REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Khider – BISKRA

Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie

Département informatique

Thèse

Présentée en vue de l'obtention du diplôme de Docteur en 3^{iéme} cycle LMD en Informatique

Option : Techniques de l'image et de l'intelligence artificielle

Une approche basée agent mobile pour le m-service web sémantique

Présentée par: HAMIDA Souraya

Soutenue le : juin 2014

Devant le jury :

- Pr. KHOLLADI Mohamed Kheiredine	Université d'Eloued Souf	Président
- Pr. KAZAR Okba	Université de Biskra	Encadreur
- Pr. AMGHAR Youssef	INSA de Lyon	Examinateur
- Pr. BENMOHAMED Mohamed	Université de Constantine	Examinateur
- Dr. TERISSA Labib Sadek	Université de Biskra	Examinateur
- Dr. BENNOUI Hammadi	Université de Biskra	Examinateur

DIDICACES

À

Mon pays

mes parents,

mes frères,

ma grande famille,

mes enseignants,

et mes amis,

je dédie ce modeste travail.

REMERCIEMENTS

Je remercie tout d'abord «Allah » le tout puissant, de m'avoir donnée la santé, le courage et la patience pour mener à bien ce projet de doctorat.

Par la même occasion, je tiens à exprimer mes sincères gratitudes aux personnes qui ont contribuées de près ou de loin à l'élaboration de la présente thèse de doctorat, plus particulièrement à l'encadreur monsieur *Okba Kazar*, directeur et professeur à l'université de Biskra, pour son large esprit, sa compréhension, sa disponibilité et son soutien durant toutes ces années.

Sans perdre de vue, l'honorable *Youssef Amghar*, sous-directeur et professeur à l'INSA de Lyon, de son chaleureux accueil au sein de son co-équipe SOC, ses conseils, ses fructueuses lectures et son suivi et qui mérite toute la gratitude et l'estime de ma part.

Sans oublier mes enseignants de l'Université de Biskra et plus particulièrement *Madame Dr. Aïcha-Nabila Benharkat*, maître de conférences à l'INSA de Lyon, pour son aide et ses conseils dont je salut chaleureusement.

Enfin, j'adresse mes intimes et profondes considération aux membres du jury : *Pr. Mohamed Kheiredine Kholladi* (de l'université d'Eloued Souf); *Pr. Youssef Amghar* (de l'INSA de Lyon) ; *Pr. Mohamed Benmohamed* (de l'université de Constantine); *Dr. Labib Sadek Terissa* (maître de conférences A, de l'université de Biskra); *Dr. Hammadi Bennoui* (maître de conférences A, de l'université de Biskra) pour leur compréhension et au temps qu'ils m'ont réservé à la soutenance.

Résumé

Le développement et la diffusion rapide de l'Internet et sa technologie qui les lie mutuellement depuis une décennie, ont permis la mutation intégrale de notre vie moderne. En outre, depuis la sortie des appareils mobiles hors des labos et leur propagation à travers le monde, leur existence parmi nous est devenue primordial en notre vie. Un avenir prospère de l'informatique mobile est prédit aux prochaines années. L'informatique mobile s'imposera et deviendra une technologie (informatique) dominante. En raison de contraintes de puissance et de bande passante inhérentes à l'informatique mobile, il est impératif de communiquer avec les services Web de manière aussi efficace que possible. Ce besoin d'efficacité motive notre recherche. Dans notre thèse, nous proposons l'utilisation de l'agent mobile pour la découverte et l'exécution de services web mobile. L'idée est d'explorer les propriétés des agents mobiles afin de s'adapter aux caractéristiques des services, afin d'éliminer les problèmes liés à l'environnement et les appareils mobiles. De plus, nous exploitons les avantages du Web sémantique pour automatiser la découverte et l'exécution de services Web, et enfin, nous utilisons le contexte de l'utilisateur afin de mieux répondre à la demande de l'utilisateur.

Mot clé : Agent mobile, service web mobile, ontologie, sensibilité au contexte, web sémantique.

Abstract

With the rapid growth of Internet and its related technology over the past decade, the landscape of our modern life has been changed significantly. Moreover, mobile devices have left the labs and have become essential in people's lives. Mobile computing will continue to grow in the next few years in power and pervasiveness and is poised to become the dominant computing technology. Due to power and bandwidth constraints inherent to mobile computing, it is imperative to communicate with web services as efficiently as possible. This need for efficiency motivates our research. In this paper, we propose the use of mobile agent for efficient discovery and invocation of M-services. The idea is to explore the properties of mobile agents in order to adapt to the characteristics of services, to eliminate the problems associated with mobile environments and mobile devices. Moreover, we exploit the advantages of the Semantic Web to automate the discovery and invocation of web services; finally we use the user context in order to better satisfy the demand of the user.

Keywords: Mobile agent, mobile web service, ontology, context-Aware, semantic web.

ملخص

ان التطور و الانتشار السريع للإنترنت والتكنولوجيا المتعلقة بها على مدى العقد الماضي، سمح بالتحول الكامل في حياتنا المعاصرة . بالإضافة إلى ذلك ؛ منذ خروج الأجهزة النقالة من المختبرات و انتشارها في جميع أنحاء العالم، أصبح وجودها ضروريًّا في حياتنا. ومن المتوقع ان يكون المستقبل مزدهرًا بالنسبة للحوسبة المتنقلة في السنوات القادمة. إذ أنّها سوف تسود وتصبح التكنولوجيا المهيمنة. بسبب متطلبات القوة وعرض النطاق الترددي الملازم للحوسبة النقالة، فإنه لابد من التواصل مع خدمات ويب بأكبر قدر من الكفاءة. وهي أبرز عوامل انتقائنا لهذا البحث.

في أطروحتنا، نقترح استخدام وكيل متنقل من اجل اكتشاف و تنفيذ الخدمات على شبكة الإنترنت النقالة. والفكرة هي استعمال خصائص وكلاء المحمول من أجل التكيف مع خصائص الخدمات، من أجل القضاء على المشاكل المتعلقة بالبيئة و الأجهزة النقالة. بالإضافة إلى ذلك، ارتأينا استغلال فوائد الويب الدلالي لجعل عمليتا اكتشاف وتنفيذ خدمات ويب اوتوماتيكية ، وأخيرا ، استخدمنا الحساسية لسياق المستخدم من أجل تلبية احتياجات المستخدم بطريقة أفضل

الكلمات المفتاحية: العميل الجوال، خدمة الواب الجوال، الانطولوجيا، الحساسية للسياق، الشبكة المعنوية.

Table des matières

Tables des figures

Introduction générale

De nos jours, les services web sont devenus un important dilemme à la création des applications web. Ces derniers sont destinés à l'accès aux données, aux contenus hétérogènes et à inter-opérer les systèmes d'information. Malheureusement, l'absence de la sémantique des technologies de base des services web génère un handicap. Car il ne permet en aucun cas l'automatisation des diverses tâches liées aux services web, tels que : l'invocation et la découverte. Pour solutionner ce problème, nous nous sommes amenés à enrichir les descriptions des services web par d'autres informations complémentaires, réutilisables, partageables et compréhensibles par les machines. Ces informations sont des données et des métadonnées, qui permettent l'interprétation des descriptions des services web. Les services web sémantiques sont une évolution nécessaire aux services web. Ils se basent entièrement sur les langages du web sémantique, particulièrement sur les ontologies.

L'émergence du réseau sans fil ainsi que la haute technologie et la fiabilité des appareils mobiles ont mis en apogée l'utilisation de l'informatique mobile. Actuellement, la majeure partie des consommateurs ne peuvent plus se séparer de leurs mobiles. En plus, l'usage du dispositif mobile est en vogue et en plein essor. Actuellement, les services web mobiles dessinent un horizon prometteur en matière économique. Ils offrent de nouveaux services personnalisés aux consommateurs. Entre autre, en raison de contraintes de puissance et de bande passante inhérentes à l'informatique mobile, il est impératif de communiquer avec les services web d'une manière plus ou moins efficace que possible. Ce besoin d'efficacité motive nos recherches.

Entre autre, les agents mobiles disposent de plusieurs avantages à l'égard des services web mobiles. Les agents mobiles fonctionnent de manière autonome et asynchrone. Ils s'adapter dynamiquement en fonction de l'environnement d'exécution, réduisent le trafic réseau, et se déplacent d'un site à un autre pour effectuer les tâches localement.

Nos objectifs sont la résolution de certains obstacles scientifiques liés à la découverte et l'invocation des services web sémantique dans l'environnement mobile. L'idée est d'explorer les propriétés des agents mobiles, afin de les adapter aux caractéristiques des services. Ceci, afin d'éliminer les problèmes liés à l'environnement et aux appareils mobiles. Additionnellement, nous exploitons les avantages du web sémantique pour automatiser la découverte et l'exécution des services web. Ensuite, nous utilisons la sensibilité au contexte afin de satisfaire au mieux la demande de l'utilisateur.

La présente étude englobe essentiellement quatre chapitres. Le premier chapitre est consacré à l'étude générale du service web. De plus, Ce chapitre réuni trois sections distinctes. La première partie est consacrée à l'étude générale sur la technologie des services web électroniques. La seconde partie sera réservée uniquement aux travaux sur les services web sémantique. Nous étudierons le web sémantique et les approches proposées à la réalisation des services web sémantiques. Enfin et en dernier lieu, nos recherches s'orientent vers les services web mobile. Par ailleurs, nous analyserons les avantages et les inconvénients des deux technologies services web électroniques et services web mobile, la classification des services web mobile, les appareils mobiles.

Le second chapitre, est destiné à l'analyse détaillée des différentes architectures développées ces dernières années en ce domaine (service web mobile). Notre objectif est la classification d'une manière efficace ces méthodes afin de mettre en évidence les particularités ainsi que les avantages et les inconvénients de chacune d'elle. Cet état de fait nous permettra, par la suite, de mettre en évidence les critères de performances qui nous faciliterons l'orientation et la fixation de notre choix sur les solutions retenues dans le cadre de notre problématique.

Le troisième chapitre est dédié à notre conception de l'architecture proposée. Dans un premier temps, nous étudions successivement, les objectifs de notre système tout en donnant une description sur l'architecture et la présentation des différentes étapes pour atteindre l'objectif fixé et escompté.

Le quatrième chapitre a été consacré à l'implémentation des différentes étapes décrites au précédant chapitre. Ensuite, nous mettons au clair les résultats obtenus.

Pour clore, nous résumons l'étude et les contributions apportées, tout en relatons les perspectives d'horizon à prendre en charge dans le futur.

Chapitre I : Services web mobile

I.1 Introduction

De nos jours Il n'échappe à personne que, l'Internet est devenu un véhicule de service, plutôt qu'un référentiel d'informations statiques. Actuellement, plusieurs entreprises rendent accessible leurs services sur le web; cette opération est un service web électronique ou « service web ».

L'émergence du nouveau réseau sans fil ainsi que la haute technologie et la fiabilité des appareils mobiles ont mis en apogée l'utilisation de l'informatique mobile. A présent, cette industrie (du réseau sans fil) est devenue omniprésente. La majeure partie des consommateurs ne peuvent se séparer de leurs mobiles. L'usage du dispositif mobile est en vogue et en plein essor. Ces derniers temps, la combinaison des réseaux mobiles et services web dénommé service web mobile, ouvrent un horizon prometteur en matière économique. Ces derniers (service web mobiles) offrent de nouveaux services personnalisés aux consommateurs sur leurs appareils mobiles.

Ce chapitre s'articule essentiellement sur trois sections. A la une, nous étudierons en général la technologie des services web. On se consacrera particulièrement aux avantages et à la méthode d'invocation des services web. A la seconde, nous mettrons en relief un aperçu général sur les services web sémantique. Nous étudierons le web sémantique et les approches proposées à la réalisation des services web sémantiques. Enfin et en dernier lieu, nos recherches s'orienteront aux services web mobile. Par ailleurs, nous analyserons les avantages et les inconvénients des deux méthodes (services web électronique et les services web mobile), la classification des services web mobile, les appareils mobiles et plates-formes supportant les services web mobiles [Hamida 12 c].

I.2 Panorama des technologies du service web

Le progrès dont connait le web, a fait surgir un besoin lors d'une application client d'invoquer un service d'une application serveur ceci en utilisant Internet. Ce besoin a été l'origine de l'appellation «services web». Ainsi on ne peut ignorer que le service web permet la connexion des différentes applications. L'utilité de cette technologie devient évidente et importante. A cette effet, les activités de recherches s'intensifient, se développement et convergent vers le service web. Celui-ci connait un important dynamisme.

Le groupe de World Wide Web Consortium (W3C[1]) dans le cadre du service web a émis la définition suivante : « Un service web est un système logiciel conçu pour permettre l'interopérabilité de machine-à-machine sur un réseau. Dispose d'une interface décrite dans un format exploitable par machine (en particulier WSDL). D'autres systèmes exerce une interaction avec le service web d'une manière prescrite par sa description en utilisant des messages SOAP, typiquement transmis avec le protocole HTTP[2] avec une sérialisation XML[3], en conjonction avec d'autres standards relatifs au web» [Booth 04].

Comparé avec le système traditionnel client/serveur, le service web travaille non seulement au sein de l'entreprise, mais aussi en inter-entreprises. Ce dernier dispose de propriétés telles que l'indépendance de langage de programmation, la gestion par message de la communication, se lie facilement à différents transports, et faiblement couplés [Wonsuk 06].

I.2.1 Avantages du service web

Le fondement du service web est le partage des applications et des programmes en un ensemble d'éléments réutilisables appelés service où chacun de ces éléments effectuent une tâche principale et efficace, afin de faciliter l'interopérabilité entre tous ces services web. Toutefois, le service web [Chappell 02] permet:

[1] WorldWideWeb Consortium (http://www.w3.org) : une communauté internationale qui élabore des protocoles et des standards pour assurer la croissance du Web à long terme.
[2]HTTP: HyperText Transfer Protocol.
[3] XML: eXtensible Markup Language.

- L'interopérabilité au sein des environnements applicatifs, de manière que les logiciels et les applications écrits dans différents langages de programmation, évoluant sur différents systèmes d'exploitation peuvent communiquer et/ou échanger des données entre eux sans difficulté;
- De profiter des différents environnements et langages de développement par une publication, localisation, description et une invocation via XML. Les services web sont très flexibles, indépendants des langages de programmation et des systèmes d'exploitation;
- D'accéder aux applications à travers les pare-feu en utilisant via le langage XML et les protocoles Internet standards comme HTTP sur le port 80, généralement ouvert. Cela permet d'assurer une transmission de données transactionnelles et sécurisée;
- Permet une utilisation à distance via n'importe quel type de plate-forme, et l'accessibilité depuis n'importe quel type de clients;
- Concourt au développement d'applications distribuées. Ils appartiennent à des applications capables de collaborer entre elles de manière transparente pour l'utilisateur, et permettent d'avoir un partage des fonctionnalités et facilitent grandement le développement.

En résumé, d'après ce que nous constatons les services web disposent de multitudes avantages, tel que: l'utilisation de standards universels, l'indépendance de plate-forme, un environnement universel pour les systèmes d'information distribués, l'utilisation de plusieurs protocoles de transfert (par exemple HTTP, SMTP, FTP, . . . etc.), le codage des messages utilisant le langage XML, un comportement compatible aux pare-feu, et la localisation par l'URI. Ceci découle du résultat de développement exceptionnel des technologies d'information, dont une mise en entente a été établie sur un certain nombre de protocoles et d'approches qui favorisent l'interopérabilité entre les plates-formes, les systèmes d'exploitation et les langages de programmation.

L'infrastructure (des services web) s'est concrétisée autour de trois spécifications considérées comme standards, à savoir SOAP, UDDI et WSDL [Ben halima 09] dont l'étude sera entamée ultérieurement.

I.2.1.1 Transport : SOAP

Le standard actuel qui assure la messagerie est le protocole Simple Object Access Protocol (SOAP) [Gudgin 03]. C'est une spécification XML définissant un protocole d'une aisance d'échange de données structurées entre un réseau d'applications au sein d'un environnement totalement distribué et hétérogène. Indépendant du contenu du message, Il permet la structuration des messages destinés à des objectifs particuliers allant d'un simple échange de données jusqu'à l'appel d'opérations à distance. Il a le pouvoir d'être employé dans tous les types de communication : synchrones ou asynchrones, point à point ou multipoint.

Le SOAP est un document XML dont la structure est spécifiée par des schémas XML. Toutefois tout message SOAP se compose d'un élément enveloppe contenant un élément entête et un élément corps. Le premier contient l'entête du protocole de transport (par exemple HTTP) ainsi que les métadonnées qui portent sur d'éventuelles propriétés non fonctionnelles du service (jeton de sécurité, contexte de transaction, certificat de livraison, etc.). La partie corps regroupe, quant à elle, les éléments métier tels que :

- les appels de méthode, avec transferts de données spécifiques, dans le cadre d'une requêté (le nom de la méthode ainsi que la valeur de ses paramètres),
- Les transferts (seuls) de données spécifiques dans le cadre d'une réponse (la valeur des paramètres de retour de la méthode).

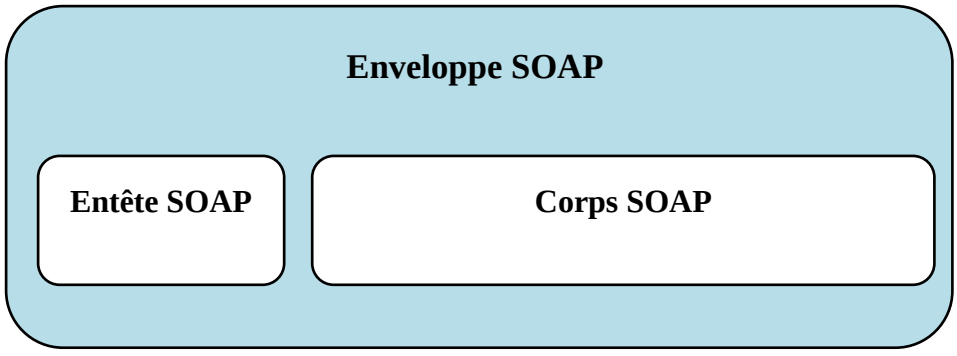

Figure I- 1: Structure d'un message SOAP [Dumez 10].

I.2.1.2 Découverte : UDDI

A l'instar de toutes les ressources disponibles sur le web, ce dernier serait pratiquement impossible à localiser sans passer par des outils de recherche. Les annuaires des

services web représentent les bases des services où les fournisseurs peuvent publier, enregistrer des informations sur leurs services et leurs profils. Les annuaires des services web peuvent être implémentés par des services web accessibles par des applications informatiques. Ces annuaires offrent des informations sur les services web disponibles dans l'annuaire correspondant à la requête de recherche du client.

Universal Description, Discovery and Integration (UDDI) [Bellwood 02] est une spécification technique pour la description, la découverte et l'intégration des services web. UDDI constitue donc une partie importante de la pile des services web, permettant aux personnes de publier et rechercher des services web. UDDI est une norme d'annuaire de services web appelée via le protocole SOAP. Les données stockées dans l'annuaire UDDI sont structurées en trois catégories.

- ***Pages blanches :*** dispose d'une liste des entreprises ainsi que les informations associées à ces dernières. Nous avons les informations tels que le nom de l'entreprise, ses coordonnées, la description de l'entreprise et également l'ensemble de ses identifiants.
- ***Pages jaunes :*** est une collecte d'informations permettant de classer les entreprises, notamment l'activité, la localisation, etc.
- ***Pages vertes :*** Elle contient des informations techniques précises sur les services fournis. Typiquement on indiquera dans ces informations les adresses web des services et les moyens d'y accéder.

Le protocole d'utilisation de l'UDDI contient trois fonctions de base :

- ***publish :*** pour enregistrer un nouveau service,
- ***find :*** pour interroger l'annuaire,
- ***bind :*** pour effectuer la connexion entre l'application client et le service.

Il est possible de constituer des annuaires UDDI privés, où l'usage se limitera à l'intérieur de l'entreprise.

I.2.1.3 Description : WSDL

Web Services Description Language (WSDL) est basé sur XML, il permet de décrire le service web, tout en précisant les méthodes disponibles, les formats des messages d'entrée

et de sortie, et la méthode d'accession. Les fichiers WSDL sont des fichiers XML sans sémantique.

L'élément racine d'une description WSDL est une définition. Chaque document définit un service comme une collection de points finals ou ports. Chaque port est associé à un rattachement spécifique qui définit la manière avec laquelle les messages seront échangés. Chaque rattachement établit une correspondance entre un protocole et un type de port. Un type de port se compose d'une ou plusieurs opérations qui représentent une définition abstraite des capacités fonctionnelles du service. Chaque opération est définie en fonction des messages échangés au cours de son invocation. La structure du message est définie par des éléments XML associés à un schéma de type spécifique [Christensen 01].

Ainsi, pour définir ces services le document WSDL utilise les éléments suivants:

- ***Types :*** il définit les types de données échangées.
- ***Message :*** il définit d'une manière abstraite les données transmises
- ***Operation :*** il décrit d'une manière abstraite les actions supportées par le service.
- ***Port Type :*** (appelé Interface depuis WSDL2.0) : il représente un ensemble d'opérations correspondant chacune à un message reçu ou émis.
- ***Binding (Rattachement) :*** c'est un protocole de communication et un format de données échangées pour un port.
- ***Port :*** c'est une adresse d'accès au service.
- ***Service :*** qui regroupe un ensemble de ports.

En résumé, le document WSDL représente un contrat entre client et service web qui détermine l'utilisation du service. Le WSDL présente l'intérêt d'une indépendance de tout langage de programmation ou plate-forme du fait qu'il est basé sur XML. En utilisant le WSDL, le client peut alors localiser le service web et invoquer une de ses fonctions publiquement accessible. Le WSDL étant un format directement interprétable par la machine, il est même possible d'automatiser ce processus afin d'intégrer de nouveaux services en écrivant peu ou pas le code manuellement.

I.2.1.4 Invocation d'un service web

Les étapes les plus importantes de l'invocation d'un service web sont (voir Figure I-2) [Ben Halima 09]:

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 URL[4] 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'accomplie 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.

[4] URL : Uniform Resource Locator.

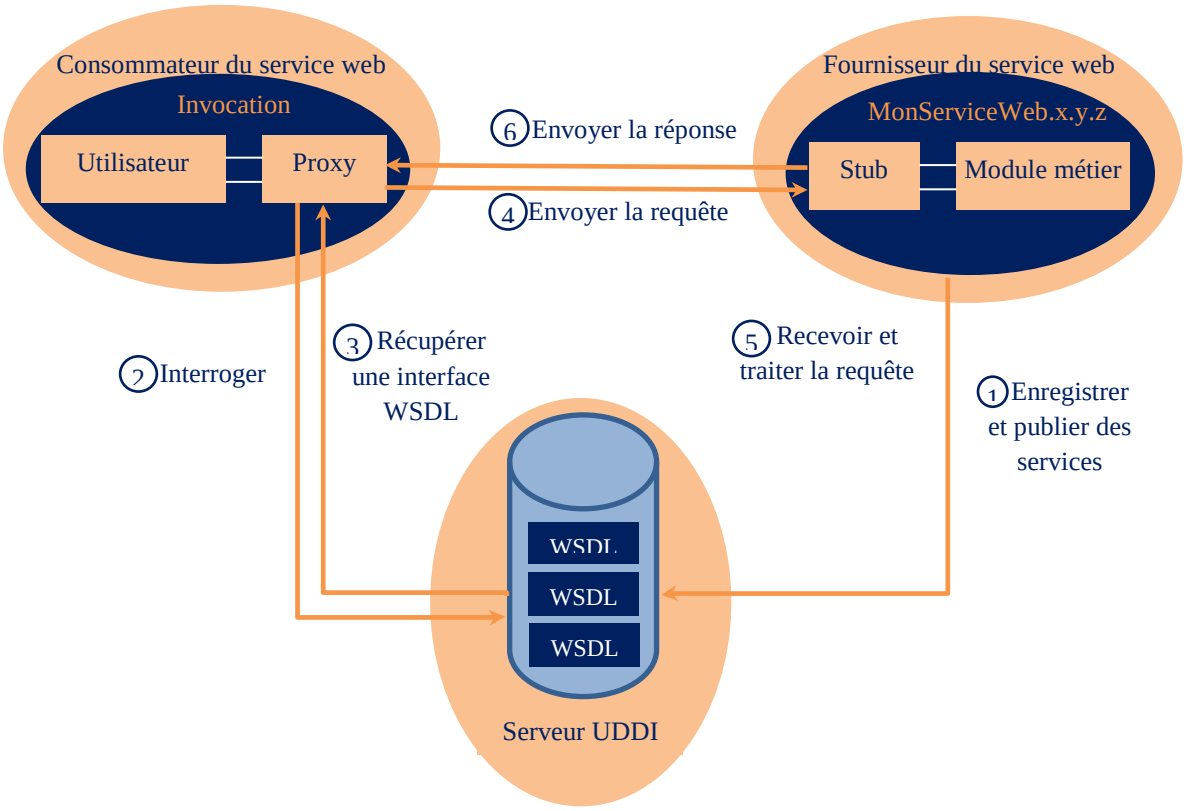

Figure I- 2: Les étapes d'invocation d'un service web [Ben halima 09].

I.3 Panorama des services web sémantique

L'absence d'une sémantique des technologies de base des services web pose un handicape avec la description en WSDL d'un service web; ce qui ne permet en aucun cas l'automatisation des diverses tâches liées aux services web, exemple: la publication, l'invocation, la découverte, . . . etc. Afin de solutionner cette carence, nous étions amenés à amender les descriptions des services web par d'autres informations complémentaires, réutilisables, partageables et compréhensibles par les machines. Ces informations sont des données et métadonnées, qui permettent d'interpréter les descriptions des services web. En d'autres termes, c'est la sémantique des descriptions des services web. Donc les services web sémantiques se situent à la convergence de deux domaines de recherche importants qui concernent les technologies de l'Internet : « le web sémantique et les services web ». Les services web sémantiques sont une évolution nécessaire des services web. Ils se basent totalement sur les langages du web sémantique, plus particulièrement sur les ontologies.

I.3.1 Le web sémantique

Inventé en 1989 par Tim Berners-Lee directeur et fondateur du World Wide Web Consortium, le web sémantique fournit un cadre de travail commun, permettant aux données d'être partagées et réutilisées. Un grand effort a été mené en collaboration avec le World Wide Web Consortium et la participation d'un grand nombre de chercheurs et d'industriels associés. La vision courante du web sémantique proposée par Berners-Lee et al [Berners-Lee 01] peut être représentée dans une architecture en plusieurs couches différentes (voir Figure I-3).

Les couches les plus bases assurent l'interopérabilité syntaxique : la notion d'URI[5] fournit un adressage standard universel permettant l'identification des ressources tandis que Unicode est un encodage textuel universel pour échanger les symboles. Rappelons que l'URL, comme l'URI, est une chaîne courte de caractères qui est aussi utilisée pour identifier des ressources (physiques) par leur localisation.

XML fournit une syntaxe décrivant la structure du document, créer et manipuler des instances des documents. Il utilise l'espace de nommage (namespace) afin d'identifier les noms des balises utilisées dans les documents XML. Le schéma XML permet de définir les vocabulaires des documents XML valides. Cependant, XML n'impose aucune contrainte sémantique à la signification de ces documents, l'interopérabilité syntaxique n'est pas suffisante pour qu'un logiciel puisse "comprendre" le contenu des données et les manipuler d'une manière significative.

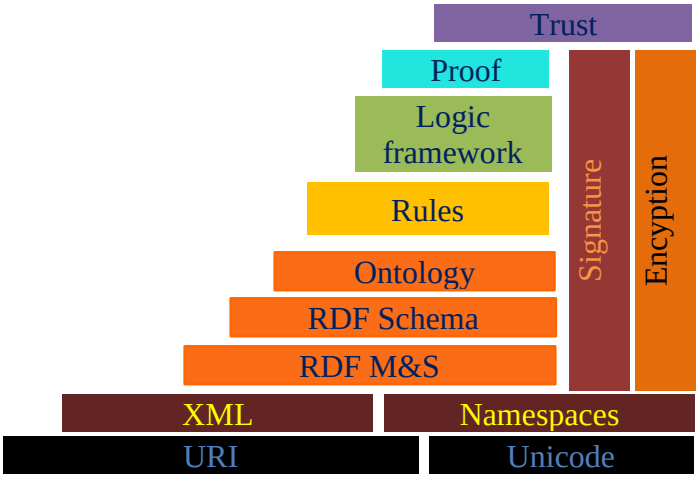

Figure I- 3: Architecture du web sémantique [Berners-Lee 01].

[5] URI : *Uniform Resource Identifier.*

Les couches RDF M&S (RDF Model and Syntax) et RDF Schéma sont considérées comme les premières fondations de l'interopérabilité sémantique. Elles permettent de décrire les taxonomies des concepts et des propriétés (avec leurs signatures). RDF fournit un moyen d'insérer du sémantique dans un document, l'information est conservée principalement sous forme de déclarations RDF. Le schéma RDFS décrit les hiérarchies des concepts et des relations entre les concepts, les propriétés et les restrictions domaine/co-domaine pour les propriétés.

La couche suivante Ontologie décrit les sources d'information hétérogènes, distribuées et semi-structurées tout en définissant le consensus du domaine commun et partagé par plusieurs personnes et communautés. Les ontologies aident la machine et l'humain à communiquer avec concision en utilisant l'échange de sémantique plutôt que de syntaxe seulement.

Les règles sont aussi un élément clé de la vision du web sémantique, la couche Règles offre la possibilité et les moyens d'intégration, de dérivation, et de transformation de données provenant de sources multiples, etc.

La couche Logique se situe au-dessus de la couche Ontologie. Certains considèrent ces deux couches comme étant au même niveau, comme des ontologies basées sur la logique et permettant des axiomes logiques. En appliquant la déduction logique, on peut inférer de nouvelles connaissances à partir d'une information explicitement représentée.

Les couches Preuve (Proof) et Confiance (Trust) sont les couches restantes qui fournissent la capacité de vérification des déclarations effectuées dans le web sémantique. On s'oriente vers un environnement du web sémantique fiable et sécurisé dans lequel nous pouvons effectuer des tâches complexes en sûreté. D'autre part, la provenance des connaissances, des données, des ontologies ou des déductions est authentifiée et assurée par des signatures numériques, dans le cas où la sécurité est importante ou le secret nécessaire, le chiffrement est utilisé.

I.3.2 Les services web sémantique

Les services web sémantiques sont des services web décrits de telle sorte qu'un agent logiciel puisse interpréter les fonctionnalités offertes par le service web. Un agent logiciel doit être capable de lire la description d'un service web pour déterminer si le service web fournit

les fonctionnalités désirées, et s'il est lui-même capable d'utiliser ce service. Pour aboutir à cette fin, la description du service web doit être supplémentée en information sémantique interprétable par machine. Les paramètres du service web doivent être décrits de façon qu'un agent logiciel puisse connaitre leur signification. Cela peut s'accomplir en définissant les vocabulaires organisés en ontologies.

Dans le domaine des services web, les ontologies sont utilisées pour annoter les descriptions des fonctionnalités des services web sémantique. Ainsi pour qu'un agent logiciel puisse avoir connaissance de la sémantique d'une description d'un service web, il lui suffit simplement d'accéder à une ontologie du domaine.

La combinaison des technologies du web sémantique avec celles des services web nous permet:

- L'automatisation de la découverte des services web, afin de permettre une localisation automatique des services web, qui fournissent une fonctionnalité particulière et qui répondent aux propriétés demandées par l'utilisateur. Afin d'effectuer une découverte automatique, le procédé de découverte doit se baser sur une similitude sémantique entre la description déclarative, faite par l'utilisateur, du service demandé et celle du service offert.
- La composition automatique des services web. est une combinaison de plusieurs services pour obtention de nouvelles fonctionnalités.
- D'automatiser l'invocation d'un service web, implique l'automatisation de l'exécution du service web par le programme d'utilisateur ou par un agent.
- D'automatiser l'interopérabilité des services web.

I.3.3 Approches proposées pour la réalisation des services web sémantiques

En littérature, diverses approches ont été proposées pour permettre la réalisation des services web sémantiques. Parmi ces approches nous avons : WSDL-S, OWL-S, IRS-II, et WSWF.

I.3.3.1 WSDL-S (Web Service Description Language-Semantic)

WSDL-S est un langage de description sémantique des services web [Akkiraju 05]. Une description WSDL-S de service web est une description WSDL augmentée de sémantique, cette sémantique est ajoutée en deux étapes :

La première étape consiste à faire référence (dans la partie définition de WSDL), à une ontologie dédiée au service à publier.

La seconde étape consiste à annoter les opérations de la définition WSDL de sémantique en ajoutant deux nouvelles balises : balise action et balise contrainte.

- ***La balise Action :*** représente l'action de l'opération
- ***La balise Contrainte :*** représente les prés et post conditions d'une opération

I.3.3.2 OWL-S (Ontology Web Language for Services)

OWL-S est un langage d'ontologie destiné aux services web. Il est basé sur le langage OWL. Cette ontologie a pour objectif de d'écrire de façon non ambiguë les services web, de telle sorte qu'un agent logiciel puisse exploiter automatiquement ces informations. OWL-S permet : La découverte automatique, la composition et l'interopérabilité de services web ainsi que la surveillance automatique de leurs exécutions [Martin 04].

OWL-S décrit un service à l'aide des trois classes définie ci-après:

- ***ServiceProfile :*** définit le service web.
- ***ServiceModel :*** définit le fonctionnement du service web.
- ***ServiceGrounding :*** définit la manière d'accéder au service web.

I.3.3.3 IRS-II (Internet Reasoning Service)

IRS-II [Motta 03] est une architecture pour les services web sémantiques. Où diverses ontologies sont définies :

- ***Ontologie du domaine (Domain model) :*** permet de décrire le domaine d'une application (véhicules, maladies).

- ***Ontologie de tâche à résoudre (Task models) :*** fournit une description générique de la tâche à résoudre, spécifie les types d'entrées (input) et sortie (output), le but à atteindre et les prés conditions à satisfaire.
- ***Ontologie des méthodes de résolution d'un problème (Problem Solving Methods (PSMs)) :*** sépare la description de ce qu'un service fait des paramètres et des contraintes d'une mise en œuvre particulière.
- ***Liens (bridges) :*** permettent la correspondance entre les différents modèles d'une application.

Les principaux composants de l'architecture IRS-II sont : le serveur IRS-II (IRS-II server), l'éditeur de services (IRS-II Publisher) et la partie client (IRS-II Client). Ces trois composants interagissent entre eux via le protocole SOAP.

I.3.3.4 WSMF (Web Service Modeling Framework)

WSMF est un modèle de représentation des divers aspects relatifs aux services web. L'objectif principal de cette approche est de permettre le développement du commerce électronique par application des technologies du web sémantique aux services web. Le WSMF se base sur quatre éléments : les ontologies fournissant la terminologie utilisée par les autre éléments ; les répertoires d'objectifs définissant les problèmes qui doivent être résolus par les services web ; les descriptions des services web et un ensemble de médiateurs contribuant à outrepasser les problèmes d'interopérabilités [Fensel 02].

I.4 Panorama des services web mobiles

Les services web mobiles sont l'application de la technologie des services web à l'environnement mobile [Farley 05]. Une large et commune définition des services web mobile est le fait de fournir et consommer des services grâce à des appareils mobiles comme les téléphones cellulaires, assistants numériques personnels, ordinateurs portables, ou n'importe quel autre appareil sans fil. Habituellement les services web mobile devraient être la prochaine vague des services web électroniques. Ces derniers permettent aux utilisateurs d'accéder aux services en tous temps et tous lieux.

I.4.1 Les différences entre les services web électronique et les services web mobile

Plusieurs aspects caractérisent toute solution des services web électronique (tels que la disponibilité, la fiabilité, l'ubiquité, la facilité d'utilisation et l'instantanéité). Sesseler et al. [Sesseler 02] ont identifié les deux derniers comme principales propriétés de différenciation entre les services web électronique et les services web mobile.

La maniabilité d'utilisation décrit le confort à l'utilisation des appareils et des services en ce qui concerne la taille (appareil, touches et écran), les capacités multimédia (couleur, son et CPU), et les interfaces utilisateur (capacités d'entrée et les menus). La souplesse d'utilisation est également affectée par la méthode d'accès (permanente ou non) et les paramètres de performance du réseau. Les appareils mobiles utilisés à l'accession aux services web mobile ont tendance à s'entendre avec un plus petit ensemble de ressources en termes de mémoire, capacité de traitement, capacité de stockage, et la taille d'affichage par rapport aux dispositifs utilisés dans les services web électronique.

Le second et principal aspect à considérer est l'instantanéité, démontre la rapidité et l'efficacité dont est effectuée une transaction. Il offre également des activités impulsives et non planifiées des utilisateurs. Considérant, un chauffeur au sein de son véhicule écoute sa chanson préférée à la radio. En ce temps, l'utilisateur désire l'achat du CD de la chanson. Au scénario traditionnel du commerce électronique, l'opération de l'utilisateur est retardée jusqu'à ce qu'il accède à un ordinateur relié à Internet. Maintenant avec les services web mobile, l'utilisateur est en mesure de passer directement sa commande, ceci est le résultat du facteur plus élevé de l'instantanéité.

Les ordinateurs de bureau sont susceptibles d'avoir une bande passante plus large et être constamment en ligne, simplement ils souffrent d'une restriction sur la vie privée et selon la politique de l'employeur en matière d'utilisation et la configuration du système. Les ordinateurs personnels fournissent un degré élevé de facilité d'utilisation du fait que l'utilisateur a le plein contrôle administratif. Les points d'accès publics à Internet n'ont aucune ou peu de fonctions de personnalisation avancées. Les ordinateurs portables et assistants numériques personnels, ces derniers font face à ces notions d'intimité et de personnalisation, mais disposent d'une bande passante plus étroite en raison de contraintes d'accès au réseau sans fil et aux capacités limitées de multimédia. Les ordinateurs de poche ont encore plus de

fonctions multimédias restreint, mais ils offrent un degré élevé de portabilité et peut être utilisé instantanément en de nombreuses situations.

Les services web mobile sont généralement représentés comme un sous-ensemble de tous les services web électronique, ce qui implique que tout service web électronique pourrait être mis en relation à partir d'un dispositif sans fil. Au lieu de cela, Sesseler et al. [Sesseler 02] voient une relation inverse entre la facilité d'utilisation et l'instantanéité couvrant une transition progressive des services web électronique aux services web mobile (voir Figure I-4). Par conséquent, les services web mobile devraient être reconnus comme une occasion d'affaires unique avec ses propres caractéristiques et contraintes, et n'est pas une extension des services web électronique.

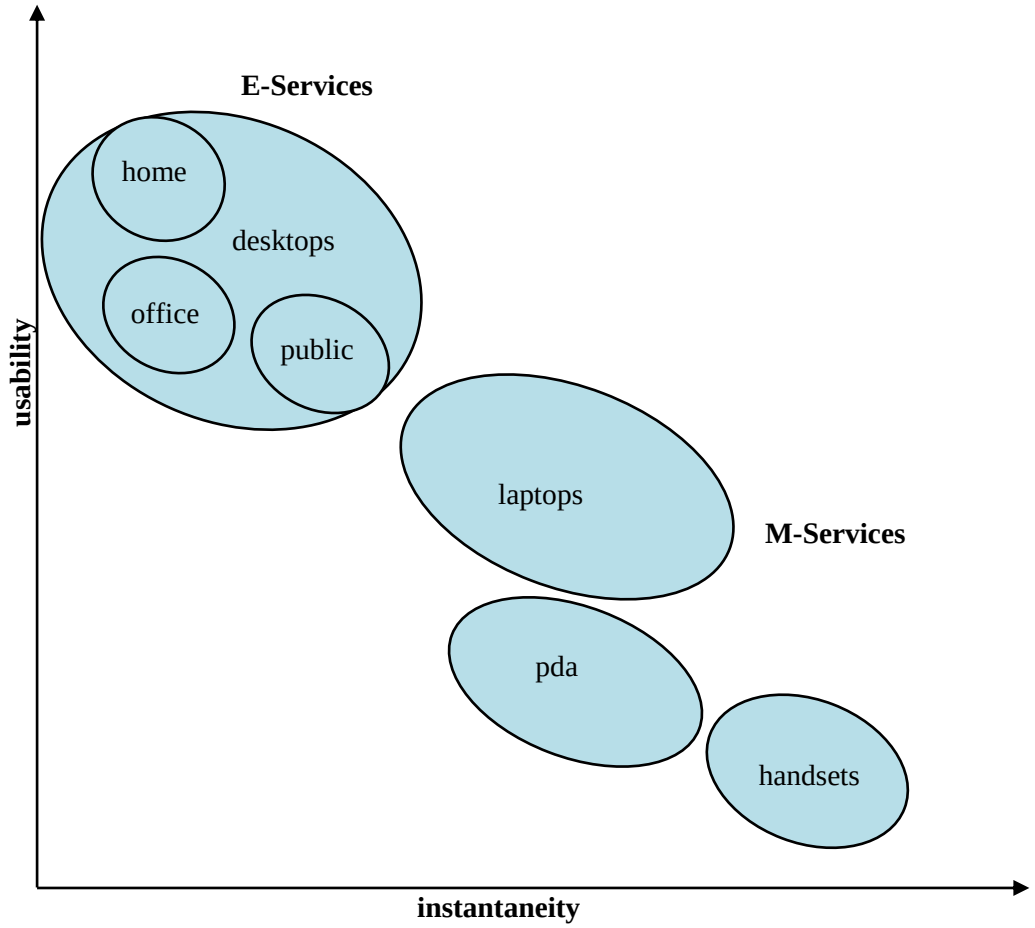

Figure I- 4:Relation entre la facilité d'utilisation et l'instantanéité [Sesseler 02].

I.4.2 Services web mobile : Perspective de l'utilisateur

L'utilisation typique du service web électronique prend plus de temps qu'une utilisation typique du service web mobile. Fréquemment, un utilisateur de service web électronique désire comparer les offres concurrentes et basculer entre eux. Il traite beaucoup plus l'information avant de procéder à la fixation de son choix. Les services web mobile répondent à un besoin ou un désir spontané tout en accomplissant une tâche bien définie dans un contexte bien défini. Le processus respectifs et l'interface utilisateur doivent être conçus pour permettre une utilisation simple.

Certaines similitudes entre les services web électronique et les services web mobile, plus particulièrement en matière des questions communes suivantes: « la sécurité, la personnalisation, l'agrégation du service, la haute disponibilité, l'indépendance du réseau et terminal » devraient être abordés. La sécurité est un défi fondamental de tout projet de business électronique ou de business mobile est confronté. C'est un sujet largement documenté dans la littérature avec des solutions impliquant l'utilisation de cryptage, l'authentification, l'autorisation et la non-répudiation des mécanismes. Personnalisation des mécanismes devant conduire à aider l'expérience d'utilisateur afin de gagner du temps à la navigation et à la saisie des données. L'agrégation des services fournira à l'utilisateur une vue homogène sur son expérience du commerce mobile.

La disponibilité est une autre contrainte commune: les systèmes doivent garantir une haute disponibilité, être robuste par rapport aux défaillances matérielles et logicielles, et de permettre les opérations de maintenance lisses comme la reconfiguration instantanée. La maintenance d'un haut niveau de fiabilité créera une confiance et une loyauté des relations entre partenaires commerciaux. Les interfaces utilisateurs doivent être suffisamment flexibles et adaptables qui équipent l'appareil cible. Les applications doivent détecter le type de terminal utilisé, de fournir des interfaces utilisateurs et le contenu en conséquence. En raison de la possibilité de couverture du réseau sans fil intermittente, les applications doivent être capables de travailler dans un mode hors ligne et de faire usage de méthodes pour restaurer le contexte de l'utilisateur après une interruption.

I.4.3 Les Services web mobiles: perspective de fournisseur

L'expérience de l'utilisateur en matière de commerce électronique (basé sur Internet) doit répondre à la demande de base qui est: le niveau de sécurité, la facilité d'utilisation, la personnalisation et la disponibilité des services de commerce mobile. Les fournisseurs des services dans l'environnement mobile devra se rattraper avec ces normes, mais aussi d'intégrer la conscience d'emplacement et de situation d'une manière cohérente. Cette conscience doit être invisible pour l'utilisateur final ou d'être facilement accessible depuis l'interface utilisateur de l'appareil utilisateur final. Les fournisseurs de services doivent intégrer l'information concernant l'emplacement, prix, disponibilité, et le contenu des services en tant que critères de sélection dans l'infrastructure de service. Ces contraintes doivent aboutir à une demande de sensibilité de contexte et d'adaptabilité d'utilisateur des appareils mobiles.

I.4.4 Classification des services web mobile

Selon Gehlen [Gehlen 07], les services web mobiles, les classe en trois catégories. Le classement s'effectue par l'intermédiaire du demandeur service web et le degré de déploiement du fournisseur.

Le cas le plus simple d'un déploiement des services web mobiles est un demandeur mobile (client) et un fournisseur fixe (serveur), (voir (1) à la Figure I- 5). L'objectif principal de ce scénario, intitulé service web à accès mobile, est l'utilisation des services web existants fournis par Internet sur des dispositifs mobiles. Un dispositif mobile utilise par exemple un service web de traduction intégré avec une application de messagerie pour traduire les messages en différentes langues. L'enjeu de ce scénario est de trouver des protocoles qui permettent de réduire le délai de l'appel de procédure à distante (Remote Procedure Call (RPC)) sur le réseau mobile sans changer l'infrastructure des services web déjà existant.

En échangeant les rôles de demandeur et fournisseur du nœud fixe et mobile, le terminal mobile offre maintenant un service web et le nœud fixe demande ce dernier (voir (2) à la Figure I- 5).

Dans le dernier cas (voir (3) à la Figure I- 5), le demandeur et le fournisseur sont deux mobiles et communiquent les uns aux autres en pair-à-pair. Ce scénario est fortement lié à la classe de services (MSN, Skype, GoogleTalk) qui a fait tant de succès.

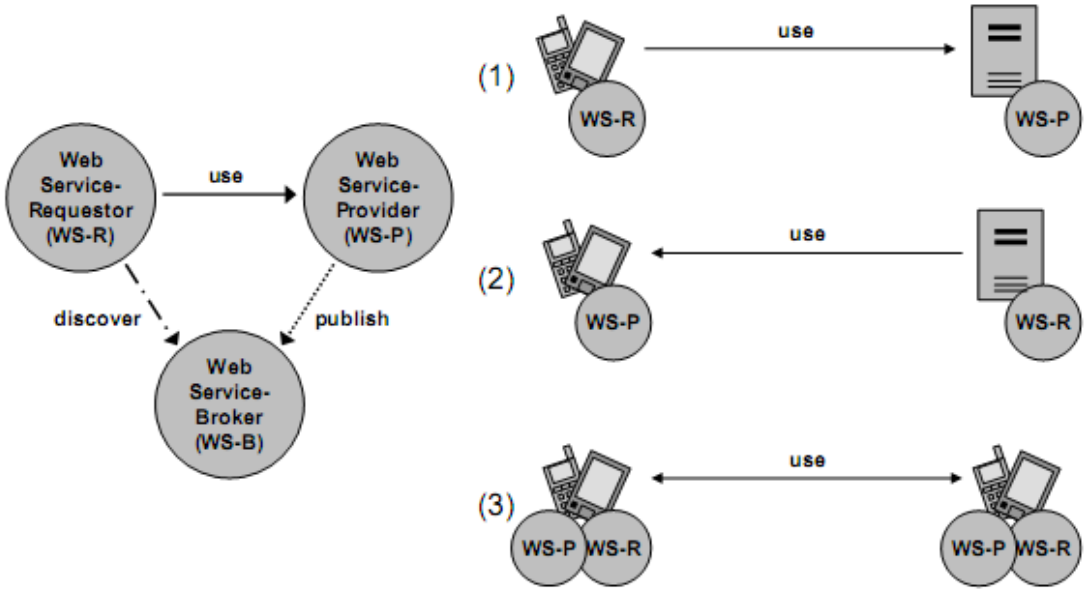

Figure I- 5: classification des services web mobiles [Gehlen 07].

I.4.5 Les appareils mobiles

Ces dernières années, les appareils mobiles sont devenus très populaires. Généralement un dispositif informatique est considéré comme un appareil mobile vu sa petitesse et sa légèreté au transport. Traditionnellement, cette contrainte pourrait être considérée comme l'unique contrainte, mais avec l'utilisation considérable des réseaux de téléphone cellulaire et réseaux locaux sans fil (WLAN[6]). Généralement, les appareils mobiles sont appeler à disposer d'une technologie de réseau sans fil. Par ailleurs, les appareils mobiles doivent garder leur mobilité au lieu d'être filaire. Notre optique s'oriente essentiellement sur les appareils mobiles avec un accès sans fil. Pour cette raison, les appareils mobiles en question seront supposés disposer de moyen adaptable aux réseaux sans fil et capable d'accéder à Internet.

Usuellement les ordinateurs portables modernes possèdent des connexions réseau sans fil. Actuellement on rencontre des ordinateurs portables légers et de petite taille. En revanche, les téléphones mobiles et assistants numériques personnels ont tendance à s'adapter au réseau sans fil qui fonctionnent à plus haute gammes et plus léger que les précédents. Les ordinateurs portables ont la taille d'un grand livre, par ailleurs les téléphones mobiles sont généralement plus petits qu'un livre de poche. Cependant, il se trouve que les ordinateurs portables, les

[6] WLAN: wireless local area network.

assistants numériques personnels et les téléphones mobiles ont des tailles variables, particulièrement si on fait la comparaison entre les vieux modèles à ceux en cours. Ainsi, les téléphones, les assistants numériques personnels disposent d'une durée de charge plus importante que les ordinateurs portables. Pour cette raison, il est difficile de déterminer les exigences pour un appareil mobile, du fait que les tailles et les capacités sans fil des ordinateurs portables est en permanente amélioration, entre autre les capacités de traitement et de raffinement des logiciels de téléphones portables sont rendu meilleur.

I.4.5.1 Les technologies de réseaux

Un appareil mobile nécessite une interface adaptable au réseau sans fil pour pouvoir se connecter à des réseaux informatiques, notamment Internet. La première génération des technologies de télécommunication (1G) fut des systèmes analogiques où la voix était considérée comme le principal trafic. En termes de qualité globale des connexions, La première génération se révèle défavorable par rapport à ses successeurs, à savoir une faible capacité, de pauvres liens de voix et sécurité nulle, ce qui les rend (appels) sensibles aux écoutes indésirables [Bhalla 10].

La deuxième génération de technologie de téléphonie sans fil (2G) est incapable de transférer normalement les données, tels que la messagerie électronique, un logiciel autre que l'appel vocal numérique lui-même. Néanmoins, la messagerie SMS est également disponible sous forme de transmission de données pour certaines normes. La deuxième génération de réseaux cellulaires télécoms ont été commercialement lancée sur la norme GSM (Global System for mobile Communications) en Finlande en 1991. Le service GSM est utilisé par plus de 2 milliards de personnes à plus de 212 pays et territoires. L'omniprésence de la norme GSM permet aux abonnés d'utiliser leur téléphone dans de nombreuses parties du monde [Bhalla 10].

La génération intermédiaire (2.5G) est une transition entre la deuxième génération et la troisième génération des technologies sans fil cellulaires. La technologie la plus commune de cette génération est le GPRS (General Packet Radio Service). Le GPRS est une extension de la deuxième génération GSM. Il a la capacité d'utiliser tous les canaux de radio libre, à savoir ces canaux ne sont pas utilisés par la communication vocale à ce moment précise. La cellule du réseau peut transmettre des données avec une bande passante allant jusqu'à 57,6

kbps. En outre, comme le GPRS utilise uniquement les canaux disponibles pour les données, la bande passante peut chuter jusqu'à 9,6 kbps dans une cellule radio encombrée [Srirama 08].

Cependant, la technologie du réseau radio est en rapide expansion ces derniers temps à savoir, de nettes améliorations ont été enregistrées en matière de données aux réseaux cellulaires. Le plus important est le passage aux réseaux de troisième génération (3G). Le réseau de troisième génération se repose sur W-CDMA[7] l'utilisation d'une technologie de canal radio d'une fréquence plus élevée que le GSM. Les implémentations varient un peu partout dans le monde, la version européenne appelée UMTS[8].

Les réseaux cellulaires W-CDMA sont conçus pour offrir des bandes passantes de données par paquets arrivant jusqu'à 384 kbps, elle est due en partie à la fréquence plus élevée et aussi en raison de l'utilisation de canaux radio plus efficace que le W-CDMA fait. Néanmoins, deux inconvénients surgissent. L'une concerne la fréquence croissante qui diminue la plage de fonctionnement permettant ainsi une perte de paquets. L'autre, dont l'efficacité des algorithmes de W-CDMA ont une surcharge de traitement supérieure que les communications radio GSM, ce qui conduit aux batteries de s'user rapidement.

L'avènement des technologies de quatrième génération des services sans fil (4G) et leur déploiement dans les pays Sud Asiatique, suggère que les transmissions de données mobiles sur le taux de quelques Go (gigaoctet) est également possible. La quatrième génération de système de communications fournira une la solution IP bout-en-bout, où voix, données et flot multimédias peuvent être mis à la disposition des usagers sur une base de «n'importe quand, n'importe où » [Bhalla 10] [Srirama 08].

La cinquième génération de réseaux mobiles ou la cinquième génération de systèmes sans fil (5G) est une dénomination utilisée en certains documents de recherche et de projets visant la désignation de la prochaine phase majeure des standards de télécommunications mobiles, qui seront finalisés entre environ 2011 et 2013. L'utilisateur peut également brancher son téléphone cellulaire de la technologie cinquième génération avec son ordinateur portable afin d'accéder à Internet à haut débit. La technologie cinquième génération y compris la caméra, enregistrement MP3, lecteur vidéo, stockage élevé en mémoire du téléphone, lecteur audio etc… [Bhalla 10].

[7] W-CDMA : Wideband Code Division Multiple Access
[8] UMTS : Universal Mobile Télécommunications System.

I.4.6 Plates-formes supportant les services web mobile

Contrairement aux applications des ordinateurs de bureau, Les applications mobiles sont particulièrement limitées par l'environnement d'exécution de ces dispositifs. Constamment une application mobile ne peut fonctionner que sur certains modèles d'appareils mobiles. Les limitations proviennent de différents aspects: le système d'exploitation de l'appareil, le langage de programmation et les plates-formes utilisées au développement de l'application, les capacités de l'appareil et la taille de stockage de l'appareil. Toutefois, de nombreux facteurs doivent être considérés au développement des applications pour les téléphones mobiles. Plusieurs plates-formes logicielles comme le PersonnelJava, Symbian C++ etc existent pour le développement d'applications de téléphonie mobile [Srirama 08]. Ce paragraphe traite une partie de ces environnements de programmation.

I.4.6.1 Symbian OS C++

Symbian est un système d'exploitation dérivé du système d'exploitation Epoc. Epoc; développé par Psion à leurs ordinateurs de poche aux années quatre-vingt. Le système d'exploitation Symbian basé sur C++ fournit un système d'exploitation sûr et fiable aux dispositifs mobiles. Spécialement conçu pour les appareils mobiles, avec une faible consommation d'énergie et faible gourmandise en mémoire, Symbian fournit une plate-forme stable à l'industrie des télécommunications et des technologies telles que GPRS, Bluetooth, SyncML, finalement la troisième génération. Symbian OS est non seulement un système d'exploitation, mais aussi un logiciel complet et une plate-forme de communication. Le système d'exploitation est livré à des appareils réels dans différentes saveurs. Plus précisément, Symbian Inc élabore un système d'exploitation de base et des licences aux fabricants de téléphone. Cependant, les vendeurs, peuvent construire une interface utilisateur basée sur le système d'exploitation mentionné.

I.4.6.2 PersonalJava

PersonalJava appelé aussi «pJava», est un environnement de programmation Java s'adressant au développement d'applications aux appareils à ressources restreintes tels que: les téléphones intelligents, assistants numériques personnels,…etc. C'était la première tentative

de Sun pour produire un environnement d'applications Java (JVM[9]) aux dispositifs mobiles. PersonalJava spécifie un ensemble de bibliothèques de classes réduites par rapport à l'environnement Java de l'ordinateur de bureau.

Le profil PersonalJava se base sur le JDK1.1 (Java Development Kit), mais offre un certain nombre de packages, classes et méthodes facultatives. Il laisse une possibilité au développeur de créer des applets web et autres applications de téléphonie mobile.

I.4.6.3 Java ME

Précédemment connu sous Java 2 Platform Micro Edition (J2ME), Plateforme Java™ Micro Edition (Java ME) est un sous-ensemble de Java 2 Platform Standard Edition (J2SE). C'est une plate-forme d'application Java la plus omniprésente pour les appareils mobiles. La plate-forme Java ME comprend des interfaces utilisateur flexible, un modèle de sécurité robuste, une large gamme des protocoles réseau intégrés, un support étendu pour le réseau et des applications hors ligne. Ces dernières peuvent être téléchargées dynamiquement. Java ME est définie par Java Community Process, et maintient également la philosophie de portabilité de Java. Java ME a obtenu un succès, ainsi la majeure partie des constructeurs intègrent Java ME sur une partie de leurs téléphones.

Initialement la technologie Java ME fut créé pour faire face aux contraintes liées à la construction de l'application Java aux petit appareil d'une capacité limitée de mémoire, d'affichage et de puissance. La plate-forme Java ME est un ensemble de technologies et de spécifications qui peuvent être combinées pour construire un environnement d'exécution Java complet spécifiquement pour répondre aux exigences d'un dispositif et un marché particulier. La technologie Java ME se base sur trois éléments: une configuration offrant un ensemble fondamental de bibliothèques et de capacités aux machine virtuelle à large gamme de dispositifs, un profil comme un ensemble d'API soutenant une gamme restreinte d'appareils et un package optionnel comme ensemble de technologie API spécifiques.

I.4.6.4 .NET Compact Framework

Microsoft .NET Framework, composant logiciel ajouté au système d'exploitation Microsoft Windows. Le .NET Compact Framework est un sous-ensemble de .NET

[9] JVM **:** Java Virtual Machine.

Framework et fournissant un environnement robuste aux développements des applications mobiles. Il gère l'exécution des programmes écrits spécifiquement au framework. Le framework est destiné à l'utilisation des applications créées à la plate-forme Windows. Le code géré .NET Compact Framework et les services web permettent un développement d'applications sécurisées, téléchargeables sur les appareils tels que: assistants numériques personnels et téléphones mobiles. Le framework utilise une partie de classes de bibliothèques que le.NET Framework complet et une partie de bibliothèques conçues spécifiquement pour les appareils mobiles. Le .NET Compact Framework a plus de succès dans le marché des assistants numériques personnels, où Windows bénéficie d'un certain succès, plutôt qu'aux téléphones intelligents.

I.4.7 Conclusion

Notre étude nous a permis de passer en revue les technologies de service web et services web sémantique. Ce dernier se positionne où convergent les deux importants domaines de recherche concernant les technologies de l'Internet dont: le web sémantique et les services web. Les services web sémantiques sont une évolution nécessaire aux multiples services web. Par la même, nous dressons un tableau succinct de la technologie des services web mobile. Cette dernière sera la prochaine vague des services web électroniques, et permettra aux utilisateurs d'accéder aux services n'importe où et n'importe quand.

Dans le prochain chapitre, nous présenterons l'état de l'art des travaux à base d'agent et agent mobile pour les services web mobile.

Chapitre II : Travaux connexes

II.1 Introduction

Dès l'émergence de la technologie des services web, celle-ci ne cesse de représenter une importante thématique de recherche. En effet, suite à sa découverte un service est mis en fonction. Une multitude de techniques et d'approches sont utilisés pour aboutir à la découverte d'un service web. Dans notre cas et plus précisément dans cette étude nous penchons vers la découverte des services web à l'environnement mobile. Ainsi, nous classons les travaux de recherche en deux parties: la découverte des services web sémantique s'abstenant à l'utilisation des agents mobiles et la découverte des services web utilisant l'agent mobile.

Dans ce chapitre, suite à un bref exposé en matière de technologie d'agent, nous entamons la présentation des travaux récents sur la découverte des services web dans l'environnement. A la fin de ce chapitre, nous mettons en relief l'analyse de ces travaux.

II.2 La technologie d'agent

De nous jour, nous avons une multitude de définition concernant l'agent. Pour notre cas nous retiendrons trois définitions qu'on juge répondant aux critères de l'étude que nous menons.

1. Un agent est une entité (logicielle ou matérielle) douée d'autonomie (notamment de décision), capable d'agir dans un environnement. Cette entité est assujettie à un objectif individuel, à une fonction à satisfaire ou à optimiser ; elle possède des ressources propres, capable de percevoir son environnement : cette perception est limitée. Pour atteindre son objectif, un agent met en œuvre un comportement basé sur le savoir-faire (ses compétences), ses perceptions, sa représentation du monde

et les communications qu'il possède avec les autres ou avec son environnement » [Mano 05].

2. Un agent est un système informatique situé dans un environnement, il agit d'une façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu [Jennings 99].
3. Un agent est une entité active autonome agissant par délégation pour le compte d'un client [Perret 97].

De ce que nous venons de voir naissent les quatre caractéristiques suivantes:

4. Délégation : l'agent est une entité agissant par délégation, respectant la stratégie de son producteur vis à vis des choix qu'il doit accomplir, afin qu'il soit responsable des tâches effectuées par son agent.
5. Autonome: l'agent est une entité autonome disposant de son propre environnement.
6. Connaissance : l'agent dispose d'une connaissance (parfois partielle), de son environnement courant. Ce qui lui permet de prendre des décisions appropriées.
7. Comportement flexible : il se caractérise par un comportement flexible (non rigide).

En littérature, il existe plusieurs types d'agents à savoir : des agents réactifs, des agents cognitifs et des agents hybrides [Hacini 08]. Ces derniers peuvent être soit stationnaires ou mobiles. Les agents stationnaires exécutent leurs tâches au niveau du site qui les a créés. Tandis que les agents mobiles peuvent se déplacer d'un site à un autre au cours de l'exécution pour accéder aux données ou aux ressources. Ils se déplacent avec leur code et leurs propres données, ainsi qu'avec leur état d'exécution [Lange 99].

II.2.1 Agent réactif

Ils se caractérisent par des comportements simples qui se résument à des réactions aux stimuli de l'environnement. La structure des agents purement réactifs tend à la simplicité, simplement, ces derniers peuvent être capables d'actions de groupe complexes et coordonnées [Tranier 07]. Les agents de ce type sont habituellement de petite taille et en grand nombre au sein de l'environnement. Ils ne sont pas nécessairement intelligents, néanmoins des

comportements collectifs intelligents peuvent émerger. La communication entre ces agents s'accomplie essentiellement par des traces ou des signaux.

L'exemple le plus classique de système d'agents réactifs est comparable aux colonies de fourmis [Corbara 93]. Les fourmis ont plutôt un comportement primitif et qu'il n'existe aucune relation d'autorité entre elles, cependant, leurs actions sont coordonnées de telle manière qu'il est possible pour la colonie de survivre et résoudre des problèmes complexes (la recherche de nourriture ou la construction de nids).

Additionnellement aux résolutions des problèmes complexes, les sociétés d'agents réactifs disposent aussi d'une capacité d'adaptation dynamiquement à leur environnement. Exemple, les fourmis sont capables de contourner un obstacle qui surgi sur la route empruntée entre la source nourriture et la fourmilière. Il est possible de créer à partir d'agents réactifs des systèmes robustes et adaptatifs. Les agents réactifs sont intéressants pas au niveau individuel, mais au niveau populations. La simplicité intrinsèque de leur comportement fait qu'ils ne présentent pas un grand intérêt individuellement, mais leur force vient du nombre et de leur capacité à faire émerger des organisations en formant des groupes.

II.2.2 Agent cognitif

Les agents cognitifs quant à eux disposent d'une mémoire. Aussi, ils possèdent une connaissance propre comprenant une représentation (partielle) de leur environnement, des autres agents, ainsi que de leur savoir-faire. Ce qui leur permet de gérer leurs interactions avec l'environnement et les autres agents [Tranier 07].

La distinction principale les agents cognitifs des agents réactifs est leur faculté d'anticipation. La capacité de raisonner sur des représentations du monde leur permet de mémoriser les situations, les analyser, prévoir les changements induits par leurs actions et finalement de décider du comportement à adopter ultérieurement. Un point crucial dans la conception d'agents cognitifs est la planification de leurs actions.

II.2.3 Agent hybride

Au préalable nous avons mis à jour la distinction entre les agents de type réactif, qui réagissent uniquement aux stimuli de l'environnement, et les agents de type cognitif, qui possèdent une représentation symbolique de l'environnement qu'ils mettent à jour

continuellement et qui leur permet de planifier leurs actions. Il est à noter qu'il existe une multitude de possibilités entre ces deux extrêmes pour ne citer comme exemple les agents qui exploitent des représentations numériques ou les représentations non symboliques [Tranier 07].

Aussi nous avons la possibilité de combiner les deux approches pour obtenir une architecture hybride. En cette architecture un agent est composé de modules qui gèrent indépendamment la partie réflexe (réactive) et la partie réfléchie (cognitive) du comportement de l'agent.

II.2.4 Différence entre agent et objet

Jadis en informatique, les seuls intervenants dans le développement du logiciel étaient le programmeur et l'opérateur du système. Le premier, du fait de la précarité des langages de programmation, détenait le contrôle de tous les états du programme qu'il a mis au point. Le second, du haut de sa console, inspectait toutes les requêtes auxquelles ce programme était censé répondre. Autrefois, l'unité atomique et indivisible d'un logiciel n'était autre que le programme lui-même dans sa totalité. Les notions récurrentes, si maîtrisées de nos jours, telles la modularité, la réutilisation et l'encapsulation n'existaient pas dans le vocabulaire des informaticiens [Kissoum 10].

Au long périple qu'a traversé l'informatique et à mi-chemin de notre époque, pour faire face à la complexité croissante des logiciels (très gourmands en espace mémoire) les programmeurs étaient contraints d'améliorer leurs applications. Ainsi, on a vu naître des portions de code réutilisables possédant un certain degré d'intégrité locale (naissance de la notion de sous-programme ou fonction). Dorénavant un programme est défini par un ensemble de fonctions pures (elle donne souvent le même résultat pour les mêmes arguments). La pureté est une propriété primordiale parce qu'un programme conforme à un moment donné, restera de même pour toujours. Cependant, le fait que l'on a tolérer le principe d'encapsulation du code à l'intérieur de ces fonctions, leurs états étaient en revanche déterminés par leurs paramètres en entrées. De plus, ces fonctions ne pouvaient détenir le contrôle qu'une fois sollicitées explicitement par une instruction d'appel extérieur.

L'approche objet étendra le paradigme procédural avec les notions d'abstraction de données, d'état explicite, de polymorphisme et d'héritage. L'abstraction des données permet le

partitionnement des programmes en plusieurs parties, qui pouvait être développées indépendamment. L'état explicite ajoute une mémoire aux programmes, ce qui introduit une dimension temporelle. L'utilisation favorable de ce paradigme se situe au niveau de la modularité: une abstraction peut être changée sans modifier le reste du programme. Le polymorphisme permet de structurer les programmes selon les responsabilités: une tâche particulière peut être traitée complètement par une seule abstraction, qui en est responsable. Finalement, l'héritage permet l'isolation des parties communes des abstractions en un seul endroit du programme. En dépit de ces innovations, les objets sont toujours considérés comme totalement passifs et n'ayant aucun contrôle sur l'exécution des méthodes invoquées par des entités externes.

La programmation en tant qu'agent se démarque parfaitement des précédents paradigmes de programmation. En effet, la notion de localité ne concerne plus seulement le code et l'état des variables, mais également les invocations des méthodes. En d'autres termes, le moment et la façon dont un agent réagit à un évènement externe (ou interne) est désormais déterminé par ce dernier. De plus, les agents possèdent des règles et des objectifs individuels les faisant apparaître comme des objets actifs, pouvant entreprendre des initiatives sans pour autant être explicitement sollicités pour cela. Ce sont les principaux points clefs du paradigme agent.

II.2.5 Système multi-agent

L'intelligence artificielle classique était un domaine de spécialistes en sciences (mathématiques ou la physique). En outre, ces chercheurs étaient focalisés sur le côté technique de la problématique [Aeken 99]. Le premier laboratoire spécialisé en intelligence artificielle était celui fondé par Marvin Minsky. A un certain niveau technique, la notion de réseau informatique était virtuellement inexistante : il n'existe que peu d'ordinateurs. De plus, le concept de computation distribuée était partiellement abandonné, bien que les réseaux neurones ont été toujours un sujet de recherche pour Minsky.

A la fin des années 70, des systèmes distribués sont apparus dans l'intelligence artificielle [Erman 80]. Les agents individuels étaient conçus comme des entités cognitives et leur but était la résolution distribuée de problèmes. En conséquence, une partie importante des recherches en ce domaine, connue sous l'appellation « Distributed Artificial Intelligence » (DAI), peut être classifiée comme « Distributed Problem Solving » (DPS).

Dès les années 80, ce domaine a connu un appréciable développement, depuis il n'a cessé de s'accroitre [Bond 88]. En ce sens, un nombre important de chercheurs a intégré le point de vue et les méthodes des courants plus récents, tel que l'école de Brooks. aux récentes recherches, les chercheurs utilisent non seulement des agents cognitifs, mais également réactifs. Ils font abstraction de l'architecture des agents et se concentrent sur les interactions entre agents. Au niveau des applications, ils prennent un point de vue différent : il n'est plus nécessaire que les systèmes résolvent des problèmes cognitifs, il suffit que leur comportement puisse être utile dans un contexte plus large, comme la modélisation ou la simulation.

Le système multi-agent est un ensemble organisé d'agents. Cela signifie que dans un système multi-agent, il existe une ou plusieurs organisations qui structurent les règles de cohabitation et de travail collectif entre agents sociaux (définition des différents rôles, partages de ressources, dépendances entre tâches, protocoles de coordination, de résolution de conflits, etc.). Ce type de système peut se révéler utile pour certains phénomènes complexes dont on ne connaît pas de modèle global [Aeken 99].

II.2.6 Agent mobile

Le concept de l'agent mobile a connu un intérêt croissant ces dernières années, ceci en parallèle à l'utilisation de l'internet et des réseaux de communication. La notion d'agent mobile invoque deux concepts: l'agent et la mobilité.

Le concept d'agent se réfère au domaine de l'Intelligence Artificielle (IA) alors que celui de la mobilité des agents repose sur les caractéristiques empruntées à la mobilité du code. Un agent mobile se déplace avec son code, ses données propres et son état d'exécution. L'agent décide lui-même de manière autonome de ses mouvements. En pratique, la mobilité ne se substitue pas aux capacités de communication des agents mais les complète. Afin de satisfaire les contraintes des réseaux de grande taille ou sans fil (latence, non permanence des liens de communication), les agents communiquent par messages asynchrones [Leriche 06] [Lange 99].

La communauté scientifique attribue à l'agent mobile les capacités suivantes :

- **Mobilité:** l'agent mobile est doté d'une capacité de *mobilité* lui permettant de se déplacer dans un réseau informatique.

- **Autonomie :** cette fonction lui donne le pouvoir agir seul, de prendre une décision en fonction des informations lui provenant de son environnement. Il gère lui-même son état interne en fonction des informations qui lui parviennent.
- **Réactivité :** cette propriété est primordiale pour un agent. Celui-ci reçoit des informations de son environnement et est capable de réagir en conséquence.
- **Comportement intentionnel (la pro-activité):** les agents sont dotés d'un comportement dirigé vers un but et de prendre des initiatives.
- **Efficacité :** elle est aussi importante que ses prédécesseurs. Elle a la capacité à résoudre le problème et atteindre les buts visés.
- **Adaptation** : cette fonction est dotée de pouvoir résoudre un certain nombre de problèmes. l'agent a la faculté de s'adapter aux variations de son environnement. Pour cela, il dispose d'un jeu d'actions possibles aussi de capteurs qui le renseignent sur son environnement et des variations des objectifs.
- ***Communication* :** Il doit pouvoir de se connecter avec d'autres agents de son environnement.

Notons que ces caractéristiques sont plus ou moins accentuées selon le domaine considéré, le but recherché et la stratégie adoptée. Toutefois, la mobilité est intrinsèque dans ce paradigme. Nous distinguons deux types de mobilité: la mobilité faible et la mobilité forte.

- **Mobilité faible (non transparente) :** l'agent doit préparer sa migration tout en sauvegardant son état dans des variables et une fois sa migration effectuée, il doit restaurer son état et reprendre son exécution, souvent à un nouveau point du programme.
- **Mobilité forte (transparente) :** après migration l'agent repart dans le même état et exactement au même point dans son code initial.

Exemple de modèle : les agents mobiles comprennent deux concepts fondamentaux; l'agent et son environnement appelé système d'agent ou encore plateforme d'agents. L'agent mobile est une entité détenant cinq attributs. Il les transporte lors de son déplacent à travers le réseau. Ces cinq attributs sont:

- *Etat :* L'état d'un agent peut être considéré comme une photo instantanée de son exécution. Il permet à l'agent de reprendre son exécution dès qu'il arrive à destination.
- *Implémentation :* l'agent mobile a besoin d'un code pour pouvoir s'exécuter.
- *Interface :* Un agent fournit une interface permettant aux autres agents et aux autres systèmes d'interagir avec lui.
- *ID :* Chaque agent possède un ID unique durant son cycle de vie, qui lui permet d'être identifié et localisé.
- *Autorité :* Une autorité est une entité dont l'identité peut être authentifiée par n'importe quel système auquel elle essaye d'accéder. Il existe principalement deux types d'autorités, le fabricant (manufacturer) qui est le fournisseur du code d'implémentation de l'agent, et le propriétaire (owner) qui a la responsabilité du comportement de l'agent.

II.2.6.1 Modes d'exécution

La conception d'applications distribuées fait intervenir trois principaux modèles qui étendent le paradigme client-serveur pour exploiter la mobilité du code: l'exécution à distance, le code à la demande et les agents mobiles [Carzaniga 97]. Ces modèles diffèrent selon l'emplacement des différents composants manipulés avant et après l'exécution d'un service. Ces composants sont:

- Le code (le savoir-faire)
- Les ressources nécessaires à l'exécution du code
- L'unité d'exécution (qui traite le code).

Les sous-sections qui suivent donnent un aperçu sur les différents modèles d'exécution [Hacini 08].

II.2.6.1.1 Le Client - Serveur

Le mode d'exécution Client – serveur est largement connu et son utilisation est assez bien maîtrisée. Un serveur offre un ensemble de services, de ressources et un savoir-faire nécessaire pour exécuter les services se trouvent sur un site A. Le client se trouvant sur un site B demande l'exécution d'un service par interaction avec le serveur A. Le serveur (A) exécute

le code correspondant au service demandé en utilisant les ressources nécessaires se trouvant sur le site B. Un client demande un service auprès du serveur, d'une façon interactive et synchrone, ceci signifie que le client est bloqué tant que le serveur ne répond pas.

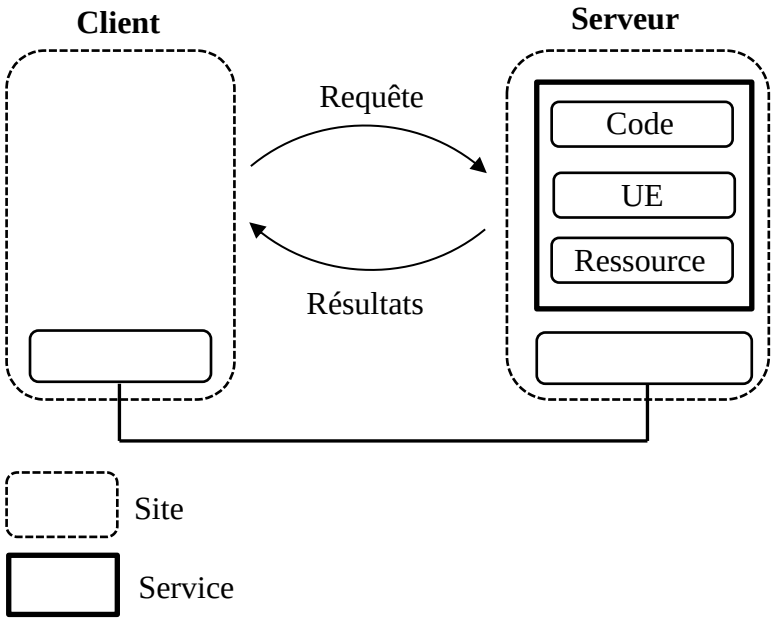

Figure II- 1: Schéma d'organisation Client – Serveur [Christophe 05].

II.2.6.1.2 Exécution à distance

L'exécution à distance est une méthode qui a la possibilité de réaliser des appels à distance grâce au mécanisme d'appel de procédure à distance ou RPC (Remote Procedure Call). Au début le RPC a été développé pour les systèmes Unix tout en ignorant la programmation objet. Le RPC a été récupéré par l'OMG (Object Management Group) avec CORBA auquel ils lui ont intégré l'approche objet. Ce concept est aussi utilisé par Java/RMI. Le service NFS (Network File System) est construit aussi sur ce mécanisme en permettant l'accès à des fichiers distants comme s'ils étaient sur disque local d'une machine.

L'exécution à distance est un mécanisme très utilisé pour la construction d'application répartie. Elle permet aux développeurs de ne pas manipuler directement le service de communication réseau mais d'appeler une procédure distante comme si elle était locale. Ainsi, un processus utilisant le RPC ne fait aucune différence entre un appel local et un appel distant. Le principe est: A détient le savoir-faire pour fournir un service mais ne détient pas les ressources nécessaires se trouvant sur un site B. A envoie le code à B détenant un

composant de calcul. Le code est exécuté au niveau de B en utilisant les ressources présentes localement. B renvoie le résultat à A par une autre interaction.

II.2.6.1.3 Le code à la demande

Dans le modèle code à la demande (COD : Code On Demand), le client dispose de l'unité d'exécution et des ressources mais pas du savoir-faire qui va être récupéré auprès du serveur. Ainsi, un client adresse une requête uniquement pour récupérer un code précis afin de l'exécuter localement avec les ressources présentes. Les trois éléments sont réunis sur le site client (Figure II- 2). L'exemple le plus répandu de l'utilisation de cette méthode est le téléchargement d'applets et Java à partir d'un serveur Web. Ces applets sont des classes Java présentes sur le site serveur qui sont téléchargées puis interprétées par la machine virtuelle du site client.

Le modèle code à la demande permet d'étendre les fonctionnalités d'une application directement chez le client sans avoir besoin d'effectuer une nouvelle installation. A l'aide de ce concept, le serveur peut rechercher les dernières versions d'un code à exécuter et le lier dynamiquement avec d'autres codes objets pour étendre une application. Il demande aux systèmes distants d'exécuter la tâche de collecte, et leur laisse le soin de récupérer le code correspondant à la tâche.

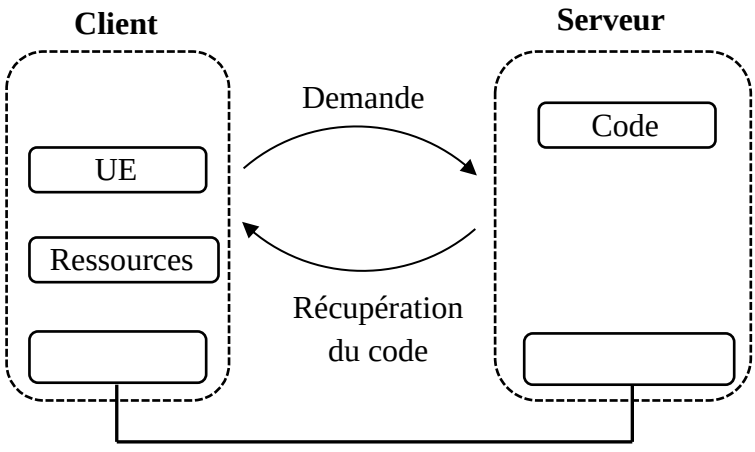

Figure II- 2: Code à la demande [Hacini 08]

II.2.6.1.4 L'agent mobile

Le concept d'agent mobile est différent aux autres modèles de conception utilisant le code mobile. On s'intéresse à l'exécution du code et à son déplacement d'un système à l'autre sous le contrôle d'une application. Pour ce paradigme, un site A dispose d'un agent mobile qui détient le savoir-faire alors que les ressources nécessaires se trouvent sur un autre site B. Cet agent mobile migre vers le site B en emportant avec lui le savoir-faire et éventuellement des résultats intermédiaires. Arrivé sur le site B, l'agent mobile poursuit son exécution en utilisant les ressources disponibles sur ce site. Ce schéma répond à nos besoins et représente le cas de figure envisagé dans cette thèse. Le savoir-faire est en effet transporté par un agent mobile qui le dirige vers plusieurs clients.

L'agent mobile est différent du paradigme de code mobile dans le sens où les interactions qui lui sont associées peuvent impliquer la mobilité de son état d'exécution. Par rapport donc aux notions précédentes, l'agent mobile offre une unité complète d'exécution (programme), et possède donc son propre contrôleur d'exécution.

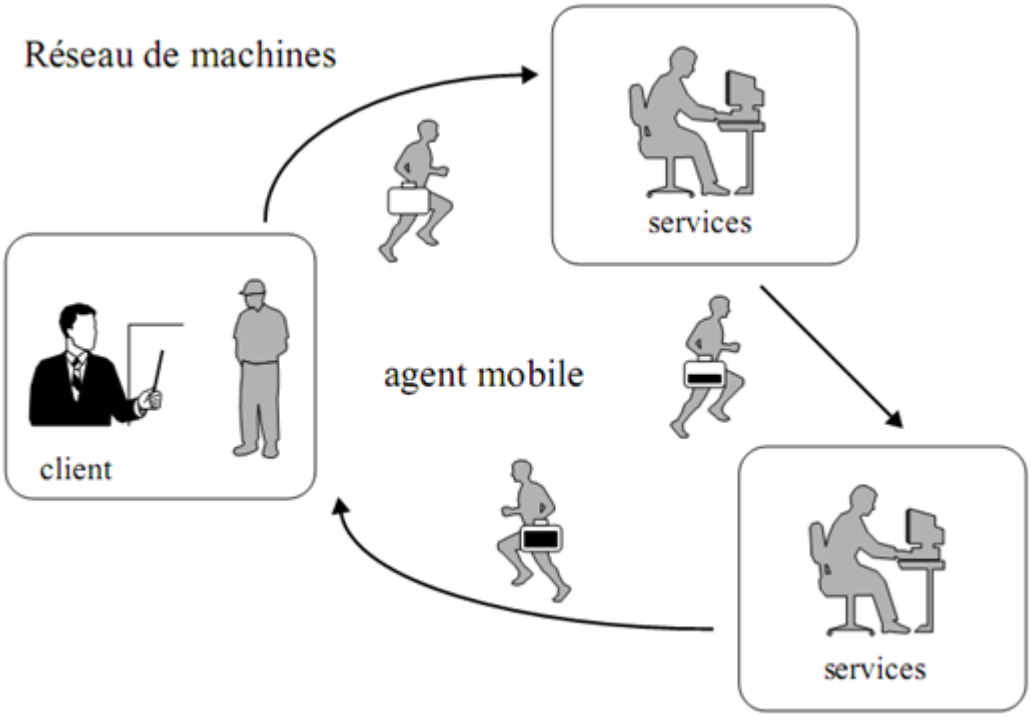

Figure II- 3: Le paradigme d'agent mobile [Perret 97].

II.2.6.2 Avantages et inconvénients des agents mobiles

La technologie des agents mobiles a fait naitre un intérêt particulier domaines de la recherche relative aux applications réparties. Cependant, cette technologie nouvelle a été évaluée afin de constater et estimer ce qu'elle pouvait apporter comme caractéristiques propres et ce qu'elle permettait d'améliorer par rapport aux méthodes de programmation classiques [Bernard 99] [Chess 94]. Dans cette section, nous allons mettre en relief les différents apports ainsi que les difficultés soulevées par les agents mobiles [Christophe 05].

II.2.6.2.1 La performance

En premier lieu, l'amélioration obtenue par les agents mobiles portent sur le gain de performances dues à une meilleure utilisation des ressources physiques mises à disposition. L'amélioration interviendra à différents niveaux tout en permettant d'optimiser la tâche globale des agents.

II.2.6.2.1.1 Diminution de l'utilisation du réseau

Le déplacement des agents mobiles permet la réduction significative des communications distantes entre clients et serveurs. En privilégiant les interactions locales, l'utilisation du réseau se limitera essentiellement au transfert des agents. Cette situation présente trois avantages à savoir:

- **Brèves périodes de communication :** on réduit considérablement le temps de connexion entre deux sites. Ceci par l'écourtement des communications distantes aux seuls transferts d'agents mobiles. Cette diminution des communications réseaux dégage une moindre inquiétude aux ruptures de liens physiques qui souvent peuvent surgir au sein des environnements sans fil [Christophe 05].
- **La diminution de la consommation de bande passante :** l'installation des agents mobiles permet l'obtention d'une réduction significative de la charge réseau en termes de nombre total de données transférées. Cette réduction est enregistrée dans différents types d'applications nécessitant d'intenses échanges d'informations entre client et serveur (la collecte d'informations dans des bases de données réparties, l'exploration d'un internet ou encore la gestion de réseaux) [Gray 02] [Sahai 98].

- **La diminution des temps de latence :** aux réseaux à large échelle, la mise en place d'applications reparties qui nécessitent des interactions fréquentes entre client et serveur, se trouve en opposition avec un temps de latence propres aux communications réseaux. Il arrive souvent de constater que le temps d'attente de la réponse d'une requête est plus long que le temps nécessaire au traitement de la réalisation du service. Il se trouve que si on rapproche client et serveur au un même site, on les place dans un environnement où les temps de réponse des interactions sont limités, ce qui nous permet de réduire le temps de latence [Johansen 98] [Jun 02].

II.2.6.2.1.2 Des calculs indépendants

Ainsi au modèle Client - Serveur, le client et le serveur restent connectés durant le temps d'exécution. En plus, on relève des problèmes liés aux fluctuations des performances réseaux. On remarque que certains services qui nécessitent de longues périodes de traitement, supportent difficilement une rupture de connexion avec le client. Afin de résoudre ce problème, ils (client-serveur) doivent souvent redémarrer entièrement leurs calculs. Mais, le maintien du lien de communication s'avére difficile aux réseaux sans fil. Avec les agents mobiles, un client peut déléguer les interactions avec le service sans entretenir une connexion de bout en bout. Avec cette possibilité de calcul indépendant, le client peut demander un service, se déplacer puis revenir pour récupérer finalement les résultats [Gray 01].

II.2.6.2.1.3 Optimisation du traitement

L'optimisation des phases de traitement se produit à deux niveaux [Ismael 99].

- En localisant les ressources et le savoir-faire sur un même site, on se débarrasse des phases de dialogue entre client et serveur qui sont perturbés par des temps de latence dus aux communications réseaux.
- Le déplacement du savoir-faire nous permet de déléguer les calculs sur des machines serveurs, qui sont souvent plus puissantes qu'une machine cliente.

II.2.6.2.1.4 Tolérance aux fautes physiques

Les agents mobiles se déplacent avec leur code et leurs données propres peuvent s'adapter aisément aux erreurs systèmes. Ces erreurs peuvent être d'ordre purement physique (disparition d'un nœud) ou d'ordre plus fonctionnel (arrêt d'un service). Si on prend le cas

d'un site qui perd une partie de ses fonctionnalités, un service tombant en panne, l'agent peut choisir le déplacement vers un autre site contenant la fonctionnalité voulue. Ceci permet une meilleure tolérance aux fautes [Rouvrais 02].

II.2.6.2.2 La conception

Les agents mobiles présentent à la fois une méthode qui permet de mieux définir certaines applications. Cependant, ils apportent aussi une complexité accrue par rapport au modèle classique client-serveur (bien mieux maîtrisé) [Christophe 05].

- Souvent, les concepteurs favorisent l'obtention d'une méthode permettant une description facile d'un comportement réel [Milojicic 99]. Cependant, avec la méthode classique, il est très contraignant de décrire des algorithmes d'exploration (de réseau) ou bien encore de caractériser les déplacements des utilisateurs nomades. A l'aide des agents mobiles, les concepteurs disposent d'une méthode qui permet de décrire naturellement ce genre de comportement. Ainsi, on peut aisément mettre en place un déploiement et/ou une maintenance d'application sur un réseau et de suivre les utilisateurs dans leurs déplacements [Picco 98] [Satoh 02].
- Les agents sont dotés d'une capacité de traitement spécifique leur permettant de s'adapter à leur environnement. Fréquemment, au modèle client-serveur et préalablement le service est caractérisé par une interface stricte, avec un protocole d'utilisation bien défini. Ainsi, si le client ignore la description du service, il ne pourra pas l'utiliser. Par contre, les agents pourront eux s'adapter aux caractéristiques des services afin d'exprimer leur requête. On cite comme exemple, si un service demande une communication sécurisée, l'agent pourra récupérer un module de sécurité, mettre à jour sa pile de communication et dialoguer avec le service [Yasuyoshi 99]. On peut aussi imaginer que l'agent pourra s'adapter aux conditions du réseau en se séparant d'une partie de ses fonctionnalités pour s'adapter aux terminaux pauvres en ressources [Libsie 02].

- La capacité de raisonnement va permettre la conception d'agents qui seront autonomes; ils adaptent leurs déplacements en fonction de l'environnement et pouvant moduler leurs fonctionnalités en cours d'exécution. Cependant ces propriétés s'accompagnent d'une conception bien plus difficile que celle du

modèle classique client-serveur [Vigna 04]. En effet, dans la majorité des applications distribuées, on essaie de cacher le plus possible la répartition en s'appuyant sur un ensemble de services système qui permettent de concevoir une application comme si tous ses éléments étaient locaux (CORBA) [Geib 99]. Afin de tirer pleinement partie des agents mobiles, la distribution doit être explicite et gérée par les agents eux-mêmes, ce qui complique la tâche des concepteurs. Cependant, si on se réfère à la conception objet où une application est définie comme un ensemble d'éléments et de relations, il est difficile de bien définir la tâche de chaque agent (élément) et surtout de savoir comment et où les interactions (relations) vont s'opérer.

- Enfin, la complexité même des services rendus par les applications impose souvent aux concepteurs l'utilisation de méthode parfaitement maîtrisée plutôt que de recourir à un mécanisme méconnu [Christophe 05].

II.2.6.2.3 Le développement

Lors des phases de développement, le domaine des agents mobiles rencontre de fortes contraintes à savoir:

- En première lieu, il existe actuellement un nombre élevé de plateforme pour agents mobiles qui possèdent chacun d'eux ses propres défauts et qualités [Johansen 04]. Ce qui soulève la difficulté de parler d'une standardisation. Sachant que les agents doivent s'adapter aux conditions de l'environnement. Les développeurs sont confrontés à un éventail de possibilités. Par manque de standardisation le défaut se trouve aussi dans les interactions entre agents, ainsi il n'existe pas de langage partagé par toutes les plateformes. Ceci représente un sérieux handicap. les agents ont besoin d'exprimer les caractéristiques des services qu'ils recherchent; afin d'obtenir des réponses précises sur leur localisation et la manière de les utiliser. Par la même, les développeurs sont en face à large éventail de possibilités du fait qu'il existe plusieurs langages. Pour résoudre ce problème, il faudra se tourner vers le domaine des systèmes multi-agents qui cherche à mettre en place des langages précis en s'appuyant sur un ensemble d'ontologies et de protocoles caractérisant les interactions inter-agents [Ametller 03].

- Le deuxième handicape, lors du développement c'est la complexité de la mise au point des programmes formant la partie critique du processus de développement. La plupart des environnements de programmation possèdent des outils permettant de suivre les différents éléments d'une application durant son exécution afin de dénicher les erreurs qui pouvaient surgir. Cependant, ce genre d'outil se manipule aisément avec des éléments statiques mais, se maitrise avec difficulté lors du déplacement des éléments. Ceci est particulièrement vrai dans les architectures hybrides, à grande échelle où il est quasiment impossible de contrôler la totalité d'un système donné. Ainsi, il s'avère impossible de récupérer l'état d'erreur d'un élément se localisant sur un site déconnecté. En l'absence outil de suppression des bogues, il est difficile de différencier les erreurs de conception et de développement. Il est à signaler que des efforts sont fournis afin de proposer un premier outil de suppression des bogues est conçu par la plate-forme JADE [Bellifemine 99].
- Le troisième et dernier hic, en rattachement avec le déboguage, on rencontre un handicape à mettre en place un contrôle des applications à base d'agents mobiles. La technique la plus usité en ce moment, malgré qu'elle n'est pas parfaitement fiable, est l'opération test. A cette méthode, on met à l'épreuve une application en lui injectant des données en ses différents points d'entrée dans le but de simuler le comportement utilisateur et de recouvrir une plage de situations la plus large possible. Si le test s'applique plutôt naturellement dans les programmes classiques, i.e non répartis, il représente un domaine de recherche à part entière dès qu'il s'agit d'applications réparties [Ulrich 99] [Benattou 02]. Dans cette optique, il faut maitriser la coordination des différentes injections sur les points d'entrée répartis. Ceci afin de garantir que le comportement simulé est bien celui qui était visé. Cette méthode est complexe dans le cas d'une application répartie statique. Si cette dernière est difficilement contrôlable que peut on dire si les points d'entrées (les agents) deviennent mobiles. Pour vérifier les applications construites sur des agents mobiles, on applique une méthode permettant la garantir du comportement et les interactions des agents avant leur déploiement. L'analyse statique de programme permet de déterminer statiquement, sans exécuter un programme, des propriétés qui seront satisfaites lors de l'exécution [Colaço 97]. Cette méthode, nous permet de garantir un certain niveau de correction du

programme analysé ; elle semble être préconisée afin de se passer des phases de test et, ce, même dans le cas des applications réparties [Kobayashi 95] [Pierce 93]. Il est à souligner que l'analyse statique a fait ses preuves dans le cadre de la programmation classique. Toutefois elle constitue un domaine de recherche pour les applications réparties.

II.2.6.2.4 La sécurité

Dans ce domaine, la sécurité constitue le talon d'Achille des agents mobiles. Du point de vue logiciel, la sécurité consiste à inhiber les accès ou les modifications, non-autorisés aux éléments d'un système informatique. Dans cette vision, nous somme amené à faire la distinction entre les demandeurs et les éléments demandés. Si on veut avoir un système sûr et sécurisé, il est nécessaire de mettre en place une politique en ce sens, garantissant la confidentialité(les données des éléments ne sont pas divulguées aux demandeurs non-autorisés) et l'intégrité(les éléments ne sont pas modifiés par des demandeurs non-autorisés). Nous omettons volontairement la disponibilité qui doit garantir, aux demandeurs autorisés, l'accessibilité aux éléments et qui est plus particulièrement du ressort de la tolérance aux fautes [Christophe 05].

La mise en place d'une politique de sécurité, demande généralement la mise en place de mécanismes à savoir:

- Authentification (mot de passe, certificat) : L'authentification a pour but d'identifier avec précision le demandeur,
- de cryptographie : la cryptographie doit assurer la confidentialité des données échangées,
- de contrôle d'accès (droit utilisateur, pare-feu) : le contrôle d'accès vérifie l'adéquation entre demandeur et éléments demandés.

Au schéma classique des applications réparties, on regroupe les éléments critiques sur des machines sécurisées et on les focalise sur les canaux de communication externes tout en utilisant les mécanismes d'authentification et de cryptographie. La mise en place de la mobilité de code, les politiques de sécurité recoure plutôt à la relation étroite entre le site stockant le programme et celui qui l'exécute. Implicitement, le possesseur du code est le même que celui de l'environnement qui l'exécute.

Pour ce qui est des agents mobiles et du point de vue sécurité, nous sommes devant un problème non encore résolu intégralement. C'est d'ailleurs le principal argument avancé pour expliquer la faible utilisation de ce paradigme. En effet, les agents mobiles représentent un nouveau champ d'investigation pour la recherche en matière de sécurité. On protège les sites vis-à-vis des agents malveillants, ceci d'une part, d'autre part on sécurise les agents vis-à-vis des sites malveillants. Nous intervenons sur deux points essentiels à savoir:

II.2.6.2.4.1 Protection des sites

La sécurisation des sites à l'encontre des attaques menées par les malveillants pose un problème, actuellement maîtrisé. En effet, plusieurs solutions permettent de déjouer d'éventuelles attaques. Les méthodes les plus connues sont [Christophe 05]:

1. **Bac à Sable** : La technique du bac à sable consiste à exécuter un agent à l'intérieur d'un environnement restreint, en interdisant l'accès au système de fichiers par exemple. Ainsi, un site peut exécuter un agent douteux dans le bac à sable sans trop se soucier des problèmes de sécurité. Cette approche peut être facilement mise en place en utilisant des interpréteurs de code dont leurs tâches sont limitées. Pour ne citer les applets Java exécutés à l'intérieur d'un navigateur Web.
2. **Signature de code** : La signature du code intervient lors de la création d'un agent, son créateur le signe numériquement afin qu'il puisse s'identifier durant ses déplacements. Cette technique permet d'obtenir une authentification de haut niveau pour les sites. Les applets Java utilisent désormais ce mode de fonctionnement. Ainsi si une applet est signée, elle est considérée comme un code de confiance et peut accéder à toutes les fonctionnalités de Java, à l'inverse elle sera placée dans un bac à sable.
3. **Contrôle d'accès :** l'amélioration des deux techniques précédentes, demande la mise obligatoire en place d'une politique de contrôle d'accès plus complexe. On peut la voir comme un raffinement d'une politique de bac à sable générale vers une politique spécifique à chaque application ou classe d'agents. En fonction des agents, le site pourra autoriser l'accès à un ensemble précis de fonctionnalités. Le contrôle d'accès permet de mixer les deux premières techniques en offrant aux

agents signés plus de fonctionnalité qu'un simple bac à sable sans pour autant accéder à toutes les fonctionnalités.

4. **Vérification du code** : La vérification de code permet d'acquérir une garantie sur la sémantique d'un code à travers l'analyse de sa structure, ou de son comportement pour un agent, en fonction d'une politique de sécurité donnée. Les bacs à sable réalisent des vérifications rudimentaires en cours d'exécution, ceci pour garantir avant tout le typage des opérandes des instructions, mais il s'avère très coûteux. On utilise une deuxième approche qui se base sur la vérification automatiquement de code avant son lancement, s'appuyant sur une preuve de conformité. Pour cela on peut utiliser l'approche PCC (Proof Carrying Code). Lors de la mise en fonction de l'agent, son créateur fournit un ensemble de preuves intégrées qui sont transportée par l'agent. Ces preuves garantissent le comportement de l'agent en fonction de critères de sécurité des sites à visiter. Lorsque l'agent commence une nouvelle visite, le site récupère la preuve lui correspondant et vérifie si elle correspond à sa politique de sécurité. Le site choisit alors d'exécuter ou non l'agent. Pour l'instant cette technique se base sur des propriétés de typage et la preuve est fournie par le compilateur.

II.2.6.2.4.2 Protection des agents

À l'inverse de la protection des sites, la protection des agents contre des sites malveillants ne dispose pas de solution éprouvée et demeure encore un champ ouvert à la recherche. Pour imaginer ce que risque un agent lors de son exécution sur un site malveillant, nous pouvons prendre comme référence les éléments transportés pouvant être prises comme cible d'attaque [Christophe 05] [Loureiro 01]:

- **Le Code:** ensemble d'instructions composant la tâche de l'agent.
- **Les données statiques:** données ne variant pas durant les déplacements (la signature par exemple)
- **Les données collectées:** ensemble de résultats obtenus au cours des déplacements réalisés par l'agent depuis son lancement.

- **L'état courant:** ensemble de données servant à l'exécution courante de l'agent.

La sécurité des agents mobiles consiste offrir la garantie des critères de confidentialité et d'intégrité de l'ensemble de ces éléments. Du points de vue données, il est évident qu'un agent ne désire pas divulguer les informations critiques à n'importe quel site. Par exemple, un site malveillant pourrait récupérer la signature d'un code et l'utiliser pour créer un nouvel agent afin de s'introduire dans des environnements auxquels ils lui sont interdits normalement. Pour le code, un agent transporte un savoir-faire propre à son concepteur qui pourrait tomber aux mains des intrus.

Dans [Christophe 05], l'auteur classe en trois importantes catégories d'attaques que les sites sont capables d'accomplir:

- **L'inspection :** elle consiste à l'examen du contenu de l'agent, ou le flot d'exécution afin de récupérer les informations critiques transportées par l'agent.
- **La modification :** elle s'accomplie par le remplacement de certains éléments de l'agent dans le but d'une éventuelle attaque. En remplaçant le code, l'agent effectuera des opérations malveillantes sur les futurs sites à visiter.
- **Le rejeu :** s'obtient en clonant l'agent puis en exécutant le clone dans plusieurs configurations pour retrouver le savoir de l'agent.

Plusieurs techniques sont en cours d'étude pour garantir aux agents qui peuvent accéder en toute confiance à des nœuds ou pour détecter les agents intrus. Cependant, aucune d'elle ne peut fournir une sécurité suffisant comme celle proposée à la protection des sites.

Il est aussi important de signaler qu'en matière de problèmes de sécurité liés à l'utilisation des agents mobiles, celle-ci est encore un champ d'investigation complet. Bien que la protection des sites est à présent maîtrisée, la protection des agents est insatisfaisante à l'heure actuelle. Les efforts menés par la communauté sur la sécurité nous poussent à croire que nous obtiendrons un niveau de sécurité escompté.

II.2.6.3 Bilan des avantages et inconvénients

À travers les différentes caractéristiques des agents mobiles, nous pouvons dire que les agents mobiles ne représentent pas le paradigme idéal pour tous les types d'applications réparties, mais simplement une nouvelle possibilité à la construction des applications. Selon les besoins de performance, de conception, de développement et de sécurité les agents mobiles pourront apporter une alternative intéressante au modèle classique client-serveur dans certains types de configuration [Christophe 05].

Le critère de performance des agents sera très utile dès que les applications vont comporter des communications intensives. Par leurs interactions locales, les agents permettent de ne pas subir les ralentissements dus aux bandes passantes limitées et aux temps de latence des réseaux (surtout sur internet). En utilisant les machines serveurs plus performantes que les clients, les phases de traitement seront plus rapides. Du point de vue conception, les agents mobiles permettent de décrire les comportements difficiles à caractériser avec le modèle client-serveur. Ainsi, l'exploration d'un réseau, la représentation d'un utilisateur, le support des ruptures de communication ou encore l'adaptation à l'environnement sont naturellement exprimables grâce aux agents mobiles.

D'autre part, ces qualités font ressortir d'importantes difficultés de développement. Ces dernières sont dues essentiellement au manque de standardisation des plateformes qui n'offrent pas l'obtention d'un langage homogène partagé par tous les agents. Ajoutons à cela, les difficultés de mise au point dues aux déplacements de l'unité d'exécution qui sont difficiles à suivre et aux problèmes de test qui nécessitent d'importants efforts de coordination. Ces difficultés de développement poussent les concepteurs à s'orienter vers des paradigmes bien mieux maîtrisables en mettant en place des mécanismes de mobilité plus limités. D'un point de vue plus informel, la programmation classique dispose d'outils graphiques d'aide au développement qui intègrent complètement les phases de programmation, de déboguage et de test. Quelques outils du même type apparaissent pour les agents mobiles mais ne sont pas totalement intégrés en restant pour la plupart focalisés sur la phase de programmation et seuls une minorité s'intéresse à la phase de déboguage.

Finalement, nous pouvons conclure que le problème majeur empêchant l'adoption actuelle des agents mobiles est la sécurité. En effet, même si la protection des sites est quasiment assurée, celle des agents reste un réel problème non résolu définitive. Différentes

études sont actuellement menées pour permettre l'obtention d'un niveau de sécurité satisfaisant.

II.2.6.4 Domaines d'application

Dans cette partie, nous présentons quelques domaines d'applications envisageables pour les agents mobiles [Christophe 05].

II.2.6.4.1 Maintenance répartie

La maintenance répartie c'est la mise à jour des ressources (services ou matérielles) réparties au sein d'un réseau. Ces mises à jour consistent à remplacer, ou à ajouter, des fonctionnalités à un ensemble d'éléments (actualiser un routeur avec la nouvelle version d'un protocole).

De manière classique, la méthode de mise à jour de ressources distribuées peut s'effectuer de deux manières:

- soit c'est la ressource elle-même qui contacte un serveur de mise à jour,
- soit c'est un service de mise à jour (ou un administrateur) qui contacte toutes les ressources à actualiser.

Dans les deux cas, nous sommes face à une vision centralisée où il est fait référence à un gestionnaire principal. Ces méthodes seront difficiles à adapter dans des réseaux non-structurés comme ceux construits sur le modèle ad hoc.

L'utilisation des agents mobiles permet de décrire simplement l'exploration d'un réseau et permet d'envisager une mise à jour totalement découplée d'un service central en déléguant une partie de l'administration aux agents mobiles. Dans ce cas, un agent possède la mise à jour à appliquer et va se déplacer de site en site et va actualiser tous les éléments trouvés. En se déplaçant, les agents peuvent plus facilement accéder aux éléments appartenant aux infrastructures décentralisées et surtout peuvent appliquer la mise à jour sans craindre d'être interrompus, cela grâce à leur traitement local [Marques 01].

II.2.6.4.2 Découverte de contexte

Les architectures sans fil peuvent s'appliquer dans des environnements ne disposant pas d'infrastructure fixe de référence. A titre d'exemple, lors d'une catastrophe naturelle les systèmes de communication pouvant être réduit à néant, il est intéressant de disposer rapidement de moyens pour coordonner les secours. Dans de tels domaines, où l'on ne dispose pas de gestionnaire centralisé, il est primordial de récupérer n'importe quelles informations permettant de caractériser la situation des éléments présents dans l'environnement immédiat, encore appelé contexte. La découverte de contexte se définit alors comme l'utilisation du contexte afin de donner des informations significatives aux applications ou des services aux utilisateurs [Abowd 99].

Dans la découverte de contexte, les caractéristiques des agents mobiles peuvent être mises à profil [Zaslavsky 04]. L'exploration de l'environnement est facilitée par la mobilité des agents qui peuvent se charger de trouver toutes les informations utiles présentes sur les sites voisins. De plus, leur capacité d'adaptation leur permet d'interpréter le contexte en fonction de besoins spécifiques et de réagir en fonction d'événements perçus dans le contexte : par exemple, récupérer certains services bien précis ou capter le déplacement d'une unité mobile.

Une application envisageable de la découverte de contexte est la localisation virtuelle des utilisateurs [Satoh 04]. En déterminant quels sont les éléments les plus proches d'un utilisateur nomade, on peut définir sa localisation et ainsi lui proposer des services adaptés. Par exemple, si un employé se déplace dans les différents étages de son entreprise, il pourra imprimer sur l'imprimante la plus proche.

II.2.6.4.3 Grille de calcul

La grille de calcul consiste à regrouper dynamiquement au sein d'un même réseau virtuel tout un ensemble de machines, de ressources et d'utilisateurs [Baker 02]. L'objectif des grilles de calcul est de permettre à des organisations dispersées (entreprises, universités, etc.) de partager des applications (services), des données et des ressources (processus, disque). Le but est de mettre en commun les capacités de chaque entité. Ainsi, on peut très bien imaginer qu'une entreprise, spécialisée dans les calculs à haute performance, loue du temps de

calcul sur ses serveurs à une entreprise souhaitant mettre en place une expertise précise (service). Nous sommes donc face à un environnement hétérogène, décentralisé et dynamique.

Avec les grilles de calcul, les agents mobiles trouvent un champ d'application naturel. En effet, les caractéristiques de déplacement, d'adaptation, de coopération vont permettre de tirer partie de ces environnements. On peut voir une grille comme un marché ouvert où des marchands (organisations) proposent un ensemble de produits (services) aux clients (utilisateurs) présents dans l'environnement. Les clients explorent le marché pour trouver les meilleurs produits en fonction de leurs besoins. Les agents mobiles vont pouvoir adopter ce comportement naturel de négociation. Si un client cherche à utiliser un service en fonction d'un certain nombre de critères, il crée un agent qui va aller de serveur en serveur afin de trouver le service correspondant le mieux aux critères définis par l'utilisateur. Une fois qu'il l'a trouvé, il réalise le service visé et ramène les résultats à l'utilisateur.

Les agents mobiles ont la capacité de représenter un utilisateur déconnecté. Dans les grilles de calcul, les organisations offrent des expertises de haut niveau nécessitant de longue phases de traitement et la mise en œuvre d'un ensemble de sous-services. Avec un utilisateur nomade, se connectant de manière intermittente à la grille, l'utilisation des agents mobiles permet de déléguer la réalisation de la tâche que l'utilisateur souhaite exécuter. L'autonomie de l'agent permet à l'utilisateur de sortir de la grille durant le temps de la réalisation de la tâche. Lorsque l'utilisateur se connectera et que l'agent aura achevé sa tâche, il pourra récupérer les résultats obtenus par l'agent.

Enfin, un des problèmes majeurs des grilles de calcul vient de leur construction à l'échelle planétaire. Avec une telle proportion, les communications distantes sont confrontées aux problèmes de faible taux de transferts et de fort temps de latence qui ne permettent pas d'engager des services nécessitant des communications intenses avec le client. Dans ce cas, les agents mobiles, par leur représentation locale du client, permettent de pallier à ces problèmes et offrent le niveau d'interaction nécessaire à la réalisation du service [Christophe 05].

II.2.6.4.4 Application orientée client mobile

Actuellement et avec la chute des prix du matériel de transmission sans fil, avec l'augmentation de l'autonomie des batteries et enfin avec l'augmentation des performances de

la miniaturisation des processeurs de forte puissance, nous permet d'avoir des unités mobiles de petite taille (PDA ou téléphones intelligents) capables de se déplacer facilement avec leur propriétaire tout en exécutant des applications complexes et en ayant la capacité de rester connectées à un réseau sans fil [Christophe 05].

Avec ces unités mobiles, les utilisateurs mobiles vont vouloir accéder à leurs applications et données favorites quels que soient les environnements les entourant. D'un point de vue matériel, on peut atteindre ce besoin d'ubiquité. Par contre, d'un point de vue logiciel, il va être difficile démettre en place une architecture type client/serveur pour suivre le déplacement des utilisateurs. En effet, les réseaux sans fil sont moins performants que les réseaux filaires et sont surtout assujettis à une faible fiabilité des liens de communication. Ces deux problèmes peuvent être contournés en utilisant les agents mobiles qui limitent l'utilisation des réseaux en interagissant localement avec les services visés [Vasiu 04].

Lors du déplacement d'un utilisateur, il va se retrouver au sein d'environnements aux caractéristiques diverses et variées (soit au niveau de la performance du réseau ou au niveau de la multitude de services). La capacité d'adaptation des agents mobiles va leur permettre de s'accommoder de cette diversité en fonction des besoins des utilisateurs[Campadello 00]. Par exemple, lorsqu'un utilisateur mobile souhaite afficher ses photos, l'agent pourra prétraiter une image haute définition pour l'adapter à la résolution de l'écran de son unité mobile. D'un côté système, l'hétérogénéité des unités mobiles, allant d'ordinateur portable jusqu'au téléphone, et celles des moyens de communication, allant du Wifi au GPRS, va demander d'adapter les services en fonction des contraintes matérielles en accord avec les critères définis par le concepteur. Ainsi, l'agent pourra s'adapter aux performances du terminal mobile en supprimant certaines fonctionnalités.

Durant ses déplacements, un utilisateur mobile peut perdre temporairement sa connexion avec un réseau (par exemple en arrivant dans une zone non-couverte). Dans ce cas, la possibilité des agents mobiles de représenter un utilisateur déconnecté permet de continuer les services engagés malgré la perte de connexion [Tomarchio 00]. On retrouve ici l'exemple d'utilisation de la grille où un utilisateur lance une simulation et vient récupérer plus tard ses résultats bien qu'il se soit déplacé entre temps.

II.2.6.5 Environnement d'exécution des agents mobiles

L'environnement d'exécution des agents mobiles est une infrastructure offrant un certain nombre de services de contrôle et de qualité pour assurer l'exécution des agents mobiles. Cette infrastructure permet :

- la création d'un nouvel agent,
- l'enregistrement et la réception des agents mobiles entrants, l'activation du code de l'agent,
- la migration des agents vers d'autres systèmes,
- la communication distante ou locale entre les agents,
- l'accès aux ressources/services locaux du système, la localisation des agents mobiles, leur sécurité/

Actuellement, plusieurs plateformes de développement d'agents mobiles ont et commercialisées. L'ensemble des services offerts varie d'une plateforme à une autre. Cette variété est due à plusieurs facteurs, notamment le domaine d'application et le modèle d'agent adopté (réactif ou proactif).

II.2.6.5.1 Aglets

La technologie Aglets et son environnement de développement ASDK (Aglets Software Development Kit) sont actuellement maintenus au travers d'un site communautaire, selon un modèle libre et open source [Jones 02]. L'ASDK permet de programmer rapidement et simplement des agents logiciels mobiles en Java. Le modèle d'agent est simple, associant à un objet java un thread et des méthodes d'activation spécifiques. Par exemple, il est possible d'exécuter du code après une opération de mobilité en implémentant la méthode onArrival(...). L'exécution est supportée par un serveur d'Aglets nommé Tahiti qui permet de gérer les communications, la sécurité et la mobilité des agents.

II.2.6.5.2 Jade

JADE est une plateforme construit sur Java afin de permettre une programmation multi-agents simplifiée. C'est une plateforme très abouti qui offre un environnement graphique pour l'aide au développement et à l'administration. Le container est construit comme une agence contenant une unique place exécutant une machine virtuelle Java. Une

région est alors définie par un ensemble de containers distribués correspondant concrètement à la plate-forme JADE. Il existe aussi une possibilité d'associer plusieurs plates-formes en reliant leur Main-container mais en limitant leurs fonctionnalités. En effet, les régions définies par les Main-container autorisent uniquement l'échange de messages mais pas la migration des agents. Tous ces éléments sont gérés par des agents qui représentent l'entité de base de la plate-forme. Les agents peuvent migrer au sein d'un même ensemble de containers. Cette migration, qui peut être proactive ou réactive, est réalisée par le main-container en se limitant aux containers sous sa responsabilité [Bellifemine 01].

II.2.6.5.3 TACOMA

TACOMA se propose de savoir comment les agents peuvent être employés pour résoudre des problèmes traditionnellement soulevés par d'autres paradigmes, comme le modèle client/serveur. Le projet de TACOMA a été développé dans le but d'offrir différentes abstractions de haut niveau pour les agents. L'accent est mis sur un découplage entre le langage et le niveau agent, semblable à celui introduit dans les normes à travers la notion de place [Johansen 02].

II.2.6.5.4 PLANGENT

La plateforme PLANGENT est basée sur le langage Java. Elle s'intéressant à l'adaptation des agents durant leurs déplacements au sein des environnements dynamiques. Le but de la plateforme PLANGENT est de proposer aux agents des mécanismes pour modifier les objectifs intermédiaires nécessaires à la réalisation de leur tâche finale. Cette plateforme est très ancienne et il existe très peu de documentation [Ohsuga 97].

II.2.6.5.5 Grasshopper

La plateforme Grasshopper est permet la création d'applications multi-agents en fournissant un environnement d'exécution approprié pour eux. Cette plateforme est mise en oeuvre en Java et supporte mobilité faible ainsi que les invocations distantes de manière transparente pour le programmeur. Plusieurs protocoles de communication entre les agents sont supportés (TCP/IP, Java RMI, CORBA IIOP, MAF IIOP, RMI SSL). De même, Grasshopper supporte des mécanismes de sécurité pour les communications basées sur le protocole SSL [Bäumer 99].

II.3 La découverte de M-service sans l'utilisation d'agent mobile

Ben Mokhtar et al [Ben Mokhtar 07] présentent une approche de découverte des services web dans les environnements mobiles nommée EASY (Efficient semAntic Service discoverY). De plus, les auteurs ont présentés :

- EASY-L : un langage pour la description de service sémantique; couvrant à la fois les caractéristiques fonctionnelles et non fonctionnelles des services,
- EASY-M : définit un ensemble de relations de conformité et prescrit la manière de les appliquer afin d'accomplir l'appariement des services, codage numériquement des ontologies. Ils transforment le raisonnement sémantique coûteux en comparaison numérique simple des codes. Ils présentent un algorithme pour organiser les services dans l'annuaire afin de réduire le nombre de filtrages effectués pour résoudre une demande de service.

Dans [Toninelli 08], les auteurs mettent en valeur un mécanisme de découverte de services personnalisé, basé sur la sémantique AIDAS (Adaptable Intelligent Discovery of context-Aware Services). Ce dernier exploite des techniques sémantiques pour réaliser la découverte de services sensible au contexte. Ceci est en fonction des besoins et préférences exprimées par les utilisateurs mobiles. L'objectif de AIDAS est d'intégrer une représentation sémantique des données contextuelles et l'analyse des correspondances au mécanisme de sélection.

Suarez et al [Suarez 11] présent une approche de découverte de services web sensible au contexte. Celle-ci transforme les descriptions BPEL de services et la demande de l'utilisateur en graphes et effectue la mise en correspondance. Ainsi, ils calculent la similarité sémantique entre deux nœuds du graphe afin de classer les services. Lorsqu'i l n'y a pas de service qui correspond exactement aux besoins des utilisateurs, l'approche donne une mise en correspondance approximative.

Sesseler et al proposent la technologie des agents pour la mise en œuvre des services mobiles tout en réduisant la complexité des infrastructures de services mobiles [Sesseler 02]. Par la suite, vient Sheu adaptant une architecture de service omniprésente ad-hoc à multi-agents [Sheu 09]. Son but est de fournir des services omniprésents à la demande pour répondre au besoin conjoncturel et dynamique de l'utilisateur.

Pour découvrir les services web performants, Steller et al optimisent le processus de raisonnement [Steller 09]. Cette approche présente l'appariement sémantique dans l'appareil mobile en développant un algorithme nommée l'algorithme mTableaux. Ce dernier optimise le processus de raisonnement par la même il facilite la sélection de service et réduit le temps de traitement.

En utilisant les ontologies et les technologies du web sémantique, Razieh et Qusay proposent une architecture pour la découverte des services web mobiles [Niazi 09]. De plus, l'architecture est basée sur le profile. Les chercheurs prennent en compte les désires de l'utilisateur, les capacités des appareils ainsi que les propriétés de qualité des services pour découvrir des services web mobiles les plus pertinents.

Peng et al. éditent une nouvelle étape dans le mécanisme de découverte. Ils effectuent l'appariement sémantique entre le demandeur et le fournisseur de service. Ceux-ci se basent sur OWL-S. Ils ajoutent au profil de service les informations de profil d'utilisateur. La technique proposée est appliquée sur un cas de test simple, les résultats expérimentaux obtenus sont positifs et efficaces [Peng 08].

Younas et soraya présentent un nouveau modèle de transactions sensibles au contexte dans les services mobiles [Younas 11]. L'objectif est la gestion des transactions de manière qu'ils respectent le contexte nécessaire des services souhaités. De plus, elles s'adaptent aux conditions environnementales ainsi qu'aux besoins des utilisateurs. De plus, les auteurs présentent trois contributions qui sont:

- le développement d'un nouveau modèle pour les transactions sensibles au contexte dans les environnements de services mobiles,
- concevoir et développer un protocole pour l'exécution d'opérations sensibles au contexte,
- mettre en œuvre le modèle proposé et évaluer sa performance.

II.4 La découverte de M-service en utilisation l'agent mobile

Dans cette section, nous traitons les différents travaux de recherche intégrant les deux technologies, l'agent mobile et les services web mobiles.

Dans [Baousis 06], l'article expose une nouvelle architecture pour la découverte dynamique. De plus, les services web sont exprimés en OWL-S. Dans ce contexte, Baousis et al exploitent les capacités offertes par les agents mobiles pour interroger et exécuter sémantiquement les services web. En outre, l'agent mobile est équipé d'une intelligence approprié pour accomplir leur tâche de manière dynamique. Ketel décrit une architecture d'intégration des services web sémantiques avec les agents mobiles [Ketel 09]. Cette architecture permet l'intégration sans changer les spécifications existantes. De plus, il montre la façon de composer les simples services dans un workflow pour former un service composite complexe.

Dans [Marwa 12], elle propose une architecture d'intégration des agents mobiles avec les services sémantiques RESTful dans les environnements mobiles. L'architecture proposée résout le problème de la déconnexion fréquente.

L'information de contexte représente les propriétés non-fonctionnelles des services web. D'après Doulkeridis et Vazirgiannis, ils défini le contexte en : « information implicite liée à la fois au demandeur et fournisseur du service pouvant affecter l'utilité des résultats retournés" [Doulkeridis 08]. La découverte sensible au contexte peut être définie comme la capacité d'utiliser l'information de contexte. Celle-ci concoure à la découverte des services web les plus appropriées au client. Ainsi, deux types de contexte sont identifiés: le contexte d'utilisateur et le contexte de service web. Le contexte de l'utilisateur se compose de: le profil de l'utilisateur, les capacités des dispositifs, préférences de l'utilisateur, localisation, les contraintes temporelles et le temps. Ces derniers se caractérisent par la situation actuelle des utilisateurs. D'autre part, les propriétés de contexte du service web peuvent être l'identité du fournisseur, la localisation du service, le coût et la méthode de paiement.

Pinsdorf et al proposent une architecture pour la réalisation de services sensibles au contexte basée sur l'agent mobile [Pinsdorf 02]. En outre, ils ont utilisé la SeMoA (Secure Mobile Agent) plate-forme.

Khedr et al mettent en oeuvre une technique combinant la technologie des agents mobiles et les informations de contexte [Khedr 02]. Ils utilisent les ontologies pour la découverte des services et comme un langage commun utilisé pour la communication. De plus, ils ont décrit le modèle d'interaction entre les différents types d'agents dans l'environnement.

In [Bellavista 06], les auteurs décrivent une architecture à base de Gateway et agent mobile. De plus, l'architecture proposée est sensible aux contextes et à la localisation. En outre, Bellavista et al adoptent la technologie d'agent mobile afin de soutenir efficacement l'autonomie, la non synchronisation, et l'accès local aux ressources. De plus, l'agent mobile est particulièrement adapté pour les clients temporairement déconnectés. L'article présente également une étude de cas guide de musée.

Xining Li et al [Xining 10] ont présenté une architecture, qui intègre la technologie d'agent mobile et la sensibilité du contexte, pour les applications M-commerce. Ils utilisent les ontologies afin de fournir des informations contextuelles personnelles et de l'environnement. De plus, pour soutenir la composition de services sensibles au contexte.

Abedi et al proposent une architecture nommée AMF (Adaptation Management Framewok) [Abedi 12]. Cette architecture est destinée à la gestion du service de paiement mobile. En outre, ils utilisent la technologie d'agent pour un paiement automatique et intelligent.

Dans [Qusay et Leslie 06], les auteurs proposent une plateforme d'agent mobile basée sur la sensibilité à la localisation. Ce dernier a pour but la comparaison des achats dans l'environnement mobil. Dans le système proposé un gateway envoie un agent mobile. Ce dernier acte, accompli des travaux idem à ceux d'un agent négociateur. De plus, il cherche les sites représentant les magasins physiques situés à proximité de l'emplacement actuel de l'utilisateur. Une fois que l'agent a terminé son travail, il revient au gateway. En ce temps même, ce dernier envoie un message SMS à l'utilisateur de l'appareil mobile l'alertant que l'agent a terminé son travail et les résultats sont prêts à être consultés.

Chiro Satoh présente un système sensible au contexte basée sur l'agent mobile [Satoh 10]. Le système est appliqué dans les lieux publics pour ne citer: les musées. Aussi, l'auteur propose un service guide dans un musée. Quand un utilisateur est en face d'une pièce au musée. Le système détecte en ce moment même l'emplacement de l'utilisateur. Il envoi l'information à un ordinateur sise à proximité de l'utilisateur via l'agent mobile lui fournissant une annotation sur l'objet. L'utilisateur dispose d'un appareil mobile.

II.5 Analyse des travaux

Les environnements sans fils et les appareils mobiles ont différents facteurs de variabilité. Ces derniers sont:

- Déconnexion fréquente.
- Décharge la batterie.
- Bande passante étroite,
- Exiguïté de ressources en termes de mémoire, puissance de traitement, capacité de stockage et la taille d'écran.

Précédemment, nous avons fait l'étude des travaux de recherche sur la découverte des services web mobile sans recourir à l'agent mobile. Il s'est révélé que, malgré l'obtention de résultats non négligeable. Les travaux dans cette catégorie robustes aux différents facteurs de variabilité sont loin d'être atteints. Par exemple, considérant une personne se dirigeant vers l'aéroport. En chemin, la personne se sert de son téléphone portable pour effectuer une réservation de vol. Malheureusement, avant recevoir l'ok, une déconnexion s'est produite. Ce dernier est contraint d'attendre le rétablissement de la connexion pour envoyer à nouveau sa demande.

Cependant, les approches de découverte basés sur l'agent mobile est une alternative prometteuse à la résolution des problèmes de robustesse. Car l'agent mobile est bien affecté au compte de bons résultats dans la découverte des services web mobile. De plus, il est en mesure d'offrir des services d'une manière efficace et plus souple aux utilisateurs mobiles. De même, il permet une diminution du coût de communication par le fait de leur capacité asynchrone et autonome sans connexion au réseau.

Comme on peut constater, les appareils mobiles utilisent des réseaux à connexion coûteuses ou fragiles. Les tâches nécessitant une connexion ouverte en permanence entre un appareil mobile et un réseau fixe ne seront probablement pas économiquement ou techniquement possible.

Pour résoudre ce problème, les tâches peuvent être intégrées dans l'agent mobile, qui ensuite peut migrer dans le réseau. Après migration, l'agent mobile devient indépendant du

processus de création et peut fonctionner de manière asynchrone et autonome (voir la Figure II- 4). Plus tard, Le dispositif mobile peut se reconnecter pour recueillir les résultats.

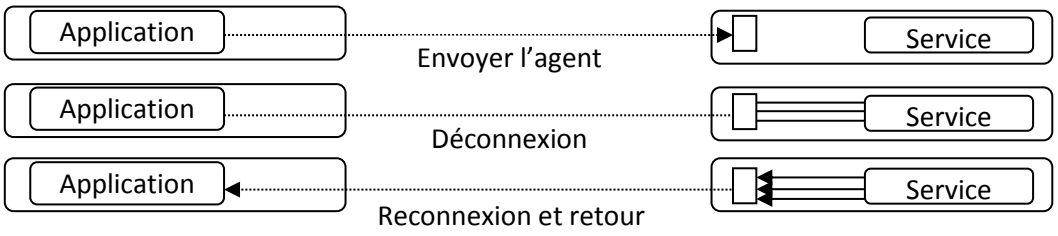

Figure II- 4: Agent mobile et la déconnexion [Lange et Oshima 98].

II.6 Conclusion

Dans ce chapitre, nous avons passé en revue les principaux travaux de découverte des services web mobile. On constate que, malgré l'obtention de résultats non négligeables en ce sens, les approches qui n'utilisent pas l'agent mobile robustes aux différents facteurs de variabilité sont loin d'être atteintes.

Par ailleurs, les travaux de recherche qui utilisent la technologie de l'agent mobile constitue une alternative prometteuse pour résoudre les problèmes. Ainsi l'avantage principal des approches basées sur l'agent mobile réside dans le fait que l'agent mobile travaille d'une manière asynchrone et autonome. Dans le chapitre qui suit, nous entamons le développement de notre contribution.

Chapitre III : Approche proposée

III.1 Introduction

Dans le présent travail, on se concentre plus particulièrement sur la résolution de certains verrous scientifiques liés aux modèles de représentation, de découverte et d'exécution des services web destinées aux environnements à forte composante mobile. L'idée consiste à explorer les propriétés des agents, afin de les adapter aux caractéristiques de services pour rendre ces derniers en éléments autonomes et adaptatifs.

L'objectif essentiel de notre recherche est la proposition d'un environnement de découverte de service web sémantique aux environnements mobiles. En outre, réaliser une architecture basée sur la technologie d'agent mobile. Ce qui rend le système robuste aux différents facteurs de variabilité (La déconnexion fréquente, les trafics de réseau, assurer une meilleure utilisation des ressources) et ainsi, exploiter les avantages du web sémantique pour automatiser la découverte et l'exécution des services web.

Dans le présent chapitre, nous analysons et décrivons notre approche. Entre autre, nous étudions l'architecture proposée et les descriptions détaillées des différentes entités, ainsi que leurs rôles. En outre, nous mettons en relief les différents algorithmes des découvertes.

III.2 Description de l'approche proposée

Ce paragraphe nous permet de mettre en évidence et en relief le développement systématique et évolutif de l'étude architecturale pour la découverte des services web sémantiques en utilisant l'agent mobile. L'idée est d'explorer les propriétés des agents mobiles afin de s'adapter aux caractéristiques des services. Cette dernière élimine les problèmes liés à l'environnement et les appareils mobiles. D'autant plus, elle ne nécessite aucun recours à l'utilisation simultanée et la présence en ligne du demandeur de service. En outre, nous

exploitons les avantages du web sémantique pour automatiser la découverte et l'exécution de services web, en définitif, nous utilisons le contexte de l'utilisateur (localisation et le type d'appareil) pour mieux répondre à la demande de l'utilisateur [Hamida 13 a] [Hamida 13 b].

Par ailleurs, l'architecture dont nous essayons de valoriser se compose de: demandeur de service web, Gateway, registre des services et fournisseurs des services (Figure III- 1)

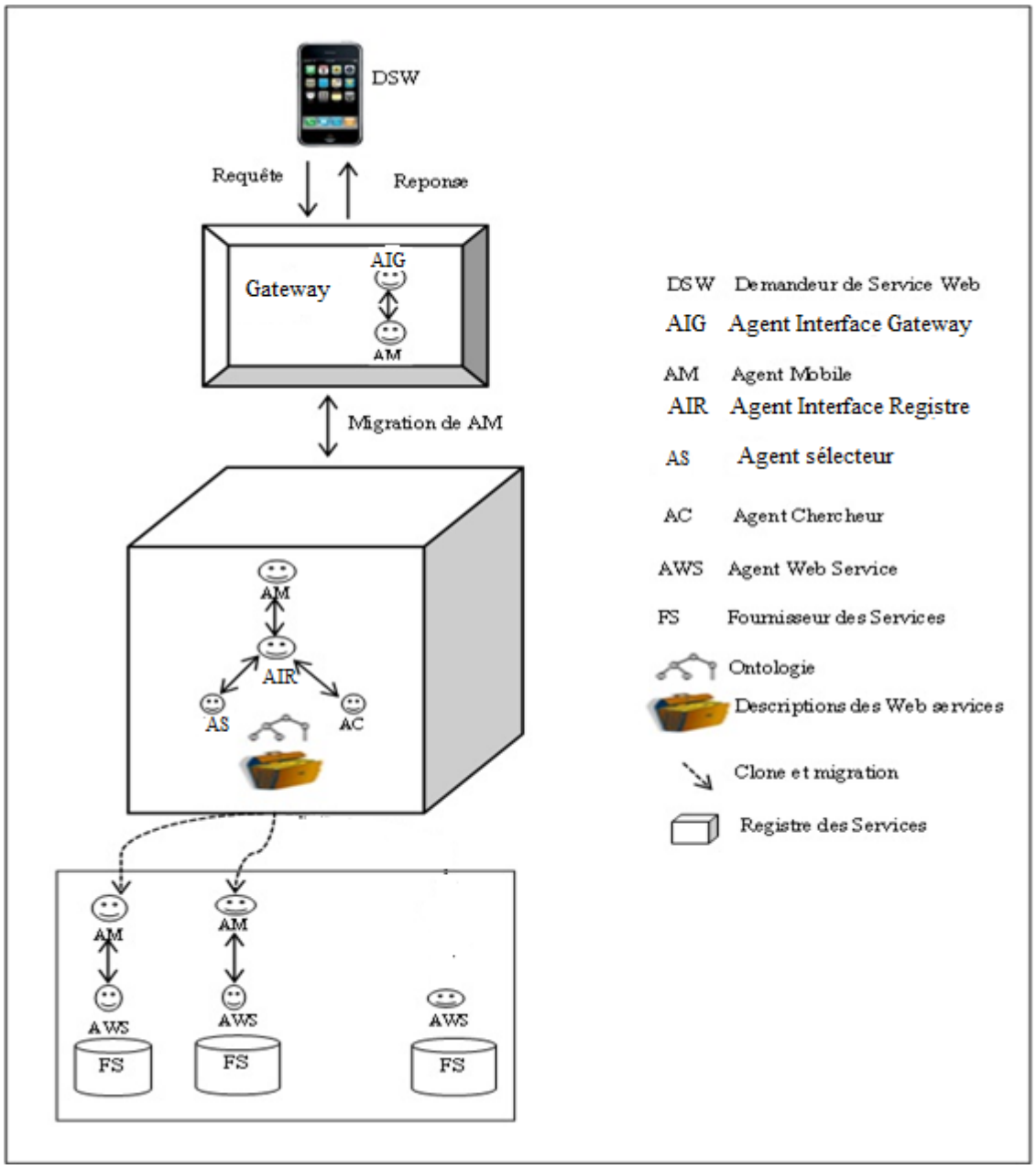

Figure III- 1: Architecture générale de l'approche proposée.

Selon le scénario de notre approche (Figure III- 2), un demandeur de service web introduit une requête à l'aide d'une interface graphique en utilisant son appareil mobile (étape 1). Par la suite, l'agent interface gateway crée un agent mobile et lui transmet la requête et l'information de contexte du demandeur (étape 2). Ensuite, l'agent mobile migre vers le registre des services pour trouver les services web répondant au besoin du demandeur (étape 3) tout en prenant en considération le contexte du demandeur. Le registre des services web permet une meilleure recherche (étapes de 4 à 8), car il est enrichi par des informations sémantiques. L'agent mobile, après avoir acquis les détails appropriés (étapes 9), envoie ces clones vers les fournisseurs de service (étapes 10). Chacun de ces clones exécute le service web, recueille les résultats (étapes 11) et revient au demandeur de services pour lui offrir les résultats (étape 12).

Dans notre cas, deux services web correspondant à la demande ont été trouvés. Cependant, les clones de l'agent mobile migrent et exécutent ces deux services (étapes 10 et 11). Au cas où la demande de service adapté compte plus de deux services, l'agent mobile envoie ces clones vers tous ces services web correspondants. Les avantages obtenus est que l'agent mobile dispose d'une intelligence nécessaire pour exécuter les meilleurs services adaptés. En évitant les exécutions des services inutiles, on permet une meilleure utilisation du réseau. L'agent mobile envoie ses clones aux fournisseurs afin qu'ils exécutent les services en parallèle (de cette façon le temps d'exécution est minimisé), au lieu de réaliser une exécution en série. La connexion permanente de l'utilisateur de dispositif mobile n'est pas nécessaire car il peut obtenir les résultats à tout moment.

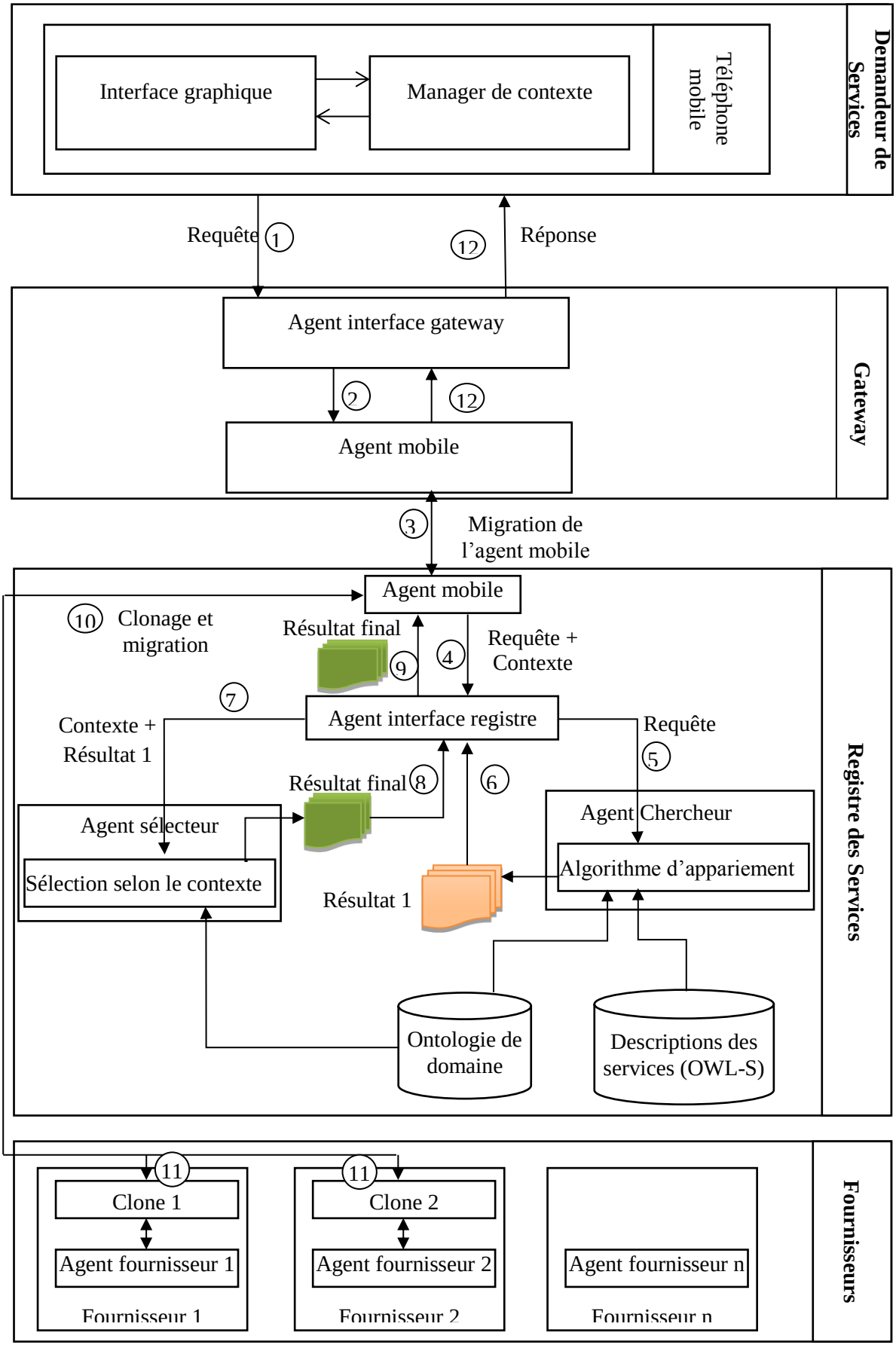

Figure III- 2: Architecture détaillée de l'approche proposée.

III.3 Description détaillée des entités

Notre architecture est constitué de:

III.3.1 Demandeur de service web

L'utilisateur d'appareil mobile appelle un service web à partir de son appareil mobile. L'appareil mobile se met en liaison avec n'importe quel appareil à accès sans fil. Il se connecte au gateway, et soumet sa requête. Ensuite, celle-ci est traitée pour donner naissance aux paramètres fonctionnels (entrée et sortie). Il permet aussi l'enrichissement de la requête utilisateur par d'autres informations relatives à son contexte tel que: le dispositif utilisé et sa localisation.

Le manager de contexte est destiné à la collecte des informations de contexte. Tandis que, le rôle de l'interface graphique est de faciliter l'interaction de l'utilisateur avec notre système (Figure III- 2).

III.3.2 Gateway

Celui-ci permet à l'utilisateur d'appareil mobile d'interagir avec le système. Il joue le rôle d'intermédiaire entre le demandeur de service web et les autres entités au sein de notre architecture. Il dispose de deux composants essentiels à savoir : agent interface gateway et agent mobile (Figure III- 2).

III.3.2.1 Agent interface gateway

Il récupère la requête introduite par l'utilisateur en termes de paramètres fonctionnels (entrée et sortie). Il permet l'enrichissement de la requête utilisateur par d'autres informations relatives à son contexte tel que le dispositif utilisé et sa localisation. Ces informations sont récupérées automatiquement du dispositif de l'utilisateur. Ces derniers sont transmis à l'agent mobile. L'architecture interne de agent interface gateway est comporte les trois modules suivants (voire Figure III- 3):

- **Le module de communication utilisateur :** reçoit les requêtes, les transfère au module de traitement. En outre, Le module de communication utilisateur

peut recevoir des résultats du module de traitement afin de les présenter à l'utilisateur.

- **Le module de traitement :** ce dernier reçoit les données du module de communication utilisateur. Il construit un message et demande au module de communication inter-agents de le transmettre. De plus, il reçoit la réponse du module de communication inter-agents et le transmet au module de communication utilisateur.
- **Le module de communication inter-agents :** il reçoit du module de traitement, des demandes de transmissions de messages vers les autres agents. Il transfère également les informations reçues du système multi-agents au module de traitement.

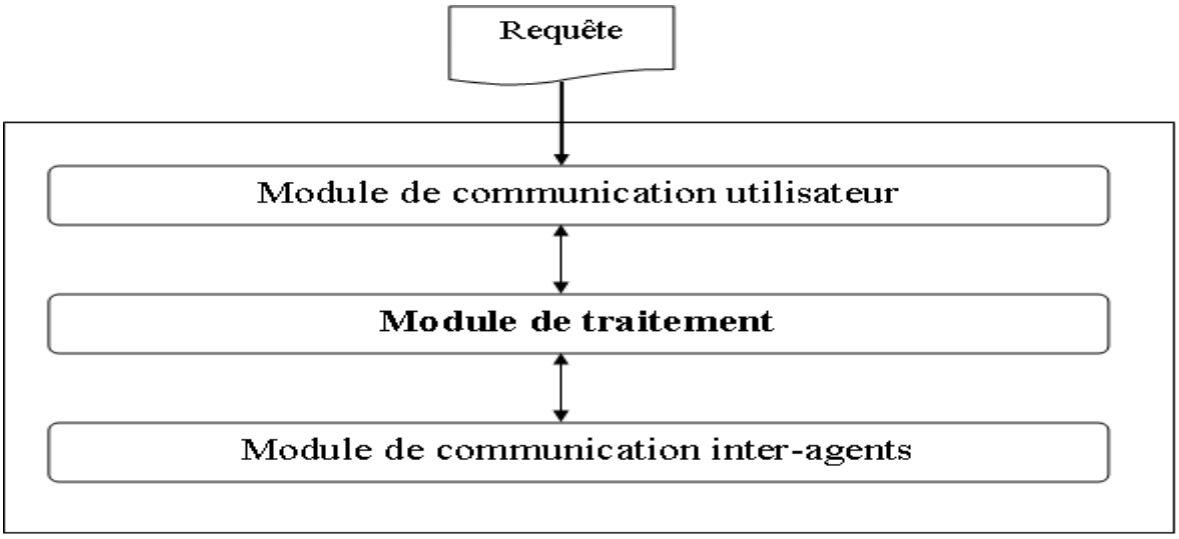

Figure III- 3: Architecture interne d'agent interface gateway

III.3.2.2 Agent mobile

Représente l'utilisateur dans le réseau fixe. Il est apte de trouver, d'exécuter des services web et de fournir des résultats à l'utilisateur. L'agent mobile peut également engendrer des clones qui exécutent les services web sélectionnés en parallèle pour minimiser le temps de traitement total. Les clones peuvent migrer et exécuter les services web simultanément et retourner les résultats au demandeur de service web.

L'architecture interne de cet agent est composée de: module de communication inter-agent, module de traitement, module de gestion de mobilité, module de gestion de clonage.

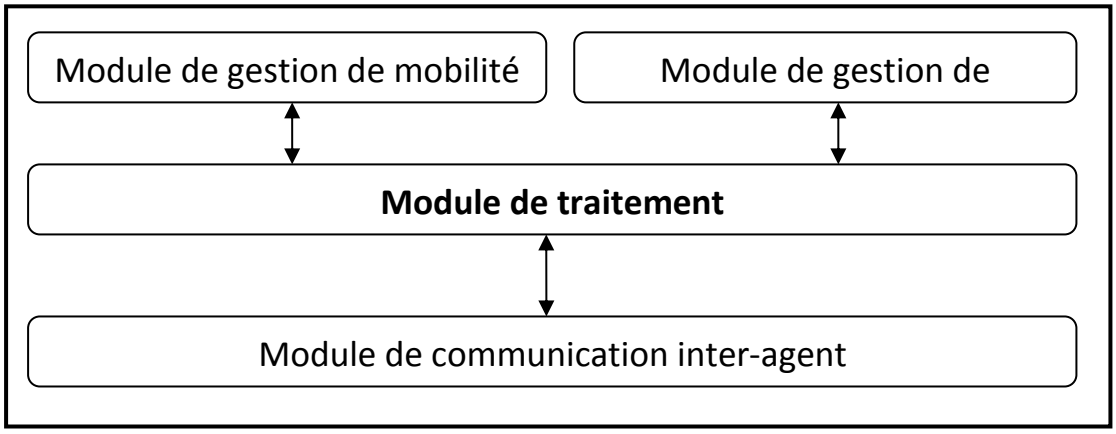

Figure III- 4: Architecture interne d'agent mobile

III.3.3 Registre des services

Le registre des services se compose de trois agents, à savoir : l'agent interface registre, l'agent chercheur et l'agent sélecteur. De plus, il contient les descriptions des services web et les ontologies de domaine.

Notre orientation vers l'ontologie OWL-S comme ontologie de représentation sémantique des services web fut le choix de notre étude [Martin 04]. Celle-ci se justifie par le fait que OWL-S est une ontologie OWL standard du web sémantique, lui donnant une perspective d'avenir d'être un standard pour la représentation sémantique des services web. OWL-S est basé sur le langage d'ontologie du web (OWL). L'objectif de OWL-S est d'assurer l'automatisation de la découverte, la composition et l'exécution. De plus, il répond aux questions suivantes :

- Quelle est la fonction que remplit ce service web?
- Comment fonctionne-t-il ?
- Comment peut-il être exécuté ?

La réponse est que : une ontologie OWL-S est constituée de trois sous ontologies : Profile, Model et grounding, utilisées pour décrire déférents aspects d'un service. La Figure III- 5 représente la description des services web dans la mesure du OWL-S.

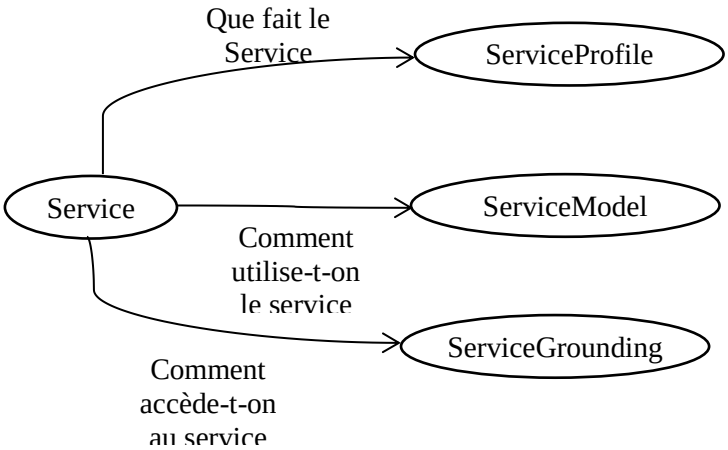

Figure III- 5: L'ontologie supérieure de OWL-S [Chabeb 11]

La Figure III- 6 illustre l'architecture de registre des services. Lorsque l'agent mobile migre vers le registre des services, il envoi la requête et le contexte d'utilisateur à l'agent interface. Par la suite, l'agent interface envoie la requête à l'agent chercheur. Celui-ci applique l'algorithme d'appariement pour trouver les descriptions répondant au besoin d'utilisateur selon les entrées et les sorties. Ensuite, l'agent interface envoie à l'agent sélecteur le résultat de l'agent chercheur et le contexte d'utilisateur. L'agent sélecteur sélectionne les descriptions des services web selon le contexte d'utilisateur.

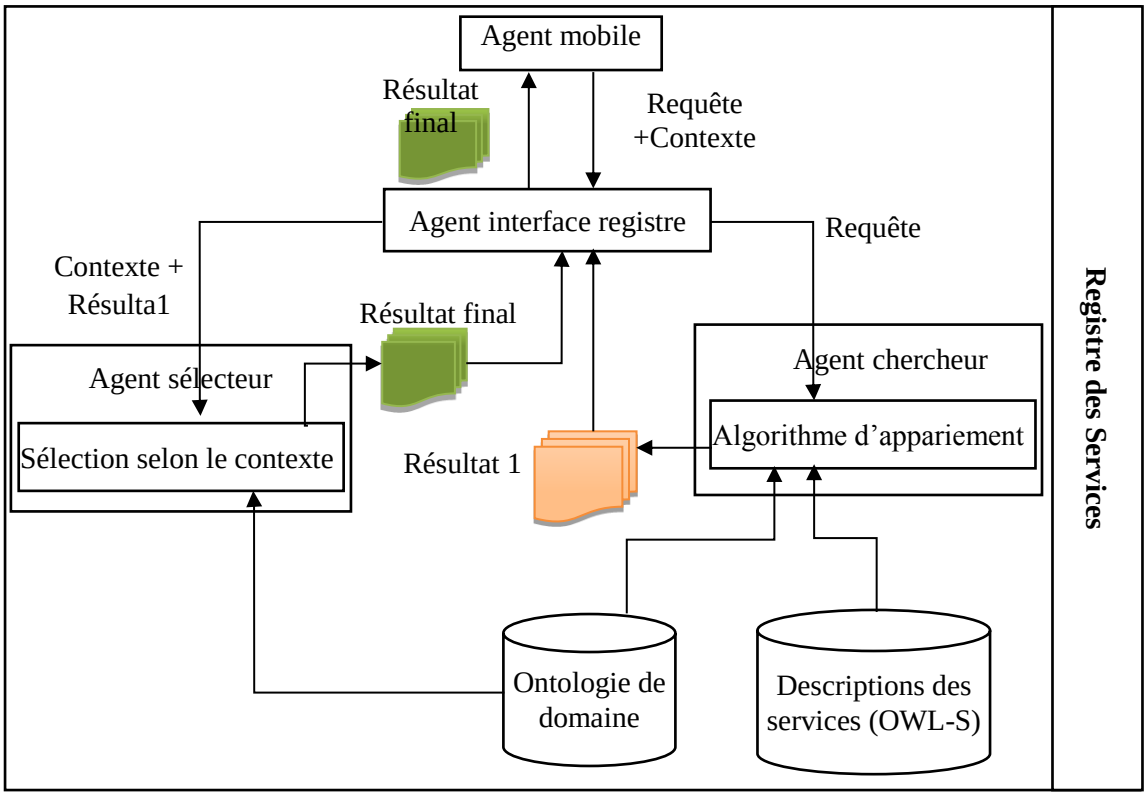

Figure III- 6: Registre des services

III.3.3.1 Agent interface registre

C'est un agent stationnaire qui agit comme agent intermédiaire entre l'agent mobile et les autres agents inclus dans le registre des services. Il agit comme un superviseur, contrôlant les interactions à l'intérieur du registre de Service.

L'architecture interne d'agent interface registre est composée de deux modules. Ces derniers sont le module de traitement (traitant les données entrées et sortie) et Le module de communication inter-agents (Figure III- 7).

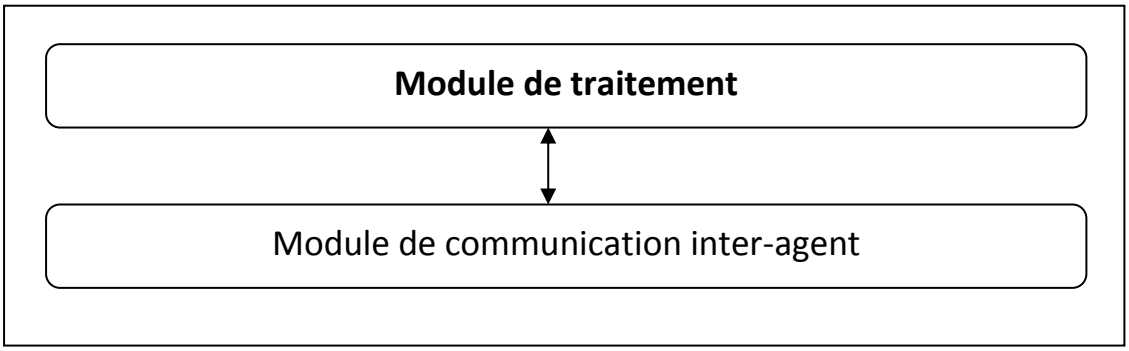

Figure III- 7: Architecture interne d'agent interface registre.

III.3.3.2 Agent chercheur

C'est un agent stationnaire, dont le rôle est la découverte des descriptions des services Web satisfaisant la requête envoyée par l'utilisateur sur le plan sémantique.

L'architecture interne de l'Agent chercheur se compose de:

- **module de communication Inter-Agent :** destiné à l'établir la communication avec les autres agents,
- **module de traitement :** fait pour l'application de l'algorithme de mise en correspondance entre la requête et les descriptions des services web.
- **base de services (AC) :** réservé au stockage des descriptions sémantiques des services Web satisfaisant la requête de l'utilisateur.

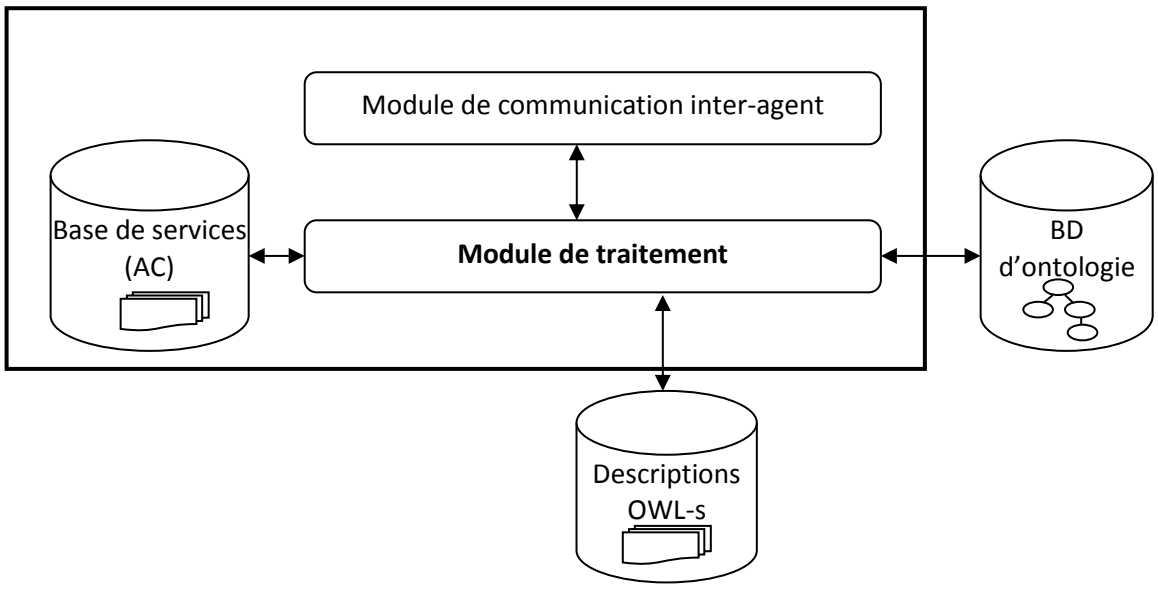

Figure III- 8: Architecture interne d'agent chercheur

III.3.3.3 Agent sélecteur

C'est un agent stationnaire. Son principal rôle est de filtrer cet ensemble en sélectionnant les services web dont les paramètres du contexte (localisation et type de dispositif) sont adéquats avec ceux déterminés à la requête de l'utilisateur (le Module de Traitement est responsable de cette tâche). La base des services (AS) contient les résultats (Figure III- 9).

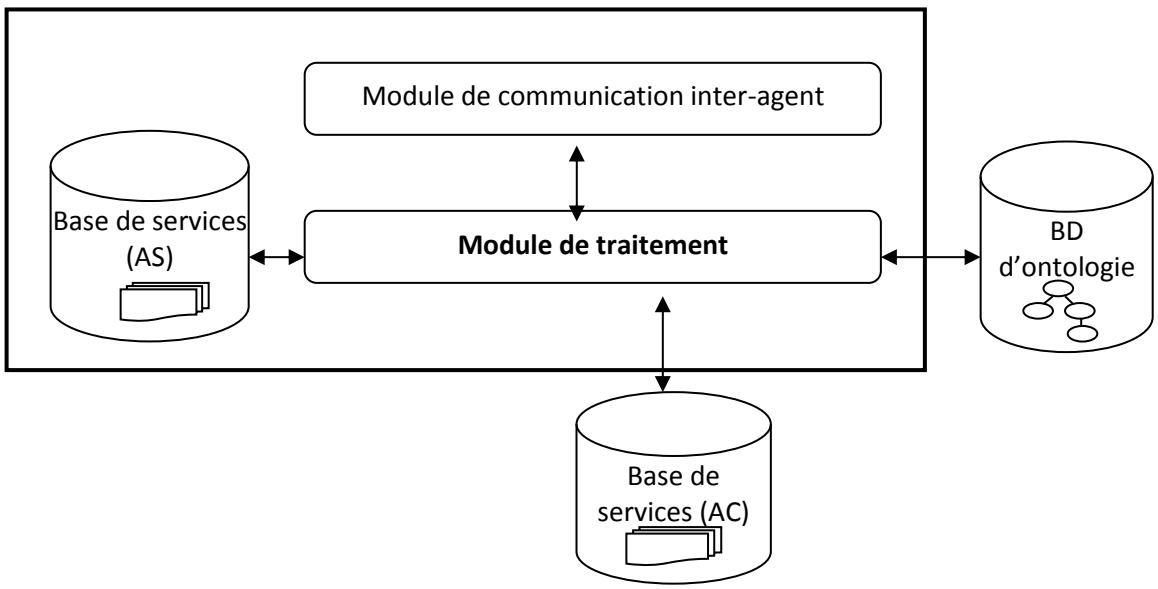

Figure III- 9: architecture interne d'agent sélecteur

III.3.4 Fournisseur de service

Il fournit un service web aux clients intéressés. L'invocation du service par l'agent mobile sur la description sémantique du service. Dans notre architecture, l'invocation de service par agent mobile est effectuée par l'agent fournisseur.

III.3.4.1 Agent fournisseur

C'est un agent stationnaire qui réside dans le fournisseur de service web. Son but est de masquer la fonctionnalité du fournisseur de service web. L'agent fournisseur est créé et maintenu par le fournisseur de services. Lorsque l'agent mobile migre vers le fournisseur de services, il obtient les résultats grâce à l'agent fournisseur.

L'architecture interne de l'agent fournisseur est composée de: module de traitement et module de communication inter-agents (Figure III- 10).

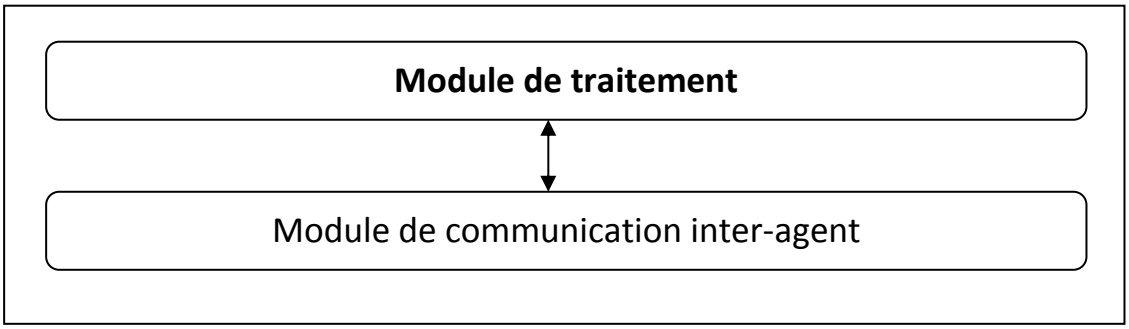

Figure III- 10: Architecture interne d'agent fournisseur

III.4 L'appariement des services et contextes

Dans l'approche proposée, la découverte des services Web sémantiques s'accomplie en deux étapes (Figure III- 11). La première étape se limite à la recherche des descriptions sémantiques des services Web pour satisfaire la requête de l'utilisateur en termes de paramètres fonctionnels (Entrées, Sorties). La seconde consiste uniquement à sélectionner les services Web issus de l'étape de recherche, selon les paramètres du contexte (localisation, type de dispositif de l'utilisateur).

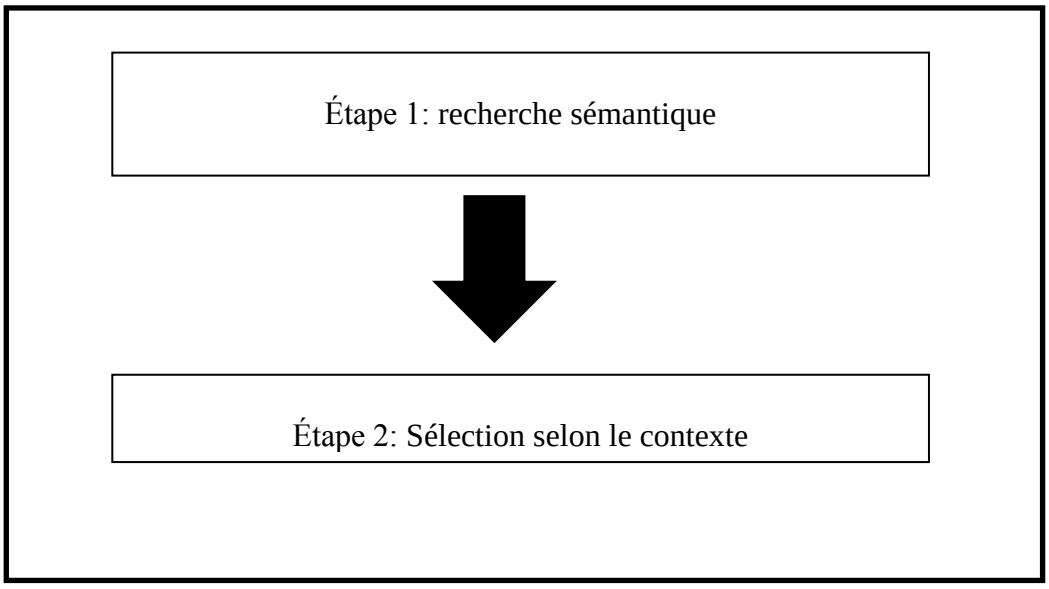

Figure III- 11: Étapes de découverte

III.4.1 Etape 1 : Recherche sémantique

L'agent mobile transmet à l'agent chercheur la requête de l'utilisateur via l'agent interface registre. Une fois la requête acquise, l'agent chercheur applique l'algorithme de mise en correspondance [Paolucci 02]. L'algorithme permet le calcul du degré de correspondance entre les paramètres fonctionnels des services Web et ceux réclamés par l'utilisateur. Le résultat est un ensemble de services Web, satisfaisant la requête de l'utilisateur en terme d'Entrées et Sorties.

Cependant, l'idée principale de cet algorithme est de faire correspondre tous les paramètres d'entrée et de sortie du service à tous les paramètres d'entrée et de sortie de la demande respective. Les paramètres d'entrée et de sortie sont mis en correspondance avec les concepts de l'ontologie de domaine. Le degré de similitude entre ces paramètres est obtenu sur la base de leur relation et de l'inférence dans l'ontologie OWL. Dans [Paolucci 02], quatre niveaux de similitude entre les paramètres sont définis.

- **Exact:** Deux concepts sont classés comme exact, seulement si ces concepts sont exactement identiques dans l'ontologie ou ils ont une relation directe avec la classe et sous-classe dans cette ontologie.
- **Plug-in:** Deux concepts sont définis comme plug-in, seulement si le paramètre d'entrée de la requête est superclasse du paramètre d'entrée de service à

l'exception de la relation directe qui est déjà traitée par degré exact de mis en correspondance.

- **Subsumes:** Deux concepts sont classés comme un degré de correspondance de subsume, seulement si le paramètre d'entrée de demande est sous-classe du paramètre d'entrée de service.
- **Fail:** Ce degré de similitude exprime l'absence de relation sémantique entre deux concepts.

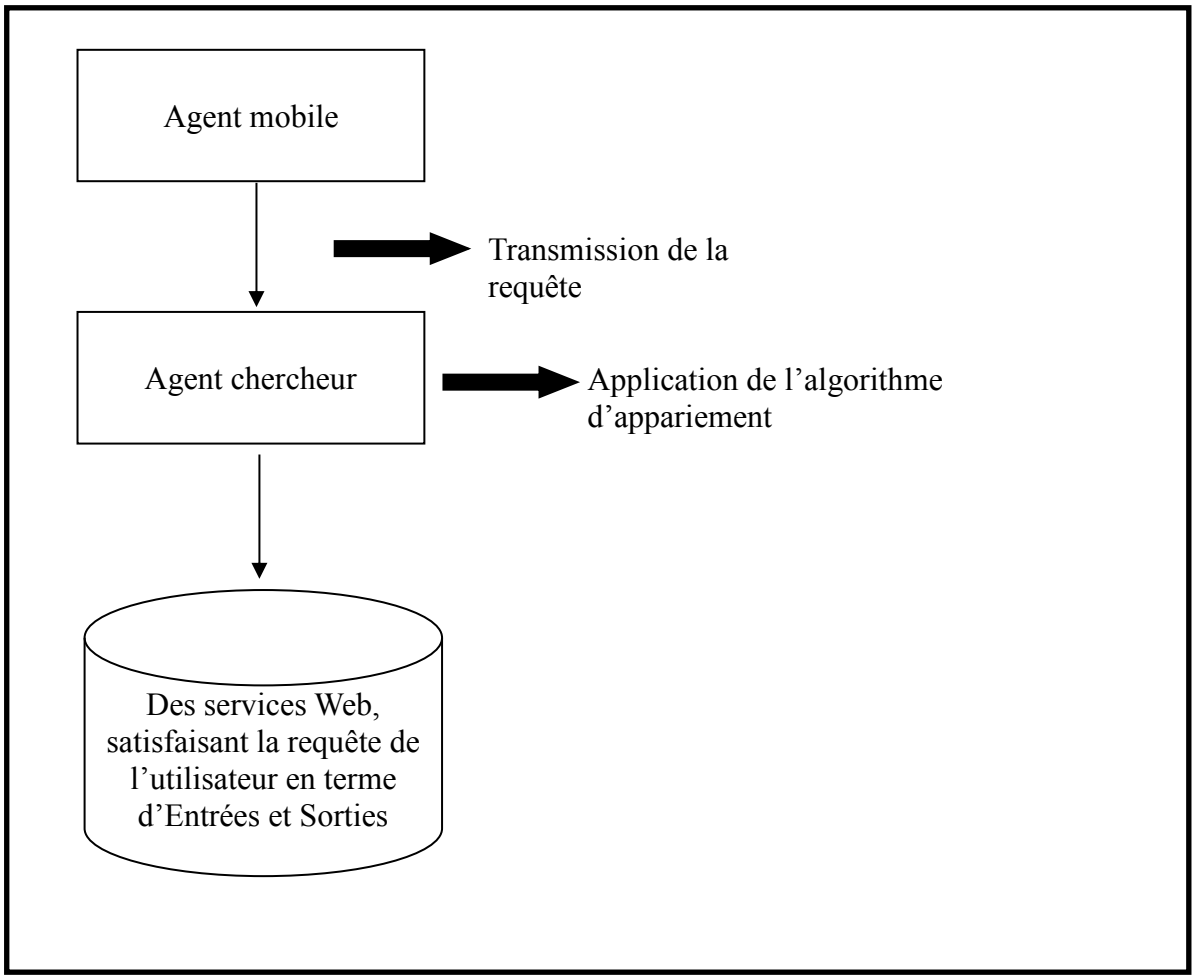

Figure III- 12: Étape 1: recherche sémantique

III.4.2 Étape 2 : Sélection selon le contexte

Les entités entrante en jeu lors de la découverte des services Web sont : l'utilisateur du dispositif mobile et les services Web eux même. Le contexte d'un service Web est constitué de deux paramètres : la localisation du service et les types de dispositifs pris en charge. Le contexte de l'utilisateur quant à lui se caractérise par les paramètres suivants : le type de dispositif utilisé, sa localisation.

Différents modèles de représentation du contexte ont été proposés. Mais nous optons pour les ontologies de la représentation du contexte. Notre choix se justifie par le fait que les ontologies sont considérées comme étant le modèle de représentation le plus expressif. Elles permettent une représentation sémantique existante entre les différents paramètres qui forment le contexte ainsi que leurs relations. Elles sont basées sur les langages du Web sémantique ce qui permet un partage et une réutilisation [Hamida 12 b].

La découverte de services Web sémantiques (OWL-S) est basée sur les paramètres fonctionnels des services. Ces paramètres sont précisés au niveau de la classe profil sous classe de ServiceProfil de l'ontologie OWL-S.

Cependant, d'autres paramètres non fonctionnels sont déterminés au niveau de la classe ServiceParameter sous classes de Profil et qui ne sont pas considérés lors de la découverte des services Web. Ces paramètres forment le contexte du service Web.

Pour avoir des services Web performant à contexte adéquat avec celui de l'utilisateur, nous avons établi une subtile combinaison des paramètres non fonctionnels du service avec ceux des paramètres fonctionnels.

Le choix des paramètres qui forment le contexte du service Web se traduisent par: la localisation du service et les types de dispositifs dont le service peut les adapter à l'information qu'il diffuse.

Suite à la détermination de l'ensemble des services Web, satisfaisant la requête de l'utilisateur en matière de paramètres fonctionnels. L'étape actuelle se consacre à filtrer cet ensemble tout en sélectionnant les services Web dont les paramètres du contexte (localisation et type de dispositif) sont adéquats avec ceux déterminés à la requête de l'utilisateur. L'agent sélecteur qui est responsable de cette étape. La sélection contextuelle s'accomplie en deux phases.

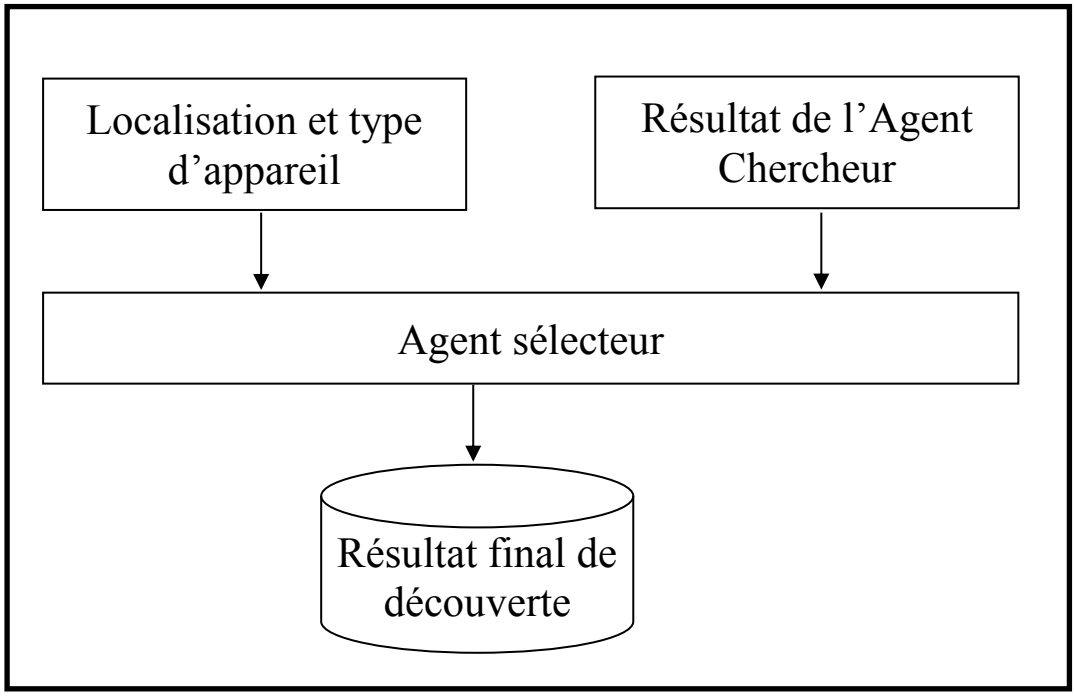

Figure III- 13: Étape 2 : Sélection selon le contexte

III.4.2.1 Phase 1 : sélection selon la localisation de l'utilisateur

Actuellement, la localisation de l'utilisateur est devenue un important paramètre à considérer lors de la sélection des services Web. À cet effet, lors de l'interaction de l'utilisateur avec le système de découverte proposé, le système détermine la localisation de l'utilisateur de façon automatique.

Entre autre, l'information de localisation est ensuite transmise à l'agent sélecteur via l'agent mobile et l'agent interface registre. En se basant sur l'ontologie de localisation relative à un pays ou une région donnée, l'agent sélecteur peut inférer d'autres informations sur la localisation de l'utilisateur, tel que région, ville, etc.

Algorithme 1 selection_localization

/* **Dispositif.Loc** est la localisation de demandeur de service
WSD est la liste des descriptions des web services */

1. **INPUT:** Dispositif.Loc, WSD={WSD_1, WSD_2, …, WSD_{n1}};
2. **OUTPUT:** LWSD;
3. **BEGIN**
4. LWSD=vide;
5. **for**(i=1,i<= WSD.size(), i++) **do**
6. **if**(WSD_i.loc == dispositif.Loc) **then**
7. LWSD.add(WSD_i);
8. **else**
9. **if**(getChildrenClass(WSD_i.loc,Onto) == dispositif.Loc)
10. LWSD.add(WSD_i);

```
11.          end if
12.        end if
13.    end for
14.  END
```

Figure III- 14: Algorithme de la sélection selon localisation

III.4.2.2 Phase 2 : Sélection selon le type de dispositif

La deuxième phase réside en la sélection des services Web s'adaptant au type du dispositif mobile de l'utilisateur. Chaque description de service Web inclut les types de dispositifs supportés par ce dernier. La description du type de dispositif est concrétisée selon l'ontologie du dispositif.

À partir des informations du type de dispositif de l'utilisateur, l'agent sélecteur sélectionne les services web capables d'adapter les résultats transmis aux caractéristiques du dispositif utilisé.

Pour ajuster une information transmise par le service Web aux capacités du dispositif utilisé, nous avons mis en application une ontologie de description du dispositif. Cette ontologie procure un vocabulaire commun aux fournisseurs de services pour décrire les divers types de dispositifs pouvant supporter les données transmises par leurs services Web.

Algorithm 2 selection_dispositif

```
1.  INPUT: dispositif.Type, LWSD={WSD1, WSD2, …, WSDn2};
2.  OUTPUT: LFWSD;
3.  BEGIN
4.      LFWSD=vide;
5.      for(i=1,i<= LWSD.size(), i++) do
6.          if(WSDi.DType == dispositif.Type) then
7.              LFWSD.add(WSDi);
8.          end if
9.      end for
10. END
```

Figure III- 15: Algorithme de sélection selon le type de dispositif

En fin de compte, on a un ensemble de services Web. Ces derniers sont classés selon le degré de correspondance sémantique, dont leur contexte (localisation, type de dispositif) est adéquat avec celui de l'utilisateur

III.5 Les diagrammes de séquence

Les diagrammes de séquences permettent de représenter des collaborations entre entités selon un point de vue temporel. L'ordre d'envoi d'un message est déterminé par sa position sur l'axe vertical du diagramme; le temps s'écoule de haut en bas de cet axe.

Donc cette section, nous développons l'étude des différents diagrammes de séquences, à savoir : le diagramme de séquence d'interaction dans le registre des services, le diagramme de séquence d'interaction dans le fournisseur de service et le diagramme de séquence de fonctionnement général du système [Hamida 12 a].

III.5.1 Diagramme de séquence d'interaction dans le registre des services

A l'arrivée de l'agent mobile au registre de services, il envoie les paramètres fonctionnels et le contexte d'utilisateur à l'agent interface gateway. Celui-ci envoie à l'agent chercheur les paramètres fonctionnels, afin qu'il applique l'algorithme d'appariement sémantique. À son tour, l'agent chercheur enverra le résultat trouvé à l'agent interface registre. L'agent interface registre envoie à l'agent sélecteur le résultat de l'agent chercheur et le contexte d'utilisateur. L'agent sélecteur sélectionne les services web qui répondent au besoin d'utilisateur selon le plan contextuel. Finalement, les résultats finaux sont transmis à l'agent mobile (Figure III- 16).

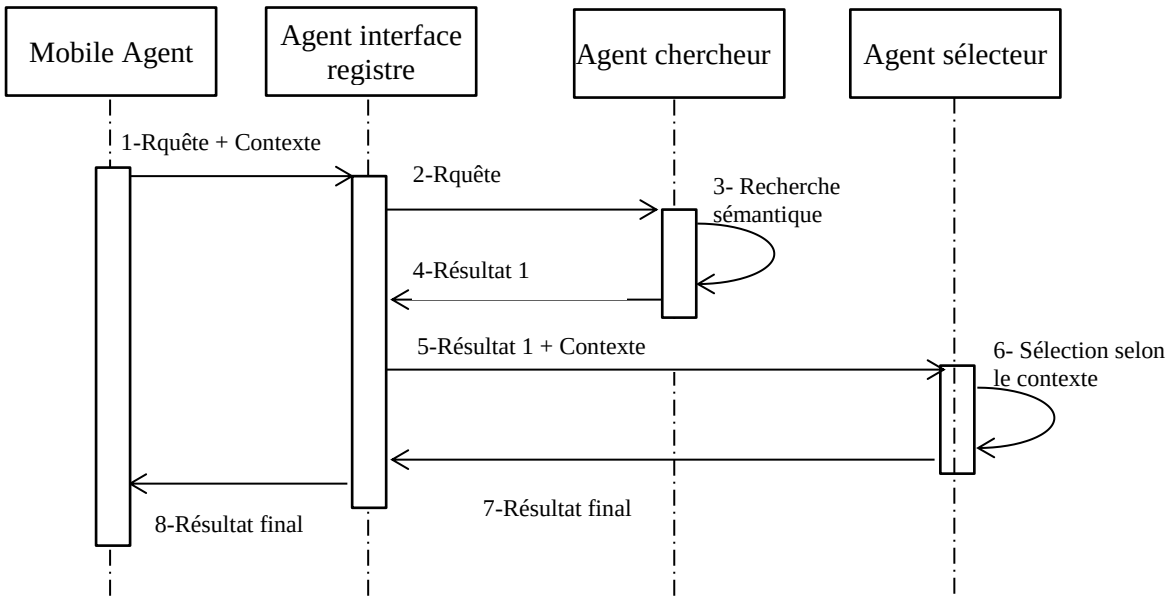

Figure III- 16: Diagramme de séquence d'interaction dans le registre des services

III.5.2 Diagramme de séquence d'interaction dans le fournisseur de service

Dans cette section, nous décrivons les interactions qui ce passe au sein du fournisseur de service par un diagramme de séquence (Figure III- 17). Lorsque l'agent mobile trouve les services web qui répondent au besoin d'utilisateur selon le plan sémantique et contextuel. Il envoie ses clones aux fournisseurs des services web concernés. A l'arrivée des clones aux fournisseurs, chacun d'eux envoie la requête et le contexte d'utilisateur à l'agent fournisseur concerné. À son tour, l'agent fournisseur envoie les résultats nécessaires au clone.

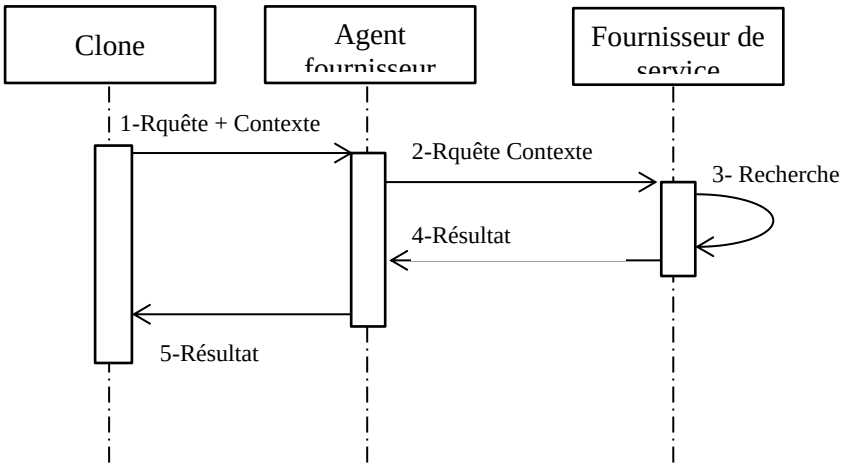

Figure III- 17: Diagramme de séquence d'interaction dans le fournisseur de service

III.5.3 Diagramme de séquence de fonctionnement général

La Figure III- 18 montre le diagramme de séquence de de fonctionnement général. De plus, il décrit le processus de fonctionnement général de notre système. La description des différentes étapes est comme suit:

- Le demandeur de service web (DSW) se connecte au gateway et crée sa demande.
- L'agent interface gateway (AIG) crée un agent mobile qui migre au registre des services pour trouver les services web répondant aux besoins des utilisateurs tout en tenant compte du contexte de l'utilisateur (localisation et dispositif) (étape 2) . L'agent interface gateway réside dans le gateway.
- Dans le registre des services, agent mobile (AM) envoie la requête de l'utilisateur à l'agent interface registre (AIR) (l'étape 3).

- L’agent interface registre envoie à l'agent chercheur (AC), les paramètres fonctionnels de requête de l'utilisateur (entrée et sortie) (étape 4).
- D'autre part, l'agent chercheur initialise le système de raisonnement avec l’ontologie du domaine. Celui-ci sera utilisée pour calculer le degré de correspondance entre les concepts sémantiques visées aux paramètres d'entrée et de sortie spécifiés dans la requête et les services Web (étape 5).
- Dans l'étape 7, l'agent interface registre envoie à l’agent sélecteur (AS) le contexte de l'utilisateur et le résultat de l'agent chercheur. l’agent sélecteur est responsable de la sélection des services web dont les paramètres du contexte (lieu et le type d'appareil) sont adéquats avec ceux déterminés à la demande de l'utilisateur.
- L’agent mobile, suite à l’acquisition des détails appropriés (étape 9), il migre vers le fournisseur de service, exécute le service web, recueille les résultats (étape 10) et revient au service demandeur pour fournir les résultats aux utilisateurs mobiles. L’agent mobile est doté d’une intelligence à exécuter seulement les meilleurs services et évite les services inutiles, ce qui permet une meilleure utilisation du réseau. L’agent mobile envoie ses clones aux fournisseurs des services Web. L’exécution des services en parallèle (réduction du temps d’exécution) s’avère plus performante. La connexion de l'utilisateur de l'appareil mobile est inutile et peut obtenir des résultats dans un proche avenir.

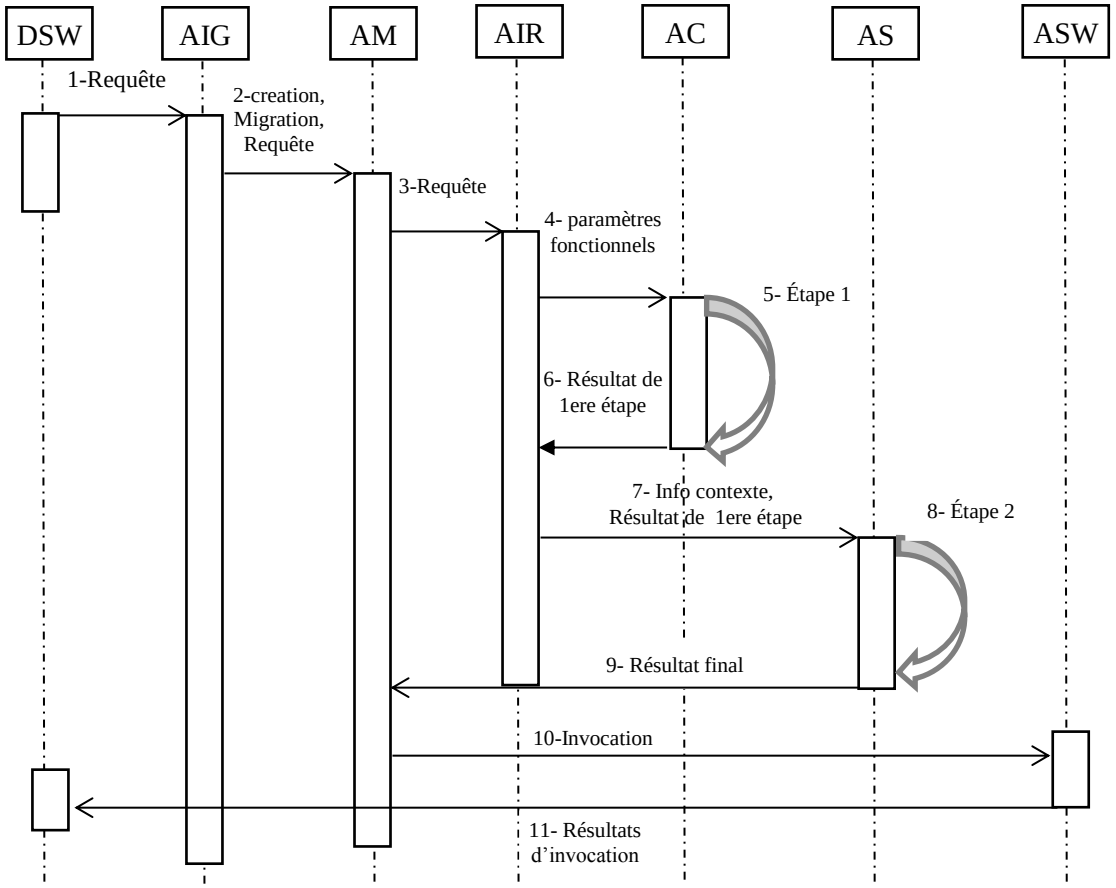

Figure III- 18: Diagramme de séquence de de fonctionnement général.

III.6 Conclusion

Dans ce chapitre nous avons fait l'analyse de la description et la modélisation de notre système proposé. Ce dernier consiste à développer une approche de découverte et d'exécution de services. Celle-ci repose sur trois technologies essentielles qui sont : la technologie d'agent et plus particulièrement d'agent mobile, la sensibilité au contexte et le web sémantique. Le chapitre suivant sera consacré essentiellement à l'implémentation du notre système.

Chapitre IV : Mise en œuvre et résultats

IV.1 Introduction

Dans le précédent chapitre, nous avons décrit en détail notre approche. Celle-ci s'articule sur trois technologies essentielles, à savoir: le web sémantique, la sensibilité au contexte et finalement la technologie d'agent et plus particulièrement d'agent mobile.

Nous consacrons cette partie de notre étude à la valorisation des différentes étapes d'implémentation de notre architecture. Nous tenons à souligner que celle-ci repose sur différentes phases qui seront traitées dans notre étude. De même, nous présentons les outils et les plateformes de développement utilisés à l'implémentation des différents composants du prototype. Finalement, nous exposons les résultats obtenus.

IV.2 Les outils de programmation

Afin d'implémenter notre approche, nous avons utilisé plusieurs outils et plateformes. Ces derniers sont décrits dans cette section.

IV.2.1 Le langage de programmation

Pour édifier notre système, nous avons choisi le langage de programmation JAVA vu ses qualités de simplicité, de robustesse, de portabilité et de dynamisme, en plus il est multi-plate-forme, il est maniable et facile à utiliser.

IV.2.2 La plateforme IBM-Aglet

A la construction des différents agents, notre choix fut focalisé sur la plateforme IBM-Aglet. Cette orientation est justifiée par le fait que celle-ci (la plateforme IBM-Aglet) est un environnement destiné au développement des agents mobiles utilisant le langage JAVA.

Aussi, elle s'agit d'une open source gratuite. Entre autre, elle assure une bonne interface utilisateur graphique au développement de l'agent [Lange et Oshima 98].

L'API Aglet a été développée à Tokyo au début 1995 par une équipe de chercheurs au laboratoire de recherche IBM. Son but est la fourniture d'une plate-forme uniforme destinée à la création d'agents mobiles en environnement hétérogène (ex Internet)

Les aglets (Agents Applets) sont des objets Java mobiles pouvant se déplacer d'une machine à l'autre. Ainsi, un aglet qui s'exécute sur un hôte peut stopper son exécution, se déplacer vers un hôte distant et continuer cette exécution dans son nouveau environnement. Les principaux éléments de cette plateforme sont:

- Proxy : un proxy est un représentant d'un aglet. Il sert de bouclier à l'aglet contre l'accès direct à ses méthodes publiques. De même, Il fournit la transparence à l'emplacement pour l'aglet. C'est-à-dire qu'il peut cacher le vrai emplacement de l'aglet. Proxy de l'Aglet ne se déplace pas.
- Aglet : l'Aglet est un objet mobile. Il est mis en œuvre en Java. Il est autonome et réactif à son environnement.
- Contexte : le contexte est l'environnement d'exécution de l'aglet. Il fournit des services pour l'Aglet et protège l'hôte contre les Aglets malveillants. D'un autre côté, l'hôte est généralement un nœud dans un réseau. De plus, c'est une machine capable d'héberger plusieurs contextes. Les contextes sont nommés et peuvent ainsi être localisés par: le hôte, le port et le nom.
- Identité: chaque Aglet a un identifiant unique.

Les opérations fondamentales supportées dans le modèle d'aglets incluent :

- Création : l'aglet est chargé dans un contexte. L'exécution immédiate de l'aglet commence et un identifiant unique est assigné.
- Clonage : création d'un clone dans le même contexte que l'original. Un identifiant différent est alors attribué. De plus, C'est une deuxième façon pour créer un nouveau aglets
- Dispatching: elle déplace un aglet d'un contexte à un autre. À son arrivée, il démarre en s'exécutant dans un nouveau thread à partir de son point de départ.
- Rétractation : l'aglet est récupéré dans son contexte d'origine.

- Activation et Désactivation : la désactivation d'un aglet est une interruption temporaire de son exécution et stockage de son état dans un support secondaire de stockage. Après un certain temps l'aglet peut être activé à nouveau dans le même contexte.
- Libération ou destruction: c'est la fin de la vie de l'aglet et son retrait du context.

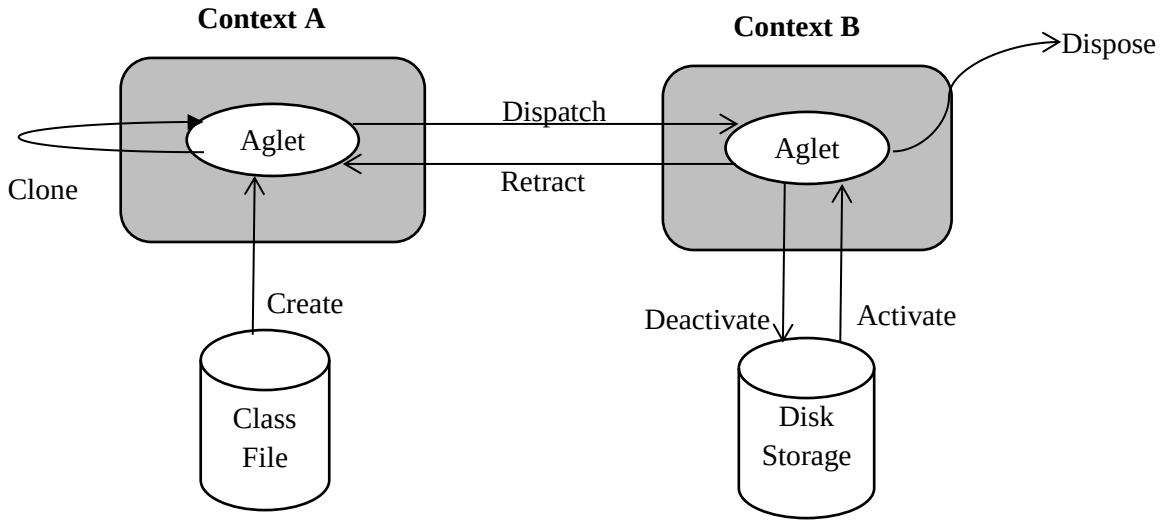

Figure IV- 1: Le cycle de vie d'un Aglet [Lange et Oshima 98]

IV.2.3 Java 2 Micro Edition (J2ME)

L'utilisation de Java 2 Micro Edition (J2ME) a permis le développement de l'interface graphique d'utilisateur pour le téléphone mobile [Muchow 01]. Java 2 Micro Edition (J2ME) est la version Sun de Java est destinée aux machines avec des ressources matérielles limitées tels que les PDA, les téléphones cellulaires et autres appareils électroniques embarqués. J2ME est constitué d'un ensemble de profils. Chaque profil est destiné à un type particulier de dispositif (téléphones cellulaires, les PDA, etc).

IV.2.4 Jena

Jena est un framework Java pour créer des applications Web sémantique [Company 14]. Il offre un environnement de programmation pour RDF, RDFS et OWL, SPARQL et

comprend un moteur d'inférence à base de règles. Nous utilisons principalement ce cadre à des fins de persistance: la lecture et la vérification des ontologies.

IV.2.5 reasoner Pellet

Le moteur Pellet est un des projets du MINDSWAP Group, un groupe de recherche sur le web sémantique de l'université du Maryland [Sirin 07]. Il est disponible en OpenSource et développé en Java. Pellet travaille sur des ontologies décrites en RDF ou OWL. Pellet permet de raisonner sur les logiques de description et aussi sur les instances de concepts.

IV.2.6 Mindswap OWL-S API

Mindswap OWL-S API fournit une API Java pour lire, exécuter et rédiger des descriptions de service OWL-S [Sirin 04].

IV.2.7 Protégé 2000

Protégé est un éditeur d'ontologies distribué en open source par l'université en informatique médicale de Stanford. Protégé n'est pas uniquement un outil spécialement dédié à OWL, mais aussi un éditeur hautement extensible, capable de manipuler des formats très divers [John 03].

IV.3 Exemple d'expérimentation

Notre étude embrasse plusieurs cas et elle est destinée à la démonstration de la validité et la fiabilité de notre architecture. En effet nous allons mettre en application notre approche sur un exemple typique pour les affaires touristique conduites à l'aide d'appareils mobiles sans aucun égard au temps et au lieu.

Nous utilisons la collection de test OWLS-TC4 et d'étendre les descriptions des services web OWLS-TC4 par les informations de contexte. Car OWLS-TC4 ne contient que des descriptions des services de base et sont basées sur OWL-S. dans un autre terme, les descriptions des services web OWLS-TC4 ne disposent pas d'informations de contextes. De plus, nous avons choisi cette collection de test, du fait qu'elle fournit un grand nombre de services de plusieurs domaines, requêtes de test et des ontologies. OWLS-TC4 offre 1083 des services Web sémantiques écrites en OWL- S 1.1 et aussi OWL- S 1.0 à partir de 9 domaines

(éducation, soins médical, nourriture, Voyage, communication, économie, armes, géographie et simulation). Il fournit un ensemble de 42 requêtes de tests associés à la pertinence fixe pour réaliser des expériences d'évaluation de performances. Sur la base de cette collection, nous avons mis en place notre système. Nous l'utilisons pour étendre les descriptions des services OWLS-TCS avec les informations contextuelles.

Supposant qu'une personne s'est rendue à Alger ville (capitale de l'Algérie), cette dernière est dans l'obligation de trouver un hôtel pour passer la nuit. L'hôtel doit être proche par rapport à sa localisation. Pour atteindre son objectif, cette personne se connecte à notre système à l'aide de son téléphone intelligent et établi sa requête (Figure IV- 2 et Figure IV- 3).

Figure IV- 2: connexion au système

Figure IV- 3: écriture de la requête

Après cela, l'agent interface gateway crée l'agent mobile. Celui-ci migre vers le registre des services afin de trouver les services Web qui correspond aux besoins des utilisateurs (Figure IV- 4). La Figure IV- 5 illustre l'agent mobile dans le registre des services avec l'agent interface registre, agent chercheur et agent sélecteur.

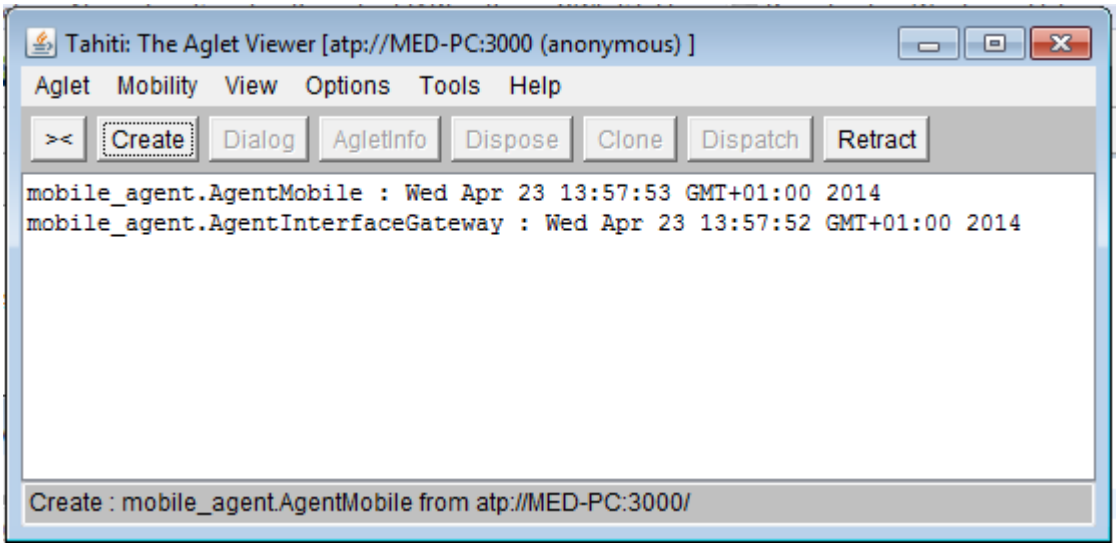

Figure IV- 4: l'agent interface gateway crée l'agent mobile.

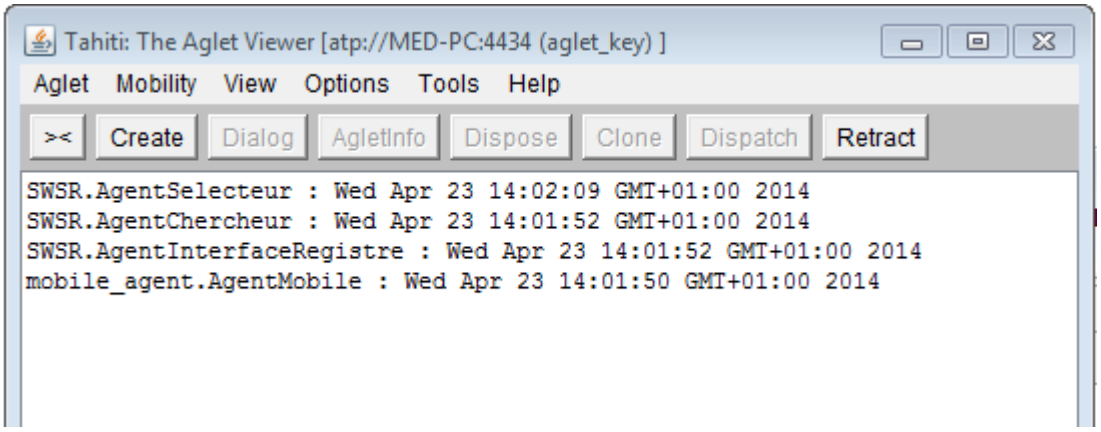

Figure IV- 5: agent mobile dans le registre des services.

En outre, agent mobile envoie ses clones pour invoquer les fournisseurs des services (Figure IV- 6 et Figure IV- 7). Dans notre exemple, mobile agent crée seulement deux clones parce que nous ne disposons que de deux services web répondant aux besoins des utilisateurs.

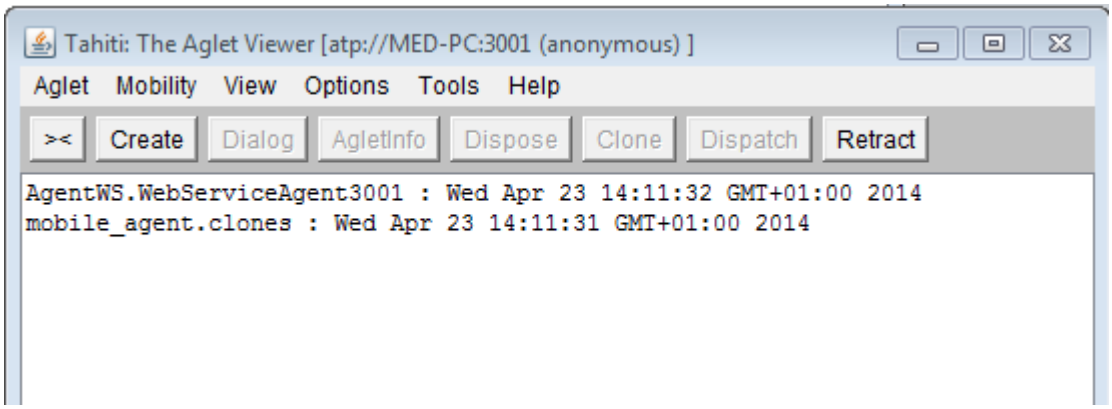

Figure IV- 6: Le premier clone invoque le fournisseur de service

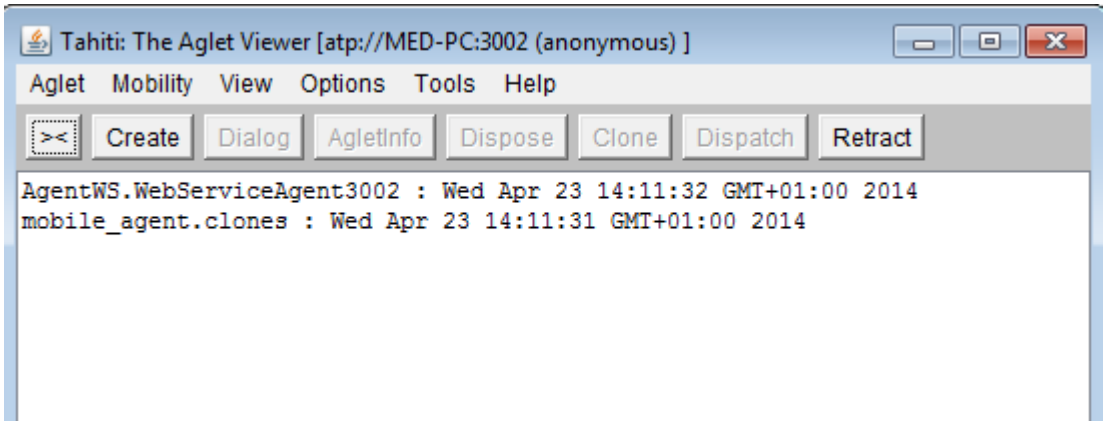

Figure IV- 7: Le deuxième clone invoque le fournisseur de service

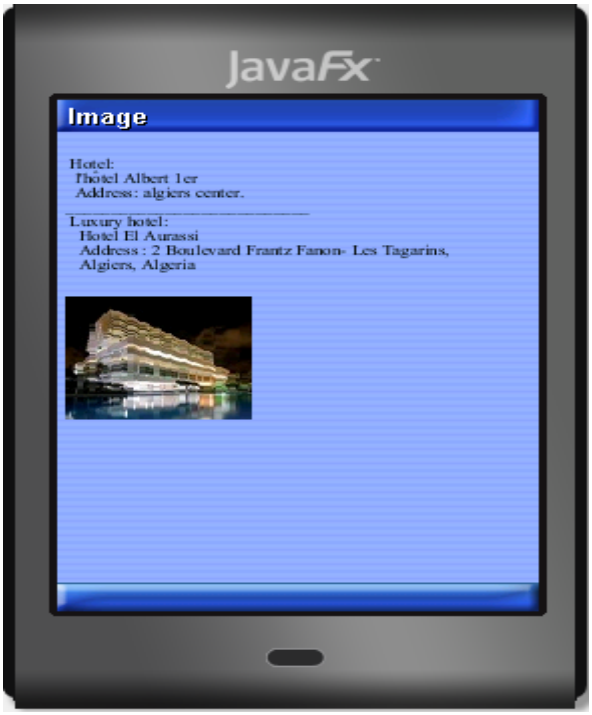

Figure IV- 8: Résultats Finals

IV.4 Conclusion

Suite à ce que nous venons de décrire (partie conception) et la concrétisation de notre architecture, nous exposons finalement les résultats obtenus suite à nos recherches menées à ce jour. De plus, nous avons essayé de mettre en œuvre l'ensemble des idées qui caractérisent l'architecture proposée en se concentrant: sur l'implémentation de la mobilité des agents, la sensibilité au contexte et la sémantique.

Nous constatons qu'avec la mise en service du dispositif mobile de l'utilisateur les résultats avantageux que dégage l'élaboration de notre système sont jusqu'à présent adéquats et satisfaisants. De plus, Le système réduit également les communications dans le réseau pour surmonter la bande passante faible et les fréquentes déconnexions du réseau. D'une seule interaction, l'utilisateur peut exécuter un ensemble de services. L'architecture proposée améliore la fonctionnalité de service en accomplissant sa tâche sans interruption et sans l'intervention constante de l'utilisateur. En outre, elle assure une meilleure utilisation des ressources.

Conclusion générale

L'étude que nous avons réalisée s'oriente et se focalise particulièrement vers le domaine du service web mobile. Son but est destiné à la découverte et l'exécution des services web n'importe où et à tout moment. Dans cette optique, nous avons traité deux problématiques majeures et complémentaires rencontrées en ce domaine. D'une part, nous avons la découverte des descriptions des services web ; d'autre part, l'exécution des services web.

Les services Web mobiles s'avèrent d'un intérêt croissant. Il se traduit par l'extension des services web électroniques aux environnements mobiles. Cependant, les limites et les caractéristiques d'un environnement mobile (la déconnexion intermittente d'un réseau sans fil, le changement fréquent d'informations de contexte). Ainsi que, les appareils mobiles ont des ressources limitées. Ces derniers relèvent des difficultés en la découverte et l'exécution des services web au sein des réseaux sans fil. Cette thèse présente une architecture de découverte et d'exécution des services Web dans un environnement mobile en utilisant un agent mobile et la sensibilité au contexte.

Entre autres, nous avons élaboré une approche utilisant l'agent mobile pour la découverte de m-service web sémantique. Le système développé adopte un registre enrichi d'informations sémantiques fournissant une correspondance sémantique de la requête avec les descriptions des services Web. Étant donné, la diversité des utilisateurs et les conditions d'accès aux services Web, d'autres paramètres doivent être pris en considération lors de la découverte, tel que : le type du dispositif utilisé (PDA, ordinateur portable, etc.), les préférences de l'utilisateur, la localisation de l'utilisateur, etc. Tous ces paramètres forment un contexte d'utilisation particulier. Dans notre étude nous avons décrit les différents paramètres du contexte de l'utilisateur et les services à travers les ontologies (ontologies du contexte de l'utilisateur, de service Web et de paramètres de dispositif).

Ultérieurement, nous procéderons à des extensions à l'architecture en question pour permettre la composition contextuelle des m-services web sémantique en utilisant l'agent mobile. De plus, nous intégrons l'agent mobile et les services web à l'informatique dans les nuages ou le ««Cloud Computing ».

Bibliographie

[Abedi 12] Leila Abedi, Mohammadali Nematbakhsh et Abbas Abdolmaleki. “A Model for Context Aware Mobile Payment”. Journal of Theoretical and Applied Electronic Commerce Research, vol 7, issue 3, pp 1-10, december 2012?

[Abowd 99] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. «Towards a better understanding of context and context-awareness». Lecture Notes in Computer Science, 1707 :304–307, 1999.

[Aeken 99] Francis Van Aeken. « Les systèmes multi-agents minimaux- Un Modèle Adapté à l’Etude de la Dynamique Organisationnelle dans les Systèmes Multi-Agents Ouverts», thèse de doctorat, Institut National Polytechnique de Grenoble, soutenue le 30 mars 1999.

[Akkiraju 05] R. Akkiraju, J. Farrell, J.Miller, M. Nagarajan, M. Schmidt, A. Sheth et K. Verma. «Web service semantics : Wsdl-s». Technical report, IBM, 18 avril 2005.

[Ametller 03] Joan Ametller, Sergi Robles, and Joan Borrell. «Agent migration over FIPA ACL messages». In MATA, pages 210–219, 2003.

[Baker 02] Mark Baker, Rajkumar Buyya, and Domenico Laforenza. «Grids and grid technologies for wide-area distributed computing». International Journal of Software : Practice and Experience (SPE), 32(15) :1437–1466, December 2002.

[Baousis 06] Vasileios Baousis, Elias Zavitsanos, Vasileios Spiliopoulos, Stathes Hadjiefthymiades, Lazaros Merakos et Giannis Veronis. “Wireless Web Services using Mobile Agents and Ontologies”. IEEE International Conference on Pervasive Services (ICPS), Lyon, France, pp : 69 – 77, 26-29 June 2006

[Bellavista 06] Paolo Bellavista, Antonio Corradi et Rebecca Montanari. “A Mobile Computing Middleware for Location-and Context-Aware Internet Data Services”. ACM Transactions on Internet Technology, Vol. 6, No. 4, Pages 356–380, November 2006.

[Bellifemine 99] Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. «JADE —

A FIPA-compliant agent framework». In Proceedings of the 4th International Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-99), pages 97–108, London, UK, 1999. The Practical Application Company Ltd.

[Bellwood 02] T. Bellwood, L. Clement, D. Ehnebuske, A. Hately, M. Hondo, Y. L. Husband, K. Januszewski, S. Lee, B. McKee, J. Munter et C. V. Riegen. « Uddi version 3.0». Published specification, Oasis. 19 juillet 2002.

[Ben halima 09] Riadh BEN HALIMA. « Conception, implantation et expérimentation d'une architecture en bus pour l'auto-réparation des applications distribuées à base de services web ». Thèse de doctorat, Université de Toulouse et de l'université de Sfax. 14 mai 2009.

[Ben Mokhtar 07] Sonia Ben Mokhtar, Davy Preuveneers, Nikolaos Georgantas, Valerie Issarny et Yolande Berbers. «EASY: Efficient SemAntic Service DiscoverY in Pervasive Computing Environments with QoS and Context Support», Journal of Systems and Software 81, 5 (2007) 785-808.

[Benattou 02] Mohammed Benattou and Jean-Michel Bruel. «Active objects for coordination in distributed testing». Lecture Notes in Computer Science, 2425 :348–357, 2002.

[Bernard 99] Guy Bernard, «Technologie du code mobile : état de l'art et persecptive», In Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'99), Nancy, France, Avril 1999.

[Berners-Lee 01] Berners-Lee, T., Hendler, J. et Lassila, O. (2001). «The Semantic Web». Scientific American, p35-43. may 2001,

[Bhalla 10] Mudit Ratana Bhalla et Anand Vardhan Bhalla. « Generations of Mobile Wireless Technology: A Survey ». International Journal of Computer Applications (0975 – 8887), Volume 5– No.4, page 26-32. août 2010.

[Bond 88] Alan H. Bond, Les Gasser, « Readings in Distributed Artificial Intelligence», Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[Booth 04] David Booth, Hugo Haas, Francis McCabe, Eric Newcomer, Michael Champion, Chris Ferris, et David Orchard. «Web services architecture». W3C. 11 février 2004.

[Campadello 00] S. Campadello, H. Helin, O. Koskimies, P. Misikangas, M. Mäkelä, and K. Raatikainen. «Using mobile and intelligent agents to support nomadic users». In Proceedings of the 6th International Conference on Intelligence in Networks (ICIN2000), pages 199–204, Bordeaux, France, January 2000.

[Carzaniga 97] A. Carzaniga, G. P. Picco and G. Vigna, «Disigning Distributed Applications with Mobile Code Paradigms», 19th International Conference on Software Engeneering. ACM Press, May 1997.

[Chabeb 11] Yassin Chabeb. "Contributions à la Description et la Découverte de Services Web Sémantiques", thèse de doctorat, Télécom SudParis, 2011.

[Chappell 02] David Chappell et Tyler JEWELL. «Java Web Service ». édition O'Reilly. ISBN: 0-596-00269-6. mars 2002.

[Chess 94] David Chess, Colin Harrison, and Aaron Kershenbaum, «Mobile Agents : Are They a Good Idea ? », Technical Report RC 19887, IBM Research Division, T.J. Watson Research Center, 1994.

[Christensen 01] Erik Christensen, Francisco Curbera, Greg Meredith, et Sanjiva Weerawarana. « Web Services Description Language (WSDL) Version 1.2 ». W3C, http ://www.w3.org/TR/wsdl, 15 mars 2001.

[Christophe 05] Christophe Cubat Dit Cros « Agents Mobiles Coopérants pour les Environnements Dynamiques », thèse de doctorat, Institut National Polytechnique de Toulouse ,2 décembre 2005

[Colaço 97] Jean-Louis Colaço. «Analyses Statiques de Langages d'Acteurs par inférence de types». Thése de doctorat, ENSEEIHT, Toulouse, octobre 1997.

[Company 14] Hewlett-Packard Development Company. Jena semantic web framework. http://jena.apache.org/, 2014.

[Corbara 1993] B. Corbara, A. Drogoul, D. Fresneau, et S. Lalande. « Simulating the sociogenesis process in ant colonies with manta», In Towards a Practice of Autonomous Systems II, Cambridge, 1993. MIT Press.

[Doulkeridis08] Christos Doulkeridis et Michalis Vazirgiannis «CASD: Management of a Context-Aware Service Directory». Pervasive and Mobile Computing journal, Volume 4 Issue 5, Pages 737-754 October, 2008

[Dumez 10] Christophe Dumez. « Approche dirigée par les modèles pour la spécification, la vérification formelle et la mise en œuvre de services Web composés ». Thèse de doctorat, Université Université de Technologie de Belfort-Montbéliard. 31 août 2010.

[Erman 80] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Raj Reddy, « The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty », ACM Computing Surveys, 12(2), juin 1980, pp. 213-253.

[Farley 05] P. Farley et M. Capp. «Mobile web Services». BT Technology

Journal, vol. 23, no. 2, pp. 202-213 (2005).

[Fensel 02] D. Fensel et C. Bussler. « The web service modeling framework wsmf». Eletronic Commerce : Research and Applications, 1(2) : 113–137. 2002.

[Gehlen 07] Guido Gehlen. « Mobile Web Services - Concepts, Prototype, and Traffic Performance Analysis ». Thèse de doctorat, Université de RWTH Aachen, Allemagne. 17 october 2007.

[Geib 99] Jean-Marc Geib, Christophe Gransart, and PhilippeMerle. «CORBA : des concepts à la pratique». Editions Dunod, Paris, France, Octobre 1999.

[Gray 01] R. S. Gray, G. Cybenko, D. Kotz, and D. Rus. «Mobile agents : Motivations and State of the Art». AAAI/MIT-Press, 2001. publié aussi dans [GCKR00].

[Gray 02] Robert S. Gray, George Cybenko, David Kotz, Ronald A. Peterson, and Daniela Rus. «D'agents : Applications and performance of a mobile-agent system», Softw., Pract. Exper., 32(6) :543–573, 2002.

[Gudgin 03] M. Gudgin, M. Hadley, N. Mendelsohn, J. J. Moreau, H. F. Nielsen,A. Karmarkar, et Y. Lafon. «Soap version 1.2 part 1: Messaging framework». http://www.w3.org/TR/soap12-part1/. juin 2003.

[Hacini 08] SalimaHacini. « Sécurité des Systèmes d'Information : Mise en œuvre de la confiance et de l'adaptabilité pour la protection de l'agent mobile ». Thèse de doctorat, Université deMentouri, Constantine, 21 avril 2008.

[Hamida 12 a] Souraya Hamida, Okba Kazar et Youssef Amghar. «Integration of Mobile Agent and Semantic Web Service in a mobile environment», In Proceedings of IEEE INTECH 2012 Casablanca, Morocco, pp: 349-354, 18-20 September. 2012

[Hamida 12 b] Souraya Hamida, Okba Kazar et Youssef Amghar. «Integration of Mobile Agent and Mobile Web Service», Progress in Machines and Systems journal, Volume 1, Number 2, pp: 43- 51, October 2012.

[Hamida 12 c] Souraya Hamida, Okba Kazar et Youssef Amghar. «La technologie des services Web mobile État de l'art et perspectives». Des 2 emes Journées Doctorial en Informatique de Guelma (JDI'2012), 18-19 Nouvembre 2012.

[Hamida 13 a] Souraya Hamida, Okba Kazar et Youssef Amghar. « An Agent Oriented Approach for Modeling Web Services in Mobile Environments », International Journal of Web Applications, Volume 5, Number 1, pp: 13- 26, March 2013.

[Hamida 13 b] Souraya Hamida, Okba Kazar et Youssef Amghar. «Mobile agent approach for web services discovery in mobile environments», Poster presented at Workshop on Artificiel Intelligence Information and Communication Technologies Biskra : 5-6 May 2013.

[Ismael 99] Leila Ismael and Daniel Hagimont. «A performance evaluation of mobile agent paradigm». In Proceedings of the International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA' 99), pages 306–313, Novembre 1999.

[Jennings 99] Nicholas R. Jennings . «On agent-based software engineering». Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK. Artificial Intelligence 117 (2000) 277–296, 21 september 1999

[Johansen 04] Dag Johansen. «Mobile agents : Right concept, wrong approach». In Mobile Data Management, pages 300–301, 2004.

[Johansen 98] Dag Johansen. «Mobile agent applicability». Lecture Notes in Computer Science, 1477 :80–98, 1998.

[John 03] John H. Gennaria, Mark A. Musenb, Ray W. Fergersonb, William E. Grossod, Monica Crubezy, Henrik Erikssonc, Natalya F. Noyb et Samson W. Tub. «The evolution of Protégé: an environment for knowledge-based systems development». International Journal of Human-Computer Studies, Volume 58, Issue 1, pp: 89–123, January 2003.

[Jun 02] Lu Jun, Lu Xianlang, Han Hong, and Zhou Xu. «Application of mobile agent in wide area network». In IEEE, 2002.

[Ketel 09] Mohammed Ketel. «A Mobile Agent Based Framework for Web Services». Proceedings of the 47th Annual Southeast Regional Conference. March 19-21, 2009.

[Khedr 02] M. Khedr, A. Karmouch, R. Liscano, T. Gray. «Agent-Based Context-Aware Ad hoc Communication». MATA 2002, pp 105-118, 2002.

[Kissoum 10] Yacine Kissoum. « Test des systèmes multi-agents», thèse de doctorat, Universite Mentouri Constantine, le 03 Juin 2010.

[Kobayashi 95] Naoki Kobayashi, Motoki Nakade, and Akinori Yonezawa. « Static analysis of communication for asynchronous concurrent programming languages». In Second International Static Analysis Symposium (SAS'95), volume 983, pages 225–242. Springer-Verlag, 1995.

[Lange 99] Lange, Danny B. et Oshima, Mitsuru. (1999). «Seven Good Reasons for Mobile Agents», Communications of the ACM, 42 (3) 88-89

[Lange et Oshima 98] Danny B. Lange et Mitsuru Oshima : «Programming And Deploying

Java Mobile Agents with Aglets», Addison -Wesley, 1998.

[Leriche 06] Sébastien Leriche. « Architectures à composants et agents pour la conception d'applications réparties adaptables », Thèse de doctorat, Université Toulouse III - Paul Sabatier, 2006.

[Libsie 02] Mulugeta Libsie andHarald Kosch. «Content adaptation of multimedia delivery and indexing using MPEG-7». In Proceedings of the tenth ACM international conference on Multimedia (MM-02), pages 644–646, New York, December 1–6 2002. ACM Press.

[Loureiro 01] S. Loureiro. «Mobile Code Protection». Thèse de doctorat, ENST Paris / Institut Eurecom, 2001.

[Mano 05] Jean-Pierre Mano, Marie-Pierre Gleizes et Pierre Glize. « Résolution émergente et collective de problèmes par systèmes multi-agents: principes et applications». Revue traitement du signal, volume 22, numéro 4, pages 375-391, 1er mars 2005.

[Marques 01] Paulo Marques, Paulo Simões, Luís M. Silva, Fernando Boavida, and João G. Silva. «Providing applications with mobile agent technology». In OpenArch'01 - Fourth IEEE International Conference on Open Architectures and Network Programming, Anchorage, Alaska, April 2001.

[Martin 04] David Martin, Mark Burstein, Jerry Hobbs, Ora Lassila, Drew McDermott, Sheila McIlraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan et Katia Sycara. « Owl-s: Semantic markup for web services ». Technical report, W3C. 2004.

[Marwa 12] Marwa El-Sayed Mohamed. «Wireless Semantic RESTFUL Services Using Mobile Agents». OJEEE, Vol. (4) – No. (1), janvier 2012.

[Milojicic 99] D. Milojicic, F. Douglis, and R. Wheeler. «Mobility : processes, computers and agents». Addison-Wesley, 1999.

[Motta 03] Enrico Motta, John Domingue, Liliana Cabral, et Mauro Gaspari. «Irs-II : A framework and infrastructure for semantic web services». International Semantic Web Conference, pages 306–318. 2003.

[Muchow 01] John W. Muchow: « Core J2ME™ Technology MIDP», December 21, 2001, ISBN: 0-13-066911-3

[Niazi 09] Razieh Niazi et Qusay H. Mahmoud. «An Ontology-Based Framework for Discovering Mobile Services». Seventh Annual Communications Networks and Services Research Conferenc, pp. 178–184. IEEE, May 2009.

[Paolucci 02] Paolucci, Massimo., Kawamura, Takahiro., Payne, Terry R. et Sycara,

Katia P. (2002). «Semantic matching of web services capabilities». In: ISWC ’02: Proceedings of the First International Semantic Web Conference on The Semantic Web, p. 333–347, London, UK.

[Peng 08] Peng Rongqun, MI Zhengkun et WANG Lingjiao. «An OWL-S Based Adaptive Service Discovery Algorithm for Mobile Users». WiCOM '08: 4th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–5. IEEE, October 2008.

[Perret 97] Stéphane Perret. «Agents mobiles pour l’accès nomade à l’information répartie dans les réseaux de grande envergure», Thèse de Doctorat. Université Joseph Fourier - Grenoble I, 19 novembre 1997

[Perret 97] Stéphane PERRET. « Agents mobiles pour l’accès nomade à l’information répartie dans les réseaux de grande envergure », Thèse de doctorat, Université Joseph Fourier - Grenoble I, 19 novembre 1997.

[Picco 98] Gian Pietro Picco. «Understanding, Evaluating, Formalizing, and Exploiting Code Mobility. PhD thesis, Politecnico di Torino, Italy, February 1998.

[Pierce 93] Benjamin C. Pierce and Davide Sangiorgi. «Typing and subtyping for mobile processes». In Proceedings 8th IEEE Logics in Computer Science, pages 376–385, Montreal, Canada, 1993.

[Pinsdorf 02] Ulrich Pinsdorf, Jan Peters, Mario Hoffmann, et Piklu Gupta. «Context-aware Services based on Secure Mobile Agents». In Proceedings of 10th International Conference on Software, Telecommunications & Computer Networks (SoftCOM 2002), pages 366-370, IEEE Communication Society, Ministry of Science and Technology Republic of Croatia and University of Split, R. Boskovica, HR-21000 Split, Croatia, October 2002..

[Qusay et Leslie 06] Qusay H. Mahmoud et Leslie Yu. « Havana agents for comparison shopping and location-aware advertising in wireless mobile environments ». Electronic Commerce Research and Applications . vol 5, issue 3, pp 220–228, 2006.

[Rouvrais 02] Siegfried Rouvrais. Utilisation d’Agents Mobiles pour la Construction de Services Distribués. Thèse, Université de Rennes 1, 2002.

[Sahai 98] Akhil Sahai and Christine Morin. «Mobile agents for enabling mobile user aware applications», Proceedings of the 2nd International Conference on Autonomous Agents (Agents’98), pages 205–211, New York, May 9–13, 1998. ACM Press.

[Satoh 02] Ichiro Satoh. «Physical mobility and logical mobility in ubiquitous computing environments». Lecture Notes in Computer Science, 2535

:186–202, 2002.

[Satoh 04] Ichiro Satoh. «Linking physical worlds to logical worlds with mobile agents». In 5th IEEE International Conference on Mobile Data Management (MDM 2004), page 332, Berkeley, CA, USA, 19-22 January 2004. IEEE Computer Society.

[Satoh 10] Chiro Satoh. «Mobile Agent-based Context-aware Services». Journal of Universal Computer Science, vol. 16, no. 15, pp : 1929-1952, 2010.

[Sesseler 02] Ralf Sesseler, Alexander Keiblinger et Nicolas Varone. « Software Agent Technology in Mobile Service Environments ». ISMIS. 2002.

[Sheu 09] Ray-Yuan Sheu, Michael Czajkowski, Martin O. Hofmann et Greg Schow. «Multiagent-based Adaptive Pervasive Service Architecture (MAPS) ». Proc. of the 3rd Workshop on Agent-oriented Software Engineering Challenges for Ubiquitous and Pervasive Computing, ACM ICPS-2009, Imperial College, London, UK, pg.: 3-8, 2009, ISBN:978-1-60558-647-2.

[Sirin 04] Evren Sirin et Bijan Parsia. «The OWL-S Java API». In: Proceedings of the 3rd International Semantic Web Conference (ISWC). Hiroshima, Japan, November 2004.

[Sirin 07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur et Yarden Katz. «Pellet: A practical OWL-DL reasoned». Web Semantics: Science, Services and Agents on the World Wide Web, Volume 5, Issue 2, pp : 51–53, 2007.

[Srirama 08] Satish Narayana Srirama. « Mobile Hosts in Enterprise Service Integration ». Thèse de doctorat, Université RWTH Aachde, Allemagne. 19 september 2008.

[Steller 09] Luke Albert Steller, Shonali Krishnaswamy et Mohamed Methat Gaber. «enabling scalable semantic reasoning for mobile services». International Journal on Semantic Web & Information Systems, Volume 5, Issue 2, 2009.

[Suarez 11] Luis Javier Suarez, Luis Antonio Rojas, Juan Carlos Corrales et Luke Albert Steller. « Service Discovery in Ubiquitous Computing Environments». The Sixth International Conference on Internet and Web Applications and Services (ICIW 2011), pp.1-9, 2011.

[Tomarchio 00] Orazio Tomarchio, Lorenzo Vita, and Antonio Puliafito. «Nomadic users' support in the map agent platform». Lecture Notes In Computer Science, 1931 :233–242, 2000. Proceedings of the Second International Workshop on Mobile Agents for Telecommunication Applications.

[Toninelli 08] Alessandra Toninelli, Antonio Corradi et Rebecca Montanari,

«Semantic-based discovery to support mobile context-aware service access». Computer Communications, vol. 31, n. 5 (2008), (2008) 935–949.

[Tranier 07] John Tranier. « Vers une vision intégrale des systèmes multi-agents Contribution à l'intégration des concepts d'agent, d'environnement, d'organisation et d'institution », thèse de doctorat, Université des Sciences et Techniques du Languedoc, Soutenue le 18 décembre 2007.

[Ulrich 99] Andreas Ulrich and Hartmut König. «Architectures for testing distributed systems». In Testing of Communicating Systems : Method and Applications, IFIP TC6 12th International Workshop on Testing Communicating Systems (IWTCS), IFIP Conference Proceedings, pages 93–108, Budapest, Hungary, September 1-3, 1999. Kluwer.

[Vasiu 04] Luminita Vasiu and Qusay H. Mahmoud. «Mobile agents in wireless devices». IEEE Computer, 37(2) :104–105, 2004.

[Vigna 04] Giovanni Vigna. «Mobile agents : Ten reasons for failure». In 5th IEEE International Conference on Mobile Data Management (MDM 2004), pages 298–299. IEEE Computer Society, 19-22 January 2004.

[Wonsuk 06] Wonsuk Lee, Kangchan Lee1 et Seungyun Lee. « Intermediary based Architecture for Mobile Web Services ». ISBN 89-5519-129-4. Pages : 1973-1978, 20-22, 2006.

[Xining 10] Xining Li Jiazao Lin et Lian Li. «On the Design of a Mobile Agent Environment for Context-aware M-commerce». Computer Science and Information Technology (ICCSIT), 2010.

[Yasuyoshi 99] And Yasuyoshi Inagaki, Katsuhiko Toyama, and Nobuo Kawaguchi. «Magnet : Adhoc network system based on mobile agents», September 30 1999.

[Younas 11] Muhammad Younas et Soraya Kouadri Mostéfaoui . «A new model for context-aware transactions in mobile services». Journal of Personal and Ubiquitous Computing, Volume 15, Issue 8 , pp 821-831, 2011.

[Zaslavsky 04] Arkady B. Zaslavsky. «Mobile agents : Can they assist with context awareness ? » In 5th IEEE International Conference on Mobile Data Management (MDM 2004), pages 304–305. IEEE Computer Society, 19-22 January 2004.