

G R O U P E

ESEO

Angers
Paris
Shanghai

www.eseo.fr

2016 / 2017

NOM MENARD

PRENOM Alexandre

Stage Sc. et Technique I2 (S7)

Rapport de stage / emploi

>ENTREPRISE : *ILTR*

>DATES : *Du 04/09/17 au 01/12/17*

>SUJET MISSIONS : Au sein de la société ILTR, j'ai mis en place un système automatisé de reprise de données pour les nouvelles Bases de données des clients d'ILTR.

>TUTEUR ENTREPRISE : *M Karim JLASSI – Directeur technique*



Remerciements

Je souhaite tout d'abord remercier l'entreprise dans laquelle j'ai effectué mon stage scientifique-technique. J'ai pu acquérir pendant cette période de 13 semaines de nombreuses compétences, notamment dans le domaine de la programmation. J'ai aussi pu mettre en pratique mes connaissances afin d'aboutir dans le projet qui m'a été confié.

Je voudrai particulièrement remercier Karim JLASSI, le directeur technique de la société ILTR et mon tuteur, qui m'a contacté, m'a proposé un stage et qui m'a guidé tout du long. Je voudrais aussi en profiter pour remercier Yann GOBRAIT, le directeur de la société ILTR qui a accepté mon offre de stage au sein de sa société.

Je souhaiterai remercier les employés de la société ILTR pour leur amabilité, leurs conseils et leur aide qui ont permit le bon déroulement de mon stage.

Je voudrai remercier plus particulièrement Michel RICHARD ainsi que Sincère BRANDY qui m'ont aidé dans la recherche de mon stage scientifique et technique et qui m'ont fait connaître auprès de Karim JLASSI.

Je souhaiterai remercier l'ESEO pour m'avoir accompagné dans les démarches à suivre pour la recherche d'un stage technique mais aussi pour m'avoir aider à mieux intégrer la vie professionnelle.



Figure 1 - Locaux d'ILTR

Sommaire

Introduction.....	4
 1 - Fiche de synthèse du stage.....	4
 2 - La société ILTR	6
2.1 - Généralités	6
2.2 - Environnement	7
2.3 - Qu'est-ce qu'ILTR ?	7
2.4 - Historique	9
2.5 - Futur d'ILTR.....	10
2.6 - Les Pôles.....	10
2.7 - Reprise de données	11
2.8 - DbVisualizer	12
 3 - Ma mission.....	14
3.1 - Problématique	14
3.2 - Les solutions	15
3.3 - Mise en place de la solution	16
3.3 - Mise en place du projet	21
3.4 - Planning d'avancement	23
3.5 - Partie conceptuelle	25
3.6 - Partie développement.....	30
3.7 - IHM.....	38
3.8 - Problèmes rencontrés	44
 4 - Rapport Personnel	46
4.1 - Environnement du stage.....	46
4.2 - Avis sur l'organisation d'ILTR	46
 Conclusion	47

Introduction

Être préparé à la vie professionnelle après ses études est très important. C'est pour cette raison que l'ESEO s'engage auprès de ses étudiants à suivre une formation qui les prépare à la vie en entreprise. En effet, l'école met en place des périodes de stage en plus des cours. Ces périodes sont au nombre de trois : le stage découverte d'une période d'un mois, le stage scientifique-technique, d'une période de trois mois et enfin le stage de fin d'étude, d'une période de six mois, qui clos le cursus scolaire de l'ESEO.

Le stage scientifique-technique doit servir à acquérir de nouvelles compétences au sein de l'entreprise mais aussi d'utiliser ses connaissances et savoirs pour mener à bien le projet ou le sujet proposé dans le stage. Cette période du stage technique est très importante dans le cursus scolaire de l'ESEO. En effet, c'est la première fois que l'on est confronté directement au métier d'ingénieur.

1 - Fiche de synthèse du stage

J'ai travaillé au sein de la société ILTR à Angers pendant 13 semaines (3 mois) à 35h par semaine du 4 septembre 2017 au 1er décembre 2017. Mon sujet de stage s'intitule "Reprise et Base de données". Pendant cette période de stage j'ai travaillé dans différents domaines tels que la programmation, la modélisation ou encore la Base de données.

Pendant ce stage j'ai mis au point un système d'automatisation de reprise de données. J'ai créé un programme permettant d'aider les employés ILTR à traiter les données de leurs clients.

Au tout début du stage, mon tuteur m'a donné une problématique liée à la reprise de données. En effet, le fichier Excel permettant de recueillir les informations des clients est trop souvent mal complété et conduit directement à une importante vérification des données. Les employés ILTR mettent entre 50 et 70 heures pour corriger les fichiers où sont stockées les données des clients. J'ai donc dû mettre au point une solution permettant de faciliter la reprise de données.

J'ai travaillé en totale autonomie pendant mon stage technique. J'ai été confronté à une problématique, mis au point une solution et enfin j'ai travaillé seul sur un projet en méthode agile. Je voyais mon tuteur lors de "points d'avancements" pour que je présente mon projet et le renseigne sur son avancée. J'ai notamment été aidé par des collègues qui sont chargés de la reprise de données afin de mieux comprendre les besoins et de mettre au point les différentes fonctionnalités de mon système.

Pendant toute la durée de mon stage j'ai rencontré des problèmes plus ou moins importants. Les problèmes seront plus détaillés dans la partie "Problèmes rencontrés" de mon rapport.

problème 1

Le premier problème est survenu dès le début de mon stage car on m'avait fourni le mauvais fichier Excel de reprise de données et j'avais établi mon planning prévisionnel et l'estimation du temps du projet sur une mauvaise base. En effet, le mauvais fichier Excel était moins complexe ce qui a augmenté considérablement le temps de développement du projet. Les tableurs qui composaient le classeur Excel étaient tous différents et certains avaient des liens entre eux, le temps de développement du projet a été multiplié par 3.

problème 2

Le second problème était lié au fichier Excel que l'on reçoit des mairies. En effet, certains clients suppriment, du fichier d'origine, des colonnes d'un tableur car elles ne sont pas utilisées ou encore certains tableurs sont manquants car le client n'en a pas besoin. J'ai dû donc adapter mon programme pour qu'il puisse traiter ces fichiers Excel modifiés.

problème 3

Un dernier problème est survenu lors de la copie des reprises de données. En effet, pour que le programme puisse traiter un tableur en particulier, il faut le séparer du fichier d'origine et donc le copier sur un autre fichier Excel. Cependant, lors de la copie, des sauts de lignes apparaissaient et modifiaient la taille du tableur à traiter et le programme n'arrivaient pas à filtrer les informations.



Figure 2 - Etablissement de la banque Axa (lieu d'ILTR)

2 - La société et le contexte

2.1 - Généralités

La société ILTR a été créée en 2002 par le directeur actuel Yann GOBRAIT, elle travaille actuellement pour des solutions informatiques dans le domaine des Marchés municipaux et de l'ODP (Occupation du Domaine Public). Elle possède plus de 7000 clients, principalement les mairies de grandes villes telles que Lyon, Lille, Biarritz mais aussi à l'étranger en Espagne et en Belgique. La société ILTR propose un service informatique fonctionnel de gestion des Marchés municipaux et de l'ODP à ses clients. Depuis 2 ans l'entreprise ne cesse de se développer et se place en tant que leader dans le marché.



Figure 3 - logo ILTR

La société ILTR dispose de 20 employés repartis dans 5 pôles différents : le pôle Marketing, Projet, Développement, Support/Maintenance et Administratif. De plus, 4 autres personnes sont employées en alternance au sein d'ILTR, 3 dans le pôle Développement et une personne dans le pôle Marketing.

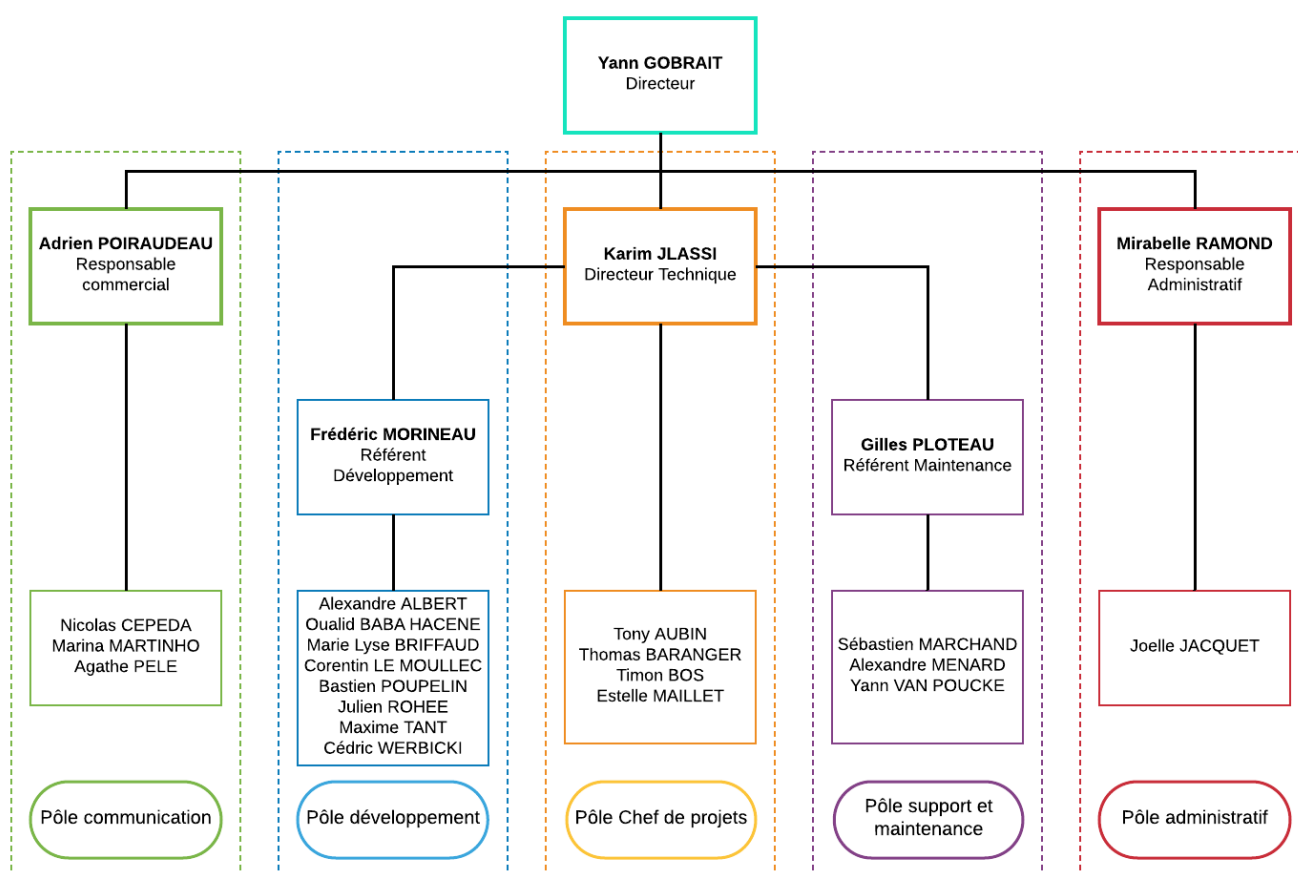


Figure 4 - Organigramme d'ILTR

2.2 - Environnement

L'entreprise a très récemment déménagée au mois de Mai 2017 en vu de son développement et du nombre croissant d'embauches. Elle se localise actuellement à la Roseraie, à Angers, dans les locaux de la banque Axa. A présent la société dispose d'assez d'espace pour pouvoir se développer.

Le lieu de travail est une grande salle de 150 m² où tout les pôles sont réunis. Cette configuration est optimale pour les conditions de travail et pour faciliter la communication entre les différents pôles. De plus, 3 salles de réunions sont mises à disposition des employés et chefs de projet pour pouvoir discuter sans déranger le reste du personnel.

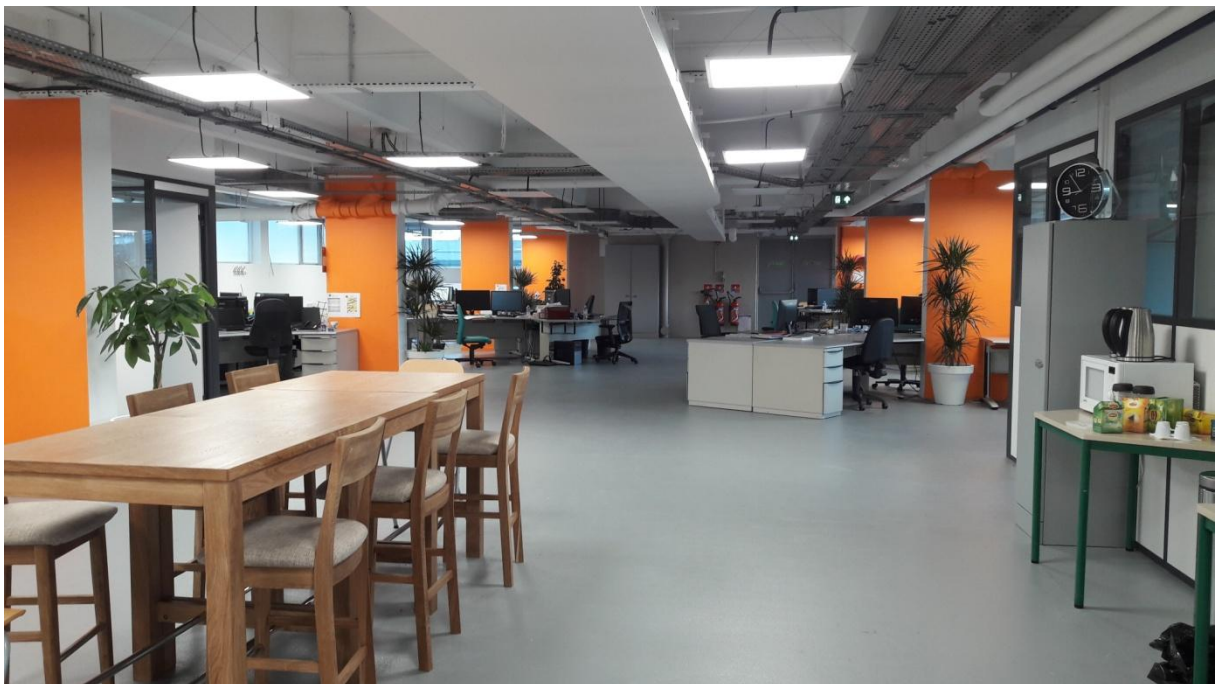


Figure 5 - Locaux d'ILTR

2.3 - Qu'est-ce qu'ILTR ?

ILTR est une société dynamique et profitable. Elle développe et commercialise des solutions informatiques et se concentre de plus en plus sur sa gamme GEODP.

La société garantie à l'ensemble de son équipe de bonnes conditions de travail en proposant :

- Une organisation articulée autour de processus (cf. annexes organisations d'ILTR).
- Diffuser et structurer les échanges d'interventions
- Proposer annuellement des objectifs de réalisation et de développement personnel mesurés par des indicateurs internes.

De plus, ILTR met en œuvre un plan stratégique pour développer et pérenniser l'activité. Ce plan suit plusieurs étapes :

- Analyse de la situation
- Cadre stratégique
- Mise en œuvre
- Cadre de suivi et d'évaluation



Figure 6 - logo du Logiciel GEODP

Le principal atout d'ILTR est un logiciel de gestion des Marchés Municipaux et de l'ODP, appelé GEODP (GESTion de l'Occupation du Domaine Public). Ce logiciel va faciliter le travail des employés de mairies sur de nombreux points. Tout

d'abord, l'ensemble des informations liées à la ville d'une mairie vont être stockées sur une plate-forme de gestion des données. De plus, l'utilisation du logiciel est multiple car nous pouvons l'utiliser via les Smartphones, tablettes et ordinateurs. GEODP va donc permettre aux mairies d'être plus "connectées".

Le logiciel GEODP, créé par ILTR, peut être utilisé dans 9 domaines tels que l'ODP, Les "*Marchés Municipaux*" mais aussi dans les "*Voiries*", les "*TLPE*" (Taxes Locales sur les Publicités Extérieures), La gestions des "*taxis*", les "*terrasses*", les "*parkings*", les "*chantiers*", et les "*taxes de séjour*". Suivant ses besoins, le client va choisir les domaines qu'il souhaite. En effet, une grande ville telle que Lyon va avoir plus de besoins qu'une commune comme Avrillé par exemple.

ILTR propose aussi l'utilisation d'un PDA (Personnal Digital Assistant) lors des marchés Municipaux, les utilisateurs du module "*GEODP Marchés municipaux*" vont utilisé un PDA qui va leur permettre d'optimiser le temps de vérification de facturation. En effet, le PDA va avertir si tel ou tel commerçant a payé son emplacement lors d'un marché. De plus, si un exploitant n'a pas payé son emplacement le PDA va imprimer une facture. Le PDA possède un TPE intégré ce qui va permettre aux commerçants de payer par carte bancaire.



Figure 7 - Photo d'un PDA

2.4 - Historique

Yann GOBRAIT, le directeur d'ILTR, a travaillé pour une branche du Crédit Agricole pour les marchés bestiaux. Il répondait aux offres des clients et mettait en place un système informatique qui répondait à leur attentes. En 2002, M. GOBRAIT a reçu une demande d'un client qui souhaitait un logiciel de gestion des marchés bestiaux durable pour sa ville. Après avoir créé ce logiciel fonctionnel, Yann a décidé de le généraliser pour répondre aux attentes de n'importe quelle ville. Il a ensuite vendu son produit et créé sa propre entreprise, ILTR.

Le logiciel n'englobait à cette époque que le module "*Marchés municipaux*", alors qu'aujourd'hui il traite 9 modules différents. Au fur et à mesure des années, la demande des clients était de plus en plus forte pour ce logiciel de gestion et de nouveaux modules ont vu le jour, notamment le module "*ODP*".

Depuis 2015, la société s'est considérablement développée grâce aux demandes des mairies et ILTR a embauché 15 personnes supplémentaires.

La société était, dans les années 2010, seule dans les solutions informatiques pour l'ODP et les Marchés Municipaux. La concurrence était moindre et a permis à la société de se développer.



Figure 8 - Grande salle de réunion

2.5 - Futur d'ILTR

Dans une optique d'évolution, ILTR souhaite mettre en place une nouvelle version du logiciel GEODP plus performante que l'ancienne.

Depuis le début de mon stage, la société a fusionné avec l'un de ses concurrents, PANTERGA, une société de gestion de l'ODP à Nice, dans l'optique d'étendre son marché dans le sud de la France. La société ILTR a dû s'adapter et revoir toute son organisation car les clients de PANTERGA se sont ajoutés à ceux d'ILTR. De plus, l'entreprise PANTERGA propose un autre logiciel de gestion dans le domaine de l'ODP. Ce logiciel se nomme DIBTIC et est adapté dans la gestion des Marchés Municipaux, de l'ODP et de la TLPE.

2.6 - Les Pôles

Pôle administratif

Ce pôle est axé sur la communication interne de l'entreprise. Il va aussi gérer le bon déroulement des commandes comme par exemple envoyer les devis, les factures ou encore gérer la livraison du matériel (PDA).

Pôle communication

Le pôle communication va gérer la promotion des produits d'ILTR et donc la tenue du site Web. De plus, ce sont les employés de ce pôle qui vont effectuer des appels d'offres aux clients.

Pôle développement

Le pôle développement représente la majorité des salariés d'ILTR. Les développeurs mettent en place des solutions informatiques et innovent les produits de l'entreprise.

Pôle chef de projet

Les membres du pôle chef de projet vont gérer le déroulement d'un projet avec le client. Ils vont définir les besoins, planifier et vont suivre l'avancée du projet auprès du pôle développement et support/maintenance.

Pôle maintenance et support

Les 3 employés du pôle de maintenance ont pour principale mission de gérer la satisfaction des clients. Ils sont constamment en relation avec eux via un microphone. Le pôle maintenance travaille sur 2 grandes activités que sont le déploiement et le support client.

Le support client est le service après-vente d'ILTR, le client va appeler le pôle de maintenance pour un problème ou un besoin. Pour les problèmes mineurs (soutis d'utilisation de GEODP ou erreur de manipulation), les employés du pôle de maintenance vont corriger le problème en ligne avec le client. Ils utilisent notamment le logiciel "*TeamViewer*" pour se logger chez le client afin de faciliter la résolution du problème. Lorsque le problème devient plus important (problème de fonctionnement de GEODP), le pôle maintenance résout le problème sur plusieurs jours.

En parallèle de la tâche de support, les employés du pôle de maintenance doivent assurer le déploiement du logiciel GEODP chez un nouveau client. En effet, lorsque un nouveau client souhaite utiliser le progiciel GEODP, les employés du pôle se chargent de le former, de l'aider à l'installation du logiciel et d'effectuer la reprise de données.

2.7 - Reprise de données

La reprise de données c'est le transfert de l'ensemble des informations du nouveau client vers une Base de données. Chaque client possède une Base de données qui lui est propre avec ses données afin qu'il puisse utiliser le logiciel GEODP, par exemple la Base de données de Rouen s'intitule GEODPROUEN. Ces informations vont être stockées dans un tableur Excel. Ce fichier contient des onglets qui traitent de parties différentes, par exemple les domaines, les activités, les redevables ou encore les tarifs. Ces informations sont très importantes pour le logiciel de gestion GEODP car plus le fichier Excel sera complet, plus il y aura d'informations et plus l'utilisation du progiciel sera optimisée.

Lorsque un nouveau client souhaite utiliser le logiciel GEODP pour la gestion de sa ville, le pôle "Chef de projet" se charge d'envoyer au client le fichier Excel type.

Ce tableur Excel type contient 14 onglets différents traitant des :

- Groupes d'activités
- Activités
- Domaines
- Types de contact
- Liste des employés
- Zones
- Tarifs
- Liste des éléments
- Eléments ODP
- Dossiers
- Redevables
- Pièces justificatives
- Dossiers photos
- Secteurs d'activités

Le fichier Excel de reprise de données contient les données les plus importantes Tables de la Base. Les 14 tableurs du fichier de reprise de données permet de compléter 34 Tables dans la Base de données et ces Tables constituent le cœur de la Base de données (cf. annexe tables.pdf).

Certains tableurs du fichier Excel de reprise de données couvrent 5 Tables de la Base de données et d'autre n'en couvrent qu'une. Par exemple le tableur traitant des Tarifs couvre 5 Tables différentes (TARIF, TARIF_DATE, TARIF_DATE_PERIODE, TARIF_LANGUE et TARIF_SOUS_ZONE) alors que le tableur traitant des redevables ne couvre que la Table EXPLOITANT.

3 - Ma mission

3.1 - Problématique

Ma mission au sein de la société d'ILTR était liée au pôle de maintenance car une problématique y était soulevée. En effet, les employés du pôle de maintenance passaient trop de temps sur une reprise de données, entre 50 heures et 70 heures ce qui était beaucoup trop. Une reprise de données est une vérification et une intégration d'informations dans une Base de données. Le temps estimé pour une reprise de données "idéale" est d'une dizaine d'heures.

Les reprises de données trop longues sont un vrai "frein" au bon développement d'ILTR car tant que les employés du pôle de maintenance sont en activité de reprise de données, le service support pour aider les clients à distance est impacté. En effet, les employés du pôle de maintenance ne pourraient pas assurer en même temps une reprise données et un service support efficace. Cette perte de temps pourrait entraîner, à long terme, une baisse de la satisfaction client et une diminution du chiffre d'affaire qui impacterait directement l'entreprise et ses salariés.

De nombreuses erreurs apparaissent dans le fichier Excel. Ces erreurs interviennent dans l'insertion de données dans la nouvelle Base de données mais aussi dans la période de test du logiciel GEODP. Par exemple, pour l'insertion, certaines informations d'une colonne dépassent la taille maximale et l'insertion est impossible. Lors des tests du logiciel GEODP, certains numéros de téléphone ou code SIRET n'étaient pas corrects ainsi que certaines adresses de redevables. De plus, certains clients rappelaient le pôle "Chef de projet" afin de signaler qu'ils avaient envoyé un fichier Excel incomplet et qu'il manquait certaines informations importantes.

Ces différents problèmes sont liés aux données du fichier Excel et proviennent de fautes humaines. Certains tableurs complétés par les employés de mairies de grandes villes (Nice par exemple) font plus de 100 000 lignes, des erreurs peuvent donc survenir lors de la saisie des informations. C'est pour ces différentes raisons que le temps d'une reprise de données peut être très long et peut mobiliser un employé du pôle de maintenance jusqu'à 2 semaines.

Ma mission était donc de mettre en place une solution afin de diminuer considérablement le temps consacré à l'activité de reprise de données. J'ai donc dû mettre au point une méthodologie afin de répondre à la problématique.

Pour cette mission j'étais en totale autonomie, j'ai dû donc faire plusieurs propositions de solutions, choisir la meilleure solution et la développer. Si j'avais travaillé avec des employés du pôle de maintenance ils m'auraient influencé dans mes propositions de solutions et c'est pour cette raison que j'ai dû agir seul.

3.2 - Les solutions

Afin de répondre à la problématique, 2 propositions se distinguent. On peut appliquer une solution sur :

- L'envoi du fichier
- La réception du fichier

Proposition 1 :

Pour cette méthodologie, il suffit de trouver des solutions sur le type du fichier à envoyer.

Solution 1

Création d'un formulaire où chaque champ doit être référencé par le client. Le but étant d'envoyer un autre type de fichier que le tableur Excel. Il faudrait envoyer au client un formulaire pour qu'il puisse le compléter. Si certains champs ne sont pas bien complétés un message apparait et signale l'erreur.

Points forts	Points faibles
Faciliter l'insertion	Saisie redondante pour le client
Meilleur visuel	Augmentation du temps de saisie pour le client
Optimisation du temps passé dans la vérification pour les employés ILTR	

Solution 2

Restreindre certaines modifications du fichier Excel envoyé au client. Il faudrait utiliser les paramètres d'Excel pour restreindre la liberté des clients lors de la saisie des informations. Le client ne pourra alors que remplir les cellules du fichier Excel.

Points forts	Points faibles
Empêcher les fortes modifications du document	Faible optimisation du temps passé à la vérification
	Diffusion d'un mot de passe au client pour déverrouiller la page, le client peut enlever la protection

Solution 3

Utilisation de "Google sheets" pour que le client remplisse correctement les tableaux.

A la différence du tableur Excel, les cellules correctement remplies seront en vert et les celles comportant des erreurs seront en rouge. Cette méthode permettrait au client d'éviter les erreurs de saisie.

Points forts	Points faibles
Prévenir le client si une information est correcte ou non	Augmentation du temps de saisie du client
Pas de temps de vérification pour les employés d'ILTR. Temps de reprise de données limités à quelques heures.	Nécessité d'avoir un compte Google pour utiliser "Google sheets".

Proposition 2 :

Solution

Utiliser une méthode qui « filtre » le fichier Excel renvoyé par le client. Nous allons apporter une modification au fichier pour qu'il soit intégrable sur « DbVisualizer », le logiciel de gestion de Base de données d'ILTR, lors de l'insertion dans la base de données.

Points forts	Points faibles
Optimisation du temps pour la vérification pour les employés ILTR	Solution complexe car il faut envisager tous les cas possibles
Optimisation du temps pour le client	Beaucoup de temps sur le développement

La seconde proposition a été retenue, il faut agir sur la réception du fichier. En effet, le but de la mission est de diminuer le temps consacré à la reprise de données pour les employés d'ILTR tout en conservant une certaine satisfaction client. Il ne suffit pas de diminuer le temps de la reprise de données pour en rajouter au client.

Le client va donc remplir le même fichier Excel qu'avant car certaines mairies possèdent des milliers de commerçants et des centaines de rues. Agir sur l'envoi du fichier aurait été un inconvénient pour les clients.

3.3 - Mise en place de la solution

L'idée principale de la solution est de créer un programme qui va "filtrer" les données du tableur Excel et de détecter les erreurs éventuelles qui ont échappé à la vérification du pôle "Chef de projet" d'ILTR.

Suite à cela, les données "filtrées" par le programme vont être intégrées sur le nouveau logiciel de gestion de Base de données d'ILTR, DbVisualizer.

Etude du logiciel DbVisualizer



Figure 10 -
saas.hpe.com

La société ILTR utilise depuis peu ce nouveau logiciel qui est plus fonctionnel et plus optimisé que leur ancien logiciel de gestion de Base de données, Oracle. DbVisualizer n'étant pas très connus des employés

d'ILTR, j'ai dû procéder à une étape de recherche afin d'exploiter le potentiel du logiciel. J'ai remarqué que DbVisualizer possède une fonctionnalité intéressante au niveau de l'import des données. En effet, il est possible d'importer des données dans une nouvelle Base via un tableur Excel.

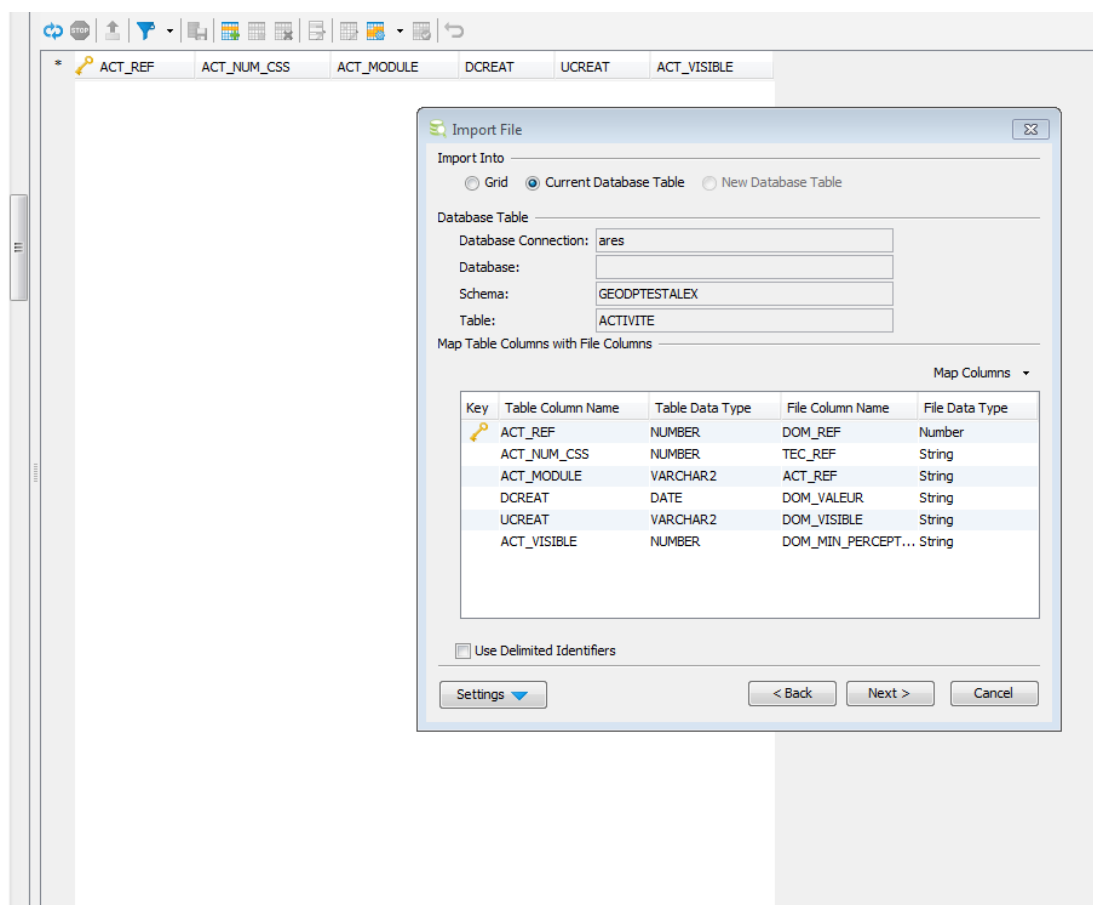


Figure 11 - fonction d'import de DbVisualizer

Grâce à cette fonctionnalité on peut plus facilement vérifier les données que l'on souhaite intégrer à la Base de données car elles seront structurées dans un tableur et non dans des lignes d'insertion avec le langage SQL (INSERT INTO).

Pour vérifier les capacités d'import de DbVisualizer, j'ai dû procéder à différentes étapes. J'ai d'abord créé une nouvelle base de données dans le réseau d'ILTR « Ares » intitulé « GEODPTESTALEX » copie conforme de « GEODPROUEN ». La Base de données de la ville de Rouen est très complète, mon tuteur m'a conseillé de tester le logiciel avec cette Base de données. J'ai correctement étudié DbVisualizer j'ai procédé à plusieurs étapes :

- Vérifier les fonctionnalités d'import de « DbVisualizer » avec une table de données simple
- Vérifier de nouveau les fonctionnalités d'import du logiciel avec une base de données simple avec des relations entre les tables.
- Vérifier les fonctionnalités d'import du logiciel avec le tableur "Activités" du fichier de reprise de la ville de Rouen.

Etape 1

La fonctionnalité d'import à partir d'un fichier Excel pour une table sans clé étrangère est simple et ne présente aucun problème particulier.

	A	B	C
1	Tableau Etudiant		
2	NUMETUDIA	NOM	PRENOM
3	1	Richard	Léa
4	2	Menard	Alexandre
5	3	Cochet	Nicolas
6	4	Barraud	Axel
7	5	Le Gacque	Tristan

Figure 12 - Table Etudiant sur Excel


ETUDIANT	
	NUMETUDIANT INTEGER
	NOM VARCHAR2(20)
	PRENOM VARCHAR2(20)

Figure 13 - Table Etudiant sur DbVisualizer

Etape 2

Pour cette étape, il suffit d'effectuer le même processus que pour l'étape 1 avec plusieurs tables présentant des liens entre elles (clés étrangères) pour se rapprocher des Bases de données principalement utilisées.

A	B
1	Matiere
2	CODEMATIERE INTITULE
3	1 Maths
4	2 Anglais
5	3 Français
6	4 Physique
7	
8	

A	B
Ue	
CODEUE	INTITULE
600	Langues
900	Sciences

Carte			
NUMCARTE	CREDIT	NUMTUDIAN	
11	10	1	
22	20	2	
33	30	3	
44	40	4	
55	50	5	

A	B	C	D	E
Epreuve				
CODEEPRU	JOUR	LIEU	CODEMATIERE	
105	22/09/2017	Angers	1	
103	23/09/2017	Paris	2	

Passe			
NUMTUDIA	CODEEPRU	NOTE	
1	105	20	
2	105	18	
3	105	6	
4	103	9	
5	103	15	

EstComposéeDe		
CODEUE	CODEMATIERE	
600	2	
600	3	
900	1	
900	4	

Figure 14 - Tables de la Base à tester

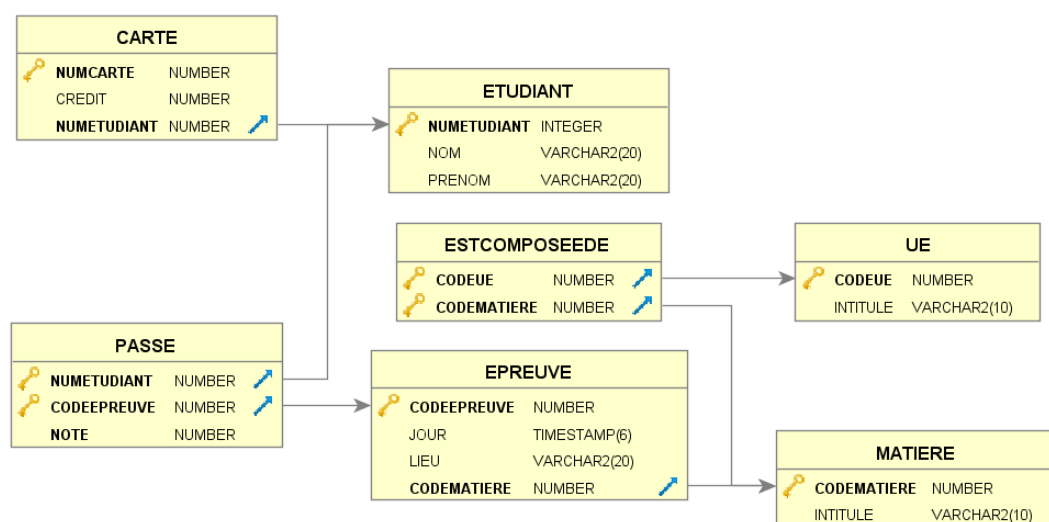


Figure 15 - Tables à tester sur DbVisualizer

Pour cette étape, j'ai pu remarquer que le module d'import de DbVisualizer comportait 2 particularités. En effet, à partir d'un tableur Excel on peut créer une nouvelle Table mais aussi importer des données dans une Table préalablement créée. Pour une reprise de données, la Base est créée avant d'importer les informations des nouveaux clients. Il faut donc utiliser la seconde fonctionnalité du module d'import du logiciel.

Etape 3

Pour vérifier l'import du tableur traitant des activités de la ville de Rouen il faut adapter le tableur Excel. Pour que des données puissent être intégrées dans des Tables de la Base de données il faut d'abord séparer les informations dans des tableurs différents et il faut que les colonnes des nouveaux tableurs soient exactement les mêmes que celles des Tables de la Base.

En effet, le tableur traitant des activités de Rouen possède des informations relatives à 2 Tables de la Base de données (ACTIVITE et ACTIVITE_LANGUE).


*	 ACT_REF	ACT_NUM_CSS	ACT_MODULE	DCREAT	Ucreat
1	2		13 placier	2017-05-04 09:29:12	iltr
2	3		11 odp	2017-06-30 09:58:46	iltr
3	4		12 odp	2017-06-14 11:46:18	iltr

Figure 16 - Table ACTIVITE sur DbVisualizer



*	 ACT_REF	 LAN_REF	ACT_NOM	DCREAT	Ucreat
1	2		1 Marchés municipaux	2017-05-04 09:29:12	iltr
2	3		1 TLPE	2017-06-30 09:58:46	iltr
3	4		1 ODP	2017-06-14 11:46:18	iltr

Figure 17 - Table ACTIVITE_LANGUE sur DbVisualizer

Il faut donc séparer les données et les ajouter dans 2 nouveaux tableurs

Nom de l'activité	Module	Nom du groupe d'activité	Couleur de l'activité
TLPE	odp	ODP	11
ODP	odp	ODP	12



ACT_REF	ACT_NUM_CSS	ACT_MODULE	DCREAT	Ucreat
1	11	odp	10/10/2017	ILTR
2	12	odp	10/10/2017	ILTR

ACT_REF	LAN_REF	ACT_NOM	DCREAT	Ucreat
1	1	TLPE	10/10/2017	ILTR
2	1	ODP	10/10/2017	ILTR

Figure 18 - Séparation des données

3.3 - Mise en place du projet

J'ai eu l'idée de créer un programme qui va d'abord filtrer les données du tableur Excel et en détecter les erreurs mais qui va aussi mettre en forme les données pour qu'elles puissent être intégrées. En effet, après avoir testé les fonctionnalités d'import de DbVisualizer, il est nécessaire que les tableurs soient adaptés pour être insérés dans la nouvelle Base de données. Un système de mise à jour est aussi envisagé dans le programme afin de faire face au problème des tableurs incomplets. En effet, on pourra importer le nouveau tableur pour que les données soient remplacées et mises à jour.

Les fonctionnalités du programme sont :

- "Filtrer" les fichiers tableurs défectueux.
- Aider les employés d'ILTR à l'insertion des informations des clients dans la nouvelle Base de données.
- Effectuer une mise à jour de la base de données si le client décide d'envoyer d'autres fichiers Excel.



Figure 19 - Schéma description du programme

Avant de commencer à créer le programme il est nécessaire de passer par une étape de réflexion. Il faut réfléchir à la procédure du logiciel, c'est à dire comment le programme va pouvoir être utiliser par les employés ILTR. De plus, il faut mettre en place une IHM afin de connaître les fonctionnalités importantes du programme.

Une question importante était : "Comment peut-on filtrer les données des tableurs ?". En effet, il suffit de savoir que les fichiers CSV contiennent des informations séparées par des points virgules en colonnes. Il est donc facile ensuite de traiter les tableurs sachant que les informations sont séparées par un caractère connu.

Nom de l'activité	Module	Nom du groupe d'activité	Terme exploitant	Couleur de l'activité
TLPE	odp	ODP	Redevable	11
ODP	odp	ODP	Redevable	12

Figure 20 - Tableur des activités de la ville de Rouen

```
Nom de l'activité;Module;Nom du groupe d'activité;Terme exploitant;Couleur de l'activité
TLPE;odp;ODP;Redevable;11
ODP;odp;ODP;Redevable;12
```

Figure 21 - Conversion du tableur des activités de Rouen en CSV

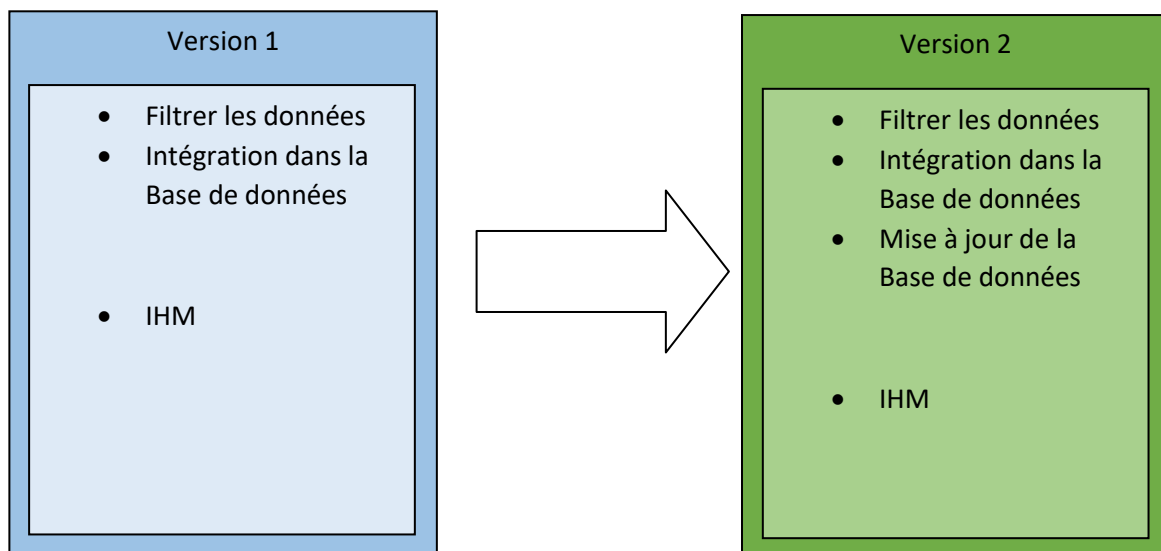
Il est donc indispensable que les fichiers Excel qui vont être traités par le programme soient des fichiers convertis en CSV.

Les employés ILTR vont donc dans l'ordre :

- Recevoir du client le fichier Excel à traiter
- Séparer les onglets du fichier Excel d'origine (par exemple si le fichier Excel d'origine possède 8 onglets différents, il faut créer 8 tableurs) et les convertir en fichier CSV.
- Importer chaque tableur à traiter dans le logiciel dans un certain ordre.
- Intégrer dans la Base de données les nouveaux fichiers traités.

J'ai choisi de développer mon programme sur Eclipse et donc d'utiliser le langage Java car c'est le langage dans lequel je dispose de plus de connaissances grâce notamment au projet "BeAnArtist" que j'ai codé lors de ma première année de cycle ingénieur. En effet, le projet "BeAnArtist" était complet car j'ai pu développer un programme dans son intégralité ainsi qu'une IHM. Je me suis donc servi des bases que j'ai acquises lors de ce projet pour créer mon programme.

De plus le programme va être découpé en deux versions. En effet, il est nécessaire de créer une première version qui soit fonctionnelle avant de commencer la seconde version. La première version possède les deux fonctionnalités basiques pour une reprise de données idéale. La seconde version sera utilisée pour tout scénario de reprise de données (lorsque le client envoie des fichiers incomplets par exemple).



3.4 - Planning d'avancement

Planning d'avancement

Activité	Plan Début	Plan Durée	Réel début	Réel durée	Jours écart	Pourcentage écart	Pourcentage Achevé	Plan Fin	Réel Fin
Analyse du problème	04/09/2017	35	04/09/2017	35	0	0,00%	100%	10/09/2017	10/09/2017
Conception de la version 1	11/09/2017	15	11/09/2017	15	0	0,00%	100%	13/09/2017	13/09/2017
Programmation de la version 1	14/09/2017	120	14/09/2017	133	13	13,00%	95%	06/10/2017	10/10/2017
Etude de cas (Test de la version 1)	06/10/2017	45	10/10/2017	140	95	95,00%	95%	16/10/2017	06/11/2017
Développement IHM	16/10/2017	40	06/11/2017	25	-15	-15,00%	90%	26/11/2017	
Etude export DbVizualizer	26/11/2017	20						30/11/2017	
Conception de la version 2	30/10/2017	20						03/11/2017	
Programmation de la version 2	03/11/2017	80						21/11/2017	
Etude de cas (Test de la version 2)	21/11/2017	45						28/11/2017	

Figure 22 - Planning d'avancement du projet

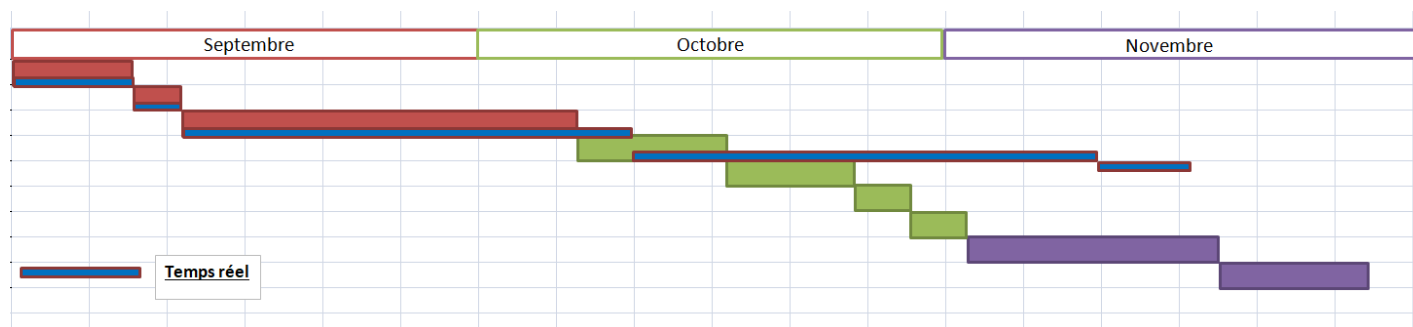


Figure 23 - Planning d'avancement du projet (Suite)

Il est très important de fixer un planning prévisionnel dans un projet pour avancer étape par étape. Le planning sert à décomposer le projet en différents jalons et de se fixer des objectifs secondaires pour aboutir à l'objectif final.

J'ai travaillé en méthode agile dans ce projet. En effet, lorsque la première version a été créée, j'ai dû passer par une période de test pour vérifier les fonctionnalités de mon programme. Il était important de ne pas concevoir le logiciel d'un coup car s'il ne répondait pas aux besoins, beaucoup de temps aurait été perdu. Sans le travail en méthode agile, il faut recommencer au tout début pour le modifier ce qui prend énormément de temps.

C'est pour cette raison qu'à chaque étape de mon planning je dois tester mon programme, par exemple pour le développement de la version 1 et 2 ou encore pour l'IHM. A chaque période de test, le programme doit agir en situation réelle, il doit pouvoir assurer une bonne reprise de données.

Détails des activités du planning d'avancement.

Pour les activités du planning d'avancement, j'ai dû estimer la durée de chacune. C'était un exercice assez difficile car je n'avais fait que très peu de planning d'avancement lors de la première année de cycle ingénieur. J'ai notamment, rencontré plusieurs problèmes que j'ai détaillé dans la partie "problèmes rencontrés" de mon rapport.

Analyse du problème :

Cette activité permet de mettre en place diverses solutions pour résoudre le problème. J'ai ensuite choisi la meilleure solution pour répondre à la problématique et je l'ai développé.

Conception de la version 1:

Pour cette étape, j'ai mis en place plusieurs diagrammes et schémas avant de commencer à développer. Cette activité m'a permis d'avoir une meilleure approche pour commencer le développement de mon projet.

Programmation de la version 1:

C'est l'activité principale de mon stage technique car tout l'aspect technique du projet est centré dans cette étape.

Etude de cas (pour la version 1):

Cette étape permet de tester le programme développé dans l'activité précédente pour y corriger certaines erreurs. Après cette étape la version 1 doit être opérationnelle.

Développement de l'IHM :

Avec l'élaboration d'une IHM, les employés d'ILTR pourront utiliser la version 1 lors de reprise de données.

Etude export DbVisualizer :

C'est à cette étape que commence la mise en place de la seconde version du programme qui permettra de gérer tous les cas de figures de reprise de données.

Conception de la version 2 :

Comme lors de la conception pour la première version, il faut refaire une partie conception du projet pour la seconde version.

Programmation de la version 2 :

Pour cette étape, il suffit de créer et d'ajouter la fonctionnalité de mise à jour dans la version 1 pour créer la version 2.

Etude de cas (pour la version 2) :

Cette activité permettra de tester la version 2 dans son intégralité. Suite à cela les employés d'ILTR pourront tester tous les cas de figures de reprise de données.

3.5 - Partie conception du projet

Comme dans tout projet, une partie de conception est importante afin d'avoir un meilleur visuel du projet. Cette partie permet d'avoir une approche plus poussée du projet et de mieux expliquer les différentes fonctionnalités.

Je vais donc décrire à l'aide d'un diagramme d'activités et d'un cas d'utilisation, les différentes fonctionnalités du programme pour la première version.

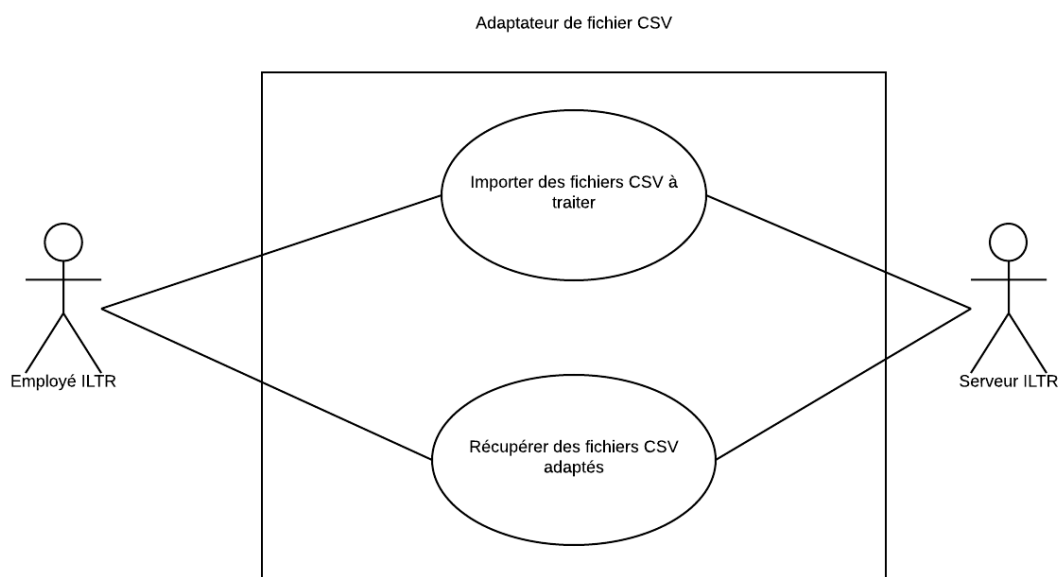


Figure 24 - Use case "Adaptateur du fichier CSV"

Pour ce cas d'utilisation de la première version, l'employé chargé de la reprise de données a besoin d'importer des tableurs Excel convertis en CSV puis de récupérer des nouveaux tableurs.

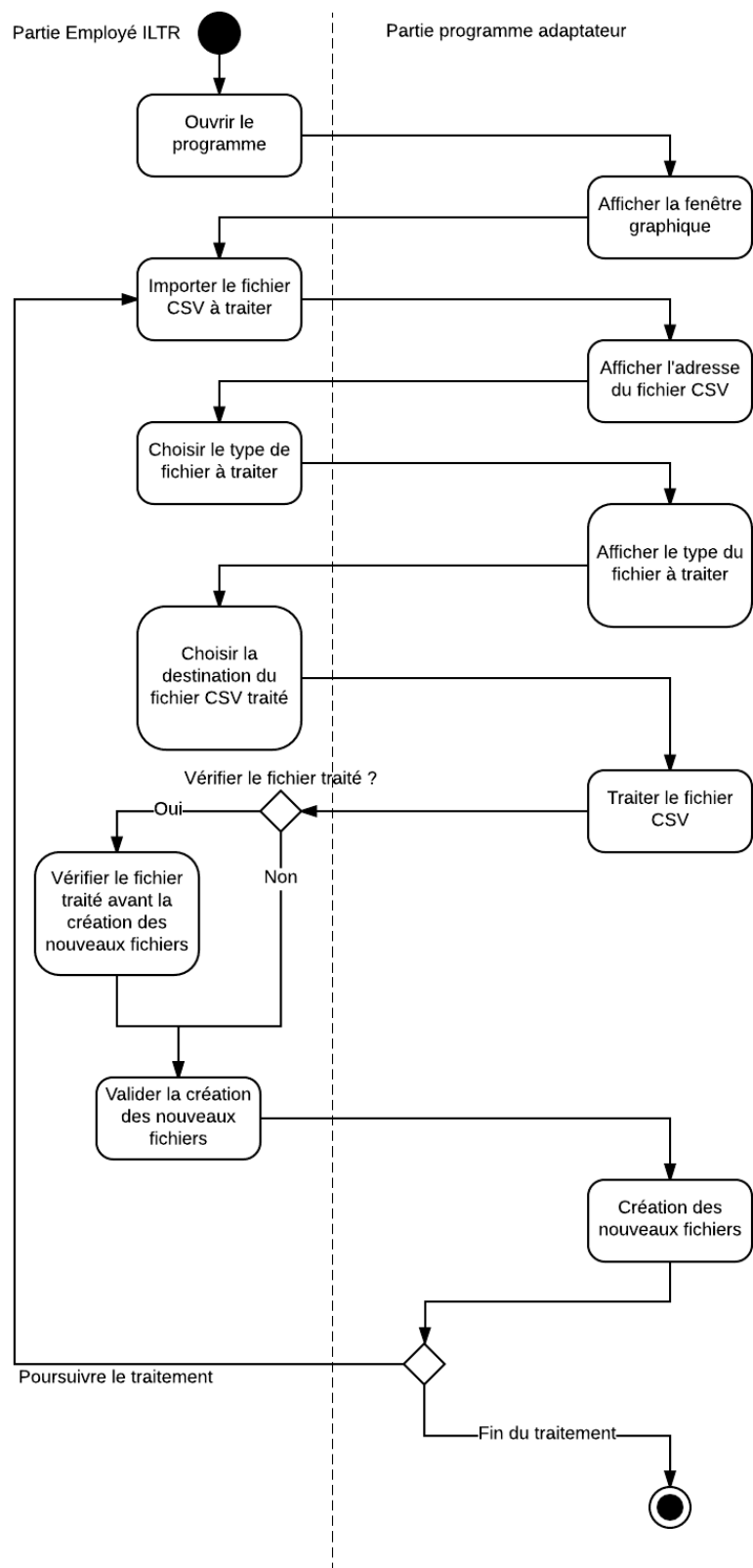


Figure 25 - Digramme d'activités du projet

Le diagramme d'activités ci-dessus permet de comprendre comment le logiciel va réagir face aux actions de l'utilisateur. Il permet aussi de s'appropriier les différentes fonctionnalités qui composent le programme.

J'ai dû élaborer un prototype d'IHM qui met en avant toutes les fonctionnalités du programme comme le module d'import, la sélection du type du fichier ou encore le choix de l'adresse d'exportation des nouveaux fichiers.

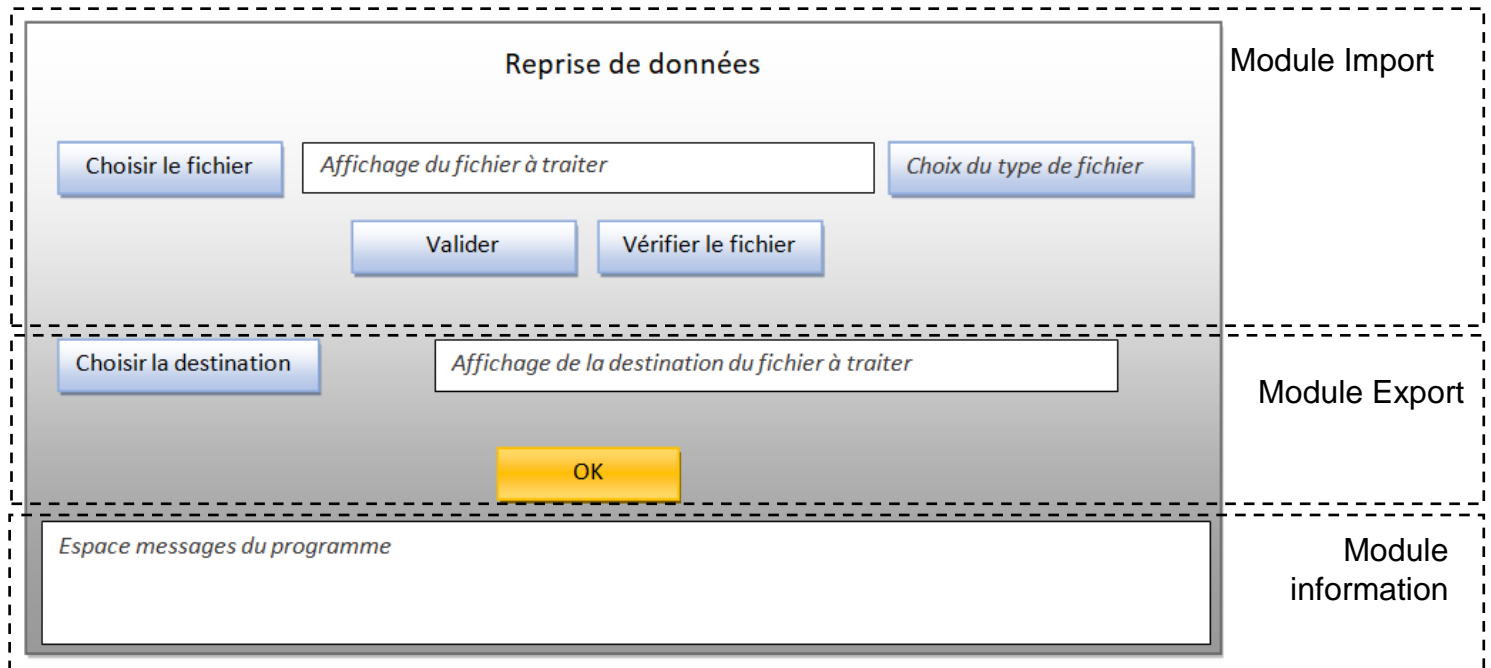


Figure 26 - Prototype IHM

L'IHM est composée de 3 modules distincts, chaque module va traiter des parties différentes du programme. Le "*module information*" n'est pas lié aux deux autres modules car il va constamment fournir des informations à l'utilisateur durant toute la durée du traitement. En effet, il pourra prévenir si une erreur est apparue et à quel niveau du traitement. Il pourra signaler quel type de fichier a été traité précédemment.

De plus, les "*modules import et export*" sont liés entre eux. Pour que le traitement puisse être effectué il faut que les différents éléments soient paramétrés :

- Le fichier CSV importé
- Le type du fichier (Domaines, Activités, Redevables...)
- L'adresse d'exportation

Cas d'utilisation "Importer des fichiers CSV à traiter"

Résumé	L'employé ILTR va importer des fichiers tableurs CSV envoyés par les clients pour que le programme puisse les traiter.
Acteur Primaire	Employé ILTR
Acteur Secondaire	Serveur ILTR
Précondition	Le client a envoyé tous les fichiers à importer. L'employé ILTR a converti les fichiers Excel en fichiers CSV.
Début	L'employé ILTR ouvre le programme.
Fin	L'employé ILTR vérifie le fichier traité
Postcondition	Si les fichiers ont pu être importés, il ne seront pas modifiés.
Scénario nominal	<ol style="list-style-type: none">1. L'employé ILTR ouvre le programme.2. Le programme lance sa fenêtre graphique.3. L'employé ILTR clique sur le bouton "Choisir le fichier".4. L'employé ILTR sélectionne le fichier CSV à traiter.5. Le programme affiche le fichier CSV choisi.6. L'employé ILTR clique sur le bouton "Choix du type de fichier".7. Le programme affiche le type du fichier choisi par l'employé ILTR.8. L'employé ILTR clique sur le bouton "Valider".9. Le programme va effectuer le traitement.10. L'employé ILTR vérifie le fichier traité.
Scénario alternatif	<ol style="list-style-type: none">5.1 Si l'employé ILTR souhaite changer les fichiers CSV, retour à l'étape 3.5.2 Si le fichier choisi ne possède pas l'extension CSV, retour à l'étape 3.7.1 Si l'employé ILTR souhaite changer le type du fichier à traiter, retour à l'étape 6.

Cas d'utilisation "Récupérer des fichiers CSV adaptés"

Résumé	L'employé ILTR va récupérer sur le serveur ILTR les nouveaux fichiers CSV qui pourront être intégrés dans la Base de données sur "DbVisualizer".
Acteur Primaire	Employé ILTR
Acteur Secondaire	Serveur ILTR
Précondition	Le programme a traité le fichier CSV sélectionné.
Début	L'employé ILTR sélectionne la destination des fichiers CSV traités en cliquant sur le bouton "Choisir la destination".
Fin	L'employé ILTR ferme le programme.
Postcondition	Les nouveaux fichiers CSV pourront être récupérés sur l'espace choisi par l'employé ILTR.
Scénario nominal	<ol style="list-style-type: none">1. L'employé ILTR sélectionne la destination des fichiers CSV traités en cliquant sur le bouton "Choisir la destination".2. Le programme affiche l'adresse de destination des fichiers traités.3. L'employé ILTR clique sur le bouton "OK" du programme.4. Le programme envoie les nouveaux fichiers à la destination choisie par l'employé ILTR.5. Le programme affiche un message de succès du traitement.6. L'employé ferme le programme.
Scénario alternatif	<ol style="list-style-type: none">2.1 Si le programme n'affiche pas l'adresse de destination, un message d'erreur apparaît, retour à l'étape 1.2.2 Si l'employé ILTR souhaite changer d'adresse de destination, retour à l'étape 1.3.1 Si l'employé ILTR clique sur le bouton "OK" sans avoir choisi d'adresse de destination, les nouveaux fichiers seront envoyés à l'adresse d'importation.5.1 Si l'employé ILTR souhaite importer d'autres fichiers, retour au cas d'utilisation "Importer des fichiers CSV à traiter".

3.6 - Partie développement

Projet Filtre

J'ai codé mon programme sur Eclipse en Java car je dispose de plus de compétences dans ce langage.

Avant d'avoir commencé à coder mon programme j'ai créé une version simplifiée de mon programme, que j'ai appelé "Filtre", disposant de plusieurs fonctionnalités de base :

- Importer un fichier CSV
- Modifier le contenu du fichier
 - Limiter la taille des cellules
 - Remplacer un caractère
 - Détecter la position d'une cellule dans un tableau
- Exporter le tableau modifié

Il était important que le mini-programme "Filtre" fonctionne pour commencer la programmation de la version 1.

Pour importer un fichier CSV, j'ai utilisé les fonctions de base de Java pour lire un fichier. J'ai utilisé un *"BufferedReader"* qui va lire le fichier CSV et va stocker les informations du fichier dans une *ArrayList<String>*. Les différents éléments de l'*ArrayList* sont séparés par une virgule et chaque élément de l'*ArrayList* est une ligne du tableur CSV. Dans chaque ligne de l'*ArrayList*, les cellules du tableau sont séparées par des points virgules.

Par exemple, ce tableur qui correspond aux activités de la mairies de Rouen,

Nom de l'activité	Module	Nom du groupe d'activité	Terme exploitant	Couleur de l'activité
TLPE	odp	ODP	Redevable	11
ODP	odp	ODP	Redevable	12

devient une fois importé dans le programme :

[Nom de l'activité; Module; Nom du groupe d'activité; Terme exploitant; Couleur de l'activité, TLPE;odp;ODP;Redevable ;11, ODP;odp;ODP;Redevable;12].

Ensuite pour parcourir le tableau, j'ai utilisé deux boucles *"for"* , une pour les lignes, l'autre pour les colonnes, qui vont me permettre de mettre en place mes différents traitements de données. Le mini-programme "Filtre" va limiter la taille des cellules. J'ai utilisé la méthode *substring()* de Java pour ne récupérer qu'une partie d'une chaîne de caractère.

```

public static List<String> traitementTailleNom(List<String> tab, int limiteNom){
    //Méthode de traitement du fichier Initial
    List<String> tabFin = new ArrayList<String>();
    //On va créer le tableau final
    tabFin.add(tab.get(0));
    for(int i=1;i<tab.size();i++){
        String lignetab = tab.get(i);
        //On découpe les lignes par ligne
        String[] colonnetab = lignetab.split(";");
        //On découpe par colonne et lorsque l'index rencontre un point virgule on crée une autre colonne
        String lignetabFin = "";
        //on va créer les lignes que contiendra le tableau final
        for(int j=0;j<colonnetab.length;j++){
            //On commence le traitement à partir de la seconde ligne
            if(colonnetab[j].length()>limiteNom){
                colonnetab[j] = colonnetab[j].substring(0, limiteNom);
            }
            //limite de la taille du nom du groupe d'activité à la limite
            //System.out.println(colonnetab[j]);
            if(j<colonnetab.length-1){
                lignetabFin += colonnetab[j]+";";
            }
            else{
                lignetabFin += colonnetab[j];
            }
        }
        //On ajoute les colonnes dans les nouvelles lignes
        //tabFin.add(tab.get(0));
        tabFin.add(lignetabFin);
        //On ajoute les lignes au tableau de fin
    }
    return tabFin;
    //On retourne le tableau final
}

```

Figure 27 - Méthode traitementTailleNom(tab, limite)

Lorsque j'utilise dans un "main" la méthode "traitementTailleNom(tab, 3);" les 3 premiers caractères de chaque chaîne de caractère vont être récupérés. Le traitement qui limite la taille des cellules n'affecte pas la première lignes qui contient le nom des colonnes du tableur.

Après avoir utilisé ce traitement, le tableau retourné est de la forme :

[Nom de l'activité;Module;Nom du groupe d'activité; Terme exploitant; Couleur de l'activité,TLP;odp;ODP;Red;11,ODP;odp;ODP;Red;12].

La dernière fonctionnalité testée est l'export du tableur. A l'inverse du module import, j'ai utilisé la fonction Java "BufferedWriter". Le module chargé de l'export va créer un nouveau fichier CSV et va stocker les informations modifiées du tableur d'origine.

Après avoir utilisé cette fonctionnalité d'export, il suffit d'ouvrir le fichier et d'observer le résultat obtenu.

	A	B	C	D	E	F
1	Nom de l'act	Module	Nom du grou	Terme explo	Couleur de l'activité	
2	TLP	odp	ODP	Red	11	
3	ODP	odp	ODP	Red	12	
4						

Figure 28 - tableur une fois importé dans le programme

On remarque bien que la taille des cellules du tableur après la première ligne n'excède pas 3 caractères. Ce premier traitement montre globalement le fonctionnement de la première version.

Projet Adaptateur

Le programme de la version 1, possède plus de 6500 lignes de code, il est capable de traiter 14 tableurs différents et créer 33 tableurs distincts qui pourront être intégrés dans la Base de données de DbVisualizer.



Figure 29 - Schéma procédure d'un traitement de données pour un tableur

Les classes du programme sont stockées dans 3 packages différents :

- contrôleur
- traitement
- vue

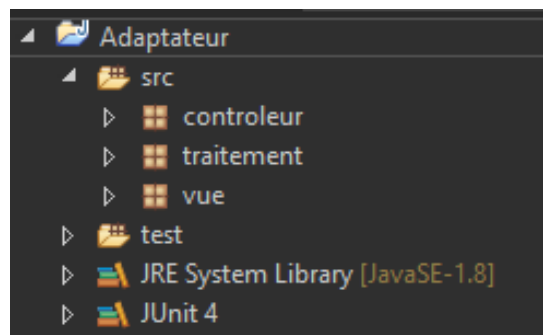


Figure 30 - Package src

Le contrôleur

Le contrôleur va gérer la partie importation et exportation du programme, il permet aussi de générer l'ouverture d'un tableur si l'utilisateur souhaite le vérifier avant de créer les nouveaux fichiers CSV.

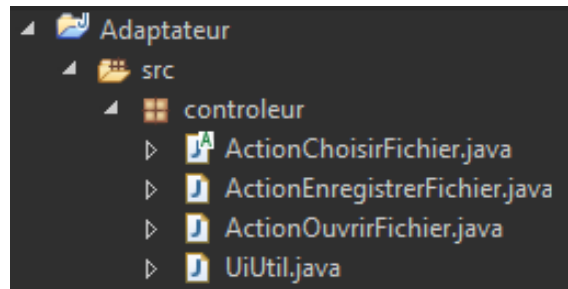


Figure 31 - Package contrôleur

Il est constitué des classes suivantes :

- **ActionChoisirFichier.java**
Cette classe permet de stocker les informations d'un tableur CSV dans une `ArrayList<String>`.
- **ActionEnregistrerFichier.java**
Cette classe va servir à sauvegarder les données traitées d'un tableur vers un autre nouveau tableur.
- **ActionOuvrirFichier.java**
Cette classe permet d'ouvrir un fichier CSV et d'en afficher le contenu pour que l'utilisateur puisse vérifier les informations.
- **UiUtil.java**
Cette classe contient diverses méthodes utiles pour les 3 autres classes du contrôleur, par exemple la méthode *"ouvrirFichier(File fichier)"*.

Le traitement

Le package traitement est le cœur du programme car il se charge de traiter, de vérifier et de modifier les données des 14 différents tableurs. Le package se compose de 14 classes spécifiques qui vont réaliser un traitement pour un tableur en particulier et aussi de 2 classes supports pour les traitements.

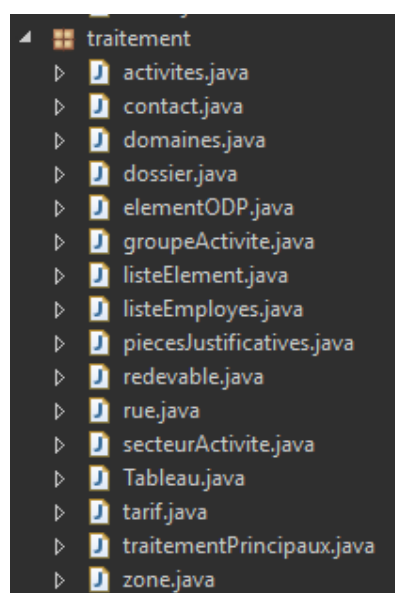


Figure 32 - Package traitement

Les classes spécifiques

Traitement et modification des données

Les 14 classes spécifiques vont stocker les données, modifier et créer les nouveaux tableurs qui seront intégrés dans la nouvelle Base de données. Ils ne peuvent traiter qu'un type de tableau par exemple la classe domaine ne peut traiter que le tableur correspondant aux domaines. En effet, chacune de ces 14 classes va appliquer des traitements différents (voir l'annexe traitements.pdf).

Création des nouveaux tableurs

Les 14 classes chargées de traiter les données de leur tableur vont aussi créer les nouveaux tableurs qui seront intégrés dans la nouvelle Base de données. En effet, si les 14 classes sont utilisées lors d'une reprise de données, 33 nouveaux tableurs CSV seront créés. Cette partie du programme va servir à moduler les données pour qu'elle soient intégrables dans la nouvelle Base de données.

Lors de la création d'un nouveau tableur, le nom ainsi que les colonnes doivent être exactement les mêmes que ceux de la Table dans la Base de données. En effet, s'ils diffèrent, les données ne pourront pas être intégrées.

Pour comprendre le fonctionnement du programme lors de la création d'un nouveau tableur je vais présenter un exemple.

```
public static List<String> creationTableActiviteLangue(List<String> tabActivite,int langue){
    //ici on va créer la première table à intégrer dans la Base de données
    List<String> tabActFin = new ArrayList<String>();

    int NbNomNonVide = traitementPrincipaux.traitementCelluleVide(tabActivite, "Nom de l'activité");
    List<String> tabActNom = traitementPrincipaux.traitementRecupInfo(tabActivite, "Nom de l'activité");
    tabActFin.add("ACT_REF;LAN_REF;ACT_NOM;DCREAT;UCREAT");//Création de la première ligne de la table
    for(int i=0;i<NbNomNonVide;i++){
        String[] colonnetab = new String[5];//la ligne possède 5 colonnes
        String lignetabActFin = "";
        for(int j=0;j<5;j++){
            if(j == 0){//la référence du groupe d'activité qui s'auto incrémente
                colonnetab[j] = ""+(i+1);
            }
            if(j==1){
                colonnetab[j] = ""+langue;
            }
            if(j==2){
                colonnetab[j] = tabActNom.get(i);
            }
            if(j == 3){ //la date d'aujourd'hui dans DCREAT
                Date date = new Date();
                final SimpleDateFormat formater = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
                String dateFormat = formater.format(date);

                colonnetab[j] = ""+dateFormat;
            }
            if (j == 4){//ILTR a créé cette table UCREAT
                colonnetab[j] = "ILTR";
            }
            if(j < colonnetab.length-1){ //On stock les infos dans les différentes lignes
                lignetabActFin += colonnetab[j]+";";
            }
            else{
                lignetabActFin += colonnetab[j];
            }
        }
        tabActFin.add(lignetabActFin); // on ajoute les lignes dans le tableau
    }
    return tabActFin; //on retourne le tableau
}
```

Figure 33 - exemple méthode de création de nouveaux tableurs

Cet exemple montre comment est créée la Table ACTIVITE_LANGUE. J'ai utilisé une méthode qui prend en paramètre l'ArrayList<String> contenant les données traitées et l'entier représentant la langue (selon la convention ILTR, le 1 représente la langue française). Cette méthode va préalablement créer un tableau vide contenant les mêmes noms de colonnes que la Table ACTIVITE_LANGUE de la Base de données. Suite à cela, le tableau vide va recevoir toutes les données traitées dans ses colonnes respectives.

- ACT_REF (1ère colonne) représente la clé primaire de la Table, elle s'auto-incrémente lors de l'ajout d'une ligne.
- LAN_REF (2ème colonne) représente la langue de référence, chaque ligne va recevoir l'entier de la langue entré en paramètre.
- ACT_NOM (3ème colonne) va recevoir les noms des activités du client.
- DCREAT (4ème colonne) correspond à la date à laquelle la ligne a été créée. J'ai utilisé la méthode *"Date()"* pour obtenir la date actuelle ainsi que la méthode *"SimpleDateFormat()"* pour mettre la date sous un certain format.
- UCREAT (5ème colonne) représente l'utilisateur qui a créé la ligne. Dans notre cas c'est toujours la société ILTR qui effectue les reprises de données.

A	B	C	D	E
ACT_REF	LAN_REF	ACT_NOM	DCREAT	UCREAT
1	1	TLPE	#####	ILTR
2	1	ODP	#####	ILTR

Figure 34 - Table ACTIVITE_LANGUE créé par le programme

La classe traitementPrincipaux.java

La classe traitementPrincipaux.java comme la classe UiUtil.java dans le package contrôleur est constituée de méthodes utiles pour les 14 classes de traitements. En effet, la classe possède des méthodes capables de limiter le nombre de caractères par cellules, de vérifier le format d'une date et la modifier mais aussi de récupérer les informations d'une colonne spécifique (voir l'annexe traitements.pds). Toutes les méthodes récurrentes nécessaires pour le traitements des données sont situées dans la classe traitementPrincipaux.java.

Nom du tableur	Nombre de traitements utilisés	Nombre de créations de Tables
groupeActivite.java	1	2
activites.java	5	2
groupeActivite.java	1	2
contact.java	1	2
groupeActivite.java	1	2
secteurActivites.java	3	2
groupeActivite.java	1	2
tarif.java	11	4
groupeActivite.java	1	2
rue.java	5	2
groupeActivite.java	1	2
dossier.java	9	1
groupeActivite.java	1	2
elementODP.java	16	4

La classe Tableau.java

Cette dernière classe est très utile pour les traitements. Elle permet de traiter des tableurs modifiés par les clients. En effet, lors de la saisie de données, certaines colonnes du fichier de reprise de données sont supprimées car elles ne sont pas utilisées. Cette classe va donc recréer les colonnes manquantes et les ajouter pour que le traitement puisse être effectué. Les méthodes qui composent cette classe sont utilisées à chaque début de traitement.

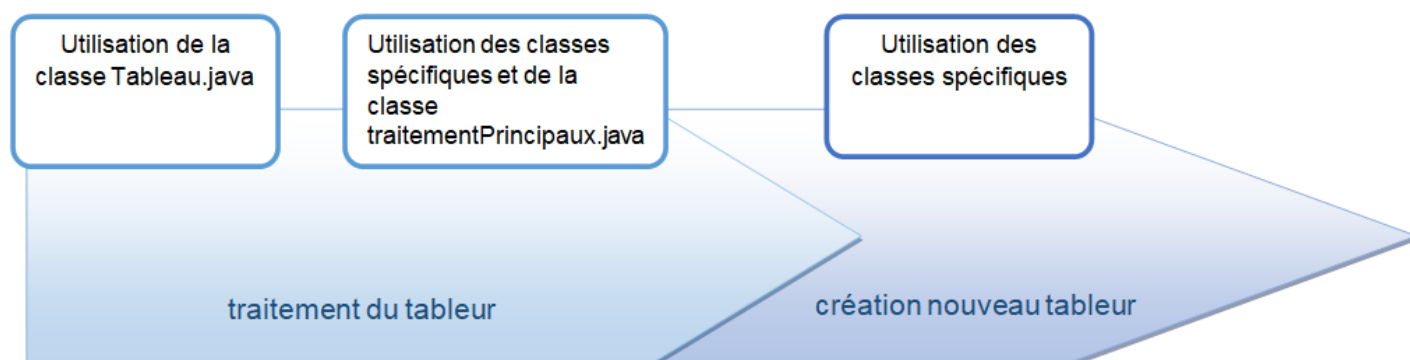


Figure 35 - Schéma procédure du module traitement

La vue

Le package `vue` représente l'IHM du programme. L'IHM va permettre à l'employé d'ILTR d'assurer le bon déroulement du traitement via différents boutons et labels. Pour créer l'IHM du programme, je me suis basé sur le projet "BeAnArtist" de l'année dernière en séparant les parties importantes.

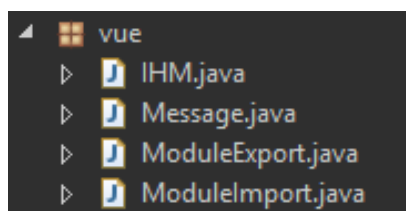


Figure 36 - Package `vue`

Le package est constitué de 4 classes dont une classe principale :

- **IHM.java**, c'est la classe principale du package, elle constitue la fenêtre graphique du programme.
- **Message.java**, cette classe est un panel rattachée à la classe `IHM.java`, elle permet d'afficher les informations utiles pour l'employé ILTR chargé de la reprise de données.
- **ModuleExport.java** cette classe est un panel rattachée à la classe `IHM.java`, elle permet de gérer l'aspect export du programme de la version 1.
- **ModuleImport.java** cette classe est un panel rattachée à la classe `IHM.java`, elle permet de gérer l'aspect import du programme de la version 1.

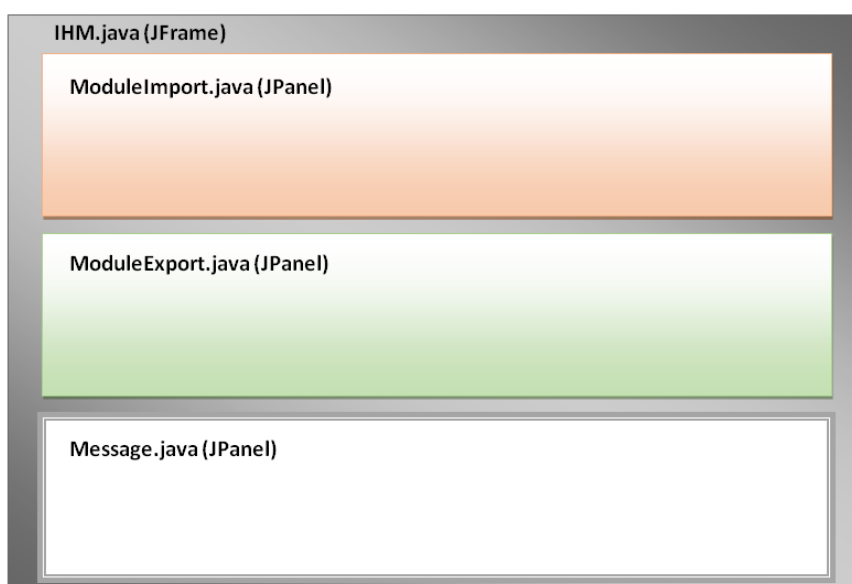


Figure 37 - Schéma du bloc IHM

3.7 - IHM

Afin d'optimiser le temps consacré à une reprise de données, j'ai créé une IHM simple avec peu de boutons et de labels. En effet, plus une interface est chargée plus elle est complexe. J'ai donc agencé l'IHM de sorte que l'employé ILTR puisse distinguer visuellement le processus à suivre pour assurer un bon traitement de données.

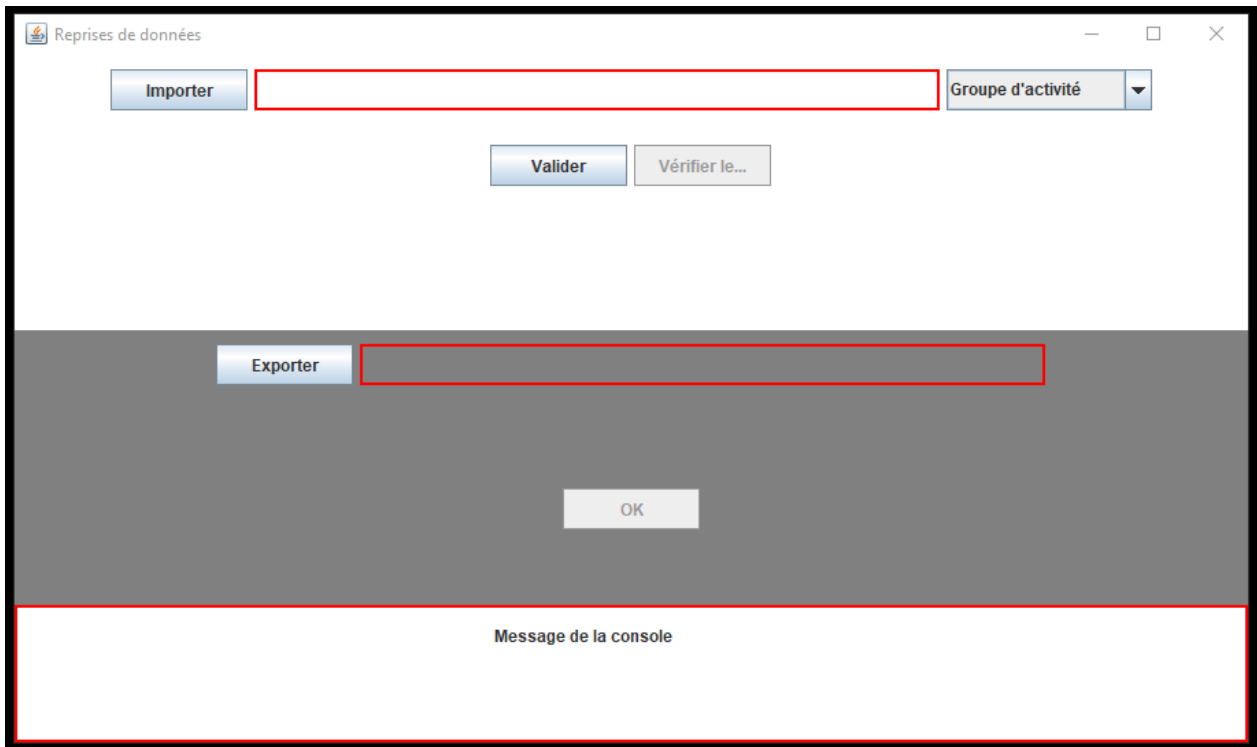


Figure 38 - IHM de la version 1

La première partie en fond blanc correspond au module d'import du programme de la version 1. Dans cette partie de l'IHM, l'utilisateur va importer le tableur CSV à traiter.

Lorsque l'utilisateur clique sur le bouton **Importer**, une autre fenêtre graphique apparaît pour qu'il puisse sélectionner le tableur CSV à traiter. J'ai utilisé la méthode de Java `"JFileChooser()"` afin d'afficher la fenêtre intitulée **Ouvrir** et d'importer le fichier CSV. Si le tableur ne possède pas l'extension CSV, il ne pourra pas être importé.

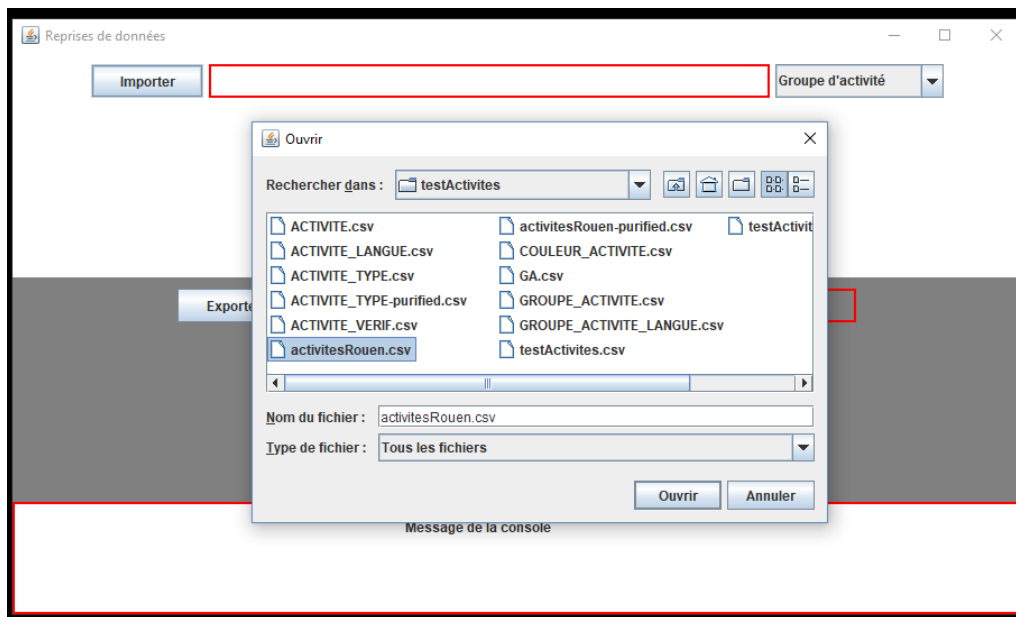


Figure 39 - Importation

Après avoir importer le fichier, l'adresse du fichier apparaît et permet de vérifier quel fichier l'utilisateur à importé.

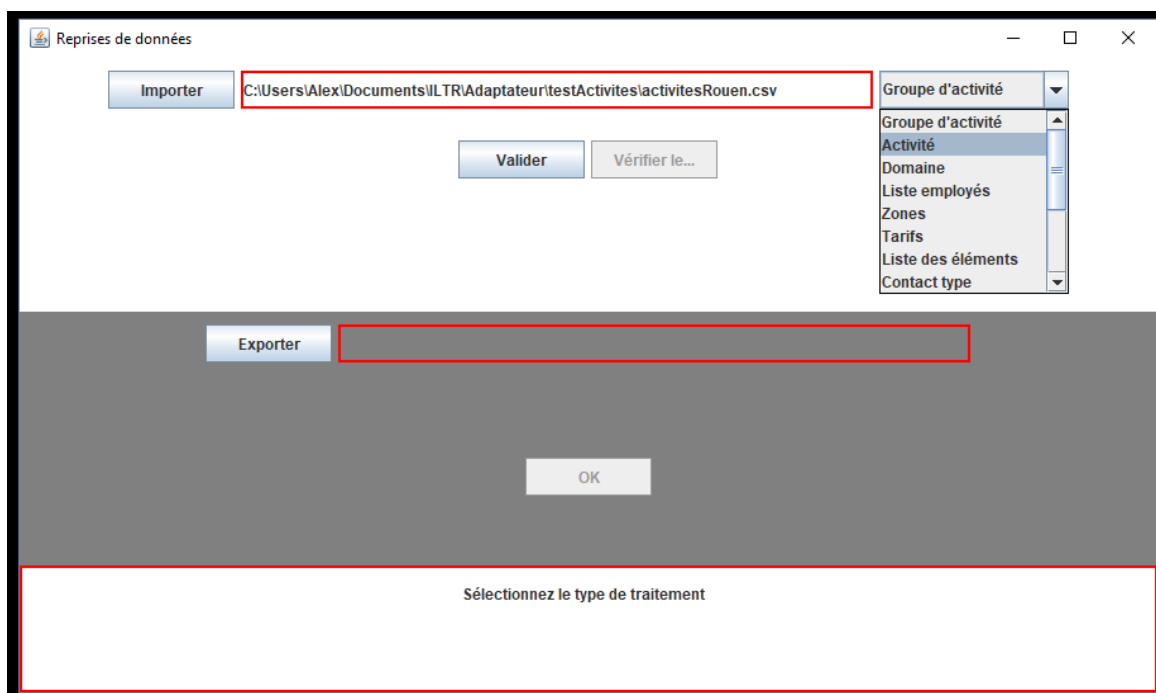


Figure 40 - Choix du type de traitement

L'employé de pôle de maintenance va ensuite choisir quel type de traitement il doit appliquer à son fichier. Dans notre exemple, le tableur est lié aux activités de la ville de Rouen, il faut donc sélectionner l'onglet **Activité**.

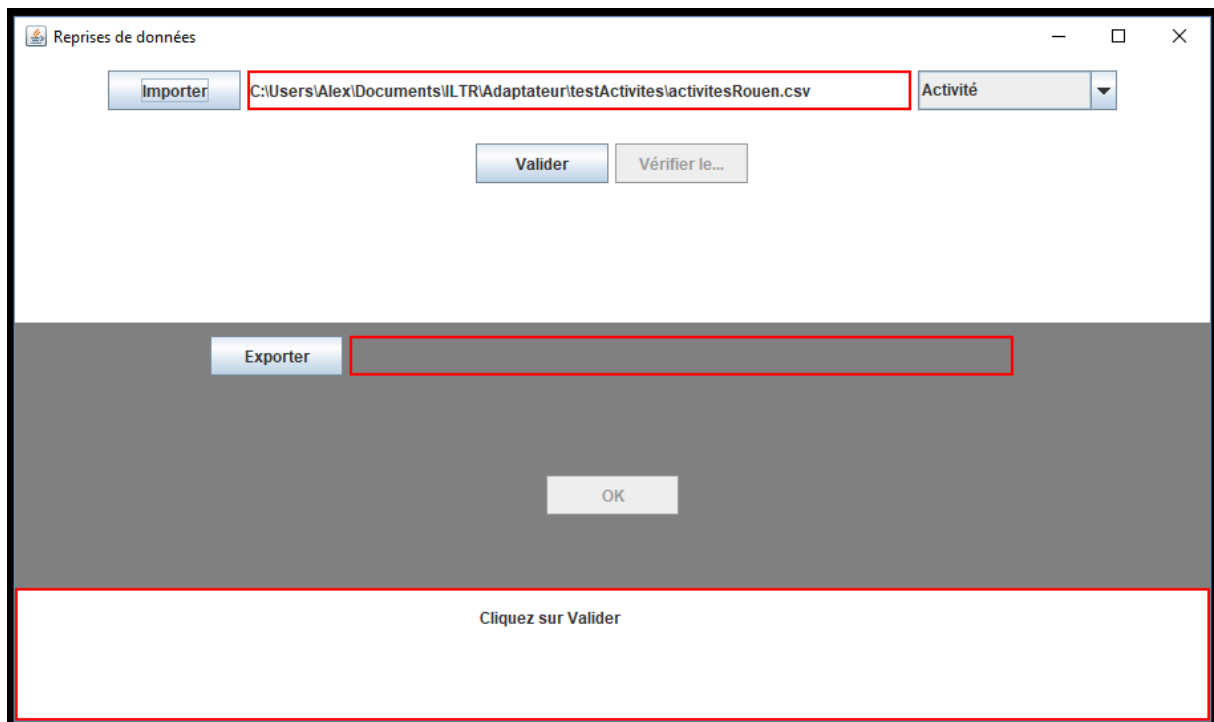


Figure 41 - Mise en place de l'import

L'utilisateur peut ensuite cliquer sur le bouton **Valider** le programme va alors traiter les données du tableur et utiliser la classe spécifique au type du traitement. Dans l'exemple la classe `activites.java` est utilisée et le tableur va subir 5 traitements différents.

Après la validation du traitement, le bouton **Vérifier le fichier** apparaît. L'utilisateur peut donc, s'il le souhaite, vérifier les informations du tableur après l'ensemble du traitement. Lors d'un clic sur ce bouton, un tableur Excel présentant les informations filtrées apparaît. J'ai utilisé la méthode `"Deskop()"` de Java pour cette opération d'ouverture de fichier.

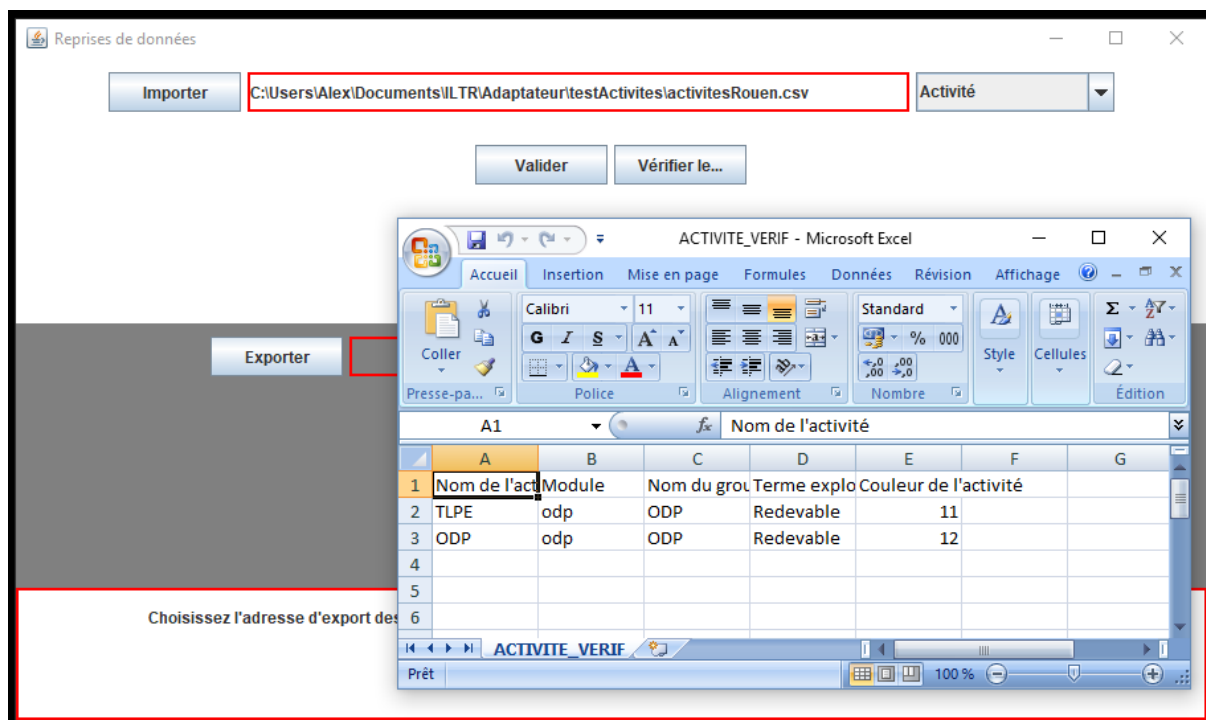


Figure 42 - Vérification du fichier traité

La partie en gris de l'IHM représente le module export. L'employé d'ILTR va utiliser cette partie pour gérer l'adresse d'export des nouveaux tableurs CSV qui pourront être intégrés dans la Base de données sur DbVisualizer.

Lorsque l'utilisateur clique sur le bouton **Exporter**, une fenêtre graphique apparaît. Il peut alors sélectionner le dossier où seront stockés les nouveaux tableurs.

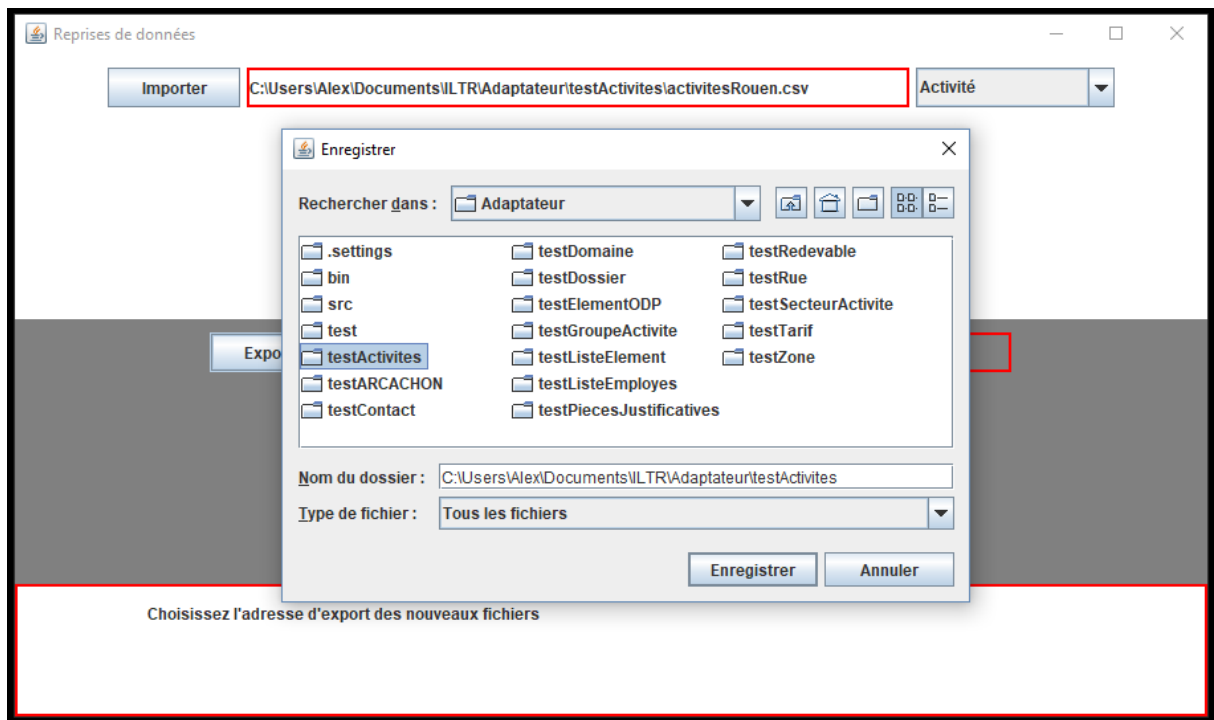


Figure 43 - Exportation

Après avoir choisi l'adresse de destination, le programme l'affiche pour que l'utilisateur puisse la vérifier.

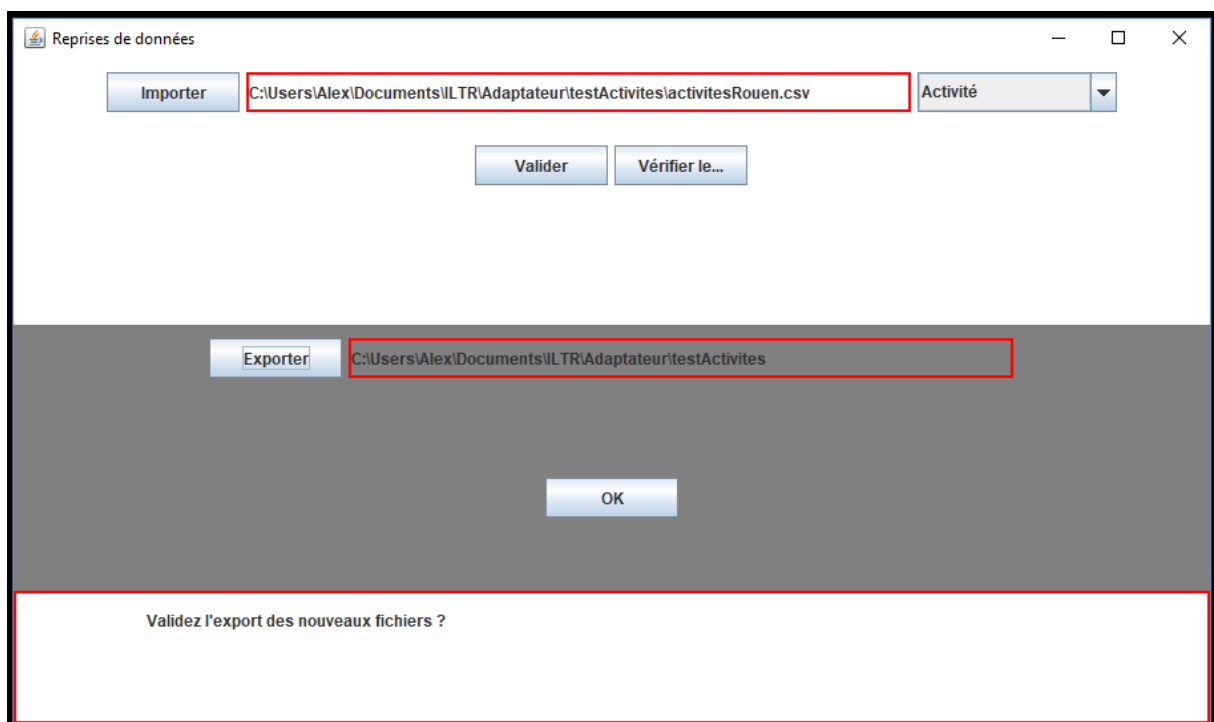


Figure 44 - Mise en place de l'export

Une fois que l'employé d'ILTR a cliqué sur le bouton **OK**, les nouveaux fichiers CSV sont enregistrés.

Ce PC > Documents > ILTR > Adaptateur > testActivites				
	Nom	Modifié le	Type	Taille
de	ACTIVITE	14/11/2017 19:56	Fichier CSV Micro...	1 Ko
015 -	ACTIVITE_LANGUE	14/11/2017 19:56	Fichier CSV Micro...	1 Ko
nts	ACTIVITE_TYPE	31/10/2017 15:08	Fichier CSV Micro...	1 Ko
jemen	ACTIVITE_TYPE-purified	31/10/2017 15:08	Fichier CSV Micro...	1 Ko
	ACTIVITE_VERIF	15/11/2017 11:28	Fichier CSV Micro...	1 Ko
	activitesRouen	14/11/2017 19:13	Fichier CSV Micro...	1 Ko

Figure 45 - fichiers CSV exportés

ACT_REF					
A	B	C	D	E	F
1	ACT_REF	ACT_NUM_CACT	MODUL	DCREAT	UCREAT
2	1	11 odp	#####	ILTR	
3	2	12 odp	#####	ILTR	
4					

ACT_REF					
A	B	C	D	E	F
1	ACT_REF	LAN_REF	ACT_NOM	DCREAT	UCREAT
2	1	1	TLPE	#####	ILTR
3	2	1	ODP	#####	ILTR
4					

Figure 46 - tableurs ACTIVITE et ACTIVITE_LANGUE

Les fichiers CSV, ACTIVITE et ACTIVITE_LANGUE sont créés, ils pourront être intégrés dans la nouvelle Base de données du client via la fonction d'import de DbVisualizer. La liste des activités est stockée dans le programme pour que les tableurs suivants puissent être traités. En effet, la méthode de traitement "remplacerActivite()" chargée de vérifier les noms des activités pourra fonctionner sur les autres tableurs. La liste des activités sera effacée lorsque l'utilisateur fermera le programme.

3.8 - Problèmes rencontrés :

Pendant toute la durée de mon stage technique au sein d'ILTR, j'ai dû faire face à de nombreux problèmes d'origines différentes. En effet, certains problèmes sont apparus avant et pendant le développement de mon logiciel de reprise de données. Suite à leur arrivée, j'ai dû mettre en place des solutions et des méthodes différentes pour pouvoir les régler.

Problème avec le fichier de reprise de données

Le premier problème est survenu dès le début de mon stage, lorsque j'ai étudié le mauvais fichier type de reprise de données. En effet, le mauvais fichier ne possédait que 6 tableurs (les tarifs, les zones, la liste des éléments, les redevables et les dossiers et les éléments ODP) .

J'ai donc étudié le mauvais fichier pendant 3 semaines. J'ai dû refaire la partie conception de mon programme ainsi que le planning d'avancement. Le bon fichier type de reprise de données possède 14 tableurs différents au lieu de 6 et pour chaque tableur il faut créer un traitement particulier. La charge de travail pour le développement du programme a été multipliée par 3 car la plupart des tableurs sont plus complexes.

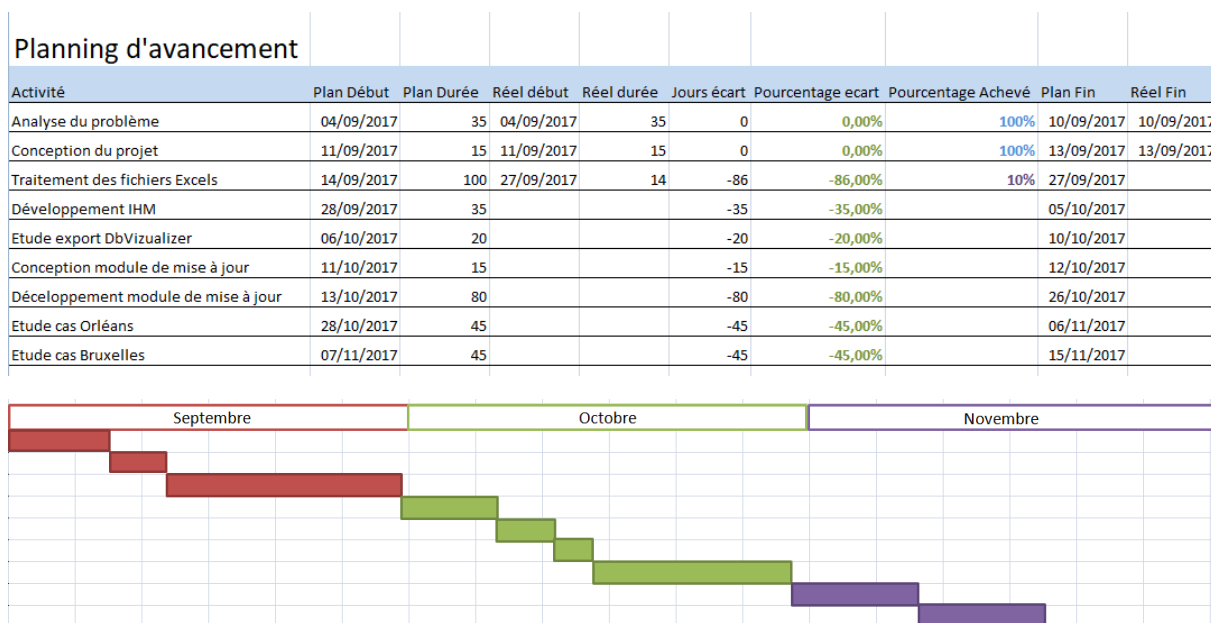


Figure 47 - Planning d'avancement (première version)

Dans l'ancien planning d'avancement je n'avais prévu d'établir qu'une seule version fonctionnelle, permettant de traiter les données, de créer des nouveaux tableurs et de mettre à jour les données.

Cette version complète correspond à la version 2 de mon programme. A cause de la charge de travail plus importante, j'ai dû envisager le cas où je ne pourrais pas terminer la seconde version du programme. Je devais donc à tout prix me concentrer sur les principales fonctionnalités du projet et mettre en place la première version.

Problème avec les colonnes manquantes

Ce second problème est survenu lorsque j'avais fini le développement de la version 1 du programme et que je l'ai testée. Pour vérifier les fonctionnalités de la version 1, mon tuteur m'a demandé de la tester avec le fichier de reprise de données de la ville d'Arcachon. J'ai découvert dans le fichier que certains tableurs étaient manquants ainsi que certaines colonnes. J'ai su que le client les avaient supprimés car ils n'étaient pas utilisés.

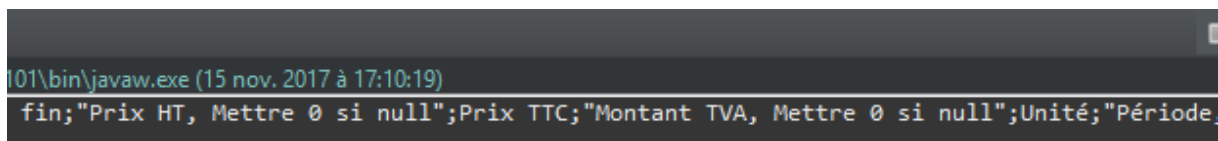
A cette période de test, mon programme fonctionnait avec le fichier Excel type de reprise de données, c'est à dire lorsque tous les tableurs et toutes les colonnes étaient présents. De plus, les employés d'ILTR du pôle de maintenance m'ont renseigné que ce problème était assez récurrent.

J'ai dû revoir entièrement mon programme pour qu'il puisse contourner ce problème de tableurs et de colonnes manquants. Cette période de test sur mon planning devait être courte (45 heures) et à cause de ce problème j'ai pris beaucoup de retard sur le planning que j'avais mis en place.

Problème avec les sauts de lignes

Ce problème est survenu pendant la période de test avec la reprise de données de la ville d'Arcachon. Pour une reprise de données, l'employé d'ILTR doit séparer préalablement les tableurs du fichier Excel d'origine et les convertir en CSV.

Cependant, lors de la copie des données certains sauts de lignes automatiques sont survenus ce qui a modifié la taille du tableur. La taille du tableur à traiter étant modifié, le programme ne pouvait pas filtrer les données du fichier. J'ai dû chercher d'où provenait cette erreur.



```
101\bin\javaw.exe (15 nov. 2017 à 17:10:19)
fin;"Prix HT, Mettre 0 si null";Prix TTC;"Montant TVA, Mettre 0 si null";Unité;"Période
```

Figure 48 - sauts de lignes

Les sauts de lignes sont caractérisées par des " et par des \n (sauts de lignes) cachés. J'ai dû créer une méthode qui permet de "purifier" les tableurs et donc d'enlever ces sauts de lignes avant de les traiter.

4 - Rapport personnel

4.1 - Environnement du stage

Arriver dans une entreprise est toujours difficile, on redoute le premier jour et on se demande si l'on va être bien intégré. J'ai tout de suite été mis à l'aise au sein d'ILTR. Être une vingtaine dans une entreprise aide beaucoup à l'intégration. De plus, l'agencement des locaux était propice à la bonne entente entre les différents employés. Le fait que l'ensemble des salariés travaillent dans une grande pièce facilite la communication.

Tous les vendredis matin, ILTR offre le petit déjeuner et propose à ses employés de se retrouver à la pause du matin. Ce moment convivial permet de penser à autre chose le temps d'un moment et de discuter avec des collègues.

4.2 - Avis sur l'organisation

L'organisation d'ILTR est stricte, à chaque nouvelle demande d'un client, un processus spécifique est enclenché et chaque pôle de la société a un rôle à jouer. Et selon moi c'est une bonne méthode. De plus, l'ensemble des pôles communiquent constamment et de nombreuses réunions plus ou moins importantes sont organisées. J'ai pu assister à 2 assemblées générales où le directeur Yann GOBRAIT présentait les différentes avancées des pôles. La communication chez ILTR est l'un de ces principaux atouts.

Concernant la reprise de données, utiliser un fichier Excel est pratique pour faire gagner du temps au client lors de la saisie des données. Les employés des mairies peuvent directement copier les informations et les stocker dans le fichier de reprise. Cependant, en utilisant un fichier Excel, beaucoup d'erreurs humaines surviennent car les clients ne vérifient pas systématiquement leurs informations.

Pendant toute la durée de mon stage j'ai dû mettre au point un système qui filtre les informations du fichier Excel reçu des clients afin de corriger les erreurs. Mon programme optimise le temps passé sur les reprises de données. Cependant ce type de méthodologie concerne 90% des reprises de données car il s'applique au fichier Excel type qu'ILTR envoie au client.

En effet, certains clients ne veulent pas remplir le fichier de reprise de données et envoient leur propre fichier où sont stockés les informations. Pire encore, certains clients envoient des captures d'écran de ce fichier. Malheureusement, mon programme ne pourra pas couvrir toutes les reprises de données et il serait impossible de mettre en place un outil 100% fiable.

L'utilisation du classeur Excel pour les reprises de données est très avantageux pour le client mais représente un inconvénient pour les employés chargés des reprises de données. Une meilleure solution serait d'envoyer au client un fichier qui indique si une donnée est correcte ou non puis utiliser une version adaptée de mon programme pour que les informations soient filtrées et modulées.

Conclusion

Le stage technique-scientifique est selon moi le meilleur moyen de découvrir le monde de l'entreprise, de mettre en pratique nos connaissances et d'en apprendre de nouvelles.

Pendant 3 mois j'ai pu acquérir de nombreuses compétences dans le domaine du développement. J'ai pu approfondir mes connaissances en langage Java, UML et traitement de données. De plus, j'ai été amené à utiliser des nouveaux outils de gestion pour la Base de données mais aussi pour les plannings. J'ai pu découvrir comment une mairie pouvait gérer sa ville via des centaines d'informations complexes.

J'ai pu intégrer l'organisation d'ILTR et découvrir comment la société fonctionne pour se placer en tant que leader du marché de l'ODP. Ces 3 mois m'ont montrés que pour une entreprise vive, il faut une communication importante entre les différents pôles qui la constitue.

J'ai pris plaisir à travailler en autonomie, à établir un projet face à de réels enjeux. Être confronté à une grande problématique, en rechercher de solutions et choisir la meilleure. Mon activité ne s'arrête pas là car j'ai dû mettre en place un projet et passer par des étapes de gestion de projet, de conception, de développement et de résolution de problèmes.

Certains événements m'ont contrains à redéfinir le projet et à avoir un visuel sur le sujet. J'ai été amené à prendre des décisions et à prioriser les éléments indispensables pour le projet. A cause de ces problèmes, je ne suis pas certain de terminer la seconde version du programme. Je ressens donc un peu de frustration.

Etant intéressé par le domaine de l'énergie et de l'environnement, le monde de l'informatique n'est pas ce que je préfère. Malgré cela, j'ai aimé que l'on m'ai confié une mission et acquérir des connaissances dans le développement pour résoudre un réel problème.