

# *Autonomous Semantic Web Services*

Katia Sycara

Carnegie Mellon University

e-mail: [katia@cs.cmu.edu](mailto:katia@cs.cmu.edu)

[www.cs.cmu.edu/~softagents](http://www.cs.cmu.edu/~softagents)



## *Outline*

- **1. What are Web Services?**
2. Mediation and Composition
3. Industry Standards
4. Semantic Web efforts
5. Future Directions
6. Discussion



# *What are Web Services*

## *Today's Web*

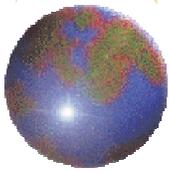
### Facts and figures

- ❏ 3 billion websites
- ❏ 450 m Internet users (33% US)
- ❏ Online B2B market volume 2000: \$282 billion

### Purpose

- ❏ Web designed for application to human interactions
  - ❏ Served very well its purpose:
  - ❏ Information sharing: a distributed content library
  - ❏ B2C e-commerce
  - ❏ Non-automated B2B interactions





# *What are Web Services*

## *The Next Killer App*

“Web services are expected to revolutionize our life in much the same way as the Internet has during the past decade or so.”  
(Gartner)

“By 2004, 40% of financial services transactions and 35% of online government services will be web service-based.”  
(Gartner)



“Just as the Web revolutionized how users talk to applications, XML transforms how applications talk to each other.” (Bill Gates)

“Web Services will be bigger than Java or XML” (Rod Smith, VP of Emerging Technology, IBM)

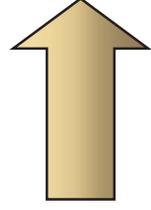


## *From the Internet to Web Services*

Old World :

“The eye-ball Web”

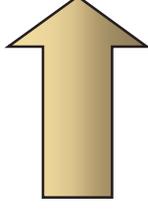
The architecture of the Web is geared towards delivering information visually (Internet filled with information)



New World:

“The transactional Web”

The architecture of the Web geared towards intelligently exchanging information between applications (Internet filled with executables)



Source: IBM

© Intelligent Agents Group, Carnegie Mellon University 2002



# *What are Web Services?*

## *Many Definitions Exist...*

It is software designed to be used by other software via Internet protocols and formats. (Forrester)

Web Services are self-describing components that can discover and engage other web services or applications to complete complex tasks over the Internet. (Sun Microsystems, Inc.)

Web Services are loosely coupled software components delivered over the Internet via standards-based technologies like XML, and SOAP. (Gartner)

“Self-describing, self-contained, modular unit of application logic that provides some business functionality to other applications through an Internet connection...” (UDDI.org)

Web Services are Internet-based, modular applications that perform a specific business task and conform to a particular technical format. (IBM)

A web service is application logic that is programmatically available, exposed using the Internet. (Microsoft)



## *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 as a programming technology*

- 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
  - HTTP defines a protocol to retrieve information from the web.
- Web services require a similar infrastructure:
  - XML provides a meta language for defining documents
  - Standards required for communication, interface/signature description, protocol description and discovery.
    - e.g. UDDI, WSDL, and SOAP



## *Key characteristics?*

- A Web Service is **accessible over the Web**.
- Web Services communicate using **platform-independent and language-neutral Web protocols**
- A Web Service provides a **specific functionality that can be used by other programs**
- A Web Service is **registered and can be located** through a Web Service Registry



## *So what is new about Web Services?*

Component-Based Model	Web Services Model
Tightly coupled <b>software applications (high dependencies between systems)</b>	Loosely coupled <b>software applications (low dependencies between applications)</b>
<b>Mainly designed for processes</b> within the enterprise	<b>Mainly designed for processes</b> across enterprises
<b>Uses</b> different protocols and technologies ( <b>e.g., Microsoft DCOM, CORBA</b> )	<b>Uses</b> common protocols and technologies ( <b>e.g., XML, SOAP, WSDL, HTTP</b> )



## *The Impact of Web Services?*

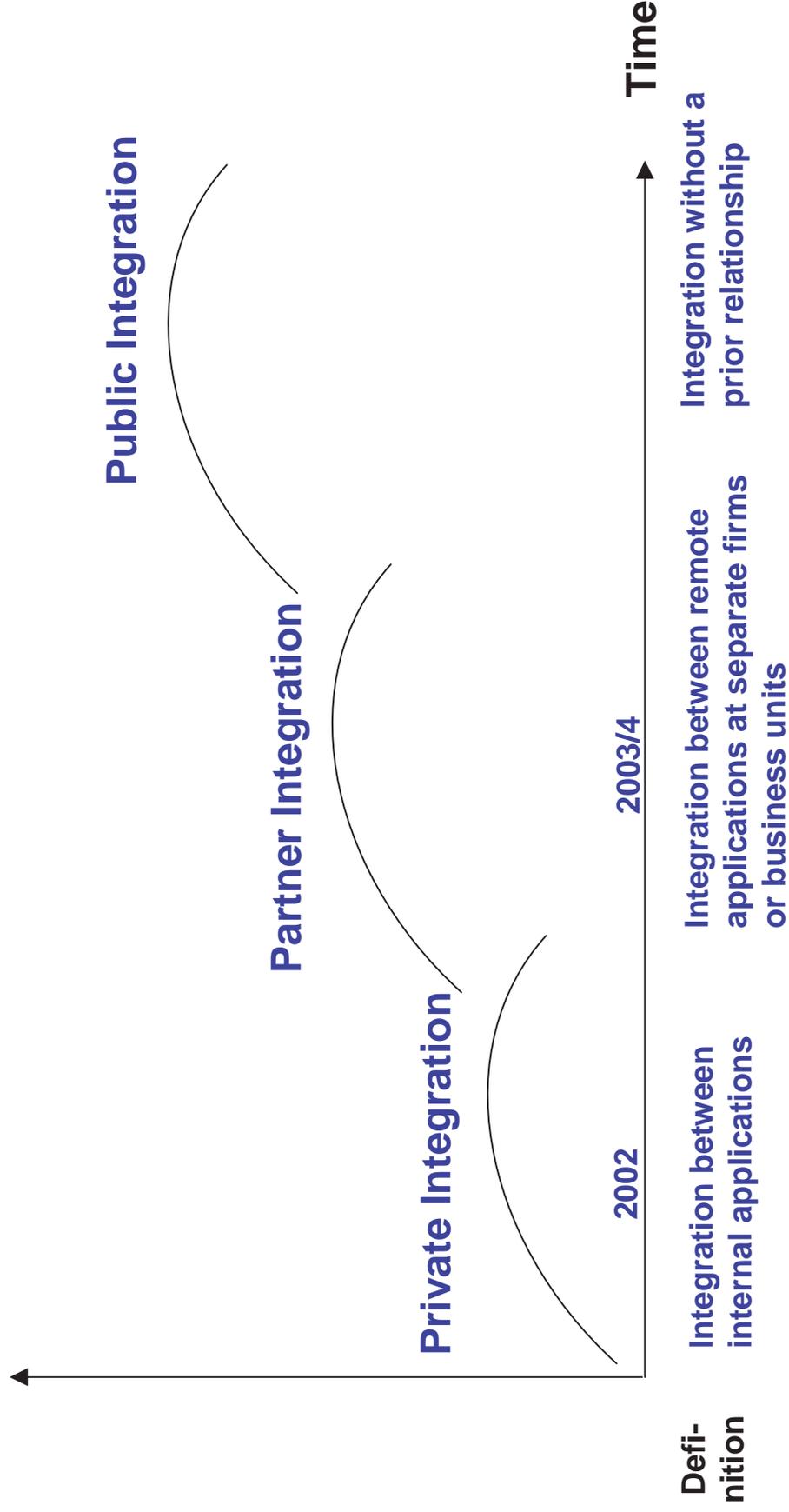
- Improvement of operations
- Agile business relationships
- Reduced cost and increased flexibility
- Shorter time-to-market for new products and services
- Leverage existing infrastructure

Web Services will remedy many expensive and painful problems of today's business uses of IT:

- ✚ Connecting business systems inside a firm is a nightmare
- ✚ Inter-enterprise process orchestration is impossible
- ✚ Inflexible systems impede business adjustments
- ✚ Fragmented personal data frustrates users

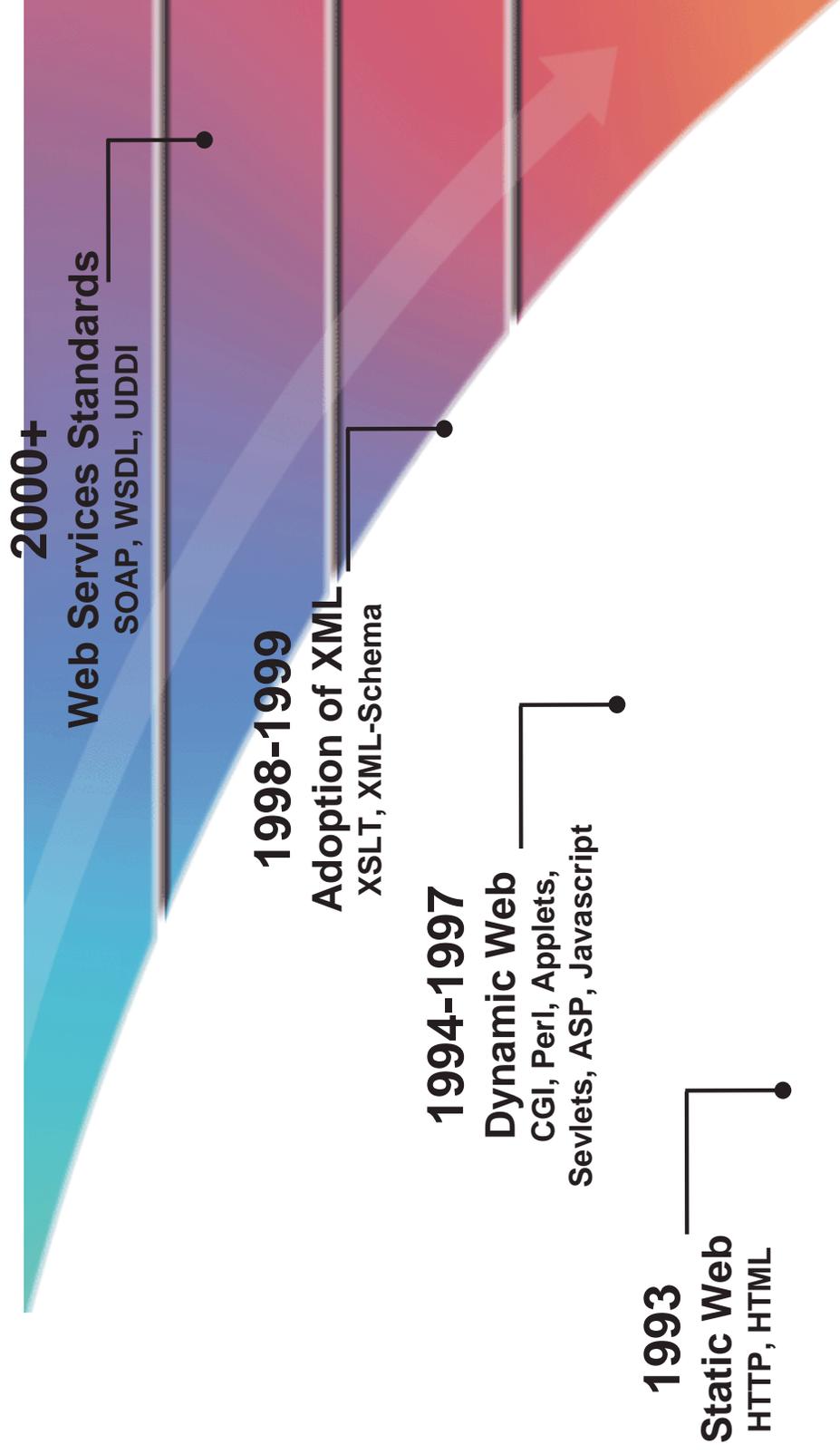


# *The Evolution of Web Services?*





# *Evolution of WWW Technologies & Tools*





# *Web Services and Multi Agent Systems*

- Web Service availability is dynamic
  - ▣ Available services are continually changing and evolving
    - Alternative or novel services appear due to the emergence of new businesses
    - Services may be withdrawn if unprofitable, or if businesses providing the service fail
  - ▣ Registries of Web Services used for service discovery
- Agents may enter or leave a MAS at will
  - ▣ No guarantees of availability
  - ▣ Middle Agents (e.g. *Facilitators* or *Matchmakers*) required to find available agents with given capabilities



# *Web Services and Multi Agent Systems*

- Web Service are heterogeneous
  - Vary in:
    - The tasks they perform (i.e. their capabilities)
    - The ontologies or taxonomies they use for:
      - Their descriptions (e.g. capability description or business model)
      - Content of exchanged data (i.e. XML documents)
  
- Agents are heterogeneous
  - Differ in
    - Functionality and capability description
    - Communication and message format
    - Infrastructure, coordination and mediation
    - Ontologies used for sharing information within and across infrastructures



# *Web Services and Multi Agent Systems*

- MAS may benefit from Web Services
  - Standards for different languages/tasks/roles
  - Industry developed APIs and Tools
  - Standardized ontologies and information repositories
  
- Web Services may benefit from MAS
  - Research into middle agents & mediation
  - Communication, conversation, and speech acts
  - Autonomous & Intelligent behaviors
  - Multi Agent Coordination Schemes



## *The Need for Semantics*

- **Semantic Interoperability is a major hurdle for**
  - **Locating Services**
    - Different terms used for advertisements and requests
  - **Invoking**
    - Constructing valid messages based on the published signature/interface of a service
  - **Understanding**
    - Interpreting the results of invoking a service
  - **Composing Services**
    - Constructing plans to achieve meta-goals based on available Services/Agents



## *Examples of Semantic Mismatch*

- Semantic Mismatch at the Content Level
  - Provider returns value *Pennsylvania*, but requester only understands two letter state codes (i.e. *PA*)
- Semantic Mismatch at the Attribute level
  - Requester needs *rainfall* but provider provides *precipitation*
- Semantic Mismatch at the level of Units
  - Requester has value in *inches*, but provider requires *cm*
- Semantic Mismatch at the Input & Output level
  - Requester has *length & width*, provider requires *area*

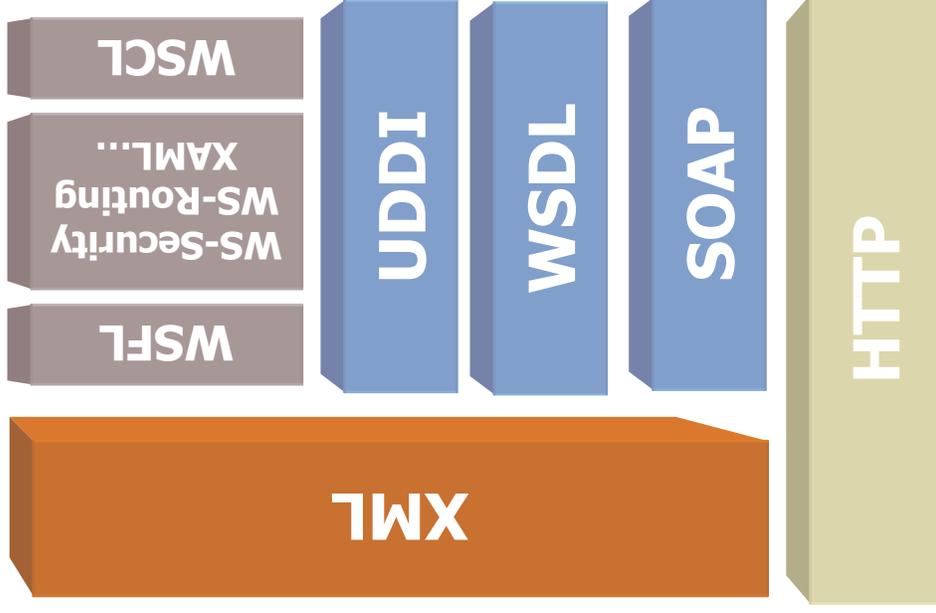


## *The Need for Semantics for WS & MAS*

- Both Web Services and Multi Agent Systems benefit from inclusion of semantics
  - For example, DAML - DARPA **Agent** Markup Language was designed to provide ontologies and description logics for Agent Markup to improve interoperability
  - Semantic Web provides open, extensible, semantic framework for describing and publishing semantic content
- Benefits?
  - Improved interoperability
  - Automated service composition, discovery and invocation
  - Access to knowledge on the internet



# Overview of Web Services Standards



- Data and Control Flow descriptions of Web Services; Security and Management
- A mechanism for registering and looking up web services
- Programmatic way of describing the Web Service Interface
- Web Services Communication protocol



## *Too Many Standards?*

- Many other Web Services Standards exist:
  - ✚ Transport
    - **DIME** - Direct Internet Message Encapsulation
    - **HTTPR** - Reliable HTTP
  - ✚ Packaging & Extensions
    - **SOAP-DSIG** - SOAP Security Extensions: Digital Signature
    - **SWA** - **SOAP** Messages with Attachments
    - **WS-License** - Web Services License Language
    - **WS-Referral** - Web Services Referral Protocol
    - **WS-Routing** - Web Services Routing Protocol
    - **WS-Security** - Web Services Security Language

Source: Pavel Kulchenko - <http://www.xml.com/pub/a/2002/01/09/soap.html?page=1>

© **Intelligent Agents Group, Carnegie Mellon University 2002**



## *Too Many Standards?*

- Many other Web Services Standards exist:
  - ▣ Description
    - **WSCM** - Web Services Component Model
    - **WSMF** - Web Services Modeling Framework
    - **WSML** - Web Services Meta Language
    - **WSOL** - Web Service Offerings Language
    - **WSXL** - Web Services Experience Language
    - **WSUI** - Web Services User Interface
    - **XLANG** - Web Services for Business Process Design
  - ▣ Discovery
    - **USML** - UDDI Search Markup Language
    - **WS-Inspection** - Web Services Inspection Language

Source: Pavel Kulchenko - <http://www.xml.com/pub/a/2002/01/09/soap.html?page=1>

© Intelligent Agents Group, Carnegie Mellon University 2002



## *Focusing on a subset of Industry Standards*

- Communication
  - **SOAP**
- Interface/Signature Description
  - **WSDL**
- Quality of Service & Service Parameters
  - **WSEL**
- Process Flow
  - **WSFL & WSCL (BPEL4WS)**
- Service Discovery
  - **UDDI**



# *Standards for Web Services on the Semantic Web*

- Representation
  - **RDF**
- Description Logic
  - **DAML+OIL**
- Service Ontologies
  - **DAML-S**
    - *Service Profiles*
    - *Process Models*
    - *Service Grounding*



## *Outline*

1. What are Web Services?
- **2. Mediation and Composition**
3. Industry Standards
4. Semantic Web efforts
5. Future Directions
6. Discussion



## *Mediation of Services*

- Two types of interaction between Agents and Web Services
  - Peer-2-Peer
    - Agents know who to talk to and how
  - Mediated interaction
    - Agents contact a 3rd party to assist with location and/or interaction with other agents



## *Mediation via Middle Agents*

- Middle Agents provide mediation for:
  - Processing agent capabilities and service descriptions
    - e.g. matching requests with advertisements
  - Semantic Interoperability between Agents & Services
    - e.g. resolving semantic mismatches between capability descriptions, or identifying articulations between ontologies
  - Management of Data and Knowledge
    - e.g. storing and managing registered advertisements; adapting and refining matching algorithms.



## *Types of Middle Agents*

- Middle Agents can be categorized by what knowledge is shared between provider and requester.
  - Providers have capabilities
  - Requesters have preferences

Preferences initially known by	Provider	Provider & Middle Agent	Capabilities initially known by	Provider, Middle Agent & Requester
Requester	Broadcaster	Front-Agent	Matchmaker/ Yellow Pages	
Requester & Middle Agent	Anonymizer	Broker/Facilitator	Personal Assistant/ Recommender	
Requester, Middle Agent & Provider	Blackboard	Introducer/ Bodyguard	Arbitrator	



## *Mediating Communication between Agents*

- Middle Agents may mediate communication between Agents & Web Services:
  - ▣ Agents may use different communication languages
    - e.g. SOAP vs. KQML
  - ▣ Agent transactions may assume different protocols and policies
    - e.g. virtual marketplaces may assume different auction policies
  - ▣ Semantic mismatch of knowledge may require intermediary to translate between ontologies



## *Mediating Service Discovery*

- Internet is a dynamic environment
  - ▣ Available services are continually changing and evolving due to competition
  - ▣ Alternative or novel services appearing due to the emergence of new businesses
  - ▣ Services may be withdrawn if unprofitable, or if businesses providing the service fail
- Scalability
  - ▣ In recent years, there has been a huge increase in the number of:
    - (Human-oriented) B2C services on the web
    - B2B services available for eCommerce



# *Mediating Reliability, Security & Trust*

- Quality of Service
  - Middle agent should comply with data requirements, standards and policies regarding knowledge and data stored
- Trust Management
  - Provide guarantees that service providers provide the service they advertise
  - Prevent abuse of shared, private information
    - such as selling contact information or preferences
- Security, Authorization and Verification services
  - Certification authorities and encryption keys



# *Service Location & Mediation for Web Services*

- Several standards can be used to describe Web Services
  - Registries provide White or Yellow Pages descriptions
    - e.g. UDDI
  - Capability descriptions can be provided using inputs/outputs
    - e.g. WSDL
  - Quality of Service descriptions can be defined through Business Descriptions
    - e.g. WSEL



# *Service Location & Mediation for Web Services*

- Few automated Web Service registries exist
  - UDDI provides keyword/wildcard searches through SOAP requests or via a Web Page
  - Web Repositories (e.g. [www.salcentral.org](http://www.salcentral.org)) provide keyword based human oriented search engines for WSDL descriptions



# *Service Location & Mediation for Web Services*

- Emerging Automated Web Service Registries
  - DAML-S Matchmaker provides automated Web Service discovery for DAML-S Profiles
    - Matches are based on capability descriptions or White page keyword searches
    - performs subsumption-based matching for agent capabilities
  - PQL - process query language
    - Used for locating Services described by process descriptions (taken from the MIT Process Handbook)



## *Composition of Services*

- Two definitions commonly used:
  - The workflow or business model of a service.  
Describes:
    - conversation & interaction between provider and requester
    - abstract description of internal model assumed by provider
  - On the fly construction of plans that achieve meta goals based on available services.



## *Composition of Services*

- Services may themselves be composed by a number of other services.
  - ▣ Can be broken down into a hierarchy of subtasks
  - ▣ Subtasks may be part of a larger service offered by a service provider
    - e.g. process of logging into an account
  - ▣ May be offered by a different service provider
    - e.g. booking a hotel as part of a travel plan

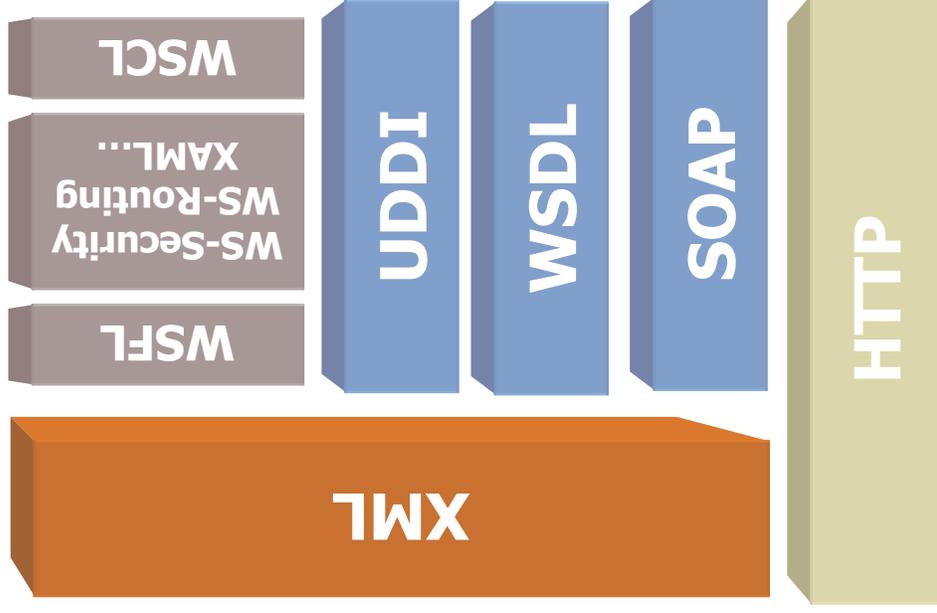


## *Outline*

1. What are Web Services?
2. Mediation and Composition
- **3. Industry Standards**
4. Semantic Web efforts
5. Future Directions
6. Discussion



# Overview of Web Services Standards

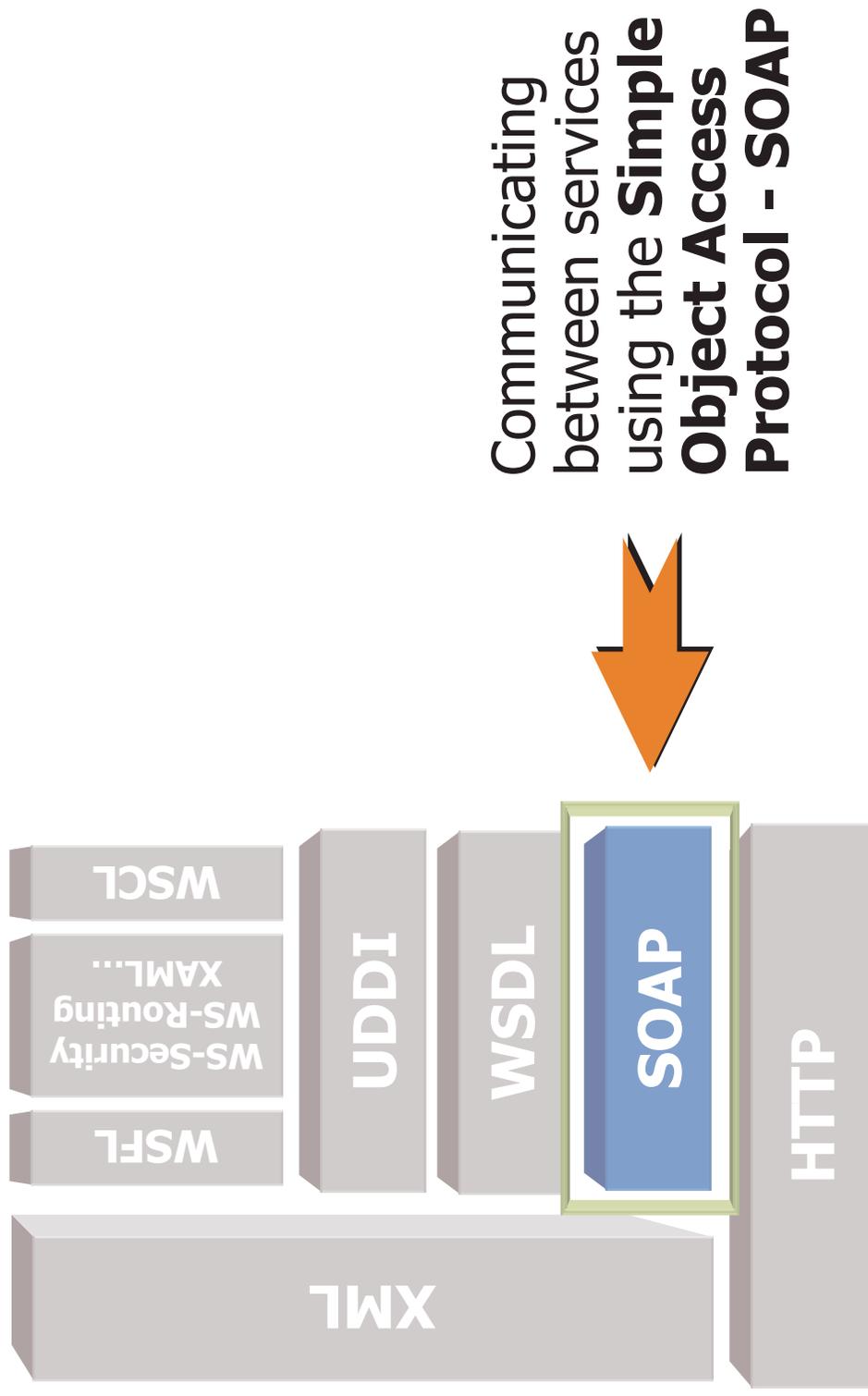


- Take a closer look at a cross-section of standards, typically used to construct and use a Web Service:

- SOAP
- WSDL
- WSFL
- UDDI



# Overview of Web Services Standards





# *SOAP (Simple Object Access Protocol)*

- Web Services communication protocol
- XML extension
- A convention for doing Remote Procedure Calls (RPC):
  - Request (SOAP message)
  - Response (SOAP message)
- Current Status:
  - Developed by Microsoft, DevelopMentor, UserLand, Lotus and IBM
  - SOAP 1.1 is an industry standard
  - SOAP 1.2 is a W3C Working Draft



# *SOAP (Simple Object Access Protocol)*

- XML based web services communication protocol
  - Provides message support for many Web Services standards such as WSDL, UDDI, and Microsoft's .NET architecture
  - Uses GET/POST across http, thus providing a platform & language independent means of communicating
  
- SOAP documents contain:
  - Header
    - optional information about the transaction
  - Body
    - contains payload (e.g. a request or response)
    - may instead contain error/fault information if requests fail



# SOAP Header

- Optional top-level child element of Envelope
- Includes:
  - Fully Qualified element name (must inc. namespace)
  - Optional SOAP encodingStyle attribute
  - Optional SOAP mustUnderstand & actor attribute

```
<soap:Header>  
  <r:req xmlns:r="http://www.softagents.ri.cmu.edu/retsina-soap/"  
    mustUnderstand="1">  
    <r:ont>default</r:ont>  
    <r:speechact>ask</r:speechact >  
  </r:req>  
</soap:Header>
```



## *SOAP Body*

- Contains the payload of the SOAP document
  - Fully Qualified element name (optional namespace)
- May alternately include fault information in case of exceptions.

```
<soap:Body>  
  <wa:GetWeather xmlns:wa="http://www.softagents.ri.cmu.edu/retsina-wa/">  
    <City>Pittsburgh</City>  
    <Date style="US">05/31/2002 </Date>  
  </wa:GetWeather>  
</soap:Body>
```



## *SOAP Body - Faults*

- Carry error or status information within the SOAP body
  - faultcode - (mandatory) qualified name of defined error
  - faultstring - (mandatory) human readable explanation
  - faultactor - identifies who raised the fault exception. Mandatory if recipient is not the ultimate destination of the SOAP document
  - detail - application specific error information

```
<soap:Body>
  <soap:Fault>
    <faultcode>soap:Server</faultcode>
    <faultstring>Server Error</faultstring>
    <detail>
      <wa:myFault xmlns:wa="http://www.softagents.ri.cmu.edu/retsina-wa/">
        <message>Failed to retrieve details from CNN's site</message>
      </detail>
    </soap:Fault>
  </soap:Body>
```

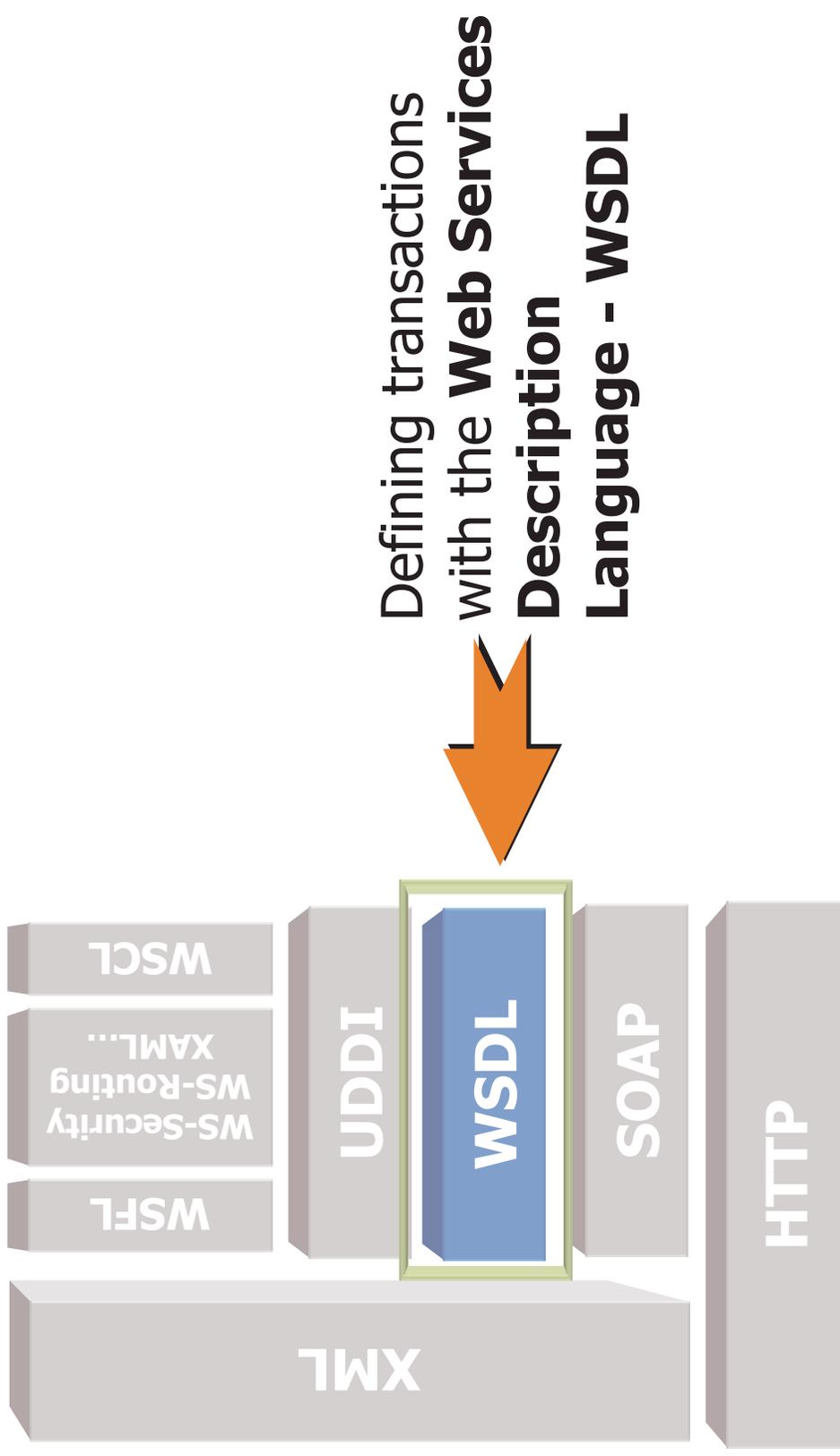


## *Limitations of SOAP*

- Unbounded message format
  - Requires a-priori agreement between Web Services on message format and protocol
  - Provided by higher level standards (e.g. WSDL)
- Has no communicative speech acts
  - No way to determine
    - The intention of the message sender
    - What the message is trying to achieve
- Agent communication languages such as FIPA KQML define speech acts, such as:
  - Basic query performatives (ask-one, ask-all,...)
  - Multi-response query performatives (stream-in,...)
  - Response (reply, sorry,...)
  - ...



# Overview of Web Services Standards





# *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



## *Web Services Description Language*

- Extensible XML based description of web based services and how they are invoked
- Allow for several interfaces using different communication languages
  - E.g. http, SOAP, etc.
- Supports simple transactions (operations)
  - E.g. request-response, solicit-response, etc.



## *WSDL Components*

- **Types** – containers for 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



## *WSDL Components: Types*

- Types enclose data type definitions used in the exchanged messages.
- Definitions typically defined using XSD, but can be extended for other typed languages (e.g. RDF or DAML)



## *WSDL Components: Message*

- Protocol independent message contained within the requester's query and the services response.
  - Corresponds to the payload in SOAP
- Typical transaction consists of two messages, though several messages may be defined for different transactions

```
<message name='Weather.GetTemperature'>  
  <part name='zipcode' type='xsd:string' />  
  <part name='celsius' type='xsd:boolean' />  
</message>  
<message name='Weather.GetTemperatureResponse'>  
  <part name='Result' type='xsd:float' />  
</message>
```



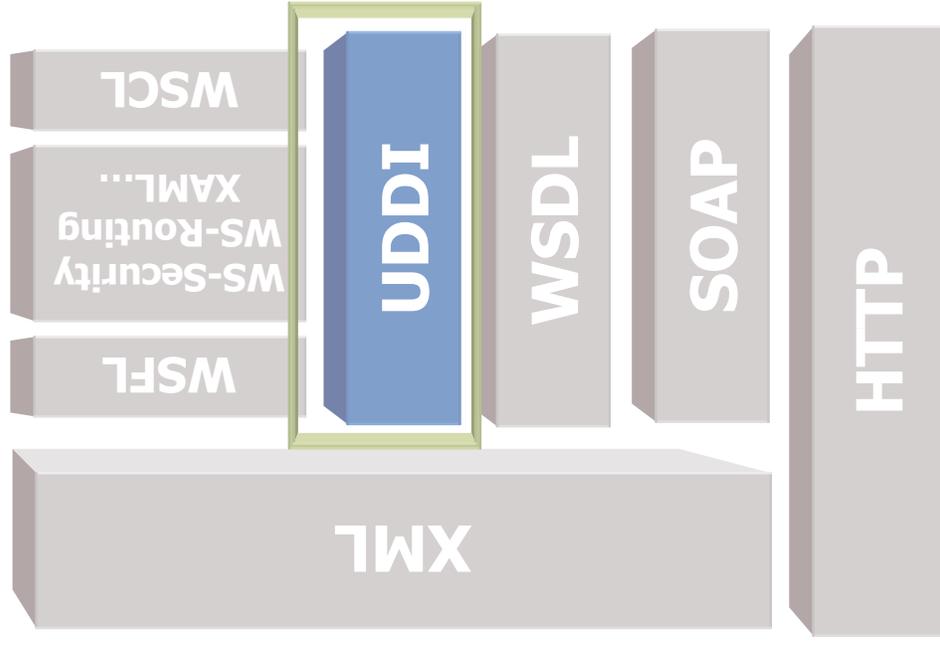
## *WSDL Components: Operation & PortType*

- PortTypes contains a set of abstract operations
- There are four transaction primitives defined by WSDL operations:
  - **One-way.** The endpoint receives a message.
  - **Request-response.** The endpoint receives a message, and sends a correlated message.
  - **Solicit-response.** The endpoint sends a message, and receives a correlated message.
  - **Notification.** The endpoint sends a message.

```
<portType name='WeatherSoapPort'>  
  <operation name='GetTemperature' parameterOrder='zipcode celsius'>  
    <input message='wsdlIns:Weather.GetTemperature' />  
    <output message='wsdlIns:Weather.GetTemperatureResponse' />  
  </operation>  
  <!-- other operations would go here -->  
</portType>
```



# Overview of Web Services Standards



Searching for Services  
using **Universal  
Discovery,  
Description &  
Integration - UDDI**





# *UDDI (Universal Discovery, Description & Integration)*



- Public directory for registering and looking up services
- A directory entry has three main parts:
  - White pages: to describe the company offering the service
  - Yellow pages: to categorize services by industry type (e.g. SIC)
  - Green pages: to describe the interface to a web service
- Uses Type Model or tModel documents
- Current Status:
  - Industry initiative led by Microsoft, IBM and Ariba; more than 300 companies participating
  - UDDI specification will be submitted to a standards group once Version 3 is completed



## ***UDDI (Universal Discovery, Description & Integration)***



- Yellow Pages Directory Service for Web Services
- Keyword searches based on standard taxonomies
  - NAICS (North American Industrial Classification System)
  - SIC (Standard Industrial Classification)
- White Pages lookup for
  - Service Providers, contact details, etc.
- Service types are registered as a unique tModel
- API to UDDI servers communicate using SOAP

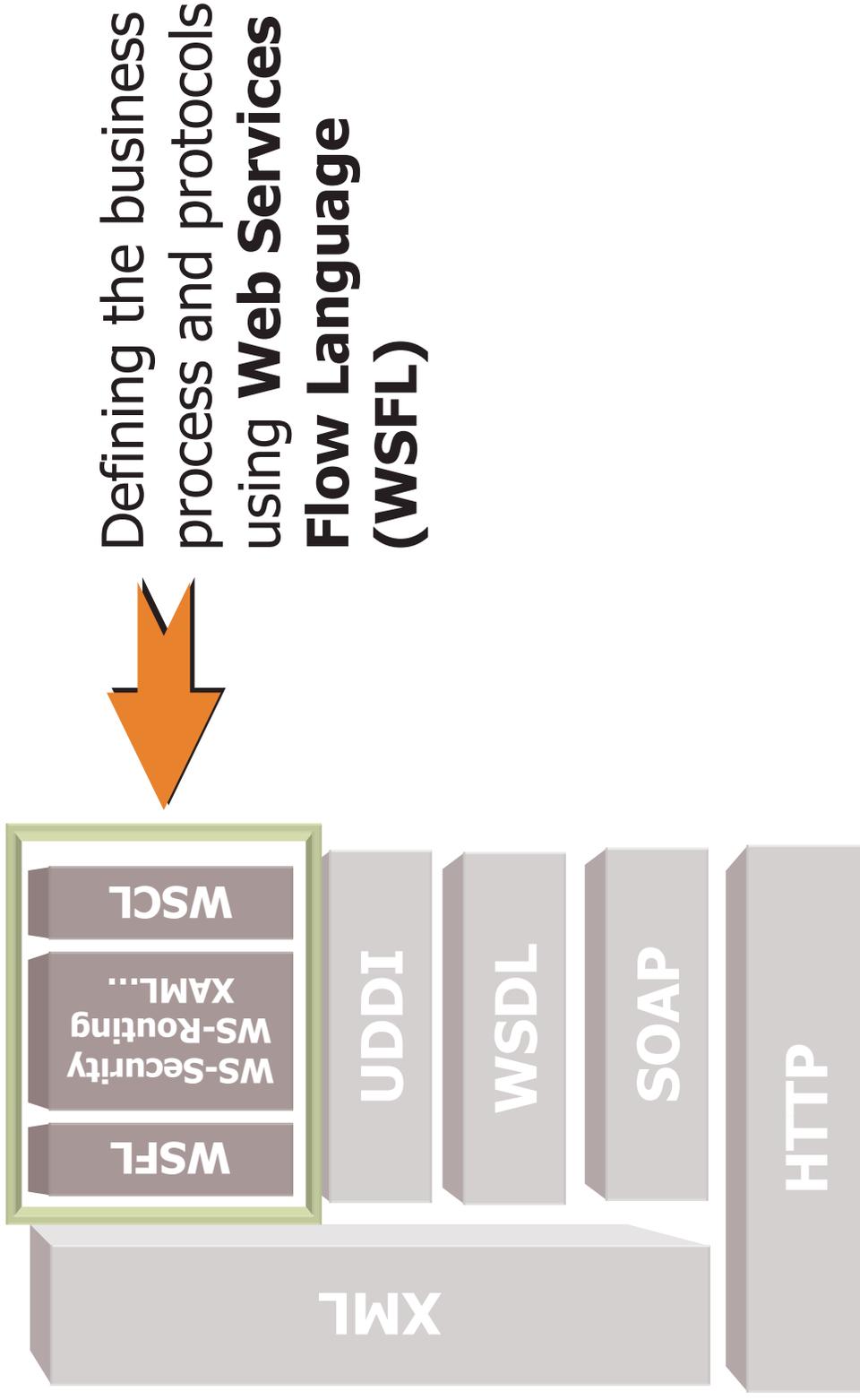


# UDDI Search Capabilities

- Four methods for searching available through API:
  - ✚ Find\_business
    - Locate information about one or more businesses.
  - ✚ Find\_binding
    - Locate specific bindings within a registered business.
  - ✚ Find\_service
    - Locate specific service within a registered Business Entity.
  - ✚ Find\_tModel
    - Locate one or more tModel Information structures.
- Arguments to searches are keyword based
  - ✚ Uses keywords to guide search:
    - *find business named IB\**
  - ✚ Use tModels to find services with a feature:
    - *find all services with WSDL specification*



# Overview of Web Services Standards





## *WSFL (Web Services Flow Language)*

- Structured description of two categories of Web Service compositions:
  - *Usage Patterns*
    - given a collection of Web Services, how to achieve a particular goal
  - *Interaction Patterns*
    - given a collection of Web Services, how several partners interact
  - Flow Models - describes the flow of control or data
  - Messages between services are specified in WSDL
- XML based
- Current Status:
  - Developed by IBM Software Group
  - Version 1.0 submitted as a W3C Note



# *Web Services Flow Language*

- Description of how Web Services are composed
  - ▣ Flow Model describes the structure of the business process in terms of:
    - WSFL Activities
      - Describe the process steps
    - WSFL Data and Control Links
      - Represent sequencing rules and information flow
  - ▣ WSFL Global Model
    - Describes interaction between provider and requester
    - Mappings between internal operations and WSDL port types
    - WSFL Plug links map between control flow and WSDL operations



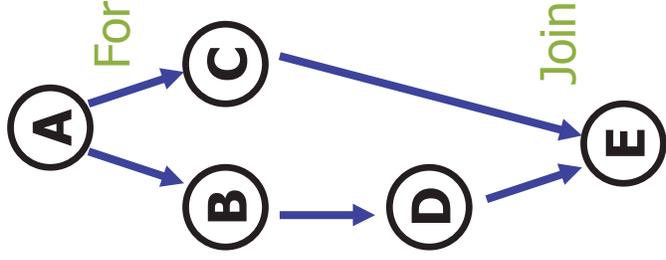
# *WSFL Service Composition Meta Model*

- **Activities**
  - Represents a business task to be performed as a single step within the overall business process
  - Has a signature relating to the operation used to implement it (i.e. input/output/fault messages etc.)
- **Control Links**
  - Directed edges within a graph that link *Activities*
- **Transition Conditions**
  - Determines whether or not *Activities* should be performed
  - Boolean expression associated with a *Control Link*



# WSFL Flow Meta Model

- Flow can be represented as a directed graph
- *Activities* can be:
  - ▣ Fork Activities (and parallelism)
    - When this completes, all control links leaving this activity will be determined and associated transition conditions evaluated (at runtime).
    - If evaluation is true, the associated *target activities* will be performed next.
  - ▣ Join Activities (and synchronization)
    - Join Activities have more than one incoming control link.
    - Decision to perform activity deferred until all incoming paths have been evaluated, based on a Boolean *Join Condition*



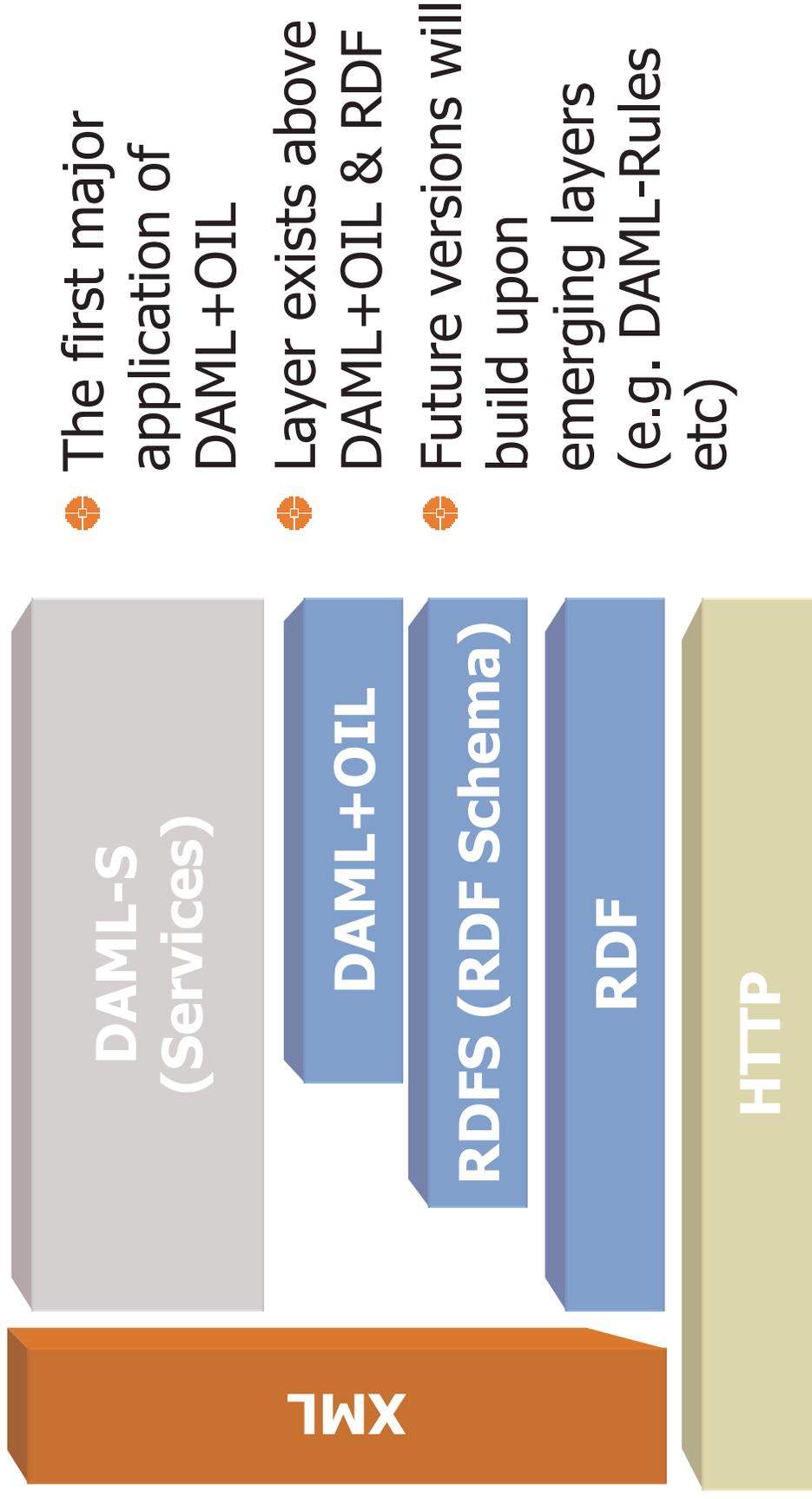


## *Outline*

1. What are Web Services?
2. Mediation and Composition
3. Industry Standards
- **4. Semantic Web efforts**
5. Future Directions
6. Discussion



## *Layered Approach to Language Development*





# RDF & DAML

- RDF (Resource Description Framework) built using
  - XML syntax
  - qualified Universal Resource Identifiers (URI)
  - Namespaces etc.
- Models Meta-Data about resources on the Web
  - Uses *subject/predicate/object* triples, which form a directed graph
- DAML+OIL extends RDF statements to provide a rich descriptive logic language
  - Supports extensible, distributed ontologies

***DAML+OIL will be succeeded by the emerging  
Web Ontology language OWL***

<http://www.w3.org/2001/sw/WebOnt/>



# RDF Concepts

## • Three fundamental concepts:

### ▣ Resources

- All entities described by RDF expressions, named by a qualified URI:

rdf:resource="http://www.cs.cmu.edu/~terryp/index.html"

rdf:resource="http://www.tac.org/2001event.rdf#PainInNEC"

### ▣ Properties

- Define characteristic of a resource

```
<rdf:Description about="http://www.w3.org">
```

```
  <dc:title>W3C Home Page</s:title>
```

```
</rdf:Description>
```

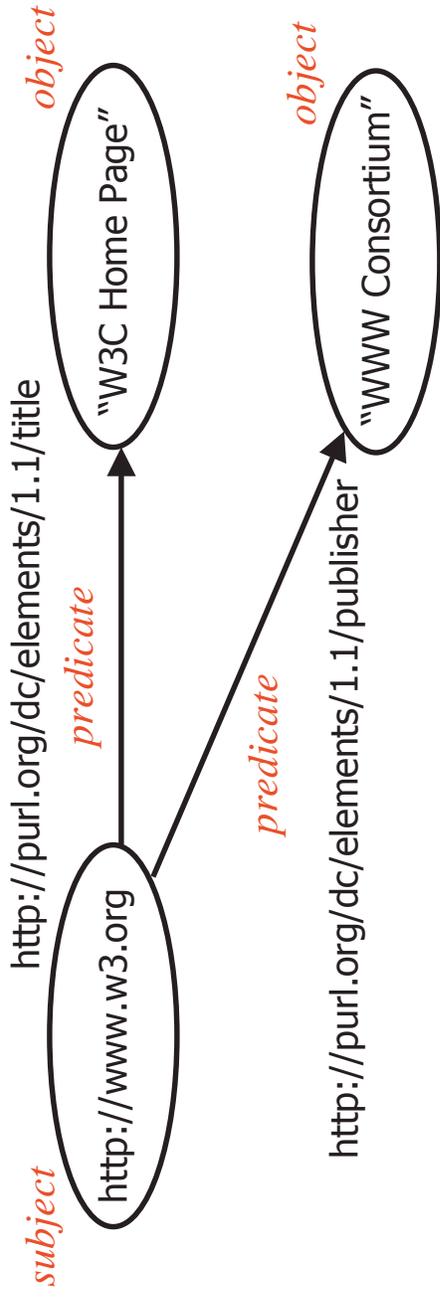
### ▣ Statements

- Resources that reify *subject/predicate/object* triples



# RDF Graphs

- The *subject/predicate/object* triples found in an RDF document form a graph:



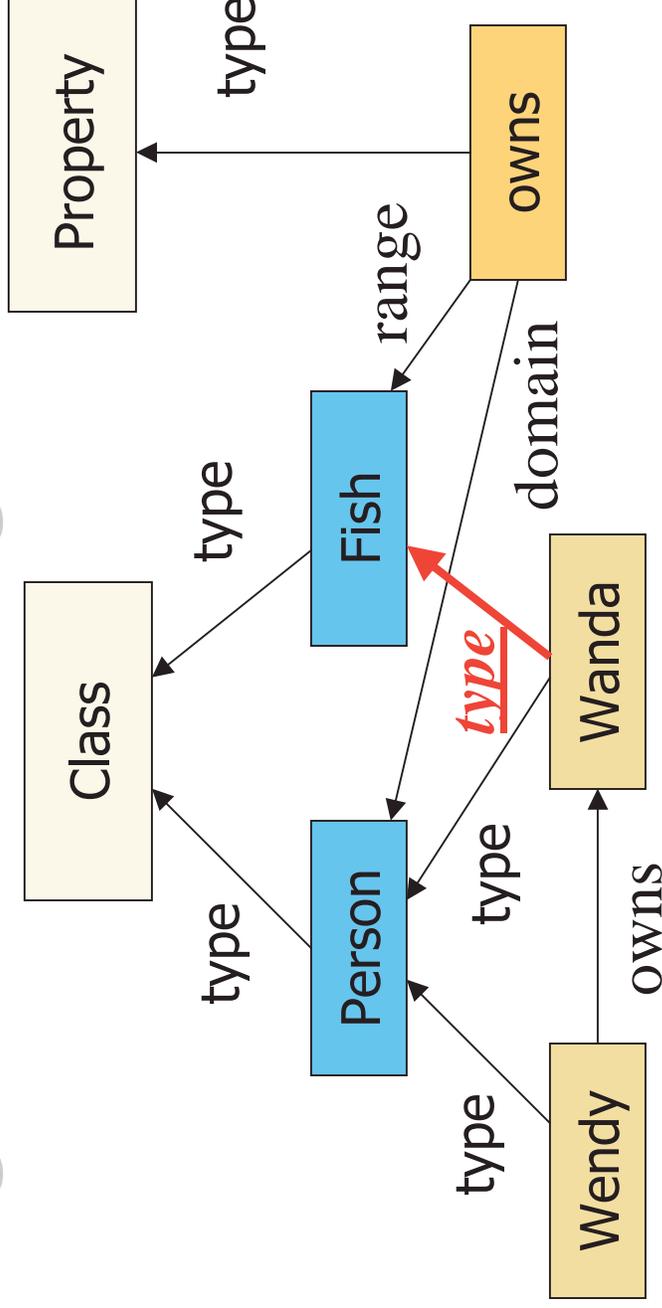


# *RDF Schema*

- RDF Schema defines new RDF vocabulary, extending beyond definitions of triples. For example:
  - ▣ `rdfs:Class`
    - resources denoting a set of resources, by means of the property `rdfs:type`
  - ▣ `rdfs:subClassOf`
    - Used to define class hierarchies.
  - ▣ `rdfs:domain` & `rdfs:range`
    - Define restrictions on the resources that have a given property (`domain`) and the set of valid values for that property (`range`)



# Ontological Reasoning in RDF



Type constraint violation: The range of owns is Fish.

**OR** There is no inconsistency: Wanda is a fish! Mermaid?



# *DAML: DARPA Agent Markup Language*

- Provides constructs for creating extendable ontologies and markup within an open, dynamic system (i.e. the Internet).
- Description Logic Extensions to RDF - The Resource Description Framework
- Latest Version (DAML+OIL) is available at:

 <http://www.daml.org/language>





## *From RDF to DAML+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

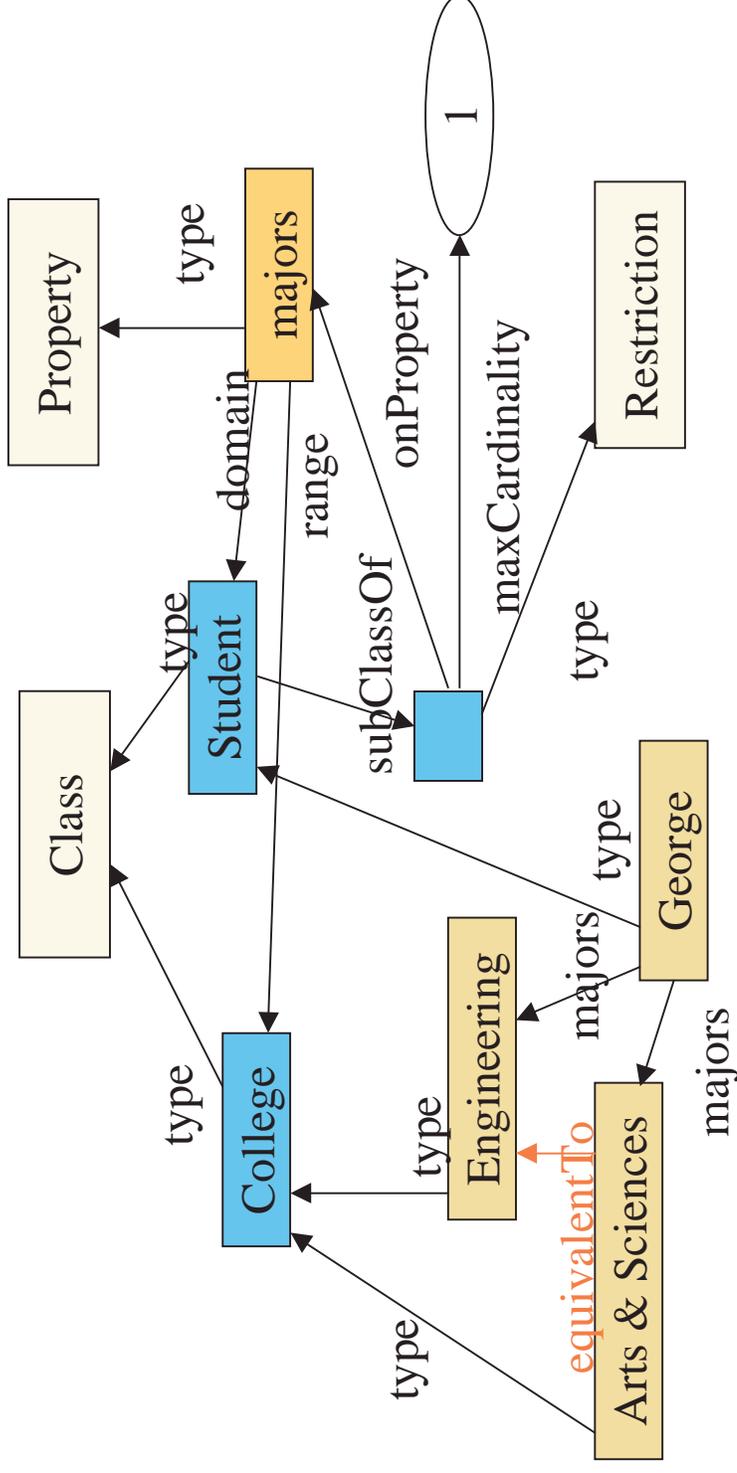
***DAML+OIL will be succeeded by the emerging Web  
Ontology language OWL***

<http://www.w3.org/2001/sw/WebOnt/>



# Ontological Reasoning in DAML

- Cardinality constraint violation: George can't have two majors
- OR There is no inconsistency: Engineering = Arts & Sciences





# DAML-S

- **DAML-S: A DARPA Agent Markup Language for Services**
  - An upper ontology for describing **properties & capabilities of agents & (Web) services** in an unambiguous, computer interpretable markup language.
  - DAML+OIL Ontology for (Web) services
- **AI-inspired markup language:**
  - tailored to the representational needs of Services
  - expressive power
  - well-defined semantics
  - ontologies support reuse, mapping, succinct markup, ...

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



## *Acknowledgements to the DAML-S Web Services Coalition*

**CMU:** Anupriya, Ankolekar, Massimo Paolucci,  
Terry Payne, Katia Sycara

**BBN:** Mark Burstein

**Nokia:** Ora Lassila

**Stanford KSL:** Sheila McIlraith, Honglei Zeng

**SRI:** Jerry Hobbs, David Martin, Srini Narayanan

**[Yale:** Drew McDermott & Manchester: Ian Horrocks]  
*Several of these slides courtesy of Sheila McIlraith,  
Stanford KSL, & David Martin, SRI*



# *DAML-S Objectives*

- Provide:
  - an upper ontology for describing **properties & capabilities of agents & (Web) services** in an unambiguous, computer interpretable markup language.
  
- Desiderata:
  - an ontology of Web services
  - ease of expressiveness
  - enables automation of service use by agents
  - enables reasoning about service properties and capabilities

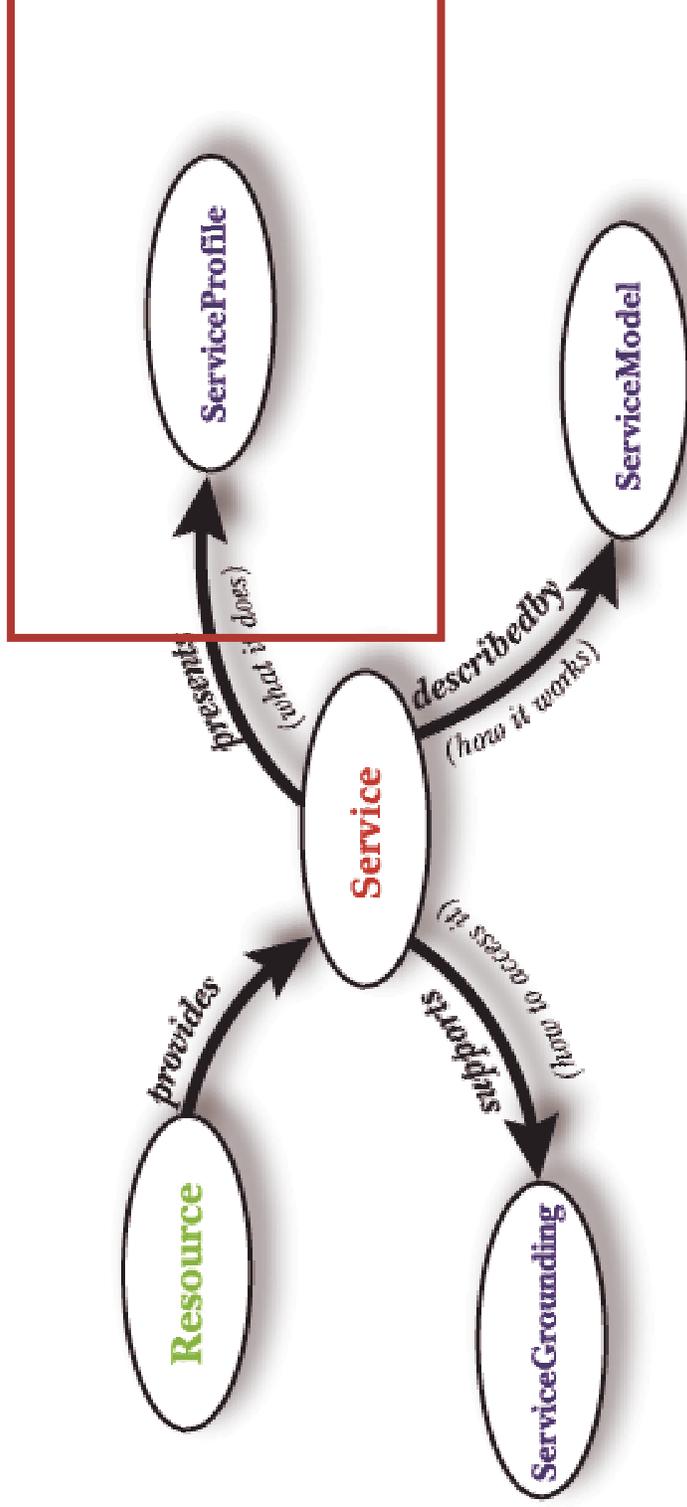


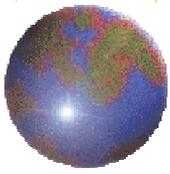
## *Automation enabled by DAML-S*

- Web Service Discovery & Selection
  - Find me an airline that can fly me to Bologna
- Web Service Invocation
  - Book flight tickets from Alitalia to arrive on June 14<sup>th</sup>
- Web Service Composition & Interoperation
  - Arrange travel & hotel in Bologna.
- Web Service Execution Monitoring
  - Has the hotel room been reserved ?

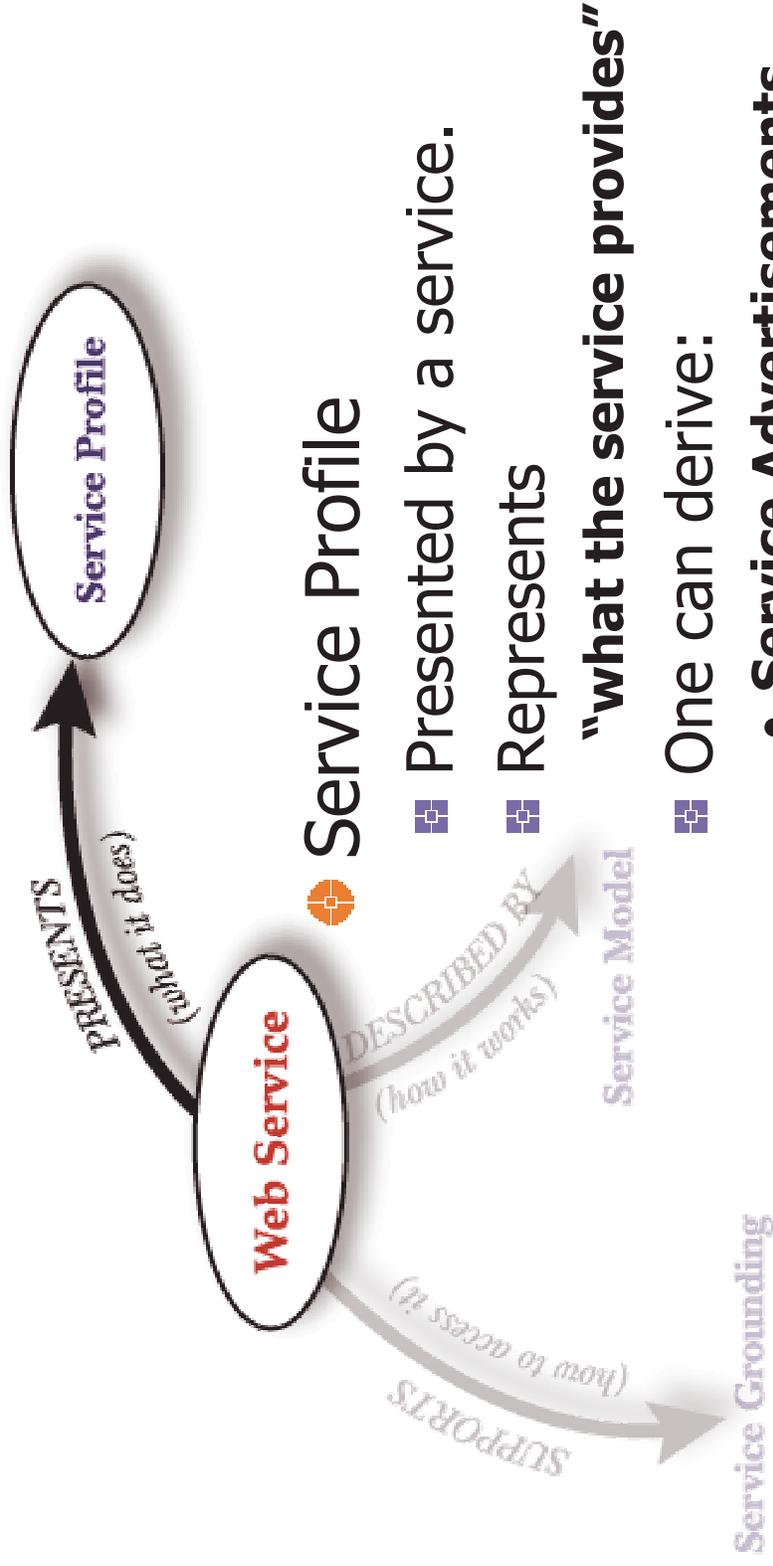


# DAML-S Service Models





# Presenting Service Profiles





## *DAML-S Service Profile (Overview)*

- High-level description of a service and its provider
  - description of service (human readable)
  - specification of functionalities service provides
  - functional attributes (requirements and capabilities)
- Profile used for
  - populating service registries
  - automated service discovery
  - Matching of capability advertisements to requests



# DAML-S Service Profile

## Functionality Description

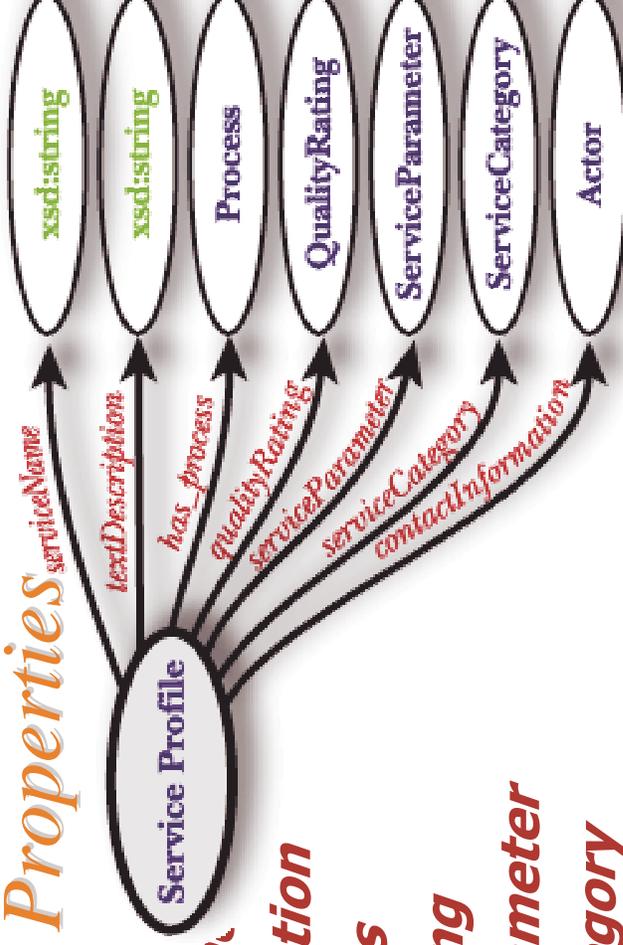
- **Preconditions**
  - Set of conditions that should hold prior to the service being invoked
- **Inputs**
  - Set of necessary inputs that the requester should provide to invoke the service
- **Outputs**
  - Results that the requester should expect after interaction with the service provider is completed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



# DAML-S Service Profile

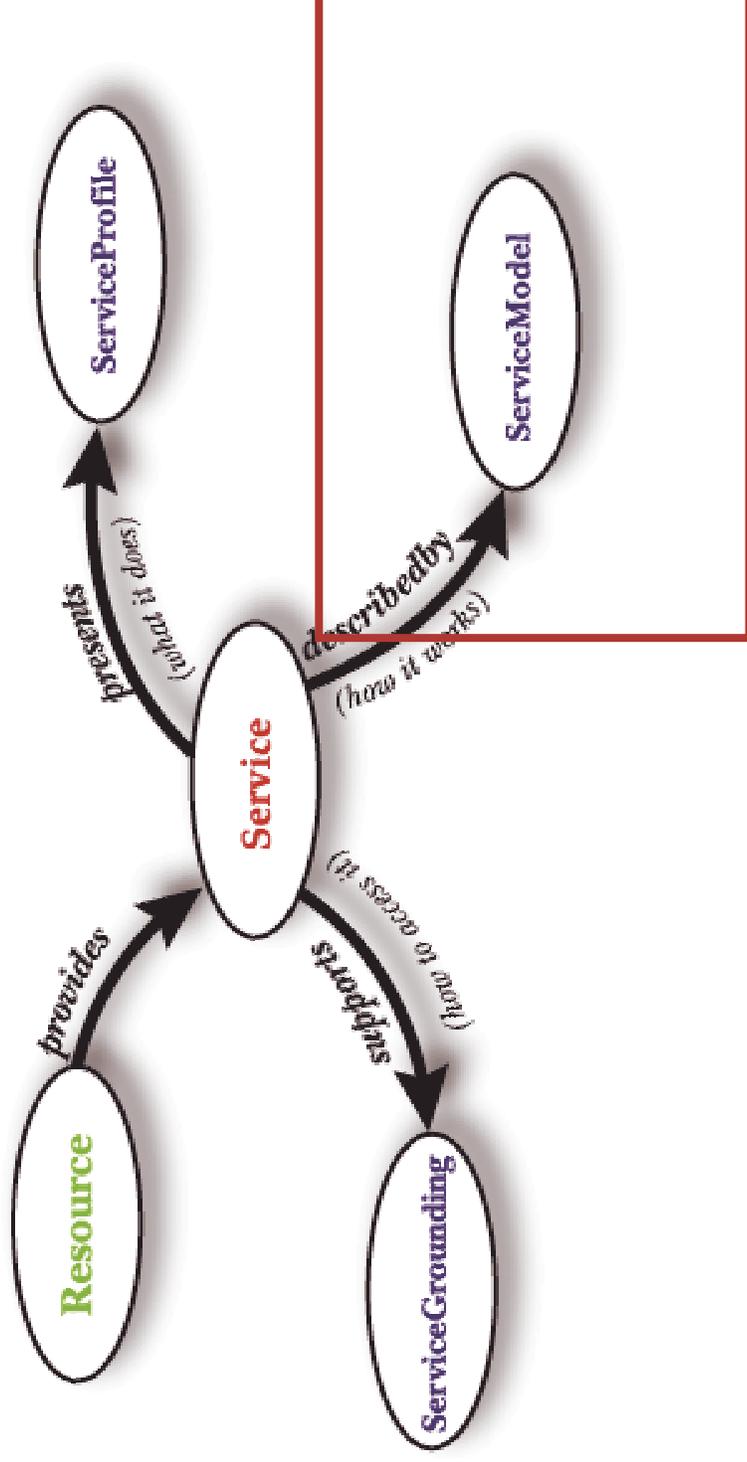
## Non Functional Properties

- These include
  - ▣ **serviceName**
  - ▣ **textDescription**
  - ▣ **has\_process**
  - ▣ **qualityRating**
  - ▣ **serviceParameter**
  - ▣ **serviceCategory**
  - ▣ **contactInformation**



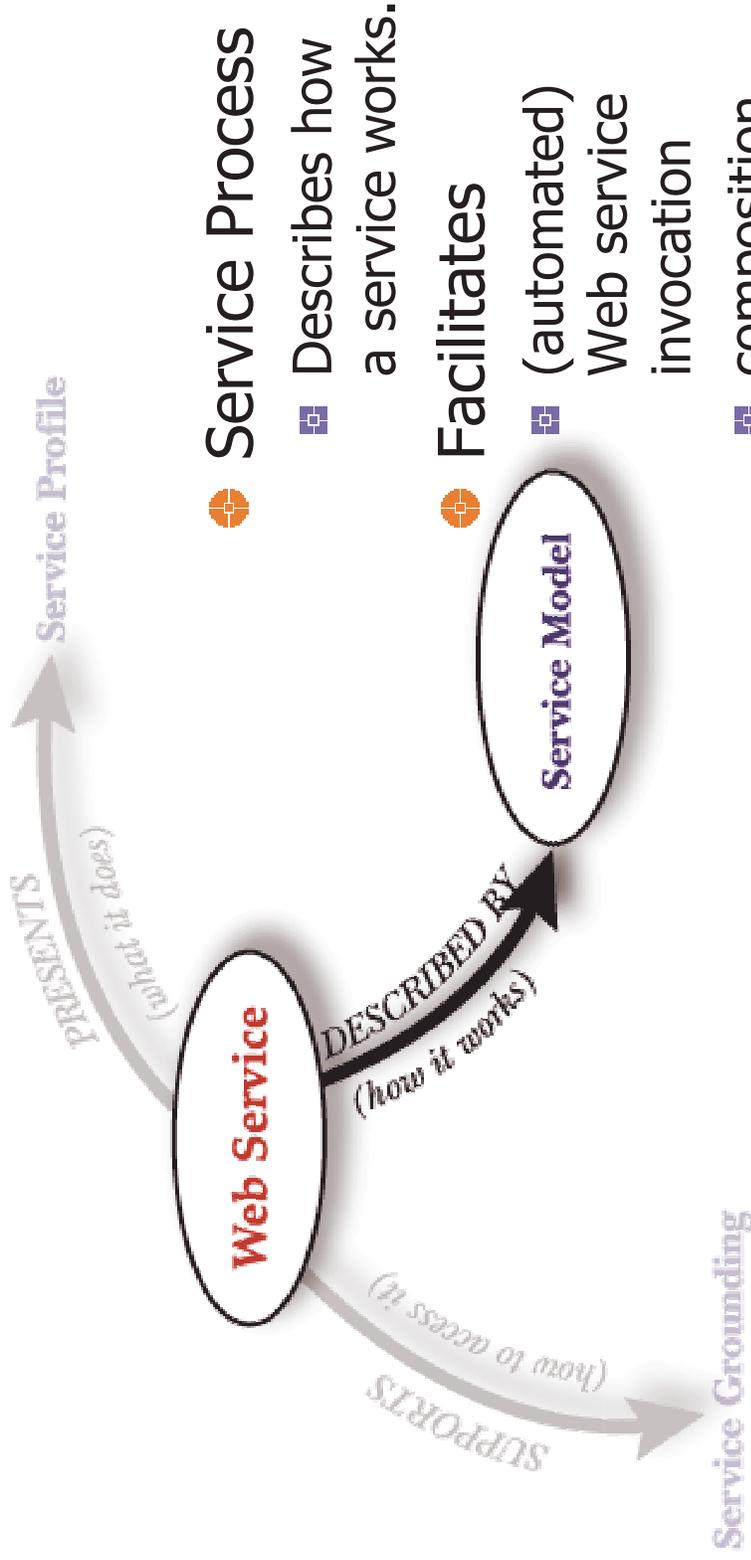


# DAML-S Service Models





# Describing Service Models



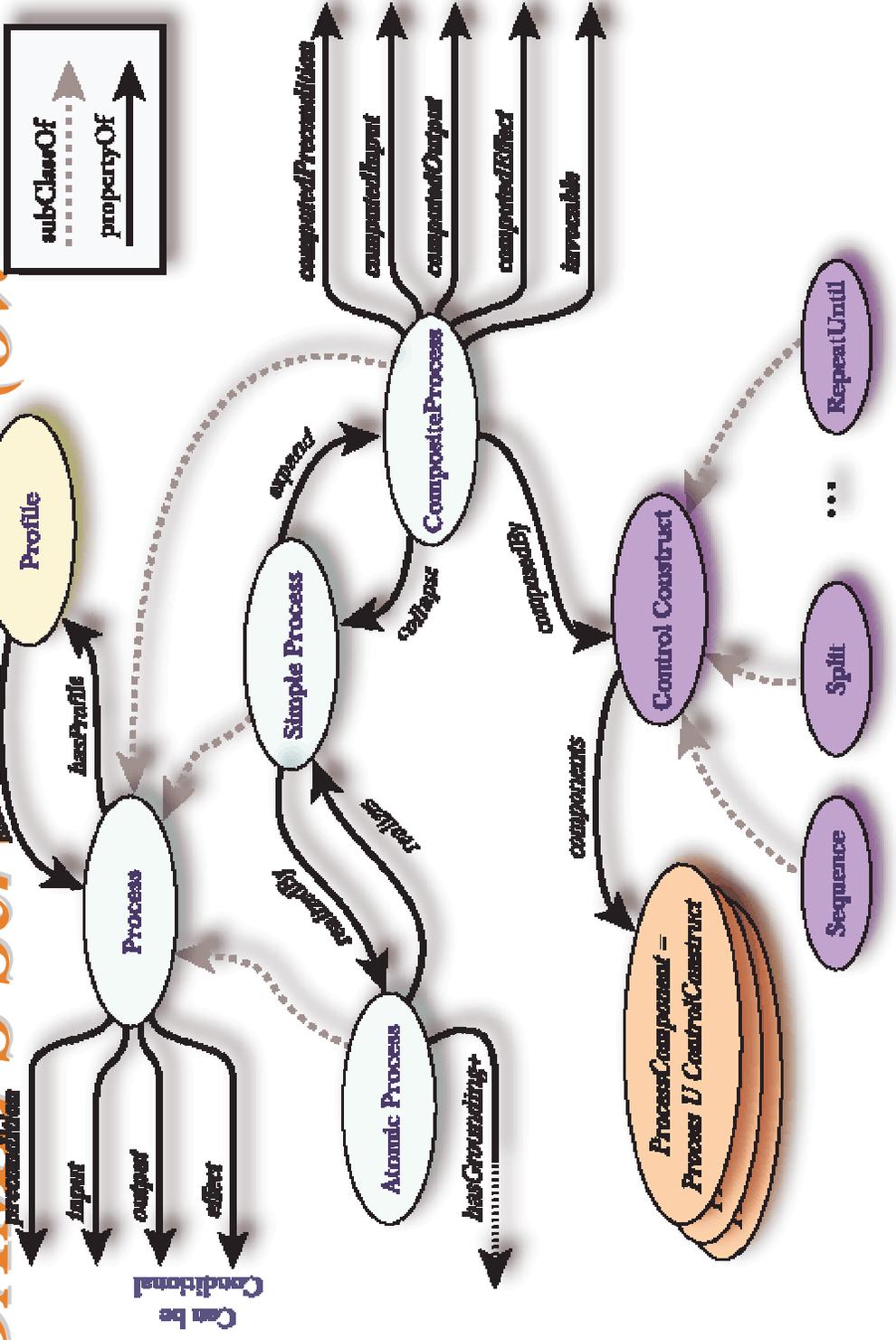
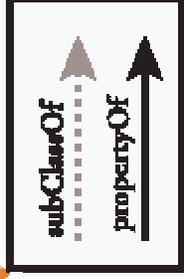


## *DAML-S Service Model (Overview)*

- Service Model allows a service requester
  - 1) to determine whether the service meets its needs;
  - 2) to compose service descriptions from multiple services to perform a specific task;
  - 3) during the course of the service invocation, to coordinate the activities of the different participants;
  - 4) to form expectations as to the execution of the service.



# DAML-S Service Model (Overview)





## *Anatomy of a DAML-S Service Model*

- Processes are conceived as:
  - Atomic
  - Simple
  - Composite
- Associated with each service is a set of:
  - Inputs
  - Outputs
  - Preconditions
  - Effects
- Invocable processes have an associated grounding:
  - Includes WSDL description to model:
    - Operation
    - Message formats
    - Ports & Bindings



## *Anatomy of a DAML-S Service Model*

- Composite processes are compositions of simple or other composite processes in terms of constructs:
  - Sequence
  - if-then-else
  - Fork
  - Etc.
- Data flow and Control flow should be described for each composite service
- A black box and glass box view may be given of each composite service

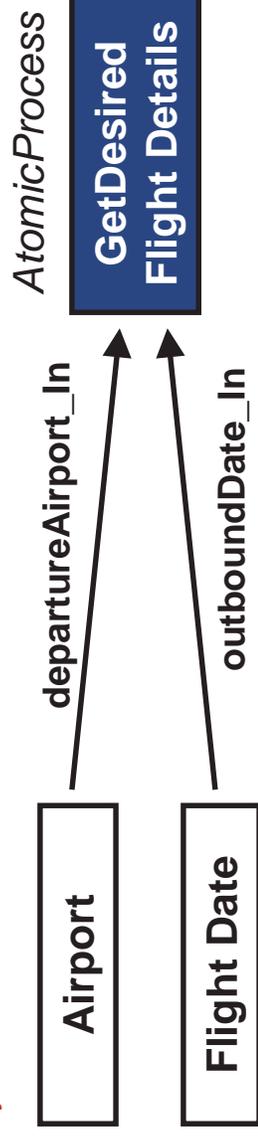


# Atomic Process Example

```
<rdf:Class rdf:ID="GetDesiredFlightDetails">
  <rdf:subClassOf rdf:resource="http://www.daml.org/Process#AtomicProcess" />
</rdf:Class>

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

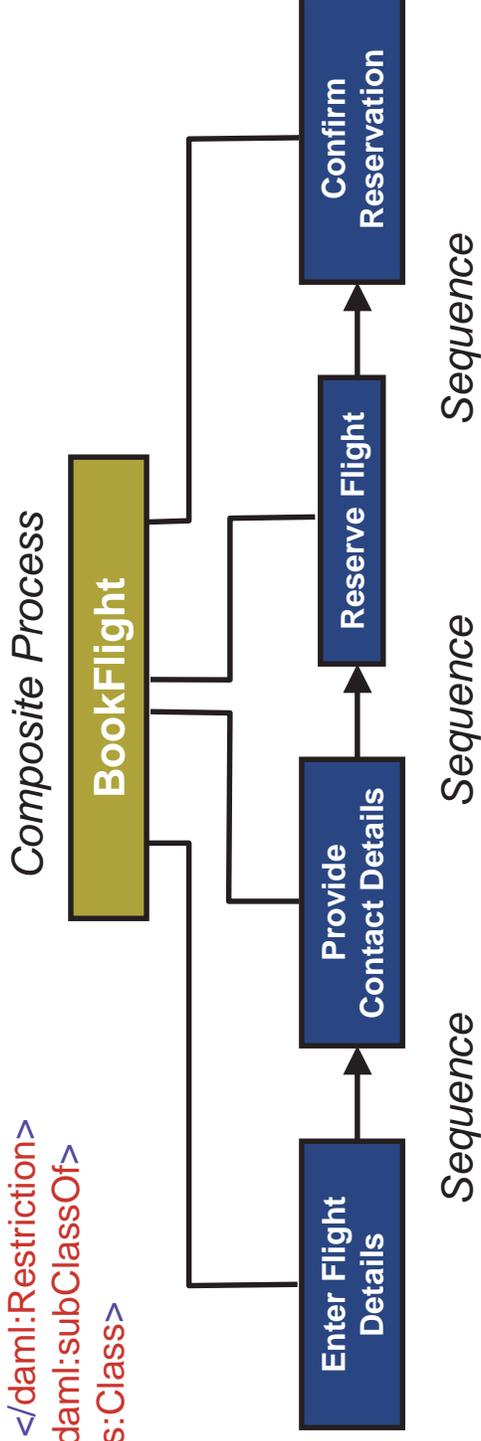
<rdf:Property rdf:ID="outboundDate_In">
  <rdf:subPropertyOf rdf:resource="http://www.daml.org/Process#input" />
  <rdf:domain rdf:resource="#GetDesiredFlightDetails" />
  <rdf:range rdf:resource="http://www.daml.ri.cmu.edu/ont/
    DAML-S/concepts.daml#FlightDate" />
</rdf:Property>
```

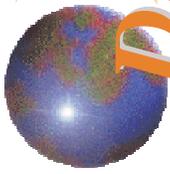




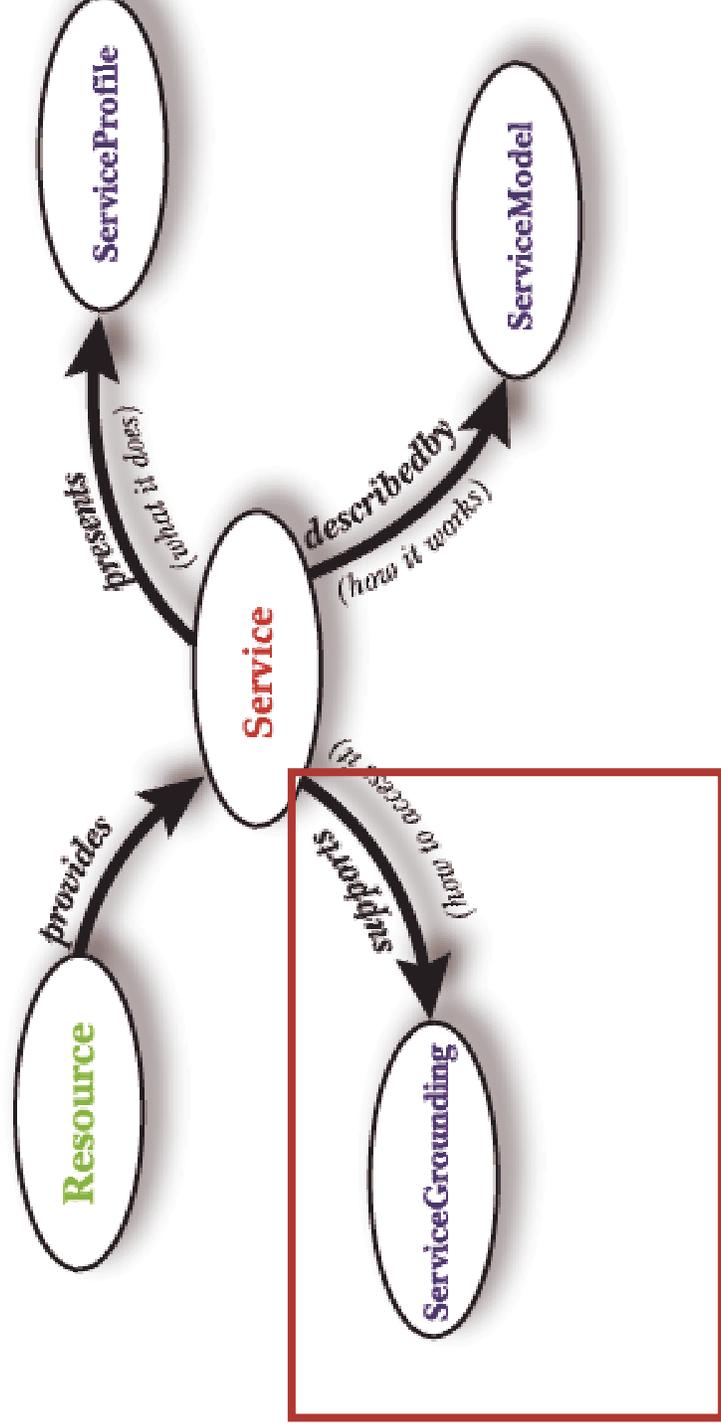
# Composite Process Example

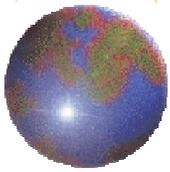
```
<rdfs:Class rdf:ID="BookFlight">
  <rdfs:subClassOf rdf:resource="#CompositeProcess" />
  <rdfs:subClassOf rdf:resource="http://www.daml.org/Process#Sequence" />
  <daml:subClassOf>
    <daml:Restriction>
      <daml:onProperty
        rdf:resource="http://www.daml.org/Process#components" />
   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



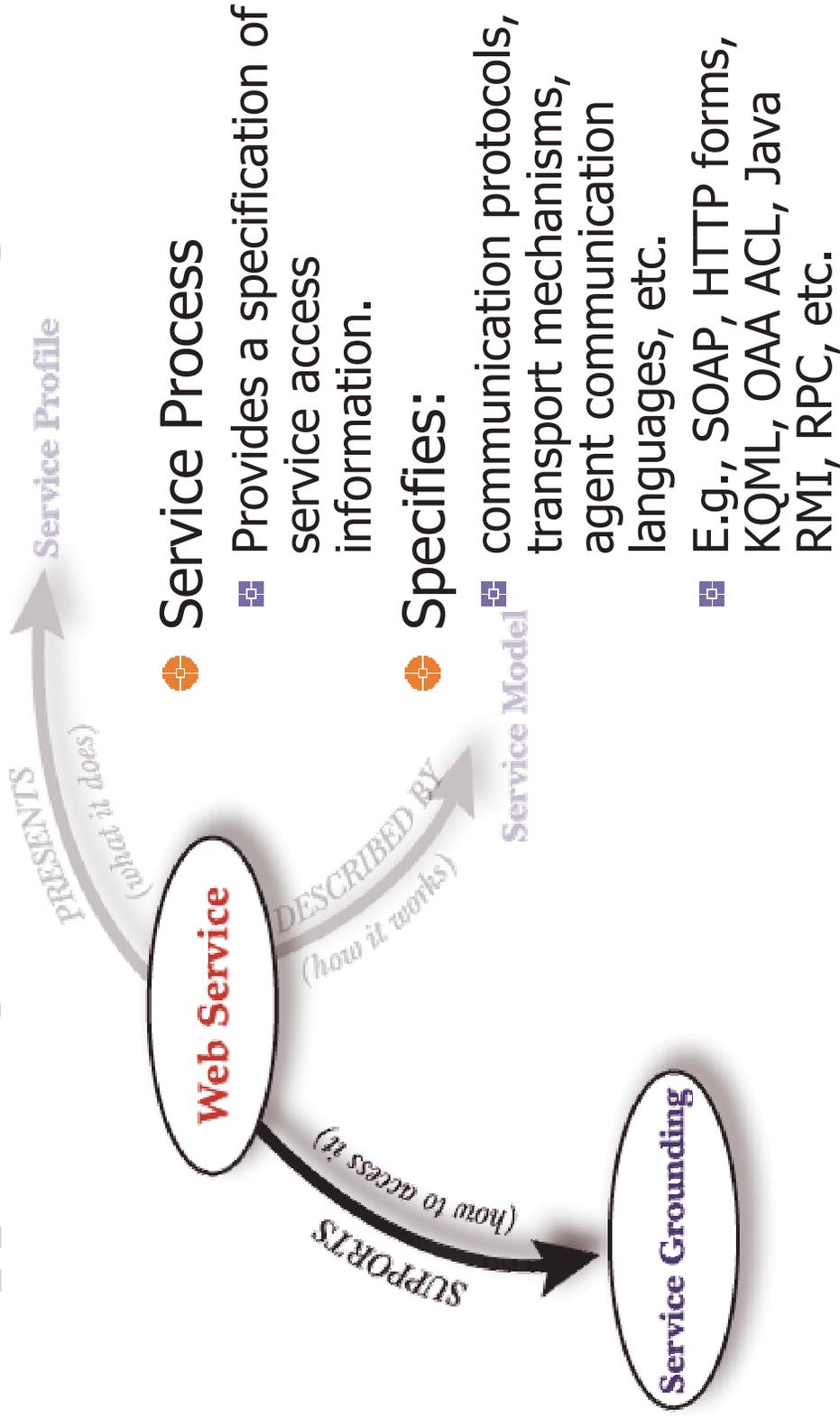


# DAML-S Service Models





# Supporting a Service Grounding



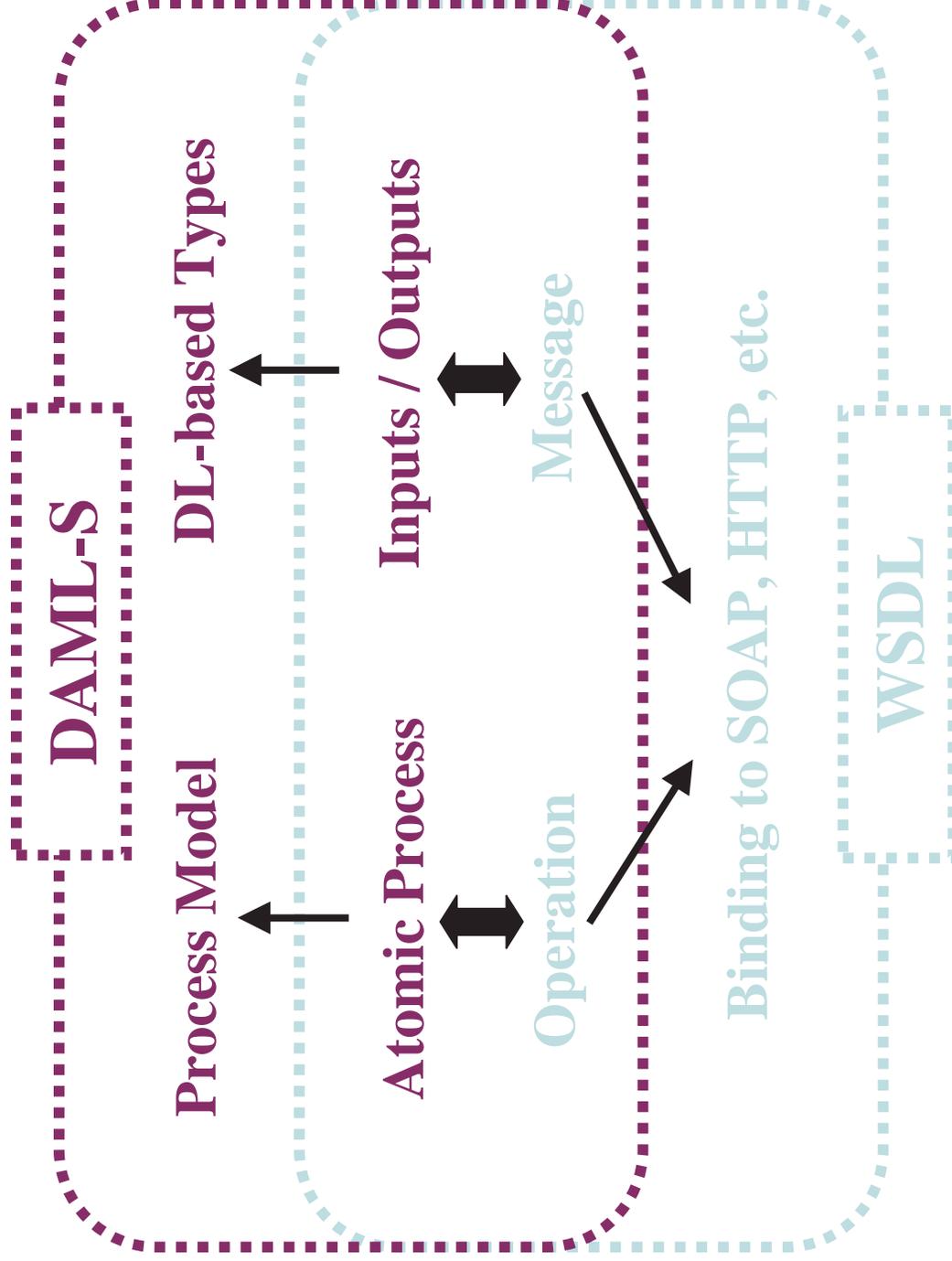


## *Service Grounding: “How to access it”*

- *How to access the service*
- Implementation-specific
- Message formatting, transport mechanisms, protocols, serializations of types
- Service Model + Grounding give everything needed for using the service



# DAML-S / WSDL Binding





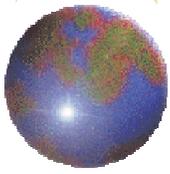
## *Supporting Infrastructure*

- DAML-S Matchmaker
  - Service Registry for DAML-S profiles
  - Subsumption Reasoning Matching Engine
  - Augments a UDDI Server
  
- DAML-S API
  - APIs are being developed to support generation of requests and reasoning about matching profiles.

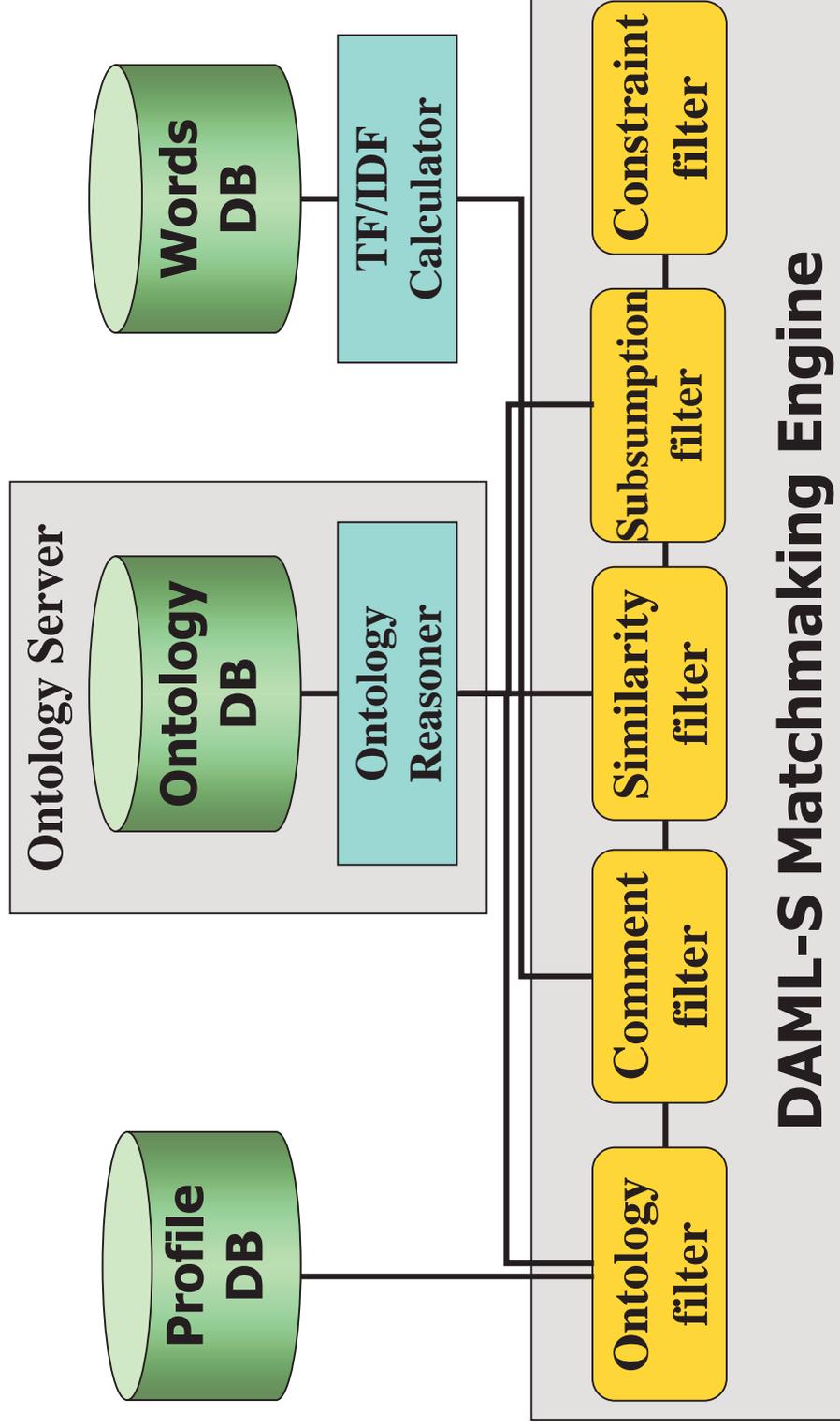


## *DAML-S Matchmaker*

- Filter-Based Semantic Matching Engine for DAML-S profiles based on heuristic filters
  - Logical inference on the DAML Ontologies guarantee correctness of matching
  - Information Retrieval techniques that help to speed up the matching
- Extends an existing UDDI server
  - Enable capability descriptions and matching of DAML enabled services registered with UDDI



# *DAML-S Matchmaker Processing Module*

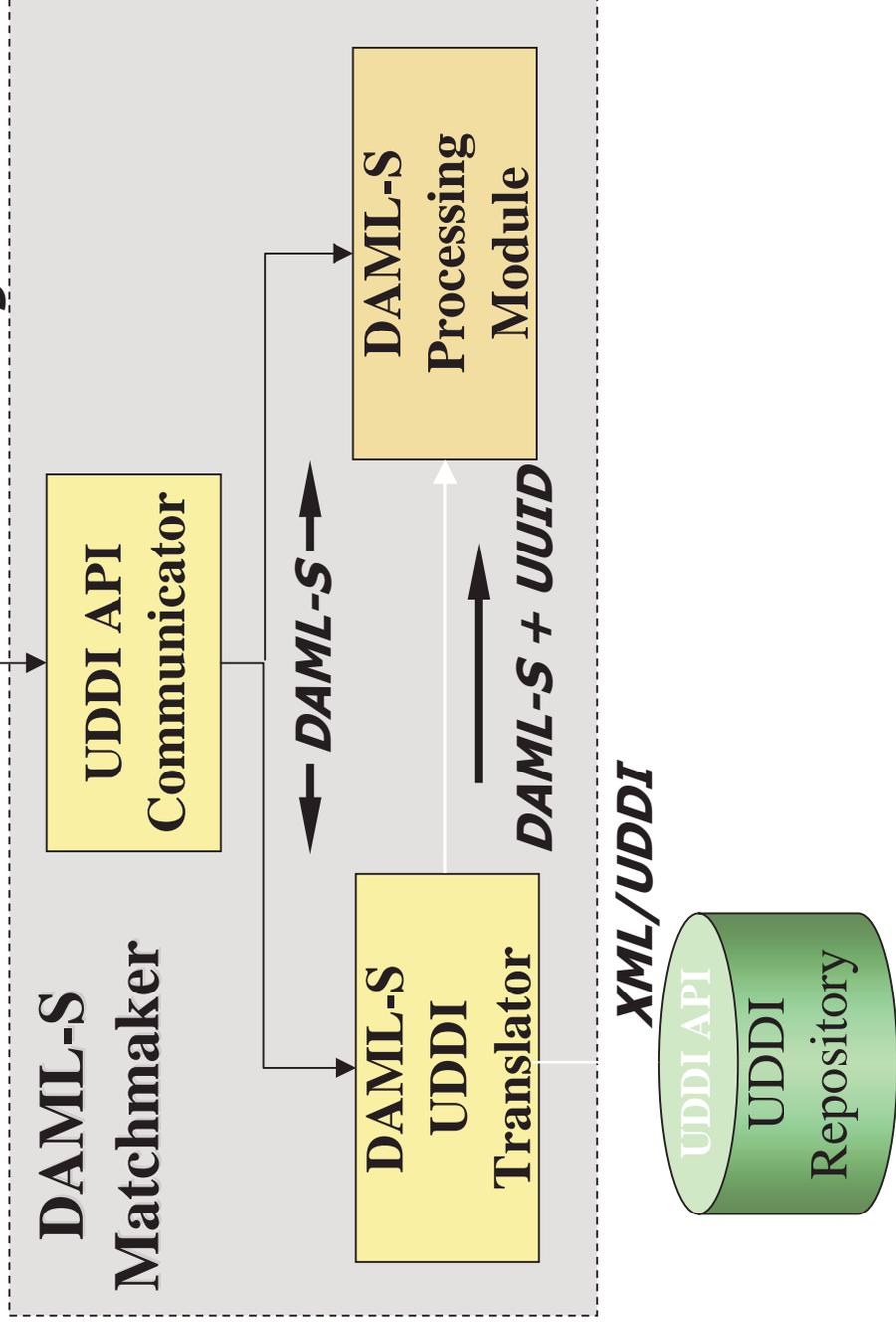


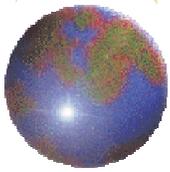


# Extending UDDI

Web Services

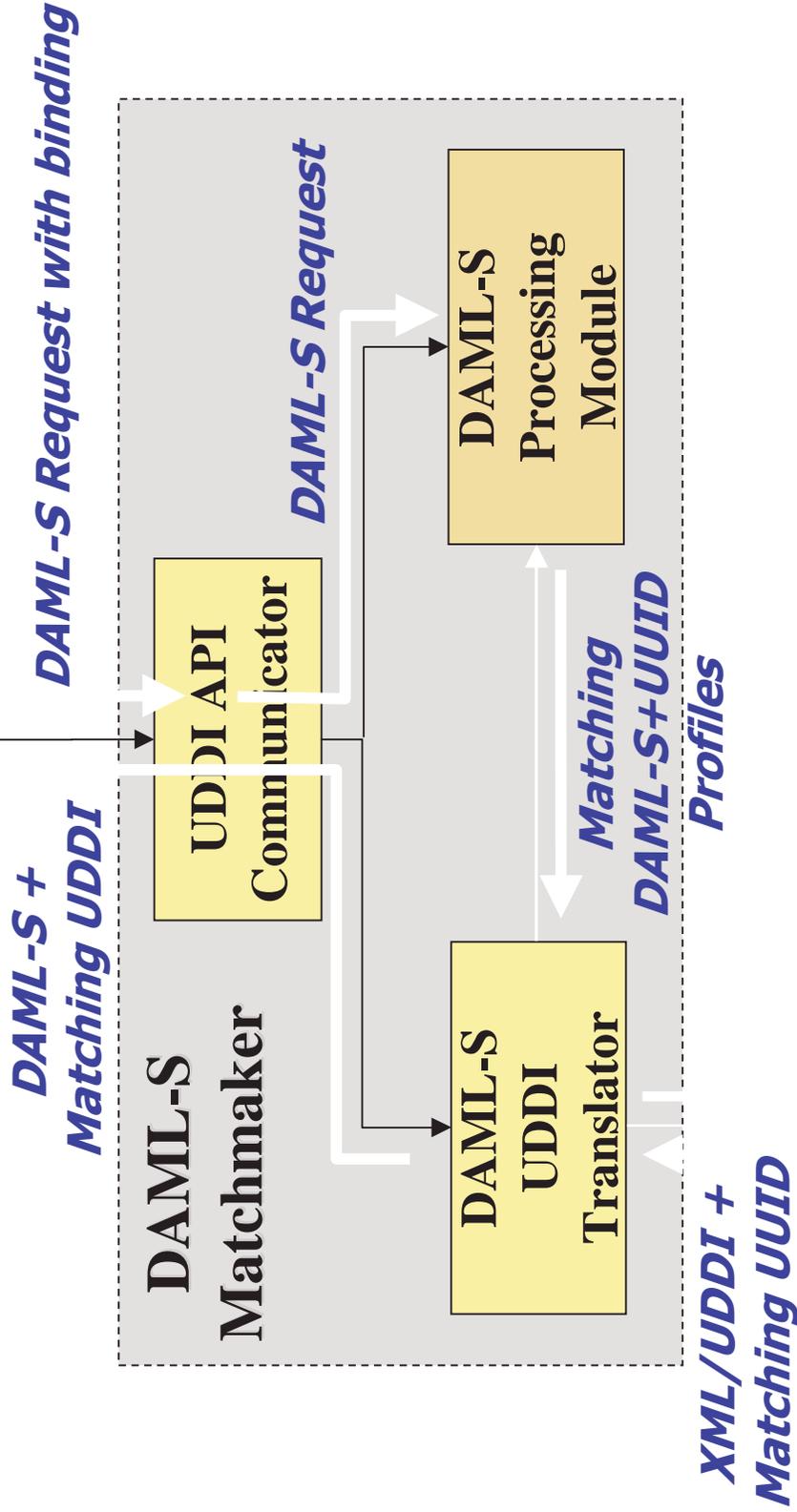
*DAML-S & KQML Binding*  
*DAML-S & SOAP Binding*





# DAML-S Request Data Flow

Web Services





## *Contrasting DAML-S with UDDI*

### UDDI

- Goal: **integration** and **semi-automation** of B2B transactions
- Provides registry of business and their services in XML
- Matching of advertisements and requests is based on **keywords**
- **Interaction** with services is done through creating **customized programs**



## Contrasting DAML-S with UDDI (cont)

- Each business description consists of:
  - ▣ *businessEntity*: describes businesses by name, key, categorization, services offered and contact information
  - ▣ *businessService* describes services by name, key, categorization and multiple bindingTemplate elements
  - ▣ *bindingTemplate* describes the service access (phone, FAX, http, ftp etc), key, TModelInstances
  - ▣ *TModelInstances* describe the protocols and interchange formats the service comprehends
- All the above elements are described through natural language text, and hence not amenable to machine comprehension.



## *Contrasting DAML-S with UDDI (cont)*

- UDDI does not specify a content language for advertisements, but candidates are WSDL or XML/edi
- DAML-S could be a language candidate to be used on top of UDDI
- Though agents can search UDDI registries, humans must be involved to interpret the descriptions and program the access interface
- UDDI does not support Process Model description, hence no support for automatic composition of services



## *Contrasting DAML-S with WSDL*

- Language to describe interface to business services in UDDI registries
- Services are defined as sets of ports
  - i.e. network addresses with associated protocols and data formats
- Messages and operations are mapped to ports
- A message is defined abstractly as a request, response or a parameter of a request or response
- Service preconditions or effects cannot be expressed



## ***Contrasting DAML-S with E-speak***

- E-speak and UDDI have similar goals to facilitate advertisement and discovery of services
  - HP is collaborating with the UDDI consortium
- E-speak describes services as sets of attributes common to a logical group of services
- Matches of lookup requests are done through attribute matching
  - Value types of attributes are String, Int, Boolean, and Double
- Basic vocabulary defining attributes as Name, Type (for string only), Description, Keyword and Version



## *Contrasting DAML-S with E-speak (cont)*

- No semantic meaning attached to any of the attributes
- No (sub)typing has been specified
- Dependencies among attributes and logical constraints are not expressible
- E-speak requires an e-speak engine to be run on all client machines
- E-speak does not support service process model so no execution monitoring is possible



## *Contrasting DAML-S with ebXML*

- Utilizes a workflow approach where information and documents flow between business
- Concept of Collaboration Protocol Profile (CPP) is a specification of the services offered by a ebXML compliant Trading Partner
- A Business process is a set of business document exchanges between the Trading Partners
- CPPs contain industry classification, contact information, supported Business Processes etc.
- CPPs are registered within an ebXML registry
- Trading Partners must agree on how business documents are specified



## ***DAML-S and the Semantic Web***

- DAML-S differs from many other Web Service standards:
  - Semantic markup will allow agents to understand content, discover, interoperate, transact and compose services
  - Designed to be fully automated, whereas other schemes involve human-in-the-loop.



# *DAML-S Resources*

- DAML-S Web page - <http://www.daml.org/services/>
- DAML-S Matchmaker - <http://www.damlsmm.ri.cmu.edu/>
- DAML Tools - <http://www.daml.org/tools/>
  - DAML Validator - <http://www.daml.org/validator/>
  - DAML Editor (Emacs Mode) - <http://openmap.bbn.com/~burstein/daml-emacs/>



## *Outline*

1. What are Web Services?
2. Mediation and Composition
3. Industry Standards
4. Semantic Web efforts
- 5. Future Directions**
6. Discussion



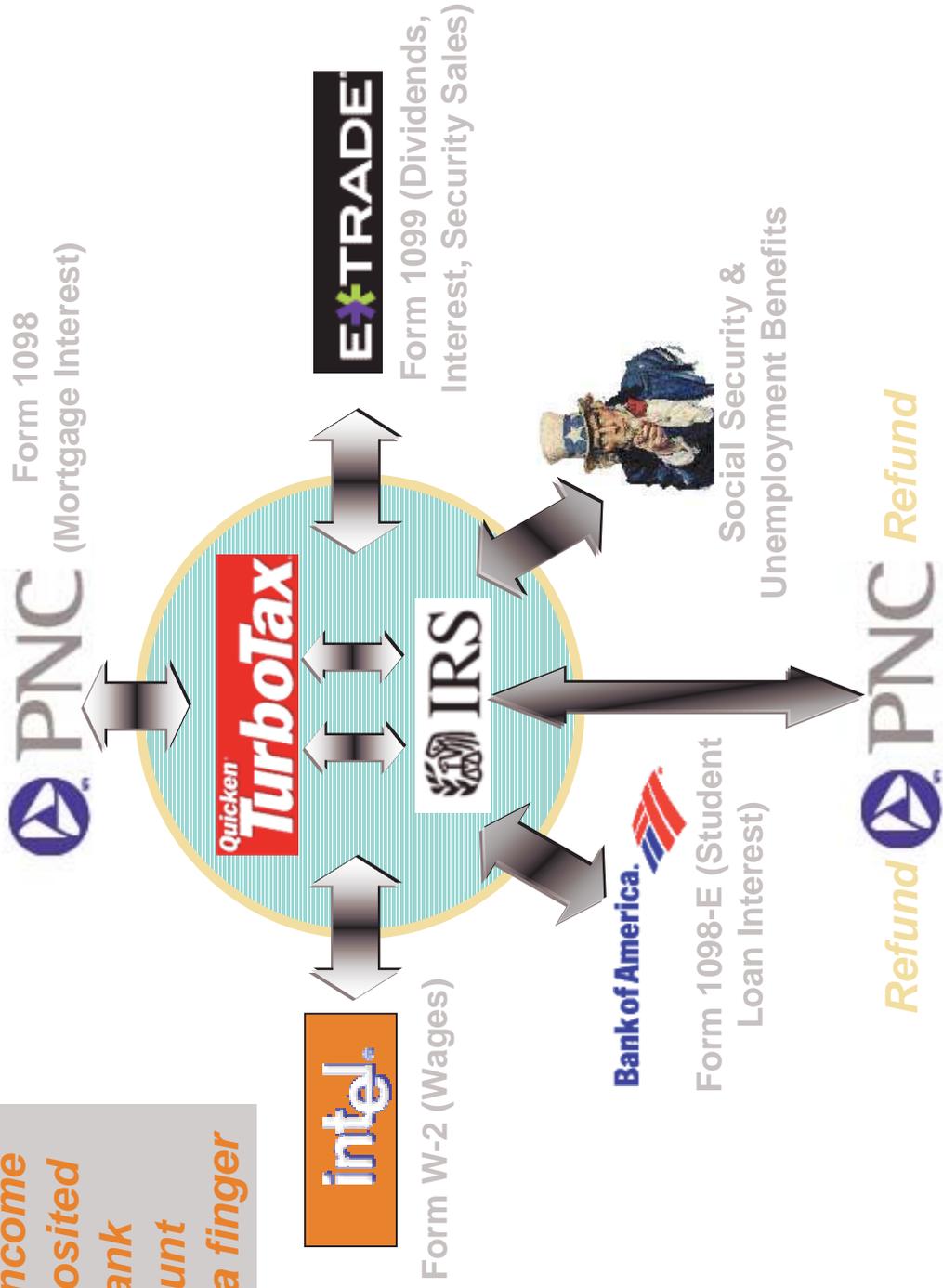
# *Web Service and the Semantic Web*

- **Current Status**
  - Where are Web Services heading?
- **Applications**
  - Case studies of emerging Web Services
- **Realizing Services on the Semantic Web**
  - What challenges / issues need to be addressed?



# Where are web services going?

*Imagine your income tax refund deposited in your PNC Bank checking account without lifting a finger*





# *Current and Emerging Applications*



- Using web services to broker shipping orders to authorized agents or independent third party transportation companies



- Using web services to create a structured data application programming interface that allows third parties to create booking engines and server-based applications



- Using web services to develop an open non-proprietary solution to connect 3rd-party service providers



- Using web services to integrate web-based applications with legacy applications



## *Agents, Web Services and the Semantic Web*

- How can a symbiosis of Agents, Web Services and the Semantic Web be achieved?
  - Tools for automatic creation of product ontologies, annotations and annunciations
  - Versioning and Managing Changes
  - Managing and Mediating between different versions of Web Service protocols and standards
  - Privacy / Trust / Security
  - Business Models - Thoughts and Solutions