

# Conception et réalisation d'une application de gestion des comptes mail et internet

---

Réalisé par : Wahid Mejri & Mohamed Slim Arafa

UNIVERSITE VIRTUELLE DE TUNIS

Licence Appliquées en Sciences et Techniques de l'Information et de  
Communications

Organisme d'accueil : INBMI



Encadré par : Awatef Mejri



## *Remerciement*

*Il me met particulièrement agréable avant de présenter mon travail, d'exprimer toute ma gratitude envers les personnes qui de près ou de loin m'ont apporté leur sollicitude.*

*J'adresse ma profonde reconnaissance au responsable du service messagerie Edunet Mme Awatef Mejri pour l'aide, les explications pertinentes et les conseils précieux qui ont le grand impact dans la réussite du projet réalisé.*

*Je voudrais exprimer, également, mes vifs gratitude et remerciements à mes collègues de l'INBMI pour leur aide et leurs remarques constructives.*

*Enfin, ma profonde reconnaissance au directeur général de l'INBMI Mr Skander Ghneya, Mr Yanes Anmar chef de service équipement et maintenance ainsi que tous les membres de l'administration pour l'effort et le soin accordés afin de réussir notre projet.*

## Contenu

<b>Chapitre 1 : Présentation générale du projet</b> .....	<b>8</b>
<b>1 Présentation du projet</b> .....	<b>9</b>
<b>2 Présentation de l'organisme d'accueil</b> .....	<b>9</b>
2.1 Le rôle de l'INBMI en tant que FSI .....	9
2.2 Architecture des réseaux des établissements.....	10
2.3 Architecture de base : les serveurs SLIS .....	11
2.4 Emplacement dans le réseau .....	12
<b>3 Conclusion</b> .....	<b>13</b>
<b>Chapitre 2 : Etat de l'art</b> .....	<b>14</b>
<b>Introduction</b> .....	<b>15</b>
<b>1 Etude de l'existant</b> .....	<b>15</b>
<b>2 Critiques de l'existant</b> .....	<b>15</b>
<b>3 Présentation du travail demandé</b> .....	<b>16</b>
<b>4 Model et méthodologie adoptés</b> .....	<b>16</b>
4.1 Model .....	16
4.2 Méthodologie.....	17
<b>5 Choix des outils de développement</b> .....	<b>17</b>
5.1 NetBeans.....	17
5.2 GWT .....	17
5.3 PostgreSQL .....	18
5.4 GXT .....	19
5.5 IReport .....	19
<b>6 Conclusion</b> .....	<b>19</b>
<b>Chapitre 3 : Analyse des besoins et spécification</b> .....	<b>20</b>
<b>Introduction</b> .....	<b>21</b>
<b>1 Analyse des besoins</b> .....	<b>21</b>
1.1 Les besoins attendus de l'application .....	21
1.1.1 Les besoins fonctionnels.....	21
1.1.2 Les besoins non fonctionnels.....	21
1.2 Résultats attendus.....	22
<b>2 Les diagrammes des cas d'utilisation</b> .....	<b>22</b>
2.1 Identification des acteurs .....	22
2.2 Les différents cas d'utilisation .....	23
2.2.1 Cas d'utilisation préliminaire .....	23
2.2.2 Cas d'utilisations détaillés .....	25
<b>3 Conclusion</b> .....	<b>31</b>

<b>Chapitre 4 : Conception</b> .....	<b>32</b>
<b>Introduction</b> .....	<b>33</b>
<b>1 Architecture de l'application</b> .....	<b>33</b>
1.1 Présentation de l'architecture à 2 niveaux .....	33
1.2 Présentation de l'architecture à 3 niveaux .....	34
1.3 Architecture adoptée .....	35
<b>2 Conception détaillées</b> .....	<b>35</b>
2.1 Vue statique .....	36
2.1.1 Diagramme des cas d'utilisation .....	36
2.1.2 Diagramme de classes .....	36
.....	37
2.1.3 Diagramme de déploiement.....	38
.....	38
2.1.4 Vue dynamique : Diagramme de séquence .....	39
<b>3 Conclusion</b> .....	<b>44</b>
<b>Chapitre 5 : Réalisation</b> .....	<b>45</b>
<b>Introduction</b> .....	<b>46</b>
<b>1 Environnement de développement</b> .....	<b>46</b>
1.1 Environnement Matériel .....	46
1.2 Environnement Logiciel .....	46
1.3 Choix des outils de développement.....	46
<b>2 Architecture générale de l'application</b> .....	<b>47</b>
<b>4 Diagramme de classe final</b> .....	<b>50</b>
<b>5 Principales interfaces graphiques</b> .....	<b>50</b>
5.1 Authentification .....	51
5.2.....	51
5.2.1 Ajouter un utilisateur .....	51
5.2.2 Consulter les utilisateurs .....	52
5.2.3 Les traces utilisateurs.....	53
5.3 Gestion des comptes Mail .....	54
5.3.1 L'ajout d'un abonné .....	54
5.3.2 La recherche d'un compte et l'impression .....	55
5.3.3 Les états d'un compte : .....	57
5.4 Gestion des serveurs Slis.....	58
<b>Bibliographie</b> :.....	<b>69</b>
<b>Netographie</b> : .....	<b>69</b>
<b>Annexe A : UML</b> .....	<b>61</b>
<b>Annexe B : GWT</b> .....	<b>64</b>

## Liste des figures

Figure 1: Architecture réseau d'un établissement .....	10
Figure 2: Schéma d'un réseau avec passerelle SLIS.....	11
Figure 3: cas d'utilisation préliminaire.....	23
Figure 4 : Cas d'utilisation «S'authentifier ».....	24
Figure 5 : Cas d'utilisation «Gérer droits d'accès ».....	25
Figure 6 : Cas d'utilisation « Gérer les comptes mail ».....	27
Figure 7 : Cas d'utilisation « gestion des serveurs Slis ».....	29
Figure 8: Architecture 2 niveaux .....	33
Figure 9: Architecture 3 niveaux.....	33
Figure 10 : diagramme de classes.....	36
Figure 11 : diagramme de déploiement.....	37
Figure 12 : diagramme de séquence du cas "authentification".....	38
Figure 13 : diagramme de séquence du cas " Gérer les utilisateurs et les droits d'accès ".....	39
Figure 144 : diagramme de séquence du cas " Gérer les utilisateurs et les droits d'accès .....	40
Figure 155 : Diagramme de séquence du cas « gérer les comptes».....	41
Figure 16 : Diagramme de séquence du cas « gérer les comptes».....	42
Figure 17 : Diagramme de séquence du cas « gérer les serveurs Slis ».....	43
Figure 168 : Architecture client-serveur de l'application.....	47
Figure 179 : Liste des évènements de l'application.....	48
Figure 20 : Liste des évènements de l'application.....	48
Figure 21 : Le contrôleur principal (Dispatcher).....	49
Figure 22 : Les Beans.....	49
Figure 2318 : Interface d'authentification.....	50
Figure 2419 : Message d'erreur.....	50
Figure 25 : Interface principale.....	50
Figure 26 : Ajout utilisateur.....	52
Figure 2720 : information sur les utilisateurs.....	52
Figure 2821 : Trace des utilisateurs.....	53
Figure 2922 : Ajout d'un abonné.....	54
Figure 30 : formulaire de génération de compte.....	54
Figure 3123 : recherche de compte par l'établissement.....	54

Figure 32 : Impression de compte.....	55
Figure 33 : Etats des comptes.....	56
Figure 34 : Validation des comptes.....	56
Figure 35 : ajout d'un serveur Slis.....	57
Figure 36 : Ajout d'un Slis.....	57
Figure 37 : Actions sur Slis.....	58
Figure 38: Les différentes vues du langage UML.....	61
Figure 39: Les cas d'utilisation.....	61
Figure 40: Les diagrammes de séquence.....	62
Figure 41: Les associations entre classes.....	63
Figure42: Le compilateur GWT.....	64
Figure 4324: Structure de l'arbre AST GWT .....	66
Figure44: Vue générale du service RPC.....	67

## Introduction générale

Les technologies de l'information et de communication TIC ont été introduites dans le système éducatif tunisien afin, d'une part, de diffuser la culture numérique auprès des jeunes de la société et, d'autre part, de faciliter l'apprentissage des connaissances par les jeunes apprenants, en développant des contenus pédagogiques adoptés.

La diffusion des TIC dans le système de l'éducation a touché toutes les composantes : infrastructure, équipements, ressources humaines, développement de contenu.

Les efforts ont débuté par l'équipement des établissements éducatifs. L'objectif poursuivi : en premier lieu équiper les écoles, les collèges et les lycées par des laboratoires d'informatique.

En deuxième lieu connecter les établissements scolaires au service Internet : cette prestation a été introduite progressivement dans les établissements, permettant ainsi de couvrir la quasi-totalité des réseaux éducatifs.

Autant que l'utilisation de l'Internet est bien généralisée, autant qu'elle présente un risque potentiel sur l'intégrité et la confidentialité de l'ensemble des données ainsi que les ressources matérielles et logicielles de l'établissement.

L'institut nationale de bureautique et de micro-informatique(INBMI) est chargé de concrétiser les choix du Ministère de l'éducation et de la formation dans le domaine des TIC, et afin de protéger les réseaux contre les accès à caractères destructifs, cherche à contrôler les accès internet à partir des serveurs installés dans les établissements.

Dans ce cadre, il nous a été proposé de créer une application web qui permet la gestion des établissements et services offerts aux utilisateurs notamment le service mail et l'accès internet personnalisé.

Après certaines recherches préliminaires et grâce à des informations collectées nous avons élaboré notre vision de l'application, son architecture, ainsi que les outils et les technologies utiles. Dans ce contexte, le travail réalisé sur cinq chapitres principaux dont on parle : dans le premier sur l'organisme d'accueil, dans le deuxième sur l'état de l'art, dans le troisième nous allons étudier les besoins de la solution adoptée et identifier les acteurs ainsi que leurs cas d'utilisations ensuite nous allons entamer la partie conception détaillée et en fin nous allons présenter notre environnement de développement permettant de réaliser notre application.



# Chapitre 1 : Présentation générale du projet

---

# 1 Présentation du projet

Le Réseau Educatif Tunisien connecte aujourd'hui l'ensemble des établissements éducatifs ainsi que l'ensemble des institutions relevant du Ministère de l'éducation et de la formation à savoir les établissements éducatifs, les centres de formation professionnelle, les CREFOC, Les DREF, CENAFFIF.

Vue l'évolution des technologies de l'information, l'émergence des services Internet et Intranet et le nombre assez important des utilisateurs de ces services, la Ministère de l'éducation et de la formation s'oriente vers la construction et le développement des applications assurant les besoins de contrôle, de sécurité et de confidentialité; protégeant les réseaux contre les accès à caractères non éducatifs ou non administratifs ou à caractères destructifs.

Dans ce cadre le présent projet consiste à :

- ✓ Réaliser une application Web qui permet de gérer les comptes mail et internet des employés du ministère,
- ✓ Concevoir et développer un module de centralisation des journaux d'accès Internet dans l' FSI de l'INBMI et exploitation de ces journaux.

## 2 Présentation de l'organisme d'accueil

L'Institut Nationale de Bureautique et de Micro-informatique (INBMI) est l'unique fournisseur de service Internet pour le domaine de l'éducation et de la formation, contribue au développement du réseau éducatif national en assurant l'interconnexion des établissements au réseau internet, et aux réseaux numériques éducatifs.

L'INBMI permet aux établissements, aux enseignants et au personnel du ministère de l'éducation et de la formation de se connecter à Internet en utilisant plusieurs types de connexion offert par l'opérateur de télécommunication (RTC mono poste ou réseau, LS, ADSL) et ce selon leurs demandes ou selon des programmes nationaux de connexions des établissements aux réseaux national éducatif.

### 2.1 Le rôle de l'INBMI en tant que FSI

L'INBMI fournit une plate-forme pour la messagerie électronique à la disposition des utilisateurs du réseau éducatif et administratif du ministère.

L'INBMI offre un service d'hébergement des sites Web.

Un serveur FTP est mis à la disposition des utilisateurs et ce pour l'envoi de fichiers.

L'équipe FSI assure le suivi, la fiabilité, la disponibilité et la bonne qualité des services offerts.

L'équipe FSI est l'interlocuteur unique des utilisateurs du réseau éducatif concernant toutes les questions relatives à leur connexion Internet. En particulier, l'équipe FSI gère l'attribution des plages d'adresses IP et les noms de domaine ainsi que la gestion des lignes de télécommunications.

## 2.2 Architecture des réseaux des établissements

L'usage pédagogique d'Internet nécessite des connexions de longue durée, voire permanentes, à partir de nombreux postes de travail auxquels accèdent des élèves, des enseignants et des cadres administratifs. Cela suppose de mettre en place des dispositifs adaptés :

- interdire les intrusions
- filtrer l'accès aux sites sensibles
- utiliser au mieux le débit disponible

Les problèmes posés par un usage scolaire, particulièrement en connexion permanente sont les suivants :

- Résoudre les problèmes de sécurité.
- Faciliter l'accès aux documents en ligne.
- Interdire l'accès aux sites dangereux.

Diverses solutions commerciales existent. L'INBMI a décidé d'adopter une solution développée dans l'académie de Grenoble et déployée dans des centaines d'établissement, le SLIS (serveur Linux pour l'Internet scolaire). Il s'agit d'un logiciel implanté sur une machine interposée entre l'arrivée d'Internet dans l'établissement et les machines vers lesquelles elle va distribuer l'accès à Internet, de façon sécurisée.

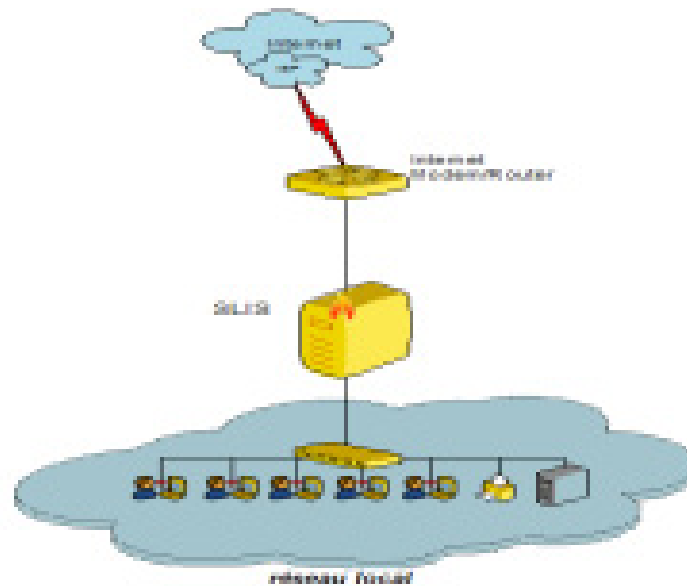


Figure 25: Architecture réseau d'un établissement

### 2.3 Architecture de base : les serveurs SLIS

Le serveur SLIS est un serveur logiciel mis en œuvre, à l'origine, par l'équipe de développeurs du CARMI de l'académie de Grenoble (chef de projet Bruno Breznik). Il fonctionne en environnement Linux et est distribué sous licence GPL. Installé à l'entrée du réseau d'une école, ce serveur joue le rôle de connecteur de l'école à l'internet en proposant un environnement entièrement sécurisé. Le SLIS fournit la sécurité, les ressources **d'archivage** et la gestion des paramètres personnels de chaque utilisateur du réseau.

Le SLIS propose également les fonctionnalités de base d'un serveur complet qui lui permettent d'être exploité dans un cadre pédagogique.

Le serveur SLIS inclut dans sa configuration de base :

- Un serveur de fichiers (Samba),
- **Un serveur proxy (Squid)**,
- Un serveur web (apache)
- **Un serveur de bases de données (Postgres)**.

L'architecture actuelle du réseau des établissements éducatifs rassemble et relie l'ensemble des SLIS, via Internet, à un site central situé dans les locaux de l'INBMI.

Ce serveur central appelé SLIM (SLIS Management) a pour rôle :

- De relayer les informations utiles à la mise à jour des SLIS des écoles (mise à jour logicielle, mise à jour de sécurité, black-List des sites Internet interdits...). Chaque SLIS est synchronisé avec le SLIS central une fois par jour (synchronisation programmée le plus souvent la nuit).

- D'agrèger des statistiques sur le fonctionnement des SLIS (tableau de bord).
- De gérer les alertes lors de la défaillance d'un SLIS du réseau.
- De permettre l'administration et la prise de commande à distance des SLIS (maintenance).

Le SLIS est entièrement administrable à distance.

### 2.4 Emplacement dans le réseau

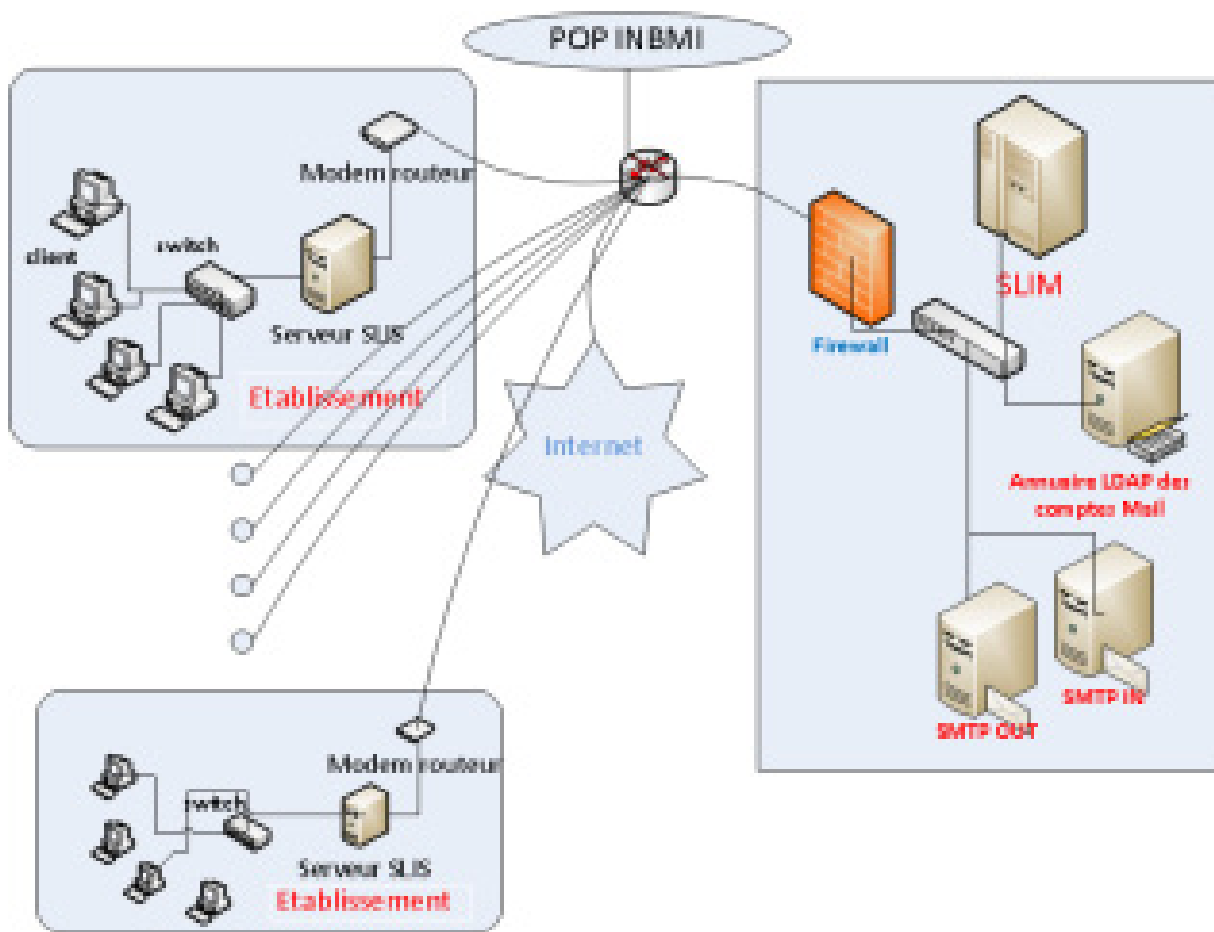


Figure 26: Schéma d'un réseau avec passerelle SLIS

Le SLIS se trouve juste après le routeur qui est connecté par Internet et toutes les informations venant ou allant vers Internet doivent passer par lui. Pour cela il est équipé de deux cartes réseaux qui ne possèdent pas les mêmes plages d'adresses.

Une fois que le SLIS est installé sur le réseau, tous les postes clients doivent être reconfigurés pour que les navigateurs Internet utilisent le SLIS comme passerelle. Cette procédure est très rapide et ne doit être effectuée qu'une seule fois. Par la suite, le SLIS transmettra automatiquement aux postes tous les changements dans la politique d'accès à Internet : services autorisés, sites interdits...

Cette passerelle est réellement indispensable dans un établissement scolaire car elle permet de soulager les professeurs de la lourde tâche de surveillance des élèves pour qu'ils n'aillent pas sur des sites "douteux". La protection du réseau apportée par cette solution est également une grande aide pour l'administrateur du réseau.

L'architecture de la solution de messagerie de l'INBMI est composée principalement de 3 serveurs :

- **Serveurs LDAP Master** : contenant toutes les informations concernant les comptes Internet et la messagerie
- **SMTP-IN**: Reçoit les emails provenant des utilisateurs, les envoie aux serveurs Mail Store et assure les services d'antispam et d'antivirus.
- **SMTP-OUT** : Reçoit les emails des clients, les achemine vers les serveurs de messageries de l'ATI et assure les services d'antispam et d'antivirus.

### 3 Conclusion

Ce chapitre nous a permis d'introduire notre projet et de présenter l'organisme d'accueil; l'étude de l'état et l'environnement de développement seront décrites dans le chapitre suivant.

# Chapitre 2 : Etat de l'art

## Introduction

Ce chapitre présente un état des lieux : il s'agit d'une étude de l'existant suivie de critique permettant au projet de présenter une amélioration résumant l'ensemble des solutions retenues.

## 1 Etude de l'existant

L'étude de l'existant est une phase importante pour bien comprendre le système actuel et définir ses objectifs. Le service informatique de l'INBMI est responsable de gérer les serveurs Slis, gérer les utilisateurs de ce serveurs et de gérer les mails et logs des internautes (enseignant, élève, cadre administratif, etc.).

Ce service utilise l'interface d'administration fournit par SLIM pour :

- ✓ Ajouter un serveur Slis et gérer la disquette de configuration
- ✓ Consulter des informations sur les serveurs

Ainsi que pour consulter les logs des serveurs il est indispensable de télécharger le fichier log qui est un fichier texte plat et le visualiser ;

De plus l'application contenant les comptes mails et internet des utilisateurs n'est pas synchronisée avec les serveurs SLIS donc les utilisateurs sont ajouté à la main.

## 2 Critiques de l'existant

L'étude de l'existant nous a permis de dégager un certain nombre de lacunes :

D'une part l'outil d'administration fournit par SLIM, bien qu'il est puissant ne satisfait pas tous les besoins nécessaires pour la gestion des serveurs Slis. En effet il' est utilisé juste pour générer la disquette de configuration. Le suivi des adresses IP des établissements est réalisé sur un fichier Excel puisque SLIS ne gère pas la langue arabe et difficile à manipuler.

D'autre part les fichiers logs téléchargés ne sont pas lisibles, un fichier texte plat trop volumineux et difficile à interroger.

De plus si le serveur Slis est afonctionnel et nécessite une mise au point, on risque de perdre les anciens fichiers logs.

L'application de gestion des utilisateurs et comptes mails n'est pas optimisé pour synchroniser avec le serveur SLIM et les serveurs SLIS.



### 3 Présentation du travail demandé

Dans un souci de concevoir une application avec plus de fonctionnalités possibles et dans le but d'avoir une interface plus conviviale et plus facile à utiliser tout en étant plus efficace, nous avons conçu une application qui regroupe tous les points cités ci-dessus. Le travail demandé se résume ainsi dans les fonctionnalités suivantes :

- Offrir à l'utilisateur une interface de gestion des abonnés et les comptes mails correspondant
- Offrir à l'utilisateur une interface permettant la gestion des établissements et des serveurs Slis.
- Donner à l'utilisateur la possibilité d'ajouter les comptes Internet directement au serveur SLIS de l'établissement et consulter l'historique de navigation des internautes utilisant un tel serveur.
- Permet à l'administrateur de contrôler les accès à chaque module de l'application.

### 4 Model et méthodologie adoptés

#### 4.1 Model

UN modèle de développement logiciel désigne toutes les étapes du développement, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, c'est-à-dire la conformité du logiciel avec les besoins exprimés, et la vérification du processus de développement. L'origine de ce découpage provient du constat que les erreurs ont un coût d'autant plus élevé qu'elles sont détectées tardivement dans le processus de réalisation. Le cycle de vie permet de détecter les erreurs au plus tôt et ainsi de maîtriser la qualité du logiciel, les délais de sa réalisation et les coûts associés.

À ce fait nous adoptons le modèle de cycle de vie en V qui part du principe que les procédures de vérification de la conformité du logiciel aux spécifications doivent être élaborées dès les phases de conception.

L'énorme intérêt du cycle en V est qu'il est un excellent support à la formalisation de notre relation avec le futur-utilisateur, Il nous oblige à réfléchir aux différents aspects de sa demande. La phase de spécification nous permet de vérifier qu'on a bien compris la demande des utilisateurs, en effet, c'est l'équipe de gestion des services Internet de l'INBMI qui valide la spécification.

## 4.2 Méthodologie

Le plus grand avantage d'une méthode orientée objet est qu'elle permet de structurer un système sans centrer l'analyse uniquement sur les données ou uniquement sur les traitements mais sur les deux à la fois. Une telle approche a pour but de modéliser les propriétés statiques et dynamiques de l'environnement du système. Elle met en correspondance le problème et la solution, en préservant la structure et le comportement du système analysé.

Ceci, nous a conduit à adopter l'approche orientée objet pour modéliser notre système en se basant sur les diagrammes UML.

## 5 Choix des outils de développement

Pour la réalisation de ce projet nous avons choisi de travailler avec :

- NetBeans comme IDE ;
- PostgreSQL comme SGBD ;
- GXT comme Framework coté Web qui basé sur GWT ;
- J2EE comme langage de programmation.
- Ireport pour génération et l'exportation au format Word et PDF de rapport



### 5.1 NetBeans

C'est un environnement de développement intégré (EDI) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS.



### 5.2 GWT

Abréviation du Google Web Toolkit ; GWT est un ensemble d'outils logiciels développé par Google, permettant de créer et maintenir des applications web dynamiques mettant en œuvre JavaScript, en utilisant le langage et les outils Java. C'est un Framework libre distribué selon les termes de la licence Apache 2.0.

GWT met l'accent sur des solutions efficaces et réutilisables aux problèmes rencontrés habituellement par le développement AJAX : difficulté du débogage JavaScript, gestion des appels asynchrones, problèmes de compatibilité entre navigateurs, gestion de l'historique et des favoris, etc.

GWT est articulé autour d'un concept original : lors de la phase de développement, l'application est écrite en Java de façon classique, dans un environnement de développement intégré Java, et peut être déboguée avec les outils Java habituels. Une fois l'application prête à être déployée, le compilateur GWT la traduit en pur JavaScript, avec support automatique et transparent pour les principaux navigateurs (Internet Explorer, Firefox, Mozilla , Safari , Opera ).

Le code JavaScript généré utilise des techniques d'HTML dynamique et de manipulation du DOM (Document Object Model) pour les aspects dynamiques de l'interface.

Ce principe est rendu possible par les différents composants de GWT:

- le compilateur Java vers JavaScript
- un navigateur spécialement modifié pour permettre l'exécution (et le débogage) de code Java natif sans nécessiter la compilation JavaScript
- une bibliothèque d'émulation JRE : il s'agit d'une implémentation en JavaScript d'un sous-ensemble de la bibliothèque de classes Java standard (en particulier quasiment tout le package `java.lang` et une partie de `java.util`)
- une bibliothèque de composants graphiques contenant des widgets de base permettant la construction d'une interface graphique

GWT est souvent appelé abusivement un Framework, mais n'en est pas véritablement un car il impose peu de choses au développeur; comme son nom l'indique, il s'agit d'une boîte à outils qui offre des solutions permettant de développer plus facilement des solutions web/AJAX de dernière génération, en profitant des outils et compétences Java existants, et en faisant abstraction de la complexité habituellement liée à ce genre de technologies.



### 5.3 PostgreSQL

PostgreSQL est un système de gestion de base de données relationnelle et objet(SGBDRO). C'est un outil libre disponible selon les termes d'une licence de type BSD.

Ce système est concurrent d'autres systèmes de gestion de base de données, qu'ils soient libres (comme MySQL et Firebird), ou propriétaires(comme Oracle, Sybase, DB2et Microsoft SQL Server).

### 5.4 GXT

Le vrai nom est EXT-GWT; basé sur le Framework GWT, c'est une bibliothèque développée par l'entreprise ExtJs leader du développement des bibliothèques JavaScript ; GWT-Ext est une librairie de composant graphique, elle fait le pont entre GWT et Ext.

On trouve dans GXT un ensemble de widgets permettant de construire une interface utilisateur avec des outils variés comme les:

- 1) Panneaux
- 2) Boutons
- 3) Cases à cocher
- 4) Tables / Grilles
- 5) Boîtes de dialogues
- 6) Primitive HTML (dont les images et les hyperliens)
- 7) Menus et barres de menus
- 8) Fenêtres défilantes
  - 9) Onglets
  - 10) Arbres



### 5.5 IReport

iReport est un outil de conception WYSIWYG (What You See Is What You Get) exclusivement réservé à la création de fichier de description pour JasperReports.

Il permet donc de produire de manière assez intuitive des fichiers .jrxml (fichiers XML) exploitables par JasperReports pour générer des rapports au sein d'une application Java. Le format de rapport généré dépend ensuite de JasperReports et du code utilisé (html, pdf, csv...).

## 6 Conclusion

Après avoir mis le projet dans son cadre et après avoir mis en place une démarche de développement qui nous aidera tout au long du projet, nous pouvons ainsi entamer la prochaine étape c'est-à-dire décortiquer le cahier de charges pour analyser les besoins qui s'y trouvent et ainsi les spécifier.

# Chapitre 3 : Analyse des besoins et spécification

---

## Introduction

Dans ce chapitre, nous présenterons les objectifs de notre application, ce qui nous amène à identifier les possibilités du système et les besoins des utilisateurs que nous essayerons de les projeter dans des diagrammes de cas d'utilisations globales et détaillés.

# 1 Analyse des besoins

## 1.1 Les besoins attendus de l'application

L'application envisagée satisfaire les besoins fonctionnels qui seront exécutés par le système et les besoins non fonctionnels qui perfectionnent la qualité logicielle du système.

### 1.1.1 Les besoins fonctionnels

Les besoins fonctionnels ou besoin métiers représentent les actions que le système doit exécuter, il ne devient opérationnel que s'il les satisfait.

Cette application doit couvrir principalement les besoins fonctionnels suivants :

- Gestion des établissements et les abonnés mail et internet correspondants
- Attribution de compte mail et internet aux abonnés suivant la catégorie de l'abonnés
- Faire un Workflow de suivi de l'état de création de compte mail avec la validation administrative et technique
- Gestion des Slis des établissements et les utilisateurs correspondants
- La gestion des logs (mise à jour des logs d'un serveur) ;
- La gestion des droits d'accès.

### 1.1.2 Les besoins non fonctionnels

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système mais plutôt identifient des contraintes internes et externes du système.

Les principaux besoins non fonctionnels de notre application ce résumant dans les points suivants :

- Le code doit être clair pour permettre des futures évolutions ou améliorations ;
- L'ergonomie : l'application offre une interface conviviale et facile à utiliser ;
- La sécurité : l'application doit respecter la confidentialité des données ;
- Garantir l'intégrité et la cohérence des données à chaque mise à jour et à chaque insertion.

## 1.2 Résultats attendus

Que l'application se déroule convenablement, de conserver ces fonctionnalités dans l'application, et d'améliorer s'il est possible les performances du système ainsi que les bases des données existantes.

## 2 Les diagrammes des cas d'utilisation

Montre les interactions fonctionnelles entre les acteurs et le système à l'étude

- **Acteur** : rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d'utilisation.
- **Cas d'utilisation (use case)** : ensemble de séquences d'actions réalisées par le système produisant un résultat observable intéressant pour un acteur particulier. Collection de scénarios reliés par un objectif utilisateur commun.
- **Association** : utilisée dans ce type de diagramme pour relier les acteurs et les cas d'utilisation par une relation qui signifie simplement « participe à ».
- **Inclusion** : le cas d'utilisation de base en incorpore explicitement un autre, de façon obligatoire, à un endroit spécifié dans ses enchaînements.
- **Extension** : le cas d'utilisation de base en incorpore implicitement un autre, de façon optionnelle, à un endroit spécifié indirectement dans celui qui procède à l'extension
- **Généralisation** : les cas d'utilisation descendants héritent de la description de leur parent commun. Chacun d'entre eux peut néanmoins comprendre des relations spécifiques supplémentaires avec d'autres acteurs ou cas d'utilisation.

### 2.1 Identification des acteurs

Un acteur représente un rôle joué par une personne qui interagit avec le système.

Par définition, les acteurs sont à l'extérieur du système. Les acteurs se recrutent parmi les utilisateurs du système et aussi parmi les responsables de sa configuration et de sa maintenance. D'où, les acteurs potentiels qui risquent d'interagir avec l'application sont :

- **L'utilisateur (Centre régional)** : C'est un utilisateur régional, il est responsable d'ajouter les comptes mails des utilisateurs faisant partie de la direction régionale. Cet utilisateur ne consulte que certains types d'abonnés (enseignants, tuteur, cadre administratif régional). Cet utilisateur ne valide pas la création de comptes.

- **L'utilisateur responsable mail** : Celui-ci valide la création et la modification mail signalé par le centre régional. Il l'a le droit de consulter toutes la base de donnée mail.
- **L'utilisateur SLIS** : Il ne consulte que les données relatives aux adresses IP des établissements et les comptes Internet des utilisateurs de cet établissement.
- **Le Super-Utilisateur SLIS** : Il a le droit de consulter les logs de tous les internautes des établissements.
- **L'administrateur (admin)** : C'est le gérant de l'application, il a une visibilité totale sur les bases de données. Il a pour tâches de gérer tout le système. Il spécifie les utilisateurs et les droit de chaque' un

## 2.2 Les différents cas d'utilisation

L'étude de cas d'utilisation a pour objectif de déterminer ce que chaque utilisateur attend du système. La détermination du besoin est basée sur la représentation de l'interaction entre l'acteur et le système.

### 2.2.1 Cas d'utilisation préliminaire

**Authentification** : permet d'identifier chaque utilisateur, et de lui donner l'accès aux fonctionnalités propices.

**Gérer les utilisateurs et les droits d'accès** : permet à l'administrateur d'ajouter ou supprimer ou modifier ou consulter un user. De plus chaque utilisateur lui est associé un droit dans cette application.

**Gestion des établissements** : ajout, suppression, modification et consultation

**Gérer les abonnés mail** : permet à l'utilisateur d'ajouter ou modifier un abonné fonctionnant dans un établissement donné.

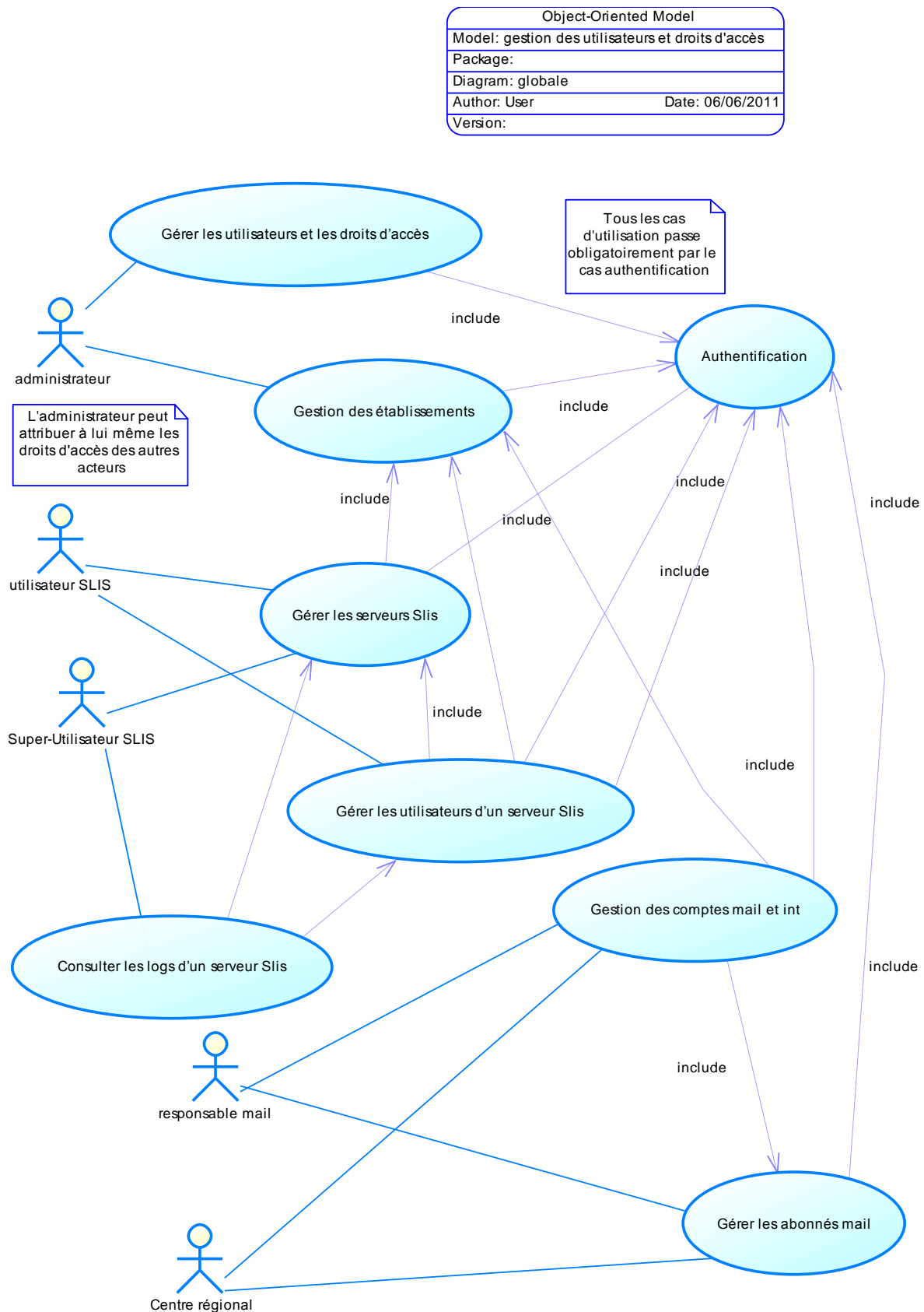
**Gestion des comptes mail et internet** : suivi de l'état de création d'un compte. On identifie dans ce cas 7 états : En attente (dossier incomplet), validé administrativement en attente de création, validé techniquement et créé, en attente de modification, modification validée, en attente de suppression, suppression validée.

**Gérer les serveurs Slis** : permet à l'administrateur d'ajouter, modifier ou supprimer ou consulter le serveur SLIS

**Gérer les utilisateurs d'un serveur Slis** : permet à l'administrateur d'ajouter, modifier ou supprimer un utilisateur d'un serveur Slis.



**Consulter les logs d'un serveur Slis** : permet à l'utilisateur de consulter et ou imprimer les logs d'un serveur Slis.



**Figure 27: cas d'utilisation préliminaire**

## 2.2.2 Cas d'utilisations détaillés

On s'intéresse aux cas d'utilisations les plus pertinents.

### 2.2.2.1 Cas d'utilisation «S'authentifier »

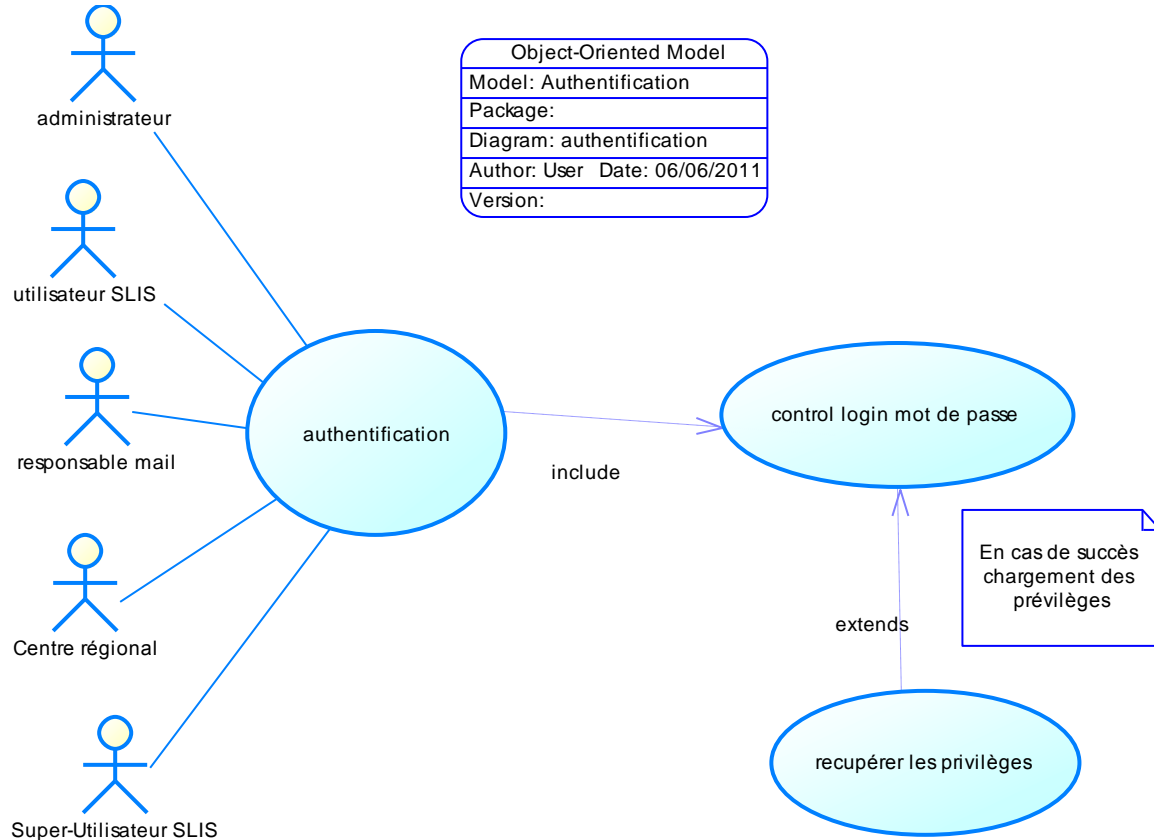


Figure 28 : Cas d'utilisation «S'authentifier »

#### ✓ Description textuelle :

Acteurs primaires : Admin, User-SLIS-Super-User-SLIS , responsable mail, centre régional.

Description : Tous les utilisateurs de l'application peuvent accéder au système. Cependant, chacun d'eux à un certain nombre de privilèges. C'est pour cela, qu'il faut au début s'identifier en donnant son login et son mot de passe et les privilèges seront attribués à l'utilisateur.

#### ✓ Analyse :

On a choisi de commencer par traiter ce cas d'utilisation par ce que c'est le cas qui initialise tous les autres cas d'utilisation.

Une réalisation de ce cas d'utilisation « S'authentifier » se fait comme suit :

L'utilisateur saisie son login et mot de passe sur la page : Authentification

Après vérification des données, le système sélectionne l'utilisateur en cours

Une requête de recherche portant le nom de l'utilisateur se déclenche dans la base de données afin d'afficher le menu général. En cas d'existence de l'utilisateur, le système charge les privilèges attribué précédemment à l'utilisateur.

### 2.2.2.2 Cas d'utilisation « Gérer les utilisateurs et les droits d'accès »

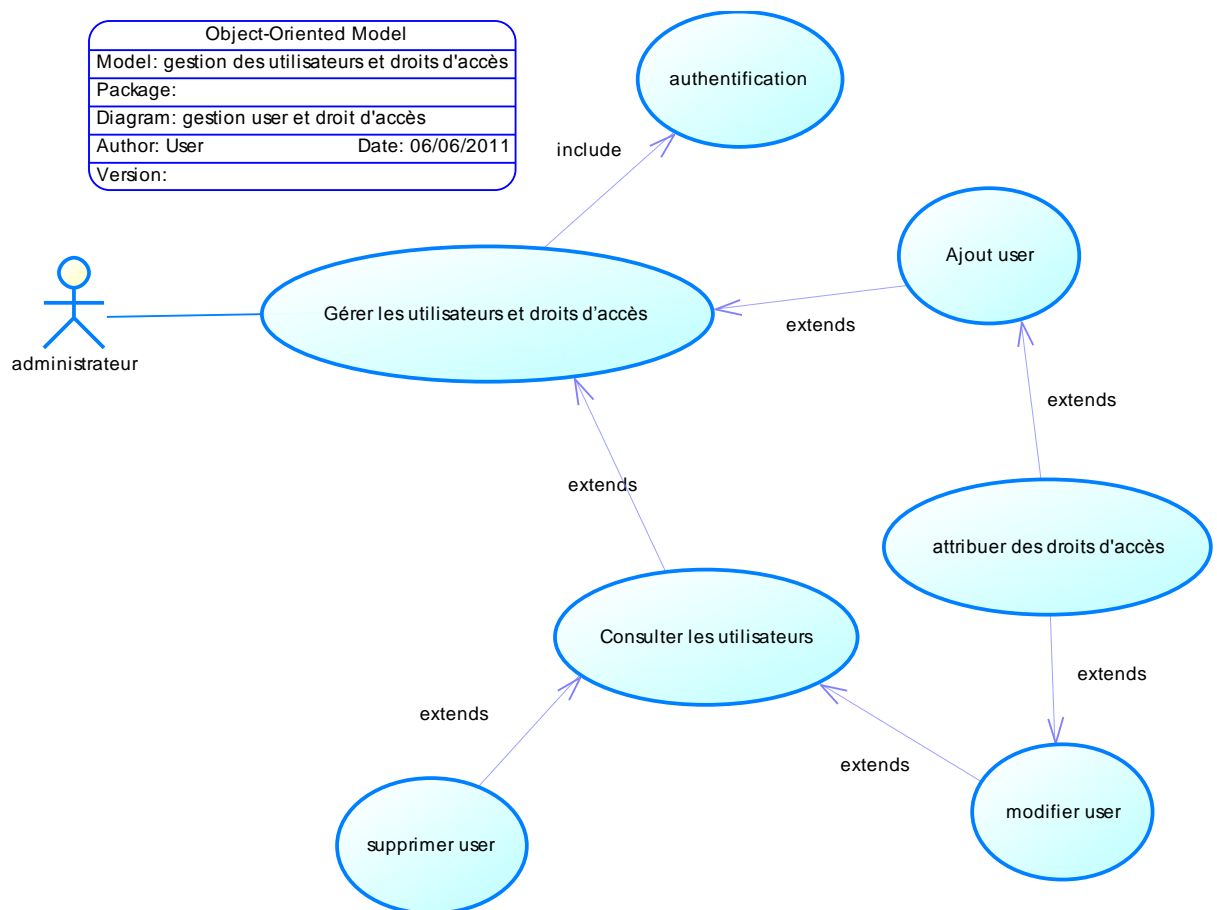


Figure 29 : Cas d'utilisation «Gérer droits d'accès »

#### ✓ Description textuelle :

Cas d'utilisation : **Gérer les utilisateurs et les droits d'accès**

Acteurs primaires : Admin.

Description : après l'authentification l'administrateur peut effectuer la saisie et l'enregistrement de nouveaux utilisateurs, la consultation des utilisateurs avec possibilité de modification, d'attribution de droits d'accès ou de la suppression d'un utilisateur donné.

#### ✓ Analyse :

Une réalisation de ce cas d'utilisation se fait comme suit :

L'administrateur peut effectuer la création d'un utilisateur en lui attribuant les privilèges souhaités.

L'administrateur consulte la liste des utilisateurs, sélectionne un utilisateur pour le modifier ou le supprimer.

On identifie les privilèges suivants :

- Droit Nomenclature : ajout suppression et modification des gouvernorats et établissements
- Droit sélection abonné : consultation d'abonnés mail seulement.
- Droit MAJ abonné : ajout, suppression et modification de compte mail (avant validation)
- Droit validation technique ; validation d'ajout, suppression ou modification de compte mail
- Droit gestion des utilisateurs : ajout suppression modification et consultation des utilisateurs et leurs droits
- Droit gestion SLIS : ajout suppression modification, consultation des serveurs SLIS avec possibilité d'ajouter les utilisateurs, télécharger disquette de config, avoir le mot de passe root pour la connexion ssh ,...
- Droit gestion des logs : consultation des journaux des internautes

2.2.2.3 Cas d'utilisation « Gérer les comptes mail »

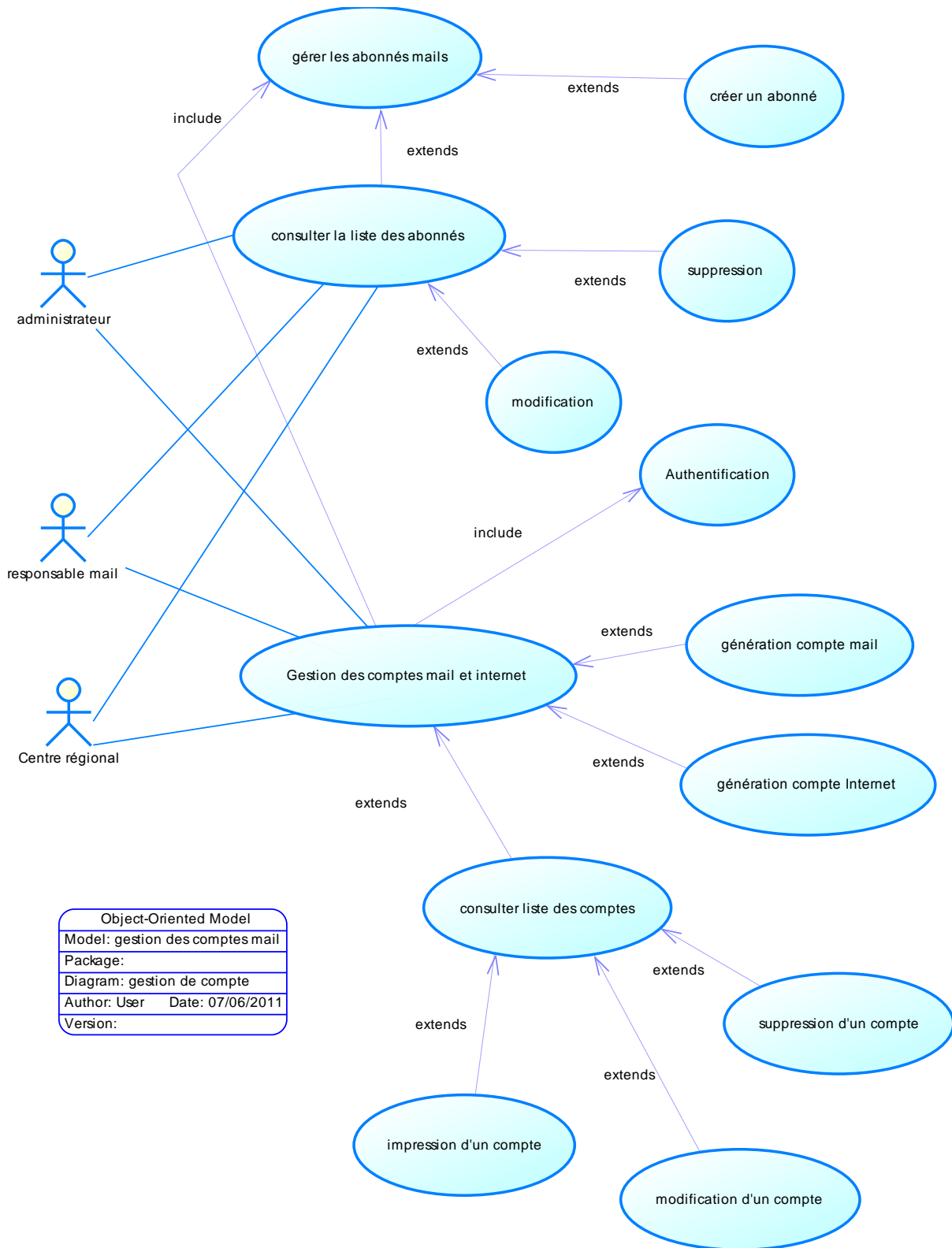


Figure 30 : Cas d'utilisation « Gérer les comptes mail »

✓ Description textuelle :

Cas d'utilisation : Gérer les comptes et internet

Acteurs primaires : Admin, responsable mail, centre régional

Description : Après authentification l'utilisateur crée un abonné mail, lui attribue l'établissement correspondant. Une fois l'abonné est enregistré, le système génère un compte mail en concaténant le nom et le prénom avec le domaine correspondant à l'établissement de l'abonné.

On distingue principalement ces domaines :

inbmi.edunet.tn : pour les employés de l'Inbmi

enseignant.edunet.tn : pour les enseignants

instituteur.edunet.tn : pour les tuteurs

atfp.edunet.tn : pour les formateurs de la formation professionnelle

minedu.edunet.tn : pour les employés au ministère

✓ **Analyse :**

Une réalisation de ce cas d'utilisation se fait comme suit :

L'utilisateur choisit un établissement ; crée un abonné, génère le compte correspondant.

Le compte ainsi créé est à l'état en attente de création.

Le responsable consulte la liste des comptes en attente de création et valide la création de ces comptes pour qu'ils soient opérationnels.

De même la suppression et la modification de comptes n'est validé que par le responsable mail.

Le centre régional signale la création, la modification ou la suppression d'un compte mail sans valider ces actions

2.2.2.4 Cas d'utilisation « Consulter les logs d'un serveur Slis »

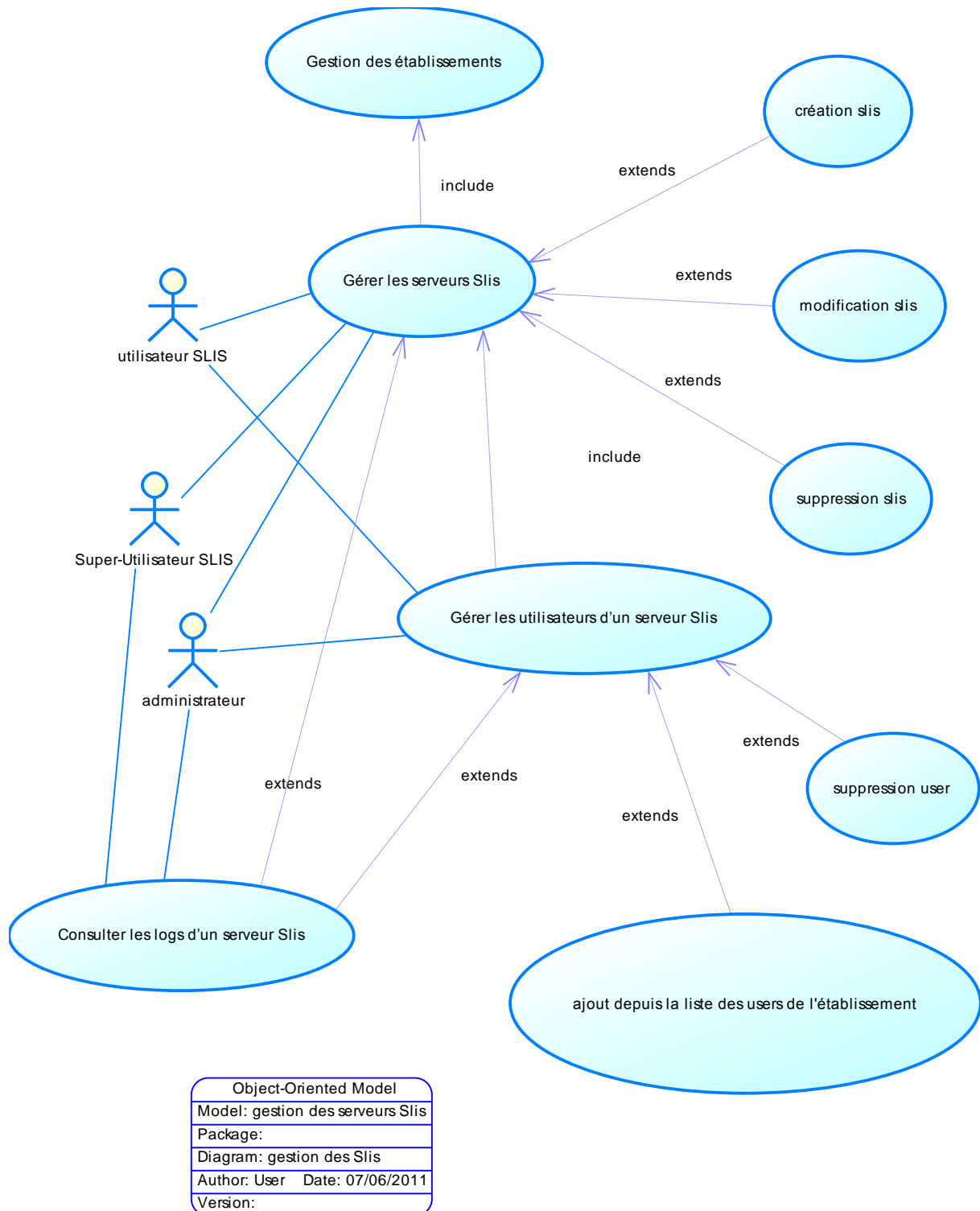


Figure 31 : Cas d'utilisation « gestion des serveurs Slis »

✓ Description textuelle :

Cas d'utilisation : Gestion des serveurs Slis

Acteurs primaires : Admin, Super-user Slis, user Slis

Description : après l'accès à l'application, l'utilisateur sélectionne un établissement et ajoute un serveur Slis. S'il existe déjà il peut le modifier ou le supprimer.

Pour que le personnel de l'établissement puisse accéder à Internet, l'utilisateur charge les comptes des utilisateurs dans le serveur Slis. Il peut ensuite voir les journaux d'accès Internet de tout l'établissement ou un utilisateur donné.

### ✓ Analyse :

Une réalisation de ce cas d'utilisation se fait comme suit :

L'administrateur ou l'utilisateur choisit un établissement crée un serveur Slis avec l'adresse ip adéquate puis sélectionne les utilisateurs de cet établissement et les insère dans le serveur Slis.

Le super user Slis peut consulter ou imprimer les logs des internautes utilisant ce serveur.

## 3 Conclusion

Ce chapitre a été consacré pour la spécification des besoins fonctionnels et non fonctionnels du système résultant, ce qui correspondait aux différentes activités de la première phase du cycle de développement du notre système. Dans le chapitre suivant, nous étudierons la phase de conception.



# Chapitre 4 : Conception

---

## **Introduction**

Vu que nous avons achevé la première phase (Démarrage) du cycle de développement, nous aborderons dans ce chapitre la deuxième phase (Elaboration) qui se concentre essentiellement sur la définition de l'architecture du système ainsi que sur l'analyse et la conception des besoins et des exigences des utilisateurs.

L'activité d'analyse et de conception permet de traduire les besoins fonctionnels et les contraintes issues du cahier des charges et de la spécification des exigences dans un langage plus professionnel et compréhensible par tous les individus intervenants dans la réalisation et l'utilisation de l'application

## **1 Architecture de l'application**

Dans les phases préliminaires du développement d'une application ou de la refonte d'un système d'information, la définition de l'architecture technique consiste à faire les choix de technologies et d'organisation de composants logiciels les plus adaptés aux besoins et aux contraintes de l'organisation d'accueil.

Ces choix sont ensuite relayés au sein de notre projet, guidant la conception et permettant la transformation d'un modèle fonctionnel en application performante et robuste.

### **1.1 Présentation de l'architecture à 2 niveaux**

L'architecture à deux niveaux (aussi appelée architecture 2-tiers, tiers signifiant étages en anglais) caractérise les systèmes clients/serveurs dans lesquels le client demande une ressource et le serveur la lui fournit directement. Cela signifie que le serveur ne fait pas appel à une autre application afin de fournir le service.

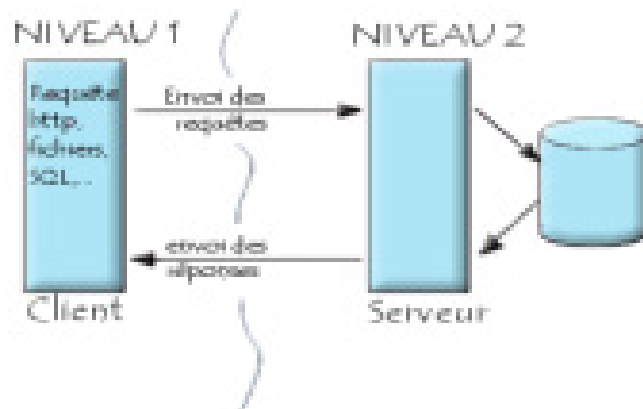


Figure 32: Architecture 2 niveaux

### 1.2 Présentation de l'architecture à 3 niveaux

Dans l'architecture à 3 niveaux (appelées architecture 3-tiers), il existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre:

1. Le client: le demandeur de ressources
2. Le serveur d'application (appelé aussi **middleware**): le serveur chargé de fournir la ressource mais faisant appel à un autre serveur
3. Le serveur secondaire (généralement un serveur de base de données), fournissant un service au premier serveur.

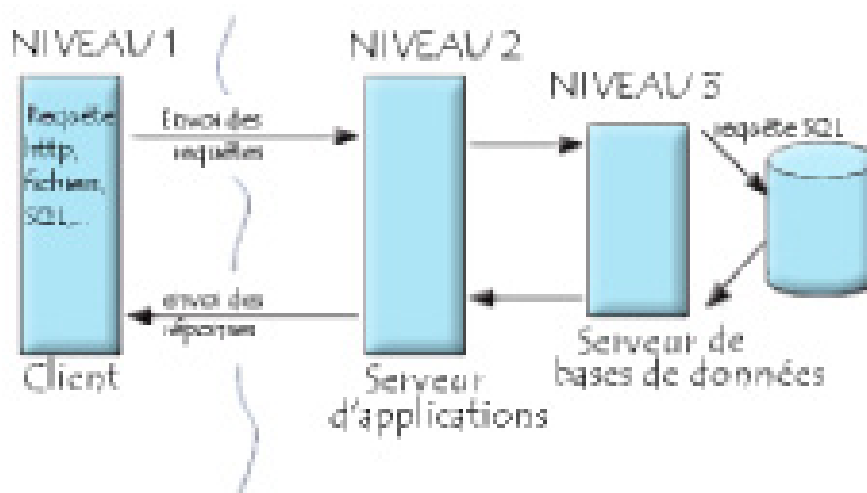


Figure 33: Architecture 3 niveaux

Tout système d'information nécessite la réalisation de trois groupes de fonctions: le stockage des données, la logique applicative et la présentation. Ces trois parties sont indépendantes les unes des autres: on peut ainsi vouloir modifier la présentation sans modifier la logique applicative. La conception de chaque partie doit également être indépendante, toutefois la conception de la couche la plus basse est utilisée dans la couche d'au-dessus. Ainsi la conception de la logique applicative se base sur le modèle de données, alors que la conception de la présentation dépend de la logique applicative.

### 1.3 Architecture adoptée

Vis-à-vis l'existant de l'INBMI: organisation, compétences, architecture du système d'information, nous avons choisi l'architecture 3 tiers car c'est une architecture :

- **pérenne**: applicable durant une très longue période de temps et accepter des changements technologiques ou fonctionnels tout en protégeant les investissements réalisés.
- **modulaire**: un élément peut être remplacé ou modifié sans devoir changer toute l'architecture.
- **ouverte**: elle doit permettre de construire ou de modifier une solution à partir de composants provenant de différents constructeurs.

## 2 Conception détaillées

UML est une approche orientée objet de modélisation qui permet de modéliser un problème d'une manière standard.

UML évite de se définir comme une méthodologie, comme son nom l'indique, c'est un langage « visuel » qui permet d'exprimer la compréhension d'un système : il comporte 9 principaux diagrammes regroupés dans deux vues différentes:

- ✓ **Vue Statique** (cinq diagrammes structurels)
  - Diagramme de Cas d'utilisation.
  - Diagramme de Classes.
  - Diagramme d'Objets.
  - Diagramme de Composants.
  - Diagramme de Déploiement.
- ✓ **Vue Dynamique** (quatre diagrammes comportementaux)
  - Diagramme de Séquence.

- Diagramme d'activités.
- Diagramme d'états transitions.
- Diagramme de Collaboration.

A cet effet on présente quelques diagrammes de modélisation, qu'on a jugé les plus importants pour la compréhension du fonctionnement du système.

## **2.1 Vue statique**

### **2.1.1 Diagramme des cas d'utilisation**

Les diagrammes de cas d'utilisation représentent toutes les interactions des utilisateurs avec le système, comme nous avons déjà les décrit dans le chapitre précédent "Analyse des besoins et spécification"

### **2.1.2 Diagramme de classes**

Le diagramme de classes représente les classes constituant le système et les associations entre elles.

Les diagrammes de classes expriment de manière générale la structure statique d'un système, en termes de classe et de relations entre ces classes.

De même qu'une classe décrit un ensemble d'objets, une association décrit un ensemble de liens ; les objets sont des instances de classes et les liens sont des instances de relations.

# Conception et réalisation d'une application de gestion des comptes mail et internet

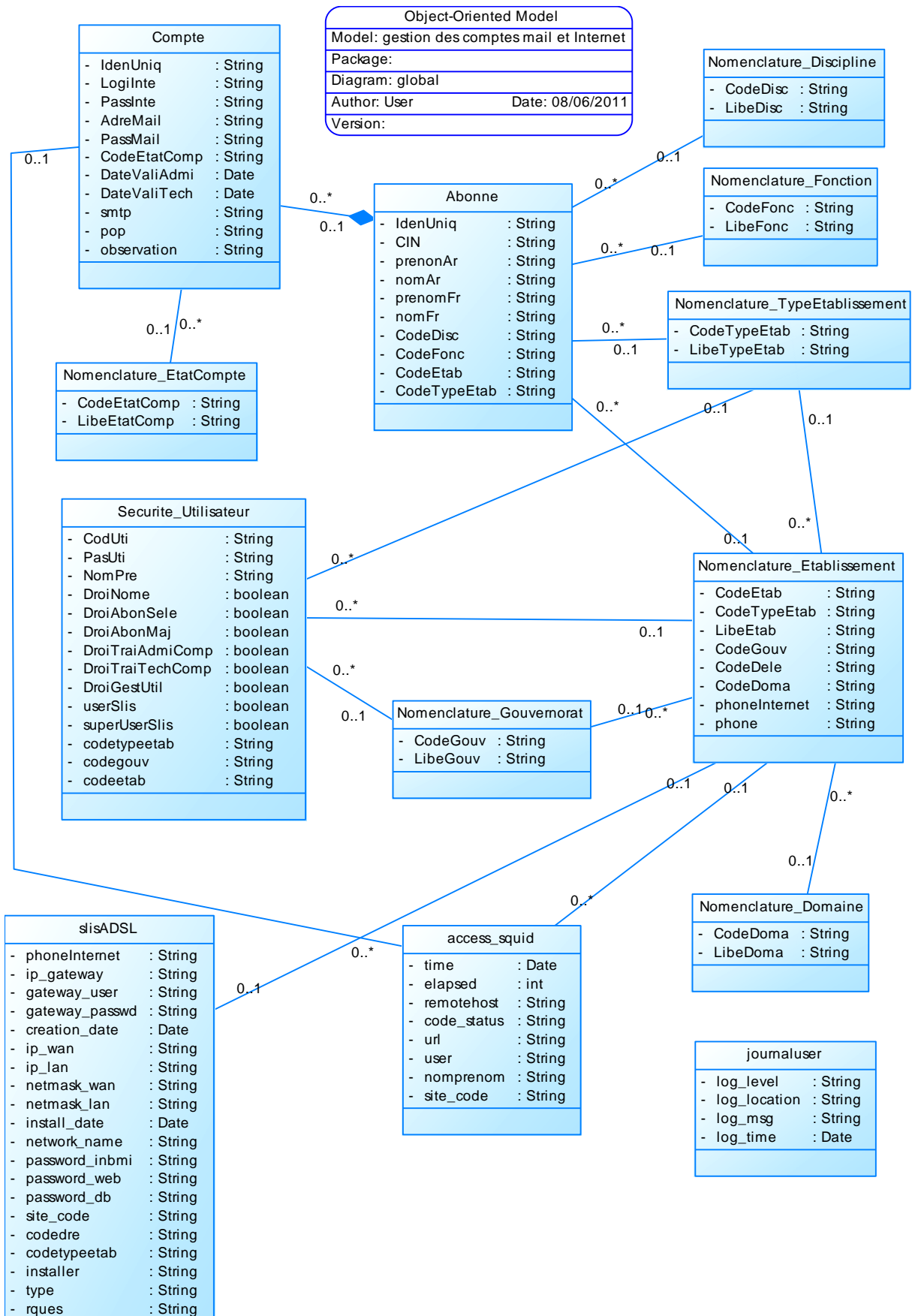


Figure 34 : diagramme de classes

D'après l'étude du système existant et des différents diagrammes de cas d'utilisation, nous avons pu dégager les principales classes illustrées dans la figure ci-dessus pour avoir une vue plus claire du système étudié.

A partir de ce diagramme, on dégage les entités de la base de données ainsi que les java beans correspondant dans l'application à développer.

Ce diagramme de classe ne contient pas des méthodes. Ces procédures seront évoquées après le diagramme de séquence.

Le mapping objet relationnel sera décrit dans la partie réalisation.

### 2.1.3 Diagramme de déploiement

Le diagramme de déploiement est une vue statique qui sert à représenter l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis ainsi que leurs relations entre eux.

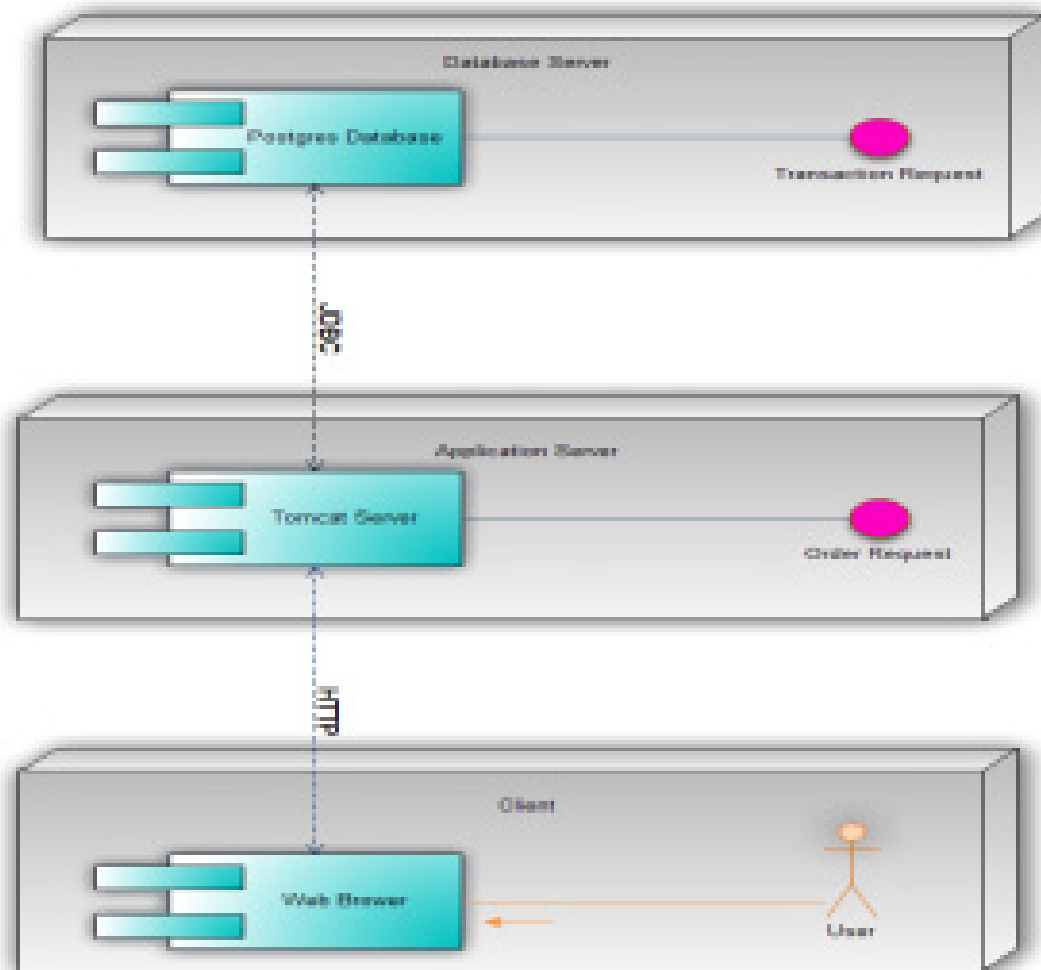


Figure 35 : diagramme de déploiement

## 2.1.4 Vue dynamique : Diagramme de séquence

Montre la séquence verticale des messages passés entre objets au sein d'une interaction

Ligne de vie : représentation de l'existence d'un élément participant dans un diagramme de séquence. Cela peut être un acteur ou le système en modélisation d'exigences, des objets logiciels en conception préliminaire ou conception détaillée.

Message : élément de communication unidirectionnel entre objets qui déclenche une activité dans l'objet destinataire. La réception d'un message provoque un événement dans l'objet récepteur. La flèche pointillée représente un retour au sens UML. Cela signifie que le message en question est le résultat direct du message précédent.

Après l'étude des cas d'utilisation, nous avons pu dégager les diagrammes de séquences correspondants dont voici les plus importants :

### 2.1.4.1 Diagramme de séquence du cas « authentification »

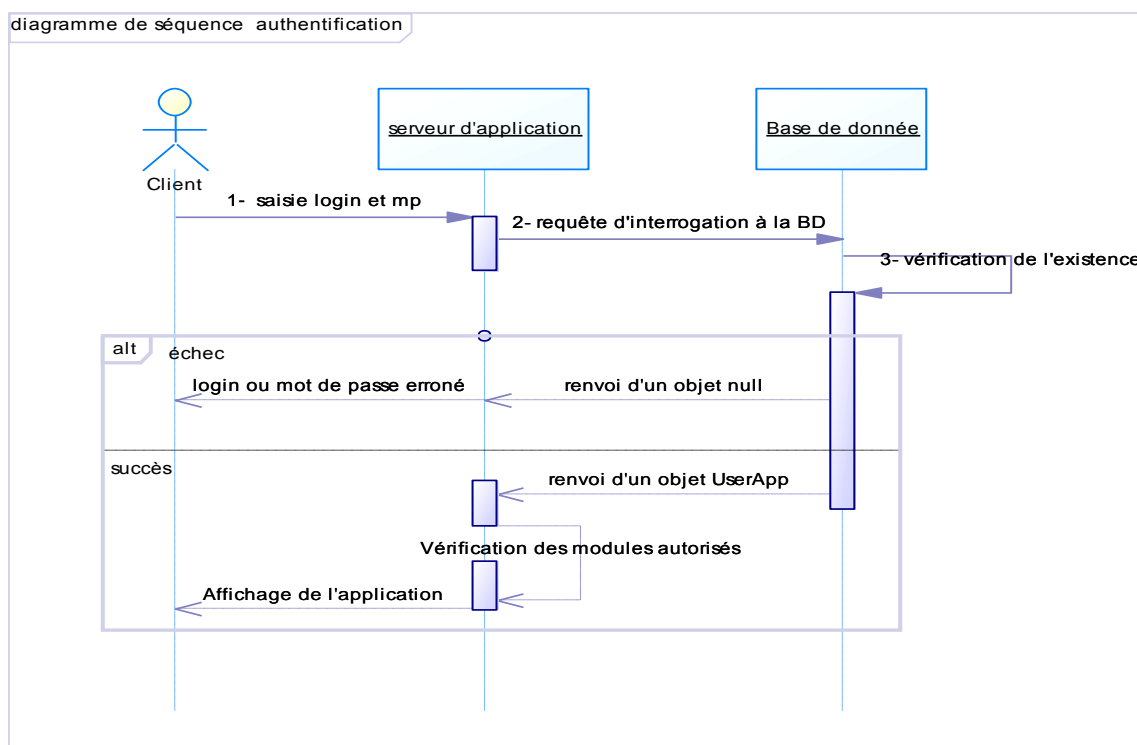


Figure 36 : diagramme de séquence du cas "authentification"

Lorsque l'utilisateur demande l'accès à l'application, il doit tout d'abord s'identifier par son login et mot de passe via le serveur d'application qui prend en charge de vérifier et consulter



la base de données.

S'il est accepté, donc il y'aura l'accès au système et aux applications du menu correspondant.

Sinon, le serveur d'application lui affiche un message d'erreur afin de rectifier ses données.

### 2.1.4.2 Diagramme de séquence du cas « Gérer les utilisateurs et les droits d'accès »

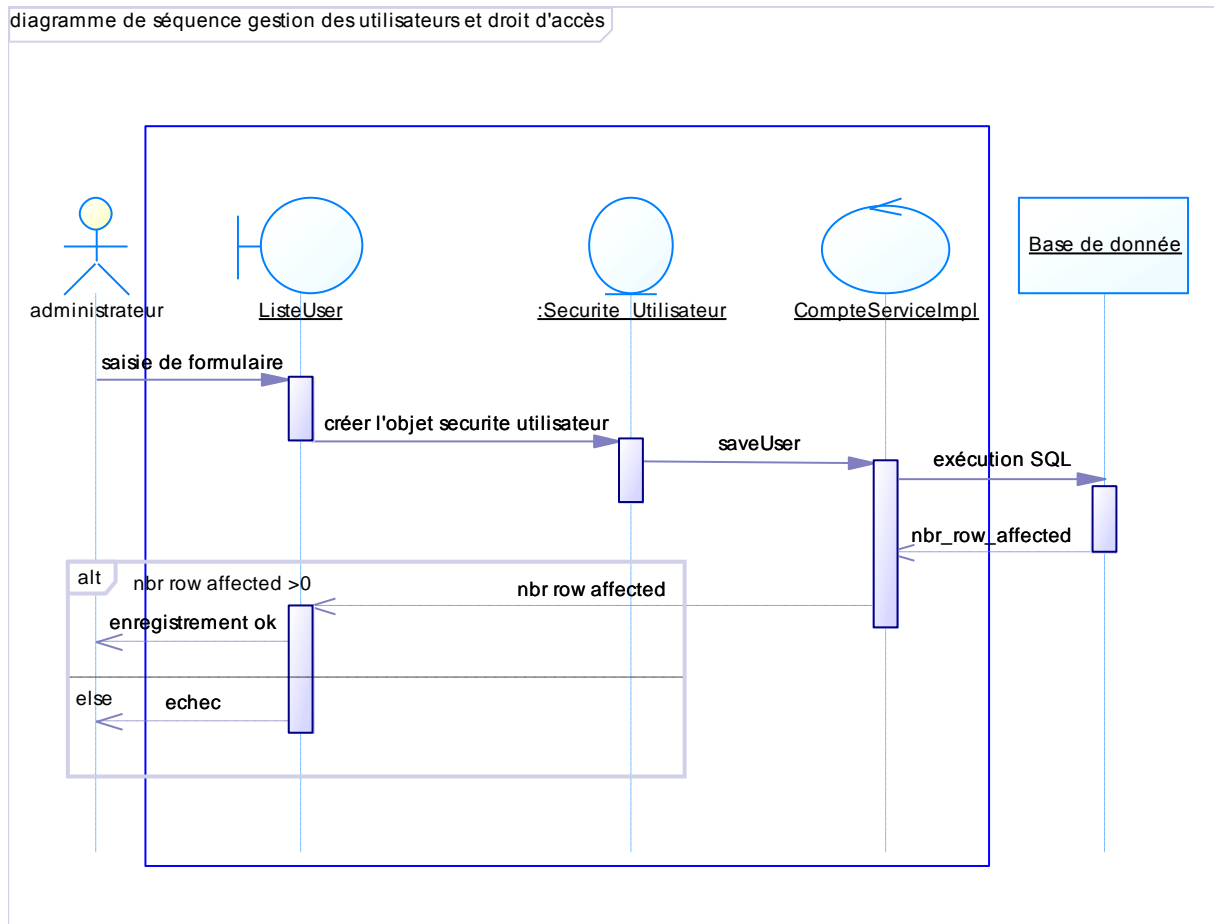


Figure 37 : diagramme de séquence du cas " Gérer les utilisateurs et les droits d'accès"

Lorsque l'utilisateur accède à l'application, il peut :

D'une part Saisir les données concernant le nouvel utilisateur puis demander l'enregistrement via le serveur d'application qui prend en charge la vérification du type des données et l'insertion dans la base de données. Si l'insertion est réalisée avec succès un message informe l'utilisateur que l'enregistrement est réalisé avec succès. Si non un message d'erreur sera affiché.

D'autre part consulter la liste des utilisateurs.

## Conception et réalisation d'une application de gestion des comptes mail et internet

L'administrateur demande la liste des utilisateurs. Une requête sera passée à la base de données puis la classe CompteServiceImpl prépare une liste des objets utilisateurs qui sera chargé par le composant GridUser de la classe ListeUser.

Une fois la liste est affichée, l'utilisateur peut sélectionner un utilisateur pour le modifier ou le supprimer.

La modification génère la boîte de dialogue NewUser avec l'utilisateur sélectionné en paramètre pour lister les informations de l'utilisateur sélectionné. Une fois les modifications sont effectuées, l'objet utilisateur modifié sera passé pour exécuter la requête SQL correspondante à la BD.

De même pour la suppression.

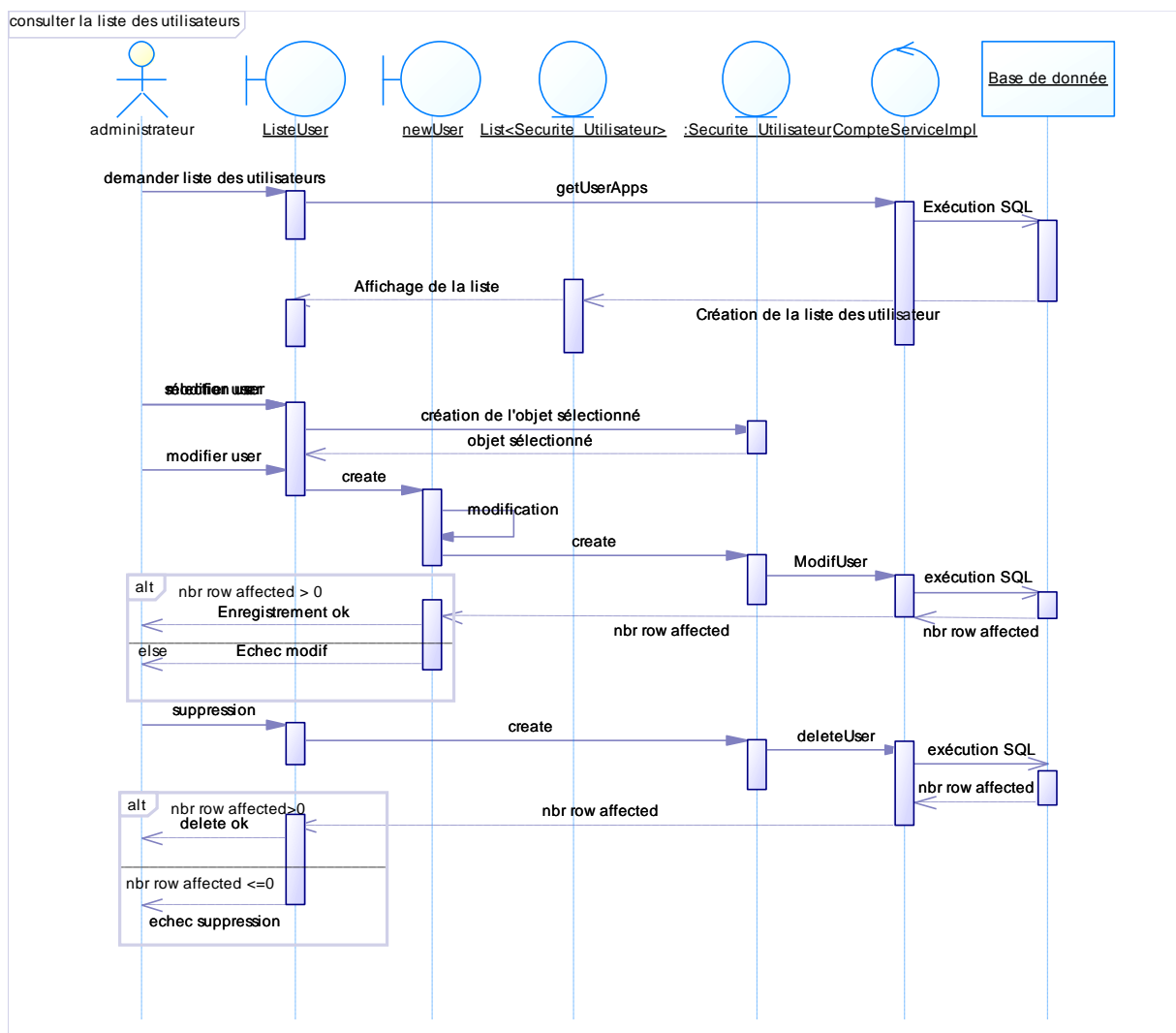


Figure 384 : diagramme de séquence du cas " Gérer les utilisateurs et les droits d'accès "

2.1.4.3 Diagramme de séquence du cas « gérer Les comptes mail »

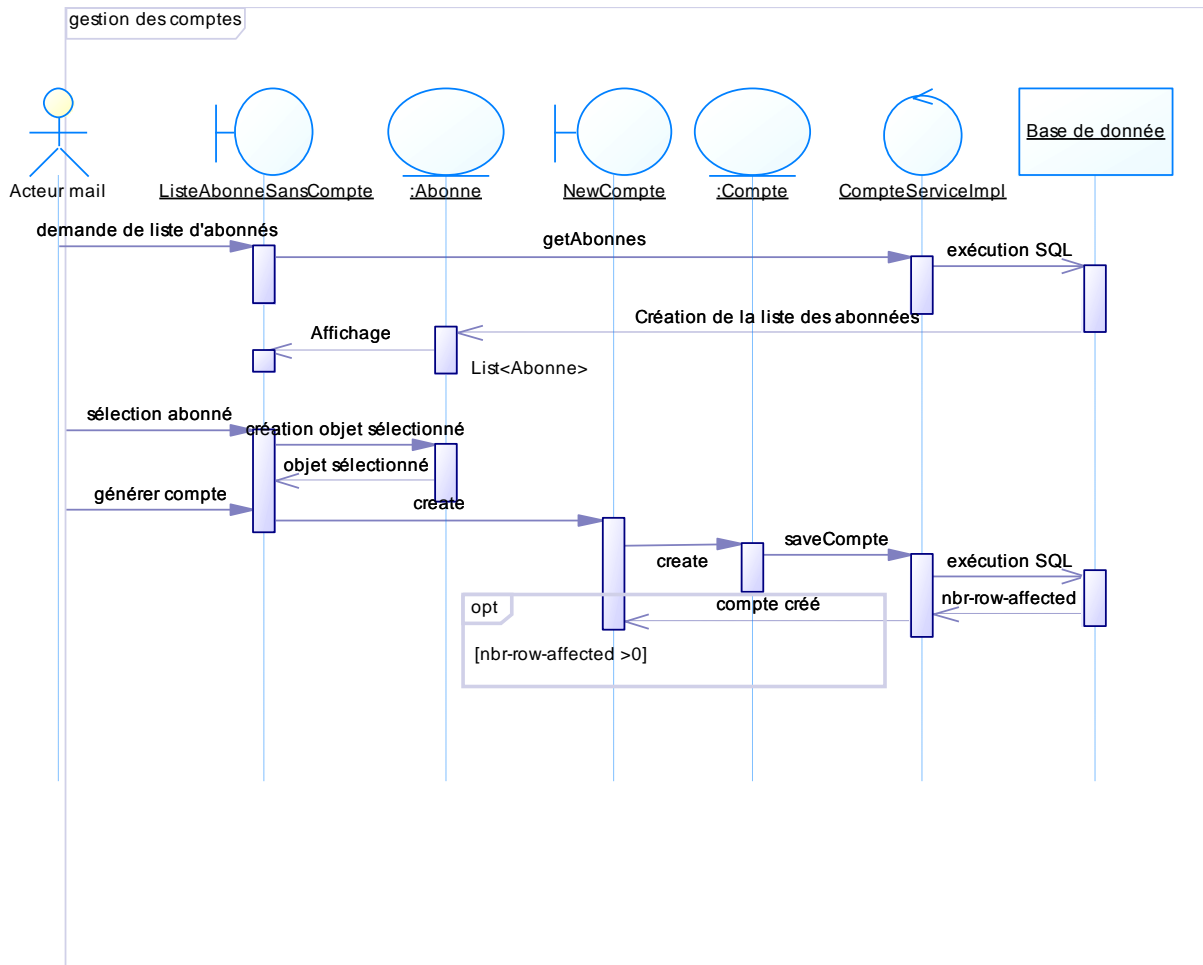


Figure 395 : Diagramme de séquence du cas « gérer les comptes»

L'utilisateur demande la liste des abonnés sans comptes (abonnés nouvellement créés) .Une fois affiché, il peut sélectionner un abonné pour le modifier, le supprimé ou générer le compte correspondant.

Afin d'éliminer la redondance, le diagramme illustre le cas de génération de compte seulement.

Lorsque l'abonné est sélectionné, un objet Abonne sera créé et passé en paramètre à l'objet dialogue NewCompte que l'utilisateur utilise pour générer les comptes mail et Internet avec mot de passe aléatoire .L'utilisateur enregistre le compte en passant l'objet Compte nouvellement crée à la classe Contrôleur CompteServiceImpl qui dialogue avec la base de données.

La recherche de compte est effectuée ainsi :

L'utilisateur peut effectuer la recherche de comptes soit par établissement soit à travers le compte lui-même.

## Conception et réalisation d'une application de gestion des comptes mail et internet

L'utilisateur choisi le type d'établissement puis le gouvernorat puis recherche l'établissement voulu. Le système lui affiche les comptes de cet établissement.

L'utilisateur peut aussi rechercher un compte en tapant le n°CIN ou une partie de nom et lance la recherche. Le système lui retourne alors le compte recherché.

Ensuite, l'utilisateur peut sélectionner un compte pour le modifier, le supprimer ou l'imprimer.

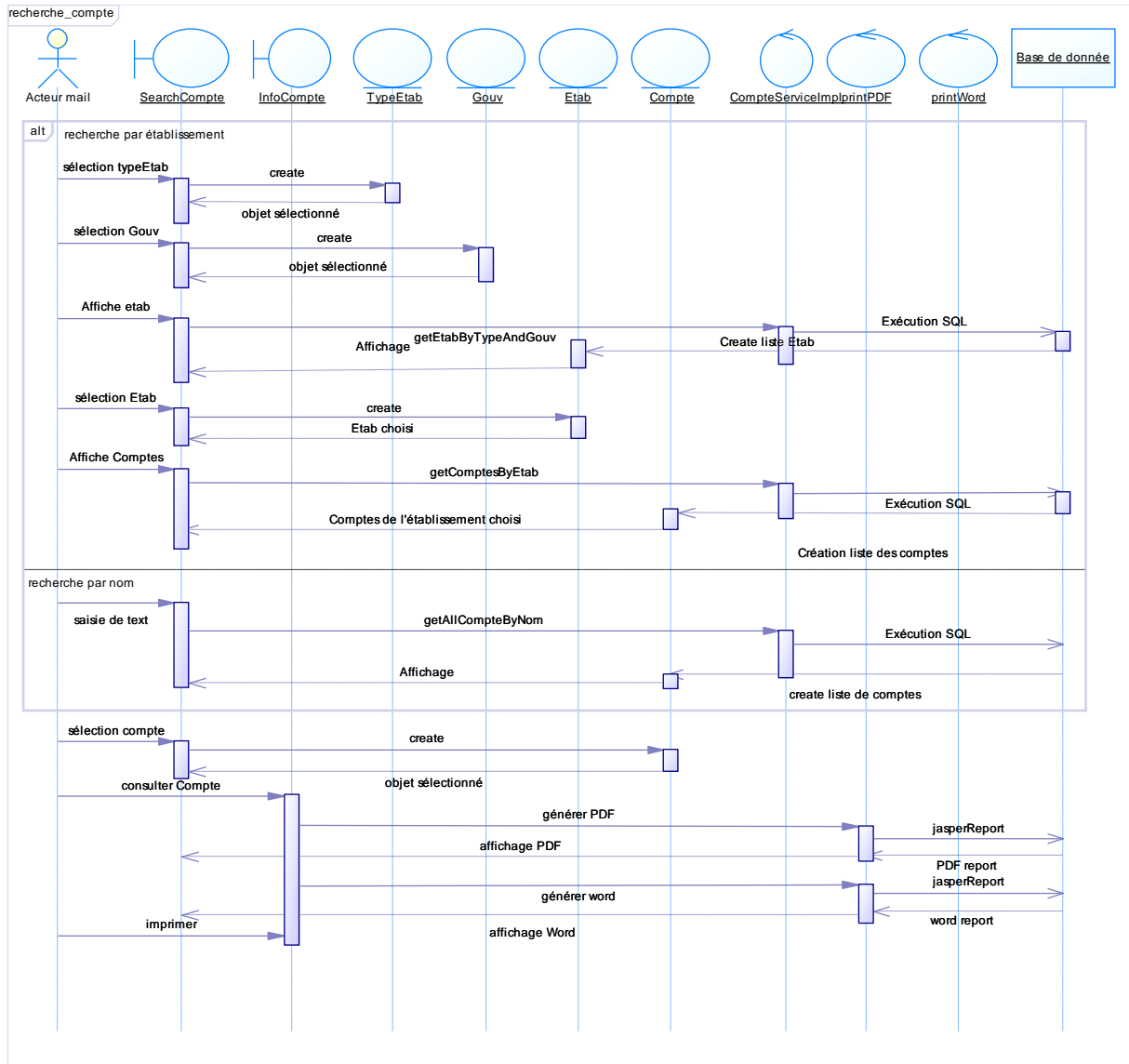


Figure 16 : Diagramme de séquence du cas « gérer les comptes »

### 2.1.4.4 Diagramme de séquence du cas "Gérer les serveurs Slis"

L'utilisateur affiche le tree des établissements. Il commence par sélectionner le gouvernorat, puis le type d'établissement puis sélectionne l'établissement voulu.

S'il trouve un Slis déjà enregistré dans cet établissement il peut alors consulter les logs ou ajouter les comptes Internet.

Sinon, il ajoute un Slis par le dialogue New Slis.

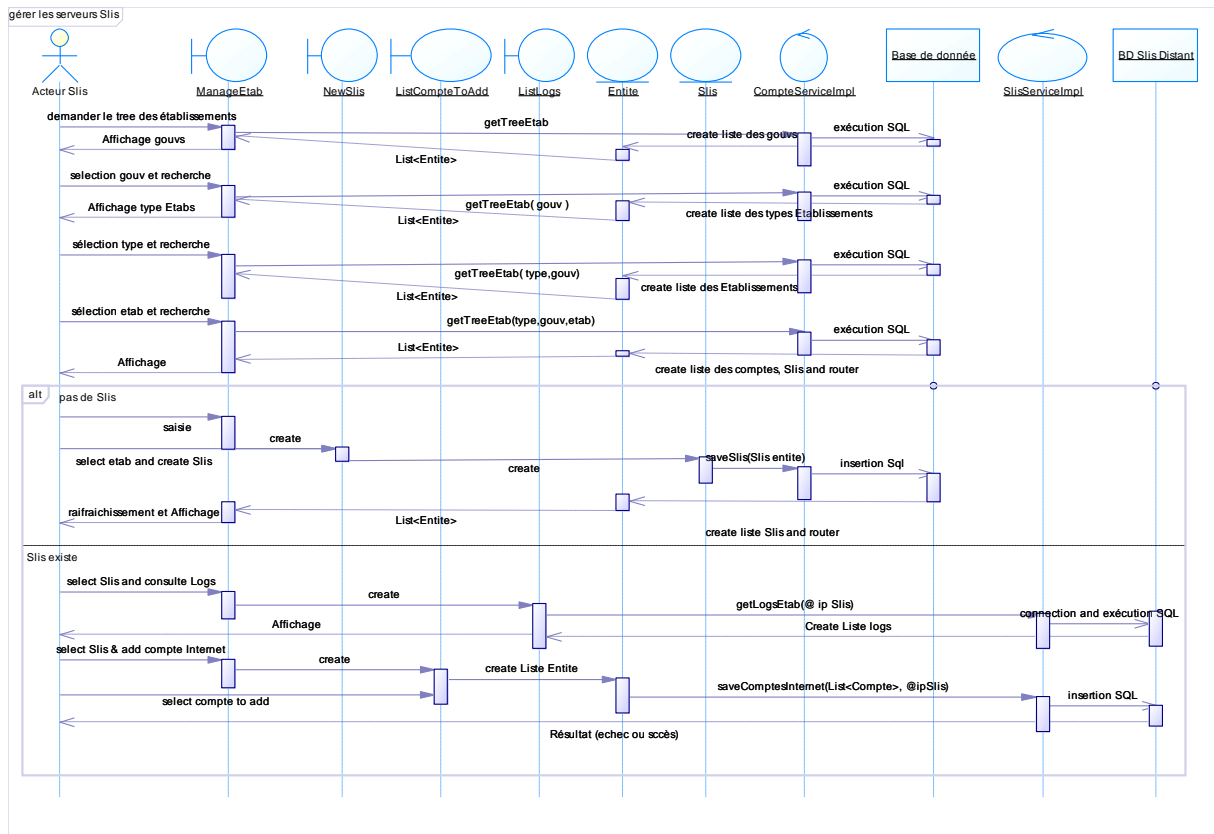


Figure 17 : Diagramme de séquence du cas « gérer les serveurs Slis »

A noter que l'utilisateur selon son profil peut sélectionner un nœud de tree le supprimer ou le modifier.

Ces actions ne sont pas traitées dans le diagramme de séquence pour alléger le diagramme.

## 3 Conclusion

Dans ce chapitre, nous avons conçu et documenté le code que nous devons produire. Dans cette phase, toutes les questions concernant la manière de réaliser le système à développer ont été élucidées. Le produit obtenu est un modèle graphique (ensemble de diagrammes) prêt à être codé. Dans le chapitre suivant nous allons étudier en détails les outils et les langages utilisés durant la phase de construction.

# Chapitre 5 : Réalisation

---

## Introduction

Après avoir achevé l'étape de conception de l'application, on va entamer dans ce chapitre la partie réalisation et implémentation dans laquelle on s'assure que le système est prêt pour être exploité par les utilisateurs finaux.

A la fin de ce chapitre, les objectifs doivent avoir été atteints et le projet doit être clos.

## 1 Environnement de développement

Pour la réalisation de ce travail, nous avons eu recours aux environnements suivants:

### 1.1 Environnement Matériel

Pour développer l'application, nous avons utilisé comme environnement matériel deux ordinateurs FUJITSU SIEMENS qui possèdent comme caractéristiques :

- Un processeur Intel Pentium® Core2 Duo, 2.49 GHz.
- Une mémoire vive de 2Go.
- Un disque dur 250 Go.
- Un écran 17 pouces.

### 1.2 Environnement Logiciel

- Windows 7 Professionnel, Service Pack 1 comme Système d'exploitation.
- PostgreSQL 9 comme système de gestion de base de données relationnel.
- Gxt comme framework de développement d'application Internet.
- NetBeans 6.9 comme Environnement de développement intégré.
- Tomcat 6.0.20 comme Moteur de servlet.
- Sybase Power Designer outils de modélisation

### 1.3 Choix des outils de développement

Nous avons déjà les décrit dans le chapitre 2 "Etude de l'art".

## 2 Architecture générale de l'application

Dans ce paragraphe, on va détailler l'architecture MVC du framework Gxt et son impact dans le développement de l'application. On finira par donner le diagramme de classe finale adapté au langage de développement choisi J2EE + GWT+GXT.

### 2.1 Architecture client-serveur GWT:

On commence par présenter le principe de développement de la boîte à outil GWT sur laquelle repose le framework GXT.

Le développement GWT couvre deux niveaux:

- le développement coté Client : on développe en Java comme si on est en train de développer avec Swing. A la phase de compilation tous ce code Java sera converti en Java Script en tenant compte des différences des browsers. Cette compilation est totalement transparente à l'utilisateur. Elle est réalisée par le fameux compilateur de Google.
- développement coté serveur: il repose sur le concept Ajax (JavaScript Et XML Asynchrones)

Le principe client/serveur de GWT repose sur la création de :

- Deux interfaces : une synchrone et une asynchrone.
- Une classe qui implémente le service

Noter que les liens entre les interfaces synchrones et asynchrones reposent sur des conventions de nommage.

Les interfaces font partis du package client : lors de sa compilation en Javascript GWT saura qu'il faut également générer ce qu'on appelle des stubs : il s'agit de morceau de code qui assure la communication entre un client et un serveur.

Il faut également noter que seule l'interface asynchrone peut être utilisée. L'appel au service ne sera pas bloquant pour le code. Si pour la suite du traitement on a besoin du résultat du traitement réalisé sur le serveur il faudra utiliser une callback.



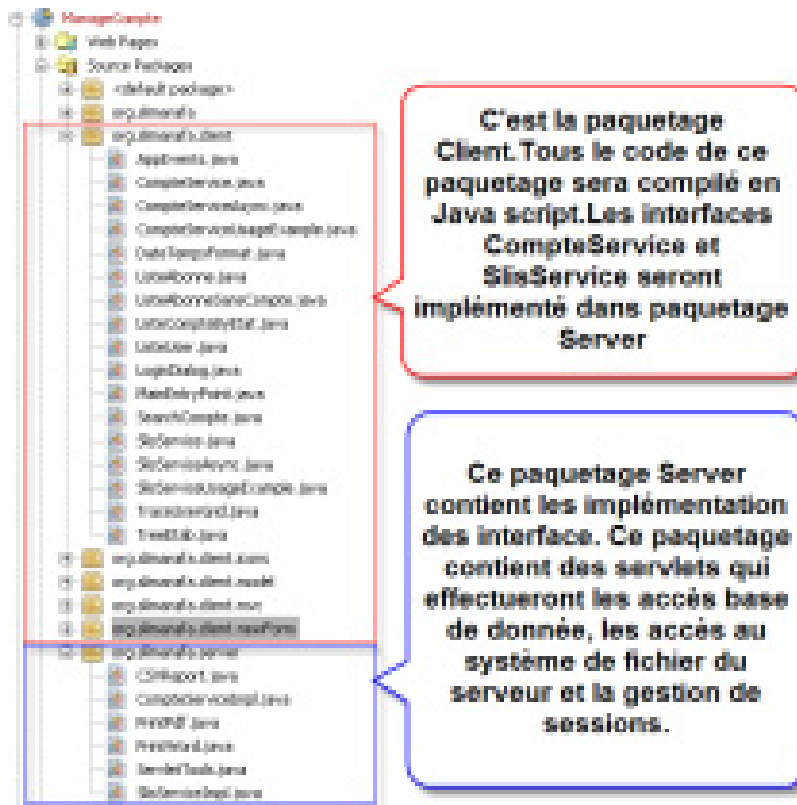


Figure 408 : Architecture client-serveur de l'application

## 2.2 Le modèle MVC de GXT :

Le Modèle-Vue-Contrôleur (en abrégé MVC, de l'anglais Model-View-Controller) est une architecture et une méthode de conception qui organise l'interface homme-machine (IHM) d'une application logicielle. Ce paradigme divise l'IHM en un modèle (modèle de données), une vue (présentation, interface utilisateur) et un contrôleur (logique de contrôle, gestion des événements, synchronisation), chacun ayant un rôle précis dans l'interface.

Les composants de Gxt reposent sur le modèle MVC. De plus Gxt offre une implémentation du modèle MVC qui consiste à :

- Créer une Classe AppEvents contenant les évènements de l'application voulus

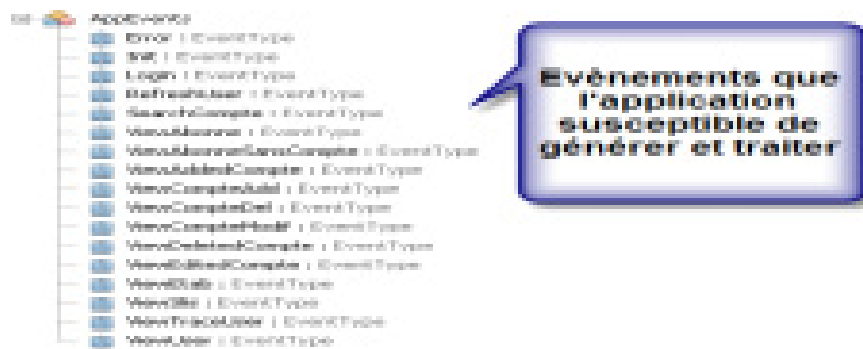


Figure 419 : Liste des évènements de l'application

- Créer une classe Controller et une classe Vue pour capter l'évènement et effectuer le nécessaire

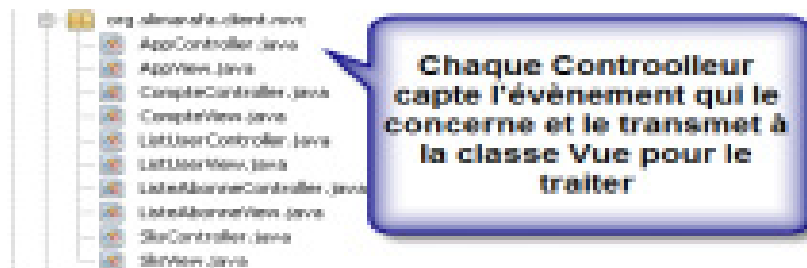


Figure 20 : Liste des évènements de l'application

- Déclarer le contrôleur auprès de contrôleur principal Dispatcher fourni par GXT

```
CompteServiceKeyno service = (CompteServiceKeyno) GWT.create(CompteService.class);
Registry.register(SERVICE, service);

final Dispatcher dispatcher = dispatcher.get();
dispatcher.addController(new AppController());
dispatcher.addController(new ListeMembreController());
dispatcher.addController(new ListUserController());
dispatcher.addController(new CompteController());
dispatcher.addController(new SiteController());
dispatcher.dispatch(AppEvents.Login);
```

Une instance de Dispatcher est créée. Tous les contrôleurs de l'application sont enregistrés au près du Dispatcher

Ce code lance l'évènement Login que le contrôleur AppController va capter

Figure 21 : Le contrôleur principal (Dispatcher)

Le dispatcher lance un évènement application (listé dans /client/AppEvents.java) à tous les contrôleurs. Si le contrôleur peut répondre à cet évènement particulier, il le

fait. S'il n'est pas concerné il ne fait rien. Le contrôleur doit accomplir l'exécution du logique métier ainsi que les MAJ des modèles. Les vues connectées à ces contrôleurs seront rafraichies en accord avec le model mis à jour correspondant.

### 4 Diagramme de classe final

D'après l'architecture client-serveur présentée et le modèle MVC tout le traitement sera effectué hors les classes du diagramme de classes. En effet les classes de ce diagramme seront les tables de la base de données et les entités beans utilisé par l'application pour le transfert de donnée depuis le serveur vers l'application.

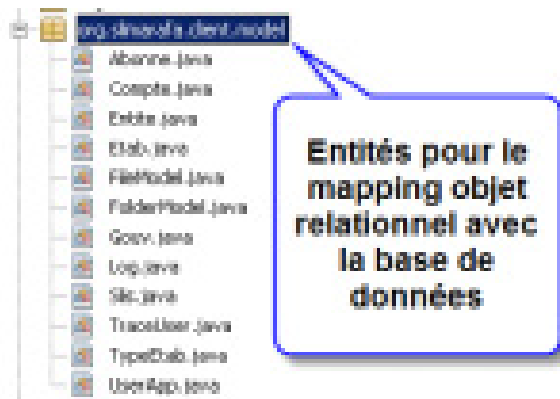


Figure 22 : Les Beans

Ainsi, le diagramme de classe final sera composé du diagramme précédent auquel on ajoute la classe `CompteServiceImpl` et `SlisServiceImpl` contenant toutes les méthodes d'ajout, suppression, modification et consultation des Entité de la base de données.

On ajoute aussi toutes les classes responsables de l'interface graphique de l'application.

### 5 Principales interfaces graphiques

La conception des interfaces de l'application est une étape très importante puisque toutes les interactions avec le cœur de l'application passent à travers ces interfaces, on doit alors guider l'utilisateur avec les messages d'erreurs et de notification si besoin, ainsi présenter un système complet.

Dans cette partie, nous allons présenter quelques interfaces de l'application, répondant aux recommandations ergonomiques de compatibilité, de guidage, de clarté, d'homogénéité et de souplesse. Nous avons choisi l'administrateur comme utilisateur vu qu'il présente à travers ces interactions la majeure partie des principales fonctionnalités de l'application.

## 5.1 Authentification

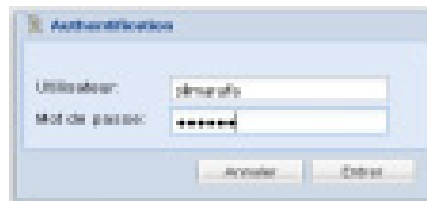


Figure 2342 : Interface d'authentification

Cette interface permet à l'utilisateur de s'authentifier et de se connecter au serveur de la base de données. L'utilisateur doit entrer son login et son mot de passe pour accéder à l'application. En cas d'erreur un message d'alerte s'affiche :

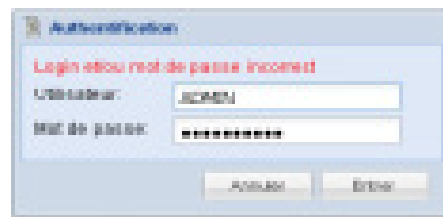


Figure 2443 : Message d'erreur

## 5.2

### 5.2.1 Ajouter un utilisateur

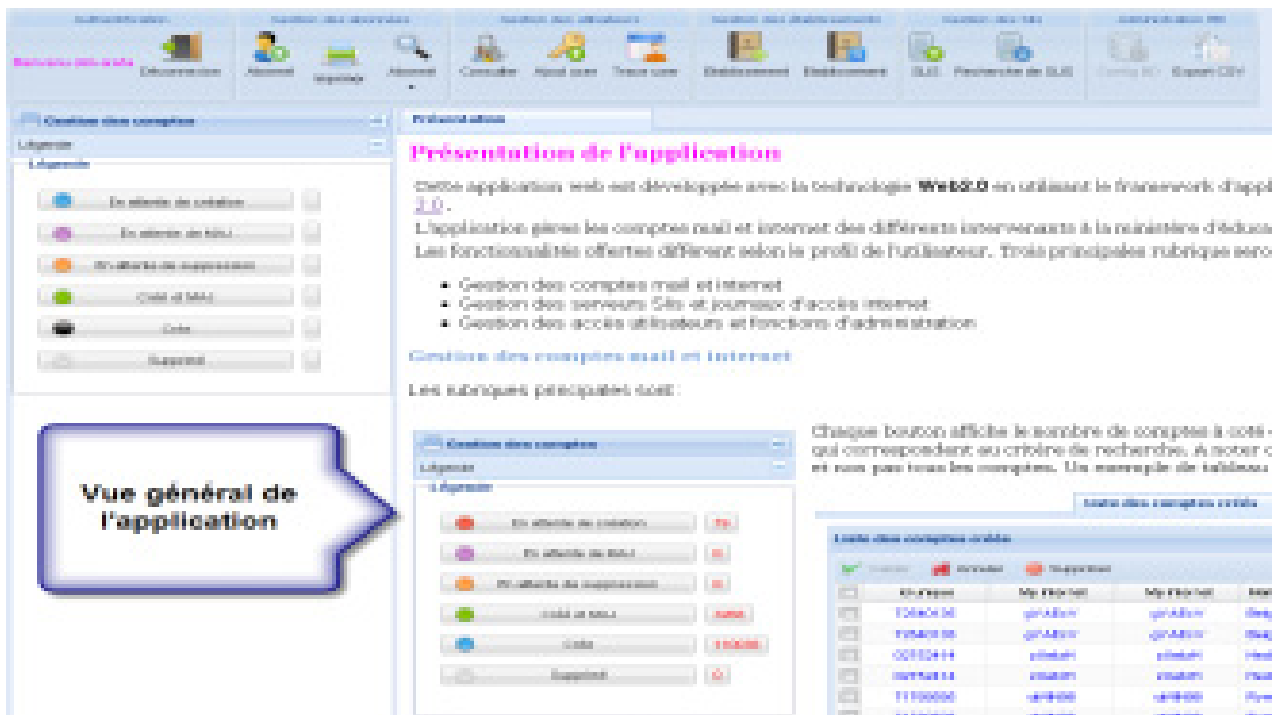



Figure 25 : Interface principale

La figure ci-dessus présente le menu principal, l'administrateur peut ajouter un utilisateur en

cliquant sur bouton ajout User  ; un formulaire comportant plusieurs champs s'affiche pour l'entrée des données. L'application gère le contrôle de saisie ainsi que la sauvegarde dans les bases des données.

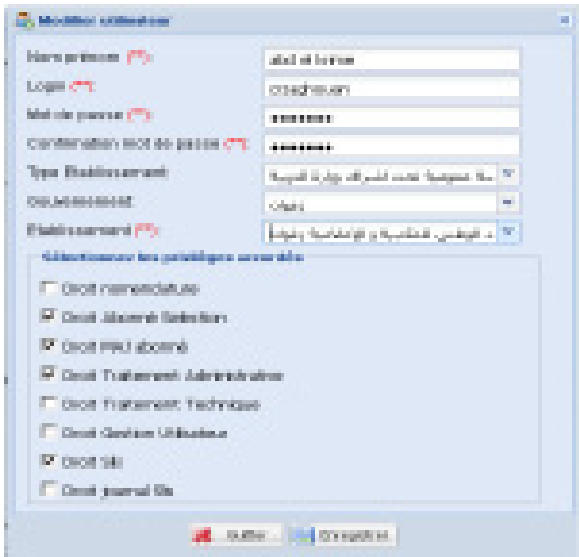


Figure 26 : Ajout utilisateur

## 5.2.2 Consulter les utilisateurs

Le bouton Consulter permet de voir la liste des utilisateurs et leur droit d'accès. On peut sélectionner un utilisateur pour le modifier ou le supprimer.

Code user	N° de passe	Utilisateur	Gouvernement	Type Club	Choixabonnement	DroitsInfo	Droit Admin	Droit Admin	Droit Train	DroitTechno	DroitGestion	Site?	Journal Site
coachman	veru01	rod erome	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	APPS	مغربي	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
stabilou	st01	Amad Ben Amad	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
amad	am01	amadlou amadlou	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
amad	am02	amad louche	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
chadine	chad01	amad chad	مغربي	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	amou code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
am	am	amou am	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
code	code	code code	الجزيرة	كرة و 1000 ريال	كرة و 1000 ريال	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 2744 : information sur les utilisateurs

### 5.2.3 Les traces utilisateurs

La trace des utilisateurs donne toutes les actions réalisées par l'utilisateur ainsi que les erreurs détectées lors de l'exécution de l'application. Pour ce faire on a recourt à la librairie Log4j et son plugin JDBCAppender qui enregistre directement à la base de données tous les messages d'info et d'erreurs.

Trace des utilisateurs

Spécifier l'utilisateur ou la date

Utilisateur:  Date:  Avant le  Après le  La

Les traces contiennent toutes les actions effectués par l'utilisateur indexées par la date et le nom de l'acteur

Type	Date	Message	Info pour débogage
INFO	29/05/2011 09:55:38	>> inconnu récupération des types des établissements	CompteServiceImpl - getAffiliés - ...
ERROR	29/05/2011 09:55:40	>> inconnuException getAffiliés : null	CompteServiceImpl - getAffiliés - 965
INFO	29/05/2011 09:55:40	>> inconnu récupération des établissements par genre null et type etab null	CompteServiceImpl - getAffiliés - 1429
ERROR	29/05/2011 09:55:40	>> inconnuException getAffiliés: by type etab et genre: null	CompteServiceImpl - getAffiliés - 1411
ERROR	29/05/2011 09:56:08	>> Exception getAffiliés : Ce statement a été fermé.	CompteServiceImpl - getAffiliés - L...
INFO	29/05/2011 09:56:29	>> inconnu récupération des types des établissements	CompteServiceImpl - getAffiliés - ...
ERROR	29/05/2011 09:56:31	>> inconnuException getAffiliés: null	CompteServiceImpl - getAffiliés - ...
ERROR	29/05/2011 09:56:33	>> inconnuException getAffiliés: null	CompteServiceImpl - getAffiliés - 965
ERROR	29/05/2011 09:56:34	>> erreur affichage des établissements par genre et type Ce statement a été fermé.	CompteServiceImpl - getAffiliés - 1429
INFO	29/05/2011 09:56:38	>> inconnu récupération des établissements par genre null et type etab null	CompteServiceImpl - getAffiliés - 1429
ERROR	29/05/2011 09:56:39	>> inconnuException getAffiliés: by type etab et genre: null	CompteServiceImpl - getAffiliés - 1411
ERROR	29/05/2011 09:56:51	>> Exception getAffiliés : Ce statement a été fermé.	CompteServiceImpl - getAffiliés - L...
INFO	29/05/2011 11:00:30	>> inconnu récupération des types des établissements	CompteServiceImpl - getAffiliés - ...

Page 1 sur 14 Page courants 1 - 32 sur 1274

Figure 2845 : Trace des utilisateurs

## 5.3 Gestion des comptes Mail

### 5.3.1 L'ajout d'un abonné



Après la cliquer sur le bouton 'Abonné +' un formulaire de saisie s'affiche permettant à l'utilisateur d'entrer les informations d'abonné.

Figure 2946 : Ajout d'un abonné

Un contrôle sur les champs est nécessaire pour garantir l'intégrité des données. Si l'enregistrement réussi un autre formulaire s'affiche à l'utilisateur générant le compte Mail en concaténant le nom et le prénom au domaine adquat selon l'établissement de l'abonné.

Figure 30 : formulaire de génération de compte

Si l'utilisateur quitte cette fenêtre sans enregistrer le compte, l'abonné reste à la base de données sans compte.

Pour générer le compte ; l'utilisateur sélectionne le bouton recherche Abonné sans compte.

### 5.3.2 La recherche d'un compte et l'impression

L'utilisateur sélectionne le bouton recherche Abonné comptes réalisés, un nouvel onglet :

Recherche de compte

Spécifier le type d'établissement ou le gouvernorat auquel appartient le compte mail

Type établissement:  Gouvernorat:  Etablissement:

Spécifier le nom du compte recherché ou l'Id unique ou le CIN

Nom, N° id ou CIN:

Listes des comptes

<input type="checkbox"/>	ID unique	Login	Mp internet	Mail	Mp mail	Etablissement	Etat	Date valid Admin	Date valid tech
<input type="checkbox"/>	0000016	0000016	0000016	ami.storace@nom	0000016	ني لفتيوار (النعامة)	مستقر	16 mai 2007	16 mai 2007
<input type="checkbox"/>	0007387	ecoleun	pRUGv	ecole.numerique@n	027724	ني لفتيوار (النعامة)	مستقر	11 mai 2011	12 mai 2011
<input type="checkbox"/>	0007387	0700427	02170q	0007387@nom	0007387	ني لفتيوار (النعامة)	مستقر	11 mai 2011	12 mai 2011
<input type="checkbox"/>	0404038	0404038	0404038	0404038@nom	0404038	ني لفتيوار (النعامة)	مستقر	13 fév. 2008	21 juil. 2008
<input checked="" type="checkbox"/>	0743143	nom001	F00075	mohamed@nom	0743143	ني لفتيوار (النعامة)	مستقر	11 juin 2007	12 juin 2007

Imprimer  
Supprimer  
Modifier

Page courants 11 - 15 sur 15

Figure 3147 : recherche de compte par l'établissement

En choisissant le bouton Impression cette page s'affiche :





République tunisienne

Ministère de l'éducation

Institut national de bureautique et  
de micro-informatique

## Fiche des paramètres de connexion EduNet

Propriétaire du compte :

محمد سليم العرافة

المعهد الوطني للمكتبة و الإعلامية تونس

تونس

**ARAFa mohamed slim**

Ces informations sont délivrées à titre personnel et confidentiel

### Paramètres de messagerie (E-mail)

Adresse E-mail :	mohamedslim.arafa@inbmi.edunet.tn
Nom du compte E-mail :	mohamedslim.arafa@inbmi.edunet.tn
Serveur POP3 (réception)	mail.edunet.tn
Serveur SMTP (émission)	smtp.edunet.tn
Mot de passe E-mail	Qod966

### Paramètres de connexion

Nom d'utilisateur	inbmi067
N° Appel téléphonique	1622
Mot de passe de connexion	Felma78
DNS 1	193.95.94.40
DNS 2	193.95.67.20
DNS 3	193.95.66.11



Institut National de Bureautique et d'informatique

3 Rue Astrucal 1082 Tunis

Tél : 71 833 800  
Fax : 71 834 190

Service messagerie

Tél : 71 833 800 - Poste 147 et 158  
Mail : feloma@edunet.tn

Figure 32 : Impression de compte

La génération de fichier PDF et Word est réalisée à travers la bibliothèque JasperReport.

### 5.3.3 Les états d'un compte :

Le compte créé précédemment est en état d'attente de création. Il ne sera exploitable qu'après son validation par le responsable Mail.



Figure 33 : États des comptes

La validation des modifications effectuées à un compte est nécessaire pour s'assurer que ces modifications seront répliquées sur le serveur LDAP du serveur Mail de l'INBMI.

D'ailleurs, le responsable Mail possède le bouton de génération CSV à travers lequel il génère un fichier CSV utilisé par le serveur LDAP.

Lorsqu'on sélectionne le bouton 'En attente de création' un nouvel onglet affiche les comptes à valider. Avec un clic droit sur le compte, le menu contextuel affiche un bouton pour valider ce compte.

On peut aussi cocher les comptes qu'on veut valider puis on clique sur le bouton 'valider' au dessus de la table.

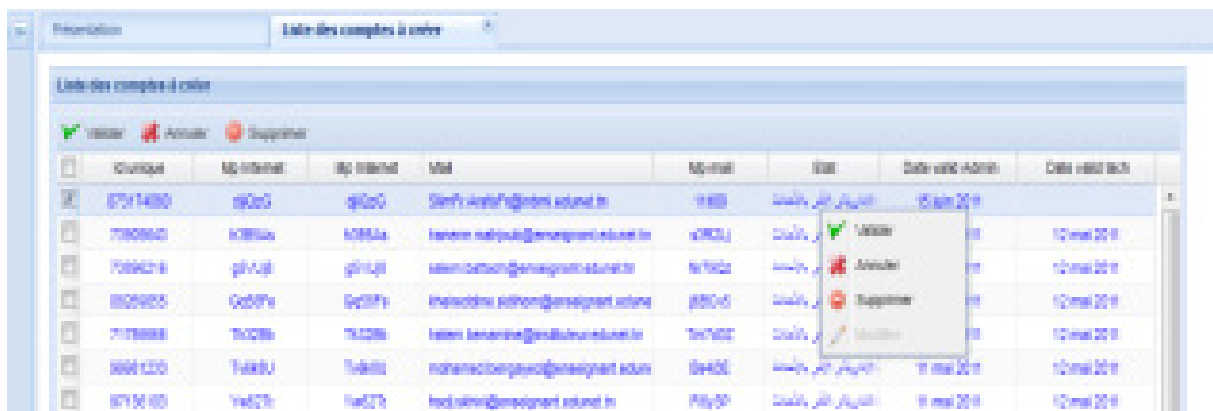


Figure 34 : Validation des comptes

Une fois validé, le compte peut être imprimé et envoyé à son propriétaire.

## 5.4 Gestion des serveurs Slis

On commence par rechercher l'établissement. L'affichage avec l'arbre des gouvernorats nous facilite la recherche.

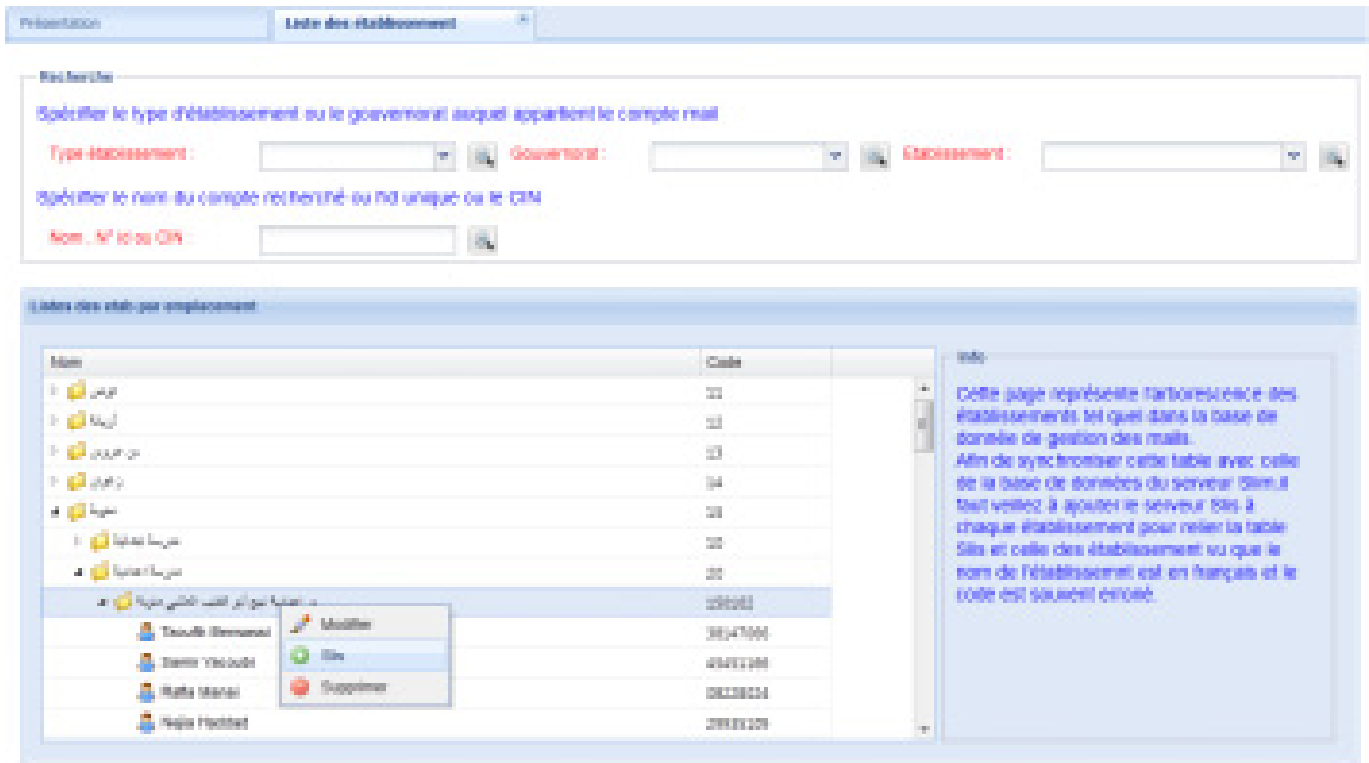


Figure 35 : ajout d'un serveur Slis

Comme dans la figure ce collège ne possède pas d'un serveur Slis enregistré alors que les comptes y existent. Un cli droit puis add Slis nous affiche cette boîte de dialogue :

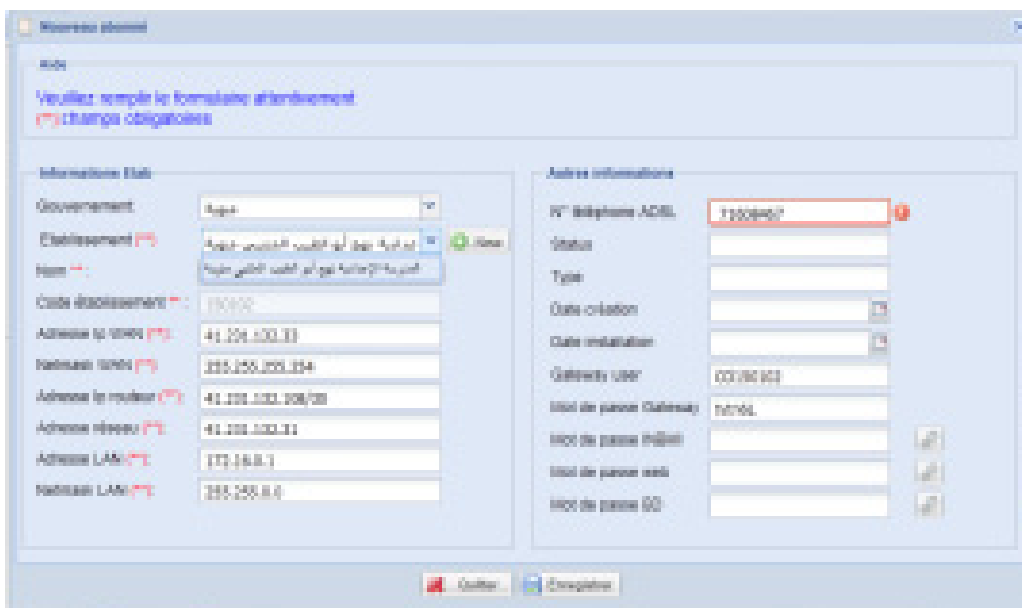


Figure 36 : Ajout d'un Slis

Comme déjà dit à l'étude de l'existant qu'un fichier Excel contient la liste des routeurs et serveurs Slis, ce fichier a été importé à la base de données dans une table nommée slis ADSL. Le problème réside dans l'impossibilité de joindre cette table avec celle des établissements. C'est pour cette raison l'utilisateur choisit dans la comboBox l'établissement correspondant de la table slisAdsl et tous les informations relatives seront affichées dans le formulaire. Si l'établissement n'existe pas dans la table slisADSL, on clique sur le bouton New et on l'ajoute carrément.

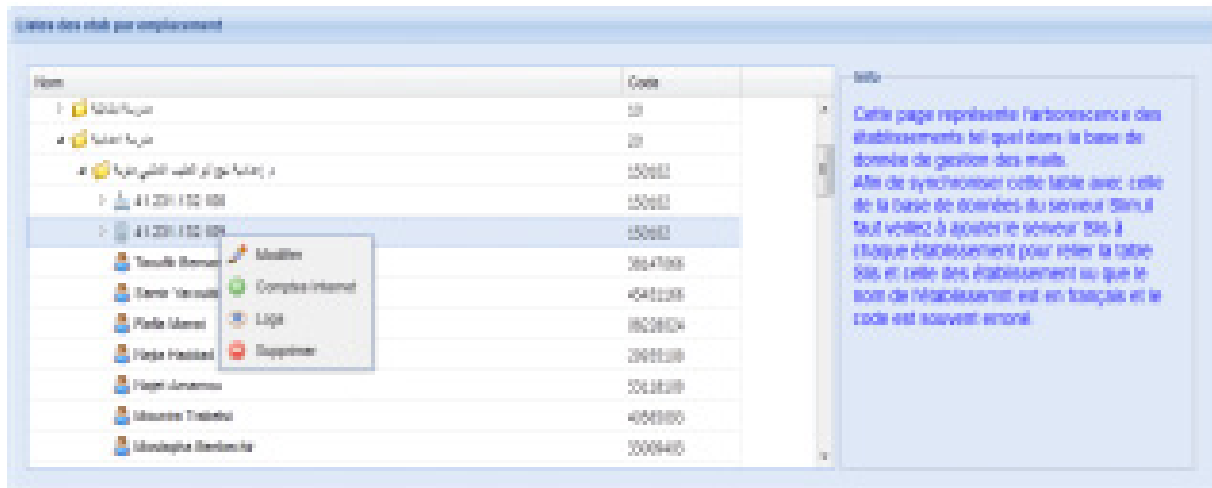


Figure 37 : Actions sur Slis

La consultation de logs et l'ajout de comptes Internet directement à Slis nécessite une connexion distante à la base de données Postgres de Slis.

Sachant que le fichier pg-hba.conf qui définit les hôtes autorisés à accéder à la BD nécessite une modification pour ajouter l'@ip de notre serveur. De plus le firewall de slis bloque les connexions entrantes au port 5432 de Postgres de Slis. Il nous faut donc une connexion à travers ssh pour effectuer ces manipulations et cela de manière automatique et transparente à l'utilisateur de l'application.

Heureusement, il existe une librairie nommée Ganymede-ssh2 qui facilite les connexions ssh à travers le langage Java. Ce qui nous a permis d'envoyer les commandes sed pour modifier le fichier pg-hba.conf et la commande iptable avant d'effectuer la connexion à la base postgres de Slis.

### Conclusion

A travers ce chapitre, nous avons présenté la réalisation de l'application en justifiant nos choix technologiques, en représentant quelques interfaces graphiques que nous avons jugé les plus importantes et en décrivant brièvement comment nous avons planifié notre projet.

## Conclusion général

L'objectif de notre projet de fin d'étude était de concevoir et implémenter une application de gestion des comptes mails et Internet des intervenants du ministère d'éducation.

Le point de départ de la réalisation de ce projet était une récolte des informations nécessaires pour dresser un état de l'existant, présenter un aperçu sur la problématique ainsi que l'architecture utiliser au sein des réseaux des établissements.

Par la suite, nous nous sommes intéressés à l'analyse et la spécification des besoins qui nous a permis de distinguer les différents acteurs interagissant avec l'application visée.

L'objectif de la partie suivante était la conception détaillée, dans laquelle nous avons fixé la structure globale de l'application.

Le dernier volet de notre projet était la partie réalisation qui a été consacrée à la présentation des outils du travail et les interfaces les plus significatives de notre application.

L'apport de ce travail a été d'une importance très considérable, en effet, il nous a permis : de suivre une méthodologie de travail bien étudié, d'approfondir nos connaissances dans le monde de développement des applications et de nous bien nous exercer sur le Framework GWT et Ext.

La réalisation d'un tel projet, nous a permis d'apprendre et de toucher du doigt une partie de divers aspects du métier de développeur et de celui du concepteur.

# Annexe A : UML

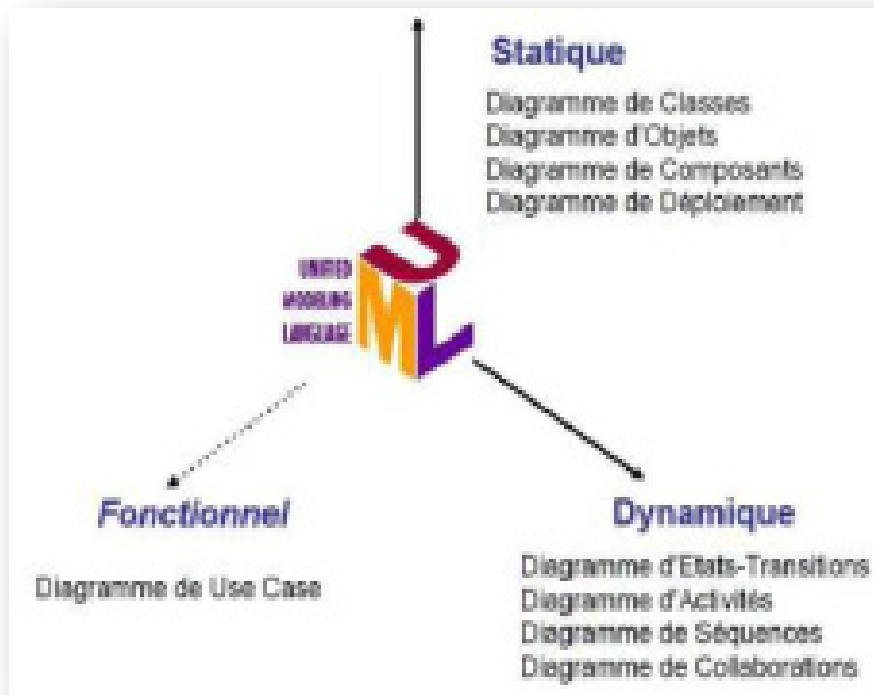


Figure 38: Les différentes vues du langage UML

## Diagramme de cas d'utilisation

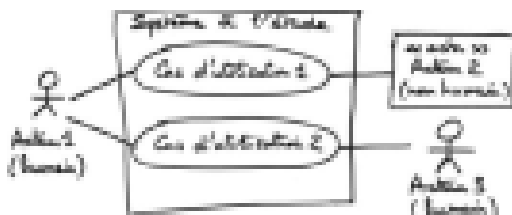


Figure 39: Les cas d'utilisation

Montre les interactions fonctionnelles entre les acteurs et le système à l'étude

**Acteur** : rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d'utilisation.

**Cas d'utilisation** (use case) : ensemble de séquences d'actions réalisées par le système produisant un résultat observable intéressant pour un acteur particulier. Collection de scénarios reliés par un objectif utilisateur commun.

**Association** : utilisée dans ce type de diagramme pour relier les acteurs et les cas d'utilisation par une relation qui signifie simplement « participe à ».

**Inclusion** : le cas d'utilisation de base en incorpore explicitement un autre, de façon obligatoire, à un endroit spécifié dans ses enchaînements.

**Extension** : le cas d'utilisation de base en incorpore implicitement un autre, de façon optionnelle, à un endroit spécifié indirectement dans celui qui procède à l'extension

**Généralisation** : les cas d'utilisation descendants héritent de la description de leur parent commun. Chacun d'entre eux peut néanmoins comprendre des relations spécifiques supplémentaires avec d'autres acteurs ou cas d'utilisation.

### Diagramme de séquence

Montre la séquence verticale des messages passés entre objets au sein d'une interaction

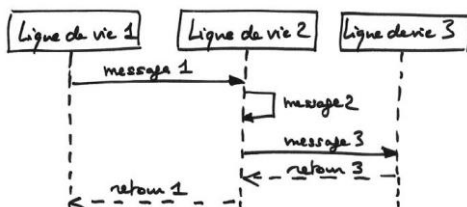


Figure 40: Les diagrammes de séquence

**Ligne de vie** : représentation de l'existence d'un élément participant dans un diagramme de séquence. Cela peut être un acteur ou le système en modélisation d'exigences, des objets logiciels en conception préliminaire ou conception détaillée.

**Message** : élément de communication unidirectionnel entre objets qui déclenche une activité dans l'objet destinataire. La réception d'un message provoque un événement dans l'objet récepteur.

**La flèche pointillée** représente un retour au sens UML. Cela signifie que le message en question est le résultat direct du message précédent.

**Spécification d'activation** : bande blanche qui représente une période d'activité sur une ligne de vie.

**Message synchrone** : envoi de message pour lequel l'émetteur se bloque en attente du retour et qui est représenté par une flèche pleine.

**Un message asynchrone**, au contraire, est représenté par une flèche ouverte.

**Occurrence d'interaction** : une interaction peut faire référence explicitement à une autre interaction grâce à un cadre avec le mot-clé ref et indiquant le nom de l'autre interaction.

UML 2 a ajouté une nouvelle notation très utile : les cadres d'interaction.

Chaque cadre possède un opérateur et peut être divisé en fragments.

Les principaux opérateurs sont :

- **loop** : boucle. Le fragment peut s'exécuter plusieurs fois, et la condition de garde explicite l'itération.
- **opt** : optionnel. Le fragment ne s'exécute que si la condition fournie est vraie.
- **alt** : fragments alternatifs. Seul le fragment possédant la condition vraie s'exécutera.

### Diagramme de classes

Montre les briques de base statiques : classes, associations, interfaces, attributs, opérations, généralisations, etc.

**Classe**: description abstraite d'un ensemble d'objets qui partagent les mêmes propriétés (attributs et associations) et comportements (opérations et états).

**Attribut** : donnée déclarée au niveau d'une classe, éventuellement typée, à laquelle chacun des objets de cette classe donne une valeur. Un attribut peut posséder une multiplicité et une valeur initiale.

Un attribut dérivé (« / ») est un attribut dont la valeur peut être déduite d'autres informations disponibles dans le modèle.

**Opération** : élément de comportement des objets, défini de manière globale dans leur classe. Une opération peut déclarer des paramètres (eux-mêmes typés) ainsi qu'un type de retour.

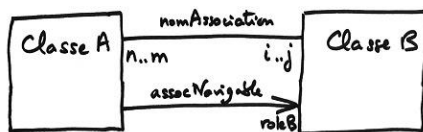


Figure 41: Les associations entre classes

**Association** : relation sémantique durable entre deux classes, qui décrit un ensemble de liens entre instances. Une association est bidirectionnelle par défaut, sauf si l'on restreint sa navigabilité en ajoutant une flèche.

**Rôle** : nom donné à une extrémité d'une association ; par extension, manière dont les instances d'une classe voient les instances d'une autre classe au travers d'une association.

**Multiplicité** : le nombre d'objets (min..max) qui peuvent participer à une relation avec un autre objet dans le cadre d'une association. Multiplicités fréquentes :

- 0..1 = optionnel (mais pas multiple)
- 1 = exactement 1
- 0..\* = \* = quelconque
- 1..\* = au moins 1



## Annexe B : GWT

L'outil GWT se décompose en deux grandes briques :

- Le framework de composants.
- Le compilateur Java vers JavaScript :

Toute l'ingéniosité de GWT est d'avoir su construire un compilateur Java vers JavaScript.

Un compilateur intelligent capable d'optimiser et de générer du code tout en respectant les préceptes de base du Web. Toute cette face cachée de GWT est encore très méconnue du grand public, qui, après tout, n'a pas à se soucier des implémentations internes du compilateur. Et pourtant, les vraies pépites, la vraie beauté de ce framework réside dans cette partie passionnante de GWT. Pour celui qui sait décrypter un minimum les nombreuses subtilités et configurations du compilateur, chaque fonctionnalité est une source d'inspiration unique.

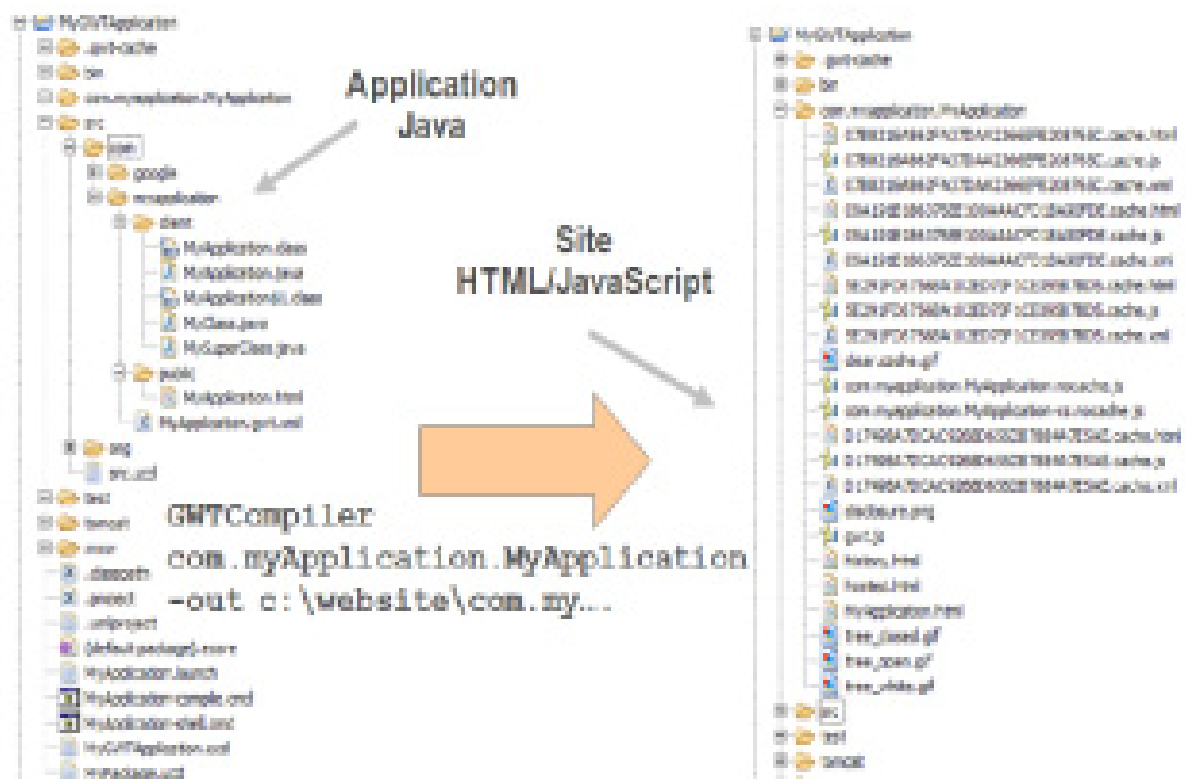


Figure42: Le compilateur GWT

Le compilateur GWT est en perpétuelle évolution car la taille du framework ne cesse d'augmenter (il suffit d'observer le nombre de nouvelles API). Les utilisateurs n'ont de cesse

de réclamer des applications réactives avec des temps de chargement instantanés ; impossible dans ce contexte de s'asseoir sur ses lauriers. Plus qu'une nécessité, l'amélioration du JavaScript généré par le compilateur est devenue pour chaque version une urgence vitale. Ce chapitre explore les nombreuses facettes du compilateur GWT et aborde la structure des fichiers générés et les différentes optimisations. Un éclairage particulier est apporté au mécanisme permettant d'étendre le processus de génération pour y ajouter des traitements spécifiques.

### 1) Introduction au compilateur

Le compilateur de GWT est l'essence même du framework. Lorsqu'on sait la richesse des sept mille classes du JDK, on réalise que celui qui ose imaginer un jour que ce langage peut produire du code JavaScript a du génie.

Même s'il est vrai qu'il existe des similitudes entre Java et JavaScript, certaines subtilités peuvent poser problème. Java propose des classes, JavaScript des prototypes de méthodes. Java dispose d'un mécanisme d'espaces de nommage (les fameux packages), JavaScript non. Java permet d'effectuer des appels polymorphiques, ils sont plus complexes en JavaScript. Java est un langage à typage statique, JavaScript un langage à typage dynamique.

Malgré tous ces points, GWT réussit à marier ces deux langages sans briser à aucun moment leur sémantique respective.

Pour le compiler en JavaScript, nous utilisons le script Ant généré par GWT lors de la création du squelette projet. Ce script contient une tâche `gwtc` à laquelle nous ajoutons les options de compilation suivante `-draftCompile` et `-style PRETTY`. La première demande au compilateur de ne pas trop optimiser le script cible. En production, cette option est à proscrire, car elle a tendance à générer un gros fichier.

En revanche, pour des raisons pédagogiques, cela permet d'obtenir une version fidèle du JavaScript avant optimisation.

La seconde option demande à GWT d'afficher un fichier source JavaScript lisible non obfusqué. Cela permet de mieux comprendre le contenu du script.

### 2) Les étapes du compilateur

Nous allons ici détailler les différentes étapes menant à la génération des artéfacts lors de la compilation d'un programme GWT. L'idée est ici de bien comprendre le processus d'optimisation et le modèle interne au compilateur.

*GWT a besoin du code source Java.*

Il y a un débat récurrent au sein des contributeurs GWT qui est celui du modèle de compilation. À l'origine, le choix a été de s'appuyer sur le code source Java pour générer le JavaScript plutôt que réutiliser le bte code. Nous ne discuterons pas ici de la pertinence de

cette décision. Il faut simplement savoir que GWT a besoin du code source car il extrait certaines informations telles que le code JSNI. Une autre raison avancée par les créateurs de GWT est la possibilité d'optimiser à partir d'informations présentes uniquement dans le code source. Si on prend, par exemple, les types génériques (`Class<T>`), ceux-ci sont réécrits par le compilateur.

Citons maintenant les différentes étapes intervenant lors de la compilation :

- *Lecture des informations de configuration*
- *Création de l'arbre syntaxique GWT*
- *La génération de code JavaScript et les optimisations*
- *La réduction de code (pruning)*
- *La finalisation de méthodes et de classes*
- *La substitution par appels statiques*
- *La réduction de type*
- *L'élimination de code mort*

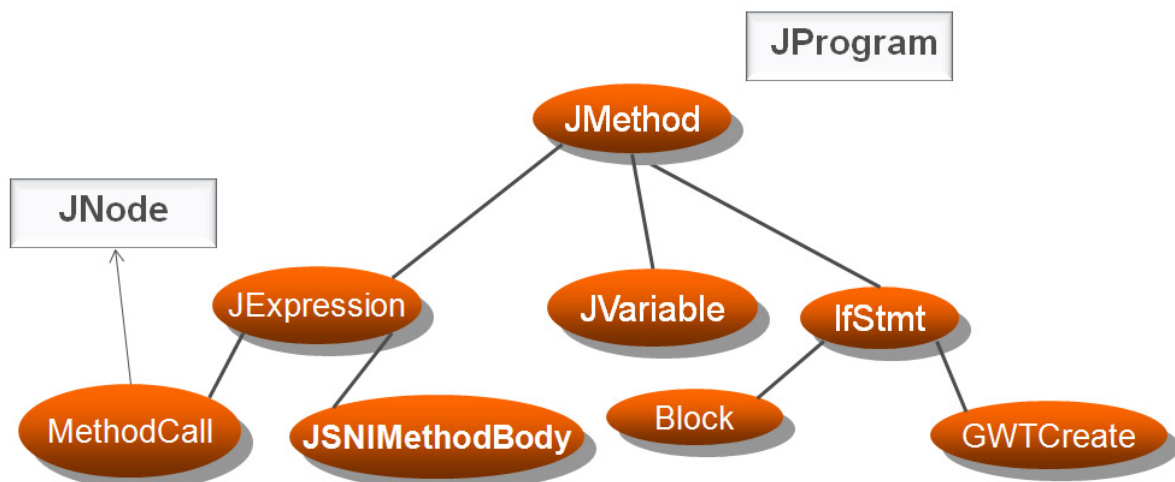


Figure 4348: Structure de l'arbre AST GWT

### 3) Accélération des temps de compilation

Vu la complexité des optimisations et le mode de fonctionnement du compilateur, ce n'est pas une surprise si l'une des préoccupations majeures des développeurs GWT concernent les temps de compilation. Avec GWT 2 et la possibilité de tester le code à partir d'un vrai navigateur, cette contrainte ne devrait plus réellement constituer un facteur de blocage. En effet, dans cette version, la compilation réelle n'intervient que pour valider définitivement le rendu visuel d'une application.

Malgré tout, il existe plusieurs optimisations possibles, dont certaines ont été abordées dans le chapitre sur la liaison différée.

Voici une liste d'actions susceptibles de réduire en moyenne de 50 % le temps de compilation, voire 800 % dans certains cas :

- Réduire le nombre de permutations : GWT est paramétré par défaut pour générer 6 permutations (une par navigateur). Si on sait que seuls IE 7 et Firefox 3 seront supportés, il suffit de définir la propriété `user.agent` de la manière suivante dans le fichier de configuration du module : `<set-propertyname="user.agent" value="ie6, gecko" />`  
Ce paramétrage réduit en moyenne de 50 % les temps de compilation.
- Ajouter l'option `-draftCompile` lors de la compilation : en phase de développement les gains peuvent aller de 5 à 30 % en fonction des scénarios.
- Donner une valeur à l'argument `-localWorkers` : GWT 1.5 a introduit la notion de worker pour la compilation. Pour une machine bi-cœur (*Dual Core*), une valeur paramétrée à 2 permet de paralléliser la compilation des permutations sur chaque cœur. Le gain est en moyenne de 10 %. Ce chiffre s'améliore progressivement dans le cas d'une machine à 4 cœurs (*Quad Core*).
- Configurer la JVM avec des paramètres adaptés : pour une application moyenne, les options suivantes permettent d'assurer suffisamment de mémoire, une taille de pile cohérente et un espace de stockage temporaire adapté.

#### 4) Le framework RPC

On développe seulement 3 classes pour les appels distants en utilisant le RPC (remote procedure calls).

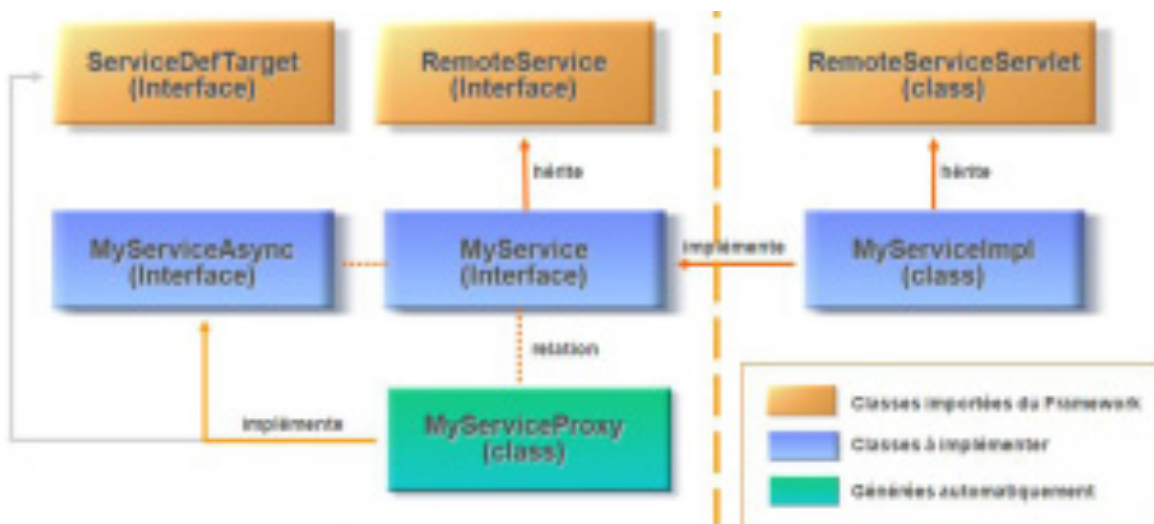


Figure44: Vue générale du service RPC

Toute communication avec la couche serveur s'opère via le protocole RPC :

- ✓ Le service s'appuie sur l'API JEE Servlet.

- ✓ Les appels sont asynchrones (classe AsyncCallback).
- ✓ Le code serveur n'est pas converti en JavaScript (Possibilité d'utiliser le JDK 1.5 et toutes les classes du Framework Java).

Les types sont sérialisés automatiquement par GWT suivant un format spécifique RPC

### 5) Les raisons d'investir dans GWT

En fin nous présentons nos raisons pour utiliser le GWT :

- ✓ Marge de progression : Le GWT présente une marge de progression considérable par rapport aux autres Framework de même catégorie.
- ✓ Productivité
- ✓ Richesse fonctionnelle (AJAX)
- ✓ Pérennisation des compétences (JAVA)
- ✓ Extensibilité
- ✓ Intégration avec l'existant
- ✓ Standards ouverts (XML, JSON, J2EE)
- ✓ Open Source
- ✓ Simplicité.

## Bibliographie :

- [1] Grant Slender **Developing with Ext Gwt** 140 pages.
- [2] John C. Worsley & Joshua D. Drake **PostgreSQL par la pratique** 628 pages.
- [3] Pascale Roques & Franck Vallée **UML en action, 2ème édition** 402 pages.

## Netographie :

- [1] <http://www.extjs.com>
- [2] <http://fr.wikipedia.org>
- [3] <http://inbmi.edunet.tn>
- [4] <http://code.google.com/intl/fr-FR/webtoolkit/>
- [5] <http://uml.free.fr/>