

Web Services WS-SOAP, Formats d'échange et langages

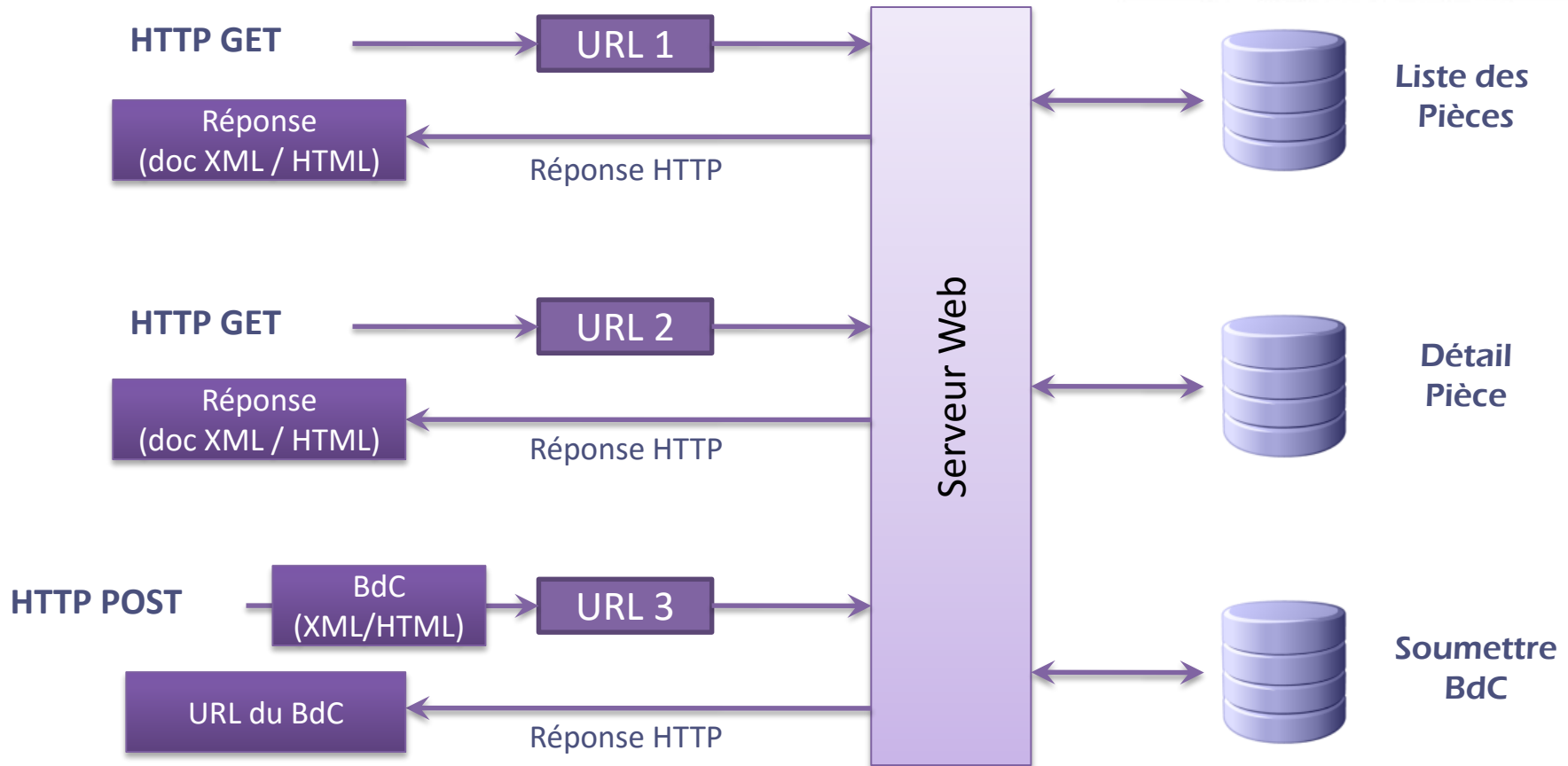
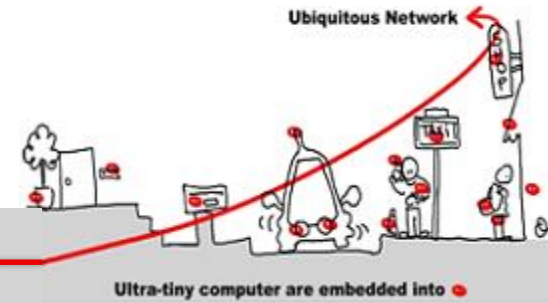
Jean-Yves Tigli

<http://www.tigli.fr>

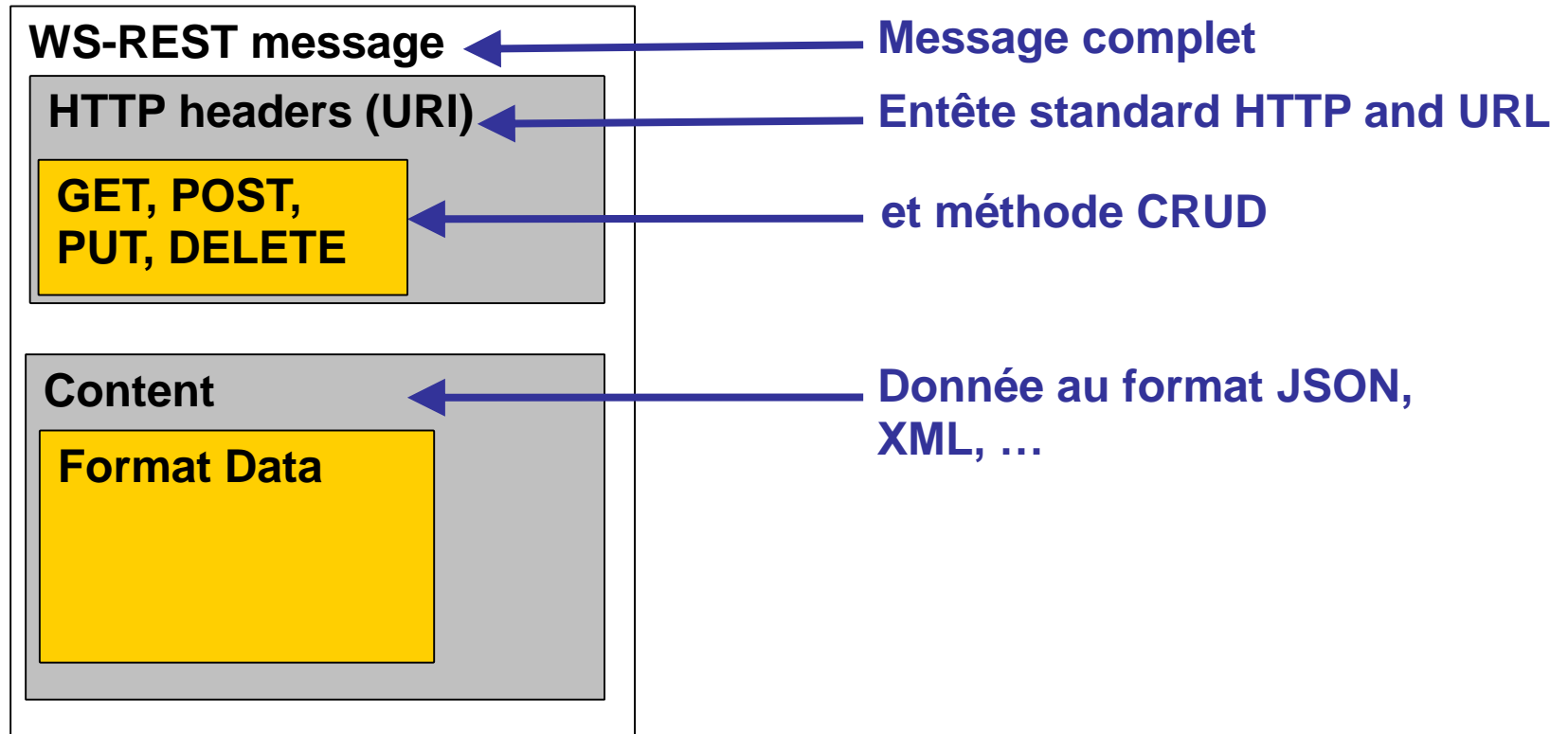
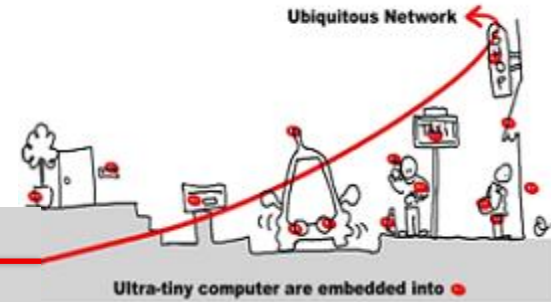
Polytech of Nice - Sophia Antipolis University

[Email : tigli@polytech.unice.fr](mailto:tigli@polytech.unice.fr)

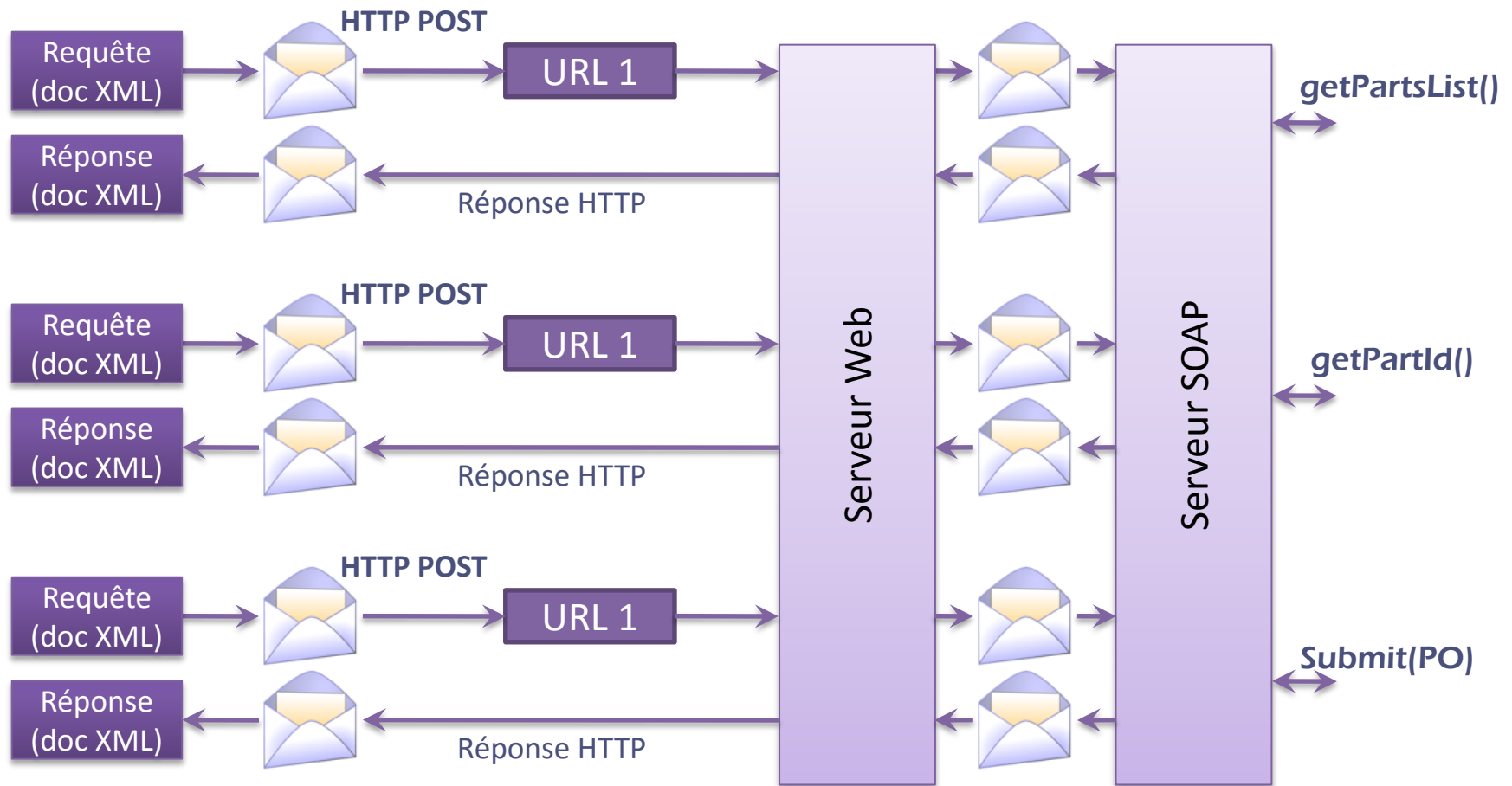
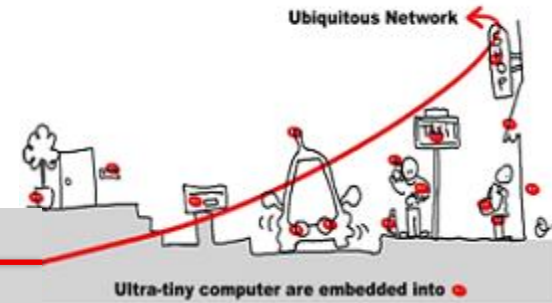
Rappel : REST pour une approche ROA



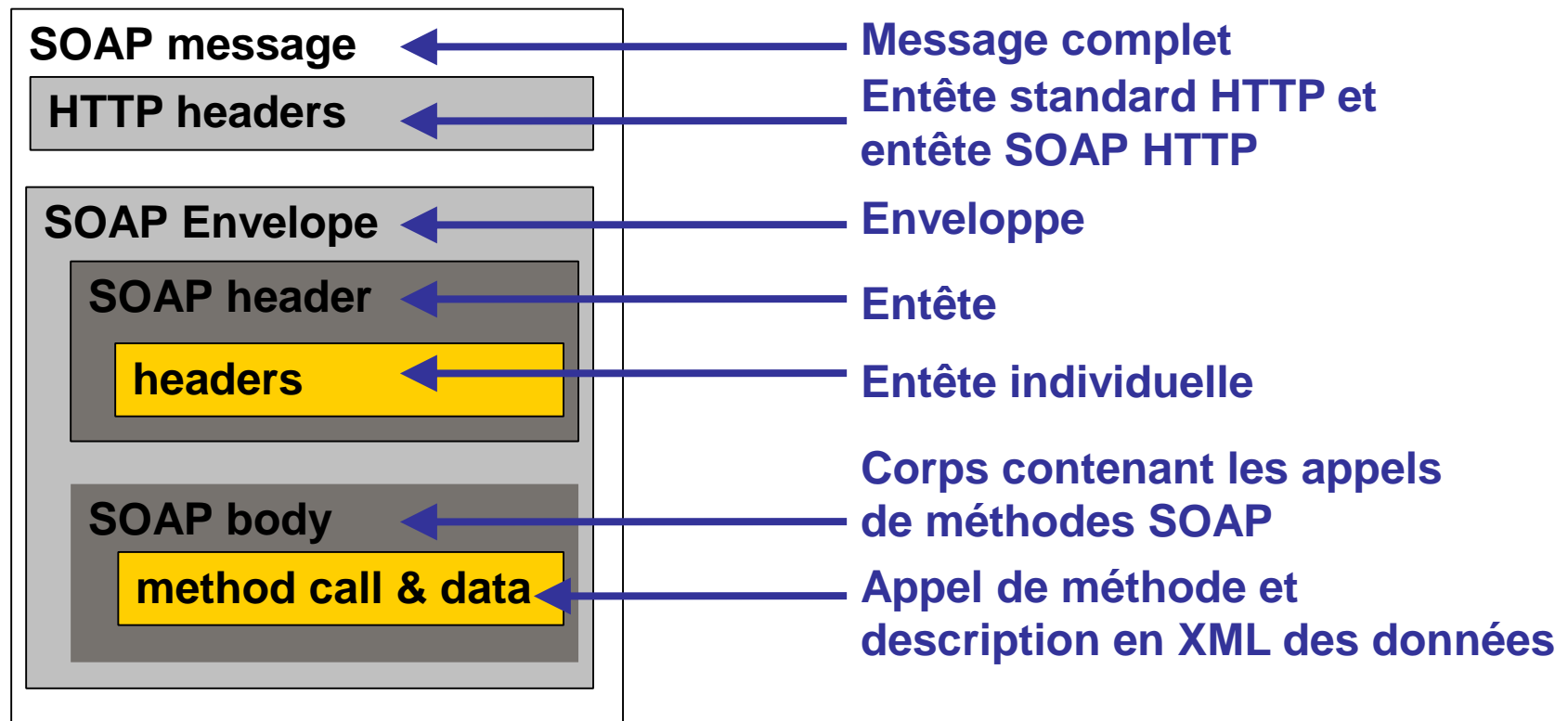
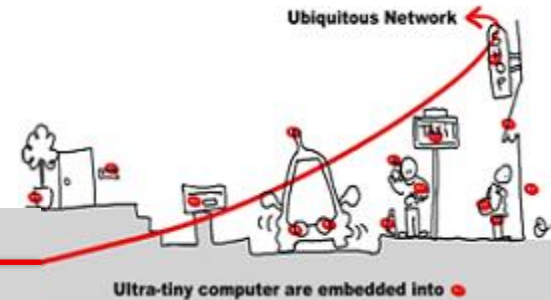
Rappel : La structure des messages WS-REST



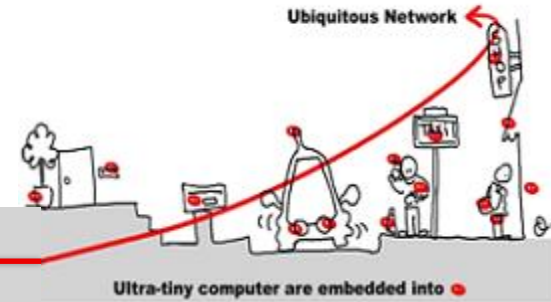
Rappel : SOAP pour une approche SOA



La structure des messages WS-SOAP



Exemple de requête SOAP utilisant HTTP

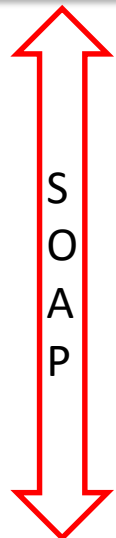


- Demande de cotation à un serveur :

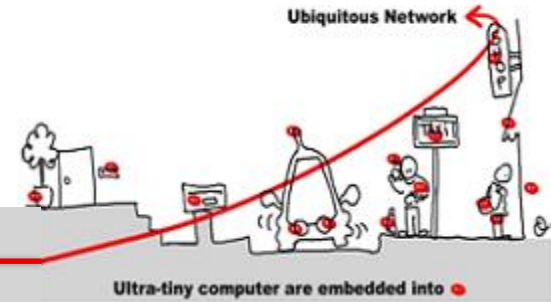
```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAP-Action: "Some-URI"
```

Une seule commande
HTTP/POST
« envoi de message SOAP »

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/enveloppe/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



Exemple de réponse SOAP utilisant HTTP



- Réponse du serveur

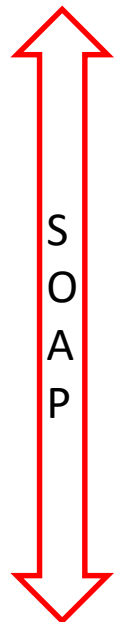
HTTP/1.1 200 OK

Content-Type: text/xml; charset="utf-8"

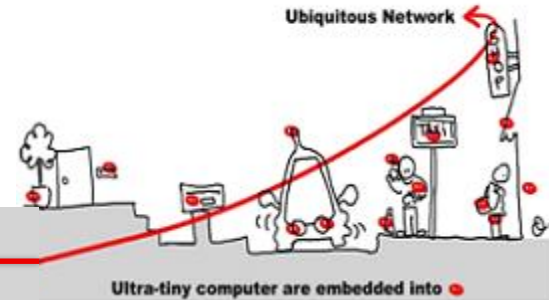
Content-Length: nnnn

Réponse au HTTP/POST

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="Some-URI">
      <Price>34.5</Price>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

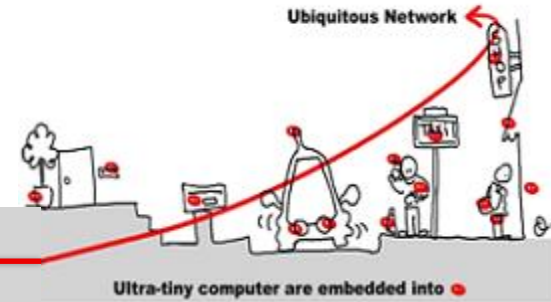


Modèle de message



- SOAP permet une communication par message
 - d'un expéditeur vers un récepteur
- Structure d'un message
 - Envelop (enveloppe)
 - Élément racine
 - Namespace : SOAP-ENV <http://schemas.xmlsoap.org/soap/envelope/>
 - Header (entête)
 - Élément optionnel
 - Contient des entrées non applicatives (transactions, session, ...)
 - Body (corps)
 - Contient les entrées du message
 - Nom d'une procédure, valeur des paramètres, valeur de retour
 - Peut contenir les éléments « fault » (erreurs)

Entête d'un message: Header



- Contient des entrées non applicatives
 - Transactions, sessions, ...
- L'attribut `mustUnderstand` par exemple
 - Si absent ou =0 l'élément est optionnel pour l'application réceptrice
 - si =1, l'élément doit être compris par l'application réceptrice sinon le traitement du message par le récepteur doit échouer

- Exemple

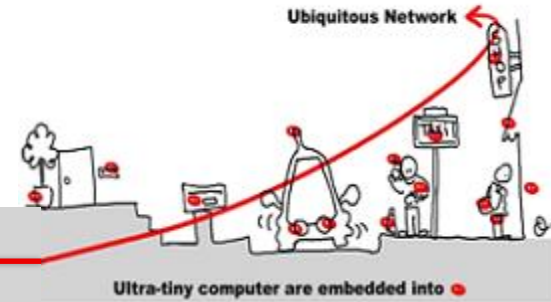
```
<SOAP-ENV:Header>
```

```
  <t:Transaction xmlns:t="Some-URI" SOAP-ENV:mustUnderstand="1">
```

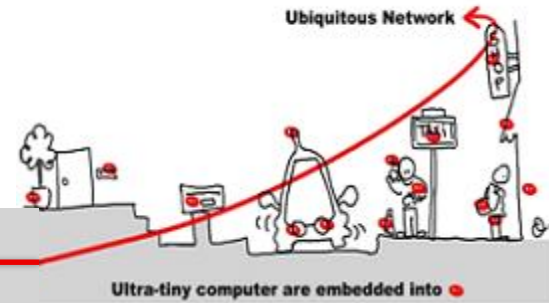
```
  </t:Transaction>
```

```
</SOAP-ENV:Header>
```

Corps d'un message

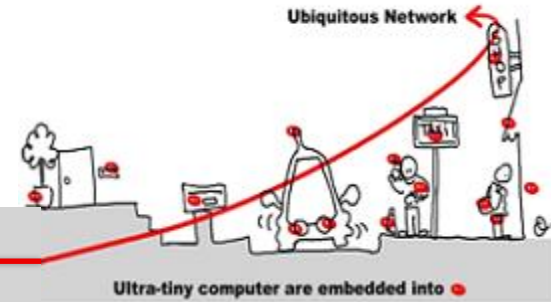


- Contient des entrées applicatives
- Encodage des entrées
- Namespace pour l'encodage
 - SOAP-ENC <http://schemas.xmlsoap.org/soap/encoding/>
 - XSD : Schéma XML
- Principe des règles d'encodage
 - Les règles d'encodage définissent un système de type
 - SOAP utilise les conventions XSD
 - Les tableaux et les références sont typés de manière spécifique en utilisant XSD



RÈGLES D'ENCODAGE

Types Primitifs



- Types primitifs

- <element name="age" type="int"/>

- <element name="price" type="float"/>

- <element name="displacement" type="negativeInteger"/>

- <element name="greeting" type="xsd:string"/>

- Utilisation / Instance

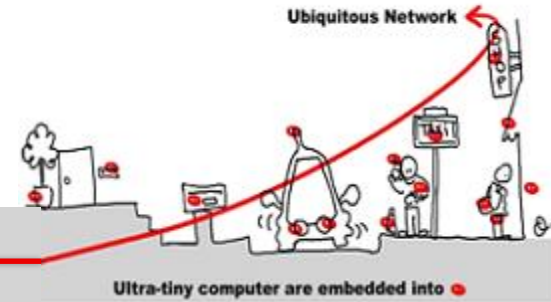
- <age>45</age>

- <price>15.57</price>

- <displacement>-450</displacement>

- <greeting>Hello</greeting>

Enumération



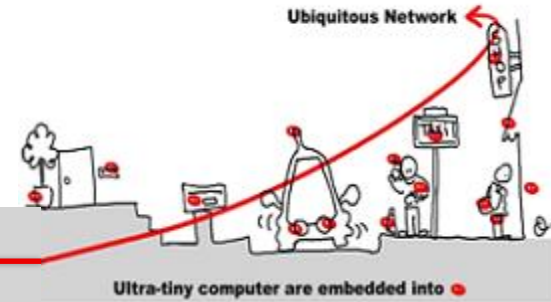
- Enumération

```
<element name="color">  
  <simpleType base="xsd:string">  
    <enumeration value="green"/>  
    <enumeration value="blue"/>  
  </simpleType>  
</element>
```

- Utilisation / Instance

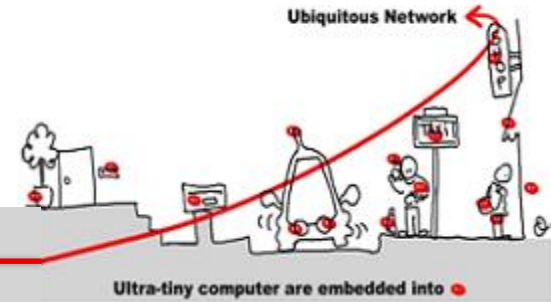
```
<color>blue</color>
```

Références



```
<element name="salutation" type="xsd:string"/>
<salutation href="#id1"/>
<e:Book>
  <title>My Life and Work</title>
  <firstauthor href="#Person-1"/>
  <secondauthor href="#Person-2"/>
</e:Book>
<e:Person id="Person-1">
  <name>Henry Ford</name>
  <address xsi:type="m:Electronic-address">
    <email>mailto:henryford@hotmail.com</email>
    <web>http://www.henryford.com</web>
  </address>
</e:Person>
<e:Person id="Person-2">
  <name>Samuel Crowther</name>
  <address xsi:type="n:Street-address">
    <street>Martin Luther King Rd</street>
    <city>Raleigh</city>
    <state>North Carolina</state>
  </address>
</e:Person>
```

Types Complexes: Structures



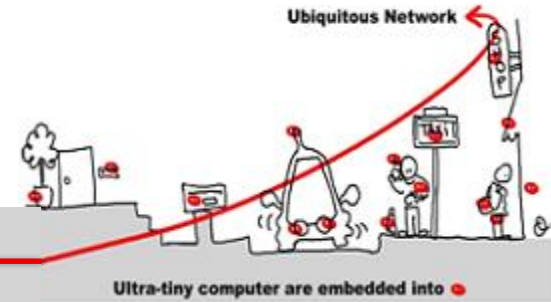
- Structures

```
<element name="book">  
  <complexType>  
    <element name="author" type="xsd:string"/>  
    <element name="title" type="xsd:string"/>  
  </complexType>  
</element>
```

- Utilisation / Instance

```
<e:book>  
  <author>J.R.R. Tolkien</author>  
  <title>A hobbit story</title>  
</e:book>
```

Types Complexes: Tableaux



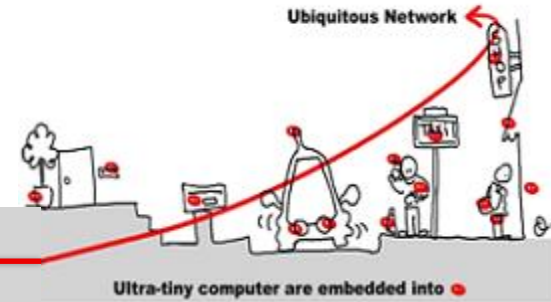
- Tableaux

```
<SOAP-ENC:Array id="id3" SOAP-ENC:arrayType=xsd:string[2,2]/>
  <item>r1c1</item>
  <item>r1c2</item>
  <item>r2c1</item>
  <item>r2c2</item>
</SOAP-ENC:Array>
```
- Tableaux d'octets

```
<picture xsi:type="SOAP-ENC:base64">
  aG913IGF0
</picture>
```
- Tableaux creux

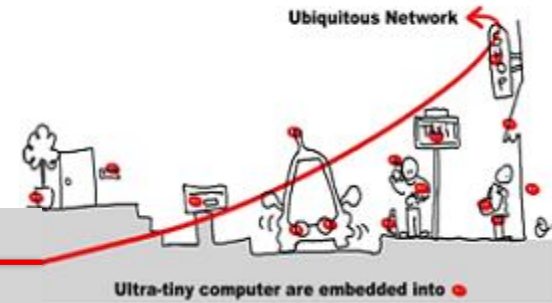
```
<SOAP-ENC:Array id="array-1" SOAP-ENC:arrayType=xsd:string[10,10]>
  <item SOAP-ENC:position="[2,2]">Third row, third col</item>
  <item SOAP-ENC:position="[7,2]">Eigth row, third col</item>
</SOAP-ENC:Array>
```


Retour d'Erreurs



- 4 éléments
 - faultcode (obligatoire)
 - Code d'erreur utilisé par le logiciel (switch(faultcode) { case ...
 - faultstring (obligatoire)
 - Explication lisible d' un humain
 - faultactor (optionel)
 - Erreur en cours de cheminement du message (firewall, proxy, MOM)
 - Detail
 - Détail de l' erreur non lié au Body du message
 - Autres
 - D' autres éléments qualifiés par un namespace peuvent être ajoutés
- Faultcode
 - 4 groupes de code d' erreur
 - Client, Server, MustUnderstand, VersionMismatch
 - Ex: Client.Authentication

Exemple de Retour d'Erreur



- Erreur sur le corps:

HTTP/1.1 500 Internal Server Error

Content-Type: text/xml; charset="utf8"

Content-Length: nnnn

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
```

```
<SOAP-ENV:Body>
```

```
<SOAP-ENV:Fault>
```

```
<faultcode>SOAP-ENV:Server</faultcode>
```

```
<faultstring>Server Error</faultstring>
```

```
<detail>
```

```
<e:myfaultdetails xmlns:e="Some-URI">
```

```
<message>my application didn't work</message>
```

```
<errorcode>1001</errorcode>
```

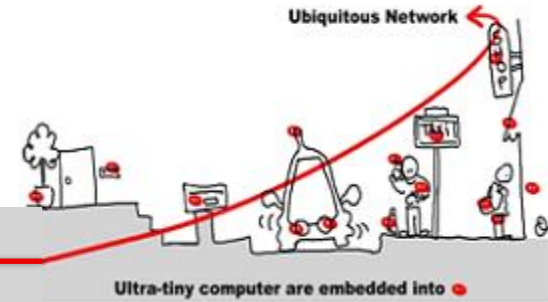
```
</e:mydefaultdetails>
```

```
</detail>
```

```
</SOAP-ENV:Fault>
```

```
</SOAP-ENV:Body>
```

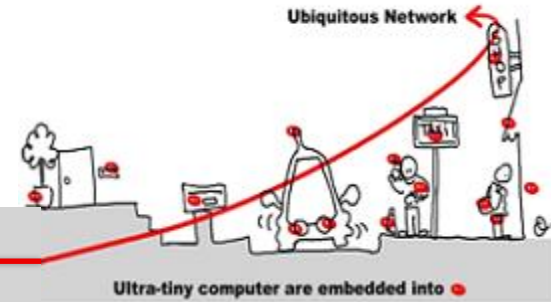
```
</SOAP-ENV:Envelope>
```



- « Contourner » les limitations

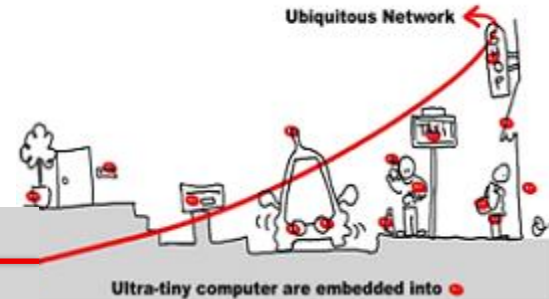
LES EXTENSIONS DE SOAP

Limitations de SOAP...



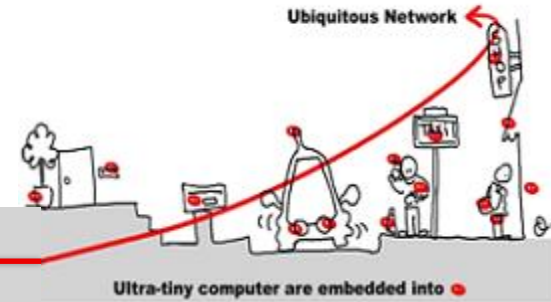
- Sécurité :
 - Limité à la sécurisation de HTTP ?
- Transfert de données :
 - Données échangées en XML. Donc
 - Données multi-parties ?
 - Données binaires ?
 - Fournir ou consommer des flux de données ?
- Des extensions pour répondre à ces problématiques

Sécurité



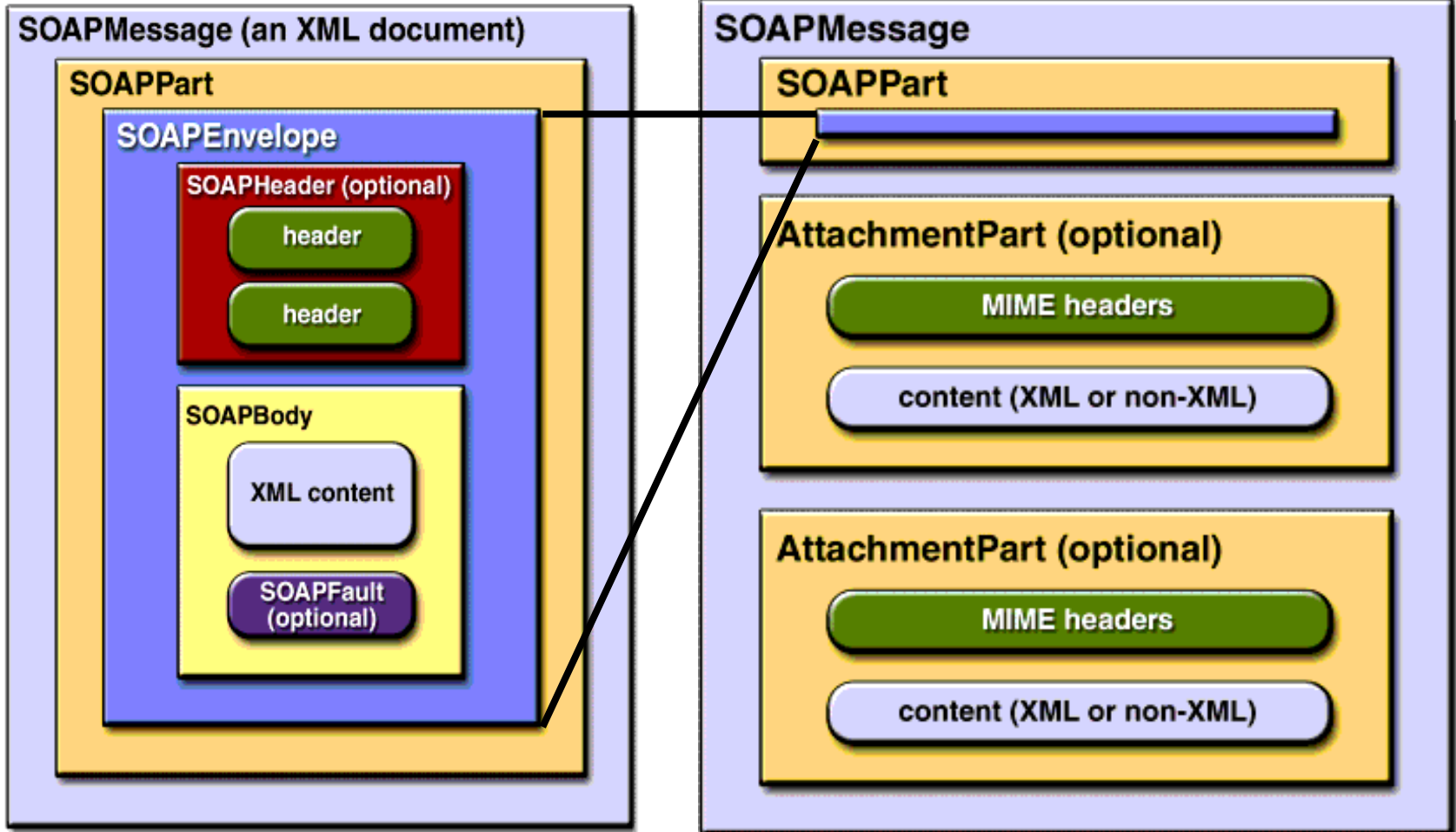
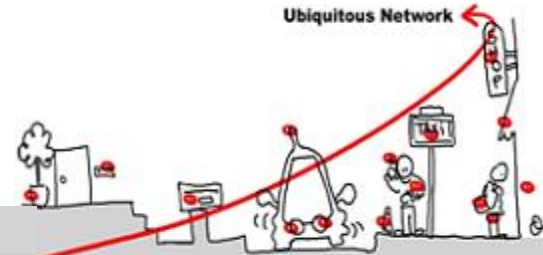
- Une sécurité « simple » mais efficace...
 - Basée sur la sécurité dans HTTP
 - HTTPS: assure une sécurité en point à point
 - S'intègre bien avec la politique des firewall
 - Pas de transfert de code applicatif
 - Uniquement des données
- ... Mais ne répond pas à tous les problématiques
 - Repousse le problème des failles
 - Déni de service, failles applicatives, ...
 - Pas un moyen de sécuriser de bout en bout dans des échanges
- Donc d'autres standards nécessaires
 - SAML 2.0: Security Assertion Markup Language (OASIS)
 - XACML 2.0: eXtensible Access Control Markup Language (OASIS)

Attachements avec SOAP

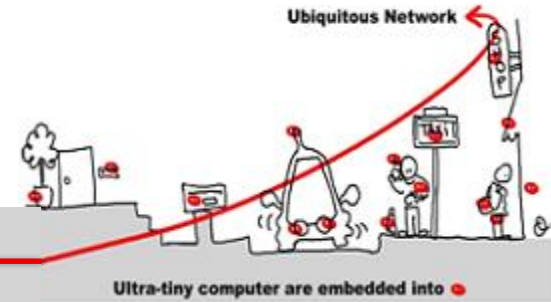


- SOAP with Attachments (SwA) ou MIME for Web Services
 - Utilise MIME
- MIME: Multipurpose Internet Mail Extensions
 - Pour le transport de gros fichiers
 - Placés en dehors de la partie XML
 - Avantages :
 - Simple
 - Inconvénients :
 - Impossible de faire du streaming
 - Le séparateur entre deux « attachements » MIME est une chaîne de caractères
 - Pas standardisé (juste une note du W3C)

Les Attachements MIME

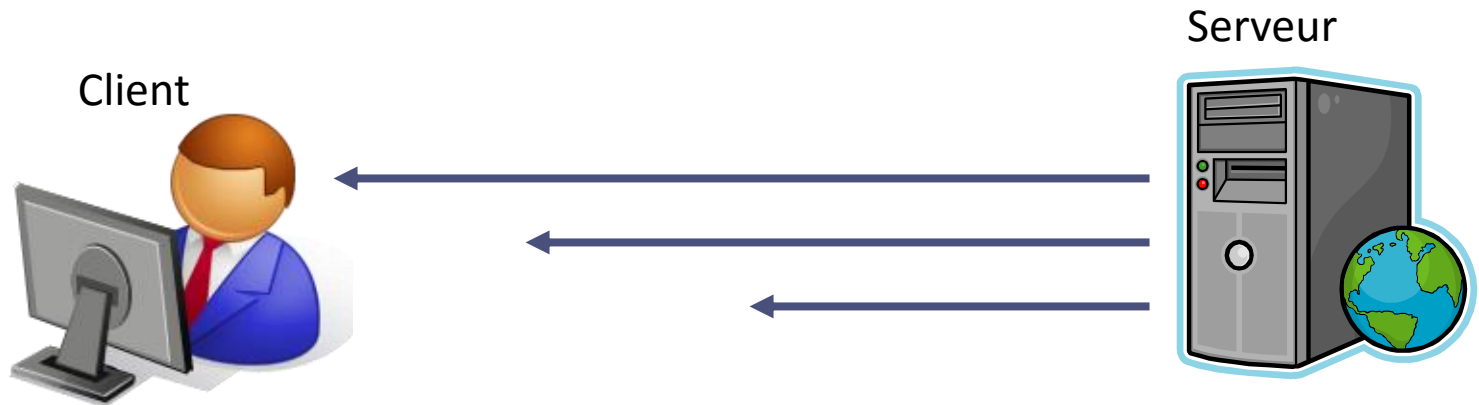
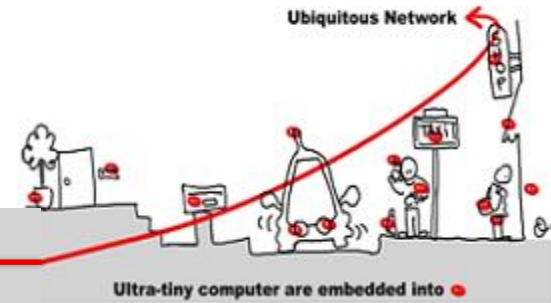


Les Attachements DIME



- DIME: Direct Internet Message Encapsulation
 - Proposé par Microsoft au début des années 2000
 - Pour le transport de flux continus
 - Placés en dehors de la partie XML
 - A chaque stream correspond un "DIME-record"
 - Un flag avertit le récepteur de la réception du dernier stream
 - Ainsi, il est possible de commencer à envoyer les premiers streams du flux avant même que tous les streams soient créés par la source
- Avantages
 - Simplicité
 - Permet la transmission de flux de données (binaires)
- Inconvénients
 - Impossible à déboguer (données binaires)
 - N'a jamais été standardisé (ni IETF, ni OASIS, ni W3C)

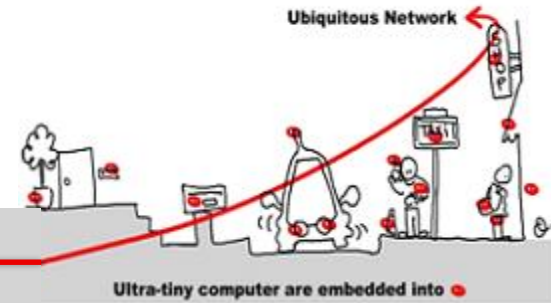
Autres extensions : les attachements DIME



Un DIME-Parser analyse et reconstitue le flux à partir des DIME-records envoyés par le serveur

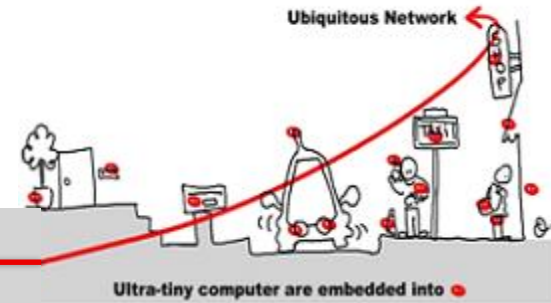
Un DIME-generator crée les streams et constitue les DIME-records à envoyer au client

Les attachements MTOM



- Message Transmission Optimization Mechanism :
 - Méthode pour l'envoi efficace de données binaires
 - Recommandation du W3C
 - Supplante SwA et DIME pour les Web Services
- Détails:
 - Abstract SOAP Transmission Optimization Feature
 - Optimized MIME Multipart/Related Serialization of SOAP Messages (XOP)
 - HTTP SOAP Transmission Optimization Feature
- Avantages:
 - Fournit un moyen d'envoyer des données binaires dans leur format d'origine (évitant l'augmentation des infos transmises)
 - La transformation en base64 augmente la taille des données de 33%

Portée et Limitations de SOAP



- SOAP est simple et extensible...
 - Format XML over HTTP
 - Multi-langages
 - Multi-plateformes
- ... mais il ne couvre pas les fonctions suivantes :
 - Distributed garbage collection
 - Objects-by-reference (qui requière en autre une distributed garbage collection)
 - Activation (qui requière objects-by-reference)