REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET
DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU
FACULTE DE GENIE ELECTRIQUE ET D'INFORMATIQUE DEPARTEMENT
INFORMATIQUE

MEMOIRE

De fin d'étude

En vue de l'obtention du diplôme de Master en informatique
Option : Conduite de Projets Informatiques.

Thème

Etude comparative des frameworks et réalisation d'une application J2EE pour l'entreprise d'assurance « Direct Assurance » (D.A)

Proposé et dirigé par :

Mr: A.Dib

Réalisé par :

M^elle : ABROUS Lamia.

M^elle : BOUDJEMA Celia

2014

Remerciements

Nous tenons à exprimer nos vifs remerciements à tous ceux qui nous ont aidés à aboutir dans notre travail et à tous ceux qui ont contribué d'une façon ou d'une autre à réaliser ce projet de fin d'études, en premier lieu notre promoteur M. Dib.

Nos plus vifs remerciements vont également aux membres du jury pour nous avoir fait l'honneur de juger ce travail.

Dédicaces

A ma famille, à mes amis et à tous ceux qui m'ont soutenu de prés ou de loin dans la réalisation de ce modeste travail.

Lamia

A ma famille, à mes amis et à tous ceux qui m'ont soutenu de prés ou de loin dans la réalisation de ce modeste travail.

Celia

Sommaire

Liste de figures

Liste des tableaux :

Introduction générale :

Introduction générale

De nos jours le nombre d'entreprises ne cesse d'accroitre et plusieurs d'entre elles usent tous les moyens possibles pour ce concurrencer, le système d'information qui est au cœur de l'activité de l'entreprise, exige une mise à jour continue, pour le mettre au diapason des défis de cette modernisation.

Désormais, l'informatique n'est plus seulement l'un des instruments de productivité. Elle devient un outil de gestion et de pilotage de l'entreprise, voire un instrument stratégique apportant les moyens d'évolution des métiers de l'entreprise.

On ne choisi donc plus les applications logicielles pour leur aptitude à remplir une fonction informatique précise, mais comme élément d'une chaîne de production de valeur : l'entreprise.

Une bonne partie de l'activité d'une entreprise consiste à communiquer, enregistrer, rechercher de l'information, effectuer des calculs, créer des documents...l'informatique, grâce aux logiciels, a investi tous les domaines d'activités de l'entreprise.

De plus, l'informatique permet de faire circuler l'information dans toute l'entreprise, grâce aux boites mails, au site de l'entreprise ou même comme nous l'avons expliqué juste avant, elle permet d'accroître la production et de rendre l'entreprise plus connue.

Les compagnies d'assurances font également partie de cette concurrence et l'évolution sans cesse croissante du marché des assurances et la pénétration bientôt des compagnies sur le marché mondial, imposent une attention particulière sur les outils techniques et technologiques utilisés pour la modernisation et le développement des fonctions de l'entreprise d'assurance.

Notre projet consiste à réaliser une application web permettant à « Direct Assurance » qui est une entreprise d'assurance française de bien gérer la liste de ses clients ainsi que leurs contrats appropriés et surtout de contrôler par l'administrateur, les privilèges et les droits d'accès des agents s'occupant de réaliser toutes ces tâches.

Le travail est donc réparti sur quatre chapitres dont deux traitent le coté théorique et les deux autres, le coté conception et réalisation de notre projet.

Le premier chapitre est consacré aux assurances dans le quel nous traiterons les notions de bases.

Le second chapitre met l'accent sur le web, le modèle client/serveur, le model MVC et principalement sur les Frameworks.

Le troisième chapitre est consacré à la réalisation et la conception de notre application.

Enfin, dans le quatrième chapitre nos présentons notre application. Nous définissons les outils nécessaires au développement de l'application, voir son fonctionnement ainsi que certaines de ses interfaces.

Chapitre I:

Généralité sur les assurances

I- Introduction :

L'existence humaine est pleine de risques, la personne de chacun est à la merci d'événements imprévus ; les maladies, les accidents corporels, les accidents de circulation entrainant de manière inopinée des invalidités, des incapacités de travail, des décès prématurés et les préjudices matériels et moraux qui en résultent pour la victime et ses proches.

D'autres événements inattendus frappent l'homme dans ses biens ; incendies, accidents, provoquant des dégâts matériels et des pertes de revenu.

Pour celui qui en est victime, la réalisation d'un risque imprévu peut être une catastrophe. De nombreux procédés ont été mis au point en vue soit de réduire les chances de survenances du sinistre, soit d'en atténuer les effets.

Et l'assurance apparait ainsi comme l'un des nombreux procédés par les quels l'homme se prémunit contre les risques qui le menacent. Elle occupe une place privilégiée parmi les procédés, car elle peut couvrir les risques extrêmement variés et elle est susceptible d'apporter une protection complète en cas du sinistre.

Pour cela nous avons donc consacré ce premier chapitre pour parler des assurances à travers le monde depuis leur apparitions dans les temps anciens jusqu'à leur présente actualité dans nos jours et nous avons cité également les différentes catégories existantes de ces assurances ainsi que les entreprises s'occupant de leur gestion, leur structure, leur fonctionnement ainsi que leurs rôles.

II- Historique：[1]

L'origine du contrat d'assurances remonte aux temps anciens probablement avant même l'avènement de Jésus-Christ. D'origine romaine, le premier type de contrat connu était une assurance au dernier survivant. De façon plus explicite, il s'agissait d'un contrat moral permettant d'aider financièrement la dernière personne survivante d'un groupe.

Ainsi, un groupe de personnes se connaissant mettaient de côté à des dates convenues une valeur. La dernière personne survivante du groupe bénéficiait de l'ensemble de ces valeurs mises de coté. Ce genre de contrat subsiste de nos jours mais sous une autre forme et porte le nom de Tontine.

Ensuite survint l'époque où les financiers garantissaient les marchandises transportées d'un port à l'autre. Quant à la première compagnie d'assurance décès (ou vie), elle semble italienne et date des années 1500. L'assurance incendie intervint plus tard. Elle remonte à l'incendie de Londres en 1666 où plus de treize mille (13000) bâtiments sont détruits. L'année suivante naquit la première compagnie d'assurance sur l'incendie. Cependant, il faut attendre le 19e siècle pour assister à l'éclosion réelle des compagnies d'assurances. Dans ce registre et sous

l'impulsion des nouvelles technologies, on cite au 20e siècle les contrats d'assurances automobiles, avion et de protection sociale.

Vers la fin de l'année 1997, les opérateurs économiques du Burkina Faso donnent naissance à la première société d'assurances IARD à capitaux privés : la Générale des Assurances. Elle démarre effectivement ses activités le 1er janvier 1998 avec un capital initial de quatre cents millions (400 000 000) de FCFA et un effectif de vingt personnes. Ce capital est porté à la somme d'un milliard (1 000 000 000) de F CFA en 2005 intégralement libéré.

Sur cette même lancée, elle procède en 2006 à la création d'une société d'assurance-vie dénommée GA-VIE avec un capital de cinq cents millions (500 000 000) de FCFA qui passe aussi à la somme d'un milliard (1 000 000 000) de F CFA depuis avril 2010. Le groupe Générale des Assurances s'impose ainsi sur le marché des assurances comme une compagnie.

III- Définition des assurances : [2]

L'Assurance est, par définition, un système qui permet de prémunir un individu, une association ou une entreprise contre les conséquences financières et économiques liées à la survenance d'un risque (événement aléatoire) particulier.

Le moyen mis en œuvre par les organismes d'assurance pour les prémunir contre ce risque est de les associer à une communauté de personnes (les assurés), qui cotise pour être en mesure d'indemniser ceux parmi ses membres qui subiraient des dommages matériels ou corporels en cas de réalisation du risque.

Ainsi, dans la mesure où c'est l'ensemble de la communauté des assurés qui prend matériellement en charge les dommages subis par ses membres frappés par la réalisation du risque, l'assurance est un système de gestion des risques basé sur la notion de solidarité.

IV- Les deux grandes catégories d'assurances : [3] [2]

Il existe deux grandes catégories d'assurances : celles qui couvrent une personne physique et celles qui couvrent les biens. Mais, il est également possible de souscrire plusieurs assurances dans un même contrat. On parle alors de « multirisques ».

1- L'assurance de personnes :

Une assurance de personnes a pour objet de couvrir les risques relatifs aux individus comme les accidents corporels, la maladie, le décès ou encore l'invalidité.

On distingue la prévoyance (garantie emprunteur, indemnités journalières, rente éducation…) et la santé laquelle est subdivisée en deux catégories bien distinctes : la garantie obligatoire (Sécurité sociale) et la garantie complémentaire (mutuelle, assureurs…).

L’assurance de personnes peut être souscrite soit à titre individuel soit à titre collectif. Certains contrats permettent la constitution et le versement d’une épargne sous forme de capital ou de rente. C’est notamment le cas d’une assurance vie.

2- L’assurance des dommages :

L’assurance des dommages permet d’obtenir une indemnisation en cas de sinistre. Elle regroupe à la fois la protection de responsabilité (responsabilité civile, responsabilité civile familiale ou responsabilité professionnelle) et celle de biens (dommages causés au véhicule, protection des biens meubles ou immeubles).

Par exemple, en cas d’accident de la route, elle garantit entre autres l’indemnisation des dommages subis par la voiture et s’avère donc nécessaire même si, dans la plupart des cas, elle n’est pas obligatoire. C’est notamment le cas de la prévoyance.

On distingue deux niveaux de garanties dommages : la garantie dommages collisions (permettant à un assuré de bénéficier d’une indemnisation en cas d’accident responsable avec la présence d’un tiers identifiable) et la garantie dommages tous accidents (permettant à un assuré de bénéficier d’une indemnisation en cas d’accident responsable même en l’absence de tiers).
En France par exemple, les assurances sont séparées en deux parties. Il y a les assurances vie et les assurances Non vie ou IARD (Incendie, Accidents, Risques Divers). Cependant, ni le code des assurances ni les grands professionnels de l'assurance telle que la FFSA n'arrivent à clairement définir ces deux termes.

Cette distinction entre ces deux types d’assurances repose sur la différence du mode de gestion des primes.

En effet, de manière générale, les Assurances Non Vie gèrent les primes par répartition (= mode de gestion collectif où les primes de la communauté des assurés servent à payer les sinistres de la communauté des assurés au titre du même exercice), tandis que les Assurances Vie les gèrent par capitalisation (= mode de gestion individuel où les primes de l’assuré servent à lui délivrer une prestation au moment de la survenance du risque).

A côté de cette distinction Assurances "Non Vie" et Assurances "Vie", on trouve une autre distinction entre :

- les Assurances IARD (Incendie, Accidents, Risques Divers) : elles regroupent les Assurances de Biens et les Assurances de Responsabilité.
- les Assurances de Personnes : elles regroupent les Assurances Santé et les Assurances "Vie".

En France, les assurances proposées se divisent en plusieurs branches. La plupart de ces branches peuvent se classer dans les deux catégories précédemment citées, mais encore une fois c'est assez flou. Donc voici à quoi ressemblent les offres fournies et dans quelles catégories elles peuvent être placées :

Assurance vie :

L'assurance vie peut comprendre plusieurs offres comme les assurances de risques (c'est à dire décès), les bons de capitalisation et toutes les autres offres d'épargnes.

Assurance Non-vie ou IARD :

Assurance de biens et de responsabilités, assurance maladie, assurance accidents corporels, assurance financière (caution, garantie chômage, etc), assurance juridique (protection juridique) et les offres d'assistance pour toutes sortes de préjudices.

Voici quelques exemples :

L'assurance auto:

L'assurance automobile est obligatoire depuis l'année 1958 et elle est gérée par le code des assurances. En France, l'assurance automobile est une assurance pour tout véhicule motorisé en circulation sur le territoire français ou dans les secteurs avec une obligation d'assurances...

L'assurance moto:

Une assurance moto est quasiment similaire à une assurance automobile. On retrouve à peu près les mêmes garanties et il y'a la formule au tiers ou la formule tous risques. Lorsqu'il y a un accident de moto, il y a dans la majorité des cas des dégâts corporels, c'est pour cela qu'il est extrêmement conseillé de souscrire à une garantie dommages corporels pour le conducteur.

L'assurance habitation:

L'assurance habitation est faite pour les particuliers, elle leur permet d'assurer leurs demeures et leurs dépendances. Cette assurance couvre la demeure, les objets qui s'y trouvent ainsi que les habitants de l'habitation (responsabilité civile). Il existe deux types de contrats d'assurances habitations...

L'assurance vie:

Comme son nom l'indique, l'assurance vie est un type d'assurance. Le but premier d'une assurance est d'assurer un capital ou une rente si jamais il arrive quelque chose à la personne (décès ou survie) ayant souscrit à l'assurance. Cependant, il existe deux différents types d'assurances...

La complémentaire santé :

Une complémentaire est une sorte d'assurance qui vous permet de vous faire rembourser la partie de vos soins non prise en charge par la Sécurité sociale. Populairement, les gens parlent de Mutuelle. Cette complémentaire santé couvre quasiment tous les soins auxquels on doit

faire face, c'est-à-dire les médicaments, les consultations médecins, les analyses médicales, les hospitalisations, la maternité, les frais optiques, les frais dentaires, etc.

L'assurance scolaire :

Le principe d'une assurance scolaire est de protéger les enfants des risques encourus lors des activités scolaires. Celle-ci peut être attribuée pour le milieu scolaire ou bien être appliquée pour un accident non condamnable à l'école. En France, l'assurance scolaire a deux critères...

L'assurance voyage :

L'essor des compagnies des low-cost a entraîné de nombreux bouleversements dans le monde des voyages, parmi ces bouleversements on trouve l'augmentation permanente de voyageurs qui profitent des prix par ces compagnies pour voyager de plus en plus. Face à cette recrudescence de voyageurs, les risques liés au voyage augmentent et de nos jours il vaut mieux voyager en ayant une bonne assurance voyage...

L'assurance entreprise :

On peut faire appel à la responsabilité civile d'une entreprise pour diverses raisons, et selon ces raisons, le montant de l'indemnisation peut être plus ou moins grand. Parfois, le montant de l'indemnisation est tellement qu'il peut amener une entreprise à mettre la clé sous la porte. C'est pour cela qu'il est fortement conseillé de souscrire à une assurance responsabilité civile.

L'assurance crédit :

L'assurance crédit est une assurance qui permet à n'importe qui d'assurer l'ensemble de ces crédits au cas où il lui arriverait quelque chose (décès, invalidité, maladie, etc.). Cela signifie que s'il arrive quelque chose à l'emprunteur, et bien l'organisme auquel vous avez souscrit une assurance crédit, va rembourser vos crédits du temps que vous puissiez retrouver une situation normale si cela est possible bien sûr.

V- Définition d'une compagnie d'assurance : [4]

La compagnie d'assurance est généralement une société regroupant des assureurs en tout genre. Elle propose des couvertures diverses visant à prendre en charge les remboursements et autres dédommagements qui peuvent éventuellement se produire dans la vie d'un individu.

Le travail d'une compagnie d'assurance est très diversifié, suivant les services qu'elle propose. Plus une société d'assurance sera importante, plus elle offrira de services.

Parmi les couvertures d'assurance les plus courantes, l'on retrouve les assurances portant sur les personnes, les biens matériels, les animaux et les assurances tous risques inclus.

La compagnie d'assurance se charge généralement d'informer le particulier sur son champ d'action dans la mesure où elle est limitée par des lois très strictes. Ainsi, le particulier ou

l'entreprise se doit de répondre à une batterie de critères pour pouvoir prétendre à une couverture de ce genre.

L'on retrouve à titre global les critères de sécurité ou les conditions de travail, définissant le taux de risque encouru par la compagnie d'assurance en acceptant de couvrir une personne physique ou morale. La compagnie d'assurance classique fait ainsi passer un questionnaire test à toute personne demandant une assurance afin d'en dresser un profil bien précis. Ce fonctionnement est commun à la majorité des compagnies existantes dans la mesure où il permet de souscrire en toute connaissance de cause.

VI- Description des différentes composantes d'une compagnie d'assurance :[2]

Abordant la notion de la compagnie d'assurance, on définit en premier lieu l'assurance comme l'activité qui consiste, en échange de la perception d'une cotisation ou prime, à fournir une prestation prédéfinie, généralement financière, à un individu, une association ou une entreprise lors de la survenance d'un risque. Cette couverture du risque est souscrite auprès d'une société qui peut en faire son activité exclusive ou une activité complémentaire. Dans cette dernière option on parle de banque et la première alternative est désignée sous le vocable de compagnies d'assurances. Diverses professions propres sont liées à cette activité sont :

1- L'Agent Général d'Assurances :

Il est le représentant ou mandataire d'une compagnie d'assurance qui place ses contrats auprès de la clientèle. Il exerce une profession libérale. En amont de l'assurance, ils analysent les risques de leurs clients, puis les conseillent sur les opportunités d'assurance, placent les risques auprès de leurs compagnies d'assurance, suivent quotidiennement la gestion des contrats et assistent leurs clients, de l'ouverture jusqu'à l'indemnisation en cas de sinistre. Ils sont aussi appelés « Assureurs conseils », mandatés par leurs clients pour les représenter face aux compagnies. C'est pourquoi ils sont responsables de leurs résultats auprès de leurs clients. Les Agents Généraux d'assurance jouissent d'un statut particulier d'intermédiaire auprès de leur compagnie mandante. Ils sont libéraux et chefs d'entreprises. Ce statut régit leurs relations avec les sociétés d'assurance. Il faut noter que la plupart du temps un Agent Général d'Assurance pratique aussi le courtage auprès d'autres compagnies et selon un pourcentage déterminé ;

2- Le courtier en assurances:

Il possède le statut de commerçant et représente le client vis-à-vis des compagnies avec lesquelles il travaille. Il est chargé par les assurés de leur trouver les contrats les mieux adaptés et/ou au meilleur coût auprès des compagnies d'assurances. Un assuré a donc le choix de passer directement par un agent ou indirectement par le biais d'un courtier. Ce sont des commerçants inscrits au registre du commerce. La réglementation les oblige à souscrire une garantie financière pour couvrir les fonds qui leur sont confiés. Ils doivent aussi être obligatoirement assurés en responsabilité civile professionnelle.

3- L'expert en sinistres:

Il établit la réalité des dommages et les responsabilités, chiffre leur valeur et détermine les montants d'indemnisation à verser. Certaines compagnies disposent de leur propre expert.

4- Les juristes:

Ils ont en charge le suivi des contentieux et la veille juridique (évolution de la réglementation, jurisprudence). Ils actualisent aussi les documents contractuels (conditions générales, conditions particulières);

5- L'employé d'assurance:

Il assure le contact de la clientèle et les opérations commerciales (front office). Il occupe alors le poste de conseiller clientèle. Si l'employé d'assurance est en charge uniquement du traitement administratif (back office), il occupe un poste de gestionnaire rédacteur. L'employé d'assurance est souvent polyvalent et réalise différents types d'opérations à savoir : rédacteurs de contrats et rédacteurs sinistres.

voici maintenant un exemple d'un organigramme represantant la structure d'une compagnie d'assurance

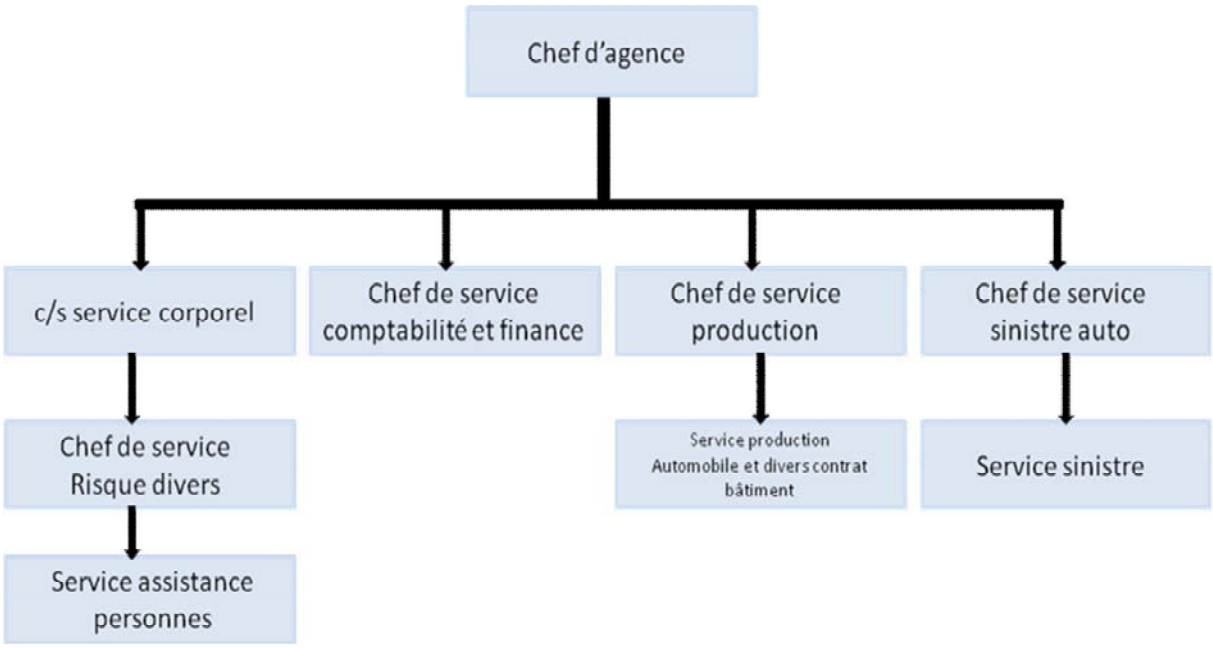


Figure I-1 : Organigramme d'une compagnie d'assurance

VII- Les services proposés par une compagnie d'assurance : [4]

D'un point de vue général, la compagnie d'assurance regroupe ses domaines d'activités suivant les besoins de l'homme moderne.

L'on retrouve en premier lieu l'assurance en responsabilité civile qui est la plus importante et la plus courante des souscriptions. Cette certification est présente dans n'importe quel autre domaine tel que l'assurance automobile ou habitation, car sert à prendre en charge les dédommagements et divers soins résultant d'une atteinte physique ou morale à la vie d'autrui. Cette assurance de la personne laisse ensuite place à une diversité d'assurances secondaires mais non moins importantes à caractère social, telles que les assurances vie, prévoyance ou enfant.

L'assurance santé et autres souscriptions du même genre existent suivant la compagnie d'assurance contactée.

Pour les biens matériels, l'on dispose notamment de l'assurance habitation, l'assurance automobile, l'assurance moto et bien d'autres qui sont considérées comme des souscriptions principales aux assurances catastrophes naturelles, bris de glace, vol ou autres collisions.

Une compagnie d'assurance efficace est capable de proposer des services personnalisés à chaque client. L'on bénéficie d'une équipe ouverte et à l'écoute, rapide en action et disponible. La profusion de compagnies d'assurances a également donné lieu à la création de nombreuses petites souscriptions plus ou moins utiles telles que l'assurance chien et chat, scolarité ou même voyage.

Choisir une bonne compagnie d'assurance est important car l'assuré doit avant tout pouvoir discuter et être à l'aise avec son assureur. Cette communication permanente offre l'avantage de pouvoir s'informer sur les termes du contrat à signer et sur les conditions réelles vécues. Si la profusion de sociétés du genre peut induire tout nouveau venu à l'erreur, il est toujours possible d'avoir recours à un comparateur d'assurances sur le web afin d'optimiser son choix.

VIII- Les assurances gérées en répartition et celles gérées en capitalisation:[5]

Il existe aussi une séparation juridique entre les sociétés d'assurances en fonction du mode de gestion financière (gestion en répartition ou en capitalisation) qui s'appliquent aux assurances qu'elles pratiquent. D'où la nécessité de distinguer les assurances gérées en répartition de celles gérées en capitalisation.

Cela signifie que les sociétés d'assurances qui pratiquent des assurances gérées en répartition ne sont pas autorisées à pratiquer des assurances gérées en capitalisation, et réciproquement, sauf s'il s'agit de risques accessoires. C'est le principe de spécialisation.

1- Les assurances gérées en répartition :

La gestion par répartition consiste, pour l'assureur, à redistribuer aux victimes des sinistres la masse des cotisations payées par l'ensemble des assurés. Cette répartition s'opère par année : les cotisations de l'année paient les sinistres de l'année.

Les assurances de biens et de responsabilité ainsi que certaines assurances de personnes comme les assurances complémentaires santé et dommages corporels sont gérées en répartition.

2- Les assurances gérées en capitalisation:

La capitalisation est une technique de gestion financière : l'assureur place une partie des primes collectées pour les faire fructifier et réinvestit les revenus financiers ainsi obtenus de manière à accroître la somme initiale.

L'assureur constitue de cette manière un capital qui lui servira à payer, au moment voulu, la prestation prévue dans le contrat.

La capitalisation concerne les assurances qui présentent les particularités suivantes :

- elles sont souscrites pour une longue période et comportent un aspect épargne ;
- elles portent sur des risques non constants dont la fréquence augmente ou diminue au cours du contrat. C'est le cas de la probabilité de décès ou de survie de la personne humaine.
- par le Code de la sécurité sociale.

3- Les autres acteurs du monde de l'assurance :[2]

Le monde de l'assurance ne se limite pas aux seuls assureurs, mais engage de nombreux autres acteurs.

Le champ de l'assurance recouvre notamment l'ensemble des métiers qui s'exercent dans les entreprises dont les activités sont régies par le Code des assurances, à savoir :

- Les métiers exercés au sein des sociétés anonymes (SA)
- Les métiers exercés au sein des sociétés d'assurances mutuelles (SAM.)
- Les métiers exercés au sein des sociétés mutuelles d'assurances (SMA).
- Les métiers d'intermédiaires tels que les agents généraux d'assurances et courtiers.
- Les métiers des auxiliaires d'assurances représentés par les experts d'assurances.

IX- Le fonctionnent des assurances et les métiers disponibles dans le domaine des assurances : [2]

Le fonctionnement d'une compagnie d'assurance est très complexe puisque celle-ci doit être capable de payer à ses assureurs si un préjudice intervient, mais elle doit aussi subvenir à ses

propres besoins. Ce n'est qu'en réussissant à équilibrer la balance "recettes/sinistres", qu'une compagnie d'assurance peut continuer à vivre, sans cet équilibre c'est la faillite assurée.

Pour parvenir à cet équilibre, les compagnies d'assurances utilisent plusieurs méthodes. L'une de ces méthodes consiste à placer les cotisations de ses assureurs et d'en retirer les bénéfices pour parvenir à indemniser les assureurs en cas de préjudices, mais aussi à assurer les frais de fonctionnement de l'entreprise. Cette méthode est particulièrement rentable par les assurances dites de longs termes (comme la responsabilité civile), car pour ces assurances les indemnités sont versées longtemps après la date du sinistre de ce fait l'argent placé à tout le temps pour fructifier, par contre cette méthode est un peu moins utile pour les assurances à déroulement court.

Malgré tout, cette méthode a ses limites, en effet si la compagnie d'assurance doit faire face à un grand nombre d'indemnisations en même temps, il se peut qu'elle perde tout son bénéfice voir même passer en déficit. De ce fait, les assurances essaient par tous les moyens de ne pas en arriver là et si nécessaire elles augmenteront les cotisations pour pouvoir rééquilibrer la balance "recettes/sinistres".

De plus, pour éviter d'être dépassé un sinistre imprévu et de très grande importance, certaines assurances peuvent elles-mêmes souscrire à une assurance auprès d'une compagnie spécialisée, on appelle ça "la réassurance". Il faut savoir que la réassurance est quasi automatique dans la plupart des risques industriels, car ils entraînent beaucoup trop de frais.

X- Sinistres et contrat d’assurance :

1- Sinistre : [17]

1-1- Définition :

Par définition, le sinistre est un évènement qui englobe tout dommage matériel ou corporel et qui entraine une indemnisation de la part de votre assurance ou de l’assurance du tiers identifié mis en faute.

1-2- Différents types de sinistre :

Incendie

Le sinistre incendie pouvant détruire l’ensemble de vos biens ou de votre entreprise, l’incendie est un des sinistres les plus dévastateurs.

Dégâts des eaux

Sinistre malheureusement très fréquent, le dégât des eaux nécessite souvent l'exercice d'un recours contre le responsable et sa compagnie d'assurance et la mise en place de mesures conservatoires pour limiter les dommages.

Perte d'exploitation

Garantie indispensable pour les entreprise et ce quelque soit le type de sinistre, la garantie perte d'exploitation, permettra une indemnisation des pertes financières liées au sinistre et à ses conséquences telles que la fermeture d'un établissement pendant la durée des travaux de reconstruction ou de réfection, la perte d'un marché et le retour à une activité pérenne.

Vol, cambriolage, hold-up, attaque à mains armées

Garantie dans les contrats en fonction du mode opératoire, le vol est un sinistre dont l'indemnisation pourra révéler beaucoup d'obstacles et en conséquence pour lequel l'intervention d'un expert représentant l'assuré sera une garantie en matière de protection des assurés.

Catastrophes naturelles, tempêtes, cyclones

Garanties législatives, ces sinistres causent souvent des dégâts très importants

L'accident : provoqué par un tiers, ou un animal sauvage, arrivé seul,… l'accident ou l'accrochage est le plus courant des sinistres.

Le vol : au retour d'une course, au réveil le matin, en rentrant d'une soirée, véhicule disparu.

Le bris de glace : un caillou projeté par un poids-lourd, le gel, un acte de vandalisme, car-jacking…

2- Contrat d'assurance : [18]

2-1- Définition :

Le contrat d'assurance peut être défini comme une opération par laquelle « l'Assureur » s'engage à garantir un risque au profit d'une personne appelée « Assuré », qui en contrepartie versera une somme d'argent appelée « prime ».

Le bénéficiaire est la personne qui bénéficie des prestations versées par l'Assureur.

Selon les garanties du contrat d'assurance, l'Assuré peut être ou non le bénéficiaire.

Exemple : Dans un contrat d'assurance vie, l'Assuré désignera un ou des bénéficiaire(s) en cas de décès de celui-ci. Dans un contrat en couverture de prêt, le bénéficiaire sera l'établissement prêteur. Enfin dans un contrat de retraite, l'Assuré sera le bénéficiaire des prestations.

2-2- Caractéristique d'un contrat d'assurance :

Le contrat d'assurance possède plusieurs caractéristiques.

Il doit être :

- **consensuel**

Il est conclu par l'accord des parties.

- **d'adhésion à titre onéreux**

Le contrat d'assurance est rédigé par l'assureur et il est conclu en contrepartie d'une prestation pour l'assuré et d'une prime pour l'assureur

- **synallagmatique**

Les obligations des parties sont réciproques.

- **de bonne foi**

La bonne foi doit être présente à chaque étape du contrat.

Les Tribunaux et le **code des assurances** se réfèrent à ce critère de bonne foi.

- **aléatoire**

L'aléa est le caractère essentiel du contrat d'assurance puisque la prestation de l'assureur ne sera versée qu'en cas de réalisation d'un évènement incertain.

- **à exécution successive**

Le contrat d'assurance est échelonné dans le temps. Il est souvent à tacite reconduction, c'est-à-dire qu'il se renouvelle de manière automatique à date fixe.

2-3- Classification des contrats d'assurance :

On peut tout d'abord distinguer deux grandes catégories de contrats d'assurance. Les contrats d'assurance de personnes et les contrats d'assurance de dommages.

Cette classification est celle utilisée par le Code des assurances. Il permet de déterminer les règles juridiques applicables.

Les assurances de personnes sont des contrats où le risque garanti est une atteinte sur la personne assurée.

Voici la liste des principaux contrats d'assurance de personne.

- Contrat d'assurance de santé
- Contrat d'assurance obsèques
- Contrat d'assurance prévoyance
- Contrat d'assurance de retraite

- Contrat d'assurance de type épargne
- Contrat d'assurance en couverture de prêt

Les assurances de dommages touchent les biens de la personne assurée ou les dommages causés aux tiers.

Exemple : assurance habitation, assurance automobile, assurance de responsabilité civile.

On distingue également les contrats d'assurance **collective** des contrats d'assurance individuelle

Un contrat d'assurance collectif est un contrat souscrit par une personne morale (entreprises, établissements financiers etc…) pour couvrir un ensemble de personnes qui vont adhérer au contrat.

L'assuré est ainsi dénommé également adhérent. Le souscripteur sera donc une personne distincte de l'assuré.

Exemple : assurance de prévoyance ou de retraite d'entreprise

Dans un contrat d'assurance individuel, l'assuré conclut directement avec l'assureur.

Enfin, il convient de distinguer les contrats d'assurance **facultative et obligatoire.**

XI- Le rôle des assurances :[6]

L'assurance joue des rôles multiples:

1- Premièrement, l'assurance remplit les fonctions de sécurité tant du point de vue individuel que du point de vue général:

Au regard de l'assuré, l'assurance a un caractère moral. En effet, elle est le produit de la vertu de prévoyance. Au lieu d'attendre d'être frappé par les coups du sort et de se trouver ensuite plus ou moins à la charge de la société, l'assuré prend des précautions: il songe à l'avenir ; et à l'avance, de façon constante, il fait, volontairement, un sacrifice personnel pour se prémunir contre le hasard. Il y a même certaines assurances où l'assuré agit, non pour lui-même, mais dans l'intérêt d'autrui, de façon désintéressée ou tout au moins pour accomplir un devoir moral.

En dehors de cette vertu morale, l'assurance a pour rôle fondamental de conférer aux assurés la sécurité dont ils ont besoin. Elle leur apporte la confiance dans l'avenir: grâce à elle, ils sont protégés contre les risques du hasard, qui les menace, eux ou leur patrimoine. L'assurance répond à un besoin incontestable de l'individu: exposé aux coups du sort dans sa personne ou dans ses biens, il ne peut agir d'une façon pleine et efficace que s'il peut se prémunir contre l'aléa qu'il redoute.

Ce besoin de sécurité individuelle auquel répond l'assurance est d'autant plus grand aujourd'hui que la vie moderne se caractérise par un accroissement des risques, donc par une augmentation de l'insécurité. L'assurance devient ainsi, de nos jours, une véritable nécessité pour l'homme, spécialement pour l'homme d'action ou l'homme d'affaire exposé professionnellement à de multiples risques (incendie, vol, responsabilité) contre lesquels il est obligé de se protéger.

L'assurance, sur le plan de la sécurité, présente un **intérêt général et social**. En donnant la sécurité aux individus, l'assurance renforce l'économie nationale: elle devient un facteur de production. Elle permet, en effet, de conserver les forces productives, travail et capital, tout au moins de les reconstituer aisément et, à cet égard, elle accroît d'autant plus la puissance économique que les biens nouveaux substitués aux biens détruits peuvent être d'un rendement supérieur. Elle accroît aussi cette puissance par son action préventive en incitant ou même en obligeant les assurés à utiliser les procédés les plus perfectionnés.

2- Deuxièmement, l'assurance permet, par l'accumulation des primes, la constitution des capitaux.

En effet, grâce à l'assurance, des sommes généralement modiques et qui, sans cela, auraient été vraisemblablement consommées, sont réunies au sein de l'entreprise, conservées, placées jusqu'au jour où elles doivent servir au règlement des sinistres.

3- Troisièmement, l'assurance remplit une fonction de crédit tant au profit des assurés que de l'économie générale:

Elle est tout d'abord, et sous des formes diverses, un moyen de crédit pour l'assuré. Elle facilite en premier lieu son crédit en renforçant les garanties qu'il offre à ses créanciers. Ainsi un débiteur hypothécaire est pratiquement obligé, par une clause de style, d'assurer contre l'incendie, l'immeuble hypothéqué, afin de donner à son créancier la certitude d'être indemnisé au cas où l'immeuble serait détruit par le feu.

L'assurance joue un rôle non négligeable au regard du crédit général. Les compagnies d'assurance sont obligées de constituer des réserves (provisions) et de les représenter en partie par les titres émis par l'Etat et les collectivités publiques, de sorte que, par leur placement imposé, elles soutiennent le crédit général du pays.

4- Quatrièmement, l'assurance joue un rôle international :

Ce rôle se réalise de deux façons. D'une part, il appartient aux compagnies nationales de souscrire directement des assurances à l'étranger; d'autre part et surtout c'est par la réassurance que se réalise le rôle international de l'assurance: après avoir traité directement avec ses assurés, l'assureur rétrocède, soit facultativement, soit obligatoirement (traité de réassurance),

une partie de ses risques à un réassureur, le plus souvent étranger, de sorte que les incidences des sinistres nationaux se répercutent en définitive sur l'économie de plusieurs pays, ce qui est un facteur d'équilibre et de stabilité générale.

XII- Compagnies d'assurances en France : [28]

Il existe **plusieurs dizaines de compagnies d'assurance** en France, proposant des contrats pour les particuliers ou les professionnels. Nous en avons établi une **liste complète**, classée en trois rubriques :

- les *compagnies généralistes* proposent une gamme complète de contrats d'assurance pour les particuliers et les professionnels. Leur taille et leur chiffre d'affaire sont en général très importants. On peut citer par Exemple : Aviva Assurances, AXA, Carrefour Assurances – CARMA, Gan Assurances, Generali, Groupe Prévoir…

- les *compagnies spécialistes* assurent seulement une gamme de produits spécifiques. Exemple : ACE Europe, Ageas France, Albingia, AMF Assurances.

- **les *compagnies en ligne*** vendent leurs contrats en direct, sans l'intermédiaire de courtiers ou d'agents généraux comme : Allsecur, AssurAvenue, Direct Assurance, Eurofil, L'Olivier Assurances, Prévoir Direct, SwissLife Direct.

XIII- Présentation de l'entreprise Direct Assurance : [8]

Direct Assurance est une compagnie d'assurances à distance créée en 1992 et filiale à 100 % du groupe AXA. Spécialisée dans l'assurance auto, elle propose également des assurances habitation et moto. Pionnière en France de l'assurance en direct multi canal - par téléphone et par Internet, Direct Assurance est le leader de l'assurance automobile en ligne.

XIV- Conclusion:

En apportant la sécurité aux hommes, l'assurance favorise l'éclosion d'un grand nombre d'activité qu'il n'oserait entreprendre sans elle.

Nombreuses sont les activités qui ne seront pas entreprises sans un tel soutient qu'il s'agisse de la pratique de sport dangereux, de métiers dangereux, de l'utilisation de nouveaux modes de transport, de l'exploitation de nouvelles formes d'énergies …

L'assurance est devenue une nécessité pour l'homme. Elle doit s'adapter à ses besoins et s'étendre sans cesse à des risques nouveaux.

Elle encourage de ce fait l'innovation, c'est un facteur de progrès social et de développement économique, et le nombre de compagnies d'assurances ne cesse de croitre et la concurrence entre ces dernières est devenue rude.

Pour cela nous avons parlé dans ce chapitre des assurances de façon générale et de leurs différents types ainsi que son rôle primordial pour la société et pour l'économie.

Le chapitre suivant sera consacré aux technologies web et leurs intérêts. Parmi ces technologies nous avons étudié les frameworks.

Chapitre II:

Etude comparative des frameworks

I- Introduction :

Dans le présent chapitre nous allons donner une vue générale sur les technologies web et les langages utilisés pour les développer. Parmi ces technologies nous avons choisi de définir la notion de framework et nous avons donc consacré une partie pour étudier les différents types existants, nous avons ensuite fait une étude comparative entre les trois framework les plus utilisés (Hibernate, Spring et Struts) pour finir par le choix du plus adéquat à la réalisation de notre application.

II- Web et applications web : [11]

1- Définition du web :

WWW ou le World Wide Web (littéralement "toile d'araignée mondiale") est le service d'information le plus récent de l'internet. Il repose sur ce qu'on appelle de l'hypertexte qui travaille en mode client-serveur. Le concept Web correspond à une base de données universelle où les documents de tous types sont identifiés de manière unique, et pointent les uns vers les autres par des liens. WWW a été développé par le CERN (laboratoire européen de recherche en physique des particules) à Genève. Le projet a démarré en mars 1989. L'objectif était de proposer une solution pour la communication de l'information dans la communauté de la physique des hautes énergies en utilisant l'internet.

Les clients et les serveurs du web utilisent un protocole de communication dit http.

La simplicité de la relation entre les clients et les serveurs permet au web d'offrir des services de commerce électroniques.

2- Technologies du web : [20]

2-1- Technologies côté client

- Les navigateurs, leurs impacts.
- Les plug-ins.
- Les URI, le protocle HTTP.
- Le langage HTML et ses limites, HTML5. CSS et CSS3.
- JavaScript et framework associés (JQuery).
- DOM, DHTML, XHTML.
- Les interfaces riches et leur ergonomie : composants ActiveX, applets Java, Java FX2, Flash, Flex, Silverlight, Ajax.
- Les interfaces graphiques XML : XUL, XAML, SVG, MXML, XForms.
- Les spécificités des terminaux mobiles et des tablettes : le responsive web design.
- Présentation.
- Les XML Schema.

- Les parseurs.
- Les traitements XSLT.
- La publication avec XSL-FO.
- Les services Web.
- Langages XML (ebxml, XHTML, BPML...).

2-2- Technologies côté serveur

- Les architectures n-tiers.
- Les approches orientées composant.
- L'architecture JEE 6 et les frameworks associés (Struts, Spring, Hibernate...).
- Le framework .NET 4 et les frameworks associés (NHibernate, ...).
- PHP, Zend, architecture LAMP, Zope/Plone...

III- L'architecture client/serveur : [12]

1- Définition du *client/serveur :*

L'architecture client-serveur est un modèle de fonctionnement logiciel qui peut se réaliser sur tout type d'architecture matérielle (petites ou grosses machines), à partir du moment ou ces architectures peuvent être interconnectées.

On parle de fonctionnement logiciel dans la mesure où cette architecture est basée sur l'utilisation de deux types de logiciels, à savoir un logiciel serveur et un logiciel client s'exécutant normalement sur 2 machines différentes. L'élément important dans cette architecture est l'utilisation de mécanismes de communication entre les 2 applications.

Le dialogue entre les applications peut se résumer par :

- Le client demande un service au serveur.
- Le serveur réalise ce service et renvoie le résultat au client

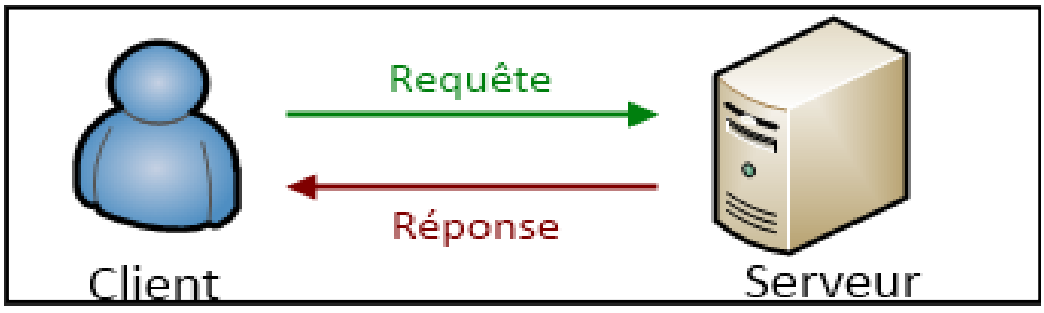


Figure II.1: architecture client /serveur [16]

2- Les différents modèles du *client/serveur* :

En faits les différences sont liées aux services qui sont assurés par le serveur.

On distingue couramment :

2-1- Le client/ serveur de données :

Dans ce cas le serveur assure les taches de gestion, stockage et de traitement de données.

C'est le cas le plus connu de client/serveur et qui est utilisé par les tous grands SGBD :

La base de données avec tous ses outils (maintenance, sauvegarde …) est installée sur un poste serveur.

Sur les clients un logiciel d'accès est installé permettant d'accéder à la base de données du serveur. Tous les traitements sur les données sont effectués sur le serveur qui renvoie les informations demandées (souvent à travers une requête SQL) par le client.

2-2- Le client/ serveur de présentation :

Dans ce cas la présentation des pages affichées par le client est intégralement prise en charge par le serveur .cette organisation présente l'inconvénient de générer un fort trafic réseau.

2-3- Le client/ serveur de traitement :

Dans ce cas le serveur effectue des traitements à la demande du client. Il peut s'agir de traitement particulier sur des données, de vérification de formulaire de saisie, de traitement d'alarme …etc.

Ces traitements peuvent être réalisés par des programmes installés sur des serveurs mais également intégrés dans des bases de données (triggers, procédures stockées…) dans ce cas la partie donnée et traitement sont intégrés.

2-4- Architecture client/serveur à deux niveaux :

L'architecture client/serveur à deux niveaux (2 tiers) est l'architecture la plus classique dont la communication se fait directement entre le client et le serveur en utilisant deux approches.

Dans la première, les traitements sont assurés par le client et les données sont gérées par le serveur. Par contre dans la deuxième approche les traitements et les données de l'interface utilisateur sont séparées, c'est le serveur qui assure les traitements, le client s'occupe seulement de l'interface utilisateur.

L'architecture client/serveur à deux niveaux se schématise comme suit :

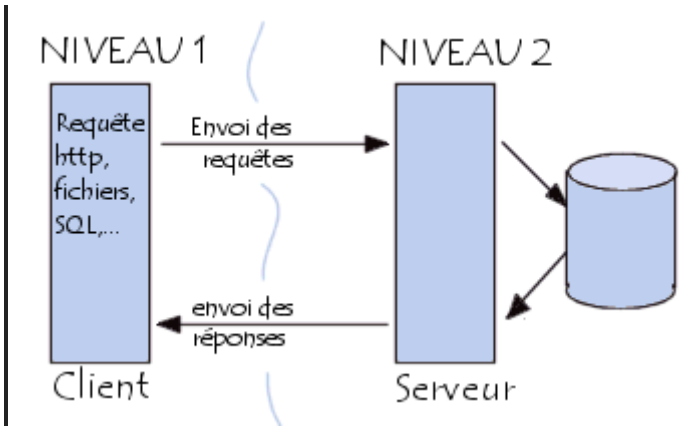


Figure II.2 : Architecture client/serveur à deux niveaux [15]

2-5- Architecture client/serveur à trois niveaux :

Dans l'architecture à trois niveaux, il existe un niveau intermédiaire. C'est une architecture partagée entre :

Le demandeur de ressources qui est le client. Le fournisseur de ressources qui est le serveur d'application (Middleware) en faisant appel à un autre serveur.

Le fournisseur de services au premier serveur (serveur secondaire qui est généralement un serveur de bases de données).

L'architecture à trois niveaux est recommandée dans des applications possédant les caractéristiques suivantes :

- Une plus grande flexibilité (souplesse).
- Une plus grande sécurité (la sécurité peut être définie pour chaque service).
- De meilleures performances (les tâches sont partagées).
- L'installation d'un gestionnaire de base de données n'est obligatoire que sur le serveur.
- Indépendance des couches par rapport à la localisation physique, le niveau traitement pouvant être redistribué à l'infini, ce qui permet le déploiement d'applications à grande échelle.

L'architecture client/serveur à trois niveaux se présente comme suit :

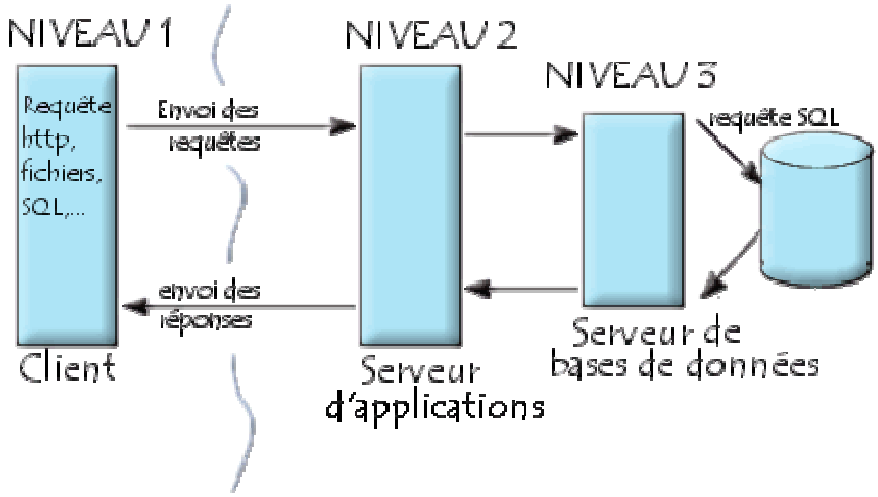


Figure II.3 : architecture client/serveur à 3 niveaux [15]

2-6- Architecture multi niveaux :

Dans l'architecture à trois niveaux, chaque serveur (Niveau 1 et 2) effectue une tâche spécialisée. Ainsi, un serveur peut utiliser les services d'un ou plusieurs autres serveurs afin de fournir son propre service par conséquence, l'architecture à trois niveaux est potentiellement une architecture à N niveaux. Cette architecture est présentée dans le schéma suivant :

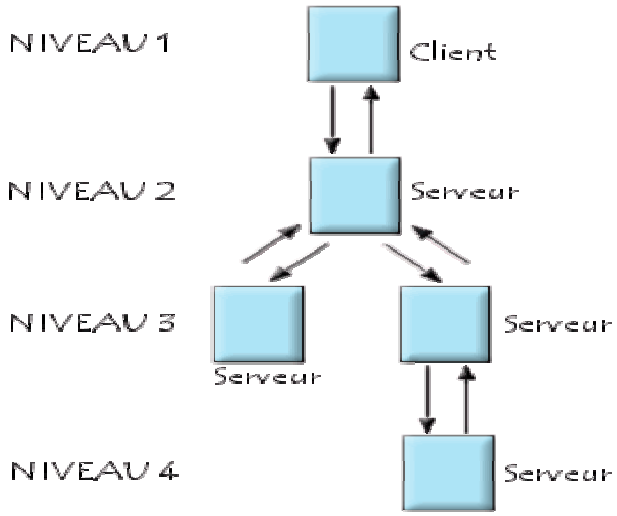


Figure II.4 : architecture client serveur à multi niveaux [15]

IV- Le modèle MVC : [9]

Le modèle MVC (Model View Controler) a été initialement développé pour le langage Smalltalk dans le but de mieux structurer une application avec une interface graphique.

Ce modèle est un concept d'architecture qui propose une séparation en trois entités des données, des traitements et de l'interface :

- le Modèle représente les données de l'application généralement stockées dans une base de données
- la Vue correspond à l'IHM (Interface Homme Machine)
- le Contrôleur assure les échanges entre la vue et le modèle notamment grâce à des composants métiers

Initialement utilisé pour le développement des interfaces graphiques, ce modèle peut se transposer pour les applications web sous la forme d'une architecture dite 3-tiers : la vue est mise en oeuvre par des JSP, le contrôleur est mis en oeuvre par des servlets et des Javabeans. Différents mécanismes peuvent être utilisés pour accéder aux données.

L'utilisation du modèle MVC rend un peu plus compliqué le développement de l'application qui le met en oeuvre mais il permet une meilleure structuration de celle-ci.

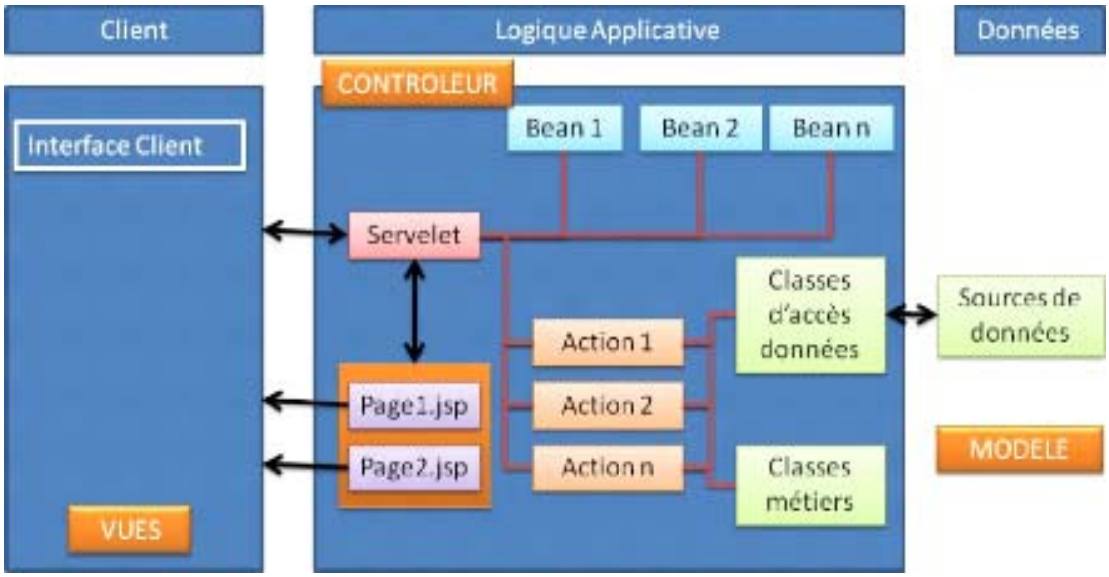


Figure II.5 : architecture du modèle MVC [1]

1- Le modèle MVC type1 :

Dans ce modèle, chaque requête est traitée par un contrôleur sous la forme d'une servlet. Celle-ci traite la requête, fait appel aux éléments du model si nécessaire et redirige la requête vers une JSP qui se charge de créer la réponse à l'utilisateur.

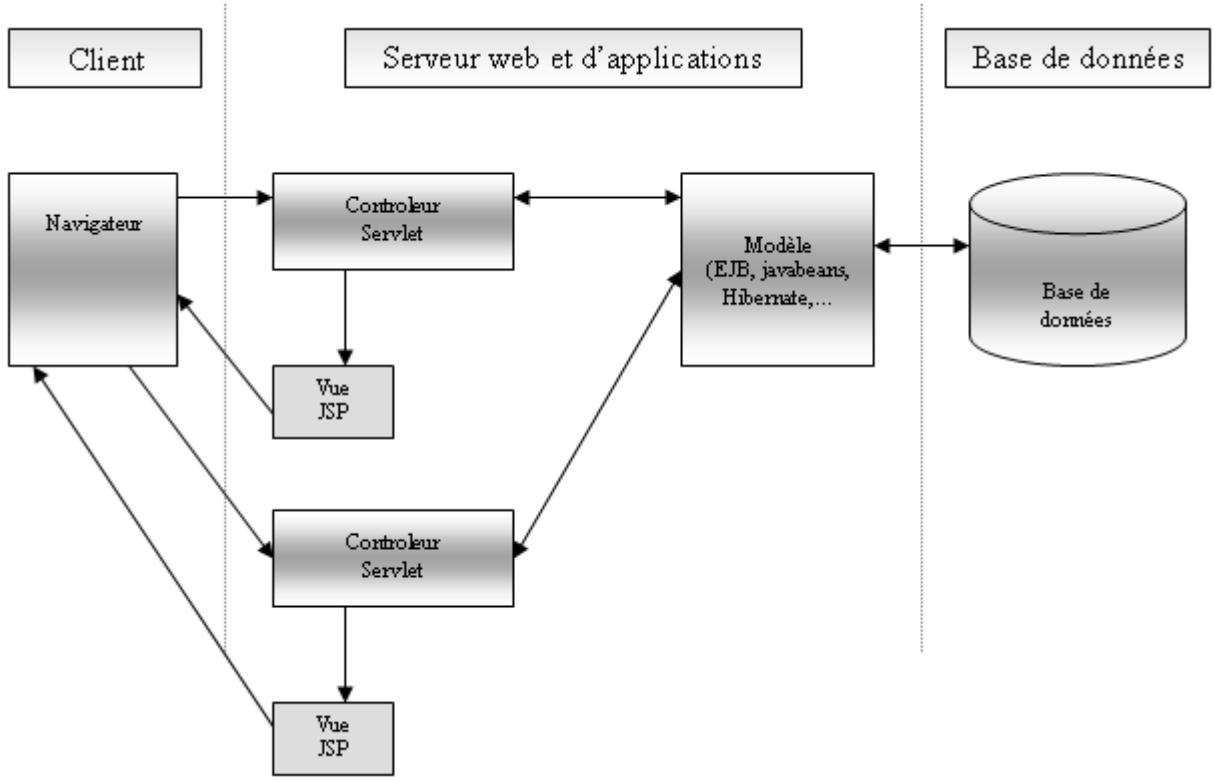


Figure II.6 : modèle MVC de type1

L'inconvénient est donc une multiplication du nombre de servlets nécessaires à l'application : l'implémentation de plusieurs servlets nécessite beaucoup de code à produire d'autant que chaque servlet doit être déclarée dans le fichier web.xml.

2- Le modèle MVC de type2 :

Le principal défaut du modèle MVC est le nombre de servlets à développer pour une application.

Pour simplifier les choses, le modèle MVC model 2 ou MVC2 de Sun propose de n'utiliser qu'une seule et unique servlet comme contrôleur. Cette servlet se charge d'assurer le workflow des traitements en fonction des requêtes http reçues.

Le modèle MVC 2 est donc une évolution du modèle 1 : une unique servlet fait office de FFF la forme d'un fichier au format XML.

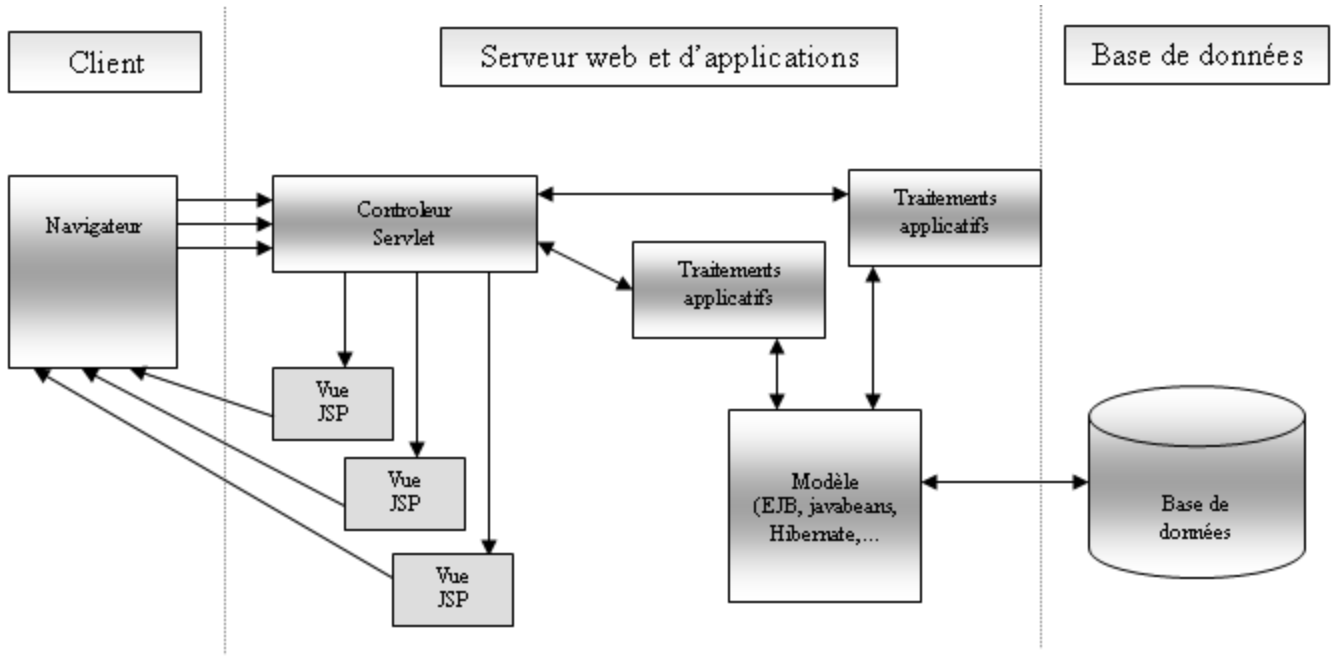


Figure II.7: le modèle MVC de type2

Le modèle MVC II conserve les principes du modèle MVC, mais il divise le contrôleur en deux parties en imposant un point d'entrée unique à toute l'application (première partie du contrôleur) qui déterminera à chaque requête reçue les traitements applicatifs à invoquer dynamiquement (seconde partie du contrôleur).

Une application web implémentant le modèle MVC de type II utilise une servlet comme contrôleur traitant les requêtes. En fonction de celles-ci, elle appelle les traitements dédiés généralement encapsulés dans une classe.

Dans ce modèle, le cycle de vie d'une requête est le suivant :

1. Le client envoie une requête à l'application, requête est prise en charge par la servlet faisant office de contrôleur.
2. La servlet analyse la requête et appelle la classe dédiée contenant les traitements
3. Cette classe exécute les traitements nécessaires en fonction de la requête, notamment, en faisant appel aux objets métiers.
4. En fonction du résultat, la servlet redirige la requête vers la page JSP
5. La JSP génère la réponse qui est renvoyée au client.

Dans une application web classique on utilise souvent des servelets pour répondre aux requêtes des utilisateurs (clients). Côté client, on développe des interfaces HTML ou des JSP. Avec les difficultés rencontrées avec ce type de technologies, les développeurs choisit de créer des moyens plus efficaces et plus pratiques tels que les frameworks. Dans la section suivante nous allons donner une vue général sur ces frameworks et leur utilisation.

V- Framework et plateformes de développement web :

1-Définition et signification de Framework : [7]

En informatique, un framework est un espace de travail modulaire. C'est un ensemble de bibliothèques, d'outils et de conventions permettant le développement d'applications. Il fournit suffisamment de briques logicielles et impose suffisamment de rigueur pour pouvoir produire une application aboutie et facile à maintenir. Ces composants sont organisés pour être utilisés en les uns avec les autres.

Un framework fournit un ensemble de fonctions facilitant la création de tout ou d'une partie d'un système logiciel, ainsi qu'un guide architectural en partitionnant le domaine visé en modules. Un framework est habituellement implémenté à l'aide d'un langage à objets, bien que cela ne soit pas strictement nécessaire : un framework objet fournit ainsi un guide architectural en partitionnant le domaine visé en classes et en définissant les responsabilités de chacune ainsi que les collaborations entre classes. Un sous-ensemble de ces classes peut être des classes abstraites.

Le déploiement à grande échelle de bibliothèques d'objets exige un framework ; celui-ci fournit un contexte où les composants sont réutilisés.

Si l'utilisation du terme bibliothèque est limitée à l'ensemble des fonctions du système, le terme de framework peut être employé par extension pour inclure également l'architecture logicielle préconisée pour cette bibliothèque (organisation en couches, utilisation du modèle MVC, etc), voire l'environnement de développement bâti autour (Microsoft .NET + Visual Studio, etc) même si celui-ci est capable de gérer différents frameworks.

Les frameworks se présentent sous diverses formes, qui peuvent inclure tout ou partie des éléments suivants :

- un ensemble de classes généralement regroupées sous la forme de bibliothèques pour proposer des services plus ou moins sophistiqués
- un cadre de conception reposant sur les design patterns pour proposer tout ou partie d'un squelette d'applications
- des recommandations sur la mise en œuvre et des exemples d'utilisation des normes de développement des outils facilitant la mise en œuvre.

2- Composant d'un framework : [13]

2-1- Bibliothèque :

Un *Framework* peut être ou peut contenir une ou des bibliothèques. Une bibliothèque, en informatique, est un ensemble d'algorithmes qui peut être répétitif. Donc les bibliothèques sont utilisées pour que nous n'ayons pas à les réécrire à chaque nouveau projet ou plusieurs fois dans le même projet. Une bibliothèque peut aussi être un ensemble d'algorithmes offert

par des développeurs pour d'autres développeurs pour leur simplifier la vie. Par exemple, la bibliothèque *Math* qui contient une grande quantité de formules mathématiques.

2-2- Structure :

En plus de contenir ou non des bibliothèques, un *Framework* peut avoir une structure prédéfinie pour concevoir des programmes. Par exemple, la fenêtre de l'application et ses composantes déjà conçues et pouvant être modifiées selon les goûts et les besoins des programmeurs. La structure peut aussi être une architecture du code permettant une utilisation facile de conception complexe.

2-3- Autres :

Les *Framework* peuvent avoir d'autres éléments intégrés pour faciliter les programmeurs selon le but ou le domaine de celui-ci. Le *Framework* XNA de Microsoft offre un Content Pipeline qui est un conteneur d'éléments extérieurs au programme, comme des images, des vidéos, des fichiers audio, des styles de textes, des effets, etc.

3- Objectif d'un framework : [14]

L'objectif premier d'un framework est d'améliorer la productivité des développeurs qui l'utilisent. Souvent organisé en différents composants, un framework offre la possibilité au développeur final d'utiliser tel ou tel composant pour lui faciliter le développement, et lui permet ainsi de se concentrer sur le plus important.

Autrement dit, le framework s'occupe de la forme et permet au développeur de se concentrer sur le fond.

La mise en oeuvre d'un framework permet notamment :

- de capitaliser le savoir-faire sans "réinventer la roue"
- d'accroître la productivité des développeurs une fois le framework pris en main
- d'homogénéiser les développements des applications en assurant la réutilisation de composants fiables
- donc de faciliter la maintenance notamment évolutive des applications

Cependant, cette mise en œuvre peut se heurter à certaines difficultés :

- le temps de prise en main du framework par les développeurs peut être plus ou long en fonction de différents facteurs (complexité du framework, richesse de sa documentation, expérience des développeurs, ...)
- les évolutions du framework qu'il faut répercuter dans les applications existantes.

4- Avantages et inconvénients d'un framework : [14]

Avantage :

L'avantage premier est le gain en productivité. Mais il en existe bien d'autres. On peut les classer en plusieurs catégories : le code, le travail et la communauté.

Tout d'abord, un framework aide à réaliser un « bon code ». Par « bon code », on entend qu'il incite, de par sa propre architecture, à bien organiser le code. Et un code bien organisé est un code facilement maintenable et évolutif. De plus, un framework offre des briques prêtes à être utilisées, ce qui évite de réinventer la roue, et surtout qui permet d'utiliser des briques puissantes et éprouvées. En effet, ces briques sont développées par des équipes de développeurs chevronnés, elles sont donc très flexibles et très robustes. On économise ainsi des heures de développement.

Ensuite, un framework améliore la façon dont on travaille. En effet, dans le cas d'un site internet, on travaille souvent avec d'autres développeurs PHP et un designer. Un framework aide doublement dans ce travail en équipe. D'une part, un framework utilise presque toujours l'architecture MVC, c'est donc une façon d'organiser son code qui sépare le code PHP, java ou autre du code HTML. Ainsi, le designer peut travailler sur des fichiers différents de ceux du développeur, fini les problèmes d'édition simultanée d'un même fichier. D'autre part, un framework a une structure et des conventions de code connues. Ainsi, on peut facilement travailler avec un autre développeur : s'il connaît déjà le framework en question, il s'intégrera très rapidement au projet.

Enfin, le dernier avantage est la communauté soutenant chaque framework. C'est elle qui fournit les tutoriaux ou les cours (de l'aide sur les forums, et bien sûr les mises à jour du framework.) Ces mises à jour sont très importantes

Un framework, c'est aussi :

- Une communauté active qui utilise le framework et qui contribue en retour ;
- Une documentation de qualité et régulièrement mise à jour ;
- Un code source maintenu par des développeurs attitrés ;
- Un code qui respecte les standards de programmation ;
- Un support à long terme garanti et des mises à jour qui ne cassent pas la compatibilité ;
- Etc.

Inconvénients :

La courbe d'apprentissage qui est plus élevée. En effet, pour maîtriser un framework, il faut un temps d'apprentissage non négligeable. Chaque brique qui compose un framework a sa complexité propre qu'il faudra appréhender.

Il est à noter également que pour les frameworks les plus récents, il faut également être au courant des dernières nouveautés du langage utilisé. Penser notamment à la programmation orientée objet et aux namespaces. De plus, connaître certaines bonnes pratiques telles que l'architecture MVC est un plus.

5- Les différents types de Framework : [7]

5-1- Framework d'infrastructure système :

Pour développer des systèmes d'exploitation, des interfaces graphiques, des outils de communication. (Exemple : Framework .Net, Eclipse, NetBeans, Struts)

5-2- Framework d'intégration intergicielle :

Pour fédérer des applications hétérogènes. Pour mettre à dispositions différentes technologies sous la forme d'une interface unique. (Exemple : Ampoliros avec ses interfaces RPC, SOAP, XML).

5-3- Frameworks d'entreprise :

Pour développer des applications spécifiques au secteur d'activité de l'entreprise.

5-4- Frameworks orientés Système de gestion de contenu :

Les principaux avantages de ces frameworks sont la réutilisation de leur code, la standardisation du cycle de vie du logiciel (Spécification, développement, maintenance, évolution), ils permettent de formaliser une architecture adaptée au besoin de l'entreprise. Ils tirent parti de l'expérience des développements antérieurs.

Ces frameworks sont en quelque sorte des progiciels extrêmement souples et évolutifs.

6- Framework et langages de programmation : [8]

On peut également classer les framework selon le langage de programmation à utiliser, on peut trouver donc des framework JAVA, framework PHP…etc

Voici quelques exemples de langages et leurs framework correspondants :

- **Python** :Django ,Flask **,** Scrapy **,** Karrigell ,Twisted **,**Web2py **,** CherryPy **,**Pyramid **,** TurboGears
- **Ruby :** Ruby on Rails **,** Sinatra **,** Merb
- **Perl :** Ruby on Rails · Sinatra · Merb
- **Smalltalk :**Seaside
- **C++ :**Wt
- **JavaScript :**
  Côté serveur : Node,js **,**Meteor **,**Express

Côté client : jQuery **,** AngularJS ,MooTools **,** Dojo **,** Meteor **,** Backbone,js **,** Ember,js **,** Ext, JS **,** qooxdoo **,**YUI **,**Cappuccino **,** script, aculo, us

- **CSS** Blueprint **,** Bootstrap ,Foundation
- **JVM :** Java EE **,**
- SpringMVC **,** Struts **,** Tapestry **,**Play ,Stripes **,**ZK **,**Cocoon **,**GWT **,**Echo **,**AppFuse **,** Grails **,**RAP **,**Wicket **,** WebObjects
- **.NET : .**NET
- **PHP :**

Le framework .NET :

Ce framework s'appuie sur la norme Common langage infrastructure (CLI). Il a pour but de faciliter la tâche des développeurs en proposant une approche unifiée à la conception d'applications Windows ou Web, tout en introduisant des facilités pour le développement, le déploiement et la maintenance d'applications.

Prado :

Actif depuis 2004, Prado est basé sur des composants, en fait des objets (plus de 700), dirigés par les évènements, comme JavaScript. Il utilise le modèle MVC. Il comporte aussi des widgets HTML.

Django :

C'est un framework de développement web en Python. Il a pour but de rendre le développement web simple et rapide. Ses vues génériques permettent de traiter les cas les plus courants de manière simple, son système d'authentification efficace, sa gestion des exceptions ainsi que la disponibilité de sa documentation sont les points fort de ce framework, toutefois, il n'aide pas à l'intégration d'Ajax, ni à la migration lors du changement de modèle.

Zend Framework :

Ce Framework a été créé en 2006 par la société Zend, L'objectif, pour Zend, est de parvenir à créer LE Framework PHP5 de référence tout en recommandant la conception orientée objets en offrant des outils puissants aux développeurs. ZF permet aussi d'utiliser nativement le principe de MVC (Modèle-Vue- Contrôleur). Cependant on reproche à Zend la lenteur de son évolution due au mode drastique qu'il impose à sa contribution par la communauté internationale.

Symfony :
Est un framework d'applications Web élaboré et un ensemble de composants . Il facilite l'utilisation de bases de données. Parmi les utilisateurs les plus notables, Dailymotion démontre que son utilisation est possible pour supporter des charges importantes. Cela contredit les résultats des benchmarks.

Le site donne six raisons pour choisir ce framework: sa popularité auprès des développeurs, sa pérennité assurée par sa large communauté, les nombreux sites qui l'utilisent, la recherche constante de l'innovation, les nombreuses ressources, l'interopérabilité par le respect des standards liés à PHP.

Tigermouse :

Tigermouse est destiné au prototypage, la modélisation, la création d'application Web et la réutilisation de composants. Ce framework ressemble à Java et se base sur la conception modèle/vue.

Laravel :
Il modernise PHP, le rend plus simple d'emploi avec notamment un système de templates. Un autre exemple, Composer permet de charger uniquement les composants nécessaires à l'application, ce qui rapproche l'environnement de Node.
En combinaison avec la version 5.5 qui ajoute les coroutines, il rajeunit le langage.

- **Java :**

Voici les 10 frammework java les plus utilisés :

Struts 2 :

Apache Struts est un framework libre servant au développement d'applications web J2EE. Il utilise et étend l'API Servlet Java afin d'encourager les développeurs à adopter l'architecture Modèle-Vue-Contrôleur. Apache Struts a été créé par Craig McClanahan et donné à la fondation Apache en mai 2000. Struts a fait partie du projet Jakarta de mai 2000 jusqu'en mars 2004.

JSF :

Java Server Faces (abrégé en JSF) est un framework Java, pour le développement d'applications Web.

A l'inverse des autres frameworks MVC traditionnels à base d'actions, JSF est basé sur la notion de composants, comparable à celle de Swing ou SWT, où l'état d'un composant est enregistré lors du rendu de la page, pour être ensuite restauré au retour de la requête.

JSF est agnostique à la technologie de présentation. Il utilise JSP par défaut, mais peut être utilisé avec d'autres technologies, comme par exemple Facelets ou XUL.

Spring MVC :

Spring est un framework open source pour les applications 3-tiers, dont il facilite le développement et les tests. En 2004, Rod Johnson a écrit le livre Expert One-on-One J2EE Design and Development qui explique les raisons de la création de Spring.

Stripes :

Stripes est un framework de présentation pour construire des applications web en utilisant les technologies Java les plus récentes.

Tapestry :

Tapestry est un framework libre facilitant la construction d'applications web Java basées sur J2EE. Initialement créé par Howard Lewis Ship, le projet Tapestry a été intégré par la fondation Apache comme sous-projet Jakarta puis il a évolué pour devenir un projet Apache à part entière.

GWT :

Google Web Toolkit (GWT) est un ensemble d'outils logiciels développé par Google, permettant de créer et maintenir des applications web dynamiques mettant en oeuvre JavaScript, en utilisant le langage et les outils Java. C'est un logiciel 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.

VI- Etude comparative des framework Spring, Hibernate et Struts :

Comme nous avons opté pour le langage java nous avons choisis d'étudier 3 parmi les framework les plus utilisés : Spring, Hibernate et Struts. Nous avons donc définit chacun de ses framework, décrit leur architecture et nous avons cité également les avantages et les inconvénients de chacun d'eux. Comme nous avons aussi essayé d'implémenter ces trois framework dans une simple application qui fait l'ajout et la suppression d'un employé. Pour ce faire nous avons implémenté au premier lieu hibernate et spring et nous avons ajouté par la suite le framework struts.

Les trois frameworks sont décrits en dessous :

1- Le Framework Hibernate :

1-1- Définition : [9]

Hibernate est une solution open source de type ORM (Object Relational Mapping) qui permet de faciliter le développement de la couche persistance d'une application. Hibernate permet donc de représenter une base de données en objets Java et vice versa.

Hibernate facilite la persistance et la recherche de données dans une base de données en réalisant lui-même la création des objets et les traitements de remplissage de ceux-ci en accédant à la base de données. La quantité de code ainsi épargnée est très importante d'autant que ce code est généralement fastidieux et redondant.

Hibernate est très populaire notamment à cause de ses bonnes performances et de son ouverture à de nombreuses bases de données.

Les bases de données supportées sont les principales du marché : DB2, Oracle, MySQL, PostgreSQL, Sybase, SQL Server, Sap DB, Interbase, ...

Le site officiel http://www.hibernate.org contient beaucoup d'informations sur l'outil et propose de le télécharger ainsi que sa documentation.

La version utilisée dans cette section est la 2.1.2 : il faut donc télécharger le fichier hibernate-2.1.2.zip et le décompresser dans un répertoire du système

1-2- Architecture : [9]

Hibernate a besoin de plusieurs éléments pour fonctionner :

- une classe de type javabean qui encapsule les données d'une occurrence d'une table
- un fichier de configuration qui assure la correspondance entre la classe et la table (mapping)
- des propriétés de configuration notamment des informations concernant la connexion à la base de données

Une fois ces éléments correctement définis, il est possible d'utiliser Hibernate dans le code des traitements à réaliser. L'architecture d'Hibernate est donc la suivante :

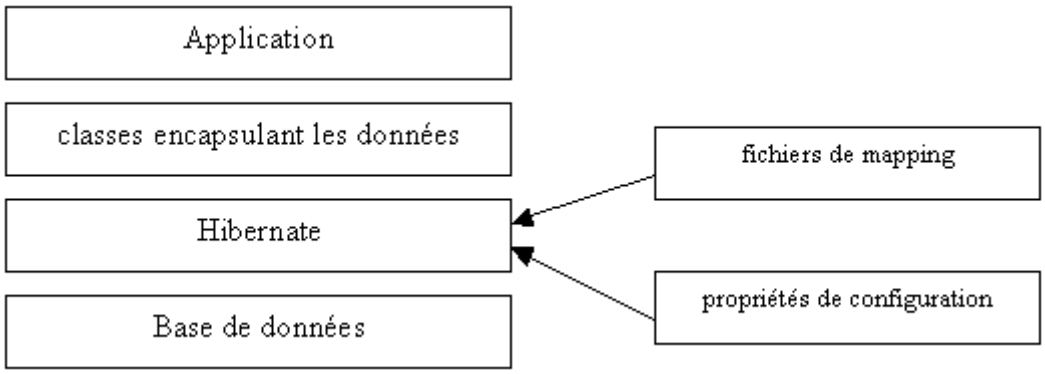


Figure II.8 : architecture d'Hibernate

1-3- Avantages et inconvénients : [10]

Les avantages :

- Hibernate génère le code SQL nécessaire, ce qui rend l'application plus portable (s'adapte à la base de données).
- La persistance est transparente. Vous pouvez faire de vos classes métiers des classes persistantes sans ajout de code.
- La récupération de données est optimisée. On peut interroger la base de données de plusieurs façons (requête SQL, langage HQL…)
- Portabilité du code en cas de changement de base de données.
- Propose une API performante et robuste
- Est mature et dispose d'un support fiable et d'une documentation abondante
- Supporte la gestion des transactions
- S'intègre facilement à Eclipse et Spring
- Appropriation rapide

Les inconvénients :

- Il est dure de faire des requête complexe avec HQL (ex : sous requêtes).
- Etant une technologie jeune, il reste des problèmes à résoudre (ex : les fichiers
- de mapping ne sont pas toujours bien formés)
- Ne se base pas sur les standards

- Ne se base pas sur les spécifications et standards JDO ou ODMG
- Nécessite un module pour la connectivité aux bases Oracle
- Possibilité de perte de vitesse après l'acceptation des spécifications JDO 2.0.

2- Le Framework Spring : [9]

2-1-définition :

Spring est un socle pour le développement d'applications, principalement d'entreprises mais pas obligatoirement. Il fournit de nombreuses fonctionnalités parfois redondantes ou qui peuvent être configurées ou utilisées de plusieurs manières.

Spring est ainsi un des frameworks les plus répandus dans le monde Java : sa popularité a grandie au profit de la complexité de Java EE notamment pour ses versions antérieures à la version 5 mais aussi grâce à la qualité et la richesse des fonctionnalités qu'il propose :

- son coeur reposant sur un conteneur de type IoC assure la gestion du cycle de vies des beans et l'injection des dépendances
- l'utilisation de l'AOP
- des projets pour faciliter l'intégration avec de nombreux projets open source ou API de Java EE

Spring était un framework applicatif à ses débuts mais maintenant c'est une véritable plate-forme composée du framework Spring, de projets qui couvrent de nombreux besoins et de middlewares.

Spring permet une grande flexibilité dans les fonctionnalités et les projets utilisés dans une application. Il est par exemple possible d'utiliser le conteneur Spring pour gérer de façon basique les beans sans utiliser l'AOP. Par contre, certains projets et certaines fonctionnalités ont des dépendances avec d'autres projets.

Spring est associé à la notion de conteneur léger (lightweight container) par opposition aux conteneurs lourds que sont les serveurs d'applications Java EE.

Le site officiel du framework Spring est à l'url http://www.springframework.org.

2-2- Architecture de Spring :

Spring Framework contient toutes les fonctionnalités de base pour développer des applications.

Le coeur de Spring Framework 3.0 est composé d'un ensemble d'une vingtaine de modules qui sont regroupés en plusieurs parties :

- Spring Core Container : regroupe les modules de base pour mettre en oeuvre le conteneur
- AOP and Instrumentation : permet de mettre en oeuvre l'AOP
- Data Acces/Integration : regroupe les modules d'accès aux données
- Web : regroupe les modules pour le développement d'applications web
- Test : propose des fonctionnalités pour les tests automatisés avec Spring

La partie Spring Core Container contient plusieurs modules :

- Spring Core et Spring Beans : contiennent les fonctionnalités de base notamment le conteneur et des utilitaires

- Spring Context : propose un support de la définition du context Spring (sa configuration) mais aussi des fonctionnalités de base comme le mail, l'internationalisation, JNDI, ...
- Spring Expression Langage (SpEL) : propose un langage d'expressions pour interroger et manipuler les objets gérés par le conteneur

La partie AOP and Instrumentation contient plusieurs parties :

- Spring AOP : propose un support de l'AOP
- AspectJ : propose une intégration d'AspectJ
- Instrumentation : propose une instrumentation des classes et plusieurs implémentations de classloaders utilisés par certains serveurs d'applications

La partie Data Acces/Integration contient plusieurs modules

- Spring JDBC : propose une abstraction de l'utilisation de JDBC avec notamment une hiérarchie d'exceptions dédiées
- Spring ORM : propose un support pour des outils de type ORM (JPA, JDO, Hibernate, iBatis)
- Spring Transaction : propose un support déclaratif et par programmation de la gestion des transactions
- Spring OXM : propose une abstraction pour le mapping objet/XML avec un support de JAXB, Castor, XMLBeans, JiBX et XStream
- Spring JMS : propose des fonctionnalités pour faciliter la mise en oeuvre de JMS avec Spring

La partie Web contient plusieurs modules :

- Spring Web : propose des fonctionnalités de base pour les développements web (initialisation du conteneur, gestion des contextes, support multipart, extraction des paramètres d'une requête http, ...)
- Spring Web-Servlet : framework pour le développement d'applications qui met en oeuvre le motif de conception MVC. Ceci permet entre autres de choisir la technologie utilisée pour la vue (JSP, Velocity, Tiles, iText, ...)
- Spring Web-Struts : propose un support de Struts
- Spring Web-Portlet : propose un support pour les portlets

La partie Test contient un seul module :

- Spring Test : propose un support pour les tests automatisés avec un support de JUnit et TestNG

Ces modules sont utilisés comme base pour le développement d'applications.

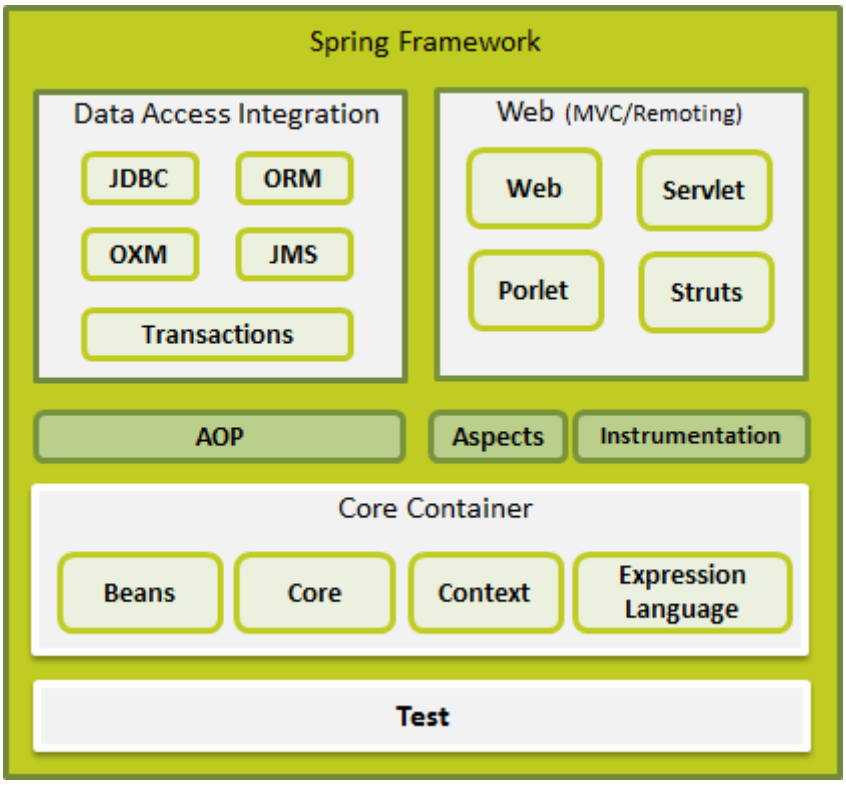


Figure II.9 : architecture de Spring [21]

2-3- Avantages et inconvénients de Spring : [9]

- Spring est un framework open source majoritairement développé par SpringSource mais il n'est pas standardisé par le JCP.

- Il est très largement utilisé dans le monde Java, ce qui en fait un standard de facto et constitue une certaine garantie sur la pérennité du framework.
- Spring propose une très bonne intégration avec des frameworks open source (Struts, Hibernate, ...) ou des standards de Java (Servlets, JMS, JDO, ...)
- Toutes les fonctionnalités de Spring peuvent s'utiliser dans un serveur Java EE et pour la plupart dans un simple conteneur web ou une application standalone.
- Les fonctionnalités offertes par Spring sont très nombreuses et les sujets couverts ne cessent d'augmenter au fur et mesure des nouvelles versions et des nouveaux projets ajoutés au portfolio.
- La documentation de Spring est complète et régulièrement mise à jour lors de la diffusion de chaque nouvelle version.
- La mise en oeuvre de Spring n'est pas toujours aisée car il existe généralement plusieurs solutions pour mettre en oeuvre une fonctionnalité : par exemple, généralement avec Spring 3.0, une fonctionnalité est utilisable par configuration XML, par annotations ou par API. Bien sûr cela permet de choisir mais cela impose un arbitrage selon ses besoins.
- Il n'est pas rare que les livrables aient une taille importante du fait des nombreuses librairies requises par Spring et ses dépendances.

3- Le Framework Struts:[9]

3-1- Définition:

Struts est un framework pour applications web développé par le projet Jakarta de la fondation Apache. C'est le plus populaire des frameworks pour le développement d'applications web avec Java.

Il a été initialement développé par Craig Mc Clanahan qui l'a donné au projet Jakarta d'Apache en mai 2000. Depuis, Struts a connu un succès grandissant auprès de la communauté du libre et des développeurs à tel point qu'il sert de base à de nombreux autres framework open source et commerciaux et que la plupart des grands IDE propriétaires (Borland, IBM, BEA, ...) intègrent une partie dédiée à son utilisation.

Struts met en oeuvre le modèle MVC 2 basé sur une seule servlet faisant office de contrôleur et des JSP pour l'IHM. L'application de ce modèle permet une séparation en trois parties distinctes de l'interface, des traitements et des données de l'application.

Struts se concentre sur la vue et le contrôleur. L'implémentation du modèle est laissée libre aux développeurs : ils ont le choix d'utiliser des JavaBeans, un outil de mapping objet/relationnel, des EJB ou toute autre solution.

Pour le contrôleur, Struts propose une unique servlet par application qui lit la configuration de l'application dans un fichier au format XML. Cette servlet de type ActionServlet reçoit toutes les requêtes de l'utilisateur concernant l'application. En fonction du paramétrage, elle instancie un objet de type Action qui contient les traitements et renvoie une valeur particulière à la servlet. Celle-ci permet de déterminer la JSP qui affichera le résultat des traitements à l'utilisateur.

3-2- Architecture:

Dans cette section, nous allons vous expliquer l'architecture de Struts. Struts est connu pour son architecture robuste et est utilisé pour développer des projets de petites et grandes tailles.

Struts est un projet opensource utilisé pour développer des applications web JEE en utilisant le modèle d'architecture MVC. Struts utilise l'API JEE qu'il étend pour permettre aux développeurs d'adopter une architecture MVC. Le framework Struts comporte trois composantes :

Le gestionnaire de requêtes fourni par le développeur de l'application qui permet de lier un comportement à une URL

Le gestionnaire de réponses qui permet de transférer le contrôle vers une autre ressource qui va s'occuper du rendu de la réponse (vers le client)

Une librairie de « tags » permettant au développeur de créer des formulaires interactifs avec des pages web.

Struts fournit toute l'architecture pour implémenter un MVC dans vos applications et ainsi permettre aux développeurs de se concentrer sur la partie « métier ».

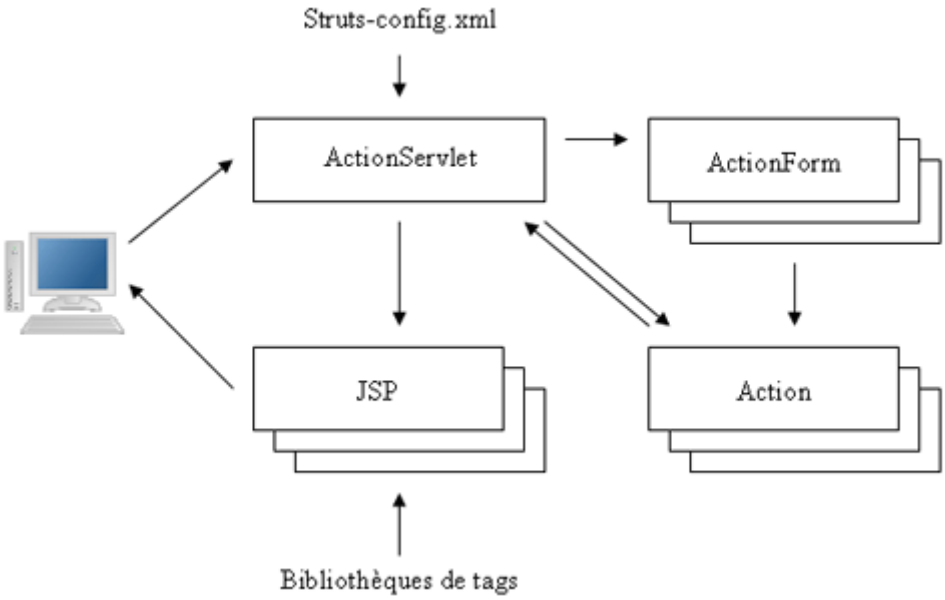


Figure II.10: architecture de struts

Les données issues de la requête sont encapsulées dans un objet de type ActionForm. Struts va utiliser l'introspection pour initialiser les champs de cet objet à partir des valeurs fournies dans la requête.

Struts utilise un fichier de configuration au format XML (struts-config.xml) pour connaître le détail des éléments qu'il va gérer dans l'application et comment ils vont interagir lors des traitements.

Pour la vue, Struts utilise par défaut des JSP avec un ensemble de plusieurs bibliothèques de tags personnalisés pour faciliter leur développement.

Struts propose aussi plusieurs services techniques : pool de connexions aux sources de données, internationalisation, ...

La dernière version ainsi que toutes les informations utiles peuvent être obtenues sur le site http://struts.apache.org/.

Il existe plusieurs versions de Struts : 1.0 (publiée en juin 2001), 1.1 et 1.2

3-3- Avantages et inconvénients: [9]

Avantage :

- Est quasiment un standard de fait
- Nombreuses sources d'informations
- Bibliothèques de tags HTML

- Intégration dans les IDE
- open source

Inconvénients :

- Son avenir
- Peu d'évolutions
- une architecture vieillissante
- beaucoup de classes et de code à produire (partiellement automatisable avec des outils dédiés)

On termine avec un tableau comparatif où nous avons fait une comparaison entre les trois selon certains critères tel que la date de création de chacun, la maturité, l'interprétation java…

VII- Le tableau comparatif :

Nous allons maintenant comparer les trois frameworks que nous avons présentés au cours des articles précédents, sous la forme d'un tableau, sur ces différents critères :

- Date de création
- Maturité
- Open Source
- MVC
- Composants ou Requête
- Interprétation directe du Java
- Templates
- Compilateur Java vers Javascript
- Facilité de prise en main / Légèreté / Simplicité

	Date de création	maturité	Open source	mvc	Composant ou requête	servelet	Interprétation direct du java	template	Compilateur java vers java script	Facilité de prise en main/légèreté/facilité
hibernate		oui	oui	mvc	requête	oui	non	Page Xhtml	Oui (javascripthibernate)	oui
spring	2000	oui	oui	Mvc2	requête	Utilisé/ servelet centrale/compétence nécessaires	non	Page jsp/velocity, freemaker	non	Non/technologie des servelet à maitriser
struts		oui	oui	Mvc2	requête	Utilisé/servelet centrale/compétence nécessaires	non	Page jsp	non	Non/technologie des servelet à maitriser

Tableau1 : comparatif des framework spring, hibernate, struts

Après cette étude comparative nous avons constaté que struts est un framework de présentation qui peut être utilisé indépendamment des autres. Hibernate est un framework de persistance qui sert à maintenir une bonne persistance entre les objets java et la base de données relationnelle. Spring quand à lui peut servir à relier ces deux derniers pour assurer la gestion déclarative des transactions. Nous avons constaté également qu'il n'y a pas de meilleur framework mais que le choix du framework à utiliser dépend plutôt des besoins, des fonctionnalités et des ressources de l'application à réaliser. Comme notre application dépend essentiellement de la base de données et d'une bonne gestion de cette dernière et comme travailler dans les deux univers que sont l'orienté objet et la base de données relationnelle peut être lourd et consommateur en temps nous avons opté pour un outil de mapping objet/relationnel que le monde de java a mise en place : Hibernate. Le terme mapping objet/relationnel (ORM) décrit la technique consistant à faire le lien entre la représentation objet des données et sa représentation relationnelle basé sur un schéma SQL.

VIII- Conclusion :

Un framework est un outil développé dans un langage spécifique permettant au développeur de ne pas à chaque nouveau projet réinventer la roue comme l'accès aux données, le logger, etc. et de développer selon des normes ce qui permet de réutiliser certains morceaux de code. Le but étant de gagner en rentabilité et de ne pas s'attarder sur les tâches rébarbatives pour se concentrer sur les tâches à forte valeur ajoutée.

Les frameworks sont donc répandus dans le monde de l'informatique, et plus particulièrement dans le monde du web les frameworks sont utilisés afin de ne pas implémenter à chaque fois certains points complexes et/ou répétitifs comme les versions de HTML/CSS/Javascript à utiliser, la gestion des différents navigateurs, etc. et les framework connaissent une évolution rapide en s'adaptant toujours aux besoins des développeurs.

Dans ce chapitre nous avons donc abordé quelques notions de base concernant les frameworks et nous avons aussi comparé les différents types existants pour finir de sélectionner le meilleur pour notre application et qui sera décrit d'une manière détaillé dans le prochain chapitre.

Chapitre III:

Analyse et conception

I- Introduction :

Apres avoir vu, dans les chapitres précédents les différents concepts nécessaires a l'accomplissement de notre travail, nous passons maintenant a la partie analyse et conception. Dans tous système d'informatique, la conception est importante et doit être traité avec précision et en détail, précédée d'une analyse profonde et réfléchie, car elle est le reflet du futur système avant même sa concrétisation. Des progrès énormes ont été consentis dans le but d'avoir une meilleur analyse et de rendre la conception plus complète. L'approche objet, s'est avérée un modèle d'analyse et de conception très puissant et se trouve de plus en plus utilisée. Pour cela nous avons adopté la conception avec UML (Unified Modling Language) qui permet de bien représenter la dynamique d'une application par la série des diagrammes qu'il offre.

II- Analyse :

L'analyse comprend les activités qui permettent d'aboutir au modèle d'analyse du système en partant des cas d'utilisation et des besoins fonctionnels. Ces activités ne font aucune supposition sur la manière d'implémenter la solution finale. Pour l'analyse d'un système, on fait appel aux cas d'utilisation, aux diagrammes de classe détaillés et de séquences.

1- Présentation d'UML :

1-1- Définition :

UML est un langage pour documenter et spécifier graphiquement tous les aspects d'un système logiciel, il est organisé autour de notations, schéma, diagrammes et autres symboles qui permettent une certaine abstraction du système à modéliser en offrant des vues du futur système.

1-2- Modélisation avec UML *:*

UML permet de représenter des modèles, mais il ne définit pas d'élaboration de modèles.

Les autres auteures d'UML conseillent tout de même une démarche pour favoriser la réussite d'un projet, cette démarche doit être :

a) Une démarche itérative et incrémentale : Pour comprendre et représenter un système complexe, pour analyser les étapes, pour favoriser le prototype et pour réduire et maîtriser l'inconnu.

b) Une démarche guidée par les besoin des utilisateurs : tout est basé sur le besoin des utilisateurs du système, le but du développement lui-même est de répondre a leurs besoins .chaque étape sera affinée et validée en fonction des besoins des utilisateurs.

c) Une démarche centrée sur l'architecture logicielle : C'est la clé de voute du succès d'un développement, les choix stratégiques définiront la qualité du logiciel.

2- Objectifs de notre application :

Le but de notre application est de permettre aux agents de l'entreprises de gérer les listes des clients déjà inscrits et leur contrats et de permettre à l'administrateur de gérer ces agents et leurs profiles correspondants.

3- Identification des acteurs :

Un acteur : représente un rôle que peut jouer l'utilisateur dans le système. L'acteur est associé à un cas d'utilisation, c'est-à-dire qu'il peut interagir avec lui et participer à son scénario, il est représenté par un personnage stylisé.

Les acteurs de notre système sont :

Agent : la personne qui s'occupe de la gestion des clients.

Administrateur : personne se chargeant de la gestion des agents

4- Identification des cas d'utilisation :

Un cas d'utilisations :

Représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable par un acteur.

Les acteurs définis précédemment effectuent un certain nombre de taches, ces taches sont résumées dans le tableau ci-dessous.

acteur	**Tâches**
agent	T1: s'authentifier T2: gestion des clients (créer, modifier, consulter) T3 : gestion contrat (modifier, consulter, créer)
Administrateur	T4: idem que l'agent T5: gérer les profils des agents (créer, consulter, modifier) T6: gestion des agents (créer, consulter, modifier)

5- Spécification des scénarios :

acteurs	**tâches**	**Scénarios**
Agent	T1 :s'authentifier	S1 : saisir login agent S2 : saisir mot de passe
	T2 : gestion des clients	S3 : consulter liste client S4 : ajouter client S5 : consulter, modifier client S6 : se déconnecter
	T3 : gestion de contrats	T7 : création contrat T8 : modification contrat T9 : accéder a liste contrat T10 : saisir sinistre T11 : se déconnecter
administrateur	T4: Idem que l'agent	S12: même scénarios que précédant
	T5: gestion des agents	S13: consulter liste d'agent. S14: modifier agent S15 : créer agent S16 : se déconnecter
	T6 : gestion profiles	S17 : créer profile S18 : consulter profile S19 : modifier profile S20 : se déconnecter

6- Spécification des cas d'utilisation :

Use case: S'authentifier.
Scenarios: S1, S2
Rôle : agent, Administrateur.
Description : 1. Saisir l'URL agent/Administrateur.
2. Le système affiche la page d'accueil spécifiée.
3. agent/Administrateur saisit son **login** et **mot de passe** puis clique sur le lien **connexion**.
4. Le système affiche l'espace personnel de l'agent/Administrateur.

Use case : « authentification »

Use case: Gestion des clients.
Scenarios: S3, S4, S5, S6
Rôle: agent.
Description : 1. Saisir l'URL agent.
2. Le système affiche l'interface agent.
3. Saisir le login et le mot de passe puis valide.
4. Sélectionner un lien parmi les liens correspondant à la gestion des clients (modifier, consultee, créer)
5. Le système affiche la page correspondante.
6. Effectuer des modifications sur la table « Client » de la base de données Puis valider.
7. Se deconnecter

Use case : « gestion des clients »

.

Use case: Gestion des Agents.
Scenarios: S13, S14, S15,S16
Rôle : Administrateur.
Description : 1. Saisir l'URL Administrateur.
2. Le système affiche l'interface administrateur.
3. Saisir le login et le mot de passe puis valider.
4. Sélectionner un lien parmi les liens correspondant à la gestion des agents (modifier, créer, consulter…).
5. Le système affiche la page correspondante.
6. Effectuer des modifications sur la table « Agent » de la base de données puis Valider
7.se deconnecter

Use case : « gestion des agents »

Use case: gestion profile
Scenarios: S17, S18,S19,S20
Rôle : administrateur
Description : 1. Saisir l'URL Administrateur.
2. Le système affiche l'interface administrateur.
3. Saisir le login et le mot de passe puis valider.
4. Sélectionner un lien parmi les liens correspondant à la gestion des profiles (modifier).
5. Le système affiche la page correspondante.
6. Effectuer des modifications sur la table « profile » de la base de données puis valider
7. se déconnecter

Use case : « gestion profile »

Use case: gestion contrats
Scenarios: S7, S8,S9, S10, S11
Rôle : agent
Description : 1. Saisir l'URL Agent.
2. Le système affiche l'interface agent.
3. Saisir le login et le mot de passe puis valider.
4. Sélectionner un lien parmi les liens correspondant à la gestion des contrats (modifier, consulter, créer…).
5. Le système affiche la page correspondante.
6. Effectuer des modifications sur la table « contrat » de la base de données puis valider
7. se déconnecter

Use case : « géstion contrats »

7- Le diagramme de cas d'utilisation général :

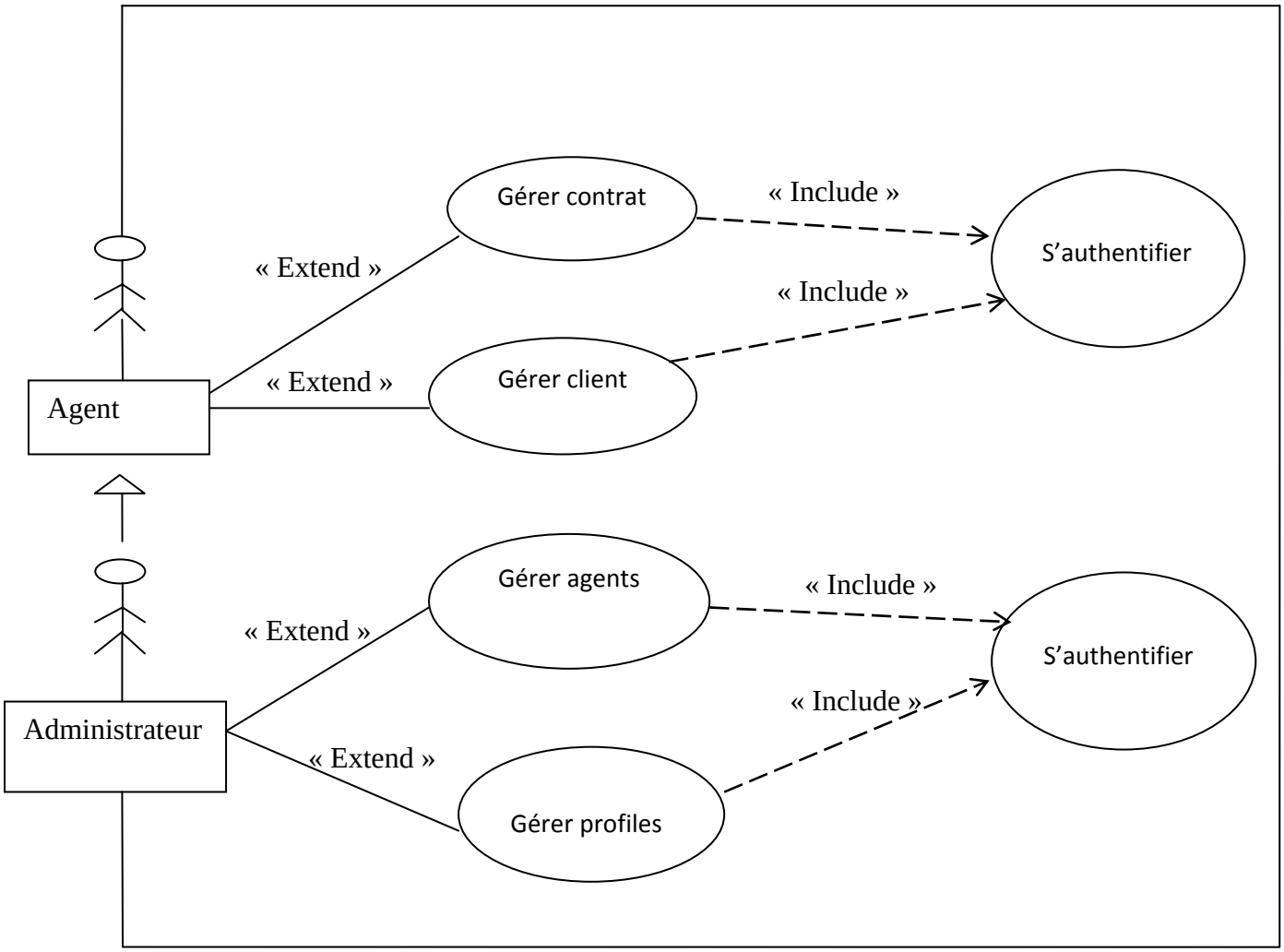


Figure III.1 : diagramme de cas d'utilisation général

III- Conception :

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur : saisir une donnée, consulter une donnée, lancer un traitement ; il met en évidence les objets manipulés ainsi que les opérations qui font passer d'un objet à l'autre. Dans notre cas on s'intéresse seulement à effectuer la représentation du diagramme de séquence pour les cas d'utilisation déjà présentés auparavant.

1- Diagrammes de séquences :

Cas d'utilisation « Authentification » :

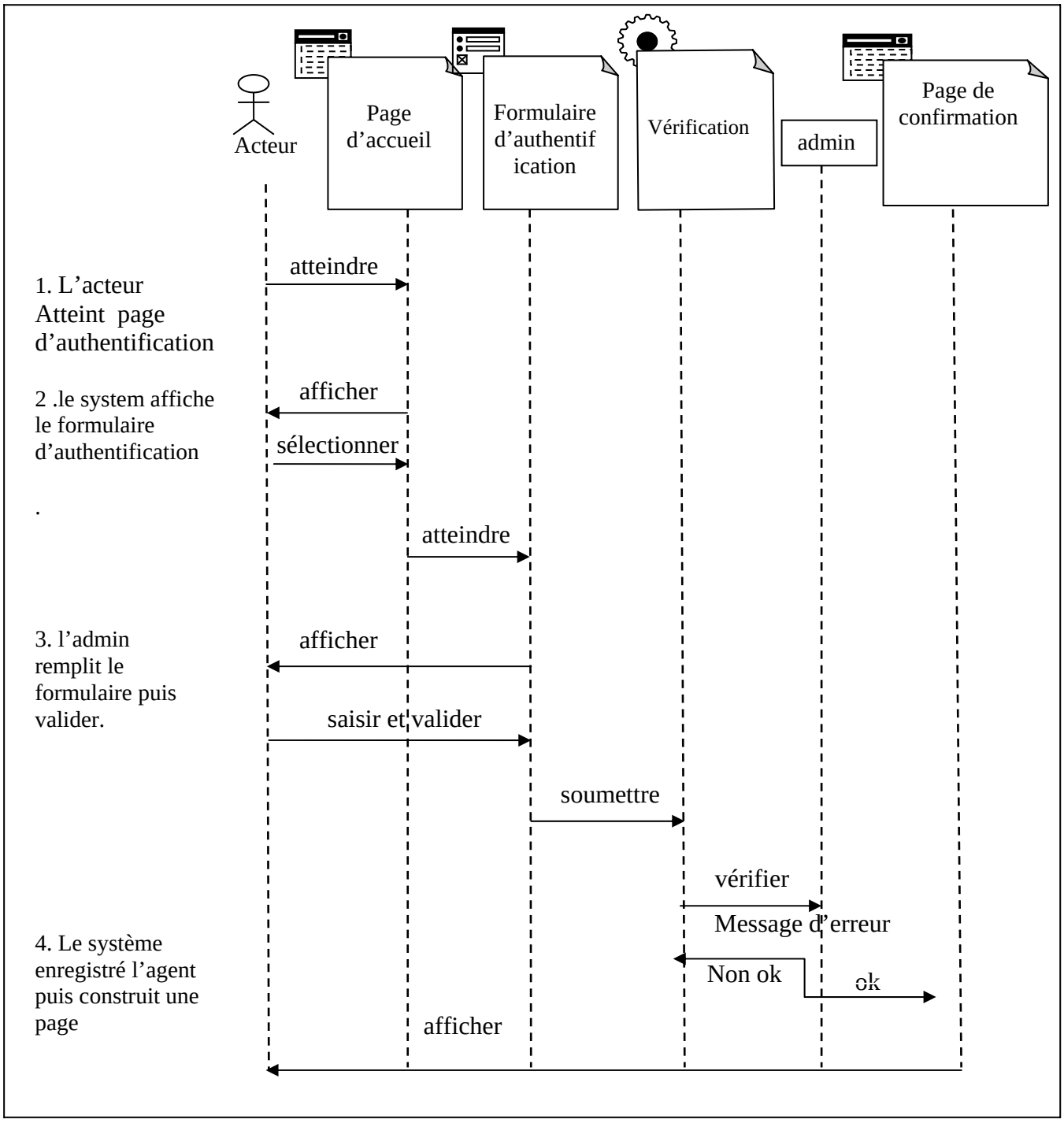


Figure III.2 : Diagramme de séquence de cas d'utilisation «Authentification »

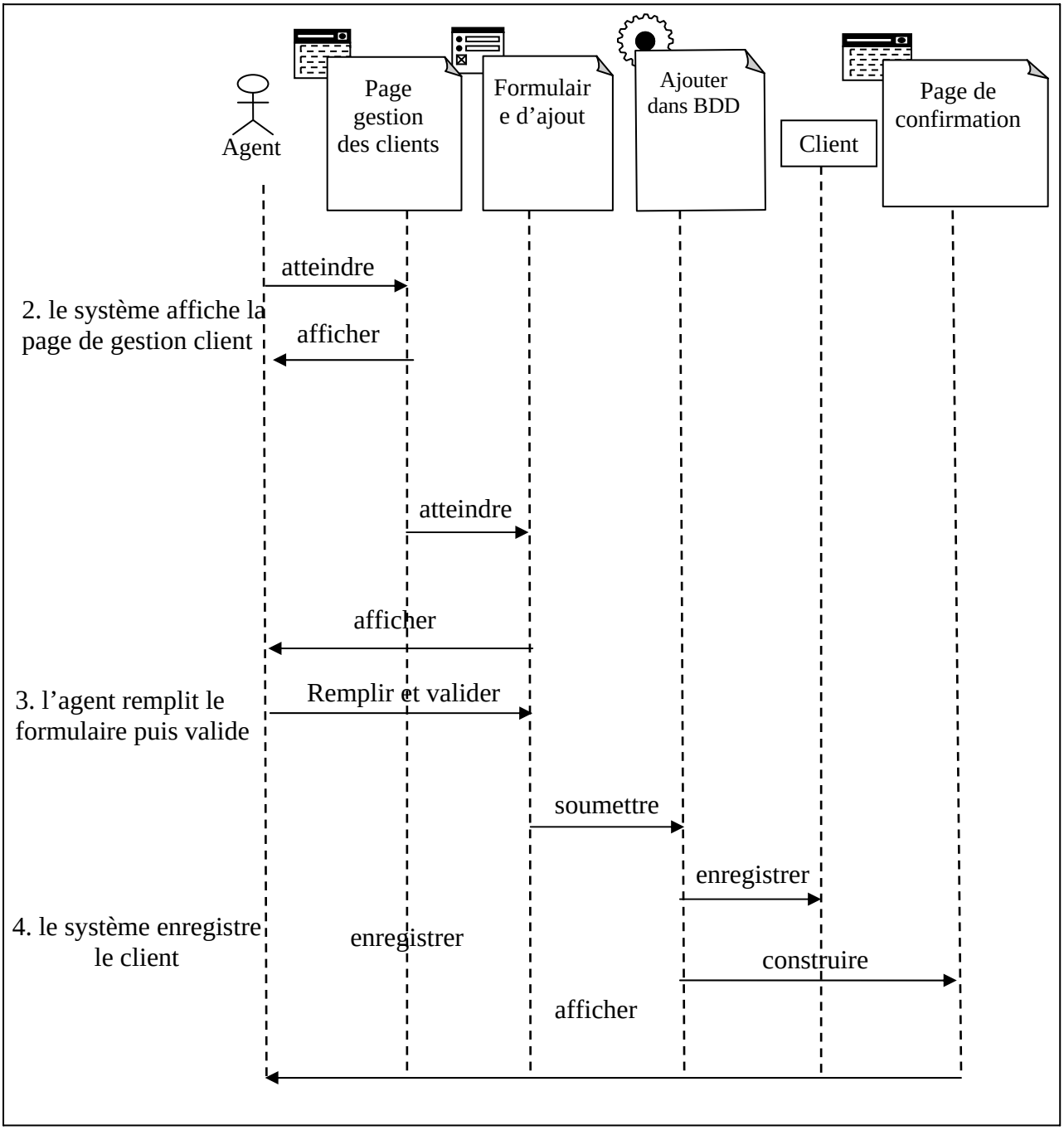


Figure III.3 : Diagramme de séquence « ajouter client »

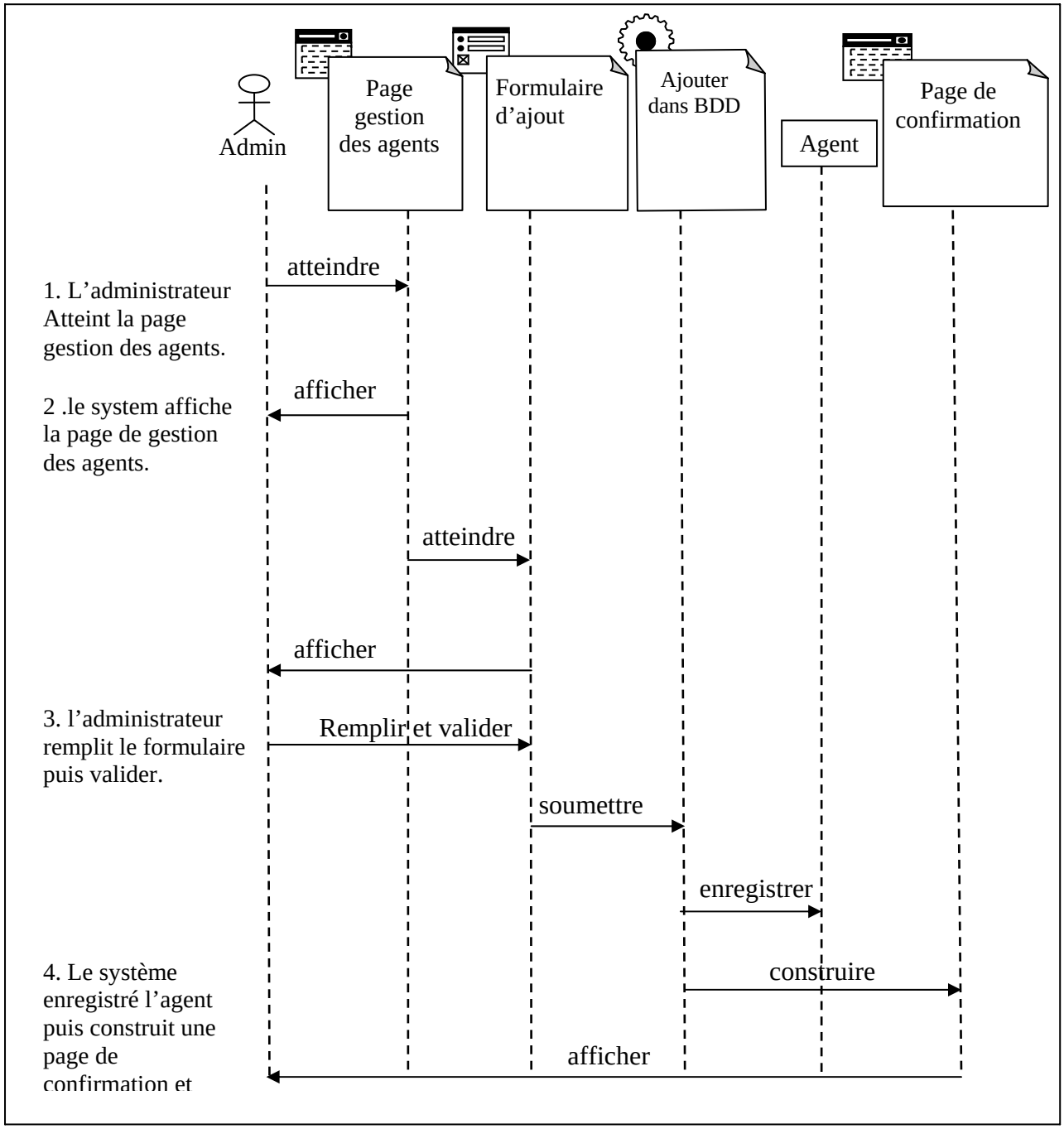


Figure III.4 : Diagramme de séquence du cas d'utilisation «Ajouter un agent».

2- Diagramme de classes :

Le diagramme de classes représente l'architecture conceptuelle du système : il décrit les classes que le système utilise, ainsi que leurs liens, que ceux-ci représentent un emboitage conceptuel (héritage, marqué par une flèche terminée par un triangle) ou une relation organique (agrégation, marqué par une flèche terminée par un diamant).

Cas d'utilisation « authentification » :

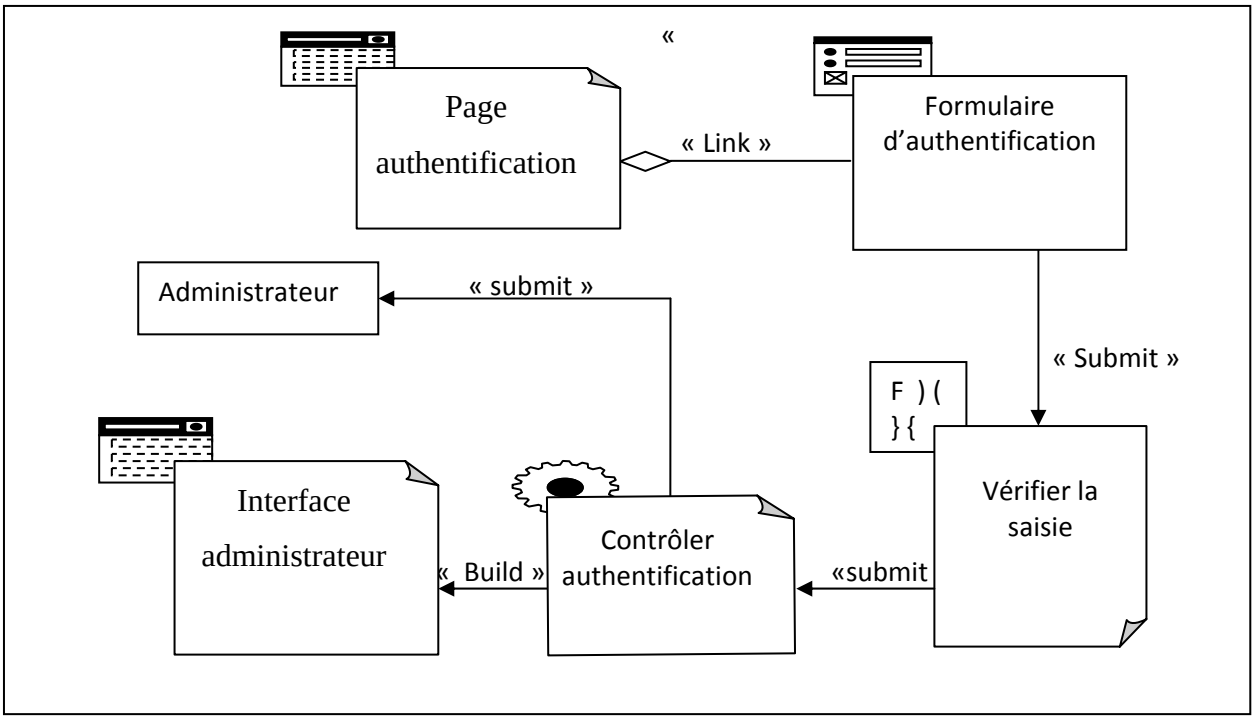


Figure III.4 : Diagramme de classe du cas d'utilisation « authentification »

Cas d'utilisation « ajouter client » :

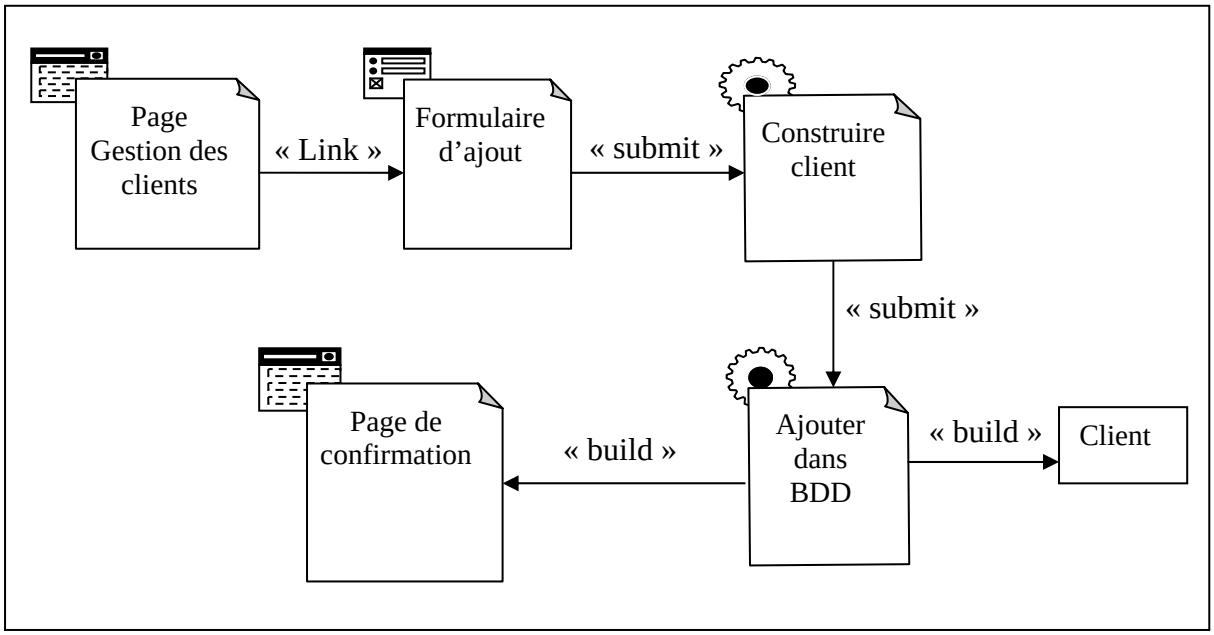


Figure III.5 : Diagramme de classe du cas d'utilisation « ajouter client »

Cas d'utilisation « ajouter agent »

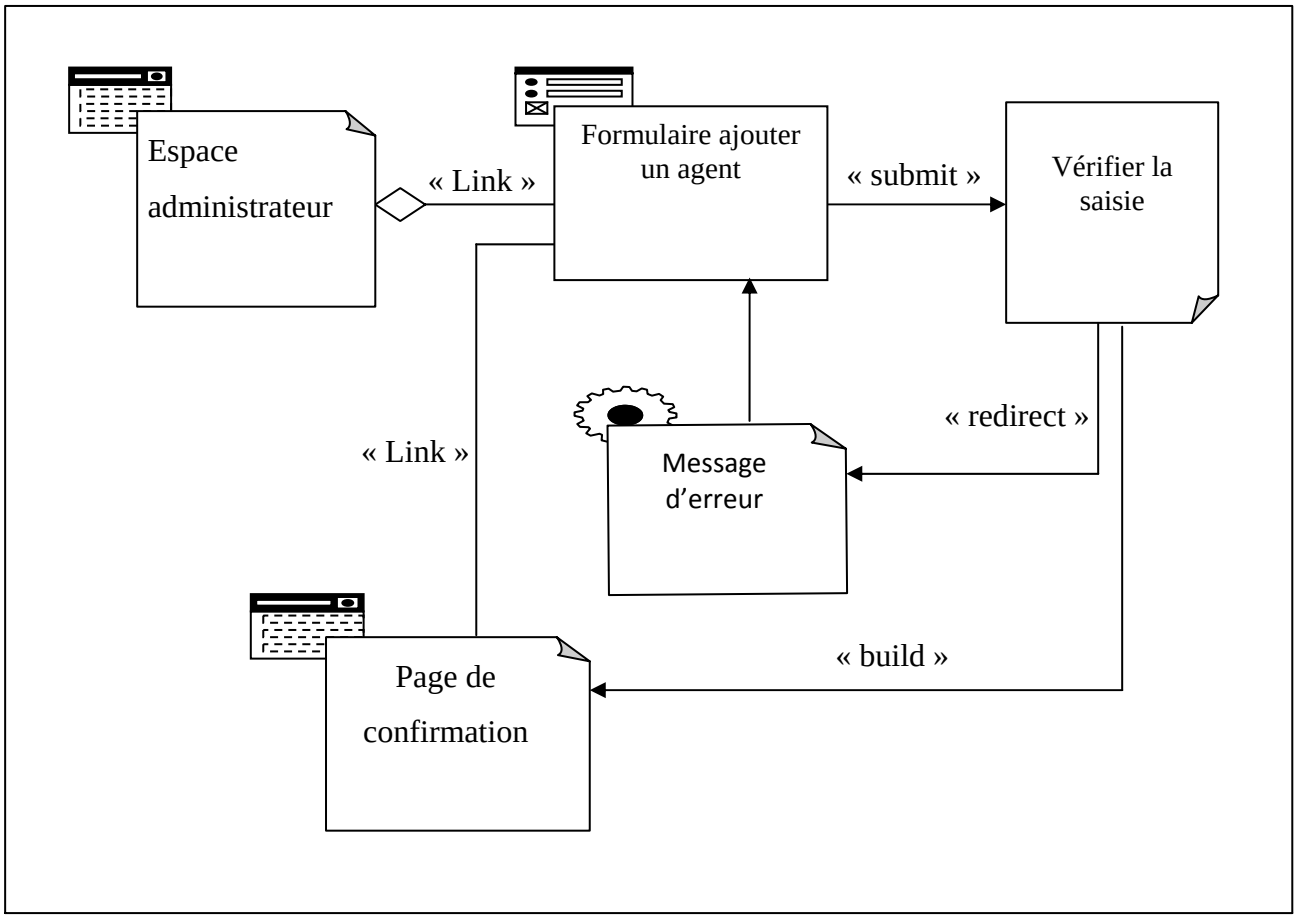


Figure III.6 : Diagramme de classe du cas d'utilisation « ajouter agent »

3- Diagrammes de classes détaillés :

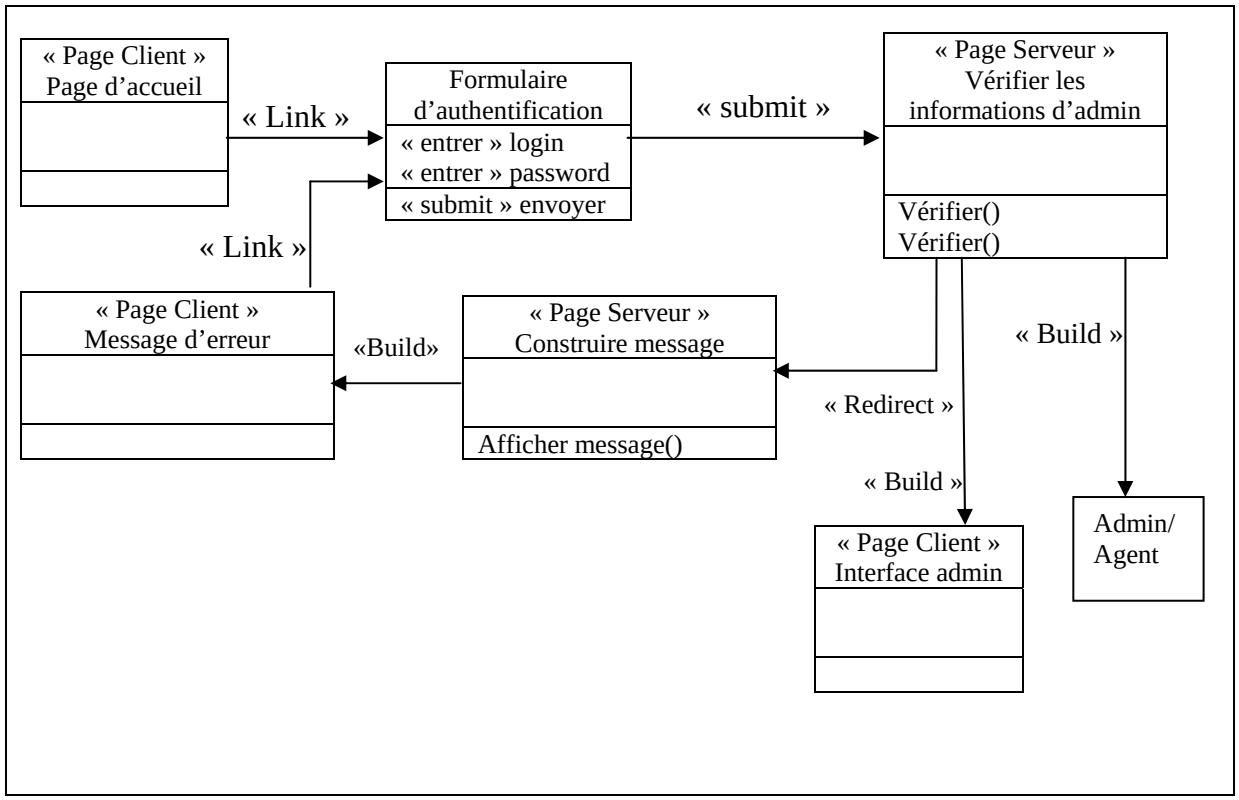


Figure III.7 : diagramme de classe détaillé du cas d'utilisation« authentification »

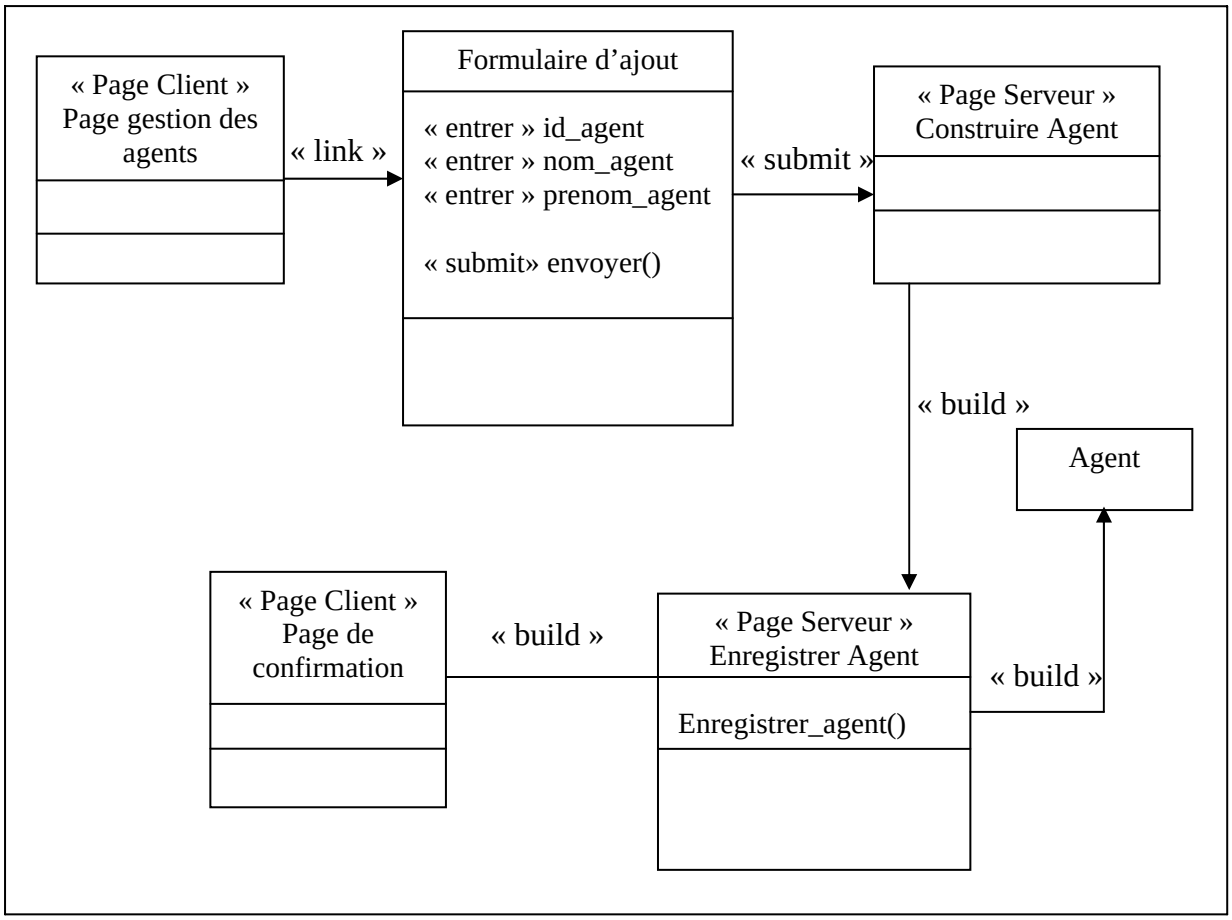


Figure III.8 : diagramme de classe détaillé du cas d'utilisation « ajouter agent »

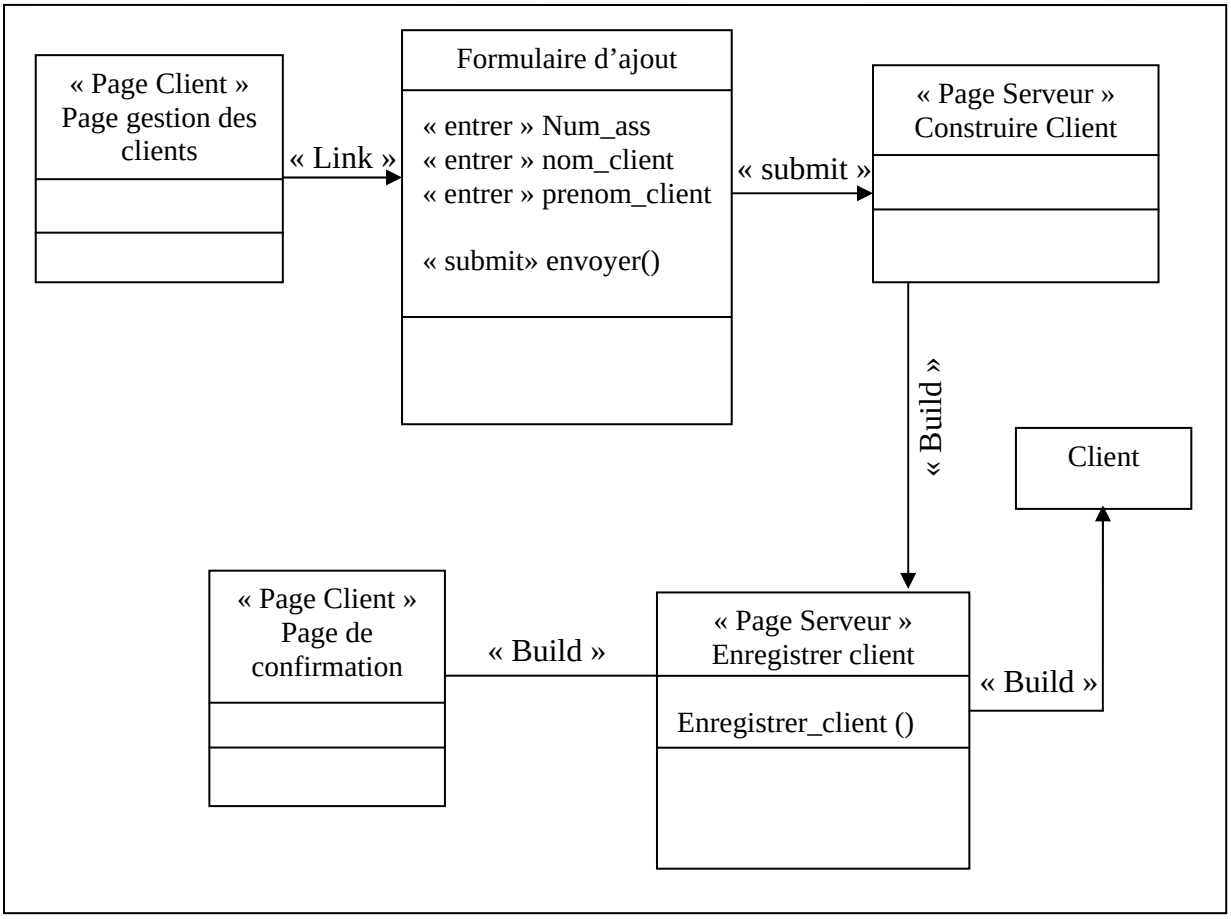
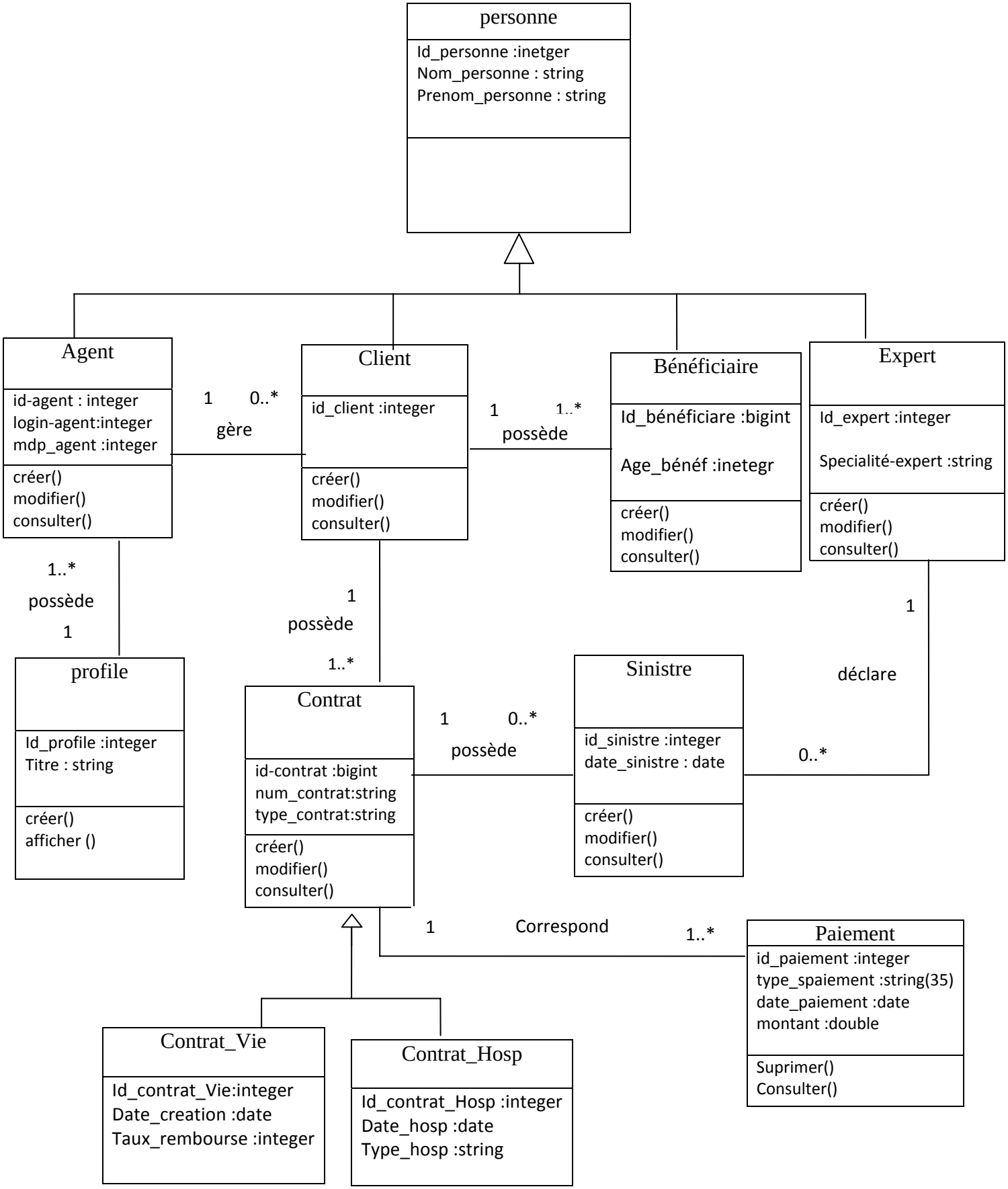


Figure III.9 : diagramme de classe détaillé du cas d'utilisation« ajouter client »

4- Diagramme de classes général :



IV- Conclusion :

Dans ce chapitre, nous avons présenté une démarche de modélisation pour développer notre application, cette démarche est basée sur le langage de modélisation UML pour le Web.

Nous avons commencé par une spécification de cas d'utilisation dans un premier temps, suivie d'une élaboration des diagrammes de séquence ensuite, une élaboration des diagrammes de classes (général et détaillé), et enfin une définition du schéma conceptuel du modèle de données.

Chapitre IV:

Réalisation

I- Introduction :

Après avoir présenté dans le chapitre précédent les différentes étapes d'analyse et de conception, nous allons présenter dans ce dernier chapitre l'environnement de développement, les outils qui ont servi à la réalisation de notre application, et nous terminerons par la présentation de ses fonctionnalités à travers ses différentes interfaces.

Notre démarche s'appuie essentiellement sur l'implémentation de la base de données, avant de présenter les outils utilisés, pour enfin aboutir au fonctionnement de l'application. Pour ce faire, nous avons utilisé un ensemble d'outils à savoir le serveur web Apache, le système de gestion de base de données relationnelle MySQL, les langages java, html, Xhtml.

II- L'environnement de travail :

1- Outils utilisés :

1-1- Matériels

- PC portable.

1-2- Logiciels

- Microsoft Windows 8.
- eclipse
- Easyphp.
- Mysql
- le Serveur Apache.7
- Hibernate Framework 3.
- Primeface/Jsf

III- Environnement de développement:

1- Eclipse : [9]

En matière d'environnement de développement, le marché offre un choix très large.
Pour la réalisation de notre application nous avons utilisé l'environnement de développement Eclipse.

Eclipse est un environnement de développement intégré (integreted development environnement) dont le but est de fournir une plate forme modulaire pour permettre de réaliser des développements informatiques. Eclipse possède de nombreux points forts qui sont à l'origine de son succès dont les principaux :

Eclipse offre un environnement de développement java très complet il fournit toutes les API (Application Programming Interface) de base. Ces API sont structurées en package, et contiennent des classes réutilisables pour construire des programmes plus complexes.

Eclipse utilise énormément le concept de module nommé « plug-in » dans son architecture. D'ailleurs, hormis le noyau de la plate forme « Runtime », tout le reste de la plate forme est développé sous la forme de plug-ins. Ce concept permet de fournir un mécanisme pour l'extension de la plate forme et ainsi fournir la possibilité à des développeurs des fonctionnalités qui ne sont pas fournis en standard par Eclipse.

Le gestionnaire de mise à jour permet de télécharger de nouveaux plug-ins ou nouvelles versions d'un plug-in déjà installées à partir de sites web dédiés.

Cette figure représente l'interface d'eclipse J2EE

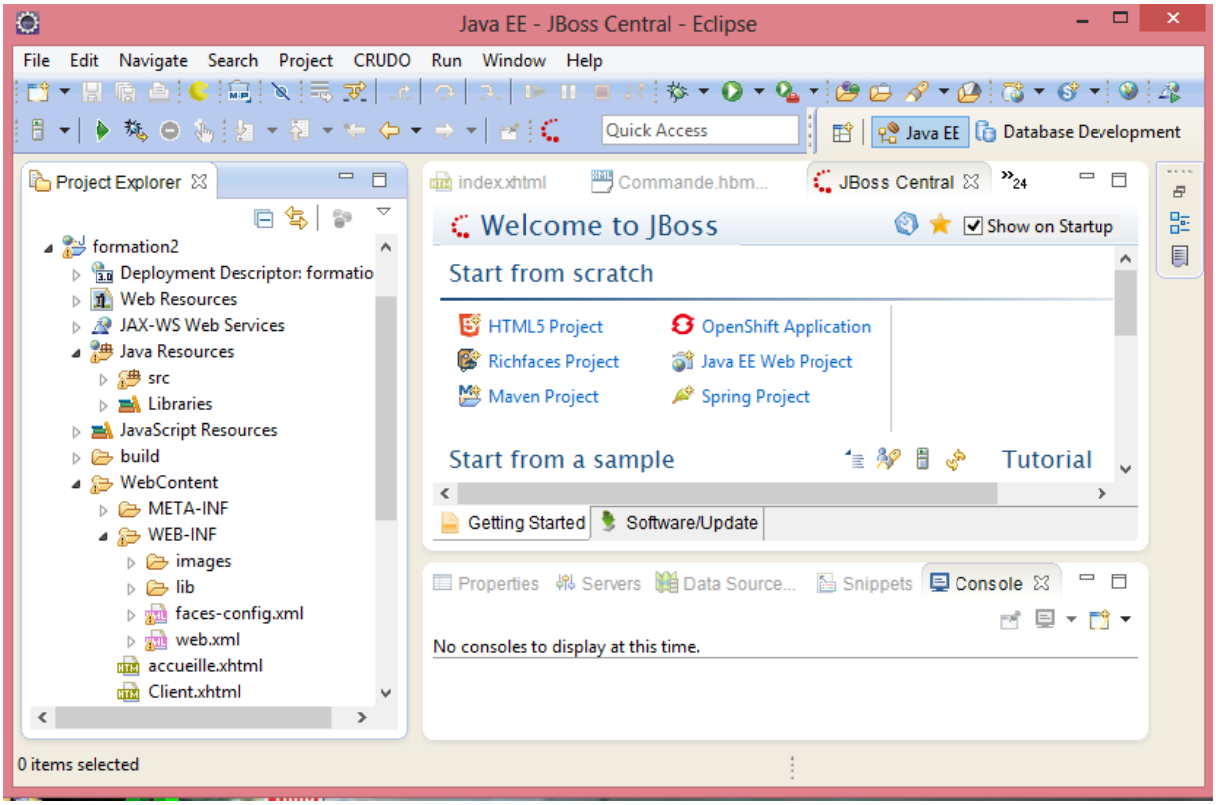


Figure IV.1: Interface eclipse Kepler

2- Platforme J2EE: [9]

J2EE est une plate-forme fortement orientée serveur pour le développement et l'exécution d'applications distribuées. Elle est composée de deux parties essentielles :

- un ensemble de spécifications pour une infrastructure dans laquelle s'exécutent les composants écrits en Java : un tel environnement se nomme serveur d'applications.

- un ensemble d'API qui peuvent être obtenues et utilisées séparément. Pour être utilisées, certaines nécessitent une implémentation de la part d'un fournisseur tiers.

Sun propose une implémentation minimale des spécifications de J2EE : le J2EE SDK. Cette implémentation permet de développer des applications respectant les spécifications mais n'est pas prévue pour être utilisée dans un environnement de production. Ces spécifications doivent être respectées par les outils développés par des éditeurs tiers.

L'utilisation de J2EE pour développer et exécuter une application offre plusieurs avantages :

- une architecture d'applications basée sur les composants qui permet un découpage de l'application et donc une séparation des rôles lors du développement
- la possibilité de s'interfacer avec le système d'information existant grâce à de nombreuses API : JDBC, JNDI, JMS, JCA ...
- la possibilité de choisir les outils de développement et le ou les serveurs d'applications utilisés qu'ils soient commerciaux ou libres

J2EE permet une grande flexibilité dans le choix de l'architecture de l'application en combinant les différents composants. Ce choix dépend des besoins auxquels doit répondre l'application mais aussi des compétences dans les différentes API de J2EE. L'architecture d'une application se découpe idéalement en au moins trois tiers :

- la partie cliente : c'est la partie qui permet le dialogue avec l'utilisateur. Elle peut être composée d'une application standalone, d'une application web ou d'applets
- la partie métier : c'est la partie qui encapsule les traitements (dans des EJB ou des JavaBeans)
- la partie donnée : c'est la partie qui stocke les données.

3- Le framework Hibernate : [19]

Hibernate est un Framework Java de persistance qui permet de faire correspondre des tables de base de données relationnelles avec des objets java simples (POJO ou «Plain Old Java Object»). Une fois la correspondance entre les deux mondes définie, le programme Java peut manipuler toutes les données en utilisant que des JavaBean, masquant alors totalement la base de données sous-jacente et ses spécificités. Le Framework assure le remplissage de ces objets et la mise à jour de la base en se basant sur leur contenu.

Le framework Hibernate permet d'obtenir une représentation XML du résultat d'une requête. Il permet aussi de rendre persistant, c'est-à-dire d'insérer ou de mettre à jour des données dans la base de données depuis des fragments de document XML de façon très simple.

3-1- Mise en œuvre du framework hibernate :

Hibernate a besoin de plusieurs éléments pour fonctionner :

Une classe de type JavaBean qui encapsule les données d'une table donnée nommée « classe de persistance ».

Un fichier de correspondance qui configure la correspondance entre la classe et la table.

Des propriétés de configuration, notamment des informations concernant la connexion à la base de données.

Une fois ces éléments correctement définis, il est possible d'utiliser Hibernate dans le code des traitements à réaliser. L'architecture d'Hibernate est donc la suivante :

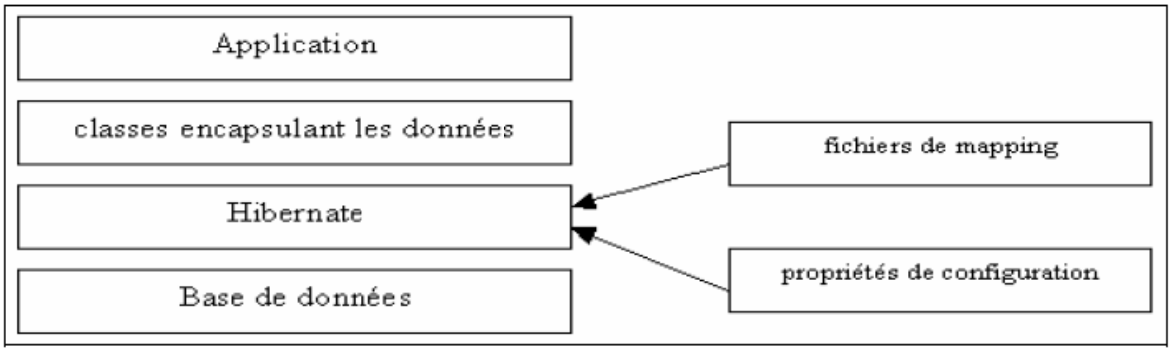


Figure IV.2: architecture d'Hibernate

3-2- Création du projet Java supportant Hibernate sous Eclipse

a- Ajout des Jars :

L'ajout du support de Hibernate revient à rajouter un ensemble de Jar du répertoire lib de Hibernate.

Ces Jar sont les suivants.

- hibernate3.jar
- ant-1.6.5.jar
- antlr-2.7.6.jar
- asm-attrs.jar
- asm.jar
- cglib-2.1.3.jar
- Commons-logging-1.0.4.jar
- blitzer 1.0.10.jar
- commons-collections-2.1.1.jar
- dom4j-1.6.1.jar
- ehcache-1.2.3.jar
- jdbc2_0-stdext.jar
- jta.jar (utile pour le déploiement avec le conteneur Tomcat)
- mysql-connector-java-3.0.14-productionbin.jar (Pilote JDBC de MySQL)

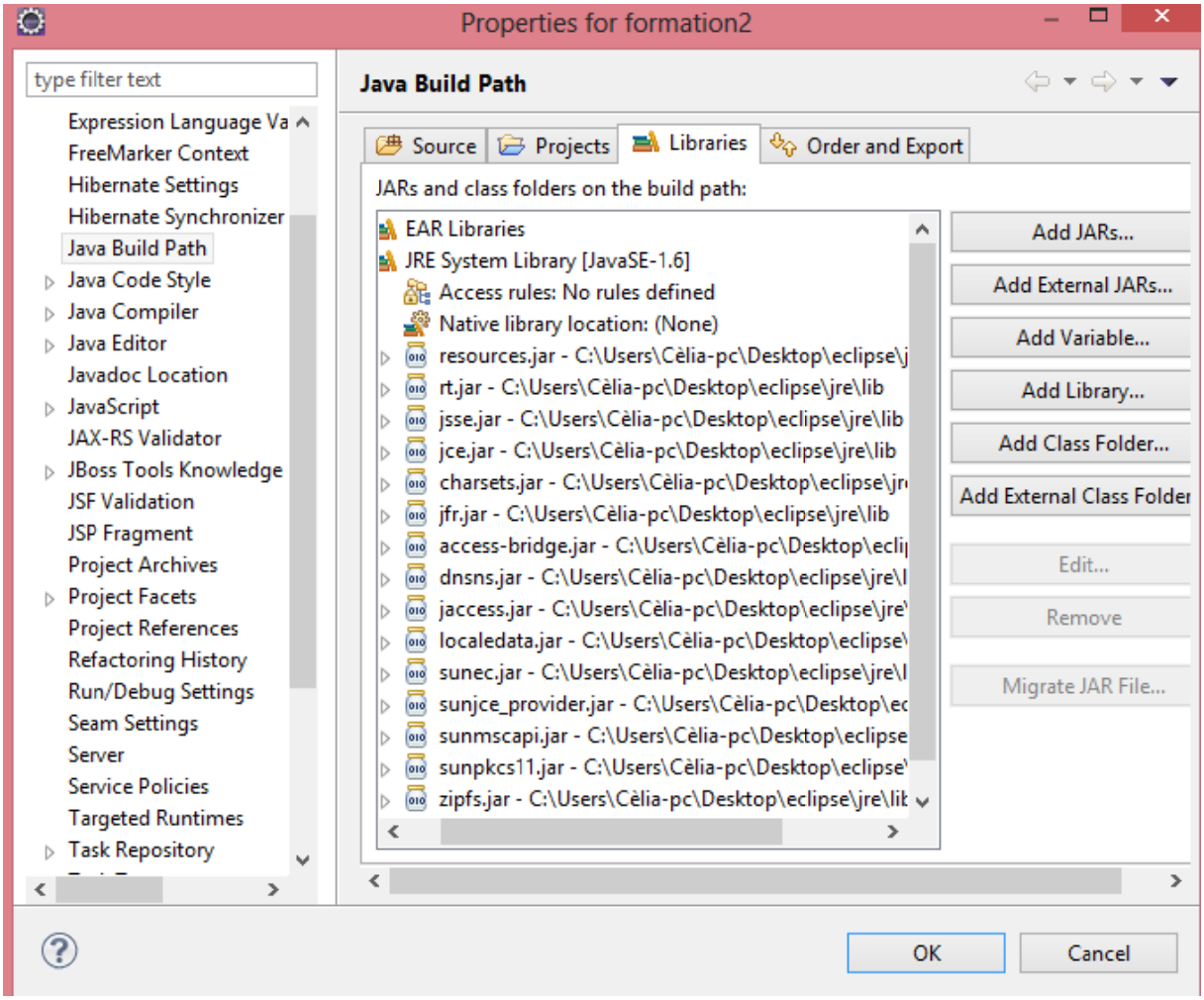


Figure IV.3 : ajout des Jar hibernate pour eclipse

b- Création du fichier de configuration : hibernate.cfg.xml

Hibernate propose des classes qui héritent de la classe Dialect pour chaque base de données supportée. C'est le nom de la classe correspondant à la base de données utilisée qui doit être obligatoirement fourni à la propriété hibernate.dialect.

Les propriétés sont alors définies par un tag <property>. Le nom de la propriété est fourni grâce à l'attribut « name » et sa valeur est fournie dans le corps du tag.

Il est possible de fournir les propriétés de configuration « Hibernate » dans un fichier hibernate.properties.

Le fichier **hibernate.cfg.xml** se présente comme suit :

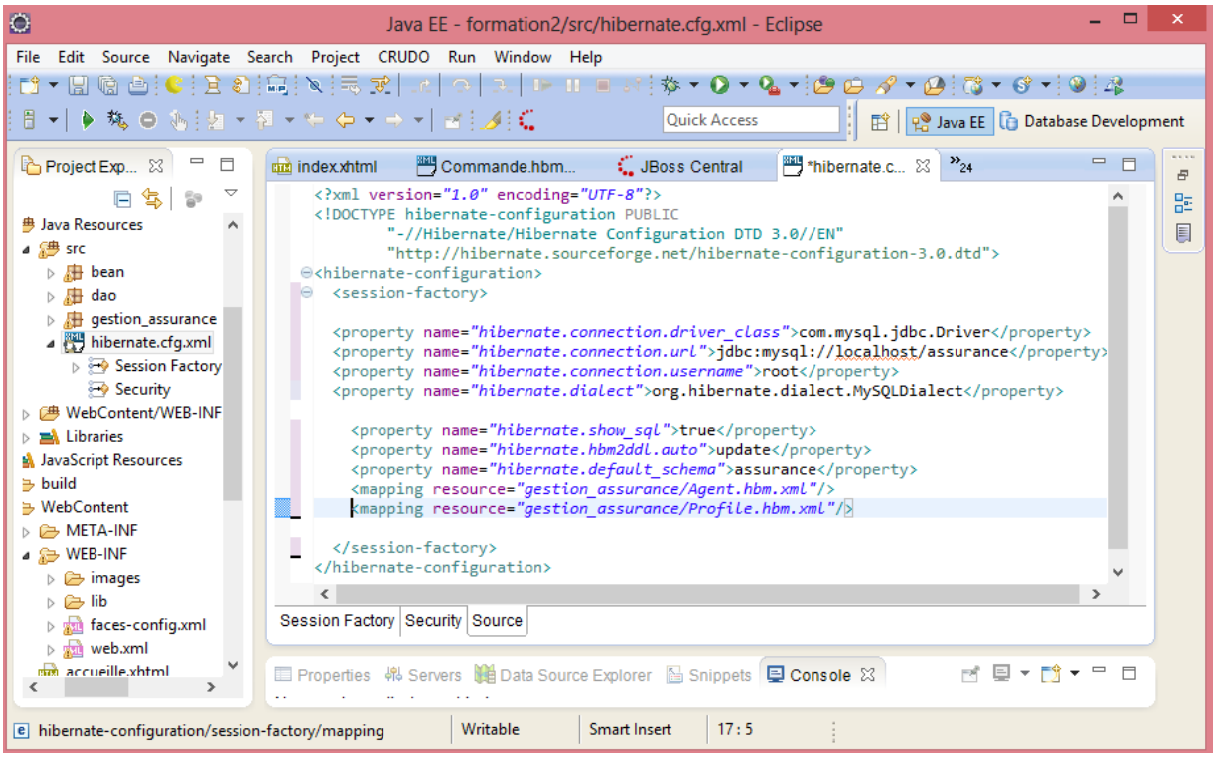


Figure IV.4 : le fichier Hibernate.cfg.xml

c- Création des fichiers xml de mapping :

Ces fichiers sont des éléments majeurs puisqu'ils vont permettre à Hibernate de faire le pont entre les classes de persistance et les sources de données.

Dans cette partie, on va présenter la du fichier de mapping relatif à la table «profile »nommée

profile .hbm.xml

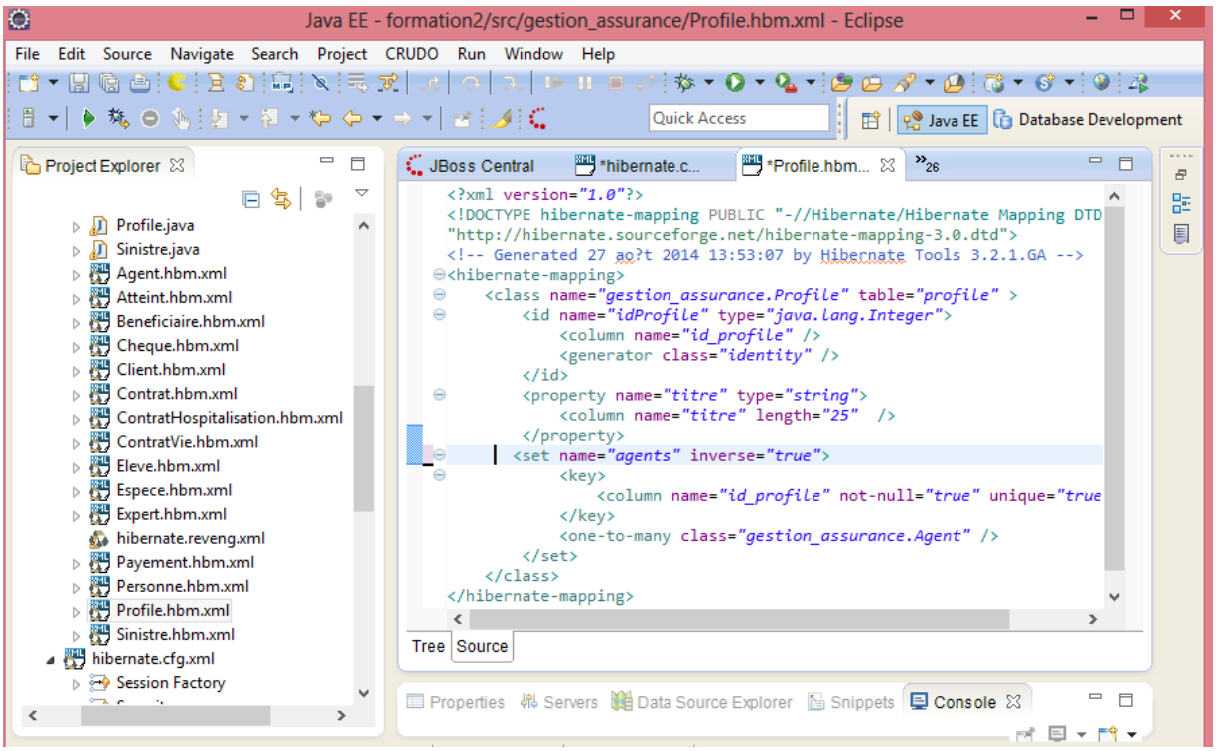


Figure IV.5 : Exemple d'un fichier de mapping (profile)

Il est absolument indispensable d'ajouter la référence du fichier « profile.hbm.xml » dans le fichier de configuration « hibernate.cfg.xml » au niveau de la balise <mapping /> comme suit:

```
<mapping resource="gestion_assurance /profile.hbm.xml" />
```

d- Création des Classes de données :

Une classe de données est un Javabean qui va encapsuler les propriétés de la table dans ses champs private avec des getters et setters et qui a un constructeur par défaut.

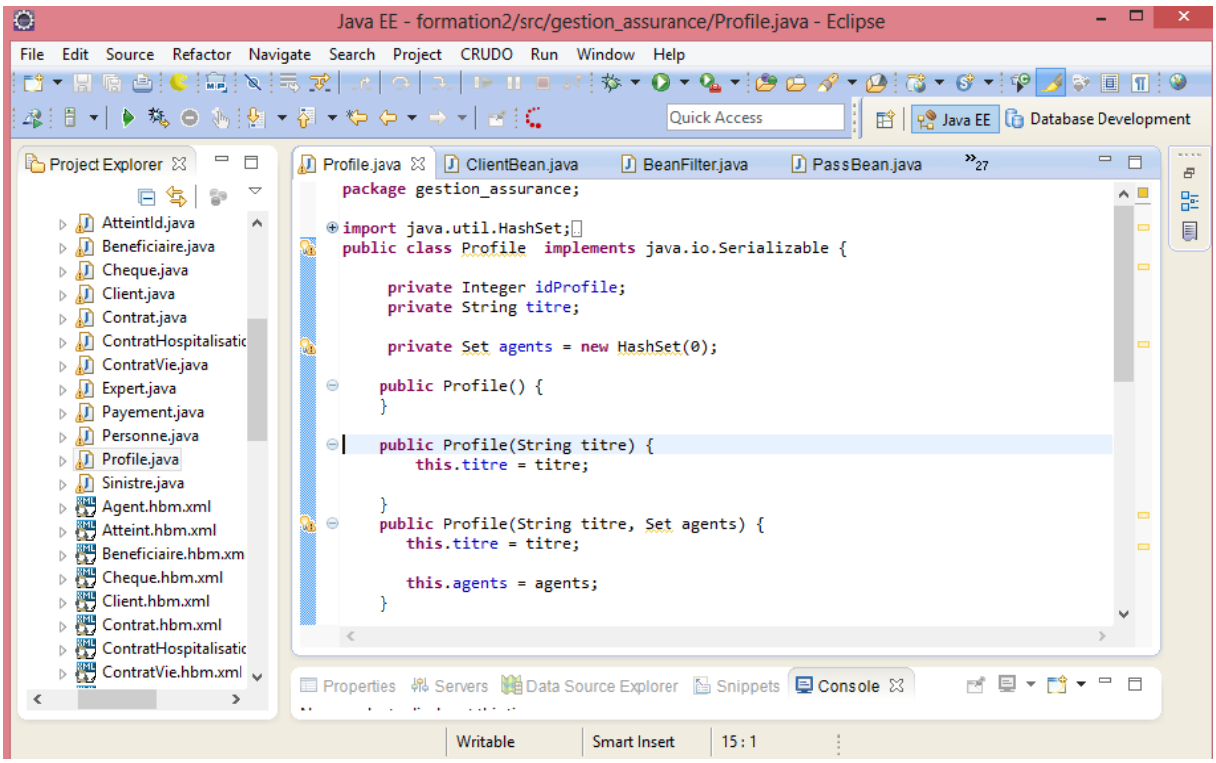


Figure IV.6 : exemple d'une classe (classe profile)

e- La classe HibernateUtil :

La classe Hibernate nommée SessionFactory permet d'établir la connexion avec la source de données à partir du fichier de configuration « hibernate.cfg.xml ».On remarque que la classe

SessionFactory serait instanciée autant de fois qu'il y a de threads, il est donc plus adapté de rendre une même instance de SessionFactory accessible par les threads. Cette classe possède une méthode appelée currentSession() qui retourne la session hibernate en cours si elle existe sinon elle se charge d'ouvrir une nouvelle session.

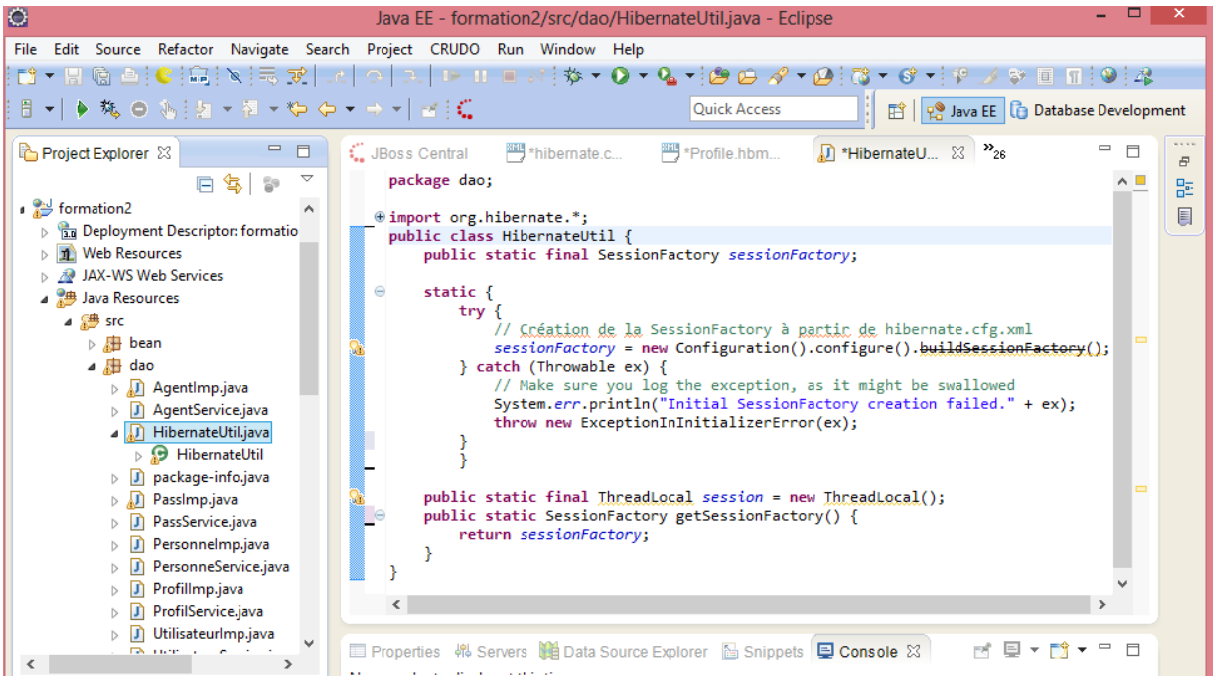


Figure IV.7 : le fichier HibernateUtil

4- Primefaces /jsf:[27]

Java server faces est un Framework de développement d'applications web en java permettant de respecter le model d'architecture MVC et basé sur des composantes coté présentation.

- Architecture MVC pour séparer l'interface utilisateur, la couche de persistance et les processus métiers, utilisant la notion d'événement, conversion des données, validation des données (par exemple des champs de formulaires requis).
- Automatisation de l'affichage des messages d'erreur en cas de problèmes de conversion ou de validation.

Le premier objectif de JSF, est de procurer un environnement de développement permettant de construire une interface de type web, sans devoir toucher au code HTML et JavaScript. Ceci est réalisé par la mise en place d'un mapping entre l'HTML et les objets concernés. JSF est donc basé sur la notion de composants, comparable à celle de Swing, ou l'état de ces composants est sauvegardé puis restauré au retour de la requête.

L'implémentation de Jsf nécessite l'implémentation des deux fichiers suivants :

- **faces-config.xml :**

Le faces-config.xml, est le fichier de configuration, qui permet de gérer les beans, et les règles de navigation entre les pages. Vous pouvez apercevoir, ci-dessous un bref, exemple de fichier de configuration.

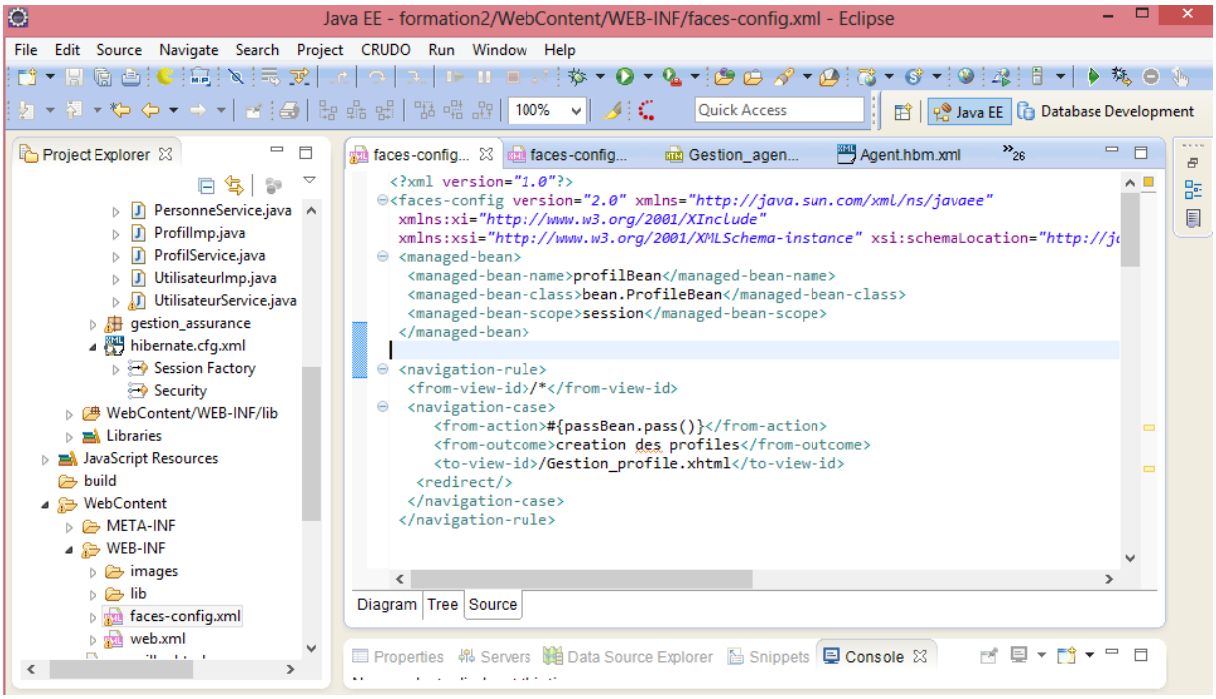


Figure IV.8 : fichier de configuration face-config.xml

- **web.xml : (**descripteur de déploiement)

Le web.xml qui permet de définir l'emplacement des pages JSP, ainsi que de possible contraintes de sécurité concernant l'accès à des pages. Il établit également l'endroit ou se situe le Faces Servlet. Faces Servlet, est la classe, qui est le moteur de chaque application JSF. Chaque application JSF à sa propre Faces Servlet, qui gère toutes les informations relatives à la requête courante.

Voici un exemple de fichier web.xml, qui est normalement créer à la génération du projet sous eclipse. (

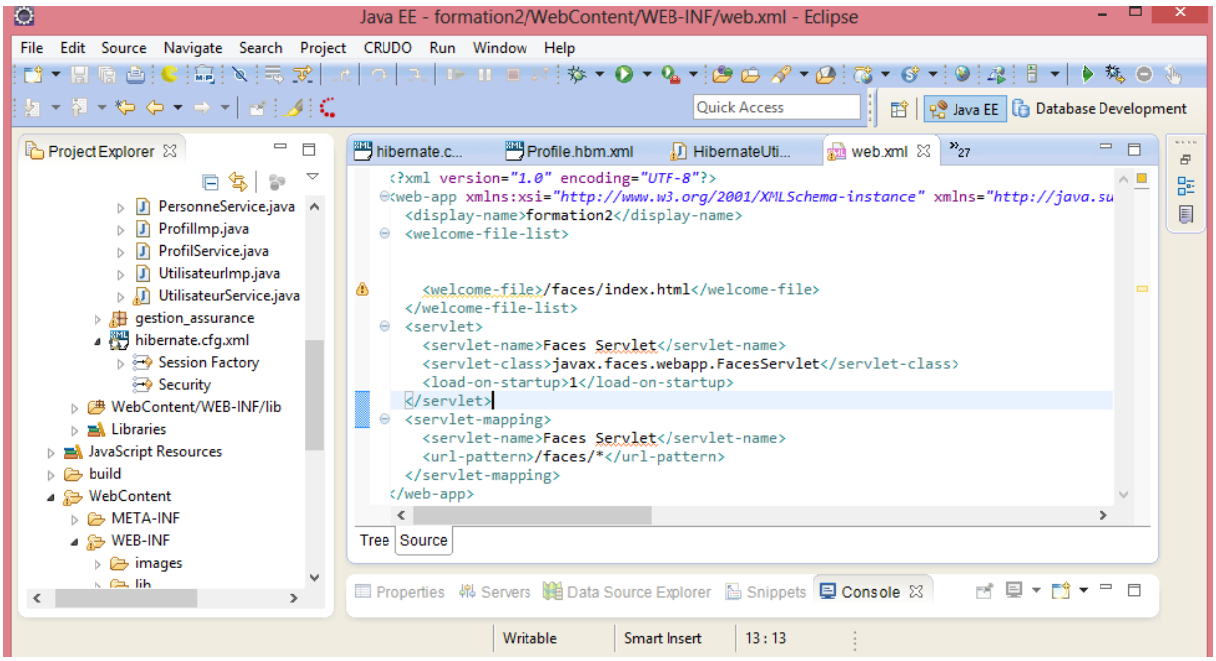


Figure IV.9 : Fichier de configuration web-xml

- **Prime faces :**

Prime faces c'est l'un des composants de JSF c'est une bibliothèque open source de composantes JSF il est basé coté serveur sur l'API standard de ce dernier.

Elle permet de mettre à disposition des interfaces clients riches dans les applications web JEE.

- Alternatives de Primefaces :

Il existe des alternatives à Primefaces dont les plus connues et les plus anciennes sont RichFaces de la communauté JBoss et IceFaces qui s'est suicidé à partir de sa version 3 en copiant en totalité le code de primefaces.

5- Apache server : [25]

Véritable moteur d'hébergement Internet, Apache est un serveur HTTP produit par l'Apache Software Foundation. C'est le serveur HTTP le plus populaire du World Wide Web. Apparu en avril 1995, Apache fonctionne principalement sur les systèmes d'exploitation Unix (GNU/Linux, BSD et UNIX) et Windows (depuis la version 2). Apache est conçu pour supporter de nombreux modules lui donnant des fonctionnalités supplémentaires : interprétation du langage Perl, PHP et Python, serveur proxy, réécriture d'URL, négociation de contenu, protocoles de communication additionnels, etc. Les possibilités de configuration d'Apache sont une fonctionnalité phare. Le principe repose sur une hiérarchie de fichiers de configuration, qui peuvent être gérés indépendamment. C'est notamment utile aux hébergeurs

Web qui peuvent ainsi servir les sites de plusieurs clients à l'aide d'un seul serveur HTTP. Pour les clients, cette fonctionnalité est rendue visible par le fichier .htaccess.

Le choix du serveur apache est basé essentiellement sur :

- Un niveau élevé de performances pour des exigences matérielles modestes
- Logiciel libre
- Son développement est actif
- Très portable (fonctionne sous les différentes plates-formes sous UNIX et sous Windows)
- Extensible, modulaire et configurable
- Gratuit
- Robuste et sécurisé.

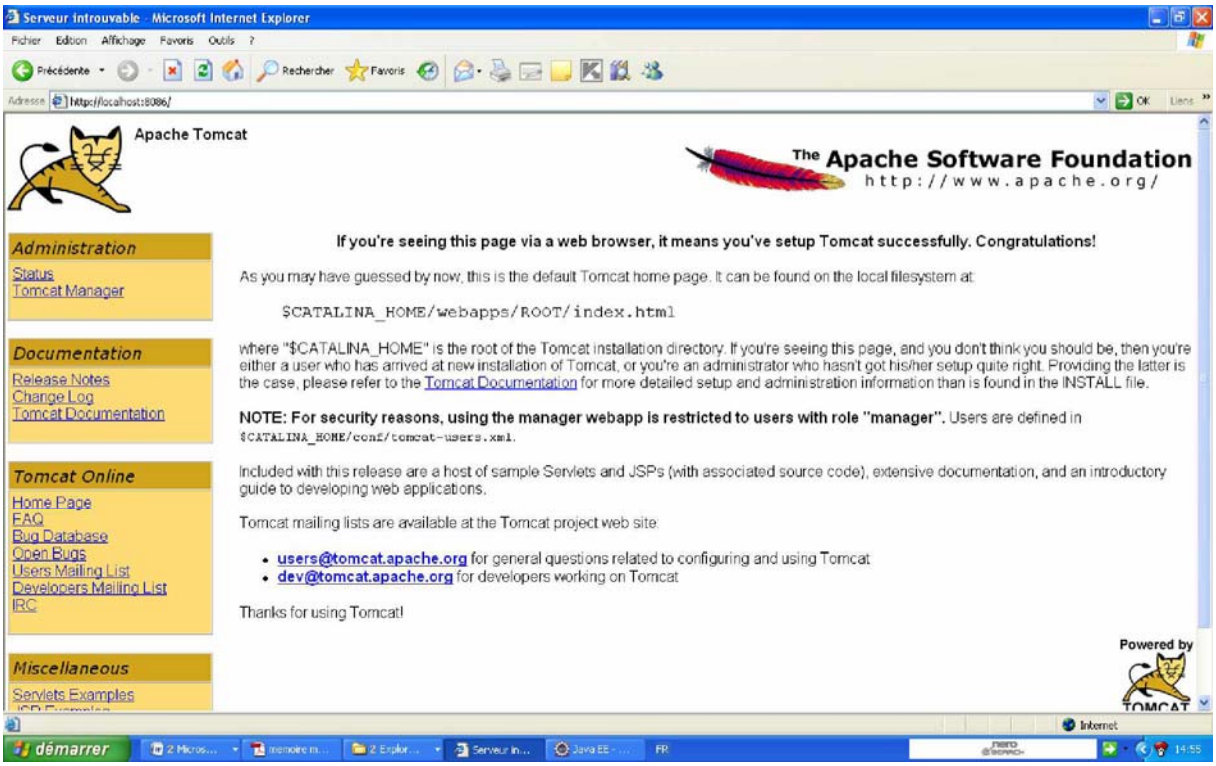


Figure IV.10 : interface d'Apache Tomcat

IV- Langages de programmation :

1- Le langage cote serveur :

1-1- java : [26]

La page web (applet) ou encore comme langage serveur (jsp). C'est un langage de programmation orienté objet, développé par SUN Microsystems. Il permet de créer des logiciels compatible avec de nombreux systèmes d'exploitations (Windows, Linux,Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphone portables et assistants personnels. Enfin ce langage peut être utilisé sur internet pour des petites applications intégrées à la page web (appelet) ou encore comme langage serveur (jsp).

1-2- Le language SQl pour les requêtes de la base de données : [14]

Le SQL (*Structured Query Language*) est un langage informatique qui permet d'interagir avec des bases de données relationnelles. C'est le langage pour base de données le plus répandu, et c'est bien sûr celui utilisé par MySQL. C'est donc le langage utiliser pour dire au client MySQL d'effectuer des opérations sur la base de données stockée sur le serveur MySQL.

Il a été créé dans les années 1970 et c'est devenu standard en 1986 (pour la norme ANSI - 1987 en ce qui concerne la norme ISO). Il est encore régulièrement amélioré.

1-3- Le langage HQL : [22]

Hibernate Query Langage est un langage d'interrogation des classes persistantes avec une syntaxe proche du SQL. Cependant les objets manipulés sont les classes et membres du mapping contrairement au SQL qui réalise les requêtes directement sur la base de données. Aussi, le HQL contient des fonctionnalités spécifiques au modèle objet**.**

2- Le langage côté client :

2-1- Le langage HTML :

Hypertext Markup Language (HTML) est le langage qui prédomine dans les pages web. Il repose sur SGML (*Standard Generalized Markup Language*), un métalangage standard permettant de définir des langages a marqueurs. HTML utilise des balises, ou marqueurs, pour structurer le texte en paragraphes, listes, liens, boutons, zones de texte, etc.

Une page HTML est un document texte utilise par les navigateurs pour présenter du texte et des images : ce sont des fichiers texte portant souvent l'extension .html ou .htm. Une page web est formée d'un contenu, de marqueurs permettant de changer certains aspects de ce contenu et d'objets externes comme des images, des vidéos, du code JavaScript ou des fichiers CSS.

2-2- XML (eXtensible Markup Language): [24]

XML (eXtensible Markup Language), est le standard soutenu par le W3C pour le balisage de documents. Il définit une syntaxe générique utilisée pour formater des données avec des balises simples et compréhensibles par l'homme. Ce format est suffisamment souple pour être adapté à des contextes aussi variés que les sites WEB, l'échange de données électroniques, les dessins vectoriels, les menus déroulants de logiciels, les descripteurs de déploiement, les procédures d ‘appel à distance (XML-RPC), etc

2-3- Le langage XHTML : [23]

XHTML a été crée peu de temps après HTML 4.01. Ses racines puisent dans HTML, mais avec une reformulation en XML strict. Ceci signifie qu’un document XHTML est un document XML qui respecte un certain schéma et peut être représente graphiquement par les navigateurs – un fichier XHTML (qui porte l’extension .xhtml) peut être directement utilise comme du XML ou être affiche dans un navigateur. Par rapport a HTML, il a l’avantage de permettre une validation et une manipulation du document a l’aide d’outils XML standard (XSL ou *eXtensible Stylesheet Language* ; XSLT ou *XSL Transformations ;* etc.).

2-4- Le Java Script : [15]

Le JavaScript est un langage de script incorporé dans un document HTML.

Historiquement il s’agit du premier langage de script pour le web, mis au point par Netscape en 1995.

Ce langage est un langage de programmation qui permet d’apporter des améliorations au langage HTML en permettant d’exécuter des commandes du côté client, c'est-à-dire au niveau du navigateur et non du serveur web.

Ainsi, ce langage est fortement dépendant du navigateur appelant la page web dans laquelle le script est incorporé, mais en contrepartie il ne nécessite pas de compilateur, contrairement au langage *Java***,** avec lequel il a longtemps été confondu.

IV- Conception de la base de données :

1- Le SGBD MySQL :

Après le choix du serveur Apache et du langage java, le SGBD à utiliser s'impose par lui-même : MySQL. MYSQL est un véritable serveur de base de données SQL (Structured Query Language) qui est un langage de requêtes vers les bases de données exploitant le modèle relationnel. Il en reprend la syntaxe mais n'en conserve pas toute la puissance puisque de nombreuses fonctionnalités de SQL n'apparaissant pas dans MYSQL (sélection imbriquées, clés étrangères…).

MYSQL est un serveur de base de données SQL multitraitements, il est caractérisé par sa rapidité et sa robustesse. Il présente l'avantage d'être portable (il peut être compilé sur plusieurs plates-formes comme Windows, Unix…etc.). De plus, il est facile à utiliser, standard (il utilise SQL), robuste, gratuit et surtout totalement pris en charge par PHP

2- PHPMyAdmin :

L'outil PhpMyAdmin est développé en PHP (ensemble de scripts PHP), il offre une interface graphique pour l'administration des bases de données MYSQL via un navigateur Web.

Les fonctions principales de PhpMyAdmin sont :

- Création de nouvelles bases de données ;
- Création/suppression/modification des tables ;
- L'édition, l'ajout et la suppression de champs ;
- L'exécution de commandes SQL et de requêtes ;
- Gérer les privilèges des utilisateurs

Table personne :

Nom du champ	Description du champ	Type de donnée	Clé (s)
Id_personne	Identifiant de la personne	Int(11)	PRIMAIRE
Nom _personne	Nom de la personne	Varchar(30)	
Prénom _ personne	Prénom de la personne	Varchar(30)	
téléphone	Téléphone de la personne	Varchar (15)	
adresse	Adresse de la personne	Varchar (40)	
profession	Profession de la personne	Varchar (40)	

Table Agent :

Nom du champ	Description du champ	Type de donnée	Clé (s)
Id_agent	Identifiant de l'agent	Int(11)	PRIMAIRE
Login-agent	login de l'agent	Varchar(8)	
Mdp_agent	Mot de passe	Varchar(8)	
Id_profile	Identifiant du profile	Int(11)	ETRANGERE
Id_personne	Identifiant personne	Int (11)	ETRANGERE

Table client:

Nom du Champ	Description du champ	Type de donnée	Clé(s)
Id_client	Identifiant de client	Int(11)	PRIMAIRE
Id_agent	Identifiant de l'agent	Int(11)	ETRANGERE
Id_personne	Identifiant de personne	Int(11)	ETRANGERE

Table profile :

Nom de champ	Description du champ	Type de donnée	Clé(s)
Id_prof	Identifiant de profile	Int(11)	Primaire
Titre_prof	Titre de profile	Varchar(25)	

Table contrat :

Nom du champ	**Description du champ**	**Type de donnée**	**Clé (s)**
Id_contrat	Identifiant de contrat	Bigint(20)	Primaire
Num_contrat	Numéro de contrat	Varchar(11)	
Montant_payé	Montant payé	Double	
Type_contrat	Type de contrat	Varchar(30)	
Desig_contrat	Désignation de contrat	Varchar(40)	
Date_contrat	Date de contrat	Date	
Etat	L'état de contrat	Varchar(36)	
Date_décés	Date décès	Date	
Taux_ rembours	Taux de remboursement	Varchar(12)	
Date_hosp	Date de l'hospitalisation	Date	
Date_reactivation	Date de réactivation du contrat	Date	
Date_Validité	Date de validité de contrat	Date	
Type_hosp	Type d'hospitalisation	Varchar(45)	
Id_Client	Identifiant de client	Int(11)	ETRANGERE

L'architecture de notre application selon le model MVC ce présente comme suit :

JSF **(java server face)** Implémentation de primfaces. (Vue)	Implémentation par l'utilisation des interfaces-(objetservice,objetImp,javabean). (Contrôle)	Hibernet «persistances des donneés »la couche model (Model)
1. Un ensemble de composants réutilisables ; qui communiquent avec les java bean. 2. Descripteur de déploiement (web.xml). 3. Fichier de configuration (faces-config).	1. Pour chaque segment de la réalité étudié on définit un service et son implémentation, ces derniers nous ont permit de répondre aux règles de gestion du champ d'étude. 2. javabean : les objets qui font le lien entre la partie métier et l'interfaces graphique.	1. Fichier de configuration Hibernate : hibernate.cfg.xml(configuration generale-acces a la base de donné). 2. Fichier de mapping : ensemble de fichiers xml-lien entre les fichiers java et les tables dans la BDD. 3. Fichier util-génération : de session pour l'exploitation du framework Hibernate et matérialisation des accès à la base de donnés.

Tableau IV.1 : tableau architecture MVC

V-Présentation de Quelques interfaces

Page accueil

Figure IV.11: Page accueil (1)

Page à propos direct assurance

Figure IV.12: Page accueil (2)

Page Authentification :

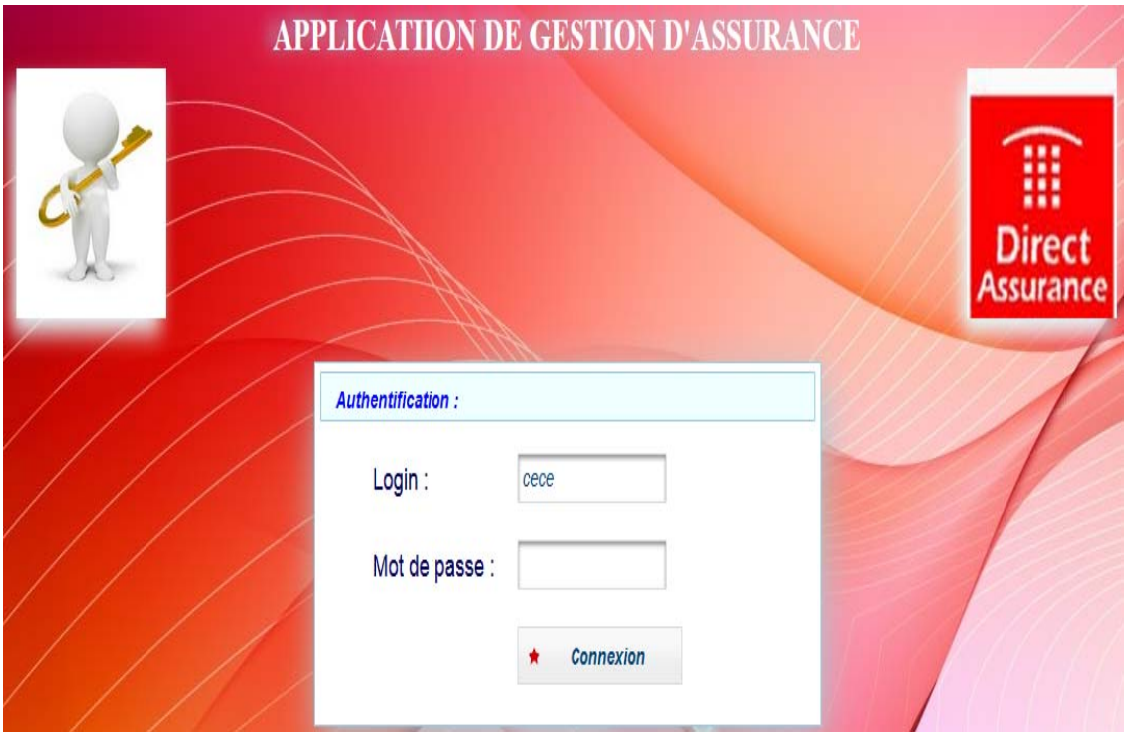


Figure IV.13:page authentification

Page gestion client et contrat

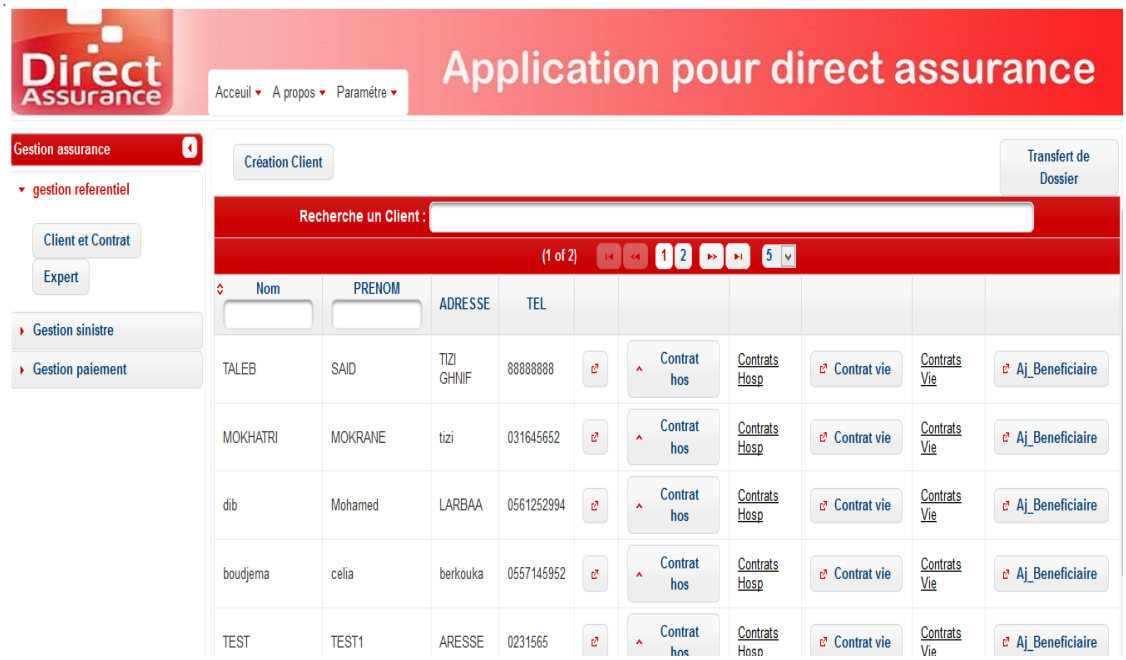


Figure IV.14 : page gestion client et contrat

Page création expert

Figure IV.15 : page création expert

Page gestion agents :

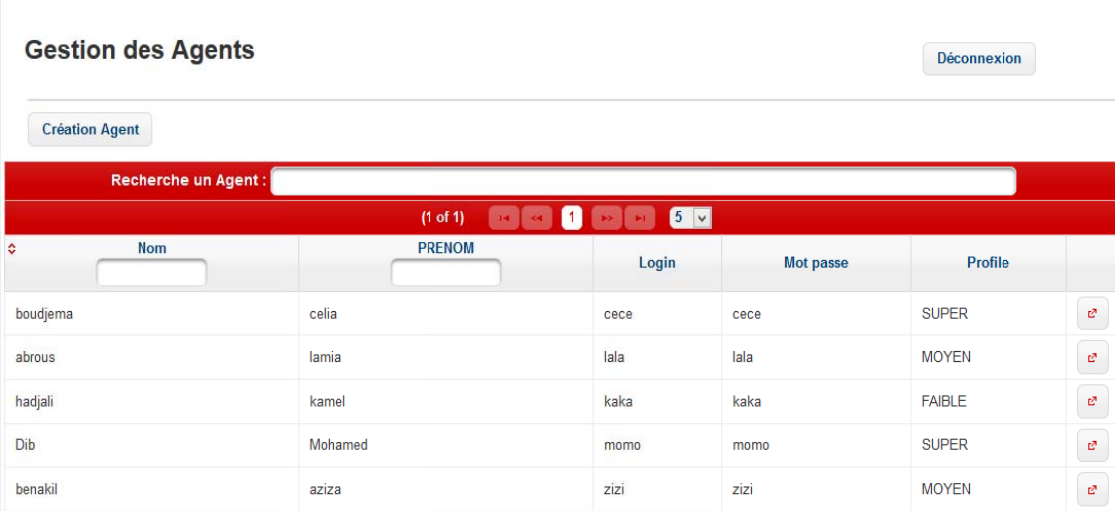


Figure VI.16 : gestion agent

VI-Conclusion :

Dans ce chapitre, nous avons présenté l'environnement et les différents outils utilisés pour le développement de notre application, ainsi que quelques exemples d'interfaces pour illustrer les principales fonctionnalités de notre application.

Conclusion générale

Conclusion générale

L'objectif de notre projet était de concevoir et de réaliser une application web pour l'entreprise « Direct Assurance » qui offre un ensemble des services tels que l'enregistrement des ses clients ainsi que leur contrats, la gestion des agents qui sont à leur service et le contrôles de leurs droits d'accès.

Pour commencer notre travail, il nous a fallu faire une étude théorique sur tous les outils nécessaires pour le développement d'une application WEB et plus précisément les outils les plus récents et les plus utilisés. Ceci nous a amené d'approfondir nos connaissances sur les framework, qui est une technologie WEB très récentes, d'étudier plusieurs d'entre eux, de les tester et de les implémenter pour pouvoir en manipuler l'un deux qui est Hibernate.

Nous avons également étudié le modèle client/serveur ainsi que le modéle MVC qui est une architecture moderne de structuration.

Ensuite, pour modéliser notre application, on a utilisé le langage de modélisation UML qui nous a permis de décrire les plans d'élaboration et de construction de notre application.
Enfin, pour réaliser notre application plusieurs technologies nous ont été nécessaires, on citera le langage XHTML, le java, le langage de base de données SQL et le HQL qui est un langage spécifique au framework Hibernate.

Après le passage par les différentes étapes précédemment citées, l'application a abouti à un logiciel fonctionnel qui répond globalement aux attentes de l'entreprise mais qui peut également êtres amélioré et perfectionner pour apporter plus de rentabilité à l'entreprise.

L'application que nous avons développée nous a permit d'acquérir des connaissances dans le domaine de la programmation, la conception et la réalisation d'application web client/serveur.

Bibliographie

Bibliographie

[1] mémoireonline.com

[2] http://formation-bts-assurances.esaassurance.com

[3] http://www.assurland.com

[4] http://www.lecomparateurassurance.com

[5] http://www.ffsa.fr

[6] http://assurancesplus.blogspot.com

[7] https://www.creditmutuel.fr

[8]wikipédia

[9 http://www.jmdoudoux.fr/

[10] guillaume.cresta.free.fr/CV/Framework_Hibernate.pdf

[11] mémoire L3

[12] http://perso.modulonet.fr

[13] cai.cegep-lanaudiere.qc.ca

[14] openclassroum.com

[15] commentçamarche.com

[16] www.brainstormsolutions.ca

[17] www.creditmutuel.fr

[18] www.juritaravail.com

[19] xmlfr.org

[20] http://www.orsys.com/

[21] www.tutorielspoint.com

[22] http://www.futura-sciences.com/
[23]books.google.dz

[24] w3.gril.univ-tlse2.fr

[25]http://www.all2all.org/fr

[26] http://www.futura-sciences.com
[27] jsf: http://primefaces-fr.blogspot.com

[28] http://www.index-assurance.fr/compagnies/directes