



Yann PUGLIESE  
Promotion 2010-2013  
Département Informatique & Gestion



Steria  
1025, rue Henri Becquerel  
Parc Club du millénaire  
34965 Montpellier

# Stage de fin d'étude

Département Informatique & Gestion  
effectué du 04/03/2013 au 31/08/2013

## Développement NTIC

Tuteur : Marco CZARNECKI  
Polytech' Montpellier  
Email :marco@univ-montp2.fr

Chef de projet : Aurélien BOUYAC  
Email :aurelien.bouyac@steria.fr  
Téléphone : +33 4 67 22 76 05

# Sommaire

<b>1</b>	<b>Préambule</b>	<b>2</b>
<b>2</b>	<b>Introduction</b>	<b>3</b>
<b>3</b>	<b>Présentation du contexte</b>	<b>4</b>
3.1	Présentation de l'entreprise . . . . .	4
3.1.1	Le groupe Steria . . . . .	4
3.1.2	Steria à Montpellier . . . . .	6
3.1.3	Les contrats au sein de la SSII . . . . .	7
3.2	Présentation de l'équipe de travail . . . . .	7
3.3	Organisation du travail . . . . .	8
3.4	Présentation de la mission . . . . .	9
3.4.1	Le sujet . . . . .	9
3.4.2	La GED avec EMC Documentum et Prodom . . . . .	9
3.4.3	L'application Prodom . . . . .	13
3.4.4	L'application IO Prodom . . . . .	16
<b>4</b>	<b>Les missions réalisées</b>	<b>19</b>
4.1	L'outil de republication . . . . .	19
4.1.1	Contexte de l'outil par rapport à l'application . . . . .	19
4.1.2	Détails de l'outil . . . . .	22
4.2	La génération des fichiers de configurations d'IO Prodom . . . . .	22
4.2.1	Contexte . . . . .	22
4.2.2	Définition du besoin . . . . .	23
4.2.3	Le développement . . . . .	23
<b>5</b>	<b>Les résultats obtenus</b>	<b>33</b>
<b>6</b>	<b>Conclusion</b>	<b>35</b>
<b>7</b>	<b>Glossaire</b>	<b>36</b>
<b>8</b>	<b>Webographie</b>	<b>37</b>

# Chapitre 1

## Préambule

L'équipe pédagogique de Polytech'Montpellier, les intervenants professionnels responsables de la formation Informatique et Gestion, ainsi que certains collaborateurs de la société Steria m'ont accueilli, aidé, initié, guidé pour mettre en forme mon projet de fin d'étude. Le stage m'a enrichi tant au niveau théorique que pratique social et relationnel. Je les en remercie tous sincèrement.

Plus particulièrement mes remerciements iront à

- Monsieur Fabrice LACLEF directeur de l'agence de Montpellier, pour avoir retenu ma candidature et ainsi me permettre d'effectuer un stage au sein de la société Steria
- Monsieur Sylvain TISSERAND directeur de projet et adjoint de Monsieur Fabrice LACLEF pour son professionnalisme et sa sympathie
- Monsieur Aurélien BOUYAC, chef de projet au sein de Steria, mon tuteur, pour l'accueil et la confiance qu'il m'a accordé dès mon arrivée dans l'entreprise
- Messieurs Christophe BESSON, Marion FELIX, Stephane LEWICKI, Yoann PETIT, Emmanuel TARROU ainsi que l'ensemble du personnel de Steria pour leur accueil sympathique et leur coopération professionnelle tout au long de ces six mois.

## Chapitre 2

# Introduction

Lors de la formation à Polytech'Montpellier en section Informatique & Gestion nous avons l'opportunité de faire un stage de fin d'étude afin d'intégrer le monde de l'entreprise. Mon stage s'est déroulé pendant six mois au sein de la société Steria.

Cette société, créée en 1969 et cotée en bourse à Paris (euronext), emploie plus de 20000 collaborateurs en Europe, Asie et Afrique du Nord. Steria est une SSII réalisant un chiffre d'affaire d'environ 1.76 milliards d'euros par an.

Dans un premier temps, nous prendrons contact avec :

- l'entreprise
- l'équipe de travail
- la mission à réaliser

Nous aborderons ensuite la partie réalisation de la mission avec :

- le déroulement
- la description du travail effectué

Enfin une comparaison des tenants et aboutissants sera réalisée puis nous terminerons par une brève conclusion.

## Chapitre 3

# Présentation du contexte

### 3.1 Présentation de l'entreprise

#### 3.1.1 Le groupe Steria

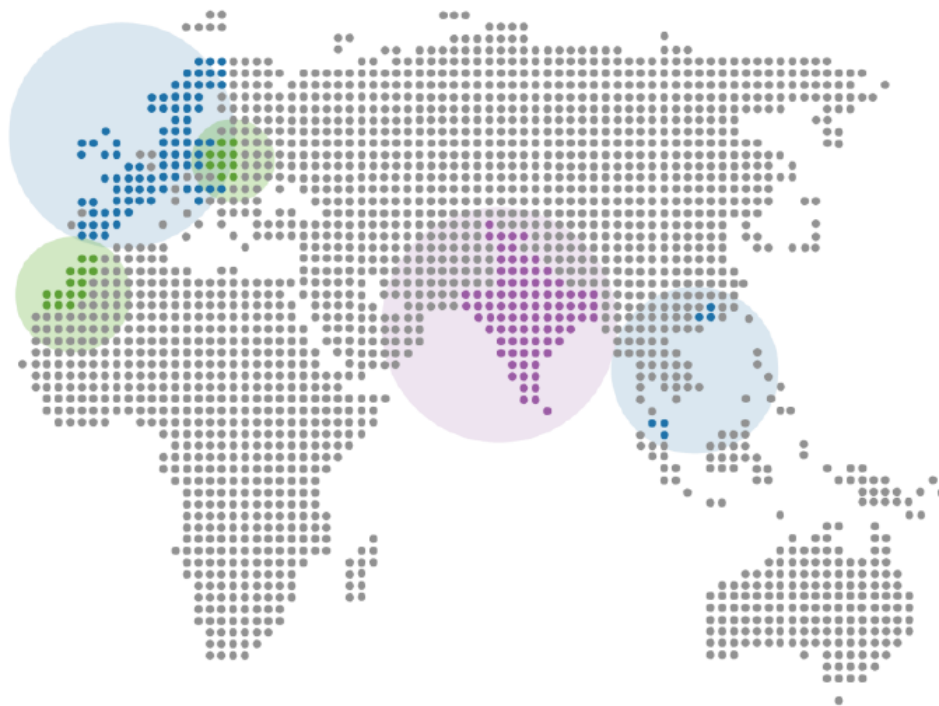
Steria est une SSII (Société de Services en Ingénierie Informatique) spécialisée dans l'intégration de systèmes et l'infogérance. Elle intervient dans les principaux secteurs de l'économie (administrations et institutions publiques, banques et assurances, industries, énergies, transports et télécommunications), et se place aujourd'hui parmi les dix premières sociétés de services informatiques en Europe.

Créée en 1969 par Jean CARTERON sur la base d'un actionnariat interne fort, Steria a su gagner au fil des années de plus en plus de marchés, se créant ainsi de solides références qui en ont fait un des acteurs majeurs de son secteur. C'est ainsi qu'au début des années 80, Steria figure parmi les cinq premières sociétés de services informatiques en France, avec déjà des implantations dans d'autres pays européens frontaliers (Belgique, Suisse). Cette montée en puissance s'est accélérée dans les années 90 par une croissance externe forte, caractérisée par le rachat de plusieurs sociétés européennes, renforçant ainsi l'implantation hors frontières de Steria.

Avec un chiffre d'affaires de 1,75 milliard d'euros en 2011, Steria compte aujourd'hui 20000 collaborateurs répartis dans 16 pays, en Europe mais également en Chine, en Inde et en Afrique du Nord (Maroc et Arabie Saoudite).

Ces pays ont une signification particulière : le Maroc représente l'activité nearshore de Steria, tandis que la présence en Arabie Saoudite découle de la mise en place par Steria de l'intégralité du système d'information de la banque centrale d'Arabie Saoudite en 1985. Ce projet fut un des plus gros contrats passés par Steria. Enfin, l'implantation de Steria à Singapour résulte elle aussi d'un important contrat avec la Direction de l'aviation civile.

Voici une carte du monde représentant l'implantation de la société Steria à travers le monde. Les points colorés représentent l'implantation de Steria.



En France, le groupe Steria est représenté par 15 agences, avec son siège social situé en région parisienne (Meudon-la-forêt, Green Office).

L'immeuble du siège social a la particularité d'être un bâtiment Green Office, c'est à dire que son objectif est de fournir plus d'électricité que ce qu'il n'en consomme. Le résultat obtenu est très positif. Pour Olivier Vallet, Directeur Général de Steria France : "Le bilan de cette première année est un franc succès. Green Office® a tenu ses promesses. Je suis fier d'offrir à nos collaborateurs la possibilité de travailler dans un environnement de très grande qualité, à la pointe de l'innovation et en phase avec les valeurs de notre entreprise."

En outre grâce à ce bâtiment, Steria a remporté le prix "green work place" lors de la 2ème édition des Green Business Awards organisée par BFM Business. Cette manifestation récompense les initiatives exemplaires en matière de croissance verte des services généraux et tertiaires des entreprises que ce soit :

- sur les déplacements professionnels
- sur l'utilisation des TIC (Technologies de l'Information et de la Communication)
- sur la consommation de papier ou la climatisation et le chauffage



### 3.1.2 Steria à Montpellier

Ouverte depuis 1980, l'agence Steria de Montpellier est spécialisée dans l'intégration de systèmes et la gestion électronique de documents .

Elle intervient dans de nombreux secteurs, tels que les banques et assurances, les administrations, la poste et les télécommunications, la santé mais également les industries aéronautiques et maritimes.

À l'heure actuelle, les principaux projets de l'agence concernent :

- La Poste
- La Société générale
- Le ministère des transports
- Le conseil Général
- AMUE SOCODI
- Total.

Total est le client du projet sur lequel j'ai travaillé pendant mon stage.

Steria Montpellier se compose d'un effectif d'environ 60 employés, dont 80 % de cadres de formation supérieure. Les locaux de l'agence n'accueillent cependant qu'une vingtaine d'employés, travaillant sur des projets au forfait, le reste des effectifs est réparti dans les locaux des clients dans le cadre de missions d'assistance technique en régie.

La société possède une forte compétence en développement, tout particulièrement dans les technologies .NET et J2EE. Elle possède également quelques ressources spécialisées dans le paramétrage de progiciels : Maximo, Siebel, SAP.

### 3.1.3 Les contrats au sein de la SSII

Le travail en société de services est divers. Les ingénieurs sont amenés à travailler sur des projets variés, pour des clients spécifiques, ce qui permet la découverte de multiples technologies et de nombreux types d'organisations. Ce mode de fonctionnement s'avère particulièrement formateur. Les salariés peuvent travailler pour les clients de deux manières distinctes : en assistance technique ou au forfait.

#### L'assistance technique

Il s'agit d'un contrat passé entre une société de service et un client, qui engage la société à lui fournir une (ou des) ressource(s) humaine(s) possédant une compétence particulière.

Dans ce mode de fonctionnement, les collaborateurs concernés interviennent sous la responsabilité d'un chef de projet client (ou d'un chef de service client), et le travail s'opère dans ses propres locaux. On parle donc d'engagement de moyens.

#### Le forfait

Il s'agit d'un contrat qui stipule un engagement de résultats et de délais de la part de la société de service, qui devra fournir en temps et en heures l'ensemble des résultats attendus par le client. Le respect des délais, l'organisation et la gestion du projet sont sous la seule responsabilité du prestataire de service.

D'une manière générale, ces projets font réponse à un appel d'offre et se déroulent dans les locaux de la société de service. Parfois, l'équipe est délocalisée chez le client pour des contraintes matérielles, politiques ou encore organisationnelles.

## 3.2 Présentation de l'équipe de travail

L'équipe est composée de 5 personnes :

- Christophe BESSON chef de projet,
- Aurélien BOUYAC chef de projet,
- Stéphane LEWICKI responsable technique,
- Yoann PETIT développeur senior,
- Emmanuel TARROU développeur senior.

L'organisation des bureaux au sein de l'équipe est classique. Le bureau de l'équipe est un "open space" afin de faciliter le dialogue et donc le travail en équipe.

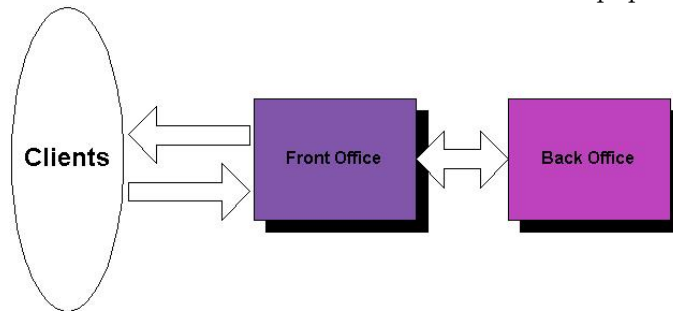


### 3.3 Organisation du travail

L'équipe travaille sur un projet de grande envergure nommé PRODOM. C'est un Gestionnaire Électronique de Documents basé sur Documentum. Documentum est leader du marché sur la GED. Ce projet pour le client, TOTAL, nécessite deux équipes distinctes au sein de Steria.

- L'équipe de Pau : le front office
- L'équipe de Montpellier : le back office

J'ai été intégré dans l'équipe de travail du back office et j'en ai fait le sujet de mon stage. Intéressons nous maintenant aux relations entre les équipes du Front Office et du Back



Office :

La première équipe (FO) a plusieurs missions. Notamment celle de vitrine avec le client pour gérer les retours. Une de ses tâches est de comprendre la demande et de nous la transmettre, si nécessaire, via un logiciel de gestion de projet et de tickets appelé Jira.

J'ai été intégré à l'équipe de Montpellier pour le back office. Lorsque le ticket nous parvient, nous examinons la demande. Deux cas distincts apparaissent lors de cette procédure :

- Soit c'est un souci léger ou un problème de configuration auquel cas nous tentons de le régler rapidement
- Soit c'est une demande d'évolution d'envergure et nous effectuons un devis à transmettre au client.

L'autre mission exécutée par le back Office est le développement des versions futures. Par exemple, lorsque je suis entré dans la société au mois de Mars, l'équipe travaillait au déploiement de la version 3.3, ainsi qu'au développement de la version 3.4 et à la conception de la version 3.5.

J'ai pu donc voir les difficultés pour gérer un planning, répondre aux demandes des clients et développer les versions futures.

## 3.4 Présentation de la mission

### 3.4.1 Le sujet

Le sujet de la mission pourrait être celui indiqué lors de la convention de stage à savoir : Développement NTIC. En effet, l'équipe dans laquelle j'ai été intégré développait un projet de GED (Gestion Electronique de Document) . Ce projet est basé sur un environnement très connu dans ce domaine qui est Documentum fourni par EMC.

L'équipe développait la surcouche du client, ainsi que différentes modifications sur les autres parties pour rendre documentum conforme à leur demande.

En outre, nous avions une activité de TMA ( tierce maintenance applicative). La tâche est donc de répondre aux demandes des clients via le front office afin d'effectuer quelques développements ou ajustements nécessaires.

Mon "sujet" de stage concernant le client Total, ne peut pas véritablement être résumé en une phrase car il y a eu plusieurs mission différentes au sein de l'application Prodom. De plus mon stage se déroule jusqu'au 31 Août et ce présent rapport est rendu pour le 20 Juin. Ainsi l'intégralité de mon stage ne pourra être résumé. Nous allons aborder, dans un premier temps, l'application existante Prodom ainsi que son outil IO Prodom.

### 3.4.2 La GED avec EMC Documentum et Prodom

#### Qu'est ce que Documentum ?

La gestion électronique des documents est devenue un outil très important pour les entreprises. En effet, des gains en qualité et en coût rapide pour les organisations peuvent être réalisés, qu'il s'agisse d'entreprises privées ou d'administrations. A titre d'exemple, le retour sur investissement d'un projet "standard" de dématérialisation de factures est inférieur à un an.

Parmi les ressources les plus précieuses d'une entreprise figurent les divers types d'informations, qui représentent des décennies de connaissances et de capital intellectuel.

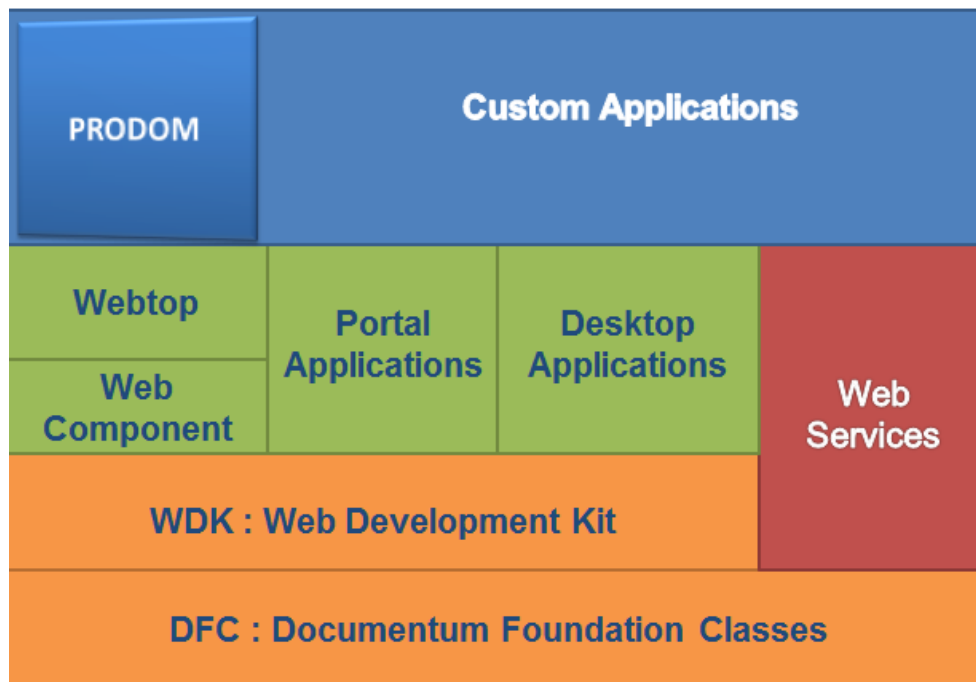
Ces informations notamment, les fiches produits, les documents marketing, les pages Web, les données de service clients, les contrats de fournisseurs, les e-mails, les photographies, etc., que l'on désigne souvent par l'expression de « contenus d'entreprise » représentent un énorme investissement en termes de capital intellectuel, de temps et d'argent.

La façon dont sont gérées, partagées, utilisées et archivées ces informations est un facteur de réussite primordial pour une entreprise. De nos jours, les contenus informatiques augmentent d'environ 200% par an. Ce qui rend cette gestion compliquée et périlleuse d'où l'intérêt d'utiliser des logiciels fiables.

**Documentum** permet de rendre les services décrits précédemment, en se basant sur 5 principaux avantages :

- Conformité : ils permettent d'être en conformité avec les différentes lois, réglementations ou normes
- Efficacité : les processus manuels sont diminués grâce à l'automatisation des procédures
- Productivité : amélioration de l'organisation et de l'accès aux contenus d'entreprise
- Archivage : aide aux besoins en termes de mise en conformité sur le long terme, respecter les contrats de niveau de service et augmenter le taux de réutilisation des contenus
- Consolidation : réduction du coût total de propriété en adoptant une infrastructure unifiée et ainsi bénéficier des économies d'échelle

Ce sont ces points qui permettent de comprendre l'utilité de **Documentum** pour les différentes organisations. Cet outil fourni est composé de différentes couches et propose plusieurs types de clients notamment des clients Web. C'est cette couche que nous modifions afin de rendre le logiciel conforme aux attentes du client et donc à son besoin.

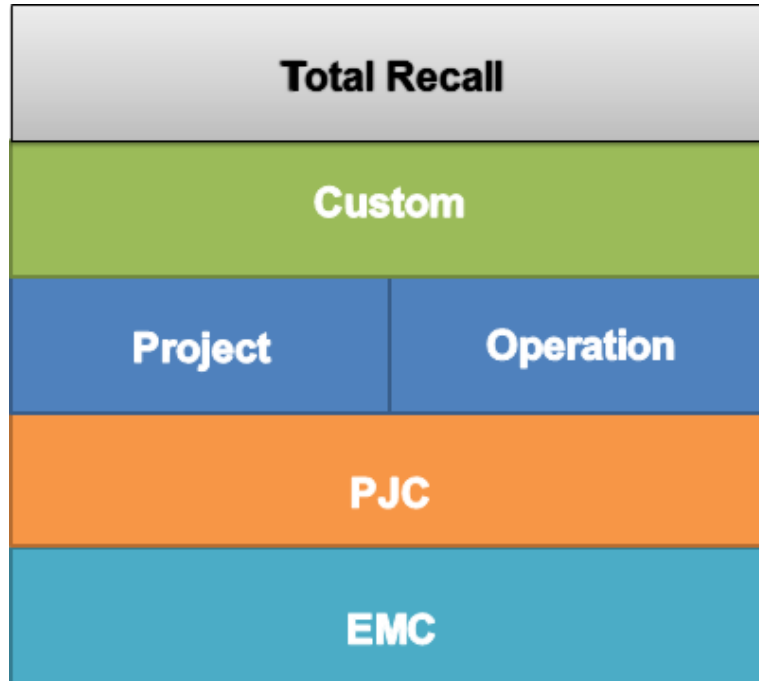


Les différentes couches sont :

- la Documentum Foundation Classes (orange) qui est la base de documentum, c'est cette couche qui attaque directement la base de données.
- le Web Development Kit (orange) contient les bibliothèques permettant de développer pour améliorer l'application.
- le Web Services (rouge) permet la communication et l'échange de données entre applications
- la couche des différents outils liés à documentum (verte) avec le Web top (portail web), le Portal Applications (gestion pour les administrateurs) et le Desktop Application qui est un client lourd
- la couche Custom Applications (bleue) permettant de personnaliser l'application pour la rendre conforme aux besoins du client.

Dans la dernière couche (bleue) se situe notre application Prodom, elle même divisée en cinq couches distinctes que nous pouvons identifier sur le schéma suivant : Les composantes de Prodom.

## Les composantes de Prodom



Nous remarquons dans ce schéma, plusieurs couches distinctes qui composent Prodom. Ce logiciel, développé initialement par EMC, est une véritable application industrielle avec toute la complexité que cela implique. En effet, voici le détail des couches logicielles de Prodom :

- la couche EMC qui a été développée par la société EMC Documentum qui est une couche applicative spécifique à plusieurs projets créés par EMC
- la couche PJC qui est spécifique à Prodom, développée aussi par EMC
- les couches Project et Operation qui sont spécifiques aux deux types de bases de données requises
- la couche Custom permettant aussi de surcharger certains éléments propres aux différentes bases de données
- la couche Total Recall

Il est donc aisé de voir que l'architecture de cette application industrielle est compliquée. Ceci est normal car Prodom doit pouvoir gérer plusieurs centaines de milliers de documents.

### 3.4.3 L'application Prodom

#### Exemple d'utilisation de Prodom

Prodom est donc un logiciel de "Gestion Électronique de Documents" spécialisé pour les bases de Total. Total en a une gestion spécifique. Ses bases, qui existent physiquement, sont situées dans le monde entier. Une base est une unité pétrolière de production et de stockage.

Voici par exemple, la plus grosse base de Total qui est une de celle gérée par l'application Prodom :



Cette unité, Pazflor est longue de 325 mètres pour une largeur de 61 mètres et une hauteur de 32 mètres, elle pèse 120.000 tonnes. Elle est dotée notamment de turbines à gaz qui lui permet de générer une puissance électrique de 120 MW et d'un quartier d'habitation pour 140 personnes, le FPSO pourra traiter jusqu'à 220.000 barils par jour et stocker 1.9 million de barils.

Prodom permet donc de gérer cette base. Avant d'être construite, cette base est un projet. Ainsi elle est créée comme projet dans Prodom par les responsables. Chaque élément de cette base doit être contenu dans la base projet, que ce soit la couleur des interrupteurs, la hauteur de chaque tuyaut, chaque fournisseur, les plans,...

Ainsi on peut imaginer que les bases seront très grandes. De plus pour chaque contrat avec chaque fournisseur, un cycle de vie existe : les ingénieurs sur la base demandent un devis pour telle chose, les fournisseurs répondent et ainsi un va et vient dure jusqu'à ce qu'un accord soit trouvé. Enfin les documents sont dans la phase "published" ce qui signifie que le devis a été accepté.

Une fois que tous les devis sont acceptés, et que la direction enterine le projet, la construction est lancée. Une fois terminée, l'unité est mise à l'eau et donc change de statut, ce n'est plus un "projet" mais une "opération". Donc il faut changer la base du statut projet à opération. Pour ce faire, il faut transférer tous les documents présents dans la base vers une nouvelle base opération. Ceci est réalisé par un logiciel appelé IO Prodom.

Évidemment ce transfert est compliqué car migrer l'intégralité d'une base à une autre peut prendre du temps... Le passage en base opération est nécessaire car d'autres actions sont possibles, notamment les évolutions de matériels (une nouvelle technologie arrive sur le marché) ou pour signaler des problèmes. Les bases de Prodom sont en constantes évolutions jusqu'à la fin de leur utilisation (durée de vie d'environ 30 ans)

## Le fonctionnement de Prodom

Prodom permet donc de gérer les documents relatifs à une base du client. Intéressons nous au fonctionnement de l'application. Pour la connexion, l'utilisateur doit s'identifier (possibilité de SSO : login automatique avec le couple : identifiant / mot passe windows



Une fois le log in effectué, l'application est structurée en deux parties bien distinctes :

- Sur la gauche de l'écran, on peut remarquer un plan de navigation et de classement des documents.
- Sur la droite une partie affichage qui liste les documents contenus dans la catégorie choisie (grâce au volet de gauche).

Ainsi, les documents sont classés suivant différents attributs dont certains sont utilisés pour les ranger et améliorer l'ergonomie de l'application. D'autres attributs sont disponibles sur la liste des fichiers comme par exemple le format du document disponible (ici word, il peut y avoir excel, PDF,...)

L'utilisateur pourra, suivant ses droits, sélectionner le document qui l'intéresse et effectuer des actions sur ce dernier (comme le lire, le modifier,...).

PRODOM V3.2

Search

Advanced Search

Tags Navigation

Navigation Views/ Common Views/ As-Built documents/ AS-Built -> Discipline -> Originator -> Equipment STD/ ACC - ACCOUNTING/ EXT - External Contractor

Equipment Class: ALL

FILEDOCUMENTTOOLSREVIEW

View/My CommentsManage reviewDistribute internalDistribute externalIssueViewEditClose

Items per page: 100Show/Hide icons

Export Search results

		Name	Revision Number	Revision Date	Document Type	Discipline	Equipment Class
		FR-PAU-CST-EXT-000020	1.0	Jun 30, 2012	ACC	ACC	ALL
		FR-PAU-CST-EXT-000024	1.0	Jul 3, 2012	ACC	ACC	ALL
		FR-PAU-CST-EXT-000026	1.0	Jul 10, 2012	ACC	ACC	ALL
		FR-PAU-CST-EXT-000028	1.0	Jul 10, 2012	ACC	ACC	ALL

OPEV3 : INSTOWNIXa

My Home Cabinet

Notifications

Subscriptions

Administration

Doziers

Navigation Views

Common Views

As-Built documents

AS-Built -> Discipline -> Doctype -> Equipment STD

AS-Built -> Discipline -> Originator -> Equipment STD

ACC - ACCOUNTING

EXT - External Contractor

(Not Defined)

FGS - FIRE - GAS

NAB - NOT APPLICABLE

QUA - QA - QC - CERTIFICATION

STR - STRUCTURE

AS-Built -> Discipline -> Sector -> Unit -> Equipment STD

AS-Built -> Discipline -> Unit -> Equipment STD

Correspondence Documents in SM

Previous view

Reviews on Technical Documents in SM

Site Modification Technical Documents

My Long Reviews

My Short Reviews

My Transverse Approval Reviews

My Technical Reviews

Cabinets



### 3.4.4 L'application IO Prodom

Prodom possède un outil, nommé IO Prodom, qui doit permettre d'importer et d'exporter les documents de Prodom. Cet outil doit être optimisé car il permet l'importation de milliers de documents simultanément. Il est en cours de développement au sein de Steria. La tâche est confiée à deux membres de notre équipe.

Le but de cet outil est donc l'importation et l'exportation de documents sous Prodom, ainsi il sera aisé de transférer des bases projets à des bases opérations, par exemple. Car, rappelons le, le cycle de vie classique d'une base est :

- "le stade projet" car ce n'est encore qu'un projet avec tous les devis des différents fournisseurs
- le projet est accepté et devient donc opérationnel : "le stade opération"

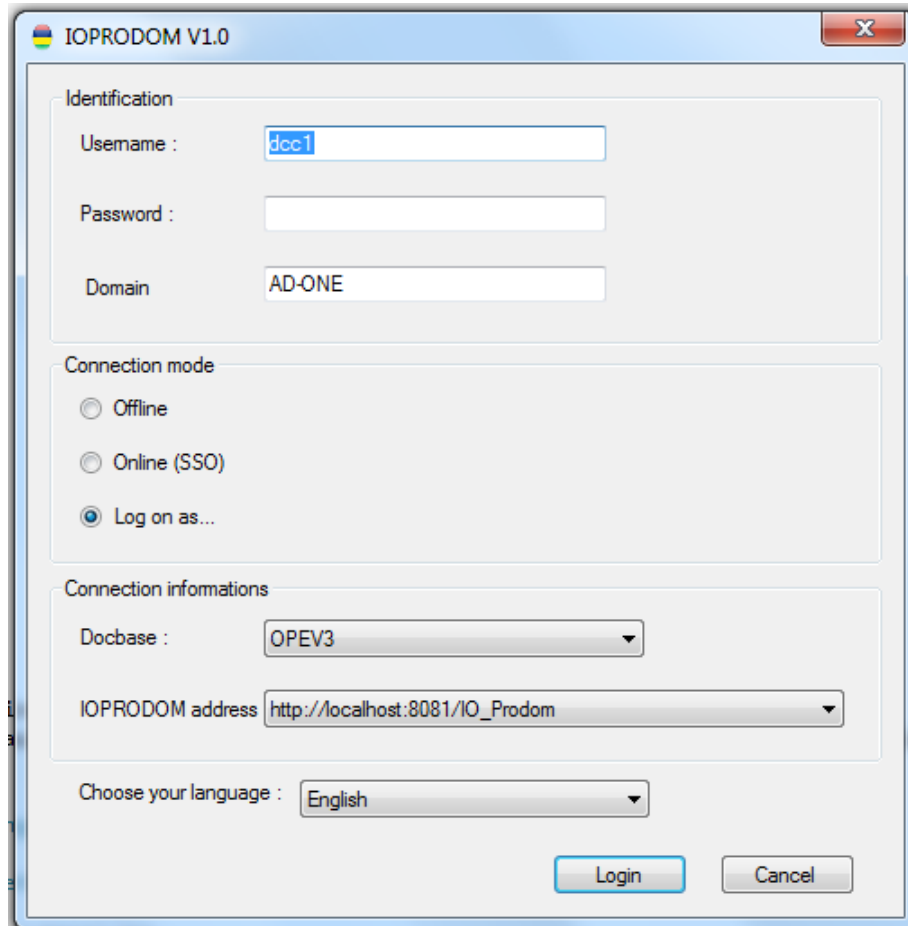
Cet outil se configure différemment suivant les bases de données. Certaines bases nécessitent d'autres informations ainsi, il fallait configurer manuellement tous les éléments afin de rendre IO Prodom compatible avec la base de données.

Ceci est un travail long et fastidieux car ce sont des documents XML à modifier manuellement suivant la base de données.

Pour mieux comprendre la suite de ce rapport voici, dans les pages suivantes, une présentation du logiciel IO Prodom.

## Présentation de IO Prodom

Au lancement du logiciel, une page de log in apparaît avec différentes options



Nous pouvons remarquer que la page se divise en quatre parties distinctes :

- La première partie : le nom d'utilisateur et le mot de passe pour se connecter à l'application. Tous les utilisateurs de Prodom n'ont pas accès à cette application, il faut avoir des droits spécifiques.
- La seconde partie est la gestion de différents log in : le mode classique log in ,le mode hors ligne et le SSO (Single Sign On) qui est effectué grâce aux identifiants windows
- L'adresse de connexion ainsi que le type de base (projet ou opération)
- Le choix de la langue (actuellement disponible en anglais ou en français)

À ce moment là, différents fichiers de configuration (notamment XML) sont nécessaires et diffèrent suivant les bases. Ainsi il faut que l'utilisateur d' IO Prodom dispose des bonnes configurations pour l'utiliser.

Une fois connecté, l'utilisateur arrive sur la page principal de IO Prodom.



Cette page propose plusieurs fonctionnalités :

- La première partie : l'import de documents
  - Nouveau
  - A partir d'un XML
  - A partir d'un XLS (document excel)
  - Consulter les logs
- La seconde partie : l'export de documents
  - Nouveau
  - A partir d'un XML
  - A partir d'un XLS (document excel)
  - Consulter les logs
- L'ouverture d'une tâche au format IO Prodom (.xio)
- L'ouverture du manuel utilisateur
- La configuration de IO Prodom

## Chapitre 4

# Les missions réalisées

### 4.1 L'outil de republication

Abordons maintenant la première mission qui m'a été confiée pour le client **Total**. C'est une mission de développement classique. Elle consiste à ajouter une fonctionnalité à l'application Prodom.

L'intitulé de cette mission : **Total : Améliorer l'outil de republication de l'application Prodom afin de le rendre compatible à une publication en masse.**

#### 4.1.1 Contexte de l'outil par rapport à l'application

Les documents sont stockés dans une base de données. Ces documents possèdent plusieurs attributs initialisés lors de leur création / modification. Les attributs permettent de classer les documents afin d'obtenir des plans de classement et donc de pouvoir naviguer facilement entre tous les documents. (il faut rappeler qu'il y a plusieurs centaines de milliers de documents).

En cas de modification nécessaire sur un ou plusieurs fichiers, il est nécessaire de garantir que le document soit reclassé. En effet certains attributs peuvent nécessiter un reclassement après leur modification.

Prenons un exemple : **une pompe doit être changée, une demande de modification est envoyée au contracteur qui fournira un devis.**

**Lorsque le devis est accepté, il faut modifier le(s) document(s) en rapport avec la pompe. Une pompe possède des branchements spécifiques mais aussi une couleur, des boulons, ....**

Ainsi il est possible que le changement impacte plusieurs centaines de documents. L'outil qui était disponible permettait de réaliser cette republication pour un seul document ainsi ce travail devenait rapidement fastidieux. **C'est pourquoi ma tâche a été de rajouter une fonctionnalité permettant de republier en masse.**

La modification unique se déroule en plusieurs phases grâce à un script prenant en paramètre certains éléments comme une requête DQL (une adaptation du SQL pour documentum).

Le problème est qu'il faut savoir créer une requête DQL et si l'application est utilisée par des non informaticiens cela peut entraîner des erreurs.

En outre, pour chaque fichier la requête doit être personnalisée ce qui est très contraignant. Ce procédé de modification des attributs entraîne une republication du document afin de le classer en fonction de ses nouveaux attributs.

Une autre contrainte était de ne pas altérer le fonctionnement de l'application actuelle : la possibilité de republication manuel doit encore exister.

Certains changements dans le code ont été apporté afin de répondre à ce besoin convenablement.

Les paramètres pour exécuter l'outil sont :

- La requête
- la liste des attributs à modifier ainsi que leur nouvelle valeur (dans un fichier XML)
- Un booléen permettant la modification de toutes les révisions du document ou non (en cas de modification par un utilisateur, une nouvelle révision du document peut être créée, ceci est visible sur la capture d'écran du précédent paragraphe)

Pour rendre l'importation en masse possible, il a fallu ajouter un certains nombres de paramètres.

Deux arguments ont été ajoutés à l'application :

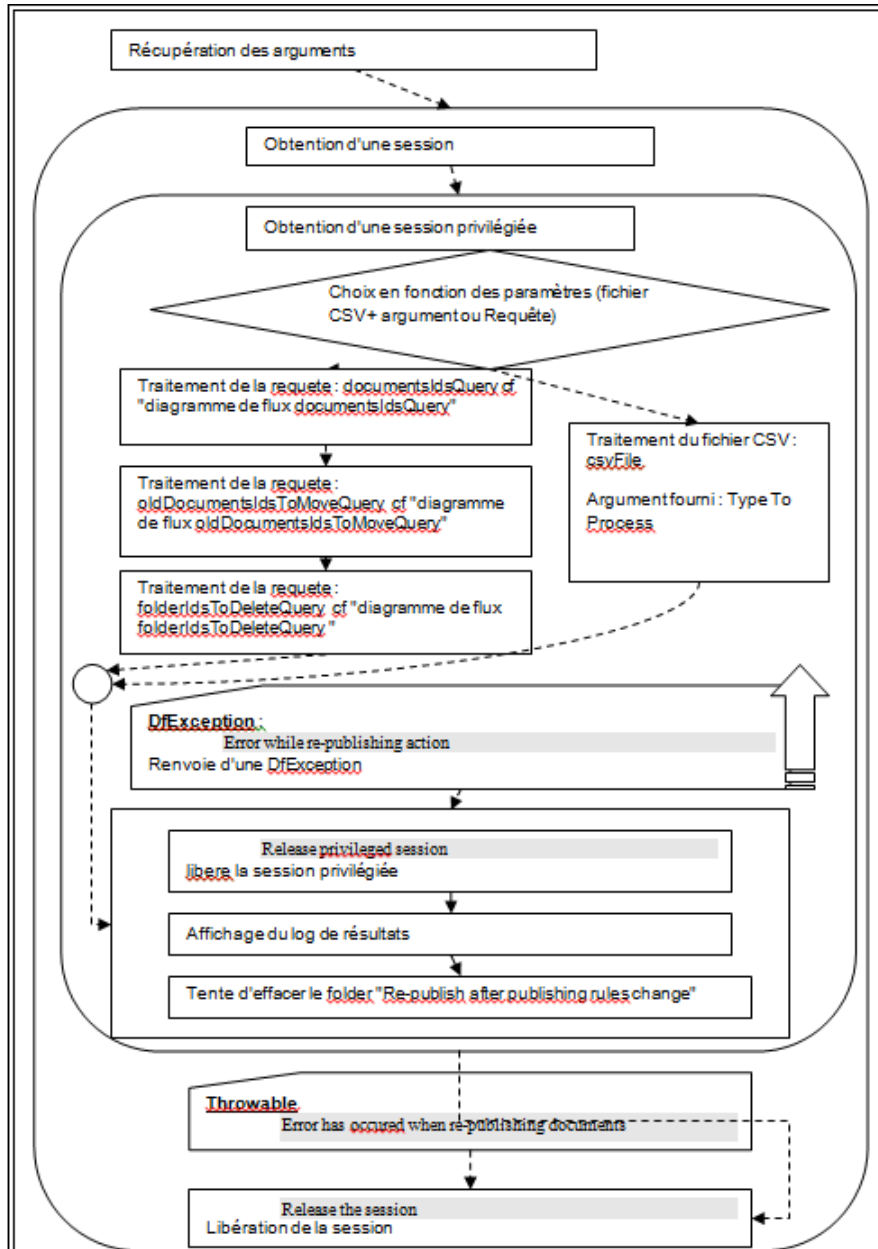
- le type des documents à traiter
- une liste des identifiants des documents

Ils sont différenciés grâce à un identifiant unique en base de données. Avec cet identifiant, on exécute la requête qui nous retourne un document sur lequel il faut effectuer les modifications présentes dans le fichier XML.

Le choix du traitement (exécution de la requête ou republication en masse via le csv) sera effectué en fonction des arguments présents.

Si aucun argument n'est fourni un message d'erreur est retourné à l'utilisateur, si le type de document à traiter ainsi que la liste des identifiants est possible alors on exécute la republication en masse. Sinon on exécute la requête comme avant la modification de l'algorithme.

Voici un schéma simplifié spécifiant le fonctionnement en détail de cette application.



### 4.1.2 Détails de l'outil

Nous remarquons que le programme nécessite des arguments (évoqués précédemment). La suite est l'obtention d'une session en base de données avec les arguments donnés. Il faut une session privilégiée car cette manipulation nécessite des droits administrateurs ainsi il est nécessaire d'obtenir les droits requis.

En cas de problème lors de l'obtention des sessions une exception est levée afin de pouvoir identifier le problème. Ceci aide l'utilisateur, par exemple, à contrôler si les paramètres entrés sont corrects.

Si l'obtention des sessions fonctionne alors un choix est fait en fonction des paramètres présents lors de l'exécution.

- Soit il y aura le traitement manuel
- Soit il y aura un traitement en masse

- Le traitement manuel : le programme récupère la requête puis l'exécute. Le retour de la requête est un Id de document. C'est ce document qui sera modifié en fonction des arguments du fichier `attributeToModify.csv`. Ensuite, il sera publié en fonction des nouveaux attributs.

- Le traitement en masse : Une reproduction en masse est nécessaire. Avec l'identifiant des documents dans le fichier `csvFile`, une reproduction est effectuée pour chaque fichier contenu. Ceci est donc une répétition de traitement manuel améliorée. Elle est améliorée car un autre argument est requis : le type afin de ne modifier que les documents correspondant à ce type. Ceci permet de gagner du temps lors de l'exécution de la requête car en premier lieu, un tri est fait sur les types. Ensuite, le programme recherche l'id correspondant à celui fourni dans le fichier.

Les erreurs sont traitées afin de pouvoir aiguiller l'utilisateur sur les raisons du problème rencontré. C'est ce qui est illustré sur le schéma précédent.

## 4.2 La génération des fichiers de configurations d'IO Prodom

### 4.2.1 Contexte

Dans le cadre de mon projet **Total**, une seconde mission m'a été confiée. Elle concerne un outil de Prodom nommé IO Prodom, qui doit permettre d'importer et d'exporter les documents de Prodom.

Cet outil doit être optimisé car il permet l'importation de milliers de documents simultanément. Il est en cours de développement au sein de Steria. La tâche est confiée à deux membres de l'équipe, Emmanuel Tarrou et Marion Félix.

Le but de cet outil est donc l'importation et l'exportation de documents sous Prodom, ainsi il sera aisé de transférer des milliers de documents d'une base à une autre.

Le développement s'est déroulé en plusieurs étapes :

- Définition du besoin
- Développement
- Tests
- Correction
- Validation

Pour réaliser ce développement nous avons utilisé différentes librairies et bonnes pratiques.

#### 4.2.2 Définition du besoin

Le logiciel doit pouvoir s'installer sous différentes versions de Prodom. Ces versions fonctionnent avec des bases de données différentes...

Ceci crée un problème car IO Prodom se base sur des fichiers de configuration '\*.xio' (XML). Ces fichiers sont construits grâce à la base de données. Ainsi à chaque installation de Prodom sur une base différente les fichiers XIO sont à modifier. Ceci engendre une surcharge de travail non négligeable.

Ma mission consiste à automatiser la création de fichier de configuration suivant les différentes bases de données. C'est un besoin réel car il est délicat de facturer un déploiement exorbitant au client, donc toutes les tâches automatisables doivent être réalisées pour avoir un coût de déploiement minimal.

En effet, pour le client, il est préférable de payer en amont la création de tous ces outils afin de limiter les coûts lors du déploiement (il y a plusieurs dizaines de bases).

La seconde mission est donc de **"créer un outil de génération des fichiers de configuration de IO Prodom"**.

#### 4.2.3 Le développement

Cet outil de génération de ces fichiers n'existait pas nous avons du le créer. Partir seulement de spécifications fonctionnelles entraîne une réflexion sur la conception du programme.

De plus ce programme est appelé à évoluer ainsi il sera repris par un autre développeur. Il faut donc le rendre lisible et compréhensible. Les différents éléments étudiés dans cette petite application sont :

- la connexion à la base de données
- la lecture des éléments de la BD, la création des requêtes automatiques, repeating
- la création des fichiers XML
- la gestion des erreurs
- la création d'une javadoc

#### La connexion à la base de données

La connexion à la base de données se fait aisément, il suffit d'utiliser les librairies existantes. Le programme nécessite d'être autonome ainsi nous avons repris la partie qui nous intéressait de la librairie afin de ne pas faire appel à des fonctions de Prodom.



Les premières informations nécessaires à la connexion à la base de données comme l'adresse IP du serveur (où se trouve la base de données) ainsi que le numéro du port sur lequel il faut se connecter sont stockées dans un fichier de configuration qui se nomme 'dfc.properties'. Ce fichier est nécessaire au lancement de l'application, s'il n'est pas présent, l'exécution retourne une erreur. Voici un extrait de ce fichier :

```
# Set the connection parameters
dfc.docbroker.host[0]=10.18.2.189
dfc.docbroker.port[0]=1489
```

D'autres informations sont nécessaires pour établir la connexion. Il a été décidé d'ajouter un fichier de configuration nommé config.properties qui contiendra tous les éléments de configuration qui seront modifiables.

D'autres éléments sont modifiables et expliqués dans la documentation comme le destDir. C'est le 'destination directory' donc le répertoire de destination où seront enregistrés les différents fichiers créés par l'application. Voici un exemple de ce fichier

```
user=INSTOWNxa
password=total
docbase=OPEV3
types=exp_inspection;exp_maintenance;exp_as_built;fop_management;fop_incoming_corres;
domain=PRODOMXP
destDir=./output
```

Nous remarquons les éléments relatifs à la connexion à la base de données comme le user, le password ou le nom de la docbase ainsi que le domaine.

D'autres éléments sont présents dans ce fichier comme la ligne "types". Cette ligne représente tous les types que nous devons traiter. Elle devait être un paramètre car la liste des types à traiter change en fonction du type de base : opération ou projet. Ainsi l'utilisateur aura IO Prodom configuré pour le type de base qu'il souhaite utiliser.

Le but est que les fichiers de configurations soient générés automatiquement à la connexion. Une des possibilités est de l'intégrer directement à IO Prodom ainsi en fonction de la base choisie les fichiers de configuration seront directement générés lors du lancement du client IO Prodom.

La connexion à la base en java est aisée, il suffit de lancer la commande de connexion avec les informations nécessaires comme la doc base et le login de l'utilisateur. Il ne faut pas oublier de gérer les exceptions afin de ne pas avoir de soucis en cas d'échec de connexion ou d'erreur dans les fichiers de configuration.

```
/**
 * DocbaseConnector is used to create a connection to the db
 *
 * @param usr : name of the user of the database
 * @param pwd : password for this user
 * @param docbase : name of the database
 * @param domain : domain name
 */
public DocbaseConnector (String usr, String pwd, String docbase, String domain) {

    Log.info("Tentative de connexion à la docbase " + docbase + " avec les paramètres suivants USER : " +

    try {
        //Connexion à la docbase
        IDfClient client = DfClient.getLocalClient();
        IDfSessionManager sessionMgr = client.newSessionManager();
        IDfLoginInfo login = new DfLoginInfo();
        login.setUser(usr);
        login.setPassword(pwd);

        sessionMgr.setIdentity(docbase, login);

        session = sessionMgr.getSession(docbase);
        Log.info("connexion docbase : OK ");
    }

    catch (DfIdentityException e) {
        Log.error( "Erreur d'identité lors de la connexion"+e);
    }
    catch (DfException e) {
        Log.error( "Erreur lors de la connexion à la docbase, vérifier les arguments de connexion dans l
    }
    catch (NullPointerException e)
    {
        Log.error ("Problème avec le fichier conf.properties ( le fichier est manquant ou les arguments
    }
}
```

## La création des requêtes automatiques

Une fois connecté à la base le programme doit récupérer les informations nécessaires à la création des fichiers de configurations. Pour cela, trois types distincts apparaissent :

- les valeurs d'attributs non présentes dans la base
- les valeurs d'attributs ayant des valeurs en base
- les valeurs présentes dans d'autres fichiers

Les attributs qui ne sont pas présents dans la base de données sont créés pour être en conformité avec les nécessités de IO Prodom. Ainsi ils ont des valeurs par défaut et ne sont que très rarement modifiés.  
 Les attributs présents dans la base sont assez facilement configurables puisqu'il suffit de lire les informations les concernant. Ceci se fait grâce au code suivant :

```
IDfId idobj = session
    .getIdByQualification("dmi_dd_attr_info where nls_key='en' and state_name is nullstring
        + idfType.getName()
        + "' and attr_name='"
        + idfAttr.getName() + "'");

if (idobj != null) {
    IDfPersistentObject infattr = session.getObject(idobj);
    String condValueAssist = infattr.getString("cond_value_assist");
    labeltxt = infattr.getString("label_text");

    if (!condValueAssist.equals(DEFAULT_VALUE_NULL)) {
        IDfPersistentObject vaAss = session.getObject(new DfId(condValueAssist));
        String default_id = vaAss.getString("default_id");
    }
}
```

Certains attributs spéciaux sont dit "repeating" : multivalués. Ceci signifie que certains attributs comme les auteurs d'un document sont enregistrés dans le même champ. On pourrait imaginer en disant que les différents auteurs sont enregistrés dans une liste. Pour lire ces informations, il suffit d'utiliser une fonction java propre à Documentum :

```
if (!default_id.equals(DEFAULT_VALUE_NULL)) {
    IDfPersistentObject queryAss = session
        .getObject(new DfId(default_id));
    // query!
    queryassist += queryAss.getAllRepeatingStrings("query_string", " ");
}
```

D'autres valeurs, comme le champ "updatable", sont plus compliquées à trouver. Le champ updatable qui exprime par un booléen si la valeur de l'attribut peut être modifié en cas de mise à jour du document ou non. Pour cela, il faut récupérer d'autres documents de Documentum, les traiter et obtenir le résultat, ceci est plus complexe que les éléments précédents et fait donc l'objet d'une fonction spécifique.

Cette fonction se divise en deux parties distinctes :

- La recherche des valeurs dans les deux tables de la base de données
- La comparaison de ces valeurs

Ces valeurs sont contenues dans des champs repeating et avec des formats différents.  
 Voici la première partie qui exécute une recherche dans le code.  
 Dans un premier lieu, nous créons les listes de stockage des éléments :

```
private static HashSet<String> getNonUpdatableAttributes(String typeName,
    IDfSession session) {

    HashSet<String> value = new HashSet<String>();
    HashSet<String> canNotProcess = new HashSet<String>();
    String typeNameSave = typeName;
    IDfCollection collection = null;
```

Ensuite nous recherchons tous les attributs présents dans cette table qui impactent le type passé en paramètre : 'typeName'.  
 Une requête spécifique à documentum sert pour cette opération.

```
try {
    IDfId idobj;
    IDfType type = session.getType(typeName);
    // get values non updatable for an_counter_resolver
    do {
        idobj = session
            .getIdByQualification("an_counter_resolver where object_name='"
                + typeName + "'");
        typeName = type.getSuperType().getName();
    } while (!typeName.equals(null)
        && (idobj.getId().equals(DEFAULT_VALUE_NULL)));
```

Le traitement qui suit est nécessaire car ce sont des attributs repeating, donc il faut les isoler un à un afin de pouvoir les utiliser.

```
if (!idobj.getId().equals(DEFAULT_VALUE_NULL) && idobj != null) {
    IDfPersistentObject infattr = session.getObject(idobj);
    String strRepeating = (infattr.getAllRepeatingStrings("keys",
        ";"));
    String[] tabStrRepeating = strRepeating.split(";");
    value.addAll(Arrays.asList(tabStrRepeating));
}
typeName = typeNameSave;
// END : get values non updatable for an_counter_resolver

// GET VALUES FOR an_publish_resolver
```

Ici se termine le traitement de la première table contenant les attributs non updatable. Le traitement de la seconde table est identique ; ainsi nous obtenons deux listes d'attributs non updatable.

Pour la seconde partie, nous comparerons les valeurs obtenues dans les deux tables et le traitement afin de supprimer les doublons et de ne garder que les valeurs communes dans les deux listes. A la fin de la fonction, la liste 'value' contient tous les attributs dont la mise à jour est non autorisée.

```

List<String> canNotProcessList = formate(canNotProcess);
Log.info("All Attributes lists "
        + canNotProcessList.toString() + "\n" +value.toString());
// on transforme canNotProcess (hash map) en liste pour
// effectuer certaines opérations dessus
canNotProcessList.removeAll(value);
canNotProcessList.removeAll(formate(aList));
// On supprime les infos en double dans les listes

value.addAll(formate(aList));
Log.info("les attributs non traités car il ne sont pas communs à tous les éléments du" +
        "| an_publish_resolver et sont absents du an_counter_resolver : "
        + canNotProcessList.toString());
}

Log.info("Attributes non updatable : " + value.toString());

```

Le traitement de l'attribut updatable est terminé. Nous pouvons donc revenir à notre fonction principale avec la création des fichiers XML en sortie.

## La création des fichiers XML de sortie

Une fois tous les éléments nécessaires à la construction des fichiers récupérés, il a fallu écrire ces fichiers XML, pour cela nous avons fait appel à une librairie java : JDOM. Elle permet assez facilement de créer des fichiers XML conformes à un template en respectant une structure spécifique :

```

// entête xml
Document document = DocumentHelper.createDocument();
Element root = document.addElement("io_config");
Element elem_type = root.addElement("dm_type");
elem_type.setText(idfType.getName());
Element elem_attr = root.addElement("attributes");
HashMap<String, String> map;
HashSet<String> updatableField;

```

Ensuite, il suffit d'ajouter les différents éléments à un élément parent :

```
addElement(attribute, "length", "" + idfAttr.getLength());
addElement(attribute, "format", formatTime);
addElement(attribute, "nonnull", notNull.equals("T") ? "1" : "0");
addElement(attribute, "repeating", "" + (idfAttr.isRepeating() ? 1 : 0));
addElement(attribute, "default_value", "" + defaultval);
```

Ceci est fait grâce à la fonction "addElement()" qui prend en paramètre le père de l'élément, le nom du noeud a créé ainsi que le contenu de l'élément créé :

```
private static Element addElement(Element parentNode, String nodeName,
    String text) {
    Element eElement = parentNode.addElement(nodeName);
    if (text != null)
        eElement.setText(text);
    return eElement;
}
```

En respectant toutes ces spécifications, la création du fichier XML est facilitée. Les seules choses à gérer sont les erreurs. Afin que le document soit lisible facilement on utilise un format de fichier disponible dans la librairie JDOM.

```
// écriture du document dans le fichier .xio
OutputFormat format = OutputFormat.createPrettyPrint();
XMLWriter writer = new XMLWriter(new FileWriter(outputDir
    + File.separator + "io_" + idfType.getName()
    + "_config.xio"), format);
writer.write(document);
writer.close();
Log.info("Fichier créé \n");
```

Le résultat obtenu est donc un fichier XML respectant une certaine forme et ayant certaines données bien spécifiques. L'image qui suit est un extrait de ces fichiers générés.

Extrait des fichiers générés :

```
<attribute>
  <reference>4</reference>
  <category>1</category>
  <label>Document Number</label>
  <attr_name>object_name</attr_name>
  <isrequired>0</isrequired>
  <datatype>2</datatype>
  <length>255</length>
  <format></format>
  <notnull>0</notnull>
  <repeating>0</repeating>
  <default_value></default_value>
  <Value_Assistance></Value_Assistance>
  <va_query></va_query>
  <va_parent></va_parent>
  <pivot_name>exp.techdoc.document.number</pivot_name>
  <updatable>1</updatable>
</attribute>
<attribute>
  <reference>5</reference>
  <category>1</category>
  <label>Affiliate Code</label>
  <attr_name>pjc_project_code</attr_name>
  <isrequired>1</isrequired>
  <datatype>2</datatype>
  <length>64</length>
  <format></format>
  <notnull>0</notnull>
  <repeating>0</repeating>
  <default_value>FOP</default_value>
  <Value_Assistance>exp_as_built_affiliate_code.xio</Value_Assistance>
  <va_query>select object_name,title as attr_display_ from mddm_exp_project
where folder('/Configuration MDDM') order by object_name</va_query>
  <va_parent></va_parent>
  <pivot_name/>
  <updatable>1</updatable>
</attribute>
```

## La gestion des erreurs

Abordons maintenant la gestion des différentes erreurs qui est effectuée au sein de cette même fonction. Cette gestion est faite avec la librairie de gestion des erreurs LOG4j qui permet de gérer les différents types d'erreurs. Nous en utilisons principalement deux :

- Les warning
- Les erreurs bloquantes
- Les warning : Les messages de type "warning" sont présents pour indiquer un comportement suspect mais non bloquant. Ils permettent de signaler l'anomalie à l'utilisateur et au programme de continuer à fonctionner.

```
if (!f.exists()) {  
    Log.info("Aucun template PivotName trouvé pour l'attribut : "+ attrName+" du type : "+ type
```

- Les erreurs bloquantes empêchent le programme de se terminer en affichant un message d'erreur à l'utilisateur.

```
    } catch (DfException e) {  
  
        Log.error("Error executing request : " + e);  
  
    } catch (FileNotFoundException e) {  
        Log.error("file not found " + e);  
  
    } catch (IOException e) {  
  
        Log.error("Input / Output error " + e);  
    }  
}
```

Les exceptions sont préfixées par un message spécifique à savoir les warning commencent par "[INFO]" alors que les erreurs bloquantes le sont par "[ERROR]". Grâce à cette convention, les erreurs sont vite identifiées dans les fichiers ".log" par le développeur. Les fichiers ".log" servent à stocker tous les messages de sortie du programme.



## La javadoc

Comme ce programme va être utilisé par d'autres développeurs une documentation est nécessaire. Pour cela, nous avons opté pour la création d'une javadoc en anglais.

```
/**
 * releaseSession
 *
 * to properly close the session
 *
 * @param dfSession
 */
public static void releaseSession(IDfSession dfSession) {
    IDfSessionManager sMgr = dfSession.getSessionManager();
    sMgr.release(dfSession);
}
```

## Chapitre 5

# Les résultats obtenus

Cette partie est délicate à rédiger : le rapport doit être rendu le 20 Juin 2013 et le stage se termine le 31 Aout 2013. Comment faire un bilan alors qu'il reste encore deux mois et demi de stage ?

Je vais donc :

- Dans un premier temps faire un bilan de tout l'enrichissement personnel que j'ai retiré de ce stage chez Steria
- Puis suivra le bilan professionnel de ces trois mois de formation.
- Ensuite je proposerai un planning pour la fin.

Durant ce stage, j'ai eu l'opportunité de travailler sur un projet de grande envergure pour le client Total. J'ai beaucoup appris sur :

- la mise en place et le suivi d'un projet
- le déroulement du travail en équipe
- le partage des tâches au sein d'un groupe

Je me suis enrichi grâce à l'expérience des autres membres de l'équipe. J'ai progressé dans ma façon de contacter les gens, de les questionner de les écouter, d'essayer de répondre à leur demande.

J'ai acquis certaines compétences qui m'aideront à savoir comment m'intégrer dans un projet en cours de développement et à travailler en collaboration avec d'autres personnes sur un même dossier.

Les deux premières missions pour Total qui m'ont été confiées sont entièrement réalisées.

- La première, l'outil de republication en masse, est en phase de test. Nous attendons les résultats afin de pouvoir estimer la justesse de mon travail.

Les premiers tests, unitaires, ont été concluants puisque l'outil effectuait correctement le travail demandé.

De plus un des membres de l'équipe a supervisé et a relu mon travail avant de l'envoyer en phase de tests.

- Pour la seconde mission, les résultats obtenus sont connus. En effet, l'outil de configuration des fichiers XML a été testé à maintes reprises et sur plusieurs bases de données.

Total possède plusieurs dizaines de bases. Nous avons des copies de plusieurs d'entre elles en local et avons pu déployer l'application qui génère les fichiers.

Chaque base de données possède ses propres spécificités impliquant des fichiers de configuration unique.

Après le lancement de notre application, les fichiers sont créés. Ils permettent de configurer l'application IO Prodom donc nous avons pu les tester.

S'ils ne sont pas conformes, IO Prodom produit des beugs et ne peut fonctionner. Les tests ont été concluants après, évidemment, plusieurs retouches.

Aujourd'hui il suffit d'exécuter notre application afin de pouvoir créer les fichiers de configuration. Par la suite il faut copier ces fichiers dans le répertoire d' IO Prodom.

Enfin il faut lancer IO Prodom et vérifier que le programme est fonctionnel.

Le prochain but est donc d'intégrer l'application à IO Prodom afin que ce soit transparent pour le client (le but est d'enlever l'étape de copie des fichiers dans le répertoire d'IO Prodom et de n'avoir qu'une application à lancer).Ceci sera fait dans les semaines à venir.

Ainsi le résultat définitif concernant cette mission ne peut être, véritablement, révélé qu'après l'intégration de cette application au sein d'IO Prodom.

De même une phase de tests complète sera requise même si les premiers tests ont été concluants.

En l'état actuel des résultats, je pense pouvoir dire que les deux premières missions sont pratiquement opérationnelles.

J'ai débuté une troisième mission "la création d'un script d'automatisation de l'installation d'IO Prodom pour Total".

Elle consiste en la création d'un script ".bat" permettant d'exécuter automatiquement les différentes étapes requises pour installer IO Prodom.

Je pense qu'à cette époque du stage, le travail en cours me permettra de terminer cette troisième mission et peut être en commencer une quatrième?

En ce qui me concerne, j'ai pu intervenir sur une infime partie du développement de Prodom pour le client Total. Ce travail m'a particulièrement intéressé et mon plus grand souhait serait de poursuivre ce développement chez Steria...

## Chapitre 6

# Conclusion

Je dois dire que ce stage fut très enrichissant à tous les niveaux. Tout d'abord j'en dégage une satisfaction personnelle : les résultats obtenus approchent ceux que j'espérais. L'expérience emmagasinée est assez significative au vu des compétences acquises et des qualités développées.

D'autre part les relations humaines entretenues avec mes collègues ont facilité mon intégration au sein de l'entreprise, un environnement qui m'était auparavant peu connu.

Cette expérience va me permettre d'appréhender le futur avec :

- plus d'optimisme
- plus de confiance
- plus de sérénité
- plus de connaissance

Mon séjour chez Steria ne fut pas toujours facile. Mais, j'ai eu la chance d'être bien tutoré et j'ai pu compter sur l'aide de mes collègues qui m'ont apporté leurs expériences et leurs conseils à chaque fois que j'en ai eu besoin.

Pour conclure, je reconnais que :

- grâce aux compétences acquises lors de ma formation à Polytech Montpellier
- grâce à la confiance que Steria m'a accordée en me déléguant entièrement la réalisation de ce projet

je pense avoir atteint mes objectifs en répondant aux attentes de l'entreprise.

Ce stage en entreprise m'a offert une bonne préparation pour mon insertion professionnelle. Il fut pour moi une expérience enrichissante et complète qui conforte mon désir d'exercer mon futur métier d'ingénieur d'étude et de développement dans le secteur de l'informatique.

## Chapitre 7

# Glossaire

**SGBD** est un logiciel système destiné à stocker et à partager des informations dans une base de données, en garantissant la qualité, la pérennité et la confidentialité des informations, tout en cachant la complexité des opérations.

**SQL** (sigle de Structured Query Language) est un langage informatique normalisé servant à effectuer des opérations sur des bases de données

**DQL** (sigle de Documentu Query Language) est le langage utilisé pour effectuer des requêtes avec Documentum.

**EMC Documentum** La plateforme EMC Documentum fournit une infrastructure client dotée d'un framework et d'outils permettant aux utilisateurs de traiter et d'exploiter les fonctions de gestion de contenu sur diverses applications résidant sur leur poste de travail.

**GED** (sigle de Gestion Électronique de Documents) désigne un procédé informatisé visant à organiser et gérer des informations et des documents électroniques au sein d'une organisation. Le terme GED désigne également les logiciels permettant la gestion de ces contenus documentaires.

**TMA** (sigle de Tierce Maintenance Applicative) est la maintenance appliquée à un logiciel ( « applicative » ) et assurée par un prestataire externe dans le domaine des technologies de l'information et de la communication.

**NTIC** (sigle de Nouvelle Technologie de l'Information et de la Communication) regroupent les techniques utilisées dans le traitement et la transmission des informations, principalement de l'informatique, de l'Internet et des télécommunications.

**XML** (eXtensible Markup Language) est un langage informatique de balisage générique.

## Chapitre 8

# Webographie

Steria <http://www.steria.com>

EMC Documentum <http://www.emc.com/domains/documentum/index.htm>

Total <http://www.total.com>

Wikipédia <http://www.wikipedia.org>

Javadoc <http://docs.oracle.com/javase/6/docs/api/>

## Résumé

Dans le cadre du stage de fin d'études clôturant ma formation en Informatique et Gestion à Polytech'Montpellier, j'ai intégré la SSII Steria pendant une durée de six mois. J'ai participé activement au projet Prodom, destiné au client Total. Ce projet a pour but de gérer les documents internes aux bases pétrolières du client. L'équipe de projet est en charge de développer deux applications Prodom et IO Prodom afin de répondre au besoin du client.

Sous la direction d'un chef de projet de Steria et avec l'appui de l'équipe d'ingénieurs travaillant sur la partie développement j'ai donc été chargé de participer au développement de l'application.

A présent, l'application est toujours en cours de développement, pour encore un certain temps puisque la livraison de la dernière version est prévu pour la fin de l'année.

Par la suite je serais affecté à d'autres missions afin d'améliorer l'application et la rendre conforme à la demande du client.

GESTION ELECTRONIQUE DE DOCUMENTS - JAVA - DOCUMENTUM

## Abstract

I was hired as a trainee by Steria, a software engineering company, in order to team up on a project entitled Prodom for the company Total. The goal was to complete my final internship, leading to the end of my studies at Polytech'Montpellier Engineering School.

The aim of this project is to provide an efficient solution to manage relevant informations, for offshore oil development sites, contained in their internal information system, in order to answer TOTAL's specific needs. The team I joined is in charge of drafting specifications, designing and implementing features for these projects called PRODOM and IO PRODOM.

The development will continue until the final delivery planned for the end of the year. During the end of my training, I will work on several tasks to improve the application.

ELECTRONIC DOCUMENT MANAGEMENT SYSTEM - JAVA - DOCUMENTUM