Optimisation de la fiabilité pour des communications multipoints par satellite géostationnaire

Fabrice Arnal

THÈSE

présentée

pour obtenir le titre de :

DOCTEUR DE L'ÉCOLE NATIONALE SUPÉRIEURE DES TÉLÉCOMMUNICATIONS DE PARIS

École Doctorale d'Informatique, Télécommunications et Électronique de Paris

Spécialité : Informatique et Réseaux

Par M. Fabrice ARNAL

OPTIMISATION DE LA FIABILITÉ POUR DES COMMUNICATIONS MULTIPOINTS PAR SATELLITE GÉOSTATIONNAIRE

Thèse soutenue le 15 décembre 2004 devant le jury composé de :

M. Christian FRABOUL, Professeur des universités à l'INP de Toulouse, président du jury ;
M. Walid DABBOUS, Directeur de Recherches INRIA, rapporteur ;
M. Giovanni GIAMBENE, Maître de Conférences à l'Université de Sienne, rapporteur ;
M. Jérôme LACAN, Maître de Conférences, à l'ENSICA, examinateur ;
M. Stéphane COMBES, Ingénieur de recherches à Alcatel Space, examinateur ;
M. Gérard MARAL, Professeur des universités à l'ENST Paris, directeur de thèse ;
M. Laurent DAIRAINE, Maître de Conférences à l'ENSICA, co-directeur ;
M. Stéphane MAY, Ingénieur de recherches au CNES, invité.

Remerciements

Ces quelques mots, bien peu originaux mais tout aussi sincères, sont destinés à tous ceux qui m'ont encouragé pendant ces années. Beaucoup savent que le travail de thèse est une expérience formidablement enrichissante, mais parfois accompagnée de périodes de doutes. Dans ces moments, et en particulier pendant la délicate période de rédaction, le soutien de ces personnes a été indispensable. Je les remercie vivement, en particulier pour leur relecture attentive du manuscrit qui m'a fait gagner un temps précieux, et leur en a sans doute fait perdre beaucoup !

En premier lieu, mes pensées vont à mon directeur de thèse, M. Gérard Maral, envers qui je suis particulièrement reconnaissant pour plusieurs raisons. Après m'avoir accepté pour suivre la formation *Télécommunications par Satellite* assurée à Toulouse, il a su m'orienter vers des pistes devenues les fondements de cette recherche. Son encadrement scientifique a été inestimable tout comme sa faculté à résoudre les problèmes pratiques qui auraient pu faire obstacle à d'excellentes conditions de travail.

Je tiens aussi à remercier Laurent Dairaine qui a co-encadré la thèse. Grâce à son optimisme, il a su me dynamiser à l'issue de toutes les conversations que nous avons pu avoir. Sans son esprit d'équipe et de pédagogie, j'imagine mal comment le travail accompli aurait été possible.

Je ne peux oublier Jérôme Lacan dans ces remerciements, pour toute l'aide qu'il a pu m'apporter, et avec qui les points de vue échangés ont été à l'origine de l'essentiel des idées évoquées. La théorie des codes correcteurs et ses applications, qu'il m'a patiemment fait découvrir, m'ont permis d'appréhender les problématiques essentielles ayant accompagné les différentes contributions.

Je suis également reconnaissant à Alcatel Space d'avoir bien voulu supporter financièrement la thèse, et en particulier à Laurent Claverotte pour avoir permis son amorçage. Merci aussi à Stéphane Combes, à M. Dabbous et à M. Giambene pour avoir accepté de faire parti du jury, malgré leur emploi du temps déjà bien rempli. Merci à Christian Fraboul d'en avoir assuré la présidence, et surtout pour son soutien pendant les dernières semaines ayant précédé la soutenance.

Merci aussi à Anne, Christophe, Florestan, Garmy, Gwladys, Hélène, Jérôme, Yann, et en particulier à Julien pour tous les bons moments que nous avons partagés.

Merci également à M. Francis Castanié pour m'avoir accueilli au sein du laboratoire Tésa, à Sylvie et à tous les membres du laboratoire pour leur gentillesse et leur bonne humeur permanente. Merci enfin à Jérôme Fimes pour les mesures qu'il m'a fournies et aux autres doctorants ou membres de l'équipe DMI de l'ENSICA.

Je dédie la thèse à mes parents.

Table des matières

Liste des figures

Liste des tableaux

Liste des acronymes

AAL-5	ATM Adaptation Layer 5
AAP	Architecture Améliorée Primaire
AC	Architecture Conventionnelle
ACC	Architecture Conventionnelle avec Compression
ACK	ACKnowledgement
ADSL	Asymmetric Digital Subscriber Line
AG	Architecture Généralisée
ALC	Asynchronous Layered Coding
AM	Architecture Modifiée
AMC	Architecture Modifiée avec Compression
AMCS	Architecture Modifiée avec Compression Simplifiée
AMR	Adaptive Multi Rate
AMS	Architecture Modifiée Simplifiée
API	Application Programming Interface
ARQ	Automatic Repeat reQuest
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
ATSC	Advanced Television System Committee
AWGN	Additive White Gaussian Noise
BAS	Broadband Access Server
BEC	Binary Erasure Channel
BPSK	Binary Phase Shift Keying
BSC	Binary Symmetric Channel
CRC	Cyclic Redundancy Check
CSMA/CD	Carrier Sense Multiple Access / Collision Detection
DAMA	Demand Assignment for Multiple Access
DCCP	Datagram Control Congestion Protocol
DNS	Domain Name System
DR	Designated Router
DSM-CC	Digital Storage Media – Command and Control
DVB	Digital Video Broadcasting
DVB-C	Digital Video Broadcasting – Cable
DVB-DSNG	Digital Video Broadcasting – Digital Satellite News Gathering
DVB-H	Digital Video Broadcasting – Handheld
DVB-RCS	Digital Video Broadcasting – Return Channel Satellite
DVB-S	Digital Video Broadcasting – Satellite
DVB-T	Digital Video Broadcasting – Terrestrial
DVMRP	Distance Vector Multicast Routing Protocol
ESP	Encapsulating Security Payload
ETSI	European Telecommunications Standards Institute
FCS	Frame Check Sequence
FDDI	Fiber Distributed Data Interface
FEC	Forward Error Correction
FMT	Fade Mitigation Techniques
GEO	GEOstationary satellite
HDLC	High-level Data Link Control
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
IPSec	IP Security protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Organization for Standardization
ISP	Internet Service Provider
LAN	Local Area Network
LDPC	Low Density Parity Check

LDGM	Low Density Generator Matrix
MAN	Metropolitan Area Network
MBone	Multicast Backbone
MDS	Maximum Distance Separable
MF-TDMA	Multi Frequency – Time Division Multiple Access
MLD	Multicast Listener Discovery
MOSPF	Multicast Open Shortest Path First
MPE	Multi Protocol Encapsulation
MPEG	Moving Picture Experts Group
MPEG-2 TS	Moving Picture Experts Group 2 – Transport Stream
MPHP	Multi Protocol Header Protection
MPLS	Multi Protocol Label Switching
NACK	Negative ACKnowledgement
NCC	Network Control Center
NORM	Nack Oriented Reliable Multicast
OBP	On-Board Processing
OSI	Open System Interconnection
PDCP	Packet Data Convergence Protocol
PID	Packet IDentifier
PIM-DM	Protocol Independent Multicast – Dense Mode
PIM-SM	Protocol Independent Multicast – Sparse Mode
PER	Packet Error Rate
PLR	Packet Loss Rate
PPP	Point to Point Protocol
PSNR	Peak Signal to Noise Ratio
PUSI	Payload Unit Start Indicator
QdS	Qualité de Service
QPSK	Quadrature Phase Shift Keying
RCST	Return Channel Satellite Terminal
RMT-WG	Reliable Multicast Transport – Working Group
ROHC	RObust Header Compression
RP	Rendezvous Point
RS	Reed-Solomon
RSVP	Resource reSerVation Protocol
RTC	Réseau Téléphonique Commuté
RTP	Real-time Transport Protocol
RTCP	RTP Control Protocol
RTT	Round Trip Time
S-DAB	Satellite – Digital Audio Broadcasting
SAP	Session Announcement Protocol
Sat-RMTP	Satellite Reliable Multicast Transport Protocol
SDP	Session Description Protocol
SDR	Session DiRectory tool
SER	Symbol Error Rate
SIP	Session Initiation Protocol
SLA	Service Level Agreement
SNDU	SubNetwork Data Unit
TCP	Transport Control Protocol
TRACK	TRee-ACK based
UDLR	UniDirectionnal Link Routing
UDP	User Datagram Protocol
UDP-lite	Lightweight User Datagram Protocol
UEP	Unequal Error Protection
ULE	Ultra Lightweight Encapsulation
UMTS	Universal Mobile Telecommunication System
VPN	Virtual Private Network
VSAT	Very Small Aperture Terminal
WAN	Wide Area Network

Résumé

La thèse traite du problème de la gestion de la fiabilité pour des transmissions multipoints par satellite reposant sur la norme de diffusion DVB-S. Cette fiabilité est en effet compromise par les fluctuations imprévisibles des conditions de propagation. Des mécanismes mis en œuvre par les protocoles de communications réduisent encore plus la fiabilité. Le travail de cette recherche reconsidère ces derniers afin d'optimiser la gestion de la fiabilité, de la couche physique jusqu'à la couche transport.

Le mémoire présente les principales techniques de fiabilisation mises en jeu dans ce contexte. Il révèle que le rôle du codage correcteur d'erreurs (FEC), déployé conjointement sur la couche physique et sur la couche transport, est primordial. Malheureusement, un manque de coordination entre les mécanismes apparaît. Il provient de l'architecture en couches, héritée des réseaux terrestres, qui rejette systématiquement les unités de données corrompues. Ce rejet peut conduire à perdre une partie non négligeable de l'information pourtant correctement acheminée. Ce constat conduit à proposer une adaptation des techniques existantes, visant à optimiser la coordination entre les différentes couches impliquées dans la transmission, jusqu'à l'utilisateur final. Selon une approche « système », la contribution aborde deux problématiques essentielles pour le développement d'une architecture de transmission optimisée.

La première problématique concerne les concepts à envisager pour concevoir une architecture protocolaire appropriée, au niveau des couches basses. L'architecture proposée prévoit non seulement de reconsidérer l'utilisation des contrôles d'intégrité des données, mais aussi de renforcer la fiabilité, spécifiquement sur les en-têtes des différentes couches protocolaires. Ce mécanisme original est baptisé *Multi Protocol Header Protection* (MPHP). La mise en œuvre pratique de ce mécanisme est également étudiée. Celle-ci est proposée dans le cadre de la méthode *Ultra Lightweight Encapsulation* (ULE), en cours de normalisation, permettant l'encapsulation de datagrammes IP dans les réseaux à diffusion de type DVB.

L'exploitation de l'architecture pour deux services de fiabilité est ensuite discutée au niveau des protocoles de transport. Le premier service, appelé *fiabilité binaire non contrôlée*, peut être offert par un nouveau protocole de transport, *UDP-lite,* visant les applications tolérantes aux erreurs binaires. L'architecture présentée constitue pour ce service un support privilégié. Des résultats obtenus par simulation démontrent que l'amélioration sur le débit offert à l'application peut aller jusqu'à un facteur 5. Un deuxième service, dit *de fiabilité totale*, repose sur l'élaboration d'un procédé de fiabilisation adapté à l'architecture considérée. Ce procédé fait notamment intervenir un algorithme de décodage FEC particulièrement adapté à cette situation. L'évaluation globale de l'approche montre que pour un service identique, il est possible de transmettre 5 à 30 fois moins de données que dans l'architecture conventionnelle.

Le mémoire est organisé de la façon suivante. Le chapitre 1 présente le cadre d'étude de la thèse et la problématique générale de la gestion de la fiabilité. Le chapitre 2 fait l'état de l'art des techniques liées à la fiabilisation, dans les communications réseaux et les transmissions par satellite. Le chapitre 3 évalue les performances des techniques les plus appropriées au référentiel présenté. Le chapitre 4 décrit l'élaboration d'un outil de mesure lié à fiabilité. Le chapitre 5 propose une architecture protocolaire dérivée de l'architecture classique pour améliorer les performances au niveau transport. Le chapitre 6 traite des mécanismes à mettre en œuvre au niveau transport, pour satisfaire les besoins applicatifs de fiabilité. La conclusion résume les différentes contributions de cette recherche puis propose quelques prolongements à ce travail.

Chapitre 1

Contexte général de la recherche

INTRODUCTION

Ce premier chapitre présente le cadre de la recherche et les hypothèses de travail retenues dans cette thèse. Il s'agit d'évoquer ici les principales motivations ayant guidé ce travail. Pour cela, nous présenterons d'abord les protocoles et mécanismes mis en jeu avec IP Multicast, avant de nous intéresser au satellite en tant que support de cette technologie. Après avoir soulig0né l'apport du satellite aux communications multipoints, nous définirons le concept de fiabilité, et nous expliquerons comment plusieurs niveaux de fiabilité peuvent être atteints. Enfin, dans la dernière partie de ce chapitre, nous établirons le cadre de référence de ce travail et nous poserons les questions auxquelles nous nous efforcerons de répondre dans ce mémoire.

1.1 LES COMMUNICATIONS MULTIPOINTS ET IP MULTICAST

1.1.1 *Présentation générale*

1.1.1.1 Définitions

Une communication *multipoints* (*multicast* en anglais) est une communication à destination d'un groupe de stations. Ces stations seront appelées les *récepteurs* ou *les membres du groupe multipoints*. Selon le nombre de stations (appelées *sources*) émettant du trafic vers ce groupe, la communication est qualifiée plus précisément de *point à multipoints* (une seule source), ou de *multipoints à multipoints* (plusieurs sources). La plupart des communications évoquées dans ce mémoire seront des communications point à multipoints, alors simplement désignées par *multipoints*.

Ce mode de communications se démarque du mode *point à point* (*unicast*) utilisé plus traditionnellement dans les réseaux, où seules deux stations participent à la communication. Il existe un troisième mode de communication, appelé diffusion (*broadcast*) dans lequel les données sont transmises à toutes les stations du réseau (avec certaines restrictions toutefois). Les communications multipoints peuvent donc être vues comme une généralisation des deux autres modes de transmission, puisque le nombre de stations impliquées peut être quelconque. Celui-ci peut même varier au cours de la communication. Toutefois, à la différence de la diffusion, les stations ne peuvent recevoir les données qu'à la condition de s'être préalablement enregistrées dans le groupe.

Pour être exhaustif, un dernier mode de transmission nommé *anycast* a été crée, notamment avec la nouvelle version du protocole IP, IPv6. Ce mode permet de limiter la transmission vers n'importe lequel des membres d'un groupe, et pour des raisons évidentes d'efficacité, c'est généralement le membre le plus proche de la source qui est choisi. L'usage du mode d'adressage

anycast reste pour l'instant limité aux informations de signalisation échangées par certains routeurs.

La définition suivante peut être donné à l'expression *communication multipoints* :

« Service de transmission offert par le réseau pour acheminer simultanément des données à un groupe de plusieurs stations enregistrées au préalable dans ce groupe ».

Cette définition appelle deux remarques. D'abord, le terme *simultanément* signifie que les données sont transmises au même instant par les équipements d'interconnexion des réseaux, mais pas forcément reçues simultanément par les membres du groupe, ceux-ci n'étant pas à la même « distance » des sources. D'autre part, cette définition exclut de l'étude ce qui est couramment appelé le mode *multipoints applicatif*, lequel concerne des réseaux de recouvrement (*Overlay Networks)* tels les réseaux *Peer to Peer*. S'il est vrai que ce mode permet d'accélérer la communication entre un ensemble de stations (grâce à une coordination efficace), la duplication des données par les membres du groupe ne peut qu'être réalisée en transmettant en parallèle les mêmes données sur plusieurs connexions.

Enfin, nous utiliserons l'expression *voie aller* pour désigner le sens de transmission principal de la communication, de la source vers les membres du groupe. La *voie de retour* indiquera le sens inverse de transmission, des membres du groupe vers la source.

1.1.1.2 Naissance du besoin du service

L'augmentation permanente du nombre de stations interconnectées, notamment par le réseau fédérateur Internet, fait que la transmission d'un même objet (p.ex. un fichier) est souvent demandée à une même station dans des intervalles de temps restreint. Par exemple, lorsqu'une mise à jour d'un logiciel populaire (*Services Packs* pour Windows, nouvelles distributions de Linux ou autres) devient disponible sur un serveur de contenus, celle-ci peut être téléchargée plusieurs millions de fois en quelques semaines.

Le transfert de l'objet, segmenté par le réseau en unités de données (ou paquets) de plus petite taille, monopolise des ressources pour son acheminement. Typiquement, on distingue trois types de ressources : celles dédiées aux traitements (consommation de temps CPU, mémoires) aux niveaux des stations, celles mises en œuvre par les équipements d'interconnexion des réseaux (traitements, remplissages des mémoires tampons des files d'attente), et celles concernant l'utilisation de la bande passante des liaisons.

Dans un souci d'économie bien légitime, de nombreuses technologies de niveau liaison (*Asynchronous Transfer Mode* (ATM), *Fiber Distributed Data Interface* (FDDI), Ethernet) intègrent un service de communications multipoints. Grâce à celui-ci, les ressources nécessaires aux transmissions d'unités de données identiques peuvent être diminuées de manière substantielle. Nous exposons le fonctionnement général des communications multipoints dans la partie suivante.

1.1.1.3 Principe et définition de l'arbre de diffusion multipoints

La situation de la Figure 1-1 représente la transmission d'un objet en mode multipoints (en flèches noires) pour un groupe composé de 5 stations. Le mode est comparé aux transmissions en mode point à point qui seraient nécessaires pour faire parvenir l'objet aux 5 mêmes stations. Ceci implique que les mêmes unités de données de l'objet (en flèches vertes) doivent transiter sur chaque lien, alors qu'elles ne sont transmises qu'une fois par lien en mode multipoints. Cela est réalisé en les répliquant sur les interfaces de sorties des équipements d'interconnexion des réseaux. Ce mode permet donc d'optimiser les ressources du réseau de trois façons :

- la bande passante nécessaire est réduite : par exemple, on peut voir sur la Figure 1-1 que le lien en amont est cinq fois moins sollicité par le mode multipoints ;
- la charge due aux traitements des unités de données est réduite : la source n'émet qu'une copie par unité de données et le filtrage au niveau des récepteurs est allégé ;
- la probabilité de congestion des files d'attentes (et donc de perte des unités de données) est réduite, particulièrement au voisinage de la source.

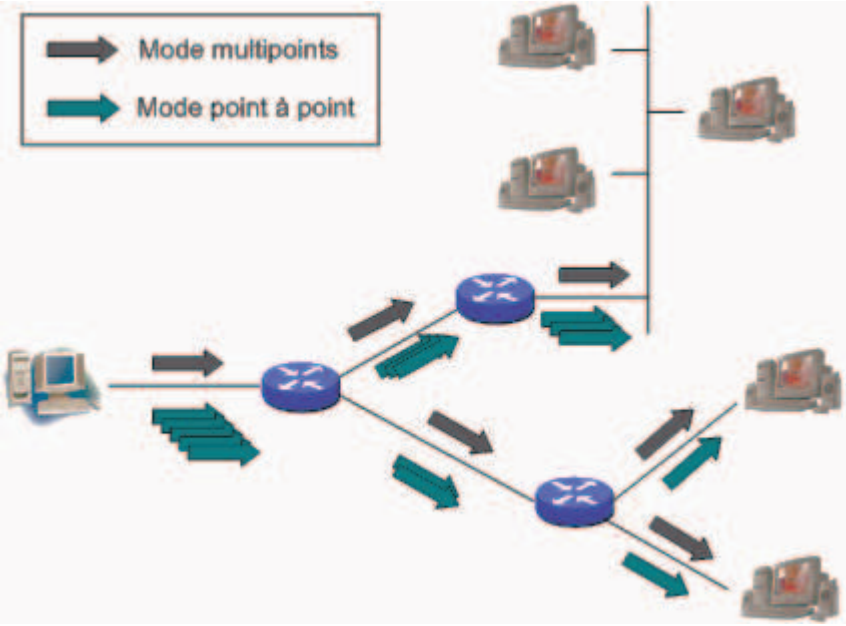

Figure 1-1: Illustration du mode de communication point à point et du mode multipoints

La structuration topologique du groupe implique *un arbre de diffusion multipoints* pour chaque source. Cet arbre est constitué :

- d'une racine, qui est la station source. Elle représente par exemple un serveur de contenus ;
- de nœuds, qui sont les éléments d'interconnexion des réseaux supportant le mode multipoints et la réplication des unités de données sur leurs interfaces de sorties ;
- de feuilles, qui sont l'ensemble des récepteurs. Les feuilles peuvent parfois être des nœuds. C'est le cas dans certaines communications multipoints à multipoints.

Améliorer l'utilisation du réseau n'offre pas *a priori* un bénéfice direct pour l'utilisateur en tant que membre du groupe (sa connexion n'est pas « plus rapide »), mais cela est rentable pour les opérateurs réseau. Cette remarque vaut aussi pour les gros serveurs de contenus qui jouent souvent le rôle de la source, puisque les ressources qu'ils doivent mettre à disposition sont moins importantes grâce au service multipoints. Au final, ces économies peuvent se répercuter sur les utilisateurs (si l'opérateur réseau le décide) et de nouveaux services peuvent leur être offert.

1.1.2 *Les applications multipoints*

De nombreuses applications tirent aujourd'hui parti des services de communications multipoints. Nous verrons dans ce paragraphe comment répertorier ces applications en catégories. Nous pourrons ensuite, pour chacune de ces catégories, distinguer quelles sont leurs principales caractéristiques.

1.1.2.1 Les catégories d'applications

- La distribution programmée de média, audio ou vidéo (MEDIA) ;
- L'éducation à distance (EDU) ;
- La mise à jour de logiciels populaires (UPDATE) ;
- L'alimentation de serveurs de cache (CACHE) ;
- Le *push* régulier de données (mises à jour de faibles volumes de données, p.ex. bulletins météo, résultats sportifs, synchronisation, etc…) (PUSH).
- Le partage d'un espace de travail (WORK);
- Les discussions au sein d'un groupe (CHAT);
- Les simulations interactives distribuées (SIM) ;
- Les jeux multi-joueurs (GAME) ;
- La téléconférence audio/vidéo (CONF) ;
- La mise en commun de données (SHARE) ;
- Le traitement parallèle distribué (PROC).

1.1.2.2 Les principaux critères de classements

Chacune de ces catégories regroupe des caractéristiques communes. Principalement, on peut évoquer :

- La nature de la communication : point à multipoints (1 vers N) ou multipoints à multipoints (M vers N). Il peut s'agir de mise à jour de logiciels ou de base de données, de service de diffusion de média (son, vidéo), où le nombre de récepteurs peut être élevé (plusieurs milliers). Au contraire, dans les communications multipoints à multipoints, les tailles de groupe sont plus réduites. L'application la plus représentative de ce type de communication est la téléconférence audio/vidéo.
- Les services requis :
  - Fiabilité totale (cf. 1.1.3.2.1.1 pour la définition des services de fiabilité). Suivant la nature des données à transmettre, certains transferts requièrent un acheminement sans erreur. Dans le contexte réseau, cela signifie qu'aucune perte ou corruption de données n'est tolérée par le niveau applicatif du ou des récepteurs. Ce service est appelé *fiabilité totale*. Par exemple, on imagine aisément qu'un service de mise à jour de cotations boursières serait inexploitable s'il tolérait les modifications de quelques chiffres de taux d'échanges. Au contraire, pour des applications diffusant des bandes-annonces de films, la corruption d'une partie de l'image pendant un instant n'a pas d'effet conséquent sur le service. Dans ce dernier cas, il n'est pas nécessaire (voire pas souhaitable) d'obtenir un service de fiabilité totale.
  - Respect des contraintes temporelles (contrôle du délai et/ou de la gigue). Des applications comme le partage d'espace de travail sont utiles quand leur réactivité est suffisante. Il peut être souhaitable de recourir à des mécanismes de Qualité de Service (QdS) pour s'assurer du type de service offert par le réseau traversé.

- Débit élevé (au moins une centaine de kilobits par seconde). Ce besoin est parfois problématique dans un réseau comme Internet, où la bande passante effectivement disponible peut varier de façon importante.

	1 vers N	M vers N	Besoin de fiabilité	Respect des contraintes temporelles	Besoin en débit élevé
MEDIA	x			x	x
EDU	x		o	x	o
UPDATE	x		x	o	
CACHE	x	o	x		
PUSH	x	o	x		
WORK		x	o	x	
CHAT		x	x	x	
SIM		x	o	x	x
GAMES		x		x	x
CONF		x		x	x
SHARE		x	x		
PROC	o	o	x		

Légende : x : applicable ; o : applicable dans certains cas

Tableau 1-1 : Les caractéristiques des applications multipoints

Le Tableau 1-1 montre les caractéristiques principales des différentes catégories d'application, et leur diversité. Nous verrons dans la suite que celle-ci est, en parti, à l'origine de la difficulté de normalisation des protocoles de transport multipoints.

1.1.2.3 Exemples de logiciels supportant le mode multipoints

1.1.2.3.1 Les outils applicatifs du MBone

Pour promouvoir l'utilisation et les bienfaits qu'apportent les communications multipoints et pour offrir aux développeurs d'applications et de protocoles multipoints un environnement de test, un réseau expérimental virtuel, *Multicast Backbone* ou MBone, a été créé par plusieurs chercheurs américains (V. Jacobson, S. Deering, S. Casner) en 1992 [MBO]. Le MBone est un réseau de recouvrement construit sur le réseau Internet, dans lequel chacune des stations a la capacité de recevoir (et d'émettre) du trafic multipoints de (et vers) l'Internet public. Le MBone diffuse régulièrement de grands évènements ; par exemple des missions de la navette *Space Shuttle* de la NASA, ou une partie d'un concert des *Rolling Stones*. Des diffusions en temps réel des réunions de groupes de travaux de l'*Internet Engineering Task Force* (IETF) sont également retransmises.

Avec le MBone, des outils applicatifs gratuits ont été développés pour pouvoir capter les retransmissions proposées. La suite de logiciels multimédia VIC/VAT/RAT composée des outils de vidéoconférence, d'audioconférence, et de *streaming* audio, est l'une des plus populaires. Elle est du reste disponible sur de nombreuses plate-formes. D'autres outils de collaboration comme WBD (*WhiteBoard*) ou NTE (*Network Text Editor*) sont aussi disponibles, voir [MBS].

Pour le transfert de données demandant une fiabilité totale, peu de logiciels déployés spécifiquement pour le MBone existent. En revanche, de nombreuses implémentations de protocoles sont disponibles, dont certaines émanent de protocoles en cours de normalisation à l'IETF. À ce jour, au moins trois équipes de recherche connues ont réalisé ce type

d'implémentation : l'INRIA (librairie MCL) [MCL], Tampere University of Technology (projet MAD/TUT) [MAD], et un partenariat entre Nokia et l'Université de Brême (code propriétaire).

1.1.2.3.2 Produits commerciaux

Les logiciels de lecture ou de diffusion de flux multimédia ont été adaptés depuis quelques années pour offrir ou supporter le mode multipoints. Le Tableau 1-2 présente quelques-uns des produits les plus connus, proposés par de grandes sociétés [HEL], [REA], [DAR], [QTS], [QT], [WMS], [WMP].

Société	Serveurs de diffusion multipoints	Clients compatibles
Real Networks	Real Server Helix	Real One
Apple	Darwin Streaming Server ; QuickTime Streaming Server	QuickTime Player ; QuickTime Pro
Microsoft	Windows Media Services	Windows Media Player

Tableau 1-2 : Quelques outils multimédia exploitant le mode multipoints

En ce qui concerne les solutions commerciales offrant des garanties de fiabilité totales aux communications multipoints, deux produits se distinguent particulièrement : DF Core de Digital Fountain [DF] et SmartPGM de Tibco [TIB].

1.1.3 ***IP Multicast***

IP Multicast a été choisi pour interconnecter les différents types de réseaux locaux en vue d'un service de communication multipoints unifié. La raison de ce choix est la même que pour les modes point à point : quelle que soit la nature du réseau sous-jacent, IP offre aux couches supérieures un service unifié. Aussi, dans la suite de l'exposé, nous restreindrons la capacité de diffusion multipoints du réseau à IP Multicast, pour ne pas nous préoccuper des technologies employées (hormis sur la liaison satellite).

Le service minimum déployé par les éléments d'interconnexion des réseaux (*i.e.* les routeurs IP Multicast) dupliquant les datagrammes n'est pas suffisant pour permettre la communication de groupe, ni pour satisfaire la plupart des besoins applicatifs. Des techniques complémentaires sont nécessaires. Celles-ci peuvent être différenciées selon qu'elles agissent plutôt avant la communication multipoints, pour mettre en place une configuration adéquate du réseau, ou parallèlement à la communication, pour la contrôler.

1.1.3.1 Services liés à la configuration du réseau

1.1.3.1.1 Le système d'adressage dans IP Multicast

Internet Protocol (IP), brique essentielle de la construction des réseaux Internet, propose depuis 1989 une extension pour le service multipoints : IP Multicast [Dee89]. IP Multicast agit au niveau 3 de la couche du modèle en couches de l'*Open System Interconnexion* (OSI), sur les datagrammes IP.

IP Multicast permet la communication multipoints en assignant au groupe de stations une adresse IP commune. C'est cette adresse qui est indiquée dans le champ *Adresse de destination* des datagrammes envoyés vers le groupe. Chaque membre utilise cette adresse IP pour accepter les datagrammes comme s'il s'agissait de sa propre adresse IP locale. Sous IPv4, toutes les adresses Multicast sont regroupées dans les adresses dites de classe D. La classe d'adresse D s'étend entre

224.0.0.0 et 239.255.255.255 (les trois premiers bits de l'adresse sont toujours à 1, suivi d'un 0). Le Tableau 1-3 montre comment sont réparties ces adresses dans l'espace d'adressage IPv4.

Classe d'adresses	8 bits de poids fort de l'adresse IPv4	Usage
Classe A	0xxxxxxx	Réseaux de classe A
Classe B	10xxxxxx	Réseaux de classe B
Classe C	110xxxxx	Réseaux de classe C
Classe D	1110xxxx	Adresses Multicast
Classe E	1111xxxx	Adresses expérimentales

Tableau 1-3 : Les classes d'adresses IP

Notons que certaines de ces adresses sont réservées à des protocoles spécifiques (224.0.0.1, 224.0.0.2, 224.0.0.4, 224.0.0.9, 224.0.0.13, plus certaines autres). De plus, une station peut appartenir, si l'utilisateur le désire, à plusieurs groupes.

1.1.3.1.2 Routage

1.1.3.1.2.1 Routage IP Multicast en périphérie de réseau : IGMP

Dans le cas général, quand une station veut recevoir les données émises à un groupe, il ne lui suffit pas « d'écouter » le réseau sur l'adresse du groupe. En effet, au préalable, le réseau doit connaître les stations désirant devenir membres du groupe, afin de leur faire parvenir le flux IP Multicast. *Internet Group Management Protocol* (IGMP) est le moyen pour ces stations d'en informer le réseau. La signalisation nécessaire est échangée entre les stations IP Multicast et le ou les routeurs IP Multicast de leur réseau local. IGMP est réservé aux réseaux IPv4, mais il a son équivalent pour IPv6 : *Multicast Listener Discovery* (MLD). Le reste du routage (entre les routeurs) est traité par d'autres protocoles.

Dans la version originale d'IGMP décrite dans le RFC 1112 d'IP Multicast [Dee89], chaque station IP Multicast doit s'enregistrer auprès d'un des routeurs Multicast de son réseau local (appelé *Designated Router*, DR) pour lui signifier qu'il veut recevoir les données émises vers un groupe particulier. Pour annuler son abonnement et sortir du groupe, il suffit que le récepteur arrête de répondre aux requêtes de son DR. Dans la version 2 [Fen97] aujourd'hui la plus répandue, la réactivité du DR est améliorée par rapport à IGMPv1, car les récepteurs quittent le groupe en le signalant de manière explicite. Ceci permet aux routeurs d'arrêter la diffusion du trafic multipoints plus rapidement. Enfin, avec IGMPv3 encore peu déployé [Cai02], les récepteurs peuvent indiquer s'ils souhaitent inclure ou exclure dans leur abonnement une source particulière émettant vers le groupe.

1.1.3.1.2.2 Routage IP Multicast en cœur de réseau

Pour toute communication de groupe au-delà du réseau local de la source, le réseau doit être en mesure d'identifier les routeurs pouvant participer à la réplication des datagrammes IP. Pour cela, ils doivent échanger des informations au travers d'un protocole de routage, dans le but de construire l'arbre de diffusion multipoints. Pour être efficace, le protocole doit être robuste aux changements de topologie, tout en évitant de générer trop de signalisation. Le protocole de routage doit aussi prendre en compte le fait que les routeurs n'ont qu'une connaissance partielle du réseau, celui-ci étant administré par domaines autonomes (*Autonomous System*, AS).

Il existe deux familles (appelées *modes*) de protocole de routage intra-domaine ayant deux approches opposées, selon la densité des récepteurs : le mode dense et le mode clairsemé. Dans le

premier mode, la densité des récepteurs est supposée élevée. L'arbre de diffusion est construit par inondation du trafic (*flooding*) puis par élagage progressif des branches (*pruning*). Ici, il est plus rentable d'agir de cette façon puisque la probabilité de présence des récepteurs est élevée. Au niveau de chaque routeur, l'arbre initial comprend toutes ses interfaces de sorties, et le flux IP multicast entrant est répliqué sur toutes ses interfaces, sauf celle d'entrée. Au fur et à mesure que le routeur reçoit les informations (des routeurs voisins et des stations), il supprime les interfaces de sorties pour lesquelles le groupe n'a pas de membres. Les principaux protocoles représentatifs de ce mode sont *Distance Vector Multicast Routing Protocol* (DVMRP) [Wai98], *Multicast Open Shortest Path First* (MOSPF) [Moy94] et *Protocol Independent Multicast – Dense Mode* (PIM-DM) [Dee99]. Ils sont adaptés aux réseaux peu étendus, de type *Local Area Network* (LAN) ou *Metropolitan Area Network* (MAN).

Dans le mode clairsemé, les protocoles de routage (*Protocol Independent Multicast – Sparse Mode* PIM-SM [Fen04], *Core-Based Tree* CBT [Bal97]) agissent à l'opposé du mode dense. La densité des récepteurs est ici supposée faible. Initialement, les tables de routage de chaque routeur sont vides. Elles sont ensuite progressivement remplies, à mesure que les routes vers les récepteurs sont déterminées. De plus, une partie de l'arbre de diffusion est partagée par toutes les sources émettant vers le groupe. Le sommet de l'arbre commun à toutes les sources est appelé le *RendezVous Point*, RP. L'avantage d'un tel routage réside dans la facilité de sa gestion (car un nombre minimal de tables est manipulé), dans la vitesse d'établissement de l'arbre, et dans l'absence de gaspillage qui serait induit par l'inondation. Il convient aux larges réseaux de type *Wide Area Network* (WAN).

1.1.3.1.3 Annonce de sessions

Une *session de transport multipoints* (ou plus simplement *session*) est l'ensemble des caractéristiques d'une communication de niveau transport s'appuyant sur un service de diffusion multipoints. Contrairement à une majorité de communications point à point, chaque session multipoints débute à l'initiative de la source, et non à celle des récepteurs. Aussi, comment faire connaître l'existence d'une telle session à l'avance pour que les stations puissent décider de s'abonner au groupe vers lequel elle sera diffusée ?

Dans un réseau public comme Internet, les communications point à point ont besoin d'adresses pour identifier les stations. Grâce à des service d'annuaires (p.ex. moteur de recherche), il est possible pour un utilisateur de trouver le nom d'un serveur inconnu en fonction de mots-clés tapés (normalement en rapport avec le contenu du serveur !). Ensuite, le service de nommage *Domain Name System* (DNS), commun à toutes les machines communiquant par IP, permet de résoudre la correspondance entre le nom d'une station et son adresse IP.

Pour les communications multipoints, ce principe de service par annuaire peut aussi être utilisé pour identifier des sessions. Il est ainsi possible de diffuser par un canal préétabli et connu de tous les différents paramètres des sessions : l'adresse du groupe, la nature de l'information diffusée, le type de contenu, le(s) codage(s) multimédia, le débit minimum, l'horaire des sessions, les numéros de ports, etc. On appelle cela *l'annonce de sessions*. Dans le cadre du MBone, l'annonce est réalisée par l'intermédiaire de l'outil *Session DiRectory tool* (SDR). Grâce à SDR, chaque session publique programmée est consultable par tous les membres du MBone. Il est également possible d'utiliser SDR sur un réseau privé. Il faut noter que SDR est une implémentation regroupant l'accès aux services des protocoles SAP (*Session Announcement Protocol*) [Han00], SDP (*Session Description Protocol*) [Han98] et SIP (*Session Initiation Protocol*) [Han99].

1.1.3.2 Contrôles des transmissions

Après l'établissement de l'arbre de diffusion multipoints et la configuration des paramètres adéquats par les récepteurs, la communication peut effectivement avoir lieu. Cela étant, parallèlement, d'autres services sont fréquemment requis par les besoins applicatifs : services de fiabilité, contrôle de congestion, respect des contraintes de débits et de délais, ou encore besoin en termes de sécurité.

1.1.3.2.1 Services de fiabilité

1.1.3.2.1.1 Définitions

Un système est dit fiable s'il maintient « son niveau de service dans des conditions précises et pendant une période déterminée » [Iso91]. Pour quantifier le niveau de fiabilité, on lui associe souvent une mesure, qui spécifie la probabilité que le système fonctionne dans les conditions et période données.

Dans notre étude, cette définition est délicate à appliquer car on considère que le système de transmission fonctionne toujours selon ce que l'on attend de lui. En revanche, des éléments externes viennent le perturber, se manifestant classiquement par la perte d'unités de données de l'objet à transférer. La notion de fiabilité est en fait dépendante du domaine dans lequel on souhaite la mesurer (cf. chapitre 2).

En termes de services de transport, on peut rapprocher la fiabilité requise à un taux de pertes maximum que l'application peut tolérer. Cette définition peut sembler satisfaisante dans un premier temps, mais en réalité elle ne spécifie pas sur quoi portent les pertes. Suivant la sémantique des données, et dans les cas favorables, cela peut concerner les unités de données du service de transport. Dans d'autres cas, par exemple avec les codages multimédias hiérarchiques, la perte d'une donnée des couches de base entraînera la perte des données des couches d'amélioration. Il est alors impossible de spécifier avec un seul nombre la fiabilité requise [Roj00]. Dans le contexte des communications multipoints, la présence de plusieurs récepteurs rend notre tentative de définition encore plus délicate. Certains récepteurs pouvant recevoir correctement les informations et d'autres non, il faut encore généraliser le type de service de fiabilité en précisant par exemple une garantie de type statistique pour le groupe.

Cinq classes de service de fiabilité peuvent être définies au niveau de l'interface entre transport et application. L'application doit spécifier sous quelle classe elle désirera être servie, et éventuellement avec quels paramètres. Le choix de la classe fixera alors dans quelle mesure la reprise d'erreur par retransmission (ou *Automatic Repeat reQuest*, ARQ) sera employée.

- *Service à fiabilité non contrôlée*. Avec ce service, les données sont transmises sans que la station source ne puisse avoir accès à une mesure de fiabilité faite par les récepteurs. À cause des besoins applicatifs ou pour d'autres raisons (absence de voie de retour ou choix de ne pas l'utiliser), la reprise d'erreur est impossible. Il appartient donc à l'application de mettre en œuvre les moyens de pallier si nécessaire la perte d'informations.
- *Service à fiabilité statistique*. Ce service n'apporte pas de garantie individuelle aux récepteurs, mais il permet globalement de connaître les taux d'erreurs (ou de pertes) sur les unités de données. Quand une voie de retour est disponible, les récepteurs peuvent fournir une estimation de la qualité de la transmission de bout en bout, et la source peut effectuer les actions nécessaires. Toutefois comme précédemment, il n'y a pas de reprise d'erreur. Une mesure possible de la fiabilité est le taux moyen de pertes des unités de données pour l'ensemble du groupe.

- *Service à fiabilité partielle pour applications à temps contraint*. Certaines applications multimédia ont des contraintes temporelles qui imposent l'arrivée des unités de données sous un délai borné ou avec une gigue maximale. Les techniques d'ARQ sont alors possibles mais limitées selon les contraintes.
- *Service à fiabilité partielle garantie*. D'autres applications multimédia n'ont pas de contraintes temporelles strictes, mais leur premier critère de satisfaction est la qualité de restitution du média. Une mesure possible de la fiabilité est le taux d'erreur maximum admissible pour chaque membre du groupe.
- *Service à fiabilité totale garantie*. Ce service assure que chacun des membres du groupe recevra le ou les objets transmis de manière parfaite. C'est un service important, car il est exigé par un grand nombre d'applications.

1.1.3.2.1.2 Techniques de fiabilisation au niveau transport

Lorsque la fiabilité d'IP Multicast n'est pas suffisante pour satisfaire l'application, une amélioration de la fiabilité, ou *fiabilisation*, est nécessaire. Ce besoin a été à l'origine de plusieurs dizaines de propositions, plus ou moins avancées au moment de la création du groupe de travail *Reliable Multicast Transport - Working Group* (RMT-WG) à l'IETF [RMT]. À partir de ces propositions, le groupe entreprend de normaliser des solutions. Comme nous le verrons dans les chapitres suivants, la raison principale de cette harmonisation tardive provient de la variété des situations rencontrées, qui a empêché de trouver une solution générique satisfaisante. Deux approches sont maintenant retenues par le RMT. La première repose sur un protocole utilisant des acquittements négatifs explicites (*Negative ACKnowldgement*, NACK) pour signaler la perte d'informations à la source, la seconde sur un protocole sans voie de retour, utilisant un code correcteur d'erreur (*Forward Error Correction*, FEC) déployé de manière préventive. Nous reviendrons largement sur ce point dans la seconde partie du chapitre 2.

1.1.3.2.2 Contrôle de congestion

Outre la fiabilité, le niveau transport peut aussi fournir un service de contrôle de congestion. Suivant la nature publique ou privée du réseau traversé et le type de service offert en termes de débit pour le trafic (*best effort* ou autre), un contrôle de congestion est désirable ou non. À ce jour, seule une proposition de contrôle de congestion a été retenue par le RMT-WG : Wave and Equation Based Rate Control (WEBRC) [Lub04].

1.1.3.2.3 Qualité de Service

De nombreuses façons d'assurer une qualité de service (QdS) existent pour les communications point à point, notamment au niveau IP avec IntServ et DiffServ, mais aussi grâce à d'autres technologies comme ATM. Indépendamment du moyen choisi, le SLA (*Service Level Agreement*) passé contractuellement entre l'opérateur du réseau et le client fixe le niveau de service qui peut être demandé. Assurer aujourd'hui la qualité de service pour les communications multipoints n'est pas complètement résolu mais de nombreux travaux de recherche s'y attachent, p.ex. [Fal98], [Fei00], [Che00], [Cui02], [Bra97]. Parmi les grandes idées citées dans ces études, on trouve le routage à qualité de service, le support du protocole de réservation des ressources *Resource reSerVation Protocol* (RSVP) et l'ingénierie de trafic multipoints avec DiffServ et *MultiProtocol Label Switching* (MPLS).

1.1.3.2.4 Sécurité

Sécuriser les connexions devient un besoin de plus en plus fréquent, y compris pour le trafic multipoints. En particulier, pour la distribution de média sur Internet, les fournisseurs de services et d'accès à Internet (*Internet Service Provider*, ISP) ont intérêt à restreindre à leurs seuls clients la possibilité d'exploiter les données qu'ils mettent à disposition. La méthode, choisie traditionnellement par cryptage/décryptage par clés, peut être mise en œuvre de bout en bout (solutions propriétaires applicatives, solutions de niveaux transport, techniques se basant sur *IP Security protocol* (IPSec) [IPS]), ou uniquement sur un type de lien (cf. par exemple [Ets03], [Doc04], [ATM01]). Seul le contexte précis permet de déterminer le meilleur choix en fonction des besoins.

1.1.4 ***Limitation du déploiement d'IP Multicast***

La diversité des besoins applicatifs conjuguée aux multiples réseaux traversés (dont Ethernet, ATM, les réseaux sans-fil, le satellite ou le câble) fait qu'aujourd'hui un large éventail de choix techniques est envisageable pour le déploiement des services multipoints. Pourtant, hormis sur les réseaux publics visant l'éducation et la recherche (p.ex. RENATER en France), ou sur des réseaux privés, les utilisateurs n'ont pas la possibilité d'émettre ou de recevoir de flux IP Multicast au-delà du réseau local. L'absence du support d'IP Multicast natif des ISP est liée à plusieurs raisons.

D'abord, le choix des solutions techniques est complexe. Afin de supporter l'une de ces solutions, les ISP devraient mettre à jour leurs équipements et d'autre part, les protocoles de bout en bout devraient être adoptés par tous. Il y a également un problème important concernant la gestion de la sécurité (quelles stations ont le droit d'émettre ou de recevoir les flux multipoints, et comment y parvenir). Un récent groupe de travail IETF (2003), le MSEC [MSE], s'attache à la normalisation de ce point. Un troisième argument vient du fait qu'il est toujours possible de faire transiter du trafic IP Multicast dans un domaine ne le supportant pas, grâce à une technique de *tunneling* (encapsulation d'IP Multicast dans IP Unicast). Malheureusement, le *tunneling* ferait perdre aux communications multipoints tout son intérêt s'il était déployé massivement. La dernière explication, la plus plausible, est le manque de services exploitant exclusivement la capacité de diffusion multipoints. En l'absence d'une « *killer application* » qui justifierait à elle seule un déploiement massif d'IP Multicast, celui-ci n'est pas véritablement encouragé, ce qui contribue à ne pas développer de nouveaux services.

Aussi, dans la situation actuelle, offrir des services exploitant IP Multicast incite les ISP et les fournisseurs de contenus à proposer des solutions propriétaires. Ces solutions ont tendance à tout intégrer au niveau applicatif. Dès lors, leurs efforts d'études et de développements doivent être rentabilisés. Pour l'utilisateur, cela entraîne une augmentation des prix pratiqués. D'autre part, à moins que l'une de ces solutions ne prévale sur les autres, cela empêche toute interopérabilité avec les solutions concurrentes. À terme, cette situation pourrait même complètement bloquer le développement d'IP Multicast.

Dans ce contexte, le satellite paraît un excellent candidat pour véritablement lancer l'intérêt des communications multipoints. Plus adapté que les réseaux terrestres pour diffuser les informations vers de grands groupes (plusieurs milliers de récepteurs), le satellite bénéficierait d'une harmonisation des mécanismes liés à son intégration avec IP Multicast. Le support d'IP Multicast est en général plus simple avec le satellite, car le nombre d'équipements à traverser pour atteindre l'utilisateur final est comparativement faible. S'il permettait effectivement d'harmoniser l'ensemble des solutions multipoints, le satellite pourrait même contribuer au support d'IP Multicast dans les autres réseaux.

1.2 LE SATELLITE GÉOSTATIONNAIRE, MOYEN DE TÉLÉCOMMUNICATION MODERNE

1.2.1 *Introduction*

1.2.1.1 Définitions

Le satellite relaie des ondes radioélectriques modulées émises et reçues par des équipements terriens que nous appellerons *terminaux satellites*. À ces terminaux satellites sont connectés des équipements d'utilisateur que nous appellerons *stations*. La station peut être un simple ordinateur personnel pourvu de logiciels applicatifs.

Le réseau de stations et de terminaux satellites peut avoir une architecture *maillée,* ou bien *en étoile*. Dans un réseau maillé, les échanges se font directement entre stations ou terminaux satellite et le plus souvent de façon symétrique. Dans un réseau en étoile, les échanges se font par l'intermédiaire d'une station centrale, de type serveur, connectée à un terminal satellite appelé *passerelle (gateway)*. Ces échanges sont alors le plus souvent dissymétriques, avec des débits d'information plus élevés du serveur vers les stations périphériques, et des débits plus faibles des stations périphériques vers le serveur. Les données transmises de la passerelle vers les autres terminaux satellite empruntent la *liaison aller*. Une *liaison retour* peut exister entre ces terminaux et la passerelle. Dans ce cas, le terminal satellite doit pouvoir recevoir *et* émettre.

Les deux sens de communications mettant en relation deux ou plusieurs stations seront désignés par *voie aller* et *voie retour*, conformément à la terminologie employée pour les communications multipoints en 1.1.1.1.

1.2.1.2 Avantages du satellite

Le satellite géostationnaire (GEO) est considéré depuis sa création comme un support privilégié pour certains types de communications. Malgré des coûts d'investissement élevés, ses avantages les plus significatifs sont :

- Une grande surface de couverture, à l'échelle d'un ou deux continents pour un seul satellite GEO. Ceci recouvre :
  - L'ubiquité. Les distances entre terminaux sont « effacées », la communication s'effectue en un ou deux bonds au maximum, suivant le service et les choix techniques (satellite à charge utile transparente ou non).
  - La simultanéité des transmissions. De la même manière qu'une station radio diffuse ses programmes pour un nombre illimité de récepteurs, les communications par satellite peuvent être captées simultanément par un nombre quelconque de terminaux.
- La facilité et la rapidité de déploiement, en particulier en ce qui concerne les terminaux satellite de type *Very Small Aperture Terminal* (VSAT).

Naturellement, ces deux points sont la clé des services qui ont fait le succès du satellite GEO. Dans cette partie, nous évoquons ceux qui ont permis au satellite de s'imposer parmi les autres technologies de télécommunications. Nous expliquons ensuite comment les communications IP sur satellite ont été intégrées aux normes *Digital Video Broadcasting* (DVB). Nous discuterons alors des caractéristiques du satellite et des progrès techniques réalisés par la prochaine génération de satellite qui pourront avoir une influence sur les services IP.

1.2.2 ***Une alternative aux autres liaisons***

1.2.2.1 Services pour terminaux satellite mobiles

Les services pour terminaux satellites mobiles offrent un moyen de communication efficace pour les stations embarquées dans des bateaux, des avions, ou des véhicules terrestres. L'organisation Inmarsat fut la première à proposer un système à couverture mondiale, à l'origine pour transporter de la voix ou d'autres types de données pour des liaisons maritimes. Plus tard, le rôle et les services d'Inmarsat se sont diversifiés, notamment avec le support des communications numériques et l'extension aux systèmes aéroportés.

1.2.2.2 Services pour terminaux satellite fixes

Grâce à sa facilité de mise en œuvre en l'absence d'infrastructure terrestre, le satellite s'est aussi montré rentable comme moyen de télécommunications pour relier des points fixes. Cette catégorie de services représente à l'heure actuelle la plus grande part de marché. De nombreuses applications tirent parti des avantages que le satellite GEO procure.

1.2.2.2.1 Téléphonie longue distance, réseaux VSAT

Historiquement, la téléphonie à longue distance par satellite a été l'un des premiers services dont le grand public a pu bénéficier, par exemple pour connecter l'Europe et les États-Unis. En effet, avant la généralisation de la technologie à base de fibres optiques, le coût d'investissement des câbles sous-marins transatlantiques était élevé, en regard de leur faible capacité. Depuis ce temps, la situation a évolué, mais le satellite reste une solution attirante pour assurer des liaisons dans des zones géographiques reculées ou économiquement défavorisées. L'installation de petites terminaux de type VSAT permet aussi d'établir rapidement un point d'accès lors de situation d'urgence (p.ex. liaison téléphonique dans une zone ravagée par un tremblement de terre).

1.2.2.2.2 Systèmes numériques de reportages d'actualités par satellite

Dans ces systèmes, les stations et terminaux satellite sont transportables, généralement par camionnette. Les équipements sont mis en service rapidement après leur arrivée sur les lieux de reportage afin de permettre la diffusion de données. Une normalisation de ces systèmes est proposée par l'ETSI depuis 1997, sous le nom DVB-DSNG (*DVB - Digital Satellite News Gathering*) [Ets99].

1.2.2.2.3 Interconnexion de réseaux, VPN et accès à Internet

L'aptitude du satellite à délivrer des données à haut débit (quand les ressources ne sont pas partagées) est exploitée dans des offres appelées *trunking*. Dans ces services, la liaison satellite constitue une dorsale (*backbone*) pour le réseau (réseau IP d'un ISP par exemple). En amont de la dorsale, de multiples connexions bas débit (données ou voix) sont agrégées dans un même flux multiplexé à haut débit, puis démultiplexé après le lien satellite. Cette solution a l'avantage de pouvoir relier deux points distants de plusieurs milliers de kilomètres tout en offrant facilement des garanties de QdS. Par exemple, un ISP peut étendre son réseau à des zones non couvertes par des réseaux câblés à haut débit (petites villes ou villages), pour ensuite desservir localement ses abonnés grâce à un réseau complémentaire.

Outre le *trunking*, les réseaux par satellite peuvent aussi servir de support aux réseaux privés virtuels (V*irtual Private Network* (VPN)). Les VPNs permettent par exemple l'interconnexion sécurisée de sites disants d'une entreprise, grâce au cryptage de données dans des tunnels qui traversent des domaines publics de l'Internet.

Concernant l'accès à Internet pour les particuliers, le satellite est de plus en plus en compétitif face aux autres types d'accès (dont *Asymmetric Digital Subscriber Line*, ADSL). Néanmoins, il impose de lever un verrou important : celui du canal de retour, également indispensable pour d'autres services IP. Une solution élégante pour contourner ce problème fut d'adopter le même type de stratégie que pour les services interactifs avec DVB grâce aux réseaux terrestres. La méthode *UniDirectional Link Routing* (UDLR) [Dur01], normalisée par l'IETF, spécifie l'émulation d'une communication bidirectionnelle au niveau de la couche liaison. Elle impose toutefois aux utilisateurs de disposer d'un abonnement auxiliaire chez un autre ISP (opérant par exemple sur le réseau téléphonique) ce qui relève le coût global de l'accès.

L'intégration du lien satellite dans un réseau terrestre pose un problème de performances, notamment avec le protocole TCP. Bien que de nombreuses adaptations de TCP aient été prévues pour le satellite (cf. [All00]), celles-ci ne sont généralement efficaces que sur des réseaux purement satellitaires. Les solutions d'intégration reposent sur des techniques de *proxying* ou de *spoofing* qui modifient (de façon transparente pour les stations) le comportement des protocoles afin de l'adapter aux caractéristiques du lien satellitaire. Ces solutions peuvent être mises en œuvre au niveau liaison, transport, ou même applicatif.

1.2.2.2.4 Diffusions simultanées

La dernière caractéristique exploitée par le satellite GEO est sa capacité de diffusion naturelle. Les diffusions, aujourd'hui pour la plupart numériques, concernent la télévision, la radio, et depuis quelques années les données informatiques pour le grand public.

1.2.2.2.4.1 Radio

D'abord en mode analogique, la diffusion de la radio par satellite permet essentiellement de recevoir les programmes n'importe où à l'intérieur de la zone de couverture du satellite. Plusieurs diffusions de programmes radio se font actuellement en mode numérique grâce à la norme *Satellite-Digital Audio Broadcasting* (S-DAB) de l'*European Telecommunications Standards Institute* (ETSI), qui permet de restituer un son de qualité CD. Si quelques stations radio sont diffusées gratuitement, comme les programmes de Radio France, des systèmes commerciaux proposent à leurs clients des abonnements pour une multitude de programmes radio numériques.

Par exemple, le système WorldSpace [WSP] diffusera des programmes radio numériques sur plusieurs continents grâce à 3 satellites (dont 2 sont déjà opérationnels, pour l'Afrique et une partie de l'Europe, et pour l'Asie). Un système similaire, XM Satellite Radio [XM] couvre les États-Unis grâce à deux satellites GEO (*Rock* et *Roll*) depuis la fin de l'an 2000. Il propose plus d'une centaine de programmes, avec une offre pour un abonnement mensuel autour de 10 $. Sur le même marché que XM Satellite Radio, le système satellite Sirius [SIR] diffuse une soixantaine de canaux et affiche des prix équivalents. En Europe, le projet *European Satellite Digital Radio* devrait bientôt être lancé par WorldSpace. Enfin, au Japon, le satellite MBSAT conçu par Space Systems/Loral pour le groupe japonais *Mobile Broadcasting Corporation* (MBC) vient juste d'être lancé ; il permettra la diffusion audio, mais également vidéo (par la norme *Moving Picture Experts Group 4*, MPEG-4) à destination de mobiles [MBC].

1.2.2.2.4.2 Télévision et vidéo à la demande

La télévision par satellite est incontestablement le marché qui a dopé l'utilisation du satellite depuis les années 90. En 2000, la télévision représentait environ la moitié de l'utilisation de la capacité fréquentielle de tous les satellites en orbite [Alc01]. La Figure 1-2 montre l'importance de l'implantation du satellite dans une zone européenne étendue (source : Eutelsat). D'ici à quelques années, la diffusion de la télévision Haute Définition pourrait continuer à contribuer au succès du

satellite GEO. Nous discuterons des aspects normatifs liés aux systèmes de diffusion par satellite dans la partie 1.2.3.

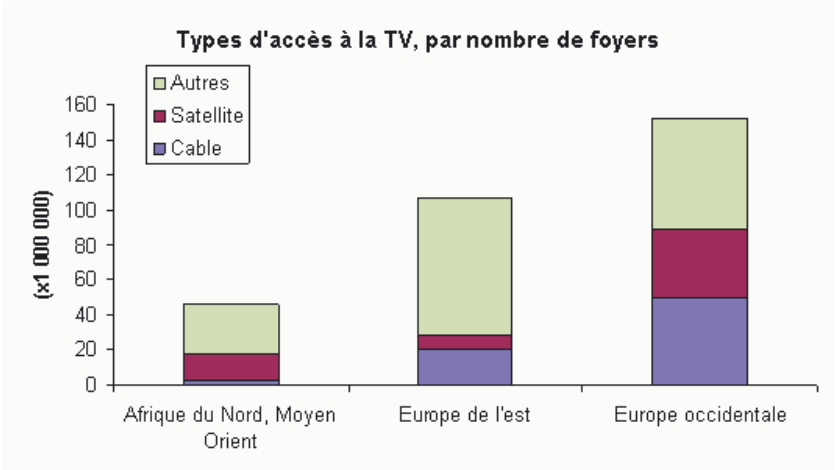

Figure 1-2 : Le marché de la diffusion de la télévision par satellite en 2002 dans une zone européenne étendue

1.2.2.2.4.3 Le Push de données par Multicast ; le projet IST GEOCAST

Avec l'apparition du satellite comme moyen d'accès à Internet, les services de diffusion de données par communication multipoints se sont diversifiés. Plusieurs opérateurs comme Eutelsat ou SES Global, proposent des offres dédiées à la diffusion multipoints de données à haut débit. La Figure 1-3 et la Figure 1-4 dressent les prévisions du marché (établies en 2000, source : projet GEOCAST) des services multipoints par satellite par genres d'application, et par cible (professionnels/particuliers). On y distingue clairement que le secteur le plus favorable à l'horizon 2010 est le *Push* de données multimédia non interactives à destination des particuliers.

Plusieurs projets de recherches portant sur IP Multicast par satellite ont été organisés, dont le projet GEOCAST (multiCAST over GEOstationnary satellite). Fondé à l'initiative de l'Information Society Technologies (IST) de l'Union Européenne, ce projet a été mené sous la tutelle d'Alcatel Space, entre plusieurs partenaires industriels et universitaires. Il a permis, entre autres, de proposer certaines adaptations de protocoles Multicast (IGMP) ou de valider des contributions concernant le niveau transport, la gestion de la QdS et la sécurité.

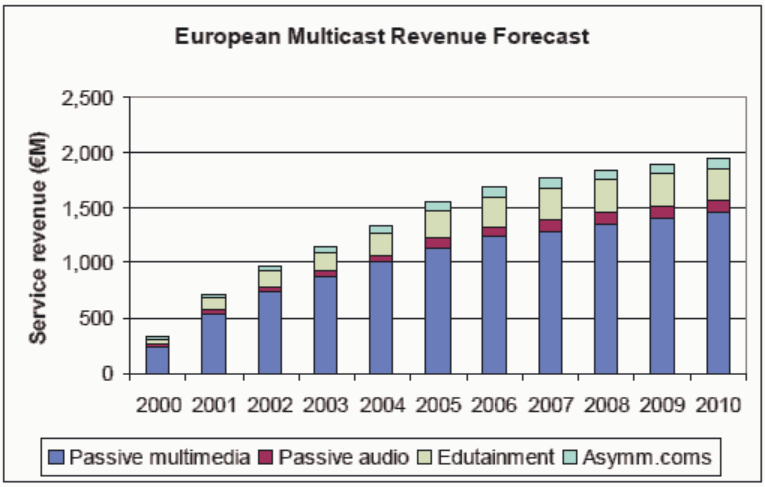

Figure 1-3 : Le marché prévisionnel pour la diffusion multipoints par satellite en Europe, par catégories d'applications

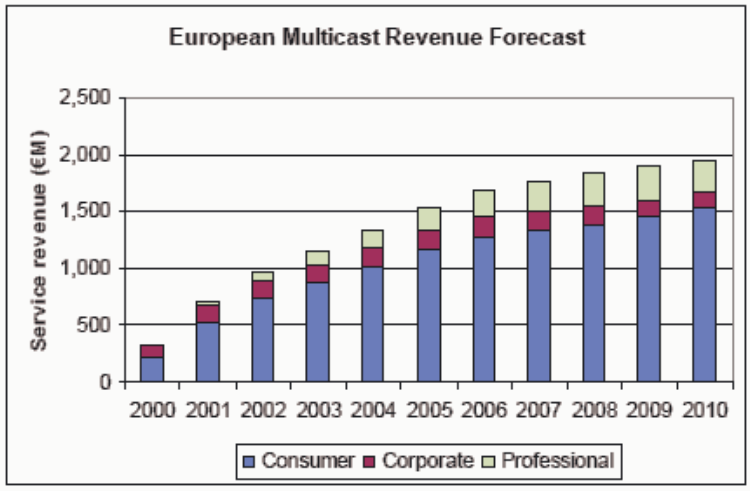

Figure 1-4 : Le marché prévisionnel pour la diffusion multipoints par satellite en Europe, par cible

1.2.3 ***Le succès de la norme de diffusion DVB-S***

1.2.3.1 Diminutions des coûts grâce au facteur d'échelle

Au début des années 90, l'ensemble des acteurs impliqué dans l'industrie de la radiodiffusion fondait le groupe DVB. Système ouvert, interopérable et orienté marché, DVB promettait au grand public la réception d'un grand nombre de chaînes de télévision avec une qualité numérique, et pour un prix raisonnable. Le satellite s'est alors rapidement imposé pour le groupe DVB face aux solutions terrestres, notamment en raison de problèmes de réglementations concernant l'attribution des fréquences pour la diffusion terrestre.

Secondé par des offres commerciales appropriées, la vente de terminaux satellite labellisés DVB a ensuite permis de diminuer les coûts pour les particuliers de manière significative (pour des terminaux résidentiels pourvus de petites antennes, de 15 000 $ en 1985 à 1 500 $ en 2002, les coûts visés pour les futurs terminaux étant de moins de 300 $). L'effet de concurrence entre les opérateurs, les fournisseurs de contenus et les constructeurs ont aussi contribué à populariser la télévision par satellite.

DVB se déclinait à l'origine en trois standards de transmission : DVB-C (*DVB - Cable*) [Ets97b], DVB-T (*DVB - Terrestrial*) [Ets01] et surtout DVB-S (*DVB - Satellite*) [Ets97a]. Des extensions à ces standards ont été introduites par la suite : DVB-RCS (*DVB - Return Channel for Satellite*) pour un support de la liaison de retour [Ets03], DVB-DNSG (*DVB - Digital Satellite News Gathering*), et récemment DVB-H (*Handheld*) pour appareils portables [Ets04a]. DVB repose sur le conteneur de données *Moving Picture Experts Group-2 Transport Stream* (MPEG-2 TS) défini par la norme ISO/IEC MPEG-2 [Iso94].

Aujourd'hui, le succès de la diffusion de la télévision par rapport aux autres types de services satellitaires est lié à plusieurs raisons :

- En premier lieu, le dimensionnement de la *gateway* est relativement simple, car les données (les bouquets, composés de flux élémentaires MPEG-2 TS) sont envoyées à débit constant. Le nombre de terminaux satellite captant le signal est sans importance sur l'ensemble du système puisqu'il n'y a pas d'interactions à gérer.
- Pour l'utilisateur, seule la phase d'installation du système de réception (antenne et terminal démodulateur/décodeur) est légèrement contraignante. L'accès au satellite est ensuite aussi pratique que le câble ou l'antenne hertzienne traditionnelle. Des

mises à jour du système et des terminaux satellite sont régulièrement faites, normalement de façon transparente.

- La souplesse apportée par la norme MPEG-2 TS a autorisé un enrichissement naturel des services par rapport à la « simple » diffusion de chaînes. Dès son lancement, des données additionnelles ont été associées aux programmes audiovisuels, comme les sous-titres en différentes langues ou le télétexte. De nouveaux types de services sont ensuite apparus : guides de programmes TV, jeux, etc… La plus récente catégorie de services intègre les communications interactives ; on peut citer entre autres les consultations de relevés de comptes bancaires, la participation à des concours, ou la commande de jetons pour des séances de vidéo à la demande. Le support de la liaison de retour satellitaire n'étant pas prévu par les systèmes DVB-S, les équipements résidentiels (ici de simples boîtiers, intégrant l'unité intérieure du terminal et la station utilisatrice) émettent grâce au réseau téléphonique commuté (RTC) de l'abonné.

1.2.3.2 IP sur DVB

1.2.3.2.1 Choix de DVB comme support d'IP sur satellite

La perspective d'accéder à Internet ou d'interconnecter des réseaux IP à haut débit grâce au satellite est devenue réaliste, techniquement et économiquement, en grande partie grâce au succès de DVB-S. DVB-S en tant que norme ouverte, propose des outils pouvant assurer le transfert de type de données dits « privés » (*i.e.* non définis par DVB ou MPEG-2). Pour ces raisons, la norme DVB-S a été considérée comme le meilleur candidat par l'ensemble des industriels du marché.

Le besoin d'une liaison retour par satellite (notamment en cas d'absence d'infrastructure terrestre) a incité l'ETSI à étendre DVB-S afin de supporter les communications bidirectionnelles par le seul réseau satellitaire. La norme DVB-RCS a ainsi été élaborée [Ets03]. Les terminaux disposant d'une capacité d'émission dans le système sont appelés *Return Channel Satellite Terminal* (RCST). Toutefois, sur la liaison retour, le prix de la ressource fréquentielle est élevé car la capacité du système doit être partagée entre tous les RCSTs émettant du trafic.

Selon DVB-RCS, la liaison retour est partagée par la méthode d'accès *Multi Frequency - Time Division Multiple Access* (MF-TDMA) entre les terminaux d'un même spot. Dans MF-TDMA, l'accès est divisé à la fois en temps et en fréquence. Cette division entraîne dans DVB-RCS une segmentation des ressources en supertrames, trames et *time slots*. Les *time slots* sont la plus petite unité d'allocation de bande possible.

L'allocation des *time slots* est faite sur demandes par les terminaux RCST au centre de contrôle du réseau satellitaire (*Network Control Center*, NCC). Les demandes sont globalement de deux types : soit la capacité est demandée pour un débit et une période donnés (méthode idéale pour disposer des ressources nécessaires à l'écoulement d'un trafic constant), soit la demande de capacité est faite ponctuellement et, le cas échéant, répétée (plus compatible avec un profil de trafic moins prévisible).

1.2.3.2.2 Le groupe IP / DVB

Le déploiement massif de la méthode *Multi Protocol Encapsulation* (MPE) pour le transport de datagrammes IP a soulevé deux problématiques, à l'origine de la création à l'IETF du groupe de travail IP sur DVB : le méthode d'encapsulation et la résolution d'adresses.

1.2.3.2.2.1 L'encapsulation

Les mises en œuvres courantes de DVB pour l'encapsulation de données privées comme IP s'appuient sur la méthode MPE, laquelle utilise le formatage des unités de données de *Digital Storage Media – Command and Control* (DSM-CC), appelées sections [Iso98]. Cependant, la méthode laisse certains détails d'implémentation à la charge de l'opérateur.

La première tâche du groupe de travail a été de proposer une normalisation décrivant complètement l'encapsulation de datagrammes IP, et en particulier celle des sections DSM-CC dans le flux MPEG-2 TS. Une première méthode, désormais abandonnée, a été baptisée *Simple Encapsulation* [Cla02]. La méthode faisait usage du champ *Adaptation Field* optionnel de MPEG-2. Ainsi, lorsqu'un paquet MPEG-2 TS transporte le début d'une section (appelée plus généralement SNDU (*SubNetwork Data Unit*)), l'adresse de son premier octet est indiquée dans le champ *Transport Private Data* de *Adaptation Field*. Avec cette méthode, il est possible d'optimiser la capacité offerte par le flux MPEG-2 lorsqu'une SNDU en cours d'encapsulation se termine alors qu'il reste de la place dans un paquet. On peut alors encapsuler le début de la SNDU suivante dans ce paquet ; cette technique est couramment appelée *Section Packing* (parfois simplement *Packing*).

Bien que le schéma d'encapsulation d'IP proposé par MPE ait été largement employé, il a été montré que cette approche n'est pas optimale [Fil04], [Fas04]. La principale raison en est que MPE n'a pas été conçu spécifiquement pour le transport d'IP. Aussi, la lourdeur de l'en-tête des sections DSM-CC (12 octets) surcharge inutilement de calculs les entités MPE émettrice et réceptrice et diminue l'efficacité de l'encapsulation. D'autre part, dans le contexte plus spécifique de la transmission IP Multicast, il aurait été désirable de pouvoir accéder à l'adresse de l'interface émettrice, pour permettre un filtrage des paquets au niveau liaison, et donc un routage plus léger au niveau IP.

Une seconde méthode d'encapsulation, *Ultra Lightweight Encapsulation* (ULE) [Fai04a] a été conçue par le groupe IP/DVB. Contrairement à la méthode précédente, ULE n'exploite plus le formatage de DSM-CC, mais est défini par un nouveau type de paquet comportant au minimum 4 octets d'en-tête. L'encapsulation des SNDUs de ULE est réalisée selon la méthode *Data Piping* prévue par DVB pour insérer directement les données dans la charge utile MPEG-2 TS sans aucun autre service. Le mécanisme indiquant l'adresse du premier octet de SNDU dans un paquet est conservé, mais l'adresse est insérée directement après l'en-tête MPEG-2 TS ; ceci permet de s'affranchir de l'utilisation du champ *Adaptation Field*, ce qui économise 3 octets par SNDU. Les paquets MPEG-2 TS incluant le commencement d'une nouvelle SNDU sont signalés lorsque leur bit *Payload Unit Start Indicator* (PUSI) est égal à 1.

Aujourd'hui, la dernière avancée du groupe concernant ULE est la mise au point du support de nouvelles fonctionnalités pour l'encapsulation [Fai04b]. Son objectif est de faciliter les traitements aux niveaux des récepteurs ou des équipements réseaux impliqués. Les fonctionnalités seront déployées grâce à des extensions d'en-têtes (en option) et pourront par exemple servir à inclure l'adresse source des datagrammes.

1.2.3.2.2.2 La résolution d'adresses

Le groupe IP / DVB s'est également donné pour objectif de normaliser les procédés de résolutions d'adresses afin d'associer les canaux logiques MPEG-2 (identifiés par le *Packet Identifier*, PID, des paquets MPEG-2) aux adresses IP, voire aux adresses de niveau 2. DVB-RCS apporte ses solutions grâce à une signalisation centralisée, par diffusion de tables (dont *IP/MAC Notification Table* et *Multicast Mapping Table*) [Fai04c]. Les défauts reprochés à cette méthode concernent sa restriction à l'encapsulation MPE, ainsi que son manque de dynamique. D'autre part, la gestion de l'adressage par le seul PID n'est pas résolue. La solution optimale semble

passer par la conception d'un protocole de résolution d'adresses IP spécifique aux réseaux à diffusion MPEG-2/DVB.

1.2.4 *Perspectives pour le satellite GEO*

La majorité des systèmes opérationnels de télécommunications par satellite GEO repose encore sur des charges utiles transparentes, c'est à dire sans traitement en bande de base à bord du satellite. La méthode est d'une grande souplesse puisque n'importe quel conteneur de données (MPEG-2 TS, ATM ou autre) peut être acheminé par le système. Pourtant, avoir recours à une charge utile régénérant le signal et à une couverture par plusieurs faisceaux (*multi-spots*) apporte des avantages comme :

- offrir un meilleur bilan de liaison sur la liaison satellite, ce qui permet de réduire les taux d'erreur binaire de la liaison ;
- appliquer des traitements par flux différenciés pour assurer une QdS à bord ;
- permettre des liaisons directes de terminal à terminal sans passer par la *gateway* (configuration du réseau satellitaire maillée). Ceci est rendu possible grâce à un accès partagé de type MF-TDMA et des traitements à bord (*On-Board Processing*, OBP) permettant le démultiplexage/remultiplexage des flux à bord. Ceci permet de :
  - diviser le délai de propagation aller-retour égal à 500 ms par deux, ce qui est important pour les communications interactives où le délai entre les stations est directement lié aux performances de la communication (p.ex. avec TCP, ou dans toutes les applications interactives) ;
  - optimiser la gestion des ressources, car les flux ne sont transmis que vers les seuls spots concernés, et alléger les charges de calcul des terminaux satellite en limitant le filtrage des paquets.

D'autre part, de nouvelles bandes de fréquences sont désormais disponibles. La bande Ka, s'étalant entre 20 à 30 GHz, est de plus en plus exploitée. Au-delà, la bande V (entre 40 et 75 GHz) pourrait aussi permettre d'étendre la bande passante disponible. Cette augmentation en fréquences permet non seulement de libérer l'engorgement spectral dans les bandes inférieures (p.ex. Ku), mais aussi de diminuer les tailles d'antennes des terminaux satellite. En contrepartie, la disponibilité statistique du service offert par la liaison est dégradée, en raison d'effets météorologiques plus importants.

1.2.5 *Spécificités du satellite pour les communications multipoints*

Les communications multipoints par satellite sont confrontées à trois obstacles majeurs. Premièrement, sur la voie aller de la communication, des détériorations du rapport signal à bruit peuvent se produire, lesquelles peuvent engendrer la corruption de données binaires. Lorsque ces détériorations deviennent trop importantes, la synchronisation du terminal satellite n'est plus possible. Dans ce cas, plus aucune donnée ne peut être délivrée par le terminal. Ces évènements sont particulièrement observés dans les nouvelles bandes de fréquences exploitées, lors de conditions météorologiques défavorables (fortes précipitations, orages) ; ils sont souvent appelés évanouissements (*fading*).

Un deuxième problème est posé par la voie retour. Certains terminaux satellite peuvent être dépourvus d'une capacité d'émission, empêchant toute garantie de livraison avec service à fiabilité totale par le seul réseau satellitaire. Ensuite, la taille des groupes exploitant la diffusion par satellite est fréquemment plus élevée que sur les réseaux terrestres car l'utilisation du satellite GEO est d'autant plus rentable que le groupe est de taille élevée. Dès lors, il faut gérer le fait que

des milliers de requêtes sur la voie de retour puissent être envoyés dans des intervalles de temps courts, par exemple à la suite de pertes de paquets. Ce phénomène est appelé implosion à la source, ou *feedback implosion*. Il peut entraîner de fortes congestions du réseau au voisinage de la source au point de bloquer le trafic si aucune mesure préventive n'est prise. Pour éviter cela, le protocole de transport doit permettre la transmission des flux multipoints sans risque de saturer le réseau, même pour de grands groupes (un million de récepteurs). On parle alors de protocoles résistants au facteur d'échelle (*scalable*).

Enfin, le délai élevé de propagation du signal peut avoir un impact important sur les communications bidirectionnelles, car il est un obstacle à la bonne réactivité des échanges.

Sur chacune des deux voies de la communication, il reste important d'être « économe » en termes de monopolisation des ressources de bande passante. Par exemple, si une fiabilité totale est exigée pour une communication, il peut être préférable, suivant la connaissance du reste du trafic, de réduire le temps d'occupation des ressources du système satellite, quitte à augmenter le débit alloué à la communication, ou au contraire de l'étaler dans le temps. Ce choix incombe à l'opérateur satellite et au fournisseur de services.

1.3 PROBLÉMATIQUE DE LA RECHERCHE

L'approche retenue tout au long de cette recherche a été pragmatique. Le but a été d'analyser au travers de l'existant (architectures protocolaires, mécanismes de fiabilité, prise en compte des contraintes de gestion de groupe, limitation d'utilisation de la voie de retour) tout ce qui permet d'offrir des services de fiabilité adaptés aux communications multipoints par satellite, en vue de proposer une architecture de transmission aussi efficace que possible.

Nous posons pour cela quelques hypothèses et le cadre de travail auxquels nous ferons référence tout au long de ce mémoire. Ensuite, nous abordons les problèmes qui sont à l'origine de ce travail et nous terminons ce chapitre en donnant l'organisation du reste du mémoire.

1.3.1 *Hypothèses*

- Concernant le satellite, nous ne ferons pas d'hypothèse sur la nature de sa charge utile. Toutefois, comme nous l'avons évoqué, les satellites multi-spots et avec OBP ayant la possibilité de commuter les données à bord sont préférables.
- Les terminaux satellite (dont la *gateway)* sont supposés être les équipements dans lesquels l'encapsulation et le réassemblage des datagrammes IP dans le flux MPEG-2/DVB-S est réalisée.
- On supposera qu'il n'y a pas de reprise d'erreur autre qu'au niveau transport.
- On supposera parfois la communication multipoints sans voie de retour, mais celle-ci sera considérée comme sans perte dans les autres cas. Nous sommes conscients que cette hypothèse est discutable (p.ex. si la voie de retour est basée sur la liaison retour de DVB-RCS) mais elle constitue un problème à part entière que nous ne traiterons pas ici.
- La communication repose sur IP Multicast. Nous donnerons parfois des exemples plutôt valables dans IPv4, mais l'extension des mécanismes à la version 6 se fera naturellement.
- Les applications n'exploiteront que des communications point à multipoints.

1.3.2 *Architecture du réseau*

L'architecture du réseau, support de la communication multipoints, est illustrée dans la Figure 1-5. En plus d'éléments déjà évoqués, il y apparaît le serveur d'accès large bande (*Broadband Access Server*, BAS), par lequel passent toutes les communications en charge de l'opérateur satellite. Il faut préciser ici que les réseaux d'accès connectant les stations à leur terminal satellite peuvent se limiter à une simple liaison (p.ex. de type Ethernet).

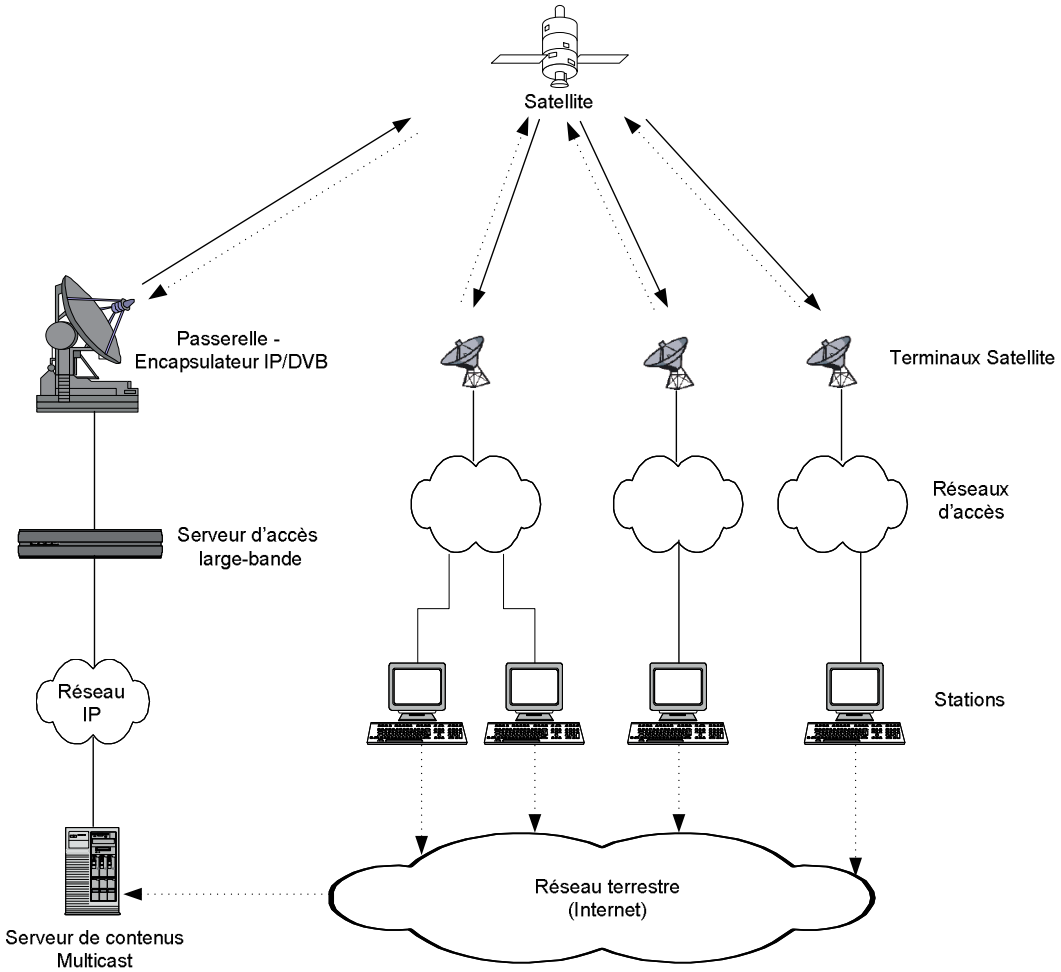

Figure 1-5 : Architecture de réseau pour des communications multipoints par satellite

1.3.3 *Problématique liée à la distribution de la fiabilité*

Comme nous le montrerons dans le chapitre 2, l'intégration de codage correcteur d'erreur depuis quelques années dans les protocoles de transport multipoints a modifié l'approche traditionnelle de gestion de la fiabilité des communications point à point. Dans le contexte satellitaire, nous pensons qu'utiliser le codage à ce niveau remet davantage en cause la gestion de la fiabilité que dans les réseaux terrestres. Ceci provient essentiellement du poids du codage canal déployé sur la couche physique des liaisons par satellite.

Dans la situation actuelle, le codage canal ne peut pas être optimisé pour la communication de groupe. Deux méthodes sont possibles : soit le taux est choisi parmi un petit sous-ensemble (5 ou 7 valeurs avec DVB-S et DVB-RCS), soit le choix est complètement statique et on ne peut qu'accepter les limites des performances du système. Malheureusement, la complexité relative à la première méthode est incontournable. En effet, il faut être capable successivement d'analyser la qualité du signal reçu, de décider à quel moment précis basculer d'un taux de codage à l'autre, et

d'en informer les terminaux satellite. Bien qu'une partie de ces problèmes soit traitée dans les normes, il n'en reste pas moins que de nombreux systèmes commerciaux préfèrent utiliser un taux de codage fixe. Le cas des communications multipoints est encore plus complexe car il faudrait prendre en compte, régulièrement, les valeurs de mesures de la qualité du signal radio pour l'ensemble des terminaux satellite connectant le groupe de stations. La lourdeur de la signalisation engendrée pour transmettre ces informations constitue un frein évident pour cette solution.

Le codage FEC, désormais réalisable au niveau des couches hautes, suggère que les réseaux traversés constituent des canaux de transmission au sens de la théorie de l'information, puisque des « erreurs de transmissions » s'y produisent. Il en est de même pour les liaisons satellite de terminal à terminal. Pour illustrer cette idée, on peut étendre la représentation habituelle du diagramme de Shannon à notre contexte pour la transmission sur la voie aller (cf. Figure 1-6). Sur la figure, il a été supposé que deux récepteurs appartiennent au groupe et qu'ils se connectent *via* deux terminaux satellite différents.

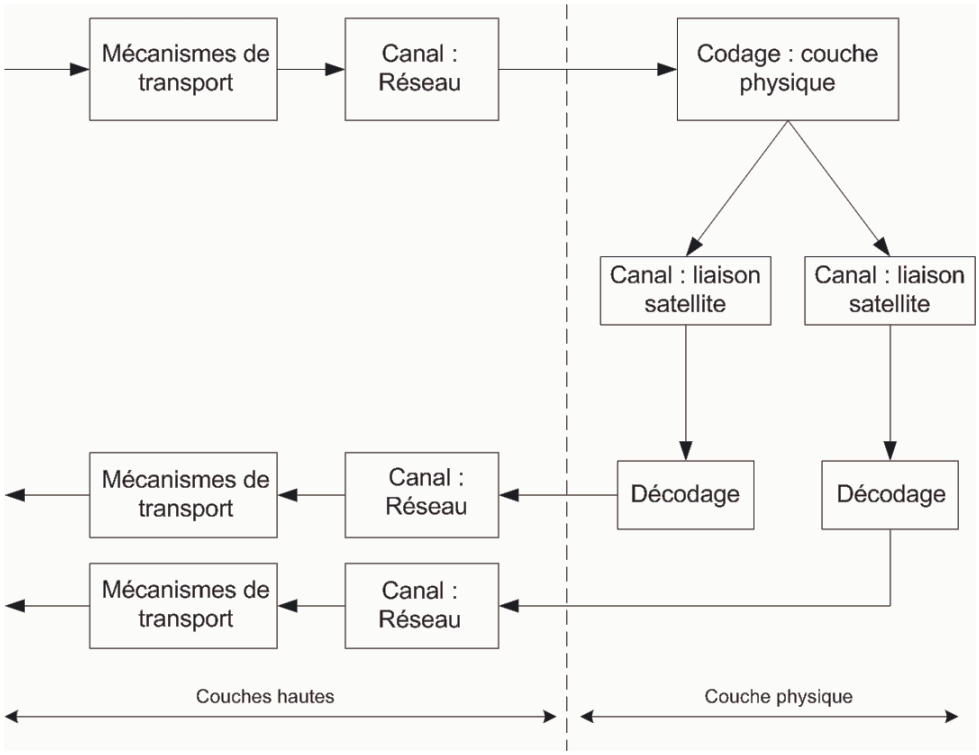

Figure 1-6 : Extension de la représentation de Shannon dans le contexte d'étude avec un groupe constitué de deux récepteurs

Dans ce mémoire, nous pensons qu'il est important d'analyser de manière globale les mécanismes de fiabilité, pour permettre d'optimiser la transmission de bout en bout, ayant pour contrainte la meilleure utilisation possible de la ressource radio. Parmi nos objectifs, nous définirons les fonctions à mettre en œuvre dans les boîtes *Mécanismes de transport* de la figure précédente. Nous verrons aussi comment une partie des autres fonctions existantes pourra être améliorée. Pour cela, nous commencerons par décrire dans le chapitre suivant l'organisation de la gestion de la fiabilité dans le contexte étudié.

BIBLIOGRAPHIE

[Alc01] Revue des télécommunications d'Alcatel 2001.

[All00] IETF RFC, "Ongoing TCP Research Related to Satellites", M. Allman *et al.*, RFC 2760, février 2000.

[Alt99] "Les réseaux satellitaires de télécommunications, technologies et services", E. Altman, A. Ferreira, J. Galtier, éditions Dunod, 1999.

[Atm01] ATM Security Specification, v1.1, af-sec-0100.002, mars 2001.

[Bal97] IETF RFC, "Core Based-Tree", A. Ballardie, RFC 2189, septembre 1997.

[Bra97] IETF RFC, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", R. Braden et *al.*, RFC 2205, septembre 1997.

[Cai02] IETF RFC, "Internet Group Management Protocol, Version 3", B. Cain *et al.*, , RFC 3376, octobre 2002.

[Che00] "A QoS-Aware Multicast Routing Protocol", S. Chen, K. Nahrstedt, Y. Shavitt, IEEE INFOCOM, mai 2000.

[Cla02] IETF Internet Draft, "Simple Encapsulation for transmission of IP datagrams over MPEG-2/DVB networks", H. D. Clausen et *al.*, draft-unisal-ipdvb-enc-00.txt, avril 2002.

[Cui02] "Scalable QoS Multicast Provisioning in DiffServ-Supported MPLS Networks", J.H. Cui *et al.*, proceedings of IEEE Globecom2002, Taiwan, novembre 2002.

[DAR] http://developer.apple.com/darwin/projects/streaming/

[Dee89] IETF RFC, "Host Extensions for IP Multicast", S. Deering, RFC 1112, 1989.

[Dee99] IETF Internet Draft, "Protocol Independent Multicast Version 2 Dense Mode Specification", S. Deering *et al.*, draft-ietf-pim-v2-dm-03.txt, juin 1999.

[DF] http://www.digitalfountain.com/licensing/products/DFcore.cfm

[Doc04] DOCSIS Specification, "Baseline Privacy Plus Interface Specification", SP-BPI+-I11-040407, avril 2004.

[Dur01] IETF RFC, "A Link-Layer Tunneling Mechanism for Unidirectional Links", E. Duros *et al.*, RFC 3077, mars 2001.

[Ets01] ETSI EN 300 744, "Digital Video Broadcasting (DVB) ; Framing structure, channel coding and modulation for digital terrestrial television, v1.4.1, janvier 2001.

[Ets03] ETSI EN 301 790, "Digital Video Broadcasting (DVB) ; Interaction channel for satellite distribution systems", v1.3.1, mars 2003.

[Ets04] ETSI EN 302 304 "Digital Video Broadcasting (DVB) ; Transmission System for Handheld Terminals (DVB-H)", v1.1.1, juin 2004.

[Ets97a] ETSI EN 300 421, "Digital Video Broadcasting (DVB) ; Framing structure, channel coding and modulation for 11/12 GHz satellite services", v1.1.2, août 1997.

[Ets97b] ETSI EN 300 429, "Digital Video Broadcasting (DVB) ; Framing structure, channel coding and modulation for cable systems", v1.1.2, août 1997.

[Ets99] ETSI EN 301 210, "Digital Video Broadcasting (DVB) ; Framing structure, channel coding and modulation for Digital Satellite News Gathering (DSNG) and other contribution applications by satellite", v1.1.1, mars 1999.

[Fai04a] IETF Internet Draft, "Ultra Lightweight Encapsulation (ULE) for transmission of IP datagrams over MPEG-2/DVB networks", draft-ietf-ipdvb-ule-01.txt, G. Fairhurst, G. Collini-Nocker, mai 2004.

[Fai04b] IETF Internet Draft, "Ultra Lightweight Encapsulation (ULE) Extension Header", G. Fairhurst *et al.*, draft-collini-ipdvb-xule-00.txt, mai 2004.

[Fai04c] IETF Internet Draft, "Address Resolution for IP datagrams over MPEG-2 networks", G. Fairhurst, M.J. Montpetit, draft-fair-ipdvb-ar-01.txt, mai 2004.

[Fal98] "QoSMIC: a QoS Multicast Internet Protocol", M. Faloutsos, A. Banerjea, and R. Pankaj, ACM SIGCOMM, Sep 2-4, Vancouver BC, 1998.

[Fas04] "IP Multicast Architecture over Next-Generation GEO satellites", J. Fasson, E. Chaput, C. Fraboul, VTC Spring, 2004.

[Fei00] "Receiver-Initiated Multicasting with Multiple QoS Constraints", A. Fei, M. Gerla, IEEE INFOCOM, mai 2000.

[Fen04] IETF Internet Draft, "Protocol Independent Multicast - Sparse Mode PIM-SM: Protocol Specification (Revised)", B. Fenner *et al.*, draft-ietf-pim-sm-v2-new-09.txt, février 2004.

[Fen97] IETF RFC, "Internet Group Management Protocol, Version 2", W. Fenner, RFC 2236, novembre 1997.

[Fil04] "Efficient Support of IP Multicast in the Next-Generation of GEO Satellites", F. Filali, G. Aniba, W. Dabbous, IEEE Journal of Selected Areas in Communications, février 2004.

[Han00] IETF RFC, “Session Announcement Protocol”, M. Handley, C. Perkins, E. Whelan, RFC 2974, octobre 2000.
[Han98] IETF RFC, “SDP: Session Description Protocol”, M. Handley, V. Jacobson, RFC 2327, avril 1998.
[Han99] IETF RFC, “SIP: Session Initiation Protocol”, M. Handley *et al.*, RFC 2543, mars 1999.
[Hed88] IETF RFC, “Routing Information Protocol”, C. Hedrick, RFC 1058, juin 1988.
[HEL] http://www.helixcommunity.org
[IPD] http://www.ietf.org/html.charters/ipdvb-charter.html
[IPS] http://www.ietf.org/html.charters/ipsec-charter.html
[Iso91] Norme ISO/CEI 9126 :1991, “Technologies de l'information - Évaluation des produits logiciels - Caractéristiques de qualité et directives d'utilisation”, Organisation Internationale de Normalisation, Genève, 1991.
[Iso94] ISO/IEC 13818-1: “Information Technology – Generic Coding of Moving Pictures and Associated Audio Information - Part 1 : Systems”, juin 1994.
[Iso98] ISO/IEC 13818-6, “Information Technology – Generic Coding of Moving Pictures and Associated Audio Information - Part 6 : Extensions for DSM-CC”, 1998.
[Lub04] IETF RFC, “Wave and Equation Based Rate Control (WEBRC) Building Block”, M. Luby, V. Goyal, RFC 3738, avril 2004.
[MAD] http://atm.tut.fi/mad/
[Mar02] “Satellite Communications Systems, Systems, Techniques and Technology”, G. Maral, M. Bousquet, 4ème édition, ed. Wiley, 2002.
[MBC] http://www.mbco.co.jp/english/
[MBO] http://www-itg.lbl.gov/mbone/
[MBS] http://www-mice.cs.ucl.ac.uk/multimedia/software/
[MCL] http://www.inrialpes.fr/planete/people/roca/mcl/mcl.html
[Moy94] IETF RFC, “Multicast Extensions to OSPF”, J. Moy, RFC 1584, mars 1994.
[MSE] http://www.ietf.org/html.charters/msec-charter.html
[Non98] “The impact of Routing on Multicast Error Recovery”, J. Nonnenmacher and E. W. Biersack, Computer Communication Review, vol.21, no.10, p. 867-79, juillet 1998.
[Oom01] IETF Internet Draft, “MPLS multicast traffic engineering”, D. Ooms *et al.*, draft-ooms-mpls-multicast-te-00.txt, 2001.
[QT] http://www.apple.com/quicktime/products/qt/
[QTS] http://www.apple.com/quicktime/products/qtss/
[REA] http://www.real.com/player/
[RMT] IETF Reliable Multicast Transport Working Group, http://www.ietf.org/html.charters/rmt-charter.html
[Roj00] “Architecture de transport à ordre et fiabilité partiels pour les applications multimédias adaptatives à temps contraint”, L. Rojas Cardenas, thèse de doctorat de l’INP Toulouse, 2000.
[SIR] http://www.siriusradio.com
[TIB] http://www.tibco.com/software/enterprise_backbone/smartpgm.jsp
[Wai98] IETF RFC, “Distance Vector Multicast Routing Protocol”, D. Waitzmann, C. Partridge, S. Deering, RFC 1075, novembre 1998
[WMP] http://www.microsoft.com/windows/windowsmedia/9series/player.aspx
[WMS] http://www.microsoft.com/windows/windowsmedia/9series/server.aspx
[WSP] http://www.worldspace.com
[XM] http://www.xmradio.com

Chapitre 2

La fiabilité dans les réseaux et les communications par satellite

INTRODUCTION

Dans un réseau comme Internet, on estime souvent que seule la couche de niveau transport est responsable de la fiabilisation des transmissions. Aujourd'hui, dans les réseaux terrestres classiques, il est vrai que l'essentiel de la gestion de la fiabilité est faite par le protocole de transport, notamment par la stratégie de retransmissions. En réalité, les actions mises en place par cette couche sont étroitement liées à la présence des couches sous-jacentes dans le processus global du transport de l'information. Par exemple, les unités de données arrivant jusqu'à la couche transport du récepteur sont toujours supposées parfaites, c'est-à-dire avoir été transmises sans erreur.

Depuis que de nouvelles technologies de liaisons se popularisent, de nouveaux services en relation avec la fiabilité leur sont demandés. Ces services peuvent directement agir sur le plan utilisateur (par exemple en fournissant au flux un certain niveau de fiabilité), ou sur le plan de contrôle (par exemple en mesurant des paramètres de QdS comme le taux d'erreur paquet). Dans le cas des transmissions par satellite GEO dans les bandes de fréquences élevées (bande Ka et V), la fiabilité binaire des transmissions numériques « brutes » est connue pour être faible (*Bit Error Rate* (BER) compris entre 10^{-6} et 10^{-2} dans les cas extrêmes). Le codage correcteur d'erreurs, traditionnellement implémenté dans ces types de liens, améliore heureusement la situation, au point d'obtenir une fiabilité nettement meilleure (BER entre 10^{-10} et 10^{-5}). Les points faibles du codage correcteur d'erreurs, sont, dans l'ordre d'importance croissante : une augmentation de la complexité du système de transmission, une dégradation inévitable de la fiabilité en cas de conditions météorologiques trop défavorables, et surtout une réduction du débit utile.

D'autre part, l'utilisation systématique de codage correcteur d'erreurs pour les communications de groupe au niveau de la couche transport peut apparaître surprenant. Rendue possible sur les ordinateurs personnels grâce aux progrès informatiques, cette nouvelle forme de FEC, à l'autre extrémité du codage canal habituel dans la pile protocolaire, fait s'interroger sur la répartition des mécanismes de fiabilité au sein de l'ensemble de la pile.

Afin de répondre à cette question dans la suite de ce mémoire, nous consacrons ce chapitre à la présentation des mécanismes liés à la fiabilité dans les réseaux en général, et dans les communications multipoints par satellite en particulier. Ces deux aspects seront traités en parallèle dans les trois parties qui constituent ce chapitre. Au travers de cette présentation, notre but est de dégager un ensemble de mécanismes potentiels adaptés au cadre de référence présenté

à la fin du chapitre 1. Les arguments seront ici essentiellement d'ordre qualitatifs, mais une analyse numérique détaillée de mécanismes importants sera faite dans le chapitre 3.

La suite de ce chapitre est organisée de la façon suivante. Nous commencerons par examiner dans les réseaux terrestres les raisons des défaillances de transmission, leur impact sur les communications, et les moyens de les limiter. Nous montrerons aussi la spécificité de la couche physique du satellite concernant ses performances en termes de fiabilité, et nous présenterons les techniques de codage utilisées dans les systèmes actuels. La partie suivante permettra d'introduire l'ensemble des mécanismes et techniques liés à la fiabilisation dans les réseaux. Comme exemple d'application de ces techniques, nous verrons comment est organisée la gestion de la fiabilité, de la couche physique à la couche réseau, dans les réseaux DVB-S et DVB-RCS pour le transport d'un flux IP par les méthode d'encapsulation MPE et ULE. Finalement, dans la dernière partie de ce chapitre, nous détaillerons les mécanismes liés à la fiabilité dans le cadre du transport multipoints. Nous verrons quelles stratégies ont été adoptées pour les réseaux terrestres, comment il a été possible de les instancier en trois classes, avant de discuter de leur adéquation au contexte satellitaire.

2.1 ORIGINE ET CARACTÉRISATION DES DÉFAILLANCES

2.1.1 *Notions générales*

2.1.1.1 Rejets et pertes de paquets dans les réseaux classiques

2.1.1.1.1 Origines

L'accès au medium dans certains réseaux à commutations de paquets est une première raison expliquant les pertes de paquets. Par exemple, au sein de réseaux locaux, des méthodes d'accès aléatoires peuvent être employées, comme CSMA/CD (*Carrier Sense Multiple Access /Collision Detection*) sur Ethernet [Iee02]. Quand des paquets issus de différentes stations sont émis à des instants suffisamment proches sur le medium partagé, ils rentrent en collision et sont perdus. Dans ces réseaux, la couche d'accès est responsable de détecter les collisions et de les résoudre en réémettant l'unité de données sur le medium.

La cause suivante de rejets des paquets est de loin la plus fréquente. Dans les grands réseaux, les équipements d'interconnexion comme les commutateurs et les routeurs traitent chaque jour davantage de trafic. Bien qu'extrêmement rapides, il n'en reste pas moins que leurs ressources de calcul et leurs mémoires sont limitées. Quand la charge de trafic devient importante, comme cela arrive souvent pour les routeurs de cœur de réseau, les paquets ne peuvent plus être traités aussi vite qu'ils arrivent. En attendant d'être aiguillés vers la bonne destination, ils sont mémorisés sur les files d'attente des interfaces d'entrées du routeur. De même, sur les interfaces de sortie, il arrive qu'il y ait une demande trop importante par rapport à la capacité de la liaison ; une nouvelle fois les paquets sont stockés dans les files d'attente. À mesure que la charge augmente, le remplissage des files devient de plus en plus important, jusqu'à saturation. Si de nouveaux paquets arrivent sur une file saturée, ils ne peuvent plus être stockés. Il sont alors rejetés.

Ce problème de congestion des files d'attente peut également se produire au niveau des stations d'extrémité. Si la vitesse du processus d'émission est supérieure à celle du processus de réception (le processus récepteur ne vient pas suffisamment rapidement lire les données qui arrivent du réseau), une saturation peut également se produire. Bien que les données puissent arriver « physiquement » à destination, les files d'attente de niveau liaison ou réseau peuvent être saturées. Les paquets sont ainsi supprimés avant d'atteindre le niveau applicatif.

Outre ces deux raisons, les paquets peuvent aussi être rejetés à cause d'erreurs de transmission (cf. 2.1.1.2) et, désormais rarement, à cause d'erreurs de traitement au niveau des routeurs ou des stations impliquées dans la communication.

2.1.1.1.2 Conséquences sur les applications

La perte des paquets peut avoir différents effets sur le fonctionnement de l'application, selon sa nature. Dans le cas où elle ne spécifie pas de contraintes temporelles au service de transport (messagerie électronique par exemple), la retransmission des données et les délais associés ne sont pas trop pénalisants. En revanche, pour beaucoup d'autres types de services (dont les transferts de données fiabilisés ou la diffusion de media à temps contraint), les pertes de paquets dégradent plus nettement le service rendu. Cela peut se traduire, selon les cas, par une augmentation du délai de service (due aux retransmissions), ou par une diminution de la qualité du media diffusé. Dans un contexte de communications interactives de groupe, cela diminue la fréquence effective de rafraîchissement des données. Dès lors, des mesures préventives ou réactives sont à envisager pour gérer les pertes de paquets quand elles deviennent trop gênantes.

2.1.1.1.3 Contrôle et limitation des pertes de paquets

Dans un contexte Internet « classique », la situation de pertes de paquets d'un flux est si fréquente que des mécanismes intégrés dans les protocoles de transport s'en servent constamment. Par exemple, dans TCP, la mesure de ces pertes sert à déterminer l'équilibre entre le débit d'émission et la fréquence des congestions sur l'ensemble des routeurs traversés par le flux, pour obtenir le meilleur débit utile à l'application et acceptable pour le réseau. Outre cette solution, assez simple à mettre en œuvre car elle ne concerne que les stations d'extrémité, des mécanismes relatifs aux équipements de réseaux peuvent aussi être envisagés. Un autre moyen de limiter les pertes est de développer l'infrastructure de réseau, en déployant plus de routeurs et/ou des liaisons plus importantes (en nombre et en capacité). Cette solution étant coûteuse pour l'opérateur, elle n'est pas considérée comme efficace. Un dernier moyen est d'avoir recours à des mécanismes de différentiation de service dans un domaine déterminé et permettant la gestion de la QdS. Grâce à eux, il est possible de limiter le rejet de paquets de certains flux soumis à des politiques de QdS. Cette solution est toutefois difficile à mettre en place, principalement en raison du caractère local (*i.e.* intra-domaine) de la politique de QdS à appliquer sur les flux.

Malgré l'intérêt indéniable des mécanismes précédents, les paquets provenant de communications multipoints sont parfois perdus. Il faut alors avoir recours à des mécanismes de fiabilisation que nous appellerons directs : FEC et ARQ. Nous les appelons ainsi car ils agissent directement et seulement sur la communication à fiabiliser. Nous décrirons précisément leur mise en application dans la partie 2.3.2.

2.1.1.2 Erreurs de transmission

Tout support (ou canal) de communication est imparfait, c'est une nécessité physique. On représente souvent cette imperfection par ce que l'on appelle « bruit », qui est le terme générique pour désigner l'ensemble des phénomènes physiques qui perturbent la bonne réception du signal d'information. L'agitation thermique, liée aux mouvements aléatoires des électrons de conduction du milieu de propagation, est toujours l'une des causes de bruit. D'autres bruits existent : le bruit de grenaille, le bruit de scintillation, etc. L'ensemble de ces bruits affecte non seulement les canaux de transmission, mais également les équipements électroniques assurant la transmission et la réception du signal.

Le bruit n'est toutefois pas la seule origine de déformation des signaux. À titre d'exemple, on peut citer la limitation de la bande de fréquence dans le canal ou dans les dispositifs électroniques

(amplificateurs, filtres, etc…), ou encore la variation du temps de groupe dépendante du milieu de propagation. Les composantes spectrales élevées des signaux sont généralement celles qui sont déformées par ces phénomènes.

Dans les réseaux terrestres câblés modernes, les erreurs de transmissions sont normalement des évènements rares. En effet, soit les stations ou routeurs sont très proches les uns des autres (moins d'une centaine de mètres), soit les liaisons longue distance sont assurées par fibre optique fiables (BER inférieur à 10^{-10}). On peut toutefois noter que sur les liaisons désormais répandues de type ADSL sur paire torsadée, l'atténuation du signal (environ 5 dB par centaine de mètres) limite la distance entre l'abonné et son répartiteur local.

Dans les cas où des paquets arrivent erronés au niveau des stations ou des routeurs, ils sont rejetés par l'un des mécanismes de la couche liaison (p.ex. dans Ethernet, dans ATM ou dans sa couche d'adaptation pour IP, *Adaptation ATM Layer-5* (AAL-5)). Nous verrons plus précisément comment en 2.2.1.1.

2.1.2 ***Spécificités dans le contexte satellitaire***

2.1.2.1 Remise en cause des hypothèses des réseaux classiques

Les réseaux par satellite ont plusieurs caractéristiques différentes des réseaux terrestres classiques, ce qui pourrait impliquer des différences importantes dans la gestion des transmissions. Pourtant, mis à part l'adaptation de la couche physique pour améliorer la fiabilité binaire de la transmission numérique, l'architecture globale permettant le transport de données dans les réseaux par satellite est proche de celle des réseaux terrestres. Nous réévaluons maintenant l'importance des différentes origines des défaillances, dans le contexte du satellite GEO.

2.1.2.1.1 Pertes de paquets

Outre les congestions pouvant se produire en amont ou en aval du lien satellitaire, l'accès compétitif entre plusieurs terminaux satellite à la ressource radio sur le canal retour est parfois responsable de collisions, impliquant des pertes d'unités de données. Toutefois, des mécanismes de contrôle d'admission (*Connection Admission Control*, CAC) orientés connexion ou encore des mécanismes d'allocation dynamique à la demande (*Dynamic Assignment for Multiple Access*, DAMA) sont couramment déployés, afin de limiter voire d'empêcher de telles collisions.

2.1.2.1.2 Corruptions de données

L'hypothèse du faible taux d'erreur binaire en environnement satellitaire peut être remise en cause, pour les raisons évoquées dans les paragraphes suivants. Pour permettre le fonctionnement correct des applications, il a fallu amener le niveau de fiabilité du lien satellitaire à celui des liens dans les réseaux terrestres. Nous verrons comment cela est possible grâce aux codages correcteurs d'erreurs anticipatifs.

Dans les liaisons sans-fil, les puissances d'émissions sont limitées en raison de problèmes d'interférences, de réglementations, ou simplement de capacité. D'autre part, la propagation du signal ne se fait pas sur support guidé, et les puissances des signaux s'atténuent rapidement (l'atténuation de puissance est proportionnelle au carré de la distance parcourue par le signal).

Dès lors, les puissances des signaux reçus sont faibles, ce qui rend difficile la tâche des démodulateurs numériques chargés de décider, en fonction du signal reçu bruité, quels symboles ont été émis. Dans le domaine des liens satellite géostationnaire entre points fixes, d'autres facteurs limitatifs sont identifiés. D'abord, il faut rappeler que la puissance du signal reçu est

atténuée de façon importante par la longueur du trajet parcouru (200 dB), rendant le signal sensible au bruit. Ensuite, la propagation des ondes dans les couches de l'atmosphère est pénalisée par des phénomènes de réflexion, de diffraction, et d'absorption. En particulier, la présence des molécules d'eau contribue de façon importante à l'absorption. Par exemple, dans la bande de transmission Ka, des dégradations supplémentaires d'une trentaine de décibels peuvent intervenir sur le bilan de liaison lors de conditions météorologiques défavorables (pluies abondantes ou orages). La fréquence de ce type d'évènements est dépendante de la zone géographique considérée.

D'autres facteurs dégradent les performances des liaisons par satellite, principalement les pertes liées aux dépointages des antennes, les pertes de démodulation, les pertes par dépolarisation, ou les pertes dues aux scintillations. Ces dernières correspondent à des fluctuations rapides de la puissance de la porteuse ; elles sont observées quelles que soient les conditions météorologiques. On les estime généralement entre 0 et 1 décibel [Geo01].

2.1.2.2 Mesure de la fiabilité binaire des liaisons par satellite

2.1.2.2.1 Performance individuelle par lien

Le bilan de la liaison satellite permet de quantifier la fiabilité d'une transmission numérique par son taux d'erreur binaire (*Bit Error Rate*, BER). Pour cela, on procède habituellement au calcul du rapport entre la puissance du signal reçu de la porteuse, C, exprimée en W, et la densité spectrale de puissance du bruit reçu, N_0, exprimée en W/Hz, sur l'une des liaisons, montante ou descendante. Le calcul complet fait intervenir plusieurs paramètres du système, de la fréquence porteuse du signal jusqu'au type de modulation et au débit symboles utilisés. D'une part, la valeur offerte par la liaison est donnée en fonction du dimensionnement des termes de droite dans l'équation suivante :

$$\frac{C}{N_0} = PIRE \cdot \frac{1}{L_{propag}} \cdot \frac{1}{L_{add}} \cdot \frac{G_r}{T} \cdot \frac{1}{k_B} \qquad (2.1)$$

Où :

- PIRE est la Puissance Isotrope Rayonnée Équivalente, égale au produit entre le gain de transmission de l'antenne et la puissance du signal amplifié qu'elle émet ;
- L_{propag} est l'atténuation due à la propagation naturelle, égale à $(4 \pi R c / f_0)^2$ où f_0 est la fréquence de l'onde porteuse, R la distance entre la Terre et le satellite, et c la vitesse de la lumière ;
- L_{add} représente d'autres types de pertes (absorption dans l'atmosphère, pertes de démodulation, pertes par dépolarisation éventuelle, etc…) ;
- G_r/T est le facteur de qualité (*figure of merit*) de l'antenne de réception ;
- k_B est la constante thermodynamique de Boltzmann ($1.379\ 10^{-23}$ J.K^{-1}).

D'autre part, on peut écrire que la valeur requise pour le rapport C/N_0 est :

$$\frac{C}{N_0} = \frac{E_s}{N_0} \cdot R_s = \frac{E_b}{N_0} \cdot R_b \qquad (2.2)$$

Avec :

- E_s/N_0 (resp. E_b/N_0) la valeur requise du rapport entre l'énergie par symbole (resp. par bit) transmis dans le canal et la densité spectrale de bruit ;
- N_0 la densité spectrale du bruit ;
- R_s (resp. R_b) le débit symbole (resp. binaire) de la porteuse.

La mesure de la fiabilité intervient dans l'équation (2.2) par l'intermédiaire de la modulation numérique choisie. En effet, la modulation donne la loi BER=$f(E_b/N_0)$, ce qui permet de relier le taux d'erreur binaire de la transmission à l'ensemble de tous les autres paramètres. Par exemple, pour la modulation *Quadrature Phase Shift Keying* (QPSK), on a :

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{N_0}}\right) \tag{2.3}$$

où *erfc(.)* est la fonction d'erreur complémentaire.

2.1.2.2.2 Performance globale sur satellite transparent

Dans le cas d'un satellite transparent, il faut d'abord utiliser l'équation (2.1) pour calculer les rapports $(C/N_0)_{up}$ et $(C/N_0)_{down}$ des liaisons montante et descendante. Ensuite, on peut montrer que le rapport $(C/N_0)_T$ équivalent sur l'ensemble (lien montant et lien descendant) s'obtient par l'équation (2.4) :

$$\left(\frac{C}{N_0}\right)_T^{-1} = \left(\frac{C}{N_0}\right)_{up}^{-1} + \left(\frac{C}{N_0}\right)_{down}^{-1} \tag{2.4}$$

On peut alors procéder au calcul du rapport E_s/N_0, puis du BER de la liaison globale, par les équations (2.2) et (2.3). Il faut préciser ici que l'équation (2.4) ne prend en compte ni les bruits dus à l'interférence avec d'autres porteuses, ni les bruits d'intermodulation éventuels.

2.1.2.2.3 Performance globale sur satellite régénérateur

Sur une liaison par satellite régénérateur, le signal est démodulé puis remodulé à bord. Les deux liaisons sont alors considérées comme indépendantes. Pour obtenir le taux d'erreur binaire BER_T sur l'ensemble de la liaison, il faut calculer les taux d'erreurs binaire individuels des liaisons montantes BER_{up} et BER_{down} par les équations (2.1) et (2.2). Ensuite, on peut calculer le taux d'erreur global BER_T grâce à :

$$BER_T = BER_{up}(1 - BER_{down}) + BER_{down}(1 - BER_{up}) \tag{2.5}$$

Cette équation prend une forme simplifiée quand BER_{up} et BER_{down} sont petits devant 1, et l'on peut alors faire l'approximation :

$$BER_T = BER_{up} + BER_{down} \tag{2.6}$$

2.1.2.3 Amélioration des performances de fiabilité de la liaison

2.1.2.3.1 Correction d'erreur par anticipation (FEC)

Généralement, le taux d'erreur binaire moyen obtenu d'après tous les paramètres est jugé trop faible et trop variable. On peut rendre le canal plus fiable, en mettant un codage correcteur d'erreurs en œuvre avant la modulation numérique. En contrepartie, le codage induit un surcoût

en terme de bande passante, car le débit d'information utile que le canal peut transmettre est diminué. Pour chaque code, on définit son taux (appelé aussi efficacité ou rendement), ρ, comme étant le rapport entre le nombre de symboles donnés en entrée du codage (*k*) et le nombre de symboles fournis par le codage (*n*, supérieur à *k*).

2.1.2.3.1.1 Les types de codes

On distingue deux grandes familles de codes : les codes convolutionnels et les codes en blocs.

- Codes convolutionnels

Cette première famille de codes est utilisée massivement, en particulier avec l'avènement des décodages à décision souples (dont une version de l'algorithme de Viterbi). L'encodage est réalisé simplement et nécessite peu d'opérations ; seules des additions binaires (XOR) ou des registres à décalage sont nécessaires. Le codage est dit convolutionnel car l'état du codeur dépend de ses *K* états précédents ; le paramètre *K* est *la longueur de contrainte*. À chaque cycle d'horloge, les mémoires des registres sont mises à jour en fonction des états précédents du codeur et des nouveaux bits en entrée. Les bits de parité sont obtenus par l'addition de certains bits en sorties des registres et éventuellement des bits en entrée.

Pour augmenter le taux de code original du codeur (1/2 ou 1/3 généralement), il est ensuite possible de poinçonner les bits de parité obtenus, c'est-à-dire de ne transmettre qu'une partie de la parité. Cette opération permet d'optimiser le compromis entre robustesse du code et efficacité, en fonction des objectifs à atteindre par le système de transmission.

Les opérations concernant le décodage sont plus complexes à réaliser. Elles ont un impact important sur les performances. Il est intéressant d'avoir recours aux algorithmes de décodage exploitant les valeurs de confiance des symboles alimentant le décodeur (on parle d'entrées souples ou *soft input*).

- Codes en bloc

Les codes en bloc sont également très utilisés dans les communications numériques, et il en existe de nombreuses variantes. Ils diffèrent essentiellement des codes convolutionnels dans le sens où le codage en bloc associe à un ensemble de *k* symboles en entrée (le message) un autre ensemble de *n* symboles en sortie, indépendamment des symboles précédents fournis en entrée (*i.e.* des blocs précédents). Les *n* symboles obtenus forment un mot de code. *n* est appelé la *longueur* du code et *k* sa *dimension*. De nombreux codes en blocs sont linéaires, c'est-à-dire que les symboles de parité sont des combinaisons linéaires des symboles d'informations. Dans certains cas, le codage est réalisé de façon systématique, ce qui signifie qu'une partie des *n* symboles de sortie est constituée des *k* symboles d'entrée. Ces *k* symboles sont appelés symboles systématiques du mot de code. On introduit parfois le paramètre *h* comme étant le nombre de symboles de parité non systématiques. *h* est donc égal à *n-k*.

  - Codes de Reed-Solomon

Une catégorie importante de codes en blocs est représentée par les codes *Reed-Solomon* (RS). Ces codes sont de type *Maximum Distance Separable* (MDS), c'est-à-dire que la plus petite distance de Hamming, *d,* entre deux mots du code est exactement égale à *n-k*+1. D'une manière générale, le code RS n'est pas un code binaire. Ses éléments appartiennent au corps de Galois GF ($p=q^m$) constitué de *p* éléments, où *q* est un nombre premier et *m* est un entier positif. La longueur du code, *n*, est égale à q^m-1. Pour des raisons pratiques, *m* est très souvent choisi comme un multiple de 8, et *q* toujours choisi égal à 2. D'autre part, pour conserver des vitesses d'encodage/décodage raisonnables, *m* est limité à 16. *n* peut alors valoir soit 255 soit 65535. Il est aussi possible de raccourcir les mots de code. Les paramètres du code deviennent alors (*n'=n-F, k'=k-F, d'=d*), où *F* est le nombre de symboles fixés. La propriété MDS reste valable pour les codes RS raccourcis.

Raccourcir le code revient à fixer arbitrairement une partie des symboles d'informations dans le message. Non transmis, ces symboles sont ensuite réintroduits lors du décodage.

Le codage RS est réalisé grâce à un polynôme générateur particulier, G(x)[1]. Ce polynôme choisi, il convient de déterminer une méthode permettant d'associer à tout message M(x) un mot-code C(x) multiple de G(x). Une première solution consiste à calculer le produit des deux polynômes M et G, afin obtenir C(x)=M(x).G(x). On peut lui préférer une variante afin de mettre le codage sous forme systématique. Ceci permet de faciliter le décodage lorsque aucune erreur ne s'est produite. Le calcul associé à cette méthode est identique à celui du CRC qui sera explicité en 2.2.1.1.1.

Le décodage des codes RS dépend du type de canal sur lequel le codage agit : le canal binaire symétrique à erreurs (*Binary Symmetric Channel*, BSC) ou le canal binaire à effacements (*Binary Erasure Channe*l, BEC). Sur le BSC, caractéristique de la couche physique, les positions et le nombre d'erreurs par bloc ne sont pas connus *a priori*. Le code peut corriger jusqu'à $(d-1)/2$ erreurs par mot-code. Sur un canal de type BEC caractéristique des réseaux de transmissions (nous verrons pourquoi en 2.2.1), où les positions d'erreurs des symboles (appelées effacements) sont connues, jusqu'à $d-1$ effacements sont corrigibles par mot-code. Ceci rend le codage MDS optimal sur le canal BEC, car il suffit de recevoir k symboles du mot-code parmi les n pour réussir son décodage. En revanche, si le mot-code reçu comporte plus d'erreurs ou d'effacements que toléré par le codage et si cette combinaison n'est pas un autre mot-code, le décodeur détecte l'erreur mais ne peut la corriger. Le décodage RS est classiquement effectué de la façon suivante :

1. *Détermination du polynôme syndrome ne dépendant que des erreurs ;*
2. *Localisation des positions d'erreurs ;*
3. *Corrections des erreurs dont les positions sont connues (effacements) ou qui ont été déterminées dans l'étape précédente.*

Figure 2-1 : Principe général de l'algorithme de décodage RS

- Les codes LDPC

Les codes *Low Density Parity Check* (LDPC) sont une autre catégorie de codes en blocs. Inventés en 1960 par Gallager [Gal63] puis oubliés, ils sont réapparus en 1995 dans [Mac95]. Ces codes, basés sur des graphes bipartites, sont construits par additions XOR comme les codes convolutionnels. Contrairement à beaucoup d'autres codes en bloc, la longueur de code utilisable en pratique est très grande (plusieurs milliers de symboles).

Comme pour les codes convolutionnels, le décodage des LDPC est complexe. À ce jour, les LDPC avec décodage itératif sont les codes les plus performants que l'on ait découvert sur le canal BSC, mais pour des longueurs de code infinies. Pour un taux de code $\rho=½$, des implémentations ont mesuré un écart avec la limite de Shannon d'environ 0.1 dB pour les LDPC dits irréguliers, avec des longueurs de code extrêmes (n égal à 10^6) [Ric01]. Pour des longueurs de code plus petites en revanche, la supériorité des LDPC sur leurs concurrents (*i.e.* les *turbo-codes*) n'est pas encore clairement démontrée. Les caractéristiques des codes LDPC sur les canaux BEC seront discutées dans les parties suivantes.

[1] Les racines du polynôme générateur doivent être puissances successives de α^i, où α est un élément primitif du corps $GF(q^m)$ et i entier, avec $n-k$ racines distinctes.

2.1.2.3.1.2 Combinaison de plusieurs codes

- Codes produits

Avec les codes produits, deux codages de même nature et construits sur le même corps sont mis en œuvre simultanément. Si l'on représente sous forme matricielle les données à coder, le premier code agit ligne par ligne alors que le second agit colonne par colonne. Les *turbo-codes* sont les codes produits les plus populaires aujourd'hui ; ils sont construits par entrelacement entre deux codages convolutionnels. Bien que complexes, ces approches permettent d'obtenir des performances proches de la limite de Shannon à moins de 1 dB [Ber93]. Un autre exemple est le code *Cross Interleaved Reed-Solomon Code* (CIRC), code produits de deux codes Reed-Solomon, mis en œuvre sur les disques compacts (CD).

- Codes concaténés

Avant que les LDPC et les turbo-codes ne soient utilisés, les codes concaténés ont connu et connaissent encore un grand succès. Le principe de la concaténation est de placer les codes (en général deux) en cascade. Contrairement aux codes produits, les opérations de codage/décodage sont réalisées en phases distinctes, ce qui permet la combinaison de n'importe quels types de codes. L'intérêt de cette concaténation est de cumuler les avantages des codes. Ainsi, il est possible de disposer de mots de codes longs avec une bonne capacité de correction tout en conservant une faible complexité [Cla81]. Aussi, on privilégie souvent un code convolutionnel en codeur interne, et un codage en bloc (de type RS) à haut rendement en codage externe. Pour disperser les erreurs à la sortie du décodage interne, un entrelaceur qui modifie l'ordre de transmission naturel des données est placé entre les deux codeurs. L'exemple d'un tel système de codage sera donné en 2.2.2.1.

2.1.2.3.1.3 Impact sur le bilan de liaison

Avec un codage de taux de codage ρ, (2.2) se généralise en :

$$\frac{C}{N_0} = \frac{E_b}{N_0}.\frac{R}{\rho} \tag{2.7}$$

On peut montrer [Cla81] que pour n'importe quel type de codage, il est possible de rendre aussi faible que l'on veut le taux d'erreur binaire (*Bit Error Rate*, BER) dans la limite de la capacité du canal. Ceci impose que la condition suivante soit vérifiée :

$$\rho < \rho_{\max}$$
$$avec\ \rho_{\max} = \frac{1}{2}.\log_2\left(1 + 2.\rho\frac{E_b}{N_0}\right) \tag{2.8}$$

On peut également montrer que pour une source binaire équiprobable et un canal gaussien, il est possible de transmettre avec une probabilité d'erreur P_e, si et seulement si le rapport E_b/N_0 vérifie :

$$\frac{E_b}{N_0} > \frac{2^{2.\rho((1-h(Pe))} - 1}{2.\rho} \tag{2.9}$$
$$où\ h(P_e) = -P_e\log_2(P_e) - (1-P_e)\log_2(1-P_e)$$

Cette équation définit la limite infranchissable des performances de n'importe quel système de codage d'erreur. Elle permet de remarquer que la réduction excessive du taux de codage est sans intérêt, puisque prise pour ρ tendant vers 0, elle est bornée par :

$$\left(\frac{E_b}{N_0}\right)_{\min} = \left(1 - h(P_e)\right).\ln(2) \tag{2.10}$$

2.1.2.3.2 FMT

Les techniques de lutte contre les évanouissements (ou *Fade Mitigation Techniques*, FMT) sont un ensemble de techniques adaptatives permettant de renforcer la fiabilité de la liaison quand cela est jugé nécessaire [Cas98], par exemple en cas de dégradations météorologiques importantes. Ces techniques portent en particulier sur :

- le contrôle de puissance adaptatif ;
- les techniques de traitement de signal, dont l'adaptation de la modulation, du taux de codage, et la réduction du débit de la porteuse ;
- la diversité de site (si disponible), permettant d'emprunter une autre liaison satellite.

Bien que très attrayantes car elles permettent d'améliorer l'utilisation du système, ces techniques ajoutent une complexité importante pour détecter ou prédire les affaiblissements et pour choisir la technique de compensation et les paramètres appropriés. Aussi choisit-on souvent de travailler sans FMT, en réservant simplement une marge dans le bilan de liaison. Il est ensuite possible, à partir de cette marge, de déterminer la disponibilité statistique du système.

2.2 CONTRÔLE ET AMÉLIORATION DE LA FIABILITÉ DANS LES COUCHES DE COMMUNICATION

2.2.1 ***Notions générales***

2.2.1.1 Mécanismes de contrôle

Nous avons vu précédemment que dans les réseaux terrestres traditionnels, la couche physique est exempte d'erreurs de transmissions, ou du moins que ces évènements sont si rares qu'ils n'entraînent aucune conséquence sur le service offert à la couche supérieure. Aussi, quelle que soit la couche protocolaire considérée, le contrôle des erreurs de transmissions passe toujours par celui de l'intégrité des unités de données après réception.

2.2.1.1.1 Détections d'erreurs au sein des paquets de données

Dans les réseaux à commutation de paquets, les paquets sont acheminés par bonds successifs entre les routeurs, suivant la politique *Store and Forward*. Puisque les erreurs de transmission peuvent se produire sur chaque lien, une détection d'erreur a systématiquement lieu après chaque bond.

Le principe consiste à calculer, pour chaque paquet, une « signature » assez courte (souvent 32 bits sont jugés suffisants). Son calcul est effectué avant la transmission et la signature obtenue est concaténée à la fin du paquet. À la réception, le calcul est reproduit. Le paquet est jugé avoir été transmis sans erreur si le second résultat est identique à la signature.

2.2.1.1.1.1 La somme de contrôle

La somme de contrôle (*checksum*) est une technique très simple pouvant détecter des erreurs. Dans un premier temps, la séquence à protéger est séparée en mots de N bits (p.ex. 8 bits). Ces mots sont ensuite combinés d'une certaine façon afin obtenir la « signature ».

Typiquement, IP, TCP et UDP ont recours à ce mécanisme. Leur somme de contrôle est effectuée est considérant des mots de 16 bits (un octet supplémentaire à 0 est ajouté pour le calcul si le nombre d'octets du message est impair). La combinaison est calculée comme étant le complément à un sur 16 bits de la somme des compléments à un de la séquence prise par mots de 16 bits.

La somme de contrôle a l'avantage d'être rapide à calculer. En contrepartie, elle est imparfaite. Ainsi, il peut suffire que 2 erreurs bit se produisent sur la même position dans des mots de 16 bits différents pour que l'erreur ne soit pas détectée.

2.2.1.1.1.2 Les CRC

Afin de pallier cette limitation, des moyens de détection plus puissants existent et sont couramment utilisés. Ils sont basés sur des codes de la famille des codes cycliques appelés *Cyclic Redundancy Check* (CRC). Le terme cyclique provient du fait que toute permutation circulaire d'un mot du code est encore un mot du code. On peut souligner que les codes RS appartiennent à la famille des codes cycliques.

Le code est toujours construit sur le corps GF(2) afin de permettre des calculs rapides. Comme pour le code RS, le calcul du CRC fait intervenir un polynôme générateur G(x). Le degré du polynôme G(x), h, est choisi par commodité comme étant un multiple de 8 (généralement h=8, 16 ou 32).

Pour le calcul du CRC en lui-même, on commence par concaténer à la séquence h bits à 0, ce qui revient à calculer P(x), le polynôme « décalé » de M(x) de h positions :

$$P(x) = M(x).x^h \tag{2.11}$$

Le mot-code C(x) est ensuite calculé, comme étant égal à :

$$C(x) = P(x) + \left(P(x) \bmod G(x)\right) \tag{2.12}$$

Appelons R(x) le polynôme donné par :

$$R(x) = P(x) \bmod G(x) \tag{2.13}$$

Par définition, R(x) est un polynôme de degré inférieur ou égal à h-1 ; il est donc adressable sur h bits. Ces bits sont appelés par abus de langage le CRC du message.

On peut montrer que ce système détecte à coup sûr toutes corruptions d'au plus h bits parmi les k+h bits transmis, ainsi qu'une majorité des autres erreurs.

Pour avoir une idée de la différence de performances entre CRC et somme de contrôle, on peut se référer à [Sto98]. Il y est stipulé que pour une distribution uniforme d'erreur avec un CRC-32, la probabilité de non-détection d'erreur est de 1 sur 2^{32}, alors que cette même probabilité est estimée à 1 sur 2^{16} avec la somme de contrôle de TCP. En contrepartie, le CRC est légèrement plus lourd à calculer (pour des paquets de 1500 octets, une implémentation logicielle réalisée sur un Pentium III à 1 GHz donne une vitesse de 130 Mo/s pour le calcul du CRC et de 650 Mo/s pour celui de la somme de contrôle.)

Comme pour la somme de contrôle, les CRC sont couramment déployés dans les réseaux, et typiquement dans les couches basses comme Ethernet (CRC-32 appelé *Frame Check Sequence*, FCS), ATM (CRC-8) ou AAL-5 (CRC-32).

2.2.1.1.2 Rejets des paquets

Quand les paquets sont détectés erronés à n'importe quel niveau de la communication, ils sont toujours supprimés. La réparation des paquets manquants est alors reportée sur d'autres couches de la communication, typiquement au niveau transport. Au préalable, avant de décider des réparations éventuelles à effectuer, il doit déceler les paquets manquants dans le flux.

2.2.1.1.3 Détections des pertes de paquets

Un mécanisme simple permet la détection de perte de paquets : il suffit d'inclure dans chaque paquet transmis un numéro incrémenté (on parle de numéro de séquence). La perte de paquets est alors repérée par une rupture du numéro de séquence dans le flux. Cette solution ne fonctionne que si les paquets sont garantis sans erreur (le numéro de séquence pourrait alors être corrompu).

En plus de ce mécanisme, le processus de réception peut « déclarer » une perte sur expiration d'un *timer*, quand un ou plusieurs paquets attendus ne sont pas reçus à temps. Cela peut permettre d'améliorer la réactivité des protocoles.

2.2.1.2 Mécanismes de corrections des pertes de paquets

2.2.1.2.1 Reprise sur erreur : ARQ

La technique classique de reprise sur erreur (ARQ) existe traditionnellement sous 4 variantes : *Stop and Wait*, *Polling*, *Go-Back-N*, et *Selective Repeat*. La sélection de la variante entraîne, comme il sera montré, d'importantes conséquences sur l'efficacité de la transmission.

Suivant la variante, les retransmissions se font avec des acquittements positifs (*ACKnowledgements*, ACKs) ou négatifs (NACKs). Ils informent la source de manière implicite ou explicite de la bonne réception des données par le ou les récepteurs. Dans la suite de protocoles reposant sur TCP/IP, la reprise sur erreur est traditionnellement opérée au niveau transport, lorsque la machine réceptrice a détecté des pertes. Toutefois, certains protocoles peuvent gérer les retransmissions au niveau liaison pour offrir un service de fiabilité totale ; c'est le cas pour le protocole *High-level Data Link Control* (HDLC) de l'*International Organization for Standardization* (ISO) ou pour des protocoles mis en œuvre sur liaisons mobiles, par exemple pour *Universal Mobile Telecommunications System* (UMTS).

2.2.1.2.1.1 Stop and Wait

C'est la variante la plus simple, mais la moins performante. Pour chaque unité de données transmise, l'émetteur attend un accusé de réception (ACK) de la part du récepteur. Dans le cadre de communications IP Multicast, ceci est particulièrement inadapté car l'émetteur devrait bloquer la transmission de tout nouveau paquet, dans l'attente de l'acquittement du paquet précédent par tous les récepteurs. D'autre part, cette approche sollicite abondamment l'utilisation de la voie de retour : pour un paquet transmis sur la voie aller, un paquet d'acquittement doit être transmis sur la voie de retour. Cette technique est particulièrement risquée si l'on se préoccupe de l'implosion à la source.

2.2.1.2.1.2 Polling

Dans ce mode, l'émetteur demande explicitement l'état de réception au(x) récepteur(s) périodiquement, mais pas pour chaque unité de données. Le récepteur indique en retour la dernière unité de données reçue. Cette variante est plus performante que *Stop and Wait*, mais son intérêt avec les sessions multipoints reste limité.

2.2.1.2.1.3 Go-Back-N

Go-Back-N est une variante plutôt orientée récepteur, dans le sens où c'est au récepteur de demander explicitement une retransmission à l'émetteur quand il n'a pas reçu des données (par NACK). Il s'aperçoit des pertes par rupture de séquence, et doit demander à l'émetteur de reprendre la transmission à partir de l'unité de données qui lui manque. Aussi, si les unités de données consécutives aux unités perdues ont été reçues correctement avant la retransmission de(s) l'unité(s) perdue(s), elles sont tout de même retransmises. Dans le contexte multipoints, cette variante demeure peu intéressante car un seul « mauvais » récepteur suffirait à ralentir la transmission pour tout le groupe.

2.2.1.2.1.4 Selective-Repeat

Enfin, *Selective-Repeat* est une amélioration de *Go-Back-N*, car seules les données perdues sont retransmises. Cette méthode est de loin celle qui est la plus efficace, en particulier pour les transmissions multipoints.

2.2.1.2.1.5 Améliorations

Des améliorations à ces quatre variantes ont été proposées, par exemple dans TCP grâce au mécanisme de fenêtres glissantes [Pos81]. Toutefois, ces techniques ne sont pas adaptées aux communications multipoints.

2.2.1.2.2 Codage correcteur d'erreurs

Certains protocoles de communications intègrent un mécanisme de corrections d'erreurs au niveau binaire, par exemple ATM. D'autre part, dans le contexte des communications multipoints ou d'applications point à point spécifiques (communications multimeéia en particulier), une nouvelle forme de codage d'erreur est apparue depuis quelques années au niveau des protocoles de transport. Dans ce type de codage, les symboles du code ne sont plus simplement des bits ou des octets, mais les paquets entiers de données. Dans ce cas, le codage sert à réparer les pertes des paquets dont l'emplacement dans le flux est déterminé ; le décodage se fait alors dans les conditions du canal BEC. Une mise en œuvre de ce codage sera donnée en 2.3.2.2.3.

2.2.2 ***Spécificités dans le contexte satellitaire avec IP sur DVB***

2.2.2.1 Couche physique/MPEG 2-TS

La norme DVB-S [Ets97a] spécifie la chaîne de codage/décodage qui doit être implémentée dans les terminaux satellite DVB-S. La chaîne est présentée dans la Figure 2-2. Les taux de code possibles du codage convolutionnel après poinçonnage sont : {1/2 ; 2/3 ; 3/4 ; 5/6 ; 7/8}. À la réception, la chaîne de décodage symétrique à celle de la Figure 2-2 est utilisée.

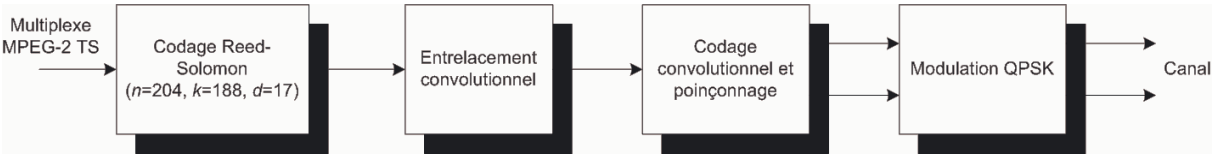

Figure 2-2 : Chaîne de codage et modulation dans DVB-S

Le choix des paramètres du code RS externe (n=204, k=188, d=17) provient de la taille des paquets du flux MPEG-2 TS, de 188 octets chacun ; chaque paquet MPEG-2 TS constitue ainsi un mot-code. Le code RS est un code raccourci bâti sur le code RS (n=255, k=239, d=17).

L'en-tête du paquet MPEG-2 TS (au minimum codé sur 4 octets) contient un champ *Transport Error Indicator* (plus simplement *Error*) qui sert à indiquer si une erreur est détectée dans le paquet lors de l'acheminement, cf. Figure 2-3. Si un paquet MPEG-2 ne peut pas être décodé correctement (le code ne peut pas corriger plus de 16 erreurs octets par paquet), les systèmes actuels l'éliminent afin d'éviter aux couches supérieures de le traiter inutilement.

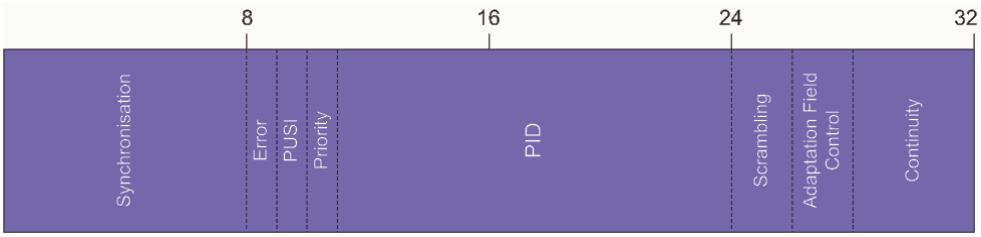

Figure 2-3 : Format de l'en-tête minimal des paquets MPEG-2 TS

Concernant la norme DVB-RCS [Ets03], deux types de flux peuvent être transportés directement : MPEG-2 TS ou ATM. La chaîne de codage originelle de DVB-S est supportée de manière optionnelle (il est d'ailleurs possible de ne pas appliquer le code RS externe), mais le turbo-codage est recommandé en raison de ses meilleures performances. Il est également possible de détecter les erreurs contenues dans la charge utile des paquets par un CRC.

2.2.2.2 MPE/ULE

MPE est le procédé classique servant à l'encapsulation des paquets IP dans le flux MPEG-2 TS. Les sections DSM-CC sur lesquelles s'appuie MPE sont limitées normativement à 4080 octets. Pour le transport de datagrammes IP, qui ne dépassent pas la MTU d'Ethernet (1500 octets de données) sous peine d'être fragmentés sur les réseaux Ethernet, il est possible d'encapsuler un datagramme IP par section.

Dans les systèmes actuels, les opérations de détections et rejets des unités de données se font en deux temps. D'abord, quand un paquet MPEG-2 appartenant à une section MPE est effacé, les paquets suivants de la même section sont aussi effacés directement par le processus de réassemblage. Cette action, qui permet d'alléger la charge de traitement, est en libre option de choix d'implémentation pour l'opérateur.

Ensuite, après réassemblage, une détection d'erreurs est faite par le biais d'un CRC-32 calculé pour l'ensemble de la section. Lorsque celle-ci est détectée erronée, la section est rejetée. Le polynôme générateur du CRC-32 est le même que pour Ethernet[2]. D'après ce qui a été dit précédemment, le rejet intervient en pratique rarement à ce niveau.

ULE, l'évolution possible de MPE faisant actuellement l'objet d'une proposition par l'IETF, repose sur la même technique de fiabilisation que MPE. En revanche, la MTU est augmentée dans ULE pour pouvoir supporter des MTU compatibles avec les réseaux de prochaine génération.

2.2.2.3 MPE-FEC

Dans la toute dernière version de la spécification DVB-S EN 301 192 [Ets04b], un codage FEC (RS, (n=255, k=191, d=64)) optionnel est prévu pour protéger la charge utile des sections DSM-CC. Le codage provient des spécifications du système DVB-H [Ets04a] destiné à l'origine à limiter les interférences dans les canaux mobiles. Le codage s'avère être assez flexible car la quantité de redondance transmise est modulable (par poinçonnage du code RS) afin que le taux de codage soit compris entre 0.75 et 1, avec une bonne granularité. Pour améliorer la capacité de

[2] $G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+1$

correction, le décodage MPE-FEC est effectué en mode effacements, car les erreurs proviennent de paquets rejetés (donc identifiés) par la couche MPE. En raison de sa très récente apparition, le mécanisme n'a pas été considéré dans la suite de l'étude.

2.2.2.4 IP

IP, dans la version 4, implémente une somme de contrôle dont le calcul est donné en 2.2.1.1.1. Celle-ci ne porte que sur l'en-tête des datagrammes afin d'alléger les traitements, puisque seuls les en-têtes IP sont modifiés à chaque traversée de routeurs. Lorsque la somme de contrôle calculée à la réception s'avère différente de celle indiquée dans le paquet, le datagramme est rejeté.

En revanche, avec IPv6, le mécanisme est abandonné, et les datagrammes ne contiennent plus de somme de contrôle. Cela peut se justifier par la fiabilité accrue des équipements d'interconnexion d'aujourd'hui par rapport à ceux d'il y a une vingtaine d'années et par une redondance avec la politique de fiabilisation employée sur la couche de niveau liaison, par exemple Ethernet.

2.3 LA FIABILITÉ DANS LES PROTOCOLES DE TRANSPORT MULTIPOINTS

2.3.1 *Introduction*

La couche transport est l'une des couches clé dans la gestion de la fiabilité de l'ensemble du processus de communication. Déjà en 1984, il était annoncé dans un célèbre article [Sal84], qu'il n'existe qu'un seul moyen de contrôler si le transfert de données entre stations s'est bien déroulé, quelle que soit la qualité du réseau de transmission sous-jacent. Ce moyen est de vérifier la validité des données reçues au point terminal de tout le processus, c'est à dire la couche transport. Plus tard, en 1990, il a été montré dans [Cla90] que son rôle était encore plus primordial. Les concepts d'*Application Level Framing* (ALF) et *Integrated Layer Processing* (ILP) y sont abordés. Selon eux, il est possible d'optimiser les mécanismes des couches hautes, dont la fiabilité, si les différents traitements ne sont pas effectués indépendamment et séquentiellement mais de façon « intégrée ». Par exemple, ALF stipule que les données applicatives à transmettre sur le réseau doivent être fragmentées par l'application elle-même, car elle est la seule à pouvoir déterminer la taille optimale des unités de données pour résoudre les problèmes de pertes, de déséquencement, etc. Dès lors, les mécanismes évoqués peuvent agir sur les mêmes unités de données en respectant ILP. Nous verrons dans cette partie en quoi le concept ILP est vérifié concernant la gestion de la fiabilité pour les communications multipoints.

Pour la fiabilité comme pour d'autres services, la couche de transport doit répondre aux besoins applicatifs tout en s'appuyant sur un réseau dont la qualité de service n'est pas garantie, comme IP. On peut illustrer schématiquement cette vision par une double contrainte exercée sur le niveau de transport, comme montré dans la Figure 2-4.

D'un point de vue moins théorique, la couche de transport offre une grande facilité de mise en œuvre, rendant possibles des techniques élaborées. Pour des raisons de simplicité, les sessions de communications basées sur IP Multicast sont programmées par l'intermédiaire des sockets, qui offrent aux développeurs toutes les fonctions nécessaires pour pouvoir facilement concevoir et tester un protocole de communication.

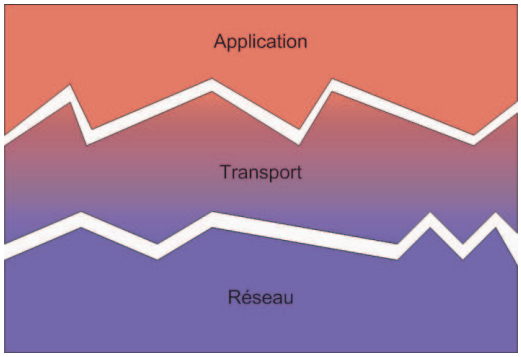

Figure 2-4 : Vue schématisée du niveau de transport dans une architecture de communication simplifiée en trois niveaux

2.3.2 ***Vue d'ensemble des mécanismes de la fiabilité***

2.3.2.1 Reprise sur erreur et implosion à la source

Les communications de groupe entre un grand nombre de récepteurs doivent être résistantes au facteur d'échelle. Cela implique de limiter autant que possible les messages de contrôle transitant sur la voie de retour. Ces messages doivent aussi être répartis dans le temps. Or, puisque certaines branches de l'arbre de diffusion permettent d'atteindre plusieurs récepteurs, ceux-ci peuvent expérimenter les mêmes pertes de paquets. On parle alors de pertes *corrélées*. Dans ce cas, il faut éviter qu'ils ne réagissent immédiatement aux pertes qu'ils détectent quasi simultanément. Si aucune action n'est prise à cette encontre, le risque de congestion, en particulier au niveau des routeurs voisins de la source, est extrême (implosion à la source). La surcharge de ces routeurs peut également entraîner le rejet d'autres types de paquets, par exemple ceux issus de la voie aller de la session multipoints, et risque de conduire à l'effondrement des performances de la communication.

On préfèrera avoir recours aux NACKs pour signaler les pertes à la place des ACKs qui indiquent explicitement quels paquets sont bien reçus, afin de diminuer l'utilisation de la voie de retour. Pour encore limiter l'implosion à la source, la technique dite de *suppression de NACK* peut être employée (cf. 2.3.2.4).

2.3.2.2 FEC au niveau transport

2.3.2.2.1 Intérêts des FEC au niveau transport

Le codage correcteur d'erreurs au niveau transport a été proposé dans les protocoles de transport multipoints pour contourner le double problème de l'utilisation de la voie de retour et de l'implosion à la source. Plutôt que de réparer les pertes de paquets *a posteriori* avec ARQ, l'emploi des FEC permet d'anticiper les pertes, limitant les pertes de temps relatives aux retransmissions, l'occupation des ressources du réseau et la charge de traitements à la source. De plus, des pertes de paquets différentes suivant les récepteurs peuvent être réparées par les mêmes paquets de redondance FEC, ce qui renforce la robustesse du protocole au facteur d'échelle. Enfin, la façon d'implémenter le codage par rapport aux autres mécanismes de fiabilisation peut aussi révéler d'autres atouts, cf. 2.3.2.3.2.

2.3.2.2.2 Types de codes employés

La plupart des codes FEC employés au niveau transport sont des codes en blocs, car ils se prêtent bien au canal à effacements BEC, caractéristique des réseaux de communications, et que

la transmission (resp. le codage) se fait indépendamment par paquet (resp. par mot-code). Aujourd'hui, les deux meilleures options sont les codes MDS et les codes LDPC.

2.3.2.2.2.1 Codes MDS

Dans les canaux BEC, les algorithmes de décodage classiques du code Reed-Solomon (tel que celui présenté en 2.1.2.3.1) ne sont pas les plus rapides, car la complexité du décodage dépend du nombre de mots (*i.e.* de la longueur) contenus dans les paquets. Les premières implémentations « rapides » de codes MDS au niveau transport ont été faites par L. Rizzo grâce aux matrices de Vandermonde utilisées comme matrices génératrices du code [Riz97]. Une variante décrite dans [Blo95] utilise des matrices de Cauchy. La propriété exploitée dans ces formes de codes MDS est que le décodage nécessite une inversion matricielle par bloc de paquets à décoder, suivie d'une multiplication matrice-vecteur. Toutefois l'inversion de la matrice génératrice du code n'est pas toujours possible, et ces codes ne remplissent pas toujours leur fonction. Une adaptation a récemment été proposée [Lac03].

Pratiquement, il faut noter que les codes MDS peuvent être mis en œuvre sur les corps $GF(2^8)$ ou $GF(2^{16})$, mais de sévères limitations concernent la complexité sur ce dernier.

2.3.2.2.2.2 Codes LDPC

Plusieurs variétés utilisant les LDPC ont été proposées pour le canal BEC : les codes Tornado [Lub97] peu utilisables en pratique, les codes LT [Lub02b], les codes Raptor [Sho04] ou encore les codes *Low Density Generator Matrix* (LDGM) [Roca03]. Grâce à la longueur potentielle des mots-codes quasiment infinie des LDPC, il est possible de coder n'importe quel volume de données en un seul mot-code. Les implémentations logicielles de ce décodage se sont vite répandues, et sont à l'origine du succès de Digital Fountain [Lub98] en raison de leur rapidité d'exécution. De plus, ce type de codage est le plus robuste aux pertes en rafales. Les LDPC sont toutefois moins efficaces que les codes MDS, car il faut recevoir $k.f$ (à la place de k) symboles pour décoder le mot de code. f est appelé le *facteur d'inefficacité*, car il est strictement supérieur à 1. Sa valeur précise dépend de la configuration d'erreurs, des paramètres du code et de l'implémentation précise ; dans les meilleurs cas on peut l'estimer en moyenne entre 1.05 et 1.1 pour des taux de codage supérieurs à ½ [Pla03].

2.3.2.2.2.3 Comparaison de performances des codes

Le Tableau 2-1 résume de façon indicative les avantages et inconvénients (+/-) de chacun des codes. Une comparaison plus précise doit prendre en compte le contexte exact (paramètres du code, taux de pertes moyen, distribution des pertes, besoin applicatifs, contraintes particulières). Deux métriques, en vue d'une comparaison numérique, seront proposées au cours du chapitre 3.

	Codes LDPC	Codes MDS $GF(2^8)$	Codes MDS $GF(2^{16})$
Inefficacité du code	-	+	+
Robustesse aux rafales d'effacements[3]	+	-	+
Complexité des opérations	+	+	-

Tableau 2-1 : Points forts et faiblesses des codages mis en œuvre au niveau du transport

2.3.2.2.3 Mise en œuvre du codage FEC dans les protocoles de transport : FEC au niveau paquets

[3] En supposant que les transmissions se font sans entrelacement (cf. chapitre 3).

Le codage au niveau transport est effectué de manière assez différente du codage sur le canal BSC. La Figure 2-5 illustre le principe du codage.

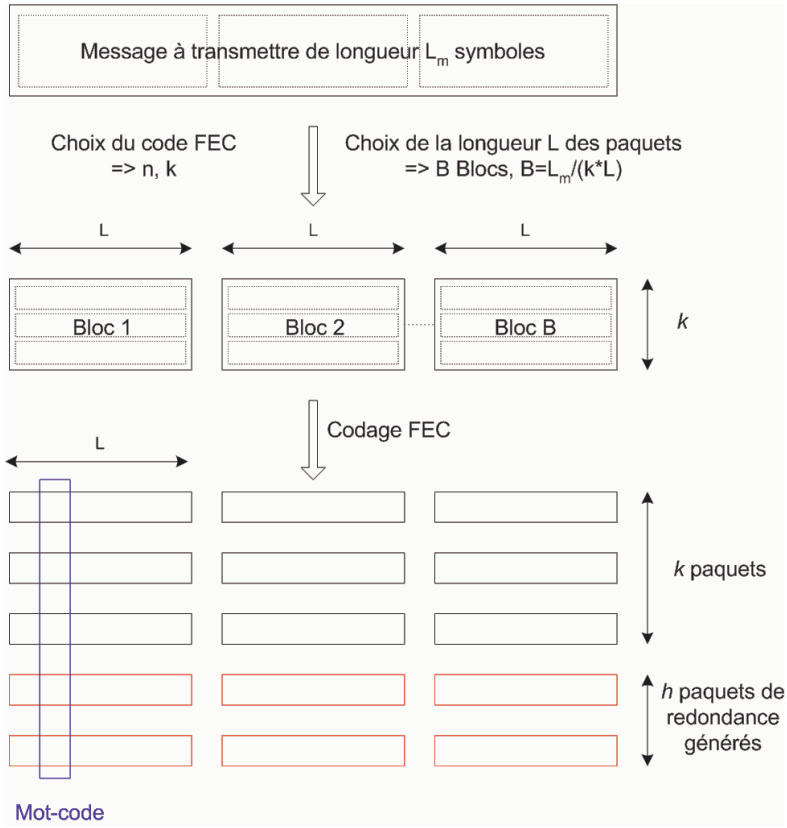

Figure 2-5 : Le codage FEC en mode paquets

Le message à coder, de L_m symboles, est d'abord fragmenté en B blocs (parfois dénommés groupes de transmission, *transmission group*) de k paquets. Ces paquets sont composés de L symboles, chacun de ces symboles étant adressés sur m bits, où m est le paramètre du corps sur lequel le code est construit (cf. 2.1.2.3.1.1). Les L symboles serviront à constituer L mots de code dans le bloc. Un de ces mots de code est encadré en bleu sur la Figure 2-5.

Après la fragmentation, le codage (souvent préféré systématique) peut avoir lieu. Il produit $h=n-k$ paquets de parité par bloc. Selon que le code est de type LDPC ou MDS, il est possible de travailler avec des valeurs de k et n pratiquement aussi grande que l'on veut ou non. Dans ce dernier cas, le choix est limité en raison de la complexité de calcul nécessaire (cf. chapitre 3).

2.3.2.3 Utilisation conjointe de FEC et ARQ

FEC et ARQ sont deux techniques de fiabilisation qu'il est possible de combiner, pourvu qu'une voie de retour soit disponible. Deux approches sont envisageables : *Hybride ARQ de type I* et *Hybride ARQ de type II*. Nous décrivons ici leur fonctionnement de manière qualitative mais l'analyse de leurs performances sera faite au chapitre suivant, en 3.1.1.3 et 3.1.1.4

2.3.2.3.1 Hybride ARQ de type I

Dans ce type d'utilisation, FEC et ARQ sont mis en œuvre de manière indépendante l'un de l'autre. Le mécanisme de FEC produit les paquets de redondance à taux fixe, sans prendre en compte le fait que les récepteurs expérimentent des pertes ou non ; il sert simplement, comme pour le codage de la couche physique, à diminuer le taux de pertes de paquets observé par les récepteurs. Quand le codage est insuffisant, les paquets manquants sont retransmis par une

reprise sur erreur traditionnelle. Certaines études, [Ker98] ou [Li98], améliorent ce principe en adaptant le taux de codage dynamiquement en fonction des informations statistiques sur les pertes, informations recueillies par la source. Cette solution, attrayante, pose toutefois le problème de l'estimation du taux de redondance adéquat. Un des moyens de contourner le problème est la mise en œuvre d'une autre technique d'intégration FEC/ARQ : Hybride ARQ de type II.

2.3.2.3.2 Hybride ARQ de type II

Parfois aussi appelé fiabilisation par redondance incrémentale (*Incremental Redundancy*), cette variante place FEC et ARQ « au même niveau », conformément au concept ILP. Dans un premier temps, les paquets de parités sont générés sans être transmis. Ensuite, la transmission des données d'information (i.e. systématiques) a lieu. Ce n'est que lorsque des demandes de « réparations » (*i.e.* messages NACK) sont reçues par la source que les paquets de redondance sont envoyés. L'avantage de cette approche peut être illustré par l'exemple suivant. On suppose que :

- La fiabilité pour cette transmission doit être totale pour tous les récepteurs ;
- Il y a deux récepteurs (R_1 et R_2) dans le groupe ;
- La transmission est codée avec un code MDS (n=255, k=32) systématique ;
- Le message est fragmenté en k paquets de longueur identique, numérotés de 1 à k ;
- Aucun des paquets de réparation (ARQ ou FEC réactif) n'est perdu.

Après la transmission des k premiers paquets de données, on suppose qu'il manque à R_1 les paquets 6 à 10 (inclus), et à R_2 les paquets 11 à 20. Sans codage, et après réception des rapports de réception, la source devrait retransmettre tous les paquets perdus au moins par un récepteur, c'est-à-dire ici les paquets compris entre 6 et 20 (15 paquets). Avec Hybride ARQ de type II, il lui suffirait de transmettre 10 paquets de parité, par exemple les 10 premiers générés par le code. Dans ce cas précis, la quantité de données de « réparation » est divisée par 2 grâce au codage FEC.

Pour un groupe de grande taille, le codage FEC, en particulier avec ARQ hybride de type II, est particulièrement adapté. Par rapport à Hybride ARQ de type I, ce mécanisme a l'avantage d'optimiser la quantité de données envoyée au groupe afin que tous les membres puissent décoder tous les blocs. En revanche, aucune perte de paquets n'est tolérée sans recourir à des transmissions de réparations, induisant des délais supplémentaires. Il est toutefois possible de coupler Hybride ARQ I et II en transmettant une partie des h paquets de redondance de manière préventive afin de réduire ces délais.

2.3.2.4 Suppression de NACKs

Afin d'augmenter encore la résistance au facteur d'échelle des protocoles de transport multipoints se basant sur les NACKs, une technique appelée plus tard *Suppression de NACKs* a été proposée en 1997 [Flo97]. En plus d'envoyer les NACKs à la source, les récepteurs ayant perdu des paquets transmettent les NACKs à destination de tout le groupe. Partager cette information permet d'indiquer aux autres récepteurs, s'ils sont concernés par des pertes, qu'il leur est inutile de transmettre une demande déjà envoyée, ce qui peut limiter l'implosion au voisinage de la source.

En pratique, cela oblige les récepteurs qui détectent des pertes à retarder les requêtes de retransmission d'un délai aléatoire. Il a été montré qu'une distribution aléatoire exponentielle doit être choisie pour que le délai de réparation reste stable quand le nombre de récepteurs augmente

[Non98a]. Pour parvenir à un compromis entre le nombre de requêtes transmises inutilement et le délai moyen supplémentaire, le paramétrage dépend du nombre de récepteurs dans le groupe, du temps d'aller-retour (*Round Trip Time*, RTT) entre le récepteur et la source, et d'un facteur supplémentaire favorisant l'une des deux métriques. Toutefois, la détermination des deux premiers paramètres peut limiter les performances du mécanisme car le RTT et surtout le nombre de récepteurs présents dans le groupe sont variables. Il y a alors deux choix possibles. Le premier, plus simple, consistant à fixer statiquement les paramètres tout au long de la session. L'autre consiste à estimer dynamiquement ces valeurs, au risque d'alourdir le protocole par une signalisation supplémentaire.

2.3.3 *Les classes de protocoles fiables définies à l'IETF*

Dès son origine en 1999, le groupe de travail RMT s'est rendu compte qu'il était impossible de concevoir un protocole de transport unique garantissant un service de fiabilité totale et adapté à toutes les situations. En effet, les types d'applications et les caractéristiques des réseaux sont tellement diversifiés qu'un seul protocole n'est pas envisageable : une solution de protocole peut être optimale dans un cas (p.ex. transferts de fichiers fiables sur un réseau satellite) et complètement inadapté dans un autre (transferts multimédias à contraintes temporelles sur le MBone).

Le RMT a adopté deux types d'approches. La première consiste à proposer un ensemble d'outils indépendants (*Building Blocks*) répondant à des fonctionnalités précises (contrôle de congestion, FEC, etc…) et accessibles par le biais d'interfaces de programmation d'application (*Application Programming Interface*, API) standardisées. La seconde approche consiste à proposer des classes de protocoles définies selon leur architecture générale, sans toutefois rentrer dans des spécifications poussées. À l'origine, trois classes de protocoles ont été proposées : *TRee-ACK based* (TRACK), *NACK Oriented Reliable Multicast* (NORM), *Asynchronous Layered Coding* (ALC).

2.3.3.1 TRACK

Cette classe supposait l'envoi régulier d'acquittements positifs (ACK) par les membres du groupe à la source. Pour limiter l'implosion, les nœuds de l'arbre de diffusion sont dotés de fonctions permettant d'agréger les ACKs sur la voie de retour, limitant ainsi la quantité du trafic remontant à la source. Avec TRACK, il était également possible d'avoir recours à des retransmissions locales vers les seules branches de l'arbre qui en avaient besoin. Malgré l'intérêt des acquittements positifs et des retransmissions locales, la complexité nécessaire à supporter dans les routeurs a été jugée trop importante. D'autre part, le problème d'implosion n'était pas vraiment réglé. Cela a conduit à l'abandon de TRACK.

2.3.3.2 NORM

La deuxième classe de protocole proposée par le RMT, NORM, impose que les données manquantes soient détectées puis demandées par les récepteurs sous la forme d'acquittements négatifs (NACK) indiquant des requêtes de retransmission. Avec NORM, il est possible d'utiliser le codage FEC.

2.3.3.3 ALC

La classe ALC [Lub02a] est conçue pour être la plus robuste possible face au facteur d'échelle car elle n'utilise pas du tout la voie de retour. Elle exploite de façon intensive le codage FEC pour assurer un service de fiabilité statistique. En complément, un mécanisme de contrôle de congestion par couches cumulatives, orienté récepteur, est aussi déployé, permettant aux membres de recevoir les données le plus rapidement possible, tout en évitant de provoquer des

congestions. Similaire aux couches d'améliorations vidéo, cette technique est aussi une manière efficace de transmettre à destination d'un groupe hétérogène, du point de vue des puissances de traitement requises et des capacités réseau [Roc02].

Dans les cas où la transmission n'est pas trop contraignante sur la voie aller (larges ressources disponibles) mais beaucoup plus sur la voie de retour, ALC est tout à fait adapté. En revanche, il lui est impossible d'optimiser la transmission sur la voie aller puisqu'elle ne peut pas anticiper le paramétrage de code idéal, ni avoir la confirmation que les récepteurs ont reçu correctement les données. De plus, il a été souligné qu'avec ALC, l'organisation des données à transmettre peut être problématique pour que les récepteurs d'un groupe hétérogène reçoivent individuellement l'intégralité des données en un minimum de temps. Des solutions d'ordonnancement et de répartition des données dans les couches été proposées selon le nombre d'objets à transférer (cf. [Cle98], [Roc02], [Don99], [Bha98]).

2.3.3.4 Gestion de la fiabilité dans les protocoles de transport multipoints existants

Avant la création du RMT, de nombreux protocoles multipoints fiables ont été proposés, et ont permis d'introduire les différentes classes. Vu leur nombre, il nous est impossible de détailler individuellement leur fonctionnement ici. En revanche, nous proposons de les présenter selon la façon dont ils organisent la gestion de la fiabilité.

Plusieurs articles ([Dio97], [Lev98] ou encore [Fai01]) ont fait l'inventaire de techniques associées aux communications multipoints. De manière générale, les protocoles de transport multipoints peuvent être séparés en plusieurs classes : ceux où la gestion de la fiabilité est à l'initiative de l'émetteur (ACK/NACK), à l'initiative des récepteurs (NACKs seulement), ceux qui reposent sur une structure en arbre hiérarchique, et enfin ceux qui organisent les récepteurs dans une structure d'anneau.

La classe où la fiabilité est gérée à l'initiative de la source est la première à avoir été étudiée. Elle impose à la source d'établir la liste des participants, et à chaque paquet (ou séquence de paquets) transmis, la source doit attendre un accusé de réception explicite de la part de chacun des membres. Pour détecter qu'un paquet est perdu, la source doit attendre l'expiration d'un *timer* déclenché au moment de l'envoi. D'après ce qui a été dit précédemment, on comprend que cette approche n'offre pas une bonne mise à l'échelle. On peut cependant l'améliorer en partie si la technique du *Selective Repeat* est employée (seuls les paquets perdus sont retransmis), et si l'on regroupe les messages ACK en acquittant périodiquement les paquets par séquence.

Dans cette seconde classe déjà évoquée au travers de NORM, seuls les récepteurs sont à l'origine de la détection des pertes et des demandes de retransmissions. Beaucoup de protocoles sont fondés sur cette classe, p.ex. URGC [Aei93], MTP [Bor94], RMDP [Riz97], AFDP [Coo96], RRMP [Cla99]. Dans [Tow97], il est démontré que ces protocoles sont beaucoup plus efficaces que la classe précédente, en termes de nombre de paquets traités. Par exemple, dans un groupe de 1000 récepteurs et pour un taux d'erreur paquet de 10^{-2}, la source doit traiter au moins 100 fois moins de paquets qu'avec un protocole où les acquittements sont demandés par la source.

D'autres protocoles (RMTP [Pau97], LBRM [Hol95], OTERS [Li98], LGMP [Hof96], TMTP [Yav95], TRAM [Chi98] ou PGM [Spe01]) exploitent une structure en arbre hiérarchique pour gérer plus efficacement les demandes de réparation et les réparations elles-mêmes (réparations locales). Cela entraîne une meilleure mise à l'échelle et permet de diminuer les temps de réparation pour les récepteurs. Pour cela, un récepteur est désigné dans chaque branche de l'arbre comme étant le responsable d'un sous-groupe de récepteurs. Aussi, toutes les demandes et gestions des retransmissions d'un membre du sous-groupe passent par ce responsable. Il s'adresse lui-même au responsable du sous-groupe dont il fait partie, et ainsi de suite jusqu'à atteindre la source. Les gains de cette approche sont variables selon la situation (FEC, topologie

du groupe, répartition des erreurs) [Non98b]. En contrepartie, le protocole doit configurer l'arbre, ce qui implique une complexité accrue et une signalisation plus importante.

La dernière classe de protocoles repose sur une structure en anneau et n'est représentative que de deux protocoles : TRP [Cha84] et RMP [Whe94], qui en est une évolution. La classe vise des applications particulières où les transmissions et réparations doivent être synchronisées parmi le groupe. La synchronisation est assurée par le biais d'un jeton qui circule dans le groupe. De bonnes performances de l'approche ont été mesurées, et une certaine robustesse aux pertes. Toutefois, la complexité d'implémentation du protocole, sa faible résistance au facteur d'échelle (pour la synchronisation de l'anneau) et la spécificité du contexte ont empêché le déploiement massif de cette classe.

2.3.4 *Spécificités dans le contexte satellitaire*

La principale contrainte des protocoles terrestres multipoints, c'est-à-dire la robustesse au facteur d'échelle, est encore plus forte dans les réseaux par satellite. Nous pensons que la différence importante entre les deux cas provient des coûts d'utilisation de la bande passante dans les deux types de réseaux et de la nature différente des erreurs évoquées dans la partie 2.1. Aussi, nous ne développerons pas beaucoup cette partie car l'essentiel des arguments énoncés dans le cas du transport multipoints terrestre restent valables pour le contexte satellitaire.

2.3.4.1 Adéquation des mécanismes

Les raisons évoquées en 2.3.2.1, 2.3.2.2.1 et 2.3.2.4 s'appliquent de manière encore plus importante dans notre cadre de travail, où des tailles de groupes de plusieurs dizaines de milliers sont réalistes. De plus, le codage d'erreur est ici une solution utile à la fois pour réduire l'effet des congestions des réseaux terrestres, mais aussi pour réparer les pertes de paquets consécutives à des erreurs non corrigées provenant de la liaison satellite.

2.3.4.2 Adéquation des classes de protocoles

TRACK n'est pas vraiment adapté à notre cadre d'étude, en raison de la différence importante des deux types de topologies. Néanmoins, dans le cas où plusieurs stations se connectent aux mêmes terminaux satellite par l'intermédiaire de plusieurs routeurs, les fonctionnalités d'agrégation des NACK et de reprise d'erreur locale (en cas de congestions des réseaux en aval de la liaison satellite) peuvent augmenter la mise à l'échelle et la réactivité du protocole.

NORM apparaît ici comme étant la plus complexe à mettre en œuvre, mais aussi comme étant la plus performante en ce qui concerne les délais de réparations de pertes, et l'efficacité de transmission (sur la voie aller). Pour ces raisons, nous privilégierons son utilisation dans le cadre du service de fiabilité totale.

La classe ALC est par nature la plus propice au satellite car elle est sans conteste la plus simple et surtout la plus résistante au facteur d'échelle. Son support de l'hétérogénéité des récepteurs est aussi un avantage décisif. En contrepartie, les performances de cette classe sont sensibles aux paramètres du codage, à moins d'accepter qu'une partie des récepteurs puisse ne pas recevoir la totalité du message applicatif lorsque de nombreuses erreurs surviennent.

CONCLUSION

Au travers de ce chapitre nous avons mis en évidence que plusieurs phénomènes, bien identifiés depuis une dizaine d'années par de nombreuses études, sont des obstacles à la gestion de la fiabilité pour les communications multipoints. Ces phénomènes (pertes de paquets, grands groupes, etc.…) sont d'autant plus mis en avant dans le domaine des réseaux par satellite, où il paraît encore plus important de les caractériser et de les prévoir.

Il semble donc que la réelle difficulté se situe aujourd'hui dans la diversité des situations dans lesquelles les protocoles de transport multipoints doivent agir. Les tentatives tardives de normalisations du groupe RMT de l'IETF confirment la réalité de cette difficulté. D'autre part, devant cette diversité, les performances des protocoles ne peuvent pas être optimisées simultanément. Il devient alors important de discerner, parmi tous les critères d'évaluations possibles, ceux qui sont les plus essentiels dans le contexte considéré.

Le cas particulier du paramétrage du codage FEC est au cœur de ce problème. Une analyse approfondie de sa mise en œuvre semble donc nécessaire. Cette analyse sera l'objet du chapitre suivant. Elle situera les compromis à accepter sur les différents critères dans le cadre des communications par satellite.

BIBLIOGRAPHIE

[Aie93] "Design of a Reliable Multicast Protocol", R. Aiello, E. Pagani and G. P. Rossi, Proceedings of IEEE INFOCOM, pp.75-81, 1993.

[Ber93] "Near Shannon Limit Error Correction Coding and Decoding : Turbo-Codes", C. Berrou, A. Glavieux, P. Thitimajshima, in Proc., IEEE Int. Conf. on Communications., Geneva, Switzerland, pp. 1064-1070, mai 1993.

[Bha98] "Efficient rate-controlled bulk data transfer using multiple multicast groups", S. Bhattacharyya, in proceedings of IEEE INFOCOM, pp. 1172-1179, juin 1998.

[Blo95] "An XOR-Based Erasure Resilient Coding Scheme", J. Blomer *et al.*, ICSI Technical Report TR 95-048, août 1995.

[Bor94] "MTP-2: Towards Achieving the S.E.R.O. Properties for Multicast Transport", C. Bormann et *al.*, in IEEE/ICCCN'94, San Francisco, USA, 1994.

[Cas98] "Fade Mitigation Techniques for New SatCom Services at Ku-band and Above : a Review", L. Castanet, J. Lemorton, M. Bousquet, Fourth Ka-Band Utilization Conference, Venice, Italy, November 2 - 4, 1998.

[Cha84] "Reliable broadcast protocols", Chang J-M, Maxemchuk NF, ACM Trans Comput Syst 2(3), pp 251–273, 1984.

[Chi98] "TRAM: A Tree-based Reliable Multicast Protocol", D. Chiu *et al.*, Sun Microsystems Labs, Chelmsford, MA, USA, Technical Report SML TR-98-66, juillet 1998.

[Cla81] "Error-Correction Coding for Digital Communications", G.C. Clark, J. B. Cain, Plenum Press, p.43, New York, 1981.

[Cla90] "Architectural Considerations for a New Generation of Protocols", D.Clark, D. Tennenhouse, in ACM SigComm, 1990.

[Cla99] "Internet over Direct Broadcast Satellites", H.D. Clausen, H. Linder, B.Collini-Nocker: IEEE Communications Magazine, Vol. 37, No.6, pp. 146-151, juin 1999.

[Cle98] "Organizing Data Transmission for Reliable Multicast over Satellite Links", A. Clerget, W. Dabbous, in proceedings of Africom'98, 1998.

[Coo96] "Why Use a Fishing Line When You Have a Net? An Adaptive Multicast Data Distribution Protocol", J. R. Cooperstock, S. Kotsopolous, Proc. USENIX '96, 1996.

[Dio97] "Multipoint Communication: A Survey of Protocols, Functions and Mechanisms", C. Diot, W. Dabbous, J. Crowcroft, IEEE Journal on Selected Area in Communication, vol. 15, n°3, avril 1997.

[Don99] "Multiple-channel Multicast Scheduling for Scalable Bulk-data Transport", M. J. Donahoo, M. H. Ammar, and E. W. Zegura, Proceedings of the INFOCOM'99, mars 1999.

[Ets03] ETSI EN 301 790, "Digital Video Broadcasting (DVB) ; Interaction channel for satellite distribution systems", v1.3.1, août 2003.

[Ets97a] ETSI EN 300 421, "Digital Video Broadcasting (DVB) ; Framing structure, channel coding and modulation for 11/12 GHz satellite services", v1.1.2, août 1997.

[Ets04a] ETSI EN 302 304 "Digital Video Broadcasting (DVB) ; Transmission System for Handheld Terminals (DVB-H)", v1.1.1, juin 2004.

[Ets04b] ETSI EN 301 192, "Digital Video Broadcasting (DVB) ; DVB specification for data broadcasting", v1.4.1, juin 2004.

[Fai01] "Reliable Multicast via Satellite: A Comparison Survey and Taxonomy", G.Fairhurst, M. Koyabe, International Journal of Satellite Communications (IJSC), vol: 24(1), 21-26, 2001.

[Flo97] "A Reliable Multicast Framework for Light-Weight sessions and Application Level Framing", S. Floyd *et al.*, IEEE/ACM Transaction on networking, pp. 784-803, décembre 1997.

[Gal63] "Low-Density Parity-Check Codes", R. G. Gallager, MIT Press, Cambridge, MA, 1963.

[Geo01] "Radio Layer Analysis for Geocast Networks Simulations", L. Castanet *et al.*, GEOC-ASPI-TN-40, juin 2001

[Han97] "A Comparison of Architecture and Performance between Reliable Multicast Protocols over the MBone", C. Hanle, Institute of Telematics, Karlsruhe, 1997.

[Hof96] "A Generic Concept for Large-Scale Multicast", M. Hofmann, in International Zurich Seminar on Digital Communication, Zurich, Switzerland, 1996.

[Hol95] "Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation", H.W. Holbrook, S.K. Singhal, and D. R. Cheriton, ACM SIGCOMM, 25(4), pp. 328 - 341, 1995.

[Iee02] IEEE 802.3, "Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications", mars 2002.

[Ker98] "Scoped Hybrid Automatic Repeat Request with Forward Error Correction (SHARQFEC)", R.

G. Kermode, SIGCOMM'98, Computer Communication Review vol.28, no.4 p. 278-89, octobre 1998.

[Lac03] "A Construction of Matrices With No Singular Square Submatrices", J. Lacan, J. Fimes, International Conference on Finite Fields and Applications, mai 2003.

[Lev98] "A Comparison of Reliable Multicast Protocols", B.N. Levine and J. J. Garcia-Luna-Aceves, ACM Multimedia Systems Journal, 6(5), 334-348, 1998.

[Li98] "OTERS (On-Tree Efficient Recovery using Subcasting): A Reliable Multicast Protocol", D. Li and D. R. Cheriton, in IEEE International Conference on Network Protocols, Texas, USA, 237-245, 1998.

[Li99] "Evaluating the utility of FEC with Reliable Multicast", D. Li, D. R. Cheriton, Proc. of 7th IEEE International Conference on Network Protocols (ICNP'99), 1999.

[Lin97] "A Forward Error Correction Based Multicast Transport Protocol for Multimedia Applications in Satellite Environments", H. Linder *et al.*, IEEE IPCCC 1997, pp 419-425, février 1997.

[Lub02a] IETF RFC, "Asynchronous Layered Coding (ALC) Protocol Instantiation", M. Luby *et al.*, RFC 3450, décembre 2002.

[Lub02b] "LT codes", M. Luby, 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002.

[Lub97] "Pratical Loss-Resilient Codes", M. Luby *et al.*, Proc. of the Twenty-Ninth Annual ACM Symp. on Theory of Computing, El Paso, pages 150–159, Texas, mai 1997.

[Lub98] "A Digital Fountain Approach to Reliable Distribution of Bulk Data", H. Byers, M. Luby, M. Mitzenmacher, and A. Rege, in Proceedings of ACM SIGCOMM'98, Vancouver, BC, 1998.

[Mac95] "Good codes based on very sparse matrices", D. MacKay and R. Neal, 5th IAM Conference: Cryptography and Coding, LNCS No. 1025, pp100-111, 1995.

[Non98a] "Optimal multicast feedback", J. Nonnenmacher and E. Biersack, in proceedings of IEEE INFOCOM'98, vol. 3, pp. 964-971, San Francisco, CA USA, mars 1998.

[Non98b] "How bad is Reliable Multicast without Local Recovery", J. Nonnenmacher *et al.*, Tech. Rep., Institut EURECOM, Sophia-Antipolis, France, janvier 1998.

[Pau97] "Reliable Multicast Transport Protocol (RMTP)", S. Paul *et al.*, IEEE Journal on Selected Areas in Communications, 15(3), pp. 407 - 421, 1997.

[Pla03] "On the Practical Use of LDPC Erasure Codes for Distributed Storage Applications", J.S Planck, M.G. Thomason, Technical Report UT-CS-03-510, University of Tennessee, septembre 2003.

[Pos81] IETF RFC, "Transmission Control Protocol", J. Postel, RFC 793, septembre 1981.

[Ric01] "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes", T.J. Richardson, IEEE Trans. on Information Theory, Vol. 47, n.2, février 2001.

[Riz97] "A Reliable Multicast Data Distribution Protocol Based on Software FEC Techniques", L. Rizzo, L. Vicisano, in proceedings of HPCS '97, Greece, juin 1997.

[ROC] http://www.inrialpes.fr/planete/people/roca/mcl/ldpc_infos.html

[Roc02] "Design of a Multicast File Transfer Tool on Top of ALC", V. Roca, B. Mordelet, 7th IEEE Symposium on Computers and Communications (ISCC'02), Toarmina, Italie, juillet 2002.

[Roc03] "Design and Evaluation of a Low Density Generator Matrix (LDGM) large block FEC codec", V. Roca, Z. Khallouf, J. Laboure, Fifth International Workshop on Networked Group Communication (NGC'03), Munich, Germany, septembre 2003.

[Sal84] "End-to-end Arguments in System Design", J.H Saltzer, D.P Reed., and D.D Clark, ACM Transactions on Computer Systems, pp. 277-288, 1984.

[Sho04] Digital Fountain White Paper, "Raptor Codes", A. Shokrollahi, http://www.digitalfountain.com/technology/whitePapers.cfm, mai 2004.

[Spe01] IETF RFC, "PGM Reliable Transport Protocol Specification", T. Speakman *et al.*, RFC 3208, décembre 2001.

[Spe03] IETF Internet Draft, "Reliable Multicast Transport Building Block, Generic Router Assist - Signalling Protocol Specification", T. Speakman, L. Vicisano, draft-ietf-rmt-bb-gra-signalling-00.txt, janvier 2003.

[Sto98] "Performance of Checksum and CRC over Real Data", J. Stone *et al.*, IEEE ACM Transactions on Networking, 1998.

[Tow97] "A comparison of sender-initiated and receiver-initiated reliable multicast protocols", D. Towsley, J. Kurose, and S. Pingali, Selected Areas in Communications, IEEE Journal on, pp. 398 –406, avril 1997.

[Wha98] IETF Internet Draft, "The RMTP-II Protocol", B. Whatten *et al.*, draft-whetten-rmtp-ii-00.txt, avril 1998.

[Whe94] "A high-performance, totally ordered multicast protocol", B. Whetten, S. Kaplan, T.

Montgomery, in Theory and Practice in Distributed Systems, International Workshop, LNCS vol. 938, pp 33-57, septembre 1994.

[Yav95] "A reliable dissemination protocol for interactive collaborative applications", R. Yavatkar, J. Griffieon, M. Sudan, 3rd ACM International Multimedia and Exhibition Conference, pp. 333-344, 1995.

Chapitre 3

Évaluations des mécanismes de fiabilité des protocoles de transport multipoints pour communications satellite

INTRODUCTION

Dans le chapitre précédent, la présentation des protocoles de transport multipoints a mis en évidence l'existence de plusieurs techniques de fiabilisation complémentaires. En particulier, les différents agencements des mécanismes de codage (FEC) et de reprise sur erreurs (ARQ) ont été évoqués. Au-delà des arguments qualitatifs avancés, il reste à identifier précisément ceux d'entre eux qui sont les plus appropriés aux communications multipoints par satellite. Pour une mise en œuvre, il convient également d'étudier leurs paramètres, notamment pour le codage correcteur. En effet, un paramétrage approprié est un critère essentiel à un fonctionnement efficace. Mal dimensionné, le codage peut devenir inefficace ou trop gourmand en bande passante. D'autre part, nous avons mentionné que les codes basés sur les LDPC introduisent un manque d'efficacité, dans la mesure où les récepteurs doivent recevoir une quantité d'information supplémentaire par rapport à la taille du message applicatif original, soumis en tant qu'unité de données au service de transport. Identifier le codage optimal implique d'en évaluer l'impact sur le fonctionnement global du protocole.

L'objectif de ce chapitre est de fournir une méthodologie permettant de répondre aux différentes questions soulevées. Grâce à une modélisation des mécanismes étudiés, nous pourrons évaluer quantitativement les performances des protocoles de transport multipoints les plus importantes. Plutôt orientés vers le service de fiabilité totale, certains aspects de l'évaluation pourront également être étendus à d'autres niveaux de service. Ces évaluations seront réalisées dans le cadre d'hypothèses réalistes de réseaux par satellite.

Comme il a été entrepris dans d'autres études sur le transport multipoints, de nombreux paramètres peuvent être évalués pour caractériser le fonctionnement du protocole de transport. Parmi eux, nous ne retiendrons que ceux qui contribuent directement à l'efficacité du service de transport :

- la *quantité de données* transmises sur la voie aller de la communication multipoints. Elle sera exprimée en nombre de paquets ;

- le *délai moyen d'attente* des récepteurs avant que le message ne soit disponible au niveau de l'application réceptrice.

Le premier critère mesure l'utilisation de la ressource satellitaire nécessaire pour parvenir à l'objectif de fiabilité demandé (fiabilité statistique ou fiabilité totale selon les cas). Le second affecte la qualité de service perçue par l'utilisateur car elle conditionne le temps d'accès au service.

La suite de ce chapitre est structurée en deux parties principales. Dans la première, nous évaluerons la quantité de données transmise au travers d'un facteur d'inefficacité dépendant de la nature de la fiabilisation. Les évaluations seront normalement faites pour des codes MDS sur $GF(2^8)$, voire sur des codes MDS sur $GF(2^{16})$ pour des valeurs de k limitées (inférieures ou égales à 1000). Les raisons de ce choix apparaîtront dans la seconde partie du chapitre, dans laquelle les délais d'attente pour un service de transport à fiabilité totale seront évalués. Nous terminerons ce chapitre en rappelant les principaux résultats obtenus à partir de cette analyse et en concluant sur les différents apports et faiblesses de deux types de codage possibles, MDS ou LDPC.

3.1 ÉVALUATION DE LA QUANTITÉ DE DONNÉES TRANSMISE PAR SESSION

Obtenir la quantité de données transmise en fonction de l'ensemble des paramètres et des choix de mises en œuvre est assez délicat. L'approche choisie dans cette partie est fondée sur une analyse probabiliste dans laquelle les pertes seront considérées comme des évènements indépendants du temps. La formulation s'inspire des travaux de Nonnenmacher [Non97]. Nous nous intéresserons d'abord à un modèle de pertes indépendantes des récepteurs, modèle que nous étendrons ensuite au cas d'un groupe hétérogène. Cette approche analytique sera toutefois remise en cause à partir du chapitre suivant, afin de mieux modéliser les pertes sur la liaison satellite.

En appelant N le nombre moyen de paquets transmis au cours de la session multipoints, N_0 le nombre de paquets constituant le message et f le facteur d'inefficacité du codage (supérieur à 1 pour les codes de type LDPC, égal à 1 pour les codes MDS), $\eta_{fiabilisation}$ un facteur d'inefficacité que nous allons étudier, relatif à l'agencement des mécanismes de fiabilité et aux paramètres (n, k) du code, N peut s'écrire sous la forme :

$$N = f.\eta_{fiabilisation} N_0 \tag{3.1}$$

L'objectif de cette partie est la détermination analytique du facteur $\eta_{fiabilisation}$. Ce facteur est dépendant des paramètres du codage, de la politique de fiabilisation employée, et de la taille du groupe. L'étude du facteur f, propre aux codages LDPC, sort du contexte de ce travail. Dépendant fortement de la mise en oeuvre, du modèle d'erreurs choisi, des taux de pertes et d'autres facteurs, sa détermination demeure complexe. Quelques études ont pu en faire des mesures sur plusieurs mises en œuvre, notamment [Pla03] [Roc04]. Pour donner un ordre de grandeur ici, f se situe généralement entre 1.05 et 1.15 pour des valeurs de k inférieures ou égales à 10000 avec des taux de codage de 2/3.

Le chapitre précédant (2.4.4) présentait les trois principales méthodes de fiabilisation *a priori* possibles pour compléter un service réseau multipoints non fiable, tel que le service *best effort* d'Internet : FEC sans ARQ, hybride ARQ de type I et hybride ARQ de type II. Toutefois, pour introduire les calculs, nous commençons par la méthode ARQ plus simple à évaluer. Dans la suite de ce chapitre, tous les paquets qui atteignent le niveau du protocole de transport seront supposés être exempts d'erreurs. Enfin, la signalisation ajoutée pour gérer les mécanismes de fiabilisation (ARQ ou FEC) est supposée être transportée dans l'en-tête des unités de données du protocole. Quelques octets (3 ou 4) par unité de données suffisent pour cette signalisation. Nous pourrons donc la négliger dans les calculs.

3.1.1 *Hypothèse d'erreurs uniformes*

3.1.1.1 Performances d'ARQ

Au niveau du récepteur, ARQ détecte tout paquet perdu par une rupture de séquence. Ainsi, dans le cadre de transmission multipoints, la transmission se poursuit tant que tous les membres du groupe n'ont pas reçu au moins une copie de chaque paquet.

3.1.1.1.1 Formulation théorique dans le cas général

Soit x_j la variable aléatoire représentant le nombre de copies (original compris) d'un paquet donné émis par la source, nécessaire pour qu'un récepteur *j* appartenant au groupe le reçoive, et X la variable aléatoire représentant le nombre de copies à émettre pour l'ensemble du groupe. Soit p_j la probabilité de perte de paquet (appelé aussi *Packet Loss Rate*, PLR) pour le récepteur *j*.

La probabilité pour que le récepteur *j* ait reçu au moins une fois le paquet après *i* tentatives est donnée par :

$$P(x_j \le i) = (1 - p_j^i) \tag{3.2}$$

Dans le groupe, la probabilité pour que *r* récepteurs aient reçu au moins une copie sans erreur du paquet après *i* transmissions est :

$$P(X \le i) = \prod_{j=1}^{r} P(x_j \le i) \tag{3.3}$$

De plus, on a :

$$P(X = i) = P(X \le i) - P(X \le i-1) \quad pour \ \ i \ne 1$$

et : (3.4)

$$P(X = 1) = P(X \le 1) = \prod_{j=1}^{r} P(x_j \le 1)$$

On cherche la valeur moyenne du nombre de copies par paquet à émettre, E(X), pour que l'ensemble du groupe le reçoive. Ce nombre est par définition égal à $\eta_{fiabilisation}$. Il est donné par :

$$E(X) = \lim_{m \to \infty} S_m \tag{3.5}$$

Avec S_m de la forme :

$$S_m = \sum_{i=1}^{m} i.P(X = i) = P(X \le 1) + 2.(P(X \le 2) - P(X \le 1)) + ... \\ ... + m.(P(X \le m) - P(X \le m-1)) \tag{3.6}$$

En réarrangeant l'expression précédente, on a :

$$S_m = m * P(X \le m) - \sum_{i=1}^{m} P(X \le i-1) \ car \ P(X \le 0) = 0 \tag{3.7}$$

On obtient :

$$S_m = \sum_{i=1}^{m} \left(P(X \le m) - P(X \le i-1) \right) \tag{3.8}$$

En faisant tendre *m* vers l'infini, et en vertu du théorème d'inversion de la somme et de la limite pour les séries, on obtient, d'après l'équation (3.5) :

$$E(X) = \sum_{i=1}^{\infty} \left(1 - P(X \le i-1) \right) \tag{3.9}$$

En remplaçant P(X≤i-1) par sa valeur donnée par l'équation (3.3), puis grâce à l'équation (3.2), on obtient finalement :

$$E(X) = \eta_{fiabilisation} = \sum_{i=1}^{\infty} \left(1 - \prod_{j=1}^{r} (1 - p_j^{i-1}) \right) \tag{3.10}$$

3.1.1.1.2 Probabilités indépendantes du récepteur

Si l'on considère les p_j indépendantes du récepteur et égales à p, l'équation précédente se simplifie en :

$$E(X) = \sum_{i=1}^{\infty} \left(1 - \left((1 - p^{i-1})^r \right) \right) \tag{3.11}$$

Les variations de E(X) sont représentées sur la Figure 3-1 en fonction de la taille du groupe, *r*, pour plusieurs taux de pertes de paquet (PLR=10^{-1}, 10^{-2}, 10^{-3}). Sur la Figure 3-2, les mêmes courbes sont tracées en fonction du PLR pour plusieurs tailles de groupe(*r*=100, 1000 et 10000). On peut voir dans ces courbes que le nombre de récepteurs dans le groupe joue, comme prévu, un rôle de première importance dans les performances, même pour un faible PLR de 10^{-2}. Par exemple, avec ce taux de pertes, et pour une transmission vers 1000 récepteurs, assez plausible dans un environnement satellite, chaque paquet doit être en moyenne transmis environ 2 fois pour satisfaire l'ensemble du groupe.

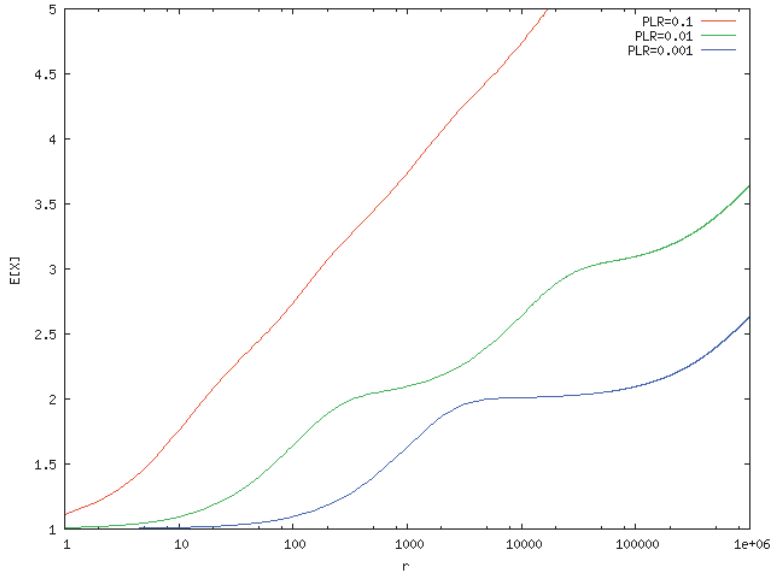

Figure 3-1 : Performances de ARQ en fonction de la taille r du groupe pour PLR=10%, 1%, 0.1 %

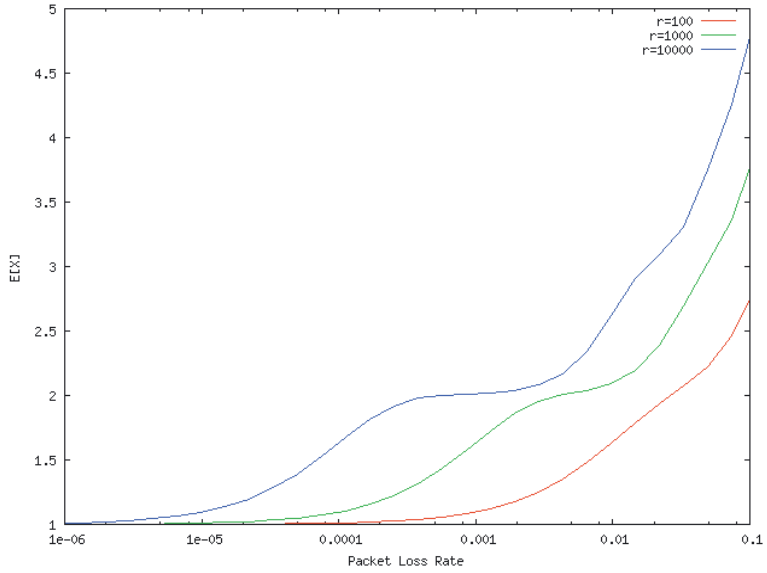

Figure 3-2 : Performances de ARQ en fonction du PLR pour r=100, 1000,10000

3.1.1.2 Performances de FEC sans ARQ

3.1.1.2.1 Formulation théorique

Quand la voie de retour n'est pas disponible ou que l'on ne souhaite pas l'utiliser, le seul moyen de fiabilisation est le codage correcteur d'erreurs. Dans ce cas, il n'y a aucune certitude que les tous les récepteurs aient acquis une copie du paquet, et le protocole ne peut assurer qu'une fiabilité statistique. Pour l'évaluer, Nonnenmacher *et al.* [Non99] définissent une probabilité d'erreur résiduelle *P*, comme étant la probabilité pour qu'au moins un récepteur dans le groupe ne puisse décoder un bloc.

On considère que le fonctionnement de la transmission est le suivant. En plus d'encoder les données, on choisit de transmettre chacun des blocs de n paquets i fois. L'évaluation de la performance dans ce type de transmission commence par celle du PLR moyen d'un récepteur après le décodage FEC. On peut montrer qu'avec un code MDS (n, k), celui-ci, appelé q_j, est donné par [Non97] :

$$q_j(n,k,p) = p_j.\left(1 - \sum_{i=0}^{n-k-1} C_{n-1}^{i}.p_j^i(1-p_j)^{n-i-1}\right) \tag{3.12}$$

On cherche ensuite la probabilité que les r récepteurs aient reçu au moins une copie d'un paquet aléatoire après i transmissions. En remplaçant p_j par q_j dans les équations (3.2) puis (3.1), on obtient [Non97] :

$$P(X \le i) = \prod_{j=1}^{r}\left(1 - q_j^i(n,k,p)\right) \tag{3.13}$$

On cherche à estimer le nombre de transmissions minimal à prévoir afin que la probabilité d'erreur résiduelle soit inférieure à un certain objectif, α. Pour cela, il suffit de chercher la plus petite valeur de i, appelée i_{min}, satisfaisant la contrainte :

$$P(X \le i) \ge 1 - \alpha \tag{3.14}$$

Si l'on peut considérer que les q_j sont indépendants, il suffit d'isoler i dans les équations précédentes pour obtenir i_{min}, alors donné par :

$$i_{\min} = \left\lceil \frac{\ln\left(1-(1-\alpha)^{1/r}\right)}{\ln\left(q(n,k,p)\right)} \right\rceil \tag{3.15}$$

Or, l'utilisation du codage FEC introduit un accroissement de l'utilisation de la bande dans le rapport n/k. Le nombre X de paquets émis après i transmissions du même bloc est égal à $i.n/k$. Le nombre minimal X_{min} de paquets à transmettre pour respecter la contrainte fixée est donc :

$$X_{\min} = \frac{n}{k}\left\lceil \frac{\ln(1-(1-\alpha)^{1/r})}{\ln(q(n,k,p))} \right\rceil \tag{3.16}$$

3.1.1.2.2 Impact du couple (n, k) sur les performances et considérations pratiques

Lorsque k est choisi petit (cf. Figure 3-3, k=16), et pour un objectif de probabilité d'erreur résiduelle égal à 10^{-4}, le codage est moins efficace que l'ARQ, à moins d'avoir recours à un taux de codage faible (typiquement inférieur ou égal à ½).

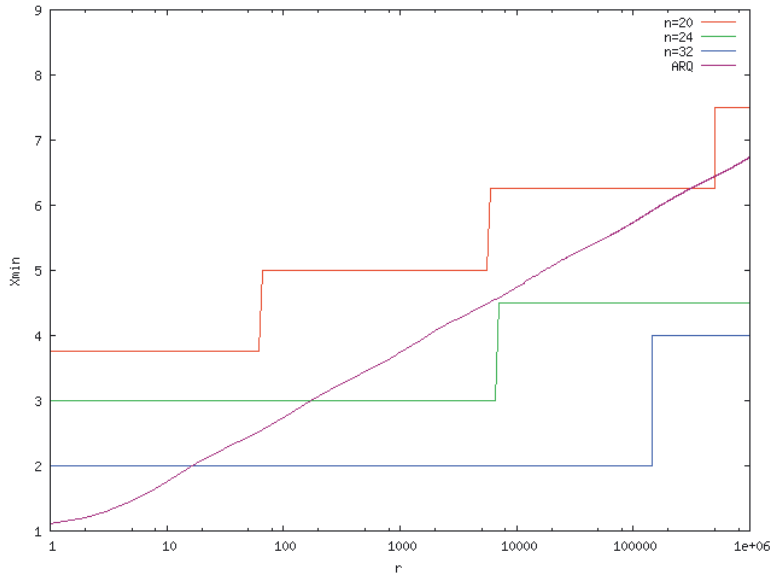

Figure 3-3 : Influence de n avec fiabilisation par FEC sans ARQ, pour k=16, $\alpha=10^{-4}$, PLR=10%

Si l'on fixe k à 64 (Figure 3-4), les résultats obtenus sont meilleurs, pour autant que h soit suffisamment grand. Par exemple, avec le code (n=96, k=64) de taux ½, X_{min} reste constant jusqu'à un million de récepteur, ce qui correspond à i_{min}=1. Dans ces conditions, une seule transmission par paquet en moyenne est donc suffisante quelle que soit la taille du groupe.

Cette analyse succincte conforte l'intuition que l'on pouvait avoir quant au domaine (n, k) du code où les performances sont les plus intéressantes : pour une valeur de k fixée, les performances augmentent avec n. De même, pour un taux de code constant, les performances sont améliorées lorsque la longueur (et donc la dimension) des codes augmentent. Par exemple, pour un taux de code égal à 2/3, le code (n=96, k=64) obtient de meilleures performances (Figure 3-4) que le code (n=24, k=16) (Figure 3-3).

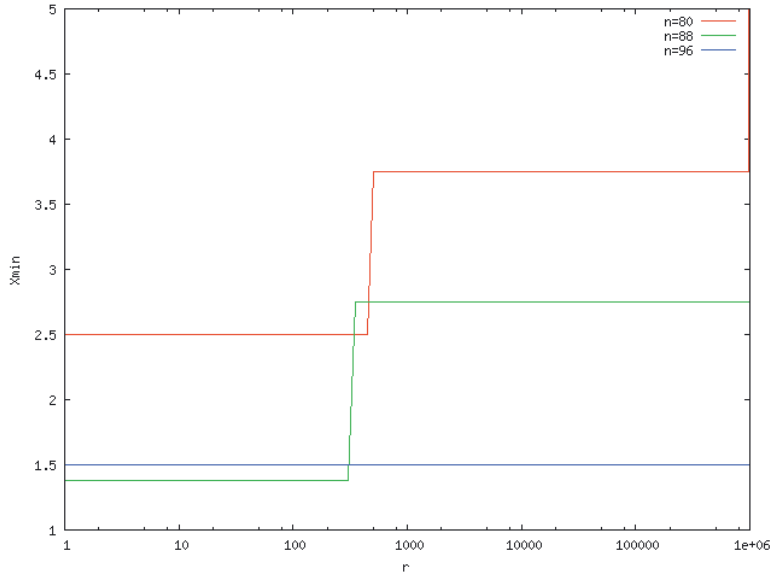

Figure 3-4 : Influence de n avec fiabilisation par FEC sans ARQ pour k=64, $\alpha=10^{-4}$, PLR=10%

En pratique, on ne peut pas augmenter *k* et *n* indéfiniment. Pour les codes sur GF(2^8) (resp. sur GF(2^{16})), *k* est limité à *n*, lui-même limité à 255 (resp. 65535). L'estimation des performances du code, données par l'équation (3.16), est difficile à obtenir quand *n* est supérieur ou égal à 1000, en raison du faible ordre de grandeur des différents termes de la somme de l'équation (3.12). Nous avons cherché à obtenir, pour plusieurs valeurs de *n*, la valeur de *k* optimisant les performances de la transmission, en considérant i_{min}=1 et une taille de groupe *r* inférieure ou égale à un million de récepteurs. La Figure 3-5 indique les valeurs des meilleurs taux de code possibles pour PLR=10^{-1}, $\alpha=10^{-4}$, ainsi que l'évolution graphique du rapport de code optimal en fonction de *n*.

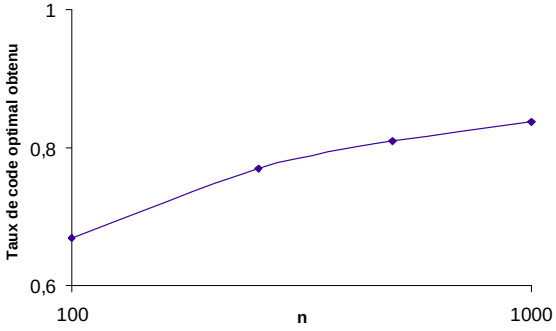

Figure 3-5 : Évolution du taux de codage optimisé en fonction de n pour un million de récepteurs avec PLR=10%, $\alpha=10^{-4}$

3.1.1.2.3 Impact de α

Quand le rapport α/r est petit devant 1 (ce qui est toujours le cas dans notre domaine d'étude), et pour un facteur λ quelconque, on obtient par équivalence de (3.15) :

$$X_{\min}(r, \lambda.\alpha) \cong X_{\min}(r/\lambda, \alpha) \qquad (3.17)$$

Aussi, dans la représentation de la courbe $X_{min}=f(r)$ à α fixé, si l'on utilise une échelle logarithmique en abscisses, une multiplication de α par λ doit entraîner une translation

horizontale de la courbe d'un facteur λ équivalent. On peut le vérifier sur la Figure 3-6 pour le code (n=255, k=196), avec λ=10 et λ=100.

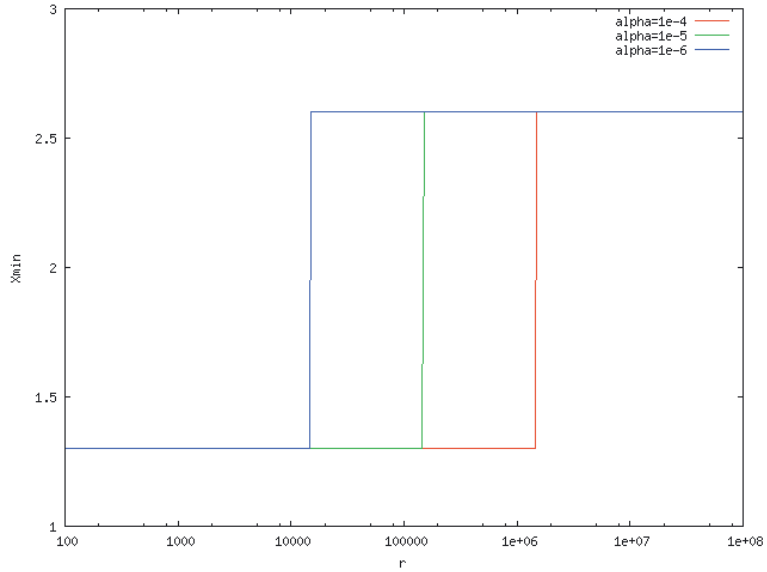

Figure 3-6 : Impact de α sur le code (n=255, k=196) pour PLR= 10%

3.1.1.3 Hybride ARQ de type I

Le principe d'Hybride ARQ de type I [Lin83] est de coupler les deux techniques de fiabilisation, FEC et ARQ de manière à ce qu'elles soient indépendantes l'une de l'autre (cf. 2.3.2.3.1). Chaque ensemble de k paquets d'information à transmettre est toujours codé en n paquets de parité qui sont tous transmis. Lorsque le décodage est impossible pour un récepteur (ayant perdu plus de h paquets dans le même bloc FEC), un mécanisme de reprise sur erreur classique assure la retransmission. Comme pour les autres paquets, les paquets de réparation, transmis sur demande, sont protégés par le codage FEC. Avec ce schéma de fiabilisation, il devient alors possible d'assurer la fiabilité totale sur tout le groupe, c'est-à-dire d'obtenir strictement α=0.

Pour évaluer les performances de ce mécanisme, il suffit d'entreprendre la même démarche que pour le calcul avec ARQ, en remplaçant la probabilité de perte p_j de ARQ (3.10) par celle obtenue après codage, q_j, donnée en (3.12), et en multipliant la moyenne E(X) par n/k pour prendre en compte l'utilisation supplémentaire de bande par le codage. Pour des pertes indépendantes des récepteurs on obtient [Non97] :

$$E(X) = \eta_{fiabilisation} = \frac{n}{k}.\sum_{i=1}^{\infty} 1 - \left((1-p^{i-1}).\left(1 - \sum_{i=0}^{n-k-1} C_{n-1}^{i}.p^{i}(1-p)^{n-i-1} \right)^{i-1} \right)^{r} \qquad (3.18)$$

La Figure 3-7 représente l'évolution de E(X) en fonction de r avec n=255 pour des valeurs de k égales à 200, 210 et 220. On y constate que les performances sont assez sensibles au choix de k. Pour une taille de groupe de plus de 10000 récepteurs, k égal à 200 semble le choix le plus adéquat.

Avec n égal à 1000 (Figure 3-8), la valeur optimale de k est 850, ce qui est proche de la valeur optimale trouvée sans reprise sur erreur. L'efficacité de la transmission est alors augmentée par rapport au cas où n=255.

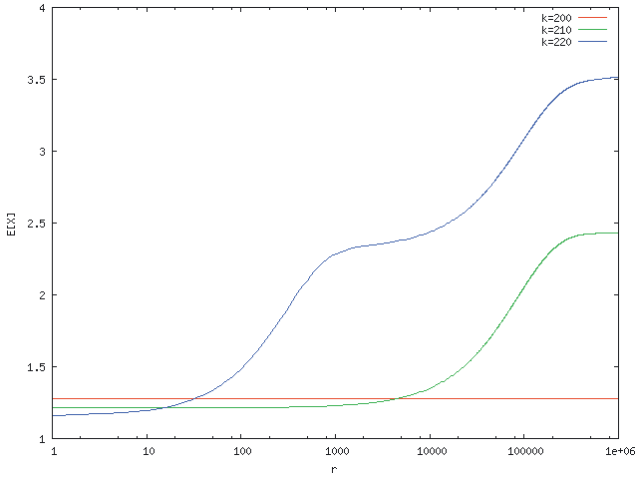

Figure 3-7 : Influence de k sur Hybride ARQ de type I avec n=255, PLR=10% en fonction de r

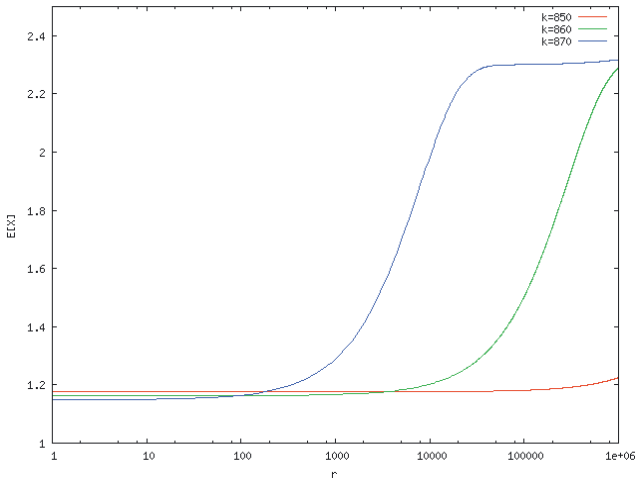

Figure 3-8 : Influence de k sur Hybride ARQ de type I avec n=1000, PLR=10%

3.1.1.4 Hybride ARQ de type II

Avec la technique Hybride ARQ de type II [Lin83], FEC et ARQ sont intégrés au sein d'un même processus (cf. 2.3.2.3.2). Contrairement à Hybride ARQ de type I, les données sont encodées par le codage (*n, k*) mais les *h* paquets de parités obtenus par bloc FEC ne sont transmis progressivement, que sur demande des récepteurs, lorsque des pertes ont lieu. Tout se passe donc comme si le taux de codage était ajusté dynamiquement par la source en tenant compte des taux de pertes des récepteurs, et en transmettant progressivement les *h* symboles poinçonnés.

De plus, on considère qu'une partie de la redondance peut être transmise de manière proactive pour réduire les temps de réception au détriment de l'utilisation de la bande passante dans le réseau dans certaines conditions. Soit *a* le nombre de paquets de parité par bloc transmis proactivement ($a<h$). Nonnenmacher *et al.* [Non97] ont montré que si *n* est infini (ce qui revient à dire que l'on peut générer autant de paquets de parité que nécessaire, sur demande), le nombre moyen de paquets transmis, E(X), serait donné par :

$$E(X) = \eta_{fiabilisation} = \frac{E(L) + k + a}{k} \tag{3.19}$$

Dans l'équation (3.17), E(L) représente l'espérance de la variable aléatoire L, qui est le nombre de paquets retransmis par bloc FEC au-delà des $k+a$ premiers paquets. E(L) est donné par [Non97] :

$$E(L) = \sum_{i=0}^{\infty} \left(1 - P(L \le i)\right) \tag{3.20}$$

Si P(l=i) est la probabilité qu'un récepteur ait besoin d'exactement i retransmissions supplémentaires pour parvenir à décoder le bloc et si les p_j sont indépendants du récepteur, on peut écrire :

$$P(L \le i) = \left[\sum_{i=0}^{\infty} P(l = i) \right]^{R} \tag{3.21}$$

La distribution de cette loi de probabilité est donnée par [Non97] :

$$P(l = 0) = \sum_{i=0}^{a} C_{k+a}^{i} . p^{i} . (1 - p)^{k+a-i}$$
$$P(l = m) = C_{k+a+m-1}^{k-1} . p^{m+a} . (1 - p)^{k} \qquad \text{si } m \neq 0 \tag{3.22}$$

Rappelons ici que ces résultats sont *a priori* valables pour n infini. Nous allons montrer dans la suite dans quelle mesure ils peuvent être étendus à n fini.

3.1.1.4.1 Influence de a

Nous examinons en premier lieu l'impact du paramètre a sur les performances d'Hybride ARQ de type II en Figure 3-9, pour k=16.

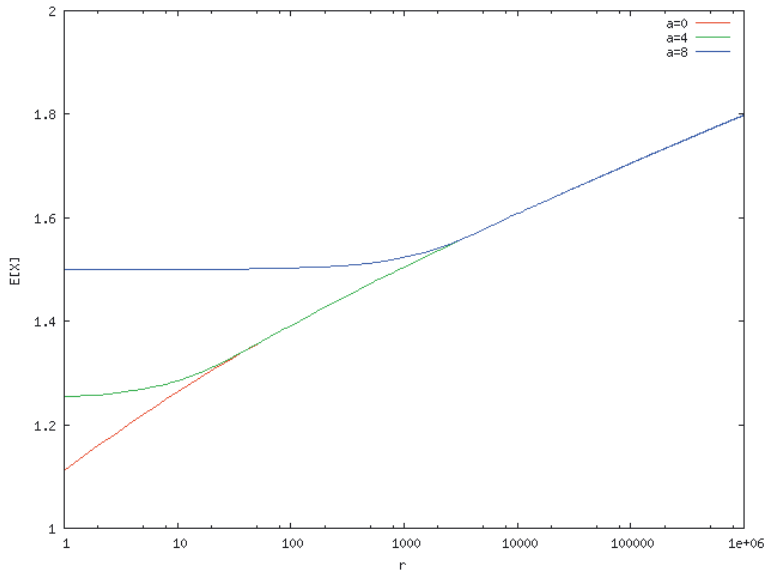

Figure 3-9 : Influence du nombre de paquets de parité par bloc transmis proactivement, a, sur Hybride ARQ de type II pour k=16, n=∞, PLR=10 %, en fonction du nombre de récepteurs, r

Trois conclusions peuvent être tirées de ces courbes. D'abord, la redondance proactive (a=4, a=8) est parfois intéressante, mais pas toujours. En effet on voit que pour des groupes de l'ordre de quelques dizaines à environ 2000 récepteurs, une partie de la bande passante est gâchée car le nombre de paquets de parité transmis pour ces valeurs de a est supérieur au nombre optimal de

paquets, obtenu lorsque *a* est nul. En contrepartie, choisir *a* non nul peut se révéler attractif pour de grands groupes car :

- cela améliore le temps de réception total puisque pour transmettre la même quantité de données avec du codage proactif, moins de cycles de retransmissions sont nécessaires ;
- cela limite la quantité de trafic transitant sur la voie de retour.

Dans la suite de l'étude, nous privilégions la transmission d'un nombre de paquets aussi faible que possible, soit « E(X) minimum ». Nous supposerons donc désormais que *a* est égal à 0.

Le second résultat observé est important. E(X) est toujours inférieur à 2 dans la figure précédente, ce qui signifie qu'en moyenne, on ne transmet jamais autant de paquets de parité (*h*) que de paquets originaux (*k*) par bloc, ce qui est en accord avec l'hypothèse de calcul. Cependant, nous validerons l'hypothèse plus correctement en 3.1.1.4.3.

3.1.1.4.2 Influence de k

La Figure 3-10 confirme qu'il est préférable de choisir *k* aussi élevé possible. Mais d'autre part les implémentations actuelles empêchent d'avoir recours à des valeurs trop élevées, sous peine de ralentir le processus de décodage et donc le temps d'accès au service pour l'utilisateur (cf. 3.2). La Figure 3-11 évalue les performances du mécanisme lorsque *k* varie entre 16 et 1000. Pour cette dernière valeur, les résultats sont obtenus par simulation en raison du problème de calcul évoqué en 3.1.1.2.2. Le principe de cette simulation sera expliqué au paragraphe suivant.

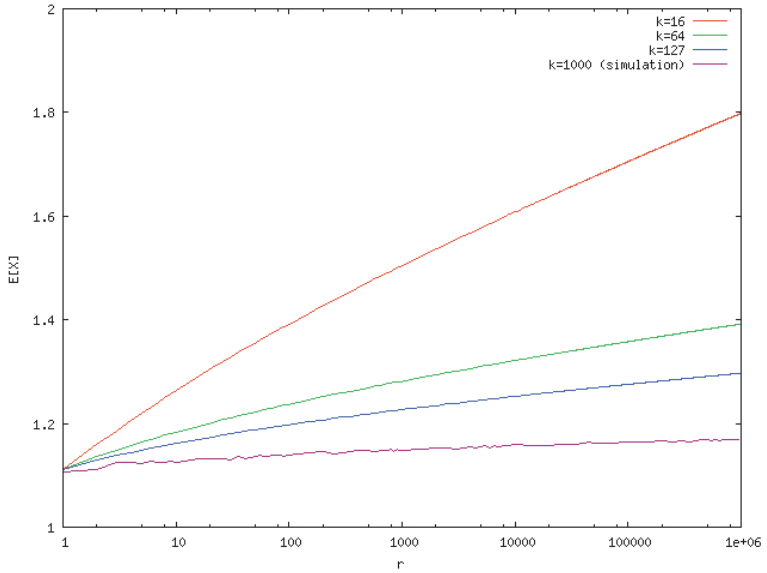

Figure 3-10 : Influence de k sur Hybride ARQ de type II, PLR=10%, en fonction du nombre de récepteurs r

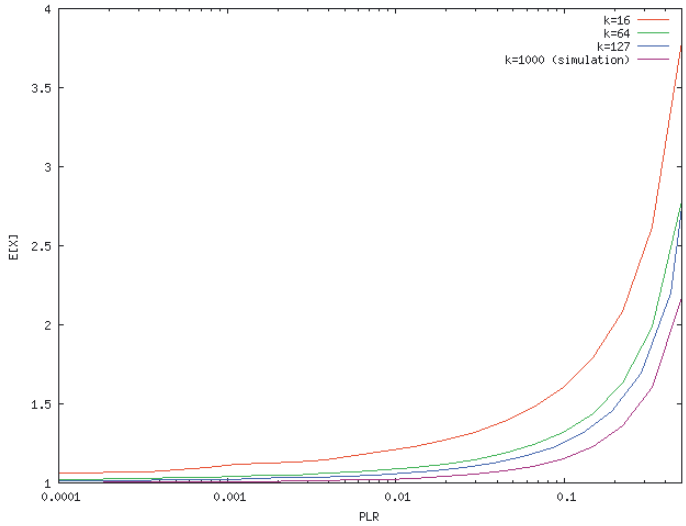

Figure 3-11 : Influence de k sur Hybride ARQ de type II, r=10000, en fonction du PLR

3.1.1.4.3 Validation de l'hypothèse de calcul pour n fini

Sous réserve de certaines conditions que nous allons déterminer, il est possible de montrer que les équations (3.19) à (3.22) restent valables pour *n* fini.

En simulant une transmission Hybride ARQ de type II par l'algorithme de la Figure 3-12 (validé pour $k \leq 127$ avec le modèle théorique donné par les équations (3.19) à (3.22)), le nombre maximum de paquets transmis a été calculé pour *k* égal à 16, sur 1000 itérations. La valeur obtenue est de 33 paquets, ce qui correspond à un facteur E(X) d'environ 2.1 (légèrement supérieur à la valeur de 2, annoncée en 3.1.1.4.1).

```
//En paramètres : k, r, PLR

Pour(j=0;j<r;j++)
  { NombrePaquetsRecus[j]=0 ; }
NombrePaquetsRestants=k ;
TantQue(NombrePaquetsRestants>0)
  {
    NombrePaquetsTransmis+=NombrePaquetsRestants ;
    Pour(j=0;j<r;j++)
       {
          Pour(i=0;i<NombrePaquetsRestants;i++)
              {
                 NbAleatoire=Rand() ; //Nombre aléatoire entre 0 et 1
                 Si (NbAleatoire>PLR)
                     { NbPaquetsRecus[j]++ ; } // le paquet i a été recu
              }                                // par le récepteur j
       }
    NombrePaquetsRestants=0 ;
    Pour(j=0;j<r;j++)
      {
          Si ( NombrePaquetsRestants < ( k – NombrePaquetsRecus[j] )  )
              {  NombrePaquetsRestants= k-NombrePaquetsRecus[j] ; }
      }
  }
Return (NombrePaquetsTransmis) ;
```

Figure 3-12 : Algorithme de simulation de performances pour Hybride ARQ de type II avec n infini

D'autre part, la Figure 3-10 montre que lorsque *k* est supérieur à 16 et *n* infini, E(X) diminue progressivement à *r* donné. On peut en conclure que l'évaluation donnée pour *n* infini reste

valable pour n fini si les deux conditions (suffisantes, mais non nécessaires) sont respectées : k>16 et n≥2.1*k.

Lorsque k est nettement supérieur à 16 (k>127), il est clair que la seconde condition est trop forte car E(X) diminue lorsque k augmente. Par exemple, si on limite n à 255, la plus grande valeur de k obtenue par simulation pour qu'au plus les 255–k paquets de parités soient nécessaires est d'environ 200.

3.1.1.4.4 Conclusion sur l'apport d'Hybride ARQ de type II dans le contexte de transport multipoints pour satellite

Si l'on compare les deux schémas de fiabilisation Hybride ARQ de type I et Hybride ARQ de type II, les évaluations précédentes démontrent la supériorité du second schéma sur le premier dans toutes les situations. On peut s'en convaincre explicitement en fixant k pour les deux mécanismes, et en faisant évoluer n pour Hybride ARQ de type I ; par exemple, dans la Figure 3-13, k a été choisi égal à 127. On pourrait montrer que dans tous les autres cas, une seule valeur de r (dépendant de n) permettrait à Hybride ARQ de type I d'atteindre la valeur optimale donnée par Hybride ARQ de type II. Ici, pour n=160, la valeur optimale d'Hybride ARQ de type I se situe à environ r=200.

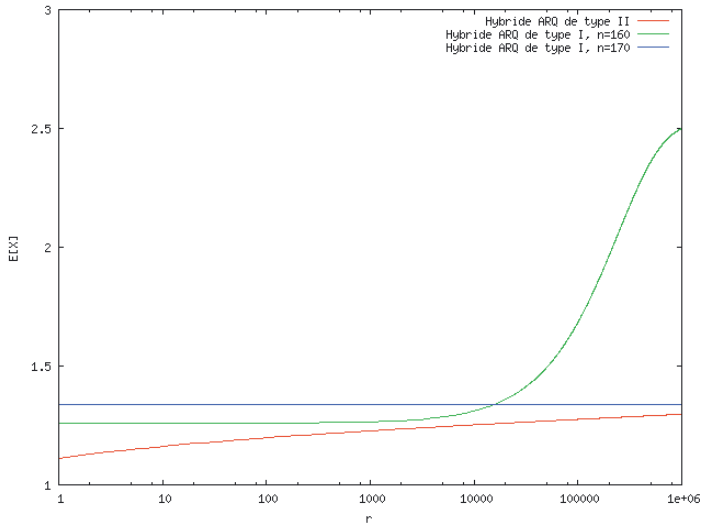

Figure 3-13 : Comparaison entre Hybride ARQ de type I et II pour k=127, PLR=10%

3.1.2 ***Modèles de pertes hétérogènes***

Le modèle de pertes que nous avons considéré précédemment est probablement peu réaliste pour une communication de groupe par satellite car l'hétérogénéité des pertes pour un grand nombre de récepteurs est probable (p.ex. si une cellule de pluie est localisée au-dessus de quelques terminaux satellite). Nous évaluons les performances pour les schémas de fiabilisation par FEC pur et par Hybride ARQ de type II.

Si l'on admet que les pertes restent uniformément réparties, mais qu'elles diffèrent selon le récepteur considéré, il est facile de modifier les équations précédentes si l'on connaît la distribution des PLR pour l'ensemble du groupe. Dans une démarche semblable, Nonnenmacher *et al.* [Non97] modélisent le groupe selon deux sortes de récepteurs. Pour les premiers, ne représentant qu'une petite fraction du groupe que nous appelons υ, le PLR est fixé à 25%. Les autres récepteurs sont caractérisés par un PLR plus faible, égal à 1%.

3.1.2.1 FEC pur

Lorsque n=255, υ=1%, $\alpha=10^{-4}$, et que l'on cherche à minimiser $\eta_{fiabilisation}$ pour n'importe quelle taille de groupe, la meilleure valeur de k obtenue est 150. La Figure 3-14 montre l'évolution de E[X] quand υ varie entre 1 et 100 %.

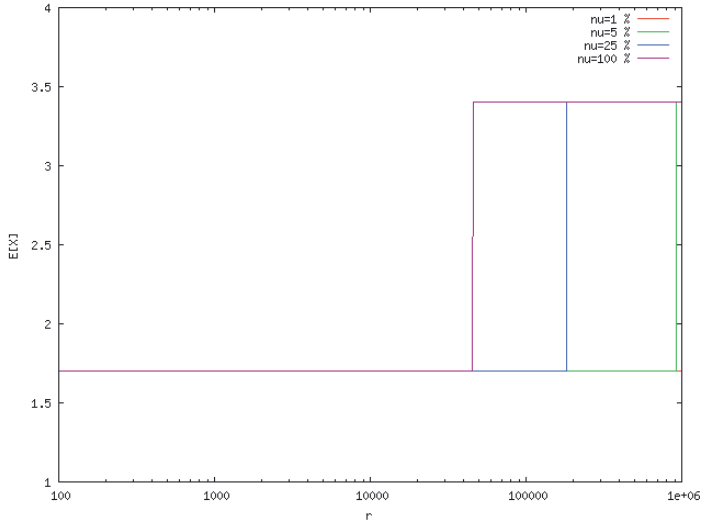

Figure 3-14 : Performances pour FEC pur en environnement hétérogène avec k=150, n=255, $\alpha=10^{-4}$

En examinant les valeurs de r pour lesquelles E(X) bascule (environ 46000, 184000 et 913000), on se rend compte que tout se passe comme si le groupe était seulement constitué de la classe des récepteurs caractérisés par le PLR élevé. On peut expliquer cela en raison de l'importante différence d'ordre de grandeur entre les deux PLR obtenus grâce au codage (environ 2.10^{-9} pour 25% de pertes des paquets dans le réseau et 2.10^{-17} pour 1% de pertes).

3.1.2.2 Hybride ARQ de type II

Dans l'étude [Non97], les performances étaient évaluées avec Hybride ARQ de type II pour k=7. À r fixé, les courbes pour lesquelles υ=1%, 5% et 25% montraient un écart quasiment constant d'environ 0.25. D'autre part, pour υ=25%, et $r=10^6$, E(X) vaut 3.25.

Si l'on s'intéresse aux performances simulées pour k=1000 (cf. Figure 3-15), une amélioration notable peut être remarquée. Si l'on considère que E(X) ne peut pas être inférieur à $[1-Max_{j\in G}(PLR_j)]^{-1}$, ici égal à 1.33 et qui représente la valeur moyenne « minimale » pour qu'un seul des « mauvais » récepteurs puisse acquérir un paquet, le fait d'augmenter la taille du groupe ne conduit qu'à une légère baisse de performances (pour υ=25%, E(X)=1.43 pour un millions de récepteurs). Le facteur d'hétérogénéité n'intervenant que dans une faible proportion (l'écart entre E(X) est réduit à environ 0.1 pour chaque courbe), on peut en conclure la robustesse du schéma Hybride ARQ de type II pour un ensemble hétérogène de récepteurs.

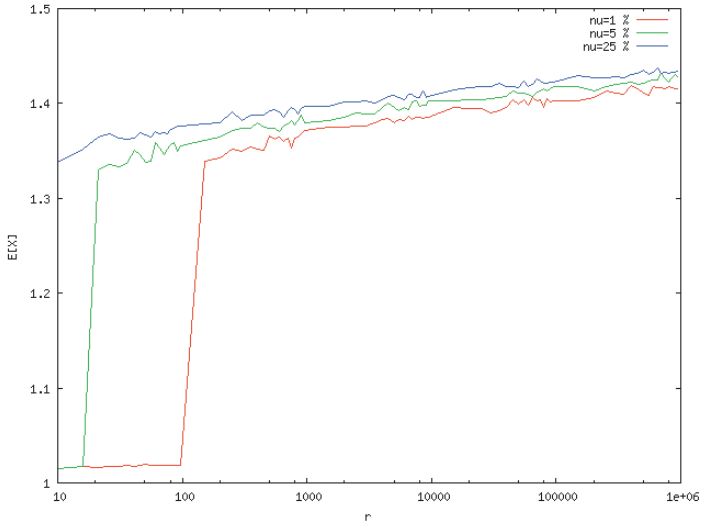

Figure 3-15 : Simulation des performances pour Hybride ARQ de type II en environnement hétérogène, avec k=1000

3.1.3 *Mesures réalisées sur des modèles de pertes issues de la couche physique*

Dans le cadre d'une étude expérimentale menée avec Alcatel Space [Arn03], des performances de protocoles de transports multipoints ont été mesurées. Les protocoles étudiés étaient *Satellite Reliable Multicast Transport Protocol* (Sat-RMTP), encore en cours de développement à cette époque par l'université d'Aberdeen, et UDPush, propriété de la société UDCast. Les protocoles ont été analysés comme deux boîtes noires, c'est-à-dire sans connaissance précise des mécanismes et des paramètres internes qu'ils utilisaient (excepté la nature MDS du codage). Les mesures ont été réalisées sur la plate-forme d'émulation du système GEOCAST, GENEME. Chaque terminal satellite étant émulé par un PC dédié dans la plate-forme, le nombre de récepteurs dans le groupe multipoints était limité (ici 2 récepteurs, chacun étant connecté à un terminal satellite différent).

Des modèles du canal de transmission, de mises en œuvre de FMT et de la couche physique ont été fournis par l'ONERA pour le projet. Ces modèles donnaient l'évolution temporelle du rapport E_b/N_0 mesurés à partir d'une liaison satellite DVB-S réelle, en bande Ka, pour différents évènements météorologiques. L'intégration de ces modèles dans l'émulateur s'est faite de manière simple sur le service réseau : lorsque le rapport E_b/N_0 était inférieur à une valeur seuil (ici 5.5 dB, correspondant au BER de 10^{-10} visé par GEOCAST), la liaison modélisée « perdait » tous les paquets transmis. Sinon, la liaison était considérée comme idéale, et aucun paquet n'était perdu. Une telle modélisation se justifie en raison de la performance des systèmes de codages correcteurs d'erreurs de la couche physique (cf. chapitre 4).

Plusieurs évènements météorologiques ont ainsi été modélisés (Figure 3-16). Sur la figure, l'état « haut » (*fading*) correspond à la perte de paquets, c'est-à-dire à une interruption momentanée du service de la liaison. L'état « bas » correspond lui à une période de transmission sans pertes. Les durées des *fading*s les plus importantes ont été inscrites au-dessus des courbes.

Parmi les différents évènements, seul l'un d'entre eux (courbe bleue sur la figure) a effectivement servi aux mesures de performances avec pertes. Cet évènement correspond physiquement à une liaison montante en ciel clair et une liaison descendante perturbée par une pluie modérée. Les autres évènements étaient peu intéressants pour des mesures, car trop peu d'erreurs se produisaient pour avoir un effet significatif sur les performances, ou au contraire, trop d'erreurs se manifestaient, empêchant alors la réussite de la transmission (après une période sans réception de données, les récepteurs se désabonnaient du groupe IP Multicast).

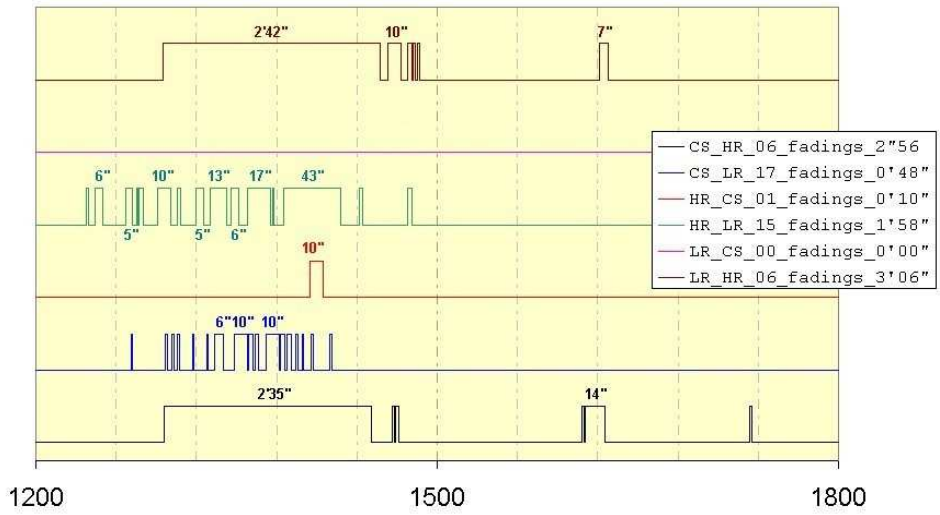

Figure 3-16 : Chronogrammes (en secondes) des pertes de paquets représentatifs d'évènements météorologiques sur des liaisons DVB-S en bande Ka

Les résultats des mesures sont montrés dans le Tableau 3-1, pour une transmission d'un objet d'environ 52 Mo. La vitesse d'émission, réglée au niveau du transport, était de 256 kbits/s. La durée totale de la session est d'environ 1700 secondes, soit près de 3 fois plus que sur le graphique de la Figure 3-16. Le profil de pertes est appliqué quelques secondes après le début de la transmission, avec un décalage de 10 secondes entre les deux terminaux satellite émulés, pour éviter une statistique de pertes identique.

En raison de la taille variable ou inaccessible des en-têtes des protocoles et de leurs unités de données, la mesure n'est pas relative au rapport entre le nombre d'unités de données transmises et le nombre d'unités de données composant l'objet, mais au rapport et la quantité de données reçues par l'interface UDP des récepteurs et la taille de l'objet. La mesure effectuée n'est donc pas rigoureusement conforme à la définition du paramètre $\eta_{fiabilisation}$ utilisée précédemment, mais elle permet néanmoins de s'en faire une idée sur deux implémentations réelles.

	Sat-RMTP	**UDPush**
Liaisons sans erreurs	1.24	1.04
Liaisons avec erreurs	1.27	1.08

Tableau 3-1 : Mesures du paramètre $\eta_{fiabilisation}$ réalisées sur deux protocoles de transmissions multipoints en environnement satellite émulé pour deux récepteurs

Concernant Sat-RMTP, une quantité importante de données (24%) est transmise en plus de la taille de l'objet transféré, même en l'absence d'erreurs. Ceci est probablement dû à l'utilisation de codage FEC proactif, alors impossible à paramétrer avec l'implémentation du protocole testée. Pour UDPush, une légère quantité de données supplémentaire est transmise (4%) ; il est vraisemblable que celle-ci corresponde en majeure partie au décompte d'une signalisation des protocoles du plan utilisateur. En cas d'erreurs, l'accroissement de la quantité de données transmise est d'environ 3 et 4 % pour les deux protocoles.

Au-delà des résultats obtenus avec les limitations évoquées, l'expérience réalisée ici soulève deux questions. La première, d'ordre pratique, concerne la définition de la « meilleure » métrique à mesurer : faut-il, ou non, inclure dans la mesure les données de signalisation, et avec quels outils la réaliser ? Il est évidemment impossible de donner ici une réponse à cette question, car elle dépend du contexte précis. La seconde question est plutôt une problématique ; elle concerne une

méthode de mesures prenant en compte le facteur d'échelle. À l'heure actuelle, seules des approches basées sur simulations pourraient y parvenir, mais elles nécessiteraient l'utilisation de modèles précis. Une solution à étudier pourrait être la mise en place d'un couplage entre émulation et simulation, permettant à la fois une modélisation réaliste du service par satellite, ainsi qu'une taille de groupe élevée pour l'évaluation de la robustesse au facteur d'échelle.

3.2 MESURE DES DÉLAIS D'ATTENTE MOYEN DES RÉCEPTEURS POUR UN SERVICE DE TRANSPORT À FIABILITÉ TOTALE

Dans cette partie, nous cherchons à déterminer la durée moyenne d'attente d'un récepteur, T, correspondant au délai qu'il devra attendre avant que le message atteigne le niveau applicatif. La durée T dépend de deux processus : le processus de réception, contraint par le débit de réception moyen des données, et le processus de décodage, contraint par sa rapidité d'exécution sur la station réceptrice.

Nous introduisons un nouveau paramètre d'inefficacité, $\eta_{décodage}$, représentant le rapport entre le temps mis pour que le message atteigne le niveau applicatif du récepteur et le temps optimal de réception du message pour un groupe d'un seul récepteur, T_0. Nous supposerons alors que l'on peut écrire[4] :

$$T = f.\eta_{fiabilisation}\eta_{décodage}.T_0 \tag{3.24}$$

La suite de l'étude consiste à évaluer le facteur $\eta_{décodage}$. Pour le déterminer, nous commençons par étudier le temps de décodage d'un bloc FEC sur plusieurs mises en œuvre de codes MDS et LDPC. Dans un second temps, un choix de mise en œuvre concernant le début de décodage sera introduit afin de parvenir à l'expression recherchée de $\eta_{décodage}$.

3.2.1 *Évaluation du temps de décodage par bloc*

3.2.1.1 Codes MDS

Lorsqu'ils sont mis en œuvre au niveau logiciel, les algorithmes de codages/décodages RS, tels que celui présenté en Figure 2-1, ne sont pas performants en terme de rapidité d'exécution. Des variantes sont généralement préférées, faisant intervenir des calculs matriciels. Dès lors, la plupart des opérations se résument en multiplications matricielles rapides, avec en plus une inversion pour le décodage. Cette dernière opération étant coûteuse en temps de calcul, la matrice génératrice du code est choisie pour être facilement inversible. Deux types de matrices conviennent particulièrement : les matrices de Vandermonde [Riz97] et les matrices de Cauchy [Blo95].

Après la réception des symboles d'un bloc, le décodage est alors réalisé en deux étapes : l'inversion d'une sous-matrice (k, k) issue de la matrice génératrice du code, et la multiplication de cette matrice avec un vecteur formé des symboles reçus.

En appelant $t_{inv}(k)$ est le temps d'inversion matricielle et $t_m(k)$ le temps de multiplication matrice-vecteur pour un bloc, le temps de décodage total par bloc, $t_d(k)$, est donné par :

$$t_d(k) = t_{inv}(k) + t_m(k) \tag{3.25}$$

Il convient de noter ici que les temps d'inversions et de multiplications dépendent non seulement du paramètre k, mais aussi de la nature du corps sur lequel les opérations sont faites ($GF(2^8)$ ou $GF(2^{16})$), et de la longueur L des paquets servant au codage.

[4] Nous sommes conscients que cette écriture reste une approximation, mais nous la jugeons suffisante pour fournir l'estimation de la durée moyenne des transferts. En tout état de cause, nous négligeons des temps additionnels, par exemple s'ils sont dus à des périodes sans émission de données à la fin de la session.

3.2.1.2 Codes LDPC

Pour les codes LDPC, les opérations sont faites sur le corps GF(2), ce qui permet au décodage d'être plus rapide. Le décodage d'un bloc est réalisé par multiplication des symboles reçus avec la matrice de parité du code. On peut alors écrire :

$$t_d(k) = t_m(k) \tag{3.26}$$

3.2.1.3 Mesures expérimentales de $t_d(k)$

Plusieurs implémentations logicielles de codes ont été réalisées ou adaptées par Jérôme Fimes dans le cadre de son travail de thèse à l'ENSICA. Il a pu mener une campagne de mesures afin de nous fournir, pour cette étude, les mesures expérimentales du paramètre $t_d(k)$. Les mesures ont été effectuées pour :

- un nouveau code développé actuellement à l'ENSICA, appelé code sur les entiers, construit sur GF(2^{16}+1) [Dai04] ;
- une adaptation des codes basés sur les matrices de Cauchy, disponible sur le site de M. Luby [LUB] ;
- une adaptation des codes basés sur les matrices de Vandermonde (VDM), disponible sur le site de L. Rizzo [RIZ] ;
- une implémentation propriétaire de codes LDPC.

La machine de test utilisée pendant cette campagne était un IBM T30, avec processeur Intel Pentium 4 Mobility, 2 GHz, 512 Mo de RAM. Le Tableau 3-2 indique les valeurs de $t_d(k)$ obtenues à taux de codage constant ($\rho = k/n = 2/3$), avec une longueur de datagramme de 1500 octets. Les valeurs ont été obtenues par moyenne sur 10 mesures différentes. Pour garantir l'équité entre les différentes mesures pour chaque valeur de k, la même proportion entre paquets systématiques et paquets de parités était utilisée par les décodeurs.

k	Entiers GF(2^{16}+1)	Cauchy GF(2^{16})	VDM GF(2^{16})	Cauchy GF(2^8)	VDM GF(2^8)	LDPC GF(2)
16	$3.30.10^{-4}$	$3.54.10^{-4}$	$5.06.10^{-4}$	$2.24.10^{-4}$	$1.42.10^{-4}$	$4.26.10^{-5}$
32	$1.45.10^{-3}$	$1.77.10^{-3}$	$2.98.10^{-3}$	$4.35.10^{-4}$	$2.93.10^{-4}$	$6.97.10^{-5}$
64	$5.81.10^{-3}$	$7.87.10^{-3}$	$1.43.10^{-2}$	$2.78.10^{-3}$	$2.11.10^{-3}$	$5.99.10^{-4}$
128	$2.17.10^{-2}$	$3.07.10^{-2}$	$6.72.10^{-2}$	$1.16.10^{-2}$	$1.15.10^{-2}$	$1.36.10^{-3}$
256	$7.81.10^{-2}$	$1.20.10^{-1}$	$3.42.10^{-1}$	N/A	N/A	$1.29.10^{-3}$
512	$3.46.10^{-1}$	$5.35.10^{-1}$	8.83	N/A	N/A	$4.39.10^{-3}$
1024	1.43	2.22	69.9	N/A	N/A	$8.83.10^{-3}$
2048	5.41	8.42	N/A	N/A	N/A	$1.89.10^{-2}$
4096	21.4	34	N/A	N/A	N/A	$3.49.10^{-2}$

Tableau 3-2 : Mesures expérimentales de $t_d(k)$ en secondes pour L=1500 octets, pour plusieurs types de codes

La complexité optimale théorique de décodage des codes MDS est une fonction en $O(k^2)$. Cette dépendance quadratique de t_d en fonction de k est facilement observable, en particulier sur les trois premiers codes du tableau précédent. On constate de plus que pour les codes MDS sur $GF(2^{16})$, les temps de décodage par bloc deviennent rapidement importants (plusieurs dizaines de secondes), par exemple pour k=4096.

Dès lors, l'impact de k sur la vitesse de réception globale doit se faire en considérant le mode de décodage (explicité plus loin), ainsi que d'autres paramètres de transmission : le débit d'émission de la source (mesuré au niveau transport), D, la taille du message à transmettre, S, et un paramètre supplémentaire, B_e, que nous allons définir.

3.2.2 *Évaluation analytique de $\eta_{décodage}$ en fonction du mode du décodage*

Pour procéder à l'évaluation, nous devons au préalable présenter le principe de l'entrelacement par bloc au niveau transport et son objectif.

3.2.2.1 Entrelacement par bloc

L'objectif de l'entrelacement à ce niveau est identique à celui de la couche physique. Il s'agit de disperser les erreurs, se manifestant sous la forme de pertes de paquets successives (*fading*), sur plusieurs blocs afin de rendre plus efficace le décodage. L'entrelacement est normalement inutile avec les codes de types LDPC puisque les mots de code peuvent être très longs (k pouvant valoir plusieurs dizaines de milliers).

On suppose que les données après codage ont été découpées en B blocs. Chaque paquet peut être identifié par le couple d'indices (i, j), où i fait référence à l'indice du paquet dans son bloc respectif (i va donc de 1 à n), et j au numéro du bloc auquel le paquet appartient (j varie de 1 à B). Une transmission conventionnelle non entrelacée se fait dans l'ordre : $\{P_{1,1}\ P_{1,2}\ \ldots\ P_{1,n}\ P_{2,1}\ P_{2,2}\ldots P_{2,n}\ldots P_{B,n}\}$. Au contraire, si l'on entrelace les données sur B_e blocs, la transmission se fait dans l'ordre : $\{P_{1,1}\ P_{2,1}\ \ldots\ P_{Be,B}\ P_{1,2}\ P_{2,2}\ \ldots P_{Be,B,}$ puis $P_{1,Be+1}\ P_{2,\ Be+1}\ \ldots P_{B,n}\}$. Cet ordre de transmission est représenté graphiquement sur la Figure 3-17. Nous ferons référence au terme *super-bloc* comme étant un groupe de B_e blocs consécutifs. Un super-bloc encadré en bleu est illustré sur la Figure 3-17.

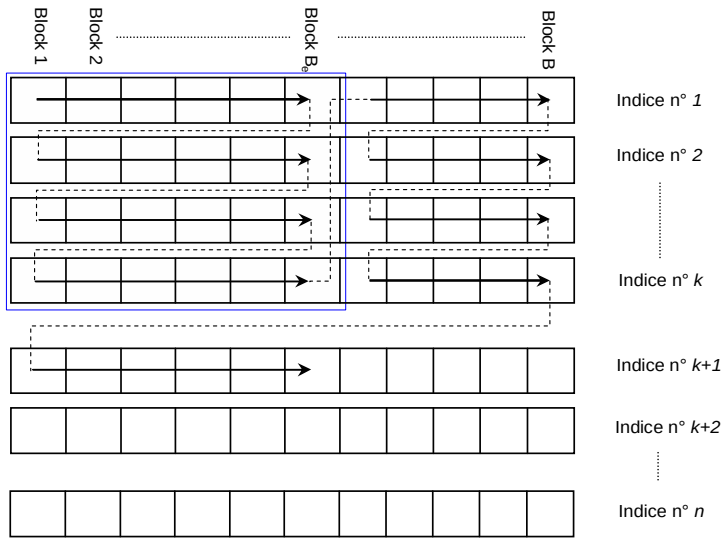

Figure 3-17 : Organisation temporelle de l'émission des données avec un entrelacement par blocs

Entrelacer les données par bloc engendre toutefois deux inconvénients. Le premier est lié à une augmentation de l'utilisation de la mémoire des stations d'extrémité, car les B_e blocs FEC doivent être stockés avant le codage. Pour des valeurs de $k \leq 1000$, et une longueur de paquets de 1500 octets, cela peut représenter jusqu'à 150 Mo si l'on a choisi $B_e=100$. En outre, il peut être préférable d'anticiper le codage avant le début de la transmission si celui-ci devait ralentir le processus d'émission ; dans ce cas un délai serait engendré entre la soumission du message applicatif au service de transport et l'émission effective des premières données sur le réseau.

Choisir une valeur de B_e élevée permettra au décodage de supporter une durée de *fading* plus importante par super-bloc. Cette durée maximale vaut :

$$T_{\max} = \frac{h.B_e.L}{D} \tag{3.27}$$

Pour donner un ordre de grandeur, T_{max} vaut environ 14 secondes si B_e est égal à 100, h à 100, L à 1500 octets et D à 1 Mo/s.

3.2.2.2 Évaluation du délai dans le mode de décodage simple

Lorsque l'on décide de procéder au décodage après la réception de *tous* les blocs (appelé mode de décodage *simple*), on peut facilement estimer le délai T de réception global. Il est égal à la somme du temps de téléchargement des données (S/D) et du délai provenant du décodage de tous les blocs. Ce dernier terme est égal au produit du nombre de blocs composant le message applicatif, $S/(L.k)$, par le temps de décodage par bloc, $t_d(k)$. Soit :

$$T = \frac{S}{D} + \frac{S.t_d(k)}{L.k} \tag{3.28}$$

En divisant l'équation précédente par S/D, on obtient le facteur $\eta_{décodage}$:

$$\eta_{décodage} = 1 + D.\frac{t_d(k)}{L.k} \tag{3.29}$$

En remarquant que $k.L/D$ représente le temps moyen de téléchargement d'un bloc (sans pertes de paquets) et en appelant ce temps t_B, on peut réécrire l'équation précédente sous la forme :

$$\eta_{décodage} = 1 + \frac{t_d(k)}{t_B} \tag{3.30}$$

Finalement, en définissant μ comme le rapport entre $t_d(k)$ et t_B, on a :

$$\eta_{décodage} = \mu + 1 \tag{3.31}$$

3.2.2.3 Évaluation du délai dans le mode de décodage anticipé

La solution précédente a l'avantage de la simplicité de mise en œuvre, mais ses performances peuvent être améliorées si le décodage est réalisé en parallèle à la réception. Pour le calcul qui suit, nous ferons l'hypothèse que le décodage de chaque super-bloc ne peut commencer qu'après sa réception complète. Ainsi, le premier décodage ne peut commencer qu'à la fin de la réception du premier super-bloc.

Deux cas sont à prévoir : soit le décodage est suffisamment rapide pour rattraper son retard sur le processus d'émission, soit il ne l'est pas.

Si le décodage est plus rapide que l'émission des données, le temps de réception total est dominé par le temps de réception des B blocs, plus le temps de décodage lié au dernier super-bloc. On a donc :

$$\eta_{décodage} = \frac{B_e}{B}\mu + 1 \tag{3.32}$$

Au contraire, si le temps de décodage est plus long que le temps d'émission par bloc, le temps de réception est égal au délai de réception du premier super-bloc plus le temps de décodage de tous les blocs. $\eta_{décodage}$ s'exprime dans ce cas par :

$$\eta_{décodage} = \mu + \frac{B_e}{B} \tag{3.33}$$

Le cas limite, pour lequel les deux équations précédentes sont égales, est atteint lorsque μ=1, c'est-à-dire lorsque les deux vitesses sont égales.

Ainsi, choisir une valeur de B_e élevée augmente l'inefficacité de décodage. Si B_e est égal au nombre de blocs B, l'anticipation n'a plus de sens, et l'inefficacité est à nouveau donnée par (3.31).

3.2.2.4 Comparaison des deux modes de décodage

Sur la Figure 3-18, $\eta_{décodage}$ est tracé en fonction de μ pour les deux modes de décodages. Lorsque μ est supérieur à 1, l'écart entre les performances des modes de décodage est constant ; il vaut 1-(B_e/B). Sinon, il vaut μ.(1-B_e/B).

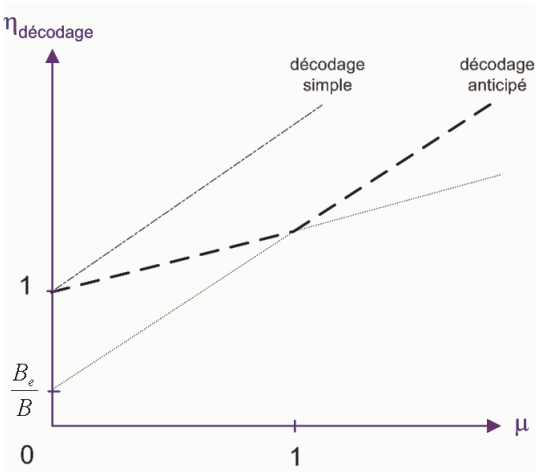

Figure 3-18 : Évolution de $\eta_{décodage}$ en fonction de μ pour les deux modes de décodage

3.2.3 *Estimation numérique de $\eta_{décodage}$*

L'estimation numérique de $\eta_{décodage}$ ne peut être faite qu'en fixant une partie des paramètres. Dans la suite, on s'intéresse à l'influence du paramètre k. On a choisi ici D=1 Mo/s, B_e=10, L=1500 octets.

Sur la Figure 3-19, $\eta_{décodage}$ est représenté en fonction de k pour les codes sur les entiers. On s'aperçoit qu'en anticipant le décodage, une valeur de k égale à 1000 n'engendre pratiquement aucun délai pénalisant par rapport au délai de transmission. Le paramètre μ (non donné ici) est toujours inférieur à 1 avec ce jeu de paramètres.

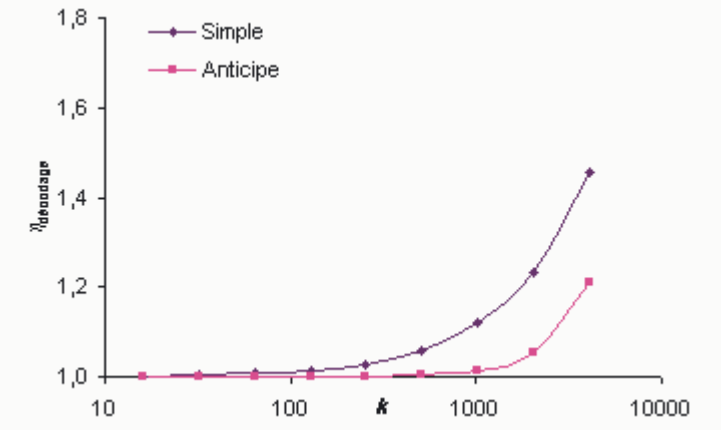

Figure 3-19 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage sur entiers construit sur GF(2^{16}+1)

La Figure 3-20 présente la même courbe que précédemment pour le codage sur matrices de Cauchy sur GF(2^{16}). L'analyse précédente est aussi valable dans ce cas de figure, mais les résultats obtenus sont un peu moins bons ici.

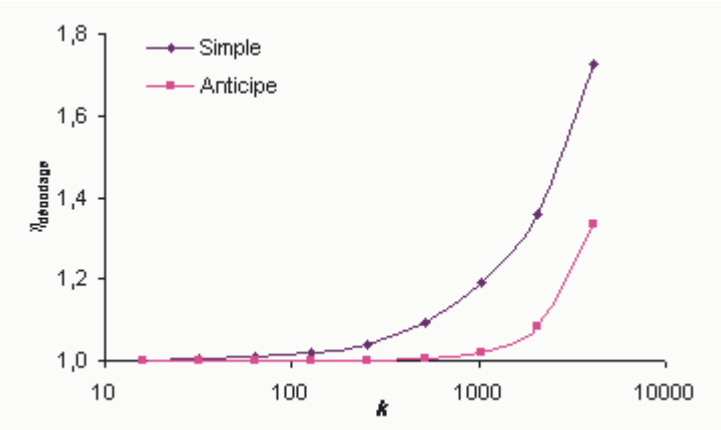

Figure 3-20 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage sur matrice de Cauchy construit sur GF(2^{16})

Le code basé sur les matrices VDM sur GF(2^{16}) (Figure 3-21) ne permet pas le décodage pour k>1024. D'autre part, l'inefficacité s'accroît considérablement lorsque k>512, ce qui rend peu intéressant ce code sur GF(2^{16}). De plus, μ est supérieur à 1 pour ces valeurs de k.

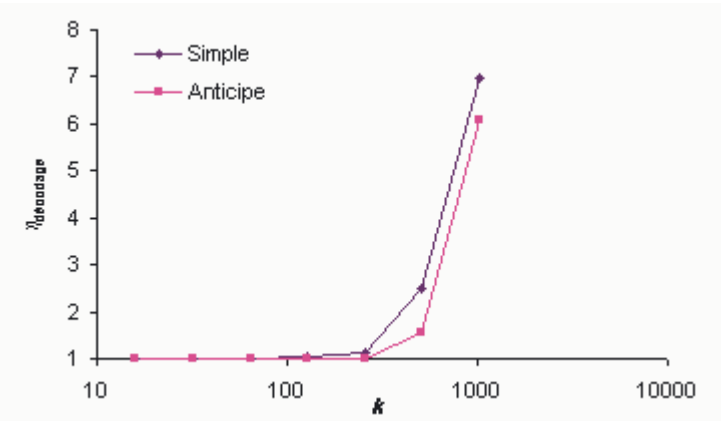

Figure 3-21 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage sur matrice de Vandermonde construit sur GF(2^{16})

Lorsque l'on exploite les codes sur matrices de Cauchy (Figure 3-22) ou matrices VDM (Figure 3-23) sur le corps GF(2^8), l'inefficacité de décodage devient très faible pour le décodage simple (entre 0 et 1%). Si le décodage est anticipé, l'inefficacité disparaît pratiquement. Dans ces deux cas, μ est toujours inférieur à 1.

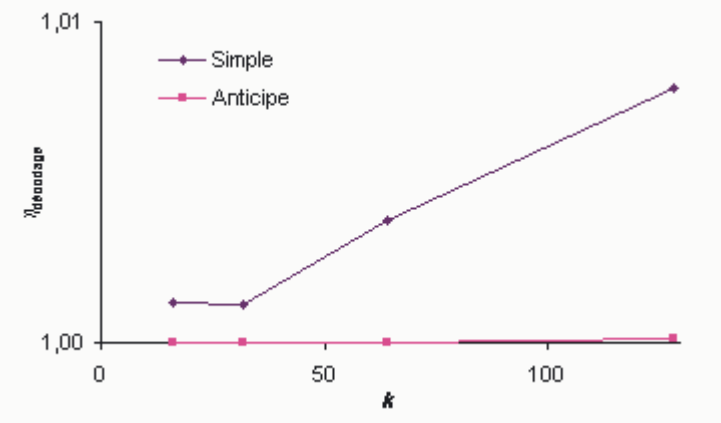

Figure 3-22 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage sur matrice de Cauchy construit sur GF(2^8)

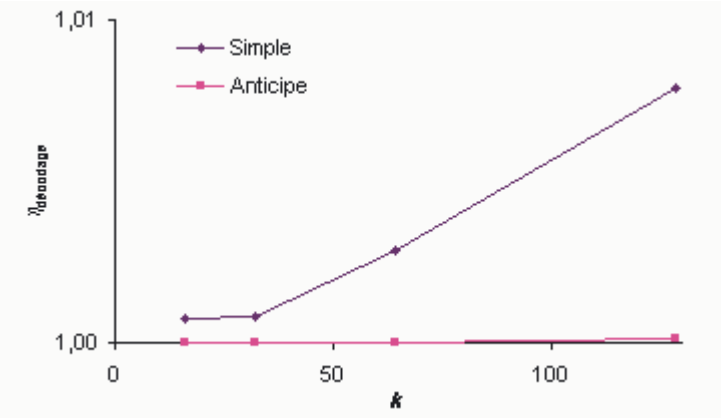

Figure 3-23 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage sur matrice de Vandermonde construit sur GF(2^8)

Enfin, avec les codes LDPC (Figure 3-24), l'inefficacité due à la rapidité de décodage est négligeable, que le décodage soit anticipé ou non. μ reste ici très inférieur à 1.

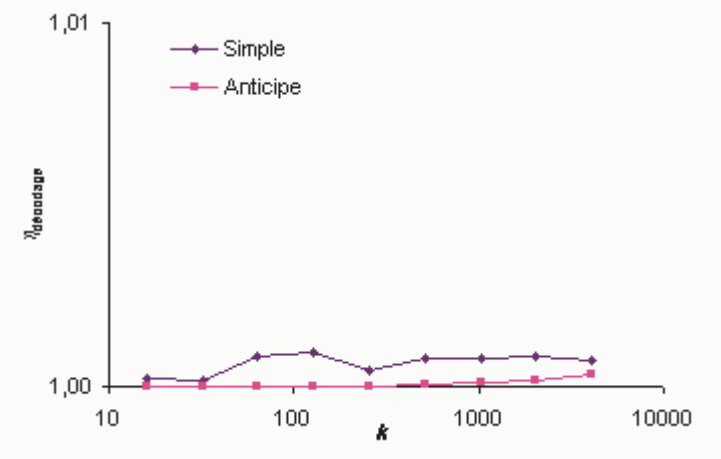

Figure 3-24 : Évolution de $\eta_{décodage}$ en fonction de k pour un codage de type LDPC

CONCLUSION

Au cours de ce chapitre, nous avons pu mettre en évidence que la gestion de la fiabilité au niveau du protocole de transport devait conduire à accepter des compromis. Plusieurs paramètres ou choix d'implémentations ont été étudiés en prenant en compte les spécificités du contexte satellitaire. Pour évaluer les performances des mécanismes, nous avons étudié deux métriques : le nombre de paquets transmis au cours d'une session et le temps d'acquisition par l'ensemble des récepteurs. Ces deux grandeurs ont été estimées par différentes méthodes au travers de deux facteurs d'inefficacité, $\eta_{fiabilisation}$ et $\eta_{décodage}$. Le premier d'entre eux indique le surplus de quantité d'information à transmettre par rapport à la taille du message original exprimée en nombre de paquets, et dépend de la politique de fiabilisation choisie, des paramètres du code FEC, et de la taille du groupe. Le second est aussi lié aux paramètres du code, à la nature du codage (MDS ou LDPC), et à la mise en œuvre éventuelle de l'entrelacement.

Ensuite, nous avons pu voir, qu'il est recommandé de ne pas utiliser les codes de type MDS pour des valeurs de k trop grandes (de l'ordre de 1000), sous peine d'augmenter de façon inacceptable les temps de décodage des données au niveau des récepteurs. Au contraire, les codes basés sur les LDPC n'entraînent pratiquement pas de délais supplémentaires, même pour d'importants volumes de données transférés.

Comme annoncé précédemment, aucune des solutions présentées ne parvient à être optimale. Si dans la majorité des cas le service de la liaison satellitaire est assez robuste pour occulter les conséquences de légères dégradations du signal (scintillations, *fading* de courtes durées), de fortes dégradations du signal peuvent avoir un effet désastreux sur les performances de la communication, au point d'interrompre le service. L'estimation théorique des performances du schéma Hybride ARQ de type II nous a révélé que le niveau de transport pouvait également réduire cet effet de façon significative, mais sous plusieurs conditions. Ces conditions dépendent avant tout du code employé.

Dans le cas d'un code MDS, il faudra veiller à optimiser la valeur donnée à k pour parvenir à un compromis entre robustesse et délai de livraison à l'utilisateur. D'autre part, pour masquer l'effet de *fading* de durée conséquente, le meilleur moyen consiste à entrelacer les données en dispersant les erreurs sur plusieurs blocs. Grâce à une mise à disposition d'une quantité de mémoire suffisamment importante au niveau des stations émettrice et réceptrices et de paramètres de codage adaptés (valeurs du couple (n, k) et du nombre de blocs entrelacés), l'entrelacement permet de s'affranchir de pertes de paquets sur plusieurs secondes, pour des débits de transmissions élevés. Lorsque la taille du message est suffisamment importante et pour k inférieur à 1000, les meilleures implémentations de codage MDS (codes basés sur les matrices de Cauchy ou codes sur les entiers en cours de développement) parviennent à être assez rapides pour ne pas pénaliser le récepteur si le décodage est anticipé. Si peu de blocs sont entrelacés, ce qu'il convient de choisir pour des applications à temps contraint ne demandant pas un service à fiabilité totale afin de limiter les délais, il suffit que la vitesse de décodage soit supérieure à celle de transmission.

Dans le cas d'un code de type LDPC, des blocs très longs peuvent être choisis (k de l'ordre de 10000, voire plus) ce qui a l'avantage de pouvoir être à la fois robuste aux pertes de paquets résultant de *fading* sans recourir à un entrelacement, tout en étant simple et rapide à décoder. L'alternative LDPC est donc une solution envisageable, à la fois pour des applications demandant un service de fiabilité totale et pour des applications à contraintes temporelles excluant un décodage trop gourmand en ressources CPU. A l'heure actuelle la plupart des implémentations logicielles de ces codes sont protégées par brevets (propriétés de Digital Fountain) et ne sont donc pas disponibles pour être intégrées dans un protocole de transport. Quelques variantes sont

apparues très récemment dans le domaine public, sous le nom *Low Density Generator Matrix* (LDGM) [Roc03]. Ces implémentations ont pu permettre de mesurer le facteur d'inefficacité *f* des codes LDPC. Pour la meilleure variante de LDGM (*LDGM Triangle*) l'inefficacité moyenne obtenue sur 1000 itérations vaut 1.055 lorsque k>10000, avec un intervalle de confiance à 99 % compris entre 1.05 et 1.065.

Enfin, les limitations des deux alternatives possibles, MDS ou LDPC, ont été soulignées. Dès lors, le choix de l'une ou l'autre (et éventuellement celui des paramètres de codage) est de la responsabilité de l'opérateur : pour une meilleure tolérance aux erreurs et des décodages plus rapides, les architectures avec codes basés sur LDPC sont préférables. En revanche, pour une économie de bande passante, les architectures de transport avec codage MDS sont à privilégier. Avec l'augmentation des vitesses de calcul et des progrès de recherches éventuelles, cette dernière solution pourrait à terme s'imposer.

BIBLIOGRAPHIE

[Arn03] "Reliable Multicast Transport Protocols Performances In Emulated Satellite Environnement Taking Into Account An Adaptive Physical Layer For The Geocast System", F.Arnal *et al.*, International Workshop of COST Actions 272 and 280 ESTEC, Noordwijk, the Netherlands, mai 2003.

[Blo95] "An XOR-Based Erasure Resilient Coding Scheme", J. Blömer *et al.*, ICSI Technical Report TR 95-048, août 1995.

[Dai04] "Content-Access QoS in Peer-to-Peer Networks Using a Fast MDS Erasure Code", L. Dairaine *et al.*, à paraître dans le journal Computer Communications.

[Lin83] "Error Control Coding, Fundamental and Applications", S. Lin, D. J. Costello Jr., Prentice-Hall, Englewood Clis, NJ, 1983.

[Lub02] "LT codes", M. Luby, 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002.

[LUB] http://www.icsi.berkeley.edu/~luby/

[Non97] "Parity-based loss recovery for reliable multicast transmission Networking", J. Nonnenmacher, E. W. Biersack, and D. Towsley, in Proceedings of ACM SIGCOMM '97, 1997.

[Non99] "Reliable multicast via satellite: Uni-directional vs. bi-directional communication", M. Jung, J. Nonnenmacher, E. W. Biersack, in proceedings of KiVS'99, Darmstadt, Germany, mars 1999.

[Pla03] "On the Practical Use of LDPC Erasure Codes for Distributed Storage Applications", J.S Planck, M.G. Thomason, Technical Report UT-CS-03-510, University of Tennessee, septembre 2003.

[RIZ] http://info.iet.unipi.it/~luigi/fec.html

[Riz97] "Effective Erasure Codes for Reliable Computer Communications Protocols", L.Rizzo, ACM Computer Communication Review, vol. 27, no. 2, avril 1997.

[Roc03] "Design and Evaluation of a Low Density Generator Matrix (LDGM) large block FEC codec'", V. Roca, Z. Khallouf, J. Laboure, Fifth International Workshop on Networked Group Communication (NGC'03), Munich, Germany, septembre 2003.

[Roc04] "Design, Evaluation and Comparison of Four Large Block FEC Codecs, LDPC, LDGM, LDGM Staircase and LDGM Tiangle, plus a Reed-Solomon Small Block FEC Codec", V. Roca, C. Neuman, rapport de recherche INRIA n°5225, juin 2004.

[Sho04] "Raptor Codes", A. Shokrollahi, Digital Fountain White Paper, mai 2004.

Chapitre 4

Élaboration d'un outil de simulation permettant la mesure de la fiabilité

INTRODUCTION

L'étude de mécanismes de protocoles de transport multipoints a été menée dans la partie précédente sous des hypothèses particulières de distributions de pertes de paquets. Malgré la complexité de l'analyse, la nature spécifique des liaisons par satellite n'a pas été correctement prise en compte, puisqu'il a simplement été admis qu'à certains instants les signaux étaient trop perturbés pour que le codage canal puisse masquer, par corrections, la corruption des données. Les mécanismes de détection et de rejets de paquets défectueux dans les couches basses justifient qu'au niveau de la couche transport les erreurs se manifestent sous la forme de pertes de paquets. En revanche, il faut s'interroger sur la qualité des modèles de pertes utilisés. Dans un premier temps nous avons supposé des pertes de paquets aléatoires et uniformément réparties au cours du temps et pour l'ensemble des terminaux satellite. Il a ensuite été possible d'améliorer le modèle en le généralisant, afin de prendre en compte le fait que des pertes différentes pouvaient affecter les terminaux. En revanche, proposer un modèle réaliste de répartition temporelle des pertes de paquets n'est pas simple si l'on souhaite le déployer à grande échelle. La difficulté provient essentiellement de la complexité de la modélisation de toute la couche physique.

Dans l'objectif d'étude de la gestion de la fiabilité des architectures de transmission que nous nous sommes fixés, connaître l'influence des mécanismes implémentés dans les couches basses (*i.e.* du niveau physique au niveau réseau inclus) est une amélioration certaine du modèle précédent. À cet effet, au moins trois méthodes paraissent envisageables : (1) mesurer des pertes réelles de paquets d'une liaison satellite DVB-S et concevoir d'après ces mesures un modèle général de pertes, (2) estimer par analyse mathématique les performances de fiabilité à partir de l'ensemble des caractéristiques du système de communication satellitaire et (3) simuler une liaison par satellite DVB-S avec les mécanismes protocolaires nécessaires pour caractériser son service.

Chacune de ces méthodes a ses points forts et ses limites. En ce qui concerne (1), la principale difficulté consiste à disposer des moyens de mesures adéquats, et de la capacité nécessaire d'utilisation d'une liaison satellitaire. Une fois ces données collectées dans des situations suffisamment représentatives, la tâche de modélisation reste complexe. Par exemple, elle peut être très dépendante de paramètres comme la bande de fréquence opérée. Toutefois, cette méthode dépasse son objectif initial, car elle permet la modélisation du canal de transmission. Si l'on parvient à mettre en œuvre cette méthode, la qualité de sa modélisation est assurée.

La seconde méthode (2) peut paraître la plus simple des trois. Par exemple, partant des performances théoriques de la modulation employée, puis du codage canal, il est possible de parvenir par étapes successives, à l'expression de grandeurs comme le taux d'erreurs paquets moyen (p.ex. au niveau du service IP). Malheureusement, ces calculs sont complexes et doivent être menés sous couvert d'hypothèses fortes pouvant être discutables (par exemple que les distributions d'erreurs-bits ou de pertes de paquets sont uniformes). D'autre part, ce type de modélisation ne permet de fournir que des valeurs statistiques (moyennes, voire écarts-type) qui laissent indéterminés les profils de pertes recherchés.

La méthode (3), tout comme (1), est complexe. Certaines fonctionnalités de la simulation sont relativement simples à intégrer individuellement mais la difficulté de la tâche réside dans l'exhaustivité et la précision de l'ensemble des fonctions à implémenter. D'autre part, la spécification des normes DVB ne détaille pas certains points de mise en oeuvre (réalisation du modulateur/démodulateur, acquisition de la synchronisation, etc…), qui sont laissés au choix des constructeurs, et qui peuvent influencer notablement les performances du système. En dépit de cette limitation, nous avons préféré cette solution à (2) qui exigeait de faire des hypothèses trop fortes, et à la solution (1) en raison du contexte de travail de la thèse rendant peu évident la réalisation de campagnes de mesures.

La suite de ce chapitre est structurée comme suit. Dans la première partie, les détails de la conception du simulateur sont donnés ainsi que son apport potentiel. Quelques exemples de code source de simulations sont ensuite commentés. Le chapitre conclut sur une série d'études caractérisant les performances d'erreurs au niveau des services MPEG-2 TS et IP.

4.1 DESCRIPTION DU SIMULATEUR

4.1.1 *Besoin*

À notre connaissance, aucun logiciel ne permet l'étude de la fiabilité d'une liaison DVB-S pour un service IP. La cause du problème est qu'il se situe au cœur de deux domaines traditionnellement dissociés : l'ingénierie de protocoles (traitée par des outils tels que OPNET [OPN] ou Network Simulator [NS2]) et le traitement de signal associé aux communications numériques (pour la modélisation de la liaison satellite). Aussi, pour envisager une telle étude, un nouveau simulateur a dû être développé.

4.1.1.1 Description générale de l'outil

4.1.1.2 Finalité

L'objectif fondamental de ce simulateur est d'obtenir un modèle d'erreurs au niveau binaire typique d'une liaison satellite DVB-S, et de la répercuter au niveau du service de livraison de datagrammes IP. Un maximum de fonctionnalités relatives à la fiabilité des données numériques a pour cela été implémenté, respectant au mieux les formats des unités de données des protocoles du plan utilisateur. En revanche, l'échange de données de signalisation (tables DVB ou autre signalisation des protocoles de la pile IP) n'a pas été considéré.

4.1.1.3 La bibliothèque IT++

Le simulateur a été implémenté à partir de la bibliothèque de fonctions mathématiques, de traitement du signal et de théorie de l'information dénommée IT++ [Itpp04]. IT++ a été développé à l'origine par une équipe de chercheurs de l'Université de Technologie de Chalmers à Göteborg (Suède), et s'est enrichie progressivement de contributions extérieures de la part

d'autres chercheurs du domaine de la théorie de l'information. Elle est développée en langage C++.

Similaire dans son utilisation à un logiciel tel que Matlab sur plusieurs aspects, IT++ offre des outils de base pratiques pour manipuler plusieurs types de données comme les vecteurs, les matrices, les données binaires, les nombres complexes, etc… La bibliothèque contient également plusieurs fonctions de traitements de l'information dans les communications numériques, en particulier les modulations numériques, les modèles de canaux de transmissions et les fonctions implantant les codages correcteurs d'erreurs.

4.1.2 *De IT++ à la conception du simulateur*

Outre l'ajout ponctuel de quelques fonctions, le simulateur met en œuvre de nombreuses fonctions existantes dans IT++ utilisées avec les paramètres adéquats. Le simulateur peut être vu comme un enchaînement de blocs (implémentés sous forme de classes) de traitement des données qui peuvent être représentés schématiquement par la Figure 4-1. Les fonctions réalisées au sein de chaque bloc sont détaillées dans la suite de cette section.

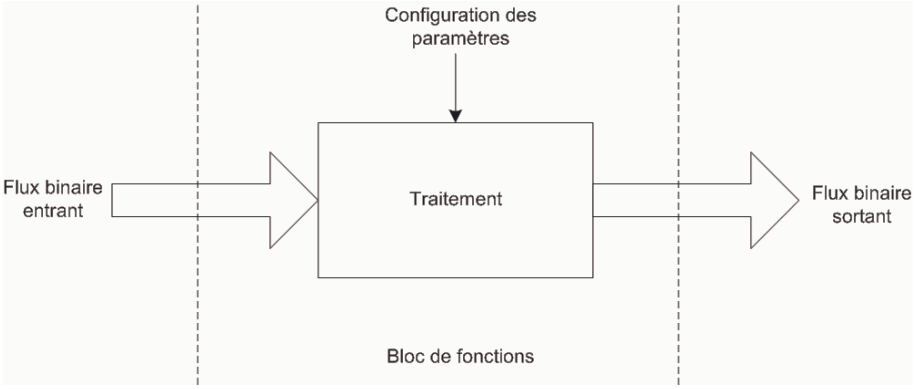

Figure 4-1 : Représentation fonctionnelle d'un bloc de traitement

4.1.2.1 Présentation du simulateur par fonctionnalité

4.1.2.1.1 Couche physique

4.1.2.1.1.1 Simulation de la chaîne de codage DVB-S

Les fonctions accomplissant les traitements de DVB-S ont été modélisées dans un bloc, dont la représentation (coté émetteur) est donnée en Figure 4-2. Néanmoins, la complexité du bloc a nécessité une division en blocs élémentaires plus faciles à gérer.

Le principal travail réalisé pour développer ce bloc a été l'implémentation de l'entrelacement convolutionnel (et du désentrelacement réciproque). Son effet sur l'ordre de transmission des données est illustré par la Figure 4-3. Individuellement, les autres fonctions n'ont pas engendré de nouveaux développements par rapport à la bibliothèque de base, sauf pour le codage/décodage RS qui a été adapté pour être raccourci (à partir du code (n=255, k=239, d=17)), et mis sous forme systématique. L'interfaçage entre les blocs élémentaires a ensuite pu être complété.

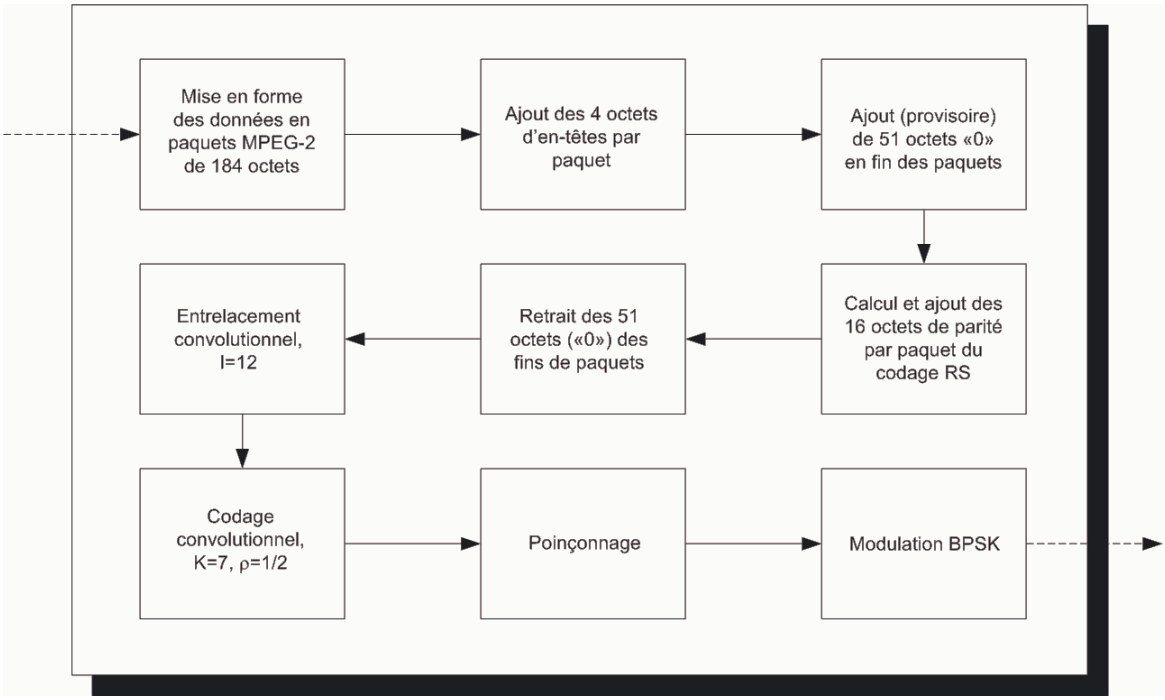

Figure 4-2 : Bloc DVB-S implémenté dans le simulateur (coté émetteur)

Dans cette implémentation, et conformément à la norme [ETSI97], la longueur de contrainte du code convolutionnel, *K*, est égale à 7, et les deux polynômes générateurs du code, donnés en notation octale, sont : $G_1(D)=133$, $G_2(D)=171$. Le décodage du code est réalisé par l'algorithme de Viterbi à entrées et décisions souples. Il s'effectue par une troncature de la mémoire (classiquement implémentée) sur une fenêtre de 5.*K* symboles binaires.

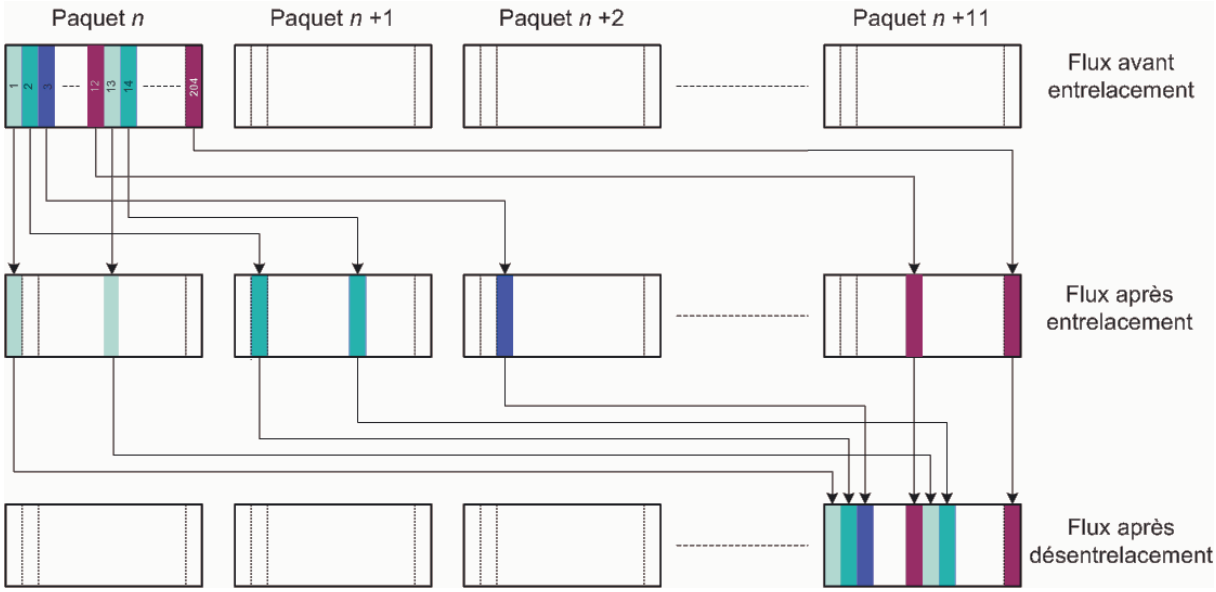

Figure 4-3 : Modification sur l'ordre de transmission des données par le mécanisme d'entrelacement convolutionnel

La modulation mise en œuvre par le simulateur est de type *Binary Phase Shift Keying* (BPSK), déjà implémentée dans IT++, au lieu de la modulation QPSK spécifiée par DVB-S. Les performances des deux modulations étant identiques du point de vue de la fiabilité, ce choix convient pour l'approche employée par le simulateur.

4.1.2.1.1.2 Simulation du canal de transmission

Pour modéliser la propagation, le signal reçu à chaque instant par les antennes (au niveau du satellite ou des terminaux terrestres) est supposé ne provenir que d'une seule composante en vue directe. D'éventuelles contributions issues de réflexions parasites ne sont donc pas prises en compte. Les terminaux étant supposés fixes d'autre part, les signaux ne sont pas perturbés par l'effet Doppler. Pour ces raisons, le canal à bruit additif blanc et gaussien (*Additive White Gaussian Noise*, AWGN) a été considéré comme un modèle adapté.

4.1.2.1.2 Limites du modèle

L'acquisition de la synchronisation pour la démodulation n'a pas été modélisée ici en raison de la trop grande complexité des opérations qu'il aurait fallu intégrer. Cela implique une importante limitation du simulateur à accepter, car lorsque le signal d'une liaison réelle est trop fortement perturbé, la synchronisation du récepteur numérique du terminal n'est plus possible. Nous devrons donc admettre que la limite du seuil d'accrochage sur la porteuse se situe en dessous des plus petites valeurs du rapport signal à bruit considéré.

De plus, on estime généralement que le démodulateur introduit une perte d'implémentation de quelques dixièmes de décibels sur le bilan de liaison, qui n'ont pas été considérés dans le simulateur. Toutefois, dans la majorité des simulations, l'analyse se fera par des comparaisons des résultats. Le « décalage » qu'il faudrait intégrer sur toutes les simulations n'aura donc pas de conséquence importante.

4.1.2.1.3 Modélisation des traitements liés à la fiabilité au sein des protocoles

4.1.2.1.3.1 Simulation du service MPEG-2 TS

L'usage de la méthode *Section Packing* (cf. 1.2.3.2.2) a été supposé pour encapsuler les unités de données de niveau supérieur (i.e. les SNDUs de MPE ou ULE) dans la charge utile des paquets MPEG-2 TS. Dès lors, au niveau de l'émetteur MPEG-2 TS, le flux est segmenté en paquets de 184 octets (correspondant à la longueur de la charge utile maximum des paquets MPEG-2 TS) et un en-tête aléatoire de 3 octets par paquet est ajouté en plus de l'octet de synchronisation.

4.1.2.1.3.2 Simulation de l'encapsulation MPE/ULE

Le simulateur met en œuvre la méthode d'encapsulation ULE en cours de normalisation. Toutefois, certaines implémentations réelles ont utilisé, avant le début de la normalisation, une technique d'encapsulation équivalente (*i.e.* sans utiliser le champ *Adaptation Field*) mais avec le formatage DSM-CC. Une simple modification du format de paquet dans le simulateur est suffisante pour modéliser cette encapsulation ; pour cette raison nous appelons l'encapsulation modélisée MPE/ULE. Nous préciserons lequel des deux formats est utilisé dans les simulations.

4.1.2.1.3.3 Simulation d'une couche réseau générique

Ensuite, une classe générique de format d'unités de données a été développée. Cette classe prend en paramètres la longueur des champs d'en-têtes et d'en-queues de la couche désirée qui est supposée constante. Du côté de l'émetteur, la classe ajoute simplement aux données des en-têtes et des en-queues, aléatoires lorsqu'elles sont sans rapport avec la gestion de la fiabilité, et qui sont retirés au niveau du processus de réception. La classe a été employée pour modéliser IP (dans sa version 4), ULE (ou MPE), et les protocoles de niveaux supérieurs.

4.1.2.1.3.4 Mécanisme de détection d'erreur et de rejets des paquets

En parallèle des modélisations du service MPEG-2 et des couches réseaux génériques, un mécanisme de contrôle d'intégrité des unités de données et de rejets est mis en place.

Le transport de datagrammes IP sur DVB par les méthodes existantes prévoit de supprimer tout paquet MPEG-2 pour lequel la correction RS échoue. Alors, dans le cas où un paquet MPEG-2 est manquant pour le réassemblage d'une SNDU, cette dernière est supprimée. Pour implémenter ce mécanisme, les paquets supprimés sont remplacés par une série de « 1 » binaires pour conserver une longueur identique entre les flux émis et reçus. La raison du choix de ce type d'implémentation apparaîtra lors de la discussion sur l'exploitation du simulateur au niveau de la couche transport.

En ce qui concerne la couche réseau générique, il est possible de choisir le type de fiabilisation désiré sur le paquet (CRC ou somme de contrôle), le positionnement de ces données (en fin d'en-tête ou à la suite de la charge utile) ainsi que la zone de données contrôlée. À la réception, le processus recalcule le CRC ou la somme de contrôle, et en cas de différence, l'unité de données est virtuellement supprimée, par remplacement de toutes ses données par des « 1 ».

4.1.2.1.4 Limites du modèle

Pour faciliter l'implémentation de l'octet *Pointer Field* indiquant l'adresse du premier octet de la SNDU, nous avons supposé que celui-ci n'était pas positionné juste après l'en-tête MPEG-2 TS, mais juste avant la SNDU, comme si son en-tête comportait un octet supplémentaire (p.ex. 13 à la place de 12 pour MPE). Ensuite, puisque le rejet éventuel de paquets intervient à partir d'une seule erreur octet dans tout le paquet, on peut considérer que l'échange de position de cet octet dans l'implémentation n'entraîne pas de conséquence sur la modélisation du service MPEG-2 TS.

4.1.3 ***Grandeurs mesurées par le simulateur***

Dans cette partie, les mesures présentant un intérêt sont principalement les valeurs moyennes des taux d'erreurs et de rejets des unités de données aux différents niveaux de service. En revanche, les distributions d'erreurs seront également utiles lorsque les mécanismes de fiabilisation mis en œuvre au niveau du protocole de transport seront abordés (cf. chapitre 6).

4.2 EXEMPLES DE SIMULATION

4.2.1 ***Transmission DVB-S***

Afin d'illustrer la présentation sur un exemple simple, nous décrivons le fonctionnement d'une transmission par MPEG-2/DVB-S dans un canal AWGN bruité. La Figure 4-4 ci-dessous en donne le code source.

```
#include "itcomm.h"
#include "packet.h"

int main(int argc, char *argv[])
{
  // Instance des objets de la simulation
  DVB dvb ;
  BPSK bpsk ;
  AWGN_Channel channel;

  // Paramètres de la simulation
  dvb.set_code_rate(3, 4);
  channel.set_noise(dvb.get_N0(2.0)); // Ec/N0 = 2 dB
  int bytes_to_send=1000*184 ;  //Données pour remplir 1000 paquets MPEG2
```

```
  // Mise en place de la simulation
  RNG_randomize();
  bvec data=randb(8*bytes_to_send);
  bvec mpeg_flow=dvb.encode_and_fit_mpeg2(data, TRUE);
  vec tx_signal=bpsk.modulate_bits(mpeg_flow);
  vec rx_signal=channel(tx_signal);

  ivec MPEG2_errors ;
  bvec out_mpeg=dvb.decode_and_fit_mpeg2(rx_signal, MPEG2_errors,FALSE);

  // Affichage des résultats
  cout << Paquets MPEG-2 TS erronés : <<  MPEG2_errors << endl ;
}
```

Figure 4-4 : Code source d'une simulation de transmission de données par DVB-S

Le code est divisé en quatre parties. Après l'instanciation des différents objets de la simulation, leurs attributs sont fixés. Le taux de codage du code convolutionnel est choisi par le couple (*n, k*) - ici (4, 3) - par la méthode *set_code_rate()*. Ceci est effectué au sein de la classe DVB en sélectionnant la matrice de poinçonnage adéquate. Puis, on fixe le niveau de bruit du canal de transmission, par l'intermédiaire du rapport E_c/N_0 (rapport entre l'énergie par bit transmis après codage et la densité de puissance de bruit). Dans l'implémentation de la classe DVB, E_c reste constant et seul varie N_0. La méthode *get_N0()* permet d'obtenir la valeur de N_0 qui convient pour obtenir la valeur du rapport E_c/N_0 désirée, donnée en argument de la méthode (ici, E_c/N_0 =2.0 dB). Enfin, la variable *bytes_to_send* indique la quantité de données en octets qui sera acheminée par MPEG-2 TS.

La troisième partie de la simulation commence par la création un flot aléatoire de données, fourni en tant que données de service à la couche DVB/MPEG-2 par l'intermédiaire de la méthode *encode_and_fit_mpeg2()*. Le flot résultant, *mpeg_flow,* alimente ensuite la fonction simulant la modulation, qui calcule le signal transmis appelé *tx_signal*. On fait subir à ce signal une déformation modélisant le canal de transmission. Le signal reçu (*rx_signal*) est alors traité par la méthode *decode_and_fit_mpeg2()*[5] de la classe DVB qui effectue la démodulation et les décodages. Elle produit en sortie le flot binaire *RX_DATA*. Les données sont ensuite traitées par les différentes couches.

La dernière phase de la simulation sert à l'affichage des résultats. Sur l'exemple de la Figure 4-5, le décodage de 9 paquets a échoué.

```
arnal@tesap26-1 ~/projets_it++/DVB
$ ./sim_DVB
Paquets MPEG-2 TS erronés : [312 480 481 482 606 749 956 957 958]
```

Figure 4-5 : Résultat d'une simulation de transmission de paquets MPEG-2 TS

4.2.2 *L'encapsulation IP/DVB*

Le second exemple de simulation (Figure 4-6) illustre le transport de datagrammes IP par DVB-S avec l'encapsulation MPE. Au début de cette seconde simulation, un message aléatoire, *input_data*, est crée. Celui-ci est délivré à la couche IP sous forme d'unités de données de service de taille constante (égale à IP_PDU_SIZE). IP se charge d'ajouter les en-têtes (aléatoires aussi) et d'effectuer le calcul de la somme de contrôle. Le flux ainsi constitué est alors transmis à l'encapsulateur MPE qui ajoute ses propres en-têtes et calcule le CRC sur chaque unité de données. L'encapsulateur fournit alors à la couche MPEG-2 TS le flot *MPE_flow* à émettre.

[5] La signification des drapeaux TRUE ou FALSE en argument de certaines méthodes sera expliquée plus tard.

```
#define IP_HDR_SIZE 20

int main(int argc, char *argv[]) {
IP_PDU_SIZE=1480 ; NB_PACKETS=1000 ;

// Construction des données, des flots IP, MPE et MPEG-2
bvec input_data=randb(8* NB_PACKETS * IP_PDU_SIZE);
ip.tx_to_IP(input_data, IP_PDU_SIZE);
bvec IP_flow=ip.tx_from_IP();
mpe.tx_to_MPE(IP_flow, TRUE, pdu_ip_size+IP_HDR_SIZE);
MPE_flow=mpe.tx_from_MPE();
bvec TX_DATA=dvb.encode_and_fit_mpeg2(MPE_flow, TRUE);

// Modulation et transmission
… … …

// Décodage, réassemblage et vérification de l'intégrité des paquets
bvec RX_DATA=dvb.decode_and_fit_mpeg2(rx_signal, MPEG2_errors, TRUE);
mpe.rx_to_MPE(RX_DATA, MPE_errors, TRUE);
bvec output_mpe=mpe.rx_from_MPE();
ip.rx_to_IP(output_mpe, IP_errors, TRUE);
bvec output_data=ip.rx_from_IP();
//Affichage des résultats
cout << "Paquets MPEG-2 errones : " << MPEG2_errors << endl << endl ;
cout << "Paquets MPE errones : " <<  MPE_errors << endl << endl;
cout << "Paquets IP errones : " <<  IP_errors << endl << endl ;
}
```

Figure 4-6 : Code source réalisant l'encapsulation de datagrammes IP par MPE

Après modulation et transmission, les paquets MPEG-2 TS, MPE puis IP sont successivement extraits à partir du flux binaire reçu et décodé. Au cours des différents réassemblages, l'intégrité des unités de données est vérifiée grâce aux CRC (pour MPE) et somme de contrôle (pour IP). Lorsque l'unité de données est erronée, celle-ci est effacée et ajoutée à la liste des unités de données perdues de chacun des protocoles ; ces listes ont ici pour nom *MPEG2_errors, MPE_errors*, et *IP_errors*. Ainsi, si une section MPE est déclarée invalide, elle est « rejetée » au niveau MPE. Au moment de la vérification de la somme de contrôle IP, le datagramme est systématiquement reconnu comme étant invalide. La Figure 4-7 montre un résultat obtenu lors de l'exécution d'une simulation.

```
$ ./sim_DVB_IP 3 4
Ec/N0= 2 dB
Paquets MPEG-2 errones : [22 23 35 67 75 76 77 89 97 115 116 139 150 151
153 157 159 160 196 205 209 224 231 232 239 297 330 331 332 334 335 337 347
348 352 376 378 396 397 447 461 465 476 477 478 479 480 481 482 483 485 503
530 531 532 539 543 554 556 557 595 601 602 603 604 605 610 625 635 636 657
660 661 662 690 691 692 693 750 758 768 769 774 775 776 777 778 779 805]

Paquets MPE errones : [2 4 8 9 10 11 13 14 16 18 19 23 24 25 27 28 29 36 40
42 45 48 54 55 56 57 58 61 64 65 67 72 73 74 75 77 79 80 83 84 90 91 92 93
94 97]

Paquets IP errones : [2 4 8 9 10 11 13 14 16 18 19 23 24 25 27 28 29 36 40
42 45 48 54 55 56 57 58 61 64 65 67 72 73 74 75 77 79 80 83 84 90 91 92 93
94 97]
```

Figure 4-7 : Résultat d'une simulation de transmission d'un flux IP par DVB-S

4.3 PREMIÈRES EXPLOITATIONS DU SIMULATEUR

Après la validation des différentes fonctions du simulateur, il a été possible de s'en servir pour étudier les performances de la couche physique, sur les services MPEG-2 TS et IP.

4.3.1 *Étude de performance du codage d'erreur de DVB-S*

Tout d'abord, nous nous intéressons à la performance de la chaîne de codage DVB-S. L'objectif de la simulation suivante est d'obtenir le taux d'erreur binaire après décodage RS, avant l'effacement des paquets erronés, en fonction du E_b/N_0. L'étude est faite pour plusieurs taux de codes convolutionnels de DVB-S (1/2, 3/4 et 7/8). La correspondance entre le rapport E_b/N_0 et le rapport E_c/N_0 donné en paramètre dans le simulateur dépend du produit, ρ, du taux de code convolutionnel et du taux de code RS, selon :

$$\frac{E_b}{N_0}(dB) = \frac{E_c}{N_0}(dB) - 10.\log_{10}\rho \qquad (4.1)$$

Les résultats obtenus sont donnés par la Figure 4-8. La caractéristique importante à noter dans ces courbes est leur pente élevée, passé un certain seuil ; par exemple une augmentation du E_b/N_0 d'environ deux dixièmes de décibels se traduit par une chute du BER après décodage d'un facteur 10. Physiquement, des fluctuations aussi faibles correspondent à un phénomène de scintillations. Aussi, lorsque le système satellite est bien dimensionné et en l'absence de *fading*, ces résultats confirment le fait que les liaisons par satellite DVB-S sont pratiquement aussi fiables que les liaisons des réseaux câblés. À titre indicatif, et en prenant en compte une marge de dégradation de 0.8 dB dans l'implémentation du démodulateur, l'ETSI [ETS97] estime qu'une liaison quasi parfaite ($BER<10^{-10}$ après le décodage RS) est obtenue, avec un taux de code convolutionnel de ¾, pour un E_b/N_0 au moins égal à 5.5 dB.

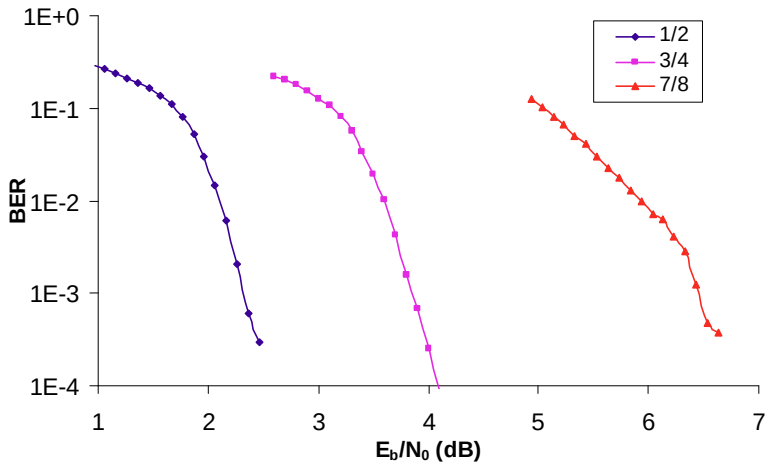

Figure 4-8 : Simulation des performances de la chaîne de codage DVB-S pour les taux de codes convolutionnels 1/2, 3/4, 7/8

Les simulations seront désormais effectuées avec un taux de codage convolutionnel égal à ¾, mais le comportement, observé pour d'autres valeurs du codage, est qualitativement identique.

4.3.2 *Étude de la distribution des rafales d'erreurs de paquets au niveau MPEG-2 TS*

Dans la seconde étude, nous nous sommes intéressés à la distribution des rafales d'erreurs de paquets MPEG-2.

De façon assez étonnante, et ce que malgré le fait que le bruit dans le canal de transmission soit modélisé par un processus stochastique sans mémoire, de nombreuses erreurs de paquets sous forme de rafales apparaissent. Ceci est facilement vérifié lorsque le taux d'erreur résiduel est faible. Par exemple, le cas de la Figure 4-5 est révélateur puisque sur 9 paquets reçus en erreurs, 6 paquets font partie d'une rafale d'au moins deux paquets. Il est également possible

de s'en convaincre si l'on compare la statistique d'occurrence des rafales avec une distribution d'erreurs aléatoires uniformément réparties. Appelons N le nombre de paquets transmis, PER (*Packet Error Rate*) le nombre moyen de paquets MPEG-2 reçus en erreur et L_R la longueur d'une rafale d'erreur exprimée en nombre de paquets successifs erronés. Avec la distribution uniforme, les probabilités d'une rafale de i paquets successifs seraient :

$$\Pr(L_R = i) = PER^i(1 - PER) \tag{4.2}$$

Pour comparer les deux distributions, il suffit de relever le taux d'erreur paquet moyen, obtenu par simulation, puis de l'utiliser comme paramètre pour le modèle décrit par (4.2). Les résultats obtenus sont schématisés sur la Figure 4-9.

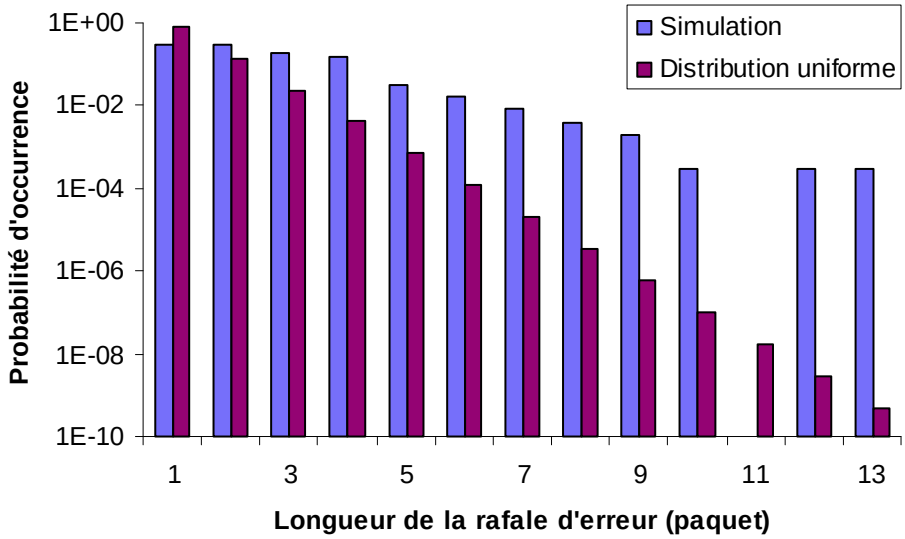

Figure 4-9 : Distribution des rafales d'erreurs de paquets MPEG-2 après décodage RS, avec E_c/N_0=1.5 dB

L'histogramme de la figure précédente démontre que la distribution des erreurs obtenues par simulation ne sont pas réparties uniformément, ce qui est conforme à ce qui a été obtenu précédemment. Dans cet exemple, les rafales d'erreurs de deux paquets ont été aussi fréquentes que les rafales d'un seul paquet. L'explication de ce phénomène est probablement liée au décodage Viterbi qui produit des erreurs en rafale lorsqu'il ne parvient pas à décoder correctement les données. Pour les valeurs de bruit que l'on étudie, il est possible que la taille des rafales d'erreur (binaires) soit si grande que l'entrelacement ne parvienne pas à les casser.

4.3.3 *Impact des performances de la chaîne de transmission DVB-S sur le service de transport de datagrammes IP*

Nous renouvelons le type d'étude mené dans le paragraphe précédent mais au niveau du service IP, par une encapsulation par MPE. Examinons d'abord le cas « idéal » dans lesquels les erreurs bit résiduelles seraient distribuées aléatoirement, ce qui correspondrait au canal BSC. Au cours de la transmission du flux IP, deux paramètres conditionnent les pertes de paquets : le taux d'erreur bit du canal et la longueur des datagrammes IP, L. Compte tenu de l'encapsulation (17 octets ajoutés par unité de données), un datagramme est effectivement reçu si et seulement si la section MPE qui le transporte est exempte d'erreur. Dans cette hypothèse, le taux de rejet moyen de paquets au niveau IP, PLR, est donné par :

$$PLR = 1 - \left(1 - BER^{8(L+17)}\right) \tag{4.3}$$

Les courbes de performances de cette loi sont données en Figure 4-10 (PLR en fonction de la longueur des paquets IP à BER résiduel fixé) et en Figure 4-11 (PLR en fonction du BER résiduel, à longueur fixée).

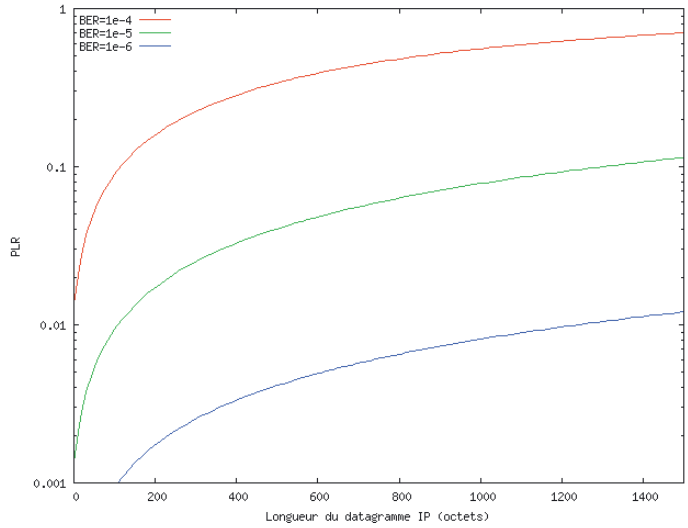

Figure 4-10 : Évolution du taux de pertes de paquets moyen (PLR) en fonction de la longueur des datagrammes IP pour des taux d'erreurs binaires résiduels de 10^{-4}, 10^{-5}, 10^{-6}

Dans la série de courbes de la Figure 4-10, on peut voir que l'augmentation du taux de pertes de paquet est assez rapide lorsque la longueur des datagrammes IP augmente jusqu'à environ 500 octets. Au delà, l'augmentation est plus modérée. Néanmoins, pour un taux d'erreur binaire de 10^{-4} et pour des datagrammes de 1500 octets, le PLR paraît élevé (70%). On voit également que la diminution du BER d'un facteur 10 correspond aussi à une division du PLR dans le même rapport. Cela est confirmé par les résultats de la Figure 4-11, où la linéarité entre le BER et le PLR est mise en évidence tant que le BER est assez petit. Ensuite, le PLR tend asymptotiquement vers 1.

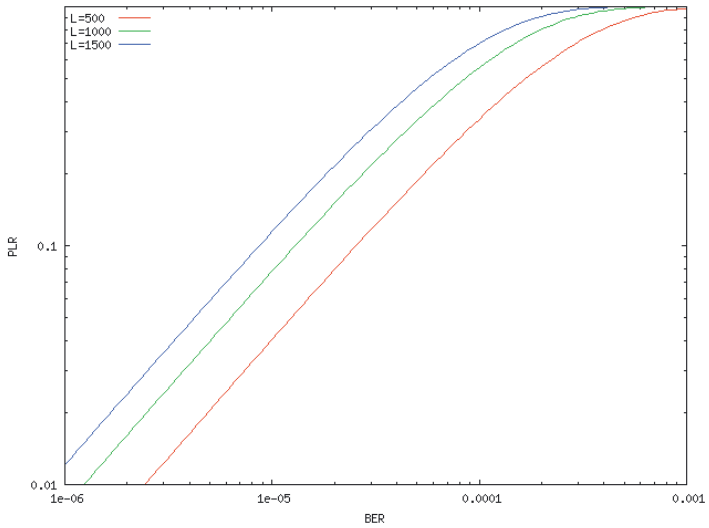

Figure 4-11 : Évolution du PLR en fonction du BER pour des longueurs de datagrammes IP de 500, 1000 et 1500 octets

Nous pouvons maintenant étudier les résultats obtenus par simulation. Dans la partie précédente, la relation entre le BER moyen, après décodage, et le rapport signal à bruit de la liaison satellite, E_c/N_0, a été établie. Nous savons que la distribution des erreurs de paquets au niveau de service de MPEG-2 TS n'est pas uniforme. La question que nous nous posons est de

savoir dans quelle mesure cette distribution influe sur la production d'erreurs de paquets IP. Les erreurs apparaissant sous la forme de rafales, nous nous attendons à ce que les résultats de simulation soient meilleurs que ceux obtenus avec le modèle à répartition uniforme paramétré avec le même PLR. En effet, puisqu'un datagramme IP est encapsulé par plusieurs paquets MPEG-2, la perte d'un seul de ces paquets suffit à la perte du datagramme IP. En revanche, si plusieurs de ces paquets sont perdus, le taux de pertes de niveau IP n'est pas modifié.

Pour établir les courbes de performances avec cette répartition, il suffit de reprendre les résultats de simulation donnés par la Figure 4-8 et d'utiliser l'équation (4.2) pour en tirer le PLR moyen. D'un autre coté, le simulateur fournit directement les statistiques d'erreurs des unités de données à tous les niveaux. La Figure 4-12 et la Figure 4-13 rendent compte des résultats de cette comparaison. On y constate, effectivement, que la proportion de datagrammes IP écartés est plus faible que si les erreurs résiduelles étaient réparties uniformément, malgré le fait que le rejet des paquets au niveau MPEG-2 TS n'ait pas été pris en compte par le modèle à répartition uniforme.

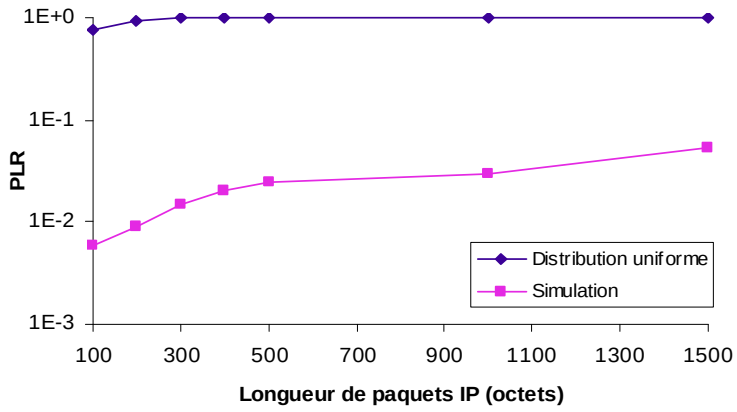

Figure 4-12 : PLR au niveau du service IP, entre les résultats de simulation et un modèle d'erreurs résiduelles uniformément réparties, en fonction de la longueur des datagrammes IP pour E_b/N_0=3.7 dB

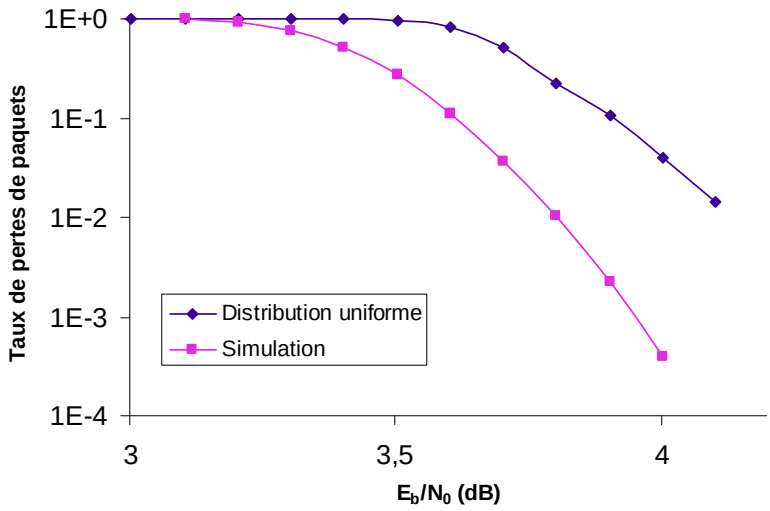

Figure 4-13 : PLR au niveau du service IP, entre les résultats de simulation et un modèle d'erreurs résiduelles uniformément réparties en fonction du rapport E_b/N_0 pour des datagrammes IP de 1500 octets

4.4 INTÉGRATION D'UN CANAL DE TRANSMISSION PERTURBÉ PAR DES AFFAIBLISSEMENTS VARIABLES AU COURS DU TEMPS

Le simulateur intègre une fonctionnalité qui ne sera pas exploitée dans le cadre des travaux présentés. Afin de disposer d'un modèle de canal intégrant les phénomènes de *fading*, nous avons voulu développer une extension du canal AWGN tenant compte de possibles variations temporelles du rapport signal à bruit. Comme nous l'avons vu dans le chapitre 2, les phénomènes à l'origine du *fading* sont complexes et variables. Aussi, l'élaboration d'un modèle de propagation analytique est délicate même si elle reste possible. D'autre part, il s'avère que les performances dépendent fortement des choix précis de certains paramètres du système satellite. Par exemple, le simple fait de doubler le débit de la porteuse fait chuter le rapport signal à bruit de trois décibels (cf. équation (2.5)). Or, d'après les résultats obtenus précédemment, les 3 dB de variations peuvent faire basculer le taux de pertes de paquets de 0 à 100 % si le système opère dans la région critique du E_b/N_0 (autour de 4 dB avec le taux de codage ¾).

Sans aller jusqu'à offrir une modélisation complète de la propagation, nous avons doté le simulateur d'une fonction permettant son intégration future. Pour cela, nous avons supposé que le modèle exploité par le simulateur était capable de prédire l'atténuation du rapport signal à bruit sous la forme de séries d'échantillons temporels, identique à celui utilisé par l'émulateur du système GEOCAST (cf. 3.1.3). Ensuite, d'après ces données, le simulateur se charge de reconfigurer périodiquement le canal. Puisque le simulateur est un système à évènements discrets, il n'y a pas de moyen direct de simuler un évènement à un instant donné.

Toutefois, en faisant l'hypothèse que l'atténuation affectant le canal est constante pendant le temps de transmission d'un paquet IP, une reconfiguration du canal après la transmission d'une séquence de N symboles. Ce nombre dépend du taux de codage employé, de la longueur des datagrammes, du débit binaire de la porteuse et de la fréquence à laquelle les séries temporelles des valeurs d'affablissement du canal sont échantillonnées.

CONCLUSION

Le fonctionnement d'un simulateur de liaison DVB-S et d'encapsulation de datagrammes IP dans le flux MPEG-2 TS a été décrit dans ce chapitre. Son premier objectif consiste à évaluer les performances de la fiabilité de la liaison satellite aux différents niveaux de service. Le simulateur peut fournir l'ensemble des statistiques de pertes des paquets IP, en particulier les taux de pertes moyens, mais aussi les distributions de ces pertes. La principale limitation, intrinsèque à la méthode employée, concerne l'absence de simulation précise de la démodulation, qui ne peut pas rendre compte des problèmes de perte de la synchronisation. Une telle tâche aurait demandé un important travail qui dépasse largement les objectifs de la thèse. Toutefois, si l'on admet que ces problèmes apparaissent à des niveaux de bruits supérieurs à ceux étudiés, l'impact sur les résultats fournis par le simulateur peut être effectivement négligé.

Les premières exploitations du simulateur ont montré, de façon assez inattendue, que les distributions de pertes de paquets MPEG-2 TS et IP ne sont pas uniformes dans un canal AWGN à affaiblissement constant. En conséquence, les performances atteintes à ces niveaux de service ne peuvent être uniquement prédites par le seul taux d'erreur bit moyen obtenu après décodage sur la liaison satellite, en particulier lorsqu'il est assez élevé.

En perspective, le support de futurs développements pour le simulateur est envisageable. Ces développements pourraient servir à simuler d'autres types de réseaux sans-fil (DVB-RCS, réseaux mobiles, etc...). Ils pourraient aussi permettre d'améliorer la qualité de la modélisation, en particulier de la couche physique. Par exemple, des modèles de propagation de pertes plus élaborés, faisant apparaître l'effet de conditions météorologiques réalistes pourraient être intégrés dans les simulations.

BIBLIOGRAPHIE

[IT++] http://itpp.sourceforge.net/

[Ets97] ETSI EN 300 421, "Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz satellite services", v1.1.2, août 1997.

[NS2] http://www.isi.edu/nsnam/ns/

[OPN] http://www.opnet.com/

Chapitre 5

Conception d'une architecture fournissant un service de livraison en mode binaire

INTRODUCTION

Au cours du chapitre précédent, il a été confirmé que les liaisons satellite DVB-S, quand elles n'étaient pas perturbées, offraient un niveau de fiabilité tel qui pouvait être comparable à celui des réseaux câblés terrestres. Pourtant, lorsque des erreurs surviennent et que le service satellitaire est exploité pour la diffusion à destination de groupes, la gestion des problèmes devient rapidement critique. Par exemple, la correction d'erreur (par anticipation ou par retransmission) entraîne obligatoirement la dégradation de plusieurs facteurs intervenant au cours du transfert : augmentations des données transmises, des délais de livraisons, des charges de calcul pour la source, etc. Finalement, les différents facteurs qui réduisent l'efficacité de la transmission ont un impact plus important que pour les communications terrestres point à point classiques car elles touchent simultanément beaucoup plus de récepteurs.

Par ailleurs, on doit noter l'absence de contrôle sur le service de livraison de DVB-S. Même s'il existe en réalité un contrôle minimal (indication d'une erreur de transport dans l'en-tête des paquets MPEG-2 TS), en cas de défaillance, la norme ne prévoit ni comment des solutions réactives pourraient être mises en œuvre, ni comment signaler les erreurs aux couches supérieures. Dans le contexte original de la télévision par satellite pour lequel la bande Ku était privilégiée en Europe, ce choix se justifiait en raison de l'incapacité des terminaux à émettre et pour des systèmes dimensionnés avec des bilans de liaisons plus favorables.

L'avènement des systèmes bidirectionnels comme DVB-RCS n'a que partiellement changé les choses. La saturation des bandes de fréquences traditionnelles impose progressivement l'utilisation de la bande Ka, qui est aussi plus propice aux atténuations. Confronté à la limitation de la puissance d'émission des petits terminaux satellite de type résidentiels, DVB-RCS a ainsi dû intégrer la prise en compte de la qualité du signal dans son système. Grâce à la présence d'une table (*Correction Message Table*), le NCC peut informer le terminal DVB-RCS de la mesure du E_b/N_0 (ou éventuellement de la correction à effectuer) afin d'optimiser sa puissance d'émission. Une telle signalisation sur le lien aller est toutefois inexistante. Parmi les diverses techniques FMT déjà évoquées, seule la gestion de puissance semble apte à être déployée facilement car elle n'impose aucune reconfiguration de l'interface air. La solution possède toutefois ses limites, car s'il est toujours possible de mesurer le rapport E_b/N_0 à des intervalles de temps réguliers par une méthode dédiée, encore faut-il que ces mesures soient réalisées sur l'ensemble de la zone de couverture du système.

Dans les couches de niveau liaison des réseaux terrestres, le niveau de performance est tel que les erreurs bits, induisant les rejets de paquets, ont très peu de conséquences sur le service de transmission. Il y a deux raisons à cela : d'une part, les pertes de paquets dues aux corruptions sont des évènements très rares (BER typiques inférieurs ou égal à 10^{-10}), et d'autre part, le temps induit par une retransmission est faible (quelques dizaines de millisecondes au maximum). Le bénéfice de ce procédé est d'alléger de manière importante les traitements sur les stations d'extrémité, en leur évitant d'avoir à détecter les unités de données corrompues. Ceci était particulièrement important il y a quelques années lorsque les ressources CPU des stations étaient limitées.

Lorsque l'on considère aujourd'hui ce mécanisme de rejets de paquets dans les communications multipoints par satellite, les arguments avancés pourraient être remis en cause. Par exemple, si un datagramme IP (ou l'un des paquets MPEG-2 qui le transporte) est reçu avec quelques octets erronés, le paquet entier est rejeté. Dans le cas où le datagramme est très long (1500 octets maximum généralement avec Ethernet, mais extensible avec d'autres réseaux comme Gigabit Ethernet), l'impact d'un tel rejet semble important dans un environnement où l'utilisation des ressources est si coûteuse. D'autre part, lorsqu'il est intéressant d'utiliser les plus longs datagrammes possibles, la probabilité qu'ils soient rejetés à cause de corruptions augmente. Enfin, si une partie des paquets est rejetée par un seul terminal satellite et que des retransmissions sont nécessaires, les performances pour tout le groupe seront affectées.

Au-delà de ces considérations, il faut mentionner que le service de livraison traditionnel de datagrammes IP offert par DVB-S fait apparaître un réel manque de coordination entre les couches hautes (essentiellement la couche transport) et basses (couches liaison et physique) du processus de communication. Dans le contexte général d'Internet, le principe de bout en bout consiste à positionner les fonctions de contrôle du réseau aux niveaux des extrémités de la communication (*i.e.* au niveau des stations hôtes, dans la couche transport, par le biais notamment du protocole TCP). Il est nécessaire à ces mécanismes de contrôles de recueillir des informations sur l'état du réseau pour fonctionner correctement. Cependant, la communication entre le réseau et la couche transport est par principe de base très limitée. Dans ce cadre, la perte de paquet est utilisée comme symptôme d'une congestion de réseau. Le mécanisme de contrôle de congestion du protocole de transport peut ainsi fonctionner sans information directe provenant de la couche réseau, simplifiant ainsi le fonctionnement global du système. Lorsque l'hypothèse du taux d'erreur bit très faible n'est plus vérifiée, le taux d'erreur paquet important issu de la détection des erreurs bit par les différentes couches induit un mauvais fonctionnement des mécanismes de contrôle de congestion.

En dépit de ces limites, il est pourtant délicat d'éviter les rejets de paquets en cas de corruptions. Le principal problème que cela pourrait engendrer est le traitement de paquets pour lesquels les données de signalisation, véhiculées dans les en-têtes et en-queues, seraient erronées. Les conséquences de tels traitements pourraient alors être particulièrement néfastes (par exemple un mauvais routage ou démultiplexage, etc…). L'objectif, au travers de ce chapitre, est de démontrer que dans le contexte des normes DVB par satellite, cette idée est pourtant réalisable si un ensemble de dispositions est entrepris au travers des différentes couches de l'architecture. Les bénéfices de cette architecture seront ensuite discutés dans le chapitre 6 pour différents services de fiabilité demandés par l'application.

Le reste de ce chapitre est structuré de la façon suivante. Dans un premier temps, une architecture offrant à la couche de transport la livraison d'un maximum d'unités de données (potentiellement erronés) sera décrite. Un tel service sera appelé *service de livraison en mode binaire* ; il sera étudié à plusieurs niveaux de la communication. L'architecture se base en grande partie sur un procédé de fiabilisation spécifiques aux en-têtes, appelé *Multi Protocol Header Protection* (MPHP). Puis, dans le cadre de l'encapsulation ULE, les problèmes de mise en œuvre de l'architecture seront données. Ils seront à l'origine d'une proposition d'adaptation du réassemblage des

SNDUs. Les limites de la méthode seront alors discutées et une première évaluation sera donnée. Ces défauts seront à l'origine de la conception d'une architecture de transmission idéalisée, capable d'optimiser l'exploitation des données correctement transmises.

5.1 DESCRIPTION DE LA PROPOSITION

Dans la suite du mémoire, l'utilisation simultanée des mécanismes proposés sera appelé *Architecture Modifiée* (AM), par comparaison avec l'*Architecture Conventionnelle* (AC) où toutes les unités de données détectées erronées sont rejetées dès la détection de l'erreur. AM sous-tend la présence de trois principales actions : la limitation de la vérification des unités de données, l'utilisation d'un protocole spécifique de protection d'en-tête (MPHP), et l'adaptation du processus de réassemblage des SNDU. Un quatrième mécanisme, optionnel, permettra d'améliorer les performances d'AM : la compression d'en-têtes. Son intégration dans la proposition sera appelée *Architecture Modifiée avec Compression* (AMC).

5.1.1 *Limitation de la vérification de l'intégrité des unités de données*

Un moyen pour rentabiliser le coût de la transmission par satellite est d'économiser la quantité de données à envoyer. Pourtant, avec le mécanisme classique de rejet des paquets, il est possible qu'une quantité d'information soit transmise plusieurs fois par le système satellite et ce, même si une partie très importante de cette information est correctement reçue par le récepteur. Afin d'éviter cela, une première idée est de supprimer simplement toute vérification d'erreur. En l'état actuel, cette action irait à l'encontre des fondements même d'une très grande majorité d'architectures réseau, mais aussi des applications qui les exploitent. Entre autres problèmes, les données de signalisations des protocoles sont traditionnellement placées dans les en-têtes (ou en-queues) des unités de données, et des corruptions sur ces données pourraient avoir des conséquences inattendues.

Plutôt que d'abandonner ce principe, nous proposons de conserver les vérifications des unités de données mais *en les limitant* aux seuls en-têtes et en-queues des unités de données. L'opération correspondante reviendrait alors à modifier légèrement le calcul existant des CRC (ou des sommes de contrôles) pour restreindre leur action à des éléments considérés comme importants (p.ex. les données de signalisation évoquées ci-dessus). Si, à la réception de l'unité de données, des erreurs étaient détectées sur ces éléments, l'unité est supprimée. Si des erreurs se produisent dans la charge utile, le mécanisme n'en est pas conscient et l'unité de données est restituée avec les erreurs à la couche supérieure.

Pour être efficace, cette action devrait être déployée sur l'ensemble des protocoles d'une pile considérée. Dans le contexte des réseaux IP sur DVB, la difficulté principale de mise en application de cette action se situe au niveau des récepteurs MPEG-2 TS et du réassemblage des SNDUs. Il faut accepter que tous les paquets MPEG-2 TS, y compris ceux dont le décodage a échoué, puissent servir au réassemblage des SNDUs. Le niveau supérieur, IP, ne pose pas de problème particulier puisque la vérification de l'intégrité des datagrammes (dans IPv4) n'est déjà faite que sur l'en-tête. Avec IPv6, les choses sont aussi simples car il n'y a aucun contrôle d'intégrité (le contrôle sur l'en-tête IPv6 est censé être fait par la couche supérieure). Au niveau des couches supérieures (essentiellement le transport) localisées sur les stations d'extrémités, ce type de modification serait en théorie assez simple puisque les protocoles y sont implémentés sous formes logicielles. Si l'on prend en compte le fait que les protocoles de transport multipoints reposent en réalité sur les capacités de communication des *sockets* UDP, une adaptation supplémentaire pourrait avoir lieu. Notons cependant que le service proposé par ces protocoles ne correspond plus dans ce cas à celui des protocoles classiques TCP ou UDP.

L'évaluation de cette première solution a été réalisée par simulation. La modification de la portée des CRC, par exemple pour la vérification des données des sections DSM-CC, est réalisée grâce aux arguments booléens fournis pour certaines méthodes. La comparaison de cette méthode, que nous désignerons par *architecture améliorée primaire* (AAP), avec l'architecture AC est illustrée dans la Figure 5-1, pour un flux de datagrammes IP de 1500 octets encapsulé par MPE, en fonction du rapport E_c/N_0. Elle a été effectuée au niveau de l'interface de service IP (pour l'architecture AC et AAP) et au niveau de l'interface de service MPE (pour l'architecture AAP).

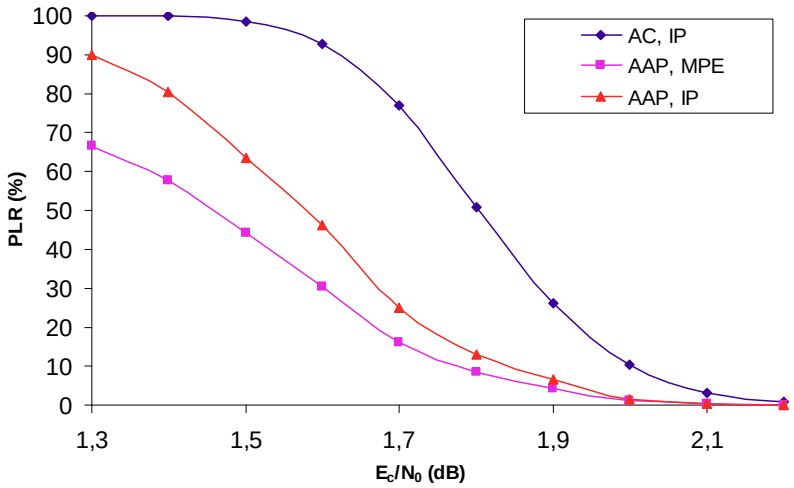

Figure 5-1 : Comparaison des PLR entre l'architecture AC au niveau du service IP et l'architecture AAP, aux niveaux MPE et IP

Les gains obtenus par l'architecture modifiée sont encourageants lorsque mesurés au niveau du service MPE, mais une nette dégradation se produit après le traitement de la couche IP. Ceci indique donc que l'intégrité des 20 octets d'en-tête IP n'est pas toujours vérifiée malgré la détection et le rejet des paquets erronés au niveau inférieur. D'autre part, l'évaluation de l'architecture modifiée ne rend pas compte de corruptions de l'en-tête MPEG-2, ni de celles d'en-têtes de protocoles de couches supérieures. Dans plusieurs cas de figure, ce type de corruption pourrait probablement empêcher la réception du paquet par le récepteur MPE/ULE. Enfin, à ce stade, nous ignorons à quel point les paquets survivants sont erronés et s'ils peuvent ou non contribuer à améliorer le service offert à l'application. En conséquence, il semble que l'architecture, envisagée telle quelle, ne soit pas suffisamment intéressante pour la mise en œuvre du service de livraison en mode binaire.

5.1.2 *MPHP (Multi Protocol Header Protection)*

La recherche de l'explication de ce phénomène amène à considérer l'hypothèse selon laquelle les en-têtes, malgré leur taille relativement faible, seraient statistiquement souvent erronés, entraînant alors le rejet des unités de données que l'on souhaite empêcher. Dans la pile de protocole étudiée, les données de signalisations IP et MPE (respectivement IP et ULE) représentent 296 bits (resp. 288 bits). Dans ces conditions, pour un taux d'erreur binaire résiduel de 10^{-3} après le décodage RS, la probabilité moyenne de rejet des datagrammes IP vaut environ 75%, ce qui semble corroborer l'hypothèse formulée.

Dès lors, une solution de codage correcteurs d'erreurs, portant uniquement sur les données de signalisations (en-têtes et en-queues des différentes unités de données) paraît appropriée. La technique dans laquelle les informations sont codées différemment suivant la sémantique des données (codage par objet, par type d'image, etc…) est appelée *Unequal Error Protection* (UEP). De nombreuses applications tirent parti de cette technique, comme par exemple la fiabilisation de

flux vidéo MPEG en plusieurs niveaux de protection selon le type d'images (*Intra*, *Bidirectionnelle* ou *Prédite*).

5.1.2.1 Protection spécifique d'en-têtes

Le mécanisme MPHP protège les données de signalisation contenues dans les protocoles en ajoutant des informations de parités dans le flux. Les données additionnelles sont calculées par le codage MPHP en connaissant le nombre d'octets d'en-tête et d'en-queue à protéger pour chaque SNDU. Afin de garantir un maximum d'efficacité, MPHP doit être mis en œuvre juste avant l'encapsulation des données en flux MPEG-2 TS. Après cette étape, l'octet *Payload Pointer* est lui aussi protégé, car sa corruption entraînerait systématiquement la perte de toute la SNDU. Ce codage est appelé *codage PUSI* pour le différencier de celui de MPHP. Les détails de ce codage seront donnés en 5.1.2.6.

Lorsque des erreurs subsistent après le codage du niveau physique, le récepteur MPEG-2 DOIT délivrer les unités de données s'il le peut à MPHP (p.ex. si le PID du paquet MPEG-2 n'est pas corrompu) sans les modifier. Après décodage PUSI, MPHP vérifie l'intégrité des données MPHP, en les corrigeant si nécessaire.

5.1.2.2 Choix du codage MPHP

Puisque les données arrivent par paquet au niveau du récepteur MPHP, le choix d'un code en bloc semble adapté. D'autre part, si ce code parvient à corriger des erreurs, cela signifie que le codage RS de DVB a été insuffisant, et a transmis les données sans les modifier. Le profil d'erreur des données alimentant le décodeur MPHP est donc celui résultant du décodage Viterbi. Une fois encore, le codage RS apparaît comme le plus approprié. Ici, trois de ses caractéristiques peuvent être exploitées :

- la propriété MDS garantit que le codage est optimal ;
- les paramètres du code sont facilement configurables (*i.e.* en le raccourcissant).

Concernant ce dernier aspect, le choix du couple (n, k) est dicté par deux considérations : la robustesse aux erreurs (réglable par l'intermédiaire du taux de codage, k/n) et la quantité de données à coder par paquet (déterminant le paramètre k). Le taux de codage doit être suffisamment faible, par rapport au code (n=204, k=188) de DVB pour garantir un taux d'erreur faible malgré la présence importante d'erreurs. Par exemple, un taux de codage de 1/3 peut convenir. La sélection de k dépend ensuite d'un choix de mise en œuvre indiquant comment sont insérées les données de parité dans le flux.

5.1.2.3 Positionnement des données de parité MPHP

À cet effet, deux choix apparaissent. La première alternative proposée consiste à fixer au préalable le code choisi. Par exemple, avec le taux de codage précédent, on fixe (n=255, k=85). Ensuite, MPHP sélectionne les données à encoder au sein de la SNDU d'après le nombre d'octets d'en-têtes à protéger, N_H, et d'en-queue, N_T. Ces données servent à alimenter une mémoire tampon qui fournit au codeur RS de MPHP des blocs de k octets. Les h octets de parité produits peuvent alors être transmis sous la forme d'une nouvelle SNDU contenant exclusivement des données MPHP. Pour le décodage, un certain nombre de SNDUs doit être stocké au niveau d'une autre mémoire tampon avant de pouvoir être décodés. Le décodage de ces données est déclenché dès la réception des n symboles du mot-code de MPHP. Le nombre de paquets à stocker avant le décodage dépend donc des paramètres du code. La limite de cette première solution est qu'en cas de perte de la SNDU ajoutée par MPHP, les autres SNDUs ne sont plus protégées.

Dans la seconde alternative, le code est choisi tel que k soit égal à la somme de N_H et N_T. Avec les protocoles usuels, cette somme reste normalement inférieure à 85, ce qui permet de protéger chaque SNDU avec un seul mot-code (avec le taux de code choisi précédemment). Ensuite, après avoir posé Y=255-3k, le code RS de paramètre (k+Y, n=255) est raccourci pour obtenir le code (k, n=255-Y=3k). Pour chaque SNDU à coder, le codage lit ses N_H premiers octets et ses N_T derniers octets. Après le calcul, les données de parités (2.(N_H+N_T) octets par SNDU) sont ajoutées à la suite de la SNDU, ce qui forme un paquet MPHP. Ce paquet est alors encapsulé puis multiplexé dans le flux MPEG-2 TS. Par rapport à la première méthode, celle-ci a l'avantage d'une mise en œuvre plus simple car le positionnement des données de parité conserve le découpage logique des SNDUs (devenus paquets MPHP). En outre, le décodage par paquet supprime la limitation de la première alternative. Son seul défaut réside dans le fait que les mots de code utilisés sont moins longs que ceux de la première approche, synonyme d'une robustesse aux erreurs plus faible.

La forme des flux obtenus après encodage MPHP suivant l'une ou l'autre méthode est illustrée dans la Figure 5-2.

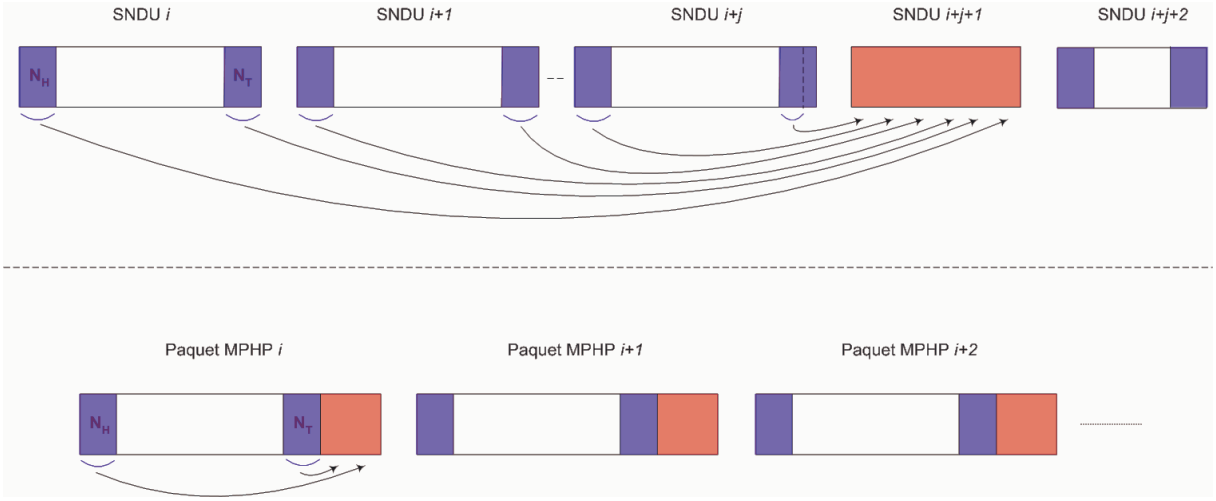

Figure 5-2 : Positionnements possibles des données de parité de MPHP

5.1.2.4 Configuration des paramètres de MPHP

Pour réaliser le codage, MPHP doit être configuré afin d'identifier le couple (N_H, N_T), qui servira à ajuster (n, k). Entre autres solutions, il est possible de reconnaître les SNDU qui font parti d'un même flux logique, et de traiter ce flux par son *profil*. Ici, un profil est défini comme la combinaison des couches protocolaires du plan utilisateur encapsulant la SNDU. L'efficacité de MPHP sera d'autant plus grande que le profil utilisé sera précis (par exemple, UDP/IPv4/ULE). En raison du nombre de combinaisons possibles, et de la variabilité de la taille d'en-têtes de certaines unités de données d'un même flux, il peut être impossible de déterminer les tailles exactes des en-têtes à protéger. Dans ces conditions, si n_h et n_t sont les plus petites tailles possible des en-têtes et en-queue du flux, MPHP doit sélectionner, pour ce flux, le profil dont les paramètres N_H et N_T sont les plus proches possibles de n_h et n_t, mais sans les dépasser. Quelques profils sont donnés à titre d'exemples dans le Tableau 5-1.

Profil	N_H	N_T
ULE	4	4
IPv4/ULE	24	4
IPv6/ULE	44	4
UDP/IPv4/ULE	32	4
RTP/UDP/IPv4/ULE	44	4

Tableau 5-1 : Exemples de profils permettant la configuration de MPHP

Le dernier problème qui se pose ici est de savoir comment associer une SNDU à un profil. Pour éviter la transmission d'une signalisation indiquant cette correspondance, ceci peut être entrepris implicitement. Par exemple, un critère de reconnaissance suivant l'adresse IP de destination du datagramme peut être adopté : adresse précise, plage d'adresses, ou condition mathématique sur l'adresse. À l'émission, les SNDUs ne pouvant pas être corrompues, l'adresse IP est directement lue dans le datagramme lors de l'encapsulation ULE. À la réception, l'adresse est reconnue par le biais du PID du dernier paquet MPEG-2 TS servant au déclenchement du décodage MPHP (cf. 5.1.3). Ceci impose que le PID utilisé pour la transmission du flux ne soit pas partagé avec un flux ne désirant pas le service de livraison en mode binaire.

5.1.2.5 Conditions d'activation de MPHP

MPHP ne doit être activé que pour la transmission de certaines SNDU afin de ne pas perturber les flux d'autres services qui ne s'accommoderaient pas des corruptions au sein de leurs unités de données. Comme précédemment, le critère de sélection peut être basé sur l'adresse IP de destination et sur un numéro de port pré-configuré.

De plus, si le système satellite permet la mesure d'un taux d'erreur moyen de la liaison satellite, il pourrait être intéressant de n'activer MPHP que lorsque les conditions de propagation deviennent mauvaises (dépassement d'un seuil critique du E_c/N_0) en raison de la consommation supplémentaire en bande passante du mécanisme.

Enfin, dans le but d'éviter la remontée et le traitement au niveau du transport de messages NACK corrompus, par exemple si la source ne diffusait pas les données par l'intermédiaire de la *gateway* mais par un simple terminal satellite RCST, il serait judicieux que ces messages utilisent un numéro de port sur lequel l'architecture MPHP n'est pas activée.

5.1.2.6 Choix du codage PUSI

Afin de prévenir la corruption de l'octet *Payload Pointer* pour retrouver la longueur correcte des paquets MPHP, il peut être protégé par un code spécifique. Un code RS sur $GF(2^8)$ ne convient pas ici car un seul symbole d'information par mot de code est insuffisant. On peut néanmoins choisir un code RS plus adapté, construit sur $GF(2^4)$, dans lequel les symboles sont de 4 bits. Pour obtenir un taux de codage équivalent à celui du codage MPHP (1/3), le code retenu est le code raccourci (n=6, k=2, d=5) du code (n=15, k=11, d=5). Ses performances sont telles que pour un taux d'erreur binaire résiduel inférieur ou égal à 10^{-2}, la probabilité que le pointeur soit corrompu est inférieure à 0.09%. Les données de parité, après codage, sont ensuite ajoutées directement après le pointeur.

5.1.3 ***Adaptation de l'encapsulation et du réassemblage : ULE « modifiée »***

Dans cette partie, nous envisageons une adaptation de l'encapsulation ULE et surtout du processus de réassemblage au niveau du récepteur MPEG-2 et ULE. Nous supposerons que

l'encodage MPHP est réalisé selon la seconde méthode de positionnement où la parité est concaténée à la fin des SNDUs.

Enfin, la méthode proposée suppose une encapsulation dans MPEG-2 TS par un mécanisme équivalent à *Section Packing*.

5.1.3.1 Opérations effectuées coté émetteur MPHP

D'abord, l'encapsulation des datagrammes IP est réalisée exactement comme ULE, à la différence près que le calcul du CRC de la SNDU ne porte que sur son en-tête. Les données sont ensuite traitées par MPHP qui construit un nouveau flux d'unités de données (*i.e.* les paquets MPHP). Ces paquets sont ensuite encapsulés dans le flux MPEG-2 TS avec un PID indiquant l'utilisation de MPHP. Pour les paquets MPEG-2 devant contenir le début d'une SNDU, le pointeur PUSI et les données de parité du codage PUSI sont ajoutés immédiatement après l'en-tête MPEG-2 et avant le commencement de la nouvelle SNDU.

Dans certains cas, il faut empêcher le commencement d'une nouvelle SNDU dans le paquet MPEG-2 TS. Ces cas correspondent à la situation où à cause du pointeur PUSI et des données de parité du codage PUSI, le premier octet de la nouvelle SNDU ne peut plus être placé dans le paquet. Par exemple, avec le codage PUSI l'encapsulation (partielle) de deux SNDUs dans le même paquet MPEG-2 TS n'est possible que si la première SNDU se termine avant le 180ème octet de la charge utile. Dans le cas contraire, ou lorsque il n'y a plus de paquet MPHP à transmettre, un mécanisme de bourrage doit compléter la fin du paquet MPEG-2 TS. Comme pour l'encapsulation ULE classique, le bourrage est indiqué en transmettant une SNDU spéciale commençant par 0xFFFF, et comportant autant d'octets à 0xFF que nécessaires pour compléter le paquet MPEG-2 TS. De plus, dans cette méthode, un moyen supplémentaire doit permettre le déclenchement immédiat du décodage MPHP sans attendre la réception du paquet MPEG-2 suivant. Cela couvre deux cas : soit le paquet MPEG-2 TS contient la fin d'un paquet MPHP et une SNDU de bourrage, soit le paquet MPHP se termine exactement à la fin du paquet MPEG-2 TS.

Nous proposons l'utilisation d'un bit de l'en-tête MPEG-2 TS, *Priorité*, modifié au cours du processus d'encapsulation pour permettre le déclenchement du décodage MPHP à la fin du traitement du paquet MPEG-2 TS côté récepteur. La signification de ce bit dépend alors de la valeur du drapeau PUSI, comme indiqué dans le Tableau 5-2.

PUSI	Priorité	Signification
0	0	La charge utile du paquet MPEG-2 TS transporte une partie d'un seul paquet MPHP.
0	1	Le dernier octet de la charge utile du paquet MPEG-2 TS correspond à la fin du paquet MPHP. Le décodage MPHP sera déclenché après la réception de ce paquet MHP.
1	0	La charge utile MPEG-2 TS contient un nouveau paquet MPHP. Si le *Payload Pointer* est différent de 0x00, le paquet MPEG-2 TS contient également la fin d'un paquet MPHP précédent. Le décodage MPHP sera déclenché après la réception du premier paquet MPHP.
1	1	La charge utile du paquet MPEG-2 TS contient la fin d'un paquet MPHP et une SNDU de bourrage. Le décodage MPHP sera déclenché après la réception de ce paquet MHP.

Tableau 5-2 : Gestion du bourrage et du déclenchement du décodage MPHP

5.1.3.2 Opérations effectuées coté récepteur MPHP

Puisque MPHP repose avant tout sur le service de livraison de MPEG-2, la perte de paquet MPEG-2 TS doit être envisagée ; elle peut par exemple se produire si le premier octet de synchronisation ou si le PID (13 bits) est corrompu. Dans tous les cas où le récepteur MPEG-2 TS n'est pas en mesure de fournir le paquet à la couche réceptrice MPHP, le réassemblage doit le prendre en compte. Lorsque une rupture de séquence de *p* paquets est détectée entre le paquet MPEG-2 TS courant et le paquet précédent (grâce au champ compteur de continuité de MPEG-2), des données « fantômes » sont introduites (*p**184 octets) à la suite du paquet MPHP en cours de réassemblage pour éviter le rejet de ce paquet MPHP et du paquet MPHP suivant. Le détail des opérations effectuées pour le réassemblage des paquets MPHP est donné par la Figure 5-3. Les tests réalisés au cours de ce processus sont explicités ci-dessous.

- Test (1) : après réception du paquet MPEG-2 TS, le récepteur détermine si le PID correspond à un flux transmis avec MPHP. S'il ne l'est pas, le paquet est traité de façon classique. Lors de ce traitement, l'échec d'un décodage RS (drapeau *Erreur de transport* levé) doit entraîner le rejet du paquet.
- Tests (2a) et (2b) : examens du drapeau PUSI de l'en-tête du paquet MPEG-2 TS.
- Test (3) : le récepteur examine si le champ *Compteur de continuité* de l'en-tête du paquet MPEG-2 TS révèle une rupture de séquence. Si c'est le cas, les données fantômes sont insérées.
- Test (4) : le décodage PUSI est effectué. Si le mot-code trouvé est valide, le traitement peut se poursuivre. Sinon, le paquet MPHP en cours de réassemblage est rejeté.
- Tests (5a) et (5b) : suivant la valeur prise par le drapeau PUSI, les données situées après l'octet *Payload Pointer* sont extraites du paquet.
- Test (6) : si le *Payload Pointer* vaut 0x00 (cf. Tableau 5-2), le paquet MPEG-2 courant ne contient pas la fin d'un paquet MPHP en cours de réassemblage ; il n'y a alors pas lieu d'effectuer un décodage MPHP.
- Test (7) : le paquet MPHP est décodé. Si le mot-code trouvé est valide, l'intégrité de l'en-tête et de l'en-queue de la SNDU est certifiée. Cette dernière peut alors être délivrée au récepteur ULE modifié. Sinon, le paquet MPHP doit être rejeté.

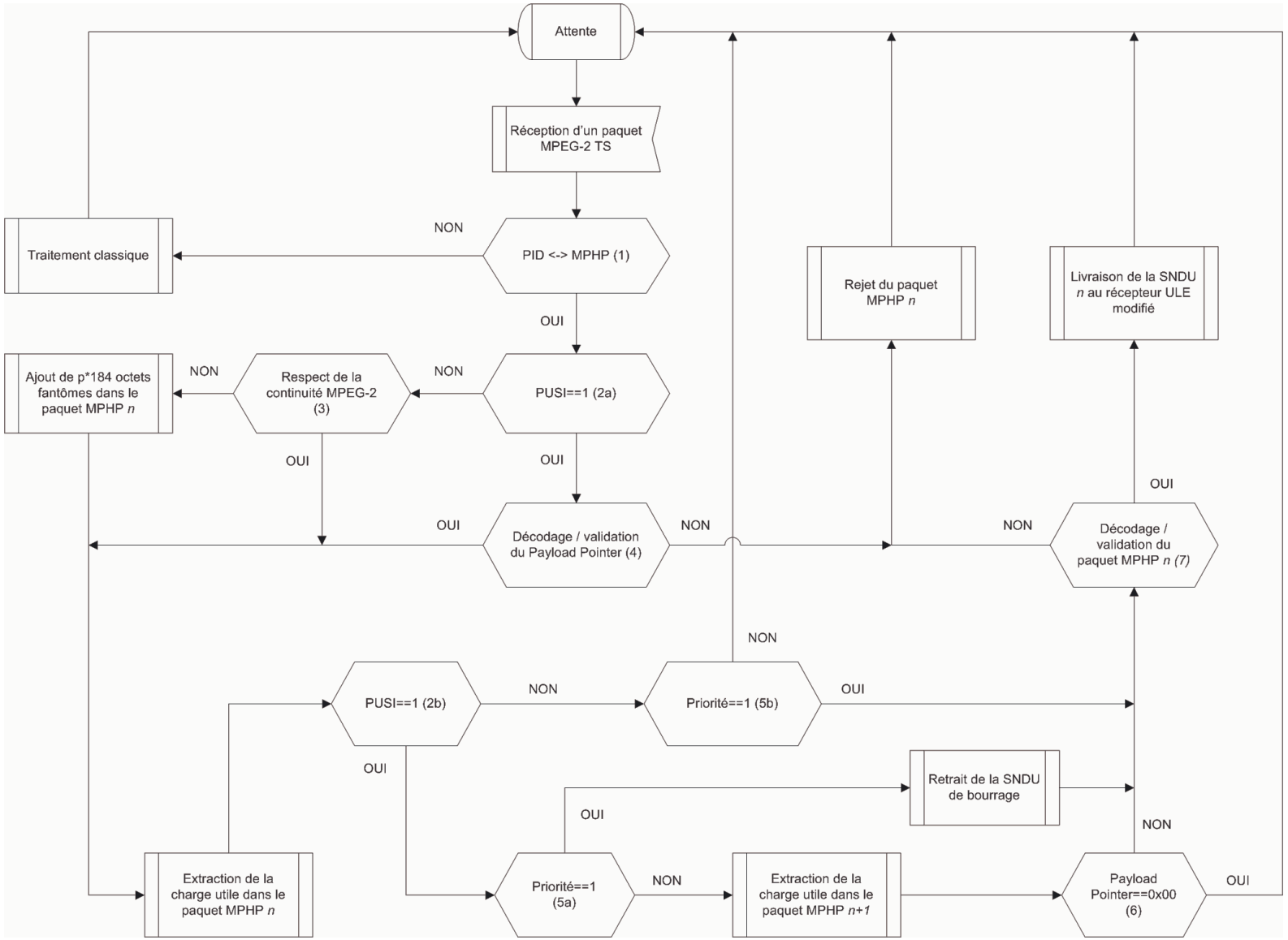

Figure 5-3 : Détails des opérations de réassemblage des SNDU avec MPHP

5.1.3.3 Faisabilité et limites de la méthode

La méthode présentée pour l'encapsulation/réassemblage impose des remarques supplémentaires complétant les hypothèses envisagées, et des limitations doivent être soulignées.

5.1.3.3.1 Pertes de paquets MPEG-2 TS

Dans certains cas (corruption de l'octet de synchronisation ou du PID, cf. 5.1.3.3.4) un paquet MPEG-2 TS ne parvient pas jusqu'au test (1). Alors, si le paquet contient deux SNDUs, le réassemblage et le décodage MPHP de chacune d'elles seront impossibles et elles seront perdues. La conséquence de ces pertes est une réduction ponctuelle de l'efficacité de la transmission, mais le service de transport sera rétabli après la réception de la SNDU suivante.

5.1.3.3.2 Usage des services de cryptage et de priorité de MPEG-2 TS

De meilleurs résultats seraient obtenus par la méthode proposée si le champ *Scrambling Control* restait inutilisé par le système. Une corruption sur ces données serait alors sans effet. Dans le cas contraire, une corruption impliquerait une mauvaise interprétation de la charge utile du paquet et conduirait à des erreurs dans le datagramme IP réassemblé.

De même, le service de priorité ne DOIT pas être utilisé puisque la méthode d'encapsulation fait usage ce bit pour initier le décodage MPHP. Si ce service était nécessaire malgré tout, l'encapsulation pourrait être supportée grâce à l'un des deux bits inutilisés de *Scrambling Control*.

Dans les cas où les deux services ne pourraient plus être réalisés par MPEG-2, un autre niveau de la communication devrait les supporter (cf. 5.2).

5.1.3.3.3 Corruption des données des drapeaux PUSI, Priorité ou du champ Compteur de continuité

Dans le déroulement des opérations, la fiabilité des drapeaux *PUSI*, *Priorité*, et du champ *Compteur de continuité* est supposée. Une erreur sur ces données pourrait entraîner la perte du paquet MPHP en cours de réassemblage, et, selon les cas, celle du paquet suivant.

Une solution pour limiter ce problème consisterait à éviter de traiter les paquets trop sévèrement touchés. Une telle action pourrait par exemple être réalisée en prenant en compte les informations de confiance du décodeur Viterbi à décisions souples. Ces informations, généralement non exploitées par les mécanismes « réseaux », pourraient alors faciliter la tâche du récepteur. En dessous d'un facteur de confiance trop faible, les paquets MPEG-2 TS seraient rejetés avant le test (1).

5.1.3.3.4 Corruption du PID

Si le PID reçu est inexistant dans la table locale du récepteur MPEG/DVB-S, le paquet est simplement effacé et après la réception du paquet MPEG-2 TS suivant, la rupture de séquence est détectée. Au contraire, si un paquet comportant un PID associé au mécanisme MPHP est corrompu en un autre PID existant, le décodage RS le détecte avec une probabilité élevée. Si ce second PID n'est pas associé à un second flux exploitant MPHP, il est rejeté. Dans les autres cas, le paquet d'un autre canal logique MPEG-2 TS sera pris pour une partie d'un paquet MPHP. Après la tentative de décodage de ce paquet MPHP, il sera rejeté.

5.1.3.3.5 Conditions sur les tailles de datagrammes IP

L'insertion des données fantômes suppose que les tailles des paquets MPHP est supérieure à 183 octets, ce qui est compatible avec les tailles usuelles des unités de données issues des

protocoles de transport multipoints pour des applications de types transfert de fichiers ou diffusion de média. Dans le cas contraire, la méthode deviendrait inadaptée car il serait impossible de traiter les cas où la fin de deux paquets MPHP se trouverait dans la charge utile d'un seul paquet MPEG-2 TS. On peut envisager de contourner le problème en encapsulant une SNDU de bourrage à la suite de tout paquet MPHP inférieur à 183 octets, mais au détriment de l'efficacité de la transmission.

5.1.3.3.6 Rapidité des processus et besoins en mémoire

Il faut s'interroger sur la rapidité d'exécution de l'ensemble du processus. Les opérations décrites nécessitent une complexité légèrement accrue par rapport aux systèmes existants (opérations décrites en Figure 5-3). Les principaux traitements ajoutés sont le filtrage des paquets correspondant aux conditions d'activation de MPHP, et, pour ces paquets, deux décodages RS supplémentaires. Compte tenu de la (relative) simplicité des opérations proposées, l'opération la plus coûteuse en termes de vitesse de traitement est le décodage RS de MPHP. Les technologies actuelles des circuits intégrés pour application spécifique (*Application Specific Integrated Circuit*, ASIC) supportant des vitesses extrêmement élevés (plus d'un gigabit par seconde pour certaines implémentations sur $GF(2^8)$), la mise en œuvre de MPHP n'ajouterait ni délai pénalisant ni diminution du débit sur le service de transport IP. Enfin, une mémoire tampon additionnelle correspondant à la taille de deux paquets MPHP (au maximum 3 kilo-octets) serait nécessaire.

5.1.3.3.7 Conclusion sur les limites de l'architecture AM

Si l'on juge le dernier point largement acceptable pour une implémentation, les limites de la méthode concernent réellement les aspects suivants :

- la diminution de l'efficacité de transmission, contrainte par la norme MPEG-2 dans laquelle cette proposition se place ;
- l'absence de support de services de QdS (utilisation du champ *Priorité*) et de services de sécurité (utilisation du champ *Scrambling Control)*.

Nous discuterons de la manière de lever ces limites en proposant une architecture idéalisée ; elle fera l'objet du paragraphe 5.2.

5.1.4 ***Compression d'en-têtes***

La compression d'en-têtes est une technique connue depuis plusieurs années, initiée par un procédé proposé par Van Jacobson [Jac90] ; elle constitue la dernière action, facultative, à mettre en œuvre dans l'architecture modifiée. Aujourd'hui, les principaux travaux traitant de la compression d'en-têtes [Deg99], [Cas99], [Khi00], [Lia01], [Miy00], ont été normalisés par un groupe de travail à l'IETF nommé *RObust Header Compression* [ROHC], notamment grâce à la RFC 3095 [Bor01]. L'enjeu de la compression d'en-têtes est la réduction des temps de transmission en améliorant l'utilisation des ressources en bande passante.

5.1.4.1 Présentation générale de ROHC

On trouve deux principaux domaines d'utilisation du schéma de compression ROHC. Les applications multimédia dans lesquelles le message applicatif est fractionné en petites unités de données constituent le premier domaine. Par exemple, pour un service de voix sur IP avec le codec *Adaptive Multi Rate* (AMR), la taille de ces unités varie entre 5 et 31 octets. Dès lors, pour leur transmission avec le protocole de transport temps réel comme *Real-time Transport Protocol* (RTP) utilisé avec la pile protocolaire RTP/UDP/IP, le poids cumulé des en-têtes représente au moins 56% de la taille totale des datagrammes IP (40 octets d'en-tête). La transmission dans les

réseaux bas-débit (p.ex. avec le protocole *Point-to-Point Protocol* (PPP)) ou les réseaux sans-fil est le second champ d'application de ROHC. Ainsi, dans les réseaux mobiles de troisième génération UMTS, la compression d'en-tête est supportée pour le service de transport de datagrammes IP au travers de la couche de convergence *Packet Data Convergence Protocol* (PDCP). Deux options y sont en fait possibles : ROHC ou un schéma alternatif de compression d'en-têtes, *IP Header Compression* [Deg99].

5.1.4.2 Principe de fonctionnement

Les en-têtes traités par ROHC sont compressés en exploitant la redondance qui existe entre les en-têtes successifs des unités de données issues d'une même couche protocolaire. Il a ainsi été remarqué qu'au cours d'une transmission de datagrammes IP d'un même flux, plus de la moitié des données d'en-tête reste invariable. Chaque équipement ROHC, émetteur ou récepteur, dispose d'un contexte où sont stockées les informations non compressées des en-têtes précédemment traités. Le contexte est utilisé au moment de la compression, et surtout de la décompression, afin de restituer à la couche supérieure les unités de données avec leurs en-têtes décompressés.

Pour réaliser la compression, ROHC classe les données traitées en plusieurs catégories : INFERRED, STATIC, STATIC-DEF, STATIC-KNOWN, et CHANGING. Les données INFERRED sont des données qui peuvent être connues par simple examen du paquet, par exemple la longueur des datagrammes IP ; ces données ne sont donc jamais transmises. Les données STATIC et STATIC-DEF sont des données invariables pour un flux et ne sont transmises qu'une seule fois au début de la communication (p.ex. la version du protocole utilisée). Les données STATIC-KNOWN sont des données dont la valeur est connue et qui ne sont jamais transmises (p.ex. l'*offset* du Fragment IP quand il n'y a pas de fragmentation). Les données CHANGING sont à leur tour sous-divisées en 5 types : STATIC, SEMI-STATIC, RARELY-CHANGING, ALTERNATING et IRREGULAR. Suivant la sous-division considérée, les données sont transmises codées par différents algorithmes, adaptés aux types de changements attendus. Le Tableau 5-3 suivant, tiré de [Bor01], rend compte de la proportion importante de données de type STATIC dans l'en-tête IPv4 qui peuvent contribuer à l'efficacité de la compression.

Champ de l'en-tête IP	Taille (bits)	Classe
Version	4	STATIC
Longueur d'en-tête	4	STATIC-KNOWN
Type de Service (ToS)	8	CHANGING
Longueur du paquet	16	INFERRED
Identification	16	CHANGING
Drapeau réservé	1	STATIC-KNOWN
Drapeau Ne pas fragmenter	1	STATIC
Drapeau Plus de fragments	1	STATIC-KNOWN
Offset du Fragment	13	STATIC-KNOWN
TTL	8	CHANGING
Protocol	8	STATIC
Header Checksum	16	INFERRED
Adresse source	32	STATIC-DEF
Adresse destination	32	STATIC-DEF

Tableau 5-3 : Classification des en-têtes du protocole IP par ROHC

5.1.4.3 Utilisation des profils

Dans la RFC originelle de ROHC [Bor01], 4 profils définissent la manière de compresser les en-têtes : le profil RTP (pile de protocoles RTP/UDP/IP), le profil UDP (UDP/IP), le profil ESP (*Encapsulating Security Payload*, ESP/IP) offrant des services de sécurité pour IP, et le profil sans compression. Depuis juin 2004, un nouveau RFC enrichit la méthode ROHC d'un profil IP pur [Jon04], alors qu'un profil TCP (pile TCP/IP) est en cours de normalisation [Pel04]. ROHC est également prévu pour être implémenté au-dessus du protocole PPP [Bor02].

5.1.4.4 Gestion de la robustesse dans ROHC

Comme tout procédé de compression dans les réseaux, la principale difficulté à laquelle ROHC est confronté est la gestion des pertes ou corruptions de paquets, entraînant la désynchronisation des contextes entre l'émetteur et du récepteur. La robustesse est obtenue, mais au prix d'une certaine complexité.

ROHC agit par étapes en utilisant plusieurs états, similaires à des niveaux de compression : *Initialization Refresh* (IR), *First Order* (FO) et *Second Order* (SO). Chacun des états définit la manière d'effectuer la compression. Comparativement à ces états, le décompresseur évolue lui entre trois autres états : *No Context* (NC), *Static Context* (SC) et *Full Context* (FC).

Au début de la communication, les contextes doivent être initialisés avec les premiers en-têtes et en-queue du flux ; le compresseur et le décompresseur se trouvent respectivement dans l'état IR et NC. Puis, au fur et à mesure de l'apprentissage, les contextes s'enrichissent des données statiques, puis dynamiques, permettant d'améliorer l'efficacité de la compression en envoyant moins de données.

Les pertes d'unités de données sont détectées à partir d'expiration de *timers* et des corruptions des données, lesquelles sont détectées à partir de CRC (codés sur 8 bits pour les données statiques, sur 3 bits sur les données dynamiques). Dans ce dernier cas, ROHC prévoit la gestion d'erreurs lorsque la petite taille du CRC le rend insuffisant pour détecter les erreurs. La propagation d'erreur de contexte, synonyme de pertes des unités de données successives, ne peut toutefois pas être exclue.

Parallèlement aux états de compression, 3 modes opératoires de fonctionnement de ROHC sont définis : le mode unidirectionnel (mode U), le mode bidirectionnel optimiste (mode O) et le mode bidirectionnel fiable (mode R). Le détail des opérations de compression/décompression, des transitions entre les différents états, de détections d'erreurs ou pertes de paquets dépendent du mode. La communication débute toujours dans le mode U. Elle y reste tant que le compresseur ne reçoit pas de message de la part du décompresseur lui demandant une transition vers un autre mode. Dans ce mode, les taux de compression sont les plus faibles. Afin d'éviter que des pertes de paquets entraînent la corruption du contexte du décompresseur, le contexte est rafraîchi périodiquement par des paquets où les données dynamiques ne sont pas compressées. Ceci garantit que le mécanisme n'introduit pas de rejets de paquets additionnels.

La robustesse de ROHC est influencée par beaucoup de facteurs : modes opératoires, état de compression/décompression, mais aussi un ensemble de beaucoup de paramètres dont les valeurs ne sont pas spécifiées par le standard. Selon les caractéristiques du réseau, le compromis entre robustesse et performance de compression doit alors être soigneusement étudié. Une étude comme [Min03] propose, entre autres, l'optimisation de ces paramètres pour les réseaux sans-fil dans lesquels les taux d'erreurs binaires peuvent valoir jusqu'à 10^{-2}. Si le paramétrage est correct, un tel taux d'erreur résiduel entraîne une perte de paquets moyenne de seulement 30 % au niveau du décompresseur. Cet exemple prouve clairement le potentiel de la robustesse de ROHC.

5.1.4.5 Support de ROHC dans l'encapsulation ULE de flux IP Multicast

En l'état actuel des choses, ROHC n'est pas prévu pour fonctionner avec la méthode d'encapsulation ULE. Toutefois, le groupe de travail IP/DVB offre la perspective de le supporter [Mon04] grâce à une indication donnée dans les SNDU sur la nature de la charge utile transportée. Dans le format des SNDU définies par ULE, le champ *Type*, prévu à cet effet, pourrait donc convenir. ROHC pourrait être appliqué sur les flux IP, mais idéalement il devrait l'être sur les SNDUs de ULE (mises à part celles de bourrage), surtout dans la mesure où l'en-tête de ULE est normalement invariable.

Dans le domaine plus spécifique des communications IP Multicast par satellite, il est préférable de restreindre les décompresseurs ROHC au mode unidirectionnel malgré la possibilité que certains récepteurs pourraient avoir d'utiliser leur voie de retour.

5.1.4.6 Mise en œuvre de ROHC dans l'architecture modifiée avec MPHP

L'architecture proposée avec MPHP pourrait tirer parti de la compression d'en-têtes pour sa meilleure utilisation du lien, profitant en plus de la diminution du taux de perte de paquet due à la réduction de la taille des en-têtes. ROHC s'insèrerait alors naturellement dans l'architecture modifiée avec MPHP en tant que couche utilisatrice des flux protégés par MPHP. Grâce à la robustesse du codage MPHP, la probabilité de défaillance du CRC de 3 bits de ROHC est considérablement réduite puisque le faible taux de codage (1/3) de MPHP détecte les mots-codes RS erronés (et les rejette) avec une forte probabilité.

Déployer ROHC conjointement à MPHP peut paraître assez paradoxal dans la mesure où les deux mécanismes procèdent de manière antagoniste : le premier retire des données avant la transmission sur le lien satellite, alors que le second en ajoute. Cette démarche est en réalité identique à ce qui est fait couramment dans n'importe quelle autre transmission : la redondance de la source est retirée (la plupart du temps par l'application), puis un certain niveau de redondance est rajoutée lors de la transmission. Ici, l'activation simultanée de ROHC et de MPHP, appelé dans la suite *architecture modifiée avec compression* (AMC) revient à rapprocher codage source et codage canal, évitant ainsi les traitements intermédiaires (*i.e.* les rejets de paquets) qui diminuent l'efficacité globale du système. La pile protocolaire de l'architecture AMC est représentée dans la Figure 5-4, par rapport aux piles utilisées dans les autres architectures.

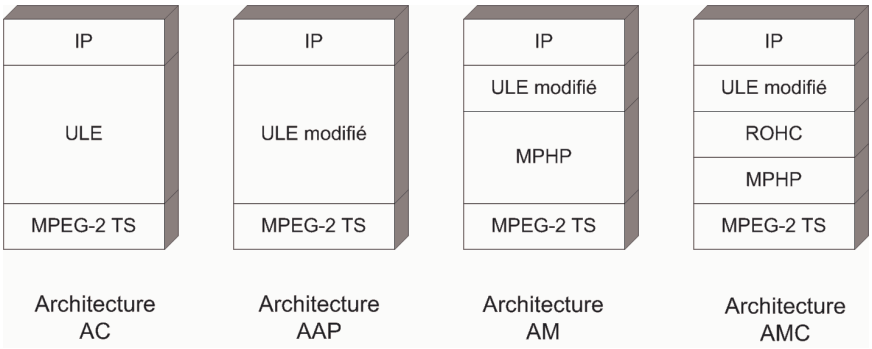

Figure 5-4 : Pile protocolaire des différentes architectures selon l'activation de MPHP et de ROHC

Le fonctionnement de MPHP et de ROHC par profils s'avère être *a priori* assez intéressant pour configurer les deux mécanismes quasi-simultanément. Malheureusement à ce niveau, un nouveau verrou à lever apparaît pour la mise en œuvre. En effet, après la compression, les tailles des en-têtes deviennent variables (mêmes pour des paquets initialement à en-têtes fixes) en fonction des états du compresseur. En imaginant qu'un nouveau profil de type UDP/IP/ULE soit défini pour ROHC, le codage MPHP pourrait déterminer, en communicant avec le

compresseur, la valeur de N_H pour encoder les données (en supposant N_T invariable et égal à 4). Le problème se situerait alors au niveau du récepteur MPHP : une fois le paquet MPHP extrait, comment connaître la valeur de N_H à utiliser pour le décodage ?

La solution à ce nouveau problème passe nécessairement par une indication de la longueur de l'en-tête après compression dans le flux MPEG-2 TS. Cette indication pourrait être donnée, par un octet, inséré par exemple juste après l'octet *Payload Pointer*. Le codage PUSI (n=6, k=2, d=5), pourrait alors être remplacé par le codage (n=12, k=4, d=5). La performance de ce code, pour un BER résiduel de 10^{-2}, assure que la probabilité de corruption des 2 octets après décodage PUSI est inférieure à 0.07 %. Au vu de la longueur moyenne des en-têtes compressés devant les en-têtes non compressés (environ 8 octets par en-tête compressé pour le profil RTP/UDP/IP dans le mode U, contre 40 pour les en-têtes normales) [Min03], l'ajout d'un octet supplémentaire reste acceptable en termes de gain de compression.

Le dernier point à étudier pour cette solution concerne la faisabilité d'un codage/décodage RS avec paramètres (n, k) dynamiques, sur circuit intégré. Dans la plupart des systèmes de codage existants, les circuits sont pré-câblés pour fonctionner avec un seul jeu de paramètres (p.ex. le code RS de DVB (n=204, k=188)). De tels circuits, capables d'être rapidement reconfiguré, semblent déjà disponibles ; le produit [AHA] serait un exemple de solution pratiquement adaptée. Ses délais de reconfiguration sont compatibles avec les débits usuels d'applications multipoints par satellite (quelques mégaoctets par seconde au maximum). Seule la variation du nombre de symboles de parité (entre 2 et 20) est trop faible pour être utilisable dans l'architecture AMC (idéalement, jusqu'à 160, pour protéger 80 octets d'en-tête). Cela suggère néanmoins la faisabilité de l'implémentation du codage par MPHP avec les circuits et technologies actuels.

5.2 GÉNÉRALISATION DE LA PROPOSITION À UNE ARCHITECTURE DE TRANSMISSION IDÉALE

Nous proposons de généraliser l'architecture AM au-delà du contexte de MPEG-2 TS, permettant ainsi de s'affranchir des limites relevées en 5.1.3.3.7. Cette architecture sera donc baptisée *Architecture Généralisée* (AG). Son but est de s'affranchir des problèmes de mises en œuvre soulevés par l'architectures AM afin de fournir à la couche IP la meilleure fiabilité possible. La contrainte que nous fixons est (comme MPEG-2 TS) d'utiliser, à bas niveau, un type d'unités de données de longueur fixe, afin d'en faciliter sa détection par les mécanismes de synchronisation.

Rappelons que les deux points principaux ayant motivé la conception des architectures AM et AMC sont :

- l'importance des données de signalisation par rapport aux autres données ;
- le besoin d'optimisation de l'utilisation des ressources.

AG définit deux niveaux d'unités de données de protocoles : les 1-PDU et les 2-PDU. Les 1-PDU sont les équivalents des paquets MPEG-2 TS (de taille fixe) et les 2-PDU ceux des SNDU de MPE ou ULE, de taille variable.

5.2.1 *Description du niveau 1-PDU*

Le format de la 1-PDU doit être structurée de la façon la plus simple et la plus efficace pour gérer la fiabilité. Aussi, ce niveau doit inclure le strict minimum de fonctionnalité pour la transmission numérique, ce qui reporte les autres fonctions à réaliser sur le(s) protocole(s) de niveau supérieur. L'en-tête proposé est composé d'un identificateur de flux (p.ex. 2 octets), d'un numéro de séquence relatif au flux de 1-PDU (p.ex. 4 bits), et d'un *second* numéro de séquence relatif à la 2-PDU transportée par chaque 1-PDU (p.ex. 4 bits). La combinaison de ces deux numéros de séquence permet d'une part d'empêcher le traitement de 1-PDU avec en-tête erroné, et d'autre part de réaliser facilement le réassemblage en conservant l'alignement correct des 2-

PDU (même si la 1-PDU contenant le début d'une nouvelle 2-PDU était perdu). Un octet supplémentaire pourrait être inclus dans les 1-PDU pour faciliter l'acquisition de la synchronisation par le récepteur. Enfin, l'en-tête des 1-PDU doit être protégé par codage UEP (p.ex. un code RS), qui est l'équivalent du codage PUSI sur GF(2^4) mentionné précédemment. Avec un taux de codage 1/3, les en-têtes des 1-PDUs pourraient alors être codés sur 12 octets.

5.2.2 *Description du niveau 2-PDU*

La 2-PDU doit permettre la signalisation de tous les autres services complémentaires qu'on peut attendre (gestion de l'adressage, sécurité, qualité de service) au niveau liaison. Le reste des traitements effectués à ce niveau intègre alors MPHP tel qu'il a été décris en 5.1.2, et, de manière optionnelle, ROHC. Dans ce dernier cas, le paramètre N_H devrait en plus être indiqué pour la décompression, par exemple au tout début de la SNDU. Il pourrait éventuellement être protégé.

5.3 IMPLÉMENTATION DES MÉCANISMES DANS LE SIMULATEUR IP/DVB

Afin de mesurer l'intérêt de l'architecture modifiée, le simulateur IP/DVB décrit dans le chapitre précédant a été adapté pour modéliser les différentes architectures proposées. Pour certaines des actions à prendre, les mécanismes ont été fidèlement reproduits, mais pour d'autres, la complexité des opérations a due être simplifiée. Nous expliquerons en quoi les simplifications présentées pourront être considérées comme suffisamment représentatives des spécifications précises.

5.3.1 *Fonctionnalités implémentées*

5.3.1.1 Couche ULE modifiée

La simulation de ce premier point n'a pas engendré de simplification particulière. Le choix de limiter (ou non) le calcul du CRC sur l'unité de données entière (architecture AAP) ou sur son en-tête (architecture AC) est donné par un booléen, du coté de l'émetteur.

5.3.1.2 Codage et décodage MPHP

Dans la simulation du codage MPHP, les données de parité sont positionnées à la fin de chaque SNDU. La mise en œuvre du mécanisme est complète.

5.3.1.3 Réassemblage

Le réassemblage tel qu'il a été décrit en 5.1.3 est modélisé dans le simulateur en excluant les cas où les paquets MPEG-2 sont traités alors que l'un des 6 bits sensibles (drapeaux PUSI, Priorité, et champ compteur de continuité) est erroné. En revanche, pour prendre en compte ces erreurs, on a supposé qu'une corruption binaire (ou plus), quelle que soit sa position dans l'en-tête de 4 octets (PID, synchronisation, etc…), était suffisante pour perdre le paquet MPEG-2 TS. Comme pour l'architecture AC, l'octet *Payload Pointer* est supposé faire partie de l'en-tête de la couche ULE modifiée, et pour cette raison il est protégé par le codage RS de MPHP.

5.3.1.4 ROHC

La complexité du mécanisme ROHC n'a pas été intégrée dans le simulateur. À la place, une couche équivalente compresse les en-têtes dans un rapport constant (3 octets d'en-tête pour 1 octet transmis, sur la base de mesures réalisées dans [Min03]). ROHC doit alors informer MPHP sur la taille d'en-tête, après compression. À la décompression, toute corruption de données compressées par ROHC entraîne le rejet de la SNDU (mais pas des suivantes).

5.3.2 *Limite du modèle de simulation*

Il faut être conscient que la simplification des opérations complexes de mise en œuvre n'est pas suffisante pour valider formellement tous les résultats obtenus pour l'architecture AM ou AMC. Dans la suite, toutes les évaluations faites dans le cadre de ces dernières seront en fait celle d'une *architecture modifiée simplifiée* (AMS) ou *architecture modifiée avec compression simplifiée* (AMCS). À titre de comparaison des deux approches, nous donnons le tableau suivant :

Évènement pour un taux d'erreur résiduel inférieur ou égal à 10^{-2}	Probabilité maximale dans l'architecture AM	Probabilité maximale dans l'architecture AMS
Perte d'un paquet MPEG-2 TS	0.21 %	0.32 %
Corruption du *Payload Pointer*	0.09%	Moins de 10^{-6} avec le code (n=87, k=29)
Traitement de paquet MPEG-2 TS avec drapeaux corrompus	6 %	0 %

Tableau 5-4 : Limite du modèle de l'architecture AMS implémentée par simulation

D'après ce tableau, il y a 6% de chances qu'un paquet MPEG-2 TS soit traité alors que l'un des drapeaux *PUSI*, *Compteur de continuité*, ou *Priorité* est faux. Ce pourcentage est assez faible pour avoir un impact limité lorsque les paquets MPHP sont courts (de 200 à 500 octets) mais pour des paquets plus longs (1500 octets) cela n'est plus vrai. Il apparaît donc que ces 6 bits du paquet MPEG-2 TS sont très sensibles à la corruption. Une fiabilisation supplémentaire sur ces données paraît difficile à réaliser car elle nécessiterait la modification d'une partie de la norme MPEG-2 TS. Une alternative préférable à cette solution serait à nouveau de recourir à l'information de confiance que produit l'algorithme de décodage à décision souple de Viterbi, et d'éliminer les paquets MPEG-2 pour lesquels le BER résiduel serait trop élevé.

5.4 PREMIÈRE ÉVALUATION DE LA PROPOSITION

Le prochain chapitre présentera une évaluation plus complète du bénéfice des architectures déployant MPHP. Nous proposons ici de mesurer un premier type de performance de l'architecture AM concernant le taux de rejets de paquets au niveau IP. Les résultats sont indiqués dans la Figure 5-5, pour laquelle les datagrammes IP sont de 1500 octets. On y constate notamment que pour E_c/N_0=1 dB, le taux de rejet des paquets est important (55%), mais moins que dans l'architecture AC, pour lequel il vaut 100% dès que le rapport E_c/N_0 est inférieur ou égal à 1.4 dB. Avec cette longueur de datagrammes, les résultats obtenus pour l'architecture AMC sont quasiment identiques à ceux de l'architecture AM, et pour cette raison, ils n'ont pas été inclus sur la figure.

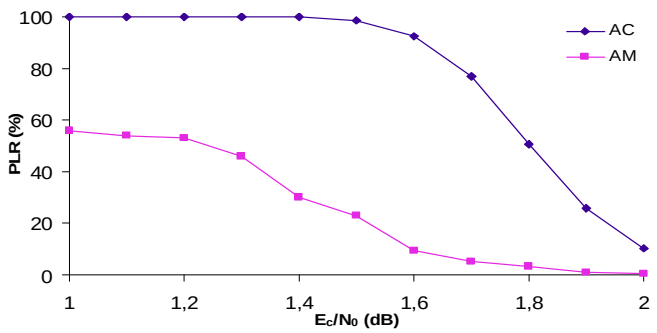

Figure 5-5 : PLR au niveau du service IP (pour des datagrammes IP de 1500 octets) pour les architectures AC et AM/AMS

CONCLUSION

Avec les techniques de transmission provenant du domaine des réseaux terrestres, les liaisons par réseau satellite intègrent classiquement des mécanismes de vérification d'intégrité sur les unités de données. Lorsque les vérifications détectent des corruptions, les unités de données sont entièrement rejetées, même si la majeure partie des données arrive sans erreur au niveau du récepteur et pourrait être exploitée si les mécanismes des couches supérieures en étaient conscients. Ces rejets introduisent des erreurs additionnelles par rapport au canal de transmission, alors que dans le domaine des réseaux par satellite, le prix d'utilisation de la bande passante est important.

Cette constatation nous a amené à considérer une modification de l'architecture conventionnelle qu'utilise actuellement le standard d'encapsulation ULE, encore en cours d'élaboration. Dans cette architecture, les vérifications des unités de données doivent être réduites à leur minimum pour ne pas rejeter celles qui pourraient être exploitées par les couches supérieures. Toutefois, pour éviter les conséquences de corruptions de données trop fâcheuses comme le traitement de signalisation erronée, les vérifications de CRC (ou sommes de contrôles) par les couches de niveau liaison doivent inclure les en-têtes et en-queues des unités de données.

L'idée, séduisante au départ, apporte pourtant de nombreuses limitations relatives (1) aux résultats obtenus et (2) à la faisabilité d'une méthode d'encapsulation pratique tolérant les corruptions. Le premier point a conduit à concevoir MPHP, une méthode de protection des données de signalisation contenues dans chaque unité de données. La méthode protège simultanément les en-têtes de plusieurs couches protocolaires à partir de la couche de niveau liaison (*i.e.* ULE). Comme il est difficile, à ce niveau, d'extraire des unités de données la taille exacte des information de signalisation, le procédé suppose que les unités de données sont identifiées par flux (grâce à des numéros d'adresse IP réservés). Chacun des flux pour lesquels MPHP peut être activé est ensuite traité selon son profil, qui définit la pile de protocole qui a servi à le concevoir. Plus le profil utilisé pour traiter une SNDU est précis, plus l'approche est intéressante. Si le nombre d'octets d'en-tête ou d'en-queue du profil est supérieur à celui de la SNDU, des octets supplémentaires sont protégés, ce qui consomme inutilement des ressources pour le service. Dans l'autre alternative, les derniers octets de signalisation de la SNDU ne sont pas protégés, au risque qu'une corruption de ces octets suffise à faire perdre la SNDU.

Dans le cadre de cette proposition, l'étude de la mise en œuvre de MPHP dans les réseaux IP/DVB-S a été réalisée. Il a été montré qu'il était possible d'adapter l'encapsulation classique réalisée par ULE sans modification du format des paquets MPEG-2 TS, au détriment d'une complexité accrue de la couche de réassemblage. Enfin, le fonctionnement de MPHP pourrait être déployé conjointement avec le mécanisme de compression d'en-têtes ROHC. La configuration par profils de ROHC pourrait alors permettre de faciliter son intégration avec MPHP. On peut remarquer que l'approche, similaire dans MPHP et ROHC, remet profondément en cause le principe fondamental de l'architecture en couches indépendantes. Partant de cette étude, une architecture de transmission généralisée au-delà du contexte de MPEG-2 a ensuite été proposée pour lever les problèmes de mises en œuvre.

Une première évaluation de l'architecture modifiée avec MPEG-2 a montré le potentiel de la méthode en comparant les taux de rejet de paquets entre l'architecture conventionnelle et l'architecture qui est proposée. Certains mécanismes, en raison de leur complexité, n'ont toutefois pas pu être intégrés complètement dans le simulateur, mais leur impact est supposé mineur sur les performances obtenues.

Le dernier chapitre de la thèse sera l'occasion de répondre à des questions essentielles. Jusqu'à présent, aucun des services offerts par les couches de transport traditionnelles comme UDP n'est supposé être compatible avec le mode de livraison binaire, car les unités de données erronées sont classiquement rejetées par ce niveau. Pourtant, si la plupart des applications existantes sont conçues selon cette hypothèse, rien n'empêche d'envisager l'existence de nouveaux protocoles de transport, compatibles avec ce mode de livraison. L'évaluation de l'architecture, proposée avec de tels services, pourra alors être donnée.

BIBLIOGRAPHIE

[AHA] http://www.aha.com/show_pub.php?id=25

[Bor01] IETF RFC, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", C. Bormann *et al.*, RFC 3095, juillet 2001.

[Bor02] IETF RFC, "ROHC over PPP", C. Bormann, RFC 3241, avril 2002.

[Cas99] IETF RFC, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", S. Casner, RFC 2508, février 1999.

[Deg99] IETF RFC, "IP Header Compression", M. Degermark *et al.*, RFC 2507, février 1999.

[DVB04] ETSI EN 301 192, "Digital Video Broadcasting (DVB), DVB Specification for Data Broadcasting", v1.4.1, juin 2004.

[Jac90] IETF RFC, "Compressing TCP/IP Headers", V. Jacobson, RFC 1144, février 1990.

[Jon04] IETF RFC, "RObust Header Compression (ROHC): A Compression Profile for IP", L-E. Jonsson, G. Pelletier, RFC 3843, juin 2004.

[Khi00] IETF Internet Draft, "Adaptive Header Compression (ACE) for Real-Time Multimedia", L. Khiem *et al.,* 2000.

[Lia01] IETF Internet Draft, "TCP-Aware RObust header Compression (TAROC version 04)", H. Liao *et al.*, draft-ietf-rohc-taroc-04, work in progress, Proceedings of the 52nd IETF, SLC, novembre 2001.

[Min03] "Compression des en-têtes sur les réseaux bas-débit", A. Minaburo Villar, thèse de doctorat de l'Université de Rennes I, décembre 2003.

[Miy00] IETF Internet Draft, "Robust Header Compression using Keywords Packets", A. Miyazaki *et al.,* draft-miyazaki-rohc-kwhc-00.txt, 2000.

[Mon04] IETF Internet Draft "A Framework for transmission of IP datagrams over MPEG-2 networks", M.J. Montpetit *et al.*, draft-ietf-ipdvb-arch-00.txt, juillet 2004.

[Pel04] IETF Internet Draft, "RObust Header Compression (ROHC): A Profile for TCP/IP (ROHC-TCP)", G. Pelletier *et al.*, draft-ietf-rohc-tcp-07.txt, juillet 2004.

[ROHC] http://www.ietf.org/html.charters/rohc-charter.html

[Rei03] "Voice Quality Evaluation for Wireless Transmission with ROHC", S. Rein, F. Fitzek, M. Reinslein, in International Conference on Internet and Multimedia Systems and Applications, août 2003.

Chapitre 6

Mécanismes de fiabilité de niveau transport exploitant le service de livraison en mode binaire

INTRODUCTION

L'architecture présentée précédemment offre au niveau du protocole de transport un nouveau type de service (*i.e.* livraison en mode binaire) que les protocoles courants ne peuvent pas totalement exploiter, quel que soit le service de fiabilité qu'ils offrent. Il est donc nécessaire de prévoir la gestion d'un telle livraison afin d'en faire bénéficier l'application. Au cours de ce chapitre, deux services seront étudiés. Le premier service sera appelé fiabilité binaire non contrôlée. Il s'agit d'un service n'offrant pas de garantie de fiabilité, comparable à UDP, pouvant délivrer à l'application des unités de données corrompues. Le second service proposé offre une garantie de fiabilité totale, plus traditionnelle. Les objectifs de ce chapitre sont multiples. Il permettra tout d'abord de cerner les applications compatibles avec le service à fiabilité binaire non contrôlée. Ensuite, l'évaluation des performances sera donnée dans des conditions de bruit élevé. Elle montrera que pour les applications tolérantes à ce mode de livraison, le débit utile peut être augmenté jusqu'à un facteur 5. Ensuite, les mécanismes nécessaires à la réalisation d'un service à fiabilité totale seront donnés. Celle-ci permettra de réduire jusqu'à 30 fois la quantité de données transmise, selon la qualité de la liaison satellite. Le chapitre se terminera en instanciant la proposition par une mise en œuvre détaillée.

6.1 SERVICE À FIABILITÉ BINAIRE NON CONTRÔLÉE

Les propositions et résultats mentionnés dans cette section ont été présentés dans [Arn04a].

6.1.1 *Applications tolérantes aux erreurs binaires*

Dans les réseaux IP traditionnels, les erreurs de transmission (et surtout les congestions) se répercutent par des pertes de paquets. Certaines applications ayant des contraintes temporelles importantes ne peuvent pas accepter les délais inhérents à des retransmissions ; elles doivent s'accommoder à des pertes de paquets. Les communications multimédia (audio ou vidéo) diffusées en *streaming* sont l'exemple type de cette catégorie d'applications. Au cours de la communication, les pertes de paquets peuvent se traduire, selon les cas, par une diminution du débit utile moyen ou par un gel de la présentation du média à l'utilisateur. Grâce à des protocoles

de contrôle (par exemple RTP *Control Protocol*, RTCP), la mesure et le contrôle des performances de la communication de bout en bout est possible. Le cas échéant, le protocole peut agir sur des paramètres impliqués dans la fiabilisation. Naturellement, dans tous les cas, la perte de paquets se traduit d'une manière ou d'une autre par une réduction de la qualité du média restitué.

La plupart de ces applications ont été développées sous l'hypothèse que le protocole de transport leur délivre des unités de données exempte d'erreurs. Pourtant, depuis peu, un certain nombre d'applications tolérantes aux erreurs binaires a vu le jour, en particulier dans le domaine des télécommunications mobiles à taux d'erreurs bits variables. Pour cela, de nouvelles formes de codeur/décodeurs source (*codecs*) ont été élaborées. Les principaux codecs tolérant les corruptions de données sont :

- *Adaptive Multi Rate* (AMR) [3gp02], choisi pour l'encodage de la voix dans les réseaux UMTS ;
- H.263 [Itu98] de l'ITU-T (codage vidéo) ;
- MPEG-4 [Iso01], intégrant des outils d'atténuation des erreurs (*Error Resilience Tools)* (techniques de codage source) en plus d'outils plus conventionnels de protections aux erreurs *(Error Protection Tools)* (techniques de codage canal impliquant notamment la protection inégale aux erreurs, UEP).

Les outils d'atténuation des erreurs de MPEG-4 pour la vidéo sont assez illustratifs de la recherche dans ce domaine. On peut les répertorier en trois modes : la resynchronisation, la récupération de données et la dissimulation des erreurs. Ces modes s'utilisent de manière hiérarchique : la dissimulation des erreurs fonctionne si la récupération des données est utilisée, alors que cette dernière nécessite la resynchronisation. D'autre part, leur activation est source d'ajout de données dans le flux, entre 4 et 11 % selon la longueur des paquets vidéo, le débit initial de la séquence, le paramètre de quantification, etc…

6.1.1.1 La resynchronisation ou approche paquets (*Video Packet Approach*)

Dans cette approche, les données sont réparties en paquets représentant une ou plusieurs rangées de *Macro-Blocs* (blocs d'échantillons de luminance et de chrominance). Dans chaque paquet vidéo, un marqueur est inséré pour resynchroniser périodiquement le décodeur. Le décodage de chaque paquet peut ainsi être réalisé indépendamment des autres. Une technique similaire est employée par H.263 où les paquets sont appelés *Groupes de Blocs* (GOB). Pour éviter des pertes de données pendant les séquences très actives, le nombre de Macro-Blocs entre deux resynchronisations est variable. Une autre technique, appelée répétition de l'en-tête (*Header Extension Code*), s'apparente à cette approche.

6.1.1.2 La récupération des données (*Data Recovery*)

Le principe de cette seconde technique est de limiter l'effet de la corruption de données au sein d'un paquet. Pour cela, la technique repose sur l'usage d'un dictionnaire dans lequel les mots-codes sont réversibles (*i.e* pouvant être lus dans les deux sens) et de longueur variable ; ils sont ainsi appelés *Reversible Variable Length Code (*RVLC). Grâce à cette propriété, le décodeur détecte plus précisément les corruptions et peut maximiser l'exploitation des données.

6.1.1.3 La dissimulation des erreurs (*Error Concealment*)

La troisième technique appelée *dissimulation des erreurs* consiste à séparer les données (*Data Partitioning*) d'un paquet en deux parties : les informations concernant les textures, et les informations de mouvements. Le but de cette séparation sémantique est d'exploiter l'information

de mouvements (plus importante) afin de réaliser le décodage même lorsque les informations décrivant les textures sont erronées.

6.1.2 ***UDP-lite***

6.1.2.1 Présentation

Comme il a été mentionné en introduction, le service de livraison en mode binaire ne serait intéressant qu'à la condition que le protocole de transport puisse exploiter sa spécificité. Malheureusement, UDP n'est pas compatible avec ce mode de livraison. Pourtant, il existe aujourd'hui une alternative à UDP qui paraît exactement adaptée à notre attente : *Lightweight User Datagram Protocol* (UDP-lite) [Lar04]. UDP-lite a été proposé par l'IETF ; il est normalisé sous forme d'une RFC depuis juillet 2004.

UDP-lite est une extension d'UDP avec une sémantique pratiquement identique. Leur distinction réside dans la politique de fiabilisation partielle au niveau du récepteur UDP-lite. Les unités de données d'UDP-lite sont séparées en deux zones : une zone protégée de la corruption d'erreurs (incluant obligatoirement l'en-tête) et une zone non protégée. La distinction entre les deux zones est indiquée dans chaque unité de données en remplacement du champ *Longueur* d'UDP, en donnant la taille de la zone protégée. Le calcul de la somme de contrôle s'effectue sur la seule partie protégée. À la réception, la corruption des données de la zone protégée est détectée par la non-conformité de la somme de contrôle, ce qui entraîne le cas échéant le rejet du paquet. En cas de corruption de la zone non protégée, l'unité de données corrompue est délivrée en l'état à l'application réceptrice. Les concepts et la mise en œuvre de la somme de contrôle partielle d'UDP-lite ont également été proposés récemment dans le cadre du protocole *Datagram Control Congestion Protocol* (DCCP) [Koh04].

6.1.2.2 Apport d'UDP-lite comme protocole de transport servant les applications tolérantes aux erreurs binaires

Séparer les paquets en deux zones parait être judicieux lorsque les paquets UDP-lite transportent des données de sensibilité inégale aux erreurs. Cela convient également à l'utilisation de la protection inégale aux erreurs. Parmi les travaux de recherche suivants, une majorité confirme l'intérêt de cette approche.

Concernant la voix sur IP avec le codec AMR (AMR/RTP/UDP-lite/IP), les résultats d'une étude [Ham03] montrent un intérêt relatif de la méthode en évaluant un critère normalisé appelé PESQ-MOS [Itu01] qui prend en compte l'influence de la perception humaine. La compression d'en-tête permet néanmoins de doubler les gains sur la métrique PESQ-MOS (avec des gains allant jusqu'à 0.5 sans compression et jusqu'à 1 avec la compression ROHC, sur une échelle de valeurs possible allant de 0 à 5).

Dans une autre étude concernant les réseaux GPRS [Fab00], l'utilisation des outils d'atténuation des erreurs de MPEG-4 pour la transmission vidéo fait apparaître un gain considérable d'une dizaine de décibels sur le rapport signal à bruit pic (*Peak Signal to Noise Ratio*, PSNR).

Singh *et al.* [Sin01] étudient la possibilité qu'une couche « PPP-lite » modifiée de la même façon qu'UDP puisse fournir au réseau GSM des unités de données corrompues. Leur étude se base en partie sur une comparaison entre une transmission vidéo conventionnelle avec UDP, et une transmission avec l'architecture UDP-lite, au-dessus d'un réseau GSM, dans un mode transparent sans reprise d'erreur. Bien que les débits mesurés au niveau de l'application réceptrice soient comparables, la réduction du taux d'erreur paquet grâce à UDP-lite est assez significative pour améliorer la qualité finale de présentation du média.

Les conclusions des travaux de Khayam *et al.* [Kha03] sont parmi les seuls à relativiser l'intérêt d'UDP-lite, dans une évaluation réalisée dans le contexte des réseaux sans-fil 802.11 pour la vidéo. Malgré cela, les auteurs estiment qu'implémenter un codage FEC au-dessus d'UDP-lite serait plus efficace avec une architecture protocolaire favorable.

6.1.2.3 Problème de sécurité

Offrir un service de sécurité pour un flux UDP-lite paraît aujourd'hui compromis. En effet, un protocole de sécurité comme IPSec s'applique sur l'ensemble de la charge utile IP, sans distinction entre zones sensibles ou insensibles aux corruptions. Dans ces conditions, des erreurs résiduelles de transmission impliqueraient le rejet systématique des unités de données, réduisant à néant l'utilité du déploiement d'UDP-lite. Seul un mécanisme s'appliquant sur la zone de données protégé par la somme de contrôle serait une solution envisageable.

6.1.3 *Support du service de livraison en mode binaire au niveau de la couche de liaison*

Afin d'être exploité, le service de transport atypique d'UDP-lite repose sur la capacité que la couche de niveau liaison tolère, elle aussi, le service de livraison en mode binaire. Actuellement seuls peu de réseaux, recensés sur le site de A. Gurtov [GUR], peuvent procurer ce service. Ces réseaux appartiennent tous au domaine des télécommunications mobiles : *Global System for Mobile communications* (GSM), *General Packet Radio Service* (GPRS), *Enhanced Data rate for GSM Evolution* (EDGE), *Universal Mobile Telecommunication Standard* (UMTS) ou encore CDMA 2000.

L'architecture protocolaire AM (et AMC), proposée dans le chapitre précédent, permet de supporter efficacement ce service dans le contexte satellitaire. Afin d'en mesurer l'intérêt, nous proposons l'évaluation des différentes architectures, rendue possible grâce au simulateur décrit précédemment.

6.1.4 *Évaluations des performances des différentes architectures pour le service à fiabilité binaire non contrôlée*

6.1.4.1 Méthode de mesures

Bien que les mesures des pertes de paquets effectuées au cours du chapitre précédent indiquent partiellement les gains que l'on peut espérer, la quantité de données exploitable par l'application dans les paquets non rejetés reste néanmoins inconnue. Une étude faite au niveau applicatif semble donc nécessaire.

Suivant la nature du média transporté, plusieurs métriques existent pour estimer la qualité de la présentation à l'utilisateur final. Certaines métriques sont normalisées [Ans00] [Itu01a] [Itu01b] ou très couramment utilisées (PSNR pour les images et la vidéo). Plus ou moins simples à obtenir, leur pertinence vis-à-vis des caractéristiques de la perception humaine n'est pas toujours vérifiée. Conjugué à une mise en œuvre assez délicate, ce type de mesures n'a pas été retenu dans le cadre de nos travaux.

Plus simplement, nous proposons de mesurer la performance du service de communication (prise à l'interface transport/application), relative au débit d'émission de la source. L'évaluation est entreprise grâce à l'outil de simulation développé et porte sur (un modèle de) la pile protocolaire RTP/UDP-lite/IP/ULE. La performance étudiée, que nous appellerons *facteur d'utilisation du lien*, prend en compte l'efficacité d'encapsulation et le taux d'erreur perçus par l'application.

Nous définissons l'efficacité d'encapsulation comme le rapport entre la taille moyenne des unités de données au niveau du service de transport (4-SDU), et de celle des SNDUs après l'encapsulation ULE (2-PDUs). Cette dernière prend en compte les données supplémentaires (codage MPHP) ou économisées (par compression des en-têtes) selon le déploiement des techniques. La différence entre la taille des 4-SDU et des 2-PDU sera appelée *D(i)* pour l'architecture (*i*). L'en-tête MPEG-2 TS et le *Payload Pointer* ne sont pas comptabilisés dans la mesure. Afin de juger l'apport de la compression d'en-tête, une nouvelle architecture est introduite : l'*Architecture Conventionnelle avec Compression* (ACC). Le Tableau 6-1 indique les valeurs de *D(i)* applicables pour chacune des architectures, en supposant que le taux de compression d'en-tête moyen est de trois octets de données pour un octet transmis.

La mesure du taux d'erreur pose un problème particulier quand les paquets sont perdus. Dans les simulations, les paquets perdus ne le sont pas réellement mais remplacés par des unités de données de même longueur, avec des valeurs arbitraires (p.ex. une série de bits à 0). Le décompte du taux d'erreur bits consécutif à une perte de paquet serait donc égal à 50 % en moyenne (pour une source générant un trafic binaire équiprobable), ce qui pourrait faire croire que ces données sont exploitables par l'application. Ce type de mesure serait évidemment dangereux. Une manière de contourner le problème est d'effectuer le décompte en mesurant le taux d'erreur par octet du flux livré à la couche applicative. Dans ces conditions, la probabilité qu'un octet d'un paquet « perdu » soit correctement remplacé ne vaut plus que 1/256 ; le taux d'erreur octet correspondant est alors supérieur à 99.5 %.

Identifiant de l'architecture *(i)*	*D(i)*
AC	48
AM	144
ACC	16
AMC	48

Tableau 6-1 : Différences entre la taille des 2-PDU et 4-SDU dans la pile RTP/UDP-lite/IP/ULE dans les architectures AC, AM, ACC, AMC

En appelant *F(i)* le facteur d'utilisation du lien satellite, SER*(i)* le taux d'erreur symbole moyen pour l'architecture *(i)* considérée et L_p la longueur de la charge utile des datagrammes IP, on pose :

$$F(i) = \frac{(L_p - 20)}{(L_p - 20) + D(i)} . \left(1 - SER(i)\right) \qquad (6.1)$$

6.1.4.2 Résultats

Dans les figures suivantes (Figure 6-1, Figure 6-2, Figure 6-3, Figure 6-4), *F(i)* est tracé respectivement aux architectures AC, ACC, AM, AMC, en fonction de L_p et pour plusieurs valeurs du E_c/N_0 (1.5 dB, 1.6 dB, 1.9 dB). La Figure 6-5 compare les architectures précédentes avec l'architecture AAP (où les portées des CRC sont limitées aux en-têtes, sans autre mécanisme), lorsque L_p est fixé à 576 octets, en fonction du rapport E_c/N_0.

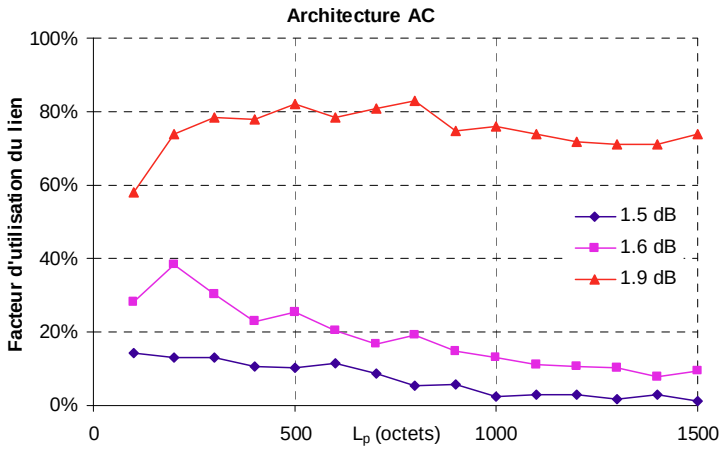

Figure 6-1 : Performance du service à fiabilité conventionnelle non contrôlée (architecture AC) en fonction de la taille de l'unité de données de service IP, L_p

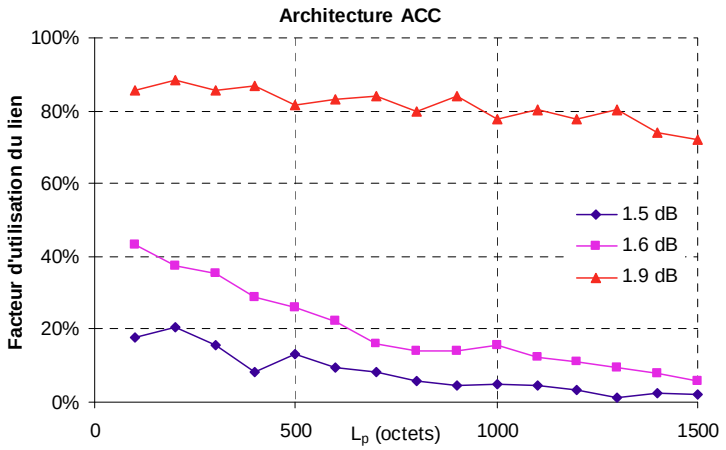

Figure 6-2 : Performance du service à fiabilité non contrôlée conventionnelle avec compression d'en-têtes (architecture ACC) en fonction de L_p

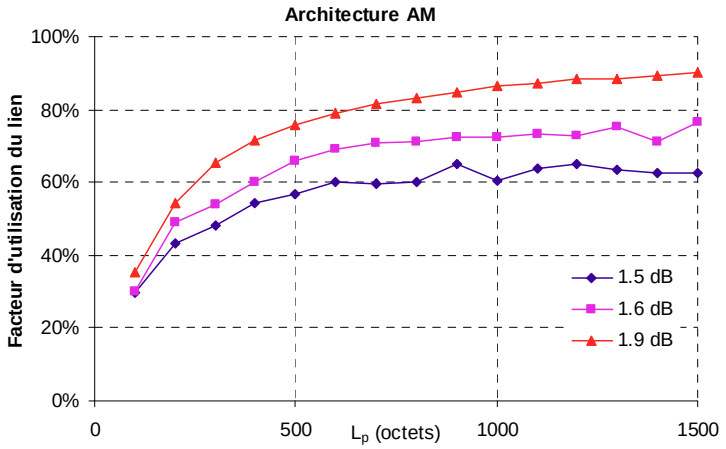

Figure 6-3 : Performance du service de fiabilité binaire non contrôlée (architecture AM) en fonction de L_p

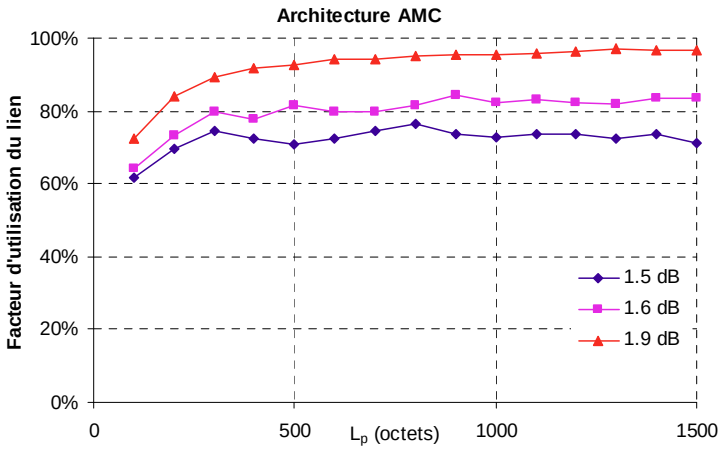

Figure 6-4 : Performance du service de fiabilité binaire non contrôlée avec compression d'en-têtes (architecture AMC), en fonction de L_p

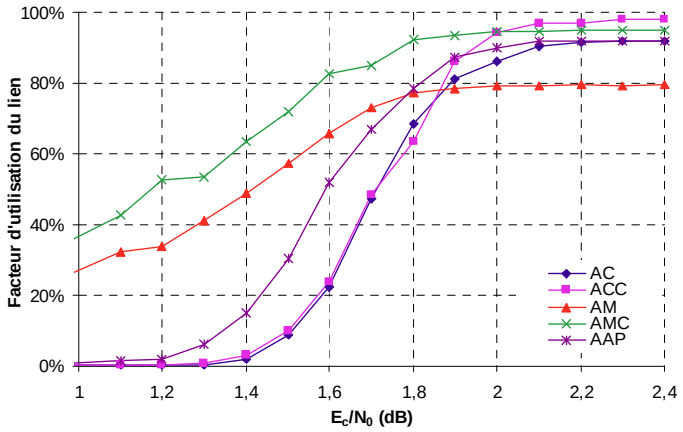

Figure 6-5 : Comparaison des performances des architectures en fonction du E_c/N_0 pour L_p=576 octets

6.1.4.3 Interprétation des résultats

Dans les deux premières séries de courbes (Figure 6-1, architecture AC et Figure 6-2, architecture ACC), on peut s'apercevoir que le meilleur facteur d'utilisation n'est pas obtenu pour la plus grande valeur de L_p. Ceci n'est pas surprenant car le taux d'erreur est ici directement relié au taux de rejets de paquet, dépendant également de L_p. Pour certaines courbes (notamment la Figure 6-2 pour E_c/N_0=1.9 dB), des ondulations apparaissent nettement. Par un modèle analytique simple dans lequel les erreurs sont uniformément distribuées, nous avons pu vérifier que celles-ci ne proviennent pas de l'imprécision des simulations, mais de la dépendance simultanée de l'efficacité d'encapsulation et du taux d'erreur résiduel en fonction de L_p.

L'effet bénéfique de la compression d'en-têtes est facilement identifiable, en particulier lorsque L_p est petit, sur les Figure 6-2 et Figure 6-4. Lorsque l'architecture déploie MPHP (Figure 6-3, architecture AM et Figure 6-4, architecture AMC), les courbes démontrent que les meilleurs facteurs d'utilisations sont obtenus avec l'augmentation de L_p. Ceci indique que le taux d'erreur symbole effectif avec MPHP n'est plus lié aux couches intermédiaires de niveau liaison, mais à la seule fiabilité de la liaison satellite.

Lorsque les conditions de propagation sont supérieures à un certain seuil (E_c/N_0=2.0 dB), MPHP n'est plus rentable, et une partie de la bande est gâchée. En revanche, l'activation

simultanée de la compression d'en-tête améliore ce manque d'efficacité, permettant à environ 95% des données transportées par MPEG-2 TS d'être exploitées par l'application.

6.2 MÉCANISMES DE NIVEAU TRANSPORT OFFRANT UN SERVICE DE FIABILITÉ TOTALE

La seconde partie de ce chapitre est dédiée à la réalisation du service de transport à fiabilité totale lorsque les données sont délivrées au transport par le biais du service de livraison en mode binaire. Après une présentation succincte de la gestion de fiabilité proposée pour la mise en œuvre du service de fiabilité totale, nous verrons comment il est possible d'adapter le décodage FEC à ce niveau pour le rendre optimal dans notre approche. Nous aborderons ensuite le problème de la gestion des retransmissions qui se pose lors de décodages infructueux, car le récepteur ignore alors la quantité de symboles erronés. En l'absence de cette information, il lui est difficile d'estimer la quantité de données supplémentaire qu'il doit acquérir afin de réussir le décodage.

Les principales propositions et résultats mentionnés dans cette section ont été présentés dans [Arn04b].

6.2.1 ***Présentation des mécanismes nécessaires***

6.2.1.1 Gestion générale de la fiabilité

Les différentes études citées ainsi que les estimations réalisées au cours du chapitre 3 montrent qu'une fiabilisation Hybride ARQ de type II est la plus efficace en terme d'utilisation de la ressource satellite. Ce critère de performances étant essentiel dans notre approche, nous privilégions le choix de cette solution pour la fiabilité totale.

6.2.1.2 Codage FEC

Le second choix de mise en œuvre porte sur le type de codage, MDS ou LDPC. Dans les architectures avec MPHP, les erreurs peuvent être des corruptions de bits (que le protocole de transport ne peut pas identifier ou comptabiliser), ou des effacements (consécutifs à des pertes de paquets) pour lesquels les positions sont connues (grâce au numéro de séquence des paquets effectivement reçus). Il est donc impossible de procéder au décodage par effacement car les positions de certaines erreurs peuvent être inconnues. Comment choisir le codage et décodage adéquat dans ces conditions ?

6.2.1.2.1 Codes MDS construits sur GF(2^8)

Une première solution consiste à utiliser un code MDS (p.ex. Reed-Solomon) construit sur GF(2^8), et de procéder au décodage par un algorithme corrigeant les erreurs. Pour ce faire, les symboles effacés sont remplacés par des symboles aléatoires. Il est possible de corriger jusqu'à $(n-k)/2$ erreurs dans ce mode, mais les positions des effacements ne sont pas prises en compte.

Une amélioration de l'efficacité du codage est réalisable grâce à certaines formes d'algorithmes de décodage pouvant à la fois corriger les erreurs et les effacements. Une implémentation de ce type (pour des codes Reed-Solomon) peut être trouvée dans [ECC] ; le principe de l'algorithme est mentionné par exemple dans [Cla82]. Si e est le nombre d'effacements et t le nombre d'erreurs dans un mot de code, le décodeur est assuré de corriger correctement un mot de code pourvu que la condition suivante soit respectée :

$$2.t + e \leq n - k \qquad (6.2)$$

6.2.1.2.1.1 Déclenchement du décodage

Un nouveau problème se pose en raison de l'observation suivante. Au moment du décodage, il existe toujours une probabilité pour qu'un mot de code, formé de $k+a$ symboles reçus, soit confondu avec un autre mot de code valide. En appelant SER le taux moyen d'erreur symbole dans ce mot de code, cette probabilité, $Pr_{ND}(k, a)$, est donnée par :

$$\Pr_{ND}(k,a,SER) = \left(\sum_{i=a+1}^{k+a} C_{k+a}^{i} SER^{i} (1-SER)^{k+a-i} \right) . \frac{1}{2^{8a}} \qquad (6.3)$$

La probabilité est obtenue en multipliant les probabilités élémentaires des deux évènements :

- il y a eu au moins $a+1$ symboles erronés sur les $k+a$ reçus ;
- la configuration des erreurs a entraîné la confusion avec un autre mot du code (probabilité égale à $1/2^{8a}$ pour un code sur GF(2^8)).

Le problème se pose donc lorsque peu de symboles (*i.e.* paquets) au-delà de k ont été reçus, c'est-à-dire lorsque a est voisin de (ou égal à) 0.

Quand a est égal à 0, la probabilité de non-détection est celle qu'il y ait eu au moins un symbole reçu en erreur dans le mot. En cas d'erreur résiduelle sur la liaison, cette probabilité est donc importante. Lorsque a augmente, la probabilité diminue rapidement, comme le montre la Figure 6-6, tracée pour k=16. L'augmentation de k (Figure 6-7) amène les courbes à converger plus rapidement vers leur asymptote, mais la valeur de celle-ci reste indépendante de k (elle vaut $1/2^{8a}$).

En pratique, la non détection d'un ou plusieurs mots de code erronés peut n'avoir aucune conséquence selon la réussite du décodage des autres mots du bloc. En effet, si le décodage d'au moins un autre mot de code s'avère défectueux, une retransmission doit avoir lieu quel que soit le résultat des autres décodages. Suite à la retransmission, un nouveau décodage des L mots de code du bloc peut être effectué. La probabilité estimée par (6.3) peut donc être considérée comme un pire cas, mais nous l'envisagerons par précaution. Aussi, nous imposons que le décodage en mode erreurs et effacements ne se fasse qu'à partir de k+3 paquets reçus par bloc FEC.

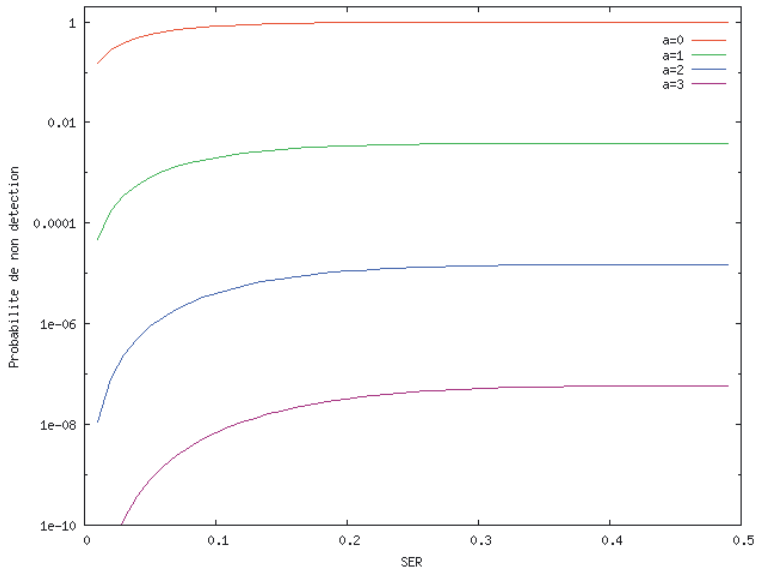

Figure 6-6 : Probabilité de non-détection d'un mot de code erroné pour k=16, a=0, 1, 2, 3, en fonction du taux d'erreur symbole, SER

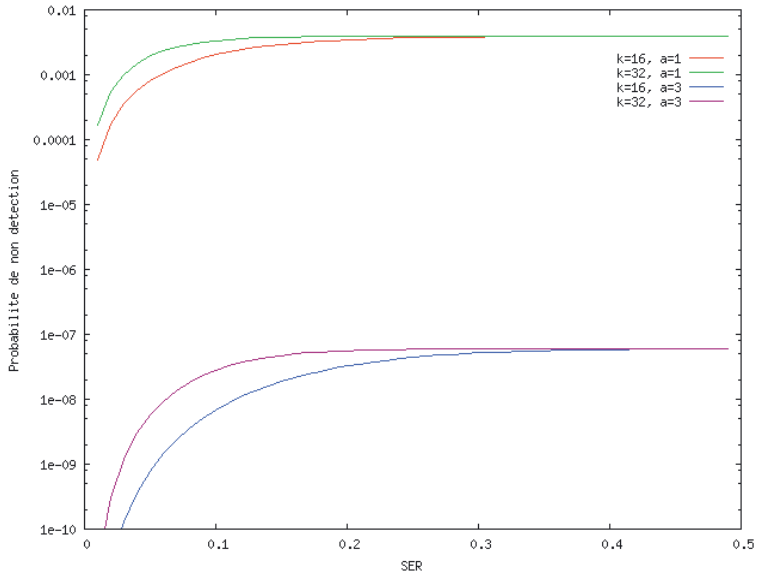

Figure 6-7 : Probabilité de non-détection d'un mot de code erroné pour k=16,32, a=1,3, en fonction du taux d'erreur symbole, SER

L'inconvénient évident de cette solution utilisée telle quelle serait de devoir recevoir k+3 paquets par bloc, sans prendre en compte le fait que les k premiers paquets puissent être reçus sans erreur. Pour ne pas imposer une telle contrainte, qui diminuerait l'efficacité de la transmission, nous proposons d'ajouter un mécanisme de détection d'erreurs (par CRC-32) au sein de chaque paquet, portant sur l'intégralité de l'unité de données de service de transport. L'objet de cette détection d'erreurs est de permettre le décodage (seulement par effacements) lorsque les paquets reçus ne sont pas entachés d'erreur.

Après le codage RS et l'intégration des données de parité du CRC, un bloc FEC se présente de la façon suivante (Figure 6-8) :

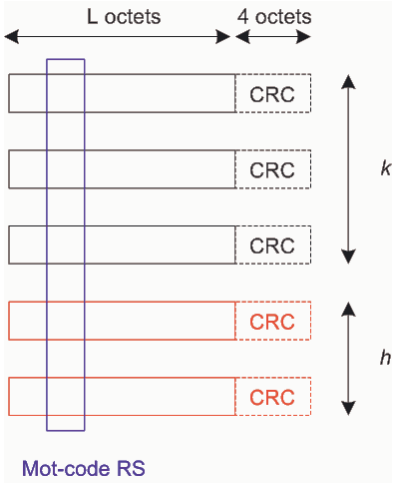

Figure 6-8 : Représentation d'un bloc FEC après codages RS et CRC

En appelant R le nombre de paquets reçus dans le bloc, le déclenchement du décodage serait ainsi réalisé selon la procédure schématisée en Figure 6-9 :

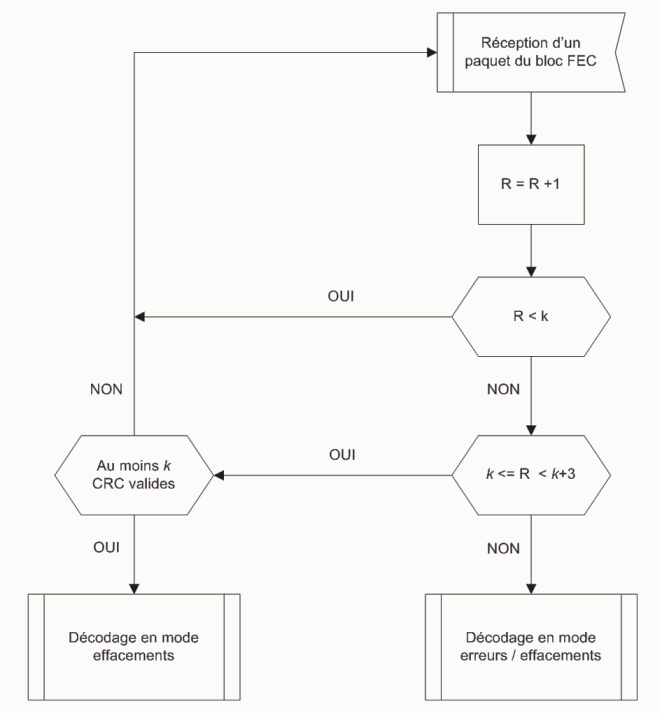

Figure 6-9 : Procédure de déclenchement du décodage

6.2.1.2.2 Codes MDS construit sur GF(2^{16})

Une piste de recherches serait d'utiliser la même méthode que précédemment, mais avec un code MDS construit sur GF(2^{16}). Outre le fait de pouvoir augmenter avantageusement le paramètre k, un tel code permettrait une amélioration supplémentaire.

Plutôt que d'adopter un CRC-32 classique (sur GF(2)) comme code détecteur d'erreurs, un code cyclique, construit sur GF(2^{16}) peut, lui aussi, être utilisé. Sur ce corps, les paramètres du codes deviennent (k=L/2, n=(L/2)+2). L'opération de codage du CRC puis du code MDS est alors un code produit (*i.e.* par lignes et par colonnes). Ce code confère aux 4 dernières colonnes du bloc FEC la propriété d'être elles-mêmes des mots de codes MDS. Dans les cas où une partie des données de CRC serait corrompue, le décodage selon Hybride ARQ de type II pourrait être achevé avec un nombre minimal de retransmissions.

L'inconvénient majeur de cette solution est lié à la complexité, car le décodage logiciel en mode erreurs/effacements n'est pas réalisable avec les implémentations rapides citées dans le chapitre 3. Il n'est toutefois pas exclu qu'un tel décodage, associé à une mise en œuvre adaptée, puisse être élaboré au cours des prochaines années.

6.2.1.2.3 Codes LDPC

Le décodage de codes LDPC pour erreurs et effacements constitue un autre axe de recherche prometteur. Par exemple, dans une note de Mitzenmacher [Mit98], l'analyse des codes LDPC est généralisée afin de proposer une classe de codes optimale pour un décodage mixte erreurs/effacements. La question de savoir comment construire précisément ces codes n'est toutefois pas encore résolue.

En raison des arguments avancés dans ces deux derniers paragraphes, la suite de l'étude et les évaluations seront restreintes aux codes MDS sur GF(2^8). Par commodité, les simulations présentées dans la section 6.2.1.3 seront réalisées pour une valeur de k fixée à 16.

6.2.1.3 Gestion des retransmissions

6.2.1.3.1 Politiques de retransmissions

6.2.1.3.1.1 Architecture AC

Pour les codes MDS décodés en mode effacements (adaptés à l'architecture AC), le décodage d'un bloc FEC est inutile tant qu'au moins k paquets ne sont pas parvenus au niveau du protocole de transport. Quand ce nombre n'est pas atteint après la première série de transmissions, les récepteurs doivent demander la transmission de paquets de parité supplémentaires. Le nombre de paquets demandés peut être exactement égal au nombre manquant de paquets pour atteindre les k requis.

Une autre solution, peut-être plus intéressante, est que chaque récepteur mesure son taux de pertes de paquets, PLR, pour le prendre en compte dans la demande de nouvelles transmissions. Une telle politique n'a de sens que si le taux de pertes de paquets ne varie pas, ou très peu, pour chaque récepteur. Si après un certain nombre de transmissions e paquets manquent encore au récepteur pour en posséder k, il demande ainsi la transmission de e/(1-PLR) paquets supplémentaires. L'objectif de cette technique est d'espérer qu'après une seule demande de réparation, les paquets de redondance supplémentaires transmis suffiront pour la réussite du décodage. Cette technique a déjà été évoquée et permet de réduire significativement le nombre de cycles de retransmission sans augmenter la quantité d'informations envoyées [Gos98]. Elle permet également de diminuer la charge de calcul requise à la source.

La première solution a l'avantage de garantir un nombre optimal de paquets transmis dans le réseau. Au contraire, avec la seconde alternative, les délais moyens d'attente peuvent être réduits ainsi que le nombre de messages envoyés sur la voie de retour.

6.2.1.3.1.2 Architecture AM

Pour le décodage erreurs/effacements approprié à l'architecture AM, la problématique précédente est encore plus difficile à résoudre car la réussite du décodage est imprévisible (sauf si k vérifications de CRC dans un bloc s'avèrent correctes, ce que nous excluons provisoirement). De même, l'estimation de la qualité du canal est impossible car le nombre d'erreurs résiduelles après le décodage (s'il a échoué) est inaccessible. En réponse à la problématique, nous proposons de simplement transmettre un nombre constant de nouveaux paquets de redondance (G) après chaque décodage infructueux, sauf lorsque le nombre de paquets reçus est insuffisant. G est appelé *granularité*.

6.2.1.3.2 Évaluation

Pour procéder à l'évaluation des différentes méthodes, nous faisons l'hypothèse que les performances du protocoles sont contraintes par le plus mauvais récepteur du groupe, c'est-à-dire celui pour lequel le rapport signal à bruit de la liaison satellite (E_c/N_0) est le plus faible. Ceci a été justifié lors de l'évaluation des mécanismes de fiabilisation en contexte d'erreurs hétérogènes (cf. 3.1.2). L'évaluation a été faite par le simulateur de liaison DVB-S sur une moyenne de 10 simulations, en lui intégrant les mécanismes de codage correcteur de niveau paquets au sein d'une couche de transport. G a été choisi égal à 2 et à 10 dans l'architecture AM et la longueur des datagrammes IP a été fixée à 1500 octets. Pour limiter les temps de simulations parfois très longs, les simulations ont été arrêtées au-delà de 100 cycles de retransmissions.

Les critères mesurés portent sur le nombre de paquets transmis équivalents[6] (Figure 6-10) et le nombre de cycles moyens (Figure 6-11) pour aboutir au succès du décodage du bloc. Pour la première évaluation, on observe que l'architecture AM permet de diminuer considérablement le nombre de paquets transmis lorsque le rapport E_c/N_0 est inférieur à 1.8 dB. Par exemple, pour E_c/N_0 égal à 1.5 dB, le nombre de paquets requis est divisé par 30 dans l'architecture AM. Le fait d'augmenter la granularité joue assez peu sur ces résultats, sauf lorsque le rapport E_c/N_0 est suffisamment élevé (supérieur à 1.8 dB) ; seul dans ce cas, une nette inefficacité apparaît.

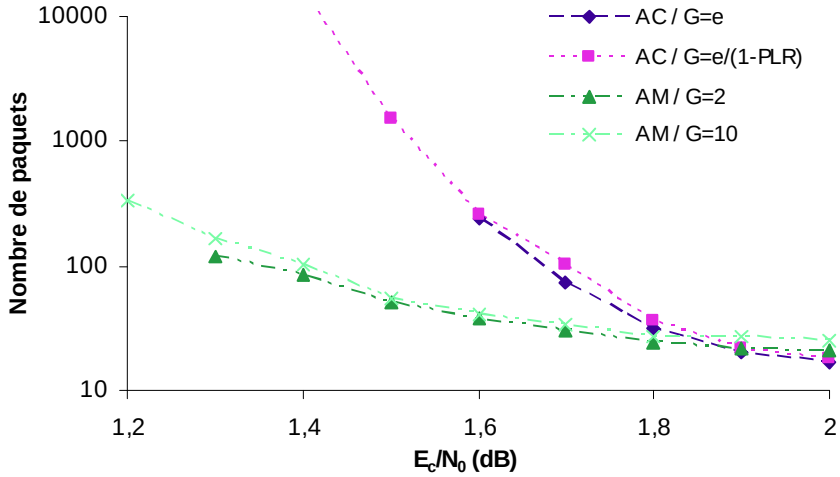

Figure 6-10 : Comparaison du nombre moyen de paquets à transmettre pour le succès du décodage entre les architectures AC et AM en fonction de la granularité G

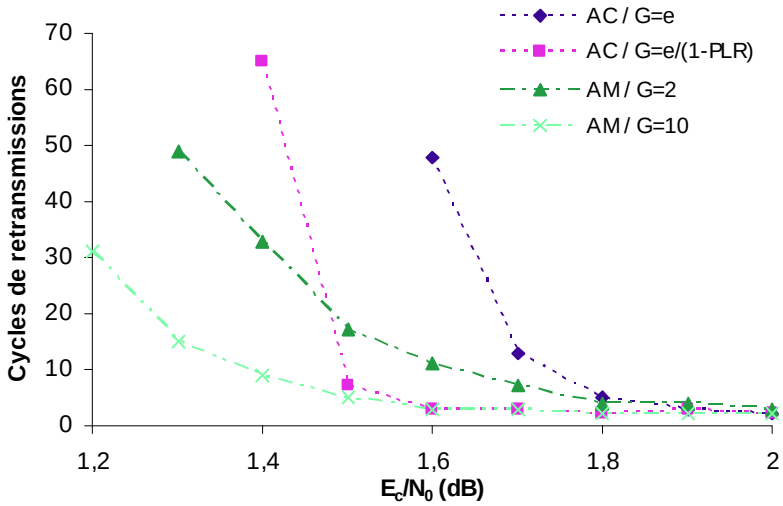

Figure 6-11 : Comparaison du nombre moyen de cycles de retransmissions nécessaires pour le succès du décodage entre les architectures AC et AM en fonction de la granularité G

L'estimation du second critère d'efficacité conduit au même type de conclusion que les auteurs de l'étude [Gos98], car l'augmentation de la granularité améliore nettement les résultats de ce critère alors que la quantité de données transmise reste pratiquement constante. Une optimisation simultanée sur les deux paramètres étant impossible, un équilibre doit être trouvé. Celui-ci pourrait consister à introduire une fonction de coût dépendante des préférences du fournisseur de services qui exploite la liaison satellite.

[6] Ce nombre prend en compte les données introduites par le codage MPHP dans l'architecture AM

6.2.1.4 Suppression des NACK

La technique de suppression de NACK originale ne convient pas exactement dans notre contexte car en l'absence d'un satellite régénérant, la transmission de NACK en multipoints « remonte » pratiquement jusqu'à la source. Une transmission de NACK en point à point à la source semble donc suffisante.

L'adaptation proposée ici consiste à faire jouer à la source un rôle de relais des NACK, en signalant à l'ensemble du groupe (sur le trafic transmis sur la voie aller) quels blocs d'informations seront sujets à des réparations. Plus explicitement, au cours de la transmission, la source indique régulièrement combien de paquets de parité seront transmis dans chacun des blocs. Lorsque des récepteurs demandent des réparations, ce paramètre, appelé par la suite *n'*, est augmenté, jusqu'à la réussite de tous les décodages. Ainsi, quand les récepteurs ne parviennent pas à décoder un bloc, ils n'envoient la demande de réparation qu'après un certain délai tiré aléatoirement, et à la condition que *n'* soit inférieur à la valeur indiquée par la source. Sinon, ils annulent leur demande. Il leur suffit alors d'attendre la réception de nouveaux symboles dans le bloc pour procéder à une nouvelle tentative de décodage.

6.2.1.5 Anticipation des demandes de retransmissions

Lorsqu'un récepteur s'aperçoit, en cours de réception, de la pertes de données systématiques d'un bloc, il peut prédire le fait qu'après la fin de cette phase, il ne pourra réussir le décodage. En ne le déclenchant pas, il pourra ainsi éviter de gâcher les ressources de traitement qui auraient été nécessaires.

Mieux, il peut également anticiper la demande de réparation avant même la fin de cette phase. Cette idée a été mentionnée par les auteurs de [Yoo99]. Par exemple, si après la transmission de 10 paquets le récepteur n'en a reçu qu'un parmi les *k* requis et que *n'-k* (supposé ici inférieur à 9) paquets additionnels seront transmis, il sait qu'au moins 9-*n-k'* manqueront pour atteindre *k* paquets.

Grâce à ce procédé, la source peut programmer les cycles de réparation successifs le plus tôt possible. Les situations d'attente de réception de demandes de réparation sans transmission sont ainsi limitées. Elles engendrent des délais supplémentaires néfastes à la qualité du service.

6.2.1.6 Prise en compte de plusieurs blocs pour les demandes de transmissions additionnelles

Les réparations pouvant être demandées pour n'importe quels blocs, il n'y a pas, *a priori*, de raison pour lesquelles le nombre de paquets demandés dans chaque bloc soit strictement égal. D'un autre côté, il peut être fastidieux que les demandes soient formulées individuellement pour chaque bloc. Cet argument de simplicité est mis en avant par RMDP [Riz97], dans lequel la valeur de *n'* demandée pour les réparations est la valeur maximale sur l'ensemble des blocs. Ce choix permet non seulement de réduire la taille des messages envoyés sur la voie de retour, mais il augmente aussi la probabilité que d'autres récepteurs suppriment leur demande de réparation.

Dans notre étude, le critère privilégié étant l'efficacité de la transmission sur la voie aller, nous préférerons opter pour la solution indiquant explicitement à la source le nombre de paquets de réparation désirés dans chaque bloc.

6.2.2 ***Mise en oeuvre du service de fiabilité totale avec le service de livraison en mode binaire***

La mise en œuvre complète qui est proposée se base sur les mécanismes discutés en 6.2.1. On suppose qu'un certain nombre d'informations (nombre de blocs composant le message applicatif,

paramètres du codage FEC) a été communiqué à l'avance par une signalisation transmise hors bande (par exemple, pendant l'annonce de la session). Deux types de paquets sont définis : les messages de données (DATA) et les messages de demande de transmissions additionnelles (REPAIR). Ces messages sont transmis en mode point à point à la source. Les deux types de messages sont différenciés par le bit de poids faible du premier octet de leur en-tête.

6.2.2.1 Variables utilisées

La mise en œuvre définit un certain nombre de variables connues implicitement, communiquées ou mises à jour régulièrement. La signification de l'ensemble des variables est donnée dans le Tableau 6-2.

Variable	Signification	Valeurs possibles	Remarque
G	Granularité des retransmissions	-	Non transmis (choix d'implémentation)
T_E	Délai maximum sans réception avant la fermeture de la session par la source	-	
T_R	Délai maximum sans réception avant la transmission de message REPAIR par récepteur	-	
T_S	Délai (aléatoire) utilisé pour la suppression de message	cf. 6.2.1.4	
k	Valeur k du code FEC	1 à 254	Transmis hors bande
n	Valeur n du code FEC	k+1 à 255	
S	Taille du message applicatif	-	
B	Nombre de blocs composant le message	-	
B_e	Nombre de blocs entrelacés	1 à B	
Session ID	Identifiant de la session	0 à 127	Transmis hors bande / indiqué dans l'en-tête des paquets
i	Indice du bloc FEC du paquet	0 à B-1	Indiqués dans l'en-tête des paquets DATA / Signification locale à la source
j	Position du paquet dans le bloc	0 à n-1	
n'_i	Valeur de n' courante pour le bloc i	k à n	
T_i	Compteur de paquets transmis dans le bloc i	0 à n	Signification locale à la source
R_i	Compteur de paquets reçus dans le bloc i	0 à n	Signification locale au récepteur
V_i	Compteur de paquets validés dans le bloc i	0 à n	
A_i	Variables d'anticipation des retransmissions	cf. 6.2.2.4.2	
B_R	Nombre de blocs à réparer	0 à B-1	Indiqués dans l'en-tête des paquets REPAIR
L	Liste des demandes de réparations	cf. 6.2.2.2	

Tableau 6-2 : Liste des variables utilisées pour le service fiable avec le service de livraison en mode binaire

6.2.2.2 Format des paquets DATA

Les paquets de données DATA (Figure 6-12) contiennent les symboles d'information du codage FEC. Ils comportent, en plus, un en-tête fixe de 8 octets, identifiant le type du paquet et l'identifiant de session, la position du bloc FEC dans l'objet à transférer, et la position du paquet dans le bloc. Un CRC-32, calculé uniquement sur les données, est ajouté à la fin du paquet.

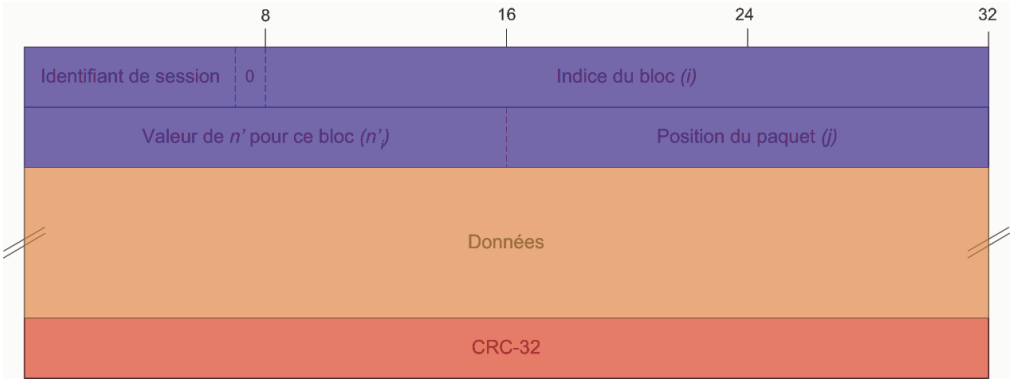

Figure 6-12: Format des messages DATA

6.2.2.3 Format des paquets REPAIR

Les paquets de données REPAIR (Figure 6-12) contiennent les demandes de réparation adressées par les récepteurs à la source. L'en-tête, de 4 octets, est fixe. Il indique le type de paquet, l'identifiant de session, ainsi que le nombre de blocs pour lesquels des réparations sont demandées. La liste des demandes de réparations est ensuite indiquée. Cette liste informe la source du nombre de paquets demandés dans chaque bloc. Elle est constituée d'un nombre variable d'éléments (appelés *entrées* par la suite) correspondant au champ précédent. Chacune de ces entrées comporte un couple de valeurs, (i, n'_i), codées respectivement sur 3 et 2 octets. La première valeur spécifie le numéro du bloc pour lesquels les retransmissions sont demandées, alors que la seconde spécifie le nombre de paquets de réparation demandé pour ce bloc.

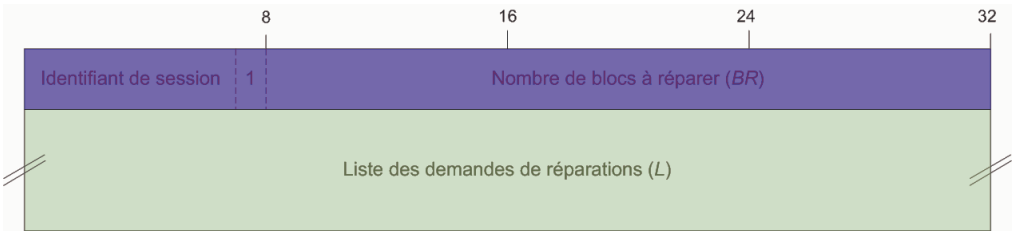

Figure 6-13 : Format des messages REPAIR

6.2.2.4 Détails des opérations

Pour la source comme pour le récepteur, une machine à états finis illustrant le fonctionnement général du protocole est donnée. L'objectif n'étant pas ici de spécifier formellement le protocole (par exemple pour le valider), mais seulement d'aider à sa présentation, cette représentation sera simplifiée. Le détail des opérations réalisées dans chaque état sera précisé dans le texte.

6.2.2.4.1 Source

Pour chaque session, la source effectue de traitements de différentes natures selon l'un des 6 états dans lequel elle se trouve : l'état *préparation*, l'état *transmission*, le *traitement de messages REPAIR*,

la réparation, l'état d'*attente,* et la *fin de session*. La machine à états finis de la source est illustrée dans la Figure 6-14.

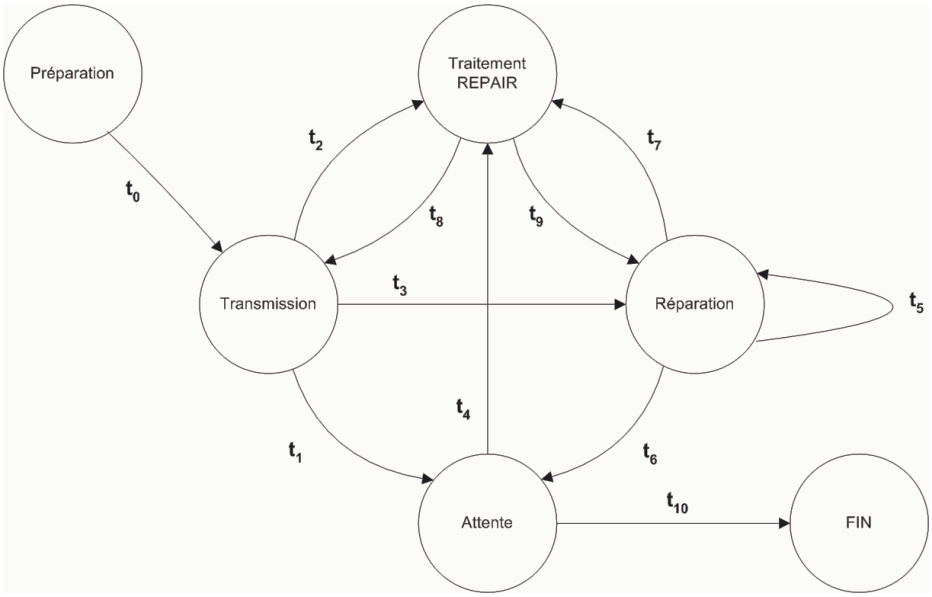

Figure 6-14 : Machine à états finis de la source

6.2.2.4.1.1 Préparation

La session de transport débute en fournissant (en plus des paramètres réseau) les paramètres de codage (*n*, *k*), le message applicatif à transmettre, la taille L_m des messages DATA (en-têtes et CRC non inclus) et le paramètre B_e. Le serveur, après avoir déterminé la taille *S* du message applicatif, le fragmente en $\frac{S}{kL_m}$ blocs, de *k* paquets. Si le résultat de la division n'est pas entier, les symboles manquants pour former le dernier bloc sont complétés par des octets positionnés à 0x00. Le codage FEC (*n*, *k*) des blocs est ensuite effectué. Si la taille du message encodé ne permet pas de le conserver en mémoire, il est stocké sur le disque (le message encodé étant *n*/*k* fois plus volumineux que le message original). Enfin, pour *i* allant de 0 à *B*-1, les n'_i sont initialisés à *k*-1 et les T_i sont initialisés à 0. Cette phase de préparation complétée, la source passe ensuite dans l'état de transmission (transition $\mathbf{t_0}$).

6.2.2.4.1.2 Transmission

Dans cet état, la source procède à la transmission des données par l'entrelacement. Pour chaque paquet, le CRC-32 est calculé et l'en-tête est ajouté. Le paquet DATA ainsi formé est proposé au service du niveau inférieur (UDP ou IP). Pour le bloc *i* de chaque paquet transmis, T_i est incrémenté. Le serveur poursuit la transmission principale jusqu'à ce que T_i soit égal à *k* pour tout *i*.

Lorsqu'elle se trouve dans cet état, la source peut recevoir des messages REPAIR de la part des récepteurs, messages qu'elle traite alors immédiatement (transition $\mathbf{t_2}$). Si aucun de ces messages n'est reçu jusqu'à la fin de transmission, la source passe dans l'état d'attente par la transition $\mathbf{t_1}$. Sinon, un premier cycle de réparation est initié (transition $\mathbf{t_3}$).

6.2.2.4.1.3 Traitement REPAIR

Lorsque des paquets REPAIR sont reçus (pendant l'état de transmission, par la transition $\mathbf{t_2}$, pendant l'état de réparation, par la transition $\mathbf{t_7}$, ou pendant l'état d'attente par la transition $\mathbf{t_4}$), la

source met à jour ses valeurs locales des n'_i si et seulement si la valeur indiquée dans le paquet REPAIR est supérieure à sa valeur locale, et inférieure à n. Les autres demandes ne sont pas prises en compte. La source bascule ensuite dans l'état transmission si elle s'y trouvait précédemment (transition $\mathbf{t_8}$). Sinon, elle repasse dans l'état réparation par la transition $\mathbf{t_9}$.

6.2.2.4.1.4 Réparation

Pour ne pas pénaliser les bons récepteurs qui reçoivent l'ensemble des données systématiques, la phase de réparation ne débute qu'à la fin de la transmission principale (transition $\mathbf{t_3}$). Ensuite, la réparation a lieu par cycle consécutif, pour lesquels tous les paquets demandés en supplément (au cours de la transmission principale ou du cycle précédent) sont envoyés. Si, pendant le cycle, des messages REPAIR sont reçus (transition $\mathbf{t_7}$), un nouveau cycle est programmé à la fin du cycle de réparation en cours (transition $\mathbf{t_5}$). Il débutera immédiatement après le cycle en cours. Sinon, la source peut basculer dans l'état d'attente (transition $\mathbf{t_6}$).

6.2.2.4.1.5 Attente et fin de session

Un *timer*, de durée T_E, est armé dès que la source entre dans cet état. Si le *timer* expire sans réception de message(s) REPAIR, la source clôture la session en passant dans l'état fin de session (transition $\mathbf{t_{10}}$). Sinon, un nouveau cycle de réparation a lieu (transition $\mathbf{t_4}$ puis $\mathbf{t_9}$).

6.2.2.4.2 Récepteurs

Les récepteurs agissent selon 8 états : l'état initial, appelé début de session, l'état réception, le décodage en mode effacements, le décodage en mode effacements et erreurs, l'ajout d'une entrée dans la liste L des demandes de réparation, la suppression/modification d'une entrée de la liste des demandes de réparation, la transmission de message REPAIR, et enfin l'état fin de session. La machine à états finis du récepteur est donnée en Figure 6-15.

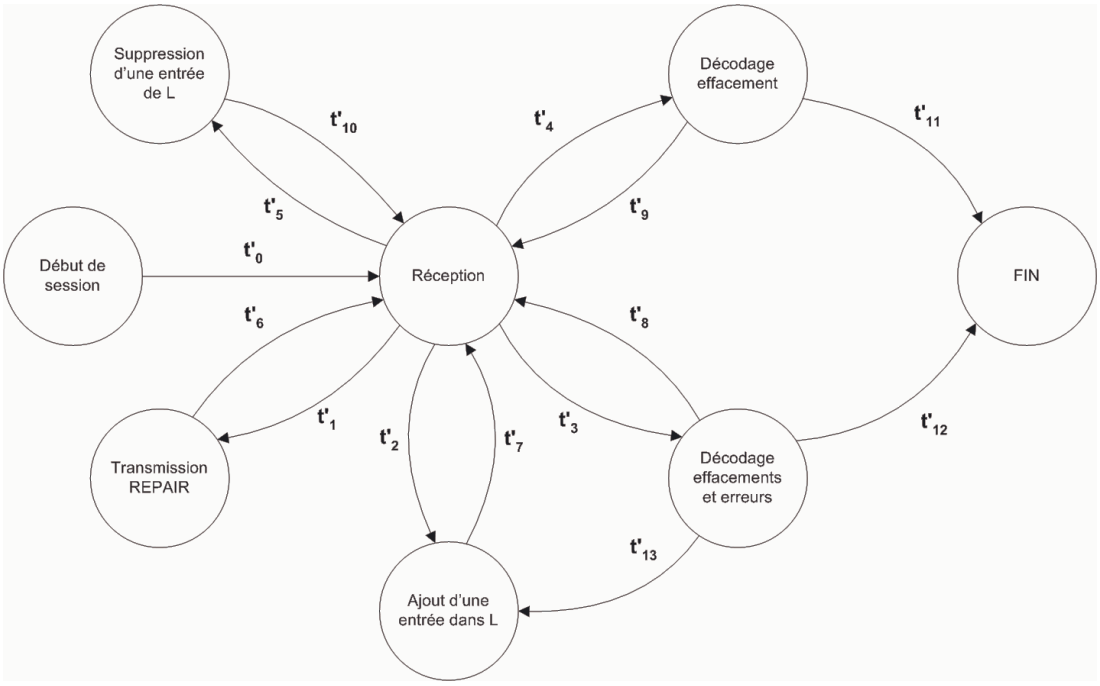

Figure 6-15 : Machine à états finis du récepteur

6.2.2.4.2.1 Début de session

Après l'abonnement à la session, le récepteur initialise R_i et V_i à 0, pour i allant de 0 à B-1. Il passe ensuite dans l'état réception DATA par la transition **t'$_0$**.

6.2.2.4.2.2 Réception de paquet DATA

Après la réception de chaque paquet DATA, le récepteur examine si le bloc correspondant a déjà été décodé ou non. Si c'est le cas, le récepteur ne tient pas compte du paquet. Sinon, R_i est incrémenté. Le récepteur procède ensuite à la vérification du CRC du paquet ; si le CRC est correct, la variable V_i est aussi incrémentée. La valeur locale de n'_i est mise à jour en fonction de la valeur indiquée dans le paquet. A_i est également mis à jour grâce à l'équation :

$$A_i = k + j + 1 - R_i - n'_i \tag{6.4}$$

Si la valeur calculée de A_i est strictement positive, une anticipation de demande de réparation est faite. La réparation de A_i paquets est demandée pour le bloc i, en créant une nouvelle entrée dans la liste L puis en programmant la transmission d'un message REPAIR. Ceci arrive aussi après une période T_R sans réception de paquet DATA. Le nombre de paquets demandés dans ce cas peut par exemple être celui que le récepteur estime avoir perdu en fonction du débit de réception moyen des paquets précédents. Ces cas correspondent aux deux conditions de réalisation de la transition **t'$_2$**.

6.2.2.4.2.3 Transmission REPAIR

Lorsque le *timer* de retransmission expire (transition **t'$_1$**), le message REPAIR programmé précédemment est envoyé. Au contraire, si le récepteur n'a reçu aucun paquet DATA après une période égale à T_E, un message REPAIR est créé. La liste du message REPAIR indique, pour chaque bloc i incomplet, le nombre de paquets manquants pour décoder le bloc en l'ajoutant à la dernière valeur de n'_i connue par le récepteur.

Après la transmission, le récepteur bascule dans l'état réception (transition **t'$_6$**).

6.2.2.4.2.4 Ajout d'une entrée dans la liste des demandes de réparations

Une nouvelle entrée dans la liste des demandes de réparation est créée dans trois cas :

- le décodage en mode effacements/erreurs d'un bloc n'a pas pu être réalisé (transition **t'$_{13}$**) ;
- le récepteur peut anticiper la demande car trop de paquets dans un bloc ont été perdus (première condition amenant la transition **t'$_2$**) ;
- le récepteur n'a reçu aucune paquet DATA après la durée T_R (seconde condition pour la transition **t'$_2$**).

Quand une nouvelle entrée a été créée, elle est ajoutée dans la liste des demandes de réparations L. Si la liste est vierge, un nouveau message REPAIR est créé. À cet instant, un *timer* de durée T_R est armé (cf. 6.2.1.4). Si la liste comporte une ou plusieurs entrées, la nouvelle entrée est simplement concaténée à la fin. Enfin, si la liste comporte une entrée correspondante à l'un des blocs (mais que le message REPAIR n'a pas été envoyé), l'entrée est modifiée en conséquence.

Le récepteur bascule ensuite dans l'état de réception (transition **t'$_7$**).

6.2.2.4.2.5 Décodage en mode effacements

Lorsque pour un bloc, V_i est supérieur ou égal à k, le bloc est décodé en mode effacements pour les k premiers paquets validés (transition $\mathbf{t'_4}$). Pour le dernier bloc, le récepteur complète les symboles manquants par des octets à 0x00. La réussite du décodage est garantie dans ce mode avec les codes MDS.

Si le bloc décodé est le dernier bloc, le récepteur peut reconstituer l'objet et basculer dans l'état de fin de session (transition $\mathbf{t'_{11}}$). Sinon, il retourne dans l'état réception (transition $\mathbf{t'_9}$).

6.2.2.4.2.6 Décodage en mode effacements/erreurs

Si la condition précédente n'est pas respectée mais que R_i est supérieur ou égal à k+3 (cf. Figure 6-9), un décodage en mode effacements/erreurs est réalisé (transition $\mathbf{t'_3}$). Si le décodage échoue, une entrée dans la liste des demandes de réparations est créée pour le bloc i, avec un nombre de paquets égal à la somme de la dernière valeur connue de n'_i et du paramètre de granularité, G. Ceci correspond à la transition $\mathbf{t'_{13}}$.

Si le bloc décodé est le dernier bloc, le récepteur peut passer dans l'état de fin de session (transition $\mathbf{t'_{12}}$). Sinon, il retourne dans l'état réception (transition $\mathbf{t'_8}$).

6.2.2.4.2.7 Suppression / modification d'une entrée de la liste des demandes de réparations

Si pour un bloc i, la source indique un nombre n'_i supérieur ou égal à n'_i de la liste de réparation (transition $\mathbf{t'_5}$), le récepteur supprime de sa liste l'entrée correspondante. Si n'_i a été augmenté mais qu'il reste inférieur au nombre demandé par le récepteur, l'entrée de la liste est modifiée en conséquence. Le récepteur repasse ensuite dans l'état réception (transition $\mathbf{t'_{10}}$).

6.2.2.4.2.8 Fin de session

Lorsque l'ensemble des blocs FEC a été décodé, le récepteur se trouve dans l'état fin de session. Il peut alors quitter le groupe multipoints s'il le désire.

6.2.2.5 Limites de la mise en œuvre

La mise en œuvre proposée ne spécifie pas comment seraient faites certaines opérations, en particulier la gestion des entrées-sorties entre le système de mémoire et le disque. Les valeurs de n et k étant très différentes pour assurer une réussite systématique de décodage, une quantité très importante d'espace ($S.n/k$) pourrait être demandée si l'implémentation n'était pas soigneusement étudiée. De même, la charge de traitement consacrée aux décodages successifs pourrait être excessive pour certaines stations disposant de peu de ressources.

CONCLUSION

Le service de livraison en mode binaire, dont une mise en œuvre au niveau des couches basses a été proposée dans le chapitre 5, permet la réalisation de deux services de fiabilité assurés par le niveau transport : un service à fiabilité binaire non contrôlée, et un service de fiabilité totale.

Le service à fiabilité binaire non contrôlée convient pour certaines applications multimédia émergentes, reposant sur des formes particulières de codage source tolérantes aux erreurs binaires. De plus, lorsque les unités de données du service de transport sont suffisamment petites, la compression d'en-têtes améliore nettement les performances d'utilisation du lien. En perspective de ces travaux, il pourrait être intéressant d'examiner l'apport des architectures protocolaires AM et AMR au niveau applicatif, par exemple par une mesure de la qualité de restitution des média.

Le service de fiabilité totale élaboré est adapté aux formes d'erreurs qui peuvent être rencontrées avec le service de livraison en mode binaire. Le principe de base de la mise en œuvre repose sur le décodage en mode mixte erreurs/effacements. Ce type de décodage est intéressant dans la mesure où les pertes de paquets ne sont pas traitées comme de simples erreurs car leurs positions sont connues. Le décodage, ainsi réalisé, offre une capacité de correction améliorée. D'autres problématiques sont adressées dans la mise en œuvre. Certaines d'entre elles (le type de fiabilisation, le phénomène d'implosion à la source, etc…) sont bien identifiées par les travaux de recherches sur les protocoles de transport multipoints. Certaines autres, spécifiques au service de livraison en mode binaire, restent à développer. Enfin, au cours de l'étude, il a été souligné que l'optimisation unilatérale de l'efficacité d'utilisation de la bande pourrait amener à augmenter de façon significative d'autres paramètres, tels que la quantité de données émises sur la voie de retour, la charge de calcul et peut-être la quantité d'espace mémoire (mémoire vive, disque) requise au niveau des récepteurs. Seule une implémentation complète de l'ensemble des mécanismes, ainsi que des expérimentations poussées pourraient permettre de déterminer l'ensemble des compromis à accepter.

BIBLIOGRAPHIE

[3gp02] 3rd Generation Partnership Project, "AMR speech Codec; General description", TS 26.071 version 5.0.0, juin 2002.

[Ans00] "Telecommunications - Digital Transport of One-Way Video Signals - Parameters for Objective Performance Assessment", ANSI T1.801.03-2003, octobre 2003.

[Arn04a] "Multi-Protocol Header Protection (MPHP), a Way to Support Error-Resilient Multimedia Coding in Wireless Networks", F. Arnal *et al.*, 7th IEEE International Conference on High Speed Networks, juin 2004

[Arn04b] "Cross-Layer Reliability Management for Multicast over Satellite", F. Arnal *et al.*, International Journal of Computer Networks, special issue on IP over DVB networks, à paraître

[Cla82] "Error-Correction Coding for Digital Communications", G.C. Clark, J. B. Cain, pp. 214-219, Plenum Press, 1981.

[ECC] http://www.eccpage.com/

[Fab00] "Real-Time Video Communications over GPRS", S. Fabri *et al.*, IEE 3G2000, London, pp. 426-430, mars 2000.

[Flo97] "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", S. Floyd *et al.*, IEEE/ACM Transactions on Networking, Volume 5, Number 6, pp. 784-803, décembre 1997.

[Gos98] "Reliable Multicast and Integrated Parity Retransmission with Channel Estimation Considerations", D. Gossink, J. Macker, Globecom'98, novembre 1998.

[Gri99] "Robust Compression and Transmission of MPEG-4 Video" ACM Multimedia'99, S. Gringeri *et al.*, novembre 1999.

[GUR] http://www.icsi.berkeley.edu/~gurtov/corruption.html/

[Ham03] "Corrupted Speech Data Considered Useful", F. Hammer *et al.*, Proc. First ISCA Tutorial and Research Workshop on Auditory Quality of Systems, Mont Cenis, Germany, avril 2003.

[Iso01] "Overview of the MPEG-4 Standard", ISO/IEC JTC1/SC29/WG11, mars 2001.

[Itu01a] "Method for Objective Measurements of Perceived Audio Quality", recommendation ITU-R BS-1387, novembre 2001.

[Itu01b] Recommandation ITU-T, "Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs", ITU-T P.862, février 2001.

[Itu98] Recommandation ITU-T, "Video Coding for Low Bit Rate Communication", ITU-T H.263, février 1998.

[Kha03] "Cross-Layer Protocol Design For Real-Time Multimedia Applications Over 802.11b Networks", S.A. Khayam *et al.*, IEEE International Conference on Multimedia and Expo (ICME), juillet 2003.

[Koh04] IETF Internet Draft, "Datagram Congestion Control Protocol (DCCP)", E. Kohler, M. Handley, S. Floyd, draft-ietf-dccp-spec-07.txt, juillet 2004.

[Lar04] IETF RFC, "The Lightweight User Datagram Protocol (UDP-Lite)", L-A. Larzon *et al.*, RFC 3828, juillet 2004.

[Mit98] "A Note on Low Density Parity Check Codes for Erasures and Errors", M. Mitzenmacher, SRC technical note 1998-017, décembre 1998.

[Rei03] "MPEG-4 and UDP-Lite for Multimedia Transmission", R. Reine, G. Fairhurst, PGNet 2003, juin 2003.

[Riz97] "A Reliable Multicast Data Distribution Protocol Based on Software FEC Techniques", L. Rizzo, L. Vicisano, in proceedings of HPCS '97, Greece, juin 1997.

[Sin01] "Performance Evaluation of UDP Lite for Cellular Video", A. Singh, A. Konrad, A.D. Joseph, NOSSDAV'01, juin 2001.

[Yoo99] "Adaptive Reliable Multicast", J. Yoon, A. Bestavros, and I. Matta, Tech. Rep. No. BU-CS-1999-012, septembre 1999.

Conclusion

Résumé des travaux

Après avoir introduit le contexte général de l'étude, ce mémoire de thèse situe les principales difficultés que rencontrent les communications multipoints dans les réseaux par satellite géostationnaire. Parmi eux, nous nous sommes intéressés au problème particulier de la gestion de la fiabilité dans une approche orientée système. Les mécanismes mis en jeu dans cette gestion ont d'abord été présentés de façon générale. Ensuite, une évaluation des mises en œuvre de techniques de fiabilisations retenues dans la partie précédente a été menée au niveau du protocole de transport. L'étude a mis en évidence les compromis à accepter selon le choix du codage et de sa mise en œuvre. Ces compromis portent essentiellement sur la quantité de données à transmettre dans le réseau, sur la robustesse aux erreurs prolongées provenant de *fading,* et sur le délai de service.

Cette première partie du mémoire a fait apparaître un manque de coordination entre certains mécanismes déployés à différents niveaux de la communication. En particulier, la politique de rejet systématique des unités de données erronées, héritée des techniques associées aux réseaux terrestres, est au cœur du problème. Dès lors, on peut se demander si le déploiement des codages correcteurs d'erreurs à plusieurs niveaux ne pourrait pas être mieux exploité. Dans un contexte satellitaire où le coût de l'utilisation de la bande radiofréquence est élevé, une telle optimisation prend tout son sens. Pour arriver à cette optimisation, un service de livraison des données en mode binaire a été proposé dans la seconde partie du mémoire. L'étude de ce service a conduit à plusieurs propositions et résultats, objets des deux premières publications mentionnées en annexe.

1) Grâce au développement d'un logiciel de simulation de la chaîne de transmission DVB-S modélisant l'encapsulation de datagrammes IP dans les conteneurs MPEG-2 TS par les méthodes MPE et ULE, l'évaluation des performances de fiabilité de la transmission IP a été entreprise.

2) Dans un second temps, le simulateur a permis d'identifier une architecture protocolaire favorable au service de livraison de datagrammes IP. L'idée directrice de cette architecture est non seulement d'empêcher le rejet des unités de données corrompues afin qu'elles parviennent au niveau applicatif du récepteur final, mais aussi de déployer une forme de protection inégale aux erreurs (UEP) sur la couche physique. Cette protection, additionnelle au codage canal habituel, doit couvrir simultanément pour chaque unité de données le plus grand nombre de couches protocolaires du plan utilisateur, afin de prévenir la corruption de leurs en-têtes. La réalisation pratique d'une telle architecture dans le cadre strict des normes DVB/MPEG-2 pose certains problèmes d'incompatibilité, mais des adaptations peu coûteuses en termes de besoin de calcul et de capacité mémoire semblent suffisantes.

3) L'exploitation de l'architecture pour deux services de transport a ensuite été proposée.

a) Le premier service, appelé *fiabilité binaire non contrôlée*, est exactement celui qu'offre le nouveau protocole UDP-lite. Ce service convient à une nouvelle classe d'applications multimédia dans lesquelles une forme particulière de codage source, tolérante aux erreurs binaires, est employée. Pour ces applications, l'architecture protocolaire intégrant UDP-lite peut augmenter la quantité d'information utilisable pour l'application, en particulier si un mécanisme auxiliaire de compression d'en-tête est déployé. Ainsi, en cas d'erreurs sur le lien satellite, le gain mesuré par simulation peut aller jusqu'à un facteur 5 sur le débit applicatif.

b) Le second service, de *fiabilité totale,* nécessite un décodage adapté, pour prendre en compte simultanément les erreurs et les effacements. L'évaluation des performances du décodage a montré qu'un gain entre 5 et 30 sur la quantité de données transmises peut être espéré. Une instanciation complète de ce service dans un composant de protocole de transport multipoints a été donnée. Elle répond aux problématiques énoncées en première partie.

Perspectives de recherche

Ce travail ouvre la voie à de nombreuses perspectives de recherche, dont la plupart ont été évoquées dans ce mémoire. Tout d'abord, un modèle d'erreurs intégrant des mesures expérimentales des variations temporelles du E_c/N_0 pourrait être employé par le simulateur. Un modèle de ce type pourrait permettre d'évaluer précisément les conséquences d'évènements météorologiques de courtes périodes sur le service IP. Ensuite, il serait envisageable de doter le simulateur du turbo-codage recommandé par DVB-RCS, voire du codage LDPC préféré dans les futurs systèmes DVB-S 2. Cette extension pourrait aussi permettre d'évaluer la fiabilité lorsque le serveur de contenus multipoints n'accède pas au réseau satellite par une *gateway* mais par un terminal du système DVB-RCS. Enfin, l'étude des performances du schéma de codage proposé par MPE-FEC dans la dernière recommandation de l'ETSI serait intéressante à mener. Elle pourrait amener à reconsidérer l'intérêt du codage FEC au niveau des récepteurs finaux.

Concernant la proposition d'architecture protocolaire permettant la livraison d'unités de données corrompues à l'application, un travail de validation sur une implémentation réelle serait essentiel. Il devrait permettre de confirmer les gains estimés par simulation, et pourrait révéler les choix de mises en œuvre les plus judicieux. Il pourrait aussi aider à l'élaboration de la spécification de MPHP et de son support pour les différent profils de protocoles.

Le troisième axe de recherches est l'évaluation du service de fiabilité binaire non contrôlé avec UDP-lite. En l'estimant directement au niveau applicatif par des méthodes appropriées et en la comparant avec une transmission reposant sur un service de fiabilité traditionnel (UDP), l'intérêt de l'architecture proposée pour ce service pourrait être explicitement mesuré.

Enfin, pour le service de fiabilité totale, il a été souligné que le décodage en mode erreurs et effacements des codes LDPC et des codes MDS sur $GF(2^{16})$ pourrait être l'objet d'études plus approfondies, afin de répondre aux problèmes de robustesse et de complexité. Pour terminer, l'implémentation, et la mesure des performances expérimentales du protocole de transport multipoints proposé seraient la dernière étape pour valider complètement l'ensemble de la proposition.

Annexe

Les travaux exposés dans ce mémoire ont donné lieu à plusieurs publications parues dans des conférences ou revues internationales :

"Cross-Layer Reliability Management for Multicast over Satellite", F. Arnal, L. Dairaine, J.Lacan, G. Maral, International Journal of Computer Networks, special issue on IP over DVB networks, décembre 2004.

"Multi-Protocol Header Protection (MPHP), a Way to Support Error-Resilient Multimedia Coding in Wireless Networks", F. Arnal, L. Dairaine, J. Lacan, G. Maral, 7th IEEE International Conference on High Speed Networks and Multimedia Computing (HSNMC'04), juin 2004.

"Reliable Multicast Transport Protocols Performances In Emulated Satellite Environnement Taking Into Account An Adaptive Physical Layer For The Geocast System", F.Arnal, A.Bolea-Alamanac, L.Castanet, L. Dairaine, M. Bousquet, L. Claverotte, G. Maral , R. Guttirez, International Workshop of COST Actions 272 and 280 ESTEC, Noordwijk, the Netherlands, 26-28 Mai 2003.