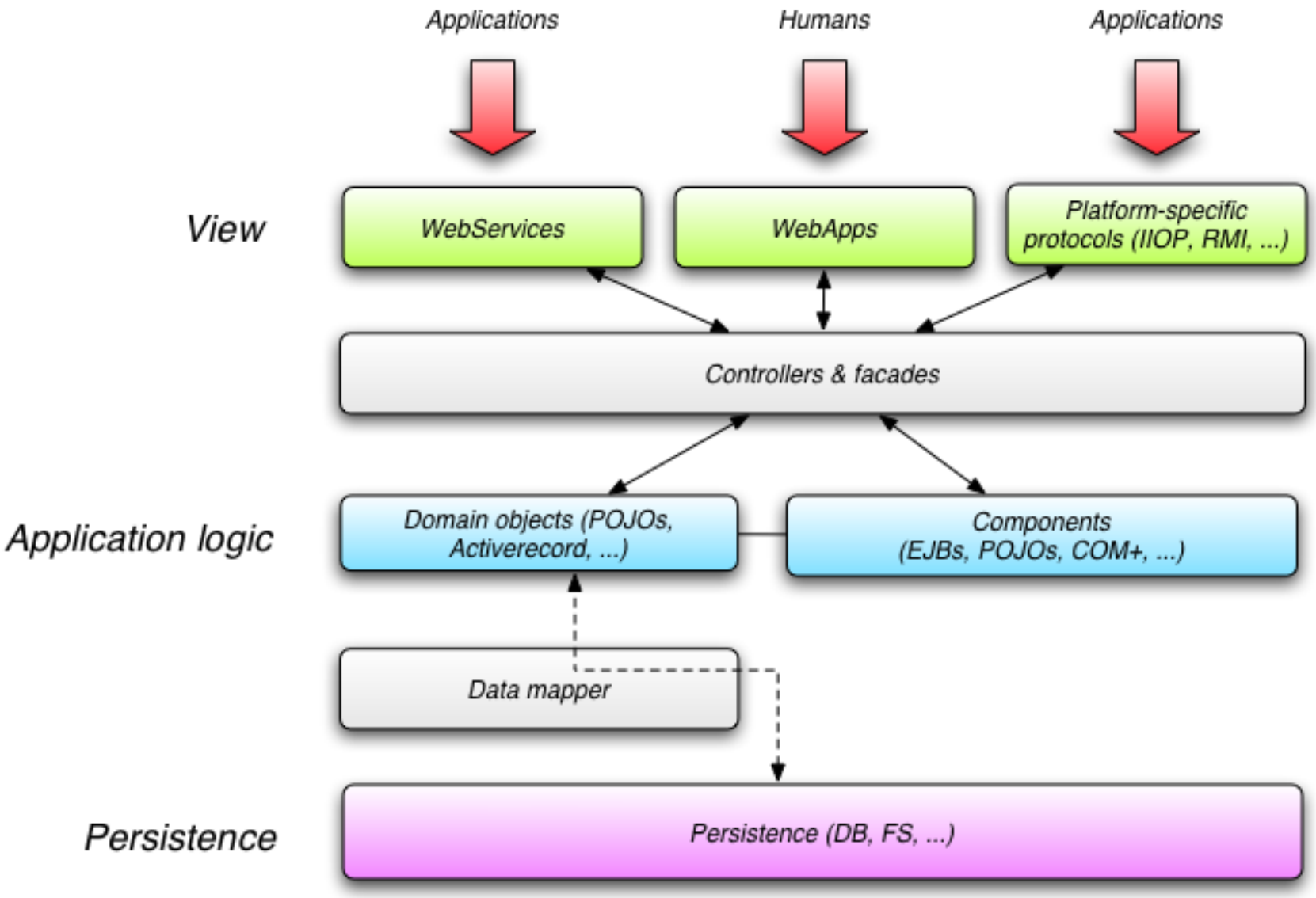
almerys
innover pour la vie

Web Services

Cours Web Services ISIMA 3F3

Olivier COUPELON -
coupelon@isima.fr



Julien Ponge

Web Services

- Communication inter-applications :
  - Technologie non adaptée à une interrogation directe par un utilisateur.
  - Couplage faible

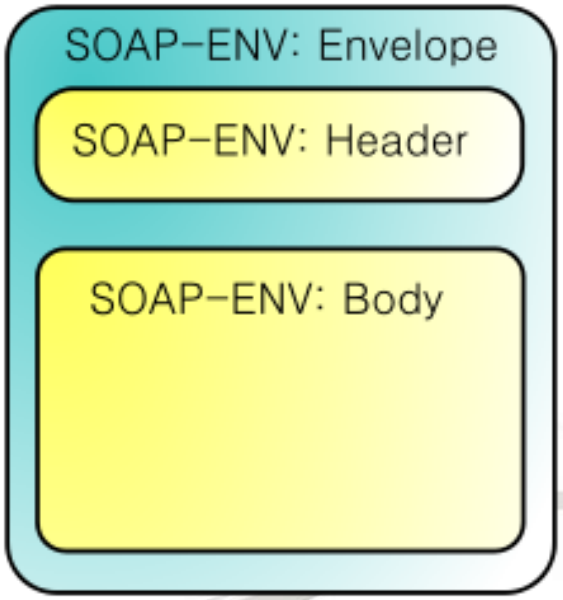
- Trois besoins :
  1. Echange de données (SOAP)
  2. Description des services d'échanges (WSDL)
  3. Découverte des services (UDDI)

Protocole SOAP

- Anciennement Simple Object Access Protocol

- Enveloppe d'échange de données, comprenant :
- Un entête (header)
- Un corps (body)

- Enveloppe transportée sur HTTP, XMPP, ou encore SMTP



Message SOAP

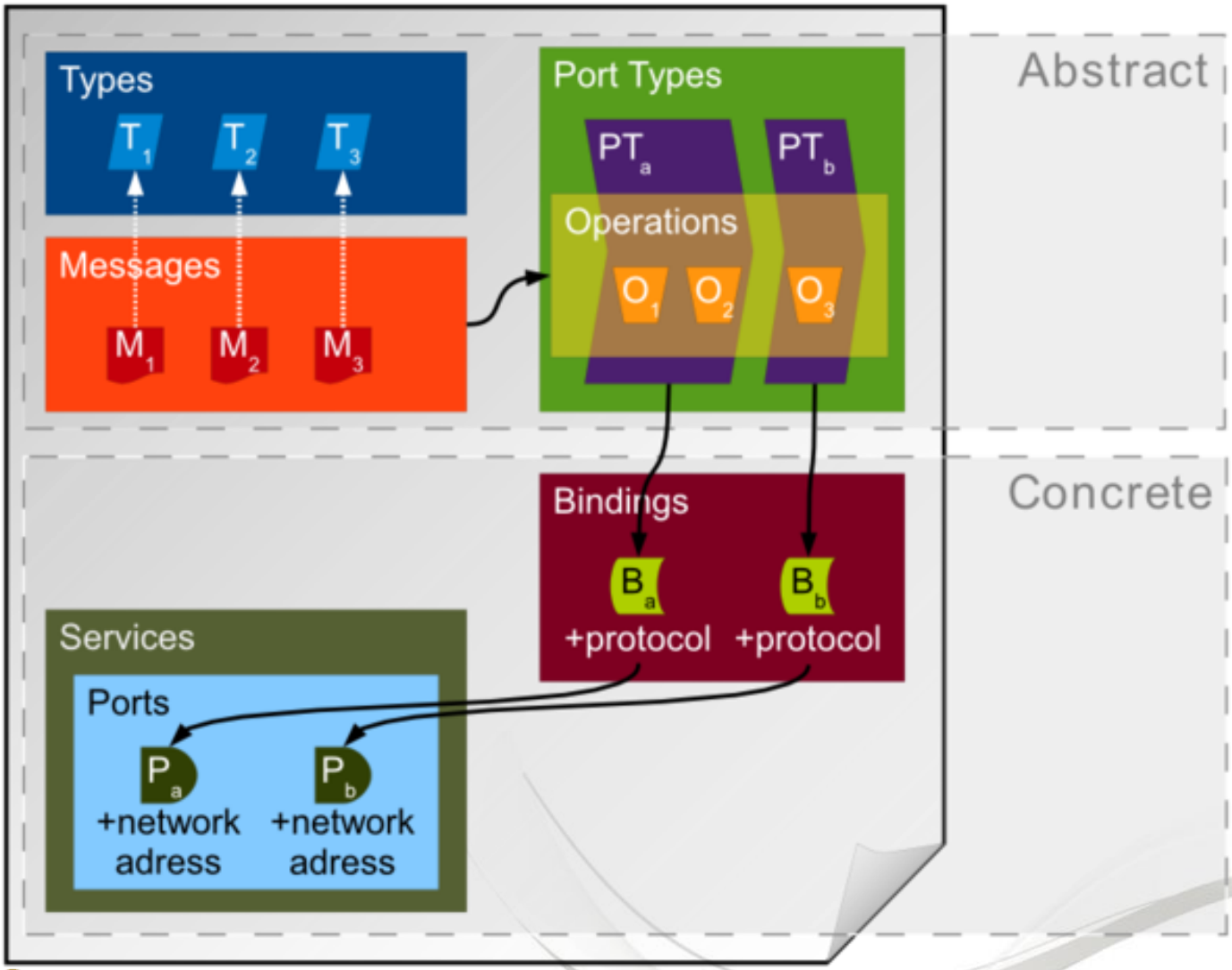
- Toutes les données échangées sont encapsulées dans une enveloppe SOAP (entrée/sortie) :

```
<?xml version="1.0" encoding="UTF-8"?>

<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
    <env:Header>
        <t:transaction xmlns:t="urn:mytransaction">
            <t:id>123456789</t:id>
        </t:transaction>
        <h:plop xmlns:h="urn:plop:da:plop" env:mustUnderstand="1">Coin Coin</h:plop>
    </env:Header>
    <env:Body>
        <s:sales xmlns:s="urn:sales">
            <s:product>Banana split</s:product>
            <s:quantity>5</s:quantity>
            <s:customer>Mr Bean</s:customer>
        </s:sales>
    </env:Body>
</env:Envelope>
```

WSDL

- **W**eb **S**ervice **D**escription **L**anguage
- Décrit les opérations disponibles d'un service web
- **Contrat** établi entre le client et le serveur

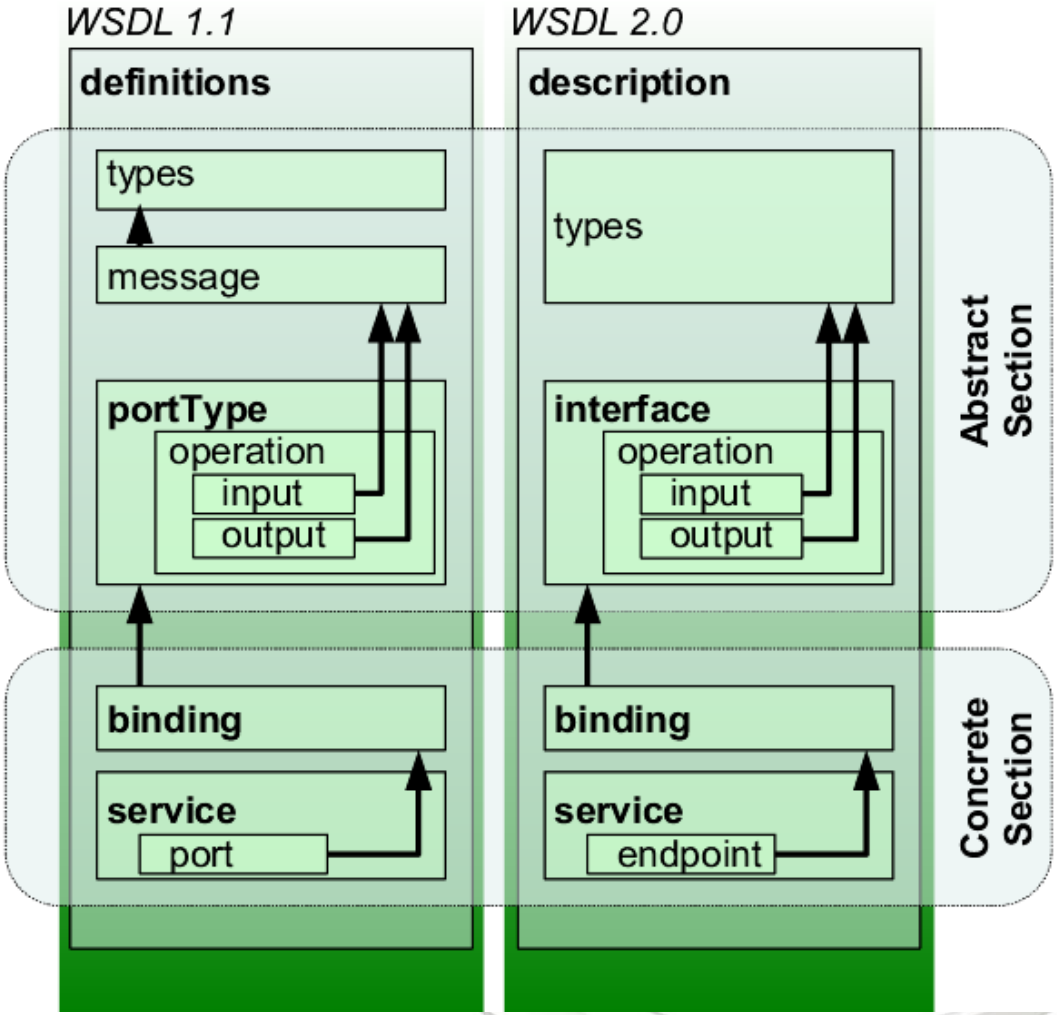
WSDL 1.1


Abstract
Types
T1
T2
T3
Messages
M1
M2
M3
Port Types
PTa
PTb
Operations
O1
O2
O3
Concrete
Bindings
Ba
Bb
+protocol
+protocol
Services
Ports
Pa
Pb
+network adress
+network adress

WSDL 1.1

```
<definitions>

<types>
  definition of types........
</types>

<message>
  definition of a message....
</message>

<portType>
  definition of a port.......
</portType>

<binding>
  definition of a binding....
</binding>

<service>
  definition of a service....
</service>
</definitions>
```

WSDL 1.1 vs 2.0

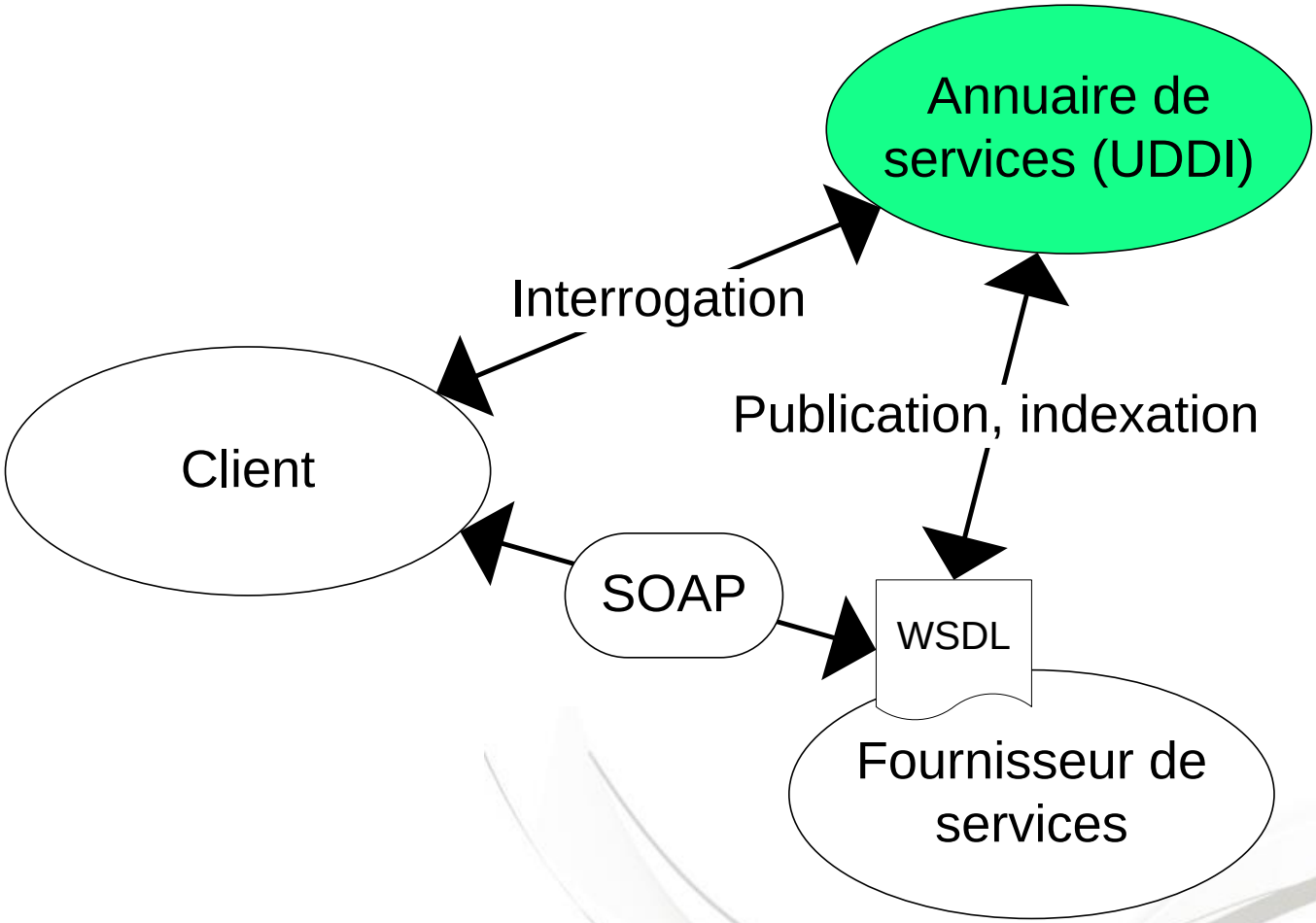

WSDL 1.1
WSDL 2.0
definitions
description
types
message
portType
operation
input
output
types
interface
operation
input
output
Abstract Section
binding
service
port
binding
service
endpoint
Concrete Section

UDDI

- **U**niversal **D**escription **D**iscovery and **I**ntegration

- Annuaire de services :
- Pages blanches : liste des enteprises, les fournisseurs de service
- Pages jaunes : liste des services (WSDL)
- Pages vertes : détails techniques des services, liaison avec les processus métiers associés

UDDI


Annuaire de services (UDDI)
Interrogation
Publication, indexation
Client
SOAP
WSDL
Fournisseur de services

UDDI – Utilisation pour l'appel de services

1. Requête vers UDDI pour obtenir les informations techniques sur un service
   - WSDL
   - Adresse
   - bindingKey : clé identifiante unique
2. Ecriture du client associé
3. En cas d'échec lors d'une utilisation, le client doit prévoir de rappeler l'annuaire avec la bindingKey afin de récupérer la nouvelle interface/adresse du service automatiquement.

Création d'un Web Service

- Top-Down :

1. Ecriture du WSDL
2. Ecriture du code associé

- Bottom-Up :

1. Ecriture du code du service
2. Génération automatique du WSDL à partir de ce code

Approche Top-Down

- Génération du code Java à partir du WSDL décrit
- Exemple en java :
- `wsimport -d src/generated http://example.org/stock?wsdl`
- `wsdl2java -d src/generated -server -client http://example.org/stock?wsdl`

Approche Bottom-Up

- Une fois le code des services écrit, on génère le WSDL
- Exemple en java avec code annoté :
- `wsgen -cp . ws.Hello`

- Avec une conteneur web et JAX-WS, ceci est automatique

Création d'un Web Service

- Technologies existantes :
- Apache Axis et Axis 2
- XFire

- JAX-WS
  - Apache CXF
  - Metro (implémentation de référence)

JAX-WS

- **J**ava **A**PI for **X**ML **W**eb **S**ervices

- JAX-WS 2.0 : Standard dans Java EE 5
- JAX-WS 2.2 : Java EE 6

- Comme JAX-RS, on utilise des **annotations** Java pour créer les services

Annotations JAX-WS

- @WebService : La classe annotée est déclarée comme étant un service

- @WebMethod : La méthode annotée doit être exposée en tant qu'opération du service
- @WebParam : Permet de spécifier les propriétés d'un paramètres

- Par défaut, les méthodes publiques de la classe sont exposées

Appeler un Web Service

- Doit toujours se baser sur le WSDL (Top-Down)
- Génération du code Java à partir du WSDL
- La plupart des IDE savent générer le code Java à partir d'un WSDL automatiquement

Appeler un Web Service

- Par injection dans un conteneur web :
- @WebServiceRef(wsdLocation=http://www.toto.fr/)
- static AppService service;

- En accès directe en utilisant les classes générées à partir du WSDL

Spécifications WS-*

- http://fr.wikipedia.org/wiki/Liste_des_spécifications_des_Services_Web_WS-*
- Les spécifications sont nombreuses, parfois non maintenues, concurrentes…
- Apporte des fonctionnalités supplémentaires à la communication Web Service (fiabilité, sécurité…)

WS-Policy

- Ajoute des informations au WSDL sur la capacité du service
- Exemple : force l'authentification.
  Doit être intégré à la partie Bindings du WSDL

```
<wsp:Policy wsu:Id="AvecUserNameToken"
    xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
    xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
    <sp:SignedSupportingTokens>
        <wsp:Policy>
            <sp:UsernameToken sp:IncludeToken=
            "http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRecipient"/>
        </wsp:Policy>
    </sp:SignedSupportingTokens>
</wsp:Policy>
```

La sécurité dans les Web Services

- Deux niveaux :

1. Transport Level Security
   - Principalement HTTPS et Authentification HTTP
2. Message Level Security
   - Différentes spécifications :
   - WS-Security
   - XML-Encryption
   - XML-Signature

WS-Security

- Permet d'ajouter une couche de sécurité aux échanges SOAP
- Utilisation de SAML, Kerberos, certificats X509…
- Exemple avec UsernameToken et Timestamp :
- Identifiant par login/mot de passe

- Ces informations sont transmises dans l'entête SOAP

WS-Security

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:val="http://prestation.almerys.com/schema/valerys">
   <soapenv:Header>
        <wsse:Security soapenv:mustUnderstand="true"
        xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility
            <wsu:Timestamp u:Id="_0" xmlns:u="u">
                <wsu:Created>2010-01-01T00:00:00.462Z</wsu:Created>
                <wsu:Expires>2999-12-31T23:59:59.462Z</wsu:Expires>
            </wsu:Timestamp>

            <wsse:UsernameToken wsu:Id="UsernameToken-32380043">
                <wsse:Username>Kermit</wsse:Username>
                <wsse:Password
                Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-
                theFrog</wsse:Password>
            </wsse:UsernameToken>
        </wsse:Security>
    </soapenv:Header>
   <soapenv:Body>

   </soapenv:Body>
</soapenv:Envelope>
```

WS-Security

- XML-Encryption : crypter le contenu des messages SOAP
- XML-Signature : Authentification des interlocuteurs

WS-Addressing

- Permet de contrôler de manière standard la communication SOAP.
- Ajoute des **M**essage **I**nformation (entêtes définies par WSA)
- Indépendant de la couche de transport
- **To** : le service cible du message
- `<wsa:To> http://host/WidgetService </wsa:To>`

WS-Addressing

- Message Information :

- **To** : le service cible du message
- `<wsa:To> http://host/WidgetService </wsa:To>`

WS-Addressing

- Message Information :

- **From** : l'emmeteur du message
- `<wsa:From>`
- `  <wsa:Address> http://client/myClient </wsa:Address>`
- `</wsa:From>`

WS-Addressing

- Message Information :

- **ReplyTo** : le service à contacter pour répondre
- `<wsa:ReplyTo>`
- `  <wsa:Address> http://client/myReceiver </wsa:Address>`
- `</wsa:ReplyTo>`

WS-Addressing

- Message Information :

- **FaultTo** : le service à contacter en cas d'exception
- `<wsa:ReplyTo>`
- `  <wsa:Address> http://client/FaultReceiver </wsa:Address>`
- `</wsa:ReplyTo>`

WS-Addressing

- Message Information :

- **MessageID** : identifiant unique du message
- `<wsa:MessageID>uuid:098765</wsa:MessageID>`

WS-Addressing

- Message Information :

- **RelatesTo** : spécifie une relation avec un autre message par son MessageID
- `<wsa:RelatesTo RelationshipType="wsa:Response">`
- `  uuid:098765`
- `</wsa:RelatesTo>`

Autres spécifications

- WS-Reliability : Permet de s'assurer qu'un message SOAP est bien a bien été livré
- WS-Transaction : Utilise WS-Coordination pour offrir des garanties transactionnelles aux services
- BPEL : **B**usiness **P**rocess **E**xecution **L**anguage