Service Oriented Architectures and Semantic Web Processes

Jorge Cardoso[1], *Francisco Curbera*[2], *Amit Sheth*[3]

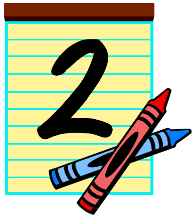
[1]University of Madeira (Portugal)

[2]IBM T.J. Watson Research Center (USA)

[3] LSDIS Lab, University of Georgia and Semagix, Inc (USA)

Service Oriented Architectures and Web Services

Semantic Web Processes



Semantic Web Processes

Part 

Service Oriented Architectures and Web Services

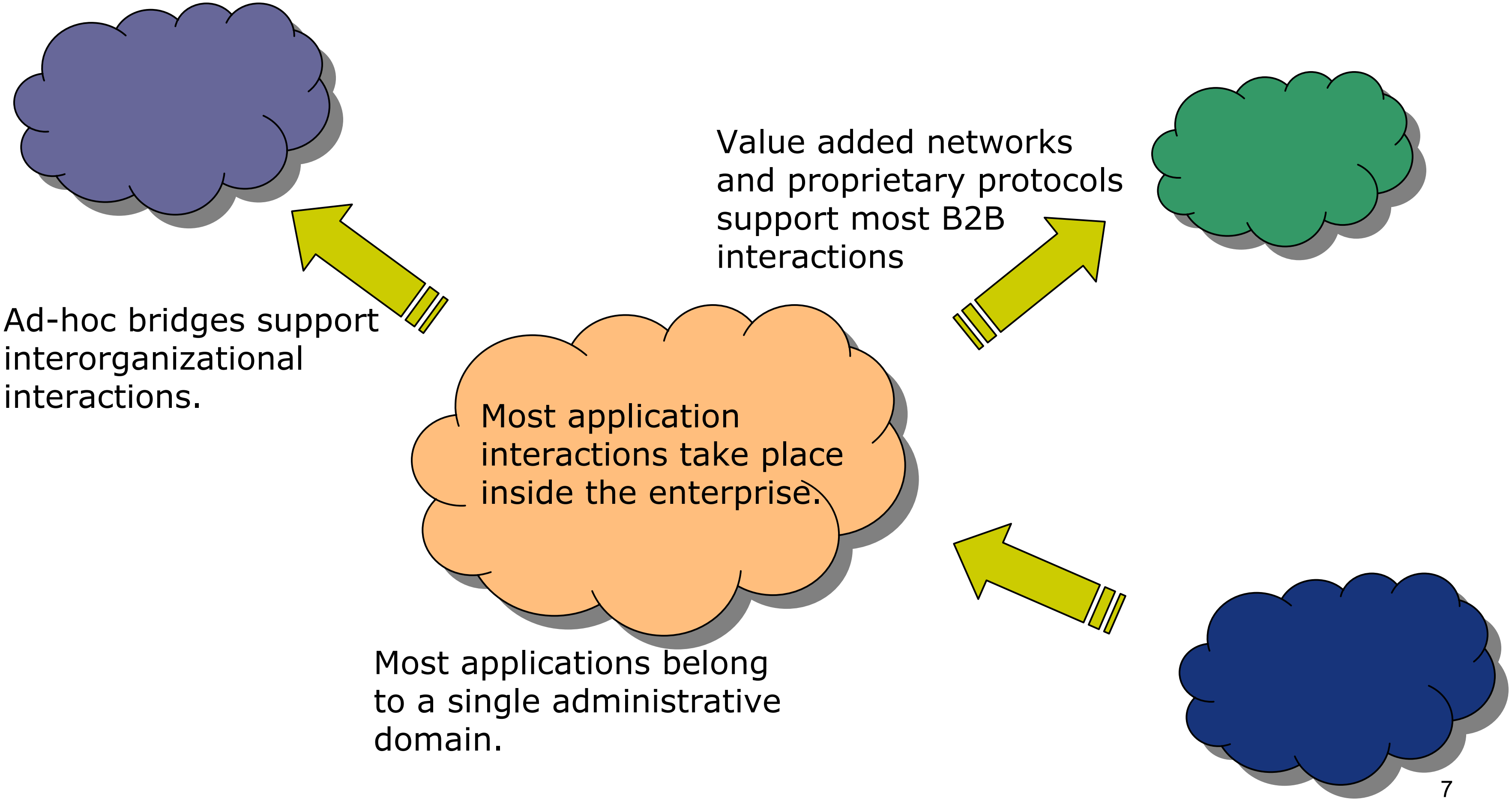
Overview

- IT for a new business model
- Service Oriented Architectures (SOAs).
- Web services as an XML based instantiation of SOA.
  - Protocols.
  - Metadata.
  - Discovery.
  - Composition.
- Summary.

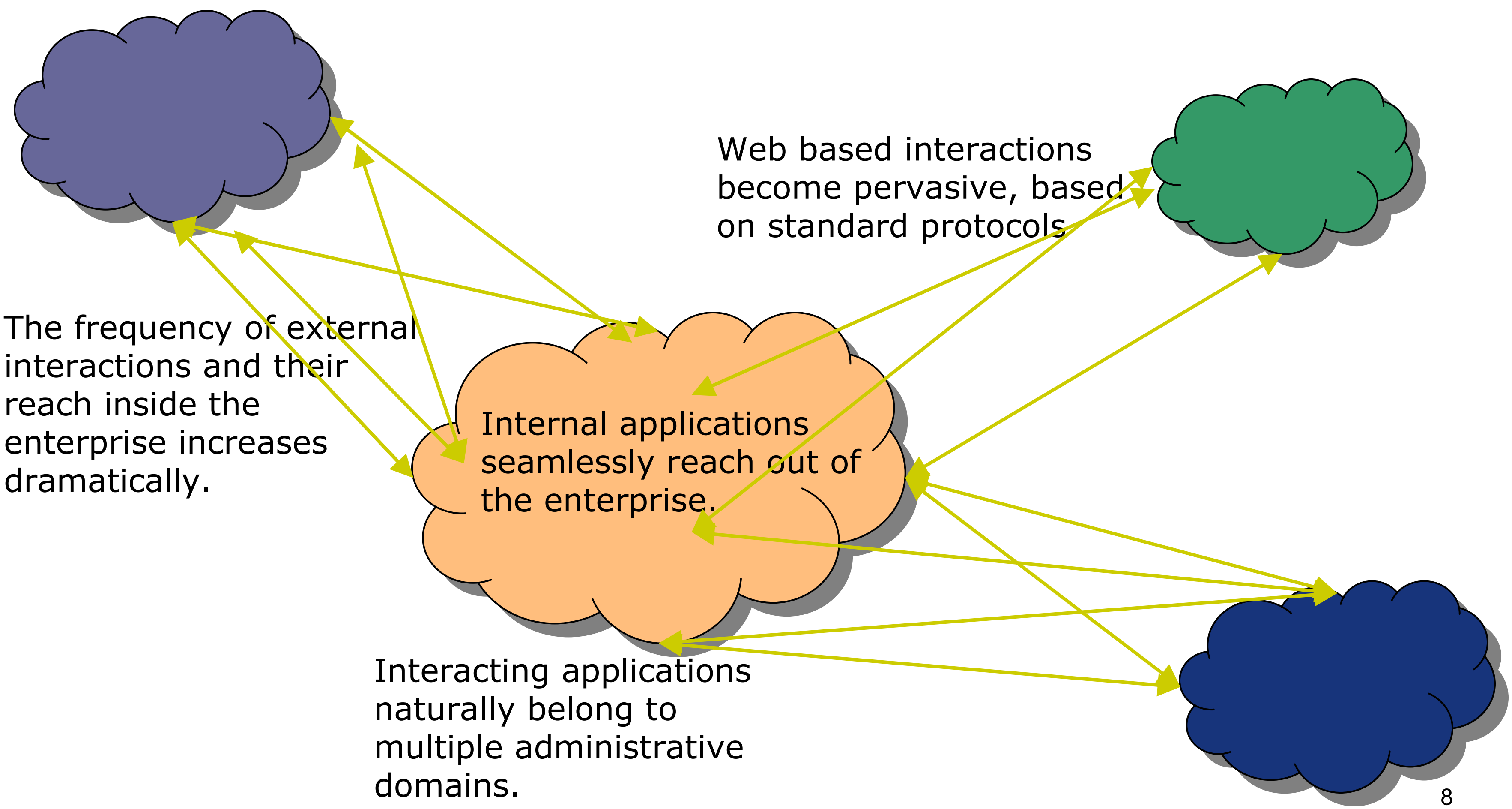
A New Business Environment

- Business outsource every non-essential function.
  - Concentrate on core function and values.
- Vertically integrated enterprises are being broken apart
  - Replaced by heavily networked ones.
  - Applications that used to be internal are now provided by outside parties.
- Corporate boundaries become fuzzier.
- Does today’s IT models support the new business environment?
  - IT is too centered on IT!
  - When enterprises where islands this was sort of OK.
  - Today it is vital to adapt the computing model to the business interaction model.

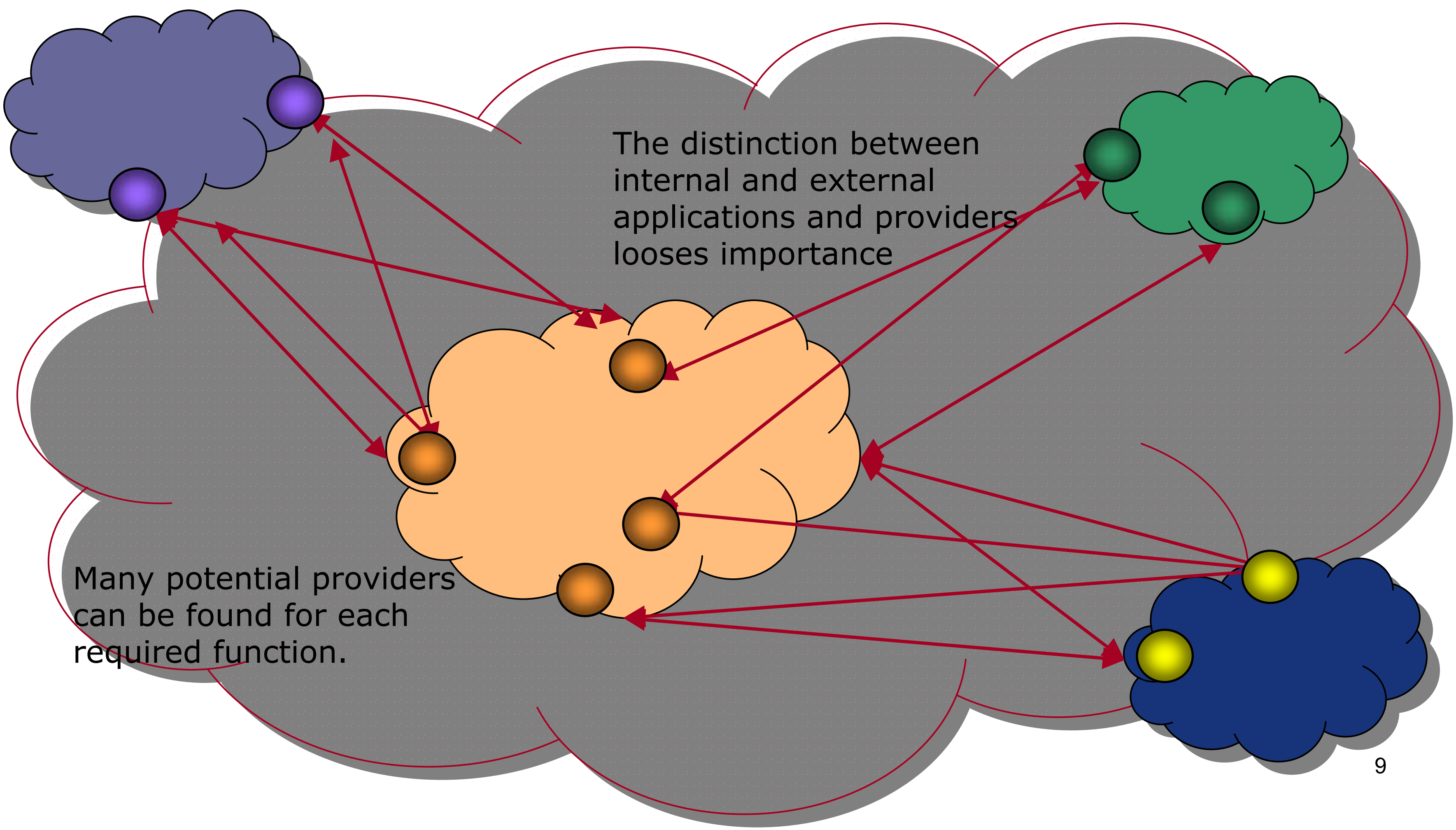
Enterprises as IT Islands



Fully Networked Enterprises



Fully Networked Business Interactions



IT for the New Enterprise: Business Components

- Need to raise the level of IT abstractions.
  - Concentrate on business function and requirements.
- Need to encapsulate business function to make it available to partners: **service components.**
  - Different level granularity – coarse grained business services vs. fine grained objects.
- Services must be defined by **explicit contracts** to allow independent party access.
  - Consequence is automatic binding.
- Core concern of business is to integrate business processes and functions.
  - Business components are integrated creating **service compositions.**
  - New value is created through integration/composition.
  - New components are recursively created.

Business Interactions

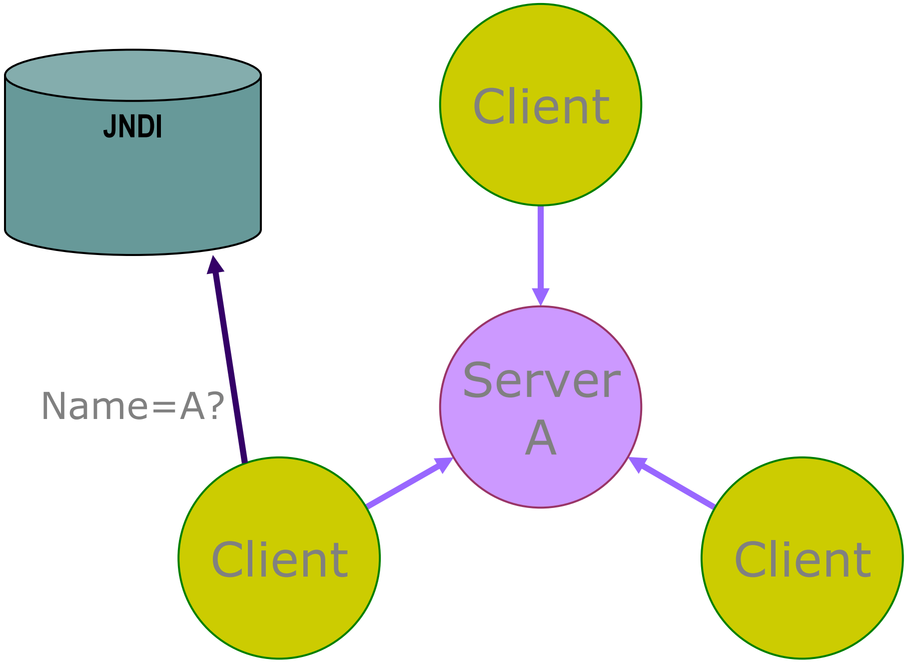
- Business interact over standard protocols.
- Businesses interact as peers:
  - Interactions are not client-server.
  - They are “**conversational**” in nature: asynchronous, stateful, bidirectional.
- Business interactions are often multi-party interactions
  - Business process integration model is intrinsically multi-party.
  - Distributed multi-party interactions are a cornerstone of advanced enterprise integration:
  - Making distributed computing truly distributed.

What About The SOA Triangle?

- Standard protocols augment the pool of technically compatible services.
- Explicit contracts allow automatic discovery.
- Central registries build on registered contracts extend the reach of the enterprise both as provider and consumer of business services.

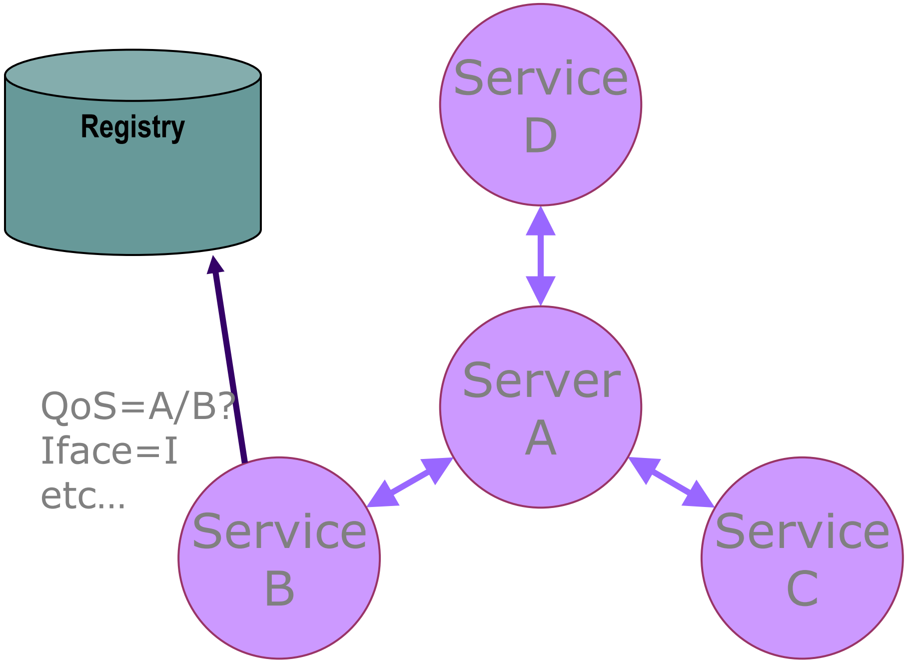
Traditional Middleware

- Distributed object systems
  - Based on client-server paradigm.
  - Heavily asymmetric interaction model.
  - Biased towards synchronous protocols.
  - Assigns public interfaces to network accessible objects.
  - Supports “name-oriented” object discovery.



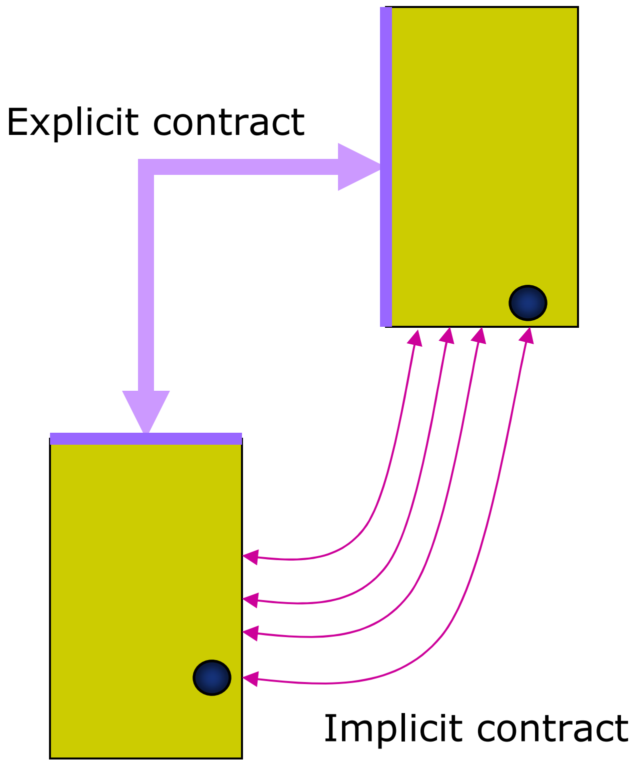
Service Oriented Middleware

- Service interactions
  - Peer to peer by nature.
  - Symmetric interaction model.
  - Mixes synchronous and asynchronous protocols.
  - Assigns public contracts to network accessible objects.
  - Supports capability based service discovery.



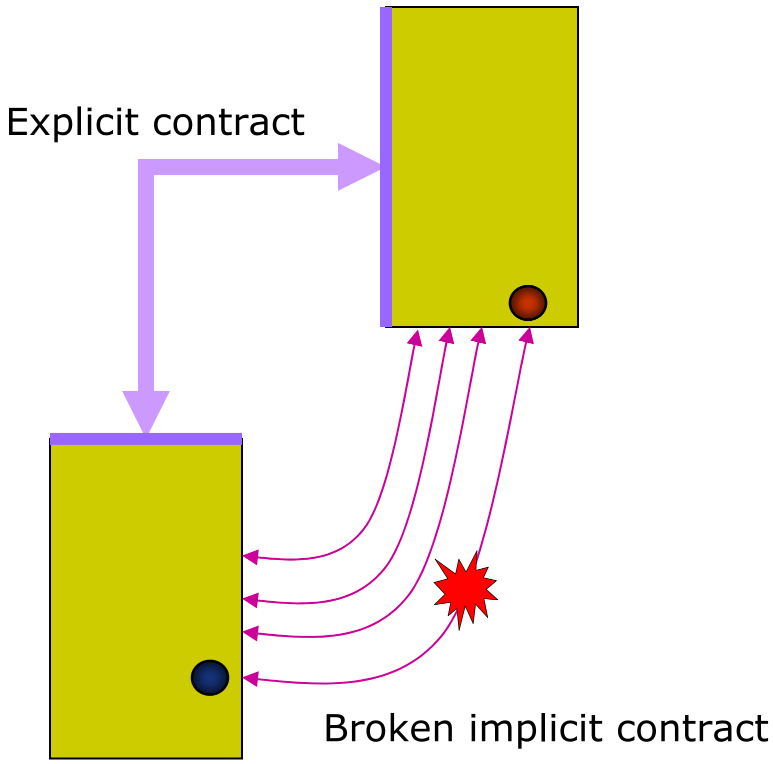
Coupling Between Applications

- Interacting applications are bound by the set of assumptions each one makes about the other:
  - What message formats can be sent/received
  - Constraints on how content of these messages
  - Sequencing information.
  - Required QoS characteristics of the interaction.



Tight and loose binding

- Tight coupling leads to monolithic and brittle distributed applications.
  - Even trivial changes in one component lead to catastrophic breaks in function.
  - Small changes in one application require matching changes in partner applications.
  - Lack of componentization and explicit contracts.



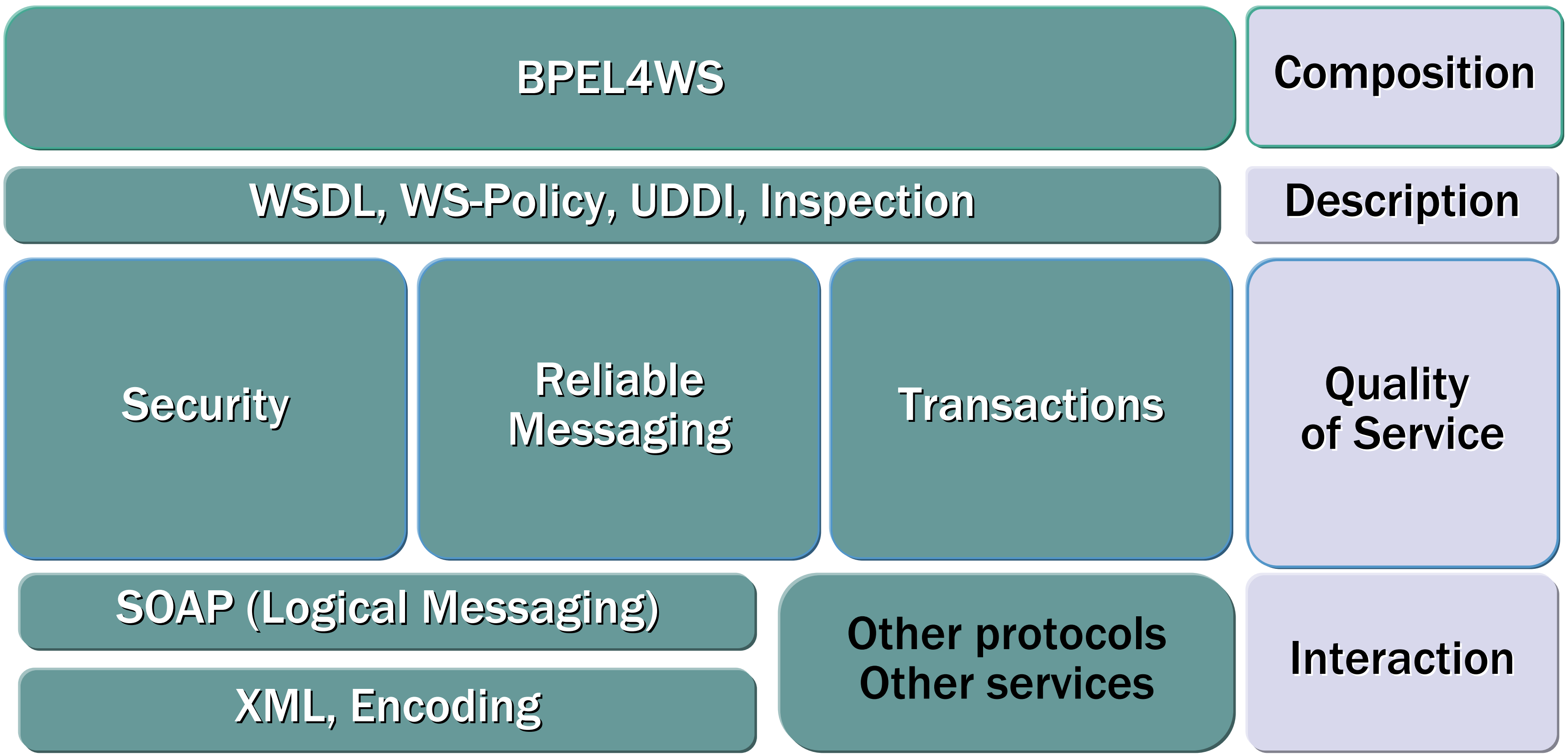
A Plan for Building a SOA

- Requirement #1: Interaction protocols must be standardized.
  - Need to ensure the widest interoperability among unrelated institutions.
- Requirement #2: Make all contracts explicit.
  - Explicit contracts define what may be changed in an application without breaking the interaction.
  - It is hard or impossible to make all assumptions explicit, but the more the better.
- Requirement #2 : Standardize contract language(s) and formats.
  - Standard metadata is the basis of interoperable contract selection and execution.
- Requirement #3: Allow for points of variability in the contract.
  - Dynamic adaptation on variability points.
  - Increases the number of possible interactions supported.
- Requirement #4: Provide native composition models and runtimes.

Web Services As a SOA

SOA and Web Services

Where Are We on Web Services?



Protocols

SOA and Web services

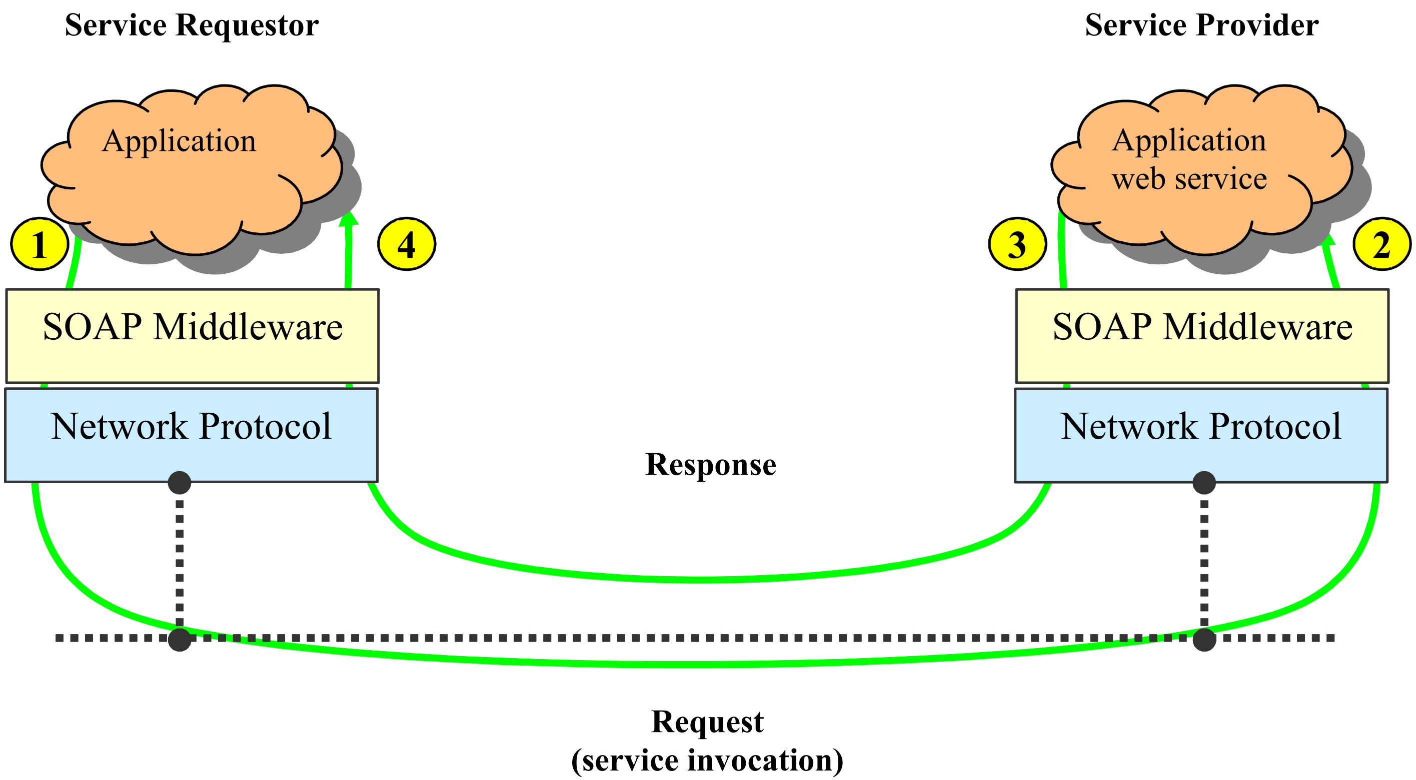
Protocols

- Provides a common set of universally supported interaction protocols.
- A basic messaging layer
  - SOAP
  - Easily extensible, allows QoS protocols to be defined on top.
- Some basic QoS protocols:
  - Basic requirements of business interactions.
  - Provide guarantees
  - Message Reliability, WS-ReliableMessaging
  - Coordination and transactional interactions.
  - Message integrity, confidentiality

SOAP (v1.1)

- A lightweight XML-based mechanism for exchanging structured information between peers in a distributed environment.
  - A transport-independent messaging model.
  - Transport bindings for HTTP
  - An encoding model for a type system, and an RPC convention: a link to “legacy middleware”.
- Built around a standard message format:
  - Envelope
  - Headers
  - Body
  - Possibly attachments.

SOAP Messaging



SOAP over HTTP

POST /StockQuote HTTP/1.1

Host: www.stockquoteserver.com

Content-Type: text/xml; charset="utf-8"

Content-Length: nnnn

SOAPAction: "Some-URI"

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV…
  SOAP-ENV:encodingStyle="…/>
   <SOAP-ENV:Header>
   </SOAP-ENV:Header>

   <SOAP-ENV:Body>
       <po:PlacePurchaseOrder xmlns:po=…>
            <OrderDate>02/06/01</OrderDate>
            <Ship_To>
              …
       </po: PlacePurchaseOrder >
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Headers

- Headers are managed and consumed by the Web services middleware infrastructure.
  - Headers support middleware protocols such as security, transactions, reliability, provisioning, etc.
- Extensible nature allows message to endowed with be an extensible set of QoS protocols.
- Header attributes
  - `actor`
    - Indicates the intended recipient of the header
    - `http://schemas.xmlsoap.org/soap/actor/next`
  - `mustUnderstand`
  - `encodingStyle`
    - Identifies serialization rules

SOAP Body and Attachments

- Body: belongs and is processed by the application level.
  - Is the only part that should be visible by the application logic.
  - Business modeling is the modeling deals with what goes in the body and how it is processed and exchanges.
  - A separation that shows up in WSDL, BPEL4WS as well.
- Attachments: Not all data can be conveniently placed within an XML document
  - SOAP Messages with Attachments: How to carry a SOAP envelope within a MIME Multipart/Related structure
    - SOAP envelope must be the root part
    - Type is `text/xml`
    - Uses `href` attribute to reference parts

SOAP Status

- SOAP 1.2/XML Protocol is now a W3C Recommendation.
  http://www.w3.org/TR/soap/

- SOAP 1.1 is still (and will be for a while) what is being deployed.
  http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

WS-Security

- SOAP header extensions for:
  - authentication,
  - confidentiality,
  - Integrity
- Built on top of W-Security:
  - Protocols for exchanging security tokens and establishing trust relationships built on top.
  - Protocols for authorization and identity propagation / mapping in multi-party communication

```
<wsse:Security?
  <wsse:UsernameToken Id="MyID">
      <wsse:Username>
       Zoe
      </wsse:Username>
  </wsse:UsernameToken>
  <ds:Signature>
       <ds:SignatureMethod
           Algorithm=
           "http://www.w3.org/..."/>
      ...
       <ds:SignatureValue>
         DJbchm5gK...
       </ds:SignatureValue>
       <ds:KeyInfo>
        <wsse:SecurityTokenReference>
          <wsse:Reference
                     URI="#MyID"/>
        </wsse:SecurityTokenReference>
       </ds:KeyInfo>
  </ds:Signature>
</wsse:Security>
```

WS Protocols - Summary

- SOAP defines a standard messaging model in which transport, service middleware and business concerns are clearly separated.
- Standardized QoS protocols ensure universal “on-the-wire” interoperability among businesses, applications.
- QoS Protocols build on SOAP header extensibility to augment business exchanges with QoS properties.

Metadata

SOA and Web services

Metadata

- WSDL: Functional descriptions.
- WS-Policy: QoS
- Points of variability: dynamic infrastructure.

What is WSDL

- An extensible, platform independent XML language for “describing” services.
- Provides functional description of Web services:
  - IDL description
  - Access protocol and deployment details
  - All of the functional information needed to programmatically access a service, contained within a machine-readable format
- Does not include
  - QoS
  - Taxonomies
  - Business information
- WSDL is a component definition language for Web service component

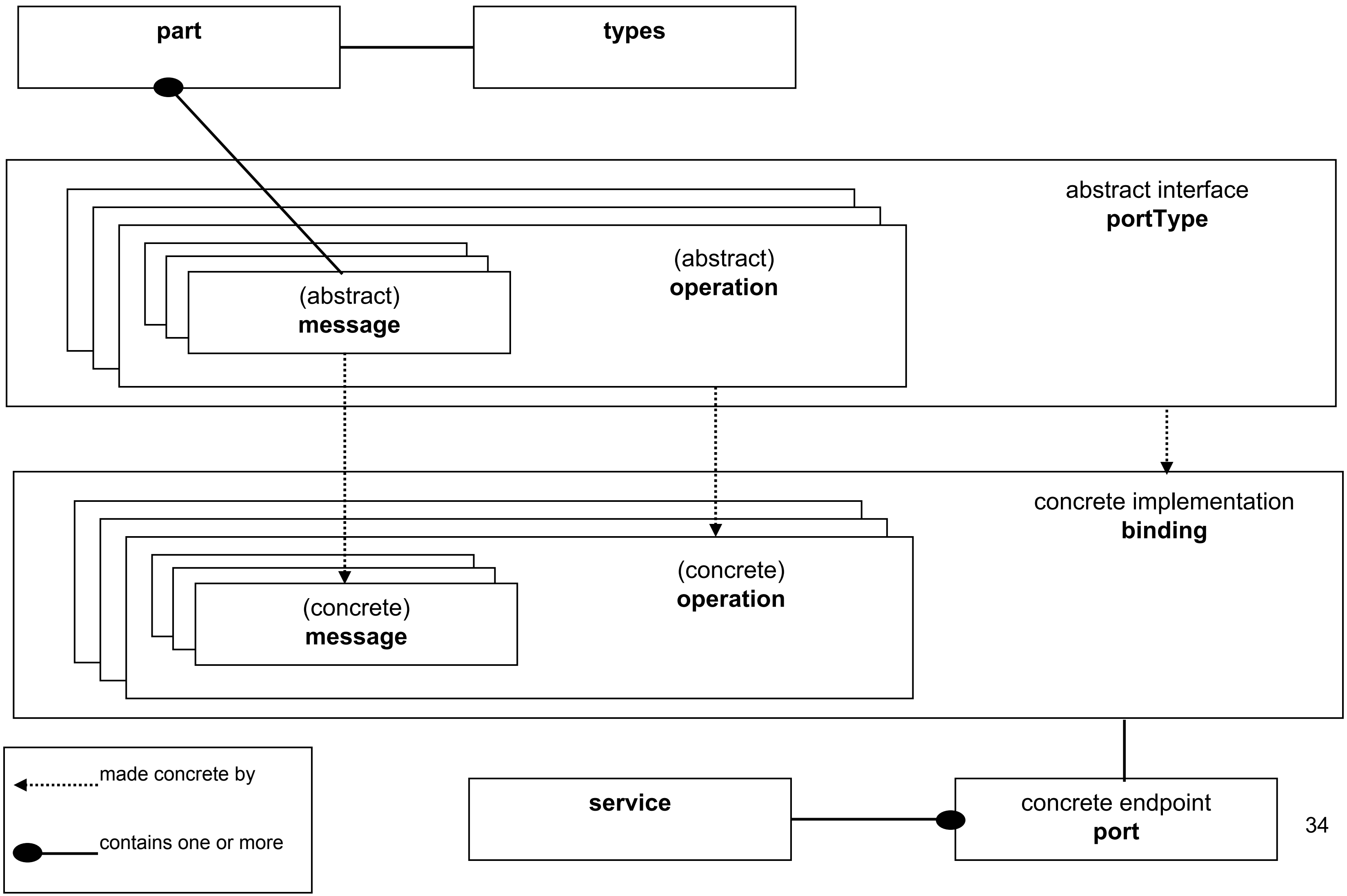
WSDL Description Structure

```
<definitions>
    <types> …
    <message name="Msg1"/> …
    <portType name="PType1"> …
    <binding name="Bnd1" type="PType1"> …
    <service name="svc1">
        <port binding="Bnd1" >
          <soap:address location="..."/>
        </port>
    </service>
</definitions>
```

Abstract/
Business

Deployment

WSDL Parts At a Glance



WSDL in SOA

1. Allow industries to define standardized service interfaces.
   - Functional contract definition.

2. As an extended IDL: base for tools generating compliant client proxy and server stub
   - Tool level interoperability.

3. Allowing advertisement of service descriptions,
   - enables dynamic discovery of compatible services and dynamic binding to the actual service provider
   - Works within registries and with discovery protocols.

4. As a normalized description of internally heterogeneous services

WSDL Status

- WSDL 1.1 was submitted to the W3C on February 2001.

  http://www.w3.org/TR/WSDL

- WSDL 2.0 is now being defined by the WS Descriptions working group at W3C.
  - Last draft (June 2002) available at

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

WS-PolicyFramework

- Complements functional description of services with QoS behaviors.
- General framework for declaratively asserting how a service may be accessed:
  - Requirements
  - Constraints
  - Capabilities
- WS-Policy provides a general framework in which arbitrary domain specific “assertions” are used.
  - Security
  - Transactions
  - Reliable messaging

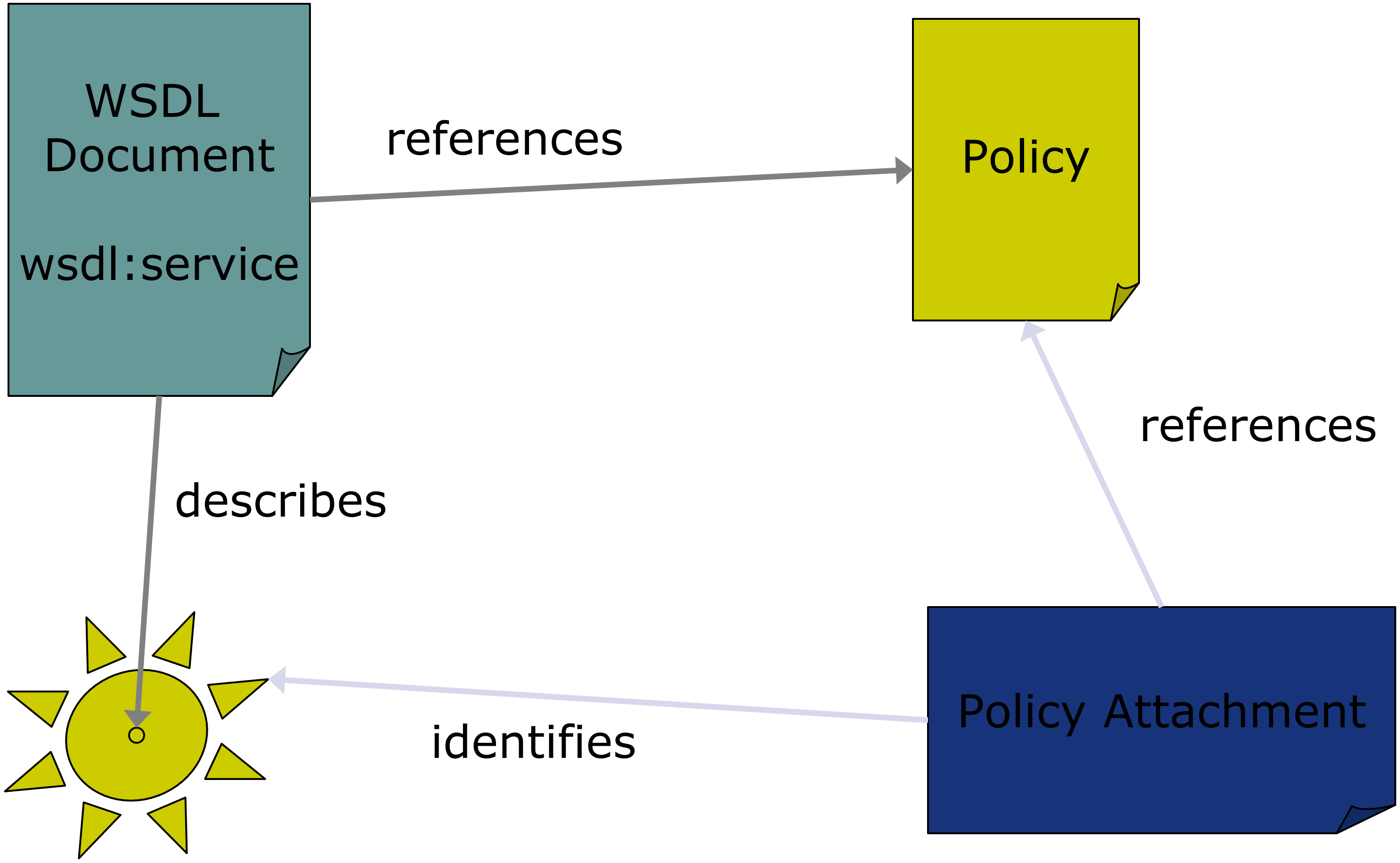
Policy Expressions

```
<wsp:Policy id=“…”>
    <wsp:ExactlyOne>
        <wsp:All>
            <wsse:SecurityToken>
                <wsse:TokenType>wsse:Kerberosv5TGT</wsse:TokenType>
            </wsse:SecurityToken>
            <wsse:Integrity>
                <wsse:Algorithm Type="wsse:AlgSignature“ … />
            </wsse:Integrity>
        </wsp:All>
        <wsp:All>
            <wsse:SecurityToken>
               <wsse:TokenType>wsse:X509v3</wsse:TokenType>
            </wsse:SecurityToken>
            <wsse:Integrity>
                <wsse:Algorithm Type="wsse:AlgEncryption“ …/>
            </wsse:Integrity>
        </wsp:All>
    </wsp:ExactlyOne>
</wsp:Policy>
```

Policy Expressions

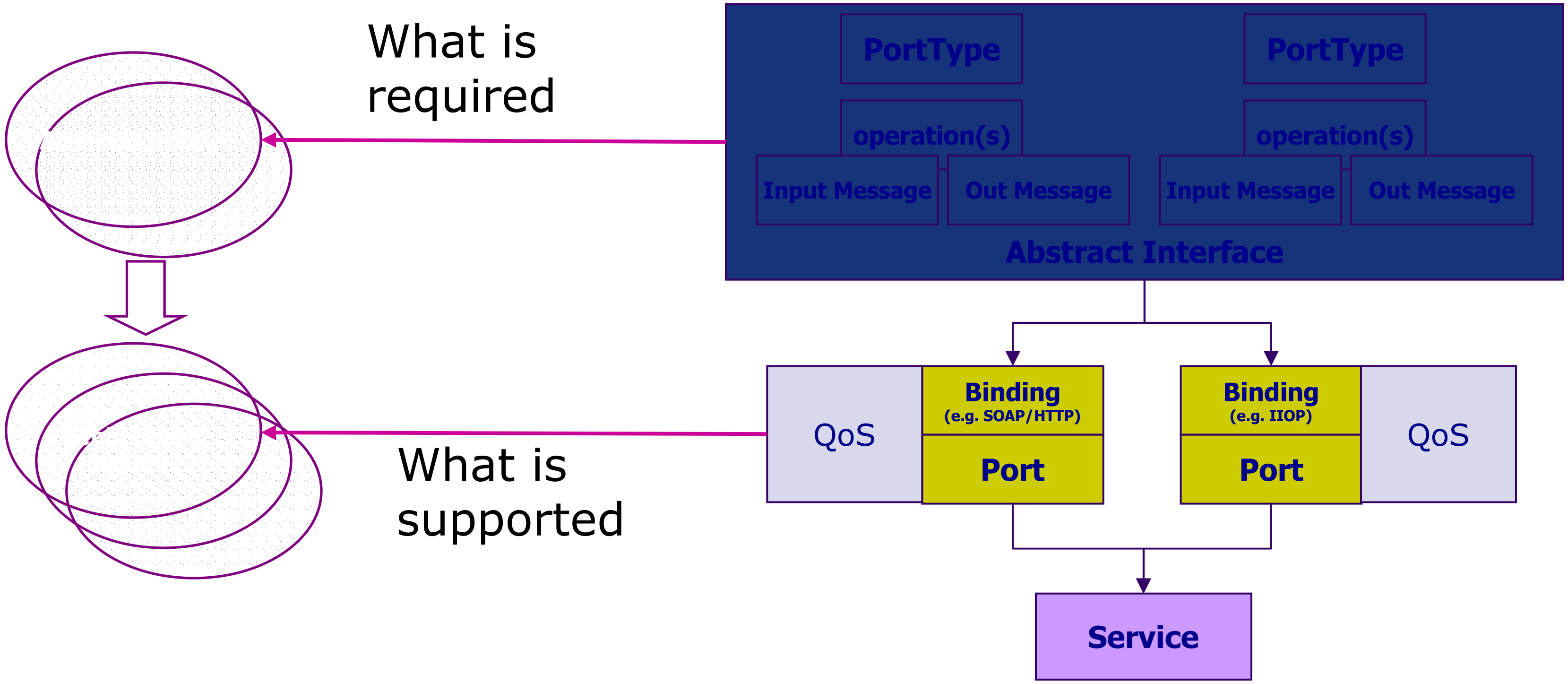
- Three generic policy operators allow combining assertions into groups, options:
  - <All>
  - <ExactlyOne>
  - <OneOrMore>
- Usage attribute allows modification of standard meaning of assertion:
  - Usage="Rejected" prevents requesters from following certain behaviors ("do not log messages!").
- Policies can be names so they can be referenced from other documents and reused.
  - Id attribute assigns a URI to the policy.
  - QName naming is also allowed.

Attaching Policies



WSDL and WS-Policy

- Abstract and deployment policies



WS-Policy and SOAP

- Policies define what QoS protocols are followed.
- Are reflected on what headers appear in the SOAP envelope.
  - QoS policies attached to a service of service endpoint represent protocols.
  - QoS protocols are supported by SOAP headers.

```
<wsp:Policy id="…">
  <wsse:SecurityToken>
    <wsse:TokenType>
    wsse:X509v3
    </wsse:TokenType>
   </wsse:SecurityToken>
</wsp:Policy>

<SOAP-ENV:Envelope>
  <SOAP-ENV:Header>
    <wsse:Security>
    <wsse:BinarySecurityToken
        Id="myToken"
        ValueType="wsse:X509v3"
        EncodingType=
            "wsse:Base64Binary">
        MIIEZzCCA9Cg...
    </wsse:BinarySecurityToken>
    </wsse:Security>
  </SOAP-ENV:Header>
<SOAP-ENV:Body> …
```

Using WS-Policy

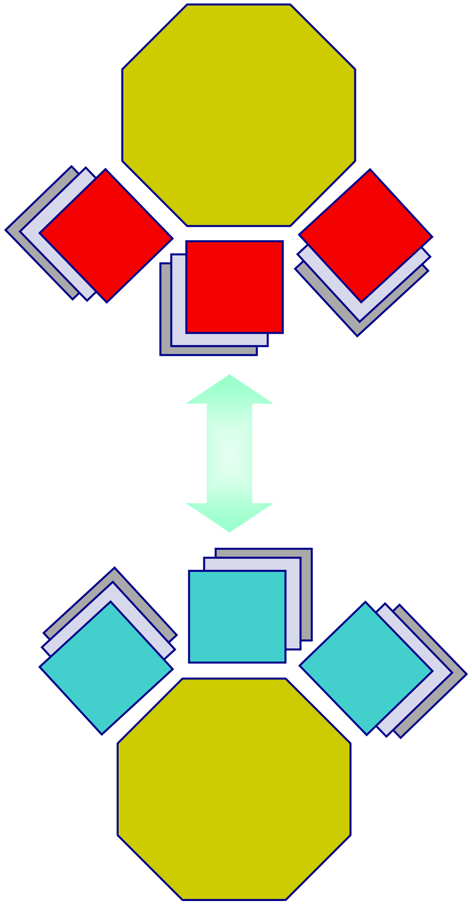
- Requester finds out QoS requirements stated by provider and configures itself accordingly:
  - Both development time and runtime usage.
  - Many options may be available
- Requester searches for services that support its QoS requirements.
  - Discovery time.
- Match-maker finds compatible services in peer to peer setting.
  - Symmetric discovery scenario.
- Contracts may be formulated based on compatibility of published policies.
  - Business implications of policy matching.

What is the Typical Usage Scenario

- Simple SOA model:
  - WSDL description or UDDI service entry identify all policies that are followed by a service.
  - Service requesters check for services whose interface and policies indicate technical compatibility with their requirements.
- It is a static model
  - Policies are used to represent the stack of technologies supported by the service.
  - A “match” represents a service using a compatible policy stack.
- Typically results in implicit binding between application implementations.
  - Loose coupling is limited to selecting among technically equivalent services, using non-functional aspects (price, ratings, etc.)
- This is a direct extension from today’s development models.
  - The stack is fixed at development/deployment time.
  - SOA model essentially introduces the publishing of descriptions and runtime selection.

Dynamic Middleware Reconfiguration

- Effective dynamic binding requires run-time adaptation of middleware configuration:
  - J2EE focused on moving middleware configuration away from the code developer and into the deployment phase.
  - SOC requires moving it further to follow runtime discovery of services:
- Seamlessly adapt to policy settings of target, select among possible options, carry on basic a policy negotiation.



WS-Policy

- Status: WS-Policy specifications published withy RF licensing terms at:
  http://www-106.ibm.com/developerworks/webservices/library/ws-polfram/summary.html
  - WS-PolicyFramework
  - WS-PolicyAttachments
  - To be submitted for standardization.

Service Metadata - Summary

- Explicit metadata is the central characteristic of SOA
- Metadata must completely define the service contract, including both functional and non-functional aspects.
  - WSDL
  - Policies
- Metadata can support service discovery as well as tooling.
- Advanced runtimes can derive greater flexibility from contract variability points.

Discovery

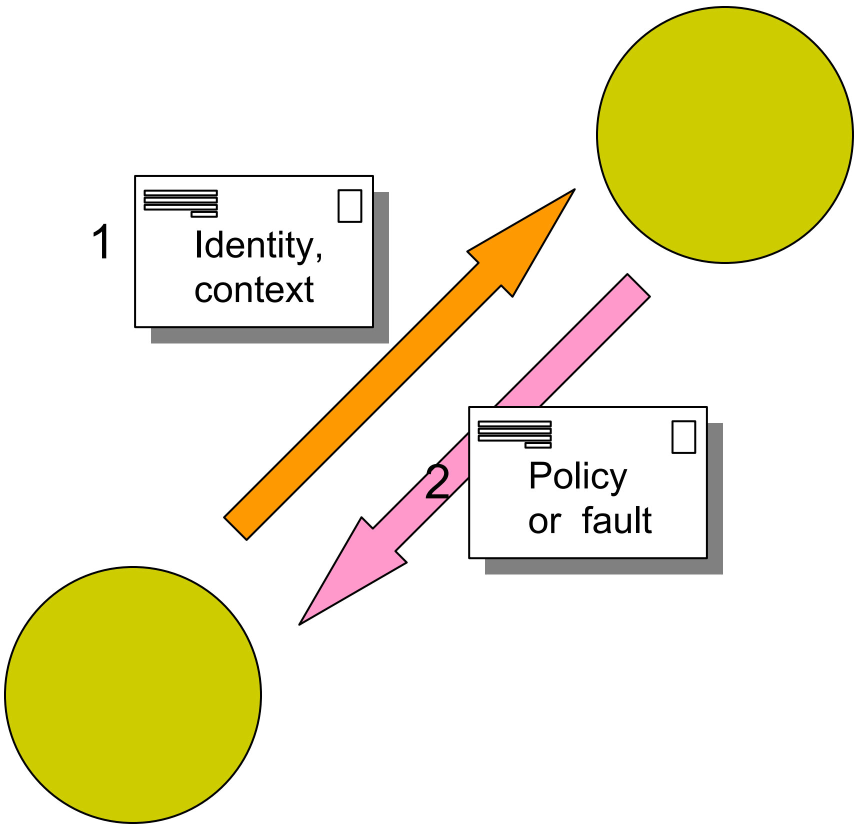
SOA and Web services

Discovery Infrastructure

- Registries
  - Requesters search for providers in third party central directory.
  - Provider policies are retrieved from registry.
  - Requester interacts according to discovered policies.
  - Will not deal with here.
- Metadata exchange
  - Requesters and providers can exchange policies directly, no third party involved.

WS-Metadata Exchange

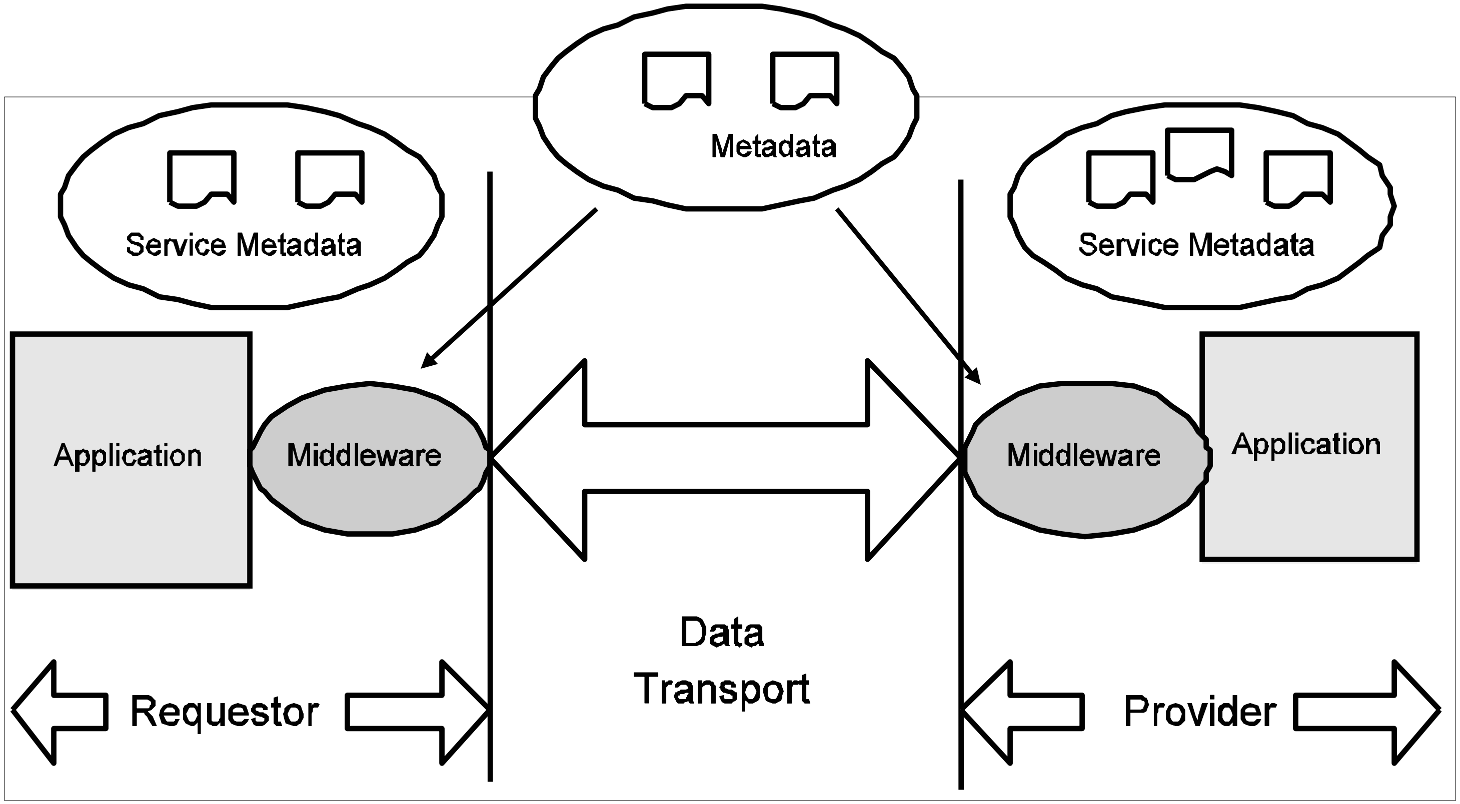
- Goal: Allow providers to customize their policies to individual requesters and interactions.
- Requesters send:
  - Requester’s policies can be explicitly communicated.
  - Requester’s execution context may be implicitly transmitted.
- Providers return set of policies to apply to interaction.
- “Faults” should be thrown if any party finds it cannot deal with the other’s policies.



More on Metadata Exchange

- Takes place at the beginning of an interaction.
  - MDE model is a request-response interaction for retrieving custom policies.
  - Policies are set from then on.
- Both parties' middleware must be able to deal with dynamically discovered policies.
  - Start-time (re) configuration of component characteristics.
  - Component is reconfigured to deal with discovered policies that apply to the interaction.
- In flight metadata exchange?
  - Any party can send unsolicited policies at any point in the interaction.
  - Applies in particular to long running transactions where changes in policies are not unlikely.
  - The scope of the new policies will need to be clearly defined.

Metadata and Channel Configuration



Cooperative Middleware

- Joint work with Nirmal Mukhi and Ravi Konuru
- Requesters and providers cooperate to optimize the interaction channel.
  - Through “cooperative” reconfiguration of their middleware.
  - Follows a dynamic exchange of policies and negotiation.
  - Distributes roles and function between the two endpoints to optimize overall interaction.
  - Optimal configuration is negotiated.
- Must assume a trusted relationship between the parties.

Cooperative Specialization Use cases

- Mobile clients and servers negotiate downloading of server function to clients.
  - Known approach, NOT metadata based.
  - Hardwired protocol essentially fixes the what function can be offloaded.
  - Metadata allows flexible reuse of a common protocol for negotiating different functions.
- Example:
  - Schema validation offloading to client app.
  - Control of the application flow can be offloaded to allow disconnected operation.
- Offloading takes place selectively based on client and server declared capabilities (policies).

Discovery - Summary

- Metadata-based discovery of services is a basic SOA capability.
- The discovery of metadata itself, however, does not necessarily need to follow the registry pattern.
- A dynamic middleware infrastructure is required to take full advantage of dynamic discovery (of both services and metadata).

Composition

SOA and Web services

Service Composition

- Service composition is the core sw. development task in SOA.
  - Applications are created by combining the basic building blocks provided by other services.
  - Service compositions may themselves become services, following a model of recursive service composition.
- Composition assumes an interaction model between components:
  - P2P conversational interactions.
  - Interactions are naturally multi-party interactions.
- Many composition models are possible. We know about two:
  - Process oriented composition – BPEL4WS
  - Distributed composition – WSFL Global models.

BPEL Concepts

- A BPEL process defines the structure of the interaction in terms of
  - participant services (partners)
    - Characterize partners
    - Provide support partner conversation
  - business logic.
    - Data
    - Control flow
    - Error handling and recovery mechanism

Structure of a BPEL4WS Process

```
<process ...>

    <partners> ... </partners>
        <!-- Web services the process interacts with -->
    <correlationSets> ... </correlationSets>
        <!– Used to support asynchronous interactions -->
    <variable> ... </variable>
        <!– Data used by the process -->
    <faultHandlers> ... </faultHandlers>
        <!–Alternate execution path to deal with faulty conditions -->
    <compensationHandlers> ... </compensationHandlers>
        <!–Code to execute when “undoing” an action -->
    (activities)*
        <!–  What the process actually does -->
</process>
```

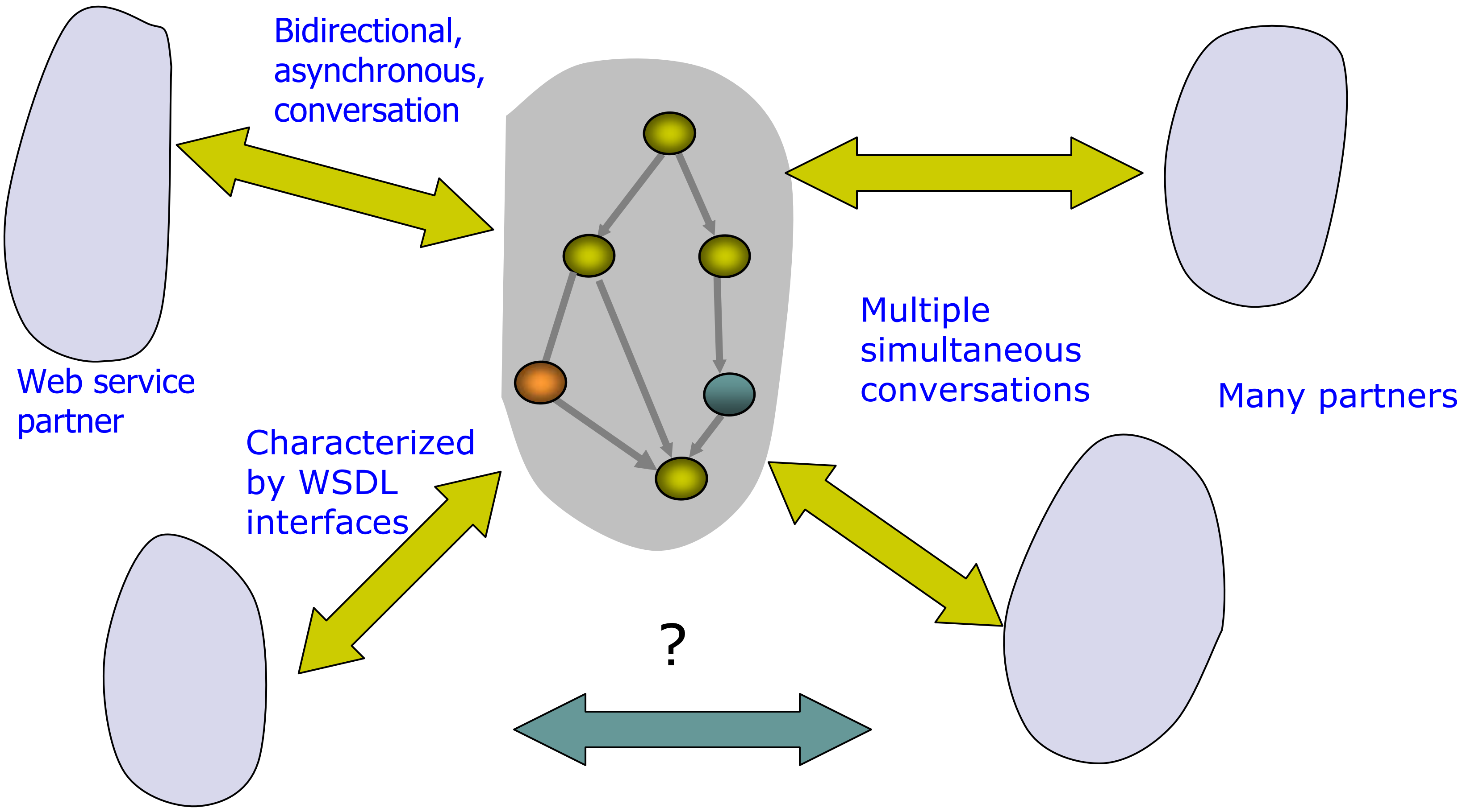
Partner information

Business logic

BPEL Partners

- Partners:
  - A composition defines a new service(s) which interacts with one or more partners.
  - Partners are characterized by a pair of abstract WSDL interfaces:
    - How the composition uses and is used by the partner.
- Interactions between partners are thus bidirectional, conversational in nature.
    - May combine synchronous and asynchronous interactions
    - Stateful.
- How is state maintained?
  - BPEL correlation mechanism uses business data to maintain the state of the interaction.
  - Other middleware mechanism are possible as well.

BPEL4WS Partners



What is Correlation?

- Correlation sets provide support for stateful interactions.
  - CSs represent the data that is used to maintain the state of the interaction (a “conversation”).
  - At the process end of the interaction, CSs allow incoming messages to reach the right process instance.
- What is a correlation set?
  - A set of application fields that capture the state of the interaction (“correlating business data”). For example: a “purchase order number”, a “customer id”, etc.
  - Each set is initialized once
  - Its values do not change in the course of the interaction.

Defining Correlation Sets

```
<correlationSet name="..." properties="..."/>
```

<!– A CS is a named set of properties. Properties are defined a WSDL extensibility elements: -->

```
<bpws:property name="..." type="..."/>
```

<!– A property has a simple XSD type and a global name (Qname) -->

```
<bpws:propertyAlias propertyName="..."
                  messageType="..." part="..."
                  query="..."/>
```

<!– A property is “mapped” to a field in a WSDL message type. The property can thus be found in the messages actually exchanged. Typically a property will be mapped to several different message types and carried on many interactions, across operations and portTypes -->

Business Logic in BPEL

- Workflow-like business logic is used to specify the sequencing of the interactions with partners.
  - Activities representing service interactions and data manipulation.
  - Control constructs that combine activities: links, sequences, conditionals, etc.
- The asynchronous nature of interactions is supported by event handlers.
- Failure conditions and recovery are supported through by fault handlers and compensatable scopes.

BPEL Basic Activities

```
<invoke partner="..." portType="..." operation="..."
        inputContainer="..." outputContainer="..."/>
   <!-- process invokes an operation on a partner:      -->

<receive partner="..." portType="..." operation="..."
         container="..."/>
   <!-- process receives invocation from a partner:       -->

<reply partner="..." portType="..." operation="..."
       container="..."/>
   <!-- process send reply message in partner invocation:     -->

<assign> <!– Data assignment betwee containers:      -->
   <copy>
     <from container="..."/> <to container="..."/>
   </copy>+
</assign>
```

BPEL Structured Activities

<sequence>

<!– execute activities sequentially-->

<flow>

<!– execute activities in parallel-->

<while>

<!– iterate execution of activities until condition is violated-->

<pick>

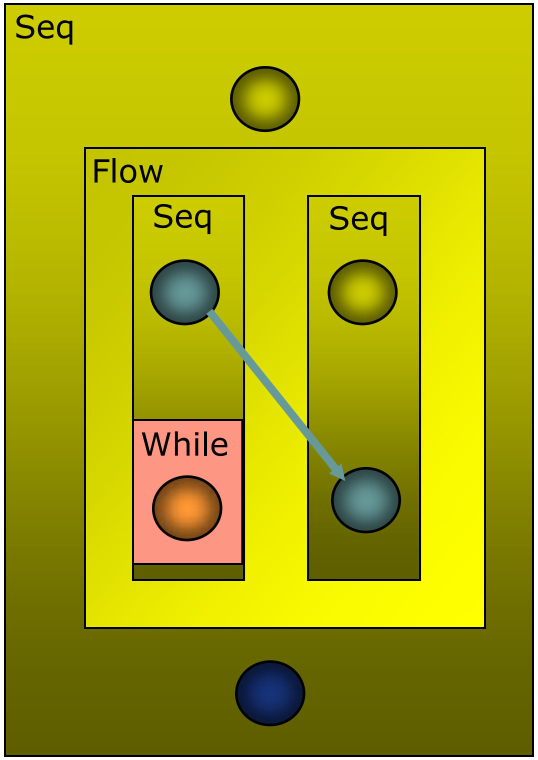
<!– several event activities (receive message, timer event) scheduled for execution in parallel; first one is selected and corresponding code executed. -->

<link ...>

<!– defines a control dependency between a source activity and a target -->

Nesting Structured Activities. Example

```
<sequence>
   <receive .../>
   <flow>
      <sequence>
         <invoke .../>
         <while ... >
            <assign> ... </assign>
         </while>
      </sequence>
      <sequence>
         <receive .../>
         <invoke ... >
      </sequence>
   </flow>
   <reply>
</sequence>
```

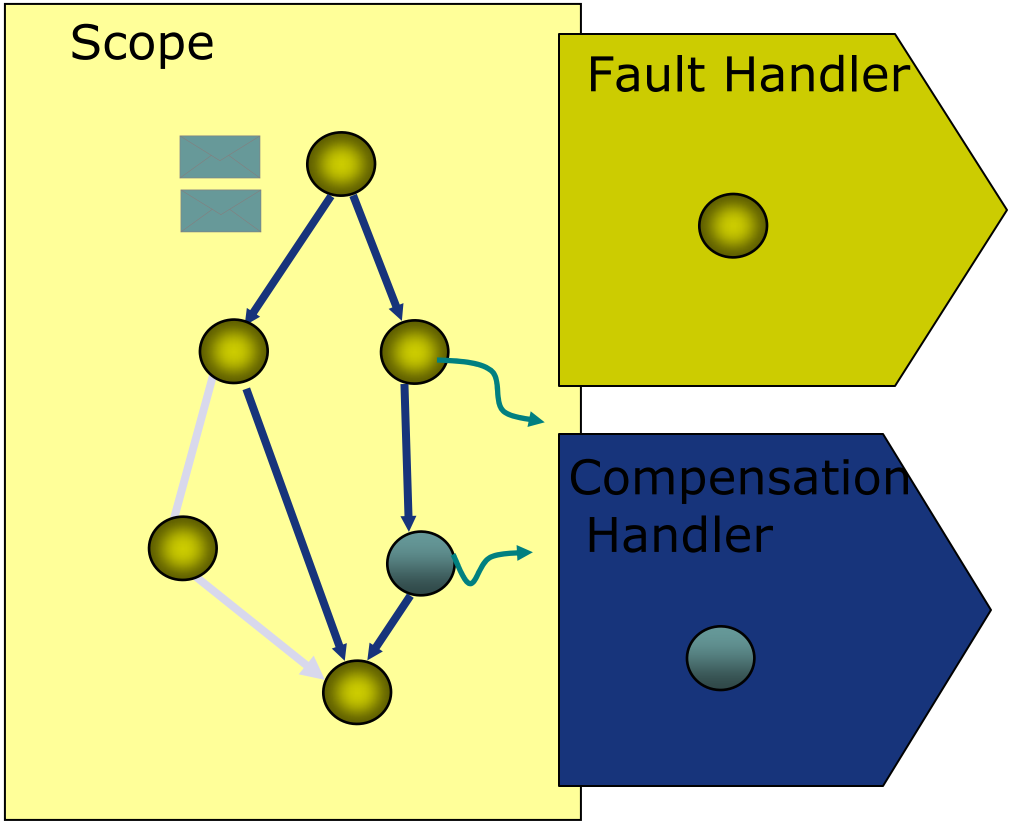


BPEL Handlers and Scopes

A **scope** is a set of (basic or structured) activities.

Each scope can have two types of **handlers** associated:

- **Fault handlers.** Many can be attached, for different fault types.
- **Compensation handlers.** A single compensation handler per scope.



How Handlers Work

- A compensation handler is used to reverse the work performed by an already completed scope
  - A compensation handler can only be invoked by the fault handler or compensation handler of its immediate enclosing scope

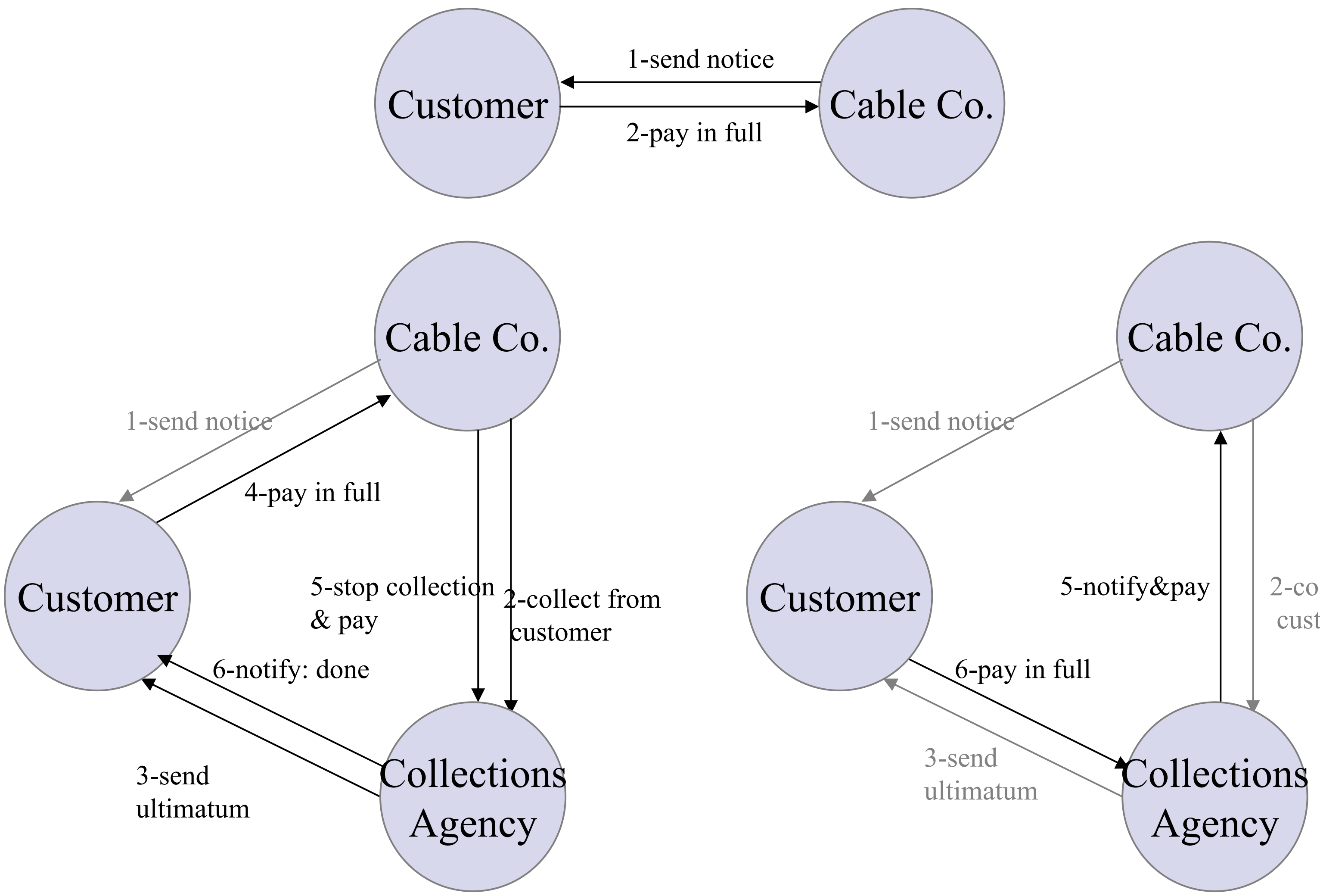
- A fault handler defines alternate execution paths when a fault occurs within the scope.

- Typical scenario:
  1. Fault is thrown (retuned by invoke or explicitly by process)
  2. Execution of scope is terminated
  3. Appropriate fault handler located (with usual propagation semantics)
  4. Main execution is compensated to “undo” business effects of unfinished work.

Global Models

- BPEL processes capture multi-party interactions from a single party perspective.
  - There isn't a well accepted format for capturing these interactions.
- Complex interactions are naturally multi-party.
  - Single party view does not capture the global sequence of interactions
  - Each party may not be involved in every relevant interaction.
- Where are global models?
  - WSFL (a BPEL precursor) introduced global models.
  - WS-Choreography WG in W3C has been working on this concept as well.

Global Models, an Example



Composition - Summary

- Business integration becomes service composition in SOA.
- An interaction model needs to be assumed for composition, and supported by the corresponding composition models.
- BPEL composition natively supports a multi-party, conversational model.
- To support the full array of distributed compositions needs a global model formalism in addition to process centric compositions (BPEL).

Summary

SOA and Web services

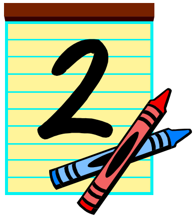
Web Services as an Instantiation of SOA

- SOA is more than “publish/find/bind”.
- Implies a completely business re-orientation of computing.
- SOA builds on:
  - Standard interaction protocols.
  - A component model, as defined by service contracts.
  - A conversational interaction model.
  - A set of service composition model.
- Web services provide an XML based instantiation of SOA.

End

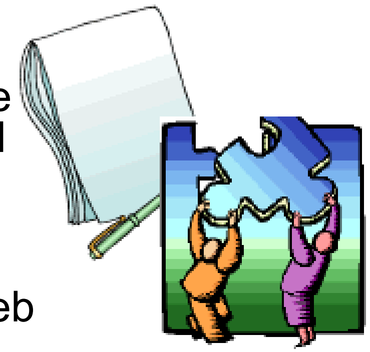
Part 

Semantic Web Processes Overview



- Introduction
- Semantic Web Processes Life cycle
- Web services Semantic Annotation
- Web services Discovery
- Semantic Process Composition
- Web service QoS
- Ontologies, Ontology Languages and Editors
- Projects/approaches: OWL-S, METEOR-S
- Conclusions

Our Focus (1)



- Supporting Web Processes on multi-enterprise and Web scale require addressing heterogeneity/integration, scalability, dynamic change and performance challenges
- Semantics is seen as the key enabler to address these challenges; Semantic Web Processes build upon Web Services and Semantic Web technologies
- This part of tutorial is about adding *semantics* to Web Services, and exploiting them in Web Process Lifecycle (Specification, Discovery, Composition, Execution)

  - Functional perspective takes form of process composition involving Web Service Discovery, handling semantic heterogeneity [modeling data i/o, state (pre/post condition) and function]
  - Operational perspective takes form of the research on QoS Specification for Web Services and Processes [modeling QoS and execution behavior]

Our Focus (2)

Semantics

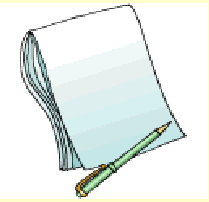
Web Processes

Web Process Composition

Execution

Web Process QoS

Web Services

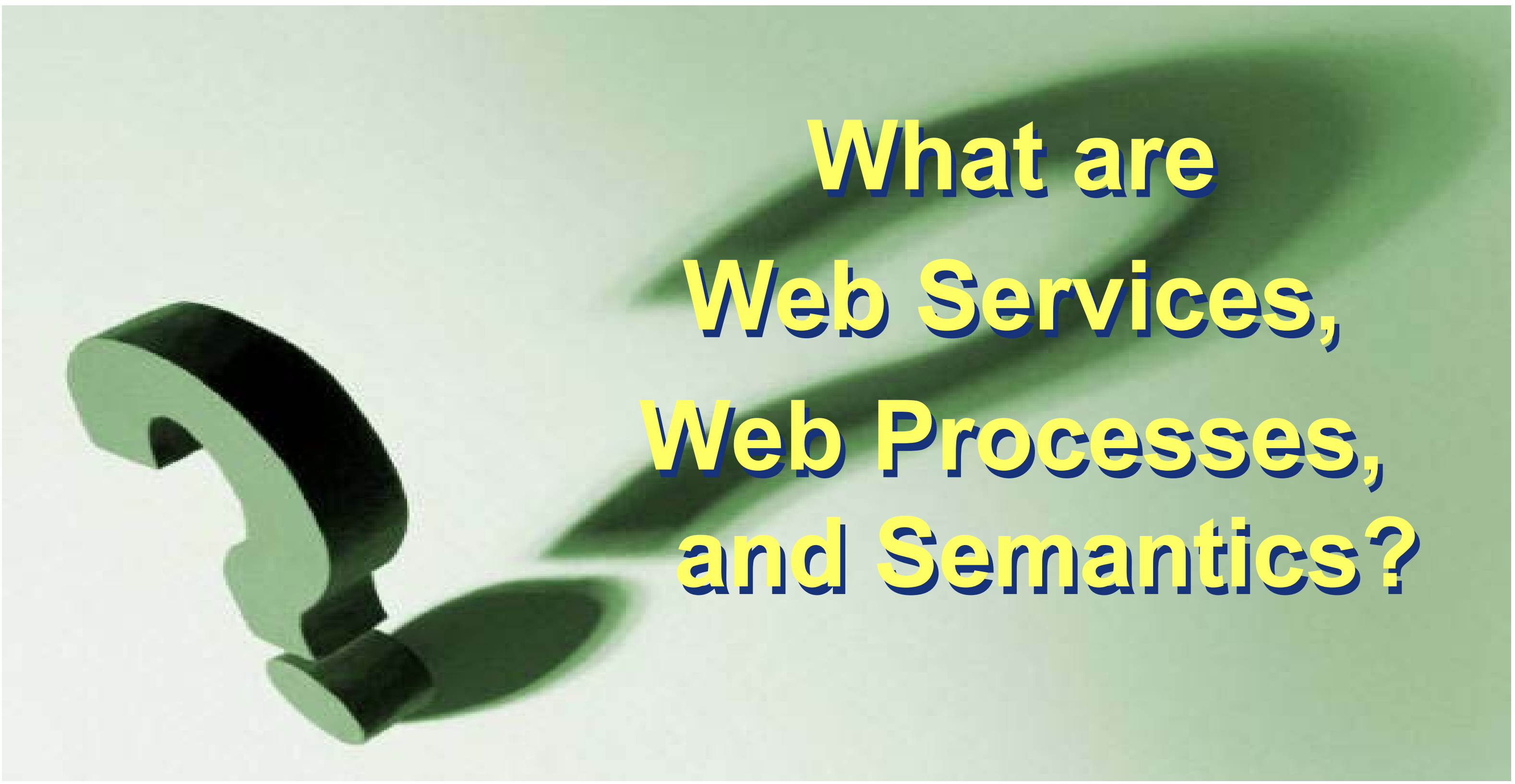


Web Service Annotation

Web Service Discovery

Web Service QoS

The Basics



What are
Web Services,
Web Processes,
and Semantics?

Web Services: Definition

Web Services

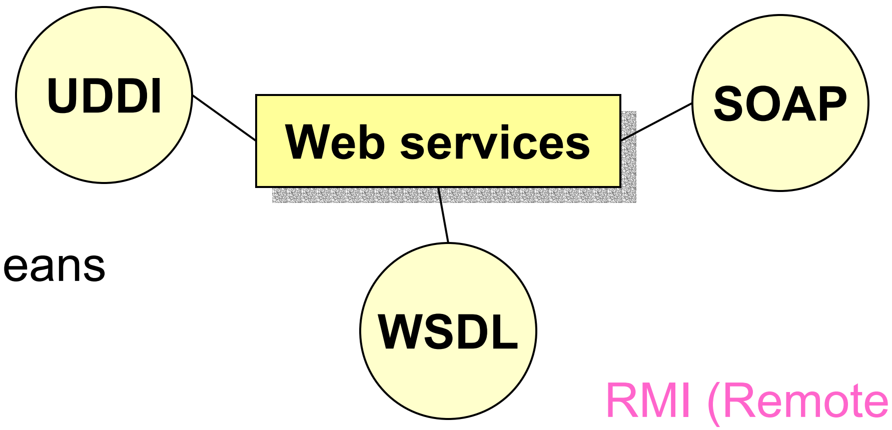
"Web services are a new breed of Web application. They are **self-contained**, **self-describing**, modular applications that can be **published**, **located**, and **invoked** across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes. …

Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service."

IBM web service tutorial

Why Web Services?

Web Services



Jini

Enterprise Java Beans

RMI (Remote Method Invocation)

Microsoft DCOM

CORBA (Common Object Request Broker Architecture)

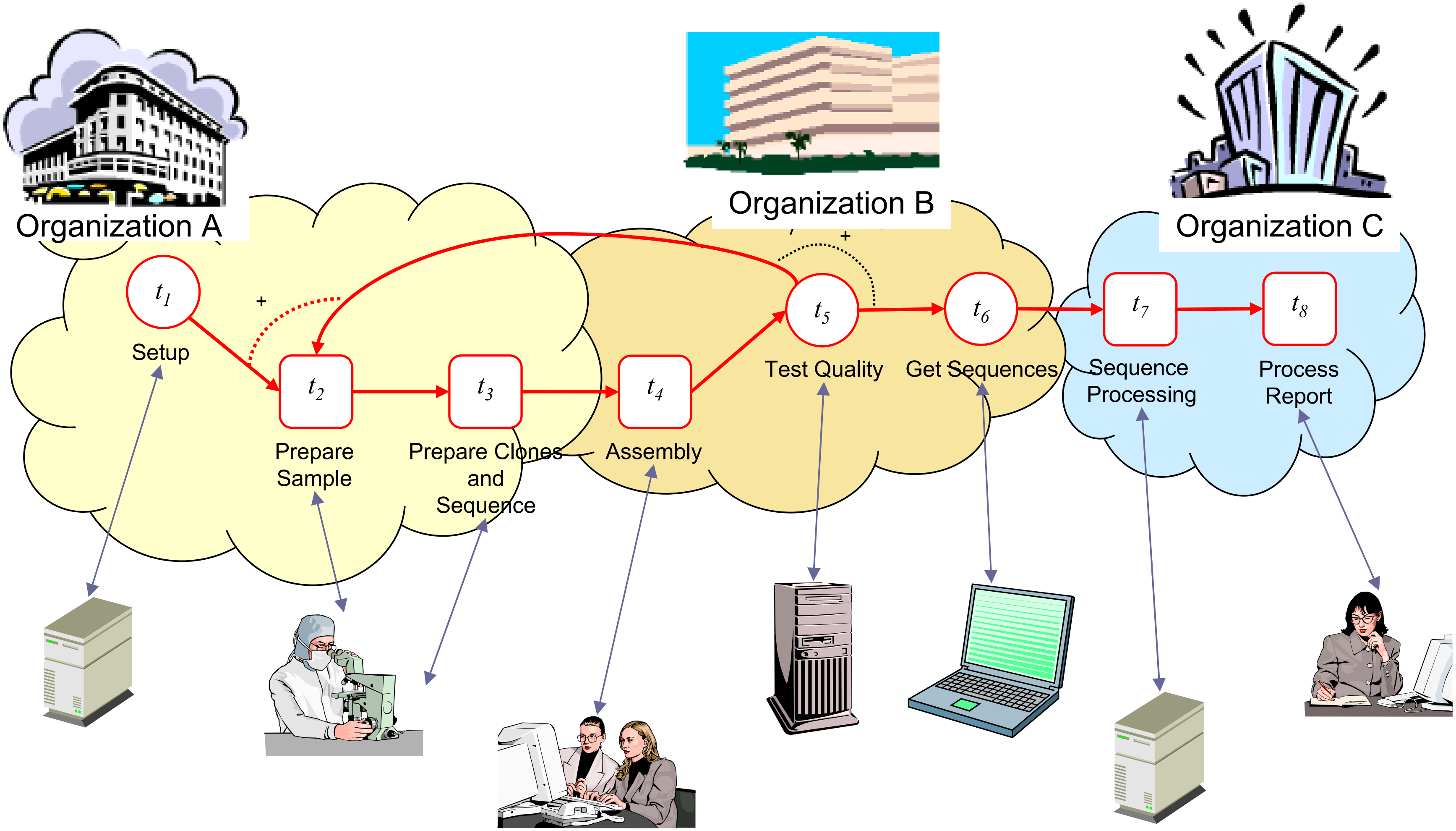
Open Software Foundation DCE (Distributed Computing Environment)

Sun ONC/RPC (Open Network Computing)

IP, UDP, TCP

Web Process
An Example

Web Processes



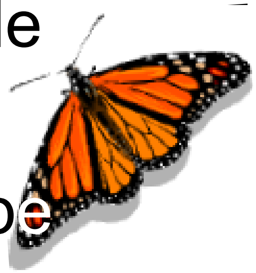
What are Web Processes (1)?

- **Web Processes** are next generation workflow technology to facilitate the interaction of organizations with markets, competitors, suppliers, customers etc. supporting enterprise-level and core business activities
  - encompass the ideas of both intra and inter organizational workflow.
  - created from the composition of Web services
  - can use BPEL4WS to represent composition, but how to get there?

What are Web Processes ? (2)

Web Processes

- Web processes describe how Web services are connected to create reliable and dependable business solutions

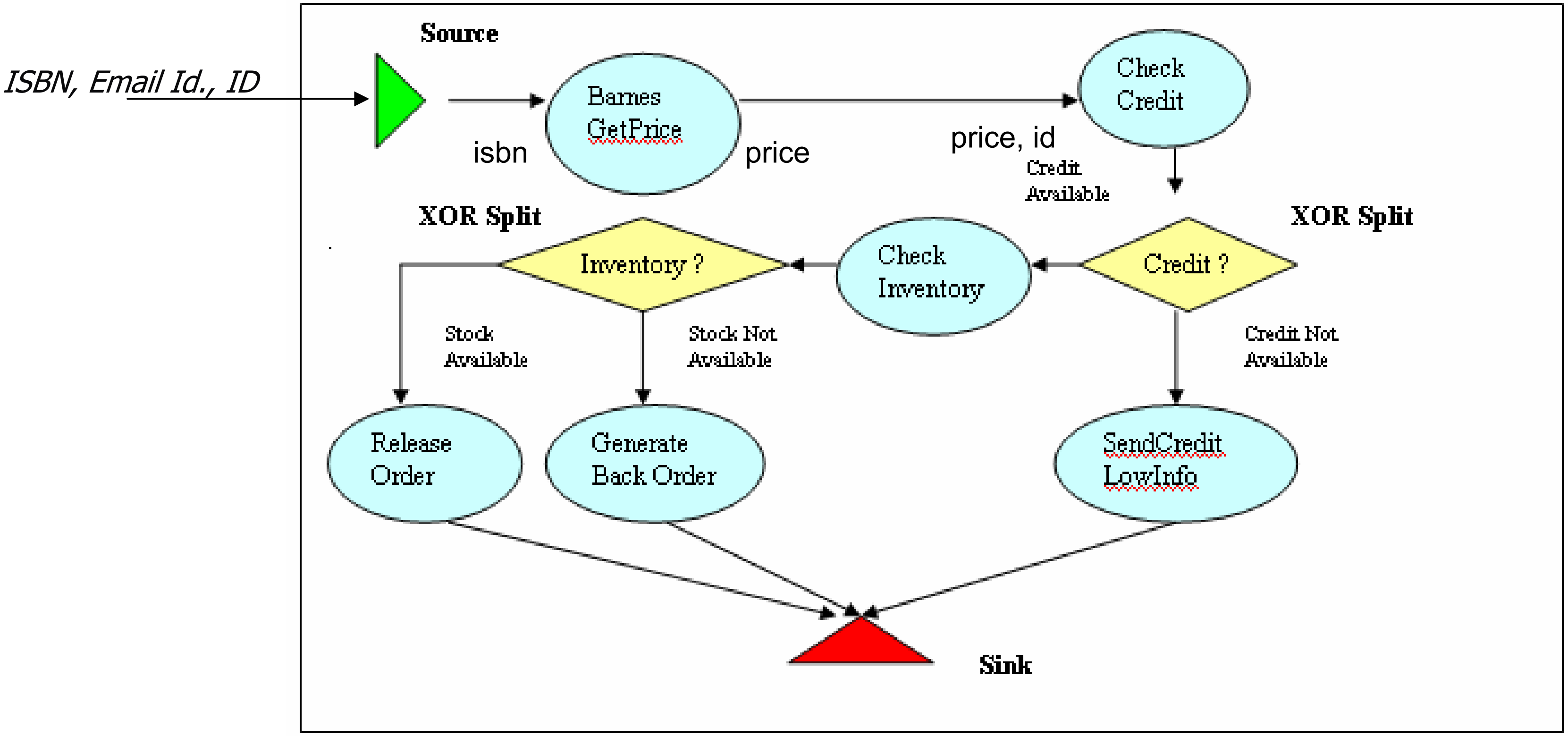


- Web processes allow businesses to describe sophisticated processes that can both consume and provide Web services
- The role of Web processes within the enterprise is to simplify the integration of business and application processes across technological and corporate domains

Web Process
An Example

Web Processes

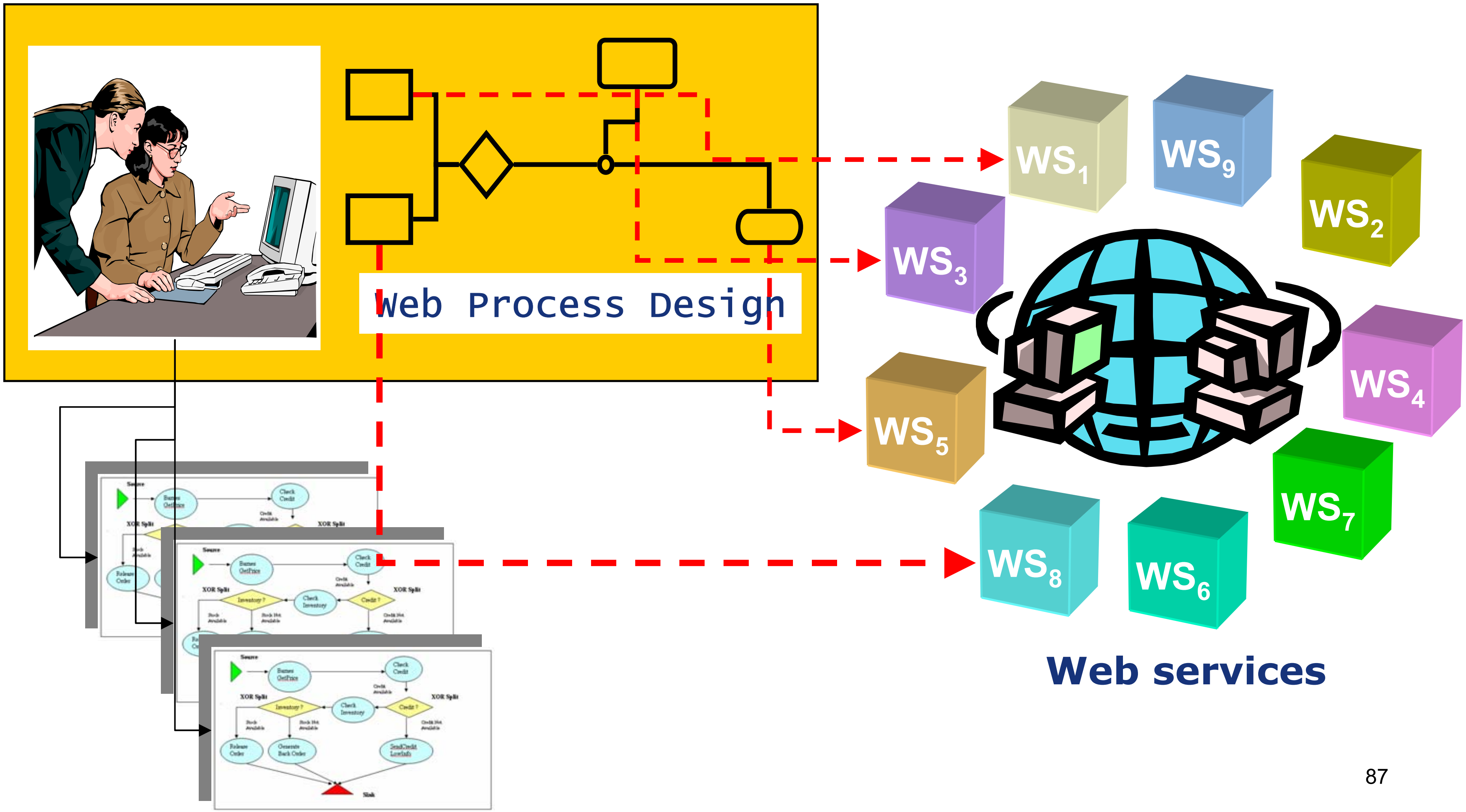
- Graphical example of a web process



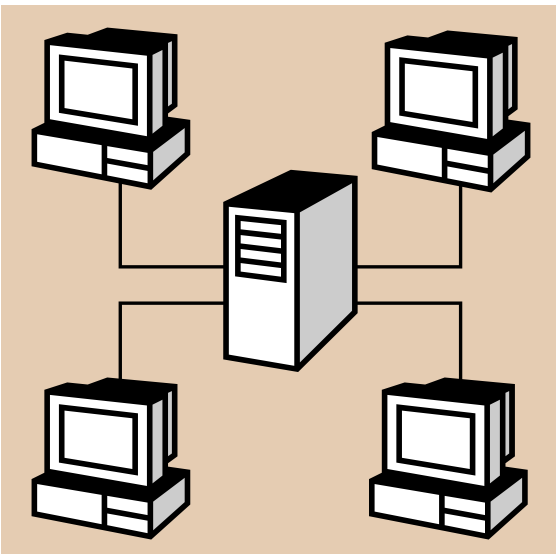
The BarnesBookPurchase process

Web Processes Composition

Web Processes



Globalization of Processes



Workflows

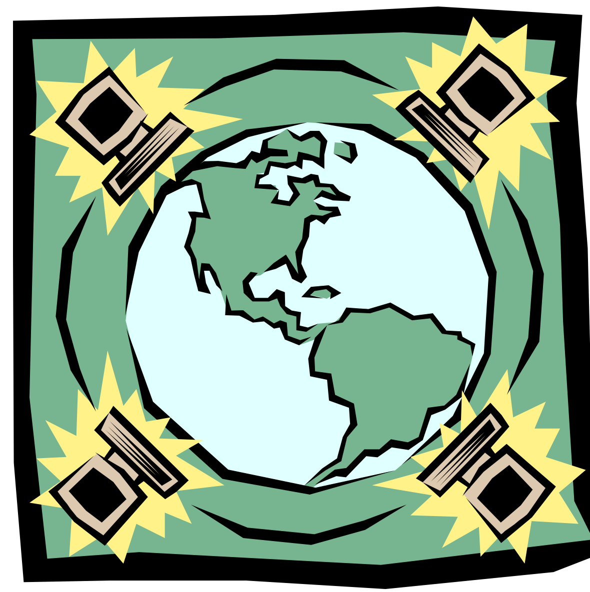
Enterprise

B2B

Distributed Workflows

Inter-Enterprise

E-Services



Web Processes

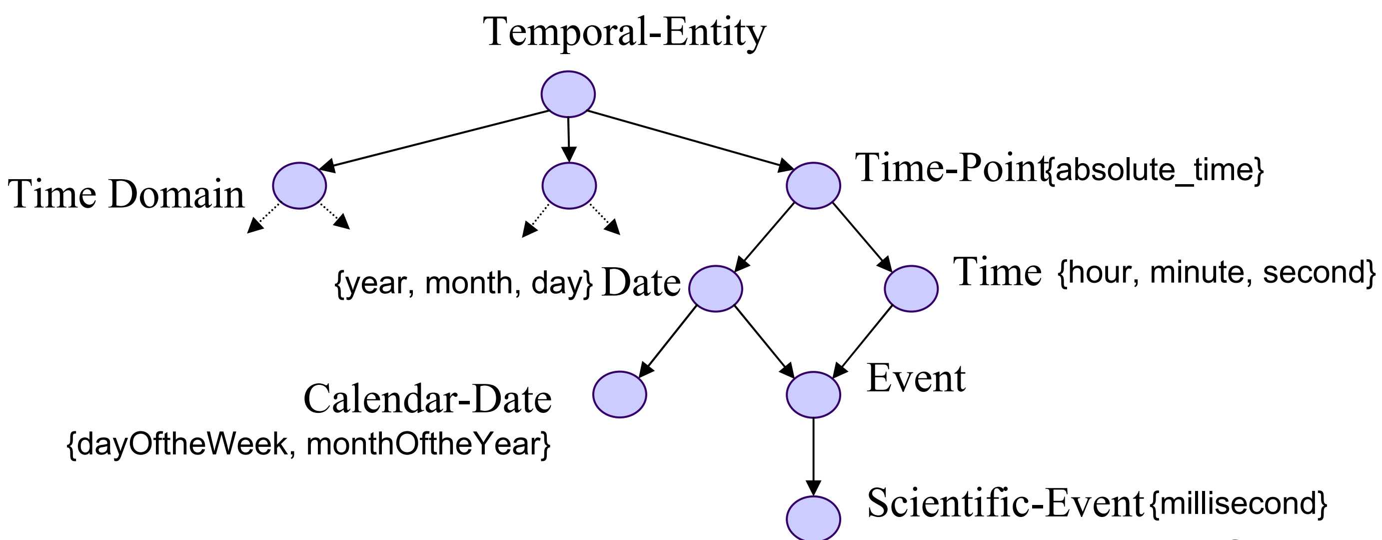
Global

BIG Challenges

- **Heterogeneity and Autonomy**
  - Syntactic, semantic and pragmatic
  - Complex rules/regulations related to B2B and e-commerce interactions
  - Solution: Machine processable descriptions
- **Dynamic** nature of business interactions
  - Demands: Efficient Discovery, Composition, etc.
- **Scalability** (Enterprises → Web)
  - Needs: Automated service discovery/selection and composition

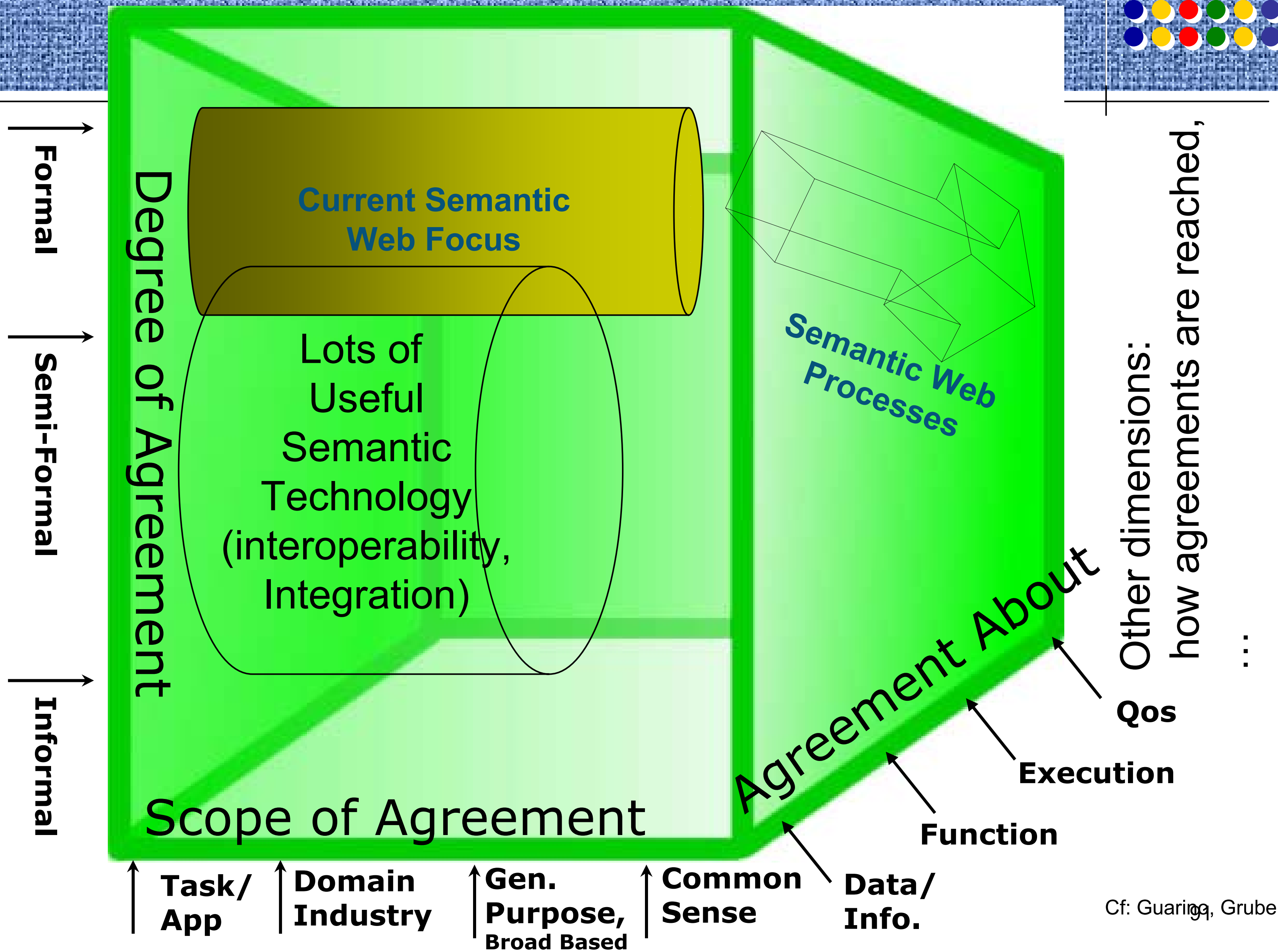
Proposition: Semantics is the most important enabler to address these challenges.

Semantics, Ontologies, Semantic Web Processes



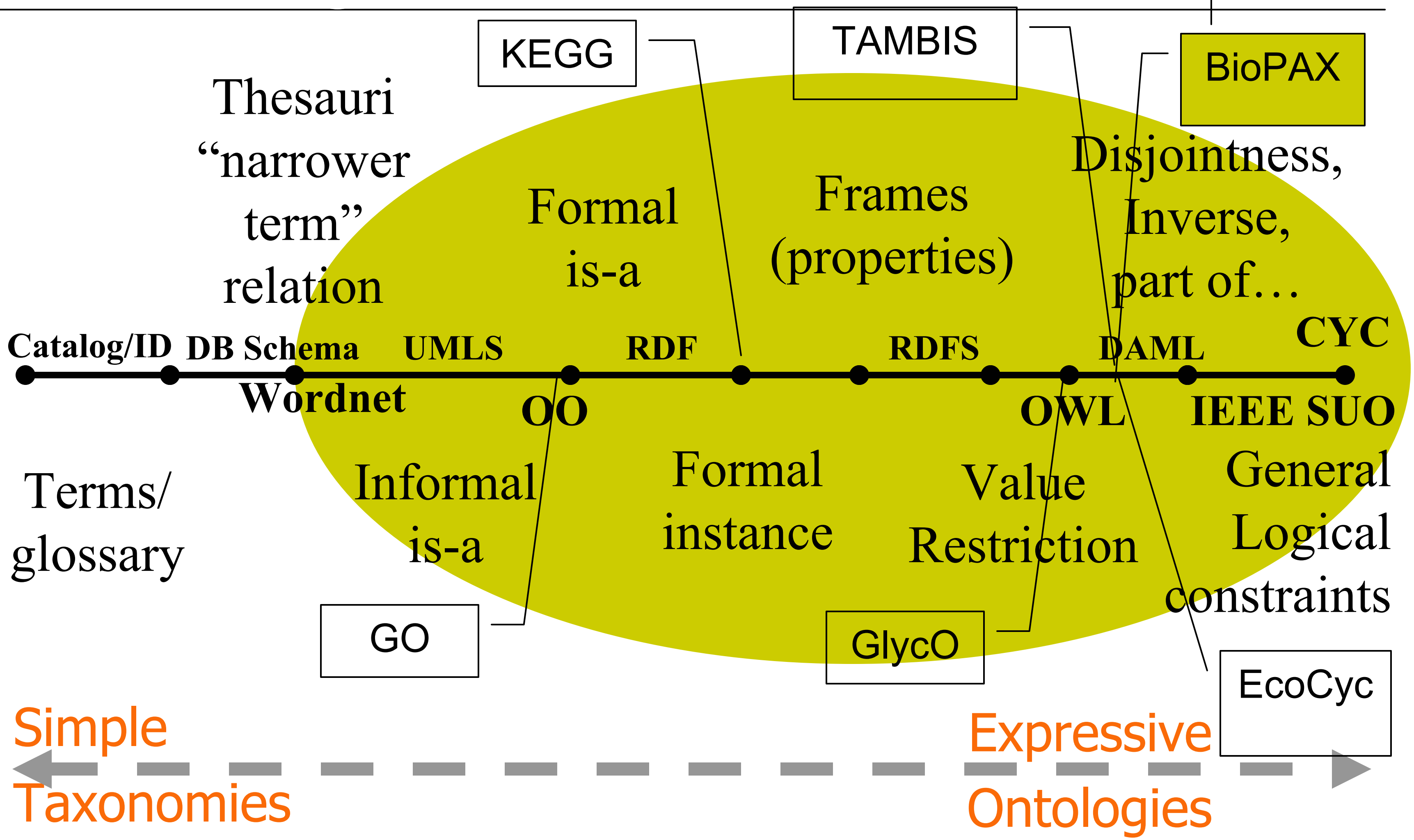
- When Web services and other descriptions that define a Web process are semantically described, we may call such process as **Semantic Web Processes.**
- An ontology provides semantic grounding. It includes a **vocabulary of terms**, and some **specification of their meaning**.
- The goal is to create an **agreed-upon vocabulary** and semantic structure for exchanging information about that domain.

Broad Scope of Semantic (Web) Technology



Cf: Guarino, Gruber

Knowledge Representation and Ontologies



Ontology Dimensions After McGuinness and Finin

WordNet

a lexical database for the English language

- Approximately 95 000 different word forms
- English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept.
- Different relations link the synonym sets.
- Create a lexical thesaurus (not a dictionary) which models the lexical organization used by humans.
- http://www.cogsci.princeton.edu/~wn/

Using WordNet

WordNet a lexical database for the English language

4 senses of eagle

Sense 1

eagle, bird of Jove -- (any of various large keen-sighted diurnal birds of prey noted for their broad wings and strong soaring flight)

=> **bird of prey, raptor, raptorial bird** -- (any of numerous carnivorous birds that hunt and kill other animals)

=> **bird** -- (warm-blooded egg-laying vertebrates characterized by feathers and forelimbs modified as wings)

=> **vertebrate, craniate** -- (animals having a bony or cartilaginous skeleton with a segmented spinal column and a large brain enclosed in a skull or cranium)

=> **chordate** -- (any animal of the phylum Chordata having a notochord or spinal column)

=> **animal, animate being, beast, brute, creature, fauna** -- (a living organism characterized by voluntary movement)

=> **organism, being** -- (a living thing that has (or can develop) the ability to act or function independently)

=> **living thing, animate thing** -- (a living (or once living) entity)

=> **object, physical object** -- (a tangible and visible entity; an entity that can cast a shadow; "it was full of rackets, balls and other objects")

=> **entity, physical thing** -- (that which is perceived or known or inferred to have its own physical existence (living or nonliving))

Using WordNet

a lexical database for the English language

Sense 2

eagle -- ((in golf) a score of two strokes under par on a golf hole)

=> **score** -- (a number that expresses the accomplishment of a team or an individual in a game or contest; "the score was 7 to 0")

=> **number** -- (a concept of quantity derived from zero and units; "every number has a unique position in the sequence")

=> **definite quantity** -- (a specific measure of amount)

=> **measure, quantity, amount, quantum** -- (how much there is of something that you can measure)

=> **abstraction** -- (a general concept formed by extracting common features from specific examples)

Semantic Web Process Life Cycle

Semantics for Web Processes

- Data/Information Semantics
  - **What:** Formal definition of data in input and output messages of a web service
  - **Why:** for discovery and interoperability
  - **How:** by annotating *input/output data* of web services using ontologies

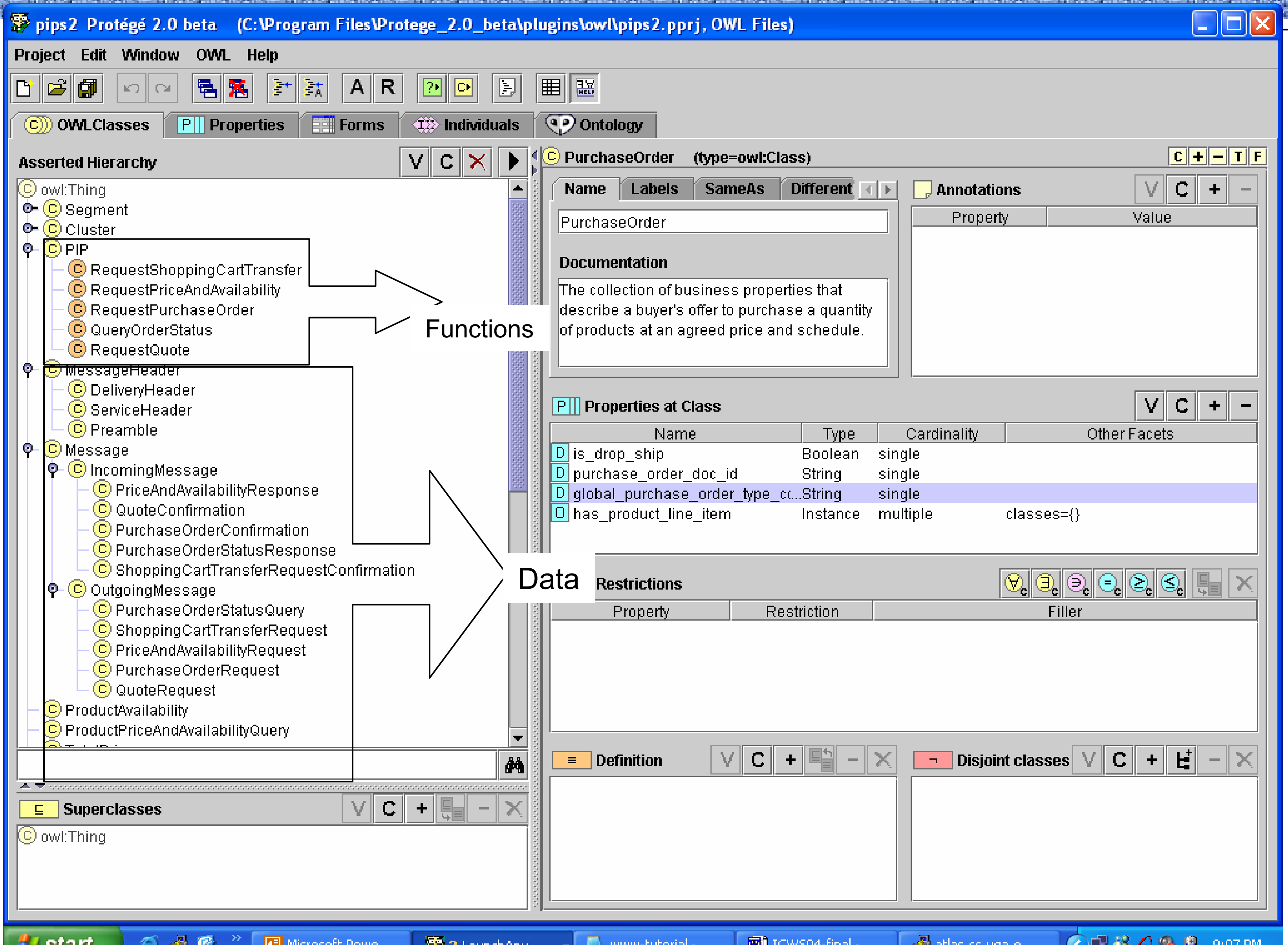
- Functional/Operational Semantics
  - Formally representing capabilities of web service
  - for discovery and composition of Web Services
  - by annotating *operations* of Web Services as well as provide *preconditions* and *effects*; Annotating *TPA/SLA (future work)*

- Execution Semantics
  - Formally representing the execution or flow of a services in a process or operations in a service
  - for analysis (verification), validation (simulation) and execution (exception handling) of the process models
  - using *State Machines*, *Petri nets*, *activity diagrams* etc.

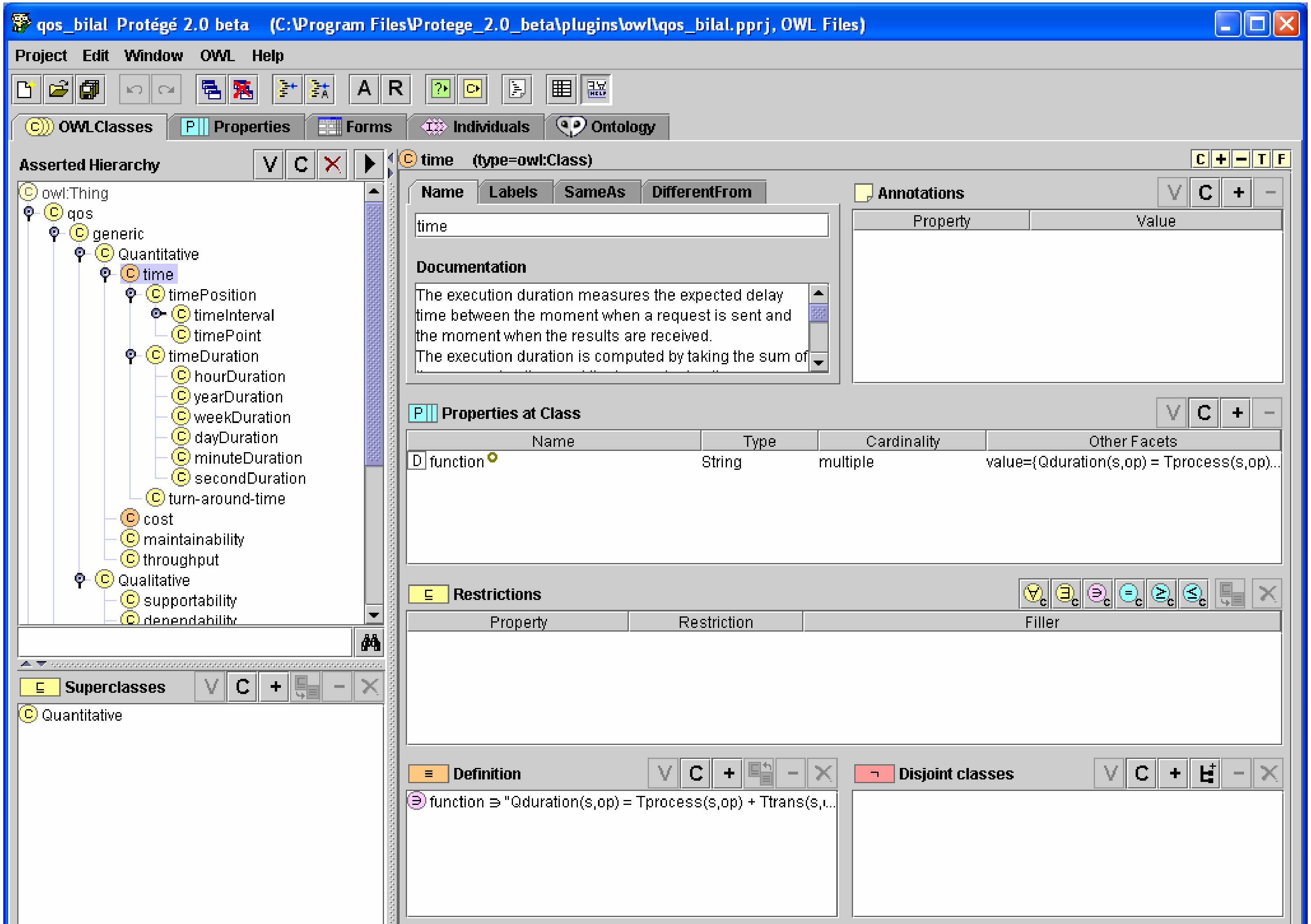
- QoS Semantics
  - Formally describing operational metrics of a web service/process
  - To select the most suitable service to carry out an activity in a process
  - using *QoS model* [Cardoso and Sheth, 2002] for web services

Data and Functional Ontology

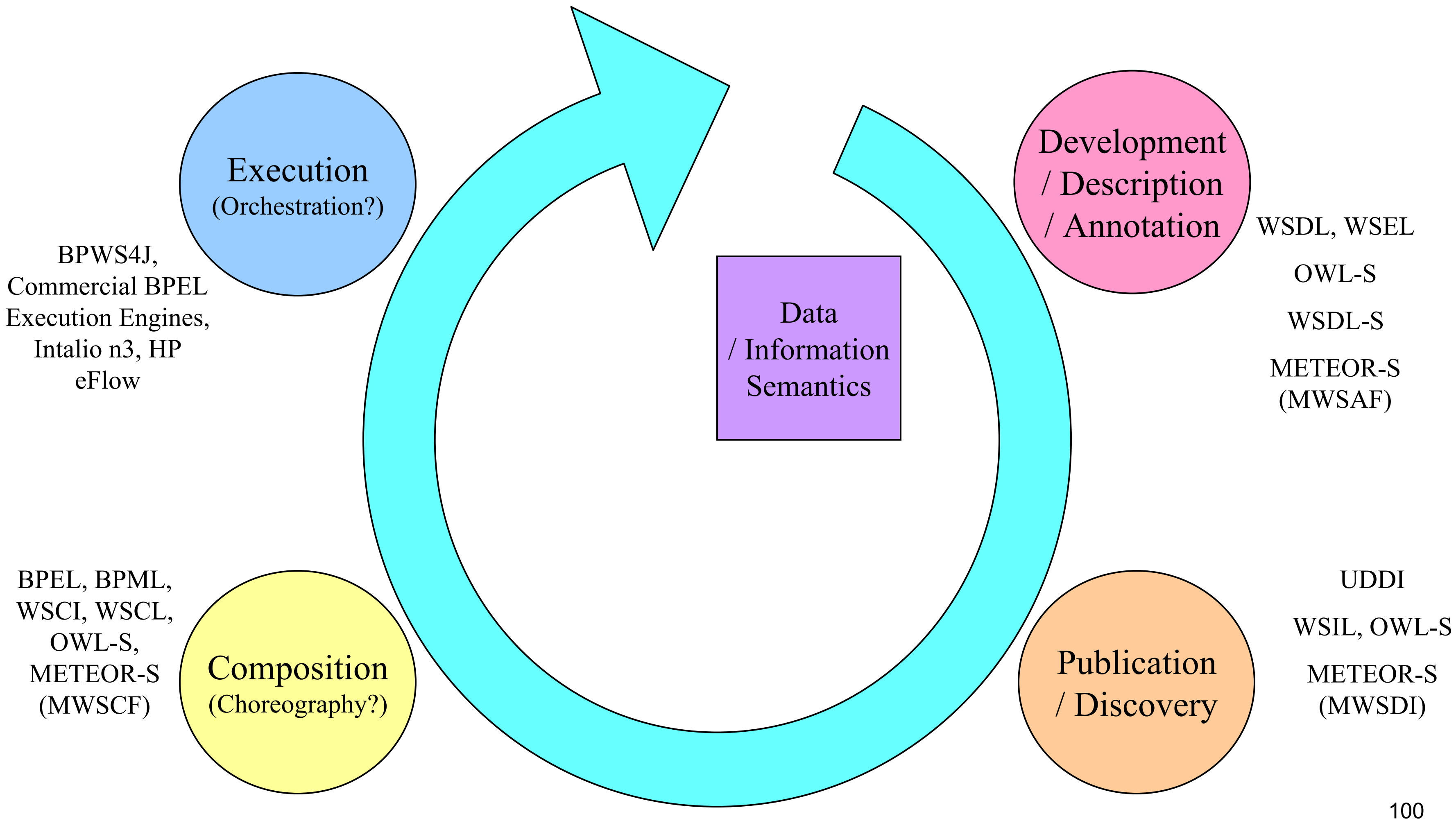
an example based on Rosettanet



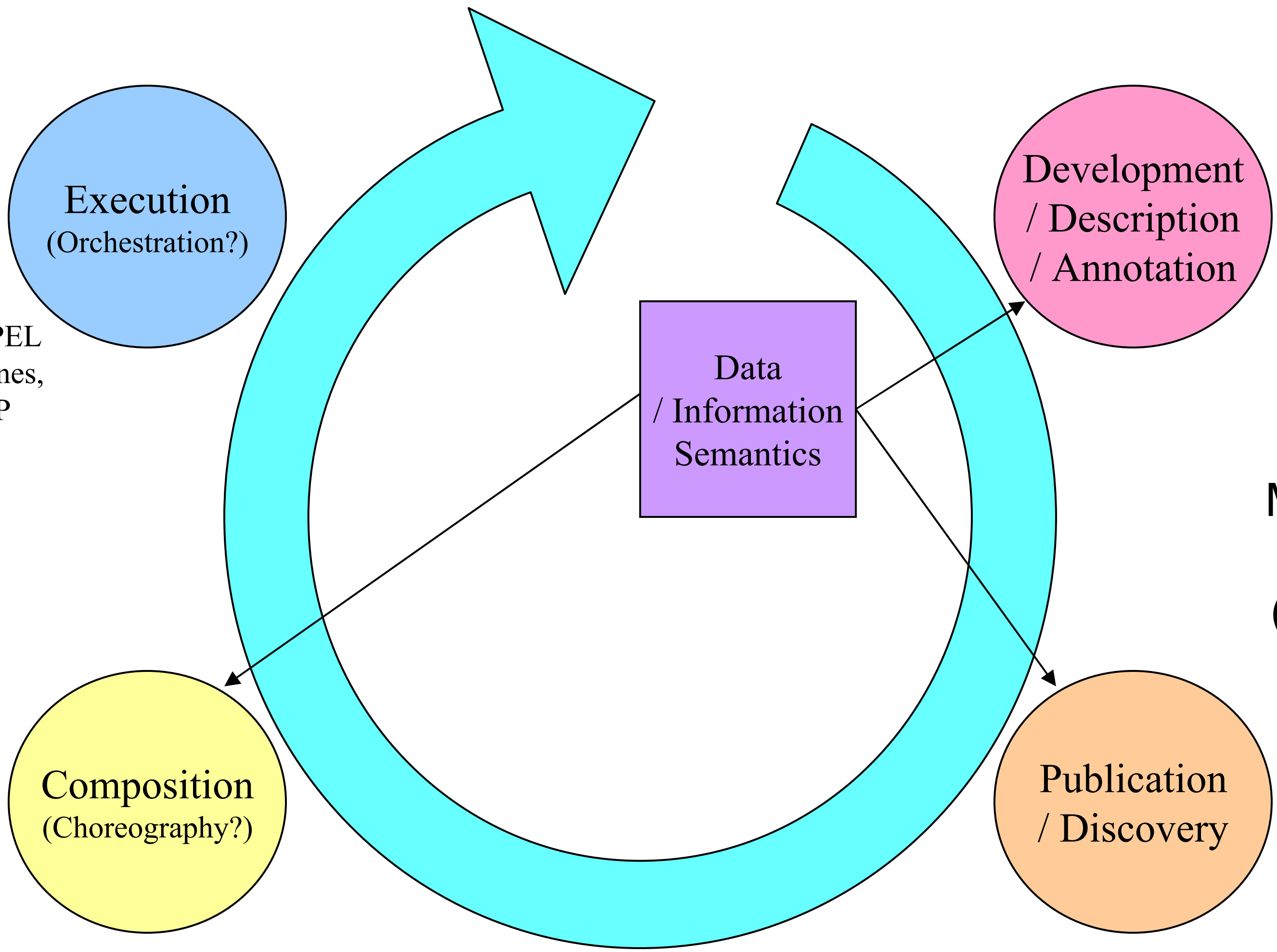
QoS Ontology in METEOR-S



Semantics for Web Process Life-Cycle



Semantics for Web Process Life-Cycle



BPWS4J, Commercial BPEL Execution Engines, Intalio n3, HP eFlow

WSDL, WSEL OWL-S WSDL-S METEOR-S (MWSAF)

BPEL, BPML, WSCI, WSCL, OWL-S, METEOR-S (SCET,SPTB)

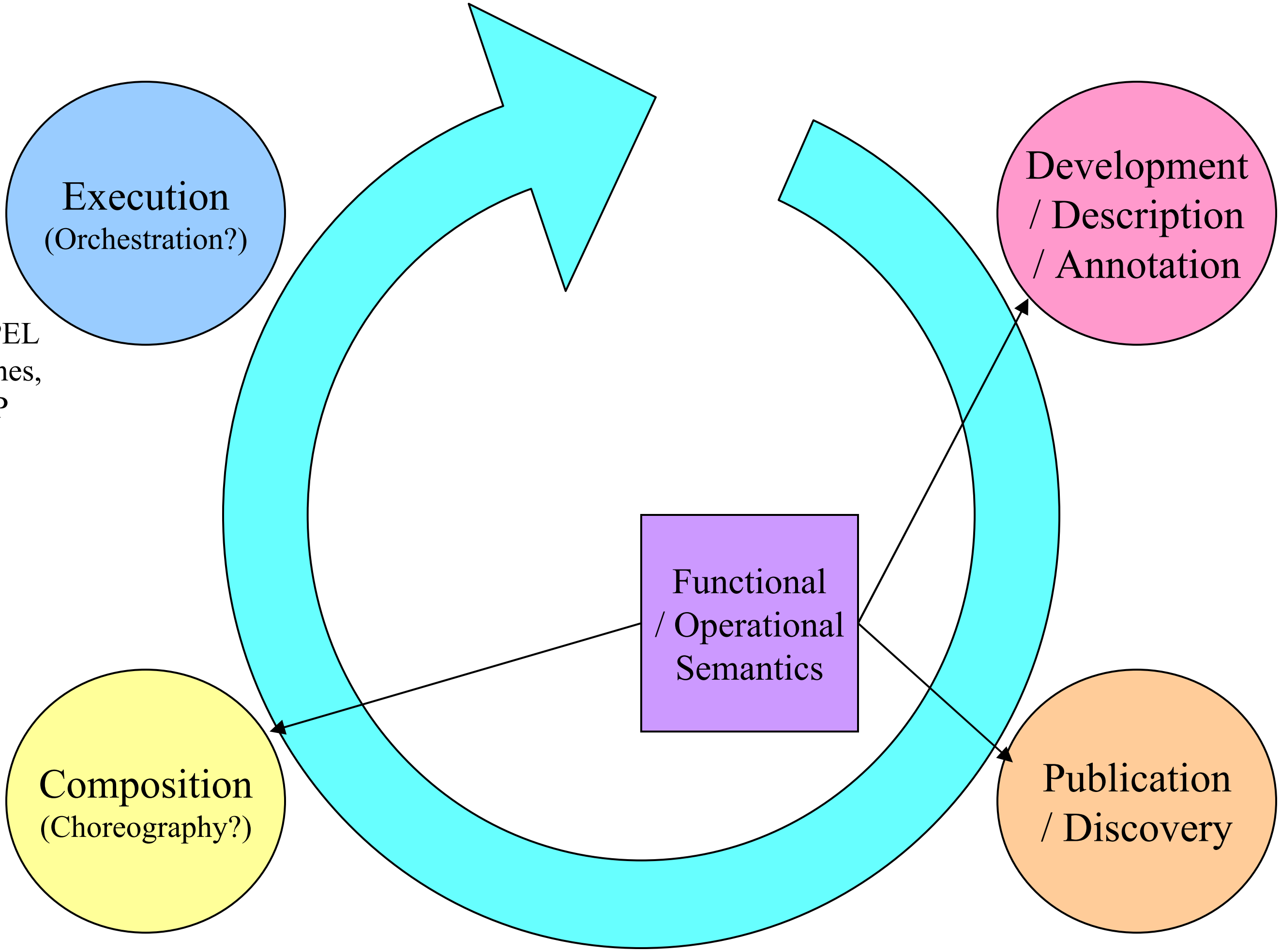
UDDI

WSIL, OWL-S

METEOR-S (P2P model of registries)

Semantics for Web Process Life-Cycle

BPWS4J, Commercial BPEL Execution Engines, Intalio n3, HP eFlow



WSDL, WSEL

OWL-S

WSDL-S

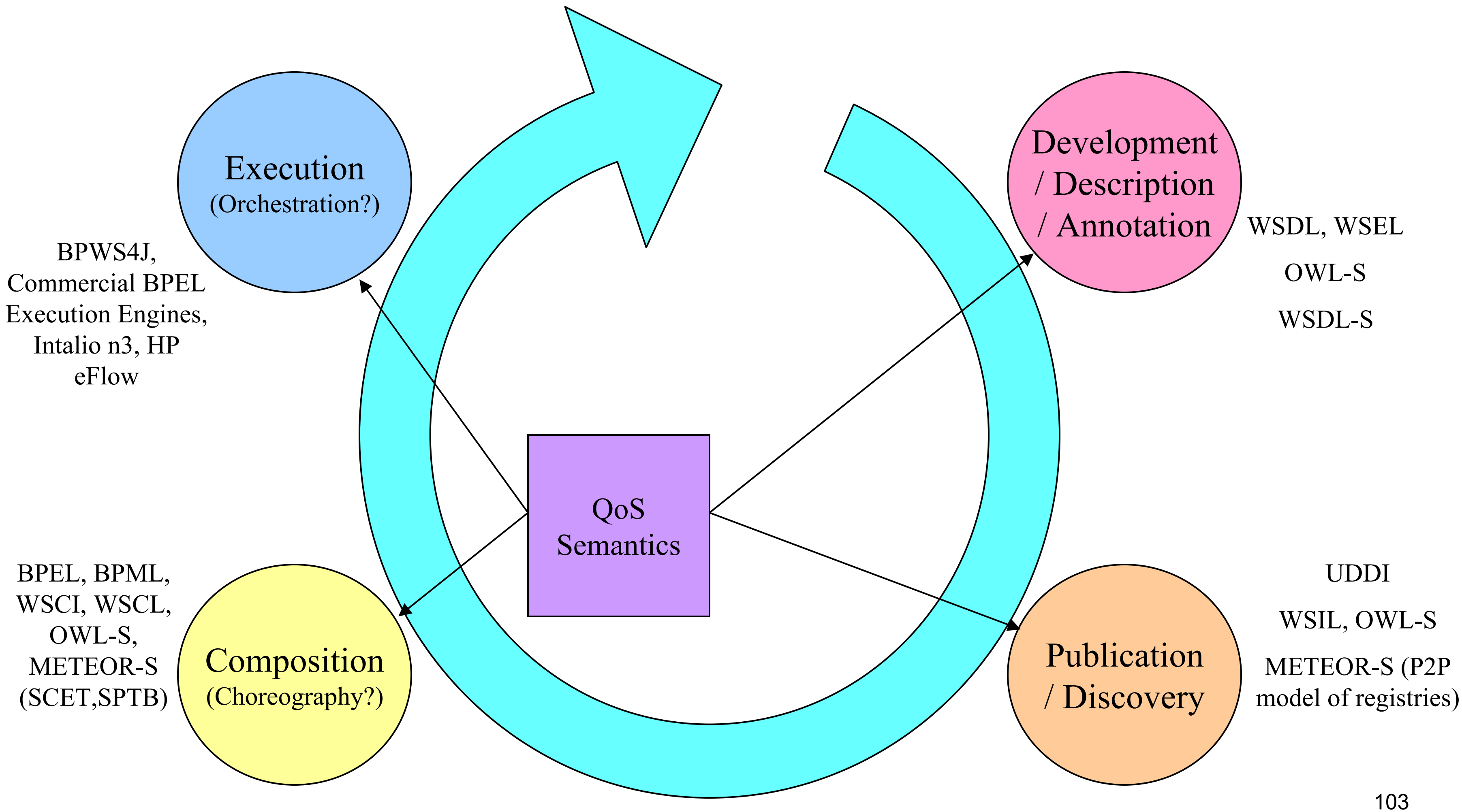
BPEL, BPML, WSCI, WSCL, OWL-S, METEOR-S (SCET,SPTB)

UDDI

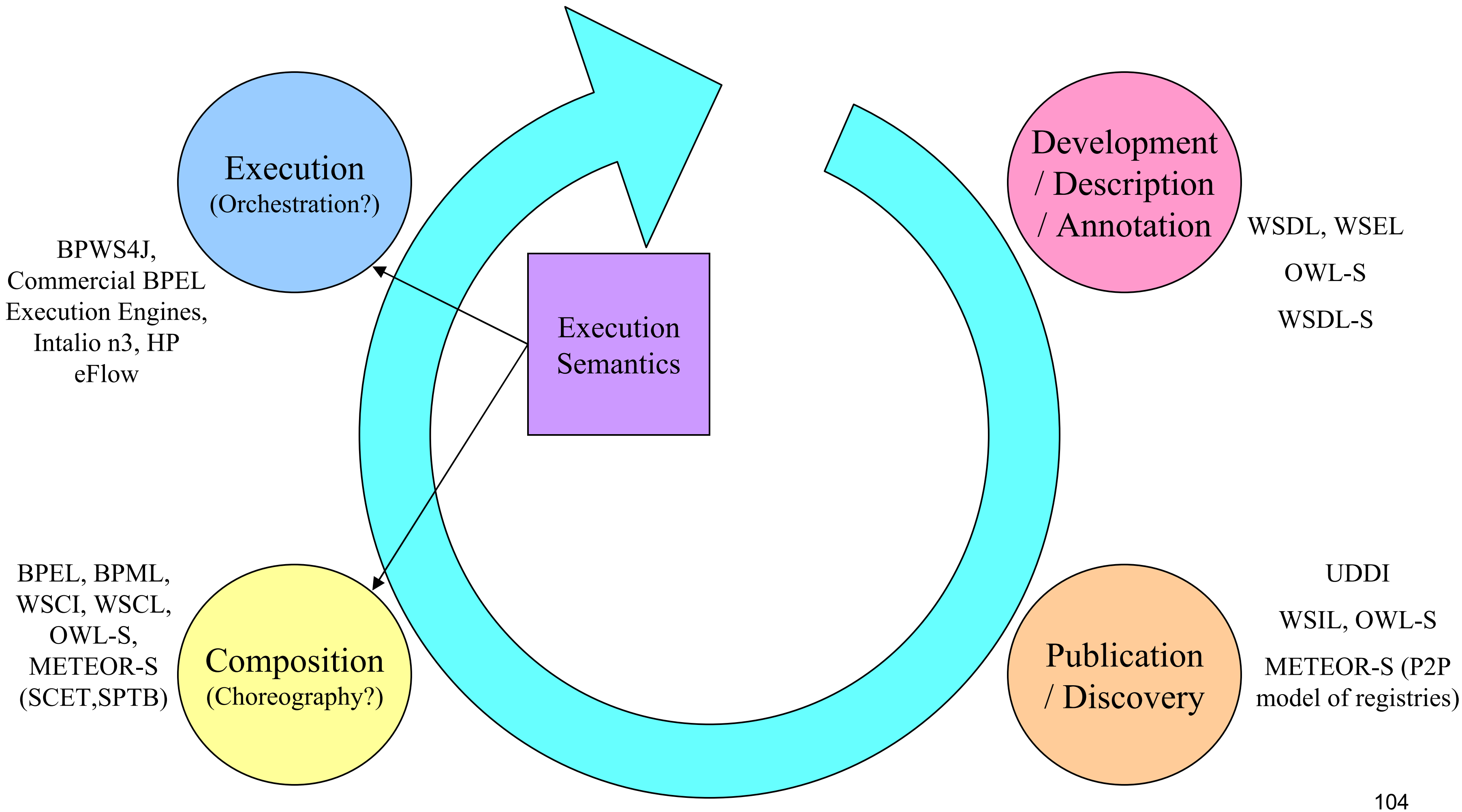
WSIL, OWL-S

METEOR-S (P2P model of registries)

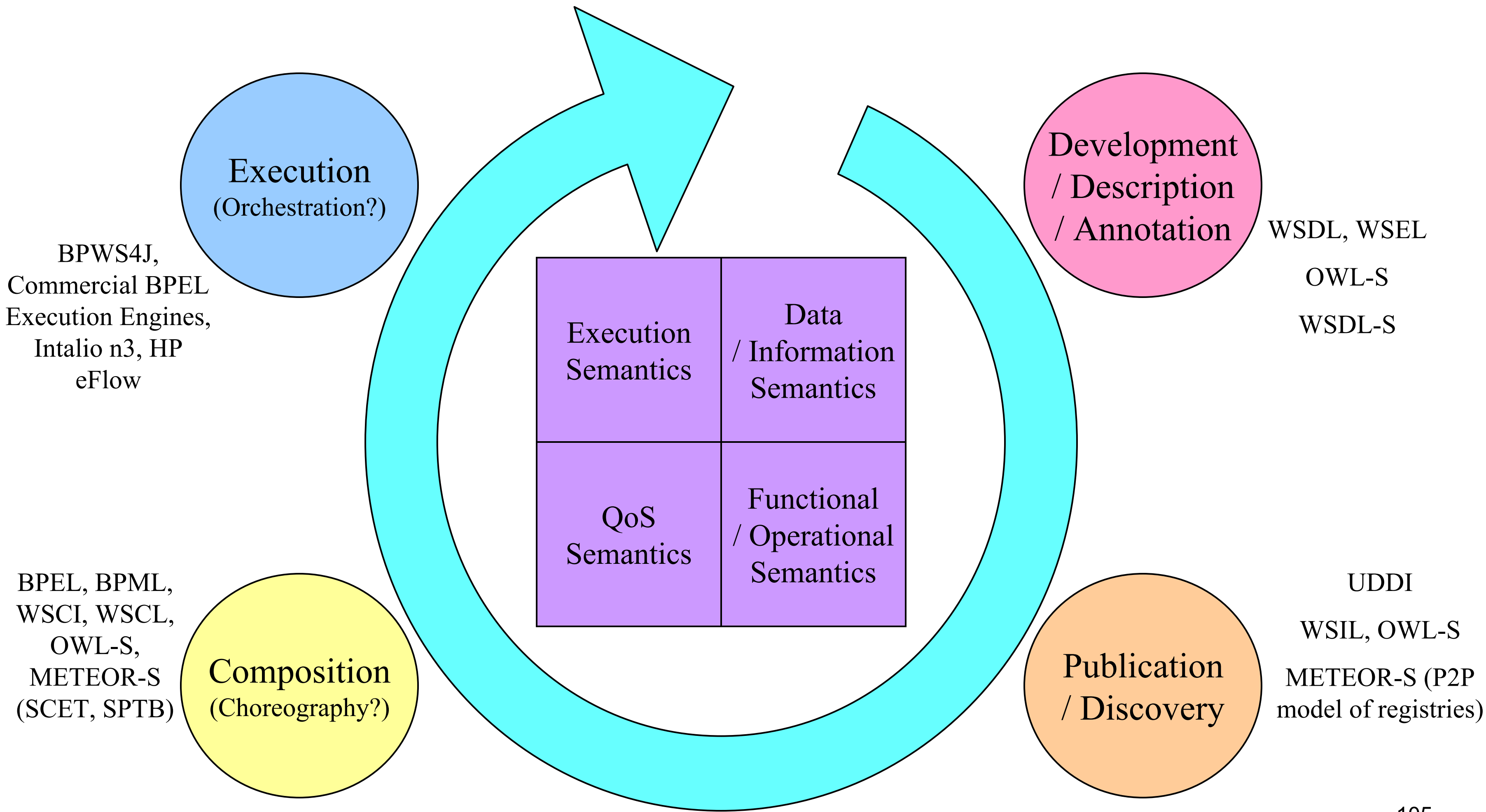
Semantics for Web Process Life-Cycle



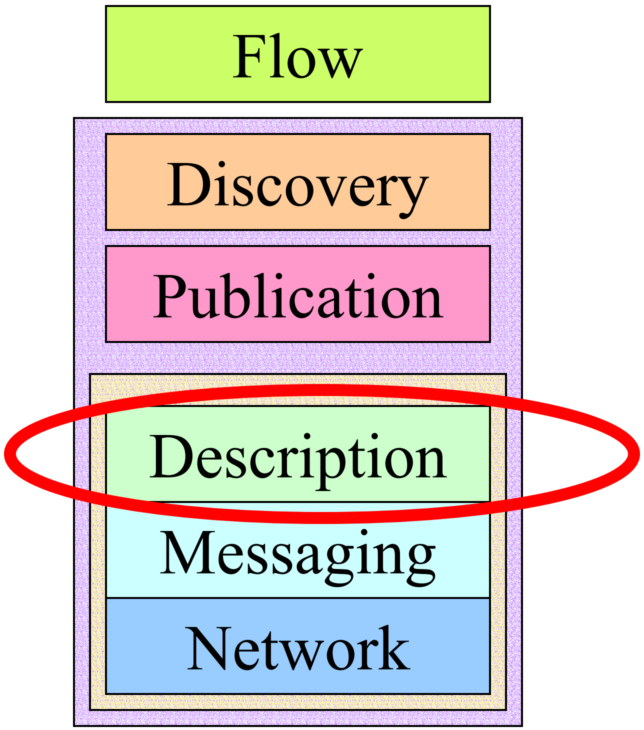
Semantics for Web Process Life-Cycle



Semantics for Web Process Life-Cycle



Semantics at Different Layers



Description Layer:

Why:

- Unambiguously understand the functionality of the services and the semantics of the operational data

How:

- Using Ontologies to semantically annotate WSDL constructs (conforming to extensibility allowed in WSDL specification version 1.2/2.0)
  - WSDL-S : Incorporate all types of semantics in the service description

Present scenario:

- WSDL descriptions are mainly syntactic (provides operational information and not functional information)
- Semantic matchmaking is not possible

WSDL-S

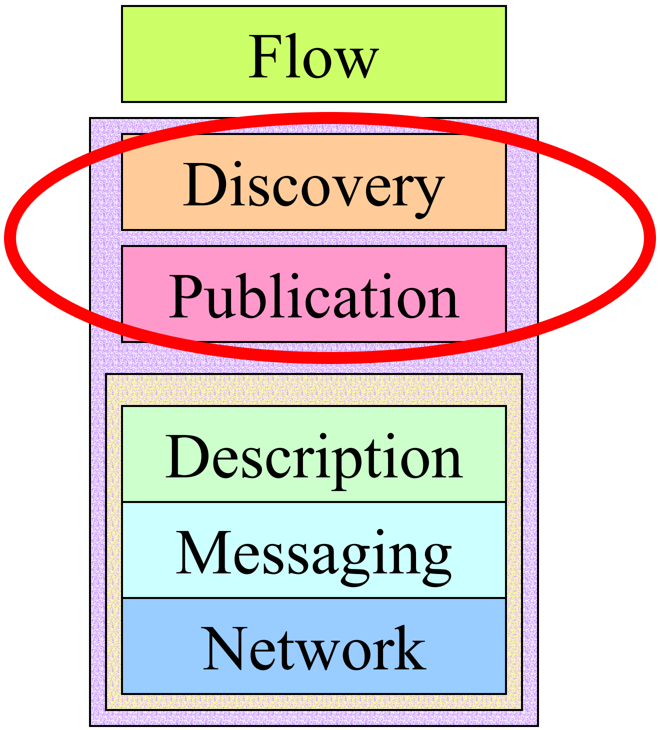
```
<?xml version="1.0" encoding="UTF-8"?>
<definitions
        name = "BatterySupplier"
        targetNamespace = "http://lsdis.cs.uga.edu/meteor/BatterySupplier.wsdl20"
        xmlns = "http://www.w3.org/2004/03/wsdl"
        xmlns:tns = "http://lsdis.cs.uga.edu/BatterySupplier.wsdl20"
        xmlns:rosetta = " http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/pips.owl "
xmlns:mep=http://www.w3. rosetta:PurchaseOrderStatusResponse org/TR/wsdl20-patterns>
  <interface name = "BatterySupplierInterface" description = "Computer PowerSupply Battery Buy Quote Order
Status "
        domain="naics:Computer and Electronic Product Manufacturing" >

    <operation name = "getQuote"  pattern = "mep:in-out"  action = "rosetta:#RequestQuote" >
          <input messageLabel = "qRequest" element = "rosetta:#QuoteRequest" />
        <output messageLabel = "quote" element  = "rosetta:#QuoteConfirmation"  />
    </operation>

    <operation name = "placeOrder"  pattern = "mep:in-out"  action = "rosetta:#RequestPurchaseOrder" >
          <input messageLabel = "order" element = "rosetta:#PurchaseOrderRequest" />
        <output messageLabel = "orderConfirmation" element  = "rosetta:#PurchaseOrderConfirmation" />
          <exception element = "rosetta:#DiscountinuedItemException" />
        <pre condition = " order.PurchaseOrder.PurchaseOrderLineItem.RequestedQuantity  > 7" />
    </operation>
    <operation name = "checkStatus" pattern="mep:in-out" action = "rosetta:#QueryOrderStatus" >
          <input messageLabel = "statusQuery" element  = "rosetta:#PurchaseOrderStatusQuery" />
          <output messageLabel = "status" element  = "rosetta:#PurchaseOrderStatusResponse" />
          <exception element = "rosetta:#OrderNumberInvalidException" />
    </operation>
  </interface>
```

Function from Rosetta Net Ontology

Data from Rosetta Net Ontology

Semantics at Different Layers

(contd..)



Publication and Discovery Layers:

Why:

- Enable scalable, efficient and dynamic publication and discovery (machine processable / automation)

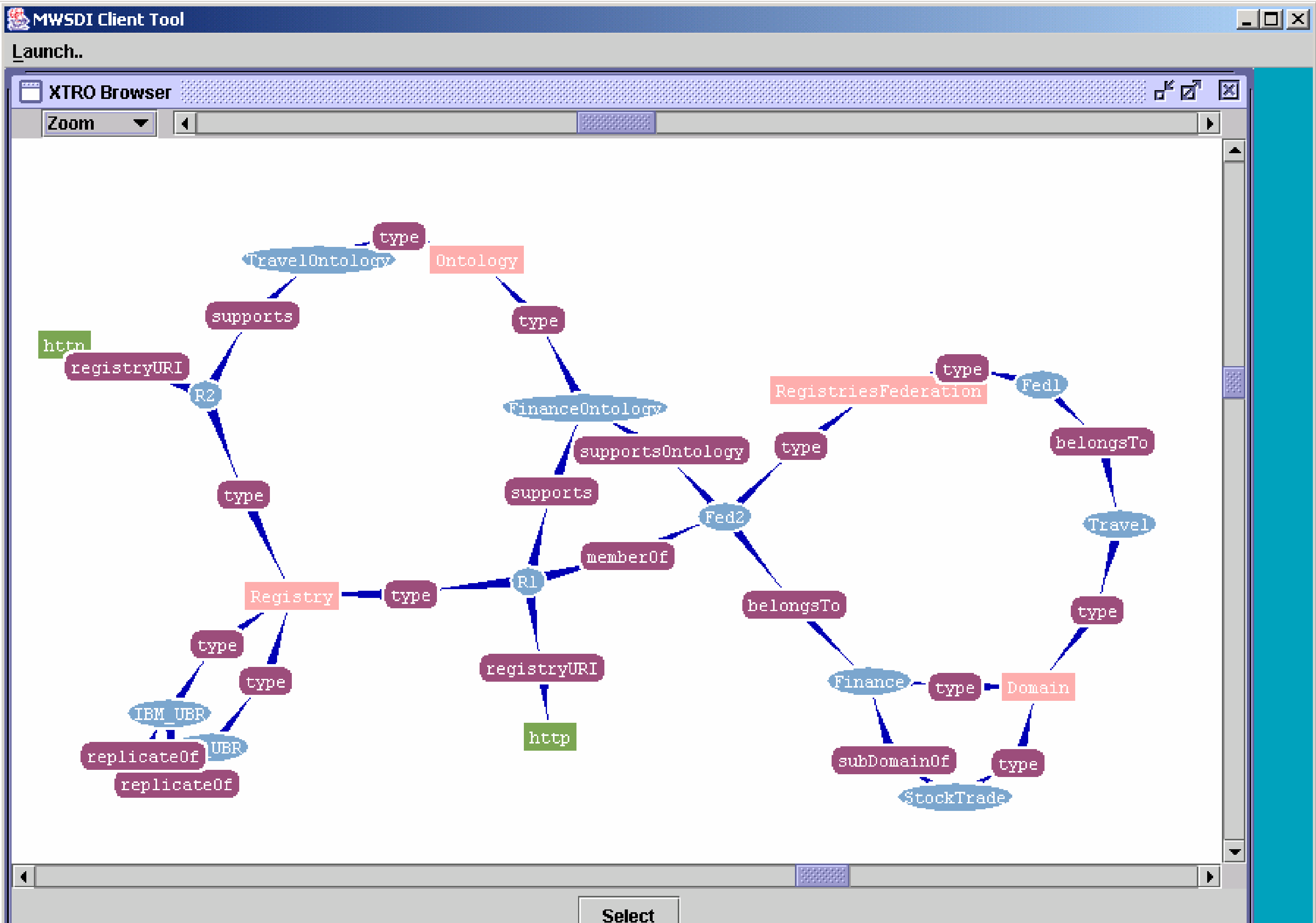
How:

- Use of ontology to categorize registries based on domains and characterize them by maintaining the
    1. properties of each registry
    2. relationships between the registries
- Capturing the WSDL annotations in UDDI

Present scenario:

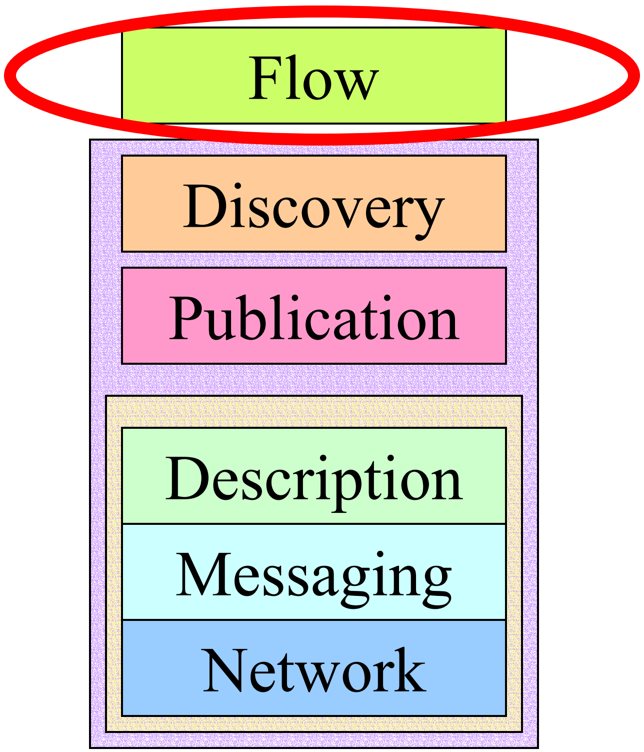
- Suitable for simple searches (like services offered by a provider, services that implement an interface, services that have a common technical fingerprint etc.)
- Categories are too broad
- Automated service discovery (based on functionality) and selecting the best suited service is not possible

MWSDI


MWSDI Client Tool
Launch..
XTRO Browser
Zoom
type
TravelOntology
Ontology
supports
type
http
registryURI
R2
FinanceOntology
type
RegistriesFederation
Fed1
supportsOntology
type
belongsTo
type
supports
Fed2
Travel
memberOf
Registry
type
R1
belongsTo
type
type
type
registryURI
Finance
type
Domain
IBM_UBR
UBR
http
subDomainOf
type
replicateOf
replicateOf
StockTrade
Select

Semantics at Different Layers

(contd..)



Flow Layer:

Why:

- Design (composition), analysis (verification), validation (simulation) and execution (exception handling) of the process models
- To employ mediator architectures for automated composition, control flow and data flow based on requirements
- To employ user interface to capture template requirements and generate template based on that

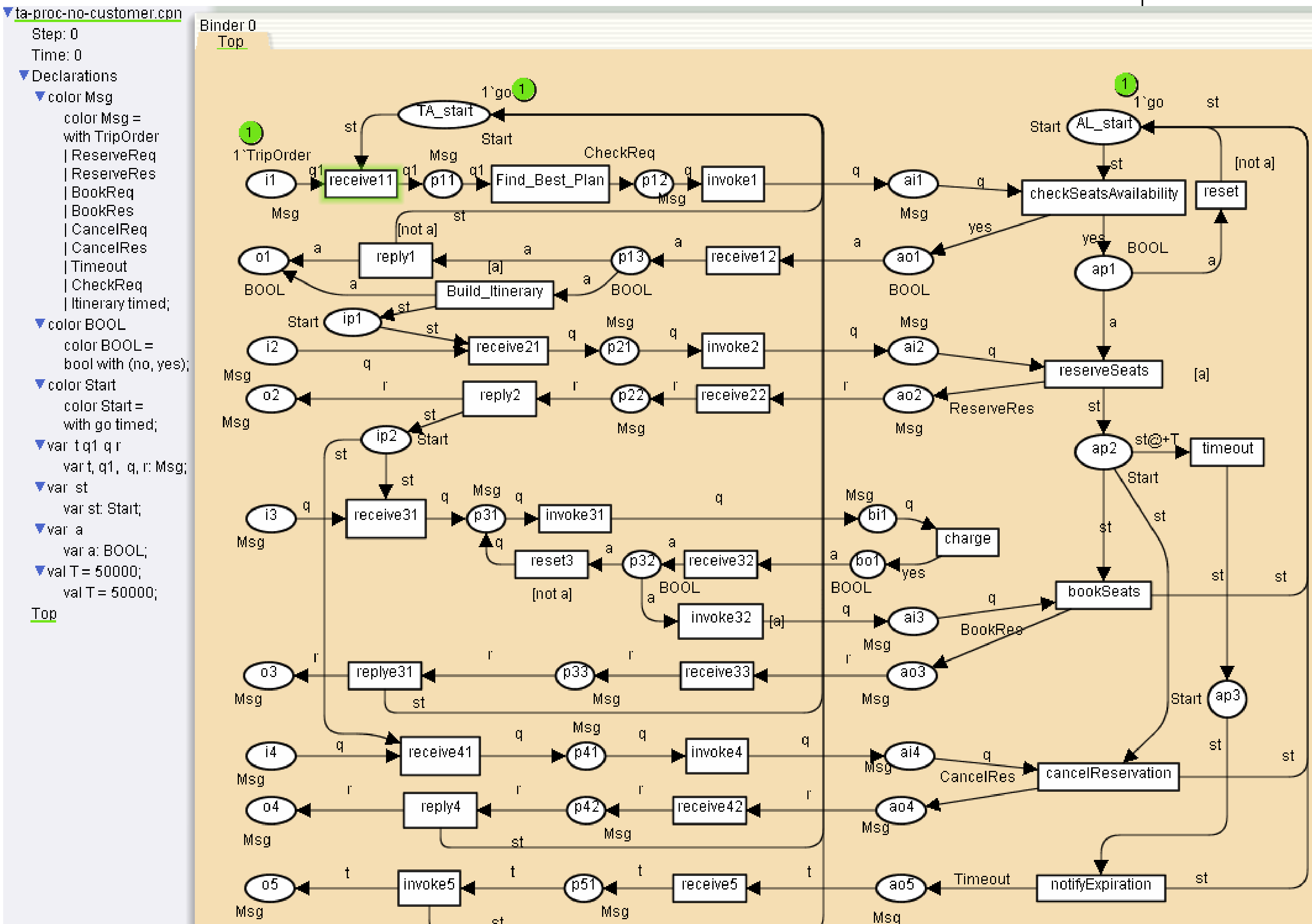
How:

- Using
  - Functionality/preconditions/effects of the participating services
  - Knowledge of conversation patterns supported by the service
  - Formal mathematical models like process algebra, concurrency formalisms like State Machines, Petri nets etc.
  - Simulation techniques

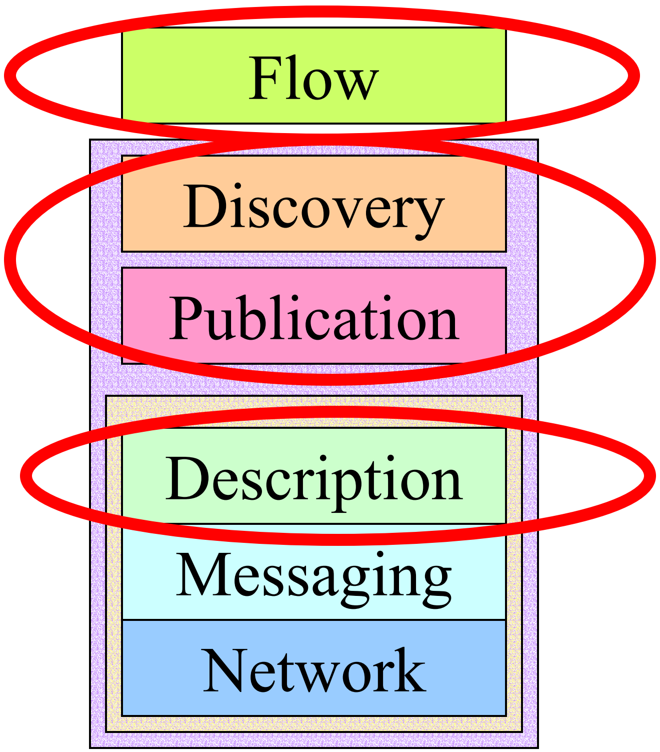
Present Scenario:

- Composition of Web services is static.
- Dynamic service discovery, run-time binding, analysis and simulation are not supported directly

Using Colored Petri nets



Semantics in WS stack and METEOR-S



MWSCF: Semantic Web Process Composition Framework

MWSDI: Scalable Infrastructure of Registries for Semantic publication and discovery of Web Services

MWSDI: Semantic Annotation of WSDL (WSDL-S)

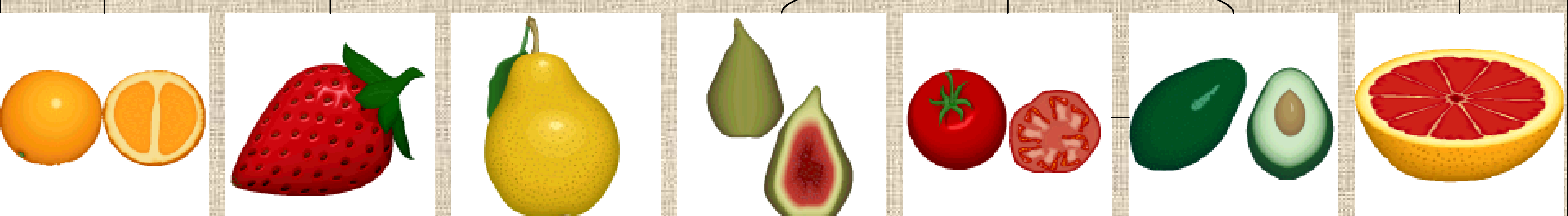
Semantic Web

Annotation of Web Services

Web Process Composition

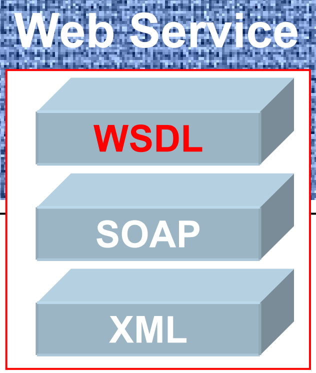
Web Service Discovery

Web Processes Quality of Service



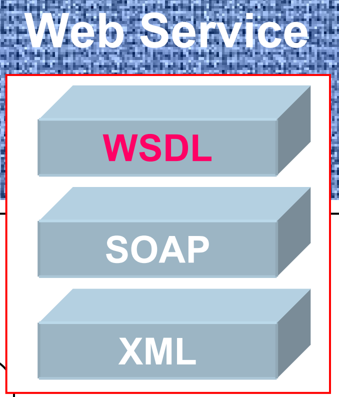
Web Service Semantic Annotation

WSDL



- WSDL stands for Web Services Description Language
- WSDL is an XML document
- WSDL is used to describe Web services
- WSDL is also used to locate Web services

WSDL




```
<definitions>
    <types>
           definition of types..
    </types>

    <message>
             definition of messages...
    </message>

    <portType>
              <operation> ..... </operation>
              <operation> ..... </operation>
    </portType>

    <binding>
             definition of binding....
    </binding>

    <service>
             <port>....</port>
             <port>....</port>
    </service>
</definitions>
```

Abstract Description

Concrete Description

From S. Chandrasekaran’s Talk

Semantic Annotation of Web Services

Annotation of Web Services

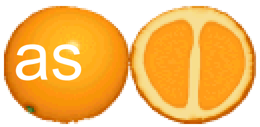
- To enhance the discovery, composition, and orchestration of Web services, it is necessary to increase the description of their interfaces.

- One solution is to annotate WSDL interfaces with semantic metadata based on relevant ontologies.

An ontology is a specification of a representational vocabulary for a shared domain of discourse.

How to Annotate ?

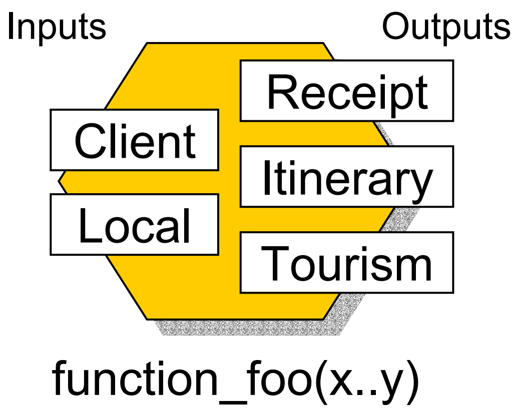
- Map Web service's input & output data as well as functional description using relevant data and function/operation ontologies, respectively



- How ?
  - Borrow from schema matching
  - Semantic disambiguation between terms in XML messages represented in WSDL and concepts in ontology

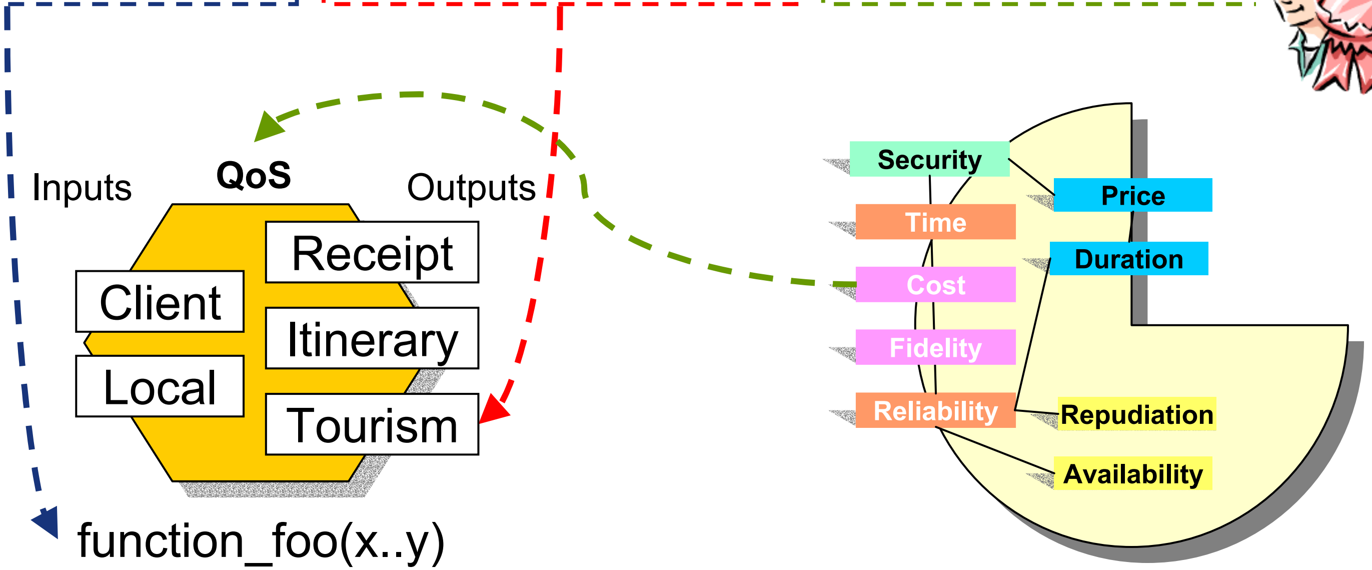
Web Services
Interfaces

- A Web service (WS) invocation specifies:
  - The number of input parameters that must be supplied for a proper WS realization and
  - The number of outputs parameters to hold and transfer the results of the WS realization to other tasks.
  - A function to invoke



Types of Annotation

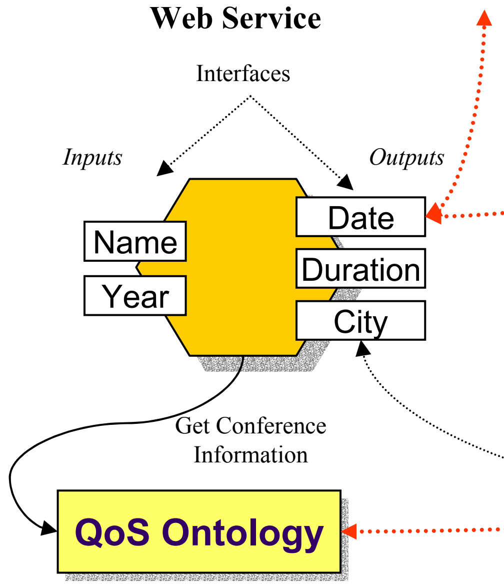
Functional Semantics | **Data Semantics** | **QoS Semantics**

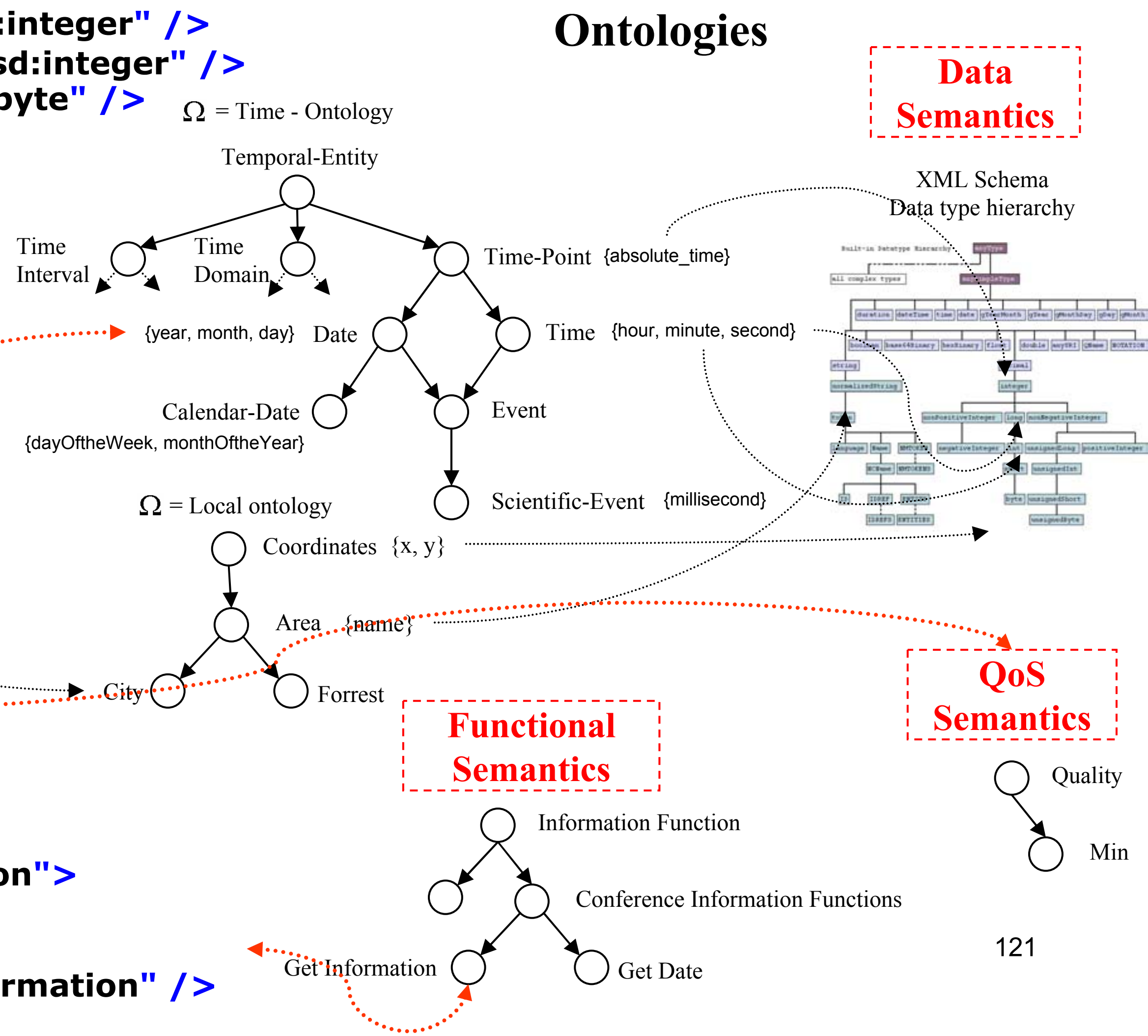


Adding Semantics to Web Services

WSDL

```
<xsd:complexType name="Date">
<xsd:sequence>
  <xsd:element name="year" type="xsd:integer" />
  <xsd:element name="month" type="xsd:integer" />
  <xsd:element name="day" type="xsd:byte" />
</xsd:sequence>
</xsd:complexType>
```





WSDL

```
<portType name="ConferenceInformation">
<operation name="getInformation">
  <input message="tns:Data" />
  <output message="tns:ConferenceInformation" />
</operation>
```

OWL-S

- OWL-S
  - Formerly OWL-S
  - Set of markup language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-intepretable form

OWL-S
Introduction

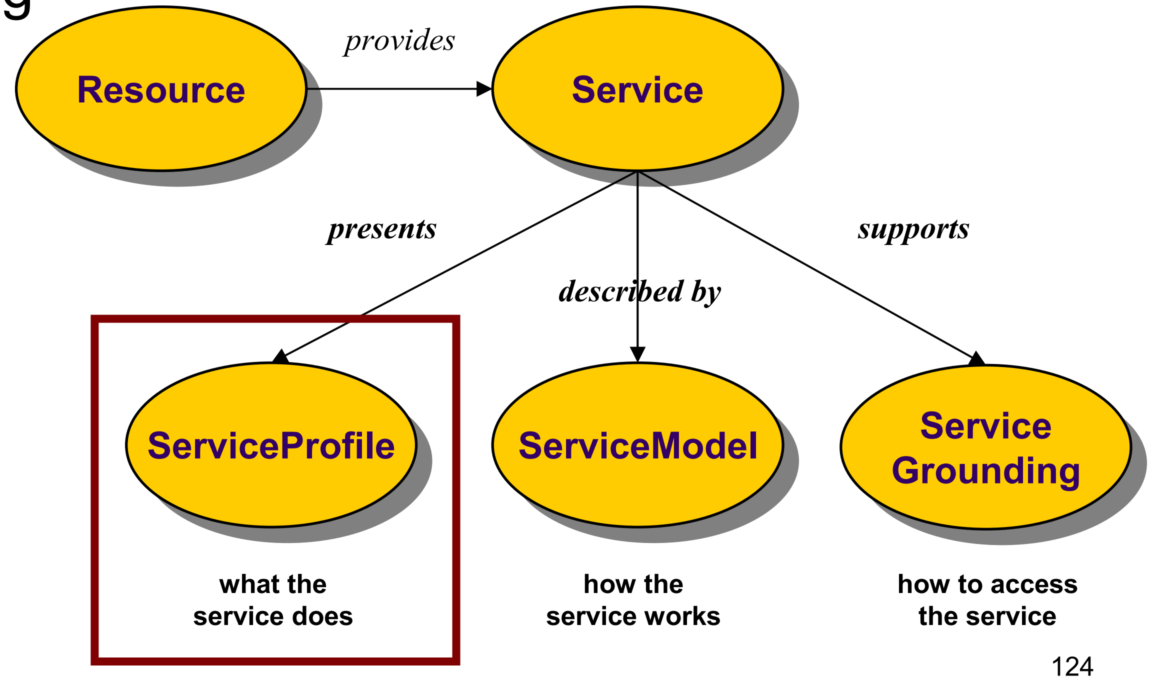
OWL-S

- OWL-S
  - DAML (DARPA Agent Markup Language)
  - OWL-S: Upper ontology of web services
- OWL-S provides support for the following elements:
  - Process description.
  - Advertisement and discovery of services.
  - Selection, composition & interoperation.
  - Invocation.
  - Execution and monitoring.

OWL-S project home page

OWL-S
Ontologies

- OWL-S defines ontologies for the construction of service models:
  - Service Profiles
  - Process Models
  - Service Grounding

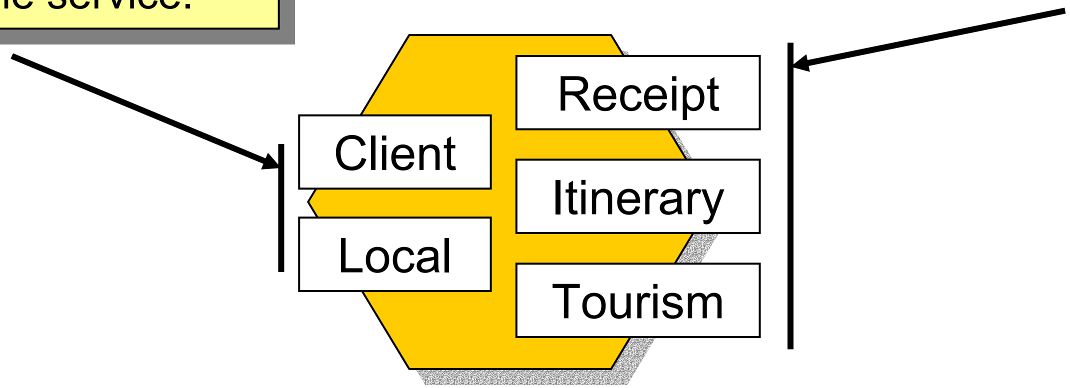


OWL-S
Service Profile

The Service Profile provides details about a service.

Inputs. Inputs that should be provided to invoke the service.

Outputs. Outputs expected after the interaction with the service.



Web Service

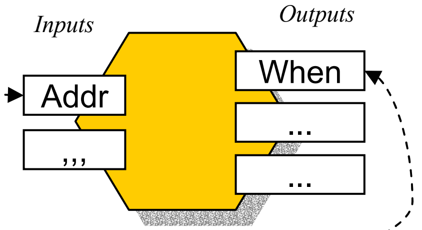
Preconditions. Set of conditions that should hold prior to the service being invoked.

Effects. Set of statements that should hold true if the service is invoked successfully.

Service Profile

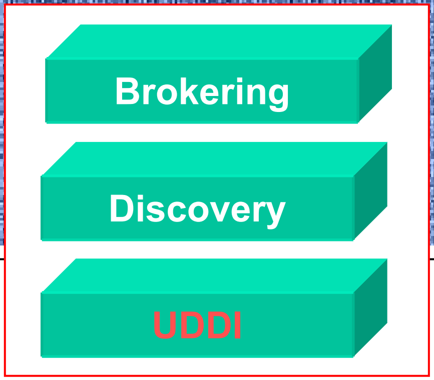
An example of Inputs and Outputs

```
...
<!ENTITY temporal "http://ovid.cs.uga.edu:8080/scube/daml/Temporal.daml">
<!ENTITY address "http://ovid.cs.uga.edu:8080/scube/daml/Address.daml">
...
<input>
    <profile:ParameterDescription rdf:ID="Addr">
    <profile:parameterName> Addr </profile:parameterName>
    <profile:restrictedTo rdf:resource="&address;#Address"/>
    <profile:refersTo rdf:resource="&congo;#congoBuyReceipt"/>
    </profile:ParameterDescription>
</input>
...
<output>
    <profile:ParameterDescription rdf:ID="When">
    <profile:parameterName> When </profile:parameterName>
    <profile:restrictedTo rdf:resource="&temporal;#Date"/>
    <profile:refersTo rdf:resource="&congo;#congoBuyReceipt"/>
    </profile:ParameterDescription>
< output >
...
```



Semantic Web Service Discovery

UDDI



- **UDDI** stands for Universal Description, Discovery and Integration
- **UDDI** serves as a "Business and services" registry and directory and are essential for dynamic usage of Web services
- A **UDDI** registry is **similar to a CORBA trader**, or it can be thought of as a DNS for business applications.
- Is a platform-independent framework for **describing** services, **discovering** businesses, and **integrating** business services by using the Internet.

How UDDI Works ?

1. 

SW companies, standards bodies, and programmers populate the registry with descriptions of different types of services

2. 

Businesses populate the registry with descriptions of the services they support

UDDI Business Registry

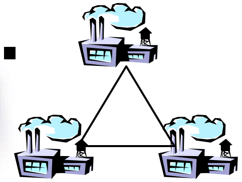
Business Registrations

Service Type Registrations

3. UBR assigns a programmatically unique identifier to each service and business registration

4. 

Marketplaces, search engines, and business apps query the registry to discover services at other companies

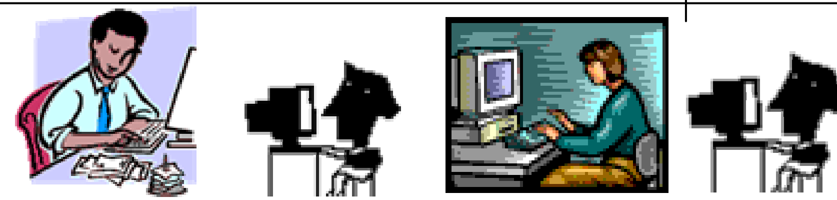
5. 

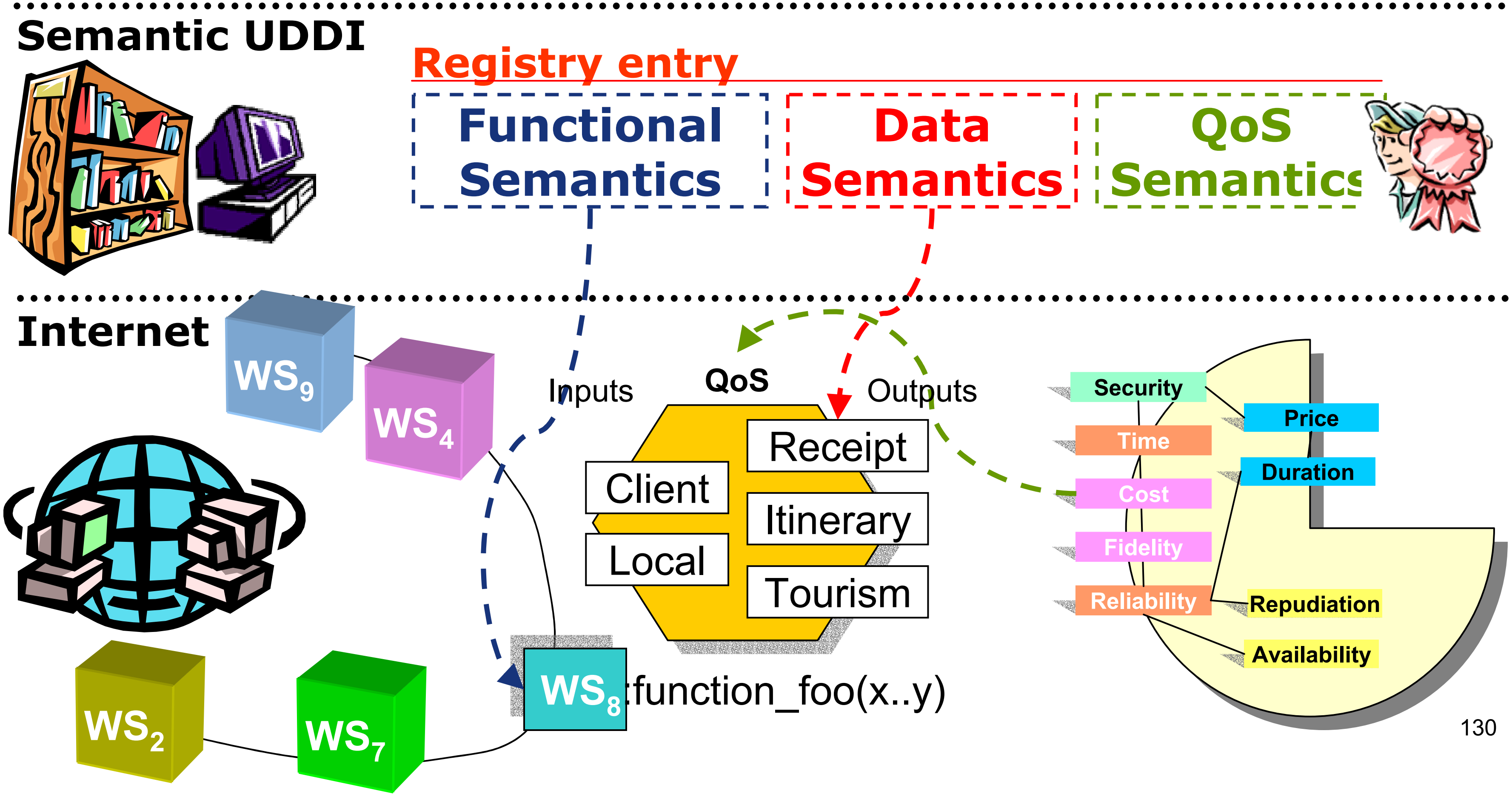
Business uses this data to facilitate easier integration with each other over the Web

Source : http://www.uddi.org/pubs/UDDI_Overview_Presentation.ppt

UDDI and Semantics

Marketplaces, search engines, and business apps query

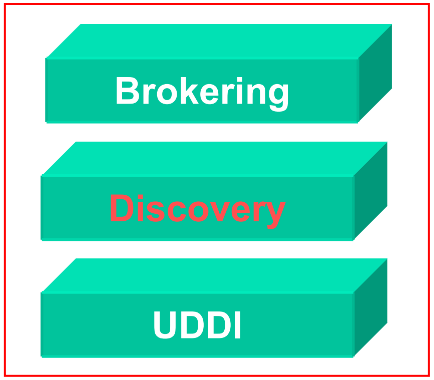




Semantic Discovery of Web Services

Web Service Discovery

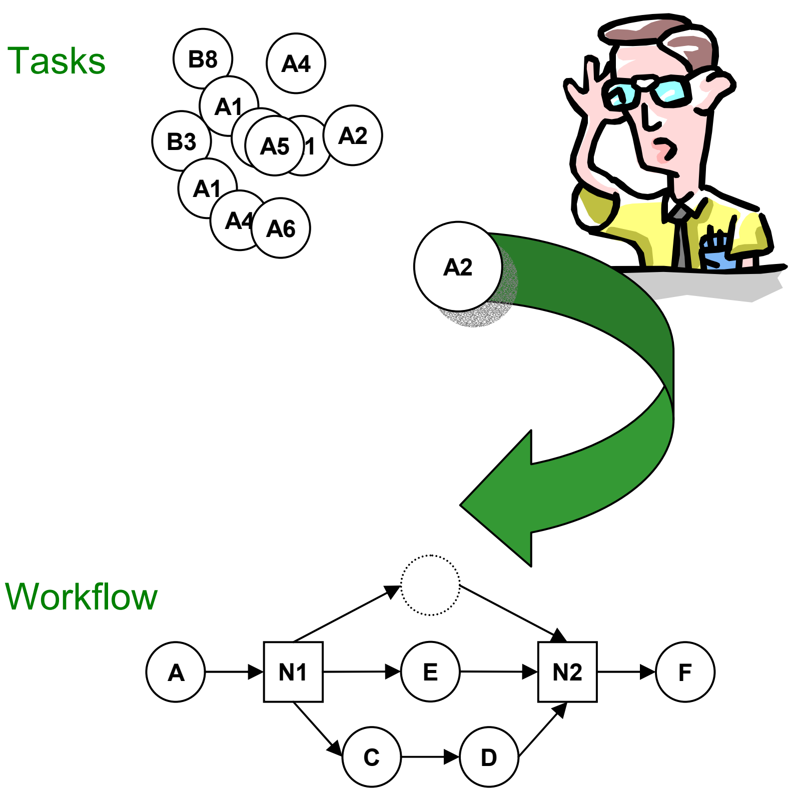
Web Services must be located (Discovery) that might contain the desired functionality, operational metrics, and interfaces needed to carry out the realization of a given task.

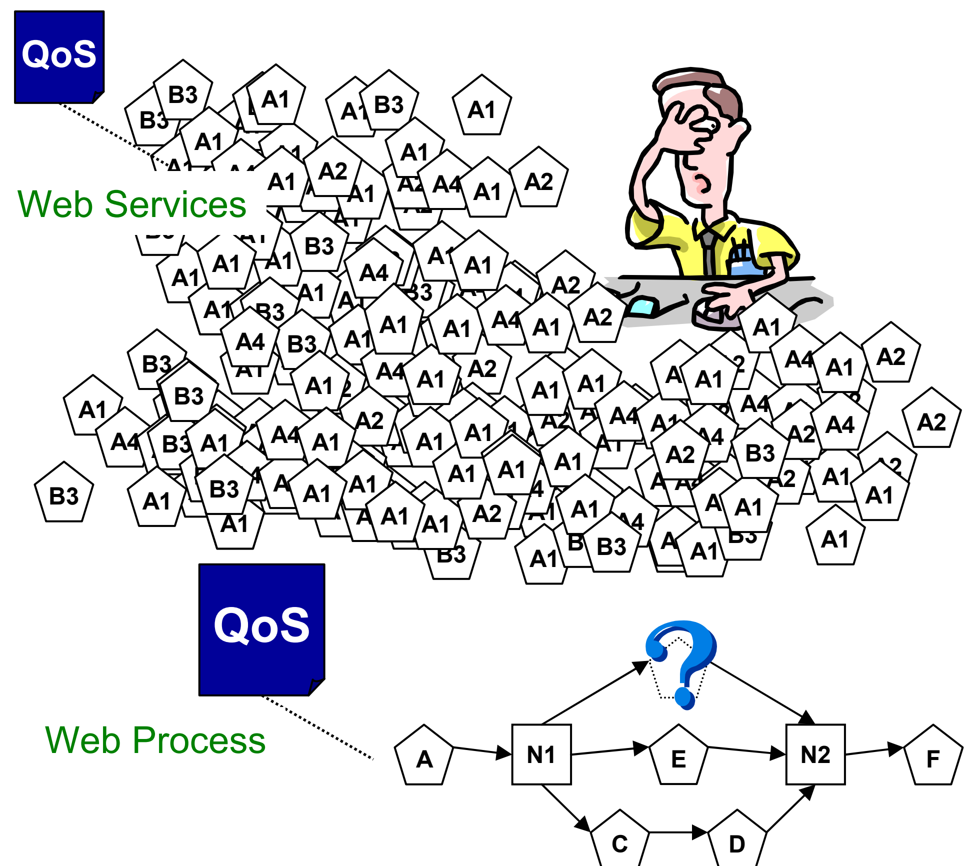


Discovery
New Requirements

Web Service Discovery

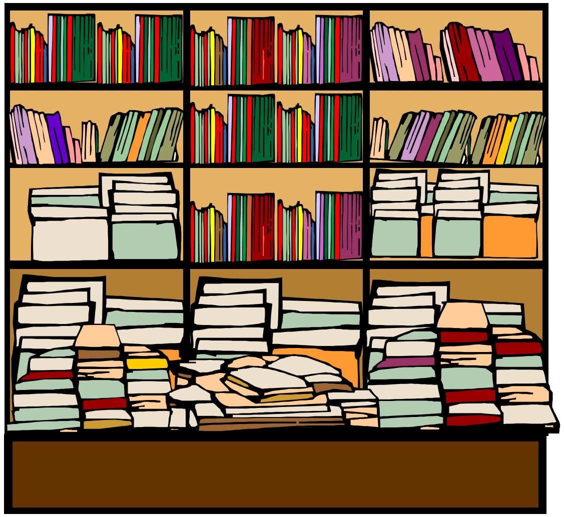
Before	Now





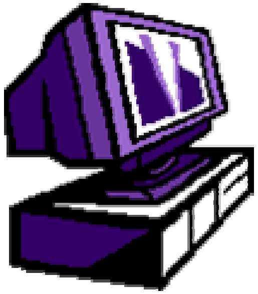
State of the art in discovery

UDDI Business Registry



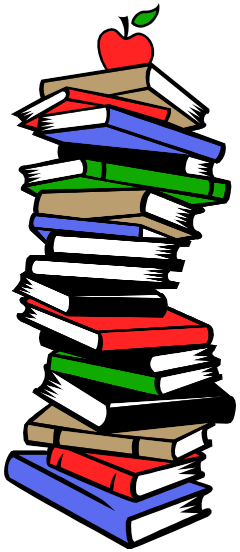
Provides non-semantic search

Search



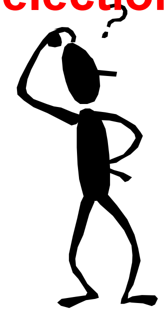
Keyword and attribute-based match

Results



Search retrieves lot of services (irrelevant results included)

Selection



Which service to select ?
How to select?

Present Discovery Mechanism
Keyword and attribute-based search

Web Service Discovery

- UDDI :Keyword and attribute-based search
- Example: “Quote”
  - Microsoft UBR returned 12 services
  - Human reading of description (Natural Language) help me understand:
    - 6 Entries are to get Famous Quotes
    - 1 Entry for personal auto and homeowners quoting
    - 1 Entry for multiple supplier quotes on all building materials
  - Categorization suggested for UDDI is useful but inadequate (what does the WS do?) :
    - 1 Entry for Automobile Manufacturing
    - 1 Entry for Insurance agents, brokers, & service
  - Alternatively read and try to understand WSDL
    - 1 Entry related to security details (Human Understanding)
    - 1 Test Web service for Quotes (which quote?)

Present Discovery Mechanism
Search for services to book an air ticket (using categories)*

- unspsc-org: unspsc:3-1
  - Travel, Food, Lodging and Entertainment Services
    - Travel facilitation
      - Travel agents
        - Travel agencies
- Services: 3 records found.
  - AirFares
    Returns air fares from netviagens.com travel agent
  - Hotel reservations
    Reservations for hotels in Asia, Australia and New Zealand
  - Your Vacation Specialists
    Web enabled vacation information
- Providers: 2 records found.

* Search carried out in one of the Universal Business Registries

Present Discovery Mechanism
Search for services to book an air ticket (using Keywords)*

- air ticket
  - 1 record with name air tickets booking
- airticket, ticketbooking, airtravel, air travel, travel agent, airticketbooking, air ticket booking, travel agency, travelagency
  - 0 records were returned
- travelagent
  - 1 record with name travelagent test
    - 4 services: BookFlight, cancelFlightBooking etc.
    - Descriptions say that both these services are “XML based Web services”
    - No URL for WSDL
- Travel
  - 15 records. Purpose/functionality understood from descriptions
    - 2 services : TravelBooks
    - 4 services : TravelInformation
    - 2 services : Reservation and cancallation of travel tickets
    - 1 service : Emergency Services for travellers
    - 1 service : Travel documentation and itinerary
    - 5 services : Description is ambiguous/not present

*Search carried out in one of the Universal Business Registries

The use of semantics

Benefits

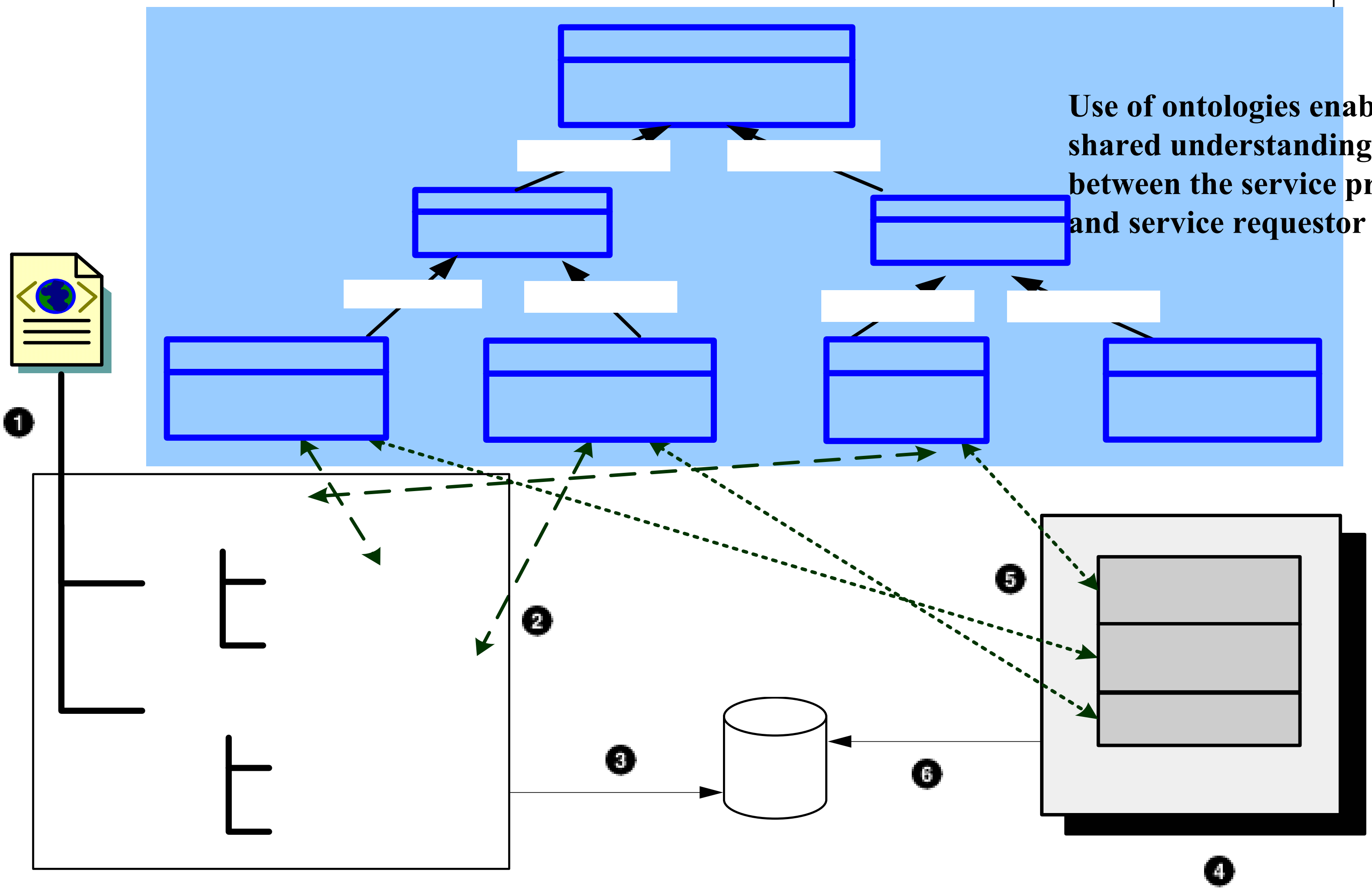
Web Service Discovery

- Search engines can better “understand” the contents of a particular page
- More accurate searches
- Additional information aids precision
- Makes it possible to automate searches because less manual “weeding” is needed to process the search results
- Facilitates the integration of several Web services

Semantic Discovery: Overview

- Annotation and Publication
  - WSDL file is annotated using ontologies and the annotations are captured in UDDI
- Discovery
  - Requirements are captured as templates that are constructed using ontologies and semantic matching is done against UDDI entries
    - Functionality of the template, its inputs, outputs, preconditions and effects are represented using ontologies
- Use of ontologies
  - brings service provider and service requestor to a common conceptual space
  - helps in semantic matching of requirements and specifications

Semantic Publication and Discovery



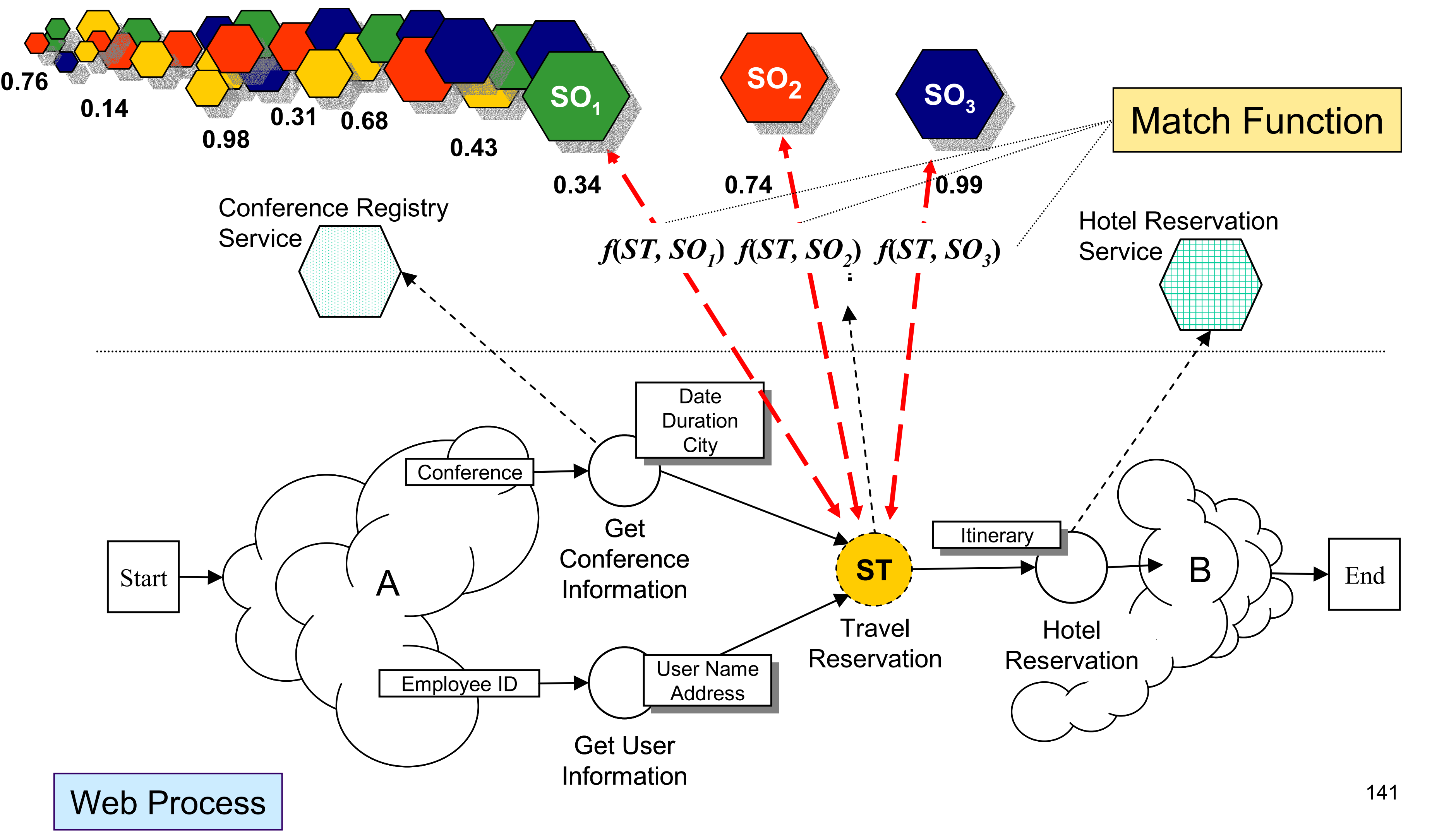
For simplicity of depicting, the ontology is shown with classes for both operation and data

[Adding Semantics to Web Services Standards]

Discovery
The Match Function

- The Web service discovery and integration process is carried out by a key operation:
  - The match function.
- The matching step is dedicated to finding correspondences between a service template (ST, *i.e.,* a query) and a service object (SO).

The Match Function



Discovery in Semantic Web
Using Semantics

Web Service Discovery

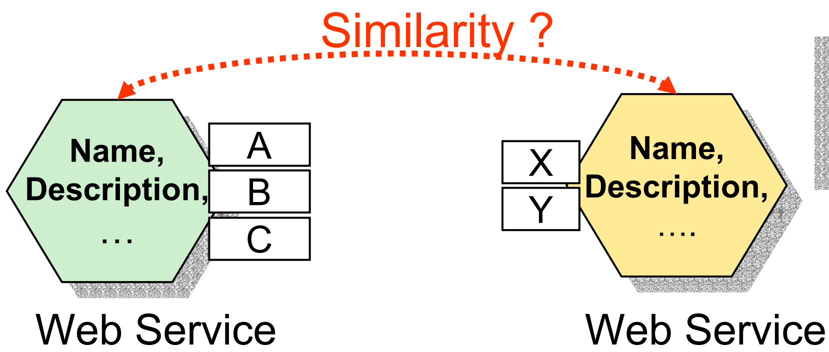
- Functionality: What capabilities the distributor expects from the service
  (Functional semantics)
- Inputs: What the distributor can give to the to the Manufacturer's service
  (Data semantics)
- Outputs: What the distributor expects as outputs from the service
  (Data semantics)
- QoS: Quality of Service the distributor expects from the service
  (QoS semantics)

(Functional semantics)
(Data semantics)
(QoS semantics)
(Syntactic description)

- Description: Natural language description of the service functionality
  (Syntactic description)

Syntactic, QoS, and Semantic (Functional & Data) Similarity

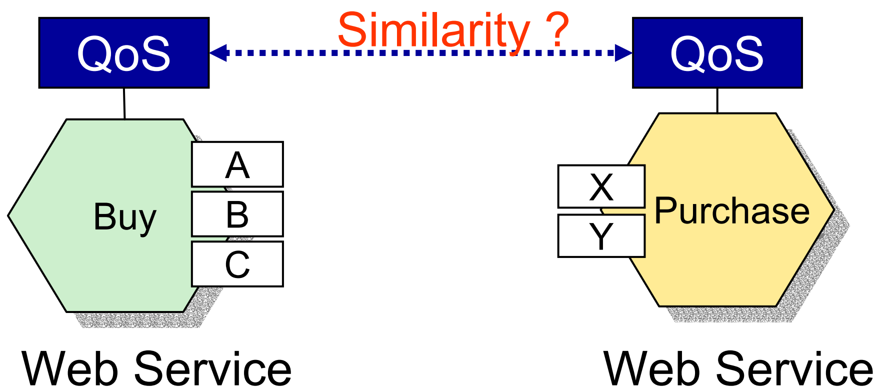
Web Service Discovery



Syntactic Similarity

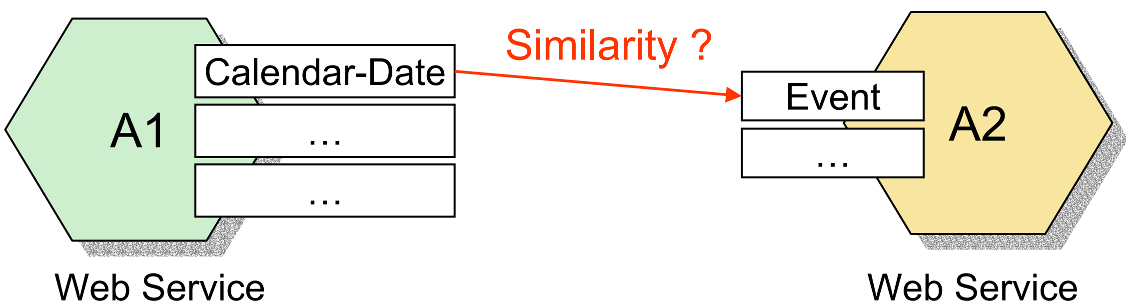
$$SynSimilarty(ST,SO)=\frac{\omega_1 SynNS(ST.sn,SO.sn)+\omega_2 SynDS(ST.sd,SO.sd)}{\omega_1+\omega_2}\in[0..1],$$

$$\text{and } \omega_1,\omega_2\in[0..1]$$

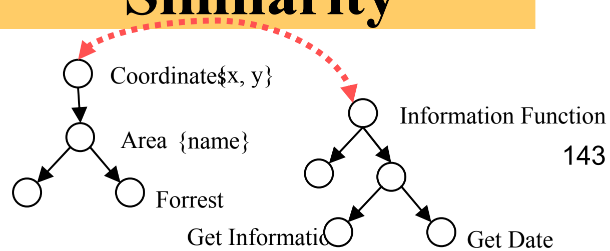


QoS Similarity

$$\mathrm{OpSimilarity}(ST,SO)=\sqrt[3]{\mathrm{QoSdimD}(ST,SO,time)*\mathrm{QoSdimD}(ST,SO,cost)*\mathrm{QoSdimD}(ST,SO,reliability)}$$



Functional & Data Similarity



The Match Function

Semantic Similarity

- Purely syntactical methods that treat terms in isolation from their contexts.
  - It is insufficient since they deal with syntactic but not with semantic correspondences
  - Users may express the same concept in different ways.
- Therefore, we rely on semantic information to evaluate the similarity of concepts that define ST and SO interfaces.
- This evaluation will be used to calculate their degree of integration.

The Match Function
Semantic Similarity

- When comparing concepts two main cases can occur:

  - The concepts are defined with the same Ontology
    $(\Omega(O) = \Omega(I))$
  - The concepts are defined in different Ontologies
    $(\Omega(O) \neq \Omega(I))$

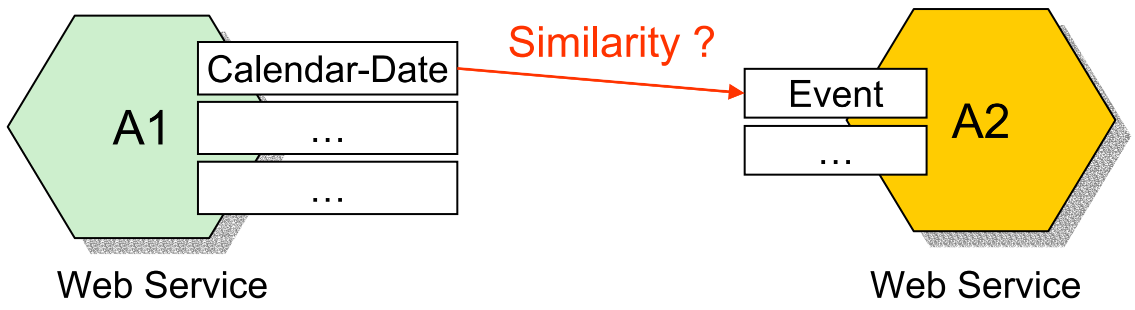
The Match Function

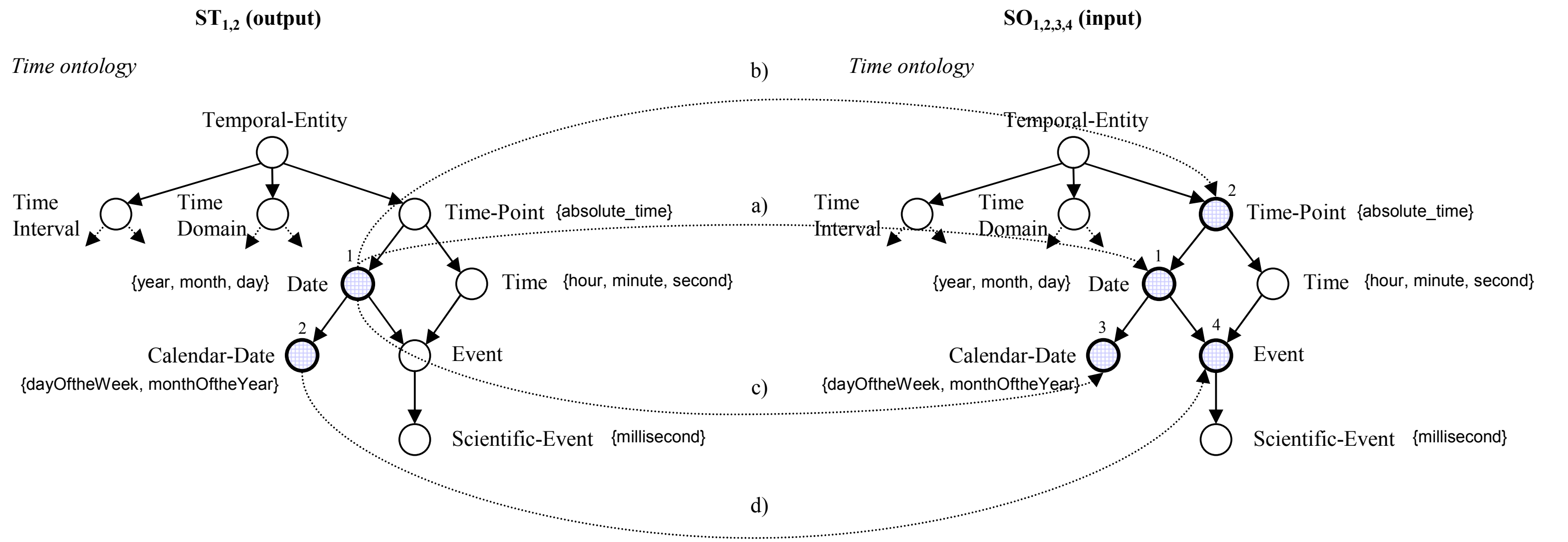
Semantic Similarity ($\Omega(O) = \Omega(I)$)

- When comparing concepts defined with the same ontology four distinct scenarios need to be considered:
  - a) the concepts are the same ($O=I$)
  - b) the concept I subsumes concept O ($O>I$)
  - c) the concept O subsumes concept I ($O<I$), or
  - d) concept O is not directly related to concept I ($O \neq I$).

The Match Function

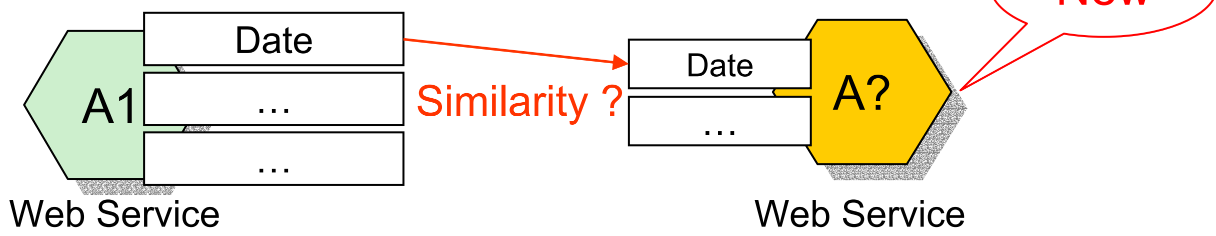
Semantic Similarity ($\Omega(O) = \Omega(I)$)

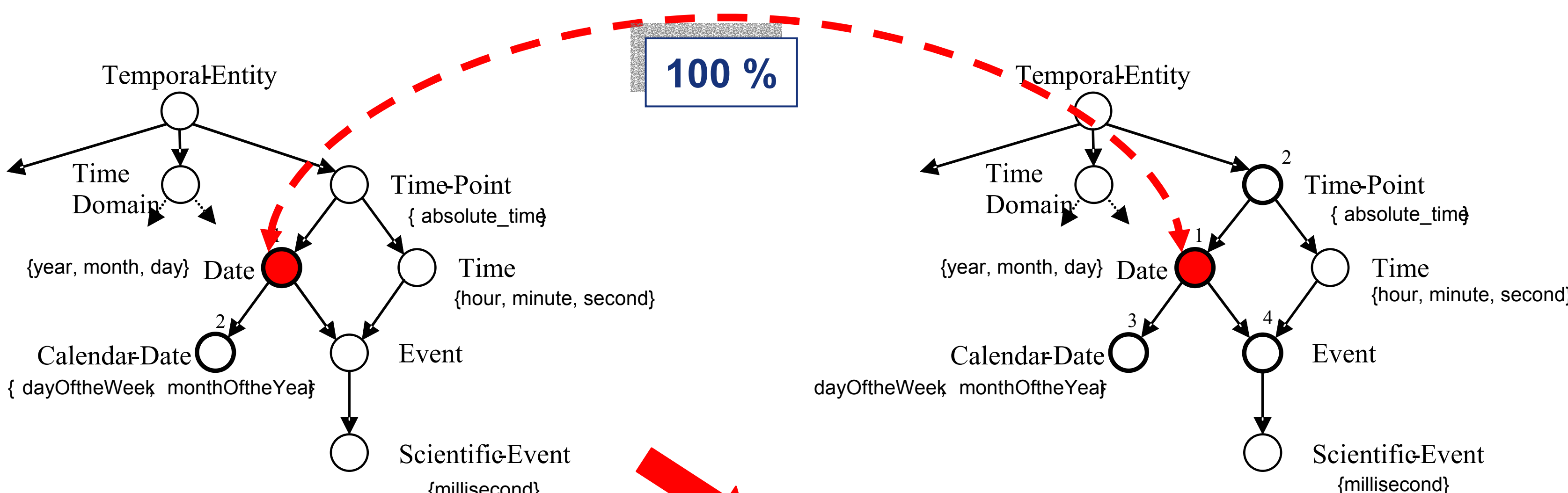




The Match Function

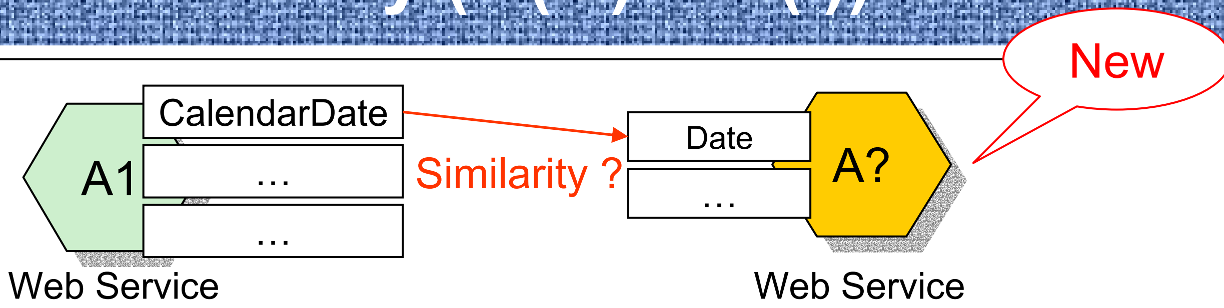
Semantic Similarity ($\Omega(O) = \Omega(I)$)

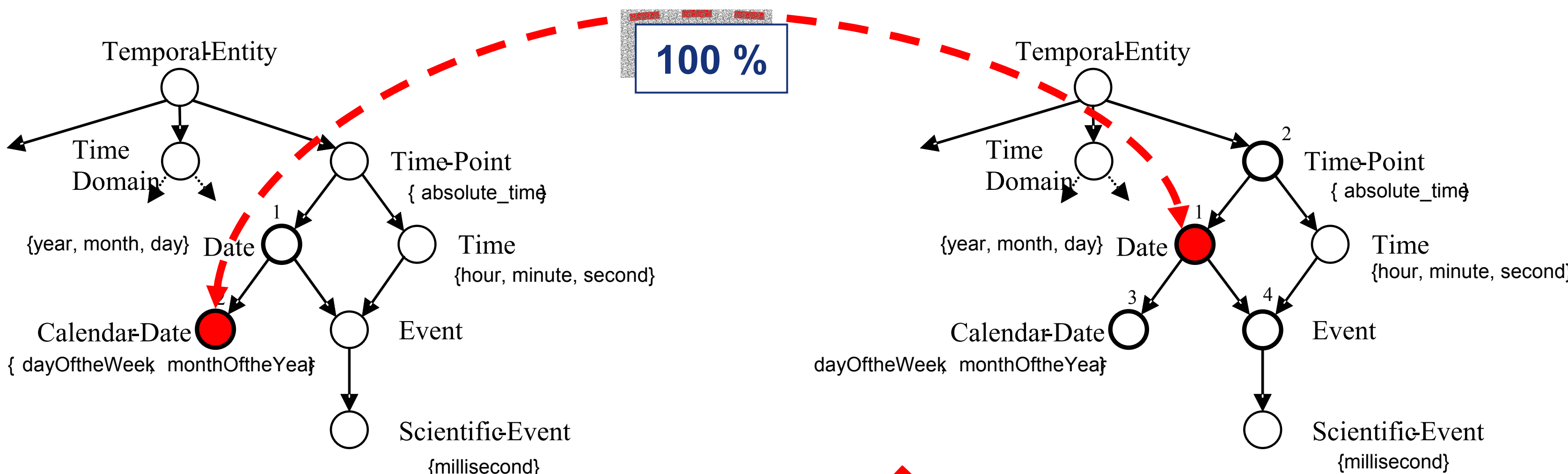




$$SemS'(O,I) = \begin{cases} 1, & O = I \\ 1, & O > I \\ \dfrac{|p(O)|}{|p(I)|}, & O < I \\ Similarity'(O,I), & O \neq I \end{cases}$$

The Match Function
Semantic Similarity ($\Omega(O) = \Omega(I)$)

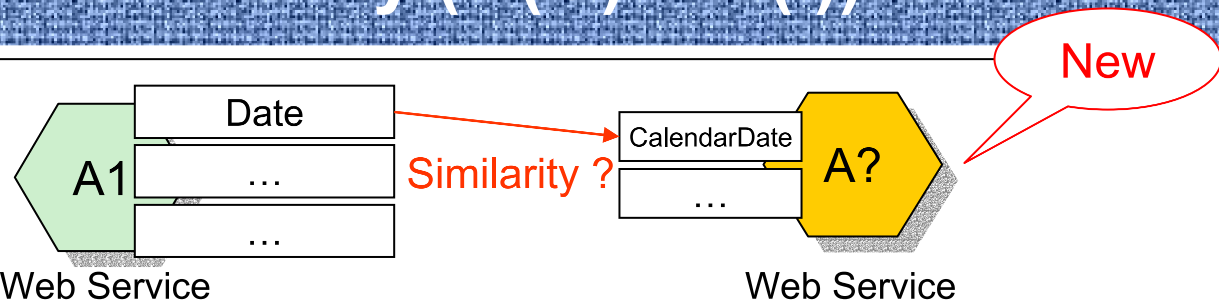


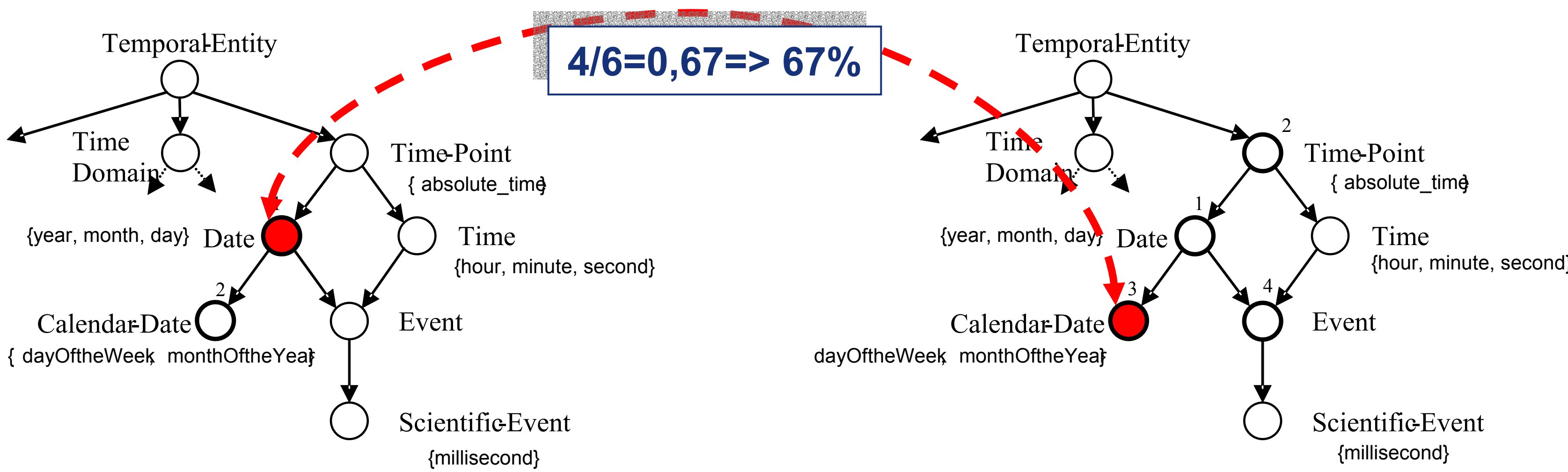


$$SemS'(O,I)=\begin{cases}1, & O=I\\ 1, & O>I\\ \dfrac{|p(O)|}{|p(I)|}, & O<I\\ Similarity'(O,I), & O\neq I\end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O) = \Omega(I)$)

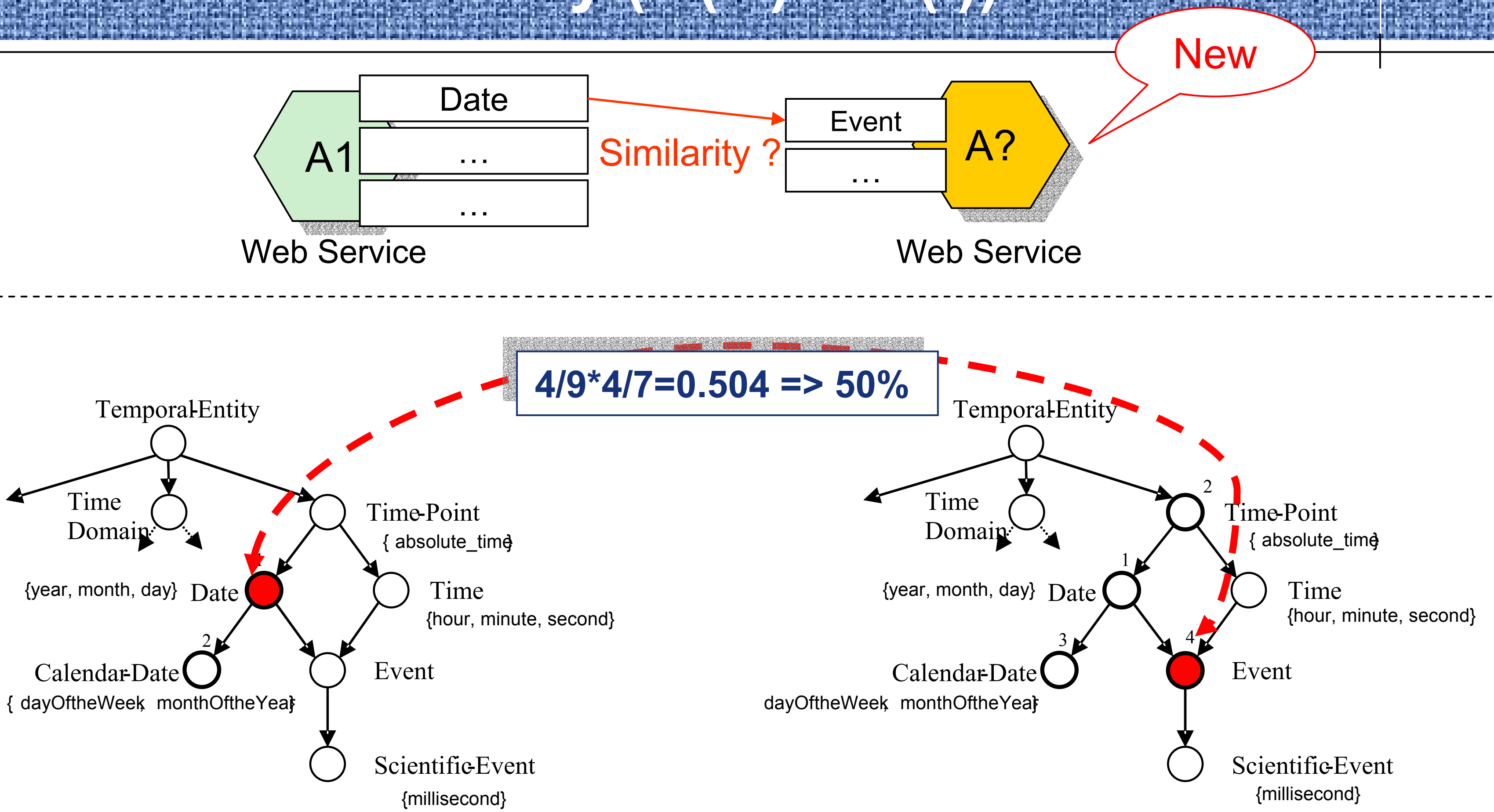




$$SemS'(O,I) = \begin{cases} 1, & O = I \\ 1, & O > I \\ \dfrac{|p(O)|}{|p(I)|}, & O < I \\ Similarity(O,I), & O \neq I \end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O) = \Omega(I)$)



$$SemS'(O,I) = \begin{cases} 1, & O = I \\ 1, & O > I \\ \frac{|p(O)|}{|p(I)|}, & O < I \\ Similarity'(O,I), & O \neq I \end{cases}$$

$$similarity'(O,I) = \sqrt{\frac{|p(O) \cap p(I)|}{|p(O) \cup p(I)|} * \frac{|p(O) \cap p(I)|}{|p(I)|}}$$

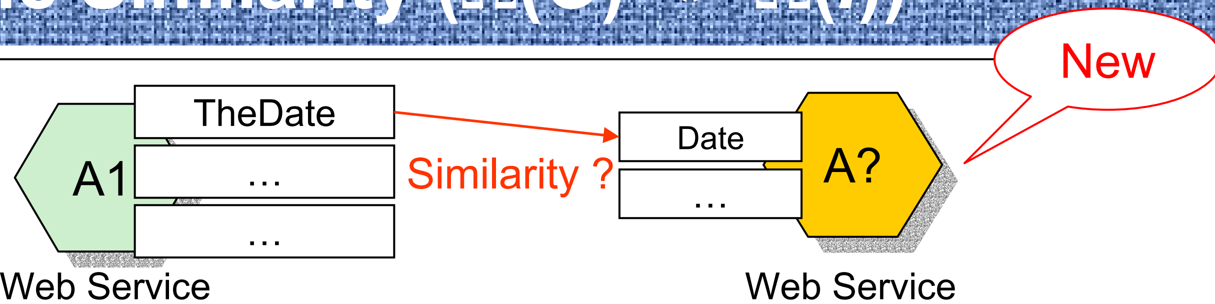
The Match Function

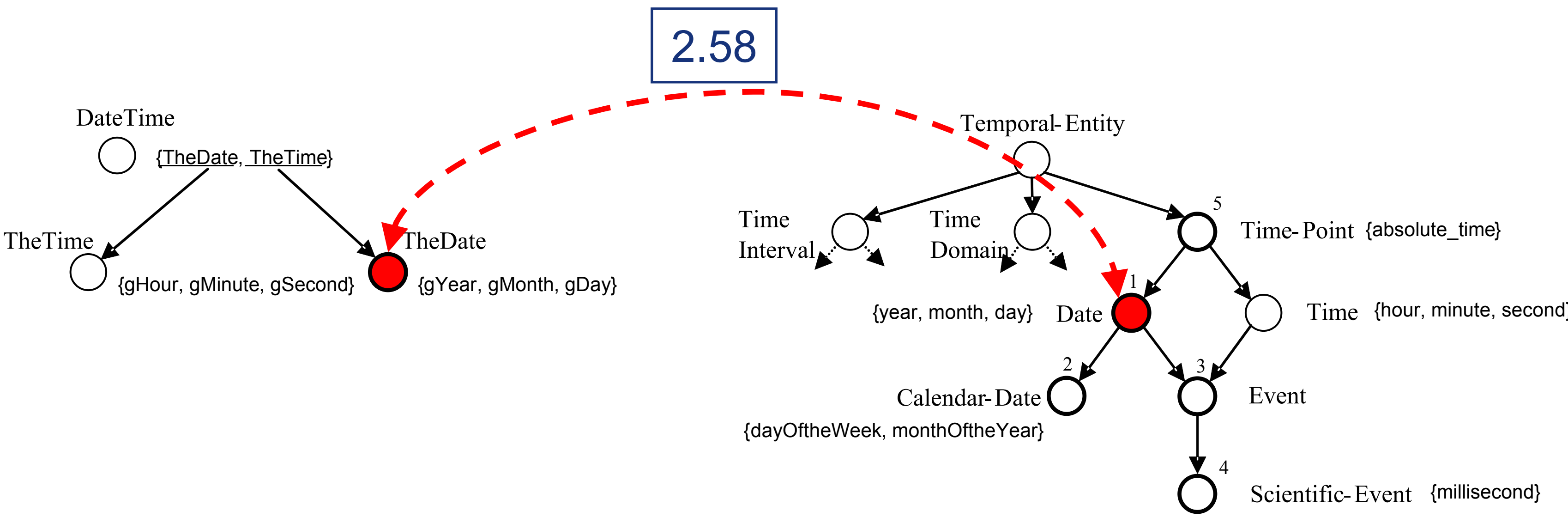
Semantic Similarity ($\Omega(O) \neq \Omega(I)$)

- When comparing concepts defined with different ontologies three distinct scenarios can occur:
  - The ontological properties involved are associated with a primitive data type
  - The properties are associated with concept classes, and
  - One property is associated with a primitive data type, while the other is associated with a concept class.

The Match Function

Semantic Similarity ($\Omega(O)$ <> $\Omega(I)$)

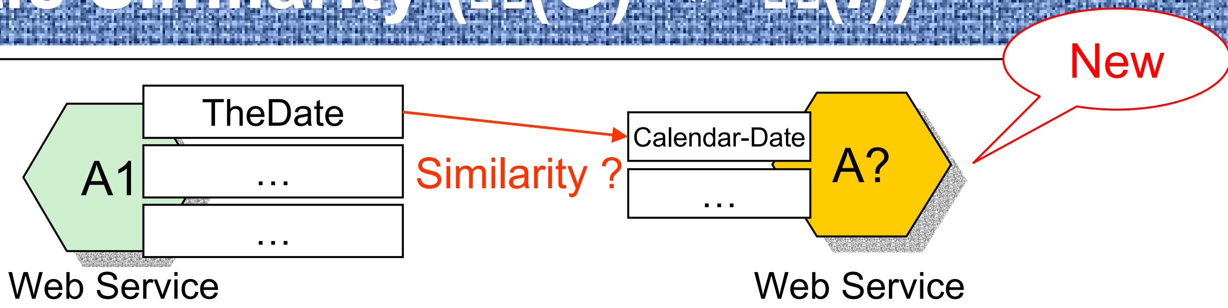


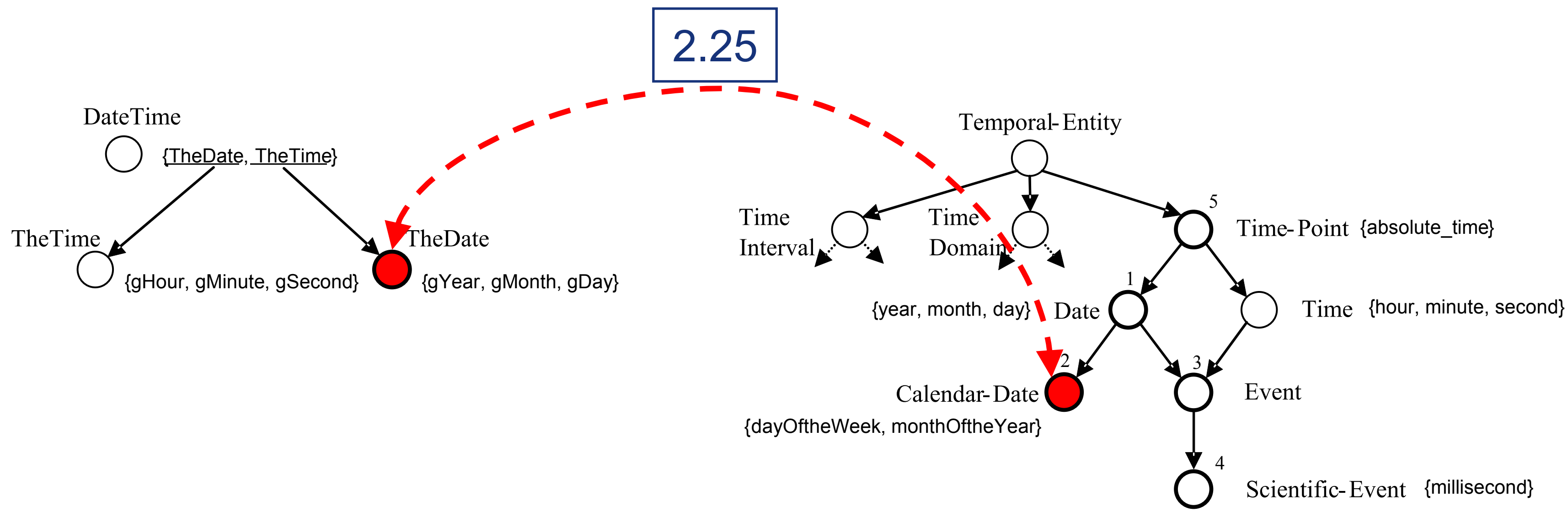


$$S(o,i) = \begin{cases} \sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types} \\ SemDS(o,i), & o \text{ and } i \text{ are concept classes} \\ f(o,i), & \text{otherwise} \end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O)$ <> $\Omega(I)$)

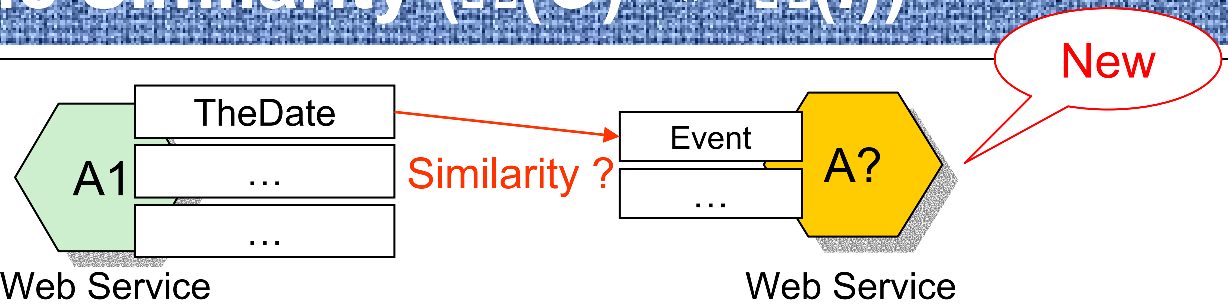


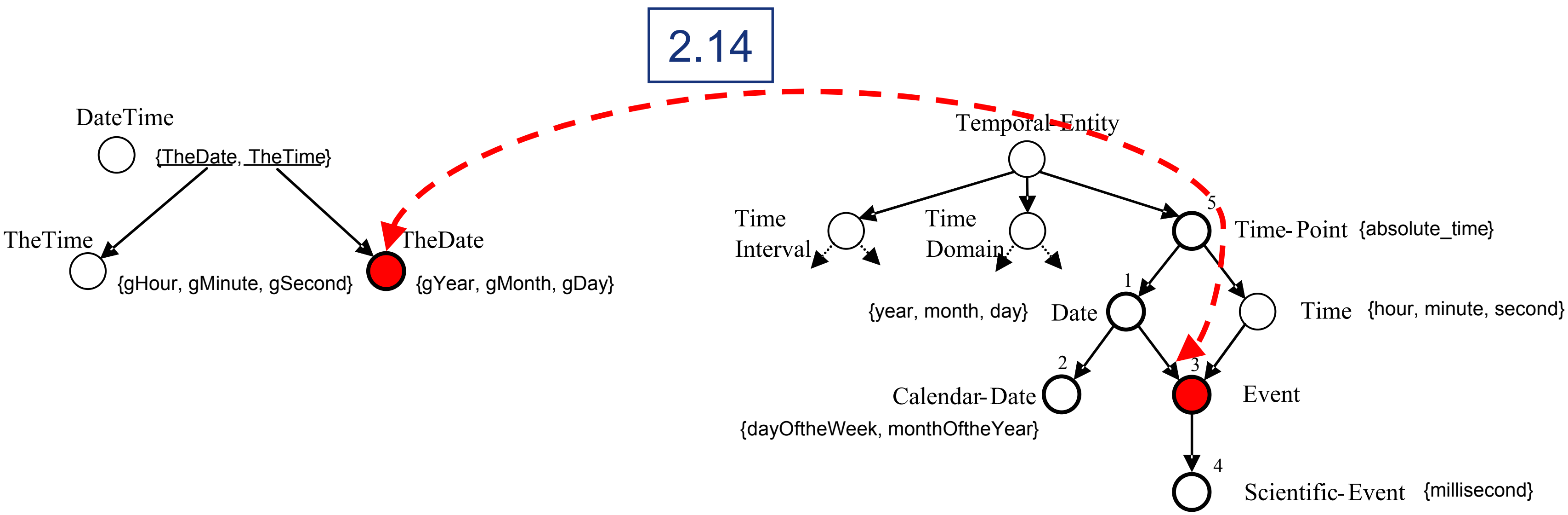


$$S(o,i) = \begin{cases} \sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types} \\ SemDS(o,i), & o \text{ and } i \text{ are concept classes} \\ f(o,i), & \text{otherwise} \end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O) <> \Omega(I)$)

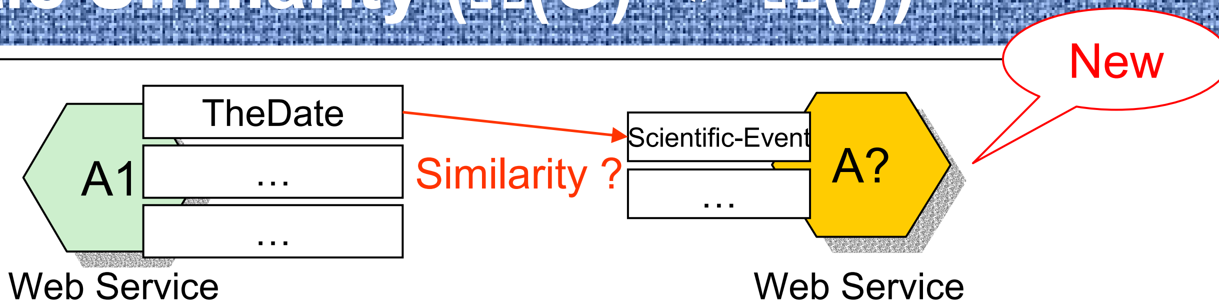


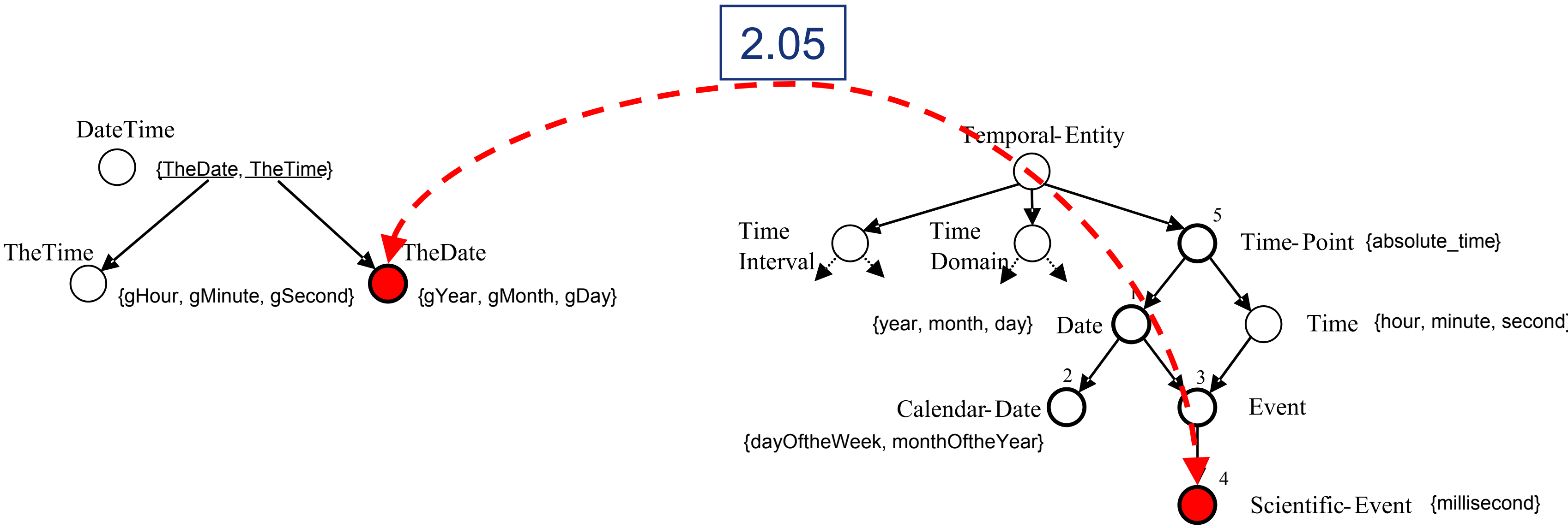


$$S(o,i) = \begin{cases} \sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types} \\ SemDS(o,i), & o \text{ and } i \text{ are concept classes} \\ f(o,i), & \text{otherwise} \end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O)$ <> $\Omega(I)$)

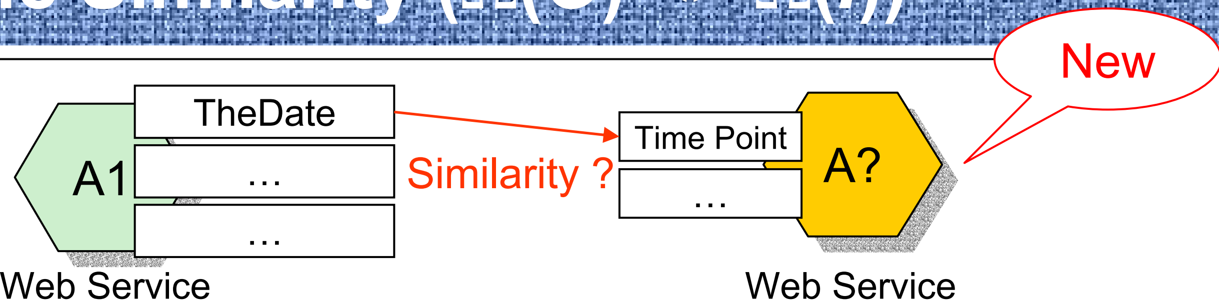


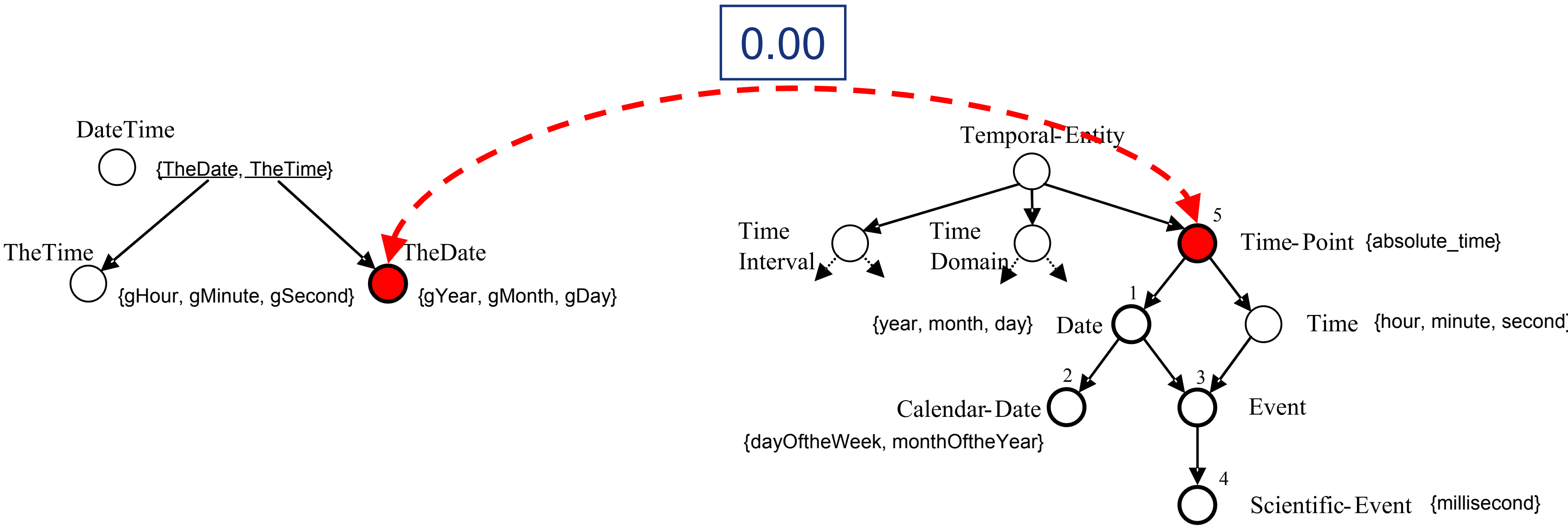


$$S(o,i)=\begin{cases}\sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types}\\ SemDS(o,i), & o \text{ and } i \text{ are concept classes}\\ f(o,i), & \text{otherwise}\end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O)$ <> $\Omega(I)$)

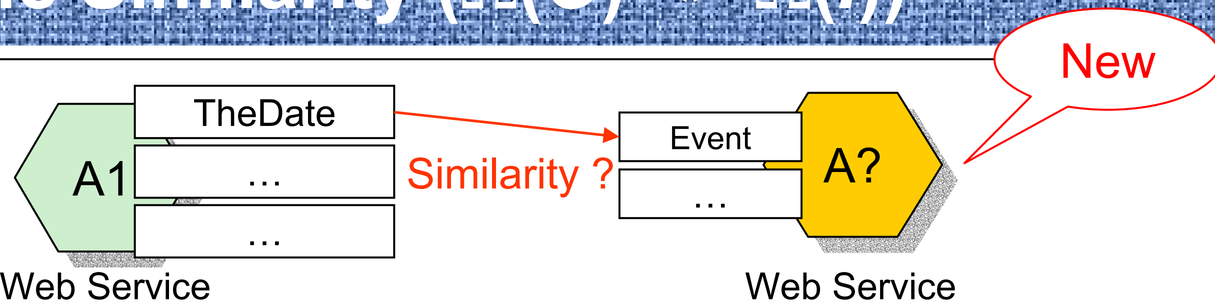


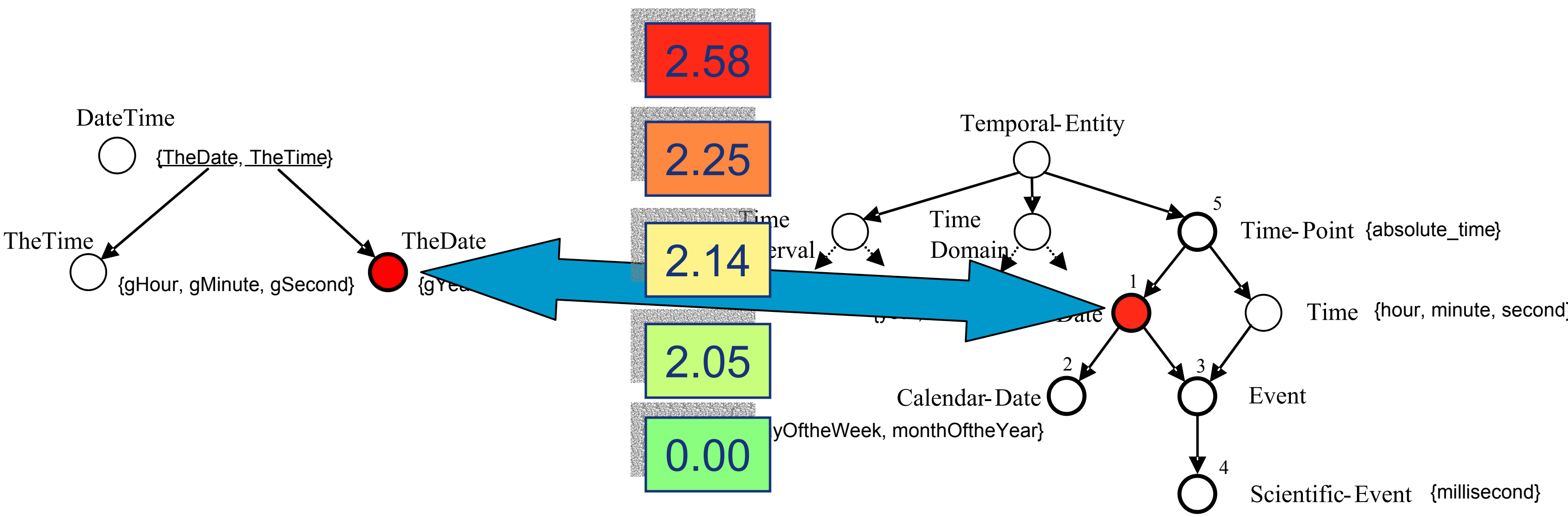


$$S(o,i) = \begin{cases} \sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types} \\ SemDS(o,i), & o \text{ and } i \text{ are concept classes} \\ f(o,i), & \text{otherwise} \end{cases}$$

The Match Function

Semantic Similarity ($\Omega(O)$ <> $\Omega(I)$)

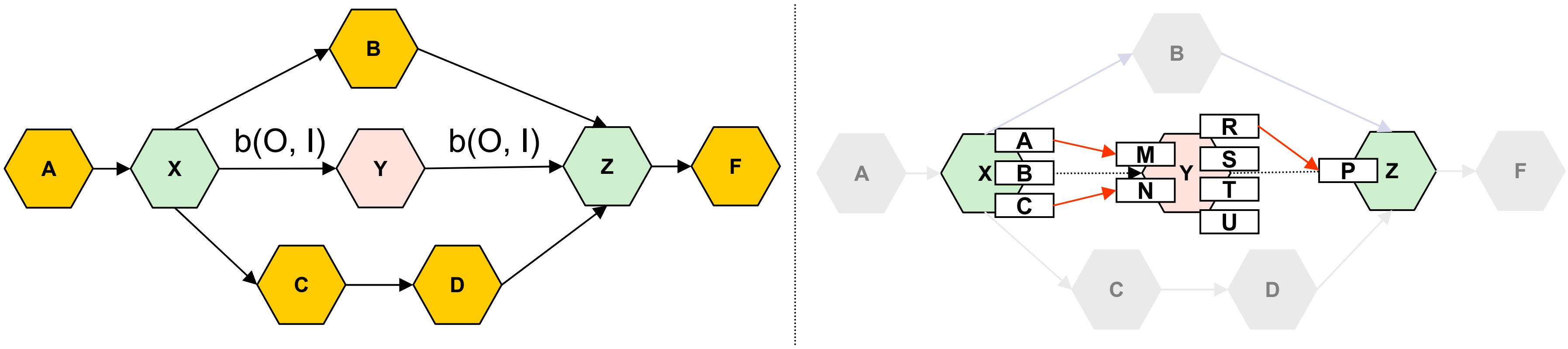




$$S(o,i)=\begin{cases}\sqrt[3]{SemDS(d(o),d(i))*SynS(n(o),n(i))*SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types}\\ SemDS(o,i), & o \text{ and } i \text{ are concept classes}\\ f(o,i), & \text{otherwise}\end{cases}$$

Web Services
Integration

- The degree of integration of a Web service is evaluated using semantic information.
- For each interface to integrate we construct a bipartite graph with a bipartition b(O, I).
- Each edge has a weight (semantic similarity).
- We then compute the optimal matching*.



*Bondy and Murty 1976

Discovery
Example of a Query

DAML

Discovery and Integration
Query Results

Web Service Discovery Results - Microsoft Internet Explorer

File Edit View Favorites Address http://localhost:8080/scube/servlet/SearchServlet Go

Web Service Discovery Results

Web service Object	
Service Name	Internet Travel
Service Description	Internet travel reservation and information service for business travelers.
Service URI	http://ovid.cs.uga.edu:8080/scube/daml/SO_A3.daml
Discovery	**Results**
Syntactic Similarity	0.33
Operational Similarity	0.93
Semantic Similarity	0.67

Operational Metrics	Min	Avg	Max
Time	9	16	22
Cost	27	34	52
Reliability	0.82	0.87	0.97

DI	ST.Output => SO.Input
1	Person (http://ovid.cs.uga.edu:8080/scube/daml/Person.daml#Person) -> Client (http://ovid.cs.uga.edu:8080/scube/daml/Person.daml#Person)
0.67	Date (http://ovid.cs.uga.edu:8080/scube/daml/Temporal.daml#Date) -> When (http://ovid.cs.uga.edu:8080/scube/daml/Temporal.daml#CalendarDate)
1	HomeAddress (http://ovid.cs.uga.edu:8080/scube/daml/HomeAddress.daml#HomeAddress) -> Addr (http://ovid.cs.uga.edu:8080/scube/daml/HomeAddress.daml#HomeAddress)
0	Address (http://ovid.cs.uga.edu:8080/scube/daml/Address.daml#Address)- **not connected**

Web service Object	
Service Name	Travel Agency

Local intranet

Semantic Process Composition

Semantic Process Composition

Web Process Composition

Composition is the task of combining and linking existing Web Services and other components to create new processes.

Types of Composition

- Static Composition - services to be composed are decided at design time
- Dynamic Composition - services to be composed are decided at run-time

Composition of Web Processes

Web Process Composition

Web Process

Composition

Web Service Discovery

Web Service Integration

Once the desired Web Services have been found (Discovery), mechanisms are needed to facilitate the resolution of structural and semantic differences (integration)

This is because the heterogeneous Web services found in the first step need to interoperate with other components present in a process host

Integration
New Requirements

Web Process Composition

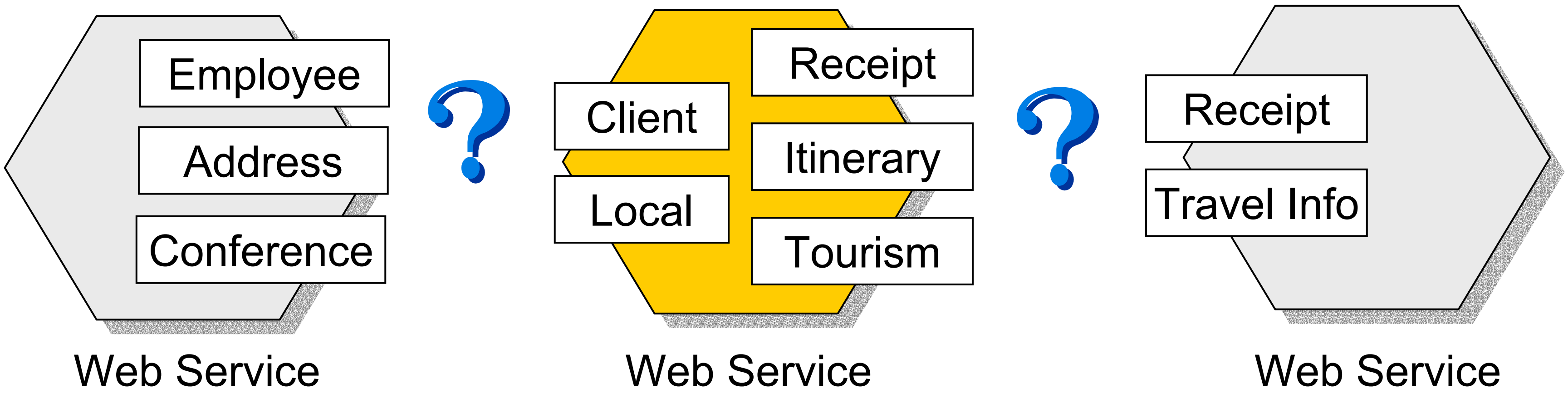
- When Web services are put together
  - Their interfaces need to interoperate.
  - Structural and semantic heterogeneity need to be resolved*.
- **Structural heterogeneity** exists because Web services use different data structures and class hierarchies to define the parameters of their interfaces.
- **Semantic heterogeneity** considers the intended meaning of the terms employed in labeling interface parameters. The data that is interchanged among Web services has to be understood.

* Kashyap and Sheth 1996

Integration
New Requirements

Web Process Composition

How to establish data connections between Web Services interfaces?



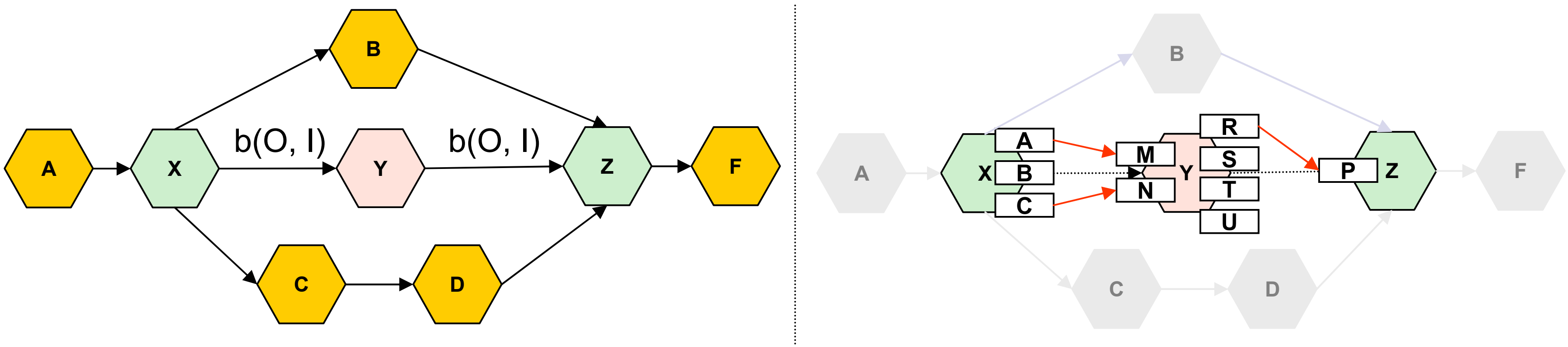
How to establish data connections between the different data structures and class hierarchies of the interface parameters?

How to understand the intended meaning of the terms used in labeling interface parameters?

Web Services
Interfaces

Web Process Composition

- To enhance the integration, Web services need to have their inputs and outputs associated with ontological concepts (annotation).
- This will facilitate the resolution of structural and semantic heterogeneities
- Compute the optimal matching (Bondy and Murty, 1976) using semantic information (Cardoso and Sheth, 2002)



Bipartite graph. Each edge has a weight (semantic similarity).

Web Service QoS

Semantic QoS

Web Processes Quality of Service

Organizations operating in modern markets, such as e-commerce activities, require QoS management.

QoS management is indispensable for organizations striving to achieve a higher degree of competitiveness.

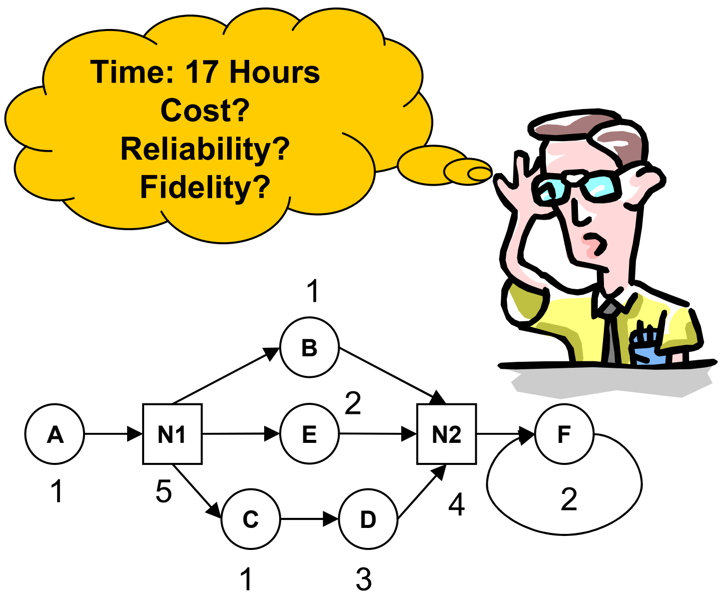
Discovery
New Requirements

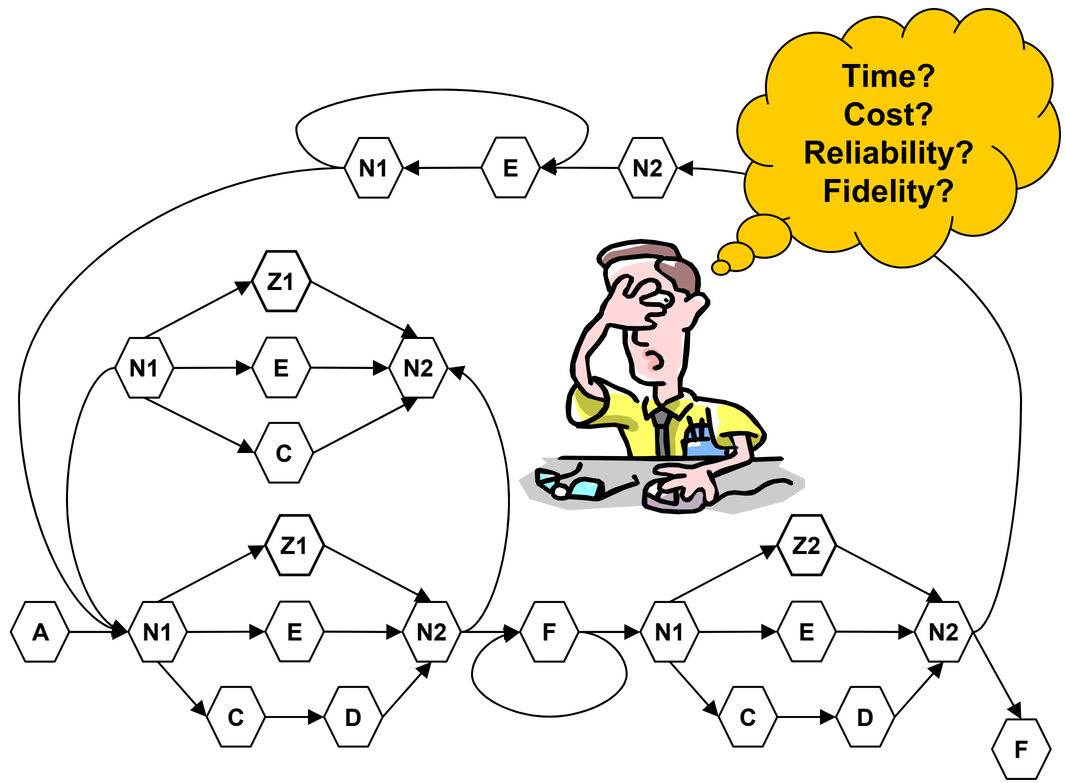
- The autonomy of Web services does not allow for designer to identify their operational metrics at design time.
- Nevertheless, when composing a process it is indispensable to inquire the Web services operational metrics.
- Operational metrics characterize the Quality of Service (QoS) that Web services exhibit when invoked.

QoS
New Requirements

Quality of Service

Before	Now





QoS Semantics

QoS

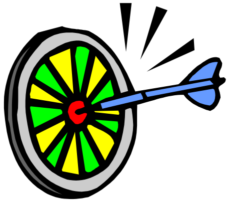
- What ?
  Formally describes operational metrics of a web service/process
- Why ?
  To select the most suitable service to carry out an activity in a process
- How ?
  Using QoS model for web services

[Cardoso and Sheth, 2002]

QoS
Benefits

QoS

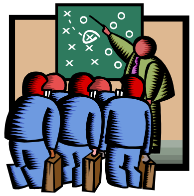
- **Composition** of processes according to QoS objective and requirements.



- **Selection and execution** of processes based on QoS metrics.

- **Monitoring** of processes to assure compliance with initial QoS requirements.

- **Evaluation** of alternative strategies when QoS requirements are violated.

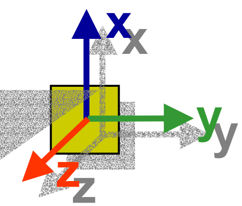


Semantic WP QoS

Research Issues

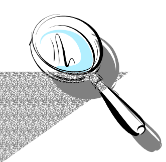
QoS

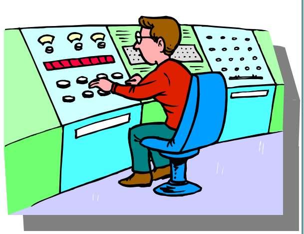
Specification. What dimensions need to be part of the QoS model for processes?



Computation. What methods and algorithms can be used to compute, analyze, and predict QoS?

Monitoring. What king of QoS monitoring tools need to be developed?





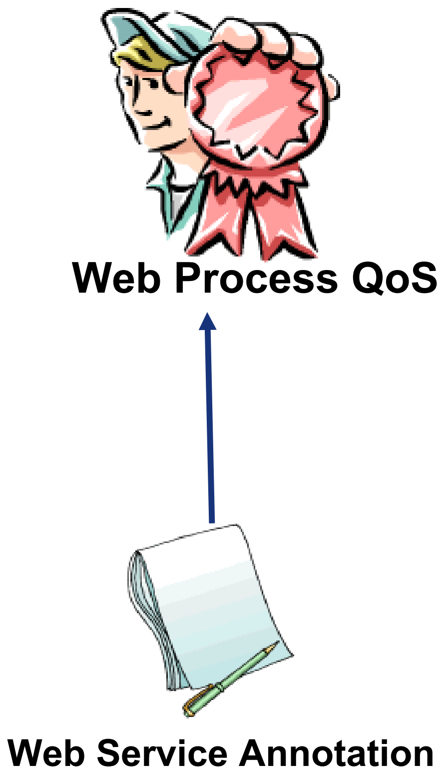
Control. What mechanisms need to be developed to control processes, in response to unsatisfactory QoS metrics?

Web Services

QoS Specification

QoS

- Operational Metrics Specification
  - Operational metrics are described using a QoS model represented with a suitable ontology.
- The specification of Web services operational metrics allows the analysis and computation processes QoS.
- Processes can be designed according to QoS objectives and requirements.
- This allows organizations to translate their strategies into their processes more efficiently.



QoS Management

- End-to-End process analysis
- QoS management is indispensable for organizations striving to achieve a higher degree of competitiveness.
- Based on previous studies* and our experience with business processes, we have constructed a QoS model composed of the following dimensions:
  - Time
  - Cost
  - Reliability
  - Fidelity

*Stalk and Hout,1990;Rommel et al.,1995;Garvin, 1988

QoS Models

QoS

A QoS Model describes non-functional properties of a process

Which dimensions should be part of a QoS model?

- Time
- Cost
- Fidelity
- Reliability

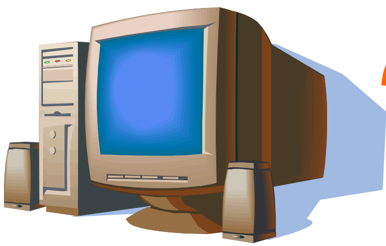
- Price
- Duration
- Repudiation
- Reliability
- Availability

- Security

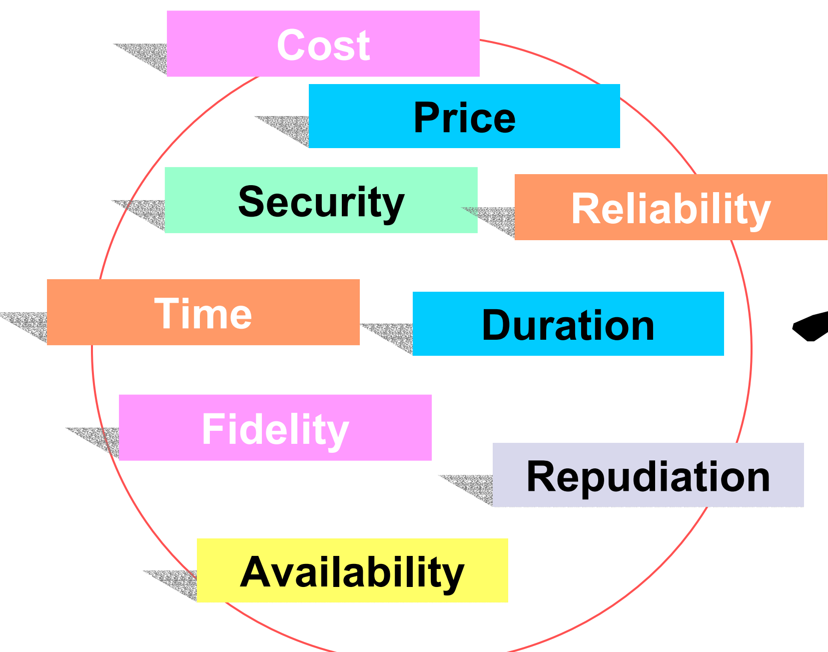
QoS Models and Semantics

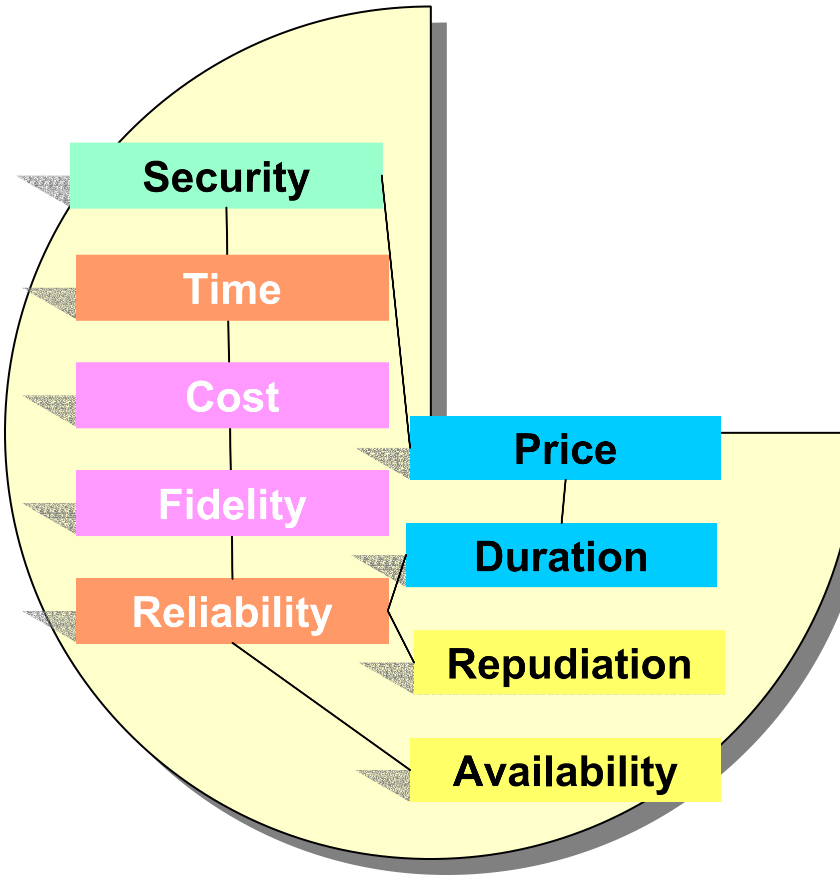
QoS

Use Semantics

?  ?

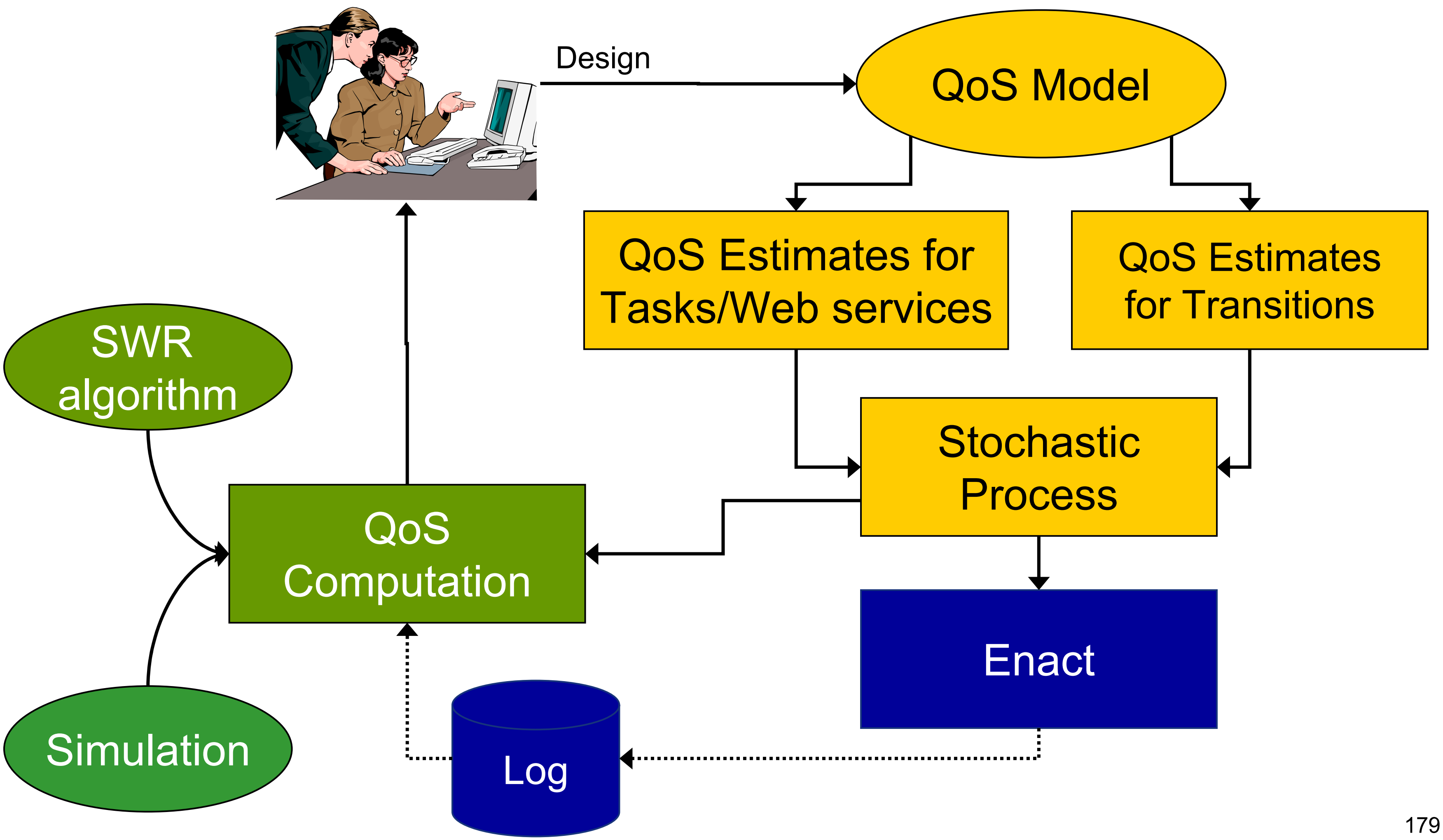
Z#$%&/





QoS in METEOR-S

QoS



QoS
Creation of Estimates

QoS

- To analyze a process QoS, it is necessary to:
  - Create estimated for task QoS metrics and
  - Create estimated for transition probabilities

Once tasks and transitions have their estimates set, algorithms and mechanisms, such as simulation, can be applied to compute the overall QoS of a process.

QoS
Estimates for Web Services

QoS

WS runtime behavior description can be composed of several classes. For example:

QoS Model

Basic class

Distributional class

	Basic class			Distributional class
	Min value	Avg value	Max value	Dist. Function
Time	0.291	0.674	0.895	Normal(0.674, 0.143)
Cost	0	0	0	0.0
Reliability	-	100%	-	1.0
Fidelity.a_i	0.63	0.81	0.92	Trapezoidal(0.7,1,1,4)

Task QoS for an automatic task (SP FASTA task)

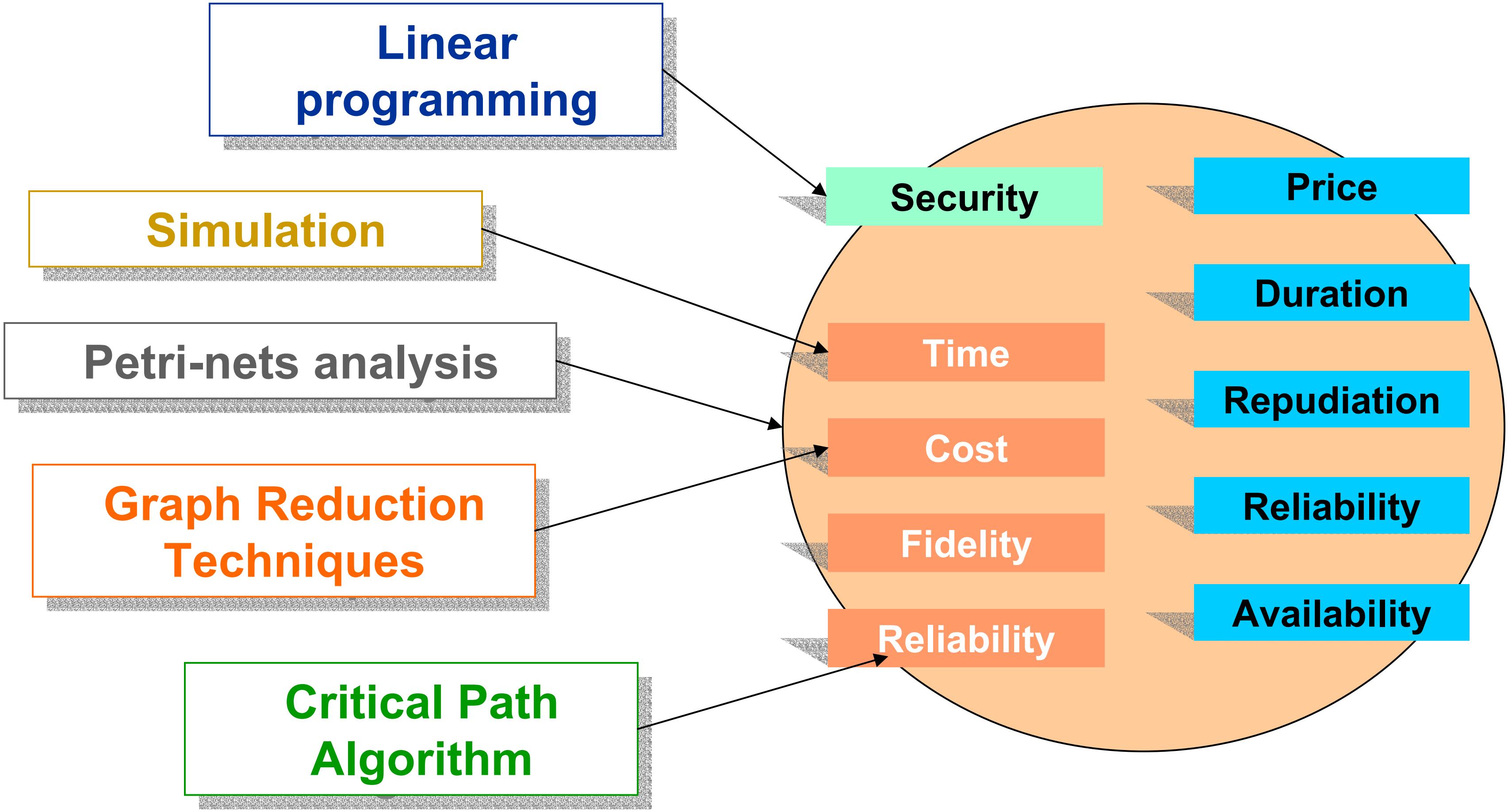
mathematical methods

simulation systems

Web process QoS computation

QoS

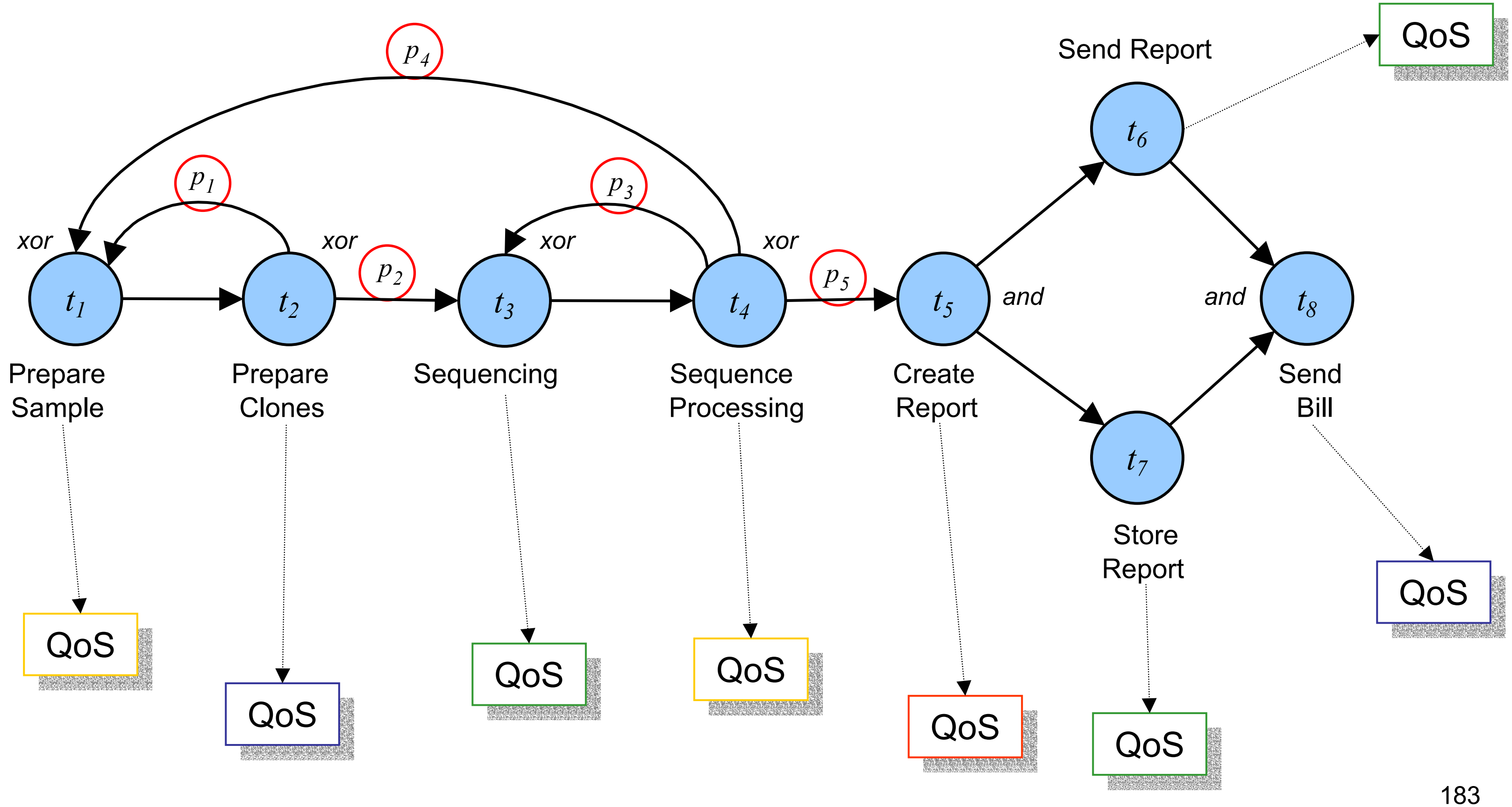
Design time| Runtime



QoS Computation

QoS

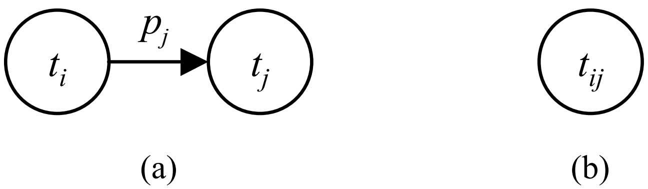
Graph Reduction Technique



QoS Computation

QoS

Graph Reduction Technique



Reduction of a Sequential System

$$T(t_{ij}) = T(t_i) + T(t_j)$$

$$C(t_{ij}) = C(t_i) + C(t_j)$$

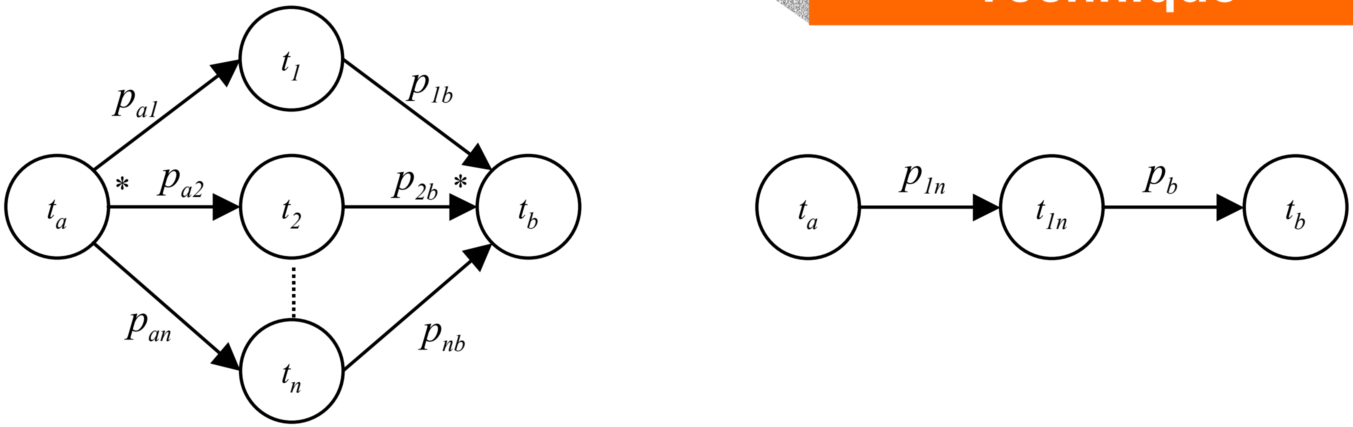
$$R(t_{ij}) = R(t_i) * R(t_j)$$

$$F(t_{ij}).a_r = f(F(t_i), F(t_j))$$

QoS Computation

QoS

Graph Reduction Technique



(a) (b)

Reduction of a Parallel System

$$T(t_{1n}) = \mathrm{Max}_{I\in\{1..n\}}\ \{T(t_i)\}$$

$$C(t_{1n}) = \sum_{1\le i\le n} C(t_i)$$

$$R(t_{1n}) = \prod_{1\le i\le n} R(t_i)$$

$$F(t_{1n}).a_r = f(F(t_1), F(t_2), \ldots, F(t_n))$$

QoS Computation

QoS

Simulation

- While mathematical methods can be effectively used, another alternative is to utilize simulation analysis[1].
- Simulation can play an important role in tuning the QoS metrics of processes by exploring "what-if" questions.
- In our project, these capabilities involve a loosely-coupled integration between the METEOR WfMS and the JSIM simulation system[2].

[1]Miller, Cardoso *et al.* 2002, [2]Nair, Miller *et al.* 1996; Miller, Nair *et al.* 1997; Miller, Seila *et al.* 2000.

QoS Computation
SCET

QoS
Simulation

- SCET (Service Composition and Execution Tool) allows
  - to compose services statically by modeling the process as a digraph in a graphical designer
  - stores the process description as WSFL based specification
  - allows execution of the composed process using Perl
  - supports a simple execution monitoring feature
  - supports performance estimation using JSIM simulation

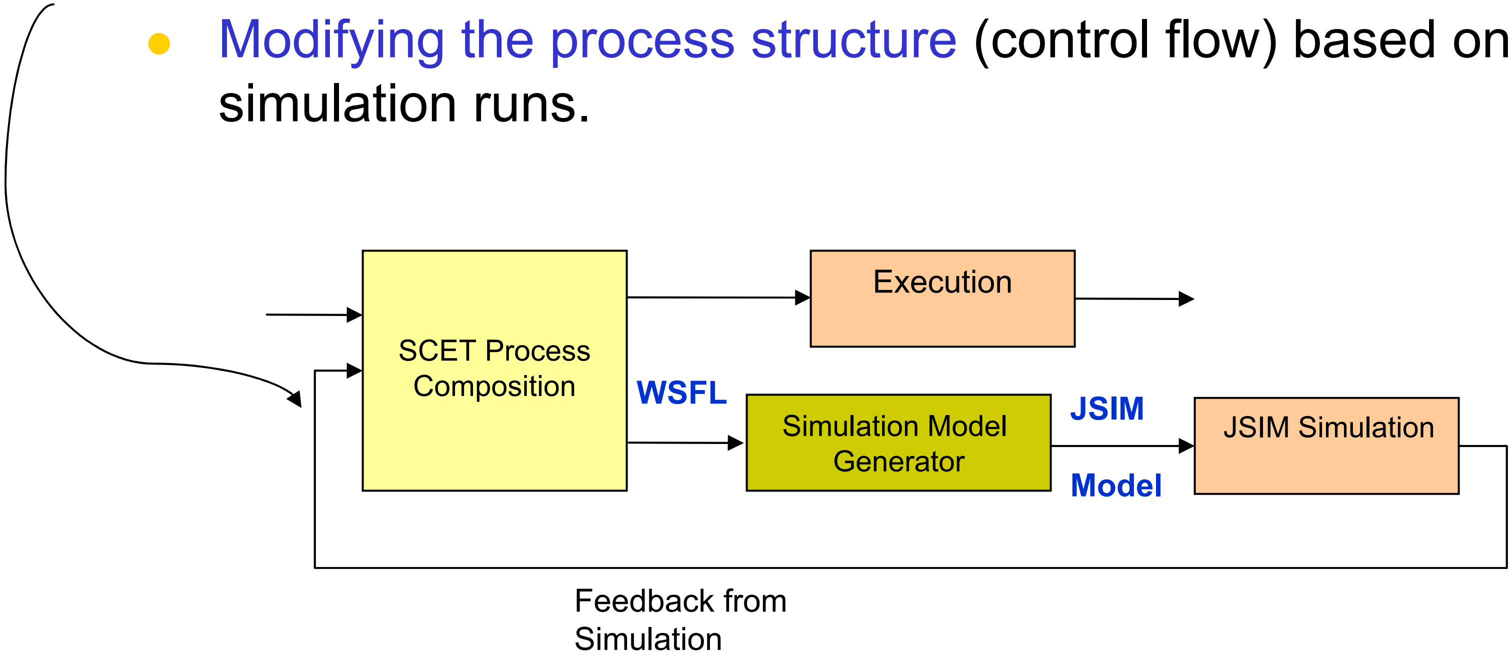
Senthilanand Chandrasekaran, M.Sc. Thesis presented at the Department of Computer Science of the University of Georgia.

QoS Computation

QoS

Simulation

- Simulation provides feedback on processes, allowing the composer to modify his process design by
  - Replacing services which do not satisfy the expected runtime behavior with more suitable Web services.
  - Modifying the process structure (control flow) based on the simulation runs.



Senthilanand Chandrasekaran, M.Sc. Thesis presented at the Department of Computer Science of the University of Georgia.

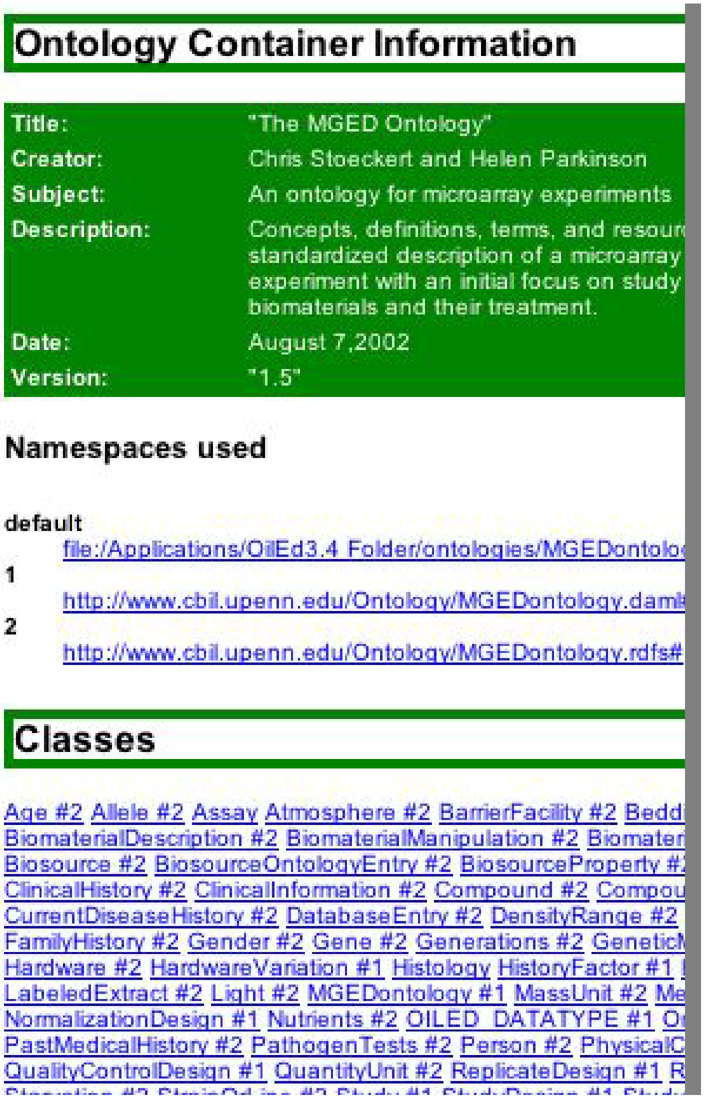
Examples of Ontologies

Examples of Real Ontologies

MGED Ontology

- The MGED Ontology
  - Provide standard terms for the annotation of microarray experiments.
  - Terms will enable unambiguous descriptions of how the experiment was performed.
  - 212 classes, 101 properties.
- The MGED Ontology is being developed within the microarray community to provide consistent terminology for experiments.
- This community effort has resulted in a list of multiple resources for many species.
  - Approximately 50 other ontologies for different species
- The concepts are structured in DAML+OIL and available in other formats (rdfs)

The MGED Ontology is Structured in DAML+OIL using OILed 3.4



Ontology Container Information

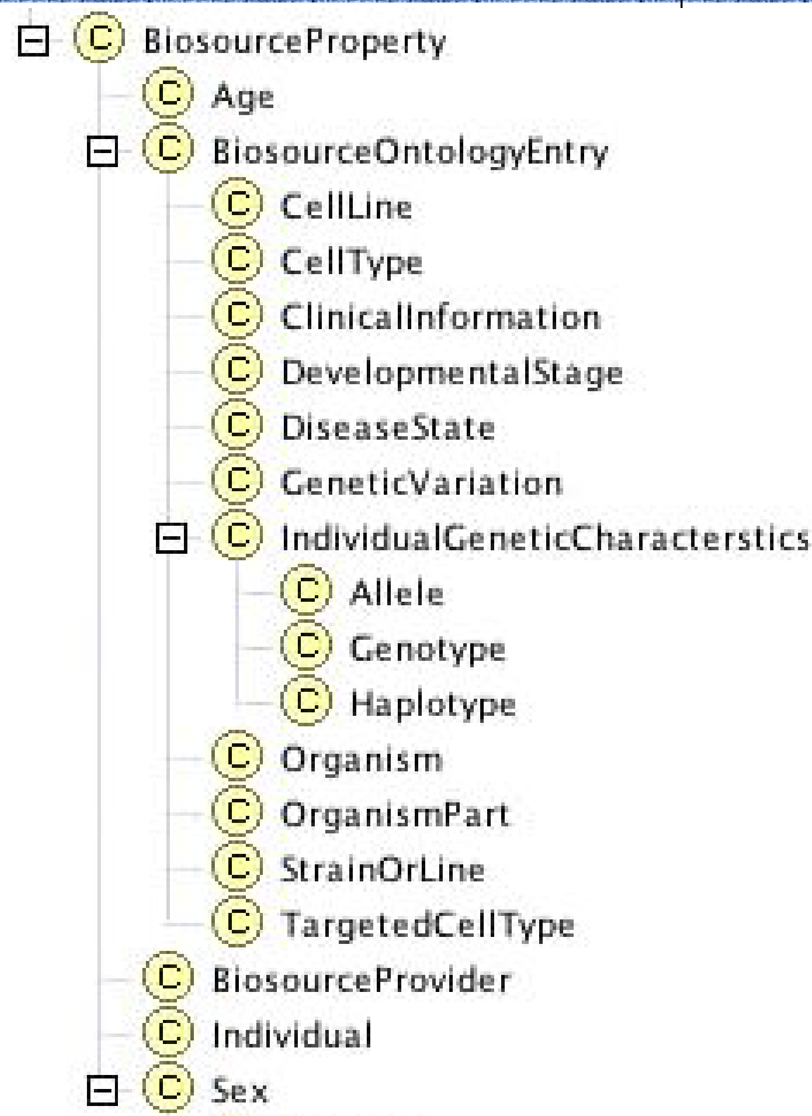
Title:	"The MGED Ontology"
Creator:	Chris Stoeckert and Helen Parkinson
Subject:	An ontology for microarray experiments
Description:	Concepts, definitions, terms, and resour standardized description of a microarray experiment with an initial focus on study biomaterials and their treatment.
Date:	August 7,2002
Version:	"1.5"

Namespaces used

default
file:/Applications/OilEd3.4 Folder/ontologies/MGEDontolo
1
http://www.cbil.upenn.edu/Ontology/MGEDontology.daml
2
http://www.cbil.upenn.edu/Ontology/MGEDontology.rdfs#

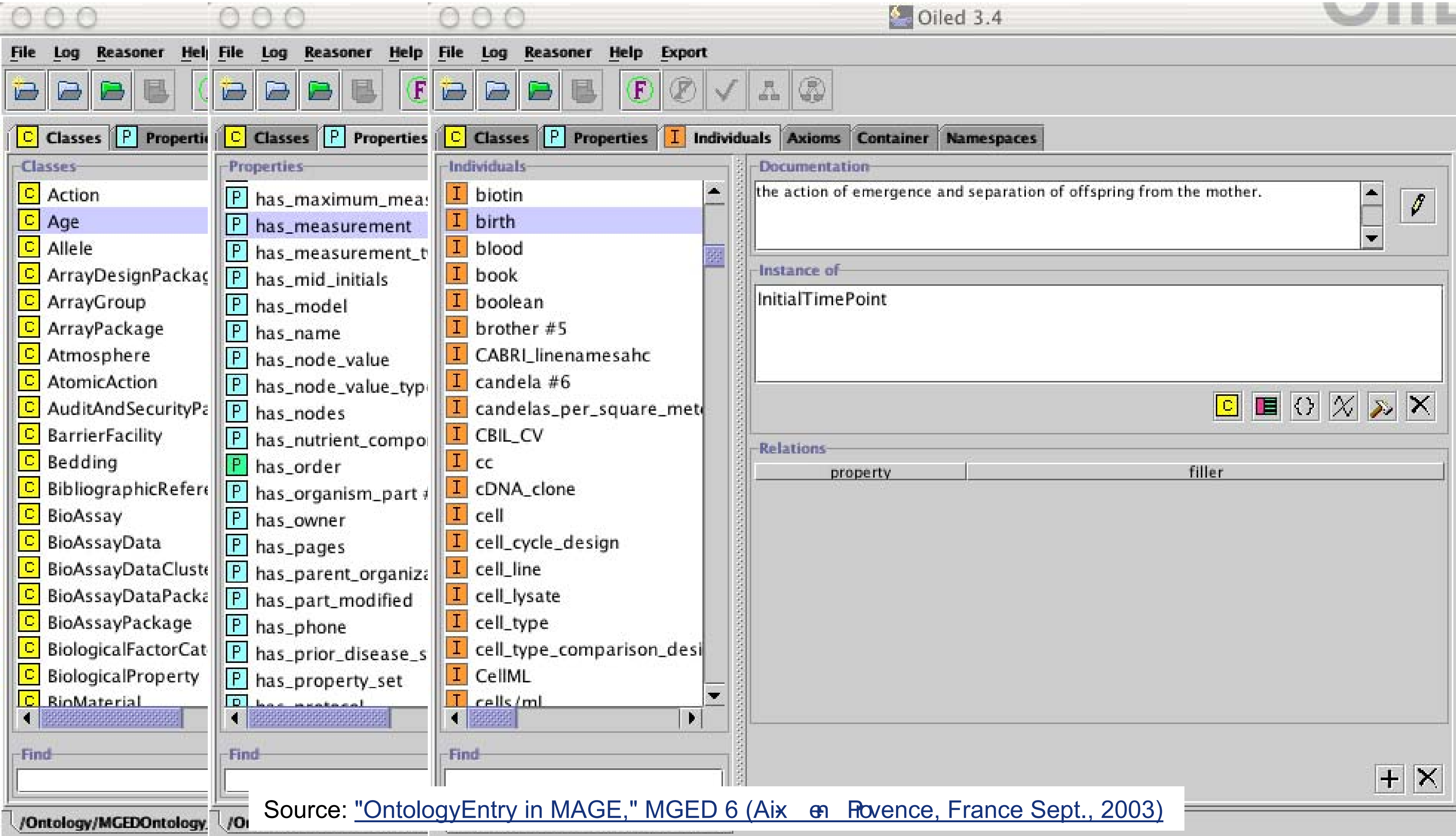
Classes

Age #2 Allele #2 Assay Atmosphere #2 BarrierFacility #2 Beddi
BiomaterialDescription #2 BiomaterialManipulation #2 Biomater
Biosource #2 BiosourceOntologyEntry #2 BiosourceProperty #
ClinicalHistory #2 ClinicalInformation #2 Compound #2 Compou
CurrentDiseaseHistory #2 DatabaseEntry #2 DensityRange #2
FamilyHistory #2 Gender #2 Gene #2 Generations #2 Genetic
Hardware #2 HardwareVariation #1 Histology HistoryFactor #1
LabeledExtract #2 Light #2 MGEDontology #1 MassUnit #2 Me
NormalizationDesign #1 Nutrients #2 OILED_DATATYPE #1 O
PastMedicalHistory #2 PathogenTests #2 Person #2 PhysicalC
QualityControlDesign #1 QuantityUnit #2 ReplicateDesign #1 R



Source: "The MGED Ontology is an Experimental Ontology," 5th Annual Bio-Ontologies meeting (Edmonton, Canada Aug. 2002)

MGED Ontology consists of classes, properties, and individuals (instances)



Source: "OntologyEntry in MAGE," MGED 6 (Aix en Provence, France Sept., 2003)

MGED Ontology: BiomaterialDescription: BiosourceProperty: Age

class **Age #1**

namespace:
http://www.cbil.upenn.edu/Ontology/MGEDontology.rdfs#

documentation:
The time period elapsed since an identifiable point in the life cycle of an organism. If a developmental stage is specified, the identifiable point would be the beginning of that stage. Otherwise the identifia

type:
primitive

superclasses:
BiosourceProperty #1

constraints:
restriction initial_time_point #
restriction has_measurement

used in properties:
initial_time_point #1

class **Measurement #1**

namespace:
http://www.cbil.upenn.edu/Ontology/MGEDontology.rdfs#

documentation:
Measured values and units.

type:
primitive

superclasses:
MGEDontology #2

constraints:
restriction value #1 has-class thing
restriction has_units #1 has-class Unit #1
restriction measurement_type #1 has-class one-of (change #1 absolute #1)

known subclasses:
BiomaterialMeasurement #1

used in classes:
Age #1
BiomaterialMeasurement #1
BiomaterialPreparation #1
ClinicalHistory #1
CompoundBasedTreatment #1
GrowthCondition #1

used in properties:
measurement_type #1

Source: "The MGED Ontology is an Experimental Ontology," 5th Annual Bio- Ontologies meeting (Edmonton, Canada Aug. 2002)

Examples of Real Ontologies

OBO

- OBO (Open Biological Ontologies)
  - Is an umbrella organization for structured shared controlled vocabularies and ontologies for use within the genomics and proteomics domains.

The ontologies must be **open** and can be used by all without any constraint other than that their origin must be acknowledged and they can not be altered and redistributed under the same name.

The ontologies are in, or can be instantiated in, a **common shared syntax**. This may be either the GO syntax, extensions of this syntax, or OWL.

The ontologies share an **unique identifier space**.

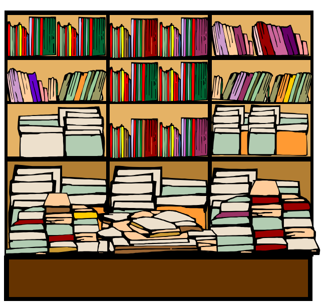
The ontologies include textual **definitions** of their terms.

The ontologies are **orthogonal** to other ontologies already lodged with OBO.

Examples of Real Ontologies
GO Ontology

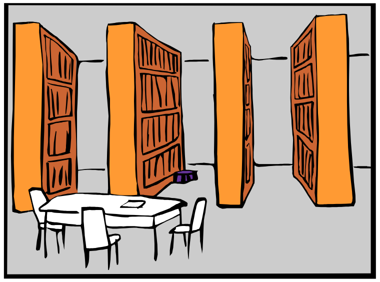
- Gene Ontology (**GO**)
  - Describes gene products in terms of their
    - Associated biological processes,
    - cellular components and
    - Molecular functions in a species-independent manner.

GO format - flat files, XML, MySQL



Component ontology
1379 terms
212 KB

Process ontology
8151 terms
4.82 MB



Function ontology
7278 terms
1.16 MB

function.ontology

<molecular_function ; GO:0003674

%antioxidant activity ; GO:0016209

%glutathione dehydrogenase (ascorbate) activity ; GO:0045174 ; EC:1.8.5.1 ; MetaCyc:1.8.5.1-RXN ; synonym:dehydroascorbate reductase % electron carrier activity ; GO:0009055 % glutathione disulfide oxidoreductase activity ; GO:0015038 % oxidoreductase activity\, acting on sulfur group of donors\, quinone or similar compound as acceptor ; GO:0016672

%glutathione-disulfide reductase activity ; GO:0004362 ; EC:1.8.1.7 ; MetaCyc:1.8.1.7-RXN ; MetaCyc:GLUTATHIONE-REDUCT-NADPH-RXN ; synonym:glutathione reductase (NADPH) activity ; synonym:glutathione-disulphide reductase activity % electron transporter activity ; GO:0005489 % glutathione disulfide oxidoreductase activity ; GO:0015038 % oxidoreductase activity\, acting on NADH or NADPH\, disulfide as acceptor ; GO:0016654

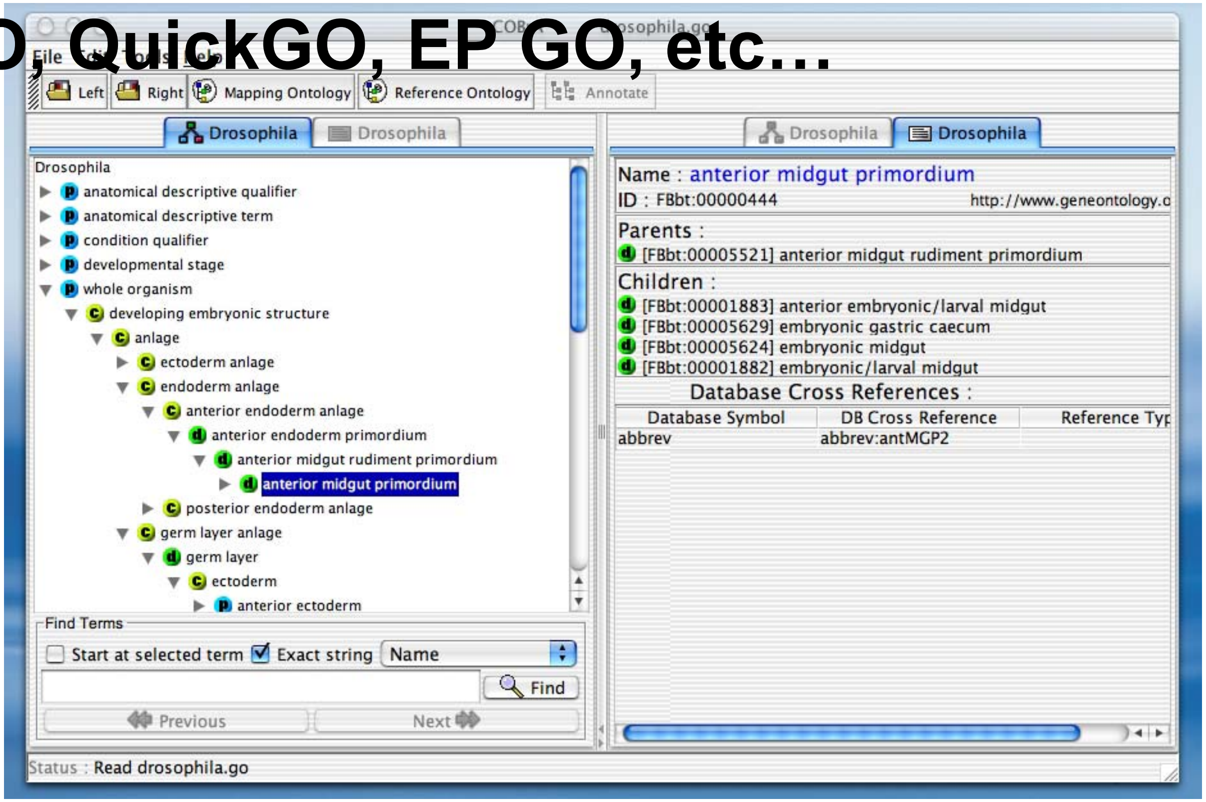
%peroxidase activity ; GO:0004601, GO:0016685, GO:0016686, GO:0016687 ; EC:1.11.1.7 ; MetaCyc:PEROXID-RXN ; synonym:eosinophil peroxidase activity ; synonym:lactoperoxidase activity ; synonym:myeloperoxidase activity % oxidoreductase activity\, acting on peroxide as acceptor ; GO:0016684

%thioredoxin-disulfide reductase activity ; GO:0004791 ; EC:1.8.1.9 ; MetaCyc:1.8.1.9-RXN ; MetaCyc:THIOREDOXIN-REDUCT-NADPH-RXN ; synonym:thioredoxin disulfide reductase activity ; synonym:thioredoxin reductase (NADPH) activity ; synonym:thioredoxin-disulphide reductase activity % electron transporter activity ; GO:0005489 % oxidoreductase activity\, acting on NADH or NADPH\, disulfide as acceptor ; GO:0016654

Examples of Real Ontologies

GO Ontology

- Gene Ontology Editors
  - **DAG-Edit, COBrA**
- Gene Ontology Browsers
  - **AmiGO, MGI GO, QuickGO, EP GO, etc…**
- Other tools
  - Aprox. 30 tools



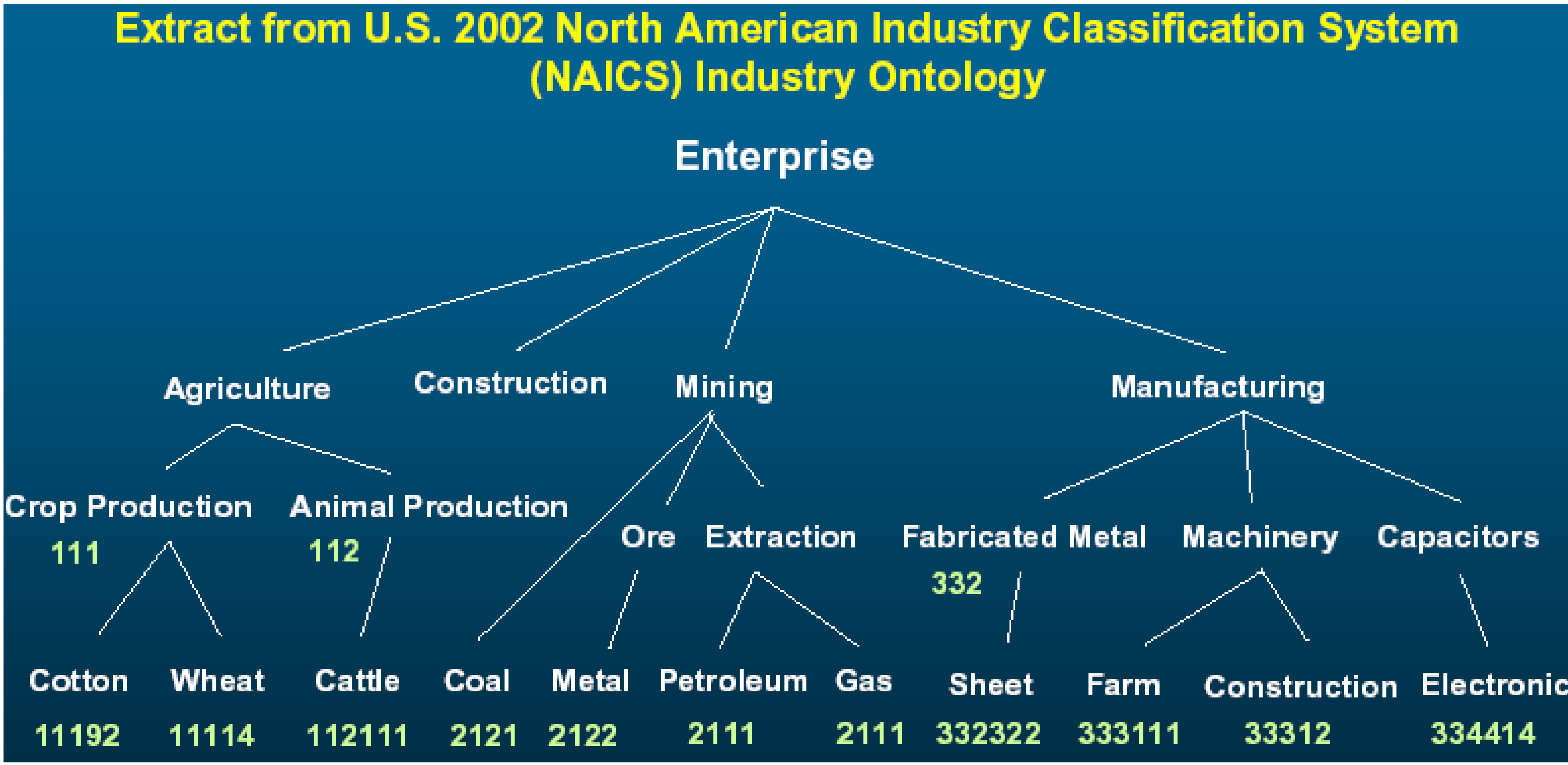
Examples of Toy Ontologies
DAML library

- DAML Ontology Library
  - 282 ontologies
- A few examples
  - http://cicho0.tripod.com/**cs_Courses_ont**
  - http://daml.umbc.edu/ontologies/**calendar-ont.daml**
  - http://mnemosyne.umd.edu/~aelkiss/**weather-ont.daml**
  - http://ontolingua.stanford.edu/doc/chimaera/ontologies/**wines.daml**
  - http://www.ai.sri.com/daml/ontologies/sri-basic/1-0/**Person.daml**
  - http://www.kestrel.edu/DAML/2000/12/**TIME.dam**l
  - http://www.daml.org/2002/08/nasdaq/**nasdaq-on**t
  - http://www.daml.org/2001/10/html/**airport-ont**
  - http://www.daml.org/2001/10/html/**nyse-ont**
  - http://www.daml.ecs.soton.ac.uk/ont/**currency.daml**
  - http://horus.isx.com/markup/2002/01/**countries2.rdf**
  - …

Examples of Toy Ontologies
wine.daml

- Classes
  - ALSATIAN-WINE, AMERICAN-WINE, ANJOU, AUSTRALIAN-REGION, BEAUJOLAIS, BLAND-FISH, BORDEAUX, BORDEAUX-REGION, BOURGOGNE-REGION, BURGUNDY, CABERNET-FRANC, CALIFORNIA-WINE, …
- Properties
  - BODY, COLOR, COURSE, DRINK, FLAVOR, FOOD, GRAPE-SLOT, MAKER, REGION, SUGAR

Ontologies Needed



Ontologies Needed

How do I know that your

<POID>

is the same data element concept as my <PurchaseOrderIdentifier>?

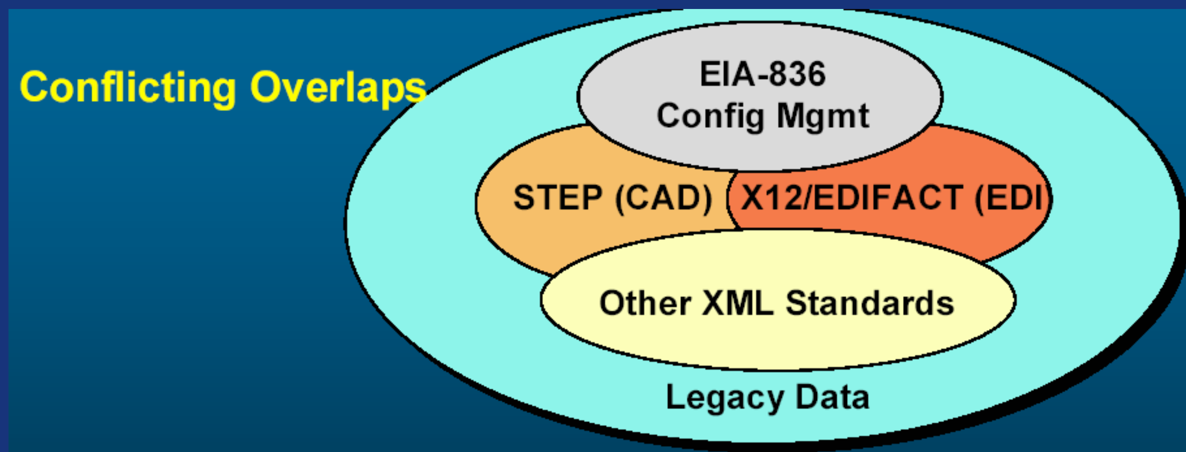
How does the doctor's medical record system knows that the data in

<currentmedications>

is the same as their systems' element labeled

<patientpharmacology>?

Ontologies Needed



- Though semantically equal, the following are 4 different XML tag names
  - <PARTNUMBER>111-222-333</PARTNUMBER>
  - <partNumber>111-222-333</partNumber>
  - <PartNumber>111-222-333</PartNumber>
  - <partnumber>111-222-333</partnumber>

Universal Data Element Framework

UDEF

- The Universal Data Element Framework (UDEF)
  - cross-industry metadata identification
  - designed to facilitate convergence and interoperability among e-business and other standards.
  - provide a means of real-time identification for semantic equivalency
  - seeks only be an attribute in the data element

UDEF ID = ebXML UID	EIA-836	X12 (EDI)	Vendor A
9_5.8	Product Part Identifier	Product/Service ID	Part No
9_9	Product Name	Product/Service Name	
y.3_9		Entity (Supplier) Name	Supplier
e.2_8			
f.g.9_			
2_33			

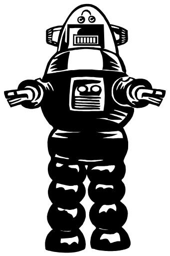
<ProductPartIdentifier PRD:GUID="9_5.8">123-456-789</ProductPartIdentifier>

<ProductServiceID PRD:GUID="9_5.8">123-456-789</ProductServiceID>

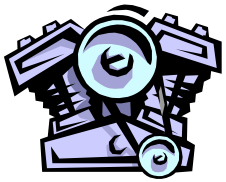
<PartNo PRD:GUID="9_5.8">123-456-789</PartNo>

Ron Schuldt, Co-Chair, AIA Electronic Enterprise Working Group, **XML Standards Relevant to the Aerospace Industry**

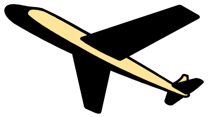
Ontology Domains



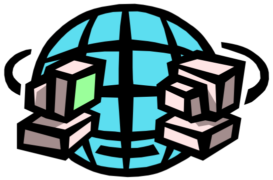
- Aerospace and defense,
- Automotive,



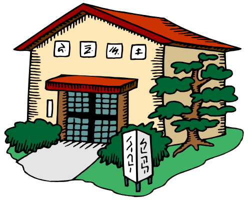
- Consumer products,
- Travel,



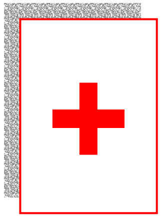
- Telecommunications



- Engineering and construction,



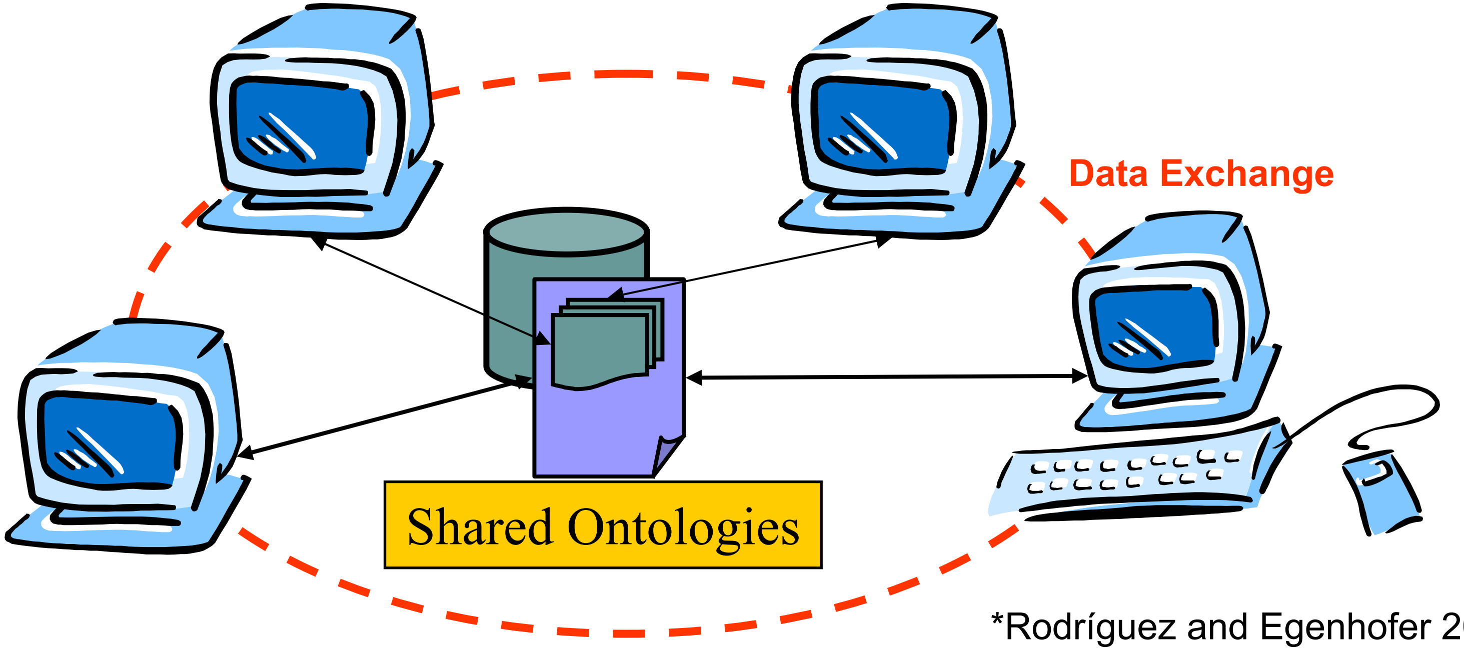
- Banking
- Health care



- …

Ontologies-based approaches
Shared Ontologies

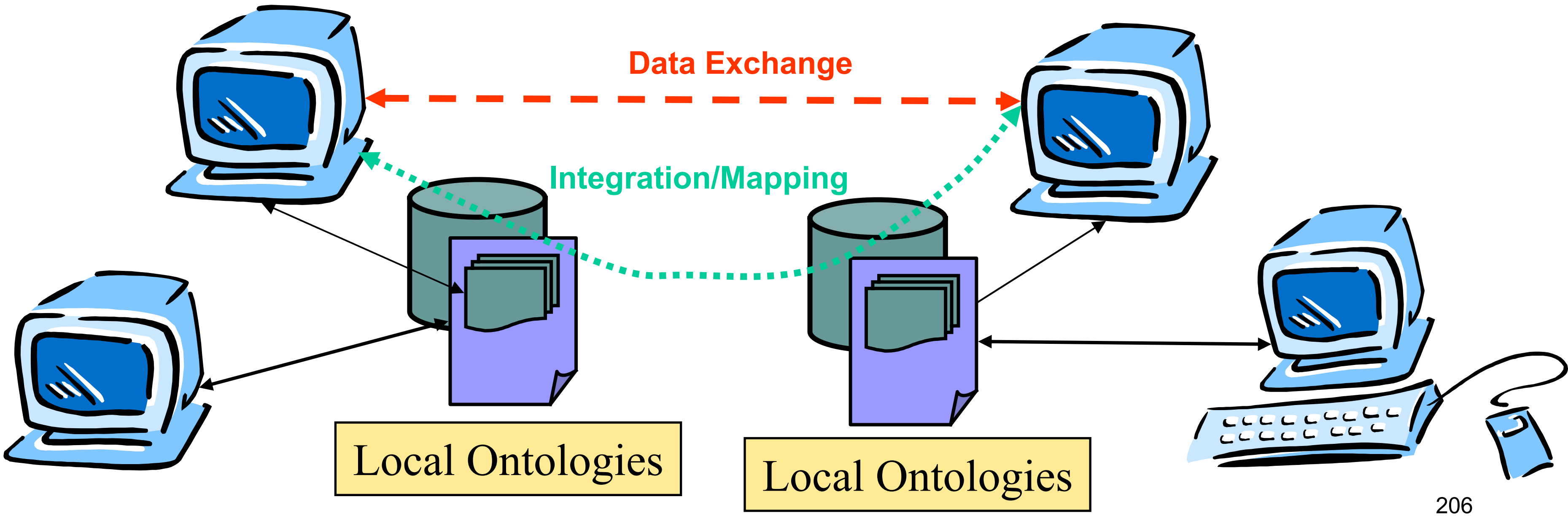
- Autonomous systems are required to commit to a shared ontology, and compromises are difficult to maintain when new concepts are added*.
- Even though a shared ontology ensures total integration, constructing such an ontology is costly, if not impractical.



*Rodríguez and Egenhofer 2002

Ontologies-based approaches
Non-Shared Ontologies

- Since the Web is a distributed infrastructure with autonomous systems, it is not reasonable to expect that all the systems will commit to shared ontologies.
- Instead, autonomous systems will use non-shared ontologies.
- This will require the integration and mapping of ontologies.



OWL Language

OWL

- OWL is a language for defining Web Ontologies
- The OWL language is a revision of the DAML+OIL
- DAML+OIL
  - Extension of RDFS
  - Allows machine understanding and automated reasoning.

OWL

- OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics.

OWL

- OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full
- *OWL Lite*
  - Classification hierarchy and simple constraints
- *OWL DL*
  - Maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computed) and decidability (all computations will finish in finite time)
- *OWL Full*
  - Maximum expressiveness and the syntactic freedom of RDF with no computational guarantees.

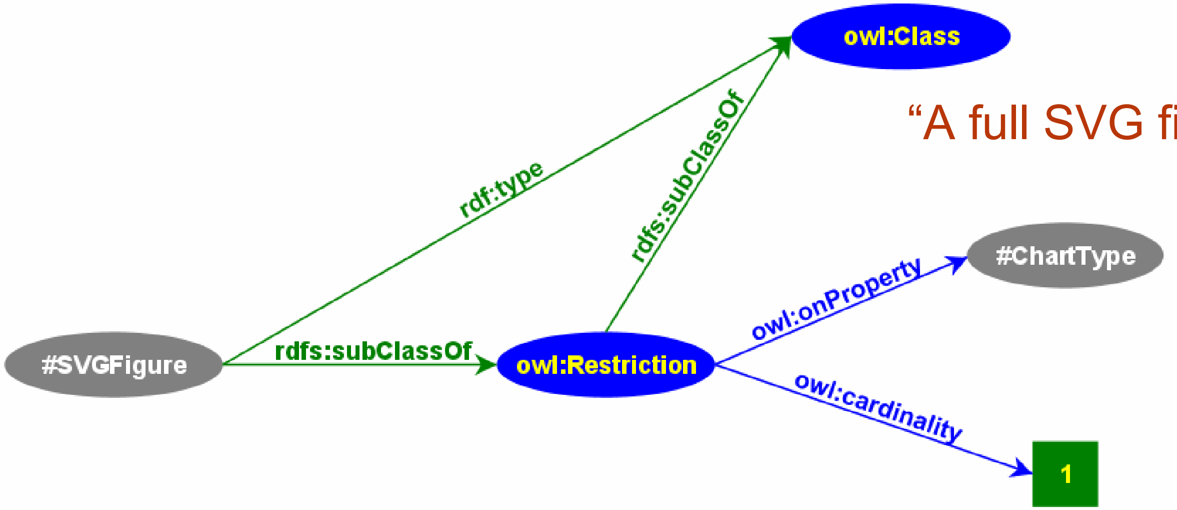
Stack of W3C recommendations

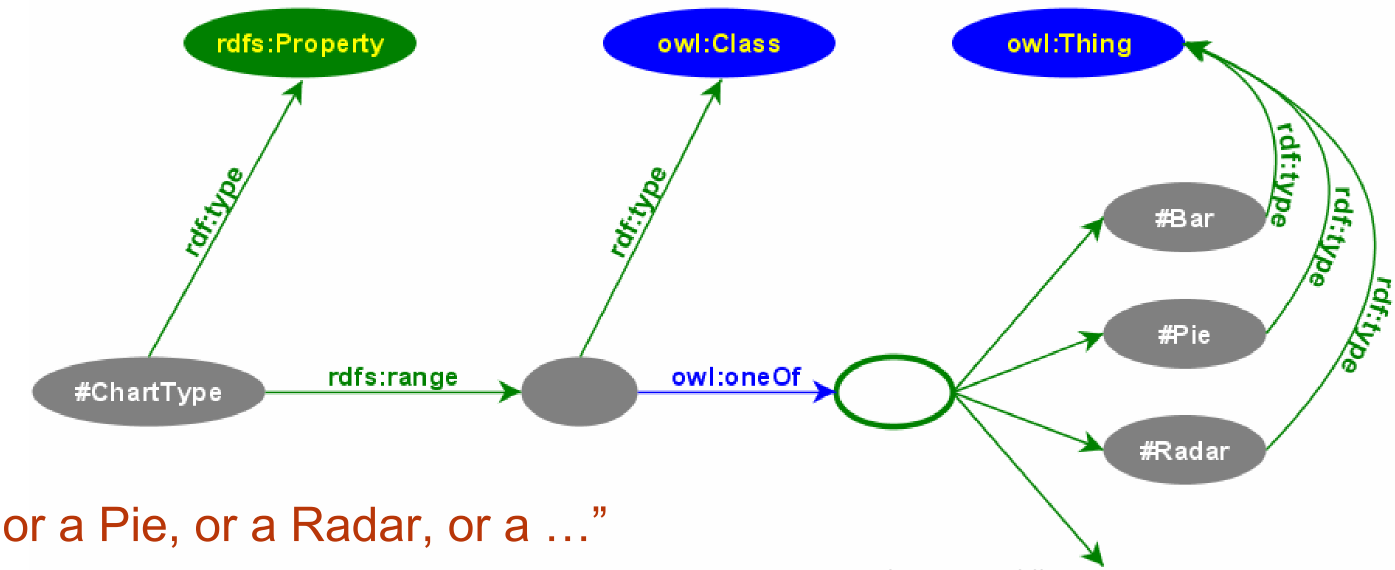
- XML
  - Syntax for structured documents
  - No semantic constraints on the meaning of these documents
- XML Schema
  - Language for defining the structure of XML documents
- RDF
  - Data model for objects and relations between them
  - Provides a simple semantics for this data model
  - Data models represented in an XML syntax.
- RDF Schema
  - A vocabulary for describing properties and classes of RDF resources
- OWL
  - Adds more vocabulary for describing properties and classes
  - For example: relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, characteristics of properties (e.g. symmetry), and enumerated classes.

http://www.w3.org/TR/2003/PR-owl-features-20031215/

OWL example

“A full SVG figure must have *one* chart type“



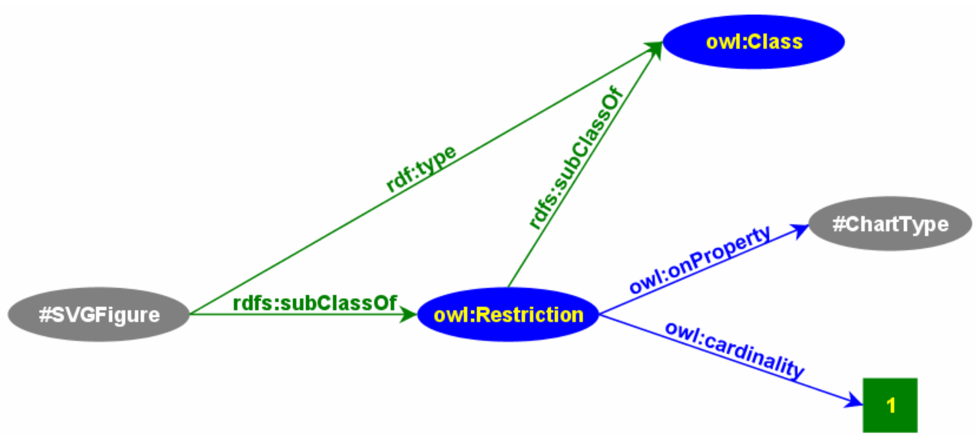


“A char type is a Bar, or a Pie, or a Radar, or a …”

http://www.w3.org/Consortium/Offices/Presentations/RDFTutorial/

OWL example

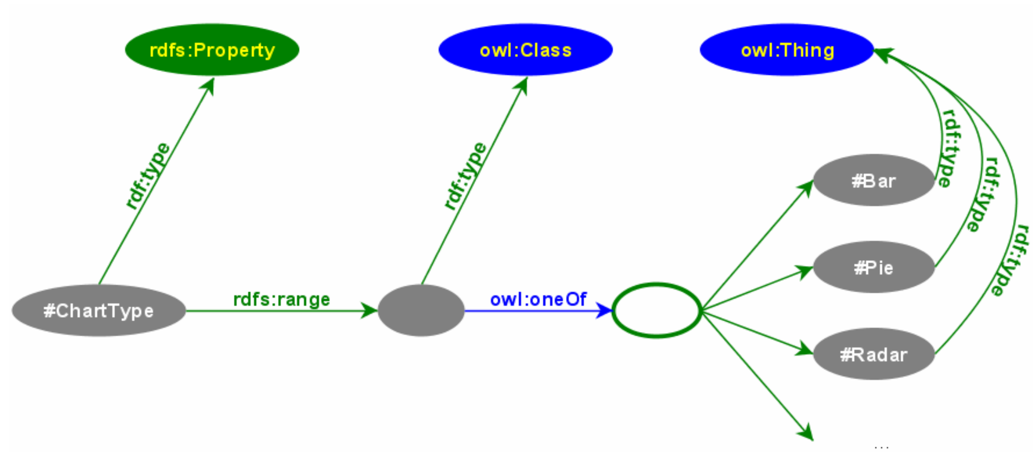
```
<owl:Class rdf:ID="SVGFigure">
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:about="#ChartType"/>
         <owl:cardinality
            rdf:dataype="...#nonNegativeInteger">
            1
         </owl:cardinality>
      </owl:Restriction>
   </rdfs:subClassOf>
</owl:Class>
```



http://www.w3.org/Consortium/Offices/Presentations/RDFTutorial/

OWL example

```
<rdf:Property rdf:ID="ChartType">
  <rdf:range>
    <owl:Class>
      <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:ID="Bar">
        <owl:Thing rdf:ID="Pie">
        <owl:Thing rdf:ID="Radar">
        …
      </owl:oneOf>
    </owl:Class>
  </rdf:range>
</rdf:Property>
```



http://www.w3.org/Consortium/Offices/Presentations/RDFTutorial/

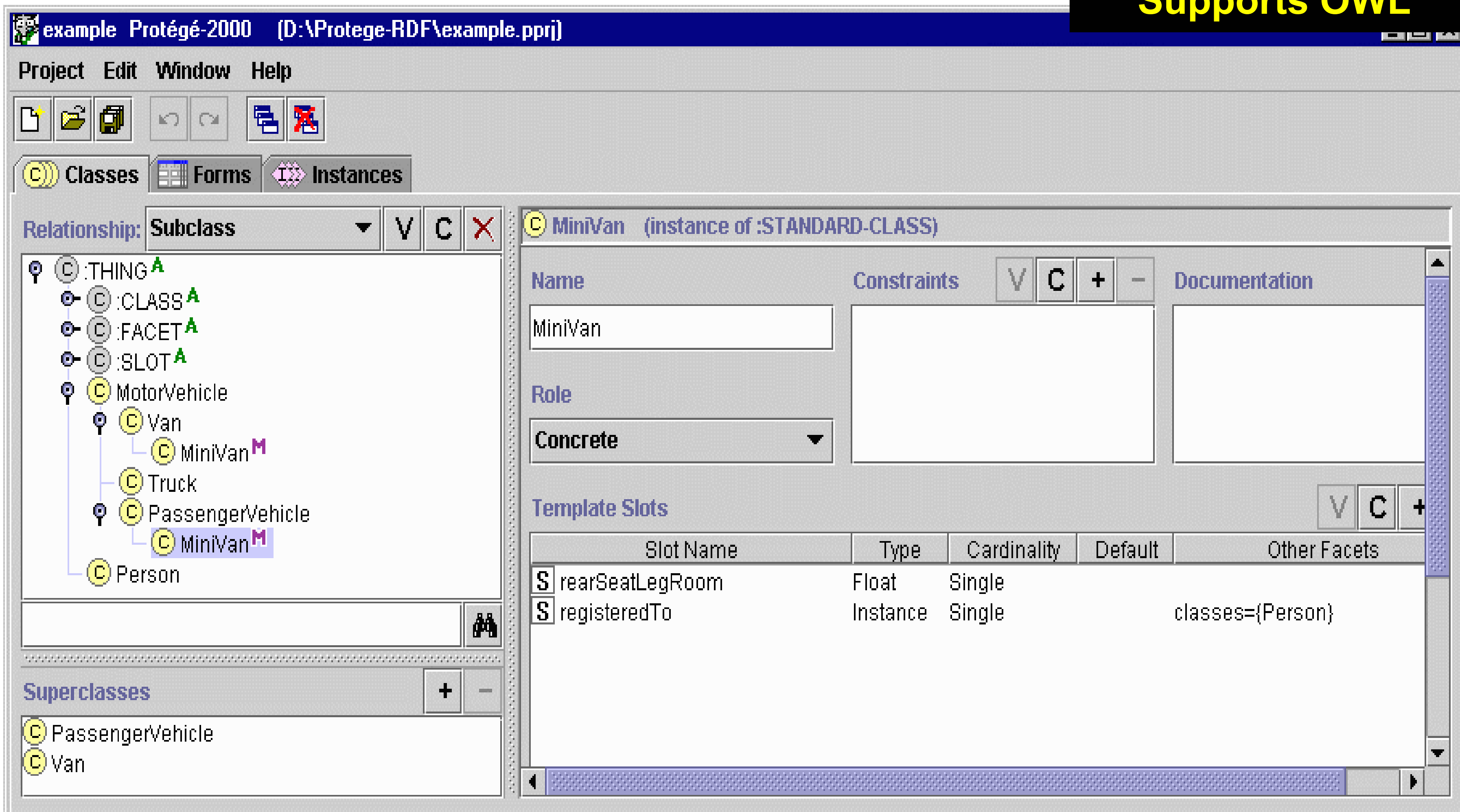
Ontology Editors

Tools: Ontology Editors

- More than 50 applications. A few examples,
  - Protégé 2000
  - OILed
  - WebOnto
  - GKB-Editor
  - Chimaera
  - …

Protégé 2000

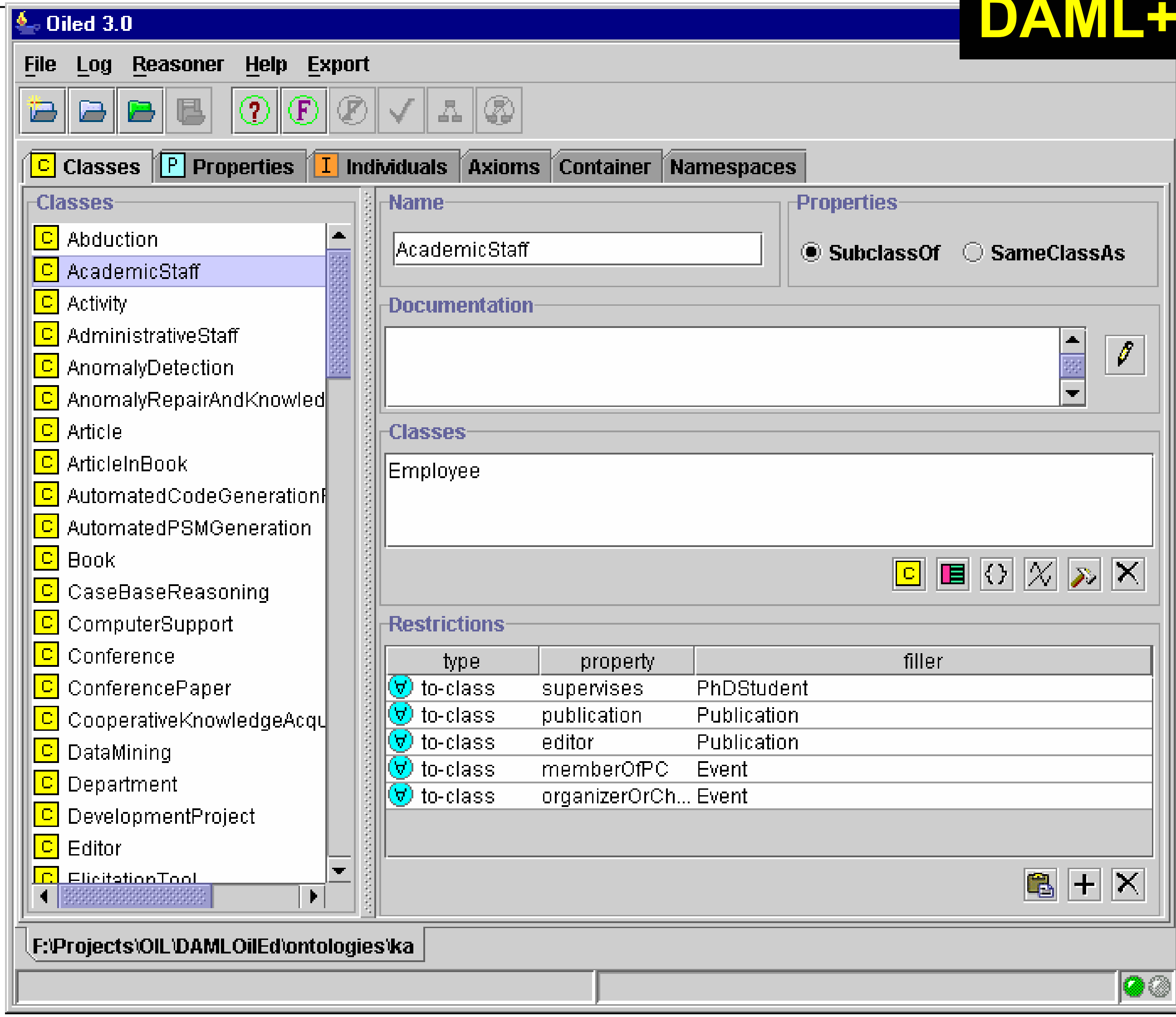
Supports OWL



http://protege.stanford.edu/

OilEd

DAML+OIL



http://oiled.man.ac.uk/

Chimaera

DAML+OIL

Analysis: 15 active commands

Class: 2 active commands

Decomposition: One active command

- One active command
- Add to decomposition [Ctrl-Sh-D]
- ----------
- [Inactive] Create disjoint decomposition - (decompositions are selected)
- [Inactive] Create exhaustive decomposition - (decompositions are selected)
- [Inactive] Create partition - (decompositions are selected)
- [Inactive] Remove decomposition - (classes are selected)
- [Inactive] Remove from decomposition - (at least one class is not in the decomposition)
- [Inactive] Subclasses are disjoint - (decompositions are selected)
- [Inactive] Subclasses are exhaustive - (decompositions are selected)
- [Inactive] Subclasses form a partition - (decompositions are selected)
- [Inactive] Upgrade decomposition to partition - (classes are selected)

File: 10 active com

Taxonomy: No a

View: 16 active cc

Name: Transpor ... tor

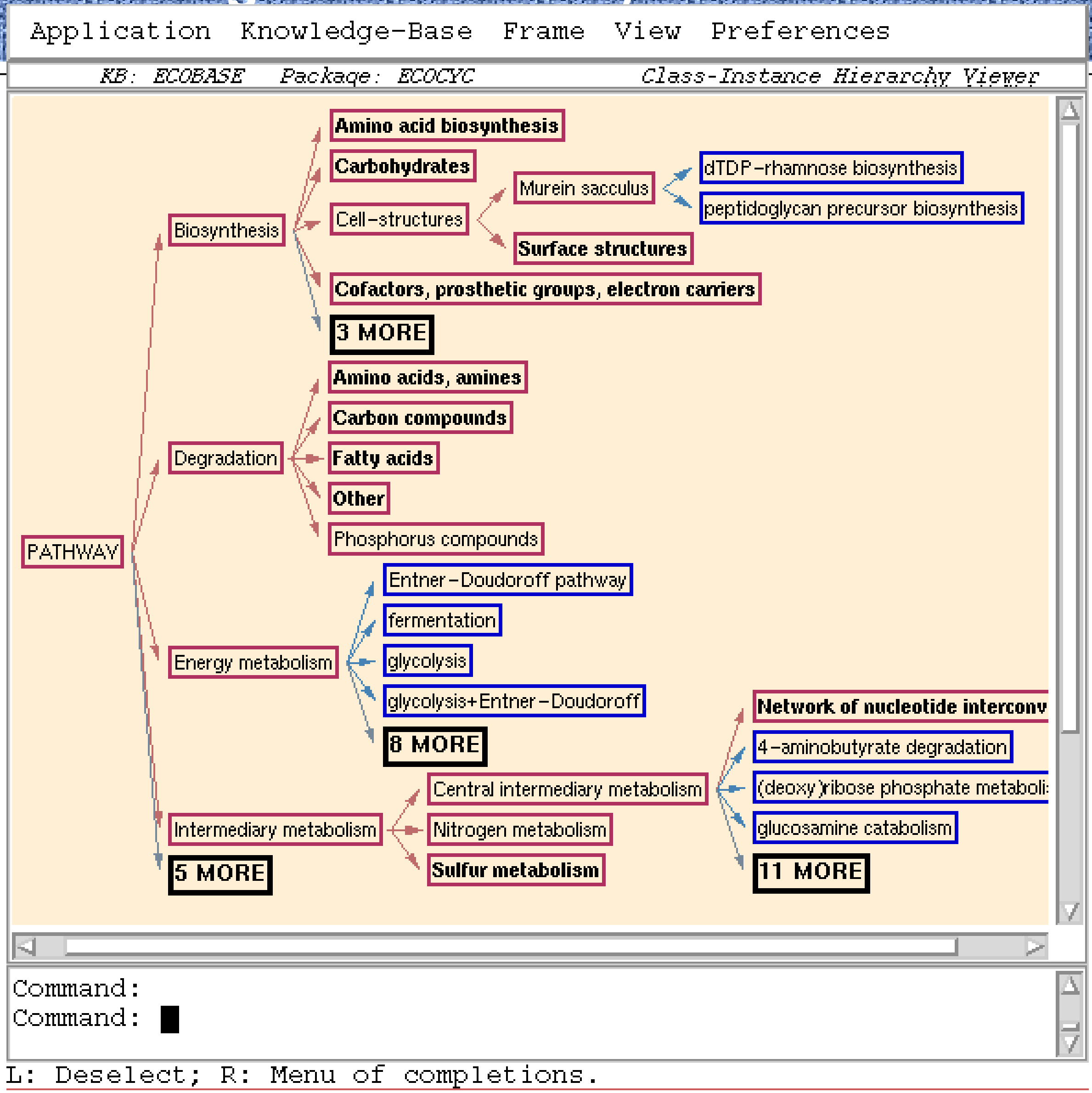
Economy-Sec
- Basic Mater
- Financial Sector {from Cmu-Web-Ontology}
- Services Sector {from Cmu-Web-Ontology}
- Utilities Sector {from Cmu-Web-Ontology}

- Agricultural-Sector {from World-Fact-Book}
- Industrial-Sector {from World-Fact-Book}
- Service-Industry {from World-Fact-Book}

- Capital Goods Sector {from Cmu-Web-Ontology}
- Conglomerates Industry {from Cmu-Web-Ontology}
- Consumer Cyclical Sector {from Cmu-Web-Ontology}
- Consumer Non-cyclical Sector {from Cmu-Web-Ontology}
- Energy Sector {from Cmu-Web-Ontology}
- Healthcare Sector {from Cmu-Web-Ontology}
- Technology Sector {from Cmu-Web-Ontology}
- **Transportation Sector** [Go] {from Cmu-Web-Ontology}

http://www.ksl.stanford.edu/software/chimaera/

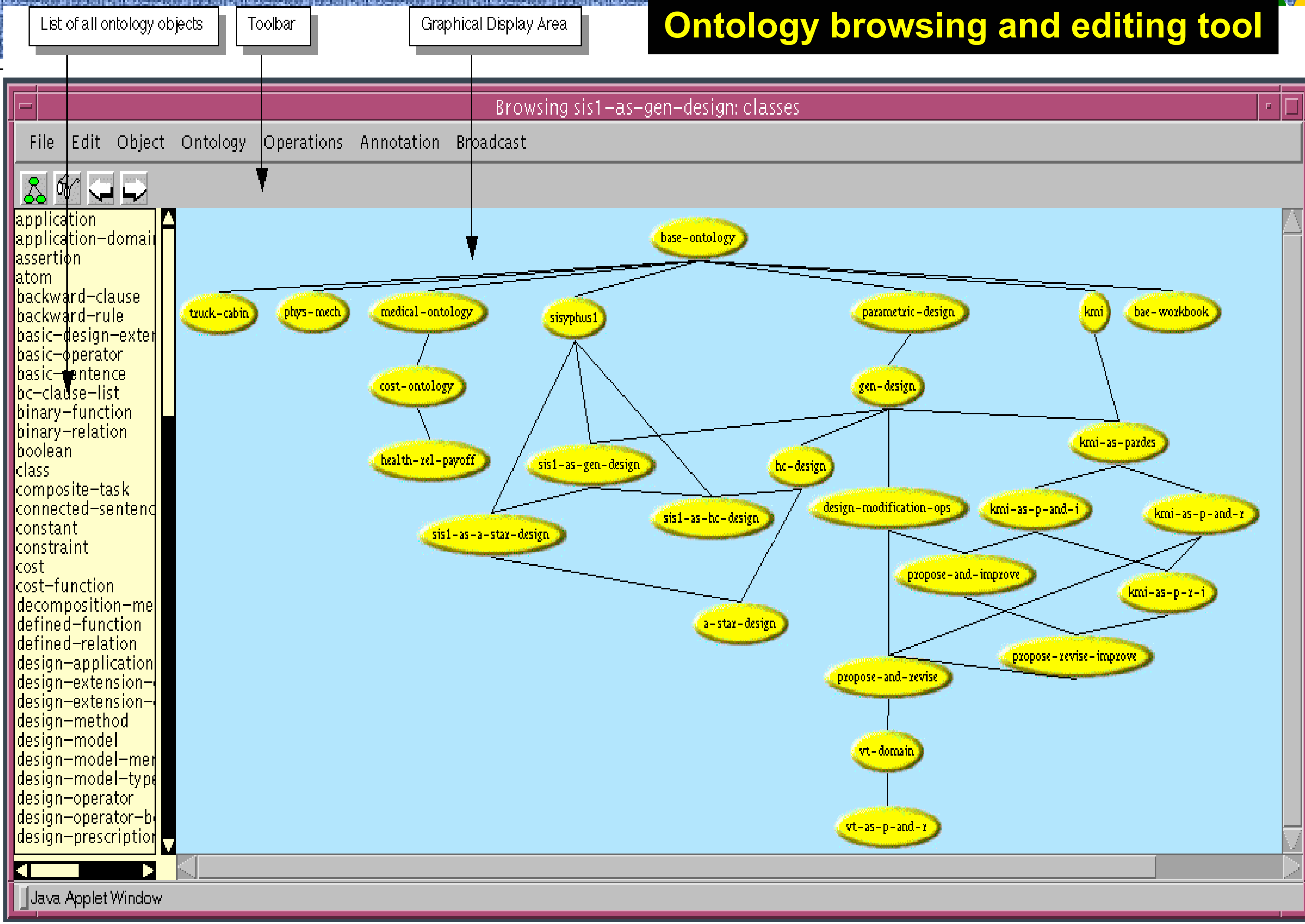
GKB-Editor
(Generic Knowledge Base Editor)



http://www.ai.sri.com/~gkb/

WebOnto Project

Ontology browsing and editing tool



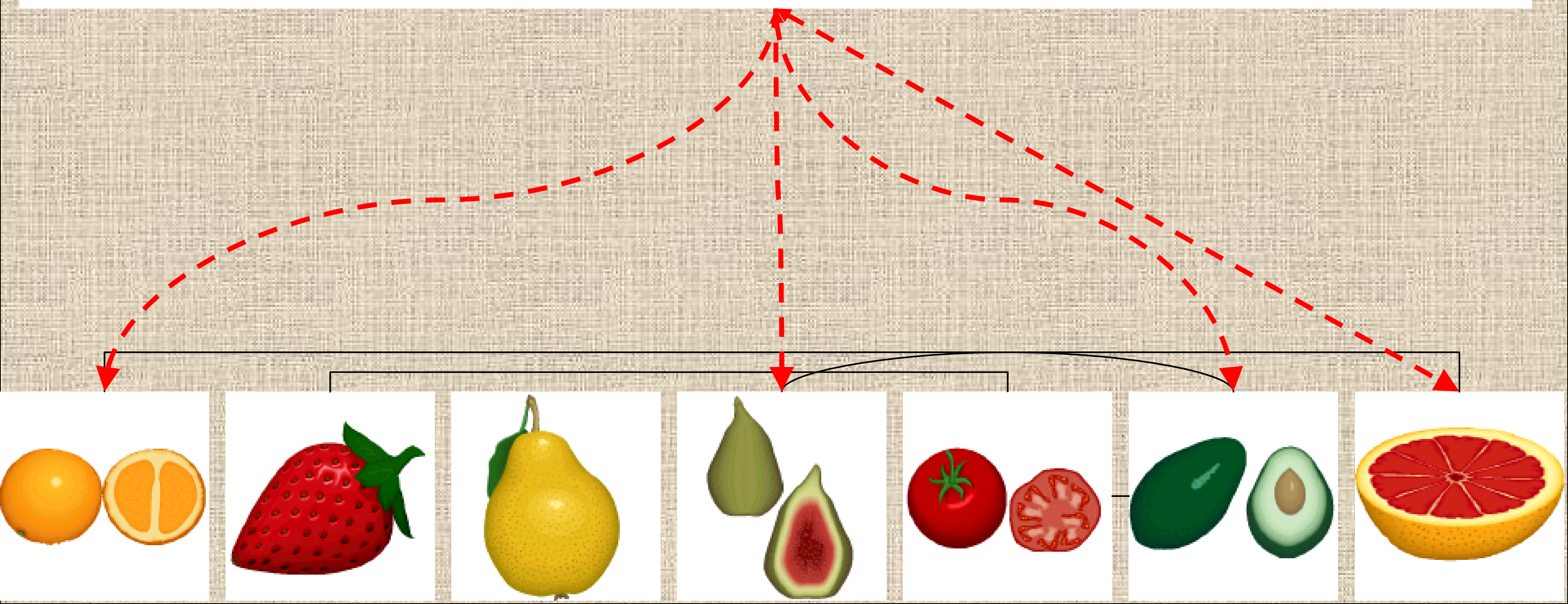
Semantic Web Processes

Questions?

NEXT: METEOR-S Project @ LSDIS lab

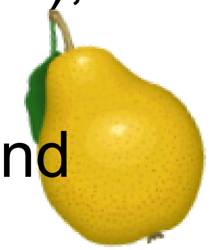
Systems and Applications

METEOR-S Project @ LSDIS lab



Semantics in METEOR-S

- Annotation, Discovery, Composition (in development), and QoS
- Focuses on two issues: **semantic Web services** and **process composition**.
- Process Composition:
  - Functional perspective
    - Web Service Discovery, handling semantic heterogeneity
  - Operational perspective
    - QoS specification for Web Services and Processes.



METEOR-S Project @ LSDIS lab

- METEOR-S exploits Workflow, Semantic Web, Web Services, and Simulation technologies to meet these challenges in a practical and standards based approach.
  - Applying Semantics in Annotation, Quality of Service, Discovery, Composition, Execution of Web Services
  - Adding semantics to different layers of Web services conceptual stack
  - Use of ontologies to provide underpinning for information sharing and semantic interoperability

http://swp.semanticweb.org, http://lsdis.cs.uga.edu/proj/meteor/swp.htm

METEOR-S components for Semantic Web Services

- **Discovery Infrastructure** (MWSDI)
  - Semantic Annotation and Discovery of Web Services [1]
  - Semantic Peer-to-Peer network of Web Services Registries [2]
- **Composer**
  - SCET: Service Composition and Execution Tool [3]
  - **Semantics Process Template Builder and Process Generator** [4]
  - QoS Management
    - Specify, compute, monitor and control QoS (SWR algorithm) [5]
- **Orchestrator** (Under development)
  - Analysis and Simulation [6]
  - Execution
  - Monitoring [6]

[1] [Sivashanmugam et al.- 1], [2] [Verma et al.], [3] [Chandrasekaran et al.], [4] [Sivashanmugam et al.- 2], [5] [Cardoso et al.], [6] [Silver et al.]

METEOR-S Web Service Annotation Framework (MWSAF)

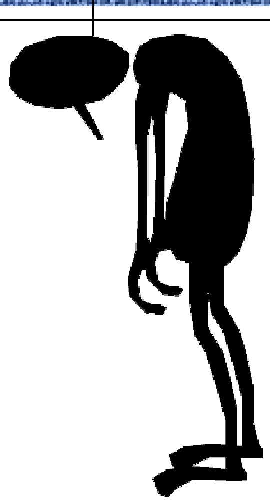
-annotates web services with semantics

Meteor-S Web Service Annotation Framework

METEOR-S Web service Annotation

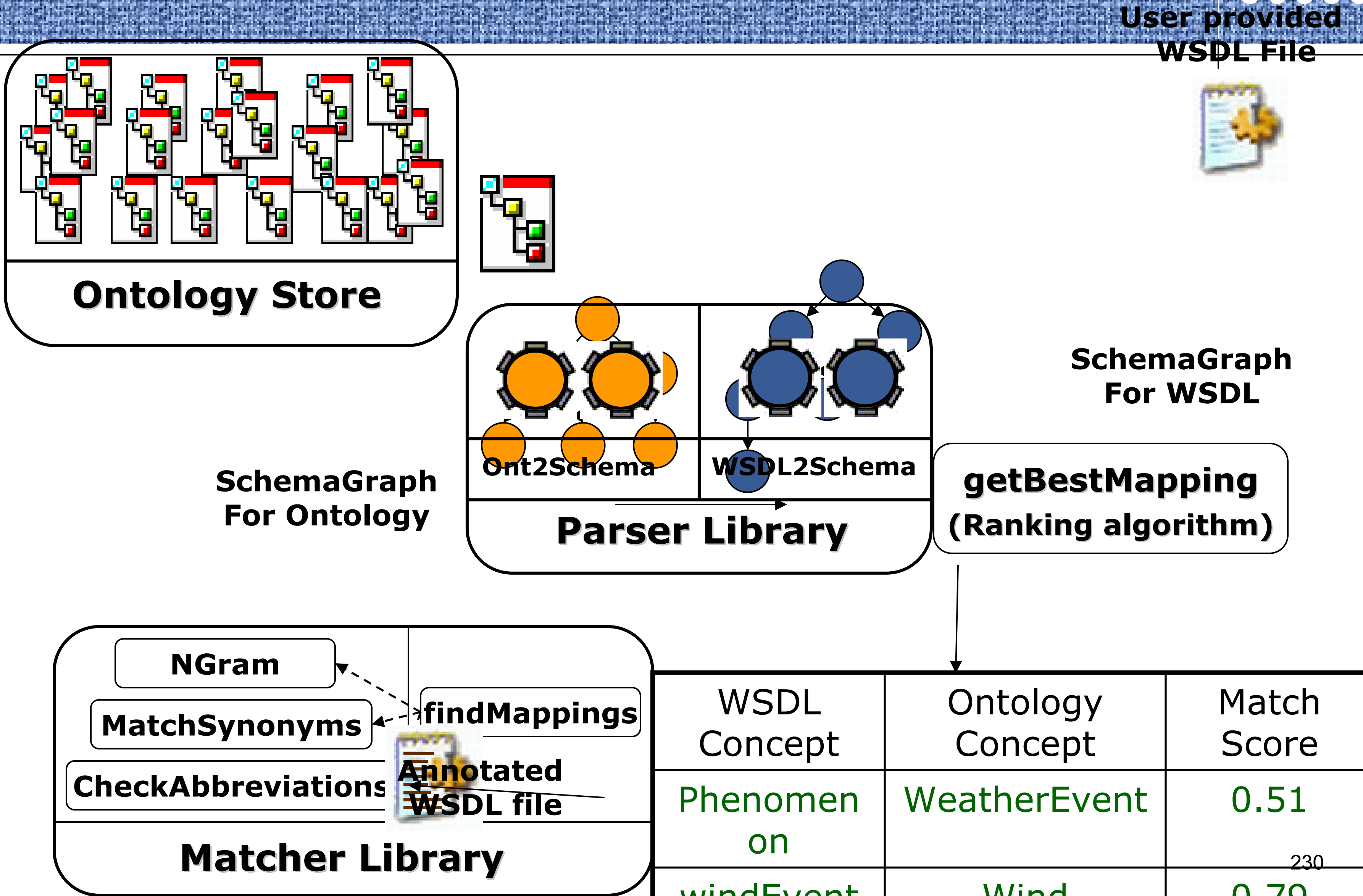
- Map Web service's input/output data as well as functional description using relevant data and function/operation ontologies, respectively
  - Annotate WSDL with Ontologies
- How ?
  - Borrow from Schema matching
  - Semantic disambiguation between terms in XML messages represented in WSDL and concepts in ontology
- Match concepts from WSDL schema to ontological concepts
  - Problems
  - Solution – MWSAF

Why Matching is Difficult ?
(General)



- Aims to identify same real-world entity
  - using names, structures, types, data values, etc
- Schemas represent same entity differently
  - different names => same entity
    - area & address => location
  - same names => different entities
    - area => location or square-feet
- Schema & data never fully capture semantics completely
  - Semantics not documented in sufficient details
  - Schemas not adequately expressive to capture semantics
- Intended semantics is typically subjective
  - IBM Almaden Lab = IBM?
- Complete Automation not possible

MWSAF – Architecture



WSDL Concept	Ontology Concept	Match Score
Phenomenon	WeatherEvent	0.51
windEvent	Wind	0.79

MWSAF – Matching two concepts

- **IOParametersMatch (w,o) =**

  ElemMatch (w,o) + SchemaMatch (w,o) + ContextMatch (w,o)

- **ElemMatch (w,o) => Element level match**
- **SchemaMatch (w,o) => Schema level match**

  subTree(w) == subTree(o)

FUNCTION	findMapping
INPUT	$wc_i \in W$, $oc_i \in O$
OUTPUT	$m_i = (wc_i, oc_i, MS)$

MWSAF – Element level Match

Definition

- Element level match is the measure of the linguistic similarity between two concepts based on their names.
- Assumption – Concepts from XML schema and ontology have meaningful names

ElemMatch (w,o) => Element level match

- NameMatch with stemming
- Description Match (future work)
- SynonymsMatch : Snow and snowFall mean the same
- HypernymRelation (w is a kind of o) : prevailing_speed is a type of speed of a wind i.e. windSpeed
- HyponymRelation (o is a kind of w)
- Acronyms : Sea Level Pressure has acronym SLP

MWSAF – Schema level Match

Definition

- The Schema level match is the measure of structural similarity between two concepts
- It is based on sub-concept similarity (subConceptSim) and sub-concept match (subConceptMatch).

$$\text{SchemaMatch} = \sqrt{\text{subConceptSim} * \text{subConceptMatch}}$$

$$\text{where, subConceptSim} \in [0,1] \quad \text{subConceptMatch} \in [0,1]$$

METEOR-S Web Service Discovery Infrastructure (MWSDI)

- uses Functional, Data and QoS semantics

Service Discovery

METEOR-S Web Service Discovery Infrastructure (MWSDI)

Service Selection

- uses Functional, Data and QoS semantics

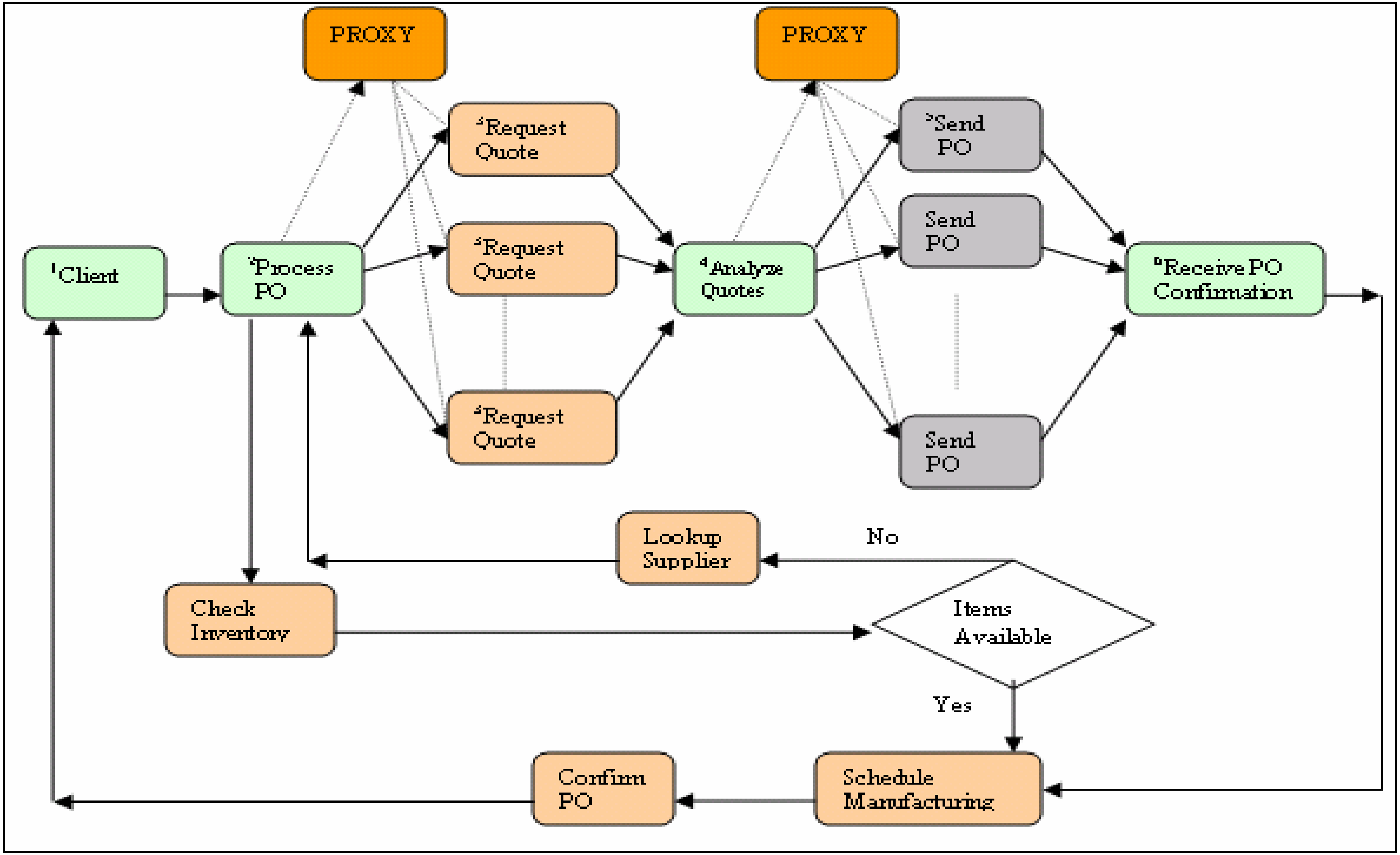
METEOR-S Web Service Composition Framework (MWSCF)

- needed for the world where business processes never stop changing

Scenario

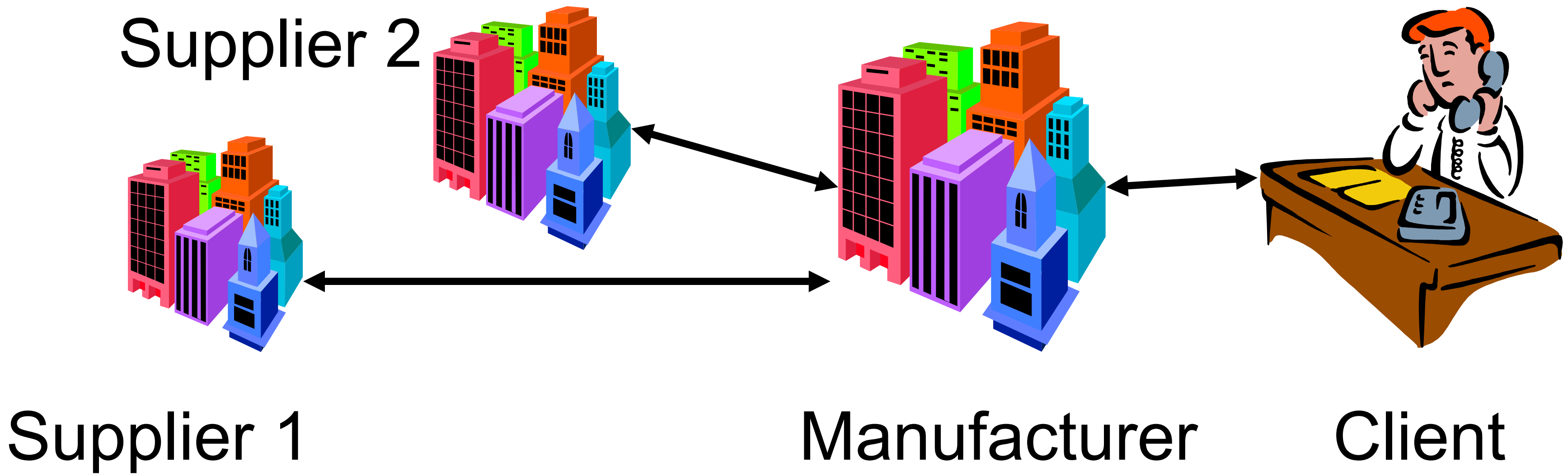
- Client Application e.g. JSP
- Process Client's Purchase Order (PO)
  - Discover Suppliers
  - Request Quote
- Analyze Quotes
  - Optimize on QoS
  - Inter Service Dependencies
  - Send PO to supplier(s)
- Receive PO Confirmation from Supplier(s)
- Confirm PO to Client

Supply Chain – QoS Based



Predefined flows

- Static binding (supported by BPEL4WS)
- Choose service at design time

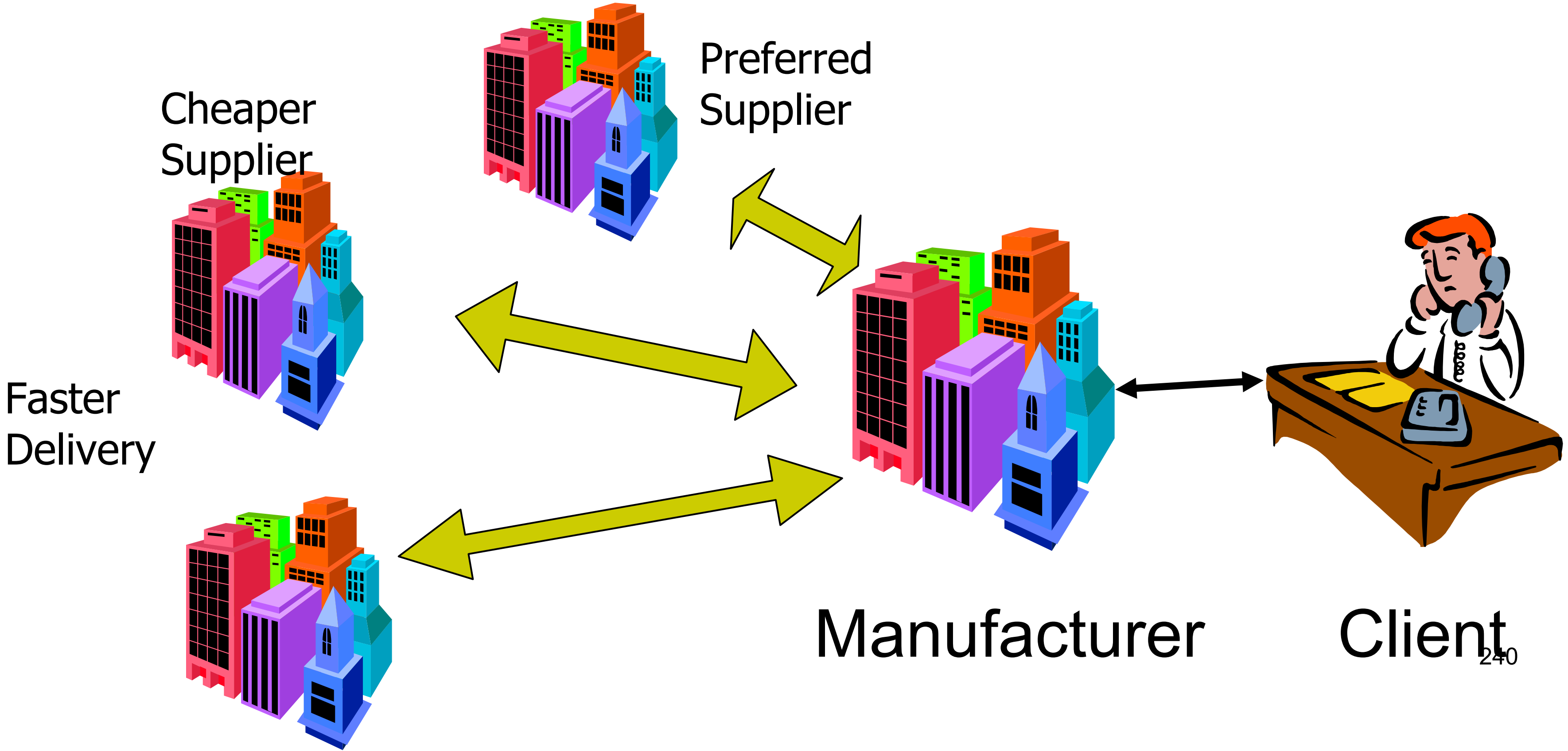


Manufacturer tightly coupled with suppliers

Predefined flows

- Dynamic binding
- Choose new services at runtime

Dynamically choose best supplier at runtime



MWSCF Architecture

Process Execution

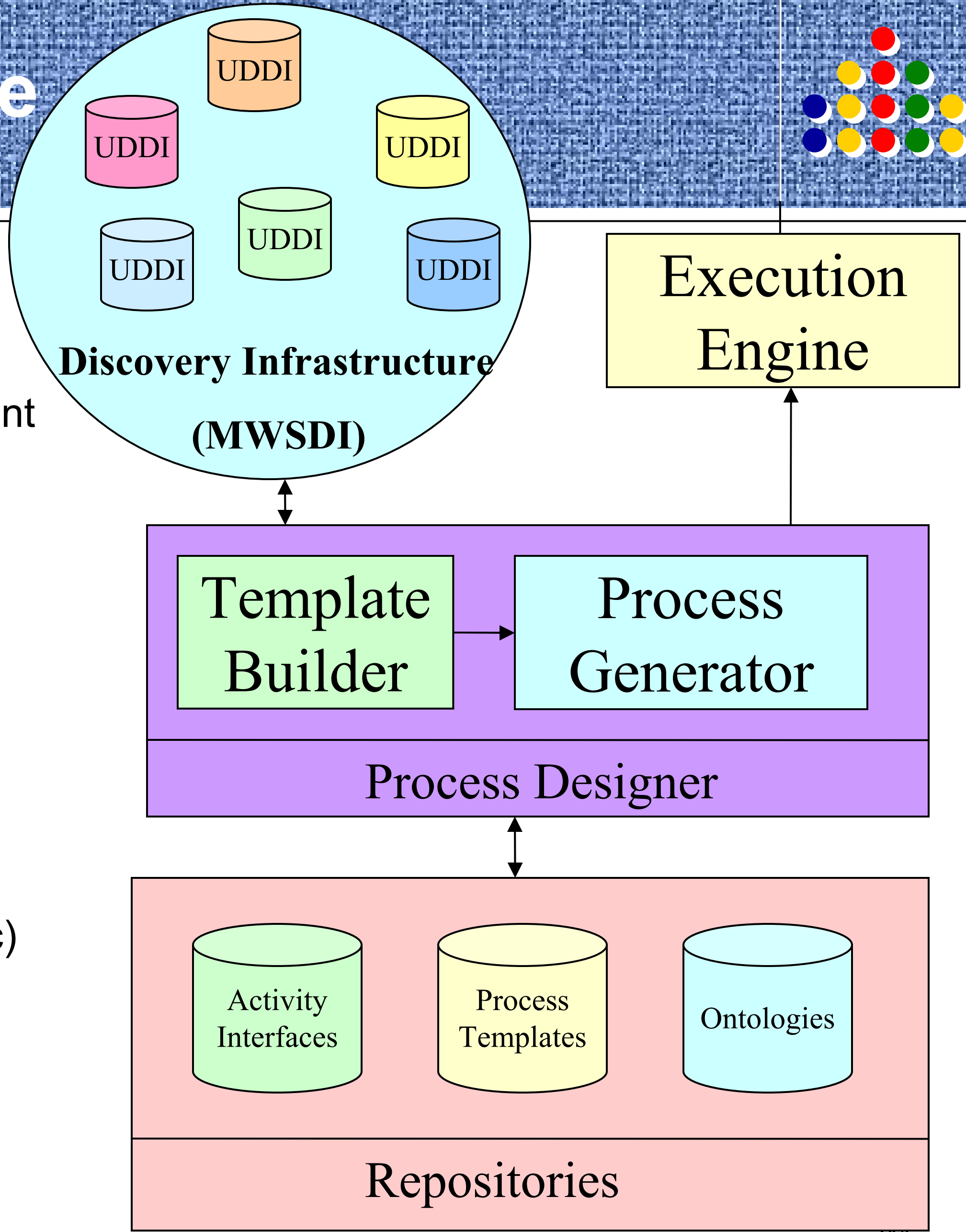
1. Validation and deployment
2. Executing the process using a client

Process Designer

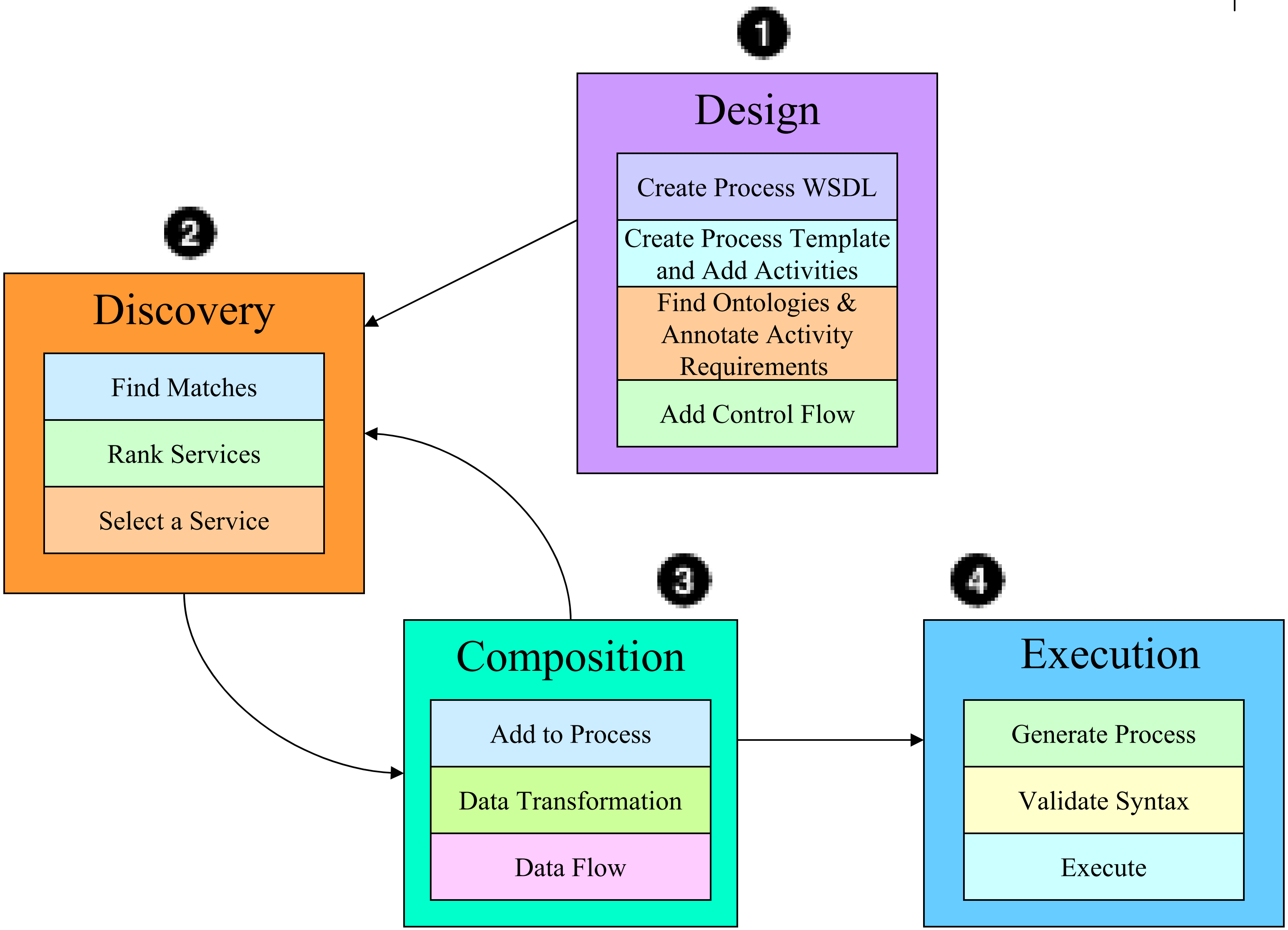
1. Template Construction
   activity specification using
   - interfaces
   - services
   - semantic activity templates
   - other details
2. Process Generation
   - Service discovery (automatic) and selection (semi-automatic)
   - Data flow

Repositories are used to store

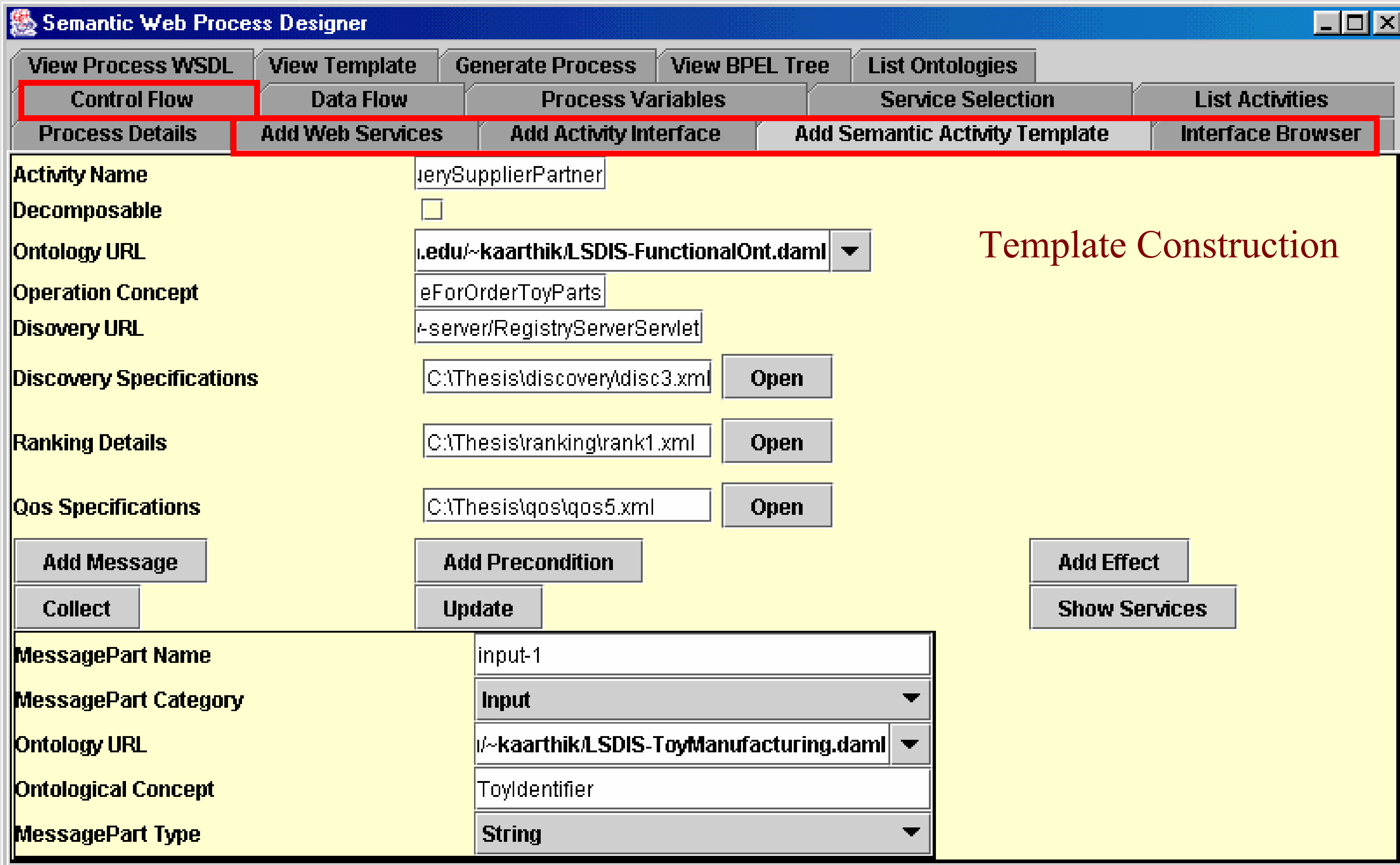
1. Web Service Interfaces
2. Ontologies
3. Process Templates



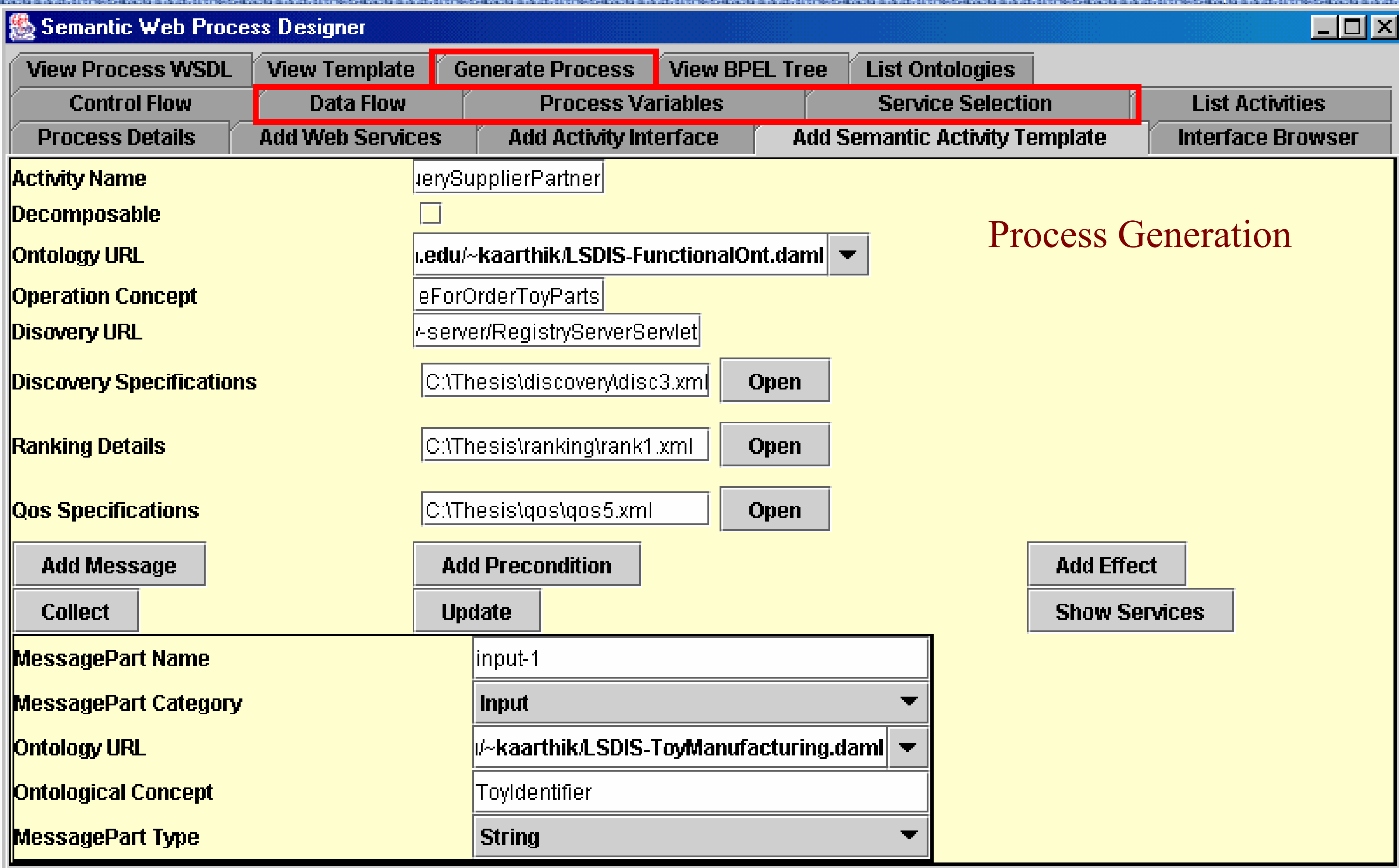
Web Process Life-Cycle



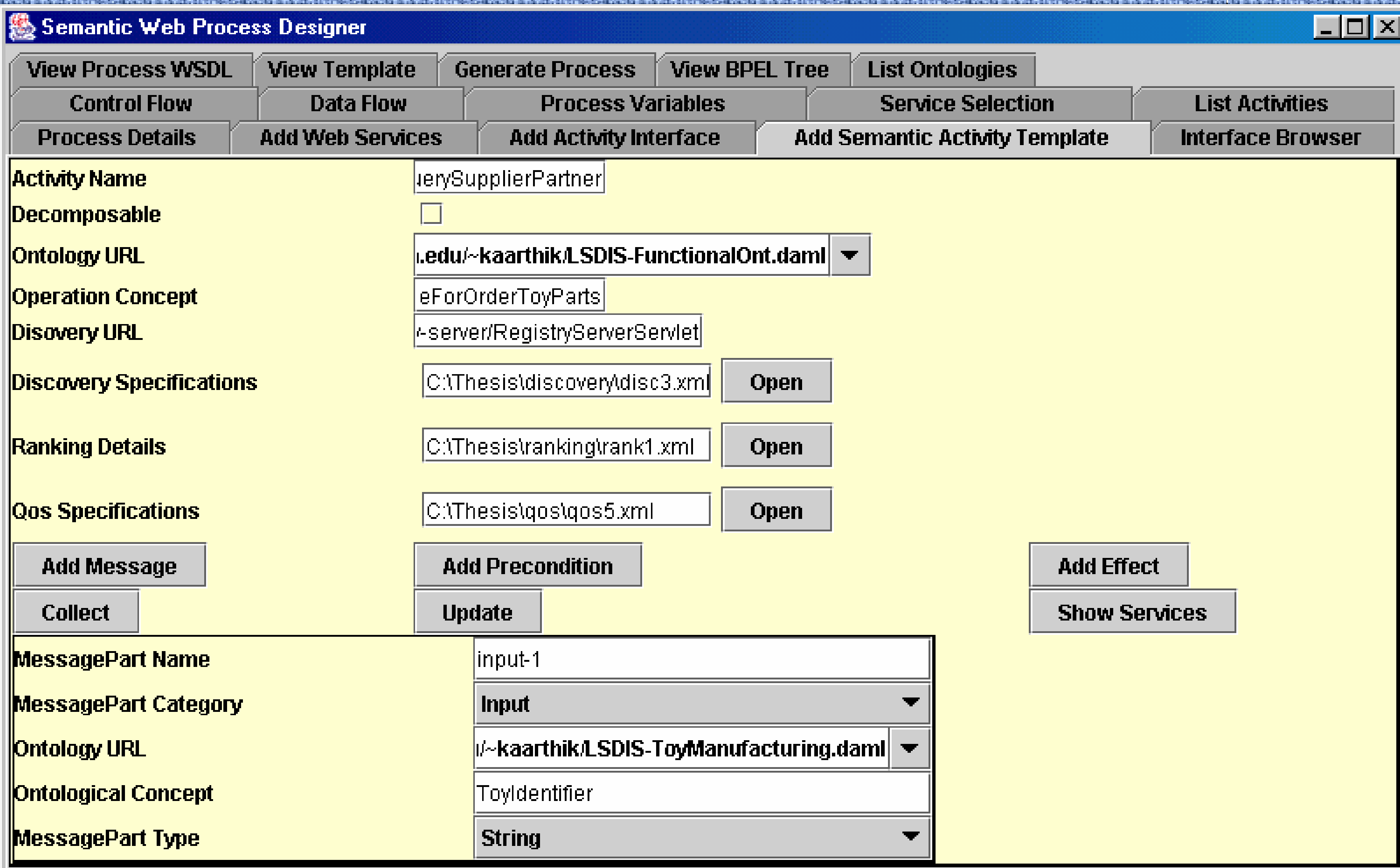
Semantic Web Process Design



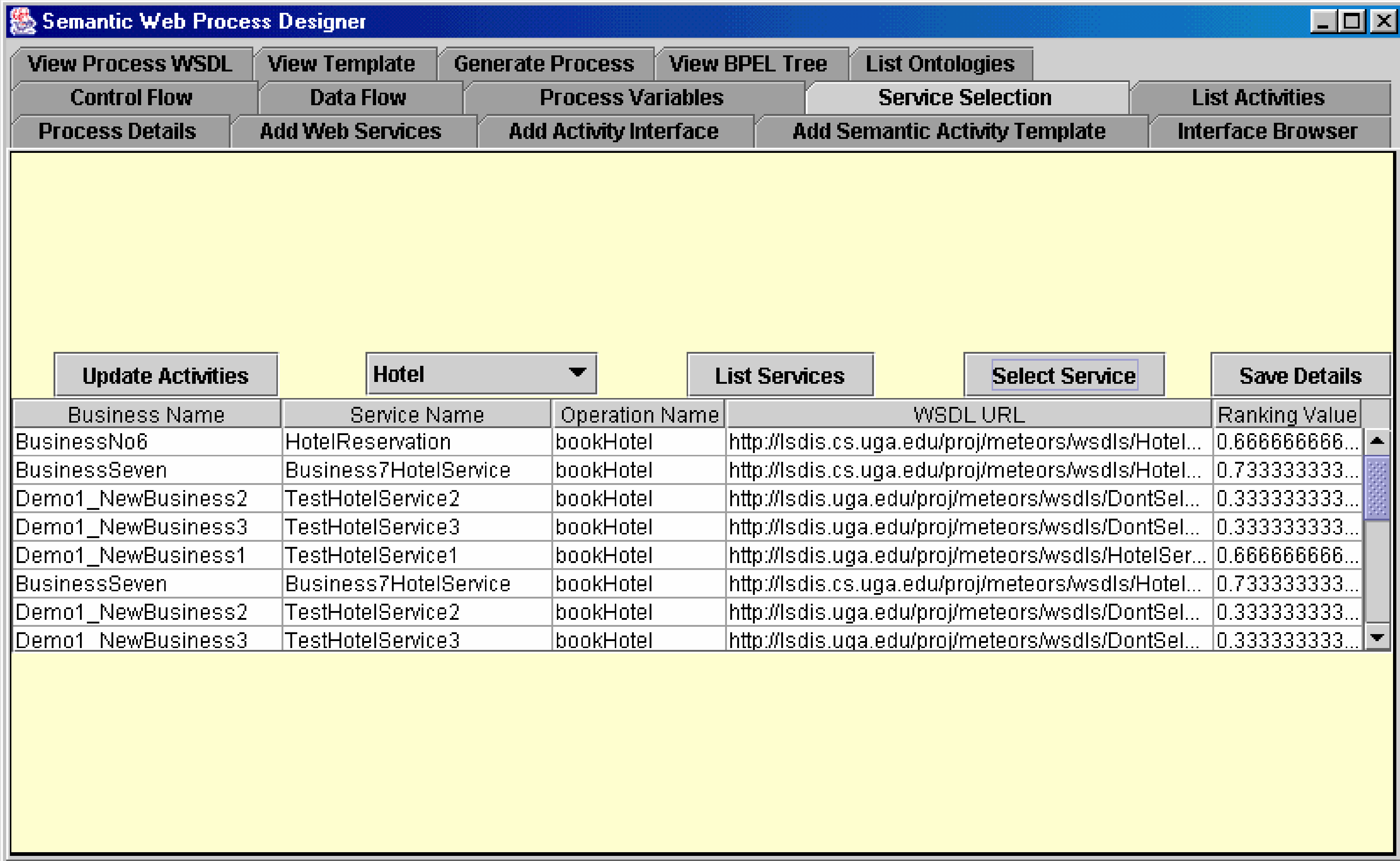
Semantic Web Process Design



Semantic Web Process Design

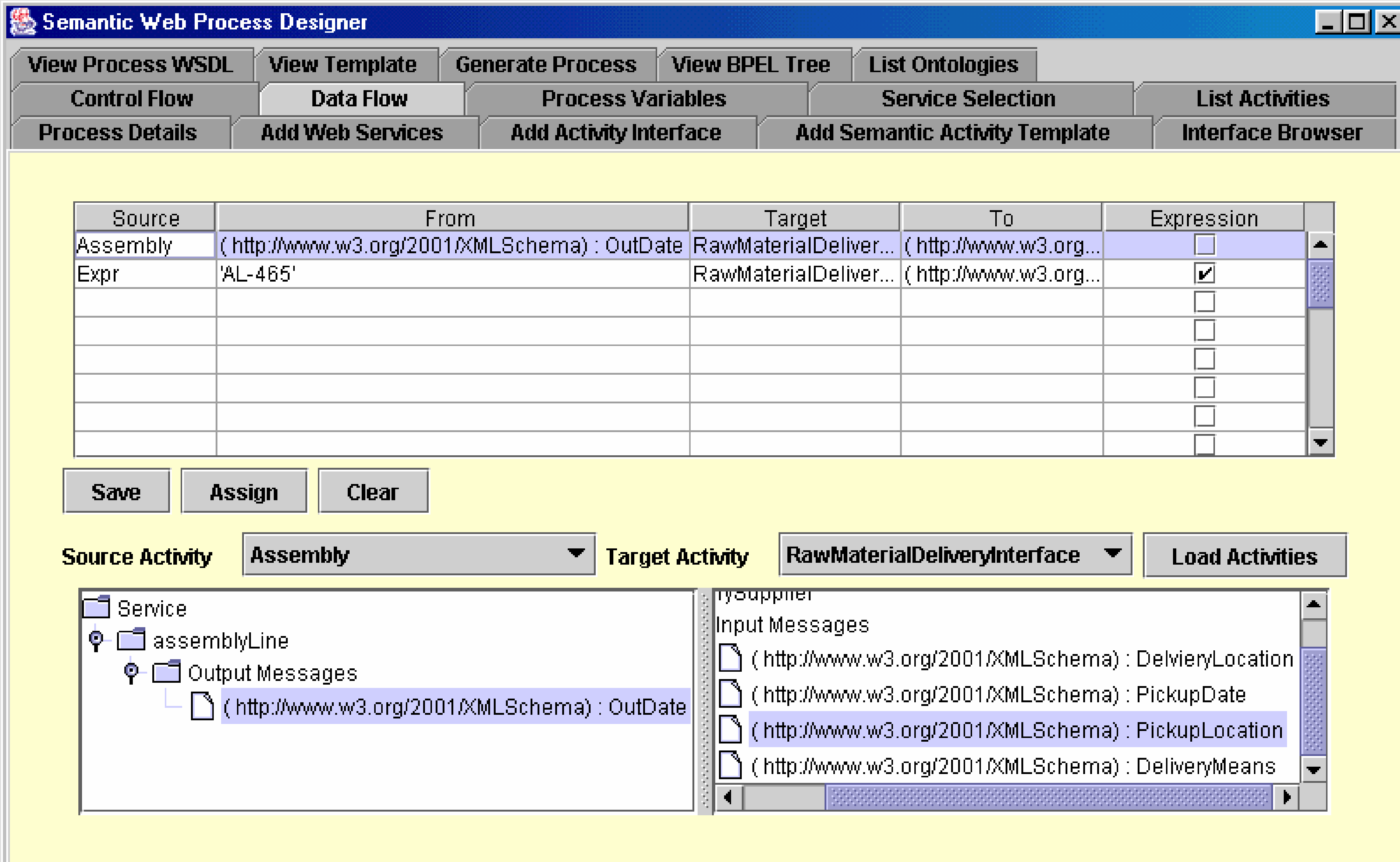


Semantic Web Process Design

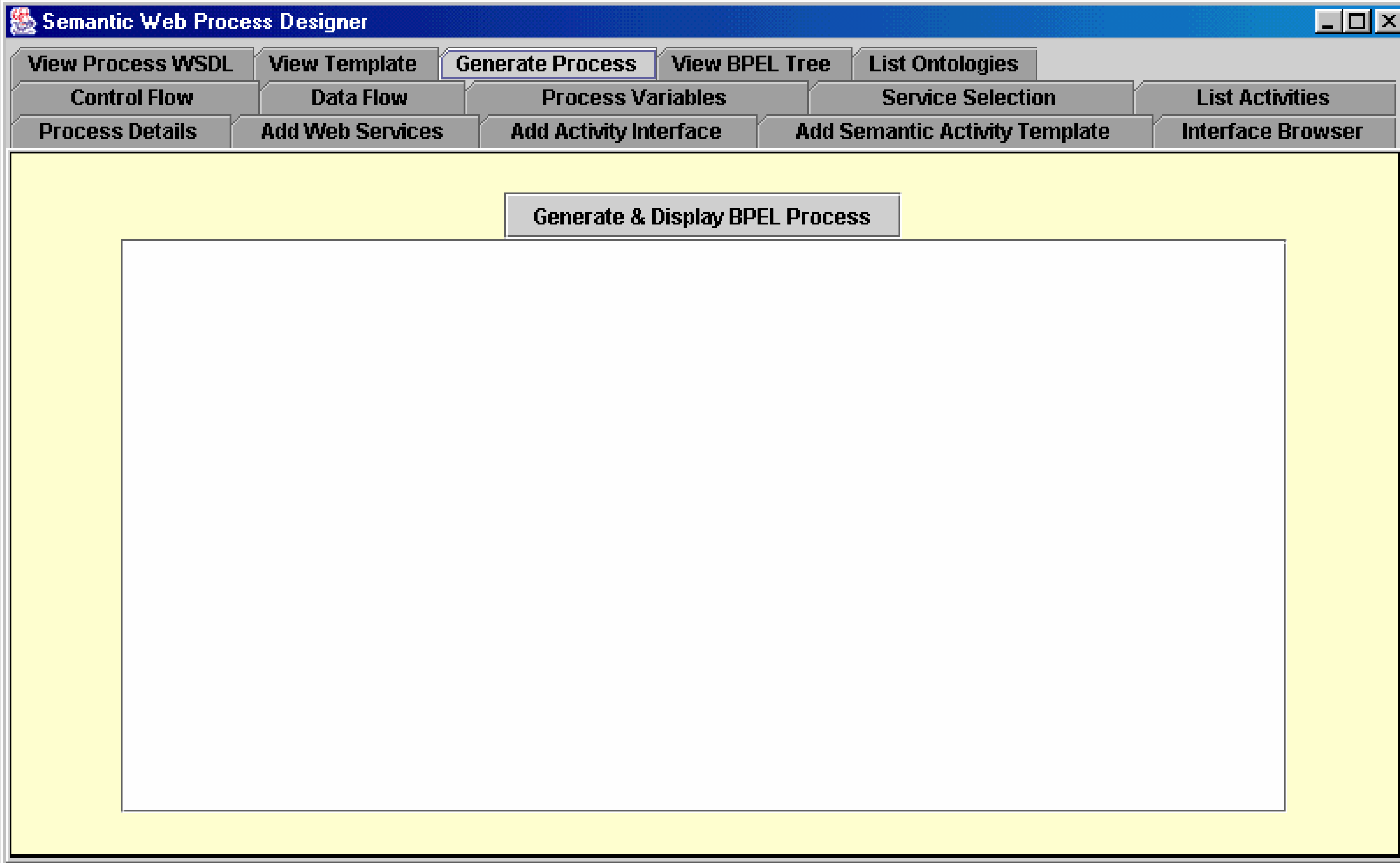


Business Name	Service Name	Operation Name	WSDL URL	Ranking Value
BusinessNo6	HotelReservation	bookHotel	http://lsdis.cs.uga.edu/proj/meteors/wsdls/Hotel...	0.666666666...
BusinessSeven	Business7HotelService	bookHotel	http://lsdis.cs.uga.edu/proj/meteors/wsdls/Hotel...	0.733333333...
Demo1_NewBusiness2	TestHotelService2	bookHotel	http://lsdis.uga.edu/proj/meteors/wsdls/DontSel...	0.333333333...
Demo1_NewBusiness3	TestHotelService3	bookHotel	http://lsdis.uga.edu/proj/meteors/wsdls/DontSel...	0.333333333...
Demo1_NewBusiness1	TestHotelService1	bookHotel	http://lsdis.uga.edu/proj/meteors/wsdls/HotelSer...	0.666666666...
BusinessSeven	Business7HotelService	bookHotel	http://lsdis.cs.uga.edu/proj/meteors/wsdls/Hotel...	0.733333333...
Demo1_NewBusiness2	TestHotelService2	bookHotel	http://lsdis.uga.edu/proj/meteors/wsdls/DontSel...	0.333333333...
Demo1_NewBusiness3	TestHotelService3	bookHotel	http://lsdis.uga.edu/proj/meteors/wsdls/DontSel...	0.333333333...

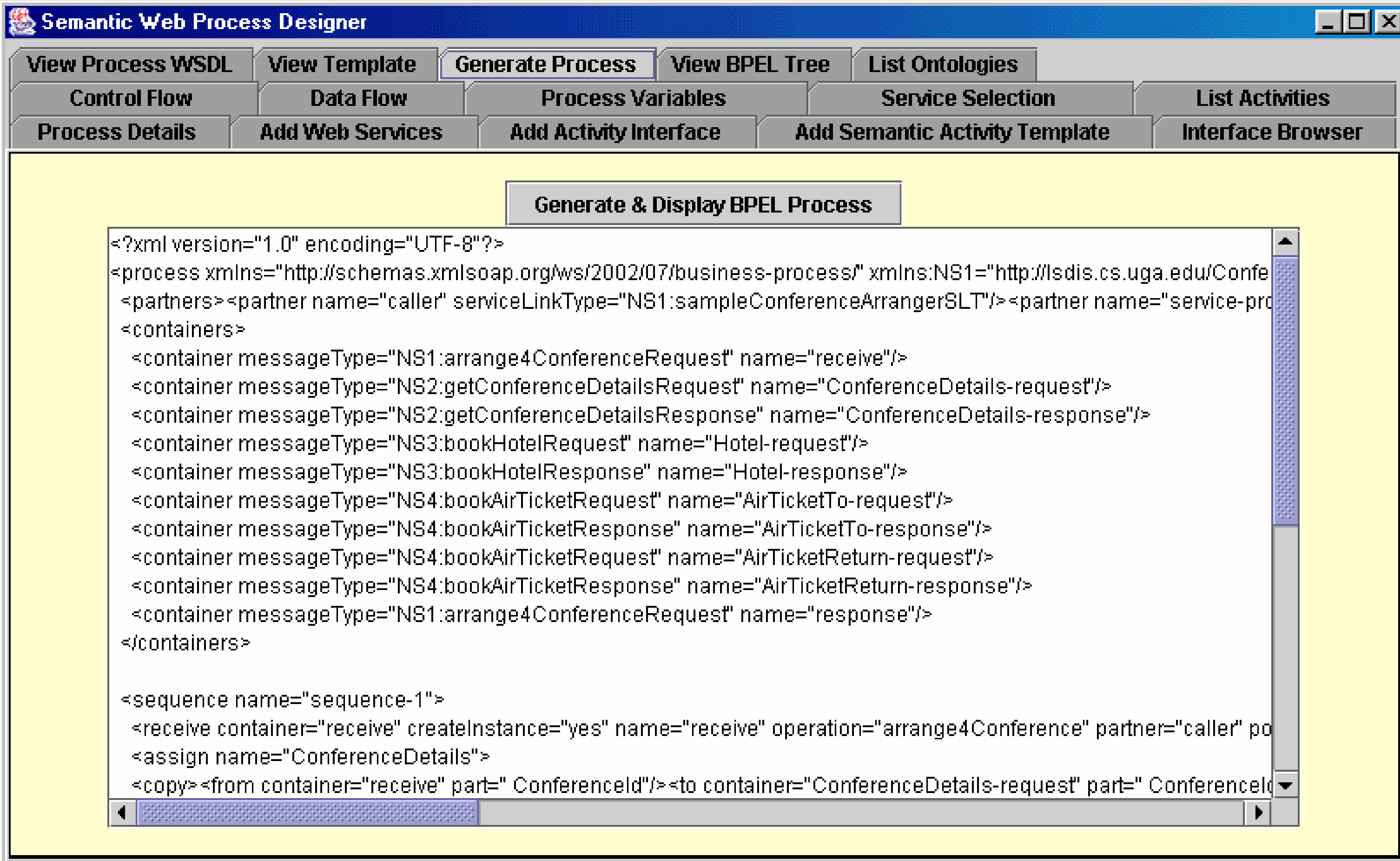
Semantic Web Process Design



Semantic Web Process Design



Semantic Web Process Design

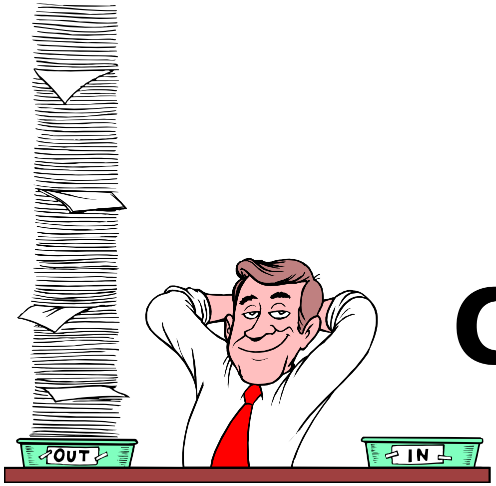


Ongoing Projects

- **SWSI**
  - **SWSA** Semantic Web Services Architecture
  - **SWSL** Semantic Web Services Language
- **WonderWeb:** http://wonderweb.man.ac.uk/
  - Development of a framework of techniques and methodologies that provide an engineering approach to the building and use of ontologies.
  - Development of a set of foundational ontologies covering a wide range of application domains.
  - Development of infrastructures and tool support that will be required by real world applications in the Semantic Web.

Ongoing Projects

- **OWL-S:** http://www.daml.org/services/
  - Set of ontologies to describe functionalties of web services
- **OWL-S Matchmaker:** http://www-2.cs.cmu.edu/%7Esoftagents/daml_Mmaker/OWL-S_matchmaker.htm
  - Match service requestors with service providers
  - Semantic Matchmaking for Web Services Discovery
- **Web Service Composer:** http://www.mindswap.org/~evren/composer/
  - Semi-automatic process for the dynamic composition of web services
- **Web Services:** http://www-106.ibm.com/developerworks/webservices/
  - WSDL, UDDI, SOAP
  - Business Process with BPEL4WS



Conclusions

Conclusions

- Semantic Web service Annotation and Discovery
  - Data semantics
  - Functional semantics
  - QoS Semantics
- Web processes vs. Semantic Web processes
  - OWL-S (OWL-S)
- Web process composition
  - Web services semantic degree of integration
  - Data, Functional, and QoS similarity
- Web process QoS computation
  - QoS Models, techniques, and algorithms

Conclusions

- **Present Problems in Process Composition**
  - Static discovery of Web Services
  - Design/deployment-time binding of Web services
  - Process Composition is based on interfaces of participating services
- **Proposition**
  - Semantics is the enabler to address the problems of scalability, heterogeneity (syntactic and semantic), machine understandability faced by Web services
- **Semantics for Web Services**
  - Semantics can be applied to different layers of Web Services conceptual stack
  - Semantics for Web Services can be categorized into at least 4 different dimensions namely Data, Functional, Execution and Quality (QoS).

Conclusions

- Semantics can help address big challenges related to scalability, dynamic environments.
- But comprehensive approach to semantics will be needed:
  - Data/information, function/operation, execution, QoS
- Semantic (Web) principles and technology bring new tools and capabilities that we did not have in EAI, workflow management of the past

More at: http://lsdis.cs.uga.edu/proj/meteor/SWP.htm

Semantic Web Processes

Questions?

References

Extensive related work at: IBM, Karlsruhe, U. Manchester, OWL-S (CMU, Stanford, UMD)

- Resources: http://lsdis.cs.uga.edu/lib/presentations/SWSP-tutorial-resource.htm
- [Kreger] http://www-3.ibm.com/software/solutions/webservices/pdf/WSCA.pdf
- [Sivashanmugam et al.-1] Adding Semantics to Web Services Standards
- [Sivashanmugam et al.-2] Framework for Semantic Web Process Composition
- [Verma et al.] MWSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services
- [Chandrasekaran et al.] Performance Analysis and Simulation of Composite Web Services
- [Cardoso et al.] Modeling Quality of Service for Workflows and Web Service Processes
- [Silver et al.] Modeling and Simulation of Quality of Service for Composition of Web Services
- [Paolucci et al.] Importing Semantic Web in UDDI
- [UDDI-v3] http://uddi.org/pubs/uddi-v3.00-published-20020719.htm
- http://www.daml.org/services/
- http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/

More at: http://lsdis.cs.uga.edu/SWP.htm

Semantic Web Processes

End