Departamento de Enxeñería Telemática
Universidade de Vigo

**24 de abril de 2013**

**ARQUITECTURAS Y SERVICIOS TELEMÁTICOS** **(Duración del examen: 2 horas)**

**NOMBRE:**

**Cuestión 1**. **(0.75 puntos)**

Indicar si las siguientes afirmaciones son verdaderas o falsas. Es imprescindible justificar adecuadamente el porqué de la decisión.

- *La cabecera SOAP incluye información para la vinculación del mensaje a un protocolo de transporte específico.*

Respuesta:

- *El uso de "mustUnderstand" en un bloque de cabecera SOAP, puede implicar la recepción posterior de un mensaje SOAP con un elemento "Fault" en su cabecera.*

Respuesta:

- *El modelo de procesado de un mensaje SOAP puede incluir intermediarios que reenvían el mensaje y/o realizan algún tipo de procesamiento intermedio.*

Respuesta:

**Cuestión 2**. **(0.25 puntos)**

¿Para qué se utiliza la etiqueta <correlationSets> en un documento BPEL?

Respuesta:

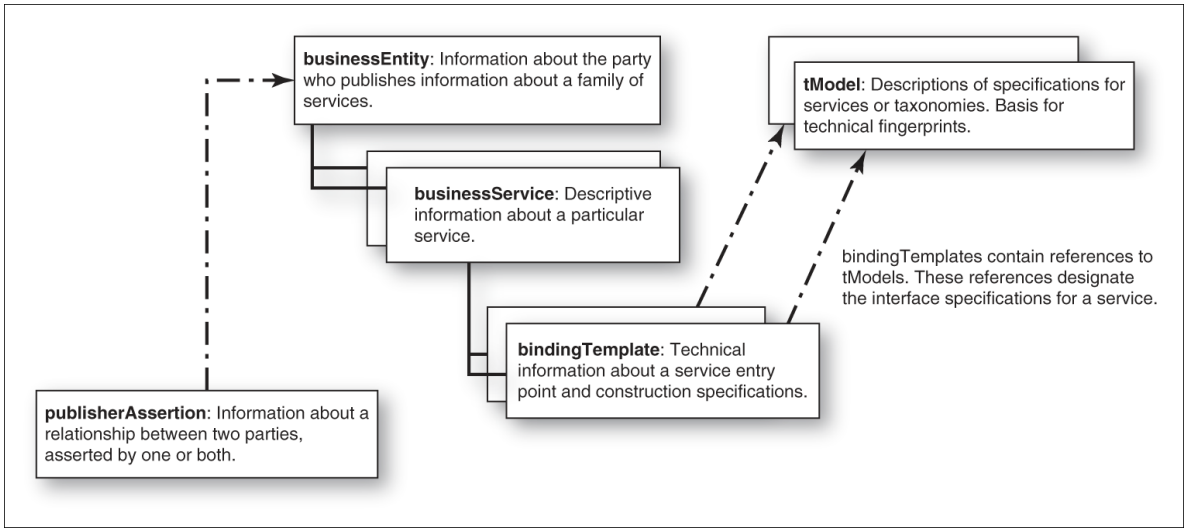
**Cuestión 3**. **(0.25 puntos)**

¿Cuáles de los siguientes elementos se podrían encontrar conjuntamente en un documento WSDL?

- ☐ import, types, ports
- ☐ import, portType, message
- ☐ port-binding, portType
- ☐ service, message

**Cuestión 4**. **(0.25 puntos)**

Describe el propósito de las estructuras de datos utilizadas en los registros UDDI mediante la analogía de páginas amarillas/blancas/verdes.




Respuesta:

**Problema 1. (0.75 puntos)**

Se dispone de la especificación parcial de un servicio Web:

```
<message name="getTermRequest">
   <part name="term" type="xsd:string"/>
</message>

<message name="getTermResponse">
   <part name="value" type="xsd:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
      <input message="tns:getTermRequest"/>
      <output message="tns:getTermResponse"/>
  </operation>
</portType>
```

Se desea extender el mensaje de solicitud con un nuevo parámetro con nombre `year`. La especificación del tipo de dato de este parámetro en XSD es la siguiente:

```
<xsd:schema>
   <xsd:simpleType name=yearType base="xsd:integer">
        <xsd:minInclusive value="2000"/>
        <xsd:maxInclusive value="2005"/>
   </xsd:simpleType>
</xsd:schema>
```

¿Cómo habría que modificar el documento WSDL facilitado?

Respuesta:

**Problema 2. (0.75 puntos)**

Se tienen 3 elementos participando en un servicio web: un servicio TravelAgent, un servicio Airline y un cliente. Se dispone del siguiente fragmento del documento BPEL que especifica la coordinación:

```
<flow>
     <links>
         <link name="order-to-airline"/>
         <link name="airline-to-agent"/>
      </links>
      <receive partnerLink="customer"
               portType="itineraryPT"
               operation="sendItinerary"
               variable="itinerary"
           <source linkName"order-to-airline"/>
      </receive>
      <invoke  partnerLink="airline"
               portType="ticketOrderPT"
               operation="requestTickets"
               inputVariable="itinerary"
           <target linkName"order-to-airline"/>
           <source linkName"airline-to-agent"/>
      </invoke>
      <receive  partnerLink="airline"
               portType="itineraryPT"
               operation="sendTickets"
               variable="tickets"
           <target linkName"airline-to-agent"/>
      </receive>
   </flow>
```

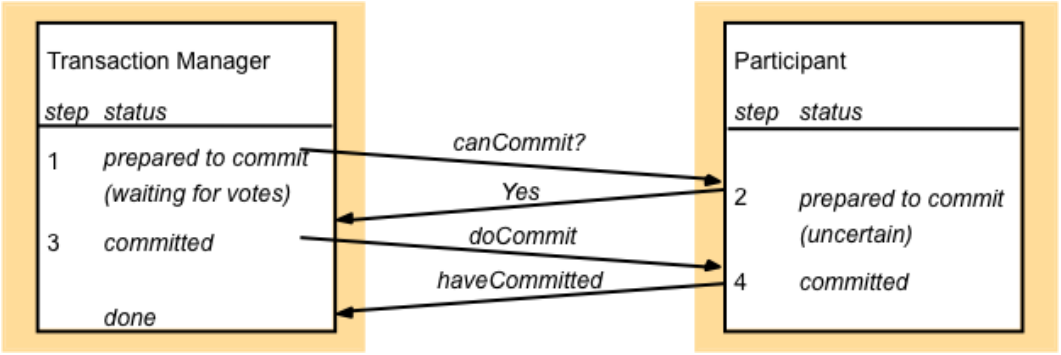
¿Qué servicio proporciona cada una de las funcionalidades expresamente indicadas en el fragmento anterior? ¿En qué `portType` están especificadas? Representa en un diagrama de secuencia las actividades descritas.

Respuesta:

## Problema 3. (0.75 puntos)

Supóngase un protocolo *Two Phase Commit* donde el coordinador (**C**) de la transacción utiliza las siguientes operaciones en su comunicación con los participantes (**P**) en una transacción con identificador **IdT**:

- ***C**: canCommit?(**IdT**) → **P**: Yes / No (repuesta de P)*
- ***C**: doCommit(**IdT**) → **P**: haveComitted (respuesta de P)*
- ***C**: doAbort(**IdT**) -> **P**: haveAborted (respuesta de P)*
- ***P**: getDecision**(IdT)***




Además, el coordinador utiliza para la recuperación los siguientes ficheros de log:

- Fichero "*Start-Two-Phase-Commit: IdT*", con una lista de participantes en la transacción, que se crea previamente al inicio de la fase de votación.
- Fichero "*Commit: IdT*", fichero creado cuando el coordinador toma la decisión de comprometer la transacción, previo a la segunda fase.
- Fichero "Abort: IdT", fichero creado cuando el coordinador toma la decisión de abortar la transacción, previo a la segunda fase.
- Fichero "*Done: IdT*", la escritura de este fichero implica que la transacción ha finalizado, habiéndose recibido todos los "*haveComitted"* de los participantes.

Estos ficheros sólo son eliminados por un proceso especial de terminación que cae fuera del alcance de este problema.

**SE PIDE**, explicar el comportamiento del coordinador, respecto a la transacción ***IdT***, después de una caída temporal del coordinador.

Respuesta:

**Problema 4. (1.25 puntos)**

Codificar un cliente java (clase `cliente.java`) que realice una llamada síncrona y una llamada asíncrona a la operación `sayHello` del servicio `Hello_Service` (localizado en `http://localhost:9090/Hello_Service`). Se detallan como adjuntos, además de la descripción del servicio `Hello_Service.wsdl`, las partes más relevantes de las clases e interfaces generadas por la herramienta de generación de código de Axis2 (ver apéndice).

Respuesta:

# Apéndices (resolución del Problema 4)

## Hello_Service.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
   targetNamespace="http://examen/HelloService.wsdl"
   xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns="http://examen/HelloService.wsdl" xmlns:xsd="http://www.w3.org/2001/XMLSchema">

   <message name="SayHelloRequest">
      <part name="firstName" type="xsd:string"/>
   </message>
   <message name="SayHelloResponse">
      <part name="greeting" type="xsd:string"/>
   </message>

   <portType name="Hello_PortType">
      <operation name="sayHello">
         <input message="tns:SayHelloRequest"/>
         <output message="tns:SayHelloResponse"/>
      </operation>
   </portType>

   <binding name="Hello_Binding" type="tns:Hello_PortType">
      <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="sayHello">
         <soap:operation soapAction="sayHello"/>
         <input>
            <soap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examen:helloservice" use="encoded"/>
         </input>
         <output>
            <soap:body  encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examen:helloservice" use="encoded"/>
         </output>
      </operation>
   </binding>

   <service name="Hello_Service">
      <documentation>WSDL File for HelloService</documentation>
      <port binding="tns:Hello_Binding" name="Hello_Port">
         <soap:address location="http://localhost:8080/Hello_Service"/>
      </port>
   </service>
</definitions>
```

## Interface Hello_Service

```
/**
 * Hello_Service.java
 * This file was auto-generated from WSDL
 * by the Apache Axis2 version: 1.6.2  Built on : Apr 17, 2012 (05:33:49 IST)
 */
package examen.helloservice_wsdl;
/* Hello_Service java interface */
public interface Hello_Service {

/* Auto generated method signature */
public helloservice.examen.SayHelloResponse sayHello(helloservice.examen.SayHello sayHello0)
throws java.rmi.RemoteException;

/* Auto generated method signature for Asynchronous Invocations */
public void startsayHello(helloservice.examen.SayHello sayHello0, final
examen.helloservice_wsdl.Hello_ServiceCallbackHandler callback) throws
java.rmi.RemoteException;

}
```

## Clase SayHello

```
/**
 * SayHello.java
 *
 * This file was auto-generated from WSDL
 * by the Apache Axis2 version: 1.6.2  Built on : Apr 17, 2012 (05:34:40 IST)
 */
package helloservice.examen;
```

```
/* SayHello bean class */
@SuppressWarnings({"unchecked","unused"})

public  class SayHello implements org.apache.axis2.databinding.ADBBean{

public static final javax.xml.namespace.QName MY_QNAME = new javax.xml.namespace.QName(
                "urn:examen:helloservice",
                "sayHello",
                "ns1");

/* field for FirstName */
protected java.lang.String localFirstName ;


/* Auto generated getter method   */
public  java.lang.String getFirstName(){return localFirstName;}


/* Auto generated setter method */
public void setFirstName(java.lang.String param){this.localFirstName=param;}

/**
* @param parentQName
* @param factory
* @return org.apache.axiom.om.OMElement
*/
public org.apache.axiom.om.OMElement getOMElement (. . .

public void serialize(. . .

public void serialize(. . .

private static java.lang.String generatePrefix( …) {...}

/* Utility methods */
private void writeStartElement( . . .
private void writeAttribute(. . .
private void writeAttribute(. . .
private void writeQNameAttribute(. . .
private void writeQName(. . .
private void writeQNames(. . .
private java.lang.String registerPrefix(. . .

/* databinding method to get an XML representation of this object */
public javax.xml.stream.XMLStreamReader getPullParser(. . .

/* Factory class that keeps the parse method */
public static class Factory{
        /* static method to create the object */
        public static SayHello parse(. . .
} }
```

## Clase SayHelloResponse

```
/**
 * SayHelloResponse.java
 * This file was auto-generated from WSDL
 * by the Apache Axis2 version: 1.6.2  Built on : Apr 17, 2012 (05:34:40 IST)
 */
package helloservice.examen;

/* SayHelloResponse bean class */
@SuppressWarnings({"unchecked","unused"})

public  class SayHelloResponse implements org.apache.axis2.databinding.ADBBean{

public static final javax.xml.namespace.QName MY_QNAME = new
javax.xml.namespace.QName("urn:examen:helloservice", "sayHelloResponse", "ns1");

/* field for Greeting*/
protected java.lang.String localGreeting ;

/* Auto generated getter method  @return java.lang.String */
public  java.lang.String getGreeting(){
                  return localGreeting;}

/* Auto generated setter method @param param Greeting */
public void setGreeting(java.lang.String param){
           this.localGreeting=param;}

public org.apache.axiom.om.OMElement getOMElement ( . . .

private static java.lang.String generatePrefix( . . .

/* Utility method to write an element start tag.*/
private void writeStartElement( . . .

 /* Util method to write an attribute with the ns prefix */
private void writeAttribute( . . .

/* Util method to write an attribute without the ns prefix  */
private void writeAttribute( . . .

/* Util method to write an attribute without the ns prefix */
private void writeQNameAttribute( . . .

/* method to handle Qnames */
private void writeQName( . . .
private void writeQNames( . . .

/* Register a namespace prefix */
private java.lang.String registerPrefix( . . .

/* databinding method to get an XML representation of this object */
public javax.xml.stream.XMLStreamReader getPullParser( . . .

/*  Factory class that keeps the parse method  */
public static class Factory{
   }//end of factory class
}
```

## Clase Hello_ServiceStub

```
/* Hello_ServiceStub.java
 * This file was auto-generated from WSDL
 * by the Apache Axis2 version: 1.6.2  Built on : Apr 17, 2012 (05:33:49 IST)
 */
package examen.helloservice_wsdl;

/* Hello_ServiceStub java implementation */
public class Hello_ServiceStub extends org.apache.axis2.client.Stub
implements Hello_Service{

protected org.apache.axis2.description.AxisOperation[] _operations;
//hashmaps to keep the fault mapping
private java.util.HashMap faultExceptionNameMap = new java.util.HashMap();
private java.util.HashMap faultExceptionClassNameMap = new java.util.HashMap();
private java.util.HashMap faultMessageMap = new java.util.HashMap();

private static int counter = 0;
private static synchronized java.lang.String getUniqueSuffix(){ . . .}

//creating the Service with a unique name
private void populateAxisService() throws org.apache.axis2.AxisFault { . . . }

//populates the faults
private void populateFaults(){ ...}

/* Constructor that takes in a configContext */
public Hello_ServiceStub(org.apache.axis2.context.ConfigurationContext configurationContext,
java.lang.String targetEndpoint) throws org.apache.axis2.AxisFault { . . .}

/* Constructor that takes in a configContext  and useseperate listener */
public Hello_ServiceStub(org.apache.axis2.context.ConfigurationContext configurationContext,
java.lang.String targetEndpoint, boolean useSeparateListener) throws
org.apache.axis2.AxisFault { . . .}

/* Default Constructor */
public Hello_ServiceStub(org.apache.axis2.context.ConfigurationContext configurationContext)
throws org.apache.axis2.AxisFault {
         this(configurationContext,"http://localhost:8080/soap/servlet/rpcrouter" );}

/* Default Constructor */
public Hello_ServiceStub() throws org.apache.axis2.AxisFault {
         this("http://localhost:8080/soap/servlet/rpcrouter" );}

/* Constructor taking the target endpoint */
public Hello_ServiceStub(java.lang.String targetEndpoint)
throws org.apache.axis2.AxisFault {this(null,targetEndpoint);}

/* Auto generated method signature */
public  helloservice.examen.SayHelloResponse sayHello(helloservice.examen.SayHello sayHello2)
throws java.rmi.RemoteException  {
      // sayHello implementaion (Synchronous)
}

/* Auto generated method signature for Asynchronous Invocations */
public  void startsayHello(helloservice.examen.SayHello sayHello2, final
examen.helloservice_wsdl.Hello_ServiceCallbackHandler callback) throws
java.rmi.RemoteException{
      // sayHello implementaion (Asynchronous)
}

/*  Utility Methods */
private java.util.Map getEnvelopeNamespaces(. . .
private javax.xml.namespace.QName[] opNameArray = null;
private boolean optimizeContent( . . .
private  org.apache.axiom.om.OMElement  toOM( . . .
private  org.apache.axiom.soap.SOAPEnvelope toEnvelope( . .
}
```

## Clase Hello_ServiceCallbackHandler

```
/**
* Hello_ServiceCallbackHandler.java
* This file was auto-generated from WSDL
* by the Apache Axis2 version: 1.6.2  Built on : Apr 17, 2012 (05:33:49 IST)
*/

package examen.helloservice_wsdl;
```

```
/* Hello_ServiceCallbackHandler Callback class, Users can extend this class and implement
their own receiveResult and receiveError methods. */
public abstract class Hello_ServiceCallbackHandler{

protected Object clientData;

/*  User can pass in any object that needs to be accessed once the NonBlocking
* Web service call is finished and appropriate method of this CallBack is called.
* @param clientData Object mechanism by which the user can pass in user data
* that will be available at the time this callback is called. */
public Hello_ServiceCallbackHandler(Object clientData){
  this.clientData = clientData;}

/* Please use this constructor if you don't want to set any clientData  */
public Hello_ServiceCallbackHandler(){
        this.clientData = null; }

/* Get the client data */
public Object getClientData() { return clientData;}

/* auto generated Axis2 call back method for sayHello method override this method for handling
normal response from sayHello operation */
public void receiveResultsayHello( helloservice.examen.SayHelloResponse result) {
}

/* auto generated Axis2 Error handler override this method for handling error response from
sayHello operation */
public void receiveErrorsayHello(java.lang.Exception e) { }

}
```