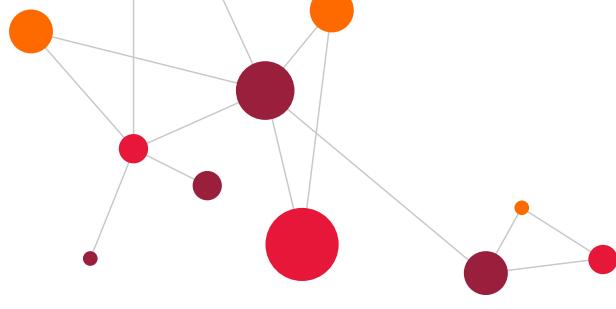




# DOSSIER DE VALIDATION

Concepteur Développeur d'Applications Numériques

Nom, Prénom	HAMON Glenn
Nom, Prénom du tuteur	MONGET-SARRAIL Sébastien
Acronyme de la certification IPI visée	CDAN
Niveau visé	Niveau II
Date de la soutenance	27 Novembre 2019
Lieu de la soutenance	EPSI BORDEAUX



## Remerciements

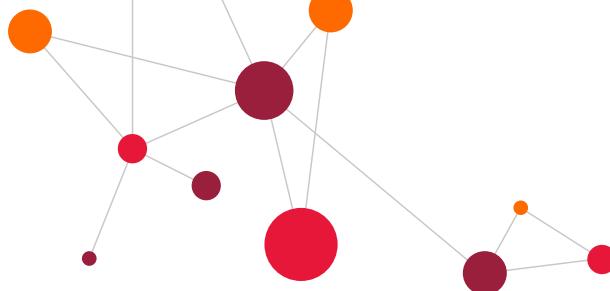
Je tiens à remercier toutes les personnes qui ont contribué au bon déroulement de cette année d'alternance.

Je remercie tout particulièrement mon tuteur Sébastien Monget-Sarrail pour sa pédagogie et sa disponibilité.

Un grand merci à toute l'équipe du Centre de Service pour leur accueil, leur gentillesse et leur bonne humeur. Je les remercie également pour la patience dont ils ont parfois dû faire preuve à mon égard et pour toutes les connaissances qu'ils m'ont transmises.

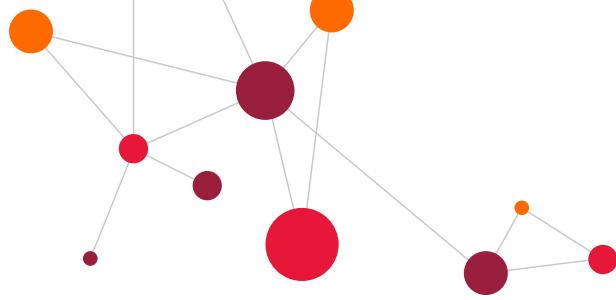
Merci également à l'équipe pédagogique de l'EPSI, je pense notamment à Michel GILLET et David GAYERIE.





# Sommaire

<b>1</b>	<b>Mon profil</b>	<b>1</b>
<b>2</b>	<b>L'entreprise CGI</b>	<b>2</b>
<b>2.1</b>	<b>CGI</b>	<b>2</b>
<b>2.2</b>	<b>ZOOM SUR CGI FRANCE</b>	<b>4</b>
<b>2.3</b>	<b>MON SERVICE (LAF)</b>	<b>4</b>
2.3.1	FONCTIONNEMENT DU SERVICE	4
2.3.2	STRUCTURE DU SERVICE	4
2.3.3	MÉTHODE DE TRAVAIL	5
<b>3</b>	<b>Présentation du poste</b>	<b>7</b>
3.1	<b>MA MISSION AU SEIN DE CGI</b>	<b>7</b>
3.2	<b>TECHNOLOGIE UTILISÉE</b>	<b>7</b>
<b>3.3</b>	<b>MON ENVIRONNEMENT DE TRAVAIL</b>	<b>8</b>
3.3.1	MON POSTE DE TRAVAIL	8
3.3.2	MES OUTILS DE DÉVELOPPEMENT	8
3.3.3	AUTRES OUTILS :	11
3.3.4	MÉTHODOLOGIE D'APPROCHE	12
<b>4</b>	<b>Missions effectuées</b>	<b>13</b>
<b>4.1</b>	<b>EN ENTREPRISE :</b>	<b>13</b>
4.1.1	EXEMPLE N°1 - PURGE FICOBA (KBP431)	13
4.1.2	EXEMPLE N°2 – GSAF PPH (HJAA24)	23
4.1.3	EXEMPLE N°3 – EXTRACTION DES FGR RTG VERS MDG (LDC131)	27
4.1.4	EXEMPLE N°4 – SERVICE APPLICATIF – MISE À JOUR DES FGR (RISCLI067)	32
<b>4.2</b>	<b>EN FORMATION</b>	<b>39</b>
4.2.1	LE PROJET DEV'HIRE	39
4.2.2	LE PROJET ANDROIDWORKSHOP	45
<b>5</b>	<b>Conclusion générale</b>	<b>53</b>
<b>Lexique</b>	<b>55</b>	
<b>ANNEXES</b>	<b>56</b>	



## 1 Mon profil

Après l'obtention d'un diplôme de Master en Analyse Chimique, j'ai exercé quelques années dans ce secteur d'activité, principalement en Recherche et Développement. Malgré tout l'intérêt que je portais à ce métier et suite à des difficultés de pérennisation de ma situation professionnelle, j'ai pris la décision de me reconversion dans le secteur informatique.

J'ai choisi le métier de développeur pour de nombreuses raisons.

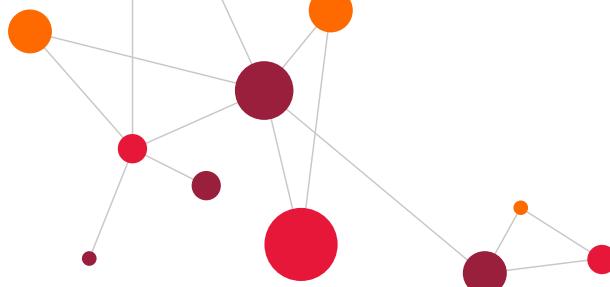
Tout d'abord, c'est un métier où la réflexion est omniprésente à travers la nécessité de comprendre l'aspect fonctionnel fourni par le client et traduit en langage informatique via des algorithmes.

Ce qui m'intéresse également c'est le faible risque de lassitude. En effet, au vu de la vitesse d'évolution du domaine des technologies de l'information, la veille technologique est primordiale et l'apprentissage permanent.

De plus, créer un produit est valorisant et je m'aperçois au quotidien que mon épanouissement personnel s'accroît au fur et à mesure de la maîtrise des outils. En effet, pour répondre au mieux au besoin du client, le rôle du développeur est forcément au centre du projet, car sans développement le produit ne peut aboutir. Le développeur est le socle technique du projet, la qualité du code produit permet de fidéliser le client.

Enfin, ayant un bon relationnel, le travail en équipe me plaît énormément. Un véritable lien de confiance se crée avec l'analyste qui permet au développeur de conserver une certaine autonomie et liberté dans la rédaction du code.

Je suis très satisfait d'avoir intégré le cursus UDEV' et d'avoir pu bénéficier de cette année d'alternance pour réellement apprendre ce qu'est le métier de développeur.



## 2 L'entreprise CGI

### 2.1 CGI

J'ai effectué mon année d'alternance dans l'entreprise CGI.

CGI est une ESN (Entreprise de Service du Numérique) canadienne qui a été fondée en 1976 par Serge Godin (actuel Président exécutif du conseil) et André Imbeau (actuel conseiller du Président exécutif).

Le sigle CGI signifie « Conseillers en Gestion Informatique » mais la signification officielle en anglais est « Consultants to Government and Industry »).

Le rêve de Serge Godin : « Créer un environnement où nous avons du plaisir à travailler ensemble et où, en tant que propriétaire, nous participons au développement d'une entreprise dont nous sommes fiers ».

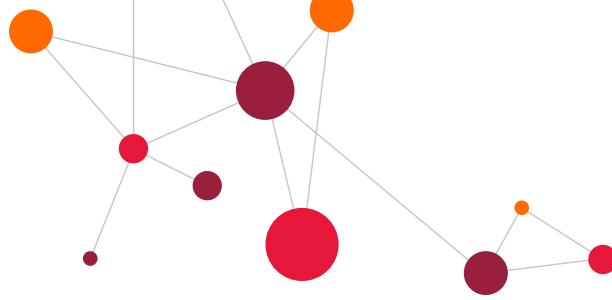
Depuis sa création, la société n'a cessé de croître. Aujourd'hui, la société est devenue une structure internationale qui réalise un chiffre d'affaire annuel de 11,5 milliards \$CA (soit environ 7,6 milliards d'euros) et regroupe 77 500 collaborateurs (membres).

L'entreprise est présente dans 40 pays réparties sur 400 sites au total.

A l'échelle mondiale, CGI est organisée en différentes SBU (unités d'affaires stratégiques). La France fait partie d'une SBU comprenant également le Brésil, la Belgique, l'Espagne, le Luxembourg, le Maroc, le Portugal et la Roumanie.



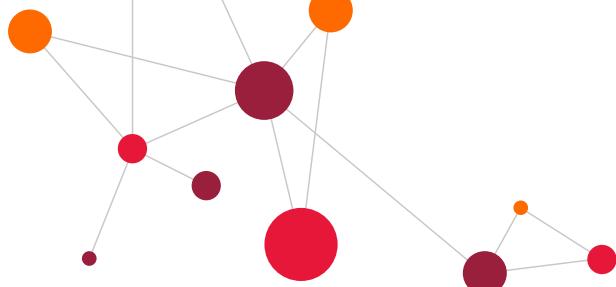
Figure 1 : Répartition mondiale de CGI



CGI travaille en collaboration avec plus de 5000 clients. CGI intervient dans de nombreux secteurs tels que les Assurances, les Banques, le Commerce de détail et services aux consommateurs, la Communication. CGI travaille également pour différents gouvernements.

SECTEUR MANUFACTURIER	PÉTROLE ET GAZ	COMMERCE DE DÉTAIL ET SERVICES AUX CONSOMMATEURS	TRANSPORT ET LOGISTIQUE	SERVICES PUBLICS
SECTEUR BANCAIRE	COMMUNICATIONS	GOUVERNEMENTS	SANTÉ ET SCIENCES DE LA VIE	ASSURANCE

Figure 2 : Les clients CGI



## 2.2

### Zoom sur CGI France

En 2012, CGI s'est implanté en France avec le rachat de LOGICA lui permettant de doubler ses effectifs passant de 31 000 à 68 000 membres.

L'organisation de CGI se découpe en grandes unités appelées BU (Business Unit) : En France, il existe une dizaine de BU.

A l'agence de Bordeaux, trois BU cohabitent :

- GO (Grand Ouest) : Elle réunit 1600 membres sur un large secteur Ouest du territoire national. Elle adresse des clients à proximité quel que soit le secteur d'activité.
- FGDC (France Global Delivery Center) : Répartie dans toute la France, elle adresse des clients nationaux tous types confondus.
- TPSHR (Transport Public Service and Human Resources) : Elle répond aux grands enjeux des clients dans les domaines du Transport, du Secteur Public et de l'organisation RH.

## 2.3

### Mon Service (LAF)

Dès mon arrivée, j'ai intégré la BU Grand-Ouest. J'ai été affecté sur le Centre de Service LAF (Lutte Anti-Fraude) pour le client La Banque Postale (LBP). Ce centre de Service consiste en un regroupement d'ingénieurs et de techniciens spécialisés sur la technologie Mainframe z/OS intervenant à distance sur les projets de développement ou de maintenance applicative.

#### 2.3.1

##### Fonctionnement du service

#### 2.3.2

##### Structure du service

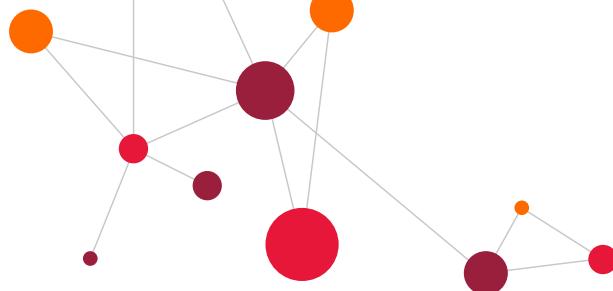
Le service s'organise de la manière suivante :

Un Directeur de Projet (Philippe Masson) gère l'ensemble du service. Il est secondé par un Chef de Projet (Samuel Da Silva). L'équipe est également constituée de trois analystes et de trois développeurs.

Le service gère actuellement trois applications : A2G (Authentification forte 2<sup>ème</sup> Génération), RSK (Risques et Réglementaires) et LAT (Lutte Anti-Terroriste).

Ainsi, chaque analyste est responsable d'une application. Ce sont eux qui réalisent les devis, rédigent les spécifications et documentations. Ils conçoivent les applications à partir du cahier des charges du client.

Les développeurs (dont je fais partie) ne sont pas affectés à une application spécifique. Nous sommes en mesure d'intervenir sur n'importe quel domaine en fonction de notre disponibilité.



### 2.3.3 Méthode de travail

Dans mon service, notre méthode de travail se base sur un modèle de cycle en V :

- Le client nous fournit un cahier des charges qui correspond à une description du besoin.
- L'analyste rédige un devis qui reformule ce besoin, explique la solution envisagée et réalise un chiffrage.
- Suite à la validation de ce devis, l'analyste rédige un dossier de conception générale (DCG).
- Une fois ce DCG également validé par le client, l'analyste réalise un dossier de conception détaillé (DCOD).
- Ce DCOD est alors transmis au développeur qui va alors coder le programme et réaliser des tests unitaires afin de s'assurer du bon fonctionnement de ce dernier.
- L'analyste va ensuite effectuer des tests d'assemblage afin de s'assurer que le programme soit compatible avec l'ensemble de la chaîne.

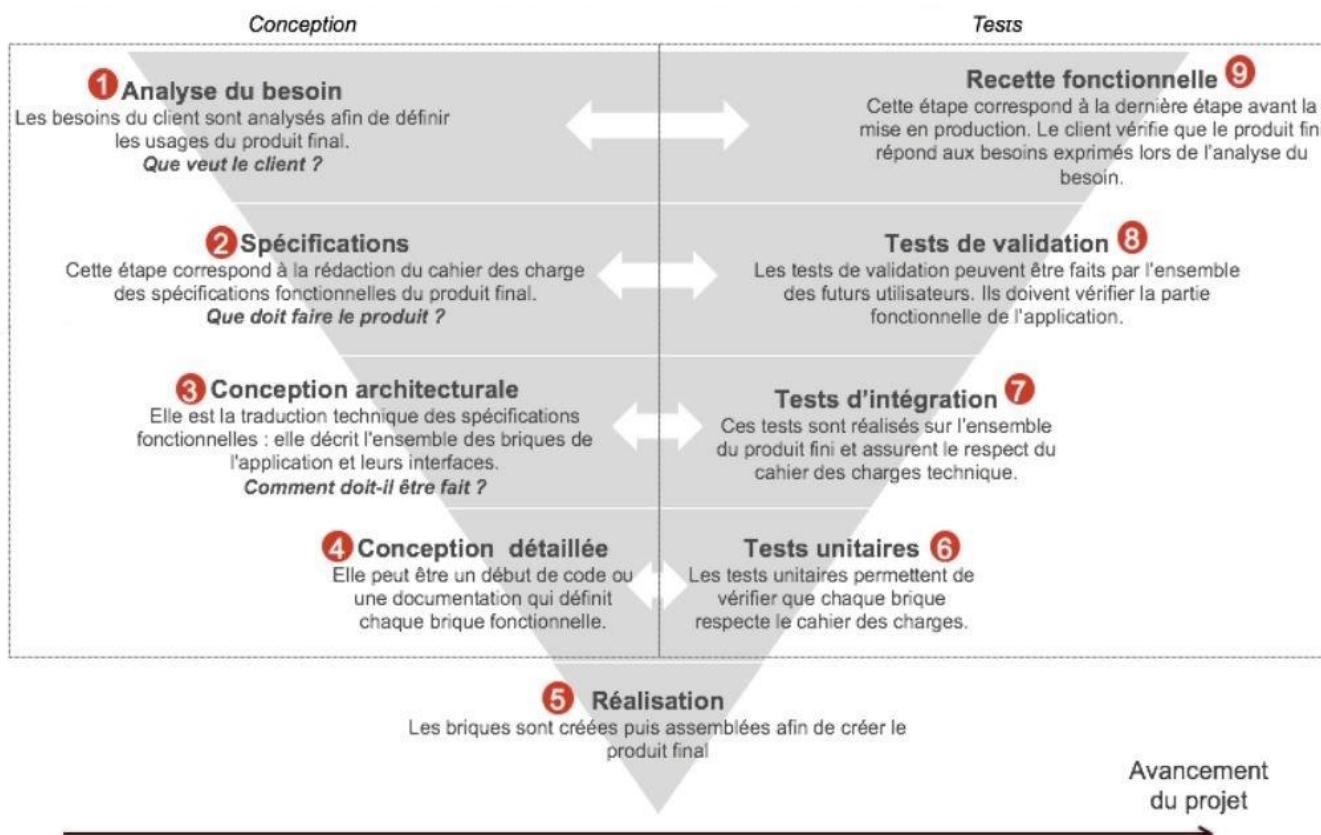
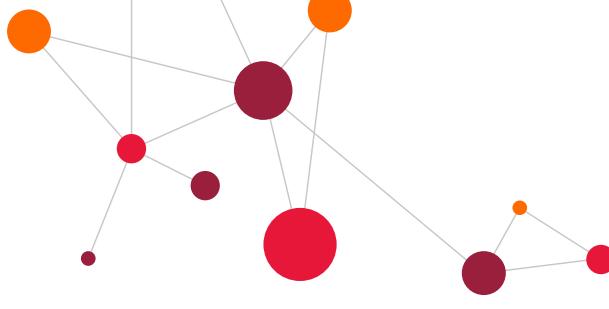


Figure 3 : schéma d'un cycle en V



Chaque jour, l'équipe se réunit pour un « daily meeting ». Ce rapide point se fait devant un tableau et chaque collaborateur explique très brièvement l'avancée de ses travaux et ses éventuels points de blocage. Lorsque plusieurs développeurs travaillent sur la même chaîne, cet échange permet de connaître l'avancement global du sujet. Cela permet également à l'analyste en charge d'un sujet de suivre son avancement ainsi que sa rentabilité.

Devis/analyse	DCG	DCOD	DMP	RTU	TA / TI	LIV	PROD	Infos
		Stockage				INTEGRATION REP LIV	HJ ROUTE2 20/03/19	RJO : Absent du 16/03 16/08
				LEADER	MCO - CIA	UPLF	OSD/RC Réunion ✓ 16/03/19 Notarisation L+4 ✓ 16/03/19	SAS : Absent 16/03
	NAT RTC phase 2		Supervision Méthode Cléverse				KLN814AA vers SPRITE 02/03/19	GHA : Absent du 23/02 au 16/08 et du 26/08 au 27/08
	Double Personnel B2S						R2-CENDOPOLIS 04/10/19	BTA : Absent du 23/02 au 27/08
							FATCA	CME :
							R2-LBPLC 17/07/19	À Daily meeting: Présenter les nouveaux sujets
							Defri-Purge 09/18/10/19	SDA : congés 17 juillet au 01/07
							R2-LBPA IARD 06/05/19	GMA : congés 16 août au 18/08
							FIP-KU61AAA TFE 13/05/19	SGR : NGU :
	GMA	ED Sensibilité	LX LGP PFLN		MCO GIE		ART-DAL 13/03/19	
	R.B3	ED ENCAH					FATCA TC 14/10/19/19	
							R2-LAM DREV 06/10/19	

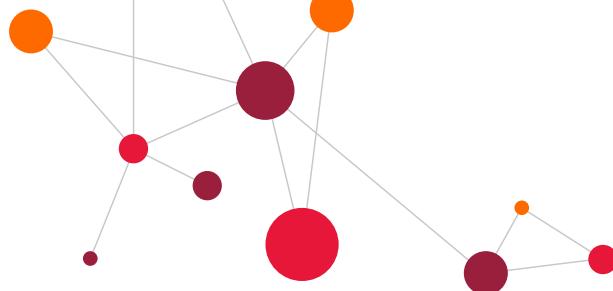
**Infos**

- RJO : Absent du 16/03 16/08
- SAS : Absent 16/03
- GHA : Absent du 23/02 au 16/08 et du 26/08 au 27/08
- BTA : Absent du 23/02 au 27/08
- CME :
- À Daily meeting: Présenter les nouveaux sujets
- SDA : congés 17 juillet au 01/07
- GMA : congés 16 août au 18/08
- SGR :  
NGU :

**A faire**

- MCO - CIA 49 (panuel) GHA
- CIA (quotidien) | Tous
- Sraaf (mensuel) | GHA
- GIDE - Indication Score PMO (mensuel) 16/08
- GIDE - Extraction Stock FC (mensuel avec 3 mois) 16/08
- SCORE - National (entre le 6 et le 7 de mai) 16/08
- SCORE - Régional (à 18+23 avec) 16/08
- FICOBA (mardi + Vendredi) | Tous

Figure 4 : management visuel tableau du daily meeting



## 3 Présentation du poste

### 3.1 Ma mission au sein de CGI

J'occupe chez CGI le poste de **Développeur Mainframe**. Ma mission consiste au développement de programmes en respectant les instructions fournies dans le DCOD en veillant à ce que celui-ci respecte bien le cahier des charges du client et les normes de la Banque Postale.

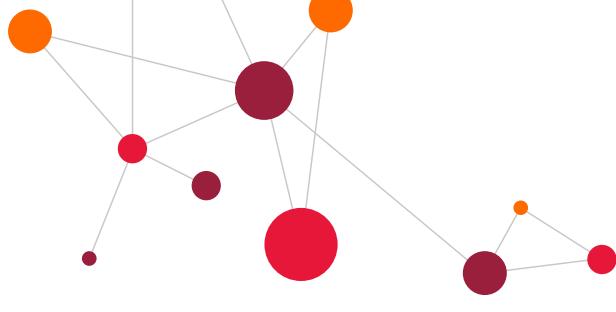
### 3.2 Technologie utilisée

Au quotidien, je travaille sur une technologie Mainframe zOS, c'est-à-dire sur une architecture de type informatique centralisée. On se connecte à cet ordinateur de grande puissance via des terminaux. Cette technologie est surtout utilisée par les banques et les assurances. Elle permet de traiter de gros volumes de données. C'est une technologie particulièrement fiable et éprouvée. Le mainframe est un écosystème fermé qui garantit un haut-niveau de sécurité.

Les mainframes utilisés par notre client sont des modèles Z doté du système d'exploitation z/OS (successeur de MVS).



Figure 5 : Système Z d'IBM



L'environnement sur lequel je travaille est principalement basée sur le langage Cobol (Common Business Oriented Language). Il est surtout utilisé pour faire de l'informatique de gestion.

C'est un langage procédural. Ainsi, il permet l'écriture de fonctions pouvant être appelée à n'importe quelle étape de l'exécution du programme engendrant un code modulaire et structuré.

Cette technologie n'étant pas enseignée dans le cursus U'Dev, une formation de 5 semaines m'a été dispensée à mon arrivée dans l'entreprise. De plus, j'ai été accompagné tout au long de cette année en alternance par les collègues de mon service.

### 3.3 Mon environnement de travail

#### 3.3.1 Mon poste de travail

Pour travailler, j'utilise un poste fixe à deux écrans. Ce poste fixe est connecté au réseau sécurisé de LBP. C'est par le biais de cet ordinateur que j'accède au z/Os.

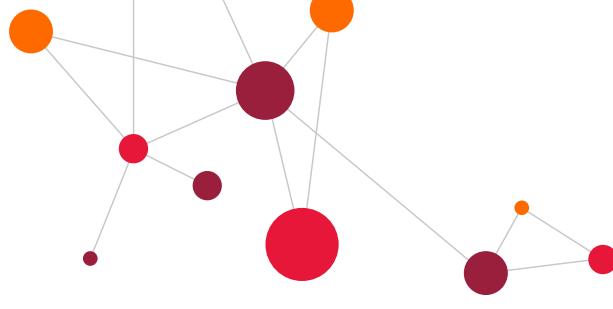
Je possède également un ordinateur portable afin de me connecter au réseau CGI pour l'utilisation des outils internes type outils RH ou pour l'utilisation d'internet.

#### 3.3.2 Mes outils de développement

A l'origine, on communiquait avec le z/Os via des terminaux passifs. Maintenant, on émule ces terminaux sur un PC via un logiciel supportant le protocole 3270 du Mainframe (le plus connu étant Rumba).



Figure 3 : terminal IBM 3270



La communication entre un terminal et le z/Os peut se faire de deux manières distinctes :

- Soit à l'aide du sous-système CICS (Customer Information Control System) qui est un moniteur transactionnel. Des codes sont associés à des transactions (ex : pvxx pour se connecter à Pacbase ou uvst pour accéder aux tables Armides) permettant d'activer les fonctionnalités développées.
- Soit avec TSO (Time Sharing Option) qui est un moniteur de temps partagé. TSO permet d'accéder à des outils comme Xpediter pour faire du débogage, Platinium pour accéder aux tables DB2, Spufi pour faire des requêtes sur ces mêmes tables ou d'accéder à différentes organisations de fichiers (PDS, fichiers séquentiels ou indexés).

Jusqu'à présent, pour réaliser nos développements, nous utilisions un AGL (Atelier de Génie Logiciel) francophone nommé Pacbase. Cet outil génère des programmes en langage COBOL.

Dans un souci de modernisation des outils et des pratiques de développement, et parce que Pacbase n'est plus maintenu par IBM; CGI et son client ont mis en place une nouvelle plateforme de développement Cobol utilisant directement du cobol natif.

Cette nouvelle plateforme appelée UDC (Usine de Développement Cobol) embarque différents outils :

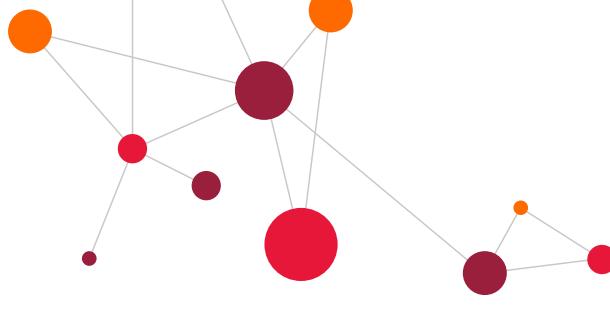
- Un émulateur permettant d'utiliser tout de même les anciens outils
- RDz (Rational Developer for Z) basé sur l'environnement de travail intégré (IDE) Eclipse.
- PTF : Framework de développement Cobol proposant un dictionnaire de données d'entreprise, un générateur de code Cobol intuitif et un contrôle de qualité de code.
- MAP : outil de cartographie permettant de faire des analyses d'impact.

La migration vers ces nouveaux outils est lente et fastidieuse car elle nécessite un remaniement de tout le « parc programmes » du SI. A la Banque Postale, le patrimoine Pacbase à migrer représente 173 applications soit 240 000 composants dont 28% ont été créés ou modifiés dans les trois dernières années.

Dans cette période de transition, mon service n'ayant pas encore migré, j'ai jusqu'à présent principalement travaillé sur l'outil Pacbase tout en m'efforçant d'utiliser au maximum la nouvelle plateforme afin de me familiariser avec celle-ci.

Le client utilise deux types de programmes :

- Les programmes batch : Ce sont des programmes automatisés qui permettent des suites d'opérations sans qu'il soit nécessaire qu'un opérateur intervienne, c'est à dire des traitements de masse.  
Pour lancer ces programmes sur le Mainframe, on utilise le langage de commande JCL (Job Control Language). Il permet la mise en relation des données (BDD et/ou fichiers) et des programmes ainsi que le séquencement des traitements.
- Les Services Applicatifs (SA) hébergés dans le moniteur CICS : il est possible de converser avec le système via un ensemble de modules appelés Service applicatif transactionnel. Ces programmes fonctionnent sur un principe de Question/Réponse. Ils sont composés de plusieurs modules dont le nombre et l'organisation varie en fonction des besoins. Leur structure de base correspond tout de même à une architecture de type Appelant/Connecteur/Appelé. C'est en utilisant ce type de programme que l'utilisateur va pouvoir interagir avec les bases de données.



Le cycle de vie d'un composant est matérialisé par le schéma suivant :

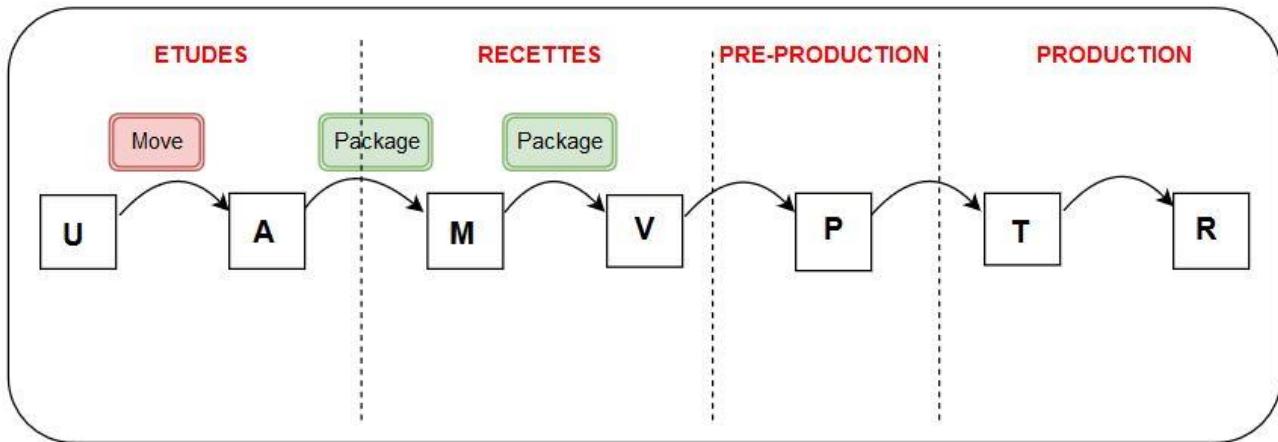


Figure 6 : Les différentes étapes de la vie d'un composant

Lors du développement et de la réalisation des test unitaires, le composant se trouve en stage U. On le passe en stage A à l'aide d'un simple « move » afin de réaliser les tests d'assemblages. Ces deux stages font partie de l'**environnement d'études**.

La promotion d'un composant (programme, JCL, documentations) est réalisée par le système de livraison.

Les livraisons s'effectuent via un outil appelé **COCA/ENDEVOR**. **Plus on avance dans le cycle de vie du composant, plus le protocole de promotion est rigoureux.**

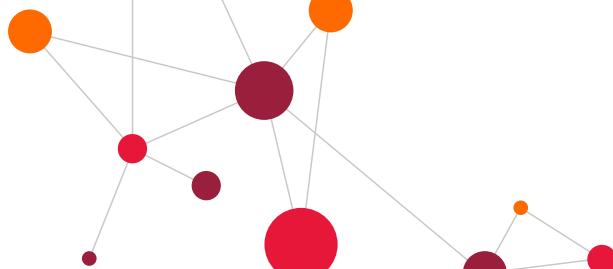
**La livraison** consiste à passer le ou les composants en **environnement de recettes** selon ce qu'il a été convenu lors de la demande c'est-à-dire de le passer du stage A au stage M en faisant un package ou également du stage M au stage V.

Le stage M concerne l'**IMOE** (Maitrise d'œuvre) qui y effectue les tests d'intégration.

Le stage V concerne la **RMOA** (Maitrise d'ouvrage) qui effectue le recettement.

Il est à noter que dans le processus de livraison, si une étape n'est pas satisfaite, il y a alors retour au stage U.





### 3.3.4

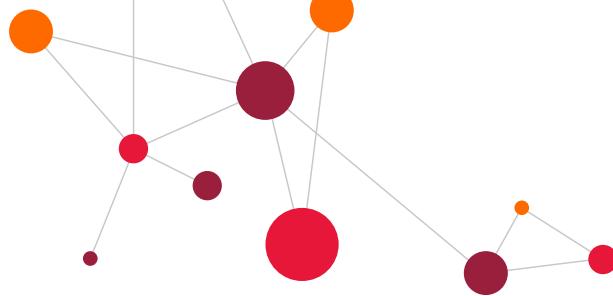
### Méthodologie d'approche

Mon travail au quotidien consiste à développer les programmes. Cependant, il m'est arrivé de prendre le rôle d'analyste et de rédiger des dossiers de conception détaillée.

Lorsque je réalise une tâche de développement, je travaille méthodiquement de la façon suivante :

**Compréhension / Déclinaison technique / Synthèse / Écriture des tests / Écriture du programme / Passage des tests unitaires.**

- Je lis le CDC et le DCG afin de comprendre globalement les besoins fonctionnels du client.
- Je lis ensuite scrupuleusement le DCOD afin d'étudier les aspects techniques. Lorsque des interrogations subsistent, je reviens vers l'analyste en charge du dossier.
- Je rédige alors mes fiches de tests unitaires (TU) que je fais ensuite valider par l'analyste ayant rédigé le DCOD.
- Je passe au développement des tests et du ou des programme(s).
- Je réalise l'ensemble des cas de tests unitaires rédigés précédemment. Si un cas de test n'est pas validé, je modifie le programme et repasse l'ensemble des mes cas de tests afin de m'assurer que cette modification n'a pas eu d'autres impacts sur le programme.
- Je transmets ensuite l'ensemble de mon travail à un autre membre de l'équipe qui va réaliser les tests d'assemblage avant de réaliser la livraison.



## 4 Missions effectuées

### 4.1 En entreprise :

Au cours de cette année passée chez CGI, j'ai eu l'opportunité d'intervenir sur de nombreux programmes que ce soit pour une simple modification d'une ligne de code dans un programme ou une véritable création.

Le client pour lequel je travaille impose de **nombreuses normes à respecter** et a fixé des **critères de qualité d'écriture de code**. Par exemple, toutes les fonctions doivent avoir un titre, ces fonctions ne doivent pas être trop longues, le code doit être correctement commenté, l'utilisation de l'expression GO TO est proscrite etc...

Suivant le respect de ces différents critères, une note nous est attribuée lors de la compilation. Cette note, appelée **note PQC** (Pacbase Quality Control) ne doit pas être inférieure à 90/100 avec un objectif fixé à 95/100.

Lors de la livraison, on ne fournit pas systématiquement les fiches TU au client mais celui-ci peut nous les demander ultérieurement. Par conséquent, celles-ci doivent être parfaitement rédigées et archivées sur le réseau.

Dans cette partie, j'ai choisi de vous présenter quelques projets sur lesquels j'ai travaillé.

#### 4.1.1 Exemple n°1 - Purge FICOBA (KBP431)

Ce programme fait partie du domaine RSK (Risques et Réglementaires). C'est un programme batch.

##### Expression des besoins :

Ce programme consiste à purger certaines tables afin de se mettre en accord avec la réglementation RGPD<sup>1</sup>.

Ce traitement doit donc permettre de supprimer tout enregistrement antérieur à 5 ans dans l'application FICOBA.

Les règles de gestion étaient les suivantes :

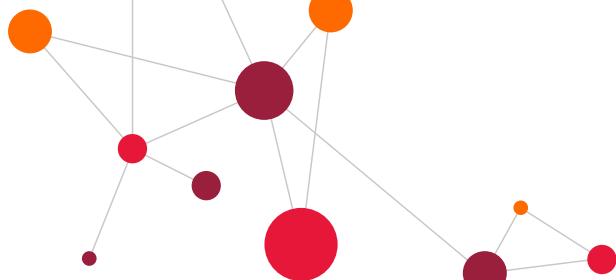
Règle	Description
RG01	Les traitements de purge concernent toutes les tables FICOBA (10 tables).
RG02	Le traitement est annuel et tournera le 1 <sup>er</sup> Mardi ouvré du mois de Janvier
RG03	La date max de conservation correspond à la date du jour – le délai de conservation qui est pour l'instant de 5 ans (60mois). Ce délai de conservation est positionné dans la table de paramétrage KBPURGE et sera commun à toutes les tables.
RG04	Toute donnée au-delà du seuil (5 ans) sera effacée.

Les livrables attendus me concernant étaient :

- Le programme en lui-même KBP431 développé sous Pacbase.
- Les fiches de tests unitaires
- Le JCL généré par l'outil EGEN permettant de lancer ce programme batch.

---

<sup>1</sup> Réglementation Générale sur la Protection des Données



## Analyse :

Le schéma du traitement est le suivant :

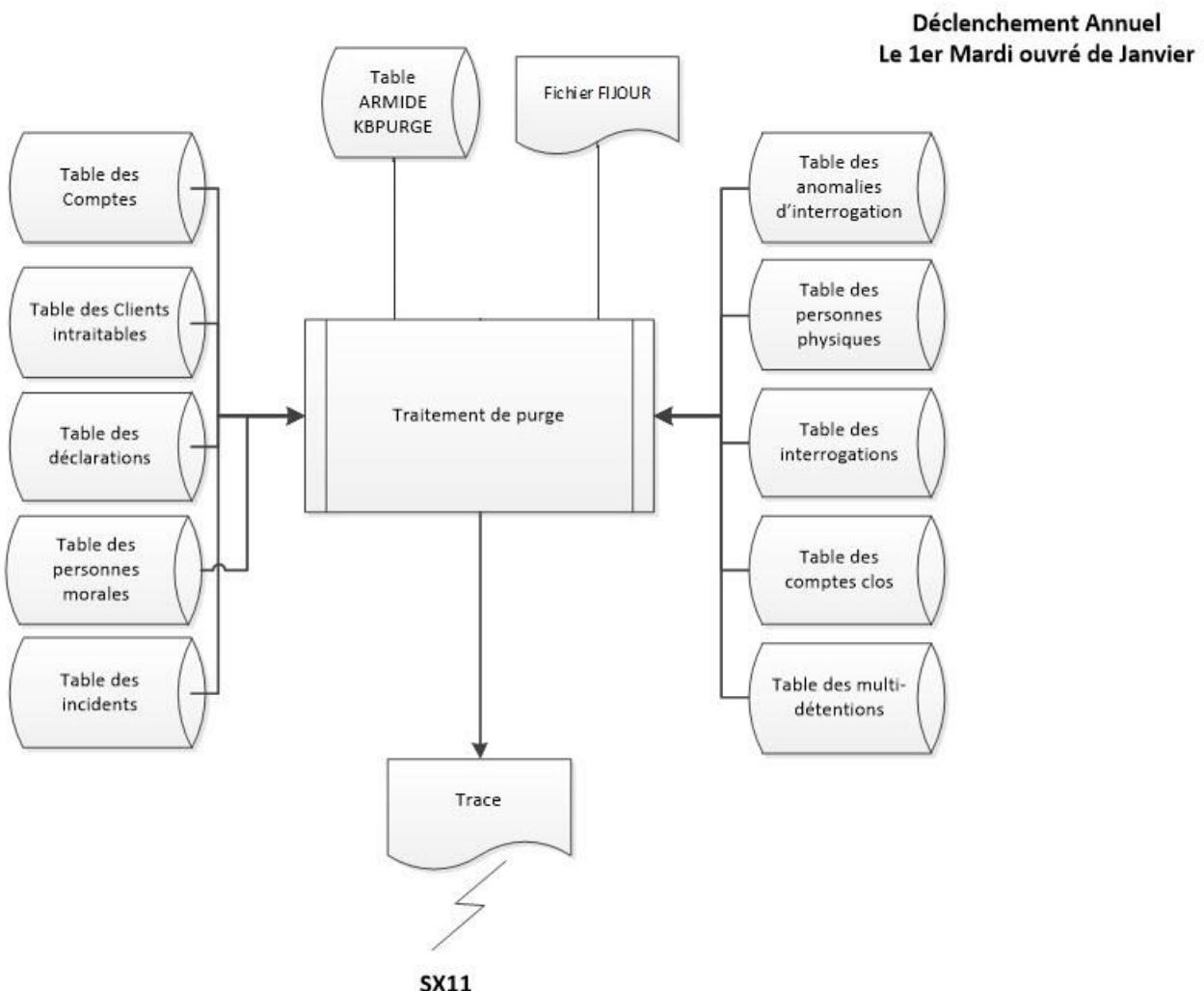
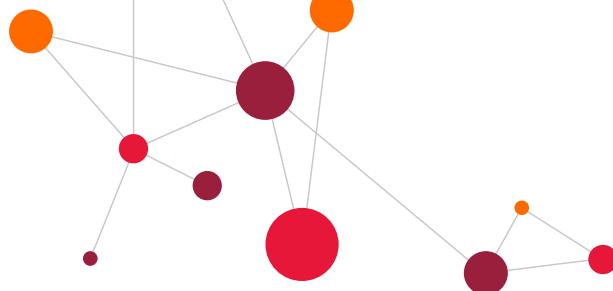


Figure 9 : schéma du traitement KBP431

En entrée de programme : nous avons la table Armide KBPURGE qui permet de stocker le délai de traitement (5 ans). Externaliser ce paramètre du programme permet d'anticiper les évolutions au cas où la législation change. En plus de la table Armide, nous avons un fichier FIJOUR qui correspond à la date du jour selon un format spécifique sur laquelle va se baser le traitement.

En sortie de programme : En plus de la sysout générant les logs, nous avons un fichier trace qui est une copie de cette sysout.

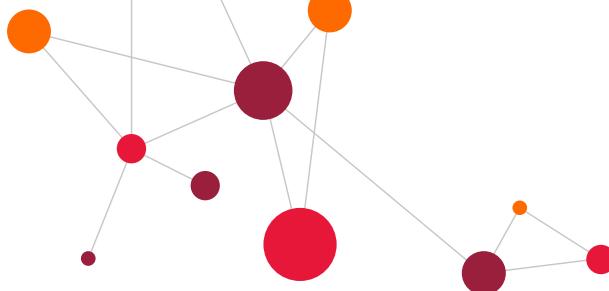


## Réalisation et tests unitaires :

Dans un premier temps, j'ai rédigé mon plan de tests :

	B	C	D	E	F	G	H	I
1	CGI		LA BANQUE POSTALE		Avancement : 100%			
2								PDF Word
3	<b>IDENTIFICATION TYPE de TEST</b>			<b>DESCRIPTION</b>				
4	N° cas	Versi on	N° cas outils	Type de cas de test	Outil	Objectifs des tests	Description du cas de test Action à réaliser	Résultat attendu
5								
6								
7	1	1	1	Traitement en erreur	JCL	Fichier FIJOUR vide	DUMMY dans le JCL	code retour attendu : 12 Ecriture en TRACE et en SYSOUT : "Fichier FIJOUR vide"
8	2	1		Traitement en erreur	JCL	Vérification date	Fichier FIJOUR en entrée avec une date non valide (2019-13-31)	code retour attendu : 12 Ecriture en TRACE et en SYSOUT : "date FIJOUR invalide"
9	3	1		Traitement en erreur	JCL	Erreurs Armide. Problème d'accès Armide	Supprimer les joblibs Armide	code retour attendu : 13 Ecriture en TRACE et en SYSOUT
10	4	1		Traitement en erreur	JCL	Erreurs Armide. Ne trouve pas la clé.	La clé 'RGPD' ayant été ajoutée seulement aujourd'hui; elle n'a pas encore été prise en compte par Armide. Ainsi, en mettant UV5XIAAA au lieu de UVEXIAAA, nous pouvons atteindre notre code erreur,	code retour attendu : 13 Ecriture en TRACE et en SYSOUT "KBPURGE - délai non trouvé. Clé CRQPUR = RGPD"
11	5	1		Traitement en erreur	JCL	Erreur DB2	Modifier le plan DB2 dans le JCL	code retour attendu : 12 Ecriture en TRACE et en SYSOUT : "Erreur DB2 SELECT COUNT KBT0AI00 - SQL CODE"
12	6	1		Traitement normal	JCL	Cas passant	Fichier FIJOUR en entrée valide. Présence de données dans chaque table ayant des dates supérieures à 5 ans (vérification à partir des SELECT COUNT).	code retour attendu : 00 Ecriture en TRACE et en SYSOUT du rapport souhaité avec le nombre d'enregistrements supprimés pour chaque table + vérification dans les tables que les données ont bien été supprimées (Les SELECT COUNT doivent valoir zéro).
13	7	1		Traitement normal	JCL	Vérifier le comportement du programme lorsqu'il n'y a rien à supprimer	Repasser le programme une deuxième fois. Après avoir vérifié que les SELECT COUNT valent bien zéro.	code retour attendu : 00 Ecriture en TRACE et en SYSOUT du rapport avec le nombre d'enregistrements supprimés pour chaque table égale à zéro.
14	+ Ne rien insérer sous cette ligne							
15								
16								
17								
18								
	Présentation		TdB Suivi		Paramètres			

Figure 10 : élaboration du plan de tests du programme



Dans un second temps, je commence l'écriture du code.

J'ai tout d'abord créé l'enveloppe du programme aussi appelée carte de définition.

Les programmes sont structurés en carte (héritage du système historique des cartes perforées), elle se présente ainsi :

```

VA Pac 3.5 V04 FICOBA 2 SISF *XCR0573.PB12.B44.6886
CODE PROGRAMME KBP431

NOM DU PROGRAMME.....: Traitement de purge FICOBA

CODE CLASSEMENT DU PROGRAMME.....: KBP431

VARIANTE DU LANGAGE A GENERER.....: X IBM MVS/ESA OS/390
OPTION NUMEROTATION CADRAGE COBOL...
OPTION CARTES AVANT PROGRAMME...
OPTION CARTES APRES PROGRAMME...
CODE DU PROGRAMME GENERE.....: KBP431
TYPE DE PROGRAMMATION.....: P
NATURE DU PROGRAMME.....: B
TYPE DE L'ENTITE.....: P PROGRAMME
CONTROLE DE PRESENCE ZONE NUMERIQUE:
GENERATION INDICATEURS SQL AVEC '-':

MOTS CLES ASSOCIES.: PURGE, FICOBA
MIS A JOUR PAR.....: XCR0573 LE : 19/07/2019 A : 15:14:43 BIB : B44
NO DE SESSION.....: 6874 BIBLIOTHEQUE : B44 BLOCAGE :

O: C1 CH: ACTION:

```

Figure 11 : carte de définition du programme

En suivant, j'ai renseigné la carte commentaire (-gc) qui est généralement une des premières sources de documentation. Sur cette carte, on renseigne l'auteur du programme mais également les ressources en entrée et en sortie. Il est également possible d'y renseigner les codes erreur.

```

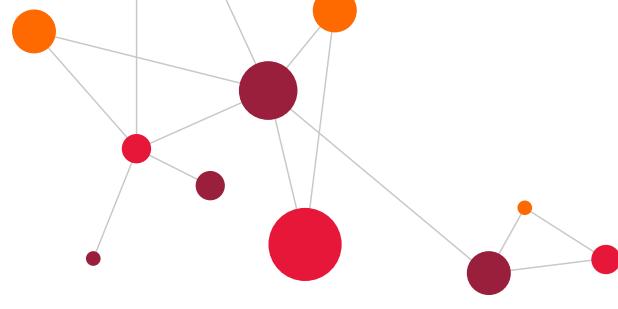
VA Pac 3.5 V04 FICOBA 2 SISF *XCR0573.PB12.B44.6886
COMMENTAIRES DU PROGRAMME KBP431 Traitement de purge FICOBA

A NLG : T DESCRIPTION
000 :
    : DATE DE CREATION : 15/07/19
    : NOM DE L'AUTEUR : GHA
    : SOCIETE : CGI
020 : * ****
030 : * Ce programme permet de purger tout enregistrement en base
040 : * FICOBA antérieur à 5ans.
050 : * ****
060 : * Entrées : - JE0611AA : fichier FIJOUR (XC62)
070 : * ----- - Table Armide : KUPURGE
080 : * ----- - Tables DB2 (KBT0AC00, KBT0AI00, KBT0AP00,
090 : * KBT0AM00, KBT0AN00, KBT0AK10, KBT0AK12,
100 : * KBT0AD00, KB0AK11 et KBT0AE00)
110 : *
120 : * Sortie : - trace FICOBA (YB13)
130 : * -----
140 : * ****

O: C1 CH:

```

Figure 12 : carte des commentaires



Lors d'une évolution du programme, on enrichit également cette carte en y apposant son nom et en décrivant brièvement les évolutions apportées.

L'étape suivante consiste à renseigner les ressources externes dans la carte des entrées/sorties. Ces ressources peuvent être des fichiers mais aussi des tables.

```

VA Pac 3.5 V04 FICOB A 2 SISF               *XCR0573.PB12.B44.6886
STRUCTURES DE DONNEES DU PROGRAMME   KBP431 Traitement de purge FICOB A
                                           
A PR SU : SD EXTERN OAMOU BLOC T    R S U RE SE M UNIT P    SELECTION F O D N EM
FJ    : XC FIJOUR SSFIU   O R      C                         *62=00   I   1
       : Z.COMPL.:
TR    : YB TRACE SSFOU   O R      CLE ACC.:
       : Z.COMPL.:
ZZ    : Z0 ZZ WSFOU   O R      CLE ACC.:
       : Z.COMPL.:
       : Z.COMPL.:
O: C1 CH:

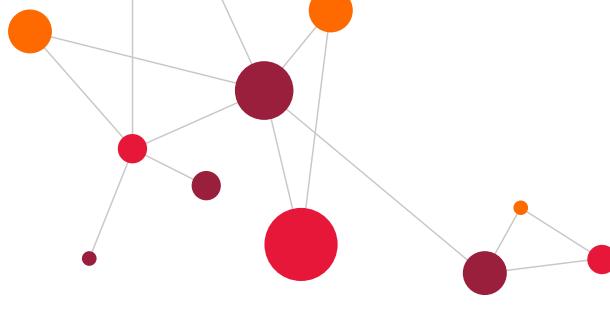
```

Figure 13 : déclaration des ressources externes

Lors de cette déclaration, je renseigne le nom du segment Pacbase correspondant à la description du fichier. Ce segment préalablement déclaré dans la bibliothèque Pacbase est lui-même décomposé en plusieurs rubriques.

Dans la figure 13, j'ai un fichier FIJOUR en entrée (SSFIU) qui correspond au segment XC62.

Ce fichier est déclaré uniquement en consultation (C). Le nom externe correspond au nom qui va apparaître lors de l'écriture du JCL associé.



Dans Pacbase, le segment XC62 est décrit de la manière suivante :

Organisation	:	Séquentiel
Format	:	FB
Segment PAC	:	XC62
Longueur	:	40

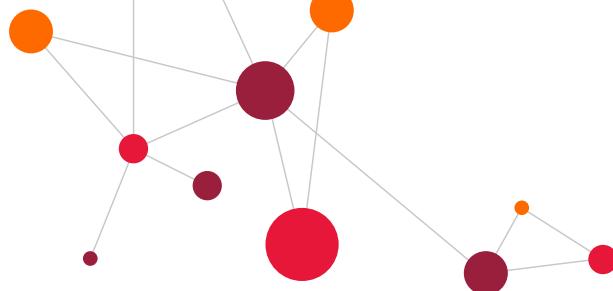
Code donnée	Libellé	Format	Observations
ZF010	Filler	X(10)	
ZDB01	Date du jour - 1ere partie	X(2)	
ZF001	Filler	X	
ZDLIM3	Libelle du mois (3 1er caractère)	X(3)	
ZF001A	Filler	X	
ZDB03	Date du jour - 3eme partie	X(2)	
ZF008	Filler	X(8)	
ZDB0A	Année de la date	99	
ZDB0Q	Quantième de la date	999	
ZDC0J	Jour de la date	99	
ZDC0M	Mois de la date	99	
ZDC0A	Année de la date	99	
ZDC0S	Siècle de la date	99	

Dans cette même figure, nous avons la déclaration du fichier trace en sortie (SSFOU) qui correspond au segment YB13. Ce fichier est déclaré en écriture (D).

Nous avons également la déclaration de la table Armide (ZZ) en entrée.

A partir de ces informations, Pacbase va générer automatiquement du code permettant par exemple :

- L'ouverture et la lecture des fichiers en entrée
- L'ouverture et l'écriture des fichiers en sortie
- La déclaration de divers compteurs pour chaque fichier (nb de lectures, nb d'écritures etc...)
- Des indicateurs de fin de fichier (dont FT qui passe à 1 lorsque tous les enregistrements du fichier en entrée ont été lus).



Ensuite, j'ai déclaré les ressources internes du programme (variables) dans la working-section (-w) :

```

VA Pac 3.5 V04 FIC0BA 2 SISF *XCR0573.PB12.B44.6886
ZONES DE TRAVAIL DU PROGRAMME P KBP431 Traitement de purge FIC0BA

DEBUT DU NUMERO DE LIGNE : BB
A NLG S NIVEAU           DESCRIPTION TABLE
  100 * ***** **** DATE DE PURGE POUR CHAQUE TABLE
  120   01     WBB0-DATE PICTURE X(10) VALUE SPACE.
  140
  160 * ***** **** VARIABLE NB ENREGISTREMENTS A SUPPRIMER ****
  180 I 01     WBB0-ZQENR VALUE ZERO.
  190 I 01     WBB0-Z9009E VALUE ZERO.

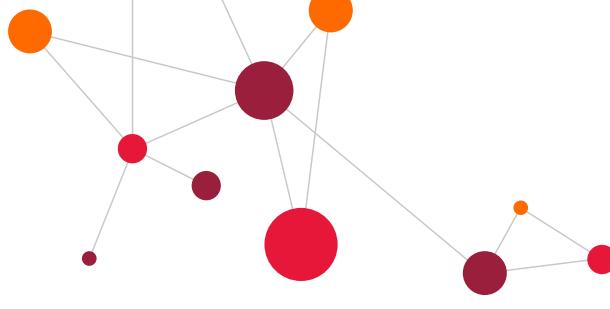
O: C1 CH:

```

Figure 14 : extrait de la working-section

Dans la figure 14, j'ai déclaré une variable WBB0-DATE de type alphanumérique sur 10 caractères (X10). Celle-ci va me permettre de stocker la date limite de purge que je vais déterminer dans le programme. Elle va par la suite être utilisée dans toutes les clauses WHERE de requêtes DB2.

Dans cette même figure, je déclare également des variables me permettant de stocker le nombre d'enregistrements que j'ai supprimés. Ces variables ont un format interne correspondant à une rubrique dans Pacbase. Je n'ai donc pas besoin de déclarer leur type. Je me contente de les initialiser à zéro.



La prochaine étape consiste à déclarer les MSP (macrostructures) que je vais utiliser dans le programme (-cp). Ces macrostructures sont des fonctions standard paramétrables m'évitant ainsi l'écriture de nombreuses lignes de code à la manière d'un framework.

VA Pac 3.5 V04 FICOB A 2 SISF MACRO-STRUCTURES DU PROGRAMME		*XCR0573.PB12.B44.6886 KBP431 Traitement de purge FICOB A	
A	MACRO NL S : SIGNIFICATION OU VALEUR DES PARAMETRES	D	V
ABAB74	: 10/DA/KBT0AI00/	* DELETE	DB2
ABAB74 05	: 15/DB/KBT0AC00/	* DELETE	DB2
ABAB74 10	: 20/DC/KBT0AP00/	* DELETE	DB2
ABAB74 15	: 25/DD/KBT0AM00/	* DELETE	DB2
ABAB74 20	: 30/DE/KBT0AN00/	* DELETE	DB2
ABAB74 25	: 35/DF/KBT0AD00/	* DELETE	DB2
ABAB74 30	: 40/DG/KBT0AE00/	* DELETE	DB2
ABAB74 35	: 45/DH/KBT0AK10/	* DELETE	DB2
ABAB74 40	: 50/DI/KBT0AK11/	* DELETE	DB2
ABAB74 45	: 55/DJ/KBT0AK12/	* DELETE	DB2
ABAB8K	: 10/CA/**/KBT0AI00/	* SELECT	COUNT DB2
ABAB8K 05	: 15/CB/**/KBT0AC00/	* SELECT	COUNT DB2
ABAB8K 10	: 20/CC/**/KBT0AP00/	* SELECT	COUNT DB2
ABAB8K 15	: 25/CD/**/KBT0AM00/	* SELECT	COUNT DB2
ABAB8K 20	: 30/CE/**/KBT0AN00/	* SELECT	COUNT DB2
ABAB8K 25	: 35/CF/**/KBT0AD00/	* SELECT	COUNT DB2
ABAB8K 30	: 40/CG/**/KBT0AE00/	* SELECT	COUNT DB2
ABAB8K 35	: 45/CH/**/KBT0AK10/	* SELECT	COUNT DB2

paramètres

Figure 15 : les macros

Une fois toutes ces mises en place effectuées, j'ai réalisé l'écriture des traitements spécifiques.

**Un programme Pacbase a une architecture bien définie (cf. annexe 8 : la boucle Pacbase).**

**Cette boucle se situe entre les fonctions f05 et f90.**

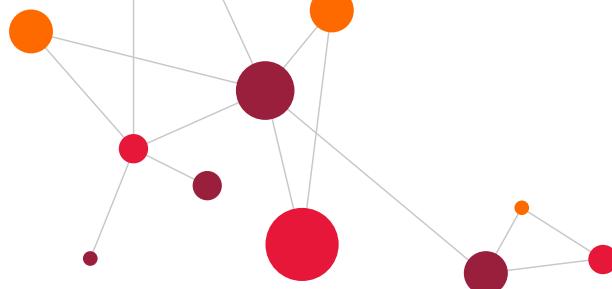
**Ainsi, le programme passe une seule fois par les fonctions f01 et f04 en début de programme.**

**Le cobol est un langage procédural. Par conséquent, il est possible d'écrire des fonctions en dehors du corps du programme et d'y faire appel dans le programme. Dans Pacbase, ces fonctions (appelées fonctions performées) s'écrivent entre la fonction f91 et f99. Elles permettent la factorisation du code et une meilleure lisibilité.**

Dans ce programme, nous avons un seul fichier en entrée, contenant seulement un enregistrement (date du jour). La boucle Pacbase ne va alors s'effectuer qu'une seule fois.

En début de programme, entre les fonctions f01 et f05, j'ai effectué les diverses opérations de vérification demandées par l'analyste dans le DCOD.

- Je vérifie que le fichier FIJOUR ne soit pas vide. Si c'est le cas, j'alimente le code retour à 12, j'écris le message suivant : 'Fichier FIJOUR vide' dans le fichier trace et dans la sysout et je stoppe le programme.
- Je vérifie la validité de la date. Cette vérification est effectuée à l'aide de la macrostructure ABDB11. Elle consiste à découper la date du fichier FIJOUR en quatre variables distinctes (jour/mois/année/siècle) puis à effectuer les vérifications nécessaires sur chacune d'entre-elles. Cette vérification s'effectue en f93.



Si la date est incorrecte, j'alimente le code retour à 12 et j'écris le message suivant : 'Date FIJOUR invalide' dans la systout et dans le fichier trace.

```

VA Pac 3.5 V04 FICOBA 2 SISF *XCR0573.PB12.B44.6886
TRAITEMENTS PROGRAMME P KBP431 Traitement de purge FICOBA FONCTION: 02

A SS NLG OPE OPERANDE NVTY CONDITION
CE 200 M 12 WSX0-ZCSRT7
CE 220 P F99SX

----- DE N CONTROLE DATE FIJOUR 10BL
DE 100 M WFJ0-ZDATE XD11-ZDB0Y8
DE 120 P F93D8
DE 140 * LA DATE N'EST PAS VALIDE 99IT EN-PRE NOT = '1'
DE 160 M W02D-ZL100 WSY0-ZX100
DE 180 P F91SY
DE 200INI TR00
DE 220 M W02D-ZL100 TR00
DE 240 P F90TR
DE 260 M 12 WSX0-ZCSRT7
DE 280 P F99SX

----- * XA N INITIALISATION ARMIDE ZONE TECH. 10BL
* XA 200 M SPACES XA00
XA 205 M 'RGPD' FM17-CRQPUR

O: C1 CH:

```

Figure 16 : extrait de la fonction f02 (KBP431)

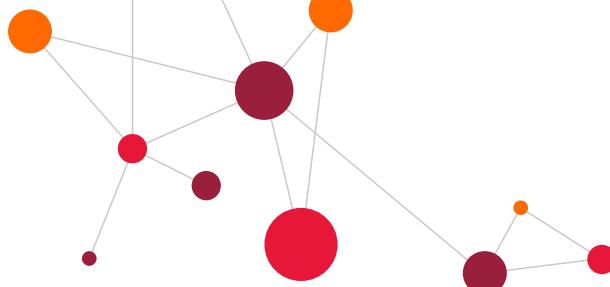
- Je récupère le délai de purge dans la table Armide. Si une erreur se produit, j'alimente le code retour à 13 et je stoppe le programme.
- Une fois ce délai récupéré, je calcule la date de purge

Entre les fonctions 50 et 70, j'insère le corps du programme. Pour chaque table j'effectue une requête de type SELECT COUNT afin d'écrire en trace et en sysout le nombre d'enregistrements à supprimer puis j'effectue la requête de suppression des enregistrements concernés. Les requêtes en elles-mêmes ont été placées à l'aide de différentes macrostructures selon le type de requête dans des fonctions performées.

A la fin de la boucle Pacbase, le programme passe par la fonction f20 (**cf. annexe 8 : la boucle Pacbase**), les fichiers d'entrée et de sortie sont alors fermés et le programme se termine.

Une fois l'écriture du programme terminée, je procède à la compilation du programme. Cette étape peut être longue et fastidieuse. En effet, le rapport de compilation précisant les erreurs est assez compliqué à déchiffrer et les erreurs ne sont pas toujours signalées de manière explicite.

Ce n'est qu'une fois que ce programme est compilé qu'il est possible de voir le cobol généré dans l'outil Coca.



C'est lors de cette étape que la note PQC est attribuée au programme.

```

Menu Utilities Compilers Help
BROWSE TSOSX11.XCR0573.KBP431EE.S1.QSX11.IPACBA Line 0000000019 Col 001 080
Command ===> _                                         Scroll ===> CSR
 0101     XSJR768  18OCT19 09:57      2165 C0178620    FICOBA AV C0164848
 0102     XSJR768  18OCT19 10:53      2051 C0178620    FICOBA
GENERATED XSJR768  18OCT19 10:53      C0178620    FICOBA

%+0102 *      *PKBP431KBP431 PB12BDK7103T 2019/10/18 10:53:28XSJR768 100
+0100 000100 IDENTIFICATION DIVISION.
+0100 000200 PROGRAM-ID. KBP431.
+0100 000300*AUTHOR.       Traitement de purge FICOBA.
+0101 000400*DATE-COMPILED. 18/10/19.
+0100 000500 ENVIRONMENT DIVISION.
+0100 000600 CONFIGURATION SECTION.
+0100 000700 SOURCE-COMPUTER. IBM-370.
+0100 000800 OBJECT-COMPUTER. IBM-370.
+0100 000900 INPUT-OUTPUT SECTION.
+0100 001000 FILE-CONTROL.
+0100 001100      SELECT      FJ-FICHIER      ASSIGN      UT-S-FIJOUR.
+0100 001200      SELECT      TR-FICHIER      ASSIGN      UT-S-TRACE.
+0100 001300 DATA DIVISION.
+0100 001400 FILE SECTION.

*ISRBROB

```

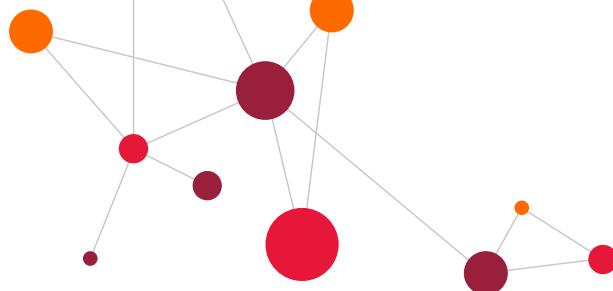
Figure 17 : extrait de l'outil coca

Une fois le programme compilé, je réalise mes tests-unitaires. Lorsque ces derniers ne sont pas concluants, j'ai la possibilité de faire du débogage à l'aide de l'outil Xpediter.

### Conclusion du projet :

Je n'ai pas rencontré de difficultés pour réaliser ce programme. Son aspect fonctionnel relativement basique et sans grosses difficultés techniques (accès direct aux tables DB2, un seul fichier en entrée, une seule boucle Pacbase effectuée) a facilité l'acquisition de compétences dans ce nouvel environnement. La seule complexité a été la vérification de la validité de la date facilitée par l'emploi d'une macrostructure.

De plus, la réalisation des jeux de données pour la phase de tests-unitaires était relativement aisée.



#### 4.1.2

#### Exemple n°2 – GSAF PPH (HJAA24)

Ce programme fait partie du domaine A2G. C'est aussi un programme batch.

J'ai choisi de présenter ce programme car ce dernier a la particularité de réaliser des ruptures sur une table DB2.

#### Expression des besoins :

L'objet de ce programme est de calculer des compteurs d'alertes et de constituer un fichier de compteurs par site pilote. Ces compteurs sont des indicateurs à destination de la Direction des opérations et de la Direction des Risques et Contrôles.

Ce programme doit fonctionner sur cinq périodes différentes :

- Hebdomadaire
- Mensuelle
- Trimestrielle
- Annuelle de l'année en cours
- Annuelle de l'année précédente

#### Analyse :

De par toutes ces caractéristiques souhaitées, c'est un programme jugé complexe par les abaques CGI.

Le schéma du traitement de la chaîne dans laquelle s'insère le programme est disponible en annexe ([cf.annexe 10](#))

#### Réalisation et tests unitaires :

Le paramétrage de la période se fait dans le JCL. Suivant ce paramètre rentré en SYSIN, différents blocs du programme vont alors être exécutés.

```

//*****
//SYSIN   DD  *
DSN SYSTEM(DB2T)
RUN PROGRAM(HJAA24) PLAN(HJPBD0AA)
END
/*
/*
//SYSIN   DD  *
H015
/*
/*
/** Fichier(s) pris en Sortie
//SAD08   DD DSN=PUTUSR.$PERM.CRO573F.HJGSCKTU.CASXX,
//          DISP=(CATLG,DELETE),
//          RECFM=FB,
//          LRECL=100,
//          SPACE=(TRK,(100,50),RLSE),
//          UNIT=3390
//TRACE    DD DSN=PUTUSR.$PERM.CRO573F.HJTRCKTU.CASXX,
//          DISP=(CATLG,DELETE),
//          RECFM=FB,
//          LRECL=80,
//          SPACE=(TRK,(100,50),RLSE),
//          UNIT=3390
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSOUA   DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
/*

```

Figure 18 : Extrait du JCL comprenant la SYSIN (HJAA24)

Un certain nombre de compteurs étaient déjà calculés dans un ancien programme. Ainsi, il a été possible de faire des tests de non-régression.

Concernant les compteurs restants, il m'a fallu créer des requêtes DB2 afin de vérifier les valeurs.

Exemple de requête DB2 pour tester certains compteurs :



The screenshot shows a DB2 SQL editor interface. The title bar says "Connexion : DB2T01B [XCRO573]". The query window contains the following SQL code:

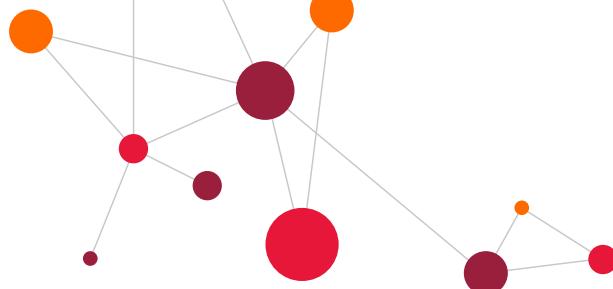
```

49 SELECT COUNT (CCXORH) AS "NB DE DOSSIER D'ORIGINE4 POUR LIMOGES"
50 FROM GDBBHJD0.HJT0AD08
51 WHERE
52 ZDATSC BETWEEN '2019-04-15-00.00.00.000000' AND '2019-04-21-23.59.59.999999'
53 AND CCASDD = 3
54 AND CCXORH = 4
55 AND NOOALC = 'F02291' ;
56

```

The results window shows a single row with the value "6" highlighted in a red box. The results table has one column labeled "NB DE DOSSIER D'ORIGINE4 POUR" and one row with the value "6".

Figure 19 : Exemple de requête DB2 pour TU (HJAA24)



Globalement, l'écriture de ce programme se rapproche du programme présenté précédemment, je vais donc n'en présenter que quelques particularités.

Dans la working-section, je déclare un curseur afin d'exécuter la requête suivante dans la table HJT0AD08.

```

SELECT
NOOALC, CCXORH, CCASDD,
CCXNAD, CCXTOD, CLUTYA,
ZDATSM, ZDATSC, NAGMBN
FROM HJT0AD08
WHERE
ZDATSC >= :WWW0-ZDATSC
AND
ZDATSC <= :WWW1-ZDATSC
ORDER BY
NOOALC, CCXORH, NAGMBN, CCASDD DESC
WITH CS

```

L'algorithme va nécessiter trois ruptures.

Les ruptures se déclarent en même temps que la déclaration des segments (en -ce).

```

VA Pac 3.5 V04 LBP SECURITE - SECURISATION A2G (HJ) *XCR0573.PB12.B17.7155
STRUCTURES DE DONNEES DU PROGRAMME HJAA24 CALCULS COMpteURS GSaf PPH

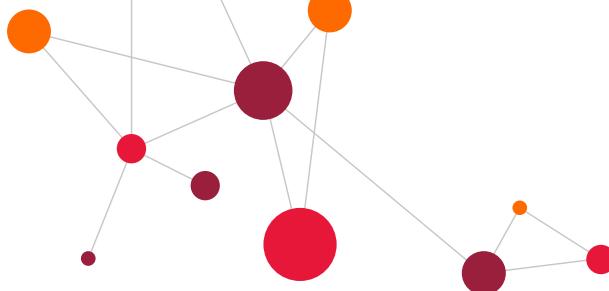
A PR SU : SD EXTERN OAMOU BLOC T R S U RE SE M UNIT P SELECTION F O D N EM
  BR : PF TRACE SSFOU 0 R D *09=00 I 1
  : Z.COMPL. : CLE ACC. : IDENT. :
  SA : PF SAD08 SSFOU 0 R D *20=00 I 1
  : Z.COMPL. : CLE ACC. : IDENT. :
  T1 : AD H108 2SFIU 0 R 3 C ABC I 1 1
  : Z.COMPL. : CLE ACC. : IDENT. :
  T1 SU : AD H108 2 0 C *08=00 I 1 1
  : Z.COMPL. : CLE ACC. : IDENT. :
  :
  : Z.COMPL. : CLE ACC. : IDENT. :
  :
  : Z.COMPL. : CLE ACC. : IDENT. :
  :
  : Z.COMPL. : CLE ACC. : IDENT. :
  :
  : Z.COMPL. : CLE ACC. : IDENT. :
  :
  : Z.COMPL. : CLE ACC. : IDENT. :

O: C1 CH:

```

Figure 20 : Déclaration des ruptures (HJAA24)

On précise les rubriques à déclarer en rupture à avec une lettre (A, B et C) que l'on reporte directement sur le segment AD08.



```

VA Pac 3.5 V04    LBP SECURITE - SECURISATION A2G (HJ)    *XCR0573.PB12.B17.7155
DESCRIPTION DU SEGMENT : AD08 TABLE ALERTE

A NLG : CORUB FORM.INT. U OCC GR I CMS456 CRNS VALEUR/FCT MAJ/TABLE DOC BIBLI
010 : NLUALA          0           4434
020 : NEPCSS5         0           4521
030 : NAGMBN          C P        6621
050 : NOOALC          A P        6621
070 : CCASDD          P          4434
090 : CUGCPF          P          4434
110 : CCXTOD          P          4434
130 : CCXNAD          P          4434
150 : LCACJN          W P        4434
170 : CCXORH          B P        6621
190 : CMITCE          P          4434
210 : CADIP           P          4434
230 : CMITCF          P          4434
250 : NMIDEA          P          4434
270 : CLUTYA          P          4434
290 : CLUSF           P          4434
310 : DASMPA          P          4434
: LIBELLE :          :           4434

O: C1 CH:

```

Figure 21 : déclaration du segment AD08

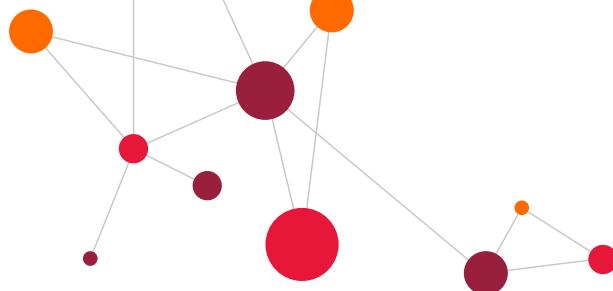
Dans ce segment, nous avons déclaré une rupture sur la rubrique NOOALC (site de traitement Alerte), sur la rubrique CCXORH (Code type Origine) et sur la rubrique NAGMBN (identifiant agent investigateur).

### Conclusion du projet :

J'ai rencontré quelques difficultés à réaliser ce programme que j'attribue :

- À la complexité de calcul des dates suivant la période souhaitée.
- Aux ruptures difficiles à appréhender.
- À un problème d'analyse concernant quatre compteurs.

De plus, la création des jeux de données pour ces tests a été assez complexe et fastidieuse.



#### 4.1.3 Exemple n°3 – Extraction des FGR RTG vers MDG (LDC131)

Ce programme fait partie du domaine RSK et plus précisément de l'application ARP (Application Risque personne). Cette application assure au sein du SI du client, la centralisation d'évènements de risque pour les personnes physiques (PPH) et les personnes morales (PMO). Ces évènements sont communément appelés Faits Générateurs de Risques (FGR). Pour cela, ARP est alimenté quotidiennement par des informations contrat ou client gérées par d'autres applications ou bien mises à jour directement via une IHM (Interface Homme Machine).

Un programme similaire existait déjà dans le SI du client mais une refonte totale a été imaginée afin d'inclure également les filiales de la Banque Postale.

Dans cet exemple, je vais essentiellement présenter le cobol généré.

#### Expression des besoins :

Ce programme consiste à extraire des informations provenant de deux tables :

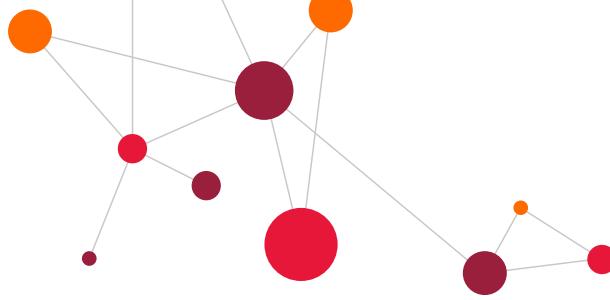
- RP01 : table des faits générateurs de risques courants
- RH01 : table des faits générateurs de risques historisés

Ces extractions vont donner lieu à l'élaboration d'un fichier en sortie à destination d'un autre programme (MDG).

Les règles de gestion sont les suivantes :

Extraction des FGR ND vers MDG

Règle	Description
RG01	Le traitement d'extraction des FGR ND permet de produire des flux CRE d'ouverture ou de fermeture d'incident en fonction de la date de début d'alimentation contagion (MDG), de la date comptable de traitement, de la date de début de FGR, des données attribut et du type de FGR.
RG02	Contagion n'a pas la notion de FGR, seulement d'incidents. Il faut donc notifier tout changement sur ces incidents à MDG via émission de CRE.
RG03	Un incident est calculé en fonction du type FGR et de son attribut grâce à la table Armide LDFGRRRTG
RG04	Le jour de l'ouverture CRE est activé pour un incident (Date du jour est égale à la date de début de contagion en LDFGRRRTG pour un couple (FGR / Attribut) donné) tous les incidents créés avant la date comptable donnent lieu à l'émission d'un CRE d'ouverture.
RG05	Lors d'une modification détectée sur les FGR un CRE fermeture sur l'ancien incident puis un CRE d'ouverture sur le nouvel incident sont envoyés
RG06	Lorsqu'une création et une suppression sont détectées sur les FGR, aucun CRE n'est envoyé.
RG07	Le type de personne et les identifiants locaux des filiales ne sont pas alimentés, ils sont déterminés par MDG.
RG08	Pour chaque couple de la table LDFGRRRTG lu et dont la date d'alimentation pour contagion inférieure ou égale à la date fonctionnelle de traitement un enregistrement « Compteur CSV » doit être écrit afin de répertorier les CRE d'ouverture et de fermeture.



## Analyse :

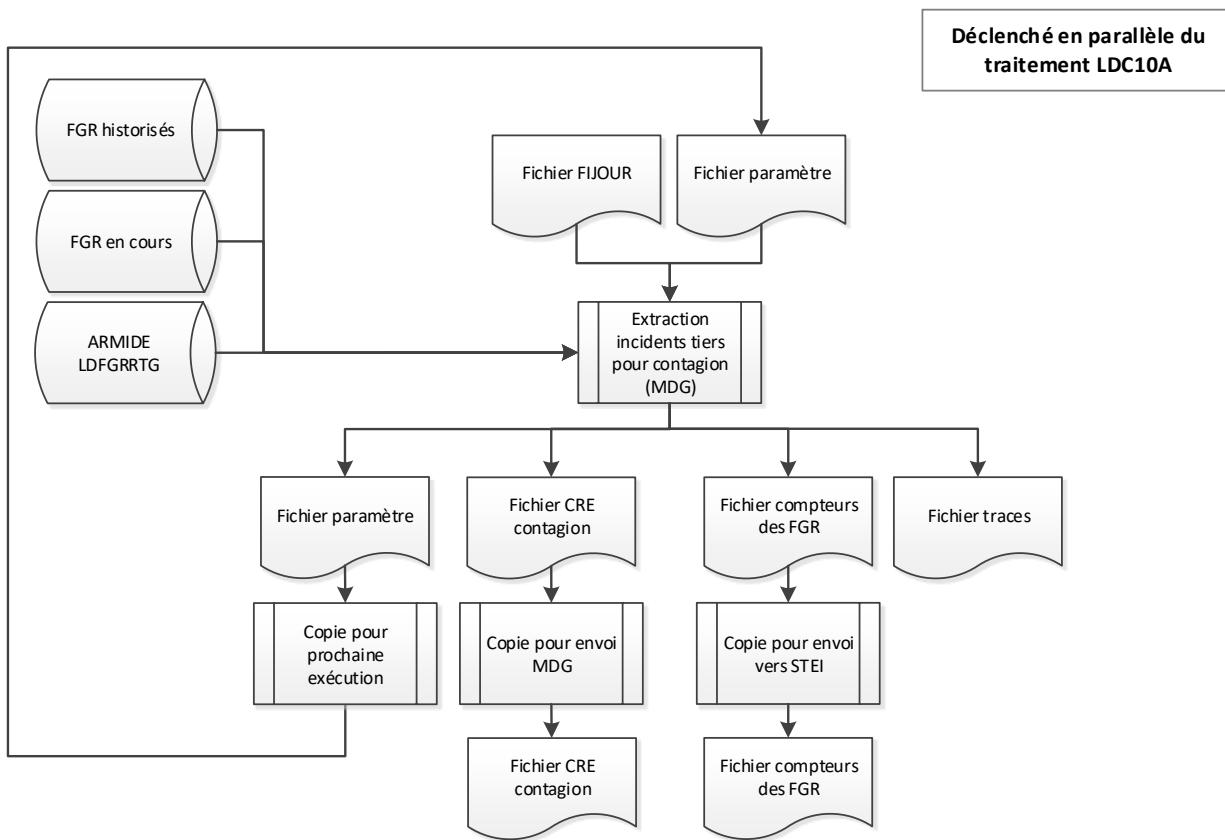
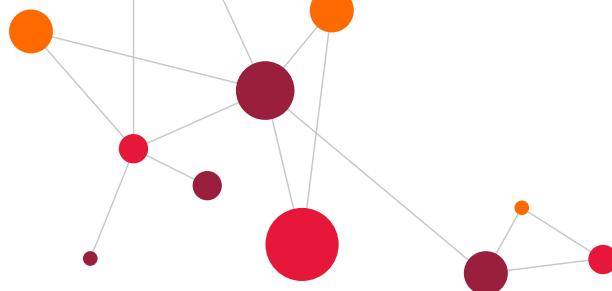


Figure 22 : schéma du traitement LDC131



### Réalisation et tests unitaires :

Dans la working-section, j'ai déclaré de nombreux compteurs afin de comptabiliser les ouvertures et fermetures de CRE :

```

053800 01 WC00-ZGROUA.
053900 10 WC00-ZX044A VALUE 'NB CRE OUVERTURE FGR ECRITS'
054000           PICTURE X(44).
054100 10 WC00-QRQFC VALUE ZERO
054200           PICTURE 9(9).
054300 10 WC00-ZX044B VALUE 'NB DE DATES DEBUT FGR NON-RENSEIGNNEES'
054400           PICTURE X(44).
054500 10 WC00-QRQDZ VALUE ZERO
054600           PICTURE 9(9).
054700 10 WC00-ZX044C VALUE 'NB DE CRE OUVERTURE FGR ECRITS'
054800           PICTURE X(44).
054900 10 WC00-QRQFC1 VALUE ZERO
055000           PICTURE 9(9).
055100 10 WC00-ZX044D VALUE 'NB DE DATES DEBUT FGR NON-RENSEIGNNEES'
055200           PICTURE X(44).
055300 10 WC00-QRQDZ1 VALUE ZERO
055400           PICTURE 9(9).

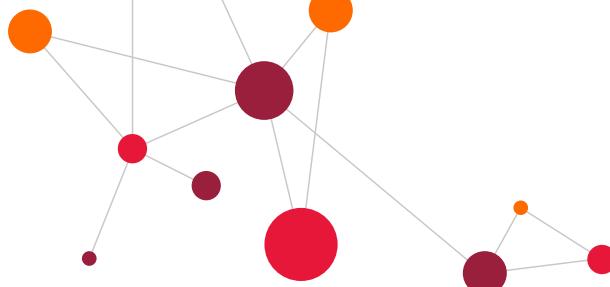
```

Ce programme a nécessité la création de 5 curseurs. La déclaration de ces curseurs se fait également dans la working-section.

```

155100*****EXEC SQL DECLARE CURS5-RH01
155200*****
155300*****      CURSOR
155400*****      FOR SELECT
155500*****      DRQDF,NTRIDG,ZDATSM,ZDATSC,ZDATSS,DRQDT,CRQFE, NRQVE
155600*****      FROM
155700*****      LDT0RH01
155800*****      WHERE
155900*****      CRQFG =      :WW00-CRQFG
156000*****      AND CRQARA = :WW00-CRQARA
156100*****      AND ZDTRA7 > :WWW2-ZDATS
156200*****      AND ZDATSC < :WWW1-ZDATS
156300*****      ORDER BY
156400*****      NTRIDG,ZDTRA7 DESC,NRQVE
156500*****      WITH CS
156600*****      FOR FETCH ONLY
*****END-EXEC.

```



- Entre les fonctions f01 et f04, je code les vérifications.

Par exemple, la vérification de la présence de contenu dans le fichier FIJOUR :

```

170400 F01DA.
170500   OPEN INPUT      DA-FICHIER
170600   MOVE 0 TO IK
170700   READ          DA-FICHIER
170800   AT END MOVE 1 TO IK.
170900     IF    IK = 1
171000   PERFORM    F01TR THRU F01TR-FN
171100   MOVE        'FICHIER LDDATEAA EN ENTREE VIDE
171200-   '' TO TT10-ZX200
171300   PERFORM    F91TT THRU F91TT-FN
171400   PERFORM    F2098 THRU F2098-FN
171500   MOVE        13 TO WSX0-ZCSRT7
171600   PERFORM    F99SX THRU F99SX-FN.
171700 F01DA-FN. EXIT.

```

En effet le fichier DA00 correspond au fichier FIJOUR déclaré en (-ce).

Après ouverture, si le fichier est vide, alors l'indicateur IK généré par Pacbase est alimenté à 1. J'écris donc 'Fichier LDDATEAA en entrée vide dans la sysout et la trace et j'alimente le code retour à 13.

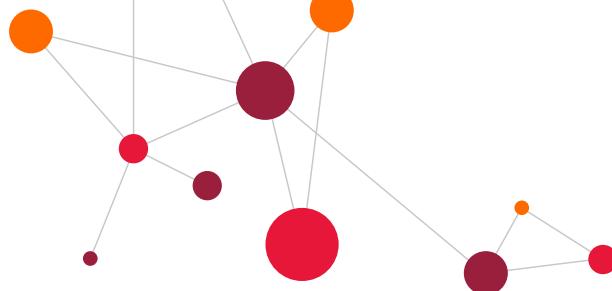
- En fonction f50, j'ai codé les écritures d'en-tête des trois fichiers de sortie (Fichier CRE, fichier CSV des compteurs et fichier TRACE).

Par exemple, le cobol généré pour l'en-tête du fichier CSV est le suivant :

```

197000   INITIALIZE SC00
197100   STRING   'DATE;FGR;ATTRIBUT;INIT - NB '
197200   'CRE OUVERTURE;INIT - NB '
197300   'DATE DEBUT FGR A ZERO;CREATION'
197400   ' - NB CRE OUVERTURE TOTAL;'
197500   'CREATION - NB CRE OUVERTURE '
197600   'A ZERO;MODIF - NB MODIF DATE '
197700   'DEBUT FGR;MODIF - NB CRE '
197800   'FERMETURE;MODIF - NB CRE '
197900   'OUVERTURE;MODIF - NB MODIF '
198000   'DATE DEBUT ATTRIBUT;MODIF - '
198100   'NB CRE FERMETURE;MODIF - NB '
198200   'CRE OUVERTURE; MODIF - NB '
198300   'MODIF ATTRIBUT;MODIF - NB '
198400   'CRE FERMETURE PRP;MODIF - '
198500   'NB CRE OUVERTURE PRP; SUPPR - '
198600   'NB CRE FERMETURE TOTAL'
198700   DELIMITED BY SIZE INTO SC00
198800   PERFORM    F90SC THRU F90SC-FN.
198900 F50BG-FN. EXIT.

```



L'écriture de l'en-tête du fichier TRACE est le suivant :

```

199000 F50DA.
199100 MOVE      WA00-ZX200 TO WB00-ZX200
199200 MOVE      'DATE PARAMETRE LUE' TO WB00-ZX068
199300 MOVE      DA00-ZDATE (7 :2) TO WB00-JJ
199400 MOVE      DA00-ZDATE (5 :2) TO WB00-ZDMOI
199500 MOVE      DA00-ZDATE (1 :4) TO WB00-SSAA
199600 MOVE      WB00-ZX200 TO TT10-ZX200
199700 PERFORM   F91TT THRU F91TT-FN
199800 MOVE      WA00-ZX200 TO WB00-ZX200
199900 MOVE      'DATE PARAMETRE INSCRITE' TO
200000 WB00-ZX068
200100 MOVE      EA00-ZDTRAR (7 :2) TO WB00-JJ
200200 MOVE      EA00-ZDTRAR (5 :2) TO WB00-ZDMOI
200300 MOVE      EA00-ZDTRAR (1 :4) TO WB00-SSAA
200400 MOVE      WB00-ZX200 TO TT10-ZX200
200500 PERFORM   F91TT THRU F91TT-FN.
200600 F50DA-FN. EXIT.

```

- La principale fonction de ce programme correspond à la fonction f51.

J'y ai effectué une boucle de parcours de la table Armide et pour chaque couple de FGR, j'effectue les contrôles nécessaires et selon les cas, j'alimente le fichier CRE avec des lignes d'ouverture et de fermeture ainsi que les compteurs adéquats.

**L'algorithme de cette fonction engendre de nombreuses conditions imbriquées (cf. annexe 9). Il est difficile de s'y repérer dans le cobol généré car il n'y a pas d'indentations.**

```

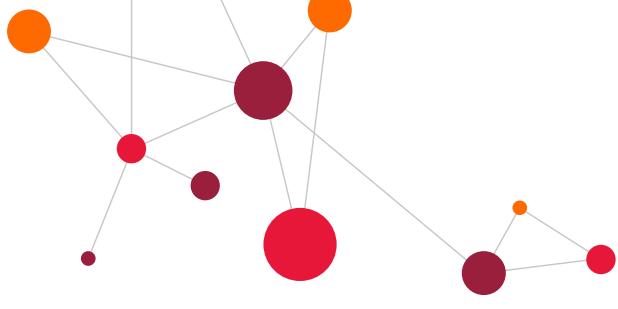
VA Pac 3.5 V04 Application Risque Personne (LD)          *XCR0573.PB12.B14.7155
TRAITEMENTS PROGRAMME LDC131 Extraction FGR RTG VERS MDG  FONCTION: 51

A SS NLG OPE SOURCE PROGRAMME
- AD (P) N  PARCOURIR TOUT LE DÉCHARGEMENT 10D01 WFA0-ZNENR
- BD (P) N  SI TOP ALIMENTATION = 0 15ITFA8C-BRQAC(J51ADR) = '0'
- BE (P) N  INITIALISATION DE CONTAGION 20ITDA00-ZDATE =
- BF (P) N  CURSEUR 1 : FGR CREEES DEPUIS LE 25BL
- BH (P) N  BOUCLE DE FETCH SUR LE CURSEUR 1 30DWIK = '0'
- BL (P) N  DATE DE DÉBUT DU FGR = 0 35ITC101-DRQDF = '01.01.0001'
- BP (P) N  DATE DE DÉBUT DU FGR <> 0 35ITC101-DRQDF NOT = '01.01.0001'
- BT (P) N  FGR BPE/SOFIAP 40ITFA8C-CRQFG(J51ADR) NOT =
- BV (P) N  FGR BPE/SOFIAP 40ITFA8C-CRQFG(J51ADR) =
- DD (P) N  PLAN RECEVABLE 45ITFA8C-CRQARA(J51ADR) = 'A0'
- DL (P) N  PLAN DE REDRESSEMENT 45ITFA8C-CRQARA(J51ADR) = 'A1'
- DP (P) N  PLAN DE REDRESSEMENT 45ITFA8C-CRQARA(J51ADR) = 'A1'
- DT (P) N  LECTURE SUIVANTE 35BL
- DV (P) N  FERMETURE DU CURSEUR 1 25BL
- FD (P) N  ALIMENTATION QUOTIDIENNE 20ITDA00-ZDATE >
- FF (P) N  CURSEUR 2 : CREATION 25BL
- FH (P) N  BOUCLE DE FETCH SUR LE CURSEUR 2 30DWIK = '0'
- FL (P) N  DATE DE DEBUT FGR = 0 35ITC201-DRQDF = '01.01.0001'

O: C1 CH:

```

Figure 23 : Extrait des sous-fonctions de la fonction f51



Après compilation, je réalise mes tests-unitaires. Lorsque ces derniers ne sont pas concluants, j'ai la possibilité de faire du débogage à l'aide de l'outil Xpediter. Cet outil m'a été particulièrement utile dans ce programme. Je marque ainsi mes points d'arrêts au niveau des fonctions souhaitées puis je fais du pas à pas pour vérifier si mon programme passe dans le bloc concerné. Si c'est le cas, j'ai la possibilité de consulter les valeurs de mes variables aux moment souhaités.

Le JCL permettant de lancer ce programme batch est disponible en annexe (cf. annexe 11 : JCL LDC131).

### **Conclusion du projet :**

Ce projet a été difficile à appréhender par l'aspect fonctionnel complexe mais également par l'algorithme. Le parcours de la table Armide oblige à écrire une bonne partie du code dans une seule fonction ne facilitant pas sa lecture.

### **4.1.4 Exemple n°4 – Service Applicatif – Mise à jour des FGR (RISCLI067)**

Ce service applicatif concerne également l'application ARP. Il est lié au programme présenté ci-dessus (LDC131) car il consiste en la mise à jour de la table des faits générateurs de risques courants (RP01).

#### **Principe général d'un Service applicatif (SA) :**

**Un service Applicatif est un service métier organisé en couches et utilisant un système de transactions.**

**A la Banque Postale, certaines règles interdisent les échanges Informatique Centralisée (IC)/ Informatique Distribuée (ID) entre deux applications différentes. Il est également interdit d'utiliser la base de données DB2 d'une autre application.**

**Afin de pallier à ces contraintes, on passe par des Services Applicatifs (SA).**

**Un SA est composé de différents modules. Le schéma général est le suivant :**

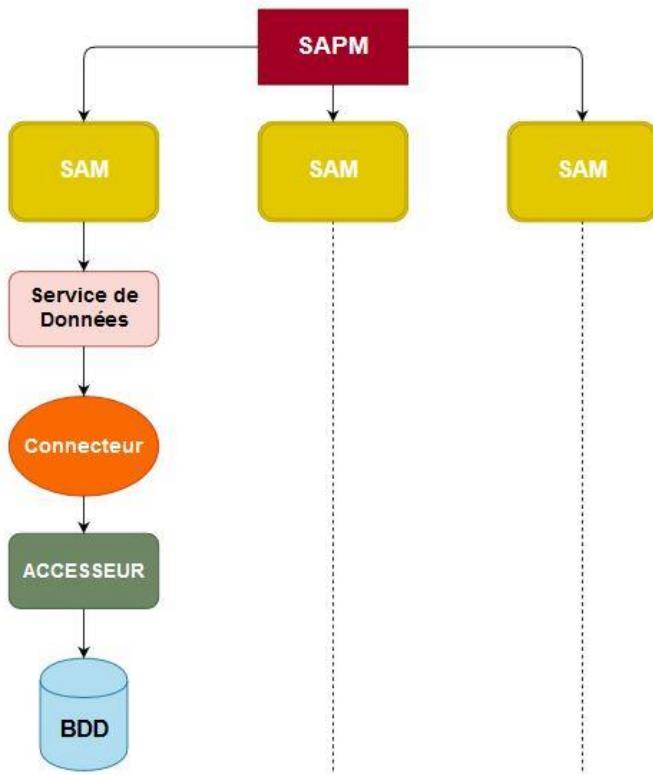
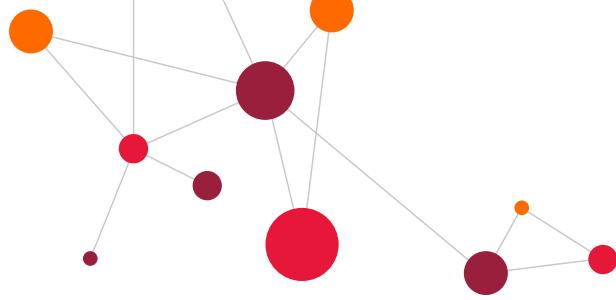
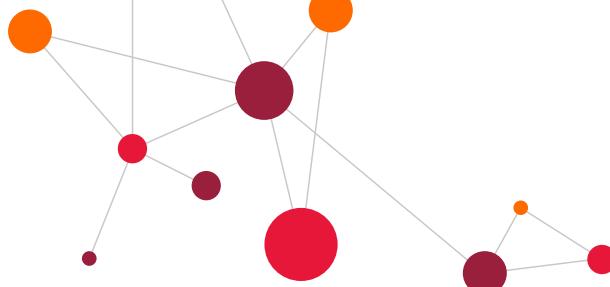


Figure 24 : schéma général d'un Service Applicatif Processus Métier (SAPM)

**Le SAPM (Service Applicatif Processus Métier) correspond au point d'entrée du Service Applicatif. Il est considéré comme l'ordonnanceur de processus métier. Un SAPM permet de répondre à une question complexe décomposable en plusieurs questions, surtout lorsqu'il est nécessaire d'accéder à plusieurs tables. Le SAPM fait alors appel à plusieurs SAM (Service Applicatif Métier). Chaque SAM correspond ainsi à un seul bloc fonctionnel et possède son SD (Service de Données) ainsi que son connecteur et son accesseur. Le SD fournit un accès logique en consultation ou en mise à jour aux données. L'accesseur réalise l'accès physique aux données.**

**Ce type de programme permet la communication entre un opérateur et le SI selon un principe de question/réponse. Cet opérateur réalise des appels via une couche de présentation distante (IHM).**



### Expression des besoins :

Ce programme doit permettre de créer, modifier ou supprimer un fait générateur de risque en fonction du code mouvement fourni en entrée du service.

Le Service Applicatif à mettre en œuvre a la particularité d'être destinée à être réutilisable par n'importe quel projet du client (module NK). Le principe de fonctionnement est similaire mais il y a des contraintes supplémentaires tels que l'attribution des noms des composants et des segments utilisés, la validation du DCOD par une cellule spécialisée etc...

La présence d'un SAPM n'est pas requise dans cet exemple car le SA fait appel à une seule table.

### Analyse :

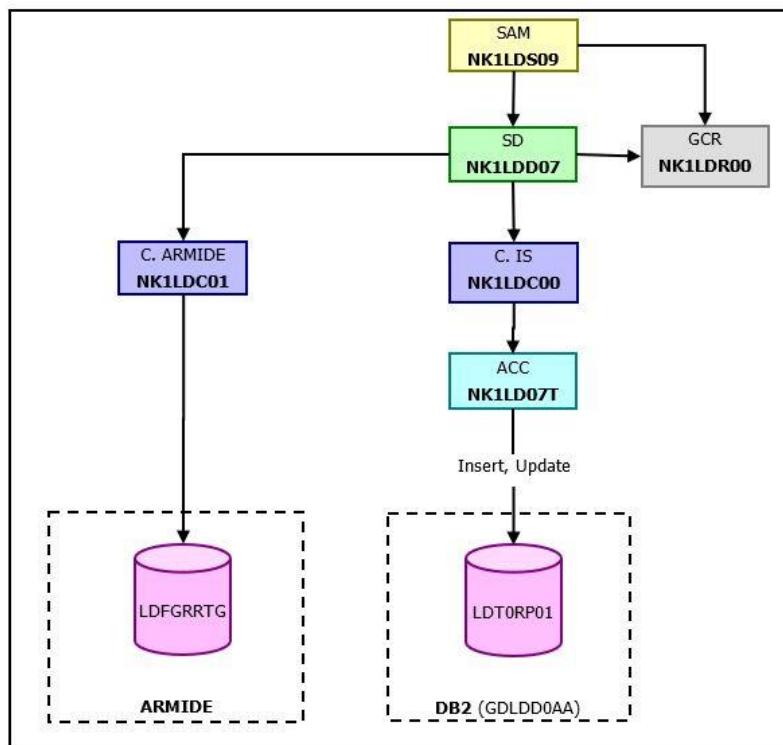
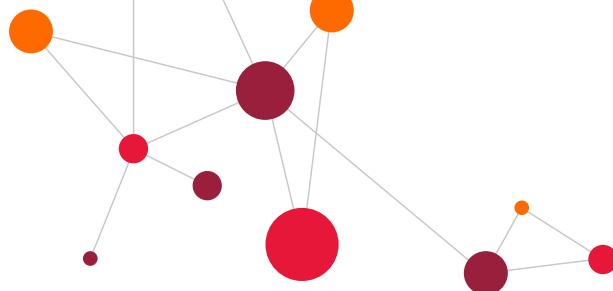


Figure 25 : schéma général du RISCLI067



Le diagramme de séquence est le suivant :

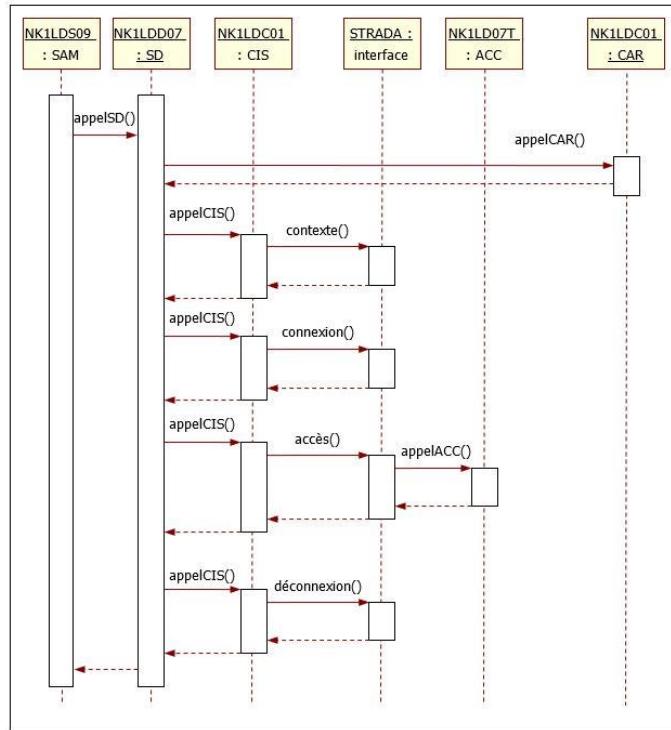
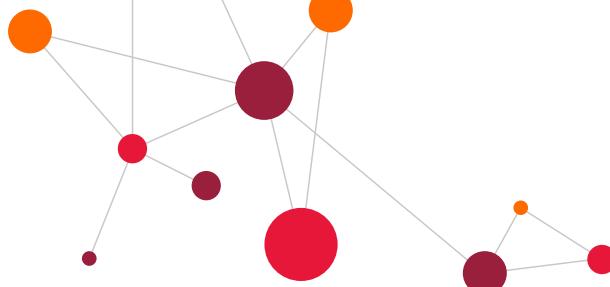


Figure 26 : diagramme de séquence RISCLI067



## Réalisation et tests unitaires :

Le point d'entrée se situe au niveau du SAM.

Les contrôles des données en entrée se font principalement dans ce composant excepté les codes retour 0106 et 0110 qui ont été codés dans le SD car ils nécessitent d'avoir la connexion à la table Armide pour faire les vérifications (cf.figure 25).

ZCPIK	ZCRET	ZCRUT	Libellé du code retour
<b>0</b>	<b>00 Traitement normal</b>		
		0000	Traitement normal
<b>1</b>	<b>01 Critères invalides</b>		
		0101	N° de version du SA absent
		0102	Code mouvement absent
		0103	Code mouvement incorrect
		0104	Identifiant RTG absent
		0105	Code fait générateur de risques absent
		0106	Code fait générateur de risques incorrect
		0107	Date de début du fait générateur de risques incorrecte
		0108	Date de fin du fait générateur de risques incorrecte
		0109	Date de fin du fait générateur de risques inférieure à la date de début du fait générateur de risques
		0110	Code attribut du fait générateur de risques incorrect
		0111	Date de début de l'attribut du fait générateur de risques incorrecte
		0112	Date de fin de l'attribut du fait générateur de risques incorrecte
		0113	Date de fin de l'attribut du fait générateur de risques inférieure à la date de début de l'attribut du fait générateur de risques
		0114	Top attribut du fait générateur de risques incorrect
		0115	Montant attribut du fait générateur de risques incorrect
		0116	Code user absent
		0117	Code entité affectation fonctionnelle acteur SF absent
		0118	Le code application DPI absent
		0119	Date de notification du fait générateur de risques incorrecte

Entre les différents modules, les informations transitent via des segments.

Les différents segments utilisés sont :

- Le segment critère : il est composé de toutes les rubriques saisissables par l'utilisateur.
- Le segment réponse : il s'agit du segment contenant les informations de la base qui vont être restituées à l'utilisateur.
- Le segment chapeau : ce segment est utilisé pour le transfert des données lors d'un appel. Il regroupe le segment critère, le segment réponse et des segments techniques.

Le traitement principal du code se situe dans l'accesseur. Il correspond à la mise à jour de la table DB2 en fonction du code mouvement (données en entrée).

```
V A Pac 3.5 V04 Application Risque Personne (LD) *XCR0573.PB12.B14.7146
TRAITEMENTS ECRAN O LDAC07 ACC RISCLI067 FONCTION: 35

A SS NLG OPE OPERANDE NVTY CONDITION
* BJ N CREATION RP01 15IT UB11-ZCTTP1 = 'C'
* BJ 50 XW RP01
* BJ 60 M '02' UB11-ZCRET UB11-ZCRSQ1
* BJ 62 M SQLCODE UB11-ZCPIK
* BJ 65 M '1' UB11-ZCPIK
* BJ 70 M 'E' CATG
* BJ 75 M '0201' UB11-ZCRUT
* BJ 76 * AUTRE ERREUR : INSERTION
* BJ 190 GT 10
* BJ 200 P F81ES 99IT SQLCODE NOT = 0
-----
* BM N ANNULATION RP01 15IT UB11-ZCTTP1 = 'A'
BM 50 * SUPPRESSION APPEL
* BM 50 XD RP01
* BM 60 M '02' UB11-ZCRET UB11-ZCRSQ1
* BM 62 M SQLCODE UB11-ZCPIK
* BM 65 M '1' UB11-ZCPIK
* BM 70 M 'E' CATG

O: C1 CH:
```

Figure 27 : Extrait de la fonction f035 de l'accesseur (RISCLI067)

Dans l'exemple de la figure ci-dessus, en fonction f035BJ, il est écrit :

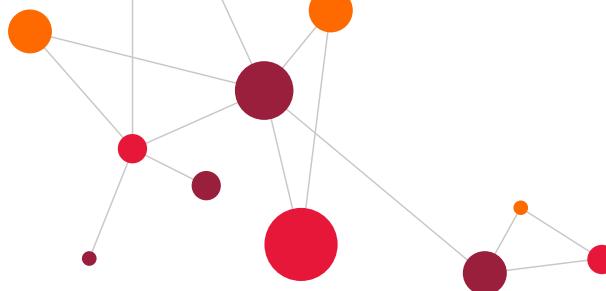
Si le code mouvement en entrée est 'C' (création), alors on fait une requête insert (XW sur le segment RP01). Cette requête se situe en fonction performée (f80).

Si la requête SQL retourne un SQLCODE = -803, alors cela signifie qu'il existe déjà en table des données concernant ces clés.

On alimente alors différents codes retour à partir des informations du DCOD (ZCRET= 02, ZPIK = 1, ZCRUT=0201).

ZCPIK	ZCRET	ZCRUT	Libellé du code retour
1	02 Réponse introuvable		
		0201	Création impossible, le fait générateur de risques existe déjà en table
		0202	Modification impossible
		0203	Suppression impossible

On alimente également l'indicateur CATG à 'E' afin d'éviter de rentrer dans d'autres blocs. Le programme va alors se terminer.



Concernant les tests-unitaires, en l'absence d'IHM spécifique à notre programme, on utilise un outil développé par le client appelé SALEPH. Afin de mimer le comportement de notre service applicatif, on charge sur cet outil des copy-cobol :

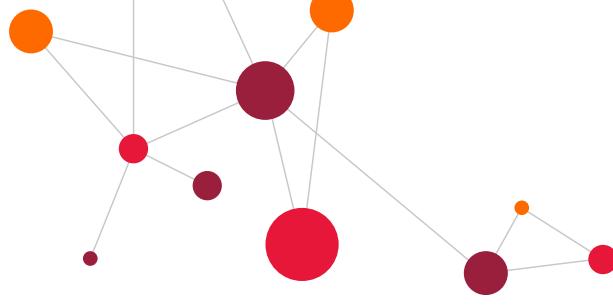
- La copy-cobol critères correspond aux champs à renseigner.
- La copy-cobol réponse correspond aux champs que l'on obtient en retour
- La copy cobol initialisation correspond aux données rentrées par l'utilisateur

Figure 28 : L'outil SALEPH

Après avoir validé, le Service Applicatif va s'exécuter et nous renvoyer une réponse.

Figure 29 : Réponse SALEPH (RISCLI067)

Dans la figure 29, la réponse nous retourne un code 0000, cela signifie que l'insertion a bien été exécutée.



## **Conclusion du projet :**

Ce type de projet est intéressant à réaliser. On y retrouve des éléments de développement d'application N-Tiers.

## **4.2 En formation**

Au cours de cette année de formation, outre les cours dispensés, nous avons réalisé deux projets majeurs en groupe :

- Une application web (site web – projet Dev'Hire).
- Un système d'information comprenant une application web, une application mobile, une API et une BDD (projet AndroidWorkshop).

### **4.2.1 Le projet Dev'Hire**

Dans ce projet, nous étions un groupe de quatre étudiants.

#### **Expression des besoins :**

Les besoins du projet étaient relativement simples et ouverts :

- Concevoir et développer une application web en PHP en utilisant une base de données.

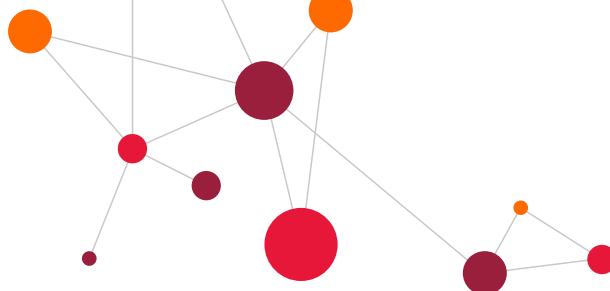
#### **Conception et Analyse :**

Après une séance de brainstorming, nous avons opté pour la création d'un site web d'une agence d'intérim spécialisée dans le développement informatique.

Au départ, nous avons imaginé de nombreuses fonctionnalités à notre application :

- Possibilité pour un candidat et une entreprise de s'inscrire à l'aide d'un formulaire.
- Un candidat doit pouvoir déposer son CV sur la plateforme.
- Une entreprise doit pouvoir déposer une annonce.
- Candidat et entreprise doivent avoir leur espace particulier.
- Possibilité pour un candidat de postuler en ligne
- Mise en place d'un moteur de recherche d'annonces par ville.
- Mise en place d'un système de notation pour l'entreprise et pour le candidat.
- Présence d'un espace administrateur permettant de gérer une annonce, de valider ou dévalider l'inscription d'un candidat ou d'une entreprise, de gérer la partie facturation, de gérer la partie paie.

Nous avons pour cela :



- Maquetté l'application web à l'aide du logiciel de mockup Balsamiq.
- Conçu une base de données en utilisant la méthode Merise (cf. annexe 13). Un extrait du script SQL de génération de la base se trouve en annexe (cf. annexe 14).
- Crée des diagrammes d'activités et de séquences (UML) (cf. annexe 15).

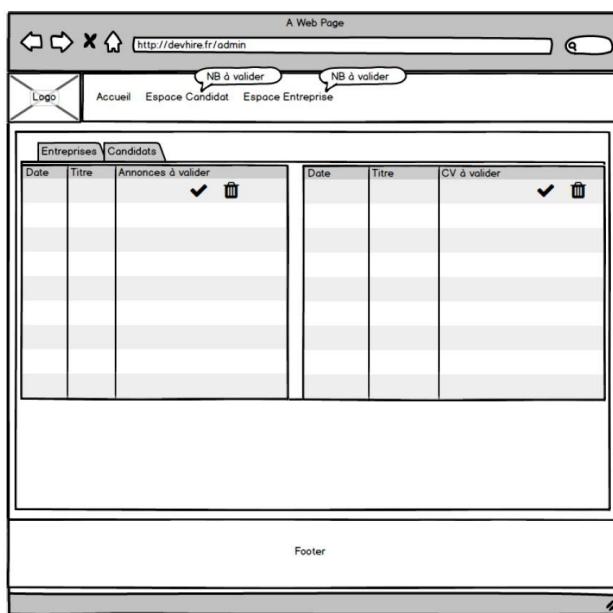


Figure 30 : Extrait de la maquette du site web Dev'Hire

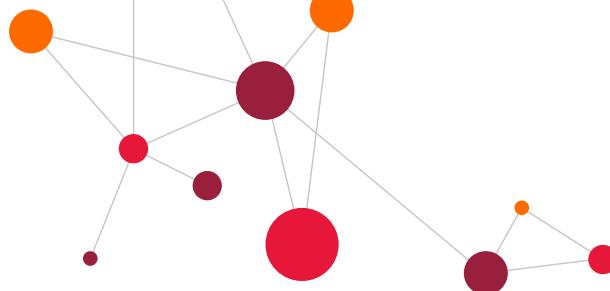
Afin de garantir une bonne organisation du projet, nous avons mis en place différents outils :

- Slack comme plateforme de collaboration afin de faciliter la communication.
- GitHub comme solution de partage de sources et gestion des versions.
- Trello pour la partie gestion de projet.

## Réalisation :

Dans ce projet, nous avons utilisé les langages suivants :

- HTML5/CSS3/Javascript pour la partie Frontend.
- PHP 7.2.1 couplé à une base de données MySQL 5.5.41 pour la partie Backend



### La partie frontend :

Nous avons utilisé le framework Bootstrap pour la partie design et notamment pour ses caractéristiques responsives avec son système de grille.

```
<!-- Search Bar -->
<div class="container-fluid" id="blocksearchbar">
  <div class="col-md-6 col-sm-12 col-xs-12" id="searchbar">
    <form methode="get" action="listeAnnonces.php" class="navbar-form formsearchbar" role="search">
      <div class="input-group add-on">
        <input class="form-control inputsearch" placeholder="Recherche" name="recherche" id="srch-term"
               type="text" required="required" value=<?php echo isset($_GET["recherche"]) ? $_GET["recherche"] : '' ;?>">
      </div>
    </div>
    <div class="col-md-6 col-sm-12 col-xs-12" id="ville">
      <div class="input-group add-on" >
        <input class="form-control inputsearch" placeholder="Ville" name="ville" id="ville"
               type="text" required="required" value=<?php echo isset($_GET["ville"]) ? $_GET["ville"] : '' ;?>">
      <div class="input-group-btn" >
        <button class="btn btn-default" id="boutonRecherche"
               type="submit"><i class="glyphicon glyphicon-arrow-right accueil" ></i></button>
      </div>
    </div>
  </div>
</form>
</div>
```

Figure 31 : Extrait du code de la page d'accueil

La majeure partie des vérifications des champs des formulaires ont été réalisées en Javascript.

```
//Vérification remplissage des champs avant validation du formulaire!

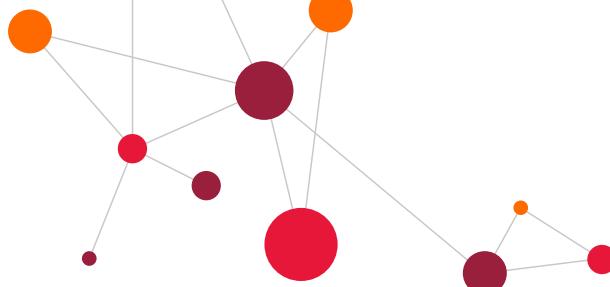
function fonverif(){
  var ttNom=document.getElementById("champNom");
  if(ttNom.value.trim()==''){
    alert("Vous devez indiquer votre nom ou votre raison sociale.");
    ttNom.focus();
    return false;
  }

  var ttEmail=document.getElementById("champEmail");
  if(ttEmail.value.trim()==''){
    alert("Vous devez indiquer votre adresse e-mail.");
    ttNom.focus();
    return false;
  }

  var ttSujet=document.getElementById("champSujet");
  if (ttSujet.selectedIndex==0){
    alert("Vous devez préciser le sujet de votre requête.");
    ttNom.focus();
    return false;
  }

  var ttMessage=document.getElementById("champMessage");
  if(ttMessage.value.trim()==''){
    alert("Vous n'avez pas rédigé votre message!")
    ttMessage.focus();
    return false;
  }
  return true;
}
```

Figure 32 : Code Javascript du formulaire de contact



## La partie Backend

Je vais présenter la partie Administration, partie que j'ai moi-même développée.

Pour accéder à cette partie, il faut se connecter en tant qu'administrateur.

On y accède par l'url suivante : 'devhire/adminlog.php' afin de se connecter à la session administrateur. Une fois l'identifiant et le mot de passe renseignés, nous sommes redirigés vers la page 'devhire/admin.php'.

Cette page permet de gérer trois fonctionnalités à partir d'onglets :

- L'activation d'une annonce
- Le statut des candidats
- Le statut des entreprises

Date publication	Titre	Ville	Statut	Action
2018-11-28	expert cobol	Paris	Desactive(e)	<input checked="" type="checkbox"/> <input type="checkbox"/>
2018-11-28	developpeur front	Toulouse	Active(e)	<input checked="" type="checkbox"/> <input type="checkbox"/>
2018-11-28	developpeur java	Bordeaux	Desactive(e)	<input checked="" type="checkbox"/> <input type="checkbox"/>
2018-11-28	chef de projet	Marseille	Desactive(e)	<input checked="" type="checkbox"/> <input type="checkbox"/>
2018-11-28	developpeur back	Versailles	Active(e)	<input checked="" type="checkbox"/> <input type="checkbox"/>

Figure 33 : page administrateur du site web Dev'Hire/ onglet 'Gérer Annonces'

Je vais par exemple vous présenter l'onglet 'Gérer Annonces'.

L'affichage des données du tableau se fait avec la fonction suivante :

```
//AFFICHAGE DES DONNEES ANNONCES

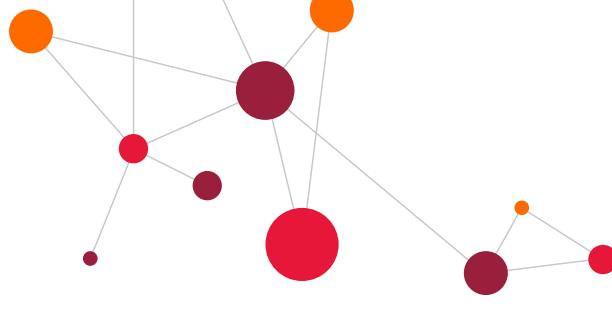
function getAllAdvert(){
    $dbConn=Database::connect();

    $query='SELECT id_Annonce as id, libelleAnnonce,datePublication, descriptionAnnonce, ville, statut.libelleStatut as statut
           FROM annonce
           inner join statut on statut.id_Statut=annonce.id_Statut
           ORDER BY datePublication desc';

    $statement=$dbConn->query($query);
    $result = $statement->fetchAll(PDO::FETCH_CLASS);
    $statement->closeCursor();

    header('Content-Type: application/json');
    echo json_encode($result);
}
```

Figure 34 : fonction getAllAdvert() (Devhire)



Il est possible d'activer ou désactiver une annonce ou même de cliquer sur une ligne du tableau afin d'afficher la description de l'annonce.

J'ai développé ces fonctionnalités à partir de fonctions php allant requêter sur la base. Pour récupérer l'id de la ligne du tableau concernée, j'ai utilisé les fonctions click(), closest() et attr() de la bibliothèque Jquery.

```

function initEvents(){

    //Pour l'activation d'une annonce
    $(".btnActivedAdvert").click(function(event){
        idAnnonce=$(this).closest('tr').attr('id');
        setValAdvert(idAnnonce);

    });

    //Pour la désactivation d'une annonce
    $(".btnDesactivedAdvert").click(function(event){
        idAnnonce=$(this).closest('tr').attr('id');
        setInvalidAdvert(idAnnonce);

    });

    //Pour la description d'une annonce
    var ligne precedente=0;

    $(".adv").click(function(event){
        idA=$(this).closest('tr').attr('id');
        getDescription(idA);
        $("#da").show();

        var ligne_courante= $(this).parent().attr('id');

        if (ligne precedente==ligne_courante){
            $("#da").hide();
            $(this).parent()
            ligne precedente=0;

        }else{
            $("tr[id='"+ligne precedente+"']")
            ligne precedente=ligne_courante;

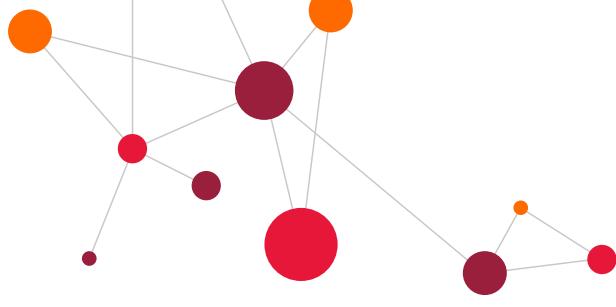
        }

    });

}

```

Figure 35 : Gestion d'une annonce



```
//VALIDATION D'UNE ANNONCE
function setValAdvert($id){

    if ($id > 0){
        $dbConn=Database::connect();
        $SQLQuery = 'UPDATE Annonce set id_Status=1 where id_Annonce=:id';

        echo $SQLQuery;
        try{
            $SQLStatement = $dbConn->prepare($SQLQuery);
            $SQLStatement->bindValue(':id', $id);

            if ($SQLStatement->execute()){

                }else{
                    print("Erreur d'exécution de la requête de validation !<br />");

                }
        }catch (Exception $ex){
            print("Erreur de préparation de la requête de validation !<br />");
            print($ex->getMessage());
        }
    }
}
```

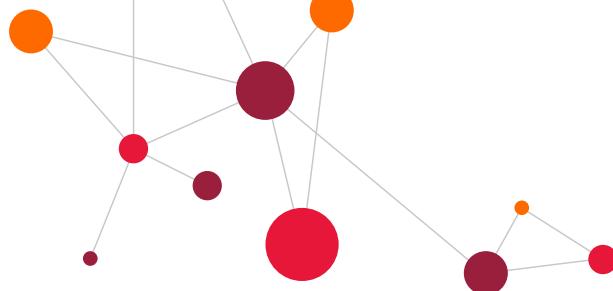
Figure 36 : exemple de la fonction de validation d'une annonce

## Conclusion du projet :

Cette application web a été le premier projet réalisé en groupe. Nous avons effectué la partie conception tous ensemble puis nous nous sommes partagés le travail d'écriture du code. Nous nous sommes confrontés à des difficultés de partage de code car Git fut une découverte pour nous quatre. C'est un outil puissant mais difficile à maîtriser. La gestion des conflits a été assez laborieuse.

L'organisation du code a aussi été problématique. En effet, nous n'avons pas assez structuré l'arborescence du code et nous n'avons pas utilisé le modèle MVC afin de séparer les aspects traitements/données/présentation et les interactions entre ces trois aspects.

Nous sommes tout de même arrivés à réaliser un produit fini et fonctionnel. Un aperçu du site web est disponible en annexe (**cf. annexe 16**). Il reste cependant de nombreuses améliorations et optimisations à implémenter.



#### 4.2.2

#### Le projet AndroidWorkshop

Sur ce projet, nous étions un groupe de trois étudiants.

#### Expression des besoins :

Le but de ce projet était de mettre en place un système de gestion du matériel réseau d'une entreprise en intégrant la possibilité de déclarer un incident sur une application web et la possibilité de traiter cet incident depuis une application mobile.

Sur ce projet, le script SQL de la base de données nous a été fourni.

#### Analyse :

Pour cela, nous avons dû :

- Développer une application Android.
- Développer une application Web.
- Faire interagir ces deux applications avec une API Web permettant une liaison avec une base de données.  
Il était imposé que cette API soit développée en Java EE.

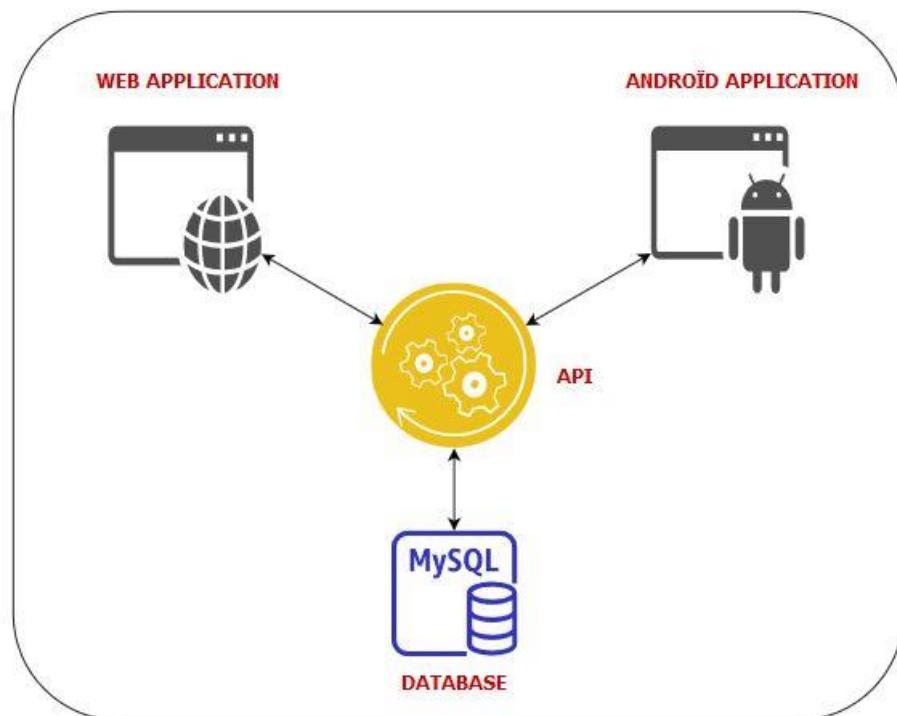
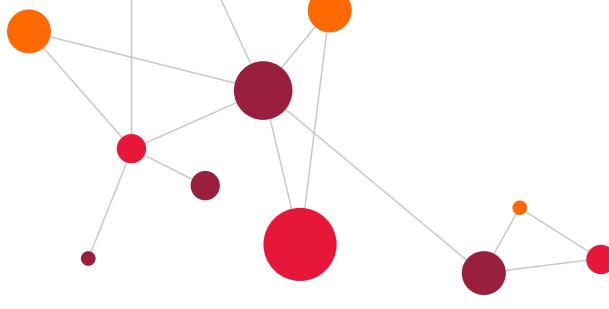


Figure 37 : Architecture globale du projet



La méthodologie que nous avons utilisée se rapproche d'une méthode agile. Afin de partager notre code, nous avons utilisé gitLab comme solution de partage de codes et de gestion de versions.

## Réalisation :

Nous avons débuté par le développement de l'application Android. Pour cela, nous avons commencé par la maquetter avec ses fonctionnalités essentielles.

Ce maquettage a également été réalisé à partir du logiciel Balsamiq Mockup.



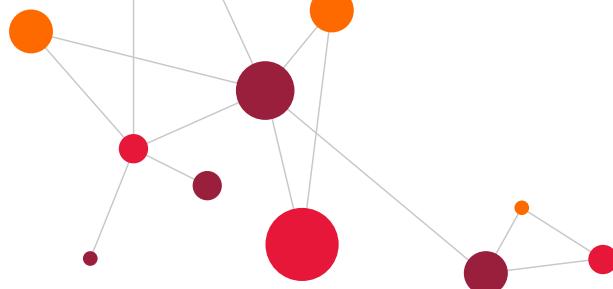
Figure 38 : Extrait du maquettage de l'application Androïd

Après le maquettage, nous sommes passés à la partie développement. Pour cela, nous avons utilisé l'IDE Android Studio.

Nous avons créé des « activity » correspondant à chaque écran de l'application à partir des maquettes réalisées précédemment.

Ensuite, nous avons développé la partie backend java. Afin de présenter le produit à la fin de notre premier sprint, nous sommes allés interroger une API créée par notre encadrant.

J'ai beaucoup aimé développer sur cet environnement particulier mêlant frontend et backend.



Le résultat de notre application est le suivant :

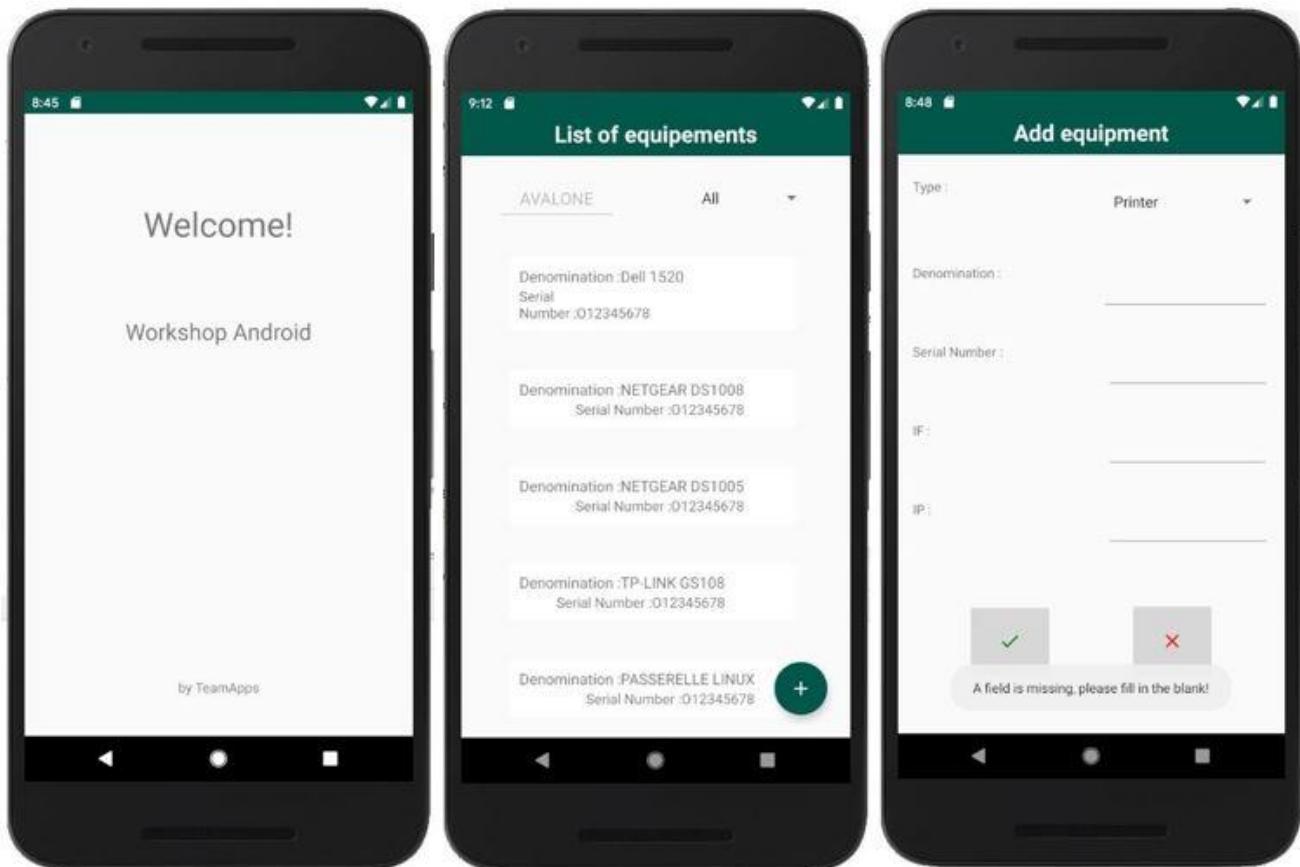
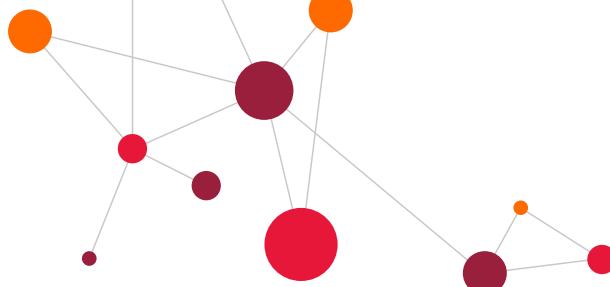


Figure 39 : Extrait de l'application Android



L'API a été mise en place au cours du second sprint.

Concernant celle-ci, nous avons opté pour l'architecture suivante :

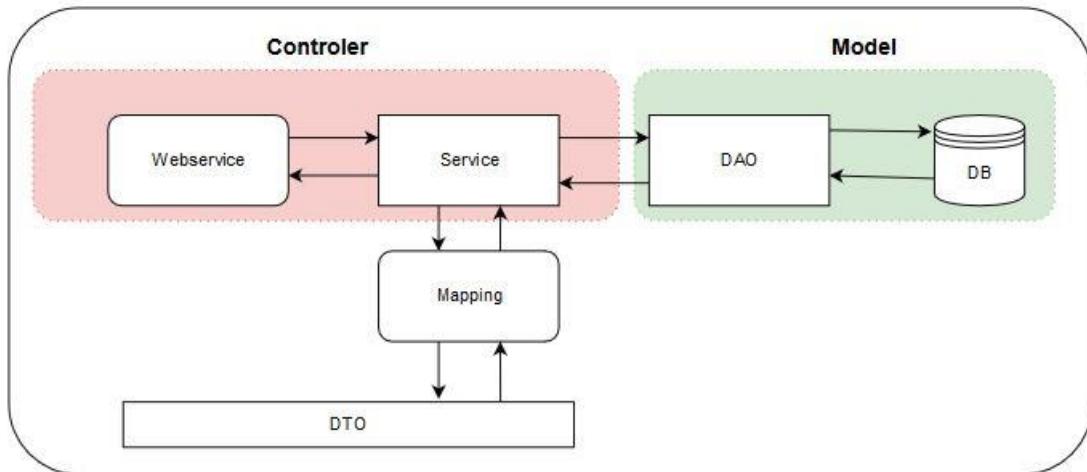


Figure 40 : schéma de l'architecture de l'API

Nous exposons au travers de l'API un point d'entrée appelé Webservice. Nous adressons à ce Webservice des requêtes utilisant le protocole HTTP : à l'aide de méthodes (GET/POST/PUT/DELETE etc...), nous réalisons des traitements divers.

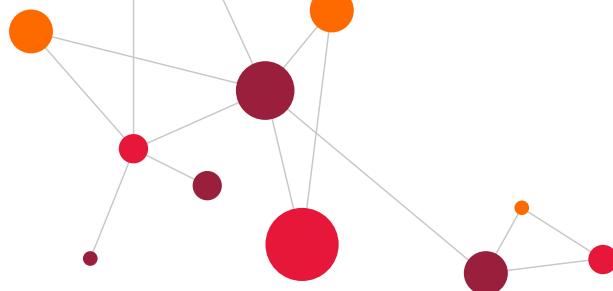
Une fois la requête adressée au webservice, celui-ci fait appel à l'interface « Service » qui contient les méthodes nécessaires au traitement de la requête. De là, nous pouvons décrire deux principaux cas de figure :

- Dans le cas où la requête engendre une modification sur la base de données, le corps de la requête HTTP contient un DTO (Data Transfer Object). Il s'agit de la représentation d'un objet au format JSON. Cet objet transite via la couche de Service afin d'y subir les différents traitements nécessaires (vérifications, transformations, agglomérations etc...). Ensuite, ce DTO est converti en DAO (Data Access Object) à l'aide de la classe de mapping associée. Ce DAO est renvoyé à l'interface Service qui le persiste ensuite en base de données au travers de l'interface DAO.
- Dans le cas où la requête n'engendre aucune modification en base, le webservice fait appel au Service qui interroge la base de données au travers du DAO. L'objet retourné (DAO) est alors converti en DTO avant de subir d'éventuels traitements et d'être renvoyé en réponse au format JSON.

Pour réaliser l'ensemble de cet API, nous avons utilisé :

- Le framework SpringBoot
- Le framework Hibernate qui reprend la norme JPA permettant ainsi de gérer la persistance des données en tables
- La librairie Lombok qui permet à partir d'annotations de générer par exemple les accesseurs (@Data) à la compilation.

Afin d'illustrer mes propos, prenons l'exemple, de l'exécution d'une requête afin d'obtenir les informations d'un client :



Je vais aller interroger l'API en passant par le webservice Client :

```

application.properties *ClientController.java ClientService.java
1 package com.udev2.workshop.androidWorkshopAPI.controller;
2
3 import java.util.ArrayList;
4
5 @RestController
6 @RequestMapping(value = "/client")
7 public class ClientController {
8
9     @Autowired
10    ClientService clientService;
11
12    @RequestMapping(method = RequestMethod.GET, value = "/{id}", produces = MediaType.APPLICATION_JSON_VALUE)
13    public @ResponseBody ClientDTO client(@PathVariable Long id) {
14
15        return clientService.findClientById(id);
16    }
17}

```

Figure 41 : Extrait de la classe ClientController.java

Dans cette classe, nous avons une méthode findClientById() appartenant à l'interface ClientService.java.

```

10
11 @Service
12 public interface ClientService {
13
14     /**
15      * Permet de récupérer un client par son ID
16      * @return Un client
17     */
18     ClientDTO findClientById(Long id);
19

```

Figure 42 : Extrait de l'interface ClientService.java

Cette méthode est elle-même implémentée dans la classe ClientServiceImpl.java qui surcharge cette méthode.

```

41
42     @Override
43     public ClientDTO findClientById(Long id) {
44         return clientMapping.toDto(clientDAO.findClientEntityById(id));
45     }
46

```

Figure 43 : Extrait de la classe ClientServiceImpl.java

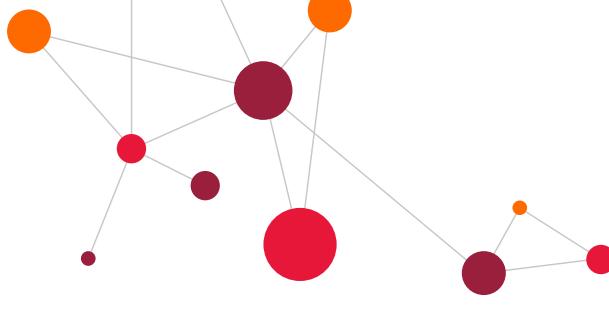
Cette méthode utilise l'interface clientDAO.java afin d'aller requêter sur la base.

```

10 @Repository
11 public interface ClientDAO extends JpaRepository<ClientEntity, Long> {
12
13
14     /**
15      * Permet de récupérer un client par son ID
16      * @return Client
17     */
18     ClientEntity findClientEntityById(Long id);
19

```

Figure 44 : extrait de l'interface clientDAO.java



Cette réponse arrive alors sous la forme d'un objet DAO. Il est converti en objet DTO via la classe ClientMapping.java avant d'être renvoyé au format Json.

```

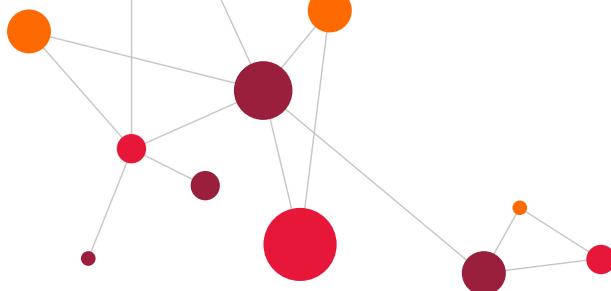
application.properties *ClientController.java *ClientService.java ClientServiceImpl.java ClientDTO.java ClientDAO.java ClientN
1 package com.udv2.workshop.androidWorkshopAPI.mapping;
2
3@import org.springframework.stereotype.Component;□
4
5 @Component
6 public class ClientMapping {
7
8     public ClientDTO toDto(ClientEntity entity) {
9         if (entity == null) {
10             return null;
11         }
12
13         VilleMapping villeMapping = new VilleMapping();
14         ClientDTO clientDTO = new ClientDTO();
15
16         clientDTO.setId(entity.getId());
17         clientDTO.setNom(entity.getNom());
18         clientDTO.setAdresse1(entity.getAdresse1());
19         clientDTO.setAdresse2(entity.getAdresse2());
20         clientDTO.setVille(villeMapping.toDTO(entity.getVille()));
21
22         return clientDTO;
23     }
24
25 }
26 }
```

Figure 45 : Extrait de la classe ClientMapping.java

Dans la base de données, la table client contient les données suivantes :

	123 id	ABC nom	ABC adresse1	ABC adresse2	123 idcpville
1	1	AVALONE	152 Avenue Jean-Jaurès	batiment B	1
2	2	CGI	6 rue des Comètes	batiment Andromède	10
3	3	EPSI	114 rue Lucien Faure	campus	3
4	4	CAP GEMINI	19 allée James Watt	batiment C	2
5	5	EDF	rue Centrale	entrée principale	9
6	6	ORANGE	33 route de Pauillac	batiment U	8
7	7	SFR	118 rue Sainte-Catherine	proche Macdo	3

Figure 46 : table client de la base de données NetGest



A la fin de notre sprint, lorsque l'on vient interroger notre API, nous obtenons par exemple pour le client 2 les informations suivantes au format JSON.

```

{
  "id": 2,
  "nom": "CGI",
  "adresse1": "6 rue des Comètes",
  "adresse2": "batiment Andromède",
  "ville": {
    "id": 10,
    "codePostal": "33187",
    "ville": "LE HAILLAN"
  }
}

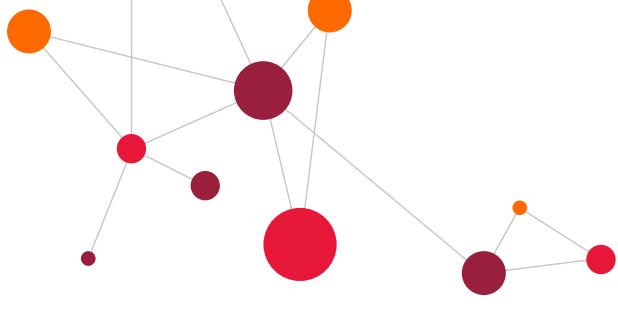
```

Figure 47 : Résultat Json d'un appel sur l'API pour le webservice Client

Pour finir, l'application web a été réalisée en Node.js. Cette application est pour le moment très minimalist mais nous avons réussi à la faire communiquer avec notre API.

id	nom	adresse1	adresse2	idcpville	materiel
1	AVALONE	152 Avenue Jean-Jaurès	batiment B		<a href="#">liste materiel</a>
2	CGI	6 rue des Comètes	batiment Andromède		<a href="#">liste materiel</a>
3	EPSI	114 rue Lucien Faure	campus		<a href="#">liste materiel</a>
4					

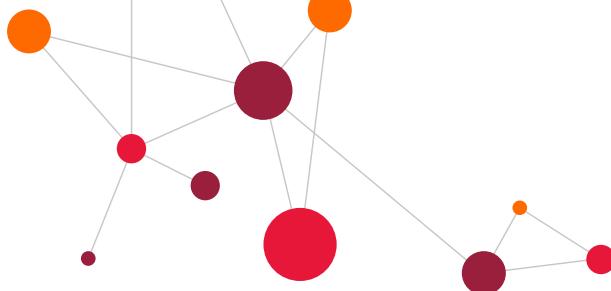
Figure 48 : Extrait de l'application Web



### Conclusion du projet :

Il y a encore de nombreuses évolutions à apporter à l'ensemble de ce projet mais ces premiers sprints nous ont permis de mettre en application de nombreuses compétences apprises au cours de cette année et de présenter les fonctionnalités principales souhaitées.

Faire communiquer plusieurs applications entre elles et échanger des données m'a particulièrement intéressé.



## 5 Conclusion générale

Cette année d'alternance m'a permis de monter en compétences sur différents aspects :

- Méthodologie de travail
- Technologie Mainframe z/OS et son écosystème
- Programmation procédurale (Cobol)
- Programmation objet (PHP, Java)
- Utilisation de base de données relationnelles (DB2 et MySQL)

En entreprise, malgré un environnement de travail très différent des technologies que nous avons étudié à l'école, j'ai tout de même pu établir de nombreuses analogies.

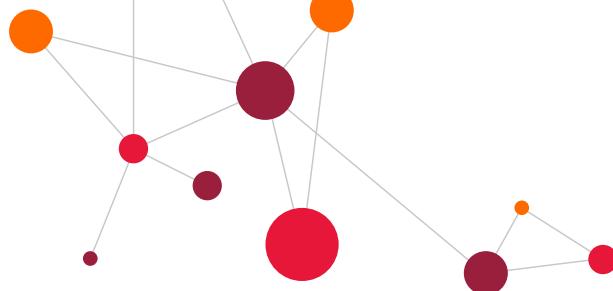
Mon intégration au sein de CGI et du centre de service s'est très bien déroulée. Mes objectifs sont clairement définis et je suis épanoui dans mon travail au quotidien.

Concernant mon niveau de maturité : mon autonomie progresse, je suis maintenant capable de développer un programme à partir de spécifications techniques moins détaillées.

Une des difficultés rencontrées fut l'évaluation de mon Reste à Faire mais avec l'expérience, j'arrive à être de plus en plus précis sur mes estimations.

Je souhaite continuer sur cette orientation, y compris technologique sans pour autant délaisser mon intérêt pour les nouvelles technologies afin de me laisser la possibilité d'évoluer.





## Lexique

API : Application Programming Interface

ARMIDE : Base de données spécifiques contenant des tables de paramétrages

BDD : Base De Données

CDC : Cahier Des Charges

COBOL : COmmon Business Oriented Language

DAO : Data Access Object

DCG : Dossier de Conception Général

DCOD : Dossier de Conception Détailé

DSN : Data Set Name

DTO : Data Transfer Object

ESN : Entreprise spécialisée en services numériques

IDE : Environnement de Développement intégré ou Integrated Development Environment

IHM : Interface Homme/Machine

MSP : Macro-Structure Paramétrée

MOA : Maitrise d'ouvrage : personne ou groupe qui exprime le besoin

MOE : Maitrise d'œuvre : personne ou groupe qui assure la production du projet

MVC : Model Vue Controller (patron de conception)

MVS : Multiple Virtual Storage

NTI : Nouvelles Technologies de l'Information

PDS : Partitioned Data Set (fichier comprenant plusieurs membres)

RDz : Rational Developer for Z

RMOA : Responsable Maitrise d'Ouvrage

SA : Service Applicatifs

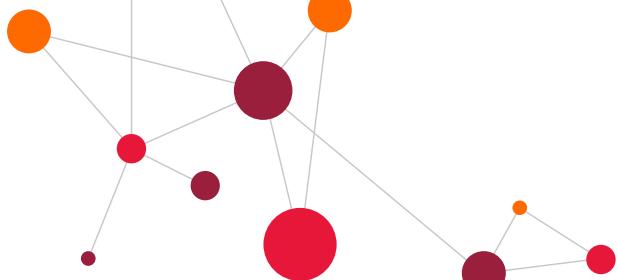
SGBD : Système de Gestion de Base de Données

TA : Test d'Assemblage

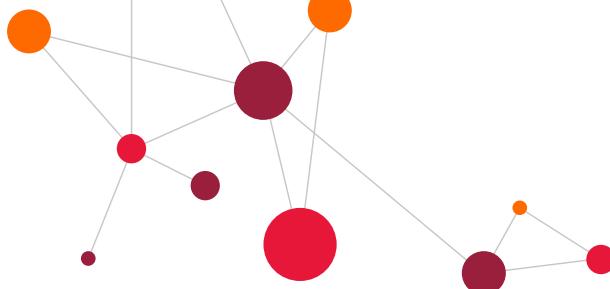
TU : Test Unitaire

UDC : Usine de Développement Cobol

z/Os : Système d'exploitation d'un mainframe IBM



## ANNEXES



## Annexe n°1 – Fiche d'évaluation

<b>CERTIFICATION PROFESSIONNELLE EVALUATION ENTREPRISE</b>	
<u>STAGIAIRE</u>	<u>ENTREPRISE</u>
Nom : HAMON	Nom de la société : CGI
Prénom : Glenn	Tuteur(trice) : Sébastien MONGET-SARRAIL
Certification visée :	Courriel : sebastien.monget-sarrail@cgi.com

«tuteur\_Civilite»,  
Afin de parfaire la validation de la certification professionnelle de votre stagiaire, nous vous remercions de remplir exhaustivement cet **original** et nous le retourner dans *les meilleurs délais*. Ce document sera examiné par le jury en vue de l'attribution de la certification professionnelle de votre stagiaire.

L'intégration dans l'équipe de travail est-elle satisfaisante ?  Oui /  Non

Le comportement en situation professionnelle est-il adapté à la culture de l'entreprise ?  Oui /  Non

Le comportement relationnel est-il adapté au métier visé ?  Oui /  Non

La capacité de travail fournie correspond-elle au niveau d'un professionnel du métier ?  Oui /  Non

Le niveau de responsabilités attendu est-il satisfait ?  Oui /  Non

Des questions sont-elles posées à bon escient ?  Oui /  Non

Des difficultés en relation avec le niveau de responsabilités sont-elles surmontées ?  Oui /  Non

Le niveau d'obligation de réserve demandé est-il pris en compte ?  Oui /  Non

Les activités effectuées concourent-elles à la couverture du champ de compétences ?  Oui /  Non

Le métier associé à la certification visée est-il compris ?  Oui /  Non

Le stagiaire est-il apte à occuper la fonction visée même comme débutant ?  Oui /  Non

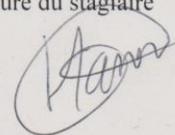
Appréciation générale suite au stage en entreprise :

Glenn est appliqué et parfaitement intégré au sein du projet  
Le travail qu'il fournit est toujours de qualité et ne nécessite pas d'être modifié. Il aborde les tâches qui lui sont confiées avec un œil critique appréciable.

Fait à Haillan, le 06/11/10

**CGI**

**CGI Signature du tuteur**  
Immeuble Andromède  
6, rue des Comètes - CS 10026  
33187 Le HAILLAN Cedex  
Tél : +33 5 57 70 57 00  
SAS au capital de 127 913 933 €  
RCS Nanterre B 342 042 755

Signature du stagiaire 

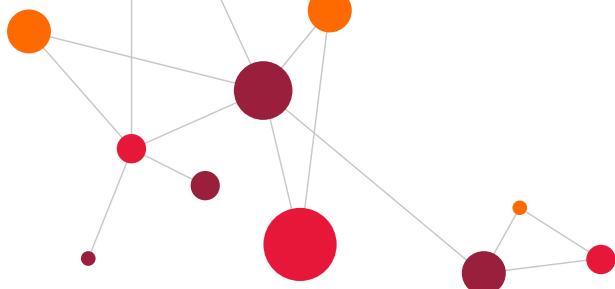
Association ADIP régie par la loi du 1<sup>er</sup> juillet 1901  
Siret : 409 801 677 00017 Code APE : 85.59 A  
N° organisme : 11 75 27 97 775

Centre de formation  
44 bis, quai de Jemmapes 75010 Paris  
Siège social  
12, rue Alexandre Parodi 75010 Paris

Tél : 01 80 97 35 00  
[mlahhit@groupe-igs.fr](mailto:mlahhit@groupe-igs.fr)  
[www.ipi-ecoles.com](http://www.ipi-ecoles.com)

**GROUPE IGS**

© Institut de Poly-informatique (2011)



## Annexe n°2 – Curriculum Vitae

### Glenn HAMON

19 rue du professeur Bernard  
33170 Gradignan  
06 30 57 77 17  
glenn.hamon@orange.fr

31 ans

#### EXPÉRIENCES PROFESSIONNELLES

##### ▪ CGI (alternance)

Le Haillan (33) [DECEMBRE 2018 – NOVEMBRE 2019]  
✓ Développement Mainframe

##### ▪ Institut Technique des gaz et de l'air (ITGA)

Mérignac (33) [OCTOBRE 2017-AVRIL 2018]

- ✓ Prélèvements atmosphériques
- ✓ Numérisation 3D de bâtiments

##### ▪ Institut des Sciences Analytiques (CNRS) - CRMN

Villeurbanne (69) [MAI -NOVEMBRE 2016]

- ✓ Maintenance du parc de spectromètres de Résonance Magnétique Nucléaire (RMN)
- ✓ Assistance technique pour les utilisateurs

##### ▪ Laboratoire SERVIER

Croissy-sur-Seine (78) [DECEMBRE 2014 – SEPTEMBRE 2015]

- ✓ Détermination structurale de molécules issues de la Recherche Servier par RMN et Spectroscopie Infrarouge (IR) et Chromatographie liquide (LC/MS)

##### ▪ Laboratoire SERVIER (stage)

Suresnes (92) [MARS-AOÛT 2014]

- ✓ Etudes et implémentations des nouvelles techniques d'acquisition rapide en RMN multidimensionnelle

#### FORMATIONS

- 2018-2019 : **Formation U'DEV** – Certification IPI -Concepteur Développeur d'Applications Numériques [EN COURS] – Bordeaux (33)
- 2013- 2014 : **Master 2 PROFESSIONNEL Méthodes d'Analyse** – Université Rennes I – Beaulieu (35)
- 2012-2013 : **Master 1 Chimie Analytique** – Université Toulouse III - Paul Sabatier (31)
- 2009-2012 : **Licence 3 Chimie** – Université Bordeaux I (33)
- 2007-2009 : **1<sup>ère</sup> année faculté de Pharmacie** – Université Bordeaux II – Victor Segalen (33)
- 2006-2007 : **Baccalauréat Série Scientifique** – Lycée Pape Clément Pessac (33)

#### LANGUES

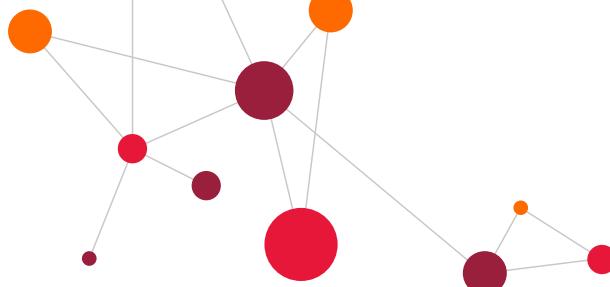
- Anglais : intermédiaire
- Espagnol : élémentaire

#### LOISIRS

- Guitare

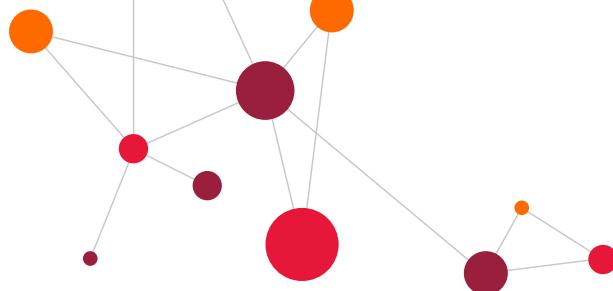
### Annexe n°3 - Tableau des compétences : Qualité et sécurisation du code réalisé

Compétences ou capacités évaluées	Critères d'évaluation	Exemples d'activités et tâches	Activité pratiquées	Origine de l'acquisition	Preuves apportées & ref.annexe
Formaliser, identifier les résultats attendus.	La liste de contrôle des attendus fonctionnels est paraphée.	Etude de l'existant. Rédaction du cahier des spécifications fonctionnelles.	Avant chaque réalisation ou modification de programme à l'aide de la documentation (CDC, DCG, DCOD), je rédige mes cas de tests unitaires. Les TU sont correctement paraphés avec mon trigramme.	E (32 semaines)	Fiche TU RISCL067 (Annexe 12)
Respecter des contraintes,	Un plan d'assurance qualité est observé.	L'application est organisée en couches indépendantes. Les règles métier sont encapsulées dans des services logiciels. L'accès aux données est réalisé par des services logiciels indépendants du mode de stockage.  L'exécution de l'application est répartie entre un nombre d'ordinateurs adapté au contexte. Un formulaire d'estimation des risques est rempli. Une norme de présentation des données est respectée. Les interfaces Homme/Machine sont validées.,	Réalisation d'application N-tiers  Conception /Architecture d'applications logicielles. Conception de services métiers. Conception de services d'accès aux données.  Détermination du nombre de tiers de l'application. Anticiper les évolutions. Qualifier les risques	E (32 semaines)  Prise en compte du CDC du client à respecter et des consignes du DCOD.	Règles de gestion dans un CDC-DG
	Respecter les recommandations qualité de la norme en vigueur pour l'architecture des logiciels.			Schéma de l'architecture de l'API - AndroidWorkshop (Figure 40 : schéma de l'architecture de l'API)	
	Respecter une norme de présentation des écrans et documents de sortie.			Capture d'écran de la note PQC (Figure 17 : extrait de l'outil cora)	
	Concevoir des programmes avec une orientation objets.	Une programmation orientée objets est utilisée.	Conception de logiciels  Programmation de logiciels aux données de l'entreprise.	F(10 semaines)	Projet Dev'Hire, Android Workshop
				F(5 semaines)	Schéma de l'architecture de l'API - AndroidWorkshop (Figure 40 : schéma de l'architecture de l'API)
	Garantir un accès sécurisé aux données			E (12 semaines)	Fiche TU RISCL067 (Annexe 12)
	Livrer le logiciel déverminé.			E (32 semaines)	
	Livrer le logiciel conforme aux attentes.				Attribution d'une note de contrôle du code.
	Cloûter une mission				Absence de retours provenant des équipes de recettage (IMOE/RMOA) du client
					Le PV de réception du logiciel est validé.
					Mise en exploitation



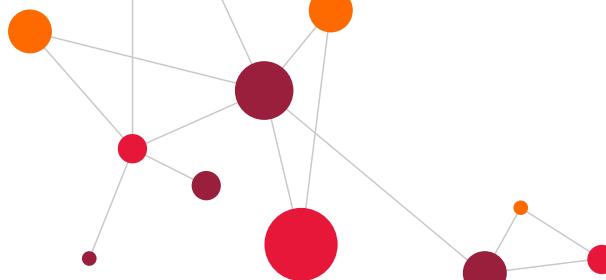
## Annexe n°4 - Tableau des compétences : Audit, conception, méthode de projet

Compétences ou capacités évaluées	Critères d'évaluation	Exemples d'activités et tâches	Activité pratiquées	Origine de l'acquisition	Preuves apportées & ref.annexe
Formaliser des processus	La procédure du service utilisateur est formalisée et validée. La procédure du service utilisateur est conforme aux règles du système de management des services de l'entreprise. La circulation du document résultat du traitement prévu est matérialisée dans un diagramme de workflow. La proposition de la reconstruction de la procédure est validée. La base de donnée est modélisée.	Etude de l'existant. Identification des procédures en place. Contrôle de la conformité des procédures utilisées avec la gouvernance de l'entreprise.			
Formaliser les règles de gestion et d'organisation des données de l'entreprise		Recensement des documents utilisés, identification de leur circulation et des acteurs concernés. Reconfiguration de procédure. Conception d'une base de données.	Réalisation de MCD. Création d'un script SQL de génération de base de données	F (3 semaines)	Base de données Dev'Hire (Annexe 13 et 14)
Une méthode de conception par objets est utilisée.	Concevoir des éléments logiciels réutilisables.	Conception de l'architecture applicative.	Conception de webservice (POO)	F (5 semaines)	Projet Android Workshop Schéma de l'architecture de l'API (Figure 40 : schéma de l'architecture de l'API)
Une méthode AGILE est utilisée. Absence de signaux d'alertes au point de contrôle du projet.	Produire du logiciel en équipe. Remonter les alertes au(x) décideur(s).	Programmation en équipe. Ecriture de code. Coordination de l'avancement.	Pas d'utilisation de méthode agile en entreprise mais apprentissage de la méthode SCRUM en formation.	F (1 semaine)	Schéma du rôle du Scrum Master (Annexe 17)
Les étapes du projet sont planifiées.	Estimer des délais.	Planification des tâches du projet.	Utilisation d'outils de gestion de ressource (Planning, MyPM permettant de renseigner son avancement et son Reste à Faire)	E (32 semaines)	Planning du projet (Figure 7 : Extrait du planning LAF) MyPM (Figure 8 : Extrait de l'outil de gestion MyPM)
					Application Web Dev'Hire



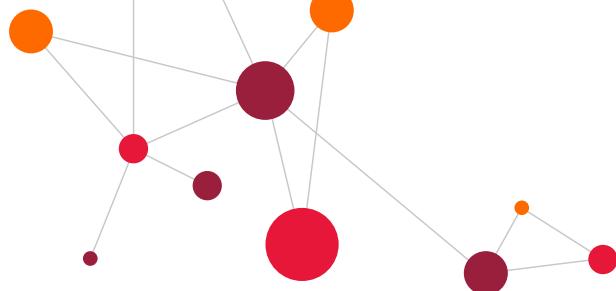
## Annexe n°5 - Tableau des compétences : Réalisation d'Applications logicielles

Compétences ou capacités évaluées	Critères d'évaluation	Exemples d'activités et tâches	Activité pratiquées	Origine de l'acquisition	Preuves apportées & ref.annexe
Encapsuler des solutions logicielles spécifiques dans des services logiciels génériques.	Le service d'accès aux données est opérationnel.	Programmation. Investigations documentaires fonctionnelles ou techniques complémentaires. Transcription des spécifications fonctionnelles en algorithmes.	Dans le cadre du développement mainframe, j'ai réalisé des services applicatifs métiers comprenant des composants NK, composants génériques permettant un accès à une ou plusieurs bases de données et pouvant être ainsi utilisés par différentes applications.	E (6 semaines)	Ensemble des modules du SAM RISCLI067 (Figure 25 : schéma général du RISCLI067)
Produire du logiciel générique réutilisable.	Des services logiciels internes sont réutilisables.	Transcription des algorithmes en code source. Compilation du code source. Agglomération des différents éléments du logiciels en unités de traitement, réalisation des test unitaires.	Lorsque je développe un programme, je réalise un script d'exécution (JCL) que je fournis également à la livraison.	E (20 semaines)	JCL du programme LDC131 (Annexe 11)
Produire du logiciel partageable.	Des services logiciels sont partageables en local. Des services logiciels sont partageables à distance.	Le logiciel est livrable, prêt pour la mise en production.	Il m'arrive régulièrement d'intervenir sur des programmes afin de réaliser des modifications. Afin de m'assurer que la modification du code n'a pas eu d'impact, autre les tests concernant la modification, je réalise des tests de non régression.	E (4 semaines)	
Intégrer des éléments logiciels hétérogènes et produire des exécutables livrables.		La modification n'entraîne pas de régression fonctionnelle.	Chaque jour, à l'aide d'ordre de travail (OT), je renseigne mon activité, sur un outil de gestion de ressource (MyPm) et j'indique mon RAF.	E (32 semaines)	MyPM (Figure 8 : Extrait de l'outil de gestion MyPM)
Modifier un algorithme sans générer de dysfonctionnements.		Le compte-rendu d'activité est renseigné, les écarts sont constatés.	Mise à jour du planning de réalisation.		
Contrôler des délais.					



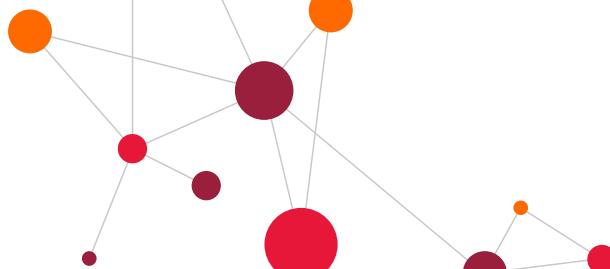
## Annexe n°6 - Tableau des compétences : Communiquer avec les acteurs du projet

Compétences ou capacités évaluées	Critères d'évaluation	Exemples d'activités et tâches	Activité pratiquées	Origine de l'acquisition	Preuves apportées & ref.annexe
User d'une communication professionnelle tant en français qu'en anglais.	Le compte-rendu de la réunion est validé. Le score du TOEIC est >749.	Elaboration et rédaction de documents techniques, commerciaux ou internes à destination, des utilisateurs, des clients ou des collaborateurs,...	Au cours d'une de mes expériences professionnelles dans un laboratoire de recherche CNRS, j'ai travaillé avec des chercheurs et doctorants internationaux, le langage de communication principal était l'anglais.  Mon score lors du TOEIC d'entraînement a été de 795/990.	E (6 mois)	Tableau Daily meeting (Figure 4 : management visuel tableau du daily meeting)
Interagir efficacement dans un environnement de travail collaboratif.	Le document collectant l'expression des besoins des utilisateurs est validé.  L'aide du logiciel est rédigée. La documentation du livrable est diffusée.  La présentation est appréciée.  Les utilisateurs sont opérationnels, le transfert des nouvelles compétences est validé.	Rédaction des spécifications fonctionnelles de la solution informatique. Ecriture des interfaces homme/machine. Relations avec les clients. Animation de réunions de travail et interviews d'utilisateurs. Démonstrations, recettage de livrables.  Formation des utilisateurs au logiciel.	Au quotidien, je travaille en équipe, j'entretiens de bonnes relations avec mes collègues et n'hésite pas à interroger ces derniers quand je rencontre un point de blocage, principalement au cours du daily meeting. Surtout lorsque celui-ci est fonctionnel (demande à l'analyste). Réalisation d'IHM Réalisation d'IHM	E (32 semaines)	Application Web Dev'hire (Annexe 16)



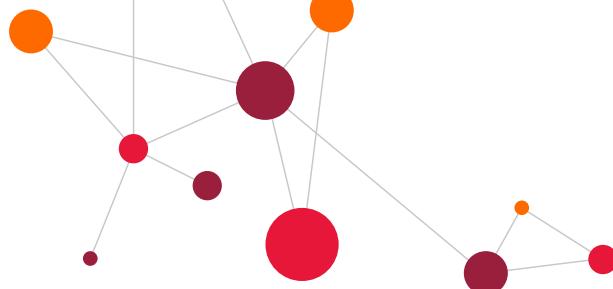
Annexe n°7 – Tableau des compétences : Adapter l'environnement d'exécution, échanger des données entre logiciels

Compétences ou capacités évaluées	Critères d'évaluation	Exemples d'activités et tâches	Activité pratiquées	Origine de l'acquisition	Preuves apportées & ref.annexe
Réaliser des échanges de données informatisés (EDI).	Automatiser des traitements.	Réalisation d'un procédé d'échange de données informatisées. Rétro-documentation de logiciels et de bases de données. Consolidation, agrégation de données. Programmation de l'interface d'échange de données.	Réalisation d'une API communicant au format JSON avec une application Web et une application Mobile.  Installation d'un serveur Apache TomCat sur mon ordinateur.	F (4 semaines)	Android Workshop



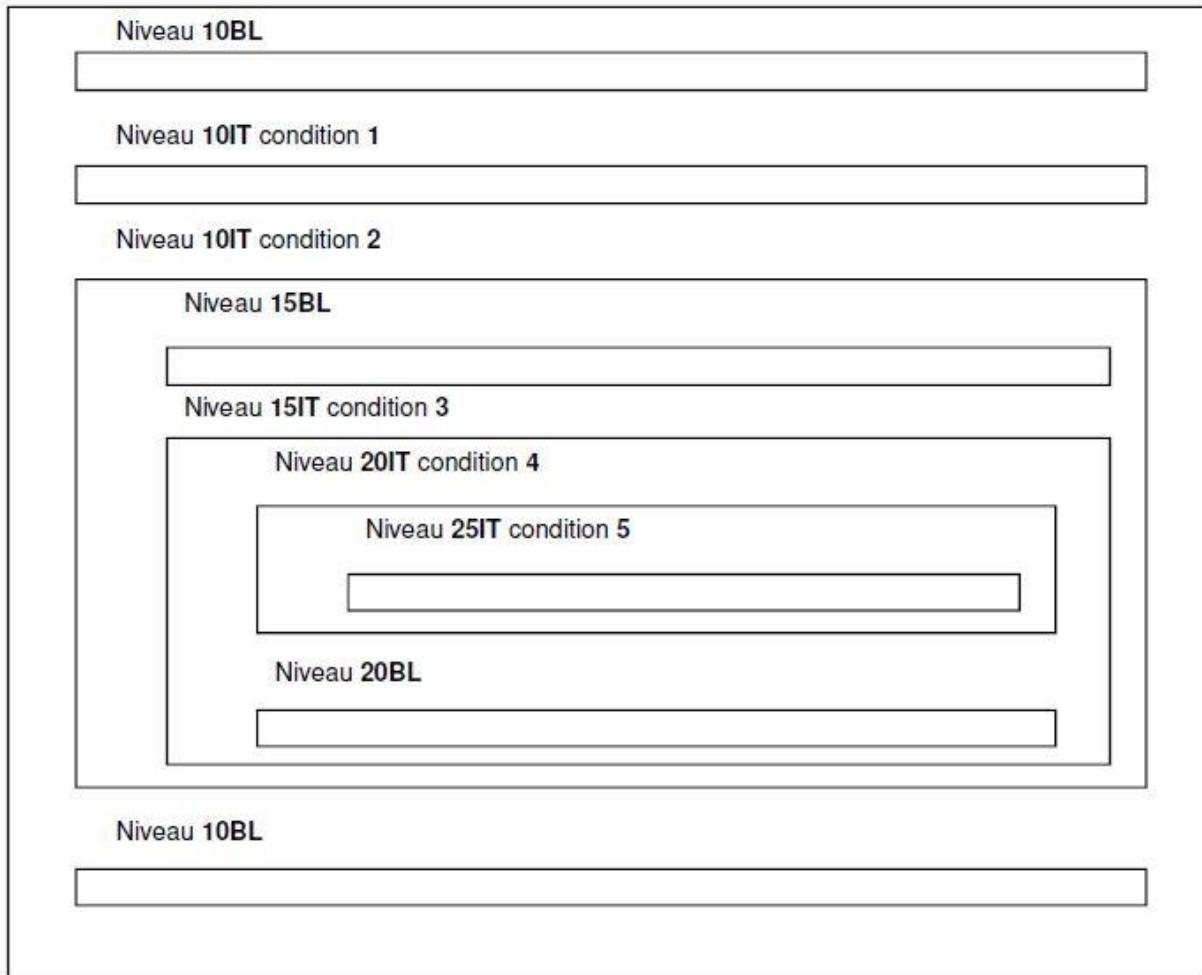
## Annexe n°8 – La Boucle Pacbase

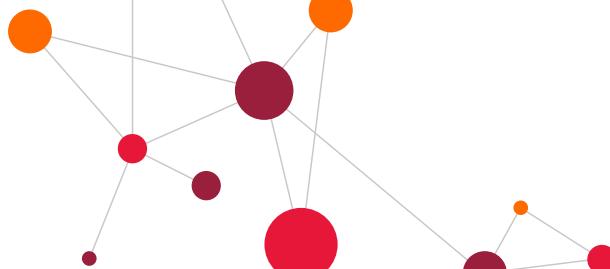
F01	Initialisation PAC
F01xx	Ouverture du fichier xx déclaré en -CD
F02 à F04	Initialisation personnelle
F05	Lecture des fichiers sans ruptures
F05xx	lecture du fichier xx et positionnement des indicateurs de fin itération et fin de fichier xx (xx-FT et xx-FI)
F06 à F09	Insertions spéciales
F10	Lecture des fichiers avec ruptures
F10xx	Gestions des zones de ruptures du fichier xx Positionne les indicateurs de fin Bascule les zones de lectures en zone de travail ( xx00 → 1-xx00)
F11 à F19	Insertions spéciales
F20	Fin de traitement (si FT= all 1 avec FT = somme des xx-FT)
F20xx	Fermeture du fichier xx
F2099	Stop RUN
F22	Calcul des ruptures
F22xx	Calcul des ruptures du fichier xx
F24	Calcul des configurations (synchronisations)
F24xx	Config du fichier xx (zone xx-CFn)
F26	Calcul des ruptures totales Calcul de RTPn RTDn NRP NRD
F30 à F79	Insertion du corps du programme ATTENTION ceci est vrai dans un cas général, dans le cadre d'un programme de contrôle et MAJ automatique, les fonctions 30/33/36/39/42/45/73/76 sont utilisées <b>On s'insère en général entre F50 et F70</b>
F8E	Edition de l'état E ( si condition mise en -D de l'état) 8EZz Libellés 8E00 Structures 8E99 Ecritures physiques
F90	Ecriture des fichiers
F90xx	Ecriture du fichier xx
F9099-iter-fin	GOTO F05 (retour lecture suivante)
F91 à F99	Insertion personnelle à appeler par PERFORM



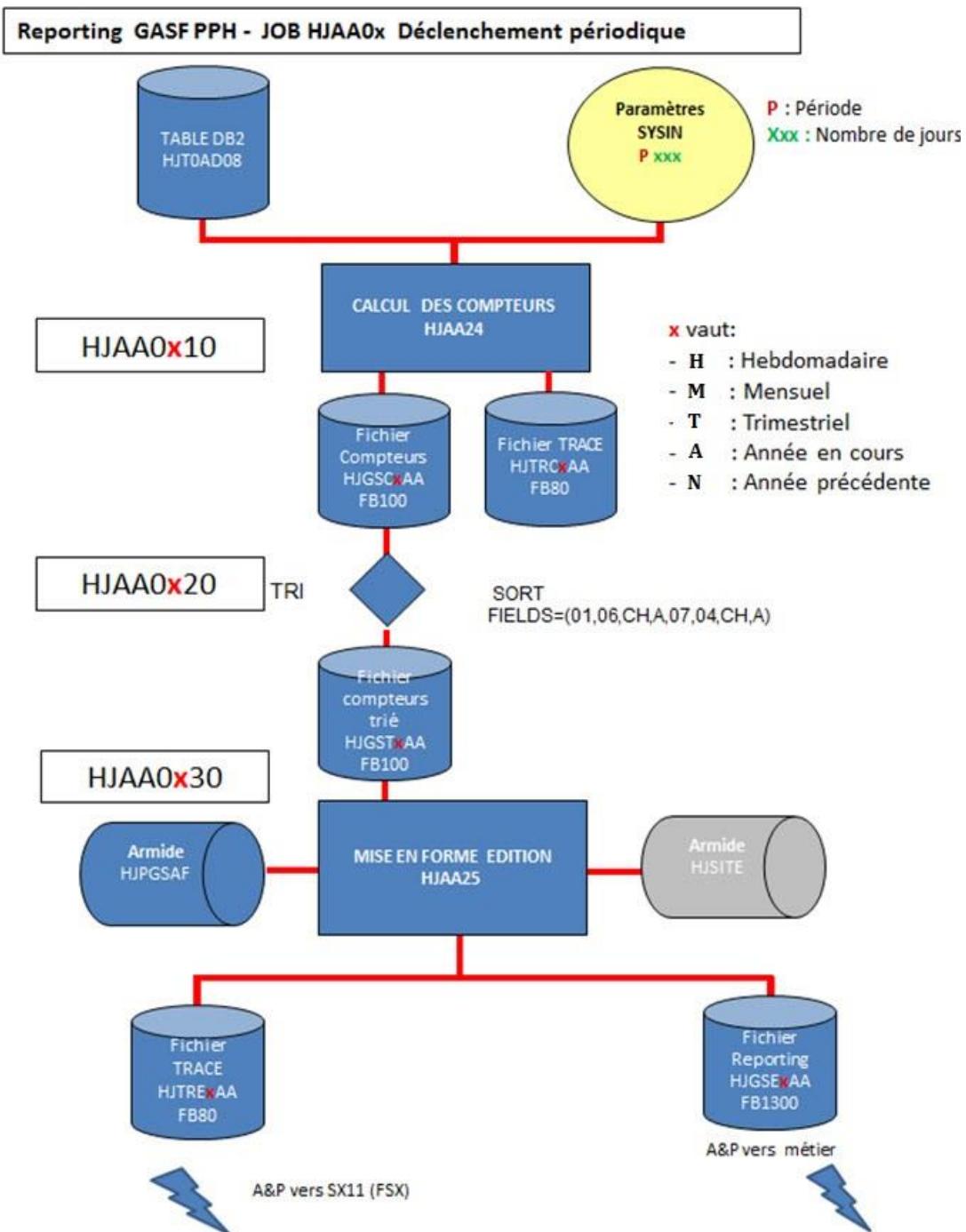
## Annexe n°9 – Niveau des fonctions PACBASE

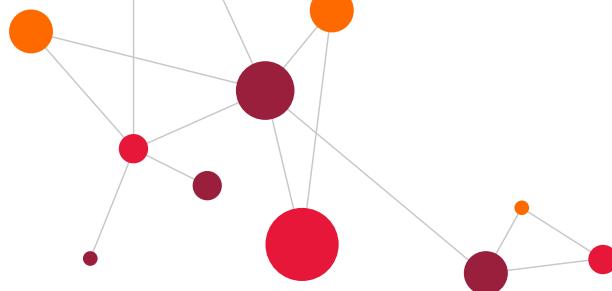
### Niveau 05BL





## Annexe n°10 – Schéma du traitement Reporting GSAF PPH



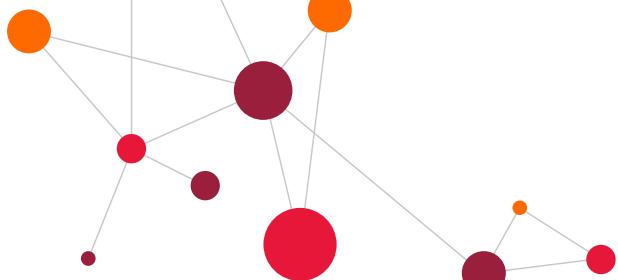


## Annexe n°11 – JCL du programme LDC131 (1/2)

```

//LDC10CTU JOB (LDXX,E,_500), 'JCLTU', CLASS=C,MSGCLASS=K,
// NOTIFY=&SYSUID,COND=(8,LE)
//*****
//** EXTRACTION DES FGR RTG VERS MDG
//*****
//*
//JOBLIB DD DSN=RSX.NEXX.NEULMD80,DISP=SHR
// DD DSN=RSX.NEXX.NEALMD80,DISP=SHR
// DD DSN=RSX.NEXX.NEMLMD80,DISP=SHR
// DD DSN=RSX.NEXX.NEVLMD80,DISP=SHR
// DD DSN=RSX.NEXX.NEPLMD00,DISP=SHR
// DD DSN=RSX.NEXX.NETLMD00,DISP=SHR
// DD DSN=RSX.NEXX.NERLMD00,DISP=SHR
// DD DSN=DB2.DB2VB10.SDSNLOAD,DISP=SHR
// DD DSN=SX1.W9E0.DB2T.VB10.RUNLIB.LOAD,DISP=SHR
//*
//*
//*****
//DEBJCL EXEC PGM=IDCAMS
//*****
//**COM-DESC=STEP DE DELETE AUTOMATIQUE EN DEBUT DE JOB
//*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DELETE PUTUSR.$PERM.CR0573F.LDC131.TRACE.CASXX
DELETE PUTUSR.$PERM.CR0573F.LDC131.SA.CASXX
DELETE PUTUSR.$PERM.CR0573F.LDC131.CRE.CASXX
DELETE PUTUSR.$PERM.CR0573F.LDC131.CSV.CASXX
SET MAXCC = 0
/*
//*****
//LDC10010 EXEC PGM=IKJEFT01,
// DYNAMNBR=20
//*****
//**COM-DESC=EXTRACTION DES FGR RTG VERS MDG
//*****
//*
//IAIAI DD DSN=D0TJX2.$TPXX.UV5XIAAA,DISP=SHR
//IAAII DD DSN=D0TJX2.$TPXX.UV5TIAAA,DISP=SHR
//IBIBI DD DSN=D0TJX2.$TPXX.UV5XIBAA,DISP=SHR
//IBBII DD DSN=D0TJX2.$TPXX.UV5TIBAA,DISP=SHR
//ICICI DD DSN=D0TJX2.$TPXX.UV5XICAA,DISP=SHR
//IFIFI DD DSN=D0TJX2.$TPXX.UV5XIFAA,DISP=SHR
//IHIHI DD DSN=D0TJX2.$TPXX.UV5XIHAA,DISP=SHR
/*
//SYSTSIN DD *
DSN SYSTEM(DB2T)
RUN PROGRAM(LDC131) PLAN(LDPBD0AA)
END
/*
//*

```

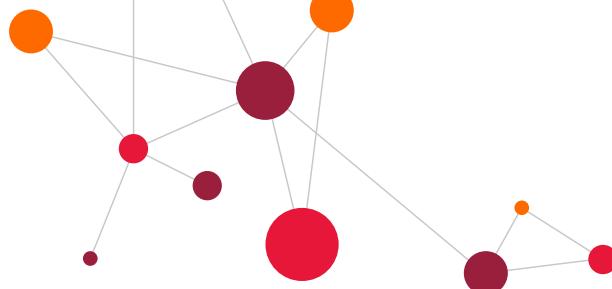


## Annexe n°11 – JCL du programme LDC131 (2/2)

```

//EA      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.PARAM.CASXX,
//          DISP=SHR
//DA      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.FIJOUR.CASXX,
//          DISP=SHR
/* Fichier(s) pris en Sortie
//TR      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.TRACE.CASXX,
//          DISP=(,CATLG,DELETE),
//          RECFM=FB,
//          LRECL=200,
//          SPACE=(TRK,(20,20),RLSE),
//          UNIT=3390
//SA      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.SA.CASXX,
//          DISP=(,CATLG,DELETE),
//          RECFM=FB,
//          LRECL=80,
//          SPACE=(TRK,(15,15),RLSE),
//          UNIT=3390
//SB      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.CRE.CASXX,
//          DISP=(,CATLG,DELETE),
//          RECFM=FB,
//          LRECL=300,
//          SPACE=(TRK,(400,160),RLSE),
//          UNIT=3390
//SC      DD DSN=PUTUSR.$PERM.CR0573F.LDC131.CSV.CASXX,
//          DISP=(,CATLG,DELETE),
//          RECFM=FB,
//          LRECL=700,
//          SPACE=(TRK,(400,160),RLSE),
//          UNIT=3390
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSOUA   DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*

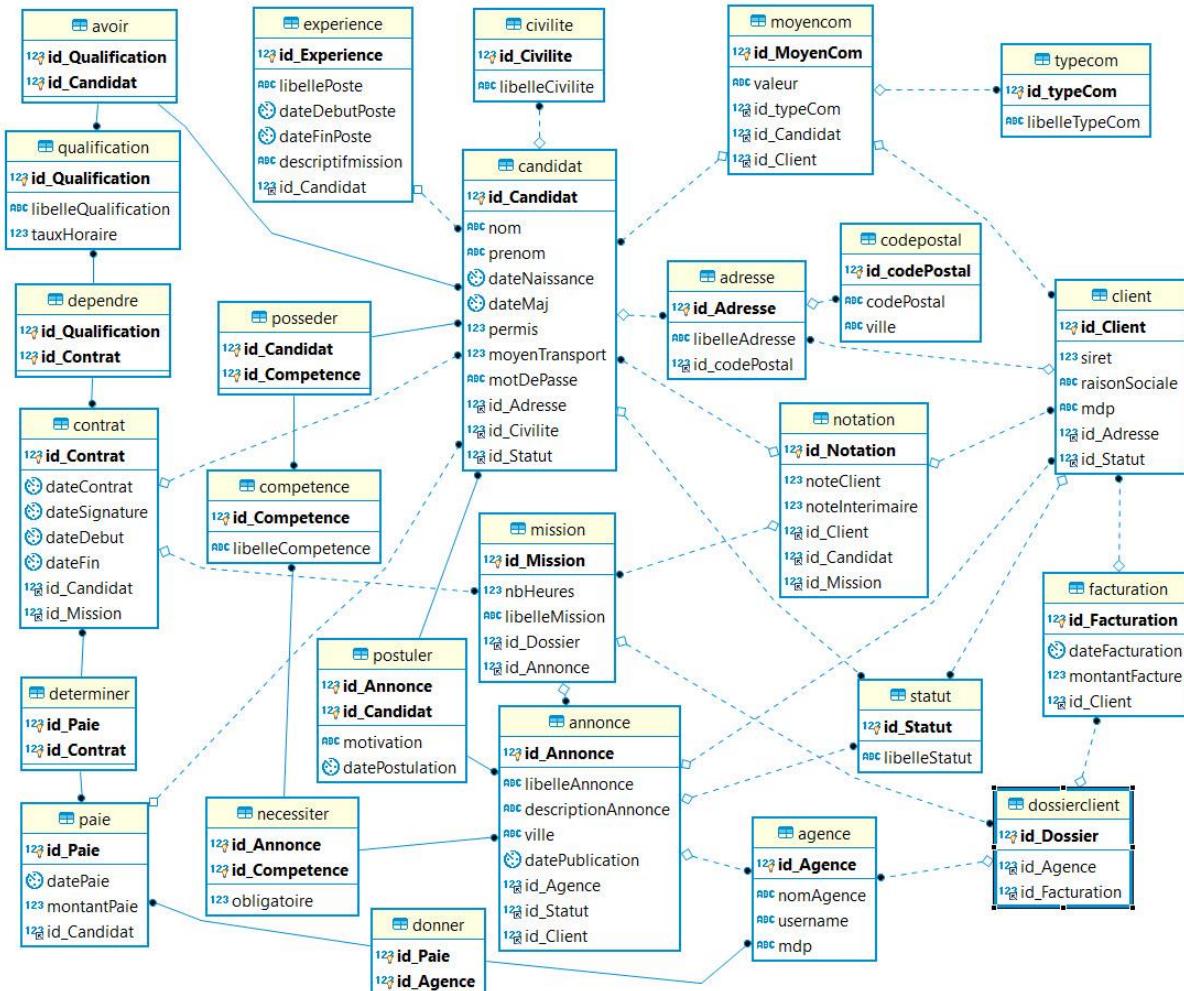
```

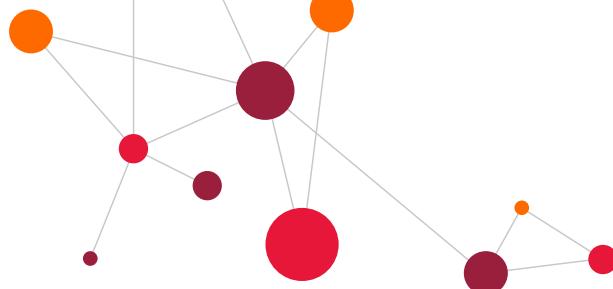


## Annexe n°12 – Fiche de test unitaire – RISCLI067

		Tableau de préparation et de suivi des tests							
<b>Projet</b>									
Libellé de l'évolution   RSK - ARP - MAJ FGR pour un client RTG ND N° MDSPEC   2019-09-11-0013									
<b>Scénario de test</b>									
Date des tests	06/11/2019	Date du jour							
Scénario de test rédigé par	GHAM								
Scénario de test validé par	RBO								
Outils de test utilisé	SALEPH								
Type de test	TU								
<b>Programme testé</b>									
Type de programme	SAM	Renseigner le type de programme testé							
Code application	LD	Renseigner le code application							
Type d'action	Mise à jour	Renseigner le type d'action réalisé par le programme							
Nom externe	RISCLI067	Renseigner le nom externe du programme							
Nom PACBASE	LDSM07	Renseigner le nom PACBASE du programme							
Note PQC	100/100	Indiquer la note PQC							
Session PAC		Indiquer en quelle session a été compilé le programme testé							
Commentaires									
<b>Programmes intervenant dans les tests (ne pas oublier les sous-programmes, utilitaires)</b>									
Type de composant	Nom externe	Nom PACBASE	Note PQC	Session PAC	Commentaires				
SAM	NK1LDS09	LDSM07	100		Note PQC correcte				
SD	NK1LDD07	LDSD07	95		Note PQC correcte				
Gestionnaire code retour	NK1LDR00	LDCR01							
Connecteur	NK1LDC01	LDCA01	97		Note PQC correcte				
Connecteur	NK1LDC00	LDCS01	94		Note PQC correcte				
Accesseur/SAD	NK1LD07T	LDAC07	95		Note PQC correcte				
<b>Paramètres des tests via SALEPH</b>									
Environnement									
	CICS WOR	WOR112							
	CICS AOR	CICSCD39							
SALEPH									
	URL	https://rmoa-stmcv1.sf.intra.laposte.fr:444/ws							
	Programme de test générique								
	Emplacement fichiers Q4CRIT								
	Emplacement fichiers Q4REP								
	Nom copy critères	NK1LDS09_criteres							
	Nom copy réponse	NK1LDS09_reponses							
	Nom fichier initialisation	NK1LDS09_initialisation_casXX							
<b>Ressources (accédées par les programmes ou sous-programmes appelés)</b>									
MQ Serie	QALIAS	QLOCAL	Type	Description					
Fichiers	DDNAME	DSNAME	Type	Description					
TS									
Nom de la TS accédée				Indiquer le nom de la TS accédée si cas échéant					
DB2									
Database	Creator	Moteur DB2	Nom Table	Type d'accès	Description				
LDDDBD0AA	GDLDD0AA	DB2T	LDT0RP01	Ecriture	FGR RTG courants				
Armide	Nom Table		Version	Description					
	LDFGRRTG		V1	FAITS GENERATEURS DE RISQUES RTG					

## Annexe n°13 – La base de données Dev'Hire





## Annexe n°14 : Extrait du script SQL de génération de la base de données Dev'Hire

```

create database devhire;
use devhire;
drop database devhire;

CREATE TABLE Agence(
    nomAgence int (1) NOT NULL
    ,CONSTRAINT Agence_PK PRIMARY KEY (nomAgence)
)ENGINE=InnoDB;

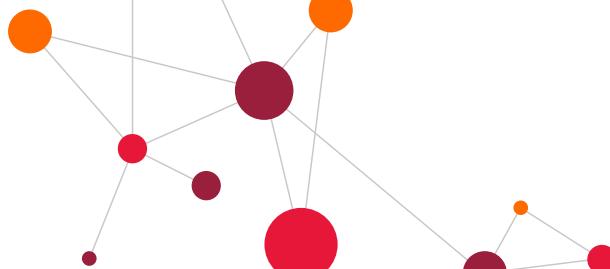
CREATE TABLE Competence(
    id_Competence      Int (11) NOT NULL auto_increment,
    libelleCompetence Varchar (50) NOT NULL ,
    valeur              Bool NOT NULL
    ,CONSTRAINT Competence_PK PRIMARY KEY (id_Competence)
)ENGINE=InnoDB;

CREATE TABLE Qualification(
    id_Qualification      Int (11) NOT NULL auto_increment,
    libelleQualification  Varchar (50) NOT NULL ,
    tauxHoraire            Float NOT NULL
    ,CONSTRAINT Qualification_PK PRIMARY KEY (id_Qualification)
)ENGINE=InnoDB;

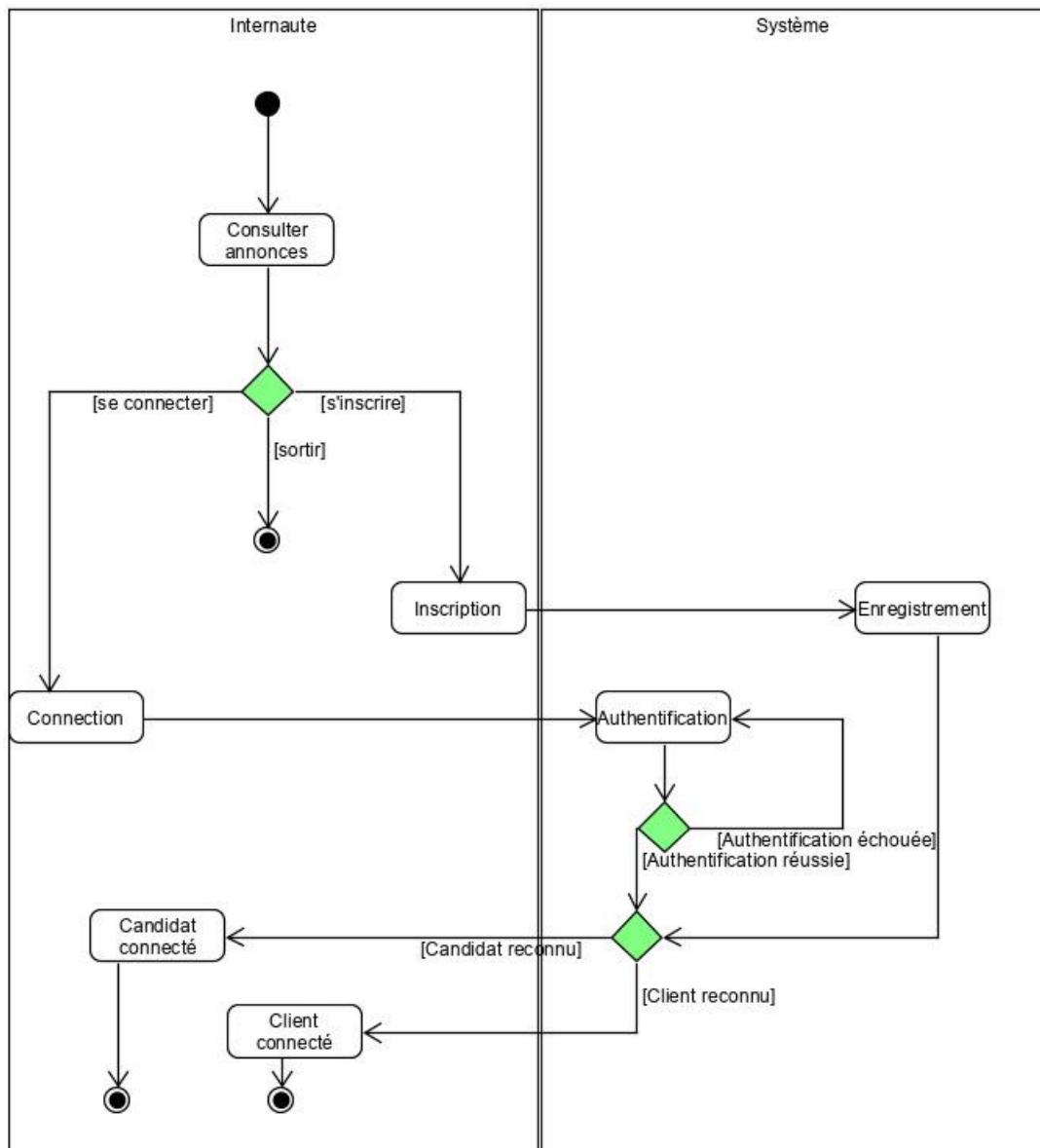
CREATE TABLE Codepostal(
    id_codePostal int (11) NOT NULL auto_increment,
    codePostal     Varchar (10) NOT NULL
    ,CONSTRAINT Codepostal_PK PRIMARY KEY (id_codePostal)
)ENGINE=InnoDB;

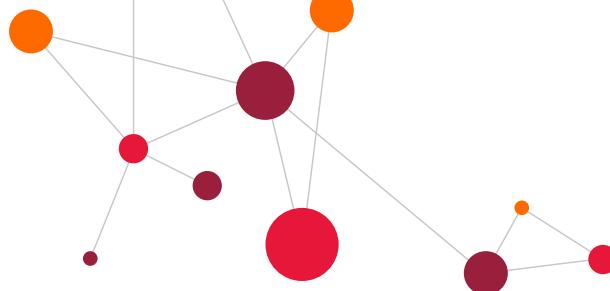
CREATE TABLE Adresse(
    id_Adresse      Int (11) NOT NULL auto_increment,
    libelleAdresse Varchar (200) NOT null,
    id_codePostal  int (10) NOT null
    ,CONSTRAINT Adresse_PK PRIMARY KEY (id_Adresse)
    ,CONSTRAINT Adresse_Codepostal_FK FOREIGN KEY (id_codePostal) REFERENCES
Codepostal(id_codePostal)
)ENGINE=InnoDB;

```



Annexe n°15 : Diagramme d'activité de connexion à un espace particulier



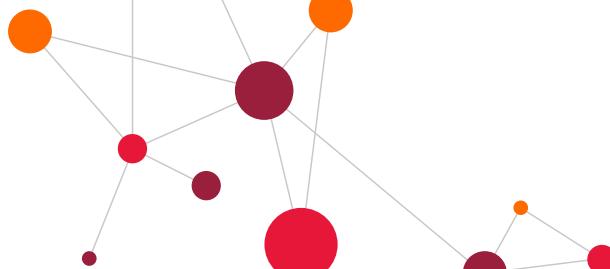


**CGI**

## Annexe n°16 – Extrait du site Web Dev'Hire

The collage of screenshots illustrates the Dev'Hire platform's features:

- Connexion (Top Left):** Shows the login form with fields for Email and Mot de passe.
- Bienvenue dans l'univers de DEV'HIRE !! (Top Right):** A welcome message on the homepage.
- Recherche (Second Row Left):** The search page featuring a search bar and sections for DERNIÈRES ANNONCES (Recent Ads) and CV à déposer (CV to post).
- Créer votre CV (Second Row Right):** The CV creation form with sections for Personal information, Qualifications, Competencies, Transportation methods, Experience, and Identifiers.
- Inscription Entreprise (Third Row Left):** The employer sign-up form with fields for Raison sociale, Adresse, and Code postal.
- Partie Administrateur (Bottom Left):** The administrator panel for managing candidates, showing a table with columns: Dernière MAJ, Id, Nom, Prénom, Qualification, Email, Téléphone, Status, and Action.
- Validation des candidats (Bottom Left):** A detailed view of candidate validation with columns: Nom, Prénom, Qualification, Email, Téléphone, Status, and Action.
- Contact (Bottom Right):** A contact form with fields for Nom, Prenom, Email, Sujet, and Message.



## Annexe n°17 – Rôle du Scrum Master

### Rôles



**AVIS DE CONFIDENTIALITE :** Ce dossier est à destination des membres du jury pour la certification IPI CDAN. Toutes diffusions et/ou reproductions totales ou partielles est proscrite. Ce document peut contenir des renseignements confidentiels appartenant exclusivement au groupe CGI ou à ses filiales.