Arquitecturas Orientadas a Servicios

DESCRIPCIÓN Y DESPLIEGUE DE SERVICIOS WEB

Manuel Lama Penín

manuel.lama@usc.es

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

Descripción de servicios Web >> WSDL

- Los servicios Web están descritos en un lenguaje basado en XML Schema denominado **Web Service Description Language (WSDL)**.

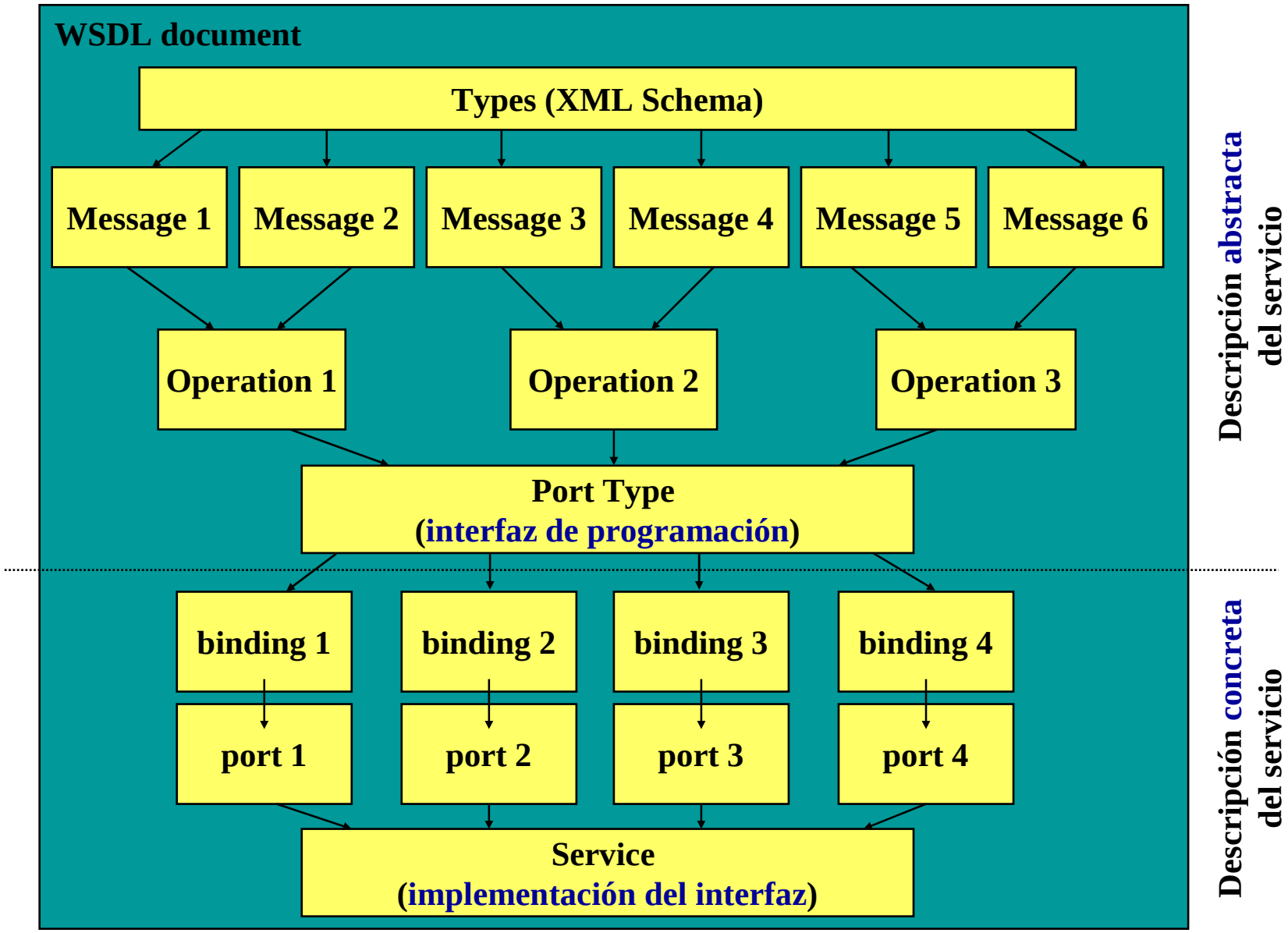
- WSDL describe un servicio Web desde **dos** perspectivas diferentes y complementarias:
  - **Servicio abstracto**, en el que se describen las características funcionales de las operaciones (**los interfaces de programación**) que son accesibles a través de la Web. (**reutilización de los interfaces**)
  - **Servicio concreto**, en el que se detalla la forma en la que el servicio abstracto será **invocado** por los clientes y el punto de la red en el que se **encuentran** las operaciones.

  Los clientes WSDL deberán realizar el **mapeo** entre la descripción abstracta del servicio Web y su realización concreta.

Descripción de servicios Web >> WSDL

- Para que un cliente pueda invocar la ejecución de las operaciones asociadas a un servicio Web es necesario que **el documento WSDL que lo describe sea accesible** a través de la Web.
  - Un documento WSDL **tiene asociada una URL** que lo hace disponible a cualquier programa.
  - Típicamente el servidor Web que aloja el fichero WSDL también dará soporte al **servidor de aplicaciones** en el que se **despliegan** las operaciones del servicio Web.
  - Los clientes de los servicios Web acceden a la URL del documento WSDL y generan **automáticamente** el código necesario para poder realizar la invocación de las operaciones.
    - Para generar este código se necesita **acceder** tanto a la parte **abstracta** (para generar las clases de los tipos de datos) como a la **concreta** de descripción del servicio Web (para conocer qué protocolo se usará en la invocación).

Descripción de servicios Web >> WSDL



WSDL >> TYPES

- Describen los **tipos de datos** asociados a los parámetros de las operaciones de los servicios web.
  - **Entradas**: los argumentos de las operaciones.
  - **Salidas**: lo que devuelven las operaciones.

- Están especificados en el **lenguaje XML Schema**, que permite expresar la estructura de un modelo de datos sin necesidad de incluir una DTD externa.
  - Se pueden crear tipos de datos complejos con **atributos** (tipos de datos básicos) y **relaciones** (otros tipo de datos complejos).
  - Se puede especificar la **multiplicidad de los atributos** de los tipos de datos complejos (**0 … ***).
    (Se pueden especificar **listas** o arrays de datos)
  - Los types pueden estar definidos en un fichero externo.
    (**importar**)

WSDL >> TYPES >> XML Schema

Tipos de datos **predefinidos** en el estándar XML Schema.

XML Schema Type	Java Data Type
xsd:string	java.lang.String
xsd:integer	java.math.BigInteger
xsd:int	int
xsd.long	long
xsd:short	short
xsd:decimal	java.math.BigDecimal
xsd:float	float
xsd:double	double
xsd:boolean	boolean
xsd:byte	byte
xsd:QName	javax.xml.namespace.QName
xsd:dateTime	javax.xml.datatype.XMLGregorianCalendar
xsd:base64Binary	byte[]
xsd:hexBinary	byte[]
xsd:unsignedInt	long
xsd:unsignedShort	int
xsd:unsignedByte	short
xsd:time	javax.xml.datatype.XMLGregorianCalendar
xsd:date	javax.xml.datatype.XMLGregorianCalendar
xsd:g	javax.xml.datatype.XMLGregorianCalendar
xsd:anySimpleType	java.lang.Object
xsd:anySimpleType	java.lang.String
xsd:duration	javax.xml.datatype.Duration
xsd:NOTATION	javax.xml.namespace.QName

WSDL >> TYPES >> XML Schema

PURCHASE ORDER TYPE

```
<element name="PO" type="tns:POType"/>
<complexType name="POType">
   <all>
      <element name="id" type="string”/>
      <element name="name" type="string"/>
      <element name="items">
         <complexType>
            <all>
               <element name="item" type="tns:Item" minOccurs="0" maxOccurs="unbounded"/>
            </all>
         </complexType>
      </element>
   </all>
</complexType>
```

ITEM TYPE

```
<complexType name="Item">
   <all>
      <element name="quantity" type="int"/>
      <element name="product" type="string"/>
   </all>
</complexType>
```

INVOICE TYPE

```
<element name="Invoice" type="tns:InvoiceType"/>
<complexType name="InvoiceType">
   <all>
      <element name="id" type="string"/>
   </all>
</complexType>
```

WSDL >> MESSAGES

- Se pueden entender como las "variables" del documento que serán los **parámetros de entrada y salida** de las operaciones del servicio.
  - Las **entradas** de las operaciones están descritas por un mensaje.
  - Las **salidas** de las operaciones están descritas por un mensaje.
  - Constan de varias partes (**parts**), cada una de las cuales está relacionada con los parámetros de entrada/salida.
  - Dos o más variables pueden formar parte de un mensaje. (Una operación puede tener más de un argumento de entrada y **puede devolver más de una variable de salida**)
  - Un mensaje **no define** ninguna forma de interacción o coreografía entre el proveedor y el consumidor.

```
<message name="PO">
      <part name="po" element= tns:PO/>
      <part name="invoice" element="tns:Invoice" />
<message/>
```

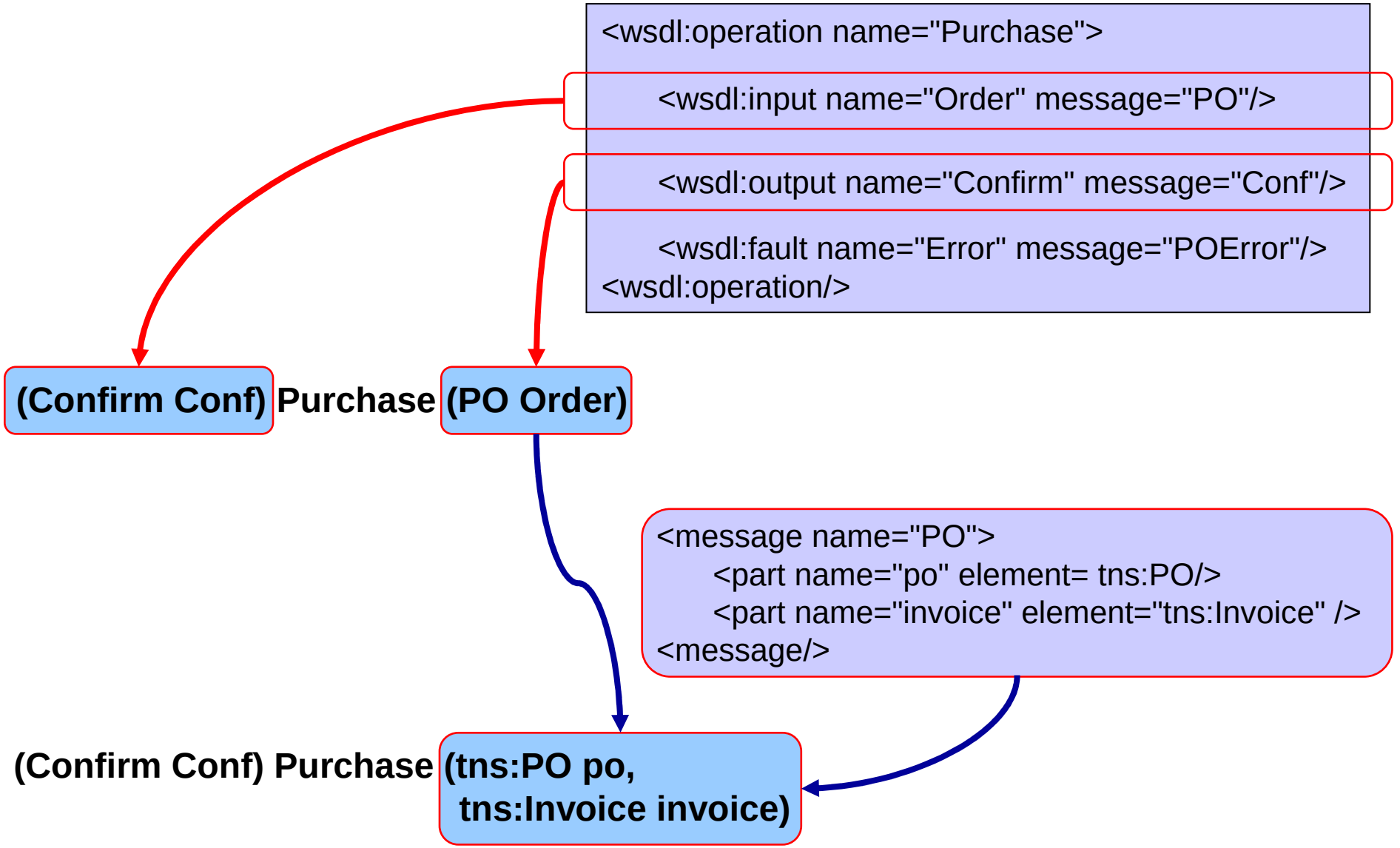
WSDL >> OPERATIONS

- Describen desde un punto de vista abstracto y en formato XML las operaciones o **métodos de los servicios Web** que son accesibles a los clientes.
  - **Nombre**.
  - Mensaje de **entrada**.
  - Mensaje de **salida**.

```
<wsdl:operation name="Purchase">
    <wsdl:input name="Order" message="PO"/>
    <wsdl:output name="Confirm" message="Conf"/>
    <wsdl:fault name="Error" message="POError"/>
<wsdl:operation/>
```

- **Tipos básicos** de operaciones:
  - **one-way**: el cliente envía un mensaje al servidor (**solo entrada**).
  - **request-response**: donde el cliente envía un mensaje y el servidor responde con otro mensaje (**entrada-salida**).
  - **solicit-response**: el servidor envía un mensaje y el cliente responde con otro mensaje (**entrada-salida**).
  - **Notification**: el servidor envía un mensaje al cliente (**solo salida**).

WSDL >> OPERATIONS



WSDL >> PORT TYPES

- Se corresponde con lo que entendemos por **interfaz de programación**; en este caso, interfaz que describe el servicio Web.
  - Es la **definición abstracta** de un servicio Web.
  - Está formada por **las operaciones** que son accesibles a través de Internet.
- Un *port type* **no** especifica ni la forma en la que se invoca el servicio Web ni da ningún detalle acerca del servidor en el que está desplegado.
  - Define el servicio como un **servicio abstracto**.

```
<message name="m1">
  <part name="body" element="tns:GetCompanyInfo"/>
</message>

<message name="m2">
 <part name="body" element="tns:GetCompanyInfoResult"/>
 <part name="docs" type="xsd:string"/>
 <part name="logo" type="tns:ArrayOfBinary"/>
</message>

<portType name="pt1">
  <operation name="GetCompanyInfo">
      <input message="m1"/>
      <output message="m2"/>
  </operation>
</portType>
```

WSDL >> BINDINGS

- Define el **formato de los mensajes y el protocolo** que se utilizará para invocar la ejecución de las operaciones especificadas en el servicio abstracto.
  - Un binding es la **realización concreta** de un port type. (atributo **type** del elemento binding)
  - Para un determinado port type (**interfaz de las operaciones**) se pueden definir varios bindings.
  - Un binding está ligado a **un único protocolo**. (puede haber un único elemento **soap:binding**)
- Las operaciones definidas en el binding se **mapean** con las operaciones especificadas en el port type al que está ligado el binding.
  - $\textbf{Operation}_{\textbf{PORT\_TYPE}} \leftrightarrow \textbf{Operation}_{\textbf{BINDING}}$ (esto es especialmente relevante cuando el estilo de invocación es **rpc**)

WSDL >> BINDINGS

- El identificador **soap:operation** describe las operaciones del servicio desde el punto de vista de su invocación a través del protocolo SOAP.
  - El identificador **soap:Action** se puede entender como el "método" que se ejecutará.
- Los parámetros del método (**soap:Action**) se codifican atendiendo a dos estilos:
  - **Estilo RPC** en el que los parámetros hacen una referencia directa a la variables (**parts**) definidas en los mensajes.

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
 <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetTradePrice">
    <soap:operation soapAction="http://example.com/GetTradePrice"/>
    <input>
      <soap:body use="encoded" namespace="http://example.com/stockquote"
       encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded" namespace="http://example.com/stockquote"
       encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
  </operation>>
</binding>
```

WSDL >> BINDINGS >> Invocación SOAP

- **Estilo document** en el que los parámetros se codifican en un formato XML determinado que típicamente hace referencia a un tipo de contenidos.

```
<operation name="GetCompanyInfo">
    <soap:operation
        soapAction="http://example.com/GetCompanyInfo"/>
    <input>
        <soap:body use="literal"/>
    </input>
    <output>
        <mime:multipartRelated>
            <mime:part>
                <soap:body use="literal"/>
            </mime:part>
            <mime:part>
                <mime:content part="docs" type="text/html"/>
            </mime:part>
            <mime:part>
                <mime:content part="logo" type="image/gif"/>
                <mime:content part="logo" type="image/jpeg"/>
            </mime:part>
        </mime:multipartRelated>
    </output>
</operation>
```

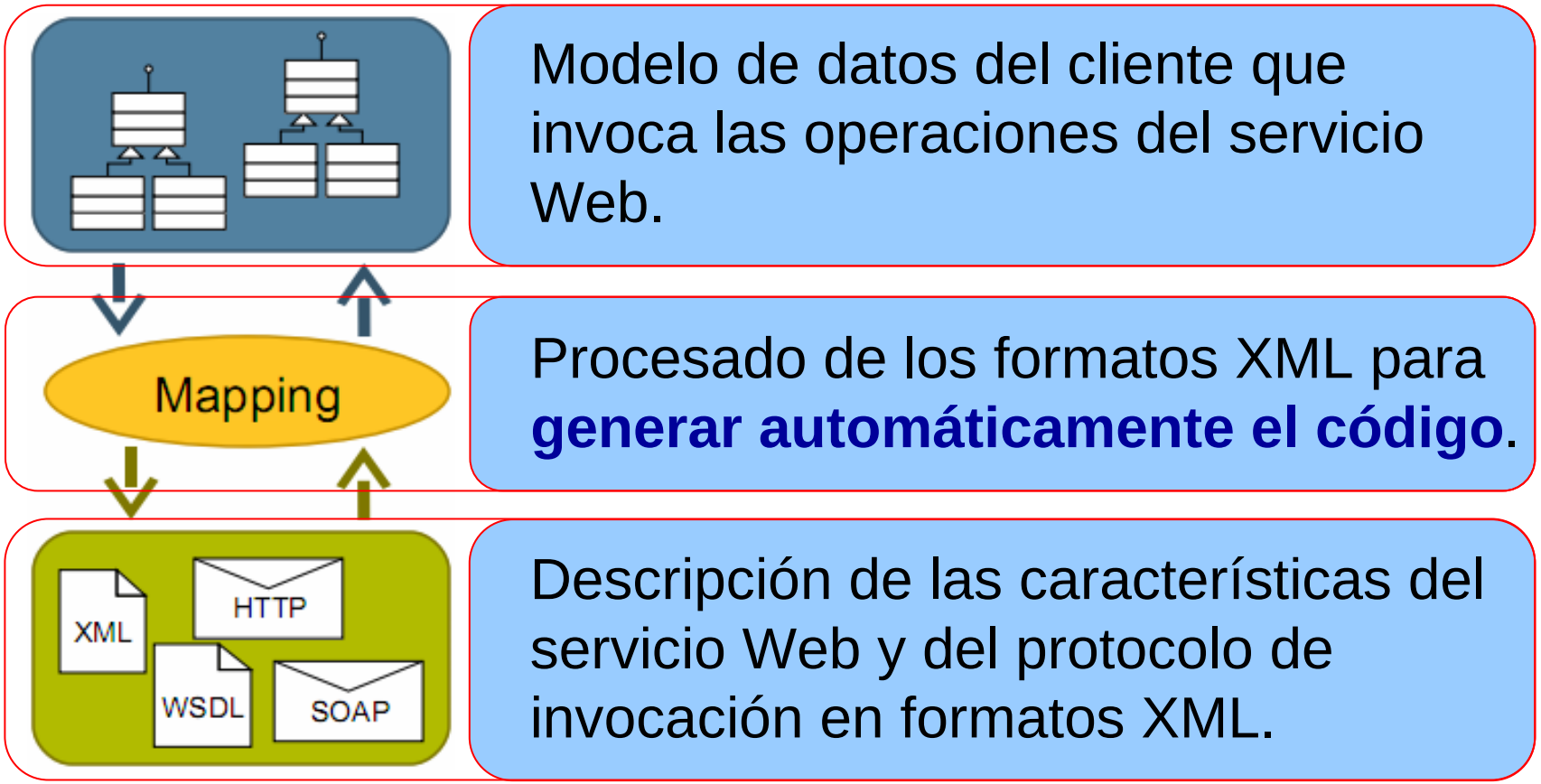
WSDL >> SERVICES

- Consisten en una colección de puertos en los que se define el punto final (**soap:address**) que el servidor de aplicaciones tiene abierto para recibir peticiones de ejecución de las operaciones.
  - Las operaciones que se ejecutarán en esa localización se especifican a través de los bindings (que contienen los elementos **soap:operation**).
  - Los puertos que forman parte del mismo servicio **no se pueden comunicar entre sí**.
    (son **alternativas** a la hora de acceder al punto final en el que se encuentran las operaciones)

```
<service name= "CompanyInfoService">
    <documentation>Information service of a company</documentation>
    <port name= "CompanyInfoPort" binding="tns:b1">
        <soap:address location="http://example.com/companyInfo"/>
    </port>
</service>
```

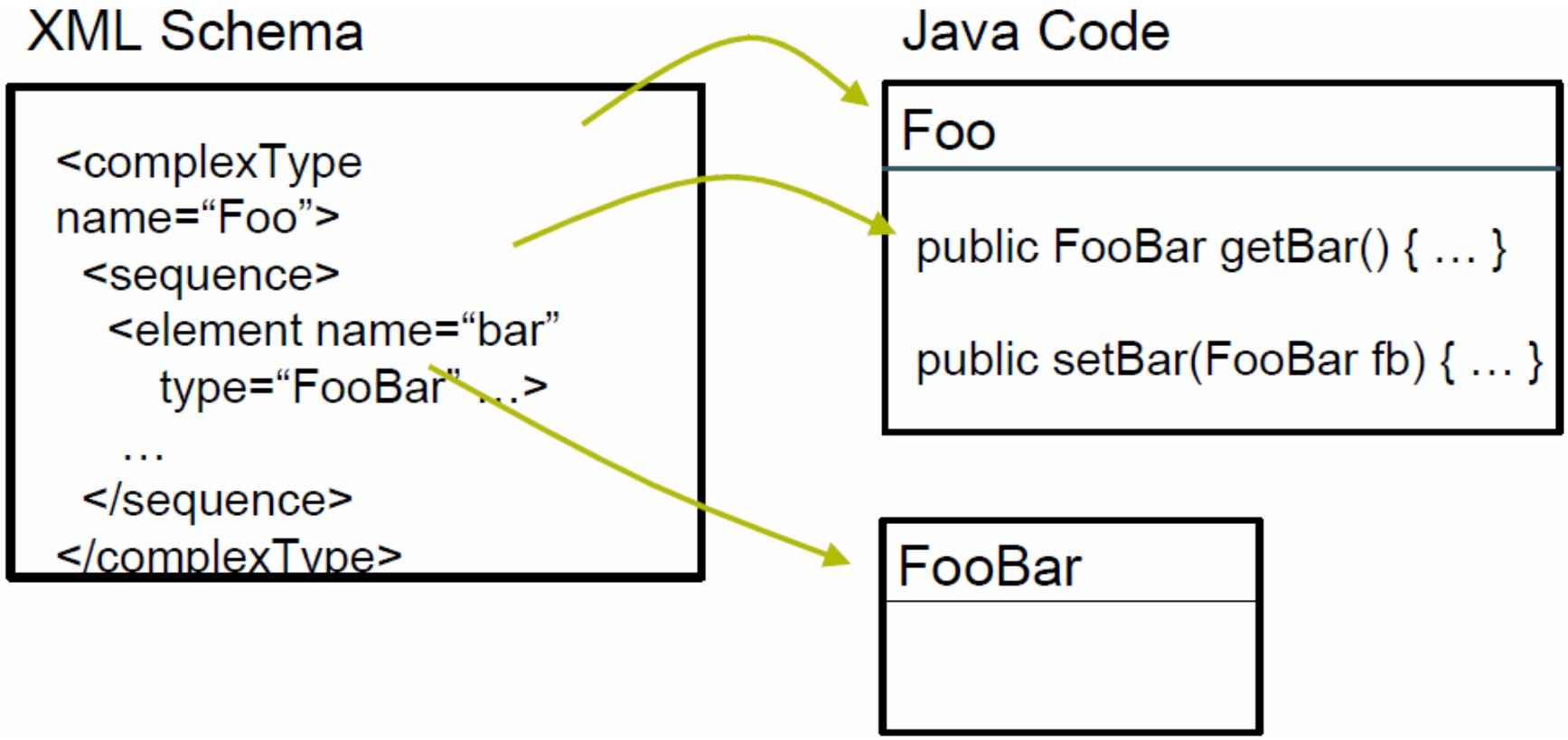
Procesamiento WSDL

- WSDL está representado en XML Schema debido a que permite **automatizar** la traducción de los componentes del servicio a los modelos de datos de los lenguajes de programación (específicamente **a clases**).



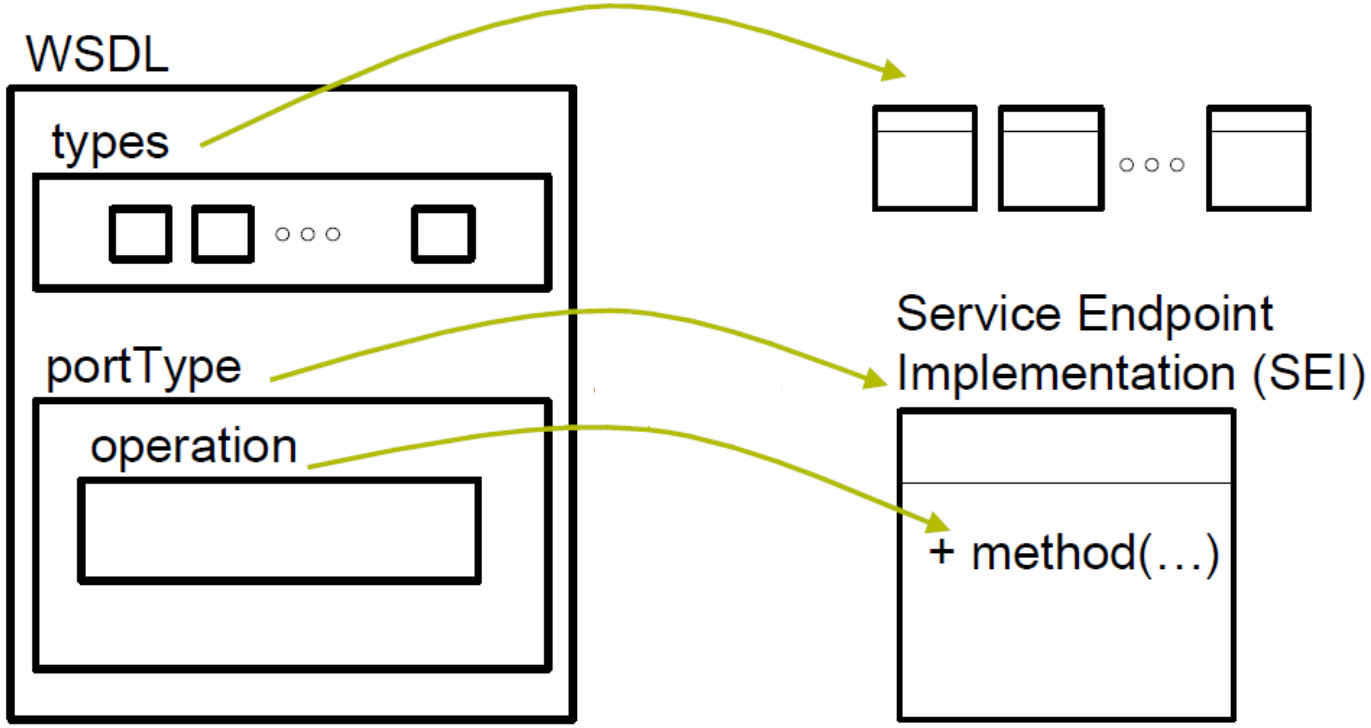
Procesamiento WSDL

❶ Los clientes de los servicios Web traducen los **types** a los modelos de datos del lenguaje de programación. **(TypesWSDL → Clases)**



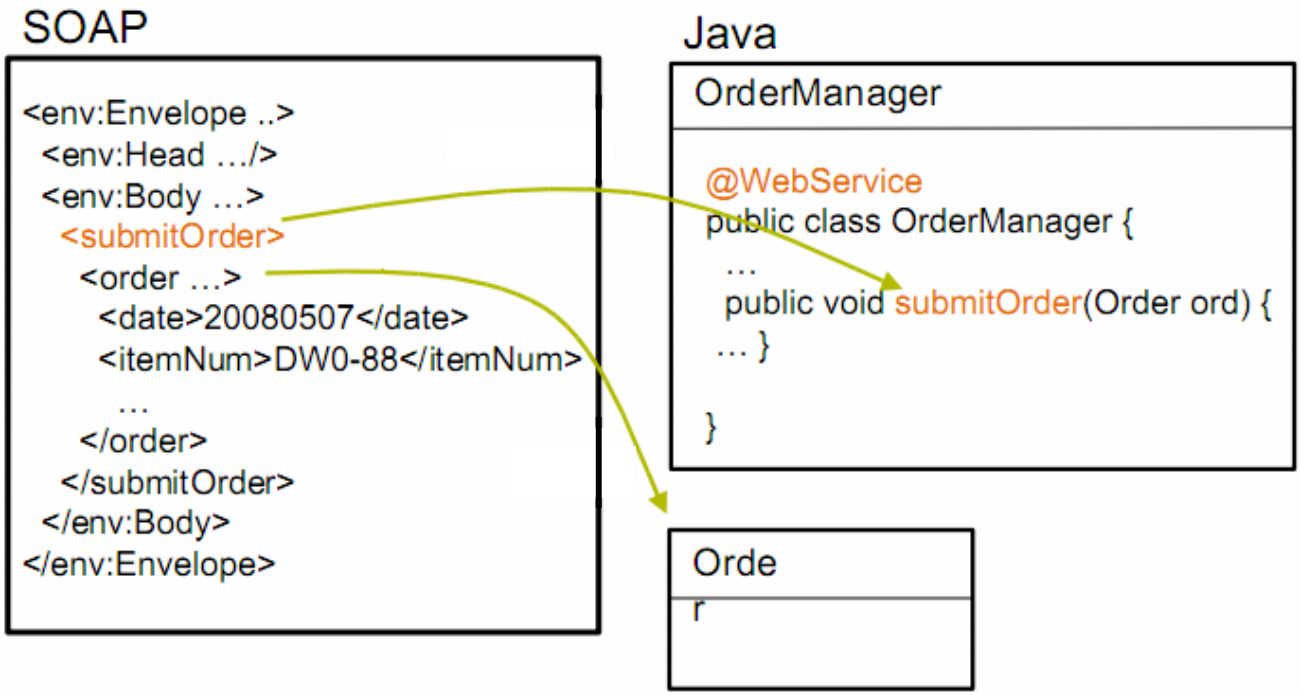
Procesamiento WSDL

❷ Los clientes de los servicios Web traducen las **wsdl:operations** a las funciones o métodos del lenguaje de programación.
(**Operations$_{WSDL}$ → Métodos de las clases**)



Procesamiento WSDL

❸ Los clientes de los servicios Web traducen las **soap:operations** a modelos de datos que contienen el código necesario para realizar las invocaciones de los servicios Web.
(**$Operations_{SOAP} \rightarrow$ Métodos de las clases**)



Procesamiento WSDL >> Tecnología J2EE

WSDL	Código Java	Tecnología
Types *Representación en XML Schema de los tipos de datos de las operaciones*	**Clases** *Representación de los tipos de datos de los parámetros de entrada y salida de las operaciones.*	**JAXB**
wsdl:operation *Interfaz de programación representado en XML de las operaciones.*	**Clases y métodos** *Creación de los métodos que se ejecutarán desde clientes Java.*	**JAX–WS**
soap:operation *Representación en XML Schema de la invocación de las operaciones de los servicios.*	**Clases y métodos** *Creación de los métodos que manejan el protocolo SOAP y que serán invocados desde los métodos asociados a las operaciones WSDL.*	**JAX–WS**