SOMMAIRE

LISTE DES FIGURES

Chapitres I : MIDDLEWARE

Chapitres II : LE LANGAGE XML18

Chapitres III : LES SERVICES WEB

Chapitres IV : LA CONCEPTION ET REALISATION DE L'APPLICATION

LISTE DES TABLEAUX

Chapitres II : LE LANGAGE XML

API	*Application Pogramming Language*
ASP	*Active Server Pages*
CORBA	*Common Object Request Broker Architecture*
DCOM	*Distributed Component Objet Model*
DII	*Dynamic Invocation Interface*
DOM	*Document Objet Model*
DSI	*Dynamic Skeleton Interface*
DTD	*Document Type Definition*
FTP	*File Transfer Protocol*
HTML	*Hyper Text Markup Language*
HTTP	*Hyper Text Transfer Protocol*
IBM	*International Business Machines*
IDE	*Integrated Development Environment*
IFR	*Interface Repository*
IIOP	*Internet Inter-Orb Protocol*
IMAP	*Internet Message Access Protocol*
ImplR	*Implementation Repository*
JSP	*Java Server Pages*
OA	*Object Adapter*
OASIS	*Organization for the Advancement of Structured Iformation Standards*
ODP-RM	*Open Distributed Processing Reference Model*
OMG	*Object Management Group*
OMT	*Object Modeling Technique*
OOSE	*Object Oriented Software Engineering*
ORB	*Object Request Broker*
OSGI	*Open Service Gateway Initiative*
POP	*Post Office Protocol*
REST	*Representational State Transfer*
RMI	*Remote Method Invocation*
RPC	*Remote Procedure Call*
SAX	*Simple API For XML*
SGBD	*Système de Gestion de Base de Données*
SII	*Static Invocation Interface*
SMTP	*Simple Mail Transfer Protocol*
SSI	*Skelton Static Interface*
SSL	*Secure Sockets Layer*
SOAP	*Simple Object Access Protocol*
TLS	*Transport Layer Security*
UDDI	*Universal Description, Discovery and Integration*
UML	*Unified Modeling Language*
URL	*Uniform Resource Locator*
W3C	*World Wide Web Consortium*
WSDL	*Web Services Description Language*
XML	*eXtensible Markup Language*

Il existe aujourd'hui un certain nombre de plates-formes pour créer des applications. Chacune d'entre elles utilise généralement ses propres protocoles, le plus souvent de type binaire, pour l'intégration de machine à machine. En conséquence, les applications fonctionnant sur des plates-formes différentes n'ont qu'une faible capacité de partage de données. La prise de conscience de ces limites a entraîné un gros effort de standardisation des formats de données et d'échange de données. En effet, les regards se tournent de plus en plus vers un nouveau paradigme informatique : une intégration transparente des services Web qui dépasse les barrières logicielles et matérielles traditionnelles.

Au cœur de cette vision se trouve le concept d'interopérabilité, c'est-à-dire la capacité pour des systèmes disparates de communiquer et de partager des données de façon transparente. C'est l'objectif des services Web. Un service Web est une logique d'application programmable accessible à l'aide des protocoles Internet standard, que l'on peut aussi décrire comme l'implémentation de standards Web pour une communication transparente entre les machines et entre les applications.

Un certain nombre de technologies des services Web, telles que le protocole SOAP (*Simple Object Access Protocol*), le langage WSDL (*Web Service Description Language*) et le protocole HTTP (*HyperText Transfer Protocol*), sont actuellement utilisées pour transférer les messages entre les machines. La complexité de ces messages est très variable, pouvant aller de l'appel de méthode à la soumission d'un bon de commande. Une fonction courante - de niveau plus élevé - d'un service Web consiste à implémenter la communication de type RPC (*Remote Procedure Call*, appel de procédure à distance permettant à un programme sur un ordinateur d'exécuter un programme sur un autre ordinateur). Ce mémoire présente une introduction pratique aux questions courantes d'interopérabilité, notamment celles relatives à la communication de type RPC utilisant le protocole SOAP.

Rapport-Gratuit.com

Dans le cadre de ce travail, nous sommes intéressés à l'interopérabilité des services web. On a essayé dans ce mémoire de concevoir et de réaliser une application services web pour la réservation des vols, les services web sont implémentés en langage java et le client qui va consommer ce service est codé en langage C#.

Notre service web permet aux passager de recherche des vols et de réserver un certain nombre de place.

Ce document est organisé comme suit :
Le premier chapitre est consacré aux technologies du middleware.
Le deuxième chapitre est une introduction aux concepts du langage XML.
Le troisième chapitre est une présentation des technologies de services Web.
Le quatrième chapitre concerne la présentation de l'architecture de notre système, la présentation des outils utilisés ainsi qu'une étude de cas détaillé.

Et finalement une conclusion

1 INTRODUCTION

En architecture informatique, un **middleware** (anglicisme) est un logiciel tiers qui crée un réseau d'échange d'informations entre différentes applications informatiques. Le réseau est mis en œuvre par l'utilisation d'une même technique d'échange d'informations dans toutes les applications impliquées à l'aide de composants logiciels. [5]

2 Définition

Un middleware(en français intergiciel) est un logiciel de connexion qui se compose d'un ensemble de service et/ou de milieu De développement d'applications distribués qui permettent à plusieurs entités (processus, Objet,…etc..) résidents sur un ou plusieurs ordinateurs, d'interagir a travers un réseau d'interconnexion en dépit des différences dans les protocoles de communication, architecture des systèmes locaux, systèmes opérationnels etc... . [10]

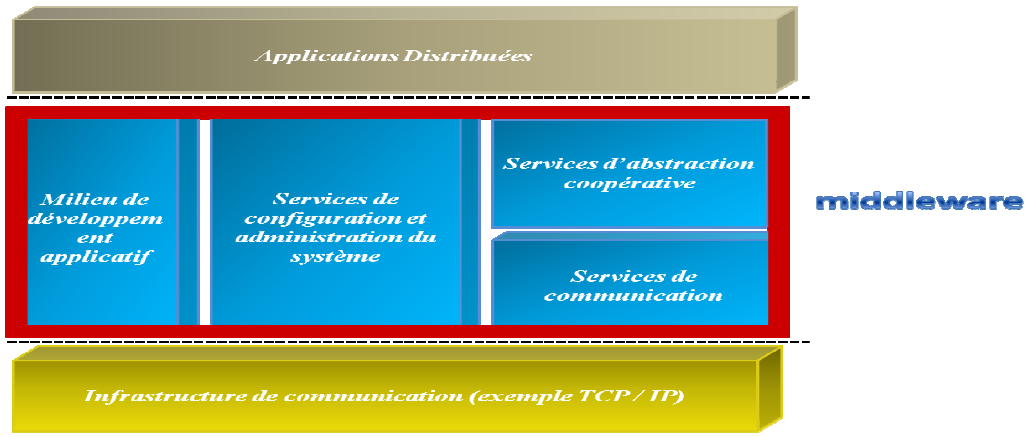


Figure I.1 : L'architecture de middleware

3 Les systèmes distribués

3.1 Définition

Un système distribué est une collection de postes qui sont connectés à l'aide d'un réseau de communication.

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Chaque poste exécute des composantes et utilise un intergiciel qui s'occupe d'activer les composantes et de coordonner leurs activités de telle sorte qu'un utilisateur perçoive le système comme un unique système intégré. [19]

3.2 Le modèle client / serveur

3.2.1 Définition

- un schéma général d'interaction au niveau application
- mis en œuvre sous des formes divers (différents protocoles)

❖ ***RPC, Java RMI, Service Web, etc.***

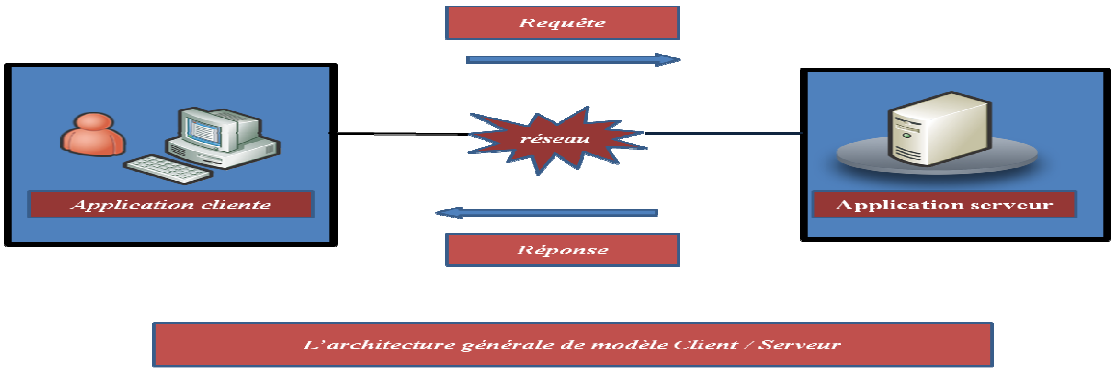


Figure I .2 : architecture générale client/serveur

❖ Le client demande l'exécution d'un service ; le serveur réalise le service

❖ Client et serveur sont(en général, pas nécessairement) localisés sur deux machines distinctes

❖ Indépendance interface-réalisation

❖ **Communication par message (plutôt par partage de données, mémoire ou fichier)**

- ***Requête*** : paramètre d'appel, spécification du service requis
- ***Réponse*** : résultats, indicateur éventuel d'exécution ou d'erreur
- ***Communication synchrone (dans le modèle de base)*** : le client est bloqué en attente de la réponse. [4]

3.3 Les architectures basée sur le modèle client / serveur

3.3.1 L'architecture 1 tiers

Dans ce contexte, les utilisateurs se connectent aux applications exécutées par le serveur central (le mainframe) à l'aide de terminaux passifs se comportant en esclaves. [10]

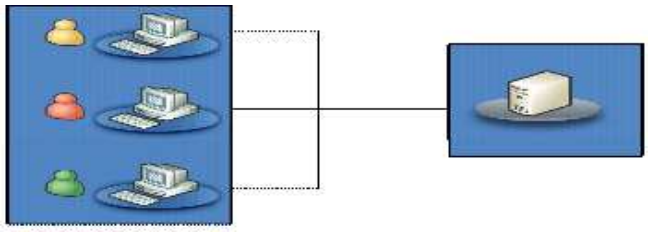


Figure I.3 : architecture 1-tiers

3.3.2 L'architecture 2 tiers

Une architecture 2-tiers est composée de deux éléments, un client et un serveur et où le tiers fait référence non pas à une entité physique mais logique. [10]

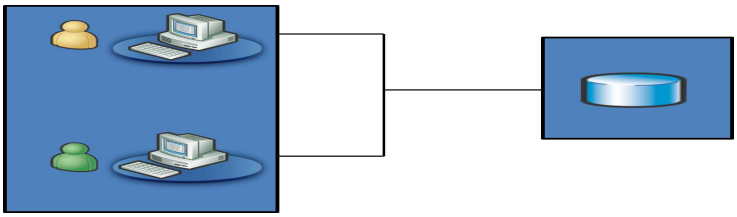


Figure I.4 : architecture 2-tiers

3.3.3 L'architecture 3 tiers

L'architecture 3-tiers est composée de trois éléments (trois couches) qui sont :

i. La couche présentation (ou affichage si l'on souhaite) associée au client qui de fait est dit « léger » dans la mesure où il n'assume aucune fonction de traitement à la différence du modèle 2-tiers.

ii. La couches fonctionnelle liée au serveur, qui dans de nombreux cas est un serveur Web muni d'extensions applicatives.

iii. La couche de donnée liée au serveur de base de données (SGBD). [10]

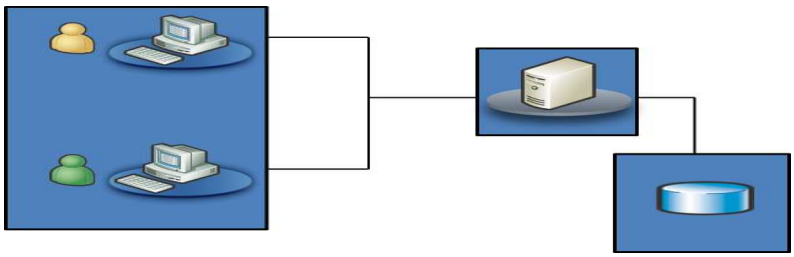


Figure I.5 : architecture 3-tiers

3.3.4 L'architecture n-tiers

L'architecture n-tiers a été pensée pour pallier aux limitations des architectures 3-tiers et concevoir des applications puissantes et simples à maintenir. [10]

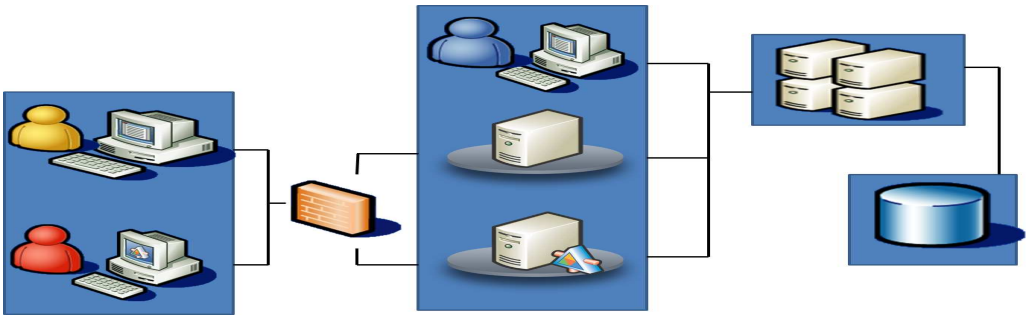


Figure I.6 : architecture n-tiers

4 L'évolution des middlewares

4.1 RPC (Remote Procedure Call)

4.1.1 Définition

Un appel aux procédures distant, RPC transforme l'interaction client / serveur dans un appel à la procédure, semblable à celle locale, en cachant au programmateur la plus grande partie des mécanismes implémentées qui la composent, comme :

- L'échange de message
- La localisation du serveur qui fournit le service
- Les différentes représentations possibles des données des machines impliquées dans l'interaction

Ce déguisement arrive en quatre phases :

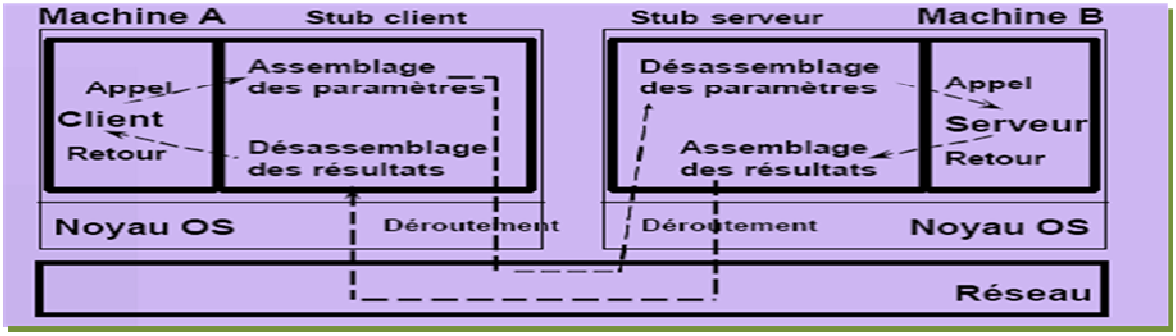


Figure I.7 : Fonctionnement de RPC

❖ A temps d'écriture du code. Les RPC employés/ fournis devront être déclarée par le programmateur explicitement à travers l'import / export des définitions des interfaces

❖ A temps d'exécution. chaque machine sur laquelle il est en exécution un programme client et / ou serveur devra avoir un support à temps d'exécution pour les RPC, RPC run-time support, apte à exécuter quelques opération des RPC comme par exemple la localisation de serveur ou l'enregistrement d'un nouveau service offert par un nouveau serveur

❖ A temps de compilation. Pendant la compilation pour chaque appelle à la procédure distante, des lignes de code sont accrochées au programme originaire (stub). Ces lignes permettent des opérations standards sur les données (empaquetage et codage reconnue universellement) et les appelles au RPC run-time support

❖ A temps de liaison. Pendant la liaison le programme source est mis en communication avec les RPC run-time support pour obtenir le code exécutable. [14]

4.2 CORBA

4.2.1 Définition

CORBA ou Common Object Request Broker Architecture est né dans les années 1990 du besoin de faire communiquer ensemble des applications en environnement hétérogène (plusieurs systèmes et plusieurs langages).

CORBA n'est qu'un sous ensemble du projet ODP-RM ou Open Distributed Processing Reference Model vise à mettre en place une norme d'architectures distribuées ouvertes.

L' OMG (Object Management Group), créé en 1989 et regroupant plus de 750 membres (vendeurs de logiciels, développeurs et utilisateurs), collabore avec l'ODP pour la mise en place d'une norme sur le middleware, et a créé à cet effet la norme CORBA. . [16]

4.2.2 Le bus d'objets répartis (ORB)

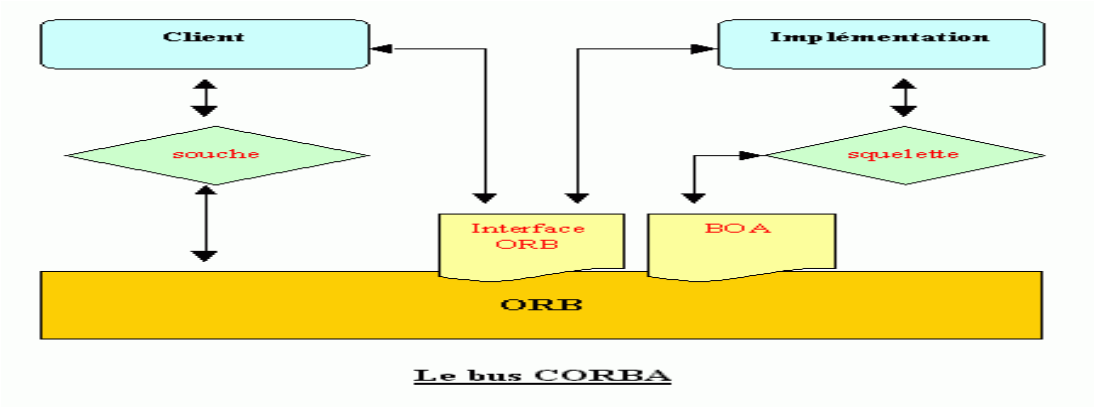


Figure I.8 : le bus CORBA

Le bus d'objets répartis est la clé de voûte de l'architecture globale de l'OMG. Il assure le transport des requêtes entre tous les objets CORBA. Il offre un environnement d'exécution aux objets masquant l'hétérogénéité liée aux langages de programmation, aux systèmes d'exploitation, aux processeurs et aux réseaux.

4.2.3 Les composantes

Le bus CORBA fournit les composantes suivantes :

- **ORB** (Object Request Broker) : est le noyau de transport des requêtes aux objets. Il intègre au minimum les protocoles GIOP et IIOP. L'interface au bus fournit les primitives de base comme l'initialisation de l'ORB.
- **SII** (Static Invocation Interface) : est l'interface d'invocations statiques permettant de soumettre des requêtes contrôlées à la compilation des programmes. Cette interface est générée à partir de définitions OMG-IDL.
- **DII** (Dynamic Invocation Interface) : est l'interface d'invocations dynamiques permettant de construire dynamiquement des requêtes vers n'importe quel objet CORBA sans générer/utiliser une interface SII.

- **IFR** (Interface Repository) : est le référentiel des interfaces contenant une représentation des interfaces OMGIDL accessible par les applications durant l'exécution.
- **SSI** (Skelton Static Interface) : est l'interface de squelettes statiques qui permet à l'implantation des objets de recevoir les requêtes leur étant destinées. Cette interface est générée comme l'interface SII.
- **DSI** (Dynamic Skeleton Interface) : est l'interface de squelettes dynamiques qui permet d'intercepter dynamiquement toute requête sans générer une interface SSI. C'est le pendant de DII pour un serveur.
- **OA** (Object Adapter) : est l'adaptateur d'objets qui s'occupe de créer les objets CORBA, de maintenir les associations entre objets CORBA et implantations et de réaliser l'activation automatique si nécessaire.
- **ImplR** (Implementation Repository) : est le référentiel des implantations qui contient l'information nécessaire à l'activation. Ce référentiel est spécifique à chaque produit CORBA. [17]

4.2.4 Le langage OMG-IDL

Le langage OMG-IDL est la base de la force de CORBA.

Il a été créé pour rendre compatible l'utilisation de plusieurs applications écrites dans différents langages de programmation, ceci s'appelle l'interopérabilité. Il cherche à masquer l'hétérogénéité des applications.

Pour ceci, il a été défini pour décrire uniquement les interfaces indépendamment du langage d'implantation. [18]

4.3 RMI

RMI (*Remote Method Invocation*) est une API java permettant de manipuler des objets distants (un objet instancié sur une autre machine virtuelle (réseau)) de manière transparente pour l'utilisateur c'est-à-dire de la même façon que si l'objet était sur la machine virtuelle de la machine locale.

Ainsi un serveur permet à un client d'invoquer des méthodes à distance sur un objet qu'il instancie.

Deux machines virtuelles sont donc nécessaire (une sur le client et l'autre sur le serveur) et l'ensemble des communications se fait en java. [19]

4.4 DCOM

DCOM (*Distributed Component Object Model*) le système d'objets distribués de Microsoft.

Le modèle DCOM offre également une transparence au niveau du réseau et une automatisation des communications, de sorte que les communications peuvent intervenir entre des objets sans qu'un objet connaisse nécessairement l'emplacement de l'autre objet. Les objets peuvent se situer dans des processus différents sur la même machine ou sur des machines distinctes.

La technologie des objets distribués peut également être utilisée pour faire apparaître des réseaux et des ressources d'information globaux comme locaux, facilitant et accélérant pour les utilisateurs l'accès aux informations stratégiques de l'entreprise.

Par l'intermédiaire du DCOM et de l'Automation à distance, les utilisateurs peuvent situer et exécuter des composants sur des réseaux globaux, sans même être conscients que les informations sont à des milliers de kilomètres. **[20]**

5 Conclusion

Le middleware est au cœur de toute solution de traçabilité. Il joue le rôle d'intermédiaire et d'interprète entre la couche métier et la couche matériel de la solution de traçabilité. Ces deux couches sont en effet gérées de manière bien différente : l'une relève du domaine de l'informatique de gestion, utilisant des données « métier », l'autre, de l'informatique industrielle, traitant des données « terrain » en provenance de différents capteurs.

1 Introduction

XML est l'abréviation d'Extensible Markup Language. Contrairement aux « on dit », XML n'est pas un langage de programmation. On ne peut pas faire de tests, ni inclure un fichier dans un autre. En très gros, XML sert uniquement à stocker des données.

XML est simplement une méthode pour représenter les données. Celles-ci sont écrites entre des balises ou sous forme d'attributs, et l'ensemble est écrit sous forme d'un arbre.

2 Qu'est ce que XML ?

XML (Extensible Markup Language, « langage de balisage extensible ») est un langage informatique de balisage générique. C'est un ensemble de conventions pour la conception de formats texte permettant de structurer des données. On entend par "données structurées" des éléments tels que des feuilles de calcul, des carnets d'adresses, des paramètres de configuration, des transactions financières, etc.

Le World Wide Web Consortium (W3C), promoteur de standards favorisant l'échange d'informations sur Internet, recommande la syntaxe XML pour exprimer des langages de balisages spécifiques. [19]

3 La syntaxe XML

Les documents XML ont des structures syntaxiques et sémantiques. La syntaxe (pensez à orthographe et à ponctuation) est constituée de règles minimales dont (la liste n'est pas exhaustive):

- **Les documents XML ont toujours un et un seul élément racine**
- **Les noms des éléments sont sensibles à la casse**
- **Les éléments doivent toujours être fermés**
- **Les éléments doivent toujours être correctement emboîtés**
- **Les attributs d'éléments doivent toujours être entre guillemets**
- **Il y a seulement cinq entités définies par défaut (<, >, &, ", et '). [5]**

4 Structure d'un document XML

En réalité un document XML est structuré en 3 parties :

Rapport-Gratuit.com

La première partie, appelée prologue permet d'indiquer la version de la norme XML utilisée pour créer le document (cette indication est obligatoire) ainsi que le jeu de caractères (en anglais encoding) utilisé dans le document (attribut facultatif pour permettre de prendre en compte les accents français).

Le second élément est une déclaration de type de document (à l'aide d'un fichier annexe appelé DTD - Document Type Definition)

Et enfin la dernière composante d'un fichier XML est l'arbre des éléments. [23]

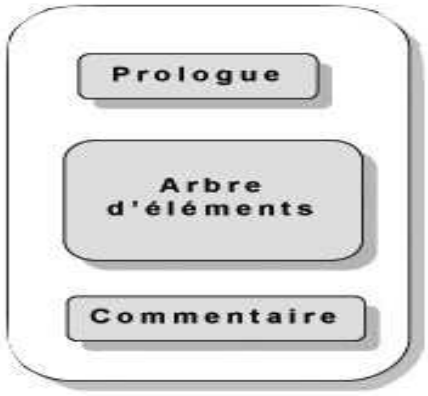


Figure II.1 : Structure d'un document XML

5 Espace de nommage (spacename)

XML définit un système permettant de créer des balises modulaires, c'est-à-dire pouvoir donner la possibilité d'utiliser des balises provenant de différents langages à balise au sein d'un même document grâce à la notion d'espace de noms.

La définition d'un espace de nom permet d'associer toutes les balises d'un langage à un groupe afin d'être capable de mêler différents langages à balise dans un même document XML (être capable de dissocier les éléments de HTML contenus dans le document des balises XML, ou mieux : pouvoir mettre du HTML, MathML, et CML dans un même document).

Fondamentalement il n'y a pas de risques que des balises XML interfèrent avec des balises HTML car ces deux langages ont été mis au point par le même organisme (W3C). Par contre, étant donné que XML est un méta-langage, il permet par définition de définir de nouvelles balises. Ainsi, il se peut que deux organismes mettent au point des langages dont certaines balises portent le même nom, ce qui pose un problème si on désire utiliser des éléments des deux langages au sein d'un même document. [2]

6 GRAMMAIRE XML

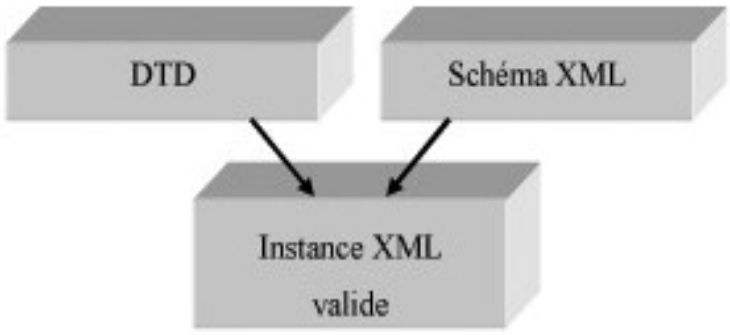


Figure II.2: Une instance XML valide par rapport à une DTD ou à un schéma XML

6.1 DTD (Document Type Definition)

6.1.1 Definition

Le DTD ou Document Type Declaration ou encore Document Type Definition est l'ensemble des règles et des propriétés que doit suivre le document XML. Ces règles définissent généralement le nom et le contenu de chaque balise et le contexte dans lequel elles doivent exister. Cette formalisation des éléments est particulièrement utile lorsqu'on utilise de façon récurrente des balises dans un document XML.

Une DTD peut être définie de 2 façons :

Le DTD interne: On peut inclure son propre DTD au code source du fichier XML. On parlera alors d'un DTD interne.

Le DTD externe : soit en appelant un fichier contenant la grammaire à partir d'un fichier local ou bien en y accédant par son URL (adresse web).

6.1.2 Déclarer un élément

Pour pouvoir créer un document XML il est utile dans un premier temps de définir les éléments pouvant être utilisés, ou plus exactement les informations que l'on désire utiliser.

Ainsi pour définir un élément on utilisera la syntaxe suivante :

<! ELEMENT Nom Modèle >

Le paramètre modèle représente soit un type de donnée prédéfini, soit une règle d'utilisation de l'élément.

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Les types prédéfinis utilisables sont les suivants :

Type prédéfini	Description
ANY	L'élément peut contenir tout type de données
EMPTY	L'élément ne contient pas de données spécifiques
#PCDATA	L'élément doit contenir une chaîne de caractères

Table II.1 : Type prédéfini

Ainsi un element nommé Nom contenant un type #PCDATA sera déclaré de la façon suivante dans la DTD :

<! ELEMENT Nom (#PCDATA) >

Le mot clé *#PCDATA* doit nécessairement être écrit entre parenthèses, au risque sinon d'obtenir une erreur du parseur !

Cet élément pourra être écrit de la façon suivante dans le document XML :

<Nom>Pillou</Nom>

D'autre part il est possible de définir des règles d'utilisation, c'est-à-dire les éléments XML qu'un élément peut ou doit contenir. Cette syntaxe se fait à l'aide de notations spécifiques dont voici un récapitulatif :

Opérateur	**Signification**	**Exemple**
+	L'élément doit être présent au minimum une fois	A+
*	L'élément peut être présent plusieurs fois (ou aucune)	A*
?	L'élément peut être optionnellement présent	A?
|	L'élément A ou l'élément B peuvent être présents	A|B
,	L'élément A doit être présent et suivi de l'élément B	A,B
( )	Les parenthèses permettent de regrouper des éléments Afin de leur appliquer les autres opérateurs	(A, B)+

Table II.2 : Les opérateurs et leurs significations

6.1.3 Déclarer des attributs

Il est possible d'ajouter des propriétés à un élément particulier en lui affectant un attribut, c'est-à-dire une paire clé/valeur. Ainsi avec XML la syntaxe pour définir un attribut est la suivante :

<! ATTLIST Elément Attribut Type >

Type représente le type de donnée de l'attribut, il en existe trois :

- **littéral**: il permet d'affecter une chaîne de caractères à un attribut. Pour déclarer un tel type il faut utiliser le mot clé *CDATA*
- **l'énumération**: cela permet de définir une liste de valeurs possibles pour un attribut donné, afin de limiter le choix de l'utilisateur. La syntaxe de ce type d'attribut est :

  <! ATTLIST Elément Attribut (Valeur1 | Valeur2 | ...) >

Pour définir une valeur par défaut il suffit de faire suivre l'énumération par la valeur désirée entre guillemets :

<! ATTLIST Elément Attribut (Valeur1 | Valeur2) "valeur par défaut" >

- **atomique**: il permet de définir un identifiant unique pour chaque élément grâce au mot clé *ID*.

Enfin chacun de ces types d'attributs peut être suivi d'un mot clé particulier permettant de spécifier le niveau de nécessité de l'attribut :

- **#IMPLIED** signifie que l'attribut est optionnel, c'est-à-dire non obligatoire
- **#REQUIRED** signifie que l'attribut est obligatoire
- **#FIXED** signifie que l'attribut sera affecté d'une valeur par défaut s'il n'est pas défini. Il doit être immédiatement suivi de la valeur entre guillemets

Ainsi on pourra avoir une déclaration d'attribut du type :

<! ATTLIST disque IDdisk ID #REQUIRED type(K7|MiniDisc|Vinyl|CD)"CD" >

Ce qui signifie que l'on affecte à l'élément disque deux attributs IDdisk et type. Le premier attribut est de type atomique, il s'agit d'un identifiant unique obligatoire. L'élément type peut être soit *K7*, *MiniDisc*, *Vinyl* ou *CD*, sachant que ce dernier sera affecté par défaut...

6.1.4 Les limites des DTD

Les DTD souffrent de nombreuses limites :

- Les DTD ne sont pas au format XML. Cela signifie qu'il est nécessaire d'utiliser un outil spécial pour analyser un tel fichier, différent de celui utilisé pour les fichiers XML.
- Les DTD ne supportent pas les "espaces de nom". En pratique, cela implique qu'il n'est pas possible, dans un fichier XML défini par une DTD, d'importer des définitions de balises définies ailleurs.

- Le "typage" des données est extrêmement limité. [19]

6.2 Les schémas XML

Pour pallier aux manques précités des DTD, les schémas XML ont été conçus. Ces derniers proposent, en plus des fonctionnalités fournies par les DTD, des nouveautés:

- Le typage des données est introduit, ce qui permet la gestion de booléens, d'entiers, d'intervalles de temps, etc. Il est même possible de créer de nouveaux types à partir de types existants.
- Le support des espaces de nom.
- Les indicateurs d'occurrences des éléments peuvent être tout nombre non négatif (rappel : dans une DTD, on était limité à 0, 1 ou un nombre infini d'occurrences pour un élément).
- Les schémas sont très facilement concevables par modules.
- La séparation entre le modèle de validation et l'instance XML. [26]

7 Manipulation des documents XML

7.1 L'analyseur XML

L'analyseur syntaxique (généralement francisé en parseur) est un outil logiciel permettant de parcourir un document et d'en extraire des informations. Dans le cas de XML (on parle alors de parseur XML), l'analyseur permet de créer une structure hiérarchique contenant les données contenues dans le document XML.

On distingue deux types de parseurs XML :

- les parseurs validants (validating) permettant de vérifier qu'un document XML est conforme à sa DTD
- les parseurs non validants (non-validating) se contentant de vérifier que le document XML est bien formé (c'est-à-dire respectant la syntaxe XML de base)

Les analyseurs XML sont également divisés selon l'approche qu'ils utilisent pour traiter le document. On distingue actuellement deux types d'approches :

- Les API utilisant une approche hiérarchique : les analyseurs utilisant cette technique construisent une structure hiérarchique contenant des objets représentant les éléments du document, et dont les méthodes permettent d'accèder aux propriétés. La principale API utilisant cette approche est DOM (Document Object Model)

- Les API basés sur un mode événementiel permettent de réagir à des événements (comme le début d'un élément, la fin d'un élément) et de renvoyer le résultat à l'application utilisant cette API. SAX (Simple API for XML est la principale interface utilisant l'aspect événementiel

Ainsi, on tend à associer l'approche hiérarchique avec DOM et l'approche événementielle avec SAX. [19]

7.2 DOM (Document Object Model)

DOM (Document Object Model, traduisez modèle objet de document) est une spécification du W3C (World Wide Web Consortium) définissant la structure d'un document sous forme d'une hiérarchie d'objets, afin de simplifier l'accès aux éléments constitutifs du document.

Plus exactement DOM est un langage normalisé d'interface (API, Application Programming Interface), indépendant de toute plateforme et de tout langage, permettant à une application de parcourir la structure du document et d'agir dynamiquement sur celui-ci. Ainsi Javascript et ECMAScript utilisent DOM pour naviguer au sein du document HTML, ce qui leur permet par exemple de pouvoir récupérer le contenu d'un formulaire, le modifier, ...

DOM se divise en deux spécifications :

- La spécification **DOM level 1** (DOM niveau 1) se séparant en deux catégories
  - ✓ Core DOM level 1: La spécification pour les documents en général (dont XML)
  - ✓ HTML DOM level 1: La spécification retenant uniquement les méthodes applicables à HTML
- La spécificaion **DOM level 2** ajoutant de nouvelles fonctionnalités comme la prise en compte des feuilles de style CSS dans la hiérarchie d'objets. [19]

7.3 SAX(Simple API for XML) :

SAX est une API basée sur un modèle événementiel, cela signifie que SAX permet de déclencher des événements au cours de l'analyse du document XML.
Une application utilisant SAX implémente généralement des gestionnaires d'événements, lui permettant d'effectuer des opérations selon le type d'élément rencontré. [19]

Une application basée sur SAX peut gérer uniquement les éléments dont elle a besoin sans avoir à construire en mémoire une structure contenant l'intégralité du document.

7.4 Comparaison entre DOM et SAX

DOM et SAX sont deux moyens de parser (= analyser la syntaxe) un document XML et en utiliser le contenu.

DOM est le plus simple, le plus intuitif. Il charge le document en mémoire sous forme d'arborescence et permet au programmeur d'appliquer des fonctions sur les éléments de l'arbre.

Sax est plus rapide et consomme moins de mémoire. Il est orienté évènements. Il associe des méthodes aux balises, elles sont activées quand les balises sont atteintes lors de la lecture. Les éléments sont lus en séquence, une seule fois. Il faut fournir son propre modèle de document, alors qu'il en est fourni un avec DOM.

Dans les cas ne nécessitant pas de manipuler les documents XML, mais juste de les lire, la méthode SAX peut également être choisie car elle traite les éléments de façon successive sans charger le document en mémoire. Elle s'impose donc quand la taille du document excède la capacité de la mémoire. **[6]**

8 Conclusion

XML fournit un moyen de vérifier la syntaxe d'un document grâce aux DTD (*Document Type Definition*). Il s'agit d'un fichier décrivant la structure des documents y faisant référence grâce à un langage adapté. Ainsi un document XML doit suivre scrupuleusement les conventions de notation XML et peut éventuellement faire référence à une DTD décrivant l'imbrication des éléments possibles. Un document suivant les règles de XML est appelé *document bien formé*. Un document XML possédant une DTD et étant conforme à celle-ci est appelé *document valide*.

1 INTRODUCTION

Les services Web proposent un mécanisme de communication standard pour faire dialoguer deux applications basées sur des technologies hétérogènes. La communication repose, le plus souvent, sur l'échange de messages XML.

2 DEFINITION D'UN SERVICE WEB

Un service Web(en anglais *web services*) est un système logiciel conçu pour permettre l'interopérabilité de machine à machine interaction sur un réseau. Il a une interface décrite dans un format traitable par machine (en particulier WSDL). D'autres systèmes d'interagir avec le service Web de la manière prescrite par sa description à l'aide messages SOAP, typiquement transmis via le protocole HTTP avec une sérialisation XML. [5]

3 CARACTERISTIQUE D'UN SERVICE WEB

3.1 Web based

Les Web services sont basés sur les protocoles et les langages du Web, en particulier HTTP et XML.

3.2 Self-described, self-contained

Le cadre des Web services contient en lui-même toutes les informations nécessaires à l'utilisation des applications, sous la forme de trois fonctions : trouver, décrire et exécuter.

3.3 Modular

Les Web services fonctionnent de manière modulaire et non pas intégrée. Cela signifie qu'au lieu d'intégrer dans une seule application globale toutes les fonctionnalités, on crée (ou on récupère) plusieurs applications spécifiques qu'on fait intéropérer entre elles, et qui remplissent chacune une de ces fonctionnalités. [34]

Service web = HTTP + SOAP + WSDL + Composantes logiciels.

4 Architecture des Web Services

L'interopérabilité qu'implique les services web doit être soutenue par une architecture stable et fiable. On quitte par ailleurs une architecture client/serveur (qui requière de hauts niveaux d'intégration de ses différentes composantes propriétaires et suppose un branchement qui tient compte des fonctionnalités spécifiques de chacune des applications) pour s'orienter sur une architecture distribuée ne nécessitant plus de connaître le langage, la machine, le système d'exploitation et tous les autres détails habituellement indispensables pour permettre une communication aux deux extrémités du continuum de communication. [31]

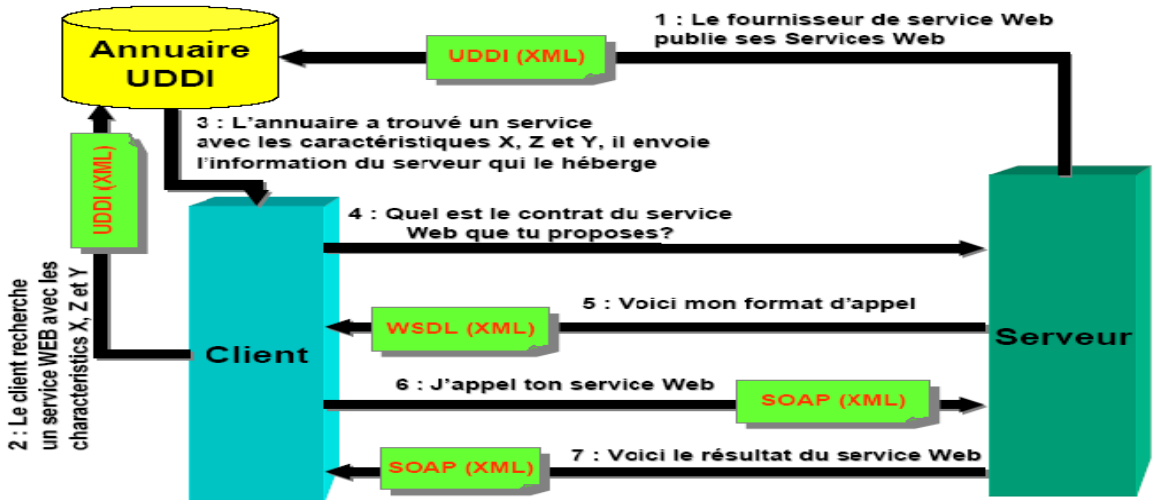


Figure III.1 : architecture des services web

5 STANDARDS ET PROTOCOLES DES SERVICES WEB

- Un nouveau Protocole : SOAP = HTTP + XML

  Requête/réponse = message XML

- WSDL - Description de service web

  Description des interfaces des services

- UDDI - Découverte automatique des services (dynamicité)

  Annuaire contenant les interfaces (Pages Jaunes, Vertes, Blanches)

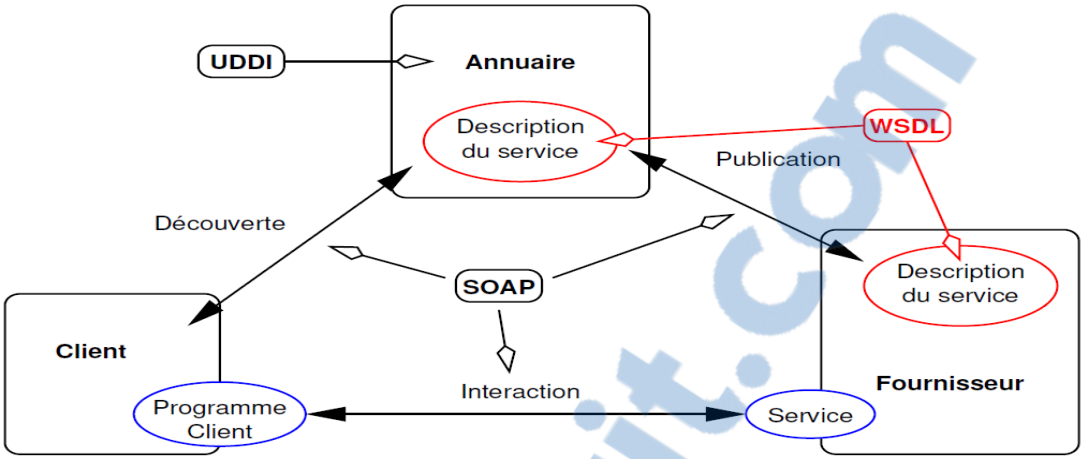


Figure III.2: les standards des services web

6 LA PILE DE PROTOCOLE DES SERVICES WEB

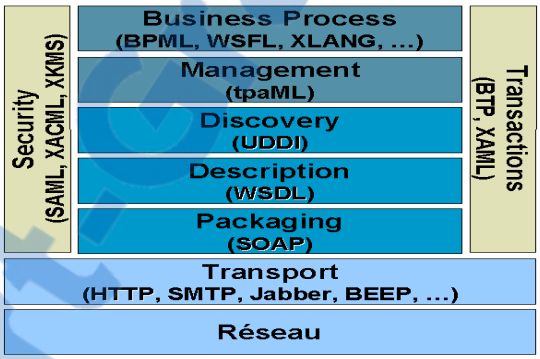


Figure III.3 : la pile de protocole des services web

6.1 Le protocole HTTP

Le Hyper Text Transfer Protocol, plus connu sous l'abréviation HTTP, littéralement le « protocole de transfert hypertexte », est un protocole de communication client-serveur développé pour le World Wide Web. HTTPS (avec S pour *Secured*, soit « sécurisé ») est la variante du HTTP *sécurisée* par l'usage des protocoles SSL ou TLS. [32]

6.2 LE PROTOCOLE SOAP (Simple Object Access Protocol)

SOAP (pour *Simple Object Access Protocol*) est un protocole souple mais complexe, servant à transporter les informations échangées entre deux applications (généralement des services Web). Le principal avantage de SOAP face à l'utilisation d'un document purement XML est la possibilité offerte aux développeurs d'adjoindre leurs propres détails aux messages. [33]

Rapport Gratuit.com

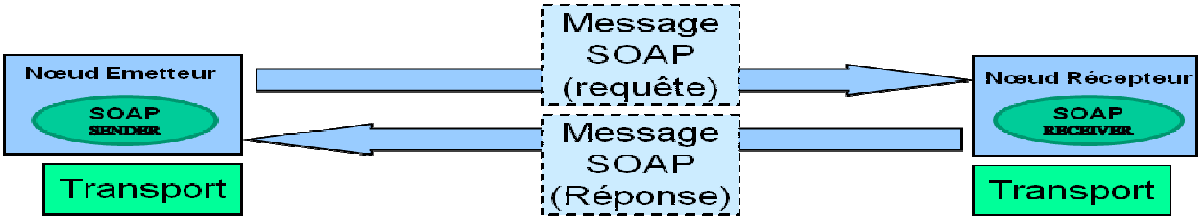


Figure III.4 : Modèle d'échange un message SOAP

6.2.1 Structure d'un message SOAP

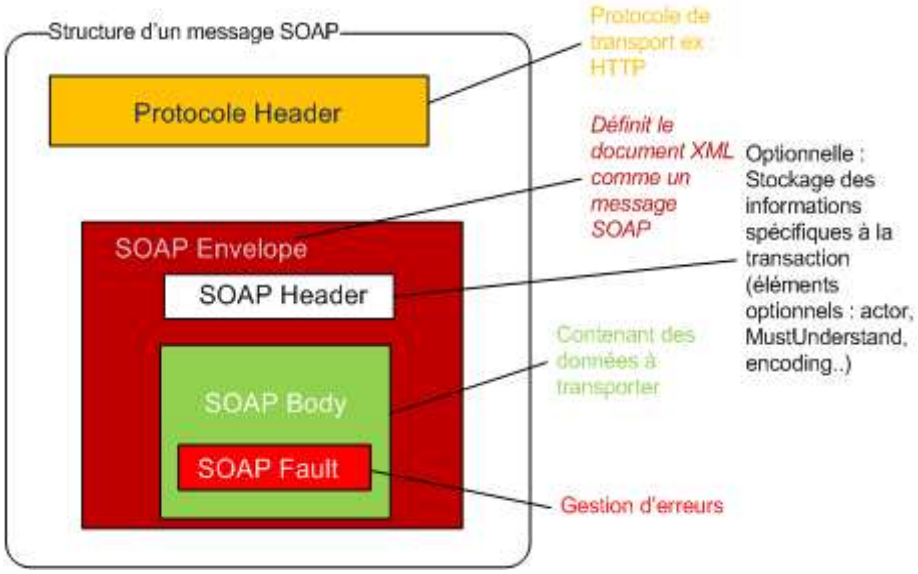


Figure III.5 : Structure de l'enveloppe SOAP

• **Enveloppe**: c'est lui qui contient le message et ses différents sous-blocs. Il s'agit du bloc racine XML. Il peut contenir un attribut *encodingStyle* dont la valeur est une URL vers un fichier de typage XML qui décrira les types applicables au message SOAP.

• **Header (en-tête):** c'est un bloc optionnel qui contient des informations d'en-têtes sur le message. S'il est présent, ce bloc doit toujours se trouver avant le bloc Body à l'intérieur du bloc Enveloppe.

• **Body (corps):** c'est le bloc qui contient le corps du message. Il doit absolument être présent de manière unique dans chaque message et être contenu dans le bloc Enveloppe. SOAP ne définit pas comment est structuré le contenu de ce bloc. Cependant, il définit le bloc Fault qui peut s'y trouver.

• **Fault (faute)**: ce bloc est la seule structure définie par SOAP dans le bloc Body. Il sert à reporter des erreurs lors du traitement du message, ou lors de son transport. Il ne peut apparaître qu'une seule fois par message. Sa présence n'est pas obligatoire. [35]

6.3 WSDL (Web Services Description Language)

Un document WSDL est un document XML contenant toutes les informations nécessaires pour contacter et utiliser un service, indépendamment de toute plate-forme ou langage de programmation. [33]

6.3.1 STRUCTURE D'UN DOCUMENT WSDL

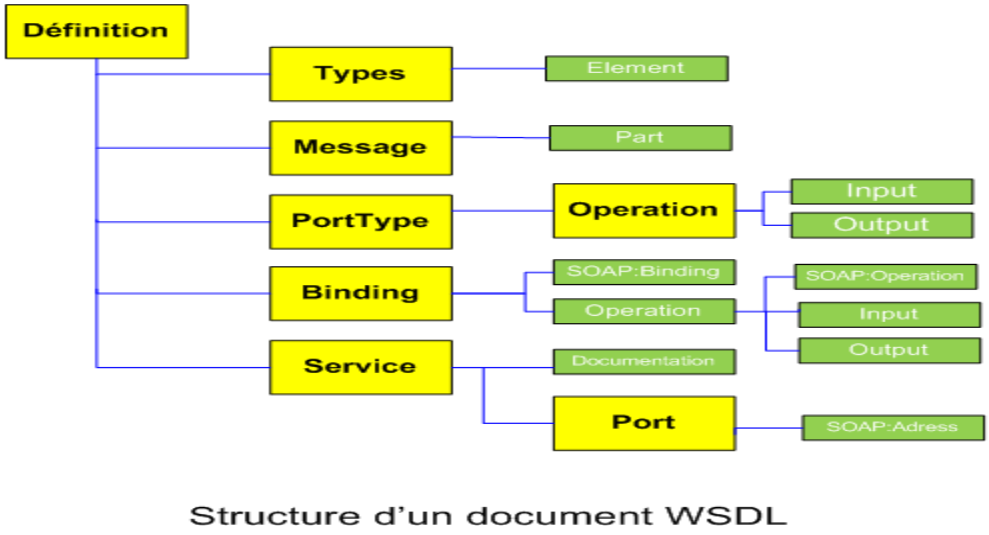


Figure III.6 : structure d'un document WSDL

- **`Définitions`**: Il contient le nom du service décrit et les namespaces faisant référence aux types utilisés dans le document.
- **`Types`**: fournit des définitions de type de données utilisé pour décrire les messages échangés. Il existe deux sortes de types, l'une est type simple, et l'autre est type complexe qui est composé des types simples.
- **`Message`**: représente une définition abstraite des données transmises.
- **`Operation`**: est une description abstraite d'une fonction supportée par le SW.
- **`PortType`**: est un ensemble d'opérations abstraites. Chaque opération fait référence à un message d'entrée et de sortie.
- **`Binding:`** précise le protocole concret et les spécifications pour les opérations et les messages définis par un `PortType` particulier.
- **`Port`**: associe une adresse unique à un élément `Binding`.
- **`Service`**: est utilisé pour regrouper un ensemble de ports associés. [34]

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

6.4 UDDI (Universal Description, Discovery and Integration)

Universal Description Discovery and Integration, connu aussi sous l'acronyme UDDI, est un annuaire de services fondé sur XML et plus particulièrement destiné aux services Web.

La spécification UDDI part d'une initiative lancée par ARIBA, Microsoft et IBM. Cette spécification n'est pas gérée par le W3C mais par le groupe appelé OASIS (Organization for the Advancement of Structured Information Standards). [33]

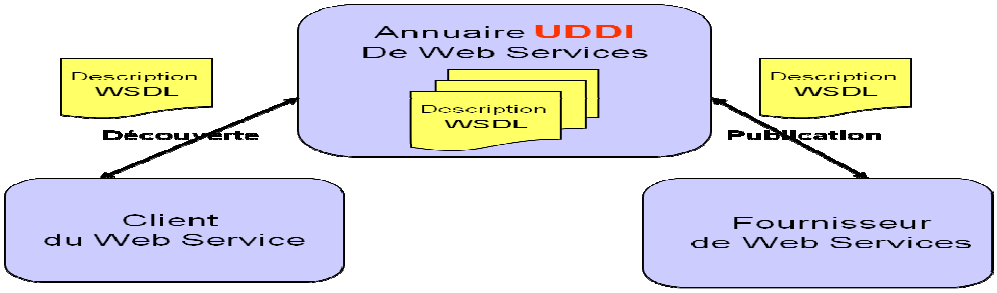


Figure III.7 : Annuaire UDDI de web service.

6.4.1 LA STRUCTURE D'UN UDDI

Conceptuellement, une organisation peut enregistrer trois types de renseignement dans un répertoire UDDI:

- ***Les pages blanches (BusinessEntity) :*** renseignement de base sur les contacts et les identifiants d'une organisation ; y compris le nom de l'organisation, son adresse, les renseignements sur les ressources et les identifiants unique. Ces renseignements permettent aux requérants de découvrir des services web basés sur l'identification d'une organisation.
- ***Les pages jaunes (BusinessService) :*** renseignements décrivant un service web en utilisant des catégorisations différentes. ces renseignements permettent aux requérants de découvrir des services web à partir des catégorisations (tel que par ministère, organisme ou partenaire).
- ***Les pages vertes (bindingTemplate) :*** renseignements techniques décrivant les comportements et les fonctions supportées d'un service web d'une organisation.ces renseignements incluent des liens aux regroupements d'information des services web ainsi que la localisation de ces derniers. [5]

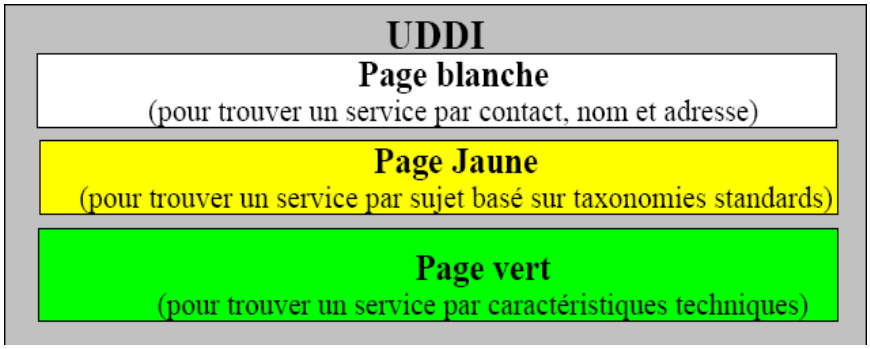


Figure III.8: Structure d'UDDI

Le protocole utilise 3 fonctions de base :

- Publish (publier) pour enregistrer un nouveau service
- Find (découvrire) pour interroger l'annuaire.
- Bind (lier) pour effectuer la connexion entre l'application cliente et le service

7 Conclusion

Un "Service Web" est une application logicielle à laquelle on peut accéder à distance à partir de différents langages basés sur XML.

De nombreuses normes sous-tendent cette architecture : "SOAP" pour l'échange de messages, "XML" langage de base pour décrire tous les documents sur lesquels les messages sont construits, "HTTP" pour transporter les messages, "WSDL" pour décrire les services et enfin "UDDI" pour les publier.

1 INTRODUCTION :

UML (*Unified Modeling Language*) est un langage de modélisation orientée objet développé en réponse à l'appel à propositions lancé par l'OMG (*Object Management Group*) dans le but de définir la notation standard pour la modélisation des applications construites à l'aide d'objets. Il est hérité de plusieurs autres méthodes telles qu'OMT (*Object Modeling Technique*) et OOSE (*Object Oriented Software Engineering*) et Booch. Les principaux auteurs de la notation UML sont Grady Booch, Ivar Jacobson et Jim Rumbaugh. [3]

1.1 LES DIAGRAMMES EN UML

1.1.1 LES DIAGRAMMES STATIQUES

- **les diagrammes des cas d'utilisation (use cases) :**

Un **cas d'utilisation** permet de mettre en évidence les relations fonctionnelles entre les acteurs et le système étudié. [6]

- **les diagrammes de classe :**

Les diagrammes de classes expriment de manière générale la structure statique d'un système, en termes de classes et de relations entre ces classes. [6]

- **les diagrammes d'objets :**

Les diagrammes des objets modélisent des exemples de classes. Ce type de diagramme est employé pour décrire le système à un instant particulier. [6]

- **Diagramme des composants :**

Un genre de diagramme qui modélise l'implémentation et le déploiement du système. [6]

- **Diagramme de déploiement :**

Le diagramme de déploiement modélise les composants matériels utilisés pour implémenter un système et l'association entre ces composants. [6]

1.1.2 LES DIAGRAMMES DYNAMIQUES :

- **Diagramme des séquences :**

Les diagrammes des séquences documentent les interactions à mettre en œuvre entre les classes pour réaliser un résultat, tel qu'un cas d'utilisation. [6]

- **Diagramme des collaborations :**

Comme les autres diagrammes comportementaux, les diagrammes de collaboration modélisent les interactions entre les objets. [6]

- **Diagramme d'état :**

Les diagrammes d'état sont utilisés pour documenter les divers modes ("état") qu'une classe peut prendre, et les événements qui causent une transition d'état. [6]

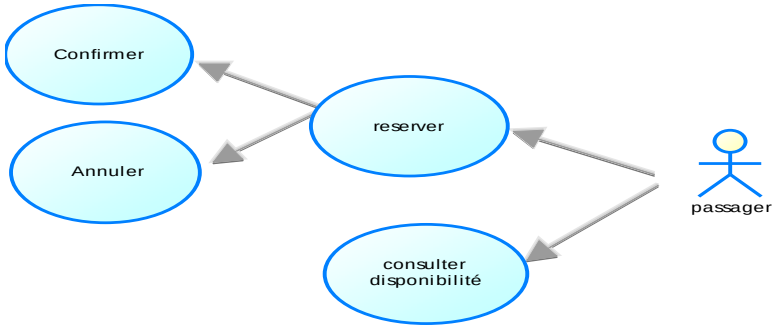
- **Diagramme d'activité :**

Les diagrammes d'activité sont utilisés pour documenter le déroulement des opérations dans un système, du niveau commercial au niveau opérationnel (de haut en bas). [6]

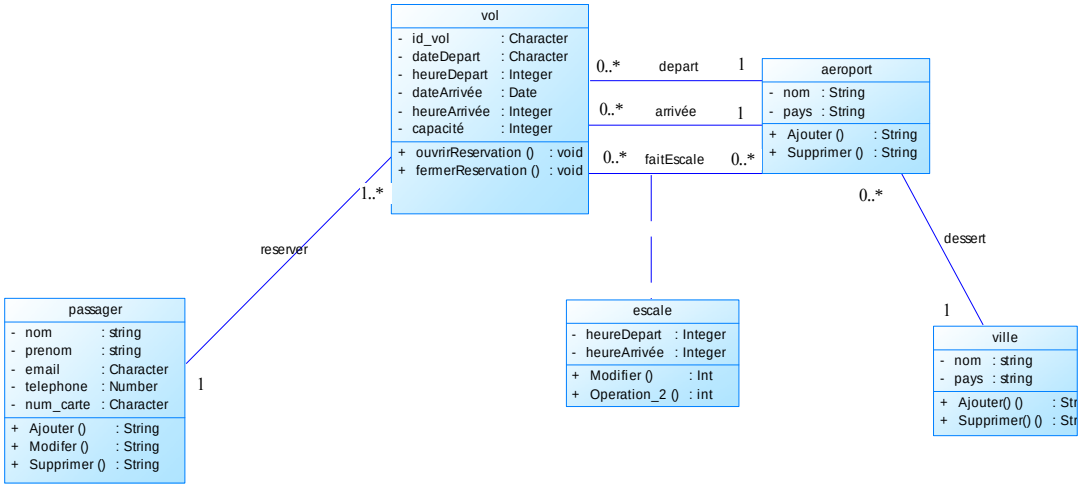
1.2 MODELISATION DE L'APPLICATION

Notre application concerne la réservation d'un voyage. Pour réserver Il suffit de sélectionner la ville de départ et d'arrivée, d'indiquer vos dates de voyage et de choisir votre itinéraire. Pour accomplir cette l'opération de réservation, le client doit saisir des formulaires qui portent sur des informations personnelles, numéro de téléphone, email,…. et doit préciser le mode de paiement convenable. Si tout passe bien alors vous êtes maintenant prêt à voyager.

1.2.1 DIAGRAMME CAS D'UTILISATION :



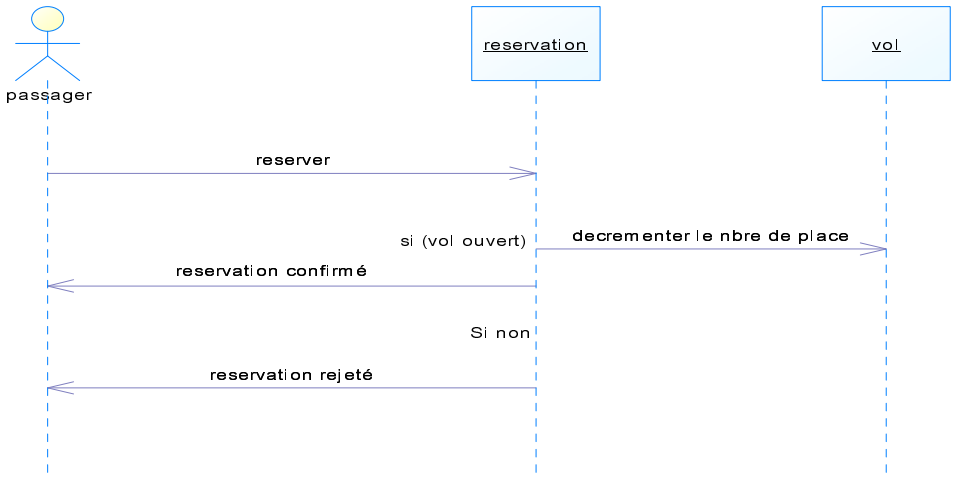
1.2.2 DIAGRAMME DES CLASSES :



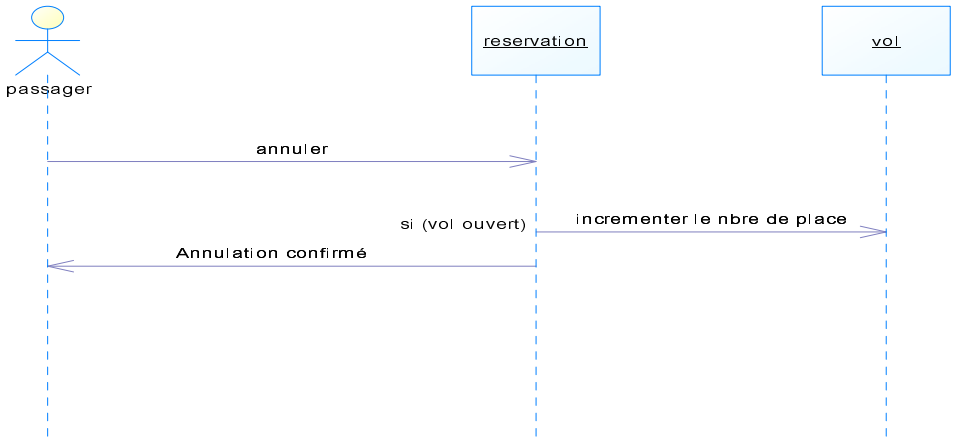
1.2.3 DIAGRAMME DE SEQUENCE :

Ouvrir un vol :

Réservation simple :



Annulation :



2 Implémentation

On a implémenté notre application en utilisant les outils suivant :

JAVA Eclipse, Visual studio2008, apache-tomcat-6.0.32, axis2-1.4.1, jdk1.6, MS access. [2]

2.1 Éclipse

Eclipse IDE est un environnement de développement intégré libre (le terme *Eclipse* désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions. [3]

La création de services web sous la plateforme Eclipse nécessite la configuration d'Apache Tomcat et Axis:

2.1.1 Apache Tomcat

Tomcat est un conteneur de servlet J2EE. Issu du projet Jakarta, il est désormais un projet principal de la fondation Apache. Tomcat implémente les spécifications des servlets et des JSP de Sun Microsystems. Il inclut des outils pour la configuration et la gestion, mais peut également être configuré en éditant des fichiers de configuration XML. Comme Tomcat inclut un serveur HTTP interne, il est aussi considéré comme un serveur HTTP.

2.1.2 Apache Axis

Axis est un projet de l'Apache Software Foundation. C'est un package Java libre qui fournit :

- Un environnement pouvant soit fonctionné comme un serveur SOAP/Rest indépendant soit comme un plug-in de moteurs de serveur (en particulier Tomcat).
- Une API pour développer des services web SOAP RPC ou à base de messages SOAP.

- Le support de différentes couches de transport : HTTP, FTP, SMTP, POP et IMAP, ...
- La sérialisation/désérialisation automatique d'objets Java dans des messages SOAP.
- Des outils pour créer automatiquement les WSDL correspondant à des classes Java ou inversement pour créer les classes Java sur la base d'un WSDL. [36]

Axis est publié sous licence Apache 2.0. [6]

2.1.3 JDK

JDK (Java Development Kit) Logiciel édité par Sun pour le développement d'application en Java.

2.2 MS ACESS

MS Access est un logiciel utilisant des fichiers au format Access (extension de fichier *mdb* pour Microsoft DataBase (extension *.accdb depuis la version 2007)). Il est compatible avec les requêtes SQL (sous certaines restrictions) et dispose d'une interface graphique pour saisir les requêtes (QBE - Query par exemple). Il permet aussi de configurer, avec des assistants ou librement, des formulaires et sous-formulaires de saisie, des états imprimables (avec regroupements de données selon divers critères et des totalisations, sous-totalisations, conditionnelles ou non), des pages html liées aux données d'une base, des macros et des modules VBA. [1]

2.3 Visual studio (.net)

Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications Web ASP.NET, des Services Web XML, des applications bureautiques et des applications mobiles. Visual Basic, Visual C++, Visual C# et Visual J# utilisent tous le même environnement de développement intégré (IDE, Integrated Development Environment), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du Framework .NET, qui fournit un accès à des technologies clés simplifiant le développement d'applications Web ASP et de Services Web XML grâce à Visual Web Developer. [7]

3 Réalisation de l'application

3.1 Partie service web

- **Méthode chercher_vol :**

La méthode chercher_vol est devisée en deux parties, la première partie qui établir une connexion à notre base de données et la deuxième parties qui permet d'afficher les vols disponibles à travers la méthode interne « Select from vol » qui retourne les données de la table vol de notre base de donnée.

- **Méthode réserver :**

La méthode réserver commence par la connexion avec notre base de données, ensuite ajouter les informations du passager dans la table passager de notre base de donnée on utilisant la méthode interne « insert into passager ».

- **Le fichier WSDL :**

On obtient la page suivante qui représente le fichier de description de notre service web en exécutant le code source serveur :

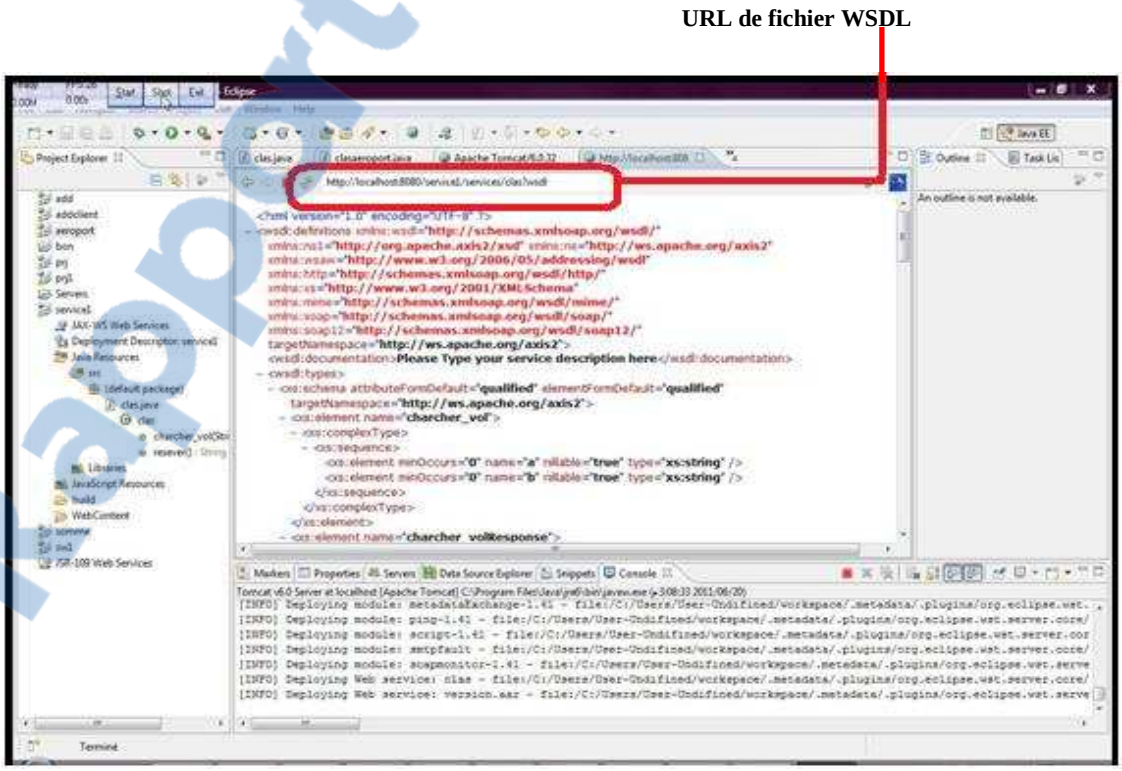


Figure IV.1 : Fichier WSDL

Rapport-Gratuit.com

3.2 Partie client

Le client .net est une application qui va appeler chacune des méthodes exposées par le Web Service dans le but de voir la représentation en .net des types Java exportés. Après avoir créé le projet aéroport, la première étape consiste à ajouter une référence au Web Service java.

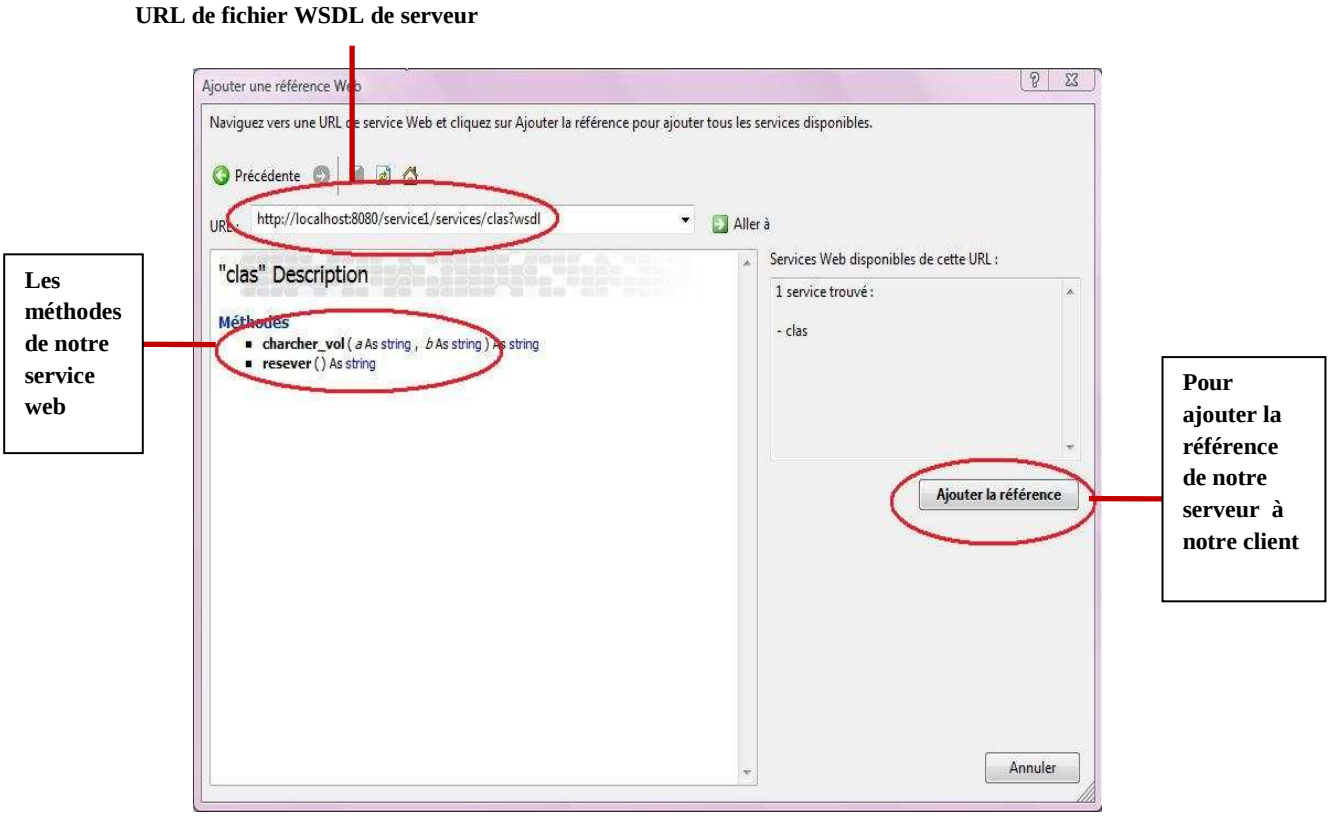


Figure IV.2 : Ajouter une référence web

Le client .Net appelle chacune des méthodes du Web Service pour voir si tout se passe bien.

Voila c'est la première interface dans notre application :

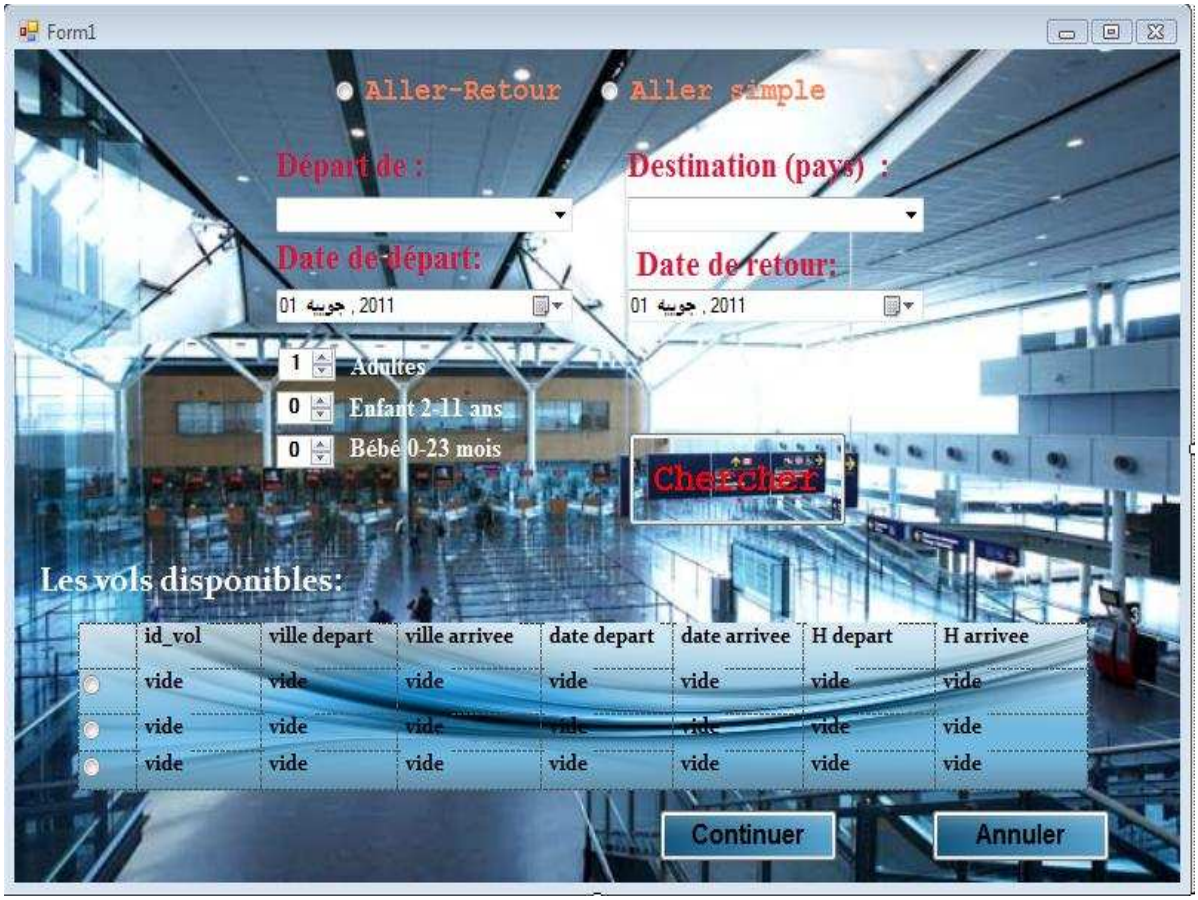


Figure IV.3 : cherche vol

Après le choix de vol il s'affiche une autre interface pour remplir les informations personnelles

Figure IV.4 : Réservation

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Quand le passager termine de remplir ces informations il doit choisir le mode de payement

Figure IV.5 : Mode payement

Si le passager choisit le mode cash il va s'afficher cette boite de dialogue

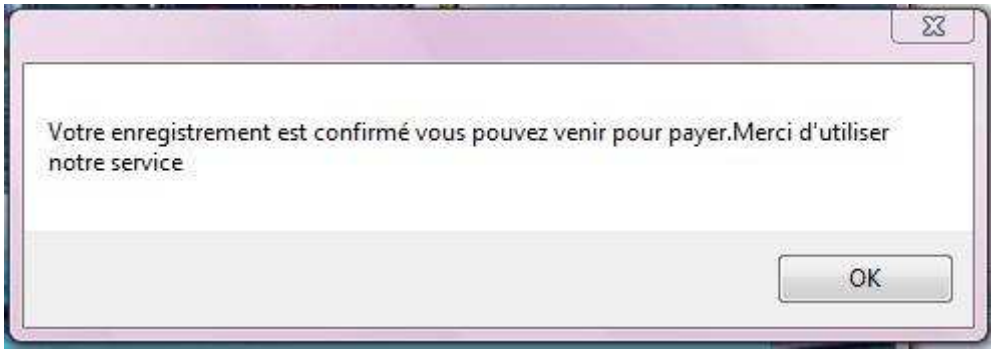


Figure IV.6 : Confirmation

Si le passager choisit le mode carte-bancaire il s'affiche une autre interface pour remplir les informations su r la carte-bancaire

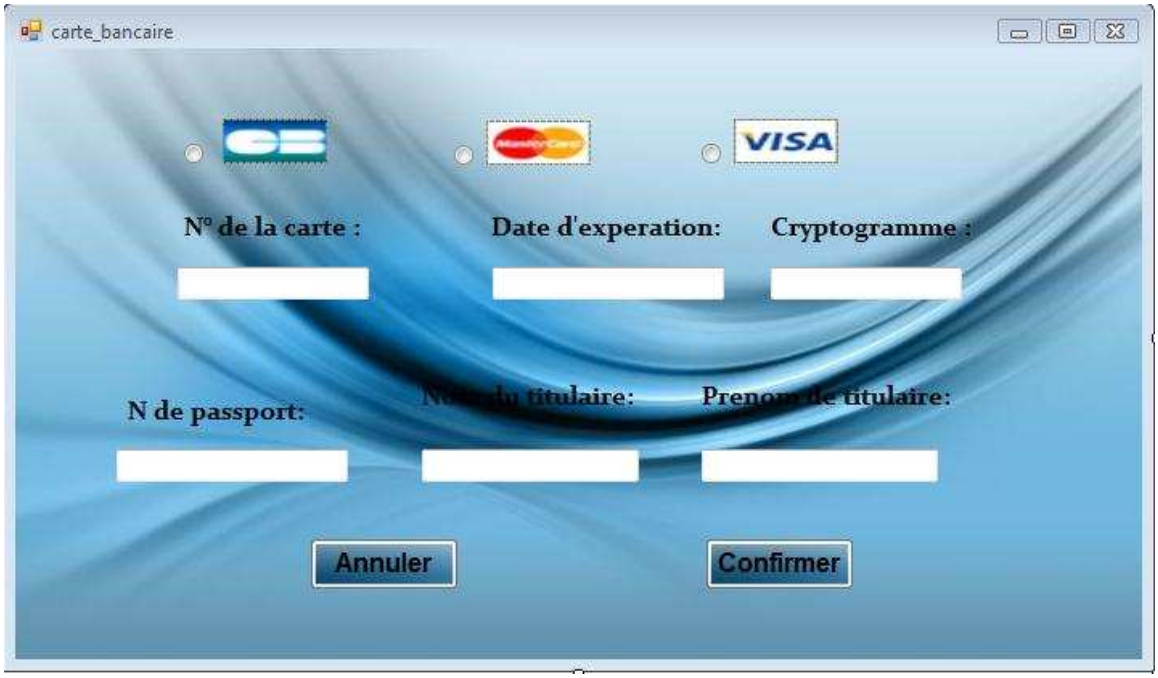


Figure IV.7 : Information concerne la carte-bancaire

Et finalement le vol est confirmé et il s'affiche une boite de dialogue contient le message de remerciement

Figure IV.8: Remerciement

En conclusion, il est nécessaire de faire le point sur la technologie des services Web. Les services Web est un terme qui décrit un ensemble de protocoles standards utilisés pour établir un domaine d'intégration des applications.

L'un des facteurs ayant contribué au succès des services Web est sans doute l'utilisation des standards Internet tels que XML et HTTP. En conséquence, tout système capable d'analyser du texte et de communiquer via un protocole de transport Internet standard peut communiquer avec un service Web. XML a engendré l'apparition de nouveaux protocoles tels que SOAP pour l'échange de messages, WSDL pour la description de services et UDDI pour la publication et la découverte de services.

Notre application concerne la réservation d'un biais : tout d'abord le passager va chercher des vols disponibles, ensuite il choisie le vol qui correspond à ses besoins et remplir ses informations (nom, prénom, email,…), il doit préciser le mode de paiement convenable. Si tout passe bien alors votre réservation est valide et maintenant je vous souhaite un bon voyage.

LES OUVRAGES:

[1] L i b e r o M a e s a n o, C h r i s t i a n B e r n a r d, X a v i e r L e G a l l e s, - Eyrolles - Services Web avec J2EE et .NET,2003

[2] Jérôme Molière, - Eyrolles - Les cahiers du programmeur J2EE, 2005

[3] Jean Michel DOUDOUX, Développons en Java avec Eclipse, 15/12/2008

[4] Jean Michel DOUDOUX, Développons en Java avec Eclipse, 30/12/2010

[5] **:** Hubert Kadima, Valérie monfort, Les web services techniques, démarches et outils XML, WSDL, UDDI, UML, Dunod, 2003.

LES MÉMOIRES :

[6] Cheikh et Djannane , La conception et la réalisation d'une application services web sous la plateforme « Java », Mémoire de licence ,université Abou Bakr Belkaid ,juillet 2010.

[7] : Bouzbiba , La conception et réalisation d'une application d'un système de la gestion hôtelière avec la méthode UML, Mémoire de licence ,2003.

[8] Jean-Marc Geib - Christophe Gransart - Philippe Merle, CORBA : des concepts à la pratique**,** Laboratoire d'Informatique Fondamentale de Lille,Université des Sciences et Technologies de Lille, Mémoire de l'ingenieur, France, 1997.

[9] Belkhouja et Benali, La conception et la réalisation des services web sous l'approche « .NET ». Mémoire de licence, université Abou Bekr Belkaid, département informatique, juillet 2009.

[10] M. Rached Achouri, Conception et développement de la publication des services d'intermédiation documentaire via des services web, École supérieur des communications de Tunis, Mémoire de licence ,2006/2007.

LES SITES WEB:

[11]: http://www.emse.fr/~vercouter/cours/intergiciels/intergiciels.pdf.

[12]: http://remi.leblond.free.fr/probatoire/node5.html.

[13]: http://www.bd.enst.fr/~dombd/Cours/Applications/3tiers/index.html.

[14]: http://rangiroa.essi.fr/cours/systeme2/02-rpcgen.pdf.

[14]: http://www.centralweb.fr .

[16]: http://referencement360.com/van-laethem/ergologie/Corba.pdf.

[17]: http://search.mywebsearch.com/mywebsearch/GGmain.jhtml?st=hp&searchfor=architecture%20corba&ptnrS=zcman000&ptb=v6LgfOxOSku4bdqvGUWLbA&n=77cecab9.

[18]: http://www.infosys.tuwien.ac.at/Research/Corba/software.html.

[19]: http://www.commentçamarche.net.

[20]: http://pagesperso-orange.fr/visual.basic/mandcom.htm#_ Toc416940505.

[21]: http://fr.wikipedia.org/wiki/systèmedistribuées.

[22]: http://www.dotnetguru.org.

[23]: http://www.tetraedre.com/advanced/xml.php.

[24]: http://www.w3c.org/XML.

[25]: http://fr.wikipedia.org/wiki/namesspaces.

[26]: http://www.w3c.org/XMLschémas.

[27]: http://fr.wikipedia.org/wiki/Cours javaetXML.

[28]: www-igm.univ.fr/dr/XPOSE2004/woolams/presentation.html.

[29]: http://www-igm.univ mlv.fr/~dr/XPOSE2004/ woollams/ objectifs.html.

[30]: http://www.smart-service.fr/exemple-de-service-web-a-quoi-sert-un-web-service-gestion-de-base-de-sonnee-web-service-annuaire-uddi-gestion-web-services-gestion-base-de-donnees-distantes_404.html.

[31]: http://www-inf.it-sudparis.eu/cours/WebServices.

[32]: http://fr.wikipedia.org/wiki/HTTP.

[33]: http://www-igm.univ-mlv.fr/~dr/XPOSE2004/woollams/soap.html.

[34]: http://www.emse.fr/~vercouter/cours/webservices /webservices.pdf.

[35]: http://www.expert-consulting.net/doc/IntroductionJ2EE.pdf.

RESUME

Un service web est un programme informatique permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués. Il s'agit donc d'un ensemble de fonctionnalités exposées sur internet, par et pour des applications ou machines, sans intervention humaine, et de manière synchrone.

Mots clés : service web, applications, système hétérogène, environnements distribués.

ABSTRACT

A web service is a computer program that allows communication and exchange of data between heterogeneous systems and applications in distributed environments. So It is a set of functionalities displayed on the Internet, by and for applications or machines, without human intervention, and synchronously.

Keywords: web service, applications, heterogeneous systems, distributed environments.

تلخيص

خدمة الويب هي برنامج الكمبيوتر الذي يتيح التواصل وتبادل البيانات بين التطبيقات و الأنظمة غير المتجانسة في بيئات موزعة. ومن ثم فهي مجموعة من وظائف تعرض على شبكة الإنترنت، عن طريق التطبيقات أو الأجهزة، دون تدخل الإنسان، وبشكل متزامن.

الكلمات المفتاحية :خدمات الويب، التطبيقات الأنظمة غير المتجانسة ، بيئات موزعة .