SOAP Specification

Neila BEN LAKHAL (PhD @ Tokyo Institute of Technology)
By Neila.benlakhal@enicarthage.rnu.tn

What is SOAP ?

SOAP Version 1.2 (SOAP) is a lightweight protocol intended for exchange of structured information(XML) in a decentralized, distributed environment.

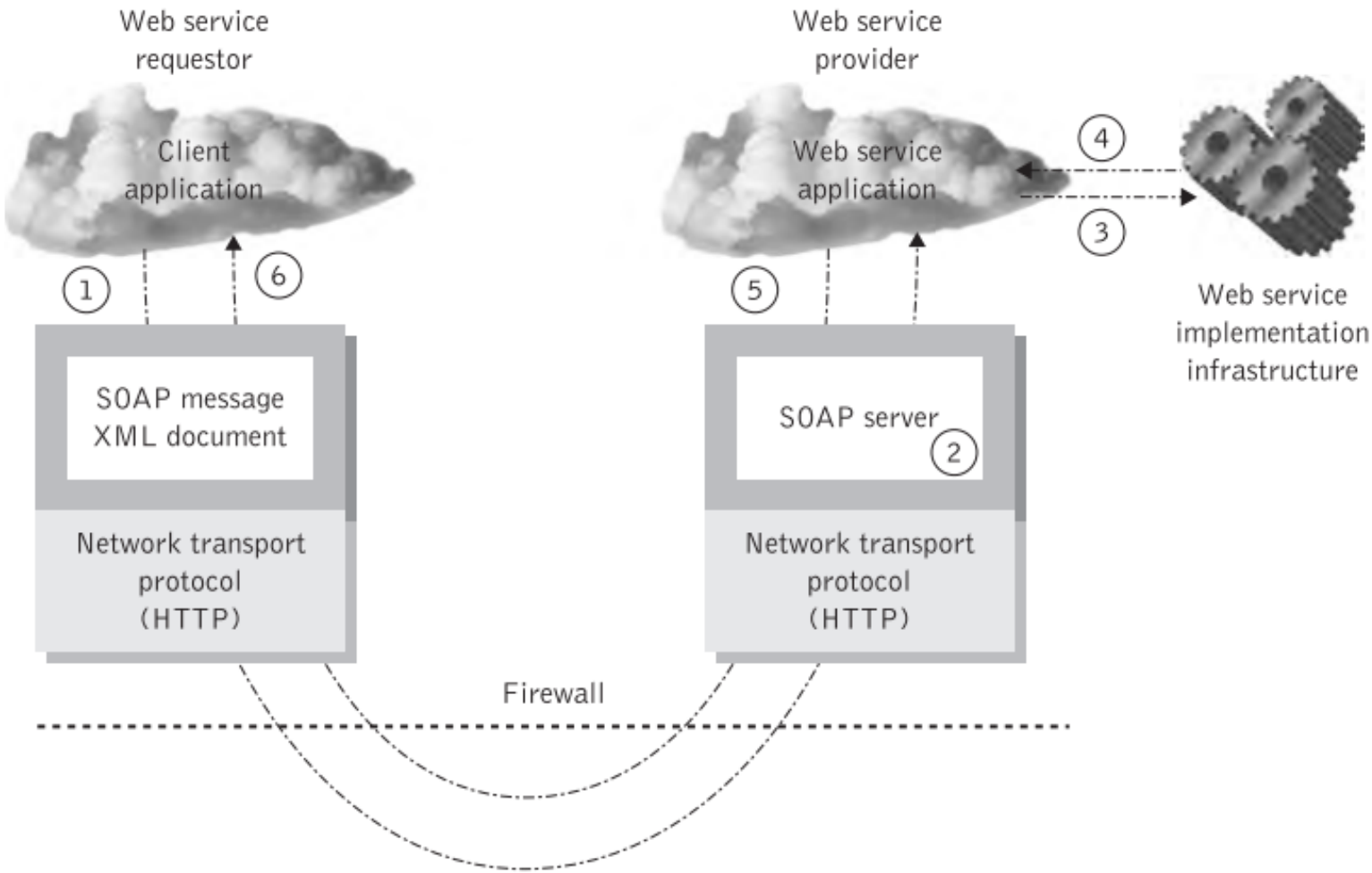
Why SOAP?

- In Web services, service requesters and service providers communicate with each other in the form of XML messages. If you would like your software to use a web service, it must send a request message and it should receive a response.

- These messages conform to SOAP : a standard developed initially by Microsoft in 1998 and originally standing for a Simple Object Access Protocol. That meaning has since been dropped, but it does give you an idea of its purpose; which is to allow a service requester to invoke services.

La naissance de SOAP

- L'idée en elle-même n'était pas nouvelle :
  - Fortement inspirée de COM Internet Services (CIS) qui supportaient l'acheminement des appels DCOM à travers le protocole TCP utilisant le port 80
  - Appel RPC d'une procédure distante placé sur une machine Windows.
  - Dans le temps c'était une idée ingénieuse qui a permit la survie de plusieurs projets freinée par le mise en place de firewalls.


Web service
requestor
Client
application
1
6
SOAP message
XML document
Network transport
protocol
(HTTP)
Web service
provider
Web service
application
4
3
5
SOAP server
2
Network transport
protocol
(HTTP)
Web service
implementation
infrastructure
Firewall

DCOM : prédécesseur de SOAP

- Du moment que DCOM le permettaient, pourquoi SOAP ?
  - La complexité du déploiement d'une application DCOM
  - La complexité de l'outil de configuration dcomcnfg.exe
  - Très difficile de faire en sorte que les appels DCOM contournent les firewalls.

(excerpt from :Building Web Applications with ADO.NET and XML Web Services)

What makes these new Web services interesting is SOAP. SOAP is just a protocol specification that defines how to invoke methods on servers, typically over HTTP. It provides an extensible way for applications to communicate using simple XML messages over the Web. Regardless of operating system, brand of parser, or language, the messages themselves are the standard. The SOAP specification goes on to mandate a specific XML vocabulary that defines parameters, return values, and exceptions. This means that two software developers with completely different backgrounds, platform preference, and love for software companies can coexist peacefully in the world — and work together efficiently. Clients written in Visual Basic .NET can easily invoke services running CORBA on Unix boxes, while Windows Scripting Host (WSH) clients can call mainframe code and access DB2. Even your Macintosh boxes can access Perl Web services running on Red Hat Linux. The list goes on and on.

SOAP : Objectifs visés

- Assurer la communication entre applications d'une même entreprise (intranet)
- Assurer les échanges interentreprises entre applications et services (Web)
- Il permet d'appeler une méthode et d'envoyer des messages aux machines distantes via HTTP le plus souvent, mais peut également se faire par un autre protocole, comme SMTP…

SOAP :Qui est derrière?

- Initiative conjointe de Microsoft et IBM,
- **SOAP 1.1**, W3C note soumise en mai 2000,
- **SOAP 1.2**, recommandation du W3C, juin 2003,
- Initialement SOAP désignait l'acronyme de Simple Object Access Protocol
- SOAP n'est plus un acronyme depuis la version 1.2. La notion d'objet (spécifiée dans **Simple Object Access Protocol**) devient donc obsolète.
- Les deux plus fortes fonctionnalités de SOAP sont sa simplicité et le fait que tout le monde a accepté de l'utiliser.

Rôles de SOAP

- SOAP permet à des méthodes (d'un code source) de toute nature– sur n'importe quelle plateforme, dans n'importe quel langage – de communiquer.

- À l'heure actuelle, la spécification SOAP a été mis en œuvre dans plus de 60 langages sur plus de 20 plateformes.

La philosophie SOAP

◉SOAP codifie simplement une pratique existante

- Utilisation conjointe de XML et HTTP

◉SOAP est un protocole **minimal** pour appeler des méthodes des serveurs, services, composants, objets

- Ne pas imposer une API ou un runtime
- Ne pas imposer l'utilisation d'un ORB (CORBA, DCOM, …) ou d'un serveur web particulier (Apache, IIS, …)
- Ne pas imposer un modèle de programmation
  - Plusieurs modèles peuvent être utilisés conjointement
- Et "ne pas réinventer une nouvelle technologie"

Fonctionnement de SOAP

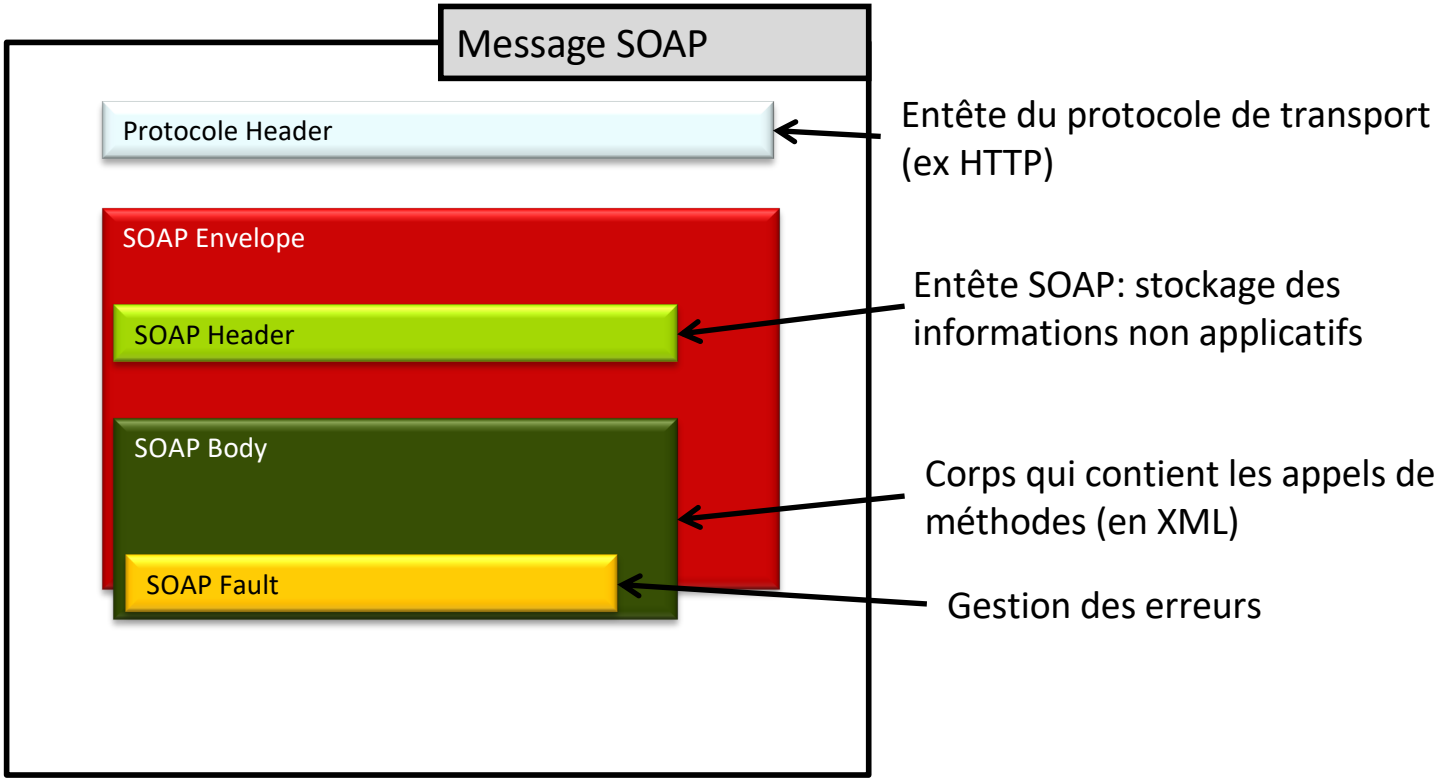
◉Un message SOAP est envoyé d'un expéditeur (client) à un destinataire (serveur), directement ou via des intermédiaires.

- Fonctionnement côté Client (simplifié)
  - Ouverture d'une connexion HTTP et envoie d'une requête
  - Requête SOAP est un document XML décrivant :
  - une méthode à invoquer sur une machine distante
  - Les paramètres de la méthode
- Fonctionnement côté Serveur (simplifié)
  - Récupère la requête
  - Exécute la méthode avec les paramètres
  - Renvoie une réponse SOAP (document XML) au client

Structure d'un message SOAP 1/2

- Un message SOAP est un document XML constitué d'une enveloppe composée de deux parties :
  - Enveloppe **/ <envelope>** (Élément racine obligatoire), il contient:
    - Entête / **<header>** (Élément optionnel)
    - Contient des entrées non applicatives (Ex: Transactions, sessions, …)
    - Peuvent être adressées à certains intermédiaires
    - Leur traitement peut être marqués comme obligatoire
    - Corps / **<body>** (Élément obligatoire)
    - Contient les entrées applicatives du message :Nom d'une méthode, valeurs des paramètres, valeur de retour.
- Comme les messages SOAP ne peuvent pas être envoyés en lots, l'enveloppe contient un seul message,
- La requête et la réponse ont la même structure.

Structure d'un message SOAP 2/2



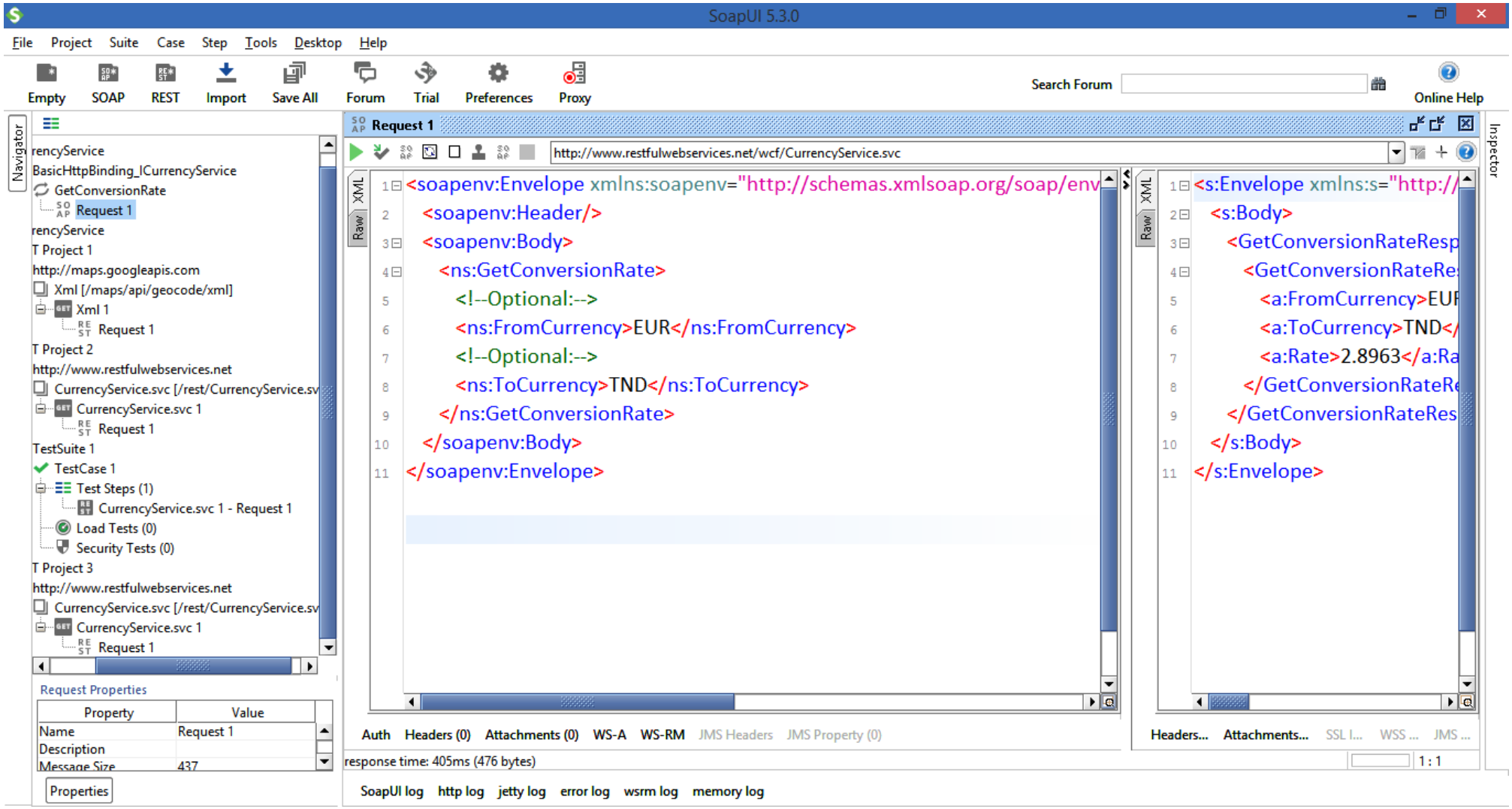
SOAP par l'exemple : Requête vers un service

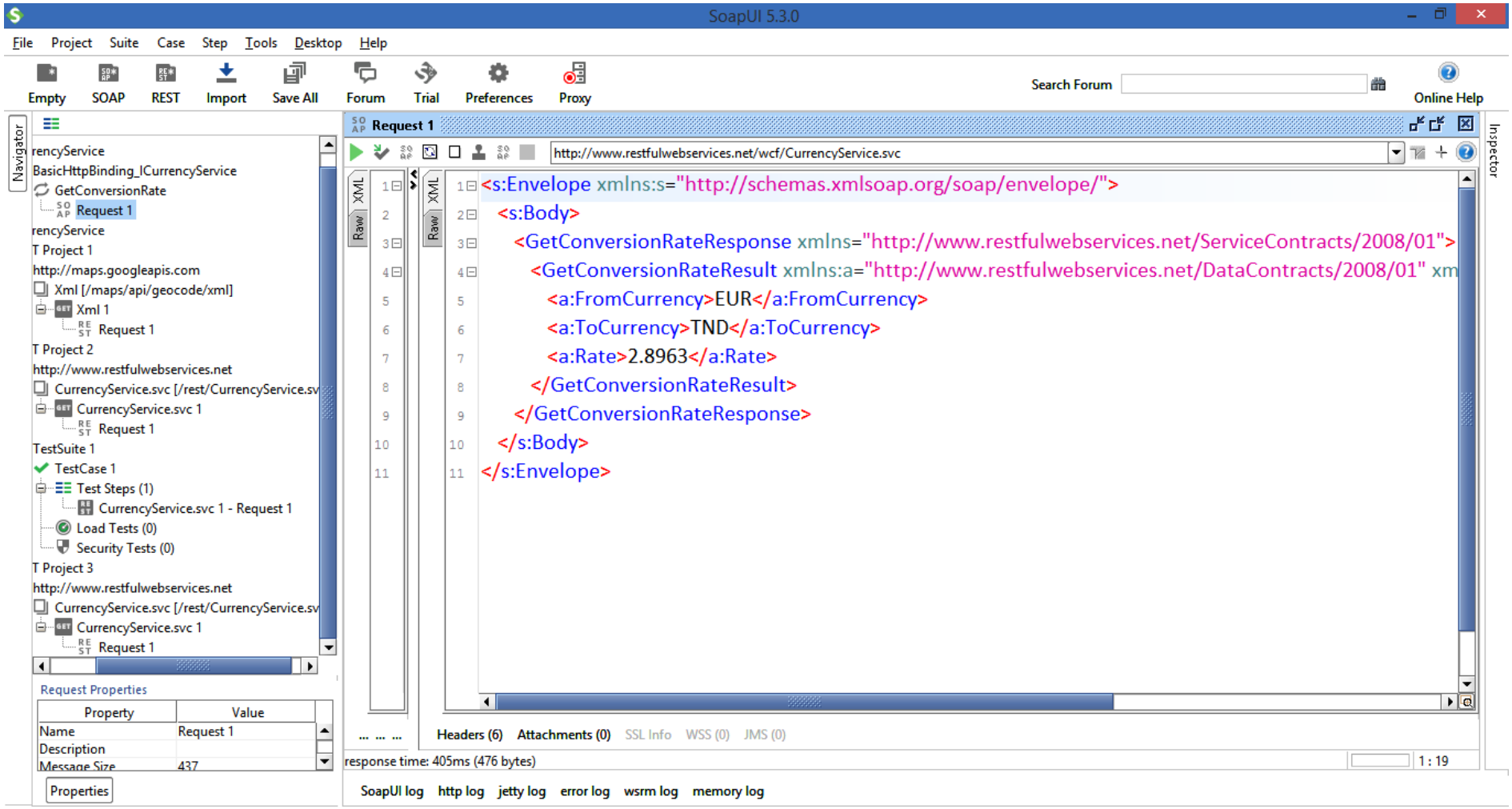
L'appel SOAP du service et de sa méthode GetCountryByCurrencyCode avec le paramètre CurrencyCode=TND:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body>
<GetCountryByCurrencyCode xmlns="http://www.webserviceX.NET">
<CurrencyCode>TND</CurrencyCode> </GetCountryByCurrencyCode> </soap:Body>
</soap:Envelope>
```

La réponse SOAP du service

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Body>
<GetCountryByCurrencyCodeResponse xmlns="http://www.webserviceX.NET">
<GetCountryByCurrencyCodeResult>Tunisia</GetCountryByCurrencyCodeResult>
</GetCountryByCurrencyCodeResponse> </soap:Body>
 </soap:Envelope>
```


SoapUI 5.3.0
File Project Suite Case Step Tools Desktop Help
Empty SOAP REST Import Save All Forum Trial Preferences Proxy
Search Forum
Online Help
Navigator
rencyService
BasicHttpBinding_ICurrencyService
GetConversionRate
Request 1
rencyService
T Project 1
http://maps.googleapis.com
Xml [/maps/api/geocode/xml]
Xml 1
Request 1
T Project 2
http://www.restfulwebservices.net
CurrencyService.svc [/rest/CurrencyService.sv
CurrencyService.svc 1
Request 1
TestSuite 1
TestCase 1
Test Steps (1)
CurrencyService.svc 1 - Request 1
Load Tests (0)
Security Tests (0)
T Project 3
http://www.restfulwebservices.net
CurrencyService.svc [/rest/CurrencyService.sv
CurrencyService.svc 1
Request 1
Request Properties
Property
Value
Name
Request 1
Description
Message Size
437
Properties
Request 1
http://www.restfulwebservices.net/wcf/CurrencyService.svc
XML
Raw
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/env
<soapenv:Header/>
<soapenv:Body>
<ns:GetConversionRate>
<!--Optional:-->
<ns:FromCurrency>EUR</ns:FromCurrency>
<!--Optional:-->
<ns:ToCurrency>TND</ns:ToCurrency>
</ns:GetConversionRate>
</soapenv:Body>
</soapenv:Envelope>
<s:Envelope xmlns:s="http://
<s:Body>
<GetConversionRateResp
<GetConversionRateRes
<a:FromCurrency>EUR
<a:ToCurrency>TND</
<a:Rate>2.8963</a:Ra
</GetConversionRateR
</GetConversionRateRes
</s:Body>
</s:Envelope>
Auth Headers (0) Attachments (0) WS-A WS-RM JMS Headers JMS Property (0)
Headers... Attachments... SSL I... WSS ... JMS ...
response time: 405ms (476 bytes)
1 : 1
SoapUI log http log jetty log error log wsrm log memory log
Inspector


SoapUI 5.3.0
File Project Suite Case Step Tools Desktop Help
Empty SOAP REST Import Save All Forum Trial Preferences Proxy
Search Forum
Online Help
Navigator
Inspector
Request 1
http://www.restfulwebservices.net/wcf/CurrencyService.svc
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
<s:Body>
<GetConversionRateResponse xmlns="http://www.restfulwebservices.net/ServiceContracts/2008/01">
<GetConversionRateResult xmlns:a="http://www.restfulwebservices.net/DataContracts/2008/01" xm
<a:FromCurrency>EUR</a:FromCurrency>
<a:ToCurrency>TND</a:ToCurrency>
<a:Rate>2.8963</a:Rate>
</GetConversionRateResult>
</GetConversionRateResponse>
</s:Body>
</s:Envelope>
rencyService
BasicHttpBinding_ICurrencyService
GetConversionRate
Request 1
rencyService
T Project 1
http://maps.googleapis.com
Xml [/maps/api/geocode/xml]
Xml 1
Request 1
T Project 2
http://www.restfulwebservices.net
CurrencyService.svc [/rest/CurrencyService.sv
CurrencyService.svc 1
Request 1
TestSuite 1
TestCase 1
Test Steps (1)
CurrencyService.svc 1 - Request 1
Load Tests (0)
Security Tests (0)
T Project 3
http://www.restfulwebservices.net
CurrencyService.svc [/rest/CurrencyService.sv
CurrencyService.svc 1
Request 1
Request Properties
Property
Value
Name
Request 1
Description
Message Size
437
Properties
Headers (6) Attachments (0) SSL Info WSS (0) JMS (0)
response time: 405ms (476 bytes)
1 : 19
SoapUI log http log jetty log error log wsrm log memory log

SOAP envelope

Règles de syntaxe - Un message SOAP :

- DOIT être codé en utilisant le langage XML
- La spécification impose que les balises, les sous balises ainsi que tous les attributs contenus dans l'enveloppe SOAP doivent **explicitement** être associés à un *namespace*, de manière à supprimer toute ambiguïté.
- Ces namespaces doivent être définis correctement.
- NE DOIT PAS contenir ni une référence DTD ni des instructions de traitement XML.
- Toutes les balises XML associées à SOAP doivent avoir :
  - Le préfixe soap: ou soap-env: pour SOAP 1.1.
  - Le préfixe env: pour SOAP 1.2.

SOAP envelope

SOAP envelope (enveloppe) :

- C'est l'élément de racine du message SOAP.
- L'enveloppe SOAP sert de conteneur aux autres éléments du message SOAP, elle est définie au début par la balise <soap:Envelope> et se termine par la balise </soap:Envelope> .
- Le prologue XML contient seulement une déclaration XML <?xml version="1.0" encoding="UTF-8" ?> spécifiant la version de XML et l'encodage des caractères du message XML.

Syntaxe:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
...
</soap:Envelope>
```

SOAP envelope

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
...
</soap:Envelope>
```

- Par convention, la spécification SOAP définit plusieurs *namespaces* :
  - soap-env: ou soap: comme préfixe associé à l'URI http://schemas.xmlsoap.org/soap/envelope/ pour définir le *namespace* de l'enveloppe dans la version 1.1.
  - env: comme préfixe associé à l'URI http://www.w3.org/2003/05/soap-envelope/ pour définir le *namespace* de l'enveloppe dans la version 1.2.
  - L'attribut encodingStyle associé à l'URI http://www.w3.org/2001/12/soap-encoding pour la définition des formats de types de données.

SOAP header (L'en-tête)

SOAP header (L'en-tête) :

- L'en-tête est un élément facultatif dans un message SOAP.
- Si présent, il doit être le premier élément dans l'enveloppe SOAP.
- Il sert à communiquer des informations pour :
  - Authentifier l'émetteur,
  - Définir des règles de sécurité et des algorithmes de chiffrement à utiliser pour la lecture du message,
  - Donner le contexte d'une transaction,
  - Etc.
- La spécification SOAP n'indique pas comment ajouter ce type d'information dans l'entête.

SOAP Header

- A noter que l'authentification et la gestion de session sont en dehors du cadre du protocole SOAP, même si les concepteurs de SOAP autorisent une certaine flexibilité dans la transmission de messages SOAP, de telle façon que les personnes qui les implémentent puissent inclure de telles informations.

L'entête SOAP

◉L'en-tête d'un message SOAP commence avec la balise <soap:Header> et se termine avec la balise </soap:Header>,

- Note : on peut aussi trouver: <soap-env:Header></soap-env:Header>.

◉2 attributs associés à l'en-tête SOAP :

- soap:mustUnderstand : cet attribut prend la valeur 1 ou 0:
  - La valeur 1 : le récepteur doit reconnaître l'information présente dans l'en-tête, son traitement est obligatoire.
  - La valeur 0 : l'en-tête peut être ignoré par le récepteur.
- soap:actor : cet attribut identifie le destinataire de l'élément fils. Comme un message SOAP peut traverser plusieurs composants avant d'atteindre le destinataire final du message, il peut être nécessaire d'adresser des informations aux composants intermédiaires qui composent le chemin vers le destinataire final. C'est par l'intermédiaire de cet attribut que cela est rendu possible, en lui donnant comme valeur une URL qui correspond à ou aux destinataires de l'élément.
  Note : En l'absence de cet attribut, l'élément fils est à destination du destinataire final du message SOAP.

SOAP Header

- L'attribut mustUnderstand

  La valeur 1 : le récepteur doit reconnaître l'information présente dans l'en-tête, son traitement est obligatoire.

- Exemple :

```
<soap-env:header>
    <t:transaction xmlns:t="some-uri"
    soap-env:mustunderstand="1">
    5
    </t:transaction>
</soap-env:header>
```

SOAP body

◉Le corps SOAP <soap:Body> est un élément obligatoire dans le message SOAP.

- Il contient l'information destinée au receveur.

◉Le contenu du corps SOAP est utilisé pour spécifier un appel de méthode à un ordinateur distant avec les valeurs de paramètres.

- Par exemple, la demande du solde d'un compte bancaire.

◉Le corps du message SOAP commence avec la balise <soap:Body> et se termine avec la balise </soap:Body>.

Body: Syntaxe

- La partie Body encapsule un unique tag de méthode qui porte le nom de la méthode elle-même:
  - <ns1:maMethode ... > ou <ns1:maMethodeReponse ... > (le même nom suivi du mot « Response » dans le cas du message de réponse).
- Le tag de la méthode reçoit typiquement le namespace correspondant au nom du WS, pour assurer l'unicité.
- Le tag de méthode encapsule à son tour un certain nombre de paramètres, tel que le tag <param1 ... > dans l'enveloppe d'une requête. En présence de plusieurs paramètres, leur ordre d'apparition n'a pas d'importance.
- Dans le message de réponse, on ne trouve qu'un seul tag de paramètre représentant la valeur de retour de la méthode (généralement appelé <return>).

SOAP body (requête) exemple

Exemple : L'extrait suivant représente un corps SOAP qui fait appel à une méthode appelée GetPrice():

- L'élément <GetPrice> fournit le nom de la méthode à appeler: **GetPrice().**
- L'élément <Item> est un paramètre qui est passé dans la méthode **GetPrice().**

```
<?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>
  <m:GetPrice xmlns:m="http://www.w3schools.com/prices">
    <m:Item>Apples</m:Item>
  </m:GetPrice>
</soap:Body>
</soap:Envelope>
```

SOAP body (réponse) : exemple

◉Exemple : L'extrait suivant représente un corps SOAP qui est la réponse à l'appel à la méthode appelée GetPrice():

- L'élément <GetPriceResponse> fournit le nom de la méthode qui a été appelée: **GetPrice().**
- L'élément <Price> contient la valeur de retour de **GetPrice().**

```
<?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>
 <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
  <m:Price>1.90</m:Price>
 </m:GetPriceResponse>
</soap:Body></soap:Envelope>
```

Types de données /encoding

- Les messages SOAP peuvent supporter des encodages de données complexe et c'est l'une des forces de ce protocole:
- Exemple : Le dialogue résultant de l'appel de **getEmployeeDetails** : Le client envoie **un entier (l'identification de l'employé)** et reçoit un tableau de chaînes de caractères à 2 éléments contenant : le nom de l'employé et le numéro de téléphone

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope    …xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/1999/XMLSchema">
            <SOAP-ENV:Body>
                  <ns1:getEmployeeDetails xmlns:ns1="urn:MySoapServices">
                        <param1 xsi:type="xsd:int">1016577</param1>
                  </ns1:getEmployeeDetails>
            </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Types de données /encoding

- Le client reçoit un tableau de chaînes de caractères à deux éléments contenant : le nom de l'employé et le numéro de téléphone
- Dans la balise **<return>** du message de réponse, nous avons le type de la structure complexe (le tableau de chaines) :

```
<SOAP-ENV:Envelope …
        <SOAP-ENV:Body><ns1:getEmployeeDetailsResponse   xmlns:ns1="urn:MySoapServices"
                    SOAP-NV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
                     <return xmlns:ns2="http://schemas.xmlsoap.org/soap/encoding/"
                                        xsi:type="ns2:Array"
                                        ns2:arrayType="xsd:string[2]">
                                        <item xsi:type="xsd:string">Bill Posters</item>
                                        <item xsi:type="xsd:string">+1-212-7370194</item>
                                </return>
                                </ns1:getEmployeeDetailsResponse>
        </SOAP-ENV:Body> </SOAP-ENV:Envelope>
```

SOAP <Fault>

L'élément optionnel SOAP <Fault> est utilisé pour indiquer les messages d'erreur.

- Si un élément <Fault> est présent, il doit apparaître comme un élément enfant de <Body> .
- Un élément <Fault> ne peut apparaître qu'une fois dans un message SOAP.

```
<soap:Envelope …

<soap:Header> … </soap:Header>
<soap:Body> …
   <soap:Fault> … </soap:Fault>
</soap:Body>
</soap:Envelope>
```

SOAP <Fault>

- L'élément <Fault> comporte les éléments suivants comme fils :

élément	Description
<faultcode>	Contient un code qui identifie l'erreur. SOAP définit 4 codes d'erreur: •VersionMismatch (namespace du bloc Envelope non valide), •MustUnderstand (un élément fils du bloc Header qui devait absolument être compris ne l'a pas été), •Client (message mal formatée ou non valide), •Server (problème lors du traitement du message).
<faultstring>	Contient un texte décrivant brièvement l'erreur pour un humain.
<faultactor>	Contient le composant (i.e., son URL) qui a généré l'erreur.
<detail>	Contient des informations spécifiques à l'application et concernant l'erreur.

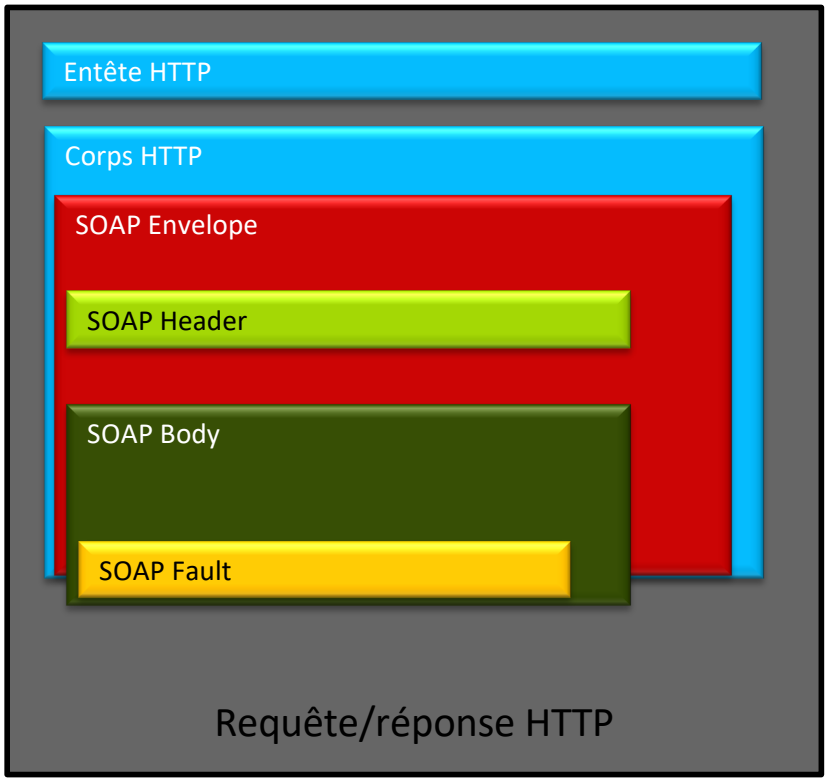
Exemple 1 d'un bloc <fault>

- Ici, l'erreur provient du composant (messageDispatcher) qui n'a pas été en mesure de traiter le message (i.e., de le diriger vers le destinataire final).

```
<soap:Envelope …<soap:Header>…</soap:Header>
<soap:Body>
 <soap:Fault>
 <faultcode>soap:Server</faultcode>
 <faultstring>Impossible de router le message.</faultstring>
 <faultactor>http://www.exemple.com/messageDispatcher</faultactor>
 <detail>
 <m:error xmlns:m="http://www.exemple.com/errors"> E_NO_ROUTE
 </m:error>
 </detail>
 </soap:Fault>
 </soap:Body>
</soap:Envelope>
```

SOAP transporté par HTTP

- SOAP utilise un protocole de transport pour véhiculer les messages SOAP de l'émetteur au récepteur
  HTTP, SMTP, FTP, …
- Le modèle requête/réponse de SOAP convient parfaitement au modèle requête/réponse HTTP.



Exemple Requête HTTP/SOAP

L'appel SOAP du service Country et de sa méthode GetCurrencyByCountry avec le paramètre CountryName=Tunisia:

```
POST /country.asmx HTTP/1.1 Host: www.webservicex.net
Content-Type: text/xml; charset=utf-8
Content-Length: 539
SOAPAction: "http://www.webserviceX.NET/GetCurrencyByCountry"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xmlns:xsd=http://www.w3.org/2001/XMLSchema
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
 <GetCurrencyByCountry xmlns="http://www.webserviceX.NET">
                <CountryName>Tunisia</CountryName>
 </GetCurrencyByCountry>
 </soap:Body>
 </soap:Envelope>
```

Exemple Requête HTTP/SOAP

- Méthode de type **POST** (Envoi de données au service situé à l'URL spécifiée)
- **Content-Type** : renseigne sur le type MIME de l'entité transportée, doit être obligatoirement text/xml, pour un message SOAP. À cette valeur suit l'attribut **charset** (encoding).
- **SOAPAction** : SOAP introduit dans l'en-tête HTTP POST le nouveau champ de requête SOAPAction, pour indiquer la destination du message SOAP.
  - La spécification SOAP 1.1 précise que ce champ est obligatoire dans une requête HTTP véhiculant un message SOAP.
  - soit contient un URL(qui peut être aussi une chaîne de caractères vide) ;
  - soit ne contient aucune valeur (mais le champ est présent quand même).

```
POST /country.asmx HTTP/1.1 Host: www.webservicex.net
Content-Type: text/xml; charset=utf-8
Content-Length: 539
SOAPAction: "http://www.webserviceX.NET/GetCurrencyByCountry"
```

Exemple Réponse HTTP/SOAP

La réponse SOAP du service

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: 465
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
 <GetCurrencyByCountryResponse xmlns="http://www.webserviceX.NET">
 <GetCurrencyByCountryResult>TND</GetCurrencyByCountryResult>
 </GetCurrencyByCountryResponse>
 </soap:Body>
 </soap:Envelope>
```

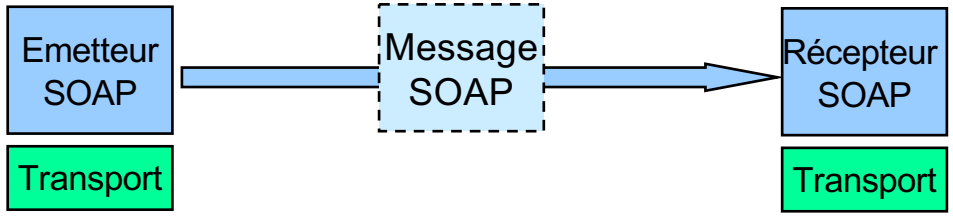
Exemple Réponse HTTP/SOAP

- La réponse HTTP véhicule les codes de retour HTTP. Dans la liaison SOAP/HTTP, les codes 2xx indiquent que le message SOAP a été reçu.
- Si code 500, message en erreur, le corps SOAP doit contenir l'élément <fault>.

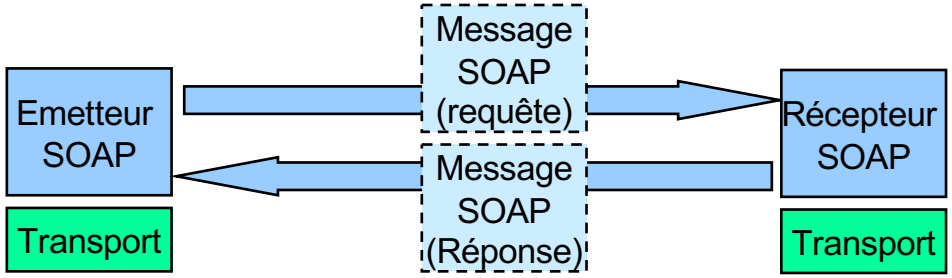
```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: 465
```

Echange de messages SOAP

- Modèle d'échange de base : Message à sens unique (one-way)



- Modèle d'échange implicite : Requête/Réponse (Appels RPC)



Les Styles de message

Style RPC

- Appels de procédures distants, Degreé élevé d'automatisation
- Paramètres proches des types des langages de programmation
- <soap:Body> contient un élément avec le nom de la méthode ou de procédure distante invoquée. Cet élément contient à son tour un élément pour chaque paramètre de cette procédure.

Style DOC (Document)

- Echanges de messages conformes à des schémas arbitraries.
- Plus d'expressivité , Encouragé par .Net
- < soap:Body> contient un ou plusieurs éléments enfant appelé parties. Pas de règles de formatage SOAP pour ce que le <soap:Body> contient, l'expéditeur et le destinataire se mettent d'accord sur le contenu.

Les Styles de message: asynchrone ou synchrone

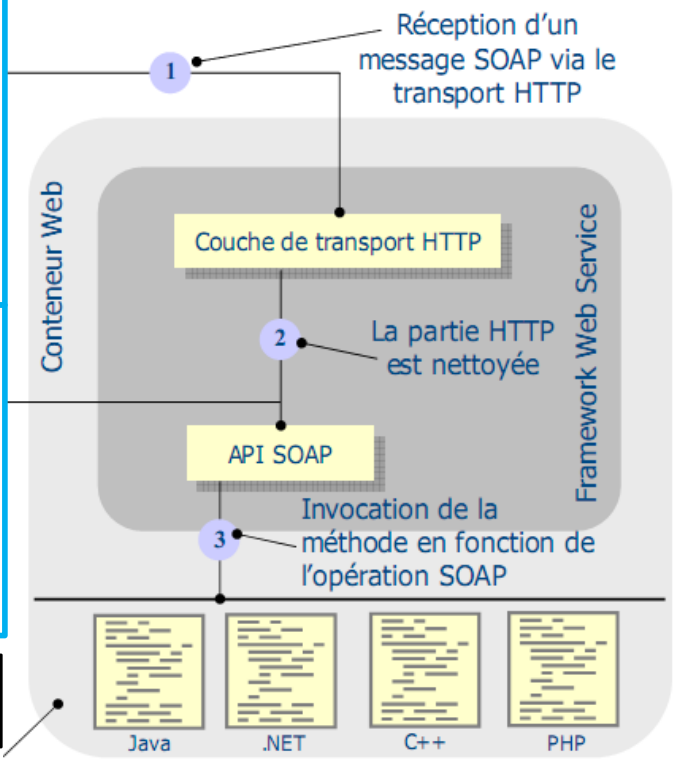
- Aujourd'hui : beaucoup de services synchrones,
  - au dessus d'HTTP.
  - Appels bloquants
  - Pas de garanties (timeout ?)

- Plus robuste : Échanges asynchrones
  - SMTP, JMS, …
  - Non-bloquants
  - Garanties de service (Exactly Once)

Traitement d'un message SOAP

```
POST /country.asmx HTTP/1.1 Host: www.webservicex.net
Content-Type: text/xml; charset=utf-8
Content-Length: 539
SOAPAction: "http://www.webserviceX.NET/GetCurrencyByCountry"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xmlns:xsd=http://www.w3.org/2001/XMLSchema
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
 <GetCurrencyByCountry xmlns="http://www.webserviceX.NET">
                        <CountryName>Tunisia</CountryName>
 </GetCurrencyByCountry>
 </soap:Body>
 </soap:Envelope>
```

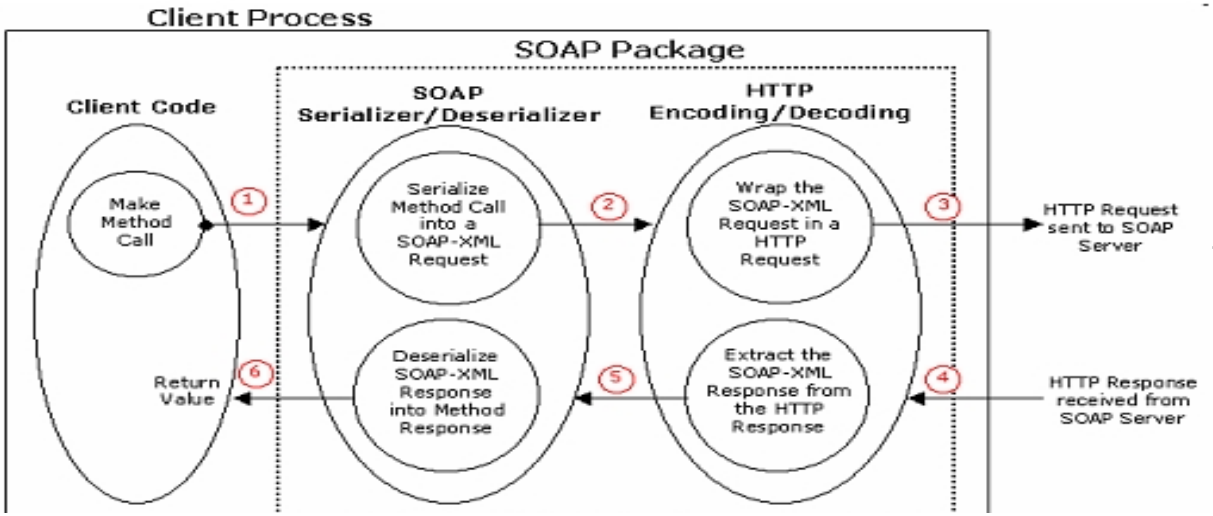
```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xmlns:xsd=http://www.w3.org/2001/XMLSchema
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
 <GetCurrencyByCountry xmlns="http://www.webserviceX.NET">
                        <CountryName>Tunisia</CountryName>
 </GetCurrencyByCountry>
 </soap:Body>
 </soap:Envelope>
```



Couche d'accès aux implémentations des services Web (différents langages)

Architecture technique côté client

- Les messages envoyés au serveur seront des requêtes SOAP-XML enveloppées dans des requêtes HTTP.
- De même, les réponses du serveur sont des réponses HTTP qui renferment des réponses SOAP-XML.
- Du côté client, pour ne pas prendre en charge la sérialisation SOAP et l'encodage HTTP, nous utilisons un package SOAP spécifique.
- Nous invoquons ensuite le service, simplement en invoquant la méthode appropriée du package SOAP (typiquement en spécifiant l'URL du service, le nom du service et tous les paramètres requis).
- **Le premier travail d'un package est de sérialiser l'invocation de ce service en requête SOAP. Il doit ensuite encoder ce message dans une requête HTTP et l'envoyer à l'URL spécifiée.**



- Le code client crée un appel de service en invoquant la méthode appropriée du package SOAP (1).
- Le serialiseur SOAP du package SOAP convertit cette invocation en requête SOAP et l'envoie à l'encodeur HTTP (2).
- L'encodeur HTTP enveloppe le message SOAP dans une requête HTTP et l'envoie au serveur SOAP (3).
- La réponse est reçue par le module d'encodage/décodage HTTP du serveur SOAP(4);
- ce module décode et extrait la réponse SOAP qui la remet au deserialiseur SOAP (5).
- Le deserialiseur SOAP desérialise le message et passe le résultat au code client (6) comme valeur de retour de l'invocation originale (1).

Architecture technique côté serveur

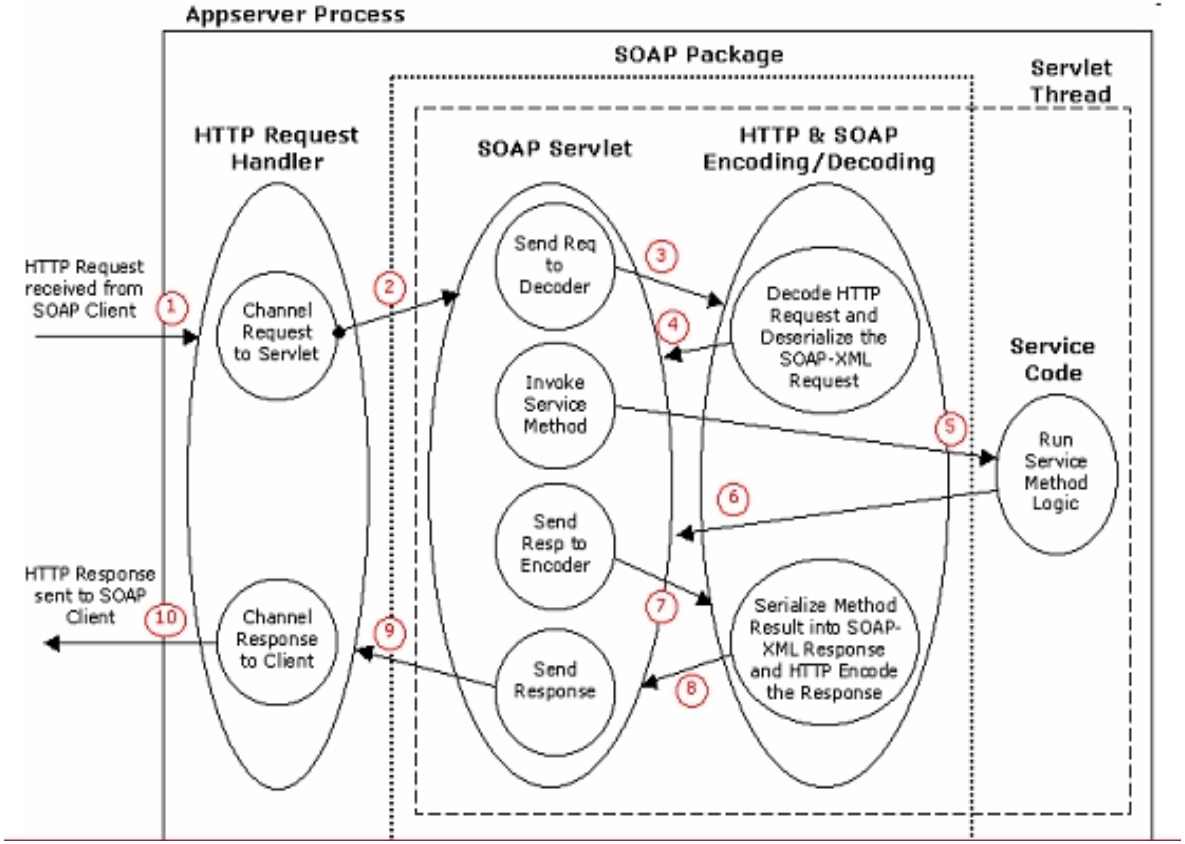
- Il est important de se rappeler que le choix du langage, de plate-forme et de package SOAP pour consommer des services web du côté client est **entièrement indépendant** du langage, de la plate-forme et du package SOAP utilisés par le côté serveur pour fournir des services web.

- De cette façon, le même service web basé sur SOAP (déployé par exemple sur UNIX , écrit en Java et exploitant Apache SOAP pour Java) peut être consommé par tout type de client écrit pour n'importe quelle plate-forme, dans n'importe quel langage, exploitant n'importe quel package SOAP applicable à cette combinaison langage/plate-forme. C'est l'une des grandes forces de la technologie SOAP.

Architecture technique côté serveur

- Nous avons besoin d'un process "**listener**" (Le Listener est le process serveur qui est en attente de connexion client).
- Nous avons également besoin d'une implémentation du service lui-même. A part cela, nous nous reposons sur un package SOAP de la même façon que du côté client.
- Le listener est souvent implémenté au travers d'une servlet qui s'exécute comme une application web sur un appserver (comme c'est le cas lorsque nous utilisons Apache SOAP du côté serveur).
- Le serveur sera configuré pour passer toutes les requêtes destinées à une certaine URL (l'URL du service SOAP) à une servlet particulier (appelons-la servlet SOAP).
- Le travail de la servlet SOAP est d'extraire le message XML-SOAP de la requête HTTP, de le désérialiser (de ce fait, séparer le nom de la méthode et les paramètres fournis), et d'invoquer la méthode du service en conséquence. Le résultat de la méthode est alors sérialisé, encodé HTTP et renvoyé au demandeur.

- Le serveur d'application web reçoit une requête HTTP du Client SOAP destinée à l'URL de service SOAP(1) et, en conséquence de quoi, le passe à la servlet SOAP (2). La servlet SOAP utilise les fonctionnalités de décodage SOAP et HTTP incluses dans le package pour extraire les détails de l'appel des services (3 et 4), c'est-à-dire le nom et les paramètres de la méthode. U ne fois muni de ceux-ci, la servlet SOAP peut invoquer la méthode (5 et 6), encoder la réponse (7 et 8) et fournir la réponse HTTP au handler de requêtes HTTP (9), qui à son tour, répond au client SOAP (10)



Implémentations de SOAP

◉List of Web services Framework (SOAP) :

- Apache AXIS, AXIS2, CXF (JAVA)
- NUSOAP, PHP5 SOAP API
- Gsoap(C/C++)
- .NET WCF (C#/VB.Net)
- JAX-WS (Java)
- Ksoap 2 (android)
- Zeep, PySimpleSOAP(PYTHON)
- Etc.

Service Testing : SOAPUI

- https://www.soapui.org/soap-and-wsdl/getting-started.html
- https://www.soapui.org/rest-testing/getting-started.html