
Article:
Prior, David, Reuben, Federico orcid.org/0000-0003-1330-7346, Biscoe, Ian et al. (1 more author) (2017) Designing a system for Online Orchestra: Computer hardware and software. The Journal of Music, Technology and Education. 185–196. ISSN 1752-7074

https://doi.org/10.1386/jmte.10.2-3.185_1

Designing a system for Online Orchestra:

Computer hardware and software

David Prior | Federico Reuben | Ian Biscoe | Michael Rofe

Abstract

Online Orchestra sought in its pilot performance to enable musicians in four remote locations in Cornwall, United Kingdom, to make music together over the Internet. This article considers the processes by which computer hardware and software platforms were selected, integrated and optimized for the performance. Starting with an overview of guiding design principles, details of the computer hardware and software platforms used are provided. Audio- and video-streaming solutions are surveyed, leading to a detailed analysis of JackTrip and its deployment in the final system. The network environment in which the Online Orchestra performance took place is also considered, including specifications of each of the four venues.

Keywords

Online Orchestra; hardware; software; network; JackTrip; latency

Introduction

Online Orchestra sought to enable a telematic performance environment that would allow young and amateur musicians in remote locations to make music together online. As described in Rofe et al. (2017b), in this special issue, based on this central aim, a number of starting premises were established early in the project, and these had significant ramifications in terms of requirements for system design. In particular, the following project aims each impacted upon design approaches:

1. *Enable access from remote locations*. Connecting musicians from geographically isolated locations would mean responding to the available Internet connections in each location, in terms of bandwidth and connection quality.
2. *Design a scalable solution*. A key aim of Online Orchestra was to allow community groups and schools to be able to make music together. Therefore, solutions that relied on relatively cheap equipment, or, better, equipment that future users might already own, would be preferred over specialist or expensive technologies, in order to enable a design solution that could be repeated and scaled in the future.
3. *Enable a conductor*. Whilst other similar projects have used a variety of approaches (see Rofe and Geelhoed 2017), these are often more suited to experienced, professional musicians. The focus here on young and amateur musicians meant that a more traditional orchestral format, using a

conductor, would be preferable. This would require a video stream in addition to audio.

4. *Enable a large, multinodal ensemble*. Having multiple nodes (four in the case of the pilot performance) would require the streaming of large amounts of data between multiple locations.
5. *Enable a connected and immersive musical experience*. Benchmarks of connectedness and immersion on the parts of performers were consistently referenced throughout the design process. In order to achieve these, a high-quality audio-visual stream would be required.

As discussed in Rofe and Reuben (2017), in this special issue, the key challenge for telematic performance is how to handle latency – the time delay involved in processing data and sending/receiving it between locations. As such, priority was given to the development of Online Orchestra's bespoke latency-control programme (see Rofe and Reuben 2017). Time and budgetary limitations meant that resources were not available to develop similarly bespoke hardware and audio-visual streaming tools. Instead, the aim was to select, integrate and in some cases optimize through a process of action research the most suitable tools already available.

This article reports the computing hardware and software solution designed for Online Orchestra (peripheral equipment is considered in a separate article; see Prior et al. 2017). It also reports the decision-making process behind this design. In many cases, decisions stem directly from the aforementioned starting premises. In other cases, the design team adopted an action research approach (see Kolb 1984), in which potential solutions were tested and evaluated with respect to the above aims, and then reiterated

until a suitable design solution was established. At times, this testing involved participant musicians.

Network context

One of the key drivers of Online Orchestra was to facilitate telematic performance across the kinds of Internet connectivity typically found in remote community contexts such as schools, community and arts spaces, or church halls. This in turn had a significant impact on decision-making with respect to computer hardware and software. Although it was possible to draw on the literature associated with telematic performance practice to date (see Rofe et al. 2017b), the majority of recent telematic performances have been developed in the context of unusually high-speed networks such as Jisc's Janet (Joint Academic Network) in the United Kingdom, Géant in Europe and Internet2 in the United States. One particular challenge for Online Orchestra then was to create a telematic environment across a range of remote locations served by Internet connections that were nowhere near the speeds often available to previous telematic performance projects.

Online Orchestra's pilot performance took place in relatively remote locations around the county of Cornwall, United Kingdom, and its outlying Isles of Scilly. Although Cornwall has been a recent recipient of significant investment in network infrastructure, the reality outside its few major urban areas is inconsistent broadband access, with the last mile or so of the connection often still being made over legacy copper wires, which limit the speed potential of the main fibre-based infrastructure. Even in larger towns and cities, the possibility for a fibre optic final connection to the property

(FTTP, or fibre to the premises) is often still hampered by difficulties in replacing underground wire circuits in areas that are hard to access or areas occupied by listed buildings or important heritage sites.

The venues used for the pilot performance provided excellent case studies for the variety of contexts in which Online Orchestra might be used in the future. Each presented its own peculiarities in terms of both the overall speed of the connection possible and issues such as reliability, symmetry of upload and download speeds, and the negotiation of firewalls. Detailed descriptions of each context demonstrate the range of considerations that were necessary.

Falmouth University (Penryn Campus)

Falmouth University is connected to the Janet (Joint Academic Network) network operated by Jisc. During the period in which the project took place, the university had a 1Gbps connection to the nearest Janet hub at Plymouth, which in turn was connected to the rest of the 10Gbps Janet backbone. During early workshop tests, where multiple performance nodes were simulated within the campus, this additional bandwidth was useful for testing 'ideal' versions of the system. It also proved useful in the final design solution, enabling Falmouth to act as a server node for other locations.

Truro Cathedral

Prior to the commencement of Online Orchestra, Truro Cathedral had a single low bandwidth (1–2Mbps) ADSL (asymmetric digital subscriber line) connection over a

wired circuit in the shop adjoining the cathedral itself. However, offices used by Cathedral staff – located across the road from the Cathedral – had an FTTP connection, and an exiting conduit with a cabled Ethernet link existed between the Cathedral and the offices. The link to the fibre connection in the Cathedral's offices provided a download speed of around 73Mbps and an upload speed that averaged around 18.5Mbps. This asymmetric connection facilitated the streaming of more audio and video channels into the venue than it was possible to stream out. However, the upload speed from Truro, in combination with the maximum download speeds at the two remote venues in Mullion and on the Isles of Scilly (see below), still represented a key limitation of the system in terms of the number and quality of audio and video channels that could be steamed between nodes.

Mullion School, Lizard Peninsula

Mullion School had two existing wire connections, the first, of approximately 12/4Mbps, used to support day-to-day school operations, and a second of 27.7/7.7Mbps, which was dedicated to Online Orchestra for the duration of the project. With the lowest upload speed of the four nodes (7.7Mbps), Mullion School effectively set the system-wide limits with respect to the number and quality of video streams that could be used in the overall performance and also the number of audio channels that could be sent from Mullion to the other nodes.

Five Islands' School, St. Mary's, Isles of Scilly

Five Islands' School had until the year of the project relied mostly on satellite and radio links with the mainland. As a part of the Cornwall Superfast Broadband programme, a project had been initiated in 2014 to lift a disused transatlantic fibre optic telecommunications cable and relay it to St. Mary's, the main island in the Isles of Scilly. This had given the islands their first high-speed network to the mainland. Five Islands' School is located on St. Mary's, relatively close to the main exchange, so was able to be connected using existing infrastructure at the new download speed of 41Mbps and upload of 8Mbps.

Computer platforms

Ideally we would use a single computer at each of the locations, handling both the audio and video input and output to the network. These tasks include the capture of audio and video, encoding and decoding of video into compressed data streams (audio was uncompressed in the final design; see below) and driving local display screens and audio interfaces; it would also run Online Orchestra's latency-control programme.

An initial decision was taken to base the project entirely around Macintosh computers: JackTrip had been chosen as the audio-streaming solution (see below), which was developed for Mac and is only available as a stable release on the Mac platform. However, once testing of a full system began, it became clear that a single, low-cost computer per node was not sufficient. Even on Mac Pro computers,[1] CPU usage was

averaging in excess of 80 per cent once multiple channels of video were introduced, with frequent higher spikes, and there were significant additional limitations in terms of i/o bottlenecks and the interrupt needs and scheduling of many simultaneous flows of audio and video data. This resulted in dropped frames and dynamic variation in the video signal. As more nodes were introduced, these effects were exacerbated by the peer-to-peer nature of the system (described below). Whilst these effects might have been deemed acceptable in some conference calls, trials were run during the Online Orchestra design phase in which participant musicians found interruptions of this kind to be very distracting and reported difficulties in concentrating on the gestures of the conductor.

As discussed in the introduction to this article, the project aimed to deploy readily available open source and commercial software, and this at times rendered exact diagnostics difficult. With this in mind, a simple solution was to add capacity to the computing platforms. As such, two machines were deployed per node: a Mac streaming audio and a PC streaming video. By using separate hardware and software solutions for video and audio, CPU usage and scheduling problems were significantly diminished, which in turn reduced video artefacts to the point where they no longer caused disruption to musicians.

Software platforms

The peripheral equipment and interfaces used to capture audio and video are discussed in a separate article in this special issue (Prior et al. 2017). Once audio and video had been acquired, software solutions were needed to control latency, to synchronize audio and

video across the two computer systems (see Rofe and Reuben 2017) and to encode and decode audio and video into appropriate data streams and integrate with the network. It was immediately clear that the former task of managing audio and video capture and processing required a bespoke solution, for which Cycling 74's Max was used, with Jitter to manage the video components. Max with Jitter provided a flexible programming environment in which both audio and video could be processed.

As described above, in the earlier stages of the project, both audio and video were processed on a single computer, and this was achieved via a Max patch that handled both elements. Once audio and video streams had been divided across Mac and PC computers, respectively, Max patches were adapted on each hardware platform to reflect their now-simplified roles, adding features to both that enabled UDP (user datagram protocol)-based synchronization between the two systems. Whilst Max with Jitter provided the necessary functionality for capturing and processing audio and video streams and for controlling network latency, Max's own streaming tools are of lower quality than other open source and commercially available software. As such, a range of streaming platforms were evaluated and tested.

A key aim for Online Orchestra was to enable the formation of a large, multinodal ensemble. This presented a scenario that required multiple channels of simultaneously sounding audio. Platforms such as Skype were therefore rejected due to their echo cancellation algorithm and automatic gain control, which in effect only enables sound from one location at a time (see Fazlul Haque et al. 2013). Moreover, audio compression algorithms used by software such as Skype tend to be designed specifically for speech, filtering frequency components that are not fundamental to speech transmission (see Sun

et al. 2013). Given the ambition to create, as far as was possible, a highly immersive musical experience for performers, high-quality audio was found to be significant – participants in testing had a strong preference for high-quality audio, even if this came at the detriment of video quality. Platforms that have limited control over audio quality and echo management were therefore rejected.

In terms of video, the capacity to manipulate the resolution was crucial, in order to respond to the network conditions of different nodes. Also, as discussed in Prior et al. (2017), participant feedback led to the decision to have a separate screen per video feed in each location, rather than a single screen showing tiles of all other locations. This created the need to be able to isolate feeds into dedicated windows, which could then be dragged around a multi-display desktop. Few platforms offer that functionality without significant impact on CPU: web-based systems such as appear.in, for instance, enable multiple simultaneous instances, which in turn enables multiple isolated video streams, but this requires much higher CPU loading because of its mesh network architecture.

Of the platforms supporting both audio and video that were tested (AnyMeeting, appear.in, GoToMeeting, Hangouts, ooVoo, Skype, Vconect, Visimeet, VSee), none provided the necessary flexibility with respect to the above criteria. Given the decision to run two separate computer systems handling audio and video separately, a workable solution emerged in which two separate streaming solutions would be deployed, one on each system, in order that the necessary functionality in both domains could be achieved.

Of the a/v platforms tested, VSee was selected to stream video (see Chen 2008). VSee was developed primarily as a vehicle to enable TeleHealth and has formed part of a number of TeleHealth studies and software evaluations (Dorsey et al. 2010; Carbone et

al. 2016; Narasimha et al. 2016). Although not developed for telematic performance, it offered precisely the functionality needed within the video domain of Online Orchestra: (1) it can host up to six video streams; (2) users can control resolution in the upload stream; (3) that resolution can be asymmetrical across the system, meaning clients with lower upload bandwidths can send at lower resolutions than those with higher bandwidths; (4) its stand-alone, multi-client interface is relatively CPU-light in comparison to web-based systems; and (5) individual streams within a multi-node call break out into independent windows, enabling a multi-display desktop.

In the pilot performance, video of the conductor was streamed at 720p, which was sufficient to enable clear and smooth realization of gestures. Given the lower upload bandwidth at other nodes, video was streamed at 480p from Truro, Mullion and the Isles of Scilly. As detailed in Rofe et al. (2017a), in this special issue, participant musicians in the final performance reported no issues with the resolution of the conductor but would have preferred higher resolution streams of other musicians. However, given bandwidth limitations, a lower resolution was a necessary price to pay for a stable video signal.

Whilst VSee provided excellent functionality with respect to video streaming, its audio is not optimized for music, and, as detailed above, participant musicians in working groups prioritized audio quality in terms of overall experience. As such, a more sophisticated audio-streaming platform was needed, and for this, JackTrip provided an excellent solution. Given the significance of high-quality audio in the system design, and the highly customizable nature of JackTrip, it is useful to consider this programme in more detail and how its user settings were optimized for the Online Orchestra pilot performance.

JackTrip

JackTrip is a high-fidelity audio-streaming solution for telematic performance developed by Juan-Pablo Cáceres and Chris Chafe (CCRMA, Stanford University; see Cáceres and Chafe 2008, for JackTrip documentation, and Cáceres 2008, for download). At the time of writing, the application supports uncompressed audio and as many audio channels as the computers and network can handle (Cáceres and Chafe 2010). JackTrip is a Unix command-line program implemented in C++ that uses JACK (Davis 2009) as its audio host, allowing users to connect multiple audio channels from different applications through a low latency server. Through the JACK interface, it is possible to route audio channels to different nodes *via* peer-to-peer connections that can be established with JackTrip. Audio is transmitted using UDP packets across the network (Cáceres and Chafe 2010). When a JackTrip transmission is established, uncompressed audio can be streamed from and to another location, providing that UDP ports are open.

JackTrip was selected over other audio-streaming solutions such as Source-Connect (Sound Elements 2005), eJamming (Kantor et al. 2010) and SoundJack (Carôt 2013) for its ability to handle uncompressed audio, multiple audio channels and several simultaneous peer-to-peer connections. It also provides highly customizable user settings that can be changed depending on available user bandwidth and computer processing power. Amongst the available user settings, JackTrip supports the following options: server or client mode, port number offset, number of channels, queue buffer length, packet redundancy, bit-rate resolution and zero underrun (Cáceres et al. 2009). These

settings have different effects on the network latency, audio quality, processing power and bandwidth use. The network latency in a telematic performance context is the amount of time the audio data take to travel from one location to another. This will depend on the amount of information that is transmitted as well as the speed of the connection. Audio quality will depend on the integrity of the data that are transferred through the network.

UDP losses and JackTrip optimization

Because JackTrip uses UDP to transmit information, some data packets might be lost or delayed depending on the conditions and settings of the transfer. UDP does not provide a way to predict the order in which packets arrive or to know if a packet reaches its destination. If audio data are lost or delayed in the transfer, the quality of the signal is compromised. Packet losses usually translate into audible artefacts in the audio signal – the missing audio data typically will produce clicks/glitches due to significant amplitude change or micro-silences in the signal (Bouillot and Cooperstock 2009). The user settings have to be carefully chosen as they have a direct effect on the amount of data that is processed (affecting computer processing power), transferred (determining bandwidth use) and lost (degrading audio quality).

The latency in the network is affected by the *sample rate* and *buffer size* settings in the JACK server. Higher sample rates and lower buffer sizes will result in shorter network latency, while lower sample rates and higher buffer sizes will result in higher network latency. Buffer size settings are more commonly used to control network latency because sample rates smaller than 44.1kHz are not desirable for high-definition audio. The *queue (-q)* buffer length setting in JackTrip is an easy way of changing the buffer

size while the JACK server is running. This setting will increase the buffer size by the chosen queue value: 'JACK server buffer size setting + JackTrip queue buffer length = total length of buffer' (see Cáceres and Chafe 2008). It is important to understand that the Internet connection speed, bandwidth use and quality of the service all need to be considered while selecting JackTrip settings as these conditions will have a direct relationship to audio quality. Lower quality of service or less available bandwidth may result in more glitches in the audio signal as a consequence of more audio data packets being lost or delayed. In this situation, better audio quality can be achieved by choosing lower *sample rate* and higher *buffer size* settings (Cáceres and Chafe 2008) at the expense of higher network latency.

JackTrip has a redundancy algorithm that is built into the system to diminish the amount of lost or misplaced packets. By adding extra redundant copies of data packets to the network, the algorithm is able to recover misplaced or lost packets. If the original packet is lost, it will be replaced by a copy (see Cáceres and Chafe 2010). The user can select the amount of additional copies of data packets by changing the *redundancy (-r)* setting. It is possible, therefore, to reduce audio glitches by increasing the value for redundancy. However, the bandwidth use will also increase by the redundancy factor since the amount of data in the network will be doubled for each redundant packet (Cáceres and Chafe 2008). The maximum transmission unit (MTU) size in the network also needs to be considered when choosing the length of a buffer and redundancy values, as this directly relates to packet size – if the packet size exceeds the MTU size on the network path, this will result in fragmentation of the audio data.

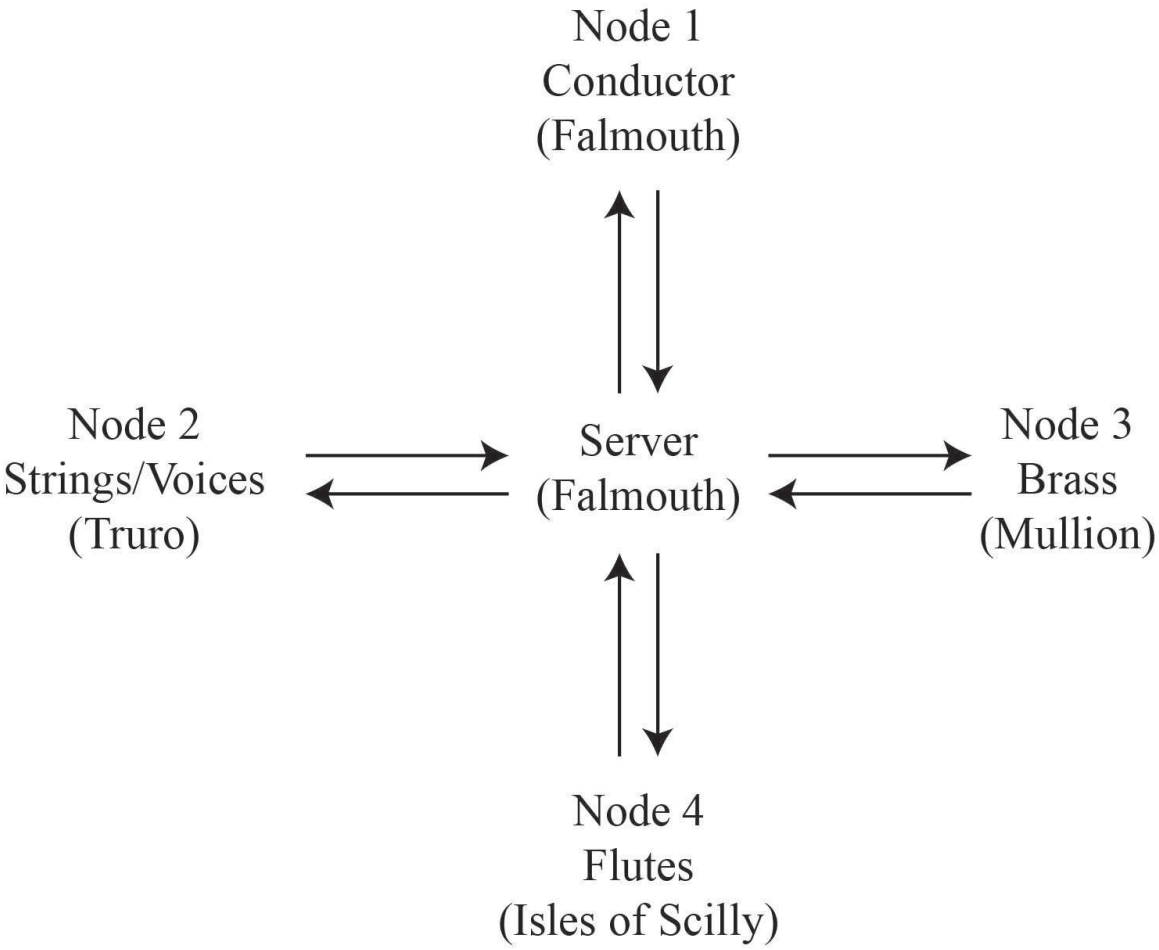
JackTrip provides two different modes to deal with audio glitches caused by missing data packets. When the *zero underrun* setting is off (default), the system will run in wavetable mode and will use the last available packet when new packets are not available – this results in a sound similar to a wavetable synthesizer because of the static timbre produced by the repetition of the packet data. When the setting is on *(-z)*, a packet of zeros will be sent if no new packets are available, resulting in silence. JackTrip can also support more than one peer-to-peer connection through the *port offset (-o)* setting. By using this setting, the user can create multiple connections with different port numbers by offsetting the value from the default UDP port (Cáceres and Chafe 2008).

JackTrip implementation in Online Orchestra

For the Online Orchestra audio-streaming system, an effective way of applying JackTrip's peer-to-peer model to multiple locations was to establish a star topology for the audio transmission. As shown in Figure 1, the audio of the nodes is summed at the central hub (server) and then transmitted to each location; this clients-to-server model allows for custom audio mixes to be made for each location at the server (see Prior et al. 2017). This topology relied on having a fast and good-quality Internet connection at server point but avoided an unnecessarily high number of audio channels at other locations with lower quality connections and less bandwidth. Falmouth University had the best quality and fastest broadband connection of all the locations and therefore functioned simultaneously as the server and conductor's node. To establish a star topology from various peer-to-peer connections, it was necessary to set different port numbers at server point for each individual node. In order to do this, the *port offset*

setting in JackTrip was used: for each added node, the offset value was increased by *–o* 10. The default UDP port in JackTrip is 4464 (Cáceres and Chafe 2008); therefore, it was also important to make sure ports 4464 to 4485 were open for streaming.

Figure 1: Online Orchestra audio network topology.



As described above, the quality of Internet connections and the available bandwidth varied across all locations. For this reason, JackTrip was tested extensively, simulating the network and processing power conditions expected for the performance. For each location, consideration was given to the quality of the connection, the total available bandwidth and the bandwidth needed for handling video. For each peer-to-peer

connection, the effect of different JackTrip settings on network latency, bandwidth use and audio quality was monitored. A network protocol analyser[2] was used to monitor bandwidth use and to analyse packet data (packet loss and destination). The total system latency was analysed using a custom-designed measuring tool that calculates the time an audio signal takes to travel from the server to a location and back (see Rofe and Reuben 2017). The audio quality of the connection was assessed by transmitting a sine wave signal through the system at a stable amplitude and counting the amount of glitches in a ten-minute period. A sound with a stable non-zero amplitude was ideal as the missing audio information would generate an identifiable glitch in the signal. In each location, it was decided that, for each connection, the lowest acceptable audio quality would be one glitch per two minutes. By using these three types of analysis, it was possible to test and adjust different JackTrip settings to arrive at the most suitable configuration for each location.

As described above, it was decided that high resolution audio was a priority; therefore, the JACK server setting for *sample rate* was established at 44.1kHz and the *bit resolution* (*-b*) setting in JackTrip was set to 16 bit. Initially, it was determined that the *number of channels* for each node would be *-n* 2 (stereo transmission). However, after a long period of testing using the three aforementioned analysis tools, it was clear that some compromises had to be made, particularly in locations with lower quality service and less bandwidth. This included having to work with higher network latency in the system and lowering the number of audio channels to mono transmission.

The JACK server *buffer size* and JackTrip *queue* buffer length settings impact network latency and audio quality. After testing, it was determined that the JACK *buffer*

size setting would be set at 64 and the *queue* buffer length in JackTrip at -*q* 48. The *redundancy* setting was also crucial in diminishing the amount of glitches, particularly in locations with low-quality service but with sufficient bandwidth. After testing, it was determined that for the majority of locations, a *redundancy* value of –*r* 2 yielded the best results. This value, however, was not sufficient for the location with the lowest quality service. This connection exhibited a high number of glitches despite having sufficient bandwidth. As part of the assessment of audio quality, JackTrip's *zero underrun* was tested. It was quickly established that the -*z* setting (*zero underrun* on) yielded the best results in all locations, diminishing the disruption of glitches and avoiding sustained artificial wavetable sounds.

Conclusion

Online Orchestra sought to enable a multi-node telematic performance environment that required the streaming of multiple audio and video channels. Working groups during the design phase of Online Orchestra established through participant tests the need for high-quality audio and a visual image of the conductor that was sufficiently clear and stable to facilitate music-making. The unique nature of each location in the pilot performance in terms of network bandwidth and quality further complicated the design solution.

Attempts at a single-system solution proved ineffective, given these requirements: CPU loading on a single machine was too high, and no single piece of software delivered the necessary functionality in terms of user control over audio and video processing and streaming. The final design solution was certainly more complex than would be ideal: a

PC driving video streaming using VSee, a Mac driving audio using JackTrip and Max used on both machines (1) to control network latency and (2) to synchronize audio and video feeds. However, in the absence of custom-designed software that integrates all of this functionality, this solution proved to be highly effective. A dedicated computer driving video through VSee enabled sufficiently high-quality streaming of the conductor that musicians could follow without the disruptive effects of frame drops. At locations with lower upload bandwidth, VSee enables user selection of video resolution and also isolates video streams into independent windows, enabling a multi-display desktop. JackTrip was chosen as the audio-streaming solution as it supports uncompressed audio and has highly customizable user settings such as multiple audio channels, simultaneous peer-to-peer connections and controls to fine-tune network latency, bandwidth use and audio quality. Arriving at acceptable user settings proved challenging for connections of lower quality and less bandwidth. However, with fine tuning, JackTrip enabled an audio environment in which glitching was diminished to less than one glitch every two minutes in all locations during the performance.

As detailed in Rofe et al (2017a), in this special issue, four participant musicians in the pilot performance were interviewed after the concert and asked to comment on their experience. Analysis of interviews shows that no participants commented on the audio-visual quality per se: their suggestions for improvement relate more to details of camera position, audio volume and screen size. This is testament to the feasibility of Online Orchestra's system design. Future research and development are now needed to bring together the functionalities required by Online Orchestra and similar community-

based telematic performance projects, in order to develop a single, integrated audio-visual solution.

References

Source elements (2005), 'Remote record, approve and monitor in real-time' http://source-elements.com/. Accessed 25 January 2016.

Bouillot, N. and Cooperstock, J. R. (2009), 'Challenges and performances of high-fidelity audio streaming for interactive performances', *Proceedings of International Conference of New Interfaces for Musical Expression*, http://www.nime.org/proceedings/2009/nime2009_135.pdf, Pittsburgh, PA, United States. Accessed 24 January 2016.

Cáceres, J. P. (2008), JackTrip, https://github.com/jcacerec/jacktrip. Accessed 24 January 2016.

Cáceres, J. P. and Chafe, C. (2008), 'JackTrip documentation', https://ccrma.stanford.edu/groups/soundwire/software/jacktrip/. Accessed 24 January 2016.

——— (2010), 'JackTrip: Under the hood of an engine for network audio', *Journal of New Music Research*, 39:3, pp. 183–7.

Cáceres, J. P., Chafe, C., Weaver, S. and Dessen, M. (2009), *JackTrip Manual*, https://sites.google.com/site/jacktripdocumentation/home. Accessed 24 January 2016.

Carbone, M., Freschi, C., Mascioli, S., Ferrari, V. and Ferrari, M. (2016), 'A wearable augmented reality platform for telemedicine', *Augmented Reality, Virtual Reality and Computer Graphics*, 9769, pp. 92–100.

Carôt, A. (2013), 'SoundJack: A realtime communication solution', http://www.soundjack.eu/. Accessed 25 January 2016.

Chen, M. (2008), 'VSee: World's largest video telecommunication platform', www.vsee.com. Accessed 1 March 2016.

Davis, P. (2009), 'JACK: Audio connection kit', http://jackaudio.org. Accessed 24 January 2016.

Dorsey, E. R., Deuel, L. M, Voss, T. S., Finnigan, K., George, B. P., Eason, S., Miller, D., Reminick, J. I., Appler, A., Polanowicz, J., Viti, L., Smith, S., Joseph A. and Biglan K. M. (2010), 'Increasing access to specialty care: A pilot, randomized controlled trial of telemedicine for Parkinson's disease', *Movement Disorders*, 25:11, pp. 1652–9.

Fazlul Haque, A. K. M, Raton Mondol, S. I. M. M., Muzahid, A. A. M. and Zahirul Islam, Md. (2013), 'Acoustic echo cancellation for the advancement in telecommunication', *Beyond Limits: International Journal of Advance Innovations, Thoughts & Ideas*, 2:1, pp. 1–7.

Kantor, G., Glueckman, A. J., Redmann, B., Scherrey, B., Dyer, B. and Hagin, B. (2010), 'eJAMMING AUDiiO: Now play and record together live with musicians anywhere in the world', http://www.ejamming.com. Accessed 25 January 2016.

Kolb, D. (1984), *Experiential Learning: Experience as the Source of Learning and Development*, Englewood Cliffs, NJ: Prentice-Hall.

Lamping, U., Sharpe, R. and Warnickle, E. (2015), Wireshark, https://www.wireshark.org/. Accessed 26 January 2016.

Narasimha, S., Agnisarman, S., Chalil Madathil, K., Gramopadhye, A. K., Welch, B. and Mcelligot, J. (2016), 'An investigation of the usability issues of home-based video telemedicine systems with geriatric patients', *Proceedings of the Human Factors and Ergonomics Society 2016 Annual Meeting*, http://journals.sagepub.com/doi/pdf/10.1177/1541931213601412. Accessed 4 January 2017.

Prior, D., Reeder, P., Rofe, M., Biscoe, I. and Murray, S. (2017), 'Designing a system for Online Orchestra: Peripheral equipment', *Journal of Music, Technology and Education*, 10: 2-3, pp. 197-212.

Rofe, M. and Geelhoed, E. (2017), 'Composing for a latency-rich environment', *Journal of Music, Technology and Education*, 10: 2-3, pp. 231-56.

Rofe, M. and Reuben, F. (2017), 'Telematic performance and the challenge of latency', *Journal of Music, Technology and Education*, 10: 2-3, pp. 167-84.

Rofe, M., Geelhoed, E. and Hodsdon, L. (2017a), 'Experiencing Online Orchestra: communities, connections and music-making through telematic performance', *Journal of Music, Technology and Education*, 10: 2-3, pp. 257-76.

Rofe, M., Murray, S. and Parker, W. (2017b), 'Online Orchestra: Connecting remote communities through music', *Journal of Music, Technology and Education*, 10: 2-3, pp. 147-66.

Sun, L., Mkwawa, I. S., Jammeh, E. and Ifeachor, E. (2013), 'Speech compression', in *Guide to Voice and Video over IP*, London: Springer-Verlag, pp. 17–52.

Notes

1. Mac Pro: Quad Core; 3.7GHz; 12GB RAM.

2. *Wireshark* (Lamping et al. 2015) was chosen for its flexibility and level of detail in the analysis of packet data and network traffic.