

Rapport d'Alternance

Développement COBOL en informatique de gestion

Année scolaire : 2018 - 2019



Apprenti : Pierre-Antoine PAUWELS

Maître d'apprentissage : Monsieur Julien DESQUENNE

Tuteur : Madame Bénédicte TALON

CONTRÔLE DU DOCUMENT

Historique des versions :

Date de début : 18/10/2018

Auteur : PAUWELS Pierre-Antoine

Date	Version	Principales modifications/remarques
18/10/2018	v1.0	Création du Rapport
18/12/2018	V1.1	Corrections

Distribution :

Pierre-Antoine PAUWELS : 1 exemplaire

Polyèdre : 1 exemplaire

IUT du Littoral Côte d'Opale : 1 exemplaire

État

Travail ☒

Terminé ☐

Validé ☐

Archivage ☐

Sécurité et confidentialité

Non précisé

Remerciements

Je tiens à remercier en premier lieu la société Polyèdre pour avoir accepté ma candidature en tant qu'apprenti en 2^{ème} année de DUT Informatique.

Je remercie également mon maître d'apprentissage Julien Desquenne pour ma formation et mon intégration au sein de l'entreprise, ainsi que son soutien pédagogique durant mon alternance.

Je remercie aussi ma supérieure Emilie Leroy, ainsi que toute l'équipe de Polyèdre pour leur accueil et leur sympathie.

Je remercie ma tutrice Bénédicte Talon pour son professionnalisme et son investissement sans faille dans le suivi de mon alternance, Matthieu Lannoy et Anne Pacou pour l'organisation et la gestion de la filière alternance en informatique à l'IUT du Littoral Côte d'Opale.

Je tiens enfin à remercier l'IUT du Littoral Côte d'Opale pour m'avoir donné la chance de pouvoir prendre la voie de l'alternance au sein de leur établissement.

Sommaire

INTRODUCTION	7
1 PRESENTATION DE L'ENTREPRISE.....	8
1.1 EMERGENCE	8
1.2 POLYEDRE	9
1.3 GPDIS ET MDA	9
1.4 ORGANIGRAMME DE POLYEDRE	10
2 OBJECTIFS ET MISSIONS.....	11
2.1 LES OUTILS	11
2.1.1 <i>Le Cobol</i>	11
2.1.2 <i>Adobe Flash Builder</i>	11
2.1.3 <i>ADABAS et DB2</i>	12
2.1.4 <i>Fonctions POLYFX</i>	12
2.1.5 <i>Secure Netterm</i>	12
2.1.6 <i>Editeur de texte</i>	12
2.1.7 <i>Suite Google</i>	13
2.2 INTEGRATION ET ACCLIMATATION AU MONDE PROFESSIONNEL	14
2.2.1 <i>Intégration dans l'entreprise</i>	14
2.2.2 <i>Premier programme</i>	14
2.3 MES MISSIONS	15
2.3.1 <i>Gestion des types de mail</i>	17
2.3.2 <i>Extension de la gestion des types de mail (version 2.0)</i>	21
2.3.3 <i>Amélioration de l'affichage du catalogue des articles</i>	22
2.3.4 <i>Vérification des IBAN</i>	23
2.3.5 <i>Gestion des bases tarifaires</i>	27
3 CONCLUSION	28
3.1 BILAN PERSONNEL	28
3.2 BILAN GENERAL	28

Table des illustrations

FIGURE 1 : LOGICIEL EMERGENCE, VERSION GRAPHIQUE	8
FIGURE 2 : LOCAUX DE POLYEDRE	9
FIGURE 3 : LOGO GPDIS	9
FIGURE 4 : LOGO MDA	10
FIGURE 5 : ORGANIGRAMME POLYEDRE	10
FIGURE 6 : CODE COBOL	11
FIGURE 7 : LOGO ADOBE FLASH BUILDER.....	11
FIGURE 8 : LOGO DBVISUALIZER.....	12
FIGURE 9 : LOGO NOTEPAD++	12
FIGURE 10 : VISUEL DE TYPEMAIL DANS ADABAS	17
FIGURE 11 : MASQUE TYPMAIL	18
FIGURE 12 : MENU CONTEXTUEL	18
FIGURE 13 : AJOUT D'UN MAIL	19
FIGURE 14 : FONCTIONNEMENT DE LA GESTION DES MAILS CLIENTS.....	19
FIGURE 15 : GESTION DES MAILS (MODIFICATION).....	20
FIGURE 16 : EXTRAIT DU CODE DE LA GESTION DES TYPES DE MAIL	20
FIGURE 17 : GESTION DU CATALOGUE	22
FIGURE 18 : CODE PRINCIPAL DE LA VERIFICATION D'IBAN	23
FIGURE 19 : IBAN CORRECT	24
FIGURE 20 : IBAN INCORRECT	24
FIGURE 21 : ERREUR D'ENTRER IBAN	25
FIGURE 22 : IBAN ALLEMAND.....	25
FIGURE 23 : CORRECTION AUTOMATIQUE IBAN	26

Introduction

Ce document est destiné à la présentation de mon parcours en entreprise durant ma deuxième année en DUT informatique à l'IUT du Littoral Côte d'Opale. Cette deuxième année a été effectuée en alternance auprès de l'entreprise Polyèdre, une PME basée dans le Boulonnais.

Le choix de l'entreprise s'est fondé sur mes préférences au niveau des domaines d'activités dans l'informatique. En effet, la programmation étant un domaine auquel je suis attaché, cela m'a poussé vers l'annonce de l'entreprise. Après un test de compétences générales en informatique et 2 entretiens, je fus intégré à l'entreprise en tant que nouvel apprenti.

Mes missions ne sont pas clairement définies à proprement parler. Elles sont ponctuelles et portent sur l'amélioration ou la création de fonctionnalités sur le logiciel ou à la correction de bug. Un projet plus important, personnel ou collaboratif, me sera donné par la suite.

Ainsi, il sera question dans ce document de relater mon expérience en entreprise lors de ma première année d'apprentissage. Pour cela, nous commencerons par une première partie consacrée à la présentation de l'entreprise puis une seconde partie centrée sur mes missions et leurs réalisations.

1 Présentation de l'entreprise

1.1 Emergence

De 1983 à 1988, Vincent Luttringer créait un progiciel de gestion commerciale et financière nommé Emergence¹ fonctionnant sur système INOS.

Les premiers clients et utilisateurs de ce logiciel seront SA Import Elec, SA AUfray et SA Lesot. Ces derniers font actuellement partie d'une holding nationale sous l'enseigne commerciale "Pulsat".

Le logiciel connaîtra des évolutions techniques comme son adaptation sous-système Unix. A noter que la version originelle du logiciel était en "Texte Brut", autrement dit un affichage de texte blanc sur un fond noir, un mode d'affichage encore répandu notamment dans la grande distribution avec Auchan.

En 2014, Emergence se voit doté d'une nouvelle interface graphique en Flex, ce qui va grandement changer le mode d'utilisation de logiciel. Depuis, le logiciel est en constante amélioration pour répondre aux demandes de ses utilisateurs.

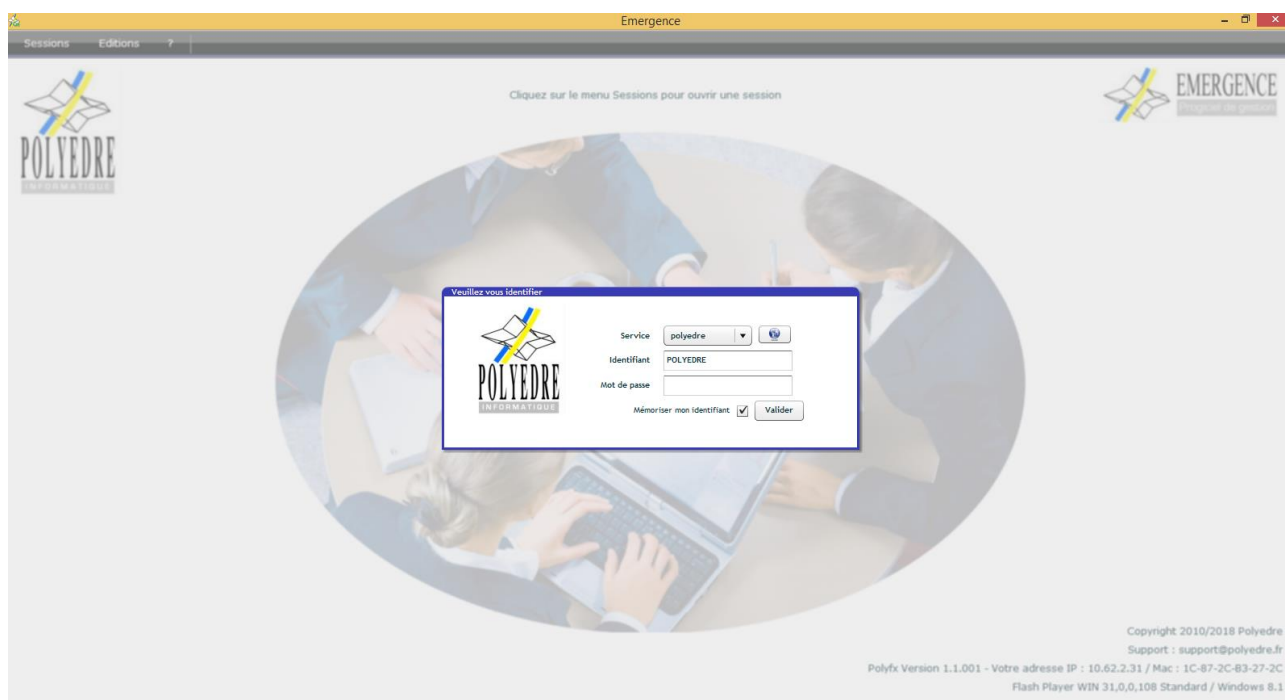


Figure 1 : Logiciel Emergence, version graphique

¹ Emergence est un ERP (Entreprise Ressource Planning)

1.2 Polyèdre

Polyèdre est une entreprise créée par Vincent Luttringer en 1989 suite au début de la commercialisation de son logiciel. Son siège social se trouve à Saint Martin Boulogne (62280), dans le Pas de Calais. L'entreprise se consacre au développement et à la maintenance de son logiciel de gestion Emergence mais aussi à l'édition de technologie de développement avec Rich Client for Cobol.

En 2011, 56% des parts de la société sont rachetées par GPdis, un des clients de Polyèdre. Cette acquisition a changé grandement le mode de travail de l'entreprise. En effet, celle-ci ne travaillera plus par devis mais bien par des demandes formulées par le nouvel actionnaire majoritaire GPdis.

L'année 2016 marque le rachat total de Polyèdre par GPdis. Ainsi, Polyèdre travaillera exclusivement pour le groupe mais tient à garder certains de ses clients historiques comme Sopano, un fabricant d'étiquette, de stickers et de bobine de caisse ou encore Casteleyn, une entreprise spécialisée dans l'emballage du décor et de l'art de table.



Figure 2 : Locaux de Polyèdre

Actuellement, Polyèdre est une Société à responsabilité limitée (SARL) de 10 salariés ayant tous pour mission d'améliorer le progiciel Emergence pour les magasins du groupe ou d'en assurer la maintenance.

1.3 GPdis et MDA

GPdis est un groupement de Plateforme de distribution dans l'électroménager, l'Hi-fi, le son, la vidéo, le numérique, l'abonnement satellite et la téléphonie. GPdis était le numéro 1 en France de l'électroménager à bas prix. Cette position sera perdue lors de son rachat par le groupe MDA.



Figure 3 : logo GPdis

MDA est une société de holding de distributeurs, expert dans l'électroménager, la télévision et le multimédia à prix discount. MDA regroupe actuellement de nombreux autres groupes et enseignes comme GPDIS, PULSAT, ART CUISINE ou encore ANDOM.



Figure 4 : Logo MDA

Le groupe représente actuellement plus de 600 millions de chiffres d'affaires, plus de 1 200 personnes pour 1 000 points de vente et plus de 55 000 références réparties sur 14 plateformes.

MDA est le leader sur le marché français du discount électroménager.

1.4 Organigramme de Polyèdre

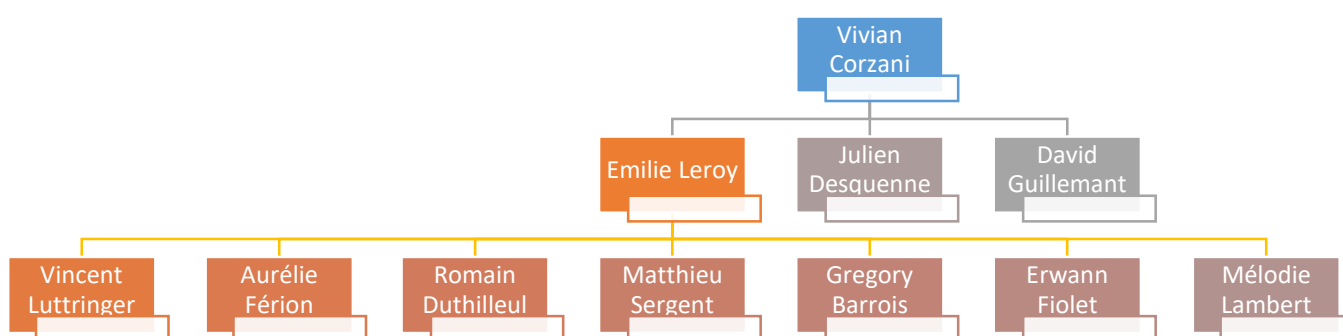


Figure 5 : Organigramme Polyèdre

Vivian Corzani : PDG de MDA

Emilie Leroy : chef de projet Emergence, référente MDA à Polyèdre

Julien Desquenue : pôle recherche et développement

David Guillemant : développeur web (front/back) et administrateur du site internet BtoB de GPdis

Vincent Luttringer : créateur d'Emergence et fondateur de Polyèdre, s'occupe principalement des clients sur l'ancienne gestion commerciale

Aurélié Ferion : support informatique

Mélody Lambert : chargé d'étude et de recherche au niveau décisionnel

Romain Duthilleul : développeur Cobol

Matthieu Sergent : développeur Cobol

Grégory Barrois : développeur Cobol

Erwan Fiolet : développeur Cobol (alternant)

2 Objectifs et missions

Pour la réalisation de mes missions au sein de l'entreprise, plusieurs outils sont mis à ma disposition ainsi qu'à celle des développeurs. Il sera donc question de les identifier et de les présenter dans une première partie. Puis nous continuerons en développant la partie pratique et technique avec mes premiers pas dans l'entreprise et mes premières fonctionnalités en Cobol.

2.1 Les outils

L'entreprise possède des outils appropriés pour le développement des fonctionnalités et la maintenance de leur logiciel Emergence.

2.1.1 Le Cobol

Le COmon Business Oriented Language (COBOL) est un langage de programmation créé en 1959 destiné à la programmation de logiciel de gestion. Le Cobol est un langage qui a su montrer sa fiabilité dans le temps, un point très important pour la justification de son utilisation pour l'ERP. Pour exemple, celui-ci est encore très répandu dans le système bancaire.

```
940 ..... AJOUT-TYPE.
941 .....
942 .....* --- Initialisation du masque ---
943 ..... INITIALIZE ..... STYPMAIL-MD.
944 ..... MOVE LOW-VALUE ..... TO STYPMAIL-SY-FL.
945 ..... MOVE "TYPMAIL" ..... TO SCBFX-MODULE.
946 ..... MOVE "Type mail" ..... TO SCBFX-TITRE.
947 ..... CALL "FX_WINDOW_INIT" USING BY REFERENCE STYPMAIL-MD.
948 ..... MOVE 0 ..... TO W-COMPTeur.
949 ..... MOVE "N" ..... TO STYPMAIL-PC-PNMODIF-VI.
950 .....
951 .....* --- Affichage du masque ---
952 ..... CALL "FX_WINDOW_SHOW" USING BY REFERENCE STYPMAIL-MD.
953 .....
954 .....* --- Paramétrage menu contextuel ---
955 ..... PERFORM ALIM-CONTEX-TYPE-MAIL THRU FIN-ALIM-CONTEX-TYPE-MAIL.
956 .....
957 ..... RECEIVE-AJOUT-TYPE.
958 .....
959 .....* --- Chargement du DG TYPE MAIL ---
960 ..... PERFORM ALIM-DG-TYPE-MAIL THRU FIN-ALIM-DG-TYPE-MAIL.
961 .....
```

Figure 6 : code cobol

Il en existe plusieurs versions : la version libre, la version d'IBM et la version de Microfocus. C'est cette dernière version que nous utilisons pour la programmation d'Emergence. Cette version a été choisie à l'origine car le Cobol libre n'était pas aussi fiable que celle proposée par Microfocus et IBM. Ce qui a fait la différence entre la version de Microfocus et IBM se trouve au niveau des fonctionnalités proposées. C'est pour cela que la version Cobol de Microfocus a été choisie pour Emergence.

Avec l'évolution des technologies, le Cobol fut rejoint par d'autres langages dans le cadre du développement et de l'amélioration du logiciel Emergence, on peut y retrouver le langage C ou encore le XML.



Figure 7 : logo Adobe Flash Builder

2.1.2 Adobe Flash Builder

L'évolution des technologies ainsi que la concurrence ont poussé Polyèdre à constamment améliorer leur logiciel. C'est pourquoi Emergence est passé d'un affichage console (texte blanc sur fond noir) à une interface graphique dont la première version est sortie en 2014. Cette transition vers un modèle graphique a pu se faire grâce à Adobe Flash Builder et à sa technologie Flash. Celle-ci fait appel à du MXML (basé sur du XML) et Action Script 3.0.

Adobe Flash Builder² a donc un environnement de développement qui a permis de développer la version graphique d'Emergence. Celle-ci repose sur des vues (ou masque) qui permettent d'interagir plus facilement avec le logiciel tout en améliorant la qualité visuelle du progiciel.

² IDE développé sur la plateforme Eclipse par Macromedia

2.1.3 ADABAS et DB2

Ces deux logiciels sont des systèmes de gestion de bases de données (SGBDD).

ADABAS (Adaptable DATAbase System) est un système utilisé par de nombreux logiciels de gestion et est le premier SGBDD utilisé par Emergence pour remplacer la gestion de fichier (ancien système de gestion des bases de données en Cobol). Ce système est encore utilisé par de nombreux clients de Polyèdre, même s'il n'y a plus de service de maintenance depuis 2012. C'est pour cela, que Polyèdre tend à migrer vers un autre SGBDD.



Figure 8 : logo DBVisualizer

DB2 est un autre système de gestion de base de données, appartenant ici à IBM. Celui-ci est la nouvelle gestion des bases de données présente sur les serveurs de test et sur le serveur officiel des utilisateurs d'Emergence. DBVisualizer est un logiciel de gestion de bases de données. Celui-ci nous permet de gérer la base de données DB2. C'est par ce logiciel que nous créons les bases de données sur les serveurs de test et le serveur de la production, accueillant l'ERP des utilisateurs.

2.1.4 Fonctions POLYFX

Le Cobol n'est pas le seul langage de programmation utilisé pour le développement de l'ERP. En plus du XML pour le côté graphique de l'ERP, le C est aussi présent lors du développement. En effet, des fonctions propres à Polyèdre ont été écrites en langage C pour faciliter le travail des développeurs. Ces fonctions font office de « Framework » pour l'entreprise permettant aux développeurs de ne plus s'occuper de lier la gestion de base de données en Cobol par exemple. Ils font directement appel à ces fonctions dans le programme.

2.1.5 Secure Netterm

Secure Netterm est un environnement d'émulation nous permettant d'accéder aux différents serveurs du groupe et nous permet aussi de compiler nos programmes et de les analyser avec le debugger. Voici différentes commandes que nous utilisons dans le terminal Secure Netterm :

- `..nougues` → permet de charger l'environnement de développement
- `fx42cob` → permet de compiler un masque
- `cobpc 1` → permet de compiler un programme
- `cobpc 3` → permet compiler un programme pour le debugger
- `cobanimsrv` → permet de lancer le mode debug sur le dernier programme en `cobpc3`.
- `gengf` → permet de compiler une table et de générer les fichiers de gestion correspondant à toutes les requêtes

2.1.6 Editeur de texte

Pour ce qui est de l'outil d'écriture pour le code, toute l'équipe des développeurs code avec Notepad++. Il est vrai que cet éditeur de texte n'est pas des plus réputés auprès des développeurs mais il a le mérite d'être simple, autant dans son utilisation que dans son ergonomie. De plus, le langage Cobol y est reconnu, ainsi que le SQL et le XML, les 3 langages principaux utilisés par Polyèdre.

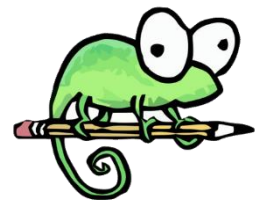


Figure 9 : logo Notepad++

2.1.7 Suite Google

La suite Google est un terme très vaste et générique, nous parlerons ici de Google Drive et de la messagerie électronique Gmail. En effet, ces outils sont utilisés par notre chef de projet pour gérer l'avancement des projets et missions de chacun des développeurs mais aussi être en communication et pouvoir partager des documents importants avec la chaîne hiérarchique de MDA.

Ainsi, on ne trouve pas, à Polyèdre, des logiciels propres à la gestion de projet (exemple : Trello). Ici chaque développeur tient à jour un journal de bord, où il notifie les programmes et tables qu'il crée ou modifie. Ce document est partagé avec le chef de projet qui connaît alors l'avancement de chacun de ses développeurs.

La documentation technique pour des fonctions propres à Polyèdre est d'ailleurs stockée en ligne sur le Drive. On y retrouve aussi divers documents relatifs à l'organisation de l'ERP. De ce fait, tout le monde peut y avoir accès à tout instant et de n'importe où sans problème de version.

2.2 *Intégration et acclimatation au monde professionnel*

L'intégration est un moment très important, car c'est le premier contact avec le monde de l'entreprise, c'est lors de ce moment que l'on accuse la marche entre le monde professionnel et le monde universitaire.

2.2.1 *Intégration dans l'entreprise*

Mon intégration au sein de l'entreprise ne s'est pas faite via mon maître d'apprentissage mais directement avec les développeurs. En effet, mon maître d'apprentissage n'étant pas là le jour de mon arrivée, il a chargé les développeurs de m'introduire auprès de l'équipe et de me former aux pratiques de développement de l'entreprise, des pratiques qui diffèrent d'une programmation personnelle ou scolaire. Ainsi, en plus des codes du langage Cobol³, l'entreprise et les développeurs tendent à développer en respectant certaines règles :

- La structuration du code, rendant la lecture plus facile pour un autre développeur
- Les noms des variables, des fonctions et des programmes répondant à une logique précise
- Le nom, composition et l'emplacement des tables de la base de données.
- Le nom et la charte graphique des masques utilisés par l'entreprise pour la version graphique d'Emergence.

Toutes ces pratiques ne me sont pas familières et représentent un défi de taille car une mauvaise manipulation en local peut produire des dysfonctionnements importants lorsque le programme passe sur le serveur de test ou en production.

Pour me permettre de mener mes missions à bien, l'entreprise m'a fourni un ordinateur portable, contenant tous les outils nécessaires au bon développement des fonctionnalités de l'ERP.

Ainsi, ce fut un premier jour riche en informations et démontrant sans égal mon manque d'expérience professionnelle dans le monde de l'informatique, ou du moins au niveau du développement en entreprise. Lacunes qui se combleront petit à petit, au fur et à mesure de ma pratique du Cobol et de l'utilisation des différents outils de développement mis à ma disposition.

2.2.2 *Premier programme*

Lors de mon premier jour, après ma matinée de formation condensée, j'ai été confié à l'un des développeurs pour qu'il me fasse pratiquer le Cobol et me fasse appliquer les règles de l'entreprise en matière de programmation. Ainsi, mon premier programme fut la réalisation du Pop-up affichant un message d'erreur. Cette fonctionnalité permettant à une caissière de remarquer que l'article qu'elle vient de passer n'a pas été scanné correctement. Ainsi elle doit appuyer sur une touche spécifique pour pouvoir scanner l'article suivant et donc est obligée de constater que l'article précédent n'est pas bien passé.

Une fonctionnalité élémentaire et basique mais regroupant déjà un certain nombre d'éléments m'étant totalement inconnus, ou du moins abstraits, pour ma vision encore très scolaire de l'informatique.

³ Le Cobol est un langage répondant à d'anciennes normes relatives aux cartes perforées, ainsi le code n'est lu qu'entre la 6^{ème} et la 73^{ème} colonne de l'éditeur de texte.

2.3 Mes missions

Cadrage de l'alternance

Mon alternance se place dans un contexte d'apprentissage et d'étude universitaire mais surtout dans le cadre d'une acclimatation au milieu professionnel et de l'entreprise. C'est un milieu vaste et complexe, surtout dans le domaine de l'informatique, secteur en constante évolution et où chacun des domaines s'interconnectent, chose rare ou avec très peu d'ampleur au niveau scolaire où les domaines sont regroupés par matières et n'interagissent que très rarement entre eux.

Ici, dans le cadre de mon alternance et de mes différentes missions, il est question de lier des interfaces graphiques à une base de données grâce à un programme respectant des règles strictes en matière de codage. Tout cela doit répondre à la demande des clients et doit être fait selon cette demande précise. Il n'est pas question d'implanter soit même une fonctionnalité selon nos préférences, sous peine de devoir recommencer le développement. Ainsi, certains programmes nécessitent une phase d'analyse plus ou moins longue selon la complexité de celle-ci.

De plus, il ne faut pas oublier que l'ERP doit répondre à des règles strictes en matière de gestion commerciale. En effet, le logiciel est utilisé par des professionnels qui basent la gestion commerciale de leurs magasins et sociétés sur ce progiciel, celui-ci doit donc être fiable, autant sur le plan technique (minimum de bug informatique) que sur le plan gestion et comptabilité. Cela pousse donc les développeurs, et moi-même, à nous pencher sur le système de gestion et de comptabilité pour connaître les tenants et aboutissants de chacune des fonctions que nous développons.

Orientation de l'alternance

Dans le cadre de mon alternance, ma mission, ou plutôt mon objectif principal, est de contribuer à l'amélioration de l'ERP Emergence en ajoutant des fonctionnalités au progiciel ou encore en corrigeant ou optimisant certaines parties déjà existantes pour améliorer l'utilisation par les utilisateurs. Dans un premier temps, de petites améliorations et fonctionnalités me sont demandées, celles-ci contribuent à mon apprentissage du Cobol ainsi qu'à la manipulation des différents outils proposés par l'entreprise dans le but de me former progressivement.

Méthode d'approche de l'alternance

Compte tenu de la nature de mes missions et de leurs évolutions dans le temps, en termes de densité et de complexité, j'ai pris l'initiative de tenir à jour un journal de bord personnel. Dans celui, j'y inscris mon avancement dans mes différentes missions, ainsi que mon ressenti par rapport à l'alternance. Cela me permet de savoir si je progresse ou non, et me permet ainsi de suivre mon avancée dans mes programmes.

Ce document s'accompagne de fichiers :

- Un Google sheet
- Un Google document
- Un fichier cobpc

Le premier est un fichier que j'ai créé à la demande de mon maître d'apprentissage. Celui-ci regroupe la liste des compétences correspondantes aux différentes technologies de l'entreprise et outils (Cobol, MXML...). Ce document est en partage avec mon maître d'apprentissage et sera destiné, par la suite, aux futurs apprentis.

Le second document est mon manuel personnel de l'utilisation des outils de l'entreprise. En effet, dès que j'apprends à me servir d'un nouvel outil ou dès que l'on m'apprend une nouvelle manipulation avec les serveurs ou le comportement des différents langages, je le note dans ce document qui me sert en quelque sorte de mémo technique personnalisé. Ce document est un complément de la documentation officielle de Polyèdre, plus complète mais moins abordable pour un apprenti, du moins au début de l'alternance.

Le dernier document est un porte folio de petit programme en cobol. Ce document est à mon initiative et a pour but de me servir de « Framework » personnel. En effet, dans celui-ci j'y stocke mes anciens programmes dans lesquels, certaines fonctionnalités peuvent être réutilisées dans d'autres programmes.

Tous ces documents ont pour but d'optimiser, de faciliter au mieux mon alternance au niveau technique et de fournir un suivi pédagogique complet.

2.3.1 Gestion des types de mail

Ma première fonctionnalité, après 2 programmes relativement anecdotiques, est de créer une gestion des types de mail. Cette fonctionnalité a pour objectif d'uniformiser les types de mail disponibles dans l'ERP.

Contexte

Dans le logiciel, jusqu'à présent, lors de l'enregistrement des mails (clients, fournisseurs...), l'utilisateur doit rentrer un mail et lui attribuer un type (facture dématérialisée, comptabilité...). Cependant, comme il n'y a pas de type prédéfini sélectionnable, les utilisateurs remplissent le champ de manière anarchique. On se retrouve alors avec plusieurs mails de même type, classés différents à cause de l'orthographe de leurs types. Cela n'a pas d'impact grave au niveau de la gestion de l'entreprise, cependant cela peut être une fonctionnalité intéressante et relativement simple à mettre en œuvre pour un apprenti novice.

Remarque : je précise que chacune de mes missions me sont attribuées par Emily Leroy.

Période d'adaptation et de formation

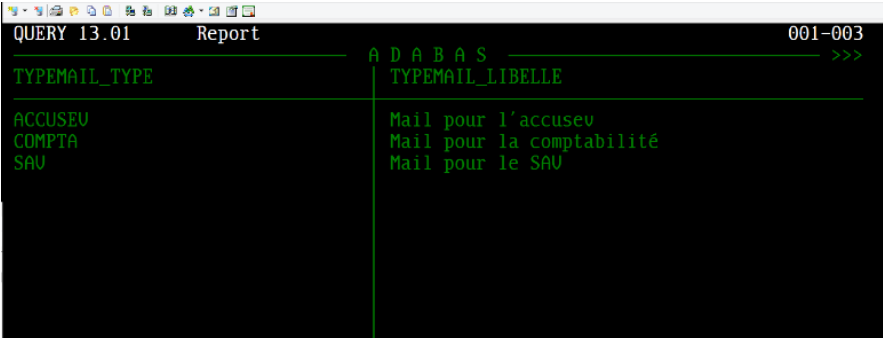
Durant mon alternance, la grande partie de mon temps, du moins durant les premières semaines, sera d'analyser les programmes déjà existants pour m'en approprier le fonctionnement. Ces périodes seront de plus en plus courtes au fur et à mesure de l'évolution de ma compréhension du langage Cobol et du mode de développement des développeurs de l'entreprise. Cette analyse est d'autant plus importante car la grande partie des fonctionnalités sur lesquelles je vais travailler, sont à inclure dans des programmes déjà existants.

Pour cette première fonctionnalité, la durée d'analyse du code a été d'environ 3 jours. Au terme de ces 3 jours, j'ai pu comprendre un grand nombre de choses :

- La compréhension du Cobol.
- La structure du Cobol a utilisé dans l'entreprise.
- Les fonctions basiques mises en place par Polyèdre.
- Les ordres SQL basiques (utilisable via les fonctions).
- La création et la manipulation des masques.
- L'utilisation du terminal pour effectuer les actions basiques de compilation de chaque élément (table, masque, programme).

Une fois ces compétences acquises, je dispose des compétences nécessaires à la réalisation de cette fonctionnalité.

Analyse des différents éléments à développer



The screenshot shows a terminal window with the ADABAS database interface. At the top, it says 'QUERY 13.01' and 'Report'. The table name 'TYPEMAIL' is visible. The columns are 'TYPEMAIL_TYPE' and 'TYPEMAIL_LIBELLE'. The data rows are: 'ACCUSEV' with 'Mail pour l'accusev', 'COMPTA' with 'Mail pour la comptabilité', and 'SAU' with 'Mail pour le SAU'. The terminal has a dark background with green text.

TYPEMAIL_TYPE	TYPEMAIL_LIBELLE
ACCUSEV	Mail pour l'accusev
COMPTA	Mail pour la comptabilité
SAU	Mail pour le SAU

Figure 10 : Visuel de TYPEMAIL dans ADABAS

Cette première fonctionnalité repose sur une table nommée TYPEMAIL qui permet de stocker le type du mail avec son identifiant et un libellé de description. Celle-ci sera gérée par un programme qui fera le lien entre la table et un masque. Le masque servira de support à

l'utilisateur pour y faire les manipulations au niveau des types de mail (ajout, modification, suppression).

Certaines fonctions propres à Polyèdre permettent de simplifier l'utilisation des requêtes en Cobol. En effet, celles-ci permettent en quelques lignes et avec les bons paramètres de faire une requête dans la base grâce à un fichier propre à chaque table regroupant l'ensemble des ordres SQL possibles pour chaque table. Ainsi, à partir d'un simple fichier décrivant la structure de notre table, une commande permet de générer automatiquement ces ordres ainsi que les fichiers pour une migration de la table sur db2 (originellement sur ADABAS). Il sera intéressant de noter que chaque table est placée dans un schéma. Chaque schéma correspond à un environnement de l'ERP. Chacun des environnements permet de filtrer les accès aux utilisateurs. Certains schémas seront accessibles ou du moins utilisables par tous les utilisateurs comme le schéma EMPUBLIC regroupant les tables publiques (exemple : TYPEMAIL). En revanche, d'autres tables comme celles des clients seront propres à chaque entreprise, elles appartiennent donc au schéma USER qui est spécifique à l'entreprise qui se connecte sur Emergence (exemple : BASTADEF).

Une fois notre table créée et nos différents fichiers générés, il ne nous reste plus qu'à déplacer ces derniers au bon endroit dans l'arborescence de l'ERP. Penchons-nous sur cette arborescence. En effet, l'arborescence permet d'organiser et de répartir chaque programme par rapport à leurs fonctions.

- Exemple : le répertoire « vte » regroupe tous les programmes relatifs à la vente.

Ainsi, l'ERP repose sur une organisation structurée de chacun de ses éléments. Cela permet notamment d'éviter les problèmes de reconnaissance des fichiers lors de la migration sur le serveur de test ou encore sur le serveur de la production.

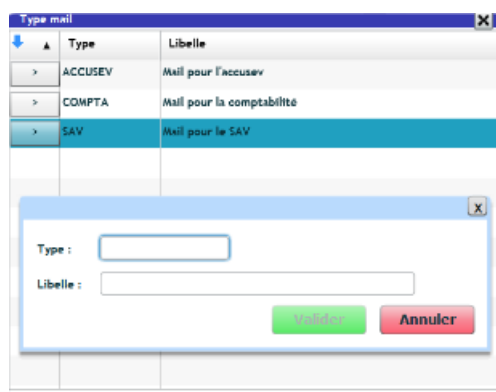


Figure 11 : Masque TYPMAIL

Pour terminer, il faudra créer un masque qui permettra à l'utilisateur d'interagir avec la base de données. Pour cela, il suffira de le créer, avec Flash Builder, de l'envoyer sur le serveur de développement et de le compiler pour que le programme puisse interagir avec lui. Ce masque comporte ce que l'on appelle un Datagrid, un tableau permettant d'afficher les données de chaque attribut de la table ligne par ligne et une fenêtre paramétrable en fonction des actions demandées par un menu contextuel appelé grâce à un clic droit sur le Datagrid. Ce menu permet à l'utilisateur d'ajouter, de modifier et de supprimer un type mail.

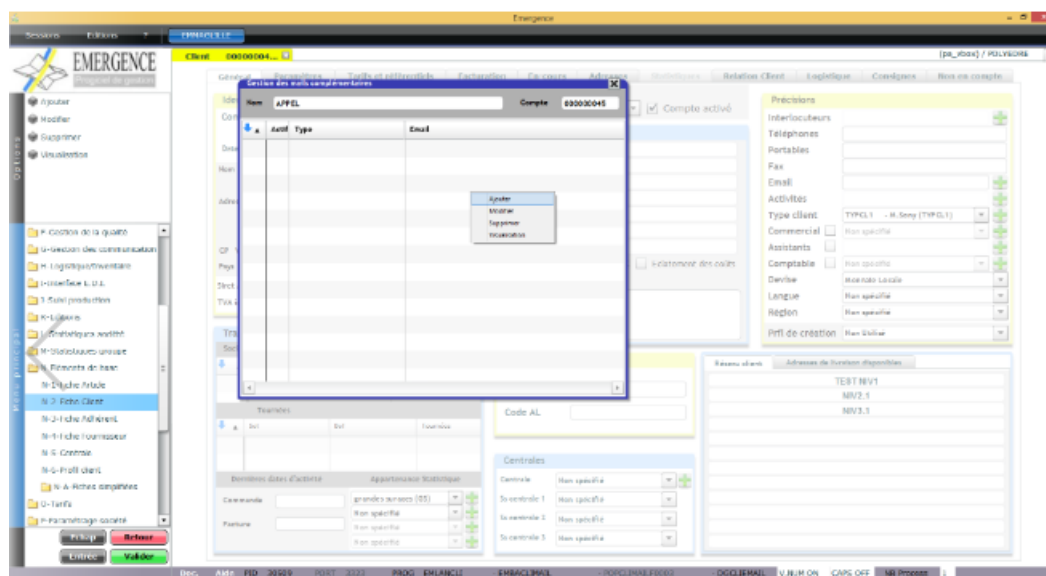


Figure 12 : Menu contextuel

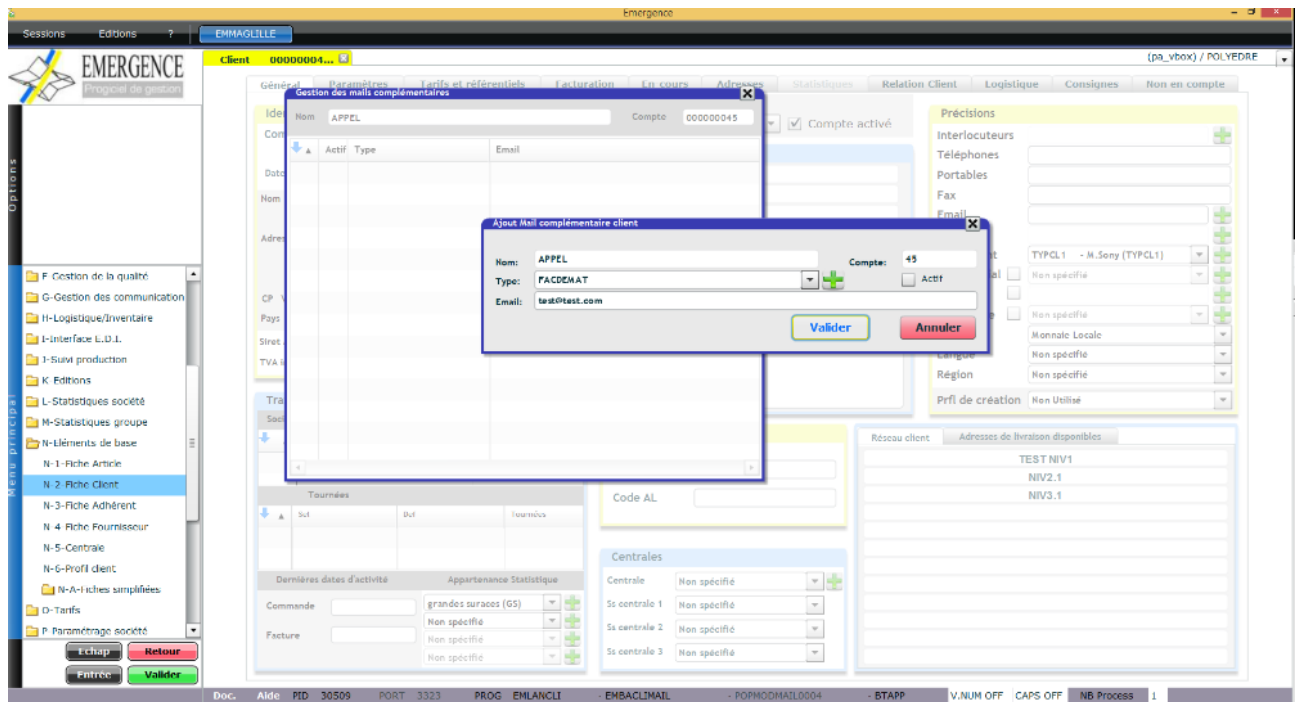


Figure 13 : Ajout d'un mail

Sur la figure ci-dessus, on peut observer l'ajout d'un mail dans la base client. On remarquera la petite croix verte qui permet d'accéder à la gestion des types de mail (voir ci-dessous).

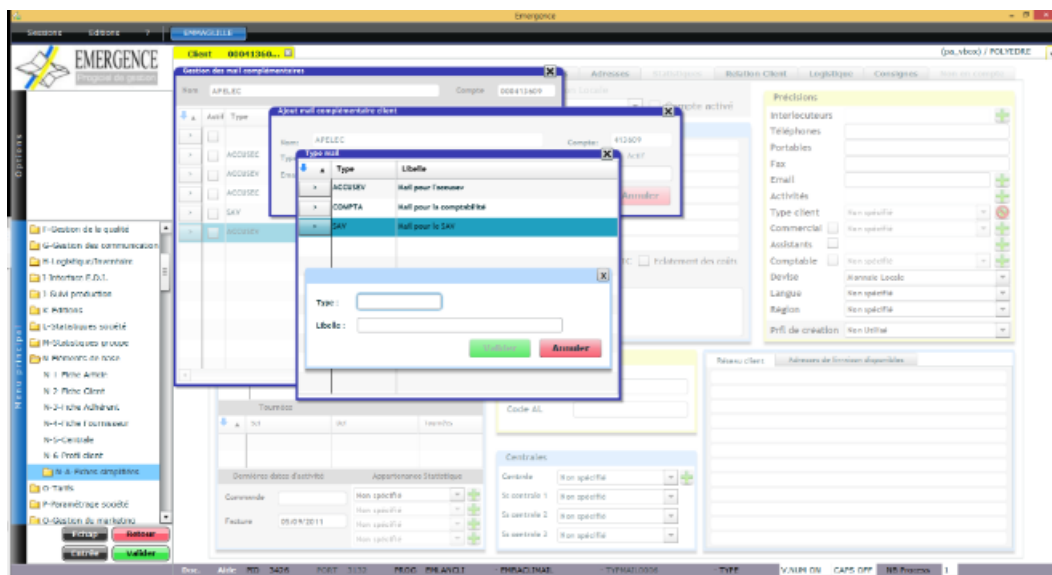


Figure 14 : Fonctionnement de la gestion des mails clients

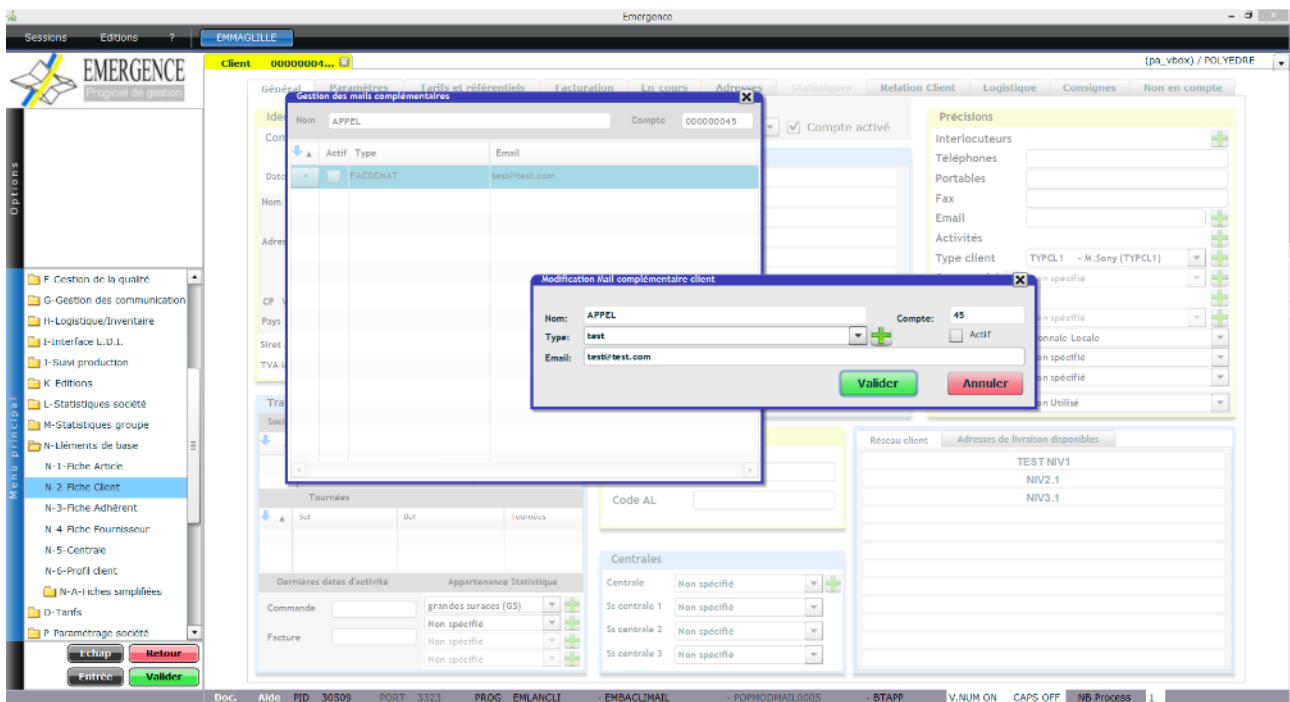


Figure 15 : Gestion des mails (modification)

```

* --- Configuration type mail ---
*
ABOUT TYPE.

* --- Initialisation du masque ---
INITIALIZE STYPMAIL-MD.
MOVE LOW-VALUE TO STYPMAIL-SY-FL.
MOVE "IYPMAIL" TO SCBFX-MODULE.
MOVE "Type mail" TO SCBFX-TITRE.
CALL "IX_WINDOW_INIT" USING BY REFERENCE STYPMAIL-MD.
MOVE 0 TO W-COMPTEUR.
MOVE "N" TO STYPMAIL-PC-PNMODII-VI.

* --- Affichage du masque ---
CALL "FX_WINDOW_SHOW" USING BY REFERENCE STYPMAIL-MD.

* --- Paramétrage menu contextuel ---
PERFORM ALIM-CONTEX-TYPE-MAIL THRU FIN-ALIM-CONTEX-TYPE-MAIL.
RECEIVE-AJOUT-TYPE.

* --- Chargement du DG TYPE MAIL ---
PERFORM ALIM-DG-TYPE-MAIL THRU FIN-ALIM-DG-TYPE-MAIL.

* --- Attente d'ordre de l'utilisateur ---
CALL "IX_WINDOW_RECEIVE" USING BY REFERENCE STYPMAIL-MD.

* --- Petite croix -> retour en arrière
IF SCBFX-EVENEMENT = SCBFX-ANNULLER

```

Figure 16 : Extrait du code de la gestion des types de mail

Bilan de ce programme

Ce premier programme m'a permis de me lancer d'avantage et de manière plus concrète dans l'apprentissage en permettant de développer une fonctionnalité utilisable par l'entreprise et ses clients. Cela apporte une certaine satisfaction de créer un programme qui servira dans la vie professionnelle mais cela importe aussi une fonctionnalité optimale du programme car si celui-ci ne fonctionne pas ou de manière aléatoire, il sera question de revenir dessus pour corriger les bugs de celui-ci pour que l'utilisateur puisse l'utiliser. Cela incombe de prévoir les différentes erreurs possibles de l'utilisateur, une approche qui reste très sommaire dans le circuit scolaire mais indispensable dans le milieu professionnel.

2.3.2 Extension de la gestion des types de mail (version 2.0)

Suite à mon dernier programme, ma chef de projet décide d'améliorer ma fonctionnalité en profitant que je suis encore disposé à connaître les différentes fonctionnalités de mon programme.

Ainsi, dans la version 2.0 de ma fonctionnalité, la finalité est de créer un type de mail par défaut pour chaque nouveau utilisateur dès l'ouverture du programme : un type pour l'envoi de factures dématérialisées (FACDEMAT).

Réalisation technique de l'insertion automatique

Cette fonctionnalité passe par le parcours de la table TYPEMAIL à la recherche du type FACEDMAT. Si le programme ne trouve rien, il l'ajoute automatiquement à la table et l'affiche en premier dans le cadre réservé au type de mail sur le masque. Une modification dans le programme assez mineure en soit mais qui passe par un apprentissage, parfois long en raison de mon niveau de compétence en cobol.

Migration de la fonctionnalité

Compte tenu de la praticité de cette fonctionnalité, ma chef de projet m'a demandé d'ajouter celle-ci à la gestion des mails des fournisseurs. Cela implique donc d'adapter le code à un autre programme, ainsi qu'à un autre masque, celui des fournisseurs.

Export sur serveur de test

Une fois la fonctionnalité implantée pour les fournisseurs, ma chef de projet m'a demandé d'envoyer les programmes (clients et fournisseurs) sur un serveur de test. Ce serveur possède un autre environnement différent de celui de notre serveur de développement. En effet, les serveurs de test fonctionnent avec DB2, l'autre système de gestion de base de données utilisé par Polyèdre et le système présent en production. Ainsi, sur ces serveurs, les programmes sont dans les mêmes conditions que sur la production. Il est donc nécessaire de tester nos programmes sur ceux-ci. La migration sur le serveur de test suit une procédure précise pour répondre à la norme de répartition des différents programmes, tables et masques.

En effet, il faut importer nos tables, mais dans une version DB2, dont les fichiers de création qui sont créés automatiquement mais qu'il faut exécuter avec le logiciel DBVisualizer. Le logiciel se chargera de créer la table sur le serveur (en faisant attention au schéma dans lequel est créée la table), il faudra par la suite importer les différents fichiers (structure de la table, ordre SQL). Il en sera de même pour le ou les masques utilisés comme pour les programmes qu'il faudra recompiler. Une fois toutes ces manipulations effectuées, il sera question de tester nos programmes pour savoir si ceux-ci fonctionnent bien dans ce nouvel environnement.

C'est d'ailleurs durant ces phases de tests que je me suis rendu compte de certains bugs que j'ai dû corrigés rapidement. Une fois ces bugs corrigés, les programmes restent en phase de test tant que certaines personnes (notamment ma chef de projet) ne les ont pas testés.

2.3.3 Amélioration de l'affichage du catalogue des articles

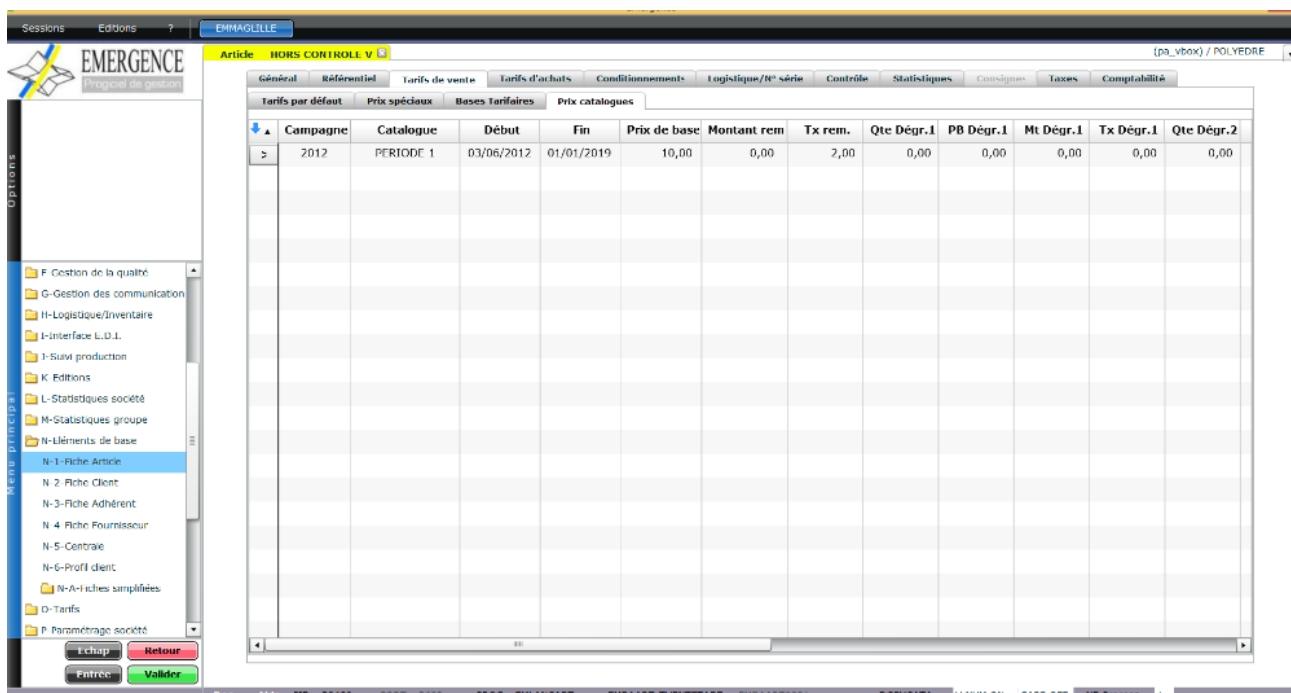
Une fois le programme de gestion des types de mail placé sur le serveur de test, ma prochaine mission fut une autre fonctionnalité mineure mais permettant d'optimiser le logiciel.

Contexte

Le logiciel étant assez gourmand en ressources et les machines sur lesquelles il tourne n'étant pas très puissantes, il est question d'optimiser le code et l'utilisation de l'ERP dès que possible. C'est dans ce contexte que ma future mission consistera à gérer l'affichage des catalogues. En effet, l'affichage des catalogues parcourt de grandes tables en base de données, dont certains éléments possèdent des informations « périmées », comme une date de promotion expirée. Cela coûte des ressources et ralentit le logiciel pour son utilisateur alors que certaines informations ne lui sont plus utiles.

Nouveautés techniques

Pour pouvoir gérer des fonctions en relation avec la gestion de date, il faut faire appel à une fonction permettant de récupérer la date actuelle. Il suffit alors de la comparer avec la date inscrite dans la table. Cependant, pour faciliter les opérations de contrôle sur la date, il sera question ici d'utiliser une structure de donnée capable de décomposer la date en année, mois et jour. Cela rend plus intuitif le contrôle de péremption.



	Campagne	Catalogue	Début	Fin	Prix de base	Montant rem	Tx rem.	Qty Degr.1	PB Degr.1	Mt Degr.1	Tx Degr.1	Qty Degr.2
>	2012	PERIODE 1	03/06/2012	01/01/2019	10,00	0,00	2,00	0,00	0,00	0,00	0,00	0,00

Figure 17 : gestion du catalogue

Sur le programme ci-dessus, on a trié les articles affichés pour ne garder que ceux n'étant pas périmés dans leur promotion (ici 01/01/2019). D'autres articles étaient présents avant dans le DataGrid mais grâce à la fonctionnalité, ils n'apparaissent plus car leurs promotions sont terminées.

2.3.4 Vérification des IBAN

Dans le cadre de cette fonctionnalité, il n'est pas seulement question de parler ici de programmation ou de simple logique de gestion et d'interaction avec une base de données regroupant des mails. Ici, il sera question de débiter par l'apprentissage d'un système de gestion et plus précisément un système de contrôle basé sur la cryptographie.

Etude de la structure des IBAN

Pour pouvoir développer cette fonctionnalité et en retirer une fiabilité, il est tout d'abord question d'étudier la structure et les méthodes de vérification des IBAN utilisées dans le système bancaire ou tout autre établissement ou structure ayant besoin de cette fonctionnalité.

La méthode de vérification se base sur l'utilisation de la cryptographie. En effet, il faut savoir que le système des IBAN est un système d'identité bancaire international, permettant les virements internationaux plus facilement tout en limitant les risques d'erreur (recopie, fraude...). L'IBAN a une taille variant de 18 à 34 caractères, chiffres ou lettres, variant selon les pays. Sur l'IBAN, ce trouve le code pays (exemple : France -> FR), ce code est suivi de la clé de vérification d'IBAN⁴ (souvent 76 pour la France). Ainsi, prenons l'IBAN suivant : FR76 3007 6028 7815 0310 0080 041.

On place ce que l'on appelle le « Code Pays + clé IBAN » ici FR76 à la fin de l'IBAN, ce qui nous donne : 3007 6028 7815 0310 0080 041 FR76. Ensuite, on remplace toutes les lettres par des chiffres en respectant le tableau ci-après, ce qui nous donne : 3007 6028 7815 0310 0080 041 152776.

A = 10	B = 11	C = 12	D = 13	E = 14	F = 15	G = 16	H+ = 17
I = 18	J = 19	K = 20	L = 21	M = 22	N = 23	O = 24	P = 25
Q = 26	R = 27	S = 28	T = 29	U = 30	V = 31	W = 32	X = 33
Y = 34	Z = 35						

Il ne suffit plus qu'à exécuter un modulo de 97 sur ce chiffre, si l'on tombe sur 1, l'IBAN est correct, sinon, il est incorrect tout simplement.

Expression technique

L'élaboration de l'algorithme n'est pas difficile, il suffit de manipuler l'IBAN en suivant le processus de vérification. L'inconvénient est le découpage imposé par la technologie du nombre final pour le modulo car un chiffre ne peut pas dépasser 9 caractères en cobol. Il faut donc faire des paquets de 9 caractères, et ne pas oublier de prendre le modulo du calcul suivant et le rajouter en avant du paquet précédent.

Compte tenu de la sensibilité de cette fonctionnalité reposant sur le système de paiement, il est aussi question de gérer les erreurs de l'utilisateur, comme le fait qu'il entre des minuscules pour les lettres au lieu des majuscules, cela perturbe le calcul et l'enregistrement en base de données.

```
VERIF-IBAN.
  MOVE "0" ..... TO W-OK.
  MOVE 0 ..... TO W-LONG-IBAN.
  MOVE SPOPFICHIBAN-TI-IBAN-TE TO W-IBAN.
  MOVE "ABCDEFGHIJKLMNPKRSTUVWXYZ" TO W-CODE-IBAN.
  INSPECT W-IBAN TALLYING W-LONG-IBAN
  ..... FOR CHARACTERS BEFORE SPACE.

* --- Test structure IBAN ---
  IF W-LONG-IBAN <14 OR W-LONG-IBAN >34
    MOVE "N" ..... TO W-OK
    MOVE "IBAN incorrect" ..... TO MSG-TEXTE
    PERFORM MSG-ERREUR THRU FIN-MSG-ERREUR
    GO TO FIN-VERIF-IBAN
  END-IF.

* --- Passage du code pays + clé en fin d'IBAN ---
  SUBTRACT 4 FROM W-LONG-IBAN GIVING W-LONG-BBAN.
  MOVE W-IBAN(1:4) ..... TO W-CODEPAYS.
  MOVE W-IBAN(5:W-LONG-BBAN) TO W-BBAN.
  STRING W-BBAN ..... DELIMITED BY SPACE
  W-CODEPAYS ..... DELIMITED BY SIZE
  ..... INTO W-IBAN.

* --- Conversion de L'IBAN en full chiffre ---
  PERFORM CONV-IBAN THRU FIN-CONV-IBAN.

* --- Vérification de L'IBAN par Le calcul du modulo ---
  PERFORM VERIF-MODULO-IBAN THRU FIN-VERIF-MODULO-IBAN.
  IF W-OK NOT = "0"
    MOVE "IBAN non valide" ..... TO MSG-TEXTE
    PERFORM MSG-ERREUR THRU FIN-MSG-ERREUR
  END-IF.

FIN-VERIF-IBAN.
EXIT.
```

Figure 18 : Code principal de la vérification d'IBAN

⁴ Cette clé se vérifie aussi, mais nous ne traiterons que de la vérification de l'IBAN en général, sachant que la vérification de celui-ci vérifie la clé aussi.

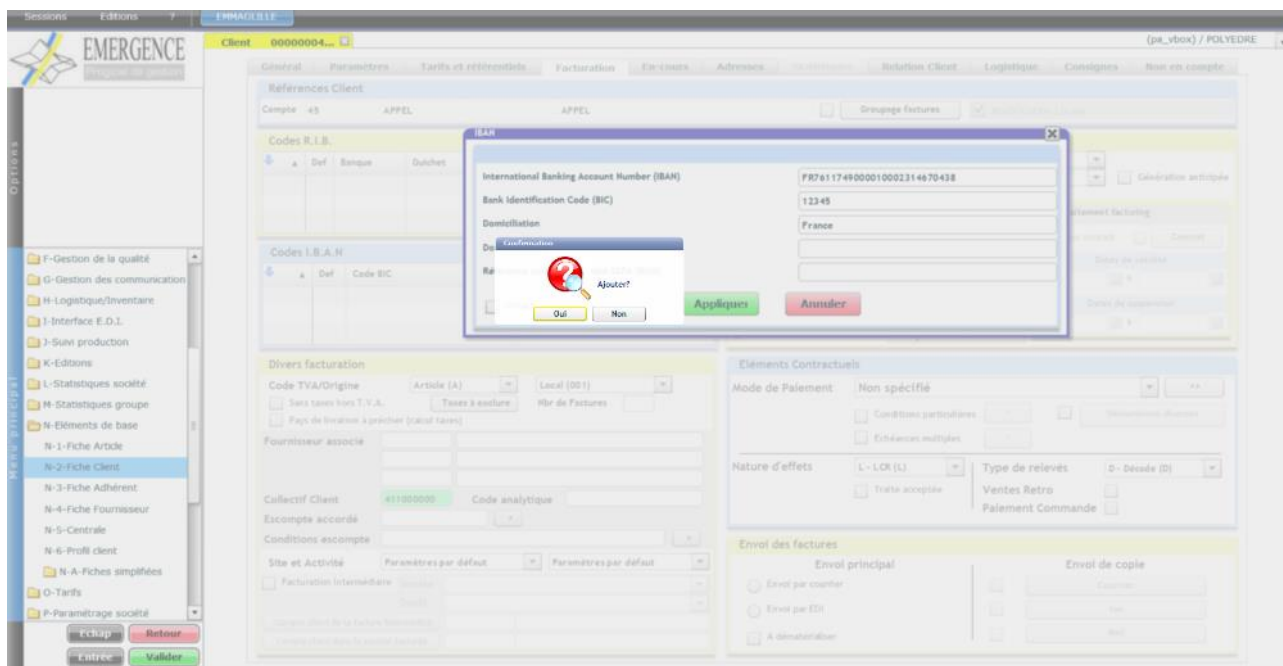


Figure 19 : IBAN correct

Dans la manipulation ci-dessus, en appuyant sur le bouton valider, une pop-up apparaît pour que l'utilisateur puisse confirmer son action d'enregistrer l'IBAN qu'il vient d'entrer. A ce moment, la vérification d'IBAN s'est déjà effectuée et pour l'IBAN rentré, celui-ci est correct, contrairement à celui ci-dessous.

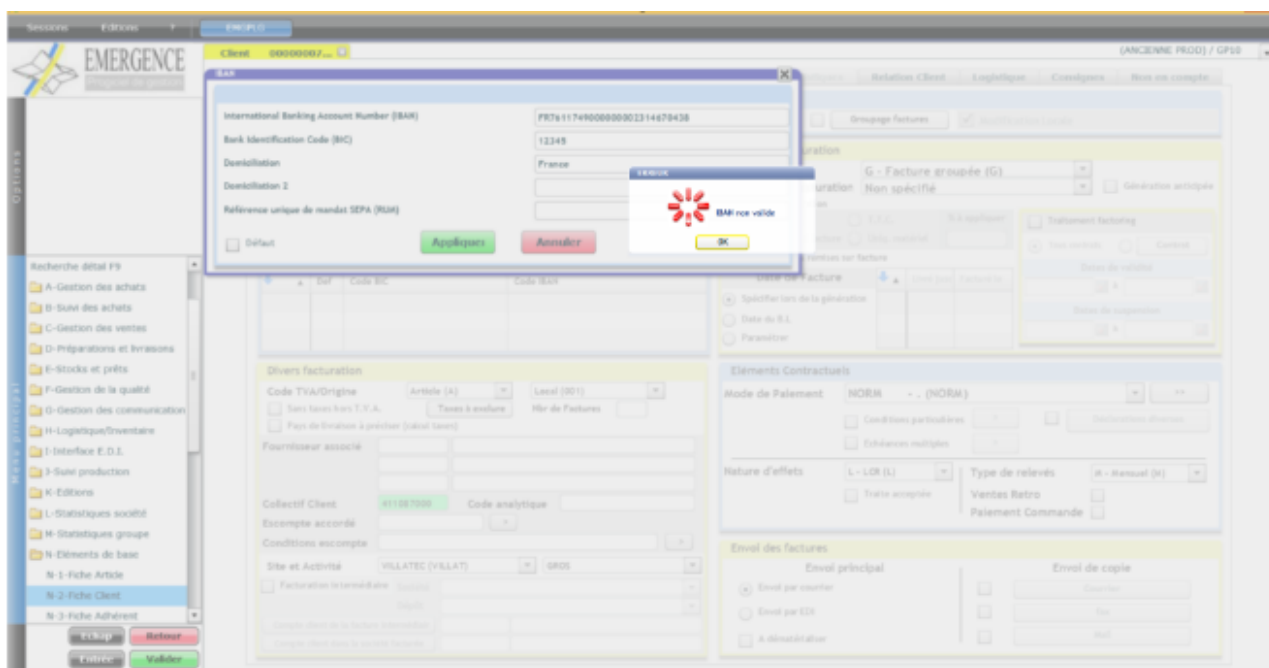


Figure 20 : IBAN incorrect

Ici, l'IBAN a été vérifié et est incorrect, cela renvoie un message à l'utilisateur pour lui signaler que l'IBAN est incorrect. Il n'est donc pas enregistré en base de données et l'utilisateur doit corriger sa faute pour pouvoir l'enregistrer. Bien entendu, celui-ci sera de nouveau vérifié par la suite.

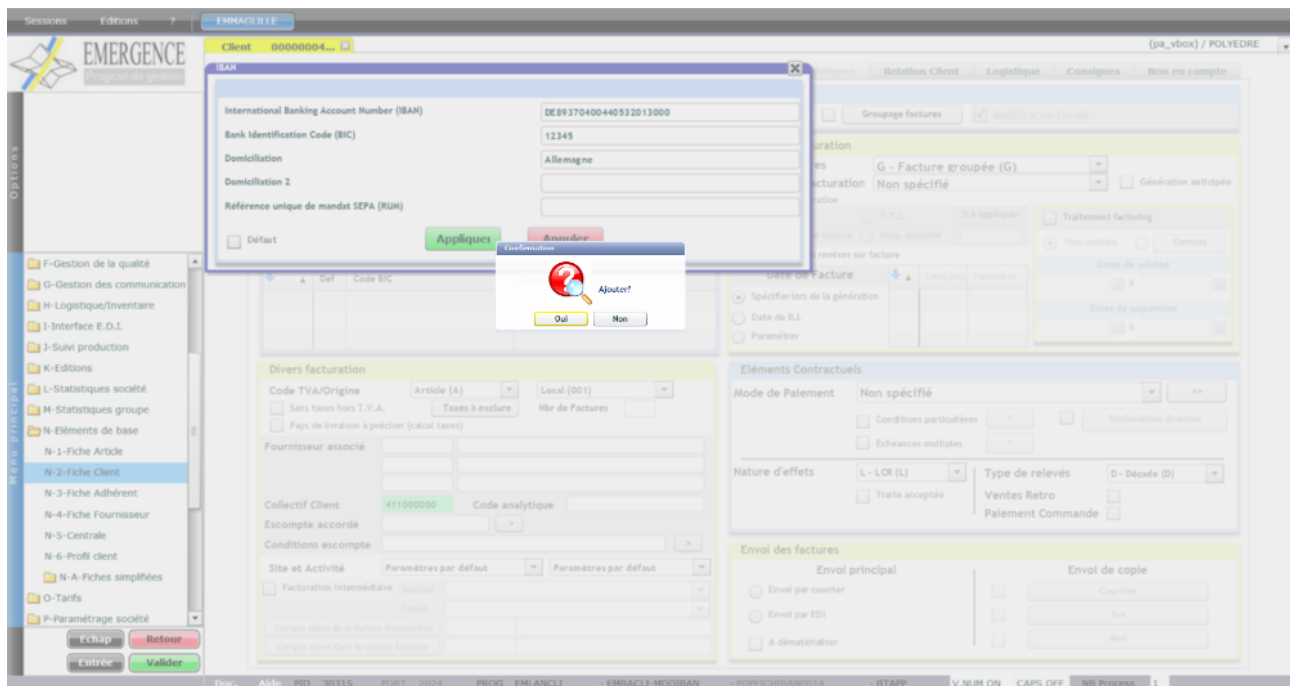


Figure 22 : IBAN allemand

Bien entendu, comme le système des IBAN est un système international, la fonctionnalité est capable de contrôler tous les IBAN internationaux.

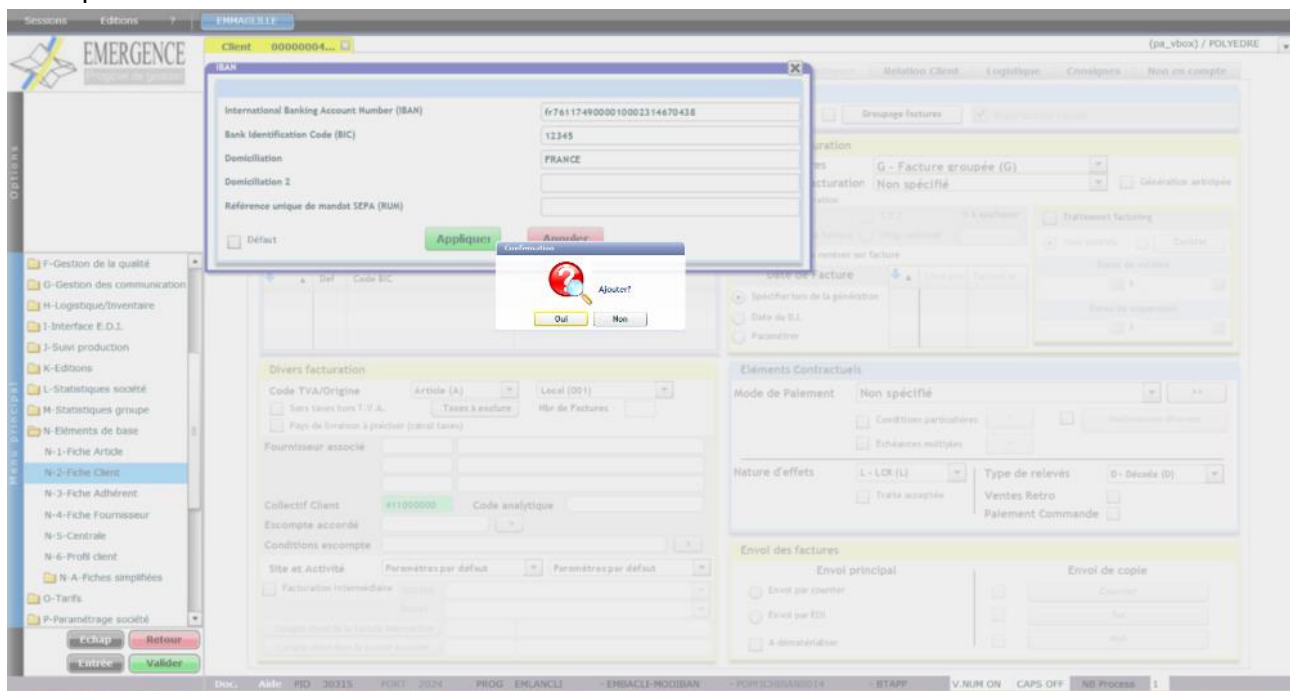


Figure 21 : Erreur d'entrer IBAN

Dans la simulation ci-dessus, on peut constater que l'utilisateur a entré un IBAN contenant des caractères en minuscules, non seulement cela rend impossible le calcul de vérification mais va aussi faire planter totalement le logiciel avec une boucle infinie. Pour cela, j'ai dû développer une fonctionnalité de « rattrapage d'erreur » permettant la transformation des minuscules en majuscules, ce qui permet de vérifier l'IBAN sans demander à l'utilisateur de tout retaper.

L'IBAN est alors réécrit correctement pour être inséré dans la base de données correctement, comme la capture ci-après.

The screenshot displays the 'EMERGENCE' software interface for client management. The client name is 'APPEL' and the client number is '00000004...'. The interface is divided into several sections:

- Références Client:** Shows the client's account details, including the account number '45', the bank 'APPEL', and the IBAN 'FR7611749000010002314670438'.
- Codes R.I.B.:** A table listing the client's bank details:

Def	Banque	Gulchet	Compte	RIB	Domiciliation
>	11749	00001	00023146704	38	FRANCE
- Codes I.B.A.N.:** A table listing the client's IBAN details:

Def	Code BIC	Code IBAN
>	12345	FR7611749000010002314670438
- Divers facturation:** Includes fields for 'Code TVA/Origine', 'Article (A)', 'Local (001)', 'Fournisseur associé', 'Collectif Client', 'Escompte accordé', and 'Conditions escompte'.
- Mode de Facturation:** Includes fields for 'Type de Factures', 'Séquence de facturation', 'Date de Facture', and 'Éléments Contractuels'.
- Envoi des factures:** Includes fields for 'Envoi principal' and 'Envoi de copie'.

The interface also features a sidebar with navigation options and a top menu bar with various tabs like 'Général', 'Paramètres', 'Tarifs et référentiels', etc.

Figure 23 : correction automatique IBAN

2.3.5 Gestion des bases tarifaires

Pour cette dernière fonctionnalité, son principe de fonctionnement reste similaire à celui de la gestion des types de mail précédemment présentée. Cependant, il ne sera pas question de modifier des tables gérant les tarifs, il sera question de créer une toute nouvelle table, dont les attributs seront des éléments d'autres tables⁵.

La difficulté ici présente est plus relative à des principes de gestion propres qu'à la subtilité technique qui sera utilisée, celle-ci a déjà été utilisée dans le passé et sera réutilisée.

Ce programme est la dernière fonctionnalité que j'ai développée. Une autre fonctionnalité, beaucoup plus complexe et dense, est en cours de développement mais son état d'avancement actuel ne permet pas d'afficher un résultat concret pour le moment. Cette fonctionnalité consiste en un listing des opérations de caisse réalisées dans un magasin.

⁵ En d'autres termes, notre table sera une table de jonction pour simplifier la lecture des utilisateurs.

3 Conclusion

3.1 Bilan personnel

Durant ces premières semaines d'apprentissage, j'ai acquis de nombreuses compétences techniques, comme l'analyse de code « étranger » qui est un exercice complexe mais aussi très pédagogique et enrichissant car il permet d'avoir une autre vision dans la programmation de certaines fonctionnalités. Une autre compétence est la mutualisation de mes compétences scolaires dans la réalisation de mes missions. En effet, il est question de lier du langage SQL, algorithmique et la gestion dans le cadre d'un même objectif.

A un niveau plus personnel, ces quelques semaines m'ont apporté beaucoup d'autonomie et de la rigueur dans mon travail, des habilités plus difficiles à développer que dans un milieu purement scolaire où l'accès à l'aide est plus disponible que dans le milieu professionnel, même si l'entraide reste la clé du travail d'équipe en entreprise.

3.2 Bilan général

En conclusion, nous avons présenté mon entreprise d'accueil, une PME appartenant à un grand groupe, ce qui induit des exigences dans la qualité du travail compte tenu du nombre de plus en plus important d'utilisateurs, mais aussi de la nécessité de fiabilité du logiciel pour un grand groupe. Dans le cadre de mon alternance, des missions me sont confiées ponctuellement en fonction des demandes et besoins des différents utilisateurs du progiciel. Ce planning ainsi que mon manque de compétences dans certains domaines, ne me permet pas d'accéder à des projets d'envergure pour le moment. Cependant, il faudra noter une augmentation progressive du niveau de compétence demandé pour mes différentes missions ainsi qu'un accroissement de l'attractivité (au niveau développement) de mes dernières missions.

Enfin, il sera intéressant de noter qu'en tenant compte des lignes suivies par mes prédécesseurs alternants, un projet de plus grande envergure, en équipe ou solitaire, me sera bientôt confié.