# Une architecture pour le jeu musical virtuel réparti avec jMax et RTP

**Bouillot Nicolas**
**CEDRIC-CNAM**
**292 rue St Martin**
**74141 Paris cedex 03 France**
**bouillot@cnam.fr**

**Résumé :** La croissance des systèmes de transmission de flux multimédia tels l'audio-conférence et la téléphonie sur IP nous laisse envisager la possibilité de fournir à des musiciens physiquement éloignés un médium permettant de jouer de la musique ensemble en temps réel. Cependant, la latence indéterministe introduite par le réseau empêche l'écoute mutuelle et instantanée entre les musiciens distants.
Nous proposons donc dans ce document une architecture de communication pour ces musiciens basée sur une synchronisation implicite effectuée grâce au protocole RTP.
**Mots Clés :** Réseaux, systèmes distribués, architecture, musique, synchronisation.

## 1. INTRODUCTION

Dans un contexte de musique « Live », l'interaction musicale se fait par les différents signes visuels et les conventions prédéterminées sur le morceau de musique joué (comme par exemple une suite d'accords avec un thème pour un morceau de Jazz). En même temps, le contenu musical ajouté par chaque musicien informe les autres sur l'évolution possible du morceau. C'est le cas par exemple des ensembles de percussions qui fonctionnent en phrases d'appels rythmiques. Toutes ces interactions sont possibles car les musiciens évoluent dans un même environnement (la scène, la pièce...) qui leur permet en particulier de s'entendre tous en même temps de façon synchronisée. Dans notre contexte de musique « Live » distribuée où les musiciens sont physiquement éloignés, les supports technologiques tels réseaux et systèmes d'exploitation sont asynchrones. Bien que nous fournissons, d'un point de vu application, une latence constante, les délais de transmission des flux audio seront généralement perceptibles à l'oreille humaine. Le problème étant de fournir aux musiciens une perception cohérente du mixage des différents flux, nous avons imaginé un type d'interaction basé sur le principe d'un chef d'orchestre (ou meneur) qui synchronise les musiciens entre eux par l'intermédiaire du protocole RTP [schulzrinne, 1998]. Pour reproduire au plus près les conditions d'une interaction musicale bilatérale, chaque musicien doit avoir une perception identique du mixage final des flux, impliquant que chaque musicien entende son propre retour avec un délai. Dans [Bouillot, 2003], nous proposons un mécanisme de synchronisation fournissant une écoute mutuelle cohérente et fournissant donc l'interaction bilatérale, mais avec une latence perceptible par les musiciens. Cette contrainte de cohérence mutuelle n'est pas toujours strictement nécessaire. Comme nous allons le voir au paragraphe 3, il n'est pas rare que les musiciens aient une interaction unilatérale. Nous proposons donc dans ce document un mécanisme pour ce type d'interaction cachant complètement la latence aux musiciens, rendant le jeu musical virtuel réparti plus abordable que dans [Bouillot, 2003]. En effet, chacun d'entre eux jouera directement sur ce qu'il entend comme retour, et sans aucun retard.

RTP est un protocole créé pour fournir les propriétés de séquencement temporel, de synchronisation intra-média et inter-média et enfin de présentation. Il est utilisé habituellement comme protocole de transport temps réel de données multimédia. C'est le cas par exemple des systèmes de conférence distribuée, tels que vic [McCanne, 1995] et rat [Hodson, 1999] où RTP est utilisé afin de rendre les débits des flux audio et vidéo adaptatifs à l'état du réseau [Diot, 1995]. De plus, l'aspect présentation de RTP permet aux applications d'introduire des paquets réparateurs d'erreurs dans le flux multimédia [Perkins, 1998], [Bolot, 1999], [Rosenberg, 1998].

Contrairement à une application de conférence sur Internet, l'orchestre réparti doit fonctionner avec un signal de très bonne qualité (flux audio PCM[1]). Le confort d'écoute est crucial, autant pour les musiciens que pour le public. Ce choix rend l'orchestre virtuel actuellement possible sur réseau local ou métropolitain mais non applicable à un réseau à bande passante limité. Nous pouvons cependant envisager dans le futur de diffuser la performance distribuée sur Internet en flux compressés.

Plusieurs expériences de performances multimédia temps réel sur réseau ont déjà été expérimentées. Plusieurs d'entre elles [Eliens, 1997], [Goto, 1996], [Goto, 1997], [Young, 1999] utilisent la norme MIDI [2]. Dans ces articles, aucun mécanisme de synchronisation

[1] Pulse Code Modulation
[2] Musical Instrument Digital Interface

entre les musiciens n'est mentionné. Seul *Open RemoteGIG* [Goto, 2002] permet de jouer avec des délais constants modulo le tempo et la métrique du morceau (les musiciens jouent sur un motif, une grille de Blues, mais les autres ne l'entendront que lorsque le motif se répétera). Cela montre que les musiciens ne vivent pas la même performance, puisqu'ils n'entendent pas tous le même résultat sonore. L'exemple décrit dans [Goto, 1997] nous montre ce décalage: pendant que le joueur joue et s'entend au deuxième motif d'un morceau, il entend ce qu'ont fait les autres au premier.

Malgré l'intérêt que peut représenter de telles performances, le protocole MIDI permet de mettre de côté certains aspects de la transmission de flux audio. Le format descriptif de la note en MIDI permet d'économiser de la bande passante. Mais il diminue aussi le domaine des sons transportables. Il est donc difficile de faire une analogie directe avec les contraintes imposées par les flux audio PCM.

Le transport de flux audio PCM temps réel a été expérimenté dans [Xu, 2000], [Cooperstock, 2001], [Puckette, 1999]. La synchronisation de la musique jouée par des musiciens distribués n'est spécifiée dans aucun de ces systèmes.

A notre connaissance, dans le domaine de l'informatique musicale en réseau, il n'existe pas de travail équivalent à ce que nous proposons dans ce papier. De plus, notre utilisation de la propriété de séquencement temporel de RTP est unique. Elle servira en effet à l'application de l'orchestre virtuel réparti pour que chaque musicien se synchronise sur le chef d'orchestre. Cela montre une utilisation nouvelle du protocole RTP: la synchronisation au service de l'interactivité musicale.

Le paragraphe 2 présente le projet du concert virtuel réparti, le paragraphe 3 l'architecture « chef d'orchestre », le paragraphe 4 le prototype du projet. Enfin, le paragraphe 5 conclut.

## 2. LE CONCERT VIRTUEL RÉPARTI

La conjonction des travaux des laboratoires de recherche de l'IRCAM et du CNAM-CEDRIC doit permettre la mise en place d'un orchestre virtuellement réparti avec des musiciens physiquement éloignés. Le CNAM-CEDRIC est chargé de la conception d'une architecture système distribuée pour supporter cette métaphore de l'orchestre virtuel.

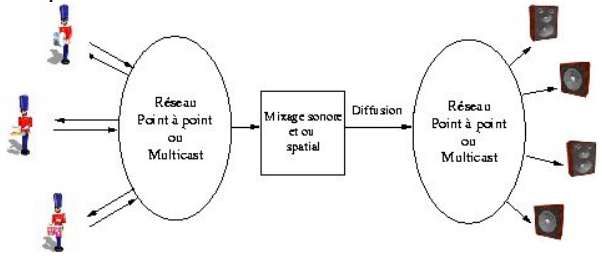


**Figure 1 :** Le concert réparti, une architecture générale

La figure 1 présente l'architecture générale du projet. Les musiciens sont physiquement répartis mais doivent jouer « virtuellement ensemble » en temps réel. Le retour ainsi que l'écoute des autres musiciens se fait grâce à un mécanisme de synchronisation des flux audio développé comme élément de la librairie du logiciel jMax[3]. Cette synchronisation suppose une latence constante au niveau de la restitution du son. Elle peut cependant prendre plusieurs formes, la synchronisation par l'intermédiaire d'un chef d'orchestre est l'une d'elles. Afin de permettre à un public d'assister au concert virtuel, un ingénieur du son collectera les flux audio des musiciens pour faire le mixage sonore et spatial de la scène. Le mixage spatial consiste à positionner les sources sonores dans un espace à trois dimensions. Afin de permettre ce type de mixage, le son transporté sera du son multi-canaux[4]. Le mixage pourra se faire soit à la source directement par un contrôle distant des périphériques des musiciens, soit par un traitement des flux après réception. Les travaux en cours sur le contrôle distant des périphériques utilisent l'implémentation OpenTaz[5] [Becquet, 2001] de la norme TASE 2.0 [tase, 1996] qui spécifie un protocole de gestion de données temps réel.

## 3. UNE ARCHITECTURE « CHEF D'ORCHESTRE »

La principale difficulté dans le système de jeu musical virtuel distribué en temps réel est la latence indéterministe introduite par les réseaux. En effet, bien qu'au niveau application nous rendons cette latence constante, dans [Bouillot, 2003] chaque musicien devra jouer sur de la musique décalée temporellement par rapport à ce qu'il joue lui-même. La solution que nous proposons ici est d'annuler cette latence grâce à une réduction des contraintes d'interactivité entre certains musiciens, car la structure musicale le permet. C'est le cas par exemple de formations intégrant un chanteur ou une chanteuse, où certains instrumentistes ont pour fonction principale d'accompagner celui ou celle-ci en contre chant. Nous pouvons citer par exemple l'accompagnement de la flûte dans la tradition vocale du style Tembang de la région de Sunda à Java Ouest, ou encore le soutient de la vièle Kamânche dans la tradition du Mugam D'Azerbaïdjan. Ce qui est intéressant, c'est que l'interaction peut être considérée ici comme unilatérale, le chant n'ayant pas besoin d'entendre l'accompagnement du contre chant de façon synchronisé avec son propre retour. Pour ce type de situations, le jeu musical virtuel réparti peut se faire sans aucune perception des délais du réseau (sauf pour le chef d'orchestre, s'il désire entendre le résultat final de la performance). En effet, le musicien qui

[3] jMax [Dechelle, 2000] est un environnement de programmation visuelle pour des applications musicales et multimédia temps réel.

[4] Permettant la reproduction d'un effet tridimensionnel (positionnement dans l'espace du son)

[5] http://savannah.nongnu.org/projects/opentaz. Mars 2003

accompagne peut jouer en même temps qu'il entend le flux contenant le chant. Son jeu se fera donc sans décalage entre le moment où il joue et le moment où il s'entend. Pour atteindre un tel objectif, nous allons introduire dans l'application une synchronisation basée sur le séquencement temporel fourni par le protocole RTP.

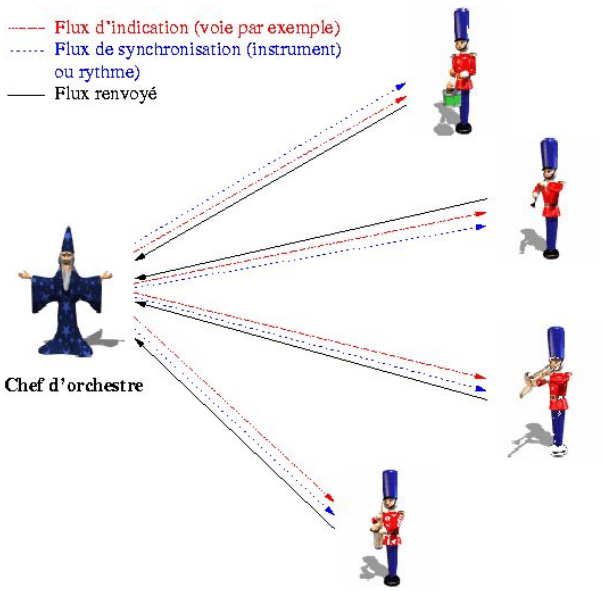


**Figure 2 :** L'architecture « chef d'orchestre »

Pour un système à $n$ musiciens, notons $m_i (0<i<n)$ un musicien et $c$ le chef d'orchestre. Notons maintenant $e_{m_i}(c,t)$ l'estampille du chef d'orchestre correspondant au son joué au temps $t$ par celui-ci et $e_{m_i}(m_i,t)$ l'estampille associée au son joué par le musicien $m_i$ au temps $t$. Ces estampilles sont incrémentées de un pour chaque échantillon sonore (échantillonné en PCM). Il y a donc une notion d'ordre total dans les estampilles, $e_{m_i}(m_i,t+1)$ est l'estampille suivante de $e_{m_i}(m_i,t)$ et $e_{m_i}(m_i,t+1)=e_{m_i}(m_{i,t})+1$. Idem pour le chef d'orchestre. Les cartes son doivent alors être réglées sur la même valeur de quantification et la même fréquence d'échantillonnage, ainsi la consommation et la production de sons seront synchrones sur les différents sites. L'estampillage utilisé par l'application est celui de RTP (champ *timestamp*). Nous pouvons maintenant imaginer la stratégie suivante(figure 2):

Le chef d'orchestre va jouer le rôle de métronome pour le groupe. Pour cela, il va transmettre un flux audio (contenant un rythme par exemple) dans lequel chaque échantillon est numéroté de façon incrémentale. Chaque musicien va recevoir ce flux avec une latence constante. Il pourra alors jouer sa musique en même temps qu'il entendra le son du chef d'orchestre.

Pour effectuer la synchronisation implicite, l'application du musicien va envoyer le son joué par celui-ci avec l'estampillage tel que:

$$\forall t\left(e_{m_i}(c,t)=e_{m_i}(m_i,t)\right) \qquad (1)$$

Autrement dit, lorsqu'un échantillon est numéroté pour être renvoyé, il est numéroté avec l'estampille du son issu du chef d'orchestre joué au même moment. Le jeu des différents musiciens se synchronise implicitement car ils prennent une référence commune: le signal envoyé par le chef d'orchestre.

A la réception des flux venant des musiciens, il suffira au chef d'orchestre (ou autre site dédié) de jouer les échantillons de même estampille en même temps et de continuer de la même façon en respectant l'ordre total. Le mixage est alors musicalement synchronisé et nous pouvons choisir une latence suffisamment grande pour minimiser les pertes. La musique étant déjà produite, nous n'avons plus de contrainte d'interactivité.

Nous pouvons aussi ajouter un flux d'indication issu du chef d'orchestre dans lequel celui-ci pourrait diriger les musiciens en leur indiquant un changement de structure ou de façon de jouer. Dans certains types de musiques improvisées, c'est le chef d'orchestre qui choisit qui va prendre un solo à un moment donné. Dans certains cas, il peut même construire le morceau en chantant une à une les parties de chaque musicien pendant le déroulement du morceau. Certaines compositions de Steve Coleman (http://www.m-base.com/) sont basées sur ce principe et peuvent donc s'adapter à l'architecture « chef d'orchestre ».

Cette architecture nous permet donc de surmonter les problèmes de latence et de gigue. Cependant, d'un point de vue réseau, le chef d'orchestre représente un goulot d'étranglement. Il envoie $2n$ flux (en comptant le flux d'indication) pour en recevoir $n$. Bien que le flux d'indication puisse être de qualité plus faible que la musique, le gain en débit ne suffirait pas à alléger la charge. Précisons que d'après nos tests, un flux audio PCM à 44100Hz en 16 bits encapsulé dans des paquets RTP de taille approchant 1500 bits représente une charge de 0,68Mb/s. Soit, pour un système à quatre musiciens, une bande passante de 8Mb/s nécessaire au chef d'orchestre et 2Mb/s pour chaque musicien. Ces chiffres considèrent des connexions point à point entre le chef d'orchestre et les musiciens. Prenons maintenant une émission multicast des flux d'indication et de synchronisation. Le chef d'orchestre émet maintenant deux flux et en reçoit $n$, soit 4Mb/s de bande passante. La solution multicast est évidemment la plus économe, c'est celle que nous avons choisie pour nos expérimentations. En contrepartie, le multicast nous contraint à installer l'orchestre sur un réseau de type MAN ou avec de la tunnellisation multicast entre sites ne pouvant se joindre qu'en unicast. De plus, le passage à l'échelle d'Internet imposera une reconsidération de la qualité des flux audio pour économiser de la bande passante.

D'un point de vue pratique musicale, il est rare que l'interaction soit unilatérale pour tous les musiciens, il sera alors nécessaire dans la mise en œuvre de donner aux musiciens soit une interaction « chef d'orchestre », soit une interaction « synchronisé » comme celle décrite dans [Bouillot, 2003] et cela en fonction du rôle de chaque musicien dans l'œuvre musicale.

## 4. IMPLÉMENTATION

L'émission et la réception multicast de flux audio ont été développées dans le logiciel jMax [dechelle, 2000]. L'émission d'un flux se fait grâce à l'objet *rtpout* (figure 3), les réceptions avec un objet *rtpin*. Nous avons utilisé la librairie RTP appelée UCL Common Code Library version 1.2.8 développée par le département informatique de l'université ``University College London''. Chaque objet *rtpout* numérote les échantillons qu'il produit dans le champ *timestamp* de RTP. Ce champ est incrémenté à chaque échantillon et respecte donc la notion d'ordre total.

Dans le prototype actuel, chaque objet *rtpin* joue simultanément un échantillon issu de chacune des sources à la même cadence. De plus, les musiciens y ont une perception identique du mixage des sources sonores, incluant alors un retard sur leur propre retour. Cette perception cohérente se fait à l'aide d'un mécanisme d'auto synchronisation [Bouillot, 2003] différent de l'architecture présentée ici.

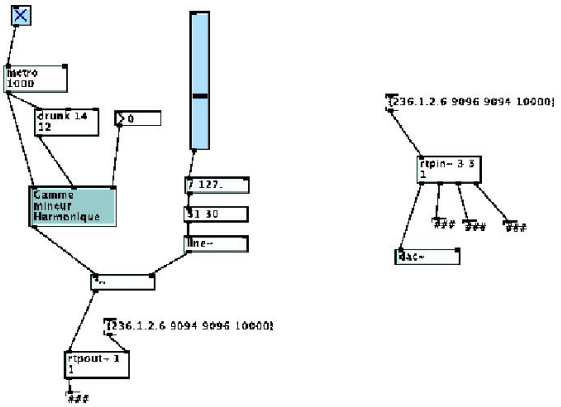


**Figure 3 :** Une configuration de jeu virtuel réparti avec jMax

Le test dont est tiré le patch jMax[6] à mis en jeu trois sites distant (sur réseau local) s'envoyant de la musique en multicast. Le ``sous patch'' *gamme mineur Harmonique* prend en entrée une impulsion (objet *metro*), un entier tiré au hasard et la hauteur de la gamme. L'ascenseur permet de régler le volume du signal audio. Durant ce test, deux musiciens et un automate son étaient connectés ensemble, l'automate envoyant une suite de notre aléatoire. La latence réseau entre les trois sites était suffisamment faible (de l'ordre de 10ms) pour ne pas gêner le jeu des musiciens.

Pour adapter les développements actuels à l'architecture avec chef d'orchestre, nous devons développer un objet spécial qui permettra de recevoir un flux et de le jouer. Il permettra aussi de récupérer celui produit localement par le musicien et de le renvoyer avec l'estampillage correspondant. Le fonctionnement indépendant des objets actuellement développés ainsi que la modularité de jMax nous permettront à l'avenir de tester différentes architectures en vue d'obtenir différents types d'interactions musicales.

## 5. CONCLUSION ET TRAVAUX FUTURS

Compte tenu des contraintes réseaux telles la latence et la gigue, il est impossible de fournir aux musiciens un outil leur permettant de s'entendre mutuellement instantanément. Cependant, d'un point de vue de la performance musicale, nous pouvons compenser ces contraintes en déterminant un mode d'interaction dédié, qui par ailleurs peut imiter ceux déjà existant en musique. C'est le cas de l'architecture à chef d'orchestre. Pour monter de telles performances, nous devons travailler sur des structures musicales dédiées et y adapter notre prototype. Nous envisageons, pour définir ces structures, de travailler conjointement avec des compositeurs et des musicologues. De plus, nous avons montré que l'utilisation du protocole RTP pouvait être étendue à d'autres domaines que le transport de média en temps réel. Dans notre exemple, à une synchronisation répartie nécessaire pour l'application: la synchronisation musicale.

D'un point de vue technique, il nous reste différents points à travailler:

**Augmenter la distance et le nombre de musiciens**

Le travail en cours vise le passage à l'échelle aussi bien en terme d'étendue du réseau que du nombre de musiciens. Les tests actuels n'ayant mis en jeu que deux musiciens réels avec un automate son sur réseau local. Le travail d'expérimentation est prévu pour être complété par une étude de dimensionnement car la garantie de latence constante est de nature fondamentalement statistique.

**Les pertes**

Les phénomènes de latence et de gigue peuvent provoquer des pertes du côté des applications multimédia. Ces pertes doivent alors être compensées sans effet boule de neige. Dans [Bouillot, 2002], deux techniques de compensation sont identifiées. D'un côté l'adaptation du débit d'émission en fonction de la charge du réseau et de l'autre les mécanismes de compensation d'erreur [Perkins, 1998].

**Dérive des horloges des cartes son**

[6] jMax est un langage de programmation graphique, un patch jMax est une fonction graphique

Lors d'une transmission de flux audio sur un réseau, plusieurs cartes son entrent en jeu. Celle de l'émetteur et celles des récepteurs. Nous savons que la période d'oscillation du quartz qui cadence la carte son varie sensiblement en fonction de la température. Les horloges des cartes vont donc différer de leur valeur nominale. Elles vont alors dériver au cours du temps, provoquant une surconsommation ou une surproduction d'échantillons sonores. Différents articles traitent de ce problème de dérive des horloges dans le contexte de l'audio [Orion, 2000], [Akester, 2002], [Fober, 2002], [Brandt, 1999].

Nous pensons inclure dans le prototype de l'orchestre virtuel réparti des mécanismes de compensation des pertes ainsi qu'un autre mécanisme de synchronisation: celui des horloges des cartes son. L'assemblage de toutes ces techniques permettra à terme d'assurer le confort adéquat aux musiciens distants et au public.

Cependant, nous privilégions actuellement les tests sur un réseau de type MAN ainsi que des collaborations futures avec compositeurs et/ou musicologues. Cela nous permettrait d'une part de dimensionner les mécanismes mis en jeu et d'autre part, de déterminer plus précisément le type d'interaction que nous allons fournir aux musiciens distants.

## BIBLIOGRAPHIE


[Akester, 2002] Akester R., Hailes S., "A New Audio Skew Detection and Correction Algorithm". ICMC (2002).

[Becquet, 2001] Becquet E., Abdallah M., Gressier-Soudan E., Horn F., Bacon L. "Object Oriented Timed Messaging Service for Industrial Ethernet: a Fieldbus Like Architecture for Power Plant Control and Factory Automation". Fieldbus Technologie (FeT'2001), IFAC. Nancy, France (2001).

[Brandt, 1999] Brandt, Dannenberg, "Time in Distributed Real-Time Systems". Proceedings of International Computer Music Conference, pages 523-526, San Francisco (1999).

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

[Bouillot, 2002] Bouillot N., "Transport du son produit en temps réel sur les réseaux best-effort". Rapport bibliographique, DEA SIR, P6, CNAM, ENST (2002).

[Bouillot, 2003] Bouillot N., "Un algorithme d'auto synchronisation distribuée de flux audio dans le concert virtuel réparti". CFSE'03, La Colle Sur Loup, France (2003).

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

[Dechelle, 2000] Dechelle F., "jMax: un environnement pour la réalisation d'applications musicales temps réel sous Linux". Actes des journées d'Informatique Musicale (2000).

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

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

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

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

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

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

[Hodson, 1999] Hodson O., Perkins C., "Robust Audio Tool (RAT) version 4". (1999)
http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/

[McCanne, 1995 ] McCanne S., Jacobson V., "vic: A Flexible Framework for Packet Video". ACM multimedia (1995).

[Orion, 2000] Orion, Hodson, Colin, "Skew Detection And Compensation For Internet Audio Applications". (2000).

[Puckette, 1999] Puckette M., Danks M., Steiger R., Sorensen V., http://visualmusic.org/gvm.htm. (1999).

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

[Rosenberg, 1998] Rosenberg J., Schulzrinne H., " An RTP Payload Format for Generic Forward Error Correction ". Internet-Draft draft-ietf-avt-fec-03.txt (work in progress) (1998).

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

[Tase, 1996] Utility Communications Specification Working Group, "TASE.2 Services and Protocol. version 1996-08. iccp inter-control centre communication protocol version 6.1". Technical Report IEC 870-6-503, IEC (1996).

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

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