Services web

Pr. Abdelhak LAKHOUAJA

Département d'Informatique
Faculté des Sciences
Oujda

abdel.lakh@gmail.com
http://lakhouaja.oujda-nlp-team.net/

Master Ingénierie Informatique (M2I)
Année universitaire : 2016/2017

Chapitre 1
Introduction

Définitions

Définition

A web service is a network accessible interface to application functionality, built using standard Internet technologies.

Définitions

Définition

A web service is a network accessible interface to application functionality, built using standard Internet technologies.

Définition (W3C Web Services Architecture Working Group)

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Définition

- Un service Web est une « unité logique applicative » accessible en utilisant les protocoles standard d'Internet (TCP/IP, HTTP et XML).
- Une « librairie » fournissant des données et des services à d'autres applications.
- Un objet métier qui peut être déployé et combiné sur Internet avec une faible dépendance vis-à-vis des technologies et des protocoles.
- Combine les meilleurs aspects du développement à base de composants et du Web.

Caractéristiques

- Réutilisable.
- Indépendant :
  - de la plate-forme (Linux, Windows, ...) ;
  - du langage de programmation (Java, C#, php, ...) ;
  - de l'architecture (.NET, JEE, ...).

Pourquoi faire ?

- Les services Web permettent d'interconnecter :
  - différentes entreprises ;
  - différents matériels ;
  - différentes applications ;
  - différents clients.
- Dédiés aux applications B2B (Business to Business), EAI (Enterprise Application Integration), P2P (Peer to Peer).
- Vers le Web sémantique : pas uniquement le Web purement interactif

Types de services web

Il existe deux types de services web :

1. services web **SOAP** (initialement, Simple Object Access Protocol, puis, Service-Oriented Architecture (**SOA**) Protocol). Le consortium **W3C** ne le considère plus comme acronyme.
   - → SOAP utilise XML pour la communication. Le client envoi un message SOAP et reçoit un réponse de type SOAP.

Types de services web

Il existe deux types de services web :

1. services web **SOAP** (initialement, Simple Object Access Protocol, puis, Service-Oriented Architecture (**SOA**) Protocol). Le consortium **W3C** ne le considère plus comme acronyme.
   - → SOAP utilise XML pour la communication. Le client envoi un message SOAP et reçoit un réponse de type SOAP.
2. services web **Rest** (Representational State Transfer) : considère une information comme une ressource et elle est désignée par son URI (Uniform Resource Identifier). Les ressources sont manipulées par des opérations simples.

Exemple : achat via « Amazon »

Solution 1 :

achat via le site de la société `http://www.amazon.com`. Le client :

1. choisit la liste des produits ;

Exemple : achat via « Amazon »

Solution 1 :

achat via le site de la société `http://www.amazon.com`. Le client :

1. choisit la liste des produits ;
2. s'enregistre (ou se connecte à son compte) ;

Exemple : achat via « Amazon »

Solution 1 :

achat via le site de la société `http://www.amazon.com`. Le client :

1. choisit la liste des produits ;
2. s'enregistre (ou se connecte à son compte) ;
3. procède au payement par carte bancaire (ou équivalent) ;

Exemple : achat via « Amazon »

Solution 1 :

achat via le site de la société `http://www.amazon.com`. Le client :

1. choisit la liste des produits ;
2. s'enregistre (ou se connecte à son compte) ;
3. procède au payement par carte bancaire (ou équivalent) ;
4. reçoit une confirmation par émail.

Exemple : achat via « Amazon »

Solution 2 : utilisation des services web. Amazon permet l'utilisation de ses services via des services web . Le client procède comme suit :

1. il dispose d'une table (d'une base de données) ou d'un simple fichier texte qui contient une liste de livres (ou d'autres produits vendus par Amazon) ;

Exemple : achat via « Amazon »

Solution 2 : utilisation des services web. Amazon permet l'utilisation de ses services via des services web . Le client procède comme suit :

1. il dispose d'une table (d'une base de données) ou d'un simple fichier texte qui contient une liste de livres (ou d'autres produits vendus par Amazon) ;
2. la table ou le fichier, contient des informations pertinentes (telle que l'ISBN pour les livres) le nombre d'articles, ...

Exemple : achat via « Amazon »

Solution 2 : utilisation des services web. Amazon permet l'utilisation de ses services via des services web . Le client procède comme suit :

1. il dispose d'une table (d'une base de données) ou d'un simple fichier texte qui contient une liste de livres (ou d'autres produits vendus par Amazon) ;
2. la table ou le fichier, contient des informations pertinentes (telle que l'ISBN pour les livres) le nombre d'articles, ...
3. le client écrit un programme (dans un langage de programmation préféré par le client) qui lit la table ou le fichier, ouvre une connexion à Amazon et vérifie la disponibilité des articles et commande les articles disponibles ;

Exemple : achat via « Amazon »

Solution 2 : utilisation des services web. Amazon permet l'utilisation de ses services via des services web . Le client procède comme suit :

1. il dispose d'une table (d'une base de données) ou d'un simple fichier texte qui contient une liste de livres (ou d'autres produits vendus par Amazon) ;
2. la table ou le fichier, contient des informations pertinentes (telle que l'ISBN pour les livres) le nombre d'articles, ...
3. le client écrit un programme (dans un langage de programmation préféré par le client) qui lit la table ou le fichier, ouvre une connexion à Amazon et vérifie la disponibilité des articles et commande les articles disponibles ;
4. le programme du client vérifie l'émail pour la confirmation. Si tout se passe bien, le client confirme la commande.

SOA(**S**ervice **O**riented **A**rchitecture)
Architecture Orientée Service

Plusieurs solutions existent pour développer des applications distribuées :

- CORBA (Common Object Request Broker Architecture)
- DCOM (Distributed Computing Object Model) propriété de Microsoft
- Java RMI (Remote Method Invocation)
- ...

SOA(**S**ervice **O**riented **A**rchitecture)
Architecture Orientée Service

Plusieurs solutions existent pour développer des applications distribuées :

- CORBA (Common Object Request Broker Architecture)
- DCOM (Distributed Computing Object Model) propriété de Microsoft
- Java RMI (Remote Method Invocation)
- ...

Le problème avec toutes ces solutions c'est :

- la représentation des données propre pour chaque solution
- pas d'interopérabilité entre les différentes solutions
- pas de protocole de transport standard

SOA

SOA permet de résoudre les problèmes précédents.

Comme la programmation orientée objets (POO), SOA n'est pas une architecture, c'est un concept. C'est une façon de réfléchir.

SOA : concepts

Les trois point clés de SOA, sont :

1. Services.
2. Interopérabilité.
3. Couplage faible.

SOA : services

Définition (W3C http://www.w3.org/TR/ws-gloss/)

A service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers entities and requesters entities. To be used, a service must be realized by a concrete provider agent.

SOA : services

Définition (W3C http://www.w3.org/TR/ws-gloss/)

A service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers entities and requesters entities. To be used, a service must be realized by a concrete provider agent.

Définition (traduction)

Un service est une ressource abstraite capable d'exécuter des tâches qui forment une fonctionnalité cohérente du point de vue des fournisseurs et des consommateurs. Pour être utilisé, un service doit être réalisé par un fournisseur concret.

SOA : Interopérabilité

Permettre à des systèmes hétérogènes de se connecter de façon facile.

C'est la base pour implémenter des services partagés sur plusieurs systèmes distribués.

SOA : Couplage faible

La communication entre les différents systèmes doit être indépendante de l'implémentation du service. Elle doit se faire à travers des messages dans le but de garantir la :

- flexibilité ;
- scalabilité (évolution) ;
- tolérance aux pannes.

Quelques ressources

- Mickael BARON
  Support de cours sur JAX-WS : Développez des Web Services étendus avec Java
  http://mbaron.developpez.com/soa/jaxws/
- The Java EE 7 Tutorial
  Part VI Web Services (Chapters 27-31)
  (http://docs.oracle.com/javaee/7/tutorial/doc/
- Martin Kalin "Java Web Services : Up and Running, Second Edition", O'Reilly, 2013.
- K. Watson, C. Nagel, J.H. Pedersen, J. Reid, M. Skinner "Beginning Visual C# 2010", Wiley, 2010.
- W3C, ...

TP

- Langages de programmation :
  Java, C#
  D'autres langages peuvent être utilisés.
- Logiciels utilisés :
  - Navigateur (FireFox ou autres)
  - NetBeans
  - Glassfish
  - VisualStudio (2010 ou plus récent)
  - SoapUI : `http://www.soapui.org/`

Chapitre 2

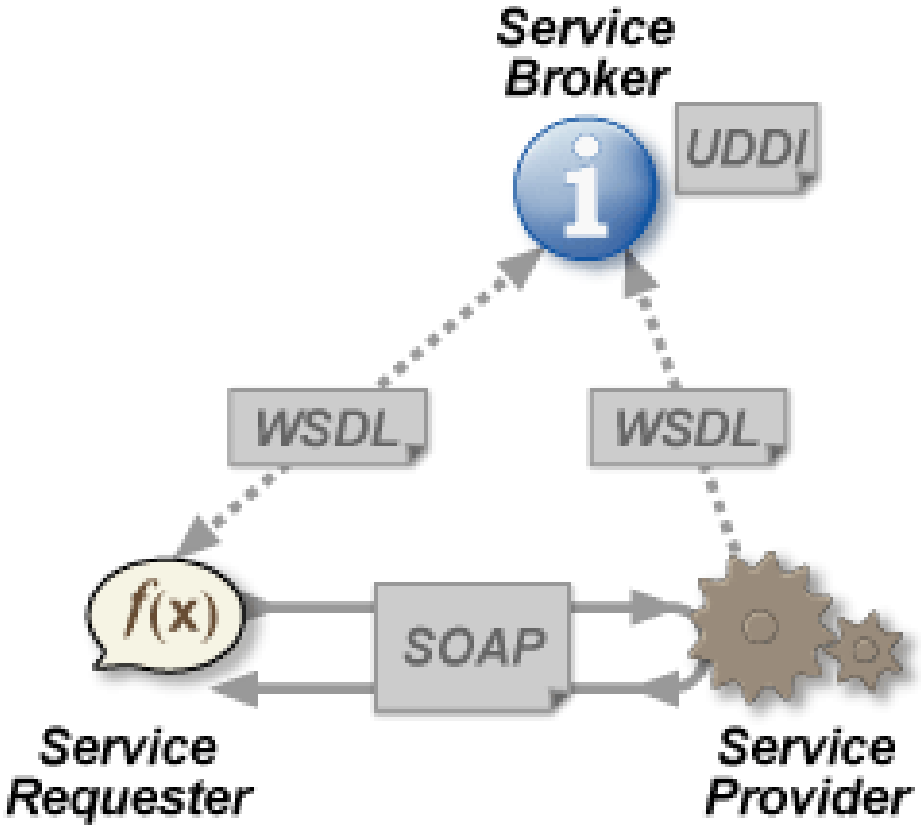
Services web SOAP ou étendus

SOAP

Une Architecture Orientée Service (**SOA**)

- Les services web SOAP répondent aux exigences de **SOA**.
- Le fournisseur de service créé le service Web, puis publie son interface ainsi que les informations d'accès au service, dans un annuaire de services Web.
- L'annuaire de service rend disponible l'interface du service ainsi que ses informations d'accès, pour n'importe quel demandeur potentiel de service.
- Le consommateur de service accède à l'annuaire de service pour effectuer une recherche afin de trouver les services désirés. Ensuite, il se lie au fournisseur pour invoquer le service.

Vue générale d'un services web



http://en.wikipedia.org/wiki/Web_service

Les quatre principales technologies utilisées

1. Langage XML : décrit les informations.
2. Protocole SOAP : pour la communication entre les services distants.
3. Langage WSDL : décrit l'interface des services.
4. Norme UDDI : trouve les services dont on a besoin.

Généralités sur **SOAP**

- SOAP a été initialement défini par Microsoft et IBM, puis est devenu depuis une recommandation du W3C.
- Initialement, **SOAP** désignait **S**imple **O**bject **A**ccess **P**rotocol. Par la suite, le consortium W3C le considère désormais comme un nom propre.
- SOAP est un protocole de communication provenant du monde XML.
- Un service SOAP est une fonction que l'on peut appeler depuis un client sur un serveur distant. Cette fonction peut en théorie accepter n'importe quelle structure de données et donner une réponse tout aussi complexe. Il est également envisageable qu'il n'y ait pas de réponse.

Généralités sur SOAP

- SOAP ne fonctionne pas tout seul, il s'appuie sur un autre protocole pour fonctionner. À priori, n'importe quel protocole de communication peut faire l'affaire (HTTP, SMTP, FTP). Dans les faits, le protocole HTTP est le plus utilisé car il correspond bien aux besoins et aux architectures en place. Son fonctionnement avec le protocole SOAP est normalisé par le W3C.
- SOAP implique souvent un dialogue qui se compose de l'appel à une procédure et d'une réponse de cette dernière.

Messages SOAP

- Tous les messages SOAP sont encodés en XML. Une application SOAP doit inclure le namespace SOAP adéquat pour tous les éléments et les attributs définis par SOAP. Elle doit pouvoir supprimer les messages qui ont un namespace incorrect.
- SOAP définit deux namespaces :
  - l'enveloppe SOAP : `http://schemas.xmlsoap.org/soap/envelope/`
  - Les règles d'encodage : `http://schemas.xmlsoap.org/soap/encoding/`

Exemple : requête SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/
    envelope/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
    soap/envelope/">
     <SOAP-ENV:Header/>
     <S:Body>
          <ns2:somme xmlns:ns2="http://cours/">
               <x>10.0</x>
               <y>200.0</y>
          </ns2:somme>
     </S:Body>
</S:Envelope>
```

Exemple : réponse SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.
    org/soap/envelope/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
    soap/envelope/">
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:sommeResponse xmlns:ns2="http://
            cours/">
            <return>210.0</return>
        </ns2:sommeResponse>
    </S:Body>
</S:Envelope>
```

Structure d'un message SOAP

La requête et la réponse ont la même structure :

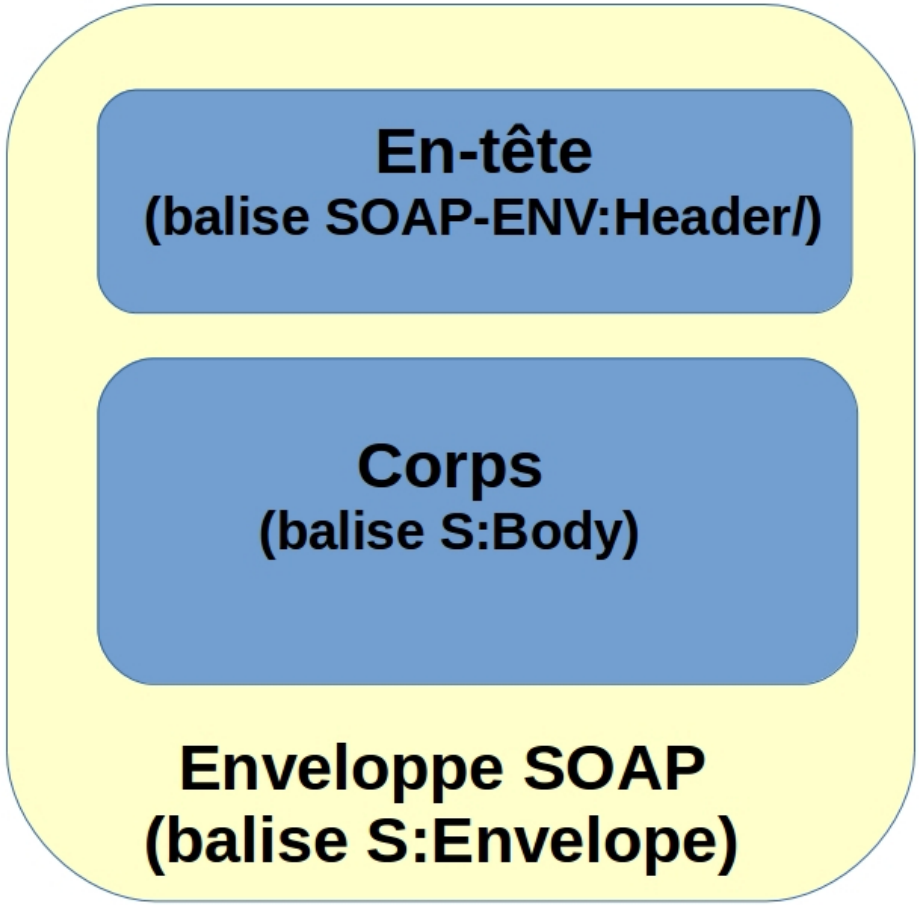
```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.
    org/soap/envelope/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
    soap/envelope/">
     <SOAP-ENV:Header/>
     <S:Body>
          ...
          Contenu
          ...
     </S:Body>
</S:Envelope>
```

Structure d'un message SOAP

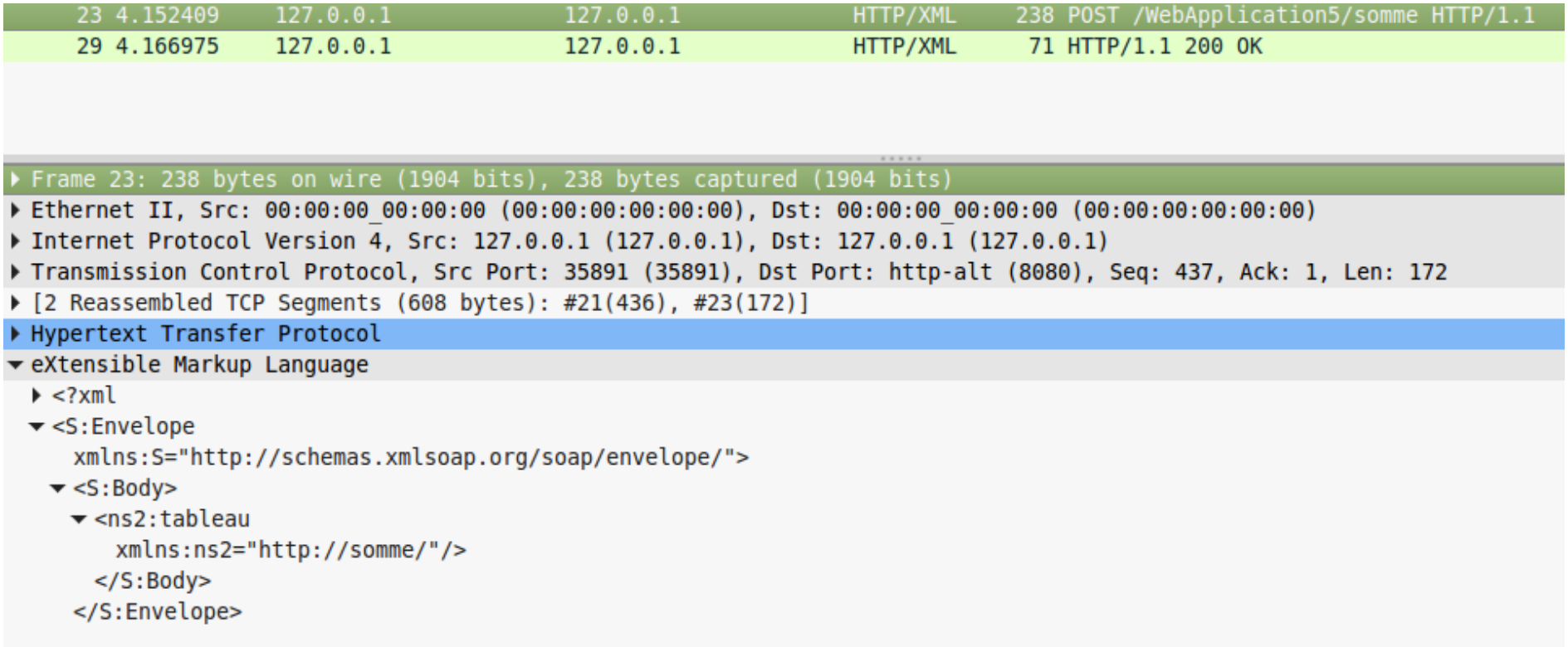
- Une déclaration XML (optionnelle),
- Une enveloppe SOAP (l'élément racine) qui est composée d' :
  - un en-tête SOAP (optionnel)
  - un corps SOAP

C'est à l'intérieur du corps (body) que l'on trouve le contenu.

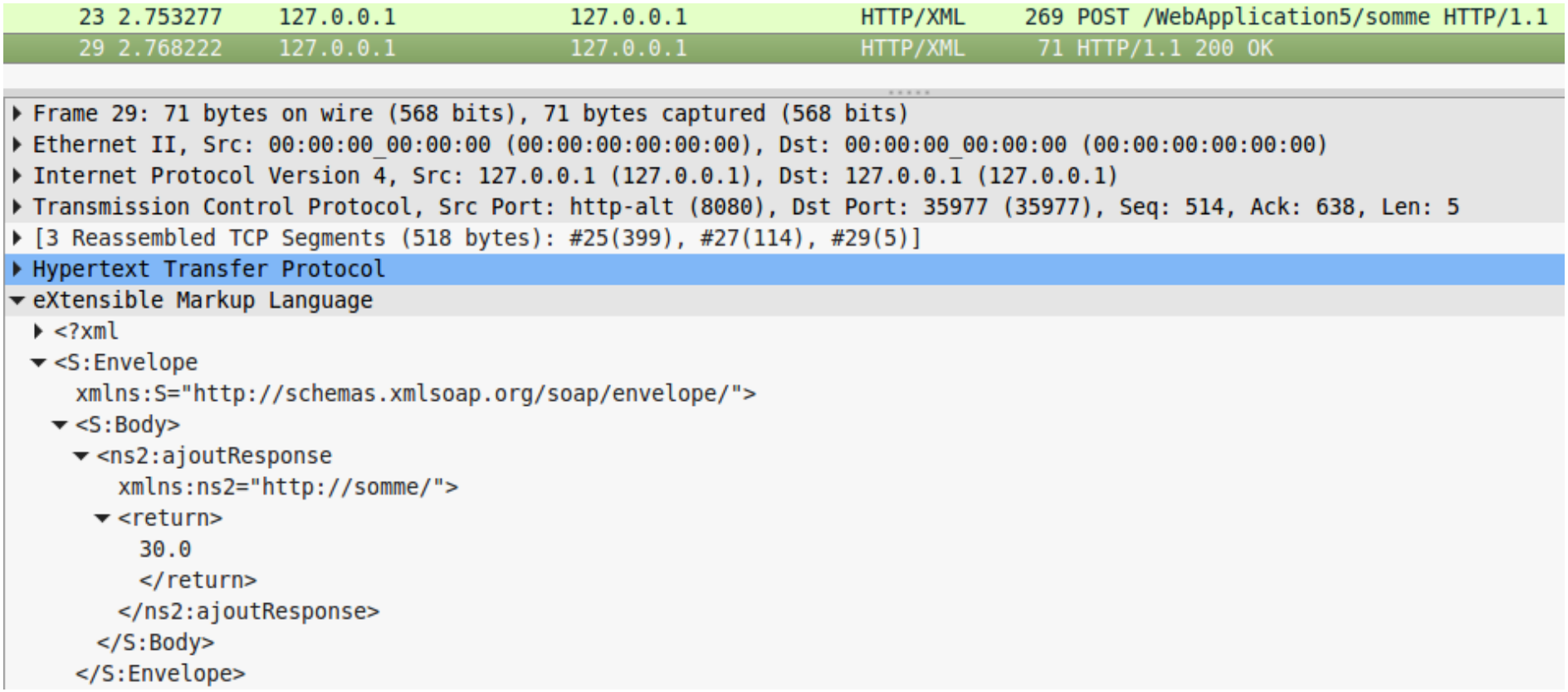
Structure d'un message SOAP



SOAP - HTTP


23 4.152409 127.0.0.1 127.0.0.1 HTTP/XML 238 POST /WebApplication5/somme HTTP/1.1
29 4.166975 127.0.0.1 127.0.0.1 HTTP/XML 71 HTTP/1.1 200 OK
Frame 23: 238 bytes on wire (1904 bits), 238 bytes captured (1904 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
Transmission Control Protocol, Src Port: 35891 (35891), Dst Port: http-alt (8080), Seq: 437, Ack: 1, Len: 172
[2 Reassembled TCP Segments (608 bytes): #21(436), #23(172)]
Hypertext Transfer Protocol
eXtensible Markup Language
<?xml
<S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns2:tableau
xmlns:ns2="http://somme/"/>
</S:Body>
</S:Envelope>

SOAP - HTTP


23 2.753277 127.0.0.1 127.0.0.1 HTTP/XML 269 POST /WebApplication5/somme HTTP/1.1
29 2.768222 127.0.0.1 127.0.0.1 HTTP/XML 71 HTTP/1.1 200 OK
Frame 29: 71 bytes on wire (568 bits), 71 bytes captured (568 bits)
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
Transmission Control Protocol, Src Port: http-alt (8080), Dst Port: 35977 (35977), Seq: 514, Ack: 638, Len: 5
[3 Reassembled TCP Segments (518 bytes): #25(399), #27(114), #29(5)]
Hypertext Transfer Protocol
eXtensible Markup Language
<?xml
<S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns2:ajoutResponse
xmlns:ns2="http://somme/">
<return>
30.0
</return>
</ns2:ajoutResponse>
</S:Body>
</S:Envelope>

L'enveloppe

C'est l'élément supérieur du document : il englobe entête et corps. Il est obligatoire. Sans enveloppe, le message ne peut être transporté et doit répondre qualifié, c'est-à-dire répondre à l'espace de nom définissant SOAP.

L'entête (Header)

- Doit être placé au sein de l'enveloppe avant le corps.
- Elle peut-être utilisé pour compléter les informations nécessaires à une requête :
  - informations sur l'émetteur ;
  - protocole utilisé (par exemple ftp ou smtp) ;
  - passage par des systèmes intermédiaires.
- Les informations de l'entête peuvent être traitées, modifiées ou effacées par les applications intermédiaires.

Le corps (Body)

- Contient les données transportées par le message SOAP.

Le corps (Body)

- Contient les données transportées par le message SOAP.
- Il doit contenir dans :
  - la requête : le nom de la méthode appelée, ainsi que les paramètres appliqués à cette méthode.
  - la réponse : une réponse à sens unique ou un message d'erreur détaillée.

Le corps (Body)

- Contient les données transportées par le message SOAP.
- Il doit contenir dans :
  - la requête : le nom de la méthode appelée, ainsi que les paramètres appliqués à cette méthode.
  - la réponse : une réponse à sens unique ou un message d'erreur détaillée.
- Ce dernier message utilise le sous-élément **Fault**, qui lui-même dispose de quatre sous-éléments possibles :
  1. **faultcode :** identifiant l'erreur par un code.
  2. **faultstring :** une explication lisible de l'erreur.
  3. **faultactor :** désigne l'origine de l'erreur.
  4. **detail :** donne des détails spécifiques.

Message « Fault »

est utilisé pour acheminer des erreurs. S'il est présent, l'élément **Fault** doit apparaître comme une entrée de **body** et doit apparaître une seule fois.

Message « Fault » : exemple 1

```
<soapenv:Body>
  <soapenv:Fault>
   <faultcode>soapenv:Server.userException</
       faultcode>
   <faultstring>java.lang.RuntimeException: No
       compiler found in your classpath! (you
       may need to add 'tools.jar')</
       faultstring>
   <detail>
    <ns1:hostname xmlns:ns1="http://xml.apache
       .org/axis/">alkhalil</ns1:hostname>
   </detail>
  </soapenv:Fault>
 </soapenv:Body>
</soapenv:Envelope>
```

Message « Fault » : exemple 2

```
<soapenv:Envelope>
<soapenv:Body>
<soapenv:Fault>
<faultcode> soapenv:Server.userException </
    faultcode>
<faultstring>org.xml.sax.SAXParseException;
    lineNumber: 1; columnNumber: 102; Le type d
    'element "calcule" doit etre suivi des
    specifications d'attribut, ">" ou "/>".</
    faultstring>
<detail> <ns1:hostname>alkhalil</ns1:hostname>
     </detail>
</soapenv:Fault>
</soapenv:Body>
</soapenv:Envelope>
```