Architecture de transport multimedia à connexions d'ordre partiel

Christophe Chassot

Année 1995

THÈSE

présentée au
Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS
en vue de l'obtention du titre de
Docteur de l'Institut National Polytechnique de Toulouse
Spécialité : **Informatique**

par **Christophe CHASSOT**

ARCHITECTURE DE TRANSPORT MULTIMÉDIA A CONNEXIONS D'ORDRE PARTIEL

Soutenue le 21 Décembre 1995, devant le Jury :

MM.	**P. AMER**	Rapporteur
	D. BONJOUR	Examinateur
	R. CASTANET	Examinateur
	M. DIAZ	Directeur de Thèse
	S. FDIDA	Rapporteur
	G. JUANOLE	Président du Jury
	G. PADIOU	Examinateur

Rapport LAAS N°95430
Thèse préparée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS
7, Avenue du Colonel Roche 31077 TOULOUSE Cedex FRANCE.

A mes parents, à mes amis

REMERCIEMENTS

Les travaux relatés dans ce mémoire ont été effectués au Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS) du Centre National de la Recherche Scientifique (CNRS), dirigé au cours de mon séjour par Monsieur A. Costes, que je tiens à remercier cordialement pour son accueil.

J'adresse mes plus sincères remerciements à Monsieur Michel Diaz, Directeur de Recherche au CNRS et responsable du groupe Outils Logiciels pour la Communication (OLC), qui m'a accepté au sein de son équipe de recherche, et qui a encadré ma thèse durant ces trois dernières années.

Je remercie également Monsieur Joseph Noailles, Professeur à l'Institut National Polytechnique de Toulouse (INP), de m'avoir accepté dans la formation doctorale Informatique Fondamentale et Parallélisme.

Je suis très reconnaissant à Messieurs :

Paul Amer, Professeur à l'Université de Delaware,
Dominique Bonjour, Ingénieur au CNET,
Richard Castanet, Professeur à l'Université de Bordeaux,
Serge Fdida, Professeur à l'Université Pierre et Marie Curie de Paris,
Guy Juanole, Professeur à l'Université Paul Sabatier de Toulouse,
Gérard Padiou, Professeur à l'INP de Toulouse (ENSEEIHT),

pour l'honneur qu'il m'ont fait en participant au jury de thèse. Je tiens à remercier en particulier Messieurs P. Amer et S. Fdida qui ont accepté d'être les rapporteurs de mes travaux, et avec qui j'ai eu la chance de travailler durant ma thèse.

Le travail présenté dans ce mémoire a été réalisé grâce au support financier du Centre National d'Étude des Télécommunications (CNET) et du Centre Commun d'Étude des Télédiffusions et Télécommunications (CCETT) dans le cadre du projet CESAME. Je remercie tous les participants de ce projet que j'ai pu rencontrer lors de ces trois dernières années.

Je remercie chaleureusement tous les collègues de bureau que j'ai côtoyés durant ces trois années, pour leurs conseils et leur soutient : André Lozes, à qui pas une ligne de cette thèse n'est inconnue, Thierry Villemur, Philippe Owezarski, Michel Fournier, Pierre Gradit, Bernard Meunier. A travers eux, mes remerciement s'étendent à tous les membres du groupe OLC, en particulier à Brigitte Pradin, Khalil Drira, François Vernadat, François Michel, Pierre Azéma, Laurent Gallon pour ce qu'ils m'ont apporté comme soutient moral ou matériel.

Je remercie le service Informatique et Instrumentation pour le support (et le réconfort) qu'ils m'ont apporté : Danielle Barthes, Marie Do. Cabanne, Marie Jeanne Laubies, Martine Aguera, Jean Michel Pons.

J'adresse également mes remerciements à tous les membres du service de Documentation-Edition, qui a permis la réalisation de ce mémoire ; à travers eux, je remercie l'ensemble du personnel administratif.

Je tiens enfin à remercier tous les amis qui m'ont écouté, encouragé, porté, soûlé, ... pendant les mois de rédaction. J'aimerais qu'ils sachent combien je leur suis reconnaissant d'avoir été présents durant cette période : Nathalie, Jean-Yves, Frédéric, Laurent, Babé, Vincent, ..., André, Brigitte, Danielle, à vous tous : merci.

TABLE DES MATIÈRES

INTRODUCTION GÉNÉRALE

Jusque dans les années 80, les mondes des Télécommunications et de l'Informatique ont évolué en parallèle. Les Télécommunications ont tout d'abord développé un service de transmission de la voix - le téléphone - basé sur une technique garantissant l'isochronisme et la quasi fiabilité des transferts de données. L'émergence des réseaux informatiques a par la suite autorisé le développement d'applications distribuées entre ordinateurs distants. Le taux d'erreur relativement élevé et la faible bande passante des supports de communication ont limité les types d'applications à des transferts de fichiers et d'images fixes. Les contraintes résultantes ont entraîné le développement des services et protocoles de communication actuels.

Évolutions et problématique future

Durant la dernière décennie, les évolutions technologiques ont conduit au rapprochement de l'Informatique et des Télécommunications ; les Télécommunications assurent dorénavant la transmission de tous les types de données, à des vitesses sans cesse croissantes. Parallèlement, l'augmentation de la puissance des ordinateurs fait qu'il est maintenant possible de manipuler et d'intégrer sur une même machine, données classiques, images et son. Cette situation ouvre les portes à de nouvelles applications, multimédia, nécessitant le traitement informatique et le transfert *en temps réel* de tous les types de données : la problématique générale consiste alors à concevoir comment coordonner ces différentes données, comme si elles étaient disponibles localement.

Les systèmes de communication supports de ces applications sont à l'heure actuelle sujets de réflexion et de recherche. De nombreux problèmes sont à résoudre, parmi lesquels les différences d'exigence en transmission et en présentation des données manipulées, les contraintes temporelles des applications, leur aspect potentiellement coopératif, ou encore l'optimisation des ressources utilisées dans les réseaux de distribution.
Dans ce cadre, de nombreuses approches sont envisageables selon les garanties temporelles du support de communication utilisé ou les caractéristiques de la machine dans laquelle sont implantés les logiciels. Le cas le plus général comprend les architectures distribuées asynchrones dans leur globalité, c'est-à-dire constituées de réseaux sans garantie de délai de transmission, et de systèmes d'exploitation sans garantie de temps de traitement.

Quel que soit l'approche envisagée, la complexité du processus de conception d'un système réparti impose l'utilisation de méthodes formelles pour la représentation du comportement distribué ; plusieurs techniques de description formelle existent à ce jour comme les standards internationaux LOTOS, Estelle ou SDL. Cependant, l'ajout de contraintes supplémentaires, notamment liées aux besoins en synchronisation des applications multimédia, impose que de nouvelles approches formelles soient adoptées afin que ces contraintes puissent être spécifiées par l'utilisateur, et que le système puisse en vérifier la cohérence et mettre en œuvre des mécanismes de gestion appropriés. Dans cette optique, de nombreux modèles ont été développés ces dernières années, notamment basés sur le formalisme des réseaux de Petri ; le modèle TSPN est l'un des représentants les plus évolués de ces modèles.

Face aux inadéquations des protocoles de Transports existants, plusieurs projets nationaux et internationaux (BERKOM en Allemagne, QoS-A en Angleterre, ...) intègrent dans leurs recherches la conception d'une architecture de Transport multimédia à qualités de service (QoS) sélectionnables par l'application et négociables entre utilisateur et fournisseur du service. Les évolutions envisagées de cette QoS se traduisent le plus souvent par l'adjonction de paramètres temporels, et la définition de nouvelles sémantiques de service indiquant clairement au pourvoyeur l'action à entreprendre en cas de dégradation de la QoS négociée. Cependant, les mécanismes de Transport associés ne sont envisagés que dans des cadres d'implantation très spécifiques, ou se limitent à une observation de la QoS telle qu'elle est fournie par le réseau.

Contributions exposées dans ce mémoire

Initié en 1992 dans le cadre d'un partenariat entre le CNET (Centre National d'Étude des Télécommunication), le CCETT (Centre Commun d'Étude des Télédiffusions et Télécommunications) et le CNRS, le projet CESAME (Conception formElle de Systèmes distribués coopérAtifs hauts débits MultimEdia) s'inscrit dans ce contexte général, et vise à concevoir une nouvelle architecture de communication, intégrant la prise en compte des besoins applicatifs à différents niveaux conceptuels. Cinq thèmes de recherche y sont clairement définis :

- l'étude et la proposition de nouveaux services et protocoles de communication, utilisant des réseaux à haut débit ;
- l'analyse des contraintes de synchronisation, en particulier temporelles, dans les objets et les systèmes multimédia ;
- l'étude des échanges dans le cadre de groupes coopératifs d'utilisateurs ;
- le développement d'applications illustratives, telles que la visioconférence et le télé-enseignement, intégrant les travaux précédemment décrits ;

- la représentation formelle des mécanismes et des protocoles en vue de l'évaluation de leurs performances et de leur test temporel.

Les travaux présentés dans ce mémoire s'inscrivent dans le premier axe de recherche, et proposent la définition d'une connexion, d'une architecture et d'un service de Transport multimédia, à garantie de qualités de service. Notre contribution est composée de trois parties majeures.

1 - Actuellement, les services et protocoles de transfert de données sont basés sur l'alternative conceptuelle entre le mode orienté-connexion et le mode sans connexion ; en conséquence, les services de transfert de données présentent soit une garantie d'ordre total (les données sont délivrées dans leur ordre de soumission par l'émetteur), soit aucune garantie (l'ordre de délivrance résulte des déséquencements et des pertes introduits par le réseau sous-jacent). Au regard du degré de parallélisme que présentent les flux multimédia, nous proposons d'étendre cette qualité de service jusqu'alors implicitement fournie, en offrant à l'utilisateur la possibilité de sélectionner tous les services d'ordres imaginables entre l'ordre total et le "désordre", c'est-à-dire les ordres partiels. Celui-ci peut en particulier choisir l'ordre partiel le plus proche de ses contraintes minimales, résultant de la disposition spatiale et/ou de la synchronisation des différents média lors de leur restitution finale.
L'extension de la notion d'ordre des services et protocoles traditionnels nous conduit ainsi à définir un nouveau type de connexion, la *connexion d'ordre partiel*, implémentant tous les services et protocoles d'ordre partiel, depuis l'ordre total jusqu'au désordre. De façon analogue, nous caractérisons le concept de *connexion partiellement fiable*, offrant une qualité de service en terme de fiabilité, paramétrable par l'utilisateur, et garantie par le fournisseur du service.

2 - Dans un deuxième temps, notre contribution porte sur la définition d'une architecture de Transport multimédia, impliquant l'instanciation et la coordination de plusieurs connexions monomédia, chacune à qualités de service distinctes. Le concept de connexion d'ordre partiel y est intégré à deux niveaux : dans chaque connexion et entre les connexions. L'ordre partiel multimédia découle du réseau de Petri utilisé au niveau applicatif dans le cadre d'une modélisation TSPN des contraintes de synchronisation de l'objet multimédia. En cela, le service de délivrance que nous proposons respecte les impératifs chronologiques déduits de la synchronisation temporelle.
L'intégration de la notion de fiabilité partielle[1] nous permet de proposer deux stratégies de délivrance des données "au plus tôt", au prix d'une dégradation acceptable de la fiabilité. Nous proposons alors la définition d'un service de Transport multimédia d'ordre

[1] Communément, un protocole est dit "fiable" lorsqu'il garantit la délivrance des données sans altération, ni perte, ni duplicata.

partiel (MM-POS : Multimedia Partial Order Transport Service), à qualités de service garanties en terme d'ordre et de fiabilité. A la différence des services définis dans d'autres projets, le service MM-POS ne considère pas toutes les garanties de qualités de service temporelles ; ce choix est motivé par le contexte asynchrone de bout en bout que nous étudions, interdisant la possibilité de garantir de façon satisfaisante ce type de QoS.

3 - Ces dernières années, l'approche la plus prometteuse dans la conception des protocoles de communication haute vitesse a conduit à la spécification du protocole XTP (Xpress Transfer Protocol) ; outre des fonctionnalités de niveau réseau, XTP offre un ensemble de procédures de niveau Transport, paramétrables par l'utilisateur, lui permettant de concevoir le protocole le plus adapté à ses besoins. La dernière partie de notre contribution concerne l'analyse de la mise en œuvre d'un protocole offrant les fonctionnalités décrites dans le service MM-POS à l'aide des procédures XTP.

Outre l'introduction et la conclusion générales, ce manuscrit est composé de cinq chapitres.

Le ***chapitre 1*** introduit la problématique générale dans laquelle se situe notre contribution. Nous y présentons les exigences des applications multimédia, et les approches développées quant à leur spécification et leur prise en compte. Les technologies de communication les plus récentes sont évoquées, et les évolutions envisagées des protocoles de Transport dans ce nouveau contexte sont exposées.

Le ***chapitre 2*** présente un état de l'art détaillé du niveau Transport. Cette étude se focalise tout d'abord sur la description des mécanismes implantés dans la majorité des protocoles de Transport ; leur mise en œuvre est illustrée à l'intérieur de sept protocoles : VMTP, NETBLT, Delta-t, TCP, OSI TP4, XTP et Datakit. Dans un second temps sont analysées les limites des services de Transport actuels ; les évolutions envisagées sont exposées et discutées au travers de la proposition OSI 95. Les problèmes relatifs à la prise en compte des besoins applicatifs sont mis en évidence par le biais de la notion de qualité de service. Nous ponctuons ce chapitre par une présentation de plusieurs projets nationaux et internationaux, parmi lesquels le projet CESAME.

Au ***chapitre 3*** est présenté le concept de *connexion d'ordre partiel*. Après avoir souligné les lacunes conceptuelles des services et protocoles traditionnels, nous analysons l'adéquation d'une connexion d'ordre partiel à exploiter le degré de parallélisme des objets multimédia. Nous montrons ensuite comment la mise en œuvre d'un service d'ordre partiel engendre une diminution de l'utilisation des ressources en terme de mémoire et de bande passante ; il apparaît de plus qu'un tel service permet de diminuer globalement le délai de transit des données utilisateur.

Par leurs principes de conception, les protocoles de Transport actuels ne garantissent que deux types de fiabilité : totale (toutes les pertes sont récupérées), ou nulle (les pertes ne sont pas récupérées). La nature même d'un objet multimédia conduit à considérer comme acceptable la perte de certaines informations. Partant de ce constat, nous étendons la notion de connexion d'ordre partiel à celle de *connexion d'ordre partiel et de fiabilité partielle*, implémentant tous les services d'ordre partiel et de fiabilité partielle.

Le ***chapitre 4*** présente les principes d'implantation d'une nouvelle architecture de Transport, multimédia, intégrant les notions d'ordre et de fiabilité précédemment exposées. Une connexion multimédia d'ordre partiel implique l'instanciation et la coordination de plusieurs connexions d'ordre partiel monomédia, chacune à qualités de services données. De façon à offrir un service de synchronisation logique intra-flux et inter-flux, la gestion de l'ordre partiel est présentée à deux niveaux : dans chaque connexion et entre les connexions. Deux mécanismes de gestion de la fiabilité sont exposés, connexion par connexion, et par groupe de connexion ; chacun induit une diminution du temps de transit des données, et permet à l'utilisateur de gérer "au plus tôt" la synchronisation de ses données. La présentation de cette architecture est suivie de la définition complète du service de Transport multimédia MM-POS ; l'ordre et la fiabilité y apparaissent comme deux paramètres de qualité de service à part entière.

Le ***chapitre 5*** de ce mémoire détaille tout d'abord les principaux concepts du protocole de Transport haute vitesse XTP. Nous étudions dans un second temps comment utiliser les procédures XTP pour mettre en œuvre un protocole de Transport multimédia d'ordre partiel, offrant les fonctionnalités précédemment décrites. Il ressort de cette étude la possibilité de concevoir une telle implantation ; nous soulignons cependant les inadéquations conceptuelles de XTP, et proposons une perspective d'évolution du protocole permettant d'envisager un support plus efficace d'une connexion multimédia d'ordre partiel.

CHAPITRE 1 - PROBLÉMATIQUE GÉNÉRALE

Introduction

Ces dernières années, les développements technologiques ont accéléré le rapprochement des Télécommunications et de l'Informatique ; les Télécommunications proposent des services de transmission de tous les types d'informations, et l'Informatique est à présent capable à un niveau local, d'intégrer données classiques, voix et images animées. Les années futures conduiront à la conception d'architectures de communication distribuées permettant de traiter ces différents média à des vitesses telles que les informations, indépendamment de leur nature et de leur provenance, seront aussi disponibles que si elles étaient présentes localement.
Un grand nombre de problèmes sont à résoudre et les choix de conceptions envisageables sont multiples. Dans ce chapitre, nous dégageons la problématique générale à laquelle se réfère notre contribution.

Ce chapitre est structuré en trois parties.

Dans la première section sont introduites les caractéristiques principales des applications multimédia ; leurs exigences et leurs besoins sont mis en évidence aux travers d'exemples illustratifs et la problématique générale est exposée.
La complexité d'un système de communication multimédia impose de disposer d'approches formelles, tant pour la conception des mécanismes que pour la représentation des données. Nous présentons ces différentes approches dans la deuxième section de ce chapitre, en portant l'accent sur les modèles de description d'objet multimédia, aptes à représenter les contraintes de synchronisation multimédia.
Nous abordons dans la dernière section les technologies de communication haute vitesse, et soulignons dans ce contexte les évolutions envisagées des protocoles de Transport.

I. Applications multimédia : caractéristiques et besoins

I.1. Média

D'une façon générale, un média définit le moyen par lequel l'information est perçue, exprimée et transmise. Comme en atteste la classification extraite du document MHEG

(MHEG - Multimedia and Hypermedia information coding Expert Group) [ISO90], le terme de média peut revêtir plusieurs sens :

- les média de perception indiquent la nature de l'information perçue par l'utilisateur, signal sonore ou image animée par exemple ;
- les média de présentation correspondent au dispositif physique utilisé pour acquérir l'information, ou pour la restituer (clavier, microphone, caméra pour l'acquisition ; écran, imprimante, haut-parleur pour la restitution) ;
- les média de représentation définissent la forme codée de l'information ;
- les média de communication représentent le dispositif physique utilisé pour la transmission et/ou le stockage des données (paire torsadée, câble coaxial, fibre optique pour la transmission ; disquette, disque dur, bande magnétique pour le stockage).

Dans la suite de ce mémoire, nous attribuerons au terme de média la signification donnée pour le média de représentation.

Parmi les différents média dont nous envisagerons le traitement et le transfert, nous distinguerons les cinq média suivants : les données binaires traditionnelles (texte par exemple), les graphiques, les images fixes, le son et la vidéo. Ces média se répartissent en deux catégories [Daun92] : les média discrets, indépendants du temps (données binaires, graphiques et images fixes), et les média continus présentant des caractéristiques temporelles marquées (son et vidéo).

I.1.1. Média discrets

Les ***données binaires*** représentent soit des fichiers texte, soit des commandes. D'une façon générale, ces données ne tolèrent aucune erreur de transmission (ni perte, ni altération), et ne présentent que peu de contrainte de délai de transmission et de variation de ces délais ou gigue. Cependant, le transfert de fichiers texte requiert une large bande passante, et s'effectue généralement par rafale.
Les ***graphiques*** proviennent le plus souvent de logiciels de dessin. Leur transfert est sporadique et ne demande qu'une faible bande passante. Les données graphiques sont généralement sensibles aux erreurs de transmissions, mais tolèrent délai et gigue.
Comme les média précédents, les ***images fixes*** présentent peu de contraintes en terme de délai, mais peuvent admettre une transmission imparfaite : une erreur sur quelques bits n'affectera au plus que deux ou trois pixels, et ne modifiera pas pour autant le sens global de l'image. Néanmoins, les images couleurs requièrent une grande quantité d'informations pour leur codage (de quelques mégabits - Mbits - à plusieurs dizaines de Mbits pour des applications d'imagerie spatiale ou médicale) ; celle-ci peut être réduite actuellement dans un rapport de 20 (valeur moyenne pour conserver une qualité acceptable), à l'aide de

techniques de compression notamment basées sur la norme JPEG (Joint Photographic Experts Group) définie par l'ISO et le CCITT [Lége91].

I.1.2. Média continus

Contrairement aux média discrets, le son et la vidéo présentent de fortes contraintes temporelles, liées au caractère isochrone de leur présentation.

Pour une transmission de la *voix* téléphonique, le codage MIC (Modulation par Impulsion Codée) nécessite une bande passante de 64 Kbits/s. Selon la qualité souhaitée, le transfert du son peut requérir jusqu'à 1,2 Mbits/s de bande passante (CD audio par exemple). La voix ne peut tolérer que des erreurs de transmission minimes, et nécessite une transmission isochrone.
La *vidéo* tolère davantage les erreurs que les variations de délais ou les retards. De plus, la vidéo est le média le plus exigeant en bande passante et peut réclamer (lorsqu'elle est non compressée) des débits allant de 150 à 400 Mbits/s. Plusieurs techniques de compression existent à l'heure actuelle afin de réduire les volumes des données à transférer, M-JPEG, MPEG I et MPEG II notamment [ISO93a] [Lu93].

I.1.3. Document multimédia - Document hypermédia

Un *document multimédia* désigne tout document regroupant un ou plusieurs média de nature semblable ou différente.

Il est cependant possible de distinguer deux types de documents multimédia, les documents de structure simple, consultable de façon séquentielle, et les documents plus structurés, ou *documents hypermédia*.

Un *document hypermédia* est un ensemble de documents multimédia reliés entre eux par des liens logiques, et activable par l'utilisateur. Ces documents peuvent être stockés sur une base de données réparties, l'activation d'un lien pouvant provoquer un changement de site.

A l'heure actuelle, deux standards font référence quant aux travaux de normalisation des documents multimédia : Hytime et MHEG.

- *Hytime* (Hypermedia/Time-Based structuring language) est un standard international de l'ISO [ISO92], basé sur l'utilisation du langage SGML [Newc91] [Koeg93].
- *MHEG* (MHEG - Multimedia and Hypermedia information coding Expert Group) est le prochain standard de l'ISO/IEC [ISO93b] pour la représentation des informations multimédia et hypermédia. Ce standard fournit une méthode de composition des documents multimédia (par synchronisation ou par l'utilisation d'hyperliens), et en

autorise une représentation codée. Ces documents peuvent alors être échangés entre les différents sites impliqués dans une application multimédia distribuée.

En conclusion, il apparaît qu'un document multimédia présente, de par sa diversité, plusieurs caractéristiques, à la fois de transfert et de présentation :

- un degré de sensibilité aux pertes et aux erreurs différents selon chaque média ;
- une quantité d'informations potentiellement élevée si le document comporte des images fixes, et une bande passante élevée pour le transfert de la vidéo ;
- des gigues de présentation bornées pour les média continus ;
- des délais de transmission éventuellement contraints ;
- des contraintes de synchronisation intra-média et inter-média.

I.2. Applications multimédia

Une application est dite multimédia lorsqu'elle implique le traitement informatique, et éventuellement le transfert de documents multimédia au travers d'un réseau de communication. Les applications multimédia *locales* ne nécessitent aucun transfert de données, et se limitent à une manipulation informatique des données disponibles localement. A l'opposé, les applications multimédia *réparties* ou *distribuées* requièrent l'échange de données entre deux ou plusieurs sites distants.

I.2.1. Applications multimédia locales

L'illustration la plus marquante de ce type d'applications est l'enseignement assisté par ordinateur (EAO). Les systèmes les plus simples sont composés de cours textuels, intégrés aux ordinateurs. Tirant partie des évolutions technologiques, les systèmes d'EAO intègrent aujourd'hui plusieurs média, améliorant ainsi la convivialité de l'application. SteelDEM [Ami93], TRAIN [Auge93], CALSA [Barr93] font partie des multiples systèmes d'EAO hypermédia développés ces deux dernières années.

I.2.2. Applications multimédia distribuées

Le CCITT [CCIT88] distingue les applications multimédia distribuées (appelées aussi *services*) en deux catégories :

- les services interactifs, incluant services conversationnels (audioconférence et visioconférence), services de messagerie (courrier électronique multimédia) et de consultation (base de donnée multimédia) ;
- les services de distribution, sans contrôle de présentation de la part de l'utilisateur (télévision par câble), ou avec contrôle (jeux électroniques coopératifs).

Dans ce contexte, il est possible de distinguer deux types d'applications, les applications *différées* et les applications *en temps réel.*

Les applications multimédia ***différées*** impliquent le rapatriement et le stockage de documents multimédia en provenance de site(s) distant(s) ; la présentation de ces documents est effectuée dans un second temps. Dans ce contexte, la principale contrainte que peut subir l'utilisateur est la durée séparant sa requête de la présentation finale ; elle est liée au temps de transfert du document. Une fois que le document multimédia est disponible localement et dans son intégralité, l'application est assimilable à une application locale ; en cela, ces applications présentent un caractère asynchrone. Actuellement, l'un des systèmes applicatifs les plus en vogue est le *World Wide Web* (W3), permettant à ses utilisateurs de consulter des documents multimédia par l'intermédiaire de logiciels tels que *Mosaic* ou *Netscape*. Parmi les documents d'apprentissage disponibles sur le réseau Internet, on peut citer [Andr95] [Mars95] [Wolf95].

Les applications multimédia ***en temps réel*** impliquent également le transfert de documents multimédia, mais la restitution de ces documents se fait en temps réel, c'est-à-dire au fur et à mesure de leur acquisition en provenance du réseau. En cela, ces applications présentent un caractère synchrone, destiné à améliorer la connectivité entre leurs utilisateurs. Nous présenterons en (I.2.3) les applications le plus significatives dans ce domaine, afin d'en souligner les exigences en termes de communication et de traitement informatique.

Nous désignerons dorénavant sous le terme d'application multimédia toute application multimédia distribuée. Comme nous allons le voir ultérieurement, les caractéristiques de ces applications diffèrent selon le nombre de sites impliqués, la nature des média échangés, l'aspect coopératif ou non de l'application, etc.

I.2.3. Exigences des applications multimédia distribuées

Il existe à l'heure actuelle de nombreux domaines d'applications qui vont pouvoir profiter de l'émergence du multimédia pour étendre et enrichir les systèmes actuels. Parallèlement, la technologie multimédia fait naître de nouveaux domaines d'applications jusque là inexploités. Parmi les domaines d'ores et déjà envisageables, on peut citer la bureautique, le travail coopératif scientifique, les applications résidentielles ou le commerce. Plusieurs bibliographies répertoriant toutes les applications ont été publiées ; elles traitent notamment de l'aspect coopération et de l'intelligence artificielle distribuée [Chai92] [Bond92]. [Will94] [Tawb92] [Naff90] fournissent également une classification de plusieurs applications multimédia.

Nous présentons dans le paragraphe suivant les applications multimédia les plus illustratives afin d'en dégager les caractéristiques majeures.

Base de données multimédia

Les bases de données multimédia permettent d'accéder à des quantités d'informations beaucoup plus importantes que les bases de données traditionnelles. Elles possèdent les fonctionnalités de base (classement, recherche par critère, etc.), mais tiennent compte du fait que les données sont à présent multimédia. L'intérêt d'une base de donnée multimédia apparaît surtout dans le domaine médical [Karm90] ; la consultation des guides touristiques et des pages jaunes est également rendue plus conviviale. Les besoins de ce type d'applications diffèrent selon le degré d'interactivité entre l'utilisateur et la base. Généralement, le temps de réponse de la base n'étant pas le critère le plus significatif de qualité, ce type d'applications ne présente pas de fortes contraintes de temps, et se rapproche en cela des applications multimédia à caractère asynchrone.

Audioconférence/Visioconférence

Les systèmes d'audioconférence et de visioconférence permettent à leurs utilisateurs de correspondre par l'intermédiaire de leurs stations de travail. Chaque participant peut entendre (audioconférence) et visualiser (visioconférence) les autres membres de l'application. Les média impliqués (le son et la vidéo) confèrent à ces applications un caractère synchrone marqué, et peuvent requérir une forte bande passante. De nombreux systèmes de visioconférence ont été développés ces dernières années, parmi lesquels IVS [Turl93], TSVS [Owe94] et PNSVS [Owe95a].

Édition multi-utilisateur

Dans une application d'édition multi-utilisateurs, un groupe d'utilisateurs est autorisé à manipuler et à modifier un même document. Deux aspects majeurs caractérisent ces applications : le partage et le contrôle d'accès aux informations (XTV [Chun94]), et la possibilité donnée aux participants de dialoguer sur une fenêtre annexe, par l'intermédiaire d'outils tels que IVS ou TSVS. Les systèmes les plus récents tels que Communique! [Para95] présentent des solutions intégrées, alliant visioconférence, tableau blanc partagé et partage d'application.

Enseignement à distance

Ces dernières années, le domaine de l'enseignement assisté par ordinateur (EAO) a connu un essor important, lié aux capacités des ordinateurs. Dans une application de télé-enseignement, un professeur peut délivrer son cours de façon simultanée et sans se

déplacer, à plusieurs élèves répartis sur des sites distants. Les communications entre élèves et professeur peuvent être point à point ou point à multipoints. Là encore, l'utilisation de systèmes d'audioconférence ou de visioconférence est le moyen utilisé pour autoriser une interactivité acceptable (CO-LEARN [Der93] [Croi94]). Au Laboratoire d'Analyse et d'Architecture des Systèmes de Toulouse, les travaux menés autour de l'application VACBI [Owe95b] sont destinés à émuler des relations réelles entre élèves et enseignant. Ces travaux intègrent notamment systèmes de visioconférence et tableaux de dialogue, ces derniers servant à la fois de tableau blanc et d'outil de partage d'application. Des communications multipoints permettant une coopération efficace entre les participants sont en cours de réalisation.

I.3. Conclusion

Il apparaît ainsi que les exigences des applications multimédia ont principalement trait :

- à la nature des média manipulés, qui requièrent potentiellement de la part du réseau une forte bande passante, mais également des gigues de transferts minimes ;
- au degré d'interactivité (ou temps de réponse) entre les membres de l'application imposant que les supports de communication utilisés présentent des garanties en terme de délai ;
- au caractère potentiellement coopératif de l'application.

Ce dernier point fait apparaître un nouveau besoin que nous précisons ici ; les applications multimédia coopératives requièrent l'établissement de communication non plus seulement de point à point, mais de point à multipoints et d'une façon plus générale de multipoints à multipoints. Cet aspect souligne la nécessité de disposer de modèles de coopération, définissant formellement les interactions entre les membres de l'application ; en outre, des protocoles de gestion de la coopération doivent être définis de façon à résoudre les conflits d'accès à des données partagées. Les travaux relatifs à ce propos constituent à l'heure actuelle matière à réflexion et à recherche. Un état de l'art complet du travail coopératif est publié dans [Vill95] ; initialement fondé sur [Diaz92] [Diaz93], une définition formelle du concept de coopération dans les systèmes distribués y est proposée, ainsi qu'un service et un protocole de gestion de groupe.

Quelle que soit la qualité de service fournie par les supports de communication, on doit admettre que le traitement informatique d'un document multimédia fait généralement appel à un système d'exploitation qui en annihile les bénéfices [Owe95c]. Ce dernier point souligne la nécessité que soient définis des mécanismes de synchronisation des flux multimédia (ou synchronisation multimédia) lors de leur restitution finale. La synchronisation multimédia consiste en la gestion des décalages temporels intra-flux et inter-flux [Rust94] [Dair94].

Enfin, il apparaît nécessaire de définir des protocoles de communication à haut débits, compatibles avec les exigences temporelles des applications multimédia.

La problématique générale consiste ainsi à définir des architectures de communication, aptes à prendre en compte les exigences des applications multimédia. Il se dégage alors naturellement un besoin en représentation de ces contraintes.

II. Les approches formelles

Ces dernières années ont vu la proposition de méthodologies de conception des systèmes distribués ainsi que le développement d'outils logiciels associés. Au regard des conclusions précédemment formulées, la complexité des futurs systèmes à concevoir ne fait que renforcer le bien fondé de ces approches.

II.1. Limites des approches traditionnelles

Les premières études sur la conception formelle ont davantage porté sur la description des mécanismes que sur les données elles-mêmes.

Les travaux sur la spécification et la validation des protocoles ont débuté dès la réalisation des réseaux d'ordinateurs complexes, et plusieurs approches ont été développées sur les bases de systèmes de transition. Dans un second temps, la volonté supplémentaire de produire des implantations de façon automatique ou semi-automatique, à partir de spécifications formelles, a conduit à aborder la conception formelle par le biais d'approches langage.

Les systèmes de transition

Un système de transition permet de décrire un protocole par un système de changement d'états. On y distingue essentiellement :

- les *machines à états*, dans lesquelles le protocole est décrit sous la forme d'états et de transitions décrivant le comportement local des entités [Boch80] [Dant80] ;
- les *réseaux de Petri*, permettant de représenter la notion de rendez vous, et d'exprimer les comportements non déterministes dans un système parallèle [Diaz82] [Cour84] [Ayac85] [Juan90] ;
- les *algèbre de processus*, décrivant le comportement distribué par des équations de comportement [Miln80]

Les approches langage

Trois techniques majeures de description formelle ont été développées, Estelle [ISO87a] [Budk87] [Diaz89], LOTOS [ISO87b] [Brin87] [VanE89] et SDL [SDL88]. Estelle et

LOTOS sont le résultat de recherches menées au sein de l'ISO (International Standardization Organization) par le groupe ISO/SC21/FDT (Formal Description Techniques), et sont aujourd'hui des standards internationaux ; initié dés 1984, le projet ESPRIT SEDOS (Software Environment for the Design of Open Distributed Systems) a également contribué au développement des techniques Estelle et LOTOS, à la définition de leur sémantique formelle et à leur évaluation sur des applications pilotes [Diaz85]. SDL a été développée en parallèle par le CCITT (Comité Consultatif International Télégraphique et Téléphonique). Chacune de ces techniques permet de décrire les mécanismes de protocole dans un langage de programmation :

- Estelle repose sur un modèle de machines à états étendues communiquant par files FIFO et fait appel au langage Pascal dans la représentation de ses données. Dans le chapitre 3, nous présenterons plus amplement Estelle, langage que nous avons retenu comme outil de spécification ;
- LOTOS repose sur le formalisme CCS et représente les données sous la forme de types abstraits algébriques ACT ONE ;
- SDL repose sur un modèle de machines à états étendues communicant par file FIFO, et représente les données par des types abstraits algébriques. Une forme graphique du langage est aujourd'hui normalisée.

Le principal reproche que l'on puisse faire à ces techniques est qu'elles ne permettent pas de satisfaire tous les critères liés au processus de conception d'un système de communication multimédia ; plus précisément, elles ne permettent :

- ni de représenter les contraintes d'un document multimédia, notamment la synchronisation multimédia,
- ni de spécifier, et à fortiori de valider, les mécanismes de gestion de ces contraintes.

II.2. Évolutions envisagées

Face à ces limites, les dernières approches envisagées dans la conception de systèmes multimédia se regroupent en deux catégories :

- la première, orientée langage, propose des extensions temporelles aux techniques de description formelles existantes ; en cela, ces exemples autorisent la description et la validation de procédures de traitement de la synchronisation multimédia. LOTOS notamment a donné lieu aux langages ET_LOTOS (Enhanced Timed LOTOS) [Ledu93] [Ledu94] et RT_LOTOS (Real Time LOTOS) [Cour94] ;

- les approches de la deuxième catégorie visent à définir des modèles de représentation d'objet multimédia, intégrant l'ensemble des contraintes de synchronisation (ordonnancement des flux multimédia, expression des dérives admissibles intra-flux et

inter-flux) [Litt90a] [Bert91]. Ces modèles autorisent la spécification non ambiguë de scénarios de présentation multimédia, et permettent de plus de les simuler et de les valider. De nombreuses études ont déjà été réalisées dans ce domaine, et des modèles ont été proposés.

Une partie des recherches actuellement menées dans le projet CESAME contribue au développement de ces études (le langage RT_LOTOS et le modèle TSPN notamment), et à leur intégration à différents niveaux du système de communication multimédia développé.

Nous utiliserons dans la suite de cette thèse le modèle TSPN, et nous montrerons comment il a conduit à la définition de nouveaux protocoles. En conséquence, le paragraphe suivant présente les différents modèles ayant conduit à concevoir le modèle TSPN.

II.3. Le modèle TSPN

Les réseaux de Petri sont largement utilisés pour représenter les systèmes discrets à évolution simultanée ; en cela, ils sont a priori adaptés pour modéliser les flux d'informations parallèles qui prennent part à la présentation d'un document multimédia. En outre, le caractère graphique des réseaux de Petri permet facilement de mettre en œuvre des paradigmes multimédia tels que le paradigme de régie numérique.

La réalisation d'une application multimédia synchronisée impose de prendre en compte les problèmes liés à la variabilité des temps de traitements des systèmes asynchrones, afin de les corréler aux caractéristiques de présentation des documents multimédia. Ces dernières années ont vu le jour de modèles de représentation de la synchronisation reposant sur le formalisme des réseaux de Pétri. Dans tous ces modèles, une place est associée à la présentation d'une donnée (image, séquence vidéo, échantillon de son, ...), la structure du réseau et les transitions exprimant les relations de dépendance intra-flux et inter-flux. Deux objectifs fondamentaux sont poursuivis :

- la validation des contraintes temporelles, c'est-à-dire l'absence d'inconsistance dans la description de la synchronisation spécifiée par l'utilisateur ;
- la vérification des algorithmes de synchronisation vis-à-vis de la spécification de ces contraintes.

Les modèles présentés ci-après ont conduit au développement du modèle TSPN. Afin d'en mesurer l'utilité, ils sont confrontés à la problématique de la synchronisation multimédia, exprimée en terme de *pouvoir d'expression et de modélisation*. Le *pouvoir de modélisation* d'un modèle est sa capacité à représenter facilement un scénario de

présentation multimédia. Son *pouvoir d'expression* est sa capacité à spécifier un scénario de façon complète [Séna94].

Le modèle OCPN

Le modèle OCPN (Object Composition Petri Nets) [Litt90a] étend le formalisme des réseaux de Petri en associant une temporisation aux places du réseau. Le modèle repose sur les sept schémas de synchronisation de base qui caractérisent toutes les positions relatives entre deux intervalles temporels [Alle83]. Ces schémas définissent l'ensemble des règles de composition utilisables pour construire des modèles complexes à partir de modèles plus simples.
Cependant, les temporisations des places représentent des durées nominales, et ne permettent de modéliser ni les variations de temps induites par les systèmes asynchrones, ni la gigue admissible intrinsèque aux différents média d'un document multimédia. Le modèle OCPN présente donc une limite en terme de pouvoir d'expression. Un exemple d'OCPN est illustré figure (1.1). Dans cet exemple, si la place a_1 a été marquée à la date t, la transition t_1 doit être tirée à la date t+1.

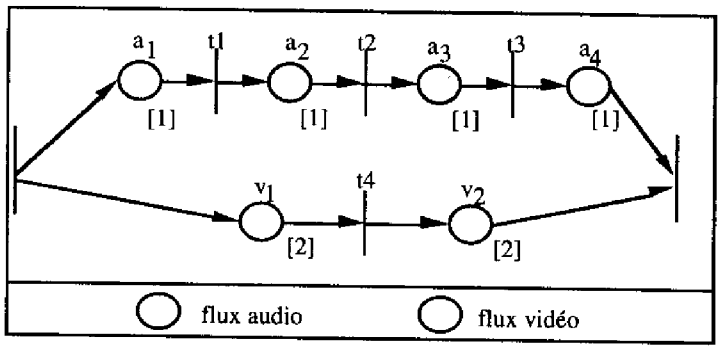


Figure (1.1) : Exemple d'OCPN

Le modèle TPN

Le modèle TPN (Time Petri Nets) de Merlin [Bert91] rend possible l'utilisation de temporisations imparfaites en attribuant à chaque transition un intervalle de temps dans lequel doit se situer le tir de la transition. Les règles de tir d'un TPN stipulent que si une transition T, ayant pour intervalle de tir [a,b], est sensibilisée à la date t, alors T devra être tirée entre les dates (t+a) et (t+b). Ce modèle possède ainsi un très bon pouvoir d'expression, équivalent à celui des machines de Turing. Cependant, il induit une synchronisation forte entre les flux, toujours conduite par le flux le plus en retard ; cette considération peut entraîner le non respect des contraintes temporelles des flux les plus en avance. Le modèle TPN ne permet donc pas une modélisation aisée d'un scénario de synchronisation inter-flux, et présente une limite en terme de pouvoir de modélisation.

Le modèle ATPN

Le modèle ATPN (Arc Time Petri Nets) [Walt83] comble les lacunes des TPN en plaçant des intervalles temporels non plus sur les transitions, mais sur les arcs sortants des places. La spécification de la synchronisation inter-flux peut s'exprimer de façon simple par une fusion de transitions : si les intervalles dynamiques associés aux arcs précédant une transition inter-flux ont une intersection vide, c'est qu'une désynchronisation inadmissible des flux considérés est intervenue ; la présentation est alors interrompue. Toutefois, les règles de tir des transitions ne permettent pas de prendre en compte ce type de problème autrement que par une rupture de connexion. C'est donc en terme de pouvoir d'expression que se situe la limite du modèle ATPN.

Le modèle TSPN

Le modèle TSPN [Diaz93a][Diaz93b][Diaz94c][Séna94] offre le pouvoir d'expression des TPN, et étend le modèle ATPN par l'ajout de règles de tir des transitions inter-flux, lorsque l'intersection des intervalles de validité des arcs précédant ces transitions est vide. En cela, les TSPN prennent en compte les phénomènes d'asynchronisme des systèmes distribués en tirant partie du degré de variation acceptable des temps de présentation d'un objet multimédia.

Aux arcs de sortie du TSPN sont attachés des triplets (x, n, y), appelés intervalles de validité temporelle, où x, n et y désignent respectivement le temps minimal, nominal et maximal du traitement de présentation d'un objet donné.
Neuf sémantiques de tir d'une transition inter-flux ont été définies de façon à contrôler précisément les dérives temporelles entre les différents flux manipulés. Elles autorisent par exemple la spécification de mécanismes de synchronisation conduits par le flux le plus en avance (synchronisation de type "ou fort"), par le flux le plus en retard (synchronisation de type "et faible"), ou par un flux donné (synchronisation de type "maître"). En cela, ces règles définissent les 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.

Le TSPN de la figure (1.4) indique que la synchronisation effectuée au niveau de la transition T est de type "maître audio". La sémantique associée à ce type de synchronisation autorise la poursuite de l'application, en dépit d'une violation des contraintes temporelles du flux vidéo.

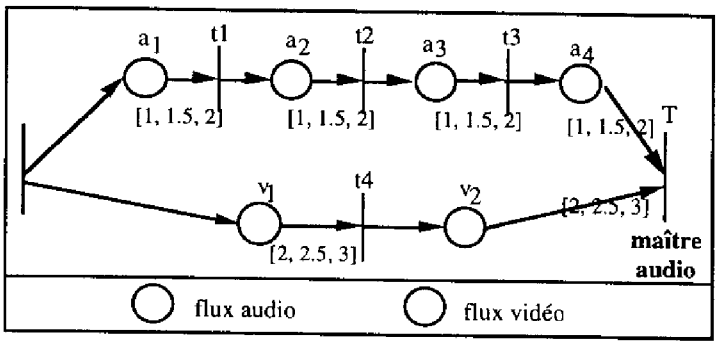


Figure (1.4) : Exemple de TSPN

En conclusion, le modèle TSPN introduit une taxonomie de la synchronisation multimédia qui autorise la détection des fautes de synchronisation, tout en garantissant la continuité des flux de données par le respect des règles de tir des transitions inter-flux. En cela, le modèle TSPN est le modèle le plus complet dans l'expression de la synchronisation temporelle d'une application multimédia. Ce modèle a été étendu aux objets hypermédia [Séna95].

Dans ce nouveau contexte général, il apparaît que les protocoles de communication, en particulier de niveau Transport, initialement conçus pour assurer des transferts de données sans aucune garantie de performance temporelle, sont à présent inadaptés.

III. Évolution des protocoles de communication

L'objet de cette section n'est pas de présenter un état de l'art des services et protocoles de Transport (le chapitre 2 étant entièrement dévolu à cette présentation), mais de souligner leur tendance d'évolution. Nous présentons tout d'abord les caractéristiques majeures des réseaux à haut débit actuellement disponibles ou en cours d'étude.

III.1. Environnements de communication haute vitesse

A l'heure actuelle, il existe essentiellement trois technologies capables de supporter les exigences des systèmes multimédia répartis : FDDI, DQDB et ATM. FDDI fait référence aux réseaux locaux, DQDB aux réseaux métropolitains et ATM aux réseaux internationaux (ou longue distance).

FDDI - Fiber Distributed Data Interface

Le réseau FDDI [Ross89] est un réseau local utilisant un double anneau contrarotatif à fibre optique. Il offre un débit de 100 Mbits/s et la possibilité d'interconnecter un maximum de 1000 stations.

DQDB - Distributed Queue Dual Bus

Le réseau DQDB a été adopté par l'IEEE comme standard des réseaux métropolitains [IEEE90]. DQDB offre un débit de 150 Mbits/s au moyen d'un réseau composé de deux bus (A et B) unidirectionnels, de noeuds connectés à chacun des bus, et de deux générateurs de trames à l'extrémité de ces bus. La taille des trames générées est de 53 octets dont 48 réservés aux données utilisateur. Deux mécanismes de contrôle d'accès aux bus ont été définis, respectivement dédiés aux transferts de données isochrone (vidéo et voix) et asynchrone (données).

ATM - Asynchronous Transfert Mode

La variabilité des besoins en débits que requiert une application multimédia a motivé la conception de réseaux haute vitesse basés sur des techniques asynchrones.

La technique ATM [Syka91] [Anag91] [Pryc91] [Bonj94], actuellement en cours de développement, a pour but de répartir dynamiquement les ressources du réseau. Cependant, pour certains types d'applications, la difficulté consiste à assurer le respect de paramètres de QoS tels que le débit, le délai ou la gigue de transfert, comme si le réseau était synchrone de bout en bout.

ATM repose sur un mode de transfert orienté paquet et fonctionne en mode connecté. Les unités d'information manipulées sont des cellules de tailles fixes (53 octets) contenant un segment d'information et un en-tête permettant le routage des cellules de bout en bout. Le protocole n'assure ni la reprise des pertes, ni le contrôle de flux ; cependant, une allocation des ressources faite durant la phase d'établissement de connexion permet de garantir un faible taux de perte ; en outre, un contrôle du débit d'émission est effectué à la source. Du fait de la (petite) taille des cellules, le modèle de référence définit une couche d'adaptation, l'AAL (AAL -ATM Adaptation Layer) ayant pour fonction de base la segmentation et le réassemblage des données utilisateur. L'AAL assure également une fonction de détection d'erreur. Une classification des services fournis par l'AAL a été établie par le CCITT ; elle repose sur trois critères : l'existence ou non de contraintes de temps réel, la variabilité ou la constance du débit d'information et le mode de service (connecté ou sans connexion).

III.2. Évolution des protocoles de Transport

Située entre la couche applicative et le système de communication proprement dit, le rôle des services et protocoles de Transport est de réaliser une jonction adéquate entre les besoins applicatifs et les caractéristiques de l'environnement réseau sous-jacent.

Les premières motivations de conception des protocoles de Transport sont étroitement liées aux caractéristiques des réseaux informatiques de la fin des années 70. Ces réseaux, en majorité fondés sur la technique de commutation de paquets, offraient une faible bande passante et des taux d'erreur importants ; en conséquence, les protocoles de Transport ont été conçus pour garantir un transfert fiable des données, tout en essayant d'optimiser l'utilisation de la bande passante. L'émergence des réseaux locaux tels que Ethernet ou Token Ring a modifié le contexte précédent. Les caractéristiques de ces réseaux, faible taux d'erreur et non déséquencement des données, ont motivé la conception de protocoles dépourvus de mécanismes de reprise des pertes, et de reséquencement des données en réception.

La fibre optique autorise à présent la conception de réseaux offrant une large bande passante et des taux d'erreur extrêmement bas, non seulement à courte distance (FDDI), mais également sur des échelles métropolitaines (DQDB) et bientôt internationales (ATM). En outre, les applications multimédia dorénavant envisageables présentent de fortes contraintes temporelles. Ces dernières évolutions soulignent l'inadéquation conceptuelle des protocoles de Transport.

Dans ce nouveau contexte, deux approches sont actuellement envisagées quant aux modifications souhaitables des protocoles de Transport :

1 - La première consiste à implanter les protocoles existants (TCP notamment) de façon suffisamment performante, par optimisation des codes, parallélisation des traitements ou implémentation matérielle. Cette approche est discutable sur deux points :

- le premier est d'ordre conceptuel ; par définition, une application distribuée requiert la mise en œuvre de protocoles de communication : ne pas adapter les fonctionnalités des protocoles aux caractéristiques applicatives peut conduire à des incohérences ou une sur-utilisation des ressources ;
- en outre, la mise en œuvre d'un protocole tel que TCP, *de façon suffisamment performante*, suppose des conditions d'implantation particulières à la fois matérielle et logicielle. Le développement d'applications dans un contexte plus général rend cette dernière considération caduque.

2 - La deuxième approche repose sur la conception d'un nouveau type de protocole, permettant une prise en compte des caractéristiques applicatives. L'utilisateur dispose d'un ensemble de procédures paramétrables, lui permettant d'implanter le protocole mettant en œuvre le service le plus adapté à ces besoins. XTP [PEI92] est le représentant le plus prometteur de cette nouvelle génération de "protocoles" de Transport.

Nous situerons et présenterons dans les prochains chapitres notre contribution dans le cadre de cette deuxième approche.

Conclusion

Ce chapitre a introduit la problématique générale dans laquelle s'inscrit notre contribution. Celle-ci résulte de la nature des documents multimédia dorénavant manipulables dans le cadre d'applications distribuées. Les contraintes de synchronisation de ces documents lors de leur présentation finale constituent l'un des aspects les plus critiques de cette problématique.

Face à la complexité que représente la conception d'un système réparti, nous avons rappelé l'importance de disposer de méthodes de description formelle des comportements distribués. Dans le contexte du multimédia, les limites de ces méthodes ont été soulignées, et les solutions envisagées ont été présentées. Une partie d'entre elles vise à définir des modèles de représentation des documents multimédia, en vue d'en exprimer formellement les contraintes de synchronisation intra-flux et inter-flux ; le modèle TSPN est à l'heure actuelle le plus abouti dans ce domaine.

Enfin, nous avons présenté les technologies nouvelles pour les supports de communication haute vitesse, et souligné les deux approches envisagées de conception des protocoles de Transport, inadaptés à ce nouveau contexte.

Afin de détailler le contexte de notre contribution, nous présentons dans le chapitre suivant un état de l'art des protocoles de Transport actuels. Ceci conduit à caractériser les inadéquations conceptuelles des protocoles traditionnels, et à discuter les évolutions envisagées. Nous présentons ensuite une analyse des services de Transport existant ou en cours de développement, sous l'angle de leur adaptation aux applications multimédia.

CHAPITRE 2 - ÉTAT DE L'ART DU NIVEAU TRANSPORT

Introduction

Ce chapitre présente un état de l'art des services et protocoles de Transport actuels, et étudie leur aptitude à prendre en compte et satisfaire les exigences des applications multimédia dans un environnement de communication haute vitesse.
En fin de chapitre sont présentés les principaux projets nationaux et internationaux actuellement menés dans ce domaine.
Ce chapitre est articulé en trois sections majeures.
La première section présente les principaux mécanismes de Transport et leur mise en œuvre à l'intérieur des protocoles les plus représentatifs : TCP, OSI TP4, VMTP, NETBLT, Delta-t, XTP et URP/Datakit. Ceci permet d'analyser l'adéquation des mécanismes de Transport aux exigences des applications multimédia dans un environnement réseau à haut débit.
La deuxième section présente tout d'abord les caractéristiques générales des deux modes du service Transport OSI [IS8072] [IS8072/ADD1] ; les lacunes de ce service sont mises en évidence et les évolutions proposées sont exposées dans le cadre du service de Transport OSI 95.
Enfin, la dernière section de ce chapitre présente quelques uns des grands projets actuellement menés dans le monde, en particulier dans la communauté européenne.

I. Protocoles de Transport : fonctionnalités et mécanismes

Située entre la couche applicative et le système de communication proprement dit, la couche Transport apparaît comme une couche charnière dont la finalité est de réaliser une jonction adéquate entre les besoins applicatifs et les caractéristiques de l'environnement réseau sous-jacent.

Suite aux évolutions technologiques de ces dernières années, on distingue à l'heure actuelle deux générations de protocoles :

- la première comprend les protocoles tels que TCP et OSI/TP4, non conçus pour être performants en temps, mais pour fournir aux utilisateurs un support de communication fiable.

- la seconde génération tient compte des dernières évolutions technologiques. Ces protocoles, plus performants que les précédants, sont destinés à des domaines d'applications ciblées. Delta-t, VMTP, NETBLT, et XTP illustrent ce type de protocole.

Quels que soient leurs objectifs, les protocoles de Transport actuels sont tous basés sur l'un des modes opératoires suivants, le mode connecté et le mode non connecté.

- la fonctionnalité majeure des protocoles *orientés-connexion* est de garantir la délivrance des données utilisateur dans leur ordre d'émission ;
- à l'opposé, les protocoles *sans connexion* tels que UDP ne garantissent pas cette qualité de service.

1 - La mise en œuvre d'un protocole orienté-connexion se décompose généralement en trois phases :

- Les unités de protocole (TPDU - Transport Protocol Data Unit) échangées initialement, ont pour but d'établir une *connexion* entre entités communicantes. Cette phase permet aux entités émettrice et réceptrice de s'accorder sur un certain nombre de variables, nécessaires à la gestion de la connexion.
- Une fois connectées, deux entités peuvent échanger des TPDUs dans le cadre d'une connexion identifiée ; cet échange définit la phase de transfert de données, durant laquelle sont mis en œuvre les mécanismes de gestion de l'ordre et de la fiabilité.
- La(les) TPDU(s) échangées en dernier permet(tent) de libérer les contextes mémorisés ; cette phase définit la libération de la connexion.

Notons cependant qu'un protocole tel XTP peut être mis en œuvre en combinant à volonté les phases d'établissement de connexion et de transfert de données, les phases de transfert de données et de fermeture de connexion, ou les phases d'ouverture de connexion, de transfert de données et de fermeture de connexion.

2 - Le mode opératoire des protocoles sans connexion ne comporte qu'une seule phase, la phase de transfert des données. Les TSDUs (TSDU - Transport Service Data Unit) soumises en émission sont traitées indépendamment les unes des autres ; aucune garantie de délivrance, ni de respect d'un quelconque ordre de délivrance n'est envisageable dans ce contexte. En cela, les protocoles sans connexion sont dits non fiables, la délivrance des données dépendant directement du degré de fiabilité du service de Réseau sous-jacent.

I.1. Présentation d'un sous ensemble des protocoles orientés-connexion

I.1.1. TCP (Transmission Control Protocol - 1977)

Développé sur l'initiative de la DARPA (Defense Advanced Research Projects Agency), TCP s'est très vite imposé comme un standard de fait dans la communauté universitaire, les réseaux locaux et le monde Unix en général [Come91].

TCP est un protocole orienté-connexion fournissant un service fiable (ni perte, ni duplicata, ni altération des données), en s'appuyant sur un service de Réseau non fiable (IP - Internet Protocol). TCP assure le transport de flux d'octets entre deux processus distants. Une connexion TCP est bidirectionnelle.
TCP est destiné à supporter des applications nécessitant un transfert de données parfaitement fiable, telles que le transfert de fichiers (FTP - File Transfert Protocol) ou le terminal virtuel (TELNET - TErminaL NETwork protocol).

I.1.2. OSI/TP4 (OSI Transport Protocol class 4 - 1982)

Le protocole OSI/TP4 défini par l'ISO est également conçu pour offrir un service de Transport fiable. Il diffère du protocole TCP, notamment par la prise en compte de paramètres de QoS négociables à l'interface des couches Session et Transport du modèle OSI, mais également entre utilisateurs distants.
A la différence de TCP, TP4 est standardisé par un organisme international ; sa spécification est définie dans la norme ISO 8073 [IS8073]. Une mise à jour du protocole a été proposée pour approbation dans le document DIS8073 [DIS92]. Les modifications soumises visent à adapter les fonctionnalités du protocole aux environnements à haut débit.

I.1.3. URP/Datakit (Universal Receiver Protocol - 1976)

Datakit [Ches79] est un réseau conçu par les laboratoires AT&T Bell, offrant un service de type circuit virtuel, avec garantie de délivrance sans déséquencement de données non altérées, mais avec possibilité de trous. Ces trous correspondent aux données erronées rejetées par la couche liaison de données. L'unité d'échange des informations de contrôle et des données est l'octet ; pour différencier données et informations de contrôle, un neuvième bit est rajouté à chaque octet.

Le couche Transport de Datakit est constitué du protocole de Transport URP (Universal Receiver Protocol). Son interface d'accès permet d'établir des connexions point à point, selon deux modes d'opérations, les modes *caractère* et *block* .
En mode caractère, le protocole URP n'assure pas la récupération des pertes.
En mode block, deux sous modes sont définis :

- le mode *message block*, imposant au protocole de fournir un service fiable en appliquant un contrôle de flux ;
- le mode *stream block* , imposant au protocole de transmettre les données avec contrôle de flux et contrôle d'erreur limités.

I.1.4. Delta-t (1978)

Le protocole de Transport Delta-t [LLB83] [Wats89] a été conçu par le Laboratoire National de Lawrence Livermore pour permettre la mise en œuvre du système d'exploitation distribué LINCS. Delta-t est conçu pour supporter des communications de type *stream* ou transactionnel, en utilisant les services d'une couche Réseau non fiable. Un stream est défini comme un flux de données monodirectionnel ; il est identifié par un port destination, un port source et un numéro de stream. Delta-t offre un service fiable de bout en bout.

I.1.5. NETBLT (NETwork BLock Transfer - 1986)

NETBLT a été développé par le MIT pour assurer le transport fiable de gros volumes de données [Clar87] ; il est conçu pour être mis en œuvre au dessus d'un service de Réseau non fiable. Une connexion NETBLT est monodirectionnelle. L'unité fondamentale de transmission est le *buffer*, correspondant à un tampon de données. Un contrôle de flux et un contrôle de débit sont utilisés pour le transfert des données.

I.1.6. VMTP (Versatile Message Transaction Protocol - 1987)

VMTP [Cher89] [Nor89] a été développé par l'Université américaine de Stanford, dans l'objectif de supporter le transport à haute vitesse de systèmes distribués incluant les accès fichiers et les appels procéduraux à distance (RPC - Remote Procedure Call).
Ce dernier type d'opérations requiert des temps de réponse courts, et implique l'échange de messages de petite taille. VMTP définit sous le terme de *message de transaction* une requête émise par un client à un groupe de serveurs. Un message de transaction peut donner lieu à un *message de réponse* en provenance d'un ou plusieurs serveurs.

Ce protocole peut également être mis en œuvre en mode non connecté, par simple indication dans une requête qu'aucun message réponse n'est attendue. VMTP offre également un service fiable en mode continu (l'unité d'échange est l'octet), destiné à supporter le transport de gros volumes de données.

I.1.7. XTP (Xpress Transfer Protocol - 1987)

XTP [Ches87] [Ches89] [PEI92] symbolise une nouvelle approche de conception des protocoles, les protocoles de transfert, regroupant fonctions de Transport et fonctions de routage de façon à améliorer le contrôle de congestion. XTP a été conçu dans le cadre du projet de Protocol Engines Incorporated, dont le but était la conception en circuits VLSI d'un protocole haute performance de niveau 3 et 4. La conception de XTP a été motivée par les besoins en communications d'applications à caractère distribué (RPC notamment),

temps réel ou multimédia. XTP est considéré comme un protocole de Transport allégé, conçu afin de minimiser le nombre d'instructions concernant la transmission et la réception de données dans un contexte exempt d'erreur. L'objectif ciblé est de diminuer les temps de traitement du protocole, au détriment de la bande passante. Une caractéristique majeure du protocole est d'offrir au concepteur du service un ensemble de mécanismes sélectionnables à volonté. L'utilisateur peut ainsi activer ou désactiver le contrôle de flux des données, le contrôle de détection d'erreur et/ou de reprise des pertes. Une présentation plus détaillée de la version 3.6 du protocole XTP sera fournie au chapitre 5 de ce mémoire.

Nous précisons maintenant les mécanismes principaux des protocoles de Transport orientés-connexion [Thai94a].

I.2. Signalisation

- dans la bande : VMTP, NETBLT, Delta-t, TCP, OSI TP4, XTP
- hors bande : URP/Datakit

La signalisation consiste en un échange d'informations de contrôle entre sous-systèmes de Transport communicants. Son objectif est de permettre l'ouverture et la fermeture des connexions, ainsi que le contrôle des connexions.

La signalisation peut être effectuée :

- *dans la bande* : les informations de contrôle sont échangées avec les données utilisateur, par le biais d'une même connexion .
- *hors bande* : une connexion supplémentaire est établie, uniquement dédiée au transfert des informations de contrôle ; dans ce cas, le sous-système récepteur n'a pas à distinguer informations de contrôle et de données.

I.3. Mécanismes d'ouverture de connexion

Une connexion de Transport peut être établie de façon implicite ou explicite. Dans les deux cas, le mécanisme mis en œuvre doit garantir l'impossibilité qu'une connexion non désirée soit établie ; ce critère sera par la suite désigné sous le terme de critère de sûreté.

I.3.1. Ouverture de connexion implicite : VMTP, Delta-t, XTP

Dans une ouverture de connexion implicite, le récepteur considère la connexion établie dès réception de la première TPDU notée CR-TPDU sur la figure (1.1). L'initiateur de la connexion est autorisé à émettre des données sans attendre l'acquittement de cette TPDU.

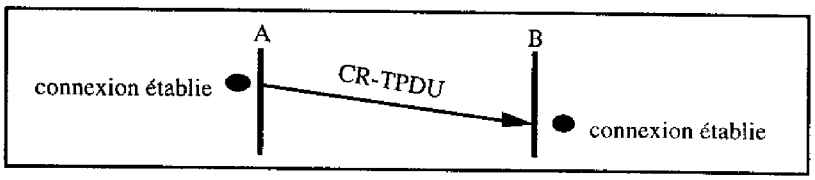


Figure (1.1) : Ouverture de connexion implicite

Quoi que très rapide, ce mécanisme nécessite d'une part que le réseau utilisé soit suffisamment fiable, et d'autre part que les TPDUs aient une durée de vie limitée dans le réseau, faute de quoi des TPDUs retardées pourraient engendrer la réouverture d'anciennes connexions, violant ainsi le critère de sûreté. A cet effet, une technique consiste à décrémenter le temps de vie d'un paquet lors de sa traversée dans chaque noeud du réseau. Une étude complète de ce mécanisme est fournie dans [Wats87]. Une ouverture de connexion implicite est particulièrement adaptée aux protocoles supports d'applications transactionnelles, ou de très courte durée.

I.3.2. Ouverture de connexion explicite

Une ouverture de connexion explicite nécessite l'échange d'au moins deux TPDUs entre entités communicantes. L'intérêt de ce mécanisme est de permettre la négociation de paramètres de QoS (taille des tampons mémoire, taille des rafales par exemple) durant la phase d'ouverture de connexion. Deux cas de figures sont à prendre en compte selon le degré de fiabilité offert par le service de Réseau sous-jacent.

I.3.2.1. Ouverture de connexion explicite à 2 messages : NETBLT, XTP, URP/Datakit

Lorsque la fiabilité du service de Réseau est *acceptable* vis-à-vis du critère de sûreté précédemment défini[2], l'échange de deux TPDUs, notées CR-TPDU et CC-TPDU dans la figure (1.2), est suffisant pour établir la connexion.

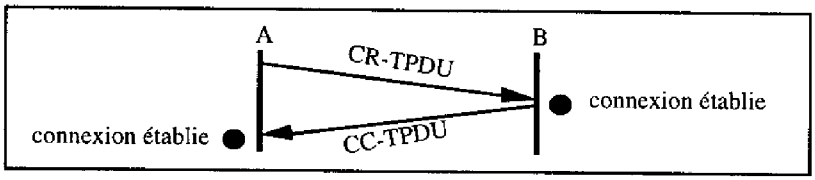


Figure (1.2) : Ouverture de connexion explicite à 2 messages

Notons que certains protocoles, XTP notamment, autorisent l'émission de données durant la phase d'établissement de la connexion, dans l'objectif d'augmenter le débit de la connexion.

---

[2] Le critère de sûreté est mis en défaut lorsqu'une CR-TPDU retardée ou dupliquée engendre la réouverture d'une ancienne connexion ; la fiabilité du service réseau est jugée acceptable (ou inacceptable) selon la probabilité qu'une telle occurrence survienne.

I.3.2.2. Ouverture de connexion explicite à 3 messages : TCP, OSI TP4

Lorsque les utilisateurs désirent établir une connexion bidirectionnelle, l'échange d'au moins trois TPDUs est nécessaire. La troisième TPDU, notée ACK-CC-TPDU dans la figure (1.3), permet à l'entité B d'être certaine que la connexion a été établie dans les deux sens.

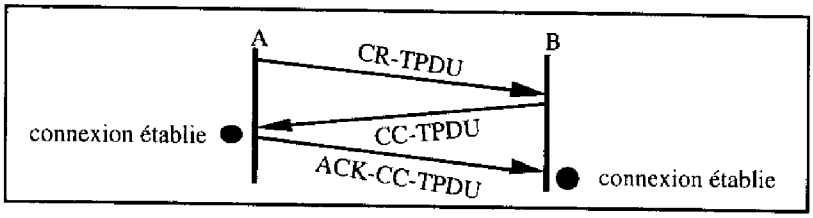


Figure (1.3) : Ouverture de connexion explicite à 3 messages

I.4. Mécanismes de fermeture de connexion

Selon le mécanisme mis en œuvre, une fermeture de connexion peut occasionner la perte des TPDUs en transit dans le réseau. Les différentes stratégies envisagées ainsi que leurs conséquences vis-à-vis de la fiabilité du service sont présentées dans ce paragraphe.

I.4.1. Fermeture de connexion implicite : VMTP, Delta-t

Ce mode de fermeture est basé sur la gestion d'un temporisateur associé à la connexion. Contextes émetteur et récepteur sont relâchés à l'expiration de ce temporisateur, dont la valeur doit être suffisamment grande pour garantir que les données échangées seront toutes délivrées. Ce mécanisme est généralement associé à une ouverture de connexion implicite.

I.4.2. Fermeture de connexion inconditionnelle : OSI TP4

La libération d'une connexion est dite inconditionnelle lorsqu'elle est autorisée à tout moment de la communication, y compris durant la phase d'ouverture de connexion. Une requête de déconnexion ne peut pas être rejetée par l'entité qui la reçoit. La figure (1.4) schématise ce mécanisme : l'entité réceptrice acquitte la requête par l'intermédiaire d'une DC-TPDU.

Ce mode de fermeture ne permet pas de garantir la délivrance des données en transit.

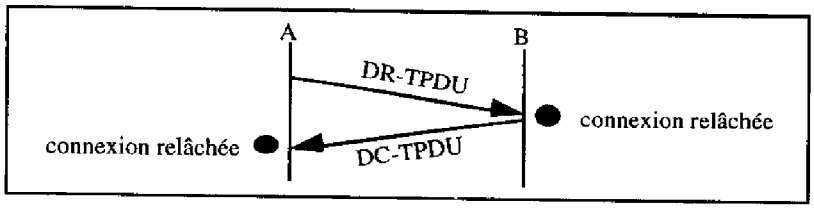


Figure (1.4) : Fermeture de connexion inconditionnelle

I.4.3. Fermeture de connexion forcée : NETBLT, TCP, OSI TP4

Une seule TPDU suffit ici à engendrer la fermeture de la connexion dans son ensemble (les deux flux de données sont fermés conjointement). L'initiateur de la fermeture relâche son contexte dès l'émission de la TPDU notée DR-TPDU sur la figure (1.5). L'entité réceptrice en fait de même, dès réception d'une DR-TPDU, sans acquitter cette TPDU.
Ce mode de fermeture ne permet pas de garantir la délivrance des données en transit.

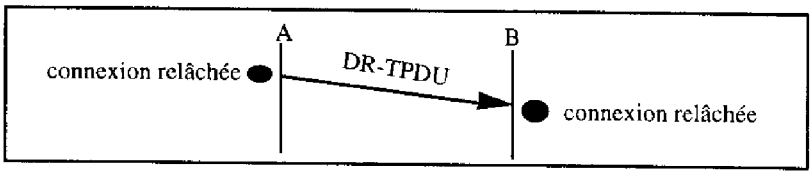


Figure (1.5) : Fermeture de connexion forcée

I.4.4. Fermeture de connexion négociée : TCP, URP/Datakit

Une fermeture de connexion est dite négociée lorsqu'elle garantit la délivrance de toutes les données en transit. Les deux flux de données d'une connexion de Transport bidirectionnelle sont fermés séparément par échange de trois ou quatre TPDUs.
L'entité désirant fermer son flux en émission le fait savoir à son entité paire par le biais d'une TPDU notée DR-TPDU sur la figure (1.6). Cette requête n'est acquittée par l'entité réceptrice que lorsque toutes les TPDUs en transit ont été correctement reçues. La connexion est entièrement relâchée lorsque les deux flux de données sont fermés.

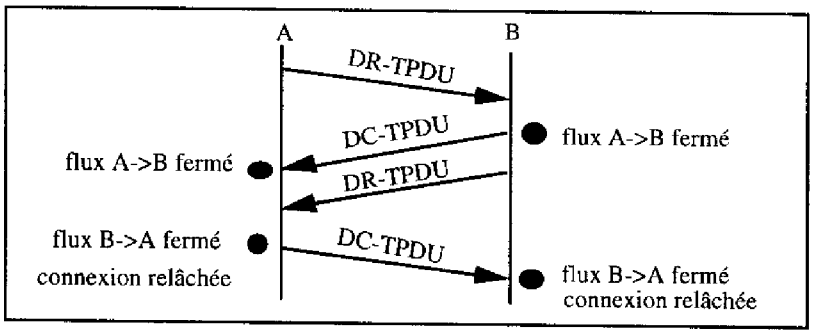


Figure (1.6) : Fermeture de connexion négociée à quatre messages

Remarque : l'échange de trois TPDUs est suffisant lorsque l'entité réceptrice requiert la fermeture de son flux d'émission en même temps qu'elle acquitte la requête de déconnexion émanant de son entité paire.

I.4.5. Fermeture de connexion forcée/négociée : XTP

La fermeture d'une connexion XTP est à considérer séparément des autres dans la mesure où elle peut être effectuée en mode forcé et/ou en mode négocié, de façon indépendante

sur chacun des flux. Comme précédemment, la délivrance des données en transit n'est garantie que pour le flux fermé en mode négocié (Figure 1.7).

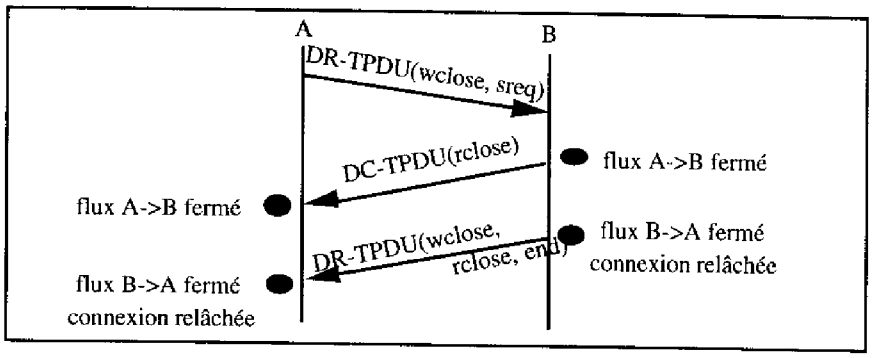


Figure (1.7) : Fermeture de connexion forcée (B->A)/négociée (A->B)

Dans ce schéma :

- le flux de A vers B (noté A -> B) est fermé de façon négocié :
  - le champ *wclose* de la TPDU notée DR-TPDU indique que l'entité A ne souhaite plus transmettre de nouvelles données. En activant le champ *sreq*, elle autorise cependant l'entité B à lui transmettre la liste des pertes observées : A retransmet alors les TPDUs perdues ;
  - lorsque l'entité B a délivré toutes les données, elle autorise l'entité A à fermer son flux en émission par activation du champ *rclose* de la TPDU notée DC-TPDU : cette événement marque la fermeture de part et d'autre du flux A -> B ;
- la fermeture du flux B -> A est quant à elle forcée :
  - l'entité B active les champs *wclose, rclose* et *end* de la TPDU DR-TPDU, ce dernier champ indiquant que le contexte associé au flux B -> A a été libéré. L'entité A est alors contrainte de fermer son propre contexte, sans assurer la délivrance des données en transit.

I.5. Mécanismes de transfert de données

I.5.1. Format des TPDUs

- fixe : XTP
- variable : VMTP, NETBLT, Delta-t, TCP, OSI TP4, URP/Datakit

Le format des TPDUs définit l'agencement des informations de contrôle et des données utilisateur dans une TPDU. Concernant la conception d'un protocole de Transport haute vitesse, la tendance actuelle est de diminuer le temps de traitement des TPDUs, au détriment de la bande passante. De façon à faciliter le décodage et le traitement parallèle des TPDUs :

- les champs d'une TPDU doivent avoir une place fixe, et une taille constante (ou multiple d'une constante), au prix d'un éventuel rembourrage ;

- les champs d'en-tête des TPDUs (champs de contrôle du protocole) doivent être positionnés de façon à autoriser un traitement parallèle ;
- le *checksum* doit suivre les données à contrôler, de façon à être évalué en parallèle lors de l'émission ou de la réception de ces dernières.

I.5.2. Segmentation et réassemblage :
VMTP, NETBLT, Delta-t, TCP, OSI TP4, URP/Datakit

Bien qu'aucune limite de taille de TSDU ne soit spécifiée dans la norme du service de Transport OSI orienté-connexion [IS8072], une taille maximale des NSDUs (NSDU - Network Service Data Unit) est généralement imposée par le service de Réseau sous-jacent. En fonction de la taille maximale des TPDUs qui en découle, il est alors nécessaire que l'entité de Transport émettrice segmente les TSDUs non véhiculables en une seule TPDU. L'entité de Transport réceptrice doit réassembler les TSDUs, avant de les délivrer à l'utilisateur.

I.5.3. Concaténation et séparation : OSI TP4

Ce mécanisme autorise l'entité de Transport émettrice à concaténer plusieurs TPDUs (appartenant à une même connexion ou à des connexion différentes) dans une seule NSDU ; l'entité réceptrice est alors contrainte de séparer les TPDUs concaténées lorsqu'une NSDU lui est délivrée. L'intérêt de ce mécanisme est de diminuer le nombre des NSDUs soumises au service de Réseau, et de diminuer ainsi l'utilisation de la bande passante ; cette optimisation est effectuée au détriment du temps de traitement des NSDUs par les entités de Transport émettrice et réceptrice. Compte tenu de la tendance actuelle visant à diminuer le temps de traitement des protocoles de Transport au détriment de la bande passante, le mécanisme de concaténation/séparation n'est plus d'un intérêt probant.

I.5.4. Groupage et Dégroupage

Le mécanisme de blocage/déblocage est également conçu pour diminuer l'utilisation de la bande passante. Plusieurs TSDUs sont concaténées par l'entité émettrice sur une seule et même TPDU ; l'entité de Transport réceptrice est contrainte de dissocier les TSDUs groupées dans une même TPDU, afin d'en assurer la délivrance. Ce mécanisme tendant à augmenter le temps de traitement des TPDUs, et donc le délai de transit des TSDUs, son utilité n'est plus justifiée.

I.5.5. Traitement des données urgentes : TCP, OSI TP4

Ce mécanisme permet à l'entité réceptrice de traiter des données de façon prioritaire.

I.5.6. Multiplexage : VMTP, NETBLT, Delta-t, TCP, OSI TP4, XTP

Le multiplexage de niveau Transport consiste en l'utilisation d'un même point d'accès au service de Réseau (NSAP - Network Service Access Point), pour véhiculer des TPDUs relatives à différentes connexions de Transport. En cela, ce mécanisme engendre une diminution des ressources réseau, utile notamment lorsque ce dernier limite le nombre de NSAP offerts. En contrepartie, la couche Transport réceptrice est contrainte de démultiplexer les paquets reçus tout en assurant un partage équitable des ressources entre les connexions de Transport multiplexées.
Outre le fait que le multiplexage induit un temps de traitement supplémentaire au niveau Transport, plusieurs raisons en diminue l'intérêt :

- les réseaux larges bandes basés sur la technologie ATM offrent un nombre de SAP quasiment infini à leurs utilisateurs ;
- les politiques de tarification seront sans doute bientôt basées non plus sur la durée de vie des connexions, mais sur la quantité de données transmises.

Lorsque le protocole de Transport multiplexe plusieurs connexions sur un même NSAP, un identificateur de connexion doit être introduit dans chacune des TPDUs, de façon à ce que le récepteur puissent associer les TPDUs reçues aux connexions de Transport correspondantes.

I.5.7. Éclatement et recombinaison

A l'opposé du multiplexage, le mécanisme d'éclatement consiste en l'utilisation de plusieurs NSAP pour véhiculer des TPDUs relatives à une même connexion de Transport. L'utilité de ce mécanisme se justifie lorsque l'utilisation de plusieurs réseaux à faible bande passante engendre une augmentation du débit global d'émission. L'émergence des réseaux à haut débit rend obsolète l'utilisation du mécanisme d'éclatement.

I.5.8. Négociation de paramètres : VMTP, NETBLT, TCP, OSI TP4, XTP, URP/Datakit
Renégociation de paramètres : VMTP, XTP, URP/Datakit
Sélection des modes d'opérations : URP/Datakit, VMTP, XTP

Avant que ne débute le transfert des données, il est nécessaire que les paramètres du protocole soient initialisés durant la phase d'ouverture de connexion. Certains d'entre eux, tels que la taille des tampons mémoires ou la valeur des temporisateurs, sont fixés pour toute la durée de la connexion ; d'autres, le numéro de séquence des TPDUs ou les paramètres de contrôle de flux ou de débit par exemple, sont mis à jour dynamiquement. La renégociation de ces paramètres durant la phase de transfert des données n'est pas envisagée dans tous les protocoles. Cependant, certains d'entre eux, XTP, VMTP ou

URP/Datakit notamment, permettent à l'utilisateur de sélectionner les modes d'opérations du protocole.

I.5.9. Politique d'acquittement

Pour qu'un transfert fiable des données de bout en bout soit garanti, il est obligatoire que l'entité de Transport émettrice conserve en mémoire toutes les TPDUs émises jusqu'à leur acquittement par l'entité réceptrice. Un acquittement n'est pas nécessairement un message de contrôle explicite, mais peut être une information de contrôle contenue dans l'en-tête d'une TPDU de données ; dans certains cas (VMTP par exemple), la réception de la réponse à une requête équivaut à un acquittement implicite de cette requête.
Plusieurs types de contrôle d'erreur ont été définis et sont détaillés au paragraphe (I.5.12). Dans un premier temps, nous présentons les différentes politiques d'acquittement existantes.

I.5.9.1. Acquittement à l'initiative de l'émetteur : VMTP, NETBLT, TCP, XTP, URP/Datakit

Une politique d'acquittement est dite dépendante de l'émetteur lorsque les acquittements sont générés par le récepteur en réponse à un événement provenant explicitement de l'émetteur. Par exemple, la réception d'un nombre donné de TPDUs valides, d'une TPDU hors séquence, ou d'une requête explicite de l'émetteur peut conduire l'entité réceptrice à générer un acquittement ;

- dans un protocole de Transport fondé sur le paradigme requête/réponse, l'utilisation de la nature transactionnelle de l'application permet d'optimiser le nombre de messages échangés, et de diminuer l'utilisation de la bande passante.
- lorsque l'émetteur réclame explicitement un acquittement des TPDUs précédemment émises, le récepteur n'a besoin de gérer ni compteur, ni temporisateur, pour déterminer le moment où il aura à émettre un acquittement : cette stratégie diminue le traitement au niveau du récepteur, au détriment de la charge de l'émetteur.
- lorsqu'il est activé, le champ *FASTNAK* de l'en-tête des TPDUs XTP impose au récepteur d'émettre un acquittement dès réception d'une TPDU hors séquence : ce mécanisme permet à l'émetteur de se voir notifier au plus tôt les pertes de TPDUs.

I.5.9.2. Acquittement indépendant de l'émetteur : OSI TP4

Dans cette stratégie basée sur la gestion d'un temporisateur, les acquittement sont périodiquement générés par le récepteur, sans aucune interaction avec l'émetteur [Wats81]. Si cette politique diminue la complexité du protocole, il apparaît cependant que la gestion du temporisateur est relativement coûteuse [Zhan86].

I.5.10. Contrôle de flux de bout en bout

Le contrôle de flux de bout en bout a pour objectif de permettre à l'entité réceptrice de réguler le débit de l'entité émettrice, en fonction de la taille de ses tampons mémoire disponibles. Ce contrôle évite que des données reçues soient rejetées, faute de place dans les tampons de réception en attente de traitement. Plusieurs stratégies sont envisageables.

I.5.10.1. Contrôle de flux de bout en bout par fenêtre :
NETBLT, TCP, OSI TP4, XTP, URP/Datakit

La première méthode est basée sur la gestion d'une fenêtre dont la taille est égale au nombre de tampons mémoire disponibles en réception. Toute TPDU émise véhicule la taille de la fenêtre de réception en tant qu'information de contrôle : l'entité qui reçoit une TPDU déduit de cette information la quantité maximale (ou crédit) de données qu'elle est autorisée à émettre. A chaque fois qu'elle envoie une donnée, l'entité émettrice décrémente son crédit d'émission et doit stopper ses émissions lorsque ce crédit devient nul [Clar82].

L'inconvénient majeur de cette stratégie réside dans le risque de périodes de silence qu'elle engendre : un crédit nul empêche une entité d'émettre des données, jusqu'à réception d'un nouveau crédit. De plus, les réseaux haute vitesse sont caractérisés par la valeur particulièrement élevée (plusieurs mégaoctets) du produit "débit x RTT" (RTT - Round Trip Time) signifiant qu'une quantité importante de données peut à présent se trouver en transit sur le réseau : si le crédit alloué ne permet pas à l'émetteur de transférer un tel volume de données, celui-ci sera contraint de stopper ses émissions par manque d'acquittement et de mise à jour de son crédit. Cette remarque justifie le codage de la fenêtre d'émission de XTP sur 32 bits.

I.5.10.2. Contrôle de débit : VMTP, NETBLT, XTP

Le contrôle de débit de bout en bout est un mécanisme conçu pour éviter les périodes de silence précédemment évoquées. Deux stratégies sont envisageables : le contrôle de débit par rafale et le contrôle débit par délai inter-message.

a) Contrôle de débit par rafale : NETBLT, XTP

Le contrôle de débit par rafale est basé sur la gestion d'un temporisateur. Selon les ressources dont elle dispose, une entité de Transport spécifie à la fois le débit maximum et la taille maximale des rafales d'octets que son entité paire aura à respecter en émission. Une fois ces paramètres échangés, le rapport (taille des rafales/débit maximum) équivaut au délai séparant les émissions consécutives de deux rafales d'octets.

Cette stratégie est mise en défaut lorsque la granularité en temps de la station sur laquelle est implanté le protocole est insuffisamment fine.

b) Contrôle de débit par délai inter-message : VMTP

Le contrôle de débit par délai inter-message est également basé sur la gestion d'un temporisateur. A la différence du contrôle de débit par rafale, aucune valeur du débit ni de la taille des rafales n'est spécifiée à l'émetteur. Dans ce contrôle, un délai inter-message de taille fixe doit s'écouler entre deux émissions consécutives de TPDUs, quelle que soit la taille des TPDUs. Ce délai peut cependant être réévalué en cours de communication, en fonction du nombre de retransmissions observées.

Notons qu'un contrôle de débit par délai inter-message équivaut à un contrôle de débit par rafale lorsque les TPDUs sont de taille fixes. Alors que ce dernier contrôle garantit théoriquement un débit d'émission constant, un contrôle de débit par délai inter-message engendre un débit variable lorsque les TPDUs n'ont pas une taille fixe.

I.5.10.3. Contrôle hybride : VMTP, XTP

Un contrôle de flux hybride peut être envisagé par la conjugaison des mécanismes de contrôle de flux par fenêtre et de contrôle de débit. Chaque entité doit respecter le délai inter-message (ou inter-rafale) imposé entre deux émissions consécutives, mais doit stopper ses émissions lorsque son crédit devient nul.

I.5.11. Contrôle de congestion

Les différents contrôle de flux précédemment décrits ont tous pour objectif d'éviter une saturation des tampons mémoires du récepteur. Cependant, ces mécanismes ne peuvent prévenir le fournisseur du service de Réseau d'une éventuelle congestion de ses propres tampons. Afin d'éviter ce cas de figure, il est nécessaire d'associer un mécanisme de contrôle de congestion aux mécanismes de contrôle de flux de bout en bout. Un tel contrôle peut être mis en œuvre de façon explicite ou implicite [Jaco88].

I.5.11.1. Contrôle d'accès explicite : XTP

Dans un contrôle explicite, le service de Réseau spécifie la valeur du débit d'émission que ne doivent pas excéder les entités communicantes. De façon à diminuer les risques de congestion au travers du réseau, les différents noeud du réseau sont autorisés à réduire la valeur de ce débit, y compris en deçà de la valeur théoriquement déduite des paramètres du contrôle de flux de bout en bout.

I.5.11.2. Contrôle d'accès implicite : VMTP, TCP

Un contrôle d'accès implicite a davantage pour but de résoudre que de prévenir une congestion du réseau. En effet, ce contrôle est mis en œuvre par l'entité de Transport émettrice au seul regard du nombre de retransmissions observées. Une telle méthode peut être considérée comme une évaluation du degré de congestion du réseau.

I.5.12. Multicast : VMTP, XTP

Un service de connexion *broadcast* (respectivement *multicast*) est offert lorsqu'un message émis peut être délivré à tous les récepteurs (respectivement à un groupe de récepteurs). Ces fonctionnalités, jusqu'alors développées au niveau liaison de données du modèle OSI, ont été étendues de façon à ce qu'un tel service soit offert par la couche Transport. Trois sémantiques de fiabilité ont été définies dans [Sant92] : dans la première, les TPDUs corrompues ou altérées ne sont pas récupérées ; la seconde garantit la récupération de toutes les erreurs ; dans la troisième, qualifiée de *statistiquement fiable*, les TPDUs sont supposés reçues correctement après un certain nombre de retransmission.

I.5.13. Contrôle d'erreur

Différents contrôles d'erreur sont envisageables ; tous dépendent de la stratégie d'acquittement adoptée.

I.5.13.1. Détection d'erreur : VMTP, NETBLT, Delta-t, TCP, OSI TP4, XTP, URP/Datakit

Pour offrir un service fiable, un protocole de Transport doit tout d'abord être pourvu d'un mécanisme de détection d'erreur. Un tel mécanisme repose toujours sur tout ou partie des trois champs suivants : le numéro de séquence des TPDUs échangées, un code de détection d'erreur, la longueur des données utilisateur contenues dans les TPDUs.

- La détection des pertes, des duplicata ou des TPDUs reçues hors séquence est gérée par l'entité de Transport réceptrice par le biais du numéro de séquence associé aux TPDUs ; ce numéro peut quantifier, soit le nombre de TPDUs émises (OSI TP4, URP/Datakit, NETBLT, VMTP), soit le plus grand numéro d'octet véhiculé dans la TPDU (XTP, TCP, Delta-t).
- Lorsque le code de détection d'erreur appliqué sur une TPDU reçue fournit le résultat de son champ "code" (TCP, OSI TP4), la PDU est jugée valide. Ce champ peut être scindé en deux sous champs, l'un portant sur l'en-tête de la TPDU, l'autre sur sa partie données (Delta-t, XTP, NETBLT) ; cette distinction permet d'envisager la délivrance de données erronées, et de réduire le temps de traitement des TPDUs en émission et en réception.

- La longueur du segment de données des TPDUs est nécessaire pour garantir une délivrance correcte des données lorsque leur taille n'étant pas constante.

I.5.13.2. Rapport d'erreur : VMTP, NETBLT, Delta-t, XTP, URP/Datakit

Un mécanisme de rapport d'erreur a pour but d'informer l'entité de Transport émettrice des occurrences de perte ou de corruption. Plusieurs stratégies peuvent être mises en œuvre.

- A chaque réception d'une TPDU hors séquence, URP/Datakit et Delta-t émettent un acquittement négatif, identifiant le dernier numéro (d'octet ou de TPDU) reçu en séquence (plus un). Ce rapport est également effectué lorsque XTP est mis en œuvre en mode *FASTNAK*.
- XTP, NETBLT et VMTP fournissent la liste des pertes observées par le biais d'acquittements dits sélectifs.

Notons qu'un mécanisme de recouvrement d'erreur par temporisation (TCP, OSI TP4), ne nécessite pas le rapport des erreurs : toute TPDU non acquittée est retransmise par l'entité émettrice à l'expiration du temporisateur qui lui a été associé.

I.5.13.3. Traitement des erreurs

Le traitement des erreurs dépend d'une part de la façon dont sont rapportées les erreurs, et d'autre part de la façon dont l'entité réceptrice traite les TPDUs reçues hors séquence (stockage ou rejet).

a) Acquittement positif avec retransmission : TCP, OSI TP4

Cette première stratégie est basée sur l'utilisation d'un temporisateur, associé à chacune des TPDUs émises. L'entité réceptrice acquitte toutes les TPDUs reçues en séquence et ne rapporte aucune des erreurs détectées. Les TPDUs stockées en émission en attente d'un acquittement sont retransmises à l'expiration de leur temporisateur. Ce traitement d'erreur allège la charge du récepteur au détriment de celle de l'émetteur.

b) Automatic repeat request (ARQ) : VMTP, NETBLT, Delta-t, XTP, URP/Datakit

Dans cette stratégie, l'entité de Transport émettrice fait usage du rapport d'erreur négatif (NACK) que lui notifie l'entité réceptrice, et peut soit appliquer une politique dite de "Go-back-N" (XTP, Delta-t, URP/Datakit), soit retransmettre sélectivement les TPDUs perdues ou corrompues (XTP optionnellement, NETBLT, VMTP).

- La technique du *Go-back-N* est appliquée lorsque l'entité réceptrice rejette toutes les TPDUs reçues hors séquence. Dans ce cas, le NACK indique seulement le dernier numéro (d'octet ou de TPDU) reçu en séquence : l'émetteur est alors contraint de

retransmettre toutes les TPDUs émises à partir de celle identifiée dans le NACK. Cette façon de procéder diminue la complexité du protocole, mais s'avère inadaptée lorsque le délai de transit des TPDUs dans le réseau est relativement long.

- La retransmission sélective des TPDUs perdues ou erronées (ou politique de *Selective Repeat*) est envisageable à condition que l'entité réceptrice stocke les TPDUs reçues hors séquence. Dans ce cas, le rapport d'erreur fournit la liste des TPDUs non reçues ou altérées, et permet à l'émetteur de ne retransmettre que ces TPDUs.

Notons enfin que de nouveaux protocoles de Transport (XTP notamment) permettent à l'utilisateur de désactiver certains des mécanismes de contrôle d'erreur du protocole. Cette flexibilité se justifie par les caractéristiques nouvelles des réseaux de communication ; de plus, certaines applications peuvent se satisfaire d'un transfert de données imparfait, et ne pas nécessiter que toute la procédure de récupération d'erreur soit mise en œuvre au niveau Transport.

I.6. Adéquation des protocoles de Transport aux exigences multimédia

En dépit de l'émergence de réseaux à haut débit, la mise en œuvre d'une application multimédia est contrariée par le goulot d'étranglement que représentent les protocoles de Transport traditionnels, incapables de répercuter la bande passante au niveau applicatif. De plus, les variations de débit et de délai de transit qu'ils génèrent sont préjudiciables du point de vue applicatif.

Dans ce contexte, il apparaît que les mécanismes définis dans le seul but d'optimiser la bande passante au détriment du temps de traitement du protocole (typiquement les mécanismes de groupage/dégroupage et concaténation/séparation) sont dorénavant à proscrire.

Au regard des caractéristiques applicatives nouvelles (contraintes temps réel, tolérance aux erreurs, isochronisme notamment), les mécanismes d'ouverture de connexion, et principalement les mécanismes de contrôle d'erreur et de contrôle de flux sont sujets à discussion et analyse [Dupu92].

Ouverture de connexion

Une ouverture de connexion implicite accélère la mise en œuvre de la phase de transfert des données, et ne présente que peu de risque de violation du critère de sûreté dans un environnement fiable. La principale critique que l'on puisse faire à ce mécanisme est qu'il interdit toute négociation d'une qualité de service. De plus, au regard des volumes de données générés dans le cadre d'une application multimédia, les gains induits par une ouverture de connexion implicite ne sont pas clairement estimables, comparativement à ceux consécutifs à une configuration optimale du système de transport. Cette observation conduit à privilégier un mécanisme d'ouverture de connexion explicite.

Contrôle d'erreur

Les données multimédia ne requièrent pas nécessairement un transfert fiable, mais rapide. Le recouvrement systématique des erreurs (TCP et OSI TP4) engendre une augmentation du temps de transit des données, potentiellement dommageable si l'application présente des contraintes temps réel. En cela, l'approche de conception de ces protocoles s'avère inadaptée au contexte actuel ; l'approche la plus appropriée est illustrée par les principes de conception du protocole XTP, autorisant l'utilisateur à définir, selon ces besoins, le type de contrôle d'erreur souhaité. On peut cependant remarquer que l'alternative "activation/désactivation" du mécanisme de reprise des pertes peut s'avérer insuffisante si l'utilisateur souhaite que certaines données soit effectivement délivrées, alors que d'autres peuvent ne pas l'être. Notons enfin que le mécanisme de *Selective Repeat* est plus approprié que le mécanisme de *Go-back-N* si l'application présente des contraintes de temps réel.

Contrôle de flux

Le contrôle de flux par fenêtre (TCP, OSI TP4, Delta-t, URP/Datakit) est source de baisse des performances ; mal configuré, il génère en alternance périodes de silences et émissions de rafales de taille variable, engendrant des variations de débit. Au regard de la constance de débit requise par des média tels que l'audio et la vidéo, ce mécanisme présente donc une inadéquation conceptuelle. De plus, si le débit global de la connexion peut théoriquement être maintenu, les émissions de rafales trop grandes sont source de perte par saturation des buffers de la carte du récepteur, et de ce fait conduisent potentiellement à une chute du débit [Mau93]. Afin d'éviter les périodes de silence, [Jaco92] propose d'élargir la fenêtre d'émission de TCP ; cette modification ne dispense pas de l'obligation qu'un contrôle de débit soit mis en œuvre afin de contrôler les rafales d'émission.

Quatre points supplémentaires sont sujets à discussion.

Parallélisation des traitements

Plutôt que de redéfinir les mécanismes de Transport, une autre approche consiste à optimiser le temps de traitement des protocoles classiques (TCP notamment). Plusieurs études [Clar89] [Jain90] présentent les résultats de cette démarche ; quoi que probant, ces résultats ne sont valables que dans des contextes d'implantation particulier. Cependant, appliquée à des protocoles haute vitesse tels que XTP, une telle approche ne peut qu'en augmenter les performances. La parallélisation des traitements soulève le problème d'un agencement adéquat des champs des TPDUs. Il ressort du paragraphe (1.5.1) la nécessité que ces champs soient situés à une place fixe, et que leur taille soit constante.

Gestion d'un contrôle d'accès

Face à la définition de nouveaux services réseaux, il est à présent obligatoire qu'un mécanisme de contrôle d'accès soit effectivement mis en œuvre à l'interface de Transport. Notons qu'en théorie, un tel contrôle permettrait de résoudre simultanément les problèmes de congestion du réseau, et le contrôle de flux de bout en bout.

Le multicast

Il est à présent admis qu'un service de multicast doit être offert au niveau Transport. Cependant, la volonté de garantir conjointement critères de performance et de fiabilité complique la conception des mécanismes correspondants ; ce problème reste à l'heure actuelle ouvert et fait l'objet d'étude [Sant94].

La synchronisation

Face aux exigences temporelles des applications multimédia, [Shep90] suggère qu'une synchronisation des flux multimédia soit gérée au niveau Transport. Si cette proposition présente théoriquement un intérêt du point de vue applicatif, elle nécessite cependant un accroissement de la complexité des protocoles ; en outre, les difficultés actuelles des systèmes d'exploitation à garantir une borne supérieure des temps de traitement laissent présager que les implémentations de ces mécanismes n'offriront pas une garantie totale. Un compromis moins contraignant consiste à ce que le Transport présente dorénavant des garanties quant au délai de transit maximum des données ; l'utilisateur pourrait alors choisir que lui soit notifiée la délivrance de données hors délai, ou que ces dernières ne lui soient pas délivrées. Une autre forme de QoS est également envisagée, la garantie d'une variation bornée du délai de transit autour de sa valeur moyenne (ou gigue) [Hehm90].
Ces dernières propositions constituent à l'heure actuelle un problème ouvert ; les mécanismes correspondants restent à définir, et nécessiteront des conditions spécifiques d'implantation du protocole (implantation matérielle ou logicielle dans un système d'exploitation temps réel).

En conclusion à cette première section, nous dégageons deux des points clefs qui contribueront à motiver les travaux présentés dans ce mémoire :

- les protocoles de Transport actuels ne présentent aucune garantie de QoS à caractère temporel : débit, délai de transit ou gigue dépendent directement de la qualité de service fournie par le réseau ;

- certains protocoles autorisent une prise en compte qualitative des exigences du multimédia. XTP, par la flexibilité de mise en œuvre de ses mécanismes, fournit la base la plus appropriée au transport des flux à haut débit [Thai95] ;

- aucun de ces protocoles ne permet de prendre en compte les relations de dépendance inhérentes aux objets multimédia ; en d'autres termes, la gestion de la synchronisation multimédia n'est pas envisagée au niveau des protocoles de Transport.

Cette dernière observation nous conduira aux chapitres 3 et 4 à proposer une extension conceptuelle des protocoles de Transport, dans le but de garantir une qualité de service adaptée aux contraintes des applications multimédia. La deuxième observation nous permet d'ores et déjà de justifier a priori le choix de XTP comme support de Transport de notre proposition (chapitre 5).

Nous présentons dans la section suivante les travaux actuellement menés autour de la définition de nouveaux services de Transport, haute vitesse et multimédia.

II. Services de Transport : limites actuelles et extensions envisagées

A l'heure actuelle, la majorité des interfaces d'accès aux protocoles de Transport ne sont pas normalisées ; en d'autres termes, il n'existe que très peu de spécifications formelles ou informelles du service offert, la qualité du service fourni (débit, délai de transit, taux d'erreur, etc.) dépendant de l'implantation du protocole et du service de Réseau utilisé.
De telles spécifications sont cependant utiles à plusieurs titres ; elles permettent par exemple d'accorder les différentes implantations sur les caractéristiques du service attendu. Leur intérêt majeur réside dans le fait qu'elles autorisent une classification précise des besoins applicatifs, dont la prise en compte aux différents niveaux du système de Transport peut être négociée entre utilisateur et fournisseur du service, en fonction des ressources du fournisseur et des caractéristiques du service de Réseau sous-jacent [Hehm90].
Actuellement, une partie significative des recherches se focalise sur la définition de nouveaux services de Transport, répondant aux contraintes d'une application multimédia [Dant92] [Ande92] [Diaz95]. Une attention particulière est notamment portée sur l'extension du concept de QoS [Viss86].
A ce jour, le service de Transport OSI sous ses modes connecté [IS8072] et non connecté [IS8072/ADD1] fait référence en la matière. Ce paragraphe souligne les limites de ce service dans le contexte d'aujourd'hui, et présente ses évolutions au travers de la proposition OSI 95 [Bagu92a].

II.1. Le service de Transport OSI et ses limites

Tel qu'il est défini, le service de Transport OSI supporte deux modes de service, *orienté-connexion* et *sans connexion*. Pour chacun de ces modes sont définies des primitives de service, dotées d'un certain nombre de paramètres. Ces primitives, ainsi que leurs règles d'enchaînement, définissent l'ensemble des interactions entre les couches 4 et 5 du modèle OSI.

Les modes du service de Transport OSI sont directement liés au mode de conception du fournisseur du service. Au regard de l'analyse des protocoles faite au paragraphe (I) :

- le service de Transport OSI en mode connecté garantit donc implicitement la délivrance dans leur ordre de soumission de toutes les TSDUs ;
- à l'opposé, en mode non connecté, le service de Transport OSI ne garantit ni la délivrance, ni l'ordre de délivrance des TSDUs soumises par un même émetteur à destination d'un même récepteur.

II.1.1. Rôle des primitives de service

Quatre type de primitive sont définis : requête, indication, réponse et confirmation :

- les primitives requête sont générées par l'utilisateur vers le fournisseur du service ;
- les primitives indication sont générées par le fournisseur vers l'utilisateur du service ;
- les primitives réponse sont générées par l'utilisateur vers le fournisseur du service en réponse à une primitive d'indication ;
- les primitives confirmation sont générées par le fournisseur vers l'utilisateur du service, en réponse à une primitive de requête.

Mode connecté

En mode connecté, les primitives de service permettent à l'utilisateur :

- d'établir et de libérer une connexion de Transport ;
- de négocier une certaine QoS avec le fournisseur du service ;
- de transférer des TSDUs sur une connexion point à point ;
- de contrôler le flux d'émission de l'utilisateur pair ;
- de transférer séparément des données urgentes.

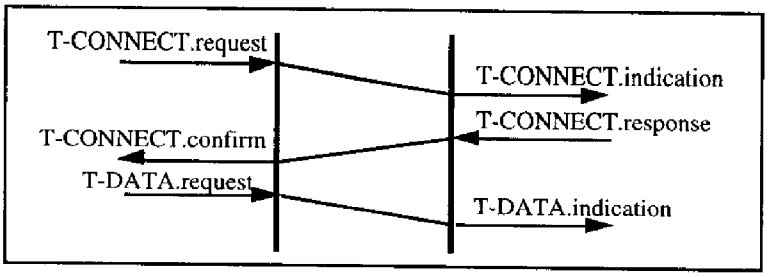


Figure (1.8) : Séquencement des primitives en mode connecté

Mode non connecté

En mode non connecté, seules sont définies les primitives de transfert des TSDUs hors connexion, avec possibilité de spécifier la QoS requise auprès du fournisseur du service. Aucune négociation de cette QoS n'étant envisagée, cette QoS n'est pas garantie.

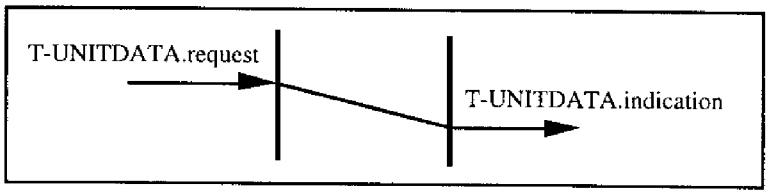


Figure (1.9) : Séquencement des primitives en mode non connecté

II.1.2. Rôle des paramètres de service

En théorie, le rôle des paramètres de service, en plus de l'indépendance des couches, est triple :

- ils permettent au fournisseur du service de sélectionner le service de Réseau le plus en accord avec les besoins de l'utilisateur ;
- ils autorisent la détermination précise des ressources que le fournisseur du service doit réserver ;
- ils permettent aux utilisateurs de s'accorder sur certains points, le débit de génération du trafic par exemple.

Certains de ces paramètres, le débit, le temps de transit ou le taux d'erreur résiduelles, sont relatifs aux performances du service durant la phase du transfert de données ; ces paramètres sont qualifiés de paramètres de QoS.

Mode connecté

En mode connecté, les paramètres de QoS sont négociés entre utilisateur et fournisseur du service ; le fournisseur est autorisé à affaiblir la valeur initialement requise de chacun d'eux. En théorie, la QoS ainsi négociée est garantie à l'utilisateur.

Mode non connecté

En mode non connecté, les paramètres de QoS ont une valeur purement indicative.

II.1.3. Limites du service de Transport OSI

Face aux nouveaux besoins applicatifs, les deux modes du service de Transport OSI présentent de nombreuses lacunes qui peuvent être résumées comme suit.

Une sémantique de QoS faible : le best effort

Dans la pratique, le fournisseur du service n'a généralement aucune latitude quant au choix du service de Réseau ; de plus, aucune spécification précise des paramètres de QoS n'étant définie, il n'existe aucune assurance que la QoS négociée soit respectée. Enfin, si la QoS n'est pas maintenue, aucune action n'est entreprise pour y remédier et l'utilisateur du service n'est même pas averti d'une dégradation de la QoS. Cette sémantique de service est qualifiée de *best effort*.

Impossibilité à permettre une QoS différente pour chaque sens de la connexion

Dans le cadre d'une connexion bidirectionnelle, les TSDUs sont échangées dans chaque sens avec la même QoS. Les applications de travail coopératifs telles que le télé-enseignement peuvent requérir une QoS différente pour chaque sens de la connexion. Cette possibilité n'est pas envisagée dans le service de Transport OSI.

Inadéquations à supporter des communications transactionnelles ou diffusées

Les modes connecté et datagramme du service de Transport OSI excluent toute prise en compte du caractère transactionnel des applications du type RPC (RPC - Remote Procedure Call). Cependant, pour de telles applications, le service de Transport CO engendre l'échange de plus de messages que n'en comporte la transaction. Le service de Transport CL s'avère inadapté dans la mesure où il contraint l'utilisateur à gérer par lui même les pertes éventuelles.

De même, par l'absence de définition de la notion d'adresse de groupe et de primitives de service adéquates, les deux modes du service de Transport OSI excluent la prise en compte du caractère diffusé (broadcast ou multicast) des applications multimédia.

Inadéquation à une prise en compte flexible des erreurs

Il est à présent admis que la sensibilité aux erreurs d'une donnée multimédia est différente de celle d'une donnée classique. Tel qu'il est défini, le service de Transport OSI ne permet pas à l'utilisateur de sélectionner de façon suffisamment flexible le type de contrôle d'erreur souhaité, détection et reprise, détection seule, ni reprise ni détection.

Inadéquation à permettre une configuration optimale du système de Transport

A la différence d'un transfert de fichier, la génération d'un trafic audio ou vidéo présente un caractère continu et isochrone, et nécessite une restitution finale analogue. La communication de ce type de média requiert donc qu'une certaine constance du débit soit

respectée. La spécification de cet isochronisme n'étant pas prise en compte dans le service de Transport OSI, le fournisseur du service ne peut pas exploiter cette caractéristique.

En théorie, les paramètres de QoS sont destinés à permettre une configuration optimale du système de Transport ; de plus, l'échange entre utilisateur de certains de ces paramètres durant la phase d'établissement de la connexion, permet de définir plus précisément les caractéristiques du trafic[3]. Alors qu'une limite de taille des TSDUs est imposée en mode non connecté, aucune limite n'est définie en mode connecté, dégageant l'utilisateur de toute contrainte de segmentation de ses données. L'absence de spécification de la taille des TSDUs complique la configuration des ressources locales du système de Transport [Hehm90].

Face à l'ensemble de ces lacunes, la tendance actuelle vise en étendre la notion de QoS ; celle-ci doit à présent être sélectionnable par l'utilisateur, et négociable entre utilisateur et pourvoyeur du service.

II.2. Les extensions du service de Transport OSI

Plusieurs évolutions ont été envisagées dans le cadre du projet ESPRIT RACE CIO ; elles ont notamment conduit aux propositions de service HSTS [Ande92], ETS [Cocq92] et OSI 95 [Bagu92a] [Bagu92b] [Dant93]. Ces extensions portent essentiellement sur les modes de service et sur les paramètres de QoS. Concernant ces derniers, une sémantique de service différente du *best effort* a clairement été définie.

II.2.1. Une sémantique de service précisément définie

Le service de Transport OSI 95 définit précisément l'action à entreprendre par le fournisseur du service en cas de dégradation de la QoS. Une dégradation de la QoS est définie par le non respect du *niveau* de l'un quelconque des paramètres de QoS.

OSI 95 définit deux niveaux de QoS, *seuil* et *obligatoire*. Le non maintien d'un paramètre à une valeur supérieure à sa valeur obligatoire constitue une dégradation inacceptable de la QoS ; l'utilisateur préfère dans ce cas que le service soit relâché. En revanche, si le niveau d'un paramètre passe en dessous de sa valeur seuil tout en restant au dessus de sa valeur obligatoire, la dégradation est jugée acceptable par l'utilisateur ; le fournisseur du service doit cependant lui notifier la valeur réelle du paramètre dégradé. Notons qu'un paramètre de QoS peut présenter une valeur *seuil* et une valeur *obligatoire*.

---

[3] Dans la pratique, seul ce dernier point est effectivement implémenté, le fournisseur du service n'assurant ni la prise en compte, ni la gestion de la QoS initialement requise. Les choix d'implantation d'une application distribuée reposent sur la connaissance, au niveau applicatif, non pas du seul service de Transport, mais des caractéristiques de l'environnement de communication utilisé (réseau local, Internet, etc.). Cet état de fait va à l'encontre des principes de structuration en couches du modèle OSI.

Ces différentes sémantiques confèrent au service de Transport une QoS garantie dans la mesure où toute dégradation d'un paramètre de QoS oblige le fournisseur du service à entreprendre une action spécifique, en accord avec le souhait de l'utilisateur.

Sur ces nouvelles bases, OSI 95 étend les modes du service de Transport par adjonction d'un certain nombre de primitives de service et de paramètre de QoS.

II.2.2. Les extensions du service de Transport OSI non connecté

Trois types de service en mode non connecté sont définis.

- Le *mode non connecté de base* est identique au mode non connecté classique, à la différence que le délai de transit peut présenter une valeur obligatoire. En ce cas, le fournisseur du service s'engage à ne pas délivrer une TSDU reçue si son délai de transit excède la valeur obligatoire requise.

- Le *mode non connecté acquitté* a pour but de fournir (sans garantie) à l'émetteur l'indication de la délivrance effective d'une TSDU transmise hors connexion. La figure (1.10) illustre les possibilités envisageables.

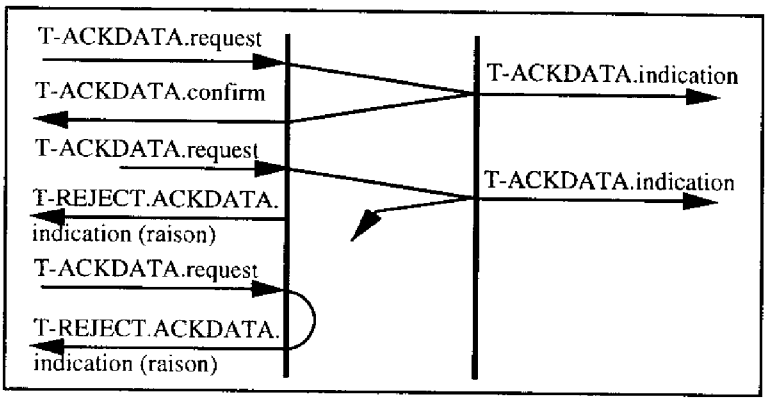


Figure (1.10) : Mode de service transport hors connexion acquitté

La primitive de confirmation T-ACKDATA.confirmation garantit de manière formelle que la TSDU émise a été correctement délivrée. La primitive de rejet T-REJECTACKDATA.indication indique qu'aucune garantie formelle ne peut être formulée quant à la délivrance correcte de la TSDU ; ceci n'exclut pas que la TSDU puisse avoir été effectivement délivrée.

- le *mode non connecté transactionnel* peut être obtenu de deux manières différentes :

  - soit en faisant deux fois appel au mode non connecté acquitté (figure 1.11) ;

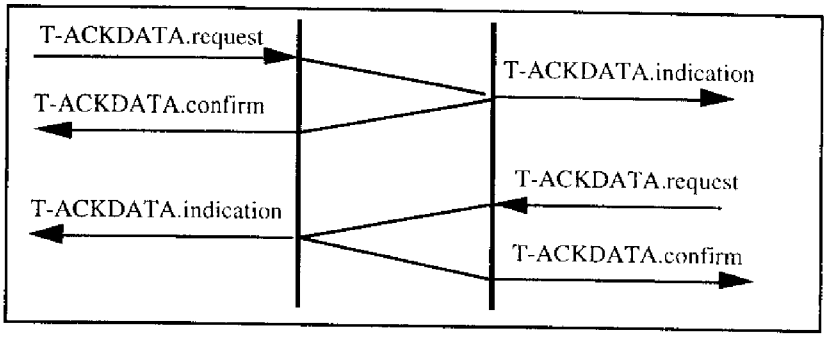


Figure (1.11) : Mode transactionnel basé sur le mode acquitté hors connexion

  - soit en associant une primitive de réponse à la primitive de requête (figure 1.12) ;

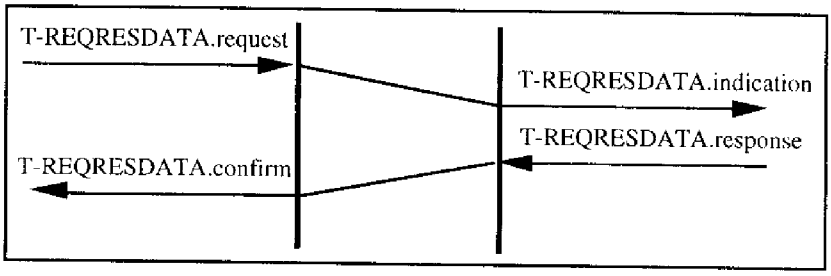


Figure (1.12) : Mode transactionnel basé sur des primitives requête-réponse

Remarquons dans ce cas que la délivrance d'une TSDU de réponse n'est pas confirmée ; ceci n'est pas un problème dans la mesure où l'initiateur de la transaction reçoit la confirmation du succès de la transaction par la TSDU de réponse.

II.2.2. Les extensions du service de Transport OSI connecté

Deux types de service en mode connecté sont définis dans OSI 95 : le *mode connecté enrichi* et le *mode orienté-connexion rapide.*

- L'enrichissement du mode connecté consiste en plusieurs points :
  - de nouveaux paramètres de service sont introduits ; l'utilisateur peux ainsi spécifier :
    - * le type de recouvrement d'erreur choisi ;
    - * un paramètre de gigue sur le délai de transit ;
    - * l'adresse d'un groupe de récepteurs ;
  - les paramètres de QoS (dont la gigue fait à présent partie) peuvent présenter les valeurs seuil et obligatoire introduites au paragraphe (II.2.1)
  - les règles de négociation des paramètres de QoS sont étendues. Elles ne sont pas présentées dans ce mémoire et sont disponibles dans [Bagu92a].

- Le mode *orienté-connexion rapide* autorise l'utilisateur à transmettre ses données avant d'avoir reçu confirmation d'une requête de connexion.

II.3. Conclusion

L'un des apports majeurs de la proposition OSI 95 réside dans la définition de nouvelles sémantiques de service, permettant de clarifier ce que signifie la *garantie* d'un paramètre de QoS. Cependant, ces définitions ne permettent pas d'envisager une extension conceptuelle de la QoS Transport : le service de Transport s'engage en fait à observer la QoS fournie par le service de Réseau sous-jacent, et à entreprendre une action spécifique en cas de violation des paramètres spécifiés par l'utilisateur. De plus, le caractère majoritairement asynchrone des systèmes d'exploitation actuels engendre une variabilité du temps de traitement des instructions : les mécanismes de Transport dédiés à une observation de paramètres tels que la gigue ou le délai de transit nécessiteront des conditions spécifiques d'implantation.

Enfin, nous soulignons ici l'aspect "monomédia" du service OSI 95 : en effet, les paramètres de service qui y sont définis ne permettent pas d'appréhender les flux multimédia dans leur globalité, c'est-à-dire en tenant compte des relations de dépendance des média entre eux.

Nous abordons dans le paragraphe suivant une présentation des différents projets actuellement menés dans le domaine des applications multimédia.

III. Panorama des architectures de communication multimédia

Différents projets sont en cours de réalisation dans les programmes de recherche ESPRIT et RACE de la Communauté Européenne. Au niveau national, plusieurs projets ont été mis en œuvre, dont BERKOM en Allemagne, MultiG en Suède [Pehr91] [Pehr92], et CESAME en France. Aux USA, le projet *Gigabit Testbed Initiative* doit conduire à la réalisation et à l'utilisation d'autoroutes numériques (*Data Super Highways*) pour transporter l'information multimédia, et en permettre le traitement distribué à travers tous les USA.

Nous présentons dans cette section les caractéristiques principales de quelques uns de ces projets. Une présentation plus détaillée de ces projets est disponible dans [Thai94a] [Thai94b].

III.1. BERKOM

BERKOM (Berliner Kommunikationssystem) [Pope89] [Delg93] est un projet allemand initié en 1986 par la "Deutsche Bundespost", le ministère de la science et de la technologie,

et le sénat de Berlin dans l'objectif de concevoir des systèmes numériques à intégration de services multimédia.
Trois services sont destinés à être supportés : un service de travail coopératif dans un environnement distribué (MMCS) [BERK92], un service de messagerie multimédia [BERK93], étendant les services X400 et ODA par l'échange de documents multimédia, et un service d'archivage multimédia.

L'architecture de communication

Actuellement, le projet BERKOM vise à définir une architecture de communication unique (MMT - MultiMedia Transport), devant intégrer la technologie ATM.
Le choix du protocole de réseau s'est porté sur ST-II dont la fonctionnalité majeure est de permettre une réservation des ressources du réseau [Topo90]. Au niveau Transport, une version du protocole XTP allégée de ses fonctionnalités réseau a été retenue (XTP-Lite) ; dans XTP-Lite, seuls les paquets DATA, FIRST, et CNTL sont conservés (un paquet ERROR faisant office de paquet DIAG), ces trois derniers paquets étant dotés d'un segment de QoS supplémentaire permettant d'envisager des phases de négociation et de renégociation de la QoS.
Quatre paramètres de QoS sont définis : la taille des TSDUs sous ses formes minimale acceptable, souhaitée et maximale, le débit souhaité et le débit minimal acceptable, le délai de transit souhaité et le délai de transit maximal acceptable et la classe de fiabilité (cinq classes sont définies et permettent à l'utilisateur de spécifier comment les pertes et les erreurs doivent être traitées - ignorées, indiquées ou corrigées). En outre, quatre classes de garantie sont définies : elles expriment la probabilité (haute ou basse) que la QoS requise sur chaque paramètre soit respectée, ainsi que l'action à entreprendre en cas de violation de la QoS (aucune ou notification).
Notons enfin que l'interface de Transport n'offre qu'un seul mode de service, le mode orienté-connexion, et ne couvre actuellement que les communications point à point ; l'établissement d'une connexion peut être effectué en mode normal ou rapide.

III.2. TENET

Le groupe TENET [TENE94], formé en 1989 à Berkeley, a pour objectifs de concevoir et de développer des services de communications temps réel, et d'étudier les implications des réseaux à haut débit sur des applications à média continus (du type audio et vidéo).

L'architecture de communication

L'architecture de communication TENET vise à garantir des performances. Deux protocoles de Transport sont disponibles : RTMP (Real Time Message Protocol) et CMTP (Continuous Media Transport Protocol).

- RTMP assure le transfert de messages de taille variable, avec garantie de délai, de gigue, de taux d'erreur et de bande passante ;
- CMTP offre un transfert de données de bout en bout à des utilisateurs générant des données à intervalles réguliers.

Tous deux reposent sur un protocole réseau orienté-connexion, RTIP (Real Time Internet Protocol), assurant la délivrance de paquets de longueur fixe sur un canal simplex, en respectant le délai, la gigue, le taux d'erreur et le débit requis par l'utilisateur.

Plutôt que de définir un service de Transport générique, directement paramétrable par l'utilisateur, l'approche adoptée dans TENET consiste à éclater le service de Transport en plusieurs services distincts. Chaque service supporte une classe d'utilisateurs ayant des caractéristiques similaires vis-à-vis de leur transfert de données. Deux types de services sont actuellement définis : le premier est dédié aux applications nécessitant un transfert sporadique de messages ; le second est destiné aux applications impliquant le transfert en temps réel de flux audio et vidéo [Wolf91].

III.3. QOS-A

L'architecture de communication QoS-A, développée à l'Université de Lancaster, a pour but de permettre la spécification et la prise en compte des contraintes de performances d'une application multimédia distribuée [Leop92] [Camp93]. Cette architecture vise à intégrer le traitement de la QoS à ses différents niveaux de protocoles.

L'architecture de communication

QoS-A est une architecture en couches incluant service et protocole de gestion de la QoS exprimée "flot par flot" au dessus d'un environnement réseau de type ATM. Un flot caractérise la production, la transmission et la consommation d'un type de données, typiquement un média d'un flux multimédia (ou flux monomédia).

Les besoins de l'utilisateur et les obligations du fournisseur du service de Transport sont négociés de façon contractuelle lors de la phase d'établissement de la connexion. Le contrat de service est formellement représenté par une structure indiquant :

- les caractéristiques de chacun des flux générés par l'utilisateur, incluant la taille et le taux de génération des SDUs, la taille des rafales, le débit crête, le délai de transit, le taux d'erreur maximum et la gigue ;
- la sémantique de service attribuée aux paramètres de performances ;
- l'adaptation requise en cas de violation de la QoS ; l'utilisateur peut choisir de s'adapter lui même à la situation, de renégocier la QoS du flux dégradé, de relâcher le service ou de ne rien faire ;

- le degré de maintenance de la QoS requis pour chaque flux : délivrance périodique des mesures de performance, maintient absolu de la QoS requise ou absence de maintenance ;
- le mode d'établissement de la connexion : normal, rapide, ou avec pré-allocation de ressources du réseau et de bout en bout ;
- le coût que l'utilisateur souhaite payer pour le service requis.

III.4. CIO

Le projet RACE 2060 CIO (Coordination, Implementation and Operation of multimedia services) regroupe 17 partenaires de 8 pays européens, et a été initié en 1992. Destiné à être achevé en 1995, l'objectif majeur de CIO concerne la spécification et le développement de deux téléservices multimédia : un service de messagerie multimédia (MMMS), et un service de travail coopératif, permettant à des utilisateurs distants de travailler sur un document dans le cadre d'une même application.

L'architecture de communication

De façon analogue au projet BERKOM, CIO vise à définir une architecture de communication multimédia unique (MMT - MultiMedia Transport), supportant les services précédants [Metz93].
Le système MMT est composé de plusieurs couches de protocoles :

- l'environnement de communication choisi repose sur les technologies Ethernet, FDDI et ATM ;
- le choix de XTP comme protocole à la fois de niveau Réseau et de niveau Transport est justifié par le fait que les fonctions réseau de XTP engendrent un overhead moins important que celles du protocole ST-II. Cependant, ni la spécification de paramètres de QoS, ni la réservation de ressources n'étant définies dans la version 3.6 de XTP, une extension du protocole a été proposée, donnant naissance au protocole XTPX (XTP eXtended). Comme dans le projet BERKOM, cette extension se traduit par l'adjonction d'un segment de QoS au format des paquets XTP [Milo93a]. La QoS porte essentiellement sur la gestion du contrôle d'erreur, ainsi que sur les paramètres de performance ; en mode connecté, ces paramètres sont le débit, le délai de transit et la gigue. Deux types de sémantiques de QoS sont définis pour chacun d'eux : le *best effort* et le *seuil*, ce dernier type équivalent à celui défini dans le service OSI 95.

Trois modes de service Transport sont définis : connecté, non connecté et transactionnel ; tous trois supportent des communications point à point, les communications de point à multipoints n'étant supportées qu'en mode connecté. Un service multipoint a été défini dans [Henc93]. Il utilise la fonction multicast du protocole XTPX. Les primitives offertes

concernent la transmission de données en multipoint, ainsi que la gestion de groupe. Différentes options peuvent être sélectionnées quant à la topologie de transmission (connexion de groupe ou connexion multipoint individuelle), l'expansion ou la réduction d'une topologie établie, l'adressage, la négociation et la maintenance de la QoS, la sémantique de délivrance, la fiabilité et le contrôle d'accès à un groupe. Les éléments de protocoles pour le multipoint sont décrits dans [Milo93b].

III.5. CESAME

Le projet CESAME (Conception formElle de Systèmes hAuts débits Multimédia coopEratifs) a été initié dans le cadre d'un partenariat entre le CNET le CCETT et le CNRS, dans le but de développer des architectures, des méthodes et des techniques permettant de concevoir des applications distribuées multimédia et coopératives [Diaz94a].

Ce projet associe neuf laboratoires du CNRS ainsi que des équipes du CCETT et du CNET. L'application qui servira d'illustration aux études menées dans le projet est une application de téléformation développée avec la société AIRBUS. Cinq domaines d'activités majeurs sont clairement dégagés dans CESAME :

- le transport et le transfert de données multimédia à haute vitesse ;
- la synchronisation multimédia ;
- la coopération de groupe ;
- l'évaluation des performances ;
- la réalisation d'une application de téléformation.

De façon à conserver le caractère générique des travaux menés dans le projet CESAME, le choix de l'environnement d'implantation des logiciels ne s'est pas porté sur un système dédié, mais sur les environnements SUN OS4, puis SOLARIS. Cette précision est importante car elle contribue à justifier les caractéristiques du service de Transport que nous proposons dans cette thèse.

L'architecture de communication

L'architecture de communication définie dans CESAME repose sur la technologie ATM. Suivant les volontés du CNET, le choix de la couche d'adaptation aux services fournis par l'ATM a été restreint à l'AAL5.

Une étude portant sur l'interfonctionnement des fonctionnalités de Transport sur l'AAL5 a confirmé l'intérêt de retenir le protocole XTP privé de ses fonctions de réseau comme base de Transport [Thai95]. Cette étude développe en particulier l'adéquation de XTP à supporter le transport de flux multimédia, tout en soulignant les lacunes du protocole à prendre en compte les paramètres de QoS à caractère temporel, tels qu'ils sont notamment spécifiés dans OSI 95 .

Deux services ont été proposés dans le cadre d'une architecture de Transport multimédia : un service de multicast partiellement fiable et un service d'ordre partiel.
Concernant le service de multicast, le protocole développé s'inspire de l'algorithme de "seau percé" de XTP, en utilisant les caractéristiques du réseau ATM [Sant92] [Sant94]. Ce service répond aux caractéristiques multipoint des applications multimédia et coopératives.
La spécification du service d'ordre partiel fait l'objet de notre contribution et sera détaillé dans les chapitres suivants de ce mémoire. Dès à présent, il est important de corréler la motivation de ce service au caractère asynchrone des systèmes d'exploitation et de communication, dans lesquels un service d'ordre partiel est destiné à être mis en œuvre.

Conclusion

Dans ce chapitre, nous avons présenté un état de l'art général du niveau Transport. Dans un premier temps, nous avons détaillé les mécanismes mis en œuvre dans les protocoles de Transport orientés-connexion. Leur adéquation aux exigences des applications multimédia a été discutée ; cette analyse a souligné l'absence de garantie de QoS que présentaient les protocoles actuels. Néanmoins, il est ressorti de cette discussion que le protocole XTP, par la flexibilité de ses mécanismes, était le plus adapté à supporter les contraintes de haut débit requises par les applications multimédia. Dans un second temps, nous avons énoncé les limites du service de Transport OSI, et détaillé ses extensions au travers de la proposition OSI 95. Les évolutions envisagées sont de trois types :

- l'ajout de nouveaux modes de services : transactionnel, datagramme acquitté, et prochainement multicast ;
- une extension de la sémantique des paramètres de QoS, ainsi que l'introduction d'un nouveau paramètre, la gigue ;
- la spécification de l'action à entreprendre en cas de dégradation de la valeur d'un paramètre de QoS selon sa sémantique de service (*obligatoire, seuil, best effort*).

La proposition de nouveaux modes de service est particulièrement intéressante car les mécanismes de protocoles correspondants existent, ou sont envisageables. Cependant, si les évolutions décrites dans les deux derniers points constituent en théorie une amélioration du service, elles ne permettent pas d'envisager une extension conceptuelle de la QoS Transport ; celle-ci repose directement sur la qualité du service de Réseau utilisé.
En outre, nous rappelons ici l'une des conclusions intermédiaires apportées en section (II.3) de ce chapitre, concernant l'aspect "monomédia" du service OSI 95. En abordant la notion de qualité de service "média par média", la proposition OSI 95 définit un service de Transport purement monomédia : elle n'autorise aucune prise en compte d'une quelconque expression des relations de dépendances entre les différents média d'un même flux

multimédia. Dans la dernière partie de ce chapitre, nous avons présenté quelques uns des grands projets actuellement menés dans le monde, et en particulier dans la communauté européenne. Hormis dans le projet CESAME, il apparaît que le point de vue adopté quant à la spécification du service de Transport est analogue à celui d'OSI 95 en ce sens que la QoS est prise en compte de façon monomédia.

Dans l'optique de définir une qualité de service de niveau Transport, nous abordons dans le chapitre suivant une extension conceptuelle de la notion de connexion, relative aux aspects ordre et fiabilité des services et protocoles de Transport. Nous montrerons que ces travaux permettent d'envisager la prise en compte des flux multimédia dans leur ensemble suivant une approche jusque là inexplorée dans le cadre d'un service et protocole de Transport. Le nouveau type de connexion que nous définissons permettra en particulier :

- d'utiliser les modèles à réseaux de Petri tel que l'OCPN et le TSPN, pour définir le choix du protocole, liant ainsi applications et protocoles (chapitre 3),
- de définir une architecture multimédia de niveau Transport, étendant, unifiant et intégrant les approches déjà développées tel UDP, TCP ou XTP (chapitre 4).

CHAPITRE 3 - LE CONCEPT DE CONNEXION D'ORDRE PARTIEL

Introduction

Le chapitre précédent a souligné que les extensions envisagées de la notion de qualité de service Transport reposaient en grande partie sur le service de Réseau utilisé et sur les conditions d'implantation des protocoles dédiés à cet effet.

Nous proposons dans ce chapitre une extension de la notion de connexion permettant de définir une qualité de service en terme d'ordre et de fiabilité. L'origine de cette réflexion résulte des lacunes conceptuelles des protocoles de Transport tels que TCP et UDP, qui nous conduisent à définir un nouveau concept de connexion : la *connexion d'ordre partiel* (*POC* - Partial Order Connection) [Amer93a] [Amer93b] [Amer94] [Diaz94b] [Conn94].

Nous montrons alors qu'au delà de l'extension conceptuelle qu'elle représente, une POC permet à l'utilisateur de spécifier les contraintes minimales de délivrance des données liées à la présentation des objets multimédia, et d'exploiter cette expression pour dégager des bénéfices en terme de bande passante, de mémoire et de délai de transit.

Ce chapitre est organisé de la façon suivante.

- La section (I) définit les notions d'ordre et de fiabilité des protocoles actuels, et souligne les limites des services offerts en terme d'ordre et de fiabilité. De cette analyse, nous déduisons la possibilité de concevoir un nouveau type de connexion permettant de définir et de mettre en œuvre tous les services d'ordre partiel et de fiabilité partielle.
- Nous caractérisons le concept de POC sous ses aspects fiable et partiellement fiable en section (II) ; l'intérêt d'une POC est mis en évidence au travers de deux exemples illustrant comment une connexion d'ordre partiel exploite le degré de parallélisme d'un flux multimédia, et optimise par là même l'utilisation des ressources en termes de mémoire et de bande passante ; de plus, il y apparaît qu'une POC engendre une diminution globale du délai de transit des données utilisateur.
- Afin de caractériser formellement le concept de connexion d'ordre partiel, une spécification en Estelle d'un protocole d'ordre partiel est présentée en section (III).

I. Généralisation des notions d'ordre et de fiabilité d'un protocole

I.1. Ordre et fiabilité d'un protocole

A l'heure actuelle, les transferts de données reposent sur deux types de protocoles, les protocoles orientés-connexion (CO - Connection-Oriented) et les protocoles sans connexion (CL - ConnectionLess).

Quatre critères sont généralement admis pour définir l'ordre et la fiabilité d'un protocole ; le premier de ces critères est relatif à l'ordre, les trois suivants à la fiabilité.

(1) la délivrance en séquence des données utilisateur ;
(2) la délivrance de données exemptes d'erreur ;
(3) la délivrance de toutes les données.
(4) la délivrance des données sans duplicata.

- Les protocoles orientés-connexion (tels que TCP) garantissent à l'utilisateur :
  - une fiabilité totale, car les PDUs perdues ou altérées sont toutes récupérées,
  - un ordre total, puisque l'ordre d'émission des données utilisateur est respecté lors de leur délivrance au récepteur,

  En cela, ils présentent des garanties d'ordre et de fiabilité totales, et définissent une classe de protocoles ***fiables et ordonnés*** ;

- A l'opposé, les protocoles sans connexion (tels qu'UDP) ne garantissent :
  - ni la fiabilité, car aucune action n'est entreprise pour détecter et corriger les erreurs,
  - ni l'ordre, dans la mesure où les SDUs sont traitées en tant que datagramme, indépendamment les unes des autres,

  En cela, ils ne présentent d'autre forme de garantie d'ordre ou de fiabilité que celle offerte par le service de Réseau sous-jacent, et définissent une classe de protocoles ***non fiables et sans ordre.***

L'avènement de nouveaux protocoles haute vitesse, XTP notamment, conduit à moduler les propos précédents. En effet, l'utilisateur des procédures XTP peut désactiver à volonté les mécanismes de recouvrement d'erreur définis dans XTP. En supposant qu'une connexion ait été établie entre entités XTP communicantes, cette connexion peut être utilisée en mode fiable ou non fiable ; dans ce dernier cas, XTP fait partie des protocoles orientés-connexion ***non fiables et ordonnés.***

Enfin, un protocole sans connexion mais dans lequel toute PDU émise est stockée en émission jusqu'à ce qu'elle soit acquittée, peut être qualifié de protocole sans connexion ***fiable et sans ordre***.
Notons cependant que le qualificatif "sans connexion" d'un tel protocole peut être discuté, la mise en œuvre d'un mécanisme d'acquittement et de retransmission impliquant un accord implicite entre émetteur et récepteur sur la numérotation initiale des PDUs.

I.2. Le concept de connexion d'ordre partiel

La mise en évidence des quatre classes précédentes permet d'envisager, au moins conceptuellement, une généralisation des services et protocoles actuels ;

- vis-à-vis du paramètre ordre, cette généralisation conduit à concevoir la classe des protocoles ***partiellement ordonnés***, garantissant aux utilisateurs tous les services d'ordre depuis l'ordre total jusqu'au "désordre" ;
- vis-à-vis du paramètre fiabilité, se dégage similairement la classe des protocoles ***partiellement fiables,*** fournissant tous les services de fiabilité entre la fiabilité totale et aucune garantie de fiabilité.

Nous définissons ce nouveau concept de service et protocole sous le terme de ***connexion d'ordre partiel*** (POC - Partial Order Connection). Une POC englobe les concepts classiques de connexion (TCP) et de service/protocole sans connexion (UDP). De façon la plus concise possible, une POC est une connexion de bout en bout permettant de définir et de mettre en œuvre tous les services et protocoles d'ordre partiel et de fiabilité partielle entre deux entités communicantes. Une POC permet aux utilisateurs de spécifier une QoS exprimable en terme d'ordre et de fiabilité, et garantit cette QoS.

Deux approches sont envisageables pour caractériser le concept de POC : par les mécanismes de protocoles, ou par les services d'ordre partiel et de fiabilité partielle. Afin d'en concevoir la portée, nous caractériserons tout d'abord ce nouveau type de connexion par l'intermédiaire du service et de son utilité dans le cadre d'exemples applicatifs.

Dans un premier temps, nous présentons brièvement les travaux relatifs à la gestion de l'ordre dans un canal de communication.

Le concept de F-Channel

Les conséquences théoriques de la gestion ou de l'absence de gestion de l'ordre dans un canal de communication ont déjà été considérées dans un contexte différent [Ahuj90]. Ahuja remarque ainsi que la notion de canal FIFO n'est pas indispensable à la conception de tous les algorithmes distribués [Lamp78] [Neig87]. S'il est vrai que cette notion en

facilite la conception et la vérification, c'est toutefois au prix d'une dégradation des performances, liée au coût de gestion du séquencement des informations échangées.
Afin de satisfaire les algorithmes qui requièrent un ordre non plus total mais partiel dans l'échange des messages, [Ahuj90] introduit la notion de *F-Channel*, canal fiable dans lequel les messages peuvent se doubler selon certaines règles ; quatre classes de message sont définies indiquant si un message :

- peut ou non doubler un autre message ;
- peut ou non être doublé par un autre message.

Cependant, ces règles ne permettent pas d'envisager tous les ordres partiels imaginables, et de ce fait ne permettent pas une extension conceptuelle des protocoles suffisamment satisfaisante.

Le protocole Psync

Peterson introduit également la notion d'ordre dans le cadre d'un protocole appelé Psync [Pete89]. L'ordre partiel est codé dans chaque message, par l'intermédiaire des durées séparant leur dates d'émission et de réception ; l'ordre partiel est mis à jour dynamiquement lors de chaque émission. L'inconvénient majeur de ce protocole réside dans sa complexité, qui préfigure d'une implantation coûteuse et donc peu performante.
Notre approche diffère de celle-ci par le fait que l'ordre partiel d'une POC nécessite d'être établi avant que les données ne soit transférées, simplifiant ainsi la complexité du protocole.

II. Caractérisation d'un service d'ordre partiel

La perception d'un document multimédia n'est pas uniquement séquentielle, mais peut présenter un caractère parallèle ; dans une vidéoconférence, image et voix sont par exemple perçues de façon simultanée. Nous montrons dans cette section comment une connexion d'ordre partiel permet de prendre en compte cette caractéristique, et de diminuer l'utilisation des ressources mémoire et bande passante.

Dans un premier temps, nous caractérisons la notion d'ordre d'une POC, sans considérer le paramètre fiabilité.

II.1. Service d'ordre partiel fiable

Une service d'ordre partiel ne présente un intérêt pratique que s'il existe des applications distribuées se satisfaisant d'un transfert non plus totalement, mais partiellement ordonné. Le fait est que de telles applications existent, notamment par les notions de parallélisme et/ou de périodicité qu'elles induisent [Litt90a]. Les paragraphes (II.1.1) et (II.1.2) suivants présentent deux exemples illustratifs des notions de périodicité et de

parallélisme d'une application, et justifient pour chacun deux l'utilisation d'un service d'ordre partiel.

II.1.1. Adéquation dans un cas particulier

Considérons tout d'abord l'exemple introductif suivant, le rafraîchissement à distance de l'écran d'une station de travail.
Dans cette application, une station serveur rafraîchit périodiquement les quatre fenêtres de l'écran d'une station cliente. La communication entre les deux stations est effectuée au travers d'un réseau fournissant un service non fiable (Figure 3.1).

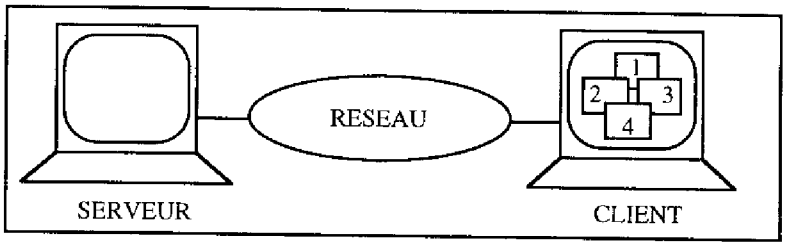


Figure (3.1) : Rafraîchissement à distance de l'écran d'une station de travail

Hypothèses

A chaque période de rafraîchissement, l'entité d'application serveur soumet le nouveau motif des fenêtres au service de Transport en tant que SDU complète, dans l'ordre : "motif de la fenêtre 1", "motif de la fenêtre 2", etc. Quatre SDUs sont donc soumises en émission à chaque période de rafraîchissement.

L'ordre d'apparition des motifs sur les fenêtres est supposé identique à l'ordre de délivrance des SDUs à l'entité d'application cliente.

Considérons les cas de figure envisagés Figure (3.2). Chacun d'eux expriment le degré de parallélisme de l'application, ici uniquement lié à la disposition des fenêtres à l'écran. La notion de périodicité se résume à celle de mise à jour des fenêtres.

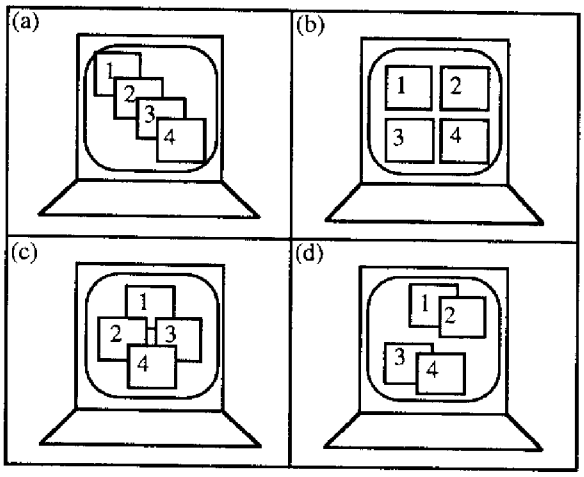


Figure (3.2) : Dispositions des fenêtres à l'écran de la station cliente

cas (a) : la disposition spatiale des fenêtres de la figure (a) rend obligatoire la délivrance des SDUs dans l'ordre (1, 2, 3, 4). Si la SDU 1 était délivrée après la SDU 2, la fenêtre 1 recouvrirait la fenêtre 2 et modifierait la disposition relative des fenêtres.

cas (b) : à l'opposé, la totale indépendance des fenêtres de la figure (b) autorisant le rafraîchissement des fenêtres dans n'importe quel ordre, toutes les séquences de délivrance sont admissibles.

cas (c) : la disposition des fenêtres de la figure (c) impose que la fenêtre 1 soit rafraîchie avant les fenêtres 2 et 3, et que ces dernières soient rafraîchies avant la fenêtre 4. En conséquence, la délivrance des SDUs suivant l'une quelconque des séquences (**1**, 2, 3, **4**) et (**1**, 3, 2, **4**) est acceptable du point de vue applicatif.

cas (d) : de façon analogue, la disposition des fenêtres de la figure (d) autorise la délivrance des SDUs suivant l'une quelconque des séquences (**1**, **2**, 3, 4), (**1**, 3, **2**, 4), (**1**, 3, 4, **2**), (3, **1**, **2**, 4), (3, **1**, 4, **2**) et (3, 4, **1**, **2**).

Partant de ce quadruple constat :

- un service de Transport totalement ordonné est nécessaire dans le cas (a) ;
- un service de Transport sans garantie d'ordre est suffisant dans le cas (b),

mais ni l'un ni l'autre ne permettent d'exploiter les caractéristiques de l'application dans les cas (c) et (d), liées à la disposition des fenêtres, et à la périodicité de leur mise à jour[4].

Intermédiaire entre ces deux extrêmes, un service d'ordre partiel tel que celui illustré sur la Figure (3.3-a) (respectivement 3.3-b) est acceptable pour l'application dans le cas (c)

[4] Notons cependant qu'un service de Transport totalement ordonné, est compatible avec la disposition des fenêtres de chacun des quatre cas.

(respectivement dans le cas d), chacun de ces ordres exprimant exactement le parallélisme de présentation des fenêtres à l'écran.

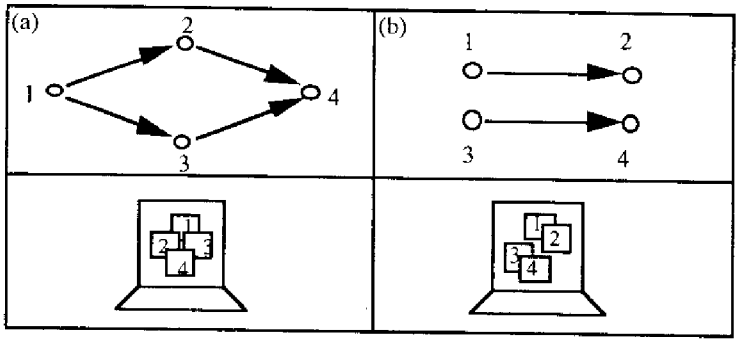


Figure (3.3) : Ordres partiels

Cet exemple illustre l'existence d'une application (et plus généralement d'un type d'applications), dont les caractéristiques de périodicité et de parallélisme sont ignorées par les services et protocoles actuels. L'extension de la notion d'ordre d'une connexion classique se justifie ici de façon pratique.

Dans le paragraphe suivant, nous montrons qu'une connexion d'ordre partiel présente un intérêt dans un nouveau contexte, les applications multimédia.

II.1.2. Adéquation au contexte des applications multimédia

Extrait de [Litt90b], l'exemple applicatif considéré consiste en la distribution d'un cours d'anatomie d'une station serveur vers une station cliente (Figure 3.4).

Sept fenêtres figurent à l'écran de la station cliente :

- deux icônes représentant respectivement un organe et la localisation de cet organe dans le corps humain figurent sur deux fenêtres ; ils sont remis à jour à chaque visualisation d'un nouvel organe ;
- des commentaires textuels sur l'organe sont affichés sur deux autres fenêtres ;
- sur deux fenêtres supplémentaires, apparaissent deux séquences d'images fixes présentant l'organe étudié en rotation autour d'un axe suivant deux angles différents ;
- une séquence vidéo présentant différentes courbes (telles que les variations du nombre de pulsations cardiaques au cours du temps, si l'organe visualisé est le coeur), apparaît sur une dernière fenêtre ;
- enfin, un commentaire audio des courbes précédentes est délivré au fur et à mesure de leur évolution dans le temps.

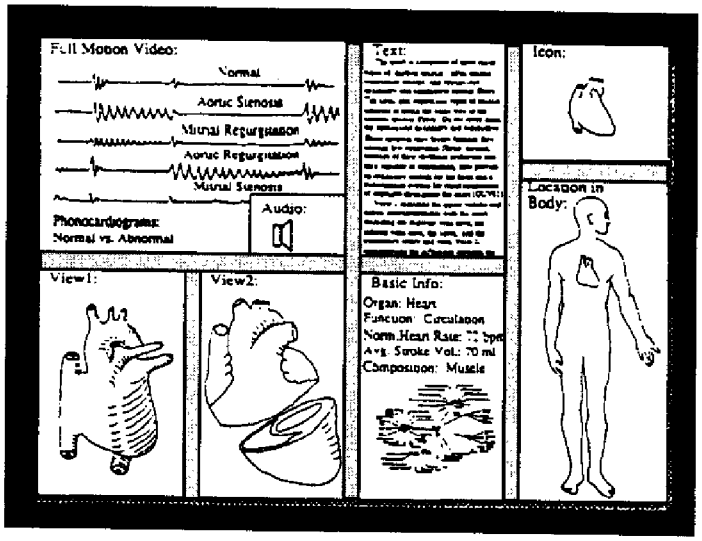


Figure (3.4) : Distribution d'un cours d'anatomie [Litt90b]

Les fenêtres ne se recouvrant pas, l'application présente un parallélisme de présentation analogue à celui du cas (b) de l'exemple des fenêtres ; cependant, il apparaît ici un nouveau type de contrainte lié à la chronologie de présentation des différents média. Ces contraintes expriment par exemple que :

- les images de chacune des séquences d'images fixes doivent être restituées dans leur ordre d'acquisition ;
- les images relatives aux deux séquences d'image fixes doivent être restituées de façon à correspondre au même cliché instantané (sous deux angles différents) de l'organe en rotation ;
- icônes et commentaires textuels doivent être à tout moment relatifs aux séquences d'images fixes qui s'affichent à l'écran ;
- etc.

Considérons le réseau de Petri déduit de l'application du modèle OCPN aux contraintes précédentes (Figure 3.5).

On y constate que l'ordre de présentation des textes et des icônes est indifférent ; on peut alors envisager toutes les séquences de présentation possibles depuis (1, 2, 3, 4) jusqu'à (4, 3, 2, 1). En revanche, les images fixes sont liées par un ordre partiel ; 7 et 8 peuvent être présentées dans un ordre quelconque, mais ni l'une ni l'autre ne peuvent précéder 5 ou 6, ni succéder à 9 ou 10. De manière globale, ce réseau de Petri fait apparaître un très grand nombre de séquences de présentation possibles des 19 objets, dont en particulier (1, 2, ..., 19), que l'on supposera être la séquence d'émission.

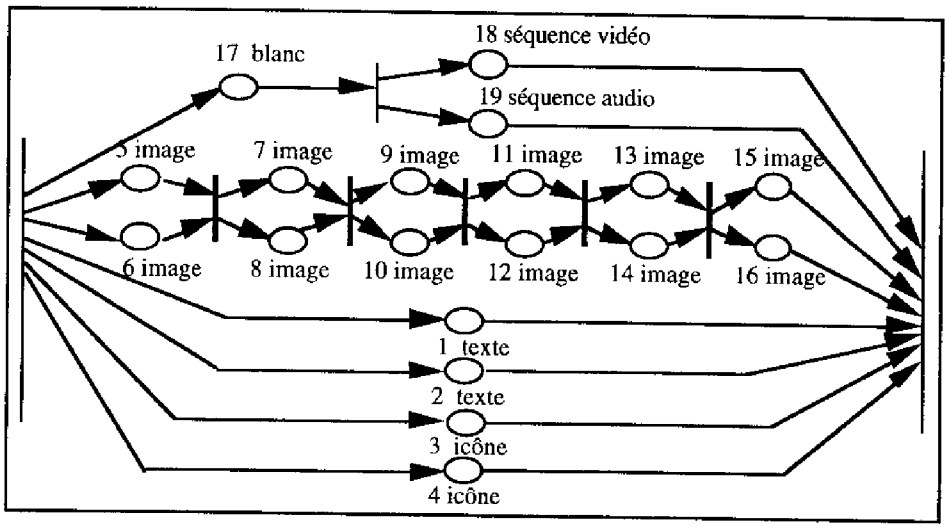


Figure (3.5) : Réseau de Petri déduit du modèle OCPN de l'application considérée

Si l'on confronte ce réseau de Petri aux garanties d'ordre d'un service de Transport.

- un service sans ordre (UDP) ne garantit pas la délivrance des objets 7 et 8 (par exemple), avant celle des objets 9 et 10, et de ce fait s'avère inadapté à l'application ;
- les données multimédia ayant été émises dans leur ordre de numérotation (de 1 à 19), un service totalement ordonné (TCP) garantit la délivrance correcte des données ; cependant, il ne tire aucun profit du parallélisme de l'application ;
- une connexion d'ordre partiel, respectant l'ordre de la Figure (2.4), prend exactement en compte les contraintes de synchronisation logique de l'application ; en cela, une POC autorise le fournisseur du service d'ordre partiel à exploiter les caractéristiques de l'application, tout en garantissant la qualité de service souhaitée par l'utilisateur en terme d'ordre de délivrance des données.

II.1.3. Gains induits par une POC

[Shaf92] suppose que la gestion d'un ordre partiel dans un canal de communication engendre une réduction du temps de transit des messages, du fait que le reséquencement de ces messages en réception n'est plus obligatoire.
Cette intuition est confirmée dans le cadre d'une POC à l'aide de l'exemple présenté au paragraphe suivant.

II.1.3.1. Diminution du temps de transit des données

Reprenons l'exemple des fenêtres avec pour disposition des fenêtres à l'écran, la configuration présentée (Figure 3.6a).

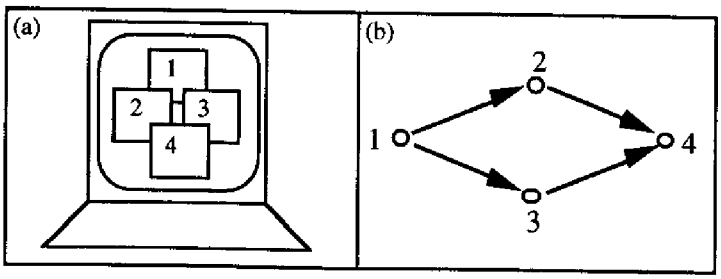


Figure (3.6) : Disposition des fenêtres

Considérons alors les cas de figure suivants :

cas (a) : l'application dispose d'un service d'ordre total ;
cas (b) : l'application dispose du service d'ordre partiel de la figure (3.6b) ;

et considérons les scénarios de la figure (3.7) dans lesquels la SDU 2 est perdue par le réseau.

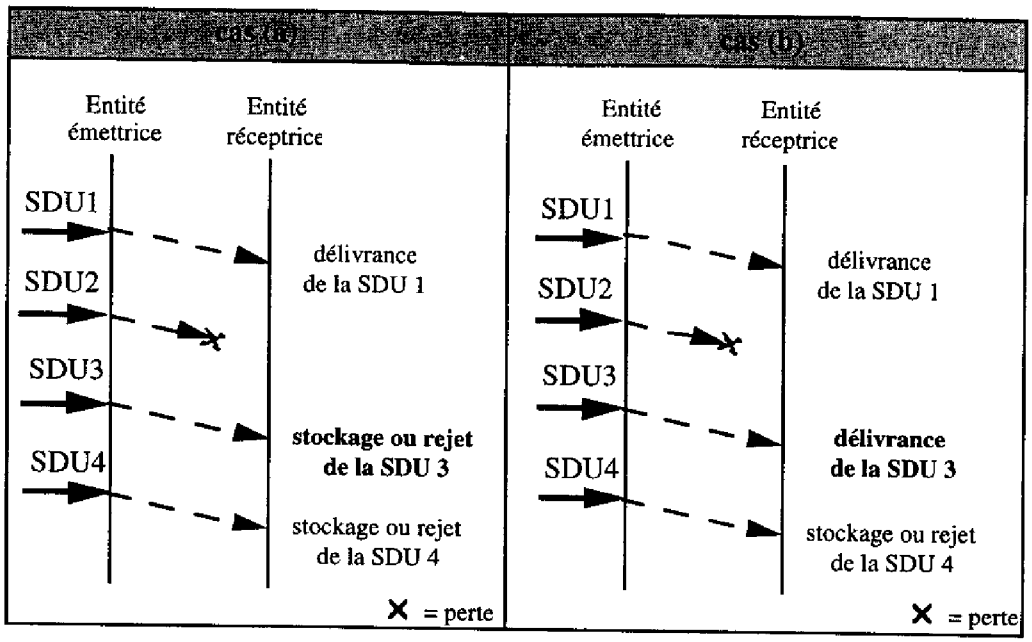


Figure (3.7) : Scénario particuliers

• Dans les deux cas, la SDU 1 est délivrée à la couche application dés sa réception.

• Sur réception de la SDU 3, le comportement des entités réceptrices diffère :

cas (a) : une seule séquence de délivrance étant admissible (la séquence d'émission), l'entité réceptrice est donc contrainte, soit de stocker la SDU 3, soit de la rejeter.

cas (b) : deux séquences de délivrance sont à présent admissibles : (1, 2, 3, 4) et (1, 3, 2, 4) ; l'entité réceptrice est donc autorisée à délivrer la SDU 3 sans avoir délivrée la SDU 2.

Du point de vue de l'application, la SDU 3 est ainsi délivrée plus tôt dans le cas (b) que dans le cas (a), tout en respectant la contrainte applicative imposée par la disposition relative des fenêtres.

II.1.3.2. Optimisation des ressources mémoire et bande passante

Le paragraphe précédent a montré qu'une connexion d'ordre partiel diminuait globalement le délai de transit des données utilisateur, par rapport à une connexion d'ordre total. Ce paragraphe montre qu'une POC diminue également l'utilisation globale de mémoire et/ou de bande passante.
Deux analyses sont présentées, l'une qualitative, l'autre quantitative. Il ressort de la première que la mise en œuvre d'un service d'ordre partiel induit toujours une utilisation de mémoire et de bande passante, au plus égale à celle induite dans le cadre d'une connexion classique. La deuxième analyse donne un aperçu quantitatif de l'économie mémoire réalisée dans une situation particulière.

Analyse qualitative

Soit un ordre partiel P de taille N, dans lequel le premier objet est indépendant des N-1 autres (c'est par exemple le cas de l'ordre partiel de l'exemple de Little et Ghafoor).

Supposons que N SDUs soient émises dans l'ordre (1, 2, ..., N) par l'entité d'ordre partiel émettrice ; supposons que la première de ces SDUs soit perdue par le réseau (Figure 3.8).

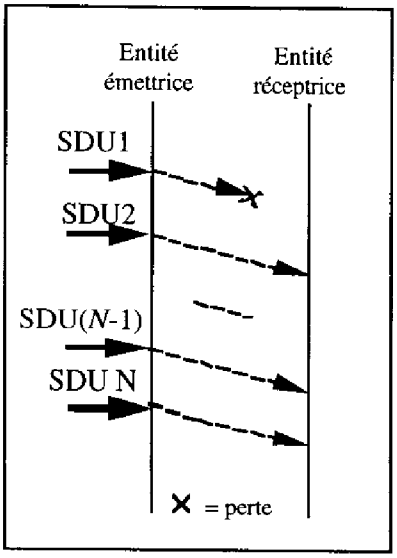


Figure (3.8) : Scénario considéré

Afin de fiabiliser la connexion, deux protocoles sont envisageables :

- le *Selective Repeat* dans lequel l'entité réceptrice (disposant d'un nombre de tampons de réception strictement supérieur à 1) stocke les données reçues "hors séquence admissible" au regard de l'ordre partiel considéré ,
- le *Go Back N* dans lequel l'entité réceptrice rejette les données hors séquence (le nombre de ses tampons de réception étant égal à 1) ; l'entité émettrice doit alors réémettre les PDUs correspondantes.

Ces deux possibilités sont à présent confrontées aux cas de figure suivants :
- l'ordre du protocole est l'ordre total ;
- l'ordre du protocole est l'ordre partiel *P*.

Hypothèse 1 : Stockage des données reçues hors séquence admissible.

• *1er cas de figure* : Ordre total

L'entité réceptrice est contrainte de stocker les SDUs numérotées de 2 à *N*, hors séquence admissible au regard de l'ordre partiel : *N*-1 tampons mémoire sont alors utilisés.

• *2ème cas de figure* : Ordre partiel

L'entité réceptrice est ici autorisée à délivrer les SDUs numérotées de 2 à *N* au fur et à mesure de leur réception, sans avoir précédemment délivrée la SDU 1. Ceci résulte du fait que l'objet 1 est par hypothèse indépendant des *N*-1 suivants dans l'ordre *P* de la connexion d'ordre partiel.

Dans l'hypothèse du scénario de la Figure (3.8), il apparaît qu'un service totalement ordonné engendre l'utilisation de *N*-1 tampons mémoire de plus que dans le cadre d'un service d'ordre partiel *P*.
Si l'application supportée ne nécessite pas la délivrance des SDUs dans leur ordre d'émission, il est clair que la mise en œuvre d'une connexion d'ordre total comme support de transfert engendre une sur-utilisation des ressources mémoire par rapport à une connexion d'ordre partiel.
Notons que ce scénario constitue le pire cas d'utilisation de mémoire.

Hypothèse 2 : Rejet des SDUs reçues hors séquence admissible.

• *1er cas de figure* : Ordre total

Le rejet en réception des SDUs numérotées de 2 à *N* engendre leur réémission par l'entité émettrice.

• *2ème cas de figure* : Ordre partiel

La délivrance autorisée des SDUs numérotées de 2 à *N* dispense des *N*-1 réémissions du cas précédent. Comparativement à un service d'ordre total, le service d'ordre partiel engendre un gain en terme de bande passante.

Comme sous l'hypothèse 1, si l'application supportée ne nécessite pas la délivrance des SDUs dans leur ordre d'émission, il apparaît que l'établissement d'une connexion d'ordre total conduit à une sur-utilisation de la bande passante, comparativement à une connexion d'ordre partiel.

L'exemple suivant évalue quantitativement l'économie réalisée en utilisation de mémoire par mise en œuvre d'un service d'ordre partiel comparativement à un service d'ordre total.

Analyse quantitative

Soit une application multimédia nécessitant le transfert de deux séquences vidéo sonorisées, l'une d'entre elles sous-titrée, et d'une présentation d'images fixes. Les trois présentations sont supposées spatialement indépendantes.

Le support de communication utilisé est un réseau ATM offrant une bande passante de 150 Mbits/s que l'on supposera entièrement dédiée à l'application.
Compte tenu du fait que 44 des 53 octets contenus dans une cellule ATM sont des données utilisateur, le débit "utile" d'émission est donc de 150*(44/53) = 125 Mbits/s.
Une seule connexion ATM de bout en bout est utilisée pour véhiculer les différents média de l'application.

Le délai de transit aller-retour (RTT - Round Trip Time), incluant le temps de passage dans les commutateurs est supposé de 200 ms.

La taille des informations de contrôle de protocole est supposée négligeable au regard de celle des données utilisateur. La taille et le temps d'émission des objets sont donnés par la figure (3.9).

média	Taille (octets)	Temps d'émission (ms)
image animée	50000	3.200
son	320	0.020
sous-titre	500	0.032
image fixe	30000	1.920

Figure (3.9) : Taille et temps d'émission des différents média

Le service d'ordre partiel est supposé celui de la figure (3.10), représentatif d'une période de deux secondes d'application.

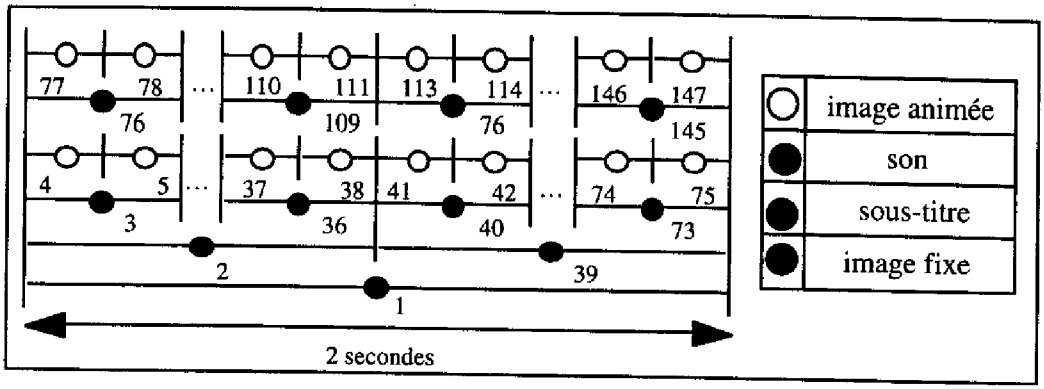


Figure (3.10) : Ordre de la connexion d'ordre partiel

Supposons que l'émetteur envoie les SDUs dans l'ordre numérique et que la première de ces SDUs soit perdue par le réseau.

En réception, selon la taille de sa fenêtre, un protocole de Transport classique va stocker ou rejeter les SDUs numérotée de 2 à *k*, où *k* désigne le dernière SDU reçue avant que la retransmission de la SDU 1 arrive. A l'opposé, un protocole d'ordre partiel délivrera les SDUs 2 à *k*, puisque leur délivrance est indépendante de l'objet 1.

Supposons que la stratégie soit de stocker toutes les données reçues hors séquence, et que le protocole utilise un mécanisme d'acquittement sélectif positif : toute PDU reçue est acquittée. Le temps de retransmission associé à l'émission d'une PDU est supposé de 250 millisecondes (ms).

Évaluons la valeur de *k* dans les deux cas de figure suivants.

Hypothèse 1 : Sans contrôle de débit en émission

Avant expiration des 250 ms, un protocole de Transport classique retransmettra les SDUs numérotées de 2 à 119 (en tenant compte des temps d'émission de la figure 3.9), soit : 2 sous-titres, 38 fragments de son et 78 images animées. La taille mémoire nécessaire au stockage de ces données est donc de 3,7 Moctets ; cette mémoire n'est pas utilisée lorsqu'un service d'ordre partiel est mis en œuvre

En outre, le délai de transit moyen de bout en bout des SDUs 1 à 119 est de 231,1 ms si le protocole est à ordre total ; il n'est que de 104,2 ms pour un service d'ordre partiel.

Hypothèse 2 : Avec contrôle de débit en émission

Supposons à présent que l'utilisateur émetteur exerce un contrôle de débit sur les données qu'il transmet.
Les images animées sont émises à raison de 1 toutes les 40 ms (soit 25 images par seconde). Seules les SDUs 2 à 24 sont à stocker avant réception de la SDU 1 retransmise. Au total, un protocole de Transport classique doit donc stoker 687 Koctets de données, stockage qui aurait été évité si un service d'ordre partiel avait été utilisé. Le délai de transit moyen de bout en bout des SDUs 1 à 24 est de 238,4 ms si le protocole est totalement ordonné ; il n'est que de 112,4 ms pour un service d'ordre partiel.

Cet exemple donne un aperçu des gains réalisés par mise en œuvre d'un service d'ordre partiel dans le cadre d'un scénario particulier.
Une campagne d'évaluation des gains induits par une POC est actuellement menée au LAAS. Elle repose sur l'utilisation de l'outil logiciel OPNET [OPNE]. Parallèlement, les travaux menés à l'Université de Delaware ont conduit à la spécification d'un modèle analytique d'un service d'ordre partiel [Mara96].

Le fait qu'il soit à présent admis que certains média ne requièrent pas une fiabilité totale, nous conduit à présent à discuter de la fiabilité d'un service d'ordre partiel.

II.2. Service d'ordre partiel et de fiabilité partielle

II.2.1. Caractérisation d'un service d'ordre partiel non fiable

Contrairement à l'ordre, la fiabilité d'un service de Transport CO a déjà été discutée et remise en cause.
L'innovation majeure consiste à offrir à l'utilisateur différents modes de contrôle d'erreur ; XTP recouvre l'ensemble de ces modes en permettant aux utilisateurs d'établir une connexion de Transport :

- avec ou sans détection des erreurs : les données altérées peuvent être délivrées ;
- avec ou sans reprise des données perdues par le réseau : les "trous" sont au mieux notifiés lors de la délivrance des données reçues.

Ces évolutions résultent de plusieurs observations :

- la fiabilité croissante des réseaux de communication justifie l'allégement des mécanismes de contrôle d'erreur initialement définis ;
- de plus, une fiabilité totale n'est pas obligatoirement requise par une application multimédia :

- le degré de sensibilité aux altérations de données différent selon le type de média considéré ; alors qu'un fichier texte nécessite un transfert exempt d'erreur, image et voix peuvent tolérer une certaine dégradation[5] ;
- de plus, la redondance de certains types d'unités d'information (les images d'une séquence vidéo par exemple) autorise un pourcentage de perte.

De façon plus générale, la perception simultanée des différents média d'une présentation multimédia étant par nature même non séquentielle, une certaine tolérance aux pertes est naturellement induite.

- enfin, la mise en œuvre d'un service de Transport fiable est susceptible d'engendrer des temps de latence relativement longs : une donnée perdue, puis récupérée dans le cadre du protocole, peut être délivrée dans des délais incompatibles avec les contraintes temporelles exprimées au niveau applicatif. Un tel service peut donc s'avérer inadapté lorsque l'application présente de fortes contraintes temporelles.

On peut cependant remarquer qu'aucune de ces évolutions ne permet de caractériser la QoS effectivement fournie en terme de fiabilité par le service de Transport ; celle-ci est fonction de la fiabilité offerte par le service de Réseau sous-jacent. Ainsi, l'approche consistant à inhiber tout ou partie des mécanismes de contrôle d'erreur d'une connexion d'ordre partiel conduit à définir une ***service d'ordre partiel non fiable*** comme un service d'ordre partiel sans autre garantie de fiabilité que celle offerte par le service de Réseau sous-jacent.

II.2.2. Caractérisation d'un service d'ordre partiel et de fiabilité partielle

Afin de définir une QoS "fiabilité" propre à la couche Transport, une solution simple en terme d'implantation consiste à l'exprimer par le nombre maximum k de pertes consécutives que le protocole peut ne pas récupérer, sans dégrader la QoS requise par l'utilisateur.

Alors qu'un service totalement fiable impose la mise en œuvre d'un protocole récupérant toutes les pertes et les erreurs, la fiabilité exprimée par k (k désignant le nombre maximum de pertes consécutives tolérées par l'utilisateur), autorise le protocole à ne récupérer les PDUs perdues qu'à partir de l'observation de k pertes consécutives.

Notons que la perte consécutive de p PDUs, avec $p \leq k$, ne constitue pas une dégradation du service négocié.

---

[5] Il n'est pas ici fait référence aux techniques de compression (d'image notamment), qui selon l'algorithme mis en oeuvre (MPEG, JPEG, ...) engendrent une tolérance aux fautes différente de celle induite par les caractéristiques propres du média considéré.

D'autres possibilités sont envisageables comme le pourcentage maximum de pertes, ou la combinaison des deux expressions précédentes[6] ; elles semblent de plus justifiées au regard de la sensibilité aux pertes que peuvent présenter des média continus.

De cette façon, on peut alors étendre les caractéristiques d'une POC fiable à celle d'une ***POC partiellement fiable***, implémentant tous les *services et protocoles d'ordre partiel et de fiabilité partielle*. En cela, une telle connexion permet de tirer partie d'une dégradation acceptable et parfois nécessaire de la fiabilité, au profit d'une diminution du délai de transit ; en outre, la gestion d'une fiabilité non plus totale mais partielle engendre une optimisation des ressources mémoire et bande passante [Amer93b].

Nous présenterons plus en détails l'impact d'une gestion de la fiabilité partielle au chapitre 4 de ce mémoire, dans le cadre de l'architecture de Transport multimédia qui y est proposée.

III. Spécification formelle d'un protocole d'ordre partiel

Les scénarios de délivrance présentés en (II.1) préfiguraient la mise en œuvre de mécanismes de gestion de l'ordre partiel. L'objet de cette section est de présenter la mise en œuvre du concept de POC au travers d'un protocole que nous avons intitulé POC. Le modèle utilisé pour la conception du protocole POC est le modèle de référence OSI de l'ISO [Zimm80].

La complexité des mécanismes mis en œuvre dans la conception d'un système réparti requiert l'utilisation de modèles représentant de manière non ambiguë le parallélisme et la communication. Ces motivations ont conduit à étudier plusieurs techniques de description formelle (ou FDT pour *Formal Description Techniques*) pour la spécification des services et protocoles du modèle OSI [Diaz89]. Les principaux résultats de cette étude sont les langages Estelle et LOTOS, normalisés au sein de l'ISO , et la technique SDL définie par le CCITT.

Afin de vérifier le fonctionnement des mécanismes de gestion de l'ordre partiel, nous avons retenu le langage Estelle comme technique de description du protocole POC. Avant de présenter le protocole, nous exposons en (III.1) les principaux concepts d'Estelle.

III.1. La technique de description formelle Estelle

Estelle (*Extended State Transition Language*) [Linn85] [Cour87] [Budk87] [ISO87a] [Tenn88] [Este88] [Este91] est fondée sur un modèle de machines à états étendues communiquant par files FIFO, et faisant appel au langage de programmation PASCAL

[6] Notons que les deux premières expressions ne sont pas indépendantes ; ainsi, une fiabilité exprimée en nombre maximum (k) de pertes consécutives conduit à accepter la perte de 50% des PDUs échangées si $k = 1$.

pour la représentation des données. Le typage des objets caractéristiques du langage PASCAL est étendue aux objets propres à Estelle tels que les modules, les points d'interaction, etc.

III.1.1. Concepts de structuration

Une spécification Estelle décrit une architecture hiérarchisée de composantes séquentielles non déterministes (appelées *instances de module*) échangeant des messages par le biais de *points d'interaction* associés aux différents modules.

Un module est défini à partir de deux objets Estelle qui décrivent son interface (*en-tête de module*), et son comportement interne (*corps de module*).
L'en-tête d'un module permet en particulier de définir les points d'interaction associés au module.
Le corps d'un module est organisé de la façon suivante :

- une partie *déclaration*, dans laquelle sont définis constantes, types de données, canaux, fonctions et procédures Pascal, ainsi que les états de contrôle de la machine à états (*états majeurs*);
- une partie *initialisation*, définissant l'état de contrôle initial, les valeurs initiales des variables et, s'il y a lieu, l'instanciation de modules fils (instruction *init*), ainsi que l'établissement de liaisons entre ces modules (instructions *connect* et *attach* appliquées sur les points d'interaction à relier par un canal) ;
- une partie *transition*, représentant la machine à états proprement dite. Une transition Estelle comprend trois parties :
  1) une partie *pré-condition*, qui permet de définir les conditions de tir de la transition considérée :
     - la clause *from* permet de définir l'état majeur à partir duquel la transition pourra être exécutée;
     - la clause *provided* permet de définir une expression booléenne comme condition de tir de la transition;
     - la clause *when* permet d'associer le tir d'une transition à la réception d'une interaction sur un point d'interaction donné;
     - la clause *delay* permet d'associer une temporisation (virtuelle) à l'exécution d'une transition ;
  2) une partie *post-condition*, qui permet de définir l'état majeur suivant l'exécution de la transition (clause *to*);
  3) une partie *action*, délimitée par un bloc *begin...end*, dans laquelle sont incluses des instructions Pascal et/ou Estelle ; une fois déclenchée, l'exécution de la partie action ne peut être interrompue.

Enfin, une spécification Estelle peut être dynamique si l'architecture des instances de module et les différents liens entre les points d'interaction de ces instances sont modifiés en cours d'exécution de la spécification.

III.1.2. La communication dans Estelle

Estelle offre un mécanisme de communication asynchrone basé sur la définition de files FIFO, associées aux extrémités des canaux de communication liant deux modules.
L'instruction *output*, définie dans le bloc *begin...end* d'une transition permet à une instance de module d'envoyer un message sur un point d'interaction donné. Ce message est alors placé dans la file FIFO liée à l'extrémité opposée du canal associé au point d'interaction considéré. Il est ensuite pris en compte par le module destinataire lors de l'exécution de la clause *when*, associée au point d'interaction correspondant, dès qu'il se trouve en tête de la file FIFO associée au point d'interaction considéré.

III.1.3. Expression du parallélisme

Deux modes de parallélisme peuvent être sélectionnés au sein d'une spécification Estelle : les modes synchrone et asynchrone.

- Les attributs *systemprocess* et *process* induisent un mode de parallélisme synchrone ; si dans une spécification Estelle, plusieurs modules contiennent une transition sensibilisée, l'exécution de la spécification est alors caractérisée par le tir simultané d'une transition par module. Cependant, dans le cas où un module M0 contient deux sous modules M1 et M2, les transitions de M1 et M2 ne seront exécutées que si aucune transition de M0 n'est sensibilisée ; cette priorité de tir des transitions d'un module par rapport aux transitions de ces modules descendants est appelée *priorité père/fils*.

- Les attributs *systemactivity* et *activity* induisent un mode de parallélisme dit asynchrone entre modules ; si dans une spécification Estelle, plusieurs modules contiennent une transition sensibilisée, l'exécution de la spécification est caractérisée par le tir d'une seule transition, le choix de la transition étant réalisé de manière aléatoire (non déterminisme). Cependant, la priorité père/fils est toujours prise en compte dans ce mode de parallélisme.

Enfin, indépendamment de l'attribut sélectionné, si plusieurs transitions sont sensibilisées au sein d'un même module, une seule d'entre elles est exécutée, le choix de cette transition se faisant de façon non déterministe.

III.2. Protocole d'ordre partiel fiable

La mise en œuvre d'un protocole d'ordre partiel nécessite qu'une phase d'établissement de connexion soit tout d'abord effectuée afin que chacune des entités communicantes prenne connaissance de l'ordre partiel de délivrance des SDUs.
Une fois l'ordre partiel connu de part et d'autre, il reste alors à définir les mécanismes permettant de déterminer si une SDU reçue peut être ou non délivrée à l'utilisateur. Nous supposerons dans ce paragraphe que le service d'ordre partiel offert est fiable.

Une première version du protocole POC a été spécifié en Estelle, et simulé à l'aide de l'outil logiciel EDB [EDB92]. Cette version a par la suite été poursuivie à l'Université de Delaware au moyen des outils Pet-Dingo [Sije93] et GROPE [Amer93]. Nous présentons dans ce paragraphe les mécanismes relatifs à la gestion de l'ordre partiel, communs aux deux versions. La spécification complète du protocole est fournie en annexe de ce mémoire ; elle est extraite de [Amer94].

L'architecture de la spécification Estelle est illustrée Figure (3.10). Le protocole POC est supposé utiliser un service de Réseau sans connexion, générant de manière aléatoire perte, duplication et/ou déséquencement des paquets échangés.

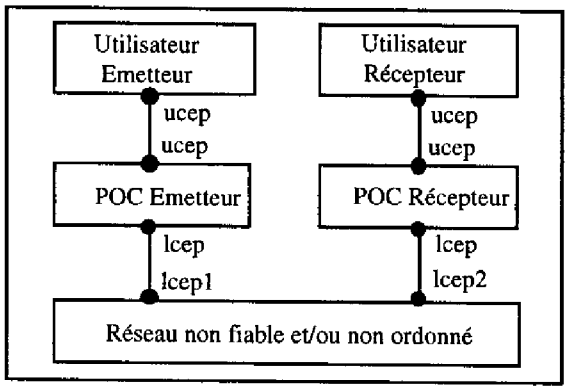


Figure (3.10) : Architecture de la spécification Estelle

Établissement d'une connexion d'ordre partiel

La figure (3.10) illustre la phase d'établissement d'une connexion d'ordre partiel :

- le paramètre ordre d'une POC est transmis par l'utilisateur émetteur, lors de la première invocation du service ;
- ce paramètre est ensuite échangé entre les sous-systèmes de Transport lors de la phase d'établissement de la connexion ;
- le sous système récepteur fournit enfin cet ordre à l'utilisateur destinataire, en tant que paramètre de service de l'indication de demande de connexion .

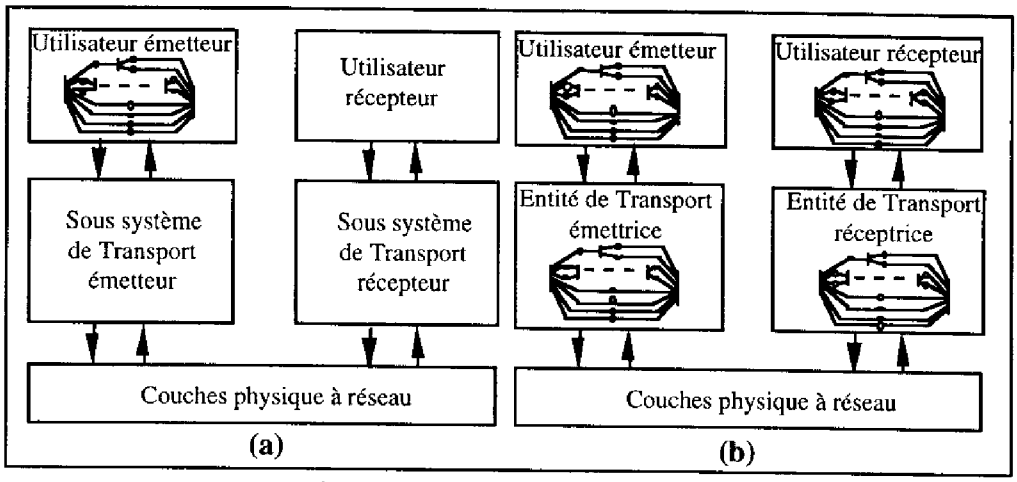


Figure (3.10) : Établissement d'une POC de niveau Transport

Cette phase n'est pas décrite dans notre spécification Estelle. Seule est spécifiée la phase de transfert des données durant laquelle est mis en œuvre le mécanisme de gestion de l'ordre partiel.

Phase de transfert de données

Afin de fiabiliser la phase de transfert de données du protocole POC, nous avons dû choisir un mécanisme d'acquittement parmi les différents possibles décrits (chapitre 2 - paragraphe I.5.14). Pour simplifier le protocole, le mécanisme adopté est l'acquittement sélectif positif. A toute PDU émise est associée un temporisateur ; à l'expiration de celui-ci, la PDU non acquittée est retransmise. Toute PDU non rejetée en réception est acquittée de façon à permettre la libération du tampon d'émission dans laquelle elle était conservée en vue d'une éventuelle réémission.
Les mécanismes propres à un protocole de Transport classique, tels que la segmentation et le réassemblage, n'apparaissent pas dans notre spécification. Pour simplifier la compréhension du protocole, chaque SDU soumise en émission est supposée véhiculée en une PDU.

Nous présentons dans le paragraphe suivant (III.2.1) les choix de représentation de l'ordre partiel. Le mécanisme de gestion correspondant est décrit en (III.2.2).

III.2.1. Représentation de l'ordre partiel

Il existe plusieurs façons de représenter un ordre partiel ; les diagrammes de Hasse ou de manière équivalente les réseaux de Petri en donnent une représentation graphique. Une autre approche consiste en l'énumération de toutes les extensions linéaires d'un ordre partiel.

Dans le cadre du protocole POC, nos choix se sont portés sur une représentation matricielle des ordres partiels, conduisant à une structure de données simple et adaptée. Cette représentation permet à l'entité réceptrice de déterminer dynamiquement si la délivrance d'une SDU est compatible avec l'ordre partiel de la connexion.

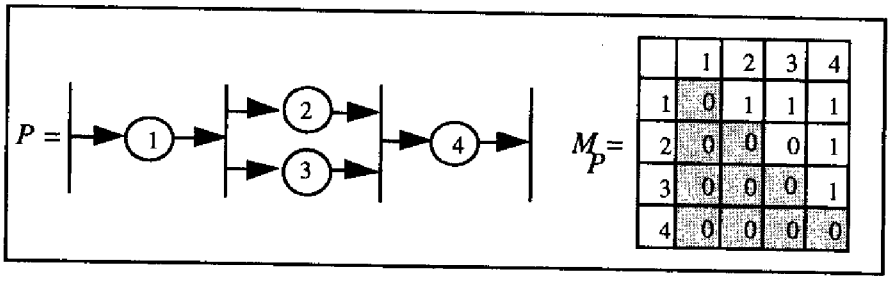
Représentation matricielle

Soit P un ordre partiel contenant N objets, dont la numérotation respecte l'ordre linéaire ; autrement dit, la séquence (1, 2, ..., N) est une extension linéaire de P.
Soit $E(P)$ l'ensemble de tous les couples (i, j), tel que l'objet i précède l'objet j dans P.

La représentation matricielle de P est donnée par la matrice M de taille $N*N$ telle que :

$$\begin{cases} M(i, j) = 1 \text{ si } (i, j) \text{ est un élément de } E(P) \\ M(i, j) = 0 \text{ sinon} \end{cases}$$

La figure (3.11) donne un exemple de représentation matricielle (MP) d'un ordre partiel initialement décrit sous forme de réseau de Petri (P).



	1	2	3	4
1	0	1	1	1
2	0	0	0	1
3	0	0	0	1
4	0	0	0	0

Figure (3.11) : Exemple

Remarquons que MP est triangulaire supérieure et à diagonale principale nulle. Cette propriété résulte du fait que la séquence (1, 2, 3, ..., N) est une extension linéaire de P. En conséquence, il n'existe aucun couple (i, j) tel que $M(i,j) = 1$ et $j \leq i$.

Cette remarque conduit à souligner que l'information attachée à la matrice M n'est contenue que dans $N*(N-1)/2$ éléments, et peut être codée par un vecteur de $N*(N-1)/2$ bits. Ce vecteur apparaît comme un possible paramètre de service, représentatif de l'ordre partiel spécifié lors de l'ouverture de connexion.
Dans l'hypothèse d'un codage de M colonne par colonne, le paramètre de service ainsi associé à l'ordre partiel de la figure (3.11) serait le vecteur : (1, 1, 0, 1, 1, 1).

Nous exposons dans le paragraphe suivant comment utiliser cette représentation pour assurer la délivrance des SDUs dans un ordre compatible avec l'ordre partiel.

III.2.2. Mécanisme de gestion de l'ordre partiel

[Shaf92] propose un algorithme basé sur une grammaire utilisant des règles, évoluant au fur et à mesure de l'arrivée des objets. Ce modèle consiste en un ensemble de canaux hiérarchisés, dans lesquels circulent des objets ayant des caractéristiques propres. Un objet d'un canal donné peut doubler tout autre objet d'un canal hiérarchiquement inférieur. Comparativement, notre approche matricielle simplifie le travail du récepteur.

Les SDUs sont supposées soumises par l'utilisateur émetteur sous la forme d'une répétition de périodes d'ordre partiel : outre le champ information, toute SDU soumise contient le numéro de la période à laquelle elle est relative, et son numéro de séquence dans la période considérée. Une SDU sera par la suite identifiée par un couple (i, j), i désignant son numéro de période, et j son numéro de séquence dans la période i.

Dans notre protocole, l'utilisateur est contraint de soumettre les SDUs dans une séquence compatible avec l'ordre partiel de la connexion. L'entité émettrice est chargée de contrôler le respect de cet ordre.

Étudions à présent le comportement des entités d'ordre partiel émettrice et réceptrice.

Entité émettrice

Afin de vérifier l'ordre de soumission des SDUs, l'entité émettrice gère une matrice d'ordre partiel correspondant à la période courante de soumission, selon les règles suivantes.
Soit (i,j) la SDU soumise, et soit M la matrice d'ordre partiel de soumission ;

- l'entité POC ne traite (i,j) que si la colonne j de M ne contient que des 0 ; en cas contraire, l'utilisateur est averti du non respect de l'ordre partiel de soumission, et la connexion est rompue ;
- l'émission de (i,j) entraîne la mise à 0 de la ligne j de M ;
- lorsque toutes les SDUs de la période courante ont été traitées, M est alors réinitialisée, et le numéro de la période courante est incrémenté.

Entité réceptrice

L'entité POC réceptrice gère une matrice M d'ordre partiel de délivrance (pour la période courante de réception) selon les mêmes règles qu'en émission. La réception d'une PDU engendre l'exécution des actions suivantes :

- la SDU (i, j) correspondante est délivrée à l'utilisateur récepteur si la colonne j de M ne contient que des 0 et si i correspond à la période courante de délivrance ; en cas contraire, (i, j) est stockée dans les buffers de réception ou rejetée si ces derniers sont pleins. Toute SDU stockée ou délivrée est acquittée par l'entité réceptrice.

- la délivrance de (i, j) entraîne une mise à 0 de la ligne j de M;
- lorsque toutes les SDUs de la période courante ont été délivrée, M est alors réinitialisée et la période courante de réception est incrémentée.
- de plus, la délivrance d'une SDU engendre une scrutation des buffers de réception, dans le but de délivrer toutes les SDUs stockées, à présent délivrables au regard de l'ordre partiel.

III.3. Protocole d'ordre partiel non fiable

Nous avons caractérisé le concept de POC partiellement fiable en (II.2.1) par le service correspondant, en soulignant que celui-ci équivalait à un service d'ordre partiel, offrant une fiabilité non plus totale ou nulle (c'est-à-dire fonction du service sous-jacent utilisé), mais compatible avec une certaine QoS fiabilité exprimée par l'utilisateur : nombre de pertes consécutives maximum, pourcentage de pertes, etc. Nous présentons dans ce paragraphe le mécanisme de gestion de la fiabilité du protocole POC jusqu'alors supposé fiable.

Nous introduisons tout d'abord une autre approche de la fiabilité reposant sur la définition de trois classes. Ces classes résultent du fait qu'une SDU peut avoir une validité temporelle finie, différente de celle des autres SDUs. Deux SDUs émises consécutivement peuvent par exemple ne pas posséder les mêmes contraintes de délai de transit.

III.3.1. Définition de trois classes de fiabilité

Les classes définies ci-après sont présentées du point de vue du protocole, et donnent une sémantique nouvelle au paramètre fiabilité d'une connexion d'ordre partiel.

Classe BART-NL

La première de ces classes comprend les PDUs dont la validité temporelle est infinie : ces PDUs véhiculent des données qui doivent être délivrées à l'utilisateur destinataire, quel que soit leur délai de transit au travers de la connexion. Cette classe est la classe *BART-NL*, pour *Buffer, Acknowledgment, Retransmission, Timer - Not Loseable*. Une PDU de classe *BART-NL* est stockée en émission jusqu'à ce qu'elle soit acquittée, et retransmise en cas de perte, autant de fois que nécessaire.

Classes BART-L(N) et NBART-L

La deuxième classe comprend les PDUs véhiculant des données de validité temporelle finie. La délivrance de ces données est inutile si elle est effectuée au delà d'une certaine limite temporelle. En corrélant la validité temporelle d'une SDU A au temps nécessaire à la retransmission des PDUs, il est alors possible, dans le cadre du protocole, d'évaluer le

nombre maximum N de retransmissions utiles de la (des) PDU(s) véhiculant A. Ce nombre permet de paramétrer la classe des PDUs *BART-L(N)*, signifiant *Buffer, Acknowledgment, Retransmission, Timer - Loseable* . L'entité d'ordre partiel émettrice libère automatiquement le tampon d'émission d'une PDU de classe *BART-L(N)* après N réémissions, si aucun acquittement n'a été reçu de l'entité réceptrice.

La classe des PDUs dont la retransmission est inutile ($N = 0$) a été définie dans [Amer94] sous le nom de *NBART-L*, signifiant *No Buffer, Acknowledgment, Retransmission, Timer - Loseable*. Ces PDUs ne sont pas stockées en émission, ni acquittées par l'entité réceptrice.

Nous présentons dans le paragraphe suivant la prise en compte de ces classes dans le cadre du protocole POC.

III.3.2. Mécanisme de gestion de la fiabilité partielle

Pour des raisons de simplicité, le mécanisme de gestion de la fiabilité défini dans le protocole POC ne prend pas en compte le paramètre N de la classe *BART-L(N)*. Ce mécanisme fait appel à une fonction particulière, *Is_Object_Still_Useful*, fournissant la liste des SDUs dont la délivrance est à présent inutile (les objets de classe BART-NL ne peuvent donc pas apparaître dans cette liste).

Comparativement au protocole POC fiable, seul le comportement de l'entité réceptrice diffère.

Entité réceptrice

En réception, la fonction *Is_Object_Still_Useful* est invoquée périodiquement, et la déclaration d'obsolescence d'une SDU entraîne la mise en œuvre des mécanismes suivants :

- toute SDU mentionnée dans la liste des données obsolète engendre l'acquittement des PDUs qui la supportent ; cet acquittement permet à l'émetteur de libérer le tampon de stockage des PDUs ;
- la déclaration de perte d'une SDU entraîne une mise à jour de la matrice d'ordre partiel, analogue à celle effectuée lors d'une délivrance de SDU ;
- une scrutation des buffers de stockage est effectuée de façon à libérer les tampons contenant des données temporellement obsolètes ; en outre, cette scrutation permet de délivrer les SDUs à présent délivrables au regard de l'ordre partiel.

Sur réception d'une SDU complète, l'entité réceptrice vérifie que celle-ci n'a pas été préalablement jugée obsolète, avant d'établir sa "délivrabilité" au regard de l'ordre partiel.

La fonction *Is_Object_Still_Useful* n'a pas été explicitement définie dans notre spécification, et peut être interprétée de deux façons différentes :

(1) soit le protocole POC dispose d'informations suffisantes pour mettre en œuvre un mécanisme de "décision et traitement" des pertes : la fonction *Is_Object_Still_Useful* est alors interne au protocole ;

(2) soit les informations précédentes doivent être fournies par l'utilisateur : le protocole exécute alors les actions faisant suite à la déclaration de perte des SDUs obsolètes.

Cette section nous a permis de caractériser le concept de POC par l'intermédiaire d'un protocole. L'objet de cette spécification a été d'illustrer de façon formelle la mise en œuvre d'un protocole d'ordre partiel et de fiabilité partielle.

Conclusion

Ce chapitre a présenté un nouveau type de connexion, la *connexion d'ordre partiel et de fiabilité partielle* (*POC*). Une POC a été définie comme une connexion de bout en bout permettant de définir et de mettre en œuvre tous les services et protocoles d'ordre partiel et de fiabilité partielle entre deux entités communicantes ; dans une POC, les SDUs peuvent être délivrées à l'utilisateur récepteur dans un ordre différent de leur ordre de soumission par l'utilisateur émetteur : la différence acceptable entre ces deux ordres résulte de la définition d'un ordre partiel de délivrance, préalablement négocié entre utilisateur et fournisseur du service ; la prise en compte de la notion de fiabilité partielle se traduit au niveau du protocole par la possibilité de ne pas récupérer toutes les PDUs perdues par le réseau si ces pertes sont acceptables du point de vue de l'utilisateur.
Au delà de son intérêt conceptuel, nous avons souligné à l'aide d'exemples l'adéquation de ce nouveau concept à prendre en compte les caractéristiques de périodicité et de parallélisme des applications multimédia, tout en satisfaisant les contraintes applicatives minimales en terme d'ordre et de fiabilité. Il est ressorti de cette analyse qu'une connexion d'ordre partiel diminuait l'utilisation des ressources mémoire et bande passante comparativement à une connexion classique, totalement fiable et ordonnée, et engendrait une réduction globale du délai de transit des données. Enfin, la notion de fiabilité partielle permet de prendre en compte le caractère redondant des flux continus, et de renforcer les bénéfices induits par une POC fiable. Afin d'en présenter une vue algorithmique dans un langage de description formelle, nous avons exposé la spécification en Estelle d'un protocole d'ordre partiel (POC). Dans le protocole POC, l'ordre partiel est représenté sous forme matricielle ; cette structure permet à l'entité réceptrice de déterminer dynamiquement si une SDU reçue est délivrable au regard de l'ordre partiel. Nous avons introduit l'aspect fiabilité du protocole par l'intermédiaire de trois classes, chacune accordant une certaine validité temporelle aux objets échangés.

Ce nouveau concept de connexion étend les deux approches traditionnelles de la notion de connexion représentées par UDP et TCP, qui apparaissent comme deux cas extrêmes. On remarquera de plus que les extensions envisagées dans XTP (mode CO non fiable et mode CL fiable), constituent également deux cas particuliers. Il apparaît ainsi que le concept de POC partiellement fiable étend, intègre et unifie toutes les approches à ce jour envisagées en matière de services et protocoles de bout en bout.

Les multiples façons d'appréhender l'ensemble des contraintes liées à la manipulation d'un flux multimédia ont conduit ces dernières années à la définition d'architectures de communication visant à intégrer la prise en compte de ces contraintes à différents niveaux, et notamment au niveau Transport. Dans ce contexte, le concept de POC fait apparaître la possibilité de considérer les flux multimédia dans leur globalité, c'est-à-dire en tenant compte des relations d'ordonnancement logiques liant chacun des flux ; en cela, on entrevoit ici une nouvelle approche dans la perception d'un flux multimédia au niveau des protocoles de Transport : nous montrons dans le chapitre suivant comment utiliser le concept de POC pour définir une architecture de Transport multimédia qui prenne en charge une certaine expression des exigences des applications multimédia.

CHAPITRE 4 - PRINCIPES DE CONCEPTION D'UNE CONNEXION DE TRANSPORT MULTIMÉDIA D'ORDRE PARTIEL

Introduction

Par nature, une application multimédia distribuée nécessite le transfert et la coordination de différents média. Ce type d'applications induit de nouvelles contraintes pour le système de communication, liées au caractère potentiellement isochrone des média manipulés, et à la nécessité que leur restitution soit effectuée de façon synchronisée. Par exemple, une application telle que la visioconférence implique le transfert et le traitement de flux de données audio et vidéo ; une restitution acceptable de ces média requiert que leur gigue de présentation n'excède pas une certaine limite et qu'une synchronisation entre eux soit effectuée en réception.
Face à cette problématique, les réflexions sur le transfert et sur la coordination des flux multimédia ont conduit aux conceptions conjointes de modèles de représentation d'objet multimédia tels que le TSPN, et d'architectures de communication multimédia, généralement inspirées du modèle OSI, répartissant la prise en compte des contraintes applicatives à différents niveaux [UK92] [Leop92] [Camp94a].

Concernant la conception d'architectures, deux tendances se dégagent clairement :

- la première repose sur une gestion des contraintes au seul niveau applicatif [Diot95]. Cette approche, qui suit le concept d'ILP (ILP - Integrated Layer Processing) de [Clark90] est discutable car elle remet en cause la structuration en couches du modèle OSI, et de façon plus générale l'approche modulaire jusqu'alors reconnue de tout système informatique. De plus, une telle approche ne présente a priori aucun caractère générique ;
- à l'opposé, la deuxième approche vise à exprimer l'ensemble de ces contraintes à différents niveaux. Vis-à-vis des protocoles de communication, ce point de vue s'est tout d'abord traduit de façon non constructive (les mécanismes correspondants restant à définir) par l'application de nouvelles sémantiques de service, essentiellement inspirées de [Dant92], à des paramètres de QoS (délai de transit et gigue notamment) nécessaires à l'utilisateur pour mettre en œuvre ses mécanismes de synchronisation multimédia. Sur les bases des extensions apportées à la technique de description formelle LOTOS, les mécanismes de gestion de ces paramètres font à l'heure actuelle l'objet d'étude à l'Université de Liège.

Les travaux décrits dans ce chapitre s'inscrivent dans le cadre de cette deuxième approche. Nous définissons les principes d'implantation d'une connexion de Transport multimédia d'ordre partiel, à QoS garantie en termes d'ordre partiel et de fiabilité partielle ; nous montrons comment une telle connexion permet de définir un service de synchronisation logique d'ordre partiel, et engendre une diminution globale du délai de transit, au prix d'une dégradation acceptable de la fiabilité, tout en respectant l'ordre partiel déduit d'une modélisation TSPN des contraintes de synchronisation temporelle.

Ce chapitre est articulé en trois parties majeures.
Dans la première section sont présentés les principes de conception d'une connexion de Transport multimédia. Nous présentons dans la deuxième section l'intégration du concept de POC partiellement fiable dans le cadre d'une telle architecture. La dernière section fournit une définition précise du service correspondant.

I. Architecture

Par définition, une application multimédia distribuée est composée de plusieurs média ayant souvent des caractéristiques de transfert intrinsèquement différentes. Ainsi, une application composée de données textuelles et de vidéo nécessite que le service de Transport utilisé garantisse à la fois une communication fiable des données textuelles, et un débit minimum de transfert des images animées.
L'utilisation d'une connexion de Transport "classique" pour acheminer les données d'une telle application nécessiterait que le service fourni satisfasse la qualité de service la plus contraignante pour chacun des flux. Cependant, une fiabilité parfaite n'est pas indispensable au transfert de la vidéo et par ailleurs, l'utilisation d'un service de Transport fiable engendrerait des délais de transmission supplémentaires lors de la récupération des pertes, a priori incompatibles avec une présentation acceptable de la vidéo.

Connexion de Transport Multimédia : multi-connexions de Transport monomédia à QoS données

Compte tenu de l'aspect multi-flux d'une application multimédia et des différences de QoS requises pour chacun de ses flux, il apparaît donc nécessaire de définir une nouvelle architecture de Transport, *multimédia*, garantissant un ensemble de qualités de service, sélectionnables pour chacun des flux d'un même utilisateur. Ce point de vue est renforcé par [Tenn89] et [Crow92], qui soulignent l'inutilité d'associer différentes QoS à des connexions de même niveau N multiplexées sur une connexion de niveau N-1 présentant une certaine QoS.

Dans l'architecture que nous proposons, une connexion de Transport multimédia implique l'instanciation, puis la coordination de plusieurs connexions de Transport monomédia, à

qualités de service distinctes ; chacune de ces connexions fournit une QoS adaptée aux caractéristiques de transfert d'un flux utilisateur particulier.

Pour illustrer cette architecture, considérons l'application multimédia présentée (Figure 4.1), reprise d'un exemple de [Amer94].

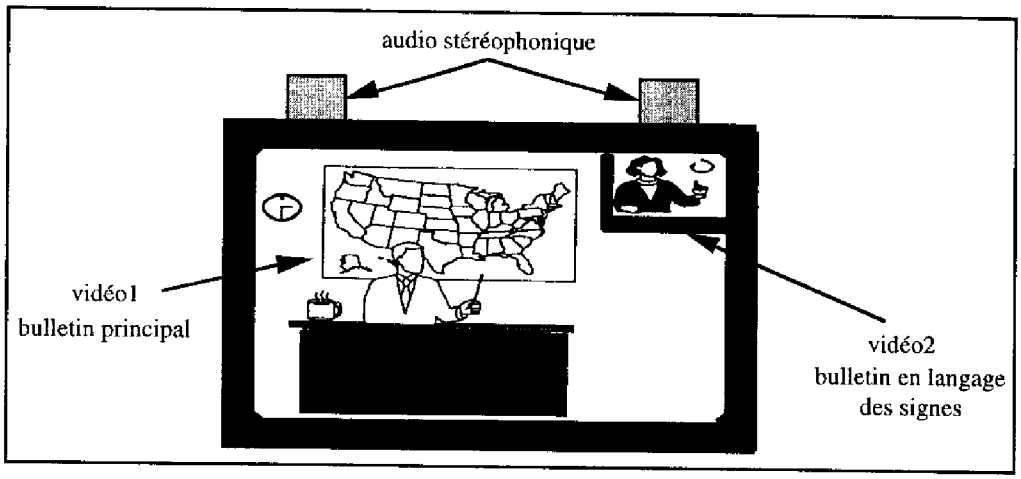


Figure (4.1) : Un exemple d'application multimédia

Cette application assure la distribution d'un bulletin météorologique sur deux fenêtres vidéo d'une station de travail distante.
Sur la plus grande de ces fenêtres sont présentées les images relatives au journaliste principal, dont les propos sont délivrés par l'intermédiaire de deux sorties audio. La plus petite des fenêtres, en haut à droite de l'écran, fait apparaître les images d'un second journaliste qui traduit les propos du premier en langage des signes. Trois flux de données sont donc véhiculés dans le cadre de cette application : deux flux vidéo et un flux audio stéréophonique. Les caractéristiques de présentation de ces différents flux sont les suivantes :

- le flux relatif à la séquence vidéo centrale est composé de 30 images par seconde ;
- l'incrustation muette est animée à raison de 10 images par seconde ;
- le flux audio est composé de 60 fragments de son par seconde.

Pour assurer le transport de cette application, l'architecture de Transport multimédia envisagée sur la figure (4.2) est ainsi composée de trois connexions de Transport monomédia, fournissant respectivement une qualité de service répondant aux caractéristiques du flux transporté : audio, vidéo 1 ou vidéo 2.

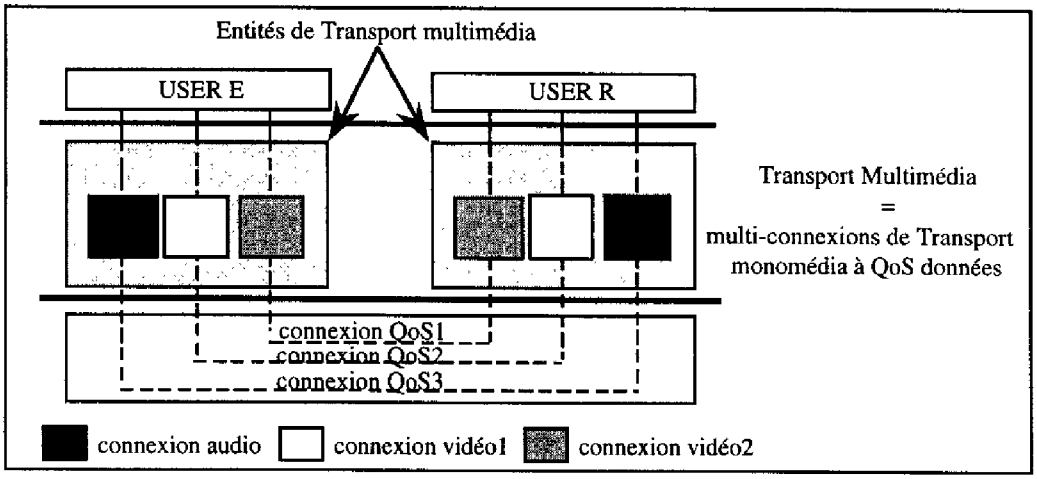


Figure (4.2) : Une architecture Transport multimédia

Soulignons ici que cette approche est partagée par [Camp94a] dans la conception architecturale de *QoS-A*, en ce sens qu'elle implique l'ouverture d'une connexion de Transport à QoS donnée par flux utilisateur.
Cependant, la prise en compte des contraintes de synchronisation des flux entre eux peut conduire à différentes architectures.

Par exemple, dans l'architecture *QoS-A*, la gestion de ces contraintes est effectuée au niveau d'une sous-couche applicative, *orchestration service*, utilisatrice du service de Transport ; la prise en compte d'une quelconque forme de ces contraintes n'est pas envisagée au niveau Transport ; en revanche, cette approche suppose que le service de Transport maintienne *flux par flux* l'ensemble des paramètres de performance nécessaires à l'utilisateur pour mettre en œuvre ses algorithmes.

Dans l'architecture de communication envisagée dans le projet CESAME, les contraintes de synchronisation sont représentées par un TSPN, qui par son formalisme fondamental (un réseau de Petri), permet de distinguer ces contraintes en deux types, chronologiques, telles qu'elles sont exprimées par le réseau de Petri, et temporelles par l'étiquetage des places. Les contraintes temporelles sont gérées à un niveau conceptuel supérieur au niveau Transport ; nous proposons de gérer les contraintes de synchronisation logiques dans le cadre de l'architecture de Transport précédemment introduite. Suivant cette approche, nous privilégions la garantie de l'ordre partiel que représente le réseau de Petri au respect des exigences de performances requises en termes de délai de transit et de gigue.

A la lumière de ce choix, nous détaillons l'architecture de Transport multimédia proposée en conséquence.

II. Principes de conception d'une connexion de Transport multimédia d'ordre partiel

Le chapitre 3 a montré qu'une connexion d'ordre partiel étendait la notion d'ordre des protocoles traditionnels ; dans cette section, nous montrons comment l'intégration du concept de POC partiellement fiable dans une architecture de Transport multimédia étend les fonctionnalités de la couche Transport, et permet de définir une QoS multimédia, adaptée aux besoins des utilisateurs.

Considérons pour cela le réseau de Petri de la figure (4.3), déduit d'une modélisation TSPN de l'application décrite en section (I). Supposons que chaque place représente une SDU (les places blanches et les places grises représentent par exemple des images entières). Ce réseau fournit une représentation logique des contraintes de synchronisation d'une période donnée de SDUs que nous qualifions par la suite d'ordre partiel multimédia.

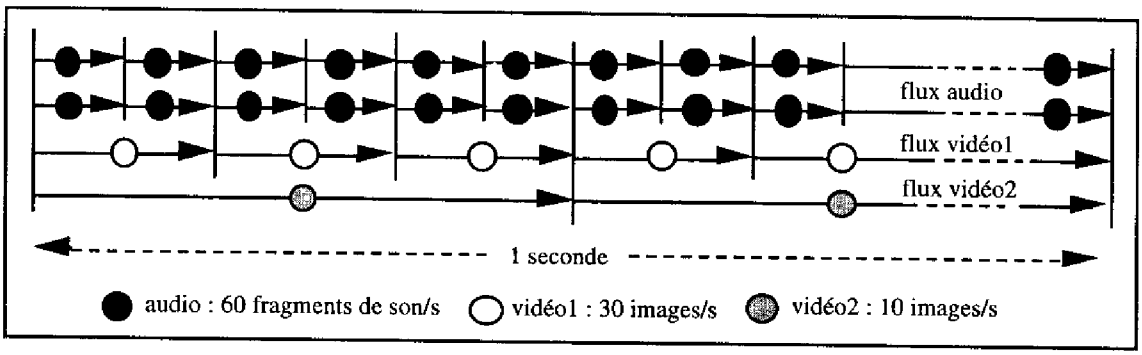


Figure (4.3) : Ordre partiel multimédia associé à l'exemple de la figure (4.1)

Étudions à présent la prise en compte d'un ordre partiel multimédia dans le cadre d'une architecture de Transport telle que celle introduite en (I) sans tenir compte, dans un premier temps, de la fiabilité de la connexion (II.1). En (II.2), nous poursuivons le raisonnement en tenant compte de l'aspect fiabilité.

II.1. Intégration de l'ordre partiel

Considérons l'ordre partiel *P* illustré Figure (4.4), représentant une période constituée des seize premiers objets de l'ordre partiel multimédia présenté Figure (4.3).

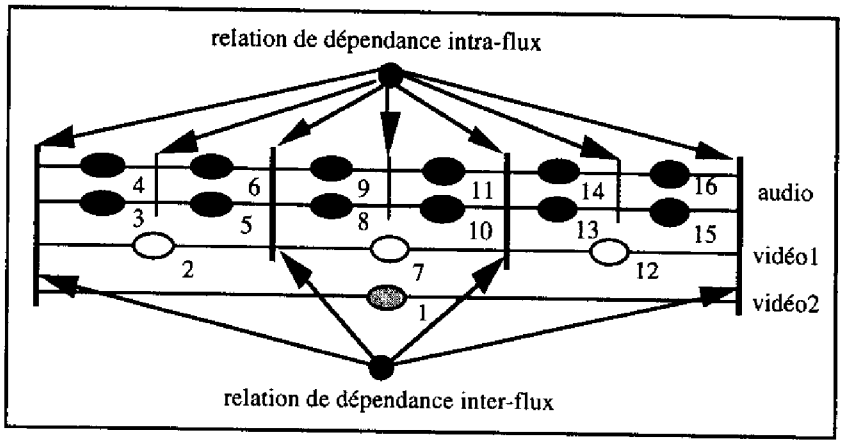


Figure (4.4) : Relations de dépendance intra et inter-flux

Il apparaît dans *P* deux types de relations de dépendance :

- les relations *intra-flux*, telles que *l'objet 4 précède les objets 5 et 6 dans l'ordre partiel monomédia du flux audio ;*
- les relations *inter-flux*, telles que *l'objet 2 précède les objets 8 et 9 dans l'ordre partiel multimédia.*

La prise en compte de *P* dans le cadre d'une architecture de Transport multimédia garantit la délivrance des SDUs selon l'une des extensions linéaires de *P*. En conséquence, l'intégration du concept de POC est perceptible à deux niveaux conceptuels, *monomédia* et *multimédia*, respectivement liés à la gestion des relations de dépendance intra et inter-flux de l'ordre partiel multimédia de la connexion. Considérons tout d'abord l'intégration monomédia du concept de POC.

II.1.1. Intégration monomédia

L'intégration monomédia du concept de POC consiste en la prise en compte des relations de dépendances intra-flux dans chacune des connexions monomédia ; en d'autres termes, les différentes connexions monomédia sont autant de connexions d'ordre partiel (Figure 4.5).

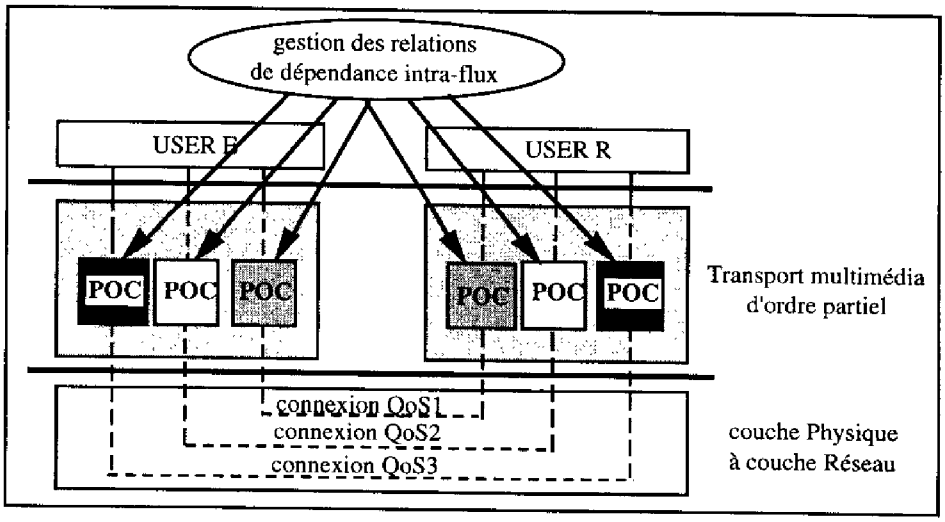


Figure (4.5) : Intégration monomédia du concept de POC

La gestion des relations de dépendance intra-flux audio de la figure (4.4) garantit par exemple la délivrance des SDUs 3 et 4 avant celle de 5 et 6, indépendamment de la délivrance des SDUs 1, 2, 7 et 12, relatives aux flux vidéo1 et vidéo2.
Remarquons que dans cet exemple, les connexions d'ordre partiels supports des flux vidéo1 et vidéo2 sont à ordre total. En revanche, l'ordre de la POC support du flux audio est partiel.

L'utilisateur peut ainsi sélectionner, flux par flux et pour chacun de ses flux, tous les services d'ordre partiel depuis l'ordre total jusqu'au "désordre". Au regard des contraintes temporelles que présente une application multimédia, l'intérêt de cette architecture réside dans la diminution du délai de transit global des données véhiculées, comparativement à une architecture similaire mais où toutes les connexions monomédia seraient à ordre total. Les bénéfices induits sur chaque POC en terme de bande passante ou de mémoire ont été discutés au chapitre 3.

II.1.2. Intégration multimédia

Dans l'architecture précédente, on constate une totale indépendance entre les POCs composant la connexion multimédia : deux SDUs échangées par le biais de deux POCs distinctes sont en effet indépendantes l'une de l'autre. En conséquence, il apparaît qu'une connexion de Transport multimédia n'intégrant le concept de POC qu'au sein des connexions monomédia instanciées ne tient pas compte des relations de dépendance inter-flux.

Considérons à nouveau la figure (4.4). L'intégration multimédia du concept de POC a pour but de garantir la délivrance des SDUs en respectant les relations de dépendances

inter-flux. Par exemple, la délivrance des SDUs audio 8 et 9 n'est concevable qu'après délivrance de la SDU 2, relative au flux vidéo1.
Contrairement aux relations intra-flux gérées de façon indépendante au travers des différentes POCs monomédia, la gestion des relations de dépendance inter-flux nécessite une synchronisation logique des POCs entre elles (Figure 4.6).

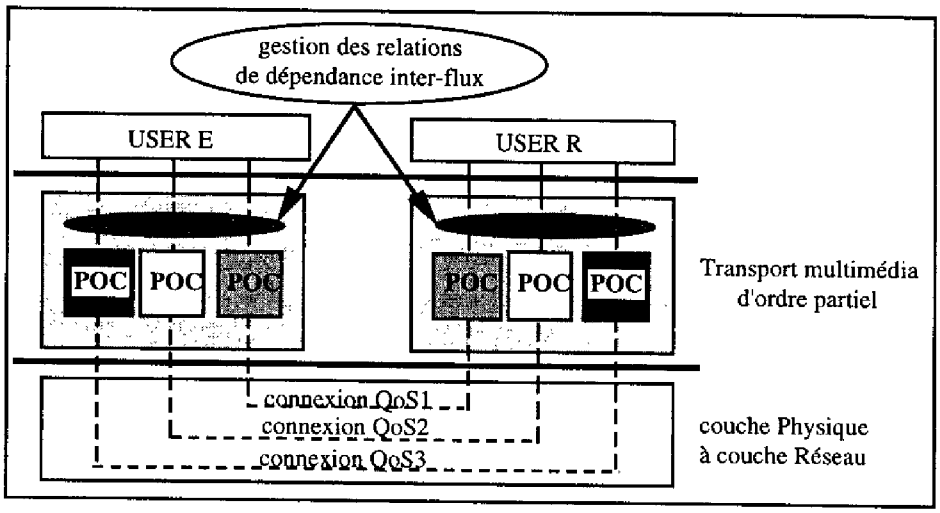


Figure (4.6) : Intégration multimédia du concept de POC

L'architecture de la figure (4.6) met donc en évidence les deux niveaux de gestion de l'ordre partiel tels qu'ils ont été présentés ci-dessus et dans le paragraphe précédent.

II.1.3. Conclusion

L'intégration à la fois mono et multimédia du concept de POC dans le cadre d'une connexion de Transport multimédia permet de définir un nouveau service de Transport, multimédia, répondant aux impératifs de chronologie induits par la synchronisation temporelle des données multimédia. L'expression de ces contraintes en tant que paramètre de service est directement déduite du modèle TSPN manipulé par l'utilisateur. Une connexion de Transport multimédia d'ordre partiel (MM-POC - MultiMedia Partial Order Connection) présente donc une extension de la notion de QoS, associée à la gestion d'un ordre logique (total ou partiel) inter-connexion. En cela, un service de synchronisation logique d'ordres partiels, garantissant les contraintes d'ordonnancement d'un objet multimédia dans son ensemble, offre donc à son utilisateur une hypothèse nouvelle pour résoudre le problème de la synchronisation.

Cette architecture permet de prendre en compte la première des caractéristiques d'un flux multimédia (l'ordre partiel) ; nous étudions à présent les conséquences d'une autre caractéristique : la différence de sensibilité aux pertes et aux erreurs des média d'un même flux multimédia.

II.2. Intégration de la fiabilité partielle

Un transfert fiable de bout en bout au travers d'un médium imparfait implique la retransmission de toutes les données perdues et donc un accroissement de leur délai de transit de bout en bout ; cette conséquence peut être inacceptable dans le cas de certaines applications en temps réel telles que la visioconférence : en d'autres termes, la satisfaction des contraintes temporelles exige éventuellement l'acceptation d'une dégradation de la fiabilité. Cependant, ces mêmes applications peuvent parfois se satisfaire d'un transfert imparfait : la perte d'une image dans une séquence vidéo n'est par exemple pas perceptible par l'oeil humain.

Partant de ce double constat, nous montrons dans cette section comment l'intégration de la notion de fiabilité partielle (chapitre 3, section II.2.2) dans une MM-POC permet de mettre en ouvre une politique de délivrance "au plus tôt" des SDUs au prix d'une dégradation acceptable de l'intégrité de la transmission (c'est-à-dire compatible avec la QoS "fiabilité" requise par l'utilisateur), tout en garantissant l'ordre partiel déduit d'une modélisation TSPN des contraintes de synchronisation du document multimédia véhiculé. [Conr95] envisage également l'exploitation d'une dégradation acceptable d'un document multimédia, mais dans le cadre d'une connexion de Transport monomédia.

La stratégie que nous proposons repose sur la possibilité de ne plus tenir compte des SDUs précédant la dernière SDU reçue, qu'elles aient été perdues, retardées, ou altérées : nous dirons que de telles SDUs sont déclarées perdues. De plus, cette politique contribue à diminuer l'utilisation des ressources.
Il s'agit donc d'examiner les conséquences de la délivrance d'une SDU en terme de dégradation de la fiabilité ; de façon à respecter l'ordre partiel multimédia, la règle suivante s'impose alors :

"la délivrance d'une SDU rend obsolète toutes les SDUs non encore délivrées ni déclarées perdues, la précédant dans l'ordre partiel multimédia".

Exemple d'application de cette règle

Considérons l'ordre partiel multimédia de la figure (4.7), paramètre de QoS "ordre" d'une connexion multimédia d'ordre partiel.
Supposons que la SDU 3 ait été déclarée perdue dans le cadre du protocole, et que les SDU 4 et 5 aient été délivrées à l'utilisateur.

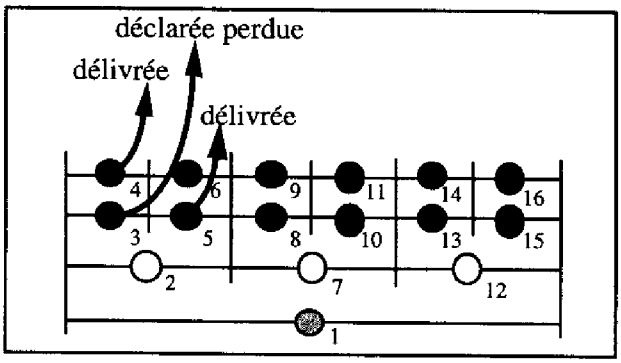


Figure (4.7) : QoS "ordre"

Supposons alors que la SDU 8 soit réassemblée par l'entité d'ordre partiel réceptrice.
Si celle-ci délivre la SDU 8, alors les SDUs 2 et 6 sont déclarées perdues dans le cadre du protocole, car elles précèdent 8 dans l'ordre partiel. Dans tous les cas, elles ne sont pas délivrées à l'utilisateur récepteur. Notons que la SDU 1, ne précédant pas 8 dans l'ordre partiel, n'est pas déclarée perdue par le protocole.

A toute délivrance est ainsi attaché un coût en terme de fiabilité. La stratégie "au plus tôt" consiste à délivrer immédiatement toute SDU dont le coût estimé de délivrance à l'utilisateur est inférieur au seuil de garantie de la QoS.
Afin de simplifier l'étude présentée, nous avons choisi de définir le paramètre *fiabilité* par le vecteur $k=(k_1, ..., k_n)$, où n désigne le nombre de POCs monomédia instanciées dans le cadre de la MM-POC[7]. La qualité de service est assurée lorsque le protocole n'engendre pas plus de k_i pertes de SDUs sur chacune des POC_i. Dans cette interprétation de la fiabilité, le coût est exprimé en nombre de pertes consécutives.

Nous proposons maintenant deux mécanismes de gestion de la fiabilité, basés sur les choix précédents, et correspondant à deux niveaux d'intégration différents : *connexion par connexion*, et *par groupe de connexions*. Nous analysons l'impact de chacun d'eux sur le délai de transit de la connexion multimédia.

II.2.1. Intégration de la fiabilité partielle "connexion par connexion"

Dans ce premier mécanisme, la délivrance d'une SDU sur une POC ne doit pas affecter la fiabilité de l'une quelconque des autres POCs. En d'autres termes, le coût de livraison d'une SDU devient infini si celle-ci rend obsolète une SDU relative à une autre POC.

[7] L'expression complète du paramètre *fiabilité* est fournie au paragraphe (III), dans le cadre de la définition du service MM-POS.

II.2.1.1. Principe

Dans le cadre d'une MM-POC, supposons que l'entité réceptrice ait à traiter la délivrance d'une SDU A de la connexion POC_i. Supposons en outre que A soit non délivrable au regard de l'ordre partiel. En d'autres termes, A est précédée dans l'ordre partiel, par une ou plusieurs SDUs, non encore délivrées ni déclarées perdues. Étudions alors les conditions qui autorisent le protocole à délivrer A.

Lorsqu'un contrôle de la fiabilité *connexion par connexion* est effectué, le protocole ne délivre A que si les deux conditions suivantes sont respectées :

- le nombre des SDUs relatives à POC_i rendues obsolètes par la délivrance de A, n'engendre pas une violation de la fiabilité partielle de POC_i ;
- la délivrance de A n'engendre pas la déclaration de perte de SDU(s) valide(s) sur l'une au moins des autres POCs.

Compte tenu de l'expression choisie de la fiabilité partielle, la délivrance de A est donc autorisée si le nombre de SDUs relatives à POC_i qui seraient alors rendues obsolètes, est inférieur ou égal à k_i, et si elle ne nécessite aucune déclaration de perte sur $POC_{j(j\neq i)}$, quelle que soit la valeur de k_j.

II.2.1.2. Exemple

Analysons les conséquences de ce contrôle d'erreur au travers du scénario suivant.

Considérons les paramètres ordre et fiabilité de la figure (4.8), tels qu'ils pourraient être spécifiés dans le cadre d'une connexion de Transport multimédia constituée de trois connexions d'ordre partiel notées POC●, POC○, POC◉ (Figure 4.9).

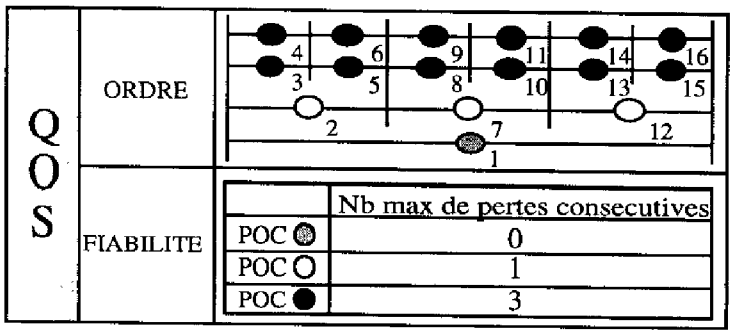


Figure (4.8) : Un exemple de paramètres de QoS *ordre* et *fiabilité*

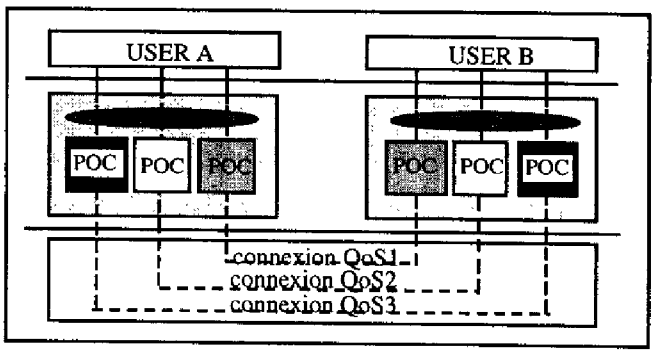


Figure (4.9) : Architecture de Transport multimédia

• Supposons tout d'abord que les SDUs 1, 2, 3, 4 et 5 aient été délivrées à l'utilisateur.

• Supposons alors que la SDU 7 soit réassemblée par l'entité MM-POC réceptrice.

La SDU 6 n'ayant pas été délivrée sur POC● et précédant la SDU 7 dans l'ordre partiel inter-flux, 7 n'est donc pas délivrable. La gestion d'une fiabilité partielle *connexion par connexion* n'autorise pas le protocole à déclarer perdue la SDU 6 sur POC●, même si cette perte n'engendre pas une violation de la QoS requise sur POC●. Par suite, la SDU 7 est soit stockée, soit rejetée. Dans le premier cas, des ressources mémoires sont utilisées ; dans le second, 7 devra être réémise, nécessitant une utilisation supplémentaire de bande passante. Supposons que 7 soit stockée dans les buffers de réception de POC○.

• Supposons alors que la SDU 10 soit réassemblée par l'entité MM-POC réceptrice.

Les SDUs 6, 8 et 9 n'ayant pas été délivrées sur POC● et précédant la SDU 10 dans l'ordre partiel monomédia de POC●, 10 n'est donc pas délivrable. Cependant, le paramètre fiabilité relatif à POC● autorise le protocole à déclarer perdues jusqu'à 3 SDUs consécutives ; en conséquence :

- les SDUs 6, 8 et 9 sont déclarées perdues ;
- la SDU 10, rendue ainsi délivrable au regard de l'ordre partiel intra-flux (et par ailleurs délivrable vis-à-vis de l'ordre partiel inter-flux) est délivrée au plus tôt à l'utilisateur, sans détérioration de la QoS requise sur POC● ;
- enfin, la SDU 7, rendue à présent délivrable au regard de l'ordre partiel inter-flux, est également délivrée au plus tôt sur POC○.

Ce scénario illustre donc comment un protocole de Transport multimédia tire partie d'une dégradation acceptable de la fiabilité pour délivrer les SDUs au plus tôt en respectant l'ordre partiel, au prix d'une déclaration de perte des SDUs retardées ou effectivement perdues par le réseau.

II.2.2. Intégration de la fiabilité partielle "par groupe de connexions"

Dans ce second mécanisme, la délivrance d'une SDU peut dégrader la fiabilité de l'ensemble des POCs dans les limites admissibles au regard de la QoS en terme de fiabilité. Cette option privilégie la diminution du délai de transit au détriment de l'indépendance des connexions.

II.2.2.1. Principe

Comme en (II.2.1.1), supposons que l'entité MM-POC réceptrice ait à traiter la délivrance d'une SDU A relative à POC_i, non délivrable au regard de l'ordre partiel.

Lorsqu'un contrôle de la fiabilité *par groupe de connexions* est mis en œuvre, le protocole d'ordre partiel délivre A sur POC_i si la condition suivante est respectée :

- pour j de 1 à n (n désignant le nombre de POCs), le nombre des SDUs relatives à POC_j déclarées perdues en conséquence n'engendre pas une violation de la fiabilité partielle de POC_j.

Compte tenu de l'expression de la fiabilité partielle, la délivrance de A est donc autorisée si le nombre de SDUs relatives à POC_j rendues obsolètes est inférieur ou égal à k_j, pour j variant de 1 à n.

II.2.2.2. Exemple

Considérons à nouveau les paramètres ordre et fiabilité de la figure (4.7). Le scénario suivant illustre la mise en œuvre du mécanisme de contrôle d'erreur par groupe de connexions.

- Supposons que les SDUs 1, 2 et 3 aient été délivrées à l'utilisateur.

- Supposons alors que la SDU 7 soit réassemblée par l'entité MM-POC réceptrice.

7 est délivrable au regard de l'ordre partiel monomédia de POC○ ; en revanche, les SDUs 4, 5 et 6 n'ayant été ni délivrées, ni déclarées perdues sur POC●, et précédant la SDU 7 dans l'ordre partiel multimédia, 7 n'est donc a priori pas délivrable.
Cependant, la mise en œuvre d'un contrôle d'erreur *par groupe de connexions* autorise le protocole à déclarer 4, 5 et 6 perdues, car ces pertes n'engendrent pas une dégradation de la QoS de POC● (3 pertes consécutives sont en effet acceptables). La SDU 7 devient alors délivrable au regard de l'ordre partiel inter-flux ; elle est ainsi délivrée au plus tôt à l'utilisateur, sans détérioration de la QoS sur les différentes POCs.

II.2.3. Conclusion

Dans cette section, nous avons défini une qualité de service multimédia relative à la fiabilité ; cette QoS permet d'envisager la mise en œuvre d'un protocole de Transport assurant de façon globale les différences de sensibilité aux erreurs d'un flux multimédia. L'introduction de la fiabilité dans une MM-POC peut être appliquée *connexion par connexion* ou *par groupe de connexions*. Dans tous les cas, elle engendre une diminution du délai de transit des données en utilisant une politique de délivrance avec garantie conjointe des contraintes d'ordre et de fiabilité. Les deux options envisagées sont de complexité équivalente, elles diffèrent par la priorité donnée aux performances (délai de transit) ou à l'indépendance entre les flux. Enfin, la simplicité des règles de mise en œuvre de ces mécanismes permet d'envisager une implantation robuste et performante du protocole.

Ayant présenté l'architecture d'une MM-POC et les caractéristiques de ses principaux mécanismes, nous précisons maintenant les paramètres de service ainsi que les primitives du service correspondant.

III. Définition du Service

Les études les plus récentes concernant la spécification de services de Transport ont conduit à la définition de nouvelles sémantiques de service, ainsi qu'à un enrichissement des paramètres de QoS [Dant92]. Dans cette section, nous présentons les paramètres de QoS ainsi que les primitives du service MM-POS (MM-POS - MultiMedia Partial Order Service). L'ordre et la fiabilité y apparaissent comme deux paramètres de QoS nouveaux. Nous précisons la sémantique de service de chacun des paramètres définis. Le concept de POC reposant sur la manipulation de messages (SDU), le mode du service que nous définissons ici est le mode message.

Tel que nous l'avons indiqué au chapitre 2, le choix de l'environnement d'implantation des logiciels développés dans le projet CESAME ne s'est pas porté sur un système dédié, mais sur un système d'exploitation asynchrone. Les conséquences de ce choix sont importantes ; [Owe95c] souligne les conséquences de cet asynchronisme notamment par une variabilité non majorée du temps de traitement des instructions : manipulée sans précaution, toute instruction relative à une gestion explicite du temps peut engendrer des incohérences, et il est nécessaire de ne pas multiplier les algorithmes à caractère temporel à différents niveaux du système de communication.
Cette spécification nous a conduit à ne pas introduire dans le service MM-POS toutes les garanties temporelles attendues d'un service de Transport multimédia tel que OSI 95. Néanmoins, deux paramètres à caractère temporel sont définis, le débit et le délai de

transit ; nous justifierons leur intérêt dans ce contexte, ainsi que la différence de sémantique de service qui leur est attribuée.

III.1. Paramètres de QoS : définition et sémantique de service

Considérons une connexion multimédia d'ordre partiel composée de n POCs monomédia, chacune à qualités de service données. Par la suite, POC_i désignera la $i^{ème}$ POC monomédia de la MM-POC.

III.1.1. Ordre

Expression

Dans la spécification Estelle du protocole POC présentée en (III.2.1) du chapitre 3, l'ordre partiel est représenté par une matrice triangulaire supérieure, à diagonale principale nulle. En conséquence, le paramètre de QoS "ordre" peut être codé par un vecteur de taille $N(N-1)/2$, N désignant la taille de l'ordre partiel. Un vecteur supplémentaire de taille N permet d'associer chacun des N objets de l'ordre partiel à l'une des n POCs monomédia.

Sémantique de service

Qu'il soit considéré sous sa forme inter-flux ou sous ses formes intra-flux, le paramètre ordre est toujours respecté. Ce choix résulte de la signification de l'ordre partiel au niveau de l'application, exprimant les contraintes minimales en terme d'ordre de délivrance des SDUs pour assurer l'intelligibilité du message. La violation de cette QoS apparaît donc incompatible avec le service attendu par l'utilisateur.

Négociation

L'ordre partiel requis est spécifié par l'utilisateur émetteur en tant que paramètre de la requête d'établissement d'une connexion MM-POC. Il est ensuite propagé à l'utilisateur récepteur en tant que paramètre de la primitive d'indication de connexion correspondante. Une négociation réelle entre utilisateur et fournisseur du service semble sans objet.

III.1.2. Fiabilité

Expression

Le paramètre fiabilité possède trois composantes, relatives à l'expression des pertes acceptables, au choix d'un mode opératoire, et à la sélection d'une sémantique de service.

1 - Inspirée des paramètres *Burst Loss Sensitivity*, *Loss Sensitivity* et *Loss Interval* de [Part92], la première composante est exprimée par la structure (k, w, l) suivante.

$$
\begin{aligned}
k &= (k_1, k_2, \ldots k_n) \\
w &= (w_1, w_2, \ldots, w_n) \\
l &= (l_1, l_2, \ldots, l_n)
\end{aligned}
$$

dans laquelle :

- k_i définit le nombre maximum de pertes consécutives de SDUs que l'utilisateur peut tolérer sur POC_i ;
- à tout moment, l_i SDUs sur les w_i dernières émises sont au plus autorisées à être perdues sur POC_i.

Ce choix se justifie par la simplicité des mécanismes à mettre en œuvre pour assurer cette qualité de service.

2 - Quatre modes opératoires sont définis pour chaque POC (Figure 4.10), afin que l'utilisateur puisse sélectionner s'il désire que lui soit délivrées les SDUs corrompues, et/ou notifiées les occurrences de perte acceptables, au regard de *(k, l, w)*.
Toute SDU non délivrable car corrompue est déclarée perdue par l'entité MM-POC réceptrice si cette perte est acceptable au regard du paramètre *(k, l, w)*.
Une occurrence de perte peut être notifiée à l'utilisateur par le nombre ou la liste des SDUs perdues.

		Délivrance des données corrompues	
		autorisée	non autorisée
Perte acceptable	non signalée	1	2
	signalée	3.x	4.x

avec { x = 0 quand les pertes sont signalées par leur nombre
x = 1 quand les pertes sont signalées par leur liste

Figure (4.10) : Modes opératoires

3 - La troisième composante du paramètre fiabilité est relative au choix de la sémantique de service parmi les deux définies ci-après.

Sémantique de service

Deux sémantiques de service sont proposées ; elles sont sélectionnables flux par flux par l'intermédiaire du vecteur de booléens $v = (v_1, \ldots, v_n)$.

1 - La première assure la fiabilité (i.e. la QoS fiabilité est toujours respectée), et de ce fait peut conduire à une augmentation non bornée du délai de transit. On remarquera que ce choix peut s'avérer inadapté compte tenu des contraintes temporelles des flux continus.

2 - Dans la deuxième sémantique, la fiabilité n'est plus assurée, et une notification de dégradation de la QoS peut être faite à l'utilisateur. Cette sémantique autorise la prise en compte de la validité temporelle des SDUs. Suivant notre démarche initiale, la détermination de la validité temporelle d'une SDU ne peut pas être effectuée par le biais d'une mesure effective du temps. Cependant, dans le cas de flux continus, il est possible d'affirmer l'obsolescence d'un objet (alors qu'il est impossible d'affirmer sa validité). En conséquence, les objets obsolètes ne sont pas délivrés, ce qui peut conduire à une violation de la fiabilité dans les termes définis précédemment.

En réponse à une violation de la fiabilité sur l'une des POCs monomédia (POC_i) :

- soit la connexion MM-POC est relâchée,
- soit la dégradation de QoS est notifiée à l'utilisateur ; dans ce dernier cas, le service MM-POS indique lequel des paramètres k_i et/ou l_i n'a pas été respecté.

Le paramètre de QoS fiabilité est résumé (Figure 4.11).

Fiabilité					
flux	mode opératoire	w	l	k	v
1					
2					
n					
Sémantique de service en cas de violation de QoS					
0		1			

avec { 0 = MM-POC relâchée ; 1 = dégradation notifiée }

Figure (4.11) : Paramètre de QoS fiabilité

Remarque

Compte tenu de la définition des classes de fiabilité NBART-L, BART-NL et BART-L(N) établie en (III.3) du chapitre 3, il est alors possible d'accorder :

- la première sémantique de service avec (l=w=1) et (k=∞) aux objets NBART-L ;
- la première sémantique de service avec (k=0) aux objets BART-NL ;
- la deuxième sémantique de service aux objets BART-L(N).

Négociation

La fiabilité requise sur chaque POC est spécifiée par l'utilisateur émetteur en tant que paramètre de la requête d'établissement d'une connexion MM-POC. Elle est ensuite propagée à l'utilisateur récepteur en tant que paramètre de la primitive d'indication de connexion correspondante.

III.1.3. Débit

Expression

Deux définitions du débit sont envisageables : en nombre de SDU, ou en quantité d'informations transférée par seconde. Nous choisissons cette dernière définition car c'est la plus communément adoptée. Le débit requis sur chaque POC est exprimé par l'intermédiaire de ses valeurs minimale D_{min} et maximale D_{max}. Lorsque le flux véhiculé est continu, D_{min} et D_{max} sont égales.

Sémantique de service

Le débit est le premier des deux paramètres de QoS à caractère temporel du service MM-POS.

La valeur maximale du débit correspond à la limite supérieure que l'utilisateur émetteur s'engage à ne pas excéder. En conséquence, la délivrance des SDUs à l'utilisateur récepteur est assurée à un débit au plus égal à D_{max}.

Concernant la sémantique de service du débit minimal, nous avons choisi d'en notifier toute dégradation à l'utilisateur pour les raisons suivantes :

- le contrôle du débit peut être effectué de façon plus ou moins stricte, SDU par SDU [Dant92], sur une plage de SDUs ou par période de temps. Dans ce dernier cas, l'observation du débit se traduit par une gestion "en moyenne" des contraintes temporelles, a priori encline à tolérer les phénomènes d'asynchronisme du système d'exploitation.
- en outre, le débit peut être contrôlé de façon indépendante en émission ou en réception : une observation du débit est donc envisageable dans le cadre d'un mécanisme qui n'accentuera pas outre mesure la complexité du protocole.
- enfin, les conséquences pour l'utilisateur d'une violation du débit minimal nous semble préjudiciable, et il est préférable que l'utilisateur puisse entamer au plus tôt une phase de renégociation de la QoS.

Selon le souhait de l'utilisateur, et en réponse à une violation du débit minimal sur l'une des POCs monomédia (POC_i), soit la connexion MM-POC est relâchée dans son ensemble, soit cette dégradation est notifiée à l'utilisateur par l'intermédiaire de sa valeur réelle.
Il est exclu de ne relâcher que la connexion POC_i du fait des incohérences que cela risquerait d'engendrer vis-à-vis de la gestion de l'ordre partiel.

Remarque importante

Soulignons qu'en terme de débit, une perte acceptable équivaut à une délivrance. En d'autres termes, toute déclaration de perte acceptable liée à l'application de la stratégie de délivrance au plus tôt, devra être traduite vis-à-vis du débit comme une délivrance.

Le paramètre de QoS débit est résumé (Figure 4.12)

Débit		
flux	min	max
1		
2		
n		
Sémantique de service en cas de violation de QoS		
0	1	

avec { 0 = MM-POC relâchée ; 1 = dégradation notifiée }

Figure (4.12) : Paramètre de QoS débit

Négociation

Le service MM-POS avorte la phase d'établissement de connexion s'il juge la valeur de D_{min} requise par l'utilisateur trop élevée. Cette décision dépend de la QoS fournie par le service de Réseau sous-jacent.

III.1.4. Délai de transit

Le délai de transit est le second et dernier paramètre de QoS à caractère temporel.

Expression

Le délai de transit est défini comme la durée écoulée entre la date de soumission d'une SDU par l'utilisateur émetteur, et la date de délivrance de cette SDU à l'utilisateur récepteur.
Le délai de transit requis sur chaque POC est défini par l'intermédiaire de ses valeurs minimale DT_{min}, et maximale DT_{max}.

Sémantique de service

La sémantique de service concernant la valeur DT_{max} est le *best effort*. Aucune notification de dégradation n'est donc fournie à l'utilisateur. Nous justifions ce choix pour les raisons suivantes :

- contrairement au débit minimal, le contrôle du délai de transit n'offre de garantie que s'il est effectué SDU par SDU ; la gestion "en moyenne" des contraintes temporelles évoquée pour le débit est donc ici impossible : compte tenu de l'asynchronisme du système, un tel contrôle ne présente aucune garantie ;
- en outre, le contrôle du délai de transit nécessite que les SDUs soient estampillées en émission et qu'une évaluation de leur date d'arrivée soit effectuée en réception : ce mécanisme impose que les horloges d'émission et de réception soient synchronisées ;
- enfin et indépendamment des garanties que présenterait un tel contrôle, nous préférons minimiser la charge du protocole de Transport, afin d'utiliser pleinement les potentialités de la stratégie de délivrance au plus tôt.

La sémantique de service vis-à-vis du DT_{max} étant le *best effort*, l'intérêt de cette valeur dépend de sa prise en compte dans le cadre du protocole. Par exemple, s'il l'on suppose que la sélection par l'utilisateur d'une QoS fiabilité non assurée (seconde sémantique de service de la section III.1.2) est implicitement liée à un flux continu, c'est à partir des valeurs du DT_{max}, du débit et de la connaissance de la valeur minimale du RTT que le protocole de Transport peut déterminer l'obsolescence d'un objet, sans gestion explicite du temps.

Contrairement au délai de transit maximal, il est possible d'assurer le respect d'une valeur minimale (strictement positive) du délai de transit sans contrôle de la part du pourvoyeur du service ; une condition suffisante pour garantir cette QoS est que la valeur de DT_{min} soit au moins égale au demi temps aller-retour des PDUs. Cette QoS peut être réclamée par l'utilisateur pour contrôler le retard de restitution des flux multimédia. Les détails d'un algorithme défini dans ce but sur les bases d'une garantie de délai de transit minimal, sont données dans [Owe95c].

Négociation

Le délai de transit minimal requis sur chaque POC n'est pas initialement spécifié par l'utilisateur émetteur : sa valeur est retournée aux utilisateurs, une fois qu'elle a été évaluée dans le cadre du protocole de Transport.
Le service MM-POS rejette la demande d'ouverture de connexion si la valeur de DT_{min} s'avère (après évaluation) supérieure au DT_{max} spécifié par l'utilisateur.

III.1.5. Taille des SDUs

La connaissance de la taille maximale des SDUs de chaque POC peut être utile au fournisseur du service pour dimensionner ses ressources [Hehm90] ; en outre, elle permet de déterminer si un mécanisme de segmentation et de réassemblage doit être mis en œuvre.

Le paramètre correspondant est spécifié en tant que paramètre des primitives de requête et d'indication invoquées lors de l'ouverture d'une connexion MM-POC. Sa négociation entre utilisateur et fournisseur du service est envisageable.

III.2. Primitives de service

Dans ce paragraphe, nous exposons les primitives du service MM-POS, ainsi que leur enchaînement. Notons que ces primitives ne sont pas spécifiques à l'une ou l'autre des POCs monomédia, mais à la connexion MM-POC dans son ensemble.

III.2.1. Primitives d'établissement de connexion

L'établissement d'une connexion MM-POC requiert l'invocation de quatre primitives de service : MM-PO_CONNECT.Req, MM-PO_CONNECT.Ind, MM-PO_CONNECT.Resp et MM-PO_CONNECT.Conf (Figure 4.13).

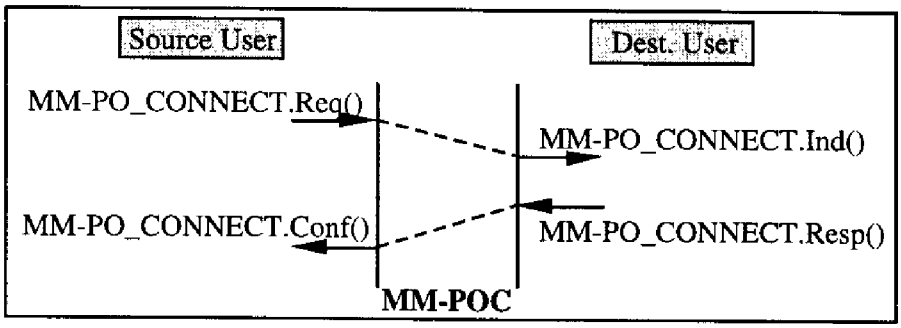


Figure (4.13) : Primitives d'établissement de connexion

Paramétrage des primitives de service

MM-PO_CONNECT.Req :

- adresse destination, adresse source
- nombre de flux (n)
- ordre partiel intra-flux pour chacun des n flux
- ordre partiel inter-flux
- (k_i, l_i, w_i, v_i) pour chacun des n flux
- mode opératoire de chacun des n flux
- sémantique de service en cas de violation de la fiabilité
- D_{min} et D_{max} pour chacun des n flux,
- sémantique de service en cas de violation du débit minimal
- DT_{max} pour chacune des n POCs
- taille maximale des SDUs de chacune des n POCs

MM-PO_CONNECT.Ind :

- adresse destination, adresse source
- nombre de flux (n)
- *cep_0, cep_1, cep_2, ..., cep_n*
- ordre partiel intra-flux pour chacun des n flux
- ordre partiel inter-flux

- (k_i, l_i, w_i, v_i) pour chacun des n flux
- mode opératoire de chacun des n flux
- sémantique de service en cas de violation de la fiabilité
- D_{min} et D_{max} pour chacun des n flux
- sémantique de service en cas de violation du débit minimal
- DT_{max} pour chacune des n POCs
- DT_{min} pour chacune des n POCs
- taille maximale des SDUs de chacune des n POCs

où :

- *cep_0* définit l'identificateur de la MM-POC en réception
- *cep_i* (pour i de 1 à n) définit l'identificateur de la connexion POC_i en réception

MM-PO_CONNECT.Resp :

- statut = acceptation ou rejet

La raison d'un rejet est supposée codée dans le champ statut.

MM-PO_CONNECT.Conf :

- statut = acceptation ou rejet
- *cep_0, cep_1, cep_2, ..., cep_n*
- DT_{min} pour chacune des n POCs

En cas d'acceptation :

- *cep_0* définit l'identificateur de la MM-POC en émission
- *cep_i* (pour i de 1 à n) définit l'identificateur de la connexion POC_i en émission.

III.2.2. Primitives de transfert de données

Le transfert des données requiert l'invocation de deux primitives de service : MM-PO_DATA.Req et MM-PO_DATA.Ind (Figure 4.14).

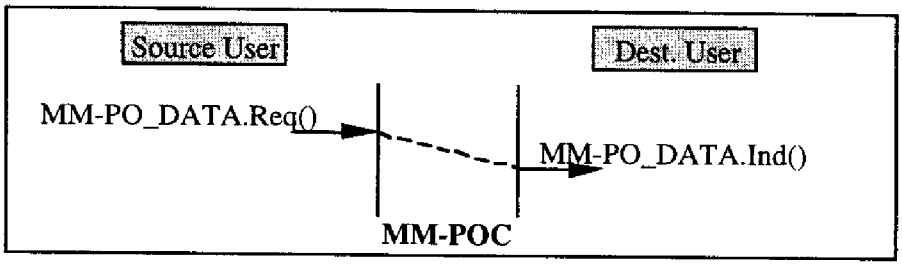


Figure (4.14) : Primitives de transfert de données

Il a été montré en (II) que la délivrance d'une SDU sur une POC donnée (POC_i) pouvait être effectuée au prix de la déclaration de perte de SDU(s) sur POC_i et/ou sur $POC_{j(j\neq i)}$; ces déclarations résultent de la stratégie de délivrance *au plus tôt*, et d'une gestion de la fiabilité *connexion par connexion* ou *par groupe de connexions*.

Ce type de délivrance peut être qualifié de :

- *délivrance avec perte acceptable* si la QoS requise en terme de fiabilité est respectée ; ces pertes sont notifiées à l'utilisateur si la classe de fiabilité sélectionnée le requiert ;

- *délivrance avec perte inacceptable* si l'occurrence de ces pertes engendre une violation de la QoS requise en terme de fiabilité ; cette possibilité dépend de la sémantique du service de fiabilité sélectionnée par l'utilisateur ; selon le mode opératoire choisi, la connexion MM-POC peut être relâchée ou la dégradation de QoS notifiée à l'utilisateur.

Paramétrage des primitives de service

MM-PO_DATA.Req : - *cep_0, cep_i*
- donnée soumise en émission
- longueur de la donnée

où :
- *cep_0* définit l'identificateur de la MM-POC en émission
- *cep_i* définit l'identificateur de la POC_i à laquelle est relative la donnée soumise en émission

MM-PO_DATA.Ind : - *cep_0, cep_i*
- donnée délivrée,
- longueur de la donnée
- *perte_acceptable*

où :
- *cep_0* définit l'identificateur de la MM-POC en réception
- *cep_i* définit l'identificateur de la POC (POC_i) à laquelle est relative la donnée délivrée
- *perte_acceptable* indique à l'utilisateur du service si la délivrance de la donnée a été effectuée au prix de perte(s) acceptables sur POC_i, sa définition précise est fournie ci-après.

Notification des pertes acceptables engendrées sur POC_i

Le paramètre *perte_acceptable* est composé de deux segments : un code et un message (Figure 4.15).

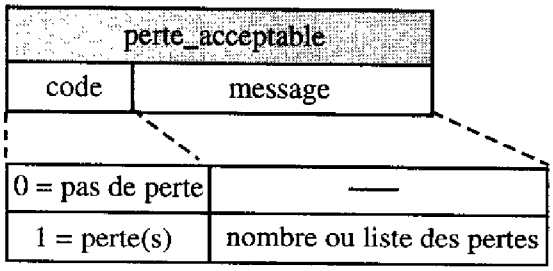


Figure (4.15) : Champ *perte_acceptable*

***code* = 0** dans l'un ou l'autre des cas suivants :
- le service n'a pas à notifier les pertes acceptables, i.e. lorsque la classe de fiabilité sélectionnée est égale à **1** ou **2** (paragraphe III.1.2) ;
- la délivrance de la donnée n'engendre aucune déclaration de perte sur POC_i .

Le champ ***message*** n'est alors pas significatif.

***code* = 1** lorsque le service doit notifier les occurrences de pertes acceptables, i.e. lorsque la classe de fiabilité sélectionnée égale 3.x ou 4.x, et que la délivrance de la donnée est effectuée au prix de perte(s) acceptable(s).

Le champ ***message*** présente alors une valeur signifiant :

- soit le nombre des pertes engendrées sur POC_i si la classe de fiabilité égale 3.1 ou 4.1;
- soit la liste de ces pertes si la classe de fiabilité égale 3.2 ou 4.2.

Les pertes engendrées sur $POC_{j(j\neq i)}$, sont notifiées à l'utilisateur par le biais d'une primitive d'indication de perte spécifique, MM-PO_LOSS.ind, définie ci-dessous (Figure 4.16).

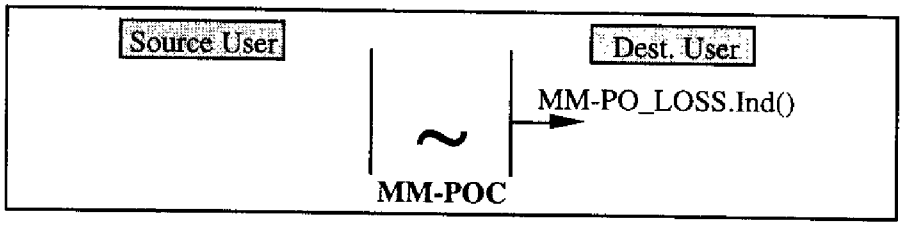


Figure (4.16) : Primitive d'indication de perte

Paramétrage de la primitive MM-PO_LOSS

MM-PO_LOSS.ind : - *cep_0*, *cep_j*
- perte_acceptable

où :

- *cep_0* définit l'identificateur de la MM-POC en réception
- *cep_j* définit l'identificateur de la POC_j à laquelle sont relatives les pertes notifiées
- *perte_acceptable* indique le nombre ou la liste des pertes engendrées sur POC_j.

Cette primitive n'est pas invoquée par le service si les pertes engendrées ont déjà été notifiées à l'utilisateur par le biais de la primitive MM-PO_DATA.Ind.

III.2.3. Primitives de notification de dégradation de la QoS

En cas de violation de la QoS débit ou fiabilité, le service MM-POS doit rompre la connexion MM-POC, ou notifier la dégradation à l'utilisateur. La sélection de l'une ou l'autre de ces possibilités est effectuée par l'utilisateur émetteur lors de l'ouverture de la connexion.

Pour notifier une dégradation de la QoS, le service MM-POS dispose de la primitive d'indication MM-PO_QoS-FAILED.Ind (Figure 4.17).

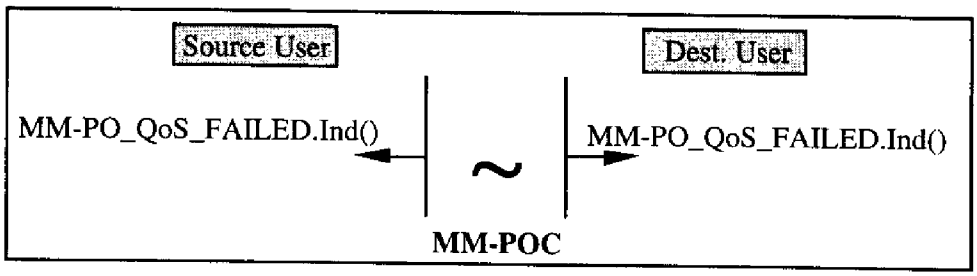


Figure (4.17) : Primitive d'indication de dégradation

Paramétrage de la primitive de service

MM-PO_QoS-FAILED.Ind : - *cep_0, cep_i*
- *dégradation_QoS*

où :

- *cep_0* définit l'identificateur de la MM-POC en réception
- *cep_i* définit l'identificateur de la POC dégradée
- le paramètre de notification de dégradation de la QoS, *dégradation_QoS*, est composé de deux segments : un code et un diagnostic (Figure 4.18)).

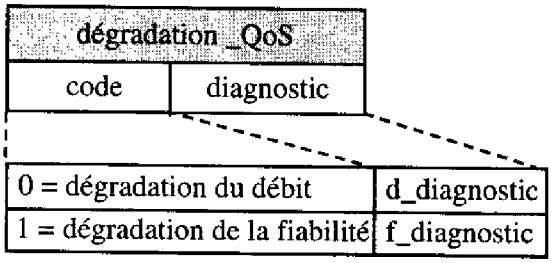


Figure (4.18) : Champs *dégradation_QoS*

code* = 0** lorsqu'une dégradation du débit minimal est observée sur l'une des *n* POCs ; le ***diagnostic correspondant indique à l'utilisateur la valeur réelle du débit (*d_diagnostic*).

code* = 1** lorsqu'une dégradation de la fiabilité est observée sur l'une des *n* POCs ; le ***diagnostic correspondant indique à l'utilisateur le nombre de pertes et/ou le nombre de pertes consécutives observées (*f_diagnostic*).

III.2.4. Primitives de renégociation de la QoS

Quatre primitives supplémentaires sont ici introduites afin que l'utilisateur puisse mettre en œuvre une phase de renégociation de la QoS, sans avoir à fermer la connexion MM-POC (Figure 4.19). Tous les paramètres de QoS initialement spécifiés peuvent être redéfinis, excepté le nombre de flux ; cette restriction permet de conserver les POCs instanciées.

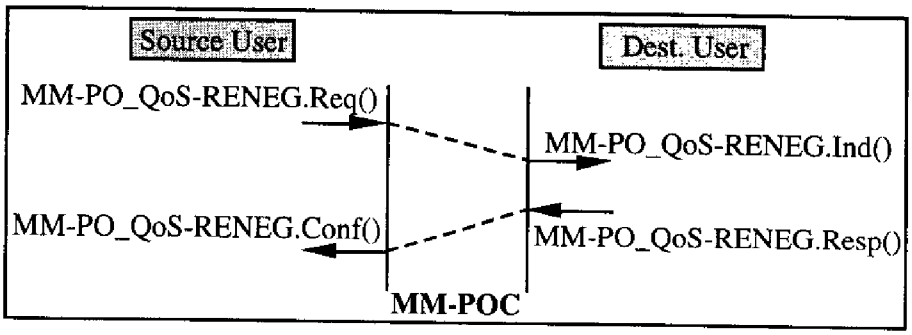


Figure (4.19) : Primitives de renégociation de QoS

Paramétrage des primitives de service

MM-PO_QoS-RENEG.Req :

- ordre partiel intra-flux pour chacun des n flux
- ordre partiel inter-flux
- (k_i, l_i, w_i, v_i) pour chacun des n flux
- mode opératoire de chacun des n flux
- sémantique de service en cas de violation de la fiabilité
- D_{min} et D_{max} pour chacun des n flux
- sémantique de service en cas de violation du débit minimal
- DT_{max} pour chacune des n POCs
- taille maximale des SDUs pour chacune des n POCs

MM-PO_QoS-RENEG.Ind :

- ordre partiel intra-flux pour chacun des n flux
- ordre partiel inter-flux
- (k_i, l_i, w_i, v_i) pour chacun des n flux
- mode opératoire de chacun des n flux
- sémantique de service en cas de violation de la fiabilité
- D_{min} et D_{max} pour chacun des n flux
- sémantique de service en cas de violation du débit minimal
- DT_{max} pour chacune des n POCs
- DT_{min} pour chacune des n POCs
- taille maximale des SDUs pour chacune des n POCs

MM-PO_QoS-RENEG.Resp : - statut = acceptation ou rejet

MM-PO_QoS-RENEG.Conf : - statut = acceptation ou rejet
- DT_{min} pour chacune des n POCs

III.2.5. Primitives de fermeture de connexion forcée

Seul le mode de fermeture de connexion forcé est ici représenté. La fermeture d'une connexion MM-POC requiert l'invocation de deux primitives : MM-PO_DISCONNECT.Req et MM-PO_DISCONNECT.Ind (Figure 4.20).

Un diagnostic est fourni lors de la fermeture de la connexion, si celle-ci fait suite à une dégradation de la QoS.

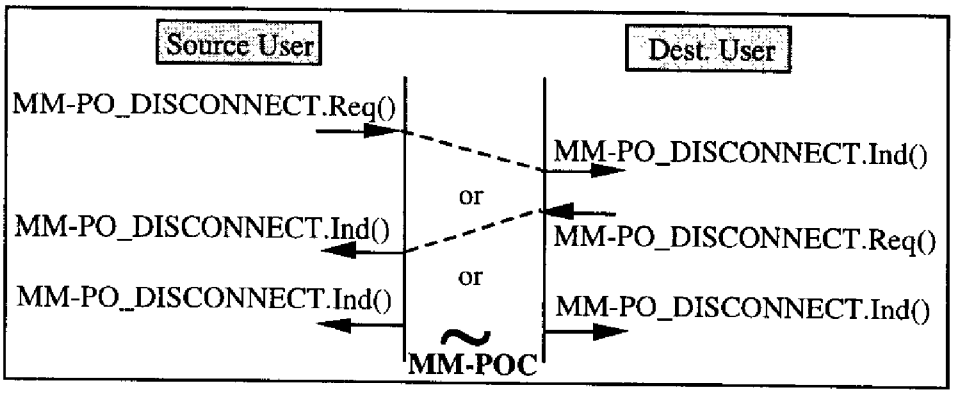


Figure (4.20) : Primitives de fermeture de connexion

Paramétrage des primitives de service

MM-PO_DISCONNECT.Req : - *cep_0*

où : - *cep_0* définit l'identificateur de la connexion MM-POC.

MM-PO_DISCONNECT.Ind : - *cep_0*
- *diagnostic*

où :

- *cep_0* définit l'identificateur de la connexion MM-POC
- le champ *diagnostic* indique si la fermeture de connexion fait suite à une requête de l'utilisateur pair ou une dégradation de la QoS (Figure 4.21) ; ce champ est composé de deux segments, un code et un message.

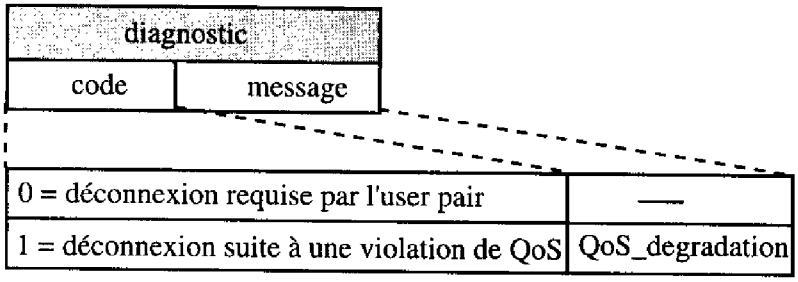


Figure (4.21) : Champs diagnostic

code* = 0** si la fermeture de connexion a été requise par l'utilisateur pair ; le champ ***message correspondant est alors non significatif.

code* = 1** si la fermeture de connexion fait suite à une dégradation de la QoS ; le champ ***message fournit une notification de dégradation de la QoS analogue à celle présentée au paragraphe précédent.

Conclusion

Concernant la conception d'architectures de communication multimédia, nous avons rappelé les deux approches envisagées à l'heure actuelle, et en particulier celle consistant à répartir la gestion des exigences applicatives à différents niveaux de l'architecture.
Au vu de la complexité et du peu d'utilité que représenterait une gestion de la synchronisation temporelle au niveau Transport, le compromis envisagé dans le second cas consiste à déléguer au niveau Transport la charge d'un certain nombre de paramètres (délai de transit et de gigue notamment) exprimés *flux par flux*. La prise en compte globale des différents composants d'un même flux multimédia est envisagée à un niveau conceptuel supérieur au niveau Transport, et repose sur l'hypothèse des garanties temporelles offertes par le service de Transport.

L'architecture de Transport que nous avons définie dans ce chapitre repose sur une analyse différente de la notion de flux multimédia. Tout d'abord, elle implique l'instanciation puis la coordination de plusieurs connexions monomédia dans le cadre d'un même contexte multimédia. En intégrant le concept de POC non seulement dans chaque connexion mais également entre les connexions, une telle architecture permet de prendre en compte au niveau du protocole de Transport l'une des caractéristiques intrinsèques d'un flux multimédia : les relations d'ordonnancement logiques entre chacun des flux monomédia le constituant. En cela, cette architecture permet de définir un service de Transport *multimédia* liant application et protocole ; de plus, elle fournit à l'utilisateur du service une hypothèse nouvelle vis-à-vis de la gestion de la synchronisation, hypothèse dont la garantie repose sur la spécification de mécanismes peu complexes. Enfin, cette architecture permet aussi de prendre en compte de façon globale les différents degrés de sensibilité aux pertes d'un flux multimédia, en en tirant partie pour diminuer le délai de transit des données.

CHAPITRE 5 -

MISE EN ŒUVRE D'UNE PROTOCOLE MULTIMÉDIA D'ORDRE PARTIEL A L'AIDE DE XTP

Introduction

Les chapitres 3 et 4 ont mis en évidence les lacunes conceptuelles des protocoles de Transport traditionnels, et en ont proposé une extension en terme d'ordre et de fiabilité. Au préalable, nous avions évoqué au chapitre 2 une nouvelle approche de conception d'un protocole de bout en bout, basé sur l'utilisation de procédures de transfert de données paramétrables par l'utilisateur. XTP (Xpress Transfer Protocol) est actuellement le représentant le plus évolué de ce type d'atelier "multi-services" : par la modularité de ses procédures, il autorise son utilisateur à mettre en œuvre le protocole le plus adapté aux caractéristiques de l'application supportée.

C'est dans ce contexte que nous situons le chapitre 5 de ce mémoire, dont l'objet consiste à étudier comment utiliser XTP pour mettre en œuvre un service de Transport multimédia.

Ce chapitre est articulé en deux parties majeures. La section (I) présente les principes de conception du protocole XTP ainsi que ses mécanismes principaux. Nous étudions en section (II) comment paramétrer XTP afin de mettre en œuvre une connexion multimédia d'ordre partiel. Nous concluons cette dernière section en soulignant les avantages et les inconvénients du protocole XTP vis-à-vis de la conception d'un service et protocole d'ordre partiel et de fiabilité partielle.

I. Présentation du protocole XTP

XTP [Ches87] [Ches89] [PEI92] est le résultat d'études initiées en 1987 dans le cadre d'un projet mené par Protocol Engines Incorporated, visant à concevoir un protocole de transfert haute vitesse, implémentable en circuits VLSI [Schw90].

Outre les caractéristiques haute vitesse de ses mécanismes, l'originalité de XTP réside dans les multiples possibilités de mise en œuvre qu'il offre à ses utilisateurs : le développeur d'une application ou d'un service de Transport dispose d'un éventail de paramètres lui permettant de mettre en œuvre les mécanismes de XTP de façon adaptée à ses besoins. Dès à présent, nous soulignons que l'étude détaillée en section (II) propose d'utiliser XTP afin de développer un service de Transport et non une application

particulière ; de ce fait, notre démarche rend impossible à l'utilisateur du service tout accès à XTP, mais présente un caractère générique dans la mesure où elle s'adresse à un ensemble d'applications ayant pour caractéristique commune des contraintes d'ordre et de fiabilité différentes des contraintes "classiques" (ordre total et fiabilité totale/désordre et fiabilité nulle).

I.1. Concepts généraux

XTP est un protocole de transfert offrant des fonctionnalités de niveau Réseau et de niveau Transport, destinées à être mises en œuvre dans un environnement réseau de diffusion. Les caractéristiques principales du protocole que nous présenterons sont les suivantes :

- un format de PDU adapté à une parallélisation des traitements ;
- la possibilité d'ouvrir une connexion de façon implicite ou explicite ;
- un contrôle de flux par fenêtre activable à volonté ;
- un contrôle de débit par rafale ;
- un contrôle d'erreur modulable ;
- une gestion de priorité inter-contexte ;

En outre, XTP présente une fonction de multicast dont le mécanisme a fait l'objet d'une proposition de révision [Sant94].

Dans la présentation qui suit ne sont abordées que les fonctionnalités Transport de XTP pour les raisons suivantes : d'une part, l'utilisation de XTP que nous proposons en section (II) se situe dans le cadre d'un protocole de Transport ; d'autre part, la toute dernière version de XTP (XTP 4.0) dépouille le protocole de ses fonctionnalités réseau.

I.2. Structure des paquets XTP

Les PDUs XTP (ou *paquets*) se divisent en deux groupes : les *paquets d'information* et les *paquets de contrôle*.

- Les paquets d'information sont au nombre de cinq.
  - Le paquet **FIRST** est le premier paquet échangé entre deux sous-systèmes pour établir une association. Il contient un segment d'adresse et peut optionnellement véhiculer des données utilisateur.
  - Les paquets **DATA** sont utilisés pour véhiculer des données utilisateur.
  - Les paquets **PATH** ne contiennent qu'un segment d'adresse, et permettent de changer de route en cas de panne d'un sous-système intermédiaire, ou de joindre une communication en multicast.
  - Les paquets **DIAG** sont destinés à rapporter les messages d'erreur émis par les sous-systèmes en réponse à l'impossibilité de traiter un paquet reçu.

- Les paquets **ROUTE** permettent l'échange d'informations de routage entre les sous-systèmes intermédiaires.

- Le rôle des paquets de contrôle est de permettre l'échange de toutes les informations nécessaires au contrôle d'une association entre deux points. Deux types de paquets de contrôle sont définis :
  - les paquets **CNTL**, échangés entre sous-systèmes d'extrémité ;
  - les paquets **RCNTL**, échangés entre sous-systèmes intermédiaires.

Par la suite, nous présenterons plus en détail les paquets FIRST, DATA et CNTL dont nous envisagerons le transfert dans la section (II) de ce chapitre.

I.2.1. Format général

Le format général des paquets XTP est défini figure (5.1), et contient :

- un *segment d'en-tête* de 40 octets ;
- un *segment intermédiaire* de taille variable multiple de 4 octets ;
- un *segment de fin* de 4 octets.

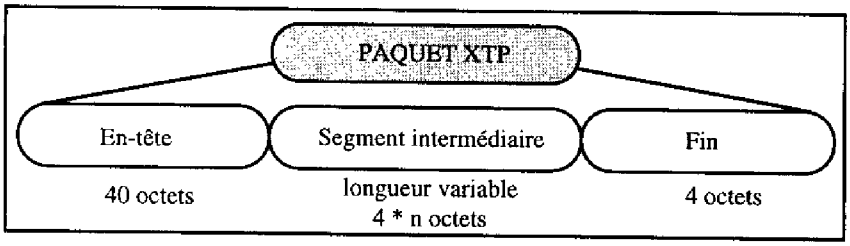


Figure (5.1) : Format général des paquets XTP

I.2.2. Segment d'en-tête

Le segment d'en-tête des paquets XTP est composé de 10 champs de 4 octets chacun (Figure 5.2).

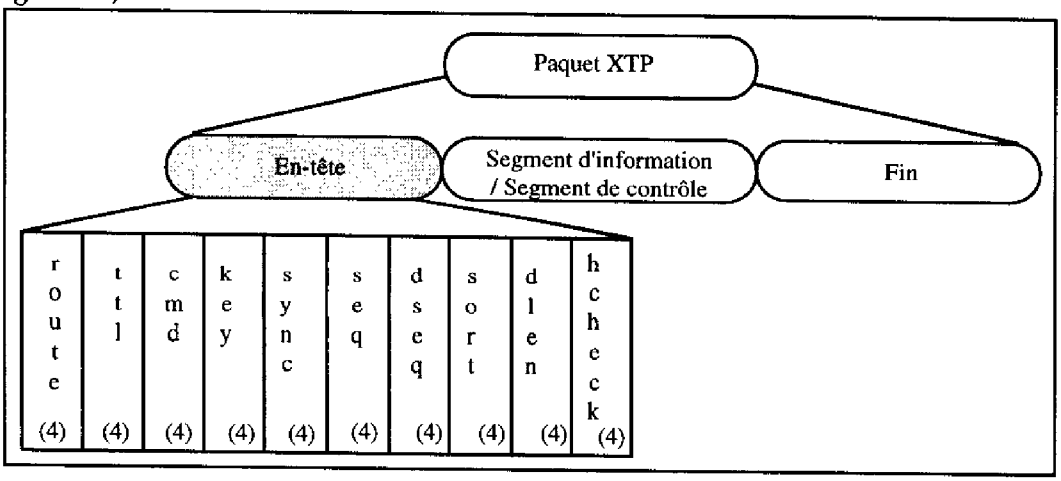


Figure (5.2) : Segment d'en-tête

- ***route*** est utilisé dans le cadre du routage des paquets par les sous-systèmes XTP traversés ;
- ***ttl*** spécifie le temps restant à vivre d'un paquet avant qu'il ne soit détruit ;
- ***cmd*** est explicité dans le paragraphe suivant ;
- ***key*** permet d'identifier le contexte auquel appartient le paquet ;
- ***sync*** est utilisé afin de synchroniser les sous-systèmes d'extrémité ;
- ***seq*** désigne le numéro de séquence :
  - du premier octet de données significatif du segment d'information : segment d'adresse pour les paquets FIRST, segment de données pour les paquets DATA ;
  - du prochain octet de données à envoyer par l'émetteur du paquet s'il s'agit d'un paquet CNTL ;
- ***dseq*** désigne le numéro de séquence du prochain octet en attente de délivrance à l'utilisateur ; recevant *dseq*, l'émetteur peut libérer les tampons de stockage des octets de numéro de séquence inférieur à *dseq*.
- ***sort*** indique le niveau de priorité du paquet ;
- ***dlen*** spécifie la longueur de la partie significative du segment intermédiaire (les octets de bourrage destinés à rendre le segment d'information multiple de 4 octets ne sont pas pris en compte) ;
- ***hcheck*** représente le total de contrôle effectué sur le segment d'en-tête (exceptés les champs *route* et *ttl* mis à jour par les sous-systèmes intermédiaires).

Le champ ***cmd*** se décompose en trois parties : *OPTIONS*, *OFFSET* et *PTYPE* (Figure 5.3)

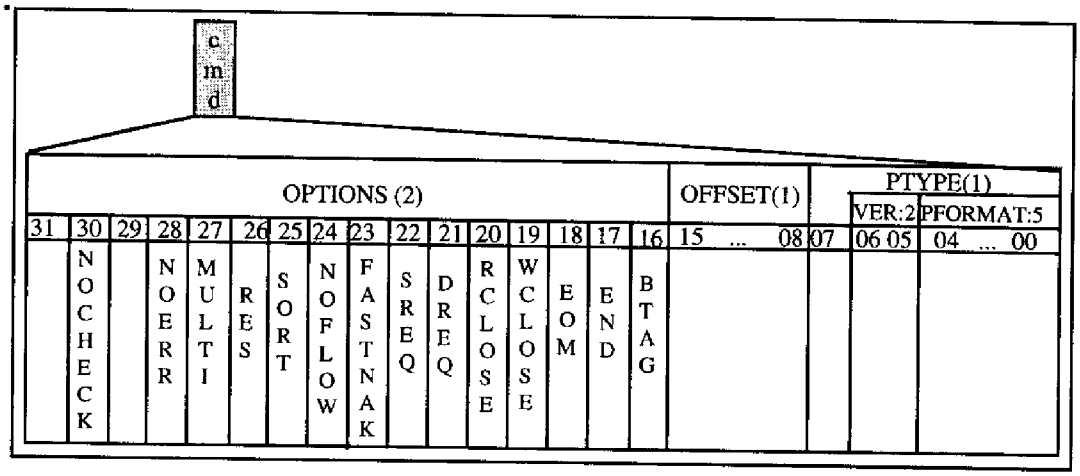


Figure (5.3) : Champs cmd

- ***PTYPE*** identifie le type du paquet (FIRST, DATA, CNTL, etc.) ;
- ***OFFSET*** indique l'emplacement du premier octet de données significatif ;
- ***OPTIONS*** comporte 14 drapeaux permettant à l'utilisateur de sélectionner les modes d'opération du protocole ; lorsqu'il est activé :

- ***NOCHECK*** indique au récepteur de ne pas prendre en compte le total de contrôle du segment intermédiaire ;
- ***NOERR*** impose la désactivation du contrôle des pertes : le récepteur ignore les pertes et l'émetteur ne procède à aucune retransmission ;
- ***MULTI*** indique que le paquet fait partie d'une association XTP en mode multicast ;
- ***RES*** indique au récepteur que la valeur du champ *alloc* du prochain paquet CNTL devra indiquer la taille des tampons mémoire disponibles au niveau utilisateur, et non au niveau XTP ;
- ***SORT*** intime l'activation en émission et en réception du mécanisme de priorité inter-contexte (la valeur du champ *sort* est alors interprétée) ;
- ***NOFLOW*** désactive le contrôle de flux ;
- ***FASTNACK*** indique au récepteur qu'il doit rapporter les pertes dès leur détection ;
- ***SREQ*** intime l'ordre au récepteur d'émettre un paquet CNTL dès la réception du paquet ;
- ***DREQ*** intime l'ordre au récepteur d'émettre un paquet CNTL dès la délivrance des données à l'utilisateur récepteur ;
- ***RCLOSE*** (respectivement ***WCLOSE***) indique que l'émetteur du paquet a fermé son flux de données en lecture (respectivement en écriture) ;
- ***EOM*** permet à l'utilisateur de délimiter les messages dans le flux de données : un paquet FIRST ou DATA dont le bit *EOM* a pour valeur 1 transporte nécessairement les derniers octets d'un message ; la fin d'un message est notifiée par l'utilisateur émetteur, et elle est signalée à l'utilisateur récepteur lors de la délivrance des données ;
- ***END*** indique que le contexte en émission est relâché ;
- ***BTAG*** indique au récepteur que les huit premiers octets du segment d'information des paquets FIRST et DATA contiennent des données particulières.

I.2.3. Segment de fin

Le segment de fin ne comporte qu'un seul champ, ***dcheck***, représentant le total de contrôle (*checksum*) effectué sur le segment intermédiaire ; *dcheck* est ignoré en réception lorsque le bit NOCHECK de l'en-tête est levé.

I.2.4. Segment intermédiaire

Le segment intermédiaire désigne :
- le *segment de contrôle* dans le cas des paquets de contrôle (CNTL et RCNTL) ;
- le *segment d'information* dans le cas des paquets de données (FIRST, DATA, PATH, ROUTE, DIAG).

Tous les paquets de contrôle ont un même segment de contrôle (Figure 5.4).

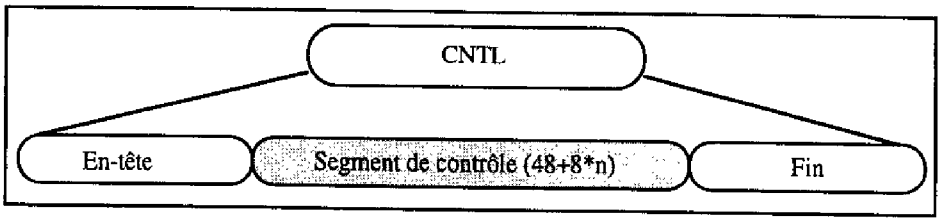


Figure (5.4) : Segment intermédiaire des paquets CNTL

En revanche, le segment d'information d'un paquet de données diffère selon son type ; en particulier :

- le segment intermédiaire des paquets FIRST est composé d'un *segment d'adresse* et d'un *segment de données* éventuellement vide (Figure 5.5) ;

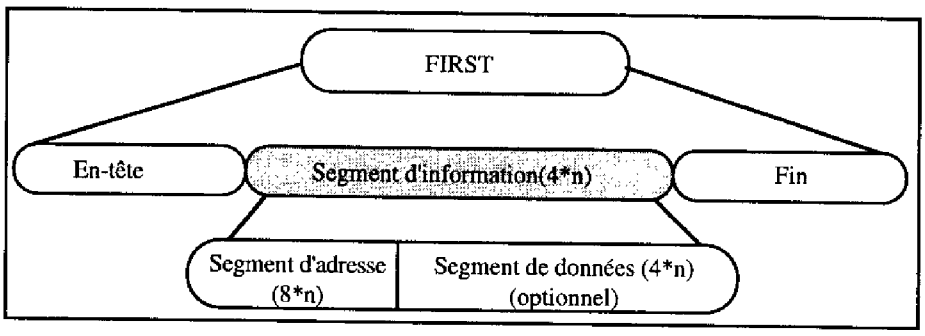


Figure (5.5) : Segment intermédiaire des paquets FIRST

- le segment intermédiaire des paquets DATA ne comporte qu'un *segment de données* (Figure 5.6).

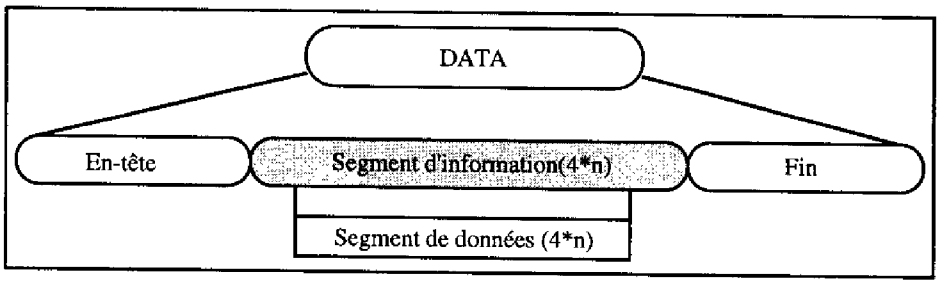


Figure (5.6) : Segment intermédiaire des paquets DATA

I.2.4.1. Segment d'adresse

Le segment d'adresse comporte toutes les informations relatives à l'adressage de l'émetteur et du récepteur (Figure 5.7).

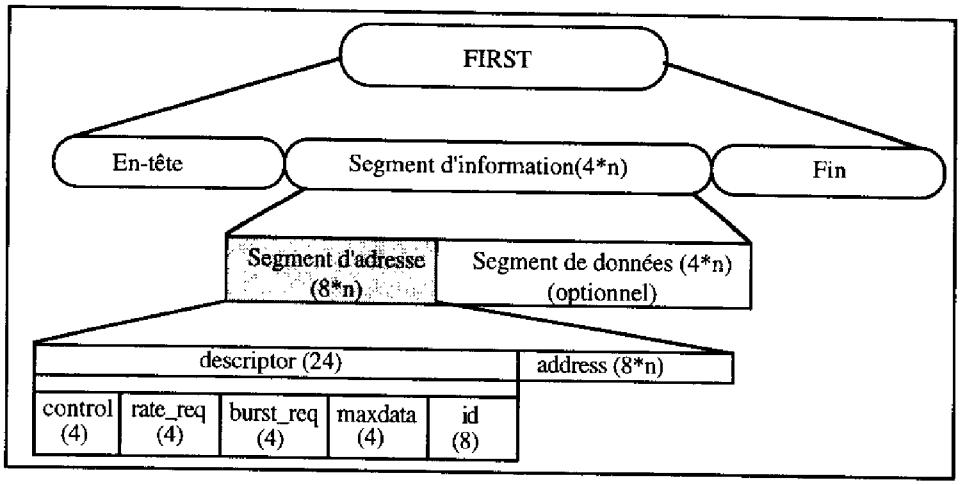


Figure (5.7) : Segment d'adresse

Le champ ***descriptor*** est composé de 5 champs :

- ***control*** indique la longueur totale du segment d'adresse (***alen***), le type de service attendu sur l'association (***service***), et la syntaxe d'adressage utilisée dans le segment d'adresse (***aformat***) ;
- ***rate_req*** indique le débit moyen en octets par seconde initialement souhaité par l'émetteur ;
- ***burst_req*** représente la longueur moyenne d'une rafale d'octets ;
- ***maxdata*** indique la longueur maximale du segment d'information que l'utilisateur désire transférer pour la durée de vie du contexte ;
- *id* contient l'adresse MAC de l'émetteur du paquet FIRST.

Remarque importante

Le champ *maxdata* peut faire échouer une demande d'établissement de connexion, si la valeur requise par l'utilisateur est susceptible de rendre la taille des paquets DATA supérieure à la taille maximale autorisée par l'ensemble des couches liaison impliquées dans le transfert de bout en bout. Cette spécification résulte du fait que XTP ne gère aucune fonction de segmentation et de réassemblage.

Le champ ***adress*** contient les adresses source et destination du paquet FIRST selon la syntaxe indiquée par *aformat*, parmi lesquels aucun adressage, l'adressage IP, ISO, etc.

I.2.4.2. Segment de données

Le segment de données est de longueur variable et comporte trois champs (Figure 5.8).

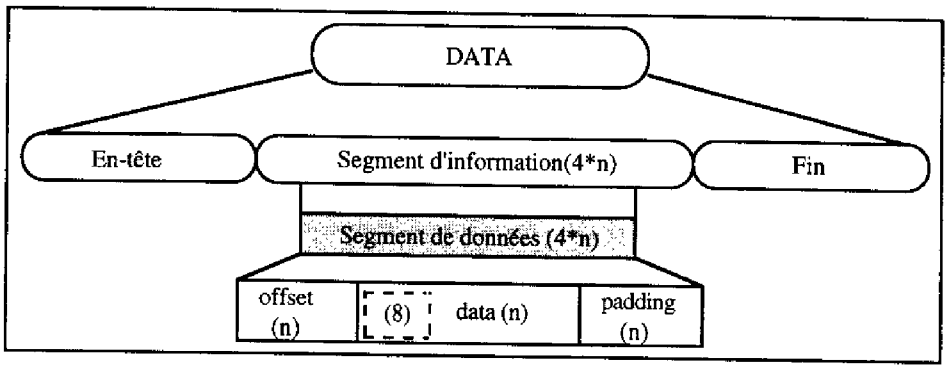


Figure (5.8) : Segment de données

- ***offset*** et ***padding*** sont deux champs de rembourrage, permettant de rendre la taille du segment de données multiple de 4 octets ;
- ***data*** contient les données utilisateur.

Remarque importante

Les 8 premiers octets du champ data présentent un caractère particulier si le bit *BTAG* de l'en-tête du paquet est positionné à un. Dans ce cas, ces octets sont délivrés à l'utilisateur récepteur en tant que données "marquées" ; nous verrons l'utilité de cette option dans la deuxième section de ce chapitre.

I.2.4.3. Segment de contrôle

Le segment de contrôle des paquets CNTL est composé de 12 champs de 4 octets chacun, et d'un champ de 8*n octets (Figure 5.9).

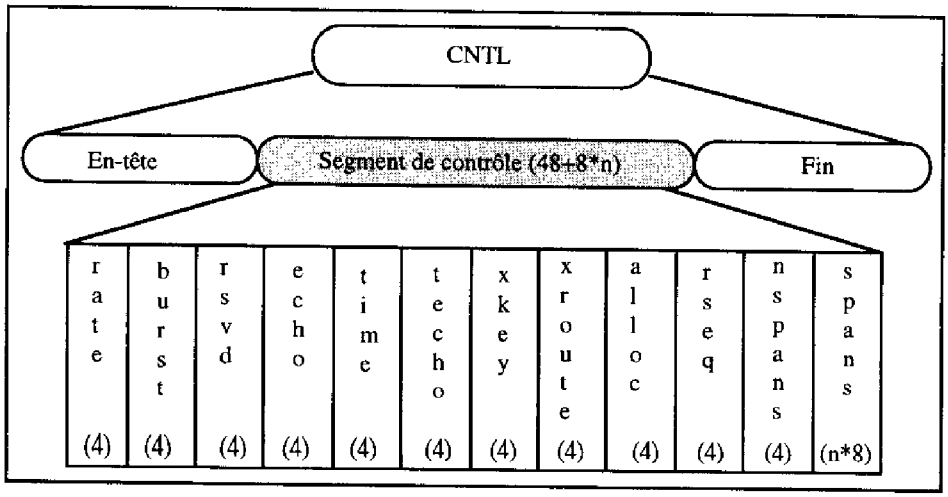


Figure (5.9) : Segment de contrôle

- ***rate*** et ***burst*** permettent de paramétrer le contrôle de débit en émission, et désignent respectivement le débit maximum en émission (en octets par seconde) et le nombre maximum d'octets émis dans une rafale d'octets ;

- ***rsvd*** n'est pas interprété dans la version 3.6 de XTP ;
- ***echo*** permet aux sous-systèmes d'extrémité de se synchroniser : lorsque l'un d'entre eux reçoit un paquet dont le bit *SREQ* a pour valeur 1, le paquet CNTL émis en retour contient dans son champ *echo* la valeur *sync* de l'en-tête du paquet reçu ;
- ***time*** et ***techo*** permettent de déterminer le temps de parcours aller-retour des paquets (RTT) : lorsqu'un sous-système XTP reçoit un paquet CNTL dont le bit *SREQ* a pour valeur 1, le paquet CNTL émis en retour contient dans son champ *techo* la valeur du champ *time* du paquet reçu ; le récepteur du paquet évalue le RTT en soustrayant la valeur de *techo* à celle du temps courant ;
- ***xkey*** est utilisé dans le cadre d'un mécanisme d'échange de clés destiné à accélérer le routage des informations en réception ;
- ***xroute*** permet de mettre en œuvre le mécanisme d'échange de route ; cette procédure a pour but d'accélérer le routage des informations ;
- ***alloc*** est une variable utilisée dans le cadre du contrôle de flux ; sa valeur fournit la taille mémoire disponible au niveau XTP si le bit *RES* a pour valeur 0, et au niveau utilisateur si le bit *RES* a pour valeur 1. Le contrôle de flux est ignoré si le bit *NOFLOW* de l'en-tête du paquet est positionné à 1 ; dans ce cas, *alloc* n'est pas interprété par le récepteur du paquet.
- ***rseq*** est utilisé dans le cadre du contrôle des pertes : sa valeur indique le prochain numéro d'octet attendu en réception ; lorsque le contrôle des pertes est désactivé, *rseq* est mis à jour à la valeur du plus grand numéro d'octet reçu plus un, et *dseq* prend la valeur de *rseq* ;
- ***nspan*** et ***spans*** indiquent le nombre de trous détectés en réception et l'emplacement de ces trous ; *spans* est constitué de *nspan* paires de numéros de séquence délimitant les octets manquants repérés dans le flux d'octets reçus.

I.3. Éléments de protocole

XTP assure l'établissement et la gestion de connexions bidirectionnelles (appelées *associations)* entre sous-systèmes de bout en bout, dans chacun desquels est instancié un *contexte* identifié par une clef (Figure 5.10).

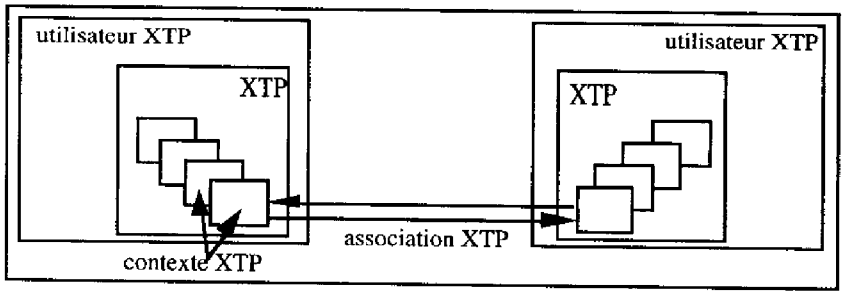


Figure (5.10) : Association XTP

Une association peut impliquer deux ou plusieurs contextes selon que la communication est point à point ou multipoint. Le modèle de transmission est orienté octets, mais le bit *EOM* des paquets DATA permet de délimiter les messages utilisateurs dans le flux d'octet. Nous détaillerons l'usage de ce bit en section (II).

L'intérêt majeur du protocole réside dans les multiples modes d'opération qu'il offre à l'utilisateur ; ce dernier peut en particulier activer/désactiver, par contexte ou par paquet, les mécanismes de contrôle d'erreur, de perte ou de flux. En outre, XTP offre un contrôle de débit paramétrable, particulièrement utile au transfert de média continus requérant une gigue de transmission la plus faible possible. Nous verrons comment utiliser ces fonctionnalités dans la deuxième section de ce chapitre.

Nous présentons maintenant les caractéristiques majeures de ces différents mécanismes.

I.3.1. Ouverture de connexion

XTP offre deux modes d'ouverture de connexion. Le premier de ces modes, présenté Figure (5.11a), permet de réaliser une ouverture de connexion explicite. Le sous-système émetteur envoie un paquet FIRST, et attend d'avoir eu réception d'un acquittement de ce paquet avant d'émettre des paquets de données. On notera dans ce cas l'obligation que soit activé le bit *SREQ* du paquet FIRST, imposant une réponse (c'est-à-dire un paquet CNTL) de la part du récepteur. Le second mode permet de réaliser une ouverture de connexion implicite : les paquets DATA sont envoyés dans la foulée du paquet FIRST, sans que celui-ci ait été préalablement acquitté (Figure 5.11b).

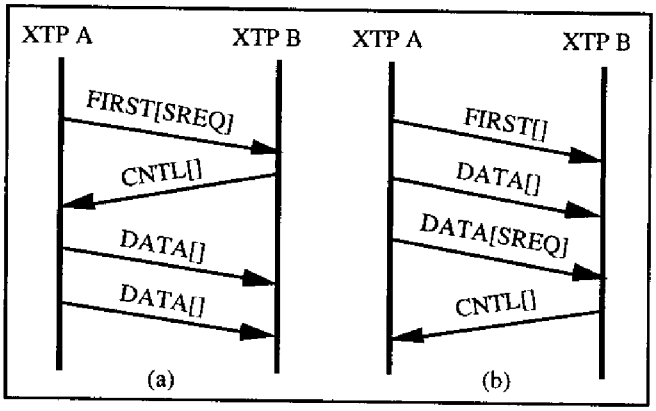


Figure (5.11) : Ouverture de connexion explicite (a)/implicite(b)

Échange de clef

Une ouverture de connexion requise par un utilisateur A se traduit tout d'abord par la création d'un contexte XTP en émission identifié par une clef (K_A). Cette clef est véhiculée dans le paquet FIRST émis en conséquence par l'intermédiaire de son champ *key*. Sur le site destinataire (B), la réception du paquet FIRST n'engendre l'établissement de la

connexion que s'il existe un contexte XTP (K_B) en attente de connexion ; dans ce cas, le récepteur enregistre la clé K_A du contexte appelant : les paquets de données dorénavant émis de B vers A comporteront dans leur champ *key* la clef K_A du contexte adressé. Cette procédure constitue la première phase du mécanisme d'échange de clef de XTP, permettant, une fois réalisé, d'accélérer le routage des paquets en réception. La seconde et dernière phase de cette procédure a lieu lorsque le sous-système d'extrémité A enregistre la clef K_B du contexte XTP pair. L'échange correspondant intervient lorsque le sous-système B émet un paquet CNTL à destination de K_A : son champ *xkey* comporte la clef du contexte K_B (Figure 5.12)

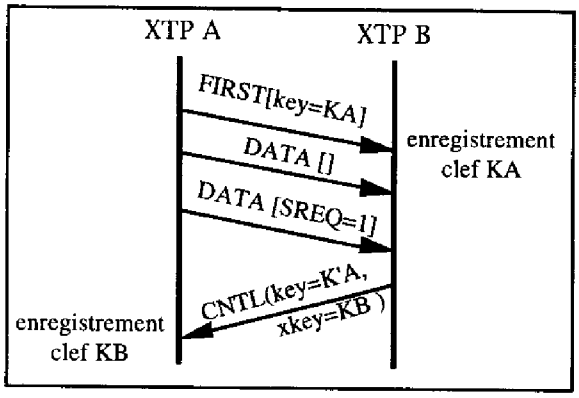


Figure (5.12) : Échange de clef

Remarque : le champ *key* est "primé" lorsqu'il indique la clef du destinataire, le premier bit du champ étant dédié à cette information.

I.3.2. Fermeture de connexion

XTP offre tous les modes de fermeture de connexion imaginables : forcé ou négocié à volonté et pour chaque sens de l'association. La libération d'une association repose sur l'utilisation des champs *WCLOSE*, *RCLOSE* et *END* :

- l'émission d'un paquet dont le champ *WCLOSE* a pour valeur 1 indique que l'émetteur a fermé son flux en émission : aucune nouvelle donnée ne sera dorénavant émise ; en revanche, la retransmission par XTP des paquets perdus par le réseau est encore possible ;
- l'émission d'un paquet dont le champ *RCLOSE* a pour valeur 1 indique que l'émetteur a fermé son flux en réception : aucune donnée supplémentaire ne sera donc délivrée à l'utilisateur du contexte ;
- l'émission d'un paquet dont le champ *END* a pour valeur 1 indique que le contexte émetteur a été relâché.

La figure (5.13) illustre un exemple de fermeture de connexion négociée de A vers B, et forcée de B vers A :

- en activant le drapeau *WCLOSE*, XTP A indique qu'il stoppe ses émissions mais autorise XTP B à faire état des pertes observées afin de retransmettre les paquets perdus ; en activant le drapeau *RCLOSE*, XTP A indique en outre qu'il ne souhaite plus recevoir de données : il est donc impossible de récupérer les pertes survenues sur le flux de données de B vers A ;
- une fois les pertes éventuelles de A vers B récupérées, XTP B ferme la connexion en activant le drapeau *END*.

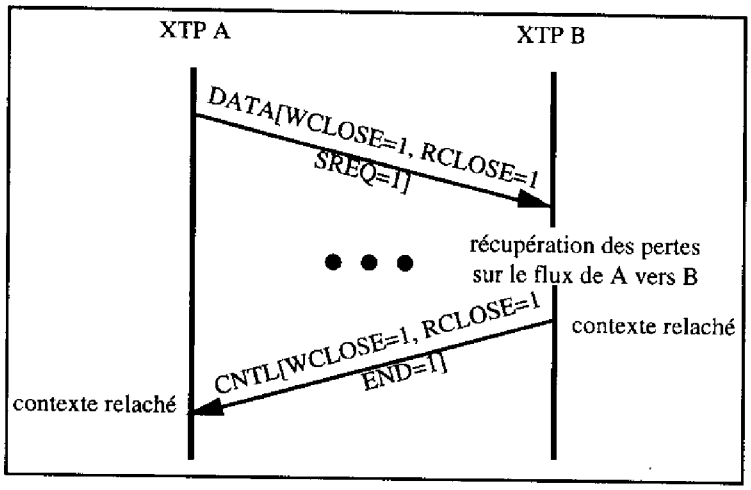


Figure (5.13) : Exemple de fermeture de connexion forcée/négocié

I.3.3. Mécanismes de contrôle

Une fois la connexion établie, un certain nombre de mécanismes de contrôle sont mis en œuvre pour assurer le transfert des données. Cinq de ces mécanismes, le contrôle de flux par fenêtre et/ou par débit, le contrôle des erreurs bits, le contrôle des pertes, et le contrôle de priorité de traitement peuvent être activés à volonté par l'utilisateur.

I.3.3.1. Contrôle de flux par fenêtre

Le contrôle de flux est basé sur une fenêtre coulissante dont la mise à jour en fonctionnement normal du réseau est effectuée à l'initiative de l'émetteur par positionnement du bit SREQ du paquet émis. Le paquet CNTL émis en retour contient dans son champ *alloc* la valeur de la nouvelle borne supérieure de la fenêtre. Selon l'implantation du protocole, les demandes de mise à jour peuvent être générées automatiquement (après émission de x% de la fenêtre d'émission), et/ou à l'initiative d'une demande explicite de l'utilisateur émetteur. Nous verrons l'intérêt de cette dernière possibilité dans la section (II) de ce chapitre.

I.3.3.2. Contrôle de flux par débit

En plus du contrôle de flux par fenêtre, un contrôle de débit par rafale est proposé par XTP. Ce contrôle est géré par l'intermédiaire de trois variables maintenues en émission, *CREDIT*, *RATE* et *BURST*, et d'un temporisateur *RTIMER*.
Coté XTP appelé, *RATE* et *BURST* sont mises à jour à partir de la valeur des champs *rate_req* et *burst_req* contenus dans le paquet FIRST initiant la connexion ; coté XTP appelant, *RATE* et *BURST* prennent la valeur des champs *rate* et *burst* contenus dans le premier paquet CNTL reçu. Initialement allouée à la valeur de *BURST*, *CREDIT* désigne le nombre d'octets que peut envoyer XTP durant la durée *RTIMER=BURST/RATE*.
A chaque émission de données, la valeur de *CREDIT* est décrémentée du nombre d'octets envoyés ; si *CREDIT* atteint une valeur négative ou nulle, l'émetteur stoppe ses émissions jusqu'à l'expiration de *RTIMER*. A l'expiration de *RTIMER*, *CREDIT* prend la valeur de *BURST* si sa valeur était positive, ou la valeur de (*CREDIT+BURST*) dans le cas contraire, et *RTIMER* est réactivée à la valeur *BURST/RATE*.

I.3.3.3. Contrôle des erreurs bits

XTP gère deux totaux de contrôle, un sur les données par l'intermédiaire du champ *dcheck*, et un sur l'en-tête par l'intermédiaire du champ *hcheck*. Celui portant sur les données peut être désactivé par l'utilisateur, le bit *NOCHECK* de l'en-tête des paquets DATA étant alors positionné à 1.

I.3.3.4. Contrôle des pertes

A l'instar du contrôle des erreurs bits, le contrôle des pertes de XTP peut être activé (respectivement désactivé) par positionnement à 0 (respectivement à 1) du bit *NOERR* des paquets échangés. Ce contrôle peut être mis en œuvre par paquet ou par contexte (c'est-à-dire pour toute la durée de vie du flux de données considéré).

Détection des pertes

La détection des pertes est basée sur une numérotation séquentielle des octets de données contenus dans les paquets DATA (le champ *seq* identifiant le premier octet significatif). XTP maintient en réception une variable locale *RSEQ* indiquant le numéro d'octet attendu dans l'ordre séquentiel.
Sur réception d'un paquet DATA, le sous-système détecte une perte lorsque la valeur de *seq* diffère de celle de *RSEQ*. Si le bit *NOERR* du paquet a pour valeur 0, XTP enregistre l'occurrence d'un trou supplémentaire dans une variable locale *NSPAN*, ainsi que l'emplacement de ce trou dans une liste *SPANS* ; *RSEQ* n'est pas mise à jour. Si le bit

NOERR a pour valeur 1, les pertes éventuelles sont ignorées : *RSEQ* est mis à jour à la valeur (*seq+dlen*), *SPANS* et *NSPAN* n'étant pas mises à jour.

Rapport des pertes

Le rapport des pertes est notifié à l'émetteur par l'intermédiaire des paquets CNTL, dont les champs *nspan* et *spans* ont pour valeur respective la valeur courante de *NSPAN* et *SPANS*. Le champ *rseq* indique le prochain numéro d'octet attendu en séquence.

Les paquets CNTL émis pour rapporter les pertes peuvent être générés de plusieurs façons, et notamment :

- sur réception d'un paquet dont le bit *SREQ* a pour valeur 1 ; dans ce cas, un paquet CNTL est immédiatement retourné ;
- sur réception d'un paquet DATA dont le bit *DREQ* a pour valeur 1 ; dans ce cas, un paquet CNTL est retourné lors de la délivrance à l'utilisateur des données contenues dans le paquet DATA ;
- sur réception d'un paquet DATA dont les données sont hors séquence et dont le bit *FASTNAK* a pour valeur 1 ; dans ce cas, un paquet CNTL est immédiatement retourné.

Ce dernier cas de figure permet de mettre en œuvre un contrôle des pertes "agressif" dans lequel toute détection de perte en réception engendre l'émission immédiate d'un paquet CNTL en retour ; ce mode de contrôle permet à l'émetteur de retransmettre "au plus tôt" les données perdues par le réseau.

Retransmissions

XTP est conduit à retransmettre des données dans le cas de figure suivant : sur réception d'un paquet CNTL, l'émetteur retransmet toutes les données indiquées manquantes par le paquet. Notons que les retransmissions peuvent être effectuées en *Go-back-N* ou en *Selective Repeat*, selon l'implantation du protocole.

Remarque
La version 3.6 de XTP préconise une utilisation du champ *NOERR* par contexte (c'est-à-dire pour tous les paquets émis dans le sens de l'association considéré), plutôt que par paquet, sans plus de précision. Cette remarque résulte de l'ambiguïté de certains scénarios lors des changements de mode de contrôle de perte. L'étude des différents cas de figure nous conduit à la conclusion que le changement de mode entre deux paquets DATA émis consécutivement nécessite qu'une synchronisation soit tout d'abord effectuée entre sous-systèmes XTP ; cette analyse est confirmée à la lecture de la dernière version du protocole. Cette synchronisation peut être effectuée par un échange de paquets CNTL avant émission du premier paquet DATA émis dans le nouveau mode (Figure 5.14a et 5.14b).

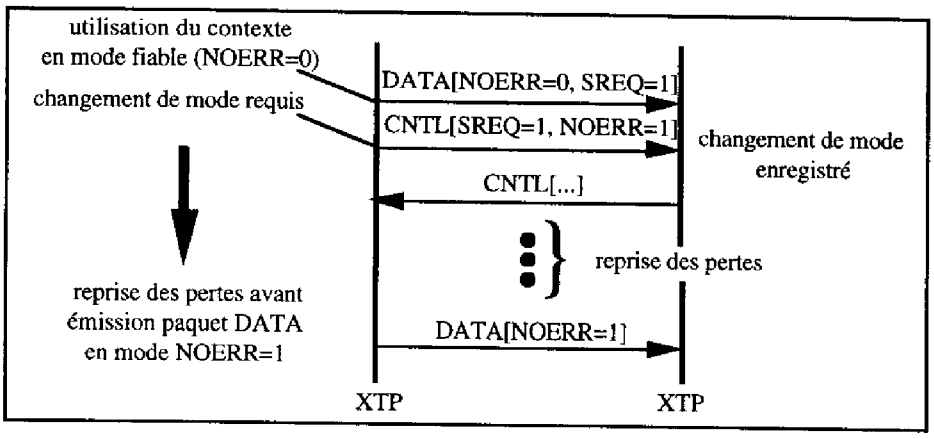


Figure (5.14a) : Passage du mode fiable au mode non fiable

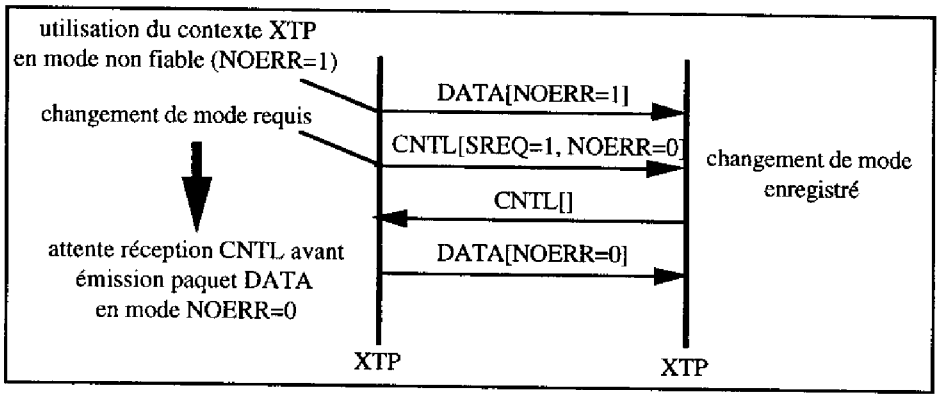


Figure (5.14b) : Passage du mode non fiable au mode fiable

I.3.3.5. Contrôle de la priorité de traitement

XTP propose un mécanisme de priorité de service entre les contextes instanciés sur un même site. Lorsqu'il est mis en œuvre, ce mécanisme se traduit par le positionnement à 1 du bit *SORT* des paquets émis, la priorité du contexte étant identifiée dans le champ *sort*. En émission, un contexte prioritaire est servi par le gestionnaire de contextes aussi longtemps qu'il a des données à émettre ou à retransmettre ; un contexte nouvellement actif présentant une plus forte priorité engendre la suspension du contexte en cours de service. En réception, les paquets de plus forte priorité sont traités de façon prioritaire, les paquets reçus et non traités étant stockés dans la file d'attente associée à leur contexte.

I.4. Conclusion

Dans cette section ont été présentés les mécanismes principaux du protocole XTP. Notre étude visant à proposer la mise en œuvre d'un service d'ordre partiel à l'aide de XTP , les mécanismes correspondant aux fonctionnalités réseau du protocole n'ont pas été abordés ; notons cependant que le routage proposé dans XTP 3.6 repose sur la mise en

œuvre au préalable d'un mécanisme d'échange de route entre sous-systèmes intermédiaires : ce mécanisme, analogue au mécanisme d'échange de clefs, a pour but d'accélérer le routage des paquets au travers du réseau de communication ; en outre, dans l'optique d'éviter les phénomènes de congestion du réseau, le contrôle de débit par rafale (décrit en I.3.3.2) est destiné à être mis en œuvre non seulement de bout en bout, mais également entre chacun des noeuds impliqués dans le cadre d'une association XTP.

Plusieurs études ont été réalisées autour de XTP dans le but d'en envisager la mise en œuvre sur des réseaux à large bande ; selon [Kure93], une implantation de XTP sur la technologie ATM soulève comme problème majeur l'adaptation de la signalisation de XTP à celle utilisée dans ATM. [Thai95] envisage la mise en œuvre de XTP sur l'AAL5 en insistant sur la nécessité de définir un service de Transport multimédia ; suivant cet objectif, des propositions d'extension de XTP ont été envisagées dans le cadre des projets BERKOM et CIO : les deux propositions, respectivement XTP-Lite [Delg93] et XTPX [Metz93], suggèrent l'ajout d'un segment de QoS au paquet FIRST et CNTL, afin de permettre la spécification et la négociation de paramètres de QoS.

L'étude que nous présentons maintenant ne suggère pas une modification de XTP, mais en propose une utilisation dans le cadre d'un protocole de Transport multimédia d'ordre partiel et de fiabilité partielle.

II. Mise en œuvre d'un protocole multimédia d'ordre partiel à l'aide de XTP

Contrairement à TCP, les principes de conception de XTP permettent d'en envisager la mise en œuvre suivant différents modes ; en particulier, l'activation par contexte ou par paquet de son contrôle d'erreur fait que XTP parait a priori tout destiné à autoriser la mise en œuvre d'un service d'ordre partiel ; au delà des caractéristiques haute vitesse du protocole, cette spécificité nous conduit à envisager l'étude présentée ci-après.

Les principes d'implantation d'une connexion multimédia d'ordre partiel (MM-POC) impliquent l'instanciation et la coordination de n POCs monomédia, assurant chacune le transport d'un flux utilisateur avec une certaine QoS. Dans cette section, nous étudions comment utiliser le protocole XTP dans chacune des POCs d'une MM-POC pour mettre en œuvre un protocole d'ordre partiel de niveau Transport, offrant les QoS ordre et fiabilité définies dans le service MM-POS présenté au chapitre 4.
Nous présentons tout d'abord les éléments de protocole définis pour chaque POC. Dans un second temps, nous décrivons l'architecture de communication choisie intégrant le protocole XTP comme base de transport. Enfin, nous étudions comment paramétrer XTP pour mettre en œuvre les éléments précédents, dans le cadre de l'architecture définie.

II.1. Éléments de protocole

Au regard de la définition du service MM-POS, un protocole multimédia d'ordre partiel présente deux niveaux de fonctionnalités :

- le premier est relatif aux fonctionnalités "multimédia" (ou de niveau MM-POC) du protocole, impliquant ouverture et fermeture de la connexion multimédia, notification de dégradation de la QoS et renégociation de la QoS.
- le second est relatif aux fonctionnalités "monomédia" (ou de niveau POC), et concerne la phase de transfert de données de chacune des POCs monomédia.

Nous présentons dans cette section les mécanismes nécessaires pour mettre en œuvre les fonctionnalités monomédia du protocole, en restreignant celles-ci à l'ordre et la fiabilité. Afin de simplifier le cadre de notre étude, une seule des deux sémantiques de service définies en terme de fiabilité (chapitre 4 - section III.1.2) est ici envisagée : la garantie au détriment de toute prise en compte de la validité temporelle des SDUs. Nous serons donc conduits à envisager le stockage des données émises jusqu'à leur acquittement par l'entité de Transport réceptrice.

La spécification Estelle du protocole POC présentée au chapitre 3 avait pour objectif de caractériser le concept de connexion d'ordre partiel dans le cadre d'un protocole (POC) décrit en Estelle. Le protocole présenté ici diffère sur les deux points suivants :

- le protocole doit à présent offrir un service de fiabilité partielle à QoS assurée, dont l'expression correspond à celle définie dans le service MM-POS ; en conséquence, il apparaît obligatoire de définir un mécanisme de contrôle permettant la récupération des pertes "inacceptables" au regard de la QoS fiabilité ;
- dans la spécification Estelle, le contrôle des pertes est basé sur un mécanisme d'acquittement sélectif positif, impliquant l'activation d'un temporisateur par PDU émise ; nous envisagerons un mécanisme différent, dont les principes sont davantage orientés vers des transferts à haute vitesse.

Pour assurer les fonctionnalités de niveau POC, le protocole défini pour chaque POC comporte trois mécanismes principaux : un mécanisme de gestion de l'ordre partiel (intra-flux et inter-flux), un mécanisme de gestion de la fiabilité partielle et un mécanisme de segmentation et de réassemblage des SDUs.
L'objet de cette section n'est pas de spécifier ces mécanismes dans un langage de description formelle tel qu'Estelle, mais d'étudier comment les mettre en œuvre à l'aide de XTP. Dans un premier temps, nous présentons informellement chacun de ces mécanismes.

II.1.1. Gestion de l'ordre partiel

Les principes de conception du mécanisme de gestion de l'ordre partiel sont identiques à ceux définis au chapitre 3 de ce mémoire : la "délivrabilité" d'une SDU est établie en réception, par rapport au numéro de séquence qui lui a été attribué en émission.
On doit ici rappeler que le service de Transport MM-POS assure la délivrance de messages et non de flux d'octets ; en conséquence, la gestion de l'ordre partiel repose sur la considération de messages, que nous désignerons par la suite sous le terme de POC-SDU.

II.1.2. Gestion de la fiabilité partielle

Dans le contexte que nous nous sommes fixés, la QoS fiabilité se résume pour chaque POC aux deux premières composantes définies (chapitre 4 - section III.1.2), en l'occurrence :

- la possibilité que les données altérées soient délivrées à l'utilisateur ;
- le nombre k des pertes consécutives acceptables, couplé au nombre l de pertes admissibles par plage d'observation de longueur w.

Pour assurer ce dernier point, plusieurs protocoles sont envisageables, mais tous nécessitent que les PDUs émises soient stockées en émission jusqu'à leur acquittement par le récepteur, excepté dans le cas particulier où la fiabilité requise est nulle (auquel cas les PDUs émises ne sont pas stockées en émission). En particulier, le mécanisme d'acquittement sélectif positif proposé dans la spécification Estelle du protocole POC est compatible avec un tel service si le récepteur acquitte les PDUs déclarées perdues afin d'autoriser la délivrance d'une SDU dans l'ordre partiel (à condition que ces pertes n'engendrent pas une dégradation de la QoS fiabilité).
Nous préférons cependant envisager l'implantation d'un mécanisme a priori plus performant, faisant actuellement l'objet de mesures de performance au LAAS/CNRS [Robe95].
Les acquittements sont générés de deux façons, à l'initiative de l'émetteur ou à celle du récepteur ; dans les deux cas, le rapport véhiculé identifie la liste des PDUs à retransmettre, et acquitte les PDUs traitées (c'est-à-dire délivrées ou déclarées perdues).

- Un acquittement est réclamé par l'émetteur lorsqu'il désire mettre à jour ses tampons de stockage ; une implantation possible de ce mécanisme consiste à activer un champ de contrôle particulier (analogue au bit *SREQ* de XTP) dans le dernier paquet émis avant arrêt des émissions, faute de tampon mémoire disponible, et de retransmettre périodiquement ce paquet jusqu'à réception de son acquittement. L'inconvénient de cette politique réside dans les périodes de silence qu'elle génère en émission : une amélioration envisageable consiste à ce que des demandes d'acquittement "anticipées"

soient effectuées avant le dernier paquet émis ; [Robe95] distingue ainsi deux types de demandes d'acquittement : les demandes impératives effectuées avant blocage de l'émetteur et nécessitant un transfert fiable, et les demandes anticipées ne nécessitant pas un transfert fiable du fait que leur perte par le réseau n'engendre pas un blocage de l'émetteur.

- En complément, la réception d'une POC-SDU non délivrable au regard de l'ordre partiel multimédia et dont la délivrance impliquerait une dégradation de la QoS fiabilité, engendre de la part du récepteur l'émission d'un acquittement permettant à l'émetteur de retransmettre au plus tôt les pertes inacceptables.

II.1.3. Segmentation/réassemblage

Chaque connexion POC fournit un service de transfert de messages, dont la taille maximale est spécifiée par l'utilisateur (chapitre 4 - section III.1.5). Selon la taille maximale autorisée des unités de données du service sous-jacent, un mécanisme de segmentation/réassemblage est donc potentiellement à mettre en œuvre.
Ce mécanisme, implicitement présent dans la spécification Estelle du protocole POC, est ici mis en évidence pour la raison suivante : XTP n'assurant aucune fonction de segmentation et de réassemblage, nous serons conduits lors de l'implantation du protocole à l'aide de XTP, à déterminer comment délimiter les SDUs de niveau POC transmises dans le segment de données des paquets XTP.

Nous présentons maintenant l'architecture de communication retenue, intégrant le protocole XTP.

II.2. Architecture

Considérons la mise en œuvre d'une connexion MM-POC supportant le transfert de n flux. Pour transférer des données, deux modes d'utilisation de XTP sont envisageables :
1 - le mode datagramme, en établissant, utilisant puis relâchant un contexte XTP à chaque échange de SDU ;
2 - le mode connecté en :

(a) établissant une association XTP,

(b) utilisant cette association comme support de transfert de toutes les données,

(c) relâchant l'association lors de la fermeture de la connexion.

Une association XTP par POC monomédia

Les principes d'implantation d'une MM-POC énoncés au chapitre 4 ont pour point de départ l'instanciation d'une POC à qualités de service données pour chaque flux utilisateur ; en accord avec [Tenn89], le multiplexage de plusieurs flux sur une même POC

n'est pas envisagé. Poursuivant ce raisonnement, l'architecture que nous proposons (Figure 5.15) impose l'établissement d'une association XTP par POC monomédia, avant le transfert des données.

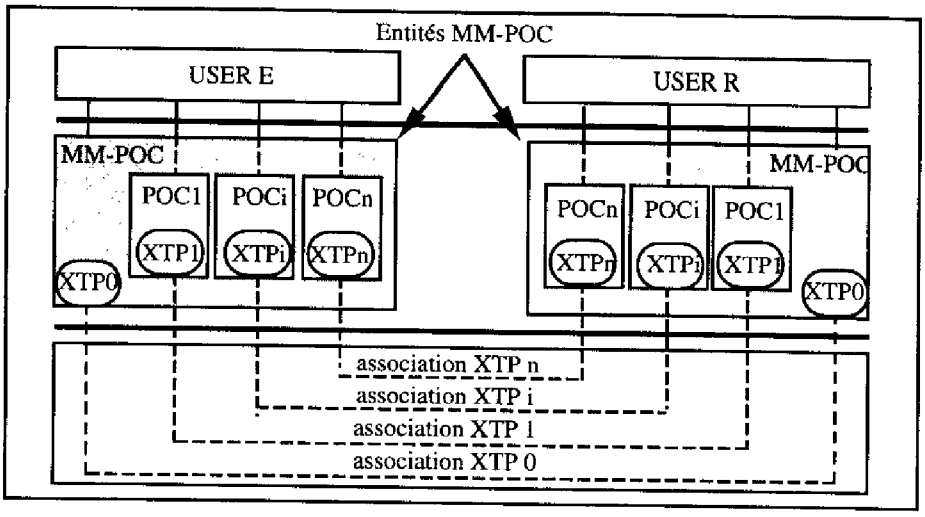


Figure (5.15) : Architecture de Transport multimédia

Quatre raisons supplémentaires motivent ce choix d'architecture :

- seul le mode connecté de XTP est susceptible d'autoriser une négociation de la QoS et une réservation des ressources en conséquence ;
- les mécanismes de contrôle de flux et de débit du protocole XTP ne sont effectifs que dans ce contexte d'utilisation ; il apparaît alors possible de décharger le protocole POC de ces mécanismes et d'en laisser le soin à XTP ;
- une association XTP est bidirectionnelle et permet de sélectionner des modes d'opération différents pour chaque sens de l'association (reprise des pertes et détection des erreurs notamment) ; de ce choix résulte la possibilité de véhiculer les PDUs de contrôle de POC_i sur l'association XTP support de POC_i, évitant ainsi d'avoir à distinguer ces PDUs par un identificateur de la connexion monomédia à laquelle elles se réfèrent ;
- enfin, l'activation d'un contexte XTP par émission de POC-SDU engendrerait un temps de transit pénalisant, contraire aux objectifs de transfert à haute vitesse ciblés.

L'étude consiste donc à établir la mise en œuvre des mécanismes définis en (II.1), dans le cadre d'une association XTP préalablement établie.

Une association XTP de contrôle hors bande

L'architecture proposée figure (5.15) fait apparaître une $(n+1)^{ème}$ association XTP (notée XTP_0), reliant les entités MM-POC. Cette association est destinée à supporter le transfert fiable des PDUs échangées pour assurer les fonctionnalités de niveau MM-POC. Le choix

de ne pas utiliser l'une quelconque des n associations XTP_i (i allant de 1 à n) pour supporter le transfert des PDUs MM-POC, permet d'envisager une gestion indépendante des PDUs de niveau POC et de niveau MM-POC. De plus, le fait que ces deux niveaux de PDUs ne réclament pas une QoS identique ne fait que renforcer ce choix.

En conclusion, il apparaît que sur chacun des sites communicants, un contexte MM-POC unique coordonne n sous-contextes POC_i, incluant chacun le contexte XTP relatif à l'association XTP_i, support de POC_i. En outre, une association XTP supplémentaire permet aux entités MM-POC de gérer l'ensemble des fonctionnalités de niveau MM-POC. Nous proposons maintenant une implantation des mécanismes définis au paragraphe (II.1), au travers de cette architecture à deux niveaux, (POC_i/XTP_i) et ($MM\text{-}POC/XTP_0$). Les choix de mise en œuvre de ces mécanismes nous conduiront à définir précisément les PDUs de niveau POC (ou POC-PDU), et à souligner les avantages et les inconvénients du protocole XTP.

II.3. Mise en œuvre des fonctionnalités de niveau POC

Pour simplifier les notations, nous considérerons le cas générique présenté (Figure 5.16), dans lequel la connexion POC représente l'une quelconque des n POCs de la connexion MM-POC, et XTP l'association support de transfert des POC-PDUs.

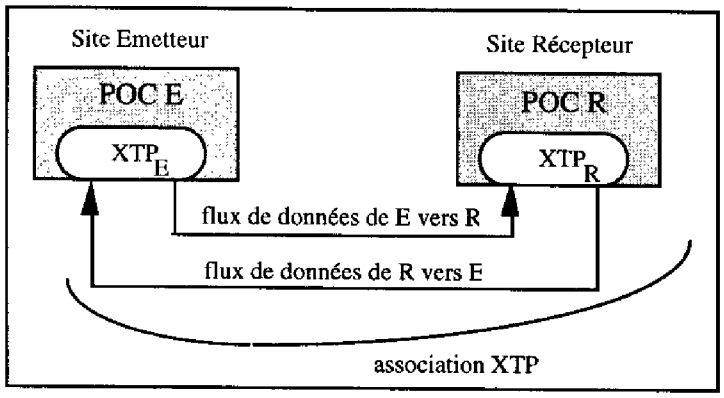


Figure (5.16) : Cas générique

Sur la figure (5.16) sont explicitement identifiés les deux contextes XTP_E et XTP_R de l'association XTP ; nous désignerons par $XTP_{(E->R)}$ le flux de données XTP de E vers R, et par $XTP_{(R->E)}$ le flux de données XTP de R vers E.

Afin de clarifier la démarche adoptée dans la présentation qui suit, nous soulignons dès à présent la contradiction à laquelle nous avons été confrontés.
Dans le protocole POC, les mécanismes relatifs à la gestion de l'ordre et de la fiabilité fonctionnent sur la base de messages (typiquement des POC-SDUs). A l'opposé, les principes de conception des mécanismes de XTP reposent sur la gestion de flux d'octets

(représentés par les flux XTP(E->R) et XTP(R->E) de la figure précédente) ; de ce fait et bien qu'il soit possible de délimiter les messages transférés, les mécanismes de XTP ne reposent pas sur une identification de ces messages ; ceci souligne simplement le fait que XTP n'envisage ni la gestion de l'ordre partiel, ni celle d'une fiabilité partielle reposant sur la notion de message.

Partant de cette contradiction, nous serons conduits dans les paragraphes suivants à envisager la gestion de l'ordre et de la fiabilité partiels au niveau POC, et non au niveau XTP. Pour cela, nous définissons dès à présent deux types de PDUs de niveau POC (où POC-PDUs) :

- les POC-PDUs *POC-DT*, permettant de réaliser une interface entre les POC-SDUs et les paquets DATA de XTP ; ces PDUs seront véhiculées sur le flux XTP(E->R) ;
- les POC-PDUs *POC-CTL*, nécessaires à la gestion des mécanismes de gestion de l'ordre et de la fiabilité partiels ; ces PDUs seront véhiculées sur le flux XTP(R->E).

II.3.1. Structure générale des PDUs

Les POC-DT sont constituées de deux parties, un en-tête et un segment de données. Le segment de données d'une POC-DT est entièrement dédié aux données utilisateur d'une et une seule POC-SDU. Par abus de langage, nous dirons qu'une POC-DT véhicule une POC-SDU (Figure 5.17). Nous déterminerons le contenu exact de l'en-tête en fonction de l'utilisation de XTP qui sera proposée.

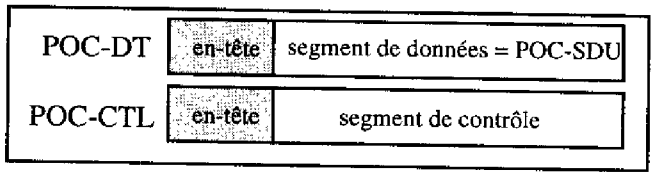


Figure (5.17) : Format général des POC-PDUs

Contrairement aux POC-DT, les PDUs POC-CTL ne contiennent pas de données utilisateur ; elles comportent un segment de contrôle véhiculant les informations nécessaires à la gestion des mécanismes de niveau POC. Nous montrerons que le segment d'en-tête des POC-CTL est en réalité vide du fait de l'utilisation que nous proposerons de XTP.

Il s'agit à présent d'étudier comment paramétrer XTP pour transférer ces PDUs, en réalisant si besoin est leur segmentation et leur réassemblage.

II.3.2. Gestion de l'ordre partiel

Comme pour tous les protocoles existants, le concept de connexion d'ordre partiel n'est pas implanté dans XTP. En conséquence, le mécanisme de gestion de l'ordre partiel n'est envisageable qu'au niveau POC.

II.3.2.1. Gestion de l'ordre partiel intra-flux

Dans le cas générique de la figure (5.16), l'association (XTP_E, XTP_R) est établie entre entités POC communicantes avant le transfert des données. Une conséquence immédiate de cette utilisation de XTP est la garantie de non déséquencement des POC-PDUs échangées dans chaque sens de l'association. Cependant, chacun d'eux peut être utilisé en mode fiable ou non fiable, par désactivation ou activation du mode *NOERR*, ce mode d'opération pouvant être sélectionné par contexte ou par paquet.

En cas d'activation du mode *NOERR* sur $XTP_{(E->R)}$, l'entité POC réceptrice peut ne pas recevoir toutes les POC-DT émises. De son point de vue, l'ordre de réception des PDUs est donc différent de l'ordre d'émission dès qu'un paquet DATA (au moins) est perdu. Il lui est alors possible de gérer l'ordre partiel monomédia de la connexion comme illustré (Figure 5.18).

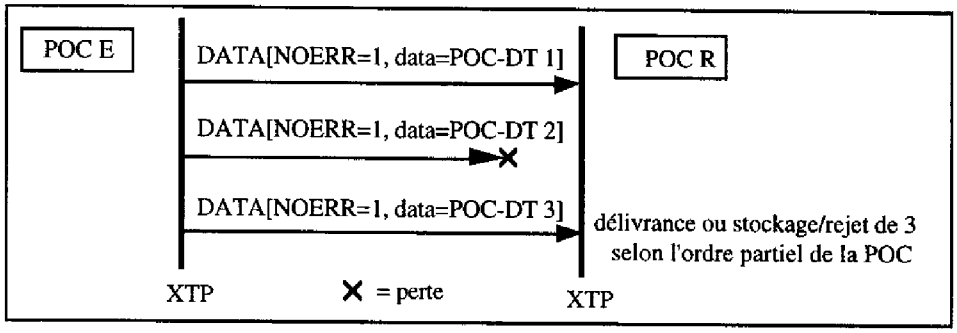


Figure (5.18) : Utilisation de XTP en mode non fiable

Les choix d'architecture énoncés au paragraphe (II.2) conduisent alors à utiliser $XTP_{(E->R)}$ en mode non fiable (*NOERR*=1) dès que l'ordre partiel monomédia de la POC supportée est différent de l'ordre total. Nous verrons comment utiliser $XTP_{(R->E)}$ dans le cadre du mécanisme de gestion de la fiabilité partielle.

Le fait d'autoriser XTP à délivrer un flux d'octets "troué" présente cependant l'inconvénient suivant.
Même si la version 3.6 du protocole prévoit de notifier les occurrences d'octets manquants, on peut d'ores et déjà affirmer que cette notification ne permettra pas à l'entité POC réceptrice d'identifier les POC-DTs manquantes ou incomplètement reçues : cette affirmation résulte d'une part de l'inaptitude de XTP à identifier les messages

véhiculés, et d'autre part de la taille variable de ces messages, qui interdit en réception toute déduction sur la nature des POC-DT perdues, y compris sous l'hypothèse qu'elles ont été émises dans l'ordre séquentiel. Il apparaît donc nécessaire de distinguer les POC-DT au moyen d'un champ particulier, *ident*, que nous situerons dans l'en-tête des POC-DT (Figure 5.19).

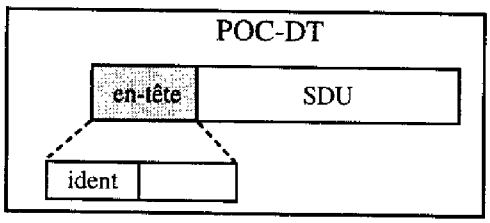


Figure (5.19) : En-tête des PDUs POC-DT

Remarquons qu'une implantation du mécanisme de gestion de l'ordre partiel telle que celle proposée dans la spécification Estelle du protocole POC, conduit à définir le champ *ident* par un couple identifiant la période à laquelle est relative POC-DT, et son numéro de séquence dans la période considérée.

On peut remarquer qu'une utilisation de XTP en mode datagramme autoriserait le déséquencement des POC-DT, et de ce fait une gestion a priori plus "naturelle" d'un ordre partiel en réception. En réalité, le mode d'utilisation de XTP que nous proposons est uniquement problématique dans le cas d'une connexion d'ordre partiel fiable ; en effet, si l'utilisation de XTP en mode fiable semble s'imposer, elle inhibe cependant toute possibilité de tirer partie de l'ordre partiel de délivrance : savoir s'il est préférable dans ce cas d'utiliser XTP en mode non fiable, et de gérer les retransmissions au niveau POC est une question dont la réponse ne peut être donnée que par des mesures de performances comparatives.

II.3.2.2. Gestion de l'ordre partiel inter-flux

La gestion de l'ordre partiel inter-flux n'impose aucune condition d'utilisation particulière de chacun des n flux de données XTP(E->R) dans la mesure où XTP assure la délivrance des données sans prendre en compte un quelconque ordre (total ou partiel) entre les différents contextes instanciés en émission/réception.

Cependant, nous montrerons en (II.3.2.2) comment utiliser le mécanisme de priorité de XTP (drapeau *SORT*) dans le cadre du mécanisme de gestion de la fiabilité partielle.

II.3.3. Gestion de la fiabilité partielle

Une fois établie, une association XTP offre pour chacun de ses sens un service de transfert de données totalement ordonné, utilisable selon plusieurs modes de fonctionnement.

Parmi les modes sélectionnables, deux concernent directement le contrôle d'erreur :

- le mode *NOCHECK* qui lorsqu'il est activé (*NOCHECK*=1), inhibe le contrôle de validité (*checksum*) du segment de données des paquets DATA échangés ;
- le mode *NOERR* qui lorsqu'il est activé (*NOERR*=1), inhibe le mécanisme de reprise des paquets DATA perdus dans le sens considéré de l'association.

II.3.3.1. Contrôle des erreurs bit

Le mode *NOCHECK* permet de décharger le protocole POC du contrôle d'altération des POC-SDUs, et de le laisser aux soins de XTP. Cependant, si le service requis autorise la délivrance de données altérées, la désactivation du total de contrôle de XTP peut s'avérer dangereuse car elle engendre un risque d'altération de l'en-tête des POC-DT. Pour remédier à ce problème, deux possibilités sont envisageables : la première consiste à activer le contrôle d'erreur de XTP pour tous les paquets DATA véhiculant l'en-tête d'une POC-DT ; la deuxième consiste à effectuer ce contrôle au niveau POC sur l'en-tête des POC-DT : dans ce dernier cas, il est alors nécessaire d'introduire un champ dédié à cet effet dans l'en-tête des POC-DT.

II.3.3.2. Contrôle des pertes

Le principe même de la stratégie de délivrance "au plus tôt" repose sur une dégradation de la fiabilité liée à la non récupération des pertes acceptables au regard de la QoS négociée. Indépendamment de l'ordre partiel monomédia de la connexion, l'utilisation du flux $XTP_{(E->R)}$ en mode fiable rend impossible la mise en œuvre de cette stratégie de délivrance ; en effet, la décision de délivrer une SDU au prix de perte(s) acceptable(s) est établie en réception lors du réassemblage des SDUs au niveau POC : une utilisation de XTP en mode fiable engendrerait obligatoirement la mise en œuvre de l'algorithme correspondant sur un flux de données délivré en séquence et sans trou par XTP.
Il apparaît ainsi obligatoire d'utiliser $XTP_{(E->R)}$ en mode non fiable (*NOERR*=1), dès lors que la fiabilité requise par l'utilisateur est différente de la fiabilité totale. Dans les propos suivants, nous nous placerons hors de ce cas particulier.

Afin d'assurer la QoS requise en terme de fiabilité, cette utilisation de XTP impose alors de stocker au niveau POC les POC-DT émises, et de gérer demande d'acquittement, acquittement et retransmission au même niveau ; en effet, lorsque le drapeau *NOERR* d'un paquet DATA est levé en émission, XTP ne stocke pas les données émises en vue d'une éventuelle réémission, d'où la nécessité de conserver celles-ci au niveau POC afin de pouvoir les retransmettre si besoin est.

Deux remarques découlent de cette contrainte :

1 - La retransmission d'une POC-DT ne se traduit pas par une retransmission de niveau XTP, mais par l'émission de nouveaux paquets DATA ; en conséquence les données véhiculées dans ces paquets sont traitées dans leur ordre de soumission à XTP (c'est-à-dire suivant une politique FIFO), sans qu'il soit possible de leur attribuer une priorité de traitement. Ceci résulte du fait que le mécanisme de gestion de priorité de XTP ne s'applique pas aux données d'un même contexte ; en revanche, ce mécanisme permet d'envisager le traitement prioritaire d'une retransmission de niveau POC_i, par rapport au traitement d'une émission de niveau $POC_{j/(j\neq i)}$. De façon analogue, les données d'un paquet DATA contenant une retransmission de niveau POC ne sont mises à disposition de l'entité POC réceptrice qu'après délivrance par XTP de toutes les données précédemment reçues, y compris lorsque celles-ci sont antérieures à la retransmission ; l'entité POC réceptrice est donc contrainte de stocker les POC-SDUs correspondantes jusqu'à réception des données retransmises.

2 - En outre, on met ici en évidence deux niveaux de stockage en émission : au niveau POC pour les données émises non acquittées, et au niveau XTP pour les données soumises et en attente de traitement. Cette remarque fait apparaître la taille des tampons de stockage associés au contexte XTP en émission comme un paramètre critique vis-à-vis des performances : une bonne implantation devra donc déterminer la taille optimale de ces tampons permettant un fonctionnement à "flux tendu".

Gestion de la fenêtre d'émission

Le stockage des données en émission étant dédié au niveau POC, il est nécessaire de gérer le contrôle de la fenêtre d'émission au même niveau. Au regard du mécanisme que nous avons choisi en (II.1.1.2), nous sommes alors conduits à introduire un autre champ d'information, *ack_req*, dans l'en-tête des POC-DT, indiquant au récepteur qu'un acquittement est réclamé (Figure 5.20).

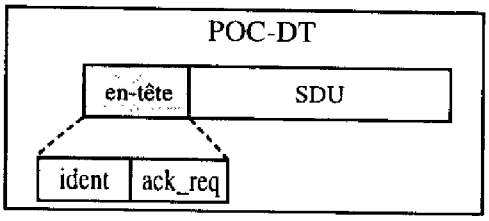


Figure (5.20) : En-tête des PDUs POC-DT

Remarquons qu'en dépit de l'analogie entre ce champ et le champ *SREQ* des paquets XTP, il n'est pas possible d'utiliser *SREQ* pour réclamer un acquittement de niveau POC (il s'agit en effet d'acquitter les POC-DT et non les paquets DATA).

Tel que nous avons décrit le mécanisme de contrôle des pertes du protocole POC, l'activation du champ *ack_req* exprime de la part de l'émetteur soit une demande d'acquittement intermédiaire, soit la dernière demande avant fermeture de la fenêtre d'émission (lorsque les tampons mémoire de l'émetteur sont pleins).

Dans le premier cas, il n'est pas nécessaire de fiabiliser le transfert de la PDU, sa perte n'engendrant pas le blocage de l'émetteur. En revanche, il est obligatoire dans le deuxième cas de figure que la POC-DT(*ack_req*) émise arrive correctement à destination ; deux possibilités sont envisageables : la gestion d'un temporisateur de retransmission au niveau POC, ou l'utilisation de XTP en mode fiable, par désactivation du mode *NOERR* ; cette deuxième possibilité, illustrée (Figure 5.21), permet de tirer partie de la modularité du mécanisme de contrôle des pertes de XTP.

Afin de simplifier le chronogramme représenté, nous supposons qu'un seul paquet DATA permet de transférer la POC-DT considérée. L'activation du bit *SREQ* impose la mise en œuvre par XTP d'un mécanisme d'acquittement sélectif positif du paquet DATA : à l'expiration du temporisateur *WTIMER* associé au paquet DATA émis, XTP_E entame une phase de synchronisation avec XTP_R en émettant un paquet CNTL dont le bit *SREQ* a pour valeur 1 ; le paquet DATA est retransmis si le paquet CNTL reçu en retour permet de conclure à la perte des octets de données véhiculés dans le paquet.

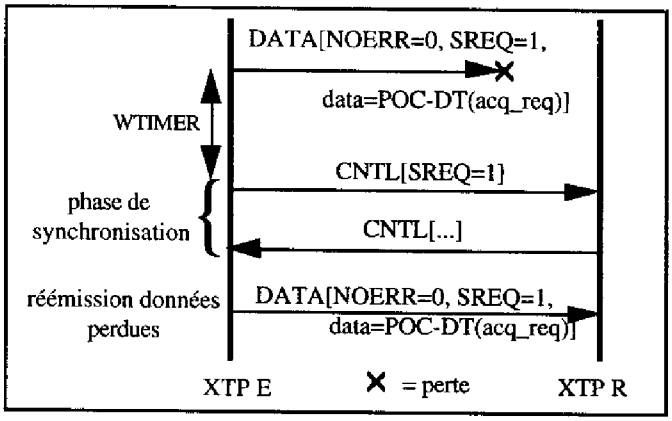


Figure (5.21) : Émission d'une POC-DT avec réclamation d'acquittement "ultime"

En disposant ainsi de la modularité de XTP, il est clair que l'on simplifie la spécification du protocole POC. Soulignons cependant que le changement de mode du contrôle des pertes nécessite au préalable une synchronisation entre sous-systèmes XTP communiquants ; cette procédure peut engendrer un retard incompatible avec des contraintes temporelles strictes (une synchronisation ayant un coût en temps au moins égal au RTT) ; le principe des demandes d'acquittements intermédiaires a pour intérêt de rendre cette situation exceptionnelle.

Gestion des acquittements et des retransmissions

Le rôle d'une POC-CTL est de permettre au récepteur du paquet de mettre à jour sa fenêtre d'émission et de retransmettre, s'il y en a, toutes les POC-DTs réclamées par l'émetteur. Ces PDUs sont générées de deux façons : à l'issue d'une demande explicite de la part de l'émetteur, ou suite à la détection en réception d'un nombre inadmissible de pertes au regard de la QoS fiabilité. Dans ce dernier cas, l'entité POC réceptrice ne délivre aucune des données reçues tant que les pertes inadmissibles n'ont pas été récupérées : ceci souligne la nécessité que les POC-CTL parviennent à destination.

L'un des intérêts de XTP réside dans la possibilité d'utiliser chaque sens de l'association suivant des modes différents. Pour chaque POC, nous proposons de véhiculer les POC-CTL par le biais du flux de données de R vers E de l'association XTP ; afin de fiabiliser le transfert de ces PDUs, une façon simple consiste à utiliser XTP(R->E) en mode fiable (*NOERR*=0) et sans erreur (*NOCHECK*=0) (Figure 5.22).
Outre le fait qu'elle permet de simplifier le mécanisme de gestion de la fiabilité au niveau POC, cette utilisation de XTP évite d'avoir à définir un en-tête aux POC-CTL : en effet, il n'est pas nécessaire d'identifier les POC-CTL au niveau POC, mais simplement de les délimiter ; cette possibilité est offerte par XTP en activant le bit *EOM* de l'en-tête du paquet DATA véhiculant les derniers octets d'une POC-CTL.

Remarque : on peut faire l'hypothèse qu'un seul paquet DATA suffira à transmettre une POC-CTL dans son ensemble.

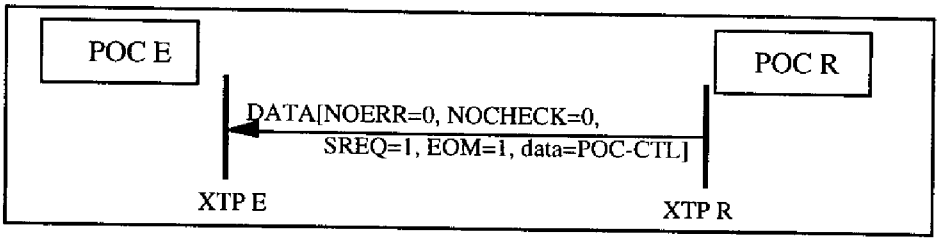


Figure (5.22) : Émission d'une POC-CTL

Notons que l'activation du bit SREQ impose la mise en œuvre par XTP d'un mécanisme d'acquittement sélectif positif des paquets DATA émis, adapté au transfert des POC-CTL dans la mesure où ces dernières sont émises sporadiquement, et doivent arriver à destination.

Gestion des retransmissions

Compte tenu de la sémantique de service choisie en terme de fiabilité, les POC-DT à retransmettre doivent impérativement parvenir à destination. La fiabilisation du transfert de ces PDUs, envisageable au niveau POC, peut être laissée aux soins de XTP en

paramétrant les paquets DATA comme illustré (Figure 5.23) ; afin de simplifier le chronogramme représenté, nous supposons là encore qu'un seul paquet DATA suffit à transférer une réémission. En fiabilisant au niveau XTP le transfert des POC-DT retransmises, l'entité POC émettrice peut alors libérer au plus tôt les tampons mémoire dans lesquels elles étaient stockées, et mettre à jour en conséquence sa fenêtre d'émission. Là encore, la modularité de XTP permet de simplifier le protocole POC.

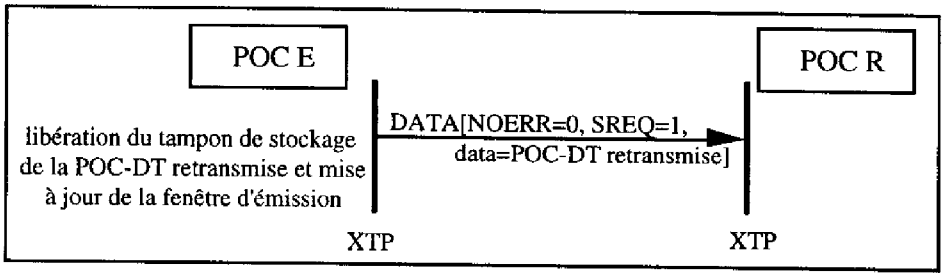


Figure (5.23) : Émission d'une retransmission

Remarque

Deux cas particuliers permettent de simplifier la gestion de la fiabilité en supprimant stockage, demande d'acquittement et acquittement de niveau POC.

1- Fiabilité nulle et ordre quelconque

Lorsque la fiabilité requise est nulle (aucune perte n'est récupérée), les POC-DT ne sont ni stockées en émission, ni acquittées en réception.

2 - Fiabilité totale et ordre total

Lorsque l'ordre et la fiabilité requis sur une POC sont l'ordre total et la fiabilité totale (aucune perte n'est tolérée), il est alors possible de décharger le protocole POC de ses mécanismes de contrôle des pertes, en utilisant $XTP_{(E->R)}$ en mode fiable. En conséquence, le flux de données de R vers E de l'association XTP n'est pas utilisé, et les POC-SDUs sont délivrées à l'utilisateur au fur et à mesure de leur réassemblage.

II.3.4. Délimitation des SDUs - Segmentation et réassemblage

Dés le début de cette étude, nous avons évoqué la possibilité offerte par XTP de délimiter les PDUs de niveau POC, sans toutefois pouvoir les identifier. Jusqu'à présent, nous avons toujours supposé qu'un seul paquet DATA suffisait à véhiculer une POC-PDU, sans envisager une opération de segmentation et de réassemblage : si cette hypothèse apparaît plausible quant au transfert des POC-CTL, elle n'est pas valable dans le cadre du transfert des POC-DT.

Dans ce paragraphe, nous étudions comment délimiter les POC-DT (une POC-DT véhiculant une POC-SDU complète), en tenant compte de leur éventuelle segmentation/réassemblage.

Concernant la délimitation des POC-SDUs dans le flux de données XTP(E->R), plusieurs mécanismes sont envisageables ; nous proposerons un mécanisme basé sur l'utilisation combinée des drapeaux *EOM* et *BTAG* des paquets XTP.
Au préalable, rappelons que XTP impose une limite de taille au segment de données de ses paquets alors que le service MM-POS ne limite pas la taille des POC-SDUs (leur taille maximale étant cependant spécifiée par l'utilisateur) ; en conséquence, le mécanisme de segmentation et le réassemblage des POC-SDUs est dédié au niveau POC.
Une POC-SDU constituant l'intégralité de la partie données des POC-DT, il s'agit donc d'envisager la segmentation et le réassemblage des POC-DT dont le format fait apparaître deux parties distinctes, un segment d'en-tête et un segment de données.

II.3.4.1. Délimitation des SDUs

La vocation première du champ *BTAG* d'un paquet DATA est de spécifier (lorsqu'il est activé) le caractère particulier des huit premiers octets du segment de données.
Nous proposons une utilisation de ce champ à double titre, d'une part pour véhiculer l'en-tête des POC-DT (dont on peut légitimement supposer la taille fixe et inférieure à huit octets) en tant que données marquées, et d'autre part pour indiquer le début d'une POC-DT dans le flux de données XTP. Recevant des données "marquées", l'entité POC réceptrice peut ainsi traiter l'en-tête (de niveau POC) contenu dans les huit premiers octets, et entamer parallèlement (si c'est utile) le réassemblage de la POC-SDU.
L'activation du champ *EOM* du dernier paquet DATA émis est suffisante pour indiquer en réception la fin d'une POC-DT.
Tout paquet DATA présentant les drapeaux *BTAG* et *EOM* baissés contiendra donc des données intermédiaires d'une POC-DT. Un paquet DATA véhiculant une POC-DT dans son intégralité, aura les drapeaux *BTAG* et *EOM* de son en-tête levés.

II.3.4.2. Segmentation et réassemblage

Soit *maxdata* la taille maximale autorisée du segment de données des paquets DATA.
Soit T la taille d'une POC-SDU soumise en émission, et A la POC-DT véhiculant cette SDU.
Deux cas de figure sont envisageables selon que T est inférieure ou non à (*maxdata* - 8).

• Si ($T \leq$ *maxdata* - 8) alors A peut être transmise en un seul paquet DATA (Figure 5.24) :
- l'entité POC émettrice requiert l'activation des champs *EOM* et *BTAG* : un seul paquet DATA est émis en conséquence.
- en réception, l'entité POC récupère les informations de contrôle contenues dans les huit octets marqués et traite en conséquence la POC-SDU codée dans le restant des octets délivrés.

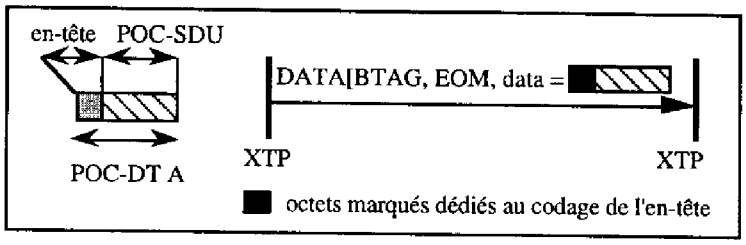


Figure (5.24) : Segmentation/réassemblage (1)

Remarquons que la perte d'un paquet DATA engendre la perte de toute la POC-SDU véhiculée.

- Si (T > *maxdata*) alors le transfert de A nécessite l'émission d'au moins deux paquets DATA. D'une façon générale, T peut être exprimée sous la forme :

$$T = (maxdata - 8) + k * maxdata + R \text{ avec } R < maxdata \text{ et } k \in N ;$$

pour simplifier l'illustration du protocole, supposons $k = 1$ et étudions séparément le comportement des entités POC émettrice et réceptrice (Figure 5.25).

Entité POC émettrice

1 - L'entité POC émettrice requiert l'émission avec activation du champ *BTAG* de l'en-tête de A en tant que données marquées, et des (*maxdata* - 8) premiers octets de données de A en tant que données normales ;
2 - elle invoque ensuite l'émission des *maxdata* octets de données suivants de A, sans activer le champ *BTAG* ;
3 - elle requiert enfin l'émission des R derniers octets de A, toujours sans activer le champ *BTAG*, mais en notifiant une fin de message : le bit *EOM* est donc levé dans l'unique paquet DATA émis en conséquence.

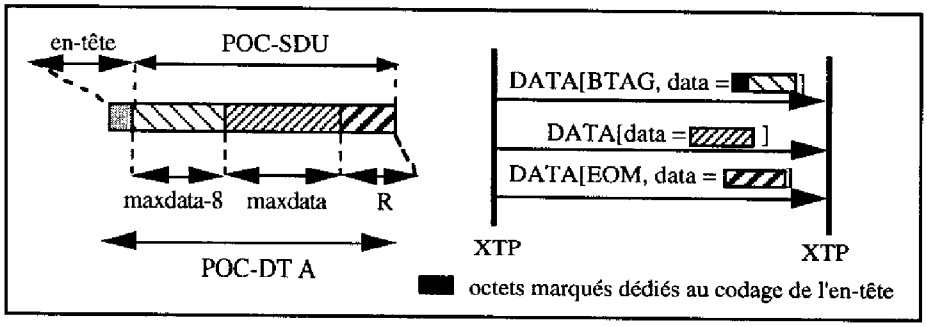


Figure (5.25) : Segmentation/réassemblage (2)

Entité POC réceptrice

Lorsque le flux de données XTP(E->R) est utilisé en mode non fiable (*NOERR*=1), les paquets DATA perdus ne sont pas récupérés par XTP : il est alors possible que l'entité

POC réceptrice ne reçoive pas les POC-DT dans leur intégralité, et ne puisse pas réassembler correctement les POC-SDUs. Dans le protocole que nous proposons, la perte d'un paquet DATA engendre la perte de toute la PDU.

En fonctionnement normal du réseau, les trois paquets DATA émis (Figure 5.25) arrivent correctement à destination, et l'entité POC réceptrice réassemble la POC-SDU au fur et à mesure de la délivrance des données par XTP.
La norme 3.6 du protocole stipule que XTP, lorsqu'il est utilisé en mode non fiable, doit cependant notifier à son utilisateur toute occurrence de pertes ; le mécanisme correspondant n'est pas défini, mais il est cohérent d'envisager que cette notification accompagne la délivrance des données qui suivent le "trou" détecté. Cette fonctionnalité évite d'avoir à spécifier la longueur de la POC-SDU véhiculée dans un champ particulier de l'en-tête des POC-DT.

Pour illustrer ces propos, envisageons le cas d'une perte en supposant que l'entité POC réceptrice ait terminé le traitement de la dernière POC-DT reçue, et soit en attente de la POC-DT *A* (elle attend donc des données marquées). Trois cas de figure se présentent selon que le paquet DATA perdu véhicule le début, le milieu ou la fin de *A* :

- cas 1 : si les données perdues sont celles du premier paquet DATA, l'entité POC réceptrice rejette toutes les données qui lui sont délivrées par XTP, jusqu'à délivrance de données marquées identifiant le début d'une autre POC-DT ; elle n'évalue l'acceptabilité de la perte qu'à ce moment là ;
- cas 2 : si les données perdues correspondent aux données intermédiaires, l'entité POC réceptrice stoppe le réassemblage de la SDU en cours, et en réclame sa retransmission si la perte occasionnée est inacceptable ; elle rejette les données suivantes jusqu'à délivrance d'un nouvel en-tête ;
- cas 3 : si les données perdues correspondent à des données de fin de message, l'occurrence de perte sera notifiée par XTP lors de la délivrance de l'en-tête de la POC-DT suivante ; l'entité POC réceptrice stoppe alors le réassemblage de la SDU en cours, en réclame sa retransmission si la perte occasionnée est inacceptable, et entame le réassemblage de la SDU suivante.

Remarquons cependant que dans le cas particulier où sont perdus consécutivement un paquet DATA de fin de message (*EOM*=1) et un paquet DATA marqué (*BTAG*=1), le comportement de l'entité POC réceptrice est identique à celui du cas 1 précédent.

Remarque
Dans l'hypothèse d'une gestion de la fiabilité par groupe de connexions, le réassemblage d'une POC-SDU n'est pas entamé ou est interrompu si la POC-SDU est déclarée perdue ;

dans ce cas, les données suivantes reçues sont rejetées, et l'entité POC réceptrice se met en attente de données marquées.

II.4. Conclusions de l'étude

Les conclusions que nous donnons à cette étude sont de deux types :

- le premier est favorable à XTP du fait que la modularité de ses mécanismes permet d'envisager la mise en œuvre d'une MM-POC et de simplifier le protocole mis en œuvre dans chacune des POC (1) ;
- le second est défavorable à XTP car son modèle de fonctionnement (orienté octets) complique le protocole POC (définition de champs d'information supplémentaires), et interdit toute prise en compte prioritaire des données les unes par rapport aux autres, dans le cadre d'un même contexte XTP. Nous donnerons une interprétation de ce dernier aspect en fin de conclusion afin de dégager les perspectives d'évolution du protocole (2).

(1) Tout d'abord, nous avons montré qu'il était possible d'envisager l'implantation d'une MM-POC à l'aide de XTP, offrant une qualité de service en termes d'ordre et de fiabilité ; cette conclusion est importante car elle répond à l'objectif premier de notre démarche. En particulier, on doit souligner l'intérêt du caractère bidirectionnel d'une association XTP par le fait qu'il autorise pour chaque sens la sélection de modes d'opération différents ; cette possibilité permet d'envisager une simplification du protocole POC vis-à-vis de la gestion des acquittements, en requérant une fiabilité totale du flux de retour de XTP ($XTP_{(R->E)}$), et en fiabilisant sélectivement le flux $XTP_{(E->R)}$ selon les caractéristiques des POC-DT à transférer (demandes d'acquittement impératives et retransmissions)

(2) Cependant, plusieurs points critiques ont été mis en évidence que l'on peut résumer comme suit.
Rappelons au préalable que la contradiction majeure à laquelle nous avons été confrontés résulte de l'opposition des modes de fonctionnement des protocoles POC et XTP. Les mécanismes du protocole POC reposent sur la manipulation de PDUs (POC-DT) identifiant chacune un objet utilisateur (i.e. une POC-SDU) ; à l'opposé, XTP est bâti sur un modèle identifiant les octets de données, sans se préoccuper de la notion d'objet. Alors qu'une utilisation de XTP en mode fiable (telle que celle du flux $XTP_{(R->E)}$) permet d'identifier les POC-PDUs en les délimitant à l'aide du bit *EOM* des paquets XTP, la désactivation de son contrôle des pertes (*NOERR*=1) supprime cette possibilité dans la mesure où le POC récepteur ne peut évaluer ni le nombre, ni la nature des pertes advenues.

Dans le cadre de mise en œuvre de XTP que nous nous sommes fixés (l'établissement d'une association XTP par POC monomédia), nous avons dégagé l'obligation d'utiliser XTP en mode non fiable pour le transfert des POC-SDUs, afin d'autoriser une gestion de l'ordre et de la fiabilité (exception faite des cas particuliers "ordre total/fiabilité totale" et "ordre quelconque/fiabilité nulle") ; ceci résulte du fait que XTP en mode connecté fournit un service de délivrance des données exempt de tout déséquencement. Conséquences de cette obligation, les deux écueils soulevés dans cette étude résultent du stockage en émission des données utilisateur au niveau POC, stockage nécessaire pour assurer la QoS fiabilité du service de Transport :

- compte tenu du rappel précédent, cette contrainte impose tout d'abord que les POC-SDUs soient encapsulées dans un en-tête de niveau POC (l'ensemble définissant une POC-DT), avant d'être véhiculées dans le segment de données des paquets DATA de XTP ; dans le cadre du mécanisme de contrôle des pertes proposé, cet en-tête est constitué de deux champs, *ident* et *ack_req* ; quel que soit le mécanisme, on peut remarquer que l'identification des POC-DT est nécessaire pour que l'entité POC réceptrice puisse distinguer deux PDUs. Nous soulignons ici que le protocole XTP ne permet pas la mise en œuvre d'une MM-POC en n'utilisant que les champs d'informations des paquets XTP ;
- le deuxième écueil résulte de l'impossibilité d'accorder une priorité de traitement "intra-contexte" aux retransmissions de niveau POC (seule une priorité "inter-contexte" est envisageable par utilisation du drapeau *SORT*). En conséquence, la retransmission d'une POC-DT n'est prise en compte en émission par XTP, qu'après traitement des POC-DT préalablement soumises, alors qu'il est d'ores et déjà certain que ces dernières ne seront pas délivrables en réception (l'entité POC réceptrice étant bloquée en attente de réception de la POC-DT retransmise) ; de façon analogue, XTP ne permet pas d'envisager la délivrance prioritaire d'une retransmission de niveau POC, avant celle de toutes les données qui la précèdent dans le contexte XTP auquel elle est relative : l'entité POC réceptrice est donc contrainte de stocker ces PDUs jusqu'à ce qu'elle reçoive la POC-DT retransmise ; la délivrance prioritaire de cette donnée conduirait à diminuer le délai de transit de la connexion.

On doit cependant remarquer que ce dernier point ne constitue pas une faille du protocole au regard de sa logique de conception : il reflète simplement le fait que XTP repose sur les modes CO et CL traditionnels, auxquels vient s'ajouter la possibilité d'activer ou de désactiver le contrôle des pertes ; en d'autres termes, comme l'ensemble des protocoles existants, XTP n'est pas conçu pour supporter un service d'ordre et de fiabilité partielle.

Cette dernière remarque nous conduit alors à envisager comme perspective d'évolution de XTP l'intégration en son sein du concept de POC ; en cela, cette conclusion rejoint les

développements présentés au chapitre 3 de ce mémoire. L'un des champs d'information du protocole, le champ *service*, situé dans le segment d'adresse du paquet FIRST initiant une association XTP, indique le type de service attendu par l'utilisateur ; ce champ permet d'envisager la possibilité d'étendre la logique du protocole, en vue d'y intégrer le concept de connexion d'ordre partiel et de fiabilité partielle. Cependant, cette extension n'est envisageable qu'à la condition que XTP intègre la notion de message dans ses mécanismes, le bit *EOM* étant insuffisant à cet effet. Nous pensons que cette étape constitue le préambule à une intégration du concept de POC dans le protocole XTP.

Conclusion

Ce chapitre a tout d'abord présenté le représentant le plus évolué d'une nouvelle génération de protocoles de transfert, le protocole XTP. XTP offre à ses utilisateurs des procédures de transfert de données paramétrables, leur permettant de disposer à volonté d'une grande partie des fonctionnalités des protocoles de Transport traditionnels. En outre, de par le format de ses PDUs et ses principes de conception, XTP présente un caractère haute vitesse marqué, adapté à des environnements de communication à haut débit. L'un des points clef du protocole repose sur la modularité de son contrôle d'erreur ; c'est au regard de ce dernier aspect que nous avons envisagé la possibilité de mettre en œuvre une MM-POC à l'aide de XTP, offrant une qualité de service en terme d'ordre et de fiabilité telle que celle définie dans le service MM-POS.

L'étude présentée dans la deuxième section de ce chapitre a eu pour objectif de proposer une utilisation de XTP permettant de mettre en œuvre un service d'ordre et de fiabilité partiels. Les mécanismes principaux du protocole d'ordre partiel ont tout d'abord été décrits. Dans un second temps, nous avons présenté l'architecture de Transport retenue pour étudier le paramétrage de XTP. L'étude menée dans ce contexte nous a permis de conclure à la possibilité d'implanter une MM-POC à l'aide de XTP, en soulignant cependant les limites protocole. Ne disposant pas d'un support XTP, nous n'avons pas pu mettre en pratique les développements de cette étude. Cette réalisation nous semble nécessaire afin de donner un prolongement concret à nos travaux.

CONCLUSION GÉNÉRALE

Dans ce mémoire, nous avons présenté les travaux effectués autour de la définition d'un nouveau type de connexion : la connexion d'ordre partiel et de fiabilité partielle. Située dans le cadre de la recherche de nouveaux services et protocoles de Transport multimédia, notre démarche a consisté à définir et caractériser ce nouveau concept, puis à l'intégrer dans une architecture de Transport multimédia, offrant une qualité de service à la fois monomédia et multimédia, exprimable en terme d'ordre et de fiabilité. Nous rappelons maintenant les principales contributions exposées dans ce mémoire, avant de dégager les perspectives d'évolution de ces travaux.

Rappel des contributions

S'il est à présent admis que les fonctionnalités des protocoles de Transport tels que TCP doivent être révisées, plusieurs approches sont envisageables quant à leur spécification future ; dans l'optique de permettre des transferts à haut débit, l'une des approches les plus évoluées à l'heure actuelle est représentée par XTP, autorisant son utilisateur à paramétrer les mécanismes du protocole selon les caractéristiques du flux à supporter. C'est également dans ce contexte que nous avons situé le premier point de notre contribution, en proposant la définition de fonctionnalités nouvelles pour la couche Transport, relatives à l'ordre et la fiabilité du transfert de données.

• ***Un concept nouveau : la connexion d'ordre partiel et de fiabilité partielle***

Partant du constat que les services de transfert de données actuels n'offraient comme garantie d'ordre et de fiabilité que la seule alternative (ordre total et fiabilité totale/ni ordre, ni fiabilité), nous avons proposé une nouvelle approche de conception d'un service/protocole de Transport, permettant de définir et de garantir une qualité de service exprimable en termes d'ordre et de fiabilité.

Le premier point de notre contribution (chapitre 3) a donc porté sur la définition d'un nouveau concept de connexion, *la connexion d'ordre partiel et de fiabilité partielle (POC)*, permettant de définir et de mettre en œuvre tous les services et protocoles d'ordre partiel (respectivement de fiabilité partielle) compris entre l'ordre total et le désordre

(respectivement la fiabilité totale et la fiabilité nulle). En autorisant son utilisateur à sélectionner le protocole le plus proche de ses contraintes applicatives en terme d'ordre et de fiabilité, le concept de POC établit un lien jusque là inexploité entre application et protocole ; de plus, il permet d'étendre et d'intégrer toutes les approches développées jusqu'à présent, telles que TCP, UDP et plus récemment XTP.

Au delà de son intérêt conceptuel, nous avons montré l'adéquation d'une connexion d'ordre partiel à exploiter le degré de parallélisme d'un document multimédia, tout en satisfaisant les contraintes minimales de l'utilisateur. Les bénéfices de cette exploitation ont été évalués de façon qualitative en terme de mémoire et de bande passante, mais également en terme de délai de transit.

• *Une Architecture de Transport multimédia à connexions d'ordre partiel*

Outre de hauts débits et des temps de réponse bornés, les contraintes applicatives les plus problématiques résultent de la nature même des flux multimédia, et consistent en la nécessité de synchroniser les différents média lors de leur restitution finale (problématique de la synchronisation multimédia).

Dans ce contexte, l'un des thèmes de réflexion parmi les plus ouverts à l'heure actuelle concerne la conception de nouvelles architectures de communication multimédia, intégrant la prise en compte des exigences temporelles et de la synchronisation multimédia à différents niveaux.

Dans la majorité des grands projets menés dans ce domaine tels que QoS-A, BERKOM, CIO, cette démarche a en particulier conduit à une évolution de la notion de qualité de service Transport. En règle générale, cette évolution se traduit par l'ajout de paramètres de service à caractères temporels, et par la définition de nouvelles sémantiques de service ; en cela, elle s'inscrit dans le cadre de la proposition OSI 95, visant à attribuer à la couche Transport un rôle de gestionnaire de la QoS, telle qu'elle est en fait fournie par le réseau sous-jacent. Cependant, outre le fait que la mise en œuvre d'une telle QoS nécessite l'élaboration de mécanismes complexes et des conditions d'implantation très spécifiques, la caractéristique majeure de cette démarche est qu'elle est ne permet pas la prise en compte, au niveau Transport, de la synchronisation multimédia. En d'autres termes, les services de Transport envisagés suivant ce point de vue sont purement monomédia.

Partant de ce constat et des premiers résultats de nos travaux, la deuxième partie de notre contribution (chapitre 4) a consisté à intégrer le concept de connexion d'ordre partiel dans le cadre d'une architecture de Transport destinée à supporter le transfert des flux multimédia ; ces travaux nous ont conduit à définir les principes d'implantation d'une connexion de Transport multimédia d'ordre partiel (MM-POC), impliquant l'instanciation

et la coordination de plusieurs connexions de Transport monomédia, chacune à qualités de services propres ; l'ordre et la fiabilité y apparaissent en tant que paramètres de service à la fois monomédia et multimédia.

Dans cette architecture, la prise en compte de l'ordre partiel se situe à deux niveaux : dans chaque connexion et entre les connexions, permettant ainsi de définir tous les services et protocoles d'ordre partiel multimédia, tels qu'il se déduisent des contraintes de synchronisation de l'application exprimées à l'aide d'un modèle tel que le TSPN ; en cela, ce service étend le lien établi entre application et protocole grâce au concept de POC, en autorisant la prise en compte de la notion d'objet multimédia au niveau Transport. L'intégration de la notion de fiabilité partielle dans cette architecture nous a conduit à définir deux mécanismes de contrôle d'erreur : *connexion par connexion* et par *groupe de connexions*, autorisant une gestion monomédia et/ou multimédia de la fiabilité, et induisant chacun une diminution du temps de transit des données au prix d'une dégradation acceptable de la fiabilité. Ceci renforce l'intérêt potentiel d'une MM-POC, qui au travers du service offert en terme d'ordre et de fiabilité, effectue au sens exact du terme un *best effort* vis-à-vis de l'amélioration du délai de transit.

Au terme de la présentation des principes d'implantation d'une MM-POC, nous avons défini les primitives ainsi que les paramètres du service correspondant (MM-POS). Les seules garanties de ce service sont relatives à l'ordre et la fiabilité ; le choix de ne pas garantir les paramètres à caractère temporel permet de simplifier la conception du protocole comparativement à une approche intégrant la garantie de ces paramètres. Cette décision est justifiée en pratique car elle tient compte de la nature majoritairement asynchrone des systèmes d'exploitation actuels.

• ***XTP, le support de Transport adéquat d'une MM-POC ?***

Le dernier chapitre du mémoire a tout d'abord porté sur une présentation de XTP, protocole dont les mécanismes sont conçus afin de permettre des transferts rapides sur des réseaux à haut débit et présentant un faible taux d'erreur. L'intérêt majeur de XTP réside dans le fait qu'il autorise son utilisateur à paramétrer ses mécanismes principaux, et de ce fait à mettre en œuvre le protocole selon différents modes. C'est de ce point de vue que nous avons envisagé l'étude présentée dans ce chapitre, visant à proposer une utilisation de XTP assurant la mise en œuvre d'une connexion multimédia d'ordre partiel. Cette étude a démontrée la faisabilité de l'implantation d'une MM-POC à l'aide des procédures XTP ; cependant, elle a conduit à souligner les limites de ce protocole vis-à-vis de la prise en compte du concept de connexion d'ordre partiel.

Ce dernier point, prévisible du fait de la différence entre les notions de base (en terme d'ordre et de fiabilité) dans l'approche classique et l'approche conduisant à définir une POC, souligne qu'une implantation "optimale" doit prendre en compte les notions d'ordre et de fiabilité partiels au niveau le plus fondamental. Cependant, il est raisonnable d'envisager une implantation performante et moyennement complexe d'un protocole d'ordre partiel à l'aide des fonctions offertes par XTP selon les principes que nous avons définis.

Perspectives

Nous terminons ce mémoire en dégageant les perspectives de recherche consécutives à ces travaux.

- ***Exploitation des résultats***

Évaluation de performance et Implantation

L'évaluation des performances d'un protocole multimédia d'ordre partiel repose sur des mesures effectuées soit à partir d'une implantation réelle, soit par utilisation de modèles tels que les Réseaux de Petri temporisés stochastiques ou d'outils logiciels de simulation tel que OPNET.

Un prototype d'implantation d'un protocole multimédia d'ordre partiel est en cours de réalisation au LAAS. L'évaluation des performances du protocole constitue l'une des contributions d'une thèse de doctorat actuellement menée dans le groupe OLC. A moyen terme, il est envisagé d'étendre ces travaux par l'implantation d'une MM-POC sous forme de *driver*, afin d'instancier autant de connexions multimédia d'ordre partiel qu'il y a d'applications à supporter.

Impact sur une application multimédia

L'application illustrative retenue dans le projet CESAME a été définie par la société AIRBUS TRAINING. Celle ci assure la formation des personnels intervenant sur les avions produits par la société AIRBUS INDUSTRIE. Les cours sont dispensés en utilisant les concepts de l'enseignement multimédia assisté par ordinateur (EAO). Jusqu'alors, le support d'enseignement était composé de PC situés dans une même salle, mais ne pouvant pas communiquer entre eux ; les travaux réalisés en collaboration avec le CNRS dans le cadre du projet CESAME ont eu pour but d'explorer l'extension de l'application existante à une application de télé-enseignement dans laquelle instructeurs et étudiants ne sont plus regroupés en un même lieu : il est donc nécessaire de connecter les postes de travail via un réseau de communication à haut débit.

L'architecture de communication multimédia définie dans le cadre du projet CESAME vise à supporter la mise en oeuvre d'applications telles que le télé-enseignement. Pour cela, un outil de visioconférence a été implémenté au LAAS, permettant à plusieurs utilisateurs de communiquer par l'intermédiaire de la voix et de l'image. Actuellement, l'architecture développée utilise le service de Transport fourni par UDP. L'impact d'un service de Transport multimédia d'ordre partiel sur l'application visée est donc à évaluer. En particulier, il serait intéressant de tester dans un contexte réel l'intérêt des mécanismes de gestion de la fiabilité partielle (connexion par connexion et par groupe de connexion). Cette réalisation constitue une perspective pratique à donner aux travaux effectués autour du concept de connexion d'ordre partiel et de fiabilité partielle.

- ***Extension des résultats***

Approfondissement du concept de fiabilité partielle

Le concept de fiabilité partielle doit être étendu, en particulier pour faire le lien avec d'autres qualités de service demandées par l'application. Par exemple, l'expression actuelle de la fiabilité partielle ne permet pas de prendre en compte aisément les spécificités d'un flux MPEG composé de trames de nature différente (I, B, P).

Il apparaît prometteur d'étendre la notion de *classe de fiabilité* introduite au chapitre 3, attribuant un paramètre de fiabilité non plus à une connexion donnée, mais à chacun des objets échangés au travers de cette connexion. L'introduction de ces classes au niveau du service accroîtrait la complexité du protocole (les mécanismes de reprise des pertes étant à redéfinir), et l'impact sur le délai de transit de la connexion serait à évaluer.

Spécification de la renégociation

La signification des paramètre 'ordre' et fiabilité' pour l'utilisateur d'un service de Transport multimédia fait qu'il est tout à fait possible d'envisager une modification de leur valeur en cours de communication. Par exemple, une fiabilité nouvelle peut être réclamée sur un flux donné, sans pour autant que la structure du document multimédia soit modifié (l'ordre reste donc inchangé) ; de même, l'utilisateur peut souhaiter relâcher la synchronisation entre deux flux, modifiant ainsi le paramètre ordre de la connexion de Transport utilisée. Pour autant, il n'est pas nécessaire ni souhaitable de rompre la connexion et d'établir une nouvelle connexion ; en d'autres termes, il est utile de pouvoir renégocier les paramètres 'ordre' et 'fiabilité'.

Une poursuite de nos travaux consiste donc à définir les modalités d'une phase de renégociation des paramètres de QoS. Plus précisément, il s'agit tout d'abord d'étendre le service MM-POS en y incluant paramètres et primitives de service nécessaires à une

renégociation de la QoS (une premier ensemble de primitives a été défini en ce sens dans la thèse) ; ensuite, le protocole à mettre en oeuvre doit être spécifié, puis validé dans un langage de description formelle tel qu'Estelle. La difficulté principale dans la phase de renégociation tient à la nécessité de synchroniser d'une part les entités communicantes de niveau Transport , et d'autre part l'utilisateur et le fournisseur du service.

Lors de la présentation du concept de POC, nous mis en évidence le lien entre *ordre partiel* et *modèle* (tel que le TSPN) de la façon suivante : l'utilisateur d'un service multimédia d'ordre partiel dérive la QoS 'ordre' (exprimant les contraintes de synchronisation logique) à partir d'une modélisation TSPN exprimant les contraintes de synchronisation temporelles de l'application. On peut s'interroger sur la nature d'un lien analogue entre renégociation de l'ordre et un modèle d'application plus complexe, le modèle HTSPN [Sénac95], qui étend le modèle TSPN pour prendre en compte les caractéristiques des documents hypermédia. Chaque place du modèle HTSPN est susceptible de modéliser un scénario de synchronisation multimédia dans son ensemble : on entrevoit ici la possibilité qu'une transition du HTSPN marque le début d'une phase de renégociation de la QoS. Il s'agit donc dans en premier temps de définir précisément les relations entre HTSPN et ordre partiel. Dans un second temps, il s'agira d'intégrer ces travaux et ceux concernant la spécification de la renégociation.

• *Nouveaux axes de recherche*

Spécification de nouveaux mécanismes de synchronisation

L'une des perspectives les plus intéressantes au plan théorique résulte de l'approche adoptée quant à la prise en compte de la structuration d'un document multimédia au niveau des protocoles de Transport. Cette approche, visant à concevoir un document multimédia dans sa globalité plutôt que média par média, a permis de dégager une qualité de service relative à l'ordre et la à fiabilité du transfert du document dans son ensemble.

Cette qualité de service constitue une nouvelle approche dans la prise en compte des exigences d'une application multimédia, en particulier de ses contraintes de synchronisation. Jusqu'à présent, les approches envisagées au niveau applicatif étaient majoritairement fondées sur l'hypothèse d'un service de Transport présentant des garanties de qualités de services monomédia en terme de délai de transit, voire de gigue ; cette hypothèse est à la base de la logique suivie à l'heure actuelle quant à la spécification de nouveaux services de Transport.

En garantissant la délivrance des données dans un ordre compatible avec une expression logique des contraintes de synchronisation, un service de Transport multimédia d'ordre partiel permet à l'utilisateur de prendre en compte la synchronisation sur les bases d'une

hypothèse différente. En outre, la garantie d'une fiabilité partielle contribue à diminuer le délai de transit des données, et de ce fait simplifie la tâche de synchronisation de l'utilisateur.

Les techniques de synchronisation temporelle à mettre en oeuvre dans le cadre de cette approche constituent donc un nouvel axe de réflexion, tant du point de vue théorique que pratique, c'est-à-dire en tenant compte des caractéristiques des systèmes d'exploitation actuels, et des limites engendrées par leur asynchronisme.

Architecture

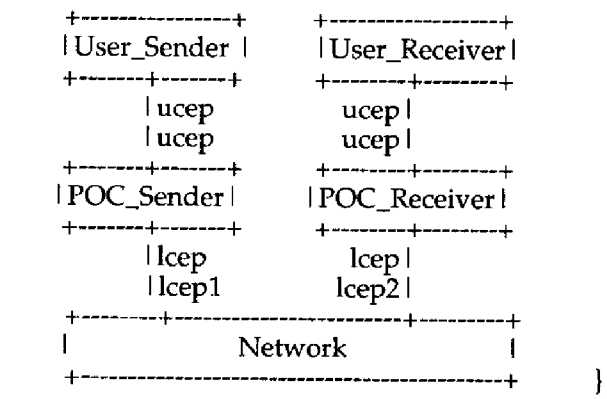
Afin d'en évaluer l'impact dans un environnement réseau à haut débit, les travaux réalisés au LAAS autour de l'implantation d'un protocole multimédia d'ordre partiel reposent sur l'hypothèse d'un environnement réseau ATM.

A l'heure actuelle, l'explosion des services dans l'Internet explique en grande partie l'initiative américaine sur les autoroutes de l'information. Cependant, l'Internet n'impose pas de réseau physique spécifique, que ce soit pour les réseaux locaux ou les liaisons longues distances : il utilisera aussi bien Ethernet ou FDDI qu'ATM. En revanche, la technique de routage actuelle, IP (Internet Protocol), fédère l'ensemble de ces réseaux, et en est la clé de voûte. Ses services de routage et d'adressage entrent donc directement en concurrence avec ceux de l'ATM. L'évolutions d'IP la plus récente, IPnG (IP nouvelle génération), a pour but de permettre la fusion de ces deux espaces de routage et de fonctionner aussi bien sur des réseaux à haut débit tel que l'ATM que sur des réseaux à faible bande passante. IPnG entend non seulement répondre aux exigences actuelles, mais surtout anticiper les besoins futurs notamment liés aux développement des applications multimédia.

En conséquence, l'un des thèmes de recherche qu'il est nécessaire de développer concerne l'aspect architecture, intégrant à présent les fonctionnalités et les services d'IPnG, à la fois à l'ATM et aux logiciels applicatifs. Plus précisément, il s'agit de définir les éléments d'une architecture de communication multimédia, intégrant Transport multimédia à connexions d'ordre partiel, IPnG et ATM. Ensuite, afin de prendre en compte les exigences en débit et les contraintes temporelles des applications multimédia, la mise en oeuvre dans une telle architecture de protocoles à réservation de ressource tels que RSVP devra également être étudiée. Enfin, il sera nécessaire de préciser le concept de qualité de service aux différents niveaux de cette architecture afin de paramétrer dynamiquement tous les services requis.

ANNEXE : Spécification Estelle du protocole POC

```
specification PartialOrderConnection systemactivity;
        default individual queue; timescale millisecond;
```




```
const
    N = ...; { # tsdus in negotiated PO per period}
    NUM_PARTIAL_ORDER_BITS  = (N*(N-1)/2) { # bits to encode partial order  }
    NUM_DATA_PER_N_OR_T_SDU = ...; { info data bytes per nsdu or tsdu }
    NUM_RCV_BUFFERS = ...; { # of receive buffers }
    MAX_PERIOD_PER_RW = ...; { = ceiling((NUM_RCV_BUFFERS - 1)/N) + 1}
    MAX_PERIOD_PER_RW_MINUS1 = (MAX_PERIOD_PER_RW - 1) ;
    NUM_SND_BUFFERS  = ...; { # of send buffers}
    SIZE_OF_NEEDS_ACK = ...;{ >= 2* max # periods before needs_ack array cycles}
    SIZE_OF_NEEDS_ACK_MINUS1 = (SIZE_OF NEEDS_ACK - 1);
    EMPTY = -1; { represents empty/null snd/rcv buffer }
    ACK_TIMEOUT = ...; { sender's timeout before retransmissions }
    CHECK_VALIDITY_INTERVAL  = ...;
                    { timeout for check if objects in rcv buffers have temp value }
type
    sdu_info_type = array [1..NUM_DATA_PER_N_OR_T_SDU] of integer;
   tsdu_type  = record seqnum, period: integer; info: sdu_info_type; end;
   nsdu_type = record header: integer; seqnum, period: integer; info: sdu_info_type; end;
   ident_type = record period, seqnum: integer; end;
    PO_matrix_type = array [1..N,1..N] of 0..1;
    partial_order_type = array [1..NUM_PARTIAL_ORDER_BITS] of integer;
    reliability_type = (NBART_L,BART_NL,BART_L);
    partial_reliability_type  = array [1..N] of reliability_type;
    array_1_N_of_boolean_type = array [1..N] of boolean;

channel tsdu_channel(usr,pvd); { connects user/application and partial_order_service }
   by usr   : t_data_req(tsdu: tsdu_type);
   by pvd : t_data_ind(tsdu: tsdu_type);
            QOSLoss_failed; po_not_respected(period,seqnum: integer);

channel nsdu_channel(usr,pvd); { connects partial_order_service and network}
   by usr,pvd: n_data_req(nsdu: nsdu_type); n_data_ind(nsdu: nsdu_type);
               ack(period: integer; seqnum: integer);

{ ------------------------- Application: Transport Service User ---------------------------- }

module User_type activity;  ip ucep: tsdu_channel (usr);  end;

body User_Sender_body for User_type; external;
```

```
{ sends tsdus to the transport protocol }
body User_Receiver_body for User_type; external;
{ receives tsdus from the transport protocol }

{ ------------------------------ PO Transport Protocol ------------------------------------ }

module POC_type activity; ip ucep: tsdu_channel(pvd); lcep: nsdu_channel(usr); end;
body POC_Sender_Body for POC_Type;
state active;
type needs_ack_type   = array [1..N,0..SIZE_OF_NEEDS_ACK_MINUS1] of boolean;
     snd_buffers_type = array [1 .. NUM_SND_BUFFERS] of tsdu_type;
var nsdu : nsdu_type;
   snd_PO_matrix : PO_matrix_type;
   partial_order : partial_order_type;
   PR_vector : partial_reliability_type; { vector describing partial reliability }
   partial_reliability : partial_reliability_type; { negotiated partial reliability }
   snd_used_buffers : integer; { # send buffers currently in use }
   snd_buffers : snd_buffers_type; { contains sent BART tsdus awaiting ack }
   needs_ack : needs_ack_type;
            { true if tsdu i of per. [j mod SIZE_OF_NEEDS_ACK] awaits ack}
   earliest_per : integer; { period represented by first column of needs_ack  }
   snd_curr_per : integer; { period for which tsdu are being sent; }
                           { not necessarily 1st period of objects awaiting ack }
   num_curr_per_tsdu_sent: integer; { num tsdus of current period already sent once }

procedure FreeSndBuffer(period, seqnum: integer; var snd_buffers: snd_buffers_type;
              var snd_used_buffers: integer);  primitive;
      { find stored tsdu(period,tsdu) and free the buffer }
procedure Init_PO_Matrix(var snd_PO_matrix: PO_matrix_type; partial_order:
partial_order_type); primitive;
      { initialize upper right triangular snd_PO_matrix with negotiated partial order }
procedure Init_PR_Vector(var PR_vector : partial_reliability_type;
                partial_reliability: partial_reliability_type); primitive;
      { initialize PR_vector with negotiated partial_reliability }
function IsObjectInSndBuffers(period: integer; seqnum: integer): boolean; primitive;
      { true when tsdu (period,seqnum) is in one of the snd buffers }
function IsPORespected (period: integer; seqnum: integer): boolean; primitive;
      { true when given tsdu respects partial order. }
function IsObjectSendable(period: integer; seqnum: integer): boolean; primitive;
       { returns true when tsdu(period,seqnum) can be sent for the first time procedure
Store_Sent_tsdu(tsdu: tsdu_type; var snd_buffers: snd_buffers_type;
                var snd_used_buffers: integer); primitive;
      { find an empty buffer in snd_buffers and store tsdu in it }
procedure Update_Needs_Ack(var needs_ack: needs_ack_type; var earliest_per: integer;
                                period, seqnum: integer); primitive;
      { when ack arrives, update knowledge of outstanding acks }
procedure Update_Snd_PO_Matrix(var snd_PO_matrix: PO_matrix_type;
                                     var num_curr_per_tsdu_sent: integer;
                                    var snd_curr_per: integer; seqnum: integer); primitive;
      { each time an object is sent, the dynamic PO matrix has to be updated }
procedure Prepare_nsdu(var nsdu: nsdu_type; tsdu: tsdu_type); primitive;
      { use info in tsdu to form nsdu  }

initialize to active
  var i,j: integer;
  begin
   snd_curr_per := 1;
   num_curr_per_tsdu_sent := 0;
```

```
   snd_used_buffers := 0;
    for i := 1 to N do for j := 0 to SIZE_OF_NEEDS_ACK_MINUS1 do needs_ack[i,j] :=
true;
   earliest_per := 1;
   for i := 1 to NUM_SND_BUFFERS do
     begin
      snd_buffers[i].period := EMPTY;
      snd_buffers[i].seqnum := EMPTY;
       for j := 1 to NUM_DATA_PER_N_OR_T_SDU do snd_buffers[i].info[j] := EMPTY;
      end;
   Init_PO_Matrix(snd_PO_matrix,partial_order);
   Init_PR_Vector(PR_vector,partial_reliability);
   end;

trans
from active to active
{ User_Sender wishes to transmit an object to User_Receiver }
when ucep.t_data_req(tsdu)
provided IsObjectSendable(tsdu.period,tsdu.seqnum)
   name Data_Request:
  begin
    Update_Snd_PO_Matrix(snd_PO_matrix,
num_curr_per_tsdu_sent,snd_curr_per,tsdu.seqnum);
   if (needs_ack[tsdu.seqnum, tsdu.period mod SIZE_OF_NEEDS_ACK]) then
     begin
      Prepare_nsdu(nsdu,tsdu);      { nsdu is based upon info within given tsdu }
      output lcep.n_data_req(nsdu);  { transmit the object }
      { buffer all BART objects in case later retransmission is needed }
       if (PR_vector[tsdu.seqnum] = BART_L) or (PR_vector[tsdu.seqnum] = BART_NL)
then
         Store_Sent_tsdu(tsdu,snd_buffers,snd_used_buffers)
      else  { assert: PR = NBART_L;  by def, on sending NBART_L object, no need to wait
for ack          }
         Update_Needs_Ack(needs_ack,earliest_per,tsdu.period,tsdu.seqnum);
      end;
   { else tsdu has been previously declared lost; it is not sent }
   end;

from active to active
{ User_Sender tries to transmit an object not according to the partial order   }
when ucep.t_data_req(tsdu)
provided not(IsPORespected(tsdu.period,tsdu.seqnum))
   name Error_in_Data_Request:
   begin { user is warned it did not respect the negotiated partial order }
   output ucep.po_not_respected(tsdu.period,tsdu.seqnum)
   end;

from active to active
{An ack returned from the User_Receiver arrives from the Network}
when lcep.ack(period,seqnum)
   name Ack_Management:
   begin
   if IsObjectInSndBuffers(period,seqnum) then
      begin
       Update_Needs_Ack(needs_ack,earliest_per,period,seqnum);
      FreeSndBuffer(period,seqnum,snd_buffers,snd_used_buffers);
      end
   else
```

```
      begin { ack is either duplicate ack or ack of a declared lost tsdu not yet sent }
      { check if ack is in one of the periods being monitored by needs_ack array }
        if (earliest_per <= period) and (period < (earliest_per + SIZE_OF_NEEDS_ACK))
then
          Update_Needs_Ack(needs_ack,earliest_per,period,seqnum);
      { when (snd_curr_per < period) then ack = duplicate; discard it }
      end;
   end;

from active to active
{ Retransmit objects if expected ack has not arrived }
any X: 1..NUM_SND_BUFFERS do
provided snd_buffers[X].period <> EMPTY
delay (ACK_TIMEOUT)
   name Timeout_Retransmit:
   begin
    Prepare_nsdu(nsdu,snd_buffers[X]);
    output lcep.n_data_req(nsdu);
   end;
end;

{ ---------------------------------------------------------------------------------------------------- }
body POC_Receiver_body for POC_type;
state   active;
type rcv_buffers_type = array [1..NUM_RCV_BUFFERS] of tsdu_type;
    stored_type      = array [0..MAX_PERIOD_PER_RW_MINUS1,1..N] of integer;
    rcv_proc_obj_event_type = (LOSE_OBJ, DELIVER_NEW_OBJ, DELIVER_BUF_OBJ);
    rcv_adj_per_event_type  = (OBJ_PROCESSED, OBJ_STORED, INIT_EDGES);
var tsdu: tsdu_type;
   rcv_used_buffers : integer; { # rcv buffers currently filled}
   rcv_PO_matrix : PO_matrix_type; { dynamic partial order matrix }
   partial_order : partial_order_type; { negotiated static partial order }
   PR_vector : partial_reliability_type; { dynamic vector describing partial reliability}
   partial_reliability: partial_reliability_type; { negotiated partial reliability }
   delivered : array_1_N_of_boolean_type; { true if tsdu[i] was delivered }
   lost : array_1_N_of_boolean_type; { true if tsdu[i] assumed lost }
   rcv_buffers : rcv_buffers_type; { buffers for out of order tsdus }
   check_buffers : boolean; { if true, check stored tsdus for possible delivery}
   rcv_curr_per : integer; { period of objs being delivered}
   stored : stored_type; { indicates buffer location of stored tsdu }
   num_buffable_per : integer;
                {# successive periods which may be fully or partially buffered }
   last_buffable_per : integer; { last period for which objects may be buffered }
   rcv_proc_obj_event: rcv_proc_obj_event_type;
   rcv_adj_per_event : rcv_adj_per_event_type;
   num_obj_curr_per_deliv_lost : integer;
                { # objects in current period already deliv'd or lost }
   max_num_obj_last_buffable_per : integer;  { max objects in last_bufferable_per }
   curr_num_obj_last_buffable_per: integer;  { current objects in last_buffable_per }

function IsObjectStillUseful(seqnum: integer): boolean; primitive;
      { return true if particular object still has temporal value; otherwise false }
function IsQOSLossExceeded: boolean; primitive;
      { true if most recent loss results in less than negotiated QOS }
function IsObjectDeliverable (rcv_PO_matrix: PO_matrix_type; tsdu: tsdu_type):
boolean; primitive;
      { true when tsdu respects partial order and can be immediately delivered}
```

```
procedure Init_PO_Matrix(var rcv_PO_matrix: PO_matrix_type; partial_order:
partial_order_type); primitive;
      { initialize PO_matrix with negotiated partial order }
procedure Init_PR_Vector(var PR_vector      : partial_reliability_type;
                  partial_reliability: partial_reliability_type); primitive;
      { initialize PR_vector with negotiated partial reliability class information procedure
Store_Received_tsdu(tsdu: tsdu_type; var rcv_buffers: rcv_buffers_type;
                  var rcv_used_buffers: integer); primitive;
      { find an empty buffer in rcv_buffers and store tsdu in it }
procedure Process_Object(seqnum: integer;
                              rcv_proc_obj_event: rcv_proc_obj_event_type); primitive;
      { object has been: delivered immediately upon arrival (DELIVER_NEW_OBJ) }
      { or delivered from}
{ a buffer (DELIVER_BUF_OBJ) or presumed lost (LOSE_OBJ); perform needed
processing }
function NumUndelivableObjects(matrix: PO_matrix_type): integer; primitive;
      { determine # of buffers to reserve for objects in current period }
function InBuffer(tsdu: tsdu_type): boolean; primitive;
      { Determine if object is currently being buffered }
function AlreadyProcessed(tsdu: tsdu_type): boolean; primitive;
      { check if object is :
      {A. Before receiving current period; }
      { B. In receiving current period and either Delivered or lost; }
      { or C. Buffered }
function IsObjectBufferable(tsdu: tsdu_type): boolean; primitive;
      { check if object can be buffered}
function IsObjectReceivable(rcv_PO_matrix: PO_matrix_type; tsdu: tsdu_type)
                                                                  : boolean; primitive;
      { check if object is either deliverable or bufferable                    }
procedure Adjust_Last_Period(rcv_adj_per_event: rcv_adj_per_event_type;
                              period: integer); primitive;
      {keep track of latest period for which an object can be buffered }
procedure Prepare_tsdu( nsdu: nsdu_type; var tsdu: tsdu_type); primitive;
      { extract info from nsdu to produce tsdu; no need to extract nsdu header }

initialize to active
  var i,j: integer;
  begin
   num_buffable_per := 0;
   check_buffers := false;
   rcv_used_buffers := 0;
   for i:= 1 to NUM_RCV_BUFFERS do
     begin
      rcv_buffers[i].period := EMPTY;
      rcv_buffers[i].seqnum := EMPTY;
       for j:=1 to NUM_DATA_PER_N_OR_T_SDU do rcv_buffers[i].info[j] := EMPTY;
      end;
   for i := 1 to N do
     begin
      for j := 0 to MAX_PERIOD_PER_RW_MINUS1 do stored[j,i] := EMPTY;;
      lost[i] := false;
      delivered[i] := false;
      end;
   Init_PO_Matrix(rcv_PO_matrix,partial_order);
   Init_PR_Vector(PR_vector,partial_reliability);
   rcv_curr_per := 1;
   num_obj_curr_per_deliv_lost    := 0;
```

```
    curr_num_obj_last_buffable_per := 0;
    max_num_obj_last_buffable_per  := 0;
    Adjust_Last_Period(INIT_EDGES,rcv_curr_per);
    end;

trans
from active to active
{ An object from User_Sender arrives from the network }
when lcep.n_data_ind(nsdu)
   name Check_Newly_Arriving_Object:
   begin
   Prepare_tsdu(nsdu,tsdu);    { extract important info from arriving nsdu }
   if IsObjectReceivable(rcv_PO_matrix,tsdu) then
      begin
      if IsObjectDeliverable(rcv_PO_matrix,tsdu) then
         begin { deliver the tsdu }
          output ucep.t_data_ind(tsdu);
          Process_Object(tsdu.seqnum,DELIVER_NEW_OBJ);
           Adjust_Last_Period(OBJ_PROCESSED,tsdu.period);
         end
      else
         begin
          Store_Received_tsdu(tsdu,rcv_buffers,rcv_used_buffers);
           Adjust_Last_Period(OBJ_STORED,tsdu.period);
         end;
      if (PR_vector[tsdu.seqnum] <> NBART_L) then
         output lcep.ack(tsdu.period,tsdu.seqnum);
      end
   else
      { cannot deliver nor buffer; send ack if a duplicate }
      begin
      if AlreadyProcessed(tsdu) and (PR_vector[tsdu.seqnum] <> NBART_L) then
         output lcep.ack(nsdu.period,nsdu.seqnum);
      end;
   end;

from active to active
{ Check if any currently buffered objects can be delivered to User_Receiver }
provided check_buffers and (1 <= rcv_used_buffers)
   { enabled when an object delivered or lost provided at least one buffer is full }
   var i, buf_number: integer;
   name Check_Buffers_For_Delivery:
   begin
   while check_buffers do
      begin { loop until one complete loop fails to deliver an object}
      check_buffers := false;
      for i := 1 to N do
         begin
         buf_number := stored[rcv_curr_per mod MAX_PERIOD_PER_RW,i];
         if (buf_number <> EMPTY) then
            if IsObjectDeliverable(rcv_PO_matrix,rcv_buffers[buf_number]) then
               begin { a deliverable buffered object has been found }
                output ucep.t_data_ind(rcv_buffers[buf_number]);
                Process_Object(rcv_buffers[buf_number].seqnum,DELIVER_BUF_OBJ);
                 Adjust_Last_Period(OBJ_PROCESSED,rcv_buffers[buf_number].period);
               end;
         end;
      end;
```

```
   end;

from active to active { Periodically check objects for temporal value }
delay (CHECK_VALIDITY_INTERVAL)
   var i: integer;
   name Validate_Temporal_Value:
   begin
    for i := 1 to N do
      if (not (delivered[i] or lost[i] or IsObjectStillUseful(i))) then
        begin { found nonuseful object; presume it lost }
         Process_Object(i,LOSE_OBJ);
         if IsQOSLossExceeded then output ucep.QOSLoss_failed;
         if PR_vector[i] = BART_L then output lcep.ack(rcv_curr_per,i);
          Adjust_Last_Period(OBJ_PROCESSED,rcv_curr_per);
        end;
   end;
end;

{ ------------------------------------ Network Layer ---------------------------------------- }

module Network_type activity;
  ip lcep1: nsdu_channel(pvd); lcep2: nsdu_channel(pvd);
end;

body Network_body for Network_type; external;
{ network service between transport protocol entities}

{ ------------------------------------ Main Specification ----------------------------------- }

modvar User_Sender,User_Receiver  : User_type;
         POC_Sender, POC_Receiver : POC_type;
         Network                  : Network_type;
initialize
  begin
   init User_Sender   with User_Sender_body;
   init User_Receiver with User_Receiver_body;
   init POC_Sender    with POC_Sender_Body;
   init POC_Receiver  with POC_Receiver_body;
   init Network       with Network_body;
   connect User_Sender.ucep   to POC_Sender.ucep;
   connect User_Receiver.ucep to POC_Receiver.ucep;
   connect POC_Sender.lcep    to Network.lcep1;
   connect POC_Receiver.lcep  to Network.lcep2;
   end;
end.
```

PUBLICATIONS DE L'AUTEUR

Revues Internationnales

Thai (K-L), Chassot (C.), Fdida (S.), Diaz (M.). Transport Layer for Cooperative Multimedia Application. Annals of Telecommunications, May-June 94, tome 49, n°5-6.

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, May-June 94, tome 49, n°5-6.

Amer (P. D.), Chassot (C.), Connolly (C.), Conrad (P.), Diaz (M.). Partial Order Transport Service for Multimedia and other Applications. IEEE/ACM Transactions on Networking, Oct. 1994, vol.2, n° 5.

Chassot (C.), Diaz (M.), Lozes (A.). From the Partial Order Concept to Partial Order Multimedia Connections. A paraître dans JHSN (Journal for High Speed Network) 1995 suite à une sélection de l'article publié à HIPPARCH'94 .

Conférences avec actes

Amer (P. D.), Chassot (C.), Connolly (C.), Diaz (M.). Partial Order Transport Service for Multimedia Applications : Reliable Service. Proc. 2nd High Performance Distributed Computing Conference, July 1993.

Amer (P. D.), Chassot (C.), Connolly (C.), Diaz (M.). Partial Order Transport Service for Multimedia Applications : Unreliable Service. Proc. 3rd International Networking Conference, INET'93, Aug. 1993.

Chassot (Ch.), Diaz (M), Lozes (A.). Principes d'implantation d'une connexion multimédia d'ordre partiel. 4ième Colloque Francophone sur l'Ingénierie des Protocoles, CFIP'95, Rennes, Mai 1995.

Chassot (C.), Diaz (M.), Lozes (A.). From the Partial Order Concept to Partial Order Multimedia Connections. First HIPPARCH workshop. INRIA Sophia Antipolis, December 15-16, 1994.

Diaz (M), Drira (K.), Lozes (A.), Chassot (Ch.). 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.

Chassot (C), Fournier (M), Lozes (A.), Diaz (M). Service Definition of a Multimedia Partial Order Connection. 2nd COST 237 workshop on Teleservices and Multimedia Communications, Copenhagen, Denmark, November 21-22, 1995.

Rapports du contrat CNET-CNRS (Projet CESAME)

Chassot (C.), Diaz (M.), Lozes (A.), De Saqui Sannes (P). Spécification en Estelle d'un sous-ensemble du protocole de Transport haute vitesse XTP et mesures de performances. Rapport du marché CNET France Télécom 92 1B 178 - Lot 1 et Rapport Technique LAAS-CNRS, Novembre 1992.

Amer (P.D.), Chassot (C.), Connolly (T.), Diaz (M.), Lozes (A.). Service d'ordre partiel fiable. Rapport du marché CNET France Télécom 92 1B 178 - Lot 1 et Rapport Technique LAAS-CNRS, Juillet 1993.

Amer (P.D.), Chassot (C.), Connolly (T.), Diaz (M.), Lozes (A.). Service d'ordre partiel non fiable. Rapport du marché CNET France Télécom 92 1B 178 - Lot 1 et Rapport Technique LAAS-CNRS, Juillet 1994.

Chassot (C.), Diaz (M.), Lozes (A.). Principes d'implantation d'une connexion multimédia d'ordre partiel - Rapport du marché CNET France Télécom 92 1B 178 - Lot1 et Rapport Technique LAAS-CNRS, septembre 1994.

RÉFÉRENCES BIBLIOGRAPHIQUES

[Ahuj90] M.Ajuha. "FLUSH Primitives for Asynchronous Dist'd Systems". Info Processing Letters, February 1990.

[Alle83] J.F.Allen. "Maintening knowledge about temporal interval". Communication of the ACM, 26, n°11, November 1983.

[Anag91] M.E.Anagnostou, M.E.Theologou, K.M.Vlakos, D.Tournis, E.N.Protonotarios. "Quality of Service requirements in ATM-based B-ISDNs". Computer Communications, vol.14, n°4, May 1991.

[Amer93] P.D.Amer, D.H.New. "Protocol visualisation in Estelle". Computer Networks and ISDN Systems, vol.25, n°7, pp.741-760, February 1993.

[Amer93a] P.D.Amer, C.Chassot, T.Connolly, M.Diaz. "Partial order transport service for multimedia applications : reliable service". Proc. 2nd High Performance Distributed Computing Conference, July 1993.

[Amer93b] P.D.Amer, C.Chassot, T.Connolly, M.Diaz. "Partial order transport service for multimedia applications : unreliable service". Proc. 3rd International Networking Conference, INET'93, Aug. 1993.

[Amer94] P.D.Amer, C.Chassot, T.Connolly, M.Diaz, P.Conrad. "Partial Order Transport Service to Support Multimedia Connections". IEEE/ACM Transactions on Networking, vol.2, n° 5, Oct. 1994.

[Ami93] A. Aminmansour. "Multimedia : A revolutionary tool to enhance teaching and learning of structural steel design". Proccedings of ED'MEDIA 93.

[Andr95] K. Andrews. "Embedding Courseware into the Internet : Problems and Solutions". Proceedings of ED'MEDIA 95.

[Ande92] Andersen (S. C.). "High-speed Transport Service (HSTS) specification". Proposed contribution to ISO/IEC JTC1 SC6/WG4, June 1992.

[Auge93] F. Augenstein, T. Ottman, J. Schoning. "Logical markup for hypermedia documents : the TRAIN system". Proccedings of ED'MEDIA 93.

[Ayac85] J.M.Ayache, J.P.Courtiat, M.Diaz, G.Juanole. "Utilisation des réseaux de Petri pour la modélisation et la validation des protocoles". Revue TSI, vol.4, 1985, ed.Dunod.

[Bagu92a] Y. Baguette, L. Léonard, G. Leduc, A. Danthine. "Four types of enhanced Transport Services and their LOTOS specification". Technical Report for the ESPRIT II project 5341 (OSI95), May 1992.

[Bagu92b] Y. Baguette, L. Léonard, G. Leduc, A. Danthine, O. Bonaventure. "OSI 95 Enhanced Transport Facilities and Functions". OSI 95/ Delivrable ULg-A/P/V1 S.A.R.T 92/25/05, December 1992.

[Barr93] A.E. Barron, K. Ivers. "Simulation training for electronic mail systems". Proccedings of ED'MEDIA 93.

[BERK92] "The BERKOM Multimedia Collaboration Teleservice". Release 2.0, BERKOM Technical Report. October 1992.

[BERK93] "The BERKOM Multimedia Collaboration Teleservice". Release 2.2, BERKOM Technical Report. May 1993.

[Bert91] B. Berthomieu, M. Diaz. "Modeling and Verification of Time Dependent Systems using Time Petri Nets", IEEE Transactions on Software Engineering, 1991.

[Boch80] G.V. Bochmann. "A general transition model for protocols and communication services". IEEE Trans. on Communications, COM-28, n° 4, pp. 643-650, April 1980.

[Bond92] A. H. Bond et L. Gasser. A Subject-Indexed Bibliography of Distributed Artificial Intelligence. IEEE Transactions on Systems, Man, and Cybernetics, 22(6):1260-1281, Novembre/Décembre 1992.

[Bonj94] D.Bonjour, T.Haudoin. "ATM and AAL layer issues concerning multimedia applications". Annals of Telecommunications, tome 49, n° 5-6, May-June 1994.

[Brin87] E. Brinksma. "The specification language LOTOS". Computer Networks and ISDN Systems, North- Holland, 1987.

[Budk87] S. Budkowski, P. Dembinski. "The Formal Specification Technique Estelle, Computer Networks and ISDN Systems". North Holland, 1987.

[Camp92] A. Campbell, G.Coulson, D.Hutchinson. "A Multimedia Enhanced Transport Service in a Quality of Service Architecture". 4th International workshop on Network and Operating Systems Support for Digital Audio and Video, Nov. (3-5) 1993, Lancaster, UK.

[Camp93] A. Campbell, G.Coulson, D.Hutchinson. "A QoS enhanced transport service for multimedia communications over local ATM networks in a quality of service architecture". COST 237, Private communication, Nov. 1993.

[Camp94a] A. Campbell, G.Coulson, D.Hutchinson. "A Quality of Service Architecture". ACM Computer Communications Review, April 1994.

[Camp94b] A. Campbell, G.Coulson, D.Hutchinson. "Flow Management in a Quality of Service Architecture". 5th IFIP Conference on High Performance Networking, June 27 - July 1, 1994.

[Crow92] J.Crowcroft, I.Wakeman, Z.Wang. "Layering Considered Harmful.IEEE Network, vol.6, n°1, January 92.

[CCIT88] CCITT, "Terms and Definition Related to the Qualify Of Telecommunication Services". CCITT E.800, Blue Book, 1988.

[Chai92] B. Chaib-Draa, R. Mandiau et P. Millot. Distributed Artificial Intelligence: An Annotated Bibliography. ACM SIGART Bulletin, 3(3):20-37, Août 1992.

[Cher89] Charinton (D.R.), Williamson (C.L.). "VMTP as the Transport layer for high-performance distributed systems". IEEE Communications Magazine, pages 37-44, June 1989.

[Cher88] Cheriton (D.). "VMTP : Versatile Message Transaction Protocol - protocol specification". RFC 1045, Feb. 1988.

[Ches79] Chesson (G. L.). "Datakit software architecture". Proc. ICC (1979), pp. 158-169.

[Ches87] G. Chesson . "The Protocol Engines Project". Unix Rev, September 1987.

[Ches89] G. Chesson. "XPP/PE Design Considérations". IFIP International Workshop on Protocols for High-Speed Networks, Rüschlikon, May 9-11, 1989.

[Chun94] G. Chung, K. Jeffay, H. Abdel-Wahab. "Dynamic participation in a computer based conferencing system". Computer Communications - vol.17, n°1, Juanuary 94.

[Clar82] D.D. Clark. "Window and acknowledgment strategy in TCP". RFC 813, July 1982.

[Clar87] D.D. Clark, M.L. Lambert, L. Zhang. "NETBLT : a bulk data transfer protocol". RFC 998, March 1987.

[Clar89] D. Clark, V. Jacobson, J. Romkey, H. Salwen. An analysis of TCP processing overhead. IEEE Communications Magazine, vol.27, n°6, pages 23-29, June 1989.

[Clar90] D.D. Clark, D.L. Tennenhouse. "Architectural Considerations for a New Generation of Protocols". Proceedings of ACM SIGCOMM, 1990.

[Cocq92] P. Cocquet (P), C. Diot. "Enhanced Transport Service (ETS)". Proposed contribution to ISO/IEC JTC1 SC6/WG4, June 1992.

[Come91] D.E. Comer. "INTERNETWORKING WITH TCP/IP". Prentice-Hall International Editions, 1991.

[Conn94] T.Connolly, P.D.Amer, P.Conrad. "An extension to TCP : Partial Order Service". Internet RFC 1693, November 1994.

[Conr95] P.Conrad, P.Amer, R.Marasli. "Graceful degradation of multimedia documents via partial order and partial reliability transport protocols". IEEE Workshop on Multimedia Synchronization, Virginia, May 1995.

[Cour84] J.P. Courtiat, J.M. Ayache, B. Algayres. "Petri Nets are good for protocols". Computer Communications Review, 14, n° 2, 1984.

[Cour87] J.P.Courtiat, P.Dembinski, R.Groz, C.Jard. "Estelle : Un Langage ISO pour les Algorithmes Distribués et les Protocoles". TSI, Technique et Science Informatiques, Vol. 6, n° 2, pp. 89-102, 1987.

[Cour94] J.P.Courtiat, Cruz de Oliveira. "About Time Non-determinism and Exception Handling in a Temporal Extension of LOTOS". IFIP Int Conf on Protocol

Specification, Testing and Verification, Vancouver, June 1994, North Holland, Vuong & Chanson Editors.

[Croi94] P. Croisy, D. Clement, L. Barme. "Co-Learning at a distance : FIRST trial of an integrated learning environment", Proceedings of ED'MEDIA 94.

[Crow92] J.Crowcroft, I.Wakeman, Z.Wang. "Layering Considered Harmful.IEEE Network, vol.6, n°1, January 92.

[Dair94] L. Dairaine. "Techniques de synchronisation dans les systèmes de communication haut-débit multimédia", Thèse de doctorat, Université Pierre et Marie Curie, Paris, France, Septembre, 1994.

[Dant80] A.Danthine. "Protocol representation with finite-state models". IEEE Transactions on communications, vol.28,n°4, April 1980.

[Dant92] A.Danthine, Y.Baguette, G.Leduc, L.Léonard. "The OSI 95 connection-mode transport service". Proc. 4th IFIP Conference on High Performance Networking, HPN'92, Decembre 1992.

[Dant93] A. Danthine, O. Bonaventure : From Best Effort to Enhanced QoS, Technical report of the CIO project, July 1993.

[Daun92] J.Y. Dauneau, P. Varaiya, M. Diaz, P. Sénac. "Multimedia data structures and documents", Rapport LAAS n°92142, mars 1992.

[Der93] A.C. Derycke, P. Croisy, P. Vilers. "Computer Supported Cooperative Learning : a real time multimedia approach". Proccedings of ED'MEDIA 93.

[Diaz82] M.Diaz. "Modelling and Analysis od Communication and Cooperation Protocols using Petri Nets based Models". Tutorial paper in Computer Networks, vol.6, n°6, December 1982.

[Diaz85] M. Diaz, C. Vissers, J.P. Ansart. "SEDOS : Software Environment for the Design of Open Distributed Systems", Proceedings of the 2nd ESPRIT Technical Week, Brussels, September 1985, North Holland.

[Diaz89] M. Diaz, C. Vissers, J.P. Ansart. "SEDOS : Software Environment for the Design of Open Distributed Systems", The Formal Description Technique Estelle, North-Holland, 1989.

[Diaz92] M.Diaz. A Logical Model of Cooperation. "IEEE Future Trends of Distributed Computing Systems", Taiwan, April 1992.

[Diaz93] M.Diaz. "Coopération, logique et partage des données", 1ères Journées Francophones Intelligence Artificielle Distribuée et Systèmes Multi-Agents, Toulouse, Avril 1993.

[Diaz93a] M.Diaz, P.Senac, P.de Saqui-Sannès. "Un modèle pour la spécification et le contrôle de la synchronisation multimédia en environnement distribué", CFIP'93 Ingéniérie des Protocoles, R.Dssouli & G.V. Bochmann Editeurs, Hermès 1993.

[Diaz93b] M. Diaz, P. Senac. "Time Stream Petri nets : a model for timed multimedia streams synchronization", Int. Conf. on Multimedia Modeling, Singapore, pp. 8-9, November 1993, World Scientific, November 1993, 257-273.

[Diaz94a] M.Diaz, G.Pays. "The Ceasme Project : Formal design of high speed multimedia cooperative systems". Annals of Telecommunications, tome 49, n° 5-6, May-June 1994.

[Diaz94b] M.Diaz, A.Lozes, C.Chassot , P.D.Amer. "Partial order connections : a new concept for high speed and multimedia services and protocols". Annals of Telecommunications, tome 49, n° 5-6, May-June 1994.

[Diaz94c] M. Diaz, P. Senac. "Time Stream Petri nets for Multimedia Information", Int. Conf. on Application and Theory of Petri Nets, Zaragoza, June 1994.

[Diaz95] M.Diaz, K.Drira, A.Lozes, C.Chassot. "On the Definition and Representation of the Quality of Service for Multimedia Systems", 6th IFIP Conference on High Performance Networking, 1995.

[Diot95] C.Diot. Adaptive Applications and Qos Guaranties. Invited paper to IEEE Multimedia Networking. Aizu (Japan). September 1995.

[DIS92] ISO/IEC JTC1 : IT - TIES - Connection-Oriented Transport Protocol Specification, voté le 5 juin, 1992.

[Delg93] L.Delgrossi, J.Sandvoss. "The BERKOM-II Multimedia Transport System", Aug. 1993.

[Doer90] W.A.Doeringer, D.Dykeman, M.Kaiserswerth, B.W.Meister, H.Rudin, R.Williamson. "A survey of light-weight transport protocols for high-speed networks". IEEE Transactions on Communications (Nov. 1990), 38, n° 11.

[Dupu92] S.Dupuy, W.Tawbi, E.Horlait. "Protocols for high-speed multimedia communications networks". Computer Communication, vol.15, n°6, July-August 1992.

[EDB92] S. Budkowski et al. "The EDB User Manual", INT doc., January 1992.

[Elli90] C.A/Ellis, S.J.Gibbs, G.L.Rein. "Design and use of a group editor. In Engineering for Human-Computer Interaction", North-Holland, Amsterdam, 1990, 13-25.

[Este88] ISO IS 9074. "Estelle : A Formal Description Technique Based on a Extended State Transition Model", International Standardization Organization, November 1988.

[Este91] ISO/IEC JTC1/SC21 N 5710, "Estelle Tutorial", 1991.

[Four96] M.Fournier. "Implémentation et Mesures de Performances de Protocoles de Transport". Mémoire de thèse à paraître en 1996.

[Henc93] L.Henckel. "Multimedia Communication Platform : Specification of the Enhanced Broadcast Transport Service", CIO RACE Project 2060, September 1993.

[Hehm90] D.Hehmann, M.Salmony and H.Stüttgen. "Transport services for multimedia applications on broadband networks", Computer Communications, vol 13, n°.4, May 1990.

[IEEE90] IEEE. "Distributed Queue Dual Bus (DQDB) subnetwork of Metropolitan Area Network (MAN)", Standard P802.6/D15, 1990.

[IS8072] ISO International Standard 8072, Information Processing Systems - Open Systems Interconnection - Transport service definition, 1986.

[IS8073] ISO International Standard 8073, Information Processing Systems - Open Systems Interconnection - Transport protocol specification, 1986.

[IS8072/ADD1] ISO International Standard 8072 ADD1, Information Processing Systems - Open Systems Interconnection - Transport service definition - Addendum 1 : Connectionless-mode transmission, 1986.

[IS8073/ADD2] ISO/IEC JTC1 : IPS - OSI - Connection-Oriented Transport Protocol Specification, Addendum 2: Operation of Class 4 over Connectionless Network Service., ISO 8073 ADD2, 1989.

[ISO87a] ISO/TC 97/SC 21/WG1-FDT/SG-B. "Estelle, a formal description dechnique based on an extended state transition model", Draft International Standard ISO/TC97/SC21-DP9074, September 1987.

[ISO87b] ISO/TC 97/SC 21/WG1-FDT/SG-C. "LOTOS, a formal description technique based on the temporal ordering of observational behaviour", Draft International Standard ISO/TC97/ SC21-DP8807, September 1987.

[ISO90] 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.

[ISO92] ISO. "Hypermedia/Time-Bases Structuring Language (HyTime)", ISO/IEC IS 10744, August 1992.

[ISO93a] ISO. "Coding of Moving Pictures and Associated Audio", Committee Draft for standard ISO/IEC JTC1/SC29, WG11/602, November 1993.

[ISO93b] ISO. "Coded Representation of Multimedia and Hypermedia Information Objects (MHEG)", Committee Draft, ISO/IEC CD 13522-1, June 1993.

[Jaco88] V.Jacobson. "Congestion avoidance and control". Proc. ACM SIGCOMM'88, pp. 314-329, August 1988.

[Jaco92] V.Jacobson, R. Braden, D.Borman. "TCP Extensions for High Performance", Internet Requests for Comments n°1323, May 1992.

[Jain90] N.Jain, M.Schwartz, T.Bashkow. "Transport protocol processing at GBPS rates". ACM SIGCOMM'90, Philadelphia, Sept.1990.

[Juan90] G.Juanole, M.Ayoub Dit Ayadi, C.Faure. "Using Petri Net based Models for Formal Specification and Verification un Computer Networks Application

Examples". Lecture Notes on Specification and Verification of Distributed Systems, Report for the Comett-Discuss project, October 1990.

[Karm90] A.Karmouch, L.Orozco-Barbosa, N.Georganas, M.Goldberg. "A Multimedia Communication System", IEE Journal on Selected Areas in Communication, vol.8, n°3, April 1990.

[Koeg93] J. Koegel, L. Rutledge, J. Rutlege, C. Keskin. "HyOctane: A Hytime Engine for an MMIS", Proceedings of ACM Multimedia'93, August, 1993.

[Kure93] O.Kure, I.Sortberg. "XTP over ATM", Computer Networks and ISDN Systems, vol.26, 1993.

[Lamp78] L.Lamport. Time, Clocks and the Ordering of Events in a dist'd System. CACM, 21(7),558-565, July 1978.

[Ledu93] G. Leduc, L. Léonard. "An Enhanced Version of Timed LOTOS and its Application to a Case Study", Proceedings of FORTE' 93, USA, 1993.

[Ledu94] G. Leduc, L. Léonard. "A formal definition of time in LOTOS". Technical report, Université de Liège, Belgium, 1994.

[Lége91] A. Léger, T. Omachi, G. Wallace. "JPEG still picture compression algorithm", Optical Engineering, Vol. 30, N° 7, July 1991.

[Leop92] Léopold (H.), Campbell (A.), Hutchinson (D.), Singer (N.). "Towards an integrated quality of service architecture (QoS-A) for distributed multimedia communications". Proc. 4th IFIP Conference on High Performance Networking, HPN'92 (Dec. 1992).

[Linn85] R. J. Linn. "The Features and Facilities of Estelle". Proceedings of the IFIP Conference on Protocol Specification, Testing and Verification V, Toulouse, Juin 1985, pp. 271-297.

[Litt90a] Little (T.), Ghafoor (A.). "Network Considerations for Dist'd Multimedia Objects Composition and Communication". IEEE Network Magazine, 32-49, November 1990.

[Litt90b] Little (T.), Ghafoor (A.). "Synchronization and Storage Models for Multimedia Objects". IEEE J on Selected Areas in Comm., 8(3), 413-427, April 1990.

[LLB83] Delta-t protocol specification. Rep. UCID-19293, Lawrence Livermore Lab. Apr. 1983.

[Lu93] Guo-Jun Lu. "The MPEG Standard, Tutorial in The International Conference on Multimedia Modelling MMM'93", Singapore, 1993.

[Mars95] A.D. Marshall."Developing Hypertext Courseware on the Worls Wide Web" Proceedings of ED'MEDIA 95.

[Mara96] R.Marasli, P.D.Amer, P.Conrad. "Retransmission-based partially reliable transport service : An analytic model", Proc IEEE INFOCOM 96, San Fransico, March 1996.

[Maur93] A.Maurand, C.Bourdeau, M.Aguerra. "Principes de conception d'un protocole de Transport haute vitesse", Rapport du marché CNET France Télécom 92 1B 178 - Lot 1, Decembre 1993.

[Merl76] P.M. Merlin, D.J.Farber. "Recoverability of communication protocols, Implication of a theoritical approach", IEEE Transactions on Communications, Septembre 1976.

[Metz93] B.Metzler, I.Miloucheva. "Multimedia Communication Platform : Specification of the Broadband Transport Protocol XTPX", CIO, Race Project 2060, 60/TUB/CIO/DS/A/002/b3, september 1993.

[Milo93a] I.Miloucheva, K.Rebensburg. "The use of XTP in a large european communication project". Proc. of the 2nd Intern. Conf. on Broadband Islands, Athens, Greece, June 1993.

[Milo93b] I.Miloucheva. "Specification of enhanced protocol facilities for multicast and broadcast", CIO RACE Project 2060, R2060/TUB/CIO/DS/P/003/b1, Oct. 1993.

[Naff90] N.Naffah. "Multimedia applications". Computer Communications, vol.13, n°4, May 1990.

[Neig87] G.Neiger, S.Toueg. "Substitution of Real Time and Common Knowledge in Asynchronous dist'd Systems", in Proc 4th Symp. on Principles of Dist'd Computing, 1987.

[Newc91] S. Newcomb, N. Kipp, V. Newcomb. "The Hytime - Hypermedia/Time-base Document Structuring Language", Communications of the ACM, vol. 34, n°11, November, 1991.

[Nord89] E. Nordmark, D. R. Cheriton. "Experiences from VMTP : How to achieve low response time", Protocols for High-Speed Networks, H. Rudin and R. Williamson (Editors), Elsevier Science Publishers B.V. (North-Holland), IFIP,1989.

[OPNE] OPNET. Trademark Holder : MIL 3, Inc, Xerox Corporation.

[Owe94] P.Owezarski, M.Diaz. "Développement d'un système de visioconférence sur réseau local". Rapport LAAS n°94354, Juillet 1994.

[Owe95a] Owezarski, P. Diaz, M. Sénac, P. Modélisation et implémentation de mécanismes de synchronisation multimédia dans une application de visioconférence. Actes du colloque francophone sur l'ingéniérie des protocoles, Rennes, France, 10-12 mai 1995.

[Owe95b] P.Owezarski, V. Baudin, M. Diaz, J.F. Schmidt. "Multimedia Teleteaching : Introduction of Synchronization and Cooperation Mechanisms in Distance Learning". Proceedings of ED'MEDIA 95.

[Owe95c] P.Owezarski, M/Diaz. "Gestion de la synchronisation multimédia et de la QoS dans une application de visioconférence", Soumis à TSI, Numéro spécial : Multimédia et Collecticiel.

[Part92] C.Partridge. "A proposed Flow Specification", Internet Request for Comments n°1363, Sept.1992.

[Para95] Parallax - Video Source (Workstation Video News from Parallax Graphics) - Winter 95 - Vol.3, n°1.

[Pehr91] B.Pehrson, S.Pink. "Multimedia and high speed networking in MultiG", Computer Networks and ISDN Systems, 21, pp 315-319, 1991.

[Pehr92] B.Pehrson, P.Gunningberg, S.Pink. "Distributed multimedia applications on gigabit networks", IEEE Network Magazine, pp 26-35, January 1992.

[PEI92] XTP protocol definition, Revision 3.6. Protocol Engines Incorporated, January 1992.

[Pete89] L.Peterson, N.Buchholz, R.Schlighting. "Preserving and Using Context Information in Interprocess Communication". ACM Trans. on Computer System, 7(3), 217-246, August 1989..

[Post81] J.Postel. "Transmission Control Protocol". Internet Request for Comments n°793, September 1981.

[Pope89] R.Popescu-Zeletin. "From broadband ISDN to multimedia computer networks", Computer Networks and ISDN Systems, 18, pp 47-54, 1989.

[Pryc91] M. de Prycker. "Asynchronous Transfer Mode: Solution for Broadband ISDN", Book Computer Communications and Networking, Ellis Horwood, 1991.

[Rafi83] O.Rafiq, J.P.Ansart. "VADILOC : A Protocol Validator and its Applications". Proceedings of 3rd IFIP workshop on Protocol Specification, Testing and Verification, Ruschlikon, North-Holland, May 1983.

[Robe95] L.Robert, M.Diaz. "Mesures de performances d'un protocole haut débit multimédia de niveau Transport", rapport de DEA, septembre 1995.

[Rust94] L.Rust. "Méthodes de synchronisation dans les systèmes répartis multimédia : une approche intégrant relations de causalité et contraintes temporelles". Thèse de doctorat, Université Paul Sabatier, Toulouse, Novembre 1994.

[Ross89] F. Ross. "An overview of FDDI: The Fiber Distributed Data Interface, IEEE Journal of Selected Areas in Communications, Vol. 7, N° 7, September 1989.

[Sant92] H. Santoso, S. Fdida. "Transport layer multicast : an enhancement for XTP bucket error control". Proc. 4th IFIP Conference on High Performance Networking, HPN'92, Dec. 1992.

[Sant94] H. Santoso, S. Fdida. Transport Layer Statistical Multicast based on the XTP Bucket Algorithm, Special Issue, Annal. Télécommunications, T 49, n° 5-6, May-June 1994.

[SDL88] CCITT/SGXI/WP3-1. "SDL, Specification and Description Language", CCITT Recommendations Z100-Z104, 1988.

[Séna94] P.Sénac, M.Diaz, P. de Saqui-Sannes (P. de). "Toward a formal specification of multimedia synchronization scenarios". Annales des télécommunications, tome 49, n° 5-6, Mai-Juin 1994.

[Sena95] P.Sénac, M.Diaz, A.Léger, P. de Saqui Sannes. "Modeling logical and temporal synchronization in Hypermedia systems", accepted for publication, IEEE Journal on Selected Areas in Communications, 1995.

[Shep90] D.Shepard, M.Salmony. "Extending OSI to support synchronisation required by multimedia applications". Computer Communications, vol.13, n°7, September 1990.

[Shep92] D.Shepard, D.Hutchison, F.Garcia, G.Coulson. "Protocol Support for Distributed Multimedia Applications". Computer Communications, vol°15, n°6, July-August 1992.

[Sije93] R.Sijelmassi, B.Strausser. "The Pet and DINGO Tools for deriving distributed implementations from Estelle", Computer Networks and ISDN Systems, vol.25, n°7, February 1993.

[Shaf92] Shafer (K.), Ahuja (M.). Process-channel(agent)-Process Model of Asynchronous dist'd Communication. In Proc, ICDCS 12, Yokohama, Japan, 4-11, June 1992.

[Sta86] R.Stanley. "Enumerative Combinatorics Volume 1". Wadsword + Brooks/Cole advanced Books and Software, Monterey, CA, 1986.

[Syka91] E.D.Sykas, K.M.Vlakos, M.J.Hillyard. "Overview of ATM Networks : functions and procedures". Computer Communications, vol.14, n°10, December 1991.

[Schw90] W.D.Schwaderer. "XTP in VLSI : Protocol Decomposition for ASIC Implementation". Proceeding on the 15th Conference on Local Area Networks, IEEE 1990.

[Tawb92] W.Tawbi , S.Dupuy, E.Horlait. "High speed protocols: state of the art in multimedia applications", Fourth Int. Conf. on Information Networks and Data Communication (INDC-92), Espoo, Finlande, Elsevier Science Publishers, March 1992.

[TENE94] TENET group. "Recent and Current Research", University of California, Berkeley, April, USA, 1994.

[Tenn88] R. L. Tenney. "A Tutorial Introduction to Estelle", Invited Paper at the 1st International Conference on Formal Description Techniques, Stirling, September 1988.

[Tenn89] D.L. Tennenhouse. "Layered multiplexing considered harmful", in H. Rudin, R. Williamson (Eds) : Protocols for high-speed networks, Elsevier (North-Holland) (1989).

[Thai94a] K. Thai, Ch. Chassot, S. Fdida, M. Diaz. "Transport Layer for Cooperative Multimedia Applications", Special Issue, Annal. Telecommunications, T 49, N° 5-6, May-June 1994.

[Thai94b] K.Thai, E.Anique, S.Fdida, M.Diaz. "Architecture de communication pour applications mulimédia réparties", Rapport du marché CNET France Télécom 92 1B 178, Sept. 1994.

[Thai95] K. Thai, S. Fdida. "XTP over AAL5 for Multimedia Applications", to appear in Special Issue, Annal. Telecommunications 1995.

[Topo90] C.Topolcic. Experimental Internet Stream Protocol, Version 2 (ST-II). RFC 1190, Oct. 1990.

[Turl93] T.Turletti. "H261 Software Codec for Videoconferencing over the Internet" Rapport de recherche n°1834, Pgme 1 Architecture parallèles, Bases de données, Réseaux et Systèmes Distribués, Juanury 93 .

[UK92] Contribution from BSI, 2 Park Street, London W1A2BS, United Kingdom. A suggested QoS Architecture for Multimedia Communications. ISO/IEC JTC1/SC21/WG1, september 1992.

[VanE89] P. Van Eijk, C. Vissers, M. Diaz Editors. "The Formal Description Technique LOTOS", North-Holland, 1989.

[Vill95] T.Villemur. "Conception de services et de protocoles pour la gestion de groupes coopératifs". Thèse de Doctorat, Janvier 1995.

[Viss86] C.Vissers, L.Logrippo. "The importance of the concept of service", 5th Int. Workshop on Protocol Specification, Testing and Verification, Toulouse, June 1985, North-Holland, M. Diaz Editor, 1986.

[Walt83] B.Walter. "Timed petri-nets for modeling and analysing protocols with time", Proceedings of PSTV, III, IFIP, 1983

[Wats81] R.W.Watson. "Timer-based mechanisms in reliable transport protocol connection management", Computer Networks, 5, pp. 47-56, 1981.

[Wats87] R.W.Watson, S.A.MamrakS.A. "Gaining efficiency in transport services by appropriate design and implementation choices", ACM Transactions on Computer Systems, 5, n° 2, May 1987.

[Wats89] R. W. Watson. "The Delta-t Transport Protocol : Features and Experience", Protocols for High-Speed Networks, H. Rudin and R. Williamson (Editors), Elsevier Science Publishers B.V. (North-Holland), IFIP,1989.

[Will94] N.Williams, G.S.Blair. "Distributed Multimedia Applications : a Review", Computer Communications, vol.17, n°2, February 1994.

[Wolf91] B. Wolfinger, M.Moran. "A continuous media data transport service and protocol for real-time communication in high speed networks", Proceedings of the 2nd International Workshop on Network and Operating System Support for Digital Audio an Video, Heidelberg, November 1991.

[Wolf95] K.D.Wolf. "The implementation of an open learning environment under World Wide Web", Proceedings of ED'MEDIA 95.

[Zhan86] L.Zhang. "Why TCP timers don't work well". Proc. ACM SIGCOMM'86, pp. 397-404, Aug. 1986.

[Zimm80] H.Zimmerman. "OSI reference model, the ISO model of architecture for open systems interconnection". IEEE Transactions on Communications, Vol.COM-28, Avril 1980.

RÉSUMÉ

Les évolutions technologiques conjointes de l'Informatique et des Télécommunications autorisent la conception d'applications multimédia distribuées, impliquant le transfert et la coordination de plusieurs types de données (texte, image et son). Face aux inadéquations conceptuelles des protocoles de Transport, plusieurs travaux de recherche se sont focalisés sur la définition de nouveaux protocoles. Les travaux présentés dans ce mémoire s'inscrivent dans ce contexte d'étude et proposent la définition et le développement d'un nouveau concept, *la connexion d'ordre partiel* (POC - Partial Order Connection), établissant un lien conceptuel entre les services et protocoles orientés-connexion tels que TCP, et sans connexion tels que UDP. Une POC est une connexion de bout en bout au travers de laquelle les données émises peuvent être délivrées suivant toutes les séquences admissibles au regard d'un ordre partiel spécifié par l'utilisateur ; une POC optimise l'utilisation des ressources mémoire et bande passante, et diminue globalement le délai de transit des données. Une formalisation des mécanismes mis en œuvre est proposée au moyen de la technique de description formelle Estelle. Ensuite, l'auteur présente les principes de conception d'une architecture de Transport multimédia intégrant le concept de POC, et définit le service correspondant. L'ordre et la fiabilité y apparaissent en tant que paramètres de QoS exprimant respectivement les contraintes de synchronisation logiques inhérentes aux documents multimédia et leur niveau acceptable de dégradation. L'étude finalement proposée définit les conditions d'utilisation du protocole haute vitesse XTP (eXpress Transfer Protocol) comme support de Transport d'une connexion multimédia d'ordre partiel.

MOTS CLÉS

Systèmes Distribués
Réseaux Hauts Débits
Services et Protocoles de Transport
Réseaux de Petri
Applications Multimédia Distribuées
Architecture de Communication Multimédia
Qualité de Service
Techniques de Description Formelle

ABSTRACT

Technical advances in Computer Science and Telecommunications lead to the design of distributed multimedia applications, that imply transfer and coordination of several data types (text, movie and audio). Work presented inside this dissertation addresses research in new Transport services and protocols with regard to multimedia applications constraints. The proposed approach is based on the definition and the development of a new concept : the *partial order connection* (POC), establishing a conceptual link between the current concepts of service/protocol represented by TCP and UDP. A POC is an end to end connection that provides a partial order service where the received objects can be delivered to the user in an order that is different from the order they have been sent. A POC allows the saving of memory and bandwidth, and generates decrease in latency times. A formal description of a POC is proposed using the Estelle formal description technique. An integration of the POC concept in a multimedia Transport architecture is then proposed ; a complete description of the corresponding service is given : 'order' et 'reliability' appear as QoS parameters allowing the service user to express both logical synchronisation constraints, and acceptable reliability degradation levels of multimedia documents. In the final study, the author shows how to use the high speed protocol XTP (eXpress Transfer Protocol) to support a multimedia partial order connection.

KEY WORDS

Distributed System
High Speed Networks
Transport Services and Protocols
Petri Nets
Distributed Multimedia Applications
Multimedia Communication Architecture
Quality of Service
Formal Description Techniques