Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Emerging Web Service Technology

Introduction to Web Services

WS 2009/2010

Mohammed AbuJarour

Tobias Vogel

Nov 12, 2009

Schedule

Name	Topic	Date	Supervisor
Daniel Eichler	BPEL-Mora: Lightweight Embeddable Extensible BPEL Engine	Nov 26, 2009	Mohammed AbuJarour
Jan-Felix Schwarz	Enabling business experts to discover web services for business process automation	Dec 3, 2009	Mohammed AbuJarour
Christoph Thiele	Service selection by choreography-driven matching	Dec 10, 2009	Tobias Vogel
Abdelfattah Elnaggar	A logic-based approach for service discovery with composition support	Dec 17, 2009	Mohammed AbuJarour
Martin Lorenz	Composite web services	Jan 7, 2010	Tobias Vogel
Edgar Naether	Model Driven Design of Web Service Operations using Web Engineering Practices	Jan 14, 2010	Tobias Vogel
David Jaeger	Model-Driven Performance Evaluation for Service Engineering	Jan 21, 2010	Mohammed AbuJarour
Fabian Lindenberg	Reputation Propagation in Composite Services	Jan 28, 2010	Mohammed AbuJarour
Henrik Steudel	Tools for Semantic Web Services	Feb 4, 2010	Tobias Vogel

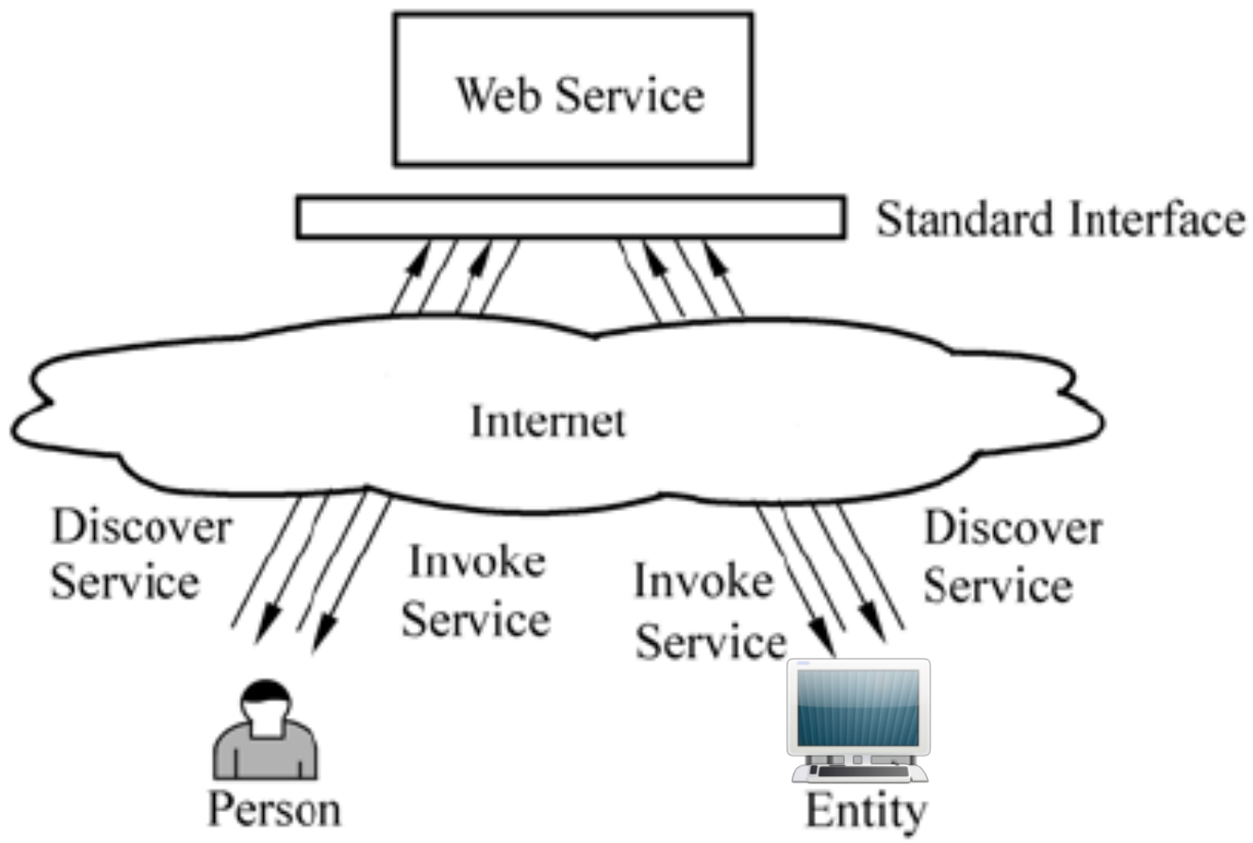
Contents

- Previously
- WS Standard Stack
- Basic Concepts of Web Services
  - WSDL
  - SOAP
  - Binding
  - UDDI
- Composite Web Services
  - BPEL
- Quality of Service
- Semantic Web Services

Area	Topic
Service Discovery	Enabling business experts to discover web services for business process automation
	Service selection by choreography-driven matching
	A logic-based approach for service discovery with composition support
Service Composition	Composite web services
	Model Driven Design of Web Service Operations using Web Engineering Practices
Service Management	BPEL-Mora: Lightweight Embeddable Extensible BPEL Engine
Quality of Service	Model-Driven Performance Evaluation for Service Engineering
	Reputation Propagation in Composite Services
Semantic Web Services	Tools for Semantic Web Services

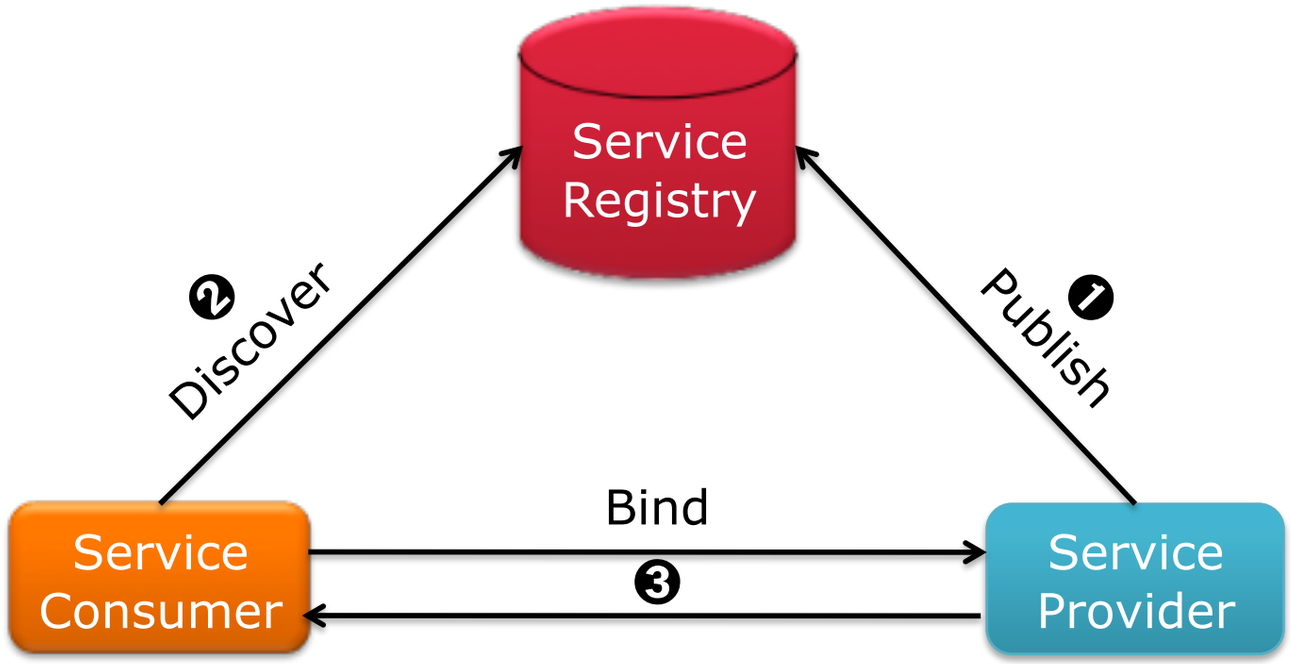
Definitions

- **What is a Service?**
  - "The performance of work (a function) by one for another" [4]
- **What is a Web Service?**
  - "A Web Service is a platform-independent programmable module with standard interface descriptions that provide universal accessibility through standard communication protocols" [5]



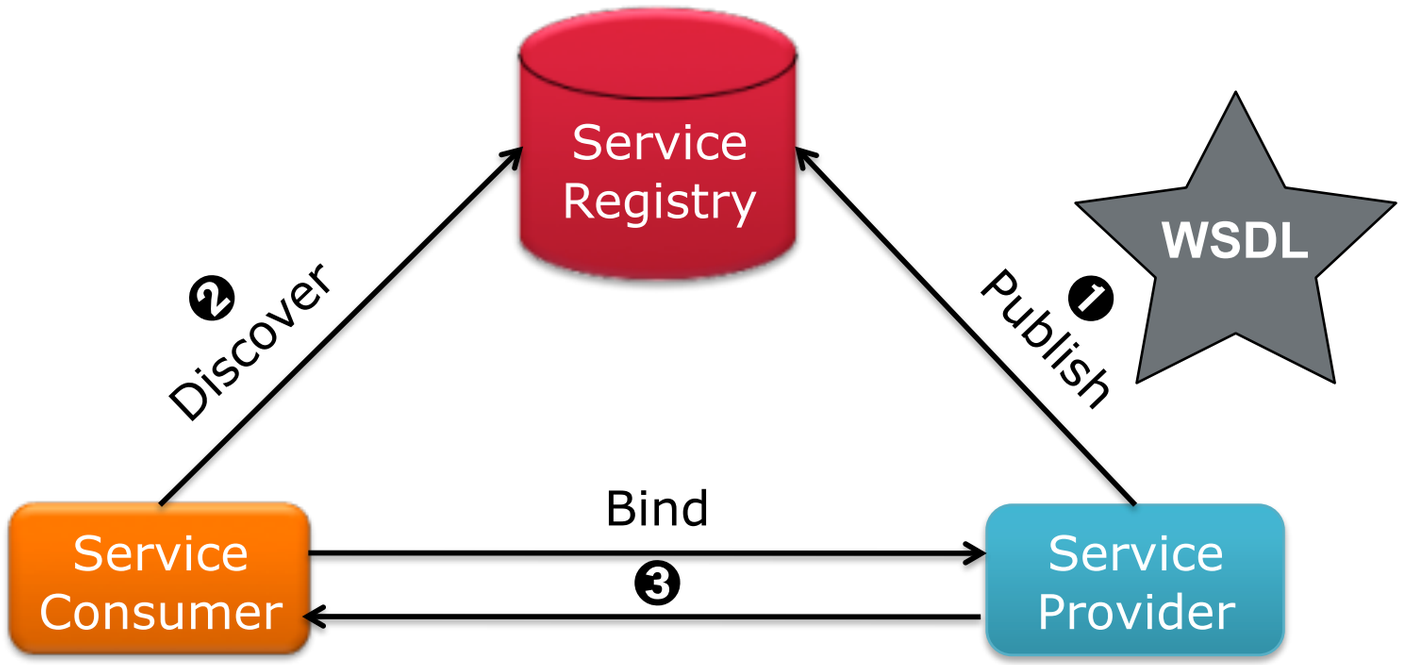
Previously ...

Triangular SOA Operational Model



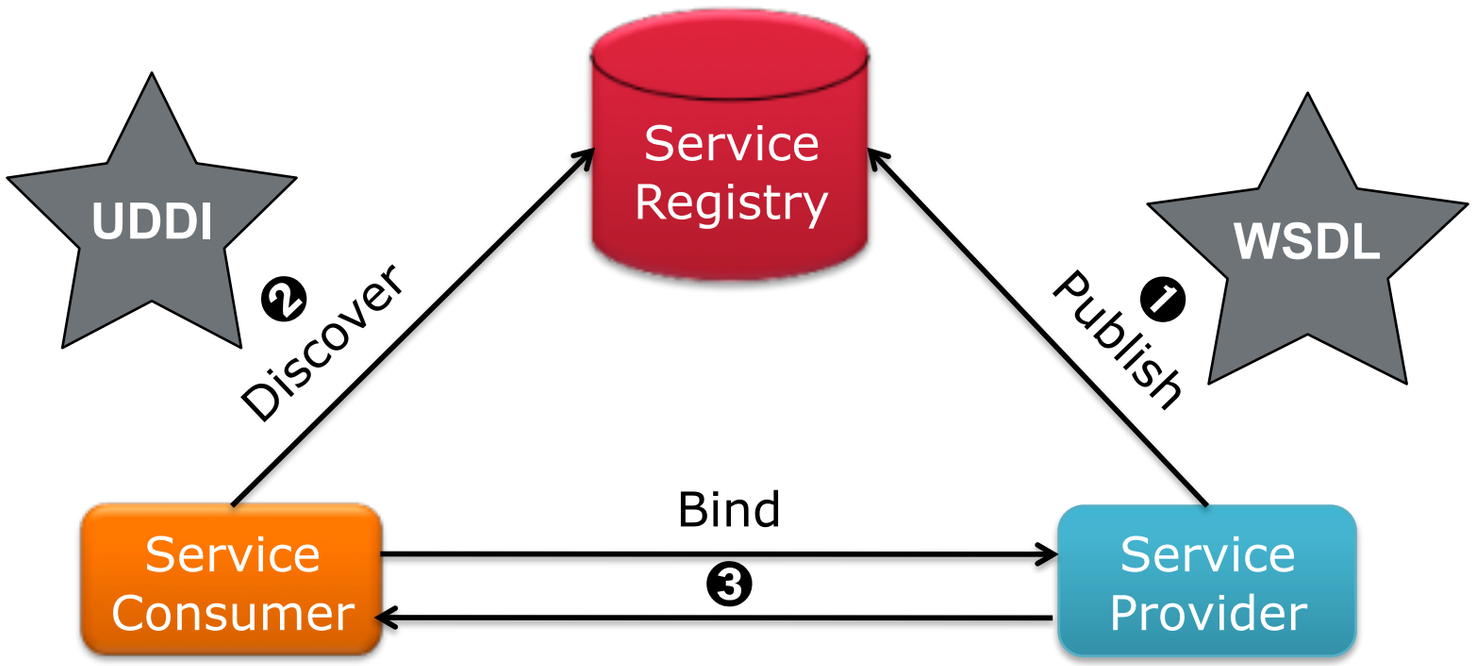
Previously ...

Triangular SOA Operational Model



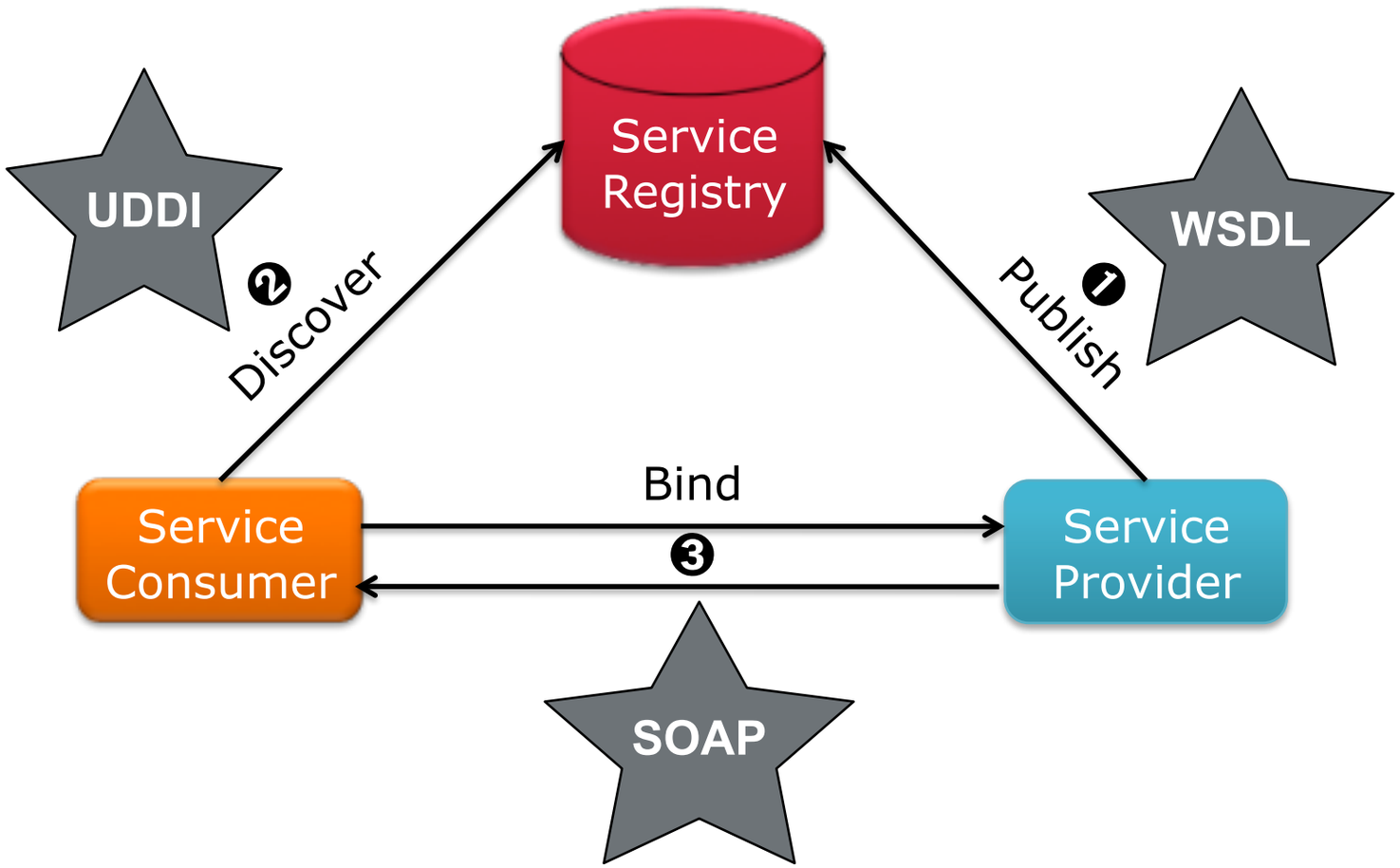
Previously ...

Triangular SOA Operational Model



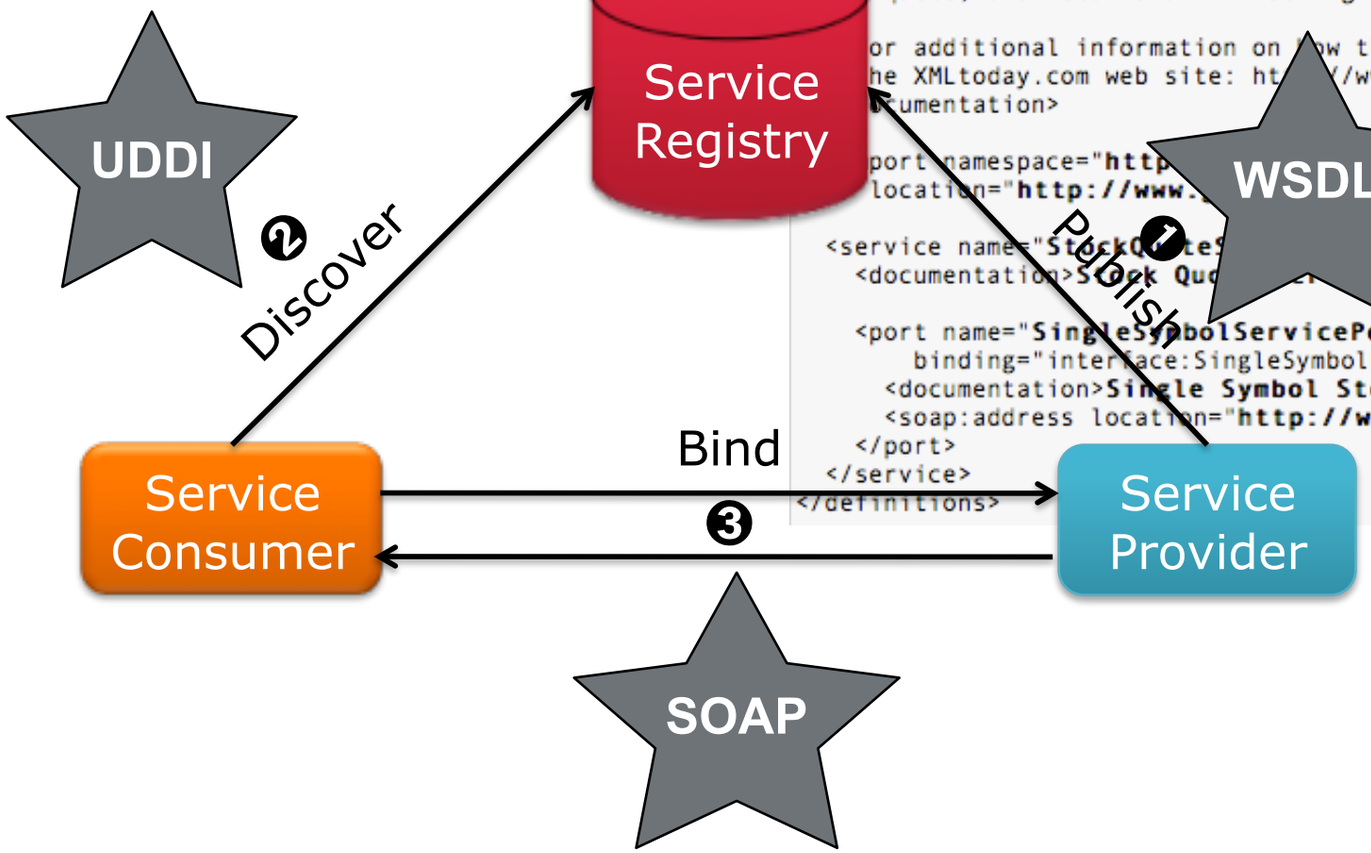
Previously ...

Triangular SOA Operational Model



Previously ...

Triangular SOA Operational Model



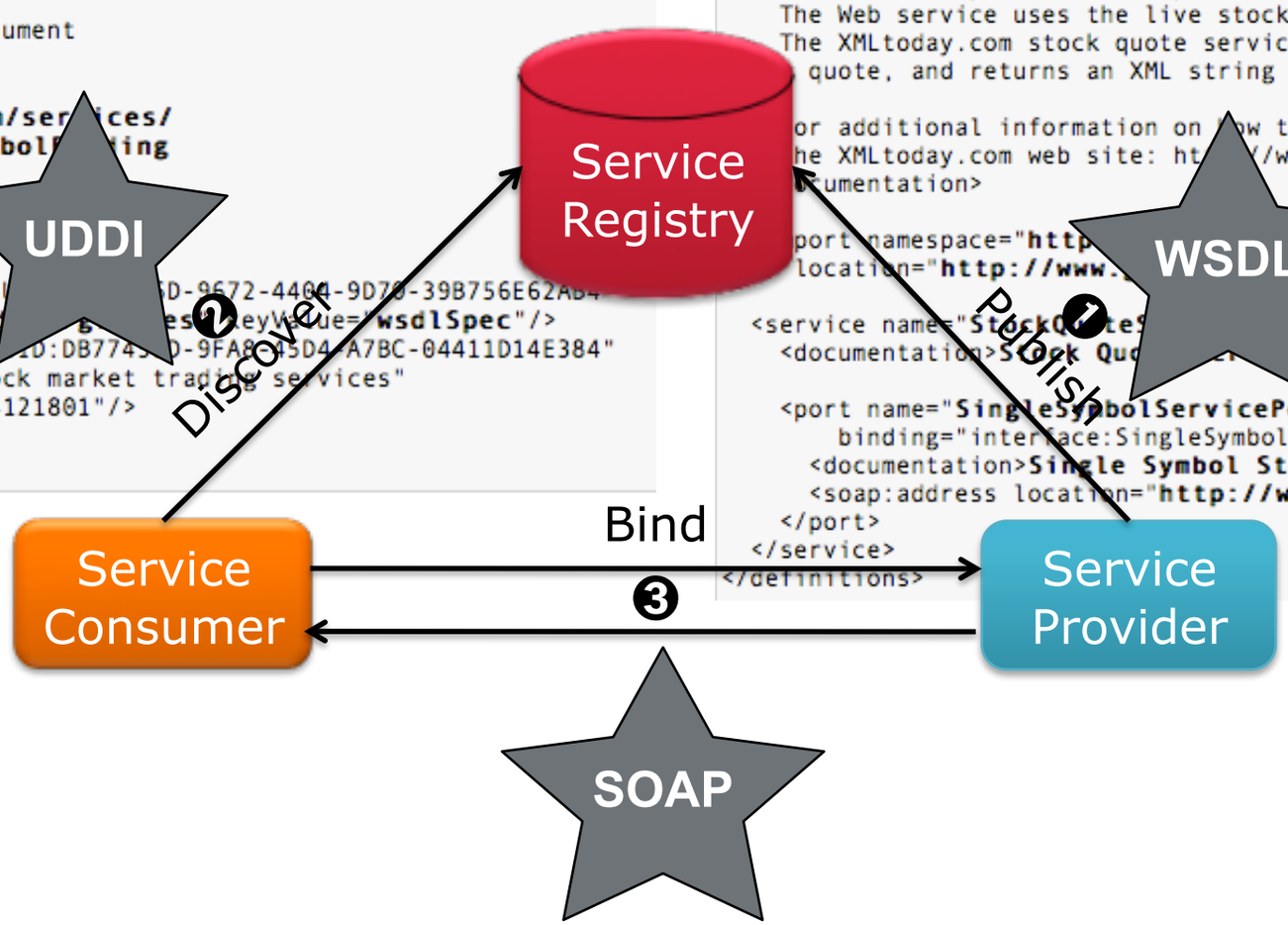

```
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:interface="http://www.getquote.com/StockQuoteService-interface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <documentation>
    This service provides an implementation of a standard stock quote service.
    The Web service uses the live stock quote service provided by XMLtoday.com.
    The XMLtoday.com stock quote service uses an HTTP GET interface to request
     quote, and returns an XML string as a response.

    or additional information on how this service obtains stock quotes, go to
    he XMLtoday.com web site: ht  //www.xmltoday.com/examples/soap/stock.psp.
    umentation>

    port namespace="http  om/StockQuoteService-interface"
     location="http://www.  wsdl/SQS-interface.wsdl"/>

  <service name="StockQ  te
    <documentation>Stock Quo  /documentation>

    <port name="SingleSymbolServicePort"
        binding="interface:SingleSymbolBinding">
      <documentation>Single Symbol Stock Quote Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

Previously ...

Triangular SOA Operational Model

```
<?xml version="1.0"?>
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-interface</name>

  <description xml:lang="en">
    Standard service interface definition for a stock quote service.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/
SQS-interface.wsdl#SingleSymbolBinding
    </overviewURL>
  </overviewDoc>

  <categoryBag>
    <keyedReference tModelKey="UUID:DB77450D-9672-4404-9D70-39B756E62AB4
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```

```
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:interface="http://www.getquote.com/StockQuoteService-interface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <documentation>
    This service provides an implementation of a standard stock quote service.
    The Web service uses the live stock quote service provided by XMLtoday.com.
    The XMLtoday.com stock quote service uses an HTTP GET interface to request
     quote, and returns an XML string as a response.

    or additional information on how this service obtains stock quotes, go to
    he XMLtoday.com web site: http://www.xmltoday.com/examples/soap/stock.psp.
    umentation>

    port namespace="http://www.getquote.com/StockQuoteService-interface"
     location="http://www.getquote.com/wsdl/SQS-interface.wsdl"/>

  <service name="StockQuoteService">
    <documentation>Stock Quote Service</documentation>

    <port name="SingleSymbolServicePort"
        binding="interface:SingleSymbolBinding">
      <documentation>Single Symbol Stock Quote Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

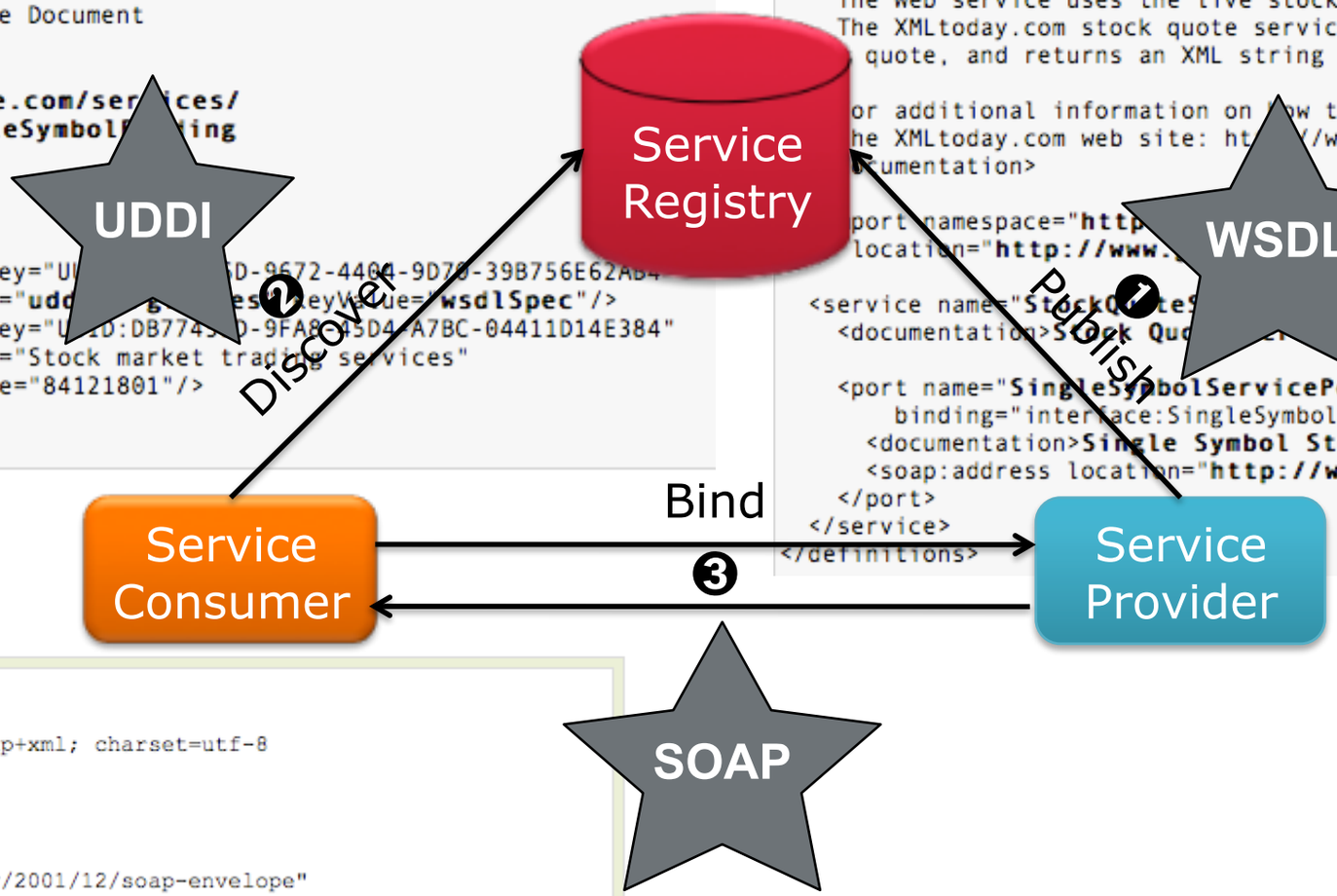


Previously ...

Triangular SOA Operational Model

```
<?xml version="1.0"?>
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-interface</name>

  <description xml:lang="en">
    Standard service interface definition for a stock quote service.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/
SQS-interface.wsdl#SingleSymbolBinding
    </overviewURL>
  </overviewDoc>

  <categoryBag>
    <keyedReference tModelKey="UUID:DB77450D-9672-4404-9D70-39B756E62AB4
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB77450D-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```

```
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:interface="http://www.getquote.com/StockQuoteService-interface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <documentation>
    This service provides an implementation of a standard stock quote service.
    The Web service uses the live stock quote service provided by XMLtoday.com.
    The XMLtoday.com stock quote service uses an HTTP GET interface to request
     quote, and returns an XML string as a response.

    or additional information on how this service obtains stock quotes, go to
    he XMLtoday.com web site: http://www.xmltoday.com/examples/soap/stock.psp.
    umentation>

    port namespace="http:// com/StockQuoteService-interface"
     location="http://www. wsdl/SQS-interface.wsdl"/>

  <service name="StockQuoteS
    <documentation>Stock Quo /documentation>

    <port name="SingleSymbolServicePort"
        binding="interface:SingleSymbolBinding">
      <documentation>Single Symbol Stock Quote Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```



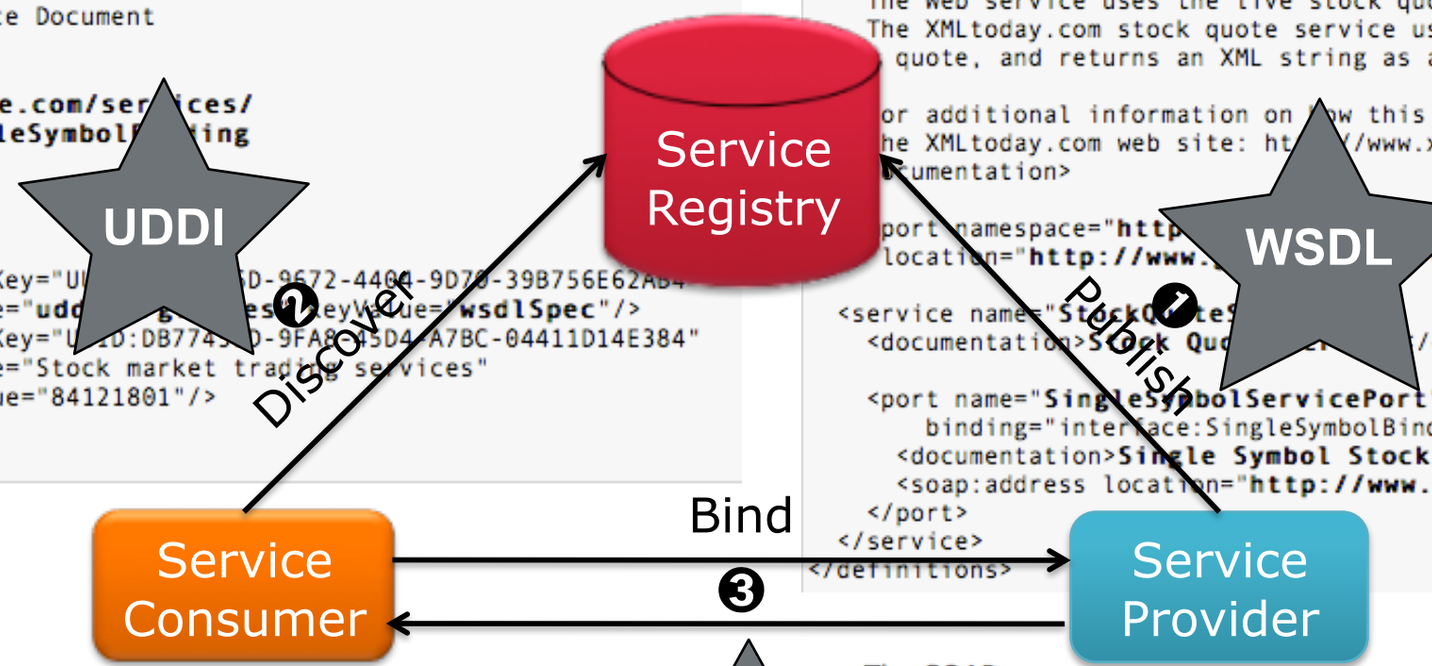
A SOAP request:

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

Previously ...

Triangular SOA Operational Model

```
<?xml version="1.0"?>
<tModel tModelKey="">
  <name>http://www.getquote.com/StockQuoteService-interface</name>

  <description xml:lang="en">
    Standard service interface definition for a stock quote service.
  </description>

  <overviewDoc>
    <description xml:lang="en">
      WSDL Service Interface Document
    </description>
    <overviewURL>
      http://www.getquote.com/services/
SQS-interface.wsdl#SingleSymbolBinding
    </overviewURL>
  </overviewDoc>

  <categoryBag>
    <keyedReference tModelKey="UUID:...-9672-4404-9D70-39B756E62AB4"
                    keyName="uddi-org:types" keyValue="wsdlSpec"/>
    <keyedReference tModelKey="UUID:DB7743...-9FA8-45D4-A7BC-04411D14E384"
                    keyName="Stock market trading services"
                    keyValue="84121801"/>
  </categoryBag>
</tModel>
```




```
<?xml version="1.0"?>
<definitions name="StockQuoteService"
  targetNamespace="http://www.getquote.com/StockQuoteService"
  xmlns:interface="http://www.getquote.com/StockQuoteService-interface"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <documentation>
    This service provides an implementation of a standard stock quote service.
    The Web service uses the live stock quote service provided by XMLtoday.com.
    The XMLtoday.com stock quote service uses an HTTP GET interface to request
     quote, and returns an XML string as a response.

    or additional information on how this service obtains stock quotes, go to
    he XMLtoday.com web site: http://www.xmltoday.com/examples/soap/stock.psp.
    umentation>

    port namespace="http...com/StockQuoteService-interface"
     location="http://www.../wsdl/SQS-interface.wsdl"/>

  <service name="StockQuoteS...
    <documentation>Stock Quo.../documentation>

    <port name="SingleSymbolServicePort"
        binding="interface:SingleSymbolBinding">
      <documentation>Single Symbol Stock Quote Service</documentation>
      <soap:address location="http://www.getquote.com/stockquoteservice"/>
    </port>
  </service>
</definitions>
```

A SOAP request:

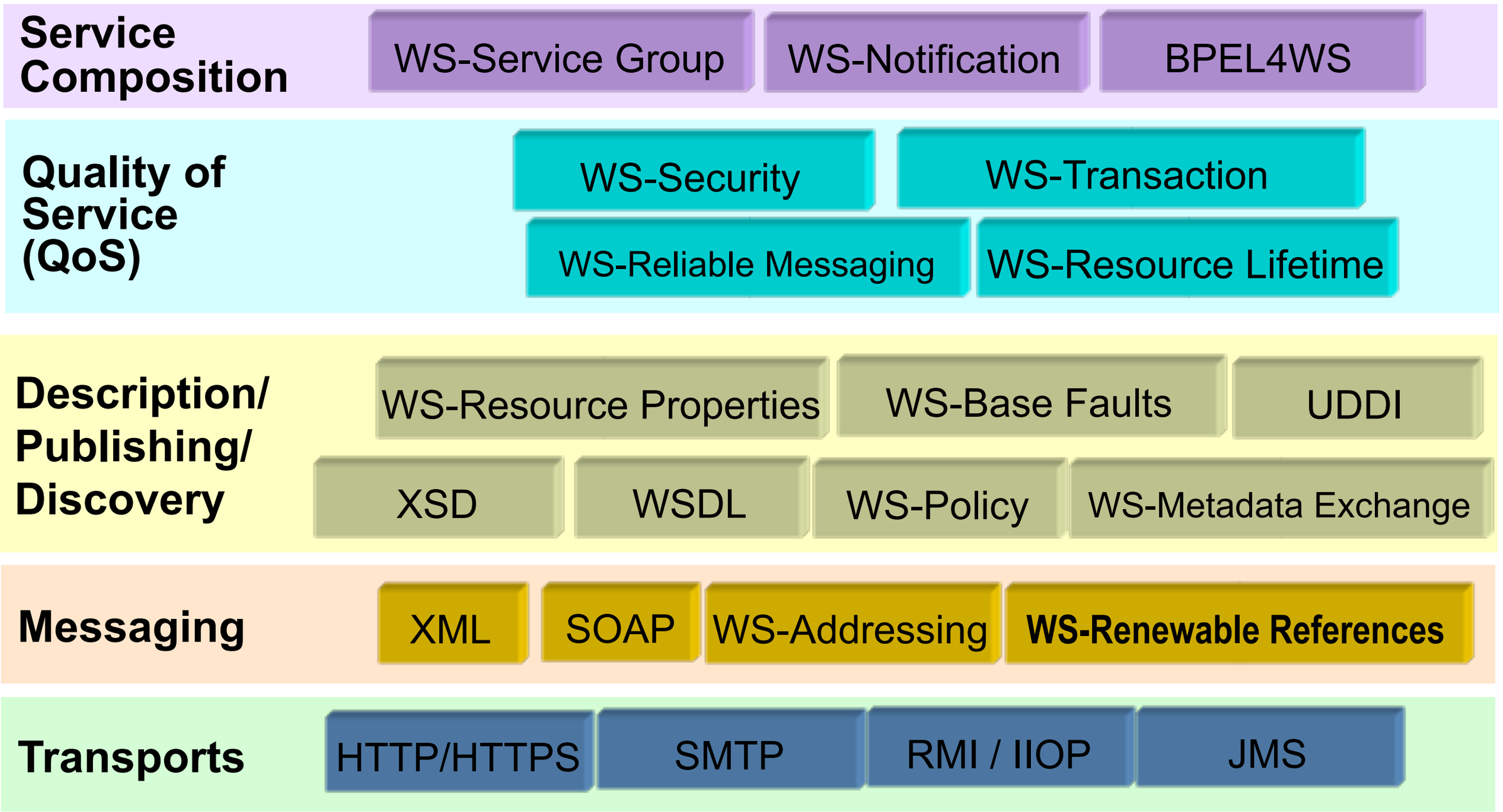
```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

The SOAP response:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPriceResponse>
    <m:Price>34.5</m:Price>
  </m:GetStockPriceResponse>
</soap:Body>

</soap:Envelope>
```

Previously ...

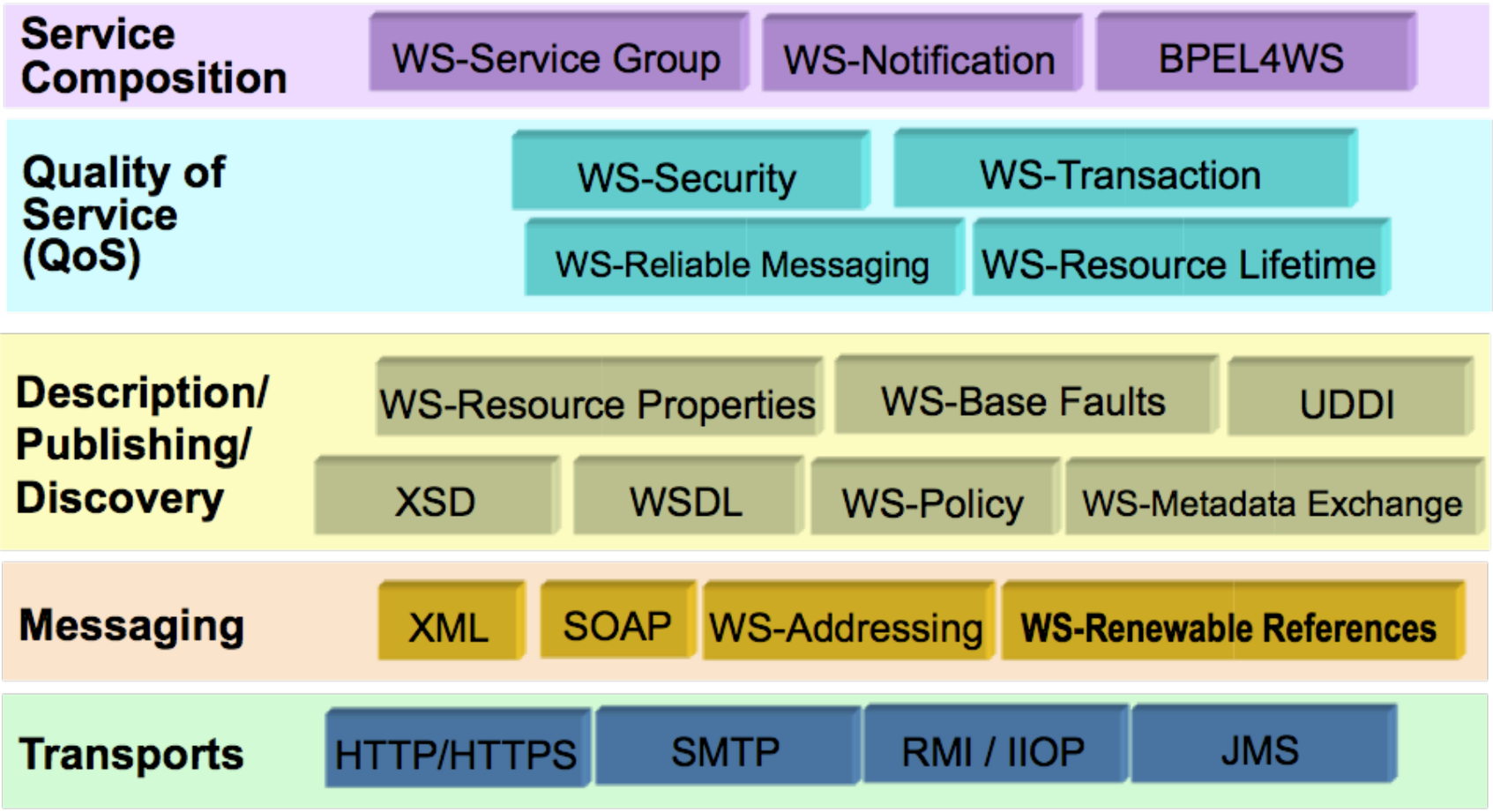
Web Services Standards Stack

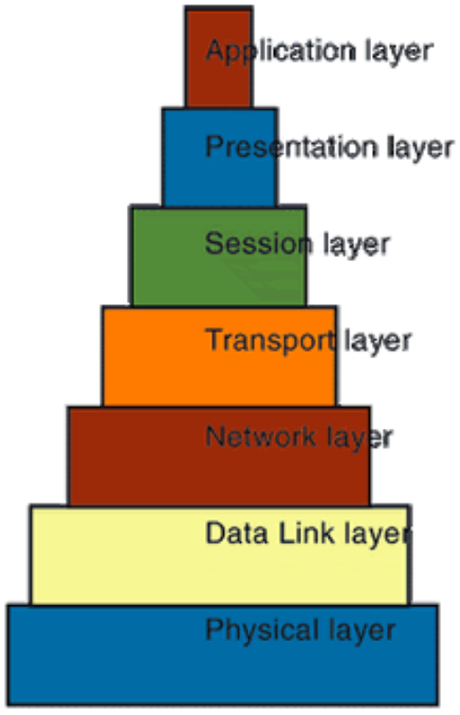


Source: Dr. Daniel Sabbah, Vice President of Strategy & Technology, IBM Software Group, Globus World 2004

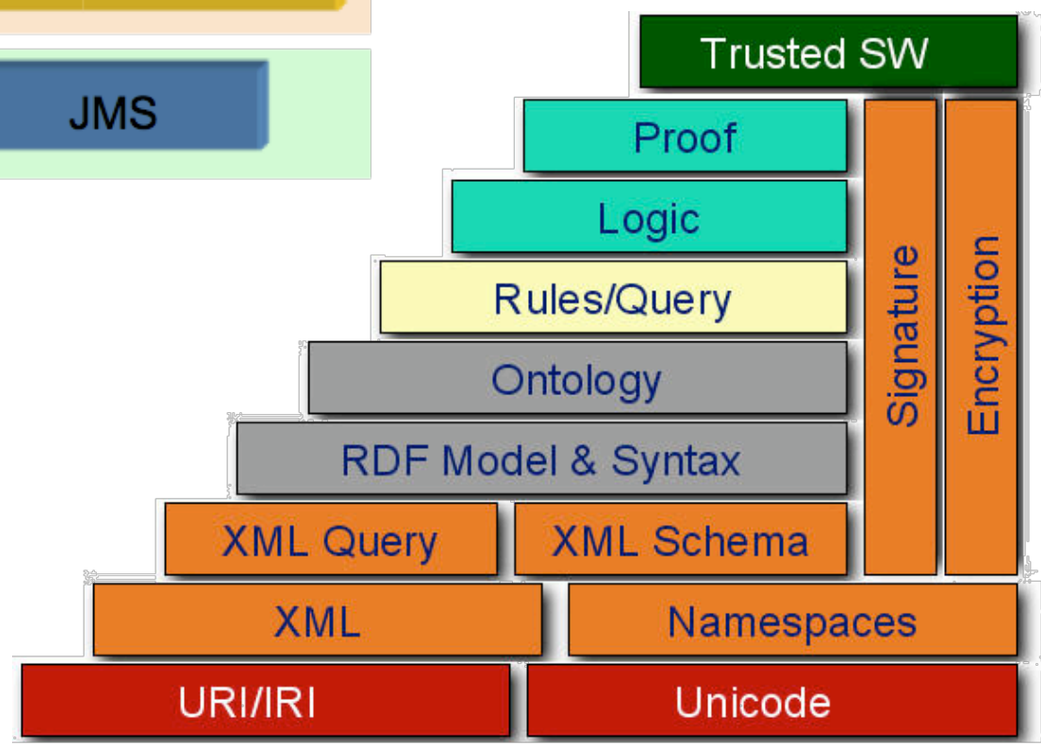
Previously ...

Web Services Standards Stack



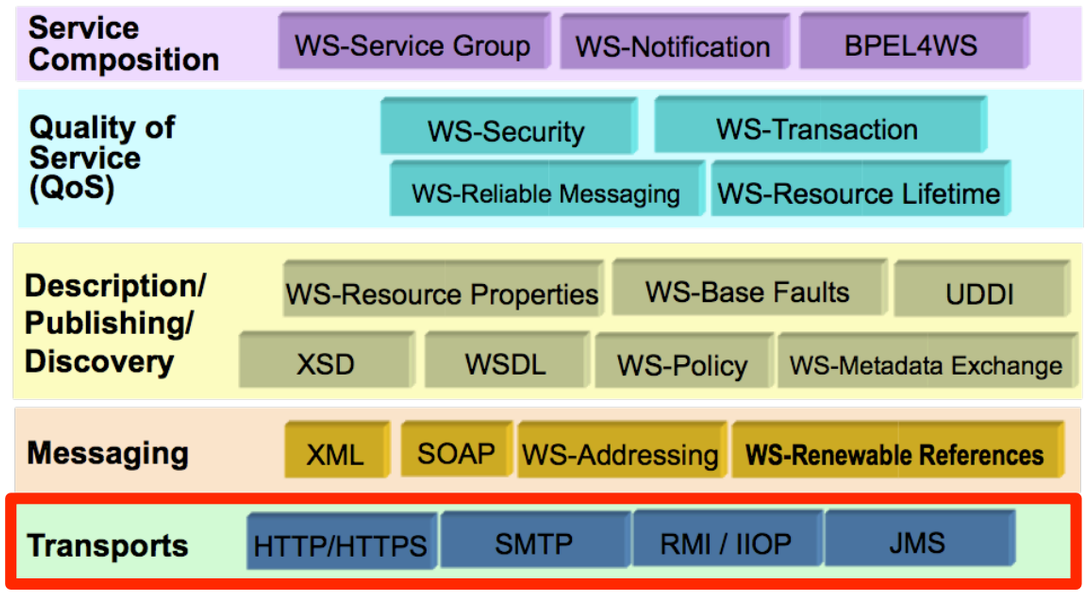


- Stacks are common
- Standardization Organizations for WSs
  - W3C, OASIS, WS-*
  - Protocols, formats, languages



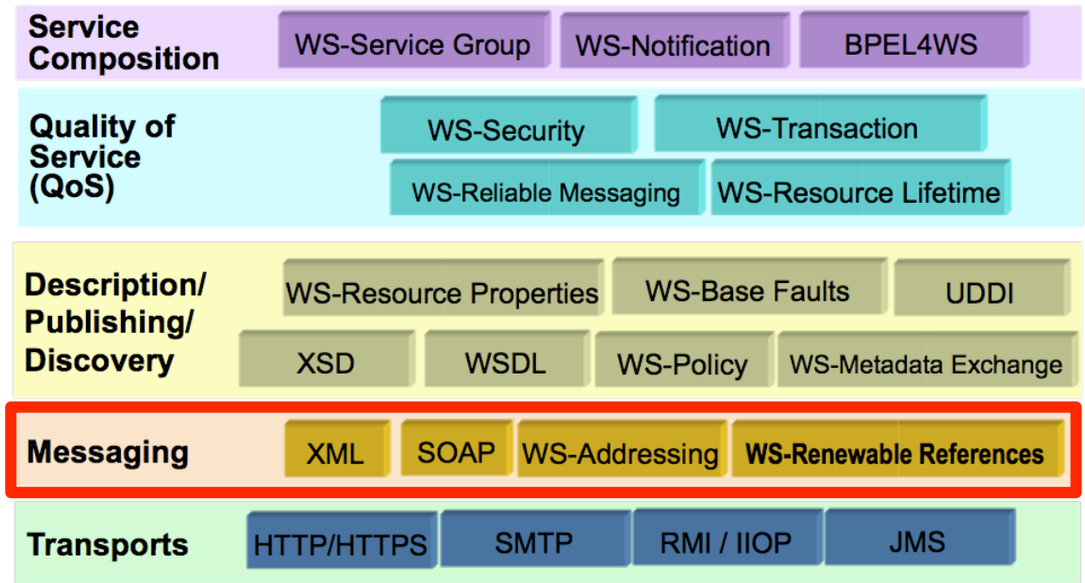
Transport Layer

- Invocation of distributed functionality
- Core communication mechanisms
- Typical protocols
  - HTTP/HTTPS
  - (SMTP)
  - (FTP)
- High compatibility with enterprise IT infrastructures



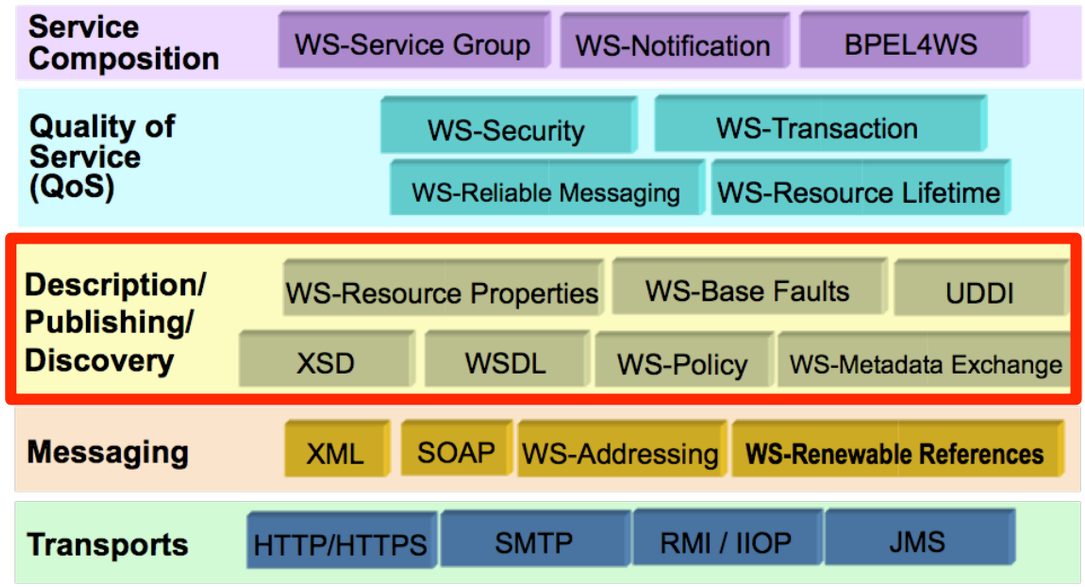
Messaging Layer

- Format of WS messages
  - Operations
  - Parameters
- Styles
  - SOAP (based on XML)
  - Anything (following REST paradigm)



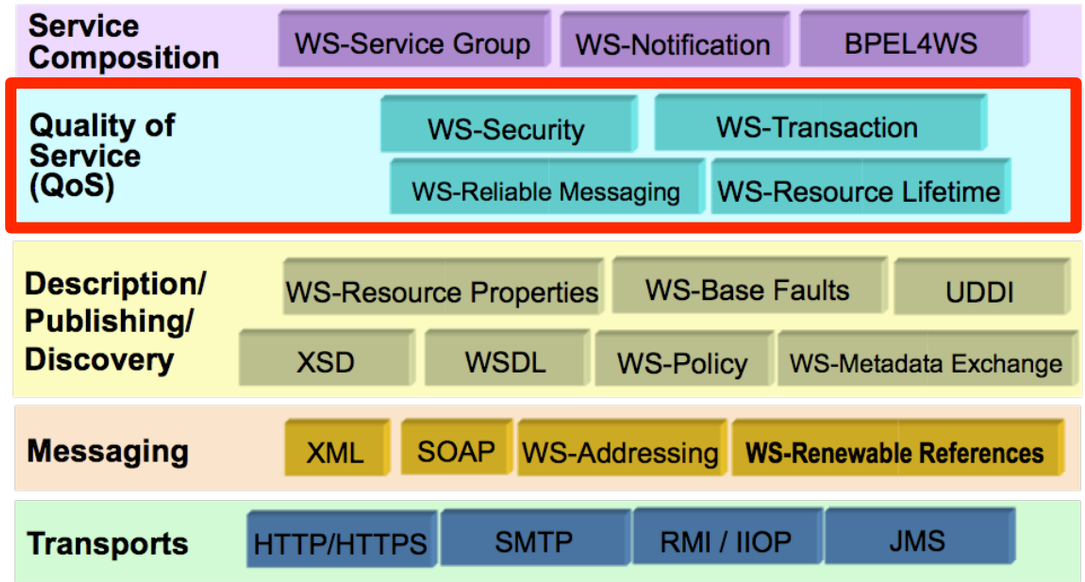
Description/Publishing/Discovery

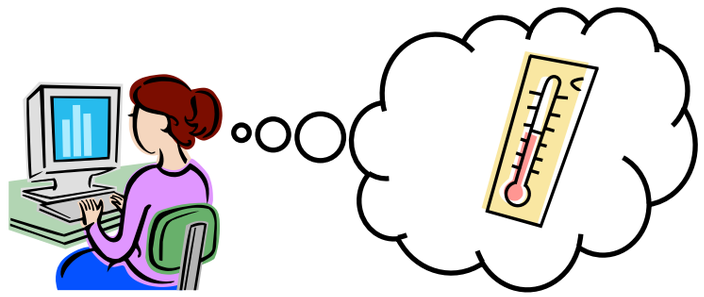
- Connecting different Web Service participants
  - Describe WS capabilities and messages
  - Offer WSs
  - Find WSs



Quality of Service

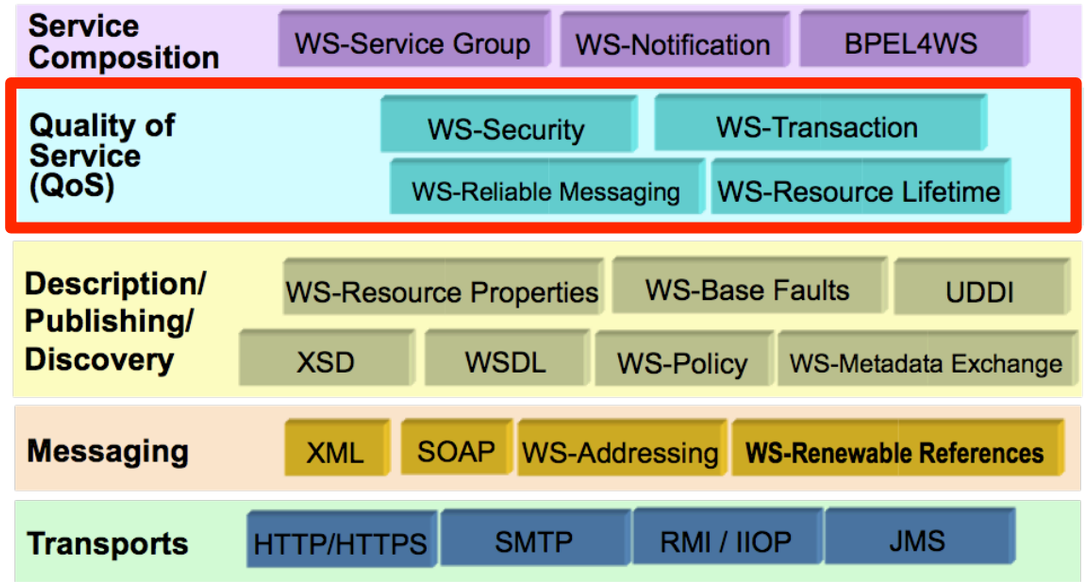
- Defines non-functional properties
  - Security (Authorization, Authentication, Confidentiality, Integrity)
  - Timing constraints/SLAs
  - Costs

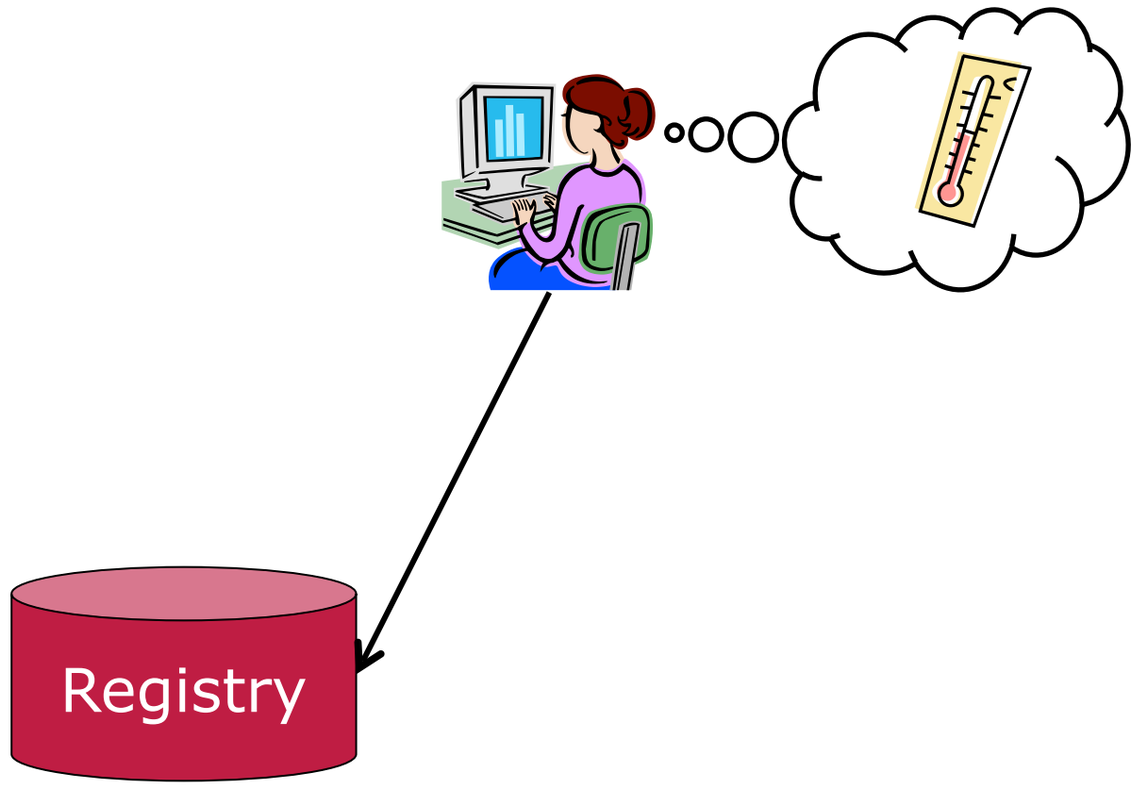




Quality of Service

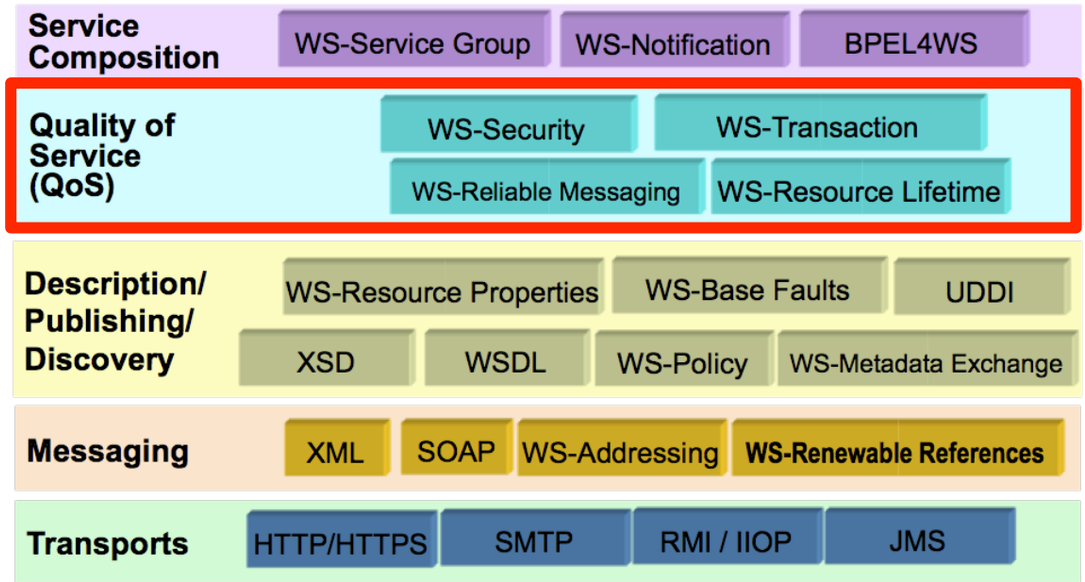
- Defines non-functional properties
  - Security (Authorization, Authentication, Confidentiality, Integrity)
  - Timing constraints/SLAs
  - Costs

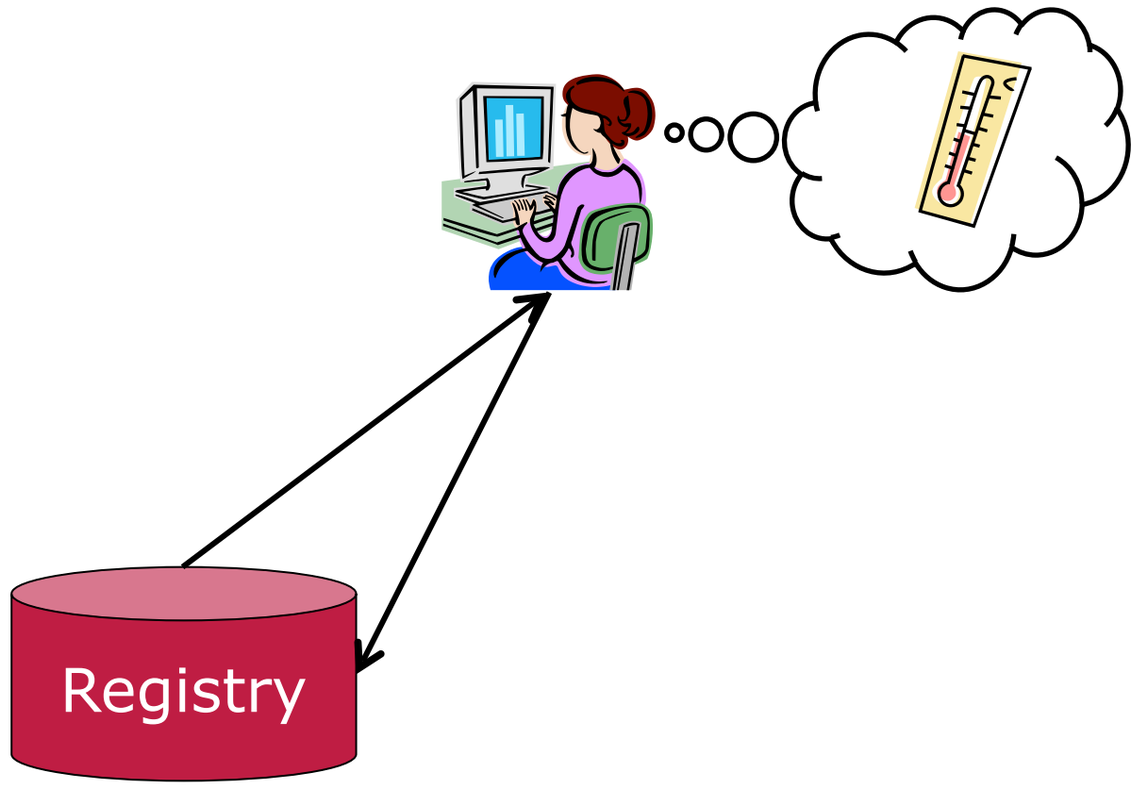




Quality of Service

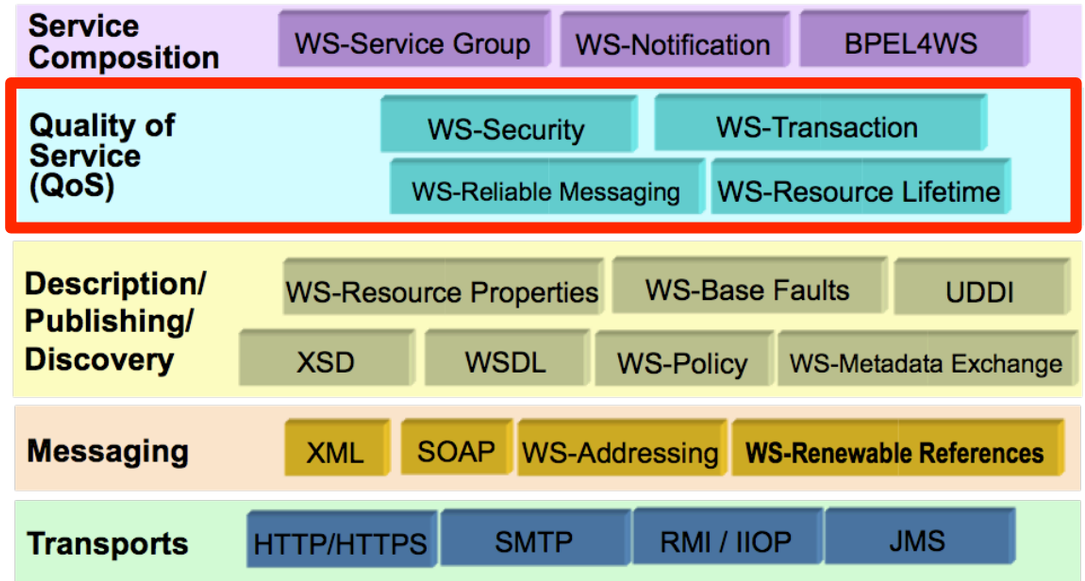
- Defines non-functional properties
  - Security (Authorization, Authentication, Confidentiality, Integrity)
  - Timing constraints/SLAs
  - Costs

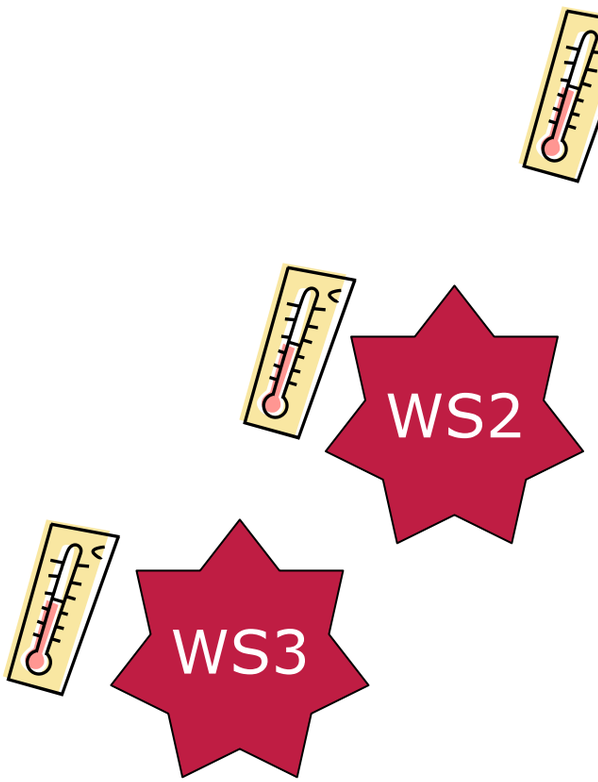


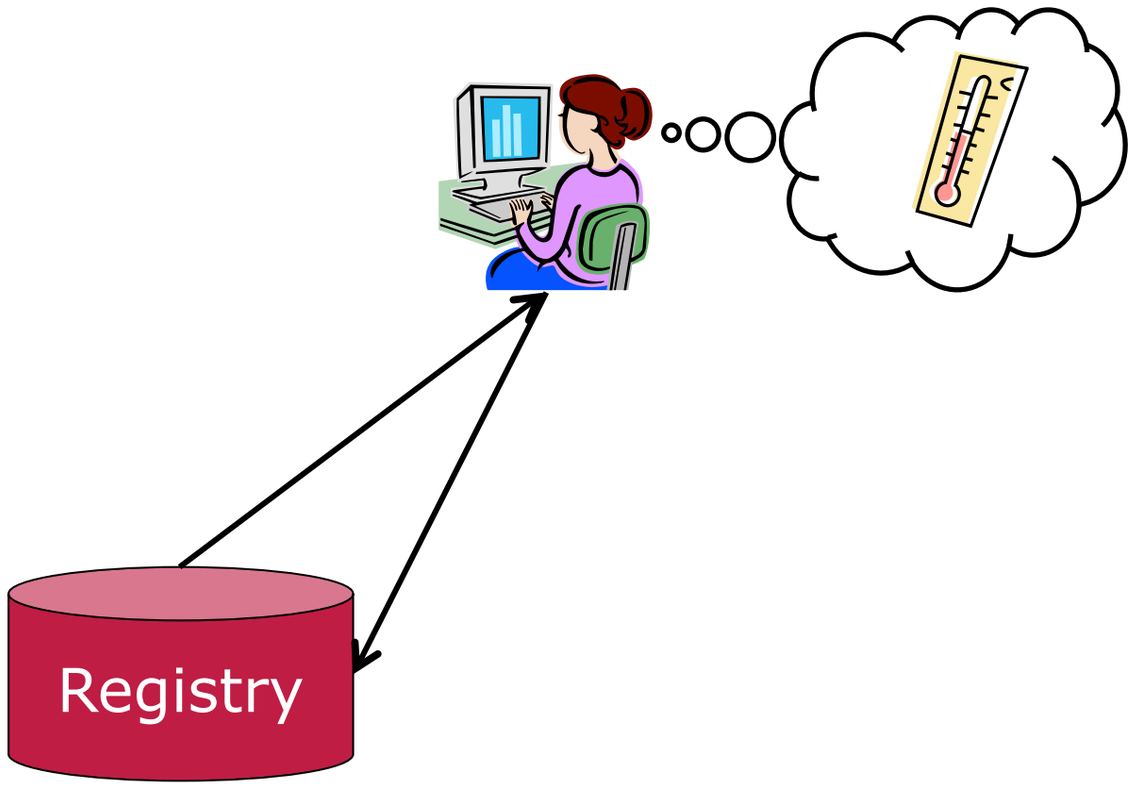


Quality of Service

- Defines non-functional properties
  - Security (Authorization, Authentication, Confidentiality, Integrity)
  - Timing constraints/SLAs
  - Costs

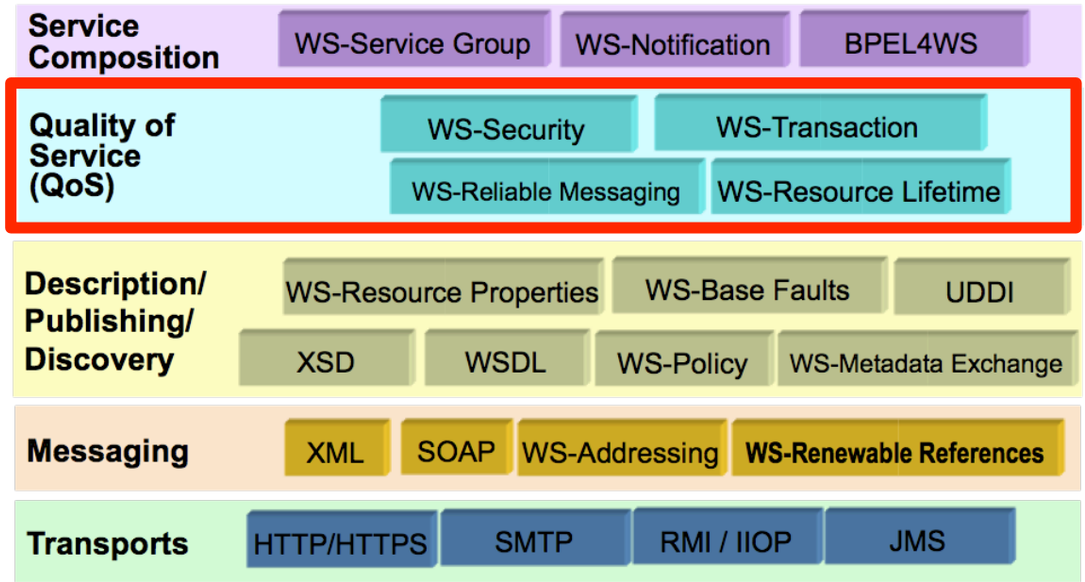


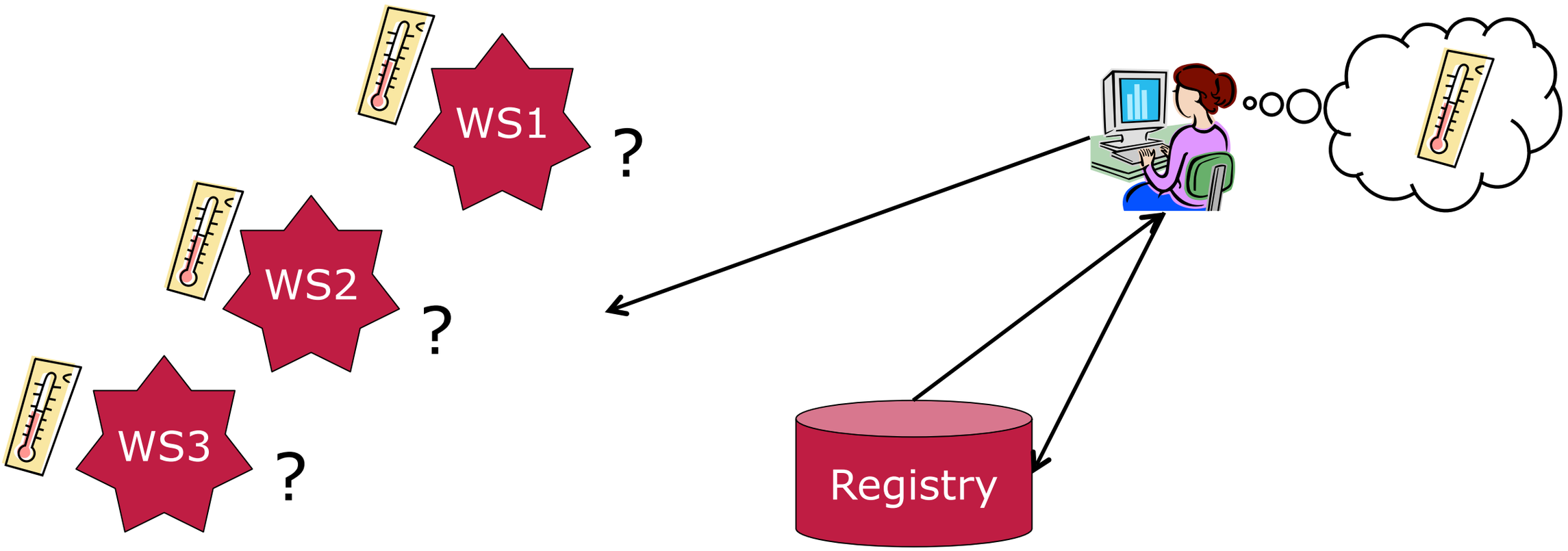




Quality of Service

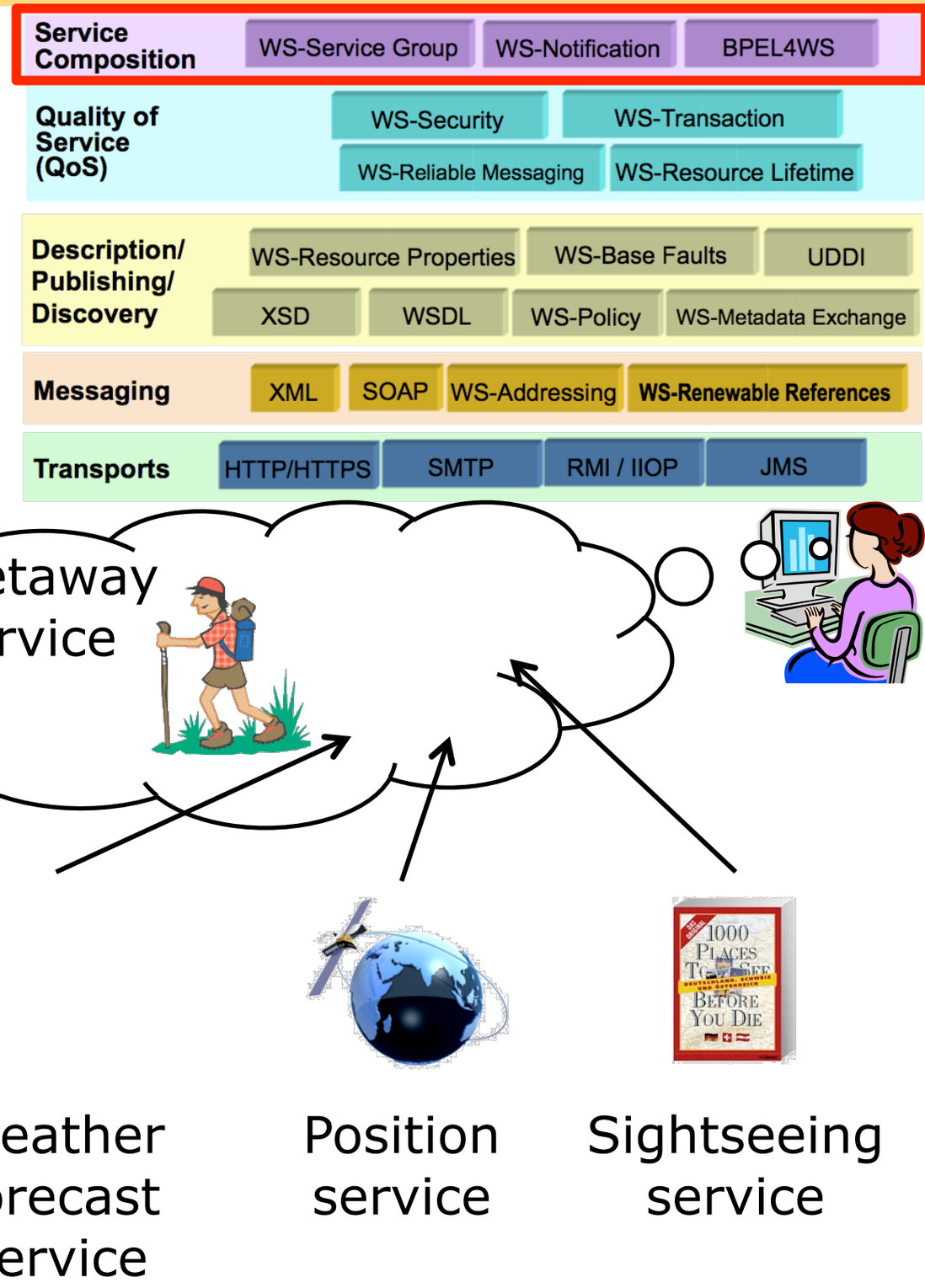
- Defines non-functional properties
  - Security (Authorization, Authentication, Confidentiality, Integrity)
  - Timing constraints/SLAs
  - Costs





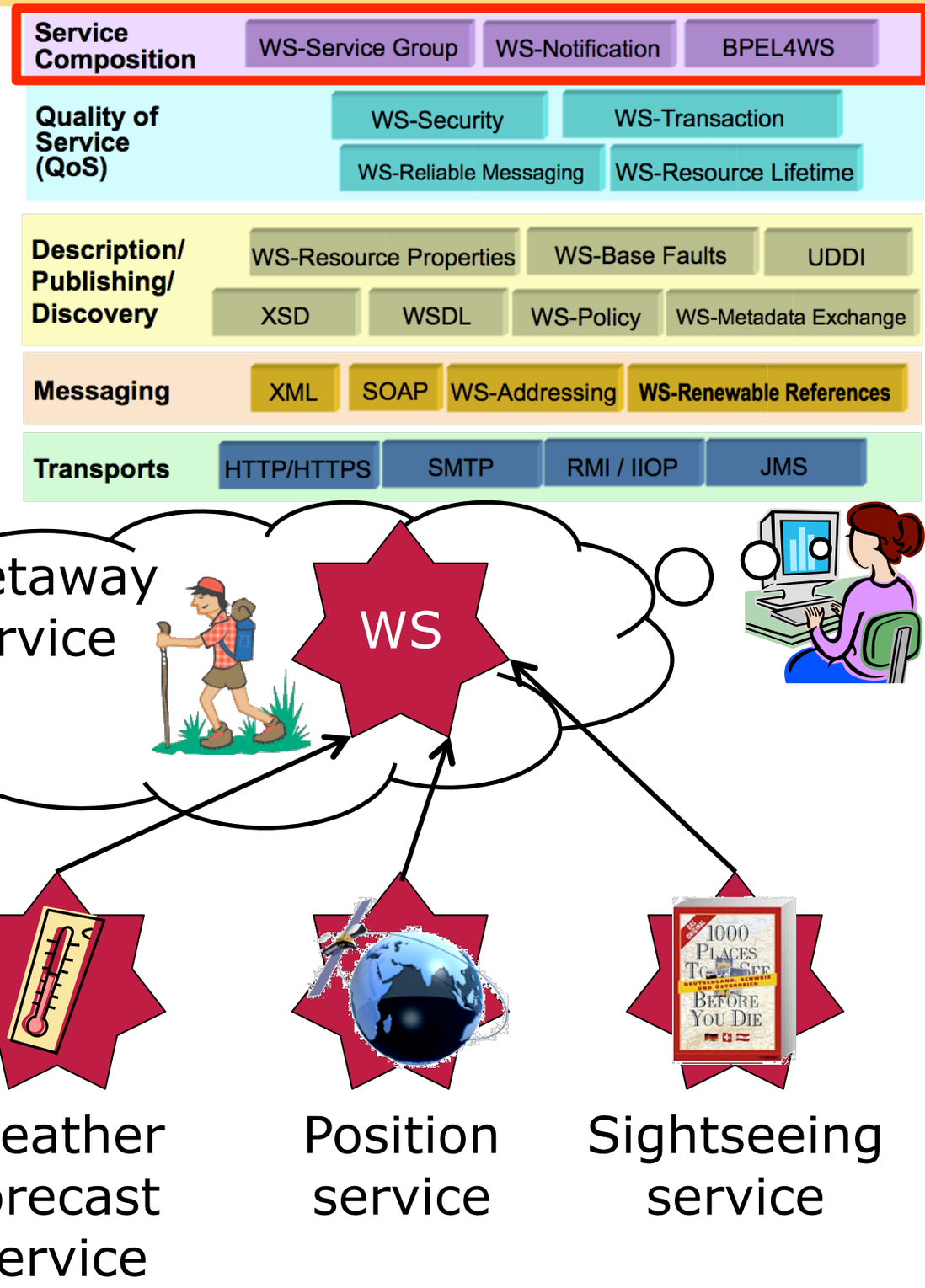
Service Composition

- Create higher-level service compositions out of existing functionality
  - Sequences of message exchanges
  - Flow of control (workflow)
- Approaches
  - Hard-coding
  - Tool support/modeling
    - BPEL
    - WS-Notification



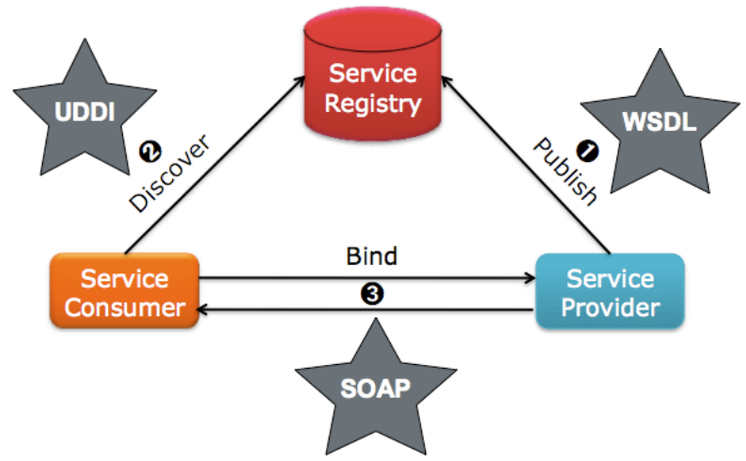
Service Composition

- Create higher-level service compositions out of existing functionality
  - Sequences of message exchanges
  - Flow of control (workflow)
- Approaches
  - Hard-coding
  - Tool support/modeling
    - BPEL
    - WS-Notification

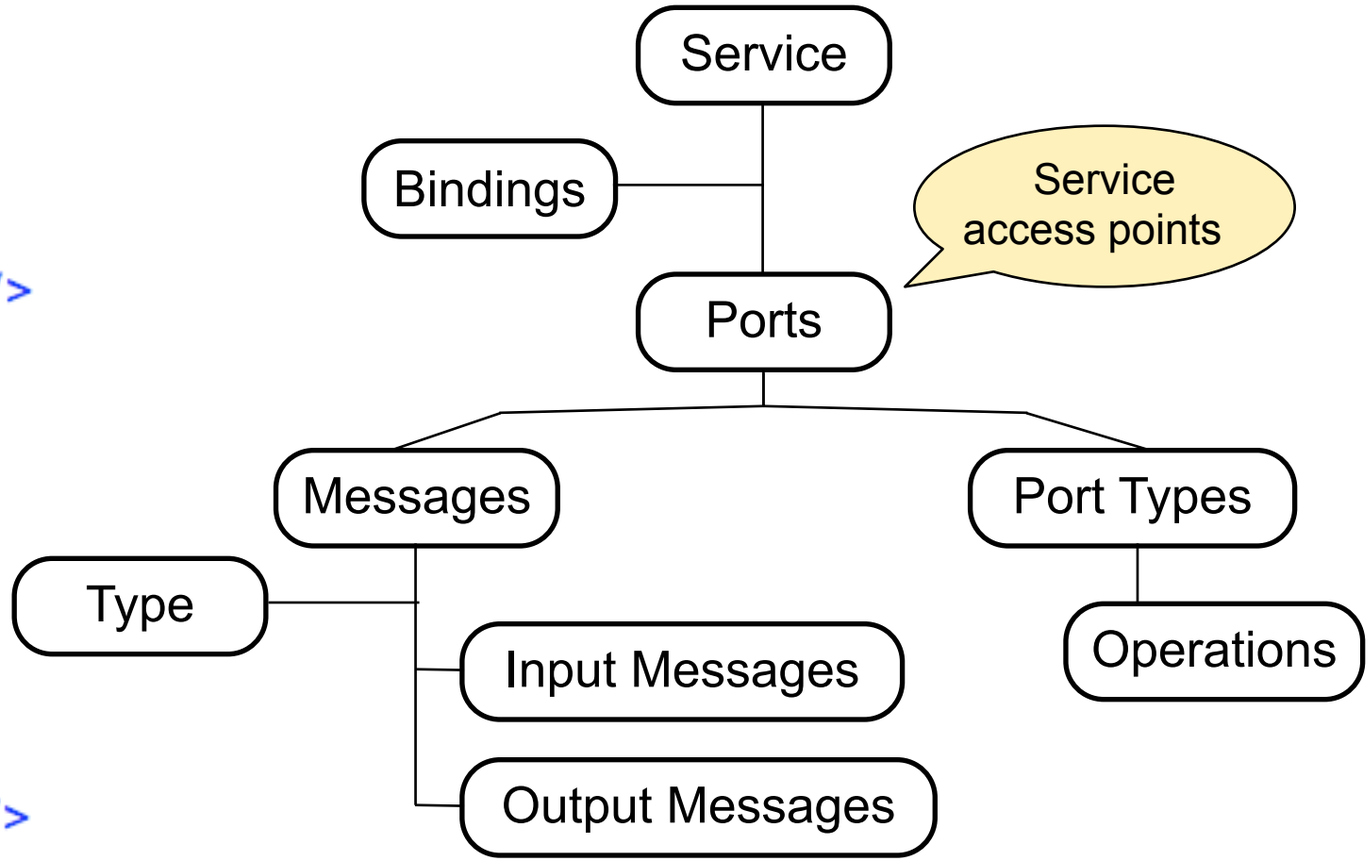


Web Service Description Language WSDL

- "Web Service Description Language"
- XML-based
- Answers these questions:
  - *What* is the service about?
  - *Where* does it reside?
  - *How* can it be invoked?
- Constructs:




```
1  <message name ="getPriceRequest">
2     <part name="productid" type="xs:string" />
3  </message>
4
5  <message name="getPriceResponse">
6     <part name="value" type="xs:string" />
7  </message>
8
9
10 <portType name="productPrice">
11    <operation name="getPrice">
12       <input message="getPriceRequest" />
13       <output message="getPriceResponse" />
14    </operation>
15 </portType>
```

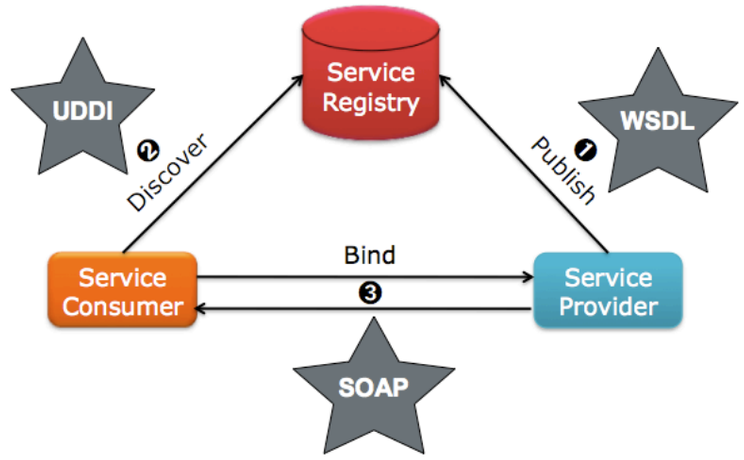


Web Service Description Language WSDL

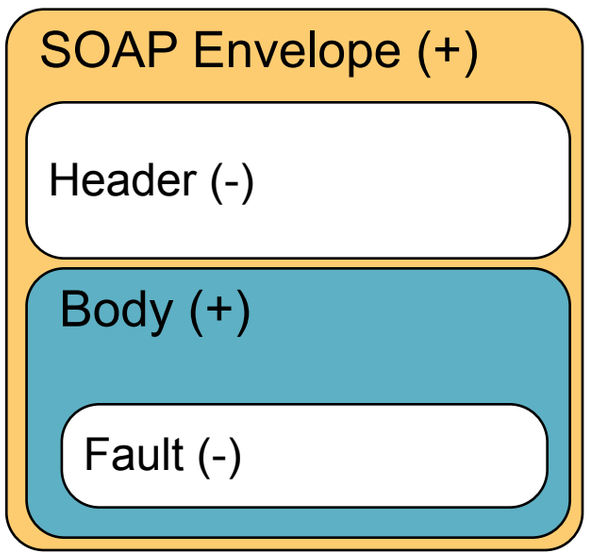

```xml
<message name ="getPriceRequest">
   <part name="productid" type="xs:string" />
</message>

<message name="getPriceResponse">
   <part name="value" type="xs:string" />
</message>


<portType name="productPrice">
   <operation name="getPrice">
      <input message="getPriceRequest" />
      <output message="getPriceResponse" />
   </operation>
</portType>
```

```xml
<types>
   <schema targetnamespace="http://servicescomputing.org/ProductInfo"
      xmlns="http://www.w3.org/2001/XMLSchema"
      xmlns:wsdl=http://servicescomputing.org/wsdl/ >

      <xs:element name="id" type="xsd:string" />
      <xs:element name="name" type="xsd:string" />
      <xs:element name="vendor" type="xsd:string" />
      <xs:complexType name="ProductInfo">
         <xs:sequence>
            <xs:element ref="tns:id" />
            <xs:element ref="tns:name" />
            <xs:element ref="tns:vendor" />
         </xs:sequence/>
      </xs:complexType>
   </schema>
</types>

<message name ="getPriceRequest">
   <part name="productInfo" type="xs:ProductInfo" />
</message>

<message name="getPriceResponse">
   <part name="value" type="xs:string" />
</message>


<portType name="productPrice">
   <operation name="getPrice">
      <input message="getPriceRequest" />
      <output message="getPriceResponse" />
   </operation>
</portType>
```

Web Service Communication Protocol SOAP



- "Simple Object Access Protocol"
- Structured and typed information exchange
- XML-based
- Bound to transport protocol (HTTP/S, SMTP, etc)
- Interaction patterns:
  - Remote Procedure Call (RPC): Synchronous request/response
  - Document-Oriented: Asynchronous
- SOAP Message Constructs
  - Envelope: required
  - Header: optional
  - Body: required
  - Fault: optional



Web Service Communication Protocol SOAP

XML

Header

Body

```
<?xml version ="1.0" ?>

<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
    soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

    <soap:Header>
     <m:Payment xmlns:m="http://www.servicescomputing.org/payment/"
        soap:actor="http://www.servicescomputing.org/appml"
        soap:mustUnderstand="1">
           123
     </m:Payment>
    </soap:Header>

    <soap:Body>
    <!-- Request message-->
     <m:GetPrice xmlns:m="http://www.servicescomputing.org/prices">
        <m:Item> Product </m:Item>
     </m:GetPrice>

     <soap:Fault>
     ....
     </soap:Fault>

    </soap:Body>

</soap:Envelope>
```

Web Service Communication Protocol SOAP

XML

Header

Body

```
<?xml version ="1.0" ?>

<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
    soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

    <soap:Header>
     <m:Payment xmlns:m="http://www.servicescomputing.org/payment/"
        soap:actor="http://www.servicescomputing.org/appml"
        soap:mustUnderstand="1">
           123
     </m:Payment>
    </soap:Header>

    <soap:Body>
    <!-- Request message-->
    <m:GetPrice xmlns:m="http://www.servicescomputing.org/prices">
        <m:Item> Product </m:Item>
    </m:GetPrice>

     <soap:Fault>
     ....
     </soap:Fault>

    </soap:Body>

</soap:Envelope>
```

```
<!-- Response message-->
 <m:GetPriceResponse xmlns:m="http://www.servicescomputing.org/prices">
    <m:Price> 70 </m:Price>
 </m:GetPriceResponse>
```

Web Service Communication Protocol SOAP

SOAP-HTTP Binding

XML

Header

Body

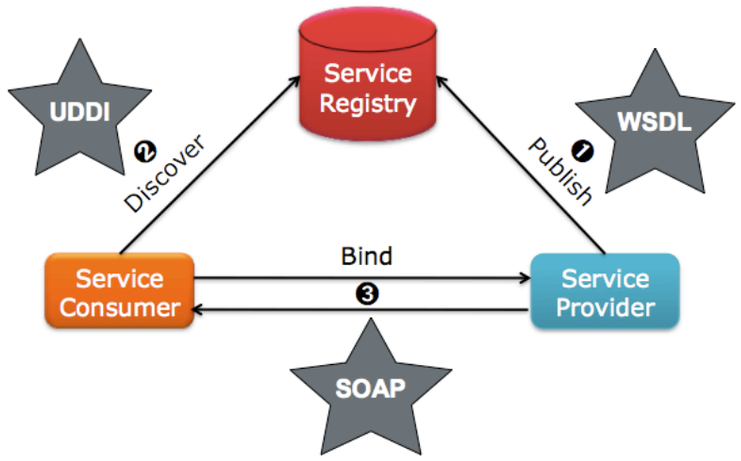
```
POST / InServicesComputing HTTP/1.1
Host: www.servicescomputing.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 100

<?xml version ="1.0" ?>

<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
    soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

    <soap:Header>
     <m:Payment xmlns:m="http://www.servicescomputing.org/payment/"
       soap:actor="http://www.servicescomputing.org/appml"
       soap:mustUnderstand="1">
          123
     </m:Payment>
    </soap:Header>

    <soap:Body>
    <!-- Request message-->
    <m:GetPrice xmlns:m="http://www.servicescomputing.org/prices">
       <m:Item> Product </m:Item>
    </m:GetPrice>

     <soap:Fault>
     ....
     </soap:Fault>

    </soap:Body>

</soap:Envelope>
```

```
<!-- Response message-->
 <m:GetPriceResponse xmlns:m="http://www.servicescomputing.org/prices">
    <m:Price> 70 </m:Price>
 </m:GetPriceResponse>
```

WSDL-SOAP Binding


```
<message name="getPriceRequest">
   <part name="productid" type="xs:string"/>
</message>

<message name="getPriceResponse">
   <part name="value" type="xs:string" />
</message>

<portType name="productPrice">
   <operation name="getPrice">
      <input message="getPriceRequest" />
      <output mesage="getPriceResponse" />
   </operation>
</portType>


<binding type="productPrice" name="b1">
   <soap:binding style="rpc" transport="http://schema.cmsoap.org/soap/http/" />
   <operation>
      <soap:operation soapAction="http://servicesComputing.org/getPrice" />
      <input>
         <soap:body use="literal" />
      </input>

      <output>
         <soap:body use="literal" />
      </output>
   </operation>
</binding>
```

Publishing a Web Service in Registry UDDI



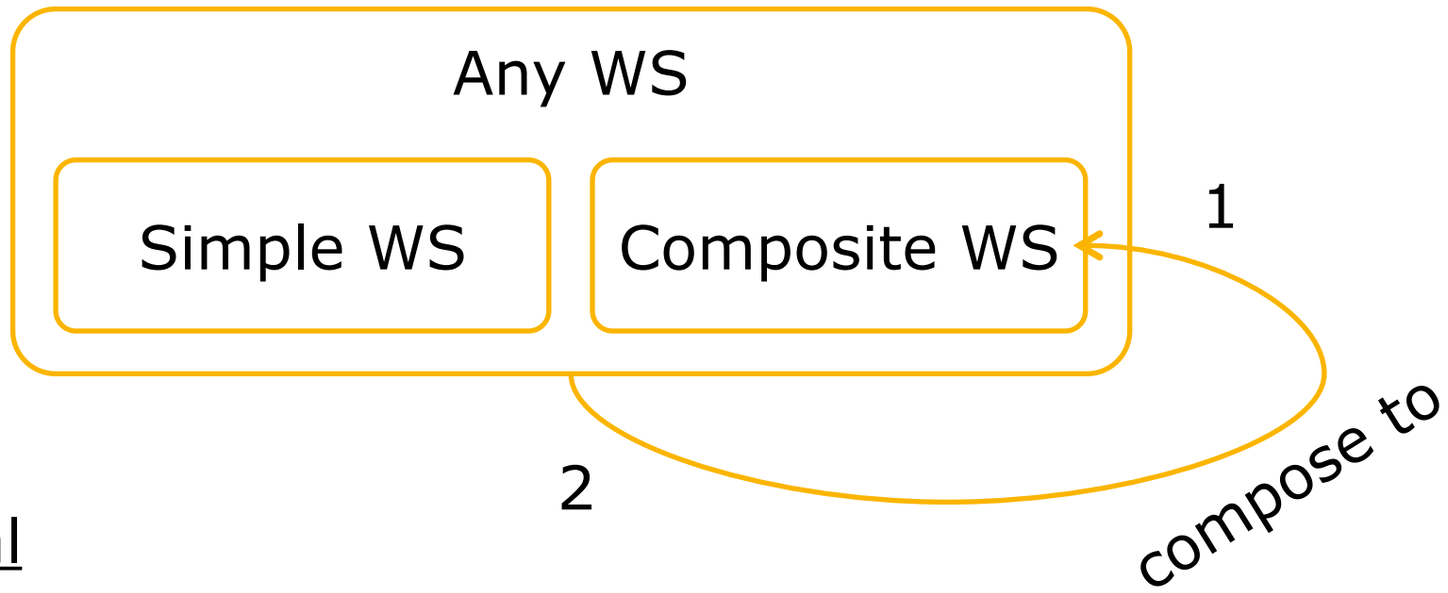
- "Universal Description, Discovery, and Integration"
- XML-based

- Answers these questions:
  - *Who*: information about a business, such as name, contact.
  - *What*: classification information about industry, products, registered Web Services
  - *Where*: registration information, such as URL
  - *How*: registration references about interfaces

```
1 <discoveryURL useType="businessEntity">
2     http://www.servicescomputing.org?
3     businessKey=uddi:servicescomputing.org:registry:sales:100
4 </discoveryURL>
```

Composite Web Services

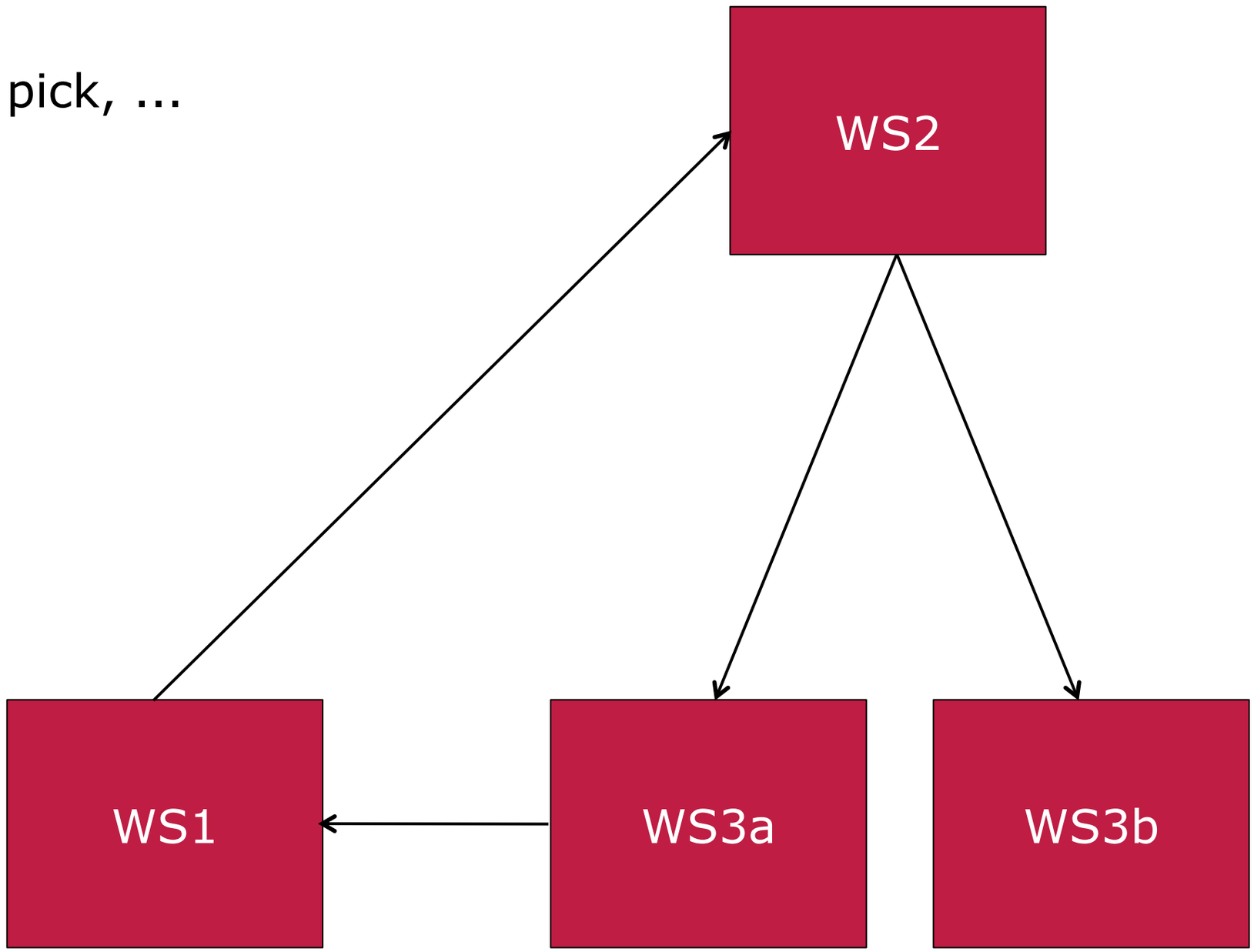
- Complex business logic requires functionality of several WSs.



- Approaches
  - Static vs. dynamic
  - Automated vs. manual
  - Model driven vs. business rule driven vs. declarative composition
- Business Process Execution Language (BPEL, BPEL4S, WS-BPEL)
  - Language
  - Development Environment
  - Runtime Environment

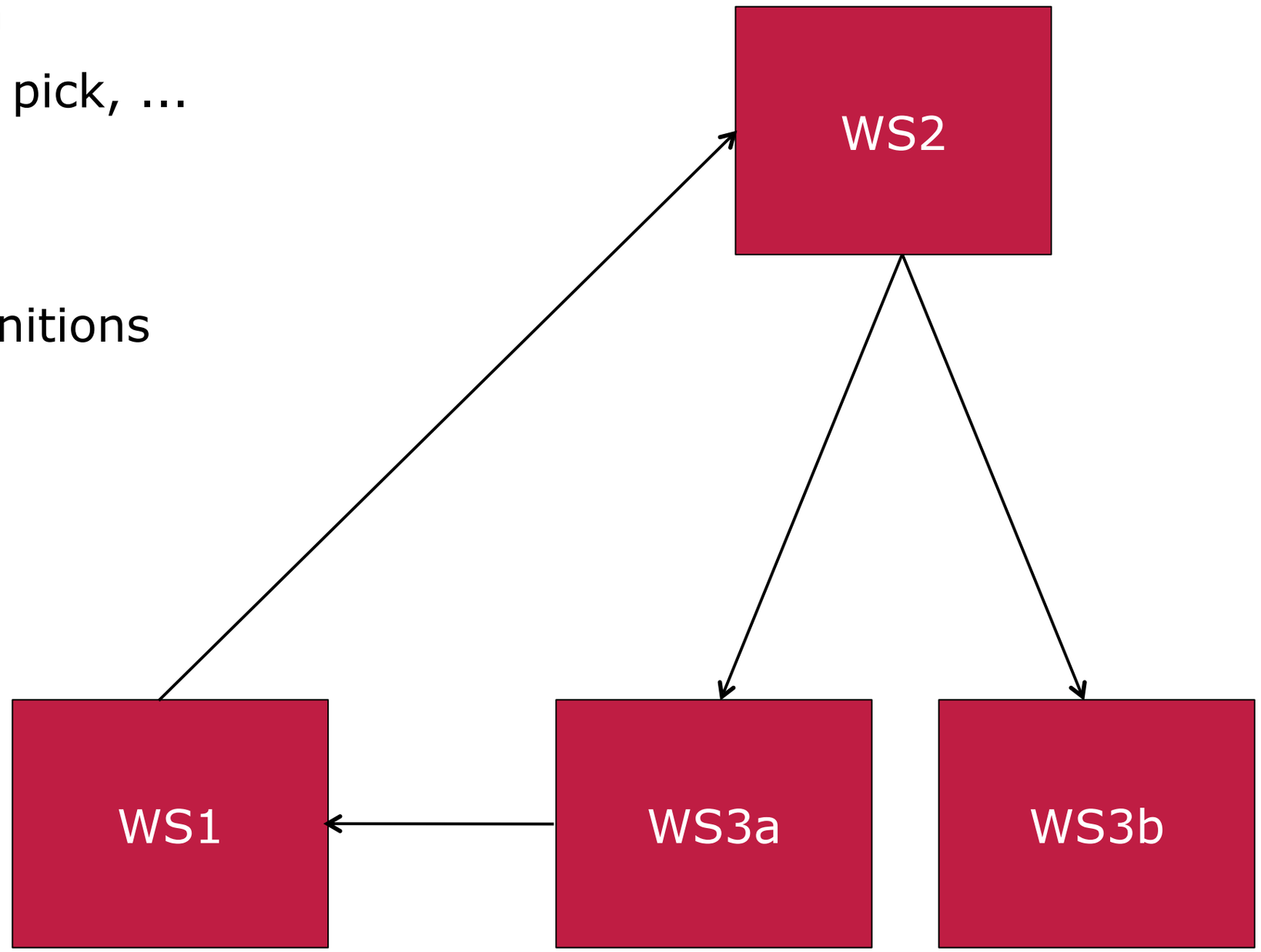
BPEL

- Workflow definition
  - Sequence, flow, pick, ...
- XML based
- 3 sections



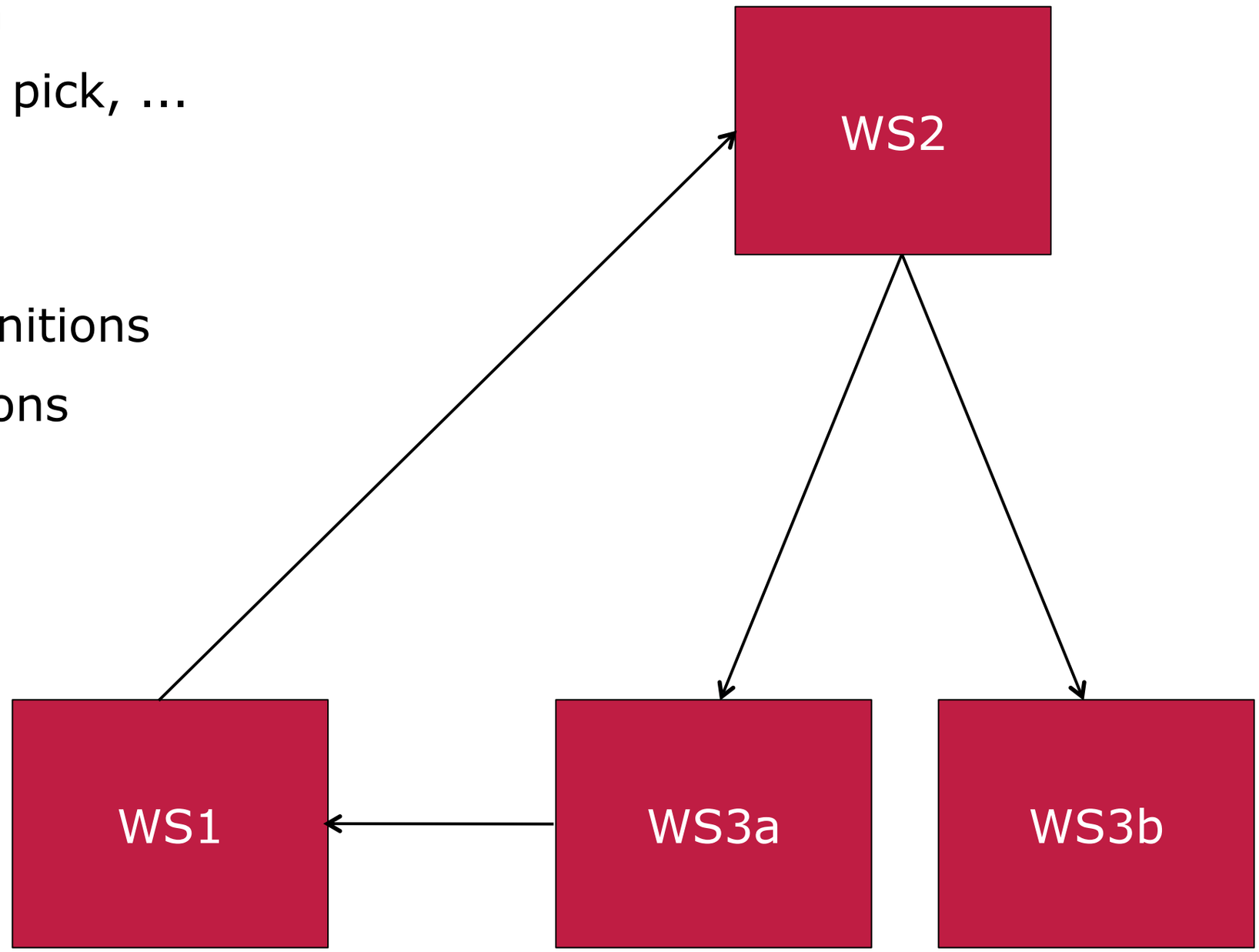
BPEL

- Workflow definition
  - Sequence, flow, pick, ...
- XML based
- 3 sections
  - Partner link definitions



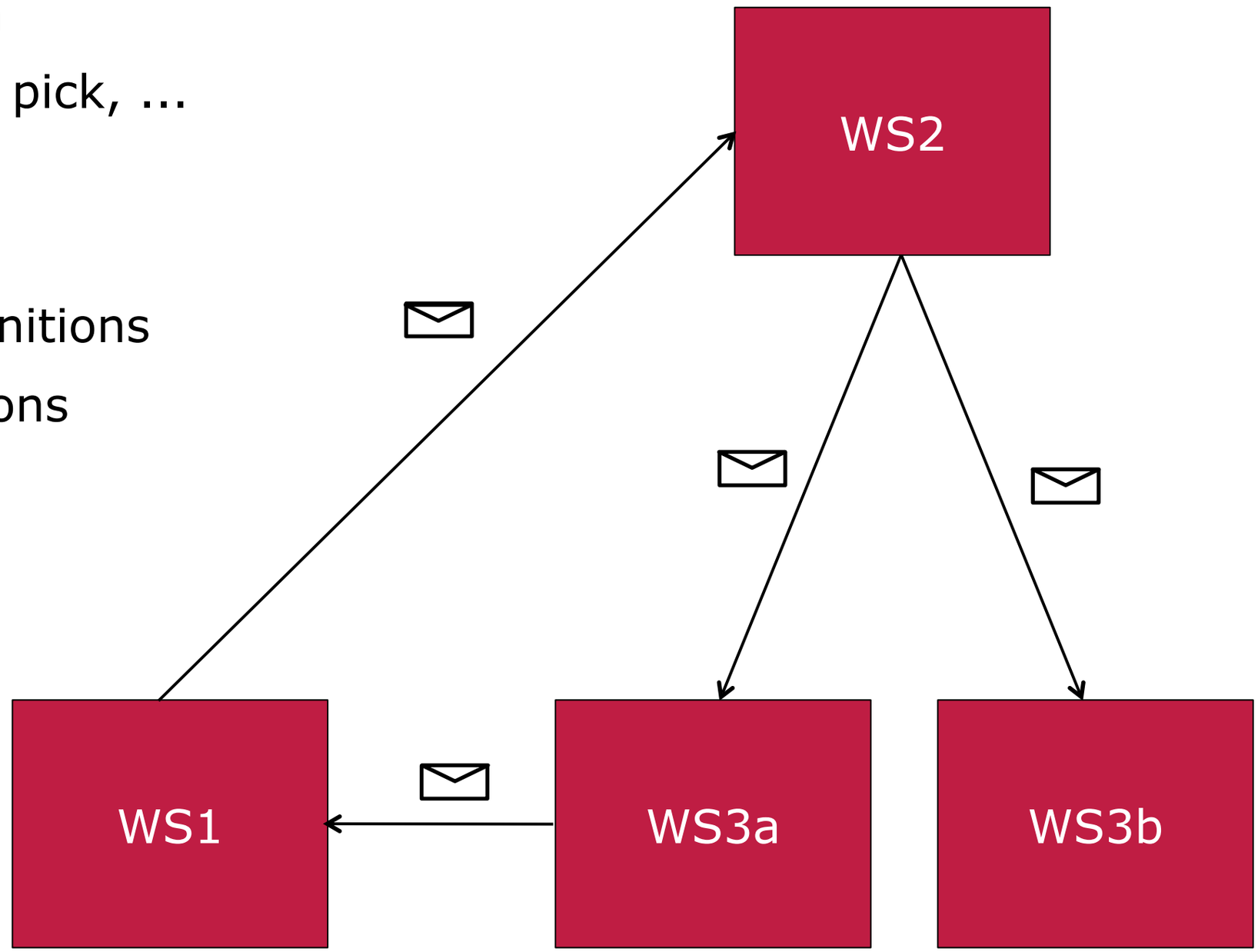
BPEL

- Workflow definition
  - Sequence, flow, pick, ...
- XML based
- 3 sections
  - Partner link definitions
  - Variable definitions



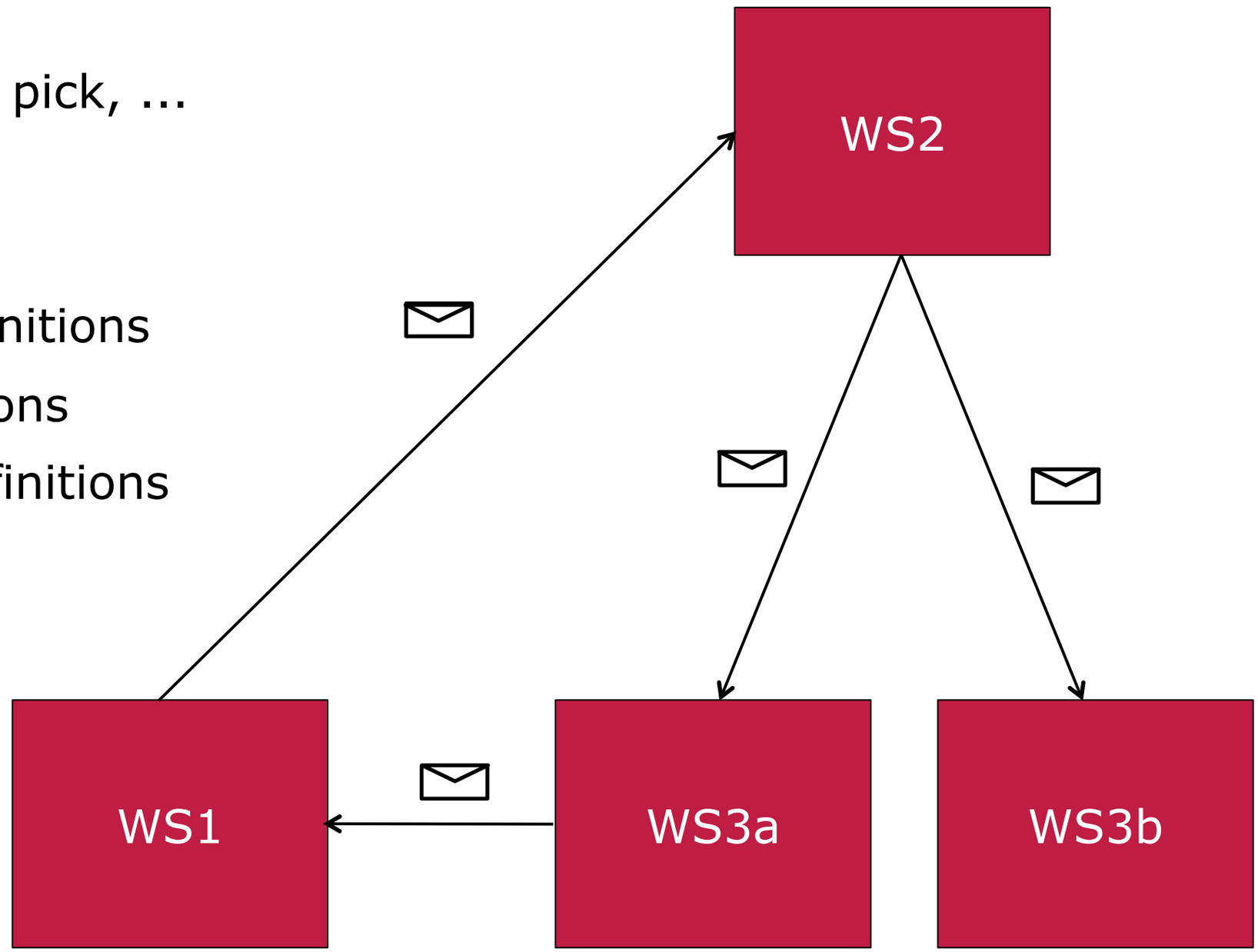
BPEL

- Workflow definition
  - Sequence, flow, pick, ...
- XML based
- 3 sections
  - Partner link definitions
  - Variable definitions



BPEL

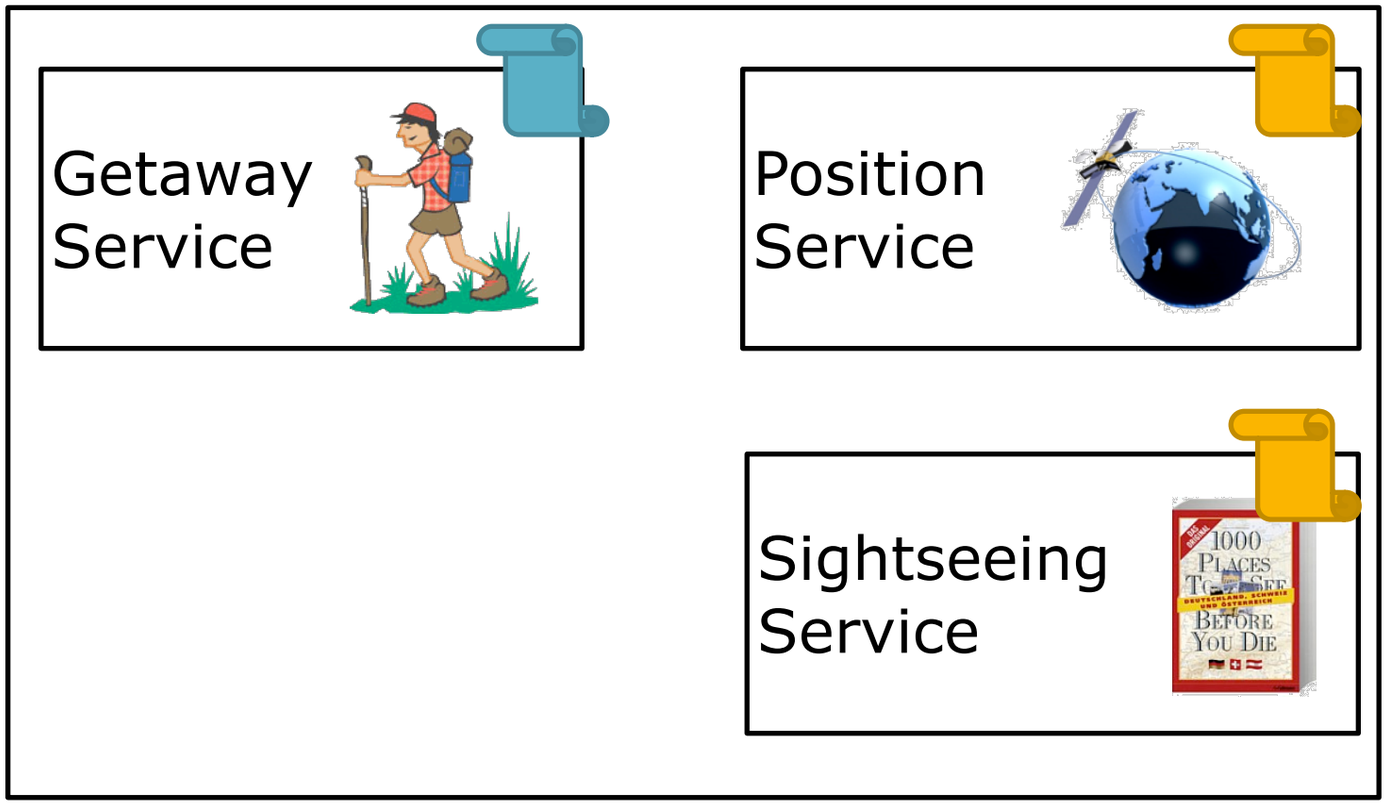
- Workflow definition
  - Sequence, flow, pick, ...
- XML based
- 3 sections
  - Partner link definitions
  - Variable definitions
  - Process flow definitions



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

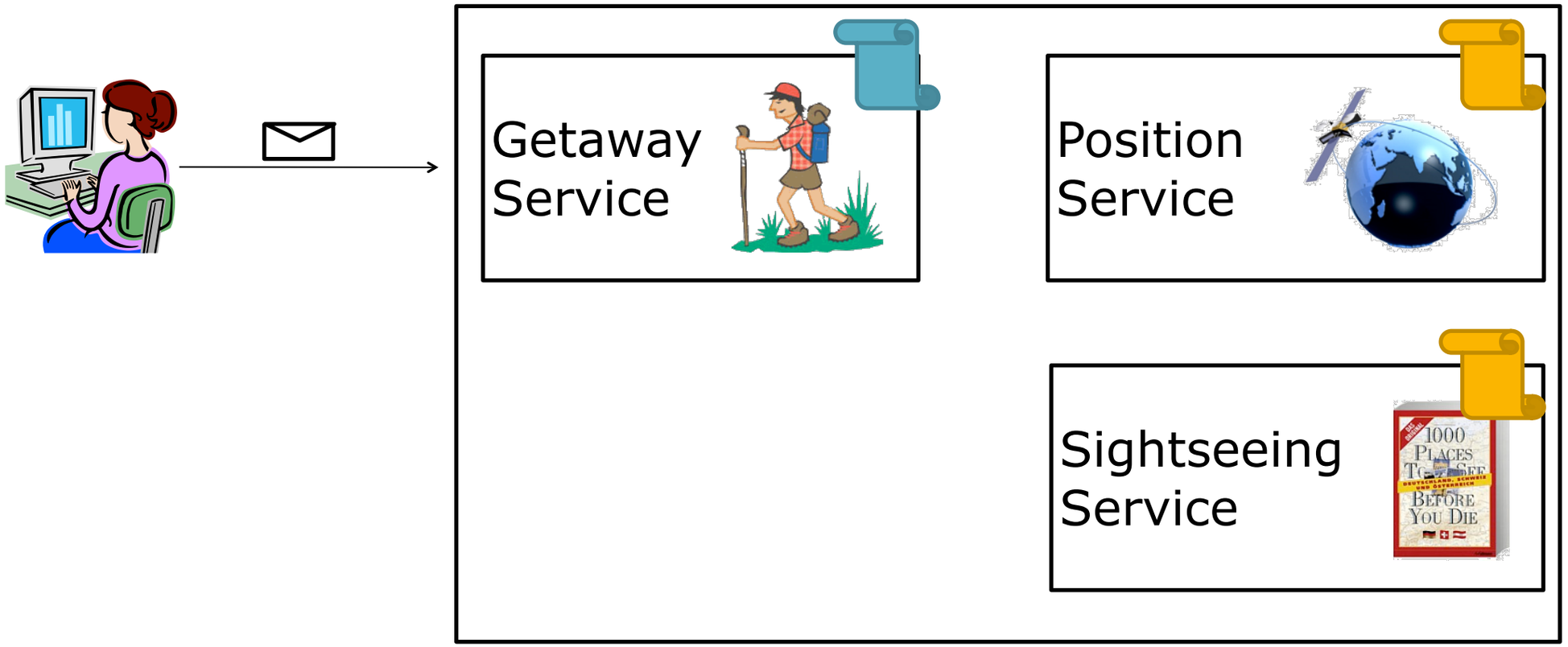
    ➔ Finds sights near the current position



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

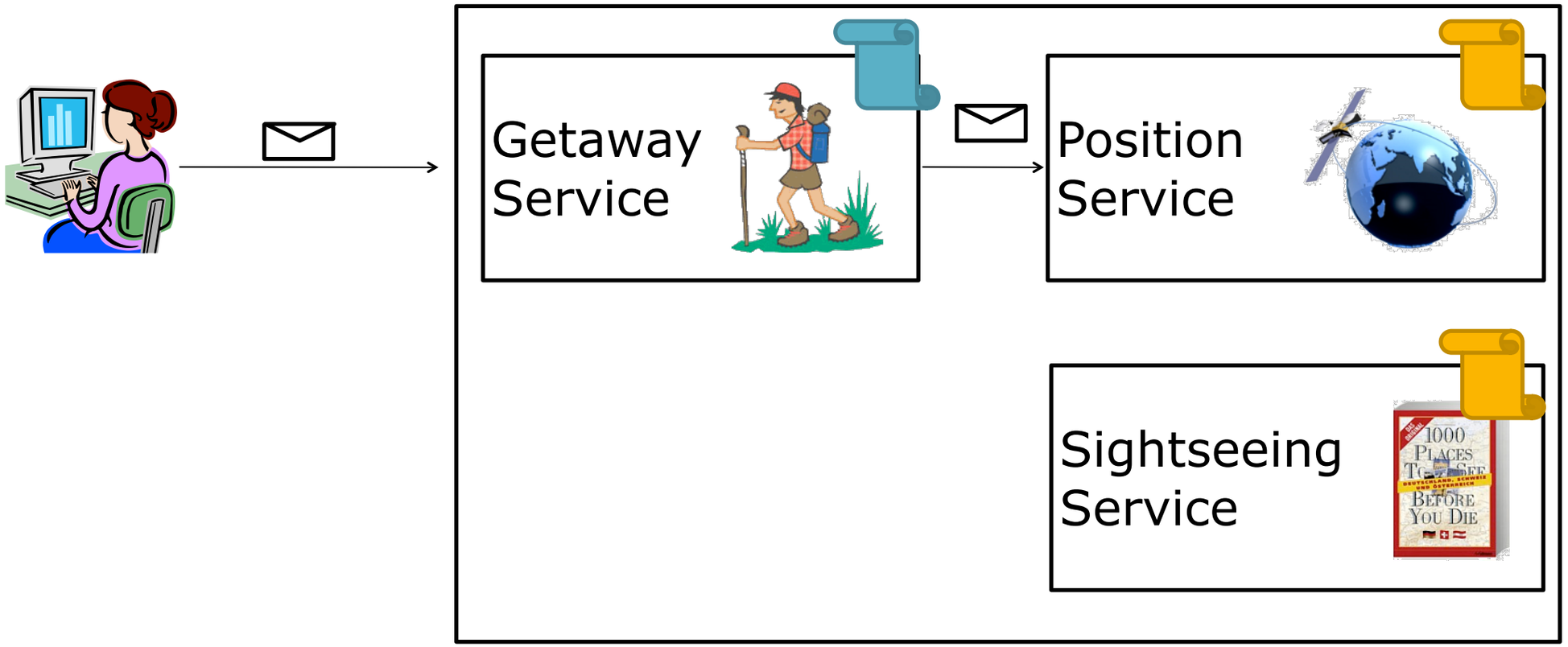
    ➔ Finds sights near the current position



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

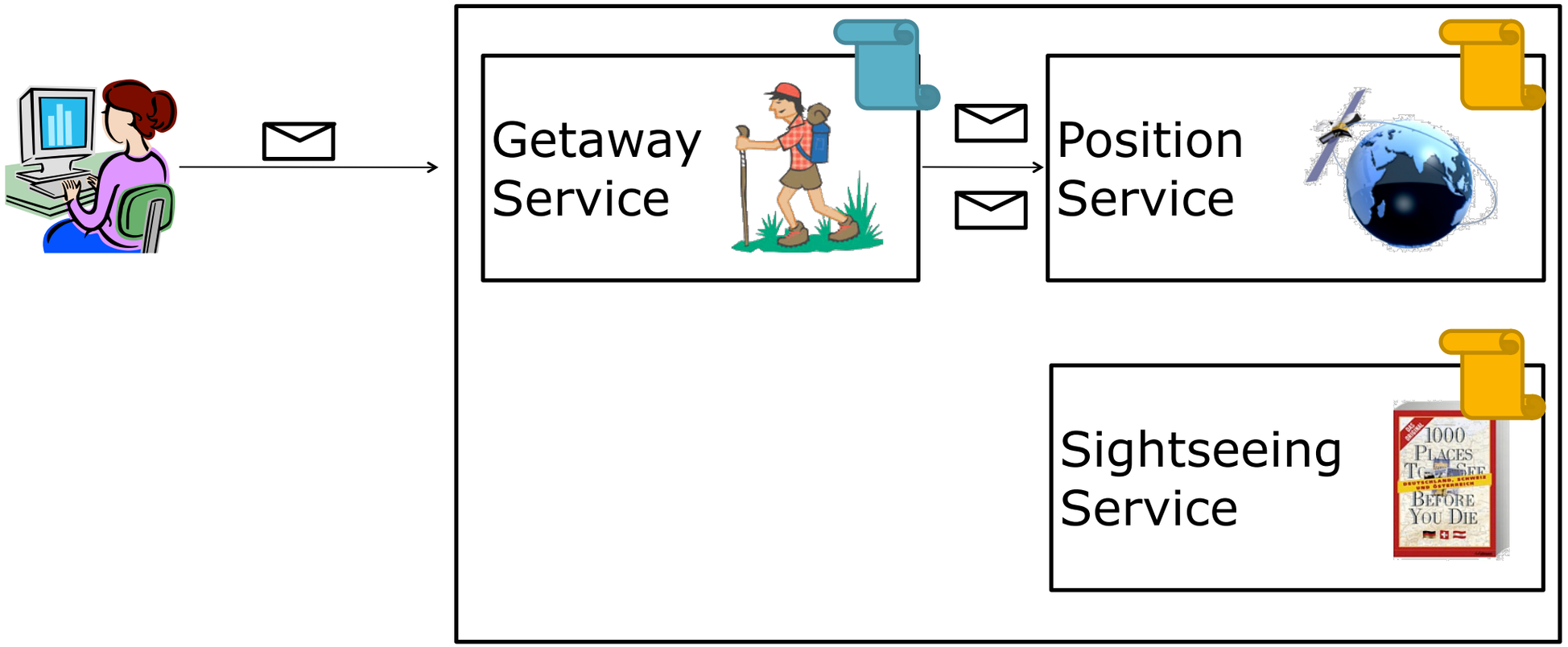
    ➔ Finds sights near the current position



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

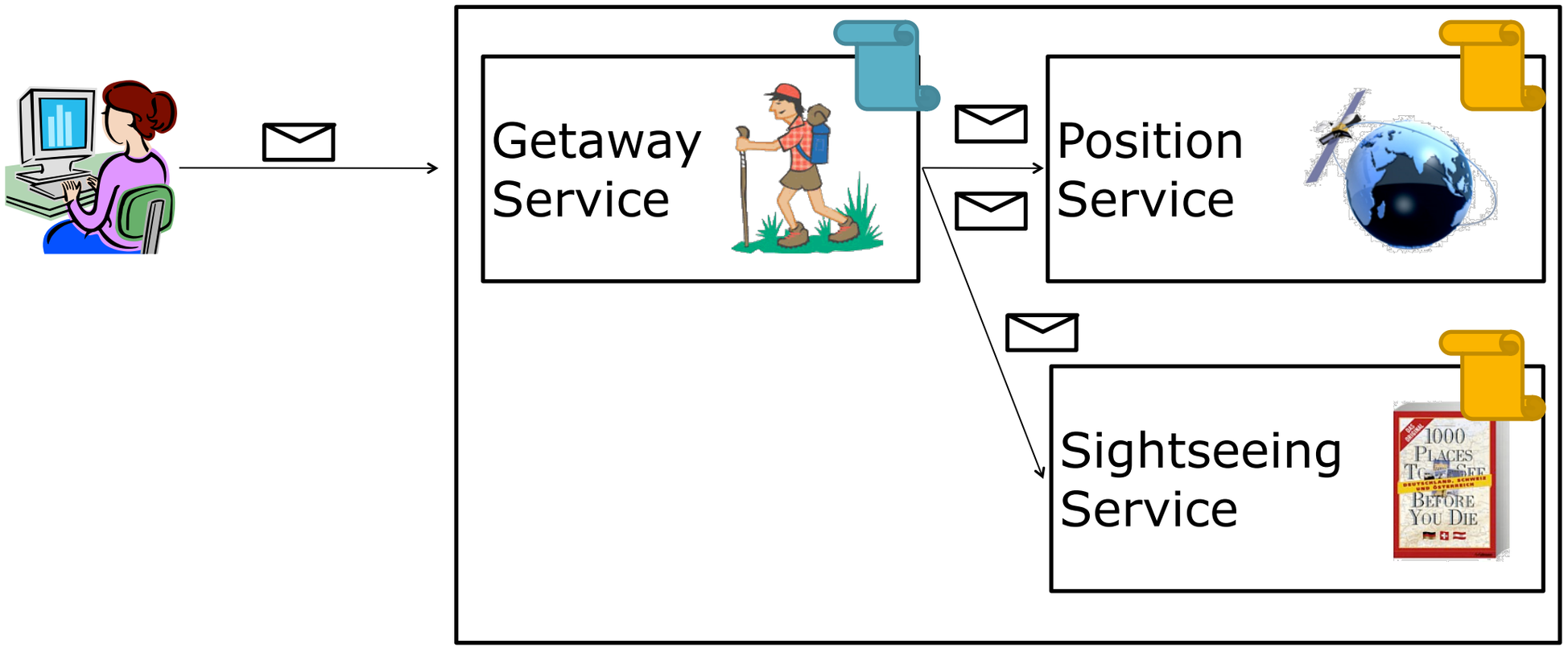
    ➔ Finds sights near the current position



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

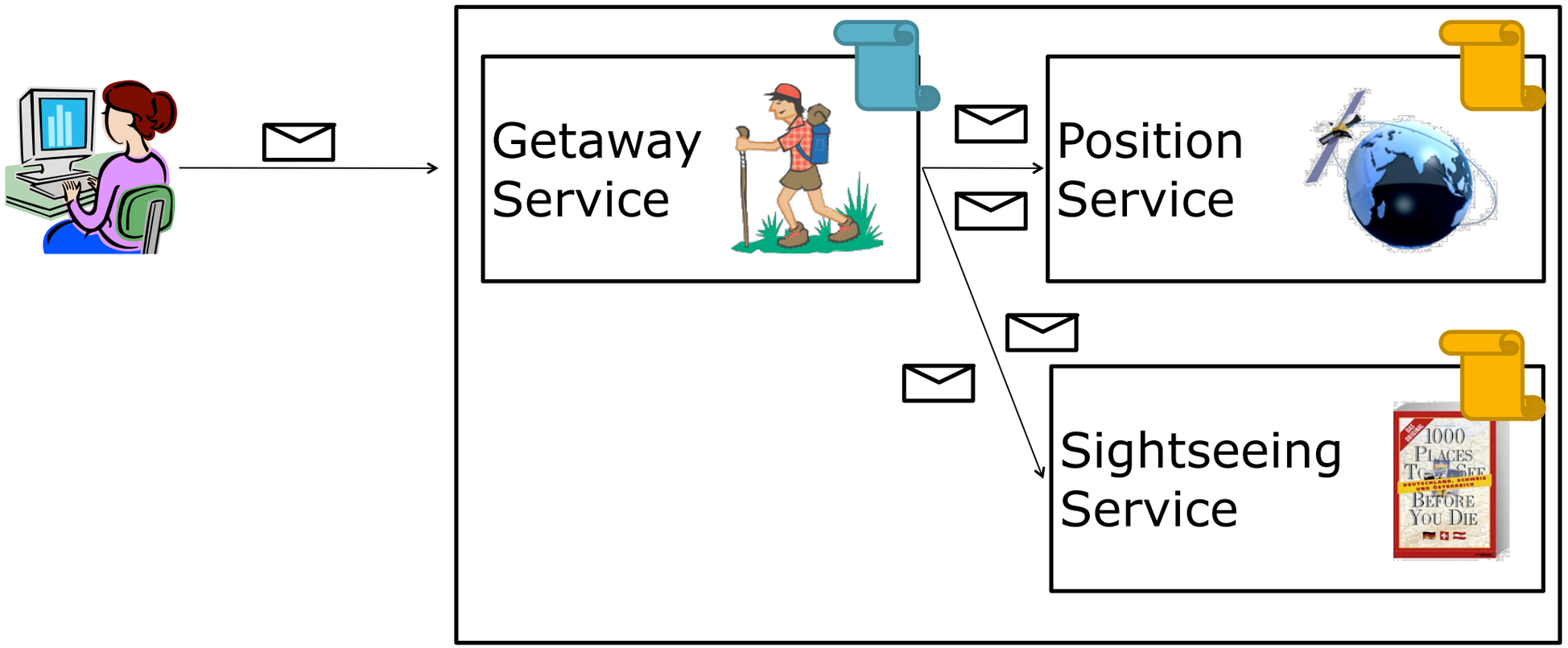
    ➔ Finds sights near the current position



BPEL by Example

- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

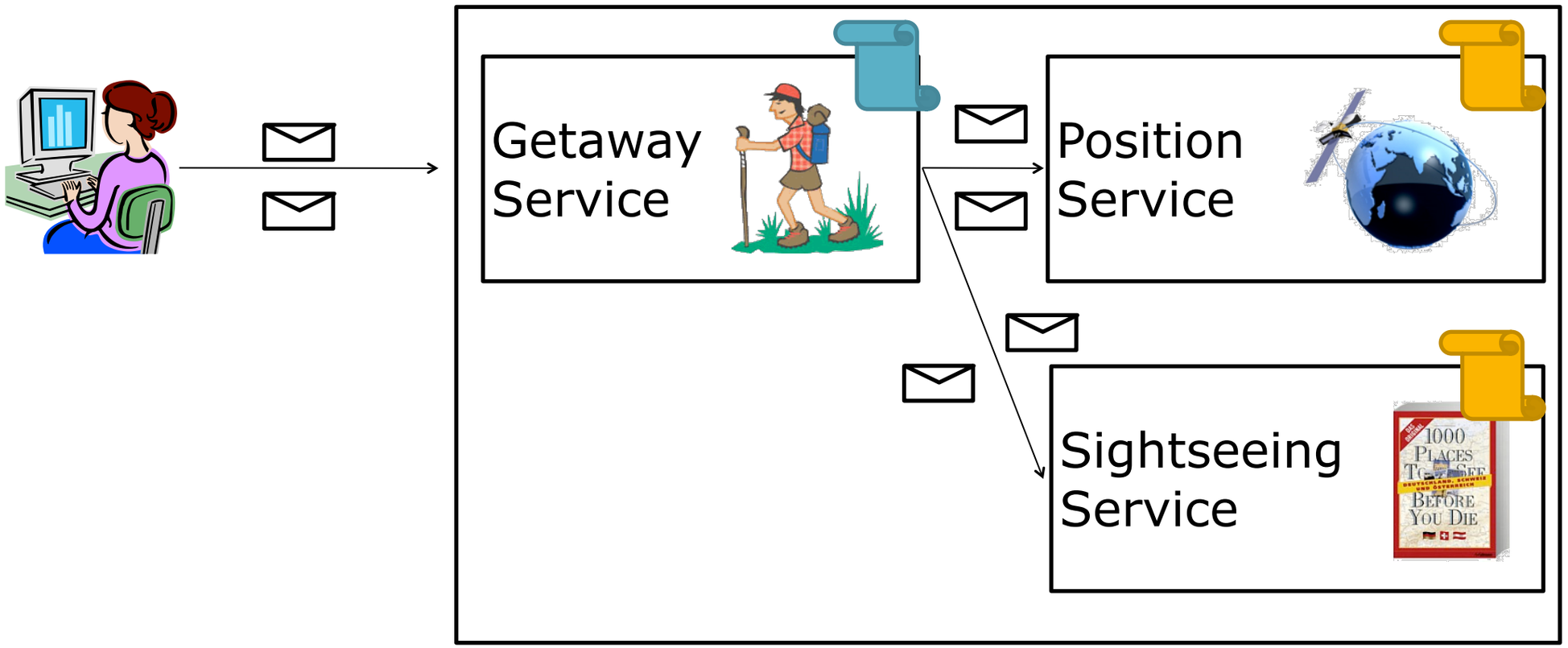
    ➔ Finds sights near the current position



BPEL by Example

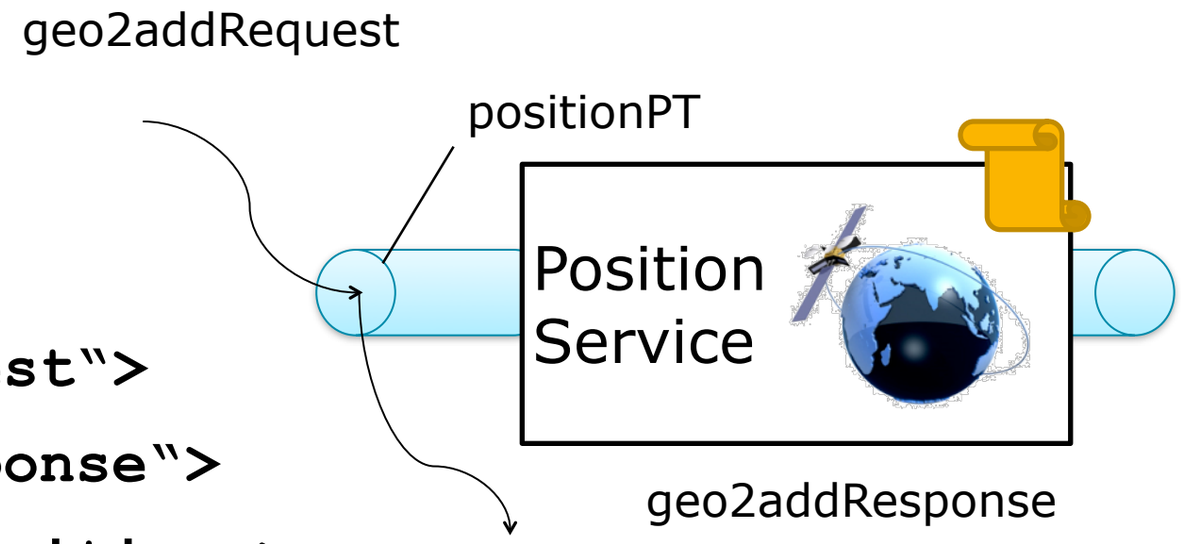
- Composite Getaway service
  - Converts the position into address
  - Finds sights near an address

    ➔ Finds sights near the current position



WSDLs of Existing Services

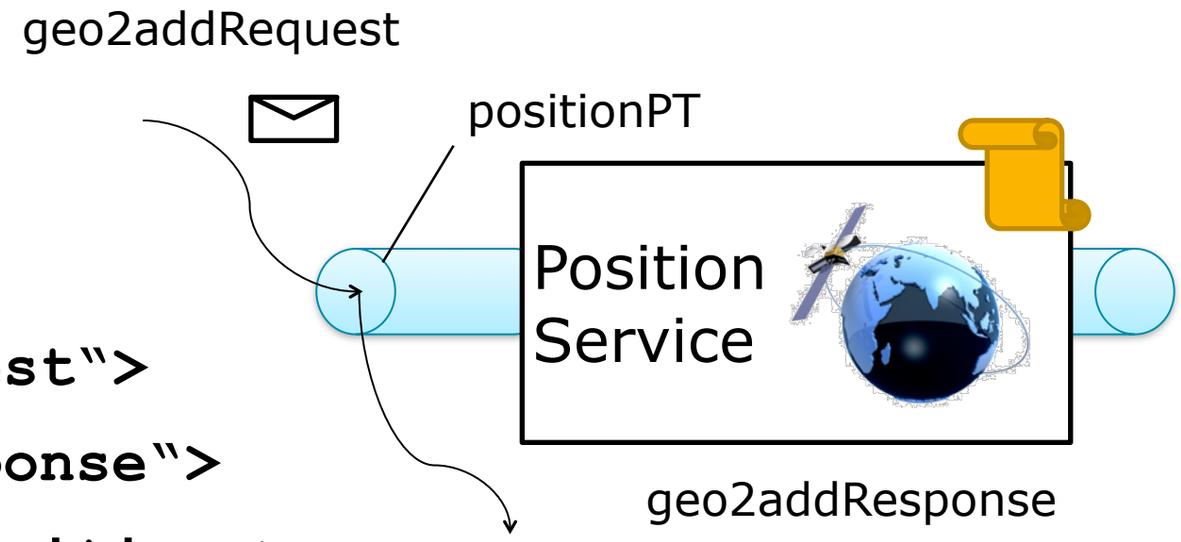

```
<definitions>
 <portType name=„positionPT“>
  <operation name=„geo2add“>
    <input message=„geo2addRequest“>
    <output message=„geo2addResponse“>
</operation></portType>...</definitions>
```




```
<definitions>
 <portType name=„sightsPT“>
  <operation name=„sightsAt“>
    <input message=„sightsAtRequest“>
    <output message=„sightsAtResponse“>
</operation></portType>...</definitions>
```

WSDLs of Existing Services

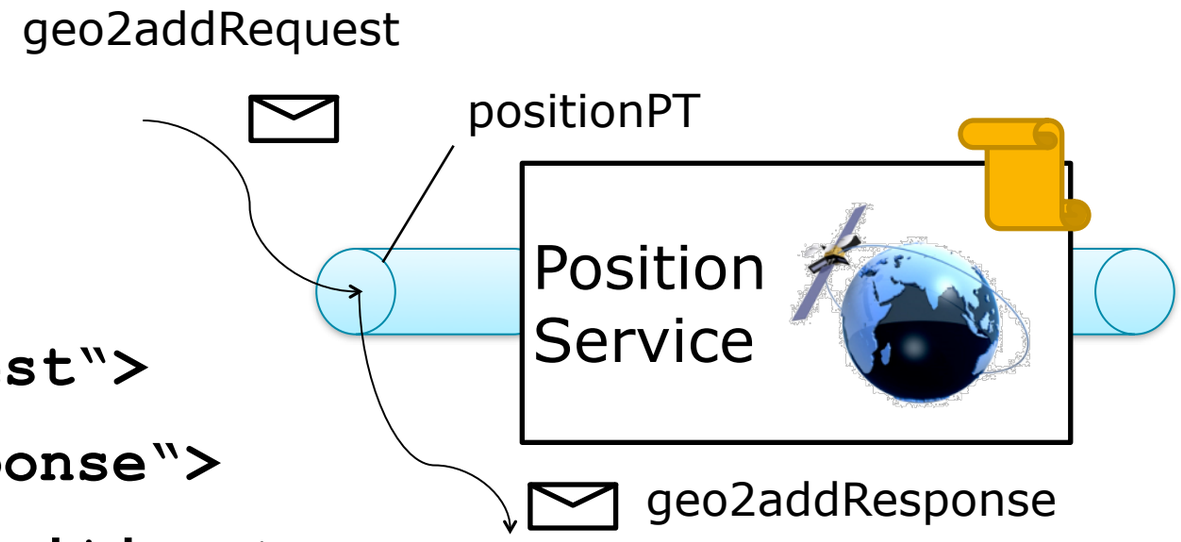

```
<definitions>
 <portType name=„positionPT“>
  <operation name=„geo2add“>
    <input message=„geo2addRequest“>
    <output message=„geo2addResponse“>
</operation></portType>...</definitions>
```




```
<definitions>
 <portType name=„sightsPT“>
  <operation name=„sightsAt“>
    <input message=„sightsAtRequest“>
    <output message=„sightsAtResponse“>
</operation></portType>...</definitions>
```

WSDLs of Existing Services


```
<definitions>
 <portType name=„positionPT“>
  <operation name=„geo2add“>
    <input message=„geo2addRequest“>
    <output message=„geo2addResponse“>
</operation></portType>...</definitions>
```




```
<definitions>
 <portType name=„sightsPT“>
  <operation name=„sightsAt“>
    <input message=„sightsAtRequest“>
    <output message=„sightsAtResponse“>
</operation></portType>...</definitions>
```

Getaway Service WSDL file

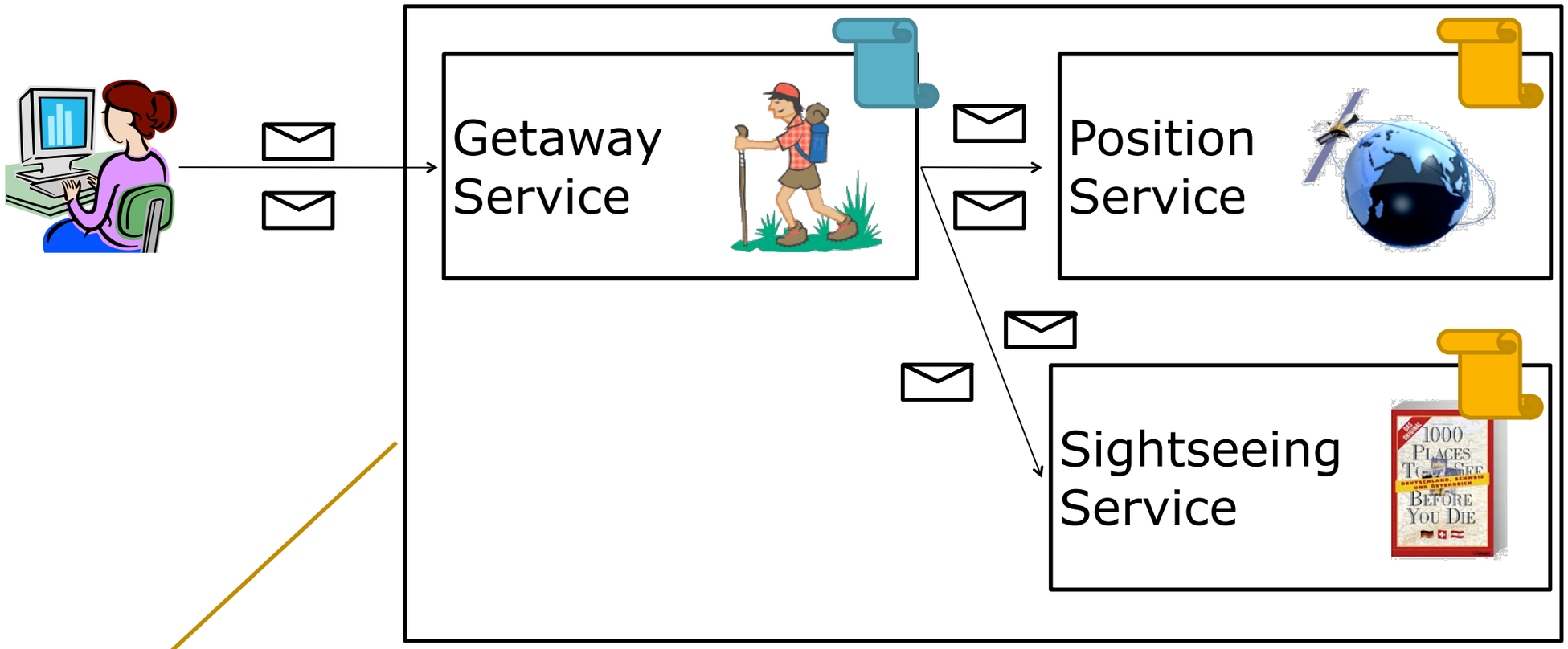

```
<definitions>
 <message name=„closeSightsRequest“>...</message>
 <message name=„closeSightsResponse“>...</message>
 <message name=„geo2addRequest“>...</message>
 <message name=„geo2addResponse“>...</message>
 <message name=„sightsAtRequest“>...</message>
 <message name=„sightsAtResponse“>...</message>

 <portType name=„getawayPT“>
  <operation name=„closeSights“>
   <input message=„closeSightsRequest“>
   <output message=„closeSightsResponse“>
 </operation></portType>

...

</definitions>
```

Getaway Service BPEL file


```
<process>
 <partnerLinks>
  <partnerLink
name=„closeSightsPL“
myRole=„getaway“ />
  <partnerLink name=„geo2addPL“
myRole=„addRequestor“
partnerRole=„addService“ />
  <partnerLink name=„sightsAtPL“
myRole=„sightRequestor“
partnerRole=„sightService“ />
 </partnerlinks>
```

```
<variables>
  <variable
name=„closeSightsRequest“ />
  ...
  <variable
name=„sightsAtResponse“ />
 </variables>

 <sequence>
  <receive />
  <invoke />
  <invoke />
  <reply />
 </sequence>
</process>
```

Getaway Service

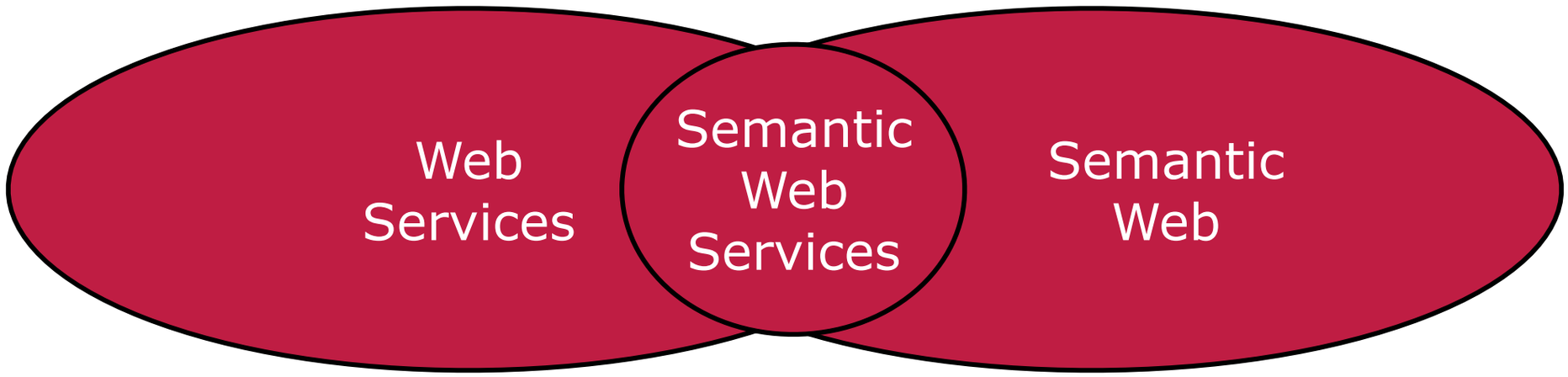



```
<process...>
 <partnerLinks...>
 <variables...>
 <sequence...>
</process>
```

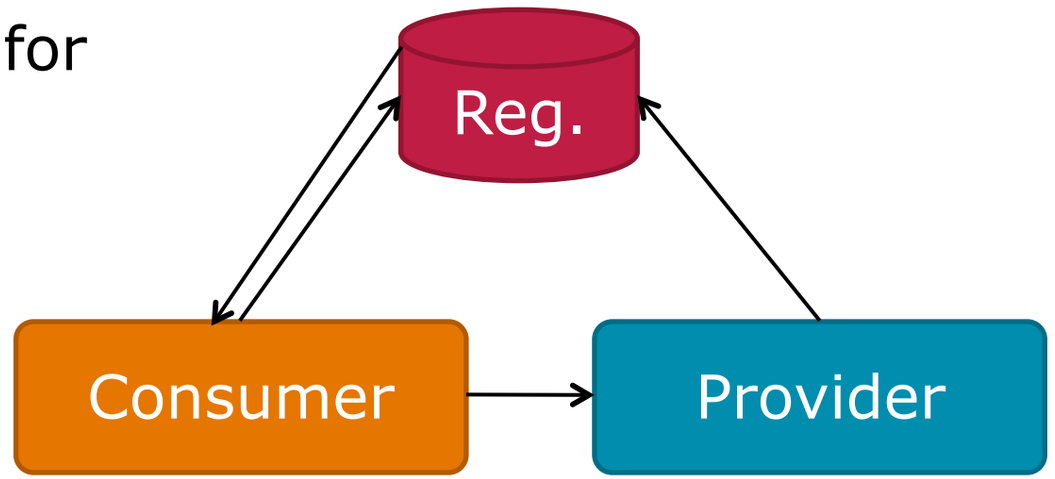
Quality of Service QoS

- Non-functional Properties (NFP)
- Four categories:
  - Security: authentication and authorization of users, message integrity, message encryption
  - Transaction: all-or-nothing
  - Reliable Messaging: deliver messages reliably between Web Services
  - Resource Lifetime Management: immediate and time-based schedule destruction

Semantic Web Services

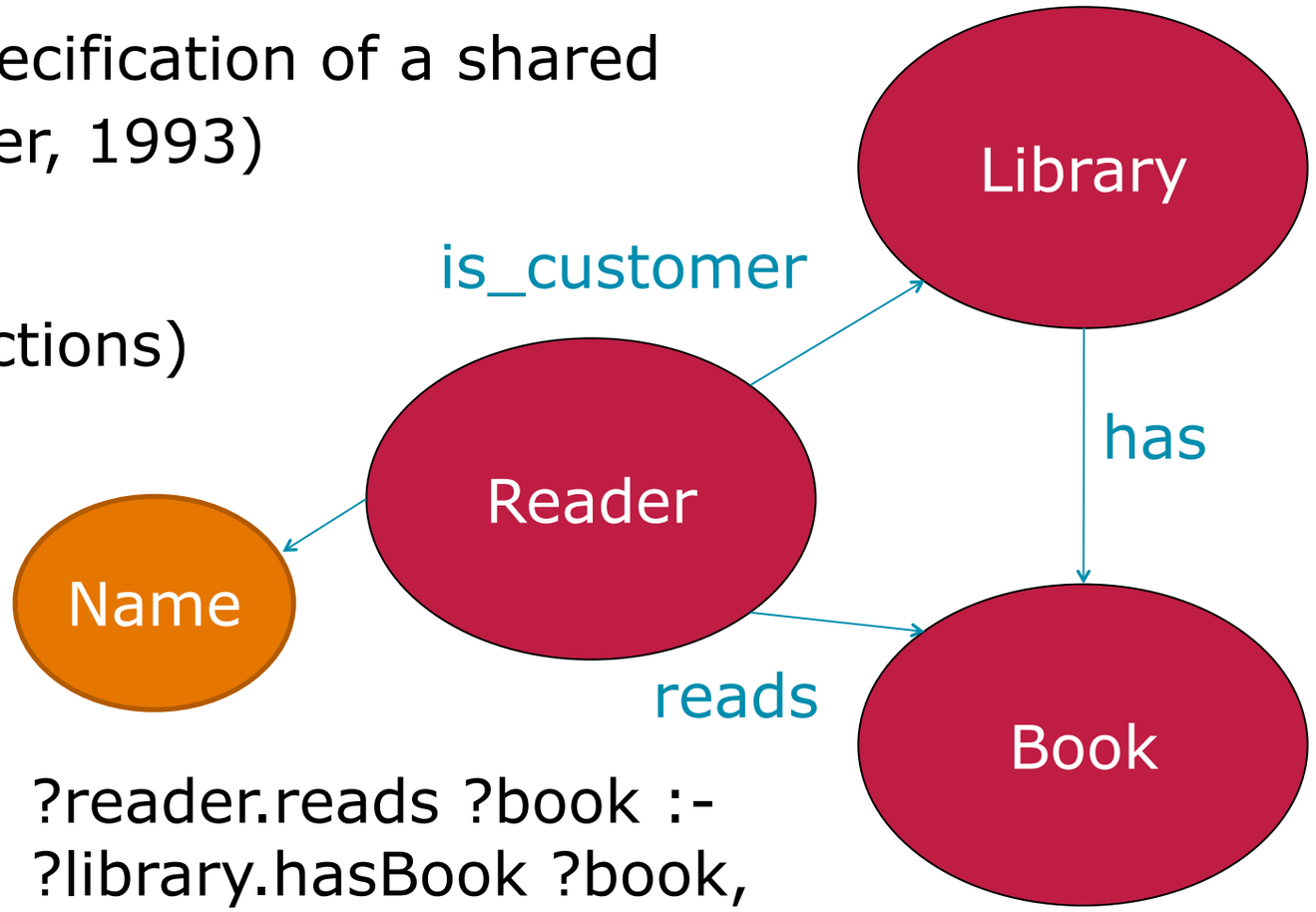


- Combining features from the Semantic Web with Web Service technology
- Objective: high degree of automation for
  - Description
  - Discovery
  - Selection
  - Invocation
- Machine-interpretable meta-information



Semantic Ingredients

- Ontology: "Formal, explicit specification of a shared conceptualization" (Tom Gruber, 1993)
  - Machine-readable
  - Unambiguous (no contradictions)
  - Commonly accepted
  - Model of a domain
- Knowledge Base
  - Contains objects
- Resource Description Framework
  - Knowledge (meta-data) representation format
  - Triples
    - Subject, relation/predicate, object



?reader.reads ?book :-
?library.hasBook ?book,
?reader.is_customer ?library.

OWL-S

- Web Ontology Language for Web Services
  - Ontology for semantic markup of Web Services
- Service
  - Concept which has to be specified for each annotated Web Service
- ServiceProfile
  - Overall description of the service (discovery)
- ServiceModel
  - Fine-grained specification of the internal functionality of the service (selection)
- ServiceGrounding
  - Information about how to invoke the Web Service (invocation)

Service Profile

- Service Profile

```
<owl:Class id=„AirlineTicketing“>
 <subClassOf resource=„#E_Commerce“ />
 <subClassOf>
  <restriction>
   <onProperty resource=„#merchandise“ />
   <allValuesFrom resource=„#CommercialAirlineTravel“ />
  </restriction>
 </subClassOf>
</owl:class>
```

Service Model

- Service Model

```
<process:AtomicProcess id=„GetFlightDetails“>
 <hasInput resource=„#DepartureAirport“ />
 <hasInput resource=„#ArrivalAirport“ />
 <hasInput resource=„#OutboundDate“ />
 ...
 <hasOutput resource=„#FlightsFound“ />
</AtomicProcess>

<process:AtomicProcess id=„SelectFlight“>
 <hasInput resource=„#FlightsAvailable“ />
 <hasOutput resource=„#SelectedFlight“ />
</AtomicProcess>
```

Summary

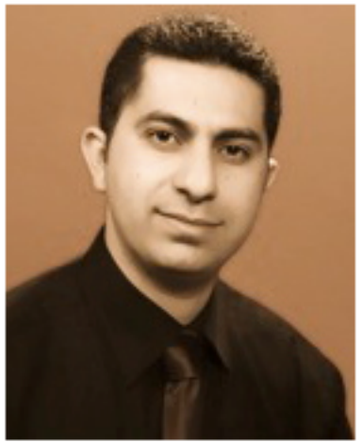
- A set of standards already evolved to serve realizing Web Services
- WSDL is an XML-based description language for single Web Services.
- SOAP is an XML-based protocol for service messaging.
- UDDI is an XML-based language for publishing and discovering Web Services.
- BPEL is an XML-based description language for composite Web Services.
- Non-functional Properties of Web Services are vital criteria in service selection.
- Semantic Web Services aim at automating the process of finding, selecting and consuming Web Services.

Closing Remarks

- Web Services' related files are create automatically using special tools
- RESTful Web Services are emerging
- ebXML is becoming the successor of UDDI
- DO NOT hesitate to contact us!

Mohammed AbuJarour

Contact Information



Ph.D. Student

Hasso-Plattner-Institut for IT Systems Engineering
Prof.-Dr.-Helmert-Str. 2-3
D-14482 Potsdam, Germany

Phone: ++49 331 5509 276
Fax: ++49 331 5509 287
Room: A-1.11
Email: mohammed(dot)abujarour(at)hpi.uni-potsdam.de

Member of the HPI Research School on "Service-Oriented Systems Engineering"

Tobias Vogel

Hasso-Plattner-Institut
für Softwaresystemtechnik
Prof.-Dr.-Helmert-Straße 2-3
D-14482 Potsdam, Germany

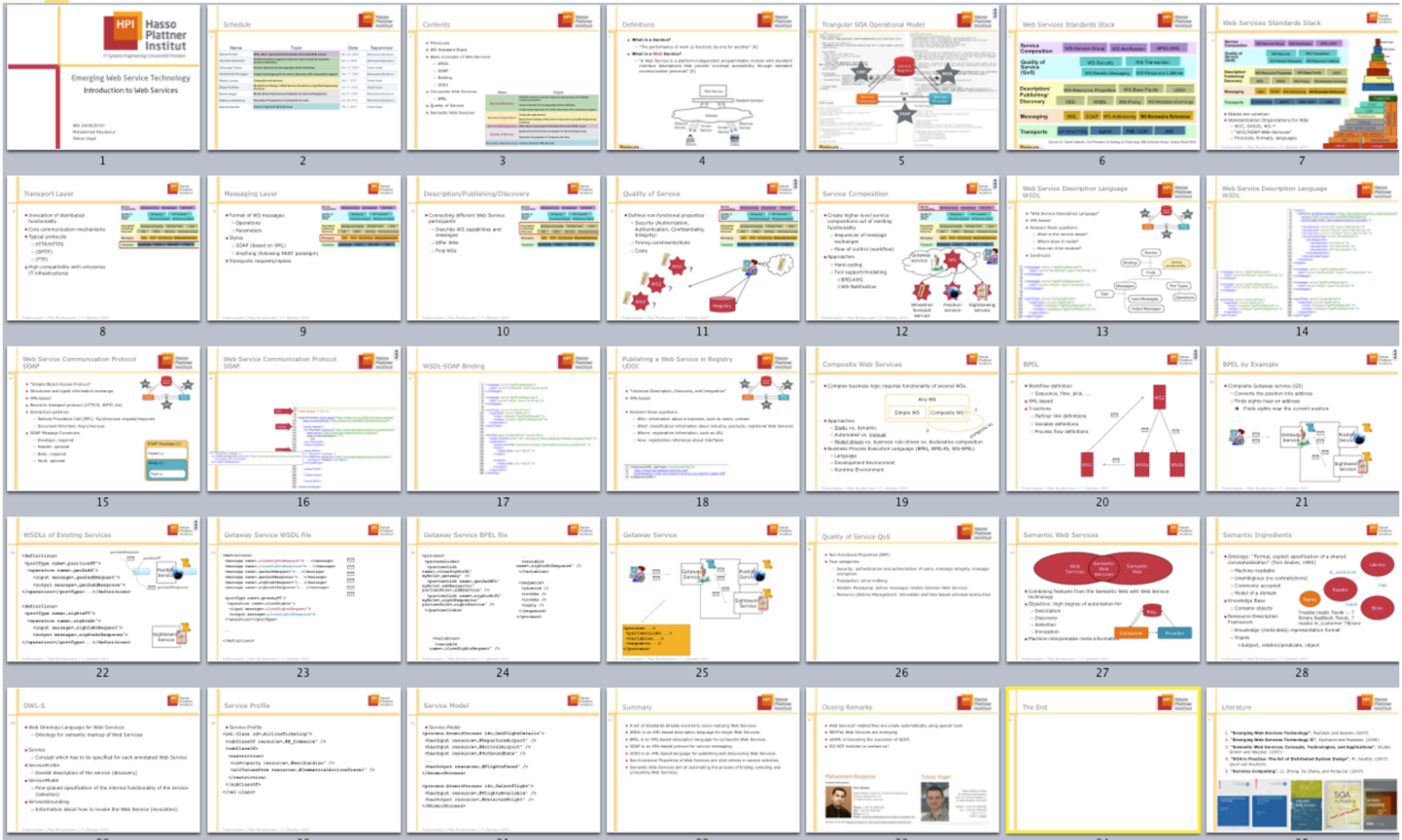
Telefon: ++49 331 5509 292
Fax: ++49 331 5509 287
Raum: A-1.11
E-Mail: T. Vogel

Contents

- Previously
- WS Standard Stack
- Basic Concepts of Web Services
  - WSDL
  - SOAP
  - Binding
  - UDDI
- Composite Web Services
  - BPEL
- Quality of Service
- Semantic Web Services

Area	Topic
Service Discovery	Enabling business experts to discover web services for business process automation
	Service selection by choreography-driven matching
	A logic-based approach for service discovery with composition support
Service Composition	Composite web services
	Model Driven Design of Web Service Operations using Web Engineering Practices
Service Management	BPEL-Mora: Lightweight Embeddable Extensible BPEL Engine
Quality of Service	Model-Driven Performance Evaluation for Service Engineering
	Reputation Propagation in Composite Services
Semantic Web Services	Tools for Semantic Web Services

The End


Emerging Web Service Technology
Introduction to Web Services
Schedule
Contents
Definitions
Triangular SOA Operational Model
Web Services Standards Stack
Web Services Standards Stack
Transport Layer
Messaging Layer
Description/Publishing/Discovery
Quality of Service
Service Composition
Web Service Description Language WSDL
Web Service Description Language WSDL
Web Service Communication Protocol SOAP
Web Service Communication Protocol SOAP
WSDL-SOAP Binding
Publishing a Web Service in Registry UDDI
Composite Web Services
BPEL
BPEL by Example
WSDLs of Existing Services
Getaway Service WSDL file
Getaway Service BPEL file
Getaway Service
Quality of Service QoS
Semantic Web Services
Semantic Ingredients
OWL-S
Service Profile
Service Model
Summary
Closing Remarks
The End
Literature

Literature

1. **“Emerging Web Services Technology”**, Pautasso and Bussler. (2007)
2. **“Emerging Web Services Technology II”**, Gschwind and Pautasso. (2008)
3. **“Semantic Web Services: Concepts, Technologies, and Applications”**, Studer, Grimm and Abecker. (2007)
4. **“SOA in Practice: The Art of Distributed System Design”**, M. Josuttis. (2007) [auch auf Deutsch]
5. **“Services Computing”**, LJ. Zhang, Jia Zhang, and Hong Cai. (2007)