(Semantic) Web Services

M.-S. Hacid
University Claude Bernard, Lyon – France
http://www710.univ-lyon1.fr/~dbkrr

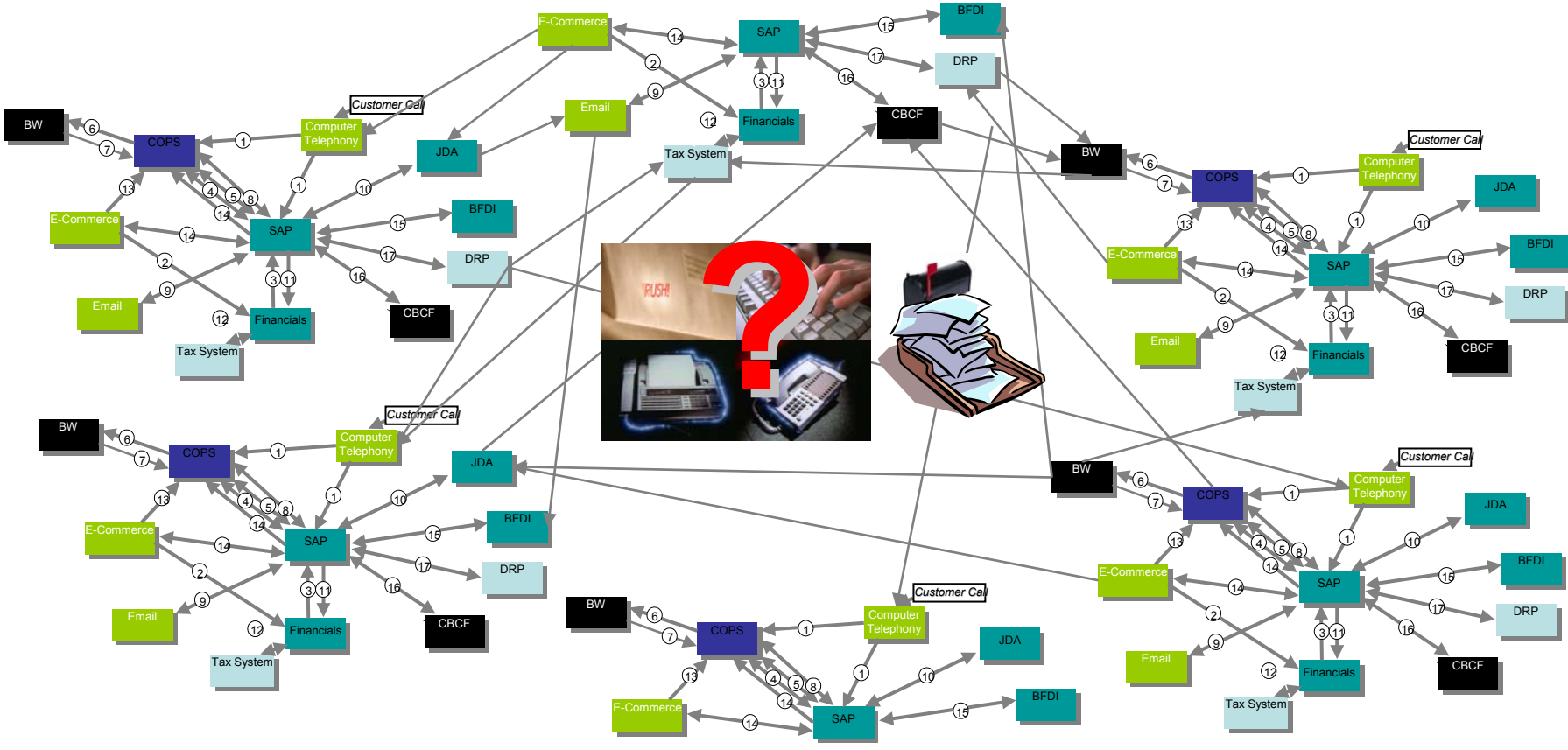
Enterprise Applications Integration (EAI

What is EAI?

Enterprise Applications Integration is a solution that supports real-time seamless access to information resident in a variety of repositories.

Business processing logic is extracted from application code and placed into an EAI tool where it is graphically represented and manipulated.

Today's Business Reality



Why EAI?

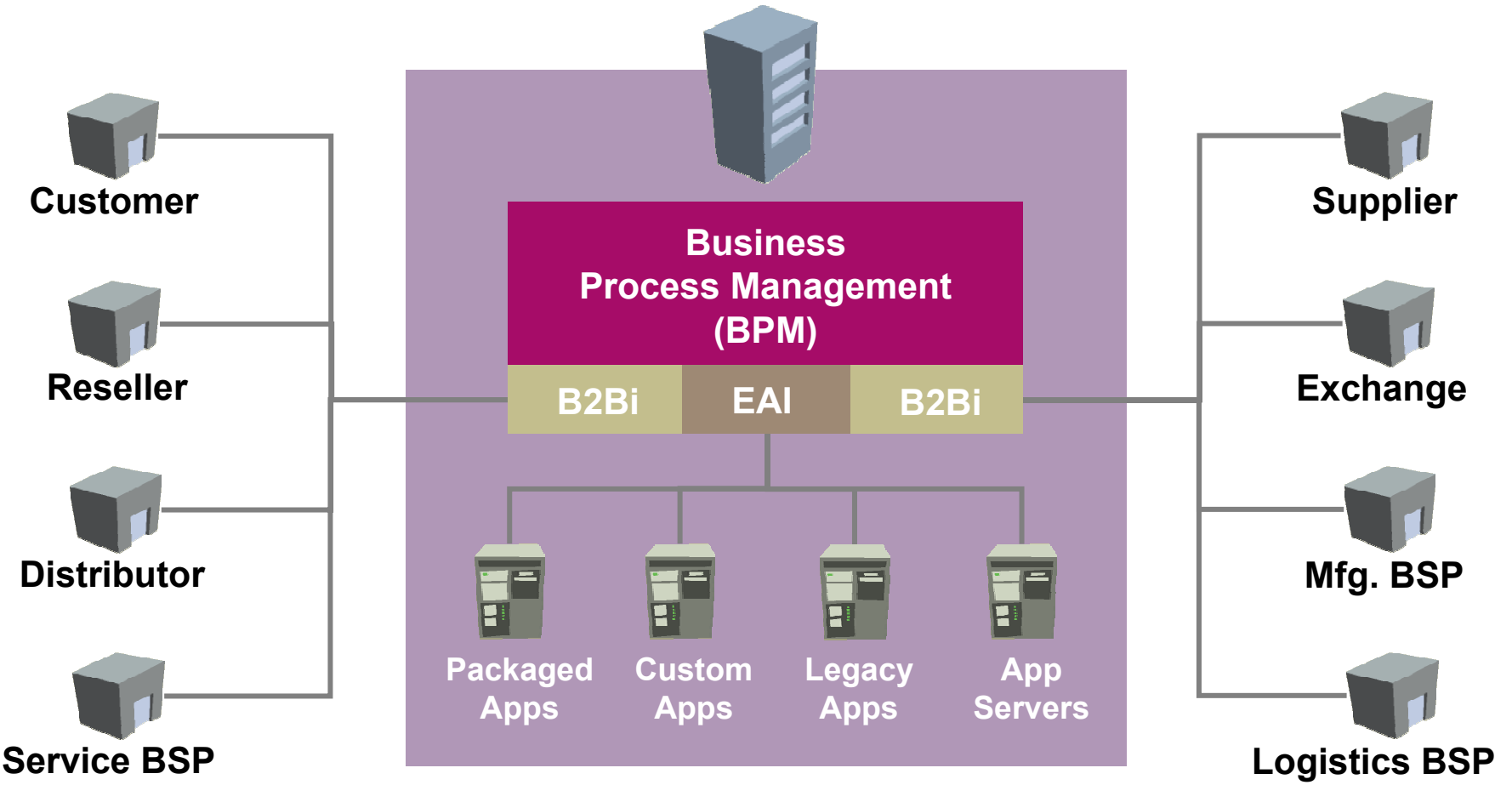
The needs for AI stem primarily from the following business and technical objectives:

1. **Integrate** with outside partner/customer (as part of a merger) or a «net new entity, which varies widely from a new set of e-commerce applications to supply chain integration.
2. **Migrate** towards a customer centric operating model to gain additional Insigl in customer behaviors and identify new revenue streams and cross-selling opportunities.
3. **Isolate** components of huge monolithic systems so they can be replaced or retired because the legacy systems become un-maintainable.
4. **Layer** an end-user application (especially portals, CRM, and Web-based self-service) that must access data across multiple systems and/or databases.
5. **Lower** total cost of ownership by reducing system management complexity and maintenance costs.

EAI provides a structured and efficient way to integrate not only he applications but also the ***business process***. EAI solutions ffer the following unique benefits:

1. **Reduced development and maintenance** cost (separation of business logic from transaction processing capability…).
2. **Enhanced performance and reliability** (asynchronous messaging mechanisms…).
3. **Centralized information bus** (unification of isolated applications…).
4. **Extension of legacy system lifecycle**.
5. **Reduced time to market** (customize existing business rules and extend application functionality).

Enterprise Integration Solution



Mediate the interactions between the applications to integrate

Customers, Distributors, Resselers

-Community
-Content
-Commerce

-Web Site
-Profiling
-Personalization
-Customization

Data

Distribution Channels

Rules Engine

Data

ustomer Service Relationship

Business Process Integration

Message Routing & Transformation

Message Transport

Product/Customer Analysis

RM
all Center
Mail
stant Messaging

Customer Service

Data

Information Systems

-Data Warehouse
-Data Analysis
-Targeted Marketing

Data

Partners

-Payroll
-HR
-Benefits
-Etc.

Operational Systems

Rules Metadata

-Order Management
-Warehouse Management
-Backend Support
-Billing.

Data

Web Services

*We look at Web services as a way to **expose** the functionality of an information system and make it available through **standard web technologies**. The use of standard technologies reduces heterogeneity, and is therefore key to facilitate **application integration**.*

Web services represent the first concerted effort that has gathered wide support for standardizing interactions across information systems.

Difficulties of integrating applications across the Internet:

1. Firewalls
2. Lack of standardized protocols
3. Need for loosely-coupled interactions
4. Etc.

Web Services and their Approach to Distributed Computing (main ingredients of Web services)

Defining Web services

Generic definition

A Web service is seen as an application accessible to other application over the Web [Fisher 2002, Menasce and Almeida 2001].

➔ Anything that has a URL is a Web service (ex. cgi script)

A program accessible over the Web with a stable API, published with additional descriptive information on some service directory.

M. Fisher. ***Introduction to Web Services.*** Part of the Java Web Services Tutorial. Aug. 2002. http://java.sun.com/webservices/docs/1.0/tutorial/.

D. Menasce and V. Almeida. ***Capacity Planning for Web Services***. Prentice Hall, 2001.

Definition by UDDI Consortium

A Web service is considered as a **self-contained, modular** business applicati
that has **open**, Internet-oriented, standards-based interface.

published interface that can be invoked across the Internet

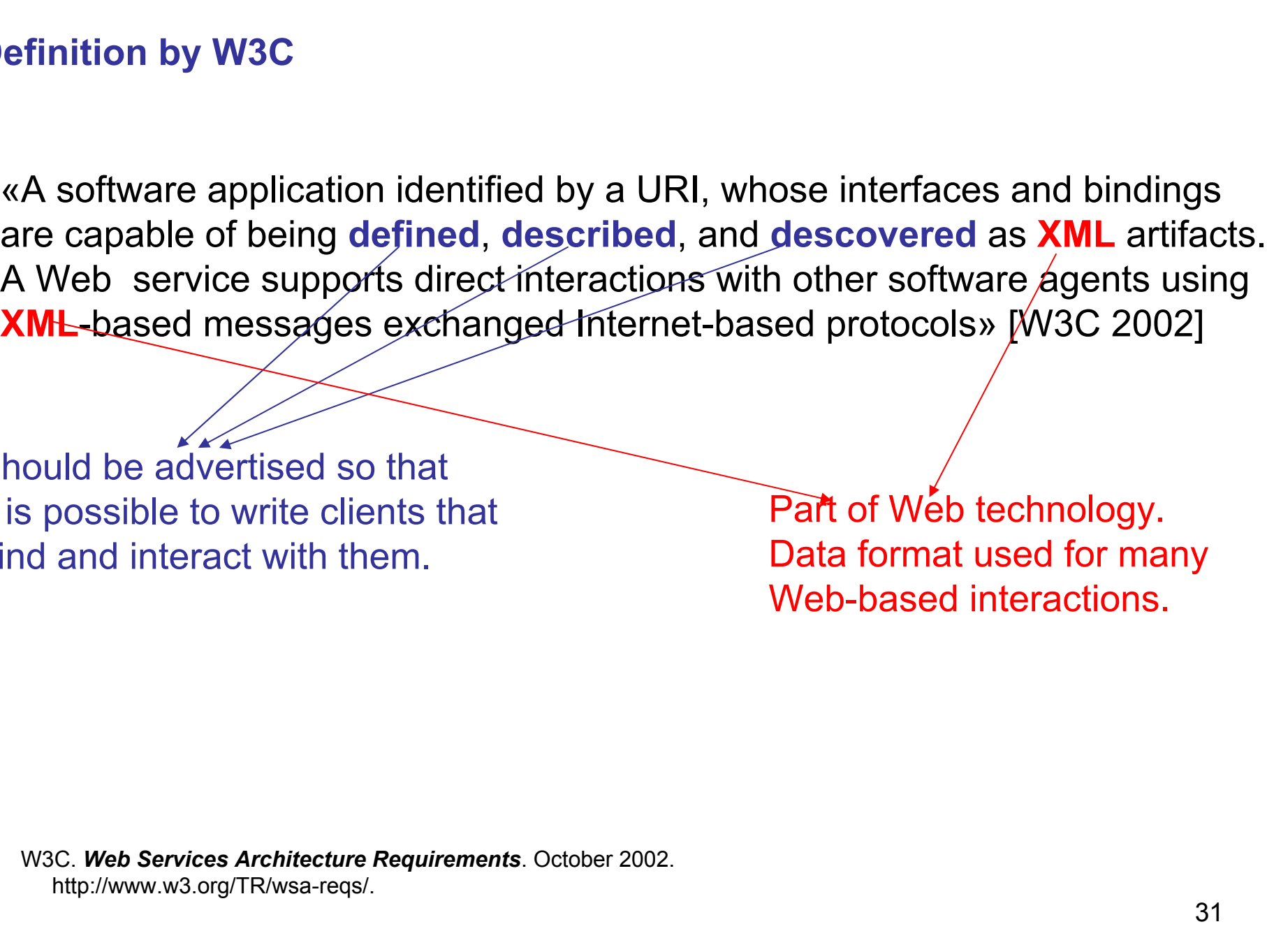
?

efinition by W3C

«A software application identified by a URI, whose interfaces and bindings are capable of being **defined**, **described**, and **descovered** as **XML** artifacts. A Web service supports direct interactions with other software agents using **XML**-based messages exchanged Internet-based protocols» [W3C 2002]

hould be advertised so that
is possible to write clients that
ind and interact with them.

Part of Web technology.
Data format used for many
Web-based interactions.



W3C. ***Web Services Architecture Requirements***. October 2002. http://www.w3.org/TR/wsa-reqs/.

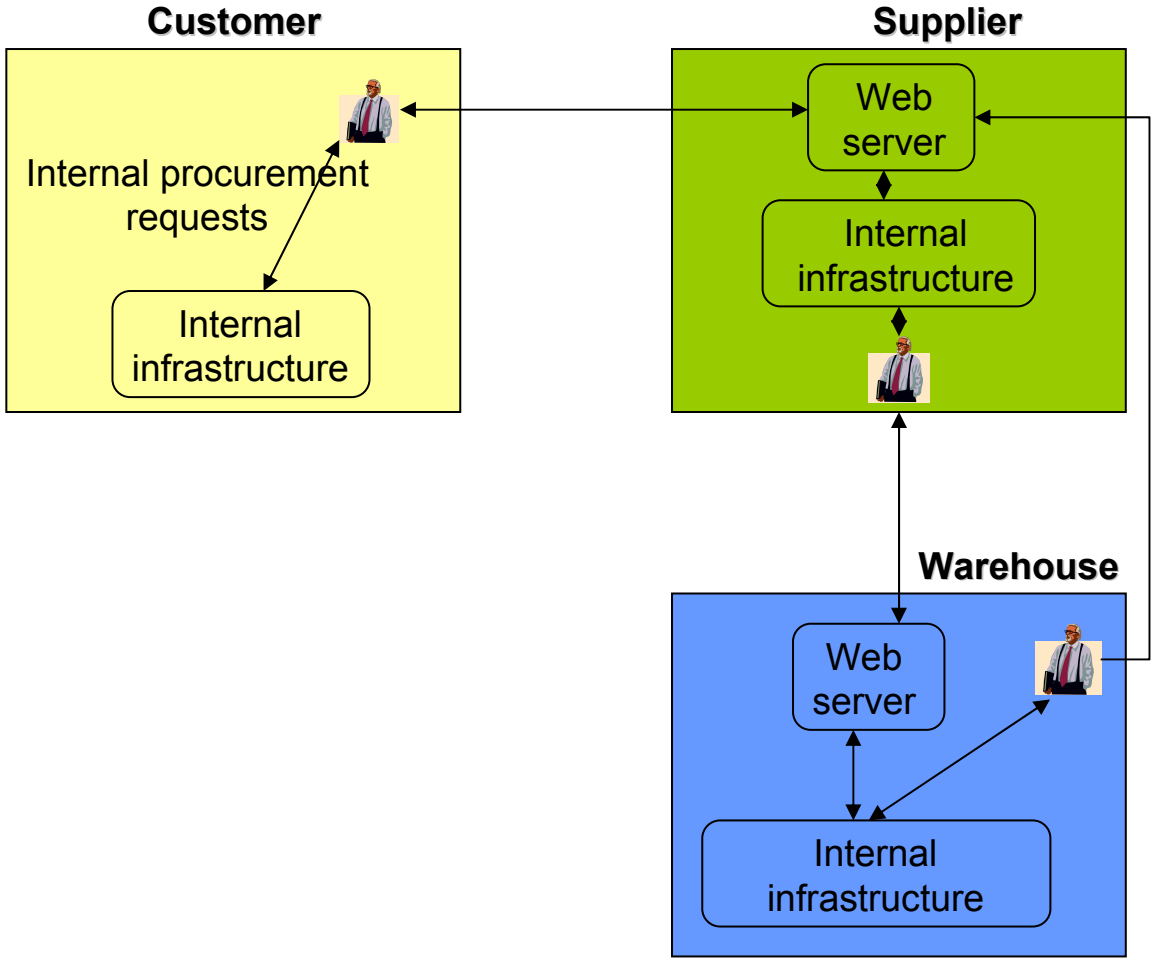
Another Definition [Jupitermedia Corporation]

A standardized way of integrating Web-based applications using the XML, SOAP, WSDL, and UDDI open standards over an Internet protocol backbon

- XML is used to tag the data
- SOAP is used to transfer the data
- WSDL is used for describing the services available
- UDDI is used for listing what services are available

Jupitermedia Corporation. ***Webopedia: Online Dictionary for Computer and Internet Terms***. http://www.webopedia.com/.

Example: B2B integration

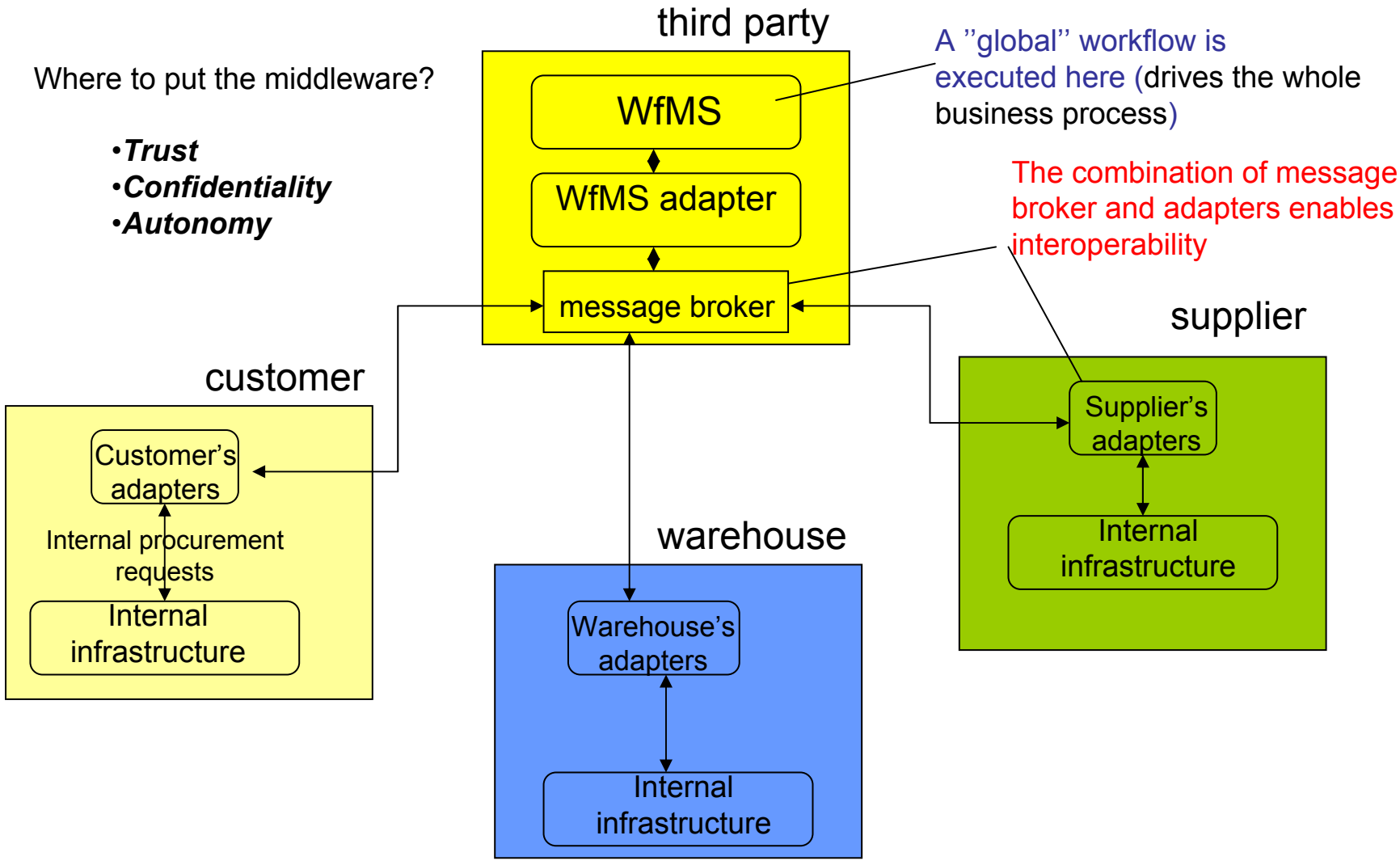


B2B interactions occur by accessing Web pages, filling Web forms, or via email.

Automation is driven by the goals:

- ***Lower costs***
- ***Streamlined and more efficient process***
- ***Ability to monitor and track process executions***
- ***Ability to detect and manage exceptions***

Limitations of Conventional Middleware in B2B Integration



The Web brought

- Standard interaction protocols (HTTP)
- Data formats (XML)

Adopted by many companies

Creation of a basis for establishing a common middleware infrastructure that reduces the heterogeneity among interfaces and systems.

B2B integration with Web Services

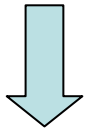
Three main aspects

- Service-oriented architectures.
- Redesign of middleware protocols.
- Standardization.

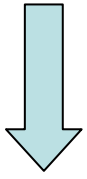
ervice-oriented paradigm

Assumption

The functionality made available by a company will be exposed as a servic



A service is a procedure, method, or object with a stable, published interfac
that can be invoked by clients



Requesting and executing a service involves a program calling another progra

Services are loosely-coupled

liddleware protocols

Veb services ➔ redesign of the middleware protocols to work in a peer-to-pee
fashion and across companies.

In conventional middleware: lack of trust and confidentiality issues often mak
a case against a central coordinator.

needs to be redesigned to allow more flexibility in terms of locking resources.

tandardization

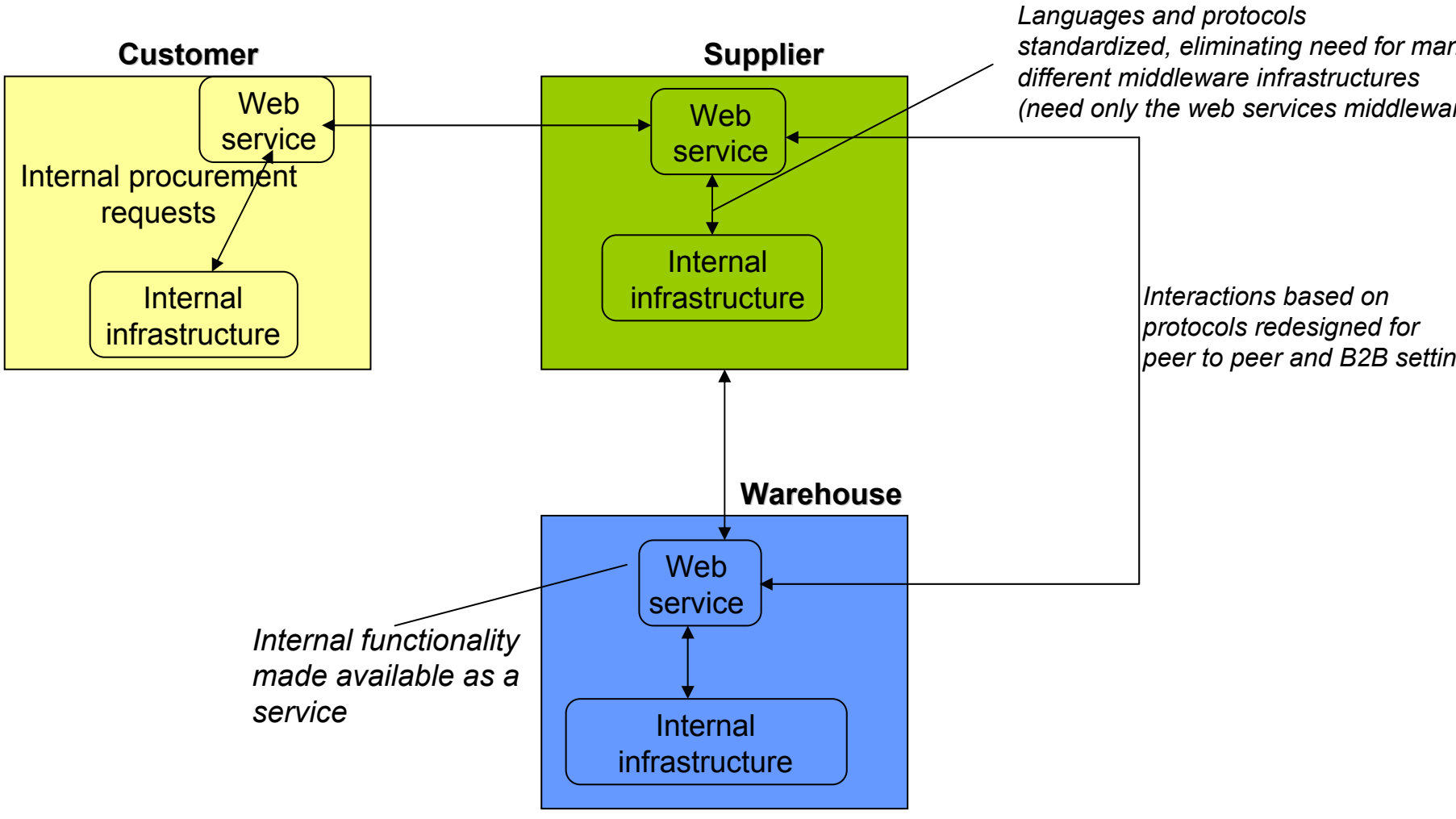
In conventional application integration: CORBA and Java enabled the development of portable applications.

Service-oriented architecture
Redefinition of middleware protocols } Not sufficient ⟹ standardization

OASIS (Organization for the Advancement of Structured Standard
W3C

B2B integration is what generated the need for web services

It is possible to make web services available to clients residing on a local LAN



No centralized coordination!

However, the challenge and ultimate goal of web services is inter-company interaction (a long-term goal!)

Web Services Technologies

The first required issues:

- What exactly a service is?
- How it can be described?

Service description in conventional middleware is based on interfaces an interface definition languages (IDL).

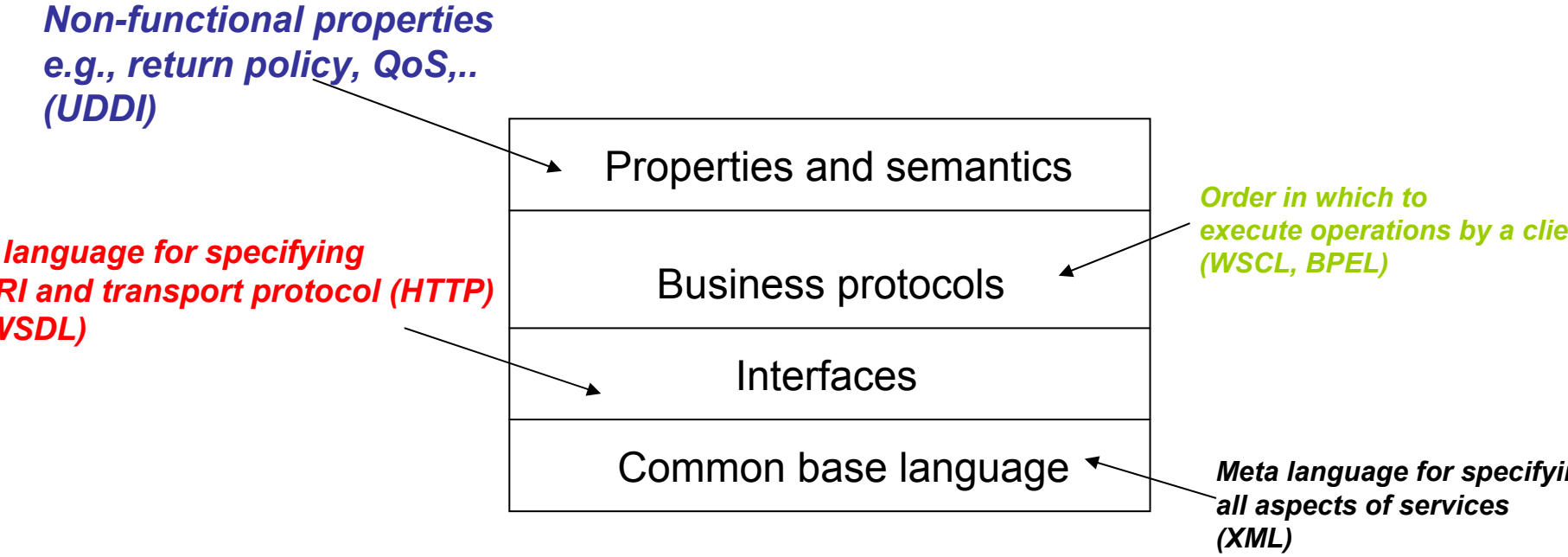
Implicit context:

- Clients and services are developed by the same team.
- Semantics of operations + order of invocation known in advance.
- The middleware platform defines and constrains many aspects of the servi description and binding process.

In web services and B2B interaction ➔ no such implicit context!

⟹ service descriptions must be **richer** and **more detailed**.

Service description and discovery stack



WSDL: Web Services Description Language

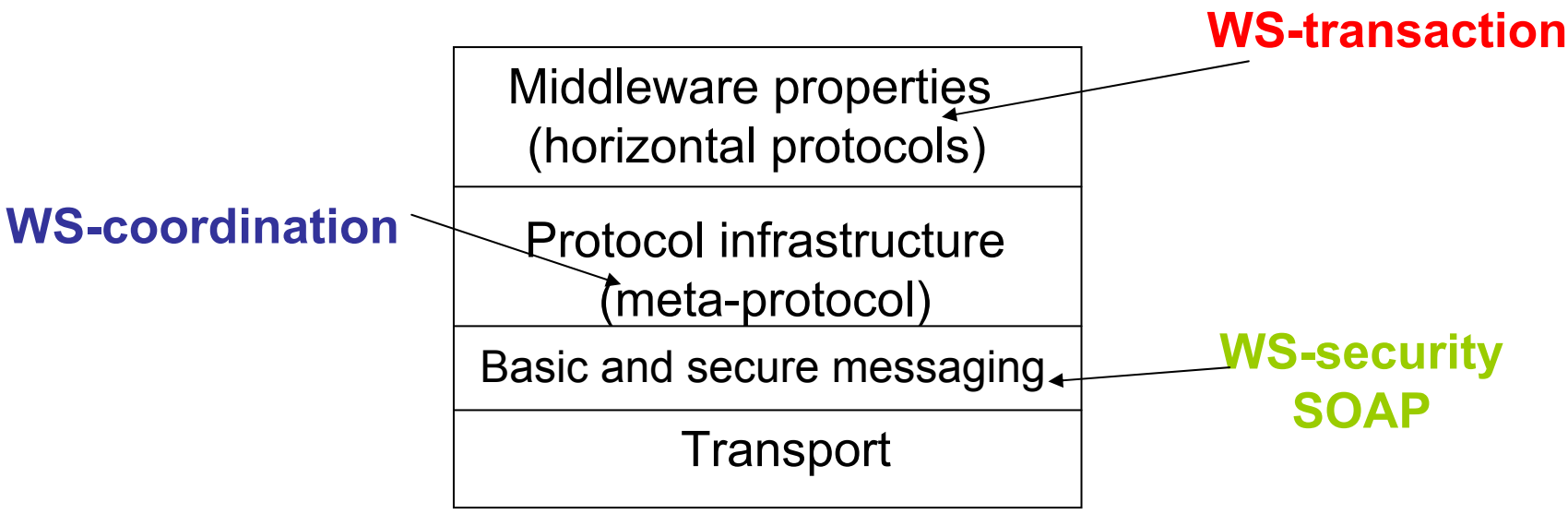
WSCL: Web Services Conversation Language
BPEL: Business Process Execution Language

UDDI:Universal Description, Discovery and Integration

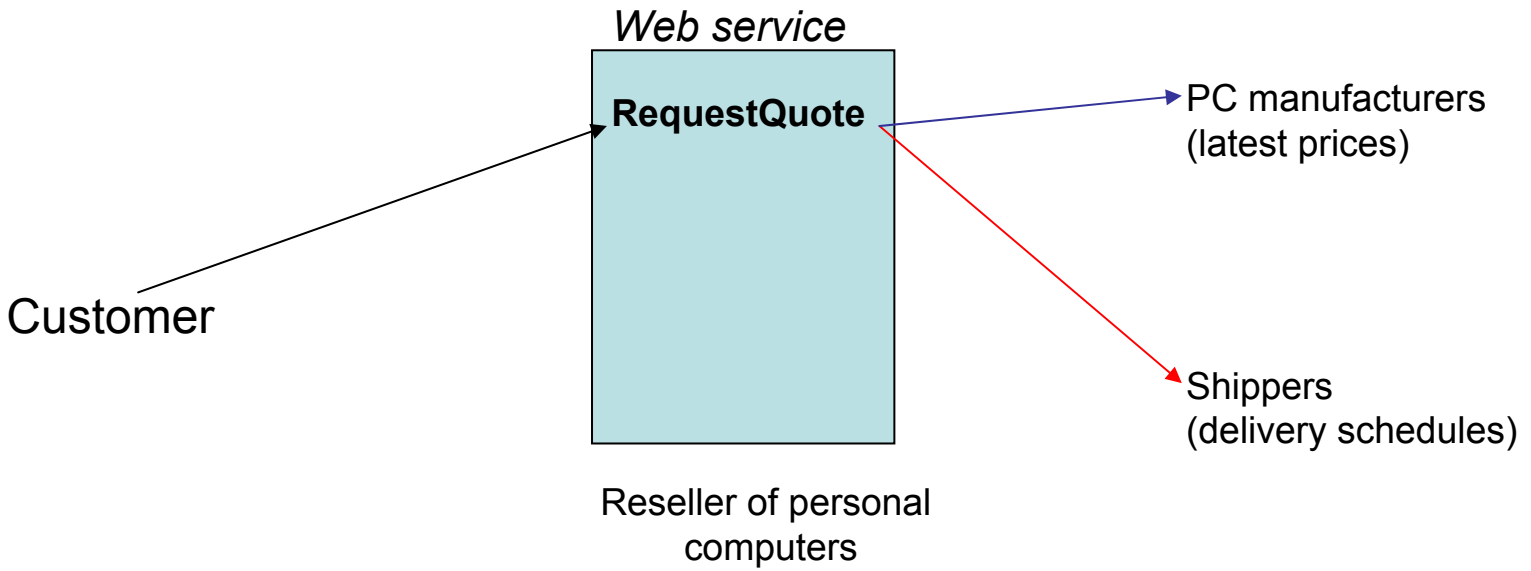
Service discovery

- At design-time (static binding)
- At run-time (using dynamic binding techniques)

Service interactions (a set of abstractions and tools that enable interactions among services

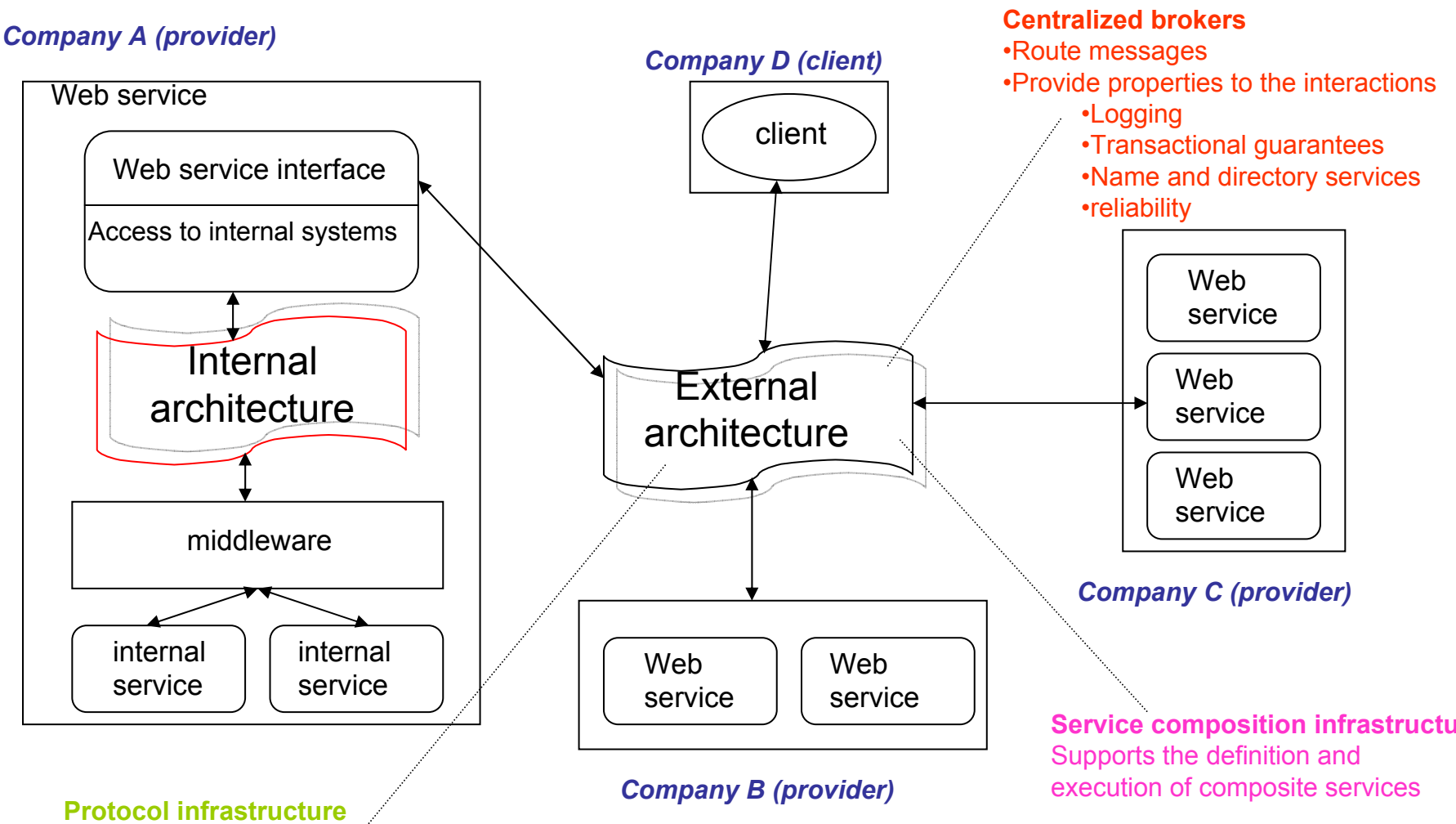


Combining Web Services : Composition

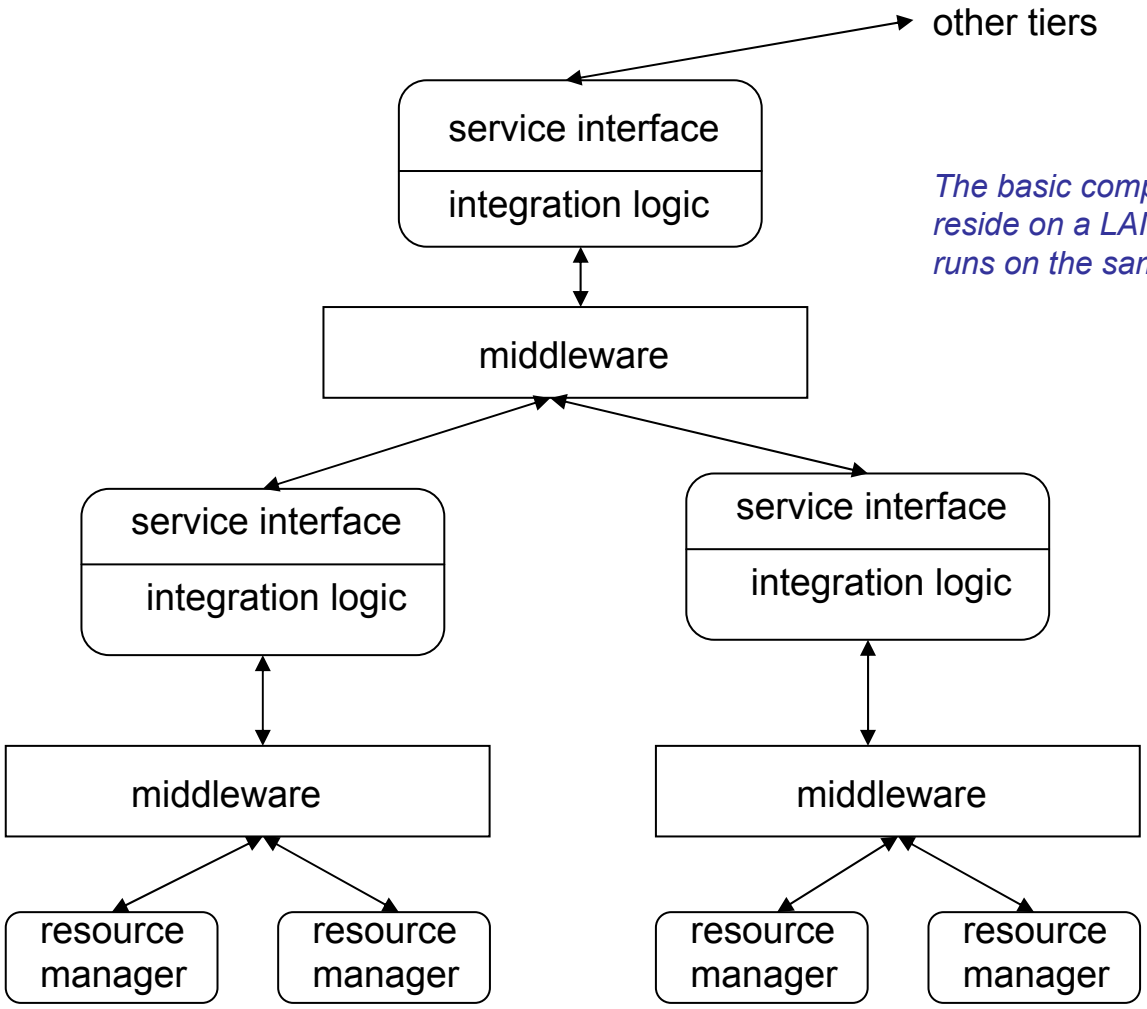


Web Services Architectures

Web services are a way to expose internal operations so that they can be invoked through the web.



Internal Architecture of a Web Service



The basic components of each middleware instance reside on a LAN and the resulting application also runs on the same LAN.

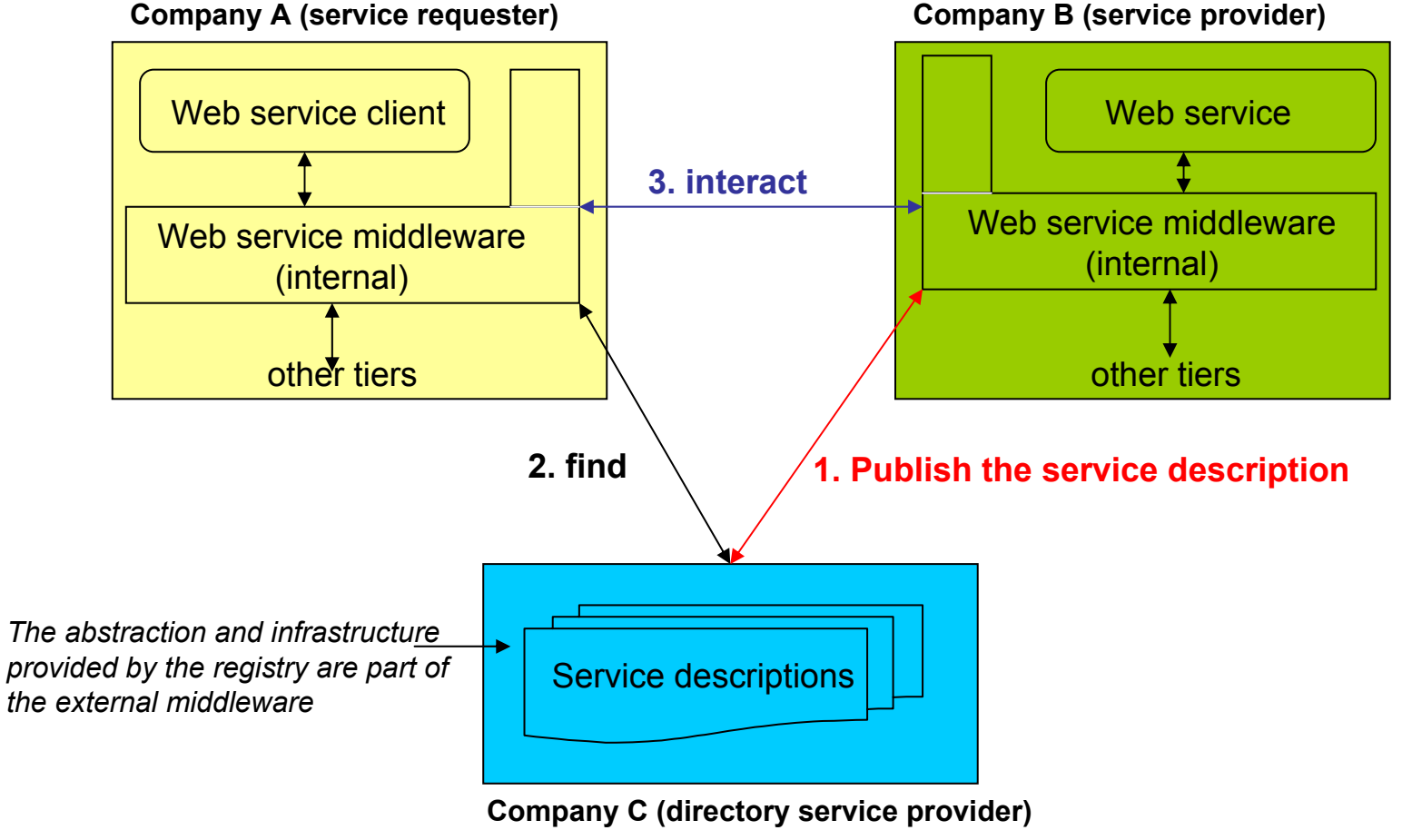
xternal Architecture of a Web Service

Wrapping internal functionality as a Web Service ➔ brokers and workflow management systems in the case of conventional middleware

External middleware for web services ➔ where this middleware should reside?

Two solutions:

1. Implement the middleware as a peer-to-peer system (appealing but problem of reliability and trustworthiness)
2. Introduce intermediaries or brokers acting as the necessary middleware.


Company A (service requester)
Company B (service provider)
Web service client
Web service middleware (internal)
other tiers
Web service
Web service middleware (internal)
other tiers
3. interact
2. find
1. Publish the service description
The abstraction and infrastructure provided by the registry are part of the external middleware
Service descriptions
Company C (directory service provider)

Basic Web Services Technology

Web services architectures are mainly based on three components:

1. The service requester
2. The service provider
3. The service registry

Thereby closely following a client/server model with a explicit name and directory service

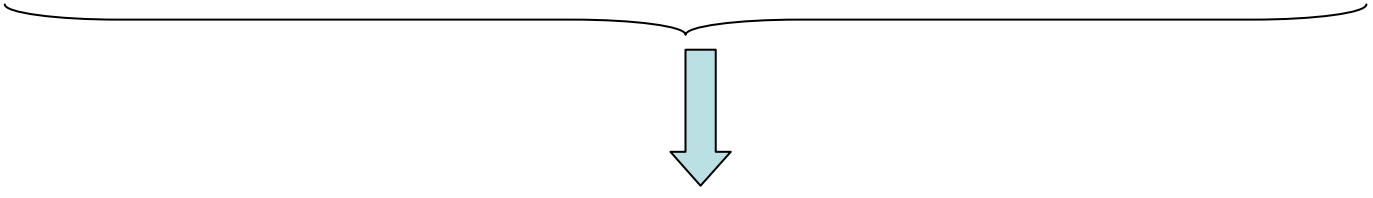
Basic infrastructure necessary to implement web services:

- A way to communicate (SOAP)
- A way to describe services (WSDL)
- A name and directory server (UDDI)

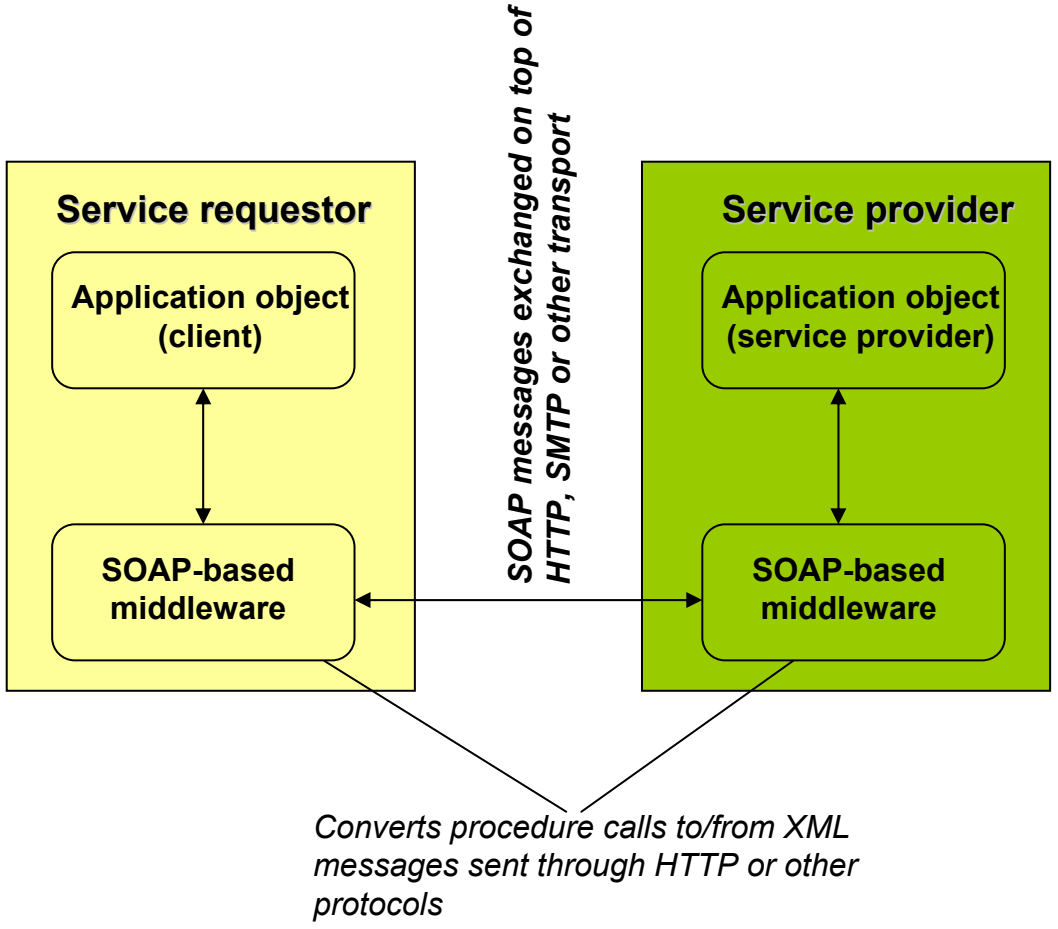
} Core of web services

A minimalist infrastructure for web services

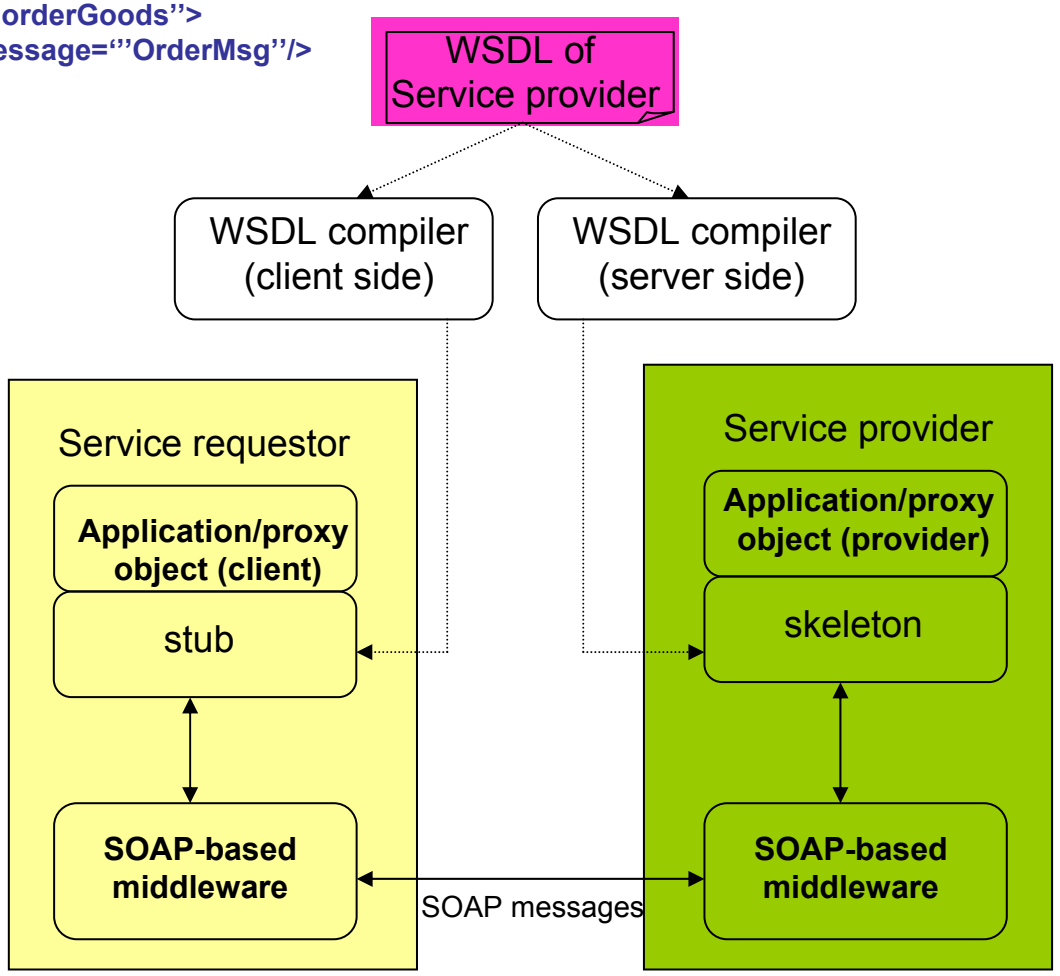
1. Common syntax for all specifications (XML)
2. A mechanism to allow remote sites to interact with each other
    a. A common data format for the messages being exchanged
    b. A convention for supporting specific forms of interaction (messaging or RPC)
    c. A set of bindings for mapping messages into a transport protocol (TCP/IP, HTTP, SMT

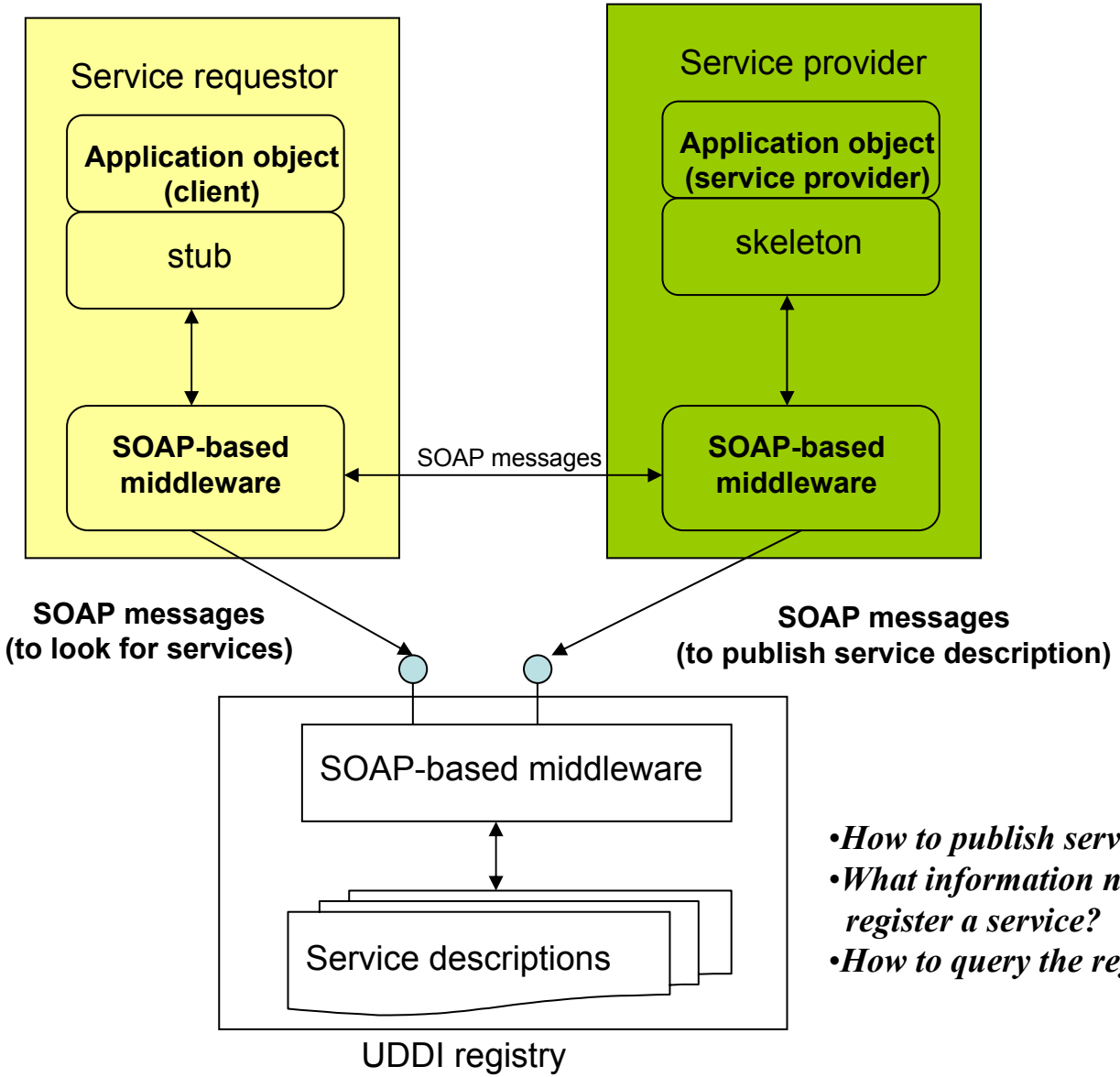


Messages as basic unit of communication


Service requestor
Application object (client)
SOAP-based middleware
SOAP messages exchanged on top of HTTP, SMTP or other transport
Service provider
Application object (service provider)
SOAP-based middleware
Converts procedure calls to/from XML messages sent through HTTP or other protocols

```
peration name=''orderGoods''>
        <input message='''OrderMsg''/>
peration>
```





- *How to publish services?*
- *What information needs to be provided to register a service?*
- *How to query the registry?*

SOAP : Simple Object Access Protocol

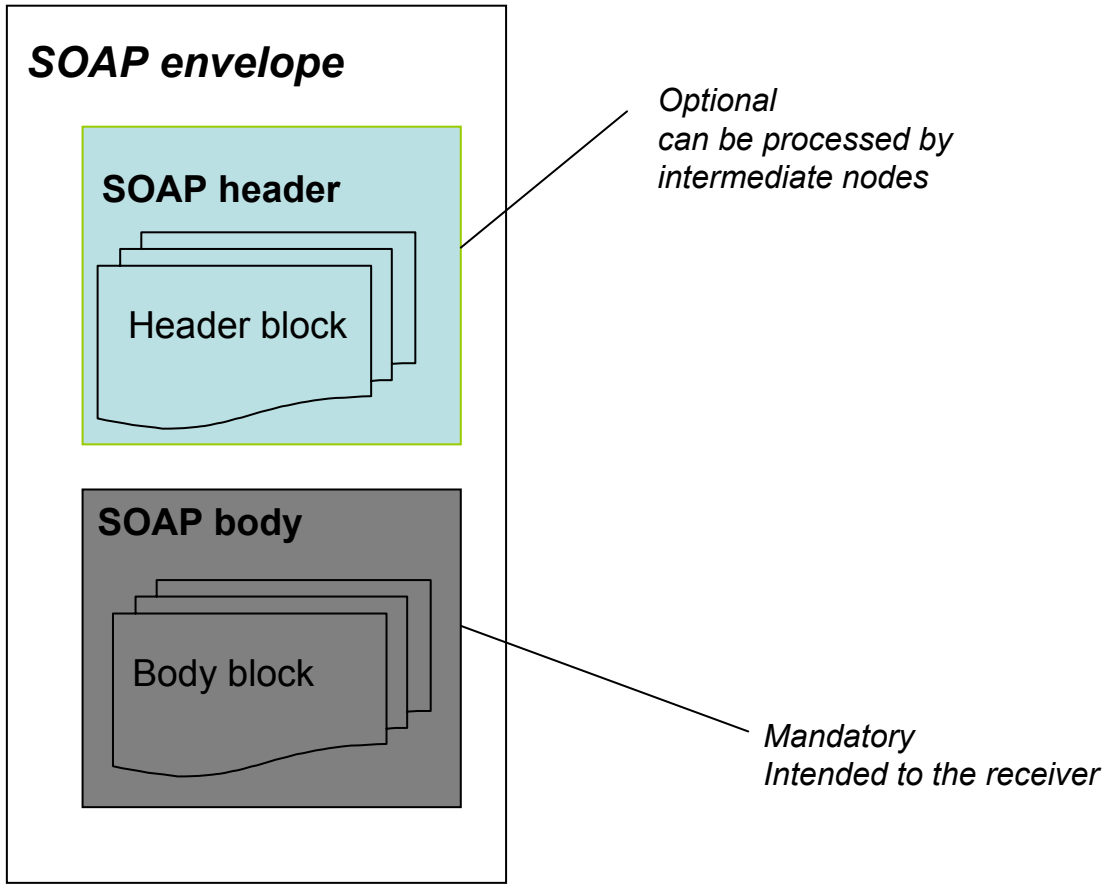
A joint effort from Canon, IBM, Microsoft and SUN

First version (1999) based on HTTP
Current version (2003) XML encoding

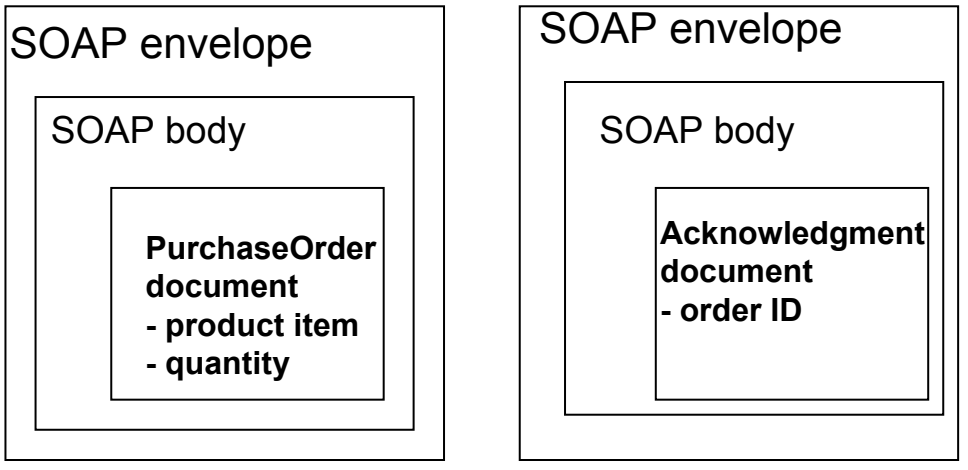
SOAP defines how to organize information using XML in a structured and typed manner so that it can be exchanged between peers.

- A message format describing how information can be packaged into an XML document.
- A set of conventions for using SOAP messages to implement the RPC interaction pattern, defining how clients can invoke a remote procedure by sending a SOAP message and how services can reply by sending another SOAP message back the caller.
- A set of rules that any entity that processes a SOAP message must follow.
- A description of how a SOAP message should be transported on top of HTTP and SMTP.

Schematic Representation of a SOAP message

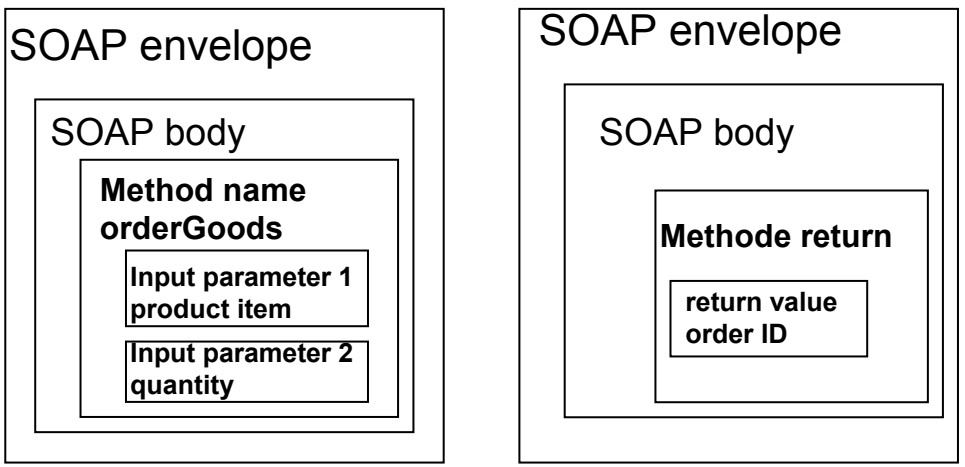


*greement on the structure
f the document*



Document-style interaction

*greement on the RPC method
gnature*



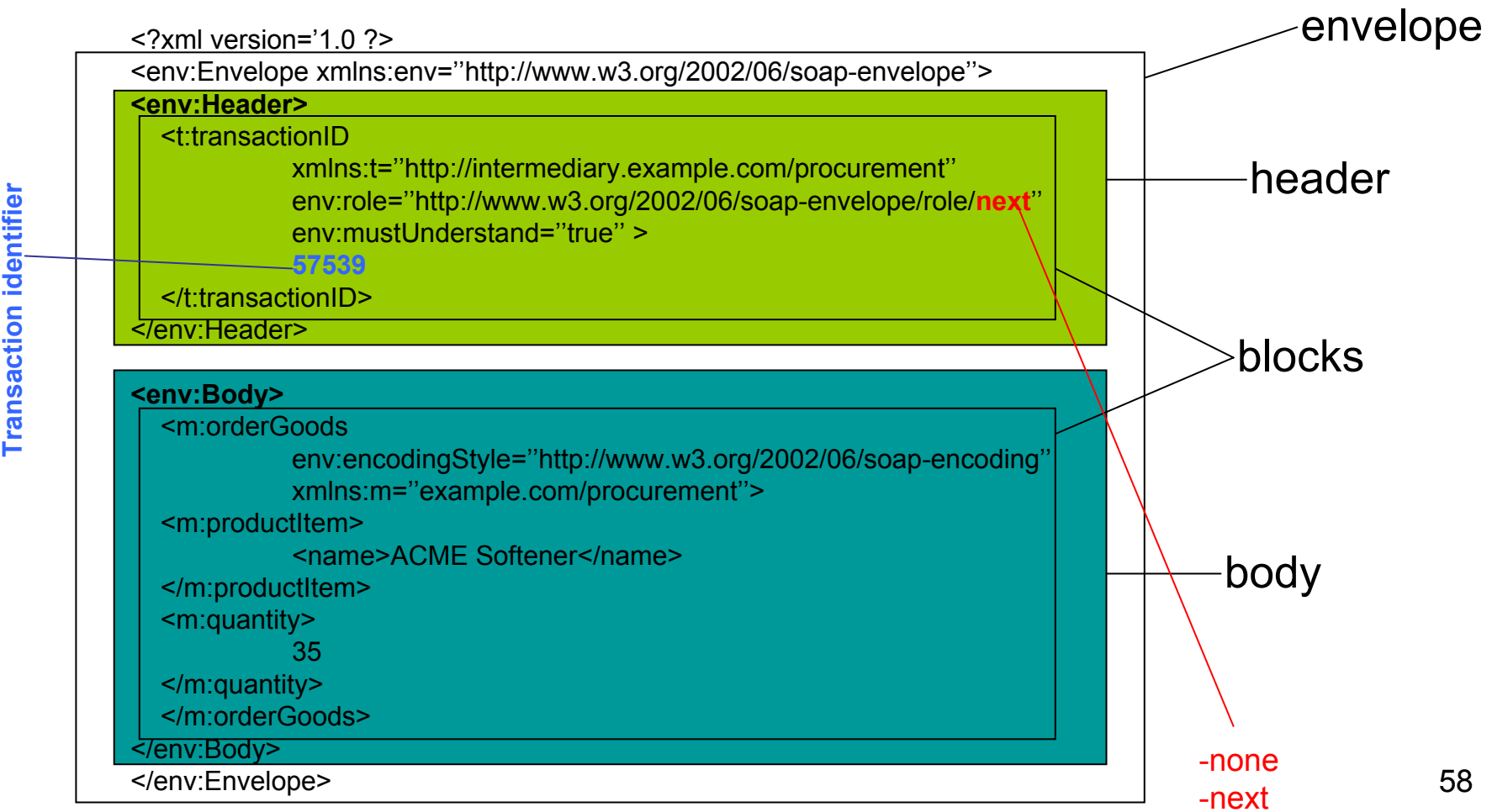
RPC-style interaction

The structure of a SOAP message is also influenced by encoding rules, which defin
how a particular entity or data structure is represented in XML.

```
<ProductItem>
    <name>…</name>
    <type>…</type>
    <make>…</make>
</ProductItem>
```

```
<ProductItem
    name=”…”
    type=”…”
    make=”…”
/>
```

```
<ProductItem name=”…”
    <type>…</type>
    <make>…</make>
</ProductItem>
```



```
<!-- request GetLastTradePriceInput is of type TradePriceRequest -->
<wsdl:message name="GetLastTradePriceInput">
        <wsdl:part name="body" element="xsd1:TradePriceRequest"/>
</wsdl:message>

<!-- request GetLastTradePriceOutput is of type TradePrice -->
<wsdl:message name="GetLastTradePriceOutput">
        <wsdl:part name="body" element="xsd1:TradePrice"/>
</wsdl:message>

<!-- wsdl:portType describes messages in an operation -->
<wsdl:portType name="StockQuotePortType">

<!-- the value of wsdl:operation eludes me -->
        <wsdl:operation name="GetLastTradePrice">
                <wsdl:input message="tns:GetLastTradePriceInput"/>

                <wsdl:output message="tns:GetLastTradePriceOutput"/>
        </wsdl:operation>
 </wsdl:portType>
```

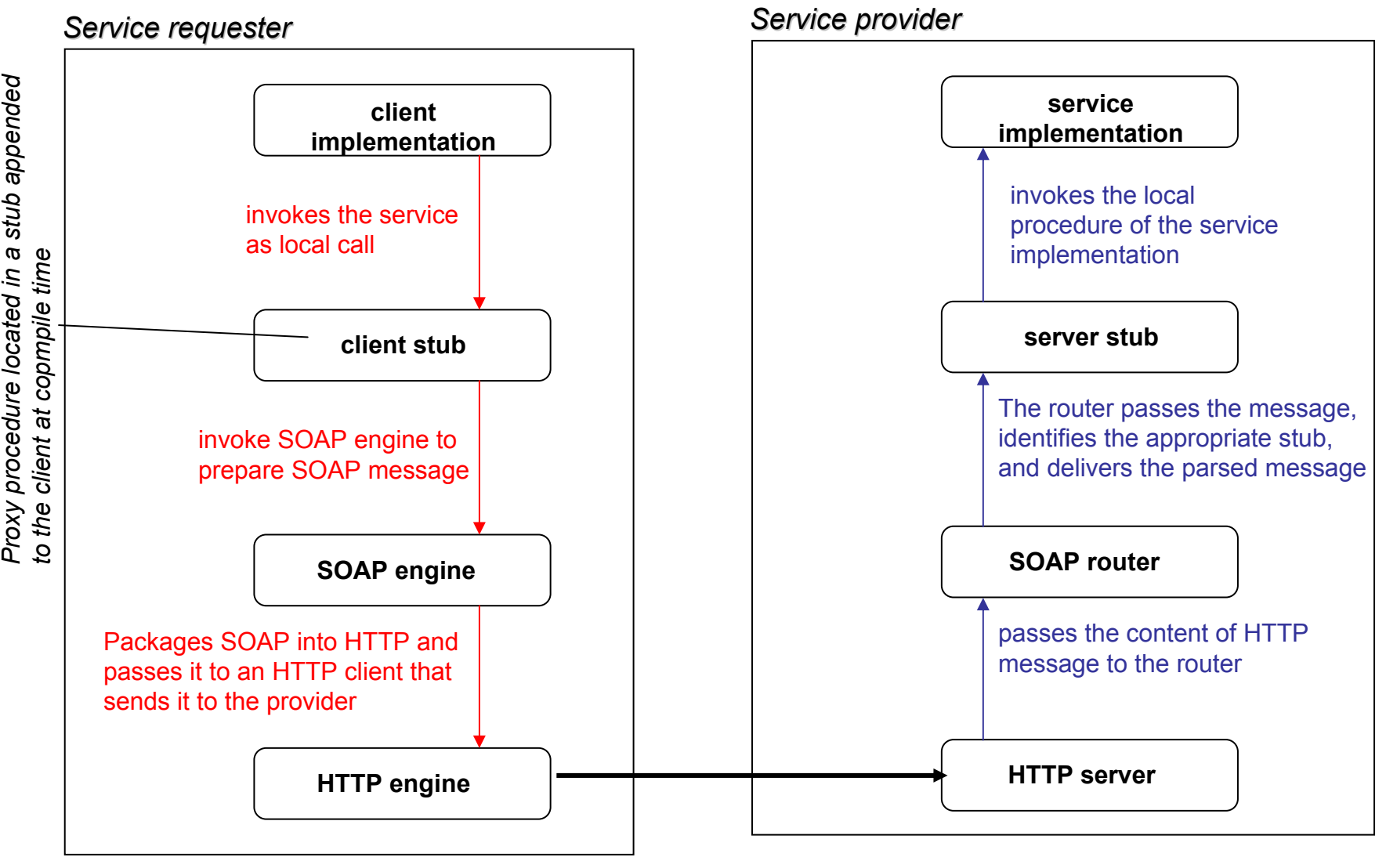
SOAP Message Embedded in HTTP Request

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
        <soapenv:Body>
                <m:GetLastTradePrice xmlns:m="Some-URI">
                        <m:tickerSymbol>DIS</m:tickerSymbol>
                </m:GetLastTradePrice>
        </soapenv:Body>
</soapenv:Envelope>
```

SOAP Message Embedded in HTTP Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
        <soapenv:Body>
                <m:GetLastTradePriceResponse xmlns:m="Some-URI">
                        <m:price>34.5</m:price>
                </m:GetLastTradePriceResponse>
        </soapenv:Body>
</soapenv:Envelope>
```

A Simple Implementation of SOAP



WSDL: Web Services Description Language

Originally created by IBM, Microsoft, and Ariba
WSDL = merge of three previous proposals:

- Microsoft SOAP Contract Language (SCL) and Services Description Language (SDL)
- IBM's Network Accessible Service Specification Language (NASSL)

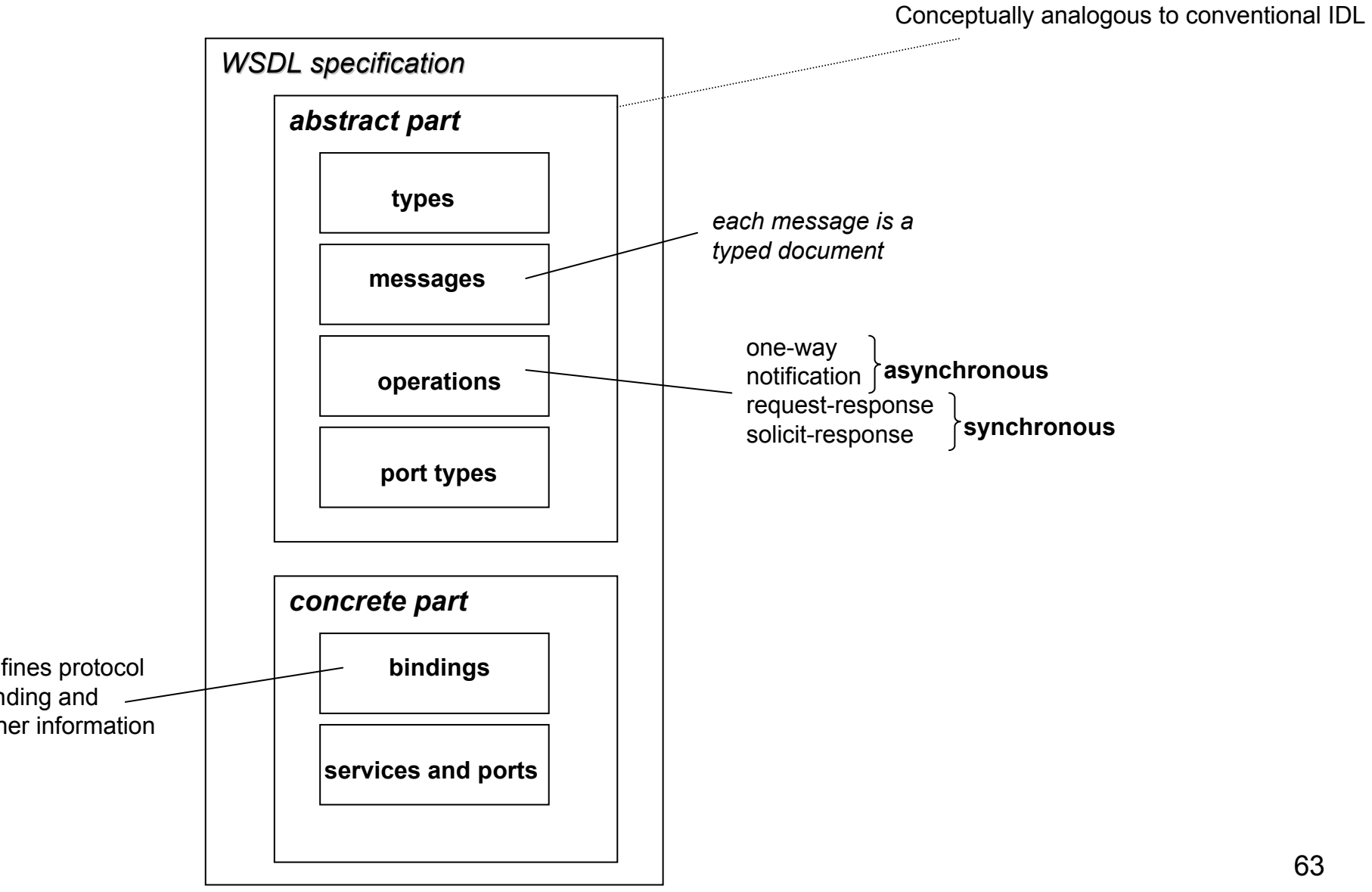
In WSDL, specifications are XML documents that describe Web services (service interfaces); that is ***operations*** offered by a Web service.

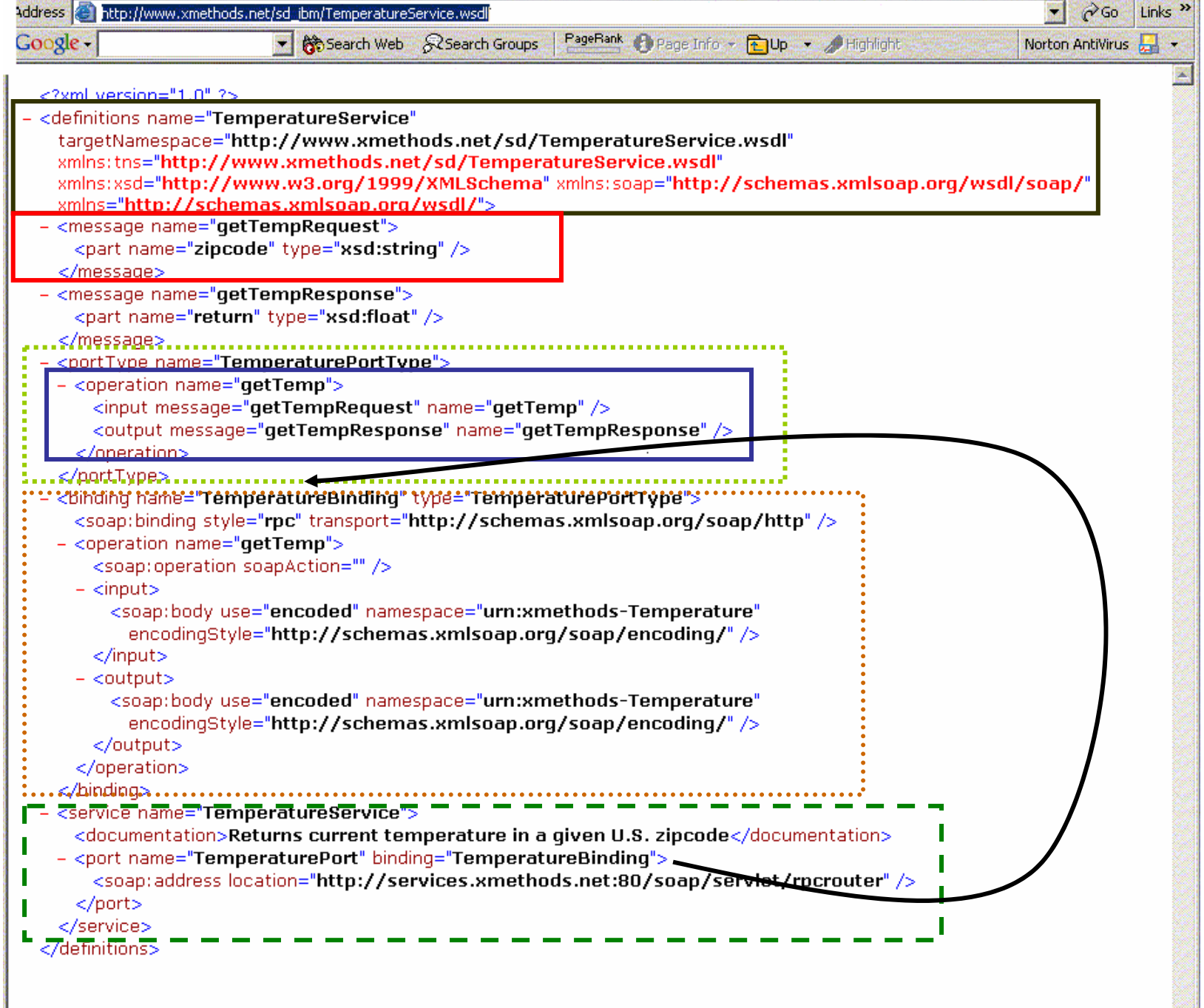
Existing IDLs are tied to a concrete middleware platform.

WSDL also needs to define the ***mechanisms*** to access the Web service.

Lack of a common middleware platform ➔ the need for defining the location at which the service is available.

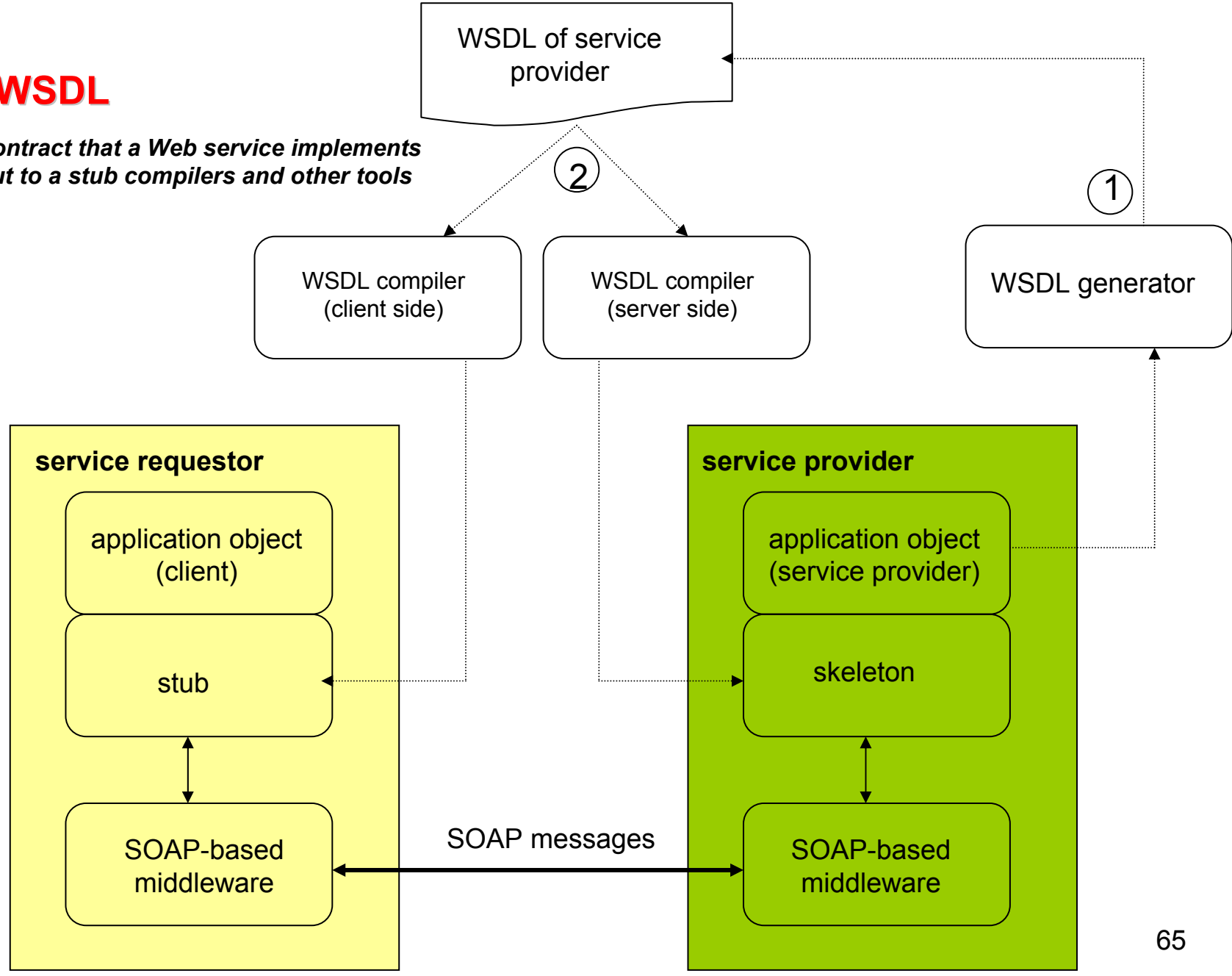
Structure of a WSDL interface




Address http://www.xmethods.net/sd/ibm/TemperatureService.wsdl
Go Links
Google Search Web Search Groups PageRank Page Info Up Highlight Norton AntiVirus
<?xml version="1.0" ?>
- <definitions name="TemperatureService"
targetNamespace="http://www.xmethods.net/sd/TemperatureService.wsdl"
xmlns:tns="http://www.xmethods.net/sd/TemperatureService.wsdl"
xmlns:xsd="http://www.w3.org/1999/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns="http://schemas.xmlsoap.org/wsdl/">
- <message name="getTempRequest">
<part name="zipcode" type="xsd:string" />
</message>
- <message name="getTempResponse">
<part name="return" type="xsd:float" />
</message>
- <portType name="TemperaturePortType">
- <operation name="getTemp">
<input message="getTempRequest" name="getTemp" />
<output message="getTempResponse" name="getTempResponse" />
</operation>
</portType>
- <binding name="TemperatureBinding" type="TemperaturePortType">
<soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
- <operation name="getTemp">
<soap:operation soapAction="" />
- <input>
<soap:body use="encoded" namespace="urn:xmethods-Temperature"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
- <output>
<soap:body use="encoded" namespace="urn:xmethods-Temperature"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</operation>
</binding>
- <service name="TemperatureService">
<documentation>Returns current temperature in a given U.S. zipcode</documentation>
- <port name="TemperaturePort" binding="TemperatureBinding">
<soap:address location="http://services.xmethods.net:80/soap/servlet/rpcrouter" />
</port>
</service>
</definitions>

sing WSDL

1. ***A contract that a Web service implements***
2. ***Input to a stub compilers and other tools***



UDDI: Universal Description Discovery and Integration

Specification of a framework for describing and descovering Web services

UDDI specification originated from:

1. Ariba and IBM collaborations on B2B
2. IBM and Microsoft collaborations on XML and SOAP
3. Microsoft and Ariba collaborations on BizTalk and cXML

First specification appeared in 2000

UDDI defines data structures and APIs for publishing service descriptions in the registry and for querying the registry to look for published descriptions

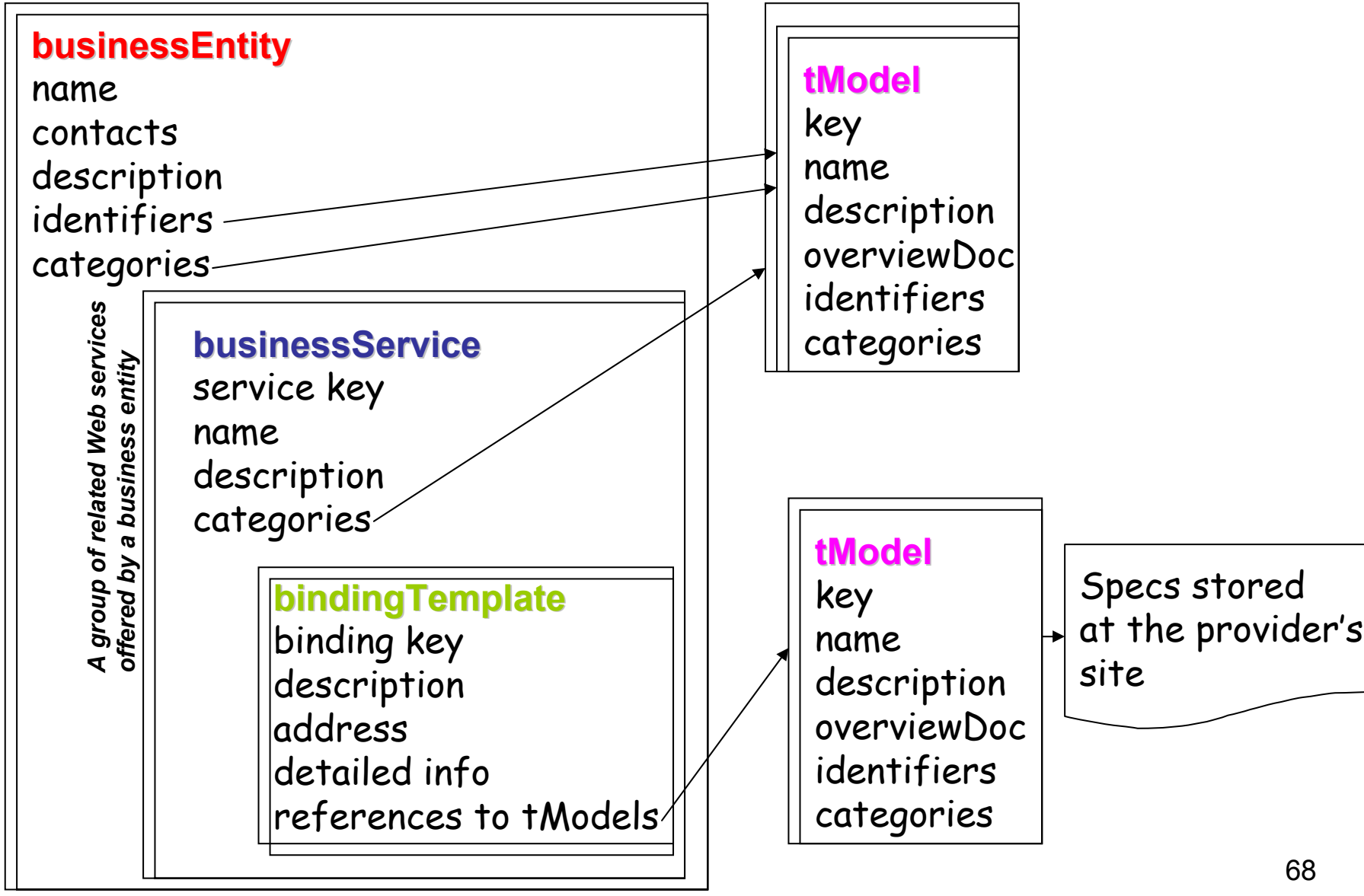
Information in a UDDI registry

White Pages : organizations and contact information (e.g., telephone, email address) and the services the organizations provide.

Yellow Pages : classifications of both companies and Web services according to taxonomies.

Green Pages : how a given service can be invoked (pointers to service description documents, typically stored outside the registry)

UDDI data structures



UDDI tModel: example

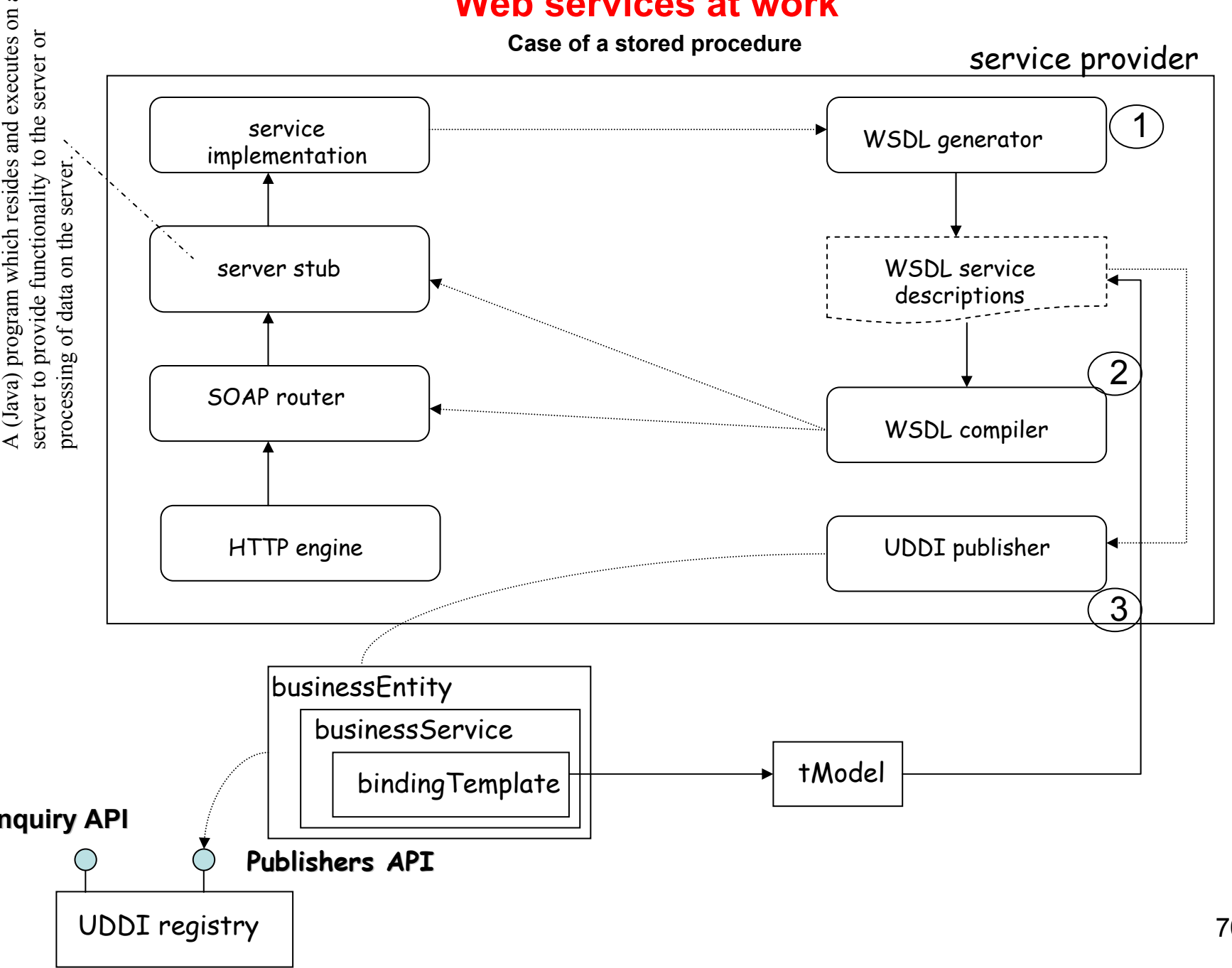
```
<tModel tModelKey=''uddi:uddi.org:v3_publication''>
        <name>uddi-org:publication_v3</name>
        <description>UDDI Publication API V3.0</description>
<overviewDoc>
        <overviewURL useType=''wsdlInterface''>
http://uddi.org/uddi_api_v3_binding.wsdl#UDDI_Publication_SoapBinding
        </overviewURL>
</overviewDoc>
<overviewDoc>
        <overviewURL useType=''text''>
        http://uddi.org/pubs/uddi_v3.htm#PubV3
        </overviewURL>
</overviewDoc>
<categoryBag>
        <keyedReference keyName=''uddi-org:types:wsdl''
                keyValue=''wsdlSpec''
                tModelKey=''uddi:uddi.org:categorization:types''/>
        <keyedReference keyName=''uddi-org:types:soap'
                keyValue=''soapSpec''
                tModelKey=''uddi:uddi.org:categorization:types''/>
        <keyedReference keyName=''uddi-org:types:xml''
                keyValue=''xmlSpec''
                tModelKey=''uddi:uddi.org:categorization:types''/>
        <keyedReference keyName=''uddi-org:types:specification''
                keyValue=''specification''
                tModelKey=''uddi:uddi.org:categorization:types''/>
</categoryBag>
</tModel>
```

overviewDoc
(refer to WSDL specs and to API specs)

Classification information (specifies that this tModel is about XML, WSDL and SOAP specs

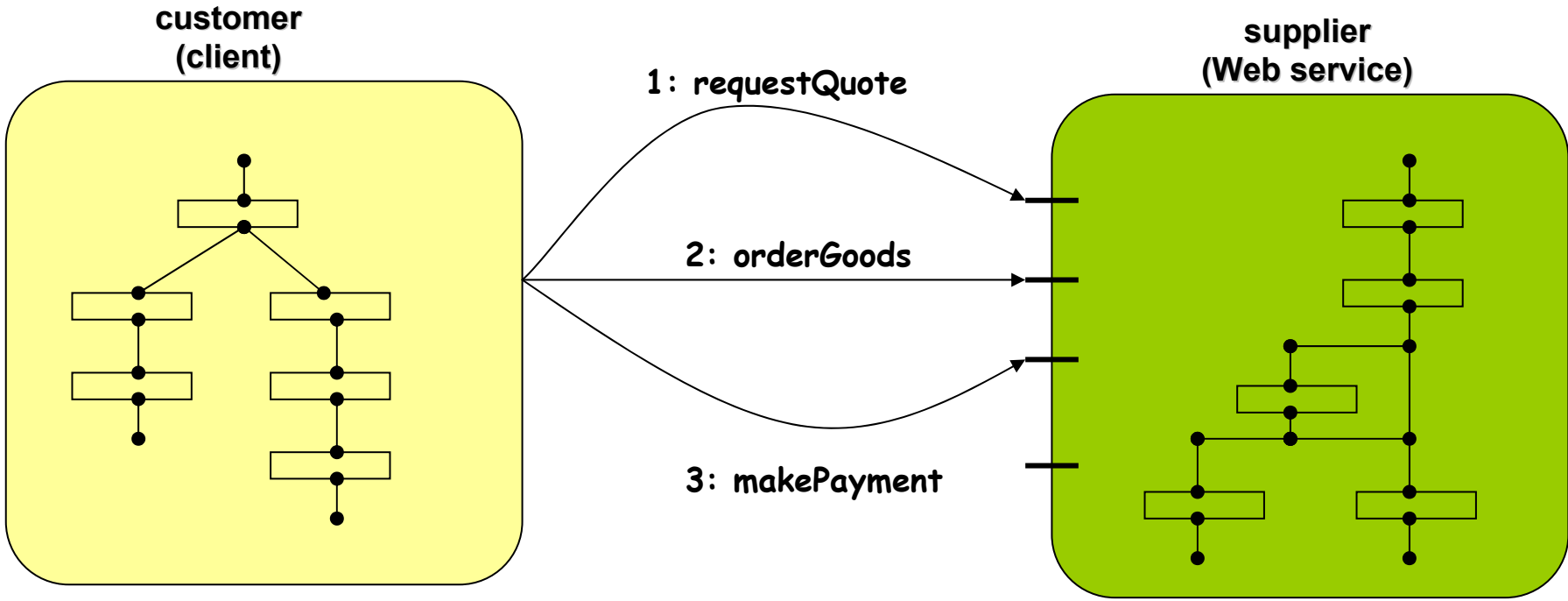
Web services at work

Case of a stored procedure



Service Coordination Protocols

In real applications, interactions are typically more complex than single, independent invocations. Using a particular service typically involves performing sequences of operations in a particular order.



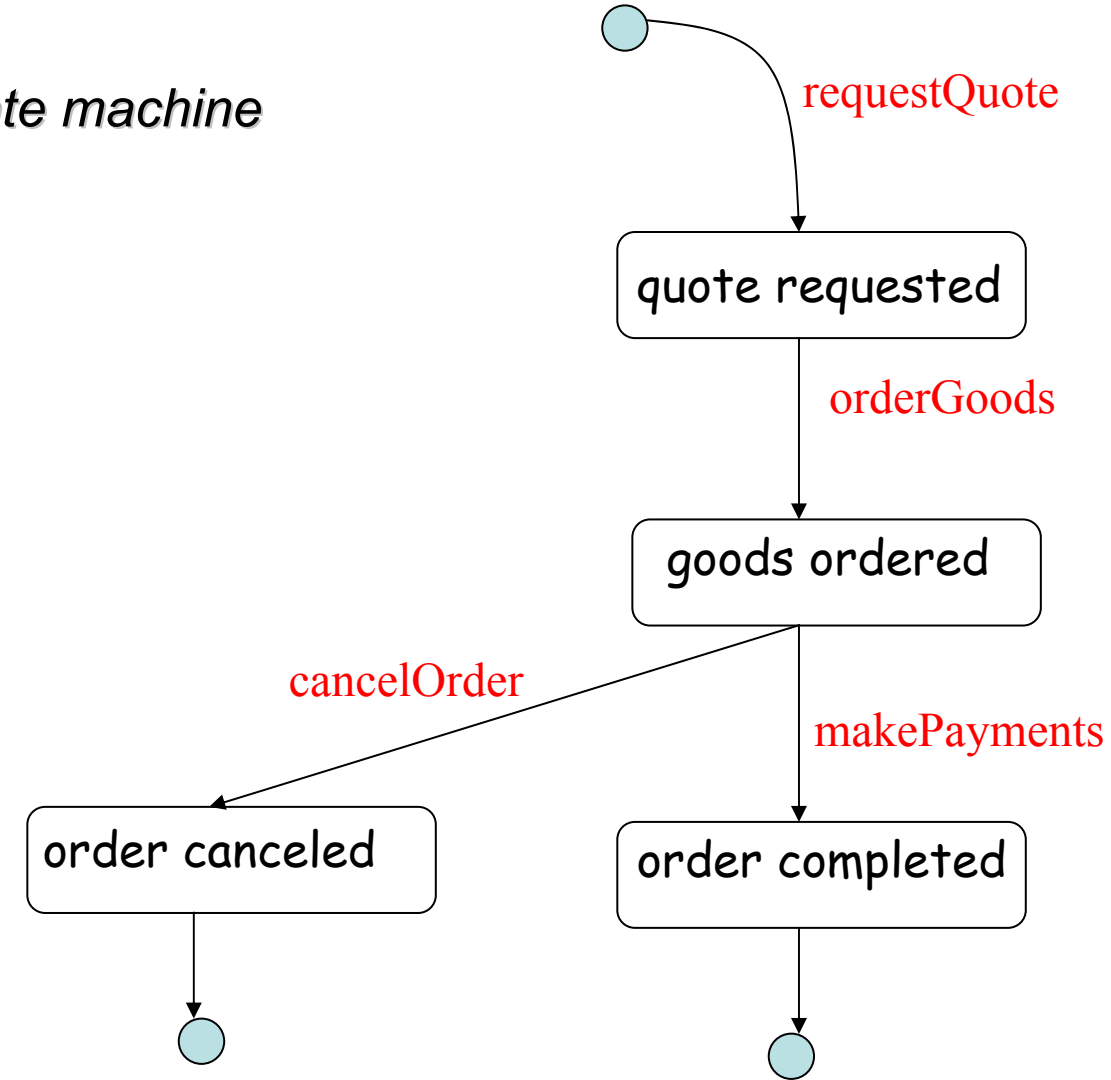
Complex internal logic
Context information
(**conventional programming language** or
service composition)

Modeling Conversation between a Client and a Web Service

Conversation: sequences of operations (i.e., message exchanges) that could occur between a client and service as part of the invocation of a Web service.

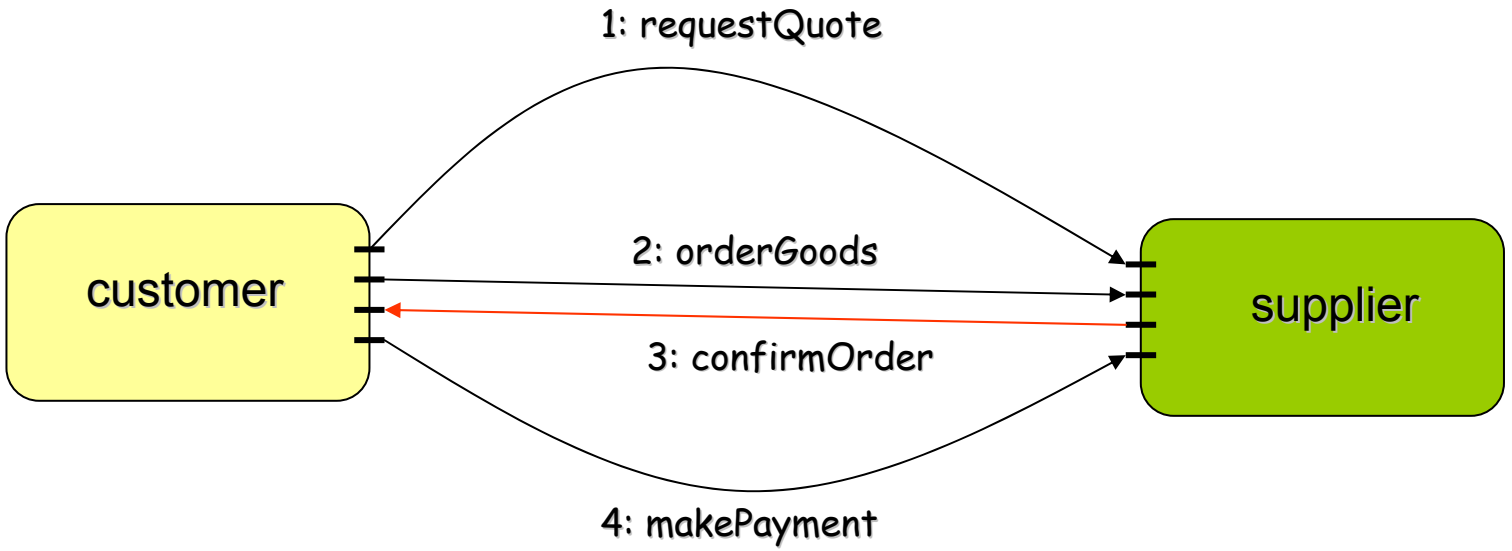
Coordination protocol: the specification of the set of correct and accepted conversations.

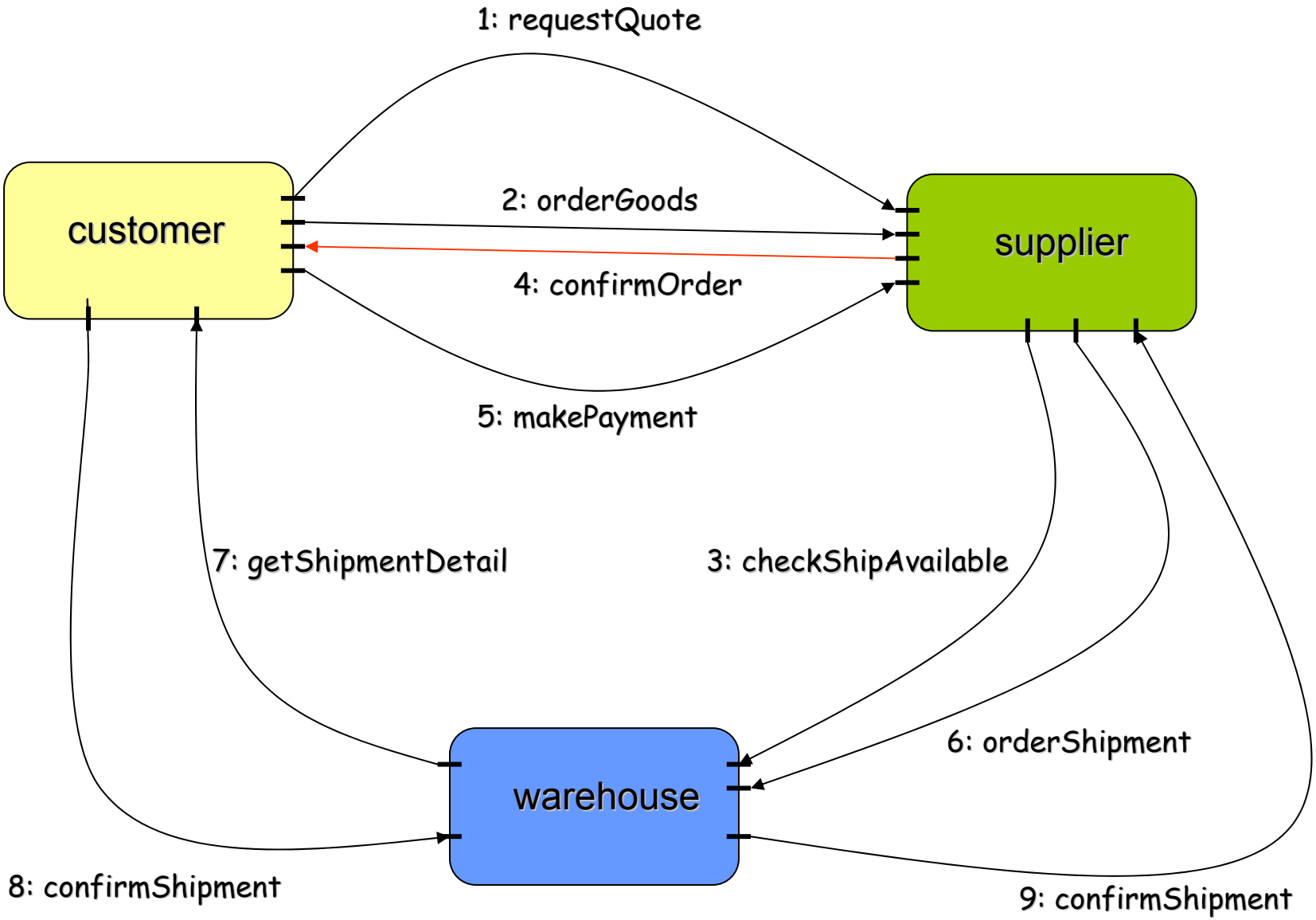
State machine



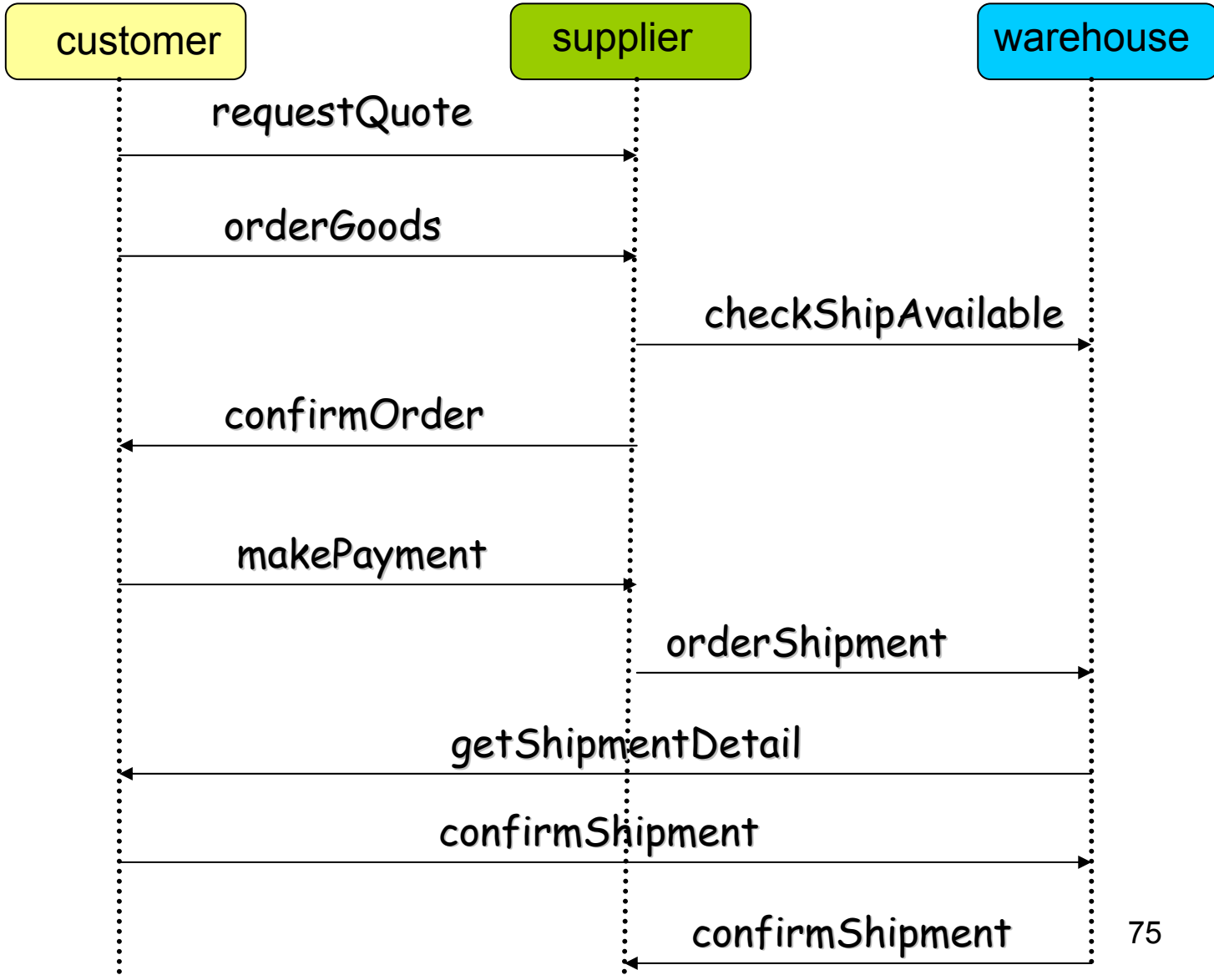
WSCL

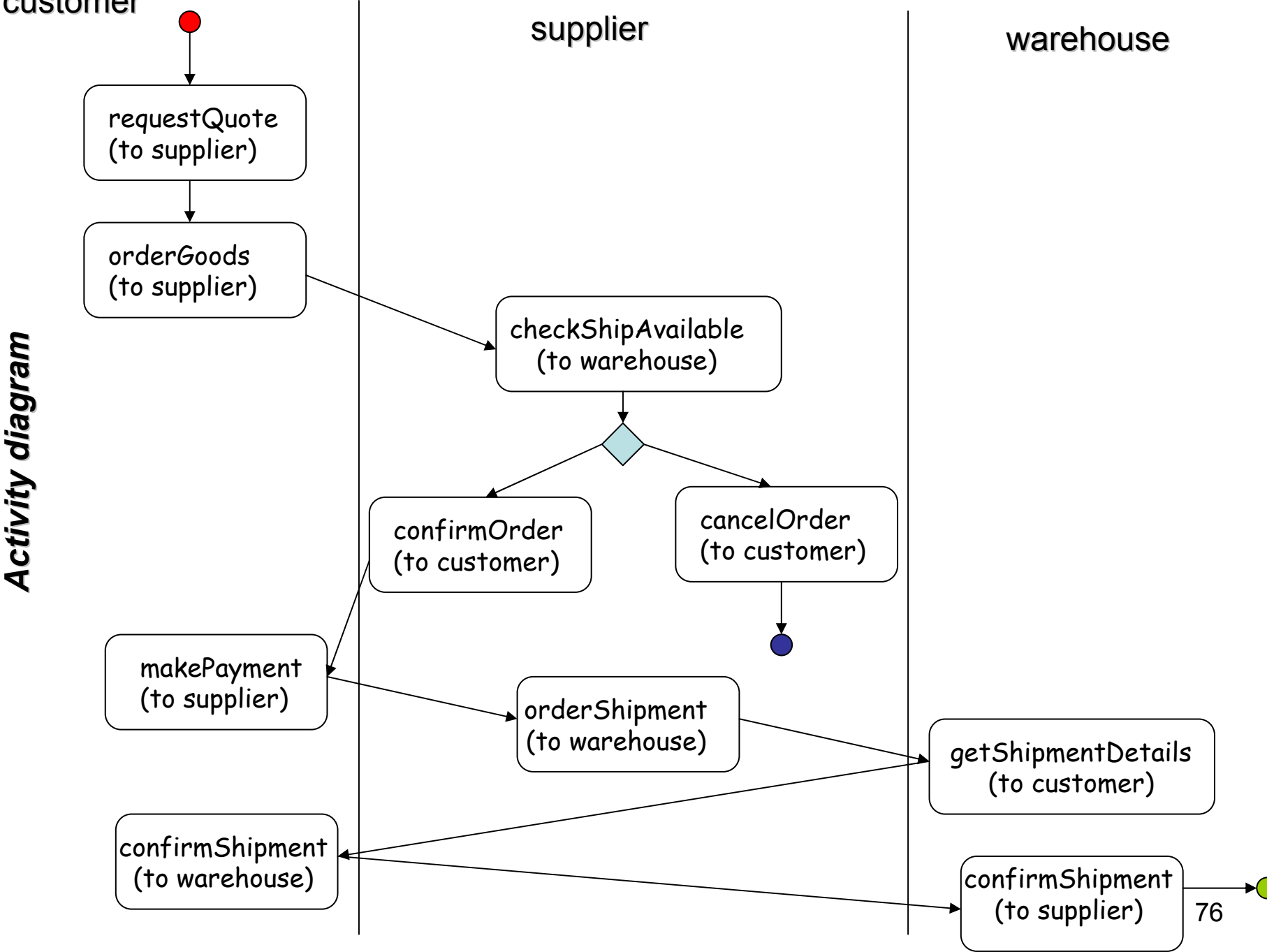
Reason: asynchronous nature of Web services




1: requestQuote
customer
2: orderGoods
supplier
4: confirmOrder
5: makePayment
7: getShipmentDetail
3: checkShipAvailable
6: orderShipment
warehouse
8: confirmShipment
9: confirmShipment

Sequence diagram

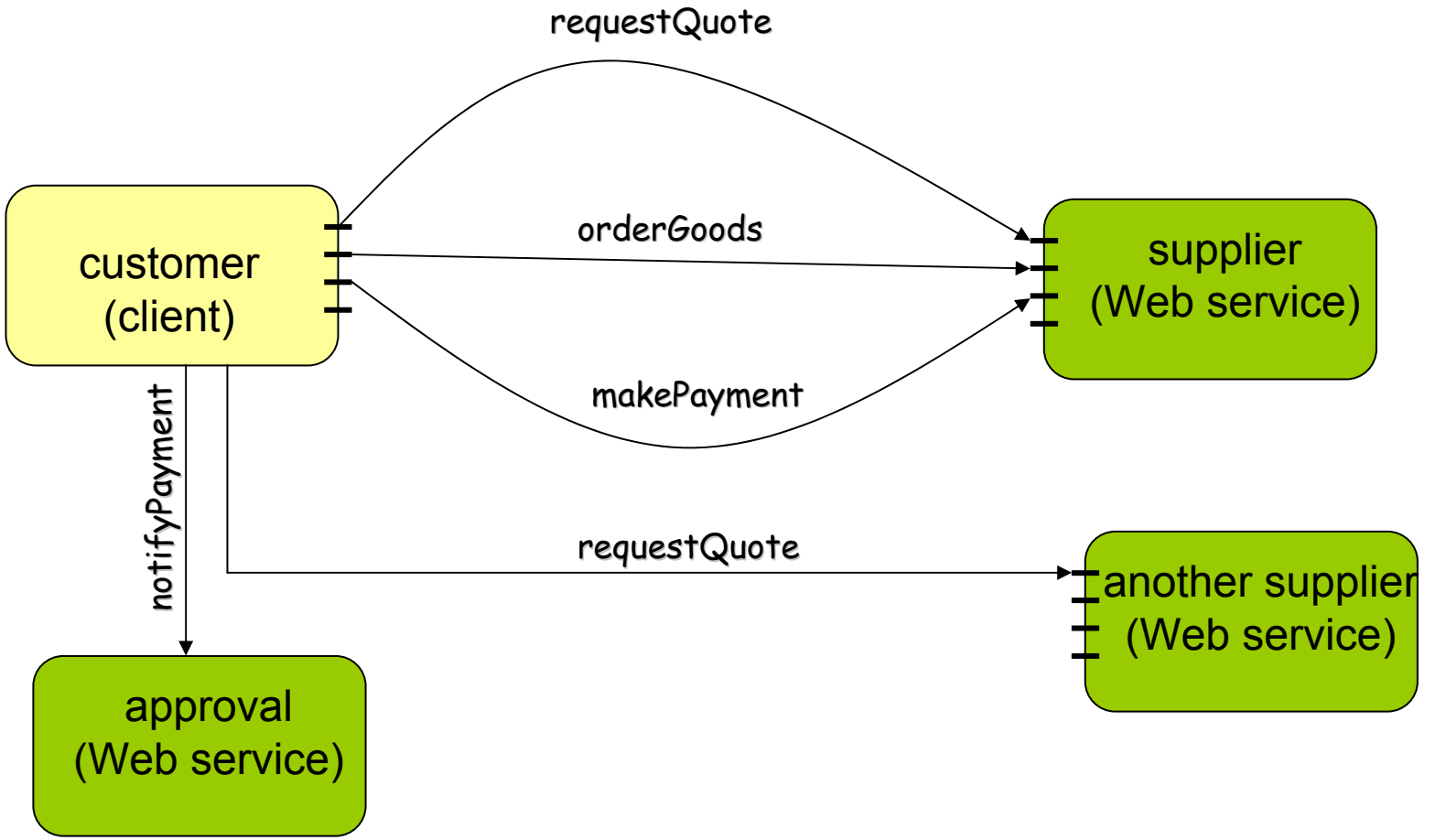


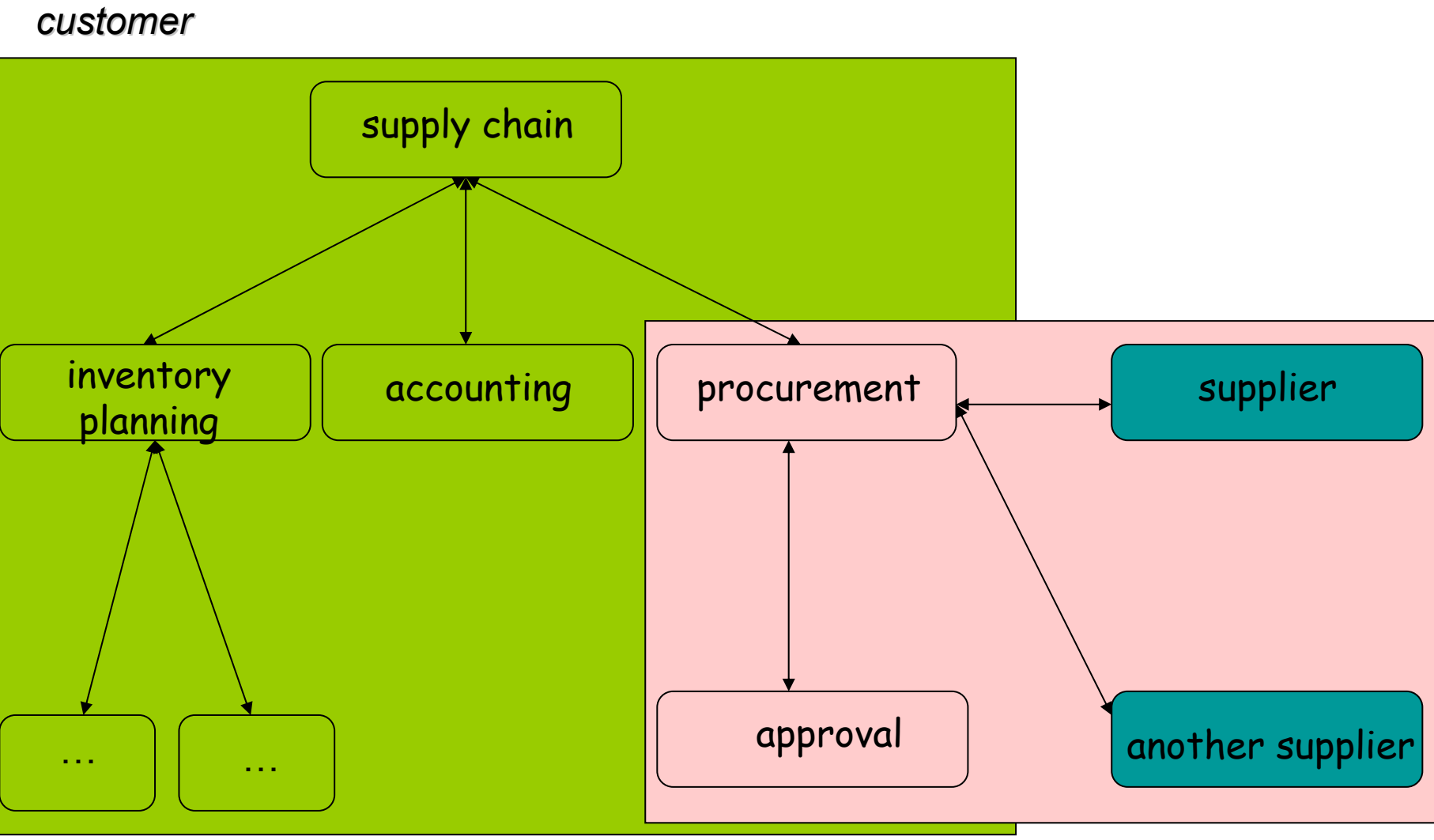


Activity diagram

Service Composition

Composition as a way to master complexity




customer
supply chain
inventory planning
accounting
procurement
supplier
approval
another supplier
…
…

Semantic Web

Semantic Web

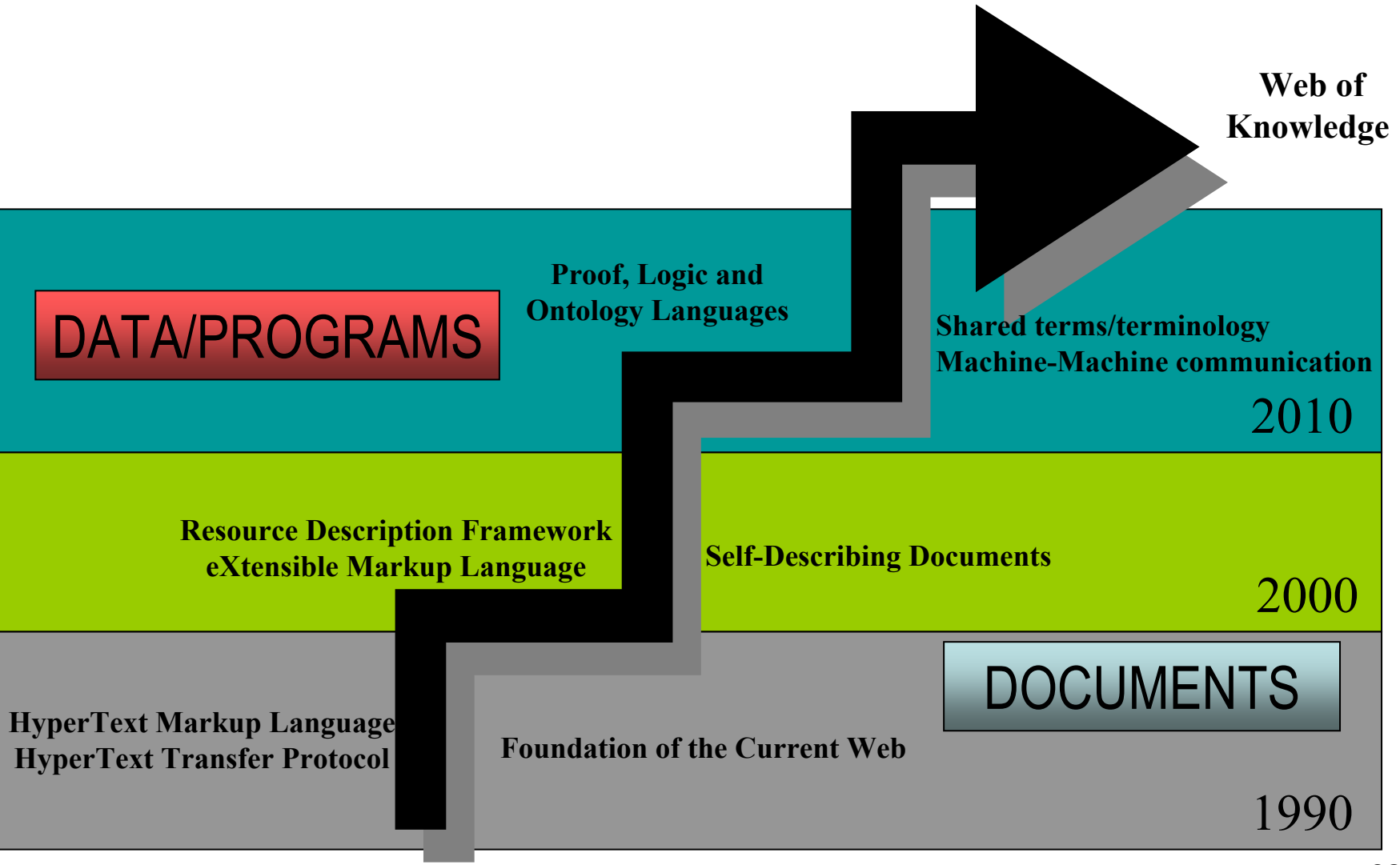
- It is the sucess of the web that creates serious needs for its improvement.
- The web uses the computer as a device for rendering information for the human reader but neither for information processing nor computing.

The semantic web is aiming on bringing back the computer as an information processing device.

Semantic Web

- The semantic web is based on **machine-processable** semantics of data.
- It will significantly change our information access based on a higher level of service provided by computers.
- It is based on new web languages such as XML, RDF, and OWL, and tools that make use of these languages.
- Applications are in areas such as Knowledge Management (eWork, eLearning, eGoverment, ...), Enterprise Application Integration, and eCommerce.

The Evolving Web



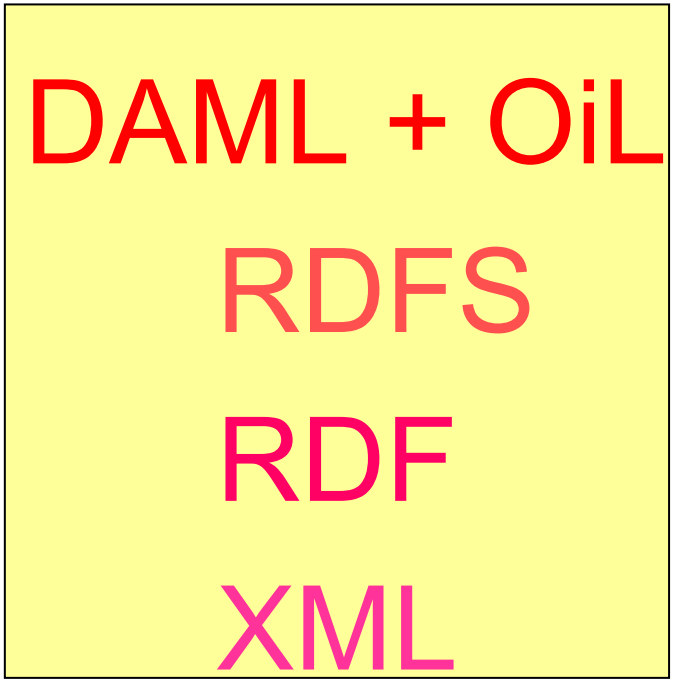
Berners-Lee, Hendler; *Nature*, 2001

Semantic Web

Main achievements:

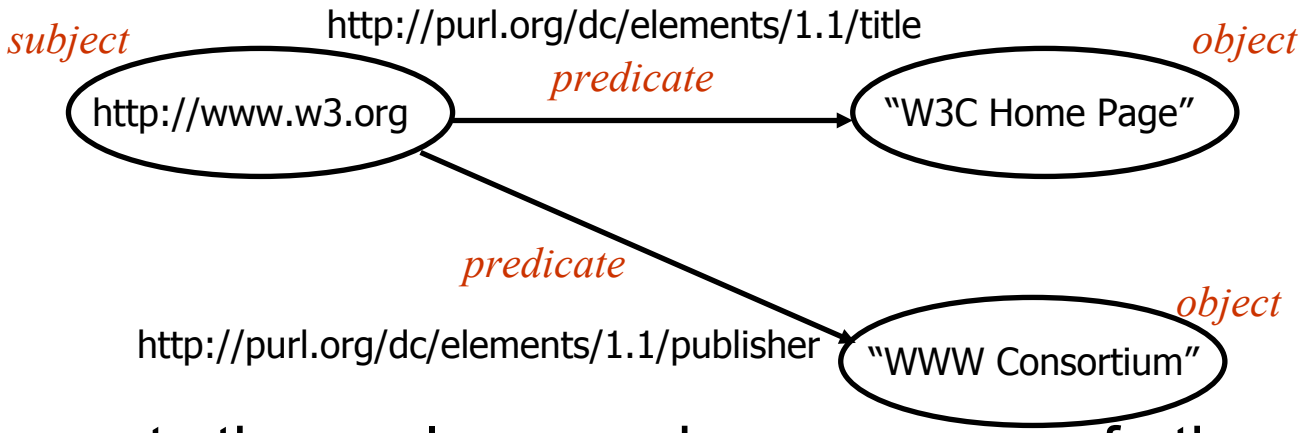
- A ontology language proposal called OWL.
- Several case studies for intranet applications and a methodology.
- A three-layered software architecture for making the semantic web a reality.
- A large number of interwoven web services that implement this vision.

Hierarchy of Languages



RDF – Resource Description Framework

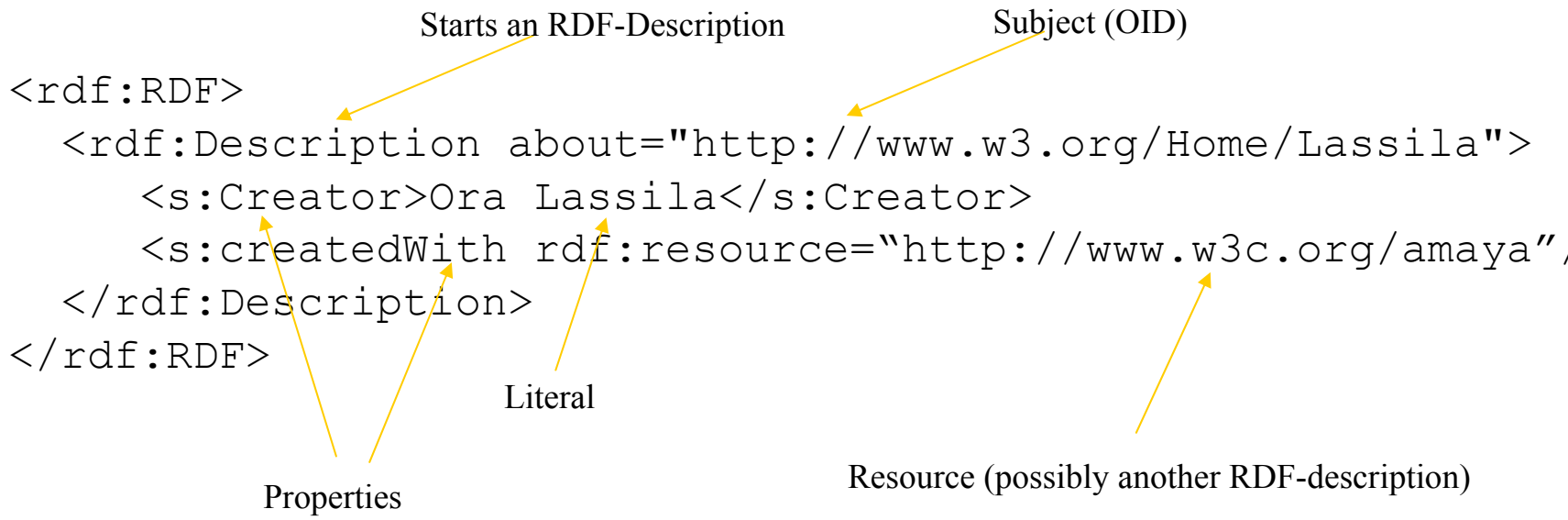
- Resources are related to each other by properties to form *subject/predicate/object* statements (triples).
- The triples can be used to construct a graph:



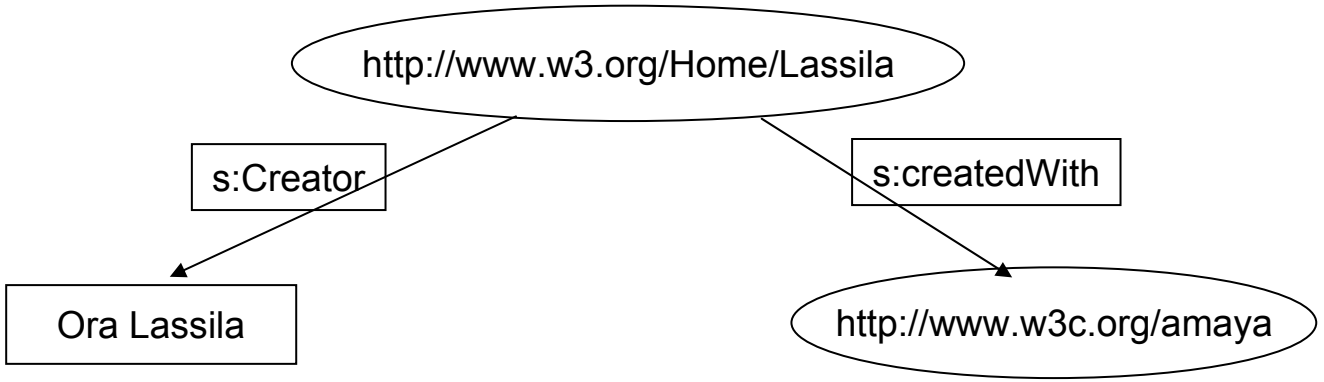
- Statements themselves can be resources of other statements (i.e. reified statements)

RDF Syntax

- Data model does not enforce particular syntax
- Specification suggests many different syntaxes based on XML
- General form:



Resulting Graph




```
<rdf:RDF>
  <rdf:Description about="http://www.w3.org/Home/Lassila">
     <s:Creator>Ora Lassila</s:Creator>
     <s:createdWith rdf:resource="http://www.w3c.org/amaya"/>
  </rdf:Description>
</rdf:RDF>
```

RDF schema

- Different from XML DTD: syntax vs. semantics
- Defines *Class, Property, subClassOf, subPropertyOf*, domain, range, and some others
- http://www.w3.org/TR/rdf-schema/
  http://www.w3.org/TR/REC-rdf-syntax/

Why RDF Is Not Enough

- Only range/domain constraints on properties (need others)
- No properties of properties (unique, transitive, inverse, etc.)
- No equivalence, disjointness, etc.
- No necessary and sufficient conditions (for class membership)
- No defined semantics

From RDF to DAML+OIL

- DAML+OIL: DARPA Agent Markup Language
  - Current version unites early DAML language with OIL
- DAML+OIL extends RDF statements to provide a rich descriptive logic language
  - Provides restrictions and additional notations on properties
    - Cardinality restrictions
    - Notations include *inverseOf*, *Transitivity*, etc
  - Provides additional properties for class definitions
    - *Disjoint-with*, *complement-Of*, *intersectionOf*, etc
  - Provides universal & existential quantification through class restriction

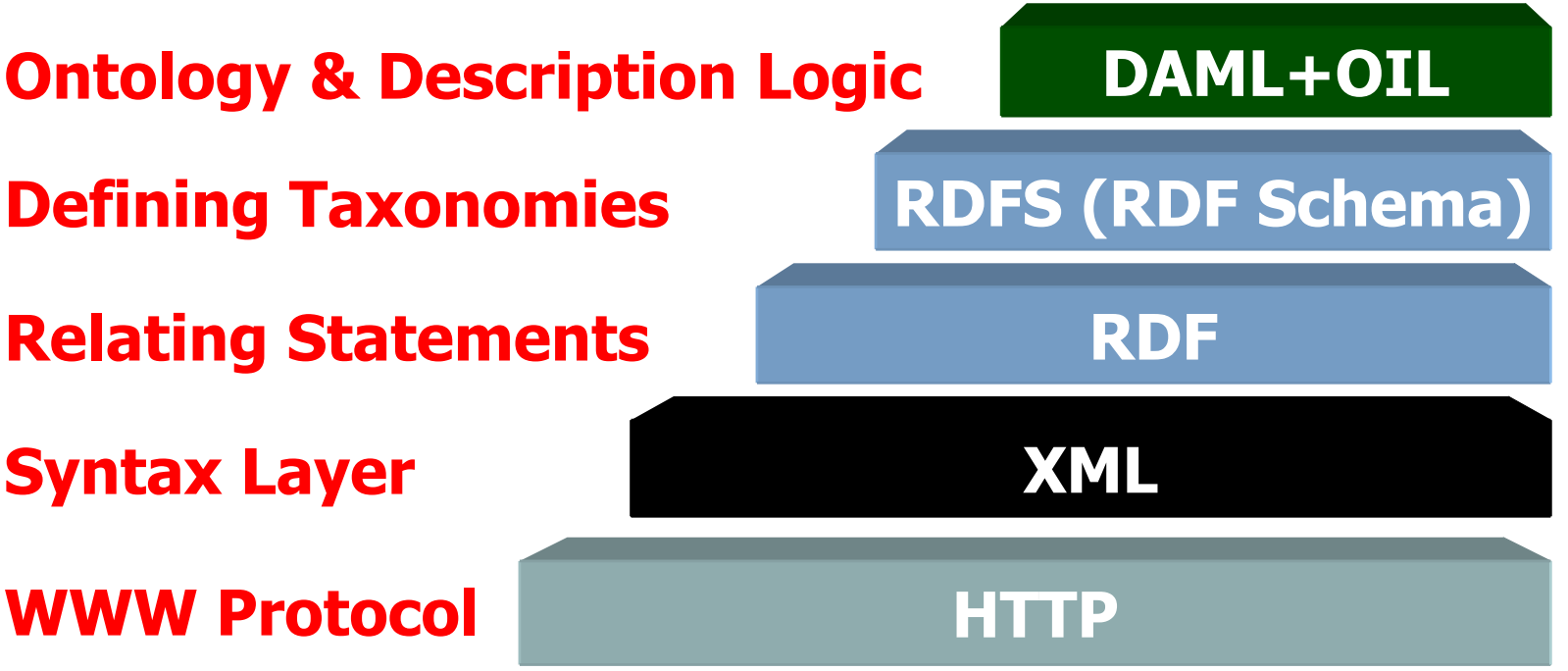
http://www.daml.org/language

DAML+Oil example

```
Define a "product number"'s domain and range..
<daml:DatatypeProperty rdf:ID="productNumber">
<rdfs:label>Product Number</rdfs:label>
<rdfs:domain rdf:resource="#Product"/>
<rdfs:range rdf:resource=
 "http://www.w3.org/2000/10/XMLSchema#nonNegativeInteger"/>
</daml:DatatypeProperty>
”Availability" is a sort of enumerated type..
<daml:Class ID="Availability">
<daml:oneOf parseType="daml:collection">
<daml:Thing rdf:ID="InStock">
<rdfs:label>In stock</rdfs:label>     </daml:Thing>
<daml:Thing rdf:ID="BackOrdered">
<rdfs:label>Back ordered</rdfs:label>     </daml:Thing>
<daml:Thing rdf:ID="SpecialOrder">
<rdfs:label>Special order</rdfs:label>     </daml:Thing>
</daml:oneOf>
```

Semantic Web Layer Cake

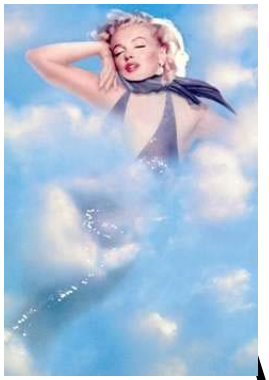
- Semantic Web layer cake proposed by Tim Berners-Lee
- Build upon successive W3C standards
- Add meaning through semantics to the existing WWW



Web Services

- Web Services will transform the web from a collection of information into a distributed device of computation.
- Web services should transform eCommerce from a nice application into a mass phenomena.
- Bringing E-commerce to its full potential requires a *Peer-to-Peer (P2P) approach*. Anybody must be able to trade and negotiate with everybody else.
- However, such an open and flexible E-commerce has to deal with many obstacles before it becomes reality!
- The issue is ***scalability*** and ***economy in price.***

Web Services



Def 2. New concept for eWork and eCommerce

Def 1. Software Architecture

Def 3.
New programming technology

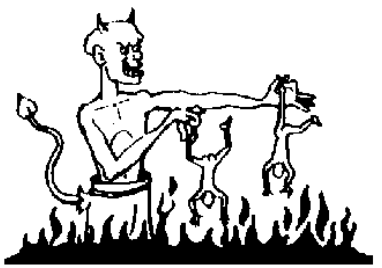


Figure Taken From Dieter Fensel Talk

Web Services

Def 1. Web Services as a Software Architecture

"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."

IBM web service tutorial

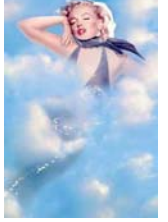
Web Services

➔ Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways.

➔ The key to Web Services is on-the-fly software creation through the use of loosely coupled, reusable software components.

➔ Software can be delivered and paid for as fluid streams of services as opposed to packaged products.

Web Services

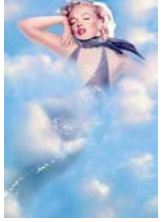


Def 2. Web Services as a new Concept for eWork and eCommerce

Web Services are Services accessible via the web

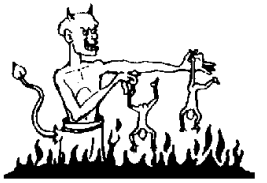
Dieter Fensel`s definition

Web Services



- Business services can be completely decentralized and distributed over the Internet and accessed by a *wide variety of communications devices*.
- The internet will become a global common platform where organizations and individuals communicate among each other to carry out various commercial activities and to provide value-added services.
- The dynamic enterprise and dynamic value chains become achievable and may be even mandatory.

Web Services



Def 3. Web Services as a programming technology

Web Services are Remote Procedure Calls (RPC) over HTTP

current state of the art

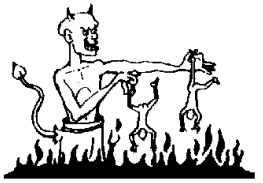
Web Services

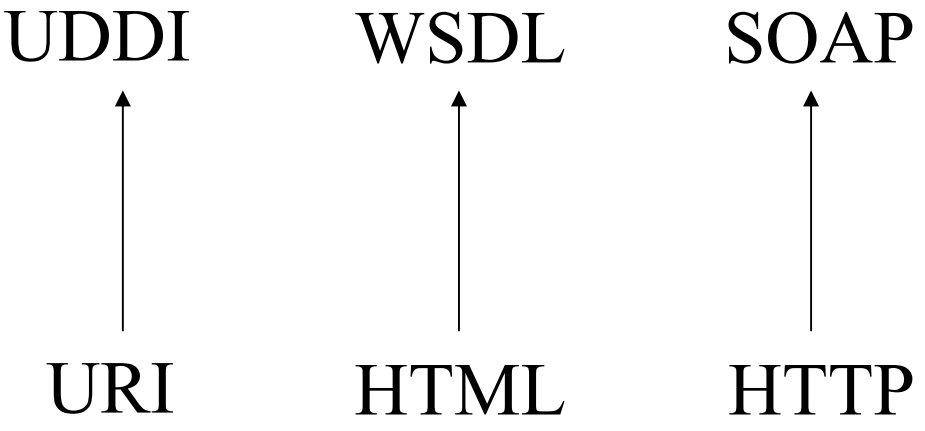
The web is organized around URIs, HTML, and HTTP.

- URIs provide defined ids to refer to elements on the web,
- HTML provides a standardized way to describe document structures (allowing browsers to render information for the human reader), and
- HTTP defines a protocol to retrieve information from the web.

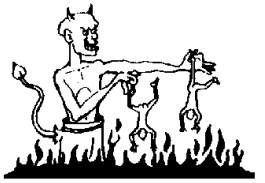
==> Not surprisingly, web services require a similar infrastructure around UDDI, WSDL, and SOAP.

Web Services



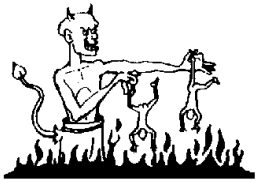


Web Services



- **UDDI** provides a mechanism for clients to find web services. A UDDI registry is similar to a CORBA trader, or it can be thought of as a DNS service for business applications.
- **WSDL** defines services as collections of network endpoints or *ports*. A port is defined by associating a network address with a binding; a collection of ports define a service.
- **SOAP** is a message layout specification that defines a uniform way of passing XML-encoded data. It also defines a way to bind to HTTP as the underlying communication protocol. SOAP is basically a technology to allow for “RPC *over the web*”.

Web Services



- UDDI, WSDL, and SOAP are important steps into the direction of a web populated by services.
  - However, they only address part of the overall stack that needs to be available in order to achieve the above vision eventually.
  - There are many layer requires to achieve ***automatic*** *web service discovery, selection, mediation and composition* into complex services.

Web Services

- Many organizations had the insight that message definition and exchange are not sufficient to build an expressive web services infrastructure.

- In addition to UDDI, WSDL and SOAP, standards are proposed such as WSFL, XLANG, ebXML, BPSS, BPML, WSCL, and BPEL4WS.

Bringing web services to their full potential requires their combination with semantic web technology.

Semantic Web Services

Imagine a travelling service:

- Decompose into elementary services
- Describe elementary services by goals instead of hardwiring them.
- Keep the human programmer out of the loop to keep it economic, on demand, and scalable.

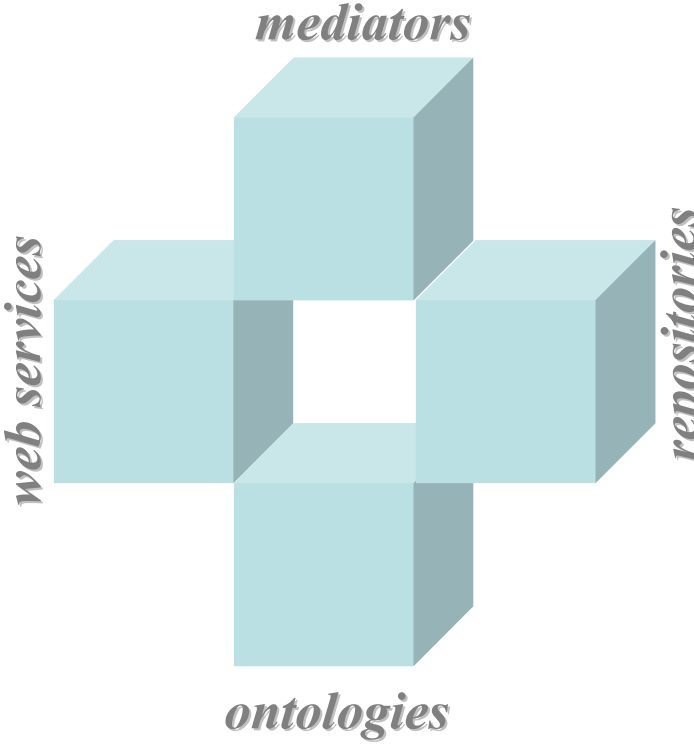
You cannot achieve this vision without semantic web technology that maintains **selection** and **combination** of heterogeneous web services during **runtime**.

Semantic Web Services

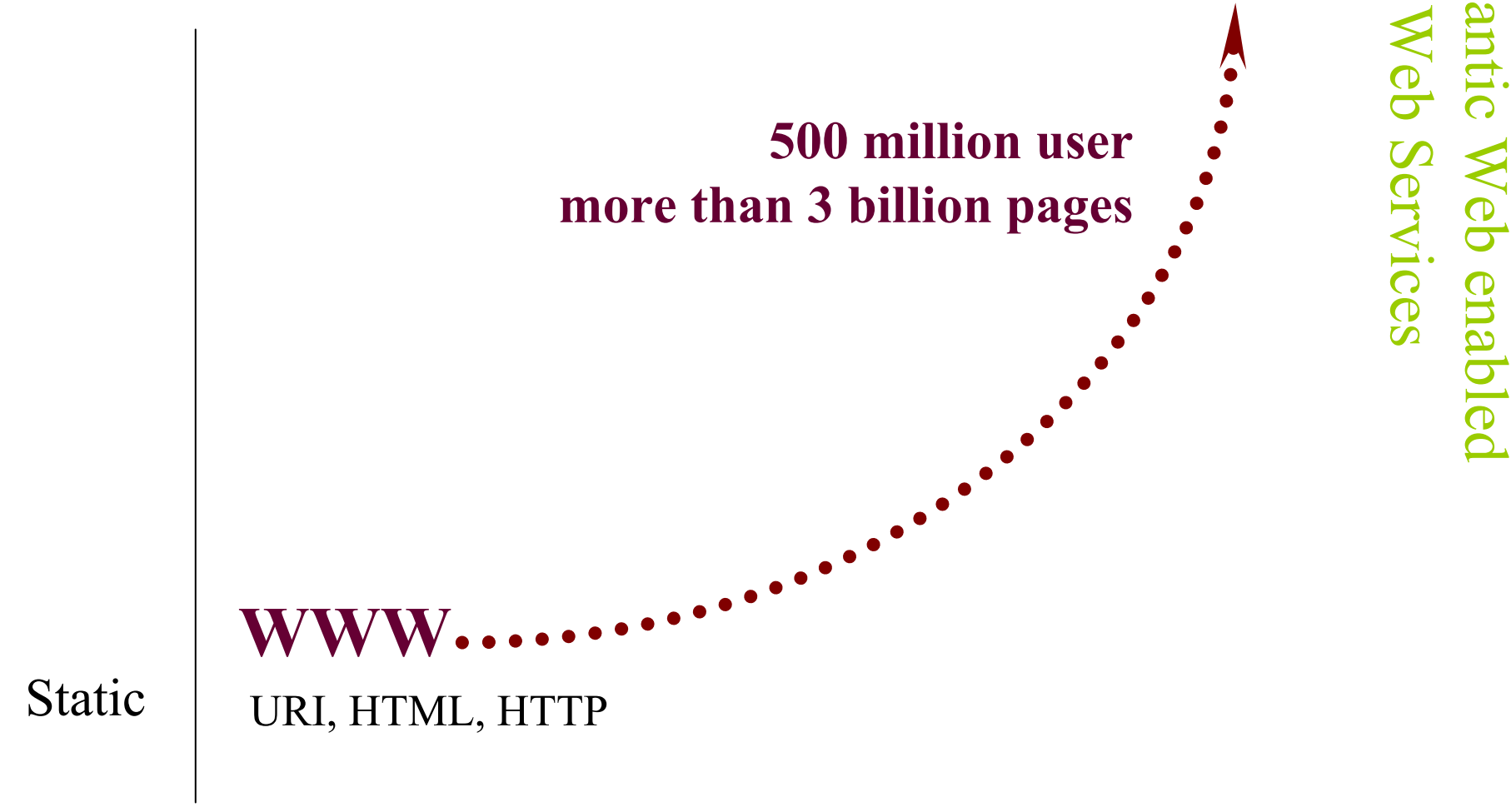
- Mechanized support is needed, for example in finding and comparing vendors and their offers. ***Machine processable semantics*** of information allows to mechanize these tasks.
- Mechanized support is needed in dealing with numerous and heterogeneous data formats. ***Ontology technology*** is required to define such standards better and to map between them.
- Mechanized support is needed in dealing with numerous and heterogeneous ***business logics***. Mediation is needed to compensate these differences, allowing partners to cooperate properly.

Semantic Web Services

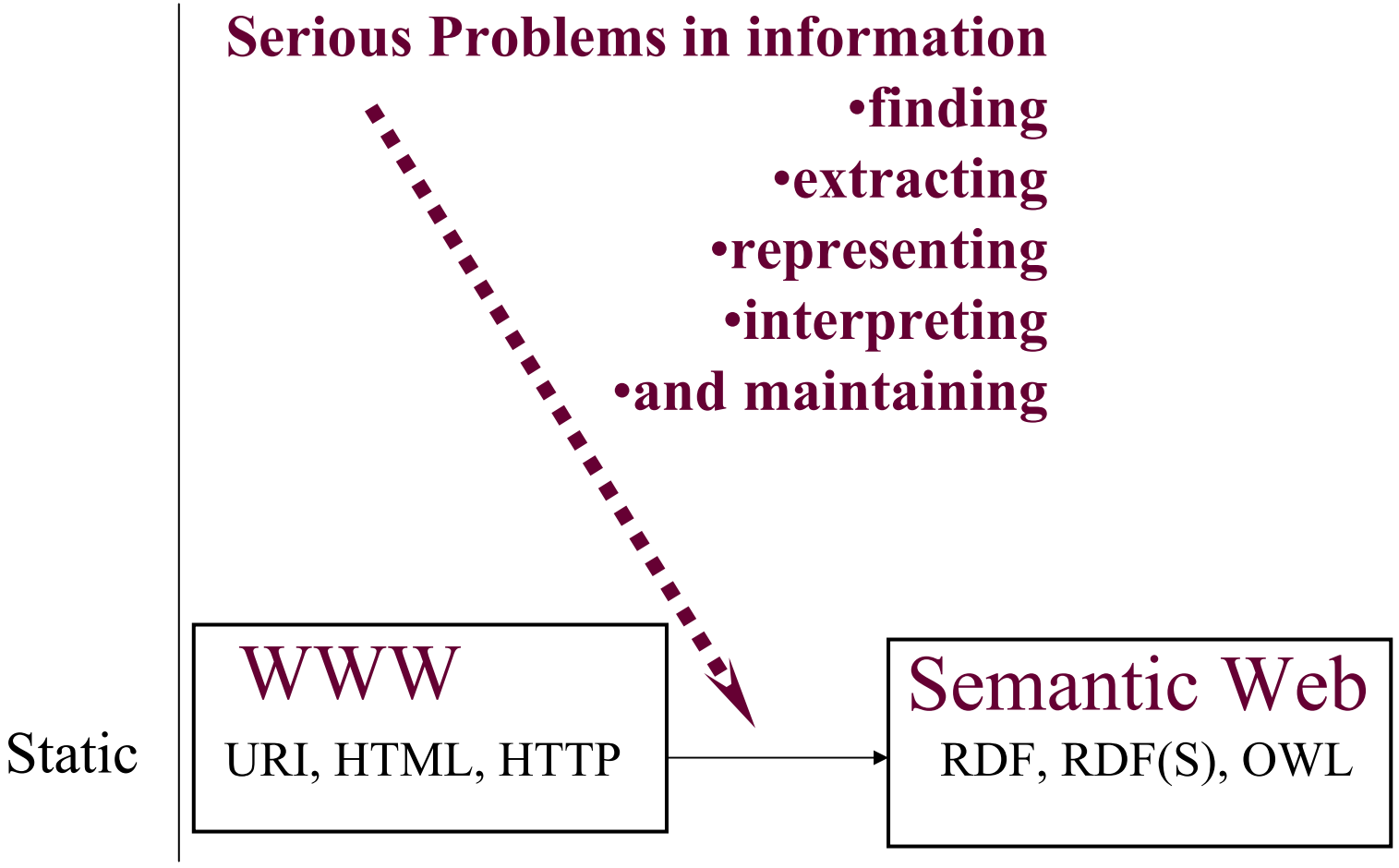
- The WSMF consists of four main different elements:
  - *ontologies* that provide the terminology used by other elements;
  - *goal repositories* that define the problems that should be solved by web services;
  - *web services* descriptions that define various aspects of a web service;
  - and *mediators* which bypass interoperability problems.



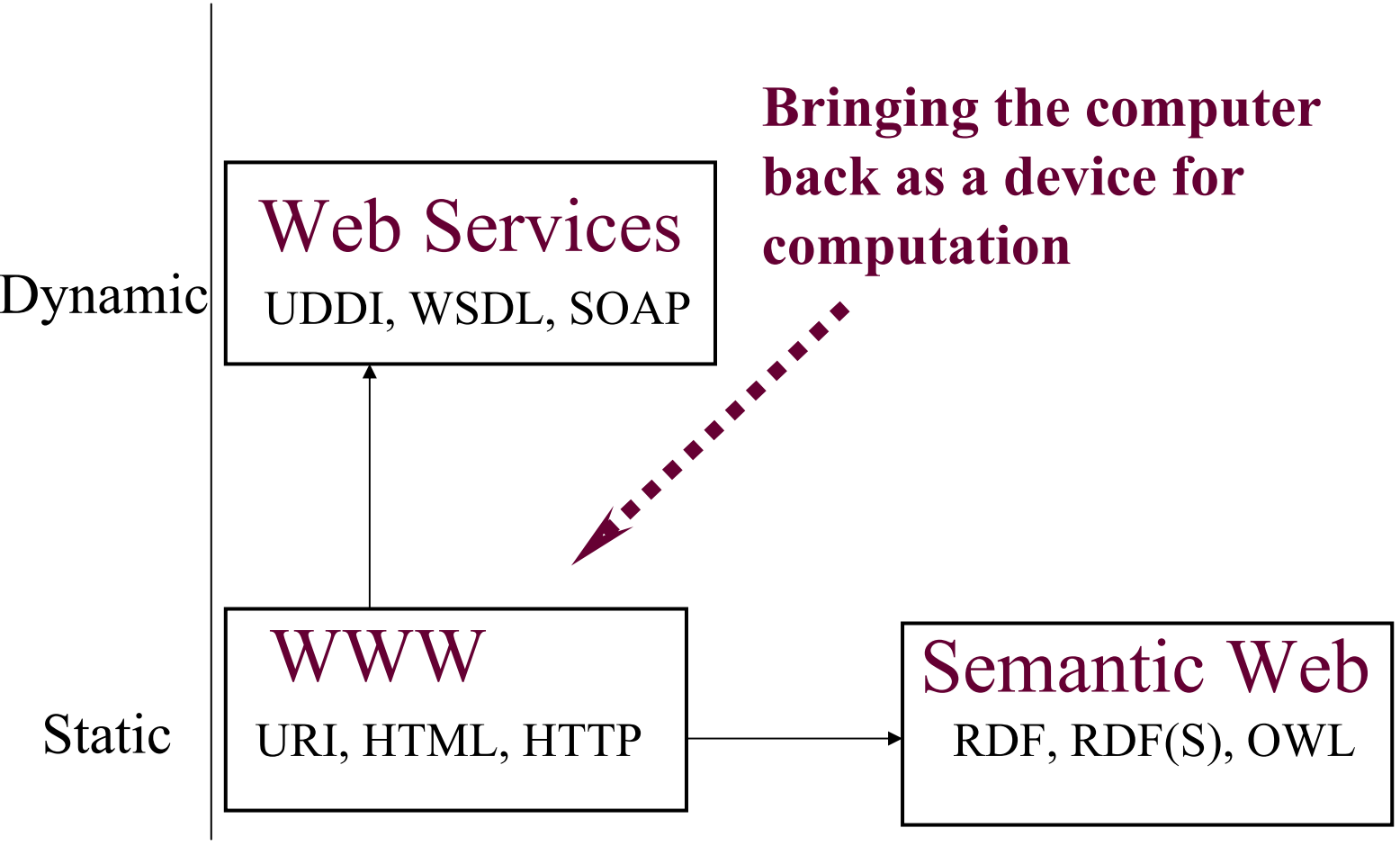
The General Vision



The General Vision

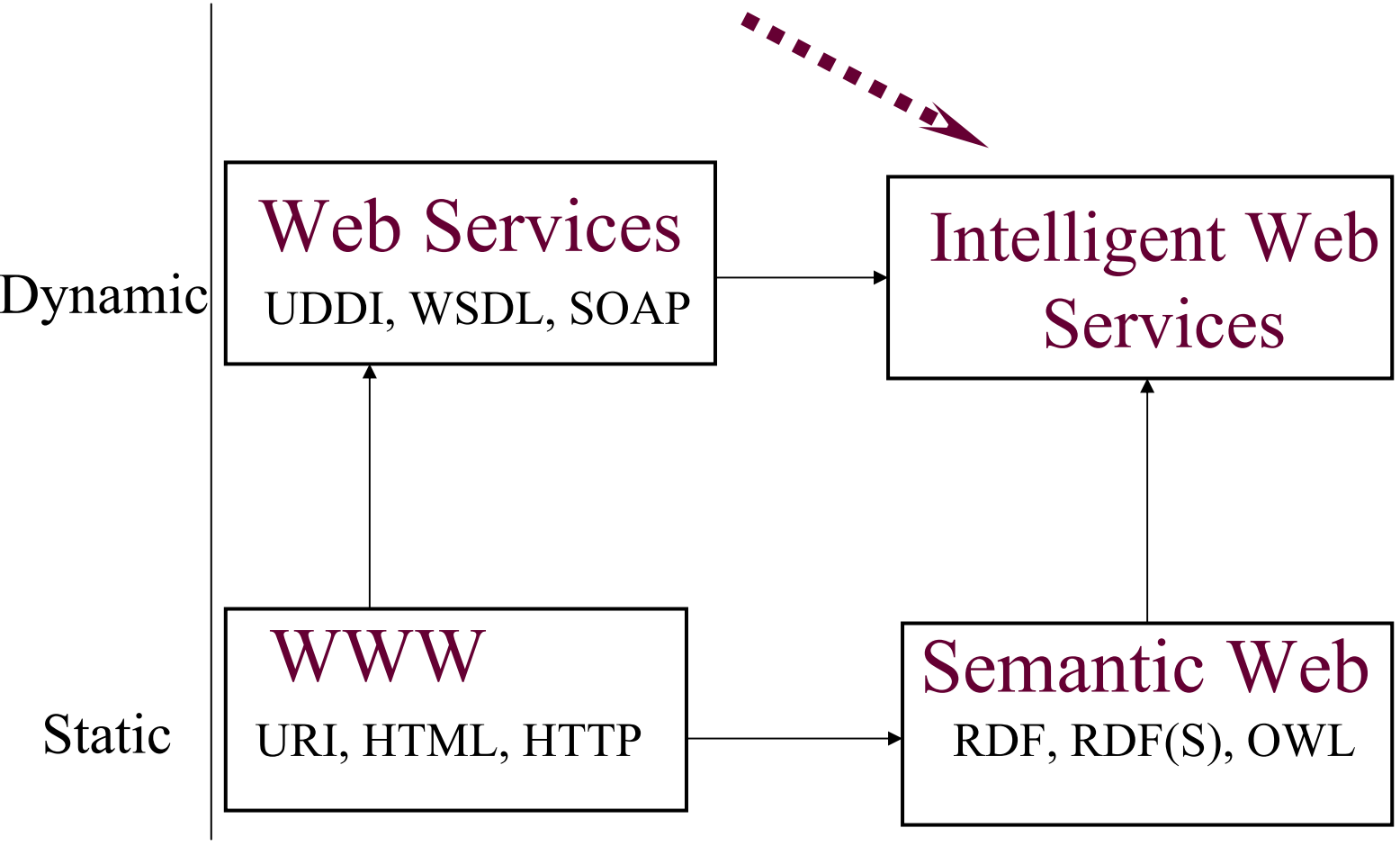


The General Vision

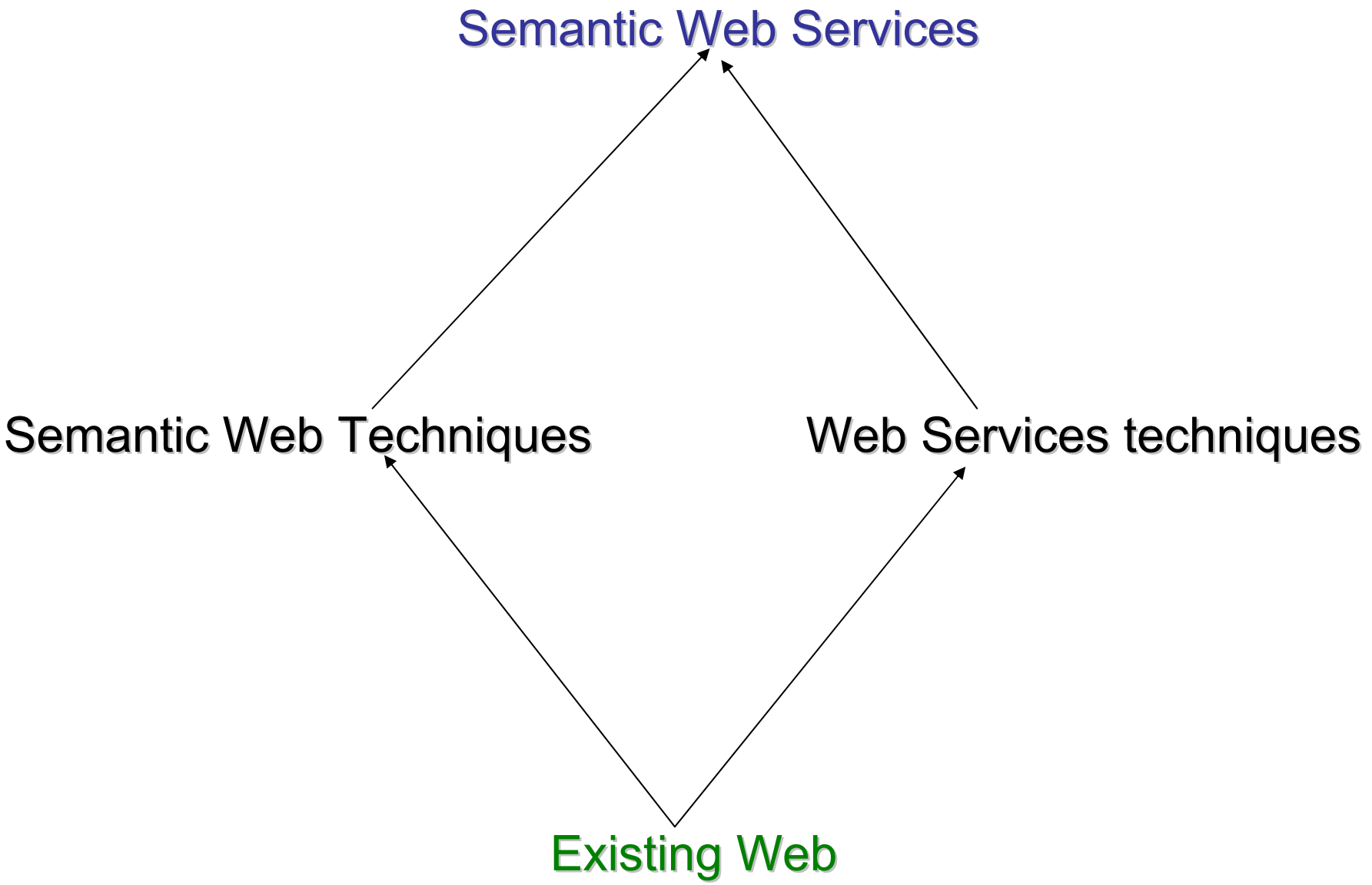


The General Vision

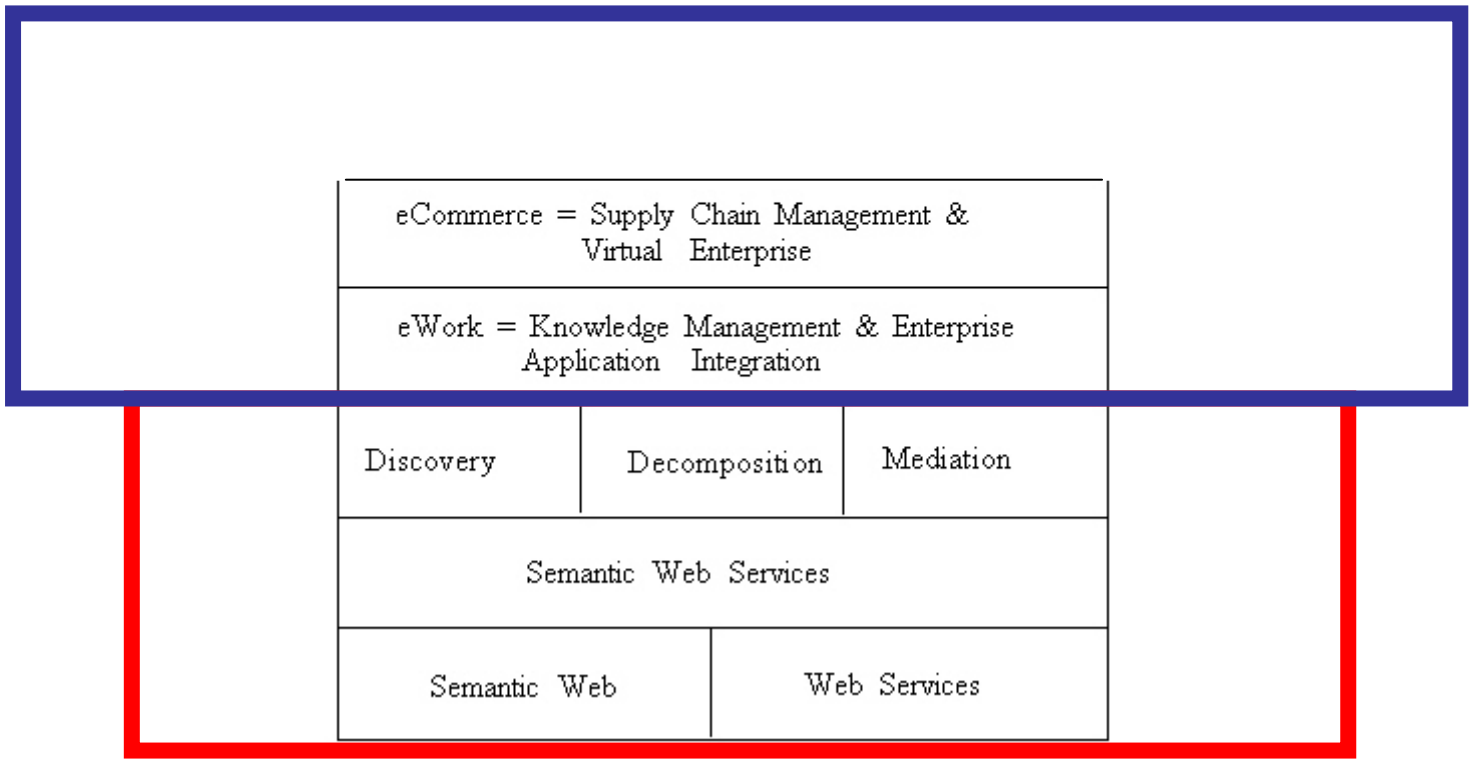
Bringing the web to its full potential

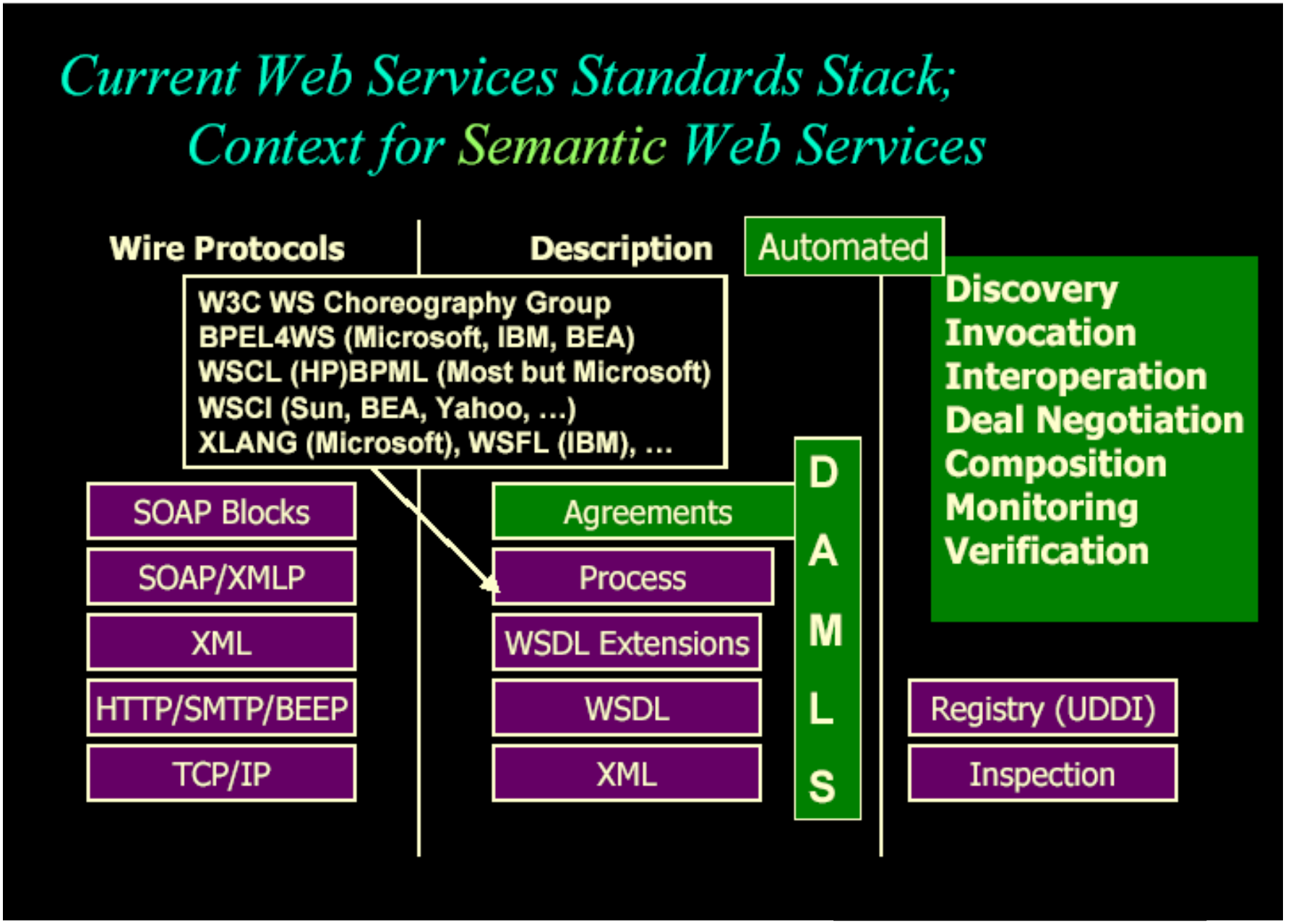


Semantic Web Services


Semantic Web Services
Semantic Web Techniques
Web Services techniques
Existing Web

Semantic Web enabled Web Services





Slide taken from Grosof's Talk

Semantic Web Services

- Convergence of Semantic Web and Web Services
- Consensus definition and conceptualization still forming
- Semantic (Web Services):
  - Knowledge-based service descriptions, deals
    - Discovery/search, invocation, negotiation, selection, composition, execution, monitoring, verification
  - Integrated knowledge
- (Semantic Web) Services: e.g., infrastructural
  - Knowledge/info/DB integration
  - Inferencing and translation

Slide taken from Grosof's Talk

SWS Tasks at higher layers of WS stack

Automation of:

- Web service discovery

  Find me a shipping service that will transport frozen vegetables from San Francisco to Tuktoyuktuk.

- Web service invocation

  Buy me "Harry Potter and the Philosopher's Stone" at www.amazon.com

- Web service deals, i.e., contracts, and their negotiation

  Propose a price with shipping details for used Dell laptops to Sue Smith.

- Web service selection, composition and interoperation

  Make the travel arrangements for my WWW11 conference.

Slide taken from Grosof's Talk

Describing & Discovering Agents & Services on the Semantic Web

- DAML – DARPA **Agent** Markup Language!
- Can be used as a tool to investigate and solve many of the agent-based semantic mismatch issues
  - i.e. Semantic mismatches in agent discovery, selection, negotiation, interoperation, & in the composition/planning of larger scale solutions
- DAML -S Coalition formed to explore DAML for Services

DAML-S

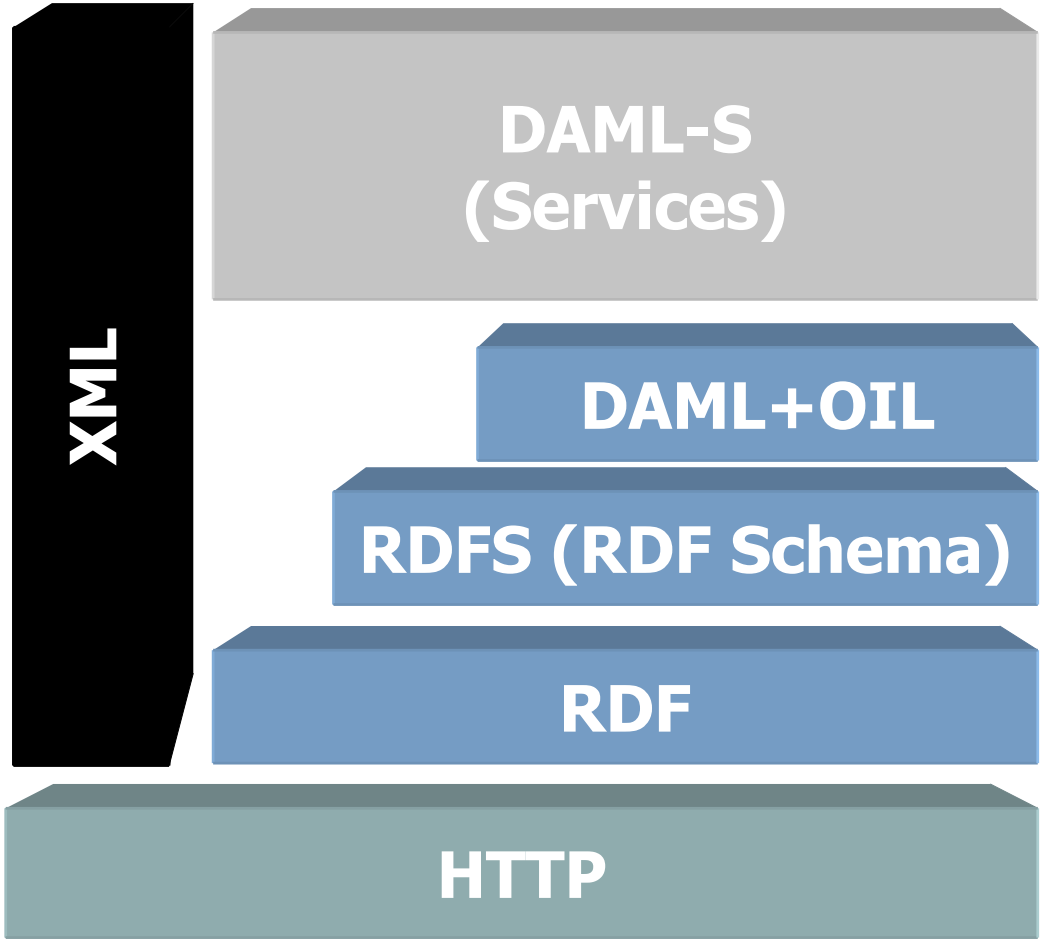
- An **upper ontology** for describing the **properties & capabilities of agents & (Web) services** in an unambiguous, computer interpretable markup language.
- Built as an additional layer above DAML+OIL
- Designed to the following automated tasks…

http://www.daml.org/services

Automation enabled by DAML-S

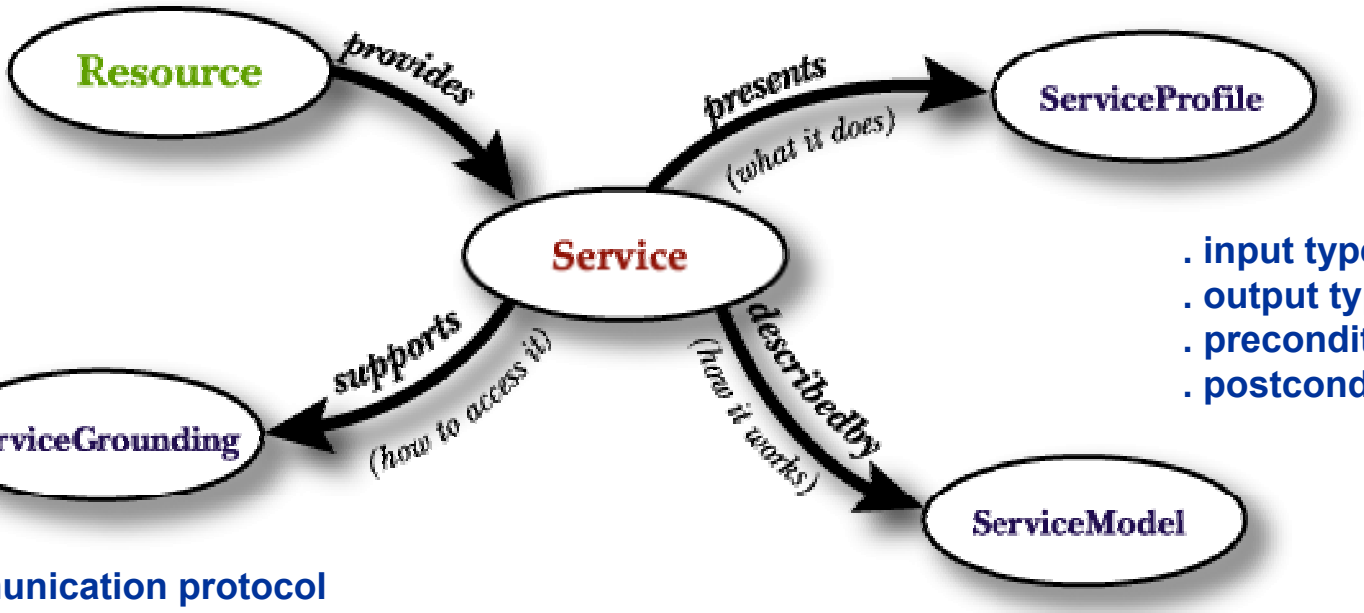
- Web Service Discovery & Selection
  - Find an airline that can fly me to Toulouse, France.
- Web Service Invocation
  - Book flight tickets from *AirFrance* to arrive 21st Aug.
- Web Service Composition & Interoperation
  - Arrange taxis, flights and hotel for travel from Lyon to Toulouse, OR, via Paris.
- Web Service Execution Monitoring
  - Has the taxi to Toulouse Blagnac Airport been reserved yet?

Layered Approach to Language Development



- The first major application of DAML+OIL
- Layer exists above DAML+OIL & RDF
- Future versions will build upon emerging layers (e.g. DAML-Rules etc)

DAML-S Upper Ontology

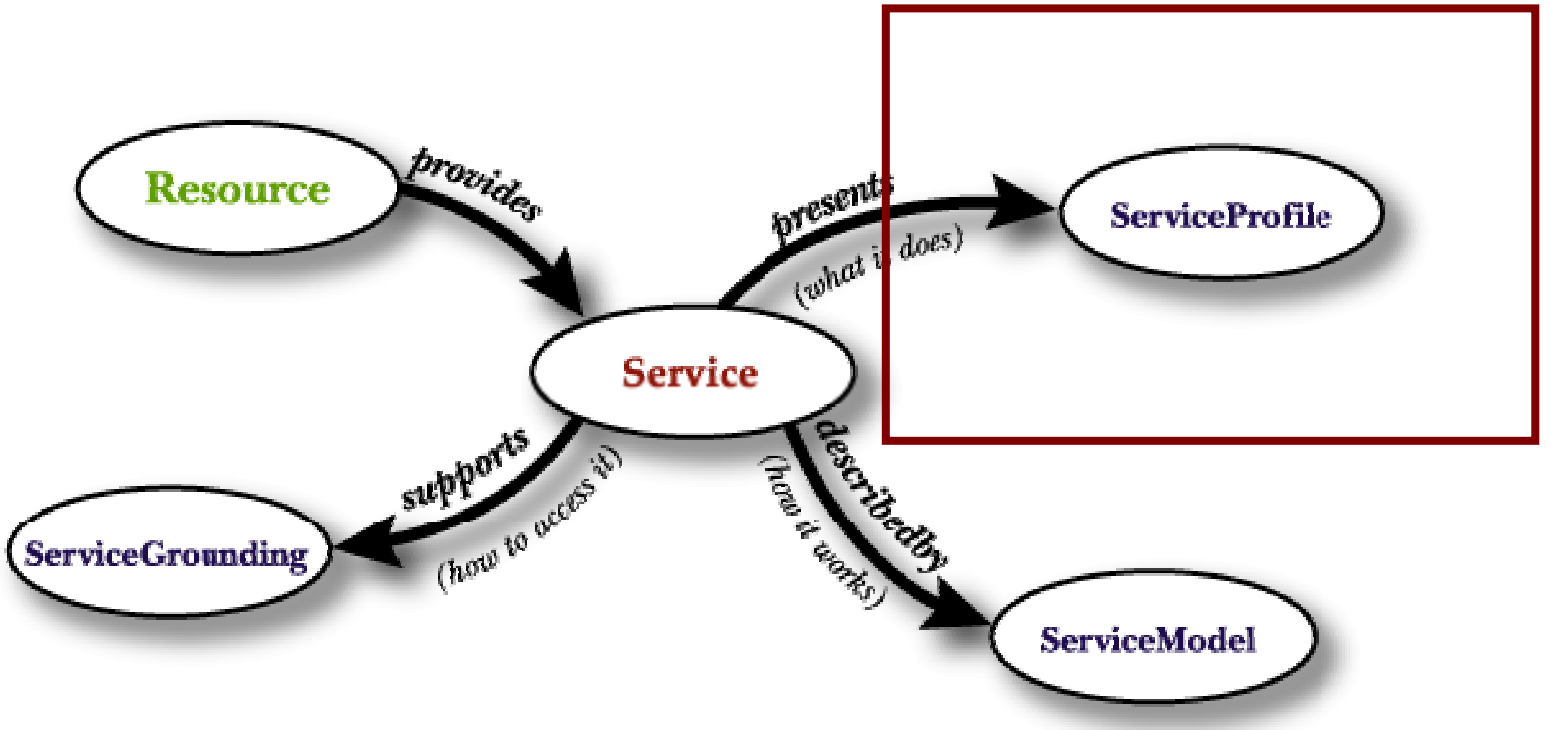


. input types
. output types
. preconditions
. postconditions

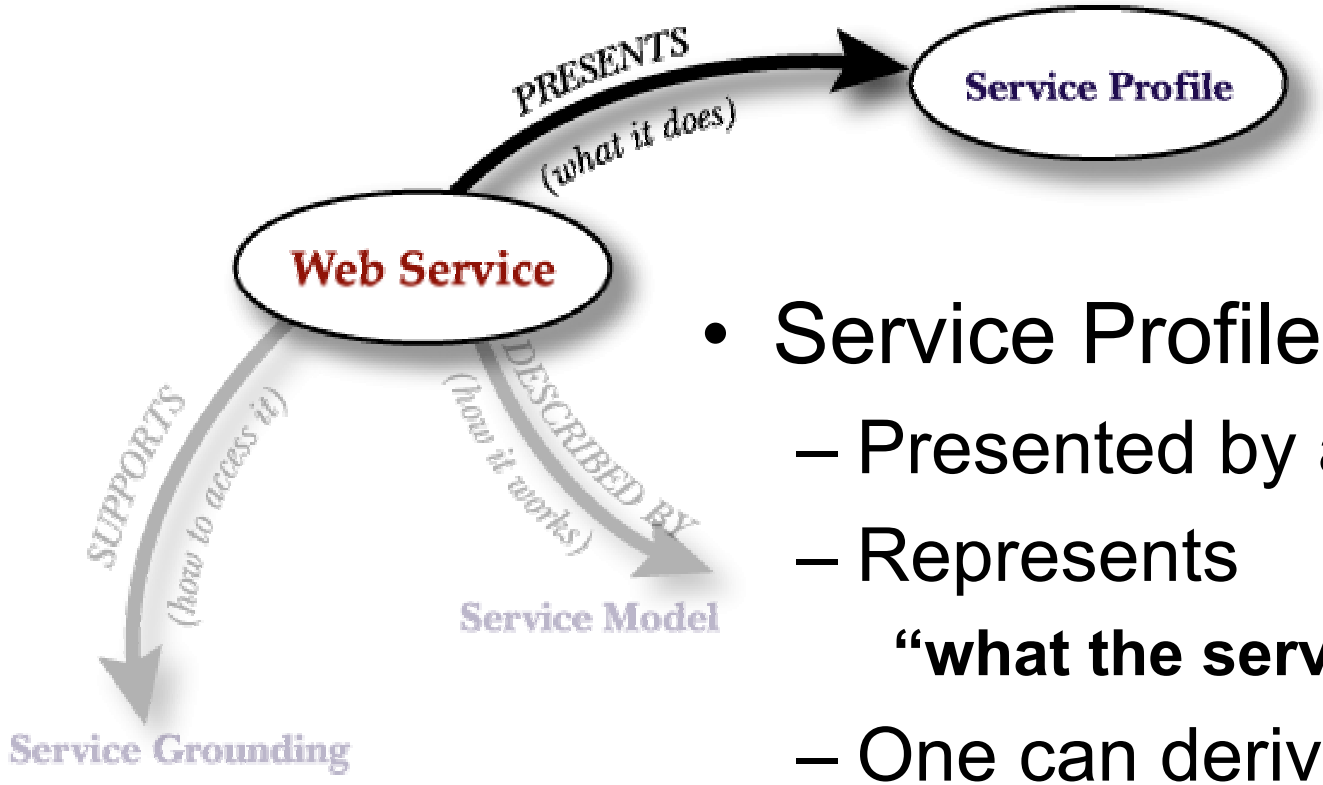
. communication protocol
(RPC, HTTP, …)
. port number
. marshalling/serialization

• process flow
• composition hierarchy
• process definitions

DAML-S Service Models



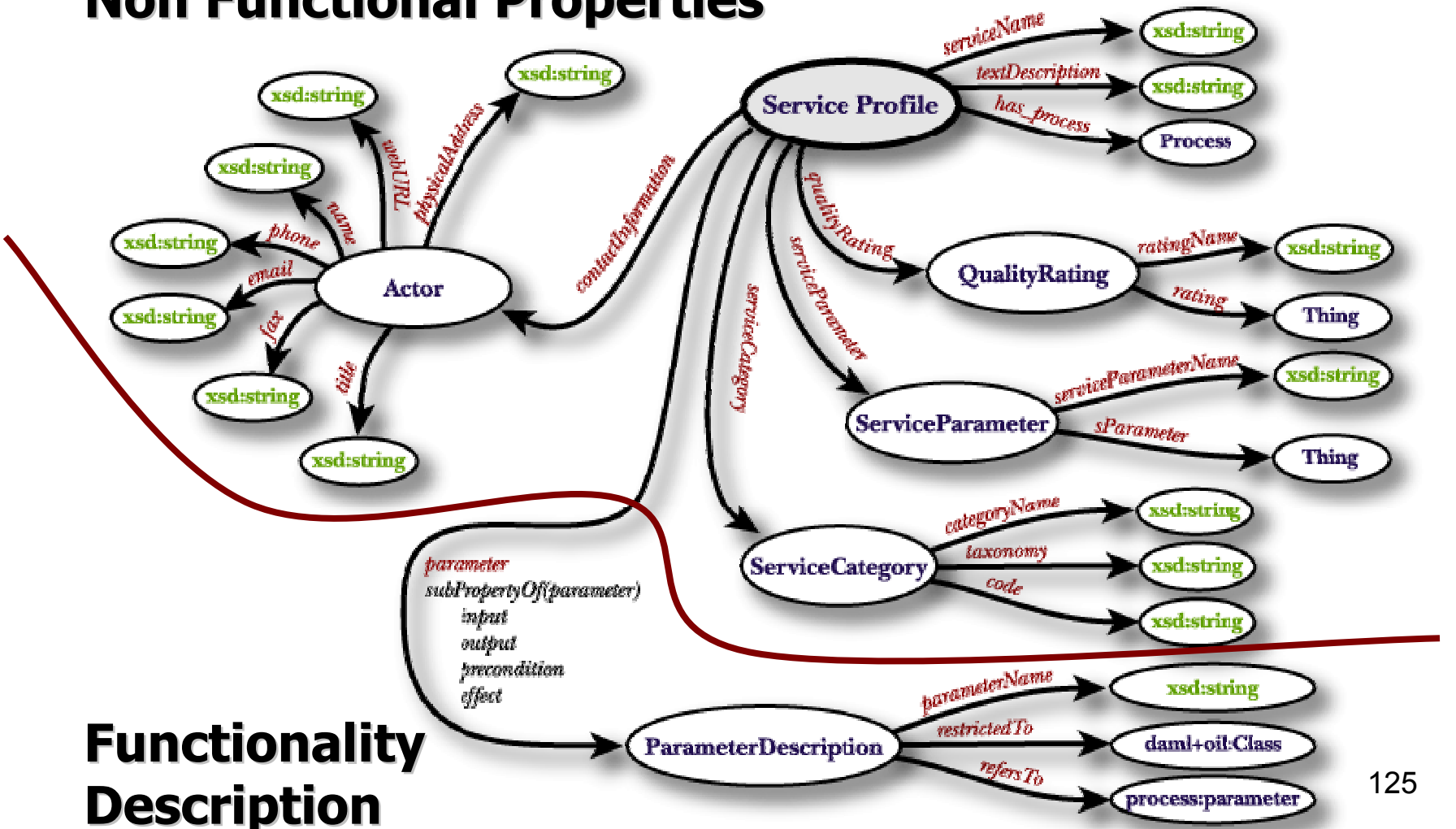
Presenting Service Profiles



- Service Profile
  - Presented by a service.
  - Represents
    "what the service provides
  - One can derive:
    - **Service Advertisements**
    - **Service Requests**

DAML-S Service Profile

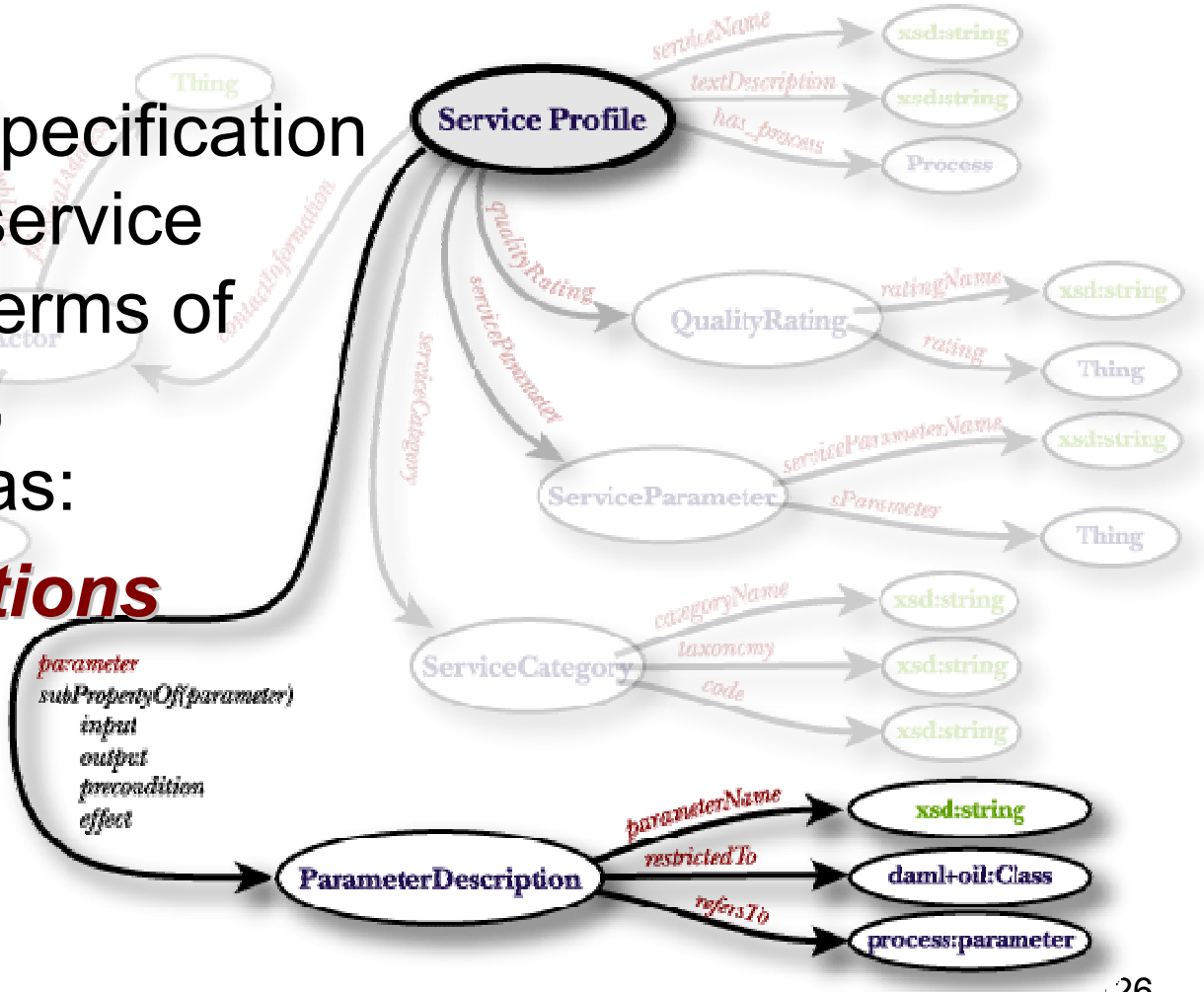
Non Functional Properties



Functionality Description

DAML-S Service Profile
Functionality Description

- Functional Specification of what the service provides in terms of ***parameters***, subclassed as:
  - ***preconditions***
  - ***inputs***
  - ***outputs***
  - ***effects***



DAML-S Service Profile
Functionality Description

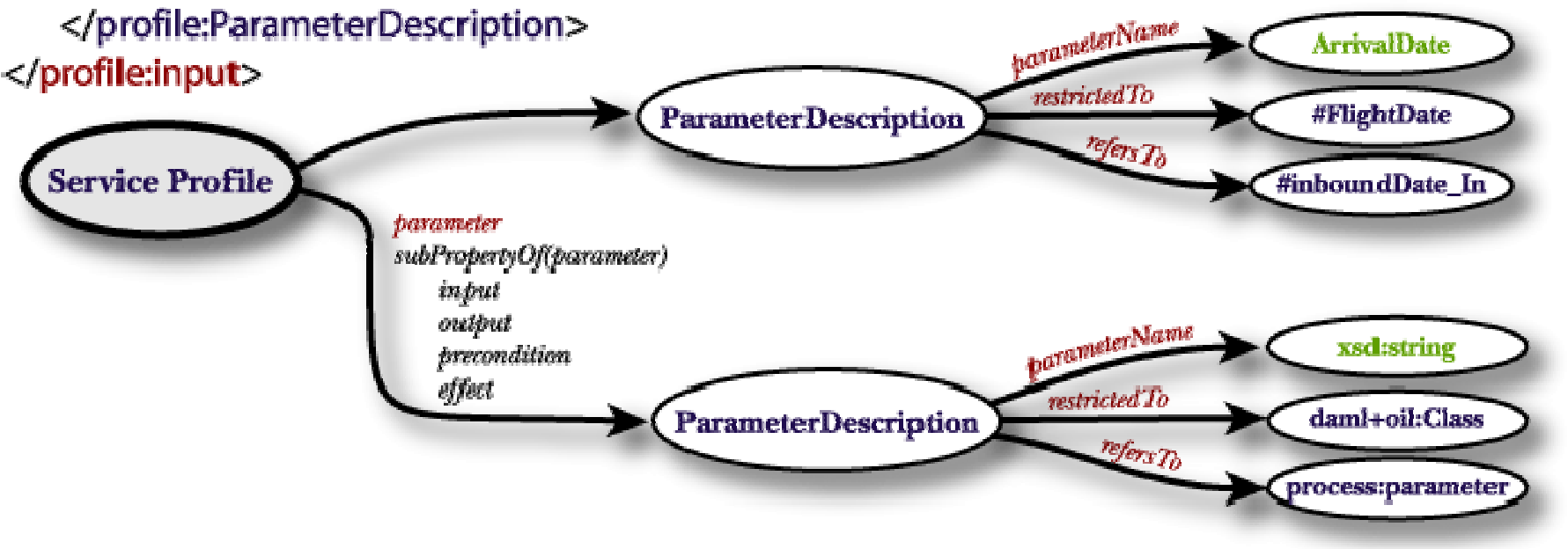
- ***Preconditions***
  - Set of conditions that should hold prior to service invocation
- ***Inputs***
  - Set of necessary inputs that the requester should provide to invoke the service
- ***Outputs***
  - Results that the requester should expect after interaction with the service provider is completed
- ***Effects***
  - Set of statements that should hold true if the service is invoked successfully.
  - Often refer to real-world effects
    - Package being delivered, or Credit card being debited

DAML-S Service Profile
Functionality Description

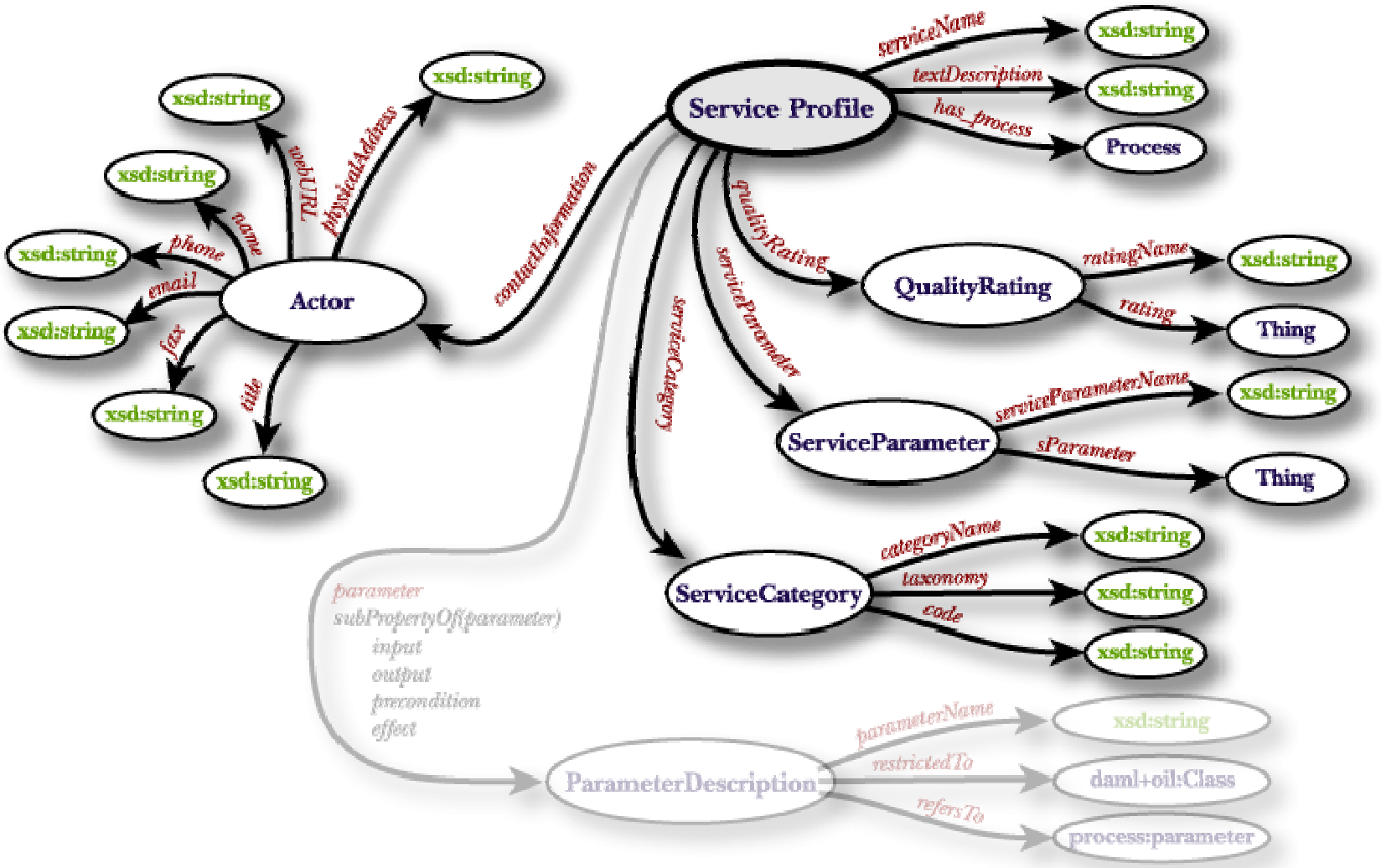
- An *Input/Output/Precondition/Effect* parameter has three properties:
  - ***parameterName***: the name of the parameter
  - ***restrictedTo*** : a resource corresponding to some RDF/DAML property type within some ontology (i.e. the range of a parameter instance)
  - ***refersTo*** : the corresponding parameter defined within the process model

DAML-S Service Profile
Functionality Description

```
<profile:input>
    <profile:ParameterDescription rdf:ID="ArrivalDate">
        <profile:parameterName>ArrivalDate</profile:parameterName>
        <profile:restrictedTo rdf:resource="http://www..../concepts.daml#FlightDate" />
        <profile:refersTo rdf:resource="http://www..../BravoAirProcess.daml#inboundDate_In" />
    </profile:ParameterDescription>
</profile:input>
```



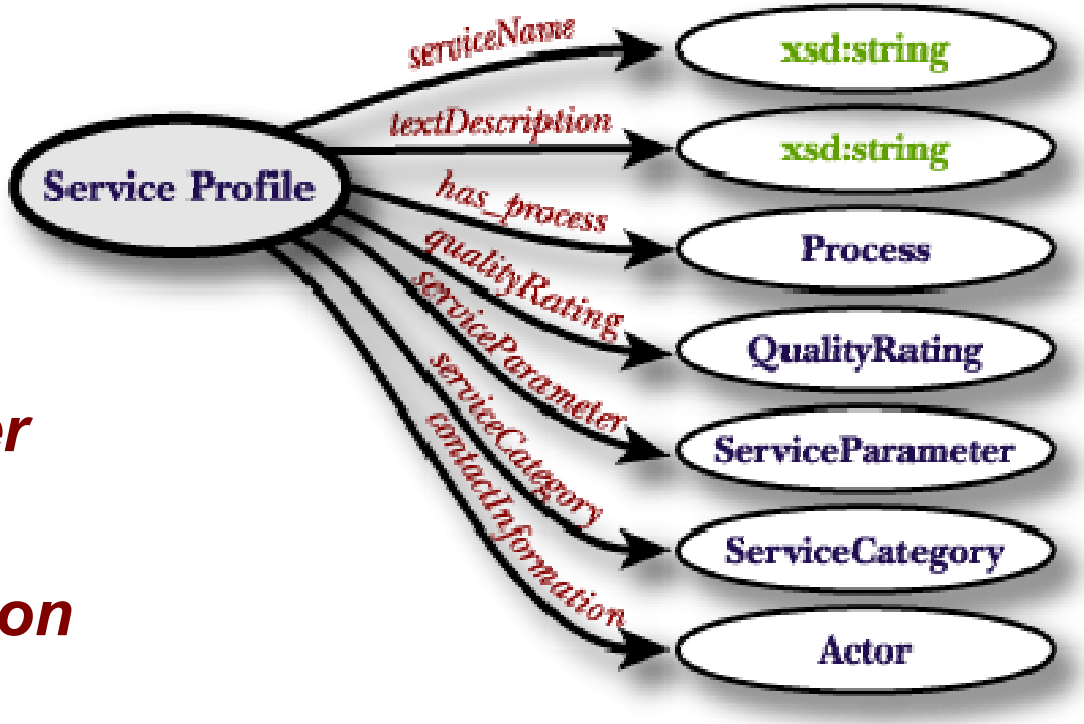
DAML-S Service Profile
Non Functional Properties



Provides supporting information about the service.

DAML-S Service Profile
Non Functional Properties

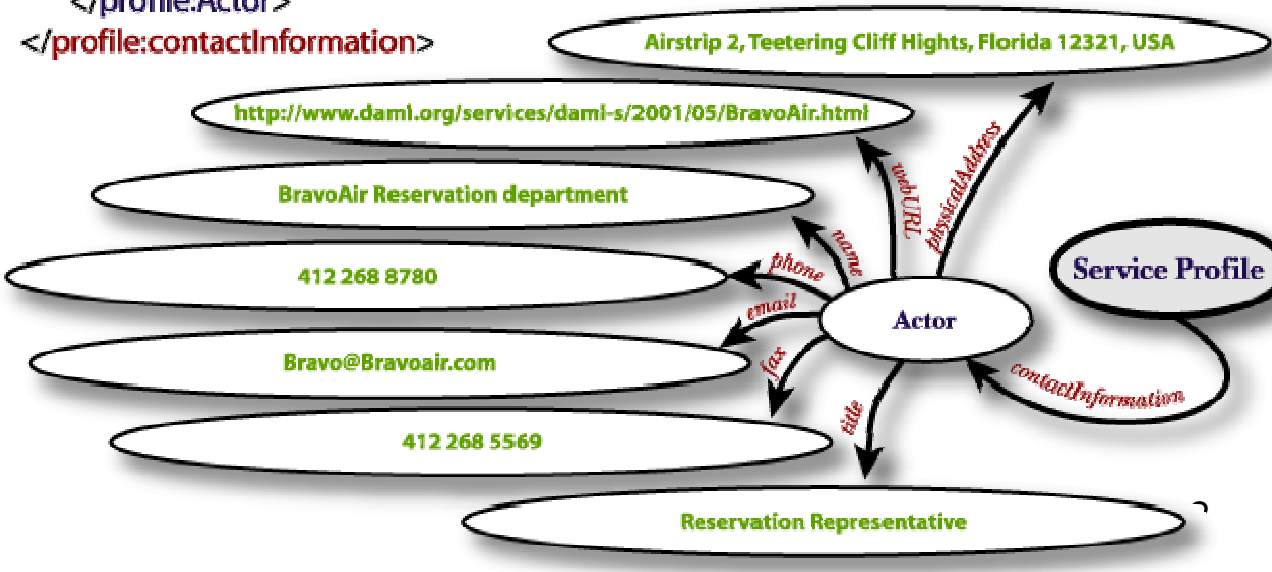
- These include
  - ***serviceName***
  - ***textDescription***
  - ***has_process***
  - ***qualityRating***
  - ***serviceParameter***
  - ***serviceCategory***
  - ***contactInformation***



DAML-S Service Profile

Non Functional Properties - Actor

```
<profile:contactInformation>
    <profile:Actor rdf:ID="BravoAir-reservation">
        <profile:name>BravoAir Reservation department</profile:name>
        <profile:title>Reservation Representative</profile:title>
        <profile:phone>412 268 8780</profile:phone>
        <profile:fax>412 268 5569</profile:fax>
        <profile:email>Bravo@Bravoair.com</profile:email>
        <profile:physicalAddress>
            Airstrip 2, Teetering Cliff Hights, Florida 12321, USA
        </profile:physicalAddress>
        <profile:webURL>http://www. .../BravoAir.html</profile:webURL>
    </profile:Actor>
</profile:contactInformation>
```



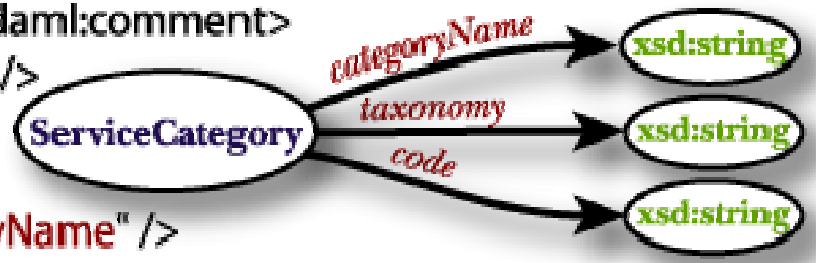
DAML-S Service Profile

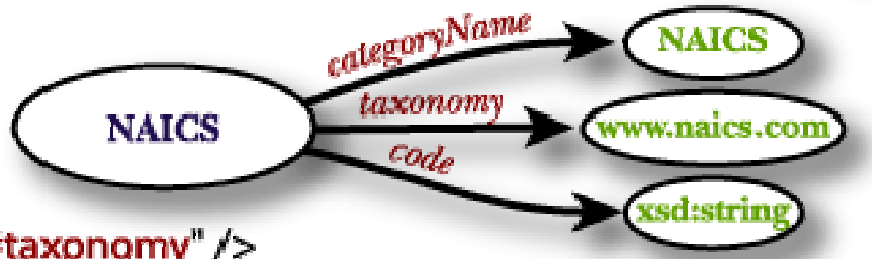
Non Functional Properties - QualityRating


```
<daml:Class rdf:ID="DAndBRating">
    <rdfs:subClassOf rdf:resource="#QualityRating" />
    <rdfs:subClassOf>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#ratingName" />
            <daml:toValue rdf:resource="Dun and Bradstreet Rating" />
        </daml:Restriction>
    </rdfs:subClassOf>
</daml:Class>
```

DAML-S Service Profile
Non Functional Properties – ServiceCategory

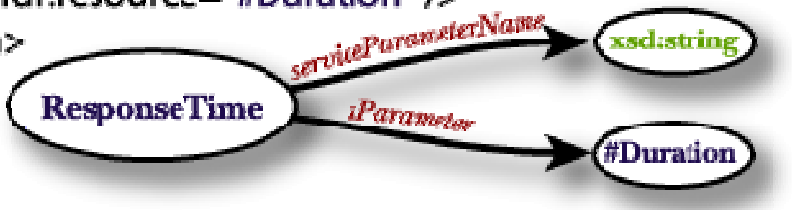
```
<daml:Class rdf:ID="NAICS">
    <daml:comment>Hook to the NAICS taxonomy</daml:comment>
    <rdfs:subClassOf rdf:resource="#ServiceCategory" />
    <rdfs:subClassOf>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#categoryName" />
            <daml:hasValue>NAICS</daml:hasValue>
        </daml:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#taxonomy" />
            <daml:hasValue>www.naics.com</daml:hasValue>
        </daml:Restriction>
    </rdfs:subClassOf>
</daml:Class>
```





DAML-S Service Profile
Non Functional Properties – ServiceParameter

```
<daml:Class rdf:ID="ResponseTime">
    <rdfs:subClassOf rdf:resource="#ServiceParameter" />
    <rdfs:subClassOf>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#sParameter" />
            <daml:toClass rdf:resource="#Duration" />
        </daml:Restriction>
    </rdfs:subClassOf>
</daml:Class>
```




```
<daml:Class rdf:ID="GeographicRadius">
    <rdfs:subClassOf rdf:resource="#ServiceParameter" />
    <rdfs:subClassOf>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#sParameter" />
            <daml:toClass rdf:resource="http://www..../Country.daml#Country" />
        </daml:Restriction>
    </rdfs:subClassOf>
</daml:Class>
```

Profile Hierarchy

Sub-classing the Profile model facilitates the creation and specialisation of service categories

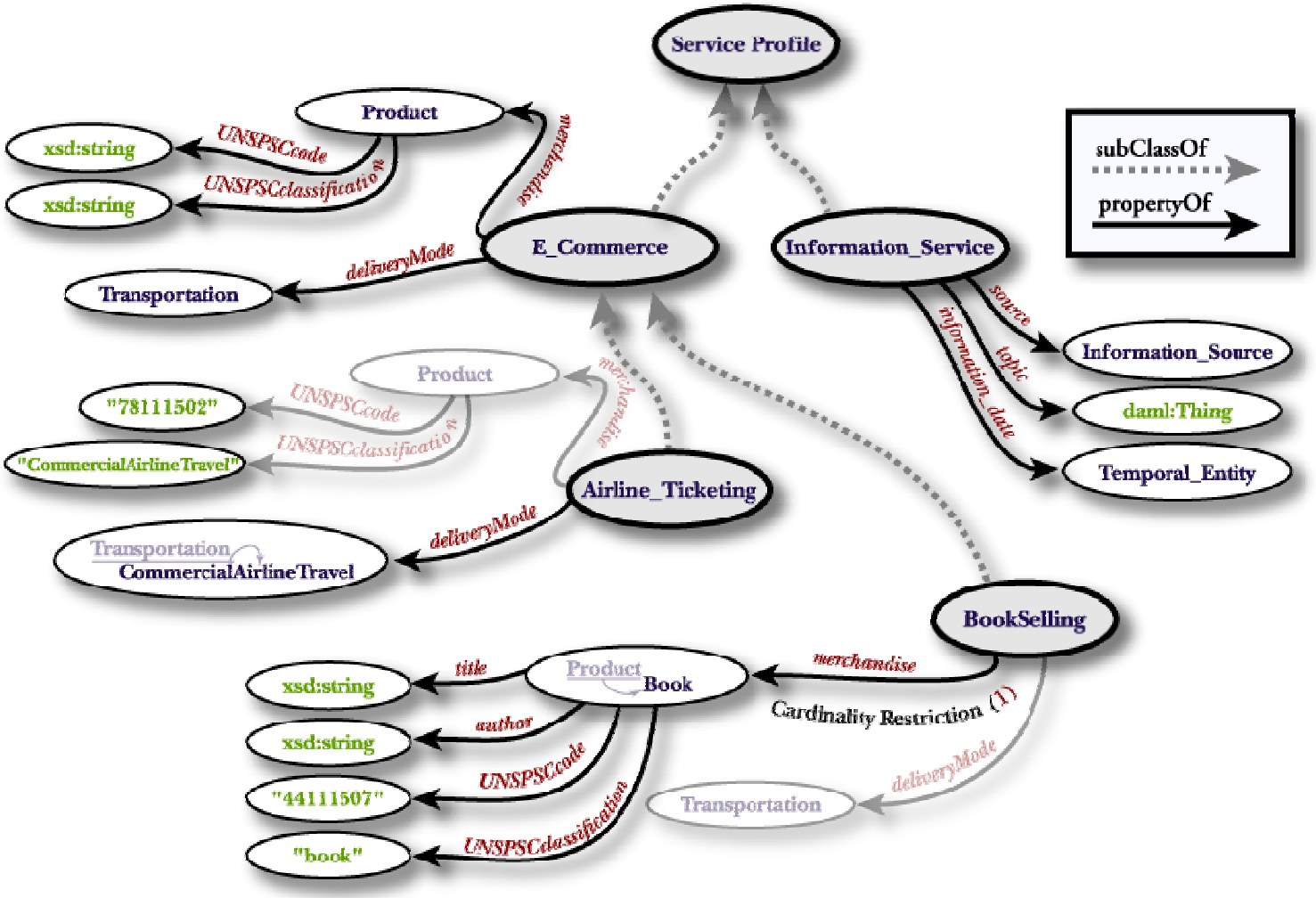
Each subclass can:

- Introduce new properties
- Place restrictions on existing properties

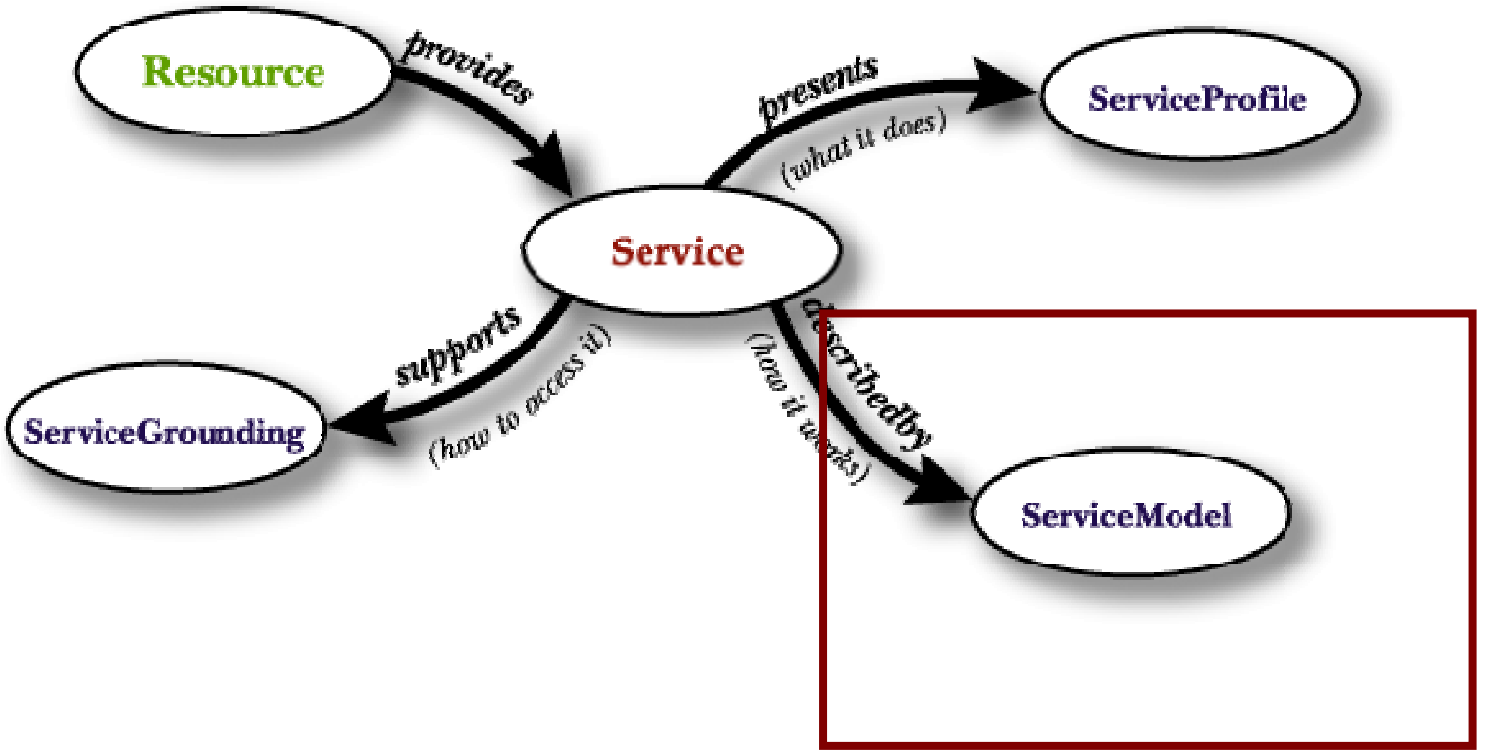
Sub-classing can also be used to specialise requests for service

An example Profile Hierarchy is provided, but others could just as easily be defined

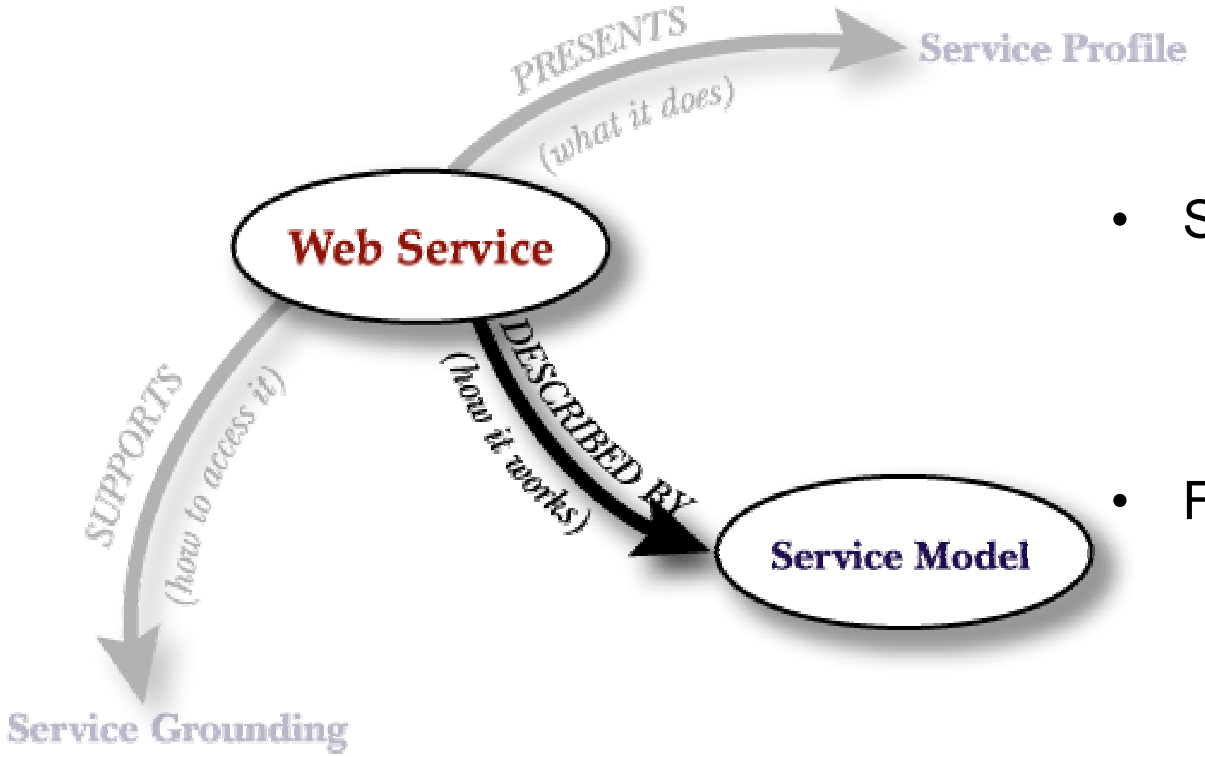
Profile Hierarchy – sample ontology



DAML-S Service Models

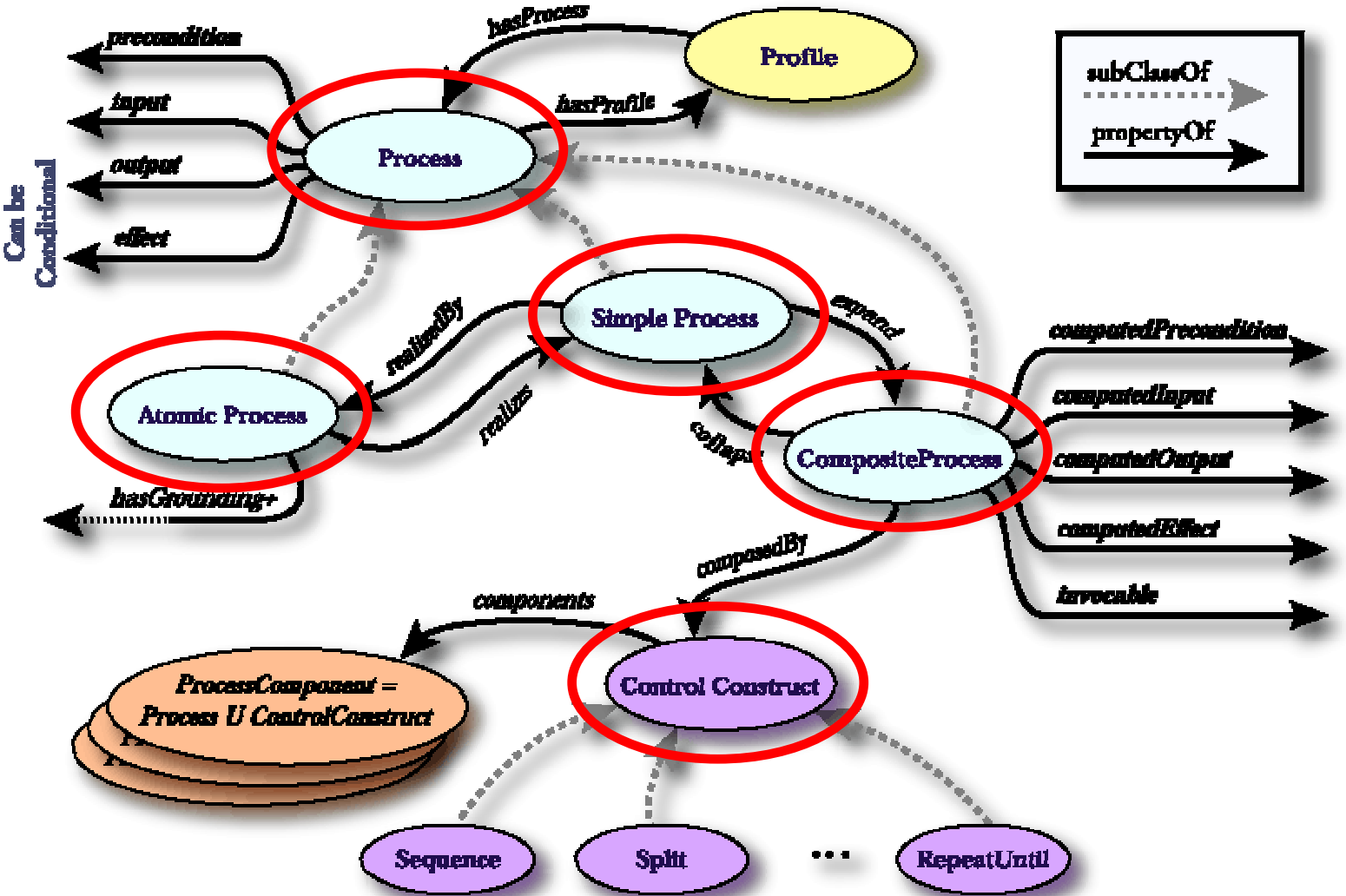


Describing Service Models



- Service Process
  - Describes how a service works.
- Facilitates
  - (automated) Web service invocation
  - composition
  - interoperation
  - monitoring

DAML-S Service Model (Overview)

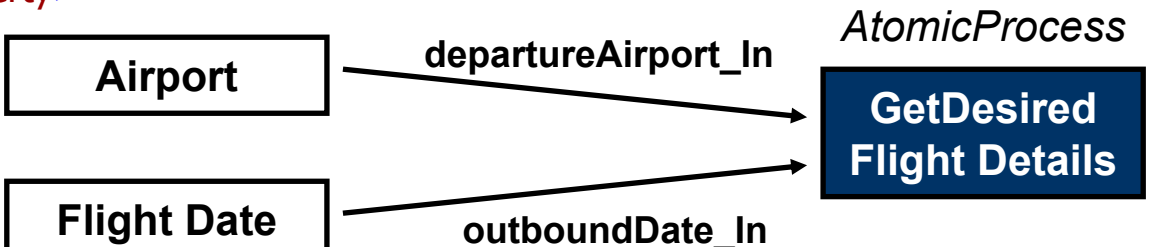


Types of the process in DAML-S

- **Atomic processes**: directly invokable (by an agent), have no subprocesses, executed in a single step.
- **Composite processes**: consist of other (non-composite or composite) processes.
  They have a *composedOf* property, by which the control structure of the process is indicated, using a ControlConstruct subclasses (see table …).
- **Simple processes**: abstract concepts, used to provide a view of some atomic process, or a simplified representation of some composite process (i.e., the "black box" view of a *collapsed* composite process).

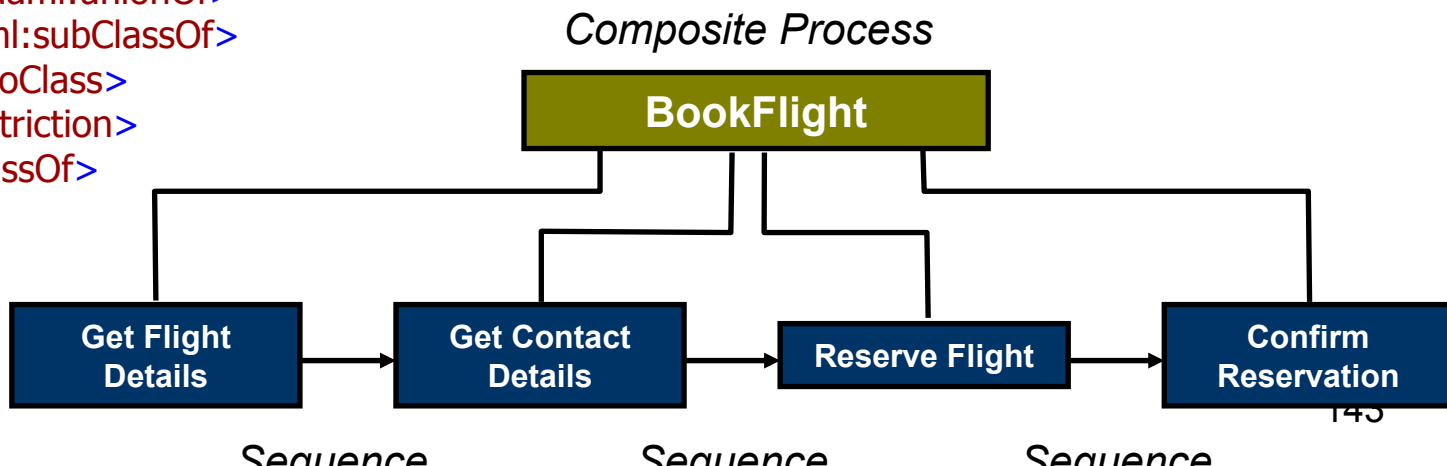
Atomic Process Example

```
<!– Atomic Process Definition - GetDesiredFlightDetails  -->
   <rdfs:Class rdf:ID="GetDesiredFlightDetails">
      <rdfs:subClassOf rdf:resource="http://www.daml.org/Process#AtomicProcess" />
   </rdfs:Class>

<!– (sample) Inputs used by atomic process GetDesiredFlightDetails -->
   <rdf:Property rdf:ID="departureAirport_In">
      <rdfs:subPropertyOf rdf:resource="http://www.daml.org/Process#input" />
      <rdfs:domain rdf:resource="#GetDesiredFlightDetails" />
      <rdfs:range rdf:resource="http://www.daml.ri.cmu.edu/ont/
                DAML-S/concepts.daml#Airport" />
   </rdf:Property>

   <rdf:Property rdf:ID="outboundDate_In">
      <rdfs:subPropertyOf rdf:resource="http://www.daml.org/Process#input" />
      <rdfs:domain rdf:resource="#GetDesiredFlightDetails" />
      <rdfs:range rdf:resource="http://www.daml.ri.cmu.edu/ont/
                DAML-S/concepts.daml#FlightDate" />
   </rdf:Property>
```

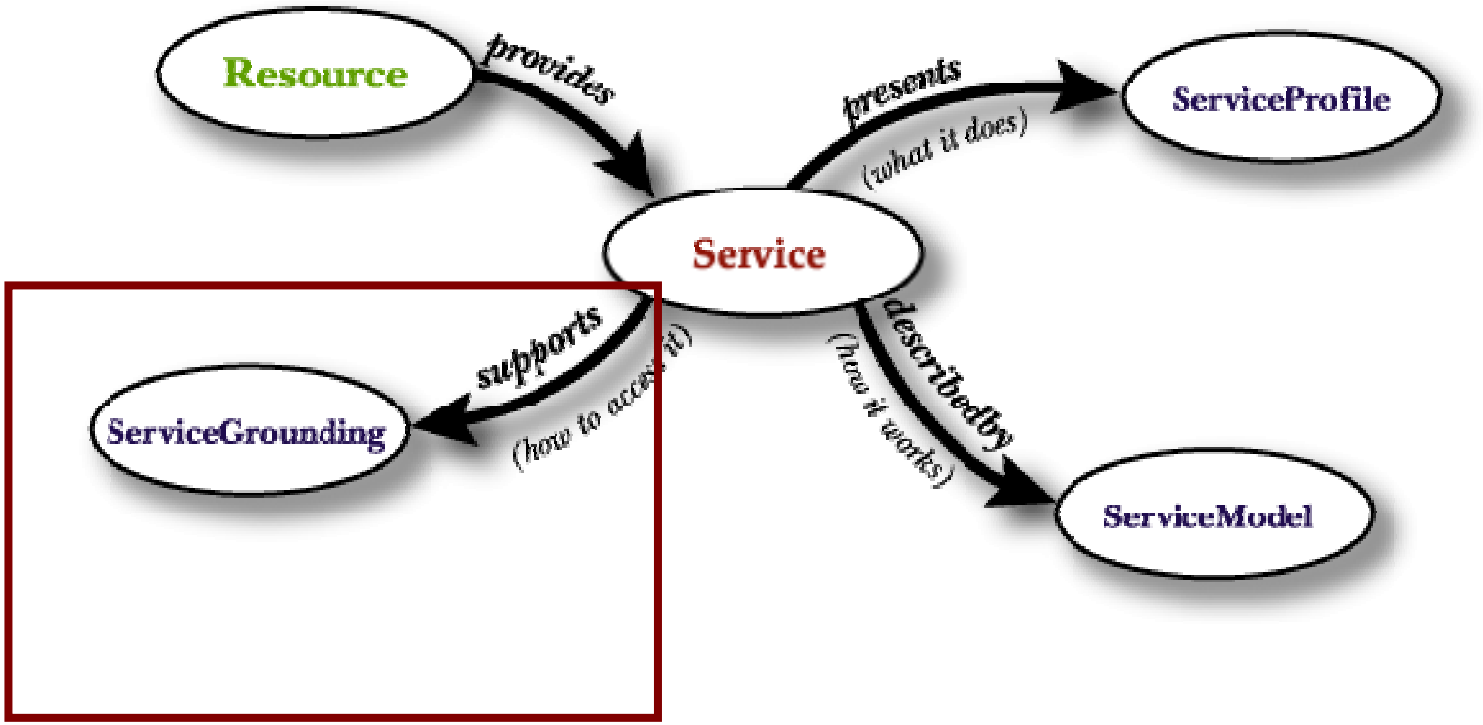


Composite Process Example

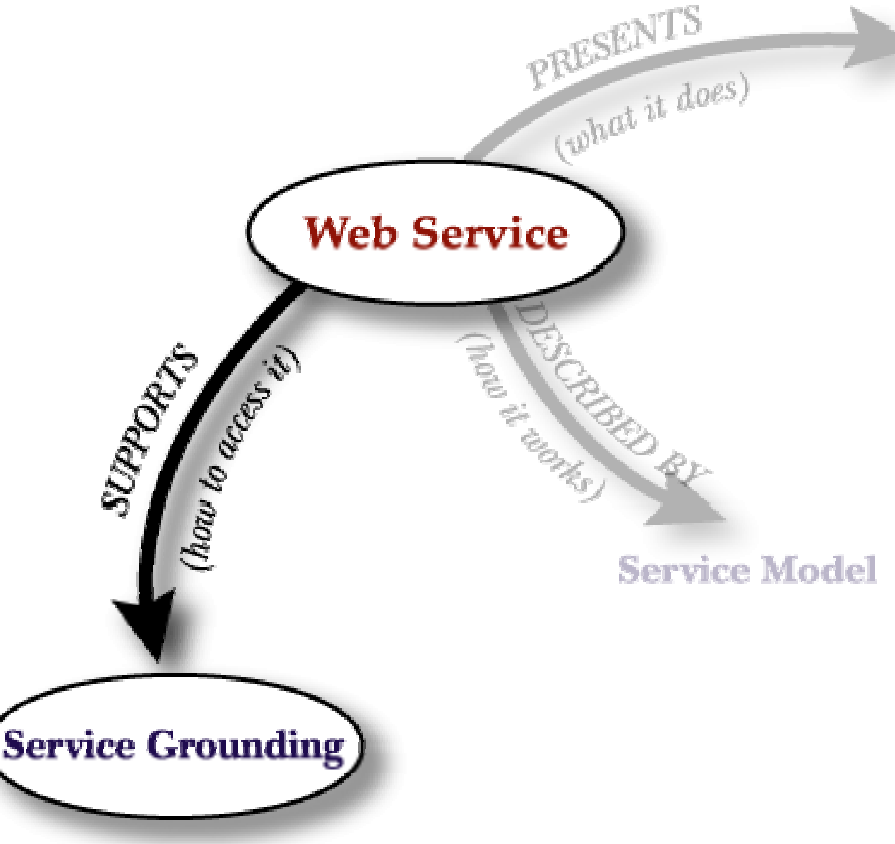
```
<rdfs:Class rdf:ID="BookFlight">
   <rdfs:subClassOf rdf:resource="#CompositeProcess" />
   <rdfs:subClassOf rdf:resource="http://www.daml.org/Process#Sequence" />
   <daml:subClassOf>
      <daml:Restriction>
         <daml:onProperty rdf:resource="http://www.daml.org/Process#components" />
         <daml:toClass>
            <daml:subClassOf>
               <daml:unionOf rdf:parseType="daml:collection">
                  <rdfs:Class rdfs:about="#GetFlightDetails" />
                  <rdfs:Class rdfs:about="#GetContactDetails" />
                  <rdfs:Class rdfs:about="#ReserveFlight" />
                  <rdfs:Class rdfs:about="#ConfirmReservation" />
               </daml:unionOf>
            </daml:subClassOf>
         </daml:toClass>
      </daml:Restriction>
   </daml:subClassOf>
</rdfs:Class>
```



DAML-S Service Models



Supporting a Service Grounding



- Service Grounding
  - Provides a specification of service access information.
  - Service Model + Grounding give everything needed for using the service
  - Builds upon **WSDL** to define message structure and physical binding layer
- Specifies:
  - communication protocols, transport mechanisms, agent communication languages, etc.

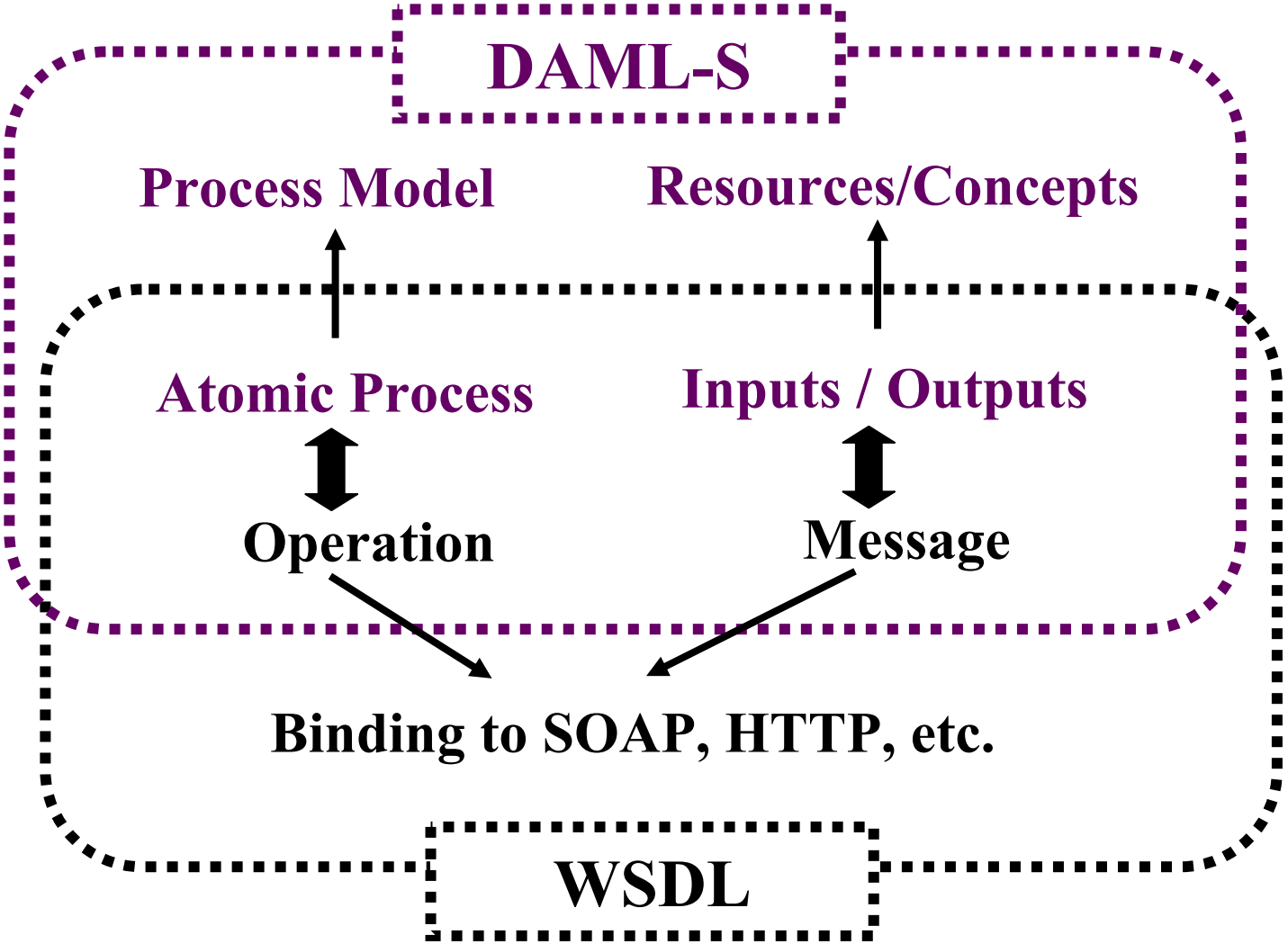
WSDL (Web Services Description Language)

- Structured mechanism to describe:
  - Abstract operations that a Web Service can perform
  - Format of messages it can process
  - Protocols it can support
  - Physical bindings to:
    - communication languages, e.g. SOAP or HTTP messages
    - Location of services, i.e. URI and port numbers
- XML based
- Current Status:
  - Developed by IBM and Microsoft
  - Version 1.1 submitted as a W3C Note

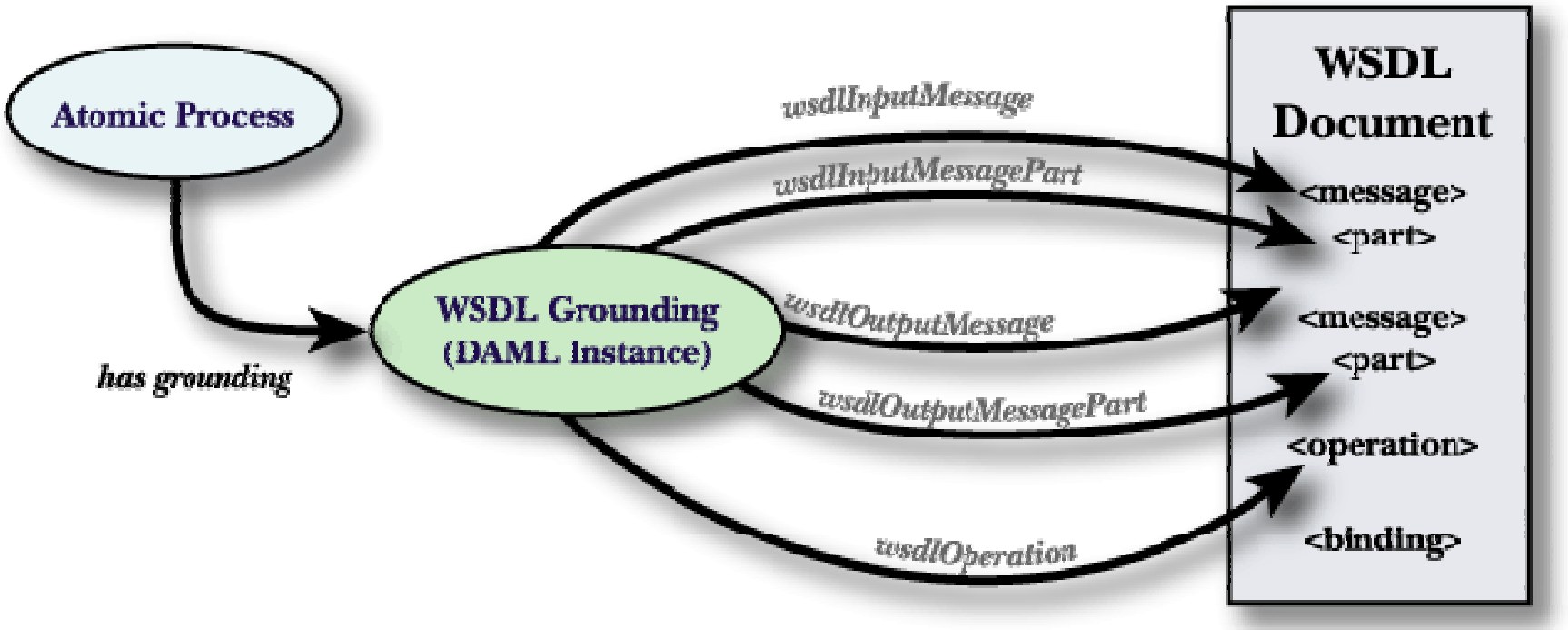
WSDL Components

- **Types** – containers for XSD data type definitions
- **Message** – abstract definition of the data being communicated
- **Operation** – abstract message exchange protocol
- **Port Type** – abstract set of operations
- **Binding** – concrete protocol and data format for a port type
- **Port** – single, physical endpoint
- **Service** – collection of related endpoints

DAML-S / WSDL Binding



DAML-S / WSDL Mapping



Service Grounding

The class **Service** *supports* a **ServiceGrounding**, that describes a mapping from an abstract (**ServiceProfile** and **ServiceModel**) to a concrete specification of the service description elements, that are required for interacting with the service, i.e. the inputs and outputs of atomic processes.

The central function of a DAML-S grounding is to show how the (abstract) inputs and outputs of an atomic process are to be realized concretely as messages, which carry those inputs and outputs in some specific transmittable format (e.g. RPC, CORBA, Java RMI, HTTP etc.).

Reasoning in DAML-S

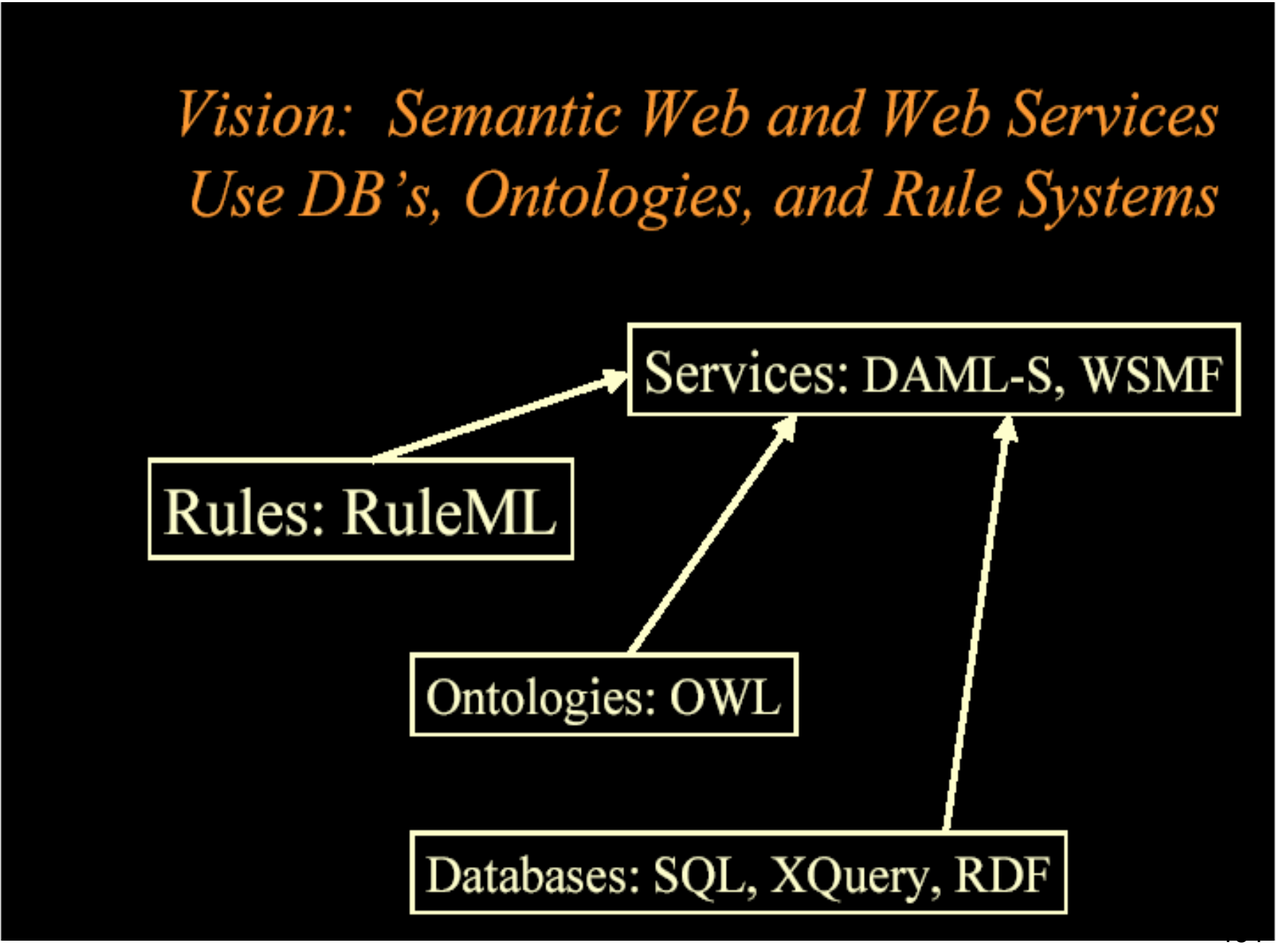
- Service requests are constructed as partial service descriptions.
- Requests are then evaluated against the advertised service taxonomy using subsumption (classification).
- Matches are generally recognized whenever the service advertised is subsumed by (is a particular case of) the service description requested.

Note: Advertisements and requests can differ sharply, in level of detail and in the level of abstraction of the terms used.

A whole example can be found at:

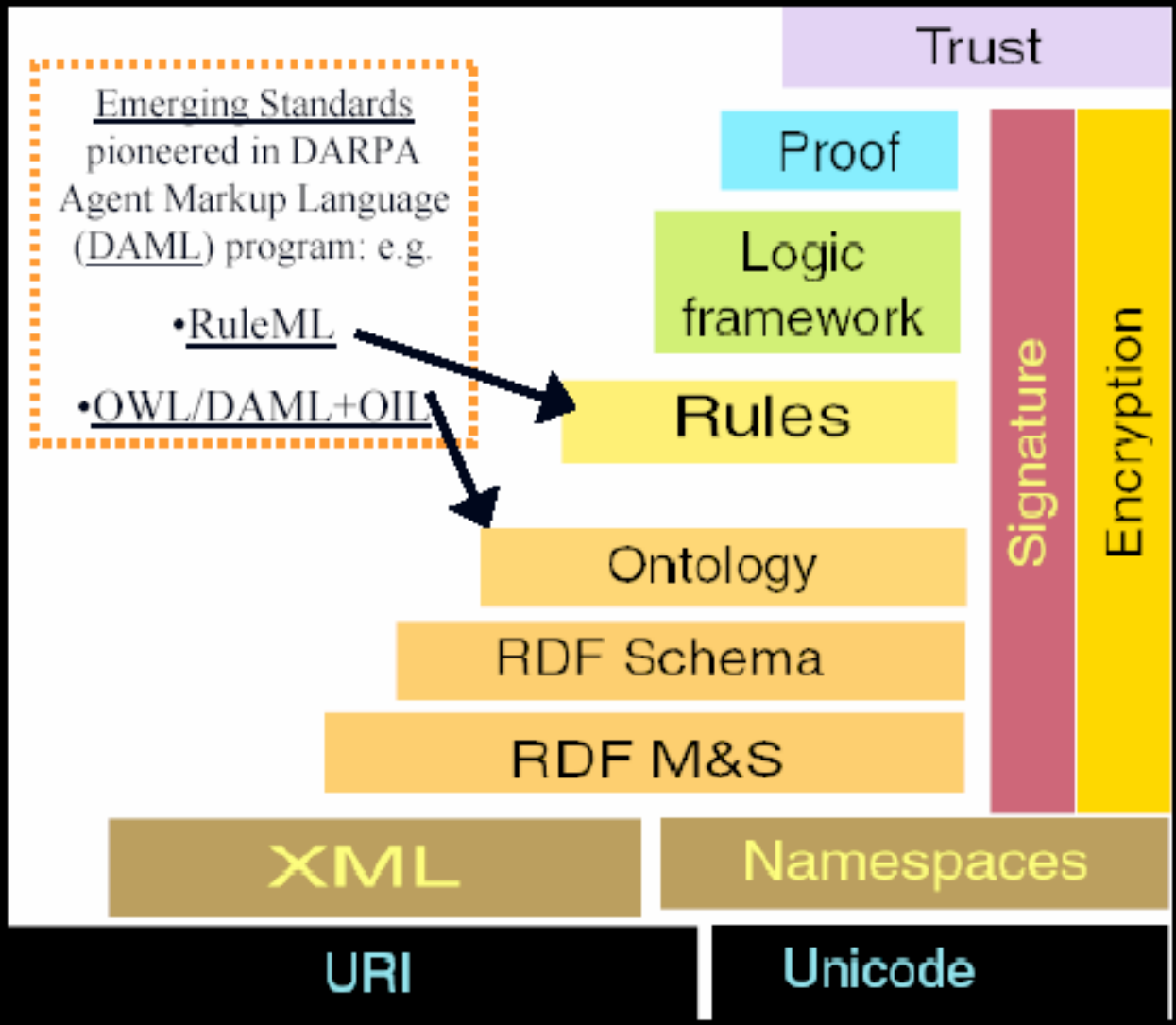
http://www.daml.org/services/daml-s/2001/05/Congo.daml

Conclusion



Slide taken from Grosof Talk

Semantic Web "Stack": Standardization Steps



[Diagram http://www.w3.org/DesignIssues/diagrams/sw-stack-2002.png is courtesy Tim Berners-Lee]

- **Logic**
  - I am an employee of UMBC.
    UMBC is a member of W3C.
    UMBC has GET access to http://www.w3.org/Member/.
    I (therefore) have access to http://www.w3.org/Member/.
- **Proof**
  - UMBC's document employList lists me as an employee.
    W3C'c member list includes UMBC.
    The ACLs for http://www.w3.org/Member/ assert that employees of members have GET access.
- **Trust**
  - UMBC's document employList is signed by a private key that W3C trusts to make such assertions.
    W3C'c member list is trusted by the access control mechansim.
    The ACLs for http://www.w3.org/Member/ were set by an agent trusted by the access control mechanism.

Some tools, software, and systems

Pellet : http://www.mindswap.org/2003/pellet/demo
OntoLink : http://www.mindswap.org/2004/OntoLink/
PhotoStuff : http://www.mindswap.org/2003/PhotoStuff/
Swoop and SwoopEd : http://www.mindswap.org/2004/SWOOP
METEOR-S : http://lsdis.cs.uga.edu/projects/METEOR-S/
GLUE : http://www.themindelectric.com
...

Resources

- **Web Services: Concepts, Architecture and Applications.** G. Alonso, F. Casati, H. Kuno, V. Machiraju, Springer Verlag 2004

- DAML+OIL
  - **http://www.daml.org/language**
- OWL
  - **http://www.w3.org/2001/sw/WebOnt/**
- DAML-S
  - **http://www.daml.org/services**

http://www.daml.org/services/

http://www.w3.org/2002/ws/

http://www.computer.org/intelligent/

http://classweb.gmu.edu/kersch/infs770/Topics/web_services.htm