access*, 4:8823–8843.

Sanchez-Vives, M. V. and Slater, M. (2005). From presence to consciousness through virtual reality. *Nature Reviews Neuroscience*, 6(4):332.

Wasserman, K., Manzolli, J., Eng, K., and Verschure, P. (2003). Live soundscape composition based on synthetic emotions: Using music to communicate between an interactive exhibition and its visitors. *IEEE MultiMedia*, 10(4):82–90.

Wu, Y., Zhang, L., Bryan-Kinns, N., and Barthet, M. (2017). Open symphony: Creative participation for audiences of live music performances. *IEEE MultiMedia*, 24(1):48–62.