

Liste des figures

Figure 1:Organigramme de G-FIT	7
Figure 2: architecture technico-fonctionnelle de koala-bpm	8
Figure 3:Architecture n-tiers d'OFBiz	13
Figure 4:Architecture d'OFBiz.....	14
Figure 5:Architecture détaillée d'OFBiz	15
Figure 6:Exemple de définition d'une entité	16
Figure 7:exemple de définition d'un service	17
Figure 8:Traitement d'une requête utilisateur par OFBiz	18
Figure 9:conversion d'un type SGBD vers un type SQL générique.....	20
Figure 10:diagramme des cas d'utilisations.....	24
Figure 11:Diagramme de classe	25
Figure 12:Arborescence du module	31
Figure 13:interface d'authentification	32
Figure 14:Page d'accueil de l'application	33
Figure 15: Formulaire d'ajout d'un nouvel écran»	33
Figure 16:interface de parametrage de l'écran.....	33
Figure 17:interface de gestion des écrans	34
Figure 18:interface d'interrogation et de reporting.....	35
Figure 19:rapport Excel.....	37
Figure 20:rapport PDF	37

Remerciements	1
Introduction	2
Liste des figures	3
Chapitre 1 : Présentation du projet.....	6
1 Organisme d'accueil.....	6
1.1 Le groupe G-fit	6
1.2 La valeur ajoutée de G-fit	6
1.3 Engagements de G-fit	7
1.4 Organigramme	7
2 Description du projet.....	8
2.1 La solution Koala-BPM	8
2.2 Architecture technico-fonctionnelle de Koala-BPM	8
2.3 Problématique	9
2.4 Solution.....	10
Chapitre 2 : Généralités sur les ERP	11
1 Définition	11
2 Avantages.....	11
3 Inconvénients	11
4 La Business Intelligence dans l'ERP	12
5 Les ERP libres.....	12
6 L'ERP libre OFBiz.....	12
6.1 Présentation de OFBiz	12
6.2 Architecture de OFBiz	13
6.3 Le Framework.....	16
6.3.1 L'Entity Engine	16
6.3.2 Le Service Engine.....	17
6.3.3 Le ControlServlet	17
6.3.4 Traitement d'une requête utilisateur.....	18
Chapitre 3 : Analyse et conception	20
1 Etude des besoins	20
2 Modélisation du projet	24
2.1 Choix de l'outil d'analyse	24
2.2 Le diagramme de cas d'utilisation.....	24
2.3 Le diagramme de classe	24

2.4	Les classes techniques	25
Chapitre 4 :	Réalisation	28
1	Outils utilisé	28
1.1	SVN	28
1.2	ANT	28
1.3	FreeMarker.....	29
1.4	PostgreSQL.....	30
1.5	SpringSource Tool Suite	30
2	Aperçue sur le travail réalisé.....	31
2.1	Arborescence du module développé	31
2.2	Présentation des interfaces.....	32
Annexe		40
Glossaire		40

Chapitre 1 : Présentation du projet

1 Organisme d'accueil

1.1 Le groupe G-fit

Créé en mai 1998, G-FIT est un groupe spécialisé en ingénierie des systèmes d'information qui réunit plus de 200 collaborateurs.

Les principaux secteurs d'activités de G-fit se focalisent essentiellement sur la Régie, le Forfait, la Tierce Maintenance Applicative, la Formation et l'Embauche ou la Pré-embauche pour le compte de ses clients grands comptes.

L'expertise de G-fit va de la direction de projets (migration technique, refonte, EURO...) à la mise en œuvre de Tierce Maintenances Applicatives et de prestations forfaitaires, en passant par l'ingénierie et l'Assistance à Maîtrise d'Ouvrage, la conception et migration de systèmes d'information, l'évaluation de la charge et la planification de projets, la mise en place de Plan d'Assurance Qualité.

G-Fit Maroc, la filiale marocaine du groupe est à la fois la plateforme pour développer des projets en mode offshore mais aussi le représentant du groupe sur le continent africain. Une présence qui traduit la volonté d'offrir à ses clients et partenaires un service de qualité mais aussi de proximité. De plus, ses références client sont le gage d'une activité dans la plupart des grands secteurs d'activité qui sont la banque, la monétique, le crédit, l'assurance, la grande distribution, la logistique, la retraite, l'industrie, les entreprises publiques et parapubliques.

1.2 La valeur ajoutée de G-fit

Confier tout ou une partie de la réalisation d'un projet en forfait à une société de service en informatique nécessite de s'assurer des compétences de ladite société. Composée d'hommes et de femmes expérimentés, G-FIT dispose d'un savoir-faire spécifique en matière de prestation de type forfaitaire.

Une prestation au forfait se définit par la mise en œuvre d'un projet décrit et spécifiable dont la maîtrise d'œuvre est confiée contractuellement à G-FIT avec un engagement de sa part sur:

- Les Résultats,
- Les Moyens,
- Les Coûts,
- Les Délais.

Pour ce faire, G-FIT réalise en amont un modèle d'évaluation de faisabilité du projet et spécifie dès le début les engagements des contractants.

Pour la bonne mise en œuvre des projets, G-FIT applique une démarche méthodologique et gère un plan de communication rigoureux avec le client, dont le Directeur de projet G-FIT est l'interlocuteur opérationnel privilégié et qui assure la responsabilité décisionnelle et contractuelle du projet. Il supervise également la mise en œuvre du Plan Management Projet (PMP) et valide « les livrables » (planning prévisionnel des livraisons).

Le Chef de Projet G-FIT intervient sur la mise en œuvre du projet, planifie et valide l'état d'avancement du projet.

Il encadre également l'équipe de réalisation (Analystes Programmeurs, Experts, Ingénieur d'Etude,...) qui assure le développement et les tests du projet.

1.3 Engagements de G-fit

Respect des normes, contraintes et principes d'élaboration : G-FIT dispose via le Plan Management Projet et notamment de sa procédure relative au Plan de conduite du projet d'une organisation permettant de respecter les normes, les contraintes et les principes d'élaboration de la solution proposée. De cette façon G-FIT maîtrise et vérifie les développements et par la même, s'assure que les exigences spécifiées sont bien satisfaites.

Performances : G-FIT respecte les règles de l'art de la profession en termes de développement ainsi que les contraintes liées aux normes et standards (Ergonomie, design, Normes de développements...).

Charges et délais : G-FIT s'engage au respect des charges et délais définis et dispose pour cela des instances de suivi et des procédures définies dans le plan qualité projet.

Démarche certification CMMI : G-fit a adapté une approche intégrée de gestion se basant sur la **norme CMMI** afin d'améliorer la gestion et le contrôle de ces processus de développement et de maintenance quel que soit la complexité technologique qu'elle soit fonctionnelle ou organisationnelle pour continuer à fournir de meilleurs produits et de meilleurs services, dans les meilleurs délais et à un meilleur prix.

1.4 Organigramme

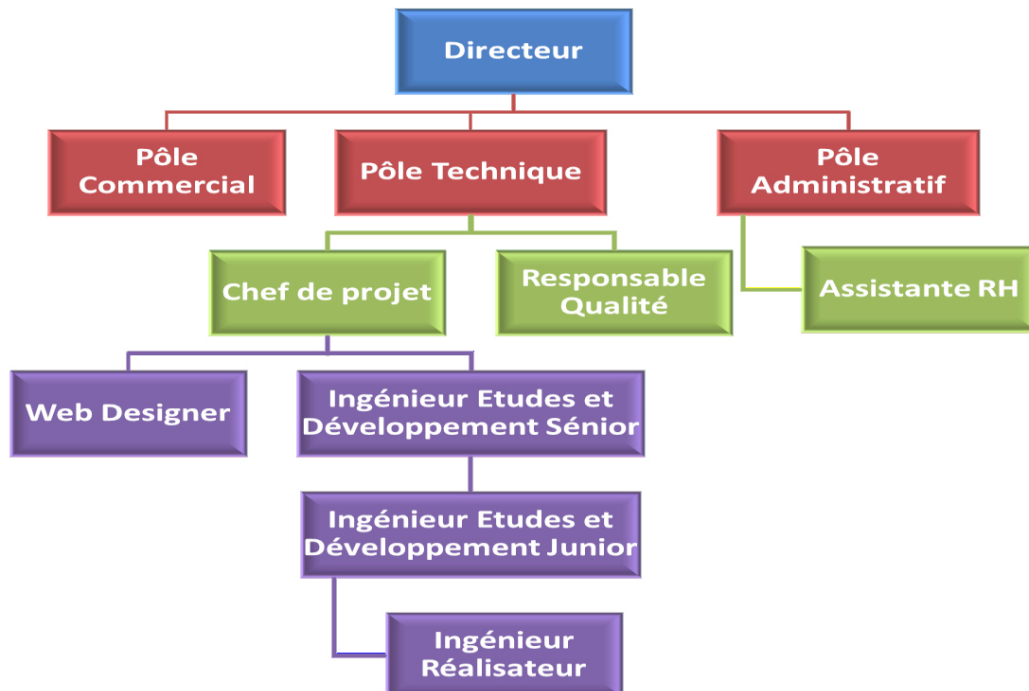


Figure 1: Organigramme de G-FIT

2 Description du projet

2.1 La solution Koala-BPM

L'ERP Koala-BPM est un outil de mesure pour l'entreprise et d'analyse entièrement paramétrable créée en utilisant l'ERP OFBiz. Il permet de répondre à tout type d'activité opérant dans le secteur **BTP** (Bâtiment et travaux publics) en assurant plusieurs services : La gestion des ressources humaines, La gestion de stock, La gestion d'achat, La gestion de production et de commercialisation, La dématérialisation des processus qualité

2.2 Architecture technico-fonctionnelle de Koala-BPM

« Koala BPM » est perçu comme une collection de composants dont chacun assure une fonctionnalité bien déterminée et un périmètre très bien défini. Les composants ont des fonctionnalités autonomes et surtout non dupliquées, ils respectent ainsi deux notions : faible couplage et cohérence forte. Il sera donc facile de gérer les différents composants, on pourra installer et désinstaller des modules en un clic selon les besoins exprimés par le client.

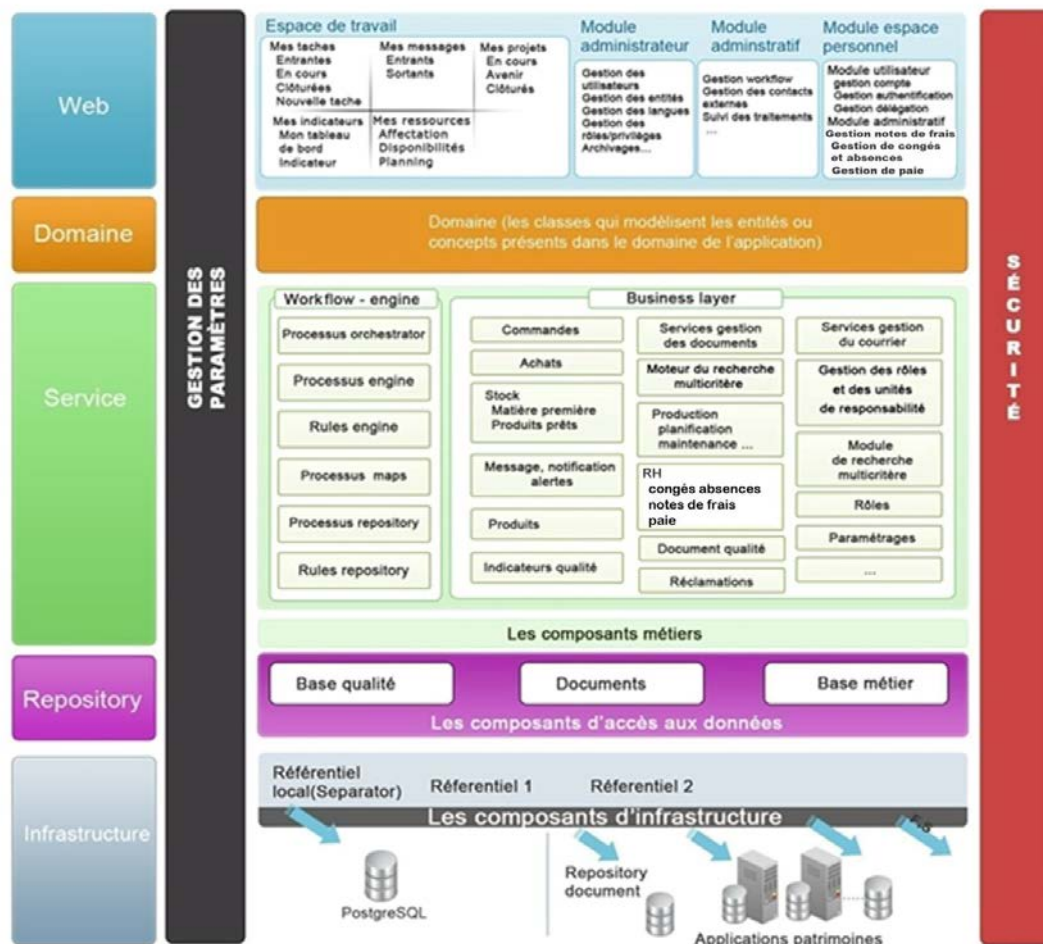


Figure 2: architecture technico-fonctionnelle de koala-bpm

L'ERP « Koala-BPM » est structuré en 4 couches :

- La couche « Web » qui représente la couche présentation de notre système de management intégré. Elle est réalisée sous forme d'une IHM unifiée qui incorpore toutes les tâches quotidiennes des utilisateurs et permet le paramétrage du système. L'accès aux fonctionnalités selon le rôle ou métier des utilisateurs est paramétré grâce au moteur de workflow **JBPM**.
- La couche « Domaine ». Celle-ci représente la couche qui décrit ou contient les données manipulées par l'application. Avec OFBiz, ces données sont regroupées dans des entités. Ces dernières assurent donc la gestion de ces données et garantissent leurs intégrités. C'est une couche intermédiaire entre la couche web et la couche des services.
- La couche « Service » représente le cœur de notre système. Elle regroupe la définition et l'implémentation de toutes les fonctionnalités offertes par Koala BPM (gestion des commandes, gestion des achats, gestion du stock, gestion RH, gestion de la comptabilité, gestion de la production, gestion de la maintenance, gestion du système de management de qualité...)
- La couche « Repository ». C'est l'entrepôt des données. Dans Koala BPM on utilise deux bases de données : une pour la sauvegarde des documents et l'autre pour les données métiers.

Toutes ces couches sont sécurisées par les mécanismes de sécurité offerts par OFBiz, et paramétrées dans des fichiers XML.

2.3 Problématique

Les outils de reporting et de requêtes sont d'une importance majeure dans un S.I. Ils offrent plusieurs avantages aux différents acteurs du système d'information (utilisateurs, opérateurs, décideurs) parmi lesquelles on peut citer :

- Ils permettent la mise à disposition de rapport périodique préformatés et paramétrables par les opérationnels.
- Ils offrent une couche d'abstraction orientée métier pour faciliter la création de rapports par les utilisateurs eux-mêmes.
- Ils permettent la production de tableaux de bord afin de dresser une analyse de l'état de l'entreprise et de ces performances, service par service.

L'entreprise pour laquelle a été développé « Koala-BPM » utilise une base de données relationnelle comme source de données pour créer leurs rapports, il se base sur le résultat de vues SQL complexe pour les alimenter. Cependant, les vues SQL génèrent une grande masse de données qui ne sont pas tous utile à l'utilisateur final. Une étape de filtrage s'avère donc nécessaire afin de sélectionner les données qui correspondent aux attentes de l'utilisateur

Bien que OFBiz dispose d'un outil qui permet de filtrer les données provenant d'une source et d'exporter le résultat vers différent support ; cette outil présente plusieurs inconvénients:

- Il nécessite un processus de programmation complexe.
- Le fait d'apporter une modification s'avère délicat.

- Il présente plusieurs anomalies techniques (incohérence entre les filtres de recherches générés et le modèle de données...).
- Le système de recherche présente une déficience de critères de recherches.

2.4 Solution

La solution que nous avons proposée pour pallier aux inconvénients précédents est de développer un outil de reporting sur la base du Framework OFBiz qui va offrir les fonctionnalités suivantes :

1. La recherche multicritère des données selon un modèle prédéfini (vue ou table).
2. Le créateur de rapport ; l'utilisateur peut sélectionner de manière visuelle et paramétrable, les éléments d'un rapport.
3. L'exportation vers différents formats PDF, Excel.



Chapitre 2 : Généralités sur les ERP

1 Définition

Un PGI progiciel de gestion intégré (en anglais Enterprise Resource Planning ou ERP) est un « logiciel qui permet de gérer l'ensemble des processus d'une entreprise en intégrant l'ensemble des fonctions de cette dernière comme la gestion des ressources humaines, la gestion comptable et financière, l'aide à la décision, mais aussi la vente, la distribution, l'approvisionnement, le commerce électronique »

Le principe fondateur d'un ERP est de construire une application (paie, comptabilité, gestion de stocks...) de manière modulaire tout en partageant une base de données unifiée. Cela crée une différence importante avec la situation préexistante car les différentes fonctions de l'entreprise étaient gérées par une multitude d'applications dédiées souvent hétérogènes. Ainsi, les Achats, la Comptabilité, la Gestion des Stocks, les Ressources Humaines, la Gestion Commerciale,... sont maintenant totalement interconnectés. Avec l'arrivée de l'ERP, les données sont désormais standardisées et partagées entre les différents modules, ce qui élimine les saisies multiples et évite l'ambiguïté des données multiples de même nature.

Ceci permet un accroissement considérable de la fiabilité des informations puisque la source des données est unique, d'où une réduction des délais et des coûts de traitements. L'autre principe qui caractérise un ERP est l'usage systématique de ce qu'on appelle un moteur de **workflow**, et qui permet, lorsqu'une donnée est entrée dans le système d'information, de la propager dans tous les modules du système qui en ont besoin, selon une programmation prédéfinie. Ainsi, on peut parler d'ERP lorsqu'on est en présence d'un système d'information composé de plusieurs applications partageant une seule et même base de données, par le biais d'un système automatisé prédéfini éventuellement paramétrable (un moteur de workflow).

2 Avantages

Comparés à des applications sur mesure, les ERP / PGI présentent plusieurs avantages :

- Optimisation des processus de gestion (flux économiques et financiers);
- Cohérence et homogénéité des informations;
- Intégrité et unicité du système d'information;
- Partage du même système d'information facilitant la communication interne et externe;
- Globalisation de la formation (même logique, même ergonomie);
- Maîtrise des coûts et des délais de mise en œuvre et de déploiement.

Il est important de remarquer que la mise en place d'un ERP dans une entreprise est souvent le Déclencheur d'une réorganisation et rationalisation de l'ensemble des tâches et processus de l'entreprise.

3 Inconvénients

Les ERP / PGI ne sont cependant pas exempts d'inconvénients :

- Coût élevé;
- Périmètre fonctionnel souvent plus large que les besoins de l'organisation ou de l'entreprise

(Le progiciel est parfois sous-utilisé) ;

- Lourdeur et rigidité de mise en œuvre;
- Difficultés d'appropriation par le personnel de l'entreprise;
- Nécessité d'une bonne connaissance des processus de l'entreprise (par exemple, une commande d'achat et une commande de vente nécessitent deux processus différents: il est important de savoir pourquoi, de savoir décrire les points communs et les différences entre ces deux processus de façon à bien les paramétrer) ;
- Nécessité d'adapter parfois certains processus de l'organisation ou de l'entreprise au progiciel;
- Nécessité d'une maintenance continues.

4 La Business Intelligence dans l'ERP

La BI consiste à exploiter les données concernant une entreprise ou son entourage et de données économiques. Pour que cela marche il faudrait avoir une source d'information est c'est là qu'intervient l'ERP.

L'ERP fournit une riche source d'informations très granulaire sur tous les aspects de l'activité d'une entreprise, il a pour vocation de bien contrôler les flux et d'assurer la cohérence et l'intégrité d'un référentiel unique. La BI pousse ces données un peu plus loin pour permettre la prise de décisions stratégiques et techniques par les décideurs, c'est-à-dire la compréhension du fonctionnement actuel et l'anticipation des actions pour un pilotage éclairé de l'entreprise.

5 Les ERP libres

Le secteur des ERP a depuis quelques années déjà subi un petit bouleversement : l'arrivée de logiciels libres (OFBiz, Tiny ERP, ERP5, Compiere, ...) sur des terres où règnent en maîtres les logiciels propriétaires : SAP, BAAN, Oracle, Dynamics, ...

Le premier avantage des ERP Libre sur leurs alter-ego propriétaires est bien sûr l'absence de coût de licence ; coût qui peut souvent apparaître comme prohibitif pour les PME.

Un autre atout important est la possibilité d'adapter et de faire évoluer soi-même le progiciel sans dépendre du bon vouloir de la société éditrice. En outre, le logiciel libre mobilise souvent des communautés, qui le font évoluer au gré des nouveaux besoins, et qui peuvent répondre rapidement à des demandes précises.

De plus comme tout logiciel libre, les ERP libre donne la garantie de travailler sur des standards ouverts et donc interopérables, avantages stratégiques pour beaucoup d'entreprises.

La présence du logiciel libre sur le marché des ERP n'est donc plus marginale et les ERP Open Source prennent leurs places dans ce secteur.

6 L'ERP libre OFBiz

6.1 Présentation de OFBiz

Open For Business, ou **OFBiz**, est un projet de progiciel de gestion intégré libre initié par deux développeurs américains, Andy Zeneski et David E. Jones, en mai 2001. L'objectif de ce projet est de fournir un ensemble de composants homogènes permettant de développer aisément et rapidement des logiciels libres de gestion. À terme, il est prévu d'obtenir tous les composants nécessaires à un PGI intégrant les modules de gestion suivants :

- un ERP (Enterprise Resource Planning)
- un SCM (Supply Chain Management)

- un CRM (Customer Relationship Management)
- un MRP (Manufacturing Resource Planning)
- un CMS (Content Management System)
- un CMMS (Computerized Maintenance Management System)
- une plateforme de E-Business / E-Commerce

Pour atteindre ces objectifs, OFBiz se base sur de nombreux logiciels libres tels que *Subversion*, *ant*, *Tomcat*, *JPublish*, *FreeMarker*, etc. Ces logiciels sont reconnus pour leur qualité et ils assurent l'indépendance du projet. De même, OFBiz respecte de nombreux standards pour garantir un maximum de compatibilité avec les systèmes existants et futurs. En particulier, J2EE et XML qui est largement utilisé dans tout le projet pour décrire les données et les traitements. Par ailleurs, le code source du projet est publié sous la licence MIT qui est libre et permissive, c'est-à-dire qu'elle ne fixe aucune obligation et/ou interdiction quant à l'utilisation, la modification, l'extension et la commercialisation du logiciel. Grâce à l'ouverture du code, une véritable communauté d'utilisateurs et de développeurs s'est formée. Cette dernière assure ainsi la réactivité et la qualité du projet.

Comme indiqué précédemment, un PGI est un logiciel complexe mettant en œuvre de nombreux concepts de gestion. C'est pourquoi, la première partie de mon stage a consisté à comprendre l'architecture d'OFBiz, son fonctionnement et à identifier le rôle de ses différents composants. La suite de cette présentation est donc la synthèse de cette étude. Elle permettra ensuite de mieux situer le développement de mon composant.

6.2 Architecture de OFBiz

OFBiz est une application java client-serveur compatible avec la spécification J2EE définissant une architecture logicielle standard. On retrouve ainsi les trois éléments caractéristiques d'une architecture 3-tiers :

- les clients : ici des clients légers, typiquement des machines peu puissantes disposant d'un navigateur internet ;
- un serveur exécutant les différentes applications OFBiz ;
- et une ou plusieurs bases de données stockant le système d'information de l'entreprise.

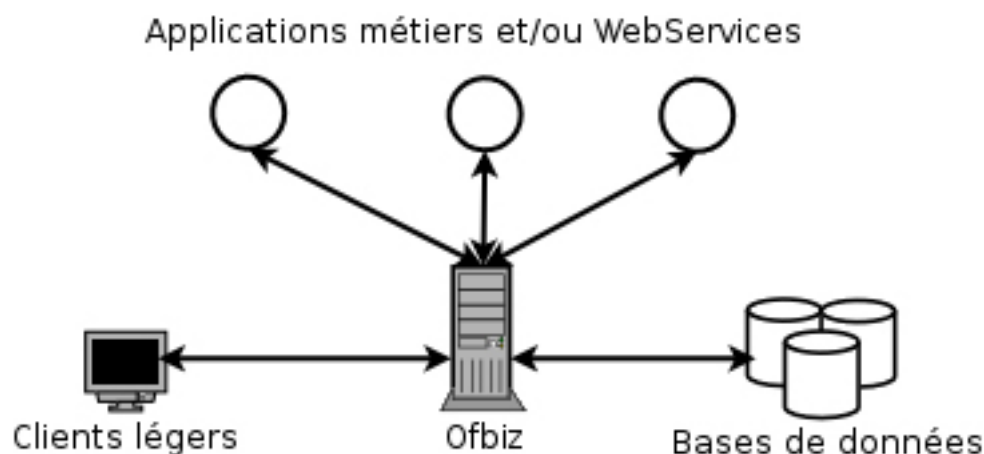


Figure 3: Architecture n-tiers d'OFBiz

Néanmoins, OFBiz peut aussi être considéré comme une architecture n-tiers car elle peut faire appel à des applications externes via des services. Ces derniers ne sont pas forcément exécutés sur la même machine, on les appelle alors WebServices. Ce type d'architecture présente de

nombreux avantages. Elle permet de distribuer plus librement la logique applicative, ce qui facilite la répartition de la charge entre tous les niveaux. Elle facilite l'intégration de l'application avec celles déjà existantes. Enfin, elle permet d'accéder à un très grand nombre de fonctionnalités.

Une des caractéristiques principales d'OFBiz est la modularité de son architecture. En effet, tout est composant. Cette approche favorise une meilleure réutilisation des composants logiciels, un développement modulaire donc plus rapide et enfin une meilleure qualité.

Ce type d'architecture permet aussi de remplacer un composant par un autre très facilement dans le cas où il existe plusieurs implémentations différentes.

Comme tout ERP qui se respecte, OFBiz est décomposé en niveaux, chacun offrant des services particuliers.

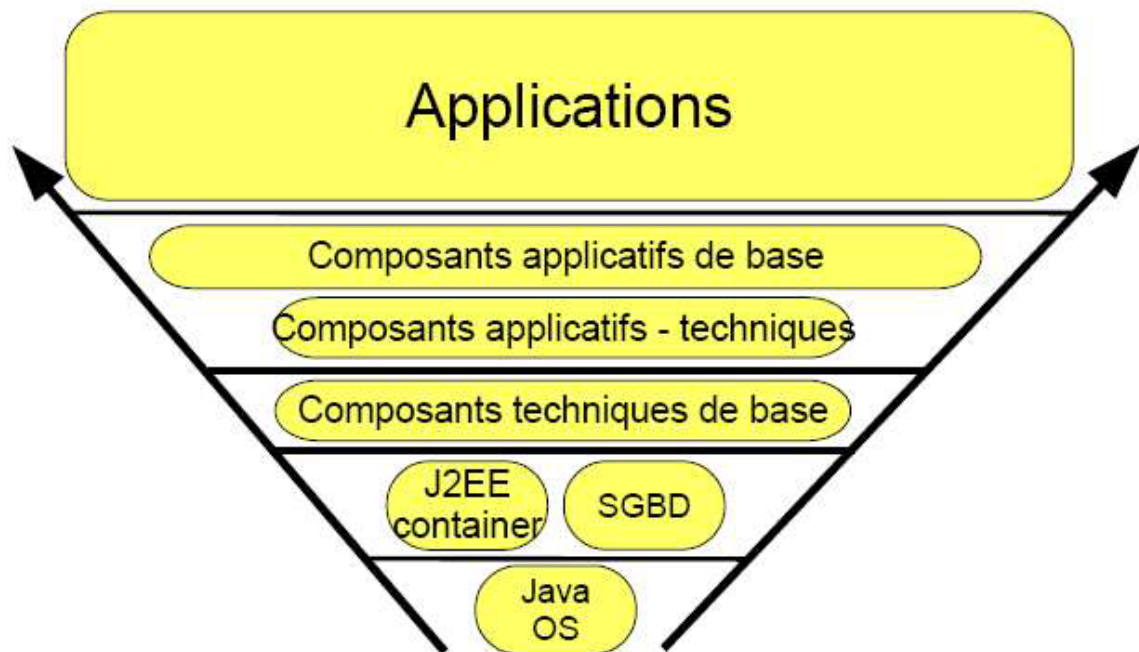


Figure 4:Architecture d'OFBiz

De plus, les niveaux applicatifs sont décomposés en module permettant un développement modulaire tout en facilitant la réutilisabilité du code. Les interactions entre modules ainsi fixées, on peut aisément changer l'implémentation d'un module sans bouleverser complètement le système.



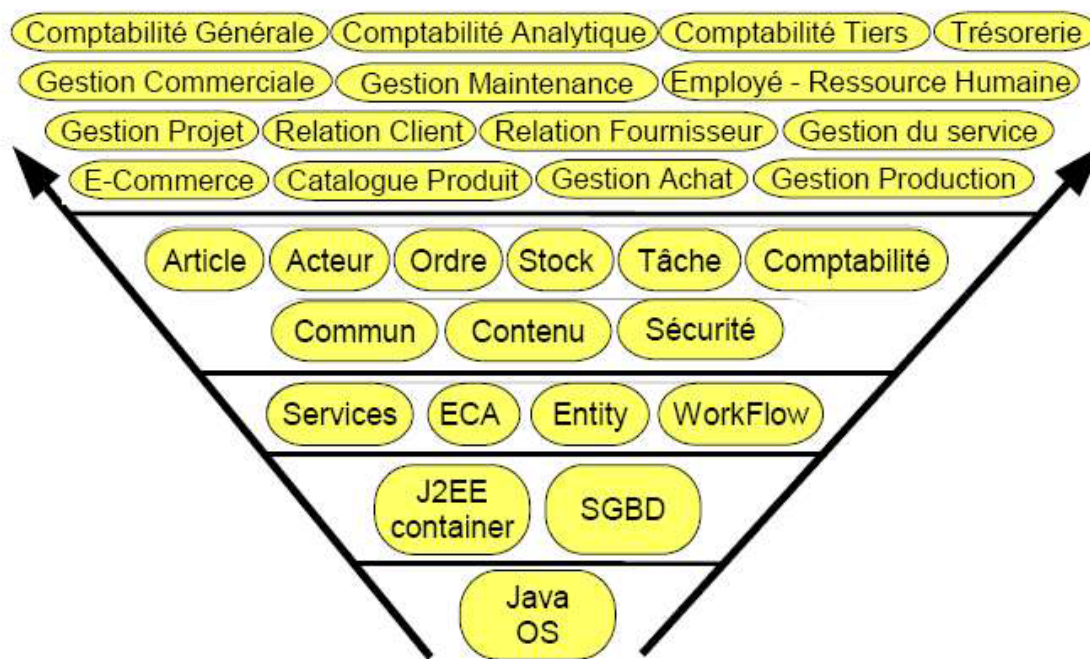


Figure 5: Architecture détaillée d'OFBiz

6.3 Le Framework

Le Framework d'OFBiz est un de ses points forts. S'il a été développé en premier lieu pour OFBiz, il est parfaitement utilisable pour d'autres applications. Son fonctionnement repose sur trois composants majeurs que nous allons décrire : l'EntityEngine, le ServiceEngine, le ControlServlet.

6.3.1 L'Entity Engine

L'Entity Engine d'OFBiz offre un ensemble d'outils et de mécanismes permettant la modélisation et la gestion des données. Une entité est un élément de donnée défini par des champs et un ensemble de relations avec les autres entités. La modélisation est basée sur le modèle entité-relation. Son but premier est de gérer automatiquement la persistance des données dans une application transactionnelle.

L'Entity Engine permet de s'abstraire de la base de données sous-jacente utilisée. Ainsi les entités sont définies dans des fichiers au format XML (composant/entitydef/entitymodel.xml) qui permettront de faire le lien avec la source de données.

```
<entity entity-name="CommEventContentAssoc" package-name="org.ofbiz.party.communication"
  title="CommunicationEvent Content Association Entity">
  <field name="contentId" type="id-ne"></field>
  <field name="communicationEventId" type="id-ne"></field>
  <field name="commContentAssocTypeId" type="id"></field>
  <prim-key field="contentId"/>
  <prim-key field="communicationEventId"/>
  <prim-key field="fromDate"/>
  <relation type="one" fk-name="COMMEV_CA_FROM" title="From" rel-entity-name="Content">
    <key-map field-name="contentId"/>
  </relation>
  <relation type="one" fk-name="COMMEV_CA_COMMEV" rel-entity-name="CommunicationEvent">
    <key-map field-name="communicationEventId"/>
  </relation>
  <relation type="one" fk-name="COMMEV_CA_TYP" rel-entity-name="CommContentAssocType">
    <key-map field-name="commContentAssocTypeId"/>
  </relation>
</entity>
```

Figure 6:Exemple de définition d'une entité

Sont donc définis :

- les propriétés de l'entité : nom, paquet, titre;
- les propriétés des champs : nom, type (qui est automatiquement traduit en type SQL avec un correspondant Java)
- Le nom des clés primaires (qui peuvent être composées) ;
- les relations : le nom de l'entité avec qui se fait la relation, la cardinalité de la relation (1 ou 0..1 ou 0..*)

Pour permettre de minimiser au maximum le code spécifique à une entité, les objets générés sont génériques et on accède aux champs par une Map, la clef étant le nom du champ.

Les classes du package **org.ofbiz.entity** définissent l'API pour interagir avec l'Entity Engine. Les classes utiles pour l'utilisateur sont :

- GenericValue : l'objet générique représentant l'entité ;
- GenericDelegator qui permet de manipuler les GenericValue, c'est-à-dire les créer, les trouver, les stocker.

6.3.2 Le Service Engine

Le Service Engine est l'équivalent de l'Entity Engine pour tous les traitements des composants OFBiz. Les traitements sont appelés Services et peuvent être exécutés localement ou à distance. Le principal intérêt de ce composant est qu'il permet de lancer des services sans avoir besoin de connaître leur localisation et leur implémentation. C'est le ServiceDispatcher qui se charge alors de trouver l'implémentation du service et de son exécution. Un autre intérêt est la possibilité de rendre disponible tout service OFBiz vers l'extérieur. Ainsi un composant peut appeler un service d'un autre composant.

Les services sont définis par des fichiers XML

L'utilisation des services permet de garantir une uniformité dans la manipulation des données et d'engendrer des contrôles tels que le login ou la gestion des erreurs.

```
<service name="updatePerson" engine="java" default-entity-name="Person"
  location="org.ofbiz.party.party.PartyServices" invoke="updatePerson" auth="true">
  <description>Update a Person</description>
  <permission-service service-name="partyGroupPermissionCheck" main-action="UPDATE"/>
  <auto-attributes mode="IN" include="pk" optional="true"></auto-attributes>
  <auto-attributes mode="IN" include="nonpk" optional="true"/>
  <attribute name="preferredCurrencyUomId" type="String" mode="IN" optional="true"/>
  <attribute name="description" type="String" mode="IN" optional="true"/>
  <attribute name="externalId" type="String" mode="IN" optional="true"/>
  <attribute name="statusId" type="String" mode="IN" optional="true"/>
  <override name="firstName" optional="false"/>
  <override name="lastName" optional="false"/>
</service>
```

Figure 7: exemple de définition d'un service

Sont ainsi définis :

- le nom du service ;
- le langage utilisé ;
- la localisation ;
- les permissions
- le nom de la fonction à appeler
- les entrées et les sorties,

La définition d'une méthode Java appelée en tant que service est la suivante :

```
public static Map updatePerson(DispatchContext dctx, Map context)
```

Les données en entrées et sorties sont contenues dans le contexte, des tests sont automatiquement effectués afin de savoir si les données sont conformes à leur types.

6.3.3 Le ControlServlet

Le ControlServlet est un ensemble de classe permettant la présentation d'une application web autour du Framework d'OFBiz. Son but premier est d'apporter un mécanisme de séparation propre entre la présentation logique des données et l'affichage selon le principe modèle-vue-contrôleur.

OFBiz propose une multitude de moteurs de rendu, les plus utilisés actuellement étant les Widgets.

6.3.4 Traitement d'une requête utilisateur

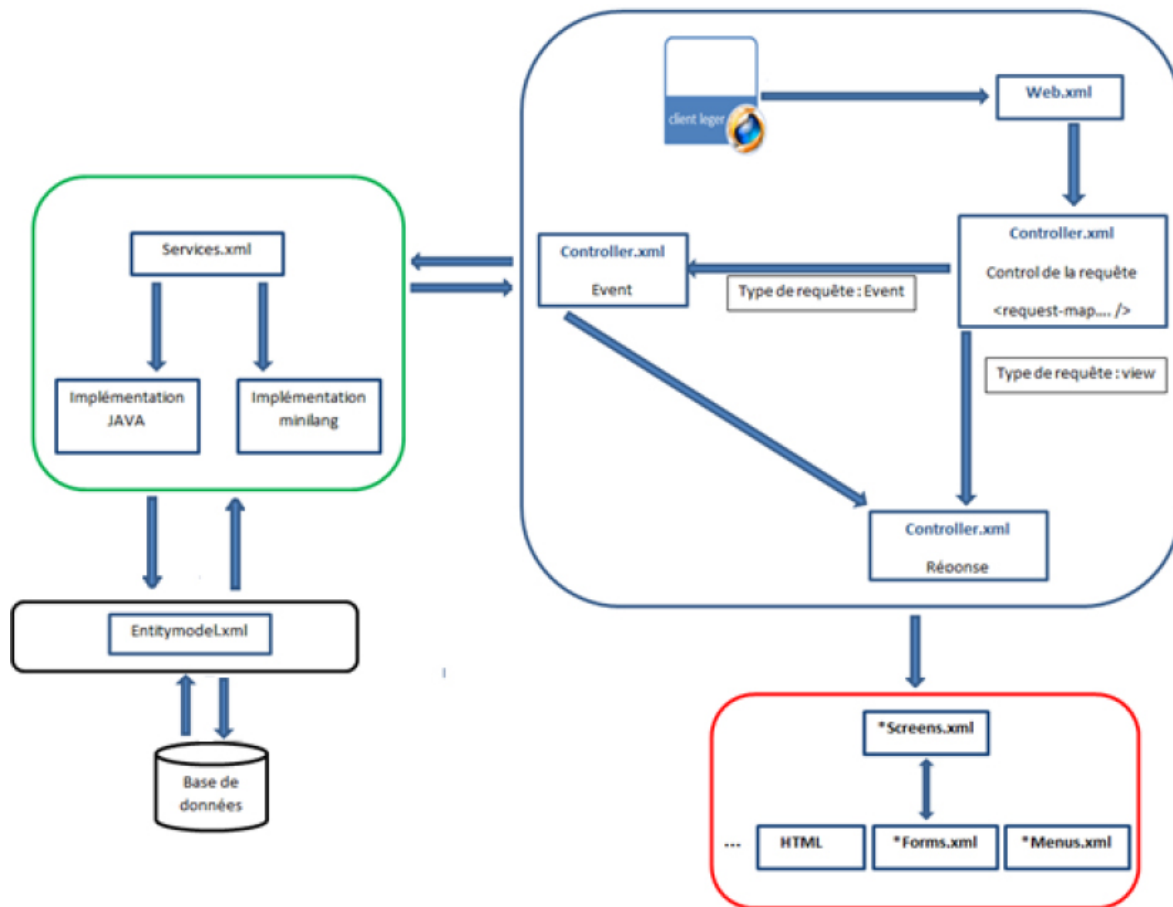


Figure 8: Traitement d'une requête utilisateur par OFBiz

Le schéma ci-dessus montre les opérations effectuées pour traiter une requête utilisateur.

1- L'utilisateur clique sur un lien hypertexte ou valide un formulaire. Le navigateur envoie alors une requête HTTP au serveur OFBiz qui est interceptée par Tomcat (le fichier « web.xml ») et transmise au ControlServlet (le fichier « controller.xml ») de l'application web correspondante.

2- Le ControlServlet vérifie si l'URI reçue est définie dans le module. Le cas échéant, il appelle le ou les événements associés à cette URI. Dans le cas contraire, il renvoie une erreur au navigateur web de l'utilisateur (Erreur HTTP 404 : page non trouvée).

3- Si l'évènement correspondant doit appeler une vue (type de requête= view), le ControlServlet sélectionne la vue à afficher et appelle le moteur de rendu adéquat.

4- À partir de la définition d'une vue, le moteur de rendu « Screens.xml » construit les différents sous éléments de cette dernière qui peuvent être des formulaires, du code HTML, etc.

5- Le moteur de rendu assemble les différents sous-éléments pour former une page web complète.

6- Le ControlServlet transmet la page générée au navigateur web de l'utilisateur.

7- Si l'évènement généré doit appeler un service (type de requête= event), il vérifie que les paramètres de la requête correspondent aux attributs du service et les convertit depuis la textuelle en objets Java correspondants.

8- L'évènement appelle le service convenable « Services.xml ».

9- Le service vérifie le langage utilisé pour son implémentation (java ou minilang) et exécute la méthode convenable qui peut effectuer des actions sur la base de données.

10- L'Entityengine convertit ces actions en requêtes SQL pour le serveur de base de données en utilisant le modèle des données «entitymodel.xml».

11- La méthode exécutée renvoie le résultat de ces actions et le transmet au ControlServlet.

12- À partir du résultat de l'évènement, le ControlServlet sélectionne la vue à afficher et appelle le moteur de rendu adéquat et exécute les étapes 3, 4, 5 et 6.

Chapitre 3 : Analyse et conception

1 Etude des besoins

L'outil doit créer ce qu'on appelle **écran**. Un écran est une interface servant à effectuer des recherches, il est lié à une entité de donnée (vue ou table) et paramétrable par les champs qui constituent cette entité.

Enfin cet outil de reporting repose sur un processus bien défini qui est constitué de 3 phases clés :

- **La phase de création de l'écran**

Dans cette phase, l'utilisateur choisit parmi une liste d'entité provenant d'une base de donnée celle qui va être associée à son nouvel écran et qui va servir à alimenter le rapport. Il donne ensuite un nom à cet écran et valide les informations pour passer à la phase de paramétrage.

- **La phase de paramétrage de l'écran**

Dans cette phase, le nom et le type de chaque champ constituant l'entité choisie dans la première phase sont récupérés. L'utilisateur est amené à sélectionner les champs qui vont participer à l'interrogation ainsi que les champs qui vont être affichés dans le résultat.

Remarque : tous les types sont convertis en un type générique (voir figure ci-dessous) ils sont utilisés dans cette phase seulement à titre informatif .Leur grande utilité proviendra dans la 3^{ème} et dernière phase.

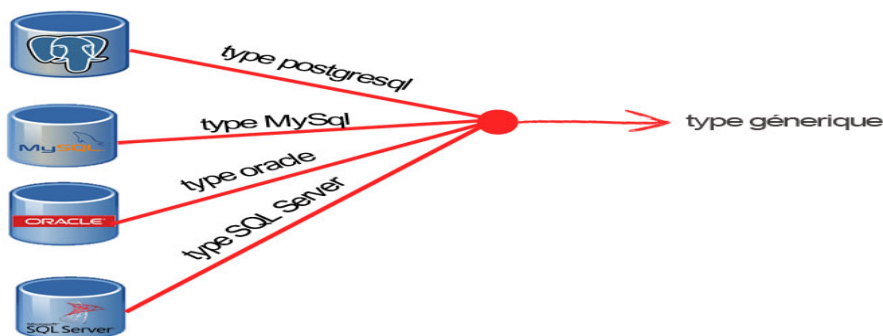


Figure 9:conversion d'un type SGBD vers un type SQL générique

▪ **La phase d'interrogation et de publication du rapport**

Cette phase représente le cœur du processus de reporting, elle fournit des capacités d'interrogation complexe en permettant de filtrer les données en fonction des conditions de requête prédéfinies .le résultat obtenu à la fin de l'interrogation peut être publié vers des fichiers PDF ou Excel.

Pour chaque champ choisi pour participer à l'interrogation, une forme de recherche avec une liste de conditions prédéfinies est générées dynamiquement et dépend du type de ce champ.

Les tableaux ci-dessous montrent les conditions appropriées à chaque type.

a)les champs avec un type texte

condition	valeur	Valeur accepté	Valeur non accepté
Contient	Microsoft	Microsoft Corporation	Micro Corp.
Ne contient pas	Soft Corp.	EMC Corporation	Microsoft Corporation
Egal	Cisco Systems	Cisco Systems	Citrix Systems
Différent	HP	DELL Sony	HP
Début par	Mac	Macintosh IIX Macintosh IICX Macintosh IICI	Power Macintosh
Se termine par	Corp.	Microsoft Corp. Oracle Corp. SAP Corp.	NetApp
Est vide			Symantec
N'est pas vide		Apache Ant Apache Maven Apache Hadoop	

b) les champs avec un type numérique

condition	valeur	Valeur accepté	Valeur non accepté
Entre	1.5 - 95	4 95	1 100
N'est pas entre	1.5 - 95	1 100	1.5 50
Plus grand	20	21 40	5 20
Plus petit	20	14 1	20 30
Plus grand ou égal	100	100 150	2 90
Plus petit ou égal	100	2 100	101 150
Egal	100	100 100.00 0100	90
Différent	100	90 101	100 100.00 0100
Est vide			100 1001
N'est pas vide		1000 500	

c) les champs avec un type Date ou Time

condition	valeur	Valeur accepté	Valeur non accepté
Entre	05/05/2013 31/12/2013	06/11/2013 20/10/2013 31/12/2013	04/05/2013 01/01/2014
N'est pas entre	05/05/2013 31/12/2013	04/05/2013 01/01/2014	06/11/2013 20/10/2013 31/12/2013
Avant	31/12/2013	20/10/2013 01/01/2013	31/12/2013 01/01/2014

après	31/12/2013	01/01/2014 03/02/2014	31/12/2013 01/01/2013
Au plus tard	31/12/2013	20/10/2013 31/12/2013	01/01/2014 03/02/2014
Au plus tôt	31/12/2013	31/12/2013 01/01/2014	04/05/2013 30/12/2013
Egal	01/01/2014	01/01/2014	31/12/2013 02/01/2014
Différent	01/01/2014	31/12/2013 02/01/2014	01/01/2014
Est vide			06/11/2013 20/10/2013 31/12/2013 01/01/2014
N'est pas vide		06/11/2013 20/10/2013 31/12/2013 01/01/2014	

Remarque :

- o tous champs avec un type non conforme avec ceux cités en haut seront ignorés (**exemple** : type binaire)
- o la taille d'un champ joue un rôle pour l'affichage d'une zone de saisie appropriée ainsi que pour l'affichage ou non de certaines conditions (**exemple** : pour un champ de type texte avec un seul caractère seulement les critères *égal*, *différent*, *est vide*, *n'est pas vide* seront affichés.)

2 Modélisation du projet

2.1 Choix de l'outil d'analyse

Pour la partie modélisation, qui m'a été utile aussi bien pour schématiser de façon globale les fonctionnalités de mon module, que pour analyser le fonctionnement des composants existant, j'ai fait le choix de « **Visual Paradigm** ».



Visual Paradigm est un outil de modélisation UML. Il permet d'analyser, de dessiner, de coder, de tester et de déployer. Le logiciel permet de dessiner tous les types de diagrammes UML, d'inverser le code source pour le modèle UML, générer le code source à partir de diagrammes et d'élaborer la documentation.

2.2 Le diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. Etant le premier diagramme du modèle UML, celui-ci s'assure de la relation entre l'utilisateur et les objets que le système met en œuvre.

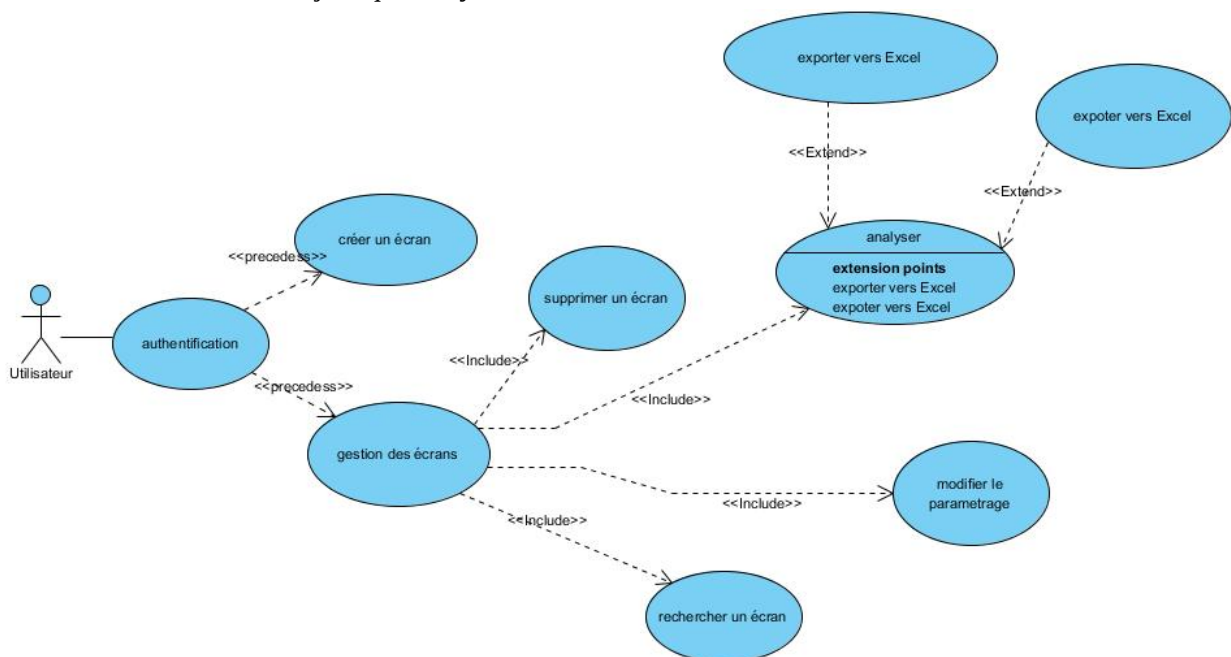


Figure 10:diagramme des cas d'utilisations

2.3 Le diagramme de classe

Le diagramme de classe, illustré dans la figure suivante, représente une vue physique des différentes entités persistantes du module développé avec leurs relations.

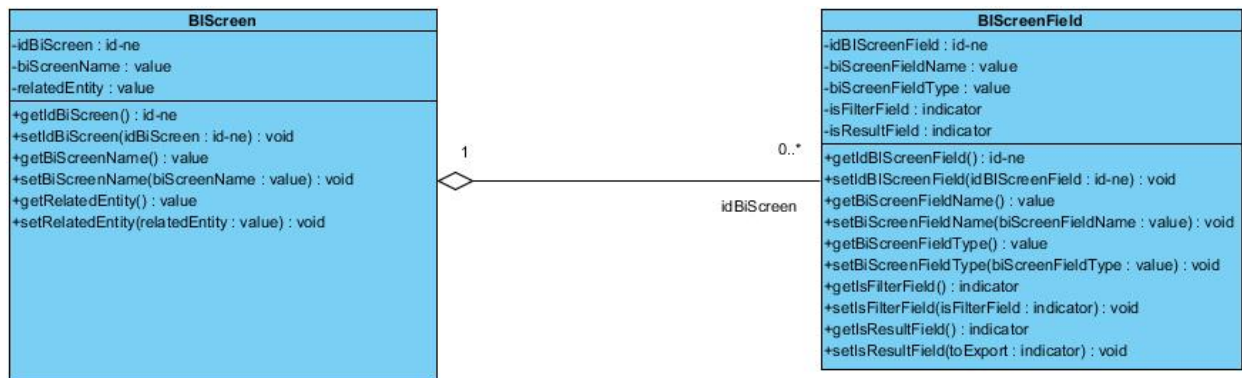


Figure 11:Diagramme de classe

Description détaillée du diagramme:

- o La classe « **BiScreen** » : contient toutes les données concernant un écran à savoir son identifiant, son nom, et le nom de l'entité avec laquelle il est liée.
- o La classe « **BiScreenField** » : elle associée à « **BiScreen** » elle contient les données concernant un champ de l'écran à savoir son identifiant, l'écran avec lequel il est associé, le nom de ce champ est son type ainsi que des indicateurs sur ce champ (s'il va participer à l'interrogation et s'il va être affiché dans le résultat de cette interrogation).

2.4 Les classes techniques

ListingEvents
+module : String = ListingEvents.class.getName()
+generateSearchFields(request : HttpServletRequest, response : HttpServletResponse) : String
+biPdfExport(request : HttpServletRequest, response : HttpServletResponse) : String
+biXlsExport(request : HttpServletRequest, response : HttpServletResponse) : String

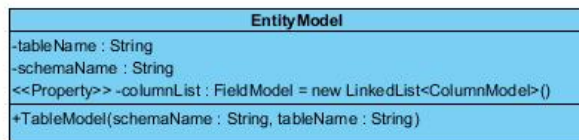
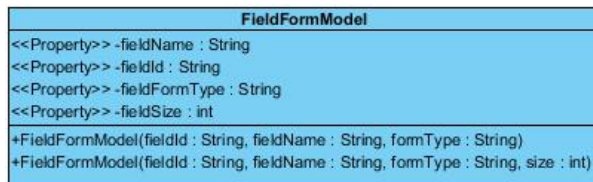
ListingServices
+module : String = ListingServices.class.getName()
+sqlOperators : HashMap<String, String>
+addBiScreen(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+updateBiScreen(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+deleteAllBiScreenFields(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+deleteBiScreen(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+generateSearchFields(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+getAnalysisResult(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>
+editBiScreenConfig(dctx : DispatchContext, context : Map<String, Object>) : Map<String, Object>

ListingWorker
-schemaName : String = "data"
~ListingWorker()
+listEntities() : List
+listEntityFields(tableName : String) : List<ColumnModel>
+getResultFromDb(fields : List<GenericValue>, tableName : String, sortField : String) : LinkedList<LinkedList<Object>>
+getResultFromDbWithCond(fields : List<GenericValue>, tableName : String, conditions : List<String>, sortField : String) : LinkedList<LinkedList<Object>>
+createFieldForm(field : GenericValue, columns : List<ColumnModel>) : FieldFormModel
+getSqlTypeName(type : int) : String
+urlEncodeUTF8(s : String) : String
+urlEncodeUTF8(map : Map<?, ?>) : String
+getFieldLabel(name : String, request : HttpServletRequest) : String

ListingServices : la classe qui implémente les services du module développé.

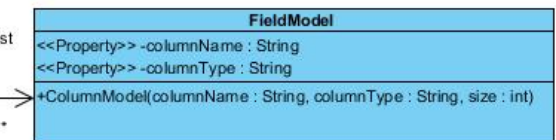
ListingEvents : même fonctionnement que la classe **ListingServices**.

ListingWorker : elle implémente des tâches déléguées par les deux classes précédentes.



columnList

X

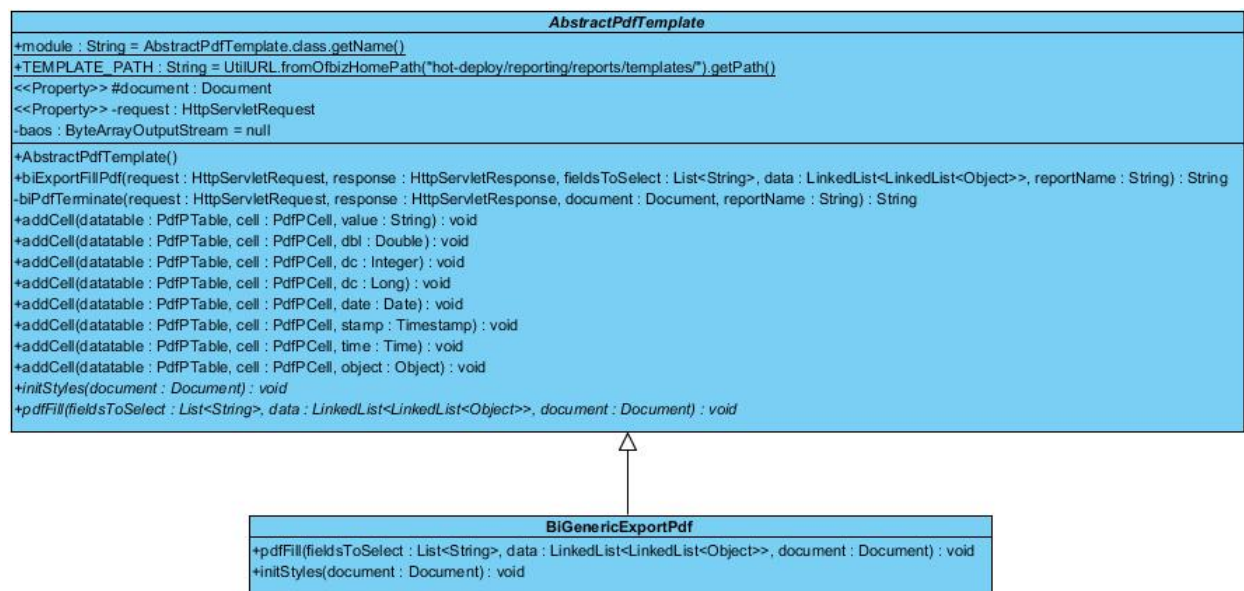


FieldFormModel : représente la modèle de la forme générée pour un champ de l'entité .

EntityModel : représente le modèle de l'entité.

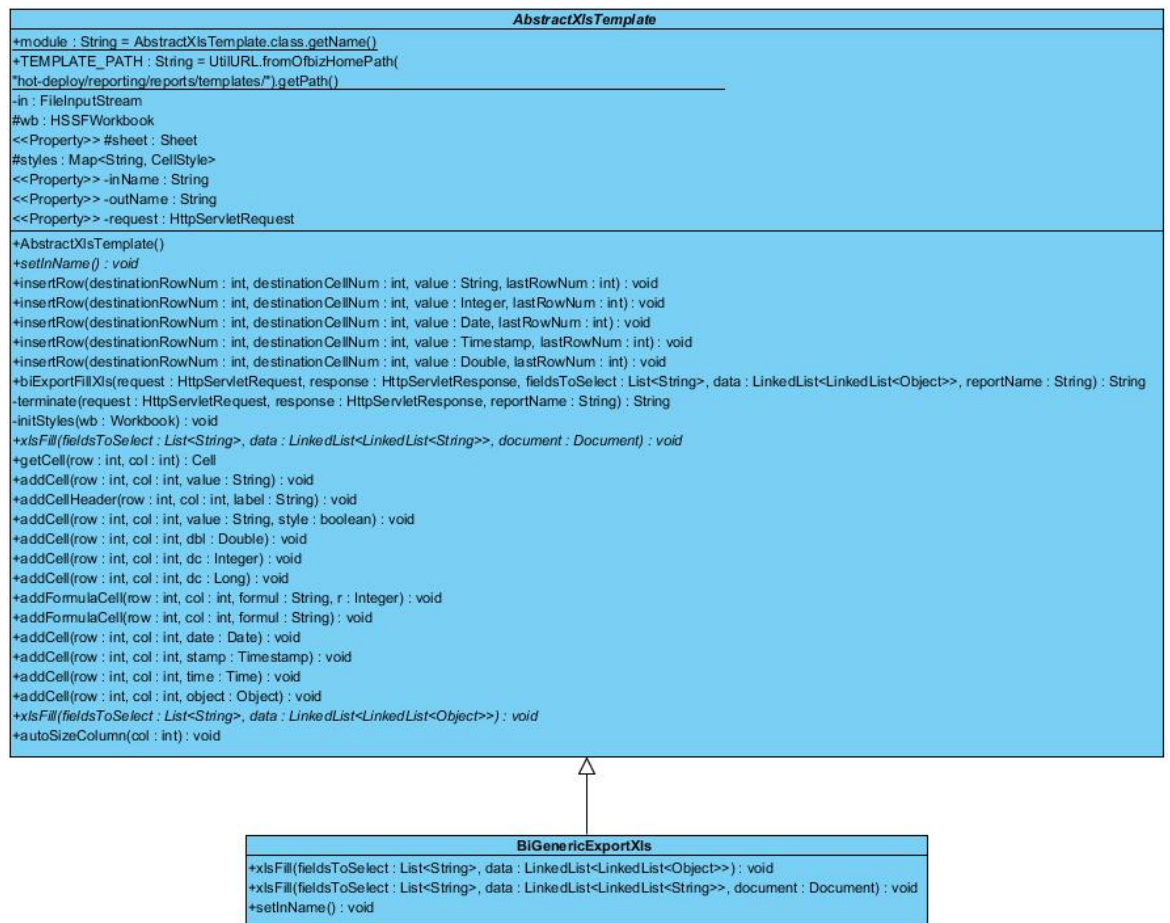
FieldModel : représente le modèle d'un champ de l'entité.

ListingConnection : représente le singleton qui maintient la connexion à la base.



AbstractPdfTemplate : la classe abstraite qui permet la création du rapport PDF.

BiGenericExportPdf : la classe qui étend la classe précédente.



AbstractXlsTemplate : la classe abstraite qui permet la création du rapport Excel.

BiGenericExportXls : la classe qui étend la classe précédente.



Chapitre 4 : Réalisation

1 Outils utilisé

1.1 SVN

a. Description



Subversion est un logiciel de gestion de sources et de contrôle de versions. Ce type de programmes a plusieurs fonctions, notamment :

- garder un historique des différentes versions des fichiers d'un projet ;
- permettre le retour à une version antérieure quelconque ;
- garder un historique des modifications avec leur nature, leur date, leur auteur... ;
- permettre un accès souple à ces fichiers, en local ou via un réseau ;
- permettre à des utilisateurs distincts et souvent distants de travailler ensemble sur les mêmes fichiers.

b. Terminologies utilisées

- « **Repository** » : répertoire ou dépôt sur Serveur Subversion qui va accueillir les fichiers et les données sur les versions. Le dépôt représente le lieu de stockage central des données. les informations y sont organisées en une arborescence de fichiers.
- « **Checkout** » : c'est la commande permettant de récupérer sur le dépôt central une copie de la dernière version.
- « **Update** » : c'est la commande qui consiste à synchroniser la copie de travail locale avec celle du dépôt en récupérant la dernière version des fichiers du dépôt.
- « **Commit** » : c'est la commande qui permet d'envoyer au serveur SVN la nouvelle version des fichiers modifiés. Une mise à jour est ainsi effectuée sur le dépôt central
- « **Merge** » : c'est l'opération qui consiste à mettre à jour un fichier local dont la copie originale a été modifiée avec les mises à jour effectuées sur le dépôt central

1.2 ANT



ANT (**A**nother **N**eat **T**ool) est un projet open source de la fondation Apache écrit en Java qui permet entre autres l'automatisation des opérations répétitives (compilation, création d'archives jar, sauvegarde, génération des documentations, etc.) tout au long du cycle de développement

logiciel. Il a été développé pour fournir un outil de construction indépendant de toute plateforme. Ceci est particulièrement utile pour des projets développés sur et pour plusieurs systèmes, ou pour migrer des projets d'un système vers un autre.

ANT utilise un fichier build.xml pour décrire les différentes tâches qui devront être exécutées par l'outil. Ces tâches sont donc indépendantes du système sur lequel elles seront exécutées. De plus, il est possible d'ajouter ces propres tâches en écrivant de nouvelles classes Java respectant certaines spécifications.

L'IDE Eclipse par exemple intègre ANT et prévoit un fichier build.xml pour tous ses projets.

1.3 FreeMarker

<FreeMarker>

FreeMarker est **un moteur de modèle**. Plus spécifiquement, cet outil génère du texte à partir d'un document modèle (Template FreeMarker) et de données (java, XML).

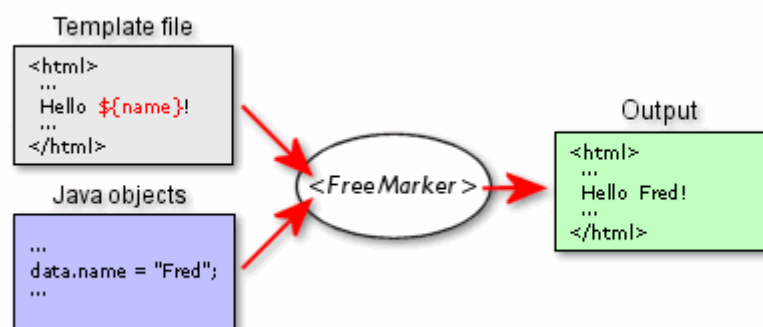
Les grands atouts de FreeMarker sont son implémentation, qui a été totalement réalisée dans **le langage java**, et sa structure, qui permet **une séparation du traitement des données et de leur représentation**. Ceci le rend extrêmement adapté à la génération de page Web HTML, particulièrement dans les applications J2EE ayant une structure en MVC (Model View Controller). Au final, l'utilisation de FreeMarker permet de garder ce découpage architectural qui laisse une cassure entre la représentation et la préparation des données.

FreeMarker est donc basé sur le concept que **les designers et les programmeurs sont des personnes différentes** lors de développement. Les designers ont la charge de créer les pages HTML, les images ou tout autre élément visuel, tandis que les programmeurs ont à leur charge la préparation des données qui seront afficher plus tard. De façon plus précise, les designers vont fabriquer les Template tandis que les développeurs vont créer le programme java structurant l'ensemble des données.

Ce concept implique donc que lors d'une transformation, 2 éléments sources soit nécessaire :

- **Le fichier modèle** : C'est un fichier .ftl définissant les règles d'affichage.
- **Les objets java** : Ce sont les données à afficher qui sont encapsulé dans ces objets.

Lors de cette transformation, le template va recevoir les données à afficher, puis va exécuter son code d'impression comme le montre le schéma suivant (principe du tampon et de l'encre). Ce code peut être composé d'un langage balisaire (de type XML) fournit avec FreeMarker .



1.4 PostgreSQL



PostgreSQL est un système de gestion de base de données relationnelle et objet (SGBDRO). C'est un outil libre et fiable qui garantit l'intégrité des données. Il possède de nombreuses caractéristiques faisant de lui un SGBD digne des SGBD commerciaux :

- Des interfaces graphiques pour gérer les tables.
- Il peut être étendu par l'utilisateur de diverses façons : nouveaux types de données, nouvelles fonctions etc.
- Il fonctionne sur diverses plates-formes matérielles et sous différents systèmes d'exploitation.
- Il est largement reconnu pour son comportement stable, proche d'Oracle mais aussi pour ses possibilités de programmation étendues, directement dans le moteur de la base de données, via PL/pgSQL.

1.5 SpringSource Tool Suite



Le SpringSource Tool Suite (STS) est un environnement de développement intégré extensible, universel et polyvalent, basé sur Eclipse permettant de créer des applications java/J2EE basées sur le Framework Spring ou non. Il est déjà configuré avec les plugins nécessaires pour travailler avec les serveurs web et OSGI - Open Services Gateway Initiative

2 Aperçue sur le travail réalisé

2.1 Arborescence du module développé

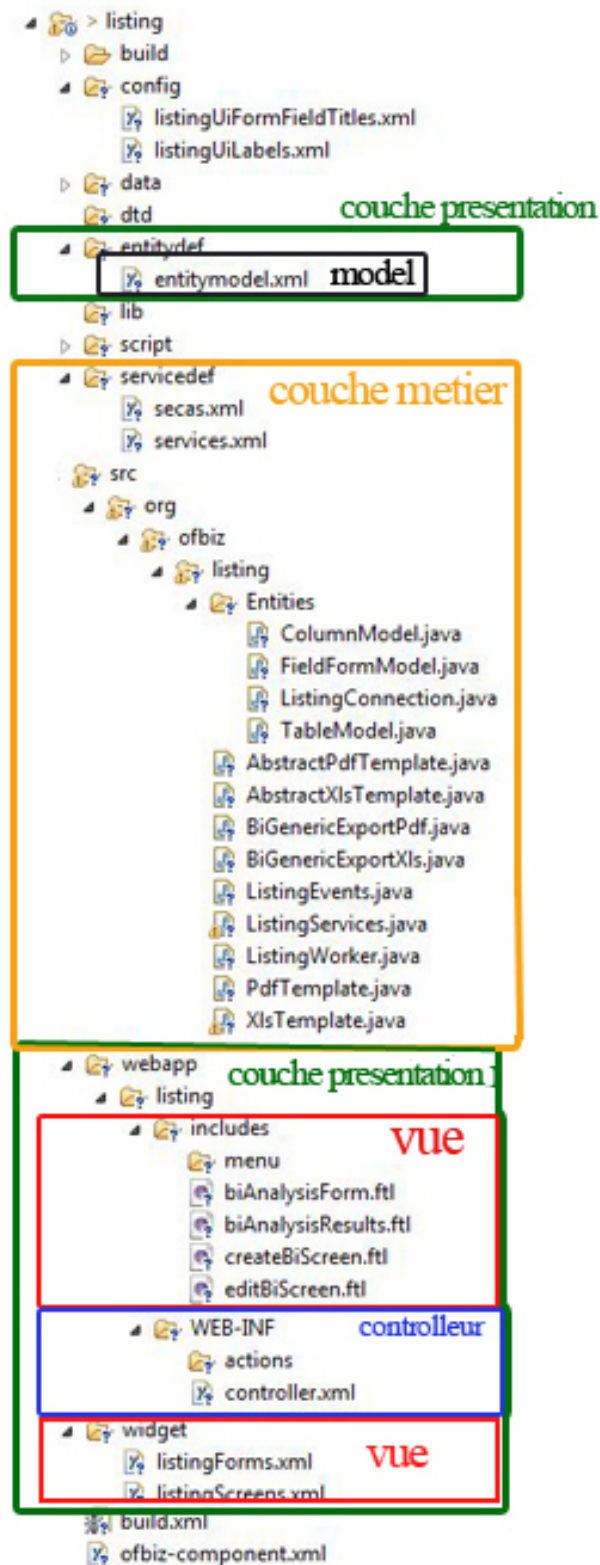


Figure 12:Arborescence du module

Le module est découpé en trois couches distinctes :

- La couche **Présentation** est chargée de tout ce qui est affichage, elle est implémentée selon le modèle MVC
- La couche **Métier** est la logique métier de l'application, elle représente le cœur du module et c'est elle qui définit toutes les classes qui interagissent avec la base de données.
- La couche **DAO** est l'intermédiaire entre les autres couches et la Base de données.

Remarque : La couche DAO est gérée par le moteur d'entité d'OFBiz « **Entity Engine** », qui se sert des entités modélisées dans des fichiers XML « **entitymodel.xml** ».

2.2 Présentation des interfaces

L'utilisateur peut se connecter à l'aide d'un navigateur web (Firefox, Google Chrome, ou internet explorer version 7.0 ou plus). Pour ce faire il devra se connecter à travers une page d'authentification comme suit :

Utilisateur enregistré

Nom de l'utilisateur

Mot de passe

Connexion Vous avez oublié votre mot de passe ?

Figure 13:interface d'authentification

Une fois connecté, on retrouve la page d'accueil suivante :

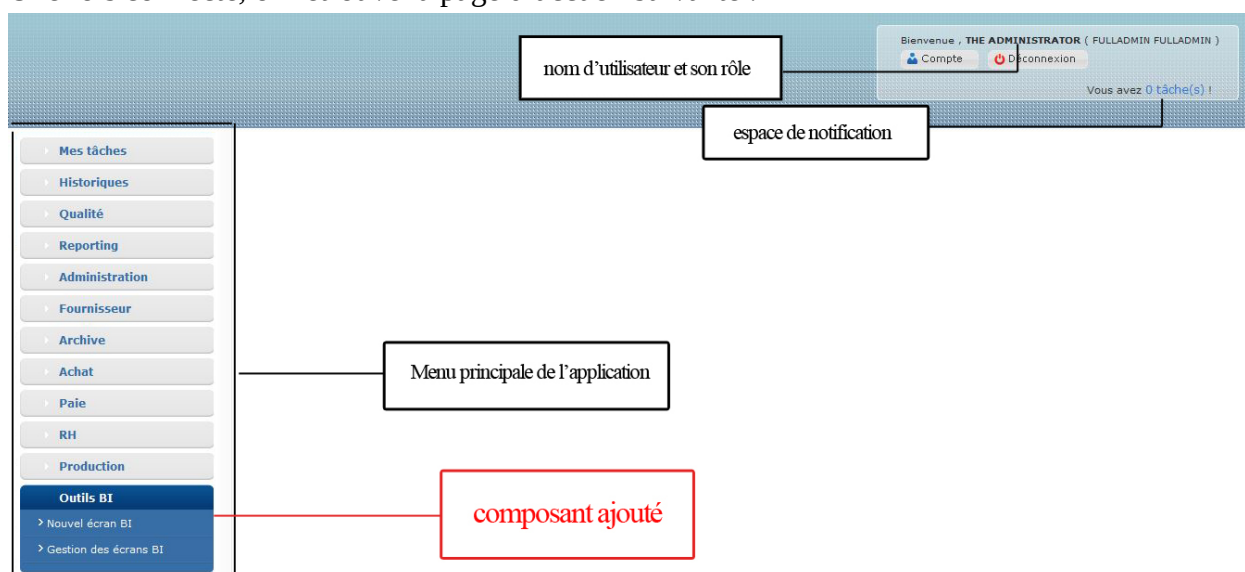


Figure 14:Page d'accueil de l'application

Pour créer un nouvel écran il faut cliquer sur le menu « **outils BI** » puis choisir le sous-menu « **nouvel écran BI** » ainsi le formulaire suivant s'affiche.

Créer un nouvel écran

Nom de l'écran:

(Vue/Table) associé:

Envoyer

Figure 15: Formulaire d'ajout d'un nouvel écran»

L'utilisateur est amené à saisir un nom pour son écran ainsi que l'entité qui va lui être associé depuis la liste déroulante qui supporte l'auto complétion. Une fois le formulaire est validé l'utilisateur est redirigé vers l'interface de paramétrage est un message de confirmation s'affiche pour indiquer que l'écran a été bien créé.

Editer la configuration de l'écran

Nom de l'écran:

(Vue/Table) associé:

Opération réussie !

Libellé	Nom du Champ	Type du Champ	Filtre de recherche	Résultat et Export
Id Commande	id_smi_commande	VARCHAR	☒	☒
Ref Commande	ref_commande	VARCHAR	☒	☒
Date Création	date_creation	DATE	☒	☒
Id Contact	id_smi_contact	VARCHAR	☒	☒
Id Ligne Commande	id_smi_ligne_commande	VARCHAR	☒	☒
Désignation	designation	VARCHAR	☒	☒
Unité	unite	VARCHAR	☒	☒
Prix Unitaire	pu	DOUBLE	☒	☒
Quantité	quantite	DOUBLE	☒	☒
Id Produit	id_smi_produit	VARCHAR	☒	☒

Envoyer

Figure 16:interface de parametrage de l'écran

Dans cette page l'utilisateur est amené à paramétrer son écran.

Le formulaire affiche les détails de l'entité choisie (nom et type de chaque champ) et invite l'utilisateur à choisir les champs qui vont participer à l'interrogation (**Filtre de recherche**) et

ceux qui vont être affichés dans le résultat **(Résultats et Export)**. Par défaut tous les champs sont sélectionnés.

Le **libellé** c'est le nom descriptif du champ .c'est lui qui va être affiché dans l'interface d'analyse et d'interrogation ainsi que dans les supports PDF et Excel. Un mécanisme est mis en place pour chercher dans l'ERP s'il existe un libellé associé à ce champ, s'il ne le trouve pas il en crée un nouveau.

Exemple : Comme le montre la figure, le champ « **pu** » a déjà un libellé défini quelque part dans l'ERP qui est « **prix unitaire** », mais pour le champ « **id_smi_produit** » il n'existe pas de libellé déjà défini donc il a créé un nouveau en enlevant les **sous-tirets** et l'acronyme **smi** et en mettant la première lettre de chaque mot en majuscule.

Une fois le formulaire est validé, l'utilisateur est redirigé vers l'interface de gestion des écrans.

Ref. BI	Nom de l'écran	(Vue/Table) associé	Action
10382	rapport cominge	smi_demande_conge	[Icones]
10383	affiliate	affiliate	[Icones]
10384	machine	bpm_machine	[Icones]
10385	absent	smi_absence	[Icones]
10386	rapport solde conge	smi_historique_solde_conge	[Icones]

Figure 17:interface de gestion des écrans

Le volet rouge indique des options de recherche, on peut rechercher un écran par sa référence, son nom ou le nom de l'entité associée.

Le volet bleu indique le résultat de recherche et propose 3 actions :

- Le bouton  permet de redirigé à l'interface de paramétrage.

Editer la configuration de l'écran

Nom de l'écran:

(Vue/Table) associé:

Configuration de l'écran:

Libellé	Nom du Champ	Type du Champ	Filtre de recherche	Résultat et Export
Id Ligne Commande	id_smi_ligne_commande	VARCHAR	☐	☐
Désignation	designation	VARCHAR	☒	☒
Unité	unite	VARCHAR	☒	☒
Prix Unitaire	pu	DOUBLE	☒	☒
Quantité	quantite	DOUBLE	☒	☒
Id Produit	id_smi_produit	VARCHAR	☒	☒
Id Commande	id_smi_commande	VARCHAR	☐	☒
Ref Commande	ref_commande	VARCHAR	☐	☐
Date Création	date_creation	DATE	☒	☒
Id Contact	id_smi_contact	VARCHAR	☐	☐

- Le bouton  permet de supprimer l'écran (un pop-up de confirmation sera affiché)

Nom de l'écran **(Vue/Table) associé**

Merci de confirmer cette action

⚠ Etes-vous sûr de vouloir supprimer cette ligne?

- Le bouton  permet de passer à l'interface d'interrogation et de reporting

Options de la recherche

Désignation:

Unité:

Prix Unitaire:

Quantité:

Id Produit:

Ref Commande:

Date Création:

Résultats de la recherche

Désignation	Unité	Prix Unitaire	Quantité	Id Produit	Id Commande	Ref Commande	Date Création
Fourniture et Pose de Bibliothèque	U	4950.0	1.0	FP.MNBS.BIBL	10310	ENTRETIEN	2012-04-11
Mobilier Spécial	EN	225.0	2.0	FP.MNBS.MBSP	10310	ENTRETIEN	2012-04-11
Fourniture et Pose de Serrure	U	60.0	2.0	FP.QUBS.SERR	10310	ENTRETIEN	2012-04-11
Fourniture & Pose de Stores Enroulables	M2	330.0	30.0	FP.STOR.STEN	10310	ENTRETIEN	2012-04-11
Poignée spéciale pour porte	U	208.0	10.0	FP.QUBS.POI1	10310	ENTRETIEN	2012-04-11
Repose De Porte	U	110.0	5.0	FP.MNBS.RPPT	10310	ENTRETIEN	2012-04-11
Fourniture et Pose de Ferme Porte Standard	U	550.0	1.0	FP.QUBS.0001	10310	ENTRETIEN	2012-04-11
Fourniture 1 Pose de Main Courante en Bois Massif Hêtre Teinté Wengué	ML	650.0	45.0	FP.MNBS.MCBS	10310	ENTRETIEN	2012-04-11
Porte Pleine Simple Vantail Dimensions Standard 0,9x2,1	PX	4000.0	10.0	FP.CLPT.0041	10311	5782	2012-04-11
Film Adhésif Décoratif	M2	480.0	1.0	FP.CLFM.0044	10311	5782	2012-04-11

10 par page < Page 1 of 18 >

Figure 18:interface d'interrogation et de reporting

- Le volet rouge indique les options d'interrogation, pour chaque champ choisi à participer à l'interrogation un filtre de recherche avec une zone de texte approprié est généré.

Exemple :

- Pour le champ **Désignation** qui est un champ de type texte le filtre approprié à un type texte est généré est une zone de saisie appropriée à la taille de ce champ.

- Pour le champ **Quantité** qui est un champ numérique le filtre approprié à un type numérique est généré ainsi qu'une zone de saisie appropriée à la taille de ce champ.

- Pour le champ **Date Création** qui est un champ de type date le filtre approprié à un type date est généré ainsi qu'une zone de saisie appropriée à la taille de ce champ.

Remarque : les critères **entre** et **n'est pas entre** nécessitent deux zones de saisies

- Le volet bleu comprend le résultat d'interrogation ainsi que les différentes actions, par default toutes les données sont affichées.

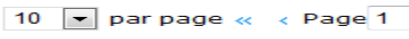
Résultats de la recherche

Désignation	Unité	Prix Unitaire	Quantité	Id Produit	Id Commande	Ref Commande	Date Création
Fourniture et Pose de Bibliothèque	U	4950.0	1.0	FP.MNBS.BIBL	10310	ENTRETIEN	2012-04-11
Mobilier Spécial	EN	225.0	2.0	FP.MNBS.MBSP	10310	ENTRETIEN	2012-04-11
Fourniture et Pose de Serrure	U	60.0	2.0	FP.QUBS.SERR	10310	ENTRETIEN	2012-04-11
Fourniture & Pose de Stores Enroulables	M2	330.0	30.0	FP.STOR.STEN	10310	ENTRETIEN	2012-04-11
Poignée spéciale pour porte	U	208.0	10.0	FP.QUBS.POI1	10310	ENTRETIEN	2012-04-11
Repose De Porte	U	110.0	5.0	FP.MNBS.RPPT	10310	ENTRETIEN	2012-04-11
Fourniture et Pose de Ferme Porte Standard	U	550.0	1.0	FP.QUBS.0001	10310	ENTRETIEN	2012-04-11
Fourniture 1 Pose de Main Courante en Bois Massif Hêtre Teinté Wengué	ML	650.0	45.0	FP.MNBS.MCBS	10310	ENTRETIEN	2012-04-11
Porte Plaine Simple Vantail Dimensions Standard 0,9x2,1	PX	4000.0	10.0	FP.CLPT.0041	10311	5782	2012-04-11
Film Adhésif Décoratif	M2	480.0	1.0	FP.CLFM.0044	10311	5782	2012-04-11

10 par page < Page 1 of 18 >

Cette section comprend plusieurs actions :

- Le bouton  permet de trier le résultat.

- o Le volet de pagination  permet de limiter les résultats par page ainsi que de naviguer vers les autres pages.

- o Le bouton  permet d'exporter le résultat dans un document Excel.

Export des données							
LOGO				VERSION			
Désignation	Unité	Prix Unitaire	Quantité	Id Produit	Id Commande	Ref Commande	Date Création
Fourniture et Pose de Bibliothèque	U	4 950	01	FP.MNBS.BIBL	10310	ENTRETIEN	11/04/2012
Mobilier Spécial	EN	225	02	FP.MNBS.MBSP	10310	ENTRETIEN	11/04/2012
Fourniture et Pose de Serrure	U	60	02	FP.QUBS.SERR	10310	ENTRETIEN	11/04/2012
Fourniture & Pose de Stores Enroulables	M2	330	30	FP.STOR.STEN	10310	ENTRETIEN	11/04/2012
Poignée spéciale pour porte	U	208	10	FP.QUBS.POI1	10310	ENTRETIEN	11/04/2012
Repose De Porte	U	110	05	FP.MNBS.RPPT	10310	ENTRETIEN	11/04/2012
Fourniture et Pose de Ferme Porte Standard	U	550	01	FP.QUBS.0001	10310	ENTRETIEN	11/04/2012
Fourniture 1 Pose de Main Courante en Bois Massif Hêtre Teinté Wengué	ML	650	45	FP.MNBS.MCBS	10310	ENTRETIEN	11/04/2012
Porte Pleine Simple Vantail Dimensions Standard 0,9x2,1	PX	4 000	10	FP.CLPT.0041	10311	5782	11/04/2012
Film Adhésif Décoratif	M2	480	01	FP.CLFM.0044	10311	5782	11/04/2012
Forfait Retouches Fond Plafond en BA13	U	300	45	FE.FPBA.RETO	10311	5782	11/04/2012
Repose De Porte	U	500	14	FP.MNBS.RPPT	10311	5782	11/04/2012
Fourniture et Pose de Revêtement Souple en PVC	M²	500	13	FE.RVSO.001	10311	5782	11/04/2012
Fourniture 1 Pose de Main Courante en Bois Massif Hêtre Teinté Wengué	ML	600	12	FP.MNBS.MCBS	10311	5782	11/04/2012
Porte Pleine Simple Vantail Dimensions Standard 0,9x2,1	PX	3 000	01	FP.CLPT.0041	10312	ZONE FRANCHE	11/04/2012
Film Adhésif Standard	M2	428	06	FP.CLFM.0045	10312	ZONE FRANCHE	11/04/2012
Cloison Bm 80 Couvre Joint Pleine Toute Hauteur Stratidecor	M²	1 045	14	FP.CL80.0019	10312	ZONE FRANCHE	11/04/2012
Cloison Bm 80 Couvre Joint Double Vitrage 6Mm	M2	1 140	08	FP.CL80.0005	10312	ZONE FRANCHE	11/04/2012
Cloison Vitrée avec Pinces de Fixation	M²	1 750	05	FP.CLVI.CLVI	10313	587844	11/04/2012
Fourniture & Pose de Porte Coupe Feu Double 1/2 H	U	25 000	01	FP.MNBS.PCFDV2	10313	587844	11/04/2012
Depose de Fenetre en Bois	U	300	06	FP.MNBS.DPFE	10313	587844	11/04/2012
Caisson Lumineux 60/60	U	800	01	FX.ELEC.0001	10313	587844	11/04/2012

Figure 19:rapport Excel

- o Le bouton  permet d'exporter le résultat dans un document PDF

Export de données							
Logo				Version			
logo Test				Version Test			
Désignation	Unité	Prix Unitaire	Quantité	Id Produit	Id Commande	Ref Commande	Date Création
Fourniture et Pose de Bibliothèque	U	4950.0	1.0	FP.MNBS.BIBL	10310	ENTRETIEN	11/04/2012
Mobilier Spécial	EN	225.0	2.0	FP.MNBS.MBSP	10310	ENTRETIEN	11/04/2012
Fourniture et Pose de Serrure	U	60.0	2.0	FP.QUBS.SERR	10310	ENTRETIEN	11/04/2012
Fourniture & Pose de Stores Enroulables	M2	330.0	30.0	FP.STOR.STEN	10310	ENTRETIEN	11/04/2012
Poignée spéciale pour porte	U	208.0	10.0	FP.QUBS.POI1	10310	ENTRETIEN	11/04/2012
Repose De Porte	U	110.0	5.0	FP.MNBS.RPPT	10310	ENTRETIEN	11/04/2012
Fourniture et Pose de Ferme Porte Standard	U	550.0	1.0	FP.QUBS.0001	10310	ENTRETIEN	11/04/2012
Fourniture 1 Pose de Main Courante en Bois Massif Hêtre Teinté Wengué	ML	650.0	45.0	FP.MNBS.MCBS	10310	ENTRETIEN	11/04/2012
Porte Pleine Simple Vantail Dimensions Standard 0,9x2,1	PX	4000.0	10.0	FP.CLPT.0041	10311	5782	11/04/2012
Film Adhésif Décoratif	M2	480.0	1.0	FP.CLFM.0044	10311	5782	11/04/2012
Forfait Retouches Fond Plafond en BA13	U	300.0	45.0	FE.FPBA.RETO	10311	5782	11/04/2012
Repose De Porte	U	500.0	14.0	FP.MNBS.RPPT	10311	5782	11/04/2012

Figure 20:rapport PDF

Conclusion

L'ERP OFBiz est un logiciel très complet proposant de nombreux composants « métier » mais également des composants « techniques » qui sont liés au Framework .Toutefois, cette richesse s'accompagne d'une complexité architecturale difficile à appréhender dans sa totalité. La création d'un composant constitue alors une tâche d'autant plus complexe.

Ces stage a été très positif et m'a permis d'enrichir mes connaissances et mon savoir-faire, mais surtout mon savoir être.

Tout d'abord, en termes de connaissances, ce stage m'a permis de mettre en pratique mes connaissances sur le développement java ainsi qu'en développement web et de découvrir de nombreuses technologies très utilisées dans le monde de l'entreprise.

Développer un composant décisionnel dédié au reporting pour un ERP, a été pour moi une occasion propice pour comprendre plus en détail les principes de fonctionnement des ERP et de découvrir le monde de la BI, et mesurer son utilité.

En terme de savoir être j'ai pu approfondir mes capacités d'adaptation à un nouvel environnement de travail. En effet, j'ai été ramené à travailler avec des outils et des technologies que je ne connaissais pas avant, et qu'il était indispensable de pouvoir assimiler dans le cadre de mon projet.

Enfin, j'ai pu appréhender à nouveau l'importance capitale de la communication et des relations humaines en entreprise.

Bibliographie

- [1] M. Simhar et J. Word, Integrated Business Processes with ERP Systems, WILEY.
- [2] R. Hoffman, Apache OFBiz CookBook, PACKT PUBLISHING.
- [3] J. Wong et R. Howell, Apache OFBiz Development, PACKT PUBLISHING.
- [4] S. Chen, «The Hitchhiker's Guide to OFBiz,» [En ligne]. Available:
<http://docs.huihoo.com/apache/ofbiz/The-Hitchhiker-Guide-to-OFBiz.pdf>.
- [5] «Apache Ant™ 1.9.3 Manual,» [En ligne]. Available: <http://ant.apache.org/manual/>.
- [6] «FreeMarker Manual,» [En ligne]. Available: <http://freemarker.org/docs/>.
- [7] «Apache OFBiz™ Documentation,» [En ligne]. Available:
<http://ofbiz.apache.org/documentation.html>.
- [8] j.-l. Lequeux, Manager avec les ERP, EYROLLES.

Annexe

Glossaire

ANT :Another Neat Tool

BPM :Business Process Management

ERP :Entreprise Ressource Planning

ISO :Organisation Internationale de Normalisation

JEE :Java Entreprise Edition

MVC :Model-View-Controller

OFBIZ :Open for Business

SI :Système d'information

SMQ :Système de Management de Qualité

SQL :Structured Query Language

STS :Springsource Tool Suite

SVN :Subversion

UML :Unified Modeling Language

XML :eXtensible Markup Language

