Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

MSE | MASTER OF SCIENCE IN ENGINEERING

Projet d'approfondissement
Master HES-SO

Qualité de réception IPTV

Master Research Unit :
Technologies de l'Information et de la Communication (TIC) / hepia

Non soumis à un contrat de confidentialité

Candidat :
ROSSIER Jérémie

Année académique :
2010 - 2011

Professeur responsable :
Andrés Revuelta

En collaboration avec :
CISCO Thomas Kernen

Résumé

L'amélioration continuelle des réseaux de transmission ainsi que la numérisation du contenu multimédia ont permis l'émergence de nouveaux services. L'augmentation des débits, notamment à l'aide des connexions xDSL et prochainement de la fibre optique, a permis l'IPTV.

Cependant, le réseau IP ne garantissant pas de protection contre la perte et corruption de paquets, et que la transmission d'IPTV soit très sensible à ceux-ci, les milieux professionnels concernés ont étudié la possibilité d'ajouter un système de correction d'erreurs de transmission au niveau applicatif. Pour mieux illustrer la sensibilité d'IPTV à la perte de paquets, ci-dessous une figure en illustre les effets :

0% de paquets perdus

0.5% de paquets perdus

5% de paquets perdus

Ce projet a pour but d'intégrer le système de correction d'erreurs décrit par la norme SMPTE 2022 au sein d'un lecteur multimédia reconnu. Cette norme est très répandue dans le monde professionnel.

Le choix du lecteur s'est porté sur le logiciel libre "VLC media player", au vu de sa popularité et des possibilités de contribuer à son développement.

Différents tests ont étés effectués pour vérifier le bon fonctionnement de l'implémentation de cette norme.

Remerciement

Je tiens à remercier M. Andrés Revuelta pour m'avoir proposé ce projet d'approfondissement, de son suivi tout au long de son accomplissement et de ces précieux conseils.

Je remercie également M. Thomas Kernen sans qui ce projet n'aurait pas eu lieu. Je le remercie aussi pour son engagement, son intérêt et son professionnalisme.

Je remercie M. David Fischer pour son algorithme SMPTE 2022, de ses précieux conseils et de son soutien moral.

Table des Matières

1. Cahier des charges

Objectif :

La TV sur IP voit son marché croître très rapidement grâce à l'augmentation des débits Internet proposé à l'utilisateur. La correction des erreurs de transmission par réémission étant impossible dans ce modèle de transmission basé sur le streaming, une des méthodes prévues pour l'amélioration de la QoS, est la correction d'erreurs de transmission SMPTE-2022.

Le présent projet consiste en l'implémentation d'un module d'autocorrection compatible avec SMPTE-2022 dans le projet de lecteur VLC, issu de l'Ecole Centrale de Paris, qui est un des lecteurs vidéo (logiciel libre) les plus utilisés du moment.

Travail demandé :

Dans un premier temps, il sera nécessaire d'étudier la norme SMPTE-2022. Puis, par la suite, d'analyser VLC et l'algorithme SMPTE-2022 existant. Cette analyse permettra, dans un troisième temps, de réaliser et d'intégré le module d'autocorrection à VLC. Des test et mesures seront effectués.

2. Introduction

Le réseau Internet est né dans les années 1970 avec le protocole IPv4 dans le but de relier les universités et les centres de recherches. Celui-ci, de par son engouement et ses multiples possibilités, a été étendu au reste du monde vers la fin des années 90.

A la base, le contenu transporté sur celui-ci était du contenu web, tel que des pages HTML contenant du texte, des liens et des images. Le développement et l'amélioration du réseau des ISP[1]s, tel que les techniques de transmission xDSL et fibre optique entre l'abonné et l'ISP, ont permis le transport de contenu requérant une bande passante insuffisante jusqu'alors. De plus, l'évolution de la représentation du contenu vers une forme numérique a permis l'émergence de nouveaux services.

Un de ces nouveaux services est l'IPTV. Cependant, le réseau IP ne garantissant pas de protection contre la perte et la corruption de paquets, que la transmission d'IPTV soit très sensible à ceux-ci, les milieux professionnels concernés ont étudié la possibilité d'ajouter un système de correction d'erreurs de transmission au niveau applicatif, en plus de ceux prévus par la norme MPEG-2 travaillant sur plusieurs niveaux de codage des programmes vidéo.

Plusieurs normes de ce système de correction d'erreurs ont été rédigées. Ce projet d'approfondissement a pour but d'étudier sa mise en application pratique à l'aide d'un des lecteurs vidéo les plus utilisés du marché : VLC.

Ce lecteur open source était à l'origine développé par les étudiants de l'Ecole centrale de Paris. Il est diffusé sous licence GNU GPL. Un de ces avantages est sa distribution sur une multitude de plateformes (Windows, GNU/Linux, BSD, Mac OS X, etc.). De par ses atouts et sa gratuité, VLC est devenu aujourd'hui l'un des premiers lecteurs multimédias, avec pas moins de 50 millions d'utilisateurs.

Le choix de la norme du système de correction d'erreurs s'est porté sur la norme SMPTE-2022.

2.1. IPTV

L'IPTV se base sur le principe de streaming, consistant en une lecture temps réel du flux réceptionné. Ce flux est mémorisé temporairement dans un tampon circulaire, le temps du traitement et la lecture de celui-ci. Lorsque les données sont traitées et affichées, de nouvelles les remplacent, ce qui permet l'utilisation d'un tampon d'une taille fixe. Le flux est adapté à la bande passante à disposition afin de garantir une bonne réception de celui-ci. La transmission se fait à l'aide du protocole RTP[2] basé sur UDP et les adresses multicast.

IPTV est un système par lequel sont délivrés les services de la télévision à l'aide du réseau Internet, comme le font les systèmes traditionnels : terrestre, satellitaires et par câble.

[1] Fournisseur d'accès à Internet (Internet Service Provider)
[2] Real-Time Protocol

La définition officielle approuvée par le groupe de travail de l'UIT[3] (ITU-T FG IPTV) est la suivante: "IPTV is defined as multimedia services such as television/video/audio/text/graphics/data delivered over IP based networks managed to provide the required level of QoS/QoE, security, interactivity and reliability."[4]

Le marché de l'IPTV a tendance à se développer très rapidement. Cisco et différents groupes d'experts prévoient plus de 80 millions d'abonnés en 2013, ce qui signifie un marché de l'ordre de 30-40 milliards de dollars en 2013.[5] Il est donc judicieux de s'intéresser à l'IPTV.

Les services d'IPTV peuvent être classifiés en 3 catégories :

- La télévision en direct (live television)
- La Vidéo à la demande (VoD)
- Les séances de rattrapage (Catch-up TV, time-shifted programming)

La télévision en direct, sujet de ce projet d'approfondissement, est basée sur le protocole IP utilisant des adresses multicast, permettant l'émission à plusieurs clients d'un seul flux.

La VoD et les séances de rattrapage sont basées sur le protocole IP utilisant seulement des adresses unicast, car chaque client ne demande pas en même temps le même flux.

2.2. Problèmes induits par l'IPTV

Comme tous les flux transitent par des réseaux IP, tous les problèmes liés à IP se répercutent sur la transmission d'IPTV. En effet, IP est basé sur le principe de "best effort". Pour qu'il n'y ait pas de discontinuité entre les réseaux, les protocoles de routages adaptent les routes entre eux.

Des problèmes liés à IP découlent ces quatre phénomènes perturbant la transmission de paquets :

- Perte de paquets (packet dropping)
- Gigue (jitter) : fluctuation de l'inter-arrivée des paquets
- Latence (latency) : temps de trajet des paquets
- Déséquencement (out-of-order delivery) : Modification de l'ordre d'arrivée des paquets

Le deuxième et quatrième problèmes peuvent être grandement diminués, voir supprimés lorsque l'on augmente la taille des buffers. Cependant, cette augmentation engendre des retards.

La perte de paquets est par contre la plus importante des problématiques. En effet, une perte de 0.5% des paquets engendre déjà un flux d'une qualité inacceptable (plusieurs macros blocs corrompus). Avec une perte de 5%, le flux est illisible (plus de 50% de l'image corrompu). L'IPTV ne tolère qu'une infime perte de paquets. Ci-dessous, un exemple illustrant cette diminution flagrante de qualité.

[3] Union Européenne des Télécommunications
[4] http://www.itu.int/ITU-T/newslog/IPTV+Standardization+On+Track+Say+Industry+Experts.aspx
[5] http://www.international-television.org/tv_market_data/global-iptv-forecast-2009-2013.html

0 % de paquets perdus

0.5 % de paquets perdus

5 % de paquets perdus

Figure 1 : exemple de la qualité en fonction de la perte de paquets

Il est possible d'établir de la QoS[6], mais celle-ci ne protège pas de la perte de paquets, elle permet tout au plus de diminuer sa probabilité. Il est impensable de retransmettre les paquets perdus, cela engendrerait de trop grands délais.

Il est donc nécessaire d'ajouter une redondance. La norme SMPTE 2022 permet l'ajout de celle-ci, et donc, de récupérer les paquets perdus.

2.3. Une solution : SMPTE

La "Society of Motion Picture and Television Engineers", fondée en 1916 et basée aux Etats-Unis, est une association professionnelle internationale composée d'ingénieurs travaillant dans l'industrie de la vidéo. SMPTE a publiée plus de 400 standards de recommandations dans le domaine de la télévision, vidéo, cinéma et la vidéo médicale.

Concernant l'IPTV, ils ont publiés une norme permettant la correction d'erreur de transmission. Cette norme, expliquée au chapitre 3.1, est composée de trois documents : SMPTE ST 2022-1-2007, 2022-2-2007 et 2022-3-2010.

[6] Qualité de Service (Quality of Service)

2.4. Contexte du projet, planification

En 2009, David Fischer, sur une proposition de Thomas Kernen (ingénieur chez Cisco), a développé en C l'algorithme décrit dans la norme SMPTE 2022. Cependant, lui manquant de temps, il n'a pu l'intégré dans le lecteur multimédia VLC.

Ce projet d'approfondissement a pour but d'intégrer la norme SMPTE 2022 au lecteur VLC et d'en valider le fonctionnement.

Les dates importantes de ce travail d'approfondissement sont les suivantes :

- Début le 22 février 2011
- Rendu du mémoire le 30 mai 2011
- Défense le 7 juin 2011

Huit heures de travail hebdomadaires sont prévue à sa réalisation.

Une des premières étapes d'un projet est sa planification. La figure à la page suivante est la représentation du planning initial, établi le 1 mars. La couleur jaune représente la partie théorique, la couleur orange le développement, le rouge les test et mesures, et le rouge foncé les échéances. La rédaction du mémoire a été effectuée tout au long du projet.

La figure à la page d'après est la représentation du planning final. De petit changement ont eu lieu, ils sont représentés par une nuance sur les couleurs. J'ai pu prendre de l'avance dans la partie théorique, mais celle-ci a été compensée par des problèmes rencontrés durant le développement. Par contre, le temps consacré aux tests et mesures n'a pas été réduit. Cette réduction est un phénomène fréquemment dans le déroulement de projets à cause d'imprévus survenus au cours du développement.

A l'aide de mes connaissances préalables, de celles acquises au cours du Master et dans certains cas de l'aide de personnes compétentes dans le domaine, j'ai pu minimiser la perte de temps lors des problèmes rencontrés.

2.4.1. Planning initial

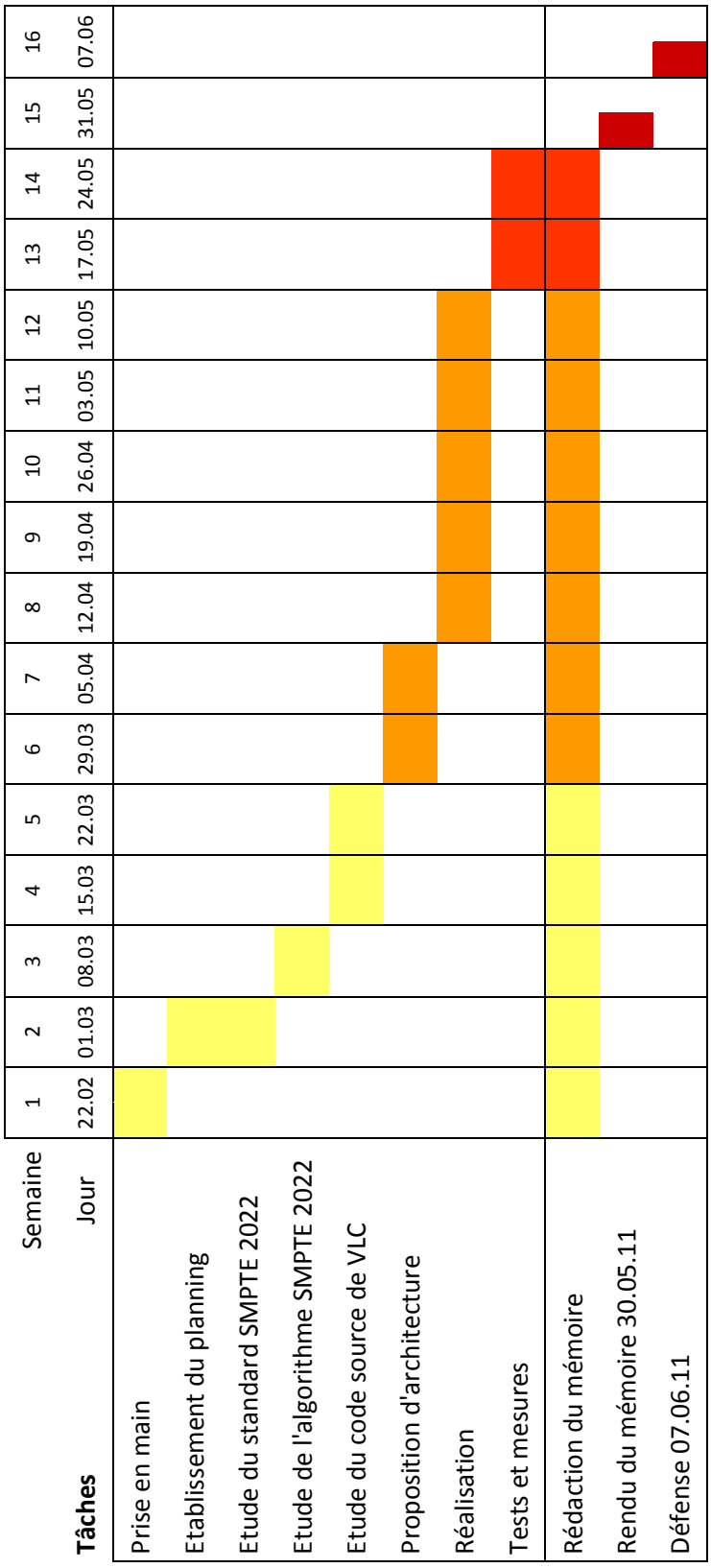


Figure 2 : planning initial

2.4.2. Planning final

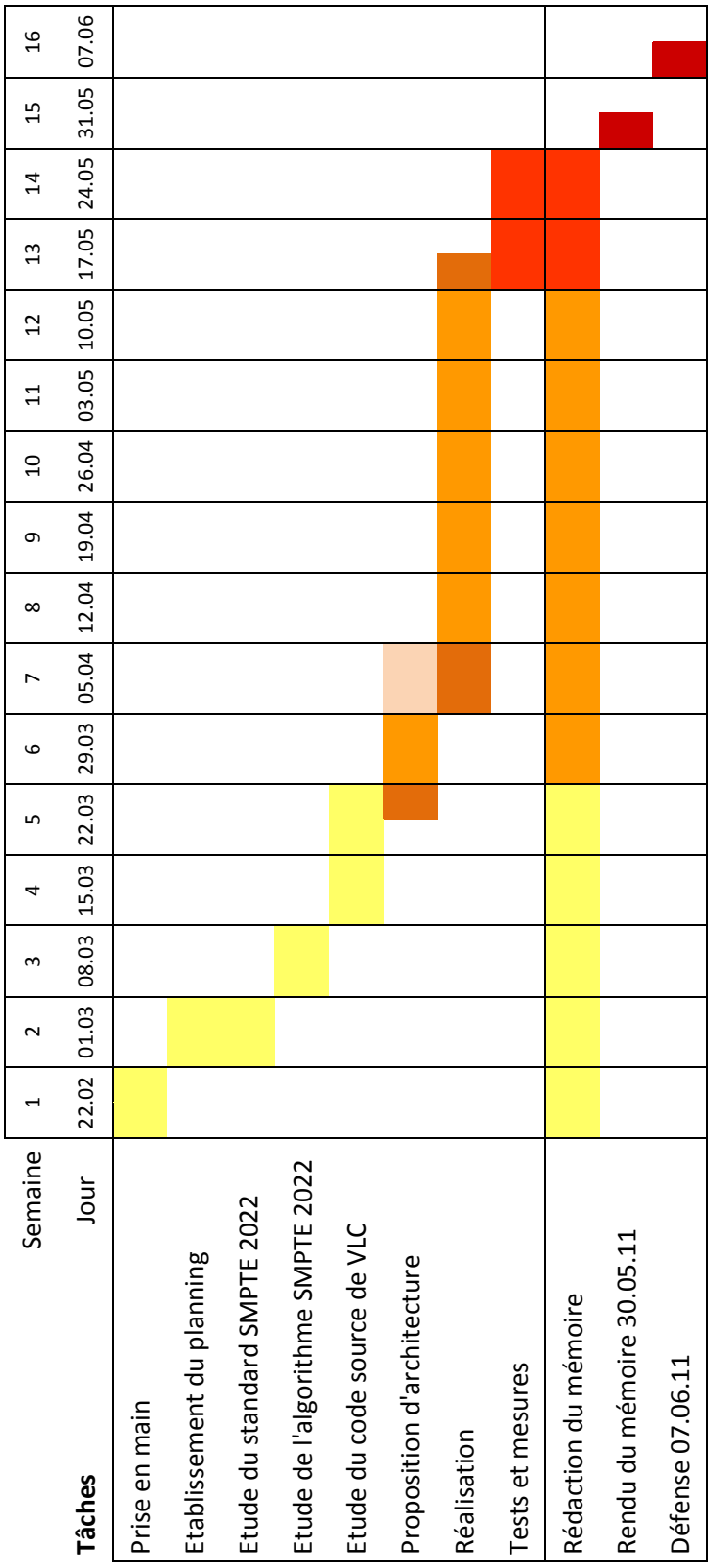


Figure 3 : planning final

3. Les acteurs

3.1. La norme SMPTE 2022

Cette norme est décrite dans actuellement trois documents publiés sous les intitulés SMPTE ST 2022-1-2007, 2022-2-2007 et 2022-3-2010, disponibles auprès de SMPTE.

La 1ère version traite de la correction d'erreurs pour le transport de flux vidéo/audio temps réel sur des réseaux IP. La 2e traite du transport unidirectionnel à débit constant de flux MPEG-2 sur des réseaux IP. Quant à la 3e, elle traite du transport unidirectionnel à débit variable.

Les paquets FEC utilisent les couches RTP/UDP/IP. Ces paquets sont liés à des paquets média choisi périodiquement. Pour une question de compatibilité, le serveur émet les paquets FEC en parallèle du flux média, le client décide s'il veut les utiliser (ou n'en a pas la possibilité). Pour une compatibilité avec des systèmes ne gérant qu'un seul flux FEC, les deux flux FEC sont émis séparément.

Deux schémas de corrections existent : FEC 1D et FEC 2D. Le FEC 2D a l'avantage d'avoir une plus grande capacité de correction d'erreur par rapport au FEC 1D. Cependant, lors de son implémentation, il est tout à fait possible d'utiliser le module 2D avec seulement des paquets FEC 1D. De ce fait et de sa plus grande capacité de correction, ce travail ne traitera que de la version 2D. Considérons le schéma de correction suivant :

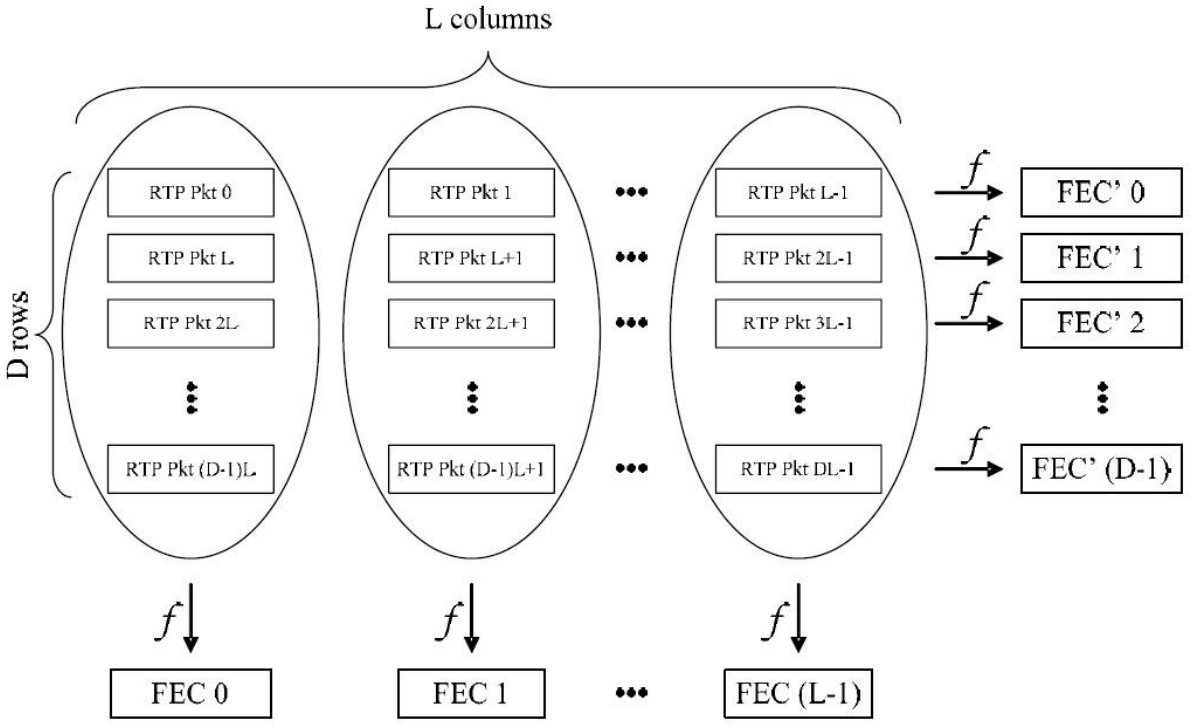


Figure 4 : schéma de correction FEC 2D

Ce schéma de correction est schématisé pour L*D paquets média. Par exemple, le paquet FEC 0 permet de corriger les paquets media 0 à (D-1)*L et le paquet FEC' 2 permet de corriger les paquets media 2L à 3L-1. Un paquet FEC ne peut corriger qu'une seule erreur sur sa ligne ou colonne.

Cependant, il est possible de corriger plusieurs erreurs sur une ligne grâce aux colonnes et inversement avec une colonne. Il est de même possible qu'une correction en cascade se produise : imaginons 2 paquets manquants dans une ligne et 1 autre paquet dans la colonne du premier paquet de la ligne. A l'aide des paquets FEC de la colonne et de la ligne concernées, il est impossible de récupérer les paquets manquants. Cependant, avec le paquet FEC de la 2^{e} ligne concernée, nous pouvons récupérer le paquet manquant de la ligne (étape 1), puis à l'aide du paquet FEC colonne récupérer le 2^{e} paquet manquant de la colonne (étape 2) et ainsi de suite (étape 3). Ci-dessous, un schéma pour mieux illustrer mes propos :

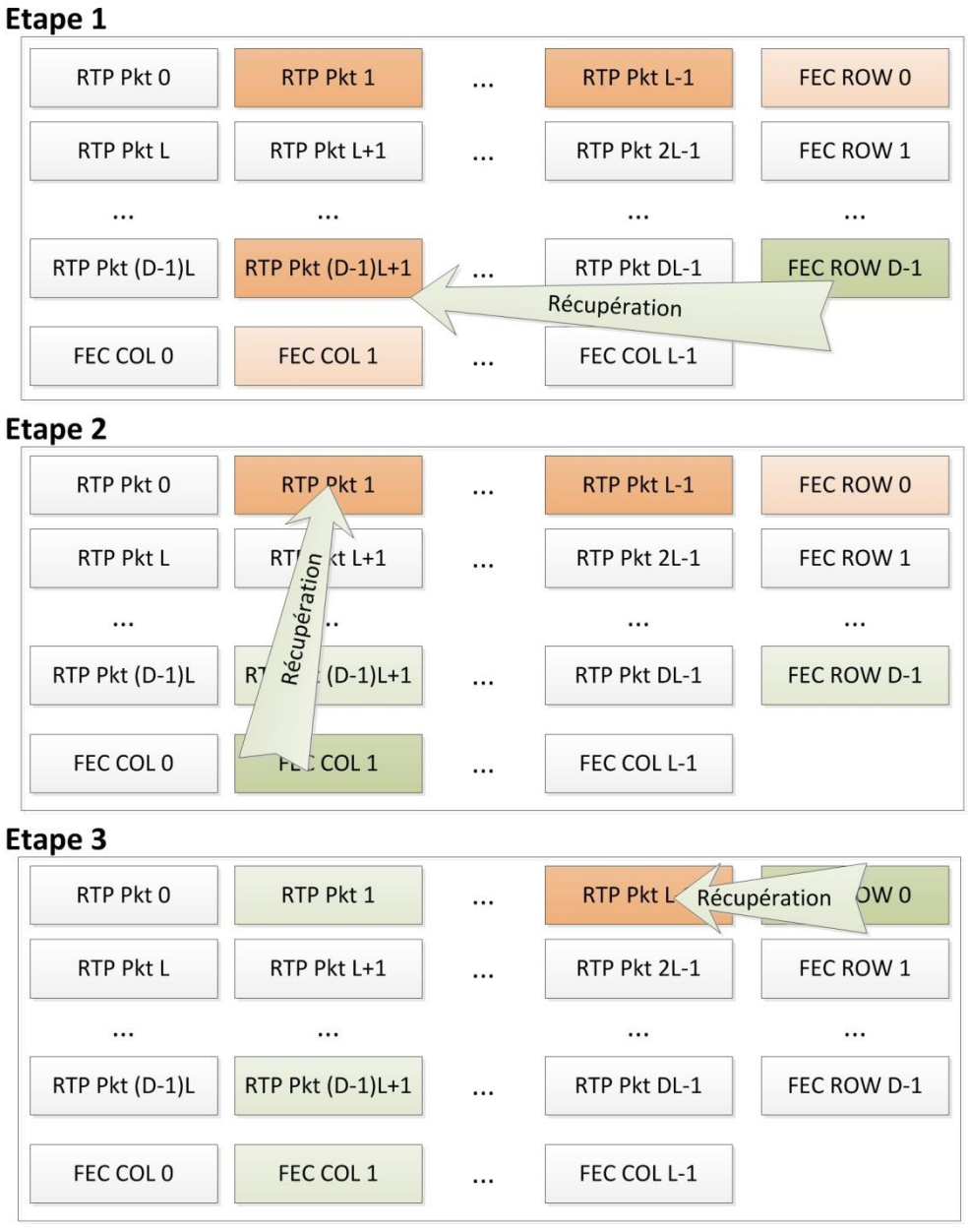


Figure 5 : schéma de correction FEC 2D exemple

La fonction utilisée pour le calcul des paquets FECs est simplement la fonction XOR.

Le tableau ci-dessous, venant de la version 2-2007, illustre bien l'overhead engendré par le FEC 2D, la latence et le nombre de paquets IP possible de récupérer.

	Overhead	Latency			Recovery	Buffer size
		3Mbps	30 Mbps	100 Mbps		
XOR (5,10)	10%	175.5 ms	17.5 ms	5.3 ms	5 IP packets	66400 Bytes
XOR (10,10)	10%	350.9 ms	35.1 ms	10.5 ms	10 IP packets	132800 Bytes
XOR (20,5)	20%	350.9 ms	35.1 ms	10.5 ms	20 IP packets	132800 Bytes
XOR (8,8)	12.5%	224.6 ms	22.5 ms	6.7 ms	8 IP packets	84992 Bytes
XOR (10,5)	20%	175.5 ms	17.5 ms	5.3 ms	10 IP packets	66400 Bytes
XOR (8,5)	20%	140.4 ms	14.0 ms	4.2 ms	8 IP packets	53120 Bytes
XOR (5,5)	20%	87.7 ms	8.8 ms	2.7 ms	5 IP packets	33200 Bytes
XOR (4,6)	16.7%	84.2 ms	8.4 ms	2.5 ms	4 IP packets	31872 Bytes
XOR (6,4)	25%	84.2 ms	8.4 ms	2.5 ms	6 IP packets	31872 Bytes

Figure 6 : Tableau de l'overhead et latence

Pour que la latence ne soit trop grande, des limites ont été fixées à la taille des colonnes et lignes :

$$L*D \leq 256$$
$$1 \leq L \leq 50$$
$$4 \leq D \leq 50$$

Figure 7 : Limite des paramètres du FEC 2D

Normalement, les flux FECs doivent être envoyés sur la même adresse de destination (unicast ou multicast) que le flux média. Le port de destination du flux média est le port N (où N est un entier pair selon la RFC 3550), le port du flux FEC colonne doit être le port N+2 et le port du flux FEC ligne le port N+4. Ce n'est pas un incrément de 1, car le port N+1 est utilisé par le flux RTCP. Cependant, il n'est pas obligatoire de respecter cette attribution de ports et d'adresses.

La norme spécifie aussi deux schémas de construction de la matrice. Le premier est, comme illustré dans les exemples ci-dessus, carré. Le deuxième est parallélogramme. Le choix entre les deux se fera selon les pertes engendrées par la transmission.

Pour de plus amples informations à propos de cette norme, je vous conseille de la parcourir, elle est bien expliquée et concise.

Format de l'entête FEC

L'entête d'un paquet FEC a été défini en premier lieu dans la RFC2733. Cependant, il a été étendu pour correspondre au schéma de correction.

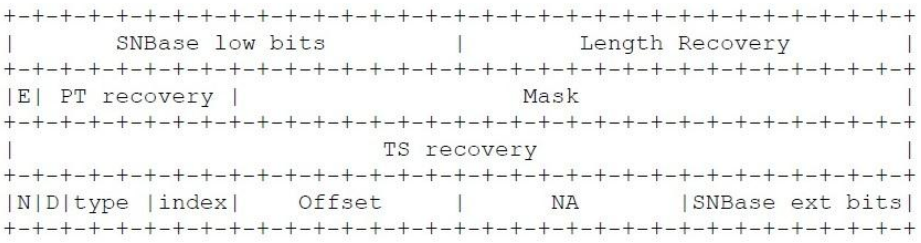


Figure 8 : Entête FEC

Les champs principaux sont les suivants :

"SNBase" correspond au plus petit numéro de paquet media protégé. "Length Recovery" est la longueur (en nombre d'octets) des paquets media protégés. "PT recovery" et "TS recovery" correspondent aux résultats du XOR de tous les "Payload Type" et "TimeStamp" des paquets media. Le bit "D" permet de différencier les paquets FEC ligne ou colonne. "Offset" est l'incrément entre chaque paquet media protégé. Quant à "NA", c'est le nombre de paquets protégés par le paquet FEC.

Dans le cas d'utilisation de la norme lors de transmission à débit variable, le bit E est à "1", correspondant à l'utilisation d'une extension de l'entête. Cette extension est expliquée dans la version 3-2010. Cependant, dans un premier temps, je n'en tiendrais pas compte.

L'entête RTP des paquets media n'étant pas compris dans le payload des paquets FEC, il a été nécessaire de protéger ses différents champs importants. Le reste de l'entête RTP est créé à l'aide des entêtes des autres paquets RTP.

3.2. VLC media player

Ce lecteur open source était à l'origine développé par les étudiants de l'Ecole centrale de Paris. Il est diffusé sous licence GNU GPL. Un de ces avantages est sa distribution sur une multitude de plateformes (Windows, GNU/Linux, BSD, Mac OS X, etc.). De par ses atouts et sa gratuité, VLC est devenu aujourd'hui l'un des premiers lecteurs multimédias, avec pas moins de 50 millions d'utilisateurs.

Les principaux langages de programmation utilisés sont le C et le C++. Son fonctionnement est modulaire, ce qui permet d'exécuter seulement les modules nécessaires à un temps voulu. Lorsque l'utilisation de l'un d'eux n'est plus nécessaire (ou plusieurs d'entre eux), par exemple lors de l'arrêt d'une lecture d'un flux vidéo, celui-ci est stoppé. Il peut à tout moment re-être exécuté. Les concepteurs de VLC ont mis en place une classification des modules, chacun d'eux devant être associé qu'à une seule de ces catégories :

- CAT_INTERFACE : Interface utilisateur
- CAT_AUDIO : module relatif à l'audio
- CAT_VIDEO : module relatif à la vidéo
- CAT_INPUT : module de démultiplexage
- CAT_SOUT : module de multiplexage
- CAT_ADVANCED : gestion avancée du CPU, réseau, etc.
- CAT_PLAYLIST : gestion de la liste de lecture

Chacune de ces catégories contient des sous-catégories. Je ne vais pas toutes les citées, elles sont disponibles sur le wiki de VLC[7]. La partie concernée par ce projet du module RTP fait partie de la sous-catégorie SUBCAT_INPUT_DEMUX de la catégorie CAT_INPUT. En effet, le flux vidéo est en entrée, il doit être démultiplexé pour qu'ensuite ses différentes parties (paquets media, audio, vidéo, etc.) soient transmises aux modules de traitements concernés.

Il faut faire attention au fait qu'il existe deux modules RTP dans le code source de VLC. En effet, il y a le module de démultiplexage et celui de multiplexage. Celui concerné par ce projet et celui de démultiplexage. Ses fichiers sont situés dans le dossier /modules/access/rtp/.

Un autre point, très important, est le fonctionnement multi-threads de VLC. En effet, chaque module étant indépendant des autres, il est plus avantageux d'utiliser une architecture multi-thread. Chaque module peut être constitué d'un ou plusieurs threads.

Vous trouverez dans les annexes un guide de téléchargement et compilation du code source de VLC, un guide pour utiliser le debugger et un guide pour rejouer une capture d'un flux media et ses flux fec associés.

[7] http://wiki.videolan.org/Documentation:Hacker%27s_Guide/Module_Writers_Guide

4. Intégration de la norme SMPTE 2022 au lecteur VLC

4.1. Algorithme SMPTE 2022

L'algorithme a été développé dans un premier temps par D. Fischer. L'analyse de celui-ci est nécessaire pour comprendre son fonctionnement et d'autre part cerner les endroits à modifier et adapter lors de son intégration à VLC.

L'algorithme mis en place consultant fréquemment les paquets media contenu dans le bufferMedia, celui-ci est basé sur un stockage des clés dans un arbre rouge-noir afin de diminuer la complexité d'une liste chaînée en O(n) en un arbre bicolore en O(lb n). Dans le rapport relatif, des mesures ont été effectuées montrant le gain apporté par cet arbre bicolore.

4.1.1. La structure des données

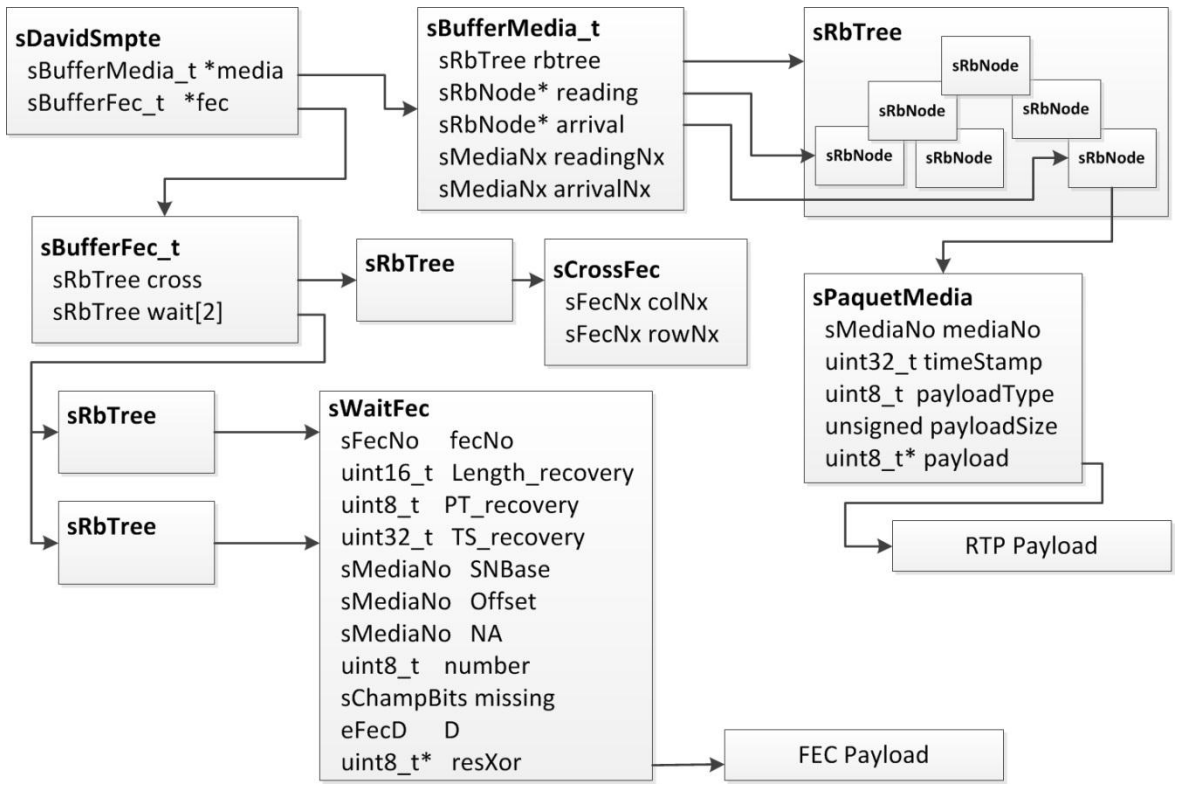


Figure 9 : Statechart de la structure des données de l'algorithme SMPTE 2022

La structure est constituée de deux buffers, l'un comprenant les paquets media et l'autre des paquets fec. Ils sont les deux basés sur des arbres bicolores. Les nœuds de ceux-ci correspondent aux paquets media et fec.

4.1.2. Les méthodes principales

Les méthodes principales de l'algorithme sont les suivantes :

- Méthode de gestion d'arrivée d'un paquet media
- Méthode de gestion d'arrivée d'un paquet fec
- Méthode de nettoyage lors de la lecture d'un paquet media
- Méthode de gestion de la perte d'un paquet media
- Méthode de récupération de paquet media

Pour mieux comprendre le fonctionnement de la gestion d'arrivée des paquets media et fec, ci-dessous les statecharts et explications relatifs à ces deux méthodes.

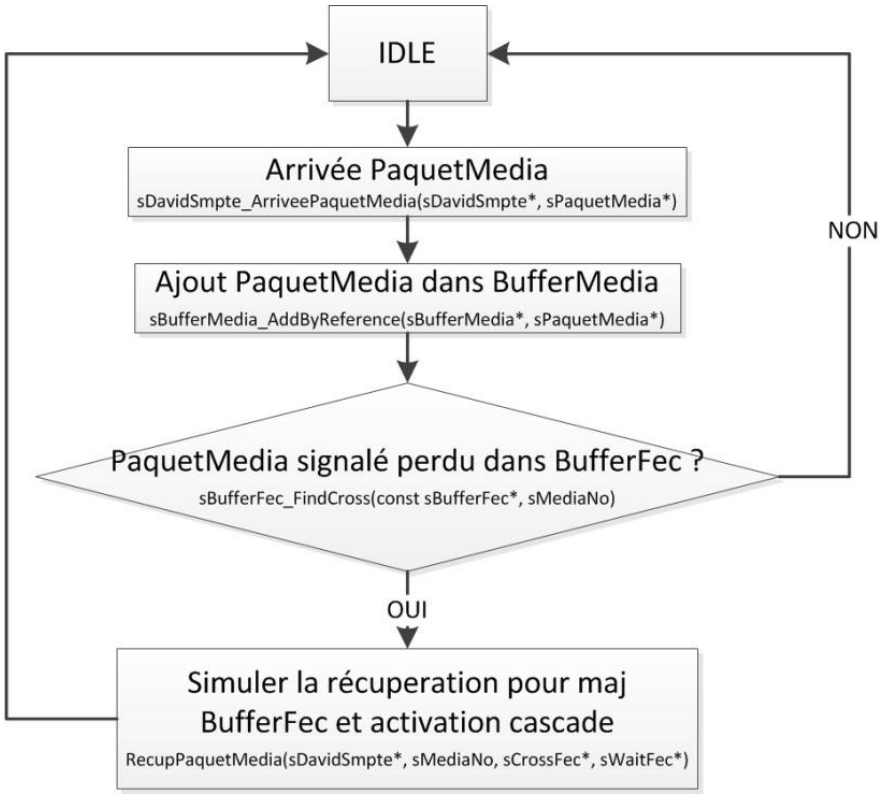


Figure 10 : Statechart de la gestion d'arrivée d'un paquet Media

Lors de l'arrivée d'un paquet media, celui-ci est annoncé à l'algorithme à l'aide de la méthode de gestion d'arrivée d'un paquet media. Cette méthode ajoute ce paquet dans la structure des données. Elle vérifie si ce paquet permet la récupération d'un autre paquet manquant concerné par les mêmes paquets fec (paquets fec colonne et ligne du paquet media annoncé). Dans le cas échéant, on vérifie si ce paquet récupéré permet à son tour la récupération de paquets manquants, cet effet est appelé cascade.

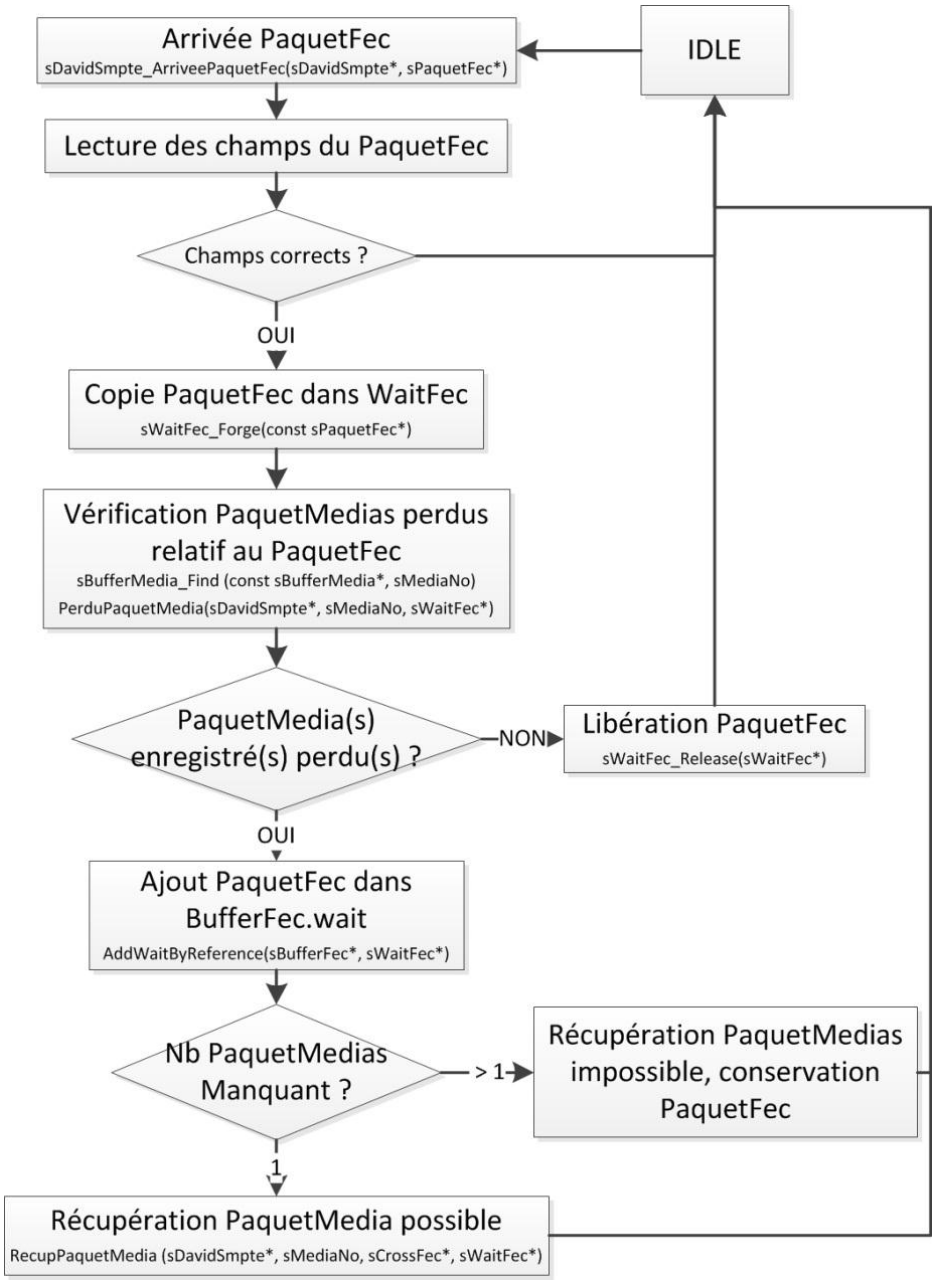


Figure 11 : Statechart de la gestion d'arrivée d'un paquet FEC

Lors de l'arrivée d'un paquet fec, celui-ci est annoncé à l'algorithme à l'aide de la méthode de gestion d'arrivée d'un paquet fec. Dans un premier temps, on vérifie la conformité du paquet reçu, puis l'ajoute à la structure des données. On vérifie si un paquet media protégé par ce paquet fec est manquant et, dans le cas où il n'en manquerait qu'un seul, on le récupère. Si sa récupération permet l'effet cascade, on l'exécute. Dans le cas où plusieurs paquets media seraient manquants, on conserve le paquet fec pour une récupération future possible. Et dans le cas où aucun paquet media n'est manquant, il n'est pas nécessaire de conserver le paquet fec.

4.2. Analyse du module RTP de VLC

Ce module étant de grande taille, mon analyse s'est fait en trois temps. En premier, j'ai analysé l'instanciation du module, la création du thread et des sockets. Ci-dessous, le statechart relatif.

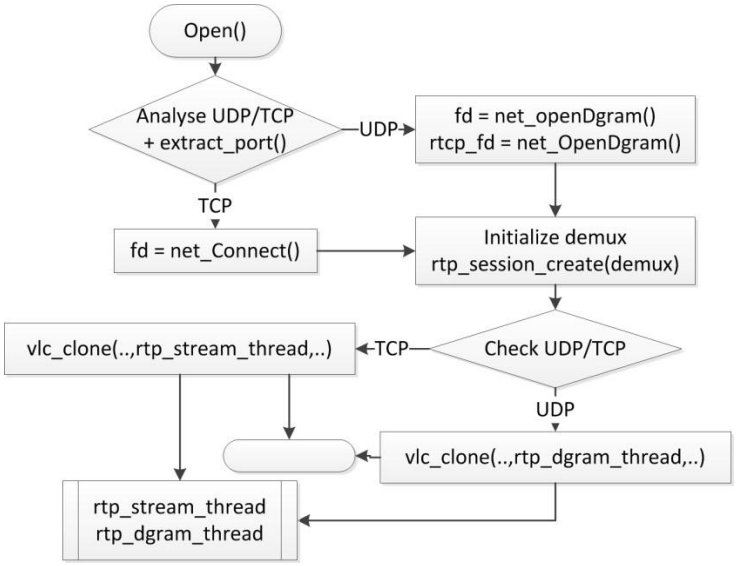


Figure 12 : Statechart du module RTP : instanciation

En deuxième, j'ai suivi le chemin qu'empruntaient les paquets RTP arrivants. Les paquets RTP arrivent dans le socket UDP, sont transmis à la fonction de traitement de paquets RTP, celle-ci faisant appel à la fonction queue(). Les entêtes sont évidements vérifié. Les paquets sont mis dans la file correspondant à la source du flux RTP en respectant un ordre chronologique des numéros de séquence. Si la source n'existe pas encore, elle est créée. Ci-dessous, le statechart relatif.

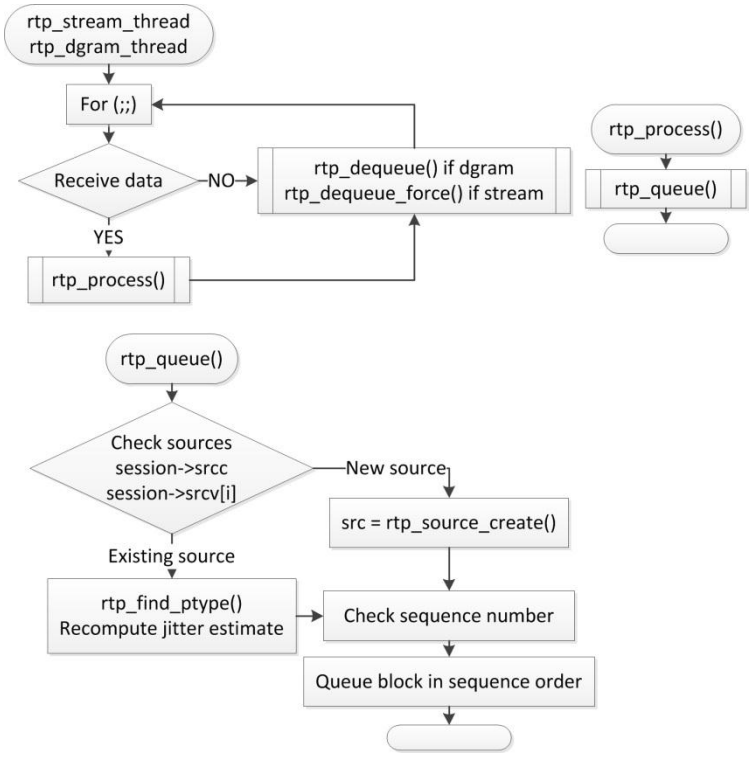


Figure 13 : Statechart du module RTP : gestion thread et fonction queue()

Et en troisième, dans la continuité du chemin parcouru par les paquets RTP, l'appel de la fonction dequeue(). Pour chaque source, on vérifie la présence de paquets RTP en attente dans leurs files respectives. S'il n'y a pas de discontinuité entre les numéros de séquences du paquet RTP et du précédent transmis au prochain module, celui-ci est à son tour transmis au prochain module. Dans le cas contraire, on attend un certain lapse de temps (~25 ms) permettant l'arrivée du(des) paquet(s) manquant(s) avant de transmettre les paquets RTP en attente dans la file au prochain module. Ci-dessous le statechart relatif.

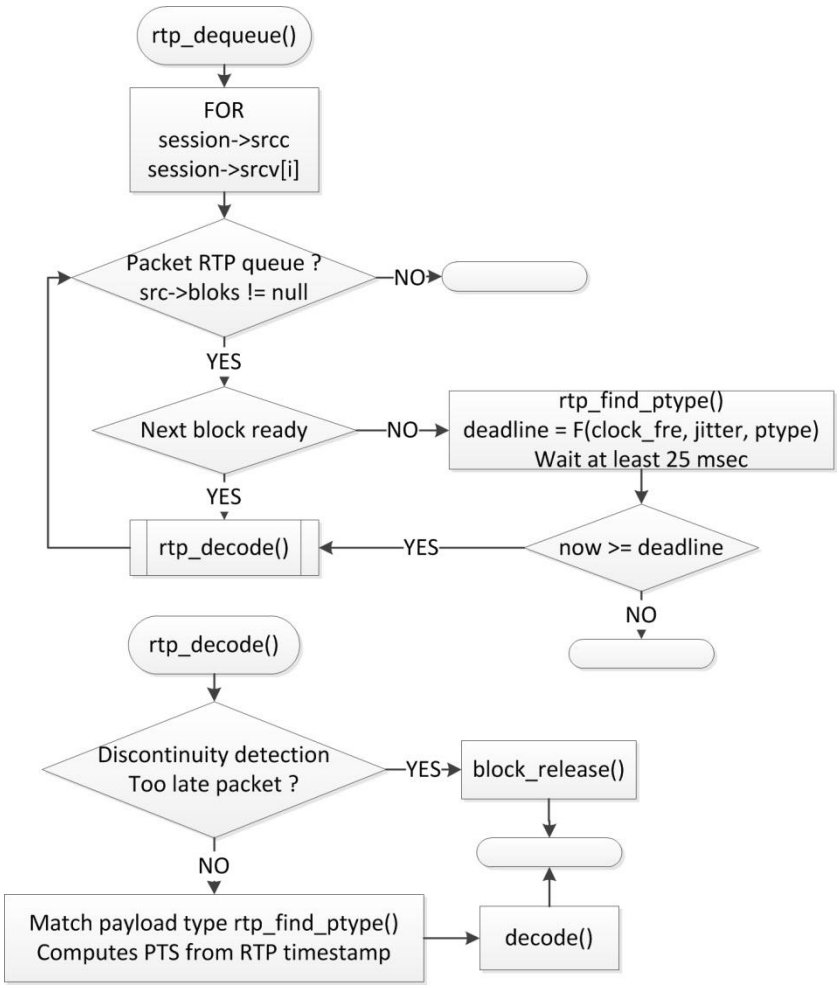


Figure 14 : Statechart du module RTP : fonction dequeue() et decode()

Cependant, l'algorithme fonctionnant sur une matrice de L*D paquets RTP, le mode de fonctionnement de la file d'attente devra être modifié. En effet, il faut que nous puissions reconstituer un(des) paquet(s) RTP perdu(s) de la matrice et que l'on puisse accéder aux autres pour pouvoir les utiliser pour reconstituer. De ce fait, nous garderons dans le buffer une fenêtre équivalente à la taille de la matrice.

A l'aide du fractionnement du module RTP en différents statecharts, j'ai pu cerner les différents endroits où je devrais intégrer l'algorithme SMPTE 2022.

4.3. Liens entre la structure des paquets RTP de l'algorithme SMPTE 2022 et de VLC

Différentes approches pour le lien entre la structure des paquets RTP de l'algorithme et de ceux du lecteur VLC sont possibles.

Afin de minimiser le changement de comportement du module RTP, il n'est pas envisageable d'insérer un buffer avant ou après le module RTP pour l'algorithme. Cet ajout ajouterait en plus un retard supplémentaire. En cas de non-utilisation de l'algorithme, les paquets RTP devront quand même passer par ce buffer.

De plus, pour optimiser l'espace mémoire, il est plus judicieux d'accéder aux mêmes espaces mémoires à l'aide de pointeurs.

De ces différents faits, ci-dessous ma proposition d'utilisation des structures et de la mémoire utilisée :

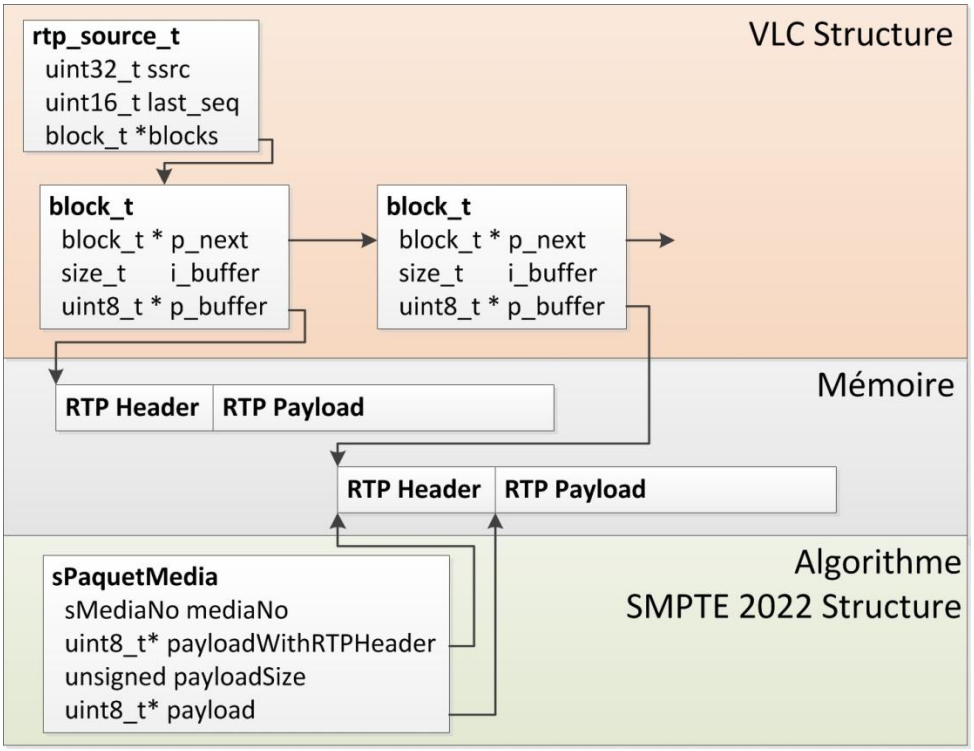


Figure 15 : Structure des paquets RTP de l'algorithme SMPTE 2022 et de VLC

J'ai rajouté un pointeur "payloadWithRTPHeader" à la structure de l'algorithme, car, lors de la récupération d'un paquet RTP, les champs de base de l'entête RTP sont identiques entre chaque paquet RTP et ne sont pas protégés par les paquets FEC. Il suffit donc de les copier d'un des paquets RTP protégé par le même paquet FEC.

4.4. Code de récupération d'un paquet media

Ne voulant pas insérer de code dans ce rapport, il est important, à mes yeux, que le pseudo-code permettant la récupération des différents champs d'un paquet media soit écrit dans celui-ci.

Evidemment, pour pouvoir récupérer un paquet media, il ne faut pas qu'il n'y ait plus d'un paquet media manquant relatif à un paquet fec. Pour récupérer les champs d'un paquet media manquant, il faut parcourir les autres paquets media relatifs au paquet fec et faire l'opération XOR entre les différents champs concernés. Pour le payload, cette opération est effectuée pour chaque octet.

```
PaquetMedia_à_reconstituer = Champs PaquetFec
for PaquetMedias_protégés_par_PaquetFec
                          (mediaNo = SNBase + j*offset, avec j[0;NA[ )
   if PaquetMedia =/= PaquetMedia_à_reconstituer
        TimeStampRecup XOR TimeStamp_PaquetMedia
        PayloadTypeRecup XOR PayloadType_PaquetMedia
        PayloadRecup XOR Payload_PaquetMedia byte per byte
   end if
end for
```

Figure 16 : Code de récupération d'un paquet media

4.5. Statechart de l'intégration de la norme SMPTE 2022 au lecteur VLC

Pour mieux comprendre où j'ai modifié le code du module RTP de VLC, j'ai transcris les différentes modifications apportées sur les statecharts ci-dessous. Les parties ajoutées ou modifiées sont mises en évidence (couleur rouge). Pour une cohérence avec mon analyse du module RTP, j'ai séparé de la même manière les statecharts. Quelques petites modifications n'apportant pas d'éléments à la compréhension globale n'ont pas étés transcris.

Lors de la création du socket UDP du flux Media, je crée les sockets des flux FEC. Et lors de la création du thread de gestion du flux Media, je crée celui des flux FEC.

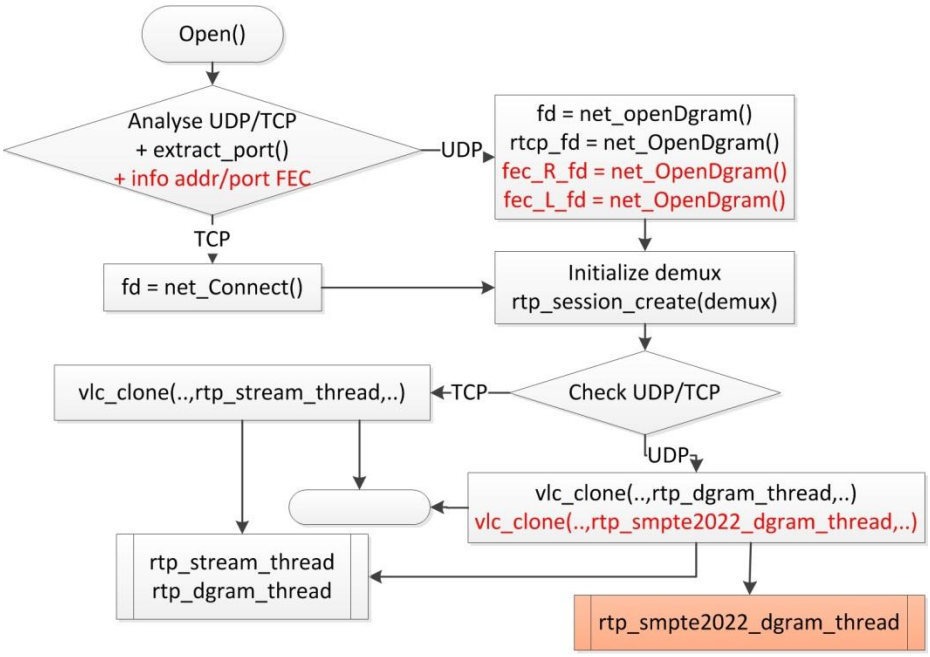


Figure 17 : Statechart des modifications apportées au module RTP : instanciation

Le chemin emprunté par les paquets fec est similaire à celui emprunté par les paquets media, dans leurs threads respectifs.

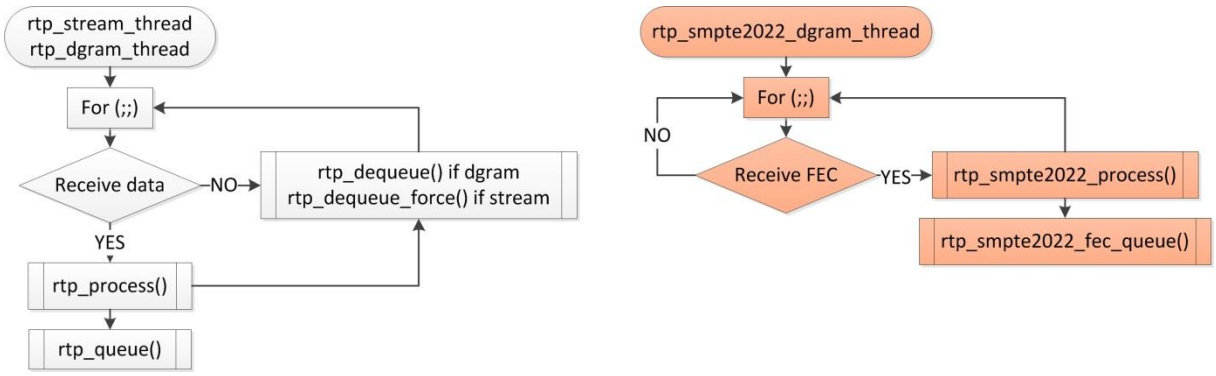


Figure 18 : Statechart des modifications apportées au module RTP : gestion threads

A l'arrivée d'un paquet media, il faut l'annoncer à l'algorithme SMPTE 2022. Cependant, il faut que le paquet media soit inséré dans la file d'attente avant de l'annoncer. Du côté des paquets fec, on les associe à la source émettrice des flux. Il faut attendre un petit délai, car le temps de parcours d'un paquet media est plus important que celui d'un paquet fec. Si l'on n'attend pas ce petit délai, il arrive fréquemment que le dernier paquet media protégé par un paquet fec ne soit pas encore annoncé à l'algorithme, et donc, une récupération de celui-ci a lieu. Après cette petite attente, le paquet fec est annoncé à l'algorithme SMPTE 2022.

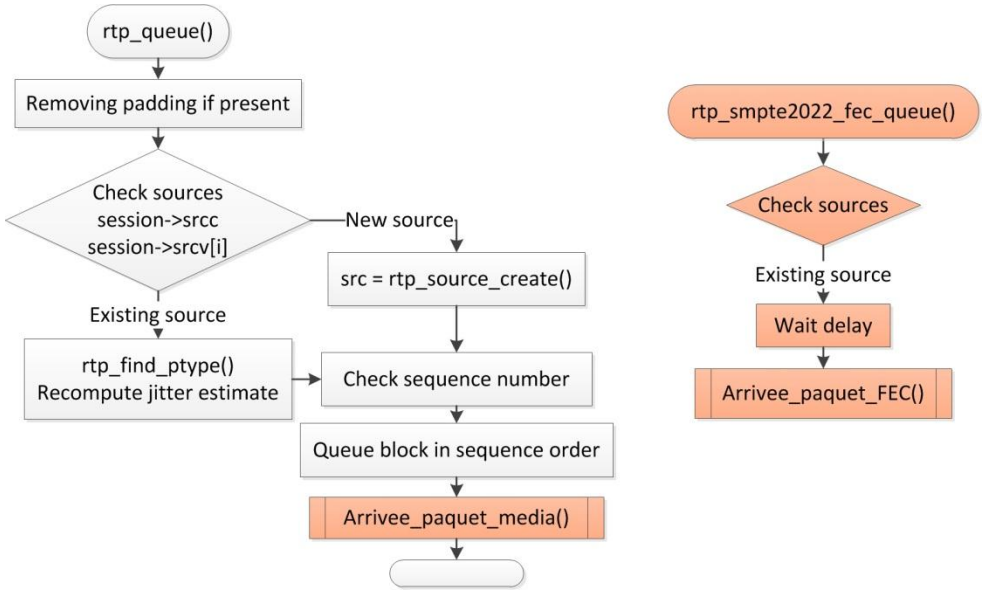


Figure 19 : Statechart des modifications apportées au module RTP : queue() paquets media et fec

Lors de l'enlèvement d'un paquet media de la file d'attente (appel de la méthode dequeue()), le paquet media en question doit être annoncé plus disponible à l'algorithme SMPTE 2022 (appel de la méthode lecture_paquet_media()). De plus, le fonctionnement de la gestion des paquets en attente a dû être modifié pour que l'on garde les paquets media de la matrice dans la file d'attente.

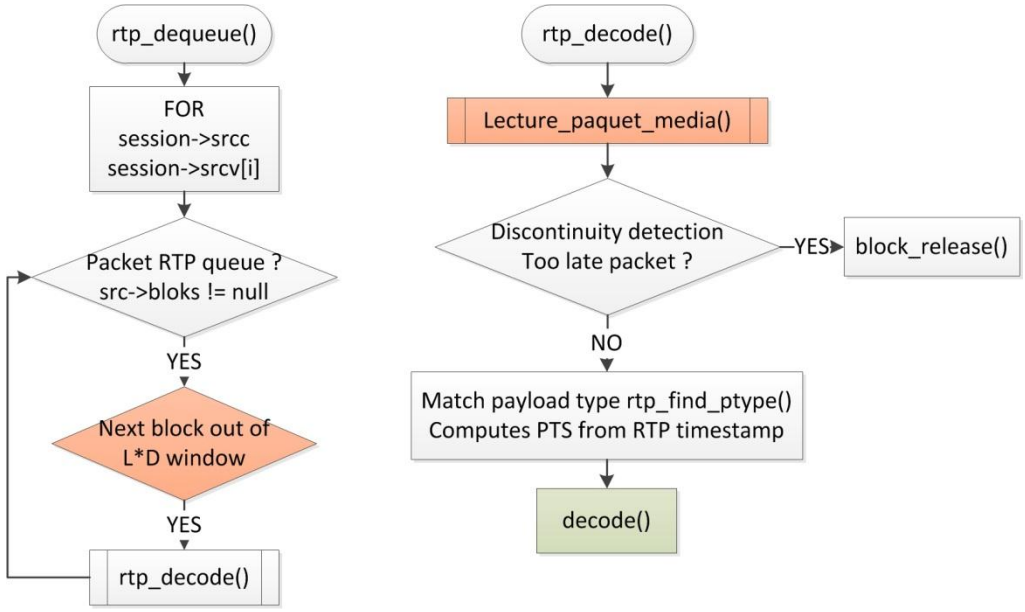


Figure 20 : Statechart des modifications apportées au module RTP : dequeue() et decode()

4.6. Modification des paramètres via l'interface des préférences

J'ai ajouté la possibilité de modifier, par l'interface des préférences, les adresses et ports des flux FEC. En cas de non-modification de ces paramètres, l'adresse du flux RTP est utilisée pour les adresses des flux FEC. Concernant les ports, ils sont incrémentés selon la norme : +2 pour le flux FEC ligne et +4 pour le flux FEC ligne.

Ci-dessous, un exemple de l'interface des préférences avec les paramètres ajoutés.

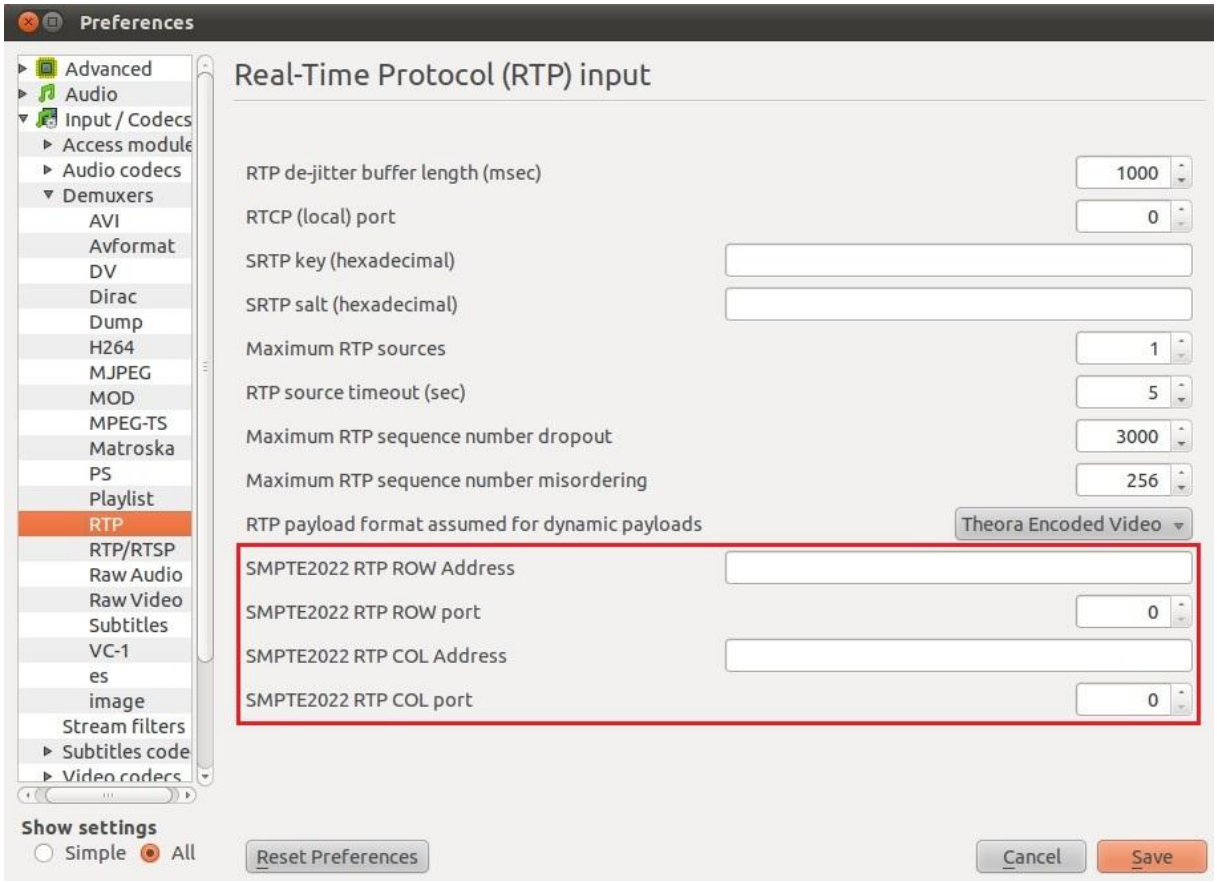


Figure 21 : modification des paramètres via l'interface des préférences

5. Déroulement du projet

22.02.11 - 01.03.11

Etude de la norme SMPTE 2022 (2022-1-2007, 2022-2-2007 et 2022-3:2010). Compilation et exécution du code source de VLC. Création de scripts pour automatiser différentes tâches lors de la compilation.

Etude de l'algorithme SMPTE 2022 (D. Fischer) et d'une analyse de VLC (L. Ricciuti).

08.03.11

Analyse de l'algorithme SMPTE 2022. Création d'un Statechart pour une meilleure compréhension.

15.03.11

Etude du code source de VLC. Les fichiers à modifier sont situés dans le dossier /modules/access/rtp/ (rtp.h, rtp.c, input.c, session.c). Depuis le travail de L. Ricciuti, le code source a subit de grands changements.

22.03.11

Compréhension du buffer interne des paquets RTP (méthode queue() et dequeue()). Ces méthodes pourront être utilisées lors de l'implémentation de SMPTE2022. Le debugger a permis de corréler la bonne compréhension.

05.04.11

Ajout des sockets UDP des flux FEC (ligne et colonne). Création d'un thread gérant les 2 sockets des flux FEC. Essai infructueux d'ajout direct du code de D. Fischer.

12.04.11

Début de l'adaptation du code de D. Fischer par morceau. L'ajout d'une structure de base smpte2022 après différents problèmes de malloc réussi.

19.04.11

Suite de l'adaptation du code de l'algorithme SMPTE 2022. Le lien entre les paquets media de VLC et ceux de l'algorithme est fonctionnel, chaque paquet media est annoncé à celui-ci (lors de l'appel de la méthode queue()). Un petit bug est corrigé dans l'implémentation de l'algorithme concernant le calcul du Timestamp et le numéro de séquence lors de la récupération d'un paquet media.

26.04.11

Chaque paquet media est annoncé plus disponible à l'algorithme SMPTE 2022 lors du passage au prochain buffer (lors de l'appel de la méthode dequeue()).

29.04.11

Les paquets FEC reçus sont convertis dans la structure de l'algorithme SMPTE 2022. Différents essais à l'aide de capture confirment la bonne interprétation des champs de l'entête des paquets FEC.

03.05.11

Mise en place de la fonction de détection de perte de paquets media et calcul des paquets récupérables. Les pertes isolées, consécutives ou en cascades sont gérées. Des essais à l'aide d'une capture fournie est concluant (TVG420_FEC_2D.pcap) : la matrice de FEC est de dimension 10*15. Chaque paquet media volontairement supprimé a été détecté et calculé.

J'ai rencontré un souci de synchronisation entre les threads s'occupant des paquets media et FEC, les paquets media mettant plus de temps à être traités. Les derniers paquets media protégés par les paquets FEC étaient considérés perdus alors qu'ils n'étaient pas encore traités. Pour pallier à ce problème, j'ai retardé de 50 microsecondes (délais maximum entre les traitements constaté) le traitement des paquets FEC. Ce délai n'influence en rien le reste.

10.05.11

Les paquets media récupérés sont adaptés à la structure de VLC. Ils sont correctement insérés au buffer du module RTP. Cependant, lors de récupérations en cascade, seul le premier paquet media est récupéré.

17.05.11

Une erreur de condition lors de la récupération d'un paquet media est la cause du problème rencontré le 10.05.11 (récupération en cascade non-fonctionnelle). Avec les corrections apportées, toute forme de pertes (isolées, consécutives ou en cascades) sont gérées et les paquets media récupérés.

J'ai ajouté la possibilité de modifier, par l'interface des préférences, les adresses et ports des flux FEC. En cas de non-modification, l'adresse du flux media est utilisée et les ports sont incrémentés selon la norme.

Un test avec toute forme de perte a été effectué avec la capture TVG420_FEC_2D.pcap. Celui-ci est disponible au chap. 6.1.

De base, une fenêtre de la taille maximum fixée par la norme (256) était utilisée pour le buffer des paquets media, celle-ci s'adapte en fonction de la taille de la matrice. En cas de non-utilisation des flux FEC, après 500ms celle-ci diminue à la taille de 15 pour garantir le minimum de délais.

24.05.11

J'ai effectué des essais à l'aide des captures TVG420_FEC_2D.pcap, DCM_FEC_2D_ANNEX_A.pcap et DCM_5Mbps_2D_5_10_Non_aligned.pcap. Avec les deux premières captures, j'ai simulé une perte de 1-2% des paquets (enlevé des paquets media à l'aide de *wireshark*). Tous les paquets supprimés ont été récupéré.

J'ai eu des problèmes de lectures des nouvelles captures fournies, ma version de VLC sur Windows Seven réceptionnait le flux RTP, tandis que celle sur Ubuntu (plateforme d'implémentation) ne le réceptionnait pas. Cette différence est due à l'enlèvement automatique de l'encapsulation 802.1q (Vlan Tag) sur Windows. J'ai donc joué la capture, enregistré le flux réceptionné sur Windows pour ensuite le rejouer sans les tags 802.1q.

J'ai constaté, lors de la lecture des captures à l'aide de *tcpreplay*, que le débit en sortie était limité. Après divers essais et différentes configurations (changement de méthode de gestion de temps, mise en mémoire ram des fichiers de capture, etc.) le débit ne dépassait pas les 10Mbps. Cela a pour conséquence, lors d'une capture à plus haut débit, que les captures sont étalées dans le temps. Tous les paquets sont bel et bien réceptionnés, mais l'image est évidemment saccadée. Cependant, le fonctionnement de l'algorithme SMPTE 20222 n'est en aucun cas modifié.

Le rapport du travail d'approfondissement a été rédigé tout au long de son déroulement.

6. Tests et analyses

Pour pouvoir tester le bon fonctionnement de mon implémentation, il est important d'avoir la possibilité de rejouer la même séquence vidéo et de travailler sur celle-ci dans le but d'ajouter des erreurs de transmission. Les séquences ont étés enregistrées par M. Kernen au sein de son laboratoire. Ces captures sont au format pcap[8]. A l'aide de l'outil *wireshark*, il est aisé de supprimer des paquets du flux media. Pour rejouer les différents flux, j'ai utilisé l'outil *tcpreplay*. Un guide à ce propos est disponible dans les annexes.

Les différents paramètres des captures sont les suivants :

BBC_ONE_SPTS_CBR_FEC_1D_8_by_5.pcap
- Débit : 8.5 Mbps
- Taille Matrice : 1D 8x5
- Addr flux media : 239.232.175.1:3000
- Addr flux fec colonne : 239.232.175.2:3002

DCM_5Mbps_2D_5_10_Non_aligned.pcap
- Débit : 5 Mbps
- Taille Matrice : 2D 10x5, non aligné
- Addr flux media : 239.232.175.202:3000
- Addr flux fec colonne : 239.232.175.202:3002
- Addr flux fec ligne : 239.232.175.202:3004

DCM_FEC_2D_ANNEX_A.pcap
- Débit : 8.5 Mbps
- Taille Matrice : 2D 10x6, non aligné
- Addr flux media : 239.232.0.222:3000
- Addr flux fec colonne : 239.232.0.223:3002
- Addr flux fec ligne : 239.232.0.224:3004

DCM_20Mbps_2D_5_5_Non_aligned.pcap
- Débit : 20 Mbps
- Taille Matrice : 2D 5x5, non aligné

DCM_20Mbps_2D_10_5_Non_aligned.pcap
- Débit : 20 Mbps
- Taille Matrice : 2D 10x5, non aligné
- Addr flux media : 239.232.175.202:3000
- Addr flux fec colonne : 239.232.175.202:3002
- Addr flux fec ligne : 239.232.175.202:3004

TVG420_FEC_2D.pcap
- Débit : 50 Mbps
- Taille Matrice : 2D 15x10, non aligné
- Addr flux media : 239.232.10.11:6000
- Addr flux fec colonne : 239.232.10.11:6002
- Addr flux fec ligne : 239.232.10.11:6004

[8] http://fr.wikipedia.org/wiki/Pcap

6.1. Test 1 : Détection et correction de pertes de paquets media avec effet cascade (17.05.11)

Ce test a été effectué à l'aide de la capture TVG420_FEC_2D.pcap (matrice 2D 15x10). Voulant tester les différents cas de figures de pertes (isolée, consécutive ou en cascade), j'ai supprimé 12.5% des paquets media. Ci-dessous, la manière pratiquée pour cette suppression.

																FEC	
															ROW	COL	COL
33970	33971	33972	33973	33974	33975	33976	33977	33978	33979	33980	33981	33982	33983	33984	52221	33357	33358
33985	33986	33987	33988	33989	33990	33991	33992	33993	33994	33995	33996	33997	33998	33999	52222	33359	
34000	34001	34002	34003	34004	34005	34006	34007	34008	34009	34010	34011	34012	34013	34014	52223	33360	33361
34015	34016	34017	34018	34019	34020	34021	34022	34023	34024	34025	34026	34027	34028	34029	52224	33362	
34030	34031	34032	34033	34034	34035	34036	34037	34038	34039	34040	34041	34042	34043	34044	52225	33363	33364
34045	34046	34047	34048	34049	34050	34051	34052	34053	34054	34055	34056	34057	34058	34059	52226	33365	
34060	34061	34062	34063	34064	34065	34066	34067	34068	34069	34070	34071	34072	34073	34074	52227	33366	33367
34075	34076	34077	34078	34079	34080	34081	34082	34083	34084	34085	34086	34087	34088	34089	52228	33368	
34090	34091	34092	34093	34094	34095	34096	34097	34098	34099	34100	34101	34102	34103	34104	52229	33369	33370
34105	34106	34107	34108	34109	34110	34111	34112	34113	34114	34115	34116	34117	34118	34119	52230	33371	
34120	34121	34122	34123	34124	34125	34126	34127	34128	34129	34130	34131	34132	34133	34134	52231	33372	33373
34135	34136	34137	34138	34139	34140	34141	34142	34143	34144	34145	34146	34147	34148	34149	52232	33374	
34150	34151	34152	34153	34154	34155	34156	34157	34158	34159	34160	34161	34162	34163	34164	52233	33375	33376
34165	34166	34167	34168	34169	34170	34171	34172	34173	34174	34175	34176	34177	34178	34179	52234	33377	
34180	34181	34182	34183	34184	34185	34186	34187	34188	34189	34190	34191	34192	34193	34194	52235	33378	33379
34195	34196	34197	34198	34199	34200	34201	34202	34203	34204	34205	34206	34207	34208	34209	52236	33380	
34210	34211	34212	34213	34214	34215	34216	34217	34218	34219	34220	34221	34222	34223	34224	52237	33381	33382
34225	34226	34227	34228	34229	34230	34231	34232	34233	34234	34235	34236	34237	34238	34239	52238	33383	
34240	34241	34242	34243	34244	34245	34246	34247	34248	34249	34250	34251	34252	34253	34254	52239	33384	33385
34255	34256	34257	34258	34259	34260	34261	34262	34263	34264	34265	34266	34267	34268	34269	52240	33386	
34270	34271	34272	34273	34274	34275	34276	34277	34278	34279	34280	34281	34282	34283	34284	52241	33387	33388
34285	34286	34287	34288	34289	34290	34291	34292	34293	34294	34295	34296	34297	34298	34299	52242	33389	
34300	34301	34302	34303	34304	34305	34306	34307	34308	34309	34310	34311	34312	34313	34314	52243	33390	33391
34315	34316	34317	34318	34319	34320	34321	34322	34323	34324	34325	34326	34327	34328	34329	52244	33392	
34330	34331	34332	34333	34334	34335	34336	34337	34338	34339	34340	34341	34342	34343	34344	52245	33393	33394

Figure 22 : test TVG420_FEC_2D.pcap 17.05.11

Du moment où un paquet fec peut corriger une perte d'un paquet media, la récupération de celui-ci a lieu. Dans le cas où les paquets récupérés permettent à leur tour de récupérer d'autres paquets media, l'effet cascade se produit. Par exemple, lors de l'arrivée du paquet fec ligne 52224, aucun paquet ne peut être récupéré (34024-34029). Cependant à l'arrivée des paquets fec colonnes, tous les paquets sont récupérés. Lors de la récupération de l'avant-dernier paquet de la ligne, un effet cascade se déclenche et récupère le dernier de la ligne.

Ci-dessous, une partie des logs montrant la bonne récupération :

```
rtp debug: SMPTE2022 : 52221,33970:15,1
rtp debug: SMPTE2022 lostPMedia : 33980
rtp debug: SMPTE2022 recup media: 33980, ts 27215169
rtp debug: SMPTE2022 : 52222,33985:15,1
rtp debug: SMPTE2022 : 52223,34000:15,1
rtp debug: SMPTE2022 : 52224,34015:15,1
rtp debug: SMPTE2022 lostPMedia : 34024
rtp debug: SMPTE2022 lostPMedia : 34025
rtp debug: SMPTE2022 lostPMedia : 34026
rtp debug: SMPTE2022 lostPMedia : 34027
rtp debug: SMPTE2022 lostPMedia : 34028
rtp debug: SMPTE2022 lostPMedia : 34029
rtp debug: SMPTE2022 : 52225,34030:15,1
rtp debug: SMPTE2022 lostPMedia : 34030
```

```
rtp debug: SMPTE2022 recup media: 34030, ts 27216274
rtp debug: SMPTE2022 : 52226,34045:15,1
rtp debug: SMPTE2022 : 52227,34060:15,1
rtp debug: SMPTE2022 : 52228,34075:15,1
rtp debug: SMPTE2022 lostPMedia : 34080
rtp debug: SMPTE2022 recup media: 34080, ts 27217384
rtp debug: SMPTE2022 : 52229,34090:15,1
rtp debug: SMPTE2022 : 52230,34105:15,1
rtp debug: SMPTE2022 Matrix change size 155
rtp debug: SMPTE2022 : 33357,33979:15,10
rtp debug: SMPTE2022 lostPMedia : 34024
rtp debug: SMPTE2022 recup media: 34024, ts 27216141
rtp debug: SMPTE2022 : 33358,33980:15,10
rtp debug: SMPTE2022 lostPMedia : 34025
rtp debug: SMPTE2022 recup media: 34025, ts 27216165
rtp debug: SMPTE2022 : 52231,34120:15,1
rtp debug: SMPTE2022 lostPMedia : 34130
rtp debug: SMPTE2022 recup media: 34130, ts 27218506
rtp debug: SMPTE2022 : 33359,33996:15,10
rtp debug: SMPTE2022 lostPMedia : 34026
rtp debug: SMPTE2022 recup media: 34026, ts 27216189
rtp debug: SMPTE2022 : 52232,34135:15,1
rtp debug: SMPTE2022 : 33360,34012:15,10
rtp debug: SMPTE2022 lostPMedia : 34027
rtp debug: SMPTE2022 recup media: 34027, ts 27216213
rtp debug: SMPTE2022 : 33361,34013:15,10
rtp debug: SMPTE2022 lostPMedia : 34028
rtp debug: SMPTE2022 recup media: 34028, ts 27216226
rtp debug: SMPTE2022 recup media: 34029, ts 27216250
```

Figure 23 : log test TVG420_FEC_2D.pcap 17.05.11

6.2. Test 2 : Détection et correction de pertes de paquets media (24.05.11)

Ce test a été effectué à l'aide de la capture DCM_FEC_2D_ANNEX_A.pcap (matrice 2D 10x6). J'ai supprimé 2% des paquets media. Tous ont étés détectés et récupérés. Ci-dessous, une partie des logs d'une version de VLC sans le fonctionnement de l'algorithme SMPTE 2022 et d'une version avec. On constate que tous les paquets ont été récupérés dans la version avec l'algorithme SMPTE 2022.

reconstructed media put into queue 50425	rtp warning: 1 packet(s) lost : 50425
reconstructed media put into queue 50445	rtp warning: 1 packet(s) lost : 50445
reconstructed media put into queue 50456	rtp warning: 1 packet(s) lost : 50456
reconstructed media put into queue 50488	rtp warning: 2 packet(s) lost : 50489
reconstructed media put into queue 50489	
reconstructed media put into queue 50517	rtp warning: 1 packet(s) lost : 50517
reconstructed media put into queue 50540	rtp warning: 1 packet(s) lost : 50540
reconstructed media put into queue 50575	rtp warning: 2 packet(s) lost : 50576
reconstructed media put into queue 50576	
reconstructed media put into queue 50607	rtp warning: 2 packet(s) lost : 50607
reconstructed media put into queue 50608	
reconstructed media put into queue 50658	rtp warning: 1 packet(s) lost : 50658
reconstructed media put into queue 50702	rtp warning: 2 packet(s) lost : 50703
reconstructed media put into queue 50701	
reconstructed media put into queue 50747	rtp warning: 1 packet(s) lost : 50747
reconstructed media put into queue 50767	rtp warning: 1 packet(s) lost : 50767
reconstructed media put into queue 50802	rtp warning: 2 packet(s) lost : 50803
reconstructed media put into queue 50803	

Figure 24 : log test DCM_FEC_2D_ANNEX_A.pcap 24.05.11

6.3. Test 3 : Détection et correction de pertes de paquets media (24.05.11)

Ce test a été effectué à l'aide de la capture TVG420_FEC_2D.pcap (matrice 2D 15x10). J'ai supprimé 1.5% des paquets media. Tous ont étés détectés et récupérés. Ci-dessous, une partie des logs d'une version de VLC sans le fonctionnement de l'algorithme SMPTE 2022 et d'une version avec. On constate que tous les paquets ont été récupérés dans la version avec l'algorithme SMPTE 2022. La récupération n'est pas forcément exécutée dans l'ordre des numéros de paquets, à cause de la matrice. Dans ce test, le paquet 38225 a été récupéré avant les paquets 38201 et 38202.

reconstructed media put into queue 38100	rtp warning: 1 packet(s) lost : 38100
reconstructed media put into queue 38118	rtp warning: 2 packet(s) lost : 38119
reconstructed media put into queue 38119	
reconstructed media put into queue 38163	rtp warning: 1 packet(s) lost : 38163
reconstructed media put into queue 38225	**rtp warning: 2 packet(s) lost : 38202**
reconstructed media put into queue 38201	**rtp warning: 1 packet(s) lost : 38225**
reconstructed media put into queue 38202	
reconstructed media put into queue 38249	rtp warning: 2 packet(s) lost : 38250
reconstructed media put into queue 38250	
reconstructed media put into queue 38304	rtp warning: 1 packet(s) lost : 38304
reconstructed media put into queue 38319	rtp warning: 3 packet(s) lost : 38321
reconstructed media put into queue 38361	rtp warning: 1 packet(s) lost : 38361
reconstructed media put into queue 38320	
reconstructed media put into queue 38321	
reconstructed media put into queue 38386	rtp warning: 1 packet(s) lost : 38386
reconstructed media put into queue 38403	rtp warning: 1 packet(s) lost : 38403
reconstructed media put into queue 38421	rtp warning: 1 packet(s) lost : 38421

Figure 25 : log test TVG420_FEC_2D.pcap 24.05.11

6.4. Test 4 : Détection et correction de pertes de paquets media (24.05.11)

Ce test a été effectué à l'aide de la capture DCM_5Mbps_2D_5_10_Non_aligned.pcap (matrice 2D 10x5). Cette capture comportait déjà une perte de paquets. Cependant, tous ont étés détectés et récupérés. Ci-dessous, une partie des logs d'une version de VLC sans le fonctionnement de l'algorithme SMPTE 2022 et d'une version avec. On constate que tous les paquets ont été récupérés dans la version avec l'algorithme SMPTE 2022.

```
reconstructed media put into queue 59057   rtp warning: 3 packet(s) lost : 59059
reconstructed media put into queue 59058
reconstructed media put into queue 59059
reconstructed media put into queue 60482   rtp warning: 3 packet(s) lost : 60484
reconstructed media put into queue 60483
reconstructed media put into queue 60484
reconstructed media put into queue 61906   rtp warning: 4 packet(s) lost : 61909
reconstructed media put into queue 61907
reconstructed media put into queue 61908
reconstructed media put into queue 61909
reconstructed media put into queue 63331   rtp warning: 4 packet(s) lost : 63334
reconstructed media put into queue 63332
reconstructed media put into queue 63333
reconstructed media put into queue 63334
reconstructed media put into queue 64756   rtp warning: 4 packet(s) lost : 64759
reconstructed media put into queue 64757
reconstructed media put into queue 64758
reconstructed media put into queue 64759
reconstructed media put into queue 645     rtp warning: 4 packet(s) lost : 648
reconstructed media put into queue 646
reconstructed media put into queue 647
reconstructed media put into queue 648
reconstructed media put into queue 2069    rtp warning: 5 packet(s) lost : 2073
reconstructed media put into queue 2070
reconstructed media put into queue 2071
reconstructed media put into queue 2072
reconstructed media put into queue 2073
```

Figure 26 : log test DCM_5Mbps_2D_5_10_Non_aligned.pcap 24.05.11

Les numéros de paquets étant sur 16 bits, la valeur maximum est donc 65535. Lorsque l'on a atteint cette valeur, nous recommençons tout simplement à compter à partir de 0. Dans cet exemple, l'algorithme SMPTE 2022 n'a pas eu de problème à ce propos.

Lors de la visualisation de cette capture dans la version sans l'algorithme SMPTE 2022, des macros blocs sont déformés et on entend une discontinuité dans l'audio. Tandis que dans la version avec l'algorithme SMPTE 2022, on ne perçoit aucune déformation de l'image et de l'audio.

7. Conclusion

Ce projet d'approfondissement m'a permis une mise en pratique de mes connaissances acquises au cours du Master et Bachelor, plus particulièrement celles concernant les transmissions multimédia. Celui-ci m'a permis aussi de participer au développement du lecteur vidéo le plus utilisé actuellement.

J'ai rencontré plusieurs difficultés durant l'intégration de l'algorithme SMPTE 2022 au lecteur VLC. La majorité de celles-ci étaient dues à la programmation en C. Une autre lors de la lecture des captures à l'aide du programme *tcpreplay*, celui-ci limitant le débit en sortie à 10Mbps.

Cependant, à l'aide de mes connaissances préalables, de celles acquises au cours du Master et dans certains cas l'aide de personnes compétentes dans le domaine, j'ai pu minimiser la perte de temps lors de la rencontre des difficultés.

Malgré les difficultés rencontrées, ce projet s'est achevé dans les temps avec une version fonctionnelle du lecteur VLC intégrant l'algorithme SMPTE 2022.

Dans le souci d'améliorer son implémentation, une optimisation du code ainsi que la correction de bugs mineurs sont à prévoir.

Dans la continuité de ce travail, il serait intéressant d'implémenter dans le programme *Wireshark* le décodeur des champs de la norme SMPTE 2022. Cette implémentation serait utile au monde professionnel utilisant cette norme. Elle permettrait aussi de participer au développement d'un des logiciels les plus utilisés dans l'analyse des flux réseaux.

Genève, le 29 mai 2011

Rossier Jérémie

8. Table des Figures

9. Bibliographie

9.1. Standard SMPTE

Forward Error Correction for Real-Time Video/Audio Transport Over IP Networks

The Society of Motion Picture and Television Engineers

May 2007

S2022-1-2007

Unidirectional Transport of Constant Bit Rate MPEG-2 Transport Streams on IP Networks

The Society of Motion Picture and Television Engineers

May 2007

S2022-2-2007

Unidirectional Transport of Variable Bit Rate MPEG-2 Transport Streams on IP Networks

The Society of Motion Picture and Television Engineers

April 2010

S2022-3-2010

9.2. Mémoire

Fischer D. (2008). *MPEG scalability & video on demand*. hepia.

Ricciuti L. (2010). *Qualité de service améliorée sur IPTV*. hepia.

9.3. RFC

RTP: A Transport Protocol for Real-Time Applications

H. Schulzrinne (Columbia University), S. Casner (Packet Design), R. Frederick (Blue Coat Systems Inc.), V. Jacobson (Packet Design).

July 2003

RFC3550

An RTP Payload Format for Generic Forward Error Correction

J. Rosenberg (dynamicsoft), H. Schulzrinne (Columbia University).

December 1999

RFC2733

9.4. Web

VideoLAN Organization, *VideoLAN*, http://www.videolan.org/vlc/

VideoLAN Organization, *VideoLAN Wiki*, http://wiki.videolan.org/

10. Annexes

10.1. Guide de téléchargement et compilation du code source de VLC

Tout d'abord, il faut télécharger le code source de VLC à l'aide de la commande ci-dessous[9] dans le répertoire désiré. L'avantage d'utiliser "git" est une gestion de version du code source, ce qui permet plus aisément de travailler à plusieurs dessus.

```
git clone git://git.videolan.org/vlc.git –depth 1
```

Le champ profondeur permet de ne pas récupérer toutes les versions du code source. Pour une mise à jour du code source et lire les commentaires de celle-ci, ci-dessous la commande :

```
git pull ## Mise à jour du code
git log  ## Affichage des commentaires
```

Pour pouvoir compiler le code source, il faut s'assurer de disposer sur notre OS de toutes les librairies nécessaires.[10]

```
sudo apt-get build-dep vlc

sudo apt-get -y install libvorbis-dev libogg-dev libtheora-dev
speex libspeex-dev flac libflac-dev \
  x264 libx264-dev a52-0.7.4 liba52-0.7.4-dev mpeg2dec libmpeg2-
4-dev faad libfaad-dev faac libfaac-dev \
  lame libmp3lame-dev ffmpeg libavdevice-dev libmad0 libmad0-dev
dirac libdirac-dev liboil-dev libschroedinger-dev \
  libdca-dev twolame libtwolame-dev libmpcdec-dev libvorbisidec1
libvorbisidec-dev libass-dev libass4 libebml2 \
  libebml-dev libmatroska2 libmatroska-dev libdvbpsi6 libdvbpsi-
dev libmodplug1 libmodplug-dev libshout3 libshout3-dev \
  libdvdread4 libdvdnav4 libdvdnav-dev livemedia-utils
liblivemedia-dev libcddb2 libcddb2-dev libcdio10 libcdio-dev \
  libcdio-utils vcdimager libvcdinfo0 libvcdinfo-dev libgpg-
error0 libgpg-error-dev libgcrypt11 libgcrypt11-dev \
  gnutls-bin libgnutls26 libgnutls-dev libdap10 libdap-bin
libdap-dev libxml2 libxml2-dev libpng12-0 libpng12-dev \
  libjpeg8 libtiff4 libsdl1.2-dev libsdl-image1.2 libsdl-
image1.2-dev libc-bin gettext libfreetype6 libfreetype6-dev \
  libfribidi-dev libfribidi0 zlib1g zlib1g-dev libtag1-dev
libcaca0 libcaca-dev caca-utils libqt4-core libqt4-dev \
  libportaudio2 libportaudio-dev libupnp-dev libupnp4 libupnp3
libexpat1 libexpat1-dev yasm libxcb-xv0 libxcb-xv0-dev \
  libx11-xcb1 libx11-xcb-dev
```

[9] wiki.videolan.org/Git
[10] http://wiki.videolan.org/Contrib_Status

Il faut mettre à jour l'une des librairies concernant la partie streaming.

```
wget http://www.live555.com/liveMedia/public/live555-
latest.tar.gz
tar xvzf live555-latest.tar.gz
cd live/
sh genMakefiles linux-64bit ## 64bit ou 32bit selon l'OS
make
```

Afin de ne pas polluer le répertoire du code source de VLC, avant de compiler, il est conseillé de copier tous les fichiers vers un dossier temporaire (ex : vlc_make).

```
rsync -vrt --exclude=.svn --exclude=.git vlc/vlc_make/
```

L'avant-dernière étape avant la compilation est la vérification de la disponibilité des librairies et la création du makefile à l'aide du bootstrap.

```
cd vlc_make/
./bootstrap
```

La dernière étape avant la compilation est la configuration du make en lui précisant quels sont les modules que nous voulons intégrer (attention, le chemin d'accès à la librairie live555 à adapter).

```
./configure --enable-x11 --enable-xvideo --enable-sdl --enable-
avcodec --enable-avformat --enable-swscale --enable-mad --
enable-a52 --enable-dca --enable-libmpeg2 --enable-dvdnav --
enable-faad --enable-vorbis --enable-ogg --enable-theora --
enable-mkv --enable-freetype --enable-fribidi --enable-speex --
enable-flac --enable-live555 --with-live555-
tree=/home/jeremie/Documents/lib_live555/live/ --enable-caca --
disable-gme --enable-skins2 --enable-alsa --enable-qt4 --enable-
ncurses --enable-opengl --enable-debug
```

Il ne reste plus qu'à compiler et exécuter l'exécutable créé.

```
make
## wait and see
./vlc
```

10.2. Utilisation du debugger

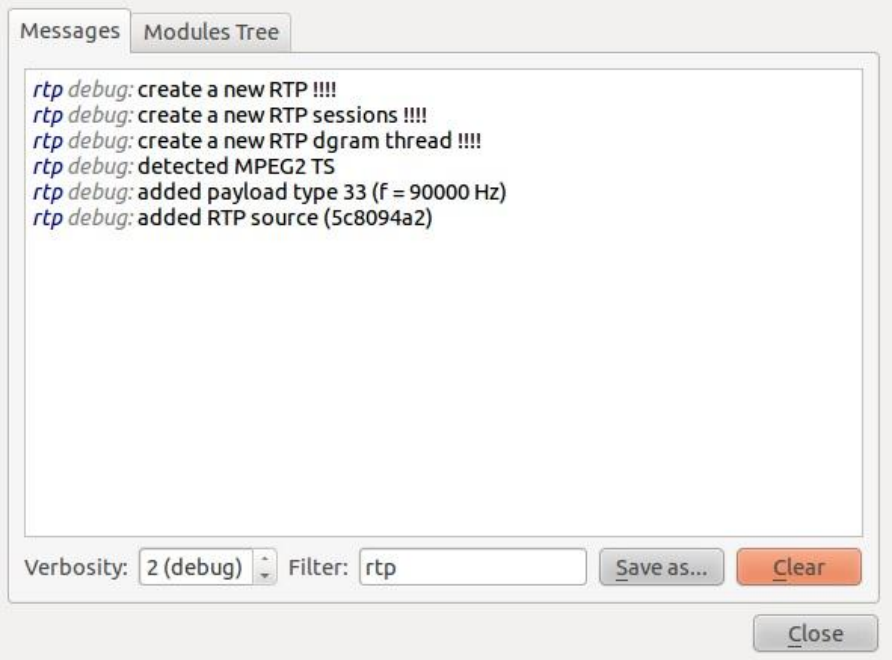
Il est important de pouvoir utiliser le debugger, celui-ci aidant grandement la vérification du bon fonctionnement de ce que l'on développe.

Avant de pouvoir l'utiliser lors de l'exécution de VLC, il faut que, lors de la compilation de celui-ci, le debugger ait été activé.

```
./configure ... --enable-debug
```

Lors de l'exécution de VLC, l'affichage des messages du debugger et de l'arbre des modules chargés est disponible dans le menu "Tools/Messages" comme illustré ci-dessous.

Le debugger dispose d'une verbosité à trois niveaux : "errors", "warning" ou "debug". Ci-dessous, un exemple lors de la réception d'un flux RTP. Les trois premières lignes ont étés rajoutées au code source pour vérifier l'ordre d'instanciation des différentes parties du module RTP.



L'onglet "Modules Tree" permet la vérification du bon chargement des modules concernés, comme illustré ci-dessous avec le même exemple que précédemment.

La lecture du debugger est aussi possible via la console, mais il n'est pas possible d'ajouter un filtre comme il est possible avec les messages et généralement la fenêtre est limitée en caractères.

10.3. Rejouer un flux media et ses flux FEC associés

Pour pouvoir tester le bon fonctionnement de mon implémentation, il est important de pouvoir rejouer la même séquence vidéo et pouvoir travailler sur celle-ci dans le but d'ajouter des erreurs de transmission. Les séquences ont étés enregistrées par M. Kernen, celles-ci sont en format pcap[11]. A l'aide de l'outil *wireshark*, il est aisé de supprimer des paquets du flux. Pour rejouer les différents flux, j'ai utilisé l'outil *tcpreplay*. Les paramètres à entrer sont : l'interface sur laquelle on veut rejouer la capture et le fichier pcap concerné. Ci-dessous, un exemple de commande que j'ai utilisé :

```
sudo tcpreplay --intf1=eth0 --timer=rdtsc TVG420_FEC_2D.pcap
```

Cependant, les plateformes et le matériel utilisé lors de mon projet ont limité le débit en sortie. En effet, même en modifiant la méthode de gestion du temps ou l'utilisation de la ram pour les accès au fichier de capture, le débit ne dépassait pas les 10Mbps. Cela entrainait un étalement des captures dans le temps. Le fonctionnement de l'algorithme SMPTE 2022 restait inchangé, mais l'affichage était saccadé.

[11] http://fr.wikipedia.org/wiki/Pcap