



Faculté des Sciences et des Techniques

2 Rue de la houssiniere,

44322 Nantes



Capgemini Nantes

16 Mail Pablo Picasso

44000 Nantes

Réalisation d'une version majeure d'une application de gestion de régimes de champ de tir (SERT)

MEMOIRE DE MASTER ALMA M2

Présenté par

Tantely RANDRIAMAHARAVOMANANA

Date de soutenance : 07 juillet 2017

Encadrant en entreprise : Xavier DESNOYERS

Encadrant universitaire : Mourad Chabane OUSSALAH

Confidentialité

Les informations contenues dans ce document doivent rester internes à l'entreprise et ne doivent pas être diffusées, à l'exception du jury de la Faculté des sciences et techniques de l'Université de Nantes chargé de l'évaluer.

Niveau de confidentialité : « Interne Capgemini »

Remerciements

Je tiens à remercier mes tuteurs à Capgemini et à l'Université de Nantes, respectivement Xavier DESNOYERS et Mourad OUSSALAH pour leur accompagnement et leur disponibilité.

Je souhaite remercier aussi l'intégralité des collaborateurs du Cluster GIS pour l'accueil chaleureux et les précieux conseils qu'ils m'ont prodigué tout au long du stage.

Je leur suis aussi reconnaissant de la confiance et des responsabilités qu'ils m'ont accordées.

Sommaire

1.	Introduction.....	1
2.	Présentation de CAPGEMINI	2
2.1	Description générale	2
2.1.1	Historique	2
2.1.2	Quelques chiffres.....	3
2.1.3	Les 7 valeurs du groupe.....	5
2.2	Présentation du cluster GIS.....	6
3.	Présentation du projet	7
3.1	Contexte général	7
3.2	Description de SERT.....	8
3.3	Analyse de l'existant.....	8
3.3.1	Fonctionnement actuel de SERT.....	8
3.3.2	Architecture logique de SERT	11
3.3.3	Architecture physique	14
3.3.4	Modèle conceptuel de données.....	17
3.3.5	Structure des codes sources de l'application	18
3.4	Objectifs du projet.....	19
3.4.1	Nouveaux cas d'utilisation.....	20
3.4.2	Amélioration continue.....	21
3.5	Missions.....	21
3.6	Gestion de projet.....	22
3.6.1	Méthode de travail.....	22
3.6.2	Conduite de projet	22
4.	Réalisation	24
4.1	Monter en compétence sur le projet SERT	24
4.2	Intégration de JavaFX avec Swing	25
4.2.1	Swing	25
4.2.2	JavaFX	26
4.2.3	JavaFX/Swing interopérabilité.....	26
4.3	Développement des évolutions.....	29
4.3.1	Structure de base de l'IHM de SERT	29
4.3.2	Gestion des ESID.....	31
4.3.3	Tableaux de bord.....	32

4.3.4	Alerte archivage.....	34
4.3.5	Historique de modification de statut	36
4.3.6	Gestion des documents	36
4.4	Tests de l'application.....	37
4.5	Corrections des anomalies	40
4.6	Contraintes et problèmes rencontrés	40
5.	Conclusion générale	42
6.	Liste des abréviations	43
7.	ANNEXE	44
8.	Liste des tableaux	47
9.	Liste des figures	47

1. Introduction

La validation de notre formation de Master II en Architectures Logicielles (ALMA) de la faculté des sciences de Nantes passe par un stage de six mois au sein d'une entreprise. Ce stage nous permet d'intégrer le monde professionnel et de mettre en pratique ce que nous avons acquis pendant notre cursus universitaire. J'ai effectué mon stage de fin d'étude au sein de la société Capgemini dans leur agence de Nantes. J'ai intégré en particulier le « cluster SIG » qui est une entité du centre d'excellence cartographique. J'ai choisi l'offre de stage de Capgemini parmi tant d'autres parce que c'est l'une des plus grandes entreprises de service numérique en France et en Europe. Les missions décrites pour le stage m'ont attiré particulièrement car il s'agit d'intervenir sur toutes les phases de construction d'un projet de système d'information en environnement SIG (Système d'Information Géographique). Ce qui m'a permis de mettre en profit mes connaissances et mes expériences en conception et développement de logiciels.

Ce rapport a pour objet de présenter les travaux d'ingénierie que j'ai réalisés pendant mon stage de fin d'études au sein l'entreprise Capgemini. Durant mon stage, j'ai travaillé sur la réalisation d'une version majeure d'une application de gestion de régimes de champ de tir appelée SERT (Système d'Edition des régimes de champs de Tir). Sur ce projet, je suis intervenu sur les phases d'analyse, de conception, de développement et de tests.

Dans un premier temps je ferai une présentation de l'entreprise et de mon environnement de travail. Puis j'exposerai mes missions, lors de ce stage avant de dresser un bilan de celui-ci.

2. Présentation de CAPGEMINI

Fort de plus de 180 000 collaborateurs dans le monde en 2016, dont plus de 25000 en France et présent dans plus de 40 pays, Capgemini est l'un des leaders mondiaux du conseil, des services informatiques et de l'infogérance. Le Groupe a réalisé en 2015 un chiffre d'affaires de 11,9 milliards d'euros.

Capgemini propose quatre disciplines clés : Service de conseil (Capgemini Consulting), Services d'intégrations de systèmes, Services informatiques de proximité (Sogeti) et Services d'infogérance. Profitant d'une bonne réputation et d'une implantation dans plusieurs régions (de la France et du monde), Capgemini est présent dans de nombreux secteurs (secteur public, services financiers, produits de grande consommation, télécommunications...) avec des clients très variés.

2.1 Description générale

Fondée en France, Capgemini est aujourd'hui la première entreprise de service du numérique dans le secteur des services informatiques en France. Elle fait également partie du top 15 mondial.

2.1.1 Historique

Le groupe Capgemini s'est fondé autour de nombreuses fusions et acquisitions, ce qui influence le groupe tel qu'il est aujourd'hui et ses activités. Le groupe prend son origine avec la fondation de la société SOGETI (SOciété pour la Gestion de l'Entreprise et le Traitement de l'Information) en 1967 à Grenoble par Serge KAMPF. Les années 73 et 74 marqueront la première grande croissance extérieure du groupe avec des prises de participation dans le groupe CAP et l'acquisition de la société Gemini Computer Systems. Une autre croissance externe permettra au groupe de prendre une part importante dans l'entreprise du marché de l'infogérance, Sesa, changeant le nom des activités françaises en Cap Sesa. Les acquisitions successives de Bossard Consulting et de la branche consulting d'Ernst & Young installeront le groupe comme un acteur majeur du consulting en Europe qui prendra le nom de Capgemini en 2004.



Figure 1 - Frise chronologique de l'histoire de Capgemini

2.1.2 Quelques chiffres

Le Groupe a réalisé en 2015 un chiffre d'affaires de 11,9 milliards d'euros, en nette évolution par rapport aux années précédentes comme le montre le graphique suivant :



Figure 2 - Evolution du chiffre d'affaires

(Source : <https://www.fr.capgemini.com/rapport-annuel/2015>)

Il me paraît important de décrire ce graphique en deux aspects pour bien comprendre le fonctionnement de Capgemini :

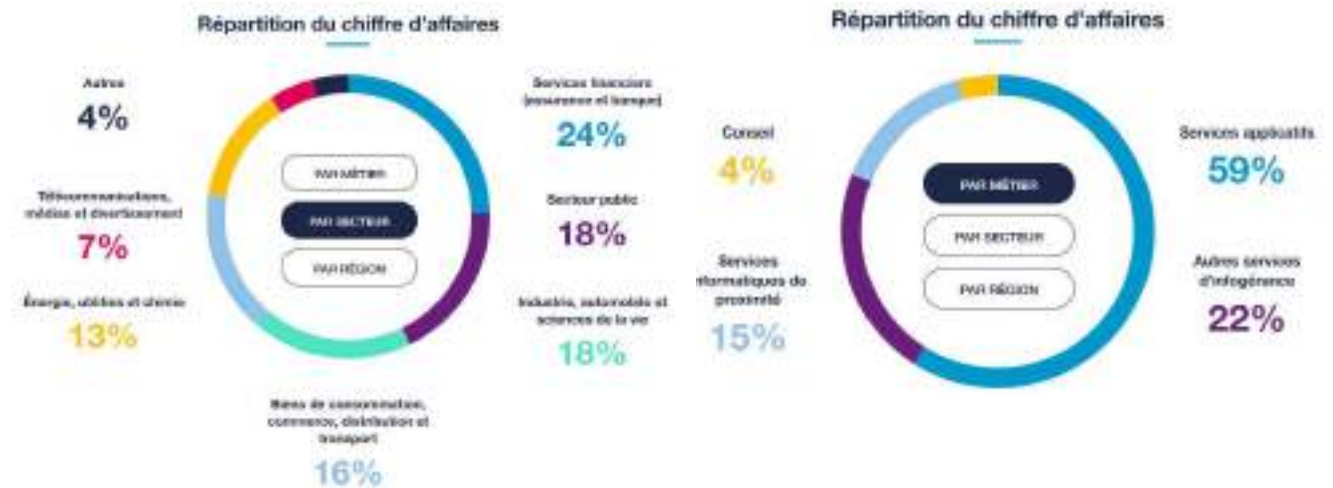


Figure 4- Répartition du C

Figure 3- Répartition du CA par métier

Le graphique de gauche montre à quel point les clients du groupe Capgemini sont diversifiés, il n'y a pas de secteur « majeur » qui prends le dessus sur tous les autres, bien qu'on puisse noter que le groupe a des échanges très forts avec les services financiers.

Le graphique de droite note l'énorme majorité des services applicatifs dans la répartition du chiffre d'affaire par métier. C'est d'ailleurs dans cette catégorie de métier que mon poste d'alternant se classe.

Cette activité importante lui permet d'employer 181000 collaborateurs dans plus de 40 pays. Le dernier graphique présenté ci-dessous expose l'énorme capacité de collaborateurs du groupe et la constante évolution de ce nombre de collaborateurs. Je souligne également que plus de 40% des embauchés sont des jeunes diplômés.



Figure 5- Evolution du nombre de collaborateurs

2.1.3 Les 7 valeurs du groupe

Capgemini accorde beaucoup d'importance à la transmission de ses valeurs fondatrices, et elles m'ont été inculquées dès mes premiers jours à Capgemini, via les échanges avec mes collègues et le programme e-learning. Ce programme est obligatoire pour tous les collaborateurs, et consiste à réaliser de petits cours et quizz en ligne, dans le but de nous transmettre les bases de l'éthique du groupe. Voici ces 7 valeurs :

- **L'honnêteté**, et en particulier le refus de toutes pratiques déloyales dans la conduite des affaires en vue d'obtenir un contrat ou un avantage particulier. Des règles précises relatives aux commissions, cadeaux et dons de toute nature ont été établies dans cet esprit.
- **L'audace**, au sens du goût d'entreprendre, de l'envie de prendre des risques et de s'engager, dans le respect d'un principe général de prudence et de lucidité sans lequel le manager audacieux peut se révéler dangereux.
- **La confiance**, qui implique la volonté de responsabiliser les hommes et les équipes, et de confronter les responsables aux effets de leurs actions et décisions. La confiance

implique également l'ouverture d'esprit et une grande transparence dans la circulation de l'information.

- **La liberté**, ou encore la créativité, l'innovation, l'indépendance d'esprit et le respect des autres dans leurs cultures, leurs habitudes et leurs différences, ce qui est indispensable dans un Groupe présent dans plus d'une quarantaine de pays et comptant près de 180 000 personnes représentant une centaine de nationalités différentes.
- **La solidarité** et la capacité à partager solidairement les bons et les mauvais moments.
- **La simplicité** dans les relations avec les autres et dans l'approche des sujets.
- **Le plaisir** enfin, sans lequel tout projet d'entreprise est difficile, voire impossible à réaliser.

2.2 Présentation du cluster GIS

De par la taille du groupe Capgemini, il est évident que sa structure est hiérarchisée en blocs et clusters. Cette structure est déterminée sur le critère du secteur d'activité (applications, infogérance..) puis de l'activité client. L'entreprise est axée sur une organisation à double entrée, favorisant la spécialisation métier et client. Elle peut ainsi se spécialiser sur une typologie de client, des rapprochements pouvant exister entre les projets d'entreprises autour d'un même domaine ; ce qui lui permet de répondre aux appels d'offre de manière plus précise avec des références plus proches des clients.

Je me situe ainsi dans la branche Application Services France, qui regroupe 5 « Practice ». Le cluster GIS est englobé dans la practice CSD (Custom Software Development) et plus précisément dans la practice CSD TMP (Transport Mutuelles Public). Enfin, cette practice est divisé en 4 « skill group », celui de mon service étant le skill group « Transport public & centre d'excellence GIS », piloté ici à Nantes.

Le cluster GIS a été créé il y a 3 ans et demi, dans le but de mutualiser des ressources et des projets à vocation SIG (Systèmes d'Informations Géographiques). Le cluster est composé d'environ 40 personnes et gère principalement des projets en TMA (Tierce Maintenance Applicative) pour du MCO (Maintien en Conditions Opérationnelles) et des évolutions fonctionnelles techniques.

3. Présentation du projet

3.1 Contexte général

L'Etablissement du Service d'Infrastructure de la Défense (SID) assure le soutien commun à l'ensemble du ministère sur les aspects immobiliers aussi bien sur le sol national qu'à l'étranger. Il réalise le système d'information métier permettant de conduire les opérations d'infrastructure du service.

Le DECSST (Département d'Expertise Champs et Stands de Tir) est le bureau qui traite des infrastructures de tir. Il est chargé d'étudier :

- Les caractéristiques techniques des installations de tir d'un type nouveau.
- Les règles techniques et administratives auxquelles sont soumises ces infrastructures pour assurer la sécurité des biens et des personnes.
- Les mesures de sécurité à appliquer lors de l'exécution des tirs dans un stand de tir ou sur un champ de tir.
- Les modifications qui doivent être apportées aux documents réglementaires ou aux infrastructures de tir de manière à répondre au mieux aux besoins des utilisateurs et de coïncider avec la réglementation générale.

A ce titre il établit les régimes de champs de tir qui fixent les règles à respecter pour assurer la sécurité des personnes et des biens lors de l'exécution des tirs. Il participe donc activement à la préparation opérationnelle des forces.

3.2 Description de SERT

Un régime de champ de tir est la notice d'emploi de l'infrastructure de tir, couverte ou à l'air libre. Il précise à partir de quelles positions et vers quels objectifs, avec quelles armes et munitions le tir est autorisé. Il décrit les zones dangereuses, les mesures de sécurité à prendre, les éventuels tirs simultanés autorisés.

SERT ou « Système d'Edition des régimes de champs de Tir » est un logiciel que le DECST du ministère de la défense s'est doté en 2014. Comme son nom l'indique, SERT est une application qui permet d'élaborer les régimes des champs de tirs en tenant compte de l'évolution de la réglementation relative aux infrastructures de tir.

Un régime de champ de tir permet de connaître les usages et les contraintes du champ de tir. C'est un document normalisé propre à chaque infrastructure de tir (champs de tir). Il est établi par les traitants du DECST.

L'élaboration des régimes de tir est complexe et ceux-ci doivent être remis à jour régulièrement, le déploiement de SERT a permis au DECST d'améliorer l'efficacité de ses rédacteurs de par la réduction des délais de production des documents.

3.3 Analyse de l'existant

3.3.1 Fonctionnement actuel de SERT

SERT est une application qui présente des fonctionnalités compliquées. Sa manipulation n'est pas une chose évidente puisque qu'il y a de très nombreux détails dans les fonctionnalités de SERT. Les documentations sur le projet m'ont beaucoup aidé à comprendre l'aspect fonctionnel. En effet, j'ai eu l'occasion de lire et d'utiliser des documents sur SERT comme les manuels d'utilisation, les documents de conception et les pages wiki. Les tests et les manipulations directes de l'application sur un environnement de test m'ont aussi aidé à voir comment sont les comportements de SERT.

3.3.1.1 Types de régime

Il y a trois types de régime qui peuvent être édités sur SERT : le RIE, le RIC et le RE.

Un RIE ou Régime intérieur élémentaire est un régime intérieur adapté à un stand de tir ou un champ de tir unique. Il définit les possibilités offertes en fonction du couple armes-munitions et de la domanialité du terrain concerné par la zone dangereuse.

Un RIC ou Régime intérieur commun est un régime intérieur adapté à un complexe de tir.

Un RE ou Régime extérieur est un régime extérieur adapté à un terrain utilisé par le Ministère de la Défense. Seul document obligatoirement diffusé aux autorités civiles, il définit l'ensemble des mesures prises pour assurer la sécurité des personnes et des biens.

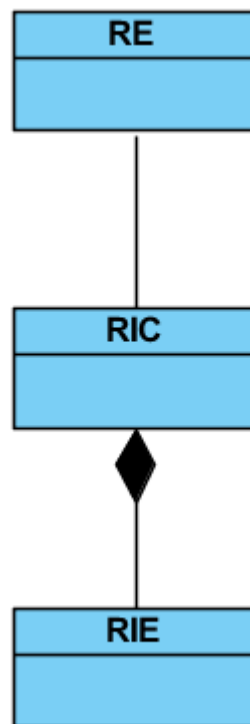


Figure 6 Composition d'un régime de tir

Le petit diagramme de classe ci-dessus montre la relation entre RE, RIE et RIC. Un RE est associé à un seul RIC. Un RIC peut être composé de un ou plusieurs RIE.

3.3.1.2 Cas d'utilisation simplifiés de SERT

SERT est un projet qui dispose de nombreux cas d'utilisation car le domaine métier est très complexe. Je vais vous en présenter quelques uns dans ce paragraphe afin de donner une image du fonctionnel du projet.

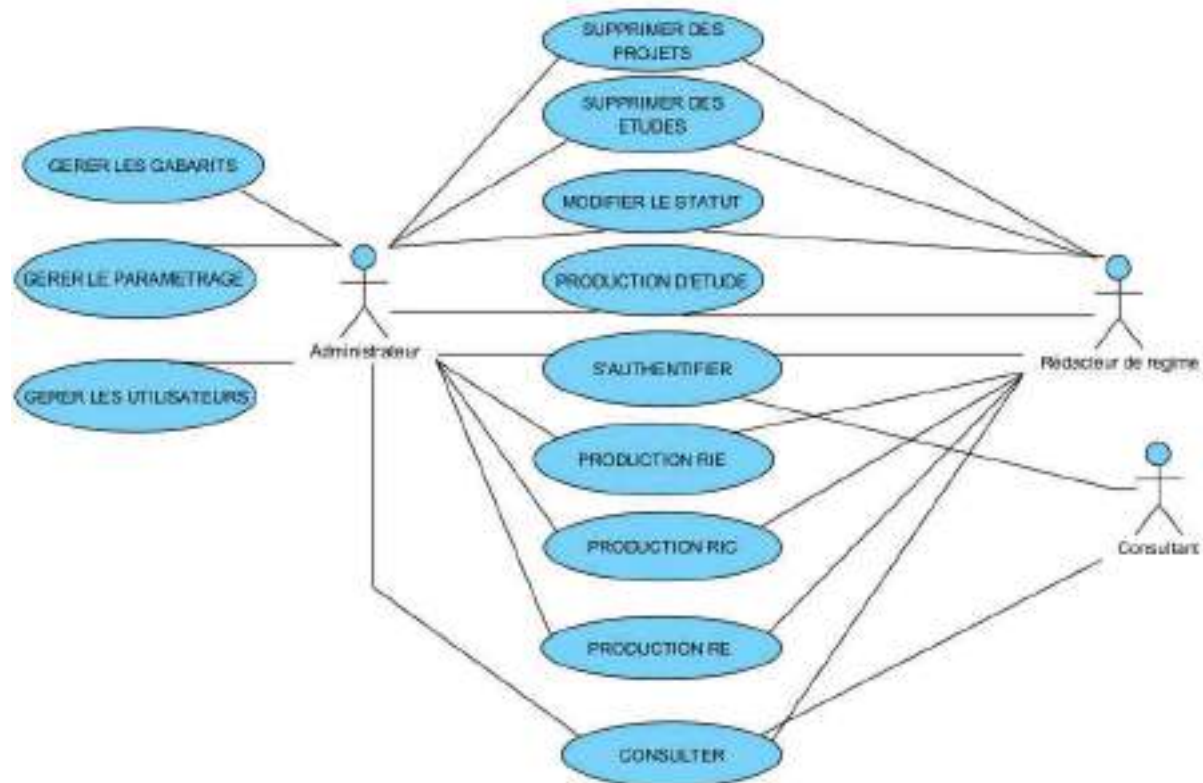


Figure 7 Diagramme de cas d'utilisation SERT 1.1.7

Il y a donc trois types d'utilisateur sur SERT : administrateur, Rédacteur de régime et consultant. Ces trois types utilisateurs sont différents par rapport aux fonctionnalités auxquelles ils ont accès.

L'administrateur a accès à toutes les fonctionnalités. C'est le seul qui a accès aux fonctionnalités d'administration comme gérer les utilisateurs, gérer le paramétrage et gérer les gabarits.

Le rédacteur de régime a accès aux fonctionnalités relatives à la production de régime comme modifier le statut de projet ou d'étude, supprimer des projets et produire des régimes. L'étude

est une sollicitation ponctuelle qui ne rentre pas dans le cadre du projet pour laquelle le traitant doit pouvoir effectuer une simulation graphique et en sortir les résultats que l'on retrouve dans les autres régimes. L'étude pourra éventuellement être transformée en projet.

Le dernier type d'utilisateur est le consultant. L'utilisateur consultant ne peut que consulter. Il peut donc rechercher des projets et les consulter mais son droit d'accès est en lecture seule.

3.3.1.3 Processus de réalisation d'un régime

Actuellement, les ESID rédigent des programmes tir sur demande de ses clients à partir d'une Fiche d'Expression du Besoin (FEB). Dans l'impossibilité de réaliser les études de faisabilité, les ESID sollicite le BECST pour palier à cette problématique. Le BECST réalise l'étude de faisabilité dans SERT et la transmet à l'ESID demandeur qui la transforme en «programme tir».

Ce «programme tir» est transformé en dossier initial. Ce dossier est à nouveau transmis au BECST. A l'aide de ce dossier, le BECST finalise et rédige le régime de tir dans SERT.

S'il garantit la qualité de la réalisation du régime, ce mode de fonctionnement est très lourd et demande beaucoup de temps (aller-retour par courrier officiel).

3.3.2 Architecture logique de SERT

Dans le contexte du projet SERT, l'architecture retenue est du type « Architecture Orientée Services » (SOA). Le présent chapitre est tiré du document d'architecture existant de SERT. Il utilise les termes service et sous-système. Leur définition étant la suivante :

- **Service** : Fonctionnalité attendue du système. Niveau terminal de la décomposition logique : les services sont les feuilles de la déclinaison en sous-systèmes & services fonctionnels.
- **Sous-système** : Regroupement cohérent de services fonctionnels. Un sous-système peut lui même être décomposé en sous-système de niveaux inférieurs.

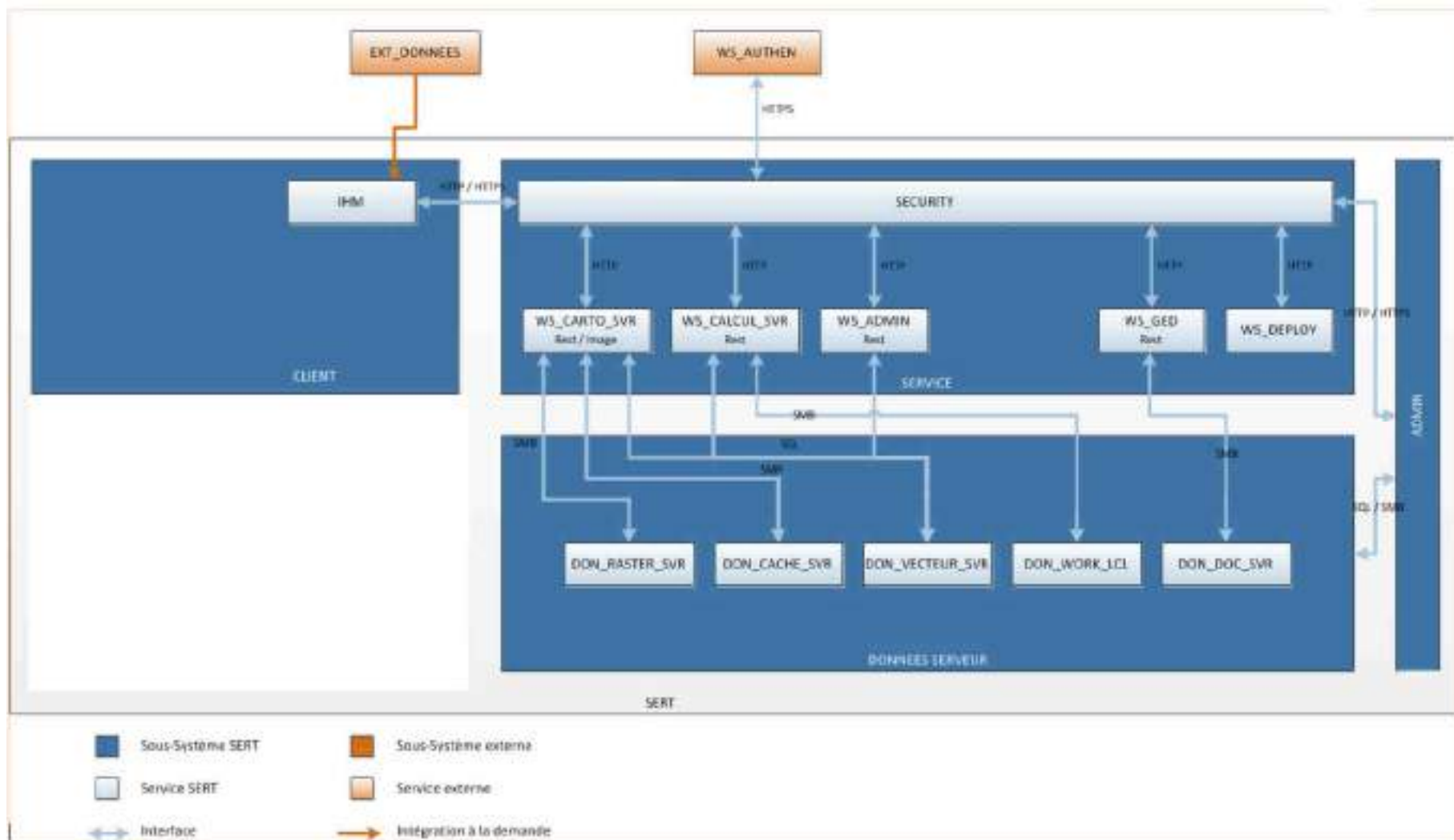


Figure 8 Architecture logique de SERT

- Sous-système « DONNEES SERVEUR »

Le sous-système « DONNEES SERVEUR » est responsable du stockage, de la sauvegarde et de l'archivage des données du système SERT. Il comporte les services suivants : DON_VECTEUR_SVR, DON_RASTER_SVR, DON_CACHE_SVR, DON_DOC_SVR, DON_WORK_LCL.

Voici les conventions de nommage des services :

- DON signifie qu'il s'agit d'un service de stockage de données.
- Ce qui suit fait référence au type de données à stocker.
- SRV signifie qu'il s'agit d'un service.

- Sous-système « SERVICE »

Le sous-système « SERVICE » constitue le cœur de l'application SERT. Il fournit les principales fonctionnalités métier sous forme de services web.

- Service « SECURITY »

Le service « SECURITY » remplit plusieurs rôles :

- Il vérifie qu'un utilisateur peut accéder à l'application.
- Si un utilisateur peut accéder à l'application, il fournit son rôle.
- A chaque requête, il s'assure que l'utilisateur est bien habilité à l'exécuter et il trace l'action de l'utilisateur.

L'accès à tous les services du sous-système (WS_CARTO_SVR, WS_CALCUL_SVR, WS_EXTRACT et WS_GED) est contrôlé par ce service.

- Sous-système « Client »

- Service « IHM »

Le service « IHM » fournit le frontal pour l'utilisateur. L'accès à l'application est contrôlé par un login / mot de passe.

- Sous-système « Admin »

Le sous-système « ADMIN » fournit les outils nécessaires à l'administration de la plateforme.

Il offre donc les services suivants :

- Gestion des utilisateurs
- Gestion des données (géoréférencement des rasters, etc ...)
- Gestion des services de carte
- Gestion du paramétrage
- Consultation des statistiques d'utilisation.
- Gestion des gabarits

3.3.3 Architecture physique

L'infrastructure physique de SERT est présentée dans ce chapitre. Le présent chapitre utilise les termes **Composant** et **Package**. Leur définition étant la suivante :

- **Composant** : Chaque brique applicative présente dans l'application SERT est appelée Composant. Niveau terminal de la décomposition physique : les composants sont les feuilles de la déclinaison en package & composants.
- **Package** : Regroupement cohérent de composants. Un package peut lui même être décomposé en packages de niveaux inférieurs.

Un composant est donc décrit pour chaque besoin distinct et appartient à un package. Les composants pourront ensuite être regroupés lors du déploiement (exemple : partage d'une base de données pour plusieurs besoins).

Les composants ont la règles de nommage suivante : <Package>.<Type>.<Nom>

Les types de composants sont :

- « A » (Applicatif),
- « B » (Base de données),
- « D » (Données dans un espace disque).

Un Bundle désigne le regroupement de plusieurs logiciels, packages ou composants définissant un ensemble cohérent et déployable unitairement.

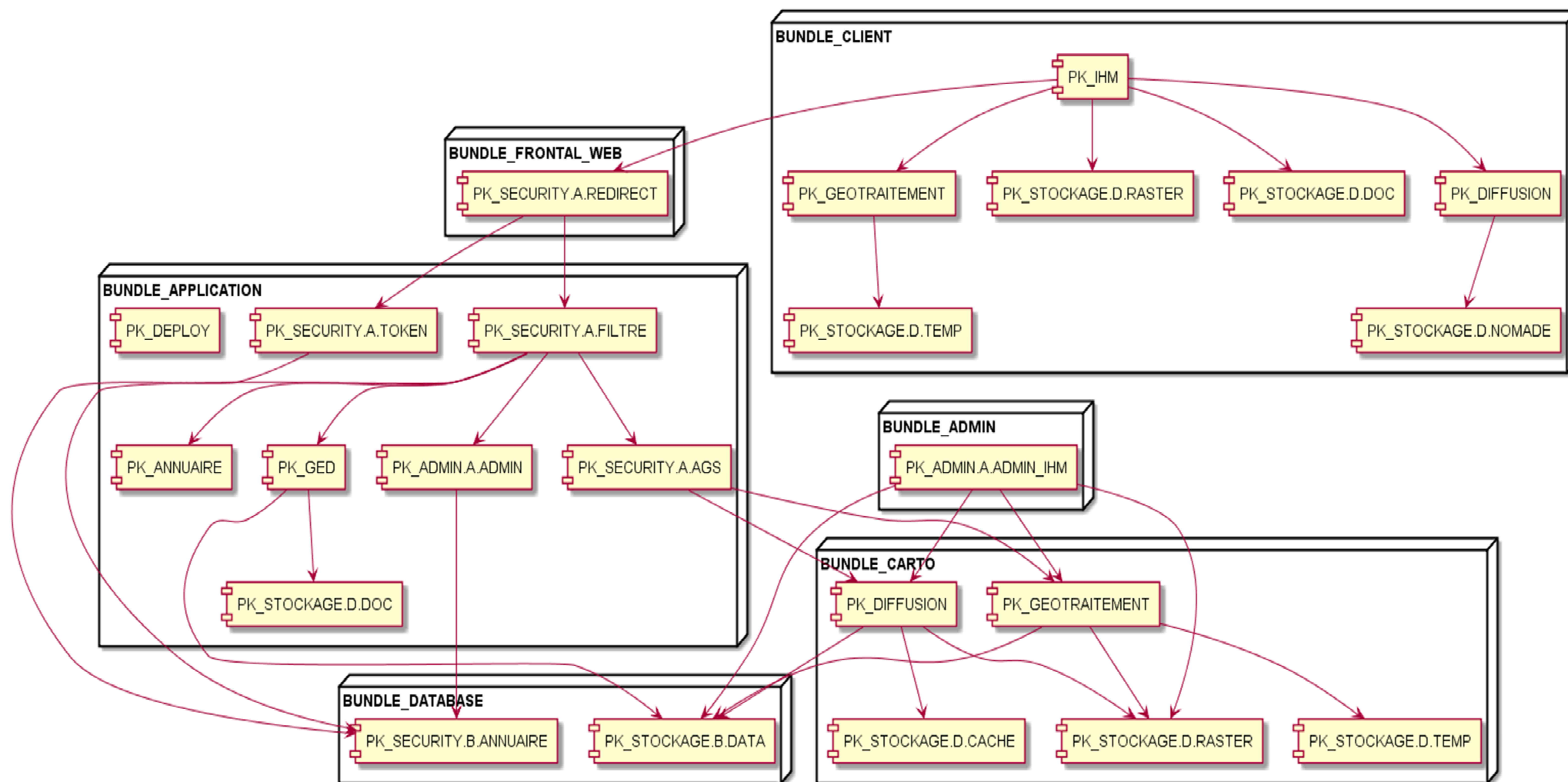


Figure 9 Architecture physique

3.3.4 Modèle conceptuel de données

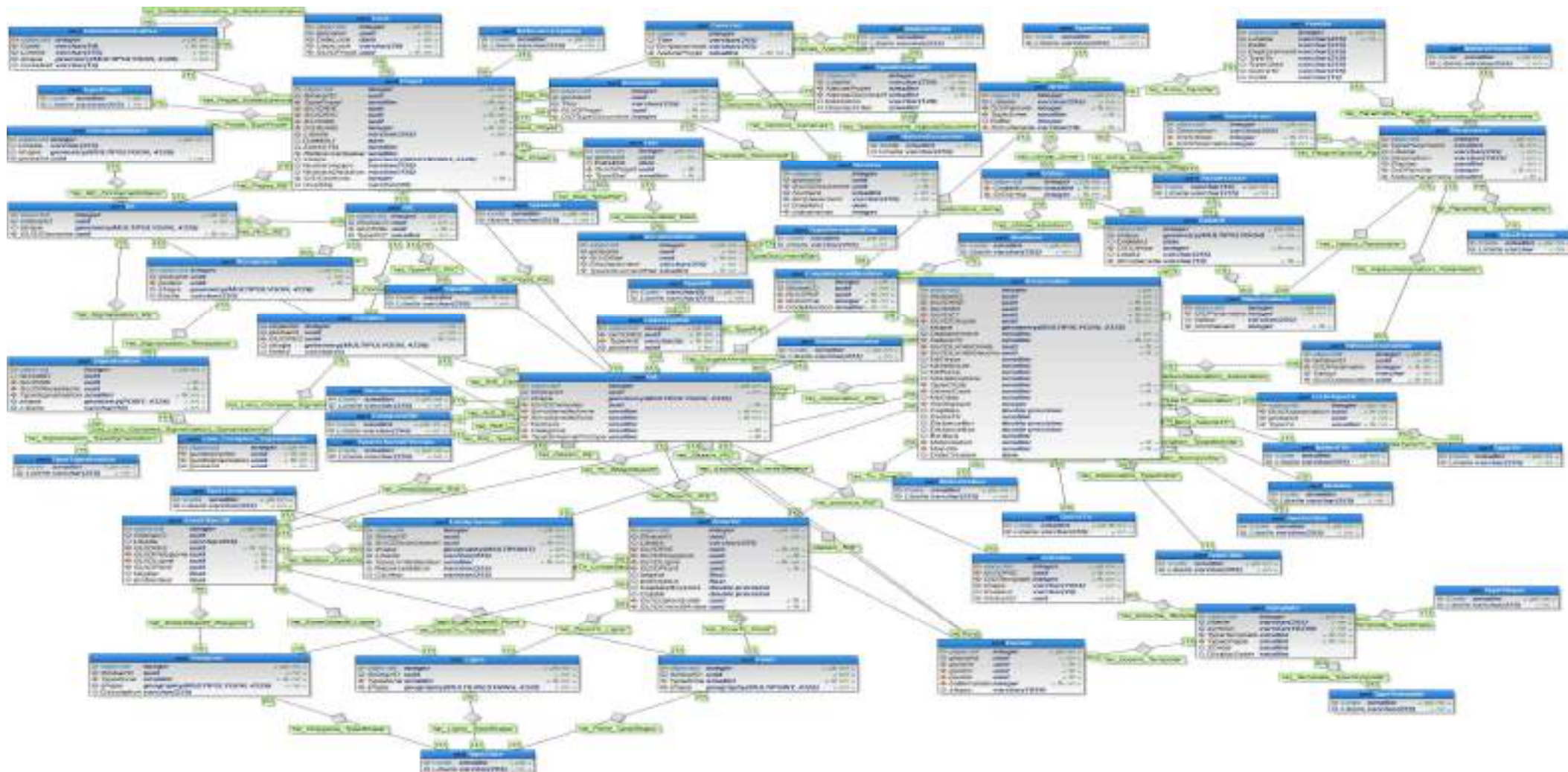


Figure 10 Modèle conceptuel de données

3.3.5 Structure des codes sources de l'application

Les services et/ou sous-systèmes définis dans l'architecture logique de SERT sont développés et implémentés physiquement dans des projets sous forme de composant. Les tâches de développement qui me sont affectées dans mes missions concernaient que quelques services de SERT.

3.3.5.1 *Les projets java*

J'ai travaillé surtout sur trois projets : le projet « SERT_IHM », le projet « SERT_WEB » et le projet « GenerateDocs ».

- Projet « SERT_IHM » : Le projet SERT_IHM implémente les sous-systèmes « Client » et « Admin ». La totalité de l'interface graphique de SERT est développée dans ce projet. C'est un projet Java géré avec Apache Maven.
- Projet « SERT_Web » : Ce projet implémente une partie du service « SECURITY ». Le but de ce projet est de sécuriser l'accès aux web services des autres composants. C'est un projet Java web géré avec Apache Maven.
- Projet « GenerateDocs » : Ce projet implémente le service « WS_GED_SVR ». Ce service regroupe les fonctionnalités nécessaires à la gestion des documents issus de SERT. Ce projet comporte aussi des éléments transverses aux deux autres projets.

3.3.5.2 *Les versions actuelles des composants utilisées*

L'application SERT a été créée en 2013. Depuis, de nombreuses évolutions ont été apportées sur elle. Ces évolutions consistaient à réaliser de nouvelles fonctionnalités et aussi à garder à jour les composants de l'application. SERT est actuellement à la version 1.1.7.

La plupart des composants de SERT tournent sur du Java. Par exemple, le composant « IHM », qui est l'interface utilisateur, est développé avec du Swing. La version de Java utilisée actuellement par SERT est la version 7. L'utilisation de Swing peut poser des problèmes pour les futures évolutions et pour les mises à jour de SERT. En effet, Swing n'est plus la bibliothèque graphique officielle dans les récentes versions de Java. Dans Java 8, Swing est obsolète et remplacé par JavaFX. Aussi, la future version 100.0.0 d'ArcGIS Runtime Java SDK

pourra être implémentée avec du JavaFX. Actuellement, SERT utilise l'API ArcGIS Runtime Java SDK 10.2.4.

Voici la liste des versions des outils actuellement utilisés dans le projet SERT :

Nom de l'outil	Version actuelle utilisée	Description
ArcGIS Runtime Java SDK	10.2.4	
Java JDK	1.7.0.79	
Itextrpdf	5.5.6	Export pdf
documents4j	1.0.2	Conversion de format de document
Pdftbox	1.8.4	
Jenkins	1.642.3	Outil d'intégration continue
Eclipse	Mars 2 (4.5.2)	IDE
Sonar Qube	5.6.3	Assurer la qualité du code source
SonarLint	3.1.0	Plugins Eclipse pour utiliser sonar

Tableau 1 Versions actuelles des outils

3.4 Objectifs du projet

L'objectif principal du projet est de réaliser la version 2.0 de l'application SERT. Cette montée de version correspond aux développements de nombreuses évolutions sur l'application.

Afin d'optimiser les temps de traitement des dossiers, le DECST souhaite faire évoluer le logiciel SERT de façon à donner aux ESID les outils nécessaires à l'élaboration de « programmes tir » fiables. Nombreuses spécifications sont alors à concevoir et à développer pour répondre à ce besoin.

3.4.1 Nouveaux cas d'utilisation

Voici une liste non exhaustive des cas d'utilisation de la nouvelle version 2.0 de SERT :

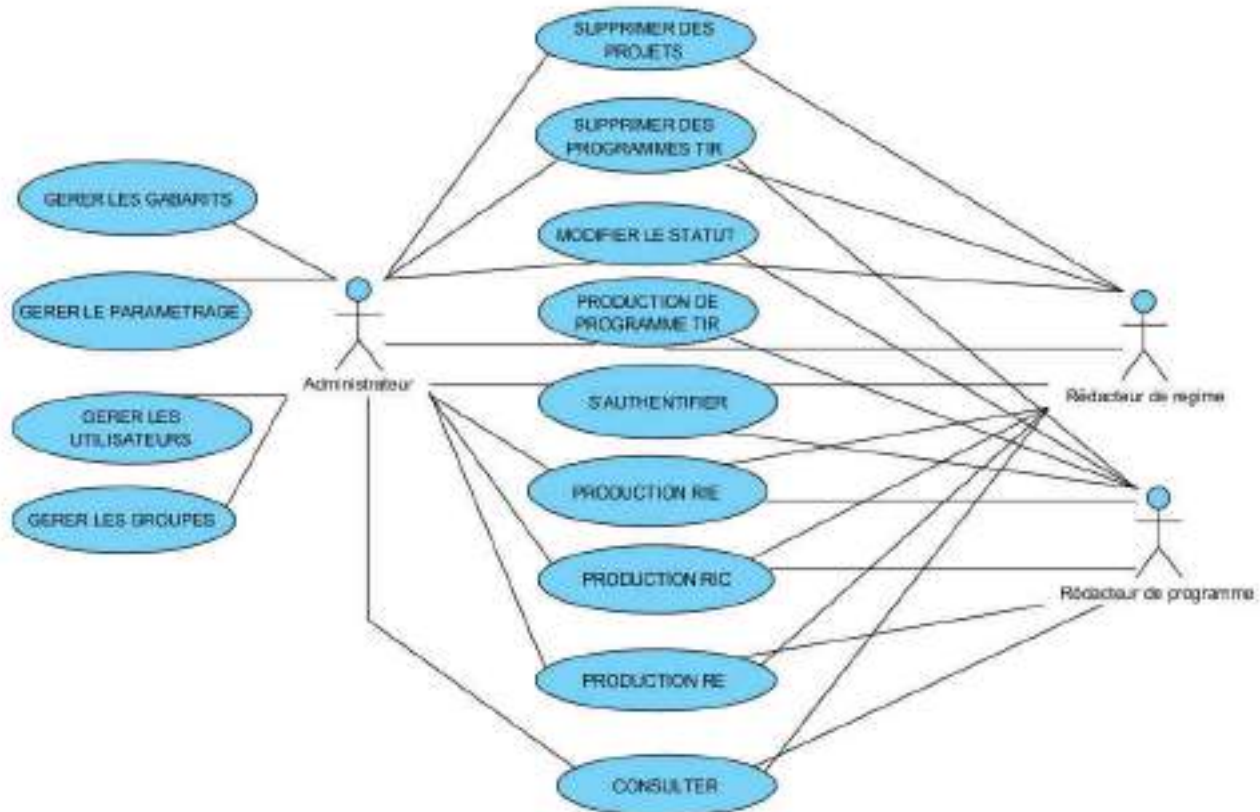


Figure 11 Diagramme de cas d'utilisation SERT 2.0

Le fonctionnement actuel de SERT (cf 3.3.1) ne permet de réaliser que des régimes de tir. Dans ce nouveau diagramme de cas d'utilisation de SERT, deux grandes évolutions sont observées. La première est la possibilité de réaliser des « programmes de tir ». La deuxième est la création d'un nouveau type d'utilisateur « rédacteur de programme ».

Le profil « rédacteur de programme » permet la rédaction et la consultation d'un « programme tir » avec restriction liée à sa région d'appartenance (groupe ESID). Les utilisateurs de ce profil ne peuvent voir que les RIE, RE ou RIC (quel que soit le statut) rattachés à leur propre groupe.

La notion de groupe est aussi introduite dans la nouvelle version de SERT. Chaque utilisateur doit appartenir à un groupe. L'administrateur fonctionnel SERT doit avoir la possibilité d'attacher un RIE, un RE ou un RIC (quel que soit le statut) à un groupe. Des nouvelles règles de gestion sont intégrées par rapport aux groupes.

3.4.2 Amélioration continue

Aussi, pour assurer l'amélioration continue de SERT, les interfaces graphiques, liées aux évolutions à développer, seront réalisées avec JavaFX. L'intégration de JavaFX dans SERT anticipe les futures mises à jour d'ArcGIS Runtime Java SDK et de Java.

Cependant, refaire l'ensemble des interfaces graphiques de l'application en JavaFX prendra beaucoup de temps et sera très couteux. Il faut alors étudier la faisabilité d'une intégration partielle de JavaFX dans SERT. C'est-à-dire qu'il faut voir si on peut développer du JavaFX sur du Swing et de faire cohabiter les deux technologies dans une même application.

3.5 Missions

Mes principales ont été de :

- monter en compétence sur le Projet SERT. L'objectif était de comprendre SERT sur ses aspects techniques et fonctionnels ;
- monter en compétence sur Swing et JavaFX. L'objectif était d'intégrer JavaFX avec Swing pour assurer l'amélioration continue de l'application ;
- développer des évolutions pour la nouvelle version 2.0 de SERT ;
- participer aux phases de test du projet ;
- participer à la correction des anomalies recensées dans les tests.

3.6 Gestion de projet

3.6.1 Méthode de travail

L'équipe projet SERT était constituée de sept personnes au début. A partir de la phase de test d'intégration, l'équipe était réduite à 5 personnes. Pascal BOENNEC, « Delivery Service Manager » du cluster GIS, est le chef de projet. Il veille à ce que les plannings et les engagements fonctionnels soient bien respectés, et surtout il gère l'aspect financier du projet. Xavier DESNOYERS, qui est responsable de l'application, est en contact permanent avec le client et les développeurs. C'est lui qui anime la « daily » ou la « DSTUM », Daily Stand Up Meeting, un élément central du scrum management mis en place par Capgemini dans la plupart des projets. Tous les matins et pendant 15 minutes maximum, chaque membre de l'équipe décrit ce qu'il a fait la veille, met à jour ses tâches et ses restes à faire (RAF), annonce ce qu'il compte faire aujourd'hui, et informe sur la présence ou non de points bloquants.

3.6.2 Conduite de projet

Une méthode de conduite de projet est toujours à adopter dans un projet de développement de système d'information. En effet, nous avons ici un projet de développement informatique dont l'objectif d'avoir un logiciel de qualité qui répond aux besoins du client. Le modèle de démarche en « V » a été adopté par notre responsable d'application dans ce projet.

3.6.2.1 Le cycle en V

La démarche en V consiste à ordonner l'exécution de chaque phase du projet d'une façon à ce que chaque phase est accompagnée d'une fin d'étape destinée à formaliser la validation de la phase écoulée avant de passer à la phase suivante.

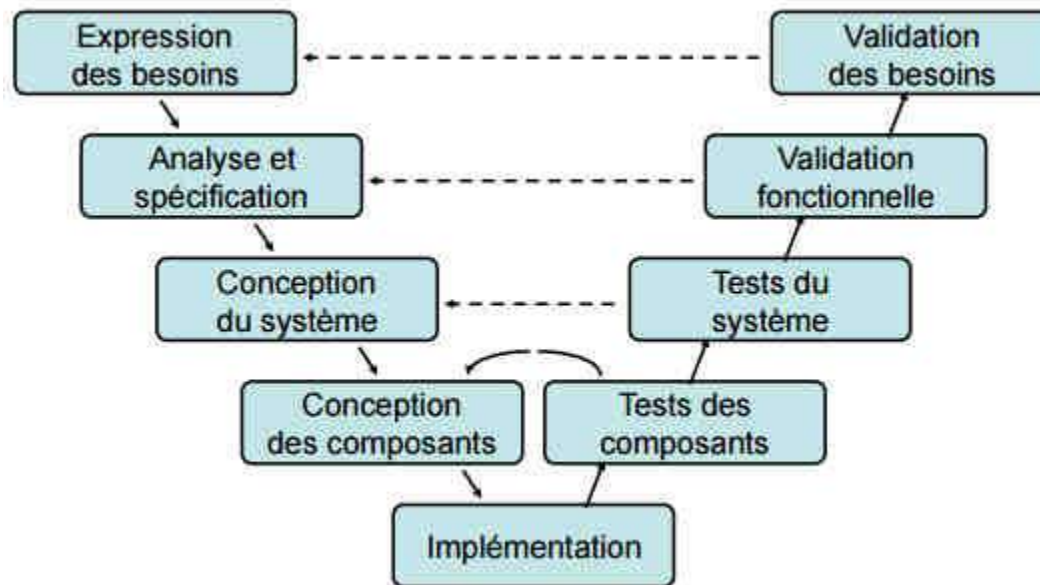


Figure 12 Démarche cycle en V

3.6.2.2 Les phases du projet

L'idée est d'abord de découper le projet en phase. Chaque phase a un objectif précis et regroupe des tâches spécifiques dans l'atteinte de cet objectif. Les phases faciliteront la gestion du projet dans sa mise en œuvre. Le projet SERT comportait les phases suivantes :

- ⇒ Phase d'expression des besoins : Le but de cette phase est de recueillir les besoins du client. Des ateliers ont été organisés avec le client pour cela. Une fois tous les besoins recueillis, on peut les catégoriser selon les fonctionnalités du système.
- ⇒ Phase d'analyse et de spécification : Le but ici est d'élaborer un cahier de charge à partir des besoins recueillis. Le cahier de charge définit les aspects fonctionnels et techniques du logiciel à développer.
- ⇒ Phase de conception : Dans cette phase on essaie de trouver les solutions et de les modéliser pour servir d'aides dans les développements.
- ⇒ Phase de développement et d'implémentation : Dans cette phase le chantier est lancé. Les modèles de la phase de conception sont développés pour devenir réel.

- ⇒ Phases de test : Les phases de test consistent à vérifier et à tester les résultats obtenus. Une phase de test peut être exécutée après chacune des autres phases du projet. J'ai participé à deux phases de test du projet SERT : la phase de test d'intégration et la phase de test de validation. Ces tests faisaient partie de la validation d'aptitude (VA) du projet. Mais une fois la nouvelle version de SERT en production, il y aura une période de vérification de service régulier (VSR) du projet.
- ⇒ Phases de correction : Les phases de correction consistent à corriger les anomalies trouvées après les phases de test. Une phase de correction peut être exécutée après chacune des phases du test.

4. Réalisation

4.1 Monter en compétence sur le projet SERT

Pendant ma période de stage, j'ai été affecté qu'au projet SERT. C'est un projet qui présente de nombreuses fonctionnalités compliquées avec une architecture très complexe. Ma première mission était de monter en compétence sur le projet SERT. Le but de cette mission était d'essayer d'appréhender SERT sur tous ses aspects. Cette mission est l'une des plus difficiles de mon stage à cause de la complexité du projet.

La première tâche de cette mission a été de se familiariser avec l'aspect fonctionnel de l'application. Pour ce faire, des documents sur le projet ont été mis à ma disposition. L'application a été aussi disponible sur un environnement d'intégration pour effectuer des tests.

La tâche suivante consistait à installer l'environnement de développement sur mon poste et de s'habituer avec les outils utilisés pour le développement du projet. Les procédures d'installation de tous les types d'environnements du projet sont déjà documentées. Certains outils ont été déjà en place et ont nécessité juste des configurations comme les outils d'intégration continue (Jenkins), de gestion de version (SVN) et de contrôle de qualité de projet (Sonar cube).

La dernière tâche de cette mission a été d'étudier les codes sources du projet. Pour cela, il a été nécessaire aussi de comprendre l'architecture technique de l'application. Les documentations des codes ont été d'une grande aide pour cette tâche. J'ai été mené surtout à travailler qu'avec les projets en java.

4.2 Intégration de JavaFX avec Swing

Cette mission consiste à monter en compétence sur les deux bibliothèques graphiques de Java : Swing et JavaFX. Plus précisément, le but de cette mission est l'intégration de JavaFX avec Swing. En effet, pour assurer l'amélioration continue de l'application, nous avons déjà pensé à la migration de Swing vers JavaFX.

Cependant, refaire les interfaces graphiques de l'application totalement en JavaFX ne fait pas partie du cahier de charge de cette nouvelle version de SERT. Seuls les composants graphiques concernés par les évolutions sont à développer avec JavaFX. Pour ce faire, j'ai été mené à faire un « Proof of concept » ou POC pour vérifier la faisabilité de l'intégration de JavaFX dans du Swing. Il était nécessaire d'abord de comprendre comment fonctionne Swing et JavaFX. Après, on pouvait voir comment faire cohabiter et interagir les deux.

4.2.1 Swing

Lorsque vous lancez une application Swing, trois threads sont automatiquement créés. Le premier est le "main application thread" qui se charge d'exécuter la méthode `main()` de l'application. Le deuxième thread est le "toolkit thread" dont le rôle est de recevoir les événements du système d'exploitation, par exemple un clic de souris, et de les transmettre au dernier thread, appelé "event dispatching thread" ou EDT. Ce dernier est extrêmement important car il répartit les événements reçus vers les composants concernés et se charge d'invoquer les méthodes d'affichage.

Le toolkit Swing a été créé en partant du principe que toutes les opérations affectant l'état des composants seraient réalisées dans l'EDT. Swing n'est donc pas une API "thread safe" et ne doit être manipulée que depuis un seul et unique thread, l'EDT. La méconnaissance du modèle de gestion des threads de Swing empêche les développeurs de réaliser des interfaces réactives.

Pour pouvoir créer des interfaces graphiques interactives tout en respectant la règle de fonctionnement sur un modèle de thread unique, L'API de Swing offre une solution sous la forme de trois méthodes de la classe `SwingUtilities` : *`invokeLater()`*, *`isEventDispatchThread()`* et *`invokeAndWait()`*.

4.2.2 JavaFX

JavaFX est la bibliothèque graphique remplaçante de Swing et d'AWT. Elle a pour avantage d'être utilisable via un langage objet et statiquement typé. Comme Swing, JavaFX a aussi un thread événementiel des composants graphiques appelé **JavaFX Application Thread (JAT)**. Par analogie avec Swing, en JavaFX, seule la méthode ***Platform.runLater*** est disponible pour exécuter une méthode dans le JAT.

Voici quelques avantages qu'offre JavaFX par rapport à Swing :

- L'utilisation du FXML : FXML est un langage de balisage, basé sur du XML, qui permet de créer des interfaces graphiques avec JavaFX. Son utilisation permet de séparer la création des interfaces graphiques à l'implémentation de ses logiques.
- JavaFX Scene Builder : C'est un outil graphique qui permet de créer des interfaces graphiques FXML.
- Le support de CSS : On peut utiliser du CSS dans JavaFX comme son utilisation sur du HTML.
- Les animations et le support des médias,
- Support de contenu HTML.

4.2.3 JavaFX/Swing interopérabilité

Nous allons voir comment ajouter des composants graphiques JavaFx dans une application Swing comme SERT. Nous allons voir aussi comment gérer les threads respectifs de JavaFx et Swing dans une seule application.

Pour ajouter un composant JavaFX dans du Swing, le SDK de JavaFX offre la classe **JFXPanel** qui se trouve le package `javafx.embed.swing.JFXPanel`. Voici comment nous avons utilisé cette dans SERT :

```
/**
 * JFXPanel pour une intégration de JavaFX dans SERT
 *
 */
public class SERTJFXPanel extends JFXPanel implements Runnable {

    //.....

    public SERTJFXPanel(String fxmlFile, final Object controller) {
        super();
        //permettre au thread javafx de ne pas se terminer même après la fermeture du dernier panel
        Platform.setImplicitExit(false);
        // chargement de la vue JavaFX depuis le XML
        this.fxmlFile = fxmlFile;
        try (InputStream stream = getClass().getResourceAsStream(fxmlFile)) {
            stream.close();
        } catch (IOException e) {
            throw new IllegalArgumentException("Le fichier " + fxmlFile + " n'est pas dans le classpath", e);
        }
        this.fxmlLoader = new FXMLLoader();
        this.fxmlLoader.setController(controller);
        // initialisation dans le thread JavaFX
        Platform.runLater(this);
    }

    @Override
    public void run() {
        // initialisation
        try {
            fxmlLoader.setLocation(getClass().getResource(fxmlFile));
            Parent rootNode = (Parent) fxmlLoader.load();
            scene = new Scene(rootNode, 400, 200);
            scene.getStylesheets().add("/javafx/css/sert.css");
            setScene(scene);
            init = true;
        } catch (IOException e) {
            LOGGER.error("Le fichier " + fxmlFile + " n'est pas dans le classpath", e);
        }
    }

    //.....
}
```

Figure 13 Classe SERTJFXPanel

Nous avons créé une classe SERTJFXPanel qui hérite de JFXPanel et implémente l'interface Runnable. Cette classe sert à créer des composants JavaFx et de les ajouter dans un conteneur Swing. En effet, par exemple, on peut ajouter un objet de cette classe dans une javax.swing.JPanel. L'implémentation de l'interface Runnable signifie déjà que la classe va être exécutée dans un thread.

Dans le constructeur de la classe, on a deux paramètres :

- String fxmleFile : ce paramètre permet de spécifier le nom du fichier FXML dans lequel est définie la vue de composant graphique JavaFX.
- Object controller : ce paramètre permet de spécifier le controller qui implémente les fonctions logiques de la vue.

Platform.setImplicitExit(false); : Cette instruction permet de ne pas terminer le thread JavaFX même si tous les composants JavaFX sont fermés. Car en effet, par défaut à la fermeture du dernier Panel JavaFX, le thread de JavaFX se termine. Si le thread JavaFX se ferme, on ne peut plus lancer de composant JavaFX.

Platform.runLater(this); : Cette instruction permet de lancer dans le thread JavaFX la construction du composant.

L'implémentation de la méthode de l'interface Runnable contient alors la construction du composant. C'est là que l'on charge le fichier FXML et qu'on l'associe à une « Scene ».

Voici comment on crée un composant JavaFX et l'ajouter à un composant Swing :

```
106 JPanel centerPanel = new JPanel();  
107 AlertePanelController alerteController = new AlertePanelController(messageAlerte);  
108 SERTJFXPanel alertefxPanel = new SERTJFXPanel("/javafx/alertePanel.fxml", alerteController);  
109 centerPanel.add(alertefxPanel);
```

On remarque que l'objet alertefxPanel de la classe SERTJFXPanel se comporte comme un composant Swing. En effet, il peut être ajouté à une JPanel de Swing.

4.3 Développement des évolutions

4.3.1 Structure de base de l'IHM de SERT

On a essayé dans la mesure du possible de séparer le design de l'écran, sa mise en forme et sa logique : fxml, css et contrôleur Java. Cela peut a permis d'implémenter à peu un modèle Modèle-vue-contrôleur (MVC). En effet, c'est l'un des avantages qu'offre JavaFX par rapport à Swing.

Les écrans sont conçus dans SceneBuilder pour les composants proposés par défaut. Ces fichiers fxml contenaient uniquement le layout de l'écran, avec les identificateurs des composants et les événements utilisés nommés (ex : clic sur bouton). Comme en HTML, les composants devant être rendus spécifiquement doivent avoir une classe CSS.

Aussi, Le CSS permet de personnaliser le rendu afin de s'intégrer dans l'application existante. On a essayé dans la mesure du possible de centraliser les styles dans le fichier sert.css.

Enfin, le contrôleur a permis de réaliser les différentes tâches afférentes :

- initialisation et alimentation de la vue,
- récupération d'éléments de la vue (valeur de champ, ...),
- gestion des événements utilisateur.

Les composants/événements sont automatiquement liés par convention de nommage si l'annotation @FXML est présente sur l'attribut/la méthode.

Afin d'intégrer une vue JavaFX dans l'application existante, un composant générique SERTFXPanel a été initialisé. Celui-ci prenait en paramètre un chemin de ressource vers le fichier fxml ainsi qu'une instance du contrôleur qui sera utilisé par cette vue. Un SERTFXPanel est un composant Swing qui peut être ajouté une vue Swing quelconque.

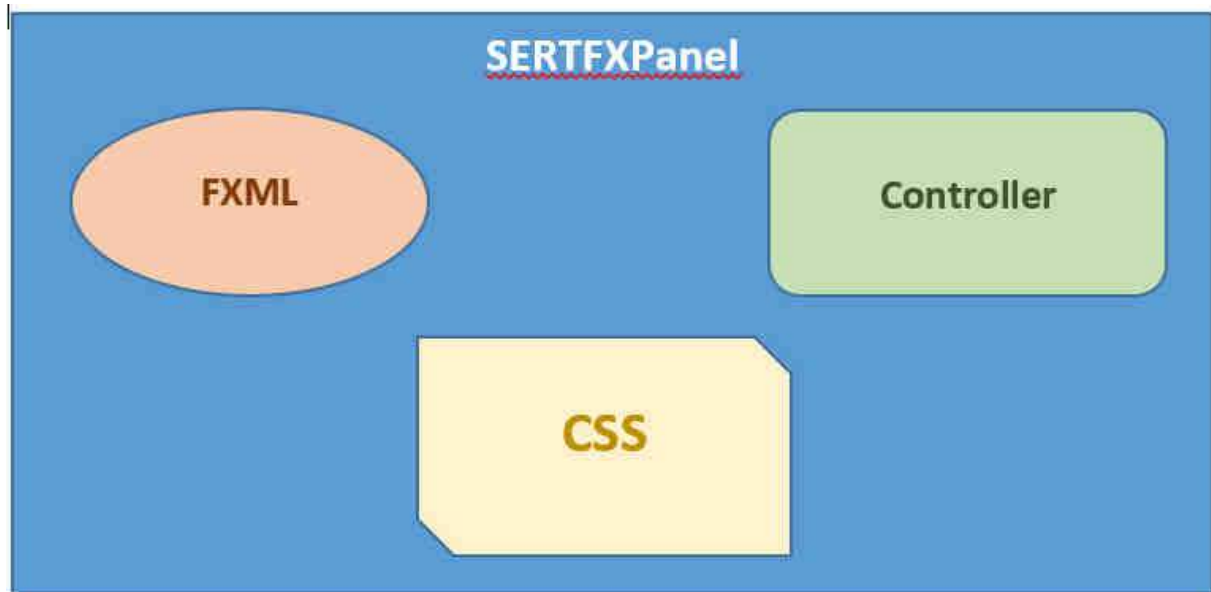


Figure 14 SERTFXPanel

Voici un exemple d'implémentation d'IHM de la nouvelle version de SERT. Il s'agit d'un diagramme de classe modélisant les tableaux listant des projets ou des programmes dans les différents tableaux de bord.

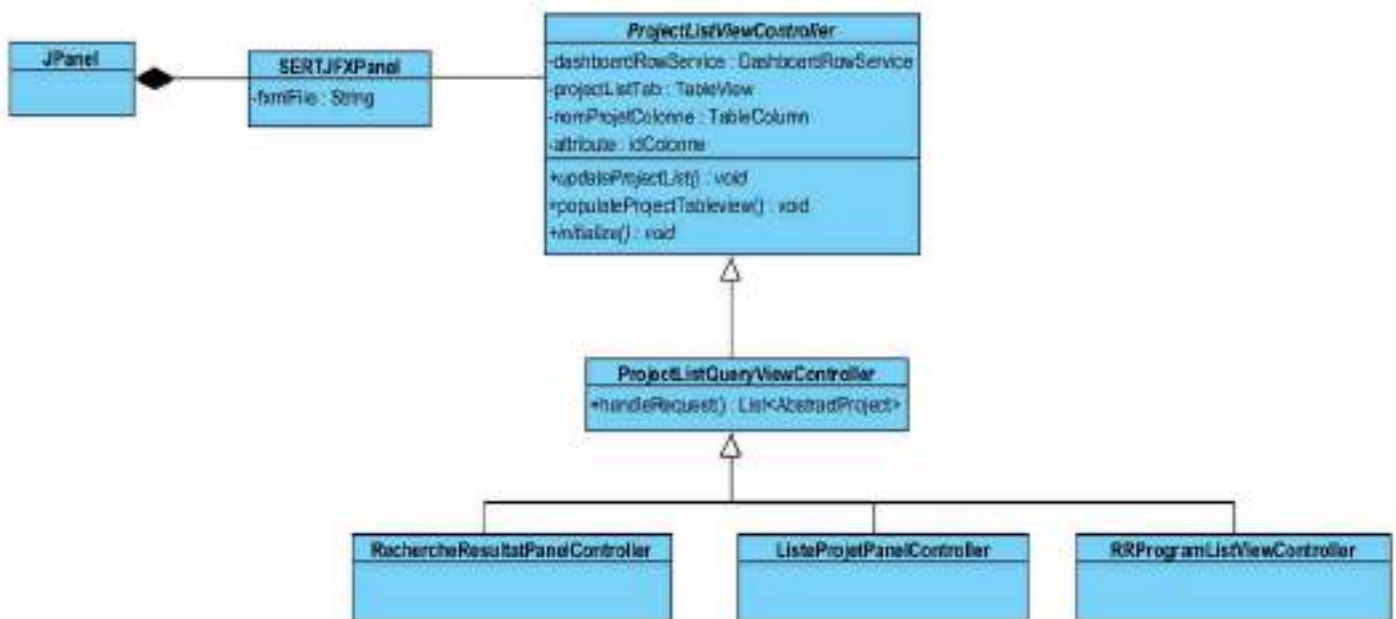


Figure 15 Diagramme de classe tableau de bord

4.3.2 Gestion des ESID

4.3.2.1 Gestion de groupes

Les groupes sont nommés par le trigramme de la ville d'attache de l'ESID (exemple Bordeaux : BDX), sauf DECST qui garde son acronyme complet. Tous les utilisateurs doivent appartenir à un groupe, un utilisateur ne peut appartenir qu'à un seul groupe.

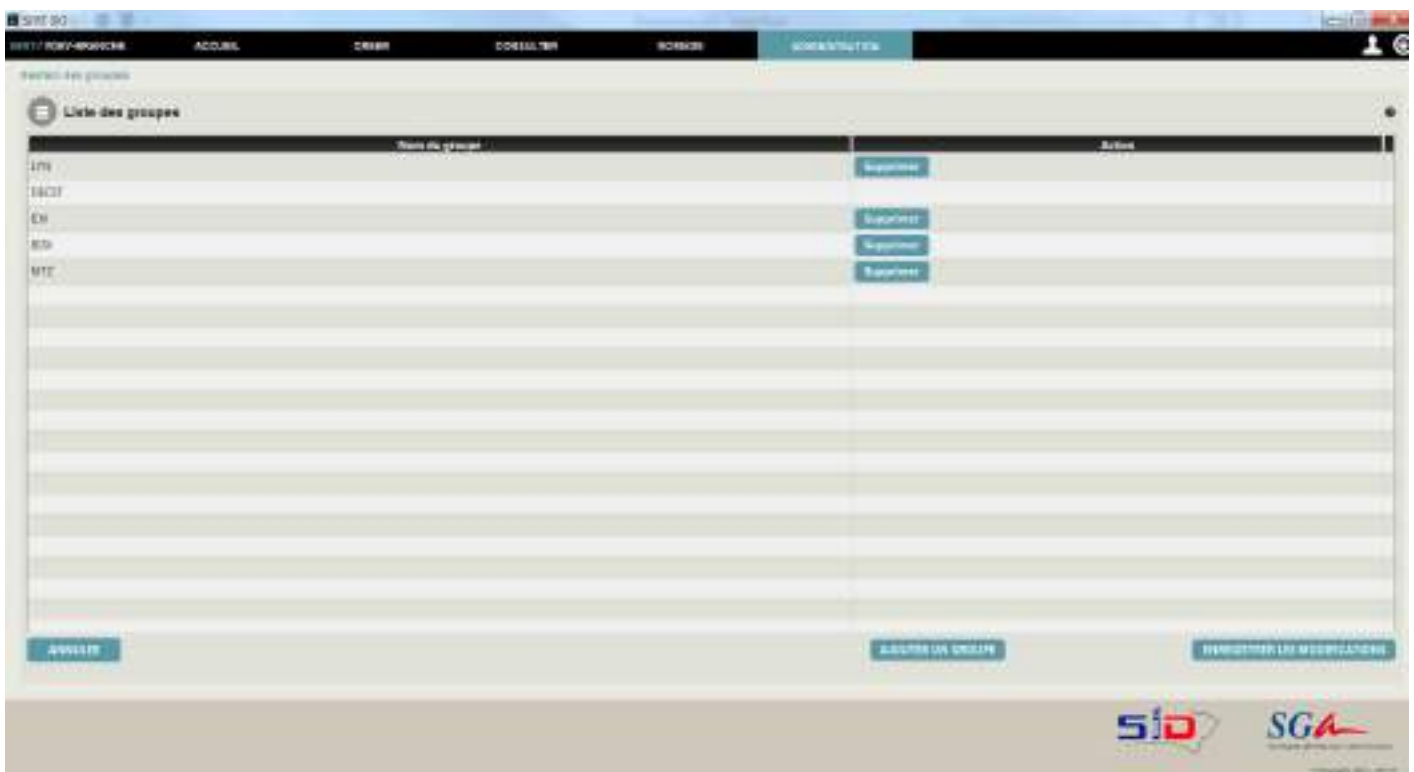


Figure 16 Gestion des groupes

- Une partie basse nommée « Régimes » où apparaissent les régimes faisant l'objet d'une alerte.

➤ Ecran rédacteur de régime (DECST) :

- La partie haute ne change pas.

La partie basse est renommée « programmes en cours de correction » (contiendra les programmes en attente de validation) et disposera d'une nouvelle colonne précisant le groupe ESID de rattachement

4.3.3.1 Tableau de bord « Rédacteur de régime »

Tableau de bord « Rédacteur de régime »

Attention 2 alerte(s) en attente !

MES PROGRAMMES TIR

Zone géographique de référence	Nom du projet	Auteur dernière modification	Date dernière modification	Action
MARNANS (38)	48_JOL_TCGMLR01_CHAMBARAN_replica	tanmandr	14/04/2017	[edit] [delete]
MARNANS (38)	38_RIC_Classique_CHAMBARAN_replica	tanmandr	14/04/2017	[edit] [delete]
MARNANS (38)	38_RIE_TCGMLR01_CHAMBARAN	tanmandr	14/04/2017	[edit] [delete]
MARNANS (38)	38_RE_CHAMBARAN	tanmandr	14/04/2017	[edit] [delete] [share]
MARNANS (38)	38_RIC_Classique_CHAMBARAN	tanmandr	14/04/2017	[edit] [delete]

PROGRAMMES TIR TRANSMIS

Statut	Zone géographique de référence	Nom du projet	Auteur dernière modification	Date dernière modification	Date d'estimation de correction	Action
Refusé	LA COURTINE (23)	23_RIC_Classique_CourtineExplicit	tanmandr	07/04/2017		[edit] [delete]
Refusé	LA COURTINE (23)	23_RT_CourtineExplicit	tanmandr	07/04/2017		[edit] [delete]
Valide avec correction	LA COURTINE (23)	23_RIC_Classique_OND02_04	tschake	03/04/2017		[edit] [delete]
Valide avec correction	LA COURTINE (23)	23_RI_Ten030617	tschake	03/04/2017		[edit] [delete]

SID SGA

Copyright SID - 2017

Figure 18 Tableau de bord du rédacteur de régime

4.3.3.2 Tableau de bord « Rédacteur de programme »

Tableau de bord

Attention 2 alerte(s) en attente !

MES PROGRAMMES TIR

Zone géographique de référence	Nom du projet	Auteur dernière modification	Date dernière modification	Action
MARNANS (38)	48_RJC_Classique_L'indomptable_napoca	tantely	14/04/2017	[Edit] [Delete]
MARNANS (38)	35_RJC_Classique_CHAMBARAN_napoca	tantely	14/04/2017	[Edit] [Delete]
MARNANS (38)	35_RJC_Classique_CHAMBARAN	tantely	14/04/2017	[Edit] [Delete]
MARNANS (38)	35_RJC_Classique_CHAMBARAN	tantely	14/04/2017	[Edit] [Delete]
MARNANS (38)	35_RJC_Classique_CHAMBARAN	tantely	14/04/2017	[Edit] [Delete]

PROGRAMMES TIR TRANSMIS

Statut	Zone géographique de référence	Nom du projet	Auteur dernière modification	Date dernière modification	Date d'estimation de correction	Action
Refusé	LA COURTINE (23)	23_RJC_Classique_CourtineExploit	tantely	07/04/2017		[Edit] [Delete]
Refusé	LA COURTINE (23)	23_RJC_Classique_CourtineExploit	tantely	07/04/2017		[Edit] [Delete]
Valable avec correction	LA COURTINE (23)	23_RJC_Classique_DNOC_D4	tantely	07/04/2017		[Edit] [Delete]
Valable avec correction	LA COURTINE (23)	23_RJC_Classique_DNOC_D4	tantely	07/04/2017		[Edit] [Delete]

Logos: SID, SGA

Figure 19 Tableau de bord du rédacteur de programme

4.3.4 Alerte archivage

Lors d'un changement de statut (archivage pour refus, archivage avec correction, programme en cours de correction ...), une alerte apparaît sur le tableau de bord du profil rédacteur de programme concerné. La suppression de cette alerte se fera de manière manuelle avec une sauvegarde à minima de la date de suppression.

Cette alerte fait également l'objet d'une notification à destination du rédacteur ayant passé le «programme tir» du statut « en cours » au statut « à valider ».

Voici le diagramme de flux de changement statut :

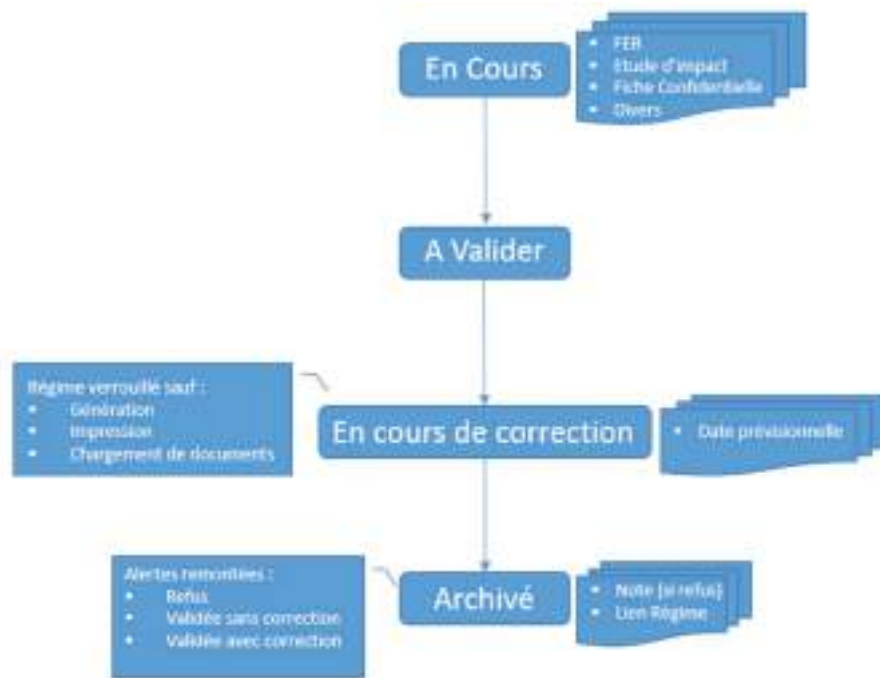


Figure 20 Diagramme de flux changement de statut

Une fois le «programme tir» archivé (validé avec ou sans correction), il sera récupéré par le DECST pour être dupliqué et transformé en régime de tir.

[illegible]

Figure 21 Ecran gestion des alertes

4.3.5 Historique de modification de statut

Lorsqu'un «programme tir» change de statut, l'identification de l'utilisateur, la date et l'heure de modification est enregistrée et consultable.

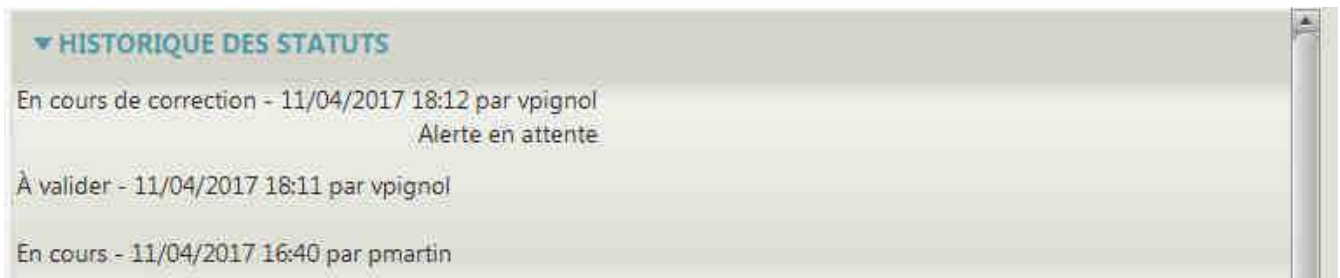


Figure 22 Historique des statuts

4.3.6 Gestion des documents

Cette fonctionnalité doit permettre de pouvoir visualiser, ajouter, et générer les différents documents d'un «programme tir», et de consulter l'historique de changement de statut, ainsi que les utilisateurs ayant opérés le changement.

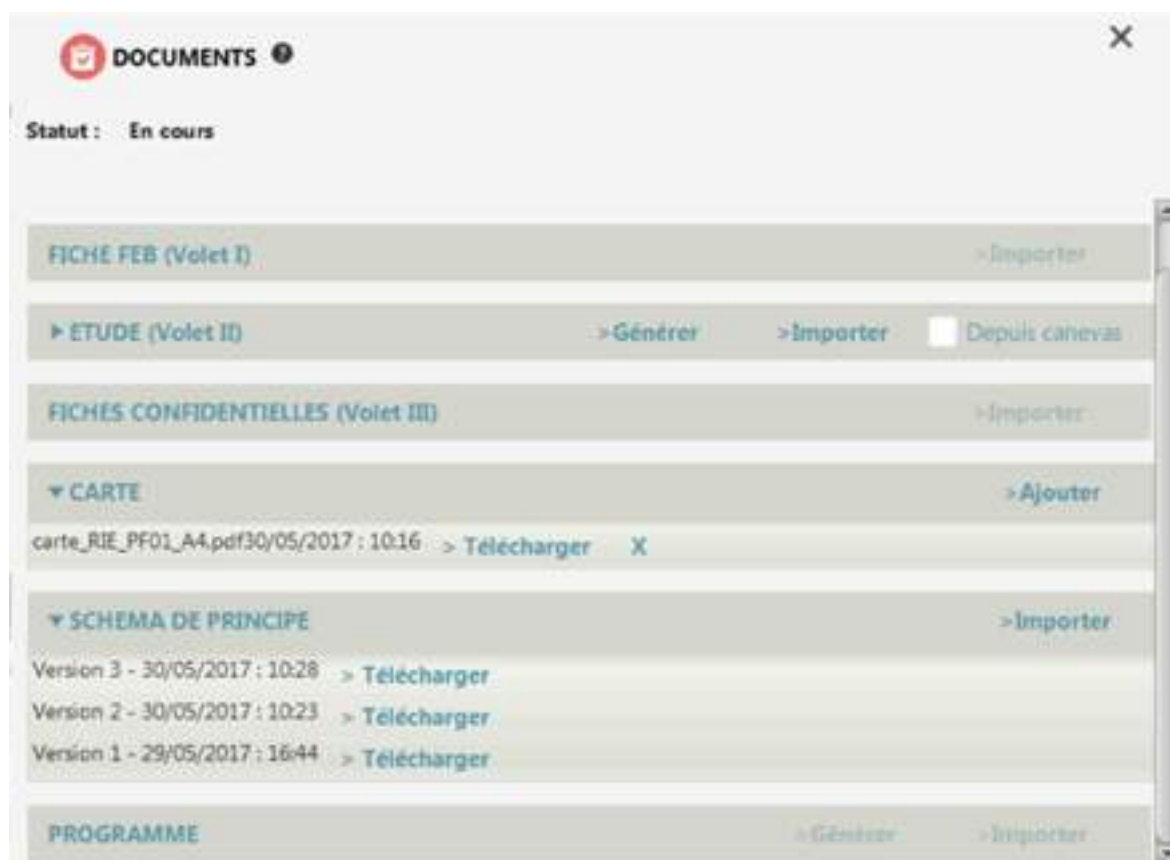


Figure 23 Gestion des documents

4.4 Tests de l'application

Cette mission faisait partie des phases de test de l'application. Elle consiste à créer de nouveaux scénarios de test pour tester les évolutions. Après, ces scénarios seront joués avec les anciens pour tester les nouveaux comportements mais aussi les régressions. Toutes les anomalies trouvées seront notées avec le plus de détails possibles. Ces anomalies seront aussi répertoriées selon leur gravité et leur priorité. En fin, elles seront assignées aux membres de l'équipe.

Il y a eu deux phases de test dans la réalisation du projet. La première phase de test était les tests d'intégration. Cette phase vient après la phase de développement. Une fois la phase de développement terminée, l'application sera déployée sur un environnement d'intégration. L'application sera testée sur cet environnement. La deuxième phase de test était les tests de validation. Cette phase vient après la correction des anomalies de la phase d'intégration. L'application sera déployée sur un environnement de validation pour effectuer ces tests.

Voici comment les phases de test se déroulent. Les anomalies sont remontées dans un premier temps par les testeurs de Capgemini ou le client. Du côté Capgemini, nous avons au préalable écrit un plan de test que nous déroulons à la main lors d'une livraison interne. Tout comme les spécifications, ce plan de test est amené à évoluer au fur et à mesure du projet. Il est saisi dans l'application HP ALM (Application Lifecycle Management), qui permet de suivre le déroulement des tests. Lorsqu'une anomalie est découverte, elle est saisie dans un tracker interne qui est intégré dans un outil de travail collaboratif nommé Collabnet TeamForge. L'anomalie doit être assignée au bon développeur pour qu'il puisse l'analyser puis la corriger, elle doit aussi avoir un scénario précis pour être reproduite, et on doit lui affecter une priorité/gravité pour déterminer l'ordre des corrections à apporter.

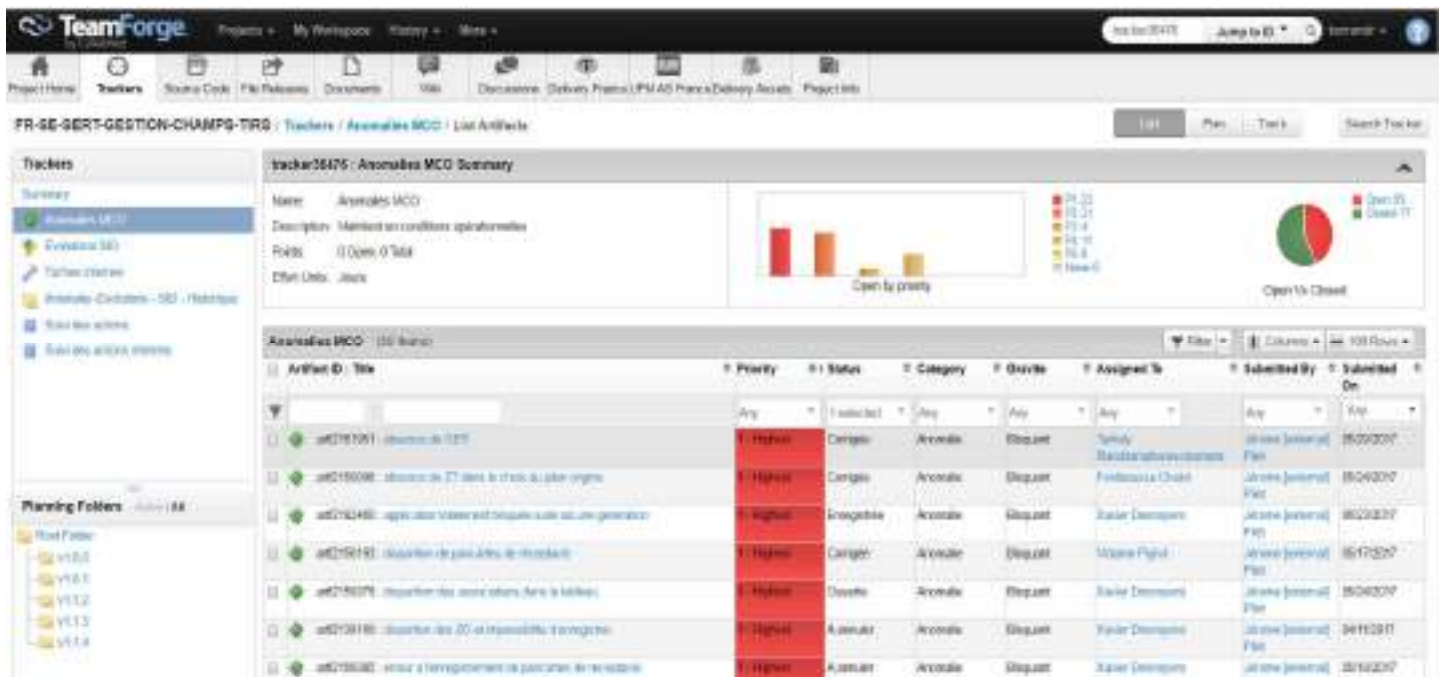


Figure 24 Outil TeamForge

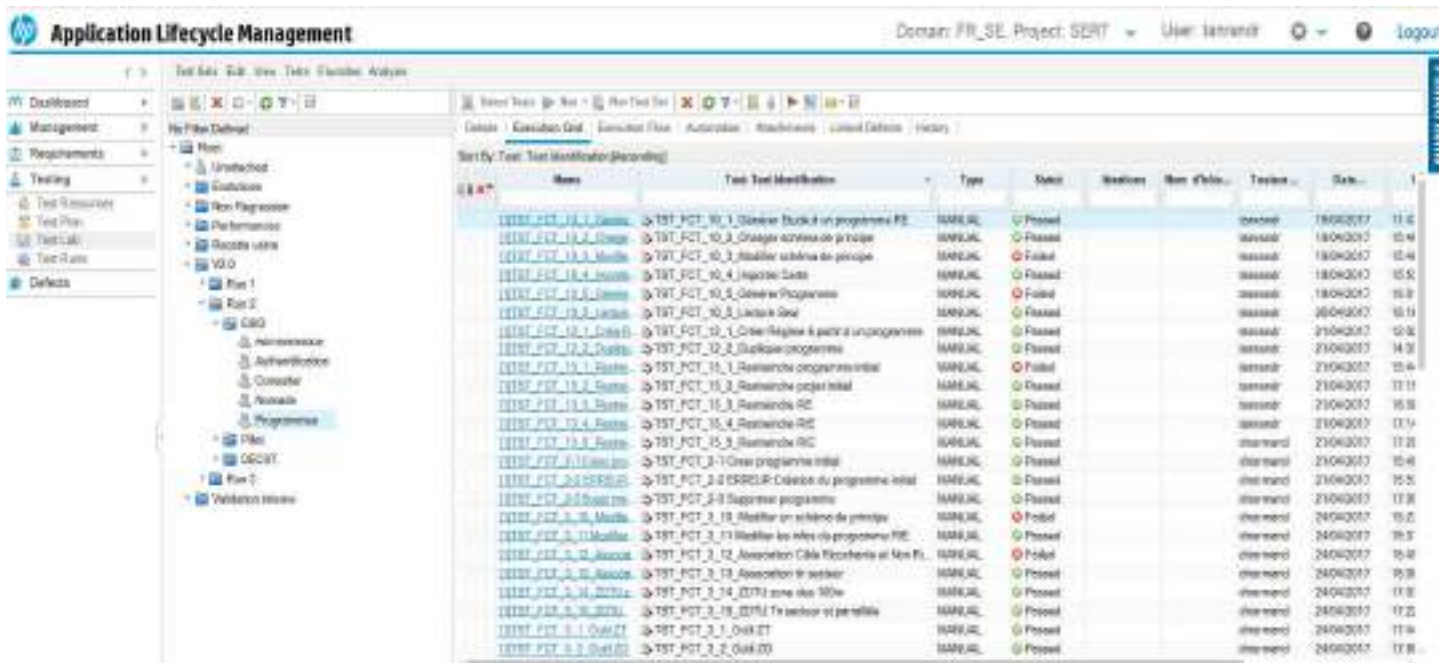


Figure 25 HP Application Lifecycle Management

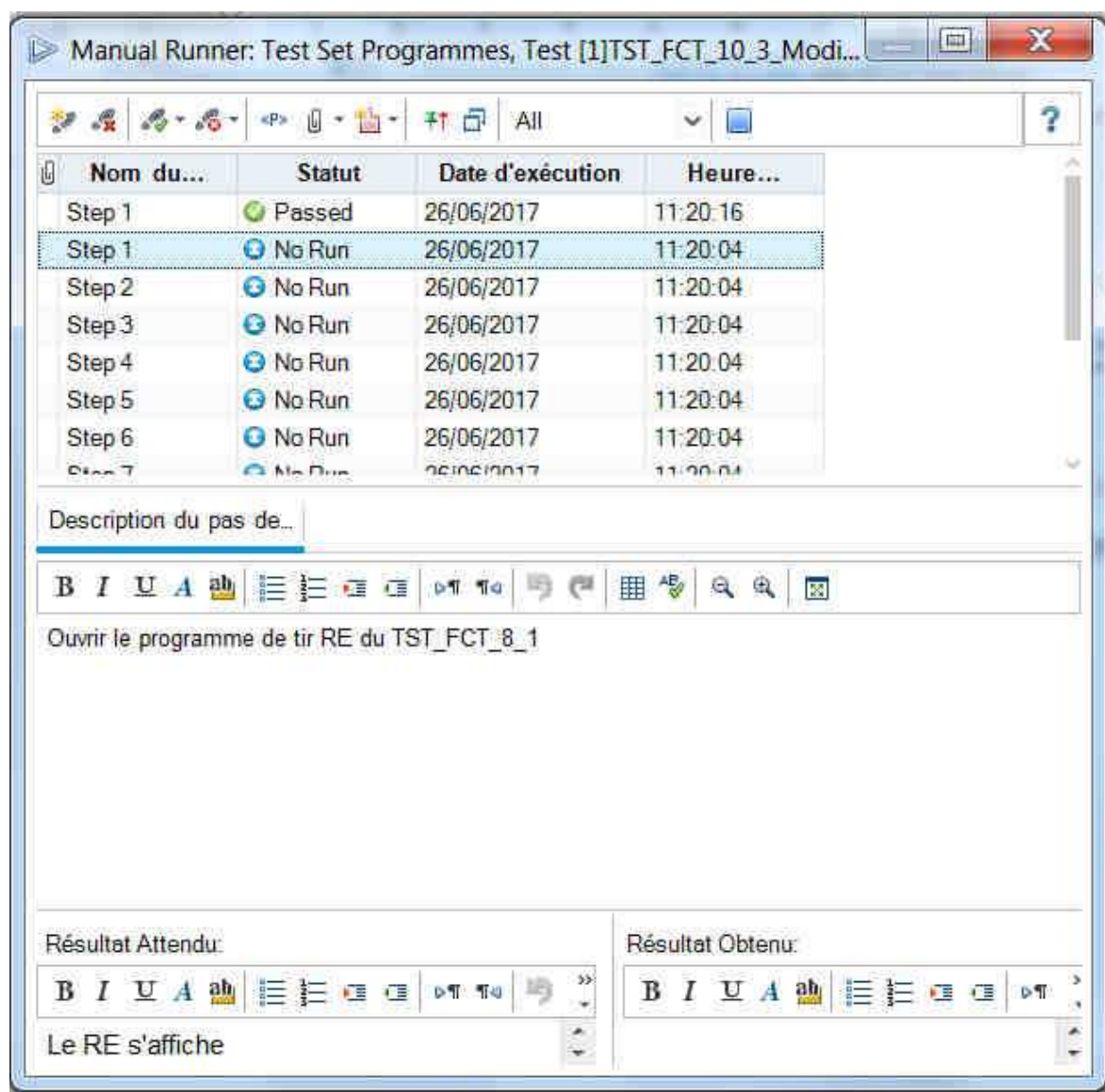


Figure 26 Scenario de test SERT

4.5 Corrections des anomalies

Dans cette partie je vais parler des différentes phases de correctifs et détailler leur déroulement. Nous étions une équipe de 4 personnes à travailler dans les correctifs. Il y avait 3 développeurs et un responsable d'application.

Les corrections des anomalies viennent juste après chaque phase de test. Il y avait trois phases de corrections d'anomalies.

La première était la correction des anomalies trouvées pendant la phase de test d'intégration. Elle a duré 2 semaines.

La deuxième était la correction des anomalies trouvées pendant la phase de test de validation. Celle-là a duré aussi 2 semaines.

La dernière était la correction des anomalies trouvées par le client pendant son test leur plateforme de recette. Cette dernière est celle qui a consommé le plus de temps car nous avons rencontré quelques difficultés. Elle a duré 4 semaines. Les anomalies que nous avons à corriger étaient difficiles à traiter car la plus part n'étaient pas reproduites dans notre environnement de test. Il a fallu quelques interventions directes chez le client pour résoudre les problèmes en vérifiant l'environnement de recette du client.

4.6 Contraintes et problèmes rencontrés

Durant la réalisation du projet, j'ai été confronté à quelques problèmes et difficultés. Il y avait des problèmes d'ordre technique et d'ordre fonctionnel.

La première difficulté, à laquelle j'ai été confronté, a été la complexité du projet SERT. Au niveau fonctionnel, SERT n'était pas facile à appréhender car elle comporte des fonctionnalités complexes. En plus, cela nécessite quelques bases de connaissance sur certaines notions comme le SIG (Système d'Information Géographique). Au niveau technique, malgré l'existence d'un guide, la mise en œuvre de l'environnement de développement de SERT était difficile. En effet, nombreux composants étaient à configurer à ne citer que SDK ArcGIS Runtime, les certificats https et les filtres à bypasser.

Après, une autre contrainte que j'ai rencontrée était le développement en JavaFX avec Java JDK 7. L'utilisation de java 7 était obligatoire car on devait utiliser la même version de java qu'en production. La mise à jour de java en production n'était pas prévue dans ce projet d'évolution car cela nécessitait une procédure compliquée chez le client. Par ailleurs, la bibliothèque graphique JavaFX n'est pas encore complète dans Java 7. En effet, des composants graphiques de base utilisés dans SERT n'étaient pas disponibles encore dans la version de Java qu'on utilisait. Voici quelques exemples de ces composants manquants : les fenêtres de dialogue pour afficher des messages, les aides de saisie dans un champ de type date ou « DatePicker ». Il fallait créer soi-même ces composants pour en disposer. Heureusement, j'ai pu trouver des sources de ces composants en ligne, créées par des auteurs qui ont aussi rencontré le même problème. Il me restait ensuite que de les personnaliser pour couvrir mes besoins et pour correspondre à la charte graphique de SERT.

Le dernier problème que j'ai rencontré pendant mon stage était lors de la phase des correctifs des anomalies recensées par le client. Le client a recensé de nombreuses anomalies qui n'étaient pas reproductibles dans notre environnement d'intégration. Cela a été causé par une différence entre l'environnement de recette du client et notre environnement d'intégration. La source des problèmes était particulièrement difficile à trouver car les anomalies n'étaient pas reproduites. Les pistes de traçage, comme les logs fournis par le client, n'étaient pas évidentes à analyser. Le temps de traitement des nombreuses anomalies a été très couteux au projet. Une intervention chez le client a été alors décidée pour vérifier leur environnement. Il a été alors découvert qu'entre le temps de déploiement de l'application sur son environnement de recette et le début des tests par le client, la sécurité des systèmes d'information du client a été modifiée voir renforcé. Ceci a causé le mal fonctionnement d'Arcgis runtime et a causé les nombreuses anomalies.

5. Conclusion générale

Le présent travail fait l'objet d'un projet de réalisation d'une version majeure d'une application d'édition de champ tir appelée SERT. SERT est utilisé par le ministère de la défense pour la gestion de champ de tir.

En développant sur SERT, je pense avoir apporté ma contribution chez Capgemini. D'un côté, j'ai apporté des nouvelles technologies et des techniques nouvelles surtout dans le développement d'IHM en java. On peut citer par exemple, l'intégration de la bibliothèque graphique JavaFX qui offre plusieurs avantages par rapport à son prédécesseur Swing. D'un autre côté, j'ai énormément participé à la réussite du projet en contribuant dans les phases de test et de correction. Malgré les difficultés rencontrés pendant ces phases du projet, la nouvelle version 2.0 de SERT a été bien livrée et validée par le client.

Animé par ma passion du développement logiciel et par la qualité professionnelle de l'équipe ClusterGIS de Capgemini, j'ai pu acquérir une expérience enrichissante sur divers plans :

- Enrichissement des expériences,
- Approfondissement de mes connaissances techniques,
- Relationnel,
- Créativité.

Ce qui m'a également beaucoup plu, a été la découverte du monde d'une grande société comme Capgemini. J'ai particulièrement apprécié à la fois la grande qualité de l'organisation et le dynamisme et la réactivité au sein de société.

6. Liste des abréviations

BECST : Bureau Expertise Champs et Stands de Tir

ESID : Etablissement du Service d'Infrastructure de la Défense

FEB : Fiche d'Expression du Besoin

IHM : Interactions Homme-Machine

RE : Régime Extérieur

RIC : Régime Intérieur Commun

RIE : Régime Intérieur Elémentaire

SERT : Système d'Edition des régimes de champs de Tir

SID : Service d'Infrastructure de la Défense

SIG : Système d'Information Géographique

7. ANNEXE



SERT

Bienvenue sur SERT / Application V2.0

Identifiant : tanrandr

Mot de passe : ●●●●

Paramètre de connexion : Sédentaire

QUITTER S'IDENTIFIER

Figure 27 Ecran de connexion de SERT



Figure 28 Ecran d'édition d'un RIE





8. Liste des tableaux

Tableau 1 Versions actuelles des outils.....	19
--	----

9. Liste des figures

Figure 1 - Frise chronologique de l'histoire de Capgemini.....	3
Figure 2 - Evolution du chiffre d'affaires	3
Figure 3- Répartition du CA par métier	4
Figure 4- Répartition du CA par secteur	4
Figure 5- Evolution du nombre de collaborateurs	5
Figure 6 Composition d'un régime de tir.....	9
Figure 7 Diagramme de cas d'utilisation SERT 1.1.7	10
Figure 8 Architecture logique de SERT	12
Figure 9 Architecture physique	16
Figure 10 Modèle conceptuel de données.....	17
Figure 11 Diagramme de cas d'utilisation SERT 2.0	20
Figure 12 Démarche cycle en V	23
Figure 13 Classe SERTJFXPanel.....	27
Figure 14 SERTFXPanel	30
Figure 15 Diagramme de classe tableau de bord.....	30
Figure 16 Gestion des groupes.....	31
Figure 17 Gestion des utilisateurs.....	32
Figure 18 Tableau de bord du rédacteur de régime	33
Figure 19 Tableau de bord du rédacteur de programme.....	34
Figure 20 Diagramme de flux changement de statut.....	35
Figure 21 Ecran gestion des alertes.....	35
Figure 22 Historique des statuts.....	36
Figure 23 Gestion des documents.....	36
Figure 24 Outil TeamForge.....	38
Figure 25 HP Application Lifecycle Management.....	38
Figure 26 Scénario de test SERT	39
Figure 27 Ecran de connexion de SERT	44
Figure 28 Ecran d'édition d'un RIE.....	44
Figure 29 Ecran d'édition RIE (détails).....	45
Figure 30 Ecran d'édition d'un RIC	45
Figure 31 Ecran d'édition d'un RE.....	46