THE AUDITORY CONSISTENCY IN DISTRIBUTED MUSIC PERFORMANCE: A CONDUCTOR BASED SYNCHRONIZATION

Nicolas Bouillot,
Doctorant en services systèmes pour le multimédia
CNAM-CEDRIC, 192 rue St Martin, 75141 Paris Cedex 03
bouillot@cnam.fr, + 33 1 58 80 85 47

Abstract: With the growth of interactive multimedia streaming on Internet, we expect to provide a tool making remote musicians play together in real-time and across the Internet. However, real-time streaming deals with delays, producing among musicians what we call an auditory inconsistency. As we will show, this inconsistency disables the collective musical practice. In this paper, our conductor driven scheme provides the auditory consistency property among chosen participant of the musical performance. This synchronization scheme hides the network latency to the musicians and enables the distributed collective musical practice.

Résumé : La croissance des systèmes de transmission de flux multimédia interactifs nous laisse envisager la possibilité de fournir à des musiciens géographiquement éloignés un médium permettant de jouer de la musique ensemble en temps réel. Cependant, le streaming de flux multimédia introduit des délais de bout en bout. Parmi les musiciens distants, ces délais provoquent ce que nous appelons une incohérence auditive. Comme nous allons le montrer, cette incohérence empêche les musiciens de jouer de la musique collectivement. Dans cet article, notre mécanisme orienté chef d'orchestre fournit la propriété de cohérence auditive entre différents musiciens choisit. De plus, ce mécanisme cache complètement la latence introduite par le réseau aux musiciens et rend possible le jeu musical collectif distribué.

Key words: Musical interactivity, real-time, PCM audio streaming, synchronisation, distributed system.

Mots clés : Interactivité musicale, temps réel, flux audio PCM, synchronisation, système distribué.

The auditory consistency in distributed music performance: a conductor based synchronization

1 - INTRODUCTION

1.1 - The distributed virtual concert project

This project comes from the collaboration between IRCAM's and CNAM-CEDRIC's research laboratories. It may enable a real-time orchestra of remote musicians.

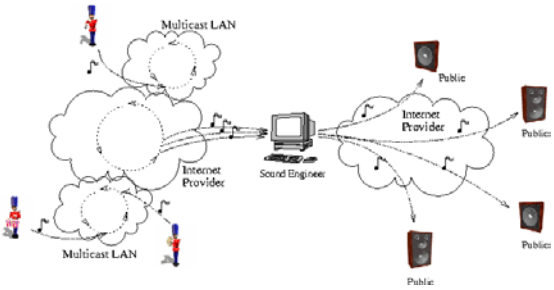

Fig 1. The distributed virtual concert

Figure 1 shows the general architecture of the distributed concert project. Remote musicians play together in real time, hearing each other through PCM audio streams. The different parts of the music (played by others) are heard after synchronization in the side fills of these streams. This allows them to play a real time and collective musical piece. In order to preserve local rhythms and audio quality in the streaming engine, latencies between musicians are kept constant. The public can hear the concert thanks to a sound engineer who performs remote control of the audio streams parameters as volume, localization in space, etc (see locher (2003)). In this way, he controls the streaming engine that achieves a traditional and spatial mixing. Spatial mixing consists in placing sources on a three dimensional space (then the sound is multi-channel). We focus here on the synchronization scheme among musicians and particularly on the way to play music according to the network constraints.

1.2 - The distributed auditory consistency problem

We will first show some differences between the traditional music practice in a band, i.e. when all musicians are in the same room, and the musical practice across the network. Then, we will define the distributed auditory consistency property.

Traditionally, the musical interaction is helped by various visual signs and conventions predetermined on the piece of music played (such as a sequence of chords with a theme for a piece of Jazz). At the same time, the musical contents added by each musician inform the others on the possible evolution of the piece. For example, groups of percussions sometime use rhythmic sentences to call each other to change rhythms. These kinds of interactions are possible due to the instantaneous hearing of the sound produced by each musician. For instance, if a bassist and a drummer play rhythms at the same time, everybody in the room will hear them together. We will next differentiate the direct sound produced by each musician and the side fills given by the environment. This allows us to consider the environment latency as the latency between the direct sound and the side fills. In a room, the environment latency exists due to the sound transmission delay in the air but it is not audible. In our work, we consider 20ms as the auditory delay perception threshold. As this latency cannot be heard musicians could synchronously feel the time as structured musical units (bars, quarter notes, triplets…). This common musical language gives them the opportunity to play synchronously their own part on the global performance. The environment keeps then the property we call the consistent auditory restitution. We can thus say that playing music with interaction is possible in a synchronous environment.

In our context of distributed “Live” music where the musicians are physically remote, networks and operating systems are asynchronous with audible and different latencies. Sound card, network, drivers and application provision the global latency. Let us show now why latencies impact on the consistent auditory restitution. Consider on figure 2 two musicians (Alice and Bob) who want to play across a network with perceptible delays.

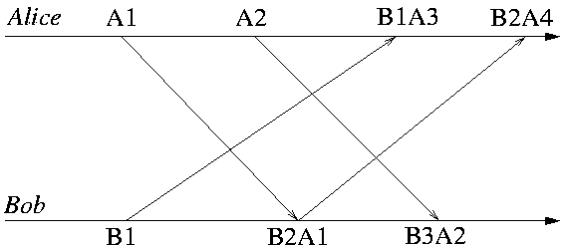

Fig 2. The auditory consistence problem

First, Alice is playing a sound at the time we call "A1". When this sound is returned to Bob at "B2", he is playing another sound. At this time, Bob hears the sounds «A1» and «B2» simultaneously. But he has to send to Alice his sound produced at "B2". It is returned to Alice when she is producing the "A4" sound. This makes Alice hears the sound "A4" and "B2" simultaneously. As "B2" is mixed with different sounds for Bob and Alice, we can say that they are not living the same experience of the musical piece. This is what we call an inconsistent auditory perception of the music. It is then difficult to get a common rhythmical language. Suppose that Alice is a drummer and Bob is a bassist. If Bob feels himself rhythmically synchronized with the drums pattern, Alice will feel Bob out of the rhythm. Therefore she will adjust her rhythm in order to be synchronous with Bob but that will cause a de-synchronization at Bob's side-fills, with a snowball effect.

1.3 - Solutions

To avoid this problem, we propose to add a mechanism that provides the consistent restitution. We distinguish two kinds of solutions. Bouillot (2003) has described the first one. This solution consists in adding consensually delays in each musician's side fills, making each musician hears himself with a constant latency. There are situations where musicians are able to play with some delays, like organists interacting with a choir in church. But the musicians have to learn to play with this latency, limiting the spectrum of musical style possible. The second solution is presented in this paper (section 3) and it is called the "conductor synchronization scheme". This solution gives to the musicians a synchronous side-fill with a zero-local latency perception. As a side effect, each site can hear only his own produced stream with the one coming from the conductor.

On section 2, a state of the art is given with short comparisons with our synchronization scheme. We will next present our "conductor" based solution. Then we will present the current status of the prototype in section 4 and to finish, we will conclude in section 5.

2 - STATE OF THE ART

Several experiments of real time multimedia performances on Internet have already been achieved. Most of them use the MIDI standard (Musical Instrument Digital Interface). In Eliens (1997), latency between the production and the consumption of sounds is not designed to be constant so, variations are perceived while hearing. In the piano lesson system of Young (1999) and Fujinaga, a single musician is teaching while the pupils are listening. Then, there is no need here to play in a synchronized way. The team of Goto (1996, 1997) developed VirJa, a tool of virtual session of Jazz in which someone can play music with two processors, but there is no mechanism of synchronization of streams specified. An experiment of remote music was born based on RMCP (Remote Music Control Protocol) used in VirJa: *OpenRemoteGIG* (Goto (2002)) allows playing with remote musicians with MIDI streams and constant latency (an entire chord sequence). In this system, the musicians from various places will hear all the musicians but none of them hears the same resulting music. The example describes in Goto (1997) shows us this shift: the player improvises while listening to other sounds delayed by the constant period of the repetitive chord progression (a 12-bar blues chord progression in the text). Because the progression is repetitive, the delayed performance can fit the chords. In this way, musicians can play in a synchronous way. They will hear themselves instantaneously but simultaneously with the other delayed by the 12-bar latency. We can say that this work is done without a distributed auditory consistence and with a big latency reducing the interactivity between musicians. Despite the interest that such performances represents, the MIDI protocol allows to put aside some aspects of the transmission of PCM audio streams. MIDI takes less bandwidth thanks to its descriptive format but it also decreases the field of the transported sounds. Thus, it is difficult to make a direct analogy

with the constraints raised by PCM audio streams. Among all of these papers no one deals with the distributed auditory consistency problem or something equivalent.

Xu (2000) and Cooperstock (2001) have tested the transport in real time of PCM audio streams created on fly. The authors used a recording studio to sample a performance that took place in another country. The musicians were located at the same place, raising no problem on interactions between musicians.

The closest work done by another team is the "SoundWire" project. They experienced the streaming of professional-quality audio across the Internet2 network (see Chafe (2000)). Few LAN and WAN experiments were deployed in the United States, where round trip time was about a 2 factor slower than the speed of the light. In these experiments, the main musical goal was to see how musicians could play with latency introduced by the network. As these latencies were closed to the one perceptible by the ear, the musicians could progress together in the musical piece but not rhythmically synchronous (see SoundWire (2002) Live WAN test audio examples). The reason is that they did not include any musical synchronization in their streaming engine. In comparison with our experimental context, we are working on Internet with bigger round trip times. But even with the smallest delays possible (the light speed), "*the theoretical round trip time across USA and back is approximately 40msec*". We take 20msec as the perceptible latency between two clicks delayed. This threshold is stated also by simulated delay tests from Schuett (2002) audio examples. We hear that with a 20ms delay, musicians can play together but with a 30ms delay, we can hear a difference between the side-fills of the two musicians. This let us imagine perceptible delays with bigger distance. Schuett's thesis (2002) tries to "*define the level of delay at which effective real-time musical collaboration shifts from possible to impossible*". The author talks about a leader-follower relationship between musicians. However, this relationship is different from the one presented in our work. In the Schuett's study, two performers are subject to symmetric delays added with a digital mixing console. He remarks that the musicians can play together if one of them follows rhythmically the other one. This relationship is based on the musicians' behavior. We point out that in our work, we synchronize the audio streams thank to the streaming engine and the time stamping. Remote performers are then subject to delays that could be asymmetric. This is here the synchronization scheme which is conductor driven.

3 – THE SYNCHRONIZATION SCHEME

3.1 – Justification of such an architecture

Playing music in a band is itself a hard task. Then we want to minimize the difficulties added by the streaming engine. Playing music with remote musicians avoids social interactions as looking to the other musicians, hearing them instantaneously… In order to provide an easier practice of the remote playing, we believe that musicians have to be in a synchronous musical environment. This immersion is possible if we include synchronization mechanisms in each local musician's side-fills. We thus have to consider things like: "what does each musician should hear?" In Bouillot (2003), we supposed that each musician hears each other to keep a musical interaction. But it depends on the kind of music, especially when the entire band must communicate. This approach provides a good interactivity among the musicians but the side effect of keeping each stream in the side-fills is that the local sound is delayed too. The musical practice becomes then more difficult, but still possible. The idea we present here is quite different: the musicians will hear only two streams, the one from the conductor and the one from themselves. This is acceptable with some kinds of music where there is a little interaction between musicians. This is the case of many written music where the most important interaction is the one with the conductor. Symphony is a good example: in a symphonic orchestra, musicians are looking at the conductor and at the same time they are hearing their neighbors to play music. We can find some other examples as the Mugam of Azerbaijan, one of the liveliest Orient's classical musical styles. Bois (1993) explains that this music requires *« a singer, […] and two musicians (sazande) playing on the luth (târ) and the spike-fiddle (kemânche). The târ converses directly with the sung phrases, while the kemânche sometimes sustains the singer*

sometimes the târ player». Then, the singer needs to hear the târ player. The târ player needs to hear the singer but the kemânche player stream is not necessary at the other's side fills to play in a synchronous way. At the question "what do musicians hear? » our conductor synchronization scheme answers that each musician must hear his own produced sound and the conductor sound. In this way, we can make easier the distributed musical practice. To compare that with our previous work, this avoids the local latency and also the interaction between musicians. With this conductor driven scheme, we can distribute the Mugam's musicians by associating the târ player with the singer as the conductor and the kemânche player as a musician. Thus the kemânche player can play without local latency.

3.2 - Description

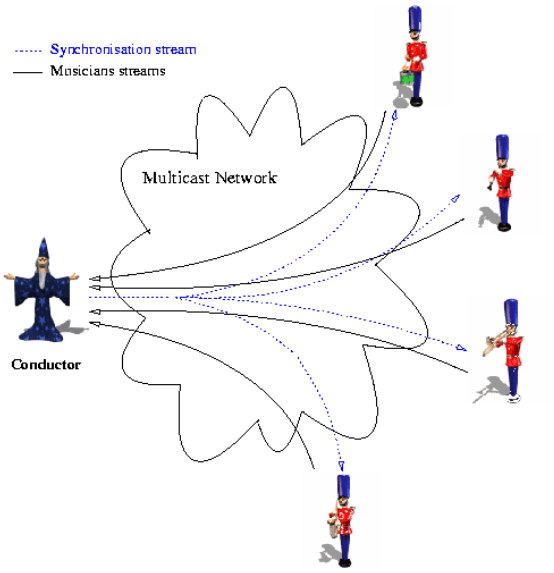

Fig 3. The conductor architecture

The idea behind the conductor driven architecture is quite simple. The conductor broadcast his own sound, for instance a beat, time stamped with the local sound card's clock. Helped by the streaming engine, the musicians can hear the sound sent by the conductor. It is easy now for all of them to play synchronously with the beat heard on the side-fills. As musicians got a global reference (the conductor's stream), it is able to mix all the streams in a synchronous one. In order to do that, each musician's sample are time stamped with the stamp of the conductor's sample perceived simultaneously. The conductor (or someone else) could receive each stream and mix them in the synchronous way by playing simultaneously the samples having the same timestamp. To keep interactivity in the musical piece, the conductor can mix his own beat with his voice to indicate to musicians how the musical piece will change.

Let's call n the number of musicians, $m_i (0 \le i < n)$ a musician, c the conductor, t_x the x's local sound card clock, s_x^j a sample produced by x at $t_x = j$ and $t_x(s_Y^j)$ the time when s_y^j is processed (produced, consumed or both) by t_x. If $x = y$, then s_y^j is produced by t_x. We can thus say that say $t_x(s_x^j) = j$ and but that $t_x(s_y^j) \neq j$ (the sample comes from another source. So it is here stamped by another clock). We call now $e_{m_i}(s_{m_i}^j)$ the timestamp associated to the sample $s_{m_i}^j$ (in the musician m_i's output stream). According to the PCM audio streams semantic, sound card sample production and time stamping respect local order. Timestamps and sample produced at the sound card's clock are thus incremented, i.e.

$$t_x(s_x^j) + 1 == t_x(s_x^{j+1})$$

and

$$e_{m_i}(s_{m_i}^j) + 1 == e_{m_i}(s_{m_i}^{j+1})$$

These considerations suppose that the remote sound cards are running at the same sample rate. If not, a conversion can be added.

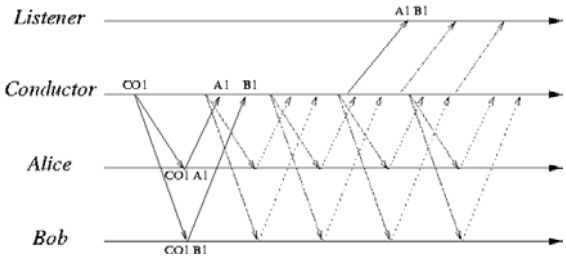

Fig. 4. The streams in the conductor driven scheme

As Figure 4 shows, in the conductor architecture, the conductor broadcast his own sound. Each musician receives thus all the s_c^j samples as a stream (Figure 4 shows us the sample CO1 broadcasted to the musicians). With that stream, each musician gets samples

(with the timestamps associated) to play out. The main objective for them is now to timestamp their own stream with the values that allow the conductor to mix them and get a resulting mix synchronized. Locally to m_i, the input stream coming from the m_i's instrument and the stream coming from the conductor are processed by the same clock (t_{m_i}). Then for each musician m_i, for each locally produced sample, the timestamp associated is $e_{m_i}(s^k_{m_i})$ where:

$$e_{m_i}(s^k_{m_i}) = j \text{ with } t_{m_i}(s^k_{m_i}) = t_{m_i}(s^j_c)$$

On figure 4, the A1 and B1 timestamp are set to CO1. After the timestamp calculation, each musician will send his produced sample with the associated timestamps to the conductor, who will mix the streams as:

$$t_c(s^{k_0}_{m_0}) = ... = t_c(s^{k_{n-1}}_{m_{n-1}})$$

$$\text{where } e_{m_0}(s^{k_0}_{m_0}) = ... = e_{m_{n-1}}(s^{k_{n-1}}_{m_{n-1}})$$

The conductor mixes these samples to hear them or to send the mix to listeners (Figure 4).

If musicians play synchronously with the conductor stream (the global reference), the mix will be synchronous. To keep an interaction between the conductor and the musicians, the conductor has to hear the resulting music, which includes a rhythmic de-synchronization between his side-fills and his direct sound. Therefore, he can send a pre-recorded sound (a beat for example) and hear the mix only, avoiding the problem. Another solution is to take two people as the conductor: one plays the synchronization stream and the other one hears the mix to give the indications. In this way, we expect a minimal latency from musicians to the conductor.

We can notice now that musicians do not perceive the network latency: they are just playing with a received stream. However, the conductor can hear all of them and perform the interactivity by giving them indications.

3.3 - Bandwidth requirement

A mono PCM stream with a 44100Hz sample rate sent with RTP (Schultzrinne (1998)) represents 0,7Mb/s. As we use IP multicast in our streaming engine, the conductor need to send 1 synchronization stream and need to receive n streams from the musicians. The conductor thus needs a 0,7Mb/s upload bandwidth and a $0,7 \times n$ download bandwidth. Each musician needs 0,7Mb/s as upload bandwidth (his own produced sound) and 0,7Mb/s as download bandwidth (the conductor stream). Nowadays, these bandwidth requirements are too big to place the conductor on a DSL connection at home but not to play through academic Internet providers, which is our actual experimentation context.

3.4 – Combination with previous work

The conductor architecture fits well with the music where there are little interactions between musicians. It provides a null latency perception to the musicians. Bouillot (2003) focus on interactions between all the musicians and includes latency in the perception of the local produced sound. These two kinds of interactions have advantages and disadvantages but are not incompatible. We can take the advantages of both, thanks to an appropriate distribution of the roles between each musician in a band, according to the expected interactions. For example, the resulting music of a group of musicians that interacts with the self-synchronization algorithm (see Bouillot (2003)) music could be the synchronization stream in the conductor architecture. The other musicians can then play with this synchronization steam in a "conductor" oriented way. Therefore, by coupling both synchronizations, the solution to the auditory consistence problem can be generalized with two levels of added difficulties at the musicians' instrumental practice.

4 – CURRENT STATUS

We have not implemented yet the conductor driven scheme in our prototype. However, as we will show, our streaming engine is modular and can be easily extended with the conductor driven scheme.

4.1 – The Streaming engine

We developed the streaming engine as part of the jMax (Déchelle (2000)) visual language. Figure 5 shows a jMax patch example. This

visual language works with a message oriented semantic. For instance, a modification on the slider will send an integer to the division object, which will process the division by 127. By implementing reception and emission of audio streams as a function of this language, we can easily route the different audio streams, as generated music, microphone input stream, speakers output stream or received streams. jMax allows us to process sound in real-time and gives us a modular approach to configure the streaming engine.

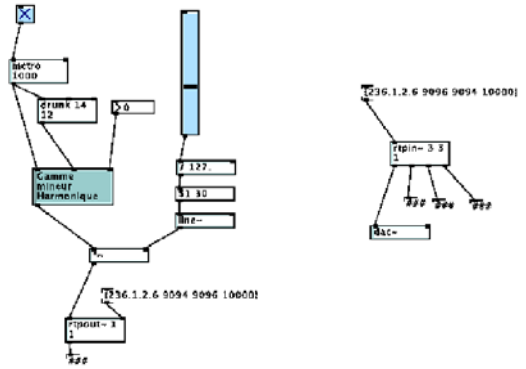

Fig 5. The rtp objects in a jMax visual program

In our first prototype, the reception of audio streams has been achieved with the *rtpin* object developed in jMax (figure 5) and the emission in the *rtpout* object. The sound samples are produced at a constant rate of 44100Hz and consumed by the *rtpin* objects (at the same rate). The transport of audio streams is done using the RTP protocol (Schulzrinne (1998)), through the RTP library called UCL Common Code Library version 1.2.8 developed by the Computer Science Department of the University London College University. Each *rtpout* object stamps the samples in the RTP's timestamp field. The timestamp is incremented per sample. Helped by the RTP's *ssrc* field, we get the music source identification. Each *rtpin* object consumes simultaneously a sample coming from each *rtpout* remote object.

4.2 – Implementing synchronization

In our prototypes, we deployed the synchronization described in Bouillot (2003). We tested successfully this kind of synchronization on LAN (jMax 2.5.1) and on MAN (with a second prototype deployed on jMax 4.1) where the musicians played a distributed Blues in a synchronous way with a synchronized start. Drums/bass were located at the IRCAM center and guitar/saxophone at the CNAM University with a distance of one kilometer across two Internet providers.

In our second prototype we took only one object for both reception and emission. With this implementation, we have local sound card clock, time stamping, streaming and a shared space to read and write samples on the sound cards. Then, we can easily implement our conductor driven synchronization scheme.

5 - CONCLUSION AND FUTURE WORKS

We have seen that the major interest here is to provide a way for distributed musicians to play without any network latency perception. In all the other distributed musical systems, the instrumental playing deals with delays, including an alteration in the instrumental practice. Although our conductor driven architecture focuses the musicians on the conductor, it makes the distributed musical practice easier for musicians and keeps the auditory consistency among people interacting together. We can then provide a more generic solution to the distributed way of playing music with a combination of the conductor architecture and the self-synchronization algorithm described in a previous work (see Bouillot (2003)). This new solution would make the system's architect define interaction between musicians, allowing each of them to one of both synchronizations schemes discussed before. However, this combination introduces some alterations on the traditional way of playing music. This introduces two kinds of musical experimentations. The first one is to try to project existing kinds of music on our solution. The second is to work with composers and musicologists to develop a new kind of musical interaction, which would be aware of these synchronization schemes. As a side effect, the solution to the auditory consistency problem could fit well with some distributed virtual reality problems.

Added to the developments, we will achieve tests with bigger distances with more sites. The work of experimentation will be supplemented by a network provisioning. The guarantee of constant latency between the musicians and the conductor has a basically statistical nature.

In the future, it will be necessary to take account of the losses and to choose a strategy

among the mechanisms presented in Perkins (1998), Bolot (1999) and Rosenberg (1998) or in the literature where the main challenge is to compensate losses without retransmissions.

The last work in progress is the skew existing between the clocks of the different sound cards. This skew occurs between the emission time stamping clock (the sound card) and the reception clock. The problem is a many-to-many one. Orion (2000), Akester (2002) and Fober (2002) present a solution but in a one-to-one context.

However our priority is to make tests on larger network as MAN and WAN. On one hand, this will allow to dimension our prototype and on the other hand, to determine more precisely the kind of interactions that we will provide to the remote musicians.

REFERENCE

Akester R., Hailes S. (2002), "A New Audio Skew Detection and Correction Algorithm". In proceedings of the *International Computer Music Conference*, ICMC (2002).

Bois Pierre (1993), "The Mugam of Azerbaïdjan". In *Sakine Ismaïlova, Anthologie du Mugam d'Azerbaïdjan, Vol. 5 booklet*. CD n. w260049, INEDIT collection. Maison des Cultures du Monde

Bolot J.C., Fosse-Parisis S., Towsley D. (1999), "Adaptive FEC-Based Error Control for Internet Telephony". *INFOCOM*, vol 3, pages 1453-1460 (1999).

Bouillot N. (2003), "Un algorithme d'auto synchronisation distribuée de flux audio dans le concert virtuel réparti". Proceedings of the *conférence française sur les systèmes d'exploitation* CFSE'03, La Colle Sur Loup, France (2003).

Chafe C., Wilson S., Leistikow R., Chisholm D., Scavone G. (2000), "A Simplified approach to high quality music and sound over IP". *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00),* Verona, Italy, December 7-9, 2000

Cooperstock J., Spackman S. (2001), "The Recording Studio That Spanned a Continent". *IEEE International Conference on Web Delivering of Music*, WEDELMUSIC, Florence Italie (2001).

Déchelle, F., Borghesi, R., Orio, N., Schnell, N., « The jMax environment: an overview of new features », *ICMC: International Computer Music Conference*, Allemagne, 2000.

Diot P., Huitema C., Turletti T. (1995), "Multimedia Applications should be Adaptive". Proc. *HPCS'95*, Mystic (CN) (1995).

Eliëns A., van Welie M., van Ossenbruggen J., Schönhage B. (1997), "Jamming (on) the Web". Proceedings of *WWW6* (1997).

Fober D., Orlarey Y., Letz S. (2002), "Clock Skew Compensation over a High Latency Network". Proceedings of the *International Computer Music Conference*, 548-552 (2002).

Goto M., Hidaka I., Matsumoto H., Kuroda Y., Muraoka Y. (1996), "A Jazz Session System for Interplay among All Players". *ICMC* Proceedings, pages 346-349 (1996).

Goto M., Neyama R. (2002), "Open RemoteGIG: An Open-to-the-public Distributed Session System Overcoming Network Latency". *IPSJ* JOURNAL, vol 43, pages 299-309, IN JAPANESE (2002).

Goto M., Neyama R., Muraoka Y. (1997), "RMCP: Remote Music Control Protocol, design and applications". *ICMC* Proceedings, pages 446-449 (1997).

Locher H-N., Bouillot N, Becquet E., Dechelle F., Gressier-Soudan E. (2003), "Monitoring the Distributed Virtual Orchestra with a CORBA based Object Oriented Real-Time Data Distribution Service". In proceedings *DOA'03 International Symposium on Distributed Objects and Applications*. Catagna, Italy. November., 2003.

Orion, Hodson, Colin (2000), "Skew Detection and Compensation for Internet Audio Applications". (2000).

Perkins C., Hodson O., Hardman V. (1998), "A survey of packet-loss recovery techniques for streaming audio". *IEEE Network Magazine* (1998).

Rosenberg J., Schulzrinne H. (1998), "An RTP Payload Format for Generic Forward

Error Correction". *Internet-Draft draft-ietf-avt-fec-03.txt* (work in progress) (1998).

Schuett N. (2002), "The effect of latency in ensemble performance". Thesis, CCRMA, department of music. Stanford University.

Schulzrinne, Casner, Frederick, Jacobson (1998), "RTP: A Transport Protocol for Real-Time Applications". RFC 1889 (1998).

SoundWire home page (2002), http://ccrma-www.stanford.edu/groups/soundwire/

Xu A., Cooperstock J. (2000), "Real-Time Streaming of Multichannel Audio Data over Internet". *AES 108th convension*, Paris (2000).

Young J.P. ,Fujinaga I. (1999), " Piano master classes via the Internet ". Proceedings of the *International Computer Music Conference*, pages 135-137 (1999).