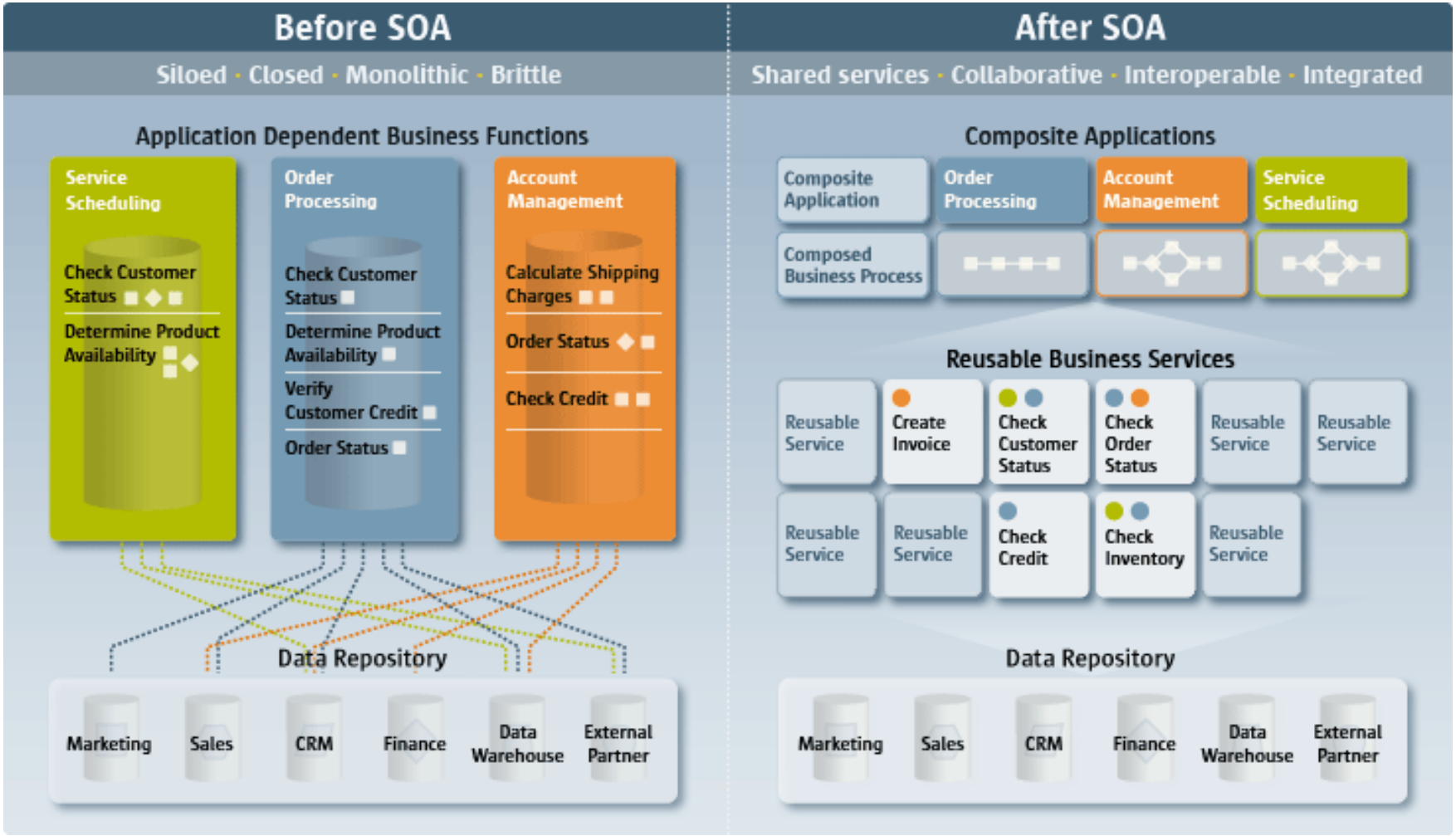
Service Oriented Architecture and Web Services

Traditional Applications vs. SOA

- Computers perform seemingly miraculous tasks, automating many of the things that people did by hand, starting with complex calculations, and moving to financials, and many other tasks.

- But traditional applications are "silos". The human resources application couldn't really talk to the financials application, which couldn't really talk to the distribution application.

Service Oriented Architecture



source:IBM

Web Services

Many organizations use multiple software systems for management. Different software systems often need to exchange data with each other, and a **Web service** is a method of communication that allows two software systems to exchange data over the internet.

Different software might be built using different programming languages, and hence there is a need for a method of data exchange that doesn't depend upon a particular programming language. Thus, Web services extensively use **XML files** for data exchange.

Web service - Definition

- A Web Service is a **URL-addressable software resource** that performs functions (or a function).
- Web services are a new breed of Web application.
- They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web.
- Web services perform functions, which can be anything from simple requests to complicated business processes. … Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service.

Web Services

- Web services are application components
- Web services communicate using open protocols
- Web services are self-contained and self-describing
- Web services can be used by other applications
- HTTP and XML is the basis for Web services

Web Services

- **Web Services take Web-applications to the Next Level**
- By using Web services, your application can publish its function or message to the rest of the world.
- Web services use XML to code and to decode data, and SOAP to transport it (using open protocols).
- With Web services, your accounting department's Win 2k server's billing system can connect with your IT supplier's UNIX server.

Web Services

- https://www.youtube.com/watch?v=mKjvKPlb1rA

Web Services

Web Services have Two Types of Uses:

Reusable application-components.

- There are things applications need very often. So why make these over and over again?
- Web services can offer application-components like: currency conversion, weather reports, or even language translation as services.

Connect existing software.

- Web services can help to solve the **interoperability problem** by giving different applications a way to link their data.
- With Web services you can exchange data between different applications and different platforms.

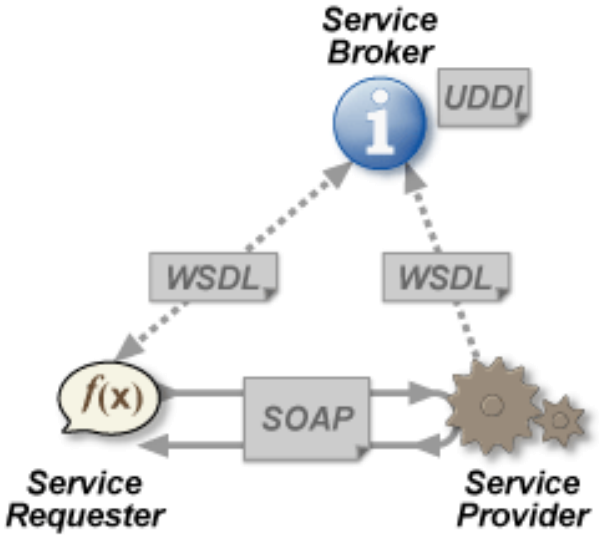
- http://stackoverflow.com/questions/969964/when-to-use-soa-service-oriented-architecture

Web Service Example: Amazon Web Services (AWS)

- Officially launched in 2006, Amazon Web Services provided online services for other web sites or client-side applications
- Most of these services are not exposed directly to end users, but instead offer functionality that other developers can use in their applications.
  - Example: Amazon Product Advertising API (formerly known as Amazon Associates Web Service (A2S) and Amazon E-Commerce Service (ECS)) exposed world's largest product database through Web Services
    - Idea: let others figure out how to sell products for us
    - Enables Web sites to link to Amazon.com and earn referral fees

Web Services

- The software system that requests data is called a ***service requester***, whereas the software system that would process the request and provide the data is called a ***service provider***.



Service Oriented Architecture (SOA)

Web Services - WSDL

Rules for communication between different systems need to be defined, such as:

- How one system can request data from another system
- Which specific parameters are needed in the data request
- What would be the structure of the data produced. Normally, data is exchanged in XML files, and the structure of the XML file is validated against an .xsd file.
- What error messages to display when a certain rule for communication is not observed, to make troubleshooting easier

All of these rules for communication are defined in a file called **WSDL** (Web Services Description Language), which has the extension .wsdl.

Web Services - UDDI

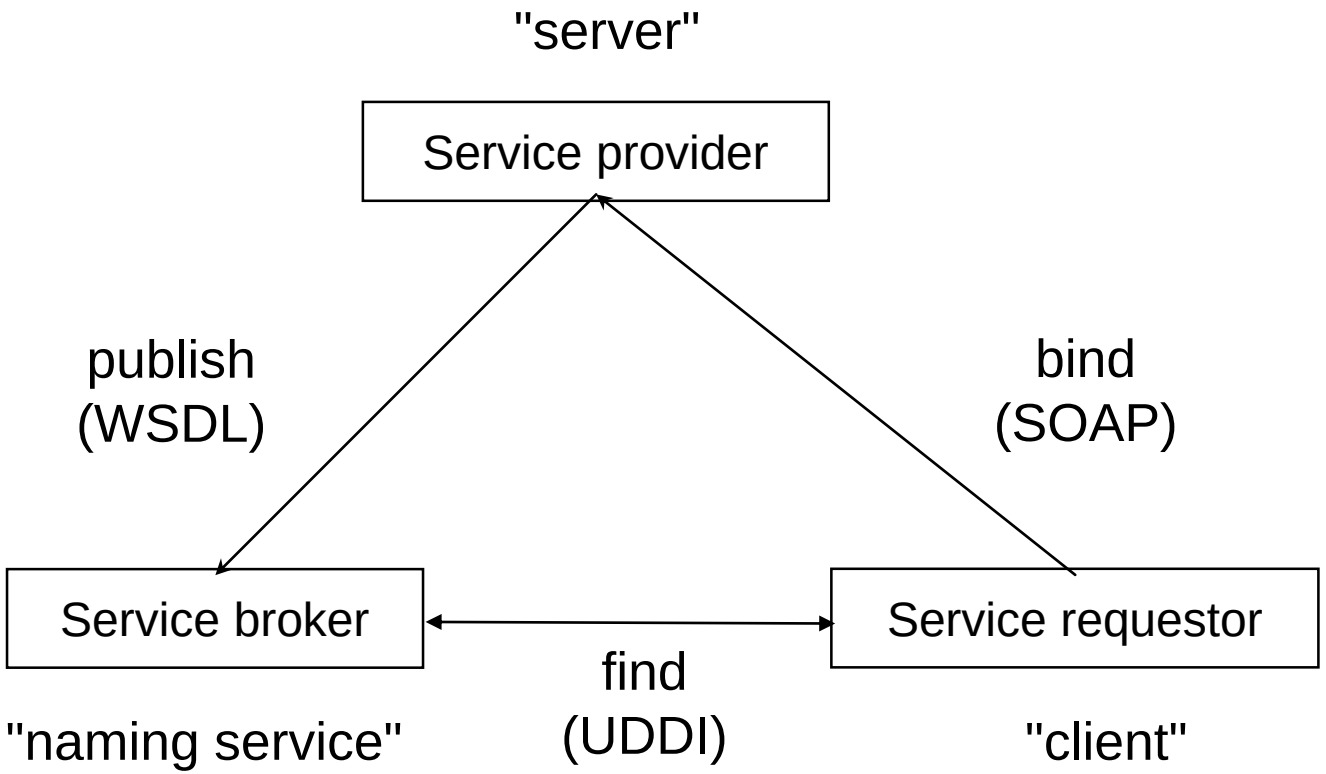
- A directory called **UDDI** (Universal Description, Discovery and Integration) defines what software system should be contacted for which type of data.
- When one software system needs one particular report/data, it would go to the UDDI and find out which other system it can contact for receiving that data.

Web Services - SOAP

- Once the software system finds out which other system it should contact, it would then contact that system using a special protocol called **SOAP** (Simple Object Access Protocol).

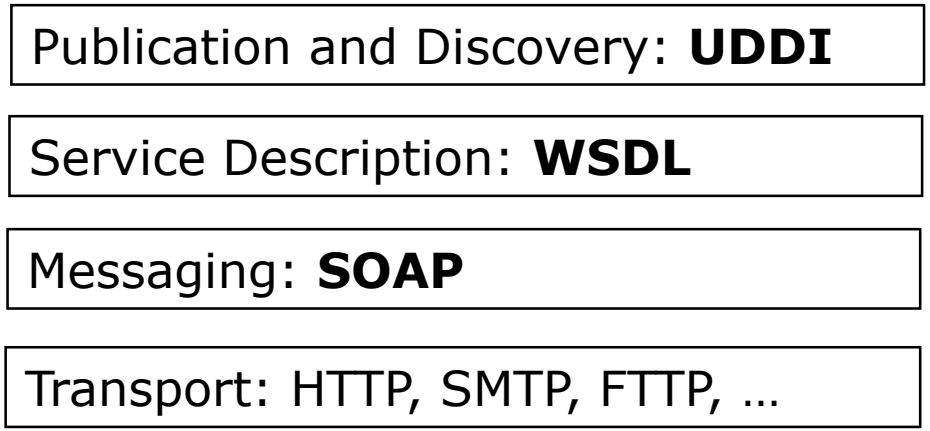
- The service provider system would first validate the data request by referring to the WSDL file, and then process the request and send the data under the SOAP protocol.

XML-Based Web Service Architecture

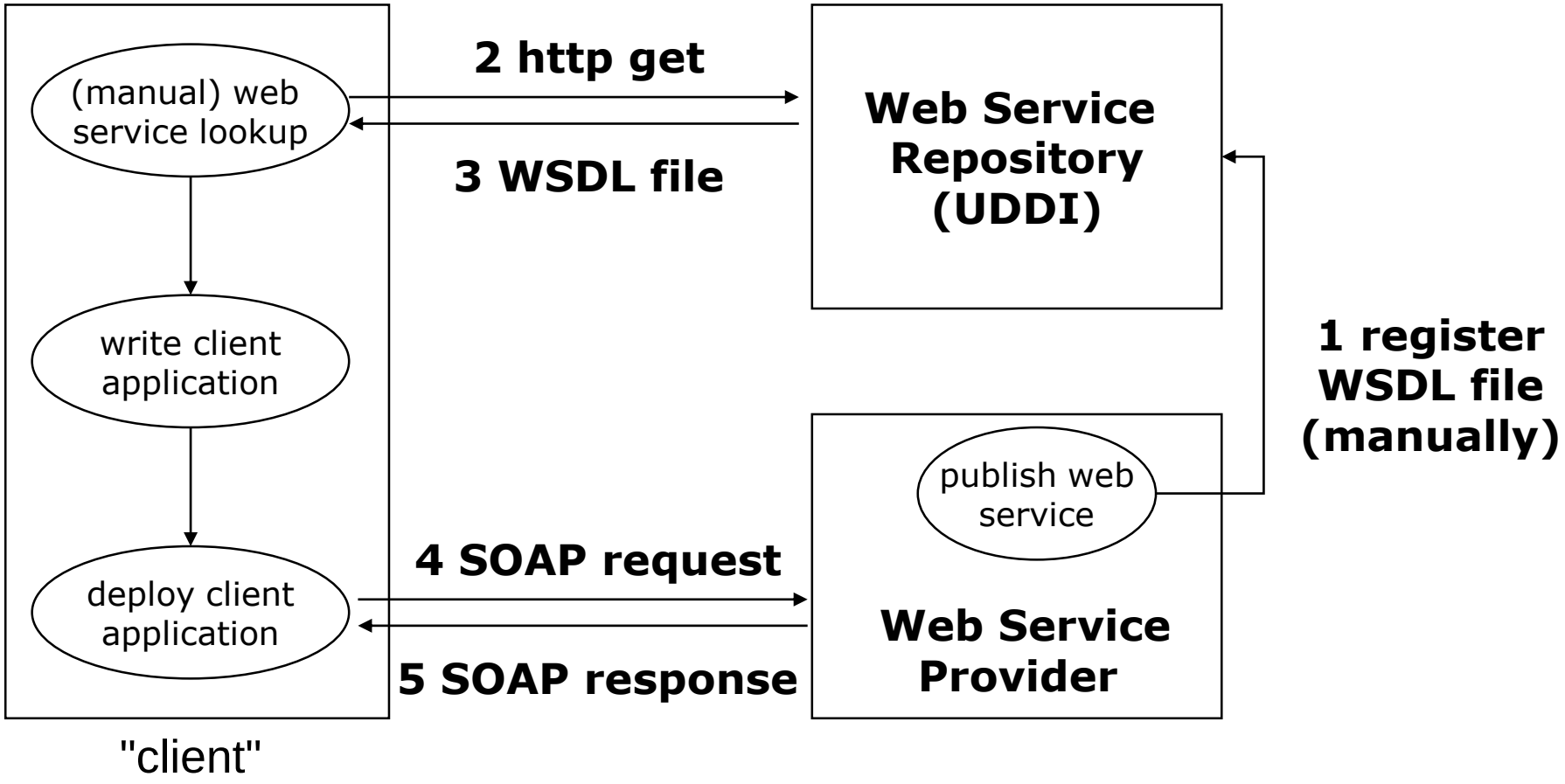


XML-Based Web Service Stack

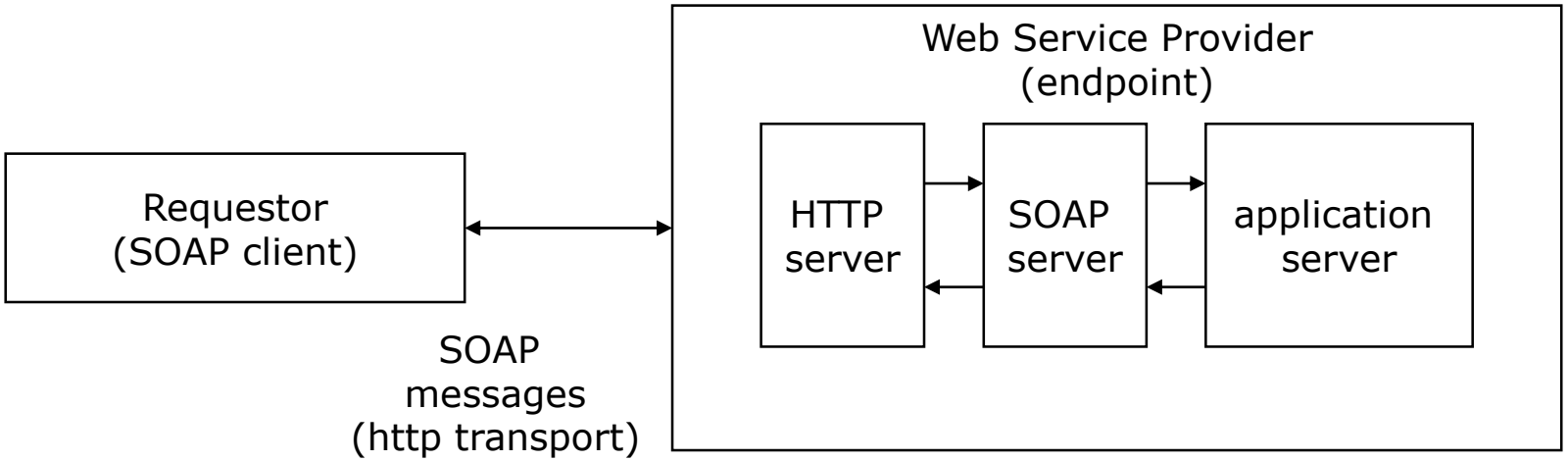
- A set of standards for implementing XML-based web services



Basic XML-Based Web Service



XML-Based Web Services Implementation



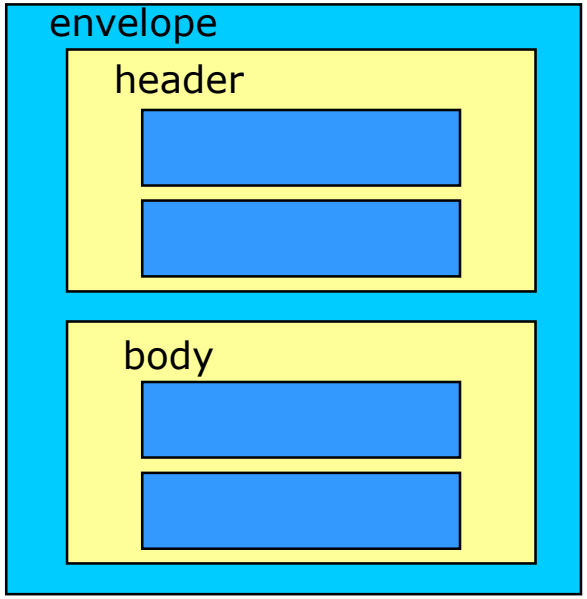
- Application Server (web service-enabled)
  - provides implementation of services and exposes it through WSDL/SOAP
  - implementation in Java, as EJB, as .NET (C#) etc.
- SOAP server
  - implements the SOAP protocol
- HTTP server
  - standard Web server
- SOAP client
  - implements the SOAP protocol on the client site

Simple Object Access Protocol (SOAP)

- Lightweight messaging framework based on XML
- Supports simple messaging and RPC
- SOAP consists of
  - Envelope construct: defines the overall structure of messages
  - Encoding rules: define the serialization of application data types
  - SOAP RPC: defines representation of remote procedure calls and responses
  - Binding framework: binding to protocols such as HTTP, SMTP
  - Fault handling

SOAP Message

- SOAP messages consist of
  - Envelope: top element of XML message (required)
  - Header: general information on message such as security (optional)
  - Body: data exchanged (required)
- Header
  - elements are application-specific
  - may be processed and changed by intermediaries or recipient
- Body
  - elements are application-specific
  - processed by recipient only



Skeleton SOAP Message

```
<?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>
  ...
  ...
</soap:Header>

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

</soap:Envelope>
```

Example: SOAP Message For Simple Message Transfer

Envelope

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2002/12/soap-envelope">
  <env:Header>
    <m:reservation xmlns:m=http://travelcompany.example.org/reservation
                   env:role=http://www.w3.org/2002/12/soap-envelope/role/next
                   env:mustUnderstand="true">
                   <m:dateAndTime>2001-11-29T13:20:00.000-05:00</m:dateAndTime>
    </m:reservation>
    <n:passenger xmlns:n=http://mycompany.example.com/employees
                 env:role=http://www.w3.org/2002/12/soap-envelope/role/next
                 env:mustUnderstand="true">
                 <n:name>Åke Jógvan Øyvind</n:name>
    </n:passenger>
  </env:Header>
  <env:Body>
    <p:itinerary xmlns:p="http://travelcompany.example.org/reservation/travel">
    <p:departure>
      <p:departing>New York</p:departing>
      <p:arriving>Los Angeles</p:arriving>
      <p:departureDate>2001-12-14</p:departureDate>
    </p:departure>
    <p:return>
      <p:departing>Los Angeles</p:departing>
      <p:arriving>New York</p:arriving>
      <p:departureDate>2001-12-20</p:departureDate>
    </p:return>
    </p:itinerary>
  </env:Body>
</env:Envelope>
```

SOAP attributes

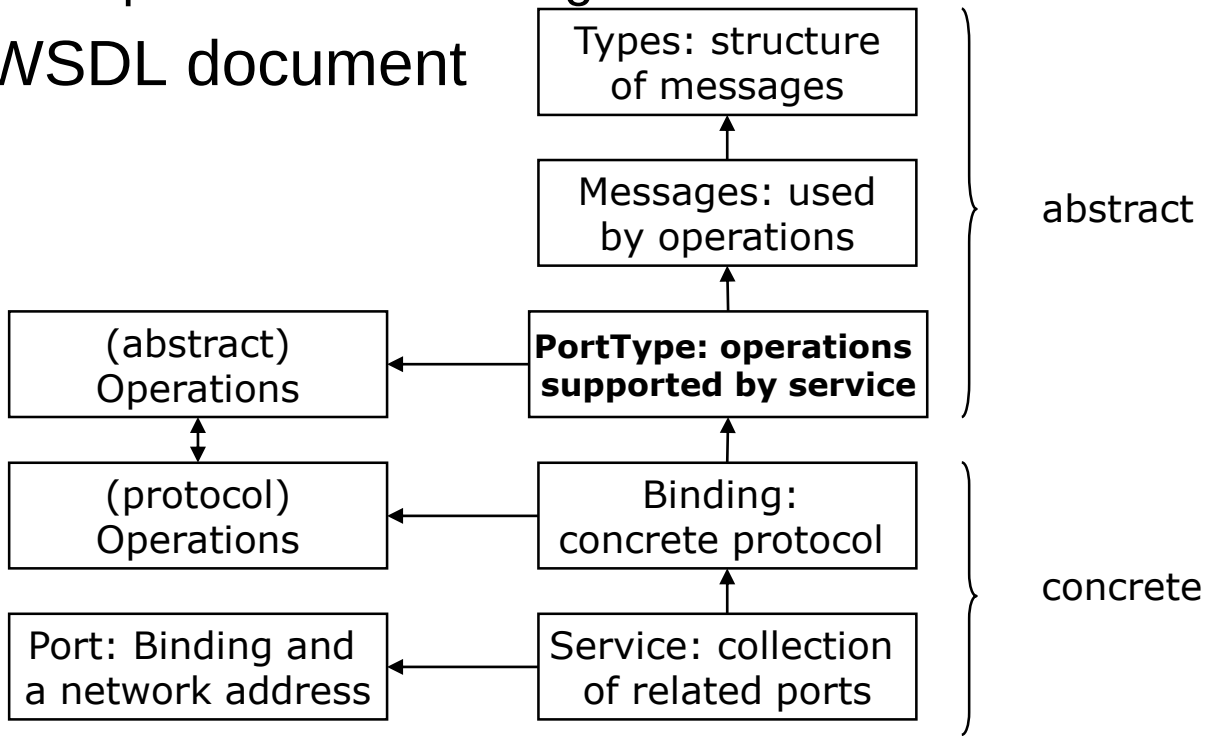
SOAP attributes

Header

Body

WSDL – Web Service Description Language

- Description of Web services in XML format
  - abstract description of operations and their parameters (messages)
  - binding to a concrete network protocol (e.g. SOAP)
  - specification of endpoints for accessing the service
- Structure of a WSDL document



Example: Overall Document Structure for WSDL

```
<?xml version="1.0">
   <definitions name="StockQuote>
   <types>
        <schema>
                definition of types in XML Schema ……
        </schema>
   </types>
   <message name="GetTradePriceInput">
        definition of a message....
   </message>
   <portType name="StockQuotePortType">
        <operation name="GetLastTradePrice">
                definition of an operation ……
        </operation>
   </portType>
   <binding name="StockQuoteSoapBinding">
        definition of a binding ……
   </binding>
   <service name="StockQuoteService">
        <port name="StockQuotePort">
                definition of a port ……
        </port>
   </service>
   </definitions>
```

http://www.w3schools.com/webservices/ws_wsdl_documents.asp

Videos

- SOAP Web Services 01 - Introduction To Web Services
- https://www.youtube.com/watch?v=mKjvKPlb1rA

- SOAP Web Services 03 - Writing a Web service Client: Stub generation
- https://www.youtube.com/watch?v=6hqDMS-oJ9k

- SOAP Web Services 04 - Writing a Web service Client: Calling the Service
- https://www.youtube.com/watch?v=KFlDdb65w3U

Videos

- SOAP Web Services 06 - Writing a Web Service: Eclipse setup
- https://www.youtube.com/watch?v=9kb0iLpqKY4&index=6&list=PLqq-6Pq4lTTZTYpk_1DOowOGWJMIH5T39

- SOAP Web Services 07 - Writing a Web Service: Code and Deploy
- https://www.youtube.com/watch?v=u5cQkVgq6jE&index=7&list=PLqq-6Pq4lTTZTYpk_1DOowOGWJMIH5T39

- SOAP Web Services 10 - Understanding the WSDLhttps://www.youtube.com/watch?v=E76xW1JTVXY

- SOAP vs. REST

https://www.youtube.com/watch?v=v3OMEAU_4HI