République Algérienne Démocratique et Populaire

الجمهورية الجزائرية الديمقراطية الشعبية

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

وزارة التعليم العالي والبحث العلمي

Université Amar Telidji Laghouat

Faculté des sciences et sciences de l'ingénieur

Département de génie informatique

جامعة عمار ثليجي الأغواط

كلية العلوم و علوم الهندسة

قسم الإعلام الآلي

MEMOIRE DE FIN D'ETUDES

Pour l'obtention du diplôme d'Ingénieur d'Etat en Informatique

OPTION : Systèmes d'informations avancées

Thème

Conception et réalisation d'une application e-commerce basée sur les services web

Présenté par :

- **Brahimi Amar**
- **Dif Ahmed**

Encadré par :

- Kachna Lakhdar

Promotion : 2010/2011

N° d'ordre :……..

Abstract

A Web Service is a software application, identified by an URI whose interfaces and bindings are defined, described and discovered in the forme of XML documents. A Web Service makes use of direct interaction with other software agents using SOAP messages.

Differents Web Services can be located in remote places and provided by differents Providers.

The objective of our work is to study the different aspects related to the new technology of java EE in the field of Web Services and to adapt and implement an e-commerce Web Application based of Web Services in order to descover this new technologie.

Key words : Web Service, XML , SOAP, Java EE,e-commerce

Résumé

Un Web Service est une application logicielle, identifiée par un URI, dont les interfaces et les liaisons sont définies, décrites et découvertes sous forme de documents XML. Un Web Service met en œuvre l'interaction directe avec d'autres agents logiciels par l'utilisation de messages SOAP. Les différents Web Services peuvent être localisés dans des endroits distants et fournis par différents fournisseurs.

L'objectif de notre travail est d'étudier les différents aspects liés à la nouvelle technologie de Java EE dans le domaine des services Web et d'adapter et d'implémenter une application web e-Commerce avec des services web associés afin de découvrir ces nouvelles technologies.

Mots clés : Web Service, XML, SOAP, java EE, e-commerce.

Table des matières

Chapitre 2 : Le e-commerce

PARTIE II : Conception et réalisation

Chapitre 3 : Conception du système

Chapitre 4 : Mise en œuvre

Listes des figures :

Listes des tableaux :

Introduction Générale :

L'intégration d'application est l'élément clé de l'évolution actuelle des systèmes d'information et des processus en interne à l'entreprise et lors d'entreprise étendue à son réseau de partenaires (e-Business). En effet, le développement de toute nouvelle application ou d'une procédure commerciale requiert une intégration manuelle et ad hoc des sources de données issues d'applications et de bases de données très hétérogènes. La complexité de l'intégration est variable mais elle n'en reste pas moins une des sources de coût des plus lourdes pour les entreprises.

Dans ce contexte de réduction des coûts d'intégration, la technologie des web services propose une solution élégante à l'accès et l'intégration de ressources distribuées et hétérogènes.

L'objectif des web service est d'aller au-delà des limites de l'intégration telle qu'elle se fait actuellement dans les technologies existantes d'intégration d'applications (comme le CORBA par exemple).

En effet ces dernières induisent en général un grand coût de mise en œuvre, ce qui en restreint l'usage aux grandes entreprises. Or, avec l'explosion et l'ouverture apportées par Internet et l'arrivée massive de petites entreprises dans le commerce en ligne, ce n'est clairement plus adapté. Ainsi, le souhait est de passer à des technologies d'intégration décentralisées où chaque partenaire possède ses propres composants dans un système d'intégration faiblement couplé qui autorise une plus grande flexibilité dans les relations inter partenaires.

Objectifs visés par le projet

Dans notre projet ,on a pas allé plus loin avec le détail du Web Service telle que la composition et la sécurisation ,mais on a illustrer un peu cette technologie a partir du développement d'une application web (Dans notre cas c'est un site web e-commerce é) ,ce dernier communique avec d'autres entreprises partenaires qui sont dans notre cas une banque (le premier web service) pour faire la validation des cartes ,et une société de transport (le deuxième web service) qui livre les produit vers leurs clients ,ces derniers seront interactif entre eux à l'aide des technologies du web service .

Structure du mémoire :

Le présent mémoire est organisé en deux parties principales : l'état de l'art, la conception et réalisation du projet.

Partie I : cette partie représente l'état de l'art . Elle est divisée en deux chapitres :

- **Chapitre 1 :** présente la théorie de services Web. Nous avons exposé les différentes couches de protocoles des services Web, qui mettent en œuvre des technologies telles que HTTP, XML, SOAP, WSDL et UDDI.
- **Chapitre 2 :** présente de façon générale, le domaine du commerce électronique.

Partie II : cette partie décrit la conception et la réalisation de notre application. Elle est composée de deux chapitres :

- **Chapitre 3 :** présente l'étude de cas d'une application de commerce électronique. La société fictive DZ-INFO veut informatiser son activité de vente de matériels informatique. Pour ce faire, elle a besoin d'un site pour les internautes, d'un client riche pour ses employés et de dialoguer avec ses partenaires externes (banque et transporteur). Dans ce chapitre, nous sommes basés sur le formalisme UML pour : déterminer les fonctionnalités du système en se basant sur le diagramme de cas d'utilisation et déterminer la vue statique de l'application, en se basant sur le diagramme de classe
- **Chapitre 4 :** se concentre sur l'architecture technique et logicielle de l'application. Ce chapitre présente brièvement les outils et API utilisés pour le développement.

Partie I

Etat de l’art

Chapitre 1
Les Web Services

1. Introduction :

Actuellement, les entreprises se retrouvent avec des systèmes de plus en plus gros et complexes, de ces systèmes sont souvent incompatibles entre eux car les langages de programmation utilisés sont souvent eux aussi différents par exemple un vieux système d'information d'une banque écrit en COBOL est incompatible avec la technologie JEE employé dans le portail web de celle-ci. Lorsqu'il survient le besoin de faire interagir des systèmes hétérogènes il faut donc soit modifier le code de cette application voir la réécrire complètement ce qui est très coûteux, très complexe et très lent.

On veut avoir un système qui bien associe avec les systèmes existants, a capacité de changer rapidement, et plus flexible, et peut être facilement déployé sur plusieurs départements à distance. Dans ce contexte là, l'architecture orientée service (notée SOA pour *Services Oriented Architecture*) est considérée comme le premier candidat pour désigner les nouveaux systèmes. Les services Web sont de plus en plus perçus comme une approche technologique acceptable pour la mise en œuvre d'une architecture SOA.

D'autres technologies telles que RMI, DCOM et CORBA ont précédemment adopté ce style architectural mais ont généralement échoué en raison de la diversité des plates-formes utilisées dans les organisations et aussi parce que leur usage n'était pas adapté à Internet (problème de passage à travers des Firewalls, etc.) d'où la lenteur, voire l'absence de réponses sur ce réseau. Les applications réparties fondées sur ces technologies offrent des solutions caractérisées par un couplage fort entre les objets. Les solutions proposées par les services Web, permettent néanmoins un couplage moins fort. De plus, l'utilisation des technologies standards du Web telles HTTP et XML par les services Web facilite le développement d'applications réparties sur Internet, et permet d'avoir des applications très faiblement couplées. L'intégration est sans doute le facteur essentiel qui favorise l'utilisation des services Web.

2. Définition des Web Services :

- Un Web Service est un programme informatique permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans les environnements distribués. Il s'agit donc d'un ensemble de fonctionnalités exposées sur Internet ou sur un intranet, par et pour des applications ou machines, sans intervention humaine, et en temps réel [**Wiki]**.

- Un Web Service est une application logicielle, identifiée par URI, dont les interfaces et les liaisons peuvent être définies, décrites et découvertes sous forme de documents XML. Un Web Service met en œuvre l'interaction directe avec d'autres agents logiciels par l'utilisation des messages au format XML, échangés sur des protocoles Internet **[W3C]**.

Le Web est par nature une plateforme interopérable. Selon la définition du W3C, une application peut être qualifiée de Web Service si elle correspond aux profils technologiques suivants :

- **Identification** : un Web Service est identifié par un URI.
- **Description** : les interfaces et les liaisons d'un Web Service sont décrites (et donc peuvent être définies et découvertes) au moyen d'un langage XML.
- **Message** : un Web Service communique avec les autres agents logiciels au moyen des messages au format XML.
- **Transport** : les messages sont transmis via des protocoles internet. La figure ci-dessous illustre le profil technologique d'un Web Service **[Mae03].**

IDENTIFICATION	**URI**
DESCRIPTION	WSDL
MESSAGE	SOAP
TRANSPORT	HTTP

Tableau 1 : Profil technologique d'un Web Service s'appuyant sur SOAP et liaison HTTP.

3. L'intérêt d'un Service Web :

Les services Web fournissent un lien entre applications. Ainsi, des applications utilisant des technologies différentes peuvent envoyer et recevoir des données au travers de protocoles compréhensibles par tout le monde.

Les services Web sont normalisés car ils utilisent les standards XML et HTTP pour transférer des données et ils sont compatibles avec de nombreux autres environnements de développement. Ils sont donc indépendants des plates-formes. C'est dans ce contexte qu'un intérêt très particulier a été attribué à la conception des services Web puisqu'ils permettent aux entreprises d'offrir des applications accessibles à distance par d'autres entreprises.

Cela s'explique par le fait que les services Web n'imposent pas de modèles de programmation spécifiques. En d'autres termes, les services Web ne sont pas concernés par la façon dont les messages sont produits ou consommés par des programmes.

Cela permet aux vendeurs d'outils de développement d'offrir différentes méthodes et interfaces de programmation au-dessus de n'importe quel langage de programmation, sans être contraints par des standards comme c'est le cas de la plate-forme CORBA qui définit des ponts spécifiques entre le langage de définition IDL (Interface Definition Language) et différents langages de programmation. Ainsi, les fournisseurs d'outils de développement peuvent facilement différencier leurs produits avec ceux de leurs concurrents en offrant différents niveaux de sophistication.

Les services Web représentent donc la façon la plus efficace de partager des méthodes et des fonctionnalités. De plus, ils réduisent le temps de réalisation en permettant de tirer directement parti de services existants.

4. Modèle d'interaction des Web Services :

D'après la figure ci-dessous la collaboration entre Web Services s'appuie sur un modèle d'interaction dont les composants assurent trois rôles : le fournisseur de services, l'annuaire de services et le demandeur de services.

4.1. Scénario général de fonctionnement des Web Services : [SWS10]

Dans le scénario de fonctionnement normal, un fournisseur de service héberge un module logiciel implémentant un ou plusieurs Web Services accessibles via le réseau. Il définit une description de service et le publie en le faisant enregistrer dans l'annuaire de service. Un client utilise alors la description du service pour établir une connexion avec le fournisseur du service et invoquer ou interagir avec l'implémentation du Web Service. Ce scénario peut être récursif dans le cadre d'invocation de services récursifs.

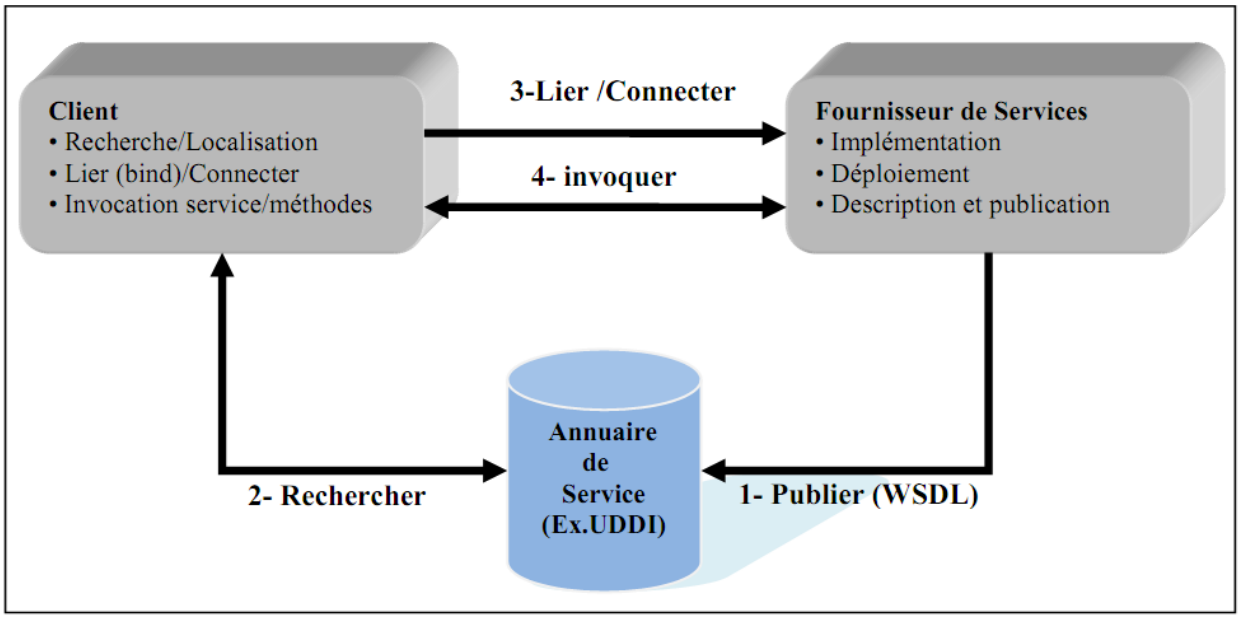

Figure 1 : Scénario général de fonctionnement des Web Services **[SCA10].**

Il existe trois types d'opérations pour tirer pleinement partie de modèle :

- La publication de description du service (**publish**) ;
- La recherche et la découverte de la bonne description du service (**find**) ;
- L'association ou l'invocation des services basés sur la description (**bind**) ;

Afin d'assurer la collaboration entre applications, chacun des composants de ce modèle d'interaction présente une facette d'architecture technique :

- **Le fournisseur de service :** correspond au propriétaire du service. D'un point de vue technique, il est constitué par la plate-forme d'accueil du service.

- **Le client :** correspond au demandeur de service. D'un point de vue technique, il est constitué par l'application qui va rechercher et invoquer un service. L'application cliente peut être elle-même un web service.

- **L'annuaire des services :** correspond à un registre de descriptions des services offrant des facilités de publication de ces services à l'intention des fournisseurs ainsi que des facilités de recherche des services à l'intention des clients.

4.2. Étapes d'exécution des Web Services : [SWS10]

Les principales étapes d'exécution d'un Web Service sont les suivantes :

➢ Etape 1. Définition et description des Web Service :

On doit décrire d'un point de vue informatique ce que fait le Web Service, la solution qu'il propose, la définition est faite en WSDL au sein du fournisseur du Web Service.

➢ Etape 2. La publication du Web Service :

Une fois le Web Service défini et décrit en terme de mise en œuvre, il peut être déclaré dans un annuaire, on parle alors de publication des Web Services afin de le

rendre accessible aux clients. La publication sera effectuée au sein d'un annuaire dédié.

- **Etape 3. Découverte du Web Service :**

  Le demandeur de service lance la recherche d'un service correspondant à ses besoins dans un annuaire UDDI qui peut être publique ou privé.

- **Etape 4. Récupération des informations de description du service :**

  Le demandeur de service récupère de l'annuaire UDDI la description du service au format WSDL.

- **Etape 5. Connections aux Web Services :**

  La communication entre le composant demandeur du service et le fournisseur de service est assurée, en phase d'exploitation à travers des SOAP(Simple Object Acces Protocol) qui servent d'interfaces entre ces composants et les protocoles de communication de l'infrastructure de déploiement. Le proxy du composant demandeur émet une requête SOAP au composant fournisseur du service. Le protocole HTTP véhicule le message SOAP jusqu'au listener du fournisseur du service.

- **Etape 6. Réponse du Web Service :**

  Le Web Service du fournisseur renvoie sa réponse au demandeur sous la forme d'un message SOAP via HTTP.

5. L'architecture des Web Services : [SCA10]

L'architecture des Web Services est implémentée à l'aide de diverses technologies organisées en quatre couches, la figure 3 illustre cette architecture :

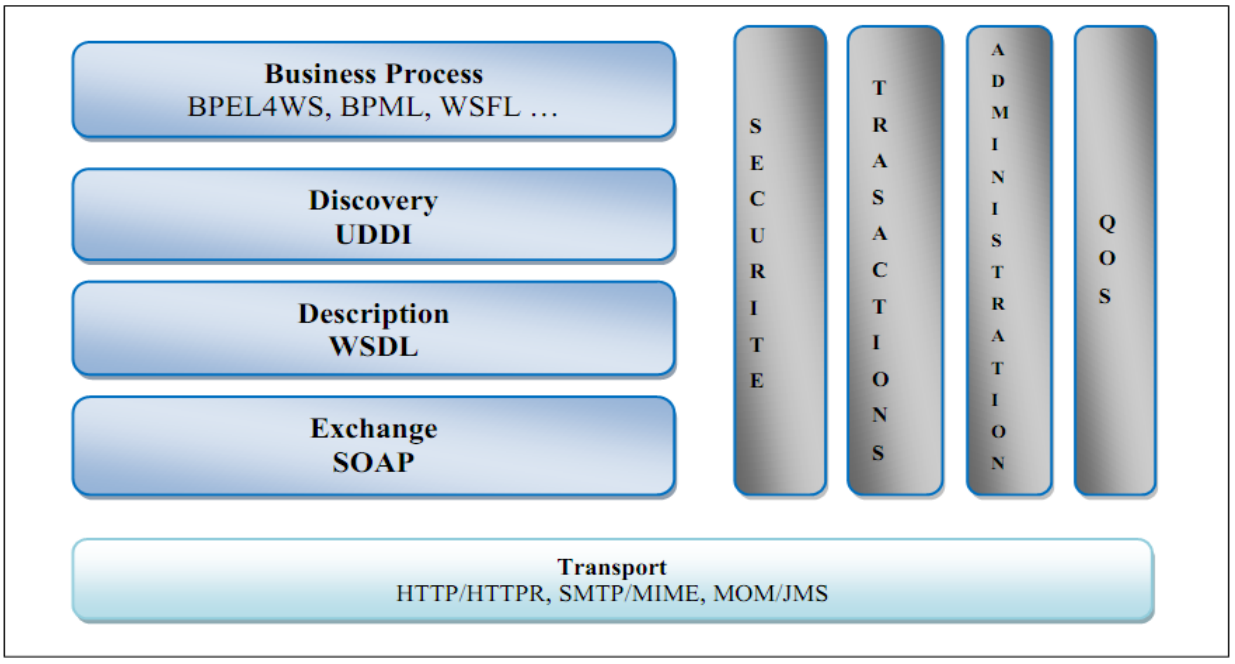

Figure 2 : Cadre architectural des Web Services **[Pat04]**

Chaque couche de la pile d'un Web Service répond à des préoccupations fonctionnelles différentes telles que la sécurité, la messagerie fiable, les transactions, le routage, le workflow etc. comme représenté sur la figure, dans un souci d'interopérabilité les différentes couches de la pile des Web Services s'interface avec des standards. Les fonctions de couches supérieures reposent sur celles de couches inférieures. La partie gauche représente les standards appliqués à chaque couche.

1-Couche transport :

Cette couche est responsable du transport des messages XML échangés entre les applications. Actuellement, cette couche inclut HTTP, SMTP, FTP, et de nouveaux protocoles tels que BEEP.

2-Couche communication :

Cette couche est responsable du formatage des données échangées de sorte que les messages peuvent être compris à chaque extrémité. Actuellement, deux styles architecturaux totalement différents sont utilisés pour ces échanges de données. Nous avons d'un côté l'architecture orientée opérations distribuées (protocoles RPC) basée sur XML et qui comprend XML-RPC et SOAP et de l'autre côté une architecture orientée ressources Web, REST (Représentationnel State Transfer) qui se base uniquement sur le bon usage des principes du Web (en particulier, le protocole HTTP).

3-Couche description de service :

Cette couche est responsable de la description de l'interface publique du service Web. Le langage utilisé pour décrire un service Web est WSDL qui est la notation standard basée sur XML pour construire la description de l'interface d'un service. Cette spécification définit une grammaire XML pour décrire les services Web comme des ensembles de points finaux de communication (ports) à travers lesquels on effectue l'échange de messages.

4-Couche publication de service :

Cette couche est chargée de centraliser les services dans un registre commun, et de simplifier les fonctionnalités de recherche et de publication des services Web. Actuellement, la découverte des services est assurée par un annuaire UDDI (Universal Description Discrovery and Integration).

6. Les standards et protocoles des Web Services : [SCA10]

Les services web sont basés sur trois technologies qui ont émergé comme standards Internet pour assurer l'interaction entre les opérations de recherche, lien, et publication des services web. Un ensemble de spécifications considérées comme des standards ont été définies par le consortium W3C : SOAP, WSDL, UDDI.

Toutes ces technologies sont basées sur le meta-langage XML. Dans ce qui suit, nous présenterons d'abord le rôle de XML dans les services web, puis nous détaillerons les standards cités ci-dessus :

6.1. XML (eXtensible Markup Language) :

Le langage XML (eXtensible Markup Language) est un langage balisé, issu d'une recommandation W3C (WorldWideWeb Consortium), ayant pour but d'encoder tout type de données, indépendamment du code machine de celle-ci. Il a été développé dans le but de partager facilement des données entres différents systèmes et en particulier à travers un réseau type Internet.

En général, XML est un standard qui permet d'écrire des données et des documents structurés transportable sur les protocoles de l'Internet (ex HTTP). Il représente la technologie de base des architectures des services web. En effet, XML rend les services web indépendant de plate-forme et langage de développement, ce qui permet leur interopérabilité.

6.2. SAOP (Simple Object Access Protocol) :

SOAP fournit une structure d'emballage standard XML, permettant de transporter des documents sur une variété de technologies Internet, comprenant HTTP, FTP. En fait ce protocole de communication permet l'interopérabilité des applications à travers Internet. Interopérabilité implique qu'un programme opérant sur un système fonctionne également sur un autre système.

6.2.1 Les principes de SOAP :

SOAP est un protocole de communication pour l'échange d'informations dans un environnement décentralisé et distribué. SOAP encourage le partage des données entre machines qui seront capables d'analyser facilement et correctement ces mêmes données.

Selon [W3C], le protocole est composé de trois parties :

- ✓ Une enveloppe qui définit un canevas pour décrire le contenu du message et comment le traiter.
- ✓ Un jeu de règles de codage à exprimer les cas de types de données défini par l'application.
- ✓ Une convention pour représenter des appels de procédure éloignés (RPC) et ses réponses.

6.2.2. Structure d'un message SOAP :

SOAP définit trois éléments composants un message :

L'enveloppe (Enveloppe), l'en-tête du message (Header) et le corps du message (Body).

- **Enveloppe SOAP :**

L'enveloppe SOAP est l'élément racine d'un message SOAP (obligatoire dans un message SOAP), il contient les deux autres parties de ce message (Header et Body).

L'enveloppe définit l'espace des noms standards (namespace : URI permettant de connaître la provenance de chaque balise), précisant la version supportée de SOAP et permet de différencier entre les schémas ex :
"http://schemas.xmlsoap.org/soap/envelope/".

- **L'En-tête SOAP :**

L'en-tête est un élément facultatif dans un message SOAP, il doit être le premier enfant (Child élément) de l'élément Enveloppe, marqué par la balise <Header>.il contient des données supplémentaires transmises aux services extérieurs, Ces données peuvent être des informations d'authentification, de gestion de transactions, etc.

- **Le corps SOAP :**

Le corps du message SOAP est obligatoire, marqué par la balise <Body>. Il Contient le message réel destiné au receveur.

Un message SOAP peut aussi contenir un ou plusieurs attachements (document, images, etc.) à transmettre avec le message. Ces données ne sont pas représentables en XML. De ce fait, SOAP utilise un mécanisme d'inclusion appelé MIME (Multipurpose Internet Mail Extensions), cette méthode est répandue pour transmettre des documents autres que du texte dans des courriers électroniques.

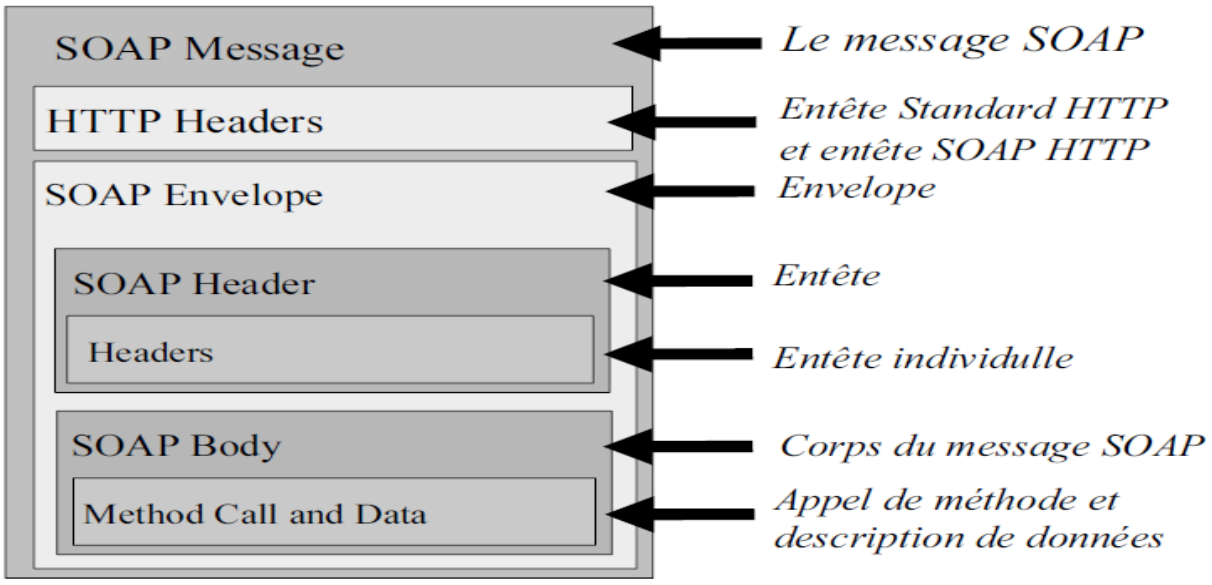

Figure 3 : La structure d'un message SOAP**. [ISW]**

En résumé, le protocole SOAP permet la communication entre des applications par l'échange de messages au format XML, en se basant sur des appels procédures à distance (RPC) et en utilisant HTTP comme protocole de communication. Il est indépendant de tout langage, plate-forme, et système d'exploitation et ne fournit aucune description sur la façon d'accéder aux services web.

6.3. L'annuaire UDDI (Universal Description, Discovery and Integration):

L'annuaire des services UDDI est un standard pour la publication et la découverte des informations sur les services Web. La spécification UDDI est une initiative lancée par ARIBA, Microsoft et IBM. Cette spécification n'est pas gérée par le W3C mais par le groupe OASIS. La spécification UDDI vise à créer une plate-forme indépendante, un espace de travail (framework) ouvert pour la description, la découverte et l'intégration des services des entreprises.

6.3.1.Consultation de l'annuaire :

L'annuaire UDDI se concentre sur le processus de découverte de l'architecture orientée services (SOA), et utilise des technologies standards telles que XML, SOAP et WSDL qui permettent de simplifier la collaboration entre partenaires dans le cadre des échanges commerciaux. L'accès au référentiel s'effectue de différentes manières :

- **Les pages blanches :** comprennent la liste des entreprises ainsi que des informations associées à ces dernières (coordonnées, description de l'entreprise, identifiants...).
- **Les pages jaunes :** recensent les services Web de chacune des entreprises sous le standard WSDL.
- **Les pages vertes :** fournissent des informations techniques précises sur les services fournis.

Les entreprises publient les descriptions de leurs services Web en UDDI, sous la forme de fichiers WSDL. Ainsi, les clients peuvent plus facilement rechercher les services Web dont ils ont besoin en interrogeant le registre UDDI.

Lorsqu'un client trouve une description de service Web qui lui convient, il télécharge son fichier WSDL depuis le registre UDDI. Ensuite, à partir des informations inscrites dans le fichier WSDL, notamment la référence vers le service Web, le client peut invoquer le service Web et lui demande d'exécuter certaines de ses fonctionnalités.

6.3.2. Structures de données UDDI :

Le scénario classique d'utilisation d'UDDI est illustré ci-dessous. L'entreprise B a publié le service Web, et l'entreprise A est client de ce service :

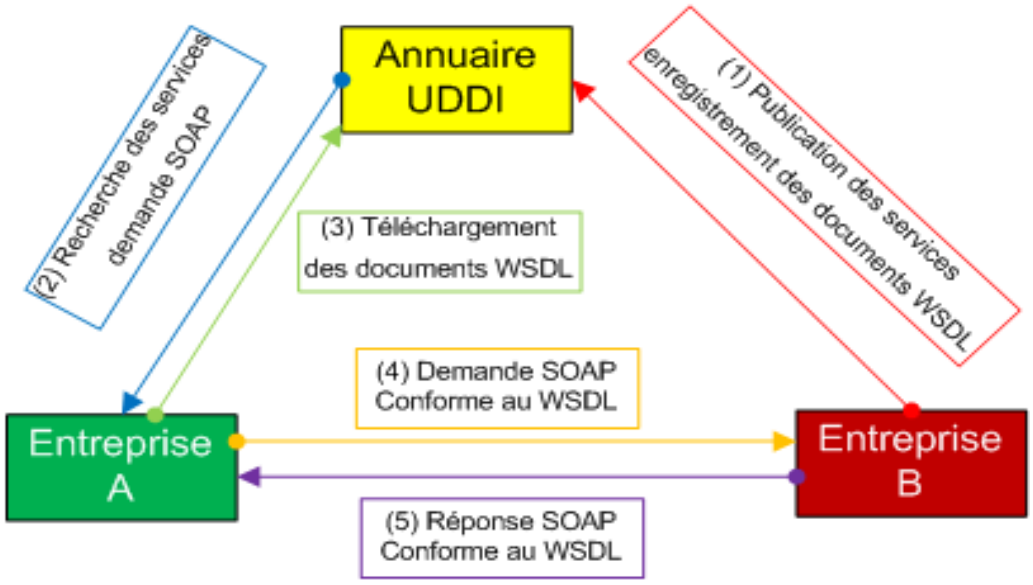

Figure 4 : Le scénario classique d'utilisation d'UDDI. **[SDZ]**

6.4. Le langage WSDL (Web Services Description Language) : [SCA10]

Le protocole SOAP met à la disposition des services Web un moyen standard de communication à travers les échanges de messages XML, mais ne fournit aucune information sur la structure que le message doit respecter et son contenu vis à vis du service web. C'est pourquoi la spécification WSDL est nécessaire pour décrire l'interface d'utilisation de services web. Grâce à WSDL les services Web sont faiblement couplés et autonomes.

6.4.1 Définition :

Le langage de description WSDL est une proposition conjointe des sociétés ARIBA, IBM et Microsoft auprès du W3C. WSDL est un langage de description des capacités de services Web basé sur XML. Un document WSDL décrit essentiellement le nom de la méthode utilisé, son nombre de paramètres, et leur type, ce qu'un service Web offre, où il réside et comment on peut l'invoquer. Nous pouvons dire que les services Web sont auto descriptifs.

Un document WSDL est un fichier XML qui définit des services comme une collection de points finaux (end points) de réseau ou des ports (W3C). Il décrit en détail comment utiliser un service web. Il est actuellement à sa version 2.0.

6.4.2. Structure d'un document WSDL :

Un document WSDL, est constitué d'une suite d'éléments décrivant un service Web sous forme d'un ensemble d'opérations exploitables depuis l'extérieur. Il est composé de deux niveaux : un niveau abstrait constitué d'éléments qui définissent l'interface des services web tel que les types de données, les messages, les types de ports. Et un second niveau appelé concret constitué des éléments qui décrivent des informations liées à un usage contextuel du service web.

❖ WSDL 2.0 :

En 2002, le consortium W3C inscrit une nouvelle activité de recherche dans ses préoccupations: l'activité sur les services Web (Web Services Activity). De cette activité découlent quatre groupes de travail dont le groupe de travail sur la description de services Web (Web Services Description Working Group). Ce groupe a été créé principalement en vue de standardiser WSDL qui né en 2002, date de fondation de ce groupe de travail, qu'une note. En 2007, WSDL dans sa version 2.0 est standardisé. La suite de cette standardisation, les études du groupe de travail sur la description de services Web ont pris fin, l'objectif final du groupe étant atteint. Sa structure générale est composée d'un élément racine (description) et de quatre sous-éléments obligatoires (types, interface, binding et service). La figure suivante illustre la structure générale de ce document.

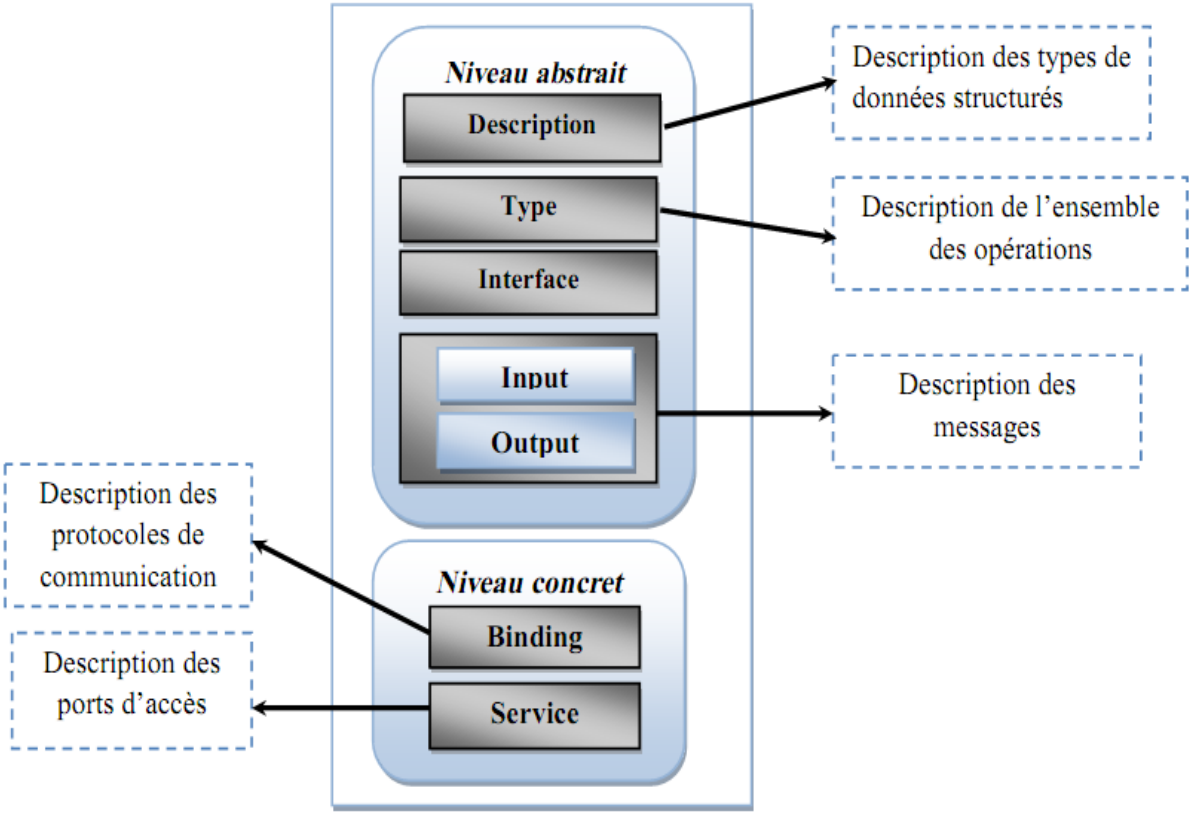

Figure 5 : Structure générale d'un document WSDL 2.0

- **Description :** C'est l'élément racine d'un document WSDL 2.0, il est utilisé afin de déclarer les espaces de nom utilisés tout au long du document.
- **Types :** Cet élément décrit les types de messages que le service envoie et reçoit lors de l'appel d'une des méthodes. Cet élément est analogue à l'élément homonyme utilisé dans la version 1.1, à la différence que des espaces de noms nécessaires à la définition de la structure de données sont inclus à ce niveau du document.
- **Interface :** Cet élément décrit l'ensemble des fonctionnalités, appelées opérations, fournies par le service Web. Une opération (sous-élément opération) représente une simple interaction entre le client et le service. Dans la version 2.0 de WSDL les messages sont définis au niveau de l'opération et sont de deux types : soit des messages que le service reçoit (sous-élément input), soit des messages que le service envoie au client (sous-élément output). De plus, l'élément opération possède un attribut pattern qui définit la séquence selon laquelle les messages sont transmis.
- **Binding :** Cet élément décrit comment accéder au service. Son rôle est le même que l'élément du même nom dans la version 1.1. L'espace de nom définissant le protocole à utiliser est défini comme attribut de cet élément.
- **Service :** Cet élément définit la localisation du service Web décrit. Pour chaque interface décrite, un élément service lui est associé. Le sous-élément « endpoint » définit un port d'accès en référençant l'élément « binding » associé et en déclarant l'URL localisant le service (avec l'attribut adresse). Ceci permet qu'une interface d'un service possède plus d'une localisation (plus d'un élément « endpoint ») pour répondre aux problèmes d'indisponibilité.

En résumé, la description WSDL joue un rôle important dans l'interopérabilité des composants services web, et indispensable à leurs déploiement. Elle sépare clairement la définition abstraite du service (échange de messages) de ses mécanismes de liaison (définition des protocoles applicatifs. la publication et la recherche d'un service web au sein de l'annuaire UDDI se font via deux types de documents WSDL : un document pour l'interface du service web et un autre pour son implémentation.

7. Conclusion :

Dans ce chapitre, nous avons appris que les services web permettent à des applications de dialoguer via Internet, par l'échange de messages fondés sur des standards, et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent.

Nous avons également exploré les différentes couches de protocoles des services Web, qui mettent en œuvre des technologies telles que HTTP, XML, SOAP, WSDL et UDDI.

Chapitre 2
Le e-commerce

1. Introduction :

Le e-commerce est devenu un composant essentiel pour les stratégies d'affaires et un catalyseur du développement de l'économie mondiale.

Le e-commerce permet aux gens de payer les factures, commander des articles, et faire d'autres activités, de chez eux depuis leurs bureaux comme il offre aux petites et moyennes entreprises plus de chance de côtoyer les grosses compagnies.

L'intégration des nouvelles technologies de l'information et de la communication dans le monde du e-commerce a permis d'améliorer la productivité, d'encourager une consommation massive importante, sans parler de la réduction des coûts.

Dans ce chapitre, nous présentons le e-commerce actuel en général (historique, définition, types, systèmes de paiements électronique, composants) pour en déduire les avantages de celui- ci et ses limites. Puis, nous parlerons comment la technologie des web service améliore le e-commerce. **[AMO09]**

2. Historique : [AMO09]

La première forme du e-commerce a été le magasin **On-Line** où les vendeurs communiquaient avec les consommateurs via leurs sites web en utilisant les méthodes de marketing traditionnelles.

C'est en 1960, qu'est apparu l'**EDI** (Electronic Data Interchange) afin d'échanger des documents dans un format standard, entre les entreprises. Puis, l'**EFT** (Electronic Funds Transfer) a été utilisé par les banques pour la transmission d'ordre de paiement.

3. Définition : [AMO09]

Plusieurs définitions ont été données au e-commerce, mais pour en donner une assez complète, on peut dire que, le E-commerce est l'utilisation des technologies de l'information et de la communication (les TIC), pour créer, transformer, et redéfinir des relations commerciales entre des organisations, et entre organisations et individus.

4. Les différents types du e-commerce : [AMO09]

Le e-commerce se divise en 6 différents types selon le déroulement des échanges d'information entre :

4.1. Deux entreprises (B2B Business-to-Business):

90% du e-commerce est de ce type, il regroupe les échanges des informations cryptées entre entreprises pour en préserver la confidentialité, les comptes d'entreprises, les échanges de biens et de services entre entreprises, fournisseurs et détaillants.

Le B2B permet de mieux cibler les entreprises clientes. Cependant, il exige un niveau de sécurité plus avancé que les autres types, car les achats des compagnies sont plus volumineux que ceux des particuliers consommateurs.

Les applications B2B couvrent généralement dans les domaines de gestion des stocks, gestion de la distribution, gestion des systèmes de payement électronique.

Les exemples des modèles les plus connus et les plus réussis du B2B sont : IBM, HP, Dell et Cisco.

La plupart des experts prévoient que le B2B continue à se développer plus rapidement que le B2C (Business-to-Consumer).

4.2. Une entreprise et un client (B2C Business-to-Consumer) :

C'est le premier type de e-commerce à avoir vu le jour, il permet aux clients de faire des achats via Internet des biens physiques (livres, produits alimentaires,..) ou numériques (E-Books, logiciels, musiques…).

Avec la crainte qui persiste encore chez les consommateurs, ce type de commerce n'a pas encore pris tout son essor.

Les applications B2C les plus répandues sont dans le domaine de vente de produits et d'information, gestion financière personnelle.

Un bon exemple de modèle B2C est « Amazon.com ».

4.2.1 Fonctionnement de B2C :

1. Les visiteurs se connectent sur le site.
2. Ils examinent les produits.
3. Ils ajoutent les produits à leurs chariots virtuels d'achats.
4. Ils confirment leurs achats.
5. La commande est crée dans la base de donnée.
6. Le paiement est autorisé.
7. Les commandes sont envoyées à la logistique.
8. Les commandes sont effectuées par la logistique (Le paiement est effectué).

4.3. Une entreprise et un gouvernement (B2G Business-to-Government) :

Connu aussi par l'appellation E-Government, est l'ensemble des agences gouvernementales sur des sites Web, servant à organiser l'évolution du e-commerce, en permettant aux entreprises et aux particuliers de se renseigner, de faire des déclarations en douane et d'impôts, obtenir des autorisations (permis, licences…).

Actuellement, le B2G n'est pas un domaine très avancé. Beaucoup de projets sont encore en phase d'étude dans plusieurs pays.

4.4. Une administration et un client (A2C Administration-to-Consumer) :

Regroupe toutes les transactions entre les organisations administratives et les particuliers. Toutefois, avec le développement du B2G et du B2C, l'interactivité des administrations pourrait s'étendre.

4.5. Deux clients (C2C Consumer-to-Consumer) :

C'est une forme d'e-commerce qui a connu un succès ces dernières années, c'est le commerce entre individus ou consommateurs.

Les exemples les plus connus de ce type d'entreprise sont **« eBay »** aux Etats-Unis ou **« iBazar »** en France, qui permettent à leurs clients de vendre des objets par des enchères publiques, on peut citer aussi les sites de publication de petites annonces où les intéressés peuvent acheter et négocier avec les vendeurs.

4.6. Une entreprise et ses employés (B2E Business-to-Employee) :

Via la mise à disposition de formulaires à leurs attentions pour la gestion de leurs carrières, de leurs congés ou de leurs relations avec le comité d'entreprise.

5. Systèmes de paiement électronique : [AMO 09]

Du point de vue conceptuel, les systèmes de paiement se divisent en deux catégories principales :

5.1. Paiement sans intermédiaire :

Le client remplit un formulaire sécurisé et donne son numéro de carte bancaire crypté, le vendeur récupère les informations cryptées et effectue lui-même des transactions avec la banque via son Terminal de Paiement Electronique (TPE).

5.2. Paiement avec intermédiaire :

C'est la banque qui prend en charge les étapes de transaction, le vendeur ne voit jamais les informations associées au client.

Il existe différents moyens de paiement électronique, on en cite : Carte de crédit, chèques électroniques, argent liquide électronique, le porte-monnaie électronique.

6. Composants du e-commerce : [AMO 09]

Le commerce électronique est la rencontre virtuelle d'intervenants (entreprises, acheteurs, vendeurs, recherchistes…) qui avec leurs équipements, logiciels, produits et services, culture, ressources et leur site Internet désirent faire des échanges.

Les variables composant le e-commerce sont multiple et sont :

- **Production :** Tout ce qui concerne la gestion des produits, achats et des stocks.
- **Marketing :** C'est la gestion des relations avec la clientèle (prix, communication…)
- **Esthétisme et convivialité :** C'est les IHMs et la gestion des designs.
- **Logiciel :** C'est l'ensemble des applications constituant le système informatique.
- **Equipements :** C'est la couche physique du système (matériel informatique).

Les composants d'une e-transaction sont le **vendeur** (ayant besoin d'un site web, un intranet et des employés), les **institutions bancaires** (traitant les paiements), les **sociétés de transport** (assurant le transport des biens physiques) et **une autorité de certification**

(Certificat numérique avec une clé publique au vendeur pour l'authentification des transactions), les **consommateurs particuliers** (dans le cas d'une transaction B2C), les **compagnies clientes** (dans le cas d'une transaction B2B) et enfin les **pouvoirs publics** (protégeant le vendeur et les consommateurs).

7. Les web services et leur impact sur le e-commerce (B2B) : [IWS 03]

Les Web Services devraient permettre aux entreprises de :

- ✓ Donner aux clients un accès direct à l'information, aux données et aux fonctionnalités dont ils ont besoin pour interagir avec une entreprise.
- ✓ Donner aux partenaires d'une entreprise un accès direct à la fonctionnalité dont ils ont besoin pour mieux servir les clients qu'ils ont en commun avec cette entreprise.
- ✓ Donner aux fournisseurs d'une entreprise un accès direct à l'information et la fonctionnalité dont ils ont besoin pour leur permettre d'ajuster les inventaires selon le modèle « juste à temps ».
- ✓ Intégrer des applications de bout en bout, de manière abordable, facile à implanter autant hors des frontières de l'entreprise qu'à l'intérieur de celles-ci.
- ✓ Avoir des équipes de développement travaillant indépendamment et efficacement sur des systèmes qui interagiront, parce que ces équipes travailleront à développer des interfaces communes plutôt que d'avoir à synchroniser les processus.

Les Web Services offrent donc aux entreprises la flexibilité de réponse et d'anticipation des besoins changeant des clients, la rationalisation des infrastructures logiciels et la flexibilité d'interaction et de configuration des alliances externes avec les partenaires et fournisseurs.

8. Les avantages des Web Services :

L'idée fondamentale derrière les Web Services est de morceler les applications et les processus d'affaires en morceaux réutilisables appelés « Service » de sorte que chacun de ces segments effectue une tâche distincte. Ces services peuvent alors servir à l'intérieur et à l'extérieur de l'entreprise, facilitant l'interopérabilité entre tous ces services.

De par leur nature, les Web Services:

- Permettent à des portions de logiciels écrits dans différents langages, ou évoluant sur différents systèmes d'exploitation, de communiquer entre elles facilement et à peu de frais .
- Permettent à des applications supportant différents processus d'une organisation ou de différentes organisations, de communiquer entre elles et/ou d'échanger des données facilement et à peu de frais.

9. L'explosion du nombre de partenaires :

Une problématique à laquelle font face les entreprises se tournant vers Internet est le nombre croissant de partenaires potentiels. Cette croissance est souvent liée à une augmentation des coûts liés à l'élaboration et le déploiement des interfaces entre les systèmes d'information de ces partenaires. En particulier, trois défis se posent :

- **Distribution des centres de contrôles :** Les entreprises peuvent dicter l'utilisation d'une plate-forme homogène à l'intérieur de leurs frontières. Ils peuvent même obliger certains de leurs fournisseurs à s'adapter à cette plate-forme s'ils ont une position dominante déterminante. Cependant, lorsque le nombre et la diversité des partenaires augmentent, il devient difficile de maintenir un seul centre de contrôle.

- **Diversité des plates-formes technologiques :** Sans un centre de contrôle unique, les entreprises se battent continuellement avec la diversité croissante des plates-formes qu'ils ont à brancher. Ces branchements se doivent aussi d'être abordables et réalisables pour les PME qui doivent aussi supporter les coûts de ses branchements.

- **L'environnement dynamique :** Dans un monde économique en mouvance perpétuelle, les entreprises se doivent d'être capable d'intégrer les nouveaux partenaires à leurs systèmes informatiques et ce, de façon efficace, rapide et économique. Ils doivent aussi avoir la flexibilité d'abandonner certaines alliances d'affaires sans avoir à radier de leurs bilans des dépenses et investissements technologiques.

En réponse à ces défis, les Web Services offrent les solutions suivantes :

- ✓ **La simplicité :** Les Web Services réduisent la complexité des branchements tout en rendant la tâche plus facile aux nouveaux participants. Cela se fait en ne créant la fonctionnalité qu'une seule fois plutôt qu'en obligeants tous les participants à reproduire la fonctionnalité à chacun des bouts (comme avec l'architecture client/serveur).

- ✓ **Composante logicielle légèrement couplée :** L'architecture modulaire des Web Services, combinée au faible couplage des interfaces associées, permet l'utilisation et la réutilisation de services qui peuvent facilement être recombinées à différents autres modules.

- ✓ **Hétérogénéité :** Les Web Services permettent d'ignorer l'hétérogénéité entre les différentes applications et modules. En effet, ils décrivent comment transmettre un message (standardisé) entre deux applications, sans imposant comment construire ce message.

- ✓ **Ouverture :** Les Web Services permettent de réduire les inquiétudes liées aux différents ***«lock-in»*** que les entreprises subissent des fournisseurs informatiques. Ils permettent aussi de tirer une valeur économique supplémentaire des infrastructures informatiques existantes et des plates- formes ouvertes tel que l'Internet.

10. Situation actuelle des Web Services dans les entreprises :

Déjà, plusieurs entreprises américaines ont utilisées les Web Services pour offrir des services à valeur ajoutée. Selon **Michael Vizard** de **InfoWorld**, les impacts des Web Services se font sentir à court terme. Par exemple, il cite le cas de **Merrill Lynch** qui a réussi un projet d'intégration à un coût de $30 000 plutôt que les $800 000 initialement prévus, grâce à l'utilisation des Web Services.

Chez Dell, on se sert des Web Services afin de rendre disponible à toutes les 2 heures les horaires de production, permettant à ses fournisseurs à leur tour d'ajuster leurs livraisons. Cette initiative a permis à Dell de réduire le temps de détention sur stock de 26-30 heures à 3-5 heures, soit une réduction de 80%.

Plusieurs autres entreprises ont aussi eu des résultats très probants en utilisant cette technologie : Amazon, Continental Airlines, Fedex General Motors…etc.

Parmis ces entreprises, certaines ont commencé à exposer des Web Services hors des frontières de l'entreprise, pour leurs différents clients et partenaires. Cela veut dire que les Web Services changent déjà l'horizon B2B.

11. Conclusion :

À notre avis, les Web Services sont suffisamment développés pour que les entreprises les utilisent dès maintenant, afin de récolter les divers bénéfices de la technologie. L'utilisation rapide de la technologie permettra à ces entreprises de développer à l'interne une compréhension et une expertise, qui pourra alors être utilisée à l'externe une fois que tous les standards seront déterminés.

Par le biais des Web Services, ces entreprises pourront assouplir les liens entre leurs applications et celles de leurs partenaires, créant des systèmes légèrement couplés et plus flexibles. Cet assouplissement doit passer par une modélisation en profondeur des actifs logiciels de l'entreprise afin de bien identifier les services et ressources qu'elle doit déployer. En procédant de cette façon, l'entreprise optimise les avantages des Web Services; elle identifiera les applications, informations et processus d'affaire à valeur ajoutée qu'elle aura avantage à rendre disponible à ses partenaires.

Partie II
Conception et réalisation

Chapitre 3
Conception du système

1. Présentation de notre travaille:

Ce chapitre présente de manière globale l'étude de cas que nous allons développer tout au long de ce projet, un site de commerce électronique, spécialisé dans la vente des matériels informatiques. Afin de décrire les besoins de la société **DZ-INFO**, nous utiliserons des diagrammes de cas d'utilisation et d'activité UML.

Cette société vend des matériels informatiques de compagnie. Elle continue d'exercer son métier tel qu'elle le faisait à ses débuts, c'est-à-dire qu'elle répertorie ses clients et ses articles sur des fiches de papier, reçoit les commandes par fax, les chèques par courrier puis envoie le bon de commande au client. Une fois le chèque encaissé par la banque **DZ-Bank**, elle utilise la société de transport **DZ-TRANS** pour acheminer les matériels vers leurs clients.

Vu l'augmentation du nombre de leurs clients, **DZ-INFO** n'arrive plus à gérer manuellement la vente et souhaite créer un système informatique pour lui permettre de faire face à sa récente croissance. Elle attend de celui-ci qu'il lui permette de vendre ses matériels en ligne, de gérer son catalogue de produits et sa base de données de clients. De plus, ses partenaires (la banque **DZ-Bank** et la société de transport **DZ-TRANS**) souhaitent avoir la possibilité d'échanger des données aux formats électroniques via Internet.

Ce système informatique est baptisé « **DZ-INFO** ». Il doit répondre à certains besoins en termes de performance et de robustesse comme la haute disponibilité puisque le site doit être accessible 24h/24 7j/7, et supporter un nombre élevé de visiteurs.

2. La modélisation :

2.1. Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation ci-après décrit les besoins de la société DZ-INFO de façon synthétique et peut être lu comme ceci : « Un employé peut gérer les articles du catalogue, gérer les clients, visualiser et supprimer les commandes. Un visiteur peut se créer un compte, visualiser et rechercher un article dans le catalogue... ».

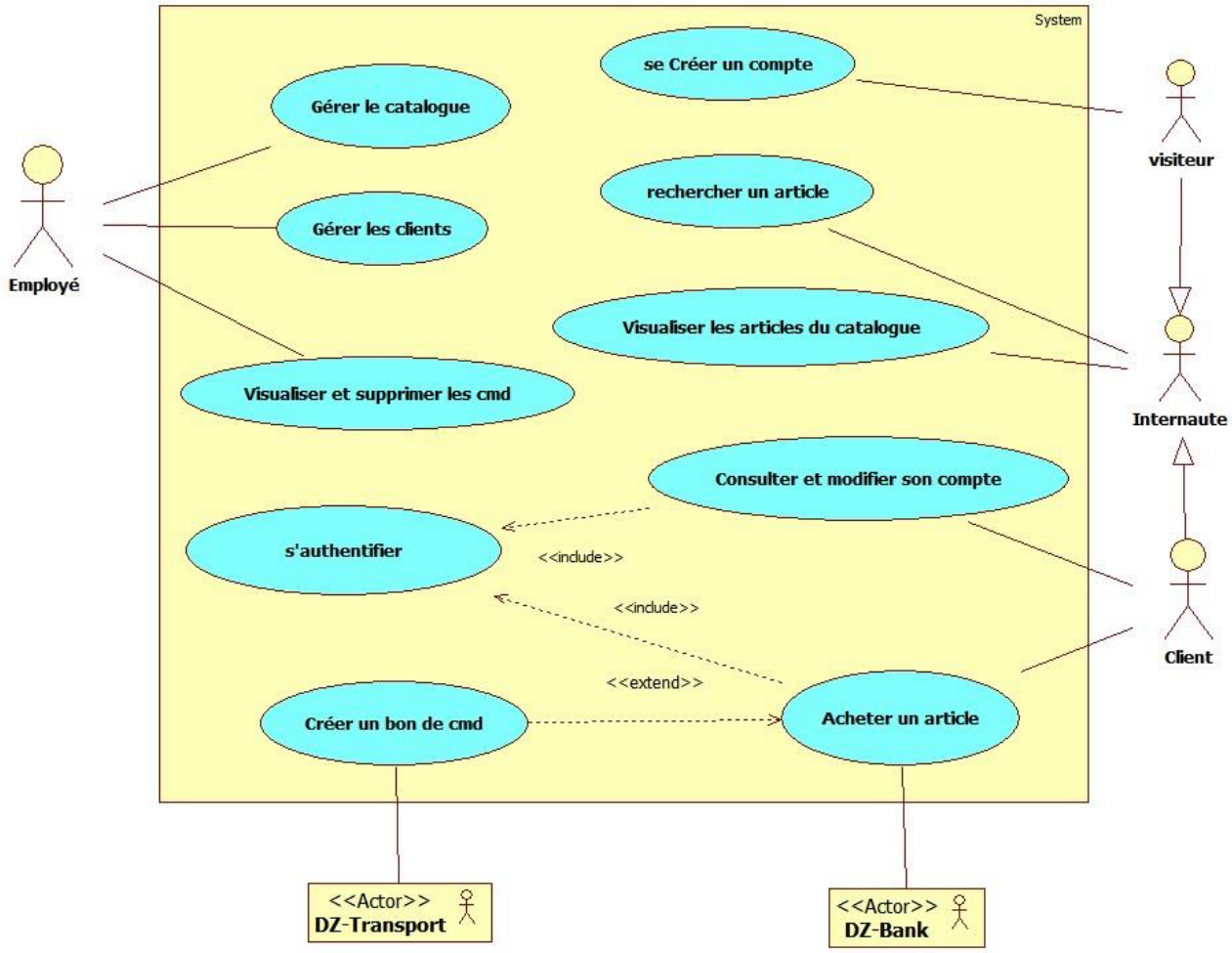

Figure 6 : Diagramme de cas d'utilisation du système.

2.1.1 Les acteurs du système :

Les acteurs humains qui utilisent le système sont les suivants :

- **Employé :** les employés de la société **DZ-INFO** s'occupent de mettre à jour le catalogue des articles ainsi que la liste des clients. Ils peuvent aussi consulter les commandes passées en ligne par les clients.
- **Visiteur :** il s'agit d'une personne anonyme qui visite le site pour consulter le catalogue des matériels. Si le visiteur veut acheter un matériel, il doit d'abord créer un compte. Il devient alors un client de la société **DZ-INFO**.
- **Client :** un client peut visualiser le catalogue, modifier ses coordonnées et acheter des articles en ligne.

Il faut aussi mentionner les systèmes informatiques externes, utilisés par la société **DZ-INFO** :

- **DZ-BANK :** DZ-INFO délègue la validation des cartes bancaires à la banque **DZ-BANK**.
- **DZ-TRANS :** la livraison des matériels informatique est assurée par la société de transport **DZ-TRANS**. Celle-ci se rend à l'entrepôt de **DZ-INFO**, charge les matériels achetés dans ses camions, puis les achemine chez les clients.

2.1.2. Les cas d'utilisation :

2.1.2.1. Gérer les clients :

- **Résumé :** Permet à un employé de créer/modifier/supprimer/rechercher/visualiser un client.
- **Acteur :** Employé.
- **Description :**

DZ-INFO veut pouvoir créer ses clients dans le système à partir des données existantes. Elle souhaite également pouvoir les modifier, les supprimer et les rechercher. Les éléments caractérisant un client sont les suivants :

- Identifiant unique du client.
- Login et mot de passe utilisés par le client pour se connecter à l'application.
- Prénom et nom de famille.
- Numéro de téléphone où l'on peut joindre le client et son adresse mail.
- Adresse postale : deux zones permettent de saisir l'adresse du client.
  Le premier est obligatoire, la deuxième optionnelle
- Pays de résidence, ville, état et code postal.
- Date de naissance.
- Age du client.

Une fois les données saisies, l'employé souhaite pouvoir les exploiter. Ainsi, à partir d'un identifiant, le système doit donner la possibilité d'afficher les coordonnées du client et proposer à l'employé de les mettre à jour ou de les supprimer. Dans le cas de la suppression, le système doit attendre une confirmation de l'employé avant de supprimer définitivement le client du système.

Le système doit aussi pouvoir afficher la totalité des clients présents dans le système.

- **Exceptions :**
  - Valeur unique **:** Si cette donnée existe déjà dans le système, une exception doit être levée.
  - Donnée obligatoire **:** Si cette donnée est manquante, une exception doit être levée.

2.1.2.2. Gérer le catalogue :

- **Résumé :** Permet à un employé de créer/modifier/supprimer/rechercher/visualiser le catalogue des articles.
- **Acteur :** Employé.
- **Description :**

Le catalogue d'articles de la société **DZ-INFO** est divisé les produits en catégories. Bien qu'elle envisage d'étendre sa gamme. Une catégorie est définie par les données suivantes :

- Identifiant unique de la catégorie.
- Nom (exemple: PC-desktop , PC-laptop , Imprimante, ..etc.)
- Description .

Chacune de ces catégories est divisée en produits. Chaque produit est défini comme suit :

- Identifiant unique du produit.
- Nom.
- La marque (exemple : HP, Sony,….etc.) .
- Description.

Enfin, chaque produit est, à son tour, divisé en articles. Ce sont ces articles qui sont proposés et vendus aux clients. Par exemple, le produit HP620 regroupe les articles suivants (NoteBook , laptop)

Chaque article est défini comme suit :

- Identifiant unique de l'article.
- Nom.
- Prix unitaire de l'article.
- Image : elle représente l'article en question.

➢ **Exceptions :**

- Valeur unique **:** Si cette donnée existe déjà dans le système, une exception doit être levée.
- Donnée obligatoire **:** Si cette donnée est manquante, une exception doit être levée.

2.1.2.3. Visualiser les articles du catalogue :

➢ **Résumé :** Permet de visualiser le contenu du catalogue des matériels informatiques.

➢ **Acteurs :** visiteur, client.

➢ **Description :**

- Les visiteurs et les clients peuvent visualiser la totalité du catalogue des matériels informatiques.
- L'organisation de l'affichage doit être intuitive, c'est-à-dire que le système doit afficher la liste des catégories, à partir des quelles le client choisit un produit puis un article.
- Pour chaque article, une image représentant le produit devra être affichée.
- À tout moment, il doit être possible d'afficher les produits d'une catégorie différente.

2.1.2.4. Rechercher un article :

- **Résumé :** Permet de rechercher un article par son nom.
- **Acteurs :** visiteur, client.
- **Description :**

En plus de visualiser le catalogue de manière linéaire (voir cas d'utilisation « Visualiser les articles du catalogue »), les visiteurs et les clients peuvent rechercher les matériels informatiques contenus dans le système à partir d'une chaîne de caractères.

La recherche ne tient pas compte des minuscules ou majuscules. Si aucun article ne correspond aux critères demandés, une information est affichée au visiteur pour lui indiquer que sa recherche n'a pas abouti et qu'il doit modifier le critère de recherche.

2.1.2.5. Se créer un compte :

- **Résumé :** Permet à un visiteur de se créer un compte dans le système et de devenir ainsi un client.
- **Acteur :** visiteur.
- **Description :**

Pour se créer un compte, le visiteur doit saisir un login (**1)**, un mot de passe et ressaisir une seconde fois son mot de passe (**2)**. Le système lui demande alors de saisir ses coordonnées et informations personnelles (identiques à celles du cas d'utilisation « Gérer les clients »).

- **Exceptions :**
  - **(1)** Le login doit être unique dans le système. Si ce n'est pas le cas, le visiteur doit en être averti et doit en choisir un autre.
  - **(2)** Si les deux mots de passe ne sont pas identiques, une exception doit être levée.
- **Post-conditions :**

Le visiteur est connu du système, il devient client de la société **DZ-INFO.**

2.1.2.6. S'authentifier :

- **Résumé :** Permet à un client de se connecter du système.
- **Acteur :** Client.
- **Pré-conditions :** Le client s'est auparavant créé un compte (cas d'utilisation « Se créer un compte »).
- **Description :**

Le client saisit son login et son mot de passe **(1) (2)**. Il est reconnu par le système, qui affiche alors son nom et prénom. Lorsque le client se déconnecte, il redevient visiteur jusqu'à sa prochaine connexion.

- **Exceptions :**

- **(1)** Si le login n'est pas connu du système, une exception doit être levée.
- **(2)** Si le mot de passe n'est pas le bon, une exception doit être levée.

2.1.2.7. Consulter et modifier son compte :

- **Résumé :** Permet à un client de consulter et de mettre à jour ses informations personnelles dans le système.
- **Acteur :** Client.
- **Pré-conditions :** Le client doit être connecté au système (cas d'utilisation « S'authentifier»).
- **Description :**

Ce cas d'utilisation diffère du cas « Gérer les clients » dans le sens où le client ne peut consulter et modifier que ses données personnelles. Celles-ci sont identiques à celles du cas d'utilisation « Gérer les clients ».

2.1.2.8. Acheter des articles :

- **Résumé :** Permet à un client d'acheter des articles.
- **Acteurs :** Client, DZ-INFO.
- **Pré-conditions :** Le client doit être connecté au système (cas d'utilisation «S'authentifier»).
- **Description :**

Un client visualise le catalogue (voir cas d'utilisation « Visualiser les articles du catalogue ») ou recherche un matériel informatique (voir cas d'utilisation « Rechercher un article »). Lorsqu'il est intéressé par un article, il lui suffit de cliquer sur un lien pour ajouter cet article dans son panier électronique. Cette opération peut être exécutée plusieurs fois sur des articles différents. Le client a ensuite la possibilité de modifier la quantité désirée pour chaque article ou supprimer un ou plusieurs de ces articles du panier. Lorsque la quantité d'un article est inférieure ou égale à zéro, l'article est automatiquement supprimé du panier.

Pendant toute la durée de sa session, le client peut visualiser le contenu de son panier quand bon lui semble. Lorsque le Caddie est vide, un message avertit le client. Sinon, le système affiche la liste des articles avec le nom, la description du produit, la quantité désirée, le prix unitaire et le sous-total (prix unitaire × quantité). Le montant total du panier est également renseigné. Ce Caddie est illimité en taille, un client peut donc acheter autant d'articles qu'il le souhaite. Lorsque le client est satisfait, il valide son panier électronique. Il doit alors saisir les informations de sa carte bancaire ainsi que l'adresse de livraison. Par défaut, l'adresse de livraison est la même que celle du client mais elle peut être modifiée. Les données de la carte bancaire sont les suivantes :

• Numéro de carte bancaire.

• Type de carte bancaire (Visa, MasterCard, …etc.).

• Date d'expiration de la carte bancaire. Le format de cette date est MM/AA, c'est-à-dire deux chiffres pour le mois et deux pour l'année, séparés par le caractère /.

Une fois toutes ces données validées **(1)**, un bon de commande est créé. Le panier électronique est alors automatiquement vidé.

- **Exceptions :**

(1) Les données de la carte bancaire sont validées par **DZ-BANK**. Si la banque rejette la carte bancaire, le client en est averti et peut ressaisir ses données.

- **Post-condition :** Exécuter le cas d'utilisation « Créer un bon de commande ».

2.1.2.9. Créer un bon de commande :

- **Résumé :** Une fois le panier électronique validé par le client, un bon de commande est créé.
- **Acteur :** DZ-TRANS.
- **Pré-conditions :** Le client achète des articles et valide son panier électronique.
- **Description :**

Lorsque le panier électronique du client est validé, le système crée automatiquement un bon de commande. Ce dernier contient toutes les informations nécessaires pour être traité :

• Un numéro de bon de commande.

• La date de création de ce bon de commande.

• Les références du client qui a acheté les articles.

• Les lignes de commande : une ligne de commande référence l'article acheté et sa quantité. Il y a autant de lignes de commande que d'articles contenus dans le panier électronique.

• Les informations de la carte bancaire.

• L'adresse de livraison.

Cette création du bon de commande entraîne plusieurs traitements :

- ✓ Le bon de commande est imprimé puis stocké dans les archives de la société **DZ-INFO.**
- ✓ Toutes les informations nécessaires à l'acheminement des matériels achetés sont envoyées au transporteur **DZ-TRANS** de manière électronique au format XML (Web Service). **DZ-TRANS** livre ensuite les produits aux clients.
- ✓ Un e-mail est envoyé au client pour l'informer du bon déroulement de sa transaction. Cet e-mail contient le numéro du bon de commande ainsi qu'un récapitulatif de son contenu.

2.1.2.10. Visualiser et supprimer les commandes :

- **Résumé :** Permet à un employé de visualiser et de supprimer les commandes présentes dans le système.
- **Acteur :** Employé.
- **Description :**

L'employé peut visualiser la liste des commandes présentes dans le système. Pour chacune de ces commandes, il peut en consulter les informations et les supprimer.

2.2 Diagramme d'activités :

UML permet de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation, à l'aide de diagrammes d'activités.

2.2.1. Diagramme d'activités d'une transaction déroulée avec succès :

Le diagramme d'activités ci-après nous donne la représentation graphique des actions effectuées pour effectuer un achat sur notre site **DZ-INFO** (une transaction s'est déroulée avec succès).

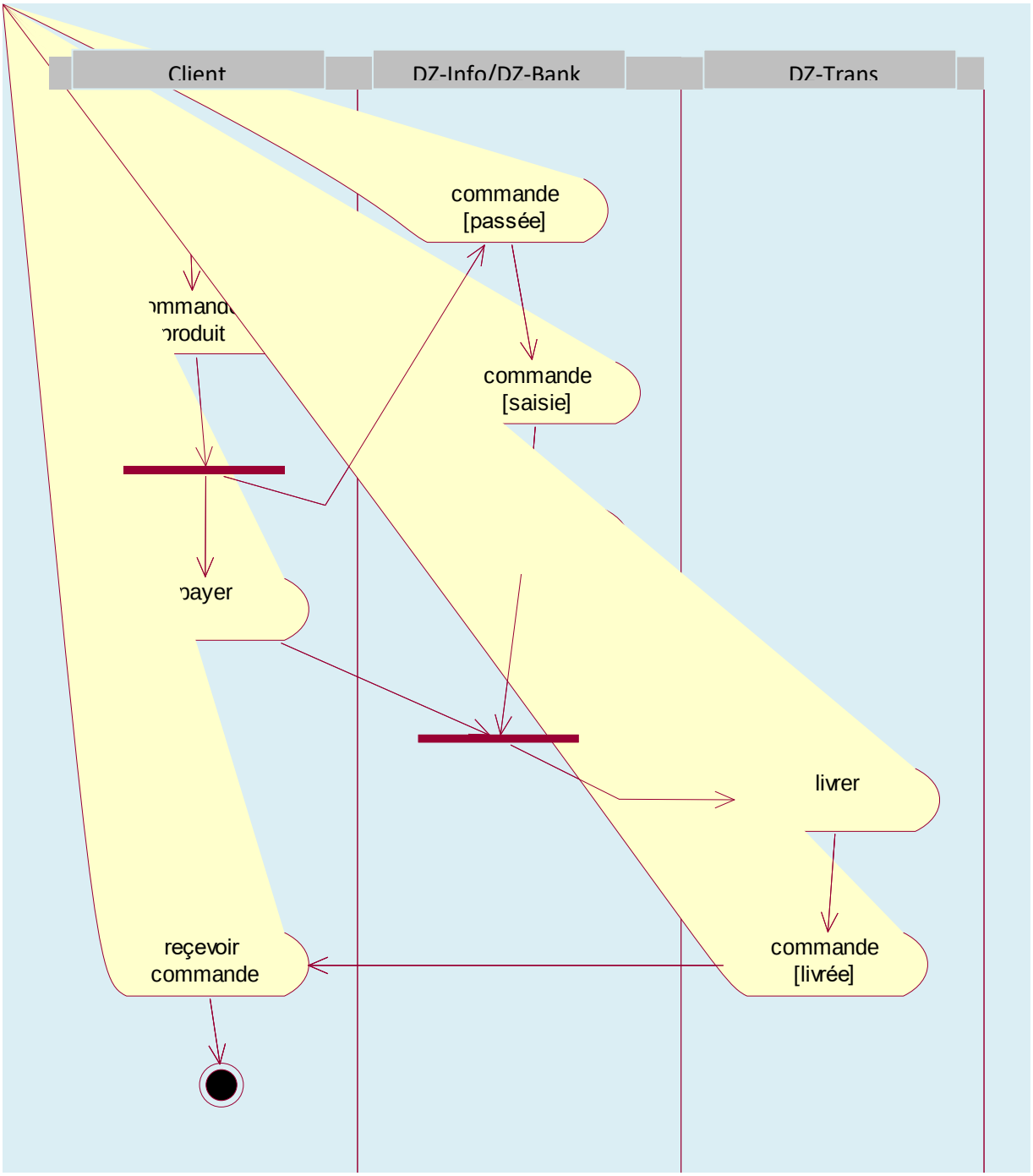

Figure 7 : Diagramme d'activités une transaction s'est déroulée avec succès.

2.2.2. Diagramme d'activité pour gérer le compte client :

- Ci-dessous le diagramme d'activité permet de voir les différents enchaînements possibles pour créer, consulter ou mettre à jour les données du client. Les activités sont différentes selon le cas où le client est connu du système ou pas. Si tel est le cas, la simple saisie de l'identifiant et du mot de passe permet à le client de consulter et de mettre à jour ses données. Sinon, il doit alors saisir un nouvel identifiant et mot de passe qui lui permettra ensuite de renseigner ses informations.

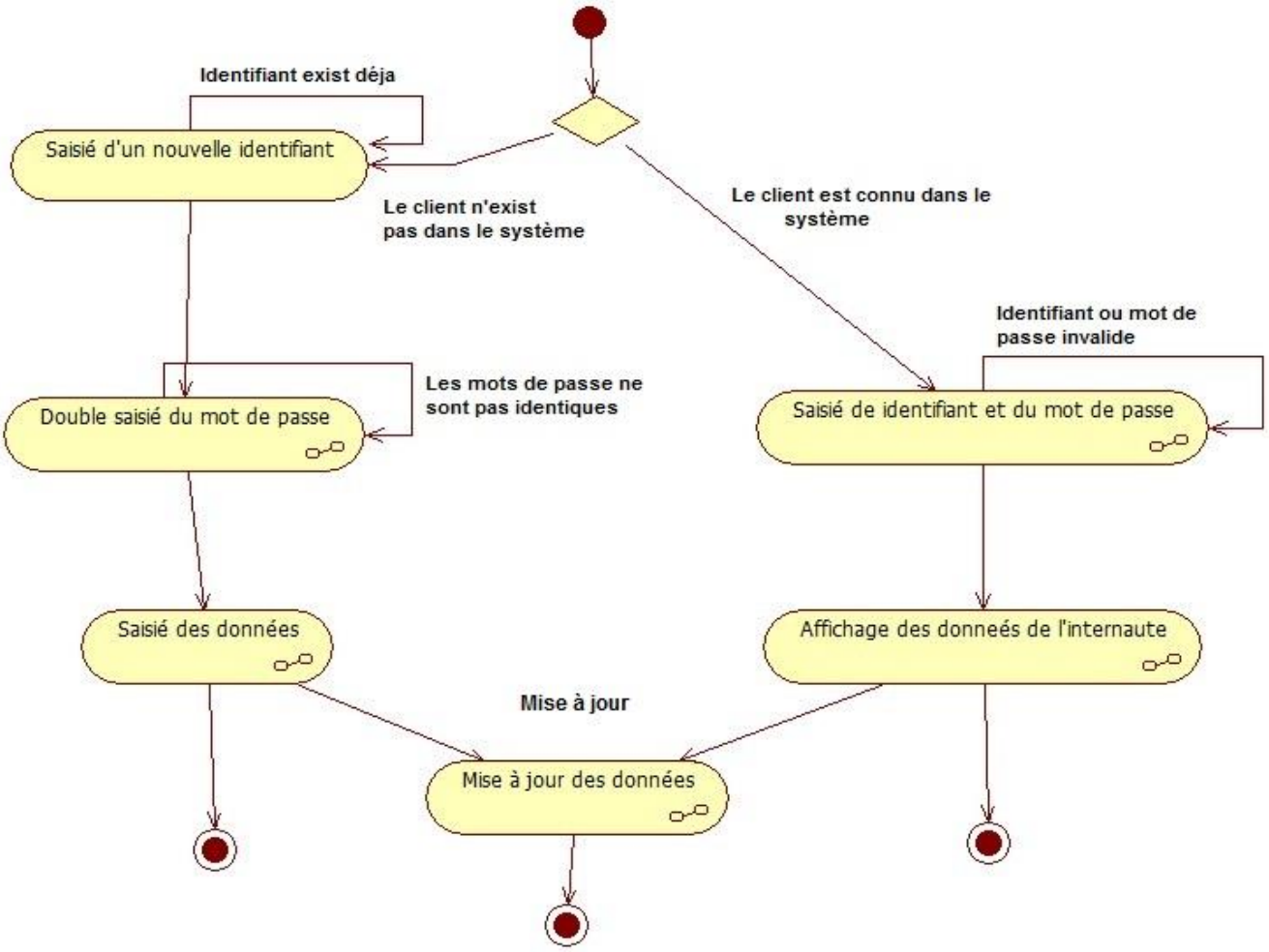

Figure 8: Diagramme d'activité pour créer/modifier le compte client.

2.3. Diagramme de classes :

Un diagramme de classes est une collection d'éléments de modélisation statiques (classes, interfaces, etc.), qui montrent la structure d'un modèle. Il fait abstraction des aspects dynamiques et temporels. Ce diagramme se concentre sur les relations entre classes (association, héritage, etc.), leurs attributs et méthodes.

Le schéma ci-dessous montre le diagramme de classes de notre application :

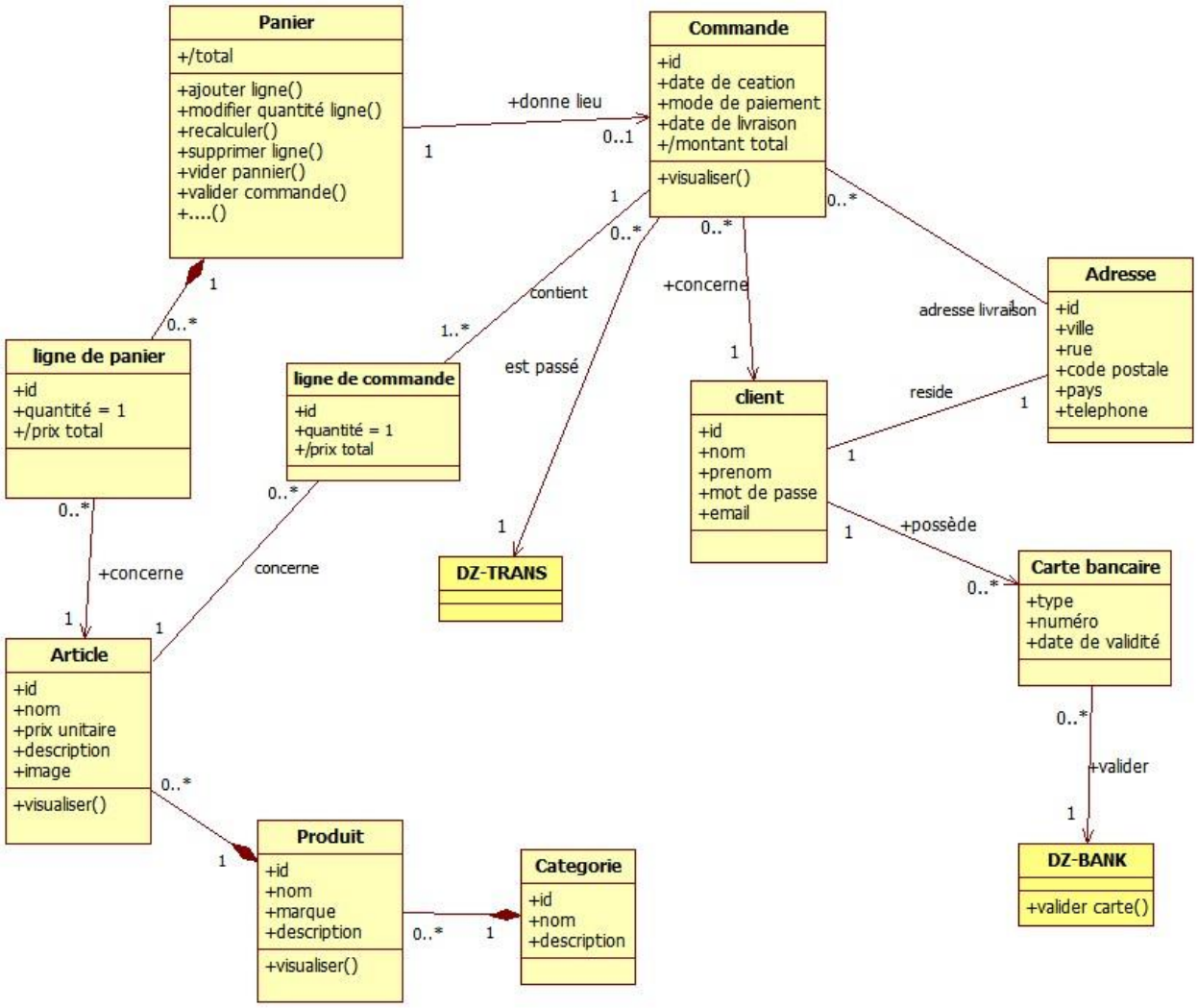

Figure 9 : Diagramme de classe pour la société **DZ-INFO.**

3. Conclusion

Dans ce chapitre, nous avons présenté l'étude de cas de l'application **DZ-INFO** ainsi que les acteurs qui l'utilisent. Le diagramme de cas d'utilisation nous a permis de formaliser les besoins de manière synthétique, puis d'expliciter chaque cas d'utilisation de manière textuelle. Cette application sera réalisée au cours du prochain chapitre.

Chapitre 4
Mise en œuvre

1. Introduction :

Ce chapitre présente les différents outils et techniques informatiques qui ont été utilisés pour la réalisation de notre application. Premièrement, nous allons présenter la plateforme JEE et l'environnement de développement que nous avons utilisés pour développer notre application.

En dernier lieu, On présente l'architecture de notre application avec un aperçu sur les maquettes des interfaces, ainsi que les différentes fonctionnalités de l'application illustrées par des images écrans.

2. Présentation des outils technologies utilisés : [CDP05]

2.1. Java SE :

Avant de parler de Java Enterprise Edition 5 (JEE), il est nécessaire de présenter brièvement le langage sur lequel s'appuie cette plate-forme :

Java 5.0. La version 5 du JSE, ou Java Standard Edition, est une révision majeure du langage Java créé en 1995 par l'équipe de James Gosling. Cette version apporte de nombreuses nouveautés telles que l'autoboxing, les annotations, les génériques, une nouvelle boucle d'itération, les types énumérés, les méthodes à arguments variables, les imports statiques et bien d'autres encore. De nouveaux outils ainsi que de nouvelles API ont vu le jour ou ont été considérablement enrichis comme l'API de concurrence, l'API de thread, la supervision de la JVM, etc.

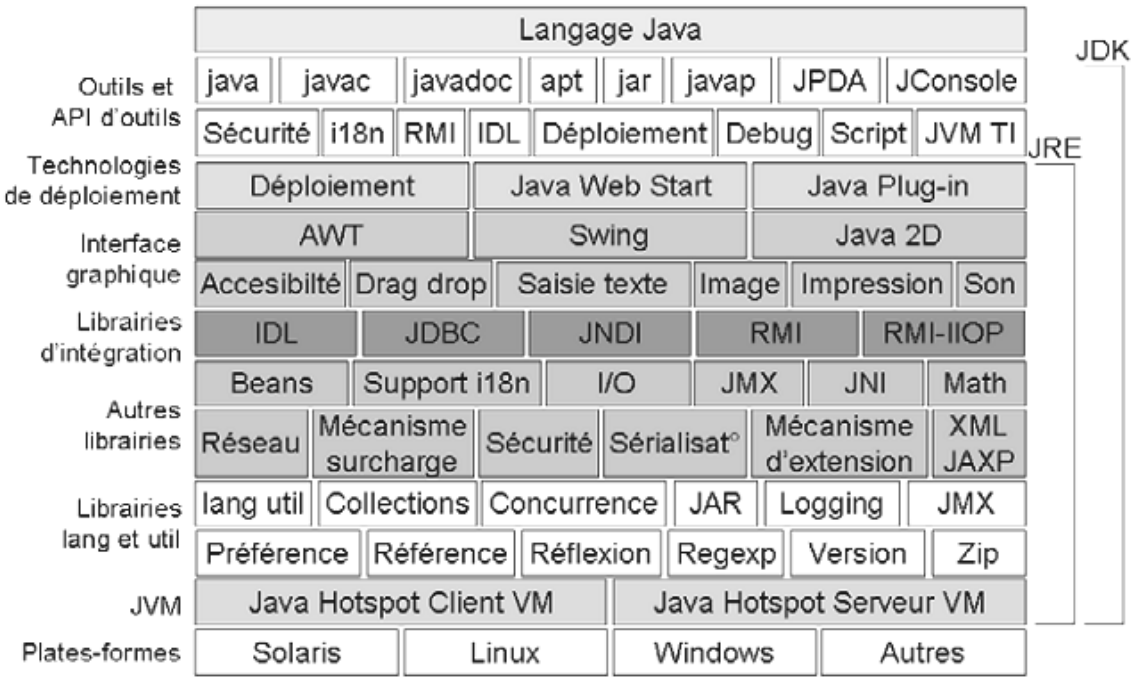

Figure 10 : Les composants du JAVA SE 5.

2.2. JNDI :

Ce composant, appelé communément service de nommage, est un service fondamental dans n'importe quel système informatique. Il permet d'associer des noms à des objets et de retrouver ces objets grâce à leurs noms. Java Naming and Directory Interface (JNDI) fournit les fonctionnalités de nommage et d'annuaire aux applications écrites en Java.

Omniprésent dans la version J2EE 1.4, JNDI se fait plus discret et est intégré directement dans le JSE 5. Il continue à être une pièce maîtresse mais de manière plus transparente.

2.3. JDBC :

JDBC (Java Data Base Connectivity) est une API permettant l'accès à des bases de données relationnelles à partir du langage Java. Elle se charge de trois étapes indispensables à l'accès aux données :

- La création d'une connexion à la base.
- L'envoi d'instructions SQL.
- L'exploitation des résultats provenant de la base.

Pour accéder à la base de données, JDBC s'appuie sur des drivers (pilotes) spécifiques à un fournisseur de SGBDR ou sur des pilotes génériques.

2.4. XML et XSD :

SGML (Standard Generalized Markup Language, ou langage normalisé de balisage généralisé), adopté comme standard en 1986, a été la première tentative pour créer des documents électroniques. L'idée principale était de séparer le contenu d'un document de sa forme. SGML a été une innovation, mais il était si complexe que sa manipulation s'est trouvée restreinte aux spécialistes.

XML (eXtensible Markup Language, ou langage extensible de balisage),issu de SGML, a été mis au point par le World Wide Web Consortium (W3C) en 1996. Contrairement à HTML, qui présente un jeu limité de balises orientées présentation (titre, paragraphe, image, lien hypertexte…), XML est un métalangage, qui va permettre d'inventer à volonté de nouvelles balises pour décrire des données et non leur représentation.

XML permet donc de définir des fichiers dont la structure est personnalisée par la création de balises. De fait, ce langage s'impose comme un standard dans les échanges inter-systèmes d'information. XML devient un format pivot, encore qualifié de format d'échanges. De plus, un certain nombre d'API offre des mécanismes pour créer, extraire et vérifier la validité d'un document XML. Cette validation n'est possible que si l'on connaît la structure du document. Cette structure est définie par un XML Schema Définition (XSD), technologie dérivée d'XML. Un schéma XML (XSD) est lui-même un fichier XML.

2.5. La plate-forme Java EE :

Java EE, ou JEE ou encore Java Enterprise Edition, est un ensemble de spécifications destinées aux applications d'entreprise. JEE peut être vu comme une extension du langage Java afin de faciliter la création d'applications réparties, robustes, performantes et à haute disponibilité. Comme beaucoup, je pense que Java EE est aujourd'hui la meilleure plate-forme de développement pour les entreprises. Elle combine les avantages du langage Java avec l'expérience acquise dans le développement au cours des dix dernières années. Elle bénéficie en outre du dynamisme des communautés Open Source ainsi que du JCP de Sun.

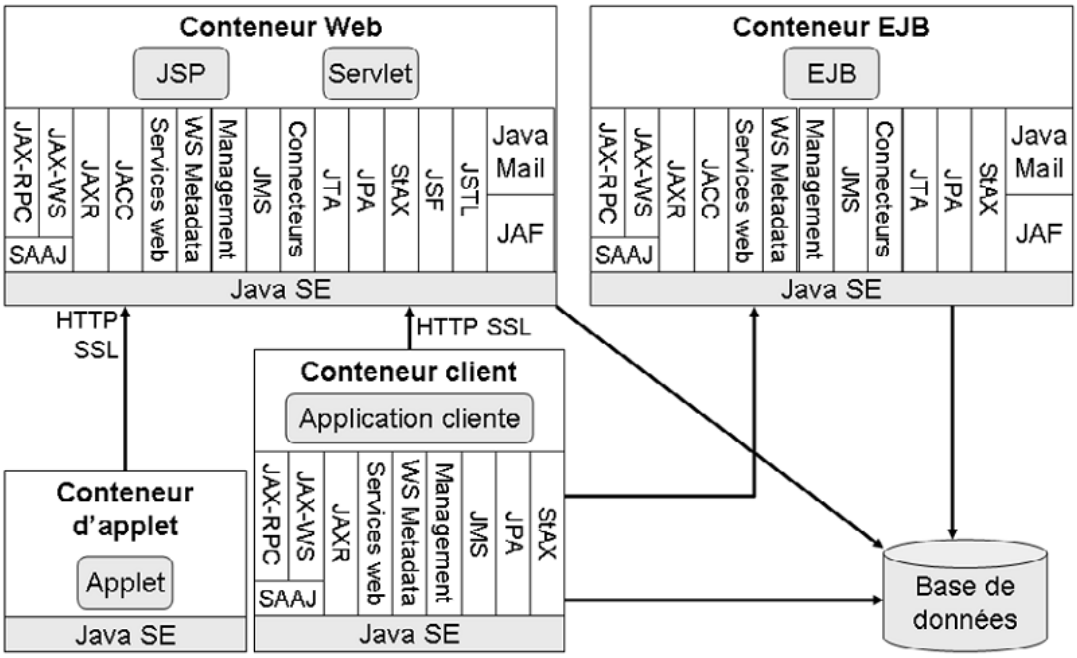

Figure 11 : Architecture JAVA EE .

Bien que prédit à un bel avenir, les promesses de cette plate-forme ne sont pas toujours honorées. Les systèmes délivrés sont souvent trop lents et compliqués, et le temps de développement est, quant à lui, fréquemment disproportionné par rapport à la complexité des demandes utilisateurs.

Heureusement, au deuxième trimestre 2006, JEE 5 est venu simplifier la précédente version (J2EE 1.4). S'appuyant sur la nouvelle mouture du langage Java et s'inspirant de frameworks Open Source, certains composants de la version 5 de JEE ont été totalement réécrits dans le but de simplifier la plate-forme.

La figure architecture java EE5 décrit les différents conteneurs spécifiés dans Java EE 5 ainsi que les spécifications qui peuvent y être employées. Les paragraphes suivants nous donnent un bref descriptif des spécifications utilisées pour le développement de notre application.

2.5. 1. JPA :

Java Persistent API s'appuie sur JDBC pour communiquer avec la base de données. En revanche, grâce à l'abstraction apportée par JPA, nous n'aurons nul besoin d'utiliser directement JDBC dans le code Java.

2.5. 2. EJB :

Les Entreprise Java Beans, ou EJB, sont des composants serveurs qui respectent les spécifications d'un modèle édité par Sun. Ces spécifications définissent une architecture, un environnement d'exécution (un conteneur) et un ensemble d'API. Le respect de ces spécifications permet d'utiliser les EJB indépendamment du conteneur dans lequel ils s'exécutent. Ce dernier fournit un ensemble de fonctionnalités comme la gestion du cycle de vie de l'EJB, la sécurité, l'accès concurrent et les transactions. Le but des EJB est de faciliter la création d'applications distribuées pour les entreprises.

Pièce maîtresse de l'architecture JEE, les EJB 3 apportent des modifications notables dans le mode de développement et intègrent de nombreuses nouveautés, notamment en ce qui concerne la persistance. Le passage des EJB 2.1 en 3.0 apporte une simplification radicale en supprimant les descripteurs de déploiement, les appels JNDI, etc., et laisse place aux annotations. Il existe deux grandes familles d'EJB : entité et session. Les EJB sessions sont différenciés entre EJB sans état, avec état ou s'exécutant en mode asynchrone.

2.5. 3. Servlet et JSP :

Les servlets sont des programmes Java fonctionnant côté serveur au même titre que les CGI et les langages de script tels que ASP ou PHP. Les servlets permettent donc de recevoir des requêtes HTTP, de les traiter et de fournir une réponse dynamique au client. Elles s'exécutent dans un conteneur utilisé pour établir le lien entre la servlet et le serveur web. Les servlets étant des programmes Java, elles peuvent utiliser toutes les API Java afin de communiquer avec des applications externes, se connecter à des bases de données, accéder aux entrées-sorties (fichiers, par exemple)…

Java Server Page, ou JSP, est une technologie basée sur Java qui permet aux développeurs de générer dynamiquement du code HTML, XML ou tout autre type de page web. Une page JSP (repérable par l'extension .jsp) aura un contenu pouvant être différent selon certains paramètres (des informations stockées dans une base de données, les préférences de l'utilisateur...) tandis qu'une page web « statique » (dont l'extension est.htm ou .html) affichera continuellement la même information.

2.5. 4. Langage d'expression

Le langage d'expression, ou Expression langage (EL), permet aux JSP d'accéder aux objets Java, de manipuler des collections ou d'exécuter des actions JSF.

2.5. 5. JSTL :

JSTL est le sigle de JSP Standard Tag Library. C'est un ensemble de balises personnalisées (Custom Tag) développées sous la JSR 052 facilitant la séparation des rôles entre le développeur Java et le concepteur de pages web. L'avantage de ces balises est de déporter le code Java contenu dans la JSP dans des classes dédiées. Ensuite, il suffit de les utiliser dans le code source de la JSP en utilisant des balises particulières, tout comme vous le feriez avec des balises HTML classiques. Les bibliothèques de balises (Tagl ibs) ou balises personnalisés (CustomTag) permettent de définir ses propres balises basées sur XML, de les regrouper dans une bibliothèque et de les réutiliser dans des JSP. C'est une extension de la technologie JSP apparue à partir de la version 1.1 des spécifications.

2.5. 6. JSF :

Entre les servlets et les JSP, il manquait un framework pour aiguiller, de manière simple, un événement utilisateur vers une action serveur. Des outils libres comme Struts sont venus aider le développeur en décarrelant la couche présentation de la couche métier. Mais aucune spécification n'existait jusqu'à l'apparition de JSF. Java Server Faces est venu combler ce vide en facilitant la conception d'interfaces graphiques web, en gérant automatiquement l'état HTTP ainsi que les événements entre client et serveur. JSF établit une norme dont le rôle est de fournir aux développeurs une large palette d'outils leur permettant d'implémenter des applications web en respectant un modèle bien précis.

2.5. 7. Le conteneur de servlet :

Le cycle de vie d'une servlet, donc d'une JSP, est assuré par le moteur de servlet (aussi appelé conteneur de servlet ou conteneur web). Celui-ci est responsable de fournir la requête HTTP à la servlet, de l'exécuter et de renvoyer la réponse. C'est le « moteur » de toute application web simple, c'est-à-dire ne mettant pas en jeu d'EJB.

2.5. 8. JavaMail :

JavaMail est l'API standard de gestion de courriers électroniques de JEE. Elle permet d'envoyer et de recevoir du courrier électronique et de manipuler les messages (en-tête, sujet, corps, pièces jointes...). JavaMail n'est pas un serveur de courrier en tant que tel, mais plutôt un outil pour interagir avec ce type de serveur. Il peut être vu comme un type de clients de messagerie au même titre que Outlook, Lotus, Eudora,… etc. Pour envoyer ou recevoir des messages, JavaMail utilise différents protocoles comme Smtp, Imap ou POP.

2.5. 9. JAXB :

JAXB est l'acronyme de Java Architecture for XML Binding. Cette API permet de générer des classes Java à partir de schémas XML (XSD) et inversement. Autrement dit, il permet de convertir les fichiers XSD en classes Java. Il est ensuite possible de manipuler le document XML au travers de ces classes. Une fois de plus, les annotations de Java 5 sont venues simplifier l'utilisation de l'API JAXB. En annotant un Pojo (Plain Old Java Object), on peut ensuite obtenir ses attributs au format XML.

3. Architecture de l'application :

Notre application va donc utiliser toutes les technologies énoncées ci-dessus. Comme nous l'avons vu au chapitre précédent (conception), l'application peut être accédée par des navigateurs (client léger utilisé par les visiteurs) et par les clients riches déployés sur les postes des employés à un serveur qui va effectuer, Pour ce faire, notre application est découpée en plusieurs couches :

L'architecture en trois couches est le modèle le plus général des architectures multicouches, ces couches sont :

- **Présentation des données :** affichage sur le poste de travail des données du système et interaction avec l'utilisateur.
- **Traitements métier :** ensemble des règles métiers de l'application.
- **Accès aux données :** manipulation et conservation des données.

Ci-après un diagramme de paquetages UML représentant ces trois couches. Chaque sous-système contient les technologies Java EE 5 qui peut être utilisées dans l'application.

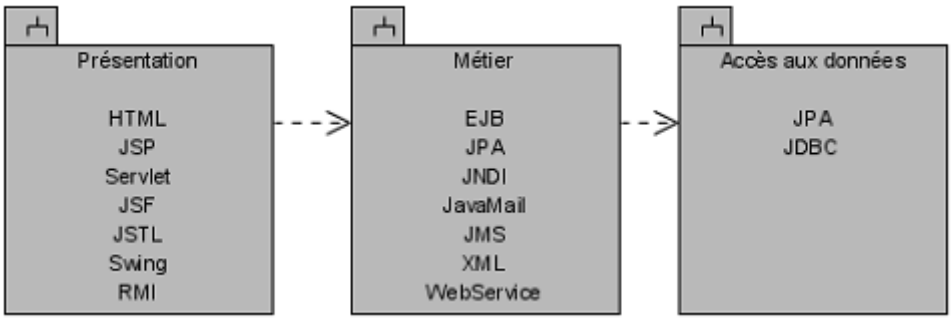

Figure 12 : Architecture en trois couches.

3.1. Architecture applicative :

Le précédent modèle en trois couches peut être affiné pour être plus fidèle à l'architecture finale de l'application. Ci-après un diagramme de paquetage décrivant les couches de l'application :

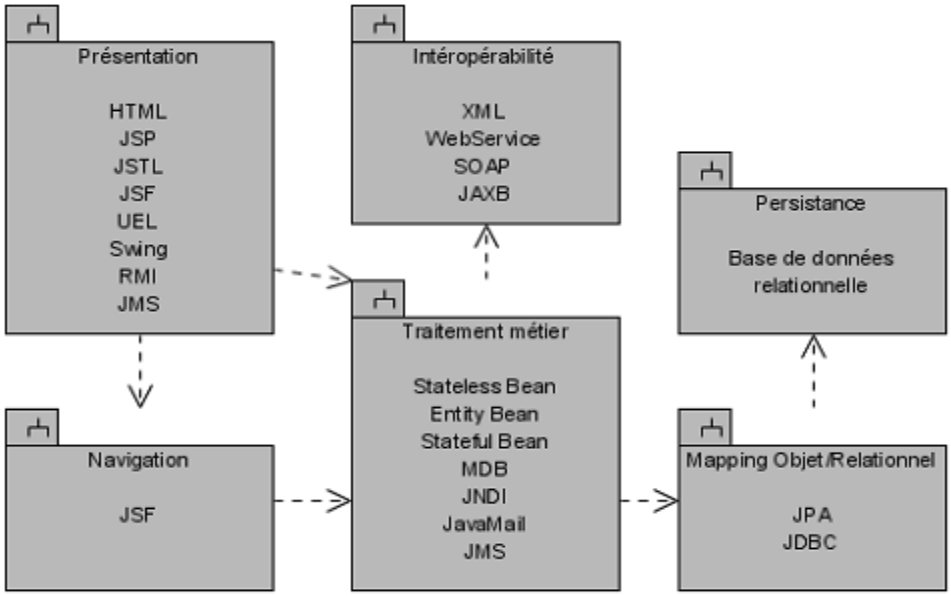

Figure 13 : Les couches de notre application.

3.1.1. Couche de présentation :

La couche de présentation est la partie visible de l'application qui permet à un utilisateur d'interagir avec le système. Elle relaie les requêtes de l'utilisateur à destination de la couche métier, et en retour lui présente les résultats renvoyés par les traitements. On parle alors d'interface homme machine (IHM), aucun traitement n'est implémenté dans cette couche.

L'application possède deux types d'interfaces homme machine : un client léger et un client riche. Le client léger, utilisé par le visiteur, est décrit en langage HTML puis interprété par le navigateur. Dans notre application, nous ne développerons pas directement l'interface en HTML mais plutôt à l'aide de JSP et de tags JSF et JSTL ainsi que du langage d'expression. Les appels à la couche métier sont délégués à la couche navigation. Le client riche, lui, est développé en Swing et utilise le protocole RMI pour communiquer avec la couche métier. Pour afficher les événements asynchrones reçus de l'application, cette couche utilise également JMS.

3.1.2. Couche de navigation :

Cette couche, uniquement utilisée par le client léger, prend en charge la logique de navigation. De ce fait, elle gère l'enchaînement des JSP ainsi que les appels aux traitements métier. Cette couche est mise en œuvre par la technologie JSF.

3.1.3. Couche de traitement métier :

La couche de traitement métier correspond à la partie fonctionnelle ou métier de l'application. Elle implémente la logique et les règles de gestion permettant de répondre aux requêtes de la couche présentation. Pour fournir ces services, elle s'appuie, le cas échéant, sur les données du système, accessibles au travers des services de la couche inférieure, c'est à-dire la couche de données. En retour, elle renvoie à la couche présentation les résultats qu'elle a calculés.

3.1.4. Couche de mapping objet/relationnel :

La couche de mapping objet/relationnel transforme la représentation physique des données en une représentation objet et inversement. Ce mécanisme est assuré par JPA qui utilise le protocole JDBC pour exécuter les appels SQL. Cette couche n'est utilisée que parce que notre base de données est relationnelle. En effet, si la persistance était faite de manière native en objet, ce mapping n'aurait pas lieu d'être.

3.1.5. Couche de persistance :

Cette couche contient les données sauvegardées physiquement sur disque, c'est-à-dire la base de données. En Java, le protocole d'accès aux données est JDBC.

3.1.6. Couche d'interopérabilité :

Pour dialoguer avec ses partenaires DZ-BANK et DZ-TRANS, l'application utilise une couche d'interopérabilité. Basée sur les technologies JAX-WS, elle accède à des services web distants via le protocole HTTP.

4. Outils utilisés pour le développement de l'application :

4.1. JDK :

Indispensable pour le développement et l'exécution de notre application, le Java Development Kit, communément appelé JDK, est le kit de développement proposé gratuitement par Sun. Il comprend plusieurs outils, parmi lesquels :

- **Javac :** le compilateur Java.
- **Java :** un interpréteur d'applications (machine virtuelle).
- **Javadoc :** un générateur de documentation.
- **Jar :** un outil de compression de classes Java.

Le JDK nous permettra de compiler et d'exécuter l'application ainsi que d'autres outils tels que Ant.

4.2. Ant :

Ant est au monde Java ce que « Make » est au monde du langage C : un outil incontournable pour automatiser des traitements répétitifs en mode batch (suppression de fichiers, compilation, compression de fichiers, etc.). Il est simple d'utilisation, bâti sur des technologies ouvertes (Java et XML), extensible et supporté par de nombreux logiciels. Cet outil est aujourd'hui plébiscité par l'ensemble des acteurs majeurs de la communauté Java et communément employé dans la majorité des réalisations d'entreprise.

Ant sera utilisé pour automatiser la compilation, le packaging et le déploiement de l'application. Il nous permettra aussi d'insérer physiquement des données en base et d'administrer le serveur d'applications.

4.3. GlassFish :

GlassFish est un serveur d'applications certifié Java EE 5. Son développement a été initié lorsque « Sun » a ouvert le code de son serveur d'applications pour le licencier en Open Source. Il utilise le moteur de persistance d'Oracle, TopLink Essentials.

GlassFish est constitué :

- ✓ D'un serveur web dédié au service de fichiers, c'est-à-dire à des pages HTML statiques, images, etc...
- ✓ D'un conteneur de servlets, hébergeant des applications composées de servlets et/ou JSP.
- ✓ D'un conteneur d'EJB, pour la gestion des composants stateless, stateful, MDB et entity beans.
- ✓ De l'implémentation de l'API de persistance JPA d'Oracle (TopLink Essentials).

Comme nous le verrons, l'administration du serveur GlassFish se fait soit par interface web, soit par ligne de commande.

4.4. Derby :

Anciennement appelée « Cloudscape », cette base de données développée en Java a été donnée à la fondation Apache par IBM. De petite taille (2 Mo), cette base de données relationnelle est intégrée au serveur GlassFish. Nous utiliserons Derby pour stocker les données de l'application.

4.5. NetBeans :

Nous avons développé notre application avec l'environnement de travail NetBeans. Cette application sera déployée sur le serveur d'application GlassFish .

4.6. Star UML : Nous avons choisi l'outil open source StarUML pour la modélisation.

5. Présentation de l'application :

Après avoir présenté les technologies et outils de développement, nous allons illustrer les fonctionnalités de notre système en effectuant quelques prises d'écran correspond à des cas d'utilisation vue dans le chapitre de conception.

Pour accéder au système, nous avons développés une interface Swing (client riche) pour les employés de la société DZ-INFO (partie administration), et une interface Web (client léger) pour les internautes (visiteurs, clients).

5.1. Maquette pour cas d'utilisation (Gérer le client) :

Les employés de notre société utilisent une application riche pour dialoguer avec le système. Pour la gestion des clients, ils utilisent un écran qui affiche la liste de tous les clients. Ils peuvent ensuite consulter les informations en cliquant sur le bouton « View » ou supprimer le client en cliquant sur « Delete ». Un autre menu permet de manipuler les informations d'un client, c'est-à-dire la création, mise à jour, suppression et recherche à partir de son identifiant.

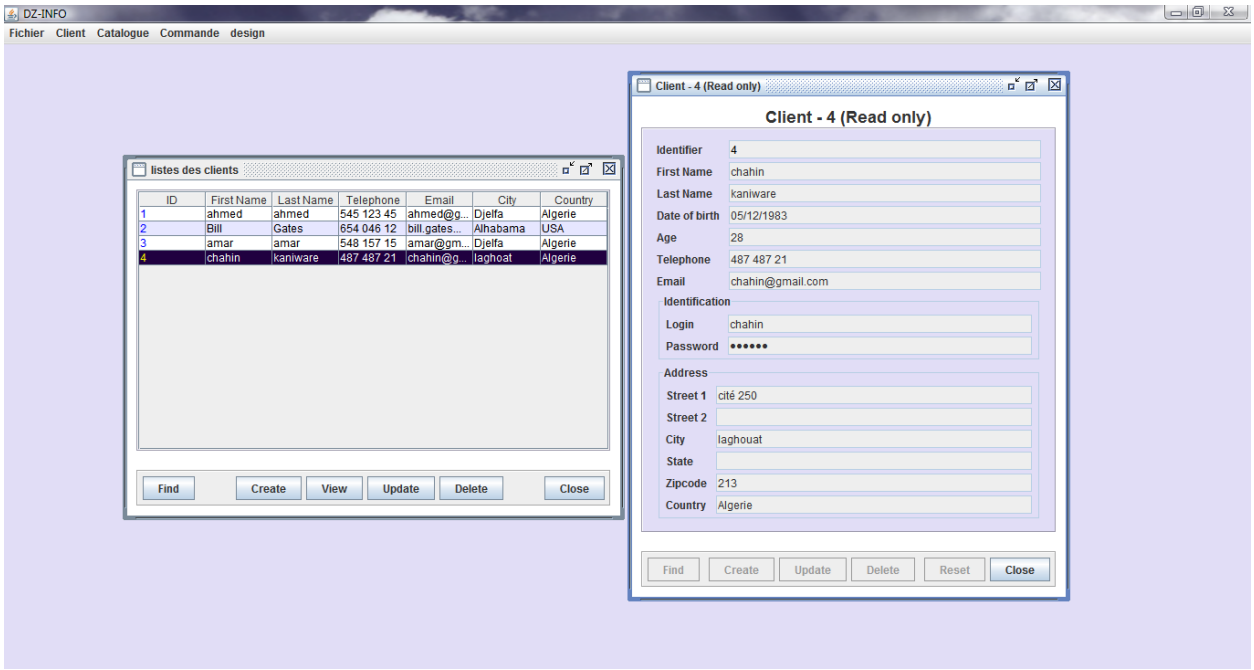

Figure 14 : Application riche de gestion des clients.

5.2. Maquette pour cas d'utilisation (Visualiser et supprimer les commandes) :

L'employé peut visualiser la liste des commandes présentes dans le système. Pour chacune de ces commandes, il peut en consulter les informations et les supprimer.

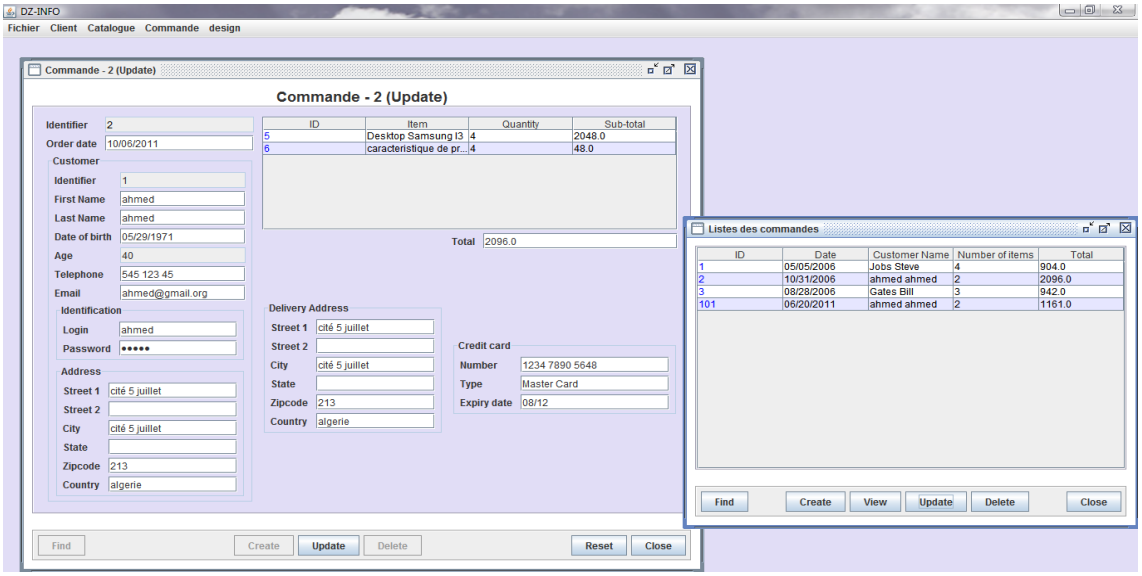

Figure 15: Application client riche de la gestion des bons de commande.

5.3. Maquette pour cas d'utilisation (Gérer le catalogue) :

L'application client riche de l'employé permet de gérer tous les éléments du catalogue, c'est-à-dire les catégories, les produits et les articles. Ci-après, les écrans permettant d'afficher la totalité du catalogue ainsi que de manipuler individuellement chacun des éléments le composant.

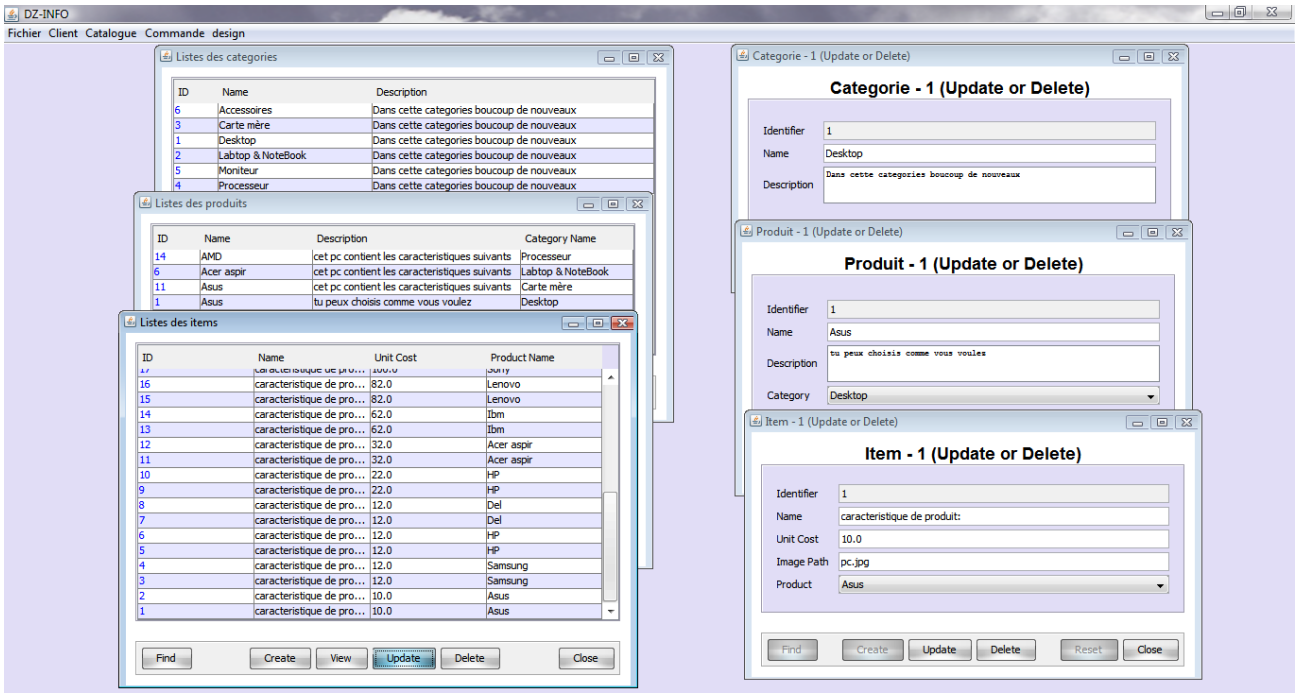

Figure 16 : Application riche de gestion du catalogue.

5. 4. Maquette pour cas d'utilisation (Se créer un compte) :

Pour se créer un compte, le visiteur clique sur le menu « Sign on », puis saisit un login unique suivi de deux fois son mot de passe. Après vérification de la validité des mots de passe et de leur concordance, le système lui demande de compléter ses informations.

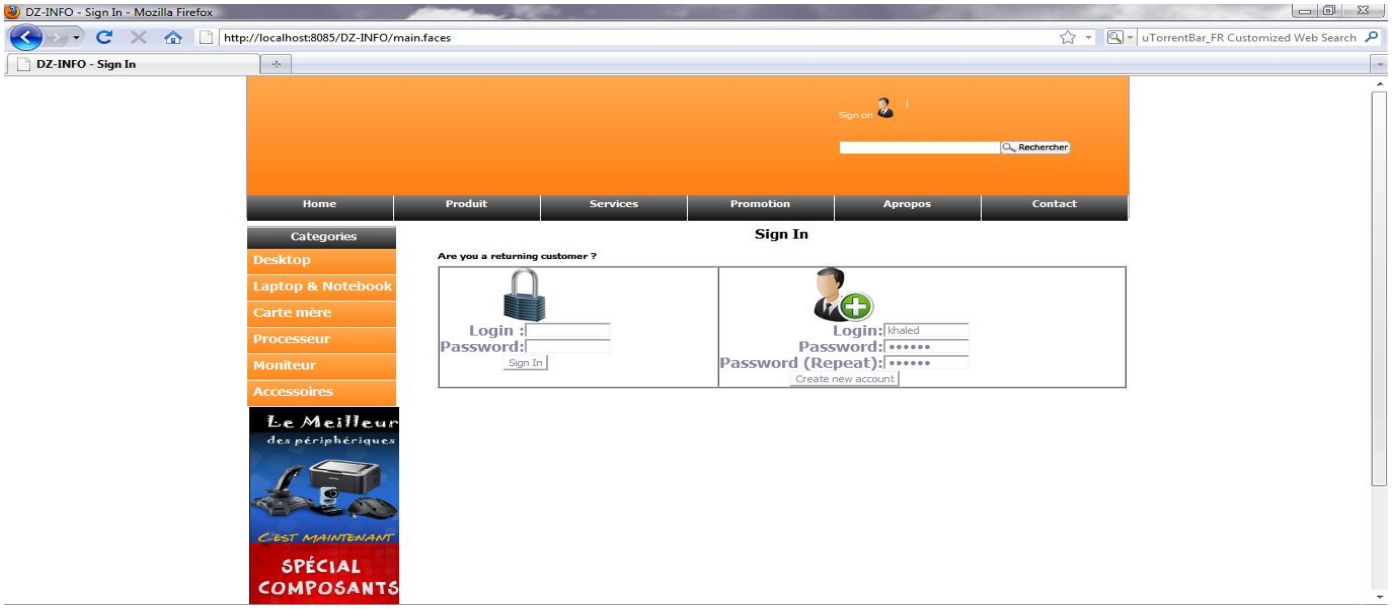

Figure 17 : Le client saisit son login et deux fois son mot de passe.

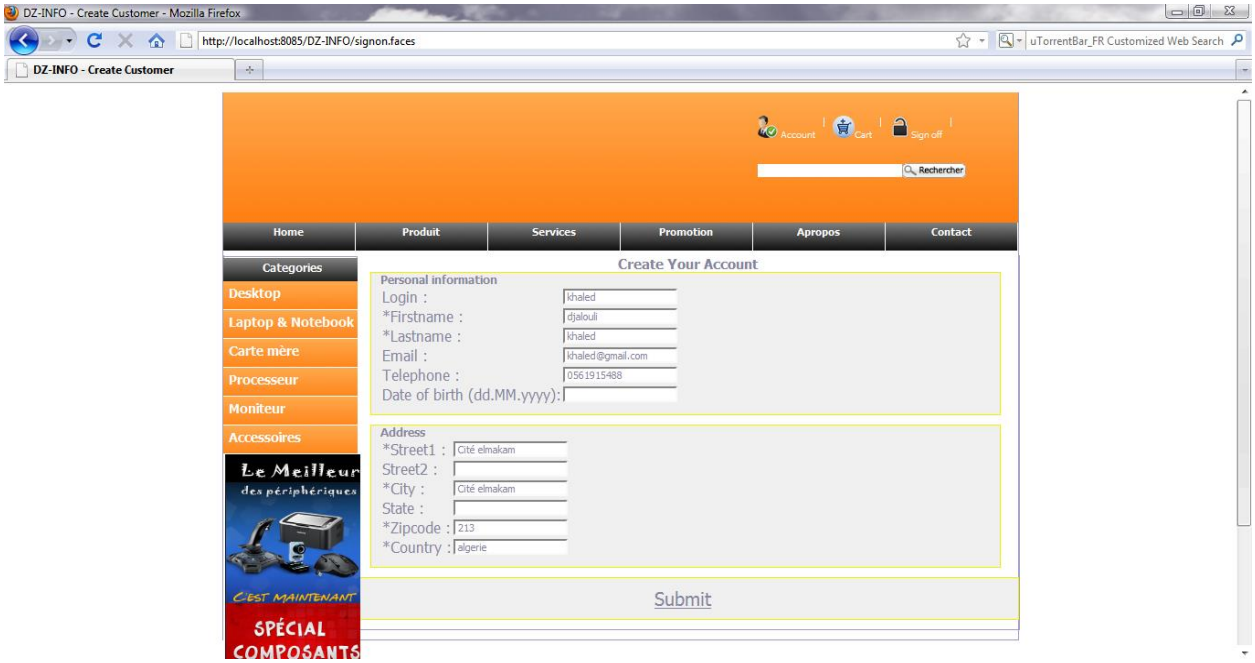

Figure 18 : Saisie des informations du client.

Cette fois, la page d'accueil affiche le nom et prénom du client ainsi que trois liens lui permettant de se déconnecter « Sign Off », de consulter se informations « Account » et de visualiser le contenu de son panier électronique (Caddie) « Cart ».

5.5. Maquette pour cas d'utilisation (Acheter des articles) :

Lorsque le visiteur s'authentifie, le menu « Cart » apparaît en haut de la page. Ce lien permet d'afficher le contenu du panier électronique. Si ce dernier est vide, la page affiche un message avertissant le client.

Figure 19 : Le panier électronique est vide.

Pour remplir le panier, il suffit de se rendre sur la page de description des articles et de cliquer sur le lien « Add to cart ». Cette action ajoute dans le Caddie l'article sélectionné avec une quantité égale à un.

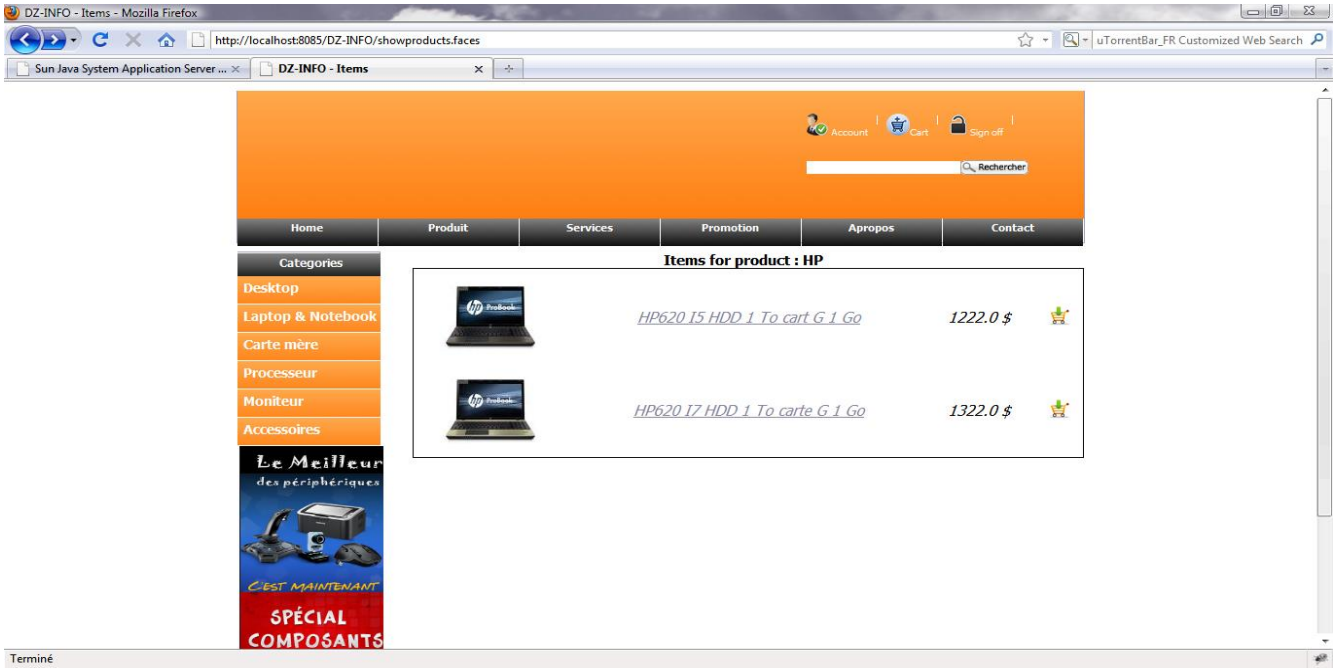

Figure 20 : Le client ajoute des articles en cliquant sur « Add to cart ».

Après avoir effectué différents achats, le client clique sur le lien « Cart » pour consulter le contenu de son panier électronique. Cette page affiche le nom des articles achetés ainsi que leur quantité et leur prix. Le client peut à tout moment modifier la quantité de chaque article en cliquant sur « Update » ou supprimer un article en cliquant sur « Remove ». En bas du tableau s'affiche le montant total du panier électronique.

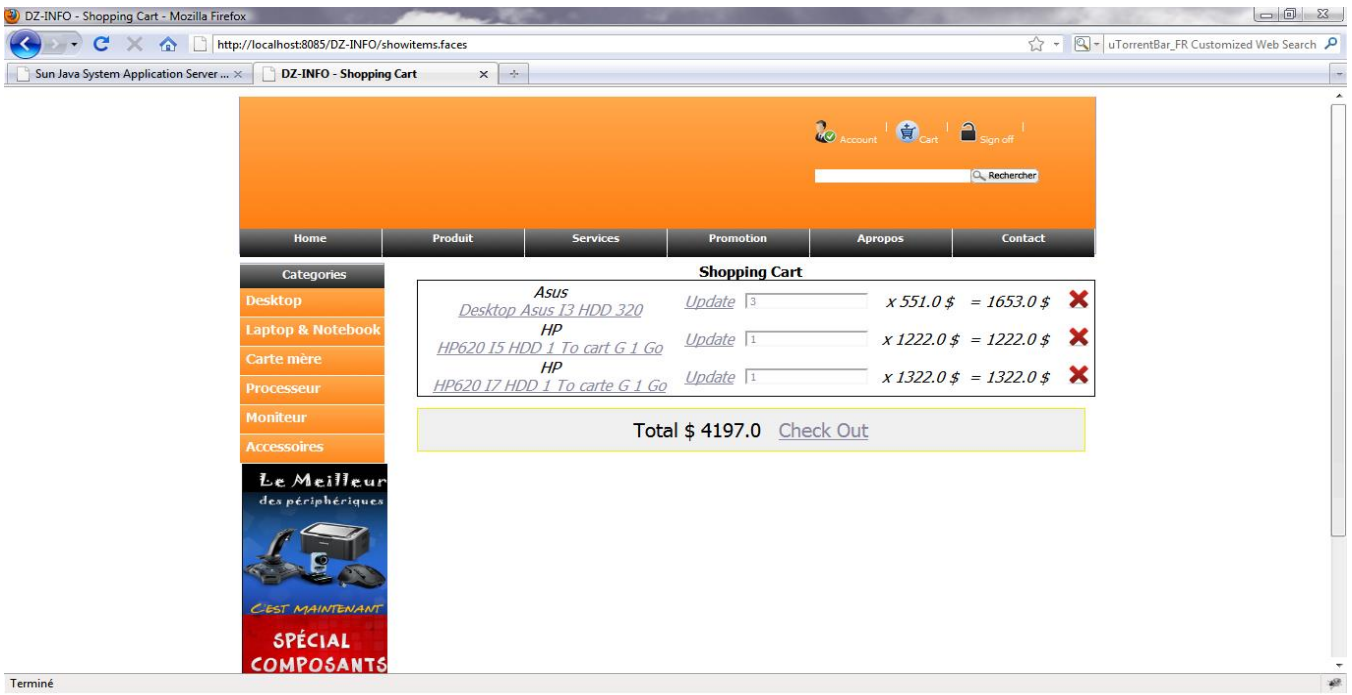

Figure 21 : Contenu du panier électronique.

Une fois les achats terminés, le client clique sur le lien « Check out ». Cette action l'amène sur une page lui demandant de saisir l'adresse de livraison et les coordonnées de sa carte bancaire.

Le client valide la page en cliquant sur « Submit ». Il est alors redirigé vers une page qui lui confirme la création de sa commande et lui en donne le numéro ainsi que son récapitulatif.

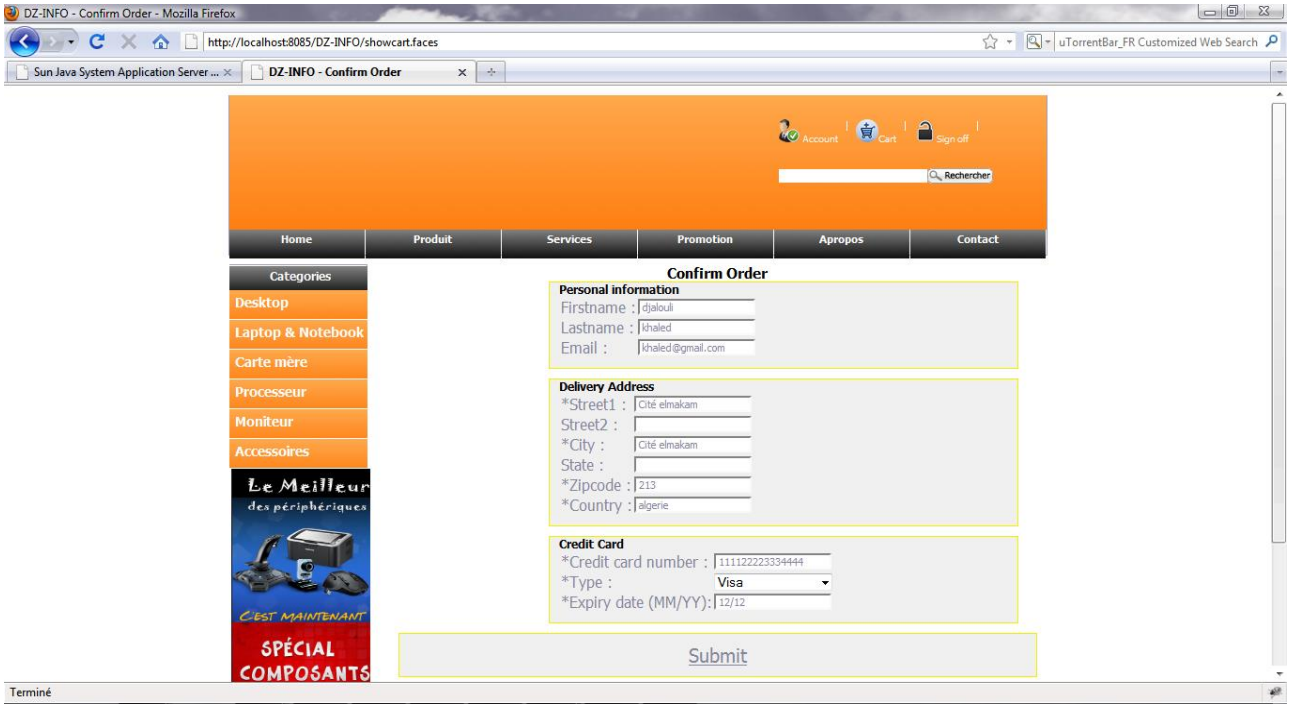

Figure 22 : Saisie de l'adresse de livraison et du mode de paiement

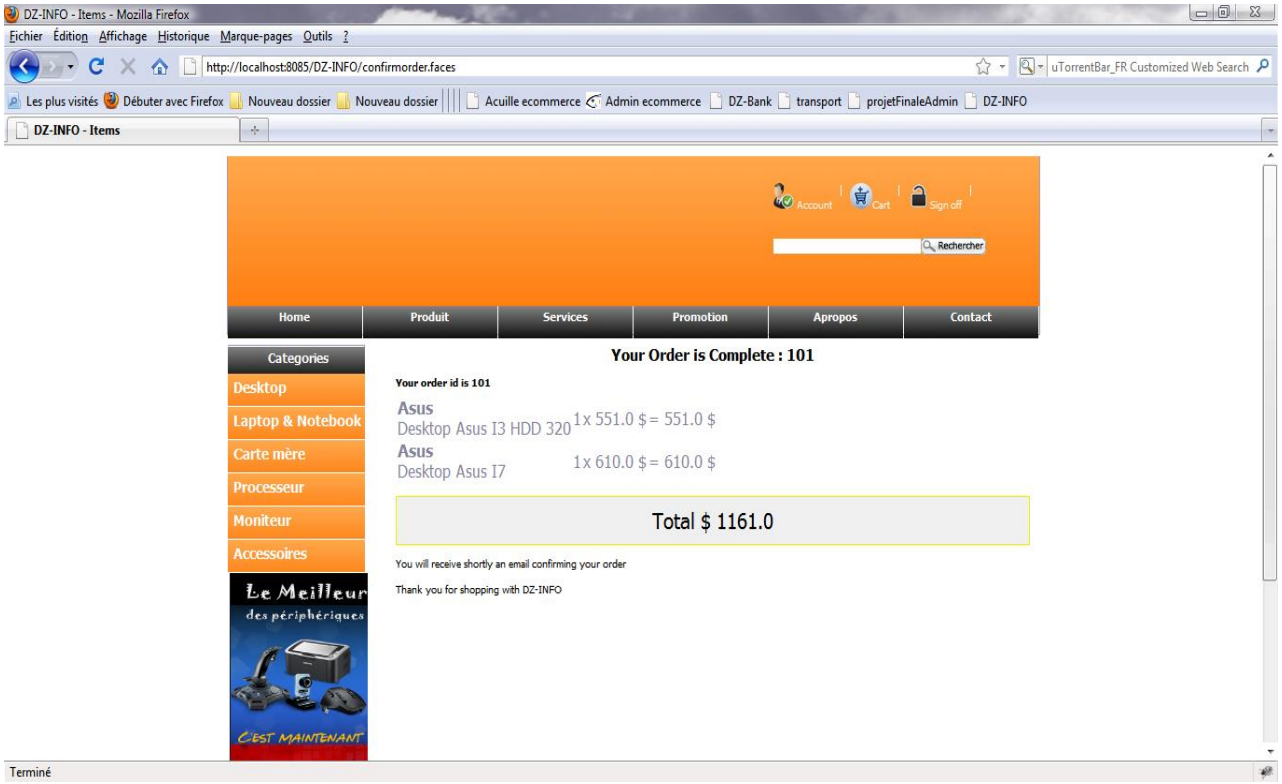

Figure 23 : Confirmation de la création du bon de commande.

6. Conclusion :

Dans ce chapitre, nous avons décrit brièvement le processus de réalisation de notre application en spécifiant la plateforme, l'environnement de développement, l'architecture de l'application, ainsi que les maquettes.

Pour accéder au système, nous avons développés une interface Swing (client riche) pour les employés de la société DZ-INFO, et une interface Web (client léger) pour les internautes (visiteurs, clients).

Conclusion Générale :

Les services Web sont des technologies émergentes, prometteuses pour le développement, le déploiement et l'intégration d'applications Internet.

Un des avantages majeurs des services Web est l'apport de l'interopérabilité sur Internet entre les plateformes, les applications et les langages de programmations, c'est-à-dire que l'architecture des services Web permet aux applications de communiquer facilement via l'Internet quelque soient le système d'exploitation et le langage de programmation.

Pour saisir ces avantages, Notre projet de fin d'études a été abordé dans le but de réaliser une application E-commerce basée sur les services web.

Ce projet de fin d'études nous a été bénéfique car il nous a permis de bien nous familiariser à concevoir et réaliser une application distribué, en utilisant : UML comme outils de modélisation, et la plateforme JEE (JSF, EJB, service Web, JPA, etc.) comme outils de réalisation.

Références

[Kad 03] Hubert Kadima &Valérie Monfort, les Services Web, éditions Dunod ,2003

[Mae 03] Libero Maesano, Christian Bernard & Xavier Le Galles, Services Web avec J2EE et .NET : Conception et implémentations, éditions Eyrolles ,2003

[IWS 03] Gilbert Babin et Michel Leblanc , RAPPORT BOURGOGNE 2003

[AMO 09] Bellani Aicha,et Deriche Imane Mémoire de fin d'etude titre : Construction d'une application e-commerce a base agent mobile INI 2009

[Wiki]. Site Wikipedia www .wikipedia.com

[W3C]. le site de http://www.w3.org

[SWS10] :Berhouche Nabil & Trahi azeddine ,mémoire fin d'étude titre Sécurité des architectures basées sur le web service INI 2010

[SCA10] : Ramdani Nabil & Benseddik Ferhat mémoire de fin d'etude titre Un système pour la composition automatique de sservice web basée sur leur comportement INI 2010

[ISW] : Introduction aux service web cours rédigé par Selimane Hamouddi & Denivalde Lopes

[SDZ] : Site du zéro : www .siteduzero.com

[JEE5 cdp, 05] :Antonio Goncalves , Cahier du programmeur Eyrolle 2005

[Pat04] : Patrick Kellert et Farouk Toumani, Les Web Services sémantiques, http://www.isima.fr/limos/, 2004.