Chapitre 3 :
Service Web

Plan

- Définitions
- Architecture Orientée service
- XML
- SOAP
- WSDL
- UDDI

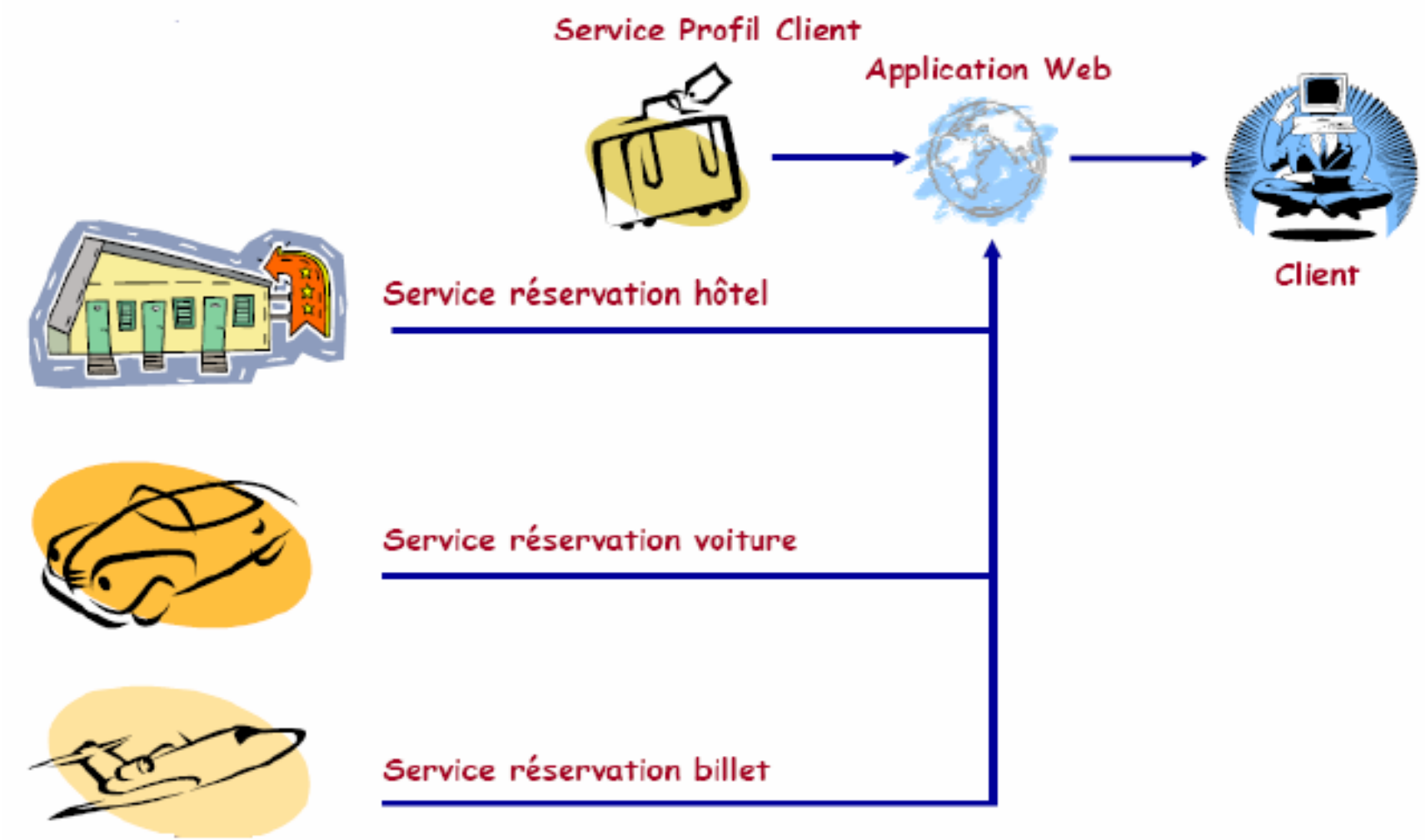
Définitions

- Les Web Services sont des services offerts via le web.
- "Un Web Service est une application logicielle identifiée par un URI (Uniform Resource Identifier), dont les interfaces et les associations peuvent être définies, décrites et découvertes par des méthodes XML, et qui peut interagir directement avec d'autres applications en utilisant des messages XML via les protocoles Internet standards.«
- Exemples d'applications
  - Commerce électronique
  - Accès à des bases de données distantes
  - Informations météo
  - ...

Caractéristiques

- Un Service Web est une « unité logique applicative » accessible en utilisant les protocoles standard d'Internet.
- Un objet métier qui peut être déployé et combiné sur Internet avec une faible dépendance vis-à-vis des technologies et des protocoles.
- Il s'agit donc d'un ensemble de fonctionnalités exposées sur internet ou sur un intranet, par et pour des applications ou machines, sans intervention humaine.
- Possibilité d'invoquer une fonction sur un serveur web distant
- Fournit une infrastructure souple pour les systèmes distribués, basée sur XML

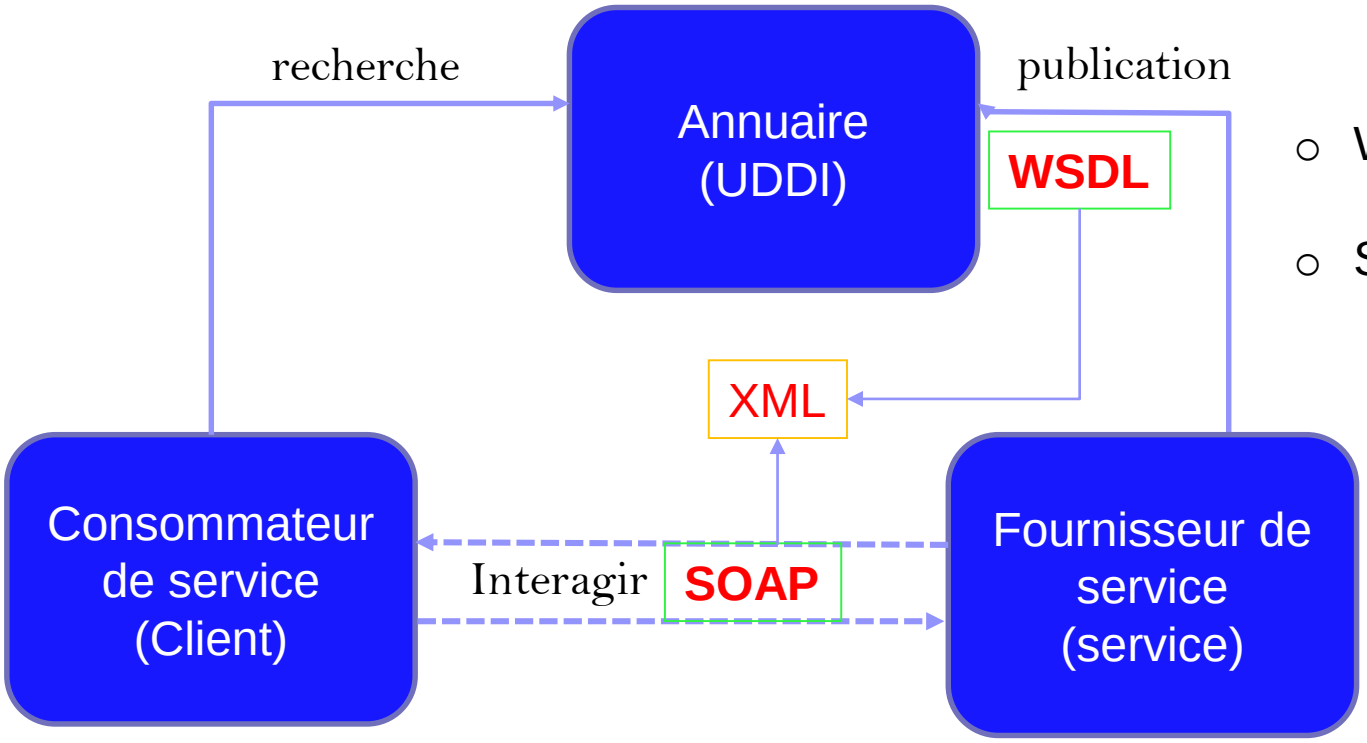
Exemple d'une service web



Les étapes d'utilisation d'une service web

1. Interroger un annuaire : quel est le fournisseur du service ?
2. Négocier avec les fournisseurs potentiels
    - Nature exacte du service fournis
    - Qualité/coût/etc.
3. Interagir avec le fournisseur du service choisi
    - Connaître les modalités d'interaction
    - Introduire le service dans ma chaîne de traitements
4. Eventuellement composer des services
5. Eventuellement publier mes propres services

Architecture orientée service (SOA)



- WSDL : web service definition language
- SOAP : Simple Object access protocol

Les composants de SOA sont :

- ❑ L'annuaire,
- ❑ Le fournisseur du Service
- ❑ Le consommateur de service (le client)

Ces composants ont des plateformes et matérielles et logicielles hétérogènes

La communication et l'échange se fait en langage connu par toutes les plateformes

→ **XML**

SOA | Technologies

- L'architecture des services web repose essentiellement sur les technologies suivantes:

SOAP
W3C
Simple Object
Access Protocol

Transporte

WSDL
W3C
Web Services
Description Language

Décrit le contrat

UDDI
Microsoft,IBM, HP
Universal Description
Discovery and Integration

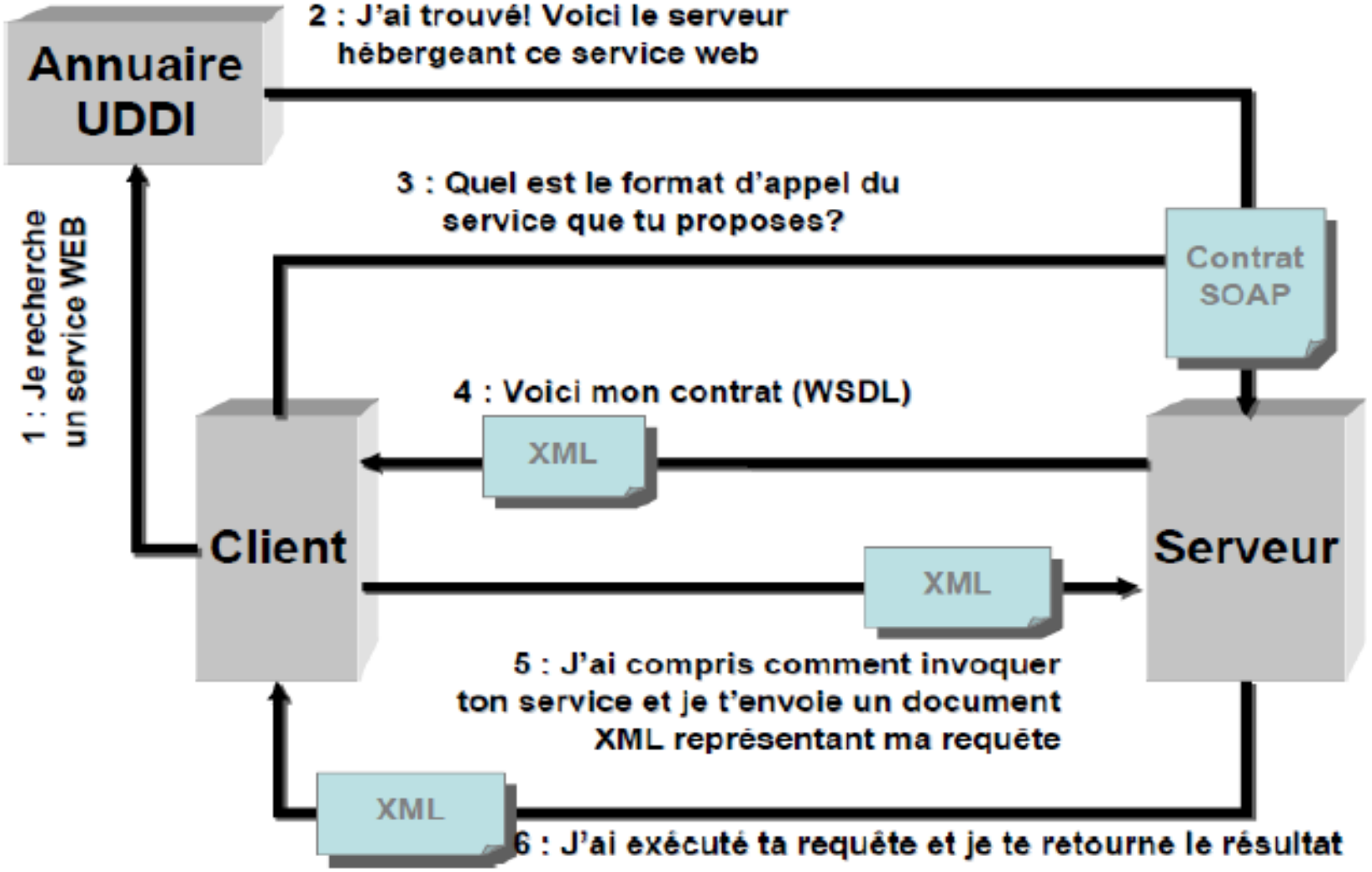
Stocke les descriptions de contrat

XML 1.0
eXtensible Markup Language

SOA | Technologies

- SOAP
  - Protocole d'échange de messages (client / serveur)
  - Message SOAP dans un enveloppe SOAP (en-tête et corps)
  - Basé entièrement sur XML
  - Porté sur HTTP, SMTP, …
- WSDL
  - Langage de définition de Web Services
  - Basé entièrement sur XML
  - Définition de l'interface, de l'URL et du port du Web Service.
  - Utilise le système de typage de XML Schéma
- UDDI
  - Référentiel de définitions de Web Service
  - Référentiel défini lui-même en WSDL
  - Référentiel Public / Privé

Un Service Web en action



XML

Extensible Markup Langage

XML | Exemple

```
<livre>
        <titre> le super livre </titre>
        <chapitre>
                <numero> 1 </numero>
                <titre> titre du chapitre 1 </titre>
                <contenu> blabla blabla </contenu>
        </chapitre>
        <chapitre>
                …
        </chapitre>

</livre>
```

XML | Principes

- Ensemble non fini de balises
  - L'utilisateur peut créer de nouvelles balises
- Définition de grammaires : XML est un Meta-Langage
  - MathML, NewsML, XMI, Doc, Slides, …
- Séparation de la forme et du fond
  - Un document XML peut être constitué de deux entités (le fond et la forme)

Grammaire

Deux façons de définir une grammaire XML :

- DTD
  - Langage de définition de grammaire XML
  - Expression faible (type, structure)
- XML Schéma
  - Langage XML de définition de grammaire XML
  - Expression puissante (**type**, structure, héritage)

Un document XML est dit **valide** lorsqu'il est **conforme** à une grammaire,

bien formé lorsqu'il respecte la syntaxe d'un document XML (pas d'erreurs d'écriture par exemple)

XML |DTD

- Le DTD ou Document Type Declaration ou encore Document Type Definition est l'ensemble des règles et des propriétés que doit suivre le document XML.
- Ces règles définissent généralement le nom et le contenu de chaque balise et le contexte dans lequel elles doivent exister.
- Cette formalisation des éléments est particulièrement utile lorsqu'on utilise de façon récurrente des balises dans un document XML.
- DTD interne ou externe
  - Interne : DTD peut être intégré au code source du fichier XML (*.xml).

  → le document XML est indépendant (standalone = 'Yes'
  - Externe : DTD externe :DTD dans un autre fichier (*.dtd).

  → Pour un document XML (.xml), il y a un fichier (.dtd)

XML |Exemple d'un DTD interne

```
<?xml version="1.0" Sandalone="yes"?>
<!DOCTYPE parent [
<!ELEMENT parent (garçon,fille)>
<!ELEMENT garçon (#PCDATA)>
<!ELEMENT fille (#PCDATA)>
]>

<parent>

<garçon>Loic</garçon>
   <fille>Marine</fille>
</parent>
```

Début du DTD interne avec parent comme élément de racine.

Cet élément racine soit parent contiendra les sous-éléments garçon et fille.

#PCDATA : l'élément fille contient des données exprimées en chiffres ou en lettres. Idem pour garçon .

Racine du document XML.

Fin du document XML.

XML | DTD externe

Le DTD externe suivra la syntaxe suivante :

<!DOCTYPE élément-racine SYSTEM "nom_du_fichier.dtd">

Le même exemple:
Le fichier XML (*.xml) : les données + le nom du fichier de DTD (*.dtd)

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE parent SYSTEM "parent.dtd">
<parent>
    <garçon>Loic</garçon>
    <fille>Marine</fille>
</parent>
```

Le fichier de DTD externe (ici dans le même répertoire) "parent.dtd" :

```
<!ELEMENT parent (garçon, fille)>
<!ELEMENT garçon (#PCDATA)>
<!ELEMENT fille (#PCDATA)>
```

XML | DTD | déclarer un élément

- Pour pouvoir créer un document XML il est utile dans un premier temps de définir les éléments pouvant être utilisés dans le DTD.
- Ainsi pour définir un élément on utilisera la syntaxe suivante :

<! ELEMENT Nom_élément (type_ou_règle)>

Type prédéfini	Description
ANY	L'élément peut contenir des éléments fils (dans n'importe quel ordre d'une façon infinie), du texte ou peut être vide
EMPTY	L'élément doit obligatoirement être un élément vide
(#PCDATA)	L'élément doit contenir une chaîne de caractères (pas d'éléments fils)

XML | DTD | déclarer un élément

- Il est possible de définir des règles d'utilisation pour chaque élément, ex : les noms des éléments qu'un élément doit contenir, des règles sur la présence d'un élément

Opérateur	Signification	Exemple
+	L'élément doit être présent au minimum une fois (1 ou plusieurs fois)	A+
*	L'élément peut être présent plusieurs fois (ou aucune) (0 ou plusieurs fois)	A*
?	L'élément peut être optionnellement présent (0 ou 1 fois)	A?

XML | DTD | déclarer un élément

- Autoriser un certain nombre de répétitions au niveau d'un élément.

Opérateur	Signification	Exemple
|	L'élément A ou l'élément B peuvent être présents (exclusivement)	**A|B**
Virgule ,	L'élément A doit être présent et suivi de l'élément B (dans l'ordre)	**A,B**
()	Les parenthèses permettent de regrouper des éléments afin de leur appliquer les autres opérateurs	**(A,B)+**

XML | DTD | déclarer un élément

```
<!ELEMENT personne (nom,prenom, telephone, email?) >
<!ELEMENT nom (#PCDATA) >
<!ELEMENT prenom (#PCDATA) >
<!ELEMENT telephone (#PCDATA) >
 <!ELEMENT email (#PCDATA) >
```

La racine du document XML est personne, elle contient obligatoirement les éléments nom, prénom, telephone qui sont tous des chaînes de caractères (#PCDATA) . La présence de l'élément email est optionnel.

XML | DTD | déclarer un attribut

<! ATTLIST Elément Attribut Type >

Avec **Type** représente le type de données de l'attribut:

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

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

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

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

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

XML | DTD | déclarer un attribut

- Enfin chacun de ces types d'attributs peut être suivi d'un mot clé particulier permettant de spécifier le niveau de nécessité de l'attribut :
- **#IMPLIED** : l'attribut est optionnel, non obligatoire
- **#REQUIRED** : l'attribut est obligatoire
- **#FIXED "val":** l'attribut sera affecté d'une valeur par défaut s'il n'est pas défini. Il doit être immédiatement suivi de la valeur entre guillemets
- **"val":** la valeur par défaut de l'attribut

SOAP

Simple Object Access Protocol

SOAP | Caractéristiques

- SOAP est un protocole de transmission de messages.
- Indépendant de la plateforme (windows, unix, mac, …)
- Définit un ensemble de règles pour structurer des messages principalement pour exécuter des dialogues requête-réponse de type RPC (Remote Procedural Call)
- SOAP n'est pas lié à un protocole de transport donné, mais il est souvent porté sur HTTP.
- Un message SOAP est un document XML ayant la forme suivante:
  - Une déclaration XML (optionnelle)
  - Une enveloppe SOAP (élément racine) composée de:
    - Une entête SOAP (Header) .
    - Un corps SOAP (body).Basé sur XML et les namespaces.

SOAP | Exemple (Requête SOAP)

```
POST /InStock HTTP/1.1
Host: www.stock.org
Content-Type: application/soap; charset=utf-8

<?xml version="1.0">
<soap:Envelope xmlns:soap=http://www.w3.org/2001/12/soap-envelope
   soap:encodingStyle=http://www.w3.org/2001/12/soap-encoding>
   <soap:Body xmlns:m=http://www.stock.org/stock>
         <m:GetStockPrice>
               <m:StockName>IBM</m:StockName>
         </m:GetStockPrice>
   </soap:Body>
</soap:Envelope>
```

Propre au portage sur HTTP

SOAP | Exemple (Réponse SOAP)

```
HTTP/1.1 200 OK
Connection: close
Content-Type: application/soap; charset=utf-8
Date: Mon, 28 Sep 2002 10:05:04 GMT

<?xml version="1.0">
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-encoding"
    soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
    <soap:Body xmlns:m="http://www.stock.org/stock">
            <m:GetStockPricesResponse>
                    <m:Price>34.5</m:price>
            </m:GetStockPricesResponse>
    </soap:Body>
</soap:Envelope>
```

Propre au portage sur HTTP

SOAP | Balises

- Des Balises Utilisateur
  - GetStockResponse
  - StockName
  - Price
- Un Namespace Utilisateur
  - xmlns:m="Some-URI"
- Des Balises SOAP
  - Enveloppe
  - Body
- Un Namespace SOAP
  - xmlns:SOAP=http://w3.org/2001/12/soap-envelope
- Un Namespace de l'encodage SOAP
  - SOAP:EncodingStyle = http://w3.org/2001/12/soap-encoding
- Des informations dans la partie HTTP

WSDL

Web Services Description Language

Version 2.0

WSDL| Présentation

- Décrit le type d'un service web (méthodes, types des paramètres)
- Décrit les aspects techniques d'implantation d'un service web (quel est le protocole utilisé, quel est l'adresse du service)
- Balises WSDL sont :
  - **types**: cette balise décrit les types utilisés
  - **message**: cette balise décrit la structure d'un message échangé
  - **portType**: cette balise décrit un ensemble d'opérations (interface d'un service web)
    - operation: cette balise décrit une opération réalisée par le service web. Une opération reçoit des messages et envois des messages.
  - **binding**: cette balise décrit le lien entre un protocole (http) et un portType.
  - **service**: cette balise décrit un service comme un ensemble de ports.
    - port: cette balise décrit un port au travers duquel il est possible d'accéder à un ensemble d'opérations. Un port référence un Binding

WSDL| Balise Types

- Cette balise décrit les types utilisés
- Description en utilisant XML Schema.

```
<wsdl:types>
        <xs:schema
                targetNameSpace="http://www.exemple.fr/personne.xsd"
                xmlns:xs="http://www.w3.org/2001/XMLSchema">
          <xs:element name="personne">
                <xs:complexType>
           <xs:sequence>
                        <xs:element name="nom" type="xs:string" />
                        <xs:element name="prenom" type="xs:string" />
                  </xs:sequence>
                </xs:complexType>
          </xs:element>
        </xs:schema>
</wsdl:types>
```

WSDL| Balise message

- Les messages sont envoyés entre deux interlocuteurs (ex: une opération reçoit des message et envoie des messages.
- Un message est composé de plusieurs **part**
- Deux façons de définir des part
  - Soit un **part** est un élément de type simple
  - Soit un **part** est un élément XML dont le type est défini dans un XML Schema
- Part de type simple

```
<wsdl:message name="personneMsg">
    <wsdl:part name="nom" type="xsd:string" />
    <wsdl:part name="prenom" type="xsd:string" />
</wsdl:message>
```

- Part qui utilise un XML Schema

```
<wsdl:message name="personneMsg">
    <wsdl:part name="personne" element="exemple:personne" />
</wsdl:message>
```

WSDL| Balise PortType

- Un portType permet d'identifier (nommer) de manière abstraite un ensemble d'opérations.

```
<wsdl:portType name="descriptionPersonnes" >
        <wsdl:operation name="getPersonne" >
                …
        </wsdl:operation>
        <wsdl:operation name="setPersonne" >
                …
        </wsdl:operation>
</wsdl:portType>
```

WSDL| Balise PortType

- Une opération :
  Reçoit des messages : <wsdl:input …>
  Envoie des messages : <wsdl:output …> ou <wsdl:fault …>
- La présence et l'ordre des input/outputs/fault dépendent du type de l'opération.

```
<wsdl:operation name="operation_name">
    <wsdl:input name="nom_optionel" message="nom_message" />
</wsdl:operation>

<wsdl:operation name="operation_name">
    <wsdl:output name="nom_optionel" message="nom_message" />
    <wsdl:input name="nom_optionel" message="nom_message" />
    <wsdl:fault name="nom_optionel" message="nom_message" />*
</wsdl:operation>

<wsdl:operation name="operation_name">
    <wsdl:input name="nom_optionel" message="nom_message" />
    <wsdl:output name="nom_optionel" message="nom_message" />
    <wsdl:fault name="nom_optionel" message="nom_message" />*
</wsdl:operation>
```

WSDL| Balise Binding

- WSDL permet de lier une description abstraite (portType) à un protocole.
- Chacune des opérations d'un portType pourra être liée de manière différente.
- Le protocole SOAP est un des protocole qui peut être utilisé.

- Un Binding :
  - peut être identifié par un nom : **name**
  - identifie le portType : **type**

<wsdl:binding **name**="*binding_name*"
type="*nom du portType*" >
…
</wsdl:binding>

WSDL| Balise Binding

- Pour préciser que le binding est de type SOAP, il faut inclure la balise suivante :

<soap:binding transport="*uri*" style="*soap_style*" />

- Transport définit le type de transport
  - http://schemas.xmlsoap.org/soap/http pour utiliser SOAP/HTTP
- Style définit la façon dont sont crée les messages SOAP de toutes les opérations
  - rpc : Encodage RPC défini par SOAP RPC
  - document : Encodage sous forme d'élément XML.
- Pour chaque opération du portType :
  - il faut préciser l'URI de l'opération : soapAction
  - Il est aussi possible de repréciser la façon dont sont créés les messages SOAP : style
- Pour chaque message de chaque opération, il faut définir comment sera créé le message SOAP

WSDL| Balise Binding

```
<wsdl:binding type="descriptionPersonnes" >
    <soap:binding
        transport="http://schemas.xmlsoap.org/soap/http"
        style="rpc" />
    <wsdl:operation name="getPersonne">
    <soap:operation  soapAction="http://www.exemple.fr/getPersonne" />
        <wsdl:input>
            <soap:body use="encoded"
              encodingStyle="schemas.xmlsoap.org/soap/encoding"/>
         </wsdl:input>
         <wsdl:output>
            <soap:body use="encoded"
              encodingStyle="schemas.xmlsoap.org/soap/encoding"/>
         </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

WSDL| Balise Service

- Un service est un ensemble de ports
- Un port a un portType
- Dans le cadre de SOAP, un port à une adresse (qui correspond à l'adresse http)

```
<wsdl:service name="MonService">
    <wsdl:port    binding="intf:MonServiceSoapBinding">
        <soap:address
        location="http://mda.lip6.fr:8080/axis/services/MonService"/>
    </wsdl:port>
</wsdl:service>
```

UDDI

Universal Description, Discovery and Integration

UDDI

- Localiser le service dont j'ai besoin ?
  - Quels sont les fournisseurs potentiels ?
  - Lequel est le meilleur pour moi ?
- Noyau: annuaires –les pages jaunes
  - Liste d'entreprises + comment les contacter
  - Classification en catégories
  - Informations en +: protocole, coût, qualité, contrat…
- Qui contrôle l'annuaire ?
  - Par exemple: qui contrôle les catégories ? Qui peut s'enregistrer dans l'annuaire ?

UDDI

- L'UDDI a 3 rôles :
  - Pages blanches : le référentiel comporte des informations sur les fournisseurs de services.
  - Pages Jaunes : le référentiel comporte des critères de catégorisation de services.
  - Pages vertes : le référentiel comporte des informations techniques (WSDL).
- Les services d'un référentiel UDDI sont des Web Services
  - Exemple

Le référentiel UDDI de microsoft est accessible via http://uddi.microsoft.com