Chapitre 9

Modélisation et implémentation d’architectures multimédias ; application au cas de la visioconférence à qualité de service garantie

PHILIPPE OWEZARSKI – MARC BOYER

9.1. Introduction

L'arrivée du multimédia dans l'informatique a bouleversé les contraintes d'écriture des applications : les ordinateurs n'ont plus pour seule tâche de faire un maximum de calculs

en un minimum de temps (*best effort*) mais également de présenter la bonne information à une date qui respecte sa validité temporelle.

Ce changement de point de vue oblige à repenser la façon dont sont conçues les applications, et à développer des modèles permettant d'appréhender les contraintes temporelles des applications, en particulier dans le domaine très important des applications multimédias distribuées.

Dans ce chapitre, nous présentons l'étude d'un système de visioconférence à qualité de service garantie. Nous montrons comment l'utilisation d'un modèle formel (les réseaux de Petri temporel à flux, présentés au Chapitre 6) a permis de dériver, de la modélisation des contraintes des flux multimédias, une architecture des contraintes de synchronisation, qui a alors guidé la conception globale de l'application.

Le paragraphe 9.2 présente les caractéristiques et besoins des informations multimédias et les systèmes asynchrones qui sont leur support matériel et logiciel. Le paragraphe 9.3 montre comment les réseaux de Petri temporels à flux permettent de modéliser les contraintes des flux multimédias manipulés ; de cette modélisation des contraintes, on peut dériver une architecture de synchronisation de l'application (paragraphe 9.4). Cette maîtrise de l'architecture de l'application nous a permis de mettre en place une renégociation dynamique de sa qualité de service qui s'est avérée intégrée dans le modèle, et donc apparaît comme une simple modification du schéma de synchronisation entre les données (paragraphe 9.5).

9.2. Problématique de la synchronisation multimédia

9.2.1. Informations multimédias : caractéristiques et besoins

9.2.1.1. Notion de flux, flux continus, flux discrets

La première caractéristique qui distingue les données multimédias des données informatiques classiques (textes, données binaires…) est leur unité de traitement : les données multimédias se manipulent par ***flux***, alors que les textes et les données binaires se manipulent par fichiers. En effet, les données audio et vidéo sont des séquences d'images ou d'échantillons sonores qui se succèdent à une cadence constante ou non. Ces données ne sont pas incompatibles avec la notion de fichier informatique, et les fichiers sont toujours utilisés pour stocker des films ou des documents sonores ; mais les traitements se font unité d'information par unité d'information (image par image, par exemple) ; ces unités d'information mises bout à bout forment un flux. Il faut noter

toutefois que les textes, graphiques et données binaires s'accommodent de ce mode de traitement.

Ainsi, les flux se caractérisent aussi par les relations temporelles qui existent entre les différentes unités d'information qui les composent. Par exemple, il n'existe pas de relation temporelle entre les caractères qui composent un flux textuel. De même, pour une image unique qui peut être vue comme un flux de bits, ou pour un graphique, il n'existe aucune relation temporelle entre les différentes unités du flux. Ce sont typiquement des fichiers, qui, à la limite, pourraient être considérés comme des flux simples ou des ***flux discrets.***

Par contre, pour la vidéo ou le son, les images ou les échantillons sonores doivent être produits, traités ou présentés avec une cadence régulière. On parle dans ce cas là de ***flux continus***. Si le temps entre deux unités de base d'un flux est constant, on parle de flux *isochrone* ; toutefois, une certaine variabilité sur ces temps peut être tolérée : cette variabilité est appelée *gigue autorisée*.

9.2.1.2. La notion de qualité de service

Dans la présentation de la notion de flux qui vient d'être faite sont déjà apparues des notions de contraintes temporelles. Ces contraintes ne sont pas les seules et d'autres paramètres de qualité existent. En fait, les flux multimédias se caractérisent par leur qualité de service (ou QdS). Une des familles de paramètres de qualité de service concerne la qualité de restitution des flux multimédias.

Par exemple, les données binaires ou textuelles ne tolèrent aucune perte (elles nécessitent une fiabilité totale) ; par contre, leurs exigences en terme de capacité de stockage ou le débit requis pour les transmettre sur un réseau sont très faibles, et ce type de données ne présente aucune contrainte temporelle.

De manière identique, les graphiques supportent très mal les pertes ou les erreurs. Ils ne requièrent que peu de ressources de stockage ou de communication, et sont relativement insensibles aux phénomènes de gigue.

Les images fixes, quant à elles, tolèrent assez bien les erreurs ou la perte de certaines informations. Toutefois, la capacité de stockage ou le débit requis sur un réseau de communication sont supérieurs à ce qui est nécessaire pour du texte ou du graphique.

D'ailleurs pour réduire la quantité de données d'une image vidéo, des algorithmes de compression ont été conçus comme GIF ou JPEG[1] [WALLACE 91].

Le son, lui, supporte difficilement les erreurs ou les pertes ; ces parasites nuisent de façon relativement importante à la qualité du flux audio. Le débit requis et la capacité de stockage sont variables en fonction du codage utilisé : seulement 64 kbps sont nécessaires pour du son de la qualité du téléphone numérique, alors que 1,2 Mbps (sans compression) sont requis pour du son Hi-Fi stéréophonique de qualité laser. De plus, le flux audio est très sensible aux perturbations temporelles qu'il peut subir.

Enfin, la vidéo animée est le média le moins contraignant en terme d'erreur et de perte : la perte d'une image dans un flux vidéo n'est pas perceptible par l'utilisateur terminal. Par contre la quantité de données engendrée et le débit requis sont considérables. D'ailleurs des efforts importants ont été effectués dans le secteur de la compression vidéo avec les algorithmes H261 [LIOU 91] [TURLETTI 93], MPEG[2] [LE GALL 91], etc. ; toutefois, les flux vidéo, même compressés, restent extrêmement gourmands en place mémoire et en bande passante. De plus, les contraintes temporelles sur les images d'un flux vidéo sont assez fortes, même si un certain niveau de gigue reste acceptable.

Ce qui précède vient de faire un petit inventaire des contraintes de qualités de service des différents médias. Toutefois, cet inventaire n'est pas exhaustif et ne prend en compte que les paramètres de QdS en rapport avec les pertes, les erreurs, la place mémoire et la bande passante requise. De toute manière, énumérer tous les paramètres de qualité de service est impossible car chaque application aura ses propres besoins, et aura donc ses propres paramètres de QdS à respecter sur ses données propres. Nous nous contentons donc de présenter les paramètres de QdS les plus habituels dans les applications multimédias actuelles comme la visioconférence.

9.2.1.3. La synchronisation multimédia

La partie précédente vient de mettre en évidence la notion de QdS multimédia et l'a illustrée en présentant les principaux paramètres de qualité de présentation associés aux flux multimédias. Toutefois, l'aspect temporel qui a été signalé dans la partie 9.2.1.1 est essentiel, et les flux continus se caractérisent surtout par leurs contraintes temporelles. Par exemple, dans une présentation multimédia qui comporte plusieurs types de données,

---

[1]JPEG : Joint Photographic Expert Group.
[2]MPEG : Motion Pictures Expert Group.

il existe des relations temporelles et spatiales dans les différents flux. Il peut également exister des relations temporelles et spatiales entre ces mêmes flux. Ces relations définissent en fait les contraintes de synchronisation multimédia. La suite va en présenter les différents aspects.

9.2.1.3.1. La synchronisation spatiale

La synchronisation spatiale exprime les contraintes d'ordonnancement visuel des différents objets multimédias sur le support de présentation (écran, mur d'image, etc.). Elle permet donc de définir la taille des différentes zones, les superpositions, les juxtapositions, etc. comme dans la norme MHEG[3] [ISO 90] [ISO 93].

9.2.1.3.2. La synchronisation temporelle

La résolution des problèmes liés à la synchronisation temporelle constitue le point essentiel et le plus délicat dans la conception des systèmes multimédias [BLAKOWSKI 96]. Il s'agit d'exprimer et de garantir les contraintes et relations temporelles qui existent entre les objets d'un document multimédia. Deux types de synchronisations apparaissent : la synchronisation intra-flux et la synchronisation inter-flux.

La synchronisation intra-flux consiste à respecter les contraintes de présentation sur chacune des unités d'information d'un flux ; ceci revient donc à contrôler la gigue de telle sorte que le temps de présentation ne s'écarte pas de sa valeur idéale d'une valeur supérieure à la gigue maximale autorisée. Par exemple, pour une animation vidéo, il faut garantir que le temps de présentation de chaque objet vidéo (image) respectera le temps qui existait entre deux captures lors de la création de l'animation.

La synchronisation inter-flux consiste à contrôler la dérive (ou décalage temporel) qui peut exister entre deux flux. La dérive est due à l'effet cumulatif de la gigue ; en effet, les gigues que subissent chacun des objets du flux peuvent s'accumuler, et le décalage qui peut apparaître entre deux flux peut devenir très grand. Il faut donc contrôler cette dérive et faire en sorte qu'elle reste en deçà d'un seuil de tolérance. C'est typiquement le cas de la synchronisation entre un flux audio et un flux vidéo, pour lesquels il faut assurer que le son correspond bien aux mouvements des lèvres ; ce problème est connu sous le nom de "synchronisation des lèvres".

[3]MHEG : Multimedia and Hypermedia information coding Expert Group.

De même, il existe une séparation entre la synchronisation discrète et la synchronisation continue.

La synchronisation discrète apparaît la plupart du temps en présence d'un flux discret ; elle consiste à synchroniser des objets lorsque le besoin s'en fait sentir. Par exemple, dans le cadre d'un film sous-titré, il faut synchroniser le texte du sous-titrage avec le film, ce qui n'est nécessaire que lorsqu'il y a des dialogues.

En revanche, la synchronisation continue consiste à introduire tout au long du flux, périodiquement, des points de synchronisation dans la présentation du ou des flux. Par exemple, dans le cas de la synchronisation des lèvres, il faut périodiquement introduire des points de synchronisation entre les flux audio et vidéo pour éviter que la dérive ne croisse trop.

9.2.1.3.3. La synchronisation hypermédia

La synchronisation hypermédia intègre les notions de synchronisation spatiale et de synchronisation temporelle. Toutefois, elle ajoute à ces notions la notion de synchronisation logique qui consiste à synchroniser les présentations de l'application lors du parcours des liens d'un document hypermédia (comme dans MHEG [ISO 90] [ISO 93], ou plus généralement [SENAC 96]). Elle ne sera toutefois pas considérée dans ce chapitre.

9.2.2. Les systèmes distribués asynchrones

9.2.2.1. Pourquoi des systèmes distribués asynchrones ?

L'étude et les travaux de synchronisation décrits dans ce chapitre ont été effectués dans le cadre des systèmes distribués asynchrones. Ce choix s'est imposé pour trois raisons :

– Tout d'abord, les systèmes distribués asynchrones représentent à quelques exceptions prés la totalité des systèmes opérationnels existant à l'échelle planétaire. Les réseaux locaux Ethernet, FDDI, Token Ring… ainsi que les lignes Internet qui les interconnectent sont asynchrones ; les systèmes opératoires qui s'exécutent sur les stations de travail sont également asynchrones (Unix, DOS, Windows…). De plus, la tendance actuelle (imposée par les constructeurs) ne fait qu'accroître l'asynchronisme des systèmes pour gagner en performance, et l'asynchronisme semble donc être l'avenir dans le domaine des machines de bureau : l'asynchronisme est le point de passage obligé vers les hauts débits. Aussi, nous avons souhaité faire une étude dans un cadre le plus général

possible pour donner des mécanismes de synchronisation qui pourront être utilisés dans un maximum d'applications sur un maximum de sites présents et futurs ;

– Ensuite, la solution au problème de la synchronisation multimédia dans un environnement synchrone est un cas particulier de celle en environnement asynchrone (c'est le cas où l'asynchronisme est nul), et nous apportons donc une solution au problème le plus général ;

– Enfin, avec l'intégration dans UNIX (POSIX) de classes d'ordonnancement temps réel (comme dans Solaris 2), il est possible de réaliser des applications à contraintes temporelles, ce qui n'était pas le cas au début de la décennie lorsque les travaux décrits dans [COULSON 94][JEFFAY 94b] et [JEFFAY 92] ont débuté, et avaient alors déclaré UNIX inapte pour supporter des applications ayant des contraintes temps réel [VOGEL 95]. Ces travaux préconisaient donc d'utiliser des systèmes temps réel qui étaient les seuls susceptibles de respecter les contraintes temporelles des applications. Il faut noter ici que la théorie sur les systèmes temps réel est une chose, mais que la réalité en est une autre [KATCHER 94]. Dès que l'on essaie d'implémenter un système opératoire temps réel, de nombreux problèmes d'asynchronisme peuvent se poser, comme avec un système classique (comme Solaris 2) :

- entrées / sorties asynchrones ;
- problèmes d'inversion de priorités lors de la synchronisation de processus ;
- le noyau doit être entièrement préemptible ;
- les tâches systèmes indispensables peuvent perturber l'ordonnancement temporel des applications ;
- etc.

En fait, pour obtenir un comportement temps réel avec un système temps réel, il faut se limiter à des processus de calcul pur qui ne font pas d'entrées / sorties [BAKER 94]. Cependant, ceci est incompatible avec les applications distribuées multimédias qui font des accès très nombreux aux cartes audio et vidéo et aux cartes réseaux. Les conditions d'utilisation d'un système temps réel sont donc très restrictives pour des applications distribuées multimédias. En fait, utiliser des systèmes classiques (UNIX ou POSIX) revêt beaucoup d'intérêt, par leur nombre et leur disponibilité, et ne pose pas plus de problèmes que lors de l'utilisation d'un système temps réel [ADELBERG 94].

De plus, le choix des systèmes asynchrones s'est imposé car [KANG 94] a montré que les systèmes synchrones à grande échelle sont impossibles à mettre en œuvre. Ainsi, pour l'exemple de la visioconférence, qui nous intéresse tout particulièrement, et qui est une application distribuée, et longue distance, il faut donc passer par des systèmes asynchrones.

9.2.2.2. Caractérisation des systèmes asynchrones

L'utilisation d'un système asynchrone, passage quasiment obligé pour le développement des applications multimédias coopératives et en particulier celui d'une application de visioconférence, introduit néanmoins de nombreux problèmes pour traiter les données multimédias isochrones (audio et vidéo). Le problème essentiel réside dans la variabilité et l'imprévisibilité des temps de traitement des différentes opérations, et ceci surtout car ce temps de traitement est non borné supérieurement. Cette variabilité apparaît à plusieurs niveaux dans les systèmes distribués multimédias asynchrones : les trois niveaux qu'il est possible de séparer sont :

– les supports et protocoles de communication. Par exemple, les modes d'accès à des réseaux comme Ethernet, Token ring ou FDDI sont non déterministes (dépendant de la charge du réseau par exemple), et aucun délai de transit maximum n'est garanti. Ce phénomène d'asynchronisme et d'imprévisibilité est encore plus marqué lorsqu'un réseau public longue distance comme le réseau Internet est considéré ;

– le système opératoire temps partagé classique (par exemple Unix) dont la variabilité dans les temps de traitement est due :

- aux mécanismes d'ordonnancement temps partagé du système qui tendent à privilégier les processus interactifs et le débit moyen du système d'exploitation, au détriment de certaines contraintes temps réel des processus. De plus, la notion de processus lourd telle qu'elle est utilisée dans Unix induit une surcharge en matière d'ordonnancement incompatible avec le haut degré de parallélisme et les hauts débits des données associées aux flux multimédias ;

- au caractère non préemptif ou localement préemptif du noyau qui induit un temps de latence non déterministe pour le traitement des interruptions ;

- à la commutation des pages mémoires et de nombreuses autres tâches système qui s'exécutent à des moments non déterministes et avec des priorités supérieures aux tâches utilisateur, et qui en cela gênent la qualité temporelle de ces dernières ;

– le matériel, et en particulier les cartes multimédias (audio et vidéo). En effet, la nouvelle génération de microprocesseurs intègre des caches d'instructions et de données multi-niveaux qui génèrent des temps d'accès mémoire et de commutation de contextes non déterministes. De même, la gestion de la mémoire virtuelle introduit une variabilité des temps d'accès mémoire. Cette variabilité est renforcée dans les architectures multiprocesseurs par les mécanismes de réactualisation des caches et les contentions pour l'accès au bus. De plus, les systèmes de gestion d'interruptions hiérarchisées induisent des temps de latence non déterministes. Les disques, les processeurs spécialisés de compression / décompression et les périphériques multimédias sont ainsi autant de sources d'asynchronisme.

9.2.2.3. Les problèmes engendrés par les systèmes asynchrones

Les problèmes qui apparaissent dans les systèmes asynchrones sont donc :

• les problèmes de gigue dus à la variabilité temporelle des opérations effectuées dans le système. Ce dernier ne peut pas garantir un temps de traitement constant et/ou prévisible pour les différentes opérations et ne peut donc pas garantir les temps de présentation des données multimédias ;

• les problèmes de dérive qui sont dus à l'effet cumulatif de la gigue et peuvent faire apparaître un décalage important après le traitement d'une longue séquence d'objets d'un flux ;

• et enfin, le problèmes de pertes et de duplications ; en effet, l'ordonnancement des taches étant imprévisible, il est possible que dans le cadre de la gestion d'un tampon, par exemple, les tâches de production accèdent plus facilement au processeur que les tâches de consommation. Ceci conduit à une saturation du tampon et à des pertes par écrasement. De même, des pertes peuvent apparaître dans les réseaux de communication. Des duplications d'objets peuvent également se produire, en particulier dans les réseaux, lorsque l'algorithme de routage duplique un paquet en plusieurs exemplaires.

Ce sont tous ces problèmes qui devront être résolus, ou du moins contrôlés, pour permettre l'écriture d'applications distribuées multimédias.

9.3. Modélisation des contraintes de synchronisation multimédia

9.3.1. Les besoins en modélisation

Ce qui précède a montré l'extrême variété des contraintes de QdS qui peuvent exister. De plus, les contraintes temporelles sont relativement complexes à exprimer, et l'un des premiers problèmes à résoudre consiste à pouvoir représenter de façon simple et complète les contraintes de synchronisation qui peuvent exister dans un document multimédia.

Ce point n'était pas apparu essentiel de prime abord, et nous avions pensé pouvoir garantir la synchronisation intra et inter-flux dans un système de visioconférence sans modéliser les contraintes que nous devions respecter. Pour cela, une solution intuitive et classique pour la synchronisation multimédia consiste à utiliser des estampilles [DIAZ 94a] qui contiennent les dates de présentation de chaque objet. Ainsi, chaque objet est estampillé à l'aide d'une date relative (la date 0 correspondant au lancement de l'application), et le processus de présentation n'a plus qu'à présenter l'objet considéré à la

date indiquée par son estampille. Cette technique a été utilisée en particulier dans le développement d'un premier prototype d'application de visioconférence synchronisée TSVS (Timestamp Synchronized Visioconference System)[4], dont les qualités et défauts sont évalués plus avant (cette technique a aussi été utilisée dans [ROTHERMEL 95]).

La technique de synchronisation par estampille est simple à implémenter, et elle résout à la fois les problèmes de synchronisation intra et inter-flux. La synchronisation intra-flux est forcément garantie, car une séquence capturée en N secondes sera jouée en N secondes avec une cadence de présentation des objets régulière. D'autre part, la synchronisation inter-flux est également obtenue, car chaque flux se synchronise par rapport à une base temporelle commune, ce qui synchronise donc les deux flux l'un par rapport à l'autre [DIAZ 94a].

Cependant, les estampilles ne prennent pas en compte la notion d'asynchronisme, car elles n'intègrent pas la notion d'intervalle temporel, et ne peuvent pas supporter une gigue sur un objet. Ainsi, si un objet arrive quelques instants après sa date de présentation, il ne sera pas présenté, ce qui crée une discontinuité dans le flux, alors que le retard correspondait peut être à une gigue acceptable ; de ce fait, la donnée a été écartée alors qu'elle aurait pu être présentée avec un léger retard[5]. La dégradation en résultant peut donc être, dans le cas général, bien supérieure à ce qu'elle aurait pu être, ce qui est inadmissible puisque l'objectif est d'obtenir la meilleure qualité de présentation possible. De plus, à cause de l'asynchronisme, il est impossible de garantir des dates fixes au niveau du système opératoire. Enfin, la technique de synchronisation par estampille ne permet qu'une connaissance a posteriori des contraintes de synchronisation à respecter ; l'entité de synchronisation terminale ne connaît la date de présentation d'un objet que lorsqu'elle reçoit cet objet, et elle ne peut donc pas anticiper des traitements à mettre en œuvre.

---

[4]TSVS est un système de visioconférence synchronisé à l'aide de la technique par estampille. La qualité de synchronisation est très bonne et l'outil a un fonctionnement tout à fait acceptable lorsque les problèmes liés à l'asynchronisme sont réduits (i.e. les machines et le réseau ne sont pas trop chargés). Cet outil est disponible sur simple demande auprès de ses auteurs.

[5]Dans TSVS, les problèmes de gigues sont résolus par un artifice. En effet, pour éviter d'avoir des problèmes de gigues positives (qui ne peuvent être résolus), les données reçues sont stockées pendant un temps supérieur à la gigue maximale généralement observée. Cependant, ce stockage est très dommageable pour l'interactivité, car il induit un retard conséquent entre l'émetteur et le récepteur. C'est pourquoi, TSVS ne fonctionne correctement que lorsque les problèmes liés à l'asynchronisme sont réduits.

Ces constatations ont tout à fait conforté notre opinion : il s'avère indispensable de modéliser de façon très précise les contraintes de synchronisation multimédia que nous voulions voir respectées dans nos applications. Pour cela, nous avons été amenés à faire une étude des modèles existants pour voir s'il n'en existait pas un qui pourrait convenir à la problématique que nous nous sommes posée.

Pour pouvoir réaliser des applications multimédias synchronisées avec les hypothèses minimalistes que nous nous sommes fixées, il faut tenir compte de tous les problèmes liés à la variabilité des temps de traitement des systèmes asynchrones (gigue, dérive) et les relier aux propriétés intrinsèques de chaque objet multimédia (par exemple relier la gigue d'un système distribué avec la gigue admissible propre à chaque objet multimédia). Aussi, pour représenter ces propriétés de synchronisation des objets multimédias, il est essentiel de disposer d'un modèle permettant à l'auteur d'une application multimédia de spécifier les contraintes de synchronisation devant être satisfaites par l'application. Cette approche formelle est particulièrement intéressante car elle permet d'une part de spécifier sans ambiguïté des scénarios de présentations multimédias, mais elle permet également de les simuler et de les valider. De nombreuses études ont déjà été réalisées dans ce domaine, et des modèles ont été proposés [DIAZ 93a]. En particulier, certains de ces modèles utilisent des approches formelles basées sur les réseaux de Petri temporisés dont le caractère graphique permet de mettre facilement en œuvre des paradigmes multimédias tels que le paradigme de régie numérique [SENAC 94]. Les Chapitres 6 et 13 ont fait un panorama des extensions temporelles des réseaux de Petri en les confrontant à la problématique de la synchronisation en terme de pouvoir d'expression et de modélisation[6]. A partir du panorama qui a été dressé dans ces deux chapitres, il apparaît que les modèles de synchronisation multimédia et en particulier les extensions temporelles des réseaux de Petri qui avaient été proposées jusqu'alors n'offraient pas un bon pouvoir de modélisation et d'expression pour spécifier des scénarios de synchronisation dans une application multimédia.

Ces limitations observées au niveau des modèles existant ont conduit à proposer un nouveau modèle (basé sur les réseaux de Petri[7]), appelé RdPFT (réseau de Petri à flux temporel, présenté au chapitre 6), offrant à la fois le pouvoir d'expression du modèle TPN et le pouvoir de modélisation nécessaire. Le modèle RdPFT [DIAZ 93a] [DIAZ 93b]

[6]Le pouvoir de modélisation d'un modèle est sa capacité à représenter de façon aisée un scénario. Son pouvoir d'expression est sa capacité à spécifier le scénario de façon complète.

[7]Pour l'aspect graphique des réseaux de Petri qui permettent de mieux visualiser les caractéristiques de synchronisation d'un document multimédia.

[SENAC 94] s'inspire du modèle TPN auquel il ajoute la composition temporelle et donc un typage des transitions par des règles de tir de transitions inter-flux.

Ainsi, les RdPFT utilisent des intervalles temporels sur les arcs sortant des places, ce qui permet à la fois de tenir compte du non déterminisme temporel des systèmes distribués asynchrones et de la variabilité des temps de présentation des objets multimédias. Les intervalles temporels dans un RdPFT sont des triplets (x^S, n^S, y^S) appelés intervalles de validité temporelle, où x^S, n^S et y^S sont respectivement les temps du traitement de présentation minimal, nominal et maximal. Les durées minimale et maximale sont utiles pour calculer la dérive temporelle sur les arcs (par rapport à la durée nominale).

Les dérives temporelles entre flux multimédias peuvent être contrôlées de façon très précise grâce à 9 sémantiques de transition différentes. D'un point de vue exécution, ces sémantiques de synchronisation sont définies comme des instants de synchronisation, prenant en compte la durée réelle des processus ; d'un point de vue modélisation, ces règles de tir définissent des intervalles de tir couvrant tous les instants de synchronisation possibles, obtenus par combinaison complète et consistante des intervalles dynamiques de validité temporelle des arcs concernés [DIAZ 93a] [SENAC 94]. Par exemple, en utilisant ces règles de transition, il est possible de spécifier des mécanismes de synchronisation conduits par le processus le plus en avance (synchronisation de type "ou fort"), par le processus le plus en retard (synchronisation de type "et faible") ou par un processus donné (synchronisation de type "maître"). Ces sémantiques de synchronisation permettent de définir les instants de synchronisation à partir d’arcs choisis statiquement ou dynamiquement.

9.3.2. Exemple de modélisation d’une application de visioconférence

Le modèle RdPFT constitue un modèle parfaitement adapté à la modélisation des contraintes de synchronisation des flux multimédias, en environnement asynchrone, pour les applications que nous considérons. Grâce à ses pouvoirs de modélisation et d'expression élevés, ce modèle permet de modéliser facilement des scénarios de synchronisation de complexité forte. De plus, à partir de la description en RdPFT d'un scénario de synchronisation, il est possible de vérifier sa validité temporelle. Cette

vérification se fait par l'intermédiaire de techniques conçues dans ce but, mais dont la description n'est pas l'objet de ce chapitre[8].

En ce qui concerne l'aspect synchronisation multimédia de ce chapitre, le point essentiel consiste à étudier comment il est possible à partir d'un RdPFT de décrire le comportement d'une couche de synchronisation, et en particulier d'obtenir l'ordonnancement temporel des processus de présentation multimédia, de façon à ce que pour tous les flux de données, les contraintes de synchronisation intra et inter-flux soient garanties.

Le modèle RdPFT servira dans la suite de ce chapitre à représenter et à implémenter les contraintes de synchronisation d'une application de visioconférence. La visioconférence illustre d'ailleurs très bien les possibilités du modèle RdPFT.

Dans la visioconférence, un flux audio et un flux vidéo doivent être synchronisés. Un certain nombre de paramètres de qualité de service peuvent être représentés par un RdPFT. Ainsi, la Figure 9.1, commentée dans le paragraphe suivant, modélise une application de visioconférence dont les paramètres de qualité de service sont :

– le débit est de 10 images par seconde ;

– la gigue maximale acceptable sur un objet, audio ou vidéo, est 10 ms [JEFFAY 94a] ;

– comme dans toute application de visioconférence, le son possède un rôle prépondérant, la vidéo n'étant que secondaire ;

– la qualité de la synchronisation est voulue excellente, c'est à dire que la dérive inter-flux ne doit pas excéder 100 ms [JEFFAY 94a], 100 ms étant la limite en deçà de laquelle le décalage audio vidéo n'est pas perceptible par l'homme.

[8][Courtiat 96] propose également une méthodologie pour vérifier la cohérence temporelle d'un document multimédia/hypermédia.

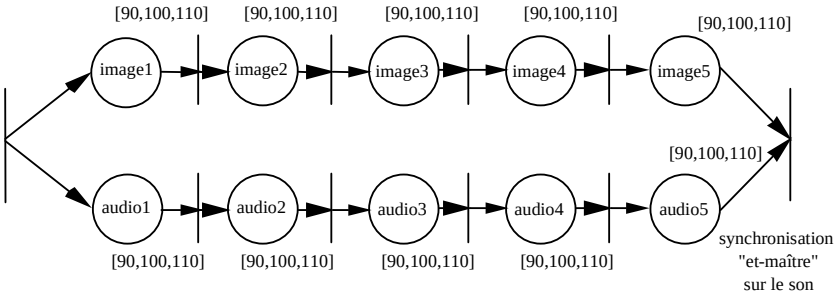

Figure 9.1. *Exemple de RdPFT pour la visioconférence à 10 images par seconde*

Ces paramètres de qualité de service ont permis de déterminer les différents paramètres du RdPFT représenté sur la Figure 9.1. Ainsi :

– Le débit de 10 images par seconde a permis de déterminer la durée de présentation nominale d'un objet vidéo, *i.e.* 100 ms (si on considère des granularités identiques sur les flux audio et vidéo, c'est aussi le temps de présentation nominal d'un paquet son) ;

– La gigue maximale acceptable a permis de déterminer les intervalles de validité temporelle qui sont donc [90, 100, 110] ;

– La synchronisation inter-flux est de type « et-maître » sur le son. En effet, le son ayant une prépondérance par rapport à la vidéo, ses contraintes de synchronisation intra-flux doivent être absolument respectées, même si celles sur la vidéo ne le sont pas. Cependant, il faut synchroniser l'un par rapport à l'autre deux flux continus, et éviter au maximum les discontinuités sur le flux vidéo (pouvant être causées par le mécanisme de l'accélération de flux). Pour cela, la règle de tir "et-maître" a été choisie, assurant ainsi que les contraintes sur le son seront garanties, mais essayant autant que possible de respecter aussi les contraintes temporelles sur la vidéo, lorsque c'est possible ;

– La dérive inter-flux ne doit pas dépasser 100 ms, aussi la période de synchronisation inter-flux correspond à la présentation de 5 images. En effet, la dérive maximale sur 5 objets audio ou vidéo est de 50 ms. La dérive entre les deux flux est donc au maximum de 100 ms.

9.4. Modélisation d'une architecture de synchronisation

9.4.1. Introduction

Cette partie a pour but d'étudier et de développer une approche et un ensemble de mécanismes permettant de garantir aux utilisateurs des paramètres de qualité de service multimédia, comme la qualité audio, la qualité vidéo, le débit vidéo, le délai de bout en bout ou les contraintes de synchronisation temporelles, dans les applications multimédias distribuées. Plus particulièrement, cette partie va se focaliser sur la garantie des contraintes de synchronisation d'une application de visioconférence en environnement asynchrone (PNSVS : Petri Net Synchronized Videoconference System). En effet, la synchronisation multimédia est la contrainte la plus importante à assurer dans le cadre des systèmes distribués multimédias [BLAKOWSKI 95]. La problématique associée à la synchronisation multimédia consiste, comme cela a été détaillé dans la partie 9.2.1.3, à assurer à la fois les contraintes de synchronisation intra et inter-flux.

La suite de cette partie suit donc le plan suivant : tout d'abord, il sera montré que le comportement de l'application de synchronisation peut être très différent de celui du scénario de synchronisation exprimé par l'utilisateur, pour tenir compte des caractéristiques spécifiques des composants matériels de l'ordinateur et du système opératoire. Cette partie (9.4.2.2) montrera en effet qu'il existe deux niveaux de synchronisation dans une application de visioconférence comme PNSVS. Puis, en analysant les résultats obtenus avec PNSVS, la partie (9.4.3) montrera qu'il est judicieux d'utiliser pour PNSVS un service transport à ordre partiel, qui permet par rapport à un transport standard de considérablement améliorer les performances et la qualité de présentation. La nouvelle architecture de PNSVS sera ainsi donnée, et elle sera évaluée, la partie (9.4.3.5) présentant les résultats d'évaluation.

9.4.2. Modélisation d'une application de visioconférence

9.4.2.1. Latence des cartes multimédias et décalage des synchronisations inter-flux

Soit le RdPFT de la Figure 9.1. Ce RdPFT de présentation montre comment les objets du scénario audio et vidéo doivent être synchronisés. En particulier pour la synchronisation inter-flux, aux problèmes de dérive prés, le son i doit être synchronisé à l'image i. Toutefois, les cartes de présentation multimédia n'ont pas toutes les mêmes temps de latence. Ainsi, si l'on considère une carte vidéo ayant un temps de latence de 50 ms et une carte audio dont le temps de latence est 250 ms, alors si le moteur de l'application respecte le RdPFT de présentation de la Figure 9.1, la présentation finale ne sera pas synchronisée. En effet, la partie son sera retardée de 200 ms par rapport à la partie vidéo, i.e. le son i ($i \in N$) sera synchronisé avec l'image i+2.

Pour résoudre ces problèmes de temps de latence différents sur les différentes cartes, des décalages doivent être introduits dans les synchronisations inter-flux (on parle aussi de rendez-vous décalés [OWEZARSKI 96a]). En effet, la différence entre les temps de latence de la carte son et de la carte vidéo étant de 200 ms, il suffit de synchroniser au niveau du moteur de synchronisation le son i avec l'image i-2 (Figure 9.2), ce qui – après passage dans les cartes de présentation multimédia – donnera une présentation dans laquelle le son i et l'image i seront synchronisés.

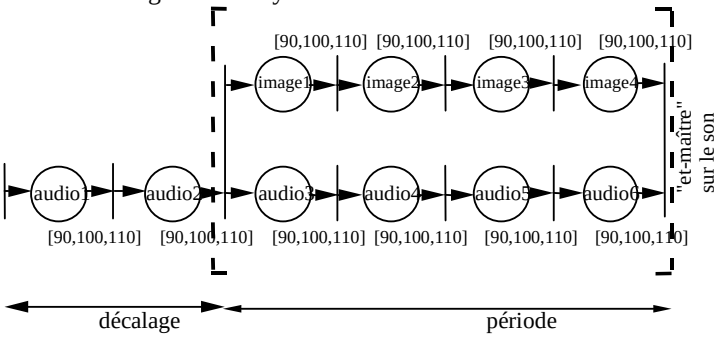

Figure 9.2. *RdPFT applicatif tenant compte des temps de latence matériels*

Cependant, l'exemple choisi est idéal, car la différence entre les temps de latence audio et vidéo est multiple du temps nominal de présentation d'un objet. Si ce n'est pas le cas, avec par exemple des temps de latence audio et vidéo égaux respectivement à 230 et 50 ms, alors la différence entre les temps de latence est de 180 ms. Dans ce cas là, comme dans le précédent, le décalage qui doit intervenir dans les rendez-vous entre les flux audio et vidéo est également de deux objets représentant 200 ms de décalage. Cependant, après ce décalage, il subsiste un décalage résiduel de 20 ms entre les flux audio et vidéo. Aussi, pour que la dérive inter-flux soit inférieure à 100 ms (cf. exemple de la Figure 9.1), la période de synchronisation inter-flux du RdPFT doit être changée. Si la synchronisation inter-flux était faite toutes les 5 images, alors maintenant à cause des 20 ms de décalage qui n'ont pu être éliminées, la dérive inter-flux par rapport au flux audio serait comprise dans l'intervalle [-80 ms, 120 ms], ce qui en valeur absolue peut être supérieur aux 100 ms autorisées. Aussi, une synchronisation inter-flux devra être faite toutes les 4 images, ce qui place la dérive inter-flux dans l'intervalle [-60 ms, 100 ms]. Le RdPFT décrivant le scénario de synchronisation multimédia et le comportement de l'application ont donc des formes différentes, à cause des rendez-vous décalés et de la période de synchronisation inter-flux modifiée.

9.4.2.2. Les deux niveaux de modélisation de PNSVS

Dans ce qui précède apparaissent clairement deux niveaux de modélisation nécessaires pour les deux niveaux applicatifs. Le premier niveau correspond à l'interface dont le comportement est modélisé par le RdPFT de présentation et qui correspond à la vue de l'utilisateur du scénario de synchronisation, c'est-à-dire à la façon dont les objets audio et vidéo sont synchronisés. Le RdPFT de présentation est le même pour les deux utilisateurs : l'émetteur et le récepteur doivent manipuler les mêmes objets ayant les mêmes contraintes temporelles (cf. Figure 9.1).

Le second niveau correspond à la synchronisation applicative. La synchronisation applicative ne peut pas être modélisée en utilisant le RdPFT de présentation, à cause des caractéristiques du système opératoire et des cartes multimédias. Ces caractéristiques obligent le moteur de synchronisation à considérer les objets multimédias d'une façon très différente de l'interface : le RdPFT applicatif a donc été défini. De plus, l'émetteur et le récepteur sont différents et un modèle pour l'émetteur et un modèle pour le récepteur sont nécessaires. La Figure 9.2 représente le RdPFT applicatif pour l'entité réceptrice de PNSVS[9]. [OWEZARSKI 96b] donne la définition complète des deux modèles RdPFT, et montre comment les RdPFT applicatifs sont automatiquement déduits du RdPFT de présentation, qui est lui même déduit de la QdS demandée par les utilisateurs.

9.4.2.3. Architecture logicielle pour la synchronisation

Dans les systèmes distribués asynchrones considérés, tous les composants sont asynchrones : supports de communication, systèmes opératoires et cartes de présentation multimédias. *Aussi, les opérations de synchronisation doivent être réalisées au plus haut niveau de l'architecture du système* de visioconférence, c'est-à-dire au niveau de l'application chez le récepteur. En effet, réaliser des opérations de synchronisation terminale au niveau des supports de communication ou des protocoles de communication est inutile, car les données synchronisées dans les couches basses seront redésynchronisées en passant dans le système opératoire et dans les cartes de

[9] A noter toutefois que le modèle RdPFT de la Figure 9.2 peut encore subir des modifications. Par exemple, à cause du mode de fonctionnement des périphériques audio, on peut être amenés à réduire la taille des paquets sons pour réduire le délai de bout en bout [OWEZARSKI 96b] [OWEZARSKI 98a]. D'ailleurs, sur le récapitulatif de la Figure 9.7, il apparaît que les paquets son ont été découpés en deux dans ce but, le temps de production des données audio étant ainsi divisé par 2, et par conséquent, le délai de bout en bout aussi.

présentation multimédias. De plus, l'application (partie utilisateur du système) est le seul endroit où le développeur peut intervenir sur l'ordonnancement des processus.

Cependant, cette application qui est écrite dans l'espace utilisateur du système opératoire doit être synchrone faible, et doit donc respecter des temps de présentation maximum. Or, le système opératoire étant asynchrone, aucun temps de traitement maximum n'est garanti en classes temps partagé et système. Pour pouvoir assurer des présentations qui ne dépasseront pas leur borne maximale, il est nécessaire d'utiliser des processus dont la priorité est supérieure à celle des tâches système, et de disposer d'un système interruptible. En Solaris 2, cette classe d'ordonnancement s'appelle "Real Time" (RT). Toutefois, utiliser la classe d'ordonnancement RT est très pénalisant pour les tâches système qui ne peuvent plus s'exécuter lorsque nécessaire et qui sont différées lorsqu'un processus RT s'exécute. Les tâches temps réel doivent donc être très courtes. Par exemple, si la machine est chargée par des tâches temps réel, les opérations de communication en classe système peuvent ne pas se faire. Egalement, si un processus temps réel fait un appel système, il perd son attribut temps réel, car l'appel système le fait passer en classe système. La classe de scheduling temps réel est ainsi incontournable, mais elle doit être manipulée avec prudence [OWEZARSKI 96b].

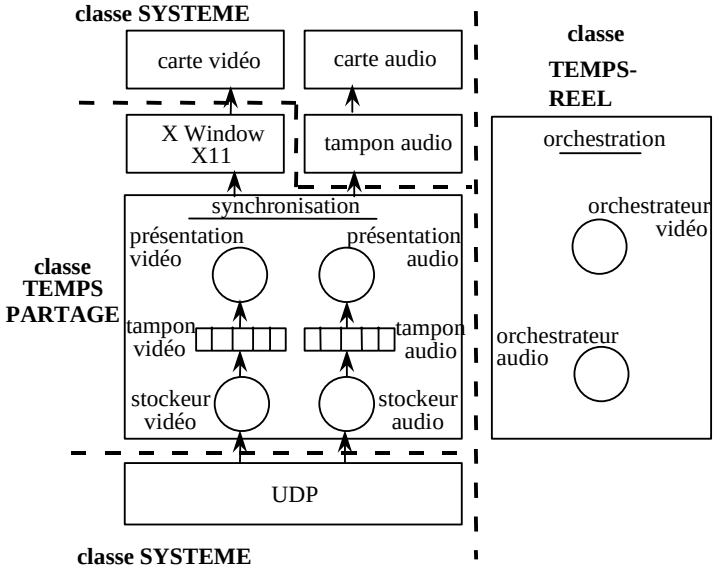

Figure 9.3. *Architecture de synchronisation pour la visioconférence*

L'architecture de l'application de visioconférence résultant de ces problèmes est représentée sur la Figure 9.3 [OWEZARSKI 95]. Elle représente les différents composants du système de visioconférence en associant à chacun d'eux leur classe d'ordonnancement dans le système opératoire. De plus, cette figure montre le découpage de l'application en sous tâches, avec :

- les stockeurs audio et vidéo qui récupèrent les données transitant sur le réseau, et qui les stockent dans les tampons audio et vidéo. Ces tampons servent à stocker temporairement les données arrivant du réseau sans aucune garantie de synchronisation, et à les conserver suffisamment longtemps pour résoudre les problèmes de gigue ;

- les processus de présentation audio et vidéo temps partagé qui réalisent les opérations et les traitements nécessaires aux présentations des sons et des images vidéo ;

- les processus d'orchestration temps réel qui jouent le scénario de synchronisation modélisé par le RdPFT applicatif récepteur et qui contrôlent les processus de présentation afin que ces derniers respectent les contraintes temporelles de présentation.

Avec ces processus, le principe de la synchronisation intra-flux repose sur un contrôle temps réel des processus de présentation par les processus d'orchestration. L'idée consiste ici à décorréler les tâches de présentation soumises à l'asynchronisme du système des tâches de contrôle temporel. Ensuite, la synchronisation inter-flux est mise en œuvre par l'intermédiaire d'un rendez vous entre les orchestrateurs audio et vidéo qui respecte la sémantique de la transition « et-maître » du RdPFT de la Figure 9.1.

9.4.3. Utilisation d'un transport à ordre partiel

9.4.3.1. Analyse de l'architecture

Les principes de gestion de la QdS précédents ont été mis en œuvre dans le cadre de l'application de visioconférence PNSVS. Des mesures des temps de présentations des objets audio et vidéo de l'application ont montré que les contraintes de synchronisation étaient parfaitement respectées [OWEZARSKI 98a]. Cependant, il est apparu un problème de pertes excessives lorsqu'un réseau non fiable est utilisé. En fait chaque perte réseau engendre beaucoup plus qu'une simple discontinuité au niveau de l'interface de présentation. En effet, si une donnée est perdue par le réseau et si cette perte est tolérable par rapport à la qualité de service demandée par l'utilisateur, alors cette perte va conduire à une duplication de la donnée précédente, ce qui d'un point de vue présentation correspond à une discontinuité (ce qui semble être la dégradation normale et minimale dans ce cas précis). Cependant, comme les processus de présentation de PNSVS [OWEZARSKI 96b] ne peuvent pas déterminer si cette donnée a été perdue ou si elle a

seulement été retardée, ils attendent au maximum avant de lancer le traitement exceptionnel correspondant à une donnée perdue ou trop retardée (jusqu'au t_{max} de la donnée précédente). Ainsi, en cas de perte, l'application n'a pas pu recouvrir l'erreur engendrée par la perte, et elle a de plus perdu du temps à attendre la donnée à traiter. *A cause de cette perte de temps, ou à l'accumulation de ce type de pertes de temps, le retard de présentation de bout en bout augmente, ce qui peut conduire à l'intervention du mécanisme de contrôle du retard par perte, et peut donc provoquer de nouvelles pertes. De même, le retard induit par l'attente d'une donnée provoque une augmentation de la dérive inter flux, et lors du tir de la transition inter flux, cela peut conduire à une accélération du flux en retard, et ainsi causer de nouvelles pertes sur ce flux.* La perte d'une image qui devrait normalement ne causer qu'une discontinuité a donc des conséquences bien plus graves.

Pour palier à ces problèmes, il faut disposer d'un mécanisme de transport qui délivre les données et détecte les pertes au plus tôt.

9.4.3.2. Une solution basée sur un transport à ordre partiel

[AMER 94] [CHASSOT 95a] [CHASSOT 95b] [DIAZ 95] définissent un nouveau protocole de transport à ordre partiel comme un transport ayant pour but de délivrer à l'utilisateur les objets transitant sur une ou plusieurs connexions, en respectant un ordre donné. Cet ordre est n'importe quel ordre compris entre un ordre total (TCP) et le non ordre (UDP), et il peut s'exprimer, en particulier, sous la forme de compositions séries et/ou parallèles des objets à transmettre. Il s'avère alors qu'un tel ordre peut en particulier être celui décrit par l'automate du RdPFT de l'application [DIAZ 94b]. Ainsi, [AMER 94] [CHASSOT 95a] [CHASSOT 95b] [DIAZ 95] définissent cette délivrance suivant un ordre prédéfini comme une synchronisation logique définie par les informations multimédias.

De plus, cette nouvelle notion d'ordre partiel est complétée par la notion de fiabilité partielle. Par rapport aux problèmes rencontrés dans PNSVS, la notion de fiabilité partielle est essentielle. Cette notion est étroitement liée à la notion de qualité de service transport qui définit une qualité de service nominale, et une qualité de service minimale en dessous de laquelle le service demandé par l'utilisateur ne sera plus rendu. Par rapport à la notion de fiabilité, cette qualité de service minimale peut s'exprimer par un nombre maximal de pertes sur une séquence, et par un nombre maximal de pertes consécutives. *Ainsi, en cas de perte acceptable, détectée lors de la réception d'un objet ordonné logiquement après l'objet attendu, l'objet initialement attendu est déclaré perdu au plus tôt (aucun essai de recouvrement n'est initié), et l'objet qui vient de parvenir à l'entité transport réceptrice est délivré à l'utilisateur (délivrance au plus tôt).* Si par contre la

perte n'est pas acceptable par rapport à la fiabilité définie par l'utilisateur, un certain nombre de retransmissions peuvent être essayées, ce nombre de retransmissions pouvant être paramétré. Tout objet reçu par l'entité transport réceptrice est donc délivré au plus tôt en accord avec l'ordre et la fiabilité partielle ; si cet objet n'est pas délivrable au regard de l'ordre partiel, il est vraisemblable qu'un problème a perturbé les objets qui le précèdent logiquement, et donc, si au regard de la fiabilité partielle les objets manquants peuvent être perdus, alors ils seront considérés comme perdus, et l'objet reçu ne sera pas retardé (délivrance au plus tôt)[AMER 93a] [AMER 93b].

En fait, il existe deux approches pour la gestion des principes de fiabilité partielle : une gestion média par média et une gestion par groupe de médias. Dans la gestion média par média, l'entité réceptrice d'un flux ne peut utiliser les mécanismes de fiabilité partielle que sur le flux qu'elle gère, et pas sur les autres flux de la multi-connexion. En revanche, avec une gestion par groupe de médias, l'entité réceptrice d'un flux peut, pour mettre en œuvre le principe de délivrance au plus tôt, provoquer des pertes sur d'autres flux au moment des synchronisations logiques inter-flux [DIAZ 95].

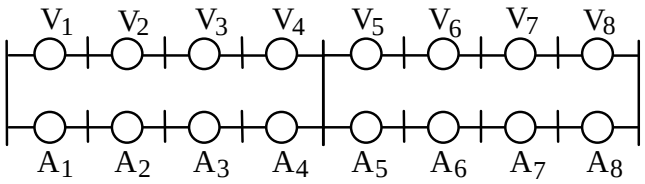

Figure 9.4. *Exemple d'ordre partiel*

Prenons pour illustrer ces propos l'exemple de la figure 9.4 qui représente le réseau de Petri des compositions série/parallèles pour une multi-connexion audio / vidéo. Supposons que le nombre de pertes maximal acceptables sur un flux et par période soit d'un élément. Supposons de plus que les objets V1, A1, A2 et A3 aient été reçus par l'entité transport réceptrice et délivrés à l'application. Supposons maintenant que l'entité réceptrice du transport reçoive l'objet V4 ; en respectant l'ordre partiel sur la connexion vidéo, la délivrance de l'objet V4 à l'utilisateur ne peut se faire, car il faudrait pour cela déclarer perdus les objets V2 et V3. Or deux pertes par période et par flux sont interdites. L'objet V4 est donc mémorisé en attendant. Si l'objet V3 arrive, il peut être délivré à condition de déclarer perdu l'objet V2. Pour ne pas ralentir les transmissions, le transport à ordre partiel délivre dès que possible les objets reçus et / ou mémorisés ; il déclare donc au plus tôt la perte de l'objet V2 (dès qu'il reçoit V3), et délivre en suivant

l'objet V3, et l'objet V4 qui avait été stocké : le principe de délivrance au plus tôt est permis grâce à une perte au plus tôt. Ceci illustre en fait une gestion par média des principes d'ordre et de fiabilité partielle.

Nous allons maintenant illustrer ces deux mêmes principes par rapport à une gestion par groupe de médias. Supposons pour cela que l'objet V5 soit reçu par l'entité réceptrice du transport. V5 n'est pas délivrable au regard de l'ordre car A4 n'est pas arrivé, et qu'il y a une synchronisation logique qui apparaît entre les deux flux audio et vidéo après les objets A4 et V4. Maintenant, si on autorise une gestion de la fiabilité par groupe de médias, l'arrivée d'un objet du flux vidéo peut provoquer la déclaration de pertes sur le flux audio, dans le cas où cela permet de délivrer plus rapidement certains objets reçus. Dans cet exemple et en mettant en œuvre ce principe, l'arrivée de V5 provoque la déclaration de perte de A4 (ce qui est autorisé au regard de la fiabilité sur le flux audio), et l'objet V5 est délivré au plus tôt. Cette gestion de la fiabilité par groupes de médias nécessite donc que l'architecture de l'entité réceptrice du transport (figure 9.5) intègre un gestionnaire de multi-connexions multimédias [CHASSOT 95c].

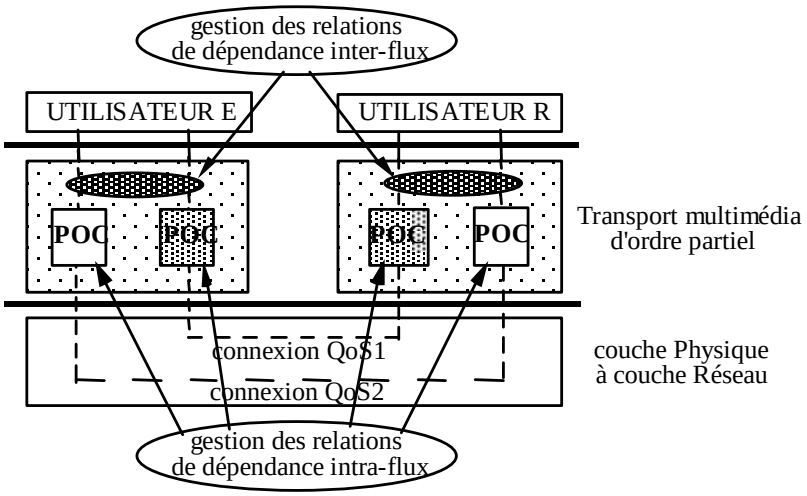

Figure 9.5. *Architecture des entités émettrice et réceptrice d'un transport à ordre partiel*

9.4.3.3. Architecture d'une application au dessus d'un transport à ordre partiel

L'architecture requise pour exécuter PNSVS au dessus d'un transport à ordre partiel est décrite sur la Figure 9.6. Cette architecture ne positionne pas directement la tâche de synchronisation au dessus de l'ordre partiel car, à cause de la gestion donnée précédemment, basée sur l'ordonnancement logique des données, le transport à ordre partiel ne peut détecter que très tard les *longues séquences de pertes*, et n'apporte dans ce cas là aucune optimisation par rapport à un transport comme UDP. Ces pertes en séquence ne peuvent pas être détectées sans une gestion explicite du temps (à l'aide d'une horloge). Aussi, la couche de pré-synchronisation a été ajoutée entre le transport et l'application pour apporter un contrôle temporel sur les données délivrées ou perdues par le Transport ; elle permet de détecter les fortes anomalies temporelles, comme les longues séquences de pertes et aussi les problèmes de dérive du réseau et les problèmes d'asynchronisme du système de communication.

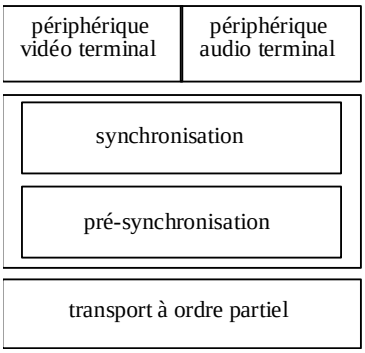

Figure 9.6. *L'architecture de PNSVS au dessus d'un service transport à ordre partiel*

D'un point de vue processus l'architecture de PNSVS 2 lorsqu'un transport à ordre partiel est utilisé est très similaire à celle de PNSVS, si ce n'est l'apparition de deux processus de pré-synchronisation qui contrôlent le comportement temporel des stockeurs, de la même façon que les orchestrateurs contrôlent le comportement temporel des processus de présentation.

9.4.3.4. Modélisation des différents niveaux de synchronisation

Il a alors pu être montré que le modèle réseau de Petri peut être réellement utilisé afin de modéliser les comportements de chacune des couches : interface utilisateur, application de synchronisation, transport, etc. et ceci avec des comportements différents

pour chaque couche. En fait, la Figure 9.7 montre comment on peut modéliser, de proche en proche, le comportement de chacune des couches en fonction de leurs fonctionnalités et des contraintes qu'elles ont à respecter dans cette architecture (à 4 niveaux).

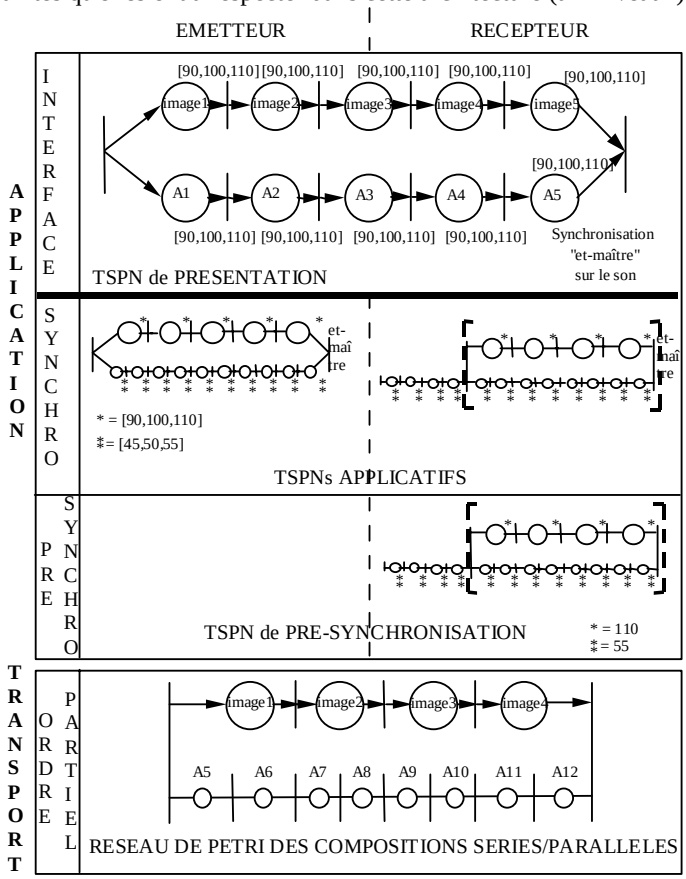

Figure 9.7 *Modélisation du comportement de chaque niveau de PNSVS 2*

9.4.3.5. Evaluation de l'architecture

PNSVS a été implémentée sur des stations Sun (Sun Sparc Station 10, 5 ou 2) avec le système Solaris 2.5. Ces machines sont équipées de cartes vidéo Parallax qui permettent de capturer, afficher, compresser et décompresser des images utilisant le format de compression M-JPEG. La carte audio est la carte Sun standard livrée avec ce type de machines. Des tests ont été réalisés sur un réseau Ethernet à 10 Mbps et sur un réseau ATM à 155 Mbps.

PNSVS est une application de visioconférence qui peut traiter 25 images/s (320 x 240 pixels x 24 bits de codage des couleurs) bi-directionnel. Le délai de présentation minimum obtenu est d'environ 400 ms et ne peut guère être réduit à cause des temps de latence audio (environ 250 ms).

Il a été montré également que les contraintes temporelles de présentation audio et vidéo (jigue et dérive) étaient parfaitement vérifiées [OWEZARSKI 98a].

De plus, les bénéfices apportés par l'utilisation d'un service transport à ordre partiel sur une application comme PNSVS ont été évalués. Ainsi, la figure 9.8 présente, en fonction du taux de perte simulé sur le réseau de test, le taux de pertes supplémentaires engendrées par l'application, pour PNSVS (qui utilise UDP) et PNSVS 2 (qui utilise un transport à ordre partiel : POC). Les courbes montrent très largement le gain en qualité de service lorsqu'un transport à ordre partiel est utilisé.

% of network losses / % of applicative losses	0	5	10	20	30	40	50	60
POC	2	0,4	0,1	0,9	1,7	2,3	2,9	3,7
UDP	2	2,4	3,7	5,5	7,4	9,1	10,9	12,7

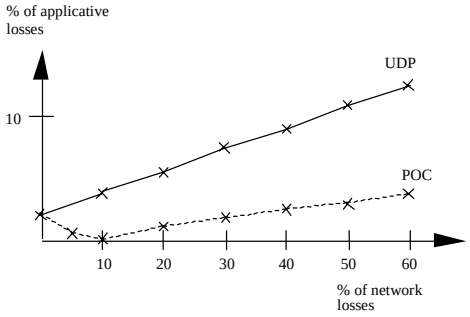

Figure 9.8. *Evaluation comparative des pertes avec POC et UDP pour PNSVS*

9.5. Renégociation de la qualité de service dans l'architecture

Dans le cas général, l'utilisateur de la visioconférence peut vouloir, en cours d'utilisation, modifier les paramètres de qualité de service de l'application (c'est à dire,

par exemple, le nombre d'images par seconde, la dérive inter-flux, ainsi que le délai de bout en bout). Ce besoin peut apparaître à la suite d'un changement des performances du réseau sous-jacent[10] ou d'un changement des besoins de l'utilisateur. Cette modification de la QdS est en fait une modification des contraintes temporelles entre les données, c'est à dire du RdPFT modélisant la QdS utilisateur.

La solution simpliste serait de quitter l'application puis de relancer la visioconférence avec une nouvelle QdS. Cette solution n'est bien sûr pas acceptable. Dans cette partie, nous allons proposer une architecture globale de commutation de QdS, qui permet de modifier la QdS à tout moment, sans interrompre ni dégrader le fonctionnement de l'application, la commutation étant mise en œuvre en un temps très court (à l'échelle humaine).

9.5.1. *Contraintes et architecture de renégociation déduite*

Nous allons ici faire une étude des nouvelles contraintes de l'application de visioconférence, qui est en fait une représentante des applications multimédia distribuées à fortes contraintes temporelles (transportant des flux vivants[11]).

9.5.1.1. Besoins et contraintes

Regardons les besoins spécifiques de la renégociation de QdS. Le premier besoin vient de l'utilisateur : le temps écoulé entre le moment où il demande la renégociation et le moment ou la nouvelle QdS est effective doit être court. L'application elle aussi à un besoin important : elle doit présenter les données avec la QdS de leur création.

Une fois ces besoins précisés, nous allons pourvoir étudier leur influence sur l'architecture.

La QdS de l'application est spécifiée par un RdPFT, qui constitue le motif de base (Figures 9.1 et 9.2). L'application présente une mise en séquence de ce motif de base. Lorsque l'on veut changer de QdS, on va en fait passer de ce motif à un autre. Le flux

[10] Une augmentation du taux de perte peut conduire à vouloir augmenter le taux de perte acceptable, afin d'éviter les retransmissions et préserver le délais de bout en bout.

[11] On appelle flux « vivant » un flux dont les données sont produites au fur et à mesure (comme une conférence), par opposition aux flux « morts » qui transmettent des données stockées (comme un serveur vidéo).

présenté sera donc une séquence du motif initial mise en séquence avec une séquence du nouveau motif. La *respect de la QdS de chaque donnée* implique bien sûr que les données soient traitées en réception suivant la QdS avec laquelle elles ont été émises. De façon évidente, il faut que l'émetteur et le récepteur connaissent la nouvelle QdS. Il devront donc dialoguer. Pour se faire, il va falloir mettre en place deux agents de renégociation de la QdS (qui utiliseront une connexion fiable, de type TCP). De plus, le changement de QdS ne peut pas se faire n'importe quand, mais doit se faire entre deux motifs.

Mais comment l'émetteur et le récepteur peuvent-ils s'accorder sur quel sera le dernier motif ayant l'ancienne QdS et quel sera le premier avec la nouvelle ? Une solution simpliste consisterait pour l'émetteur à cesser l'émission après le denier motif relevant de l'ancienne QdS, à envoyer un message de changement de QdS, à s'assurer qu'il a bien été reçu, puis à recommencer à émettre avec la nouvelle QdS. Cette solution est inacceptable car elle viole l'exigence de *continuité du service* ainsi que celle de la *réactivité à la requête de l'utilisateur*. Une autre possibilité est pour les agents émetteur et récepteur de se mettre d'accord par avance sur un moment de commutation[12], par une communication sur leur connexion de renégociation. Le problème est que l'on est jamais sûr d'arriver à se mettre d'accord en un temps fini sur un réseau non fiable ou non borné (IP ou TCP). En effet, si l'émetteur décide de commuter dans « n » motifs, il doit indiquer son choix au récepteur, mais n'est jamais sûr que son message arrive avant que le flux ne change de QdS. Si le récepteur envoie un message indiquant « n » et attend une confirmation, il se peut que le moment de commutation arrive sans qu'il ait reçu de confirmation : il ne sait alors pas si le récepteur n'a rien reçu (et qu'il ne doit pas commuter) ou a bien reçu « n » mais que l'accusé de réception est perdu (dans ce cas, il doit commuter). Ces exemples sont simplement là pour donner une idée du problème : plus formellement, c'est l'impossibilité du consensus sur réseau non borné, qui est en cause. On peut vouloir contourner le problème de façon statistique : en choisissant assez à l'avance le moment de commutation, on est à peu près sûr que le message arrivera à temps. Cette solution cependant ne peut pas être retenue car : *primo*, elle néglige le fait que si on change de QdS, c'est peut-être justement que le réseau est engorgé, et que notre message va avoir du mal à passer ; *secundo*, elle introduit un temps relativement grand entre le temps où la demande de renégociation de QdS a lieu et le moment où elle est effective ; *tertio*, elle menace la stabilité du protocole de transport POC, qui peut

[12] En fait, le choix d'un moment de commutation se réduit au choix d'un indice, puisque les données sont numérotées.

recevoir des données avec des entêtes calculés pour coder un ordre donné, et les interpréter avec un autre codage[13].

La solution retenue est en fait de faire un codage *in-band* du changement de QdS, c'est à dire d'ajouter aux données en transit un entête relatif à la QdS en cours. Nous pouvons nous permettre cet ajout car on peut le coder sur quelques bits, ce qui n'est pas grand chose pour un trafic multimédia. Bien sûr, ce codage ne remplace pas la négociation de la nouvelle QdS entre les deux agents. La renégociation a d'abord lieu, sur la connexion de contrôle, puis ensuite vient la commutation, codée avec les données.

Il est bien sûr impératif que la commutation d'une QdS à une autre soit la plus rapide possible, afin de préserver le *délai de bout en bout* et la *réactivité à la requête de l'utilisateur*. Si la nouvelle QdS demande des ressources supplémentaires[14], elles doivent être réservées avant de faire la commutation. Ceci décompose donc la renégociation en deux phases distinctes : la *négociation* au cours de laquelle les deux agents vont négocier une nouvelle QdS et réserver les ressources associées, puis, la *commutation* pendant laquelle chaque entité va modifier sa QdS courante. Notons que pour que l'émetteur puisse commuter, il doit recevoir un message du récepteur lui signifiant que les ressources sont en place.

9.5.1.2. Une commutation coordonnée

Etudions la commutation elle-même : elle doit être unique du point de vue de l'utilisateur, mais est en fait multiple du point de vue de la machine.

En effet, les différentes couches ne travaillent pas simultanément sur les mêmes données : si une couche traite la donnée d'indice «n», la couche inférieure devrait être en train de préparer la «n+p» (p>0). Chaque couche doit donc gérer sa propre commutation.

De plus, a l'intérieur d'une même couche, du fait du traitement parallèle, un thread de flux audio peut être en train de traiter une donnée d'indice «n» pendant que le thread de la même couche finit de traiter la donnée précédente d'indice «n-1». Chaque pile doit donc traiter sa propre commutation de QdS. Donc, l'architecture de commutation de QdS doit se calquer sur l'architecture multimédia de l'application (par couche et par pile).

9.5.1.3. Architecture déduite

---

[13] Cette probabilité est faible si on a introduit un assez grand délai, mais elle existe.
[14] comme de la mémoire, ou l'ouverture d'une nouvelle connexion POC

L'architecture déduite des contraintes est celle de la Figure 9.9. Elle est calquée sur l'architecture statique présentée dans les Figures 9.3 et 9.6. On y retrouve l'agent de négociation de QdS (sur TCP) ajouté à l'architecture.

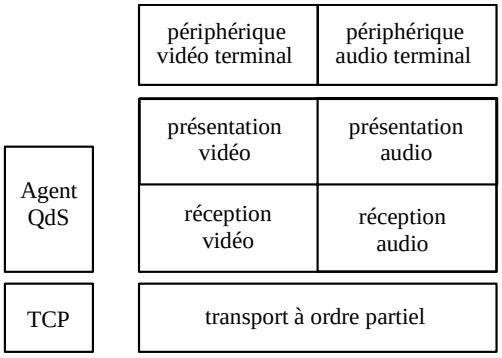

Figure 9.9. *L'architecture de renégociation de Qualité de Service de PNSVS*

Par rapport à la Figure 9.3, chaque orchestrateur a été regroupé avec sa présentation, puisqu'ils ont toujours la même QdS. La couche de pré-synchronisation, présentée à la figure 9.6, a elle été regroupée avec celle de stockage, pour la même raison. Comme présenté dans l'étude des contraintes, on a séparé les piles de traitement audio et vidéo.

Cette architecture n'est pas très parlante sans la présentation du protocole de renégociation de QdS associé.

9.5.2. Protocole de renégociation dynamique de QdS

Comme nous l'avons déjà dit, la renégociation se décompose en une *négociation* puis une *commutation*. La négociation est présentée sur la Figure 9.10. Elle commence par un dialogue entre les deux agents pour définir la nouvelle QdS[15], suivi de la réservation des

[15] Le dialogue lui-même peut prendre plusieurs formes : à l'initiative de l'un ou de l'autre, avec de multiples propositions ou une seule… Ce n'est pas le cœur du problème. Dans notre exemple, nous avons choisi de laisser le changement de QdS à l'initiative du récepteur, car c'est généralement lui qui a la meilleure vue de la QdS réellement fournie, mais on doit aussi laisser la possibilité à l'émetteur d'être initiateur de cette renégociation.

ressources associées (en fait, ces deux étapes sont mêlées, puisque pour s'assurer que des ressources sont disponibles, un agent va généralement tenter de les réserver, puis regarder la réponse du système). Elle se termine par un message `Ressources-OK` du récepteur vers l'émetteur. Lorsque le récepteur a enfin et reçu ce message et finit sa propre réservation, il peut commuter quand il le souhaite.

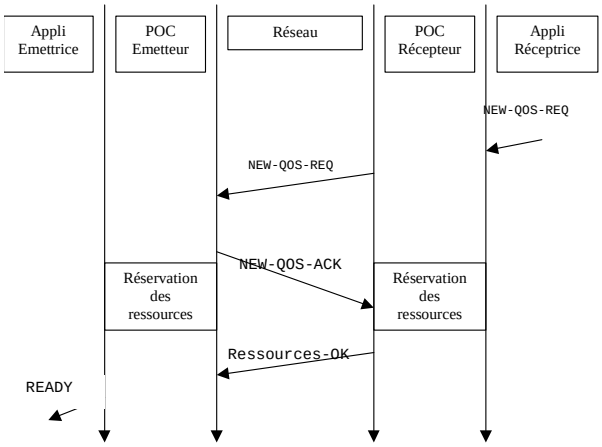

Figure 9.10. *Protocole de renégociation*

La commutation elle, se fait de façon très simple : dès qu'un module reçoit des données associées à la nouvelle QdS, il commute de l'ancienne à la nouvelle QdS (on notera que, comme les motifs associés aux deux QdS sont strictement en séquence, aucun module ne peut recevoir des données de la nouvelle QdS alors qu'il lui reste à traiter des données de l'ancienne).

9.5.3. *Evaluation de performances*

L'implémentation de la renégociation dynamique de QdS, présentée dans [OWEZARSKI 97] a consisté à offrir à l'utilisateur la possibilité de modifier le nombre de sons et d'images par seconde traité par l'application (ce qui donne lieu à un changement de l'ordre entre les données, puisqu'une synchronisation inter-flux est réalisée chaque seconde). Cette implémentation a permit de valider les principes, par des mesures de performances [BOYER 98, OWEZARSKI 98b], comme présenté sur la Figure 9.11.

La Figure 9.11 présente les dates de présentation (en seconde) des données en fonction de leur numéro de séquence. On remarque bien les changements dans la pente de la courbe qui correspondent aux changements de QdS.

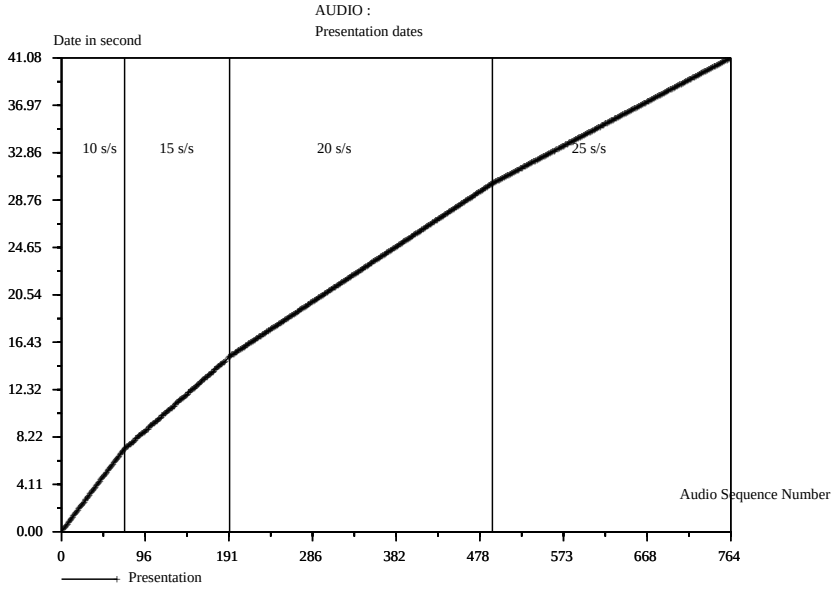

Figure 9.11. *Evolution du taux de présentation en fonction de la QdS*

Les diverses mesures présentées dans [OWEZARSKI 97] montrent avec plus de détails que les différentes contraintes de la renégociations de QdS on été complètement satisfaites par l'implémentation.

Un autre aspect bien plus difficile à quantifier est l'évaluation de l'aide apportée par la modélisation du système. En effet, dans cette expérimentation, le modèle n'a pas servi à la validation des contraintes, ni à la génération automatique de code, mais comme support de description des comportements temporels.

Pour information, le code du système PNSVS comportait, du côté du récepteur, environ 2000 lignes de codes, 9 types d'agents (threads) à synchroniser, grâce à 4 types d'objets de synchronisations, des threads en classe « temps réel » et d'autres en classe

« utilisateur », le tout soumis à des contraintes temporelles de l'ordre de 10ms. La conception et le déboguage de ce système complexe ont été grandement facilités par la description formelle des contraintes de synchronisation du système au moyen d'un modèle visuel et compact.

9.6. Conclusion

Nous avons montré dans cette étude comment l'utilisation d'un modèle formel, basé sur les réseaux de Petri, a facilité la conception d'une application complexe, et d'abord particulièrement l'appréhension des contraintes temporelles du système et des actions à mettre en œuvre pour les satisfaire.

Cette modélisation, associée à une bonne connaissance du système sous-jacent, a permis de développer une application aux très bonnes performances et à la parfaite maîtrise de la qualité de service. De plus, cette aide à l'appréhension des contraintes temporelles nous a ouvert la possibilité d'aller plus loin en proposant une renégociation dynamique de la qualité de service, tout à fait cohérente avec l'approche méthodologique de haut niveau proposée, et aux résultats eux aussi tout à fait satisfaisants.

Bibliographie

[ADELBERG 94] ADELBERG B., GARCIA-MOLINA H., KAO B., "Emulating Soft Real Time Scheduling Using Traditional Operating System Schedulers", Proceedings of the Real Time Systems Symposium, San Juan, Puerto Rico, December 7-9, 1994

[AMER 93A] AMER P.D., CHASSOT C., CONNOLLY T., DIAZ M., "Partial order transport service for multimedia applications : reliable service", proceedings of the second High Performance Distributed Computing Conference, July 1993

[AMER 93B] AMER P.D., CHASSOT C., CONNOLLY T., DIAZ M., "Partial order transport service for multimedia applications : unreliable service", proceedings of the 3rd International Networking Conference, INET'93, August 1993

[AMER 94] AMER P.D., CHASSOT C., CONNOLLY T., CONRAD P., DIAZ M., "Partial order transport service for multimedia and other applications", IEEE/ACM transactions on Networking, Vol. 2, No. 5, October 1994

[BAKER 94] BAKER T.P., MUELLER F., RUSTAGI V., "Experience with a Prototype of the POSIX Minimal Realtime System Profile", IEEE Newsletter on Real Time Systems, 1994

[BLAKOWSKI 95] BLAKOWSKI G. et STEINMETZ R., "A media synchronization survey : reference model, specification and case studies", *IEEE journal on selected areas in communications*, Vol. 14, No. 1, January 1996.

[BLAKOWSKI 96] BLAKOWSKI G., STEINMETZ R., "A media synchronization survey: reference model, specification, and case studies", IEEE journal on selected areas in communications, Vol. 14, No. 1, January 1996

[BOYER 98] BOYER M., OWEZARSKI P. et DIAZ M., "Dynamic QoS Renegociation in the PNSVS Videoconferencing Application", Proceedings of the Fifth International Workshop on Interactive

Distributed Multimedia Systems and Telecommunication Services ; IDMS'98 , Oslo, Norway, September 1998.

[CHASSOT 95A] CHASSOT C., DIAZ M., LOZES A., "From the partial order concept to partial order multimedia connections", Journal of high speed network (JHSN), 1995

[CHASSOT 95B] CHASSOT C., DIAZ M., LOZES A., "Principes d'implantation d'une connexion multimédia d'ordre partiel", 4ème colloque francophone sur l'ingéniérie des protocoles, CFIP'95, Rennes, France, 10-12 mai 1995

[CHASSOT 95C] CHASSOT C., "Architecture de transport multimédia à connexions d'ordre partiel", Thèse de doctorat de l'institut National Polytechnique de Toulouse (INPT), Décembre 1995

[COULSON 94] COULSON G., BLAIR G.S., ROBIN P., "Micro-kernel support for continuous media in distributed systems", Computer Networks and ISDN systems, Vol. 26, 1994

[COURTIAT 96] COURTIAT J.P., DE OLIVEIRA R.C., "Proving temporal consistency in a new multimedia model", Proceedings of ACM multimedia'96, Boston, November 18-22, 1996

[DIAZ 93A] DIAZ M., SÉNAC P., "Time stream Petri nets, a model for multimedia streams synchronization", proceedings of MultiMedia modelling'93, Singapore, November 1993

[DIAZ 93B] DIAZ M., SENAC P., DE SAQUI-SANNES P., "Un modèle formel pour la spécification de la synchronisation multimédia en environnement distribué", Actes du colloque francophone sur l'ingénierie des protocoles, Montréal, Canada, 7-9 septembre 1993

[DIAZ 94A] DIAZ M., OWEZARSKI P., "Développement d'un système de visioconférence sur réseau local", Rapport LAAS No. 94354, Juillet 1994

[DIAZ 94B] DIAZ M., LOZES A., CHASSOT C., AMER P.D., "Partial order connections : A new concept for high speed and multimedia services and protocols", Annals of telecommunications, Vol. 49, No. 5-6, May-June, 1994

[DIAZ 95] DIAZ M., DRIRA K., LOZES A., CHASSOT C., "Definition and representation of the quality of service for multimedia systems", 6th International conference on high speed Networking, HPN'95, Palma de Mallorca (Balearic Islands), Spain, September 11-15, 1995

[ISO 90] ISO-IEC JTC1/SC2/WG12, "Coded representation of multimedia and hypermedia information", Multimedia and hypermedia information coding expert group (MHEG), working document version 2, July 1990

[ISO 93] ISO, "Coded representation of multimedia and hypermedia information objects (MHEG)", Committee draft, ISO/IEC CD 13522-1, June 1993

[JEFFAY 92] JEFFAY K., STONE D.L., DONELSON SMITH F., "Kernel support for live digital audio and video", Computer communications, Vol. 15, No. 6, July / August 1992

[JEFFAY 94A] JEFFAY K., STONE D.L., DONELSON SMITH F., "Transport and display mechanisms for multimedia conferencing across packet-switched networks", Computer networks and ISDN systems, Vol. 26, 1994

[JEFFAY 94B] JEFFAY K., "On latency management in Time-Shared Operating Systems", IEEE Newsletter on real time systems, 1994

[KANG 94] KANG G. SHIN, PARAMESWARAN RAMANATHAN, "Real time computing : a new discipline of computer science engineering", Proceedings of the IEEE, Vol. 82, No. 1, January 1994

[KATCHER 94] KATCHER D.I., KETTLER K.A., STROSNIDER J.K., "Modeling DSP Operating Systems for Multimedia Applications", Proceedings of the Real Time Systems Symposium, San Juan, Puerto Rico, December 7-9, 1994

[LE GALL 91] LE GALL D., "MPEG: a video compression standard for multimedia applications", Comunications of the ACM, Vol. 34, No. 4, April 1991

[LIOU 91] LIOU M., "Overview of the px64 kbits/s video coding standard", Communications of the ACM, Vol. 34, No. 4, April 1991

[OWEZARSKI 95] OWEZARSKI P., DIAZ M. et SENAC P., "Modélisation et implémentation de mécanismes de synchronisation multimédia dans une application de visioconférence", Actes du colloque francophone sur l'ingéniérie des protocoles (CFIP'95), Rennes, France, 1995.

[OWEZARSKI 96a] OWEZARSKI P.ET DIAZ M., "Models for enforcing multimedia synchronization in visioconference applications", Proceedings of the 3rd MultiMedia Modeling conference – Towards the information superhighway (MMM'96), World Scientific editor, Toulouse, France, 1996.

[OWEZARSKI 96b] OWEZARSKI P., "Conception et formalisation d'une application de visioconférence coopérative. Application et extension pour la téléformation", thèse de doctorat, Université de Toulouse III, France, décembre, 1996.

[OWEZARSKI 97] OWEZARSKI P., Boyer M. et Diaz M., "Renégociation dynamique de qualité de service dans une application de visioconférence synchronisée", actes du Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'97), 1997.

[OWEZARSKI 98a] OWEZARSKI P., DIAZ M. ET CHASSOT C., "A Time Efficient Architecture For Multimedia Applications", *IEEE Journal on Selected Areas in Communication JSAC.*, April 1998, vol.16, n°3, p. 383-396.

[OWEZARSKI 98b] OWEZARSKI P., Boyer M. et Diaz M., "Mécanismes de gestion et de renégociation de la qualité de service dans une application de visioconférence", *Revue Electronique sur les Réseaux et l'Informatique Répartie (RERIR),*n°7, Septembre 1998.

[ROTHERMEL 95] ROTHERMEL K., HELBIG T., "An Adaptative Stream Synchronization Protocol", proceedings of the Network and Operating System Support for Digital Audio and Video coference (NOSSDAV'95), 1995

[SÉNAC 94] SÉNAC P., DIAZ M., DE SAQUI-SANNES P., "Toward a formal specification of multimedia synchronization", Annals of telecommunications, May / June 1994

[SÉNAC 96] SÉNAC P., DIAZ M., LÉGER A. ET DE SAQUI-SANNES P., "Modeling logical and temporal synchronization in Hypermedia systems", *IEEE Journal on Selected Areas in Communications JSAC.*, 1996, p. 84-103.

[TURLETTI 93] TURLETTI T., "H.261 software codec for videoconferencing over the internet", Rapport de recherche INRIA No. 1834, Janvier 1993

[VOGEL 95] VOGEL A., KERHERVÉ B., VON BOCHMANN G., GECSEI J., "Distributed Multimedia and QoS : A Survey", IEEE Multimedia, Summer 1995

[WALLACE 91] WALLACE G.K., "The JPEG picture compression standard", Communications of the ACM, Vol. 34, No. 4, April 1991