Overview of Web Service Technology

Lapps Grid Group

May 26, 2014

Outline

- Introduction
- Web Service Model
- Web Service Techniques
- Service Oriented Architecture
- Conclusion

Introduction

- Motivation
  - Previous distributed computing solutions (CORBA, Java RMI) imply tight coupling between various components in a system.
  - High level of coordination and shared context among business systems from different organizations needed.
  - Service computing: systems composed by loosely coupled, dynamically flexible bound elements (distributed pieces of software-called services).

What is Web Service

- Concept
  - A Web service is any piece of software that makes itself available over the internet and uses a standardized XML messaging system.
  - A Web service is a software system designed to support interoperable machine-to-machine interaction over a network (W3C).
  - Web Services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains (IBM).
  - A Web service is a collection of open protocols and standards used for exchanging data between applications or systems (tutorials point).

Application Infrastructure Architecture Evolvement

- **Client-Server Architecture**: composed by multiple fat clients where each of them needed to connect to a central server.

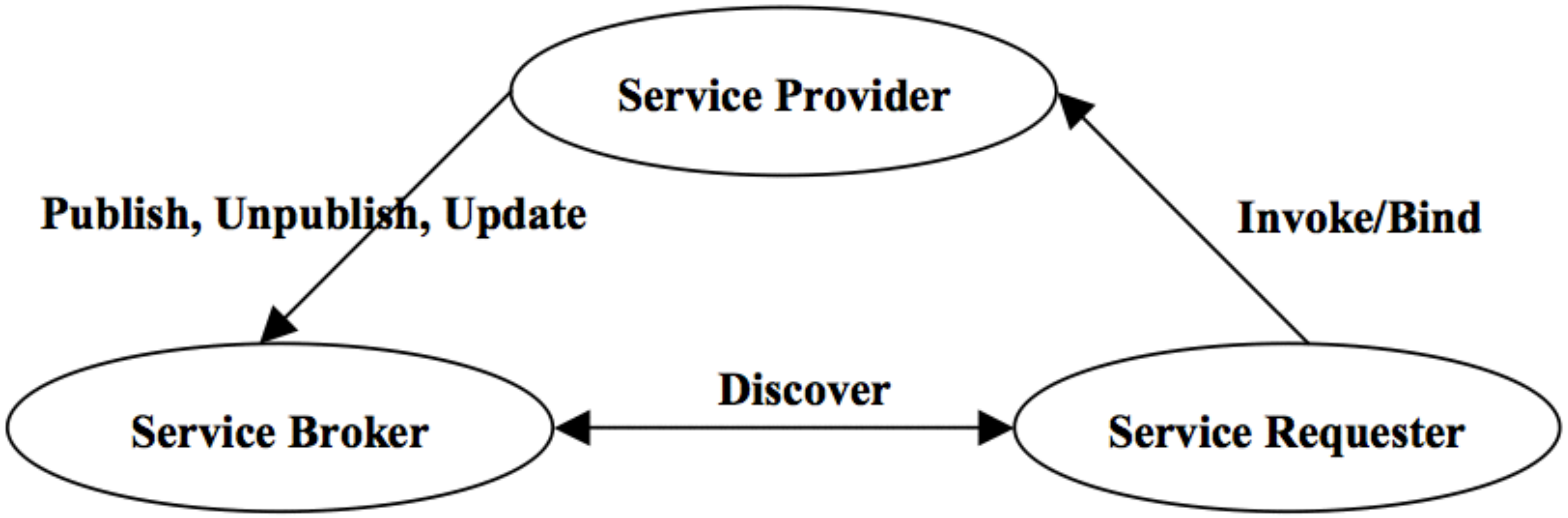
- **Distributed Internet Architecture**: multi-tier client-server applications divide the monolithic client executable into components designed to different degrees of compliance with object orientation

- **Web Services Architecture**: transformation from object-oriented systems toward systems of services can be observed, which contain behavior and messages

- **Service-Oriented Architecture (SOA)**: more complex composed services representing greater added value that applications become more flexible due to their ability to interact with any implementation of a contract

Web Service Model

- Basic Activities
  - Creation
  - Description
  - Publishing
  - Discovery
  - Invocation
  - Un-publishing



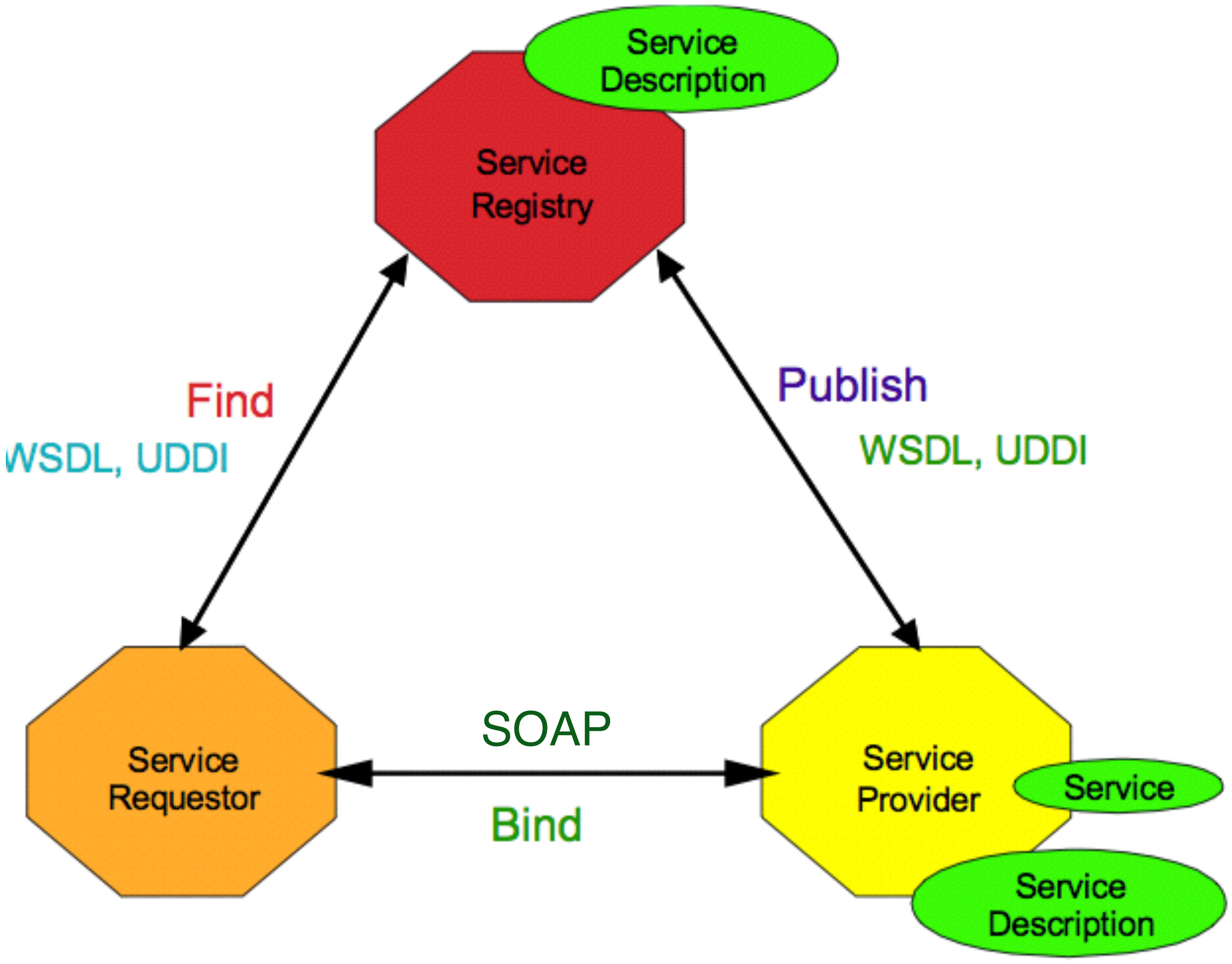
Artifacts, Roles and Operations

- **Artifacts**
  - Services: implementation of an interface described by service description.
  - Service Descriptions: including data types, operations, binding informations, and network location
- **Roles**
  - Service Provider: owner of the services
  - Service Requestor: business (user / program) that requires certain functions be satisfied.
  - Service Broker/Registry: where provider publish service description (optional)
- **Operations**
  - Publish (to be accessible and a service description needed)
  - Discovery/Find (according to service description, interface description, location description)
  - Invoke/Bind (runtime biding and invoke)

Web Service Techniques

- The web services description language (WSDL)
  - WSDL plays a role analogous to Interface Definition Language (IDL) used in distributed programming
- The simple object access protocol (SOAP)
  - SOAP is a standard for sending messages and making remote procedure calls over the Internet. It is independent of the programming language, object model, operating system and platform.
- Universal description, discovery, integration (UDDI)
  - UDDI defines a common means to publish information (type of service, locate information) about businesses and services.

How SOAP, WSDL and UDDI are Related?

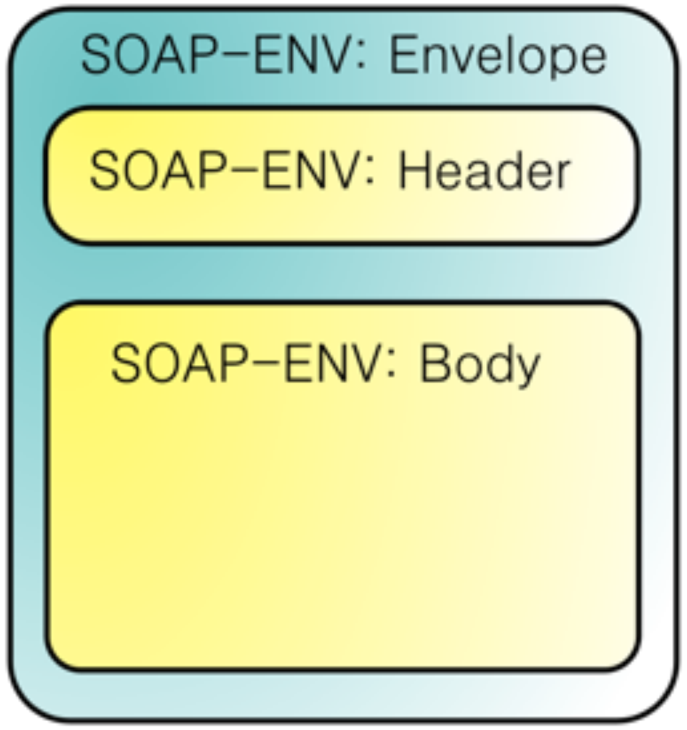


SOAP

- SOAP was designed as an object-access protocol in 1998 by Dave Winer, Don Box, Bob Atkinson, and Mohsen Al-Ghosein for Microsoft

- SOAP is the successor of XML-RPC, though it borrows its transport and interaction neutrality and the envelope/header/body from elsewhere

- SOAP becomes the underlying layer of a more complex set of Web Services, based on Web Services Description Language (WSDL) and Universal Description Discovery and Integration (UDDI)



XML-RPC Request

```xml
<?xml version="1.0"?>
<methodCall>
  <methodName>examples.getStateName</methodName>
  <params>
    <param>
        <value><i4>40</i4></value>
    </param>
  </params>
</methodCall>
```

XML-RPC Response

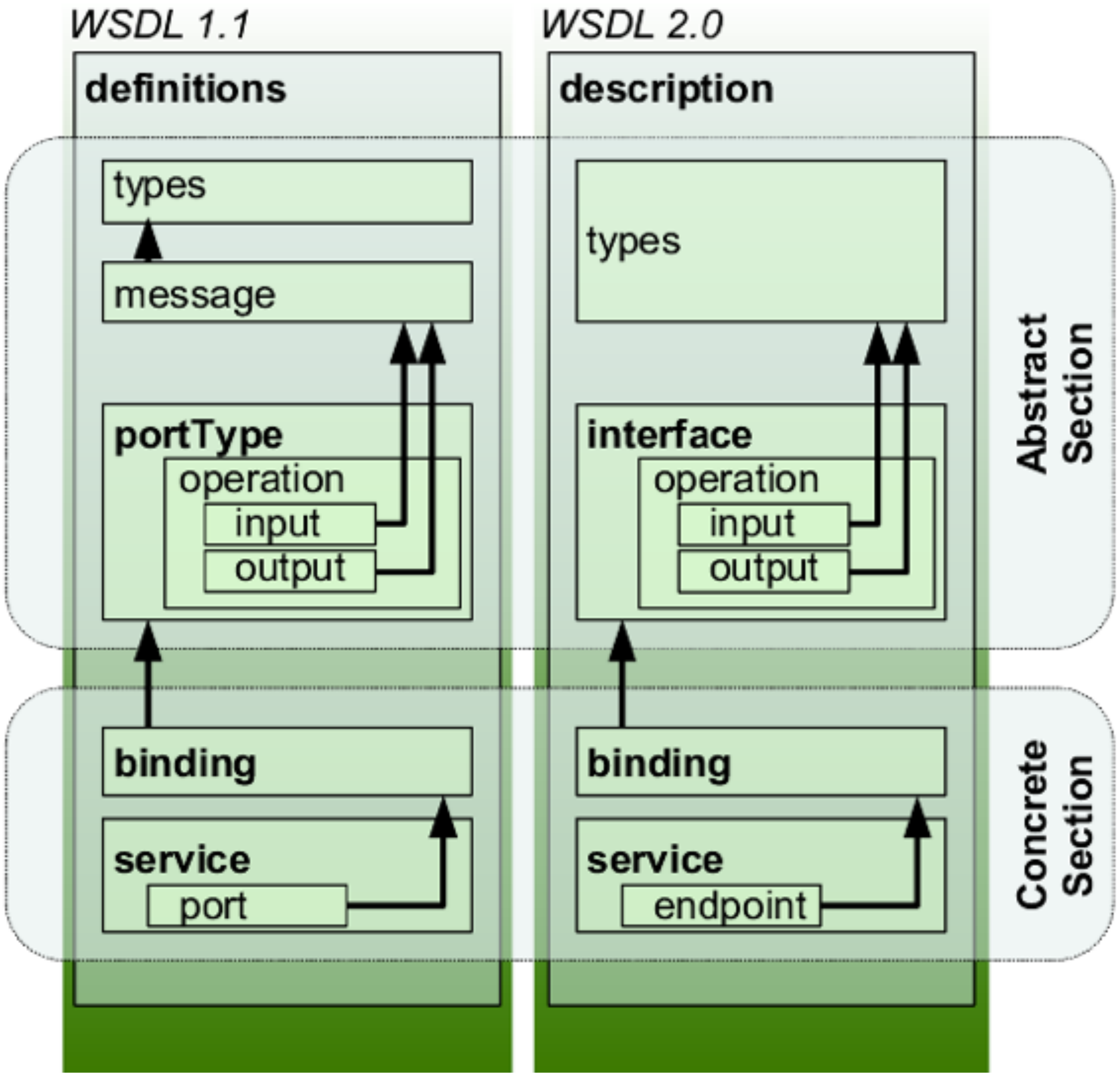
```xml
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
        <value><string>South Dakota</string></value>
    </param>
  </params>
</methodResponse>
```

SOAP

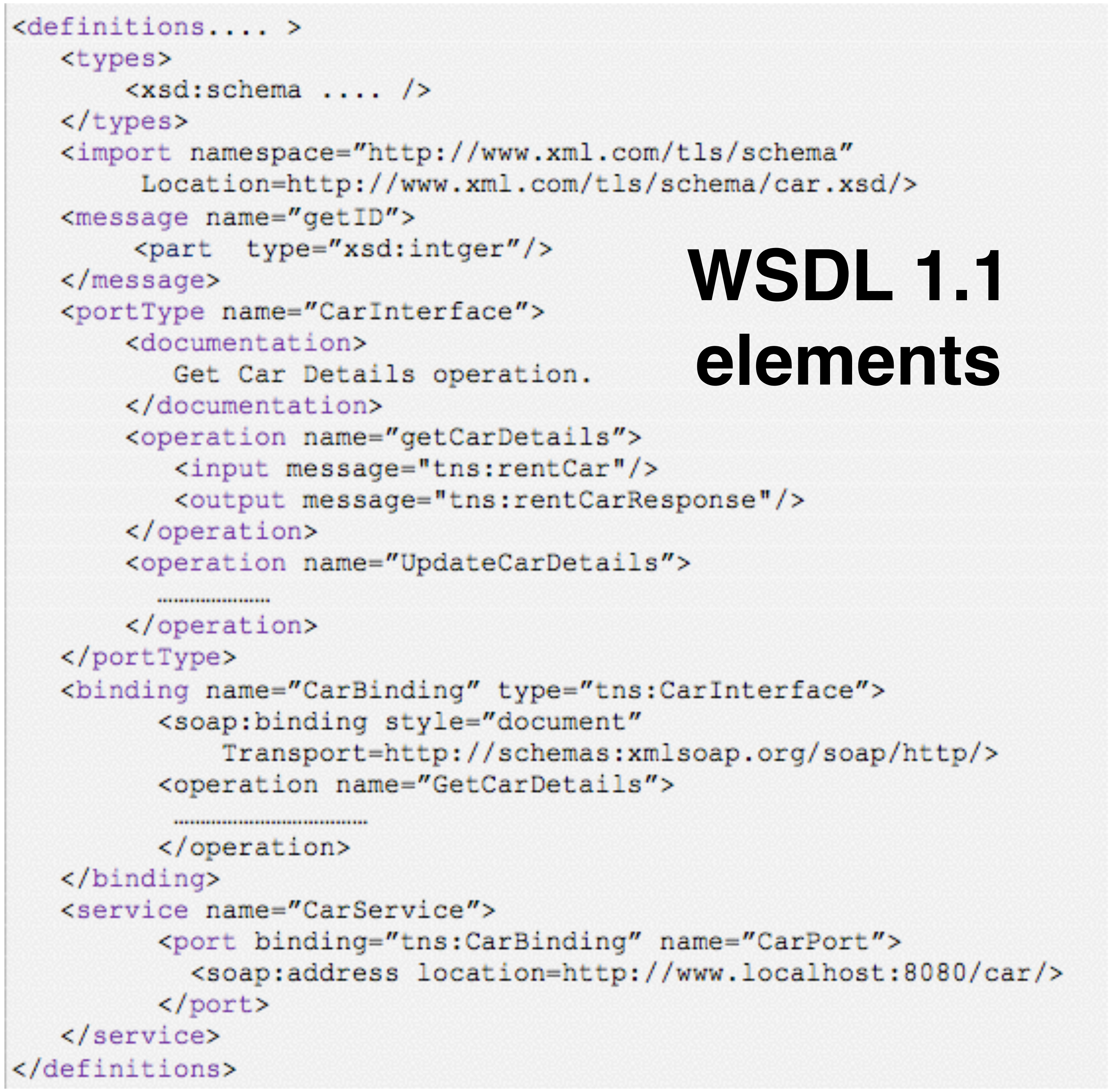
```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 299
SOAPAction: "http://www.w3.org/2003/05/soap-envelope"

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Header>
  </soap:Header>
  <soap:Body>
    <m:GetStockPrice xmlns:m="http://www.example.org/stock">
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```

WSDL

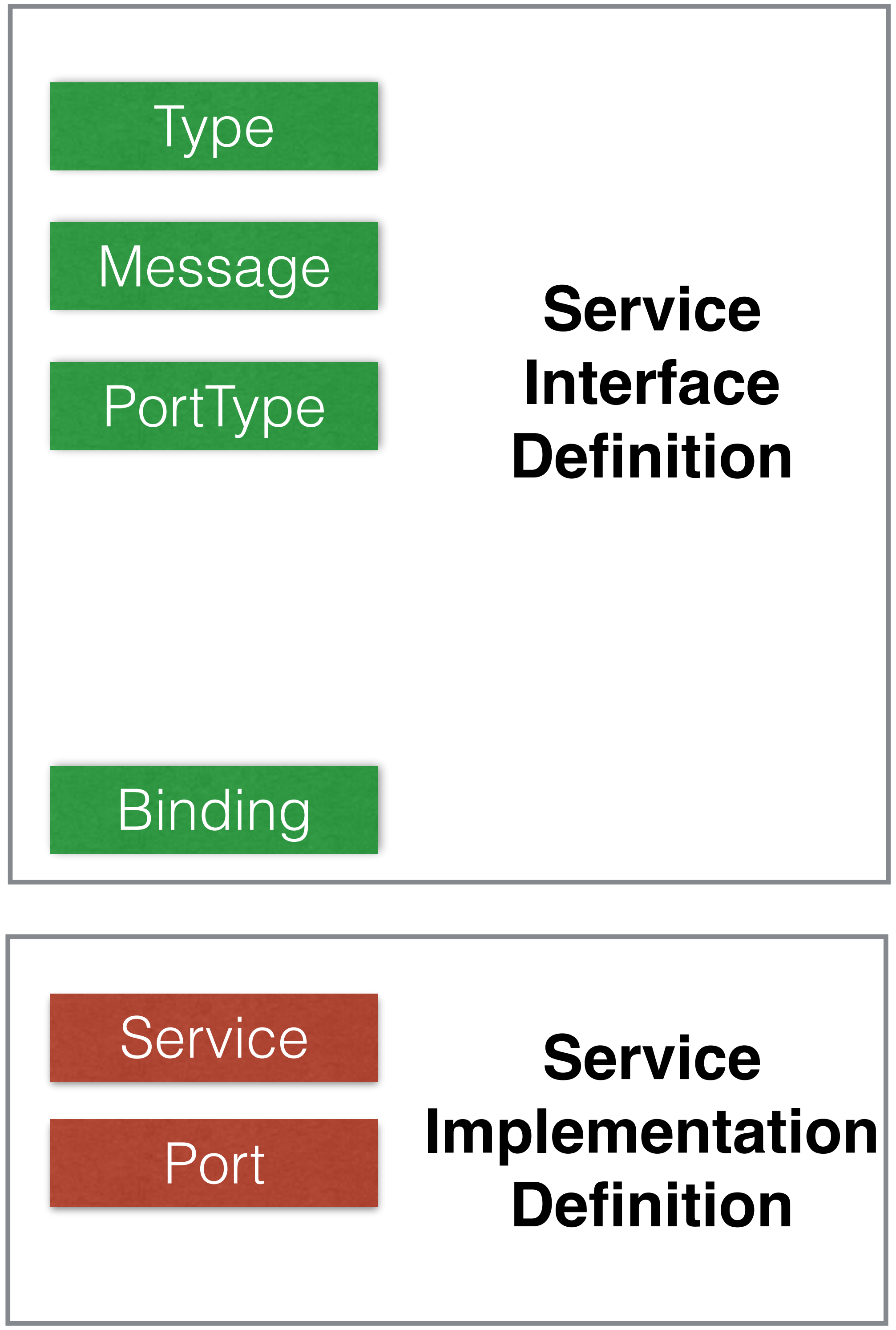
- **types**:"a container for data type definitions using some type system (such as XSD)".
- **portType/interface**: "an abstract set of operations supported by one or more endpoints".
- **binding**: "a concrete protocol and data format specification for a particular port type".
- **port/endpoint**: "a single endpoint defined as a combination of a binding and a network address".



WSDL 1.1 elements

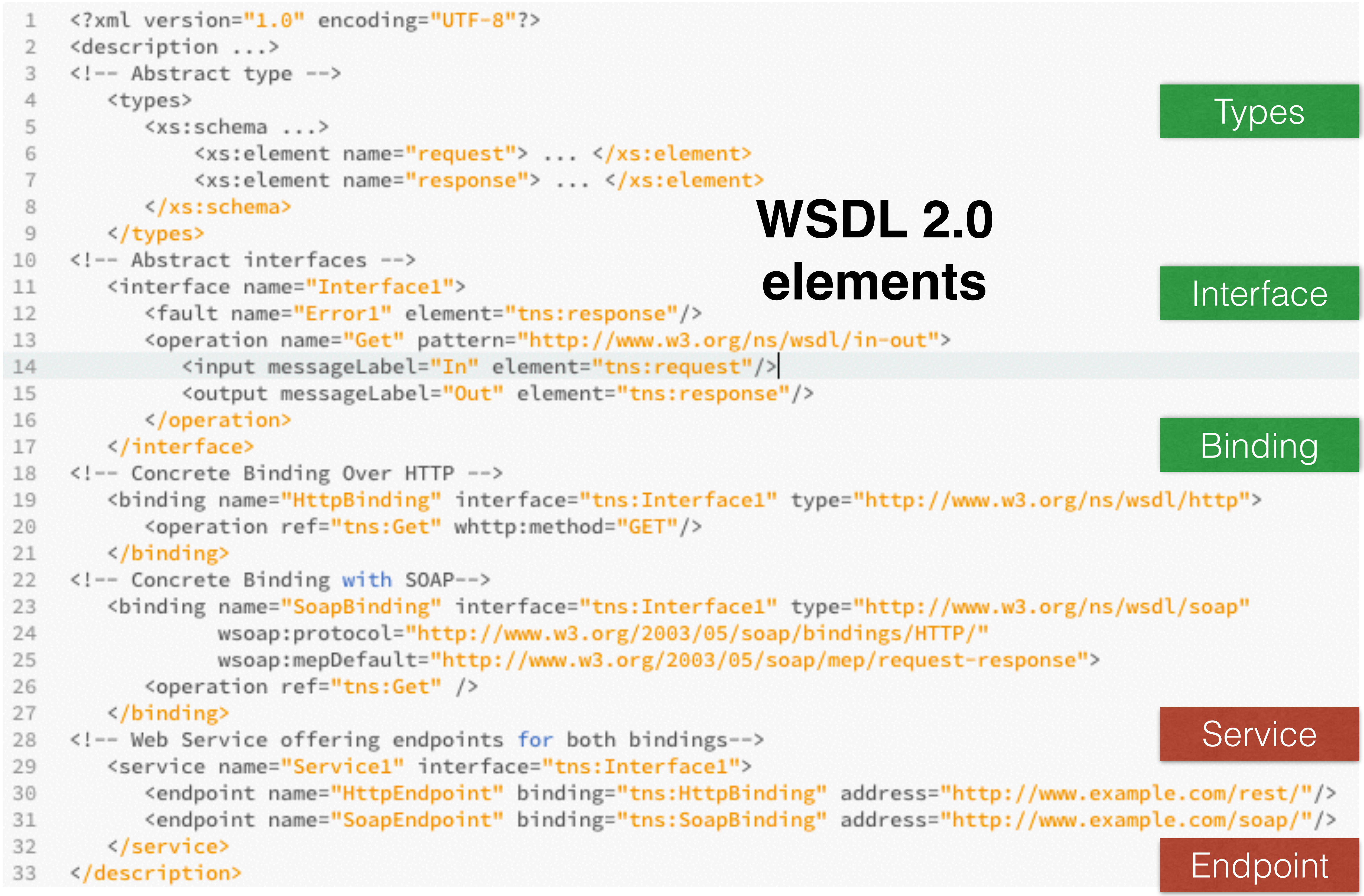


```
<definitions.... >
   <types>
       <xsd:schema .... />
   </types>
   <import namespace=”http://www.xml.com/tls/schema”
        Location=http://www.xml.com/tls/schema/car.xsd/>
   <message name=”getID”>
       <part  type=”xsd:intger”/>
   </message>
   <portType name=”CarInterface”>
       <documentation>
          Get Car Details operation.
       </documentation>
       <operation name=”getCarDetails”>
          <input message="tns:rentCar"/>
          <output message="tns:rentCarResponse"/>
       </operation>
       <operation name=”UpdateCarDetails”>
         ………………………
       </operation>
   </portType>
   <binding name=”CarBinding” type=”tns:CarInterface”>
         <soap:binding style=”document”
             Transport=http://schemas:xmlsoap.org/soap/http/>
         <operation name=”GetCarDetails”>
          ………………………………
         </operation>
   </binding>
   <service name=”CarService”>
         <port binding=”tns:CarBinding” name=”CarPort”>
           <soap:address location=http://www.localhost:8080/car/>
         </port>
   </service>
</definitions>
```



Type
Message
PortType
Binding
Service Interface Definition
Service
Port
Service Implementation Definition

WSDL 2.0 elements



```
<?xml version="1.0" encoding="UTF-8"?>
<description ...>
<!-- Abstract type -->
   <types>
      <xs:schema ...>
          <xs:element name="request"> ... </xs:element>
          <xs:element name="response"> ... </xs:element>
      </xs:schema>
   </types>
<!-- Abstract interfaces -->
   <interface name="Interface1">
      <fault name="Error1" element="tns:response"/>
      <operation name="Get" pattern="http://www.w3.org/ns/wsdl/in-out">
         <input messageLabel="In" element="tns:request"/>
         <output messageLabel="Out" element="tns:response"/>
      </operation>
   </interface>
<!-- Concrete Binding Over HTTP -->
   <binding name="HttpBinding" interface="tns:Interface1" type="http://www.w3.org/ns/wsdl/http">
      <operation ref="tns:Get" whttp:method="GET"/>
   </binding>
<!-- Concrete Binding with SOAP-->
   <binding name="SoapBinding" interface="tns:Interface1" type="http://www.w3.org/ns/wsdl/soap"
            wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP/"
            wsoap:mepDefault="http://www.w3.org/2003/05/soap/mep/request-response">
      <operation ref="tns:Get" />
   </binding>
<!-- Web Service offering endpoints for both bindings-->
   <service name="Service1" interface="tns:Interface1">
      <endpoint name="HttpEndpoint" binding="tns:HttpBinding" address="http://www.example.com/rest/"/>
      <endpoint name="SoapEndpoint" binding="tns:SoapBinding" address="http://www.example.com/soap/"/>
   </service>
</description>
```

Types

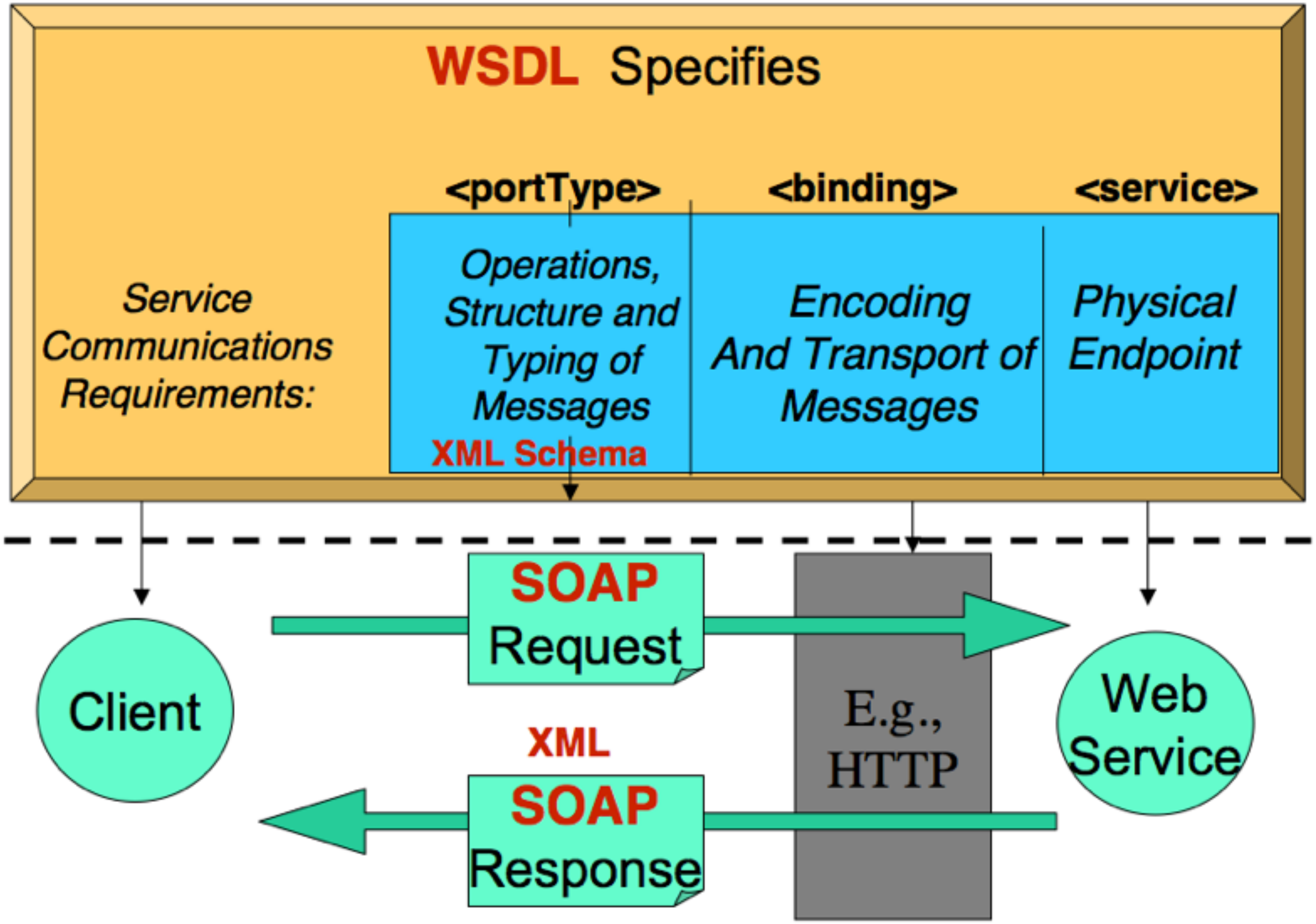
Interface

Binding

Service

Endpoint

WSDL to SOAP Mapping



UDDI

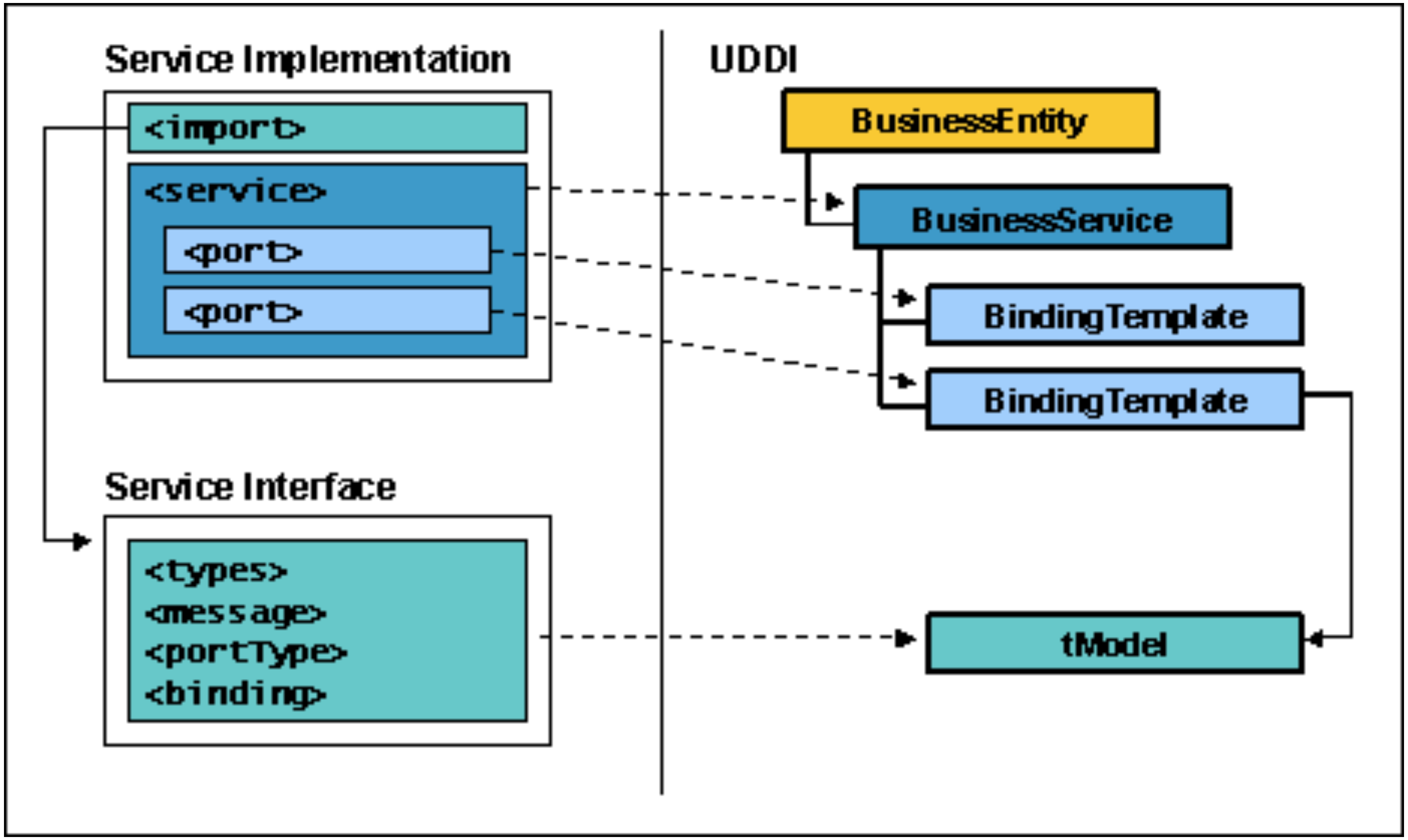
- **Business information**: information that is contained in a businessEntity structure.

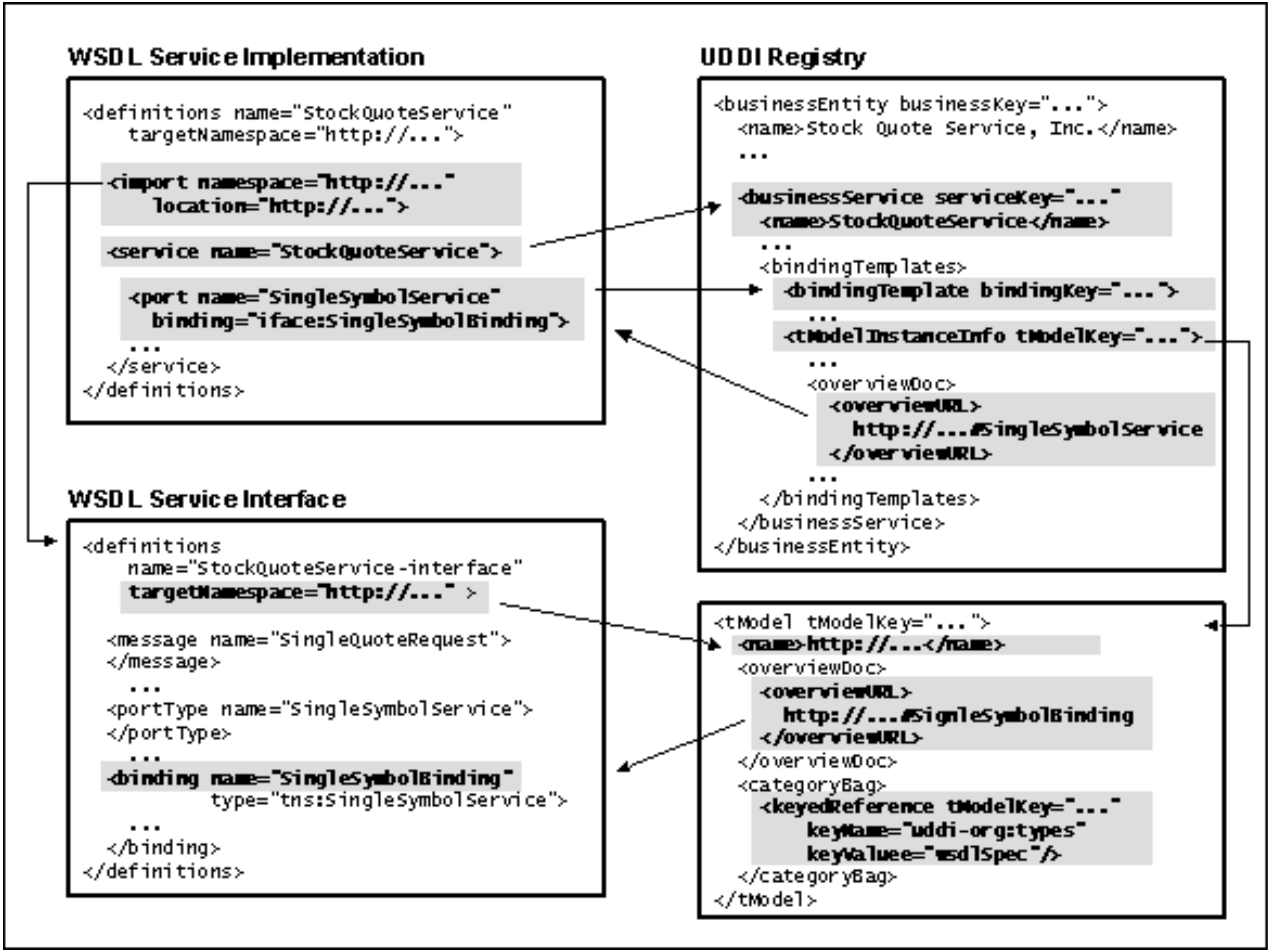
- **Service information**: information that describes a group of Web services. It is contained in a businessService structure.

- **Binding information**: information represented by the bindingTemplate structure.

- **Information describing the specifications for services**: metadata about the various specifications implemented by a given Web service represented by the tModel.

WSDL to UDDI Mapping





```
<definitions name="StockQuoteService"
    targetNamespace="http://...">

  <import namespace="http://..."
      location="http://...">

  <service name="StockQuoteService">

    <port name="SingleSymbolService"
      binding="iface:SingleSymbolBinding">
    ...
  </service>
</definitions>
```

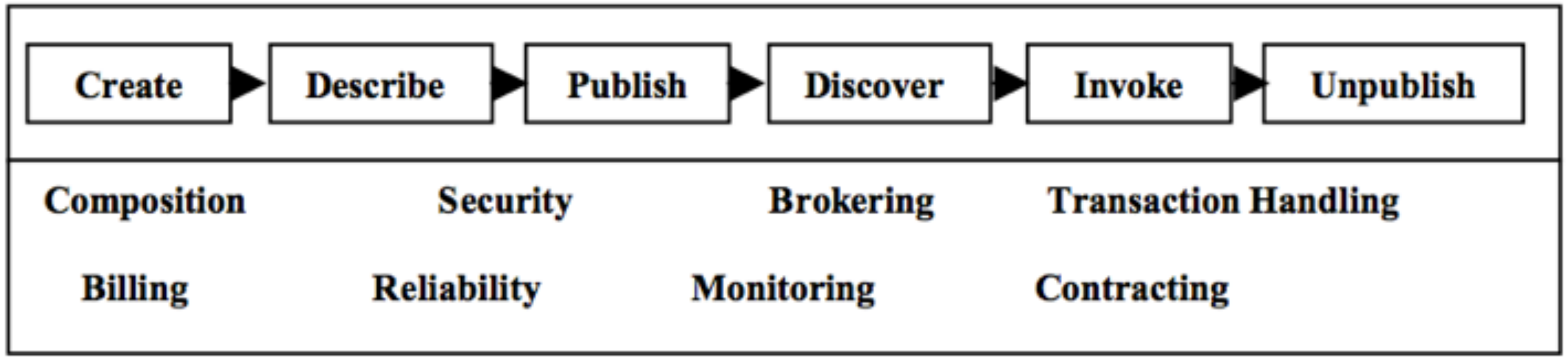
```
<definitions
    name="StockQuoteService-interface"
    targetNamespace="http://..." >

  <message name="SingleQuoteRequest">
  </message>
    ...
  <portType name="SingleSymbolService">
  </portType>
    ...
  <binding name="SingleSymbolBinding"
           type="tns:SingleSymbolService">
    ...
  </binding>
</definitions>
```

```
<businessEntity businessKey="...">
  <name>Stock Quote Service, Inc.</name>
  ...

  <businessService serviceKey="..."
    <name>StockQuoteService</name>
    ...
    <bindingTemplates>
      <bindingTemplate bindingKey="...">
        ...
      <tModelInstanceInfo tModelKey="...">
        ...
        <overviewDoc>
          <overviewURL>
            http://...#SingleSymbolService
          </overviewURL>
        ...
    </bindingTemplates>
  </businessService>
</businessEntity>
```

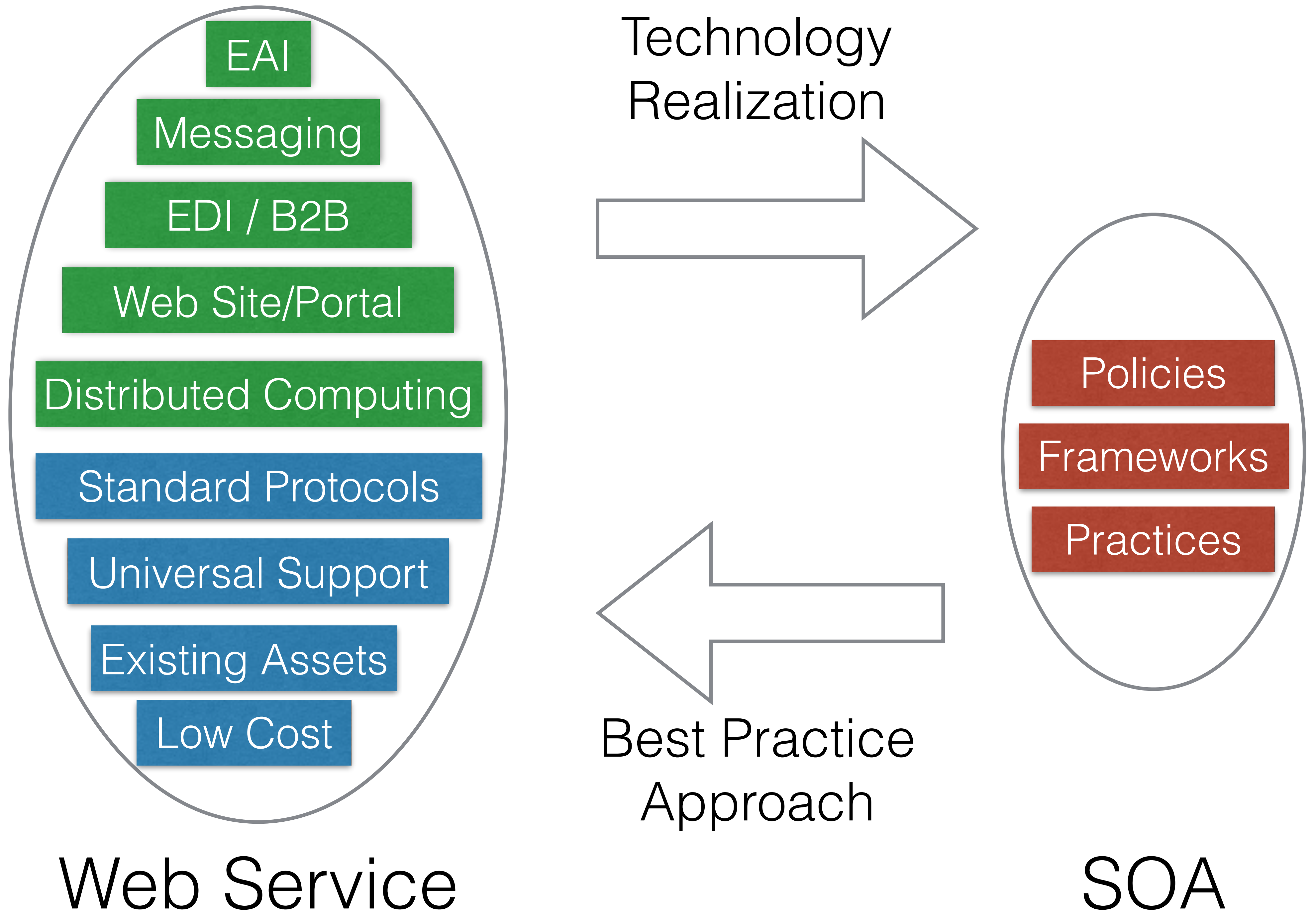
```
<tModel tModelKey="...">
  <name>http://...</name>
  <overviewDoc>
    <overviewURL>
      http://...#SignleSymbolBinding
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey="..."
        keyName="uddi-org:types"
        keyValuee="wsdlSpec"/>
  </categoryBag>
</tModel>
```

Service Oriented Architecture

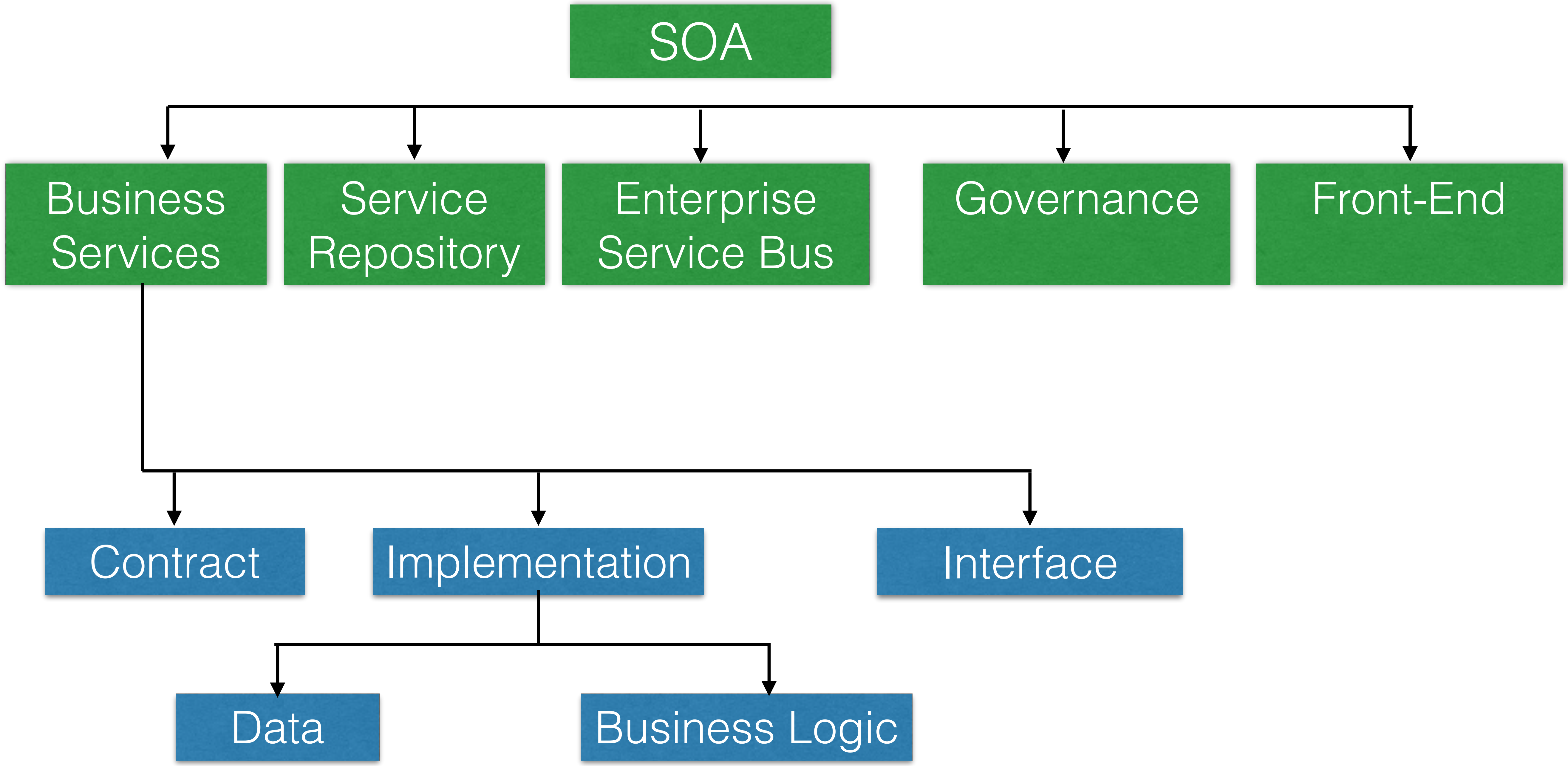
- Definition:
  - Service-oriented architecture (SOA) is a software design and software architecture design pattern based on discrete pieces of software providing application functionality as services to other applications
- Characteristics
  - Interoperable, Loosely Coupled, Reusable, Scalable
- Value-Added Layer



Web Service and SOA

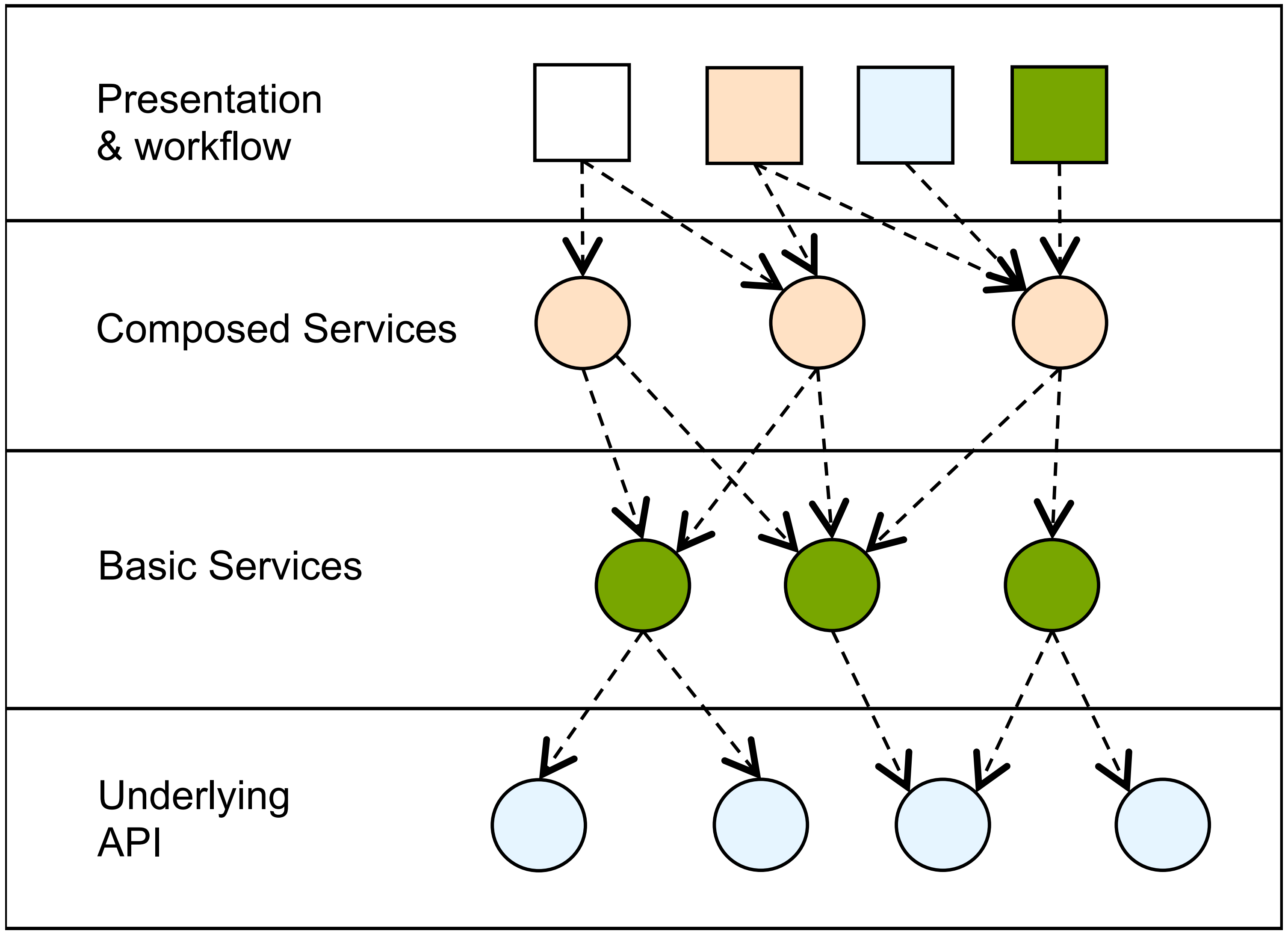


Key Components of SOA



Layers of SOA

- Flexible composition.
- Reuse.
- Functional standardization in lower levels
- Customization in higher layers
- Separation of concerns.
- Policies may vary by layer



Challenges of SOA

- Transaction management is complex in interactions between logically separate system
- Finding the right services and right interfaces
- Organizing the services – registry & repository
- Optimization
- Performance - XML brings robustness not speed
- Security challenges - loosely coupled environment

Conclusion

- Web Service
  - Available, interoperable, self-contained, modular, distributed, dynamic, of open protocols and standards
- Web Service Techniques
  - SOAP, WSDL, UDDI
- Service Oriented Architecture
  - Interoperable, loosely coupled, reusable, scalable
  - With challenges

Reference

- Wikipedia: http://en.wikipedia.org/wiki/Web_Services_Description_Language
- Wikipedia: http://en.wikipedia.org/wiki/SOAP_(protocol)
- Wikipedia: http://en.wikipedia.org/wiki/Web_service
- Wikipedia: http://en.wikipedia.org/wiki/XML-RPC
- Wikipedia: http://en.wikipedia.org/wiki/Service-oriented_architecture
- Web Services Conceptual Architecture (WSCA 1.0) May 2001. By Heather Kreger IBM Software Group.
- Web Services Technologies : State of the Art. Albreshne, Abdaldhem; Fuhrer, Patrik; Pasquier, Jacques. September 2009.
- Web Services Technologies XML and SOAP WSDL and UDDI Version 16, Object Management Group, http://www.omg.org/news/meetings/workshops/MDA-SOA-WS_Manual/00-T1_Newcomer/CH2-WSTechnologies_V16-Standard.pdf
- Understanding WSDL in a UDDI registry, http://www.ibm.com/developerworks/library/ws-wsdl/
- Service Oriented Architecture: Right on Track, TechNet & MSDN, Microsoft. http://download.microsoft.com/download/e/9/d/e9d163db-5c96-46bc-9263-aac62fc38831/Service%20Oriented%20Architecture.pdf