Programación Orientada a Servicios

Servicios Web Semánticos

Programa de
Ingeniería en Computación
UAM – Azcapotzalco

A cargo de:
Dra. Maricela Claudia Bravo Contreras
mcbc@correo.azc.uam.mx

CBI DIVISIÓN CIENCIAS BÁSICAS E INGENIERÍA
UAM - Azcapotzalco

Casa abierta al tiempo
Universidad Autónoma Metropolitana
Azcapotzalco

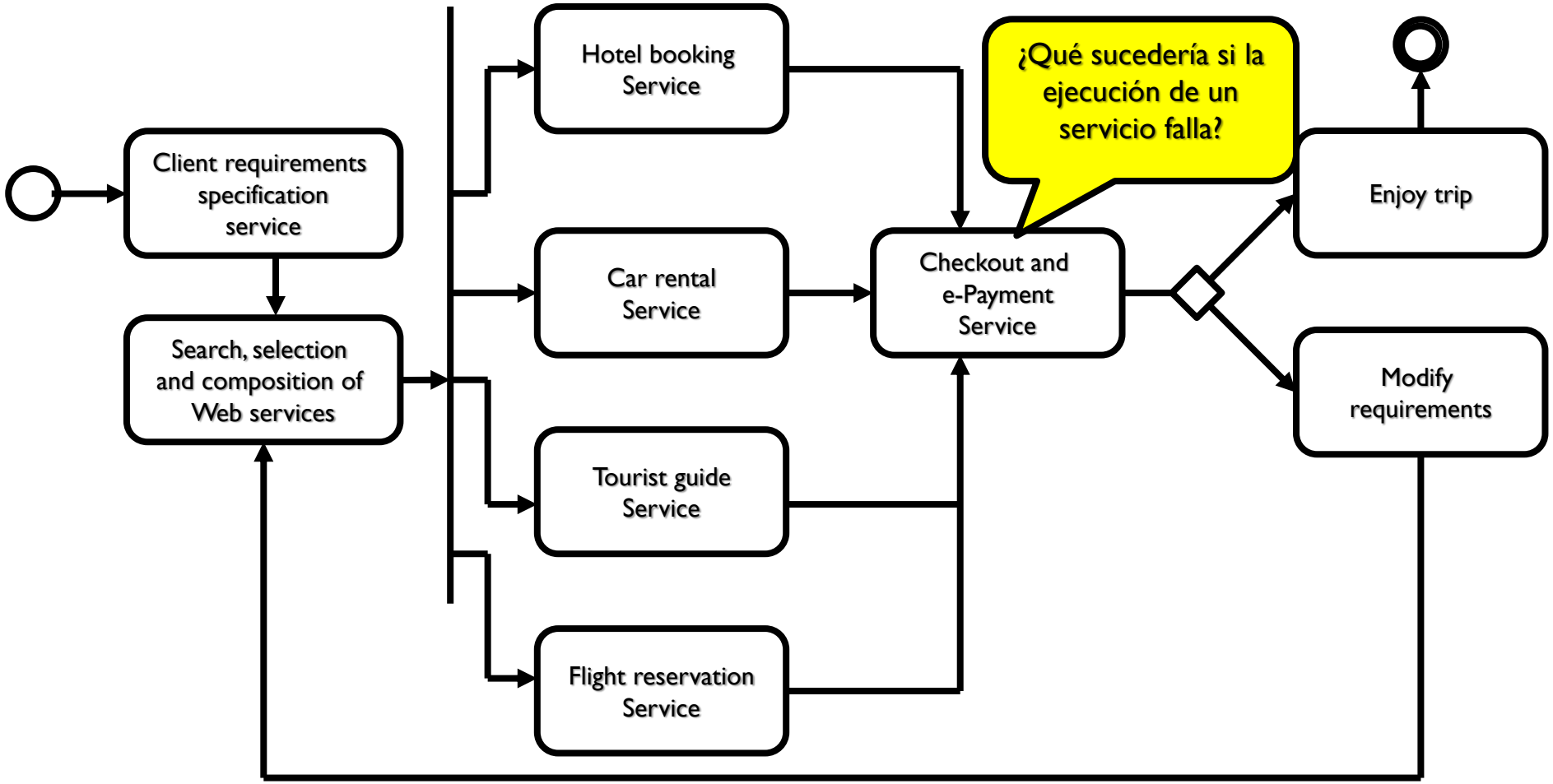
Introducción

- Internet está poblado con innumerables recursos de software reutilizables (servicios Web).
- La búsqueda y utilización de servicios Web públicos son tareas que no han sido posible automatizar debido a que las descripciones de los servicios Web
  - Carecen de semántica funcional bien definida
  - No contienen ejemplares de invocación
  - La mayoría de ellos no cuenta con documentación.

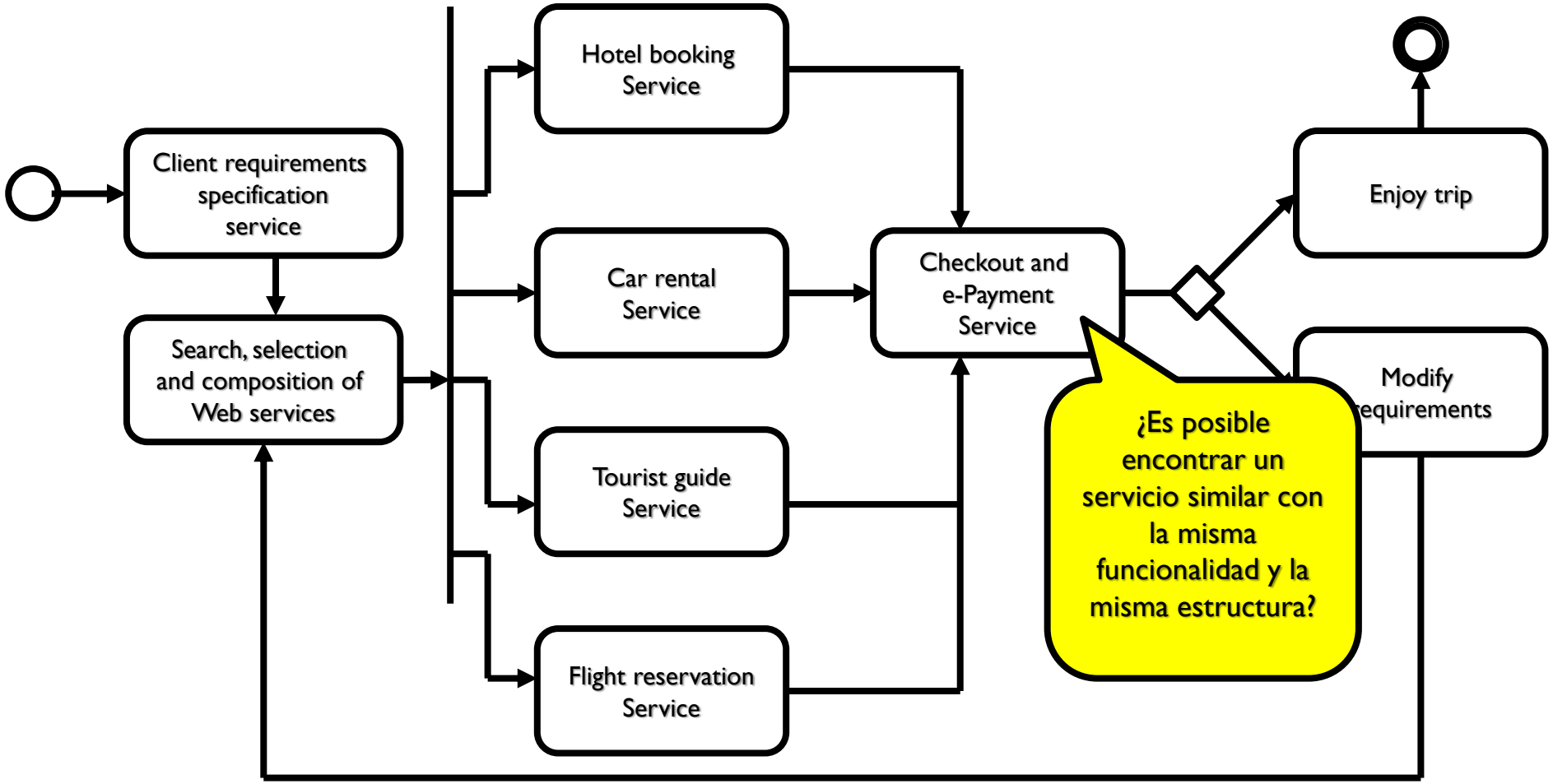
¿Qué es un servicio Web?

- Componente de software distribuido que ofrece alguna funcionalidad específica:
  - Reutilizable
  - Descubrible
  - Independiente del lenguaje y la palataforma en el que fue construido.

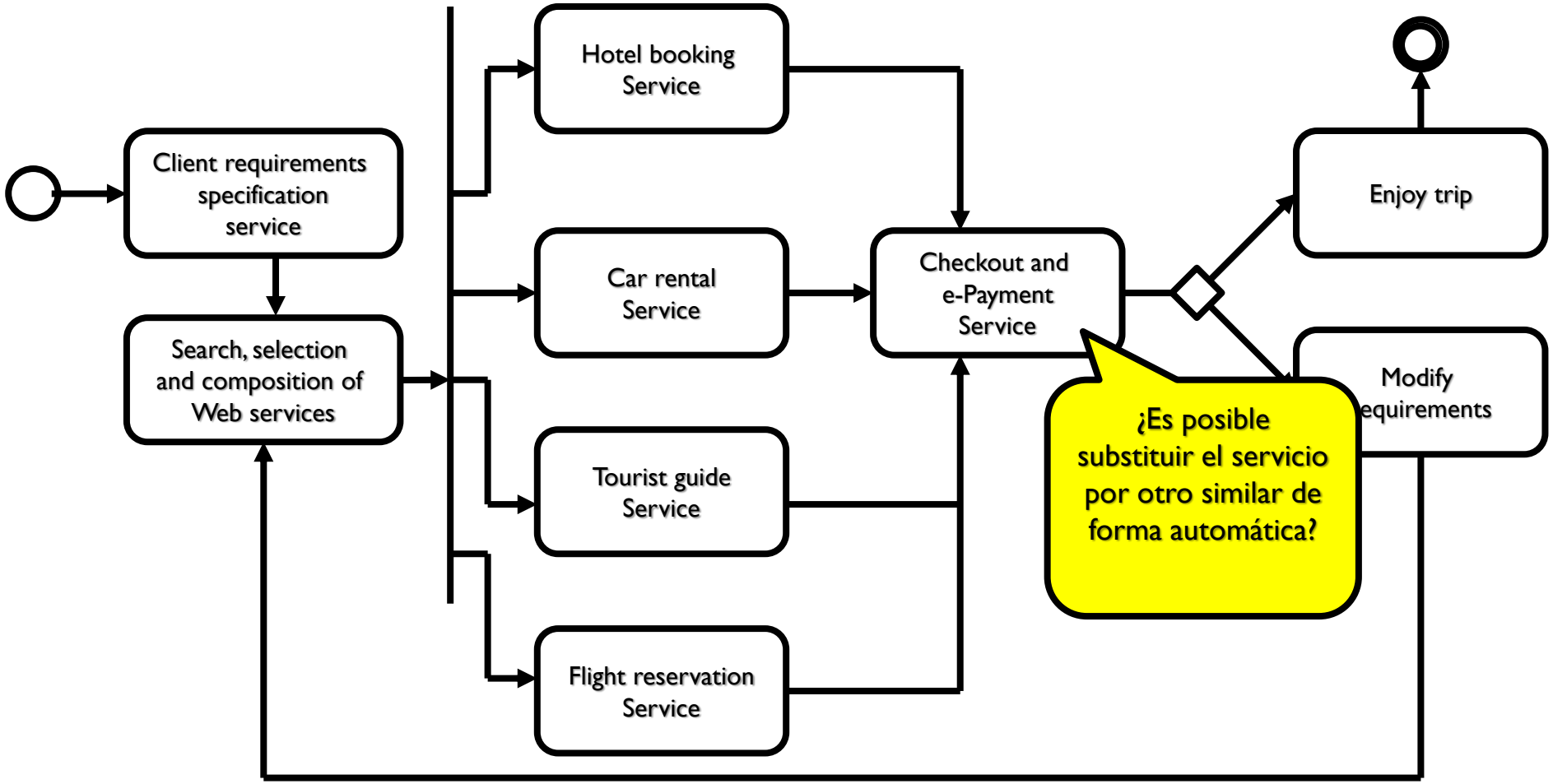
Motivación: Composición de Servicios Web



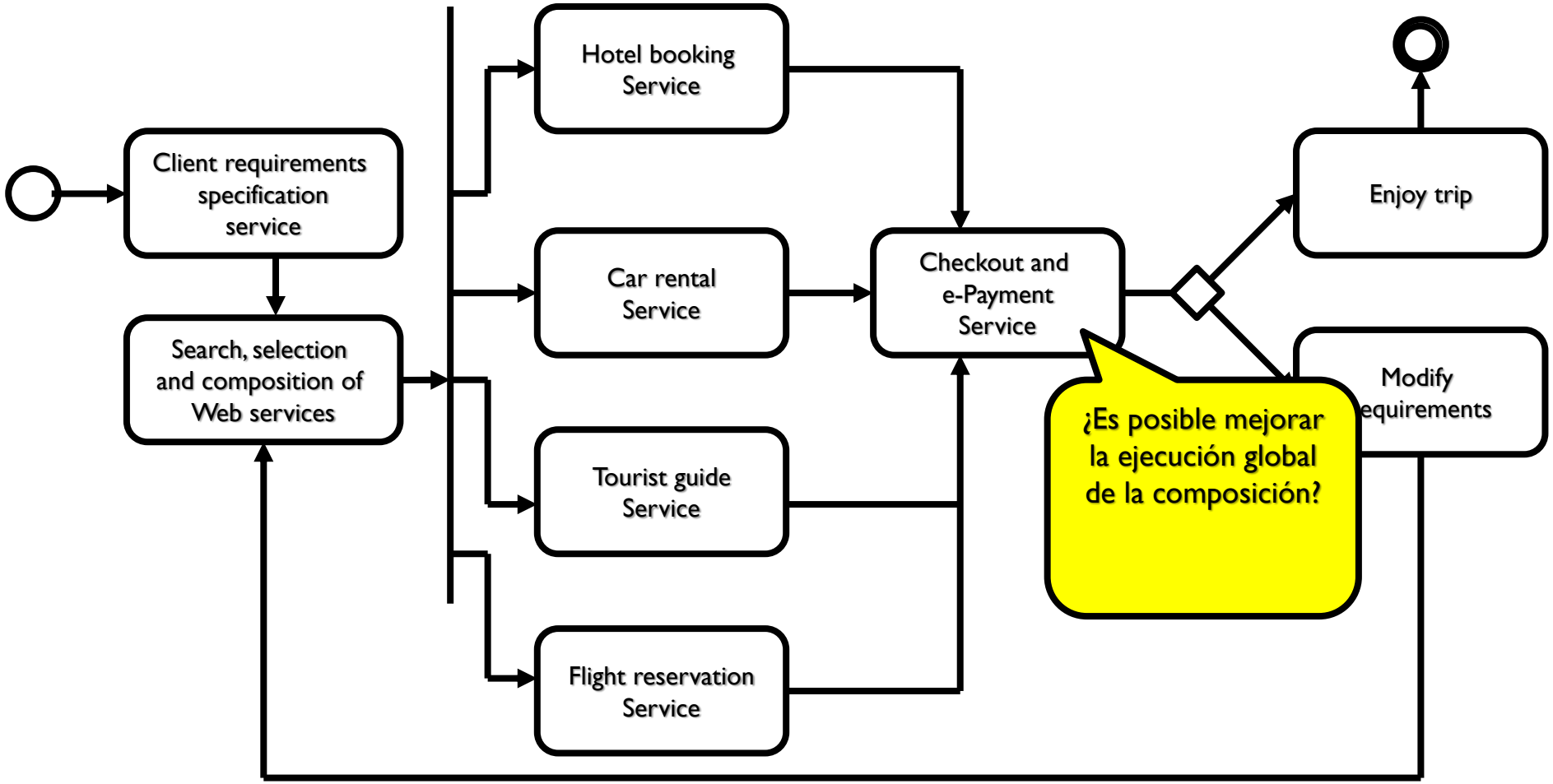
Motivación: Descubrimiento y Selección



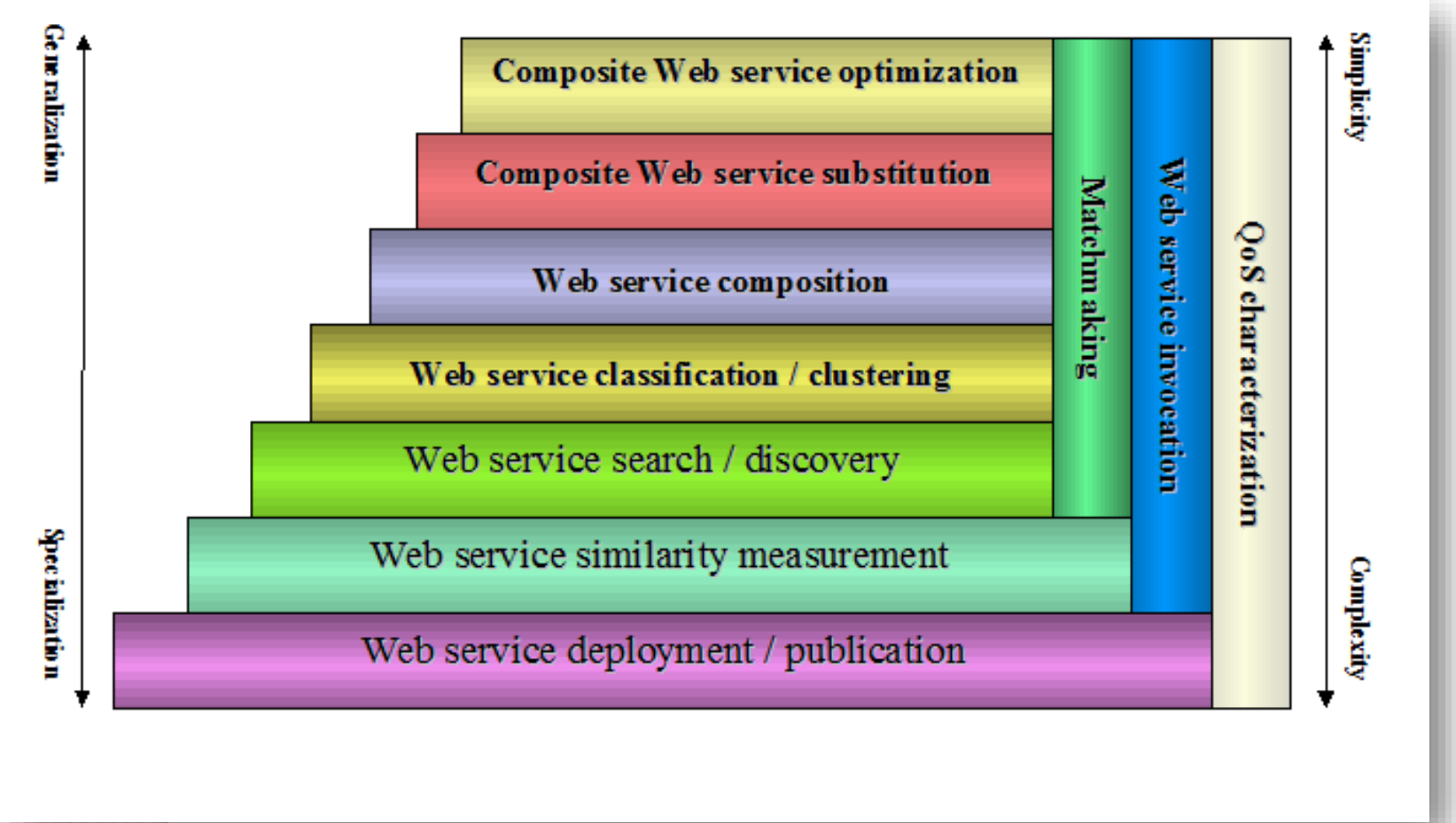
Motivación: Substitución



Motivación: Optimización



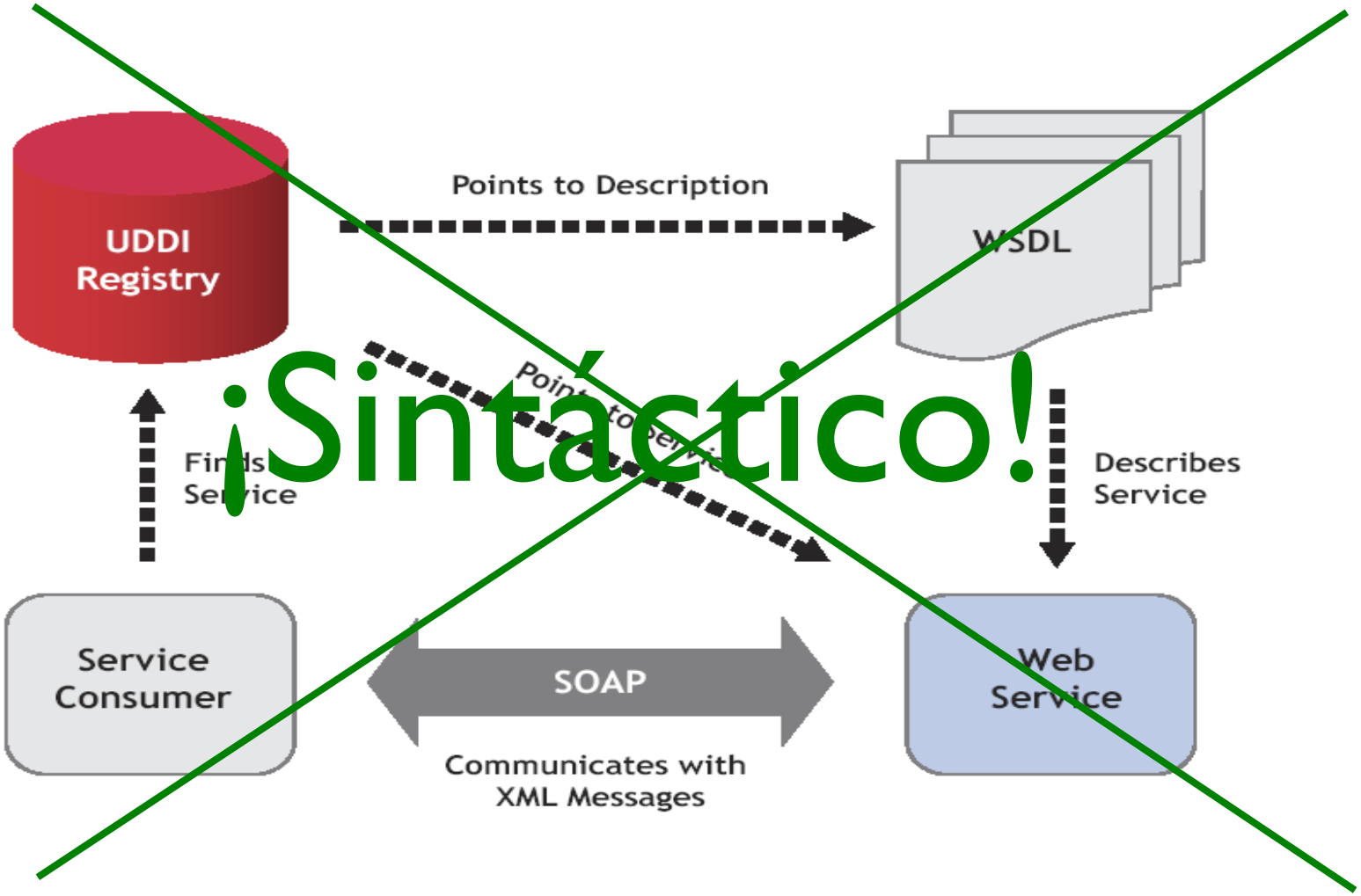
TEMAS REALCIONADOS CON LOS SERVICIOS WEB



Retos

- Retos de la Programación Orientada a Servicios (POS).
- POS apoya la interacción orientada a servicios ofreciendo las siguientes funciones:
  - Programación de servicios
  - Despliegue de servicios
  - Publicación de servicios
  - Descubrimiento de servicios
  - Acceso a servicios en tiempo de ejecución.

Estándares de WS carecen de semántica bien definida



Problema: los estándares de WS posibilitan la interoperabilidad, pero aun el descubrimiento y la composición tienen una base sintáctica, no hay forma de describir la funcionalidad.

PRINCIPALES TECNOLOGÍAS

1. OWL-S
2. WSMO
3. SAWSDL
4. BPEL

OWL-S

Ontología OWL-S

- OWL-S es una ontología en OWL para describir servicios Web
- OWL-S se basa en OWL para
  - Soportar el descubrimiento de capacidades de SW
  - Soportar la composición automatizada de SW
  - Soportar la invocación automatizada de SW

"Completar no competir"

  - OWL-S no trata de reemplazar los estándares de los servicios Web, más bien intenta proporcionar una capa semántica.
  - OWL-S depende de WSDL para la invocación de servicios Web (Grounding)
  - OWL-S extiende UDDI para el descubrimiento de servicios Web (OWL-S/UDDI mapping)

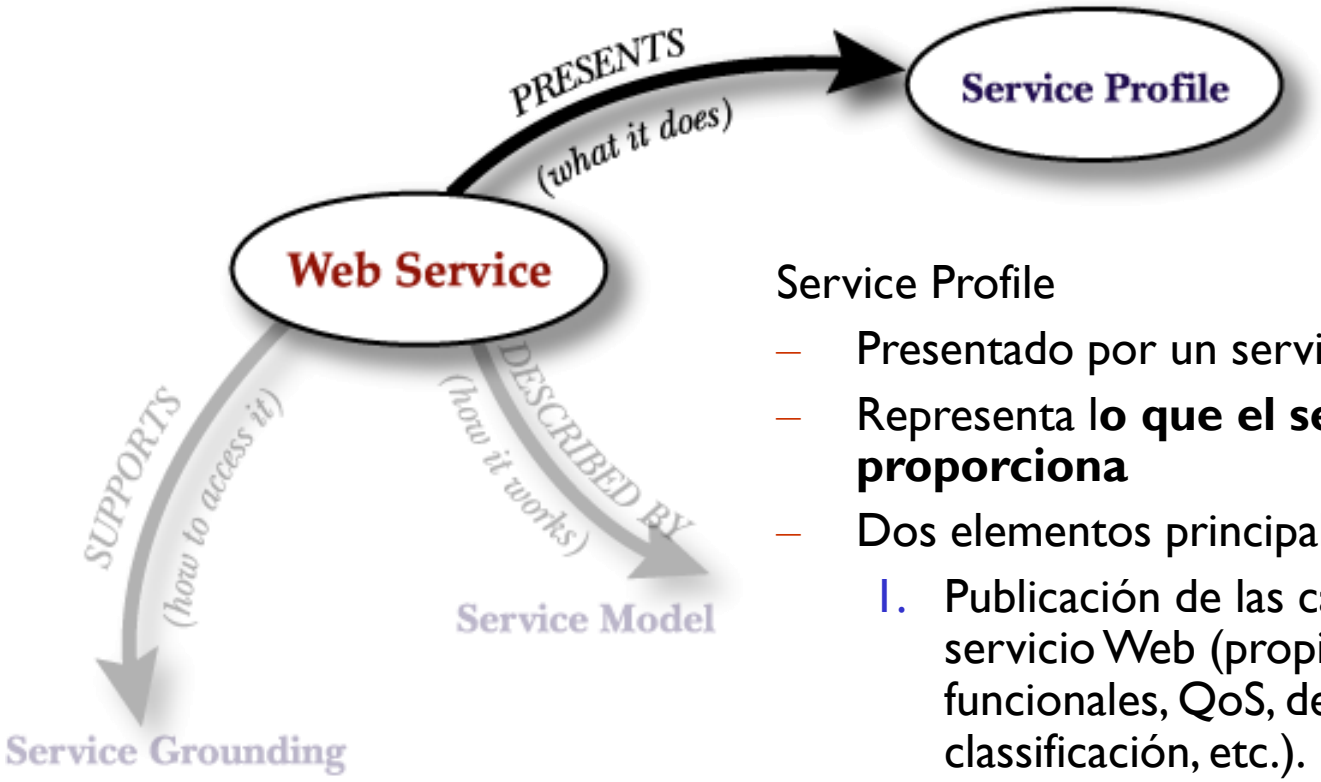
OWL-S Upper Ontology

- Especificación de capacidades
- Características generales
- Calidad del Servicio
- Clasificación en taxonomías

- Mapeo con WSDL
  - protocolo de comunicación (RPC, HTTP, …)
  - marshalling/serialization
  - transformación a y desde XSD a OWL

- Control del flujo del servicio
  - Black/Grey/Glass Box view
- Especificación del protocolo
- Mensajes abstractos

Service Profile



Service Profile

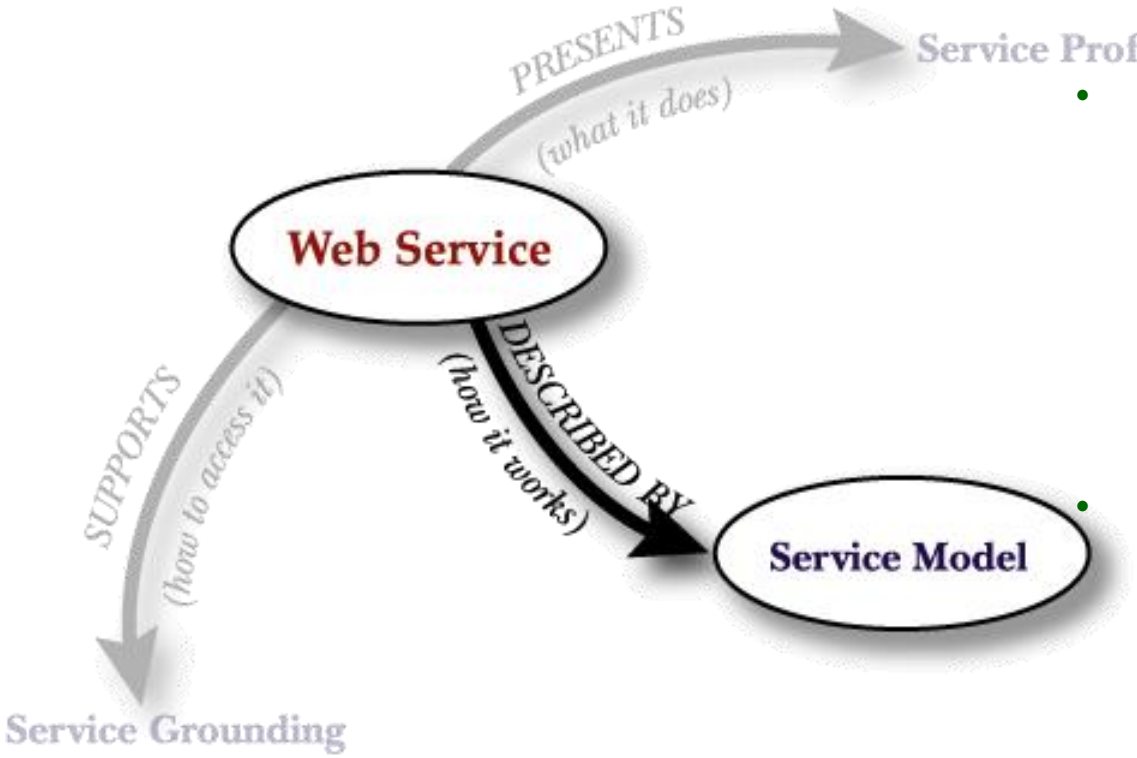
- Presentado por un servicio
- Representa l**o que el servicio proporciona**
- Dos elementos principales:
  1. Publicación de las capacidades del servicio Web (propiedades no funcionales, QoS, descripción, classificación, etc.).
  2. Solicitud de servicios Web con un conjunto dado de capacidades.

•El service Profile no especifica uso/invocación!

OWL-S Service Profile
Descripción de Capacidades

- **Preconditions**
  - Conjunto de condiciones que deben satisfacerse anets de la invocación del servicio.
- **Inputs**
  - Conjunto de inputs necesarios que el solicitante debe proporcionar para invocar el servicio.
- **Outputs**
  - Resultados que el solicitante debe esperar después de que la interacción con el proveedor del servicio esté completada.
- **Effects**
  - Conjunto de estados que deben ser verdaderos si el servicio fue invocado satisfactoriamente.
- **Service type**
  - Que tipo de servicio es proporcionado (venta, distribución, información, etc.).
- **Product**
  - Producto asociado con el servicio (viaje, libros, auto partes, etc.).

Process Model

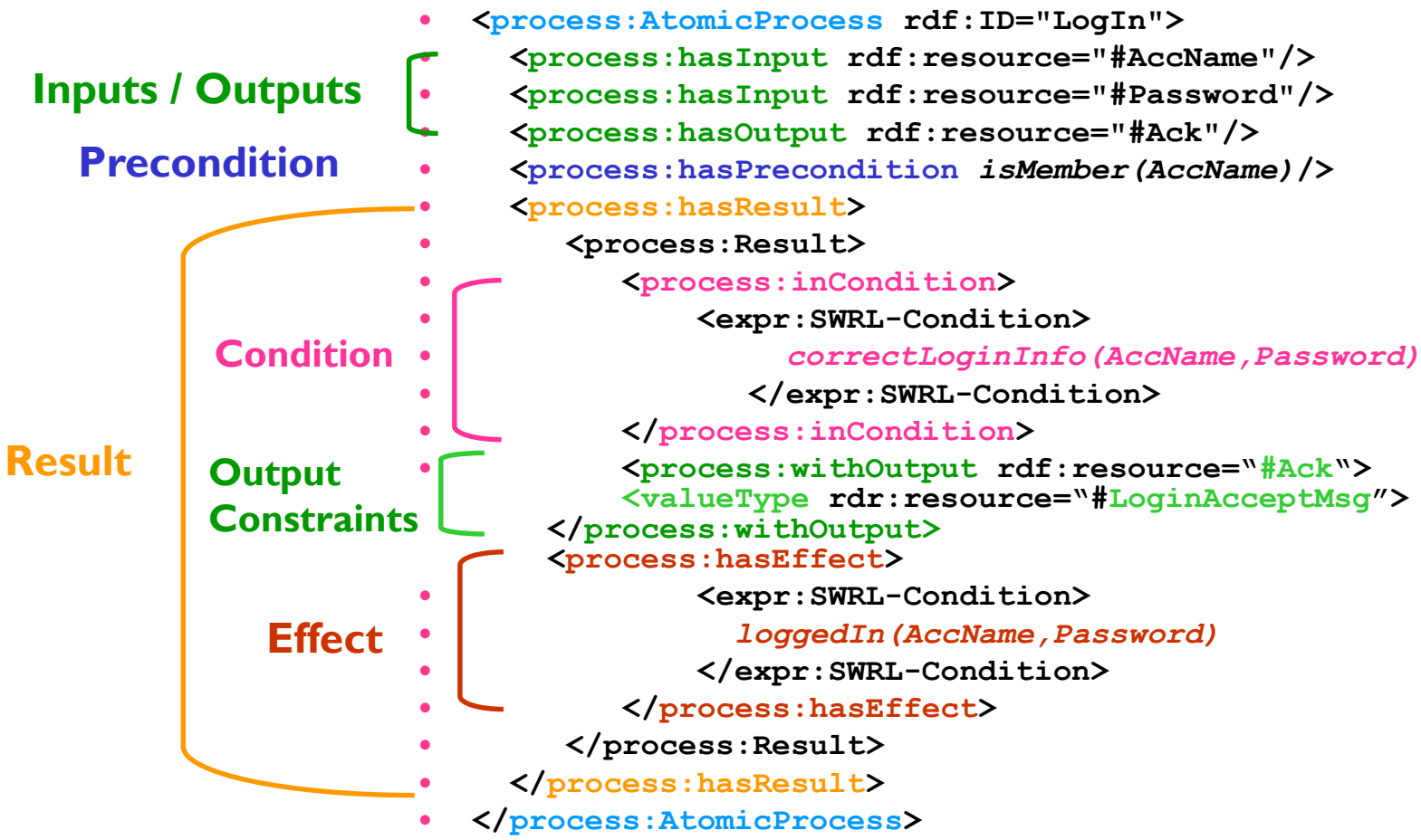


- Process Model
  - Describe cómo trabaja el servicio: procesos internos del servicio.
  - Especifica el protocolo de interacción del servicio.
  - Especifica los mensajes abstractos: tipo de ontología de la información transmitida.
- Facilita
  - Invocación del servicio Web
  - Composición de servicios Web
  - Monitoreo de la interacción.

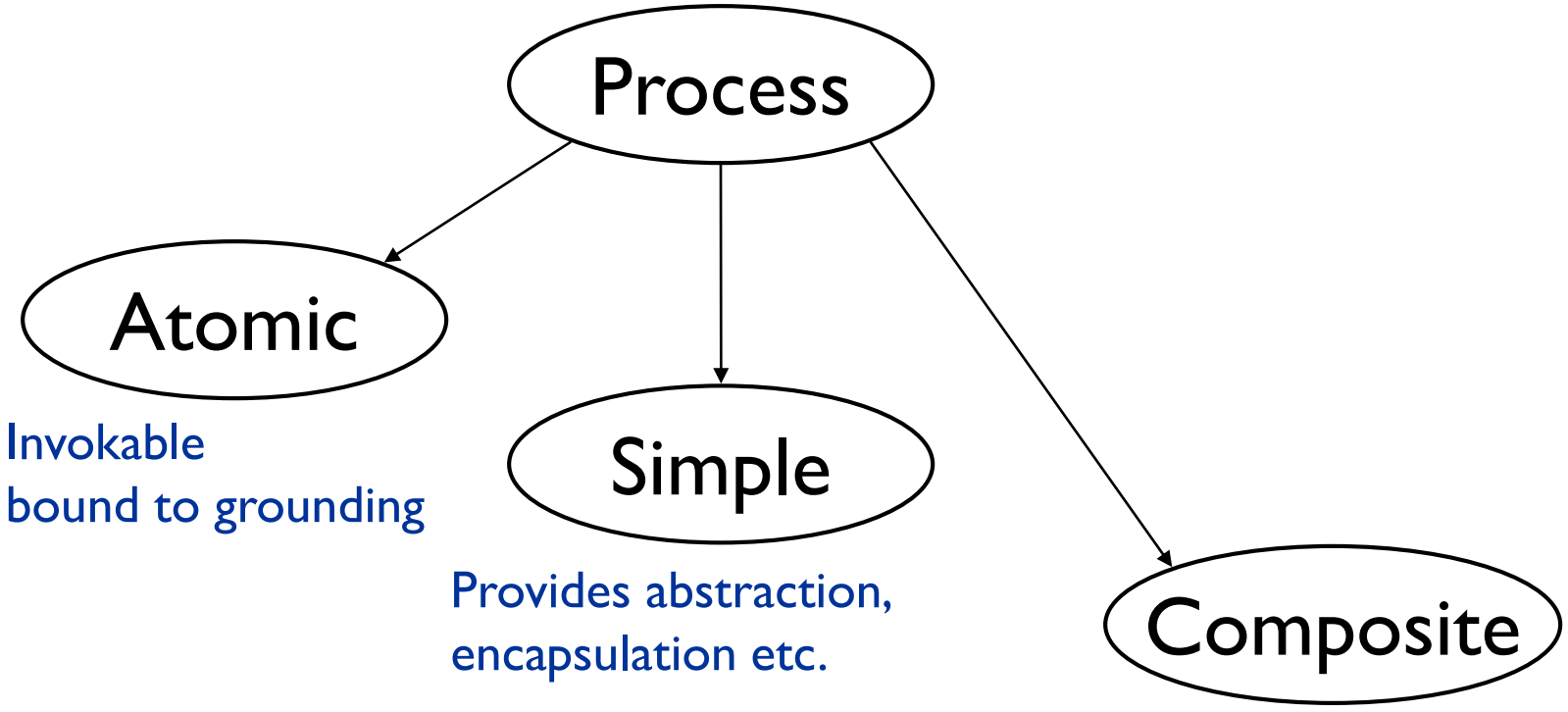
Definición del Proceso

- Un proceso representa una función de transformación
- Es caracterizado por cuatro parámetros:
  - ***Inputs***: las entradas que requiere el proceso.
  - ***Preconditions***: las condiciones que se requieren para que el proceso se ejecute correctamente.
  - ***Outputs***: la información que resulta (y es devuelta desde) la ejecución del proceso.
  - ***Results***: un proceso puede tener diferentes salidas dependiendo de alguna condición
    - ***Condition***: bajo que condición sucede el resultado
    - ***Constraints on Outputs***
    - ***Effects***: cambios en el mundo real resultantes de la ejecuión del proceso.

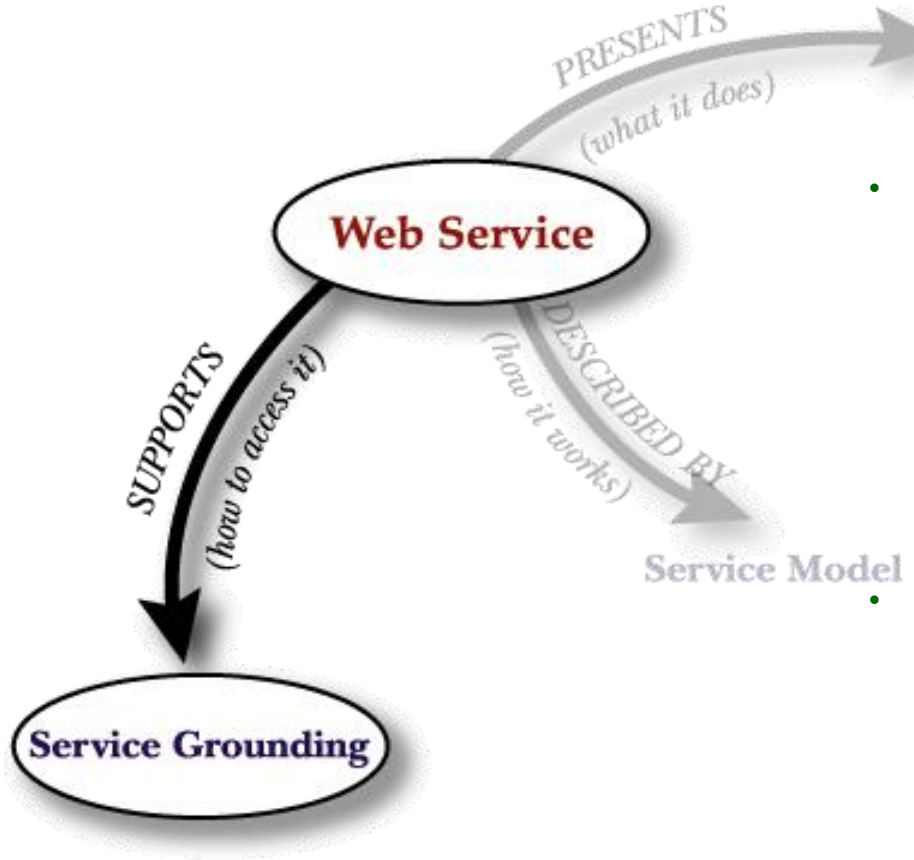
Ejemplo de un Proceso atómico



Ontología de Procesos



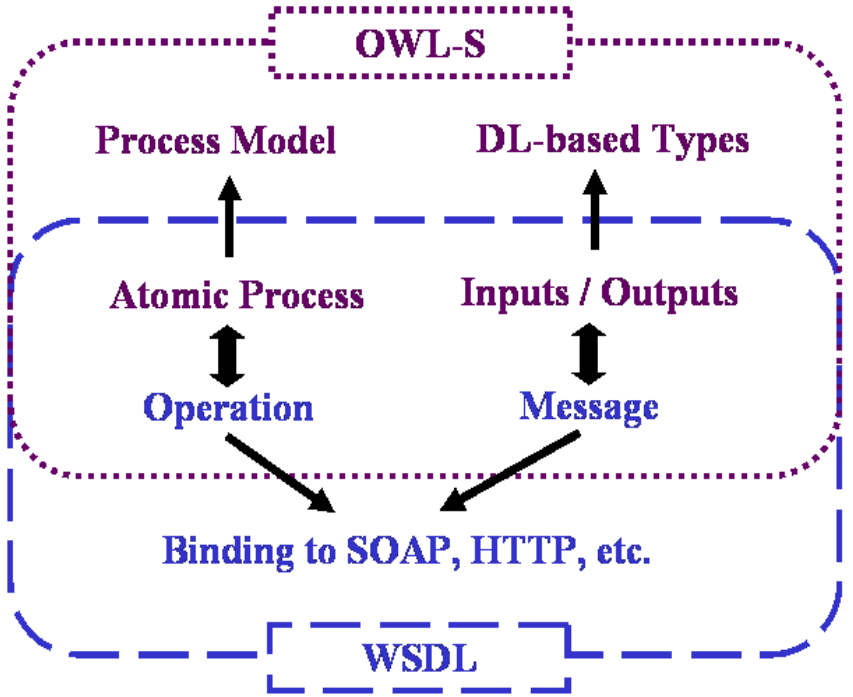
Service Grounding



- Service Grounding
  - Proporciona la especificación de acceso a la información del servicio.
  - Service Model + Grounding ofrecen todo lo necesario para utilizar el servicio.
  - Construido sobre **WSDL** para definir la estructura del mensaje y la capa de enlace físico.
- Especifica:
  - Protocolos de comunicación, mecanismos de transporte, lenguajes de comunicación, etc.

Correspondencia entre OWL-S y WSDL 1.1

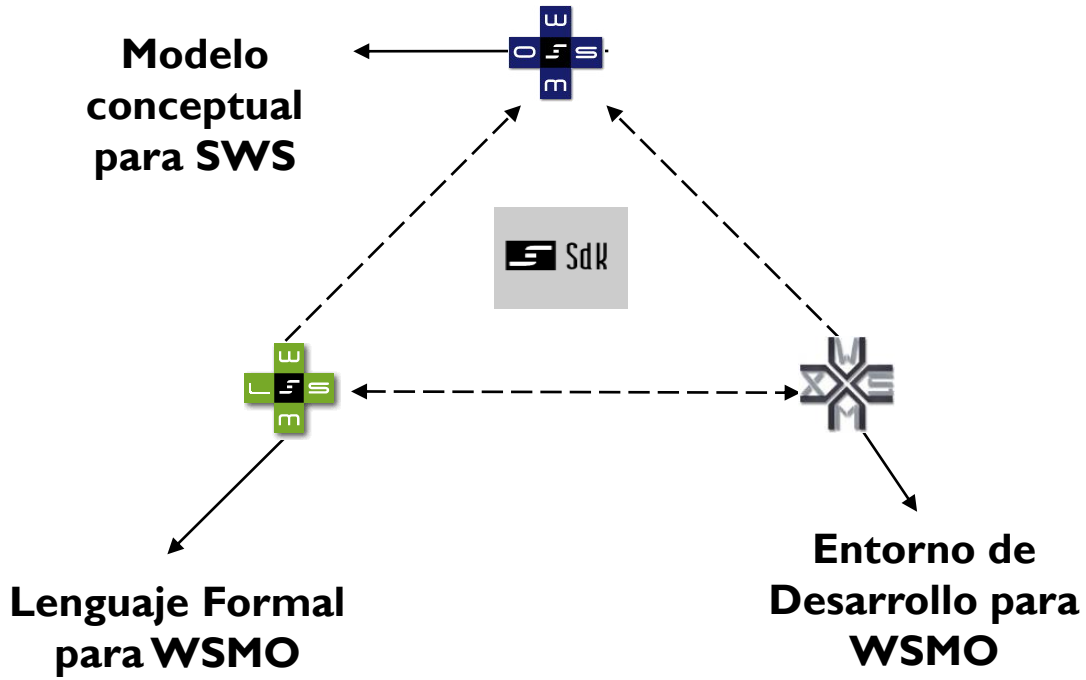
- **Operaciones** corresponden con Procesos atómicos
- **Mensajes de Input/Output** corresponden con los Inputs/Outputs de procesos.



WSMO

WSMO

- WSMO es una ontología y modelo conceptual para describir servicios Web y aspectos relacionados.
- Esta basado en el Framework para el Modelado de Servicios Web (WSMF).
- WSMO es un proyecto del grupo de trabajo de WSMO de ESSI.

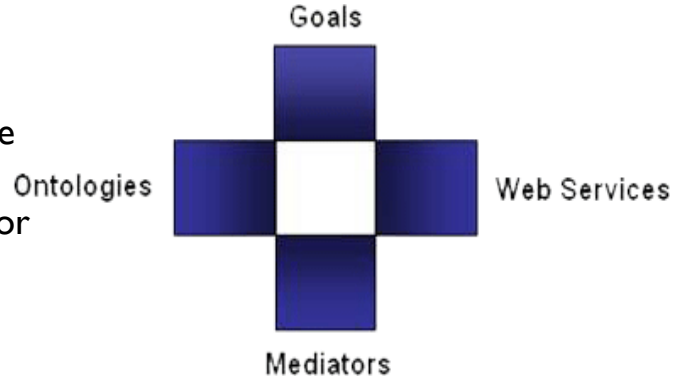


WSMO Conceptos Principales:

- **Fuerte desacoplamiento y mediación**
  - *Componentes autónomos con mediadores para la interoperabilidad.*
- **Interface vs. Implementación:**
  - *Se distingue entre la interface (descripción) y la implementación (programa).*

Objetivos que un cliente puede tener cuando consulta un servicio Web.

Proporcionan la especificación formal de la terminología de la información utilizada por todos los demás componentes.



Descripción semántica de servicios Web.

Conexión entre componentes para facilitar el manejo de la heterogeneidad.

Propiedades No Funcionales

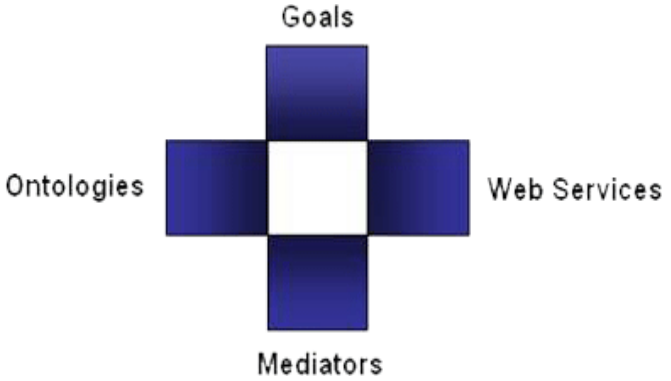
```
ontology _"http://www.example.org/ontologies/example"
  nfp
    dc#title hasValue "WSML example ontology"
    dc#subject hasValue "family"
    dc#description hasValue "fragments of a family ontology to provide WSML examples"
    dc#contributor hasValue { _"http://homepage.uibk.ac.at/~c703240/foaf.rdf",
      _"http://homepage.uibk.ac.at/~csaa5569/",
      _"http://homepage.uibk.ac.at/~c703239/foaf.rdf",
      _"http://homepage.uibk.ac.at/homepage/~c703319/foaf.rdf" }
    dc#date hasValue _date("2004-11-22")
    dc#format hasValue "text/plain"
    dc#language hasValue "en-US"
    dc#rights hasValue _"http://www.deri.org/privacy.html"
    wsml#version hasValue "$Revision: 1.13 $"
  endnfp
```

WSMO Ontologías:

- **Fuerte desacoplamiento y mediación**
  - *Componentes autónomos con mediadores para la interoperabilidad.*
- **Interface vs. Implementación:**
  - *Se distingue entre la interface (descripción) y la implementación (programa).*

Objetivos que un cliente puede tener cuando consulta un servicio Web.

Proporcionan la especificación formal de la terminología de la información utilizada por todos los demás componentes.



Descripción semántica de servicios Web.

Conexión entre componentes para facilitar el manejo de la heterogeneidad.

Ejemplo de Ontología

```
concept Human
      nonFunctionalProperties
            dc#description hasValue "concept of a human being"
      endNonFunctionalProperties
      hasName ofType foaf#name
      hasParent inverseOf(hasChild) impliesType Human
      hasChild impliesType Human
      hasAncestor transitive impliesType Human
      hasWeight ofType (1) _decimal
      hasWeightInKG ofType (1) _decimal
      hasBirthdate ofType (1) _date
      hasObit ofType (0 1) _date
      hasBirthplace ofType (1) loc#location
      isMarriedTo symmetric impliesType (0 1) Human
      hasCitizenship ofType oo#country
      isAlive ofType (1) _boolean
                  nfp
                        dc#relation hasValue {IsAlive}
                  endnfp
```

Ejemplo de Ontología

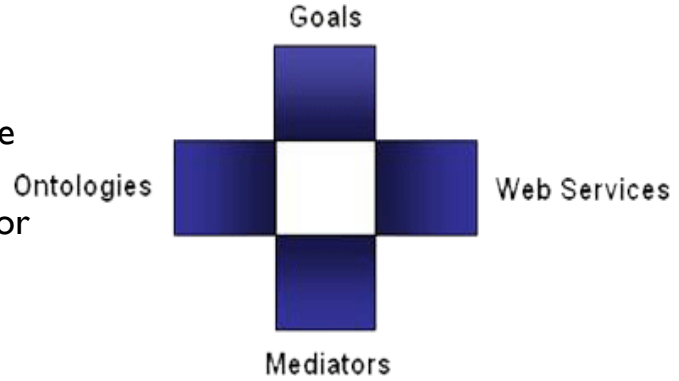
```
axiom IsAlive
      definedBy
            ?x[isAlive hasValue _boolean("true")] :-
                  naf ?x[hasObit hasValue ?obit] memberOf Human.
            ?x[isAlive hasValue _boolean("false")]
            impliedBy
                  ?x[hasObit hasValue ?obit] memberOf Human.

 axiom FunctionalDependencyAlive
       definedBy
             !- IsAlive(?x,?y1) and
                        IsAlive(?x,?y2) and ?y1 != ?y2.

concept Man subConceptOf Human
      nfp
            dc#relation hasValue ManDisjointWoman
      endnfp

concept Woman subConceptOf Human
      nfp
            dc#relation hasValue ManDisjointWoman
      endnfp

axiom ManDisjointWoman
      definedBy
            !- ?x memberOf Man and ?x memberOf Woman.
```

WSMO Capacidades/Interfaces

Solicitud/respuesta:
- **Capability** *(funcional)*
- **Interfaces** *(uso)*

Objetivos que un cliente puede tener cuando consulta un servicio Web.

Proporcionan la especificación formal de la terminología de la información utilizada por todos los demás componentes.



Descripción semántica de servicios Web.

Conexión entre componentes para facilitar el manejo de la heterogeneidad.

Especificación de Capacidades:

- **Propiedades no funcionales**
- **Importación de ontologías**
- **Uso de mediadores**
  - OO Mediator: importa ontologías como definición de terminología.
  - WG Mediator: enlaza un objetivo que es resuelto por el servicio Web.
- **Pre-condiciones**

  Qué espera un servicio Web para ser capaz de proporcionar su servicio. Definen las condiciones sobre las entradas.
- **Assumptions**

  Condiciones sobre el estado del mundo que deben ser verdaderas antes de que el servicio sea ejecutado y funcione correctamente, pero no necesariamente verificadas o verificables.
- **Post-condiciones**

  Describen el resultado del servicio Web con respecto a la entrada y las condiciones sobre ésta.
- **Efectos**

  Condiciones sobre el estado del mundo que deben ser verdaderas después de la ejecución del servicio (cambios en el estado).

Ejemplo de Capacidades

eGovernment: Effect– Service makes a child a German citizen …)

```
capability
      sharedVariables ?child
      precondition
            nonFunctionalProperties
                  dc#description hasValue "The input has to be boy or a girl
                        with birthdate in the past and be born in Germany."
            endNonFunctionalProperties
            definedBy
                  ?child memberOf Child
                        and ?child[hasBirthdate hasValue ?brithdate]
                        and wsml#dateLessThan(?birthdate,wsml#currentDate())
                        and ?child[hasBirthplace hasValue ?location]
                        and ?location[locatedIn hasValue oo#de]
                        or (?child[hasParent hasValue ?parent] and
                                    ?parent[hasCitizenship hasValue oo#de] ) .

      assumption
            nonFunctionalProperties
                  dc#description hasValue "The child is not dead"
            endNonFunctionalProperties
            definedBy
                  ?child memberOf Child
                        and naf ?child[hasObit hasValue ?x].

      effect
            nonFunctionalProperties
                  dc#description hasValue "After the registration the child
                        is a German citizen"
            endNonFunctionalProperties
            definedBy
                  ?child memberOf Child
                        and ?child[hasCitizenship hasValue oo#de].
```

WSMO Interfaces

- descripción completa
- aspectos de calidad
- manejo del servicio

Propiedades no funcionales

- publicación del servicio
- soporte para el descubrimiento

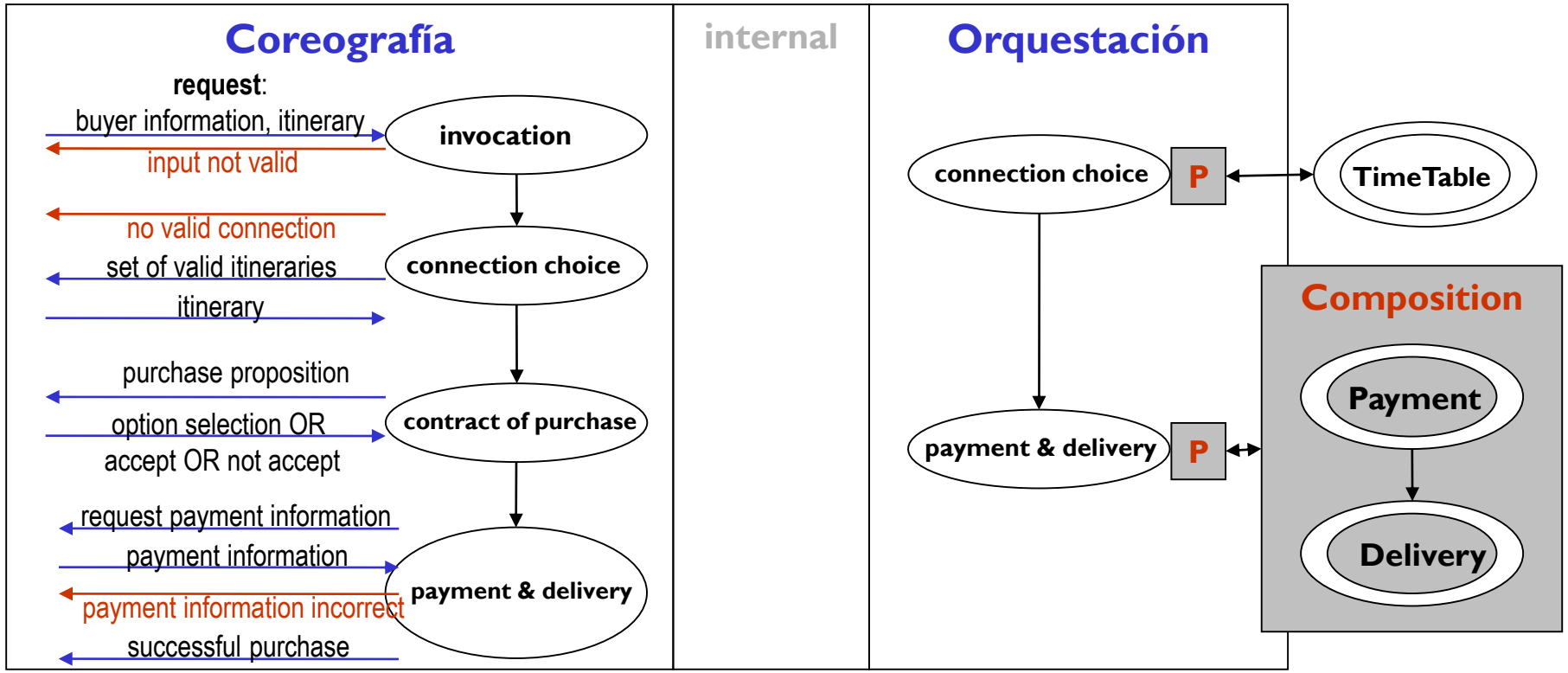
Capacidades

Core + WS-specific

Descripción funcional

Coreografía --- Interfaces --- **Orquestación**

Web Service Interfaces



SAWSDL

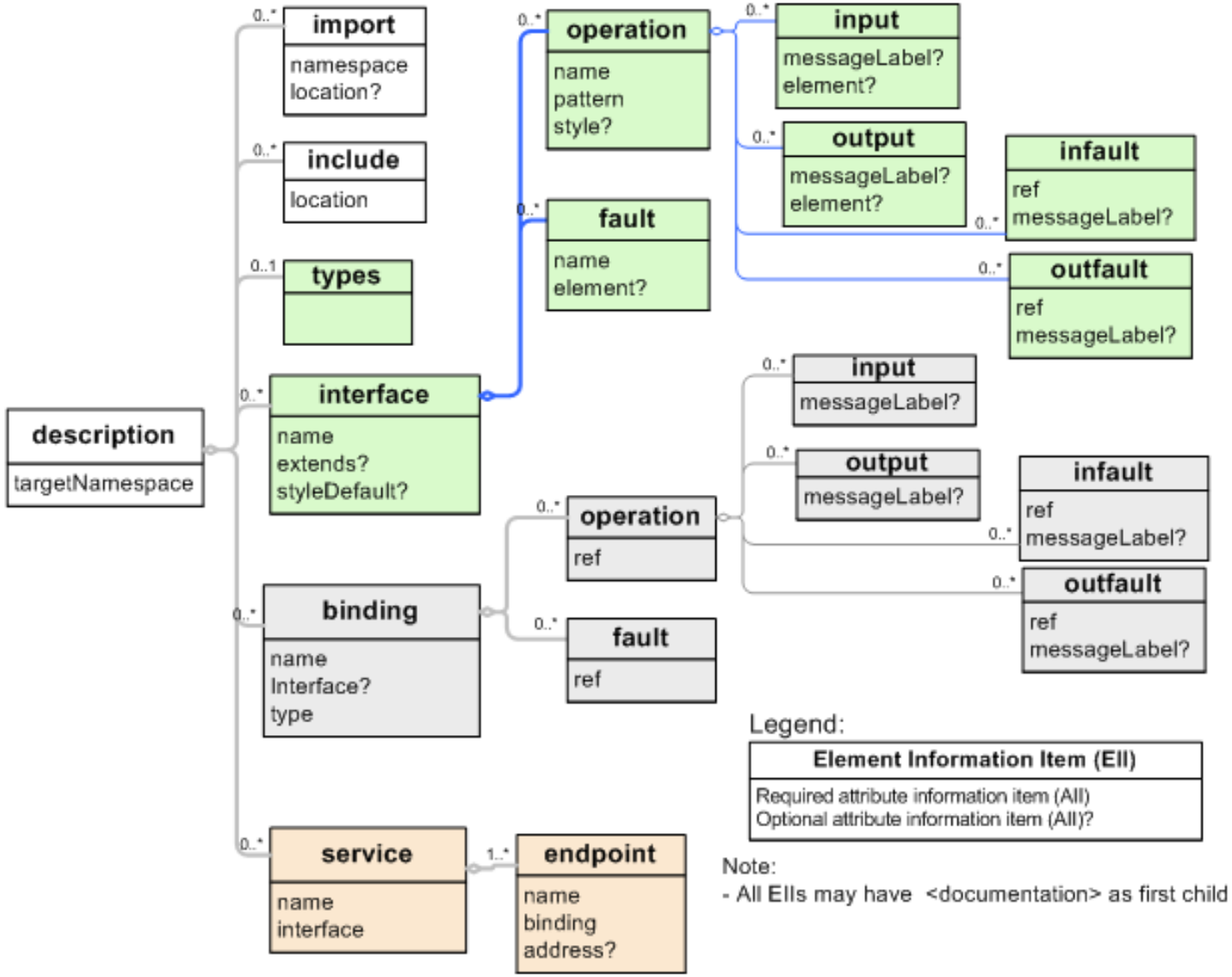
SAWSDL

- Las anotaciones semánticas para WSDL (SAWSDL) es una recomendación de la W3C
  - Define un mecanismo para agregar anotaciones semánticas a servicios web
  - Se basa en la presentación de miembros del W3C WSDL-S
  - WSDL-S fue propuesto por la Universidad de Georgia METEOR-S Team * e IBM
- Es un primer paso importante de W3C para agregar soporte para el modelado semántico de la pila de servicios web.

¿Qué es WSDL?

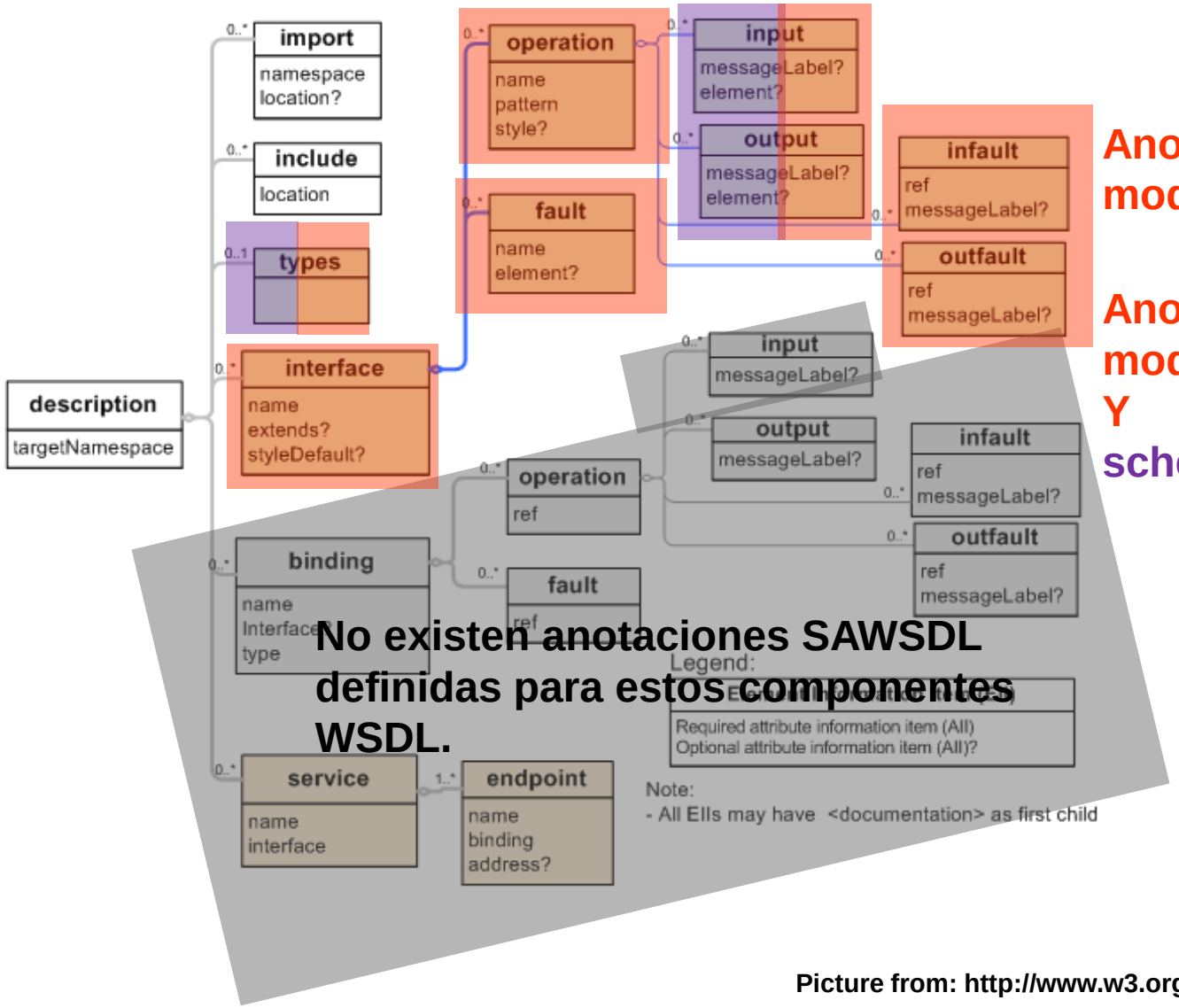
- Un lenguaje basado en XML para la descripción de servicios.
- Proporciona la descripción funcional de los servicios:
  - Descripción de las interfaces
  - Detalles de los protocolos de acceso y despliegue
  - Toda la información funcional necesaria para invocar un servicio desde un programa
- No incluye
  - Descripción de la calidad del servicio (QoS)
  - Taxonomías
  - Información del negocio

Modelo de componentes en WSDL 2.0



¿Por qué utilizar SAWSDL?

- Permite la anotación semántica de los servicios Web cuyos tipos de datos se describen en el esquema XML.
- Brinda soporte para mecanismos de mapeo entre tipos de esquema y ontologías de servicios web.
- Basado en los estándares de los servicios web existentes utilizando solo elementos de extensibilidad.
- Cuenta con un mecanismo independiente del lenguaje de representación semántica.
- SAWSDL ayuda a la integración proporcionando mapeo a modelos de dominio acordados (ontologías, estándares como Rosetta Net, ebXML).
- Mejora documentación al agregar anotación funcional.
- Facilidad en actualizaciones de herramientas.



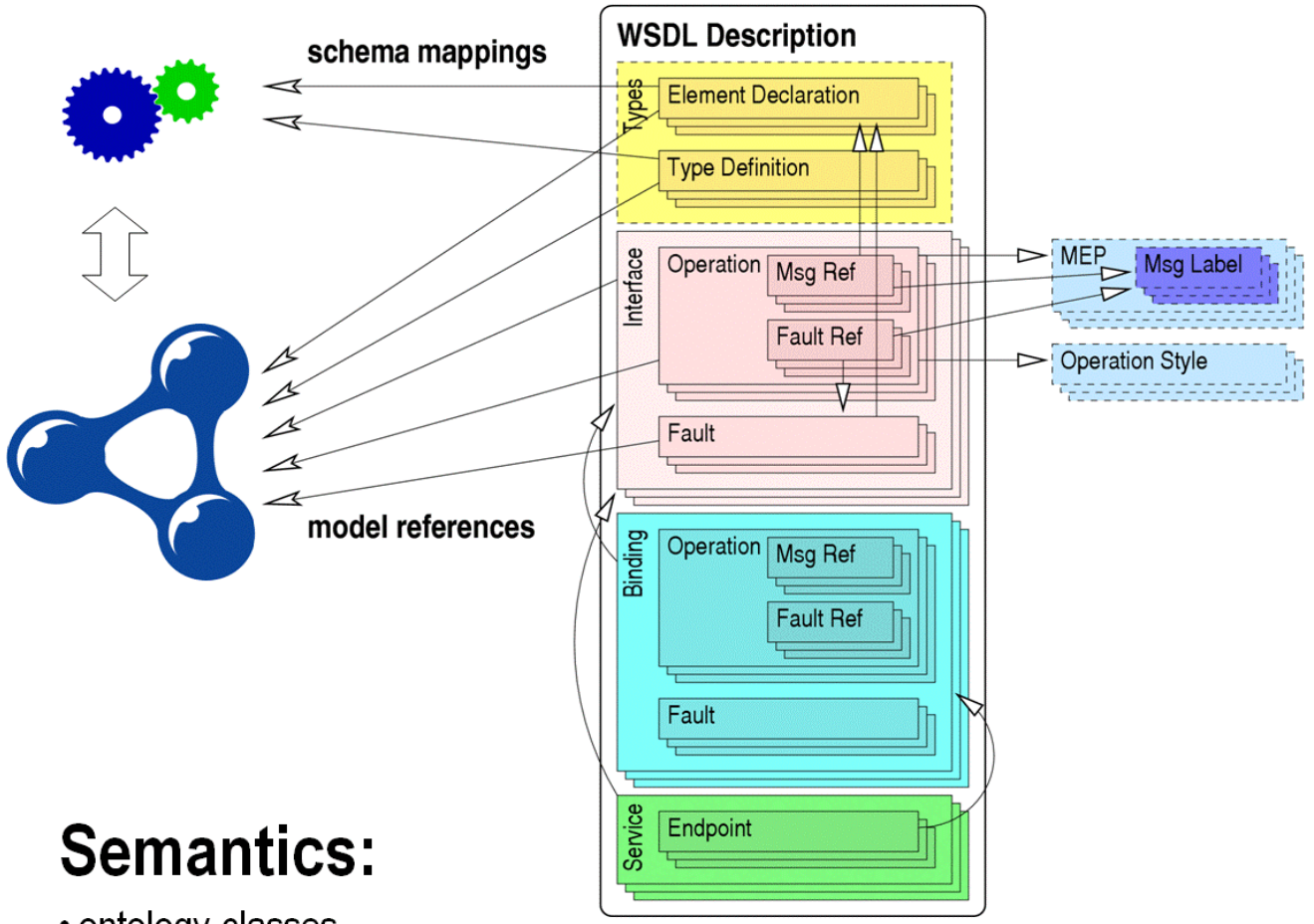
No existen anotaciones SAWSDL definidas para estos componentes WSDL.

Anotado usando modelReference

Anotado usando modelReference Y schemaMapping

Picture from: http://www.w3.org/TR/wsdl20-primer/

Modelo de referencia y mapeo de esquemas



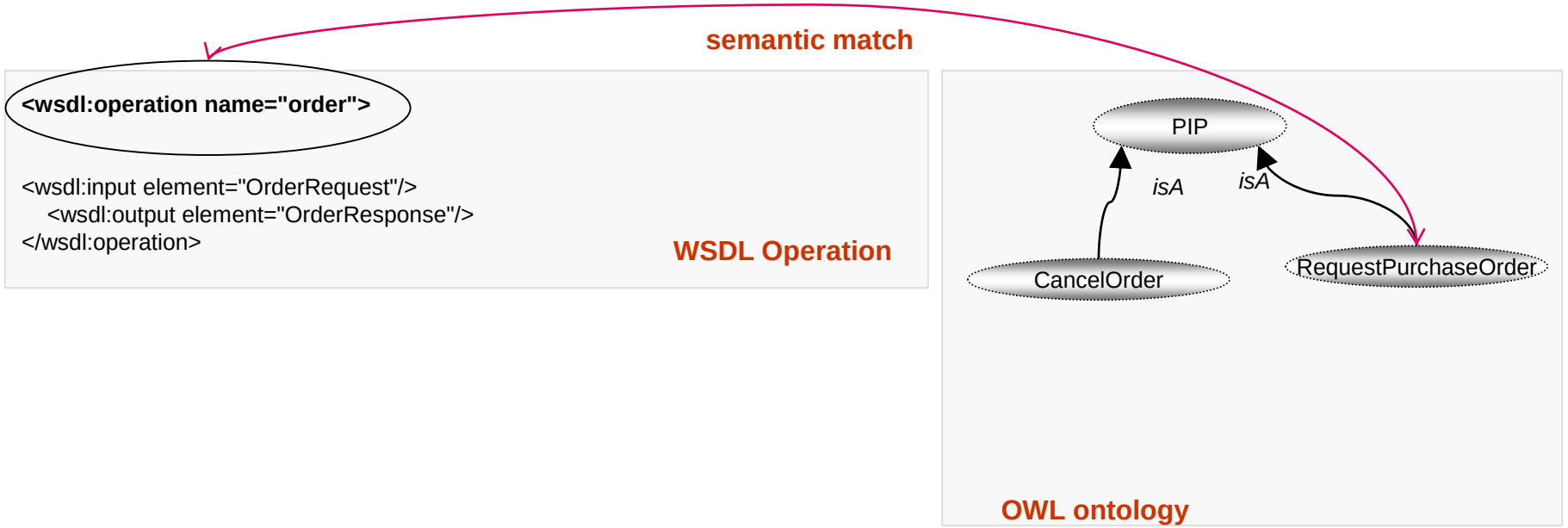
Semantics:

- ontology classes
  - discovery, composition
  - filtering, ranking
- lifting/lowering mappings
  - mediation, invocation
- functionality categories
  - publishing, discovery, composition
- anything, really

SAWSDL define dos atributos de extensibilidad

- modelReference: se utiliza para especificar la asociación entre un componente WSDL o Esquema XML y un concepto en algún modelo semántico.
- Se utiliza para anotar lo siguiente:
  - Componentes WSDL
    - Interfaces
    - Operaciones
    - fallas
  - Definiciones de tipo WSDL
    - Definiciones de tipo complejo de esquema XML
    - Definiciones de tipos simples
    - declaraciones de elementos
    - declaraciones de atributos
- liftingSchemaMapping: se utiliza para especificar asignaciones entre Definiciones de tipo WSDL en XML y datos semánticos.
- loweringSchemaMapping: se utiliza para especificar asignaciones entre datos semánticos y definiciones de tipo WSDL en XML.

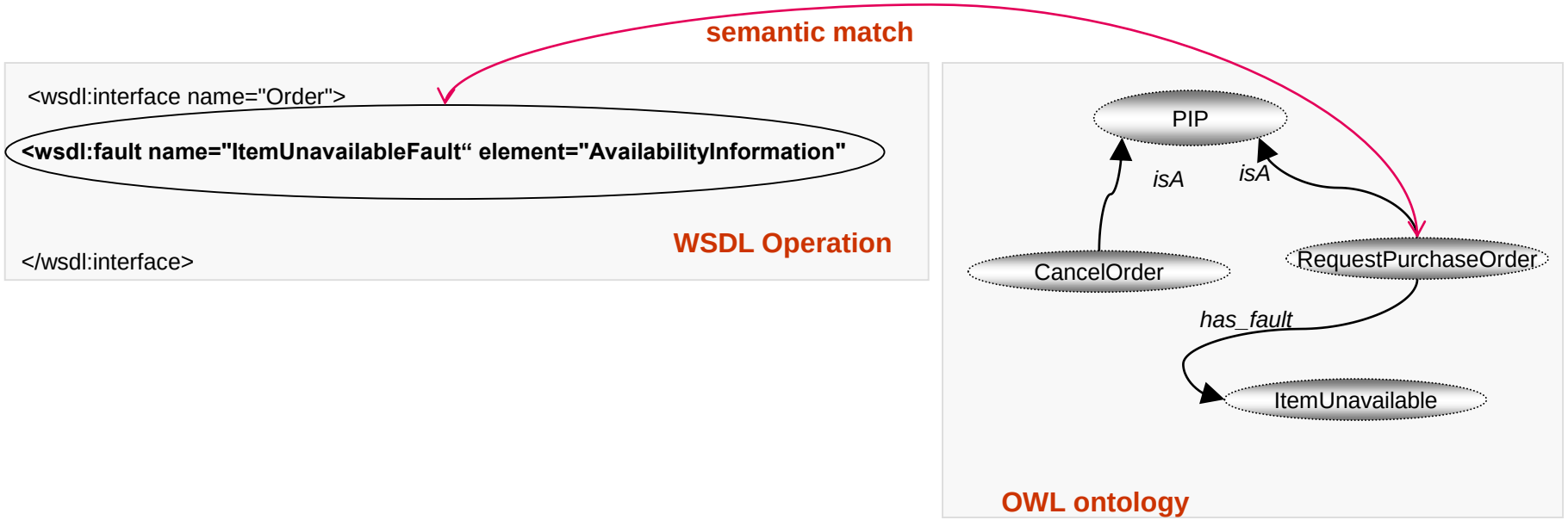
Anotación de operaciones con el ModelReference




```
<wsdl:operation name="order"
   sawsdl:modelReference="http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#RequestPurchaseOrder">
   <wsdl:input element="OrderRequest"/>
   <wsdl:output element="OrderResponse"/>
</wsdl:operation>
```

La anotación de elementos tipo operación lleva una referencia a un concepto en un modelo semántico que proporciona una descripción de alto nivel de la operación. La descripción especifica sus aspectos de comportamiento o incluye otras definiciones semánticas.

Anotación de las fallas con el ModelReference



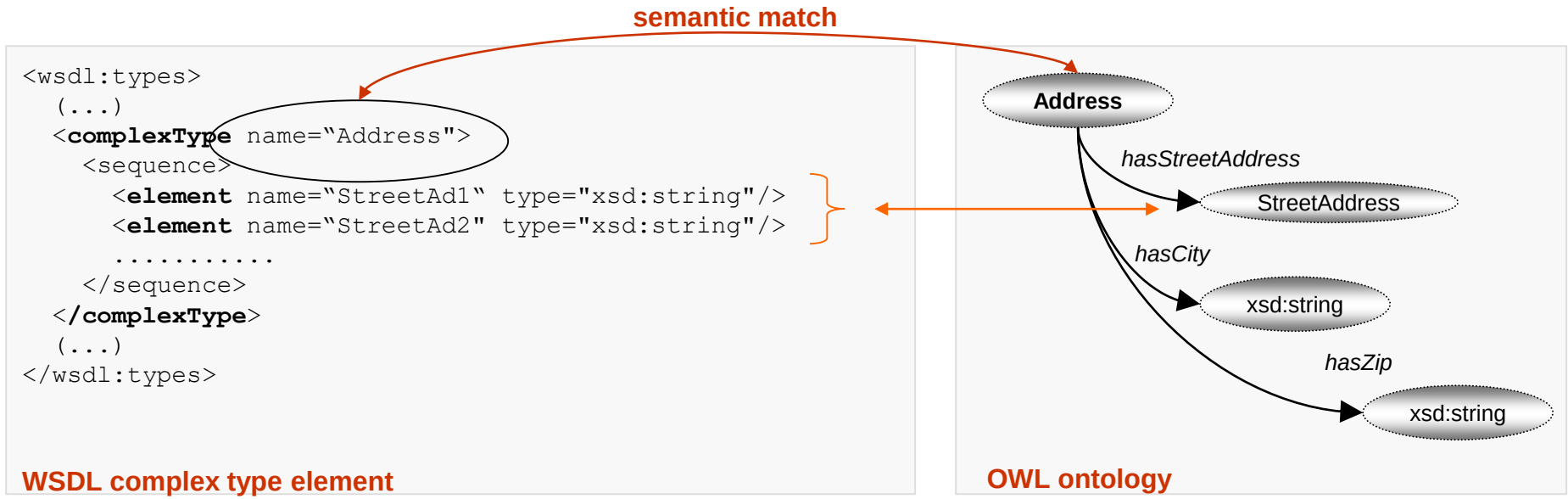

```
<wsdl:interface name="Order">
   <wsdl:fault name="ItemUnavailableFault" element="AvailabilityInformation"
      sawsdl:modelReference="http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#ItemUnavailable"/>
</wsdl:interface>
```

La anotación del elemento de falla lleva una referencia a un concepto en un modelo semántico que proporciona una descripción de alto nivel de la falla y puede incluir otras definiciones semánticas.

Anotación de Tipos

- Las siguientes definiciones de tipos en WSDL también pueden ser anotadas utilizando el *modelReference*, el *liftingSchemaMapping* , y el *loweringSchemaMapping*:
  - XML Schema Complex type definitions
  - Simple type definitions
  - Element declarations
  - Attribute declarations

Anotación de Tipos



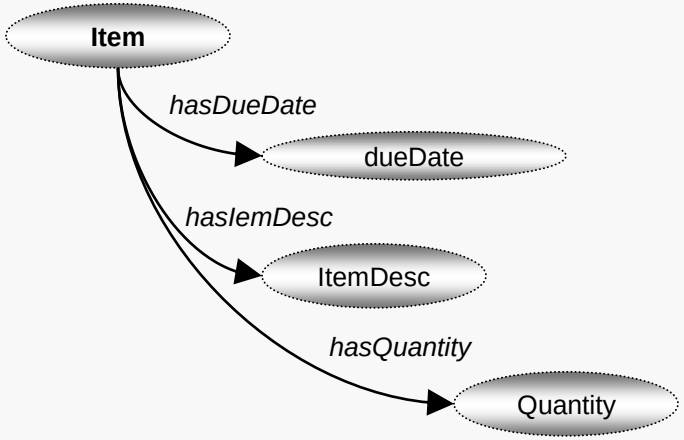
modelReference para establecer una relación semántica

liftingSchemaMapping y *loweringSchemaMapping* para realizar mapeos entre XML y el modelo semántico.

Anotación de Tipos de Datos Complejos

```
<complexType name="POItem" >
 <all>
  <element name="dueDate" nillable="true" type="dateTime"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#DueDate”/>
  <element name="qty" type="float"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#Quantity”/>
  <element name="EANCode" nillable="true" type="string"
    sawsdl:modelReference=”
      http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#ItemCode”/>
  <element name="itemDesc" nillable="true" type="string"
    sawsdl:modelReference=”
   http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#ItemDesc” />
  </all>
</complexType>
```

WSDL complex type element



OWL ontology

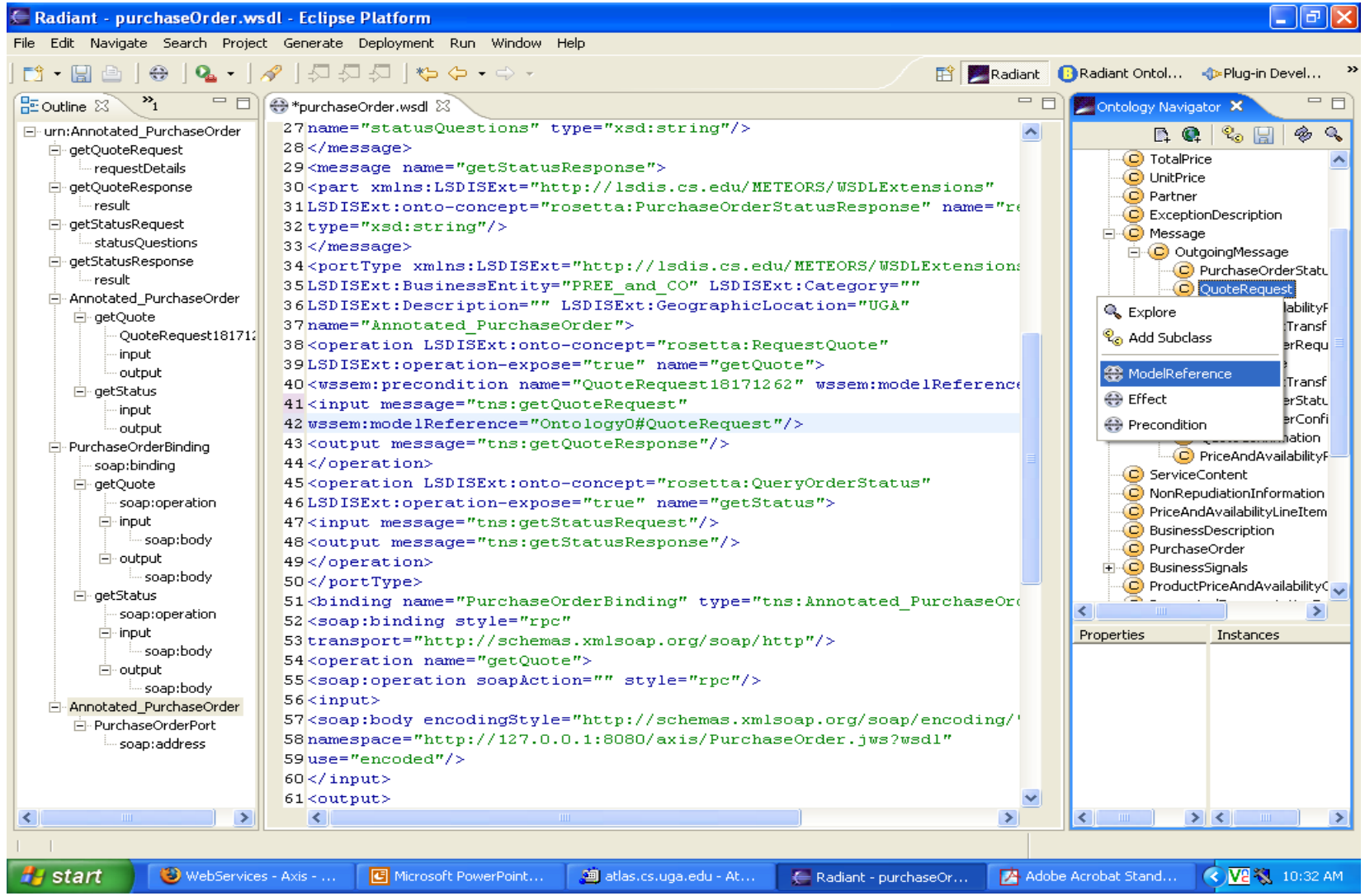
Ejemplo SAWSDL

```
<wsdl:description targetNamespace="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
   xmlns:wsdl="http://www.w3.org/ns/wsdl" xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:sawsdl="http://www.w3.org/ns/sawsdl">
   <wsdl:types>
     <xs:element name= "processPurchaseOrderResponse" type="xs:string
       sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#PurchaseOrderResponse”
       sawsdl:liftingSchemaMapping= http://www.w3.org/2002/ws/sawsdl/spec/mapping/POResponse2Ont.xslt
       sawsdl:loweringSchemaMapping= "http://www.w3.org/2002/ws/sawsdl/spec/mapping/Ont2Response.xslt>
        ……
    </xs:element>
  </wsdl:types>
  <interface name="PurchaseOrder"
  < sawsdl:modelReference="http://example.org/categorization/products/electronics />
   <operation name=“order” pattern=wsdl:in-out
        sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#RequestPurchaseOrder"" >
        <input messageLabel = ”processPurchaseOrderRequest"
           element="tns:processPurchaseOrderRequest"/>
        <output messageLabel ="processPurchaseOrderResponse"
           element="processPurchaseOrderResponse"/>
   </operation>
   <operation name=“cancel” pattern=wsdl:in-out
        sawsdl:modelReference=“http://www.w3.org/2002/ws/sawsdl/spec/ontology/rosetta#CancelOrder"" >
        <input messageLabel = ”processCancelRequest"
          element="tns:processCancelRequest"/>
        <output messageLabel ="processCancelResponse"
          element="processCancelResponse"/>
    </operation>
 </interface>
</wsdl:description >
```

Herramientas

- Grupos de trabajo
  - W3C SAWSDL Grupo de trabajo, http://www.w3.org/2002/ws/sawsdl/
  - W3C WSDL-S member submission Web page
  - http://www.w3.org/Submission/WSDL-S/
- Herramientas:
  - SAWSDL4J by Wright State University (Dayton,OH) and University of Georgia (UGA, Athens,GA)
  - Radiant: WSDL-S/SAWSDL Annotation Tool by University of Georgia
  - Semantic Tools for Web Services by IBM alphaWorks
  - WSMO Studio by DERI

Radiant


Radiant - purchaseOrder.wsdl - Eclipse Platform
File Edit Navigate Search Project Generate Deployment Run Window Help
Radiant
Radiant Ontol...
Plug-in Devel...
Outline
urn:Annotated_PurchaseOrder
getQuoteRequest
requestDetails
getQuoteResponse
result
getStatusRequest
statusQuestions
getStatusResponse
result
Annotated_PurchaseOrder
getQuote
QuoteRequest18171
input
output
getStatus
input
output
PurchaseOrderBinding
soap:binding
getQuote
soap:operation
input
soap:body
output
soap:body
getStatus
soap:operation
input
soap:body
output
soap:body
Annotated_PurchaseOrder
PurchaseOrderPort
soap:address
*purchaseOrder.wsdl
27 name="statusQuestions" type="xsd:string"/>
28 </message>
29 <message name="getStatusResponse">
30 <part xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions"
31 LSDISExt:onto-concept="rosetta:PurchaseOrderStatusResponse" name="re
32 type="xsd:string"/>
33 </message>
34 <portType xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions
35 LSDISExt:BusinessEntity="PREE_and_CO" LSDISExt:Category=""
36 LSDISExt:Description="" LSDISExt:GeographicLocation="UGA"
37 name="Annotated_PurchaseOrder">
38 <operation LSDISExt:onto-concept="rosetta:RequestQuote"
39 LSDISExt:operation-expose="true" name="getQuote">
40 <wssem:precondition name="QuoteRequest18171262" wssem:modelReferenc
41 <input message="tns:getQuoteRequest"
42 wssem:modelReference="OntologyO#QuoteRequest"/>
43 <output message="tns:getQuoteResponse"/>
44 </operation>
45 <operation LSDISExt:onto-concept="rosetta:QueryOrderStatus"
46 LSDISExt:operation-expose="true" name="getStatus">
47 <input message="tns:getStatusRequest"/>
48 <output message="tns:getStatusResponse"/>
49 </operation>
50 </portType>
51 <binding name="PurchaseOrderBinding" type="tns:Annotated_PurchaseOr
52 <soap:binding style="rpc"
53 transport="http://schemas.xmlsoap.org/soap/http"/>
54 <operation name="getQuote">
55 <soap:operation soapAction="" style="rpc"/>
56 <input>
57 <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
58 namespace="http://127.0.0.1:8080/axis/PurchaseOrder.jws?wsdl"
59 use="encoded"/>
60 </input>
61 <output>
Ontology Navigator
TotalPrice
UnitPrice
Partner
ExceptionDescription
Message
OutgoingMessage
PurchaseOrderStatu
QuoteRequest
Explore
Add Subclass
ModelReference
Effect
Precondition
PriceAndAvailabilityF
ServiceContent
NonRepudiationInformation
PriceAndAvailabilityLineItem
BusinessDescription
PurchaseOrder
BusinessSignals
ProductPriceAndAvailability
Properties
Instances
start
WebServices - Axis - ...
Microsoft PowerPoint...
atlas.cs.uga.edu - At...
Radiant - purchaseOr...
Adobe Acrobat Stand...
10:32 AM

Radiant Web

http://mango.ctegd.uga.edu/jkissingLab/SWS/RadiantWeb/index.html