Systèmes Multi-agents

Agents et Services

Claude Moulin

Université de Technologie de Compiègne

IA04

Sommaire

Interaction client serveur

- L'opération à réaliser est présentée sous la forme d'une procédure que le client peut faire exécuter à distance par le serveur ;
- Service basique (API d'appel de procédure distante)
  - Côté client : génère l'appel distant et récupère le résultat
    invoque (id_client, id_serveur, nom_procedure, paramètres) ;
  - Côté serveur : reçoit, traite un appel et répond
    traite (id_client, id_serveur, nom_procedure, paramètres) ;

Interaction client serveur

- L'opération à réaliser est présentée sous la forme d'une procédure que le client peut faire exécuter à distance par le serveur ;
- Service intégré objet :
  - Côté client : on invoque une procédure localisée à distance ref_objet_serveur.nom_procedure (paramètres) ;
  - Côté serveur : on déploie l'objet qui implante la procédure nom_procedure (paramètres) ;

Avantage RPC

- On s'affranchit du côté basique des communications en mode message ;
- On n'a pas à programmer des échanges au niveau réseau en mode message ;
- On utilise une structure familière : l'appel de procédure (il ne faut cependant pas ignorer les différences local/distant).
- On dispose de mécanismes modernes de programmation.
- La vision des applications réparties est modulaire.

Approches RPC - 1

- Intégrées dans les systèmes d'objets répartis :
- OMG CORBA (Object Management Group - Common Object Request Broker Architecture)
- Oracle Java RMI (Remote Method Invocation)
- Microsoft - DCOM (Distributed Component Object Model)

Approches RPC - 2

- Intégrées dans les systèmes de composants :
- OMG CCM (Object Management Group - Corba Component Model)
- Oracle J2EE EJB (Java 2 (Platform) Enterprise Edition - Enterprise Java Beans)
- WS-SOAP (Web Services - Simple Object Access Protocol)

Implémentation : Souches client et serveur

- La souche client ("client stub")
  - Procédure coté client qui reçoit l'appel en mode local, le transforme en appel distant, en envoyant un message
  - Reçoit le message contenant les résultats après l'exécution
  - Retourne les résultats comme dans un retour de procédure.
- La souche serveur ("server stub")
  - Procédure côté serveur qui reçoit le message d'appel
  - Fait réaliser l'exécution sur le site serveur par la procédure serveur
  - Récupère les résultats et retransmet les résultats par message.

Avantages de l'appel distant réalisé par messages

- Volume de code ou de données serveur quelconque.
- Applicable en univers hétérogènes moyennant des conversions.
- Partage d'accès sur le site serveur analogue au transactionnel.

Inconvénients de l'appel distant réalisé par messages

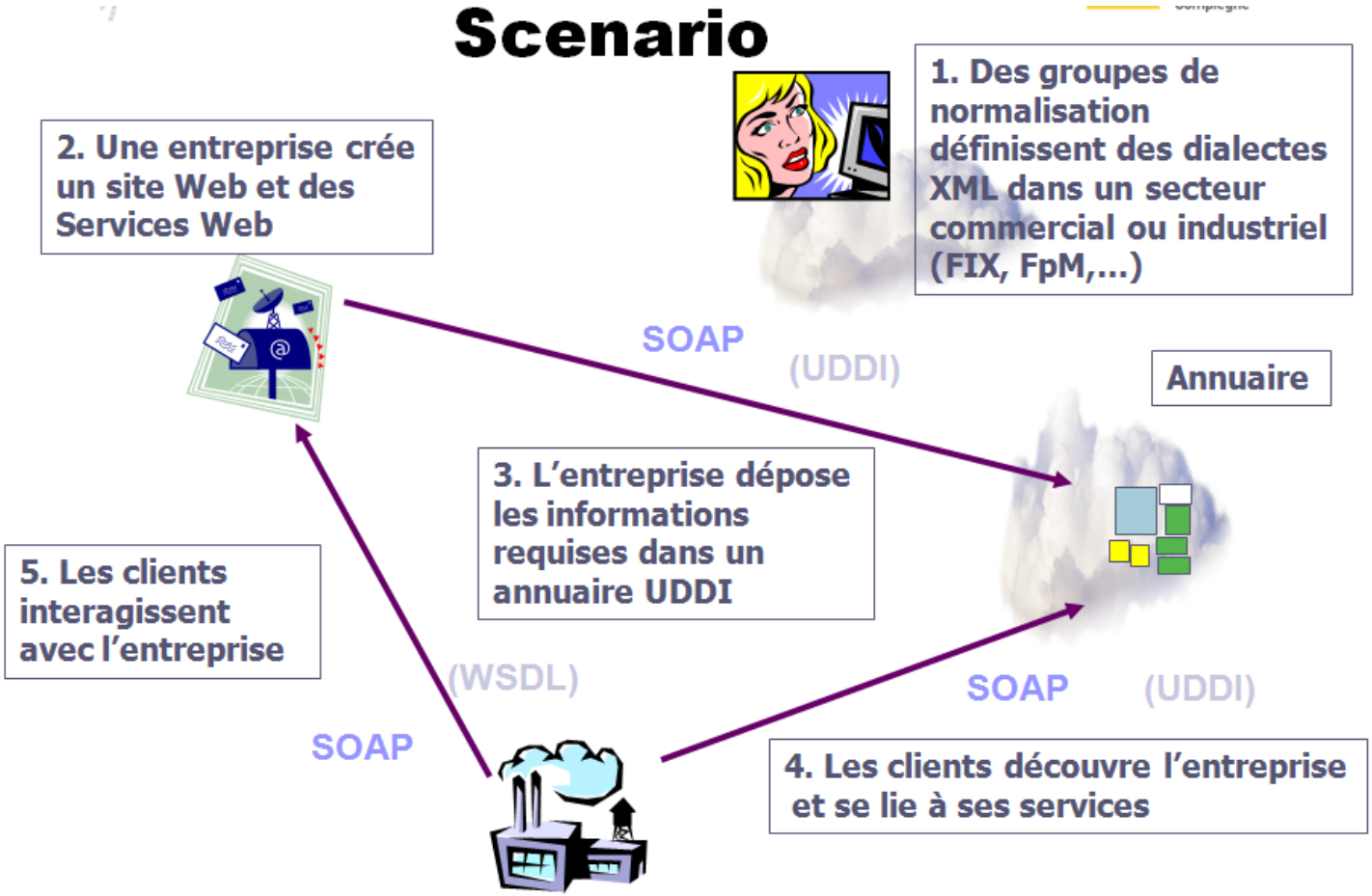
- Pas d'usage de références dans les paramètres.
- Échange de données complexes/de grande taille délicat.
- Peu efficace pour de très nombreux appels.

Sommaire

Mécanismes généraux des services Web

- Ils permettent des interactions entre applications.
- Sont basés sur des infrastructures Web ubiquitaires.
- Utilisent des appels de procédures distantes et des échanges de documents.
- Echangent des messages dans le protocole SOAP (au-dessus de HTTP).
- Invoquent dynamiquement des modules situés.
- Sont conçus avec des méthodes orientées objet
- Fournissent des interfaces auto décrites définies en XML

Scénario


Scenario
1. Des groupes de normalisation définissent des dialectes XML dans un secteur commercial ou industriel (FIX, FpM,...)
2. Une entreprise crée un site Web et des Services Web
SOAP
(UDDI)
Annuaire
3. L'entreprise dépose les informations requises dans un annuaire UDDI
5. Les clients interagissent avec l'entreprise
(WSDL)
SOAP
SOAP
(UDDI)
4. Les clients découvre l'entreprise et se lie à ses services

Protocoles pour les Services Web

- Transport : HTTP, SMTP, FTP
- Interaction : Service Oriented Access Protocol (SOAP)
- Description : Web Service Description Language (WSDL)
- Publication : Universal Description Discovery and Integration (UDDI)
  Electronic business XML Initiative (ebXML)
- Composition : Business Process Execution Language for Web Services (BPEL4WS)

Sommaire

Protocoles pour les Services Web

- premier des protocoles de services web (1998).
- construit sur HTTP, et s'appuie sur une mise en forme XML de tous les paramètres de l'invocation du service, et de la réponse, selon une syntaxe générique très simple.

Sommaire

SOAP : définition

- SOAP est un protocole basé sur XML Servant à invoquer à distance des méthodes, des serveurs, des services, des composants ou des objets,
- SOAP est une spécification qui indique à un client et à un fournisseur de service comment formater et lire un message XML utilisé par un service

SOAP : caractéristiques

- SOAP est un protocole de communication léger
- SOAP est fait pour faire communiquer des applications entre elles
- SOAP n'est lié à aucun langage de programmation
- SOAP est simple et extensible
- SOAP est une initiative conjointe d'IBM et de Microsoft

SOAP : structure

- Une enveloppe décrivant le contenu, le destinataire et la nature du message
- Des règles de codage concernant le mécanisme de sérialisation d'échange des objets
- Une Convention de représentation des appels et des réponses des procédures distantes

SOAP : protocole HTTP

- Bon moyen de faire communiquer entre-elles les applications
- Sécurité :
  - filtrage des requêtes au niveau des firewalls
  - Utilisation des mécanismes d'authentification standards (par exemple via ssl).

SOAP : les messages

- Enveloppe
- Header(facultatif)
- Body + Fault

Message SOAP incorporé dans une requête HTTP

```
POST /names HTTP/1.1
Host: www.academictutorials.com
Content-Type: text/xml; charset="utf-8"
                                    Content-Length: nnnn
SOAPAction: "Some-URI"
```

Enveloppe

```
<?xml version="1.0"?>
<env:Envelope
 xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
  <env:Body>
   <m:GetName
     xmlns:m="http://www.academictutorials.com/names">
     <m:Item>FirstName</m:Item>
   </m:GetName>
  </env:Body>
</env:Envelope>
```

Exemple

```
<env:Body>
  <p:itinerary
    xmlns:p="http://travelcompany.org/reservation/travel">
   <p:departure>
     <p:departing>New York</p:departing>
     <p:arriving>Los Angeles</p:arriving>
     <p:departureDate>2001-12-14</p:departureDate>
     <p:departureTime>late afternoon</p:departureTime>
     <p:seatPreference>aisle</p:seatPreference>
   </p:departure>
   <p:return>
     ...
   </p:return>
  </p:itinerary>
 </env:Body>
```

Exemple

```
<env:Body>
  <p:itinerary
    xmlns:p="http://travelcompany.org/reservation/travel">
   <p:departure>
     ...
   </p:departure>
   <p:return>
     <p:departing>Los Angeles</p:departing>
     <p:arriving>New York</p:arriving>
     <p:departureDate>2001-12-20</p:departureDate>
     <p:departureTime>mid-morning</p:departureTime>
   </p:return>
  </p:itinerary>
 </env:Body>
```

Clarification

```
<env:Body>
  <p:itineraryClarification
    xmlns:p="http://travelcompany.org/reservation/travel">
   <p:departure>
     <p:departing>
       <p:airportChoices>
          JFK LGA EWR
       </p:airportChoices>
     </p:departing>
   </p:departure>
   <p:return>
     ...
   </p:return>
  </p:itineraryClarification>
 </env:Body>
```

Encodage des valeurs de type simple

- directement en utilisant l'attribut type du NS des schémas XML dont la valeur correspond à un type fondamental des schémas.
  Exemple :
  ```
  <cost xsi:type="xsd:float">29.5</cost>
  ```
- directement par le nom de l'élément (choisi dans le NS d'encodage : enc)
  ```
  <enc:int>45</enc:int>
  ```

Encodage des valeurs de type complexe

- Une structure est un type composé dans lequel les membres sont accessibles uniquement grâce à des noms différents.
- Un tableau est un type composé dans lequel les membres sont accessibles par leur position.

Tableau

```
<ec:Array  xmlns:env="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsi = "http://www.w3.org/1999/XMLSchema-instance"
 xmlns:xsd = "http://www.w3.org/1999/XMLSchema"
 env:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:ec = "http://schemas.xmlsoap.org/soap/encoding/"
 ec:arrayType = "xsd:int[4]">
  <a>1</a>
  <a>2</a> <!--les noms n'ont pas d'importance -->
  <a>3</a>
  <a>4</a>
</ec:Array>
```

Sommaire

WSDL

- WEB Services Description Language
- Langage décrivant comment utiliser un service WEB
- Un document WSDL décrit la localisation d'un service et le protocole à utiliser pour invoquer ce service
- WSDL est une initiative conjointe de IBM et de Microsoft

Description d'un service

- La description d'un service en spécifie trois aspects :
- Les Opérations que le service offre
- Les Messages que le service accepte
- Le Protocole que le client doit utiliser pour accéder au service

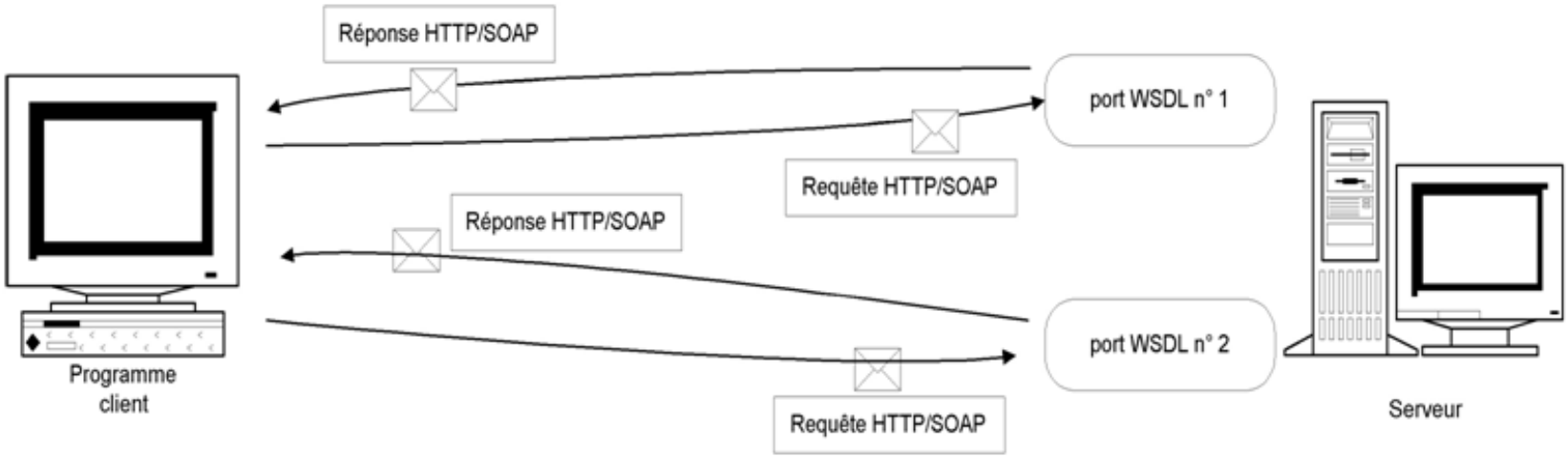
Contenu WSDL

- Un document WSDL contient principalement :
  - Les types des données échangées
  - Les messages
  - Les types de port
  - La définition du service

Utilisation

- Les Services Web utilisent WSDL pour spécifier un contrat.
- Le contrat est la description d'un ensemble de terminaisons ou points de communication (ports).
- Ils décrivent la manière de formater les documents XML pour les envoyer à ces points.
- Un port est simplement une adresse réseau.

Scénario


Réponse HTTP/SOAP
port WSDL n° 1
Requête HTTP/SOAP
Réponse HTTP/SOAP
port WSDL n° 2
Requête HTTP/SOAP
Programme client
Serveur

Balises XML - 1

- Un document WSDL est un document XML dont le root est la balise definition.
- Elément types : décrit les types de données utilisés.
- Elément message : décrit la structure d'un message échangé.
- Elément portType : décrit un ensemble d'operations (interface d'un service web).
- Elément operation : décrit une opération réalisée par le service web. Une opération reçoit et envoie des messages.

Balises XML - 2

- Elément binding : décrit le lien entre un protocole (soap/http) et un portType.
- Elément service : décrit un service comme un ensemble de ports.
- Elément port : décrit un port au travers duquel il est possible d'accéder à un ensemble d'operations. Un port référence un Binding.

SOAP : requête

```
<SOAP-ENV:Body>
 <m:GetLastTradePrice xmlns:m="Some-URI">
  <tickerSymbol>DIS</tickerSymbol>
 </m:GetLastTradePrice>
</SOAP-ENV:Body>
```

SOAP : réponse

```
<SOAP-ENV:Body>
  <m:GetLastTradePriceResponse xmlns:m="Some-URI">
   <price>34,5</price>
 </m:GetLastTradePriceResponse>
</SOAP-ENV:Body>
```

Types

```
<types>
 <schema targetNamespace="http://example.com/stockquote.xsd"
         xmlns="http://www.w3.org/2001/XMLSchema ">
 <element name="TradePriceRequest">
  <complexType>
   <all> <element name="tickerSymbol" type="string"/> </all>
  </complexType>
 </element>
 <element name="TradePrice">
   <complexType>
    <all> <element name="price" type="float"/> </all>
   </complexType>
 </element>
 </schema>
</types>
```

Messages

```
<message name="GetLastTradePriceInput">
 <part name="body"  element="xsd1:TradePriceRequest"/>
</message>
<message name="GetLastTradePriceOutput">
 <part name="body" element="xsd1:TradePrice"/>
</message>
```

Type de port

```
<portType name="StockQuotePortType">
 <operation name="GetLastTradePrice">
  <input message="tns:GetLastTradePriceInput"/>
  <output message="tns:GetLastTradePriceOutput"/>
 </operation>
</portType>
```

Types d'opérations

- Une opération est composée d'un message de type input et d'un message de type output.
- Les deux types de messages sont optionnels et leur ordre désigne le type de l'opération :
- One-way : input sans output
- Request-response : input suivi de output
- Solicit-response : output suivi de input
- Notification : output sans input

Sommaire

Publication - Recherche

- Les Services Web rendent possible le concept de la découverte des services.
  Un client utilise un annuaire de services pour trouver celui qui l'intéresse.
- Un annuaire UDDI (lui-même un service Web) contient un ensemble de descriptions de services ce qui permet à un client de découvrir la description d'un service Web.

Publication - Recherche

- Les annuaires UDDI sont utilisés de deux façons :
  - Les fournisseurs de services Web y publient la description de leurs services.
  - Les clients y retrouvent les listes de services qui satisfont leurs critères.

Entrées d'annuaires UDDI

- Une entrée d'annuaire est un fichier XML qui décrit une entreprise et les services qu'elle offre. Elle est composée de trois parties :
- Les "pages blanches" décrivent l'identité du fournisseur : nom, adresse, contacts, etc.
- Les "pages jaunes" décrivent les services offerts incluant les catégories industrielles basée sur des taxonomies standard.
- Les "pages vertes" décrivent l'interface du service avec suffisamment de détails pour pouvoir écrire une application utilisant le service.

Processus de publication

- Créer la définition de l'interface du service en WSDL.
- Rendre publiques ces définitions.
- Les programmeurs construisent des services conformes aux standards industriels.
- Des outils permettent de créer des descriptions de services conformes au standard UDDI.
- Les nouveaux services sont déployés et publiés dans un annuaire UDDI accompagné des descriptions UDDI appropriées.

Applications Propriétaires

- Les services Web peuvent servir d'interface avec des applications propriétaires de façon à les ouvrir sur le réseau.
- Ces applications peuvent être utilisées sans adaptation particulières.
- Les descriptions WSDL basées sur des schémas XML décrivent les types de données.
- SOAP permet d'activer les services.

Sommaire

Principe de composition

- Une application Web peut faire appel à plusieurs services Web.
- Il s'agit alors de spécifier de façon déclarative (dans des documents XML) la composition des services.
- Il s'agit ensuite de créer des moteurs capables d'interpréter ces documents et ainsi d'orchestrer les services.
- Ceci est semblable à la définition d'un langage de programmation (structure de contrôle, etc.)

Orchestration

- L'orchestration des services Web concerne la fourniture d'approches basées sur des standards ouverts pour composer les services Web de façon à créer des processus métier à haute valeur ajoutée.
- BPEL4WS (ou BPEL) est une spécification qui modélise le comportement des services Web lors des interactions
- BPEL4WS est essentiellement une couche au-dessus de WSDL
- WSDL définit les opérations spécifiques permises
- BPEL4WS définit comment les opérations sont séquencées.

Sommaire

WS-*

- Les services SOAP sont entièrement décrits par un document Web Services Description Language (WSDL).
- Ceci permet à des outils de créer des APIs clientes.
- Les développeurs d'application Clientes ne voient que des appels de méthodes avec paramètres.

WS-*

- SOAP représente de l'information sous forme XML.
- Pour : format commun pour exprimer les échanges.
- Contre : XML n'est pas particulièrement efficace, ne convient pas bien à certains langages.

REST

- Basé sur deux principes fondamentaux :
- On accède à toute ressource via une interface uniforme (GET, PUT, POST, DELETE).
- Toute ressource est identifiée par une URI.

Types d'opérations SOAP

- One-way : input sans output
- Request-response : input suivi de output
- Solicit-response : output suivi de input
- Notification : output sans input

Vision SOAP

- Request-response :
  - retourne et met à jour une ressource

```
interface IAccount {
  [OperationContract]
    int GetBalance(int account);
  [OperationContract]
    int UpdateBalance(int account, int amount);
}
```

Vision REST

- Request-response :
  - retourne et met à jour une ressource

```
interface IAccount {
 [Méthode Get] (string account)
  Retour : Balance;
 [Méthode Post] (string account,int amount)
  Mettre à jour ;
}
```

Opération : logique

- La logique de certaines opérations ne correspond par nécessairement à une action CRUD sur une seule ressource.
- L'invocation de service + méthode est nécessaire.
- Exemple :
  `Transfer(FromAccount, ToAccount, Amount)`

Fondamental

Service	Base
WS-*	Expose des opérations nommées Différentes applications différentes interfaces Chaque opération a sa logique
REST	Expose des ressources Chaque application a la même interface

Description

Service	Description
WS-*	Langage : WSDL Avantage : des outils peuvent créer des APIs clientes Travail de développeurs de plus haut niveau
REST	Langage : Non Inconvénients : pas d'outils disponibles Option 1 : Clients écrivent des appels HTTP bruts Option 2 : Un service RESTful fournit une bibliothèque pour un client Travail de développeurs de plus bas niveau

Annuaire

Service	Annuaire
WS-*	Oui : UDDI Avantage : possiblité de découverte des services sur le web. Possibilités de comparaison entre services.
REST	Non Inconvénients : difficultés pour découvrir les services Option : page particulière servant de description Des crawlers particuliers peuvent indexer ces pages.

Transport

Service	Transport
WS-*	HTTP, TCP, autres
REST	HTTP
Service	Protocole
WS-*	SOAP
REST	HTTP
Service	Données
WS-*	XML
REST	XML, JSON, autres

Sommaire

Système urbanisé

- Système d'information urbanisé : SI comprenant des sous-systèmes indépendants, échangeant uniquement selon leurs interfaces.
  - Un sous-système peut être refondu sans perturber le système à condition que ses interfaces avec les autres sous-systèmes soient préservées.

Encapsulation des données

- Les architectures modernes retiennent le principe d'encapsulation des données : les données, leur modélisation et leurs outils de stockage sont purement internes à un sous-système, totalement invisibles de l'extérieur.
  - les données de chaque sous-système ne peuvent pas être atteintes par d'autres sous-systèmes sans utiliser les interfaces des services qui sont exposés.

SOA

- SOA (Service Oriented Architecture) est un modèle d'architecture fondé sur la notion de service.
  Un service est une fonction qui peut être invoquée à distance, par un humain ou bien par un programme.

Service

- a une granularité moyenne : plus qu'une simple routine, mais pas non plus un applicatif complet.
- a vocation à être réutilisé, il n'est pas dédié à une utilisation unique.
- est technologiquement neutre ; il peut être invoqué par toutes sortes de programmes, depuis tout type d'environnement (langage, système d'exploitation, framework).

Service

- est potentiellement accessible depuis l'extérieur du système.
- est défini par son contrat de service et d'interface.
- masque totalement son implémentation ; on peut la changer sans impacter les clients, tant que son interface est inchangée.

Mode d'interaction

- Synchrone : le client est bloqué en attente de la réponse à un message (appel de fonction à distance).
- Asynchrone, one-way (aller-seul) : le client adresse son message de requête sans attendre de réponse.

Mode d'interaction

- Asynchrone with callback : le client adresse son message de requête sans attendre de réponse mais spécifie un service à invoquer lorsque la réponse parviendra ; client et service ne sont pas bloqués.
- Asynchrone, publish / subscribe (publication / abonnement) : des services clients s'abonnent à un flux de messages ; le service producteur publie ses messages sur le flux ; asynchrone one-way à destinataires multiples.

Message-Oriented Middleware : MOM

- Middleware à base de messages ; une application interagit avec une autre application en lui adressant des messages ; le MOM se charge d'acheminer les messages de l'une à l'autre.

Sommaire

Web

- Pas d'agent Web
- Un agent contient un aspect pro-actif
- Un service n'encapsule pas cet aspect
- Ce pourrait être l'application propriétaire qui met à jour les informations qui jouerait ce rôle.
- Un SMA peut être vu comme une orchestration de services à des fins particulières.

Agent proxy

- Un agent proxy est utilisé lorque un composant a besoin d'utiliser une fonctionnalité fournie par un service extérieur. Il est nécessaire de spécifier un agent pour gérer la sémantique de la communication avec ce service particulier.

Agent proxy

- Un agent proxy isole et situe l'appel à des services extérieurs pour le système, et doit fournir aussi des services additionnels tels que le mapping entre le format de données requis par le service et le format adopté par le système.
- L'agent service encapsule la partie client, et ainsi les détails de bas niveau de la communication avec le service.

Agent : Description

Description (Interface)	Non Assez limitée
Annuaire	Oui mais interne à une plateforme agents
Référence	Similaire à une URI
Transport	dépend de la plate forme
Communication	Message (MOM)
Type	s'apparente à de l'appel RPC Terme (action) + paramètres Certains actes de langages proches de REST

Agent : Représentation

Langage de communication	Standards peu utilisés KQML, SL0
Sémantique	basé sur des listes (LISP) Utilisation d'ontologies pour décrire les ressources
Client - Serveur	chaque agent est potentiellement Client-serveur
Session	Certains actes de langages impliquent des sessions (conversationId)

Exemple (Communication)