Web service

Master 2 MMI

Département informatique

Faculté MI , université Batna2- 2020-2021

Sommaire

- Pourquoi un service Web?
- Qu'est ce qu'un service web?
- Protocole HTTP
- Service Web SOAP
- Service web REST full

Service web **SOAP**

Technologie associées

- Protocole SOAP
- Langage WSDL
- Annuaire UDDI

Mise en œuvre en java (JAX-WS)

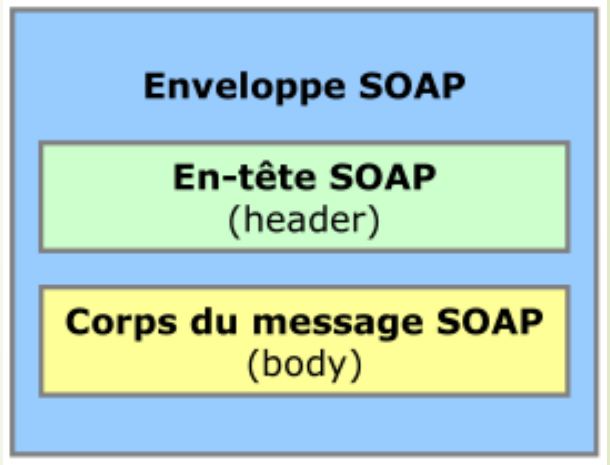
Technologies associés au service web

Document XML décrivant le service afin de rendre la solution des Web Services générique	WSDL
Protocole basé sur le standard XML pour l'échange de données structurées entre des applications réseaux	SOAP
Couche réseau (HyperText Transfer Protocol)	HTTP

Protocole SOAP (1/5)

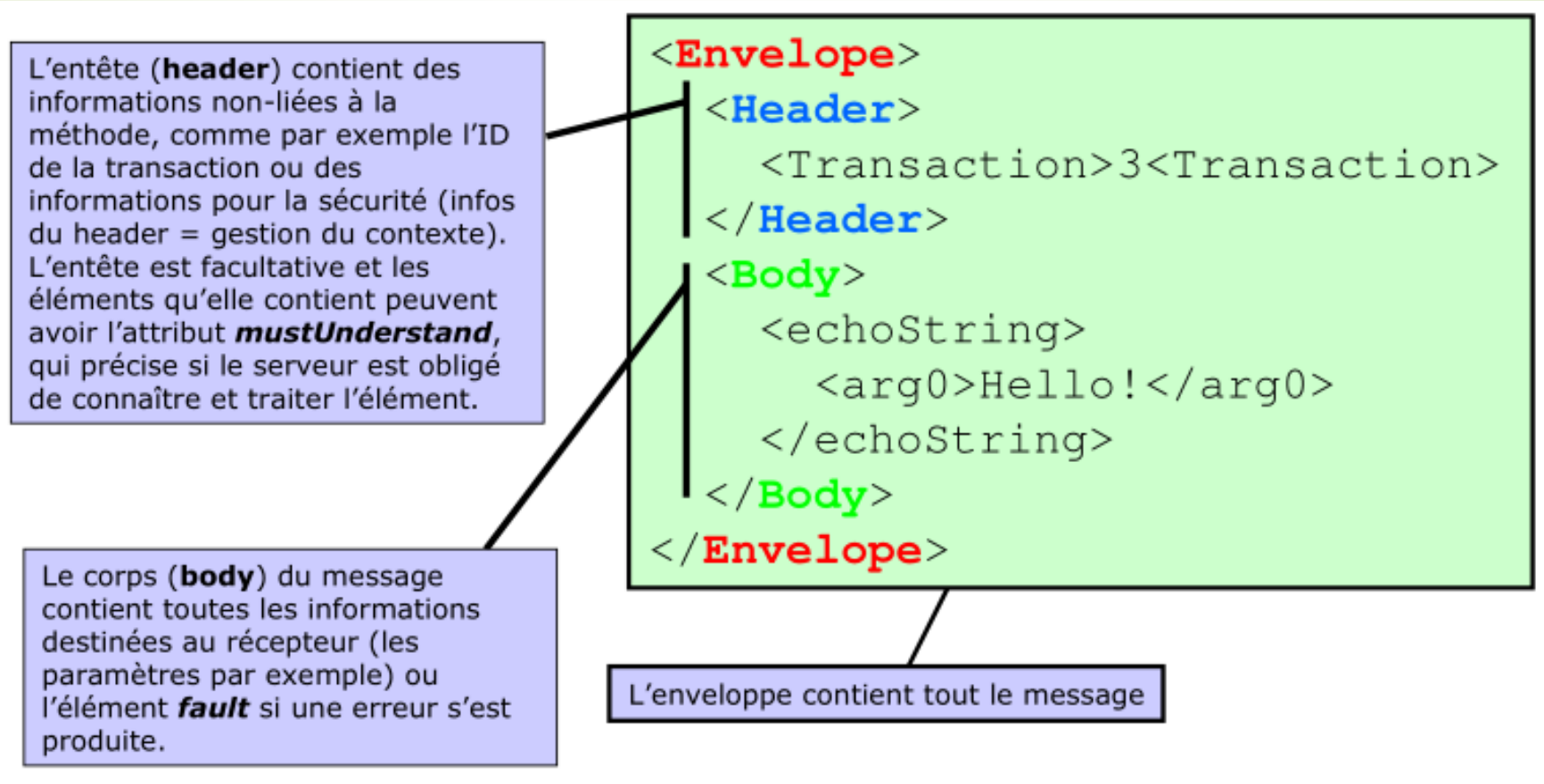
- Le protocole SOAP (Simple Object Access Protocol) assure les appels de procedures à distance au dessus d'un protocole de transport.

- Côté client: il ouvre une connexion http, il envoit la requête sous forme un document xml décrivant:
  - une méthode à invoqué sur une machine distante
  - les paramètres de la méthode
- Côté serveur : récupère la requête, puis exécute la méthode avec les paramètres , enfin une réponse SOAP est envoyé au client
- Un message SOAP est un document XML, composé d'une enveloppe qui contient une entête et le corps du message



Protocole SOAP (2/5)

- Prenons l'exemple d'un message SOAP appelant une méthode echoString(string), qui prend en paramètre une chaîne de caractères.



Protocole SOAP (3/5)

- Lorsque le serveur répond à la méthode echoString(string), il ajoute Response à la suite de la balise <echoString> (d'une manière générale, il rajoute Response à la suite de la balise contenant le nom de la requête.

- Si une erreur se produit, la réponse contient l'élément Fault (dans le corps du message).

```
<Envelope>
  <Header>
    <Transaction>3<Transaction>
  </Header>
  <Body>
    <echoStringResponse>
      <return>Hello!</return>
    </echoStringResponse>
  </Body>
</Envelope>
```

Protocole SOAP (4/5)

- Une des forces de SOAP est de permettre l'inter-opérabilité entre différentes plate-formes. Il est donc important d'avoir des règles de codage des types de données, afin que ces dernières puissent être encodées/décodées sans difficultés.
- On distingue deux types de données :

1. Les données de types simples (une chaîne de caractère par exemple) ;
2. Les types composés : structures ou tableaux.

- Dans le cas où des données binaires devraient transiter (comme une image par exemple), il est également possible d'envoyer un message SOAP avec attachement et ce grâce à un message MIME (Multimedia Internet Mail Extension).
- Pour pouvoir référencer une pièce jointe depuis le corps du message SOAP, une URI est utilisée, faisant référence à la pièce jointe.

Protocole SOAP (5/5)

Les messages précédents présentaient SOAP sans l'utilisation des espaces de noms, obligatoires d'après les spécifications du protocole.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <soapenv:Body>

    <ns1:echoStringResponse
        soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    xmlns:ns1="http://soapinterop.org/">
        <return xsi:type="xsd:string">Hello!</return>
    </ns1:echoStringResponse>

  </soapenv:Body>

</soapenv:Envelope>
```

Langage WSDL (1/3)

- WSDL (Web Service Description Langage), est un langage de description de services web en XML.
- Il décrit :

• Les informations sur les fonctions publiques du service web ;

• Les types de données utilisés durant l'échange de messages ;

• Les différents protocoles aux travers desquels le service est accessible ; et comment y accéder ;

• Une adresse permettant de localiser le service web.

- A noter : les documents WSDL ne sont jamais générés par des développeurs, mais le sont grâce à des outils qui automatisent la tâche (par exemple, il existe des outils qui prennent une classe Java et qui créent le WSDL correspondant).

Langage WSDL (2/3) exemple

```
<wsdl:definitions targetNamespace="http://192.168.0.12:8080/axis/SimpleWS.jws"/>
  <wsdl:message name="addRequest">
    <wsdl:part name="i1" type="xsd:int"/>
    <wsdl:part name="i2" type="xsd:int"/>
  </wsdl:message>
  <wsdl:message name="addResponse">
    <wsdl:part name="addReturn" type="xsd:int"/>
  </wsdl:message>
  <wsdl:portType name="SimpleWS">
    <wsdl:operation name="add" parameterOrder="i1 i2">
      <wsdl:input message="impl:addRequest" name="addRequest"/>
      <wsdl:output message="impl:addResponse" name="addResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="SimpleWSSoapBinding" type="impl:SimpleWS">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="add">
      <wsdlsoap:operation soapAction=""/>
      <wsdl:input name="addRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="http://192.168.0.12:8080/axis/SimpleWS.jws" use="encoded"/>
      </wsdl:input>
      <wsdl:output name="addResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="http://192.168.0.12:8080/axis/SimpleWS.jws" use="encoded"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="SimpleWSService">
    <wsdl:port binding="impl:SimpleWSSoapBinding" name="SimpleWS">
      <wsdlsoap:address location="http://192.168.0.12:8080/axis/SimpleWS.jws"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Décrit les messages qui circulent

Abstraction décrivant une opération

Protocole d'accès et format des messages

Définit comment est disponible (SOAP) la méthode et à quelle adresse

Langage WSDL (3/3) suite exemple

- Sur le slide précédent, quelques éléments n'ont pas été présentés. Reprenons l'élément
- <port> : il définit un point de terminaison.

```
<wsdl:service name="SimpleWSService">
    <wsdl:port binding=  "impl:SimpleWSSoapBinding"    name="SimpleWS">
        <wsdlsoap:address    location="http://192.168.0.12:8080/axis/SimpleWS.jws"/>
    </wsdl:port>
</wsdl:service>
```

- Il est à noter que l'élément <portType> peut contenir plusieurs opérations.
- A l'exemple précédent manque l'élément <type> qui permet de définir des types complexes (dans l'exemple ci-dessous, la valeur renvoyée est une chaîne de caractères).

Annuaire UDDI (1/2)

- Afin d'être découvert, un service doit être publié. Au dessue de ces trois couches de base viennent se greffer deux couches UDDI:

Découverte de services	UDDI
Publication de services	UDDI

- On publie notre service via le document **WSDL** auprès de notre annuaire
- Une application cliente peut découvrir et accéder à notre service lors de son exécution via un annuaire **UDDI**

Annuaire UDDI (2/2)

Cet annuaire contient :

- **les pages blanches** : informations générales sur une société (nom, description, adresse) ;
- **les pages jaunes** : classement des types de services ;
- **les pages vertes ou roses** : informations sur les modes d'exploitation du service.

En guise d'exemple :

- http://uddi.microsoft.com ;
- http://www.ibm.com/services/uddi.

Mise en oeuvre d'un service SOAP

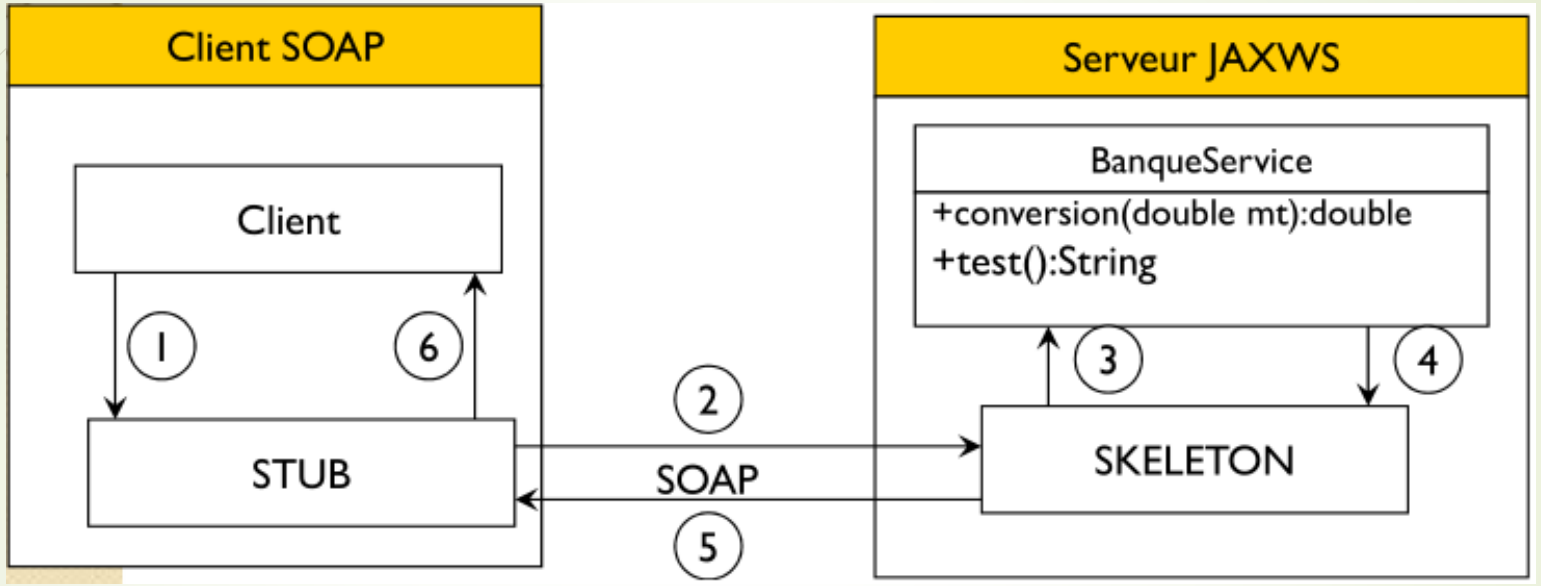
Mise en œuvre des web services avec JAX- WS

- ✓ **JAX-WS**: est une (API) librairie qui permet de développer très simplement des service web en java
- ✓ **JAX-WS**: fournit un ensemble d'*annotation* pour marquer la correspondance Java-WSDL. Il suffit pour cela d'annoter directement les classes java qui vont représenter le service web.
- ✓ Dans l'exemple du diapo suivant, une classe Java utilise des annotation JAXWS qui vont permettre par la suite de générer le document WSDL. Ce dernier est auto-généré par le serveur d'application au moment de deployement.

Mise en œuvre des web services avec JAX- WS (exemple)

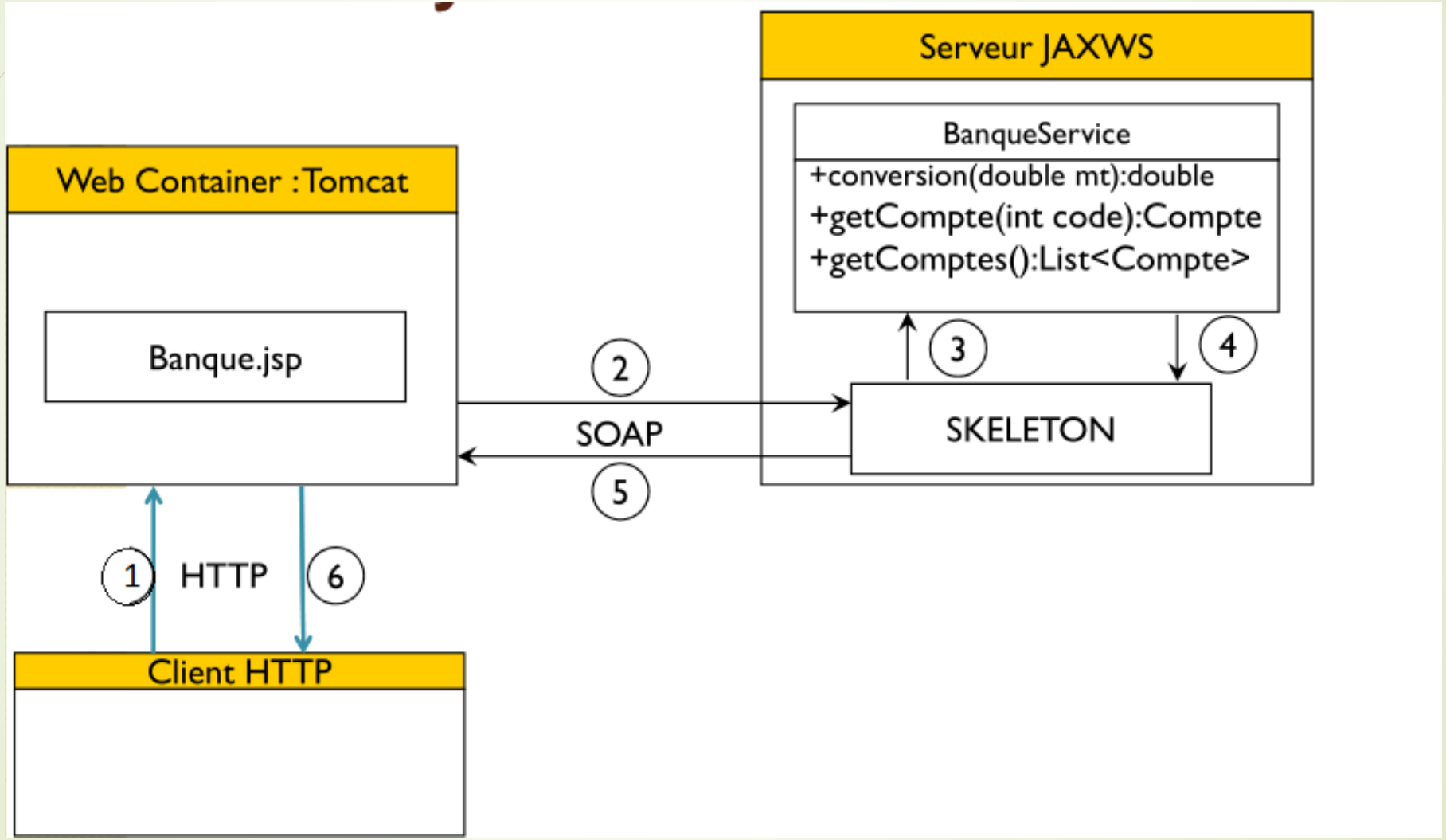
```
@WebService(serviceName="BanqueWS")
public class BanqueService {
  @WebMethod(operationName="ConversionEuroToDh")
   public double conversion@WebParam(name="montant") double mt){
          return mt*11;
   }
  @WebMethod
   public String test(){   return "Test";   }
  @WebMethod
   public Compte getCompte(){ return new Compte (1,7000);  }
  @WebMethod
   public List<Compte> getComptes(){
     List<Compte> cptes=new ArrayList<Compte>();
         cptes.add (new Compte (1,7000)); cptes.add (new Compte (2,9000));
         return cptes;
   }}
```

Architecture (client service SOAP en java)



1. Le client demande au stub de faire appel à la méthode conversion()
2. Le Stub se connecte au Skeleton et lui envoie une requête SOAP
3. Le Skeleton fait appel à la méthode du web service
4. Le web service retourne le résultat au Skeleton
5. Le Skeleton envoie le résultat dans une réponse SOAP au Stub
6. Le Stub fournie lé résultat au client

Architecture (client SOAP en jsp)



Comment développé un service web SOAP avec JAX-WS

voir TP

1. Créer le service Web
   - Développer le web service (classe +annotations)
   - Déployer le web service dans le serveur d'application (tel que Glassfich, …)
2. Tester le web service avec un analyseur :
   - SoapUI, postman,…
3. Créer application client qui consomme le service Web