RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ FERHAT ABBES -SÉTIF-

MEMOIRE

Présenté à la Faculté des Sciences
Département d'informatique
Pour l'Obtention du Diplôme de

MAGISTER

ECOLE DOCTORALE D'INFORMATIQUE (STIC)

Option : Ingénierie des Systèmes Informatiques (ISI)

Par : **Melle. HEDJERES Mouna**

THEME :

Contribution à la Conception et à la mise en œuvre d'Architectures Orientées Services (SOA) dans un environnement coopératif,

"MAeMSOS": Une plateforme Multi Agents Orientée Services pour la e-Maintenance

Soutenu le : 17/04/2011 devant le Jury composé de :

Dr. Abdellah BOUKERRAM	MCA à l'université de Sétif	Président
Dr. Makhlouf ALIOUAT	MCA à l'université de Sétif	Examinateur
Dr. Abdelhafid BENAOUDA	MCB à l'université de Sétif	Membre invité
Pr. Mohammed MOSTEFAI	PR à l'université de Sétif	Rapporteur

Table des matières

Introduction générale

Partie I
Les Architectures Orientées Services

I.1 Les Web-Services

I.1.1 Introduction

I.1.2 Principe de fonctionnement de Web-Services

I.1.2.1 Le fournisseur de service (serveur)

I.1.2.2 L'annuaire de service (Annuaire UDDI)

I.1.2.3 Le consommateur de service (Client)

I.1.3 Architecture de Web-Services

I.1.4 Les technologies utilisées

I.1.4.1 SOAP (Simple Object Access Protocol)

I.1.4.2 WSDL (Web Services Description Language)

I.1.4.3 UDDI (Universal Description Discovery and Integration)

I. 2 Architectures Orientées Services

I.2.1 Introduction

I.2.2 Définitions

I.2.3 Concepts de base d'une architecture orientée services

I.2.4 Couches SOA de la fonctionnalité abstraite

I.2.4.1 L'approche de développement top-down

I.2.4.2 L'approche de développement bottom-up

I.2.4.3 L'approche de développement meet-in-the middle

I.2.5 Les bus de services d'entreprises (ESB : Enterprise Service Bus)

I.2.5.1 L'ESB

Partie II
Les Systèmes Multi Agents

Partie III
« MAMSOS » : une plateforme Multi Agents Orientée Services pour la E-Maintenance

Introduction générale

Les systèmes informatiques actuels se caractérisent par l'évolution des réseaux informatiques et la popularité considérable de l'Internet, composés de plusieurs applications hétérogènes et distribuées. D'où la naissance des besoins d'intégration d'application et d'assurer l'interopérabilité, la coopération et la communication entre les différentes applications.

Les Architectures Orientées Service constituent un concept émergeant pour la résolution de problèmes d'interopérabilité intra et inter systèmes informatiques, en utilisant généralement les Web-Services via les Bus de Service qui permettent de résoudre les problèmes d'hétérogénéité et d'intégration d'applications hétérogènes. L'un des standards les plus importants dans les Web-Services est BPEL, qui permet la création des processus d'entreprise. Il assure la communication, l'orchestration ainsi que l'interaction de Web-Services. Mais ces concepts ne suffisent pas pour dire que la coopération est assurée. La coopération telle qu'elle est définie en partie II, assure trois principales fonctions : l'amélioration de la survie, l'accroissement de performances et la résolution de conflits.

Dans un Système Multi-Agents (SMA), chaque agent travaille d'une manière autonome pour résoudre des problèmes. L'interaction et la coopération avec d'autres agents du système provoque l'émergence de la solution globale. Mais les solutions existantes purement SMA sont propriétaires et délimite l'intégration d'application.

Dans notre approche, nous avons tenté de concevoir des SMAs orientées services pour assurer la coopération inter et intra système. Les communications se font par transfert de messages par un Bus de Services d'Entreprise ESB.

Notre processus de raisonnement sera présenté dans ce mémoire qui est organisé en trois chapitres organisés comme suit:

Le premier chapitre est composé de deux parties, nous avons consacré la première partie aux concepts de base des Web-Services. En deuxième partie, nous avons parlé des SOA, et nous avons donné les principes essentiels des Bus de Services d'Entreprise.

Dans le deuxième chapitre, nous avons introduit la notion d'Agent informatique, les Systèmes Multi Agents ainsi que les différents modèles d'architectures dans ce domaine.

En dernier chapitre, qui constitue notre propre contribution, nous avons essayé d'introduire un modèle conceptuel d'un système de maintenance multi-agents orienté service. Ce dernier se base sur les Bus de services d'entreprise pour assurer la communication et la coopération entre les différents agents du système.

Ce mémoire se termine par une conclusion générale de cette contribtion.

Partie I
Architectures Orientées Service

I.1 Les Web-Services

I.1.1 Introduction

Un Web-Service est un système logiciel destiné à supporter l'interaction entre ordinateurs sur un réseau. Il a une interface décrite en un format pouvant être traité par ordinateur (e.g. WSDL). D'autres systèmes interagissent avec le Web-Service d'une manière prescrite par sa description en utilisant des messages SOAP, typiquement transmis avec le protocole http et une sérialisation XML, en conjonction avec d'autres standards relatifs au web. [16]

IBM donne dans un tutorial la définition suivante des Web-Services:

> *« Les Web services sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le Web. Les Web services effectuent des actions allant de simples requêtes à des processus métiers complexes. Une fois qu'un Web service est déployé, d'autres applications (y compris des Web services) peuvent le découvrir et l'invoquer. »*

I.1.2 Principe de fonctionnement de Web-Services

Le modèle de fonctionnement des Web-Services est principalement composé de trois intervenants en interaction : (cf. figure I.1.1)

I.1.2.1 Le fournisseur de service (serveur)

Un fournisseur de services est la source de la fonctionnalité des services. Un fournisseur crée le Web-Service, puis publie un contrat d'interface ainsi que les informations d'accès au service, dans un annuaire de Web-Services.

I.1.2.2 L'annuaire de service (Annuaire UDDI)

Un annuaire de services est un registre des services disponibles. Chaque fournisseur publie dessus son contrat d'interface avec les informations à utiliser pour localiser le service.

Le client recherche les services appropriés dans l'annuaire et extrait le contrat d'interface correspondant (où et comment trouver un service).

I.1.2.3 Le consommateur de service (Client)

Le client accède à l'annuaire de service pour effectuer une recherche afin de trouver le service désiré. Lorsque il le trouve, il extrait le contrat d'interface correspondant et il l'utilise pour comprendre comment (méthodes disponibles) et où (adresse) accéder au service. Ensuite, il se lie au fournisseur pour invoquer le service.

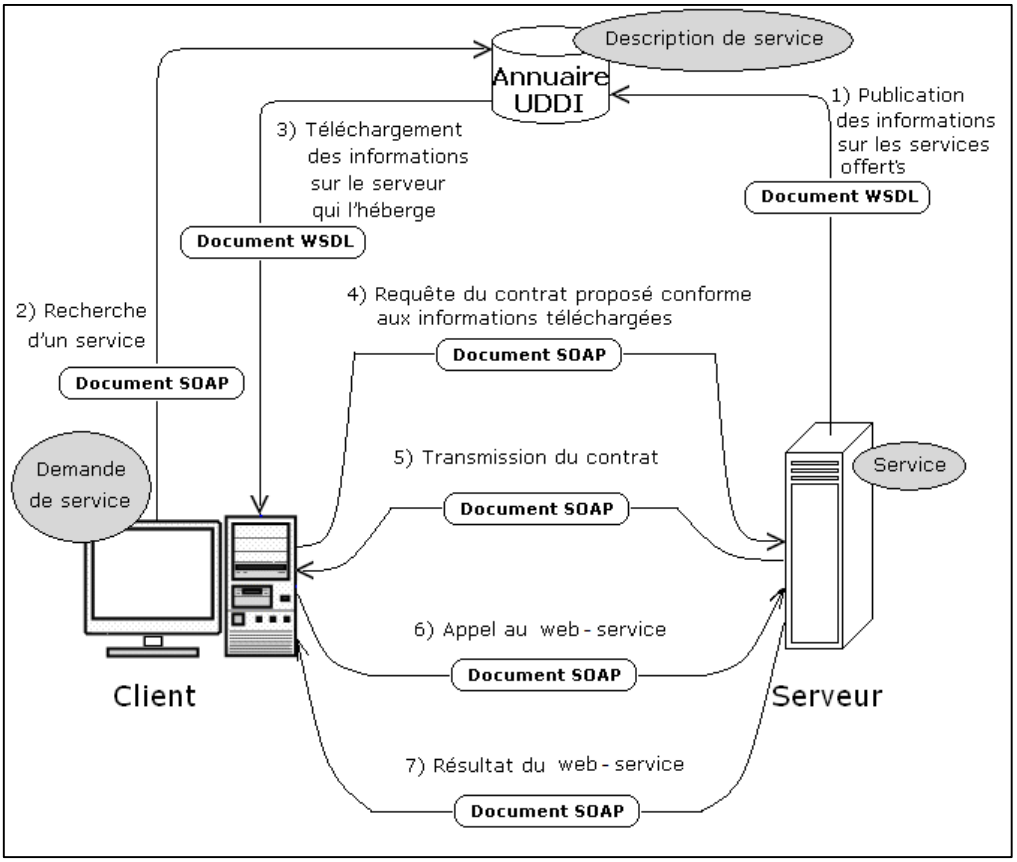


Figure I.1.1 Modèle d'interaction des Web-Services

I.1.3 Architecture de Web-Services

Différentes extensions de l'architecture de référence ont été proposées dans la littérature. Le groupe architecture du W3C travaille activement à l'élaboration d'une architecture étendue standard. Une architecture étendue est constituée de plusieurs couches se superposant les unes sur les autres, d'où le nom de pile des Web-services.

La figure qui suit décrit un exemple d'une telle pile. La pile est constituée de plusieurs couches, chaque couche s'appuyant sur un standard particulier. On retrouve, au-dessus de la couche de transport, trois couches s'appuyant sur les standards émergents SOAP, WSDL et UDDI. L'infrastructure de Web-Service de base définit les fondements techniques permettant de rendre les processus métiers (business processes) accessibles à l'intérieur d'une entreprise et au-delà même des frontières d'une entreprise. Dans ce contexte deux types de couches permettent de la compléter :

Les couches dites transversales (exemple: sécurité, administration, transactions et qualité de services (QoS)) qui rendent viable l'utilisation effective des Web-Services dans le monde industriel ;

Une couche Business processus qui permet l'utilisation effective des Web-Services dans le domaine du e-business.

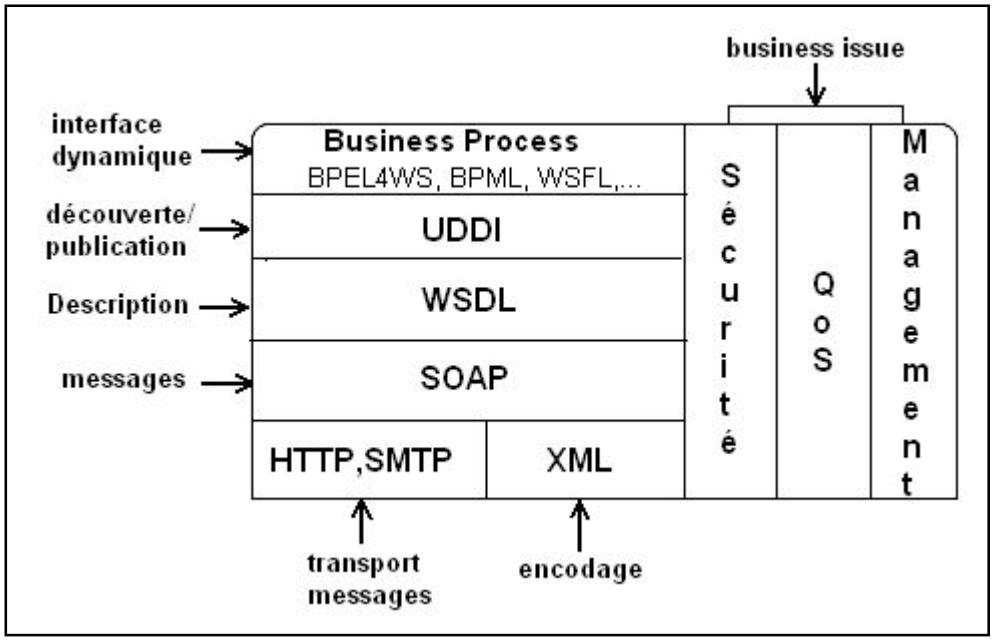


Figure I.1.2 Pile des Web-Services

I.1.4 Les technologies utilisées

Les Web-Services utilisent trois technologies :

SOAP pour le service d'invocation : il permet l'échange de messages dans un format particulier
WSDL pour le service de description : il permet de décrire les Web-Services
UDDI pour le service de publication : il permet de référencer les Web-Services

I.1.4.1 SOAP (Simple Object Access Protocol)

SOAP est un protocole de communication entre applications fondé sur XML, visant à satisfaire un double objectif : servir de protocole de communication sur les intranets, dans une optique d'intégration d'applications d'entreprise, et permettre la communication entre applications et Web-Services, en particulier dans un contexte d'échanges interentreprises. Conçu pour s'appuyer principalement sur HTTP, SOAP ne requiert pas de modification majeure aux serveurs Web déjà installés sur la Toile, tout au plus des extensions. SOAP permet également, grâce au support de XML Schema, de s'interfacer avec des applications préexistantes en capturant leurs structures de données quelle que soit la complexité de ces dernières [5].

La spécification SOAP se divise en trois parties :

- La structure de l'enveloppe SOAP définit une structure générale pour exprimer le contenu d'un message, le responsable de son traitement et le caractère facultatif ou obligatoire de ce message.

- Les règles de codage de SOAP définissent un mécanisme de sérialisation qui peut être utilisé pour échanger des instances de types de données définis par l'application.

- La représentation RPC de SOAP définit une convention qui peut être utilisée pour représenter les appels de procédures à distance et leurs réponses.

I.1.4.1.1 Structure du Message SOAP

Un message SOAP comporte plusieurs parties :

- Une enveloppe qui définit la structure du message
- Un en-tête (optionnel) qui contient les informations d'en-tête (autorisations et transactions par exemple)
- Un corps contenant les informations sur l'appel et la réponse une gestion d'erreur qui identifie la condition d'erreur des attachements ou pièces jointes (optionnel)

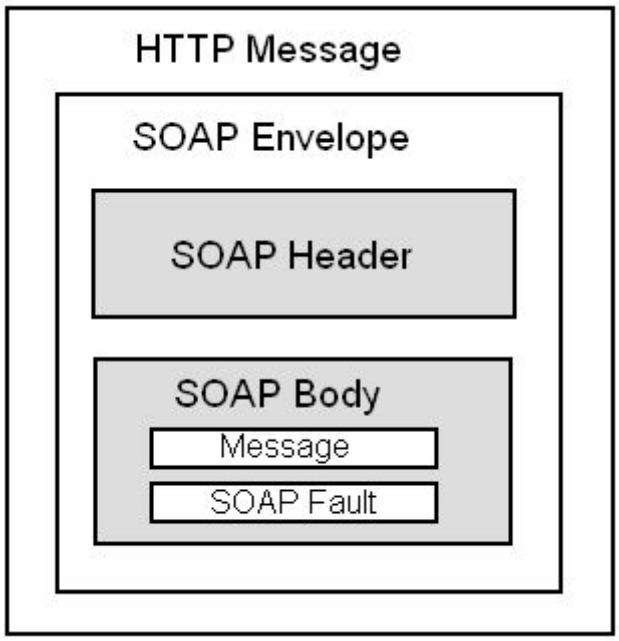


Figure I.1.3 Structure d'un message SOAP

I.1.4.1.2 Règles d'encodage SOAP

SOAP définit aussi l'encodage pour les différents types de données qui est basé sur la technologie schéma XML du W3C. Les données peuvent être de type simple (chaîne, entier, flottant, ...) ou de type composé (structure, tableau,…)

I.1.4.1.3 Modes de communication

Dans les systèmes distribués le choix du modèle d'échange de messages suscite toujours un débat opposant le modèle à sens unique (mode messagerie) au modèle requête-réponse (mode RPC) :

- SOAP RPC:

C'est un appel synchrone de procédures distantes où un client SOAP envoie une requête SOAP avec des paramètres sérialisés pour invoquer les procédures distantes et répondent avec un ensemble de résultats.

- Messages SOAP :

C'est une communication orientée message où des clients SOAP envoient et reçoivent des documents XML de manière synchrone ou non.

I.1.4.2 WSDL (Web Services Description Language)

WSDL est un fichier qui spécifie ce que doit contenir un message de requête et l'apparence du message de réponse dans une notation sans ambiguïté.

La notation utilisée par un fichier WSDL pour décrire les formats de messages est basé sur la norme du schéma XML, ce qui signifie que WSDL est à la fois neutre par rapport au langage de programmation et à la plateforme.

Outre la description du contenu des messages, WSDL définit l'endroit où le service est disponible et le protocole de communications utilisé pour converser avec le service. Cela signifie que le fichier WSDL définit tout ce qui est nécessaire pour écrire un programme fonctionnant avec un Web-Service.

Le WSDL décrit quatre ensembles de données importantes:

- Information d'interface décrivant toutes les fonctions disponibles publiquement.
- Information de type de données pour toutes les requêtes de message et requêtes de réponse
- Information de liaison sur le protocole de transport utilisé
- Information d'adresse pour localiser le service spécifié

Une fois qu'un Web-Service est développé, il faut publier sa description WSDL et faire un lien vers elle dans un annuaire UDDI de sorte que les utilisateurs potentiels puissent le trouver. Quand un utilisateur souhaite utiliser le service, il fait une demande de son fichier WSDL afin

de connaître son emplacement, les appels de fonctions et comment y accéder. A partir de cela, il peut composer, par exemple, une requête SOAP (Simple Object Access Protocol) vers l'ordinateur du service pour consommer le service.

I.1.4.2.1 Structure d'un document WSDL

Un document WSDL est constitué d'un ensemble de définitions :

```
< ?xml version="1.0" encoding="utf-8"? >
   <definitions>
      <types>!--      Les types de données          </types>
      <message>!-- La structure du message  </message>
      <portType>!--L'interface d'accès au sw </portType>
      <binding>! --         Méthode d'accès        </binding>
      <service>! --  Le fournisseur du service   </service>
   </definitions>
```

I.1.4.2.2 Eléments de WSDL

- Définitions

Elément racine du document, il donne le nom du service, déclare les espaces de noms utilisés, et contient les éléments du service.

- Message

Décrit un message unique, que ce soit un message de requête seul ou un message de réponse seul. L'élément défini les noms et types d'un ensemble de champs à transmettre.

- Types

Décrit tous les types de données utilisés entre le client et le serveur. Contient les définitions de types utilisant un système de typage. Les Schémas XML sont utilisés pour définir les types de données.

- Types de port (port Type)

Combine plusieurs éléments message pour composer une opération. Chaque opération se réfère à un message en entrée et à des messages en sortie.

- Liaison (binding)

Défini les spécifications concrètes de la manière dont le service sera implémenté: protocole de communication et format des données pour les opérations et messages définis par un type de port particulier.

- Service

Défini les adresses permettant d'invoquer le service donné, ce qui sert à regrouper un ensemble de ports reliés.

I.1.4.3 UDDI (Universal Description Discovery and Integration)

I.1.4.3.1 Présentation

UDDI est une technologie d'annuaire basée sur XML et plus particulièrement destinée aux Web-Services, notamment dans le cadre d'architectures de type SOA (Service Oriented Architecture).

Un annuaire UDDI permet de localiser sur le réseau le Web-Service recherché. Il repose sur le protocole de transport SOAP .

Né en 1999 de l'initiative d'un certain nombre d'entreprises, dont SUN, Oracle, Microsoft, HP ou encore SAP, il permet de stocker à la fois des information techniques et formelles tel que l'adresse pour accéder au Web-Services, mais également des informations beaucoup plus contextuelles, tel le nom de la personne qui s'occupe de leur gestion, la description sommaire de leur fonctionnalités ou encore le nom et la branche d'activité de l'entreprise dont ils dépendent.

L'annuaire UDDI est consultable de différentes manières :

- Les pages blanches comprennent la liste des entreprises ainsi que des informations associées à ces dernières. Nous y retrouvons donc des informations comme le nom de l'entreprise, ses coordonnées, la description de l'entreprise mais également l'ensemble des ses identifiants

- Les pages jaunes recensent les Web-Services de chacune des entreprises sous le standard WSDL.

- Les pages vertes fournissent des informations techniques précises sur les services fournis. Ces informations concernent les descriptions de services et de information de liaison ou encore les processus métiers associés.

I.1.4.3.2 Model d'informations UDDI

Comporte cinq structures de données principales :

- BusinessEntity: informations sur les entreprises qui offrent les services dans l'annuaire UDDI
- BusinessService : informations sur les services offerts par l'entreprise
- BindingTemplate : informations concernant le lieu d'hébergement du service
- tModel : informations concernant le mode d'accès au service
- publisherAssertion : assertions contractuelles entre partenaires dans le cadre des échanges d'exécution d'un service

I.2 Architectures Orientées Services

I.2.1 Introduction

Le système d'information de l'entreprise est généralement constitué d'applications et de données constituant son héritage. Avec les fusions de groupe, l'évolution des technologies, cet héritage a tendance à devenir hétérogène et à se spécialiser par métier (entité, service, etc.), ce qui provoque un fonctionnement en silo, c'est-à-dire un cloisonnement des différents métiers empêchant certaines formes de transversalité et masquant au décideur une vision globale du système d'information de son entreprise.
L'intégration des applications de l'entreprise (EAI) est une solution à ce problème. Elle consiste à développer des connecteurs spécifiques pour faire communiquer entre eux les différents silos de l'entreprise. [11]

L'objectif d'une architecture orientée services est de décomposer une fonctionnalité en un ensemble de fonctions basiques, appelées services, fournies par des composants et de décrire finement le schéma d'interaction entre ces services.

SOA (Architectures Orientées Service) est une approche émergente pour des systèmes distribués basés sur des standards qui sont faiblement couplés et indépendants du protocole où les ressources logicielles sur un réseau sont considérées comme services.
Ces services sont conçus d'une manière à être invoqué par différents clients, indépendamment des autres services. [15]

I.2.2 Définitions

Architecture logicielle :

L'architecture logicielle d'un système est la description de sa structure en termes des différents composants logiciels le constituant, ainsi que les relations entre ces composants.

Pour les systèmes distribués on distingue trois types d'architectures logicielles, chaque type complémente la faiblesse de l'autre : [14]

Architectures orientées objets :

Les architectures orientées objets nécessitent de communiquer avec l'instance particulière d'un objet qui se caractérise par son état, son comportement et son identité. Les communications sont par nature avec état, puisqu'il faut communiquer avec une instance d'objet crée au préalable. L'information d'état est donc gérée sur le serveur. Toute communication avec l'objet impose un aller-retour avec le serveur. Les protocoles utilisés habituellement (IIOP, DCOM ou RMI) ne sont pas conformes aux standards de l'Internet. C'est la faillite totale de l'extensibilité de ce style d'architecture dans un environnement largement distribué qui explique le passage aux styles d'architectures orientées composants ou services.

Architectures orientées composants :

Un composant est un artefact de granularité plus importante qu'un objet, qui est souvent, mais pas toujours développé avec un langage orienté objet et contient des objets. Un composant exécute une certaine fonction et possède une interface bien définie. Il peut interagir avec d'autres composants. En revanche une compréhension de la technologie utilisée est nécessaire pour manipuler les composants ou les déployer. CORBA, DCOM, J2EE, sont des exemples d'architectures de composants distribués [2]. Ce type d'architecture se préoccupe de la standardisation du mécanisme d'invocation. Sa dépendance de l'architecture applicative en est son point faible.

Architectures orientées services :

Le concept service est destiné aux utilisateurs. Un utilisateur peut être une personne ou un système interne. Un utilisateur ne doit pas avoir besoin de connaître la technologie sous-jacente ou l'implémentation d'un service. Il a juste besoin de connaître l'interface du service, c'est-à-dire la manière avec laquelle il doit utiliser le service. Un service peut être une agrégation de composants. [2]

Ce type d'architecture se préoccupe de la standardisation des protocoles et technologies de communication, indépendamment de l'architecture applicative. Ce qui favorise l'interopérabilité des composants et des services de plateformes et fournisseurs différents.

I.2.3 Concepts de base d'une architecture orientée services

Une architecture orientée service n'as pas une spécification officielle mais on peut donner les concepts de base pour son fonctionnement :

Les services : C'est le concept de base, ça constitue des applications auto-contenues et indépendantes de la plateforme, qui supportent une composition rapide et facile d'applications logicielles distribuées et faiblement couplées avec un moindre coût. [1]
Les fournisseurs de services et leurs consommateurs possèdent des questions sur lesquelles une SOA doit répondre (cf. figure I.2.1). Ces questions peuvent être regroupées en trois catégories :

La description du service : constituant la réponse aux questions : «Comment le représenter ?», « «Que fait il ? ». Le principal format de description de services est WSDL (Web Services Description Language), normalisé par le W3C.

La publication et la découverte des services : constituant la réponse aux questions: «Comment le publier ? », «Où l'héberger ?» et « Où puis-je le trouver ? ». L'architecture doit répondre à la première question en publiant dans un annuaire les services disponibles, et à la dernière question en offrant la possibilité de rechercher un service parmi ceux qui ont été publiés. Le principal standard utilisé est UDDI (Universal Description Discovery and Integration).

L'invocation des services : constituant la réponse aux questions : « Comment pourrai–je l'utiliser ? » et « Comment le réaliser ? » représentant la connexion et l'interaction du client avec le service. Le principal protocole utilisé pour l'invocation de services est SOAP (Simple Object Access Protocol).

Dans un système d'information il existe trois grandes catégories de services : [2]

Les services métiers : qui fournissent des fonctions métiers comme par exemple « soumission de commandes » ou « état de l'activité clients », et sont conçus pour fournir des unités de travail métier à l'intérieur de processus métiers.

Les services applicatifs : qui fournissent également des fonctions métiers, mais sont conçus autour de la mise en œuvre d'une application particulière.

Les services d'infrastructures, qui fournissent des fonctions techniques, telles que : accès aux données, accès aux canaux de communication, service d'impression…etc.

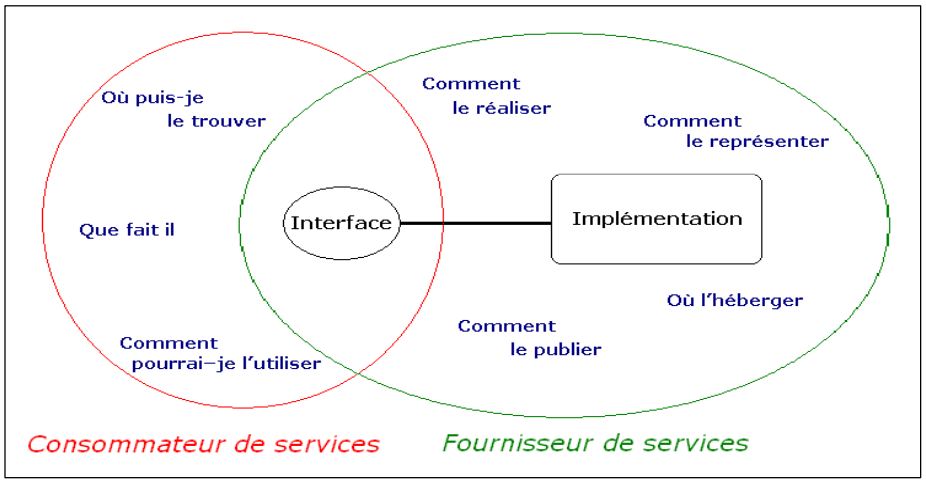


Figure I.2.1 Concepts d'une Architecture Orientée Service

I.2.4 Couches SOA de la fonctionnalité abstraite

Le fonctionnement des Architectures Orientées Services est basé sur des couches de la fonctionnalité abstraite qui facilite leur développement.

La figure I.2.2 nous montre les six couches d'abstraction d'un modèle SOA de base : domaines, processus métiers, services métiers, services d'infrastructures, services de réalisation et les systèmes opérationnels. Chaque couche possède un rôle d'un niveau approprié dans la fonctionnalité d'une architecture SOA [1].

Le flux d'information employé dans le modèle de développement en couches SOA peut suivre une approche top-down, bottom-up ou meet-in-the middle.

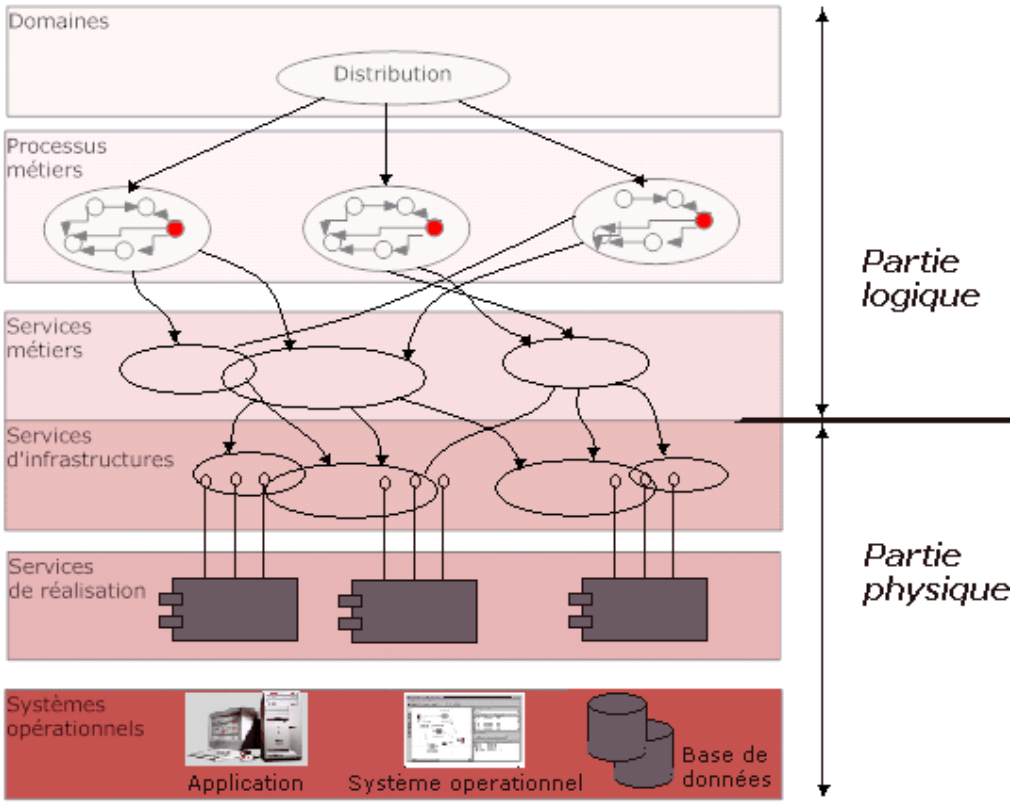


Figure I.2.2 Couches SOA de la fonctionnalité abstraite

I.2.4.1 L'approche de développement top-down

Elle se préoccupe de la manière dont les domaines métier sont décomposés en un ensemble de processus métiers, comment les processus métiers sont décomposés dans les associations de services métiers (services d'infrastructures) et comment ces services sont implémentés dans les entreprises.

I.2.4.2 L'approche de développement bottom-up

Elle suit un processus de développement inverse de la méthode top-down, qui cherche à transformer les applications d'entreprise (par exemple, des bases de données, des systèmes d'information,…) en des

services d'infrastructures et comment des services d'infrastructures sont à son leur tours décomposés en processus métiers.

I.2.4.3 L'approche de développement meet-in-the middle

L'approche bottom-up peut provenir d'un niveau départemental ou organisationnel (dans un système d'information d'entreprise), en commençant par la modélisation des applications existantes sous forme de services et de faire des développements autour de ces services. De la, se retire l'approche la plus commune qui se base sur la combinaison des deux approches précédentes. En lançant ces deux approches en parallèle qui vont se rencontrer au milieu du schéma de développement.

Le processus métier « gestion de commande » (order managment) fournit des services métiers pour la création, la modification, la suspension et l'annulation des commandes erronées. Pour créer et suivre des commandes à un produit, un service ou une ressource et saisir des détails de service sélectionnés par le client.

Les processus métiers orchestrent l'exécution de différents services métiers pour accomplir la fonctionnalité métier exigée et sont ainsi les unités de décomposition en (approche top-down) ou de composition par (approche bottom-up) des services métiers.

Dans une SOA, la couche de services métiers fournit une sorte de pont conceptuel entre les couches de haut niveau orientées métier et celles du bas niveau de l'implémentation techniques. On peut ainsi déduire que les couches : domaine de service, processus métier et services métier constituent la partie logique du cycle de vie de développement de services. De même les couches services d'infrastructures, services de réalisation et les systèmes opérationnels constituent la partie physique du cycle de vie de développement de services.

Avec l'accroissement du nombre de systèmes informatiques et leurs applications hétérogène, les gestionnaire se trouvent devant la nécessité d'intégrer d'autres application qui peuvent appartenir à différentes entreprises, l'intégration d'application permet d'apporter les données ou les fonctions d'une application à une autre application dans une même entreprise ou dans des entreprises différentes, ce qui leur permet de s'échanger des données, des messages et de collaborer à l'aide de processus communs.

Depuis le début des années 90, les solutions d'intégration se sont succédées, avec l'objectif de permettre une intégration de plus en plus fluide et de plus en plus simple. Aujourd'hui, les produits appelés EAI (Enterprise Application Integration), qui sont apparus à la fin des années 90, sont supplantés par un autre outil d'intégration appelé ESB (Enterprise Service Bus) qui proposent une nouvelle approche basées sur les normes et les standards.

I.2.5 Les bus de services d'entreprises (ESB : Enterprise Service Bus)

I.2.5.1 L'ESB

Il n'existe pas une définition normalisée du concept ESB, car avec l'émergence et la croissance des ESB, on se trouve devant le manque de spécification exacte pour les rôles d'un ESB dans une Architecture orientée services. Toutefois on peut définir un ESB par un middleware de communication pour les services s'appuyant sur des standards (Web-Services, langage XML, JMS, …), cette infrastructure assure la transformation et le routage de message entre les différentes applications hétérogènes dans une SOA.

Un bus ESB est une infrastructure logicielle qui active l'architecture SOA en agissant comme une couche intermédiaire de middleware permettant d'accéder à un ensemble de services métiers réutilisables. Un ESB permet aux entreprises de bénéficier des avantages de l'architecture SOA en augmentant la connectivité, en conférant la souplesse nécessaire pour accélérer l'évolution, et en permettant un contrôle accru de l'utilisation des ressources clés qu'il relie.

I.2.5.2 Rôles d'un ESB

Le rôle essentiel d'un ESB se résume à la connexion et à la médiation entre les différents services et applications du Système Informatique, mais ce rôle peut devenir plus large encore.

Un ESB peut posséder les caractéristiques suivantes:

- Il comporte un système de messagerie fiable, les messages consistent en des données XML et des métadonnées de message (généralement WSDL). Toutes les applications sont modélisées en services. Une application peut être un fournisseur de services, un consommateur de services, ou assurer les deux rôles. Le modèle de services est basé sur l'échange de messages.
- Il fournit des services de transformation de messages vers un format standardisé.
- Il fournit des fonctions de sécurité pour contrôler l'accès aux services.
- Il assure une administration centralisée, malgré sa distributivité.
- Il permet d'effectuer des routages de message basés sur le contenu. Idéalement ce routage s'appuie sur un annuaire.
- Il peut avoir un service d'annuaire. Cet annuaire va recenser tous les services existants connectés à l'ESB ainsi que leurs caractéristiques.
- Il automatise les processus de l'entreprise en utilisant des outils d'orchestration de processus métiers, qui définissent les échanges entre les différents départements de l'entreprise.
- Il peut être un médiateur de protocoles. Il présente à l'appelant un service dans un protocole donné et l'implémente par appel d'un service cible dans un autre protocole.
- Il peut héberger un nouveau service, implémenté par composition d'autres services lorsque l'application qui fournit les primitifs métiers ne peut pas exposer facilement ou rapidement des services conformes à la définition métier qui en a été faite.
- Il est caractérisé par un ensemble de conteneurs de service, utilisé pour l'adaptation de plusieurs applications hétérogènes à l'ESB.

I.2.5.3 Structure du bus de services d'entreprise

Un ESB permet l'implémentation, le déploiement ainsi que la gestion des solutions basées SOA, en particulier pour l'assemblage, le déploiement et la gestion des SOA distribuées. [3]

La figure I.2.3 montre une vue simplifiée d'un ESB qui intègre diverses applications et technologies, par exemples une application J2EE utilisant un Service en Java, une application .NET utilisant un client en C#, une application IBM WebSphere MQ qui s'interface avec des applications légataires et des sources de données utilisant des Web-Services.

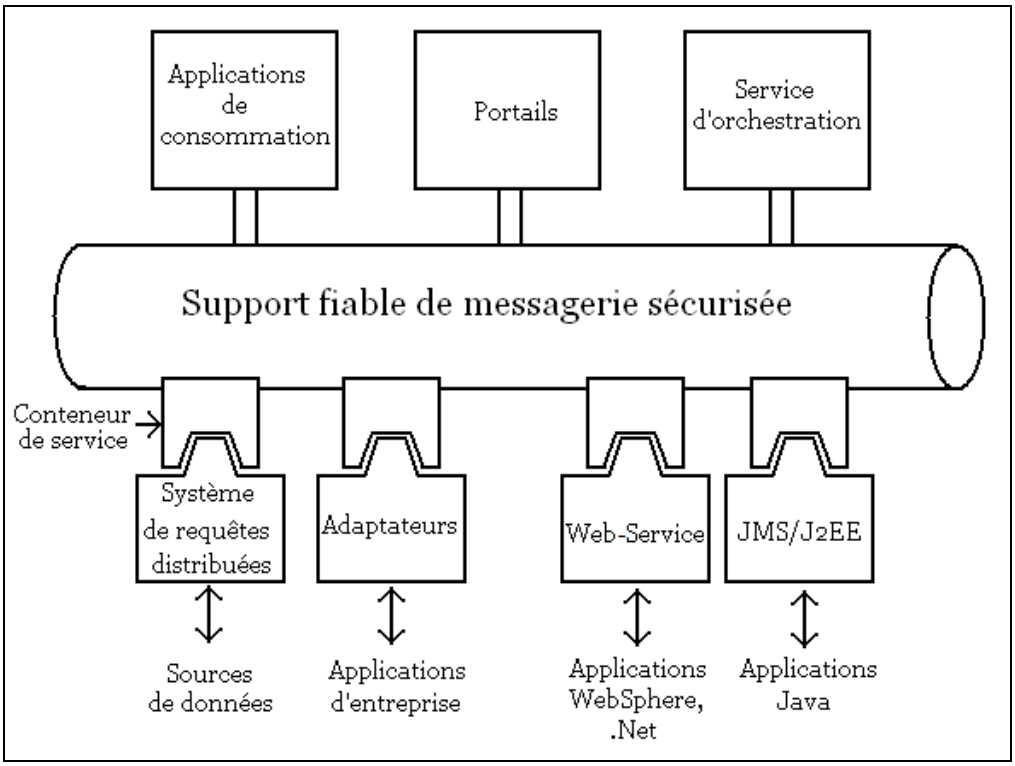


Figure I.2.3 Intégration de diverses applications et technologies par ESB

L'ESB permet l'intégration de différents composants d'application en les positionnant derrière une façade orientée service et en appliquant la technologie Web-Services au problème. Et permet aussi d'offrir des points d'accès uniques aux utilisateurs des différentes applications interactives et processus métiers disponibles, en utilisant des portails qui regroupement des applications composites.

Un autre composant de l'ESB est le moteur d'orchestration de services (Généralement suivant la spécification BPEL).Un concept émergent de base pour ce middleware est le conteneur. Un conteneur expose les fonctionnalités et les propriétés non fonctionnelles de service au monde extérieur via le réseau.

I.2.5.3.1 Le conteneur (Container)

Un conteneur de service est le point d'accès (endpoint) d'un service à l'ESB, qui fournit une implémentation de l'interface de service. Il peut contenir des composants logiciels qui constituent les services d'intégration.
Un conteneur offre plusieurs rôles qui sont déployés dans un ESB: [6] [4] :

- La connectivité et le routage.
- Le modèle d'échange de message
- La sécurité
- Les métriques de performance et QoS
- La configuration dynamique
- La découverte et l'invocation de services
- L'adaptation de protocole et le monitoring du comportement interne et l'état de services.

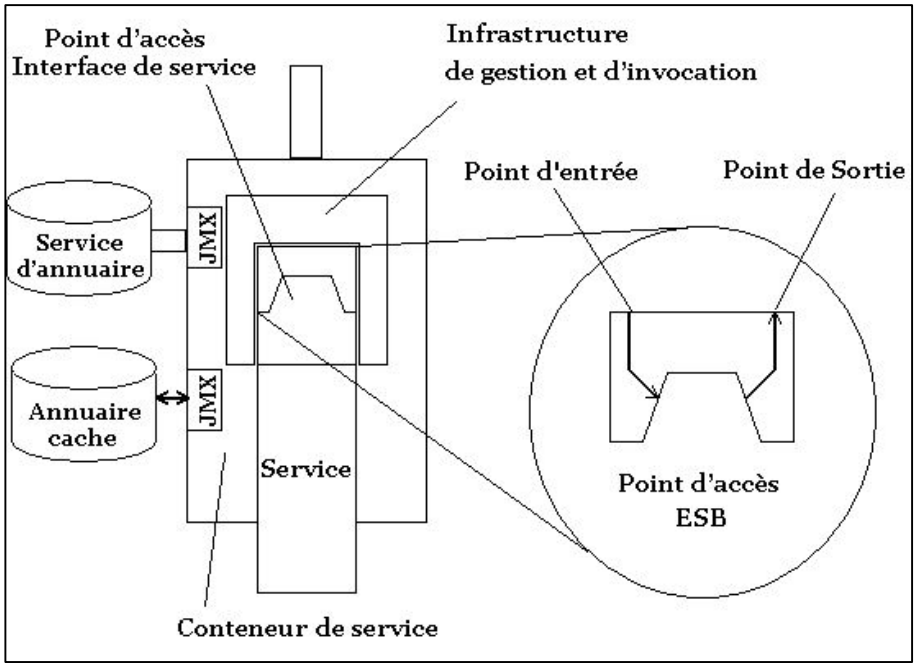


Figure I.2.4 Structure de conteneur

Les différents conteneurs de service sont administrés à distance via un service d'annuaire (Directory Service). Un service de conteneur ESB supporte l'extraction des données de configuration à partir d'un service d'annuaire (cf. figure I.2.4)

Il peut également avoir un cache local de données de configuration. Cela signifie que le service conteneur peut continuer à fonctionner avec son ensemble de données de configuration même si les autres parties de l'ESB, y compris le service d'annuaire, soient temporairement indisponibles.

Le conteneur gère dans sa partie « point d'accès ESB » un point d'entrée et un point de sortie, le point d'entrée fourni l'adresse du point d'accès (endpoint) duquel un distributeur de messages (dispatcher) accepte l'arrivée de messages pour l'invocation d'un service.

On peut avoir trois types de points de sorties :

- Un point de sortie normal: donne l'adresse de l'expéditeur fournie pour le service de réponse, pour qu'il puise lui renvoyé un message de réponse. Elle peut également être configurée pour envoyer le message de réponse à un autre service du processus métier.
- Un point de sortie pour messages rejetés : lorsqu'un message ne peut pas être livré à cause d'une adresse invalide par exemple ou lorsqu'il est mal formé, il sera envoyé à une adresse de rejet configurée par défaut.
- Un point de sortie défaut pour les messages erronés : Fourni l'adresse d'envoi d'un message d'erreur (indiquant que le service ne peut pas exécuter la requête). Cette adresse est généralement configurée à une adresse par défaut, mais elle peut être l'adresse de l'expéditeur.

Ces différents points d'accès sont utilisés par le conteneur pour envoyer un message à destination et en provenance du service.
Alors qu'on trouve dans la partie « point d'accès interface de service » une spécification de connexion et protocoles utilisés par le service.

I.2.5.4 Exemple d'invocation d'un service

Dans une infrastructure ESB toutes les applications sont modélisées sous forme de services. Chaque service d'ESB peut être configuré avec un ensemble d'adresses ESB pour les différents points d'accès, pour la réception de messages et pour l'envoi de réponses. L'utilisation de ces adresses standards dans la configuration de l'ESB permet au développeur de simplifier le traitement des services. La figure I.2.5 nous montre un diagramme de collaboration d'une configuration d'un service ESB stan-

dard, ce diagramme nous montre l'échange de messages entre les différentes classes: Classe Distributeur, Classe Adresse ESB, Classe Service ESB, Classe Points d'accès ESB.

Chaque service contient un ou plusieurs Distributeurs de messages, il accepte les messages qui proviennent à un service donné, il extrait son adresse, traite le contenu des messages, extrait les adresses de retour et invoque le service.

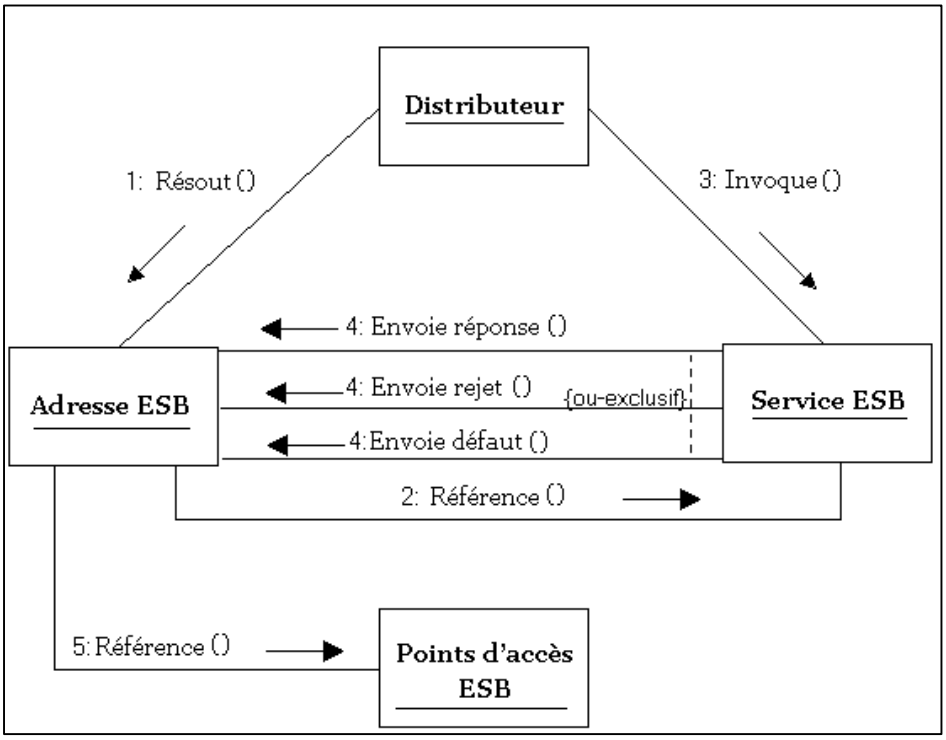


Figure I.2.5 Diagramme de collaboration d'une configuration d'un service ESB standard.

Le Service ESB envoie une réponse au service d'Adresse ESB. La réponse peut aussi être un message de rejet ou de défaut.
Le service Adresse ESB va donner référence à un distributeur pour transmettre le message au bon Point d'accès ESB référencé par le distributeur.

I.2.6 Les protocoles et les normes

En plus des différents protocoles et normes qui permettent de bâtir des Web-Services telles que (UDDI, WSDL, SOAP, XML, http, TCP/IP). Une architecture SOA pourra être complétée par :

La gestion de la sécurité avec : SSL (Secure Sockets Layer), XML Signature, XML Encryption, SAML (Security Assertion Markup Language) ou encore XKMS (XML Key Management Specification, qui gère les infrastructures à clé publique ou PKI)

L'orchestration (on parle également de chorégraphie) des services pour constituer des processus métier avec : BPEL4WS (Business Process Execution Language For Web Services) devenu WS-BPEL et qui est dérivé à la fois de WSFL (Web Services Flow Language) d'IBM et de XLang de Microsoft qu'il a remplacé, devenant de fait le standard de l'orchestration des Web-Services. On peut aussi citer WSCI (Web Services Choregraphy Interface). L'orchestration suppose l'existence d'un chef d'orchestre (WS-BPEL est un langage d'orchestration), tandis que la chorégraphie implique des interactions pair-à-pair. WS-CDL (Web Services Choregraphy Description Language) est un exemple, le plus récent, d'un tel langage.

La gestion transactionnelle (gestion du protocole de validation à deux phases, two-phase commit, pour la mise à jour contrôlée de plusieurs bases de données réparties entre plusieurs fournisseurs de services : la transaction « attend » de recevoir l'acquittement (le commit) des différents serveurs sollicités et en cas de problème avec l'un d'eux, est en mesure de demander aux autres serveurs de « défaire » les mises à jour partielles effectuées afin de maintenir l'intégrité des données) : WS-Transaction d'IBM, XAML (Transaction Authority Markup Language) ou encore BTP (Business Transaction Protocol).

I.2.7 Avantages des SOA

La mise en œuvre d'une approche SOA au sein d'une infrastructure informatique fournit des avantages considérables, parmi ces avantages:

Réduction des coûts de développement : Les services SOA sont aisément réutilisables et peuvent être rapidement assemblés, tels des composants, sous la forme de nouvelles applications composites sans nécessiter aucune programmation manuelle coûteuse.

Allégement des coûts de maintenance : Les services réutilisables diminuent le nombre et la complexité interne des services informatiques, réduisant ainsi la charge de travail de maintenance requise pour l'intégralité de la gamme de services.

Optimisation de la qualité des services : La technologie SOA favorise considérablement la réutilisation accrue des services, laquelle se traduit par une amélioration de la fiabilité et de la qualité des services par le biais de multiples cycles de test effectués par différents utilisateurs.

Diminution des coûts d'intégration : Les services normalisés fonctionnent parfaitement les uns avec les autres, garantissant ainsi la connexion rapide et aisée d'applications disparates.

Minimisation des risques : Des services moins nombreux et réutilisables contribuent à diminuer les risques globaux de non-conformité des opérations.

Adaptabilité aux changements : La possibilité de mettre en place de nouveaux processus métier ou par la composition des processus métier existants facilite la mise en œuvre de solution répondant à un nouveau besoin métier.

I.2.8 Conclusion

Dans cette partie, nous avons essayé de donné les étapes de cycle de vie d'un Web-Service depuis sa création jusqu'à sa consommation, supportées par des standards (SOAP, WSDL, UDDI) qui sont basés sur la norme XML, ce qui assure l'interopérabilité des applications les supportant sur Internet.

L'apparition des standards comme les Web-Services dans l'e-commerce favorise la popularité des architectures SOA.
Nous avons ensuite donné les concepts de base et les principes de fonctionnement d'une SOA. Nous avons constaté que ce type d'architecture est apparu comme remède aux problèmes d'hétérogénéité des systèmes informatiques distribués et particulièrement en optant sur les ESB comme outils d'intégrations.

Partie II
Les Systèmes Multi Agents et coopération

II.1 Notion d'agent informatique

Il n'existe encore pas de définition commune pour la notion d'agent ceci peut être expliqué par l'existence de différents domaines et communautés avec différentes perspectives. Néanmoins, on peut donner quelques définitions pertinentes:

« Un agent est une entité virtuelle ou réelle qui est capable d'agir sur son environnement, qui possède des moyens de perception et de représentation partielle de son environnement, qui est capable de communiquer avec d'autres agents et qui est autonome dans sa prise de décision »[12]. Son comportement autonome est la conséquence de ses observations, de ses compétences et des interactions avec les autres agents. [17]

Un agent est un système informatique, situé dans un environnement, et qui agit d'une façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu. [18]

Un agent est une entité réelle ou virtuelle dont le comportement est autonome, évoluant dans un environnement qu'il est capable de percevoir et sur lequel il est capable d'agir, et d'interagir avec les autres agents. [28]

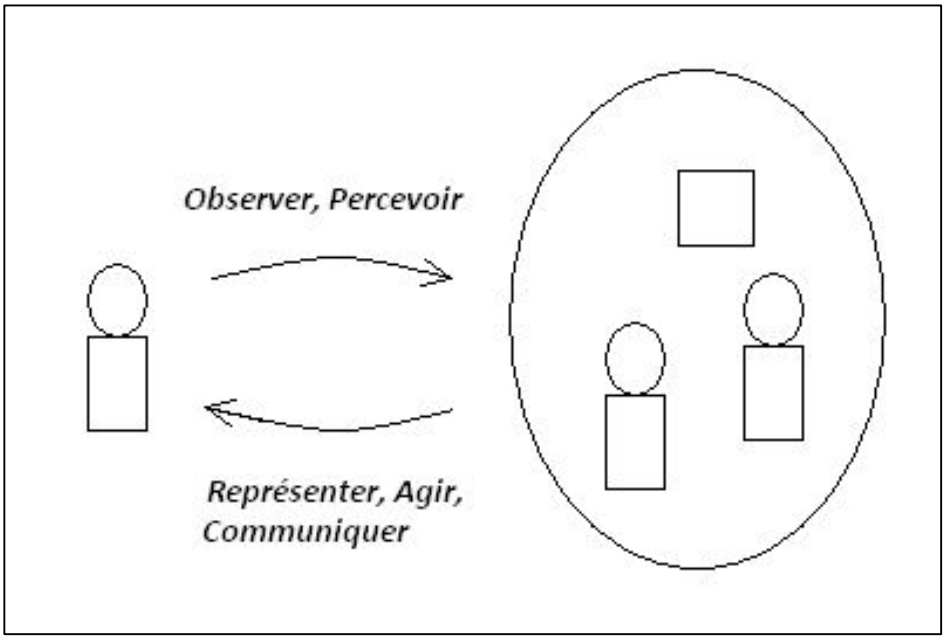


Figure II.1 Interaction agent – environnement

On peut définir un agent par une entité autonome pouvant observer sont environnement voisin ainsi que d'autres agents qui s'y trouvent. Et ayant la possibilité de représenter et agir sur cet environnement. (cf. figure II.1).
Ces définitions introduisent certains concepts et caractéristiques d'agents.

II.2 Caractéristiques fondamentales d'agent

Autonomie : un agent agit sans l'intervention des humains ou des autres agents, et il contrôle ses actions en fonction de son état interne et de son environnement. [18]

Communication : un agent est capable de communiquer avec d'autres.

Interaction : un agent peut avoir une influence locale sur le comportement des autres agents. Les agents peuvent se rencontrer, communiquer, collaborer, coopérer et agir les uns sur les autres.

Flexibilité : la flexibilité englobe les capacités suivantes :

Réactivité: un système réactif maintient un lien constant avec son environnement et répond aux changements qui y surviennent.

Pro-activité: un système pro-actif génère et satisfait des buts. Son comportement n'est donc pas uniquement dirigé par des événements.

Capacités sociales: un système social est capable d'interagir ou coopérer avec d'autres systèmes.

Suivant les différentes capacités d'agent on peut avoir trois types d'agents.

II.3 Types d'agents informatiques

II.3.1 Agent réactif

Les agents ne possèdent pas de moyen de mémorisation et n'ont pas de représentation explicite de leur environnement : ils fonctionnent selon un modèle stimulus/réponse. En effet, dès qu'ils perçoivent une modification de leur environnement, ils répondent par une action programmée.

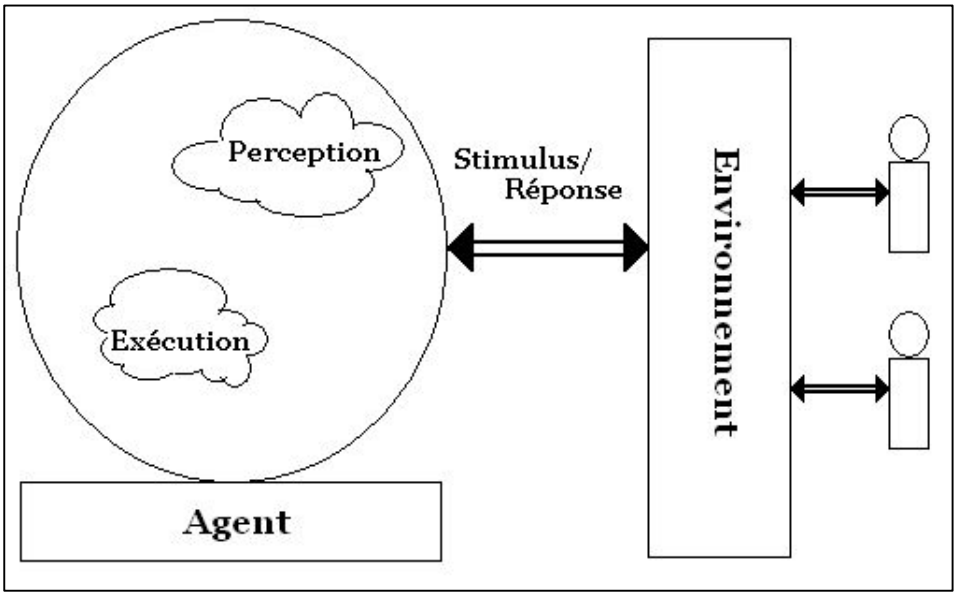


Figure II.2 Agent réactif

II.3.2 Agent cognitif ou intentionnel

Les agents sont dotés d'une forme d'intelligence, car ils se basent sur leurs historiques pour prendre des décisions [08]. Ces agents possèdent une représentation explicite de leur environnement, des autres agents et d'eux-mêmes.

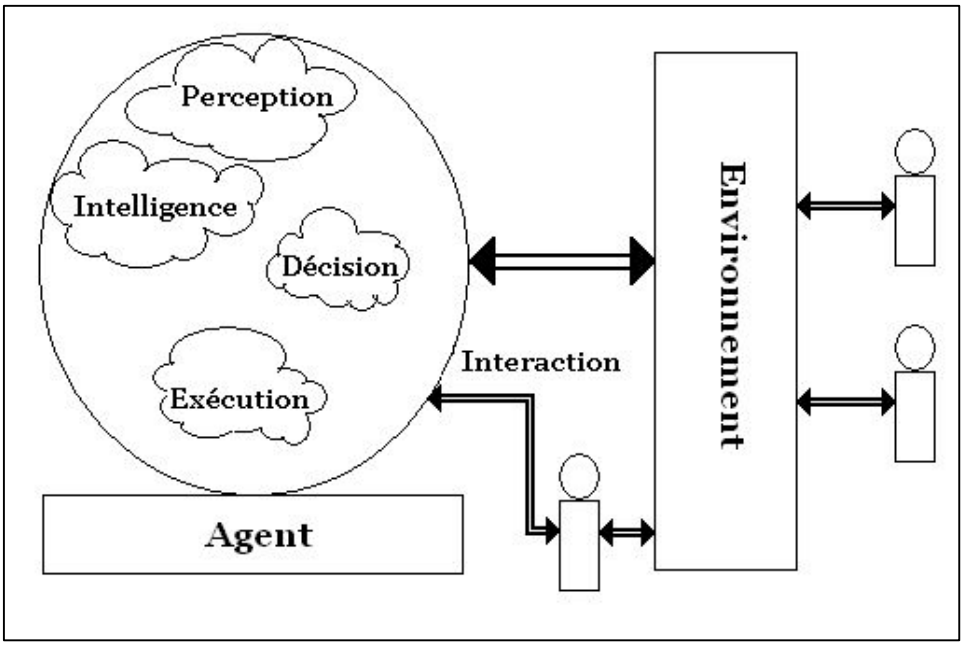


Figure II.3 Agent cognitif

II.3.3 Agent hybrides

Les agents ont des capacités cognitives et réactives. Ils conjuguent en effet la rapidité de réponse des agents réactifs ainsi que les capacités de raisonnement des agents cognitifs. [21]

II.4 Système multi agents

II.4.1 Définitions

Un système multi-agents SMA est un ensemble d'entités (physiques ou virtuelles) appelées agents partageant un environnement commun (physique ou virtuel), qu'elles sont capables de percevoir et sur lequel elles peuvent agir. Les perceptions permettent aux agents d'acquérir des informations sur l'évolution de leur environnement, et leurs actions leur permettent entre autres de modifier l'environnement. Les agents interagissent entre eux directement ou indirectement, et exhibent des comportements corrélés créant ainsi une synergie permettant à l'ensemble des agents de former un collectif organisé. [22]

On appelle système multi-agent (ou SMA), un système composé des éléments suivants: [12]

1. Un environnement E, c'est-à-dire un espace disposant généralement d'une métrique.

2. Un ensemble d'objets O. Ces objets sont situés, c'est-à-dire que, pour tout objet, il est possible, à un moment donné, d'associer une position dans E. Ces objets sont passifs, c'est-à-dire qu'ils peuvent être perçus, créés, détruits et modifiés par les agents.

3. Un ensemble A d'agents, qui sont des objets particuliers ($A \subseteq O$), lesquels représentent les entités actives du système.

4. Un ensemble de relations R qui unissent des objets (et donc des agents) entre eux.

5. Un ensemble d'opérations Op permettant aux agents de A de percevoir, produire, consommer, transformer et manipuler des objets de O.

6. Des opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on appellera les lois de l'univers.

II.4.2 Caractéristiques des SMA

Il existe des caractéristiques propres aux SMA, par rapport aux autres systèmes informatiques. Nous allons compléter les caractéristiques d'agents données en II.2 par la liste suivante :

Distribution : le système est modulaire, l'élément de base étant l'agent.

Décentralisation : les agents sont indépendants, il n'y a pas de décisions centrales valables pour tout le système.

Organisation : les interactions créent des relations entre les agents, et le réseau de ces relations forme une organisation qui peut évoluer au cours du temps.

Situation dans un environnement : les agents sont ancrés dans un environnement, source de données, de contraintes et d'incertitude, lieu d'actions et d'influences entre agents. L'évolution du SMA est la combinaison des évolutions des agents et de l'environnement.

Ouverture : le système échange des informations avec l'extérieur, des agents peuvent entrer et sortir du SMA ou encore être modifiés en cours d'évolution.

Émergence : Dans tous les SMA, une fonction globale est attendue à partir d'un ensemble de spécifications au niveau local de chacune des entités. Cette propriété du niveau global n'est pas programmée dans les agents et n'existe que par leurs interactions conduisant à des processus permanents de réorganisation.

Adaptation : il est impossible de spécifier le but global et d'organiser les agents pour l'atteindre, ou même de prouver que le SMA réalise effectivement une fonction globale adéquate. Mais le système adapte son comportement à l'environnement en cours de fonctionnement, et offre une robustesse de ce comportement, à défaut d'une optimisation.

Délégation : l'utilisateur accepte de ne pas maîtriser le comportement de l'application globale, à défaut de pouvoir supporter la complexité liée à l'ensemble des décisions prises par les agents dans le système. Il délègue une partie du contrôle de l'application globale aux agents.

Personnalisation : lorsqu'un agent représente un utilisateur, typiquement dans un SMA appartenant à la famille des systèmes intégrés dans un contexte plus large, il s'adapte à lui.

Intelligibilité : les SMA proposent une manière naturelle de modéliser d'autres systèmes ou de mettre en œuvre des applications, ce qui les rend simples à appréhender pour un utilisateur extérieur.

II.4.3 Architecture des SMA

Il existe deux types d'architectures : [23]

II.4.3.1 SMA à contrôle centralisé ou à base de tableau noir

Ce type d'architecture permet aux agents de communiquer indirectement et d'interagir entre eux et ceci en se partageant l'espace de travail qui est le tableau noir.

Ce type d'architecture est composé de trois éléments (cf. figure II.4):

- Les connaissances : qui sont représentées par des agents
- Un mécanisme de Contrôle : concerne les contraintes sur les relations entre les conversations des protocoles qui régissent le système, et aux quelles l'agent peut participer simultanément ou successivement. [24]
- Tableau noir : Le tableau noir est un espace de travail partagé par les différents agents. Chacun peut le consulter, il peut également utiliser ou modifier ses objets.

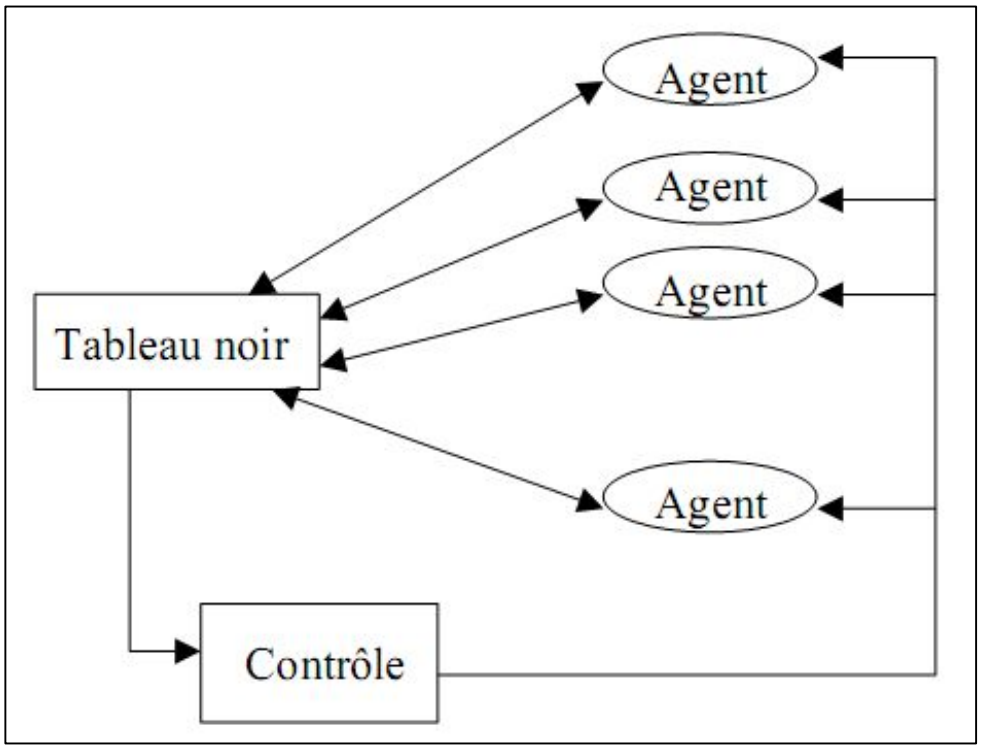


Figure II.4 SMA à base de tableau noir

En passant vers des SMA plus complexes (accroissement du nombre d'agents) le partage d'un espace commun de travail va devenir de plus en plus difficile. Dans ce cas on ne peut pas adopter une architecture à base de tableau noir mais on doit passer vers un autre type d'architecture.

II.4.3.2 SMA à contrôle distribué

Ce type d'architecture est basé sur la distribution totale des connaissances et du contrôle. Les agents effectuent un traitement local et communiquent par envoi de messages.
Dans ce cas, un Système Multi-Agents peut être vu comme ensemble de membres interdépendants qui agissent individuellement, tout en coopérant pour atteindre un objectif commun. [25]

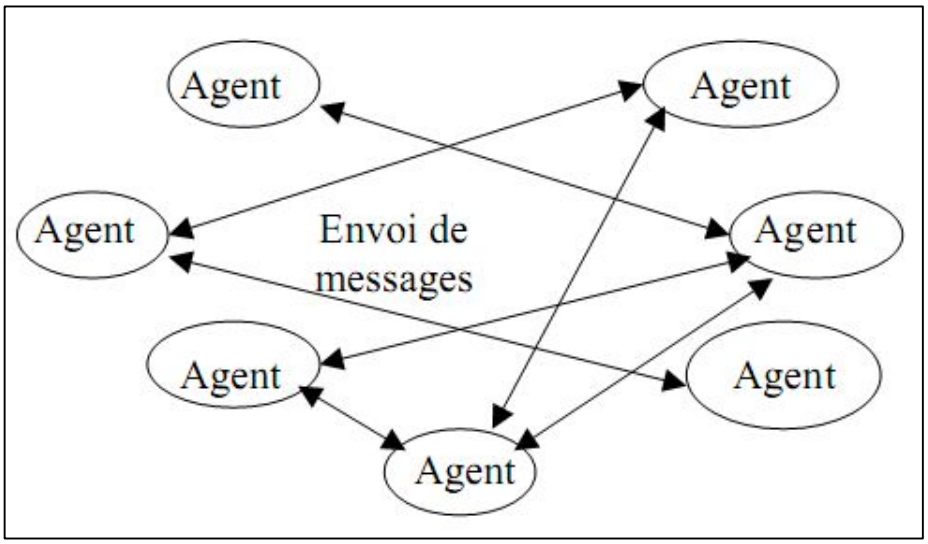


Figure II.5 SMA à contrôle distribué

II.5 Mécanisme de coopération

II.5.1 Définitions

On parle de coopération entre agents lorsqu'ils exercent des actions communes pour la réalisation d'un but commun.

On dira que plusieurs agents coopèrent, ou encore qu'ils sont dans une situation de coopération, si l'une des deux conditions est vérifiée [12]:
1. L'ajout d'un nouvel agent permet d'accroître les performances du groupe.
2. L'action des agents sert à éviter ou à résoudre des conflits potentiels ou actuels.

II.5.2 Fonctions de la coopération

Il est possible de caractériser trois grandes fonctions de la coopération : qui sont l'amélioration de la survie, l'accroissement de performances et la résolution de conflits.

L'amélioration de la survie : Qui reflètent la capacité d'un individu ou d'un groupe à maintenir son fonctionnement. On peut distinguer deux types de survie: la survie de l'individu et la survie collective qui correspond au maintien du groupe.

L'accroissement de performances : c'est l'amplification de la capacité d'un agent ou d'un groupe d'agents à accomplir leurs tâches ce qui augmente leurs performances. Les indices de performance sont variés, mais ils doivent traduire les caractéristiques intrinsèques d'un collectif.

La résolution de conflits : Du fait de leur autonomie, les agents sont amenés à se trouver dans des situations où leurs intérêts peuvent être contradictoires. Ces situations proviennent essentiellement d'un problème d'accès à des ressources limitées. Les situations conflictuelles naissent d'un manque de ressources et nécessitent des interactions supplémentaires pour sortir de ces conflits, qu'il s'agisse de techniques de négociation, d'arbitrage ou d'utilisation d'un règlement.

II.5.3 Les méthodes de coopération

Ceux sont les moyens que l'on peut mettre en œuvre pour réaliser la coopération, et sont classés en six méthodes [12]:

Le regroupement et la multiplication : Elle consiste pour les agents à se rapprocher physiquement, c'est-à-dire à constituer soit un bloc plus ou moins homogène dans l'espace, soit un réseau de communication permettant à plusieurs agents de se comporter comme s'ils étaient physiquement les uns à côté des autres.

La communication : Elle constitue l'un des moyens fondamentaux pour assurer la répartition des tâches et la coordination des actions entre agents. Les communications sont indispensables à la coopération.
Dans les systèmes cognitifs, les communications s'effectuent par envois de messages, alors que dans les systèmes réactifs, elles résultent de la diffusion d'un signal dans l'environnement.

La spécialisation : c'est le processus par lequel des agents deviennent de plus en plus adaptés à leurs tâches. La réalisation performante d'une tâche suppose souvent des caractéristiques structurelles et comportementales qui ne peuvent permettre d'effectuer d'autres tâches avec efficacité.

La Collaboration par partage des tâches et des ressources : Elle consiste à travailler à plusieurs sur un projet, une tâche commune. C'est l'ensemble des techniques permettant à des agents de (se) répartir des tâches, des informations et des ressources de manière à réaliser une œuvre commune.

La coordination d'actions : Pour qu'un ensemble d'agents autonomes poursuivent la réalisation de leurs buts, ainsi que les tâches productives ils doivent réaliser des tâches de coordination. La phase de coordination

d'actions est impliquée dans la définition de l'ordre des actions à effectuer.

La résolution de conflit par arbitrage et négociation : L'arbitrage et la négociation sont deux des moyens utilisés par les systèmes multi-agents pour résoudre les conflits. L'arbitrage conduit à la définition de règles de comportement qui agissent comme des contraintes sur l'ensemble des agents, mais dont le résultat global a pour effet de limiter les conflits et de préserver à la fois les individus mais surtout les sociétés d'agents.

Lorsque des agents cognitifs entrent en conflit d'objectif ou de ressource, on préfère souvent ne pas recourir à un arbitre, mais laisser plutôt les agents résoudre eux-mêmes le conflit par la recherche d'un accord bilatéral au cours d'un processus de négociation.

II.6 L'approche Voyelle A-E-I-O

Voyelles [Dem95] est la première méthodologie développée par l'équipe MAGMA qui propose une décomposition du système multi-agents en cinq vues : Agent, Environnement, Interaction, Organisation et Utilisateur. D'autres méthodologies telles que INGENIAS et MASSIVE, proposent d'autres décomposition multi-agents. Toutes ces méthodologies se ressemblent dans leurs décompositions. Elles ont en commun les aspects : agent, environnement, interaction et organisation. [26]

Agents : qui concernent les modèles (ou les architectures) utilisés pour la partie active de l'agent, depuis un simple automate à un complexe système à base de connaissances.

Environnements : qui sont les milieux dans lesquels sont plongés les agents. Ils sont généralement spatiaux dans la plupart des applications multi-agents.

Interactions : qui concernent les infrastructures, les langages et les protocoles d'interactions entre agents, depuis de simples interactions physiques à des interactions langagières par actes de langage.

Organisations : qui structurent les agents en groupes, hiérarchies, relations, etc.

II.6.1 Interaction

On peut la définir par la mise en relation de 2 ou plusieurs agents par le biais d'un ensemble d'actions réciproques, ce qui va constituer une organisation.

"C'est parce qu'ils coopèrent que les agents peuvent accomplir plus que la somme de leurs actions, mais c'est aussi à cause de leur multitude qu'ils doivent coordonner leurs actions et résoudre des conflits." [12]

Un SMA est un ensemble d'agents en interaction qui est un aspect très important dans les SMA. Plusieurs protocoles et langages d'interaction ont été proposés pour définir les interactions entre agents. Les protocoles d'interaction s'intéressent aux messages échangés entre des agents; ce que l'on appelle des conversations.

Langages d'interaction : Depuis plusieurs années, la communication dans les SMA cherche à s'inspirer des phénomènes de communication humains. En se basant sur le formalisme des actes de langages (ensemble des actions intentionnelles effectuées au cours d'une communication), des langages d'interactions ont été développés tels que KQML et FIPA-ACL. [27]

Protocoles d'interaction : ils permettent de donner le schéma de conversation suivi par les agents impliqués pour l'exécution d'une tâche. Ce qui facilitera leur dialogue. La modélisation peut se faire suivant plusieurs formalismes : Automates à états finis, Réseaux de Pétri…

Protocoles de coopération : Les protocoles de coopération suivent généralement une stratégie qui consiste à décomposer les tâches en sous-tâches puis à les distribuer (allocation) entre les différents agents. Ce qui va renduire la complexité des tâches en termes de temps (les sous tâches réalisent leur but commun en un temps réduit même avec des compétences inférieures) ainsi qu'en termes de coût (utilisant peu de ressources). Après la décomposition de la tâche, les sous tâches doivent être réparties sur différents agents. L'allocation des tâches peut être centralisée (hiérarchique ou égalitaire) ou distribuée (réseau d'accointances, appel d'offre par exemple le Contract Net).

Principe du contract net : [33] Fondé sur la procédure d'attribution des marchés publics.

– Les agents peuvent jouer deux rôles :
Manager (client, administrateur) ou Bidders (fournisseur, offrant).

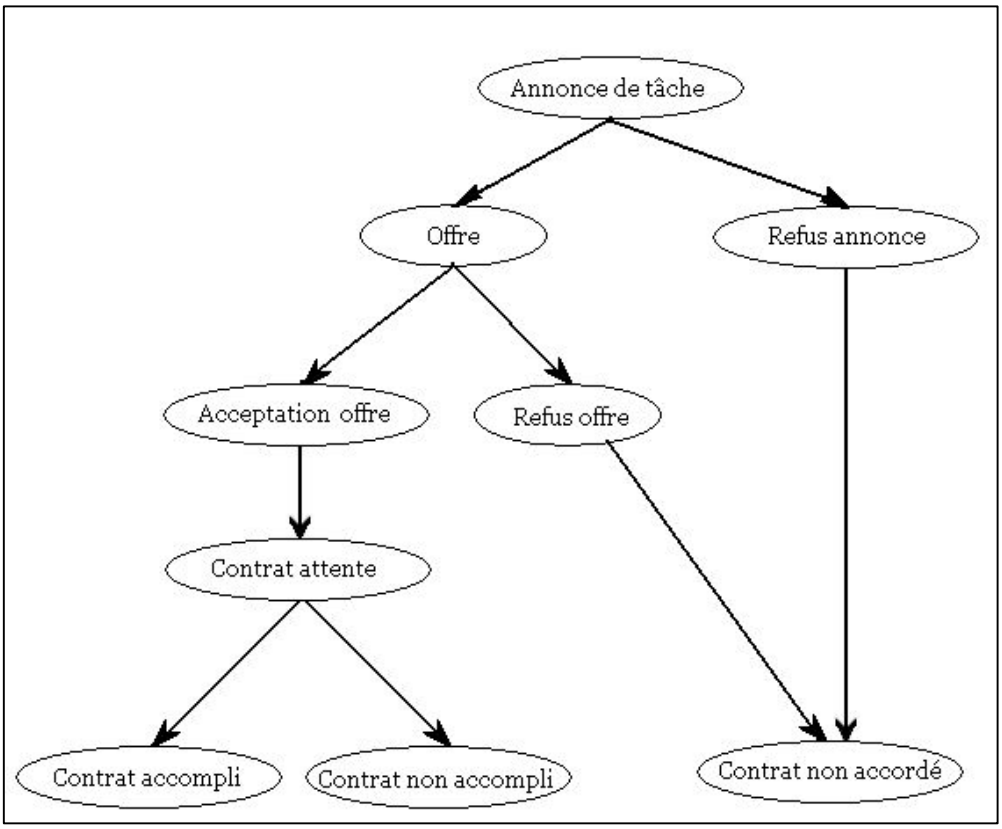


Figure II.6 Graphe de causalité entre les états d'un contract net

– Le manager ayant une tâche à (faire) réaliser, décide de la sous-traiter, sélectionne des contractants potentiels et leur soumet un appel d'offre.
– L'agent ayant soumis la meilleure proposition de réponse à l'appel d'offre est sélectionné et réalise la tâche sous le contrôle du manager. (cf. figure II.6). [34]

II.6.2 Organisation

Une organisation peut être définie comme un agencement de relations entre composants ou individus qui produisent une unité, ou système, dotée de qualités inconnues au niveau des composants ou individus. [12]

Les organisations constituent donc à la fois le support et la manière dont se passent les interactions entre agents, c'est-à-dire la façon dont sont réparties les tâches, les informations, les ressources, et la coordination d'actions.

Les organisations des systèmes multi-agents sont souvent décrites selon deux vues différentes [29] :

La description structurale, qui regroupe les relations entre agents (i.e. les possibilités d'interactions), les rôles et les groupements attribués aux agents, et qui décrit ainsi, souvent sous forme de graphes, la forme du système, Ce qui nous donne la structure organisationnelle.

La description fonctionnelle, qui regroupe les tâches attribuées à chaque rôle, et donc à chaque agent, leur participation dans le fonctionnement global du système, et l'intérêt pour le système des interactions mises en jeu.
On peut dire qu'une organisation est l'ensemble de rôle (d'agents) et de relations entre ces rôles. [30]

La construction de structures organisationnelles peut suivre plusieurs approches :

Approche descendante : organisation prédéfinie (qui constitue un modèle basé sur la définition de rôles et de structure organisationnelle ou de schémas de communication à priori, elle est définie par le concepteur, les relations sont déterminées à l'avance).

Approche ascendante : organisation émergente, absence de structure organisationnelle prédéfinie, les rôles et relations apparaissent comme le produit des comportements de chacun des agents selon un processus d'auto organisation. Cette approche est dédiée aux agents réactifs.

II.6.3 Méthodologie AGR (Aalaadin)

Elle constitue un méta-modèle organisationnel qui situe l'analyse de systèmes multi-agents à deux niveaux :

Le niveau descriptif correspond aux concepts primaires d'agent, de groupe et de rôle. C'est à ce niveau que se décrit une organisation multi-agent réelle.

Le niveau méthodologique définit l'ensemble des rôles possibles, spécifie les interactions et décrit les structures abstraites de groupe et d'organisation.

Les étapes de la méthodologie AGR :

Identification des groupes et rôles : en regroupant les agents similaires et en identifiant leurs structures fonctionnelles. (Diagramme plateau à fromage (cf. figure II.7))

Construire la structure organisationnelle : en utilisant les modèles d'enchères, e-commerce… (Diagramme de structure organisationnelle)
Décrire la dynamique de l'organisation : en utilisant des règles de gestion de groupes (création, adhésion, suppression…) (Diagramme organisationnel de séquence)

Définir les rôles : ce qui nous donne l'aspect fonctionnel (méthode Gaia). Différents formalismes Ex : attributs + diagramme d'états.

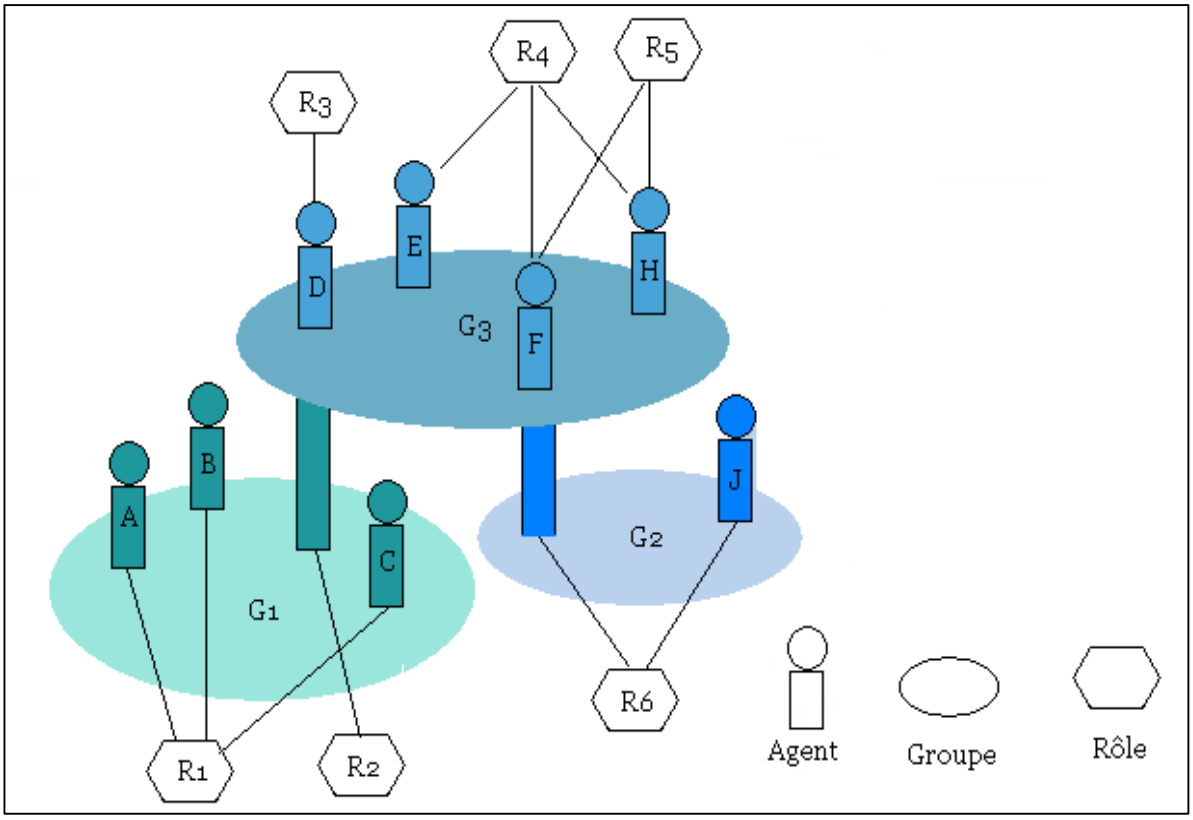


Figure II.7 Diagramme plateau à fromage

II.7 Interopérabilité des SMA dans des architectures SOA

L'interopérabilité des SMA a été promue par différentes institutions dont la FIPA qui déploie d'énormes efforts afin de proposer des normes et des standards qui assurent une interopérabilité à différents niveaux des architectures SMA.

Mais les limites des standards de la FIPA résident dans le fait que l'interopérabilité est conditionnée par la compatibilité des SMA avec les spécifications FIPA : au lieu d'assurer l'interopérabilité entre SMA hétérogènes, la FIPA tend à rendre tous les SMA compatibles (via ses spécifications) [31]. Ce qui rend difficile l'interopérabilité entre SMA hétérogènes.

Afin de maximiser l'intégration de systèmes hétérogènes et de favoriser une interopérabilité au moindre coût, il est nécessaire de minimiser les exigences en termes de spécifications.

Dans le monde du génie logiciel, et comme en témoignent les efforts de normalisation et de développements autour des Web-Services [32], ces derniers se présentent aujourd'hui comme un support crédible permettant à des applications d'exposer leurs fonctionnalités au travers d'interfaces standardisées et de plus en plus éprouvées. Les Web-Services ont pour vocation de favoriser une architecture orientée services, intégrant des systèmes hétérogènes complexes, fortement distribués et pouvant coopérer sans recourir à une intégration spécifique et coûteuse.

Avec les solutions d'intégration classiques point à point, les systèmes tombaient dans des situations d'interblocage, lorsque les applications sont complexes. D'où l'apparition de solutions EAI qui offrent une plateforme d'intégration d'application hétérogène rationalisant les flux d'information mais cette solution repose sur des technologies propriétaires chose qui limitaient l'interopérabilité et favorisait l'utilisation et l'émergence de l'approche ESB, qui reprend les principes de l'approche EAI, en se basant sur des standards [9]. Aujourd'hui on assiste à un rapprochement entre les SMA et les SOA avec l'adoption des ESB comme support de middleware de communication.

II.8 Conclusion

Dans cette partie, nous avons tenté d'introduire la notion d'agent et les systèmes multi agents, en donnant quelque définitions et caractéristiques qui nous serons utiles dans la suite de notre travail.

L'utilisation des SMA permet de simuler la nature, et la vie des sociétés tout en bénéficiant de ces formes de collaboration, coopération pour la résolution des problèmes complexes de communication qui peuvent exister entre les différents nœuds des réseaux informatiques.

Avec l'évolution des réseaux informatique actuels et l'augmentation de nombre et de nature des nœuds. L'association d'un mécanisme d'intégration et d'interopérabilité entres les différentes applications à la stratégie (mécanisme) SMA pourra remédier aux problèmes d'hétérogénéité et donner des résultats remarquable pour la coopération et l'intégration d'application. Ce que nous allons tenter de faire dans la partie suivante.

Partie III
« MAeMSOS » : une plateforme Multi Agents Orientée Services pour la e-Maintenance

III.1 Introduction

Le terme e-maintenance a été connu depuis le début de l'an 2000 comme une composante du concept de génie de la production[36], qui profite de l'émergence de l'information et des technologies de communication utilisées pour mettre en œuvre (implémenter) des environnements multi-utilisateurs coopératifs et distribués. La e-maintenance est un nouveau concept qui peut être défini comme un support de maintenance qui comprend les ressources, les services et la gestion nécessaires pour permettre l'exécution processus de décision proactifs. Ce support comprend e-technologies (TIC, basée sur le Web, sans fil, les technologies Infotronics), mais aussi les activités de e-maintenance (opérations ou de processus) telles que le e–monitoring, e-diagnostic, le pronostic ... [35]

III.2 La maintenance

Un système de maintenance est représenté par une application de fiabilité des différentes activités de la fonction de maintenance telles que logistique, planning des interventions, gestion des stocks (géré par la GMAO, ERP), diagnostic et réparation (systèmes experts, bases de données), surveillance d'un équipement (SCADA, commande numérique sur un équipement). L'architecture de ces systèmes peut varier selon les différents objectifs visés. [37] la figure III.1 est un schéma générique d'un système de maintenance, composé de deux parties principales, à savoir du système physique et du système de gestion.

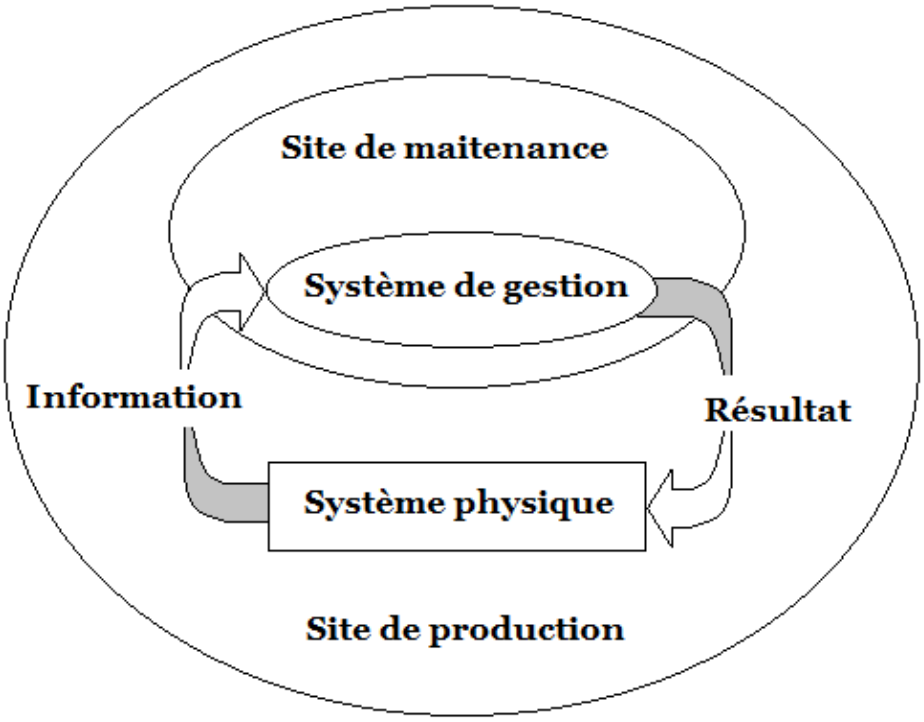


Figure III.1 Architecture d'un système de maintenance

Le système de gestion produit l'ensemble des résultats ou décisions en se basant sur les informations provenant du système physique [38]

III.3 Classification des stratégies de maintenance

Il existe différentes stratégies de maintenance selon le type d'entretien effectué : comme réponses aux échecs (correctives) ou comme des mesures préventives avant que les échecs ne surviennent (préventives).
Dans le cas de la maintenance corrective, les mesures sont engagées lorsqu'une erreur survient. La figure III.2 montre la taxonomie des différentes stratégies de maintenance.

L'objectif des stratégies préventives est de compléter les mesures de maintenance avant la production des erreurs ainsi que la prévention des temps d'arrêt qui peuvent survenir. Cette stratégie peut être réalisée au moyen de mesures basé sur le temps comme les activités d'entretien régulier, ou des mesures conditionnelles qui sont lancées en fonction du degré d'usure. [39]

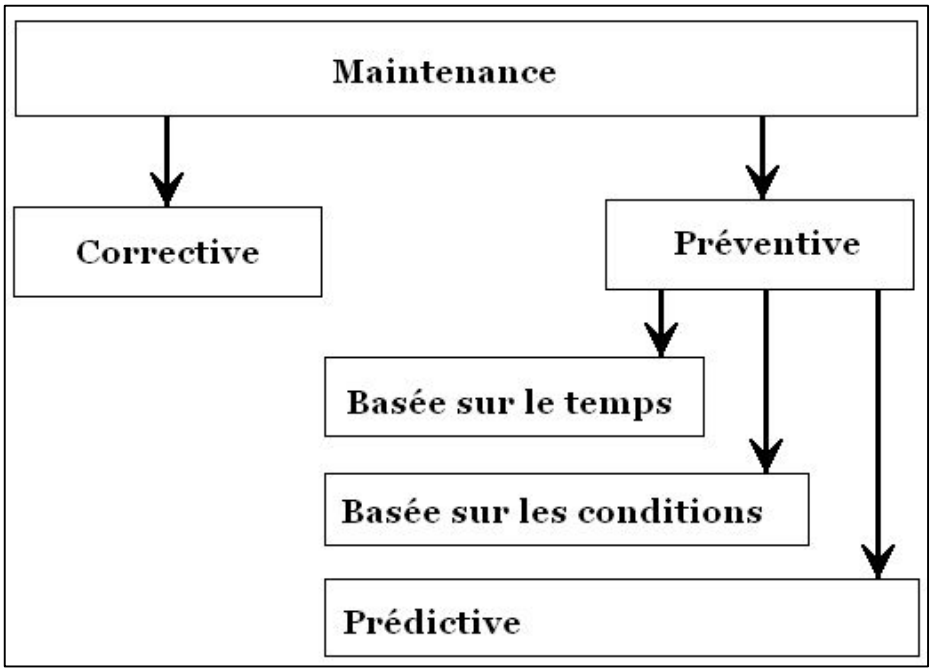


Figure III.2 Taxonomie des stratégies de maintenance

III.3.1 La maintenance préventive

La maintenance préventive (PM) peut être défini comme un programme dans lequel l'usure, et les changements sont prévus et des actions de correction continues sont prises pour maximiser l'efficacité et minimiser la détérioration. PM s'agit d'un programme planifié et contrôlé de l'inspection systématique, l'ajustement, la lubrification et le remplacement de composants, ainsi que des tests de performance et d'analyse. Le résultat d'un programme de PM prolonge la vie des installations et du matériel, et minimise les temps d'arrêt imprévu qui provoque des problèmes majeurs. Actuellement, PM est basé sur le temps, le contrôle des paramètres des équipements ou les sorties de production.

III.3.2 La maintenance basée sur les conditions

La maintenance basée sur les conditions (CBM) vise à réduire le nombre de défaillances de composants non planifiées en surveillant l'état du matériel pour prévoir les défaillances et les mesures correctives pouvant être prise avant l'apparition de défaillances. CBM améliore la disponibilité des installations, la qualité du produit, les exigences de sécurité,

et de rapport coût-efficacité des installations et réduit les coûts de maintenance, qui constituent une partie importante du budget de fonctionnement des entreprises de production. La méthodologie CBM est élaborée en tenant compte de la dégradation actuelle et son évolution. L'analyse de la dégradation est utilisée pour étudier l'évolution des caractéristiques physiques, ou les mesures de performance d'un système.

III.4 La e-maintenance

L'architecture d'e-maintenance se fait via un réseau web qui permet de coopérer, d'échanger, partager et de distribuer ces informations aux différents systèmes partenaires de ce réseau (cf. figure III.3). Le principe consiste à intégrer l'ensemble des différents systèmes de maintenance dans un seul système d'information. Les systèmes proposent différents formats d'information qui ne sont pas toujours compatibles pour le partage ce qui nécessite la coordination et la coopération entre les systèmes pour les rendre interopérables. [37]

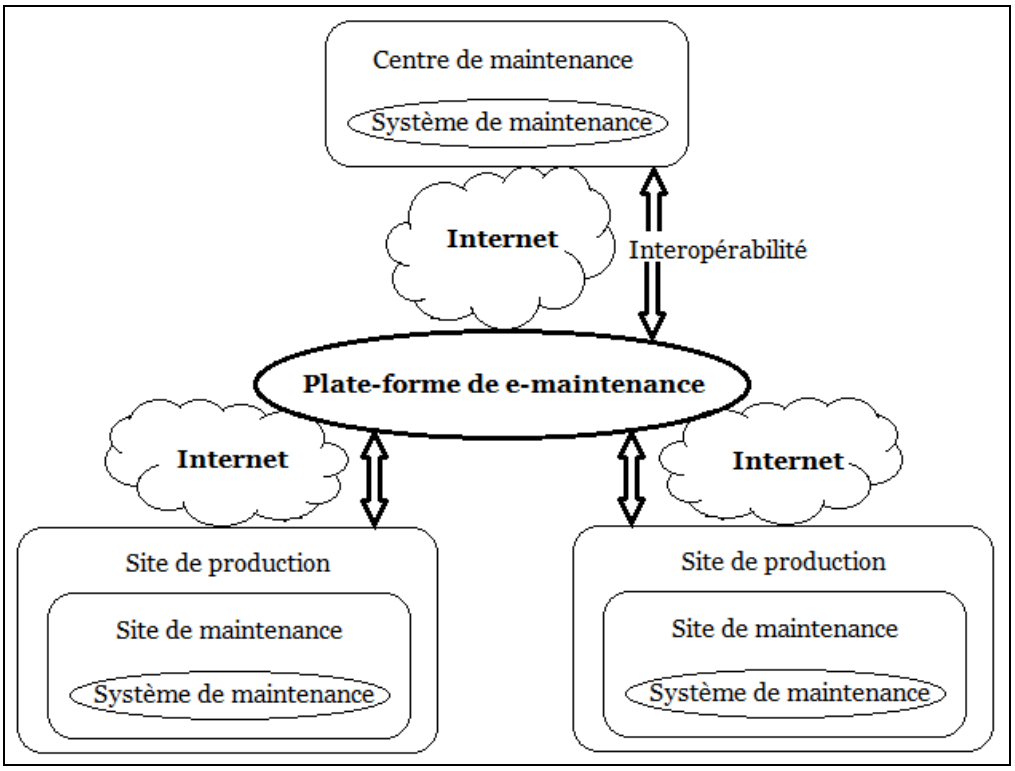


Figure III.3 Architecture du concept de e-maintenance

III.4.1 Les capacités de la e-maintenance

Les avantages offerts par la e-maintenance peuvent être classés suivant trois axes [40]:

• Offrir des opportunités pour le développement de nouveaux types et stratégies de maintenance;

• Amélioration des outils et supports de maintenance;

• Définir de nouvelles activités de maintenance.

III.4.1.1 Nouveaux types et stratégies de maintenance fournis par la e-maintenance

- ***Opérations de maintenance et la prise de décision à distance***:

  En misant sur l'information, les technologies sans fil et Internet, les utilisateurs peuvent se connecter de n'importe où et avec n'importe quel type d'appareils s'ils possèdent une connexion Internet et un navigateur. Ça leur permet de prendre des actions à distance, comme la configuration, le contrôle, le diagnostic, le suivi de la performance, et la collecte et l'analyse des données.

- ***L'intégration des processus métiers et la maintenance coopérative / collaborative :***

  Une plate-forme de e-maintenance peut introduire des niveaux de transparence et d'efficacité dans l'ensemble du secteur production, l'intégration des processus métiers qui contribue à l'accélération du processus, pour un modèle plus facile et pour synchroniser la maintenance avec la production, maximiser le débit processus et minimiser les coûts de temps d'arrêt.

- ***Une maintenance « on line » rapide :***

  Le monitoring à distance et à temps réel des états des équipements, permet à l'opérateur de maintenance de répondre à n'importe quelle situation, et de préparer une intervention optimale.

- ***Une maintenance prédictive :***

Une approche qui combine les différents outils des techniques de maintenance prédictive est un des avantages majeurs de la e-maintenance. Les applications dans ce domaine inclue les pronostiques des problèmes des équipements basés sur les conditions courantes, la durée de vie prédite et l'usage projeté.

III.4.1.2 Exemples d'améliorations possibles des supports et outils de la maintenance fournis par e-maintenance

- ***Analyse des défauts / échecs :***

Développement des technologies des capteurs, traitement du signal, les TIC et autres technologies liées à la surveillance et du diagnostic permettent d'améliorer la compréhension des causes des défaillances et des perturbations du système de maintenance.

- ***Documentation de maintenance :***

Par exemple, les informations comme la forme d'accomplissement d'une tâche peuvent maintenant être faite en premier par un utilisateur, puis envoyés à plusieurs clients (logiciels ou humains) qui ont enregistré de mêmes événements.

III.4.1.3 Améliorations des activités fournis par la e-maintenance

- ***Diagnostic/localisation des pannes :***

Un exemple clair est «E-diagnostic » qui offre aux experts la possibilité d'effectuer le diagnostic de pannes en ligne, partager leurs expériences avec les autres, et proposer des solutions aux opérateurs si une condition anormale se produit dans la machine à inspecter.

- ***Réparation et reconstruction :***

Par exemple, le temps d'arrêt pourrait éventuellement être réduit par une interaction directe (dépannage) des spécialistes ou concepteurs de l'équipement. D'autre part, le diagnostic, l'entretien à effectués et les pièces remplacées sont documentés par le biais des réponses structurées aux différents étapes de son fonctionnement.

III.5 Plates-formes de e-maintenance

III.5.1 La Plateforme PROTEUS

C'est une plateforme distribuée coopérative de e-maintenance incluant les systèmes existants d'acquisition de données, de contrôle commande, de gestion de la maintenance, d'aide au diagnostic, de gestion de la documentation, etc. Le concept de cette plateforme est défini par la description unique et cohérente de l'installation à maintenir (une ontologie), par l'architecture générique basée sur les concepts de Web-Services et en proposant des modèles et des solutions technologiques d'intégration.

Ces techniques permettent de garantir l'interopérabilité de systèmes hétérogènes afin d'assurer l'échange et le partage des informations, des données ainsi que des connaissances. Le but de la plateforme est non seulement d'intégrer des outils existants, mais aussi de prévoir l'évolution de ceux-ci au travers de l'introduction de nouveaux services [41].

Proteus est construit autour d'un noyau central appelé CSA (pour Central Service Application). Ce noyau contient différents services tels que le système de gestion des utilisateurs (CAAS pour Central Authentification Application Server), la base de lien modélisant le système (CORD pour Central Object Repository Data-link) le WorkFlow permettant de définir les conditions d'une succession d'appels aux différentes ressources.

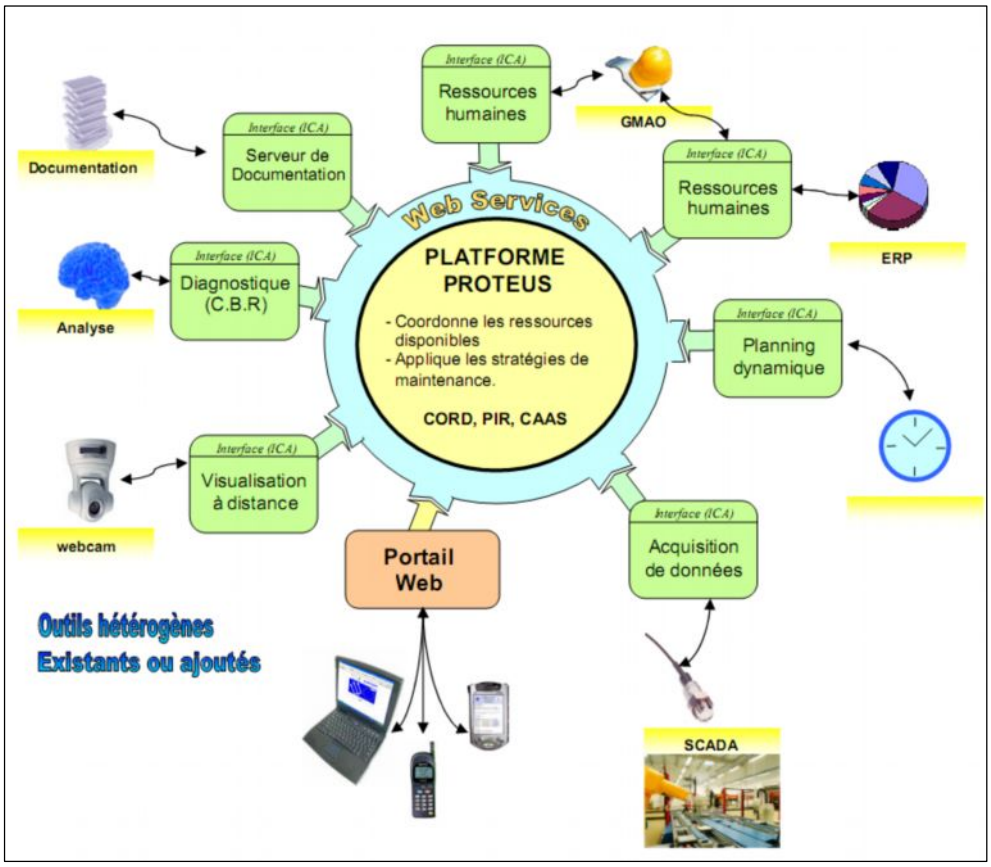


Figure III.4 Architecture de Proteus

Autour de celui-ci gravitent les ICA (Interface Core adapter) connectés directement aux outils réels. La communication entre le CSA et les différents ICA se fait par la technologie SOAP. Chaque ICA est un outil idéal. Par conséquent, un outil idéal peut être connecté à un ou plusieurs outils réels, comme par exemple la gestion des ressources humaines peut se faire grâce à un ERP et une GMAO. Chaque ICA est donc connecté aux applications spécifiques du site à gérer. C'est lui qui va rendre compatible les outils réels avec la plate-forme PROTEUS.

Classiquement, un équipement est relié à un système de d'acquisition d'information.

Un ICA est une application cliente qui va d'une part chercher les informations (par exemple sur des capteurs) via le système existant, et d'autre part, les rend disponible à la plate-forme PROTEUS.

La connexion entre l'ICA et l'outil réel se fait au cas par cas, utilisant les possibilités de l'outil. Cette structure est valable pour tous les ICA.
L'accès à la plate-forme se fait à travers un serveur Web dédié à PROTEUS. Il est possible de consulter les informations via un navigateur classique, un PDA ou un téléphone cellulaire.

III.5.2 La Plateforme ICAS

C'est une des premières e-maintenances développées. Le système d'évaluation de conditions intégré (ICAS) est désigné pour fournir un outil d'ingénierie informatisé pour l'implémentation d'une maintenance basée sur les conditions (CBM) à travers l'acquisition des données, les analyses d'experts pour assurer la disponibilité et la réutilisabilité. [42]

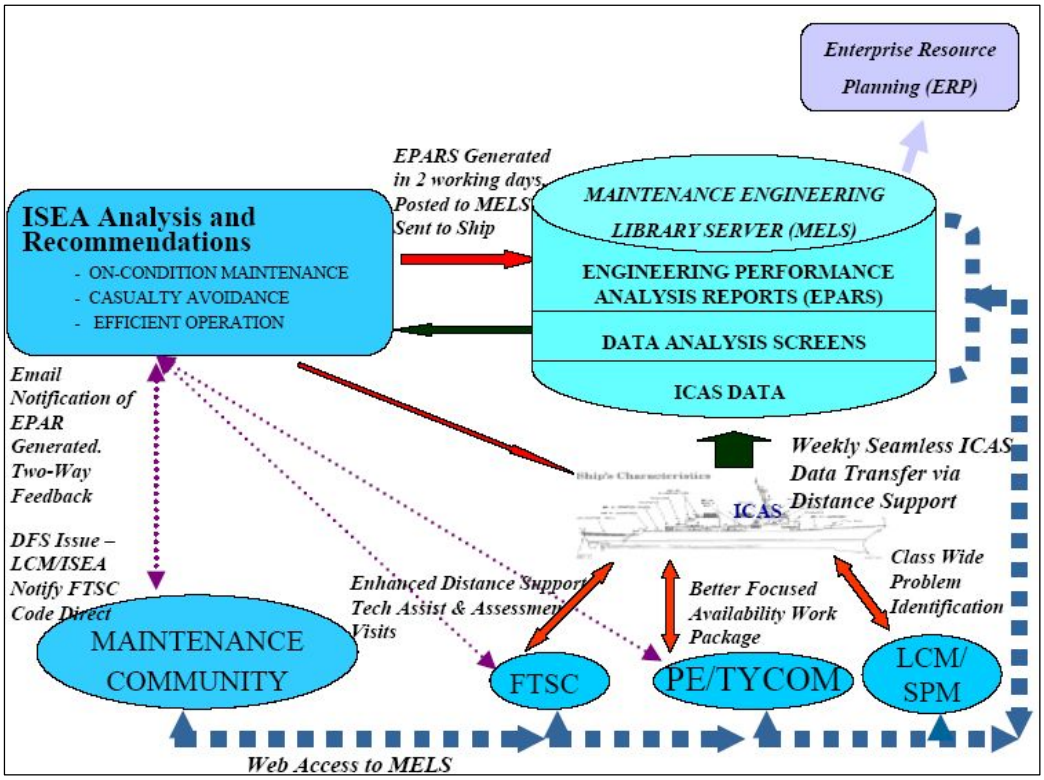


Figure III.5 Architecture de ICAS

III.6 La plateforme « MAeMSOS »

Notre approche intitulée «Multi Agent e-Maintenance Service Oriented System », a pour objectif, de concevoir une architecture Orientée Services basée sur les Systèmes Multi Agents pour la e-maintenance d'une organisation de production. Les différents acteurs de notre système sont des agents dont les comportements sont considérés comme des Web-Services.

L'approche « MAeMSOS » suit une stratégie de maintenance préventive basée sur les conditions (paragraphe III.3.2). Notre système est doté d'une base de connaissances des états normaux de fonctionnement des différents équipements, d'un système de suivi qui détecte toute anomalie dans le fonctionnement de ces équipements qui lance une requête de diagnostic de cause et ainsi le système utilise les ressources humaines, logicielles ou matérielles adéquates pour la résolution du problème.

Les agents sont des outils de solution autonomes qui coopèrent pour l'accomplissement des services demandés. Lors du lancement de n'importe quel service, un agent du SMA est désigné de manière dynamique.

Le système de e-maintenance est décomposé en plusieurs tâches pour l'accomplissement d'un but commun. Ces différentes tâches vont être distribuées sur les différents services. Ces services vont être réalisés par des SMA coopératifs, et peuvent communiquer via un bus ESB.

III.6.1 Architecture de « MAeMSOS »

Conceptuellement, la plateforme « MAeMSOS » est composée de plusieurs SMAs, chaque SMA peut être composé d'un ou de plusieurs agents. Le support software véhiculant l'information entre les différents SMAs est l'ESB dont l'entité de communication entre eux est le Web-Service par l'intermédiaire du middleware SOAP.

Un SMA est spécialisé dans un domaine. Par exemple, le SMA gestionnaire de maintenance est composé de plusieurs agents dont les rôles sont différents pour l'accomplissement d'un but commun qui est la gestion de maintenance. La communication intra-SMA se fait par des services locaux répondant aux besoins de la maintenance.

La communication inter-SMAs se fait via une interface composée de plusieurs Web-Services véhiculés sur l'ESB. L'environnement d'un agent est, à priori, l'ensemble d'agents situés dans son SMA quand

l'expertise demandée se trouve au sein d'un SMA auquel cet agent appartient. Quand cette expertise est absente en intra-SMA, le SMA fait recours à son environnement composé des autres SMAs via l'interface des différents Web-Services offerts par cet environnement et invoqué, via le Web, utilisant l'ESB (cf. figure III.6).

Afin d'expliquer la notion d'expertise intra-SMA, prenons le cas d'un agent humain réparateur d'un équipement. Ce dernier possède une expertise locale et peut faire appel à des agents locaux (humain ou software) afin de répondre à une situation donnée. Supposons maintenant qu'il a besoin d'une e-doc afin de comprendre la nature d'une panne. Comme il fait partie de cette e-plateforme, il peut invoquer un Web-service e-doc du SMA e-documentation via le Web.

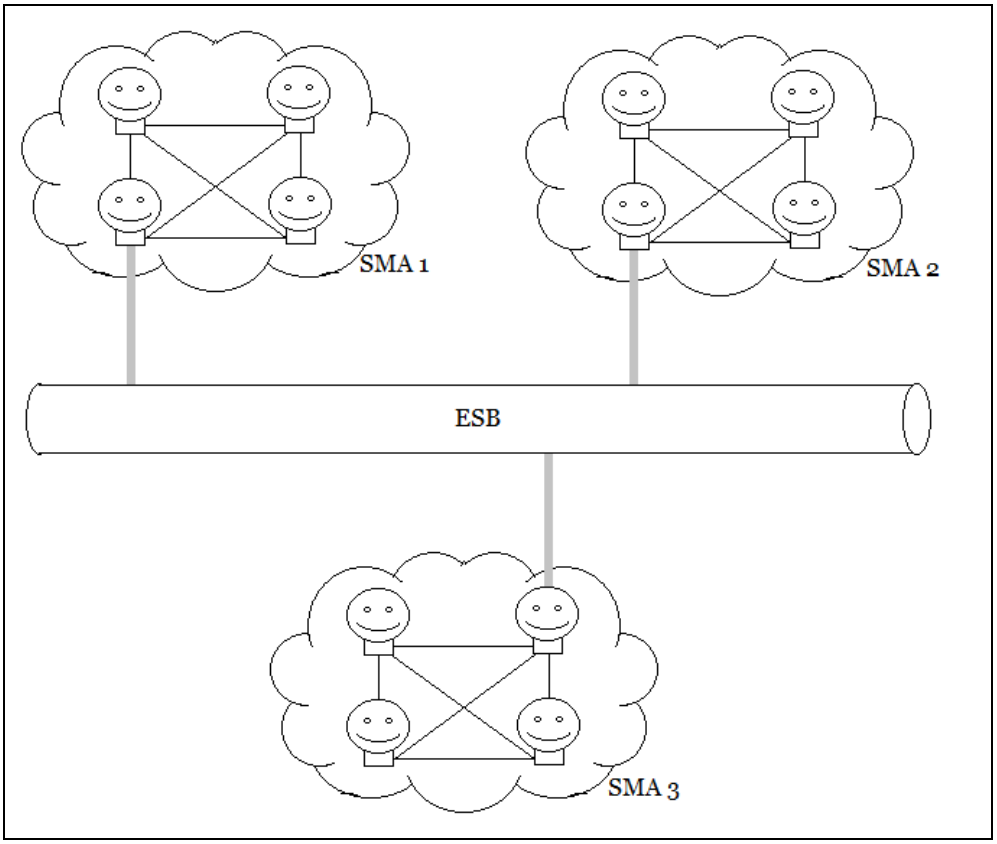


Figure III.6 Interopérabilité intra-SMA via ESB

Les fonctionnalités de la plateforme MAeMSOS sont construites donc autours de plusieurs services qui doivent être réalisés par chaque entreprise de production pour suivre la stratégie MAeMSOS de maintenance :

- Les outils existants (les Portail Web...)
- Le gestionnaire (qui se préoccupe de la gestion centralisée, il se base sur une base de données)
- Le module de Diagnostic (se base sur une base de connaissance)
- Module d'acquisition de données
- Module de e-documentation
- Module ressources humaines (clients, directeurs, gestionnaires, agents de maintenance, fonctionnaires).

Tous ces services sont réalisés par des SMAs coopératifs, chaque service est représenté par un ou plusieurs agents sur chaque site de production à maintenir.

Les différents agents vont s'intégrer à la plateforme MAeMSOS à l'aide d'un Bus ESB à travers des interfaces de Web-Services et vont donc communiquer via des messages SOAP (cf. figure III.7).

L'ESB se charge d'assurer la connectivité et le routage des différents services et agents avec un modèle d'échange de messages ainsi que les différents rôles qu'il déploie comme la découverte et l'invocation de services, la sécurité,…

III.6.2 Fonctionnement du « MAeMSOS »

III.6.2.1 Scénario

Le système « MAeMSOS » permet l'interconnexion et la coopération de différents types de composants qui peuvent ne pas été conçus pour fonctionner ensemble. Et ceci en les intégrant dans des architectures orientées service à l'aide de Web-Services, qui constitue des interfaces permettant à ces composantes et ressources de s'interconnecter au sain d'une SOA. Le système « MAeMSOS » permet de mutualiser un ensemble de ressources existantes de l'entreprise, par l'ajout d'interfaces Web-Services.

Nous nous plaçons dans le cas d'une entreprise détentrice des outils de maintenance suivants :

- Logiciels de gestion de ressources (GR): permettant de gérer les commandes, les fournisseurs, les clients, les différents équipements et toutes les factures d'une entreprise.

- Gestion de maintenance (GM) : qui permet de gérer des opérations et des ressources de maintenance.

- Système d'acquisition de données (SAD): c'est des capteurs qui permettent l'acquisition de donnée sur les équipements de l'entreprise et détecte ainsi les dysfonctionnements préalables (Webcam, scada, thermomètre…).

- Outils de Diagnostic (OD): qui permettent d'effectuer des Diagnostics, ce sont des systèmes de raisonnement qui permettent de détecter la cause d'un dysfonctionnement. Ils sont basés sur les conditions.

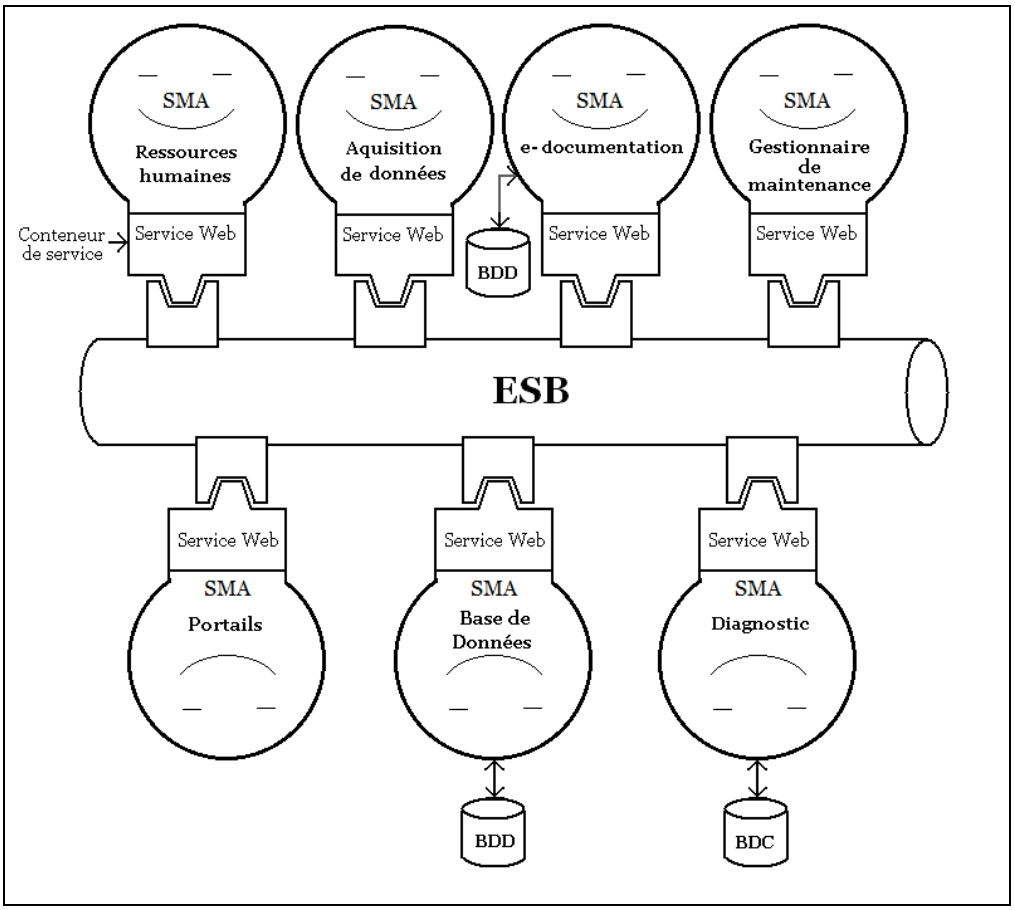


Figure III.7 Architecture de « MeAMSOS »

- E-documentation (EDOC) : Serveur de documentation sur un équipement accessible sur internet. Il peut fournir une documentation écrite sous forme doc et pdf, ou bien des vidéos.

Toutes ces ressources vont être connectées au bus ESB par des interfaces Web-Services (cf. Figure III.7). Le bus ESB doit avoir une base de descriptions de ces différents services qui constituent un annuaire sous la forme d'une Base de Données ainsi qu'un système de gestion des liens entre elles.

III.6.2.2 Explication du Scénario

Le système fonctionnel possède une stratégie de suivi des états, qui se base sur des SAD installés sur les différents équipements. Ce SAD est doté d'une base de connaissances des états normaux de l'équipement (température, bruit, rendements,...).

Lorsque le SAD détecte un dysfonctionnement ou une anomalie dans un équipement, il lance une alarme et demande d'intervention du système GM.

Le système GM va lancer les systèmes OD qui vont faire des analyses de données et un diagnostic de l'état de l'équipement. Une webcam peut être utilisée pour vérifier visuellement le problème à distance. Une fois la cause du problème déterminée, l'outil de GM va désigner un ensemble de ressources pour la résolution du problème, le système GR va déterminer la disponibilité des ressources qui peuvent être:

Des ressources humaines: intervention personnelle à distance (des vidéos on-line, explication directe entre deux personnes).
Des ressources matérielles (outils spécifiques et pièces de rechange)
Des ressources logicielles : Documentation électronique (vidéo, page Web, Document ou sous PDF), télémaintenance.

Le système GM va prendre la décision des actions à entreprendre avec l'aide d'un système d'aide à la décision, et va effectuer un plan d'action (cf. figure III.8)

Lorsque le processus de maintenance termine, le système GM clôture l'intervention, et les informations concernant cette intervention sont enregistrées sous forme d'un rapport dans la base du système GM.
Toutes les interconnexions sont réalisées via le bus de Services d'Entreprise ESB. Et tout les Services sont réalisés par des Systèmes Multi Agents.

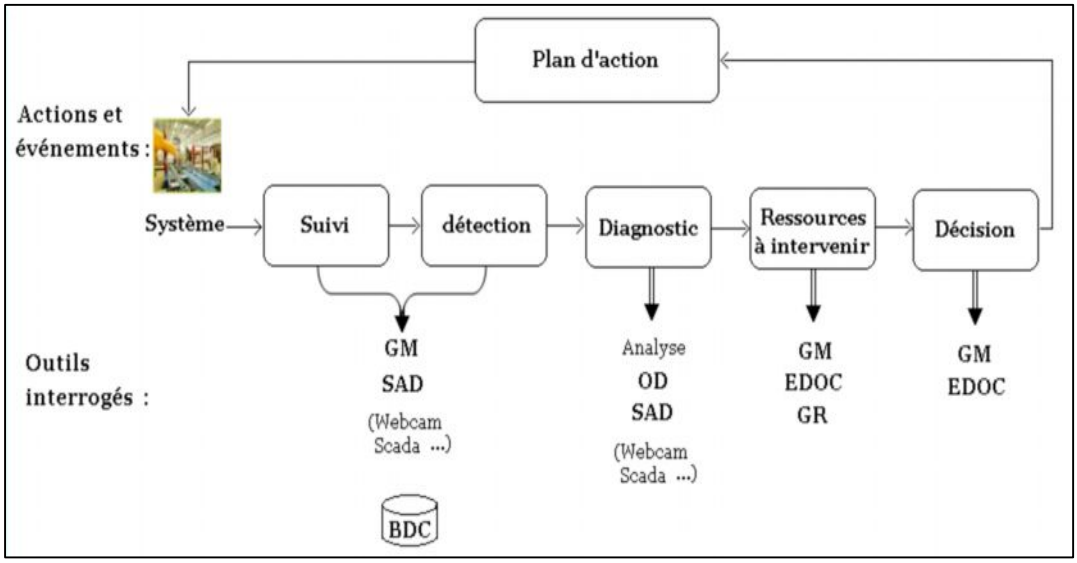


Figure III.8 Scénario et outils utilisés

III.6.3 Les différents agents dans « MAeMSOS »

Notre architecture « MAeMSOS » est composée de plusieurs Systèmes Multi Agents. Chaque site (entreprise, partie d'entreprise, système de production…) possède différentes tâches distribuées sur plusieurs agents.

Nous allons définir nos SMAs suivant l'approche AGR (Agent Groupe Rôle) :

III.6.3.1 Les agents et leurs rôles dans notre système

En réalité, chaque module de tâches est associé à un SMA car les rôles attribués aux agents au sein d'un même SMA sont nombreux et peuvent être interactifs les uns avec les autres. Pour notre cas, afin de simplifier les choses, nous allons amorcer notre façon de faire par l'association d'un agent à un SMA sachant que l'enrichissement de ce SMA peut se faire ultérieurement par l'ajout d'autres agents.

- ***Agent Gestionnaire de maintenance (AG) :*** enregistre les agents actifs du système, possède des liens avec l'Agent Diagnostic du site (AD) et l'Agent base de Données (ABD), et possède une base de connaissances (BDC). (cf. Figure III.11)
  Dans le cas de réception d'une alerte d'un équipement, l'AG va consulter l'ABD pour faire des statistiques pour décider de la nécessité de changer l'équipement en cause.

- ***Agent Acquisition de Données (AAD):*** c'est un agent de suivi, connecté aux capteurs d'un équipement donné, il reçoit les données de ces capteurs et les valide en les comparants avec les valeurs de l'état normal stocké dans une base de données. En cas d'anomalie il envoie un rapport d'erreurs à AG.

- ***Agent Diagnostic (AD) :*** en recevant un rapport d'erreurs d'un équipement de l'AG, il effectue des analyses pour trouver la cause. En se basant sur une base de données, et sur l'AAD qui va lancer les capteurs pour effectuer des mesures d'aide à l'analyse.

- ***Agent Base de Données (ABD) :*** il reçoit les requêtes à la base de données à partir des différents agents. Il possède un accès à une base de données externe, il constitue un médiateur de tous les accès à cette base. L'ABD offre des services de datamining, de modification, de mise à jour de la BDD et suivi des états d'un équipement sur différents sites.

- ***Agent Gestion de Ressources (AGR) :*** il reçoit des ordres de l'AG pour des ressources à intervenir pour la solution d'une erreur donnée. Il possède des liens avec les différents fournisseurs de ressources.

- ***Agent e-Documentation (ADoc):*** cet agent est destiné à la gestion de la documentation. Il possède une base de données qui constitue un annuaire de toutes les documentations existantes sur son site. En cas d'indisponibilité d'un document donné, une requête est faite à l'AF qui va lancer une recherche dans les ADoc des autres sites.

- ***Agent Client :*** agit comme un agent d'interface d'utilisateur. Les requêtes des/aux utilisateurs sont acheminer à partir de cet agent vers les autres agents du site.

- ***Agent Facilitateur (AF) :*** cet agent assure la coopération inter entreprises. Il possède un système de suivi qui va surveiller les erreurs survenues dans son site dans la BD. Et lors de la détection d'une succession d'anomalies d'un même équipement sur un ou plusieurs sites. L'AF va envoyer un rapport d'alerte à l'AG.

- ***Agent Facilitateur Intra entreprise (AFI) :*** cet agent assure la coopération intra entreprises.

- ***Agent Gestionnaire de Direction (AGD) :*** cet agent assure la gestion et la direction des différents agents des différents sites de l'entreprise. Les AGD de différentes entreprises vont avoir un même groupe « groupe Direction », ils vont assurer la coopération entre ces différentes entreprises. (cf. figure III.10)

Lorsqu'une ressource humaine, logicielle ou matérielle requise par une entreprise A est disposée dans une autre entreprise B, l'agent AGD de A va faire une demande de la ressource à l'AGD de l'entreprise B.

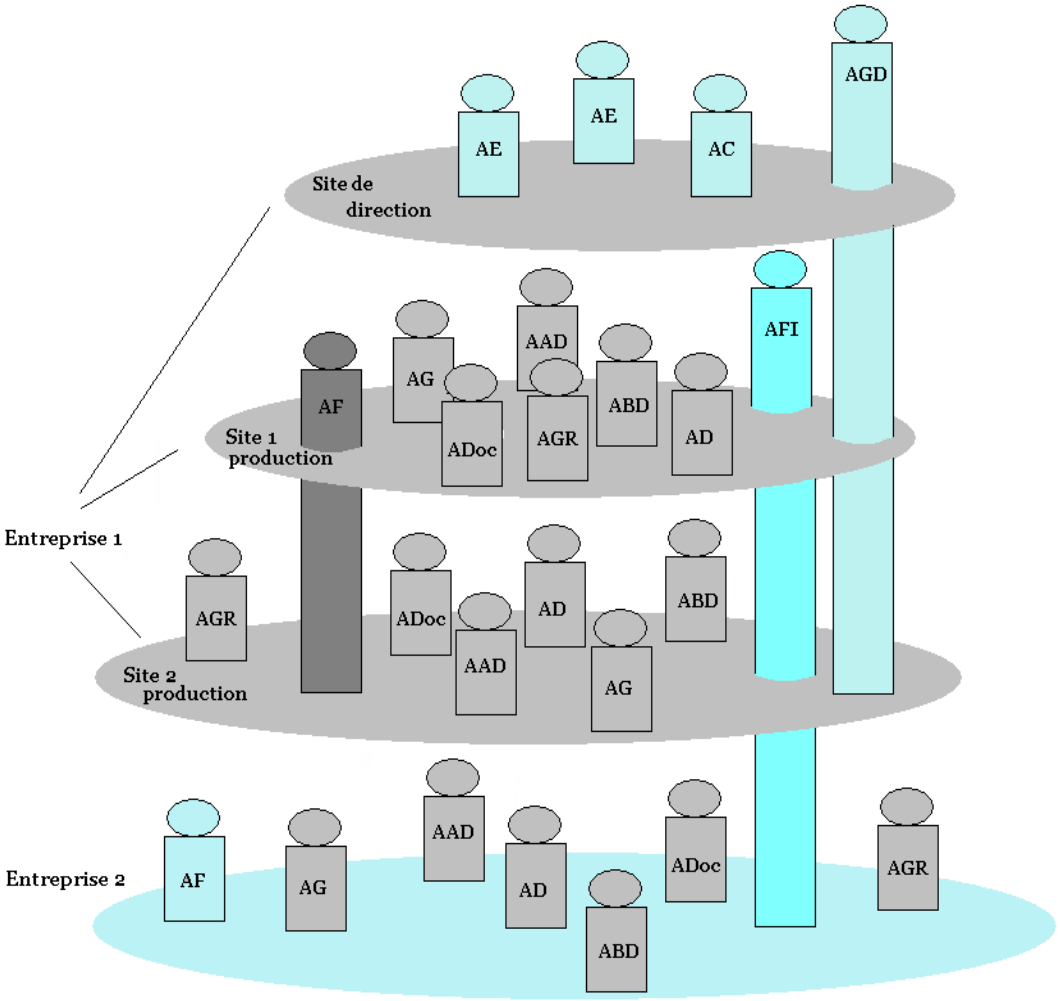


Figure III.9 Identification des groupes et rôles : diagramme plateau à fromage

III.6.3.2 Les groupes d'agents dans notre système

Dans notre système les groupes sont :

- ***Les groupes de sites de production :*** Les sites de production sont groupés suivant la spécialité, le type de production, l'emplacement (numéro d'étage, site est, ouest…) (cf. figure III.9).
- ***Groupe de direction :*** Chaque entreprise possède un groupe de direction pour l'ensemble de ses groupes. Composé de l'AGD et des différents Agents employeurs AE et personnels de la direction.

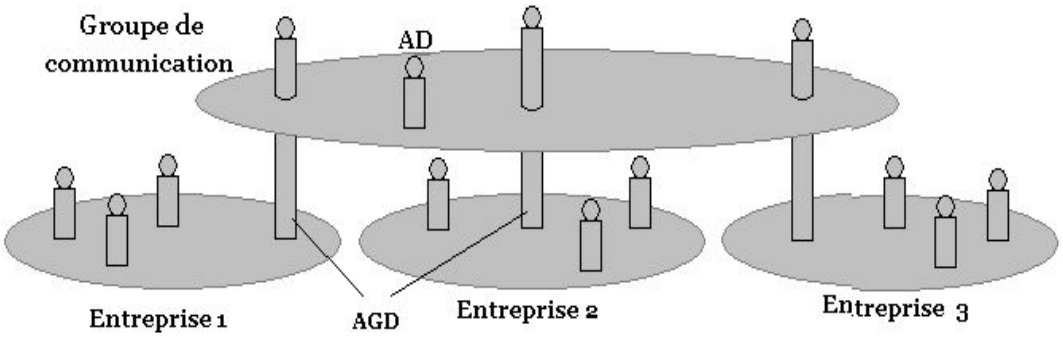


Figure III.10 Diagramme plateau à fromage de groupe de communication

- ***Groupe de communication :*** C'est le groupe comprenant les différents AG des groupes de directions des entreprises coopératives ainsi qu'un agent de direction (cf. figure III.10). Ce groupe assure la communication et donc la coopération entre les différentes entreprises. Ces différents agents AGD vont communiquer via l'ESB.

III.6.3.3 Structure organisationnelle du système :

Dans un modèle organisationnel, Chaque agent est une entité autonome et communicante qui peut jouer un ou plusieurs rôles dans un ou plusieurs groupes. Les rôles des agents représentent leurs capacités à effectuer un service pour un groupe donné. Un agent peut avoir plusieurs rôles et appartenir à plusieurs groupes, ce qui offre des possibilités de communication entre groupes. On remarque dans notre modèle (cf. figure III.10) que l'agent AGD permet d'assurer la correspondance entre les différents groupes d'entreprises et le groupe de communication.

Les groupes de notre système peuvent être organisés suivant les rôles et les interactions entre les différents agents intervenants (cf. figure III.11). Les interactions d'un cas d'étude sont retirées des rôles des différents agents prédéfinis précédemment et serons illustrer en figure III.12.

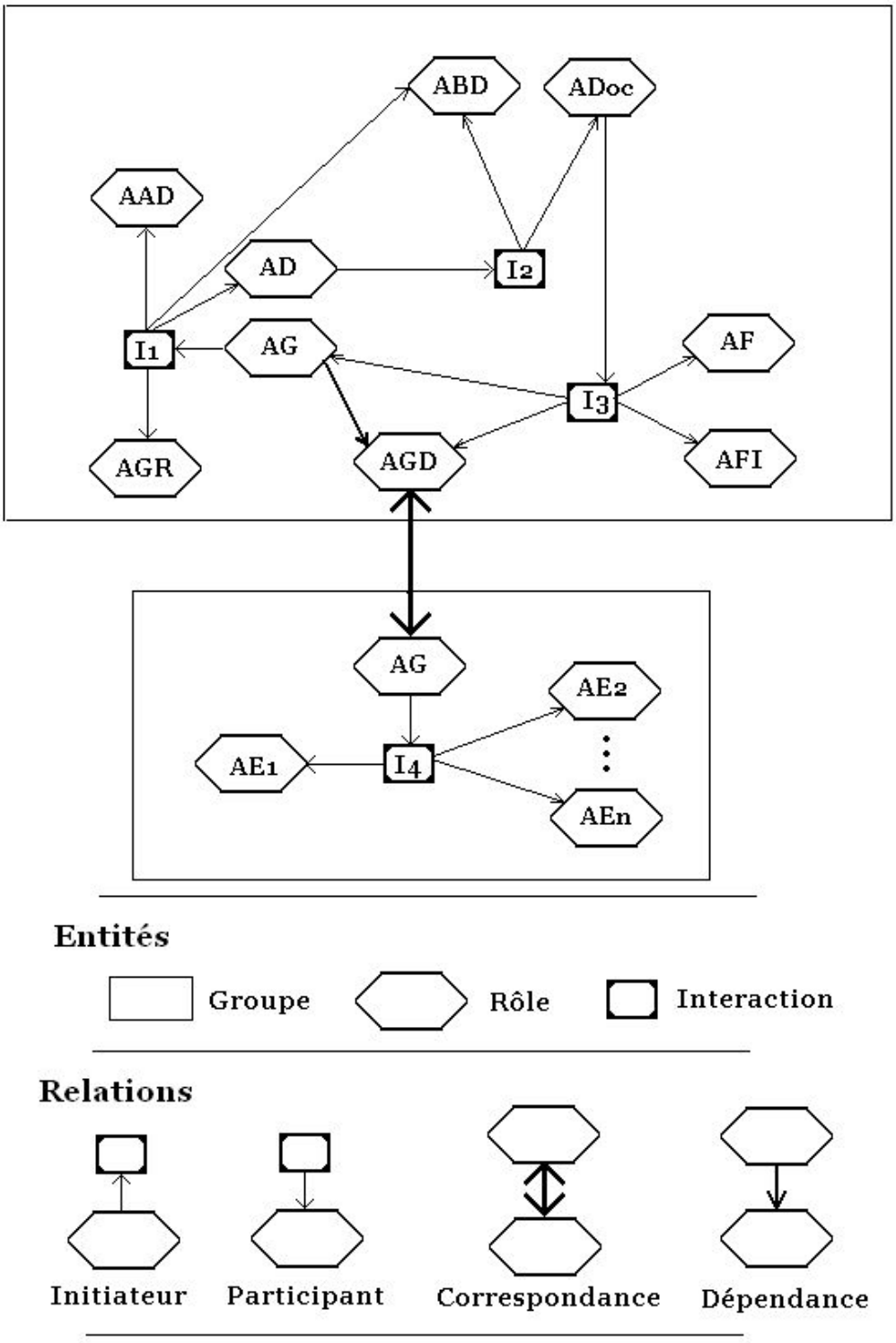


Figure III.11 Diagramme de structure organisationnelle

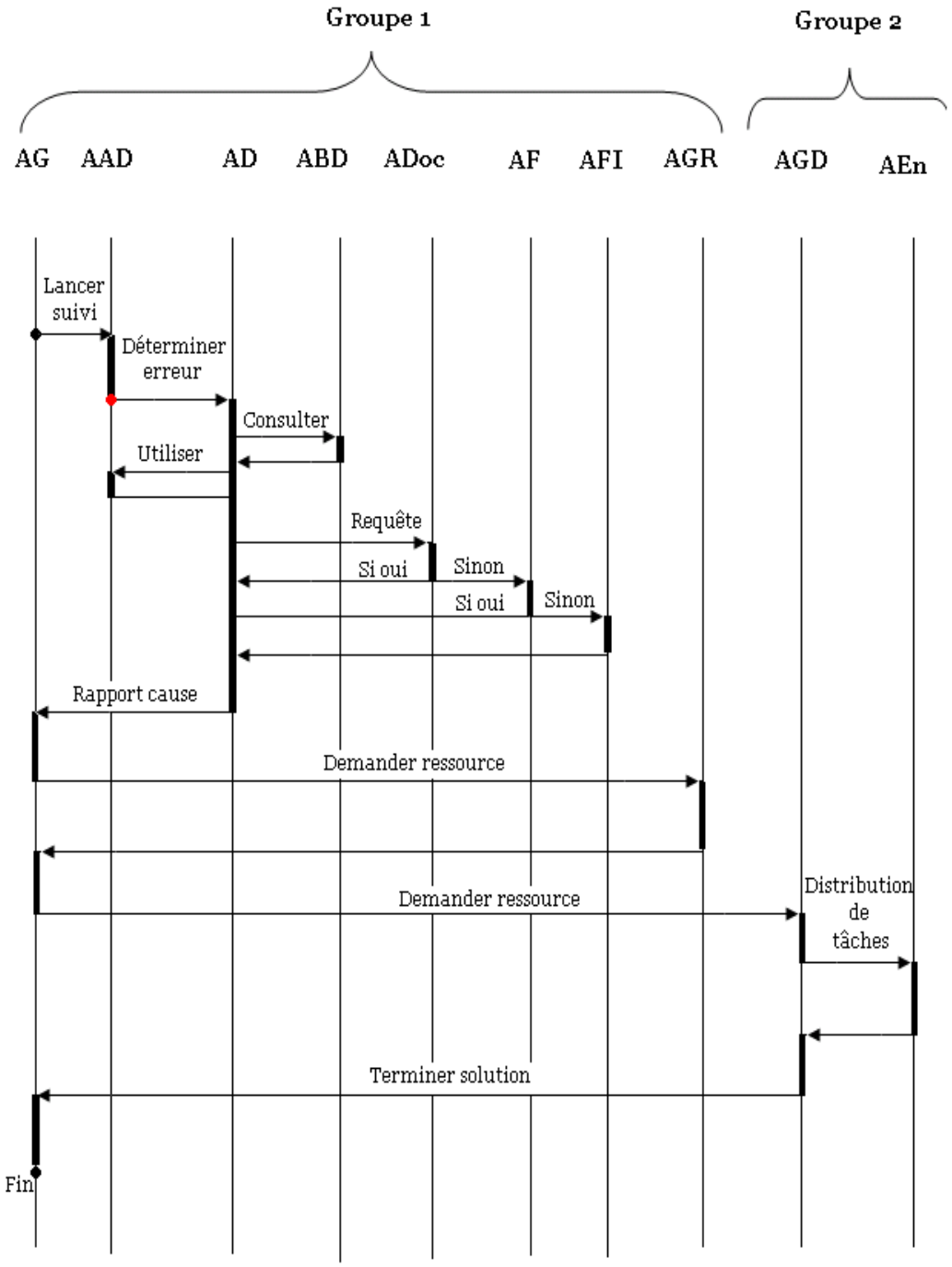


Figure III.12 Diagramme organisationnel de séquence

III.7 Validation d'un cas d'étude de la plateforme « MAeMSOS » dans une Architecture Orientée Services

III.7.1 Introduction

Dans cette partie, nous allons essayer de faire une validation d'un cas d'étude de la plateforme « MAeMSOS » dans une Architecture Orientée Services. Nous allons créer des processus métier à l'aide du moteur d'orchestration BPEL (Business Process Execution Language) qui nous permettra de définir l'ensemble des liaisons entre eux. Chacun de ces processus métier va représenter un Web-Service, qui va communiquer avec les autres via le Bus ESB par des messages.

Nous avons utilisé GlassFish ESB qui est une solution open source, basée sur les standards, outillée, bien documentée. L'un de ces concepts fondamentaux est l'intégration basée sur les messages. L'utilisation de glasseFish ESB nécessite l'installation de l'IDE Netbeans et le serveur d'application GlassFish.

BPEL a été utilisé pour la spécification formelle des processus métiers pour chacun des agents définis dans la partie III.6. 3.1 (cf. figures III.13, III.14), ainsi que leur orchestration et la définition des interactions correspondantes.

III.7.2 Orchestration des services

L'orchestration est une spécification pour les échanges métier qui propose de gérer l'ensemble de la séquence d'interactions entre plusieurs services. Elle permet de définir l'enchaînement des services selon un canevas prédéfini, et de les exécuter à travers des "scripts d'orchestration". Ces scripts sont souvent représentés par des processus métier ou des workflows inter/intra-entreprise. Ils décrivent les interactions entre applications en identifiant les messages et en branchant la logique et les séquences d'invocation. [46]

L'orchestration peut être définie comme une collaboration entre deux ou plusieurs services, gérée (orchestrée) par BPEL en général. Ce dernier consiste en un langage basé sur XML qui prend en charge la séquence complète d'invocations des divers services, ou "collaborateurs".

BPEL a été conçu pour intégrer un grand nombre d'applications, publiées sous forme de service dans un but métier particulier, sans dépendance de plate-forme ou de langage, le tout de manière automatique. C'est une spécification de IBM, Microsoft, et BEA. [46]

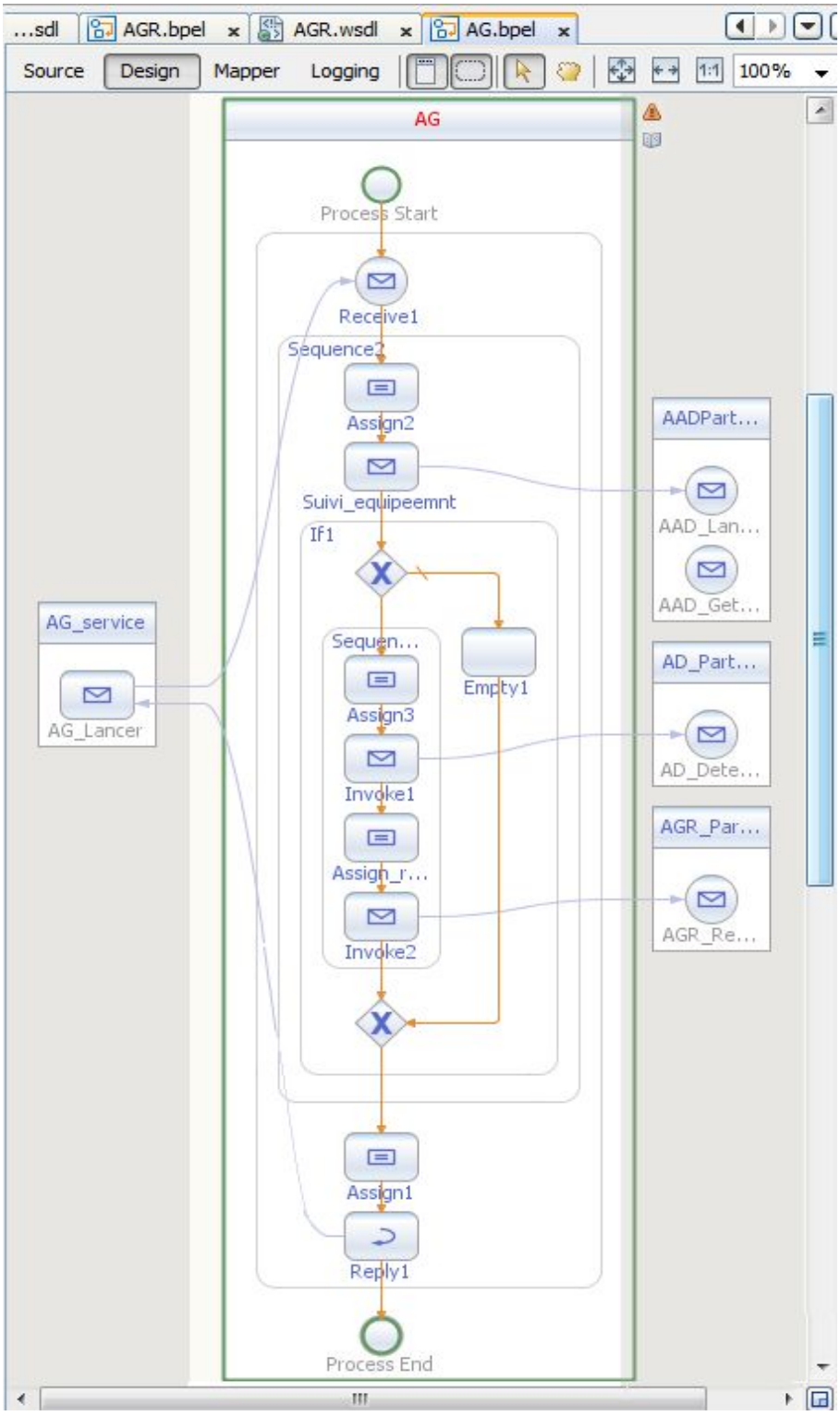


Figure III.13 Spécification formelle du processus métier AG

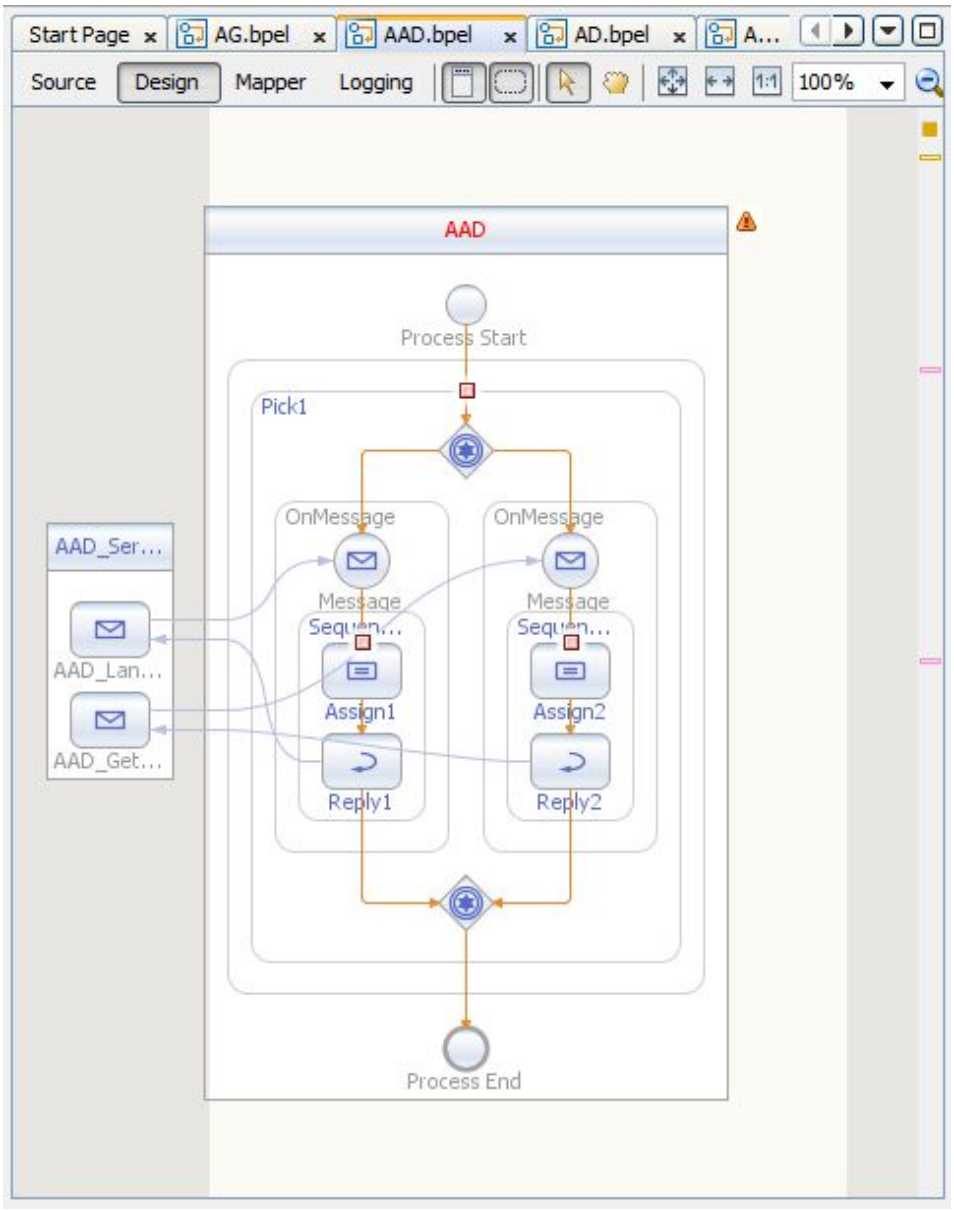
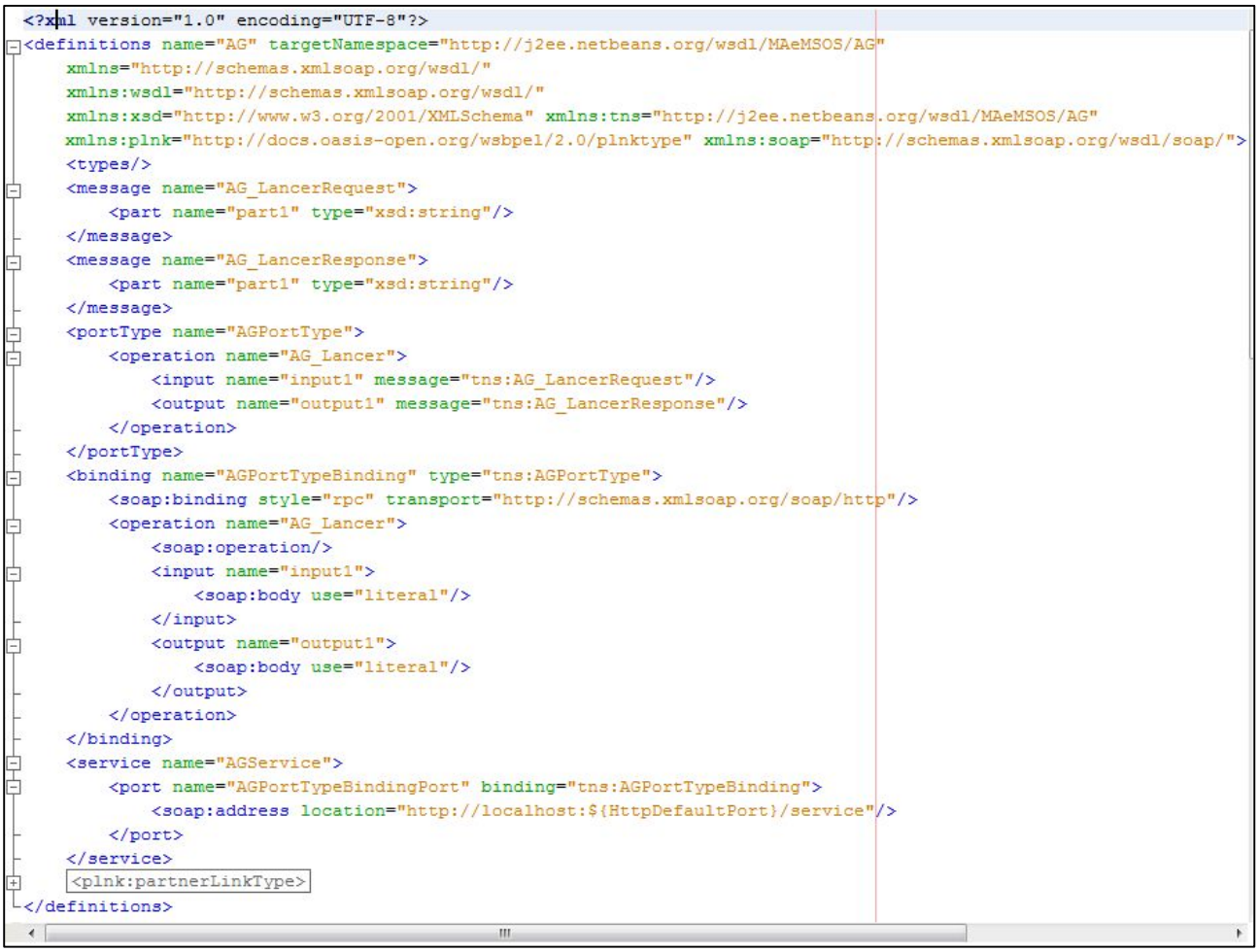


Figure III.14 Spécification formelle du processus métier AAD

Un processus BPEL dispose donc d'une logique d'invocation, celle-ci pouvant être synchrone ou asynchrone. BPEL fait fortement usage des autres langages liés aux Web-Services, à commencer par WSDL, SOAP et UDDI. Chaque processus BPEL dispose par ailleurs de sa propre définition WSDL (cf. figure III.15), un processus BPEL est donc un Web-Service à part entière.



```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="AG" targetNamespace="http://j2ee.netbeans.org/wsdl/MAeMSOS/AG"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://j2ee.netbeans.org/wsdl/MAeMSOS/AG"
    xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <types/>
    <message name="AG_LancerRequest">
        <part name="part1" type="xsd:string"/>
    </message>
    <message name="AG_LancerResponse">
        <part name="part1" type="xsd:string"/>
    </message>
    <portType name="AGPortType">
        <operation name="AG_Lancer">
            <input name="input1" message="tns:AG_LancerRequest"/>
            <output name="output1" message="tns:AG_LancerResponse"/>
        </operation>
    </portType>
    <binding name="AGPortTypeBinding" type="tns:AGPortType">
        <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="AG_Lancer">
            <soap:operation/>
            <input name="input1">
                <soap:body use="literal"/>
            </input>
            <output name="output1">
                <soap:body use="literal"/>
            </output>
        </operation>
    </binding>
    <service name="AGService">
        <port name="AGPortTypeBindingPort" binding="tns:AGPortTypeBinding">
            <soap:address location="http://localhost:${HttpDefaultPort}/service"/>
        </port>
    </service>
    <plnk:partnerLinkType>
</definitions>
```

Figure III.15 Document WSDL du service AG

III.7 Conclusion

Les nouvelles technologies de l'information permettent d'intégrer différents systèmes d'aide via des plateformes. Nous avons proposé une classification des différentes architectures de maintenance existantes pour en déduire une architecture comme système support aux services de maintenance.

Parmi les architectures existantes de e-maintenance nous avons cité Proteus : une e- plateforme de maintenance qui se base principalement sur la notion de Web-Services et ses standards. Et Nous avons mis en évidence un nouveau type d'architecture de e-maintenance MAeMSOS orienté service basé sur des systèmes multi agents coopératifs, du besoin d'avoir des systèmes intelligents. Puisque les agents sont de nature autonomes et intelligents nous avons essayé de concevoir un SMA dédié e-maintenance dans une architecture orientée services.

Le système MAeMSOS permet de décomposer le système en plusieurs tâches pour l'accomplissement d'un but commun. Ces différentes tâches vont être distribuées sur différents agents sous forme de services pouvant communiquer via un bus ESB.

Conclusion générale

Nul ne peut ignorer l'apport que peut apporter les WebServices (WS) à l'entreprise moderne. Ces derniers permettent de lui apporter, économiquement, de la valeur ajoutée en utilisant cette technologie. Le développement et l'utilisation de WS ne cessent d'augmenter, ce qui exige la structuration de ces WS en ce qu'on appelle les architectures orientées service (SOA).

Afin de faciliter la tâche aux utilisateurs des SOA et de faire abstraction des détails d'implémentation, un middleware inter-WS, appelé ESB (Enterprise Service Bus) a été mis en œuvre, il est basé, à bas niveau, sur les briques technologiques de base utilisant un WS à savoir (SOAP, WSDL, UDDI) basés tous sur XML.

Les domaines d'application de ces technologies sont nombreux et divers (e-commerce, e-administration, etc.). Pour notre cas, nous avons appliqué notre contribution à l'e-maintenance, en proposant une architecture que nous avons baptisée AeMSOS.

Sur le plan académique, la structuration des SOA exige une forme de modélisation répondant aux caractéristiques technologiques citées. Dans ce sens, le paradigme des Systèmes Multi-agents (SMA) peut répondre à ces cas d'études, car il présente toutes les analogies (SMA/SOA) :
Autonomie,
Communication,
Environnement,
Etc.

Dans cette présente contribution, nous nous sommes fixés deux objectifs :
Faire un état de l'art sur les SOA et particulièrement le middleware ESB.
Faire une contribution permettant la projection, sous la forme de modélisation, des aspects académiques des SMA sur les SOA.

Cette modélisation, nous a permis d'associer les caractéristiques d'agents et SMA (notions d'autonomie, coopération…), avec les standards des architecture orientées services pour avoir des systèmes coopératifs pour la maintenance de nombreuses entreprises de fabrication, qui ont différents systèmes hétérogènes implémentés sur différentes plateformes, distribuées géographiquement sur plusieurs sites. L'utilisation des standards SOA, permet l'intégration dynamique de nouveaux systèmes hété-

rogènes à la plateforme de maintenance, ainsi que la communication et la coopération entre ces différents systèmes tout en offrant les critères de qualité de services, sécurité assurés par l'utilisation des ESB.

Du coté de la modélisation SMA, en utilisant l'approche Agent /Groupe/Rôle (AGR), nous avons décomposé notre système en plusieurs tâches suivant l'approche VOYELLE qui définit les différents rôles associés aux différents agents. Ensuite, nous avons présenté les structures organisationnelles des différents groupes du système. Ainsi que le modèle d'interaction.

Par cette projection SMA/SOA, générant la plateforme AeMSOS, nous considérons que nous avons pu apporter une contribution à cette thématique de recherche.

En perspectives, la validation des différents aspects sur une plateforme réelle est à entreprendre. Aussi, l'introduction la s-maintenance dans notre système en prenant en compte la notion de sémantique des données échangées en utilisant les ontologies apporterait plus valeur académique.

Références bibliographiques

[1] M.P.Papazoglou, “What’s in a Service?”, VLDB, Springer-Verlag, 2007, pp. 11-28.

[2] B. Manouvrier, L. Ménard, Intégration applicative EAI, B2B, BPM et SOA, Hermès Lavoisier, 2007

[3] M.P.Papazoglou and W.J. Heuvel “Service oriented architectures: approaches, technologies and research issues ”, Springer-Verlag, 2007, pp. 389-415.

[4] D. Chappell, Enterprise Service Bus, O'Reilly, Sebastopol, 2004.

[5] JM. Chauvet, “Web Services avec SOAP, WSDL, UDDI, ebXML”, Eyrolles, 2002

[6] M.P.Papazoglou, P. Traverso, S. Dustdar and F. Leymann, “Service-Oriented Computing:State of the Art and Research Challenges”, Computer, IEEE, 2007, pp. 38-45

[7] M.P.Papazoglou, P. Traverso, S. Dustdar and F. Leymann, “Service-oriented computing: concepts, characteristics and directions”, WISE'2003, IEEE, 2003, pp. 3-12

[8] A. Benaouda, N. Zerhouni, C. Varnier, « Une approche Multi-Agents coopératifs pour la gestion des ressources materielles dans un contexte multi-sites de e-manufacturing», MOSIM'06, Rabat, Morocco, 2006.

[9] M. Hedjeres, A. Benaouda et M. Mostefai, « Contribution à la spécification fonctionnelle et architecturale du middleware ESB-SOA », ICAI’09, BBA, Algerie, 2009, pp 447-453.

[10] H. Kadima , V. Monfort , Les Web Services, DUNOD, paris,2003

[11] A. Mullenders, e-DRH: Outil de gestion innovant, de Boeck Université, 2009

[12]J. Ferber, Les systèmes multi agents, vers une intelligence collective, inter edition 1995

[13]M. P. Papazoglou Service Oriented Computing &Web Services : http://www.scribd.com/doc/6973991/papazoglou
[14] A. Welch, U. Onat, C. Sotomayor, « Service Oriented Architecture in Multi-Agent Systems», 2008

[15] M.P.Papazoglou, "What's in a Service?", ECSA, Madrid, September 2007

[16] http://www.w3.org/TR/ws-arch/

[17] JP. Georgé, « RESOLUTION DE PROBLEMES PAR EMERGENCE », Toulouse, 2004, 244p.

[18] NR. Jennings, K. Sycara, M. Wooldridge, « A Roadmap of Agent Research and Development », AAMAS, Boston, 1998, pp. 7-38.

[19] Z.Guessoum, « Modèles et architectures d'agents et de Systèmes Multi-Agents adaptatifs », thèse HDR, Paris 6, 2003

[20] X. Morel, P. Grojean, G.Plouin, C. Rognon . SOA le guide de l'architecte, DUNOD, paris, 2006

[21] M. Occello DIAMOND: UNE APPROCHE POUR LA CONCEPTION
DE SYSTEMES MULTI-AGENTS EMBARQUES, GRENOBLE, 2005

[22] S. Hassas, « Systèmes Complexes à base de Multi-Agents Situés », Lyon 1, 2003

[23] T. Lemlouma, A. Boudina, « L'intelligence Artificielle Distribuée et les SystèmesMulti-Agents » : http://opera.inrialpes.fr/people/Tayeb.Lemlouma/Papers/IAD_Presentation.pdf

[24] IM. Tani, Strategie de rendez-vous dans les systèmes multi-agents : http://www.memoireonline.com/08/08/1510/strategie-de-rendez-vous-dans-les-systemes-multi-agents.html

[25] H. ABOUAÏSSA, JC. NICOLAS, A. BENASSE, « Modelisation et évaluation de performances d'une plate-forme multi-modale : approche formelle basée sur les systèmes Multi-Agents et les réseaux de Petri », Lisbonne, 2002

[26] T. Jarraya, « Réutilisation des protocoles d'interaction et Démarche orientée modèles pour le développement multi-agents », Reims, 2006, pp 173

[27] I. Jars, « Contribution des Sciences Sociales dans le domaine de l'Intelligence Artificielle distribuée : ALONE, un modèle hybride d'agents apprenant », Lyon1, 2005

[28] Y. Demazeau and A.R. Costa, « Populations and organisations in open multi-agent systems », First Symposium on parallel and Distributed AI, Hyderabad, India, 1996.

[29] L. Lacomme, Y. Demazeau, V. Camps, « Classification des mécanismes organisationnels dans les réseaux d'agents », 2009

[30] Matthieu Amiguet, MOCA :Un modèle componentiel dynamique pour les systèmes multi-agents organisationnels, Neuchâtel, Suisse, 2000

[31] AF. Seghrouchni, S. Haddad, T. Melitti, « Interopérabilité des systèmes multi-agents à l'aide des Web Services », JFSMA, 2004, pp. 92-104

[32] F. CURBERA, W. A.NAGY, S.WARANA, « Web Services : Why and
How? », OOPSLA, 2001.

[33] F. Bourdon, « Systèmes multi-agents », cours sur les systèmes multi-agents, université de Caen, 2001

[34]E. Gouardères, « Interaction et Coordination dans les Systèmes Multi-Agents », cours sur les SMA, Université de Pau,2009

[35] AC. Marquez, « The Maintenance Management Framework », Springer London, 2007

[36] P. Jonsson, « Towards an holistic understanding of disruptions in Operations Management », Journal of Operations Management, 2000, Volume 18, pp 701-718

[37] I. Rasovska, B. Chebel-Morello, N. Zerhouni, « Classification des différentes architectures en maintenance », 7e Congrès international de génie industriel, 2007, Trois-Rivières, Québec(CANADA).

[38] H. Kaffel, « La maintenance distribuée: concept, évaluation et mise en œuvre », Thèse de doctorat, Université Laval, Quebec, 2001.

[39] H. Vermaak, J. Kinyua, « Multi-Agent Systems based Intelligent Maintenance Management for a Component-Handling Platform », IEEE, Scottsdale, USA, 2007

[40] AC. Marquez, B. Iung, « A review of e-maintenance capabilities and challenges », from Journal of Systemics, Cybernetics and Informatics, Volume 6- Number 1, 2008, pp 62-66.

[41] Proteus, Une plate-forme générique pour la e-maintenance, http://www.proteus-iteaproject.com/

[42] E. Levrat, B. Iung, « TELMA: A full e-maintenance platform », Centre de Recherche en Automatique de Nancy, France

[43] M. Diulio, « Revolutionizing Maintenance Through Remote Monitoring via ICAS & Distance Support » OSD Great Ideas Program, 2002, http://www.sae.org/events/dod/presentations/greatideas-diulio.pdf

[45] X. Kou, GlassFish Administration, Packt Publishing, 2009

[46] H. Darío ROJAS, « Orchestration à haut niveau et BPEL», Master Mathématiques Informatique, Institut d'Informatique et de Mathématiques Appliquées de Grenoble, 2006.