Arquitecturas Orientadas a Servicios

INVOCACIÓN DE SERVICIOS WEB

Manuel Lama Penín

manuel.lama@usc.es

Grupo de Sistemas Inteligentes
Departamento de Electrónica e Computación
Universidade de Santiago de Compostela

Invocación de servicios Web

- Los servicios Web están basados en la **tecnología XML** para facilitar la representación y el análisis de sus características.
  - A través de tecnologías como JAX-WS y JAXB se puede acceder a las características de los servicios para **generar el código** que los clientes usan a la hora de acceder a los servicios.
- Para ser consistente con la tecnología de servicios Web es necesario que el **protocolo de invocación** de los servicios también esté basado en un formato XML.
  - Se debería detallar la operación a invocar, los valores de sus argumentos y el resultado de dicha invocación.
    (el conjunto de mensajes que permiten **ejecutar una operación**)

```
<mat:suma>
   <x>9</x>
   <y>14</y>
</mat:suma>
```

```
<ns2:sumaResponse xmlns:ns2="http://matematicas/">
   <return>23</return>
</ns2:sumaResponse>
```

Invocación de servicios Web

- Se debería describir qué es lo que suceder cuando el contenido del mensaje enviado por el cliente **no se entiende** (por ejemplo, cuando los argumentos no son correctos) o **existe un fallo**.
- Se debería indica qué partes del mensaje deberán ser leídas (o procesadas) y **por quién**.

```
<env:Header>
 <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
          env:mustUnderstand="true">
  <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
  <m:dateAndTime>2001-11-30T16:25:00.000-05:00</m:dateAndTime>
 </m:reservation>
</env:Header>
```

- Se debería describir de qué manera los mensajes de invocación y de respuesta "**viajan**" a través de un protocolo de transporte. (como HTTP o SMTP)

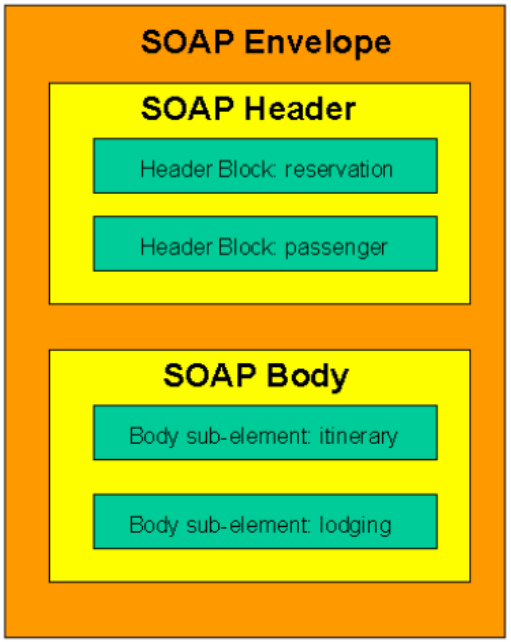
```
POST http://localhost:8080/Ejercicio11/CalculadoraBasicaService HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: ""
User-Agent: Jakarta Commons-HttpClient/3.1
Host: localhost:8080
Content-Length: 408

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mat="http://matematicas/"
  <soapenv:Header/>
```

Invocación de servicios Web >> SOAP

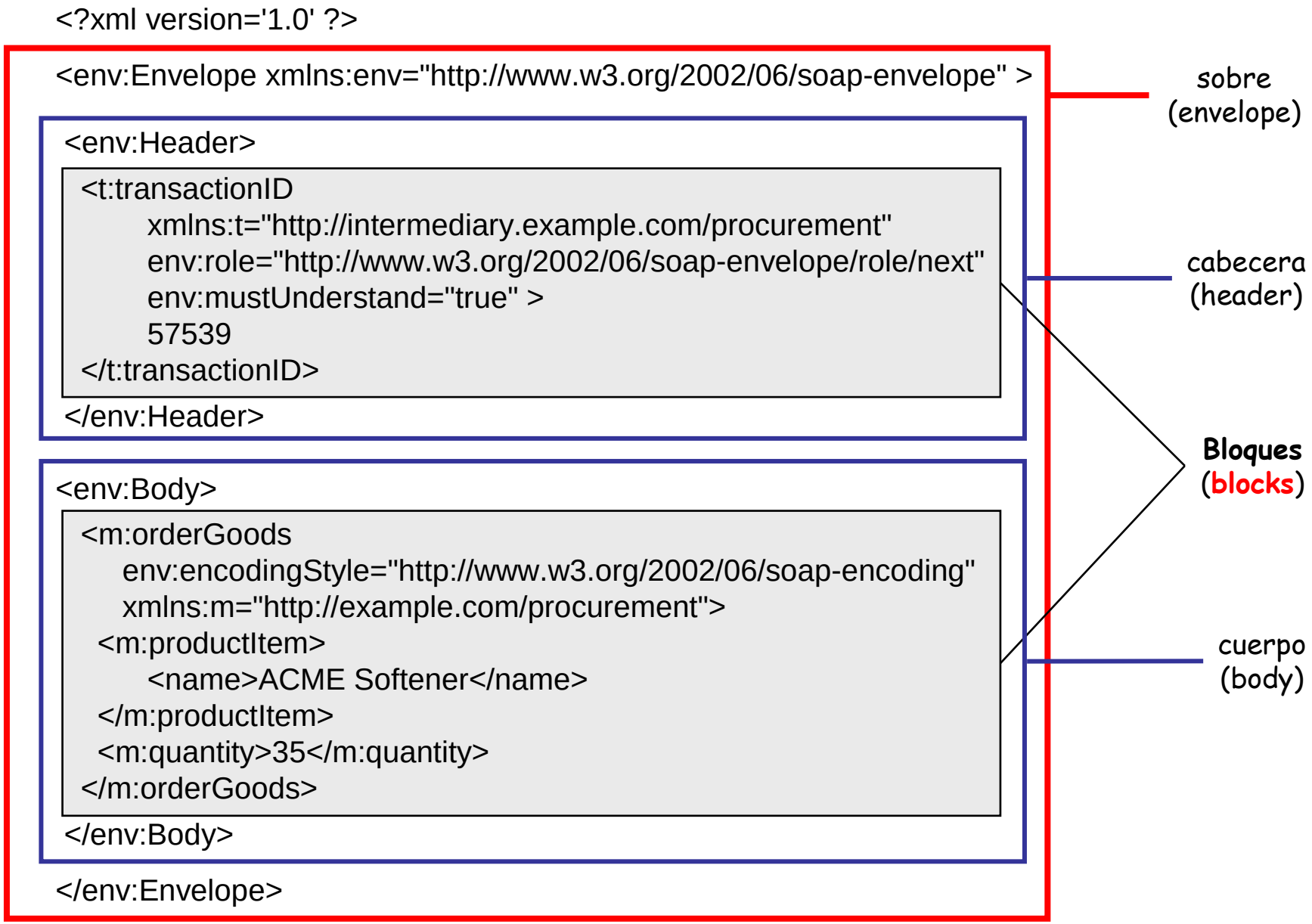
- En los servicios Web la invocación de las operaciones se realiza a través de un protocolo llamado **Simple Object Access Protocolo (SOAP)**.
  - Está basado en el **concepto de mensaje**: existe un intercambio de mensajes en los cuales se codifican las operaciones a invocar en el proveedor y la respuesta que recibirá el cliente.
  - **No tiene estado**: no existe una coreografía de mensajes en los que unos usan los resultados de otros.
  - La comunicación entre el cliente y el proveedor se establece en **una sola dirección** (one-way) y **de forma asíncrona**.
    - Para permitir una comunicación síncrona y un patrón de intercambio de mensajes más elaborado (por ejemplo, del tipo request-reponse) será necesario el **desarrollo de un middleware específico** en el que se que utilicen mensajes de una sola dirección y se aprovechen las capacidades del protocolo de transporte (por ejemplo, HTTP).

Invocación de servicios Web >> SOAP

- Los mensajes SOAP se entienden como **sobres** en los que se encierran en los que se encierran los datos que se intercambian entre los proveedores y los consumidores. (**envelopes**)
  - El sobre lo envía un **emisor** (es decir, el cliente), y
  - Lo recibe un **receptor** (es decir, el proveedor) o una serie de **intermediarios** que lo procesan y lo envían al receptor.
- Un sobre **encapsula** información:
  - De **cuerpo** (**body**), que contiene los datos que el cliente envía al proveedor para que la operación u operaciones asociadas al servicio puedan ser ejecutadas.
  - De **cabecera** (**header**), que contiene datos que pueden usar los intermediarios para ejecutar servicios o tomar decisiones.



Invocación de servicios Web >> SOAP

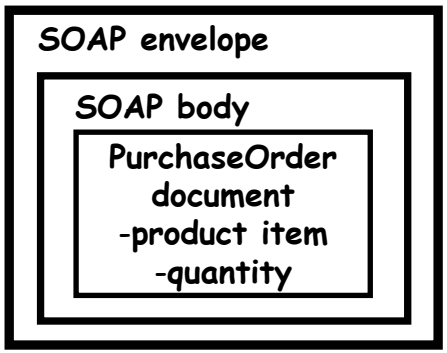


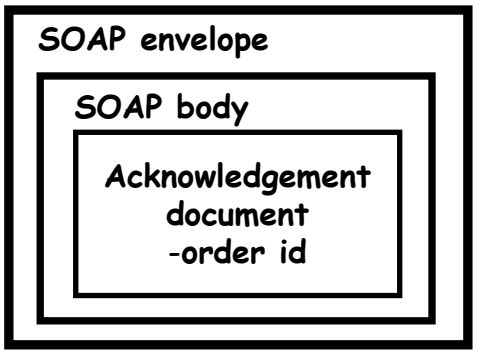
SOAP >> BODY >> Estilo documento

- En SOAP únicamente se especifica que los mensajes **deben tener un cuerpo** y que la cabecera es opcional.
  - No se impone ningún tipo de estructura o semántica ni para la cabecera ni para el cuerpo; es decir, se pueden enviar datos en **cualquier formato**.

- **Habitualmente** se consideran dos formas de construir y estructurar los cuerpos de los mensajes SOAP:
  - **Estilo documento** en el que los datos que se intercambian entre el emisor y el receptor están representados en un documento que puede tener cualquier formato (por ejemplo, HTML).

Mensaje SOAP de **invocación** enviado por el emisor





Mensaje SOAP de **respuesta** enviado por el receptor

SOAP >> BODY >> Estilo documento

Mensaje SOAP de **invocación** enviado por el remitente

```
<env:Body>
 <p:itinerario
   xmlns:p="http://empresaviajes.example.org/reserva/viaje">
  <p:ida>
    <p:salida>Nueva York</p:salida>
    <p:llegada>Los Angeles</p:llegada>
    <p:fechaSalida>2001-12-14</p:fechaLlegada>
    <p:horaSalida>última hora de la tarde</p:horaSalida>
    <p:preferenciaAsiento>pasillo</p:preferenciaAsiento>
  </p:ida>
  <p:vuelta>
    <p:salida>Los Angeles</p:salida>
    <p:llegada>Nueva York</p:llegada>
    <p:fechaSalida>2001-12-20</p:fechaSalida>
    <p:horaSalida>media-mañana</p:horaSalida>
    <p:preferenciaAsiento/>
  </p:vuelta>
 </p:itinerario>
 <q:alojamiento
  xmlns:q="http://empresaviajes.example.org/reserva/hoteles">
  <q:preferencia>ninguna</q:preferencia>
 </q:alojamiento>
</env:Body>
```

Mensaje SOAP de **respuesta** enviado por el receptor

```
<env:Body>
 <p:itineraryClarification
   xmlns:p="http://travelcompany.example.org/reservation/travel">
  <p:departure>
    <p:departing>
      <p:airportChoices>
         JFK LGA EWR
      </p:airportChoices>
    </p:departing>
  </p:departure>
  <p:return>
    <p:arriving>
      <p:airportChoices>
        JFK LGA EWR
      </p:airportChoices>
    </p:arriving>
  </p:return>
 </p:itineraryClarification>
</env:Body>
```

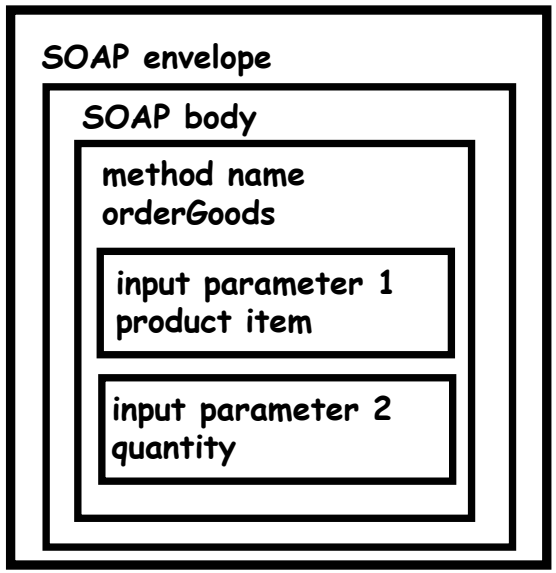
El cuerpo de los mensajes SOAP enviados por el remitente y por el receptor están representados en un **documento** que sigue un formato XML recomendado en la especificación.

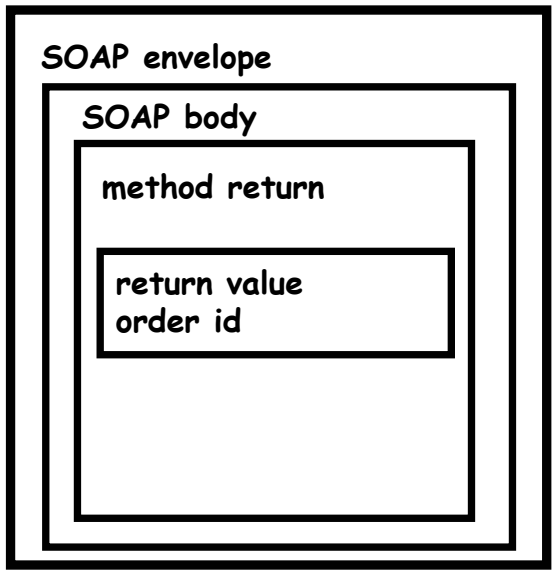
- Podría ser cualquier otro formato de documento.

SOAP >> BODY >> Estilo RPC

- **Estilo RPC** en el que los datos representan de forma explícita la operación o conjunto de operaciones que van a ser invocadas por el remitente del mensaje.
  - El mensaje SOAP de respuesta también está representado en un formato en el que se define la **operación** y los datos que **devuelve** como resultado de su ejecución.
- La especificación SOAP **no obliga a utilizar** un formato XML determinado a la hora de definir el mensaje RPC.

Mensaje SOAP de **invocación** enviado por el emisor





Mensaje SOAP de **respuesta** enviado por el receptor

SOAP >> BODY >> Estilo RPC

Mensaje SOAP de **invocación** enviado por el remitente

```
<env:Body>
 <m:chargeReservation
     env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
        xmlns:m="http://travelcompany.example.org/">
  <m:reservation xmlns:m="http://travelcompany.example.org/reservation">
   <m:code>FT35ZBQ</m:code>
  </m:reservation>
  <o:creditCard xmlns:o="http://mycompany.example.com/financial">
   <n:name xmlns:n="http://mycompany.example.com/employees">
          Åke Jógvan Øyvind
   </n:name>
   <o:number>123456789099999</o:number>
   <o:expiration>2005-02</o:expiration>
  </o:creditCard>
 </m:chargeReservation>
</env:Body>
```

Operación: chargeReservation

Argumentos:

- **reservation**
  - code = FT352BQ
- **creditCard**
  - name = Ake Jógban Oyvind
  - number = 123456789099999
  - expiration = 2005-02

Mensaje SOAP de **respuesta** enviado por el receptor

```
<env:Body>
   <m:chargeReservationResponse
      env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
          xmlns:rpc="http://www.w3.org/2003/05/soap-rpc"
            xmlns:m="http://travelcompany.example.org/">
      <rpc:result>m:status</rpc:result>
      <m:status>confirmed</m:status>
      <m:code>FT35ZBQ</m:code>
      <m:viewAt>
       http://travelcompany.example.org/reservations?code=FT35ZBQ
      </m:viewAt>
   </m:chargeReservationResponse>
</env:Body>
```

Operación: chargeReservation

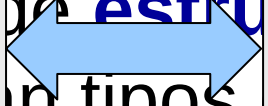
Resultado:

- **status** = confirmed
- **code** = FT352BQ
- **viewAt** = http://travelcompany...

SOAP >> BODY >> Reglas de codificación SOAP

- Aunque SOAP no impone ningún formato XML en el cuerpo de los mensajes existen **recomendaciones** a la hora de representar los tipos de datos.
  - **http://www.w3.org/2003/05/soap-encoding**
  - Estas recomendaciones son las que sigue **JAX-WS** a la hora de interpretar el cuerpo de los mensajes para traducirlos a instancias de las clases que son los argumentos de las operaciones.
- de estru ue an tipos

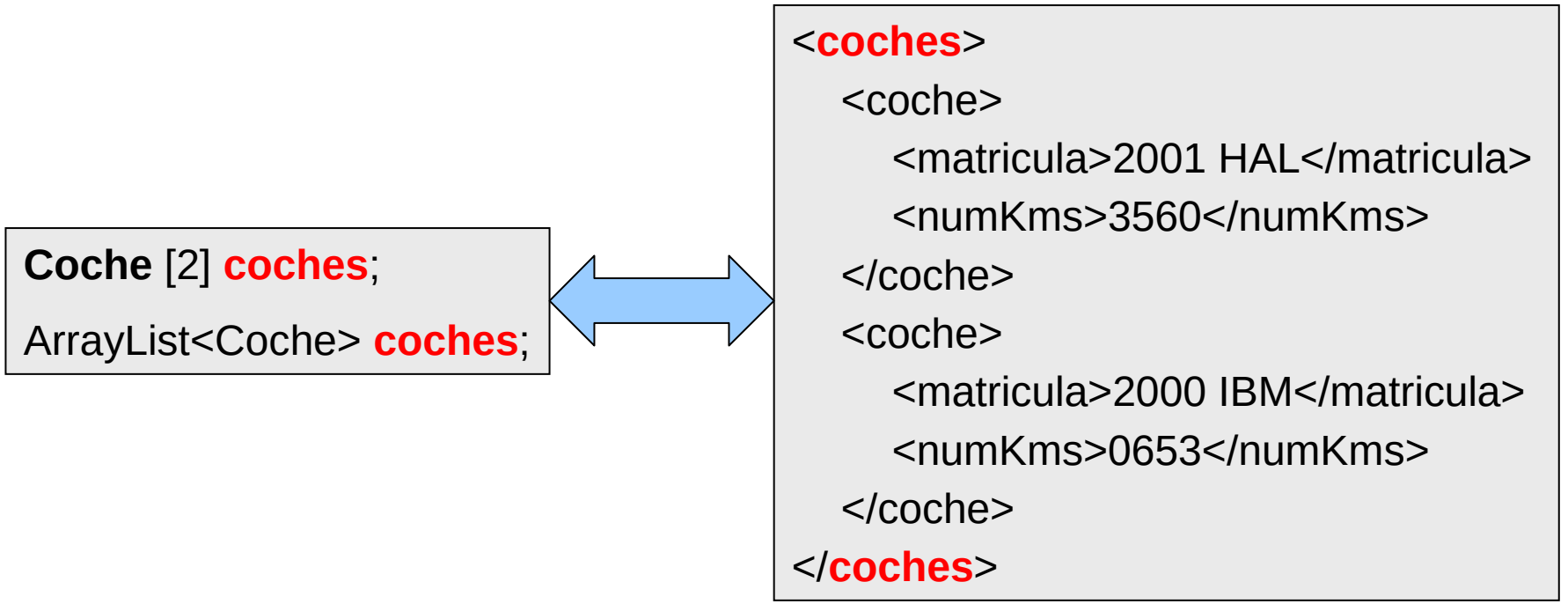
```
class Coche {
    String matricula;
    int numKms;
}
```



```
<Coche xmlns="http://taller.com">
    <matricula>2001 HAL</matricula>
    <numKms>3560</numKms>
</Coche>
```

SOAP >> BODY >> Reglas de codificación SOAP

- Codificación de **arrays** o de **conjuntos de datos** que pueden ser de cualquier tipo:
  - De tipo complejo, o
    (equivalente a **clases o estructuras**)
  - Primitivos.
    (definidos en la especificación **XML Schema Datatypes**)



SOAP >> BODY >> Reglas de codificación SOAP

operación del servicio web

```
@WebMethod(operationName = "sumaLista")
public int sumaLista(@WebParam(name = "datos") ListaDatos datos) {
    int resultado= 0;
    for(Integer dato : datos.getDatos()) resultado+= dato;
    return resultado;
}
```

mensaje SOAP para invocar la operación sumaLista

```
<soapenv:Body>
   <mat:sumaLista>
      <datos>
         <datos>12</datos>
         <datos>12</datos>
         <datos>12</datos>
         <numeroDatos>3</numeroDatos>
      </datos>
   </mat:sumaLista>
</soapenv:Body>
```

```
public class ListaDatos {
    private ArrayList<Integer> datos;
    private int numeroDatos;

    public ListaDatos() {
        datos= new ArrayList<Integer>();
    }

    public ArrayList<Integer> getDatos() {
        return datos;
    }

    public void setDatos(ArrayList<Integer> datos) {
        this.datos = datos;
    }

    public int getNumeroDatos() {
        return numeroDatos;
    }

    public void setNumeroDatos(int numeroDatos) {
        this.numeroDatos = numeroDatos;
    }
```

SOAP >> HEADER

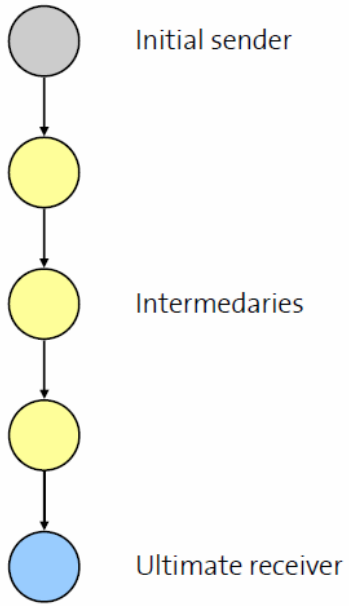
- Incluye información que típicamente puede ser usada por las **aplicaciones que manejan los mensajes SOAP** para tomar decisiones acerca de su procesamiento. (información **específica** de la aplicación)
  - En un mensaje SOAP la cabecera es **opcional**.
  - Es posible que la aplicación que recibe el mensaje **desconozca** que tiene una cabecera.
  - Habitualmente la cabecera **contiene información** que se usará para diferentes propósitos (depende de **cada** aplicación):
    - **Seguridad** (certificados).
    - **Identificadores** (transacciones).
    - **...**

```
<env:Header>
   <t:transaction
      xmlns:t="http://thirdparty.example.org/transaction"
        env:encodingStyle="http://example.com/encoding"
         env:mustUnderstand="true">5</t:transaction>
</env:Header>
```

SOAP >> HEADER >> Atributo role

- Un mensaje SOAP puede pasar por varios nodos que son susceptibles de **procesar los bloques de la cabecera**.
  - Los nodos intermedios pueden ejecutar determinados servicios de valor añadido o tomar decisiones en función de los datos de la cabecera.
  - Únicamente el **nodo final** puede procesar el cuerpo del mensaje SOAP enviado por el nodo inicial.

- En la cabecera se indica **qué nodos** pueden procesar los bloques de la cabecera a través del atributo **env:role**.
  - Valor **none** indica que el bloque correspondiente no debe ser procesado por ningún nodo.
  - Valor **next** indica que cualquier nodo que reciba el mensaje SOAP puede procesar el bloque.



SOAP >> HEADER >> Atributo role

```
<env:Header>
 <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
           env:mustUnderstand="true">
  <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
  <m:dateAndTime>2001-11-29T13:35:00.000-05:00</m:dateAndTime>
 </m:reservation>
 <n:passenger xmlns:n="http://mycompany.example.com/employees"
      env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
           env:mustUnderstand="true">
  <n:name>Åke Jógvan Øyvind</n:name>
 </n:passenger>
</env:Header>
```

  - Valor **ultimateReceiver** indica que cualquier nodo que reciba el mensaje SOAP puede procesar el bloque.

- **Resumen**: procesamiento de la cabecera en función del valor del atributo **env:role**.

Role	absent	"none"	"next"	"ultimateReceiver"
Node				
initial sender	not applicable	not applicable	not applicable	not applicable
intermediary	no	no	yes	no
ultimate receiver	yes	no	yes	yes

SOAP >> HEADER >> Atributo mustUnderstand

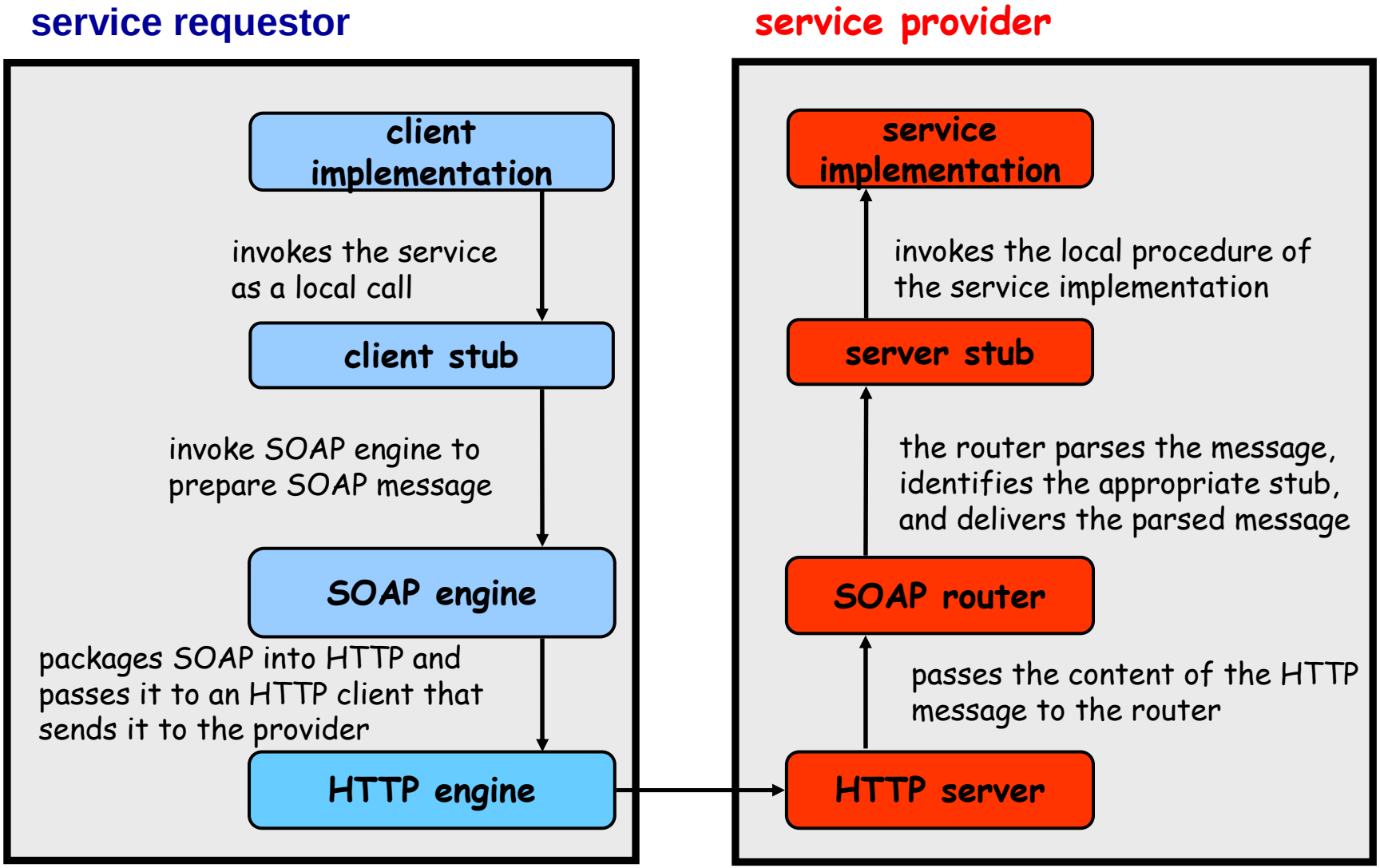
- Indica si un nodo **debe** procesar el bloque de la cabecera.
  - Este atributo tiene sentido si el nodo **puede** procesar la cabecera de acuerdo con el valor del atributo **env:role**.
  - Es un atributo booleano: **true** (debe procesarla) y **false** (no tiene que procesarla necesariamente).

```
<env:Header>
    <t:transaction
       xmlns:t="http://thirdparty.example.org/transaction"
         env:encodingStyle="http://example.com/encoding"
          env:mustUnderstand="true">5</t:transaction>
</env:Header>
```

- **Resumen**: combinación de los atributos ***mustUnderstand*** y ***role*** para requerir el procesamiento de la cabera.

Node	intermediary	ultimate receiver
mustUnderstand		
"true"	must process	must process
"false"	may process	may process
absent	may process	may process

Ejecución de mensajes SOAP



Ejecución de mensajes SOAP