



UNIVERSITE MOHAMED BOUDIAF - M'SILA
FACULTE DES MATHEMATIQUES ET
DE L'INFORMATIQUE



DEPARTEMENT D'INFORMATIQUE

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Technologie de l'Information et de Communication

Par : DEHMECHE Abdelbaqi

SUJET

**Service web Embarqué (Technologie SOAP sous Android)
pour la gestion d'une Bibliothèque MI**

Soutenu publiquement le : / /2016 devant le jury composé de :

SAHRAOUI M

ABBACHE Farid

DEBBI Emad Addin

.....

Université de M'sila

Université de M'sila

Université de M'sila

Université de M'sila

Président

Rapporteur

Examineur

Examineur

Promotion : 2015 /2016

Dédicace

Nous remercions notre dieu d'avoir donné la capacité d'écrire et de réfléchir, la force d'y croire, la patience d'aller jusqu'au bout du rêve et le bonheur.

Nous dédions ce modeste travail à celle qui nous a donné la vie, symbole de tendresse, qu'ils ont sacrifiée pour notre bonheur réussite, à notre mère...

À notre père, école de notre enfance, qui ont été notre ombre durant toutes les années des études, et qui ont veillé tout au long de vie à nous encourager, à nous donner l'aide et à nous protéger.

Que dieu les gardes et les protège.

À notre adorable sœur

À notre frère ...

À notre amies

À tous ceux qui nous aiment

Nous dédions ce travail ...

Remerciements

Nous remercions Dieu de nous avoir accordé des connaissances de la science et de nous avoir aidés à réaliser ce travail.

Au terme de ce modeste travail nous tenons à remercier chaleureusement et respectivement tous ceux qui 'ont contribué de près ou de loin à la réalisation de ce modeste projet de fin d'étude, à savoir notre encadreur MR, ABBACHE Farid.

Ce travail fut difficile mais très bénéfique à tout point de vue.

TABLE DES MATIERES

INTRODUCTION GENERALE.....	1
----------------------------	---

CHAPITRE 1 : ETUDE DE L'EXISTANT

1 Introduction.....	3
2 Présentation de la bibliothèque.....	3
2.1 Définition de la bibliothèque.....	3
2.2 Missions de la bibliothèque.....	3
2.3 Organigramme de la bibliothèque.....	4
2.4. Services de la bibliothèque.....	4
2.5. Moyens disponible dans la bibliothèque.....	4
2.5.1 Moyens humains.....	4
2.5.2 Moyens matériels.....	5
3 Diagramme de flux.....	5
4 Problématique.....	6
5 Définition des besoins fonctionnels et non fonctionnels.....	6
5.1 Les besoins fonctionnels.....	6
5.1.1 À l'Hôtesse.....	6
5.1.2 À l'adhérent.....	7
5.2 Les besoins non fonctionnels.....	7
6 Solution.....	8
7 Conclusion.....	8

CHAPITRE 2 : SERVICE WEB EMBARQUÉ

1 Introduction.....	10
2 Service web et Appareils embarqué.....	10
2.1 Les Services Web.....	10
2.2 Appareils embarqués [8].....	11

3 Les Principales caractéristiques des services Web sont [5].....	11
4 Architecture des Services web.....	11
5 Cycle de vie d'un service web	13
6 Les langages et les protocoles utilisés par les services web.....	13
6.1 XML – eXtensible Markup Language [6].....	13
6.2 SOAP – Simple Object Access Protocol.....	14
6.2.1 Structure d'un message SOAP	14
6.2.2 SOAP RPC.....	16
6.3 WSDL – Service web Description Language.....	19
6.3.1 Structure d'un document WSDL.....	20
6.4 UDDI: Universal Description Discovery and Integration	21
7 Les applications des Services web.....	21
8 Invocation d'un service web.....	21
9 Avantages et inconvénients.....	23
9.1 Avantages.....	23
9.2 Inconvénients.....	23
10 Conclusion.....	24

CHAPITRE 3 : CONCEPTION DU SYSTÈME

1 Introduction.....	26
2 Définition d'UML.....	26
3 Historique d'UML.....	26
4 Pourquoi choisir UML.....	27
5 Les points forts et faibles d'UML.....	27
5.1 Les points forts d'UML	27
5.2 Les points faibles d'UML	27
6 Dictionnaire de données.....	28
6.1 Définition	28

6.2 Présentations	28
7 Les diagrammes UML.....	30
7.1 Diagramme de cas d'utilisation	31
7.1.1 Définition	31
7.1.2 Présentation	32
7.2 Diagramme de séquence	33
7.2.1 Définition	33
7.2.2 Cadre d'interaction.....	33
7.2.3 Diagramme de séquence « Authentification »	34
7.2.4 Diagramme de séquence « Ajoute adhérent ».....	35
7.2.5 Diagramme de séquence « Réservation»	36
7.2.6 Diagramme de séquence « Emprunt »	38
7.3 Diagramme de classes	39
7.3.1 Définition	39
7.3.2 Présentation	40
8 Schéma relationnel.....	40
9 Conclusion.....	41

CHAPITRE 4 : LA REALISATION

1 Introduction.....	43
2 Les outils utilisés.....	43
2.1 L'environnement de travail	43
2.1.1 Eclipse	43
2.1.2 Android Studio	44
2.1.3 Gradle.....	44
2.1.4 Software de développement kit (SDK)	45
2.1.5 Emulateur	45
2.2 Serveur	46

2.2.1 WAMP Server	46
2.2.2 Apache Tomcat	46
2.2.3 Axis 2	47
2.3 Technologie	47
2.3.1 MySQL.....	47
2.3.2 PhpMyAdmin.....	48
2.3.3 Java.....	48
2.3.4 Java EE.....	49
2.3.5 Html	50
2.3.6 CSS.....	50
2.3.7 JSP.....	51
2.3.8 Servlet	51
2.3.9 Java Scripte	51
2.3.10 JQuery	52
2.3.11 Ajax.....	52
3 Les Interfaces	53
3.1 Interface d'Authentification	53
3.2 Interface principale Hôtesse.....	53
3.3 Interface principale d'Adhèrent.....	54
3.4 Interface de liste livre	54
3.5 Interface d'ajoute livre	55
3.6 Interface Recherche livre	55
3.7 Interface Réserve livre.....	56
3.8 Interface validé la réservation	56
3.9 Interface Profile	57
4 Conclusion	57

LISTE DES FIGURES

Figure 1.1 : Ressources humaines de la bibliothèque.....	4
Figure 1.2 : diagramme de flux	5
Figure 2.1 : Architecture d'un service web	13
Figure 2.2 : Structure D'un document XML.....	14
Figure 2.3 : Structure d'un message SOAP	15
Figure 2.4 : La structure d'un message SOAP	16
Figure 2.5 : Structure de SOAP RPC	16
Figure 2.6 : Exemple de méthode sur le service.....	17
Figure 2.7 : Structure de la Requête SOAP	17
Figure 2.8 : Structure de réponse SOAP	18
Figure 2.9 : Structure de la réponse SOAP en cas d'erreur	19
Figure 2.10 : architecture Proxy SOAP	19
Figure 2.11 : Figure structure d'un document WSDL	20
Figure 2.12 : Les étapes d'invocation d'un service web	23
Figure 3.1 : Historique d'UML	26
Figure 3.2 : Hiérarchie des diagrammes UML	30
Figure 3.3 : Diagramme de cas d'utilisation	32
Figure 3.4 : Diagramme de séquence « Authentification ».....	34
Figure 3.5 : Diagramme de séquence « Ajoute adhérent ».....	34
Figure 3.6 : Diagramme de séquence « Réservation »	35
Figure 3.7 : Diagramme de séquence « Emprunt »	38
Figure 3.8 : Diagramme de classes d'application.....	40
Figure 4.1 : Eclipse.....	43
Figure 4.2 : Android Studio.....	44
Figure 4.3 : Android SDK Manager	45
Figure 4.4 : Emulateur	45
Figure 4.5 : WAMP server	46
Figure 4.6 : Apache Tomcat	46
Figure 4.7 : Axis 2	47
Figure 4.8 : MySQL	47
Figure 4.9 : phpMyAdmin.....	48
Figure 4.10 : JAVA	48

Figure 4.11 : Java EE.....	49
Figure 4.12 : HTML	50
Figure 4.13 : CSS	50
Figure 4.14 : Servlet	51
Figure 4.15 : Java Scripte	51
Figure 4.16 : JQuery.....	52
Figure 4.17 : Ajax.....	52
Figure 4.18 : Interface d'Authentification.....	53
Figure 4.19 : Interface principale d'Hôtesse	53
Figure 4.20 : Interface principale d'Adhèrent	54
Figure 4.21 : Interface de liste livre	54
Figure 4.22 : Interface d'ajoute livre.....	55
Figure 4.23 : Interface recherche livre	55
Figure 4.24 : Interface réservation.....	56
Figure 4.25 : Interface validé réservation.....	56
Figure 4.26 : Interface profile.....	57

LISTE DES TABLEAUX

Table 3.1 : Dictionnaire de donnée	30
Table 3.2 : Le formalisme du diagramme de cas d'utilisation	31
Table 3.3 : Le formalisme du diagramme de séquence	33
Table 3.4 : Le formalisme du diagramme de séquence	39

LISTE DES ABBREVIATIONS

ABREVIATION	SIGNIFICATION
B2B	Business to Business
B2C	Business to Consumer
CORBA	Common Object Request Broker Architecture
CSS	Cascading Style Sheets
DCOM	Distributed Component Object Model
HTML	HyperText Mark-Up Language
http	HyperText Transfer Protocol
IBM	International Business Machines
J2EE	Java 2 Enterprise Edition
JSP	Java Server Page
RMI	Remote Method Invocation
SOA	Service Orienté Architecture
SOAP	Simple Object Access Protocol
UDDI	Universal Description Discovery and Integration
WSDL	Web Services Description Language
XML	Extensible Markup Language, langage de balisage extensible

INTRODUCTION GENERALE

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. Ces protocoles correspondent à des composants logiciels qui peuvent être combinés, grâce à un langage de composition, pour former de nouveaux services plus élaborés. L'objectif de notre travail est de réaliser un service Web pour la gestion embarqué (technologie soap sous Android) pour la bibliothèque de la faculté mathématique et informatique aux utilisateurs du net, afin de leur permettre de consulter les livres disponibles dans la bibliothèque et d'emprunter les livres désirés par les abonnés inscrits à cette bibliothèque après avoir fait une demande d'emprunt.

Ainsi le travail dans ce mémoire s'articule autour de quatre chapitres.

Le premier chapitre consiste d'une étude générale sur les différents concepts qui étaient relié aux problématiques de notre projet de fin d'études

Le deuxième chapitre détaille les concepts de base des services Web ainsi que les trois principaux standards : SOAP comme un protocole de transport, WSDL comme un langage de description des services Web et UDDI comme un annuaire pour assurer la publication et la découverte des services Web.

Le troisième chapitre est consacré à la conception de notre service Web pour la gestion de la bibliothèque.

Le quatrième chapitre décrit L'implémentation de notre service Web.

Enfin, nous concluons notre travail par une récapitulation des solutions proposées pour la gestion de la bibliothèque. Des perspectives futures quant à l'amélioration de notre travail sont dégagées.

CHAPITRE 1

ETUDE DE L'EXISTANT

1 Introduction

La gestion de la bibliothèque et les différents tâche effectuée par Adhérent est un objet très importants dans ces jours. Pour cela nous avons essayé de faire une application web pour résoudre le problème de gestion de bibliothèque différent de la méthode classique.

Dans ce chapitre nous avons présenté une étude générales sur les différentes concepts qui étaient relater à les problématiques de notre projet de fin d'études .telle que nous avons situé une vision général sur la bibliothèque, définir les problèmes rencontrés et nous essayons de proposer des solutions.

2 Présentation de la bibliothèque

L'objectif du présent projet est de concevoir et de réaliser une application permettant l'automatisation de gestion de bibliothèque de la Faculté de mathématique et informatique en utilisant la méthode d'informatisation UML.

2.1 Définition de la bibliothèque

Le terme bibliothèque est apparu en Grèce qui signifie Coffret du livre, mais avant l'antiquité grecque il y avait déjà en Irak et en Egypte de véritables bibliothèques. La bibliothèque est un organisme qui collecte des documents sous forme écrite comme par exemple des manuscrits, des livres, des thèses, etc. La bibliothèque a aussi pour mission de conserver et classer ces ressources et de les mettre à la disposition des lecteurs soit par consultation sur place ou bien soit par prêt à domicile. La bibliothèque dépasse largement son rôle comme instrument de travail, et est considérée comme conservatoire du patrimoine culturel et scientifique de l'humanité.

2.2 Missions de la bibliothèque

La bibliothèque assure les tâches suivantes ;

- La gestion des ouvrages et des adhérents
- L'acquisition, le traitement et la diffusion de la documentation répondant aux besoins spécifiques de ses utilisateurs.

2.3 Organigramme de la bibliothèque

La bibliothèque de la Faculté de mathématique et informatique est structurée en deux services principaux, à savoir le service d'orientation et le service de gestion du fond documentaire, comme illustré sur la Figure 1.1.

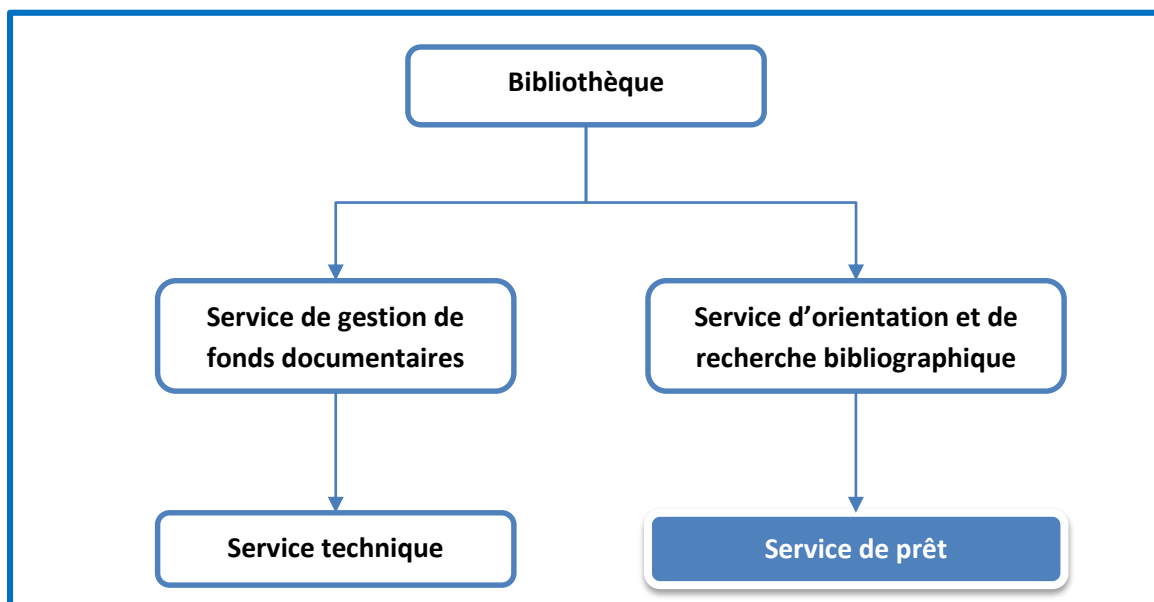


Figure 1.1 : Ressources humaines de la bibliothèque

2.4. Services de la bibliothèque

La Bibliothèque comporte 2 services principaux :

- Service de prêt
- Service technique

2.5. Moyens disponible dans la bibliothèque

2.5.1 Moyens humains

Le personnel de la bibliothèque est constitué de 12 membres

- 02 agents techniques.
- 03 assistants de bibliothèque
- 05 agents de bibliothèque
- Un chef service-un attaché de bibliothèque

2.5.2 Moyens matériels

- 02 micro-ordinateurs pour le saisi des ouvrages
- 02 micro-ordinateurs pour le saisi des thèses
- 07 micro-ordinateurs pour les étudiants
- Deux salles de lecteurs
- Rayonnages pour fonds documentaires
- Un comptoir pour l'accueil des utilisateurs

3 Diagramme de flux

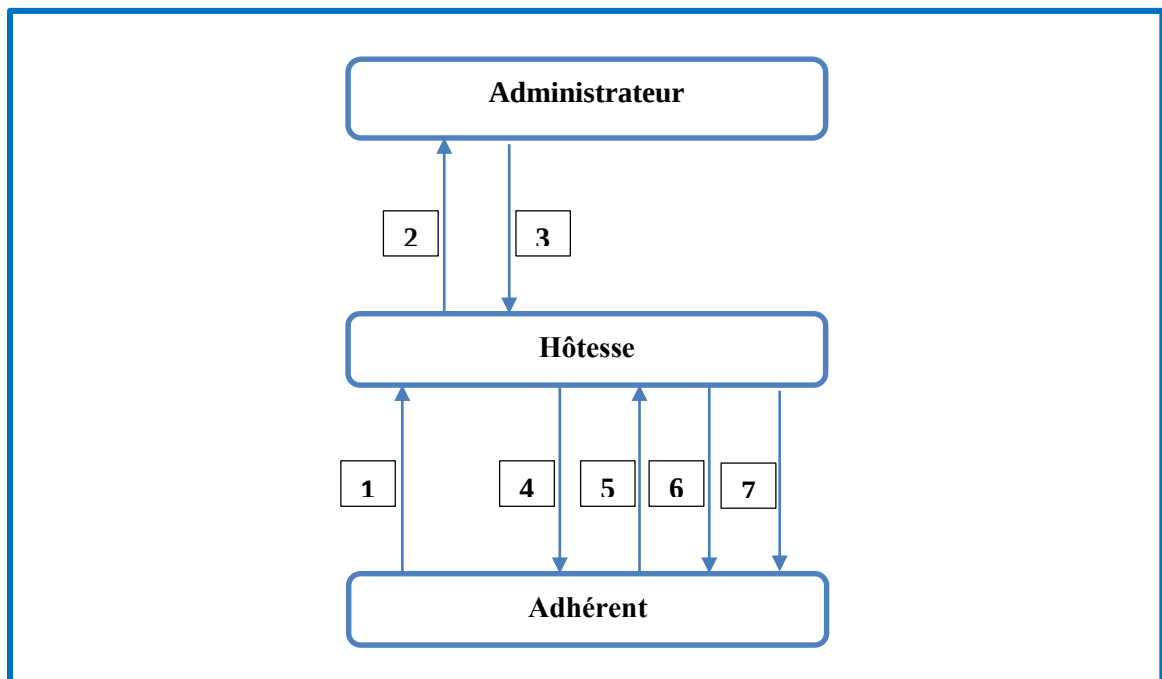


Figure 1.2 : diagramme de flux

- 1- Dossier d'inscription
- 2- Demande signé la carte
- 3- Carte signé
- 4- Carte adhérent (Abonné)
- 5- Demande de prêt
- 6- Ouvrage
- 7- Fiche de prêt

4 Critique

- La méthode de gestion de la bibliothèque, décrite plus haut, est archaïque de nos jours. Nous remarquons qu'il n'existe pas de base de données pour stocker les informations concernant les ouvrages et les membres, car l'outil Microsoft Excel n'est pas à même de jouer le rôle de système de gestion de base de données.
- Il n'existe pas de système de rappel des dates de retour d'emprunts déjà à terme, ce qui constitue un grand handicap car les dates de sortie et de rentrée sont des informations pertinentes.
- Les retours des livres ne sont pas enregistrés de façon correcte car la bibliothécaire ne fait qu'ajouter ou supprimer des lignes dans son tableau. Cela ne permet donc pas de recenser les documents perdus.
- Il n'existe pas non plus d'interface permettant l'identification de la bibliothécaire. Donc le système n'est pas du tout sécurisé et n'importe qui peut accéder aux informations de la bibliothèque.
- Ce système ne permet pas non plus aux membres de rechercher un document à partir d'un mot-clé. Du point de vue paramétrages, tout est fait de façon aléatoire.
- Il n'y a pas d'interface qui permettrait à la bibliothécaire d'effectuer des modifications des paramètres de la bibliothèque.

5 Définition des besoins fonctionnels et non fonctionnels

5.1 Les besoins fonctionnels

Il s'agit des fonctionnalités du système. Ce sont les besoins spécifiant un comportement d'entrée / sortie du Système.

Le système doit permettre :

5.1.1 À l'Hôtesse

- De s'identifier
- De consulter et mise à jour (ajouter, modifier et supprimer) les ouvrages
- De vérifier la disponibilité des exemplaires
- De consulter et mise à jour (ajouter, modifier et supprimer) les adhérents
- Gérer les emprunts
- D'ajouter une pénalité

- De supprimer une pénalité
- De consulter la liste des pénalités
- Définir la durée de la pénalité
- Définir la durée des emprunts
- Définir le max ouvrage à emprunter
- Consulter les ouvrages populaires (les plus sollicités)
- Consulter l'ouvrage le moins emprunté
- Consulter les nouveautés.

5.1.2 À l'adhérent

- De s'identifier
- D'effectuer une recherche selon un critère défini
- De réserver un ouvrage
- Gérer son profile

5.2 Les besoins non fonctionnels

Il s'agit des besoins qui caractérisent le système. Ce sont des besoins en matière de performance, de type de matériel ou le type de conception. Ces besoins peuvent concerner les contraintes d'implémentation (langage de programmation, type SGBD, de système d'Exploitation...)

Dans le cadre de ce travail, l'application devra être extensible, c'est-à-dire qu'il pourra y avoir une possibilité d'ajouter ou de modifier de nouvelles fonctionnalités.

L'application devra être capable de :

- Tourner en réseau
- Etre compatible avec n'importe quel système d'exploitation.
- IL faudra aussi noter que l'application devra être hautement sécurisée car les informations ne devront pas être accessibles à tout le monde.

6 Solution

Afin de corriger les problèmes présentés ci-dessus, je suis appelé à réaliser cette application qui assure les points suivants:

- Assuré la recherche et la réservation à partir l'internet ou 3G, 4G.
- Automatiser les tâches qui se traitent manuellement.
- Faciliter la recherche et l'accès aux informations à distance et par téléphone.
- Sauvegarder toutes les données relatives à la gestion des emprunts
- Sauvegarder toutes les données relatives à la gestion des réservations et des adhérents
- Minimiser les supports papiers utilisés.
- Faire toute modification (ajout, suppression, modification) automatiquement.
- Plus d'organisation dans le travail du responsable de bibliothèque
- Assure le meilleur partage de base de données.
- Rapidité dans l'établissement des différents documents.
- Proposer une bonne codification pour les ouvrages.

7 Conclusion

Dans ce chapitre, l'organisation de la Faculté de mathématique et informatique est présentée selon sa structure administrative et académique. La bibliothèque de la faculté a été ensuite présentée avec quelques chiffres en relation avec son fond documentaire et ses ressources humaines. Cette présentation nous a permis de dresser une vision par rapport à la méthode UML à utiliser pour la conception du logiciel de gestion.

Dans le chapitre suivant nous essayant de présenté une petite description de service web et son concept, architecture, et le protocole SOAP utilisé par les services web.

CHAPITRE 2

SERVICE WEB EMBARQUÉ

1 Introduction

L'Internet des objets est la prochaine grande possibilité et un défi pour l'Internet. S'étendant de l'architecture web vers ce nouveau domaine des dispositifs et des réseaux contraints comme (smartphone, tablette) seront essentielle pour réaliser la flexibilité et l'évolutivité nécessaires pour faire un succès. Services Web sont avérés pour être indispensable dans la création de communications interopérables entre les machines sur Internet aujourd'hui, mais en même temps frais généraux excessive et la complexité de la technologie de services web tels que HTTP, XML, WSDL et SOAP sont trop élevés pour une utilisation dans les environnements contraints souvent trouvé dans des applications machine-to-machine (p. ex. l'énergie surveillance et gestion d'actifs). Dans ce chapitre nous donnons d'abord un aperçu de l'architecture web, services web SOAP et quel est leur impact sur le domaine de l'embarqué.

2 Service web et Appareils embarqué

2.1 Les Services Web

A partir de la définition de service Web par le W3C un Web service « est un logiciel conçu pour l'interopérabilité machine-to-machine interaction sur un réseau ». Son interface est décrite dans un format « machine-traitable » prévu par le protocole Web Services Description Language (WSDL). SOAP est un protocole de transport indépendant, considéré comme un protocole d'échange des messages basé sur XML et utiliser pour interagir avec d'autres systèmes. Les données de service particulier et sa structure, qui est ensuite porté dans le cadre de message SOAP (enveloppe, en-tête, corps) est déclarée par le schéma XML langage, ce dernier également incorporé dans le fichier WSDL de modélisation. Dans la littérature, les services Web qui utilisent la combinaison de WSDL et SOAP sont souvent appelés des services Web basés sur SOAP.

Une alternative à ce genre de réalisation du service Web sont axés sur les RESTful Service web. Les services Web RESTful peuvent être décrits avec WSDL ou Web Application Description Language (WADL) et les données peuvent être échangées dans un format basé sur XML.

2.2 Appareils embarqués [8]

Le concept standardisé de service Web permettrait de simplifier les scénarios de mise en œuvre du réseau embarqués comme un actionneur (par exemple l'état de l'air) et / ou une demande de nœud de l'unité de surveillance ou de souscription des données (par exemple température) d'un nœud de capteur. Surtout, si les nœuds sont hétérogènes en termes de fournisseur et de matériel. La redondance et verbosité des protocoles de services Web XML, cependant, crée toute une série de nouvelles questions de développement si elle vient à la mise en œuvre de services Web dans le domaine de l'embarqué. Les systèmes embarqués avec des ressources très limitées tels que les microcontrôleurs ARM Cortex-M3 (24 MHz CPU, 8 Ko de RAM et 128 Ko de flash) de ST Microélectronique ont rarement suffisamment de mémoire et de puissance de traitement pour exécuter un service Web, qui vient généralement, équipée avec XML générique et SOAP Parser / moteur. A côté de cette empreinte de code significative en plus de sa complexité du code et de la mémoire de la consommation relativement élevée, le traitement de la XML-verbeux et simples messages SOAP basés sur le texte dépasse les ressources disponibles dans les domaines les plus restreints et ralentirait la transmission de données dans les réseaux capteur d'acteurs. Listing 1 illustre cet aspect. Le message SOAP transporte des données de température qui provient, par exemple, d'un service de capteur. Les données réelles (échelle de température et de la valeur de la température) sont intégrées entre le début de la température et de balise de fin. Ce message simple exige déjà 274 octets. Pour traiter un tel message et de recueillir les données de température, le nombre de modèle de chaîne correspondant doit être fait : A partir de l'élément tag enveloppe, via l'étiquette de corps à la balise de l'élément de température. Réalisant cela, une représentation de chaîne (typiquement encodé en UTF8) est nécessaire dans la mémoire pour identifier les éléments particuliers qui un service ou d'un client est intérêt. En outre, étant donné que les valeurs portées des éléments sont également texte encodé, inefficace Type conversations qui doit être fait, comme chaîne à l'énumération (par exemple, de la valeur d'échelle) ou la chaîne de flotter (par exemple, 24,4).

3 Les Principales caractéristiques des services Web sont [5]

- basé sur XML.
- Le couplage lâche entre les fournisseurs de services et les demandeurs.
- Le client et la communication de service peuvent être soit synchrones ou asynchrone
- Prise en charge RPC (Remote Procedure Call) .
- Prise en charge de l'échange des documents.

4 Architecture des Services web

Jusqu'ici, l'accès via Internet à une ressource applicative ou à une base de données s'effectuait par l'envoi d'une requête s'appuyant sur des langages de script (PHP, JSP,...).

Il s'agissait donc d'un dialogue entre une couche de présentation reposant sur HTML (protocole http) et des applications installées sur un serveur distant.

Avec les Services web, un dialogue est désormais instauré entre applications qui peuvent être installées sur des machines distantes, et ceci grâce à des standards XML. En effet, afin de dialoguer via Internet, ces applications doivent « parler » le même langage, langage basé sur le XML. L'architecture de référence des services web est basée sur les trois concepts suivants :

- **Le fournisseur de service** : C'est le propriétaire du service.
- **Le client (ou le consommateur de service)** : C'est un demandeur de service. D'un point de vue technique, il est constitué par l'application qui va rechercher et invoquer un service.
- **L'annuaire des services** : C'est un registre de descriptions de services offrant des facilités de publication de services à l'intention des fournisseurs

Ainsi que des facilités de recherche de services à l'intention des clients.

Les interactions de base qui existent entre ces trois éléments sont les opérations de publication, de recherche, d'invocation et de lien (bind). Pour bien comprendre le fonctionnement de cette architecture, nous expliquons ci-dessous le rôle de chacun des éléments précédents :

La figure 2.1 décrit l'architecture d'un service web :[18]

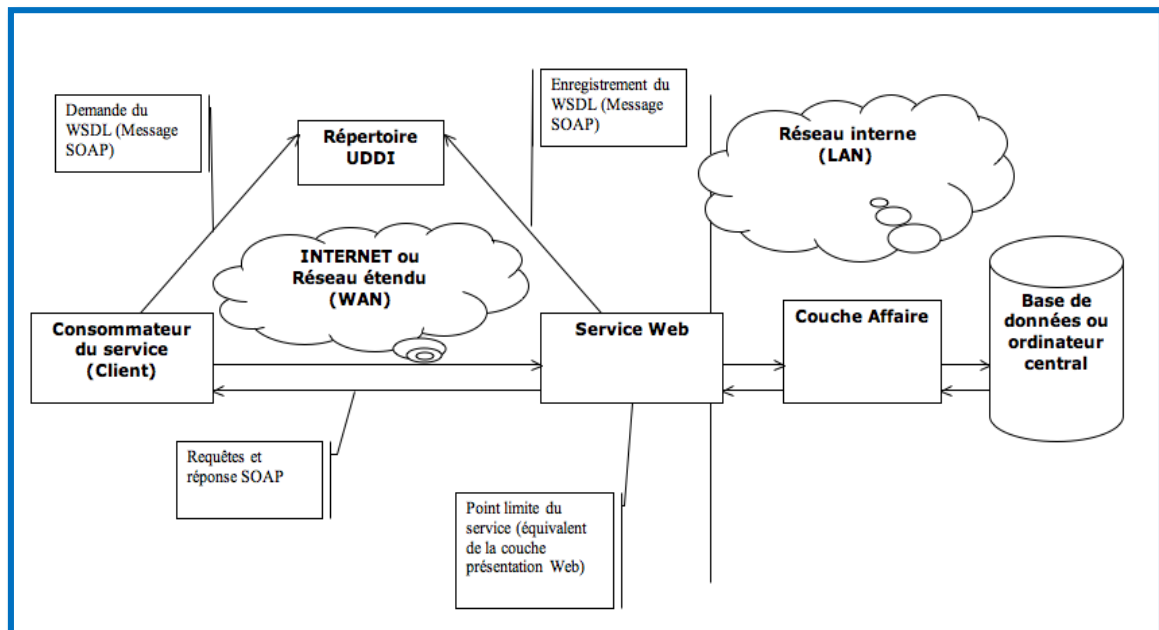


Figure 2.1 : Architecture d'un service web

5 Cycle de vie d'un service web :

L'emploi d'un service web connaît le cycle de vie suivant [9] :

- D'abord, on effectue le déploiement du service web, en fonction de la plateforme.
- Ensuite, on enregistre le service web à l'aide de WSDL (Services web Description Langage), dans l'annuaire UDDI.
- L'étape suivante est la découverte du service web par le client, par l'intermédiaire d'UDDI qui lui donne accès à tous les services web inscrits. Pour ce, on utilise SOAP.
- Enfin, Le client invoque le service web voulu, ce qui termine le cycle de vie de ce service web.

6 Les langages et les protocoles utilisés par les services web

6.1 XML – eXtensible Markup Language [6]

XML « eXtensible Markup Language » est un format texte simple et très flexible tiré du SGML et HTML.

Nous allons prendre un exemple simple pour visualiser graphiquement quelle serait la hiérarchie du document XML correspondant et quelle est la structure du document. L'exemple choisi est celui d'un livre.

```
<Livre>
  <titre>Le super livre</titre>
  <chapitre>
    <numéro>1</numéro>
    <titre>titre du chapitre 1</titre>
    <contenu>contenu de chapitre 1</contenu>
  </chapitre>
  <chapitre>
  </chapitre>
</Livre>
```

Figure 2.2 : Structure D'un document XML

XML possède une galaxie de technologies. Parmi celles-ci nous pouvons citer XML Query (langage de requêtes), XSL – eXtensible Stylesheet Language (définition de présentations de documents XML), XSLT – XSL Transformations (langage permettant la

transformation de documents XML), XPath – XML Path (langage de requêtes permettant d'accéder à chaque élément d'un document XML).

6.2 SOAP – Simple Object Access Protocol

Au sein des protocoles liés aux Services Web, SOAP fait figure de pièce centrale. En effet, SOAP définit un protocole simple destiné à l'échange d'informations.

Son objectif est de permettre la normalisation des échanges de données au sein d'architectures distribuées orientées objet [3].

SOAP constitue un standard simple et neutre puisqu'il :

- N'impose pas l'utilisation d'une API particulière.
- N'impose aucun modèle de programmation.

De plus, SOAP est fondé sur un standard : le langage XML.

6.2.1 Structure d'un message SOAP

Tout d'abord un message SOAP est un document XML qui doit avoir la forme suivante : [14]

- La structure de l'enveloppe SOAP qui définit une structure générale décrivant le contenu, le destinataire, et la nature du message.
- Les règles d'encodage SOAP qui définissent le mécanisme de sérialisation utilisé pour échanger des objets.
- SOAP RPC qui définit, pour les utilisations synchrones, une convention de représentation des appels et des réponses des procédures distantes.

La figure suivante décrit la structure d'un message SOAP : [19]

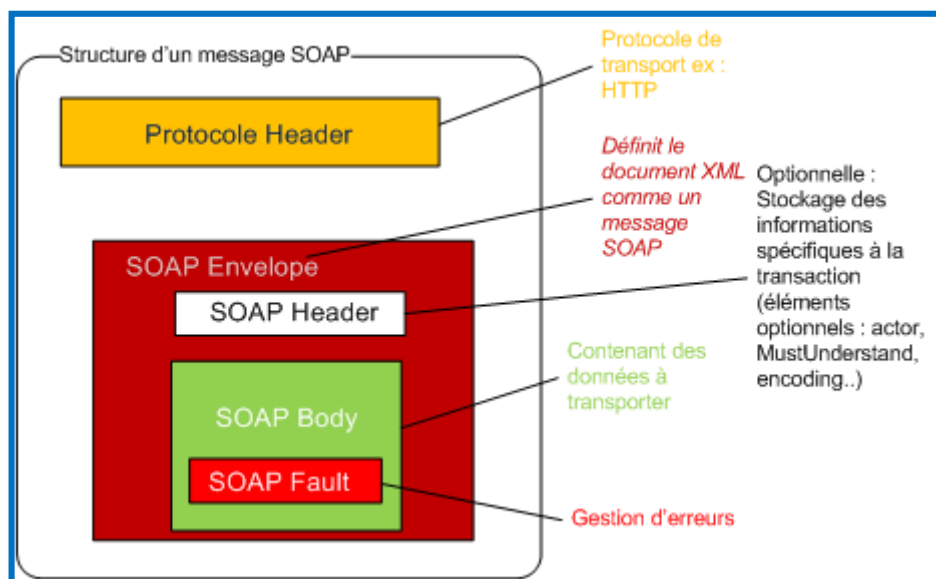


Figure 2.3 : Structure d'un message SOAP

Un message SOAP est composé des parties suivantes (figure 2.3) : [3]

a. Le HTTP Header

Le protocole HTTP envoie une requête POST. L'entête HTTP se trouve juste avant le message SOAP, et définit le destinataire du message, les règles d'encodage HTTP, etc.

Le champ **SOAPAction** peut être utilisé pour indiquer l'intention de la requête SOAP. Cette information peut être utilisée par un firewall pour filtrer les messages. Ce champ est obligatoire mais peut être vide si on n'indique pas l'intention de la requête.

b. L'enveloppe SOAP

L'enveloppe contient l'espace de nommage définissant la version de SOAP utilisée, et les règles de sérialisation, et d'encodage.

c. Le header SOAP

Cette partie du message est optionnelle. Elle sert à transmettre des informations nécessaires pour l'exécution de la requête SOAP aux dictionnaires qui recevront le message. On y précise généralement des informations liées aux transactions, à l'authentification, etc. Le header est composé d'un ou plusieurs champs : l'attribut actor, désignant le destinataire du header, l'attribut **mustUnderstand**, qui indique si le processus est optionnel.

L'attribut actor permet de préciser l'application à laquelle est destinée l'information contenue dans le header. L'URI `http://schemas.xmlsoap.org/soap/actor/next` précise en particulier que ces informations sont destinées à la première application qui reçoit le message. Dans le cas où l'actor n'est pas précisé, le header est analysé par le destinataire final du message.

Dans l'exemple suivant, on précise des informations sur l'identification de l'utilisateur et la transaction à laquelle appartient le message :

```
<SOAP-ENV: Header>
<\ "SOAP-ENV :actor="http://schemas.xmlsoap.org/soap/actor/next
<"1SOAP-ENV:mustUnderstand=">
</ "124527identifiant numero=">
</ "transaction type="compensees" numero=" YU75X>
<SOAP-ENV : Header>
```

Figure 2.4 : La structure d'un message SOAP

d. Le Body SOAP

Le body SOAP contient toutes les informations que l'on veut transmettre à l'application distante. Le contenu du Body est normalisé dans SOAP RPC, pour modéliser une requête et sa réponse. Le body de la requête contient l'identifiant de l'objet distant, le nom de la méthode à exécuter et les éventuels paramètres. Le body de la réponse contient le résultat de l'exécution de la requête.

6.2.2 SOAP RPC

SOAP RPC définit les conventions permettant d'utiliser SOAP comme un RPC, et le format des messages pour effectuer une requête ou envoyer une réponse. SOAP RPC se base sur les spécifications de XML-RPC.

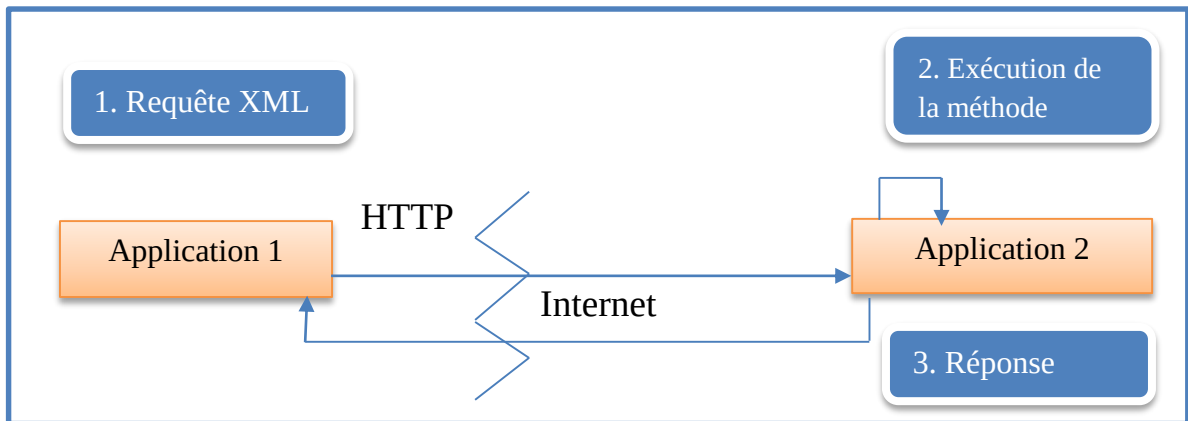


Figure 2.5 : Structure de SOAP RPC

a. Préambule

On appelle service SOAP une application dont les méthodes sont accessibles via SOAP. Ce service peut être développé dans n'importe quel langage de programmation. Un service est accessible via un identifiant unique, de type URN.

En Java toute classe peut être rendue accessible via SOAP (en utilisant Apache SOAP par exemple).

Lors de l'appel d'une méthode sur le service déployé, c'est la méthode correspondante de la classe Java qui est exécutée.

Nous donnons un exemple de classe java : « Calculatrice.java » que l'on désire rendre accessible via SOAP avec l'identifiant " urn : Calculatrice " :

```

public class Calculatrice {
    public int plus (int a, int b) {
        return a+b;
    }
}
  
```

```
}  
    public int moins (int a, int b) {  
        return a-b;  
    }  
}
```

Figure 2.6 : Exemple de méthode sur le service

b. Requêtes / réponses SOAP

Pour appeler l'opération "plus", on précise dans le Body SOAP de la requête l'identifiant de l'objet distant, l'opération à exécuter et les éventuels paramètres :

```
<SOAP-ENV:Body>  
  <m:plus xmlns:m="urn:Calculatrice">  
    <a type="xsd:int">5</a>  
    <b type="xsd:int">8</b>  
  </m:plus>  
</SOAP-ENV:Body>
```

Figure 2.7 : Structure de la Requête SOAP

Détails de la requête :

- <m:plus xmlns:m="urn:Calculatrice">: l'espace de nommage défini, a pour URN l'identifiant du service (« urn : Calculatrice »). Le nom de l'élément (plus) correspond au nom de la méthode à exécuter.
- Les paramètres d'appel de la méthode sont ensuite ajoutés les uns à la suite des autres. Ici on n'a deux paramètres a et b : de type int. On a affecté à ces deux paramètres les valeurs 5 et 8 respectivement.

Le message SOAP est alors envoyé au service SOAP. Le service exécute la méthode précisée dans la requête, ici c'est la méthode «uspl», et retourne ensuite un message SOAP dont le Body contient le résultat de l'opération :

```
<SOAP-ENV:Body>  
  <m:plus xmlns:m="urn:Calculatrice">  
    <return type="int">13</return>
```

```

    </m:plus>
  </SOAP-ENV:Body>

```

Figure 2.8 : Structure de réponse SOAP

Le résultat est contenu dans un élément `<return>` :

- L'attribut `type` précise le type de retour de la méthode exécutée.
- Le contenu de l'élément est le résultat de l'exécution de la méthode.

c. La Gestion des erreurs

En cas d'erreur lors du traitement de la requête, le serveur renvoie un message SOAP donnant les raisons de l'erreur, dans un message HTTP dont le header commence par :

- HTTP/1.1 500 Server Error

L'erreur est détaillée dans le Body SOAP, dans un élément `Fault`, donnant :

- **faultcode** : le code de l'erreur, destiné à un traitement informatique
- **faultstring** : une explication textuelle, à destination des opérateurs humains
- **faultactor (optionnel)** : en cas d'erreur dans le transport, l'acteur mis en cause peut être précisé : firewall, serveur, proxy, etc.
- **détail (optionnel)** : un détail de l'erreur (par exemple en Java la trace de l'exception renvoyée).

Par exemple, dans le cas où la signature de la méthode de la requête ne correspond pas à la signature de la méthode du service, c'est-à-dire au lieu de la méthode « plus », on met par exemple « plu », la réponse SOAP sera comme suit :

```

<SOAP-ENV:Body>
  <SOAP-ENV:Fault>
    <faultcode>
      SOAP-ENV:Client
    </faultcode>
    <faultstring>
      Method signature does not match.
    </faultstring>
  </SOAP-ENV:Fault>
</SOAP-ENV:Body>

```

Figure 2.9 : Structure de la réponse SOAP en cas d'erreur

d. Pattern d'utilisation

La norme SOAP ne définit pas les patterns d'utilisation. Ceci dit, à l'image des RPC traditionnels, on pourra être amené à utiliser un objet distant par le biais d'un proxy effectuant les requêtes SOAP, ce qui permet de manipuler l'objet distant comme un objet local :

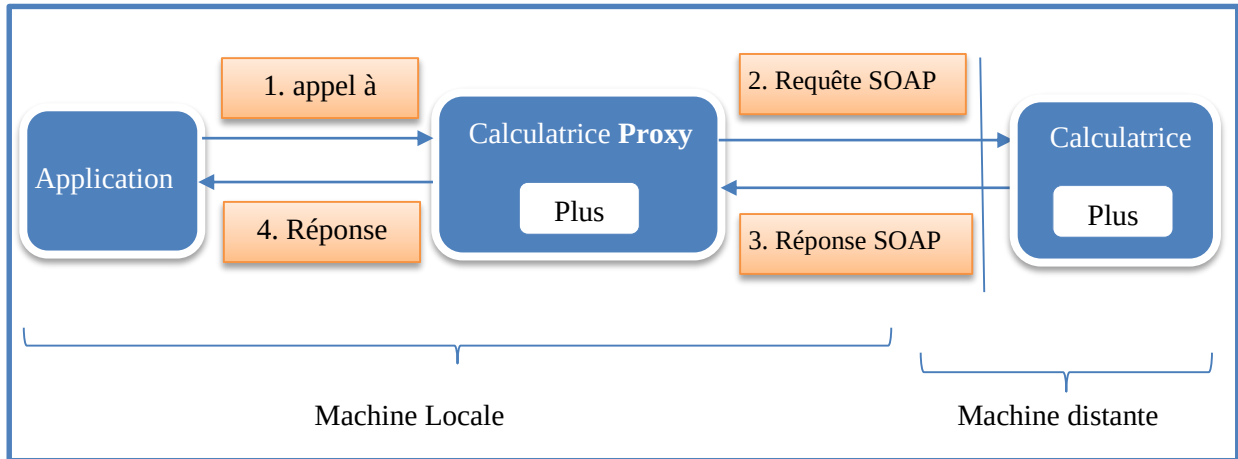


Figure 2.10 : architecture Proxy SOAP

1. Une application locale veut appeler la **méthode***plus* de l'objet distant **Calculatrice** : elle instancie un proxy et appelle sa **méthode***plus* ;
2. Le proxy est un « raccourci » vers l'objet distant : il effectue la requête SOAP destinée à l'objet distant.
3. L'objet distant decode le message XML reçu et exécute sa **méthode***plus*. Il envoie sa réponse au proxy.
4. Le proxy transmet la réponse reçue à l'application.

L'avantage de cette méthode est que l'application n'a pas à se soucier de la localisation de l'objet distant et de la construction de la requête SOAP : elle l'utilise comme un objet local.

6.3 WSDL – Service web Description Language

Le WSDL est un langage dérivé d'XML permettant de fournir les spécifications nécessaires à l'utilisation d'un Service web en décrivant les méthodes, les paramètres et les types de retour.

Le WSDL est aussi l'équivalent d'IDL (Interface Definition Language) pour la programmation distribuée (CORBA). Ce langage permet de décrire de façon précise les services Web, en incluant des détails tels que les protocoles, les serveurs, les ports utilisés, les opérations pouvant être effectuées, les formats des messages d'entrée et de sortie.

Il y eut d'autres tentatives de langages pour résoudre le problème de la définition des services Web. Microsoft a d'abord proposé SDL (Service Definition Language) avec une

implémentation fournie dans leur Toolkit SOAP, puis IBM a proposé NASSL (Network Accessible Service Specification Language), dont une implémentation est fournie dans SOAP4J. Microsoft modifia sa première spécification et proposa SCL (SOAP Contract Language), puis les différents acteurs s'accordèrent sur WSDL.

6.3.1 Structure d'un document WSDL

La figure suivante décrit la structure d'un document WSDL :

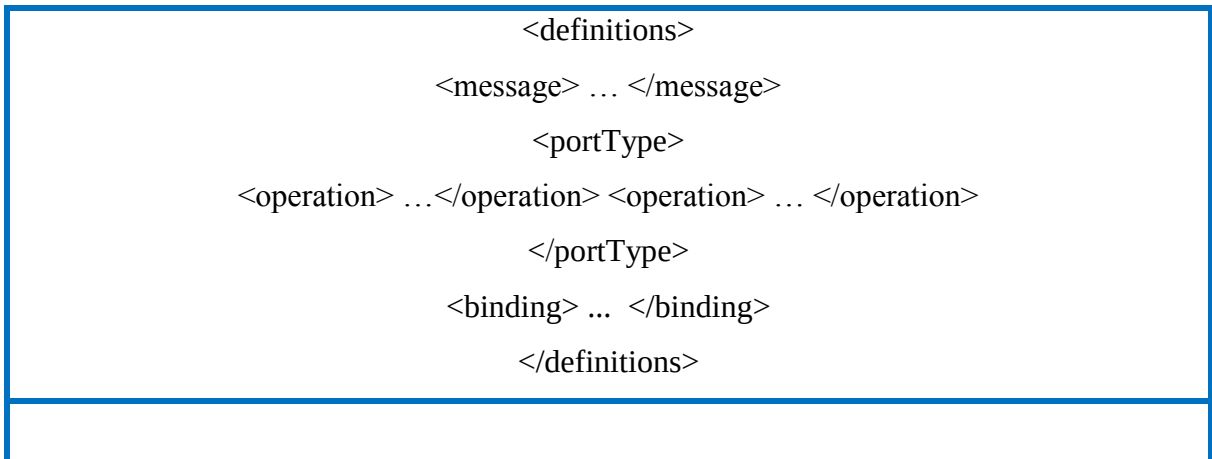


Figure 2.11 : Figure structure d'un document WSDL

- **<definitions>** : Cet élément contient la définition du service C'est. la racine de tout document WSDL. Cette balise peut contenir les attributs précisant le nom du service, et les espaces de nommage. <definitions> contient trois types d'éléments : <message>, <prototype>, <binding> et <service>.
- **<message>** et **<portType>** : Ces éléments définissent les opérations offertes par le service, leurs paramètres d'entrée et de sortie, etc. En particulier, un <message> correspond à un paramètre d'entrée ou de sortie d'une <operation>. Un <portType> définit un ensemble d'opérations. Une **<operation>** définit un couple message -entrée / message -sortie. Par exemple, dans le monde Java, une opération est une méthode et un portType une interface.
- **<binding>** : Cet élément associe les <portType> à un protocole particulier. Les bindings possibles sont SOAP, CORBA ou DCOM. Actuellement seul SOAP est utilisé. Il est possible de définir un binding pour chaque protocole supporté.
- **<service>** : Cet élément précise les informations complémentaires nécessaires

pour invoquer le service, et en particulier l'URI du destinataire. Un <service> est modélisé comme une collection de ports, un <port> étant l'association d'un <binding> à un URI.

Il est aussi possible de définir des types de données complexes dans une balise <types> juste avant la balise <message>. Enfin, chaque élément WSDL peut être documenté à l'aide de l'élément <documentation>. Cet élément contient des informations liées à la compréhension du document par les utilisateurs humains du service.

6.4 UDDI: Universal Description Discovery and Integration

L'objectif primaire d'UDDI [13] est la spécification d'un canevas pour décrire et découvrir des services Web. Le noyau d'UDDI travaille avec la notion de "business registry", qui est un service sophistiqué de noms et répertoires. Plus précisément, UDDI définit des structures de données et APIs pour publier les descriptions des services dans le registre et pour interroger ce registre afin de chercher des descriptions publiées. Parce que les APIs d'UDDI sont aussi spécifiés en WSDL avec une attache SOAP, le registre peut être invoqué comme un service Web (en conséquence, ses caractéristiques peuvent être décrites dans le même registre, comme un autre service).

Les spécifications du « registry » UDDI ont deux buts principaux en ce qui concerne la découverte d'un service : le premier, soutenir les développeurs dans la découverte d'informations sur les services. Le deuxième, permettre la liaison dynamique et en conséquence habilitier les clients pour interroger le registre et obtenir des références aux services d'intérêt.

7 Les applications des Services web

L'application des services web est multiple, autant dans les domaines du B2C, B2B que pour des domaines de gestion, par exemple gestion de stock, gestion commerciale, etc...

- **B2C (Business to Consumer)** : Qualifie une application, un site Internet destiné au grand public.
- **B2B (Business to Business)** : Qualifie une application, un site Internet destiné au commerce de professionnel à professionnel. [3]

8 Invocation d'un service web

Les étapes les plus importantes de l'invocation d'un service web sont (voir Figure 2-12) :

1. Le fournisseur (de service) se charge de l'enregistrement et de la publication des services

auprès d'un serveur UDDI. L'opération s'effectue par l'envoi d'un message (encapsulé dans une enveloppe SOAP) à l'annuaire UDDI. Ce message regroupe la localisation du service, la méthode d'invocation (et les paramètres associés) ainsi que le format de réponse. Toutes ces informations seront formalisées ensuite à l'aide de WSDL.

2. L'utilisateur qui désire consulter un service, interroge d'abord le serveur UDDI dont il connaît l'adresse afin de se renseigner sur les services disponibles correspondant à ses besoins. Le serveur lui renvoie la liste des possibilités parmi lesquelles il sélectionne l'une d'eux. A ce stade, l'utilisateur ne possède qu'une

URL⁴ identifiant le service sélectionné.

3. L'utilisateur récupère ensuite une interface WSDL, accessible depuis l'URL, lui permettant la connaissance de l'utilisation du service. De cette interface, il peut générer automatiquement le « proxy » du service. S'agissant d'un objet local il dispose des mêmes fonctions que le service distant et permettra à l'utilisateur d'accéder au service distant en toute transparence. Le "proxy" est créé grâce à un outil et peut être généré dans un grand nombre de langages de programmation différents. L'utilisation du service s'accomplit tout simplement en invoquant la méthode du "proxy" correspondant aux besoins de l'utilisateur.

4. Le proxy représente l'appel de la méthode distante sous forme d'une requête SOAP où seront inclus les paramètres fournis par l'utilisateur. Ces paramètres seront empaquetés grâce à la méthode standard de présentation des données, ce qui permet d'assurer la compatibilité entre machines. Cette requête est ensuite émise vers l'URL désignant le service web.

5. Sur la machine hébergeant le service, la requête est réceptionnée puis ouverte par un Stub.

6. Une fois la requête est comprise, une réponse SOAP est construite puis émise en direction de l'expéditeur initial.

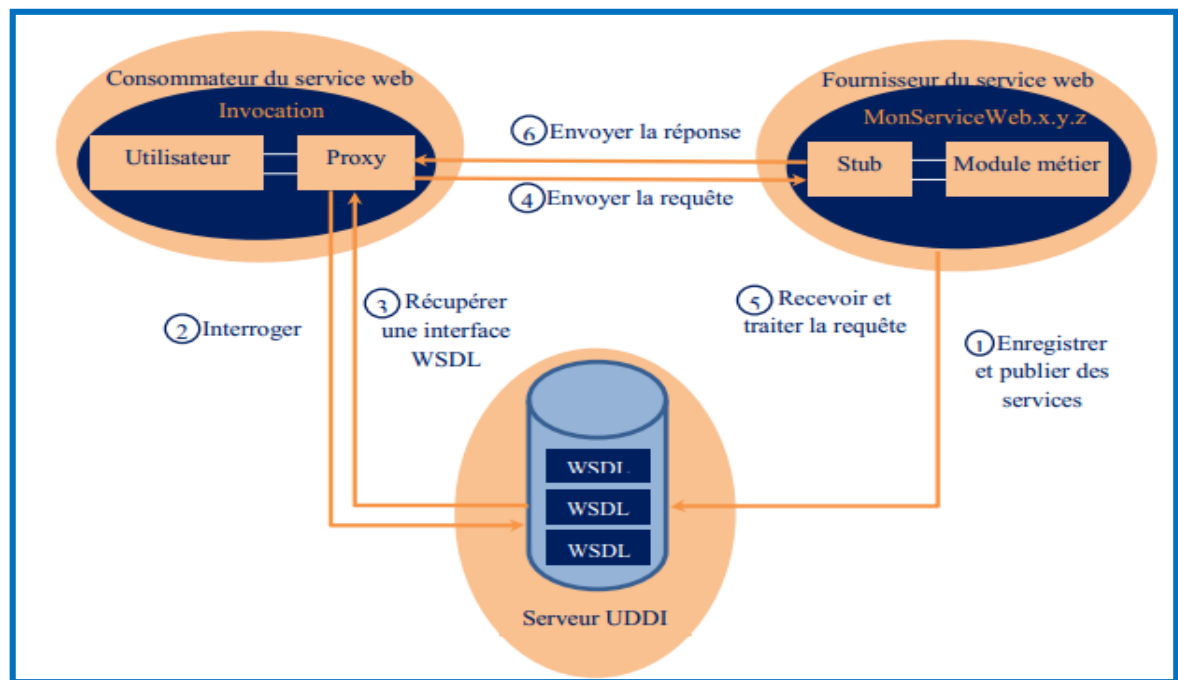


Figure 2.12 : Les étapes d'invocation d'un service web

9 Avantages et inconvénients

Les services web ont des avantages et inconvénients : [20]

9.1 Avantages

- Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- Les services Web utilisent des standards et protocoles ouverts.
- Les protocoles et les formats de données sont au format texte dans la mesure du possible, facilitant ainsi la compréhension du fonctionnement global des échanges.
- Basés sur le protocole HTTP, les services Web peuvent fonctionner au travers de nombreux pare-feu sans nécessiter des changements sur les règles de filtrage.
- Les outils de développement, s'appuyant sur ces standards, permettent la création automatique de programmes utilisant les services Web existants.

9.2 Inconvénients

- Les normes de services Web dans certains domaines sont actuellement récentes.
- Les services Web ont de faibles performances par rapport à d'autres approches de l'informatique répartie telles que le RMI, CORBA, ou DCOM.
- en l'utilisation du protocole HTTP, les services Web peuvent contourner les

mesures de sécurité mises en place à travers des pare-feu.

10 Conclusion

Un service web n'est qu'une transaction accessible par l'échange de documents XML entre deux URL .Ce dernier offre une réelle souplesse d'utilisation et permet l'accélération du développement d'application.

En adoptant les services web, on facilitera l'interopérabilité et la réutilisabilité du code, mais ceci n'est pas suffisant pour assurer l'automatisation total de l'interopérabilité .en effet on doit prendre en compte des aspects sémantiques lors de la description des services web.

Dans le chapitre suivant nous présentons l'étude conceptuelle de notre projet, et le langage de modélisation orienté objets UML et ses éléments fondamentaux.

CHAPITRE 3

CONCEPTION DU SYSTÈME

1 Introduction

Dans ce chapitre nous avons fait une Conception du système, basée sur l'utilisation du langage de modélisation UML (Unified Modeling Language).

Au préalable, nous allons évoquer UML (définition, historique, les points fort et les point faible, les diagrammes).

2 Définition d'UML

UML est le langage de modélisation de la technologie objet, standard adopté par les grands acteurs du marché. Ce document (qui doit beaucoup aux ouvrages – que je vous conseille fortement – De MERISE à UML de N. Kettani, D. Mignet, P. Paré et C. Rosenthal-Sabroux et Modélisation objet avec UML de P.-A. Muller, tous deux parus aux éditions Eyrolles en 1998 ... et à quelques recherches sur le Web) propose une présentation rapide d'UML (qui renvoie surtout à des ouvrages et à des sites Internet) et la description des modèles d'UML (illustrés de surcroît par l'exemple « jouet »). [15]

3 Historique d'UML

UML est né en octobre 1994 chez Rational Software Corporation à l'initiative de G. Booch et de J. Rumbaugh. UML 1.1 a été standardisé par l'OMG (Object Management Group) le 17 novembre 1997 suite à la demande émanant de la collaboration de plusieurs entreprises (Hewlett-Packard, IBM, i-Logix, ICON Computing, IntelliCorp, MCI Systemhouse, Microsoft, ObjecTime, Oracle, Platinum Technology, Ptech, Rational Software Corporation, Reich Technologies, Softeam, Sterling Software, Taskon et Unisys).

La version actuelle (depuis juin 1999) est UML 1.3 (la version 1.4 sera bientôt prête, afin de préparer la prochaine version 2.0). [15]

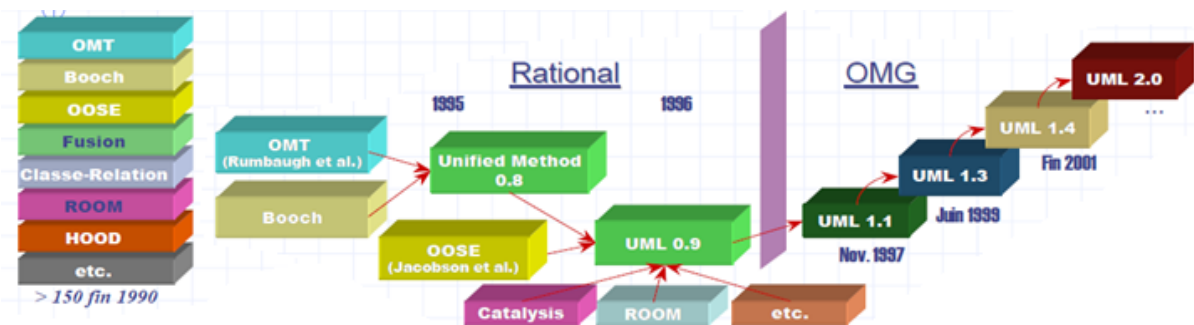


Figure 3.1 : Historique d'UML

4 Pourquoi choisir UML

- Issu (du terrain) : fruit d'un travail d'experts reconnus, résultat d'un large consensus. Langage riche : couvre toutes les phases d'un cycle de développement.
- Langage ouvert : indépendant du domaine d'application et du Langage d'implémentation.
- S'industrialise : les outils supportés se multiplient, ils permettent de respecter les normes de représentation, de gérer la cohérence de l'analyse, de générer des rapports, ...etc.
- se focaliser sur la compréhension et la résolution du problème.

5 Les points forts et faibles d'UML

5.1 Les points forts d'UML

- UML est un langage formel et normalisé :
 - gain de précision.
 - gage de stabilité.
 - encourage l'utilisation d'outils.
- UML est un support de communication performant :
 - Il cadre l'analyse.
 - Il facilite la compréhension de représentations abstraites complexes.
 - Son caractère polyvalent et sa souplesse en font un langage universel.

5.2 Les points faibles d'UML

- La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation.
- UML n'est pas à l'origine des concepts objets, mais en constitue une étape majeure, car il unifie les différentes approches et en donne une définition plus formelle.
- Le processus (non couvert par UML) est une autre clé de la réussite d'un projet.
- L'intégration d'UML dans un processus n'est pas triviale c'est une tâche complexe et Longue.

- Les auteurs d'UML sont tout à fait conscients de l'importance du processus, mais l'acceptabilité industrielle de la modélisation objet passe d'abord par la disponibilité d'un langage d'analyse objet performant et standard.

6 Dictionnaire de données

6.1 Définition

Est une collection de métadonnées ou de données de référence nécessaire à la conception d'une base de données relationnelles. Il revête une importance stratégique particulière, car il est le vocabulaire commun de l'organisation. Il décrit des données aussi importantes que les clients, nomenclateurs des produits et les services.

Un dictionnaire de données doit respecter les contraintes suivantes :

- Tous les noms doivent être monovalents et non décomposables.
- Il ne doit pas y avoir d'homonymes, ni de synonymes.
- Les données sont regroupées par entité.
- Les identifiants sont complètement précis.
- Les commentaires doivent être pertinents.

6.2 Présentations

Dans le tableau de données, chaque donnée est représentée par les critères suivants :

- Son code (informatique).
- Une signification.
- Son type (numérique, alphabétique, logique...).
- Sa longueur (en nombre de caractère).

Code	Signification	Type	Longueur
ID_Utilisateur	Id d'utilisateur	Numérique	4
Nom_Utilisateur	Nom d'utilisateur	Alphabétique	25
Mot_passe	Mot de passe d'utilisateur	Alphanumérique	25
Code_Hot	Code d'hôtesse	Numérique	4
Nom_Hot	Nom d'hôtesse	Alphabétique	25
Prenom_Hot	Prénom d'hôtesse	Alphabétique	25
Code_Adh	Code d'adhérent	Numérique	4
Nom_Adh	Nom d'adhérent	Alphabétique	25

Prenom_Adh	Prénom d'adhérent	Alphabétique	25
Date_N	Date de naissance d'adhérent	Date	
Lieu_N	Lieu de naissance d'adhérent	Alphabétique	20
Email	Email d'adhérent	Alphabétique	40
Type_Adh	Type d'adhérent	Alphabétique	20
Photo	Photo d'adhérent	image	
Groupe	Groupe d'étudiant	Alphabétique	10
Type_Ens	Type d'enseignant	Alphabétique	20
N_Res	Numéro de réservation	Numérique	4
Date_Res	Date de réservation	Date	
N_Emp	Numéro d'emprunt	Numérique	4
Date_Emp	Date d'emprunt	Date	
N_Ret	Numéro de return	Numérique	4
Date_Ret	Date de return	Date	
N_Sus	Numéro de suspendu	Numérique	4
Date_Sus	Date de suspendu	Date	
ID	Id d'ouvrage	Numérique	4
Titre	Titre d'ouvrage	Alphabétique	30
Nbr_Ex	Nombre d'exemplaire d'un ouvrage	Numérique	2
Nbr_Dis	Nombre quantité disponible d'ouvrage	Numérique	2
D_reserv	Durée de réservation d'un ouvrage	Numérique	2
Image	Image d'ouvrage	Image	
ISBN_Liv	Code ISBN d'un Livre	Alphanumérique	10
Code_Bar	Code bar d'un livre	Alphanumérique	20
Categorie	Catégorie de livre	Alphabétique	20
Description	Description de livre	Alphabétique	120
Auteur	Auteur de livre	Alphabétique	30
Edition	Année d'édition d'un livre	Numérique	4
Maison	Maison d'édition d'un livre	Alphabétique	30
ISBN_Mem	Code ISBN d'un mémoire	Alphanumérique	10
Etudiant	Etudiant qui réalisé le mémoire	Alphabétique	40
Encadreur	Encadreur qui rédigé l'étudiant	Alphabétique	40
Jure	Nom de juré de cette mémoire		100

Annee	Année universitaire de cette mémoire	Alphabétique	9
Type	Type de mémoire	Alphabétique	20
		Alphabétique	

Tableau 3.1 Dictionnaire de donnée

7 Les diagrammes UML

- Diagramme des cas d'utilisation
- Diagramme de classes
- Diagrammes d'objets
- Diagramme états-transitions
- Diagramme d'activités
- Diagramme de séquence
- Diagramme de collaboration
- Diagramme des composants
- Diagramme de déploiement

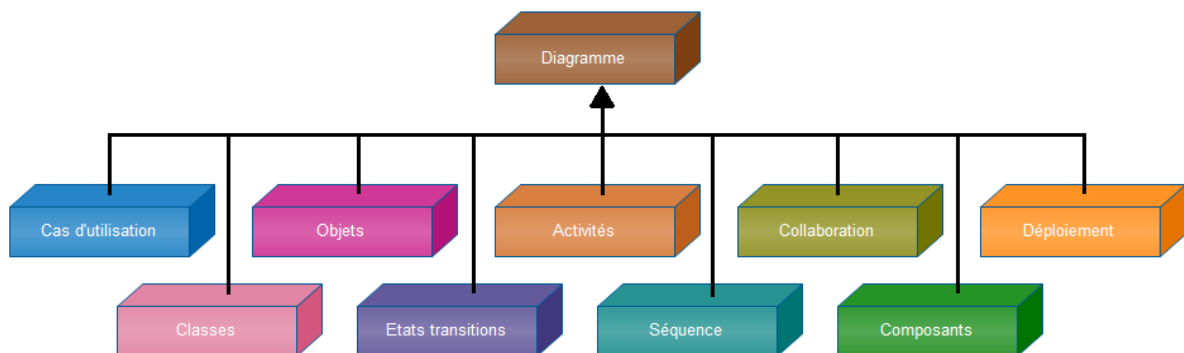


Figure 3.2 : Hiérarchie des diagrammes UML

7.1 Diagramme de cas d'utilisation

7.1.1 Définition

Les diagrammes de cas d'utilisation décrivent les services les plus importants rendus par un système.

Partant des acteurs, participants externes qui interagissent avec le système, ils représentent les cas les plus importants du système en cours d'utilisation. Un cas d'utilisation peut être divisé en diagrammes de séquence, qui détaillent les différentes fonctions du cas d'utilisation.

Les diagrammes de cas d'utilisation sont créés dans les packages, les classes, les interfaces, les acteurs, les composants et les cas d'utilisation. [1]

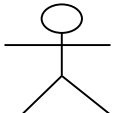
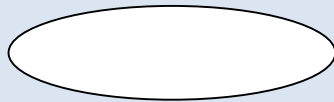
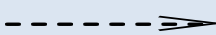
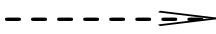
Symbole	Désignation
	Acteur
	Cas d'utilisation
	Relation entre un acteur et un cas d'utilisation
	Relation d'extension
	Relation d'inclusion

Table 3.2 : Le formalisme du diagramme de cas d'utilisation

7.1.2 Présentation

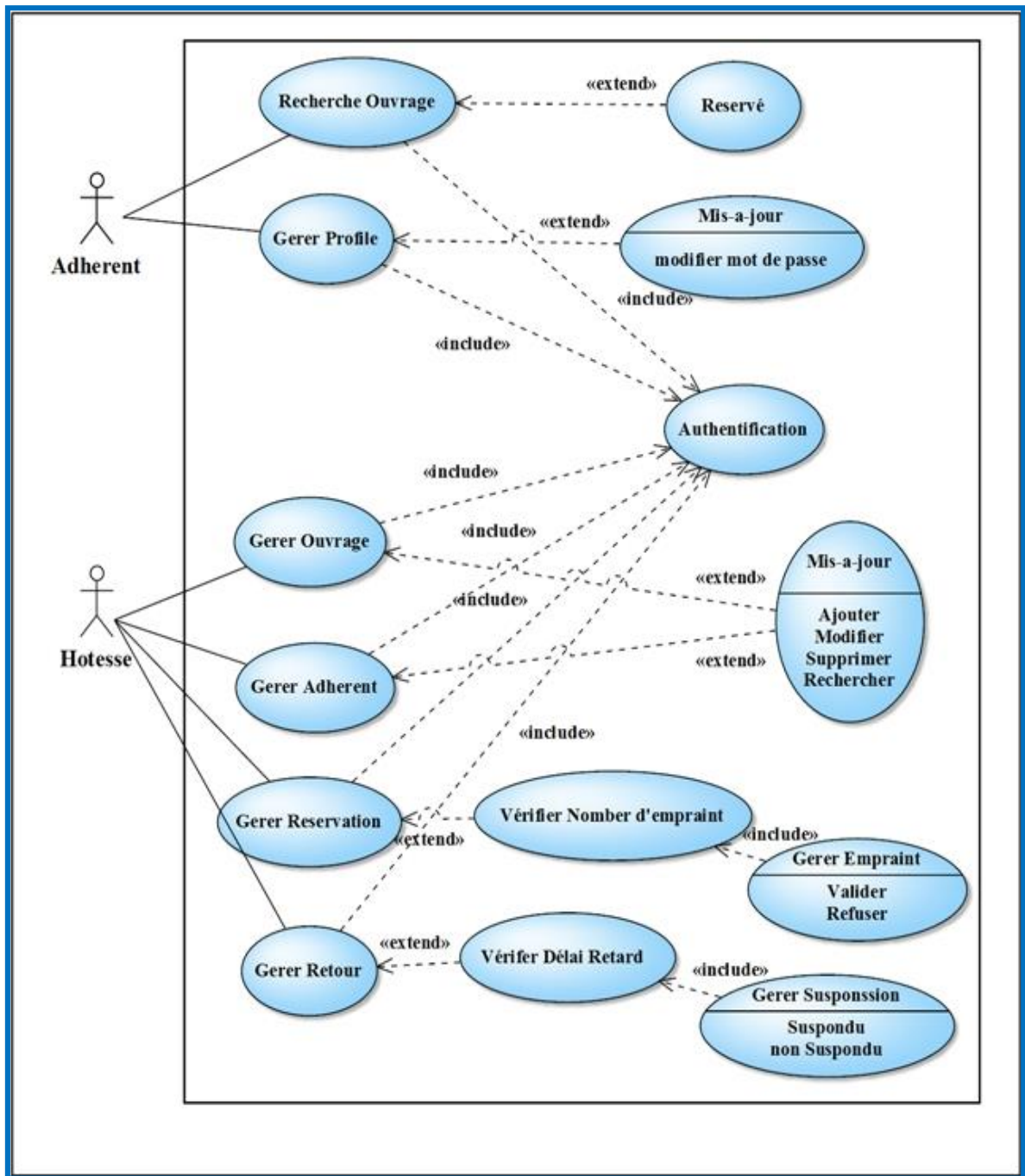


Figure 3.3 : Diagramme de cas d'utilisation

7.2 Diagramme de séquence

7.2.1 Définition

Les diagrammes de séquence présentent la coopération entre différents objets. Les objets sont définis et leur coopération est représentée par une séquence de messages entre eux.

Les objets peuvent être connectés à des classes existantes ou bien être créés indépendamment de toute classe. Si les objets sont connectés à des classes, les messages peuvent être connectés à des opérations.

Les diagrammes de séquence sont créés dans les interactions. [1]

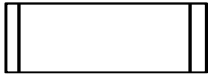
Symbole	Désignation
	Objet actif
	Activation
	Mesure
	Réécouter le message

Table 3.3 : Le formalisme du diagramme de séquence

7.2.2 Cadre d'interaction

Pour les cas plus complexes, on peut intégrer des algorithmes dans les diagrammes de séquences. Par le biais de cadres d'interaction, on peut préciser les spécificités d'un ensemble de messages :

- **alt** : fragments multiple alternatifs (si alors sinon).
- **opt** : fragment optionnel.
- **par** : fragment parallèle (traitements concurrents)

7.2.3 Diagramme de séquence « Authentification »

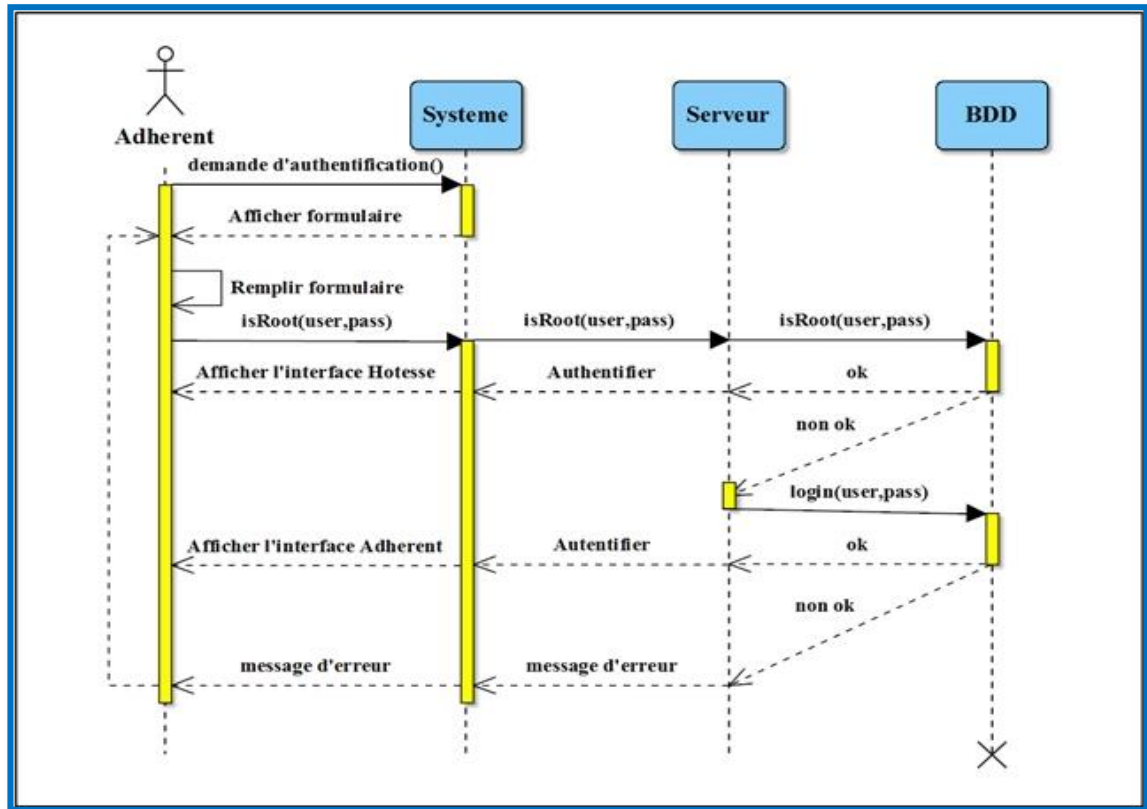


Figure 3.4 : Diagramme de séquence « Authentification »

Scénario 1 : Authentification
Objectif : Authentification
Acteurs principaux : Adhérent / Hôtesse
Préconditions : lancement de l'application
Post conditions : Affichage l'interface d'accueil (succès/authentifié) ou d'un message d'erreur (échec)
Scénario nominal
1 : l'adhérent Demande l'authentification de l'interface.
2 : l'interface présente l'interface d'authentification(Formulaire)
3 : l'adhérent remplir le formulaire
4 : l'interface appel le serveur
5 : le serveur vérifié dans BDD si les informations de l'utilisateur concernent un admin.
6.1 : le serveur appel l'interface pour afficher l'interface Hôtesse

6.2 : le serveur vérifié dans BDD si les informations de l'utilisateur concernent un Adhérent.

7 : le serveur appelle l'interface pour afficher l'interface Adhérent

Scénarios alternatifs

7a : le système affiche un message d'erreur "utilisateur inconnu".

7.2.4 Diagramme de séquence « Ajoute adhérent »

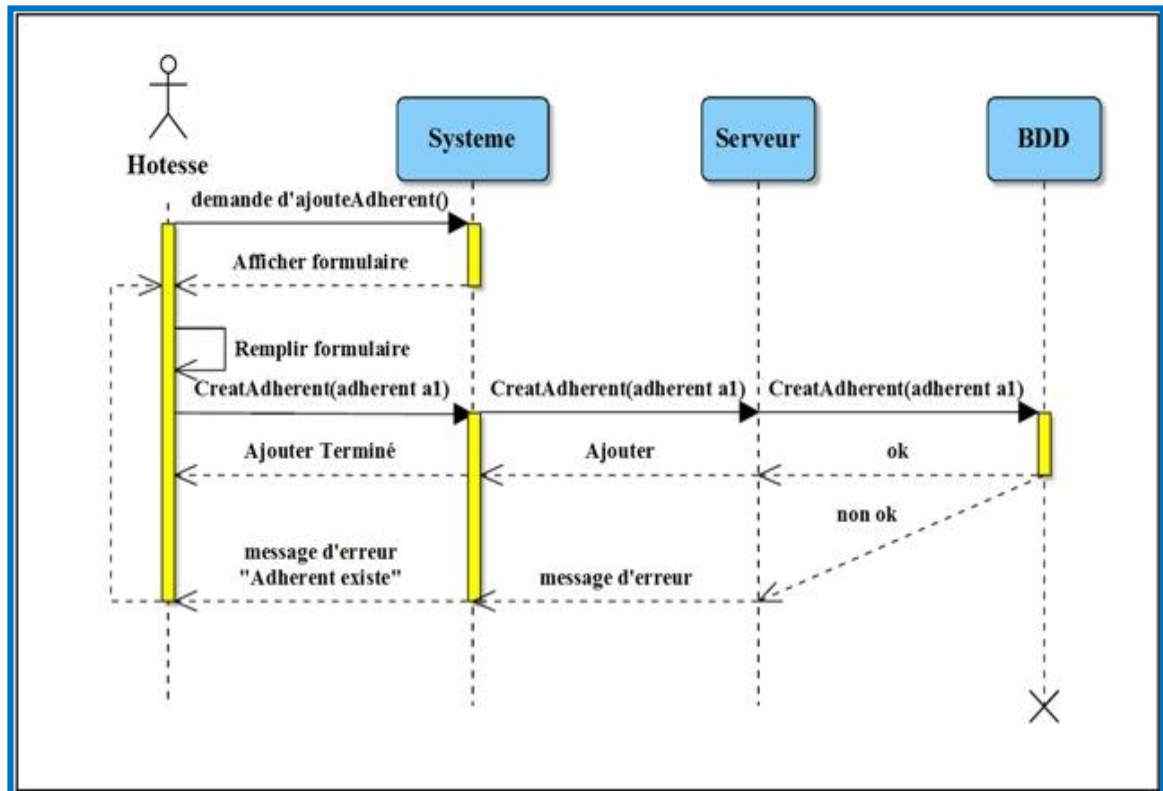


Figure 3.5 : Diagramme de séquence « Ajoute adhérent »

Scénario 1 : Authentification

Objectif : **Ajouter Adhérent.**

Acteurs principaux : **Hôtesse.**

Préconditions : **Accéder à l'Interface Hôtesse dans l'application.**

Post conditions : **Ajouter un nouveau Adhérent (succès) ou d'un message d'erreur (échec)**

Scénario nominal

1 : Après l'Authentification

2 : l'hôtesse demande l'interface ajouter Adhérent.

3 : l'interface présente l'interface d'ajouter Adhérent (Formulaire)

4 : l'hôtesse remplir le formulaire

5 : l'interface appel le serveur

6 : le serveur vérifié dans BDD.

7 : le serveur affiché dans l'interface message "adhérent ajouté"

Scénarios alternatifs

6.1 : le système affiche un message d'erreur "adhérent déjà existe".

7.2.5 Diagramme de séquence « Réservation »

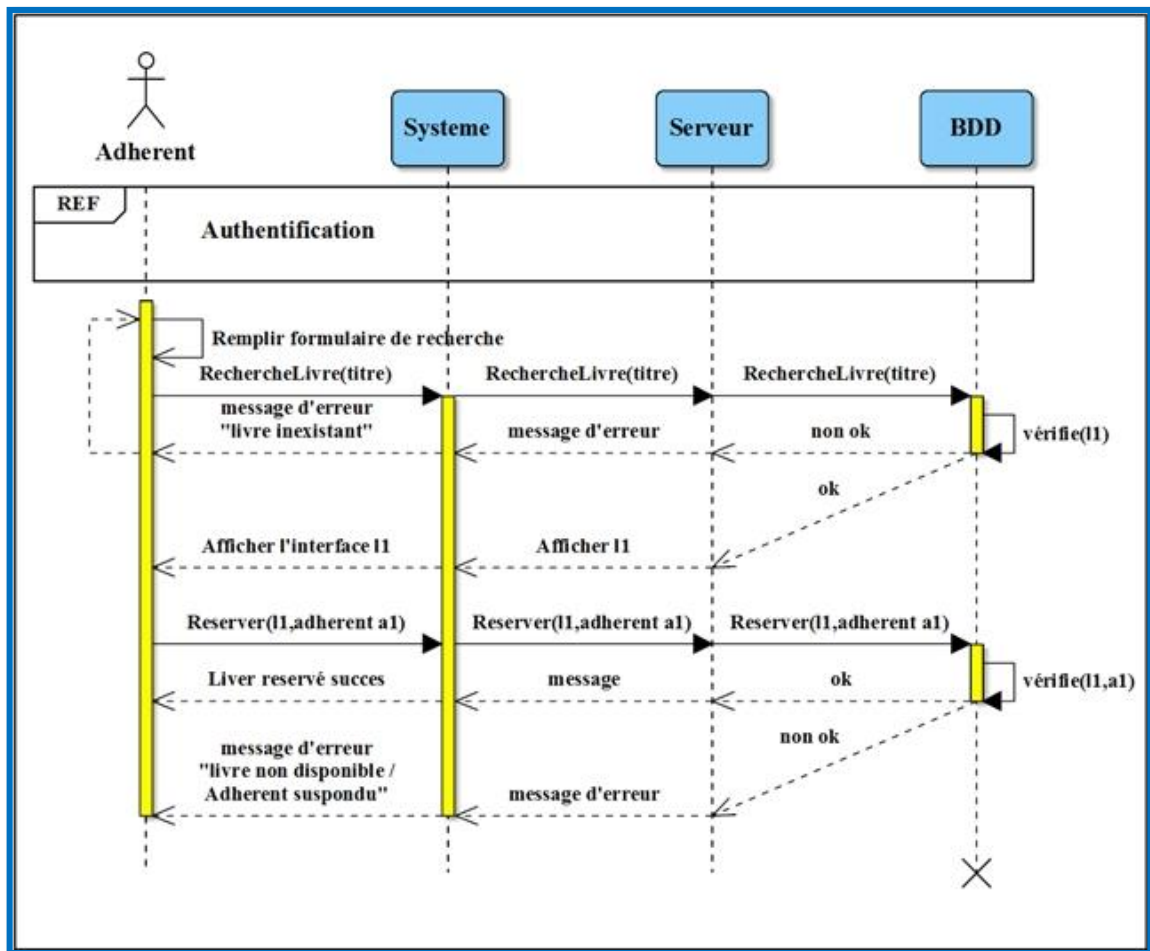


Figure 3.6 : Diagramme de séquence « Réservation »

<i>Scénario 1 : Réservation</i>
Objectif : l'adhérent Réserve un livre
Acteurs principaux : Adhérent.
Préconditions : Accéder à l'Interface Adhérent dans l'application.
Post conditions : Ajouter un nouveau Adhérent (succès) ou d'un message d'erreur (échec)
Scénario nominal 1 : Après l'Authentification 2 : l'adhérent remplir l'interface Recherche livre. 4 : l'interface appel le serveur 5 : le serveur vérifié dans BDD. 6 : le serveur affiché dans l'interface le détaille de livre. 7 : l'adhérent demande réserver livre. 8 : le serveur vérifié dans la BDD et mise à jour. 9 : le serveur afficher message "livre réserver". Scénarios alternatifs 5.1 : le serveur affiche un message d'erreur "livre n'existe pas". 8.1 : le serveur affiche un message d'erreur "livre n'est pas disponible ou adhérent suspendu"

7.2.6 Diagramme de séquence « Emprunt »

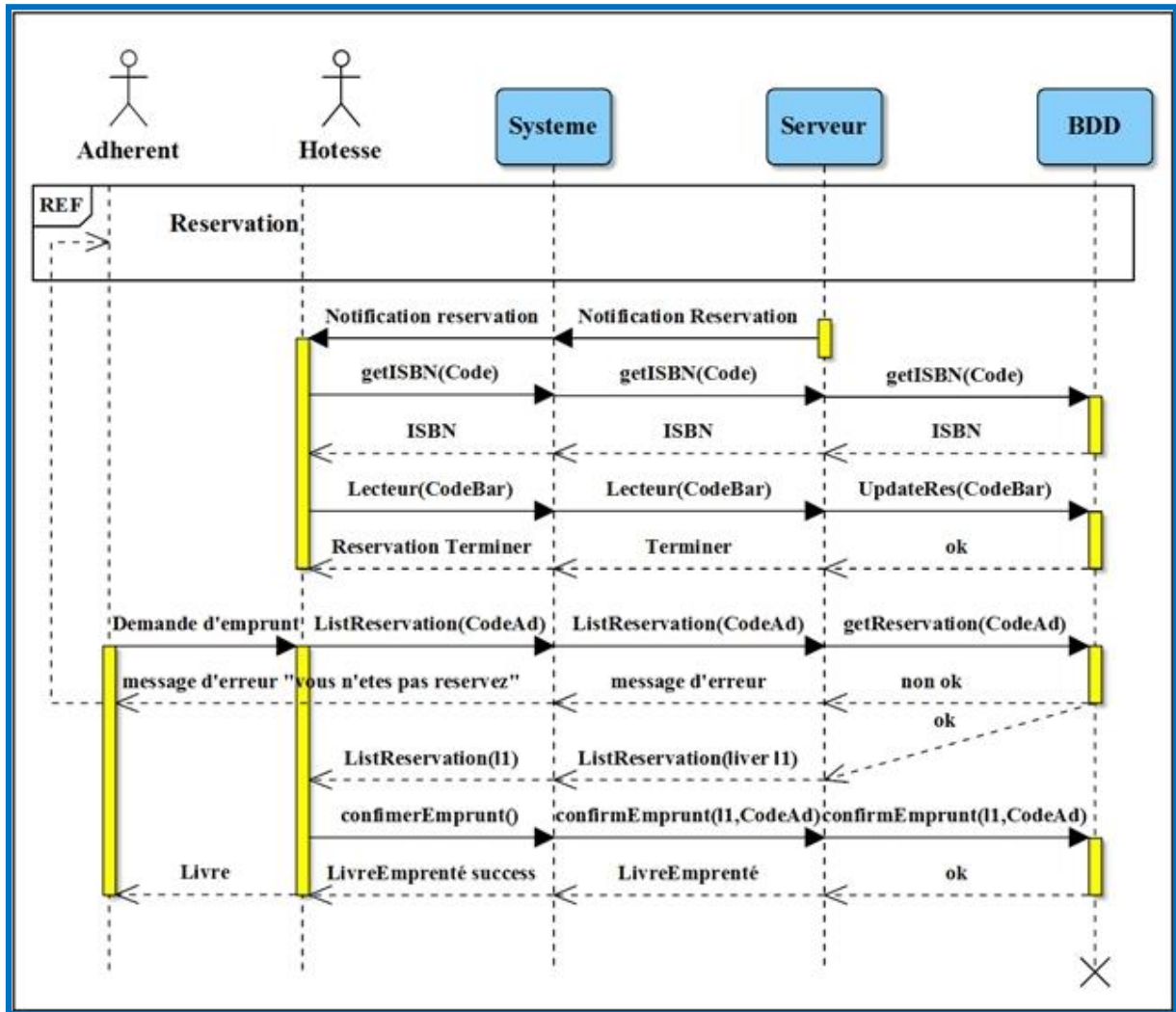


Figure 3.7 : Diagramme de séquence « Emprunt »

Scénario 1 : Emprunt
Objectif : l'adhérent emprunt un livre
Acteurs principaux : Adhérent, Hôtesse.
Préconditions : Accéder à l'Interface Hôtesse dans l'application.
Post conditions : emprunté un livre (succès) ou d'un message d'erreur (échec)
Scénario nominal
1 : Après Réservation.
2 : l'hôtesse reçoit demande d'emprunt.
3 : l'hôtesse demande recherche réservation.
4 : le serveur affiche l'interface recherche réservation.

5 : l'interface appelle le serveur

6 : le serveur vérifié dans BDD.

6 : le serveur affiché dans l'interface le détail de réservation.

7 : l'hôtesse confirme emprunt livre.

8 : le serveur à jour la BDD.

9 : le serveur affiche message "Emprunt Terminé".

Scénarios alternatifs

6.1 : le serveur affiche un message d'erreur "livre n'est pas réservé ou délai dépassé".

7.3 Diagramme de classes

7.3.1 Définition

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation.

Alors que le diagramme de cas d'utilisation montre un système du point de vue des acteurs, le diagramme de classes en montre la structure interne. Il permet de fournir une représentation abstraite des objets du système qui vont interagir pour réaliser les cas d'utilisation. Il est important de noter qu'un même objet peut très bien intervenir dans la réalisation de plusieurs cas d'utilisation. Les cas d'utilisation ne réalisent donc pas une partition des classes du diagramme de classes. Un diagramme de classes n'est donc pas adapté (sauf cas particulier) pour détailler, décomposer, ou illustrer la réalisation d'un cas d'utilisation particulier. [1]

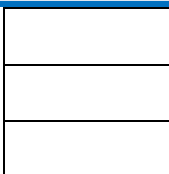
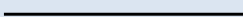
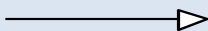
Symbole	Désignation
	Classe
	Association
	Agrégation
	Généralisation
	Composition

Table 3.4 : Le formalisme du diagramme de classe

7.3.2 Présentation

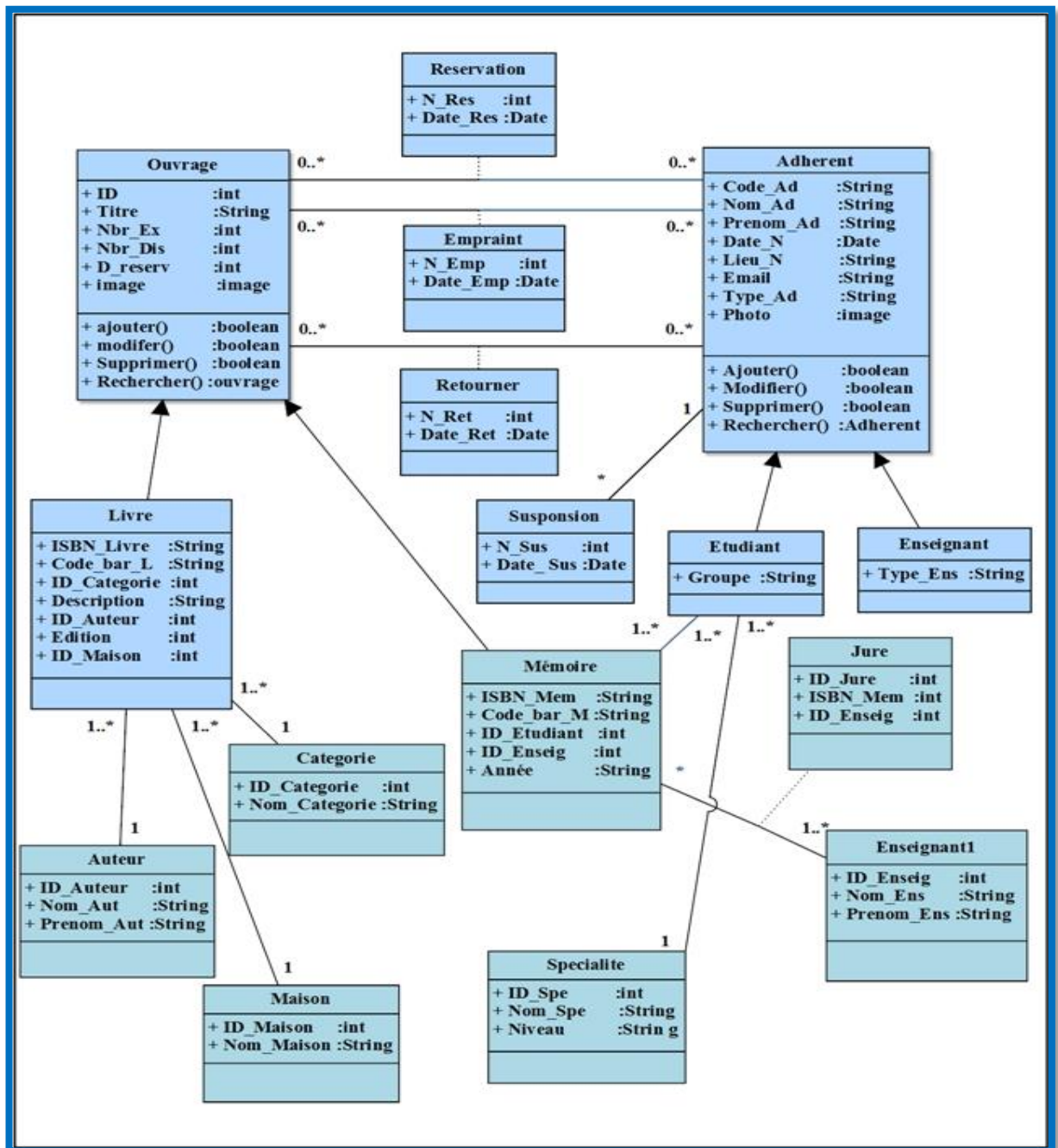


Figure 3.8 : Diagramme de classes d'application

8 Schéma relationnel

Adhérent (Code_Adh, Nom_Adh, Prenom_Adh, Date_N, Lieu_N, Email, Type_Ad, photo)

Etudiant (Code_Adh*, Groupe)

Enseignant (Code_Adh*, Type_Ens)

Ouvrage (ID, Titre, Nbr_Ex, Nbr_Dis, D_reserv, image)

Categorie (ID_Categorie, Nom_Categorie)

Auteur (ID_Auteur, Nom_Auteur, Prenom_Auteur)

Maison (ID_Maison, Nom_Maison)

Livre (ID*, ISBN Livre, Code bar L, ID_Categorie*, Description, ID_Auteur*, Edition, ID_Maison*)

Enseignant1 (ID_Enseig, Nom_Ens, Prenom_Ens)

Specialite (ID_Spe, Nom_Spe, Niveau)

Mémoire (ID*, ISBN Mem, Code bar M, ID_Etudiant*, ID_Enseig*, Annee)

Jure (ID_Jure, ISBN Mem*, ID_Enseig*)

Réservation (N_Res, Code Adh*, ID*, Date_Res)

Emprunt (N_Emp, Code Adh*, ID*, Date_Emp)

Retour (N_Ret, Code Adh*, ID*, Date_Ret)

Suspension (N_Sus, Code Adh*, Date_Sus)

9 Conclusion

Dans ce chapitre, nous avons présenté l'étude conceptuelle de notre système en élisant le langage de modélisation UML. Nous avons utilisé les principaux diagrammes tels que : diagramme de cas d'utilisation, diagrammes de séquence, et le diagramme de classe, nous avons procédé aussi au modèle logique de données (MLD) qui nous a permis d'exploiter la base de données relationnel.

Dans le chapitre suivant nous essayant de présenté les outils de développement qui nous aidons à la réalisation de notre application.

CHAPITRE 4

LA REALISATION

1 Introduction

Dans ce chapitre nous présenterons cette étape de réalisation Nous décrirons aussi les outils utilisé et les technologies exploité.

Nous exposons aussi les interfaces les plus importants de notre application (service web).

2 Les outils utilisés

2.1 L'environnement de travail

2.1.1 Eclipse

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.[11]



Figure 4.1 : Eclipse

2.1.2 Android Studio

Est un Android axé IDE. Android Studio contient tous les outils Android SDK pour concevoir, tester, déboguer et profiler toute les applications Android. Il est similaire à Eclipse avec le plug-in ADT, mais il y a de nombreuses fonctionnalités disponibles dans Android Studio qui peuvent augmenter la productivité du développement telles que :

- puissante édition de code
- Sous la direction de mise en page Riche
- Support de construction à base de Gradle
- assistants basés sur des modèles
- l'analyse de l'outil de Lint (il est une analyse statique de code qui vérifie les fichiers source de projet Android pour les bugs potentiels et des améliorations d'optimisation pour l'exactitude, la sécurité, la performance, la convivialité, l'accessibilité et l'internationalisation).

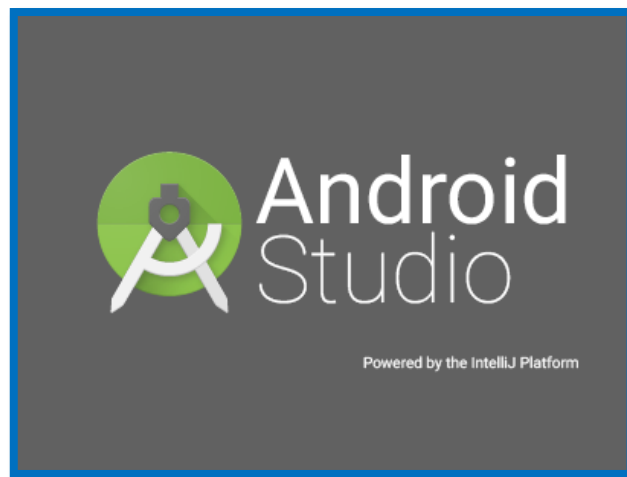


Figure 4.2 : Android Studio

2.1.3 Gradle

Gradle est un outil gratuit de génération de Java pour l'automatisation de projet, similaire à Ant et Maven. Il est l'utilisateur par les développeurs de logiciels de travail en Java pour la compilation du code, résoudre les dépendances, l'emballage, les tests unitaires, l'exécution de code et de déploiement. Il est un outil de construction d'usage général très flexible, commutable, construire par convention Cadres à la maven, le soutien très puissant pour le multi-projet construit de gestion des dépendances, très puissant.

2.1.4 Software de développement kit (SDK)

C'est un kit de développement proposé par Google aux développeurs souhaitant programmer des applications pour terminaux mobiles tournant sous Android (smartphones, tablettes,...)[11]

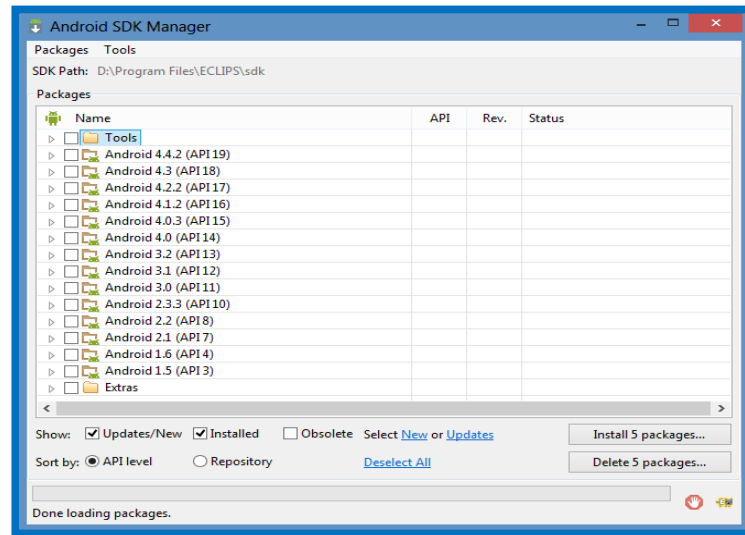


Figure 4.3 : Android SDK Manager

2.1.5 Emulateur

Le SDK propose un émulateur Android. Il permet de lancer sur la machine du développeur un terminal virtuel représentant à l'écran un téléphone embarquant Android. C'est bien évidemment un outil indispensable pour le développement mobile. A chaque version d'Android est associée une version de l'émulateur, permettant au développeur de voir exactement à quoi ressemblera son application sur un matériel réel. Nous rappelons cependant que l'émulateur ne propose pas toutes les fonctionnalités d'un vrai téléphone.[11]



Figure 4.4 : Emulateur

2.2 Serveur

2.2.1 WAMP Server

WampServer est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans se connecter à un serveur externe) des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi qu'une administration pour les deux bases SQL PhpMyAdmin et SQLiteManager.



Figure 4.5 : WAMP server

2.2.2 Apache Tomcat

Apache Tomcat est une implémentation open source d'un conteneur web qui permet donc d'exécuter des applications web reposant sur les technologies servlets et JSP.

Tomcat est diffusé en open source sous une licence Apache. C'est aussi l'implémentation de référence des spécifications servlets jusqu'à la version 2.4 et JSP jusqu'à la version 2.0 implémentées dans les différentes versions de Tomcat.[16]

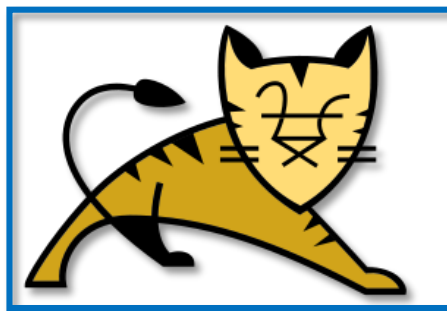


Figure 4.6 : Apache Tomcat

2.2.3 Axis 2

Axis 2 est un moteur de services web, il vise à faciliter le développement de services web en technologie SOAP.

Il a pour objectif d'être plus efficace, plus modulaire et plus orienté XML que la version précédente (Axis).



Figure 4.7 : Axis 2

2.3 Technologie

2.3.1 MySQL

MySQL est un serveur de bases de données relationnelles Open Source. Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations définies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête. Le SQL dans "MySQL" signifie "Structured Query Language" : le langage standard pour les traitements de bases de données.[7]



Figure 4.8 : MySQL

2.3.2 PhpMyAdmin

Est une interface conviviale qui permet de gérer très facilement une base de données, sans nécessiter une connaissance avancée de requêtes SQL.

Avec le gestionnaire de base de données PHPMyAdmin, on pourra rapidement créer et supprimer des base de donnée, assurer mise à jour (ajouter, modifier et supprimer...) des tables et permet d'exécuter des requête SQL.[2]



Figure 4.9 : phpMyAdmin

2.3.3 Java

Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems.

Le Java introduit une couche intermédiaire supplémentaire entre le code et le langage machine, ce qu'on appelle la Java Virtual Machine (JVM), qui doit être installée sur tout ordinateur voulant exécuter un programme écrit en Java, et qui est spécifique à cette configuration matérielle. Il s'agit donc d'un langage compilé dont le code machine sera interprété, traduit au travers de la JVM dans le langage de la machine hôte. La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation.



Figure 4.10 : JAVA

2.3.4 Java EE

De nombreuses possibilités existent pour réaliser des applications Internet depuis plusieurs années. Des langages ont été créés des architectures et des environnements de travail ont été conçus pour répondre aux besoins et faciliter la tâche des développeurs. Sun (le concepteur de Java) a donc mis en place un ensemble de technologies pour réaliser des applications Web. Ces technologies sont regroupées sous le nom J2EE (Java 2 Entreprise Edition), désormais Java EE.

Depuis la version 5, le chiffre 2 a disparu pour faciliter la compréhension de la version et ne pas mélanger le chiffre 2 avec le numéro de version.

La plate-forme Java EE s'appuie entièrement sur le langage Java. Java EE est donc une norme, qui permet à des développeurs, entreprises et SSII de développer leur propre application qui implémente en totalité ou partiellement les spécifications de SUN. En simplifiant, il est possible de représenter Java EE comme un ensemble de spécifications d'API, une architecture, une méthode de packaging et de déploiement d'applications et la gestion d'applications déployées sur un serveur compatible Java.

La plate-forme J2SE est composée des éléments suivants :

- **La machine virtuelle Java (JVM)** : permet d'exécuter des applications Java. Elle constitue une passerelle et permet une portabilité entre les architectures (Windows, Linux, Mac...)
- **La bibliothèque de classes Java** : un ensemble de composants logiciels prêt à l'emploi.
- **Les outils de développement** : le compilateur java, un interpréteur Java nommé java, le générateur de documentation javadoc, la console de supervision Jconsole... La plate-forme Java EE est une extension de la plate-forme J2SE. Elle permet un développement d'applications qui vont s'exécuter sur un serveur d'applications.

Les applications seront utilisées par des clients légers (comme des navigateurs Web) ou bien des applications lourdes (IHM). La dernière version stable de Java EE est la version Java EE 5.0 et fonctionne avec le JDK 5.0 et 6.0



Figure 4.11 : Java EE

2.3.5 Html

Le langage HTML (pour Hyper Text Markup Language) permet de créer des documents indépendant de toute plate-forme, et donc particulièrement bien adapte à des échanges d'informations dans un environnement hétérogène comme le web. HTML repose sur quelques concepts très différents de ceux que l'on peut trouver dans un traitement de texte standard. Un système de balises (d'où le terme Markup Language) permet d'indiquer explicitement, pour chaque partie du texte, quelle est sa fonction (titre, en-tête de section, légende de figure, etc.) ou son mode de présentation.[7]



Figure 4.12 : HTML

2.3.6 CSS

C'est un système destiné à mettre en forme les contenus de pages web. La partie CSS d'un document web se contente de définir les différents styles de textes ou de blocs qui seront utilisés pour la mise en forme tandis que la partie HTML ne contient que le texte encadré de quelques balises. Chaque style CSS se voit attribuer un nom. Pour attribuer l'un de ces styles à un segment de texte particulier.[17]



Figure 4.13 : CSS

2.3.7 JSP

Une JSP est un fichier contenant du code HTML et des fragments de code Java exécutés sur le moteur de Servlets Comparable aux langages côté serveur de type PHP, ASP, ... Les pages JSP sont converties en Servlet par le moteur de Servlets lors du premier appel à la JSP [17]

2.3.8 Servlet

Les servlets (on dit généralement une servlet) sont au serveur Web ce que les applets sont au navigateur pour le client. Les servlets sont donc des applications Java fonctionnant du côté serveur au même titre que les CGI et les langages de script côté serveur tels que ASP ou bien PHP. Les servlets permettent donc de gérer des requêtes HTTP et de fournir au client une réponse HTTP dynamique (donc de créer des pages web dynamiques).



Figure 4.14 : Servlet

2.3.9 Java Scripte

JavaScript est un langage de programmation de scripts, principalement utilisé dans les pages web interactives. C'est un langage orienté objet à prototype, c'est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des instances de classes, mais qui sont équipés de constructeurs permettant de générer leurs propriétés. Le langage a été créé en 1995 par Brendan Eich pour le compte de Netscape Communications Corporation. [13]



Figure 4.15 : Java Scripte

2.3.10 JQuery

JQuery est un Framework JavaScript sous licence libre qui permet de faciliter des fonctionnalités communes de JavaScript.

L'utilisation de cette bibliothèque permet de gagner du temps de développement lors de l'interaction sur le code HTML d'une page web, l'AJAX ou la gestion des événements. JQuery possède par la même occasion l'avantage d'être utilisable sur plusieurs navigateurs web (cf. Internet Explorer, Firefox, Chrome,...etc.). [17]



Figure 4.16 : JQuery

2.3.11 Ajax

Ajax est un acronyme pour Asynchrones JavaScript and XML « XML et JavaScript asynchrones » et désignant une solution informatique libre pour le développement de pages dynamiques et d'applications Web.

Le principal attrait d'Ajax réside dans le rendu dynamique des pages et une interactivité qui n'est pas sans rappeler le mode d'interaction du client lourd. L'avènement des interfaces riches (Riche Internet Application) permet au Web de fournir des sites associant une interactivité plus riche, jusque-là réservée au client lourd, tout en gardant une facilité de consultation qui leur sont propres.

Pour expliquer le concept d'AJAX en quelques mots, il s'agit de permettre l'échange d'informations entre le navigateur web (sur le poste de l'utilisateur) et le serveur web (chez l'hébergeur) sans recharger l'ensemble de la page web utilisée. Ces échanges s'effectuent de façon asynchrone (le mode synchrone est aussi possible) à l'aide de fonctions développées en langage JavaScript. [17]



Figure 4.17 : Ajax

3 Les Interfaces

Nous présentons dans cette section quelque interface principale de notre réalisation qui illustre les différent cas d'utilisation déjà vus dans le chapitre Conception du système.

3.1 Interface d'Authentification

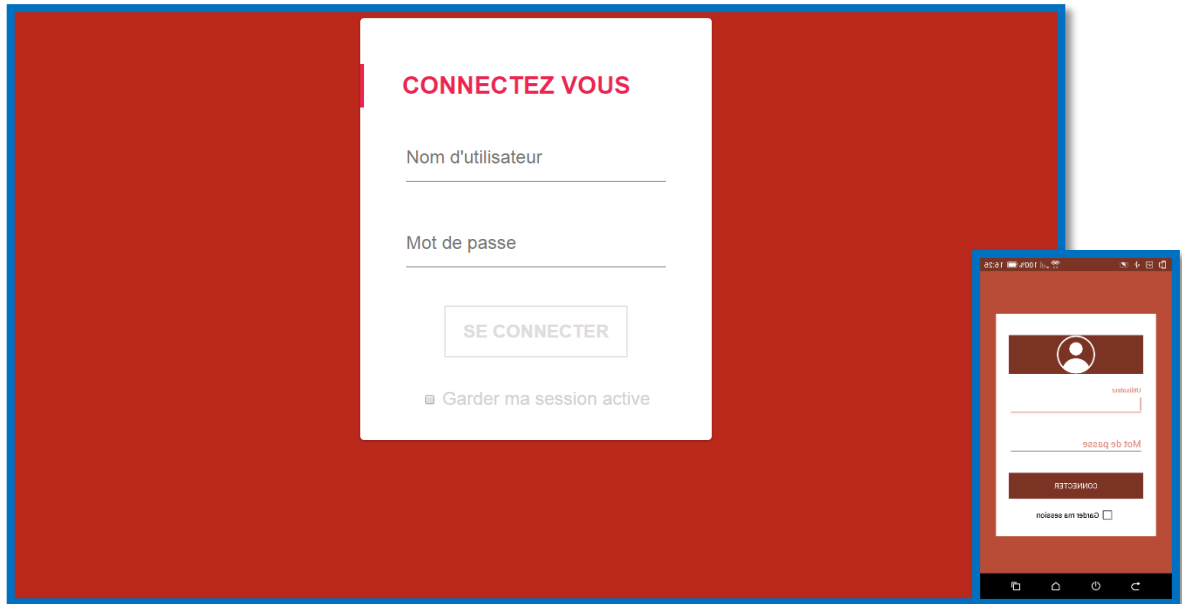


Figure 4.18 : Interface d'Authentification

3.2 Interface principale Hôtesse

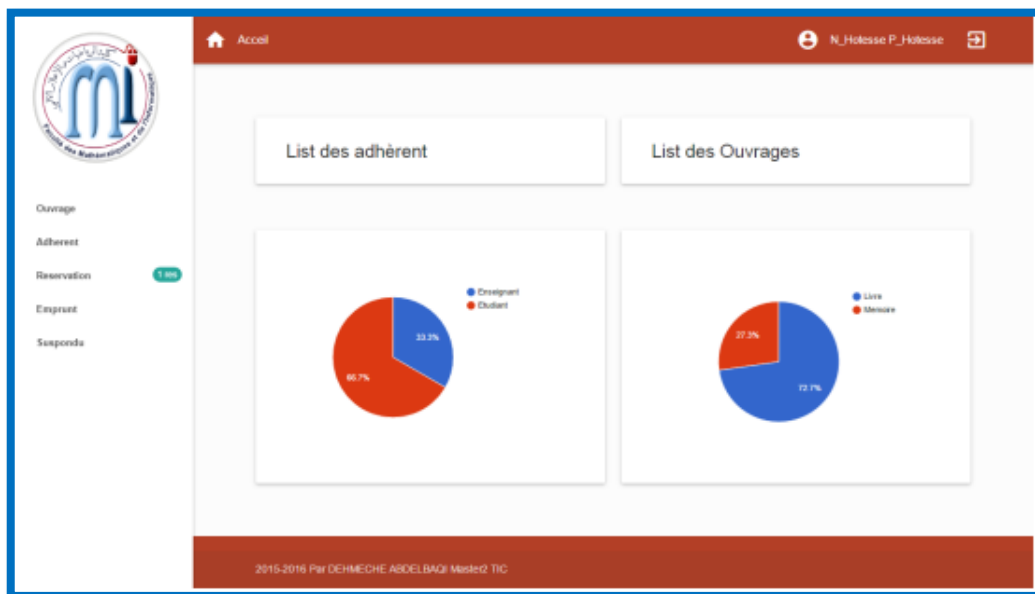


Figure 4.19 : Interface principale Hôtesse

3.3 Interface principale d'Adhèrent

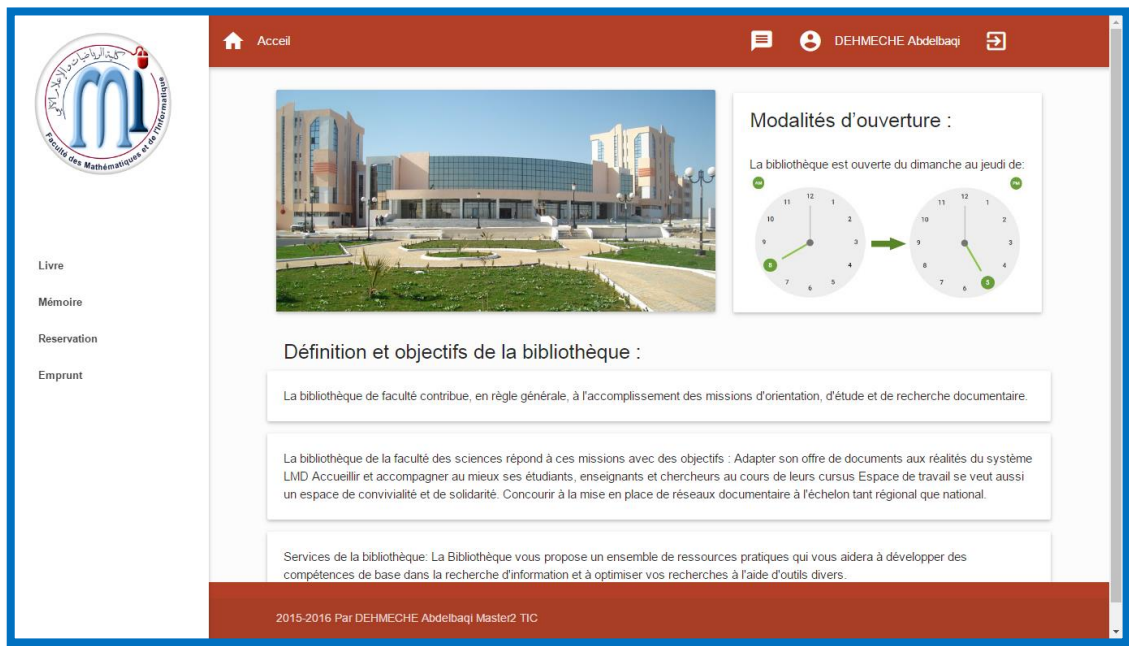


Figure 4.20 : Interface principale d'Adhèrent

3.4 Interface de liste livre

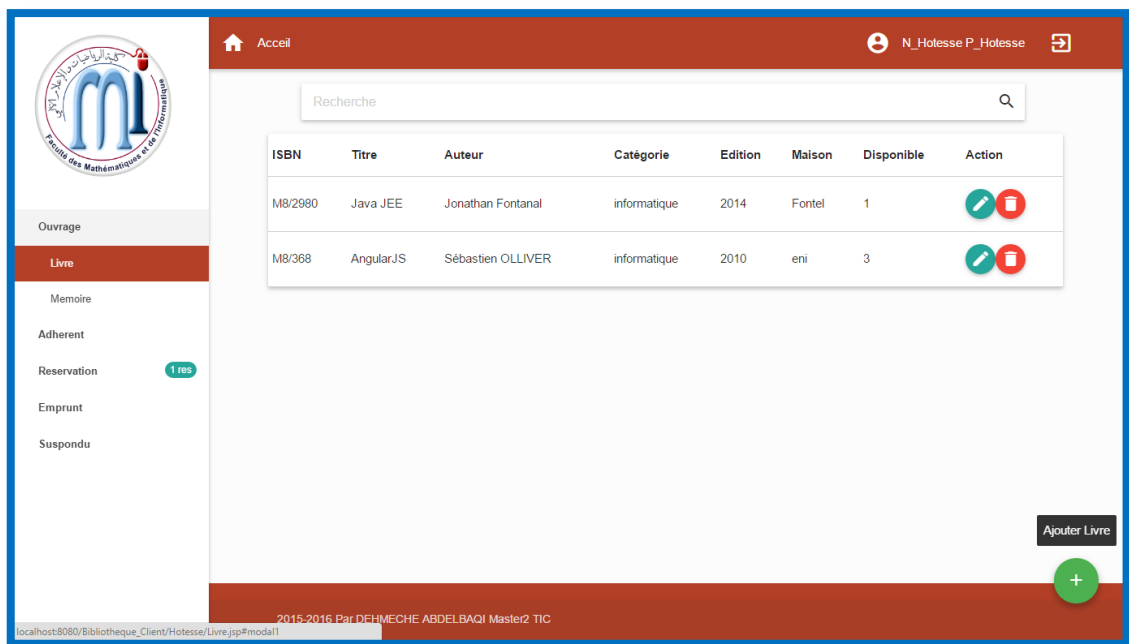


Figure 4.21 : Interface de liste livre

3.5 Interface d'ajoute livre

Figure 4.22 : Interface d'ajoute livre

3.6 Interface Recherche livre

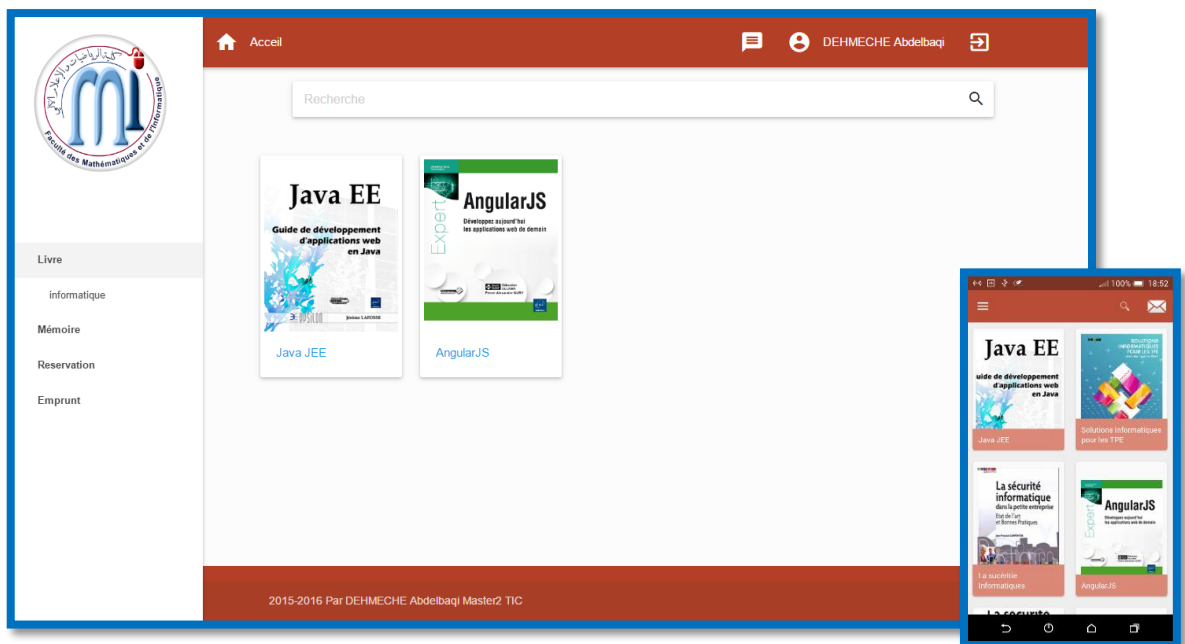


Figure 4.23 : Interface Recherche livre

3.7 Interface Réserve livre

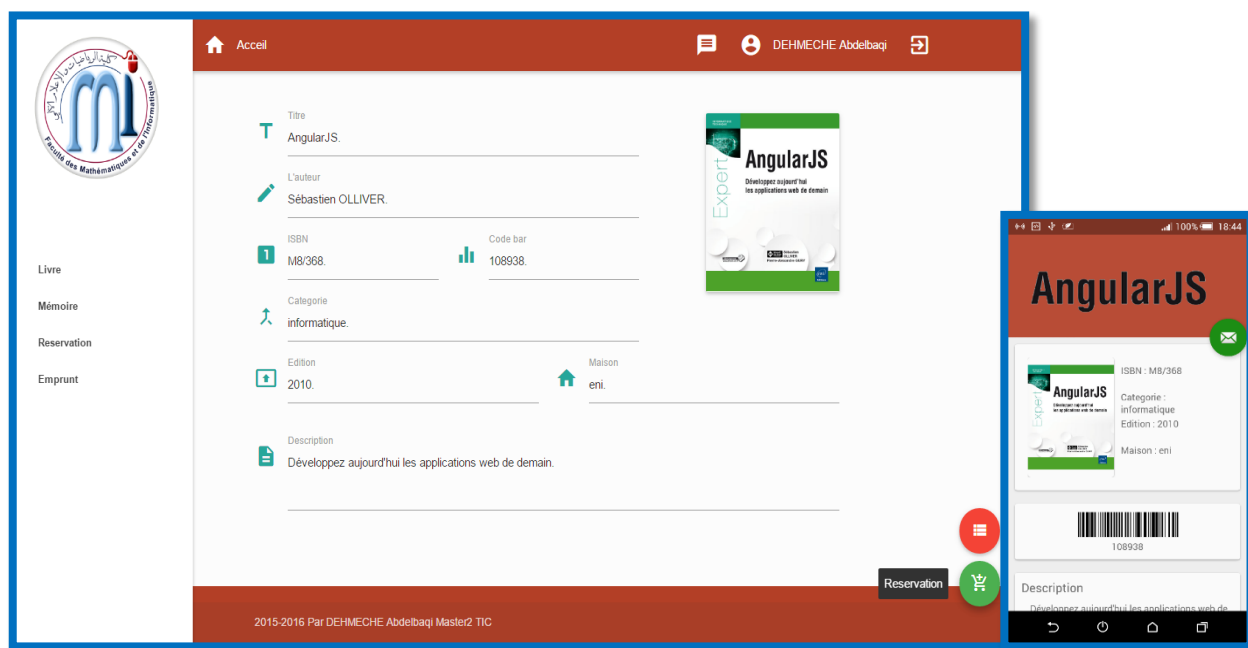


Figure 4.24 : Interface Réserve

3.8 Interface validé la réservation

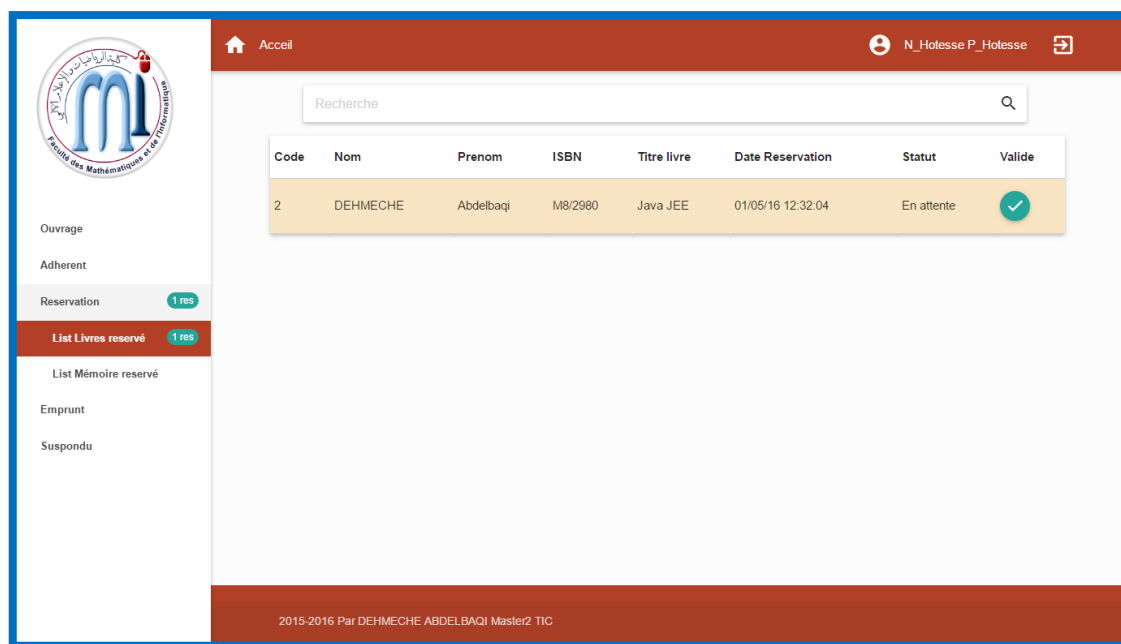


Figure 4.25 : Interface validé la réservation

3.9 Interface Profile

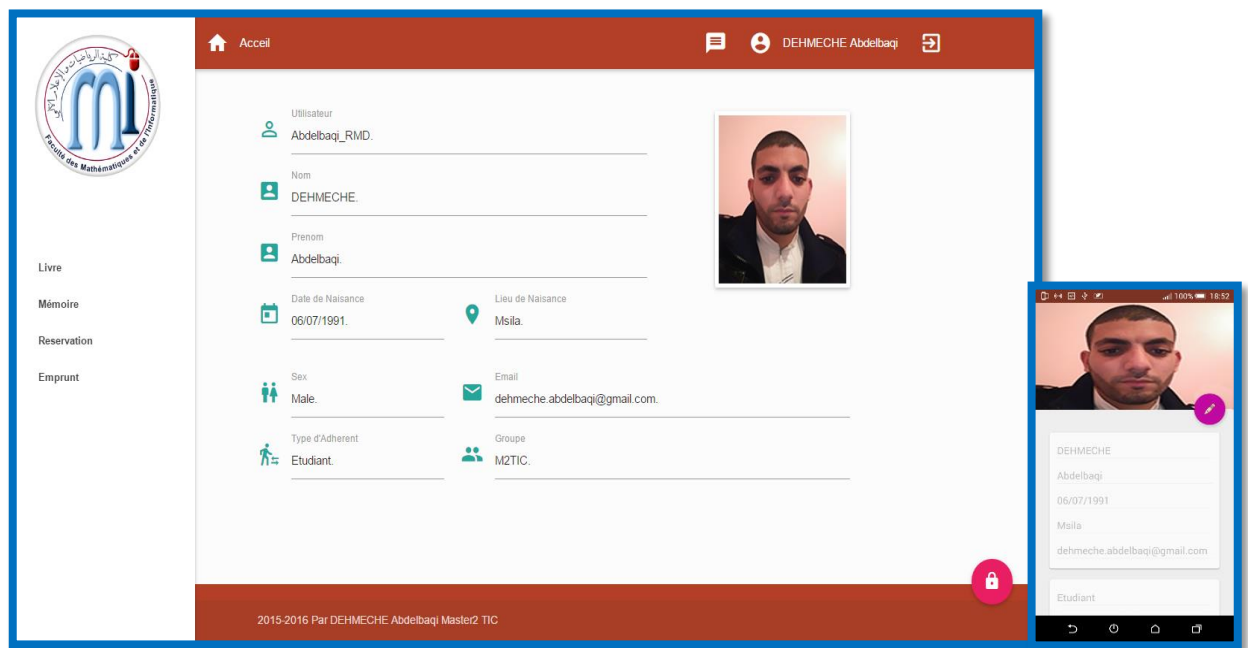


Figure 4.26 : Interface de profile

4 Conclusion

Ce chapitre a été consacré dans sa première partie à la présentation des différents outils utilisés pour la réalisation de notre service web. Nous avons introduit le maximum des concepts relatifs aux environnements de développement tels qu'Eclipse, Android Studio, MYSQL. Dans la deuxième partie nous avons présenté des interfaces de notre d'application réalisées.

CONCLUSION GENERALE

Ce projet consiste à développer un service web embarqué pour la gestion d'une bibliothèque (technologie SOAP sous Android) de la faculté des mathématiques et de l'informatique. Qui été réalisé dans le cadre du projet de fin d'études au sein de la faculté des mathématique et informatique de l'université de Mohammed Boudiaf - M'Sila -. Dans lequel nous avons appliqué toutes les compétences et les connaissances acquises durant des années d'étude.

Dans ce projet, la satisfaction des exigences des utilisateurs de notre futur système était notre premier objectif. Dans ce contexte, nous avons développé cette application pour faciliter la gestion de réservation et d'emprunt les ouvrages de ce bibliothèque.

Pour la conception de notre projet, nous avons utilisé le langage de modélisation UML, Eclipse et Android Studio comme environnement de programmation. Avec les langages de programmation Java, JSP, Servlet, HTML5, CSS, JavaScript, JQuery et Ajax, L'intégration de serveur Apache Tomcat et Axice2.

La réalisation de ce projet nous a permis d'apprendre à travailler en mode organisé reposant sur une démarche scientifique. En plus, les connaissances acquises dernière ce projet sont nombreuses que ce soit théorique à savoir : Bibliothèque, UML et Service web embarqué ou bien la pratique à savoir : Eclipse, Android Studio et MySQL

Pour la continuité de ce travail, nous proposons comme futures idées des tâches qui touchent :

- La sécurité dans ce domaine.
- Ajouter d'option de télécharger l'ouvrage électronique.
- Imprimer les états sortis.

Finalement, nous nous sommes rendu compte que le but d'un travail de recherche n'est pas forcément de donner des réponses concrètes mais d'essayer de contribuer, même si c'est d'une façon limitée, et dans notre cas nous espérons qu'on a mis le pied sur le bon chemin pour ceux qui viennent par la suite pour travailler sur ce projet ou sur des projets similaires.

BIBLIOGRAPHIE

Les Ouvrages

- [1] CHRISTIAN Sontow, UML2 pour les bases de données Modélisation - Normalisation, 2eme édition, EYROLLES, 2012.
- [2] Eric Daspet, Cyril Pierre de Geyer, PHP 5 Avancé 4^e édition Eyrolles.--> phpmyadmin

Les Mémoires / thèses

- [3] Amara Mohammed, Interopérabilité des services web hétérogène , Mémoire de projet de fin d'étude, université de Tlemcen, 2009.
- [4] Conception et réalisation d'un site web dynamique pour un magazine en ligne, BOUKERZAZA Hanane, SAOUCHI Rima, 2011-2012 Université Mentouri Constantine
- [5] D. Chappell and T. Jewell. "Java Web Services.", O'Reilly, first edition, Mar. 2002
- [6] Fabrice Rossi, Université Paris IX, "Introduction a XML".
- [7] Frédéric Moitry, Jean-Marie Cochet, Emmanuel Friedmann, Adobe Dreamweaver, DUNOD, Paris, 2007.
- [8] Guilherme Berton Machado, Frank Siqueira "Embedded Systems Integration Using Web Services", 2006.
- [12] MARKO Garqenta, Learning Android, O'REILLY Media, United States of America, 2011.
- [13] REALISATION D'UN SITE WEB DYNAMIQUE COMMERCIALE, Wahabi Mohamed, Université Virtuelle de Tunis, 2010-2011.
- [14] Riadh BEN HALIMA. « Conception, implantation et expérimentation d'une architecture en bus pour l'autoréparation des applications distribuées à base de services web ». Thèse de doctorat, Université de Toulouse et de l'université de Sfax. 14 mai 2009

Les articles

- [9] J.Garcia, E.Henriet, « Les services Web », 2004.
- [10] Kin Laskey MITRE corporation; OASIS, Reference Architecture Foundation for service oriented architecture Version 1
- [11] L.Bernard, [Stage] Développement d'une application de streaming sous Android, 12 octobre 2012.

Les Sites web

- [15] http://www.frandroid.com/android/developpement/205889_repartition-versions-android-53-kitkat-fin-mars-2016# consulté le : 03/03/2016 //uml
- [16] <http://www.jmdoudoux.fr/java/dej/chap-tomcat.html> consulté le : 14/05/2016 → Apache Tomcat
- [17] <http://www.3School.com> consulté le 01/05/2016
- [18] <http://www.benoitpiette.com/labo/introduction-aux-webservices.html#page3> consulté le : 25/04/2016
- [19] <http://www.siteduzero.com/informatique/tutoriels/les-services-web/le-protocole-de-communication-soap> consulté le : 05/04/2016
- [20] <http://connectikpeople.blogspot.com/2011/06/avantages-et-inconvenients-servicesweb.html#.UXFXn6LwmXM> consulté le : 15/02/2016

ANNEXES

1 Installation des applications sur téléphone

1.1 Installation du pilote USB

Pour pouvoir utiliser un vrai téléphone Android depuis l'éditeur de code Android studio, nous avons besoin d'installer un driver USB, puisqu'en branchant l'USB, Windows, par défaut, ne connaît pas le type de matériel.

En général, le pilote à installer se trouve dans le dossier du SDK installé et plus exactement dans un dossier qui s'appelle google-usb-driver. Selon le processeur que nous avons sur le téléphone, nous choisissons le bon pilote.

Si le pilote correspondant au téléphone n'existe pas dans le dossier indiqué, nous devons le télécharger depuis Internet.

Nous branchons le téléphone sur le port USB et l'assistant d'ajout de nouveau matériel détecté apparaît. Nous spécifions alors le chemin de pilote et procédons à son installation.

Une fois l'installation du driver terminée, nous pouvons alors commencer à utiliser le téléphone depuis eclipse.

Dans eclipse, nous ouvrons la perspective DDMS et dans l'onglet device à gauche, nous pourrions voir tous les émulateurs existants ainsi que le téléphone branché.

Nous pouvons ainsi utiliser le téléphone avec eclipse.

1.2 Paramétrage du téléphone

Nous allons configurer le téléphone pour qu'il puisse accepter le débogage et l'installation d'application de l'environnement de développement.

Les commandes à exécuter sont alors les suivantes :

- On clique sur menu puis on choisit paramètres (ou bien settings)
- On choisit applications
- On coche sources inconnues
- Cliquer sur OK quand le warning s'affiche (puisque c'est bien une application de confiance)
- Passer dans Développement
- Activer Débogage USB, Rester activé et Positions fictives

2 Installation de Tom cat et Axis2 sous Eclipse

2.1 Téléchargement des outils :

On peut télécharger Apache Tom cat et Axis à partir le site officielle : <http://tomcat.apache.org/> Et axis-2 à partir le lien suivant : <https://axis.apache.org/axis2/>

2.2 Installation dans Eclipse

Dans le menus Eclipse On suivre les étapes suivants :

```
<uses-sdk
    android:minSdkVersion="11"
    android:targetSdkVersion="21" />
<uses-permission android:name="android.permission.INTERNET"/>

<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity
        android:name=".MainActivity"
        android:label="@string/app_name" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

2.3 Crier un serveur

Puis on remplir la figure suivant :

3 Comment connecter à une base de donné MySQL

- On doit télécharger le fichier mysql-connector-java-5.1.37-bin.jar.
- Importer dans notre projet web dynamique.
- Création de la classe Connexion que contient le code suivant :

```
public class Connexion {

    public static Connection getConnection() {

        String connectionUrl=
        "jdbc:mysql://localhost:3306/bibliotheque_mi";
        Connection connection = null;

        try {
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            connection = DriverManager.getConnection(connectionUrl,
            "root", null);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        return connection;}
}
```

4 Comment faire l'appel d'un service web sous application Android

- On doit télécharger le fichier ksoap2-android-assembly-3.4.0-jar-with-dependencies.jar
- Importer dans notre projet Android.
- Création de la classe Class Connecter que contient le code suivant

```
public class Services_FUN {
    public static final String WS_NAMESPACE = "http://function.tic.com";
    public static String WS_URL = " http://localhost:
8080/Bibliotheque_Server/services/Auth_Fun?wsdl ";
    public static String ACTION_METHOD_NAME = "login";
    public static String SOAP_ACTION = "";
    public static String IP = "192.168.43.202";
    public Boolean login(String username, String password) {
        Boolean lStrResult = false;
        try {
            SOAP_ACTION = mFnGetSOAP_ACTION();
            SoapObject request = new SoapObject(WS_NAMESPACE,
ACTION_METHOD_NAME);
            request.addProperty("log", username);
            request.addProperty("mdp", password);
            SoapSerializationEnvelope envelope = new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(request);
            HttpTransportSE ht = new HttpTransportSE(WS_URL);
            ht.call(mFnGetSOAP_ACTION(), envelope);
            final SoapPrimitive lobjResponse = (SoapPrimitive)
envelope.getResponse();
            final Boolean lStrResp =
Boolean.parseBoolean(lobjResponse.toString());
            lStrResult = lStrResp;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }
    return lStrResult;
}
```

الملخص

خدمة الويب هو وسيلة للبرمجة من اجل أن تقدم خدمة على شبكة الإنترنت. خدمات الويب يمكن ان تكون مستقلة أو مرتبطة معا لتوفير وظائف محسنة. خدمات الإنترنت هي تقنيات قوية لربط أنظمة المدمجة وغيرها من الانظمة الموزعة الذكية في الشركة، نحن نعمل لتطبيق هذه التقنية من اجل حل مشكلة تسيير المكتبة الجامعية والذي يوفر جودة عالية وخدمات فعالة للطلاب على الإنترنت من خلال جهاز الموبايل. عملنا يتمثل في استعمال خدمة ويب التي تستخدم بروتوكول SOAP على الجهاز المحمول على مستوى العميل والكمبيوتر على مستوى التطبيق من جانب الخادم.

الكلمات المفتاحية : خدمة ويب، مدمج ، مكتبة، اندرويد، XML، UDDI، WSDL، SOAP، UML.

Résumé

Un service Web est un composant programmable qui fournit un service sur Internet. les services Web peuvent être autonomes ou reliés entre eux pour fournir des fonctionnalités améliorées. les services Web sont des techniques puissantes pour le raccordement des systèmes embarqués et d'autres actifs intelligents distribués dans l'entreprise. Dans notre travail, nous sommes intéressé d'appliquer cette technique au problème de gestion bibliographique afin d'offrir des services de haute qualité et efficaces à l'étudiant sur Internet à travers des équipements embarqués. Notre travail est basé sur le service Web exploitant le protocole SOAP implémenté sur des équipement mobiles de côté client et sur des pc standard pour l'application côté serveur

Mots clés : Service web, Embarqué, Bibliothèque, Android, UML, SOAP, WSDL, UDDI, XML

Abstract

A web service is a programmable component that provides a service over the Internet. Web services can be standalone or linked together to provide enhanced functionality. Web services are powerful technics for connecting embedded systems and other distributed intelligent assets into the business enterprise. in our work we are interesting to apply this technic to the bibliographic management problem in order to offer high and efficient services to the student over internet with using embedded devices. Our work is based on web service with SOAP protocol implemented on mobile devices for client side and on standard pc for server side application

Key words: web service, embedded, library, Android, UML, SOAP, WSDL, UDDI, XML.