Université Saàd Dahlab, Blida

Faculté des sciences

Département d'informatique

Réalisé par :

- *Tedjani Hana*
- *Sabri Louiza*

En vue d'obtenir le diplôme de Master
Domaine : Mathématique et informatique
Filière : Informatique
Spécialité : Ingénierie du logiciel

Conception et réalisation d'un système de gestion DRH sécurisé à partir des composants EJB

Organisme d'accueil :
Sonelgaz « SDC »

Promotrice : *Mme Touahri Dalila* ***Encadreur :*** *Mme Sandjakeddine Nassima*

Président : Mr Bensouar Djamel

Examinateur : Mr Sidoumou

Examinateur :

2011/2012

Remercîment

Nous remercions avant tout le bon dieu qui nous à donner la force d'arriver jusque là.

Nous exprimons toute notre gratitude à mademoiselle Touahri Dalila pour l'effort fourni, les conseils prodigués, sa patience et sa persévérance dans le suivi.

Nous adressons également nos remercîments à notre encadreur Mme Sandjakeddine Nassima ainsi que Mr Rasemi Nour Eddine et tout nos enseignants spécialement Mr Bennouar Djamel qui nous a facilité la compréhension et la maitrise de notre projet, sans oublier le chef de notre département Mr Massied et toute personne qui nous à aider de prés ou de loin.

Dédicace

Je dédie ce modeste travail à:

Mes parents qui m'ont éclairés le chemin et qui m'ont encouragé tout au long de mes études.

Mon allié Ishak pour son soutien précieux.

Ma chère sœur Ahlem et ses filles.

Mes braves frères Anis & Rafik.

Ma précieuse belle famille.

Tout mes amis surtout Nari, Samira, Hanane et Imene.....

Hana

Dédicace

Je dédie ce modeste travail à:

Mes parents qui m'ont éclairés le chemin et qui m'ont encouragé tout au long de mes études.

Mon allié Amin pour son soutien précieux.

Ma sœur Ghanima.

Mes frères Abd el Malek et Hicham.

Tout mes amis surtout Hana, Manel, Yasmine ....

Louiza

ملخص

الغرض من هذه المذكرة هو تطبيق مكونات JAVA BEANS ENTERPRISE (EJB) لجعل نظام إدارة الموارد البشرية لشركة سونلغاز (ش. ت. و) ا توماتيكي . كما انه لا يتم تطبيق EJB إلى كل النظام ولكن على جزء منه و الذي سيتم تعريفها في هذه المذكرة.

Résumé

Le but de ce mémoire est l'application des composants d'ENTREPRISE JAVA BEANS (EJB) pour l'automatisation du système de gestion des ressources humaine de l'entreprise Sonelgaz (SDC) .noter que les EJB ne sont pas appliqué sur tout le système mais sur une partie qui sera définis dans ce présent mémoire.

Mot clés : *Composants, Entreprise Java Beans, EJB*

Abstract

The purpose of this paper is the application of ENTERPRISE JAVA BEANS components (EJB) for the automation of system management human resources company Sonelgaz (SDC). Noted that EJB are not applied to all system but on a part which will be defined in the present specification.

Keywords: *Components, Enterprise Java Beans, EJB*

Table des matières

Bibliographie

Table des figures

Chapitre 4 : Implémentation

Introduction générale

Nous vivons dans une ère ou les applications logiciels des entreprises deviennent de plus en plus dynamique et complexes pour y remédier, de nombreux chercheurs en matière de génie logiciel ne cesse pas de développer et d'adapter des nouveaux concepts technologiques contribuant au développement des applications plus étendues et flexibles,

Dans ce contexte la société sonelgaz distribution centre SDC qui fait face à de nombreux problèmes parmi eux le traitement manuel de certaines tâches qui gère la carrière des employés; Dans le cadre de la réalisation du master informatique la société sonelgaz nous a demander de réaliser une application qui gère la carrière des fonctionnaires. Etant donné que la carrière ne suit pas une progression régulière nous avons proposé une application à base de composant caractérisée par la réutilisation et la capacité d'être mise a jour au fil du temps sans refaire la conception et l'implémentation à zéro du fait qu'on peut supprimer un composant ou modifier un composant.

Le paradigme « composant » a introduit une nouvelle méthode pour la conception des applications logicielles, le travail présentée dans ce mémoire est réalisé à base de composants « des composants logicielles sont développés par assemblage et adaptation pour produire un système complet » et il doit se baser sur les éléments suivante :

- *Adaptation d'une approche basée sur les composants.*
- *Les composants qui seront définis dans le projet doivent être très fins pour que le travail soit robuste face au changement (un composant fin pourrait être réalise, testé et documenté dans des délais très court).*
- *L'approche composant permettrait d'ajuster des fonctionnalités d'un composant ou de lui ajouter des fonctionnalités sans perturbation des autres éléments d'un système bâti par assemblage des composants.*

Ce présent mémoire s'organise comme suit :

- *Chapitre 1 : dans le quel nous présenterons l'approche à composant*

- *Chapitre 2 : portera sur notre choix des composants EJB et l'étude de ses différents types*

- *Chapitre 3 : cette partie décrit et délimite les parties à développer : par une modélisation des tâches qui intéresse notre système et l'extraction des composants persistants a partir d'un diagramme de classe pour la mise en œuvre de notre système*

- *Chapitre 4 : dans le quel nous présenterons les technologies et les outils qui vont êtres utilisées dans l'implémentation de notre application à base de composant et accompagner de justification de nos choix de développement (serveurs, langage, etc.). une présentation détaillée du fonctionnement de notre système.*

- Nous terminerons notre mémoire par une conclusion générale

PARTIE 1:ETAT DE L'ART

1- Profil global de la SDC:

La création de la filiale Sonelgaz Distribution Centre par abréviation SDC est liée a la mise en application des dispositions de la loi 02/01 du 05 janvier 2002 relative à l'électricité et la distribution du gaz par canalisation, les changements économiques induit par cette loi ont permit le passage de Sonelgaz en un holding de sociétés exerçants différents métiers.

La SDC est créé en janvier 2006 sous forme de société par action.

2- Organigramme de la division ressources humaines :

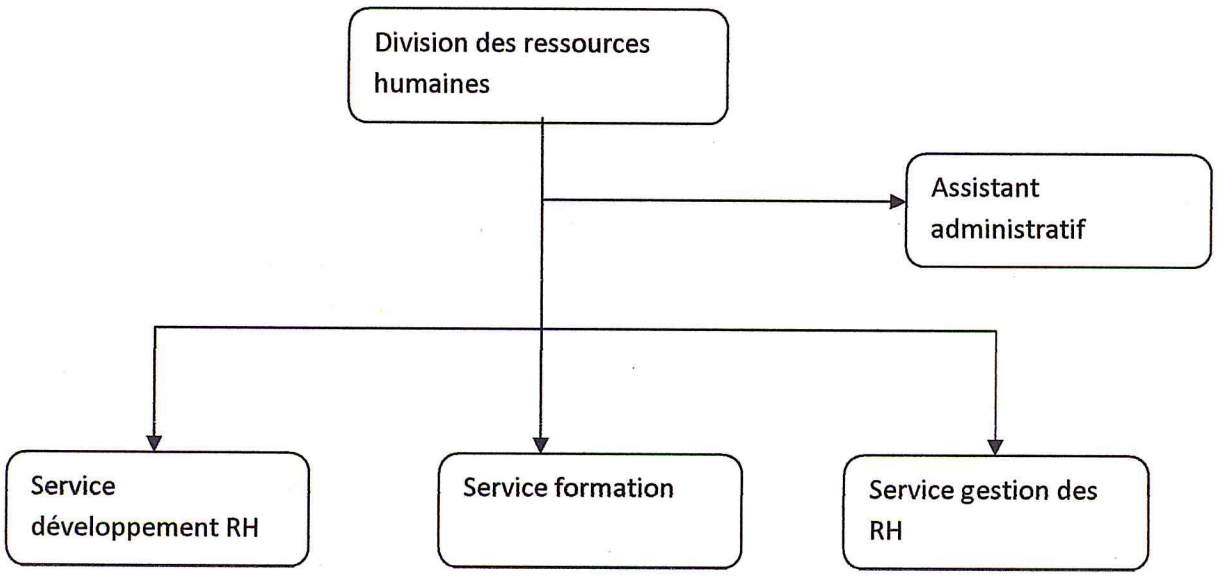

Figure 1 : Organigramme de la division ressource humaine

CHAPITRE 1: APPROCHE A COMPOSANT

1-1- Introduction :

L'approche à composants est relativement récente dans l'histoire du génie logiciel, elle est apparue autour du milieu des années 90. Cette approche promet la construction d'applications à partir de l'assemblage de composants logiciels [1].

L'approche à composants est fondée sur l'idée que le développement et l'assemblage de composants peuvent être réalisés de façon totalement séparée par des acteurs différents et dans des emplacements différents.

Par rapport à l'approche orientée objet l'approche à composants promet notamment l'emploi de la composition au lieu de l'héritage comme mécanisme de réutilisation.

1-2- Qu'est ce qu'un composant logiciel?

La notion composant n'est pas encore totalement figée, il existe actuellement plusieurs définitions du composant, nous proposons la traduction des définitions données par :

- Les participants à la première édition du 'Workshop on component oriented programming'[2] « Un composant logiciel est une unité de composition spécifiant par contrat, ses interfaces (fournies et requises) et ses dépendances explicites aux contextes. Un composant logiciel peut être déployé indépendamment et peut être sujet de composition par un tiers pour la conception d'applications logicielles » [3].

  Il en résulte de cette définition que :

  - Un composant logiciel peut être installé sur différentes plateformes.
  - Un composant est capable de s'auto-décrire ce qui permet aux constructeurs d'applications de l'utiliser facilement sans qu'ils aient besoin d'en connaitre son fonctionnement.

- Johannes Sametinger « Le composant réutilisable est autonome, a des interfaces identifiables claires dont elles décrivent ou exécutent des fonctions spécifiques. Il a des interfaces claires, des documents appropriés et la définition d'état de la réutilisation » [4].

Il en résulte de cette définition que :

- Le composant réutilisable est utilisé par un logiciel sans inclure un autre composant.
- Le composant doit être identifiable et claire et parqué dans un seul fichier.
- Le composant est construit pour résoudre un problème spécifique.
- Le composant doit avoir des interfaces et cacher tous les détails qui ne sont pas nécessaires et sans le document le composant est inaccessible.

1-3- Architecture d'un composant logiciel :

L'architecture d'un composant logiciel spécifie ses entrées et ses sorties [23]

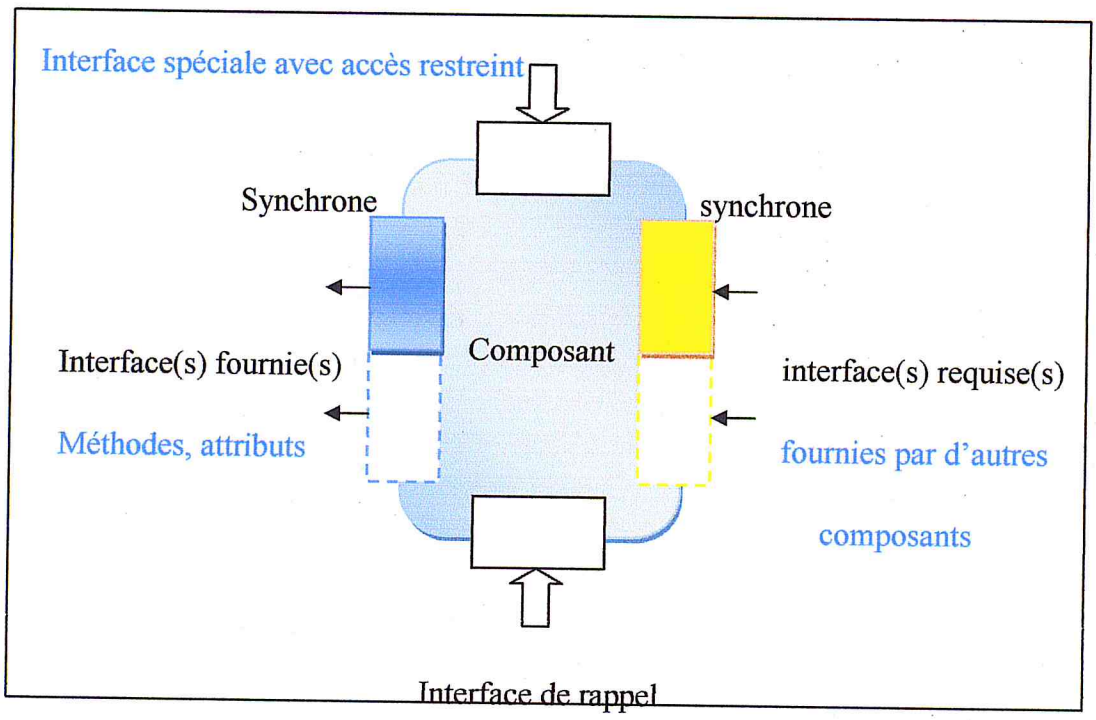

Figure 2 : Architecture d'un composant logiciel

1-4- développement d'une application à base de composants :

La construction d'applications à base des composants se décompose en trois étapes dont deux sont fondamentales : le développement de composants et l'assemblage.

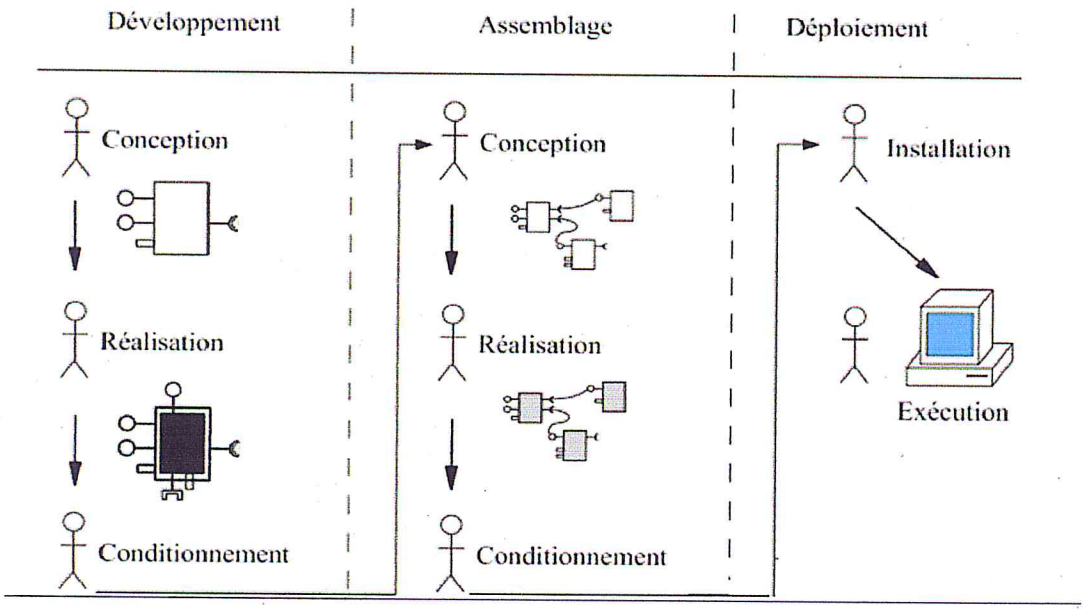

Figure 3 : Etapes de développement d'une application à base des composants

1-4-1 Développement :

Le développement de composants inclut les activités de conception, implémentation et conditionnement des composants.

Conception : la conception de composants inclut la spécification de la vue externe d'une classe de composant ainsi que de son comportement.

Réalisation : la réalisation de composants consiste à la création de l'implémentation de la classe de composant qui implémente les vues externe et interne.

Conditionnement : une classe de composant est introduite dans un paquetage de composant qui permet de réaliser le déploiement indépendant.

1-4-2 Assemblage :

L'assemblage de composants inclut les activités de conception, de réalisation et de conditionnement d'un assemblage.

Conception : La conception d'un assemblage est réalisée en étudiant la manière de composer un ensemble de composants préexistants pour créer une composition représentant soit une partie ou bien la totalité d'une application.

Réalisation : La réalisation d'un assemblage consiste à l'écriture du code permettant de réaliser la composition des instances de composants.

Conditionnement : Lors du conditionnement, un assemblage est introduit dans un paquetage d'assemblage qui peut inclure les composants employés dans l'assemblage ainsi que des ressources propres à l'assemblage telles que des fichiers de configuration ou des fichiers binaires (librairies, images, sons, etc.).

1-4-3 Déploiement :

Les étapes postérieures au développement et à l'assemblage sont les suivantes :

Déploiement : lors de l'installation, des paquetages contenant des compositions et des composants sont installés et des activités de configuration peuvent être réalisées.

Exécution : l'application est exécutée. A ce moment là, l'environnement d'exécution est actif.

1-5- Étude de divers modèles à composants

1-5-1 Entreprise Java Beans (EJB) :

Les EJB sont des composants serveurs donc non visuel qui respectent les spécifications d'un modèle édité par Sun, leurs but est de faciliter la création d'applications distribuées pour les entreprises. [5]

Ils fournissent également un modèle de programmation ayant pour objectif la réutilisabilité et la portabilité, ils visent tout d'abord à améliorer la productivité des développeurs en dissociant trois types d'activités : le développement client, le développement conteneur et le développement d'EJB. [6]

Les composants EJB sont archivés à l'aide d'un descripteur de déploiement. Ce descripteur se présente sous la forme d'un fichier XML qui indique aux serveurs d'applications comment déployer les *beans* contenus dans l'archive en définissant leurs caractéristiques [7]

1-5-2 COM+ :

Le modèle COM (Component Object Model) est un standard publié et conçu par Microsoft.com, une technologie Windows, importante et évolutive. Permet la construction de composants binaires réutilisables, accessibles à travers une ou plusieurs interfaces. Celles-ci permettent l'accès au composant qui peut être construit à partir de composants
existants par un processus de délégation ou d'agrégation. La définition des composants faite au niveau du binaire et sans modification du logiciel permet de ne pas se préoccuper du langage initial de programmation du composant.

On parle alors de DCOM (Distribueted COM), qui est l'extension de COM prenant en compte l'aspect distribué (il permet la communication entre objets situés sur des machines différentes)
Sous Windows 2000, DCOM change de nom pour devenir COM+. On peut dire alors que COM+ est l'évolution de COM. Dernièrement, Microsoft a développé un environnement de création d'application Web nommé « .NET» (prononcer dot-Net) .

COM+ fournit :

- la gestion des transactions
- le pooling d'instances
- le contrôle d'accès
- la communication en mode message

COM+ est propriétaire, et pas très flexible. [8]

1-5-3 .Net :

.Net est l'architecture de composants fournie par Microsoft pour la plate-forme
de développement Windows . L'objectif est de fournir un langage indépendant de la plate-forme, aussi bien pour le développement que pour l'exécution. Le Framework .Net repose sur le CLI (Common Language Infrastructure) .
L'inconvénient est que Microsoft fournit le Framework .Net que pour ses systèmes d'exploitations Windows et Windows CE et le code source du Framework n'est pas disponible[7].

1-5-4 CCM :

Le CORBA COMPONENT MODEL est un modèle pour définir un composant dans une architecture distribuée CORBA. Il est compatible avec les structures EJB (Enterprise Java Beans).

Ce modèle propose aussi bien une architecture pour le composant qu'une API pour l'implémentation CORBA. Il intègre aussi des interfaces pour la configuration, la définition de la composition des composants et un modèle pour le déploiement.

Le CCM propose toute une structure pour définir un composant, son comportement, son intégration dans un conteneur (application) et son déploiement dans l'environnement distribué CORBA. [9]

1-6- Conclusion :

Après une brève étude des modèles des composants, nous avons choisi comme modèle support pour l'implémentation de notre proposition le modèle EJB.

Ce choix est motivé par plusieurs raisons, non seulement leurs but est de faciliter la création d'applications distribuées pour les entreprises. En plus ce sont des composants JAVA, multi plate forme avec un accès local et à distance, portables, réutilisable, dé ployable, que l'on peut assembler et qui s'exécutent dans un conteneur (transaction, autorisation, persistance…).

CHAPITRE 2: LES COMPOSANTS EJB

2-1 Introduction :

Le modèle "*Enterprise Java Beans*" (*EJB*) a été proposé par Sun Microsystème en 1998. Ce modèle propose un ensemble de bibliothèques (classes) Java pour fabriquer des applications distribuées à base de petits serveurs logiciels. Les spécifications *EJB* fournissent un cadre et des règles pour construire ce type d'applications [10].

2-2 présentations des EJB :

Les EJB sont des composants et en tant que tels, ils possèdent certaines caractéristiques comme la réutilisabilité, la possibilité de s'assembler pour construire une application, etc. Les EJB s'exécutent dans un environnement particulier : le serveur d'EJB. Celui ci fournit un ensemble de fonctionnalités utilisées par un ou plusieurs conteneurs d'EJB qui constituent le serveur d'EJB. En réalité, c'est dans un conteneur que s'exécute un EJB et il lui est impossible de s'exécuter en dehors [5].

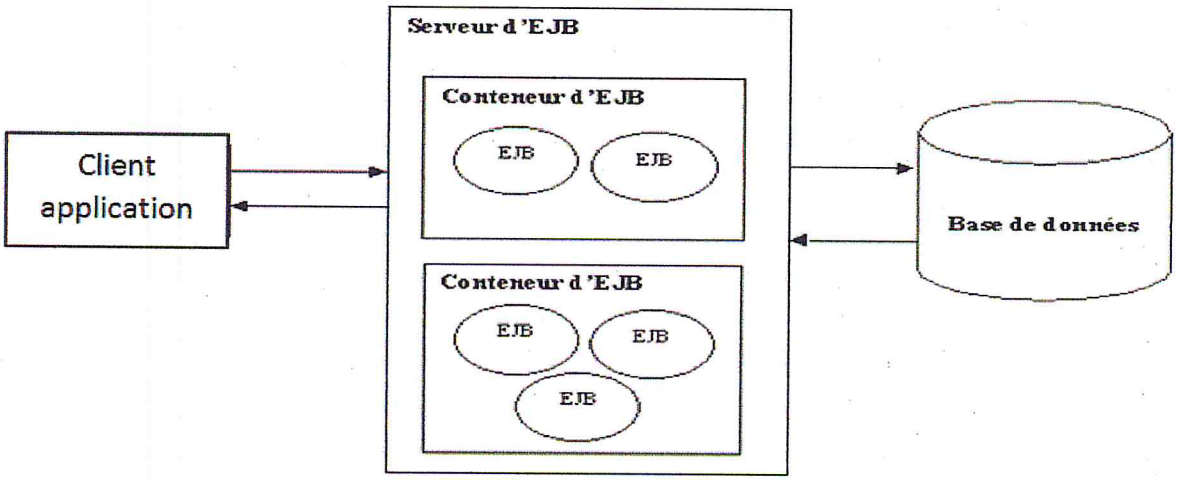

Figure 4 : présentation des EJB

Il existe plusieurs versions des spécifications des E.J.B: [3].

- 1.0 : publiée en mars 1998 intégrée dans J2EE
- 1.1 : publiée en décembre 1999 intégrée dans J2EE 1.2.
- 2.0 : publiée en septembre 2001 intégrée dans J2EE 1.3.
- 2.1 : publiée en novembre 2003 intégrée dans J2EE 1.4.
- 3.0 : publiée en décembre 2005 intégrée dans javaEE5.
- 3 .1 : publiée en fin 2008 intégrée dans javaEE6.

2-3 Les différents types d'EJB :

Les trois types de composants sont les composants de type session, les composants de type entité et les composants de type message (session bean, entity bean, message driven bean).[5]

2-3-1 Session bean :

Utilise l'interface SessionBean. Ce sont des entités non persistantes.

Il en existe deux types :

- session avec état : Le middleware mémorise le contexte d'exécution entre le client et le serveur grâce au conteneur EJB. Chaque contexte client est isolé des autres et l'état du serveur est maintenu entre chaque appel du client.
- Le deuxième type est une session sans état. Le composant exécute le service sans tenir compte ni des appels précédents ni des autres clients.

2-3-2 Entity beans :

Des composants persistants. Leurs états sont sauvegardés sur un support tel qu'un fichier ou une base de données. La persistance peut être réalisée soit par le conteneur (Container Managed Persistence) soit par le composant lui-même (Bean Managed Persistence). Un composant entité peut être, de plus, mis en sommeil (passivé) par le conteneur en cas de non utilisation ou de problème de ressource mémoire.

2-3-3 Message driven beans :

Les plus simples. L'application client se connecte au service de diffusion du middleware. Ensuite, des files de diffusions de messages peuvent être créées et utilisées. Le composant implante une seule méthode nommée on Message () qui traite l'arrivée de tous les messages. Il est possible d'utiliser le service de transaction sur les queues de messages.

2-4 Le développement d'un EJB :

Le cycle de développement d'un EJB comprend :

- La création des interfaces et des classes du bean : nécessite la création d'au minimum deux interfaces et une classe pour respecter les spécifications de Sun : la classe du bean, l'interface remote et l'interface home.
- le packaging du bean sous forme de fichier archive jar
- le déploiement du bean dans un serveur d'EJB
- le test du bean

2-5 Le choix de notre modèle support :

Jusqu'à la version 2.1, les EJB bénéficiaient d'une mauvaise réputation à cause de leur complexité. Depuis la version 3.0, leur mise en place est grandement simplifiée: plus besoin de descripteur de déploiement, plus de Home interface, plus besoin de faire un lookup via JNDI pour récupérer des EJBs, il suffit de passer par l'annotation @EJB, etc. nous avons choisi comme modèle support pour l'implémentation de notre proposition le modèle EJB 3.1 à cause des nouvelles fonctionnalités qui leur permet de devenir très intéressant même pour des applications de taille moyenne.

2-6 Les nouvelles fonctionnalités des EJB 3.1 :

2-6-1 Les interfaces locales sont optionnelles :

Les interfaces sont un excellent moyen pour limiter le couplage et assurer la testabilité : cependant dans certains cas, elles ne sont toujours pas nécessaires, avec la version 3.1, il n'est plus nécessaire de définir une interface Local pour les EJB session : la classe de l'EJB session peut être directement annotée avec @Stateless ou @Stateful. Rendre les interfaces pour les EJB session optionnelles permet à un EJB session d'être un simple POJO (Plain Old Java Object) traduisible par un Objet java simple.[5]

2-6-2 Les EJB Singleton

La version 3.1 des EJB propose un nouveau type d'EJB Session nommé Singleton, c'est un nouveau composant qui ressemble à un EJB Session mais qui ne peut avoir qu'une seule instance dans un conteneur pour une application.

Un EJB singleton est utilisé principalement pour partager ou mettre en cache des données dans l'application. L'avantage des EJB Singleton c'est qu'ils offrent tous les services d'un EJB : sécurité, transaction, injection de dépendances, gestion du cycle de vie et intercepteurs, ...

Les EJB de type Singleton se définissent avec l'annotation @Singleton et ils permettent d'ajouter de nouvelles fonctionnalités aux EJB :

- Exécution de code au lancement ou à l'arrêt de l'application
- Partage de données avec gestion des accès concurrents [5].

2-6-3 EJB Lite :

Le but d'EJB Lite est de proposer une version légère d'un conteneur d'EJB et de permettre de standardiser un conteneur d'EJB.

Les EJB Lite sont un sous-ensemble de l'API EJB qui permet une utilisation des EJB locaux en dehors d'un conteneur EJB, EJB Lite propose les fonctionnalités suivantes :

- Support des EJB de type session (stateless, stateful, et singleton)
- Support des EJB avec interface local ou sans interface
- L'injection
- Les intercepteurs
- La sécurité et les transactions (Container Managed Transactions et Bean Managed Transactions).[5]

2-6-4 La simplification du packaging :

Avant leur version 3.1, les EJB devaient être packagés dans un archive de type jar dédiée. Comme une application d'entreprise est généralement composée d'une partie IHM sous la forme d'une webapp packagée dans un archive de type war,

il faut regrouper les archives war et jar dans un archive de type ear, le packaging des EJB avait été simplifié dans la version 3.0 en rendant le descripteur de déploiement optionnel. Cependant, le packaging devait toujours être fait de façon modulaire : un pour la partie web dans une archive de type war et un pour la partie EJB dans un archive de type jar, le tout regroupé dans un archive de type ear.

Il suffit alors de définir la source de données dans le conteneur ou le serveur d'applications dans lequel le war est déployé et d'utiliser JNDI pour obtenir une référence sur cette source de données, cette facilité de packaging favorise l'utilisation des EJB dans les petites et moyennes applications web, il est cependant recommandé de réserver ce type de packaging pour des applications simples et de conserver l'usage des archives de type ear pour des applications complexes. [5]

2-6-5 Les améliorations du service Timer :

La version 2.1 des EJB propose le service Timer qui permet l'invocation de callbacks dans un contexte transactionnel selon des contraintes temporelles spécifiées, la version 3.1 des EJB enrichit le service Timer avec :

- La possibilité de créer un timer par déclaration en utilisant l'annotation @Schedule ou le descripteur de déploiement
- L'enrichissement de l'interface TimerService pour créer un timer par programmation avec les mêmes fonctionnalités que par déclaration.

2-6-6 La standardisation des noms JNDI :

La standardisation du nom JNDI par la spécification permet de définir clairement comment le nom global JNDI (global JNDI name) doit être défini, résolvant ainsi les problèmes de portabilité pour retrouver des références vers des composants ou des ressources.

2-6-7 L'invocation asynchrone des EJB session

L'invocation de traitements asynchrones est relativement fréquente dans les applications d'entreprises mais jusqu'à la version 3.0 incluse des EJB aucune solution standard n'était proposée pour ce besoin.

Comme les threads ne peuvent pas être utilisés dans les EJB, une façon couramment utilisée de permettre une invocation asynchrone d'un EJB est de passer par un message JMS traité par un EJB de type MDB. Cependant, le rôle principal de JMS est l'échange de messages et pas l'invocation de fonctionnalités de façon asynchrone, mais cette solution ne permet pas facilement d'avoir un retour à la fin des traitements réalisés.

2-7 Conclusion :

Ce chapitre constitue la charpente sur laquelle nous basons pour introduire la notion des composants EJB, il nous a permis de présenter notre choix qui est les EJB 3.1et de mettre en évidence leurs types, leurs fonctionnalités avancées et leur services. Ainsi nous consacrons le prochain chapitre de ce rapport à présenter l'outil de modélisation et d'assemblage.

PARTIE 2: CONCEPTION ET IMPLEMENTATION

CHAPITRE 3: CONCEPTION

3-1 Introduction :

Une application n'est pas composée uniquement d'instructions de programmation, elle est aussi construite à partir d'un ensemble d'éléments (organisation, concepts, techniques et outils) qui permettent de la spécifier, de la concevoir, de la tester, de la qualifier, de l'exploiter et de la faire évoluer.

Afin de développer notre application, nous avons choisi le langage UML (Unified Modeling Langage) en raison de sa qualité de langage de modélisation à objet et son formalisme relativement simple, compréhensible aussi bien par les informaticiens que par les non informaticiens, imposant toute la rigueur nécessaire au bon déroulement du projet[11].

Nous avons travaillé avec deux vues (quatre diagrammes)

- **Vue statique :**
  - diagramme de cas d'utilisation : décrit les besoins utilisateurs.
  - Diagramme des classes : définit la structure statique.
- **Vue dynamique :**
  - Diagramme de séquence : scenarios et flots de messages
  - Diagramme d'activité : montre l'enchaînement des activités qui concourent au processus.

3-2 Présentation des acteurs :

Acteurs :

- Administrateur.
- Responsable carrière.
- Responsable formation.
- Responsable développement.

Rôles :

- Administrateur : chargé de la mise à jour de la base de données tel que les postes, les services…… .
- Responsable carrière : chargé de gérer la carrière des employés.
- Responsable formation : organise les cursus de formation et définit le plan annuel de formation.
- Responsable développement : chargé de gérer les sanctions, les reclassements et les évaluations des employés.

3-3 Diagramme cas d'utilisation :

3-3-1 Diagramme de cas d'utilisation général :

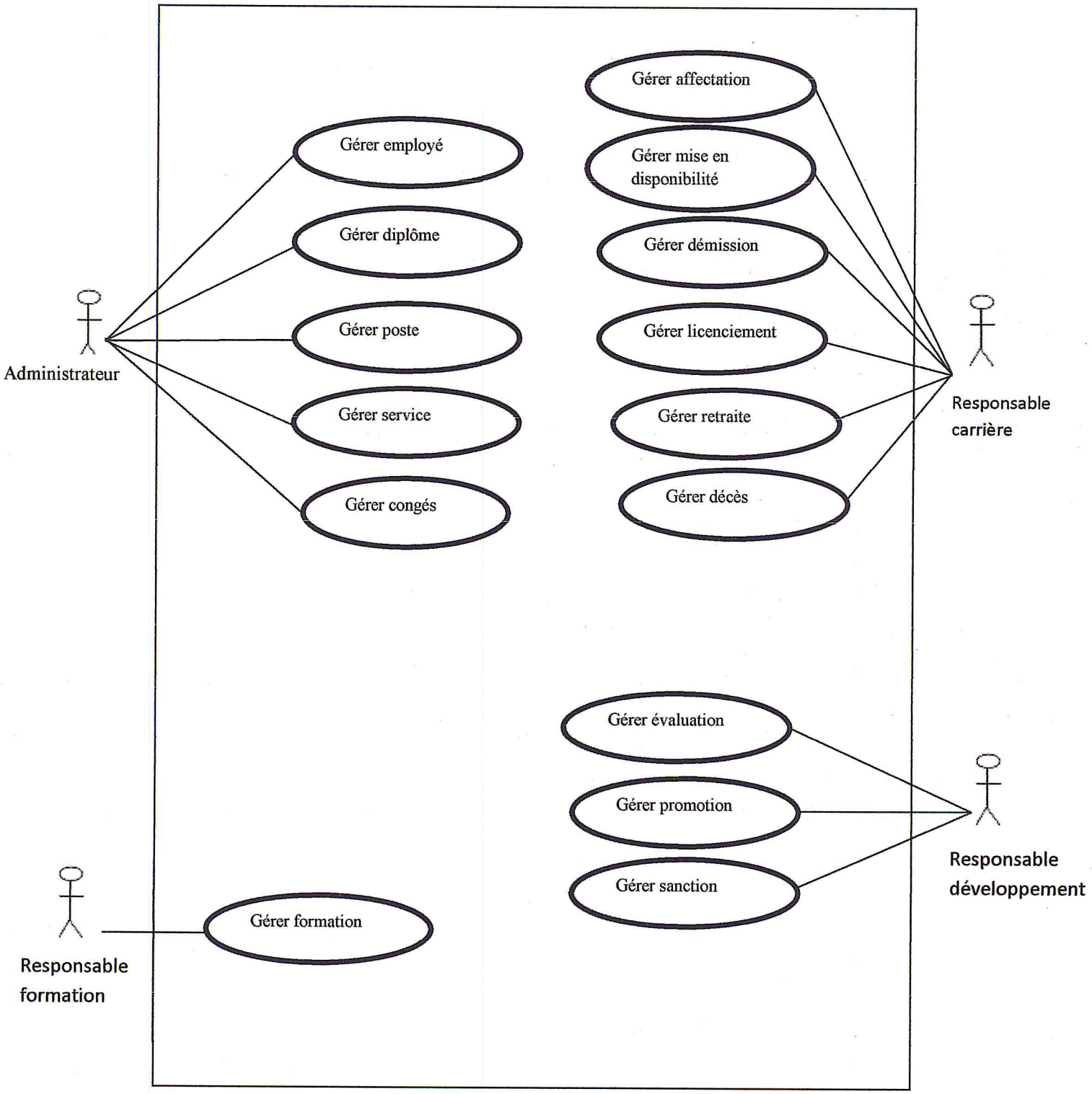

Figure 5 : Diagramme de cas d'utilisation

3-3-2 Diagramme de cas d'utilisation formation :

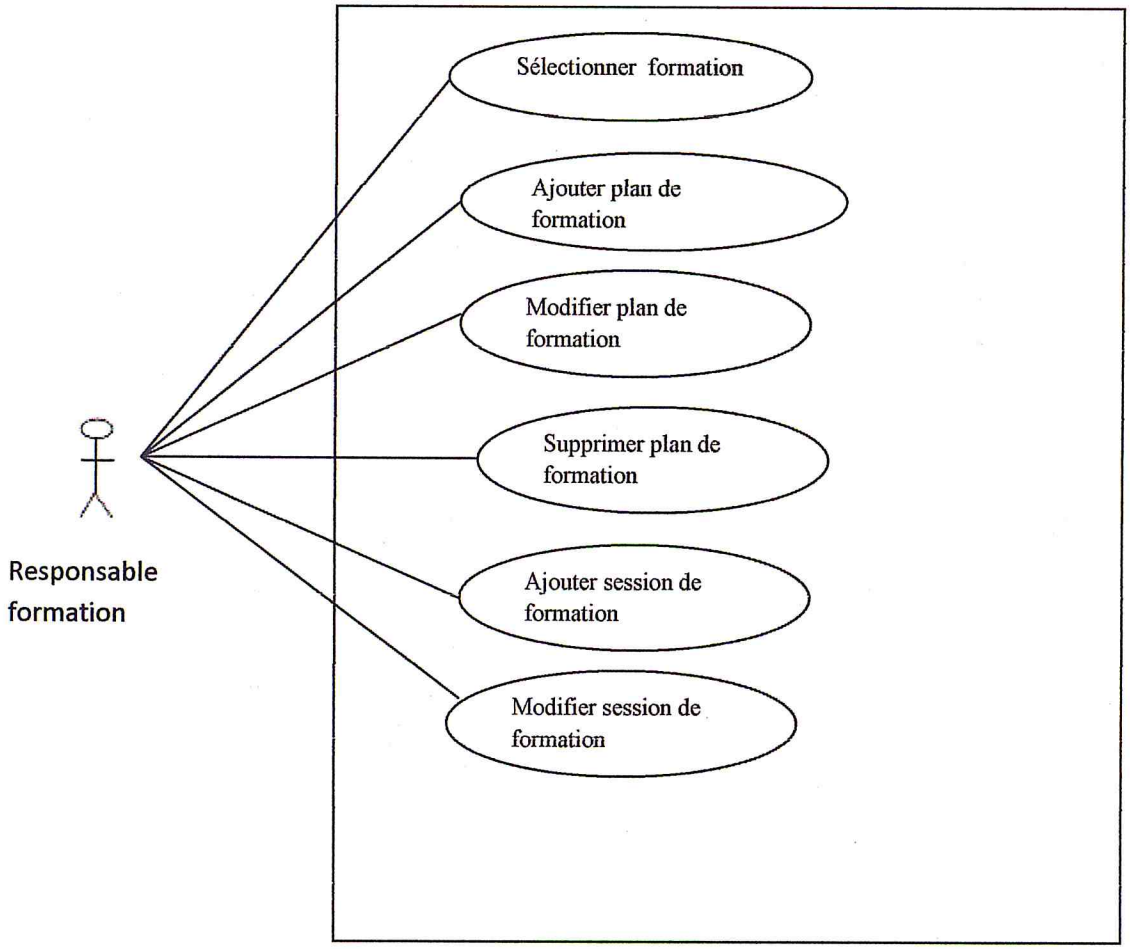

Figure 6 : Diagramme de cas d'utilisation

3-3-3 Diagramme de cas d'utilisation détaillé de l'évaluation, la promotion et la sanction :

Puisque c'est la partie traitée en détail dans notre mémoire

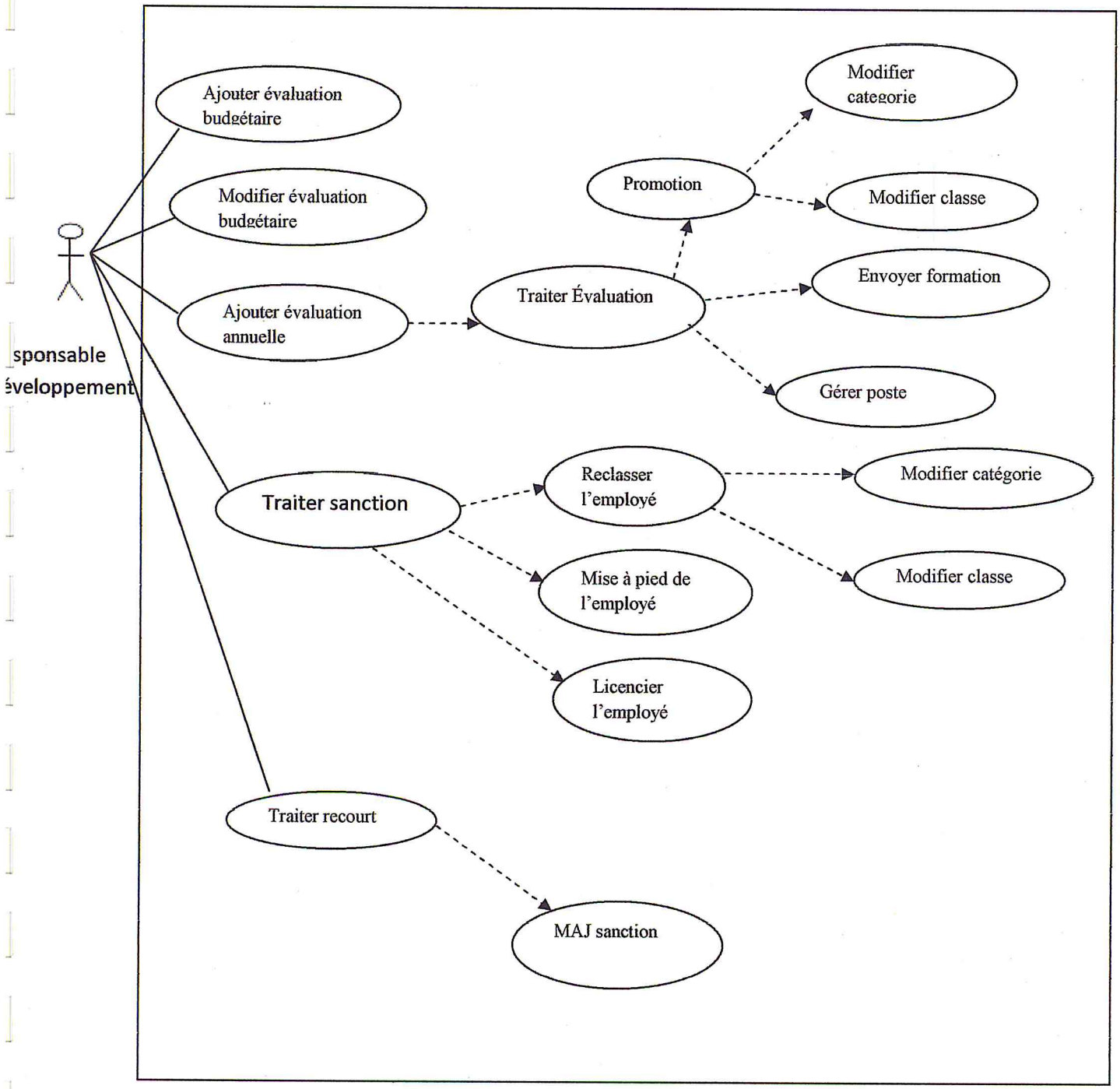

Figure 7 : de cas d'utilisation détaillé de l'évaluation, la promotion et la sanction

3-4 Description des scenarios :

3-4-1 Scenario de l'évaluation :

- **Responsable développement :**

Résumé : Traitement des évaluations.

Acteur : Responsable développement.

Description du scenario nominale :

Evénement déclencheur : besoin d'ajouter / modifier /supprimer évaluations.

<1> : Le système affiche le menu ;

<2> : Responsable développement sélectionne le menu « évaluations » ;

<3> : Responsable développement Choisi l'opération « nouvelle évaluations» ;

<4> : Le système demande de remplir le formulaire.

<5> : Responsable développement saisit les informations.

<6> : le système calcule la note annuelle.

< 7 > : le système affiche la note annuelle.

<8> : le responsable développement demande le teste de proposition.

<9> : Le système calcule la moyenne des notes d'évaluation des trois dernières années.

<10> :.le système effectue le teste de proposition.

<11> : le système affiche la moyenne et la liste des propositions.

<12> : le responsable développement fait le choix de proposition

<13> : le système teste le choix et affiche le choix possible.

<14> : le responsable développement valide le choix.

<15> : le système enregistre le choix et modifie les paramètres de l'employé

<16> : le système affiche la décision d'évaluation.

<17> : le responsable développement demande au système d'imprimer la décision d'évaluation

<18> : le système imprime la décision d` évaluations.

<29> : Le système affiche à nouveau la fenêtre « évaluations ».

3-4-2 Scenario de sanction:

- **Responsable sanction :**

Résumé : Traitement des sanctions.

Acteur : Responsable développement.

Description du scenario :

Evénement déclencheur : besoin d'ajouter / modifier /supprimer sanction.

<1> : Le système affiche le menu ;

<2> : Responsable développement sélectionne le menu « sanction » ;

<3> : Responsable développement Choisi l'opération « nouvelle sanction» ;

<4> : Le système demande de saisir les informations.

<5> : Responsable développement saisit le matricule de l'employé et sélectionne sa faute.

<6> : le système fait le traitement et affiche le degré de la faute.

<7> : le système affiche les sanctions qui correspondent au degré de la faute.

< 8 > : Responsable développement choisi la sanction.

<9> : Le système enregistre la sanction et modifie les paramètres de l'employé.

<10> : Le système affiche la décision de sanction.

<11> : Responsable développement demande au système d'imprimer la décision de sanction.

<12> : le système imprime la décision de sanction.

3-4-3 Scenario de formation:

- **Gérer formation :**

Résumé :

Le module gestion de formation permet l'ajout, la modification, la suppression de plan et session de formation, un plan de formation permet de planifier les sessions de formations.

Acteur : responsable formation.

Description du scenario nominale :

Evénement déclencheur : besoin d'ajouter une formation.

<1> : Le système affiche le menu ;

<2> : Le responsable de formation sélectionne la formation dans la liste des formations disponible.

<3> : Le responsable de formation saisit l'année du plan de formation.

<4> : Le responsable formation sélectionne et positionne les sessions de formations.

<5> : le responsable formation valide la fiche du plan de formation.

< 6 > : le système vérifie la validité des données saisies.

<7> : Le système enregistre le plan de formation.

<8> : Le système informe de la réussite de l'opération d'enregistrement.

3-5 Diagrammes de séquence :

3-5-1 Diagramme de séquence « Nouvelle évaluation»

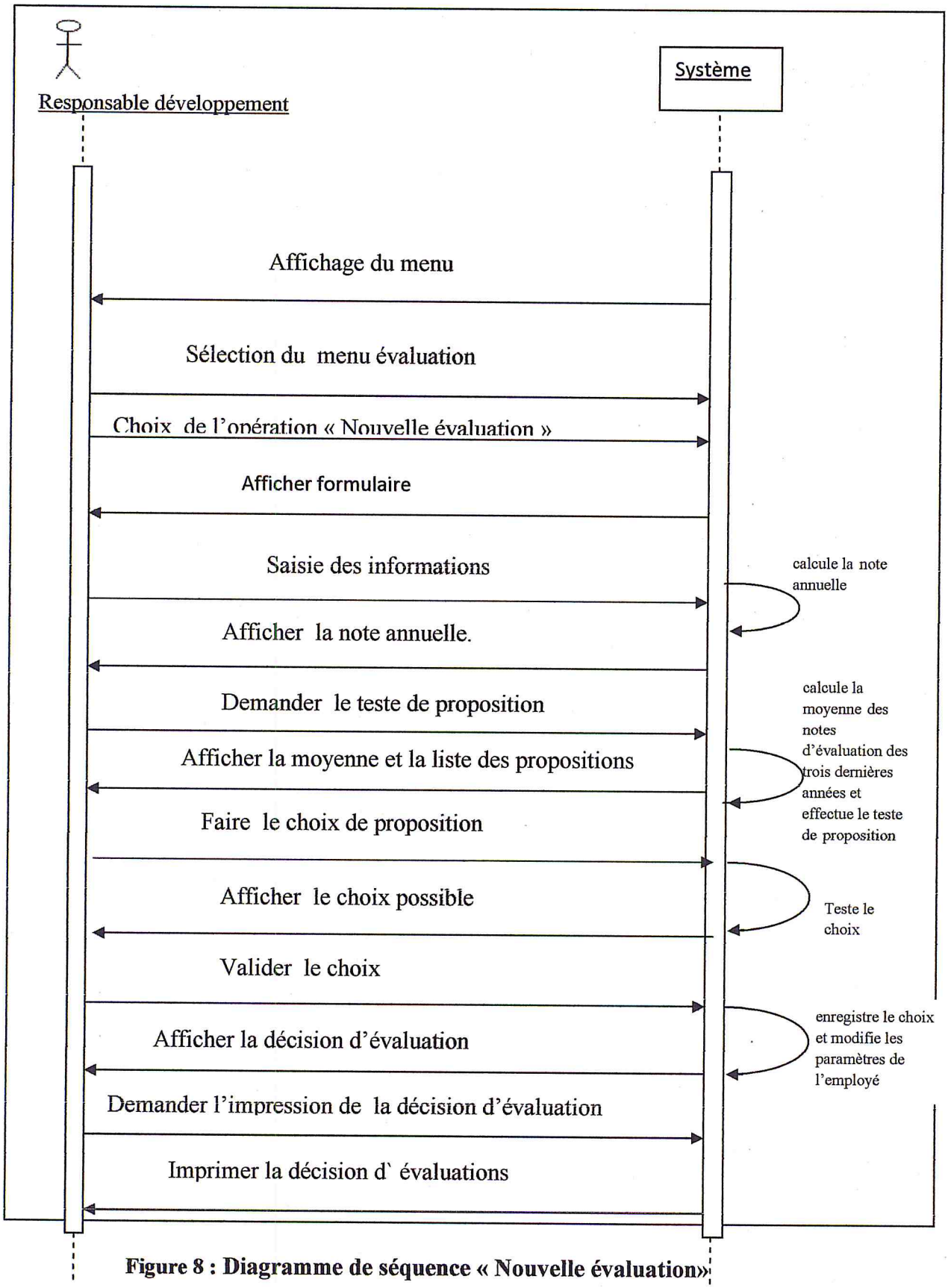

Figure 8 : Diagramme de séquence « Nouvelle évaluation»

3-5-2 Diagramme de séquence « Nouvelle promotion» :

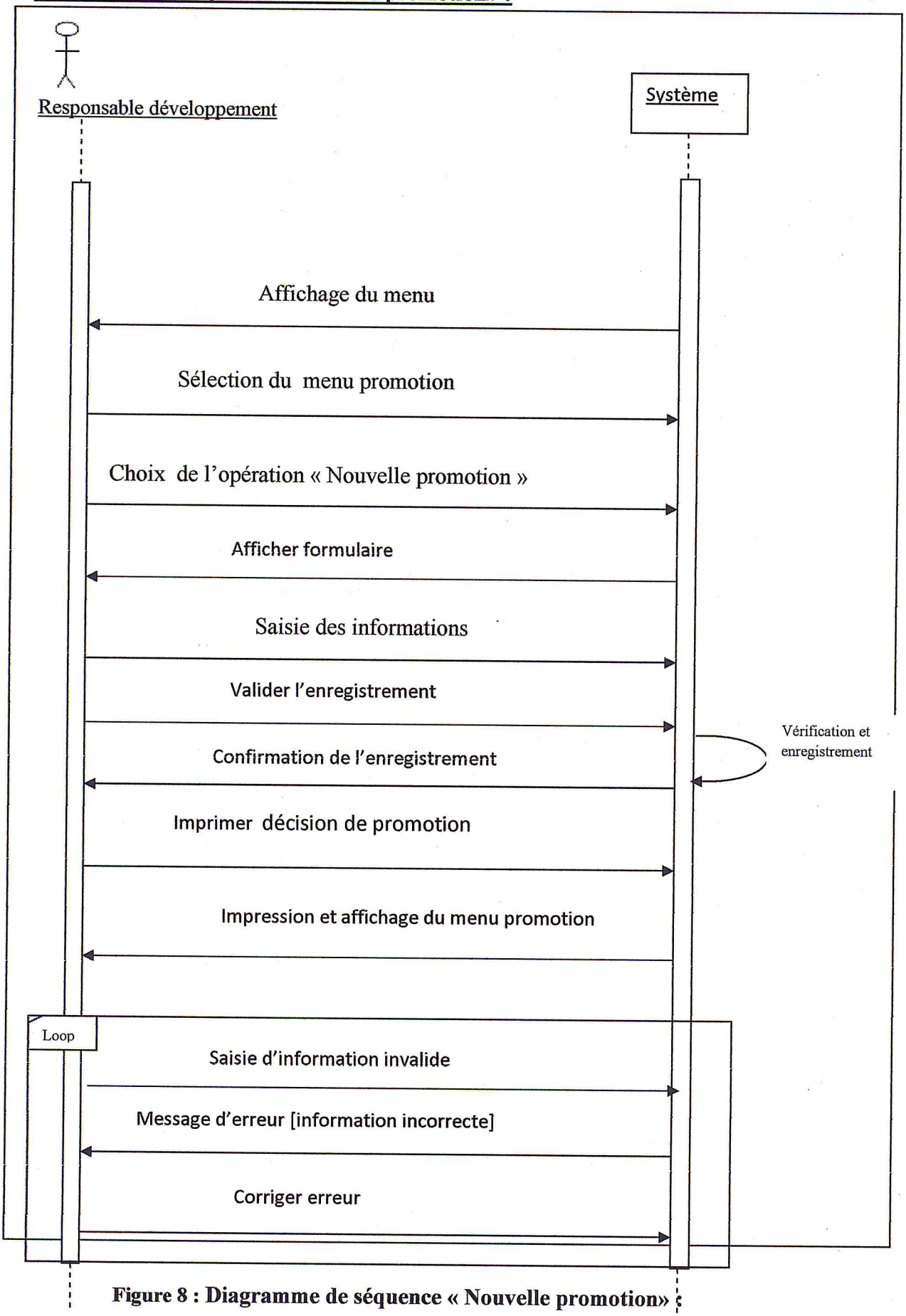

Figure 8 : Diagramme de séquence « Nouvelle promotion»

3-5-3 Diagramme de séquence « Nouvelle sanction» :

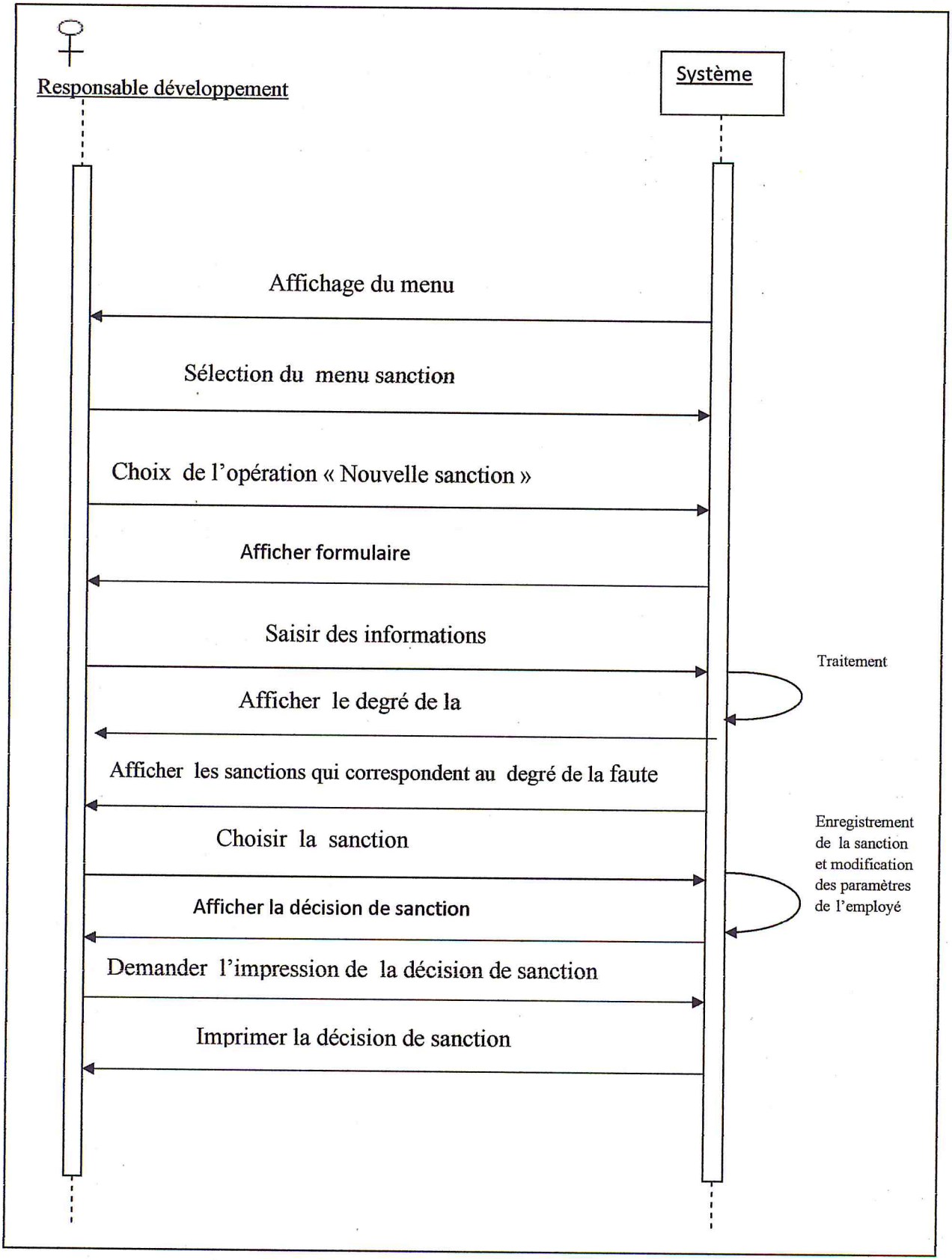

Figure 10 : Diagramme de séquence « Nouvelle sanction»

3-6 Description des classes :

classes	Attributs
Retraite	NumRetraite (int) LibelleRetraite (varchar) DateRetraite (date)
Congé	NumConge (int) AnneeConge (year) DateDebutConge (date) DateFinConge (date) DateReprise (date)
Affectation	DateAffectation (date)
Licenciement	NumLicenciement (int) MotifLicenciment (varchar) DateLicenciment (date)
Service	CodeService (varchar) LibelleService (varchar)
MisEnDisponibilite	NumDisponibilite (int) MotifDisponibilite (varchar) DureeDisponibilite (varchar) DateDisponibilite (date)
Demission	NumDemission (int) MotifDemission (varchar) DateDemission (date)

Diplôme	CodeDiplome (int) LibelleDiplôme (varchar) DateDiplôme (date)
Formation	NumFormation (int) LibelleFormation (varchar) DureeFormation (date) CoutFormation (float)
Session	IdSession (varchar) DateDebut (date) DateFin (date)
Sanction	NumSanction (int) motif Sanction (varchar) observationSanction (varchar) dateSanction (date)
reclassement	NumReclassement (int) LibelleReclassement (varchar) AnneeReclassement (year) Categorie (varchar) Classement (varchar)
Employe	Matricule (int) NomE (varchar) PrenomE (varchar) DateNaissE (date) SituationFam (varchar) NbrEnfant (int) ServiceNational (varchar) SalairBase (float) Sexe (varchar) Adresse (varchar) Nationnalite (varchar) Grade (varchar) NomJeuneFille (varchar) DateRecrutement (date)

Deces	NumDeces (int) LibelleDeces (varchar) LieuDeces (varchar) NumCertificatDeces (int) DateDeces (date)
Poste	CodePoste (varchar) LibellePoste (varchar)
NiveauQualification	LibelleNiveau (varchar)
MiseAPied	NumMisePied (int) NbrJourMisePied (int) DateDebut (date) DateFin (date)
Evaluation	NumEvaluation (int) Note (int) Annee (int)
Inscription	Etat (varchar)
Organisme	LibelleOrganisme (varchar)
PlanFormation	AnneeFormation (year)

3-7 Diagramme de classe :

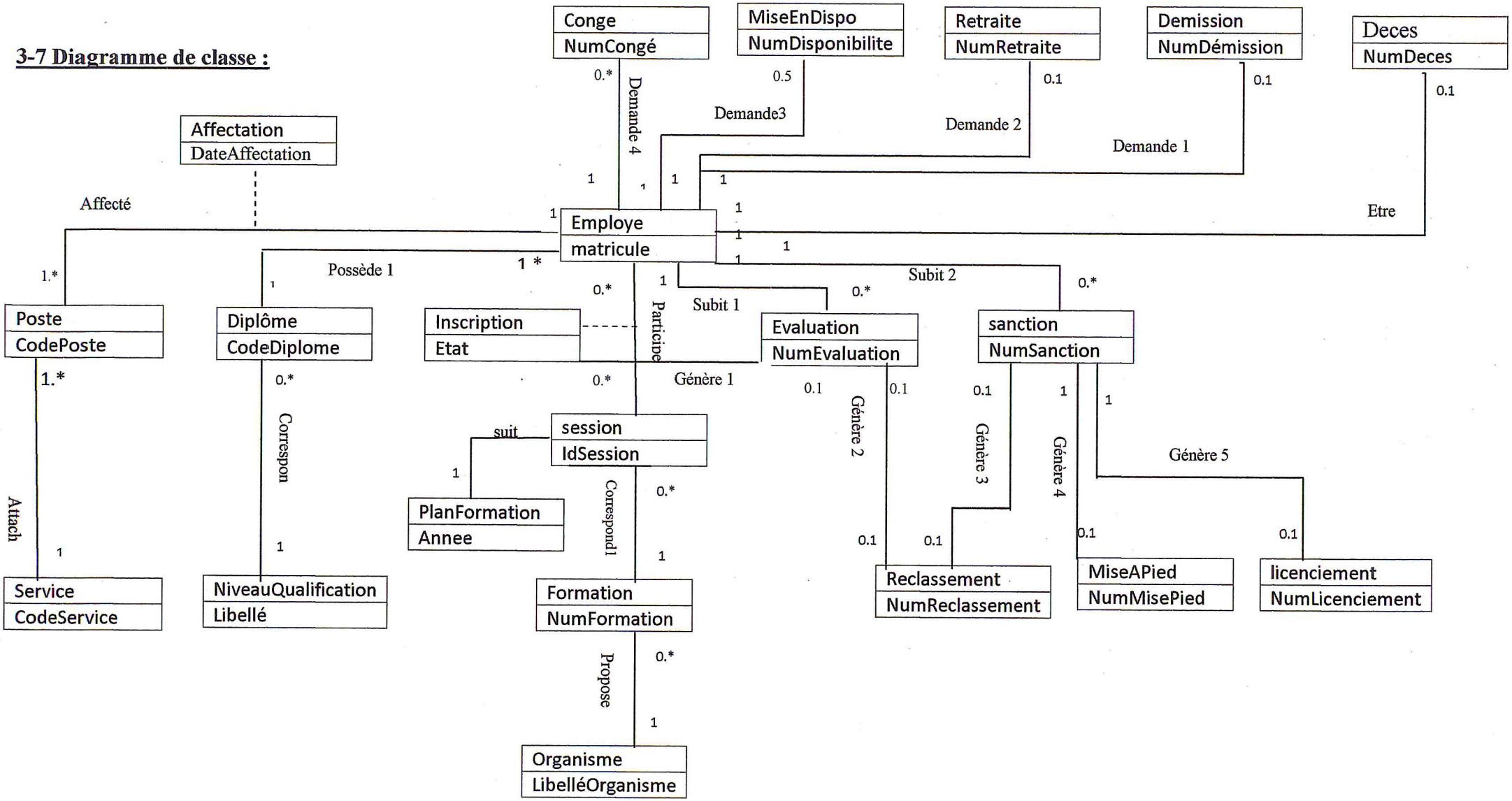

Figure 11 : Diagramme de classe

3-8 Extraction des EJB :

Partie 1 : Gestion des employés

Le but de cette partie est de pouvoir gérer les informations personnelles des employés, poste affecté, le service attaché, le diplôme possédé, et le niveau de qualification correspondant.

Ainsi que les opérations des mises à jour des tables « Employé, Diplôme, Niveau de qualification, Poste, Service, Affectation ».

1/ Le module persiste :

- **Extraction des EJB persistants :**
  A partir du diagramme de classe établie précédemment nous extractions les EJB persistants.

  Partie 1 : Gestion des employés.

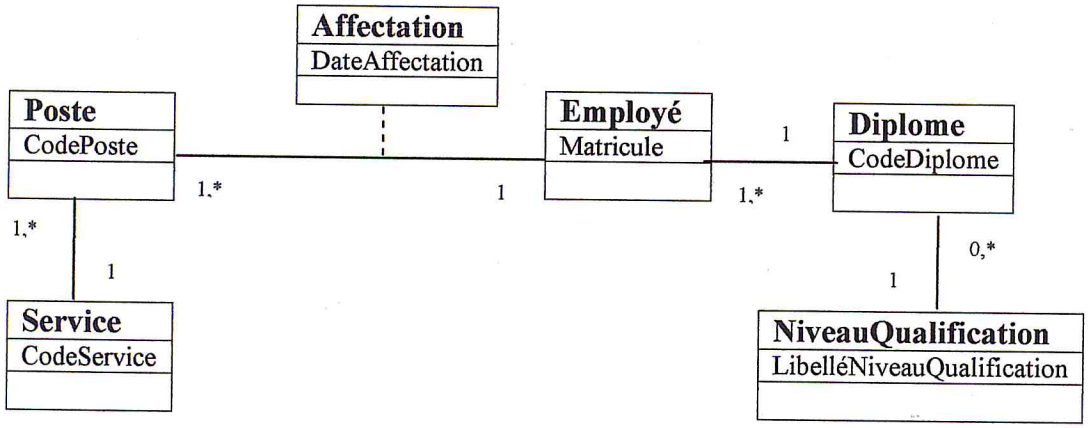

Figure 12 : Diagramme de classe « Gestion des employés »

Les EJB persistant de ce diagramme seront :

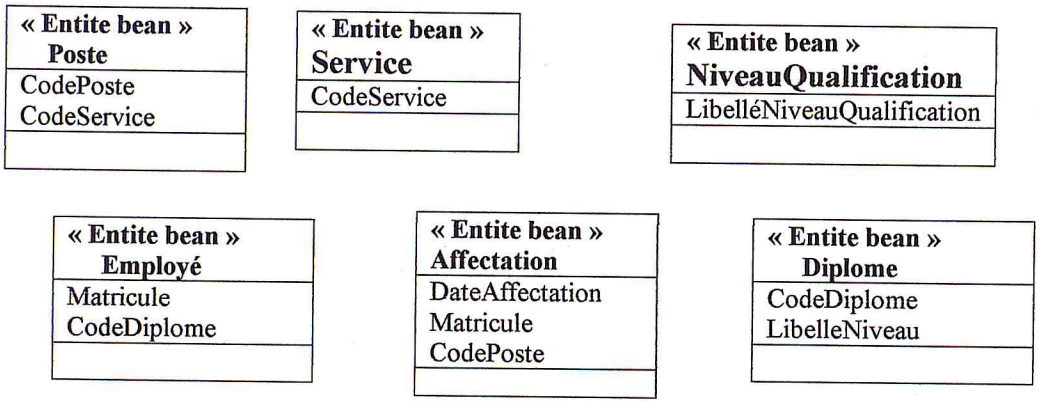

Figure 13 : Les EJB 1

Partie 2 : Gestion des formations.

Le but de cette partie est la gestion des formations effectuées au sein de Sonelgaz SDC « gérer les inscriptions des employés au session de formation, gérer les plans de formation , les formations et les organismes de formation ».

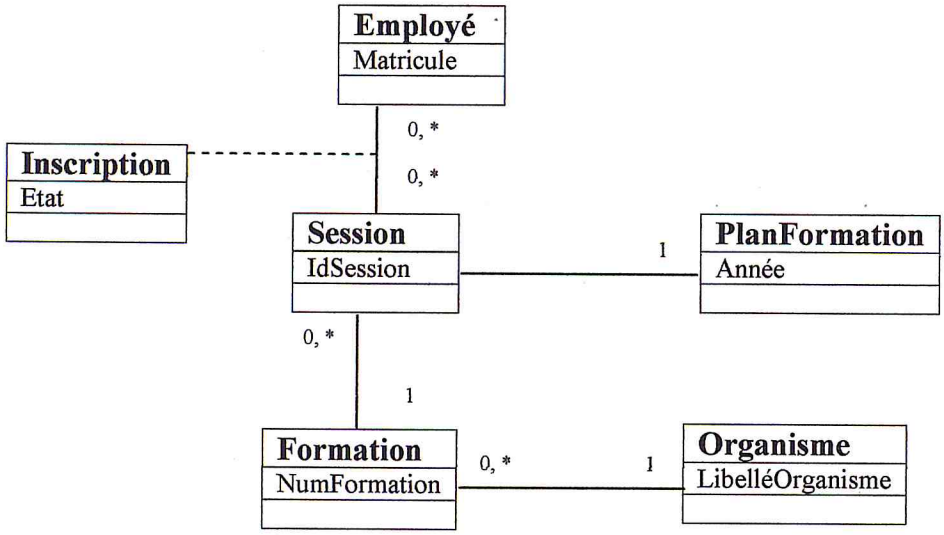

Figure 14 : Diagramme de classe « Gestion des formations »

Les EJB persistant de ce diagramme seront :

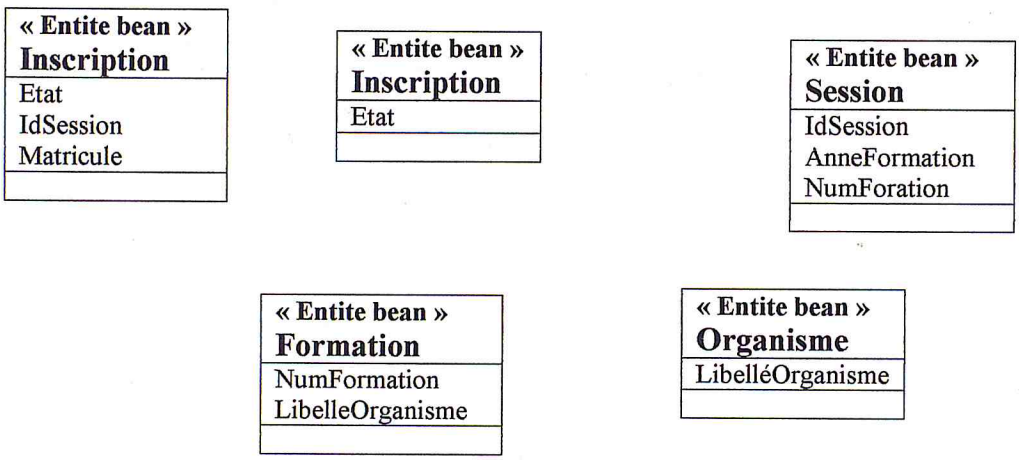

Figure 15 : EJB 2

Partie 3 : Gestion des évaluations

Cette partie a pour but de gérer les évaluations annuelles des employés, effectuer les opérations de mise à jour et d'autre opérations comme : Calcule d'évaluation annuelle (Note pondérée), calcul d'évaluation du rôle (Note pondérée), calcul d'évaluation globale de chaque employé à partir des évaluations annuelle et de rôle , test de proposabilité (si la moyenne d'évaluation globale des 3 dernières années est supérieure ou égale à 13.5 alors cet employé est proposable soit pour un reclassement " ajouter catégorie ou ajouter classement " ou bien pour une formation.

- Pour ajouter une catégorie pour un employé il doit rester au moins 3 ans sur la même catégorie.
- Pour ajouter un classement pour un employé il doit rester au moins 2 ans dans le même classement).

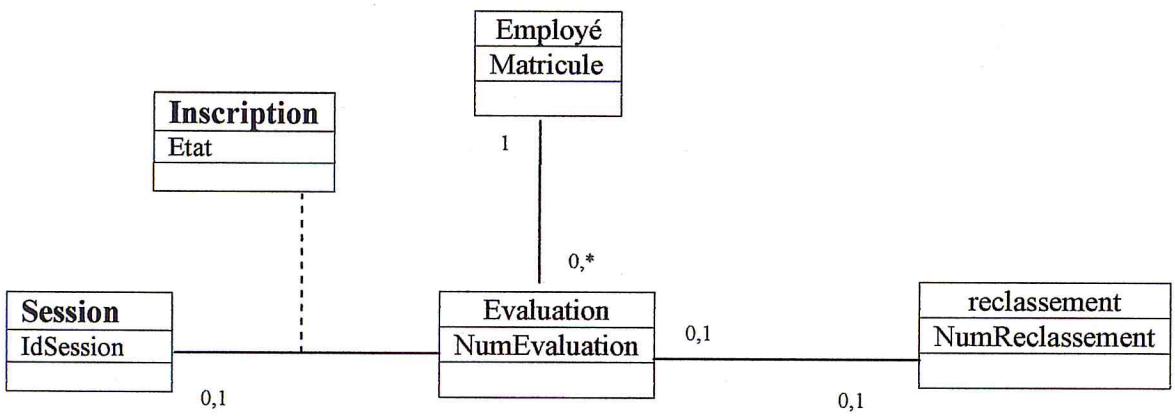

Figure 16 : Diagramme de classe « Gestion des évaluations »

Les EJB persistant de ce diagramme seront :

Figure 19 : EJB 3

Partie 4 : Gestion des sanctions

Cette partie gère les sanctions des employés et leur résultat (reclassement, mise à pied, licenciement).

Chaque employé commit des erreurs sera sanction selon le degré de ses fautes et qui sont les suivantes :

1. Les sanctions du 1er degré : répriment les fautes professionnelle du premier degré et se matérialisent par l'une des mesures précisées ci-après :
   - L'avertissement écrit.
   - Le blâme.
   - La mise à pied de 1 jour à 8 jours ouvrables.

   Sont considérés fautes du 1er degré, notamment les actes ou faits suivants :

   - Non-respect de l'horaire de travail.
   - Absences irrégulières injustifiées.
   - Absence pendant les heures de travail.
   - Interruption du cycle de travail sans raison valable.
   - Non respect des règles de propreté, de sécurité, d'hygiène et de médecine du travail.
   - Non respect de la discipline générale de travail, y compris le pointage.
   - Accès non autorisé sur les lieux de travail, installations et ouvrages tendant à perturber l'activité.
   - Non respect par négligence des procédures et normes de travail et de gestion.
   - Exécution des instructions de personnes physique ou morales autre que celles explicitement désignées par sa hiérarchie professionnelle ou sollicitées, et non respect de la voie hiérarchique.
   - Absence d'ordre de mission en cas de déplacement en dehors des lieux habituels de travail.
   - Défaut de tenue du carnet de bord pour le personnel conduisant des véhicules et engins.
   - Non-réalisation du programme de travail dans les délais.

2. Les sanctions du 2éme degré : répriment les fautes professionnelles du 2eme degré, elles se traduisent par :
   - la mise à jour de 9 à 30 jours ouvrables.

   Sont considérés fautes du 2eme degré, notamment les actes ou faits suivants :

   - récidives aux fautes du premier degré.
   - Non-respect des normes, objectifs ou rendement liés au poste de travail sans conséquence ou répercussion sur les couts, les délais ou l'organisation du travail.
   - Refus injustifié de suivre une action de formation programmée et inscrite au plan de formation.

- Divulgation, diffusion ou propagation de : note, document ou toute information non autorisée.
- Refus de participer à un séminaire ou stage de formation sans motif valable.

- Dégâts causés involontairement aux matériels et véhicules de service.

1. Les sanctions du 3éme degré : répriment les fautes professionnelles du 3eme degré, elles se traduisent par l'une des mesures précisées ci-après :
   - La rétrogradation au plus de deux catégories.
   - Licenciement.

   Sont considérés fautes du 3eme degré, notamment les actes ou faits suivants :

   - Les récidives aux fautes du 2eme degré.
   - Les manquements au devoir de loyauté à l'égard de l'entreprise.
   - Les manquements à l'obligation d'impartialité dans l'exercice de leurs fonctions.
   - Non application des procédures en vigueur de passations des marchés.
   - Refus d'effectuer une permanence sans motifs valables.

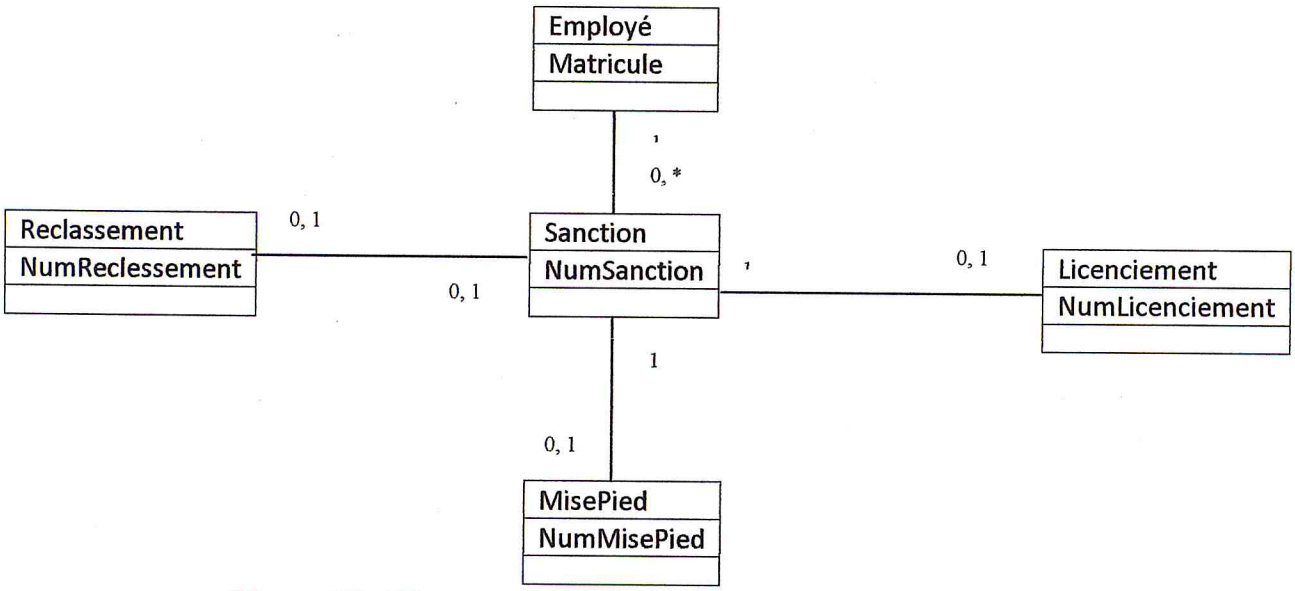

Figure 18 : Diagramme de classe « Gestion des sanction »

Les EJB persistant de ce diagramme seront :

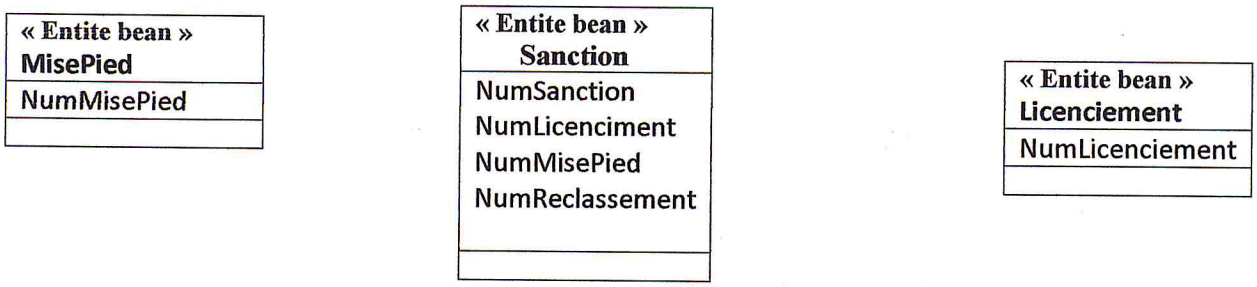

Figure 19 : EJB 4

Partie 5 : Gestion des départs.

Cette partie traite les départs des employés : congé, mise en disponibilité, retraite, démission, décès.

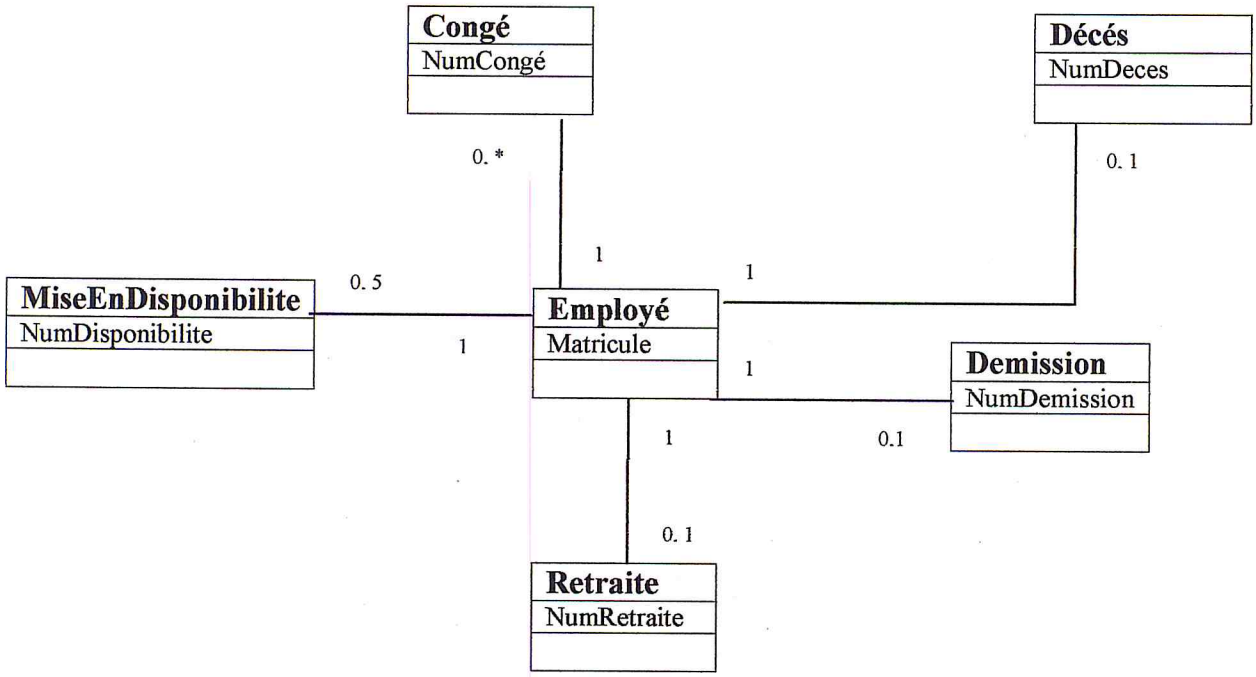

Figure 20 : Diagramme de classe « Gestion des départs »

Les EJB persistant de ce diagramme seront :

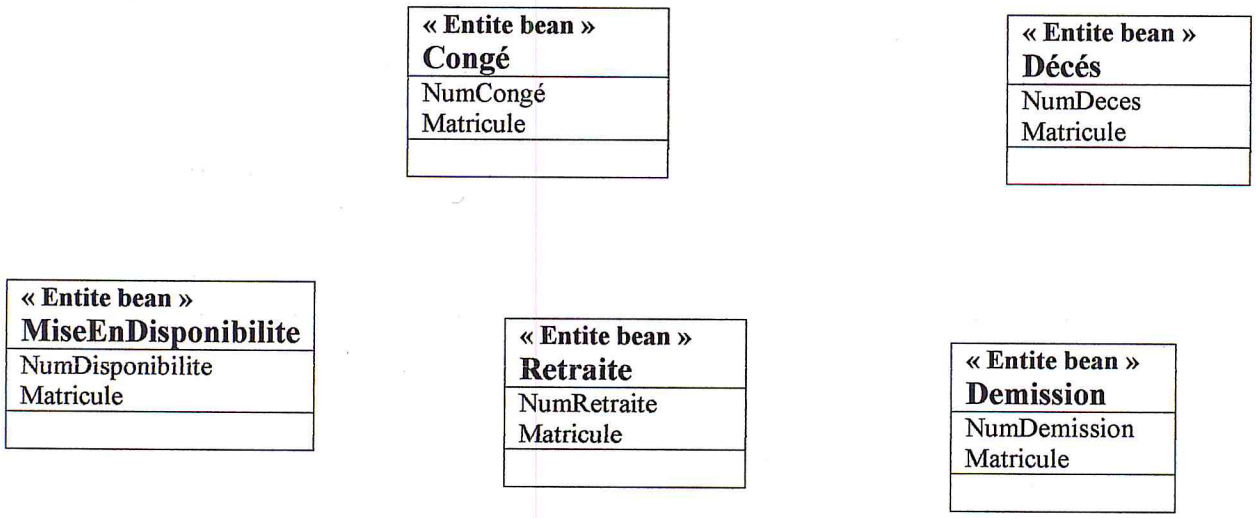

Figure 21 : EJB 5

2/ Le module session :

Le module session comporte un EJB de type session que nous l'appelons Façade de EJB.

- Générer les EJB session : Nous remarquons que nous écrivons beaucoup de code pour créer les EJB, définir ses attributs et générer ses méthodes.

Ces modules seront définit dans le chapitre suivant.

3/Le module web :

Le module web est construit de pages JSF, pour chaque EJB nous définissons les pages JSF suivantes :

- Page de création.
- Page de consultation.
- Page de suppression.
- Page de modification.
- page index.xhtml.

Pour certain EJB nous définissons d'autres pages JSF « Page de calcul d'évaluation globale …… »

3-9 Le modèle relationnel :

Il est possible de traduire un diagramme de classe en modèle relationnel. Bien entendu, les méthodes des classes ne sont pas traduites. Aujourd'hui, lors de la conception de base de données, il devient de plus en plus courant d'utiliser la modélisation UML plutôt que le traditionnel modèle entités-associations.

Cependant, à moins d'avoir respecté une méthodologie adaptée, la correspondance entre le modèle objet et le modèle relationnel n'est pas une tâche facile. En effet, elle ne peut que rarement être complète puisque l'expressivité d'un diagramme de classes est bien plus grande que celle d'un schéma relationnel.

Classe avec attributs

Chaque classe devient une relation. Les attributs de la classe deviennent des attributs de la relation. Si la classe possède un identifiant, il devient la clé primaire de la relation, sinon, il faut ajouter une clé primaire arbitraire.

Association 1 vers 1

Pour représenter une association 1 vers 1 entre deux relations, la clé primaire de l'une des relations doit figurer comme clé étrangère dans l'autre relation.

Association 1 vers plusieurs

Pour représenter une association 1 vers plusieurs, on procède comme pour une association 1 vers 1, excepté que c'est forcément la relation du côté plusieurs qui reçoit comme clé étrangère la clé primaire de la relation du côté 1.

Association plusieurs vers plusieurs

Pour représenter une association du type plusieurs vers plusieurs, il faut introduire une nouvelle relation dont les attributs sont les clés primaires des relations en association et dont la clé primaire est la concaténation de ces deux attributs.

Classe-association plusieurs vers plusieurs

Le cas est proche de celui d'une association plusieurs vers plusieurs, les attributs de la classe-association étant ajoutés à la troisième relation qui représente, cette fois ci, la classe-association elle-même.

Le Passage du diagramme de classe UML au modèle relationnel pour SGBD :

Affectation (**DateAffectation, Matricule*, CodePoste***)

Conge (**NumConge** , DateDebutConge, DateFinConge, DateReprise, **Matricule***)

Deces (**NumDeces**, LibelleDeces, LieuDeces, NumCertificatDeces, DateDeces, **Matricule***)

Demission (**NumDemission**,MotifDemission, DateDemission, **Matricule***)

Diplome (**CodeDiplome**, LibelleDiplome, DateDiplome, **LibelleNiveau***)

Employe (**Matricule**, NomE, PrenomE, DateNaissE, LieuNaissE, SituationFamiliale, NbrEnfant, ServiceNational, SalaireBase, Sexe, Adresse, Nationnalite, Grade, NomJeuneFille, DateRecrutement, **CodeDiplome***)

Evaluation (**NumEvaluation**, Note, Anne, NumReclassement, NumFormation, **Matricule***)

Formation (**NumFormation**, LibelleFormation, DureeFormation, Cout, **LibelleOrganisme***)

Inscription (**Etat, IdSession*, Matricule***)

Licenciement (**NumLicenciement**, MotifLicenciement, DateLicenciement)

MiseApied (**NumMiseApied**, NbrJourMisePied, DateDebutMisePied, DateFinMisePied)

MisEnDisponibilite (**NumDisponibilite**, MotifDisponibilite, DureeDisponibilite, DateDisponibilite, **Matricule***)

NiveauQualification (**LibelleNiveau**)

Organisme (**LibelleOrganisme**)

PlanFormation (**AnneeFormation**)

Poste (**CodePoste,** LibellePoste, **CodeService***)

Reclassement (**NumReclassement**, LibelleReclassement, AnneeReclassement, Categorie, Classement)

Retraite (**NumRetraite**, LibelleRetraite, DateRetraite, **Matricule***)

Sanction (**NumSanction**, MotifSanction, ObservationSantion, DateSanction, **NumLicenciement*, NumMiseApied*, NumReclassement***)

Service (**CodeService**)

Session (**IdSession**, DateDebut, DateFin, AnneeFormation, **NumFormation***)

Conclusion :

Aprés l'étude du sujet, l'élaboration du diagramme de classe et du schéma relationnel nous passons à la réalisation de notre application.

Le choix des outils de développement, des composants techniques, graphique et les interfaces des différentes fonctionnalités de l'outil serons développé en détail dans le chapitre suivant.

CHAPITRE 4: IMPLEMENTATION

4-1 Introduction :

Dans cette partie nous allons présenter l'ensemble des outils et technologies utilisées dans la réalisation de ce projet, tel que le langage de programmation, le SGBD, les serveurs…..etc. Nous étalerons ensuite les différentes fonctionnalités offertes par notre système.

4-2 Mise en œuvre de l'application :

Notre application permet de gérer une partie du système de gestion division ressources humaine « Gestion de carrière » à partir des composants EJB.

Notre application est une application web développée en utilisant les JSF. Pour la réalisation de notre objectif nous avons utilisé plusieurs environnements de développement qui sont cité dans ce présent chapitre.

Dans la suite de cette partie nous allons présenter des captures d'écran de l'application.

4-3 Les étapes de l'implémentation de notre application :

Avant de citer les étapes que nous avons suivies pour la création de notre projet nous allons résumer notre projet par un schéma global qui montre le fonctionnement de notre application ainsi que les différents liens entre les différents modules.

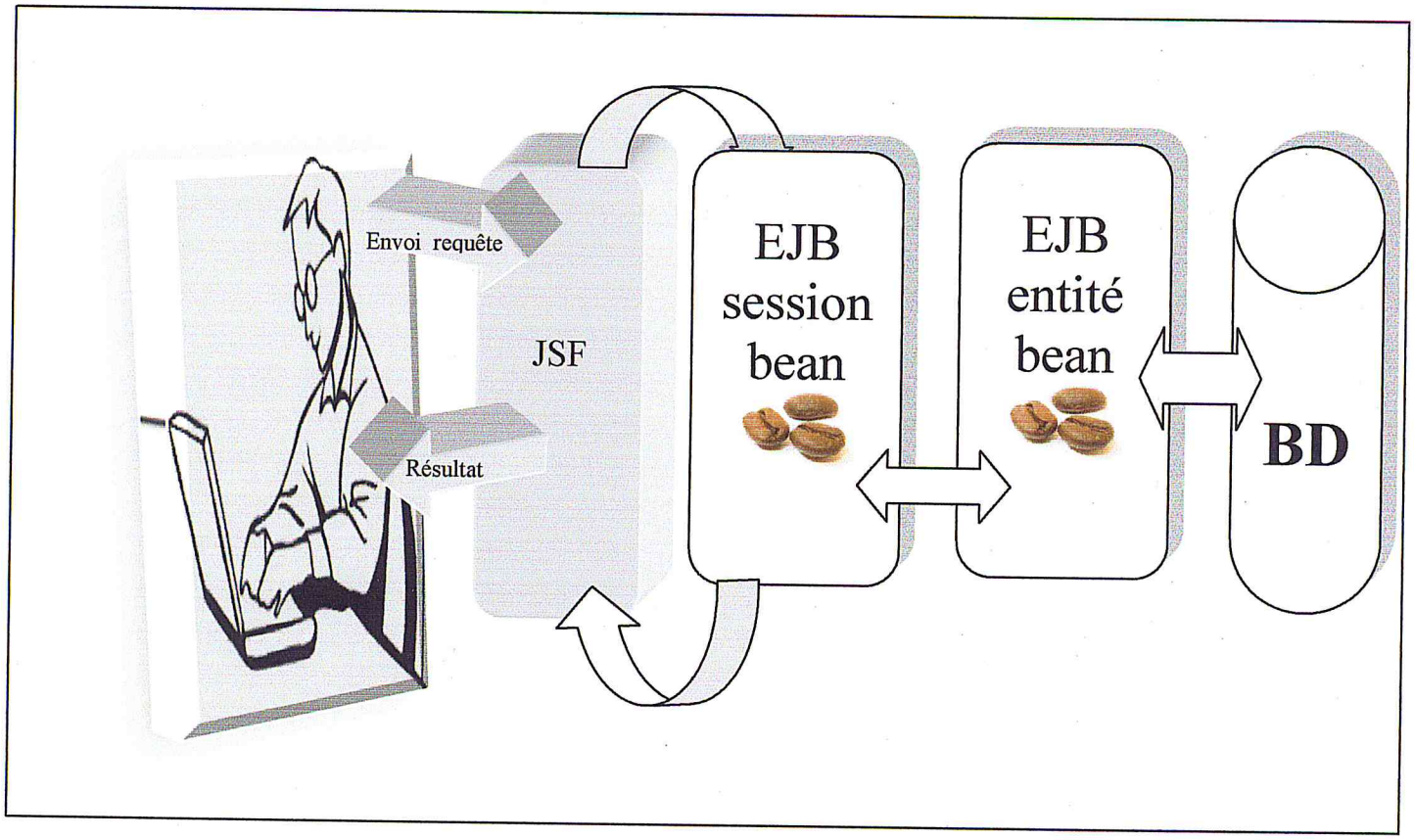

Figure 22 : Schéma globale de notre application

Pendant la création de notre application nous avons eu beaucoup de difficulté car c'est une nouvelle technique pour nous, on a essayé plusieurs méthodes d'après les recherches et la documentation [22][23][24][25][26][27] on est arrivé à cette méthode qui nous a permet d'atteindre notre but et nous avons suivi les étapes suivantes :

1/ Nous avons crée la base de donnée, le pool et la source de données,

Pur cela nous avons suivi les étapes suivantes :

A. La base de données :
   - Lancer MySQL consol
   - Taper le password = Amin
   - Créer la base de données.
   - Valider et refermer la console

Voici la fenêtre de MySQL :« les étapes de création de la base de données sur MySQL »

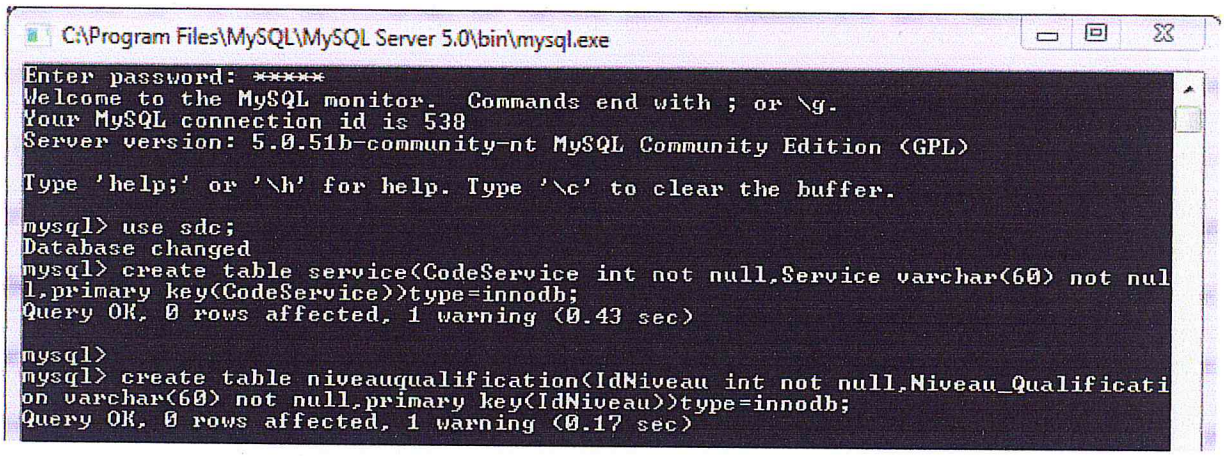

Figure 23 : La création de BD sur MYSQL

A. La ressource de données :

- Dans cette étape nous avons crée JDBC connections pool, choisir database connexion et on a ajouté les propriétés de la connexion pool :
  Data source class name: com. mysql.jdbc2.optimal.mysqlDataSource
  Type de ressource : javax.sql.connectionPolDataSource
  Description : connexion de la base de données.

B. Le pool :

- Lancer le serveur d'application " glassfish ".
- Console d'administrateur : se connecter avec le mot de passe

- Choisir ressource JDBC, connexion pool, sélectionner notre base de donnée
- Lancer la commande de Ping.

La commande Ping réussit, alors notre serveur Glasfish possède une connexion active à notre base de données sur le serveur MySQL.

D. Nous avons crée la référence à des ressources de données :

On sélectionne le Web.xml, on développe les ressources références rubrique ,

On ajoute une ressource de référence puis on saisi les informations suivantes :

Nom de la ressource : jdbc/BD

Type de ressource : javax.sql.DataSource

Authentification : container

Partage Portée : S areable

Description : Connecter la base d donnée avec l'application.

2/ Dans cette étape nous avons créé une application web en utilisant JAVA EE6, GlassFih 3.1.2 comme serveurs et JSF2.

3/ Nous avons créé les EJB persistant : Pour la création des entités bean on a commencé par la création du package entité en suite la création d'une classe java persistance API entité pour chaque table de la base de données sélectionnées,

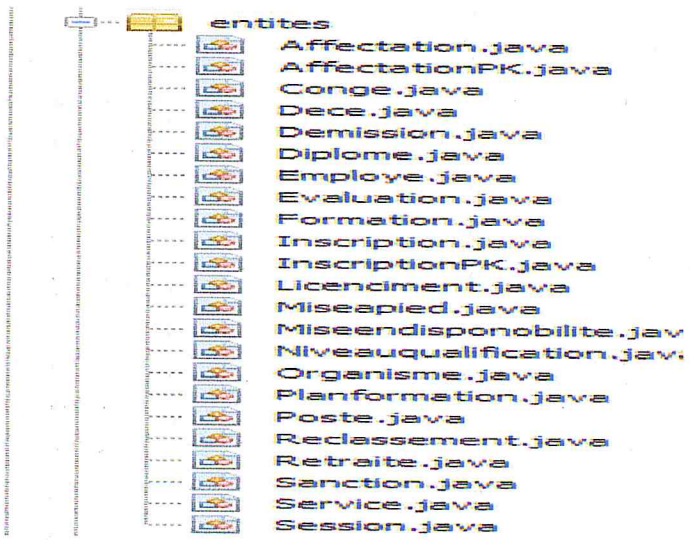

Figure 24 : Les EJB persistant " Entités bean "

Nous avons crée aussi l'unité de persistance qui se réfère à un ensemble de classes d'entités qui existent dans notre application, cela va générer n persistance.xml ce fichier va être utilisé pour spécifier les paramètres de l'unité de persistance. Dans notre cas 'Eclipse Link(JPA 2.0)'

Est la sélection par défaut pour le serveur associé à notre projet.

L'unité de persistance est une collection de classes d'entités qui existent dans notre application, elle est définit par persistance.xml fichier de configuration, qui est lu par le fournisseur de persistance.

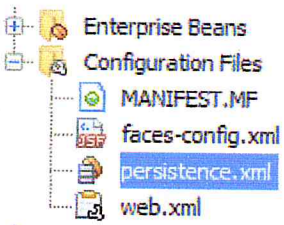

Figure 25 : L'unité de persistance

Nous avons générer en suite le code de l'entité et on a compléter avec des annotations nommées d'une requête, les champs représentant les colonnes, et les relations qui représentent les clés étrangères.

Nous avons écrit le contenu des requêtes, chaque EJB entité définie au moins la requête FindAll ().

Nous avons refait ces étapes pour les 21 EJB.

4/ Nous avons crée les EJB session :

Les EJB session sont invoquées par un client en vue d'effectuer une opération spécifique.

On a crée une façade EJB session pour chaque classe d'entité avec des méthodes d'accès à la base de donnée pour sa classe d'entité respective.

Une façade de session est un modèle de conception qui tente de résoudre les problèmes communs qui se posent dans un environnement d'application à plusieurs niveaux, tel que :

- Couplage serré, ce qui conduit à diriger la dépendance entre les clients et les objets métier.
- Invocation de trop de méthodes entre le client et le serveur, conduisant à des problèmes de performance du réseau.
- L'absence d'une stratégie d'accès client uniforme, exposant des objets métier pour abuser.

Une façade session abstraction des interactions sous-jacentes objets métier et fournit une couche de service, qui n'expose que la fonctionnalité requise. Ainsi il cache à la vue du client

Les interactions complexes entre les participants. Le Bean session (représente la façade session) gère également le cycle de vie de ces participants en créant, localiser, modifier et supprimer au besoin.

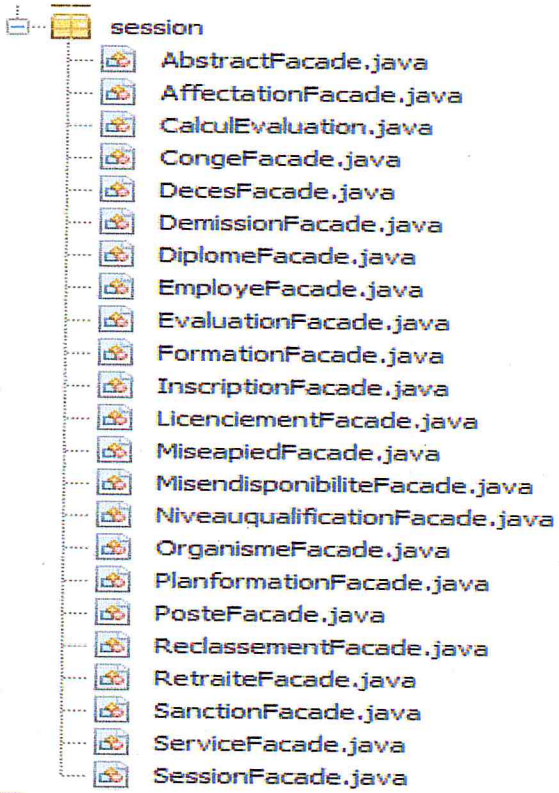

Figure 26 : Les EJB session " Session bean "

Figure 27 : Les méthodes de session bean

5/ Dans cette étape nous avons crée les pages JSF, comme on a déjà cité dans la partie conception, nous avons créé pour chaque EJB (21 EJB) les pages JSF de création, modification, consultation et suppression. Alors on aura (21*4) pages JSF.

Nous avons crée d'autre pages JSF pour les EJB qui ont besoin d'autre méthodes comme :

Calcul d'évaluation globale, Test de proposabilité……

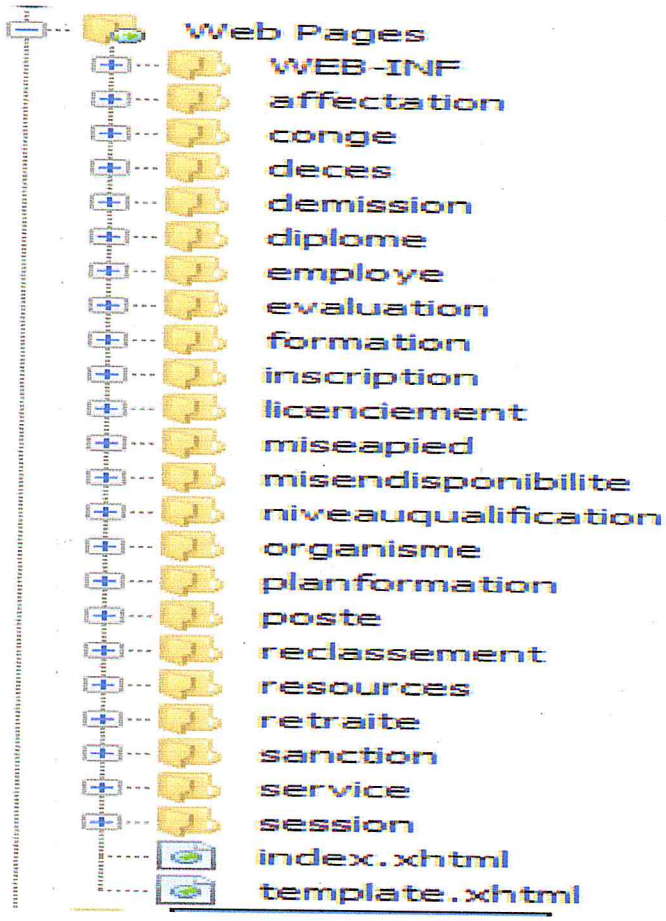

Figure 28 : les pages JSF

6/ Après toute ces étapes, nous avons vérifié que le serveur d'application GlassFish ainsi que le serveur de BD MySQL sont lancés, puis déployer le projet et l'exécuter.

1- Sélectionner créer nouvelle évaluation.

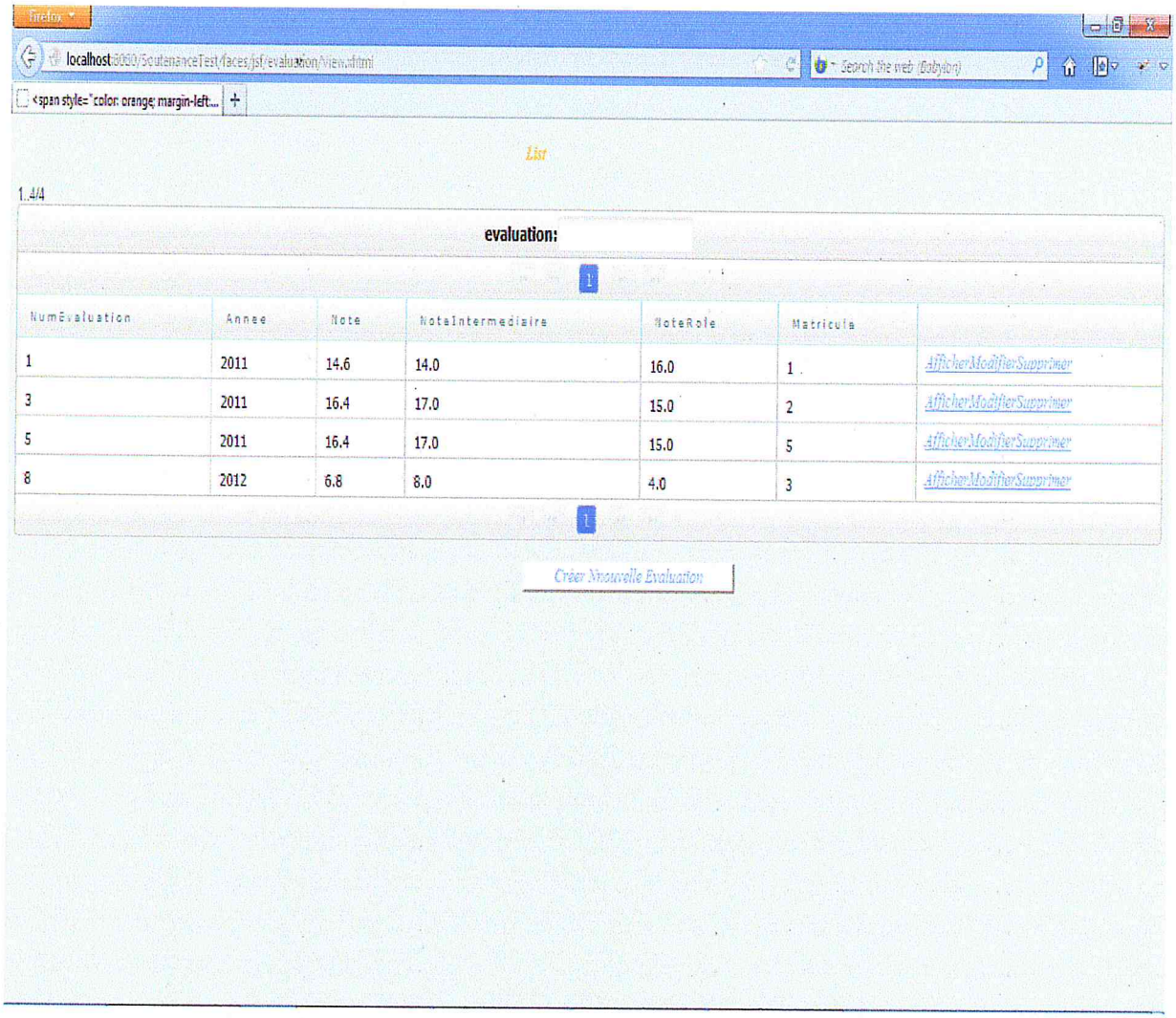

Figure 30 : liste des évaluations

3- Remplir le formulaire de l'évaluation.

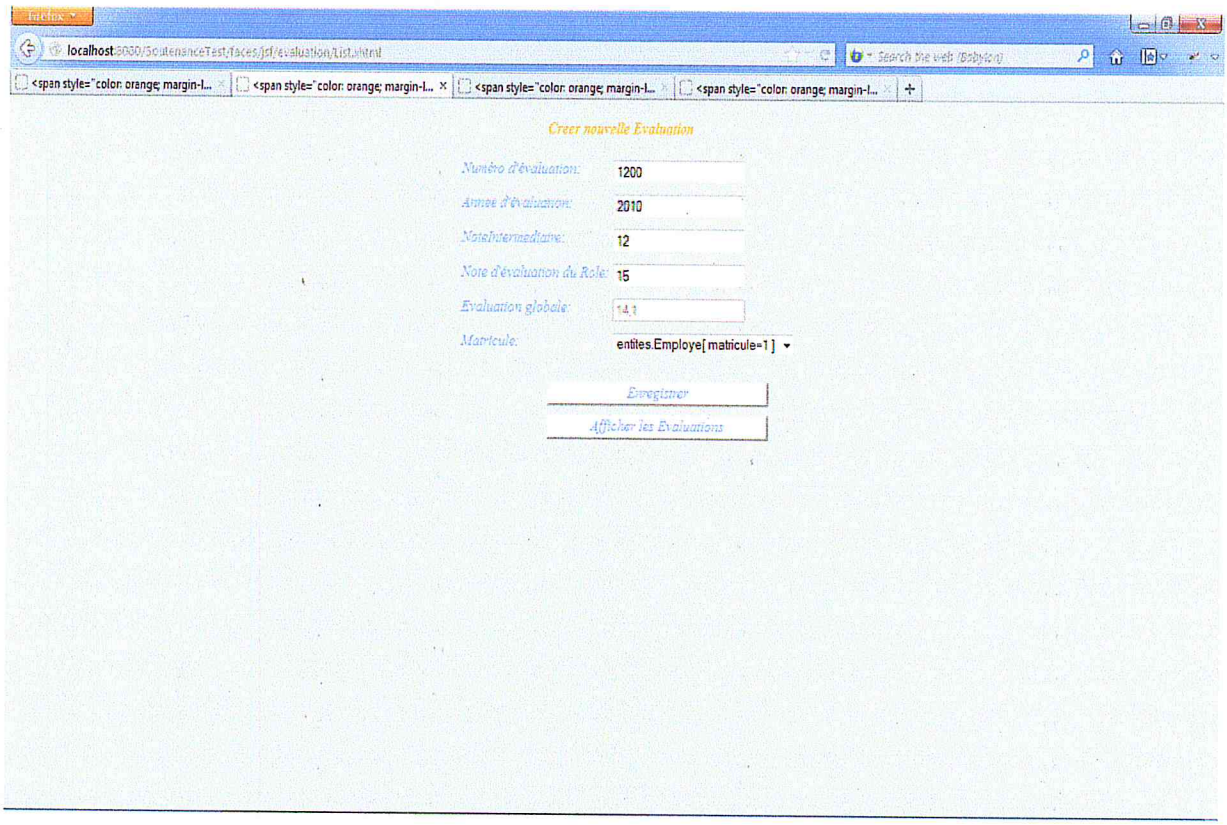

Figure 31 : formulaire de création d'une nouvelle évaluation

Cliquer sur Enregistrer, à cette étape la note d'évaluation globale va être calculée

Si la note d'évaluation globale est supérieure ou égale à 13,5 alors on aura cette fenêtre :

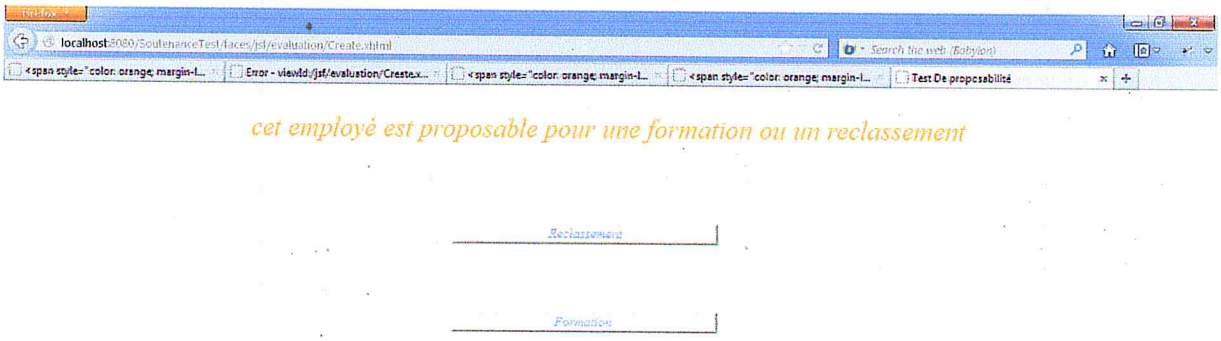

Figure 32 : Test de proposabilité

Le but de cette application est de développer des composants EJB réutilisables, Puisque les règles de gestion du DRH changent au fil du temps alors :

Supposons qu'il y aura des changements sur la méthode de calcule de la note d'évaluation globale.

Nous allons montrés la notion de réutilisabilité dans les interfaces suivante :

Nous avons :

Note intermédiaire = 17, la note d'évaluation de rôle = 15 après les calculs effectués par le composant de calcul de note d'évaluation globale on a obtenu 16.4 comme note d'évaluation globale.

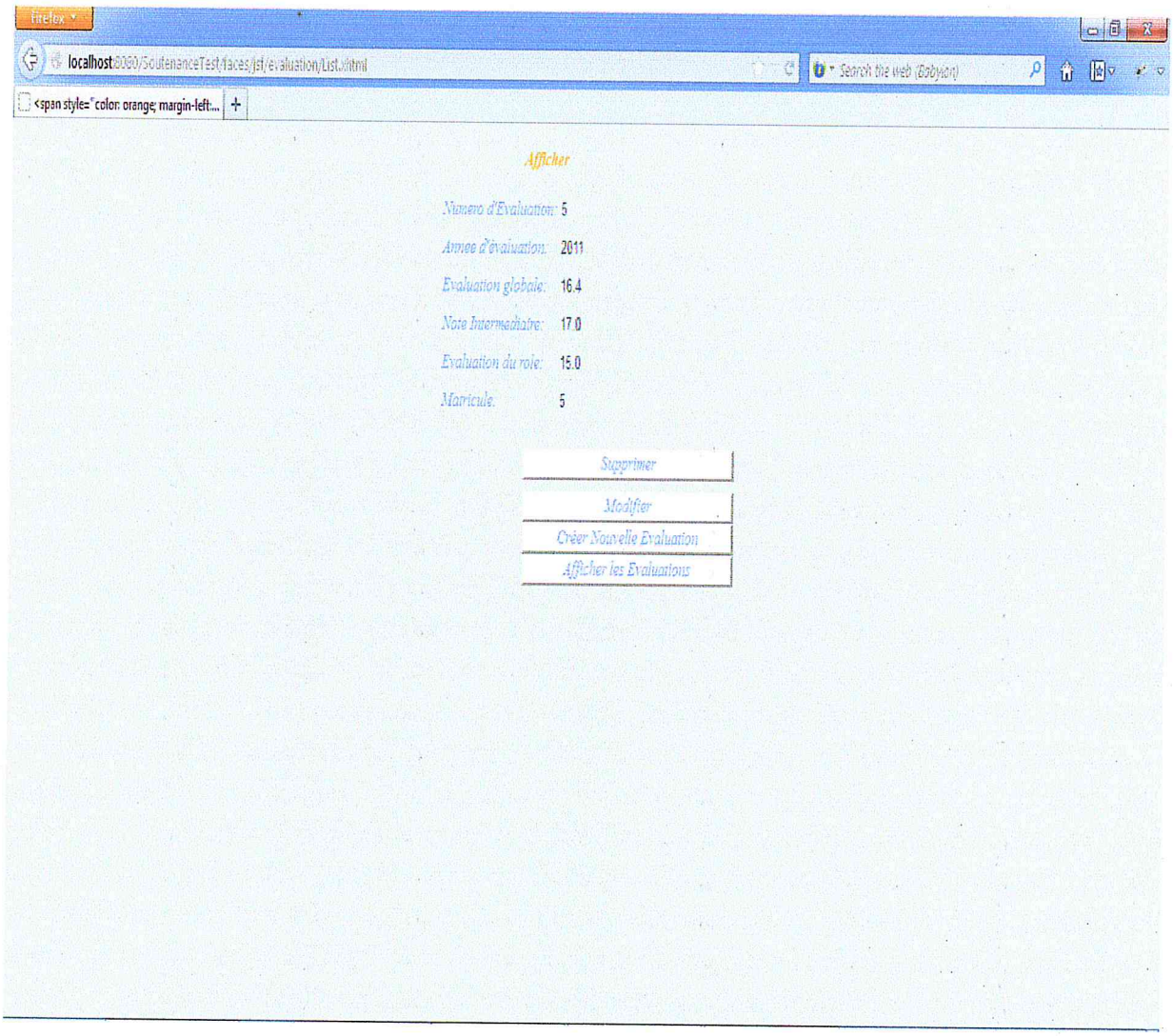

Figure 33: test d'EJB de calcul 1

Nous avons remplacé l'EJB qui fait le calcul de la note d'évaluation globale par un autre EJB fait aussi le calcul de la note d'évaluation globale avec une règle de calcul différente qui va nous donner un résultat différent

Maintenant avec les mêmes paramètres précédant, la note d'évaluation globale = 15.4

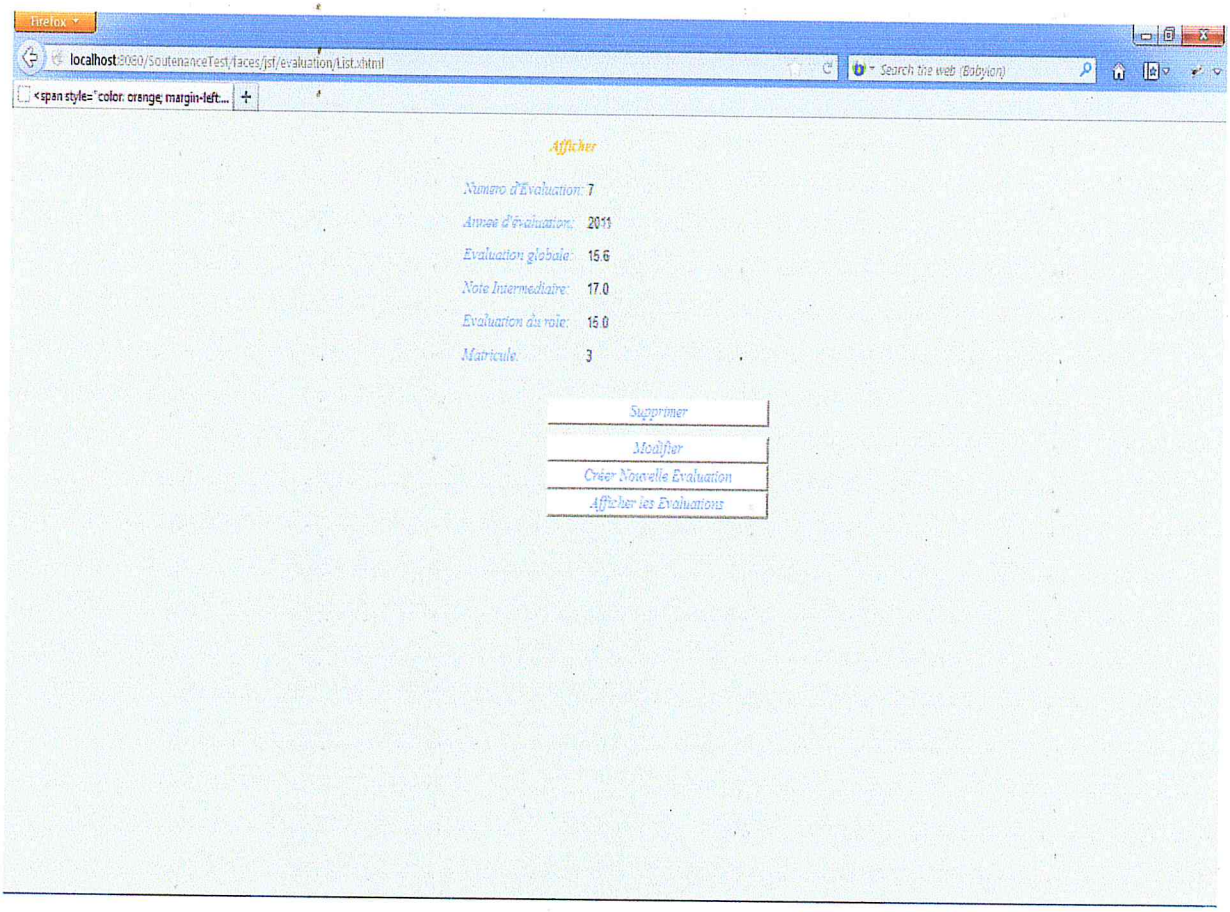

Figure 34: test d'EJB de calcul 2

Environnement de développement :

Nous présentons dans cette section les outils que nous avons utilisés pour réaliser notre travail (conteneurs web, SGBD, serveurs d'application, langages de programmation) tout en justifiant nos choix.

Langages de programmation :

Apparu en 1991, le langage JAVA a commencé à être intéressant à partir de 1995. Ce langage ne cesse de se développer. Il s'agit d'un langage orienté objet dont la syntaxe est très proche de celle du C+ +.

C'est également un langage portable, c'est-à-dire qu'il s'adapte à une foule de plates-formes différentes. C'est là l'une des qualités de JAVA [14].

Les environnements de développement intégré (IDE) :

Netbeans est un environnement de développement open source écrit en Java. Le produit est composé d'une partie centrale à laquelle il est possible d'ajouter des modules.

Netbeans est un IDE open source initialement racheté et développé par Sun Microsystems [12].

JSF (Java Server Faces) :

JSF est un Framework d'interface utilisateur combinant les technologies des JSP et les Servlets. Créé en 2003 par Craig McClanahan, il s'agit d'un framework structurant les couches de présentation de l'information comme la navigation d'une page à une autre, la présentation des données, la gestion des événements [15].

Les serveurs d'application :

Le projet GlassFish est un projet communautaire dont le but est de développer un serveur d'application open source qui implémente les spécifications de Java EE à partir de la version 5. Il est l'implémentation de référence de ces spécifications [13].

Les outils concernant les bases se données :

Les SGBD libres et gratuits sont nombreux. MYSQL, SQL, Postgres sont des exemples. Si nous avons choisit MYSQL, c'est plus pour des raisons de performances et fonctionnalités offertes. Nous citerons dans la suite ses principaux avantages :

- MYSQL est beaucoup moins complexes à installer et à administrer que d'autres systèmes.
- MYSQL permet des connexions multiples en même temps et utiliser différentes bases de données simultanément.
- MYSQL dispose d'un système de contrôle intégré qui interdit la consultation de données à ceux qui n'en ont pas l'autorisation [16].

Conclusion :

Dans ce chapitre, nous avons présenté l'implémentation de notre système, telle que nous avons définie et justifié les choix des outils utilisés pour la réalisation de notre application. En suite, nous avons présenté quelques interfaces de notre application.

Conclusion générale

L'élaboration de notre travail de fin d'étude est faite grâce aux connaissances théoriques acquises durant notre cursus. Elle nous a permis de concevoir un système de gestion DRH sécurisé à partir des composants EJB pour la société de distribution centre SONELGAZ SDC.

Et cela suite à une étude de la situation actuelle dans le service gestion ressource humaine. Après une collecte maximale des informations, nous avons pu détecter les problèmes, à qui nous somme sensés d'apporter une solution adéquate.

Ce projet nous a permis d'améliorer nos connaissances en Architecture logicielle ainsi que le développement d'une application web avec java, il nous a permit aussi d'acquérir une certaine expérience dans le domaine des composants logiciel et spécialement les composant EJB « Entreprise Java Beans »

Enfin nous espérons que l'étude menée a rependue aux besoins des utilisateurs et auront un apport bénéfique pour la mise en place des moyens nécessaire et efficace pour une meilleure gestion des ressources humaines.

BIBLIOGRAPHIE

[1] Olivier Barais « Construire et maitriser l'évolution d'un architecture logicielle à base de composants » Thèse de doctorat, Novembre 2005.

[2] Pascal ANDRE, Dan Chiorean, Franstisek Plasil, Jean-Claude Royer « Abstraction de comportement des composants logiciels à partir de code » PDF, 16 Octobre2006.

[3] Karim Dahmen « Conception d'un générateur d'intergiciels temps réel embaquée dans l'automobile » Mémoire d'ingénieur en informatique, 2007.

[4] Nguyen Trong Khanh « Réutilisabilité des composants web » Rapport de stage de fin d'étude, Octobre 2008.

[5] www.jmdoudoux.fr/java/dej/chap-ejb.htm

[6] Isabelle Valembois, lois Millecam « Java : Autoformation », livre, 2001.

[7] Chafik Merkak « La décision répartie pour le déploiement distribué » Mémoire de master recherche en informatique.

[8] Michel Riveill, Philippe Merle « Programmation par composant ».

[9] http://www-igm.univ-mlv.fr/~dr/XPOSE2002/CCM/corba.htm

[10] Sébastien Leriche « Architecture à coposans et agents pour la conception d'application réparties adabtables » Thèse de doctorat, 2006.

[11] Laurent audibert ,livre : « cours UML ».

[12] www.netbeans.org

[13] www.jmdoudoux.fr/java/dej/chap-outils.htm

[14] www.vulgarisation-informatique.com/introduction-java.php

[15] www.journaldunet.com/encyclopedie/definition/1117/53/22/javaserver-faces.shtml

[16] Antoine Cornuéjols, « Bases de données concepts et programmation » PDF, 19Octobre 2009

[17] Salah Hajab « Assemblage de composant logiciels ». Mémoire d'étude approfondie dans l'université de technologie de Troyes, 2003.

[18] Hind Lilia Bouziane « L'abstraction des modèles de composants logiciels pour la programmation d'applications scientifiques distribuées ». Thèse de doctorat présentée devant l'université de rennes 1 par, 2008.

[19] Zougagh Mohamed « », mémoire d'ingénieur d'état , université Saad dahlab Blida, 2004.

[20] Sandjakeddine Nassima « Conception et implémentation d'un modèle de sécurité pour la pose d'interaction entre composants spécifiques aux SI» , mémoire d'ingénieur d'état, université Saad Dahlab, Blida ,2004.

[21] Nathanael Cottin « Java Entreprise Edition », PDF, 2009.

[22] Alexandre Baillif, philippe lacomeet Raksmey phan « création d'une application web/jsf» pdf, juillet 2012.

[23] SachaKrako wiak « composants logiciels » pdf, 2006.

[24] Luxlog.wordpress.com/2007/12/15/basic-example-web-app-with-glassfish-netbeans-jsf-and-ejb/

[25] http://netbeans.org/kb/docs/javaee/ecommerce/intro.html

[26]http://miageprojet2.unice.fr/Intranet_de_Michel_Buffa/Cours_composants_distribu%C3%A9s_pour_l'entreprise_%2f%2f_EJB_2009/TP1_2011_EJB_3.1%2f%2fJPA%2f%2fJSF2

[27]http://miageprojet2.unice.fr/Master_MIAGE_1/Applications_Web/TP7_Applications_web_%3A_utilisation_d'une_base_de_donn%C3%A9es