

ECOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

Département Informatique

64 avenue Jean Portalis

37200 Tours, France

Tél. +33 (0)2 47 36 14 14

polytech.univ-tours.fr

Projet Recherche & Développement 2019-2020

Capture de génériques TV

Tuteur académique
Mathieu DELALANDRE

Étudiant
Yohann BENÉTREULT (DI5)

11 avril 2020



Liste des intervenants

Nom	Email	Qualité
Yohann BENÉTREULT	yohann.benetreault@etu.univ-tours.fr	Étudiant DI5
Mathieu DELALANDRE	mathieu.delalandre@univ-tours.fr	Tuteur académique, Département Informatique



Avertissement

Ce document a été rédigé par Yohann Benétreault susnommé l'auteur.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Mathieu Delalandre susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



Pour citer ce document

Yohann Benétreault, *Capture de génériques TV*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2019-2020.

```
@mastersthesis{
  author={Benétreault, Yohann},
  title={Capture de génériques TV},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université François Rabelais de Tours},
  address={Tours, France},
  year={2019-2020}
}
```

Table des matières

Liste des intervenants	a
Avertissement	b
Pour citer ce document	c
Table des matières	i
Table des figures	iv
1 Introduction	1
1 Contexte.....	1
2 Objectifs.....	1
3 Hypothèses	2
4 Bases méthodologiques	2
2 Description générale	3
1 Environnement du projet	3
2 Fonctionnalité du système.....	3
3 Structure générale du système	3
3 Veille technologique	5
1 Bus de communication PCI.....	5
1.1 PCI express	5
1.1.1 Architecture du PCI express	6
1.1.2 Protocole du PCI express.....	6
1.1.3 Utilisation du PCI express dans le cadre du projet	6

4	Analyse et conception	7
1	Parser XMLTV.....	7
2	Capture des vidéos	7
5	Mise en œuvre	8
1	Installation de la nouvelle station TV	8
1.1	Architecture interne finale	8
1.2	Capacités de la station TV	8
1.3	Multitunner.....	9
2	Parser XMLTV.....	9
2.1	Parsing par événements	9
2.2	Technologies.....	9
2.2.1	Langage de programmation	9
2.2.2	Maven.....	10
2.2.3	Librairie : SAX Parser	10
2.2.4	JUnit 5	10
2.3	Limites	10
2.4	Risques	10
2.5	Reste à faire et évolution	10
3	Capture de vidéos.....	11
3.1	Technologies.....	11
3.1.1	Langage de programmation	11
3.1.2	SDK : AVerMedia	11
3.2	Limites	11
3.3	Risques	11
3.4	Reste à faire et évolution	12
6	Bilan et conclusion	13
1	Bilan sur la qualité	13
2	Bilan auto-critique sur la gestion de projet	13
	Bibliographie	15
	Annexes	16
A	Spécifications systèmes	17
1	Choix de l'architecture	17
1.1	Contraintes liées au choix des tunners TNT	17
2	Solutions envisagées	18
2.1	Multitunner professionnel (non retenue).....	18

2.2	Augmentation du nombre de tunners TNT (retenue).....	18
3	Contraintes liées au choix des cartes de capture vidéo	18
3.1	Solution envisagée	19
B	Spécifications fonctionnelles	20
1	Parsing du fichier XMLTV	20
1.1	Identification de la fonctionnalité de parsing du fichier XMLTV	20
1.2	Description de la fonctionnalité de parsing du fichier XMLTV	20
2	Enregistrement des génériques TV	21
2.1	Identification de la fonctionnalité d'enregistrement	21
2.2	Description de la fonctionnalité d'enregistrement	21
3	Ordonnancement de l'enregistrement	21
C	Spécifications non fonctionnelles	22
1	Contraintes de développement et conception	22
2	Contraintes de fonctionnement et d'exploitation	22
2.1	Performances.....	23
2.2	Capacités.....	23
2.2.1	Disque dur	23
2.2.2	Processeur	23
2.2.3	Contrôlabilité.....	23
2.2.4	Sécurité	23
2.2.5	Intégrité	24
2.3	Maintenance et évolution du système	24
D	Gestion de projet	25
1	Découpage des tâches.....	25
2	Diagramme de Gantt	25
3	Mise à jour de la planification.....	26
4	Planification finale	26
E	Documents d'installation	27
F	Version des logiciels	28
G	Documents d'utilisation	29
H	Dossiers de tests	30
1	Tests fonctionnels	30
2	Tests unitaires	30

Table des figures

1 Introduction

1	Représentation d'une méthode agile	2
---	--	---

2 Description générale

2	Schéma général de l'installation	3
3	Schéma des modules du système	4

3 Veille technologique

4	Représentation de l'architecture PCI express (Image extraite de « Structured Computer Organisation, 6th Edition »)	6
5	Structure des paquets (Image extraite de « Structured Computer Organisation, 6th Edition »)	6
6	Débit requis par TF1 HD.....	6

4 Analyse et conception

7	Diagramme d'activité du parser XMLTV	7
8	Diagramme de classe du parser XMLTV.....	7
9	Diagramme d'activité des captures vidéo	7

5 Mise en œuvre

10	Architecture interne de la station TV	8
11	Solution retenue pour le rack.....	9

A Spécifications systèmes

12	Configurations du rack.....	17
----	-----------------------------	----

C Spécifications non fonctionnelles

13	Graphe du nombre de génériques à capturer par rapport au temps.....	24
----	---	----

D Gestion de projet

14	Diagramme de Gantt - Première partie	25
15	Diagramme de Gantt - Deuxième partie.....	26
16	Diagramme de Gantt - Première partie	26
17	Diagramme de Gantt - Deuxième partie.....	26

H Dossiers de tests

18	Cahier de recette.....	30
19	Exemple de la validité du build du projet	30
20	Exemple de la validité des tests unitaires.....	30

1

Introduction

Ce document a pour but de définir les spécifications système et le plan de développement concernant le projet de recherche et développement : « Capture de génériques TV sur station ».

Ce projet est proposé par Monsieur Mathieu DELALANDRE, docteur en informatique et maître de conférences au LIFAT¹ (Université de Tours) qui représentera la maîtrise d'ouvrage. Il sera réalisé par l'élève-ingénieur Yohann BENETREAUULT qui représentera la maîtrise d'œuvre.

1 Contexte

La consommation intensive de contenus multimédias, notamment au travers de la télévision, ainsi que les nouvelles technologies ont fait émerger de nouvelles habitudes. Les utilisateurs sont devenus « dual-screeners² » : ils utilisent leurs terminaux mobiles (Smartphone, tablette, ...) en complément des programmes TV proposés par les chaînes de télévisions ou de la vidéo à la demande pour chercher des informations, consulter des réseaux sociaux en relation avec la télévision et les programmes, etc.

Mathieu DELALANDRE et Jordan NICOT, ingénieur informatique, ont eu un projet de start-up pour répondre aux problématiques liées à ce contexte. Le but principal de cette start-up est de créer une plateforme qui proposera un guide TV/VoD nouvelle génération. Ce guide sera ergonomique, adapté aux besoins des utilisateurs.

Dans cette optique, des modules permettant d'apporter des fonctionnalités à l'utilisateur ont été pensés. Dans le cadre de ce projet, nous considérerons la capture des génériques TV grâce à une planification puis la recherche en temps réel de ces génériques TV sur les chaînes.

2 Objectifs

Dans un premier temps, l'objectif est de constituer une base de données de génériques TV extraits des flux télévisuels. Nous devons être capables de capturer plusieurs flux en parallèle, ce qui implique plusieurs contraintes. Une capture manuelle des génériques seraient trop longue et peu efficace, nous devons donc automatiser cette tâche.

1. LIFAT : Laboratoire d'informatique fondamentale et appliquée de Tours

2. Dual-screener : Personne utilisant deux appareils à écran en même temps

Pour automatiser cette tâche nous devons faire en sorte que le module de capture soit autonome et définir des intervalles de capture. La définition de ces intervalles se fera grâce à un fichier XMLTV qui est un fichier au format XML proposé gratuitement et qui permet de récupérer la programmation de la télévision. Il faudra donc un module permettant d'extraire les informations désirées de ce fichier et qu'elles soient accessibles par le module capturant les flux vidéo. Enfin, pour visualiser les flux TV des chaînes TV nous devons avoir une interface graphique les affichant.

Après la capture des génériques TV, l'objectif est de créer des signatures sur ces génériques [cette partie fait l'objet d'un autre sujet de PRD affecté à un autre étudiant, nous ne considérerons pas cette partie dans ce document] pour ensuite appliquer un algorithme de détection sur les flux en temps réel.

3 Hypothèses

La capture de génériques grâce aux cartes de capture, qui se fait de manière logicielle et donc qui prend des ressources CPU de la machine, ne sera pas un facteur limitant. En effet, les captures sur les chaînes ne seront potentiellement pas toutes effectuées en même temps étant donné que les programmations TV sont différentes selon les chaînes. Cependant, si nous observons des charges CPU trop importantes lors de la capture, nous devons chercher une solution alternative.

4 Bases méthodologiques

Le projet étant découpé en modules, la méthode choisie pour ce projet est une méthode agile. En effet, les itérations correspondront à la production d'un module avec la livraison de ce module à la fin de l'itération. De plus Mathieu DELALANDRE, encadrant et MOA de ce projet, est disponible régulièrement pour faire des réunions. La méthode *Crystal Clear* semble la mieux adaptée au projet car elle permet :

- Des livraisons fréquentes
- Une amélioration réflexive
- Une communication permettant à chaque membre de l'équipe d'apprendre des choses nouvelles (même de manière passive)
- Aucune mise à l'écart des membres de l'équipe, tout le monde est impliqué et entendu
- Une focalisation forte ce qui empêche de changer trop souvent de contexte
- Un échange fort avec le client ou un expert
- Un environnement technique avec de nombreux outils spécifiques au développement

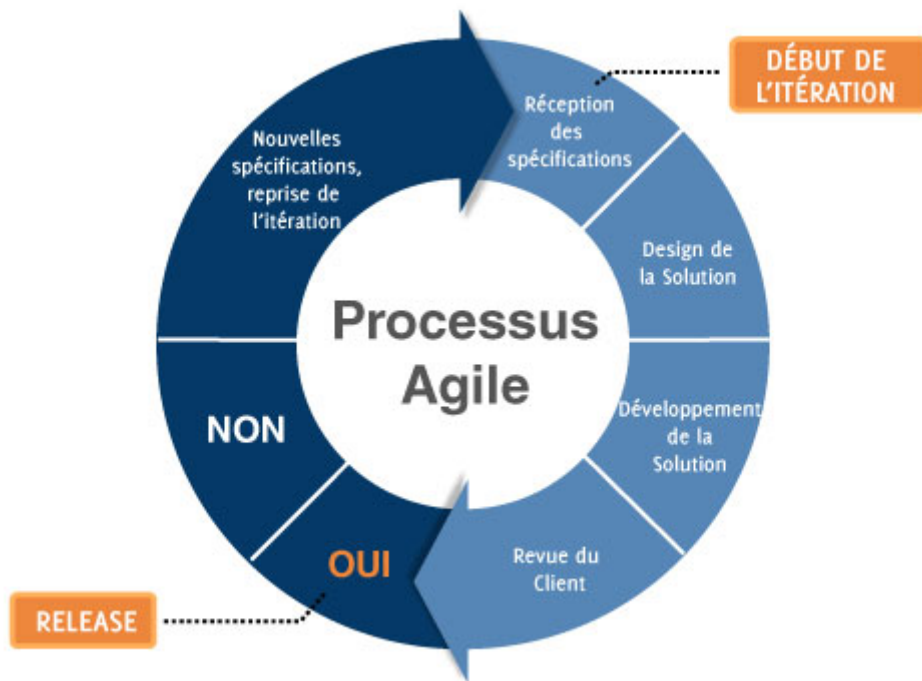


Figure 1 – Représentation d'une méthode agile

2

Description générale

1 Environnement du projet

Le projet s'intégrera à la station TV mise en place dans la salle des doctorants. Cette station est composée, actuellement, de quatre tunners TNT. Cependant, cette station va augmenter en capacité et passera au minimum à vingt tunners TNT. Cette augmentation de volume est le sujet de projet de fin d'étude proposé à des élève-ingénieurs d'informatique industrielle. La configuration finale de la station TV est cependant arrêtée et sera mise en place avant la phase de développement des applications de ce sujet. La machine sur laquelle est mise en place la station TV utilise Windows 10.

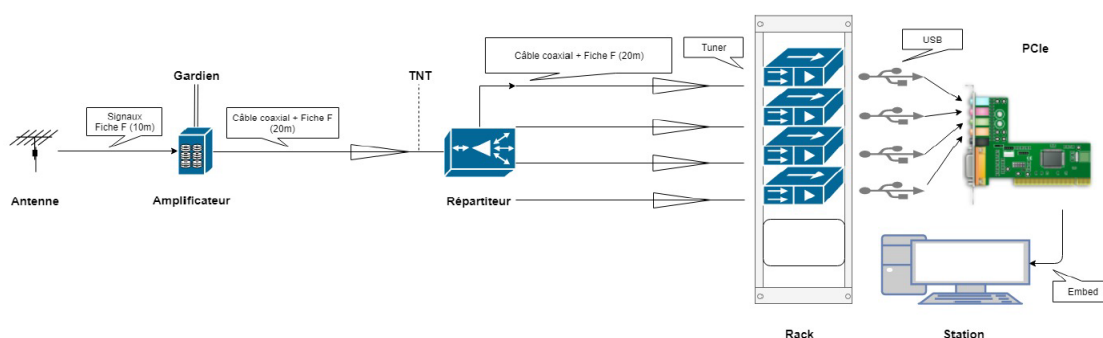


Figure 2 – Schéma général de l'installation

Une carte GPU NVidia RTX 2070 Super ainsi que d'autres cartes de capture AVerMedia CE314-HN et CL332-HN seront installées lors de l'augmentation en capacité de la station TV. Le but de la carte GPU est d'accélérer les calculs de comparaison d'images dans le cadre de la détection en temps réel.

Les cartes de type AVerMedia CE314-HN et CL332-HN sont utilisées pour récupérer les flux TV. Ces cartes possèdent une API permettant d'effectuer des opérations, notamment de capture, sur les données de sortie. Le module de capture des génériques TV utilisera donc cette API pour enregistrer les données vidéo.

2 Fonctionnalité du système

Le système aura deux fonctionnalités principales : le parsing d'un fichier XML permettant d'extraire des intervalles de temps et l'enregistrement des vidéos.

3 Structure générale du système

Le système sera décomposé en trois modules, chacun effectuant une tâche particulière au sein du système (cf. figure 3). Les modules seront les suivants :

- **Parsing du fichier XMLTV** : ce module devra extraire des intervalles de temps à partir d'un fichier XML afin de les passer en paramètre du module d'enregistrement des génériques. Le but est de ne pas avoir à capturer des durées de vidéo trop longue pour éviter de nombreuses contraintes comme par exemple la taille des fichiers.
- **Enregistrement des génériques TV** : ce module pourra prendre en entrée des intervalles de temps ou une date de début et une durée afin de capturer un flux vidéo. Cependant, il devra aussi être capable de capturer plusieurs flux en parallèle.
- **Ordonnancement des enregistrements** : ce module servira à optimiser la capture des génériques TV.

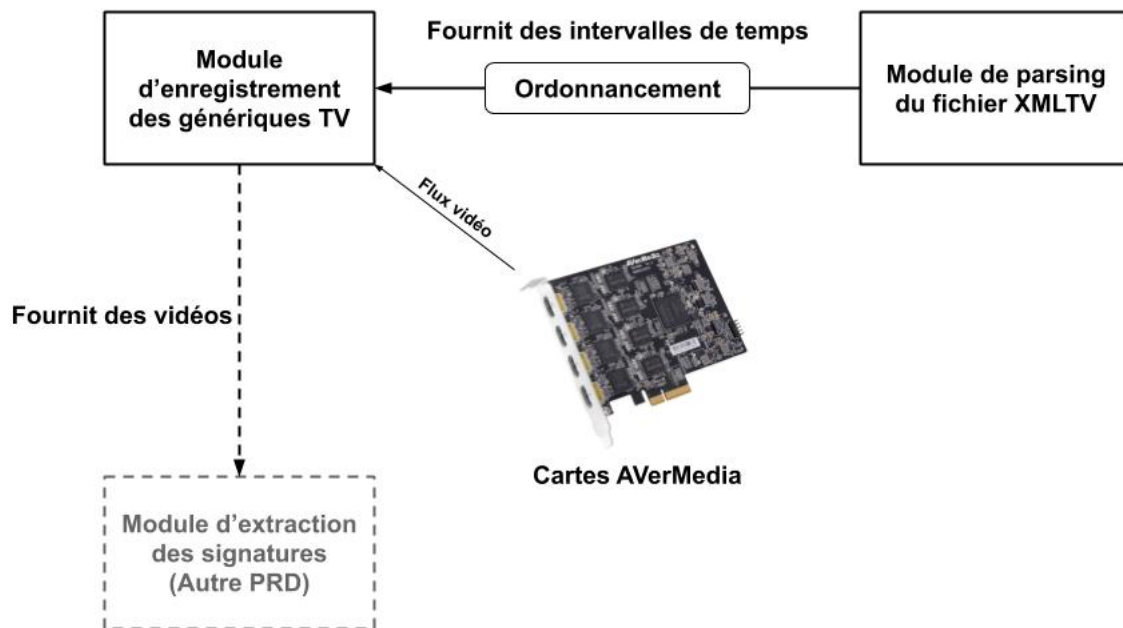


Figure 3 – Schéma des modules du système

3

Veille technologique

1 Bus de communication PCI

Le nombre grandissant des interfaces graphiques proposées par les logiciels dans les années 90, notamment Windows, ont soulevé des contraintes lors du transfert de données. Les interfaces graphiques, gourmandes en ressources, ont obligé la mise en place d'un nouveau type de communication, plus rapide. En 1990, *Intel* développa un nouveau bus de communication avec une bande passante beaucoup plus large nommé « Peripheral Component Interconnect », ou *PCI*. Afin d'encourager les producteurs de matériel informatique à utiliser cette technologie, *Intel* mis à disposition son nouveau bus.

Initialement, ce bus avait une largeur de bande de 133 MB/sec, contre 16,7 MB/sec pour le bus ISA – soit environ huit fois plus – et contre 33,3 MB/sec pour le bus EISA – soit environ quatre fois plus. Cependant, les interfaces graphiques requéraient environ 135 MB/sec. En 1993, PCI 2.0 a été proposé par *Intel*, puis une amélioration de cette deuxième version en 1995 pour finir avec un PCI 2.2 proposant une bande passante de 528 MB/sec. Cependant, bien que ce bus propose une bande passante très importante pour l'époque, de nombreuses contraintes le limitait. Aujourd'hui, le bus PCI est capable de supporter toutes les architectures existantes et est utilisé partout. Ce bus est asynchrone et fonctionne avec un modèle maître/esclaves.

1.1 PCI express

Les dernières versions de PCI sont largement viables pour nos systèmes modernes, cependant ce bus n'est plus l'élément central permettant à tous le matériel de communiquer. Nous avons de plus en plus de matériel possédant des entrées/sorties et leur vitesse est une contrainte pour le bus PCI. Cela impose à *Intel* de rajouter des ports sur les cartes pour les nouveaux types de matériel. Le volume des cartes sont donc de plus en plus importants et ne sont plus compatibles avec les systèmes ayant une place limitée, comme les ordinateurs portables par exemple. C'est ainsi que *Intel* développa le PCI express.

1.1.1 Architecture du PCI express

L'objectif du PCI express est de se défaire du modèle maître/esclaves et de proposer des connexions point à point extrêmement rapides. Pour cela, il met en place un switch plutôt qu'un partage du média par le matériel.

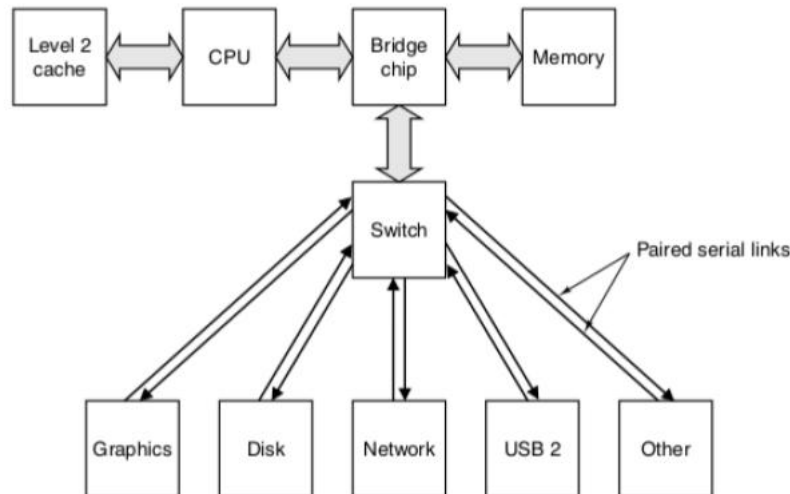


Figure 4 – Représentation de l'architecture PCI express (Image extraite de « Structured Computer Organisation, 6th Edition »)

Le PCI express s'inspire de l'architecture réseau et met en place un système de transfert de paquets ainsi que de *payload*. Cette architecture est donc comparable à un réseau.

1.1.2 Protocole du PCI express

Comme dit précédemment, PCI express s'inspire des réseaux et met en place un système de paquets. Ces paquets sont de la forme suivante :

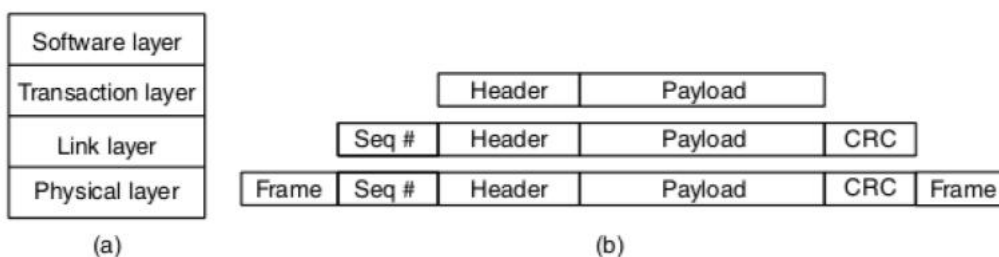


Figure 5 – Structure des paquets (Image extraite de « Structured Computer Organisation, 6th Edition »)

Il s'agit donc d'un standard et tout organisme voulant le mettre en place doit répondre aux contraintes du protocole.

1.1.3 Utilisation du PCI express dans le cadre du projet

Dans notre cas, les cartes de capture vidéo AVerMedia utilisent le bus PCIe. Plus précisément il s'agit du PCIe 3.0 (4x). Cette version a été introduite en 2012 et possède les caractéristiques suivantes :

- Fréquence : 100 MHz
- Bande passante : 8 GT/s
- Débit par ligne : 984,6 Mo/s

Puisque nous sommes en 4x, nos cartes utilisent donc quatre lignes pour un débit total d'environ 4 Go/s par carte. Les cartes AVerMedia CE314-HN ont quatre sorties vidéo, ce qui donne environ 1 Go/s par flux vidéo. Nous pouvons comparer cette valeur au débit requis par une chaîne de la télévision (TF1 HD¹) :

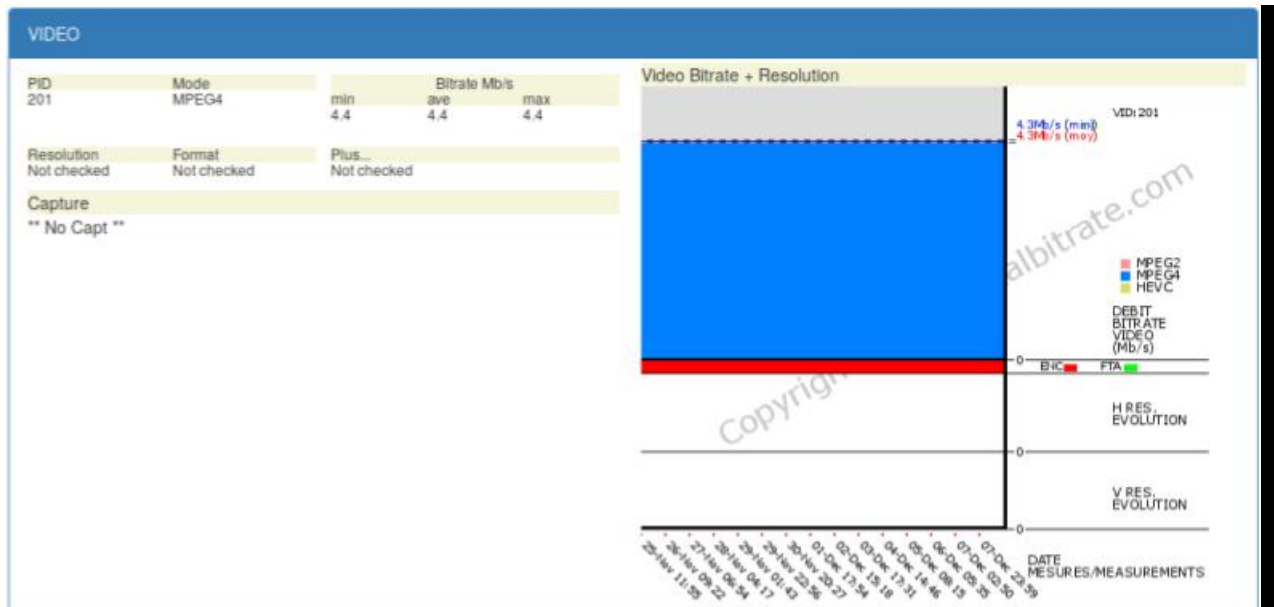


Figure 6 – Débit requis par TF1 HD

Le débit mesuré est en moyenne de 4,2 Mo/s soit 0,42 % de la capacité de transfert d'un flux par une carte. Les pics des données transmis par les chaînes de télévision est d'environ 15 Mo/s soit 1,5 % de la capacité. Ainsi, même dans les cas les plus défavorables, le bus PCIe garanti un bon acheminement des flux télévisuels par les cartes AVerMedia.

1. Valeurs fournies par le site [digitalbitrate](http://digitalbitrate.com)

4

Analyse et conception

1 Parser XMLTV

Le parser XMLTV est le premier outil de ce projet et permet de récupérer des intervalles de temps à partir d'un programme TV. Pour parvenir à cela, le parser doit effectuer les tâches décrites par le diagramme d'activité illustré par la figure suivante.

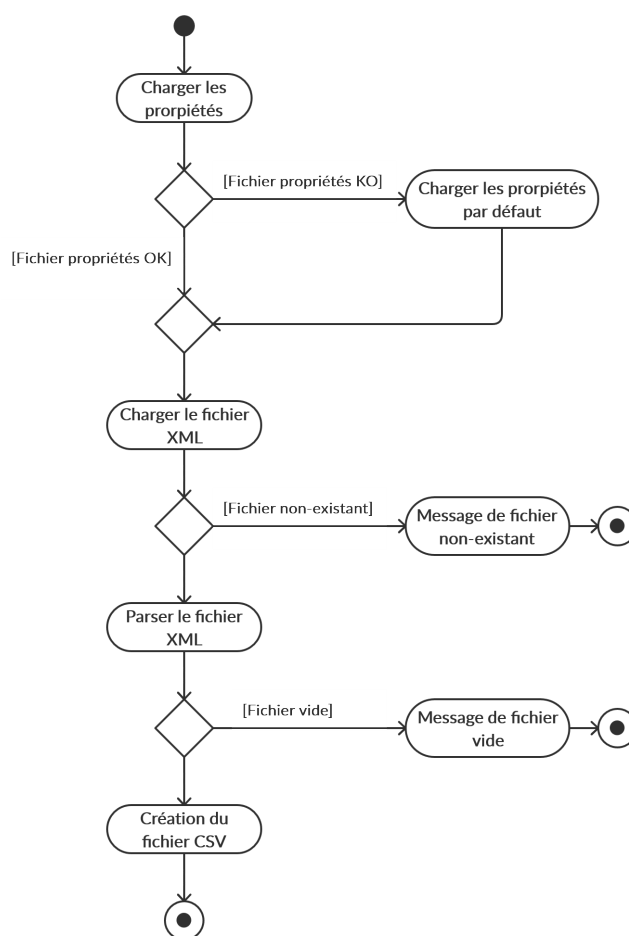


Figure 7 – Diagramme d'activité du parser XMLTV

Ce diagramme permet de suivre une feuille de route lors du développement et ainsi garantir une bonne interprétation du cahier des charges. Il permet aussi à toute personne arrivant sur le projet d'avoir une compréhension rapide de ce que doit effectuer le parser XMLTV.

En plus de ce diagramme d'activité, un diagramme de classe a été établi. Ce diagramme est illustré par la figure suivante.

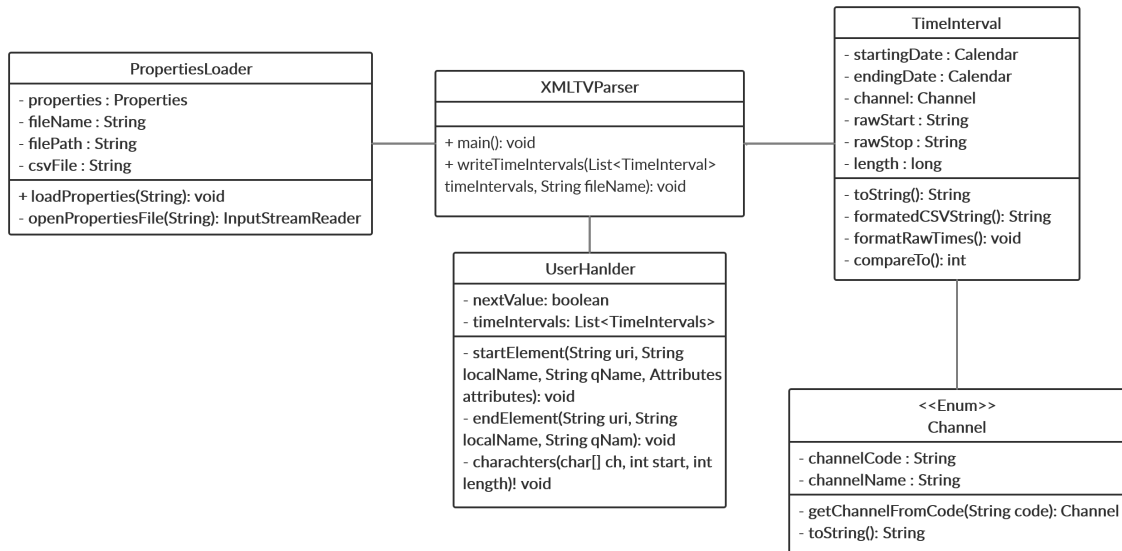


Figure 8 – Diagramme de classe du parser XMLTV

Cette découpe du programme permet une meilleure maintenabilité. Par exemple, si le format du fichier XMLTV venait à changer, seule la classe **UserHandler** serait à modifier – il s'agit de la classe permettant de sélectionner les éléments nous intéressant dans le fichier XML. Chaque classe fait l'objet d'une documentation permettant aux utilisateurs suivant d'avoir accès à toutes informations concernant le programme.

2 Capture des vidéos

L'outil de capture des vidéos sera le second à intervenir dans le projet. Il permet de capturer des vidéos sur les flux TV en fonction des intervalles de temps fournis par le parser XMLTV.

Cet outil de capture sera donc capable de lancer plusieurs threads pour pouvoir effectuer plusieurs captures en parallèle. Le diagramme d'activité suivant illustre ce parallélisme.

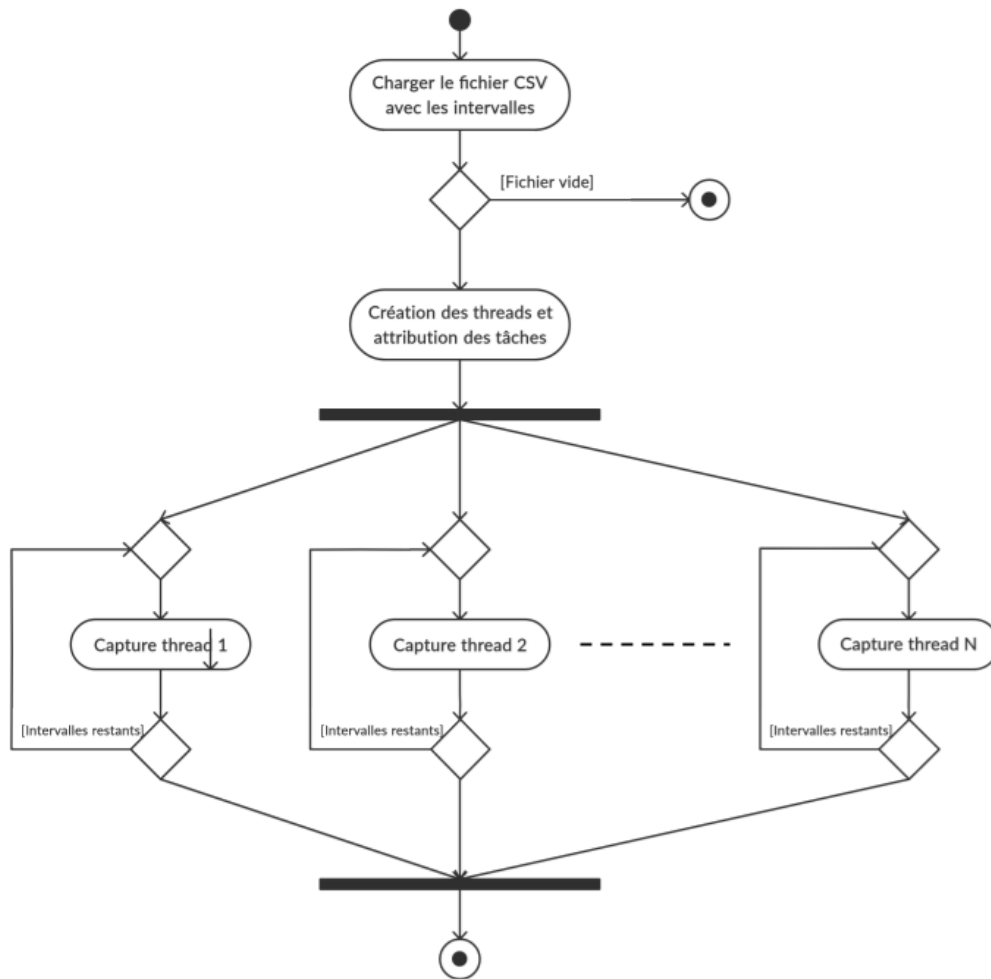


Figure 9 – Diagramme d'activité des captures vidéo

5

Mise en œuvre

1 Installation de la nouvelle station TV

Le projet prenait en compte la définition ainsi que l'installation de l'architecture de la station TV sur laquelle nous devions travailler. C'est pourquoi elle fait l'objet d'une partie dans ce rapport.

Pour pouvoir monter la station TV nous avons été trois étudiants travaillant en parallèle sur la station TV. Je tiens donc à remercier Amélie CARDOT, élève-ingénieur en informatique industrielle ainsi que Léo LEGRAND, élève-ingénieur en informatique pour notre collaboration.

1.1 Architecture interne finale

La station TV possédait un seul CPU au lancement du projet. Elle devait accueillir un ensemble de 6 cartes AVerMedia ainsi qu'un GPU NVidia RTX 2070 Super. Cependant, lors de la phase d'installation du matériel, nous avons constaté que certains bus PCIe sur lesquels le matériel devait être installé étaient dissociés du CPU.

Nous avons constaté qu'une deuxième emplacement pour CPU était disponible sur la station TV et permettrait d'accéder aux bus PCIe manquants. Il a donc fallu réorganiser l'architecture interne de la station TV.

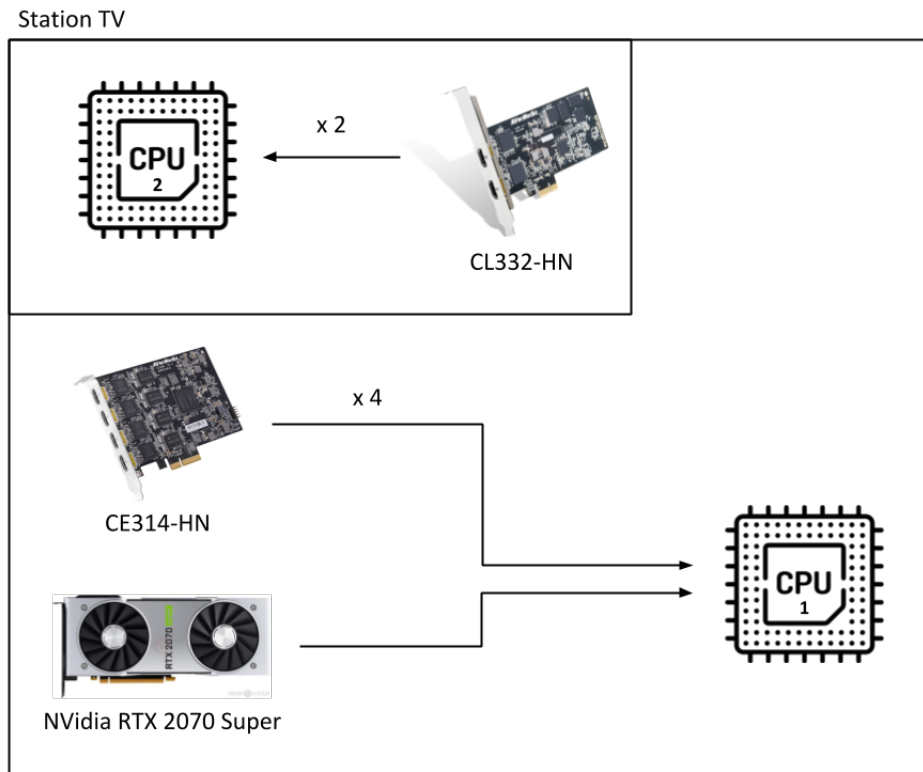


Figure 10 – Architecture interne de la station TV

1.2 Capacités de la station TV

Suite à l'installation nous avons testé le fonctionnement de tout le matériel. Les cartes AVerMedia sont toutes détectées et apportent bien les flux jusqu'à la station TV. Le GPU est lui aussi bien installé et fonctionne correctement. A noter que le GPU ne peut s'occuper que de traitements liés aux cartes CE314-HN puisqu'ils partagent le même CPU.

La station TV possède donc à présent deux cartes CL332-HN permettant de capturer 4 flux et d'effectuer du traitement hardware ainsi que 4 cartes CE314-HN permettant de capturer 16 flux pour un total de 20 flux.

1.3 Multitunner

Le multitunner a été aussi mis en place et est composé de 24 tunners TNT proposant donc 4 flux de plus que la capacité de la station TV. La figure suivante expose la solution retenue avec un étage contenant 4 tunners TNT. A noter que tous les étages contiennent chacun 4 tunners TNT.

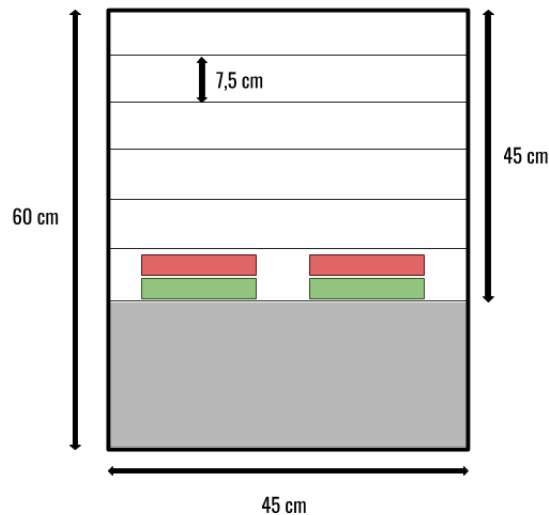


Figure 11 – Solution retenue pour le rack

2 Parser XMLTV

Cet outil doit parser un fichier au format XML. Il existe deux types de parser pour les fichiers XML : par arbre ou par événements.

Le parsing par arbre construit un arbre contenant les données du fichier. Il permet un parcours libre de l'arbre et une modification simple des données mais demande des ressources mémoires importantes et doit lire tout le document avant de pouvoir effectuer des actions dessus.

Le parsing par événements permet d'associer des méthodes à des événements pour traiter le document. Il est peu gourmand en ressources mémoires, est rapide, possède des principes simples à mettre en œuvre et permet de traiter seulement les informations utiles. Cependant, il traite les données uniquement de façon séquentielle et est légèrement plus difficile à implémenter car il faut sauvegarder les données lues avant de passer aux suivantes.

2.1 Parsing par événements

Notre choix s'est tourné vers un parsing par événements. En effet, plusieurs critères nous ont permis de prendre cette décision :

- Le document à traiter est un guide TV, par conséquent les programmes sont rangés par ordre de passage (séquentiel).
- Nous souhaitons uniquement extraire des informations sans modifier le document.
- Le volume du fichier à traiter est importante – plusieurs dizaines de milliers de lignes – et il est donc nécessaire de limiter l'utilisation des ressources mémoire.
- Pour les mêmes raisons que le point précédent, nous avons besoin d'un parser rapide.

2.2 Technologies

2.2.1 Langage de programmation

Le langage de programmation choisi est Java 11. Ce dernier possède de nombreux outils permettant le développement de notre parser XML. Il permet aussi de mettre en place différents outils pour faciliter la mise en place de l'environnement de développement.

L'environnement de développement choisi est Eclipse 2019-12.

2.2.2 Maven

Apach Maven est utilisé pour automatiser et gérer la production du projet. Cet outil permet de :

- Gérer facilement les dépendances du projet.
- Lancer plusieurs tâches lors de la phase de build du projet, notamment les tests unitaires.
- Générer l'application au format JAR automatiquement.
- Valider la qualité de la solution
- De nombreuses autres fonctionnalités.

2.2.3 Librairie : SAX Parser

Nous utilisons la librairie **SAX Parser** disponible en Java qui implémente un parser par événements. Ce dernier est implémenté de façon à extraire des informations sur les programmes TVs du fichier XMLTV. Les informations extraites sont :

- La chaîne TV sur laquelle est diffusée le programme.
- La date du début du programme.
- La date de fin du programme.

2.2.4 JUnit 5

Les tests unitaires du programme ont été mis en place grâce à JUnit 5.

2.3 Limites

Actuellement le programme n'extrait que les trois informations mentionnées dans la partie 1.2.3. Ces dernières sont utilisées pour déterminer les moments entre deux programmes TVs (5 minutes avant la fin d'un programme TV puis 5 minutes après le début du suivant) pour créer des intervalles de capture correspondant à la pause publicitaire.

2.4 Risques

Le format XMLTV possède une architecture établie. Cette dernière permet d'avoir des fichiers XMLTV standards provenant de sources hétérogènes et permet de parser le fichier facilement.

Cependant, nous ne sommes pas à l'abri que ce format change un jour auquel cas une évolution du programme devra être envisagée qui permettrait de sélectionner le format du fichier XMLTV à parser.

2.5 Reste à faire et évolution

Dans l'état actuel, le parser XMLTV est fonctionnel et permet d'extraire les intervalles de temps désirés.

Nous pourrions faire évoluer le parser XMLTV en permettant à l'utilisateur de choisir les programmes dont il souhaiterait capturer le générique.

3 Capture de vidéos

Cet outil doit capturer des vidéos depuis des flux TVs au travers de carte de capture de la marque AVerMedia.

3.1 Technologies

3.1.1 Langage de programmation

Le langage de programmation choisi est le C++. En effet, nous dépendons du matériel AVerMedia et pour pouvoir le manipuler informatiquement il est nécessaire d'utiliser le SDK fourni. Ce dernier est compatible avec quelques langages de programmation, à savoir : C++, WPF et VB.net. Ne connaissant que le langage de programmation C++, ce dernier a été le choix par défaut.

3.1.2 SDK : AVerMedia

Afin de manipuler les cartes de capture AVerMedia, le constructeur fournit un SDK. Ce dernier est disponible gratuitement sur le site officiel.

Une documentation est aussi fournie avec le SDK permettant de le manipuler facilement. Il permet notamment d'afficher les flux vidéo, d'enregistrer les flux, effectuer des captures d'images, et bien d'autres fonctionnalités.

Il est à noter que lors de l'utilisation d'une carte avec le SDK, seule une instance d'accès aux ressources est possible. Par conséquent, si deux programmes concurrents essaient d'accéder au flux d'une carte, le premier à faire la demande sera le seul à pouvoir exploiter le flux, le second se verra refuser l'accès.

3.2 Limites

Dans l'état actuel, seule une capture peut être effectuée par le programme.

3.3 Risques

Lors de la création des threads il est possible qu'un thread meurt prématurément. Cela aurait pour conséquence de bloquer le programme puisque ce dernier attend que les threads le préviennent qu'ils ont effectués toutes leurs captures.

Pour pallier à cela un système de *chien de garde* est envisageable. Le but de ce dernier est de surveiller le bon fonctionnement des threads, non pas au niveau de leurs résultats mais au niveau de leur cycle de vie. Ainsi, si un thread meurt prématurément, le chien de garde s'en aperçoit et relance le thread pour qu'il puisse continuer ses tâches.

3.4 Reste à faire et évolution

L'orchestration est encore à faire pour ce programme. Il s'agit de la création des threads ainsi que de la distribution des tâches aux threads.

Les tests sont aussi à produire pour cette partie puisqu'à cause du confinement lié au Covid-19 l'accès à la station TV est impossible.

Pour faire évoluer le système il est prévu d'associer un autre projet en cours de développement qui permettrait de changer informatiquement la chaîne TV associée à chaque entrée des cartes AVerMedia. Nous aurions ainsi une capture beaucoup plus dynamique sur l'ensemble des chaînes TV alors qu'aujourd'hui les chaînes sont statiques.

6

Bilan et conclusion

La première partie de ce projet a correctement été menée à terme. Les objectifs qui avaient été définis ont été remplis. Ces derniers étaient la rédaction de tous les documents permettant de lancer la seconde phase du projet : le développement.

La phase de développement a été interrompue par l'installation de la nouvelle station TV qui a pris du temps sur le cycle de développement. Le temps de développement a donc été réduit mais le développement du parser XMLTV a été mené à terme.

Le développement de la capture des vidéos n'a pas pu être mené à terme pour plusieurs raisons :

- Le temps de développement a été réduit par l'installation de la station TV.
- Impossibilité d'accès à la station TV.

1 Bilan sur la qualité

L'objectif était d'avoir une bonne qualité de solution qui a été atteinte grâce à :

- Une gestion de projet assez rigoureuse dans l'ensemble.
- La mise en place de nombreux outils de contrôles.
- La rédaction de nombreux documents tels que la documentation du code, la procédure technique d'installation, etc.

Cette qualité peut cependant toujours être améliorée en mettant par exemple en place :

- Une revue de code par un pair.
- Une prospection plus poussée par rapport aux technologies utilisées.
- Créer un ensemble plus hétérogène de cas de tests.

2 Bilan auto-critique sur la gestion de projet

La gestion de projet a été mise en place dès le début du projet. Elle a permis d'avoir une feuille de route très bien construite pour lancer le projet.

Malgré cela, quelques imprévus sont apparus et il a fallu revoir certains points, notamment au niveau de la planification. Cependant, la planification a été globalement bien établie et bien suivie.

Nous avons anticipé le délai que pouvait causer l'installation de la nouvelle station TV. Le développement du parser XMLTV a donc été lancé en amont afin de réduire le délai sur le cycle de développement puisqu'il ne dépend pas du matériel AVerMedia.

Ne pouvant pas anticiper les derniers événements – le confinement lié au Covid-19 – la planification n'a pas pu être corrigée en amont par rapport à la situation. Cela a bloqué le développement de la dernière partie du projet.



Bibliographie

Ressources pour PCle

- « *Structured Computer Organisation, 6th Edition* », Andrew S. Tanenbaum et Todd Austin, Pearson, ©2013

Annexes

A

Spécifications systèmes

1 Choix de l'architecture

Dans le cadre d'une augmentation en capacité de la station TV, l'architecture du système a dû être revue. L'architecture ciblée est une station TV composée d'au moins 16 tunners TNT permettant de capturer 16 flux vidéo en parallèle et en HD. Pour gérer ces 16 flux, il faut le nombre adapté de cartes de capture vidéo. L'installation des tunners TNT devra respecter un volume défini car la station TV possède une place restreinte.

C'est pourquoi une recherche de matériel a été dirigée dans le cadre de ce projet. Certaines contraintes ont limité le choix de ce matériel. Les composants du système nécessitant une recherche de matériel étaient :

- Les tunners TNT
- Les cartes de capture vidéo

1.1 Contraintes liées au choix des tunners TNT

Afin de limiter l'espace pris par l'ensemble des tunners TNT, un rack (modèle ***DIGITUS DN-19-12-U-EC-1***) a été choisi pour les installer. Les dimensions du rack sont les suivantes :

- Largeur : 45cm utiles
- Hauteur : 52,8cm utiles (possibilité d'avoir 63cm utiles si on utilise l'espace du socle)
- Profondeur : 20 cm utiles

L'espace entre les étages du rack est configurable. Il existe deux configurations possibles :

- Espace de 4,4cm entre étages
- Espace de 7,5cm entre étages

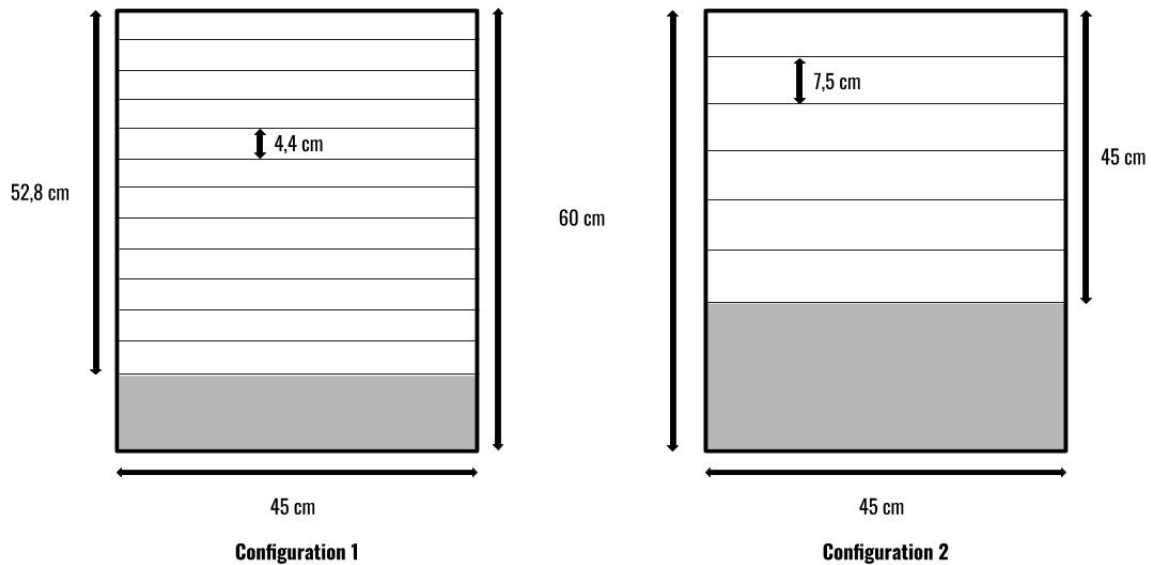


Figure 12 – Configurations du rack

En plus de la dimension des tunners TNT, d'autres contraintes entrent en compte :

- Le prix d'un tunner TNT doit être abordable
- La puissance consommée par les appareils doit être minimale car ils seront placés dans des endroits confinés
- Le type de sortie vidéo (HDMI) ainsi que la qualité de la vidéo (ultra HD)
- La possibilité de changer de chaîne manuellement
- La fonction « time shifting »
- L'ouverture du protocole infra-rouge

2 Solutions envisagées

2.1 Multitunner professionnel (non retenue)

La première solution envisagée est d'utiliser un multitunner professionnel possédant 24 sorties HDMI et permettant donc de traiter 24 flux TV en même temps. Pour cela une prospection a été faite sur les technologies existantes. Cette prospection a mis en avant un châssis **DHP400** de la marque **Dexing**. Pour capturer les flux vidéo des cartes de type DX924 et D714 ont été proposées.

Avantages de la solution

- La dimension du multitunner permet un système ultra-compact :
 - Largeur : 48,2 cm
 - Hauteur : 4,4 cm
 - Profondeur : 41 cm
- Les sorties sont en ultra HD
- Les chaînes peuvent être changées individuellement

Desavantages de la solution

- Le coût de la solution est beaucoup trop important par rapport au budget alloué

2.2 Augmentation du nombre de tunners TNT (retenue)

La seconde solution envisagée est une montée en volume de l'installation actuelle. Pour cela, le choix d'un modèle de tunner TNT a été fait pour optimiser l'installation. Ces tunners TNT seront installés dans le rack présenté plus haut.

Modèles de tunners TNT retenus

Deux modèles de tunners TNT respectaient les contraintes du système :

- **Astrell 011128.2**
- **TechniSat Terrabox T2**

Le modèle **Astrell 011128.2** étant celui déjà installé dans le système, il a été retenu pour l'installation. Une commande a donc été faite pour ce tunner TNT en 12 exemplaires.

3 Contraintes liées au choix des cartes de capture vidéo

La machine de la station TV possède une carte mère permettant d'accueillir jusqu'à 5 cartes en bus PCIe (x4). La station TV était conçue pour traiter au moins 16 flux, les cartes de capture vidéo devront donc être capables de capturer un certain nombre de flux.

Deux cartes de la marque AVerMedia (CE314-HN) sont déjà installées sur la station TV et proposent 4 entrées de flux vidéo chacune. Afin d'optimiser les performances de la station TV cependant une prospection a été menée pour trouver d'autres modèles potentiels.

Le choix des cartes se fera sur la base des critères suivant :

- La connexion à la station TV se fera par bus PCIe (x4)
- La carte devra prendre des entrées HDMI
- La résolution de la vidéo en sortie de la carte devra être en ultra HD
- L'ensemble des cartes devra traiter les 16 flux vidéo
- Les cartes devront être informatiquement manipulable de façon simple

3.1 Solution envisagée

Après prospection, il apparaît que le choix le plus préférable est d'équiper la station TV avec les cartes AVerMedia CE314-HN. Elles permettent de satisfaire les contraintes des 16 flux en ne prenant que 4 emplacements sur 5.

Avantages des cartes AVerMedia CE314-HN

- Matériel déjà connu car déjà installé
- AVerMedia fournit un SDK permettant de manipuler les cartes informatiquement

Desavantages des cartes AVerMedia CE314-HN

- L'enregistrement des captures vidéo se fait en software, c'est-à-dire qu'il requiert des ressources CPU

Le dernier emplacement de libre a été prévu pour installer une carte AVerMedia CL332-HN qui permet de faire de l'encodage matériel. Cela veut dire que les ressources CPU utilisées par cette carte sont nulles lors de la capture vidéo. Cela permet de ne pas surcharger le système en termes de ressources CPU. Cependant cette carte ne possède que deux entrée HDMI elle ne pourra traiter qu'au maximum 2 flux vidéo.

B

Spécifications fonctionnelles

1 Parsing du fichier XMLTV

1.1 Identification de la fonctionnalité de parsing du fichier XMLTV

Cette fonction est la première appelée lors du lancement du projet. Elle a pour but de générer un fichier contenant des intervalles de temps qui dirigeront le module enregistrant les flux vidéo des chaînes TV.

1.2 Description de la fonctionnalité de parsing du fichier XMLTV

Cette fonctionnalité permettra à l'administrateur de générer un fichier contenant des intervalles de temps extraits du fichier XMLTV. Le but est de générer pour chaque chaîne TV une liste d'intervalles qui définiront les moments d'enregistrement dans la journée. Le parsing du fichier sera effectué en un seul traitement et sera appelé manuellement par l'administrateur lorsqu'il estimera qu'une nouvelle génération d'intervalles est nécessaire.

Préconditions

- Le fichier XMLTV devra avoir été téléchargé dans le dossier cible par l'administrateur
- Le système devra être à l'heure courante pour respecter les horaires de la télévision

Post-conditions

- Génération d'un fichier CSV contenant les intervalles de temps exploitables par le module d'enregistrement
- Les intervalles de temps commenceront après l'heure du système auquel le fichier a été traité
- Chaque intervalle sera associé à une chaîne TV

2 Enregistrement des génériques TV

2.1 Identification de la fonctionnalité d'enregistrement

Cette fonction sera appelée après le parsing du fichier XMLTV. Son objectif est de capturer des vidéo à partir de flux TV fournis par les cartes AVerMedia.

2.2 Description de la fonctionnalité d'enregistrement

Cette fonctionnalité permettra à partir d'un fichier contenant des intervalles de temps de capturer des flux vidéo à partir des cartes AVerMedia. Il utilisera le SDK fourni par AVerMedia pour enregistrer ces flux vidéo. Les enregistrements devront pouvoir être fait en parallèle sur tous les flux vidéo mais avec une seule capture par sortie vidéo. De plus, les chaînes TV diffusent du contenu en continu ce qui implique que ce module doit pouvoir lancer un enregistrement à n'importe quel moment.

Préconditions

- Le fichier contenant les intervalles de temps se trouve dans le dossier cible
- La connexion avec les cartes AVerMedia est disponible
- Il reste suffisamment d'espace sur le disque dur

Post-conditions

- Création d'une vidéo au format MP4 et en ultra HD
- La vidéo doit avoir été capturée durant l'intervalle de temps défini
- Le nom de la vidéo doit être de la forme "[NOM_CHAINE_TV]_[TIMESTAMP].mp4"
- La connexion avec la carte AVerMedia doit être libérée à la fin de la capture
- Génération d'un rapport sur la capture d'un enregistrement

3 Ordonnancement de l'enregistrement

Cette fonctionnalité permettra d'optimiser l'enregistrement des génériques TV. Certains génériques sont récurrents et par conséquent sont moins importants à capturer à court terme. Il est donc important de gérer la priorité des génériques à capturer pour éviter d'en rater un qui serait peu diffusé. Pour cela il est possible d'ordonnancer les intervalles de temps lors du parsing du fichier XMLTV afin de mieux couvrir l'ensemble des génériques à capturer.

Aucun algorithme d'ordonnancement n'a été défini pour l'instant, il le sera lors de la phase développement.

C

Spécifications non fonctionnelles

1 Contraintes de développement et conception

Le SDK fournit par AVerMedia met à disposition des bibliothèques au format .dll ce qui impose un système d'exploitation Windows. La bibliothèque nous intéressant est AVerCapAPI qui permet de manipuler les flux vidéo des cartes en visualisation et en enregistrement.

L'API AVerMedia supporte les langages de programmation C++, WPF et VB.net. Le langage retenu est le C++ et l'IDE utilisé sera VisualStudio 2019.

Les intervalles de temps pour les enregistrements se répartissent sur la journée entière. Il faut donc que le module d'enregistrement soit capable d'enregistrer des vidéos à tout moment de la journée. Cela implique qu'un mécanisme soit mis en place pour satisfaire cette contrainte.

2 Contraintes de fonctionnement et d'exploitation

Concurrence sur les cartes AVerMedia

Lors de l'exploitation des flux vidéo des cartes AVerMedia, un problème de concurrence apparaît. En effet, bien que chaque carte mette à disposition un certain nombre de flux vidéo, chaque flux ne peut être exploité que par une seule application au travers de l'API. Par exemple, si une application affiche un des flux d'une carte vidéo à l'écran et qu'une autre application souhaite enregistrer ce même flux, cette dernière se verra refuser l'accès au flux étant donné qu'il est déjà utilisé.

Cette contrainte pose des problèmes lors de l'enregistrement des flux vidéo puisqu'il faut s'assurer qu'aucune autre application n'exploite un flux vidéo avant de lancer l'enregistrement, autrement celui-ci ne se fera pas. De plus, puisque les enregistrements sont planifiés, il est important de pouvoir accéder au flux à tout instant car il est impossible de reporter l'enregistrement.

Dépendance aux chaînes TV

La diffusion du contenu TV est indépendant de notre système, ce qui impose une contrainte. Notre système doit donc être capable de s'adapter à cette diffusion. Cela impose d'avoir un système capable de fonctionner en temps réel. De plus si un enregistrement vidéo échoue il

est impossible de relancer la capture. Cependant comme certains génériques TV sont diffusés fréquemment sur les chaînes, il est possible de définir une marge d'erreur.

2.1 Performances

Les chaînes TV diffusent des vidéos en continu, ce qui impose une contrainte de performance sur l'enregistrement en temps réel. C'est pourquoi le module d'enregistrement doit être capable à tout instant de pouvoir être lancé.

2.2 Capacités

2.2.1 Disque dur

L'enregistrement des vidéos occupe une place mémoire importante. Les vidéos étant des fichiers généralement assez volumineux il faudra s'assurer que l'appareil de stockage sur lequel seront enregistrées les vidéos soit capable de contenir un grand nombre de vidéos.

2.2.2 Processeur

L'enregistrement « software » des vidéos par les cartes AVerMedia utilise des ressources CPU. Le système devra être capable de manipuler 16 flux en même temps. Il faut donc que le coût en ressources de ces manipulations ne dépasse pas les capacités du processeur ou une certaine limite imposée.

2.2.3 Contrôlabilité

Lors de l'enregistrement d'une vidéo plusieurs informations peuvent nous intéresser :

- La chaîne TV enregistrée
- Le nom de la vidéo générée
- La date de début de l'enregistrement prévue
- La date de début de l'enregistrement réelle
- La date de fin de l'enregistrement prévue
- La date de fin de l'enregistrement réelle

Ces informations seront consignées dans un rapport associé à chaque chaîne TV et mis à jour par le module d'enregistrement. Ces rapports pourront être exploités pour vérifier si les enregistrements se sont déroulés comme prévu.

2.2.4 Sécurité

L'utilisateur lançant les modules du système devra posséder un compte administrateur sur la machine. En effet certaines fonctionnalités liées aux cartes AVerMedia requièrent un accès administrateur.

2.2.5 Intégrité

La télévision fournit du contenu en continu ce qui imposerait d'avoir un système robuste capable d'enregistrer à tout moment. Cependant, ce projet étant un projet exploratoire, nous devons répondre à une contrainte temporelle moins importante. Le système devra donc fonctionner sur des durées suffisantes pour obtenir des résultats pertinents sans pour autant devoir fonctionner en continu. C'est pourquoi il a été convenu que le système devra fonctionner sur environ une journée au minimum pour pouvoir avoir ces résultats.

2.3 Maintenance et évolution du système

Pour améliorer les performances des captures de génériques, il est possible d'ordonnancer les captures des génériques TV lors du parsing du fichier XMLTV. L'objectif serait d'appliquer un algorithme d'ordonnancement des tâches lors du parsing XMLTV. Cependant, les génériques TV sont répétés sur les chaînes TV donc des règles de priorisation devront être aussi ajoutées à cet ordonnancement. L'objectif est à long terme est d'alléger la charge CPU du module d'enregistrement des génériques TV en diminuant le nombre d'enregistrements à effectuer.

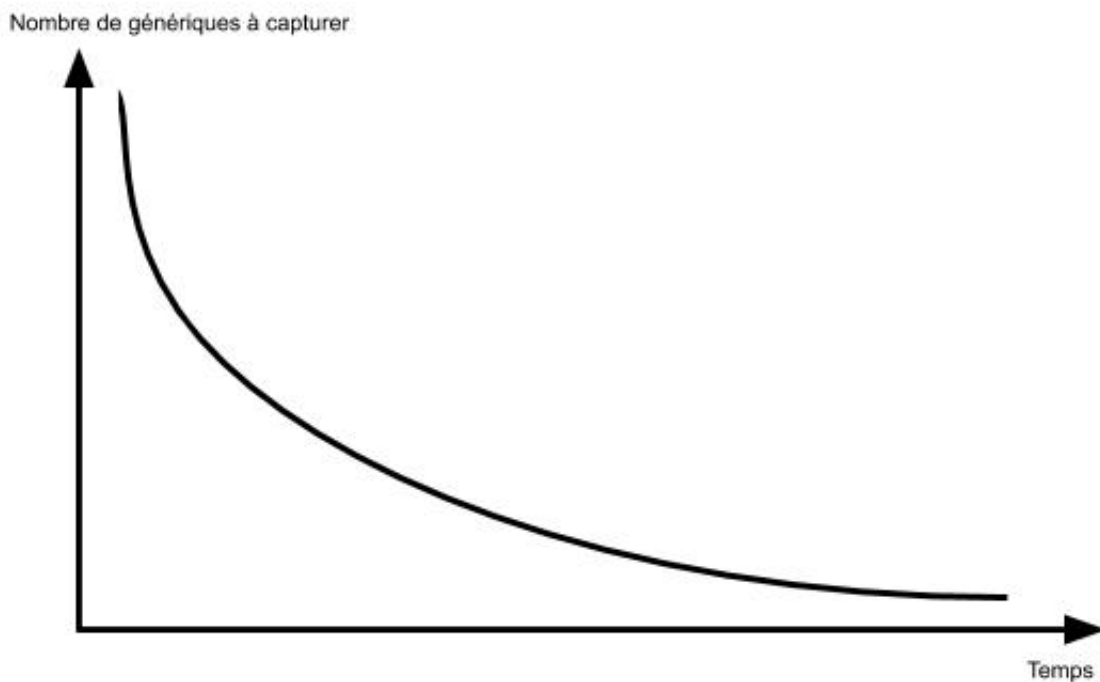


Figure 13 – Graphe du nombre de génériques à capturer par rapport au temps

D

Gestion de projet

La planification de ce projet a été faite avec l'outil en ligne *Zoho Projects*¹. Cet outil a été choisi car il ne nécessite pas l'installation de programme sur une machine ce qui permet un partage plus simple et une compatibilité avec tous les systèmes d'exploitation. Il permet de définir des tâches à effectuer ainsi que leur durée, les dates de début et de fin, les personnes devant les effectuer, etc. Une fois les tâches créées, elles seront utilisées pour créer un diagramme de Gantt consultable directement dans l'outil. Pendant la réalisation des tâches il est possible de renseigner l'avancement de chaque tâche et ainsi avoir un suivi plus précis du projet.

1 Découpage des tâches

La première étape a été de créer des tâches globales. Elles ont pour objectif d'organiser les différentes actions à effectuer. Les tâches globales sont :

- Rédaction du cahier des charges
- Rédaction du cahier de spécifications
- Développement du module de parsing du fichier XMLTV
- Développement du module d'enregistrement
- Ordonnancement des enregistrements

Ces tâches sont elles-mêmes découpées en plusieurs sous-tâches pour apporter plus précisions quant aux tâches à effectuer. Nous avons donc deux niveaux de visibilité : un niveau général avec une granularité assez importante et un niveau plus détaillé avec une granularité assez fine.

2 Diagramme de Gantt

Voici le premier diagramme de Gantt pour le projet :

1. [Site de Zoho Projects](#)

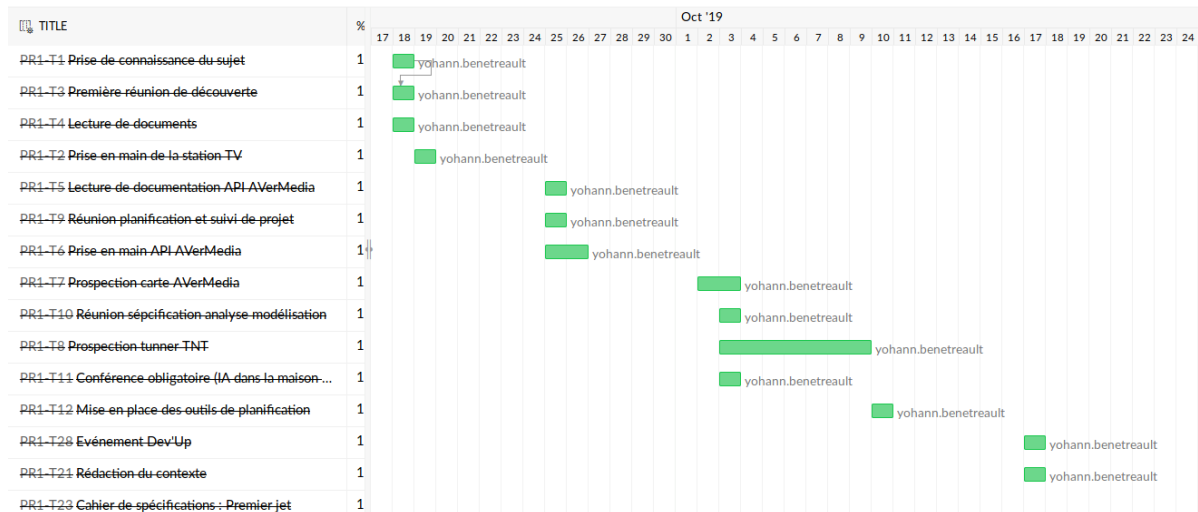


Figure 14 – Diagramme de Gantt - Première partie

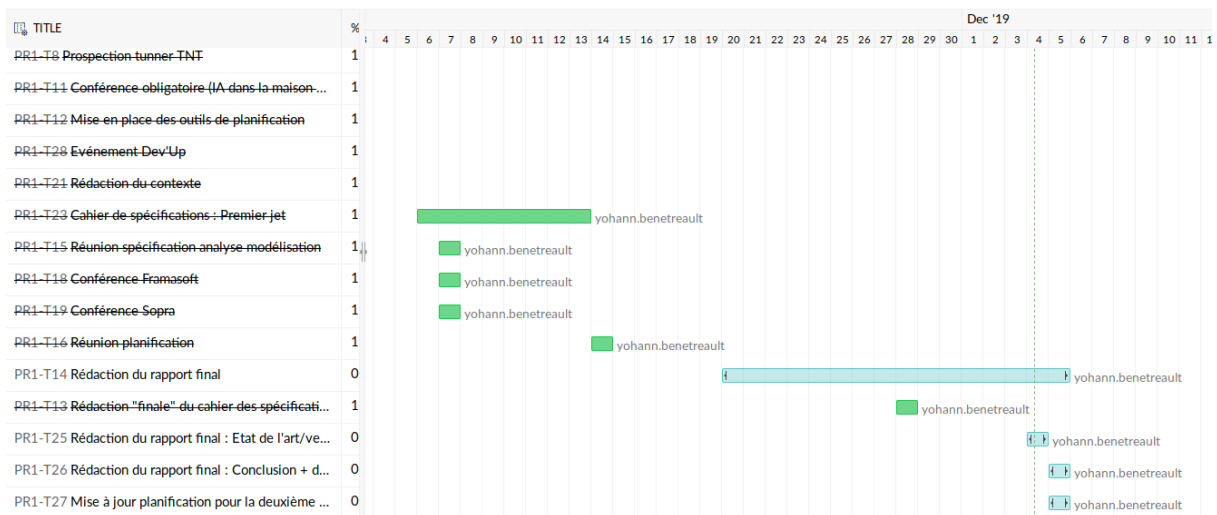


Figure 15 – Diagramme de Gantt - Deuxième partie

3 Mise à jour de la planification

La planification présentée ci-dessus est pour la première phase du projet, à savoir la rédaction des documents. La seconde phase qui consistera au développement fera elle aussi l'objet d'une planification. Cette planification sera faite lorsque le cahier des spécifications sera validé et lorsque la priorité sera définie pour chaque tâche. Les tâches *développement du module de parsing du fichier XMLTV*, *développement du module d'enregistrement*, *développement du module de visualisation des flux TV* seront précisés lors de cette deuxième planification.

4 Planification finale

Au début de la deuxième phase du projet, une plainification a été mise en place. La tâche concernant le développement de la visualisation des flux TV a été annulée puisque le fournisseur des cartes de capture fournit aussi un logiciel permettant de visualiser l'ensemble des flux TVs.

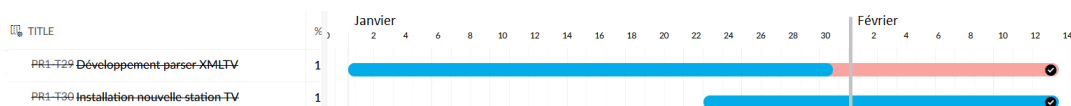
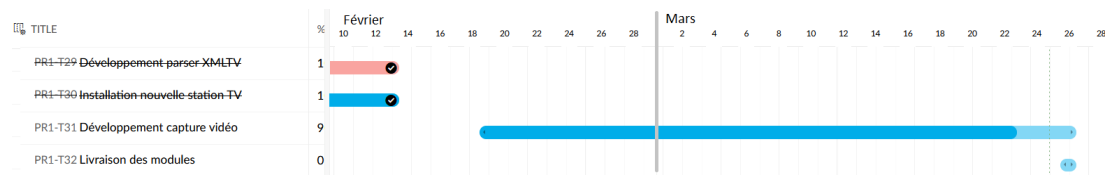


Figure 16 – Diagramme de Gantt - Première partie**Figure 17 – Diagramme de Gantt - Deuxième partie**

E

Documents d'installation



ÉCOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS
 Spécialité Informatique
 64 av. Jean Portalis
 37200 TOURS, FRANCE
 Tél +33 (0)2 47 36 14 31
www.polytech.univ-tours.fr

PROCEDURE TECHNIQUE D'INSTALLATION			
Projet :	Capture de génériques TV sur station		
Emetteur :	Yohann BENETREULT		
Date d'émission :	18/03/2020		
Validation			
Nom	Date	Valide (O/N)	Commentaires
Y. BENETREULT	25/03/2020	O	Reste à faire : installation du module de captures automatiques.
Historique des modifications			
Version	Date	Description de la modification	
0.0	18/03/2020	Version initiale	
1.0	25/03/2020	Rédaction de la procédure d'installation du parseur XMLTV	

TABLE DES MATIERES

Procédure technique d'installation.....	5
1. Introduction	5
2. Récupération du fichier XMLTV	5
3. Parseur XMLTV	6
4. Enregistrement automatique des chaînes TV	6

Procédure technique d'installation

PROCEDURE TECHNIQUE D'INSTALLATION

1. Introduction

Ce document a pour but d'expliquer l'installation des différents modules fournis.

2. Récupération du fichier XMLTV

Cette partie expliquera comment récupérer le fichier XMLTV contenant les programmes TV.

Le site fournissant le fichier XMLTV étant désormais fermé, une solution a dû être trouvée pour récupérer le fichier XMLTV. La solution trouvée à ce jour est un script PERL.

2.1. Récupération du script PERL

Le script PERL se trouve à l'adresse suivante : [GitHub de zubrick](#).

Son but est de récupérer le programme TV directement par le biais de Télérama.

2.2. Première configuration du script PERL

Le script PERL doit être configuré de la manière suivante (sur distribution UNIX) :

- Ouvrir un terminal de commande
- Se placer dans le dossier où se trouve le script PERL
 - `cd chemin/vers/script/`
- Exécuter la commande (s'assurer que PERL est installé sur la machine) :
 - `perl tv_grab_fr_telerama --configure`
- Le script vous demandera de sélectionner les chaînes TV souhaitées

2.3. Exécution du script

Pour lancer le script et ainsi récupérer le programme TV, lancez la commande suivante dans le dossier où se trouve le script PERL :

```
◦ perl tv_grab_fr_telerama > xmltv.xml
```

Le script se chargera de récupérer les programmes et de les écrire dans un fichier XML.

2.4. Modification des chaînes à récupérer

Il est possible de reconfigurer le script PERL à tout moment. Pour cela il y a plusieurs solutions :

- Relancer la commande : `perl tv_grab_fr_telerama --configure`
 - Cette solution est un peu longue et demandera une validation pour chaque chaîne

Ou :

- Ouvrir le fichier de configuration avec un éditeur de texte :
 - Le fichier de configuration se trouve par défaut dans `~/ .xmltv/`
 - Se déplacer dans le dossier : `cd ~/ .xmltv/`
 - Ouvrir le fichier `tv_grab_fr_telerama.conf`
 - `gedit tv_grab_fr_telerama.conf`

Procédure technique d'installation

- Il suffit alors de mettre ou d'enlever un # au début de chaque ligne
 - # Signifie que la ligne est commentée et donc ignorée par le script PERL
 - Possibilité de faire une recherche par nom de chaîne (CTRL + F)

3. Parseur XMLTV

Le parseur XMLTV se charge de lire le fichier XMLTV fourni et d'en extraire des intervalles de temps qui seront utilisés pour lancer les captures automatiques.

L'architecture du projet est la suivante :

- Un dossier principal contenant le `XMLTV_parser.jar` et le dossier `resources/`
- Dans le dossier `resources` se trouvent :
 - Un fichier `parser.properties` contenant les différentes propriétés garantissant le bon fonctionnement du programme
 - Un fichier `xm1tv.dtd` nécessaire au parsing du fichier XMLTV.

3.1. Lancement du parseur

Le parseur se lance au travers d'un terminal de commande de la manière suivante :

- `java -jar XMLTV_parser.jar [chemin_vers_le_fichier_xm1tv]`

Si le chemin vers le fichier XMLTV n'est pas renseigné, le programme va chercher le chemin qui sera indiqué dans le fichier `./resources/parser.properties`. Si ce fichier n'existe pas alors le programme alertera l'utilisateur avec le message « No such file or directory ».

Le chemin vers le fichier XMLTV peut être donné par l'utilisateur si ce dernier l'a placé autre part que dans le dossier `resources/` du projet.

3.2. Fichier CSV créé par le programme

Le programme fourni en sortie un fichier `time_intervals.csv` qui sera placé par défaut dans le dossier `resources/`. Cependant, il peut être enregistré ailleurs si l'utilisateur le souhaite en remplaçant le chemin de sortie dans le fichier `resources/parser.properties`.

4. Enregistrement automatique des chaînes TV

Cette partie aura pour but de détailler l'installation du module de capture des génériques TVs en exploitant le fichier `time_intervals.csv` fourni par le parseur XMLTV. Cependant, pour des raisons de confinement, ce module n'est pas encore terminé et sera mise à jour dès que possible.

F

Version des logiciels

- Java : OpenJDK 11
- Eclipse : 2019-12
- Apache Maven : 3.6.0
- JUnit 5 : 5.5.1
- Visual Studio : Community 2019
- AVerMedia : SDK v1.2.1.9

G

Documents d'utilisation



ÉCOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS
 Spécialité Informatique
 64 av. Jean Portalis
 37200 TOURS, FRANCE
 Tél +33 (0)2 47 36 14 31
www.polytech.univ-tours.fr

GUIDE DU DEVELOPPEUR			
Projet :	Capture de génériques TV sur station		
Emetteur :	Yohann BENETREAU		
Date d'émission :	18/03/2020		
Validation			
Nom	Date	Valide (O/N)	Commentaires
Y. BENETREAU	19/03/2020	O	Reste à faire : rédiger la partie module de capture vidéo.
Historique des modifications			
Version	Date	Description de la modification	
0.0	18/03/2020	Version initiale	
1.0	19/03/2020	Rédaction du guide pour le parseur XMLTV	

TABLE DES MATIERES

Procédure technique d'installation.....	5
1. Introduction	5
2. Parseur XMLTV	5
3. Enregistrement automatique des chaînes TV	8

Procédure technique d'installation

PROCEDURE TECHNIQUE D'INSTALLATION

1. Introduction

Ce document a pour but d'expliquer comment a été développé le projet.

2. Parseur XMLTV

Le parseur XMLTV se charge de lire le fichier XMLTV fourni et d'en extraire des intervalles de temps qui seront utilisés pour lancer les captures automatiques.

Le projet se trouve au lien suivant : [GitLab](#)

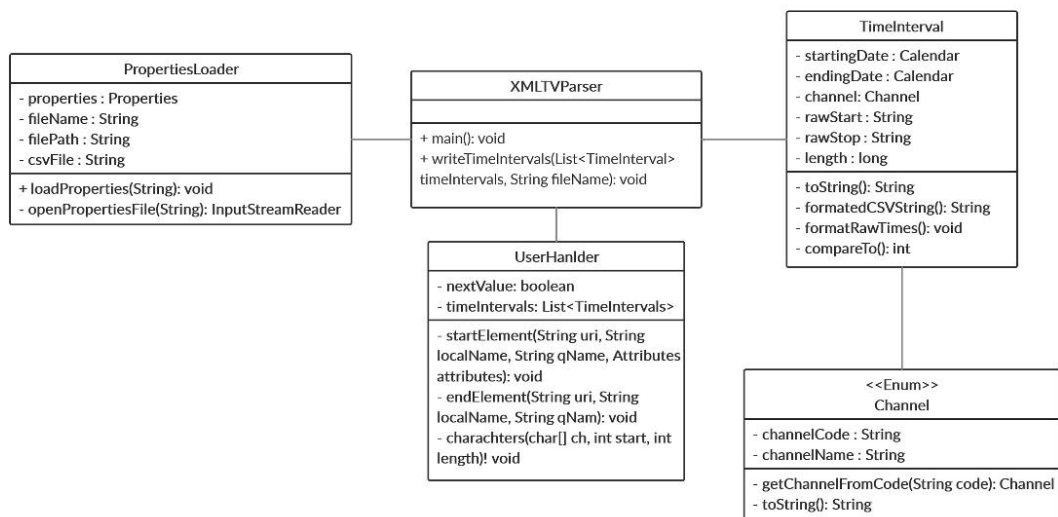
Le parseur est développé en Java 11 sous Eclipse 2019-12 et utilise Maven pour l'automatisation des tests et la gestion des dépendances. Les tests unitaires sont effectués grâce à JUnit 5.

La documentation est générée par JavaDoc et est disponible avec le projet. Elle documente les classes ainsi que leur méthode pour mieux appréhender leur utilisation.

Le projet met aussi en place de l'intégration continue avec GitLab CI et lance automatiquement un build puis les tests unitaires à chaque push. Il est aussi possible de lancer le build et les tests unitaires directement dans le projet.

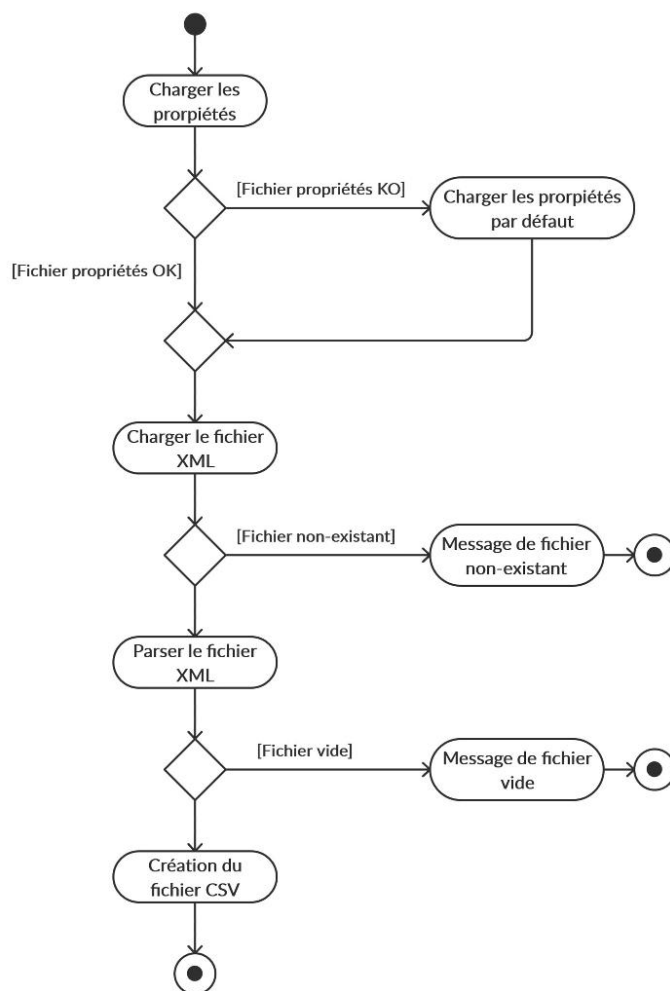
Pour utiliser cette application, veuillez vous référer à la procédure technique d'installation.

2.1. Diagramme de classe



Procédure technique d'installation

2.2. Diagramme d'activité



Procédure technique d'installation

1.1. XMLTVParser

Cette classe est le cœur de l'application et permet de lancer la routine de parsing du fichier XMLTV. Elle effectue les actions présentées dans le diagramme de séquence.

2.3. PropertiesLoader

Cette classe permet de charger les propriétés du fichier `parser.properties` présent dans le dossier `ressources/` et de fournir un objet `Properties` au programme.

Pour instancier un objet `PropertiesLoader`, il suffit d'appeler le constructeur et de lui passer en paramètre le chemin vers le fichier contenant les propriétés. Le constructeur se charge d'ouvrir le fichier, d'importer son contenu et de mettre à jour les champs de l'objet qui nous intéresse.

2.4. UserHandler

Cette classe hérite de la classe `DefaultHandler` qui permet de parser des fichiers au format XML. Il utilise un SAX Parser qui est un parser XML séquentiel permettant de récupérer les informations que l'utilisateur désire.

Si l'architecture du fichier XMLTV est amenée à changer dans le temps, il suffira de modifier les actions qu'effectue cette classe `UserHandler` afin de récupérer les informations désirées.

Pour plus de précision, veuillez vous référer à la JavaDoc.

2.5. TimeInterval

Cette classe représente un intervalle de temps qui sera utilisé par le programme d'enregistrement automatique des vidéos.

Elle contient tous les éléments nécessaires à la capture, à savoir :

- La chaîne TV
- La date de début d'enregistrement (au format Calendar et en brut)
- La date de fin d'enregistrement (au format Calendar et en brut)

Lorsque le parseur s'exécute, il construit une liste de `TimeInterval` qui sont ensuite triés par ordre croissant de date de début puis formatés pour être écrit dans un fichier CSV. Le formatage se fait grâce à la méthode `formattedCSVString()` qui renvoie une chaîne de caractères.

Les chaînes TV sont représentées par l'énumération `Channel`.

2.6. Channel

Cette énumération représente l'ensemble des chaînes de la TNT avec leur code les représentant dans le fichier XMLTV. En effet, dans ce fichier les chaînes sont définies par un code et il faut donc pouvoir faire le lien entre ce code et la chaîne.

Cette énumération fournit une méthode permettant de récupérer le nom de la chaîne grâce à son code : `getChannelFromCode(String channelCode)`.

Procédure technique d'installation

3. Enregistrement automatique des chaînes TV

Cette partie aura pour but de détailler le module de capture de générique TV. Cependant, suite aux événements récents et à l'impossibilité d'accéder au matériel permettant d'effectuer ces captures ce module est en suspend et sera mise à jour plus tard.

H

Dossiers de tests

Il est à noter qu'au vu des événements actuels, à savoir le confinement lié au Covid-19, seul l'outil de parsing XMLTV a été testé. En effet, le module de capture des génériques TV dépend du matériel mis à disposition par Polytech et qui est présent dans ses locaux. Par conséquent, ce matériel est inaccessible pour effectuer des tests.

1 Tests fonctionnels

Un cahier de recette a été rédigé pour les tests fonctionnels.

Numéro	Module	Action	Attendu	Résultat	Notes
1	Script Perl	Téléchargement du fichier XMLTV	Fichier au format XML non vide	OK	
2	Parser XMLTV	Parsing d'un fichier non existant	Message d'avertissement	OK	"No such file or directory"
3	Parser XMLTV	Parsing d'un fichier XMLTV vide	Message d'avertissement	OK	"Premature end of file"
4	Parser XMLTV	Parsing d'un fichier XMLTV	Programmes parsés et création d'un fichier CSV	OK	
5	Parser XMLTV	Parsing d'un fichier XMLTV passé en paramètre	Programmes parsés et création d'un fichier CSV	OK	
6	Enregistreur vidéo	Lancement d'une capture vidéo	Vidéo au format MP4	A faire	"Impossible d'accéder à la station TV"
7	Enregistreur vidéo	Lancement de l'automatisation des captures avec fichier CSV vide	Fin du programme avec message (aucune capture effectuée)	A faire	"Impossible d'accéder à la station TV"
8	Enregistreur vidéo	Lancement de l'automatisation des captures avec fichier CSV	Fin du programme avec message (x captures effectuées)	A faire	"Impossible d'accéder à la station TV"
9	Enregistreur vidéo	Lancement de l'automatisation des captures avec fichier CSV sur plusieurs chaînes TV	Fin du programme avec message (x captures effectuées)	A Faire	"Impossible d'accéder à la station TV"

Figure 18 – Cahier de recette

2 Tests unitaires

Un jeu de données a été créé pour le parser XMLTV. Il s'agit d'un fichier XMLTV contenant une liste de programmes valides. Les tests unitaires sont automatisés grâce à Maven. Ces derniers sont lancés dès le build du projet ainsi que lors de la sauvegarde sur GitLab grâce à un outil d'intégration continue que propose la plateforme.

Cet outil d'intégration continue nous permet de savoir si lors de la sauvegarde le projet est toujours valide.

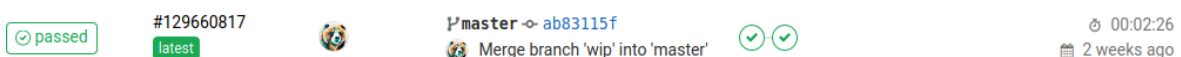


Figure 19 – Exemple de la validité du build du projet

Il nous fournit aussi un rapport plus détaillé sur l'ensemble du build ainsi que sur le lancement des tests unitaires.

```

580 [INFO] -----
581 [INFO] T E S T S
582 [INFO] -----
583 [INFO] Running fr.polytech.ybene.prd.test.TestUserHandler
584 [INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.809 s - in fr.polytech.ybene.prd.test.TestUserH
andler
585 [INFO] Running fr.polytech.ybene.prd.test.TestTimeIntervals
586 [INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.012 s - in fr.polytech.ybene.prd.test.TestTimeI
ntervals
587 [INFO] Running fr.polytech.ybene.prd.test.TestChannel
588 [INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.012 s - in fr.polytech.ybene.prd.test.TestChann
el
589 [INFO] Running fr.polytech.ybene.prd.test.TestPropertiesLoader
590 15:37:39.239 [main] ERROR fr.polytech.ybene.prd.tools.PropertiesLoader - (No such file or directory)
591 [INFO] Tests run: 9, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.061 s - in fr.polytech.ybene.prd.test.TestPrope
rtiesLoader
592 [INFO]
593 [INFO] Results:
594 [INFO]
595 [INFO] Tests run: 15, Failures: 0, Errors: 0, Skipped: 0

```

Figure 20 – *Exemple de la validité des tests unitaires*

Ici par exemple, nous constatons un succès sur l'ensemble des tests unitaires. Il nous indique aussi tous les tests qui ont été exécutés et ainsi d'avoir un aperçu de la couverture de test.

Projet Recherche & Développement

Capture de génériques TV

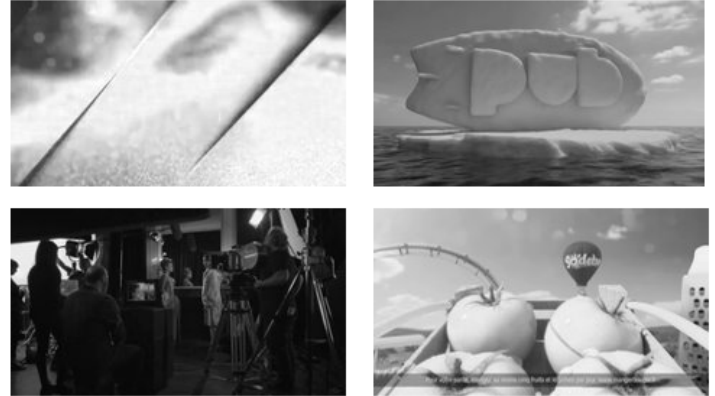
Yohann Benétreault

Encadrement : Mathieu Delalandre

Objectifs

Mise en œuvre d'une application d'enregistrement des génériques :

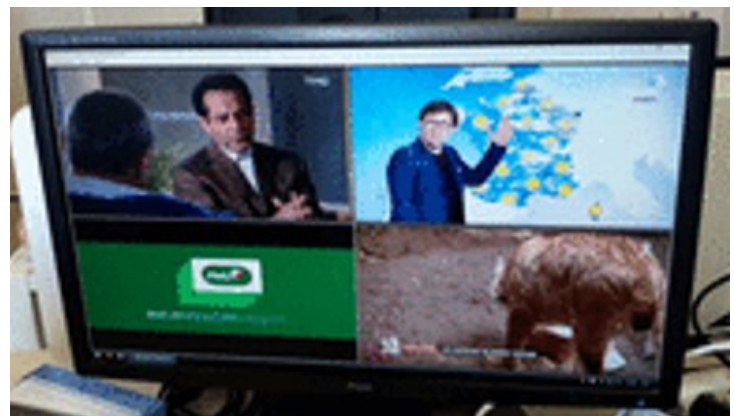
- Définir une architecture du système.
- Définir comment capturer intelligemment les génériques.
- Automatiser la capture des génériques TVs.



Exemple de génériques TVs

Mise en œuvre

- Installation de la station TV.
- Parser le programme TV pour en extraire les informations.
- Construire une liste d'intervalles de temps pour les captures.
- Automatisation de la capture des génériques TVs.



Station TV avec 4 flux vidéo

Résultats attendus

- Possibilité de manipuler 20 flux simultanément.
- Générer des intervalles de temps pour la capture.
- Optimiser l'utilisation des cartes de capture.
- Des vidéos contenant les génériques TVs.



Résultat souhaité avec 20 flux

Capture de génériques TV

Yohann Benétreault

Encadrement : Mathieu Delalandre

Objectifs

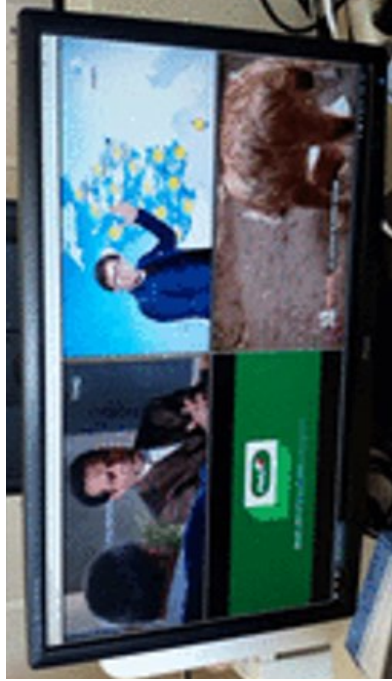
- Mise en œuvre d’une application d’enregistrement des génériques :
- Définir une architecture du système.
- Définir comment capturer intelligemment les génériques.
- Automatiser la capture des génériques TVs.

Mise en œuvre

- Installation de la station TV.
- Parser le programme TV pour extraire les informations.
- Construire une liste d’intervalles de temps pour les captures.
- Automatisation de la capture des génériques TVs.



Exemple de génériques TVs



Station TV avec 4 flux vidéo

Résultats attendus

- Possibilité de manipuler 20 flux simultanément.
- Générer des intervalles de temps pour la capture.
- Optimiser l’utilisation des cartes de capture.
- Des vidéos contenant les génériques TVs.

TF1	2	3	arte	6
5	C8	W9	TMC	TFX
N2j	LCP PUBLIC SENAT	4	BFM TV.	C NEWS
CSTAR	gulli	Ô	L'EQUIPE	6ter

Résultat souhaité avec 20 flux

Capture de génériques TV

Résumé

Ce document concerne le projet de recherche et développement de capture de génériques TV depuis la station TV installée dans la salle des doctorants de Polytech Tours. L'objectif est d'automatiser la capture des flux des chaînes TV afin d'extraire des génériques TV.

Mots-clés

génériques TV, station TV

Abstract

This document is related to the research and development project about the recording of TV jingles on the TV station at Polytech Tours. The purpose is to automate the recording of TV channels to extract TV jingles from them.

Keywords

TV jingles, TV station