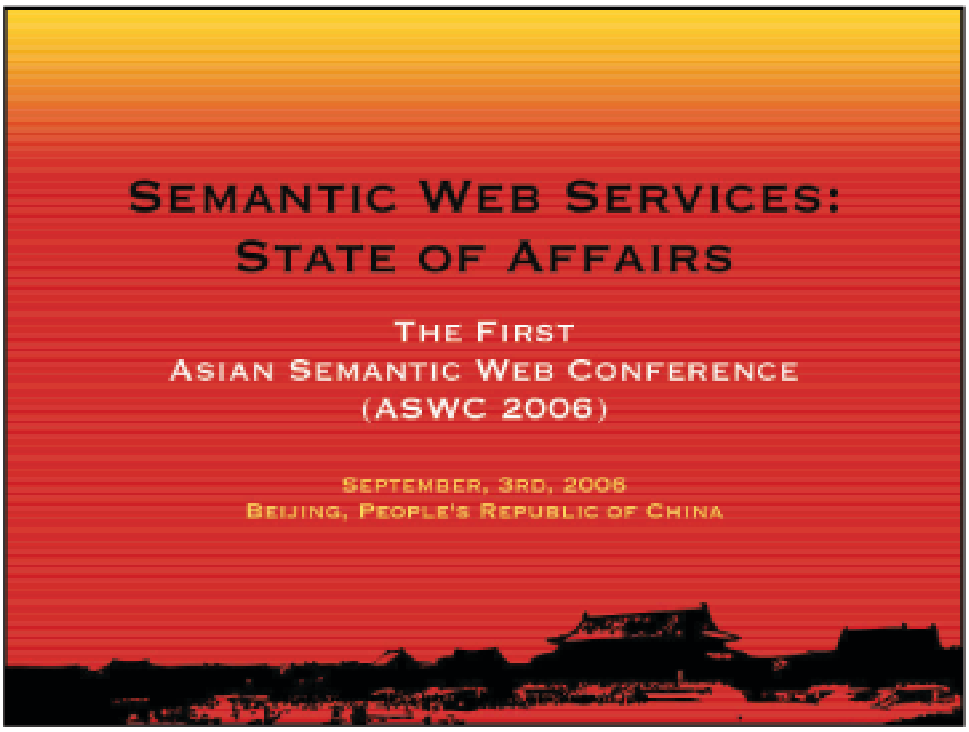


Semantic Web Services: State of Affairs

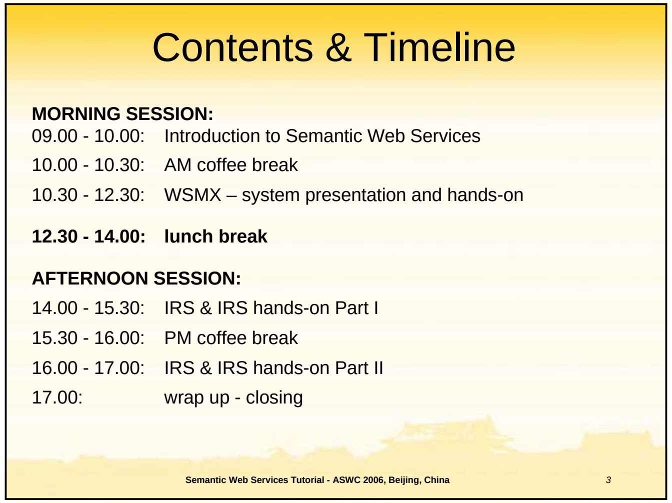
The First
Asian Semantic Web Conference
(ASWC 2006)

September, 3rd, 2006
Beijing, People's Republic of China

IRS
studio
DERI
KMi

Emilia Cimpian
Omair Shafiq
Michael Stollberg
Michal Zaremba
Dumitru Roman

Liliana Cabral
John Domingue

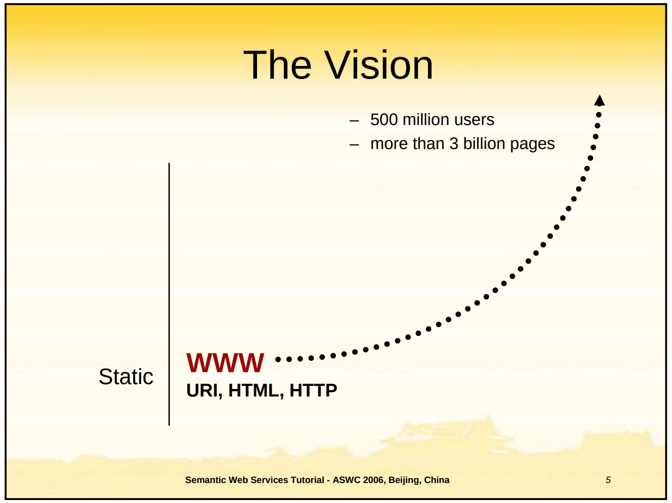


Contents & Timeline

MORNING SESSION:	
09.00 - 10.00:	Introduction to Semantic Web Services
10.00 - 10.30:	AM coffee break
10.30 - 12.30:	WSMX – system presentation and hands-on
12.30 - 14.00:	**lunch break**
AFTERNOON SESSION:	
14.00 - 15.30:	IRS & IRS hands-on Part I
15.30 - 16.00:	PM coffee break
16.00 - 17.00:	IRS & IRS hands-on Part II
17.00:	wrap up - closing

INTRODUCTION
– Semantic Web Services –

Michael Stollberg

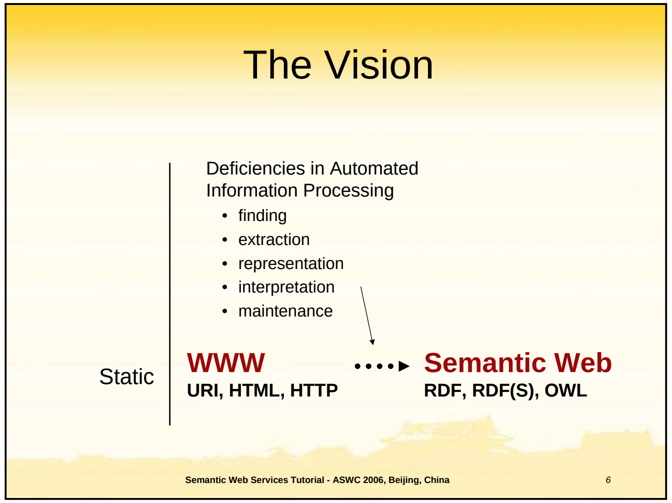


The Vision

– 500 million users
– more than 3 billion pages

Static

WWW
URI, HTML, HTTP



The Vision

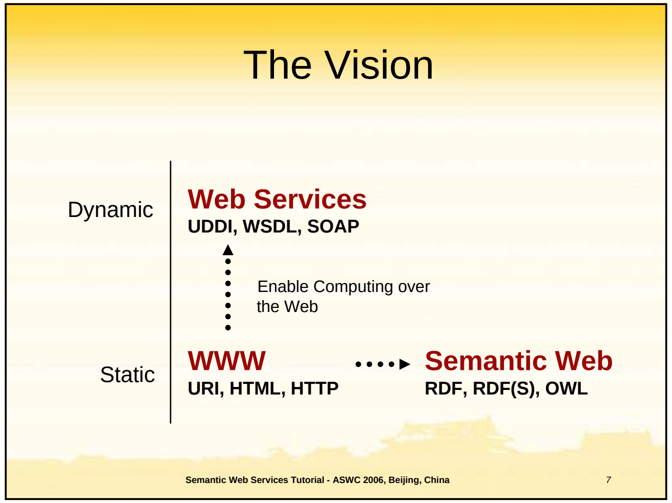
Deficiencies in Automated Information Processing

- finding
- extraction
- representation
- interpretation
- maintenance

Static

WWW
URI, HTML, HTTP

Semantic Web
RDF, RDF(S), OWL



The Vision

Dynamic

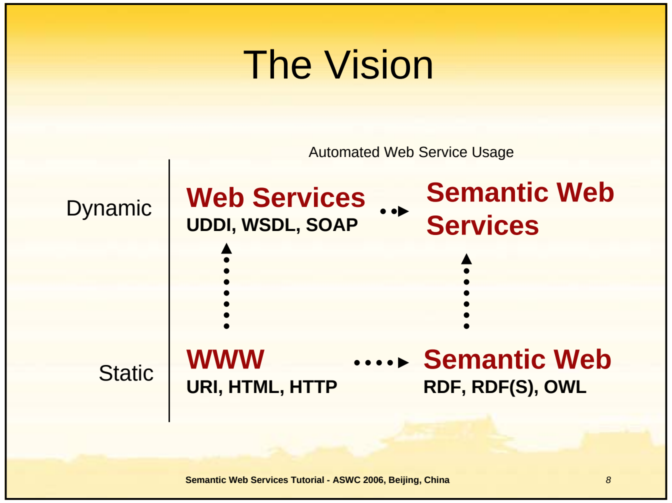
Web Services
UDDI, WSDL, SOAP

Enable Computing over the Web

Static

WWW
URI, HTML, HTTP

Semantic Web
RDF, RDF(S), OWL



The Vision

Automated Web Service Usage

Dynamic

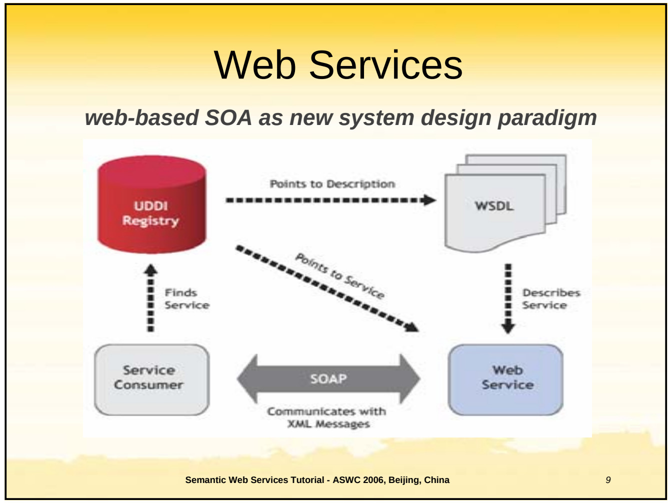
Web Services
UDDI, WSDL, SOAP

Semantic Web Services

Static

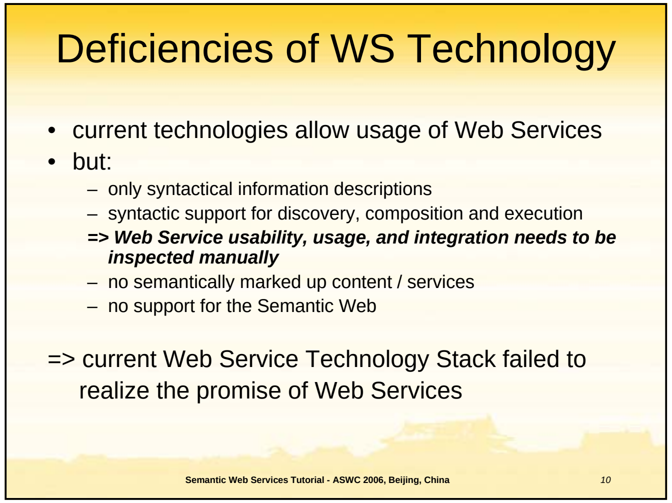
WWW
URI, HTML, HTTP

Semantic Web
RDF, RDF(S), OWL



Web Services

web-based SOA as new system design paradigm

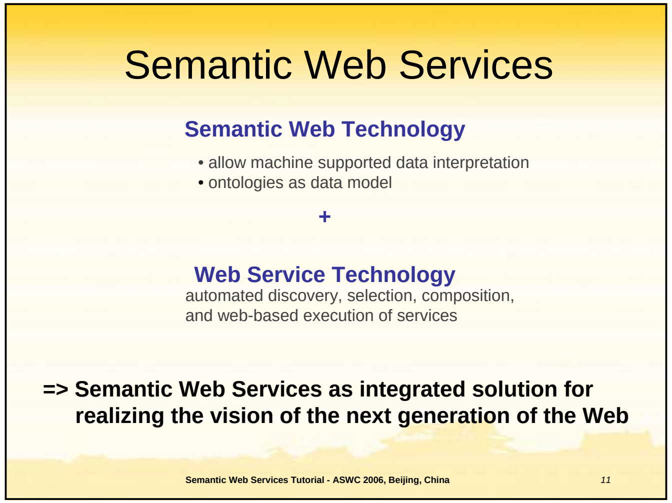


Deficiencies of WS Technology

- current technologies allow usage of Web Services
- but:
  - only syntactical information descriptions
  - syntactic support for discovery, composition and execution

  => *Web Service usability, usage, and integration needs to be inspected manually*
  - no semantically marked up content / services
  - no support for the Semantic Web

=> current Web Service Technology Stack failed to realize the promise of Web Services



Semantic Web Services

Semantic Web Technology

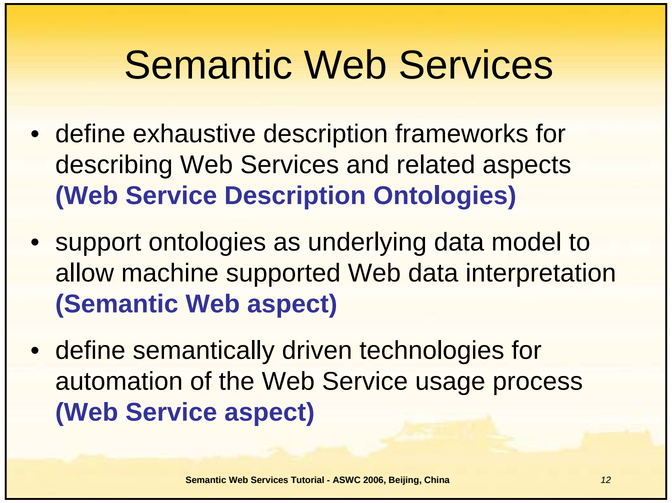
- allow machine supported data interpretation
- ontologies as data model

+

Web Service Technology

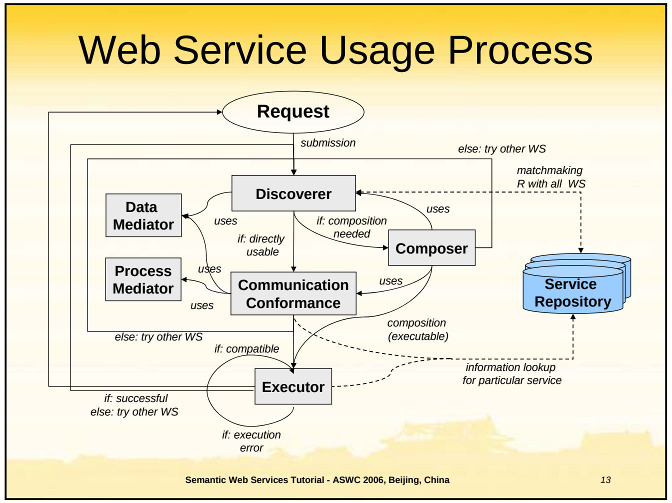
automated discovery, selection, composition, and web-based execution of services

=> Semantic Web Services as integrated solution for realizing the vision of the next generation of the Web



Semantic Web Services

- define exhaustive description frameworks for describing Web Services and related aspects **(Web Service Description Ontologies)**
- support ontologies as underlying data model to allow machine supported Web data interpretation **(Semantic Web aspect)**
- define semantically driven technologies for automation of the Web Service usage process **(Web Service aspect)**

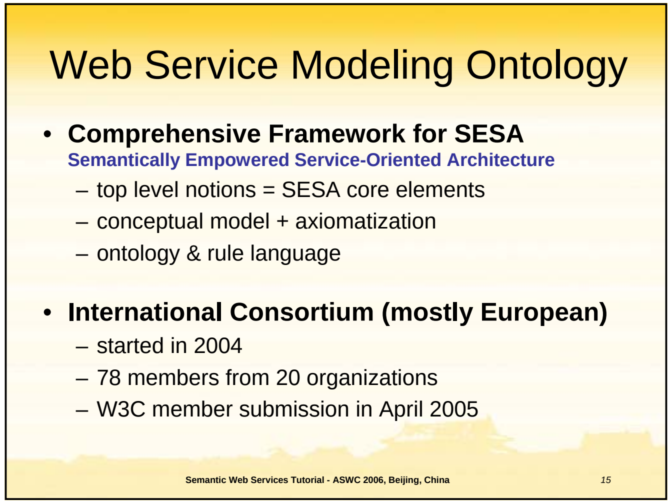


Web Service Usage Process

The Web Service Modelling Ontology – WSMO –

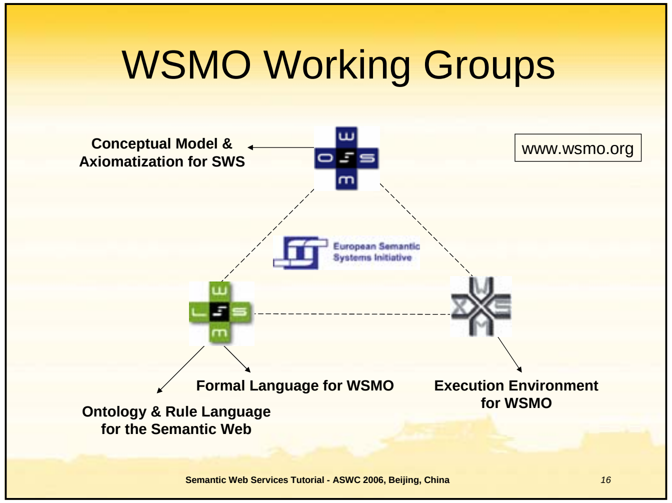
Michael Stollberg

Dumitru Roman

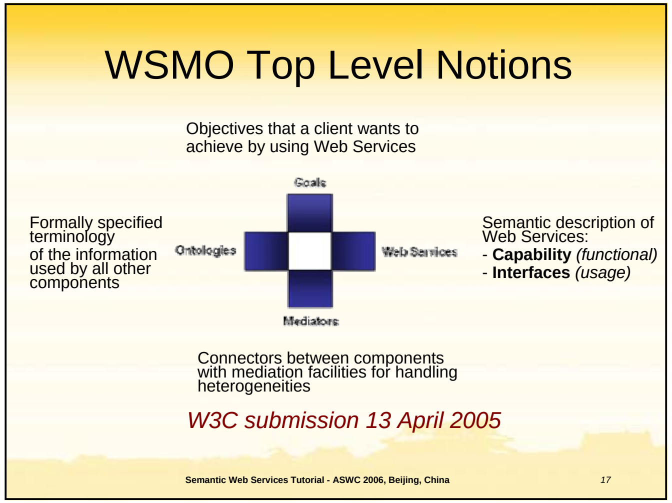


Web Service Modeling Ontology

- **Comprehensive Framework for SESA**
  Semantically Empowered Service-Oriented Architecture
  - top level notions = SESA core elements
  - conceptual model + axiomatization
  - ontology & rule language
- **International Consortium (mostly European)**
  - started in 2004
  - 78 members from 20 organizations
  - W3C member submission in April 2005



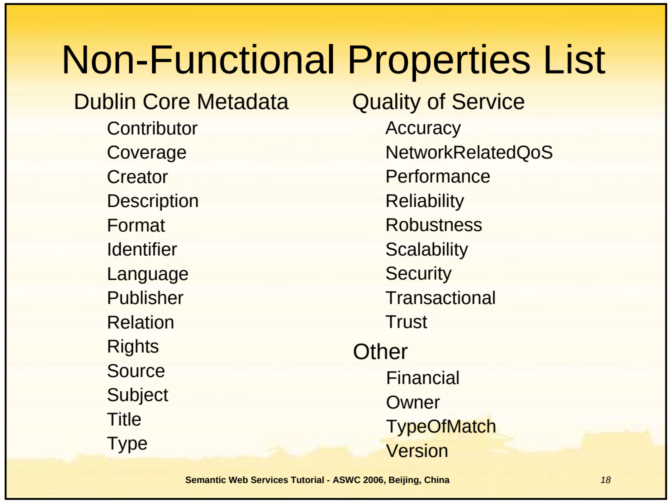
WSMO Working Groups



WSMO Top Level Notions

Objectives that a client wants to
achieve by using Web Services
Goals
Formally specified terminology
of the information used by all other components
Ontologies
Web Services
Semantic description of Web Services:
- Capability (functional)
- Interfaces (usage)
Mediators
Connectors between components with mediation facilities for handling heterogeneities

W3C submission 13 April 2005



Non-Functional Properties List

Dublin Core Metadata

- Contributor
- Coverage
- Creator
- Description
- Format
- Identifier
- Language
- Publisher
- Relation
- Rights
- Source
- Subject
- Title
- Type

Quality of Service

- Accuracy
- NetworkRelatedQoS
- Performance
- Reliability
- Robustness
- Scalability
- Security
- Transactional
- Trust

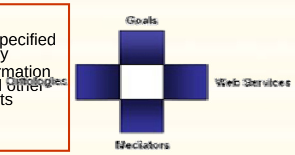
Other

- Financial
- Owner
- TypeOfMatch
- Version

WSMO Ontologies

Objectives that a client wants to achieve by using Web Services

Formally specified terminology of the information used by all other components



Semantic description of Web Services:
- **Capability** *(functional)*
- **Interfaces** *(usage)*

Connectors between components with mediation facilities for handling heterogeneities

Ontology Usage & Principles

- **Ontologies are the ‘data model’ throughout WSMO**
  - all WSMO element descriptions rely on ontologies
  - all data interchanged in Web Service usage are ontologies
  - Semantic information processing & ontology reasoning
- **WSMO Ontology Language WSML**
  - conceptual syntax for describing WSMO elements
  - logical language for axiomatic expressions (WSML Layering)
- **WSMO Ontology Design**
  - Modularization: import / re-using ontologies, modular approach for ontology design
  - De-Coupling: heterogeneity handled by **OO Mediators**

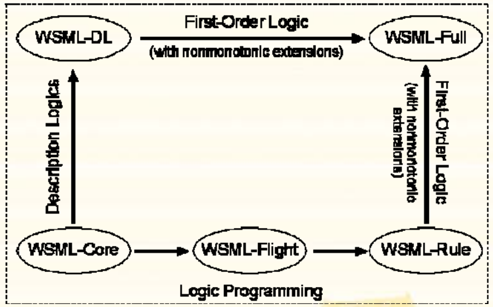
Ontology Specification

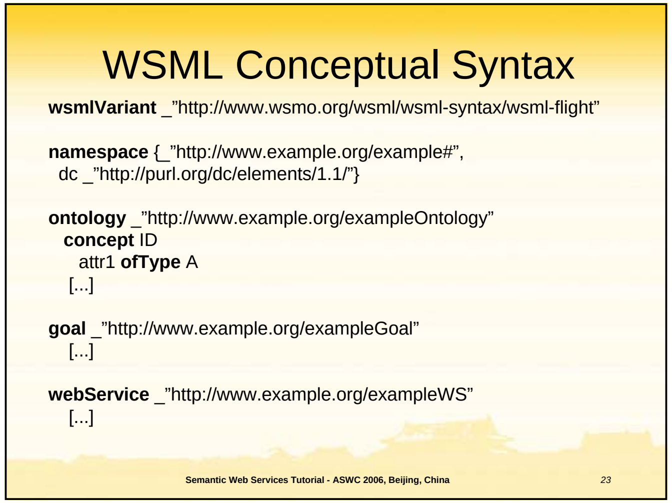
Non functional properties	(see before)
Imported Ontologies	importing existing ontologies where no heterogeneities arise
Used mediators	OO Mediators (ontology import with terminology mismatch handling)

Ontology Elements:

Concepts	set of concepts that belong to the ontology, incl.
Attributes	set of attributes that belong to a concept
Relations	define interrelations between several concepts
Functions	special type of relation (unary range = return value)
Instances	set of instances that belong to the represented ontology
Axioms	axiomatic expressions in ontology (logical statement)

Specification Language: WSML





WSML Conceptual Syntax

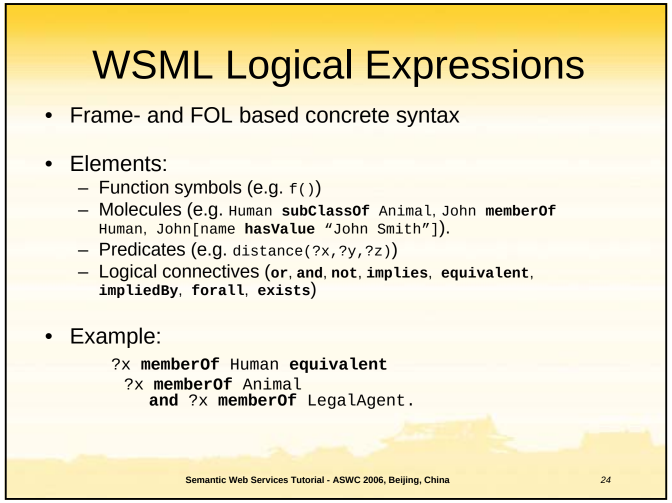
wsmlVariant _”http://www.wsmo.org/wsml/wsml-syntax/wsml-flight”

namespace {_”http://www.example.org/example#”,
dc _”http://purl.org/dc/elements/1.1/”}

ontology _”http://www.example.org/exampleOntology”
concept ID
attr1 **ofType** A
[...]

goal _”http://www.example.org/exampleGoal”
[...]

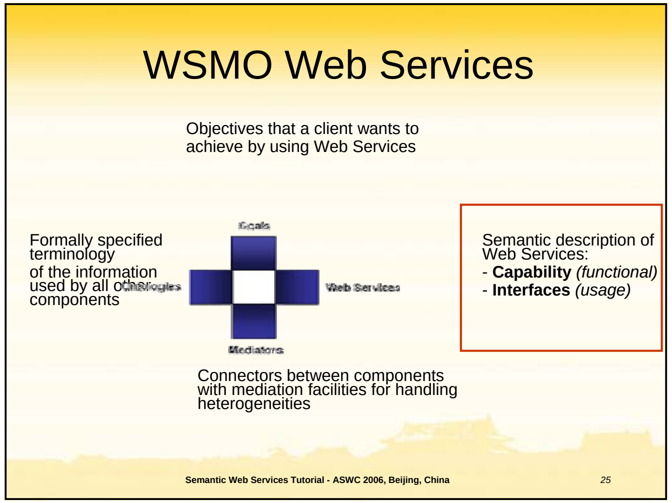
webService _”http://www.example.org/exampleWS”
[...]



WSML Logical Expressions

- Frame- and FOL based concrete syntax
- Elements:
  - Function symbols (e.g. `f()`)
  - Molecules (e.g. `Human **subClassOf** Animal`, `John **memberOf** Human`, `John[name **hasValue** “John Smith”]`).
  - Predicates (e.g. `distance(?x,?y,?z)`)
  - Logical connectives (**or**, **and**, **not**, **implies**, **equivalent**, **impliedBy**, **forall**, **exists**)
- Example:

```
?x memberOf Human equivalent
  ?x memberOf Animal
    and ?x memberOf LegalAgent.
```



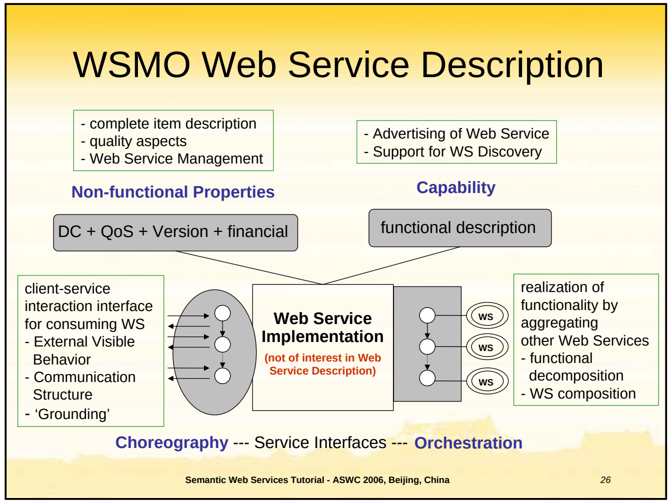
WSMO Web Services

Objectives that a client wants to achieve by using Web Services

Formally specified terminology of the information used by all other components

Semantic description of Web Services:
- **Capability** *(functional)*
- **Interfaces** *(usage)*

Connectors between components with mediation facilities for handling heterogeneities



WSMO Web Service Description

- complete item description
- quality aspects
- Web Service Management

Non-functional Properties

DC + QoS + Version + financial

- Advertising of Web Service
- Support for WS Discovery

Capability

functional description

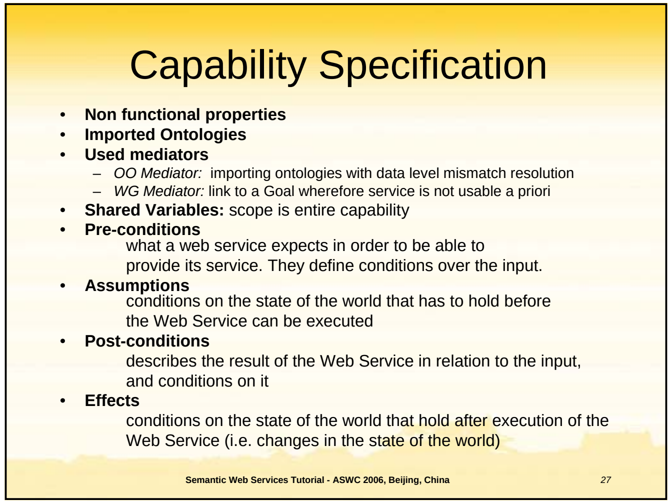
client-service interaction interface for consuming WS
- External Visible Behavior
- Communication Structure
- ‘Grounding’

Web Service Implementation

(not of interest in Web Service Description)

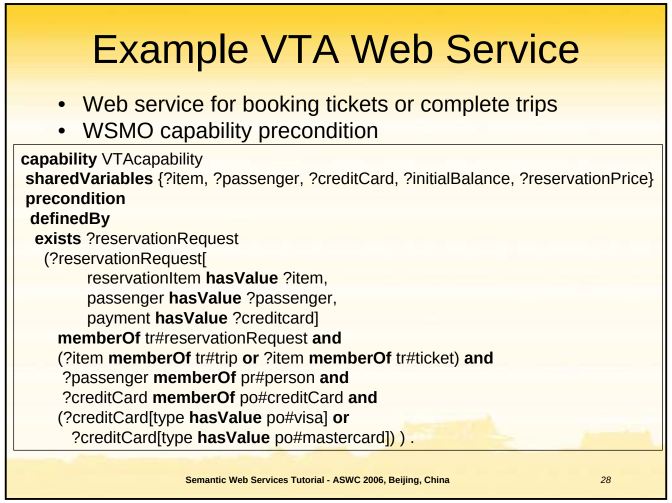
realization of functionality by aggregating other Web Services
- functional decomposition
- WS composition

Choreography --- Service Interfaces --- **Orchestration**



Capability Specification

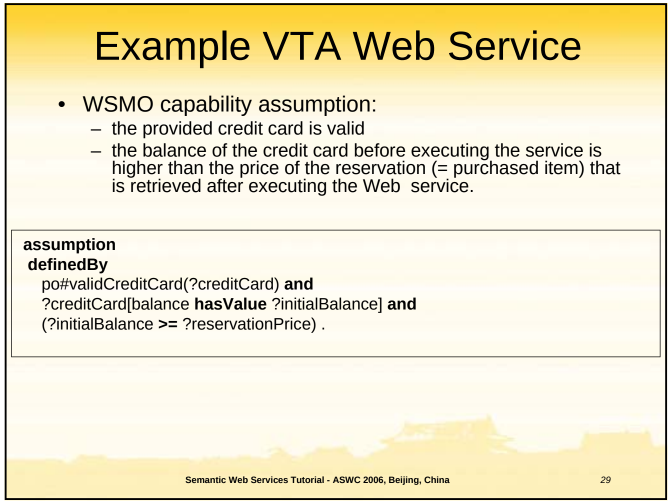
- **Non functional properties**
- **Imported Ontologies**
- **Used mediators**
  - *OO Mediator:* importing ontologies with data level mismatch resolution
  - *WG Mediator:* link to a Goal wherefore service is not usable a priori
- **Shared Variables:** scope is entire capability
- **Pre-conditions**
  what a web service expects in order to be able to provide its service. They define conditions over the input.
- **Assumptions**
  conditions on the state of the world that has to hold before the Web Service can be executed
- **Post-conditions**
  describes the result of the Web Service in relation to the input, and conditions on it
- **Effects**
  conditions on the state of the world that hold after execution of the Web Service (i.e. changes in the state of the world)



Example VTA Web Service

- Web service for booking tickets or complete trips
- WSMO capability precondition

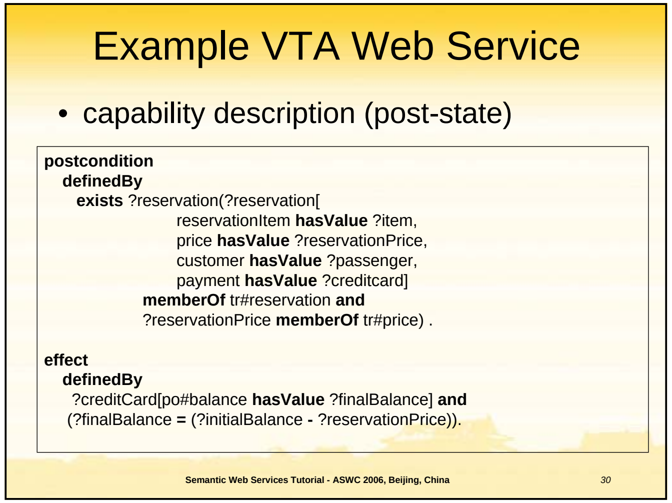
```
capability VTAcapability
 sharedVariables {?item, ?passenger, ?creditCard, ?initialBalance, ?reservationPrice}
 precondition
  definedBy
   exists ?reservationRequest
     (?reservationRequest[
           reservationItem hasValue ?item,
           passenger hasValue ?passenger,
           payment hasValue ?creditcard]
       memberOf tr#reservationRequest and
       (?item memberOf tr#trip or ?item memberOf tr#ticket) and
        ?passenger memberOf pr#person and
        ?creditCard memberOf po#creditCard and
       (?creditCard[type hasValue po#visa] or
          ?creditCard[type hasValue po#mastercard]) ) .
```



Example VTA Web Service

- WSMO capability assumption:
  - the provided credit card is valid
  - the balance of the credit card before executing the service is higher than the price of the reservation (= purchased item) that is retrieved after executing the Web service.

```
assumption
 definedBy
   po#validCreditCard(?creditCard) and
   ?creditCard[balance hasValue ?initialBalance] and
   (?initialBalance >= ?reservationPrice) .
```



Example VTA Web Service

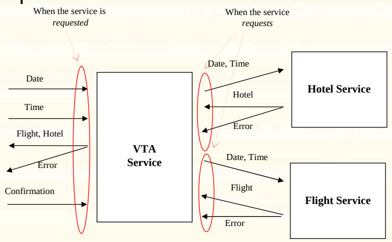
- capability description (post-state)

```
postcondition
   definedBy
     exists ?reservation(?reservation[
                    reservationItem hasValue ?item,
                    price hasValue ?reservationPrice,
                    customer hasValue ?passenger,
                    payment hasValue ?creditcard]
               memberOf tr#reservation and
               ?reservationPrice memberOf tr#price) .

effect
   definedBy
     ?creditCard[po#balance hasValue ?finalBalance] and
    (?finalBalance = (?initialBalance - ?reservationPrice)).
```

Choreography & Orchestration

- VTA example:



- **Choreography** = how to interact with the service to consume its functionality
- **Orchestration** = how service functionality is achieved by aggregating other Web Services

Ontologized Abstract State Machines

- **Description**
  - Vocabulary:
    - ontology constructs used in service interface description
    - usage for information interchange: in, out, shared, controlled
  - States:
    - a stable status in the information space
    - defined by attribute values of ontology instances
  - Guarded Transition:
    - state transition
    - general structure: ***if*** (condition) ***then*** (update)
      - condition on current state, update = changes in state transition
      - all GT(ω) whose condition is fulfilled fire in parallel
- **Usage:**
  - partners A, B commence interaction with empty Ω_A, Ω_B
  - Ω_A, Ω_B are updated via Guarded Transitions in each state
  - interaction termination state when A, B have no further transition rules

Example Hotel Web Service

- choreography interface (state signature)

```
interface htl#BookHotelInterface
  choreography
    stateSignature
      importsOntology htl#simpleHotelOntology
      in
        htl#HotelRequest withGrounding _"http://...",
        htl#HotelConfirm withGrounding _"http://...",
        htl#HotelCancel withGrounding  _"http://..."
      out
        htl#HotelNotAvailable withGrounding _"http://...",
        htl#HotelOffer withGrounding _"http://..."
      shared
        htl#Hotel,
        htl#HotelAvailable,
        htl#HotelBooked
```

Example Hotel Web Service

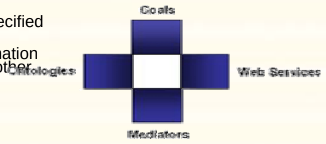
- choreography interface (transition rules)

```
ctl_state {htl#start,htl#offerMade,htl#noAvail,htl#confirmed,htl#cancelled}
transitionRules
  if (ctl_state = htl#start) then
   forall {?req,?date,?loc,?client} with
     ?req[trv#date hasValue ?date, trv#location hasValue ?loc,
            htl#client hasValue ?client] memberOf htl#HotelRequest
          do
            add(htl#offer(?req)[trv#date hasValue ?date,
              trv#hotelName hasValue ?name, trv#location hasValue ?loc,
              htl#client hasValue ?client] memberOf htl#HotelOffer)
            ctl_state := htl#offerMade
            |
            add(htl#notAvailable(?req)[trv#date hasValue ?date,
              trv#location hasValue ?loc] memberOf htl#HotelNotAvailable)
            ctl_state := htl#noAvail
          endForall
       endIf
```

WSMO Goals

Objectives that a client wants to achieve by using Web Services

Formally specified terminology of the information used by all other components



Semantic description of Web Services:
- **Capability** *(functional)*
- **Interfaces** *(usage)*

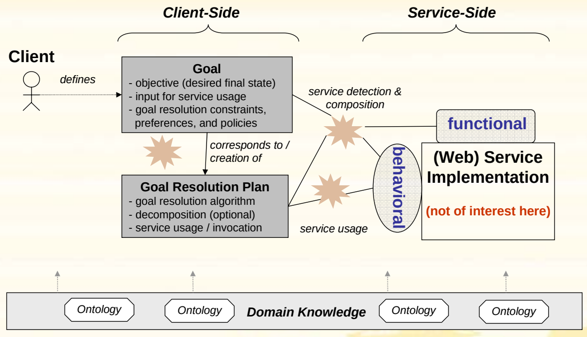
Connectors between components with mediation facilities for handling heterogeneities

Goals

client objective specification along with all information needed for automated resolution

- **Goal-driven Approach**, derived from AI rational agent approach
  - ontological de-coupling of Requester and Provider
  - ‘intelligent’ mechanisms detect suitable services for solving the Goal
  - service re-use & knowledge-level client side support
- **Usage of Goals within Semantic Web Services**
  - A Requester (human or machine) defines a Goal to be resolved independently (i.e. subjectively) on the knowledge level
  - SWS techniques / systems automatically determine Web Services to be used for resolving the Goal (discovery, composition, execution, etc.)
  - Goal Resolution Management is realized in implementations

Goal-driven Architecture

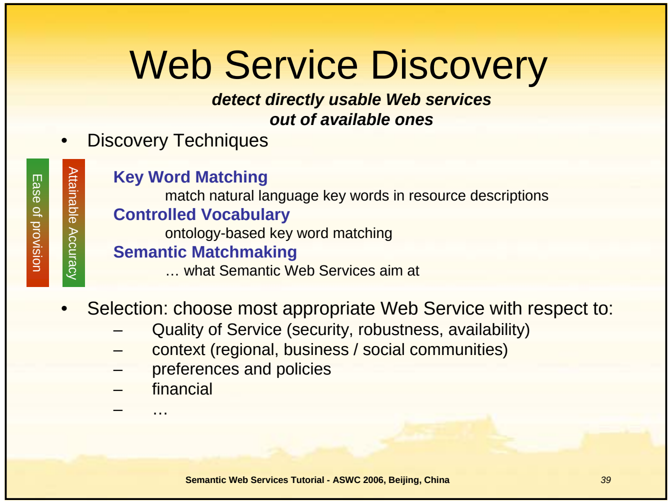


Goal
- objective (desired final state)
- input for service usage
- goal resolution constraints, preferences, and policies

Goal Resolution Plan
- goal resolution algorithm
- decomposition (optional)
- service usage / invocation

Goal Specification

- **Item Description & Terminology Import**
  - non-functional properties
  - imported Ontologies & used mediators
- **Requested Capability**
  - specifies objective (with PAPE)
  - instantiation for concrete request (concrete input data)
- **Client Choreography**
  - counterpart of Web service choreography interface
  - for invocation and consumption of Web services
  - one for each usable Web service
  - described as a WSMO choreography
- **Goal Decomposition**
  - defines “desired workflow”
  - collection of subgoals with control- & data flow
  - described as WSMO orchestration



Web Service Discovery

detect directly usable Web services out of available ones

- Discovery Techniques

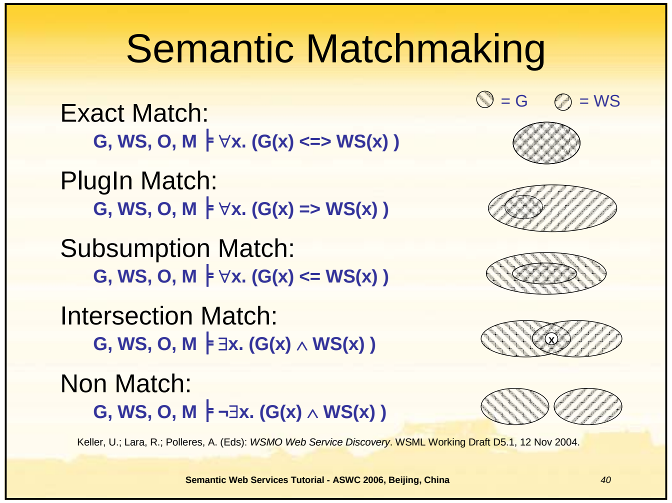
Ease of provision
Attainable Accuracy

Key Word Matching
match natural language key words in resource descriptions

Controlled Vocabulary
ontology-based key word matching

Semantic Matchmaking
… what Semantic Web Services aim at

- Selection: choose most appropriate Web Service with respect to:
  - Quality of Service (security, robustness, availability)
  - context (regional, business / social communities)
  - preferences and policies
  - financial
  - …



Semantic Matchmaking

▨ = G ▨ = WS

Exact Match:
$G, WS, O, M \models \forall x. (G(x) <=> WS(x))$

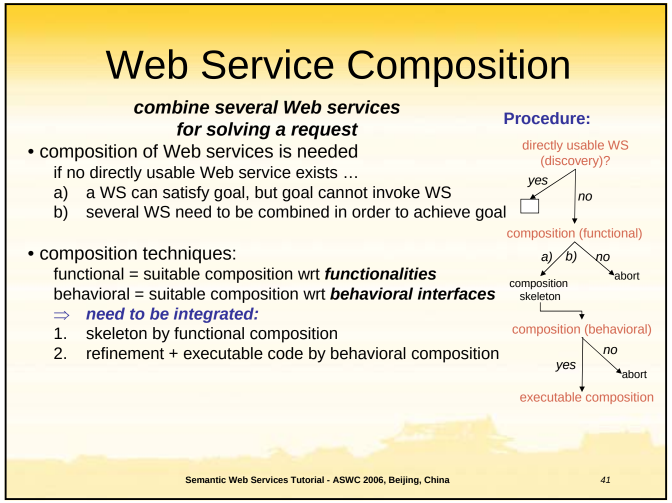
PlugIn Match:
$G, WS, O, M \models \forall x. (G(x) => WS(x))$

Subsumption Match:
$G, WS, O, M \models \forall x. (G(x) <= WS(x))$

Intersection Match:
$G, WS, O, M \models \exists x. (G(x) \wedge WS(x))$

Non Match:
$G, WS, O, M \models \neg\exists x. (G(x) \wedge WS(x))$

Keller, U.; Lara, R.; Polleres, A. (Eds): *WSMO Web Service Discovery*. WSML Working Draft D5.1, 12 Nov 2004.

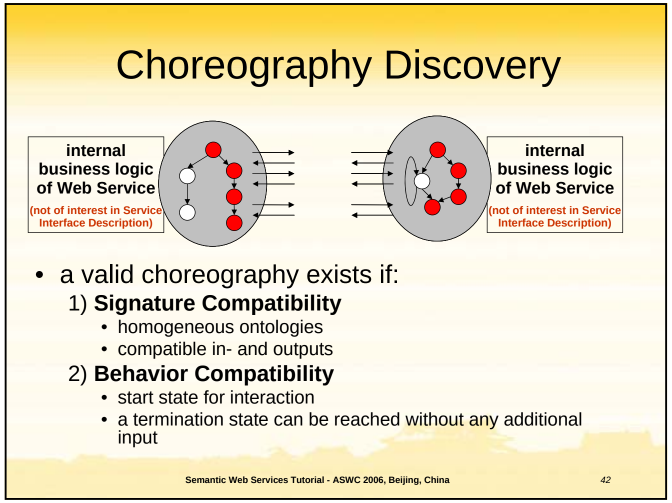


Web Service Composition

combine several Web services for solving a request

- composition of Web services is needed
  if no directly usable Web service exists …
  a) a WS can satisfy goal, but goal cannot invoke WS
  b) several WS need to be combined in order to achieve goal
- composition techniques:
  functional = suitable composition wrt ***functionalities***
  behavioral = suitable composition wrt ***behavioral interfaces***
  ⇒ ***need to be integrated:***
  1. skeleton by functional composition
  2. refinement + executable code by behavioral composition

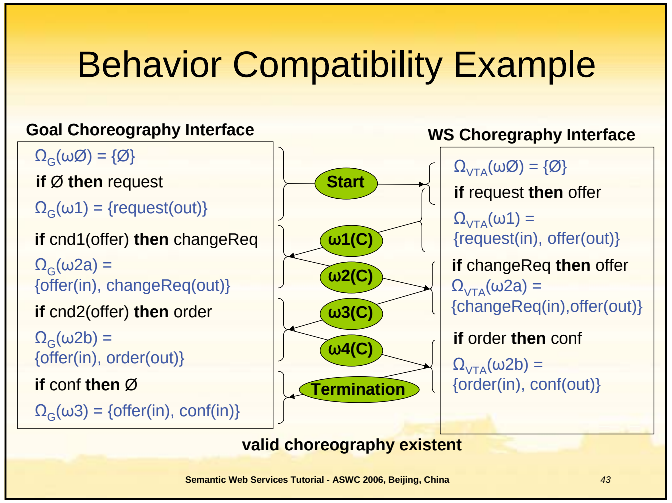
Procedure:



Choreography Discovery

internal business logic of Web Service
(not of interest in Service Interface Description)
internal business logic of Web Service
(not of interest in Service Interface Description)

- a valid choreography exists if:
  1) **Signature Compatibility**
     - homogeneous ontologies
     - compatible in- and outputs
  2) **Behavior Compatibility**
     - start state for interaction
     - a termination state can be reached without any additional input



Behavior Compatibility Example

Goal Choreography Interface

$\Omega_G(\omega\emptyset) = \{\emptyset\}$

if Ø **then** request

$\Omega_G(\omega 1) = \{request(out)\}$

if cnd1(offer) **then** changeReq

$\Omega_G(\omega 2a) = \{offer(in), changeReq(out)\}$

if cnd2(offer) **then** order

$\Omega_G(\omega 2b) = \{offer(in), order(out)\}$

if conf **then** Ø

$\Omega_G(\omega 3) = \{offer(in), conf(in)\}$

WS Choregraphy Interface

$\Omega_{VTA}(\omega\emptyset) = \{\emptyset\}$

if request **then** offer

$\Omega_{VTA}(\omega 1) = \{request(in), offer(out)\}$

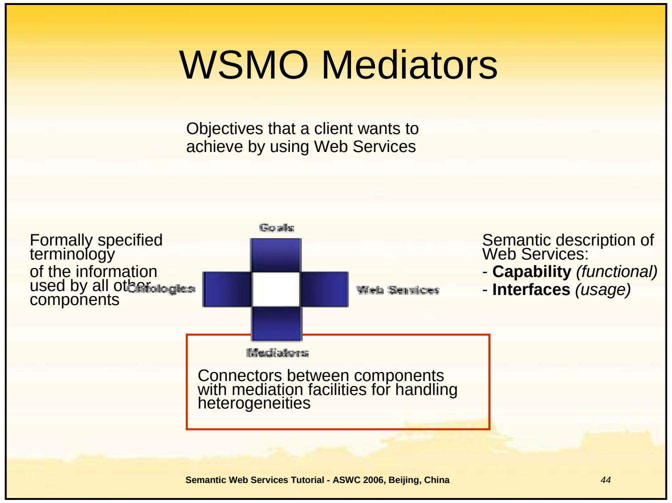
if changeReq **then** offer

$\Omega_{VTA}(\omega 2a) = \{changeReq(in),offer(out)\}$

if order **then** conf

$\Omega_{VTA}(\omega 2b) = \{order(in), conf(out)\}$

valid choreography existent



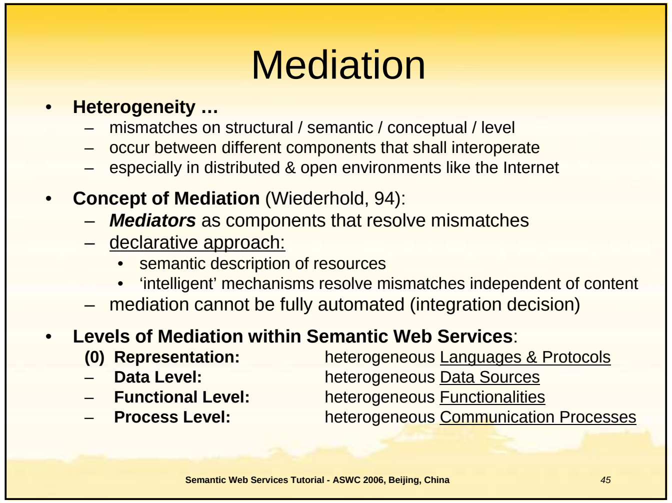
WSMO Mediators

Objectives that a client wants to achieve by using Web Services

Formally specified terminology of the information used by all other components

Semantic description of Web Services:
- **Capability** *(functional)*
- **Interfaces** *(usage)*

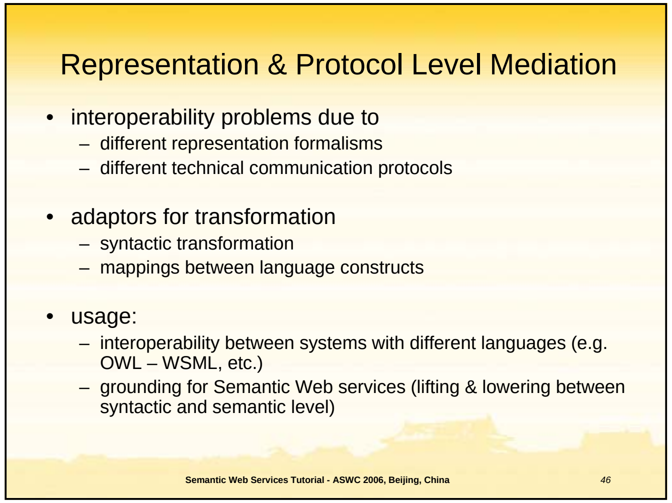
Connectors between components with mediation facilities for handling heterogeneities



Mediation

- **Heterogeneity …**
  - mismatches on structural / semantic / conceptual / level
  - occur between different components that shall interoperate
  - especially in distributed & open environments like the Internet
- **Concept of Mediation** (Wiederhold, 94):
  - ***Mediators*** as components that resolve mismatches
  - declarative approach:
    - semantic description of resources
    - ‘intelligent’ mechanisms resolve mismatches independent of content
  - mediation cannot be fully automated (integration decision)
- **Levels of Mediation within Semantic Web Services**:

(0)	**Representation:**	heterogeneous Languages & Protocols
–	**Data Level:**	heterogeneous Data Sources
–	**Functional Level:**	heterogeneous Functionalities
–	**Process Level:**	heterogeneous Communication Processes

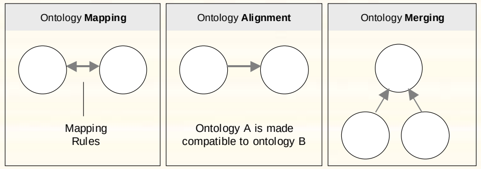


Representation & Protocol Level Mediation

- interoperability problems due to
  - different representation formalisms
  - different technical communication protocols
- adaptors for transformation
  - syntactic transformation
  - mappings between language constructs
- usage:
  - interoperability between systems with different languages (e.g. OWL – WSML, etc.)
  - grounding for Semantic Web services (lifting & lowering between syntactic and semantic level)

Data Mediation Techniques

- Ontology Integration Techniques



- semi-automatic
  - human intervention needed for “integration decision
  - graphical support for ontology mapping as central technique

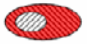
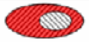
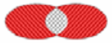
Functional Level Mediation

- requested and provided functionalities do not match precisely

=> conditions under which Web Service is usable for solving a Goal

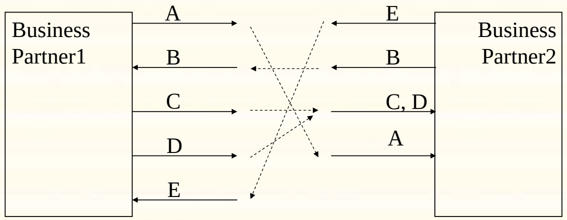
  - usage constraint explication
  - goal refinement
  - goal adjustment

- delta-relations = relation & difference of functional descriptions

 subsume	$\psi \models \phi$, i.e.: $\forall x. \psi(x) \Rightarrow \phi(x)$.	$\Delta = \phi \wedge \neg\psi$. $\phi = \psi \vee \Delta$. $\psi = \phi \wedge \neg\Delta$.
 plugin	$\phi \models \psi$, i.e.: $\forall x. \phi(x) \Rightarrow \psi(x)$.	$\Delta = \psi \wedge \neg\phi$. $\phi = \psi \wedge \neg\Delta$. $\psi = \phi \vee \Delta$.
 intersection	$\exists x.(\phi(x) \wedge \psi(x)) \wedge$ $\neg(\forall x. \psi(x) \Rightarrow \phi(x)) \wedge$ $\neg(\forall x. \phi(x) \Rightarrow \psi(x))$.	$\Delta = (\phi \wedge \neg\psi) \vee (\neg\phi \wedge \psi)$. $\phi = \psi \oplus \Delta$; $\psi = \phi \oplus \Delta$ with $A \oplus B \Leftrightarrow (A \vee B) \wedge \neg(A \wedge B)$.

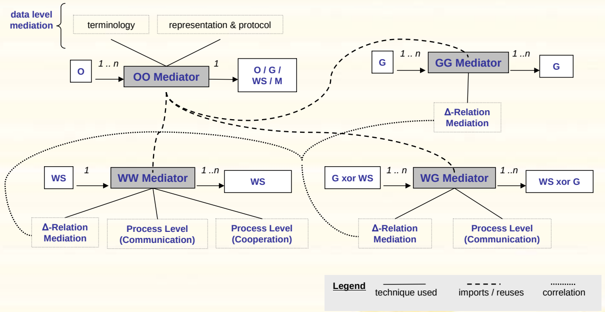
Process Level Mediation

- not a priori compatible behavior interfaces for communication & information interchange



- partially resolvable by “process mediation patterns”

WSMO Mediators Overview



Other Approaches

- WSMO is not the only proposal for an SWS Framework …

OWL-S:
- upper ontology for semantically describing Web services
- chronologically first, consortium mainly USA

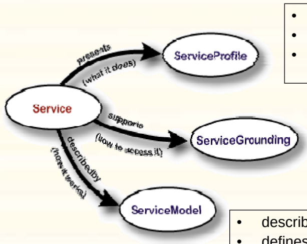
SWSF:
- process model for Web Services
- result of SWSI (international working group)

WSDL-S:
- semantic annotation of WSDL descriptions
- LSDIS Lap (Amit Seth Group) and IBM

- Discussed here:
  - Central Features
  - Commonalities and Differences

OWL-S

Upper Ontology for Web Service Descriptions



- capability description (IOPE)
- non-functional properties
- usage: (1) WS advertisement, (2) WS request formulation

- specification of service access information
- builds upon WSDL to define message structure and physical binding layer
- specifies communication protocols & language, transport mechanisms, etc.

- describes internal processes of the service
- defines service interaction protocol for (a) consumption and (b) WS interaction
- process types: simple, atomic, composite
- specifies: (1) abstract messages (ontological content), (2) control flow constructs, (3) *perform* construct

OWL-S and WSMO

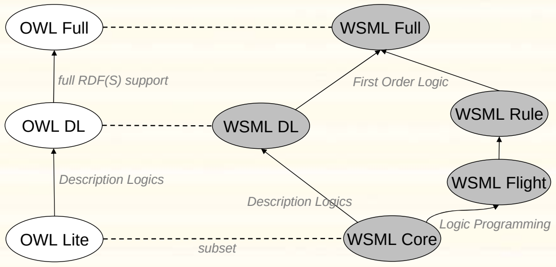
- **OWL-S** = ontology and language to describe Web services
- **WSMO** = ontology and language for core elements of Semantic Web Service systems

Main Description Elements Correlation:

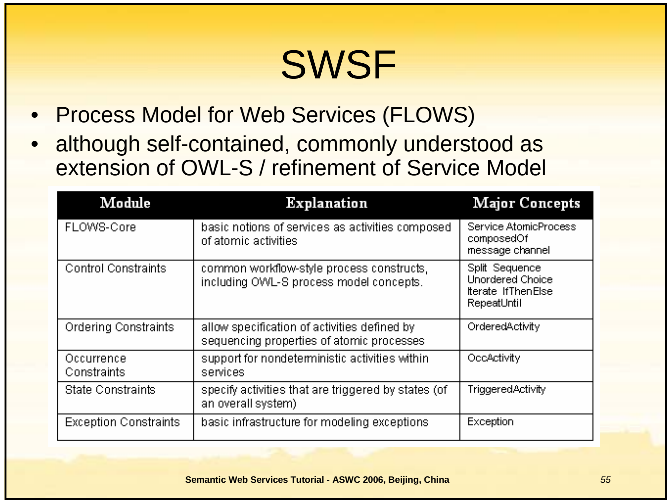
OWL-S Profile	≈ **WSMO capability + non-functional properties**
OWL-S Process Model	≈ **WSMO Service Interfaces**
OWL-S Grounding	≈ **current WSMO Grounding**

- Goals and Mediators not in scope
- deficiencies in Service Model (process description model / language not adequate) => SWSF

OWL and WSML



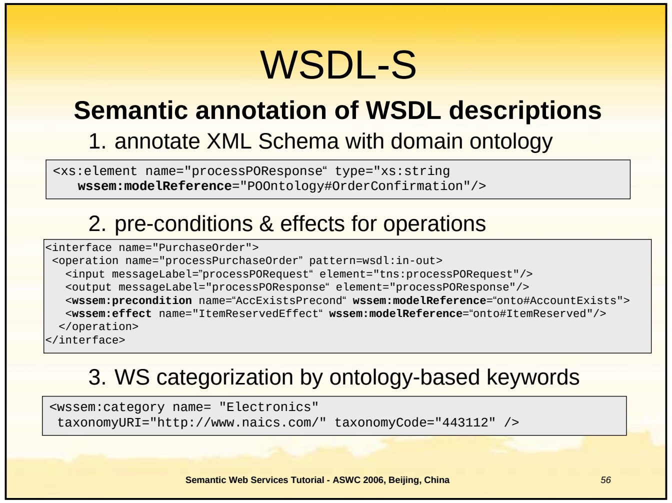
WSML aims at overcoming deficiencies of OWL



SWSF

- Process Model for Web Services (FLOWS)
- although self-contained, commonly understood as extension of OWL-S / refinement of Service Model

Module	Explanation	Major Concepts
FLOWS-Core	basic notions of services as activities composed of atomic activities	Service AtomicProcess composedOf message channel
Control Constraints	common workflow-style process constructs, including OWL-S process model concepts.	Split Sequence Unordered Choice Iterate IfThenElse RepeatUntil
Ordering Constraints	allow specification of activities defined by sequencing properties of atomic processes	OrderedActivity
Occurrence Constraints	support for nondeterministic activities within services	OccActivity
State Constraints	specify activities that are triggered by states (of an overall system)	TriggeredActivity
Exception Constraints	basic infrastructure for modeling exceptions	Exception



WSDL-S

Semantic annotation of WSDL descriptions

1. annotate XML Schema with domain ontology

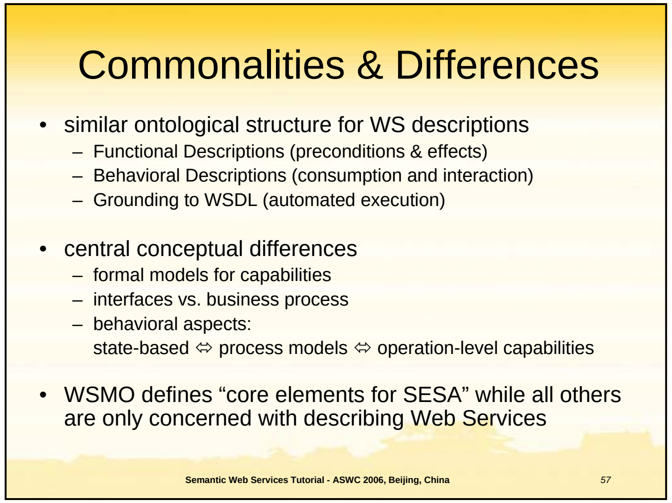
```
<xs:element name="processPOResponse“ type="xs:string
   wssem:modelReference="POOntology#OrderConfirmation"/>
```

2. pre-conditions & effects for operations

```
<interface name="PurchaseOrder">
 <operation name="processPurchaseOrder” pattern=wsdl:in-out>
   <input messageLabel=”processPORequest“ element="tns:processPORequest"/>
   <output messageLabel="processPOResponse“ element="processPOResponse"/>
   <wssem:precondition name=“AccExistsPrecond“ wssem:modelReference=“onto#AccountExists">
   <wssem:effect name="ItemReservedEffect“ wssem:modelReference=“onto#ItemReserved"/>
  </operation>
</interface>
```

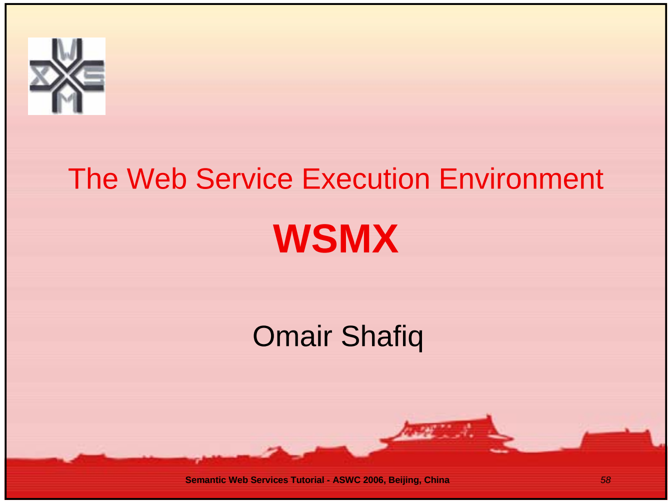
3. WS categorization by ontology-based keywords

```
<wssem:category name= "Electronics"
 taxonomyURI="http://www.naics.com/" taxonomyCode="443112" />
```



Commonalities & Differences

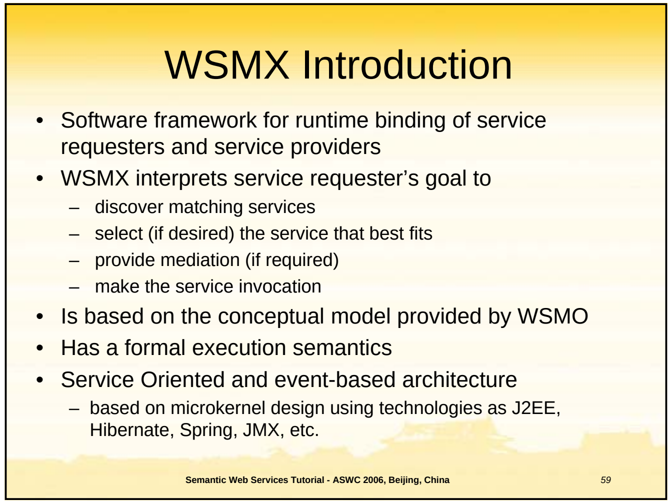
- similar ontological structure for WS descriptions
  - Functional Descriptions (preconditions & effects)
  - Behavioral Descriptions (consumption and interaction)
  - Grounding to WSDL (automated execution)
- central conceptual differences
  - formal models for capabilities
  - interfaces vs. business process
  - behavioral aspects:
    state-based ⇔ process models ⇔ operation-level capabilities
- WSMO defines “core elements for SESA” while all others are only concerned with describing Web Services



The Web Service Execution Environment

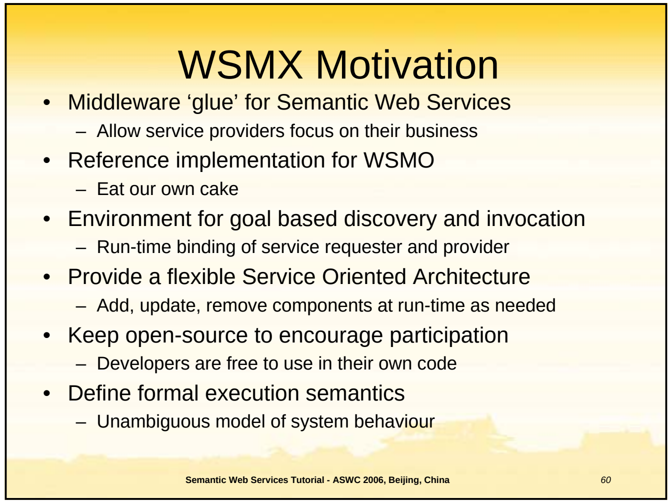
WSMX

Omair Shafiq



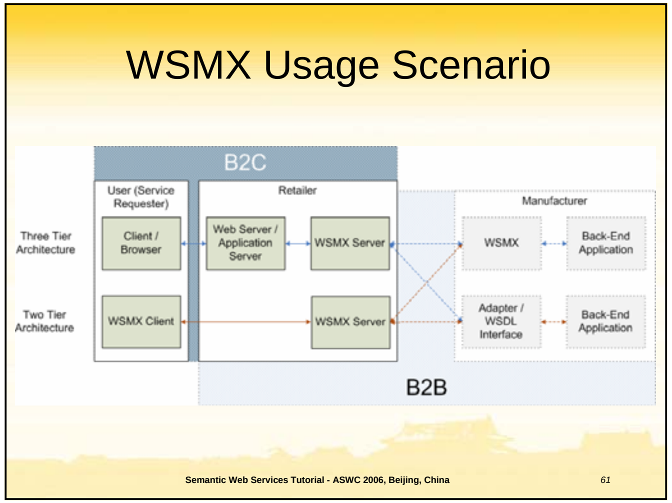
WSMX Introduction

- Software framework for runtime binding of service requesters and service providers
- WSMX interprets service requester’s goal to
  - discover matching services
  - select (if desired) the service that best fits
  - provide mediation (if required)
  - make the service invocation
- Is based on the conceptual model provided by WSMO
- Has a formal execution semantics
- Service Oriented and event-based architecture
  - based on microkernel design using technologies as J2EE, Hibernate, Spring, JMX, etc.

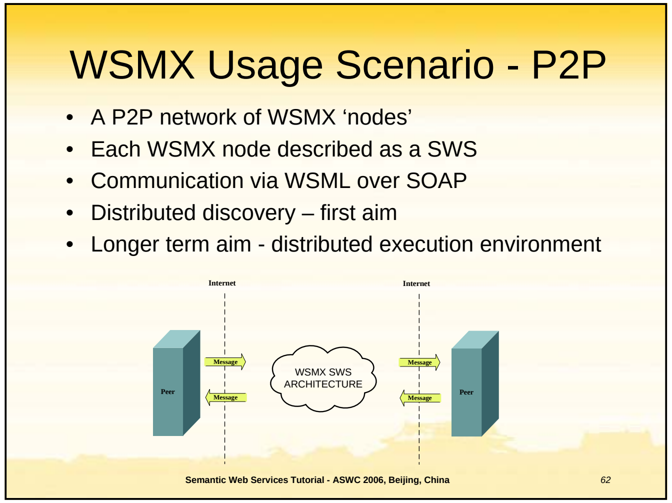


WSMX Motivation

- Middleware ‘glue’ for Semantic Web Services
  - Allow service providers focus on their business
- Reference implementation for WSMO
  - Eat our own cake
- Environment for goal based discovery and invocation
  - Run-time binding of service requester and provider
- Provide a flexible Service Oriented Architecture
  - Add, update, remove components at run-time as needed
- Keep open-source to encourage participation
  - Developers are free to use in their own code
- Define formal execution semantics
  - Unambiguous model of system behaviour



WSMX Usage Scenario

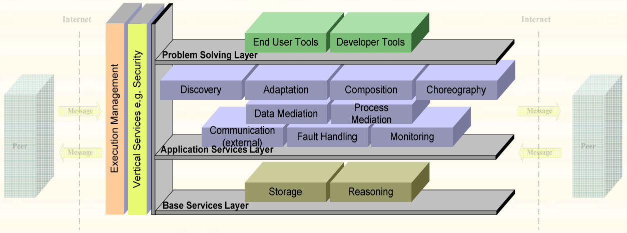


WSMX Usage Scenario - P2P

- A P2P network of WSMX ‘nodes’
- Each WSMX node described as a SWS
- Communication via WSML over SOAP
- Distributed discovery – first aim
- Longer term aim - distributed execution environment

Internet
Internet
Message
Message
Peer
WSMX SWS ARCHITECTURE
Message
Message
Peer

WSMX Usage Scenario - P2P



Design Principles

- Strong Decoupling & Strong Mediation
  - autonomous components with mediators for interoperability
- Interface vs. Implementation
  - distinguish interface (= description) from implementation (=program)
- Peer to Peer
  - interaction between equal partners (in terms of control)

WSMO Design Principles == WSMX Design Principles == SOA Design Principles

Benefits of SOA

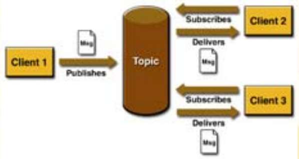
- Better reuse
  - Build new functionality (new execution semantics) on top of existing Business Services
- Well defined interfaces
  - Manage changes without affecting the Core System
- Easier Maintainability
  - Changes/Versions are not all-or-nothing
- Better Flexibility

Service Oriented State

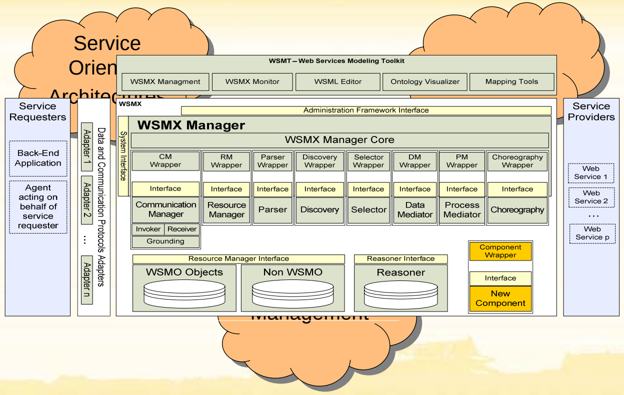
- The interface to the service is implementation-independent
- The service can be dynamically invoked
  - Runtime binding
- The service is self-contained
  - Maintains its own state

Messaging

- Messaging is peer-to-peer facility
- Distributed communication
  - Loosely coupled
- Sender does not need to know receiver (and vice versa)
- Asynchronous mechanism to communicate between software applications

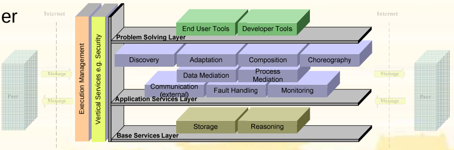


WSMX Architecture



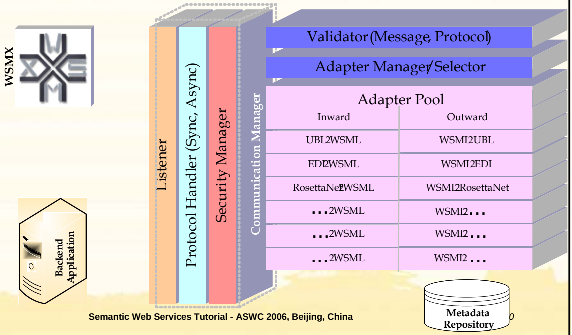
Selected Components

- Adapters
- Parser
- Invoker
- Choreography
- Process Mediator
- Discovery
- Data Mediator
- Resource Manager
- Reasoning



Adapters

- To overcome data representation mismatches on the communication layer
- Transforms the format of a received message into WSML compliant format
- Based on mapping rules



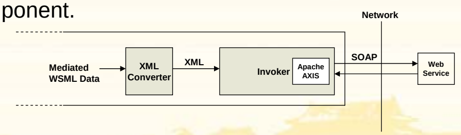
Inward	Outward
UBL2WSML	WSMl2UBL
EDI2WSML	WSMl2EDI
RosettaNet2WSML	WSMl2RosettaNet
...2WSML	WSMl2...
...2WSML	WSMl2...
...2WSML	WSMl2...

Parser

- WSML compliant parser
  - Code handed over to wsmo4j initiative http://wsmo4j.sourceforge.net/
- Validates WSML description files
- Compiles WSML description into internal memory model
- Stores WSML description persistently (using Resource Manager)

Communication Mgr – Invoker

- WSMX uses
  - The SOAP implementation from Apache AXIS
  - The Apache Web Service Invocation Framework (WSIF)
- WSMO service descriptions are grounded to WSDL
- Both RPC and Document style invocations possible
- Input parameters for the Web Services are translated from WSML to XML using an additional XML Converter component.



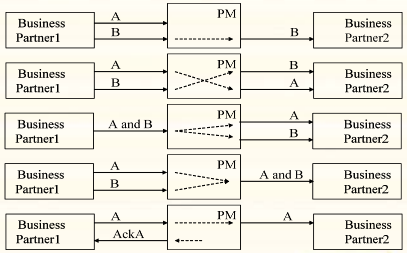
Choreography

- Requester and provider have their own observable communication patterns
  - Choreography part of WSMO
- Choreography instances are loaded for the requester and provider
  - Both requester and provider have their own WSMO descriptions
- Choreography Engine
  - Evaluation of transition rules - prepares the available data
  - Sends data to the Process Mediator - filters, changes or replaces data
  - Receives data from PM and forwards it to the Communication manager - data to be finally sent to the communication partner

Process Mediator

- Requester and provider have their own communication patterns
- Only if the two match precisely, a direct communication may take place
- At design time equivalences between the choreographies' conceptual descriptions is determined and stored as set of rules
- The Process Mediator provides the means for runtime analyses of two choreography instances and uses mediators to compensate possible mismatches

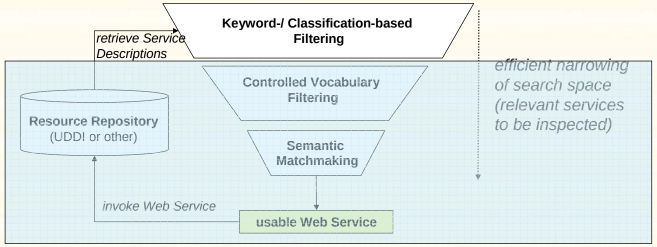
Process Mediator – Addressed mismatches



Discovery

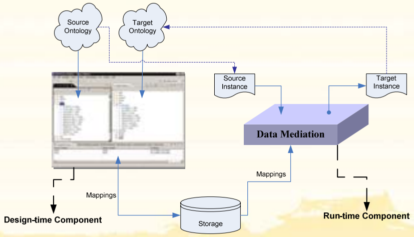
- Responsible for finding appropriate Web Services to achieve a goal (discovery)
- Current discovery component is based on simple matching
  - Keywords identified in the NFP of the goal
  - Matched against NFPs of the published WSs
  - Variable set of NFPs to be considered for this process
  - To be extended
    - Values in NFPs might be concepts from ontologies
    - More elaborate string matching algorithms
- Advanced semantic discovery in prototypical stage

Discovery


Keyword-/ Classification-based Filtering
retrieve Service Descriptions
Controlled Vocabulary Filtering
Resource Repository (UDDI or other)
Semantic Matchmaking
efficient narrowing of search space (relevant services to be inspected)
invoke Web Service
usable Web Service

Data Mediator

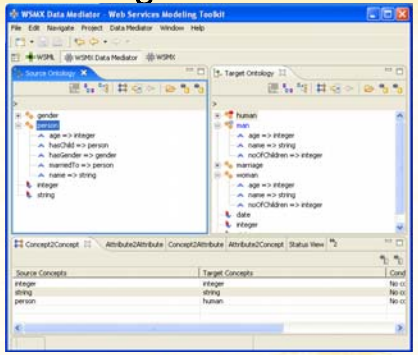
- Ontology-to-ontology mediation
- A set of mapping rules are defined and then executed
- Initially rules are defined semi-automatic
- Create for each source instance the target instance(s)



Design-time

- Inputs
  - Source Ontology and Target Ontology
- Features
  - Graphical interface
  - Set of mechanism towards semi-automatic creation of mappings
  - Capturing the semantic relationships identified in the process
  - Storing these mappings in a persistent storage
- Output
  - Abstract representation of the mappings

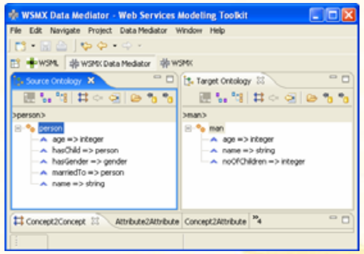
Design-time Phase



Source Concepts	Target Concepts	Cond
integer	integer	No c
string	string	No c
person	human	No c

Design-time Phase - Approach, Decomposition and Mapping Context

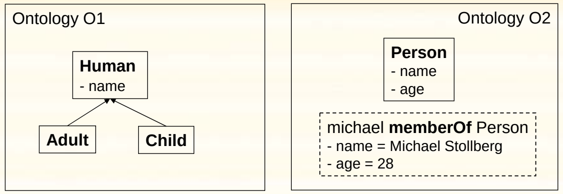
- Bottom-up -> training set
- Top-down -> decomposition, context



Ontology Mapping Language

- Language Neutral Mapping Language
  - mapping definitions on meta-layer (i.e. on generic ontological constructs)
  - independent of ontology specification language
  - “Grounding” to specific languages for execution (WSML, OWL, F-Logic)
- Main Features:
  - Mapping Document (sources, mappings, mediation service)
  - direction of mapping (uni- / bidirectional)
  - conditions / logical expressions for data type mismatch handling, restriction of mapping validity, and complex mapping definitions
  - mapping constructs (ex: classMapping, classAtrributeMapping, instanceMapping)
  - mapping operators

Mapping Language Example



classMapping(**unidirectional** o2:Person o1.Adult
attributeValueCondition(o2.Person.age **>=** 18))

this allows to transform the instance ‘michael’ of concept person in ontology O2 into a valid instance of concept ‘adult’ in ontology O1

Run-Time Data Mediator

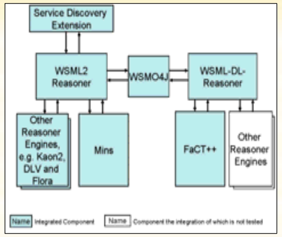
Data Mediation

- Main Mediation Scenario: Instance Transformation
- Inputs
  - Incoming data
    - Source ontology instances
- Features
  - Completely automatic process
  - Grounding of the abstract mappings to a concrete language
    - WSML
  - Uses a reasoner to evaluate the mapping rules
    - MINS
- Outputs
  - Mediated data
    - Target ontology instances

Resource Manager

- Stores internal memory model to a data store
- Decouples storage mechanism from the rest of WSMX
- Data model is compliant to WSMO API
- Independent of any specific data store implementation i.e. database and storage mechanism

Reasoner


Service Discovery Extension
WSML2 Reasoner
WSMO4J
WSML-DL-Reasoner
Other Reasoner Engines, e.g. Kaon2, DLV and Flora
Mins
FaCT++
Other Reasoner Engines
Name Integrated Component
Name Component the integration of which is not tested

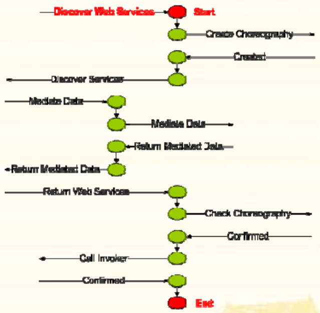
- Mins
  - Datalog + Negation + Function Symbols Reasoner Engine
  - Features
    - Built-in predicates
    - Function symbols
    - Stratified negation

- WSMO4J
  - validation, serialization and parsing
- WSML2Reasoner
  - Reasoning API
    - mapping fromWSML to a vendor-neutral rule representation
  - Contains:
    - Common API for WSML Reasoners
    - Transformations of WSML to tool-specific input data (query answering or instance retrieval)
- WSML-DL-Reasoner
  - Features:
    - T-Box reasoning (provided by FaCT++)
    - Querying for all concepts
    - Querying for the equivalents, for the children, for the descendants, for the parents and for all ancestors of a given concept
    - Testing the satisfiability of a given concept with respect to the knowledge base
    - Subsumption test of two concepts with respect to the knowledge base
    - Wrapper of WSML-DL to the XML syntax of DL used in the DIG interface

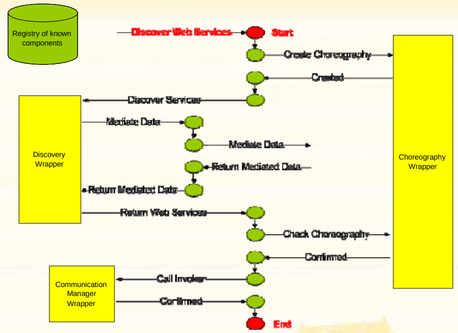
System Entry Points

- *achieveGoal* (WSMLDocument): Context
- *getWebServices* (WSMLDocument): Context
- *invokeWebService* (WSMLDocument, Context): Context

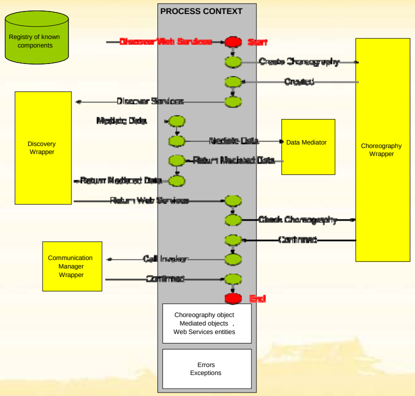
Define “Business” Process



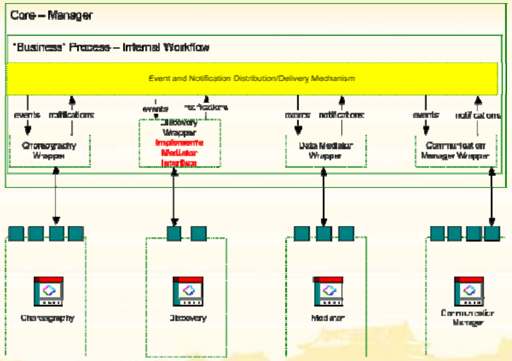
Generate Wrappers for Components



Context Data



Event-based Implementation



Web Services Modeling Toolkit

- The aim of the Web Services Modeling Toolkit (WSMT) is to provide high-quality tools for designing, mediating and using Semantic Web Services, through the WSMO paradigm.
- The focus is currently on the following areas:
  - Creation of ontologies, web services, goals and mediators in WSMO
  - Creation of mappings between pairs of ontologies to allow runtime instance transformation
  - Management of Execution Environments for Semantic Web Services like WSMX and IRSIII

WSML Perspective

- Perspectives in the Eclipse framework allow for a number of Editors and views to be grouped and positions.
- The WSML perspective offers editors and views related to engineering of semantic descriptions in WSMO through the WSML language.
- Other General features include:
  - WSML file validation
  - Problems view (errors and warnings on files in the workspace)
  - Label highlighting (marking of errors and warnings in navigator view)

WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

WSML Visualizer

Views

Navigator view

Problems view

WSML Reasoner

WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

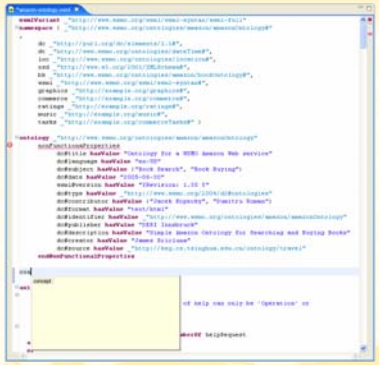
WSML Visualizer

Views

Navigator view

Problems view

WSML Reasoner



```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-full"
namespace { _"http://www.wsmo.org/ontologies/amazon/amazonOntology#"
 ,
     dc _"http://purl.org/dc/elements/1.1#",
     dt _"http://www.wsmo.org/ontologies/dateTime#",
     loc _"http://www.wsmo.org/ontologies/location#",
     xsd _"http://www.w3.org/2001/XMLSchema#",
     bk _"http://www.wsmo.org/ontologies/amazon/bookOntology#",
     wsml _"http://www.wsmo.org/wsml/wsml-syntax#",
     graphics _"http://example.org/graphics#",
     commerce _"http://example.org/commerce#",
     ratings _"http://example.org/ratings#",
     music _"http://example.org/music#",
     tasks _"http://example.org/commerceTasks#" }

ontology _"http://www.wsmo.org/ontologies/amazon/amazonOntology"
     nonFunctionalProperties
          dc#title hasValue "Ontology for a WSMO Amazon Web service"
          dc#language hasValue "en-US"
          dc#subject hasValue {"Book Search", "Book Buying"}
          dc#date hasValue "2005-06-30"
          wsml#version hasValue "$Revision: 1.10 $"
          dc#type hasValue _"http://www.wsmo.org/2004/d2#ontologies"
          dc#contributor hasValue {"Jacek Kopecky", "Dumitru Roman"}
          dc#format hasValue "text/html"
          dc#identifier hasValue _"http://www.wsmo.org/ontologies/amazon/amazonOntology"
          dc#publisher hasValue "DERI Innsbruck"
          dc#description hasValue "Simple Amazon Ontology for Searching and Buying Books"
          dc#creator hasValue "James Scicluna"
          dc#source hasValue _"http://keg.cs.tsinghua.edu.cn/ontology/travel"
     endNonFunctionalProperties

con
 concept
```

WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

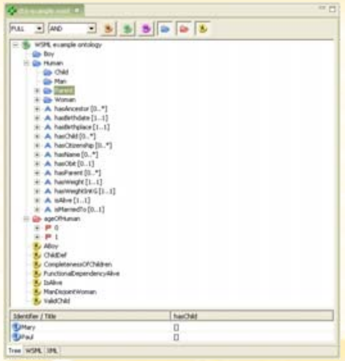
WSML Visualizer

Views

Navigator view

Problems view

WSML Reasoner



- WSML example ontology
  - Boy
  - Human
    - Child
    - Man
    - Parent
    - Woman
    - hasAncestor [0..*]
    - hasBirthdate [1..1]
    - hasBirthplace [1..1]
    - hasChild [0..*]
    - hasCitizenship [0..*]
    - hasName [0..*]
    - hasObit [0..1]
    - hasParent [0..*]
    - hasWeight [1..1]
    - hasWeightInKG [1..1]
    - isAlive [1..1]
    - isMarriedTo [0..1]
  - ageOfHuman
    - 0
    - 1
  - ABoy
  - ChildDef
  - CompletenessOfChildren
  - FunctionalDependencyAlive
  - IsAlive
  - ManDisjointWoman
  - ValidChild

Identifier / Title	hasChild
Mary	□
Paul	□

WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

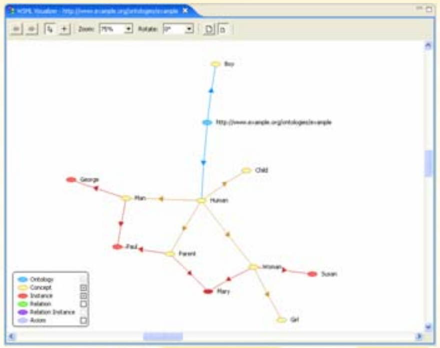
WSML Visualizer

Views

Navigator view

Problems view

WSML Reasoner



WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

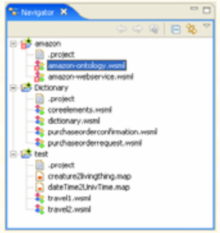
WSML Visualizer

Views

Navigator view

Problems view

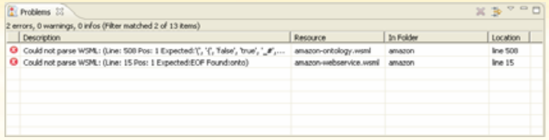
WSML Reasoner



Navigator

- amazon
  - .project
  - amazon-ontology.wsml
  - amazon-webservice.wsml
- Dictionary
  - .project
  - coreelements.wsml
  - dictionary.wsml
  - purchaseorderconfirmation.wsml
  - purchaseorderrequest.wsml
- test
  - .project
  - creature2livingthing.map
  - dateTime2UnivTime.map
  - travel1.wsml
  - travel2.wsml

WSML Perspective: Editors & Views



Problems

2 errors, 0 warnings, 0 infos (Filter matched 2 of 13 items)

Description	Resource	In Folder	Location
Could not parse WSML: (Line: 508 Pos: 1 Expected:'(', '{', 'false', 'true', '_#',...	amazon-ontology.wsml	amazon	line 508
Could not parse WSML: (Line: 15 Pos: 1 Expected:EOF Found:onto)	amazon-webservice.wsml	amazon	line 15

Views

Navigator view

Problems view

WSML Reasoner

WSML Perspective: Editors & Views

Editors

WSML Text Editor

WSML Conceptual Editor

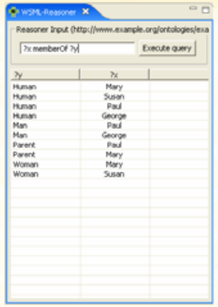
WSML Visualizer

Views

Navigator view

Problems view

WSML Reasoner



Reasoner Input (http://www.example.org/ontologies/exa

?x memberOf ?y

?y	?x
Human	Mary
Human	Susan
Human	Paul
Human	George
Man	Paul
Man	George
Parent	Paul
Parent	Mary
Woman	Mary
Woman	Susan

Abstract Mapping Language: Editors & Views

Editors

AML Text Editor

AML Conceptual Editor

AML View Based Editor

Views

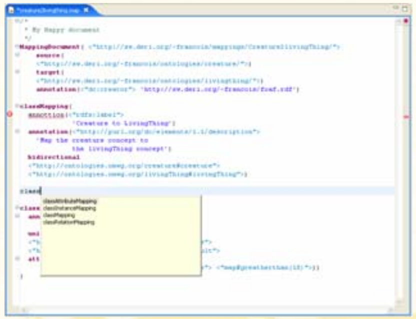
Concept 2 Concept View

Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View



```
/*
 * My Mappy document
 */
MappingDocument( <"http://sw.deri.org/~francois/mappings/CreatureZLivingThing/">
    source(
    <"http://sw.deri.org/~francois/ontologies/creature/">)
    target(
    <"http://sw.deri.org/~francois/ontologies/livingthing/">)
    annotation(<"dc:creator"> 'http://sw.deri.org/~francois/foaf.rdf')

classMapping(
  annotation(<"rdfs:label">
           'Creature to LivingThing')
  annotation(<"http://purl.org/dc/elements/1.1/description">
    'Map the creature concept to
           the LivingThing concept')
  bidirectional
  <"http://ontologies.omwg.org/creature#creature">
  <"http://ontologies.omwg.org/livingThing#livingThing">)

 class

class
  ann

  uni
  <"h                                    ">
  <"h                                   ult">
  att
                                        "> <"mapEgreaterthan(18)">))
)
```

Abstract Mapping Language: Editors & Views

Editors

AML Text Editor

AML Conceptual Editor

AML View Based Editor

Views

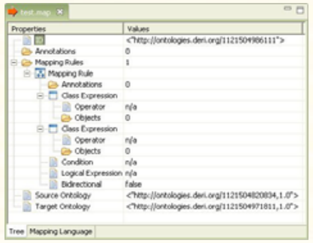
Concept 2 Concept View

Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View



Properties	Values
ID	<"http://ontologies.deri.org/1121504986111">
Annotations	0
Mapping Rules	1
Mapping Rule	
Annotations	0
Class Expression	
Operator	n/a
Objects	0
Class Expression	
Operator	n/a
Objects	0
Condition	n/a
Logical Expression	n/a
Bidirectional	false
Source Ontology	<"http://ontologies.deri.org/1121504820834,1.0">
Target Ontology	<"http://ontologies.deri.org/1121504971811,1.0">

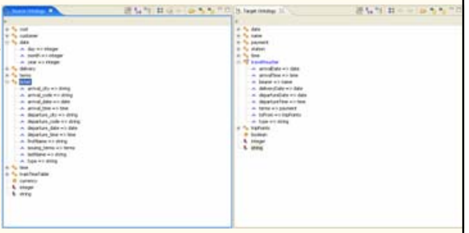
Abstract Mapping Language: Editors & Views

Editors

AML Text Editor

AML Conceptual Editor

AML View Based Editor



- cost
- customer
- date
  - day => integer
  - month => integer
  - year => integer
- delivery
- terms
- ticket
  - arrival_city => string
  - arrival_code => string
  - arrival_date => date
  - arrival_time => time
  - departure_city => string
  - departure_code => string
  - departure_date => date
  - departure_time => time
  - firstName => string
  - issuing_terms => terms
  - lastName => string
  - type => string
- time
- trainTimeTable
- currency
- integer
- string

- date
- name
- payment
- station
- time
- travelVoucher
  - arrivalDate => date
  - arrivalTime => time
  - bearer => name
  - deliveryDate => date
  - departureDate => date
  - departureTime => time
  - terms => payment
  - toFrom => tripPoints
  - type => string
- tripPoints
- boolean
- integer
- string

<u>Views</u>

Concept 2 Concept View

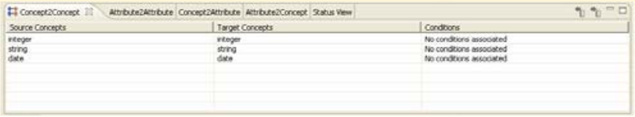
Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View

Abstract Mapping Language: Editors & Views



Source Concepts	Target Concepts	Conditions
integer	integer	No conditions associated
string	string	No conditions associated
date	date	No conditions associated

<u>Views</u>

Concept 2 Concept View

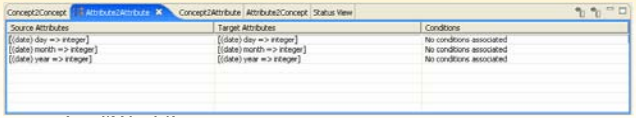
Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View

Abstract Mapping Language: Editors & Views



Source Attributes	Target Attributes	Conditions
[(date) day => integer]	[(date) day => integer]	No conditions associated
[(date) month => integer]	[(date) month => integer]	No conditions associated
[(date) year => integer]	[(date) year => integer]	No conditions associated

<u>Views</u>

Concept 2 Concept View

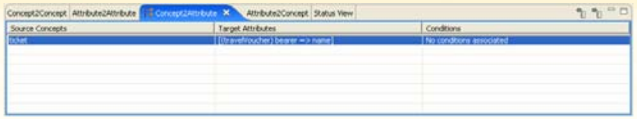
Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View

Abstract Mapping Language: Editors & Views



Source Concepts	Target Attributes	Conditions
ticket	[(travelVoucher) bearer => name]	No conditions associated

<u>Views</u>

Concept 2 Concept View

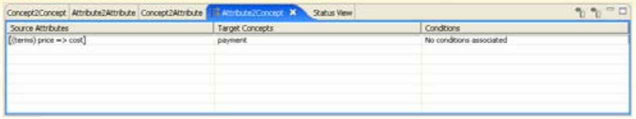
Attribute 2 Attribute View

Concept 2 Attribute View

Attribute 2 Concept View

Status View

Abstract Mapping Language: Editors & Views

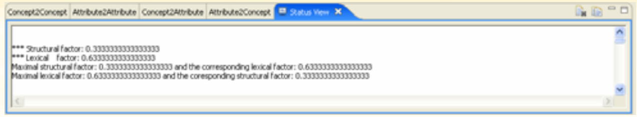


Source Attributes	Target Concepts	Conditions
[(terms) price => cost]	payment	No conditions associated

Views

Concept 2 Concept View
Attribute 2 Attribute View
Concept 2 Attribute View
Attribute 2 Concept View
Status View

Abstract Mapping Language: Editors & Views



```
*** Structural factor: 0.3333333333333333
*** Lexical   factor: 0.6333333333333333
Maximal structural factor: 0.3333333333333333 and the corresponding lexical factor: 0.6333333333333333
Maximal lexical factor: 0.6333333333333333 and the coresponding structural factor: 0.3333333333333333
```

Views

Concept 2 Concept View
Attribute 2 Attribute View
Concept 2 Attribute View
Attribute 2 Concept View
Status View

Conclusions

- Conceptual model is WSMO
- End to end functionality for executing SWS
- Has a formal execution semantics
- Real implementation
- Open source code base at SourceForge
  http://sourceforge.net/projects/wsmx/
- Event-driven component architecture
- WSMT – emerging tool to handle semantics

WSMX
Hands-on Session overview

Download URL:
www.wsmo.org/TR/d17/tutorials/aswc06-swstutorial.html

Omair Shafiq
Emilia Cimpian
Dumitru Roman
Michal Zaremba

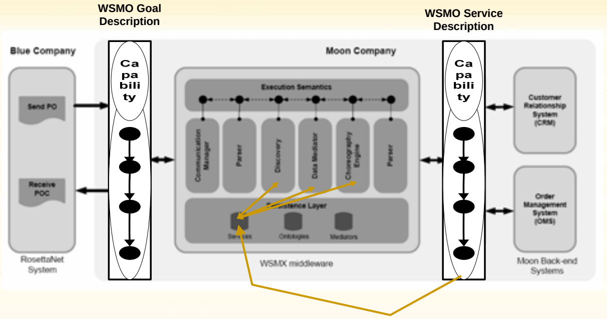
Use Case

- **Blue company**
  - Service Requestor, wants to buy computers and accessories
- **Moon company**
  - Service Provider, selling computer products
- **WSMX**
  - Acting as middleware to bring together Blue and Moon companies and to manage conversation between them

Why WSMX ?

- Blue company wants to buy computers but does not know any vendors
  - Service discovery required
- Blue does not want to change the data format with which it communicates or the order of the messages it exchanges
  - Data mediation and process mediation required
- Blue does not want to be bound to any one provider

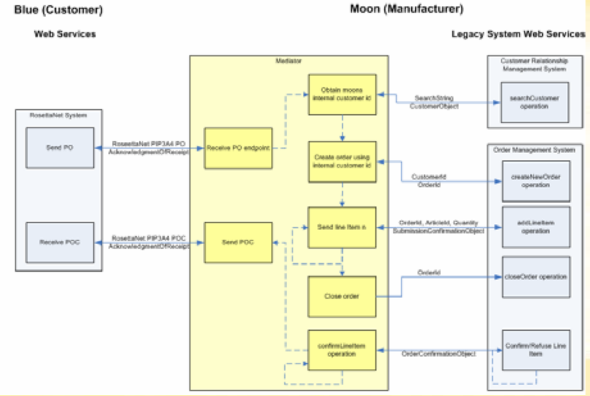
Application Scenario



Discovery Scenario Overview

- Blue's Goal
  - Purchase:
    - 20 power supplies for IBM R50 Notebooks
    - 20 SDRAM modules à 512 MB.
  - Shipment
    - 5 Notebooks R50 to customer in Bristol, UK
- Moon's Service
  - Sells and ships computers and accessories

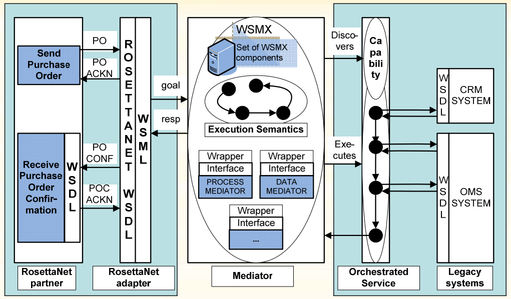
Interaction Scenario



Solution: Overview of Integration Stages

- 1 – Sending Request
  - Blue sends PO request
- 2 – Discovery and Conversation Setup
  - Discovery of service, setup of conversation
- 3 – Conversation with Requestor
  - Blue RosettaNet System: accepting purchase order request
- 4 – Conversation with Provider
  - CRM and OMS systems: opening order, adding line items, closing order
- 5 – Conversation with Requestor
  - order confirmation, end of conversation

Implementation & Demonstration



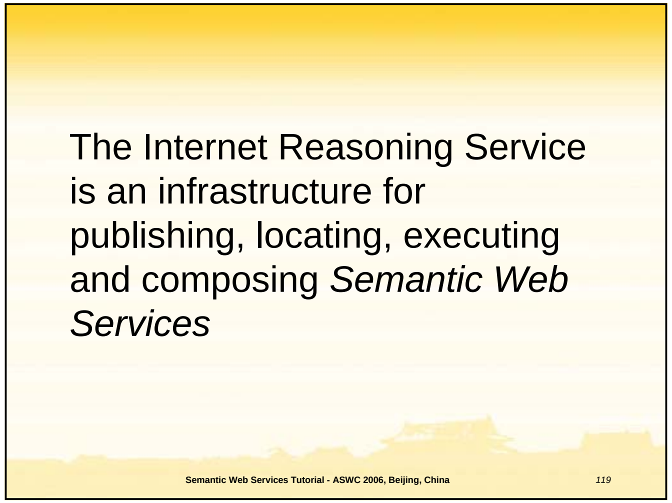
Blue company

Moon company

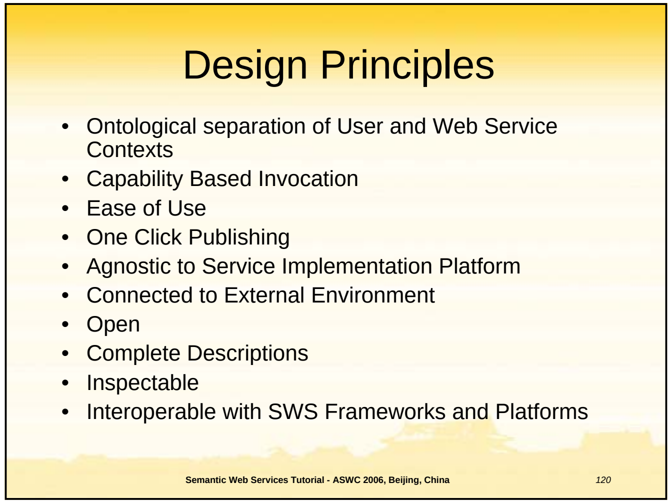
The Internet Reasoning Service
IRS III

John Domingue

Liliana Cabral

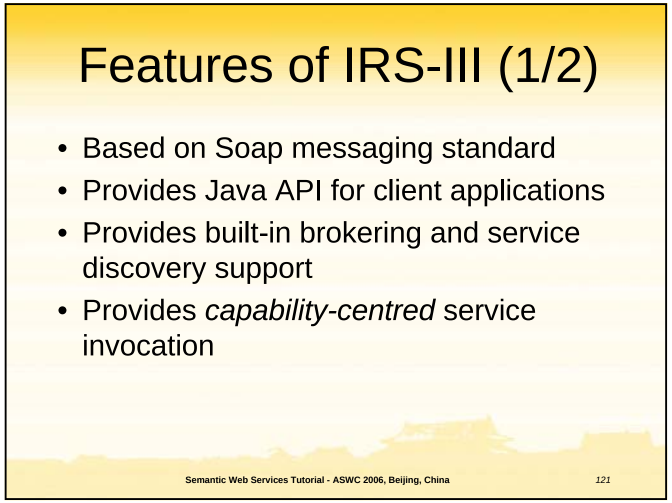


The Internet Reasoning Service is an infrastructure for publishing, locating, executing and composing *Semantic Web Services*



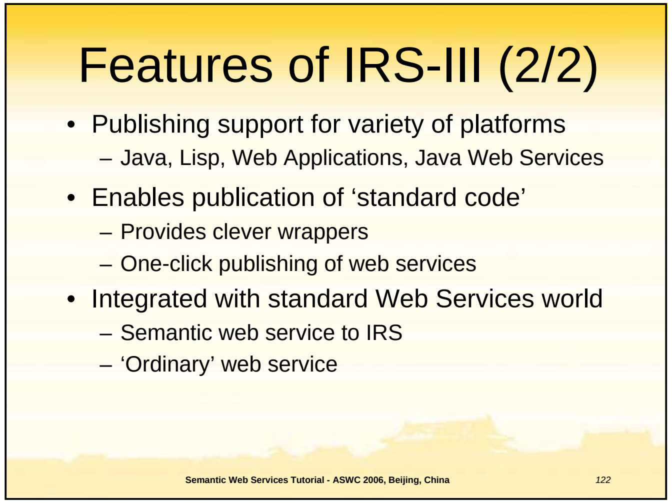
Design Principles

- Ontological separation of User and Web Service Contexts
- Capability Based Invocation
- Ease of Use
- One Click Publishing
- Agnostic to Service Implementation Platform
- Connected to External Environment
- Open
- Complete Descriptions
- Inspectable
- Interoperable with SWS Frameworks and Platforms



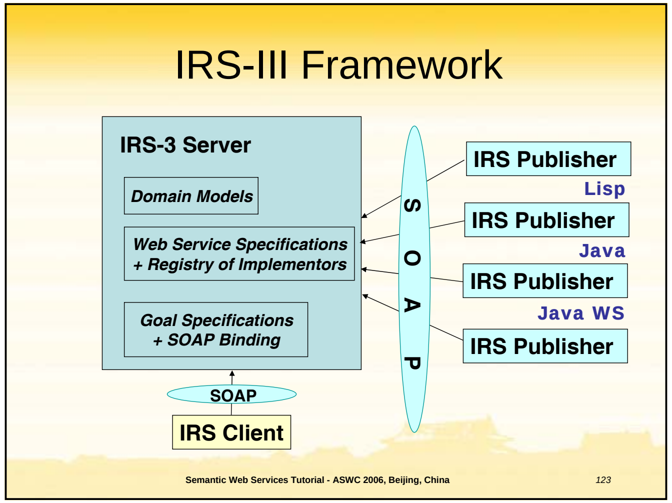
Features of IRS-III (1/2)

- Based on Soap messaging standard
- Provides Java API for client applications
- Provides built-in brokering and service discovery support
- Provides *capability-centred* service invocation



Features of IRS-III (2/2)

- Publishing support for variety of platforms
  - Java, Lisp, Web Applications, Java Web Services
- Enables publication of ‘standard code’
  - Provides clever wrappers
  - One-click publishing of web services
- Integrated with standard Web Services world
  - Semantic web service to IRS
  - ‘Ordinary’ web service



IRS-III Framework

IRS-3 Server

Domain Models

Web Service Specifications + Registry of Implementors

Goal Specifications + SOAP Binding

SOAP

IRS Client

S O A P

IRS Publisher

Lisp

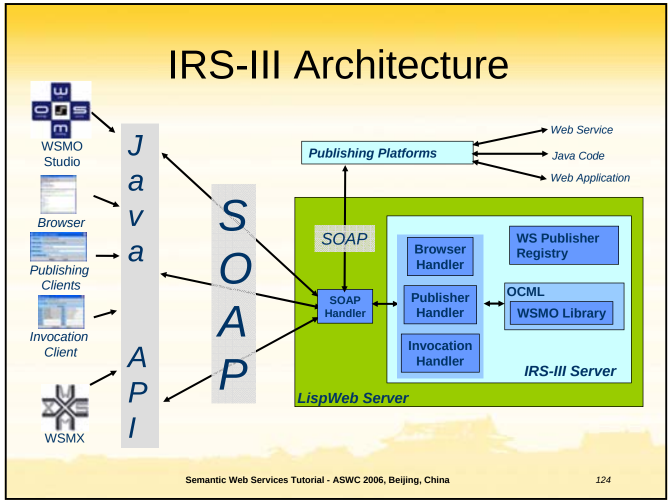
IRS Publisher

Java

IRS Publisher

Java WS

IRS Publisher



IRS-III Architecture

WSMO Studio

Browser

Publishing Clients

Invocation Client

WSMX

Java API

SOAP

Publishing Platforms

Web Service

Java Code

Web Application

SOAP

SOAP Handler

Browser Handler

Publisher Handler

Invocation Handler

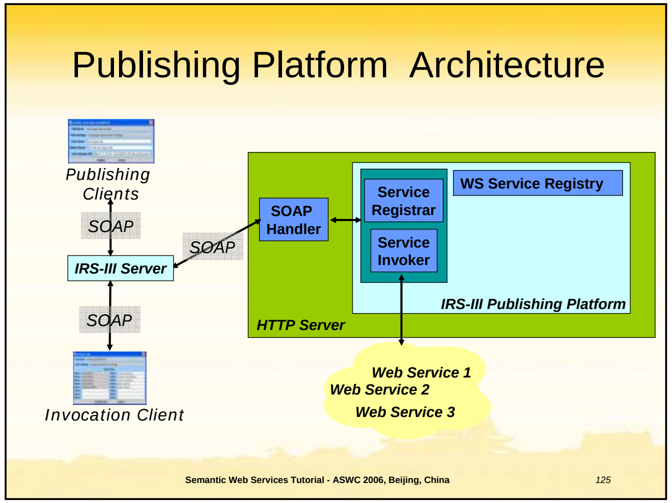
WS Publisher Registry

OCML

WSMO Library

IRS-III Server

LispWeb Server



Publishing Platform Architecture

Publishing Clients

SOAP

IRS-III Server

SOAP

Invocation Client

SOAP

SOAP Handler

Service Registrar

Service Invoker

WS Service Registry

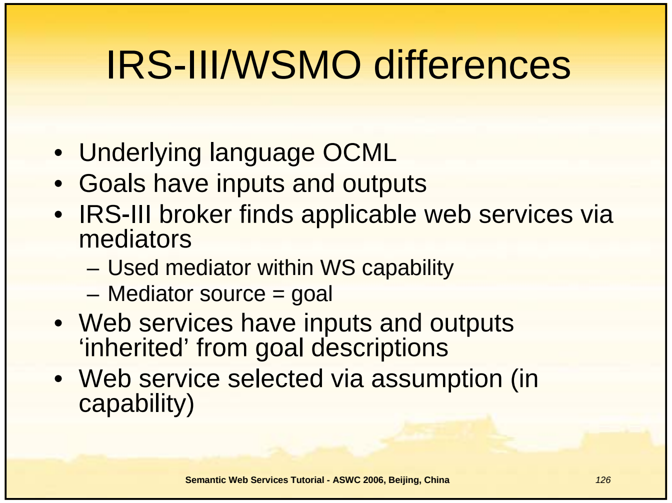
IRS-III Publishing Platform

HTTP Server

Web Service 1

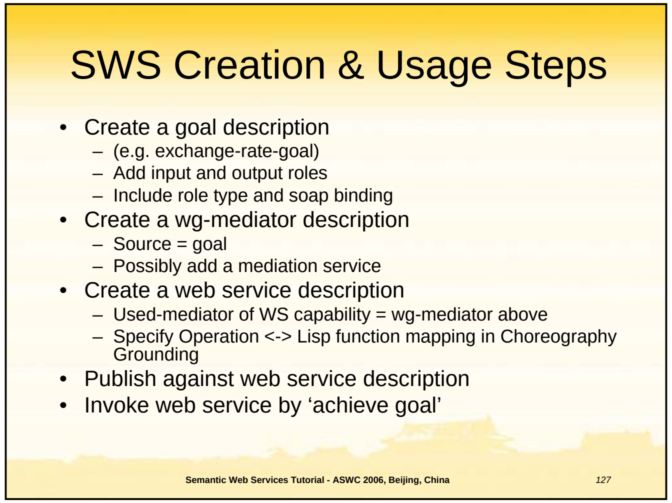
Web Service 2

Web Service 3



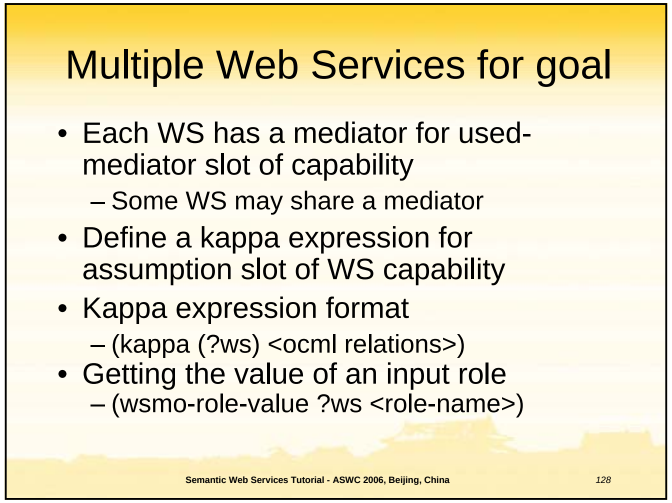
IRS-III/WSMO differences

- Underlying language OCML
- Goals have inputs and outputs
- IRS-III broker finds applicable web services via mediators
  - Used mediator within WS capability
  - Mediator source = goal
- Web services have inputs and outputs ‘inherited’ from goal descriptions
- Web service selected via assumption (in capability)



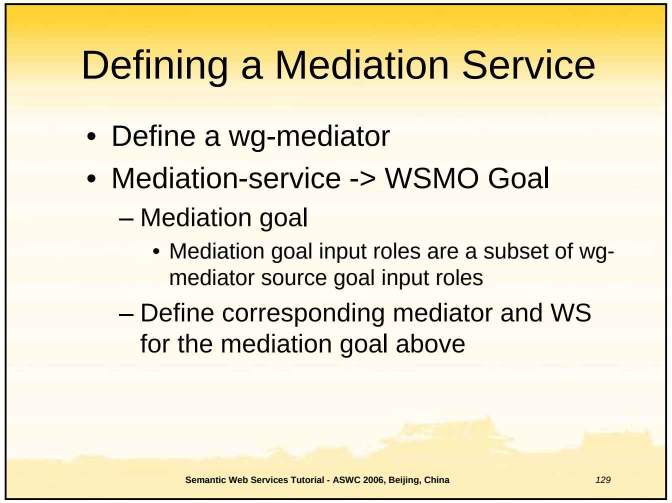
SWS Creation & Usage Steps

- Create a goal description
  - (e.g. exchange-rate-goal)
  - Add input and output roles
  - Include role type and soap binding
- Create a wg-mediator description
  - Source = goal
  - Possibly add a mediation service
- Create a web service description
  - Used-mediator of WS capability = wg-mediator above
  - Specify Operation <-> Lisp function mapping in Choreography Grounding
- Publish against web service description
- Invoke web service by ‘achieve goal’



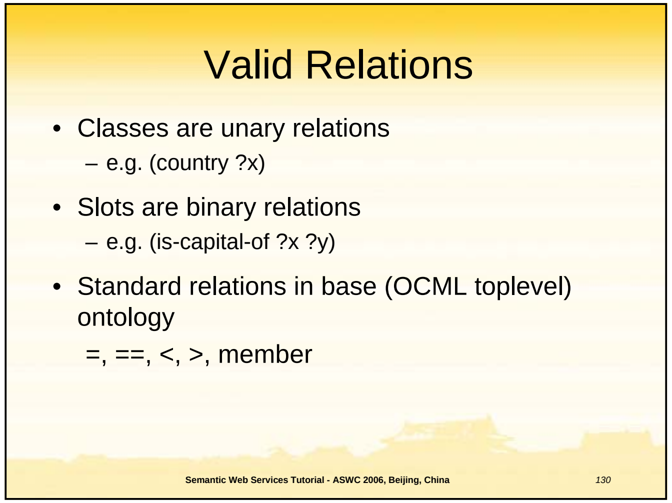
Multiple Web Services for goal

- Each WS has a mediator for used-mediator slot of capability
  - Some WS may share a mediator
- Define a kappa expression for assumption slot of WS capability
- Kappa expression format
  - (kappa (?ws) <ocml relations>)
- Getting the value of an input role
  - (wsmo-role-value ?ws <role-name>)



Defining a Mediation Service

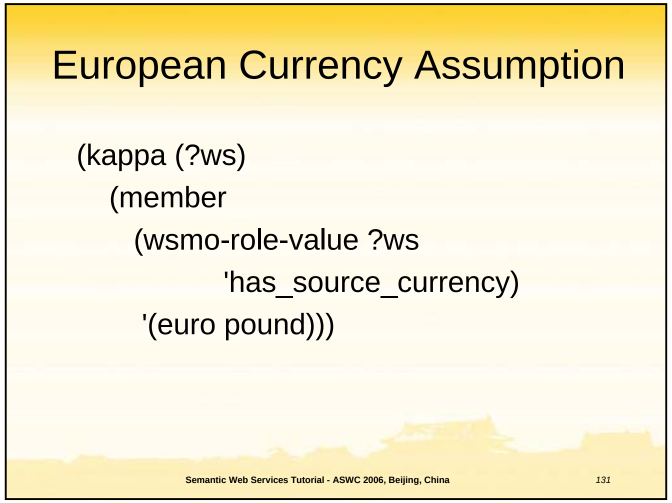
- Define a wg-mediator
- Mediation-service -> WSMO Goal
  - Mediation goal
    - Mediation goal input roles are a subset of wg-mediator source goal input roles
  - Define corresponding mediator and WS for the mediation goal above



Valid Relations

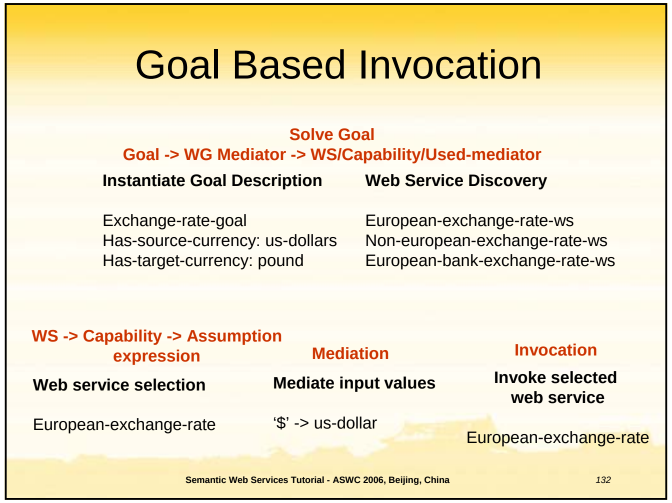
- Classes are unary relations
  - e.g. (country ?x)
- Slots are binary relations
  - e.g. (is-capital-of ?x ?y)
- Standard relations in base (OCML toplevel) ontology

  =, ==, <, >, member



European Currency Assumption

```
(kappa (?ws)
   (member
      (wsmo-role-value ?ws
              'has_source_currency)
       '(euro pound)))
```



Goal Based Invocation

Solve Goal
Goal -> WG Mediator -> WS/Capability/Used-mediator

Instantiate Goal Description

Exchange-rate-goal
Has-source-currency: us-dollars
Has-target-currency: pound

Web Service Discovery

European-exchange-rate-ws
Non-european-exchange-rate-ws
European-bank-exchange-rate-ws

WS -> Capability -> Assumption expression

Web service selection

European-exchange-rate

Mediation

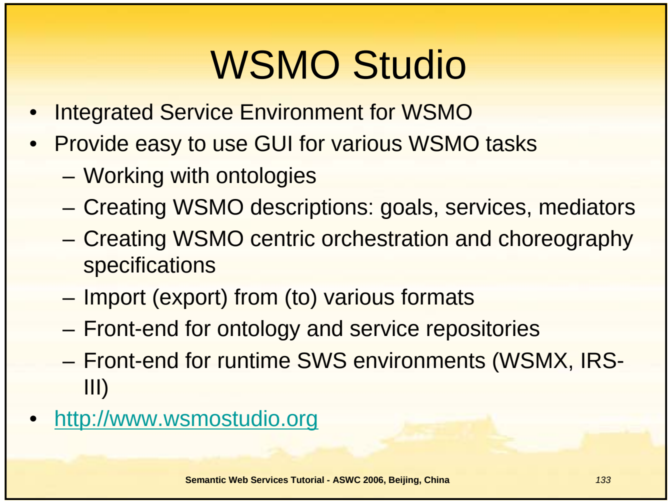
Mediate input values

'$' -> us-dollar

Invocation

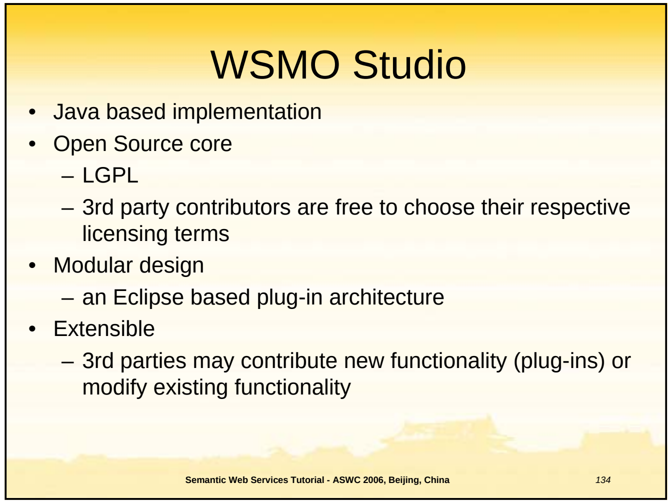
Invoke selected web service

European-exchange-rate



WSMO Studio

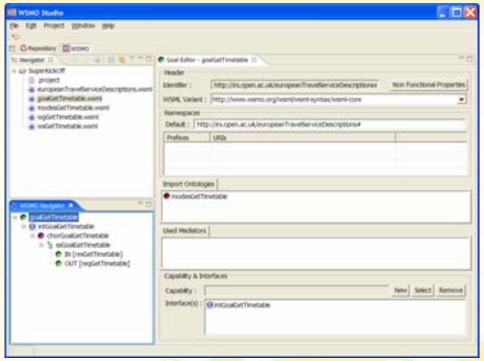
- Integrated Service Environment for WSMO
- Provide easy to use GUI for various WSMO tasks
  - Working with ontologies
  - Creating WSMO descriptions: goals, services, mediators
  - Creating WSMO centric orchestration and choreography specifications
  - Import (export) from (to) various formats
  - Front-end for ontology and service repositories
  - Front-end for runtime SWS environments (WSMX, IRS-III)
- http://www.wsmostudio.org



WSMO Studio

- Java based implementation
- Open Source core
  - LGPL
  - 3rd party contributors are free to choose their respective licensing terms
- Modular design
  - an Eclipse based plug-in architecture
- Extensible
  - 3rd parties may contribute new functionality (plug-ins) or modify existing functionality

Editing a Goal in WSMO Studio



WSMO Studio
File Edit Project Window Help
Repository WSMO
Navigator
Goal Editor - goalGetTimetable
WSMO Navigator
New
Select
Remove
Non Functional Properties

- SuperKickOff
  - .project
  - europeanTravelServiceDescriptions.wsml
  - goalGetTimetable.wsml
  - modesGetTimetable.wsml
  - wgGetTimetable.wsml
  - wsGetTimetable.wsml

- goalGetTimetable
  - intGoalGetTimetable
    - chorGoalGetTimetable
      - ssGoalGetTimetable
        - IN [resGetTimetable]
        - OUT [reqGetTimetable]

Header

Identifier : http://irs.open.ac.uk/europeanTravelServiceDescriptions#

WSML Variant : http://www.wsmo.org/wsml/wsml-syntax/wsml-core

Namespaces

Default : http://irs.open.ac.uk/europeanTravelServiceDescriptions#

Prefixes	URIs

Import Ontologies

modesGetTimetable

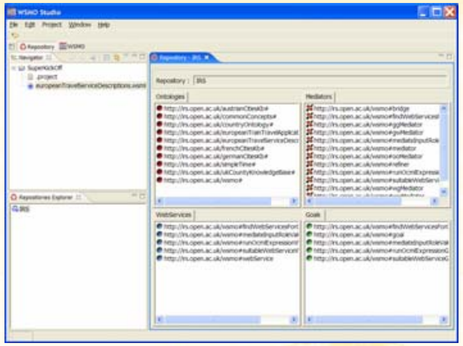
Used Mediators

Capability & Interfaces

Capability :

Interface(s) : intGoalGetTimetable

WSMO Studio view onto IRS-III



- SuperKickOff
  - .project
  - europeanTravelServiceDescriptions.wsml

Repository : IRS

Ontologies

- http://irs.open.ac.uk/austrianCitiesKb#
- http://irs.open.ac.uk/commonConcepts#
- http://irs.open.ac.uk/countryOntology#
- http://irs.open.ac.uk/europeanTrainTravelApplicat
- http://irs.open.ac.uk/europeanTravelServiceDesc
- http://irs.open.ac.uk/frenchCitiesKb#
- http://irs.open.ac.uk/germanCitiesKb#
- http://irs.open.ac.uk/simpleTime#
- http://irs.open.ac.uk/ukCountyKnowledgeBase#
- http://irs.open.ac.uk/wsmo#

Mediators

- http://irs.open.ac.uk/wsmo#bridge
- http://irs.open.ac.uk/wsmo#findWebServices
- http://irs.open.ac.uk/wsmo#ggMediator
- http://irs.open.ac.uk/wsmo#gwMediator
- http://irs.open.ac.uk/wsmo#mediateInputRole
- http://irs.open.ac.uk/wsmo#mediator
- http://irs.open.ac.uk/wsmo#ooMediator
- http://irs.open.ac.uk/wsmo#refiner
- http://irs.open.ac.uk/wsmo#runOcmlExpress
- http://irs.open.ac.uk/wsmo#suitableWebServi
- http://irs.open.ac.uk/wsmo#wgMediator

WebServices

- http://irs.open.ac.uk/wsmo#findWebServicesFor
- http://irs.open.ac.uk/wsmo#mediateInputRoleVal
- http://irs.open.ac.uk/wsmo#runOcmlExpressionV
- http://irs.open.ac.uk/wsmo#suitableWebServiceV
- http://irs.open.ac.uk/wsmo#webService

Goals

- http://irs.open.ac.uk/wsmo#findWebServicesFor
- http://irs.open.ac.uk/wsmo#goal
- http://irs.open.ac.uk/wsmo#mediateInputRoleVal
- http://irs.open.ac.uk/wsmo#runOcmlExpressionG
- http://irs.open.ac.uk/wsmo#suitableWebServiceG

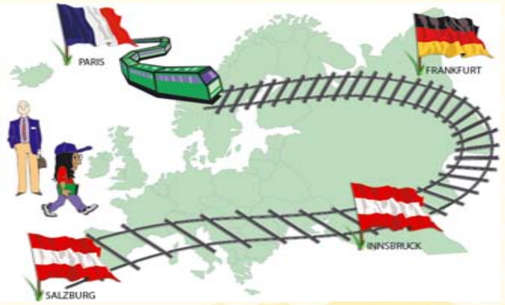
IRS ω Internet Reasoning Service

KNOWLEDGE MEDIA KMi INSTITUTE

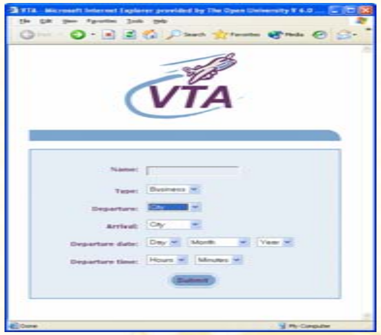
Hands-On Session with **IRS III**

Liliana Cabral
John Domingue

European Travel Scenario



European Travel Demo


VTA - Microsoft Internet Explorer provided by The Open University V 6.0
File Edit View Favorites Tools Help
Search Favorites Media
VTA

Name:

Type: Business

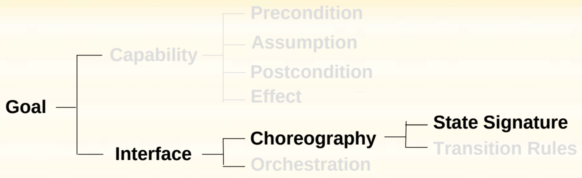
Departure: City

Arrival: City

Departure date: Day Month Year

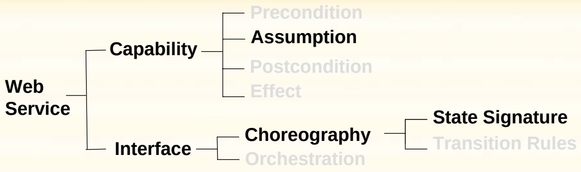
Departure time: Hours Minutes

Goal Description in Tutorial



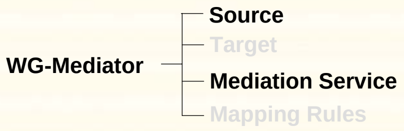
- **The steps that go into describing a goal in the tutorial are**:
  - Ontological description of the communications (request and response);
  - Creation of a goal;
  - Attachment of a choreography; Attachment of a state signature
  - Attachment of communications to state signature:
    - request as OUT mode; response as IN mode

Web Service Description in Tutorial



- **The steps that go into describing a service in the tutorial are:**
  - Ontological description of the communications (may be reused from goal);
  - Creation of a service; possibly attachment of an assumption
  - Attach a used-mediator (wg-mediator);
  - Attachment of a choreography; Attachment of a state signature
  - Attachment of communications to state signature
    - request as IN mode, grounded to LISP function;
    - response as OUT mode

Mediator Description in Tutorial

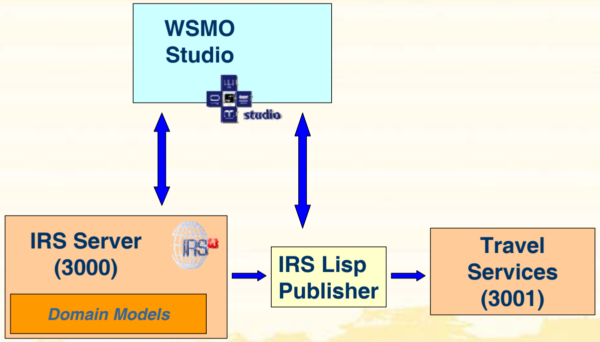


- **The steps that go into describing a mediator in the tutorial are**:
  - Creation of a wg-mediator (possibly involving a mediation service);
    - The mediation service is another SWS (goal, mediator, and ws descriptions)
  - Attachment of a source (the goal defined before)

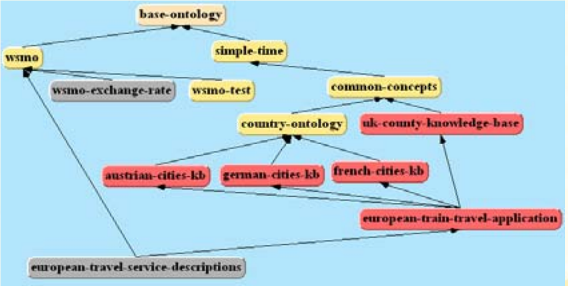
IRS-III Hands On Task

- Develop the SWS for an application for the European Travel scenario. The application should support a person booking a train ticket between 2 European cities at a specific time and date
- The following **WSMO Studio** tasks are involved:
  - Retrieve domain ontologies from IRS;
  - Create WSML ontology concepts to describe communications;
  - Create WSMO descriptions for Goals, WG-mediators and Web service descriptions;
  - Export these definitions to the IRS;
  - Create WSML ontology instances of the requests;
  - Achieve the goals against these instances.

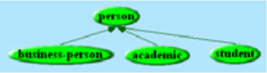
Tutorial Setup



Travel Related Knowledge Models

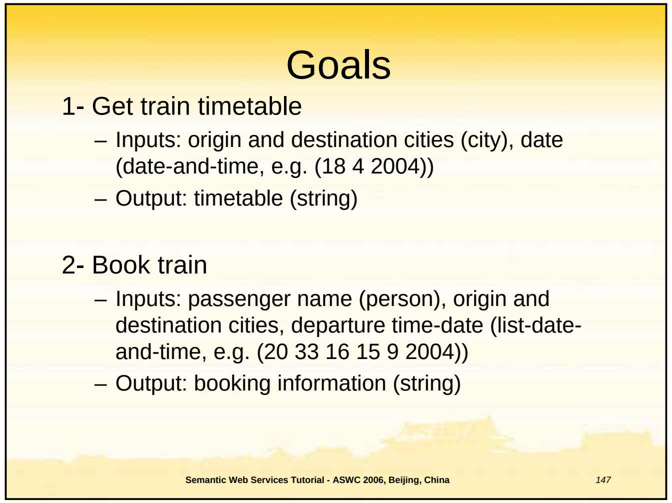


Key Classes, Relations, Instances



Is-in-country <city> <country> e.g.
(is-in-country berlin germany) -> true

(student <person>) -> true, for john matt michal
(business-person <person>) -> true, for liliana michael



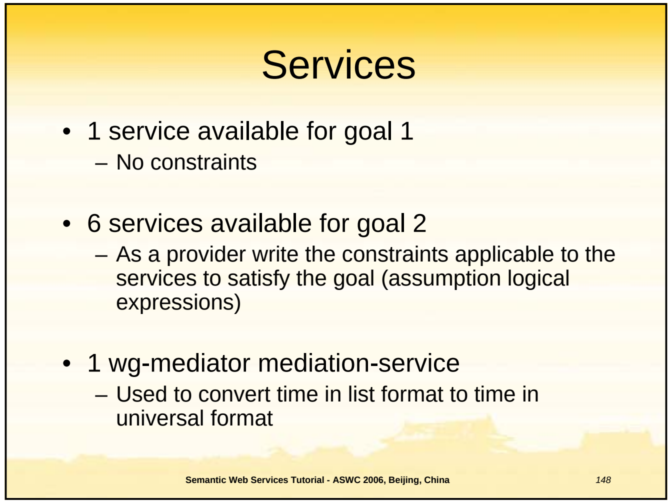
Goals

1- Get train timetable

- Inputs: origin and destination cities (city), date (date-and-time, e.g. (18 4 2004))
- Output: timetable (string)

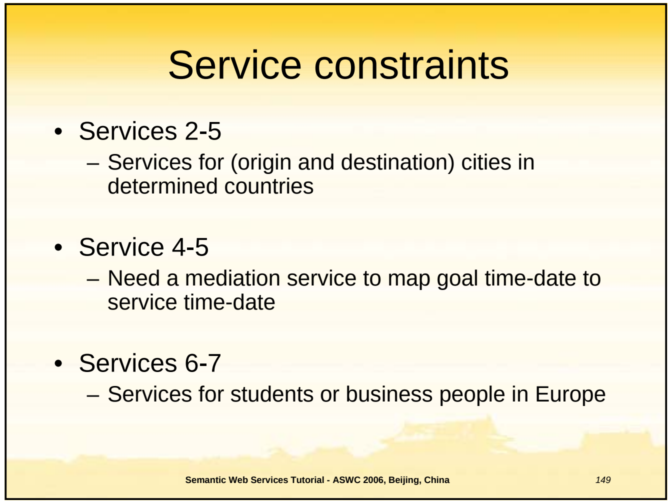
2- Book train

- Inputs: passenger name (person), origin and destination cities, departure time-date (list-date-and-time, e.g. (20 33 16 15 9 2004))
- Output: booking information (string)



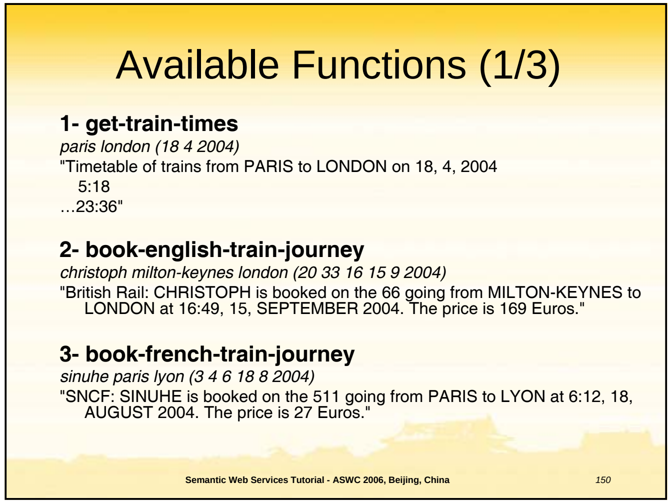
Services

- 1 service available for goal 1
  - No constraints
- 6 services available for goal 2
  - As a provider write the constraints applicable to the services to satisfy the goal (assumption logical expressions)
- 1 wg-mediator mediation-service
  - Used to convert time in list format to time in universal format



Service constraints

- Services 2-5
  - Services for (origin and destination) cities in determined countries
- Service 4-5
  - Need a mediation service to map goal time-date to service time-date
- Services 6-7
  - Services for students or business people in Europe



Available Functions (1/3)

1- get-train-times

paris london (18 4 2004)

"Timetable of trains from PARIS to LONDON on 18, 4, 2004
5:18
…23:36"

2- book-english-train-journey

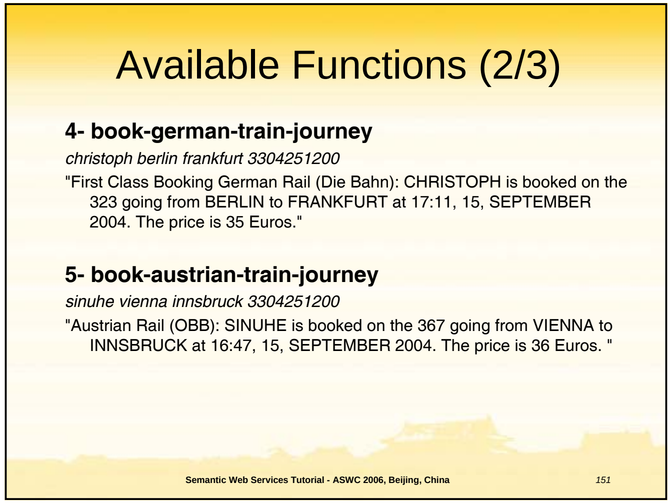
christoph milton-keynes london (20 33 16 15 9 2004)

"British Rail: CHRISTOPH is booked on the 66 going from MILTON-KEYNES to LONDON at 16:49, 15, SEPTEMBER 2004. The price is 169 Euros."

3- book-french-train-journey

sinuhe paris lyon (3 4 6 18 8 2004)

"SNCF: SINUHE is booked on the 511 going from PARIS to LYON at 6:12, 18, AUGUST 2004. The price is 27 Euros."



Available Functions (2/3)

4- book-german-train-journey

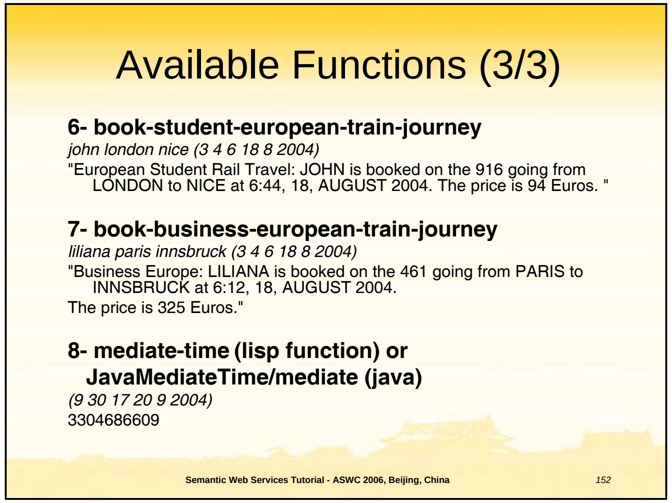
christoph berlin frankfurt 3304251200

"First Class Booking German Rail (Die Bahn): CHRISTOPH is booked on the 323 going from BERLIN to FRANKFURT at 17:11, 15, SEPTEMBER 2004. The price is 35 Euros."

5- book-austrian-train-journey

sinuhe vienna innsbruck 3304251200

"Austrian Rail (OBB): SINUHE is booked on the 367 going from VIENNA to INNSBRUCK at 16:47, 15, SEPTEMBER 2004. The price is 36 Euros. "



Available Functions (3/3)

6- book-student-european-train-journey

john london nice (3 4 6 18 8 2004)

"European Student Rail Travel: JOHN is booked on the 916 going from LONDON to NICE at 6:44, 18, AUGUST 2004. The price is 94 Euros. "

7- book-business-european-train-journey

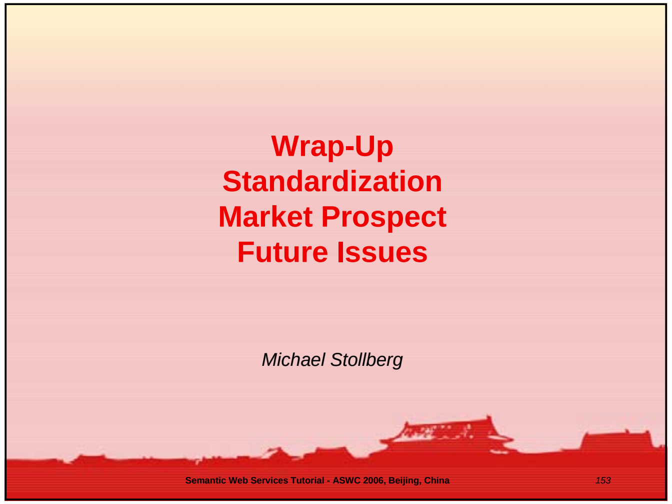
liliana paris innsbruck (3 4 6 18 8 2004)

"Business Europe: LILIANA is booked on the 461 going from PARIS to INNSBRUCK at 6:12, 18, AUGUST 2004.
The price is 325 Euros."

8- mediate-time (lisp function) or JavaMediateTime/mediate (java)

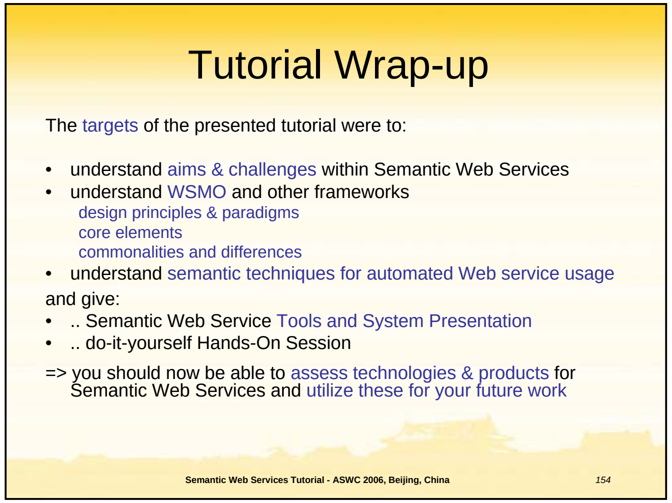
(9 30 17 20 9 2004)

3304686609



Wrap-Up
Standardization
Market Prospect
Future Issues

Michael Stollberg



Tutorial Wrap-up

The targets of the presented tutorial were to:

- understand aims & challenges within Semantic Web Services
- understand WSMO and other frameworks
  - design principles & paradigms
  - core elements
  - commonalities and differences
- understand semantic techniques for automated Web service usage

and give:

- .. Semantic Web Service Tools and System Presentation
- .. do-it-yourself Hands-On Session

=> you should now be able to assess technologies & products for Semantic Web Services and utilize these for your future work

History I

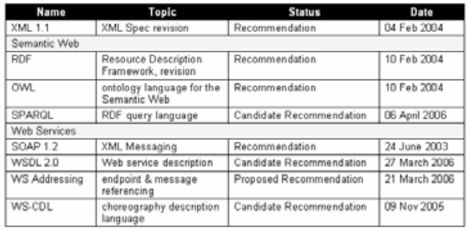
- late 90s: TBL wants the Internet to develop further
  - HTML is unstructured => not processable by machines
  - New kinds of Web Technologies needed
  - => *"turn the internet from a world-wide information repository for human consumption into a device of world-wide distributed computation"* (Fensel & Bussler, WSMF)
- American Scientific Article "The Semantic Web"
  - Pete & Lucy: a future example
  - Core Technologies:
    - **Ontologies:** unambiguous terminology definition in machine-readable format ("Semantics")
    - **Web Services:** functionality evocable over the Internet, re-usable and combinable distributed software components
    - **Agents:** electronic representatives that perform tasks on behalf of his owner
- Rising attention in Research & Industry ..

History II

- 1999: first W3C Recommendations
  - Specifications of XML Technologies (XSL, XTL,…)
  - Semantic Web Layer Cake
  - Languages: XML, RDF
- 2000 – 2001: first R&D-activities
  - 1. Web Service Technology Specifications: SOAP, WSDL, UDDI
  - related research areas become interested (AI / Knowledge Engineering; distributed computing, etc.), first projects: DAML (US), OnToKnowledge, etc.
  - "1st Semantic Web Working Symposium", Stanford (USA), ca. 100 participants
- 2002 – 2003: research & industry sets off
  - SDK-Cluster (Europe), DAML efforts (USA)
  - initial research results, still very chaotic / without a "framework"
  - industrial efforts on Web services
  - ISWC 02 / 03: double number of participants each year
- 2004 ff: the hot phase
  - W3C recommendations (OWL, XML + RDF revisions, others)
  - first set of research & development results
  - rising industrial & commercial attention

Standardization Efforts W3C

- 1st set of recommendations in 1999 / 2000, currently revised



Name	Topic	Status	Date
XML 1.1	XML Spec revision	Recommendation	04 Feb 2004
Semantic Web			
RDF	Resource Description Framework, revision	Recommendation	10 Feb 2004
OWL	ontology language for the Semantic Web	Recommendation	10 Feb 2004
SPARQL	RDF query language	Candidate Recommendation	06 April 2006
Web Services			
SOAP 1.2	XML Messaging	Recommendation	24 June 2003
WSDL 2.0	Web service description	Candidate Recommendation	27 March 2006
WS Addressing	endpoint & message referencing	Proposed Recommendation	21 March 2006
WS-CDL	choreography description language	Candidate Recommendation	09 Nov 2005

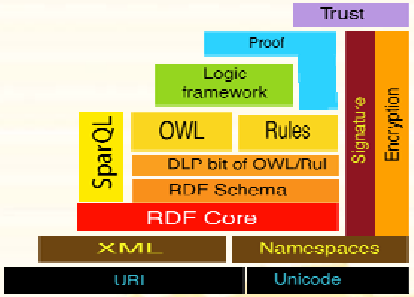
- Semantic Web Services
  - Member Submissions: OWL-S, WSMO, SWSF, WSDL-S
  - Working Groups:
    - Semantic Web Service Interest Group
    - Semantic Annotations for WSDL Group

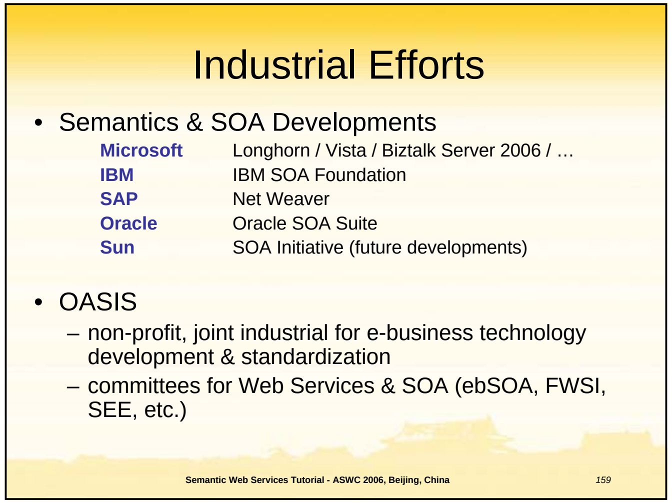
=> standardization need acknowledged, but no agreement yet on what & how

Layer Cake - Revised

W3C Semantic Web Language Layer Cake

revised version, Tim-Berners-Lee 2005



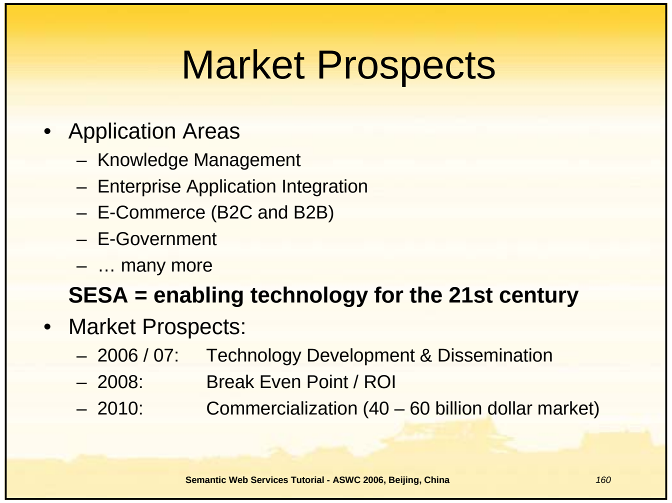


Industrial Efforts

- Semantics & SOA Developments

Microsoft	Longhorn / Vista / Biztalk Server 2006 / …
IBM	IBM SOA Foundation
SAP	Net Weaver
Oracle	Oracle SOA Suite
Sun	SOA Initiative (future developments)

- OASIS
  - non-profit, joint industrial for e-business technology development & standardization
  - committees for Web Services & SOA (ebSOA, FWSI, SEE, etc.)

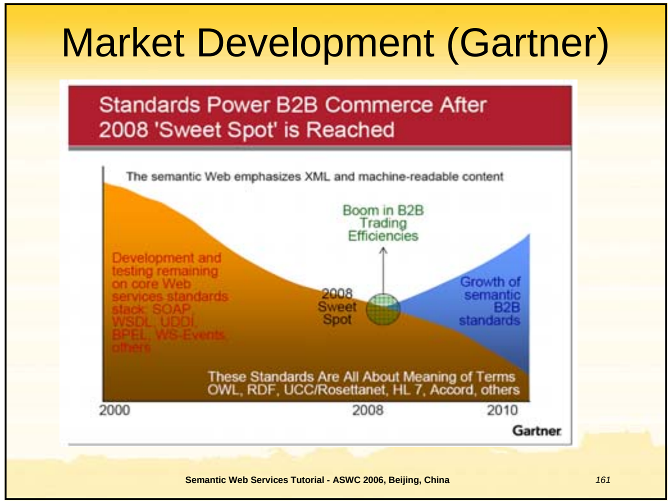


Market Prospects

- Application Areas
  - Knowledge Management
  - Enterprise Application Integration
  - E-Commerce (B2C and B2B)
  - E-Government
  - … many more

SESA = enabling technology for the 21st century

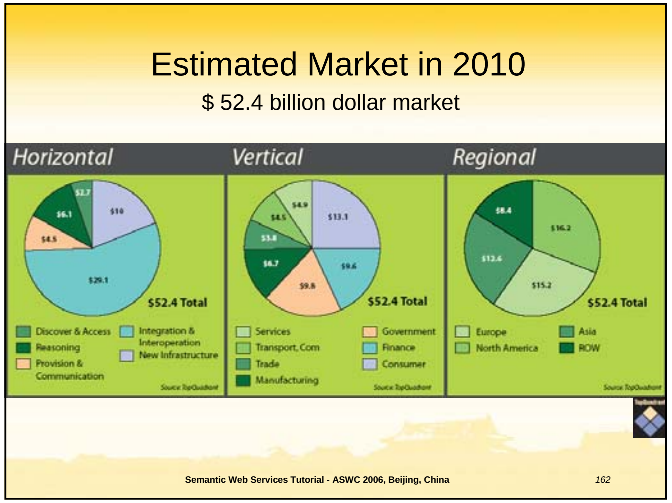
- Market Prospects:
  - 2006 / 07: Technology Development & Dissemination
  - 2008: Break Even Point / ROI
  - 2010: Commercialization (40 – 60 billion dollar market)



Market Development (Gartner)

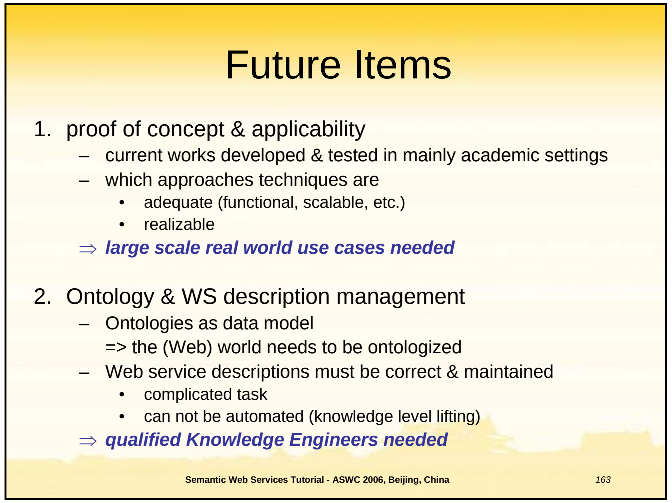
Standards Power B2B Commerce After 2008 'Sweet Spot' is Reached

The semantic Web emphasizes XML and machine-readable content



Estimated Market in 2010

$ 52.4 billion dollar market



Future Items

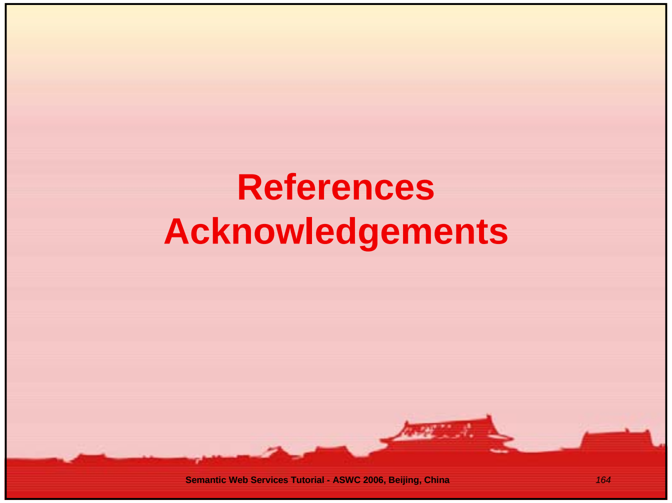
1. proof of concept & applicability
   - current works developed & tested in mainly academic settings
   - which approaches techniques are
     - adequate (functional, scalable, etc.)
     - realizable

   ⇒ ***large scale real world use cases needed***

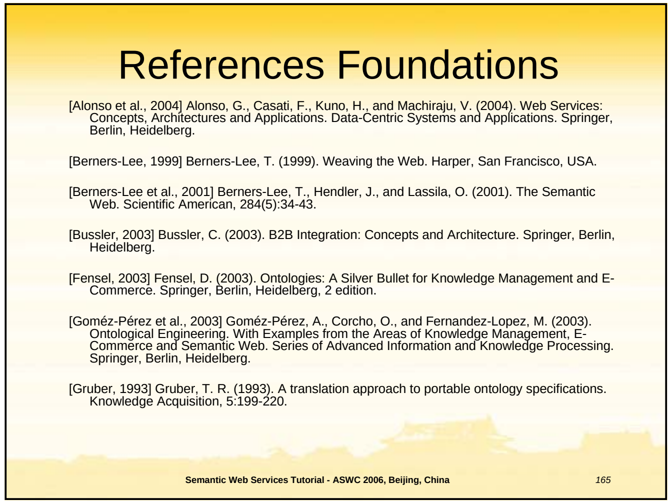
2. Ontology & WS description management
   - Ontologies as data model

     => the (Web) world needs to be ontologized
   - Web service descriptions must be correct & maintained
     - complicated task
     - can not be automated (knowledge level lifting)

   ⇒ ***qualified Knowledge Engineers needed***



References
Acknowledgements



References Foundations

[Alonso et al., 2004] Alonso, G., Casati, F., Kuno, H., and Machiraju, V. (2004). Web Services: Concepts, Architectures and Applications. Data-Centric Systems and Applications. Springer, Berlin, Heidelberg.

[Berners-Lee, 1999] Berners-Lee, T. (1999). Weaving the Web. Harper, San Francisco, USA.

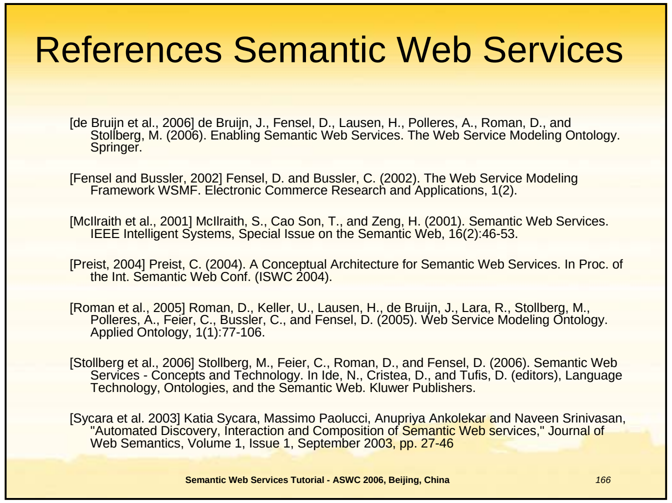
[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. Scientific American, 284(5):34-43.

[Bussler, 2003] Bussler, C. (2003). B2B Integration: Concepts and Architecture. Springer, Berlin, Heidelberg.

[Fensel, 2003] Fensel, D. (2003). Ontologies: A Silver Bullet for Knowledge Management and E-Commerce. Springer, Berlin, Heidelberg, 2 edition.

[Goméz-Pérez et al., 2003] Goméz-Pérez, A., Corcho, O., and Fernandez-Lopez, M. (2003). Ontological Engineering. With Examples from the Areas of Knowledge Management, E-Commerce and Semantic Web. Series of Advanced Information and Knowledge Processing. Springer, Berlin, Heidelberg.

[Gruber, 1993] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge Acquisition, 5:199-220.



References Semantic Web Services

[de Bruijn et al., 2006] de Bruijn, J., Fensel, D., Lausen, H., Polleres, A., Roman, D., and Stollberg, M. (2006). Enabling Semantic Web Services. The Web Service Modeling Ontology. Springer.

[Fensel and Bussler, 2002] Fensel, D. and Bussler, C. (2002). The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications, 1(2).

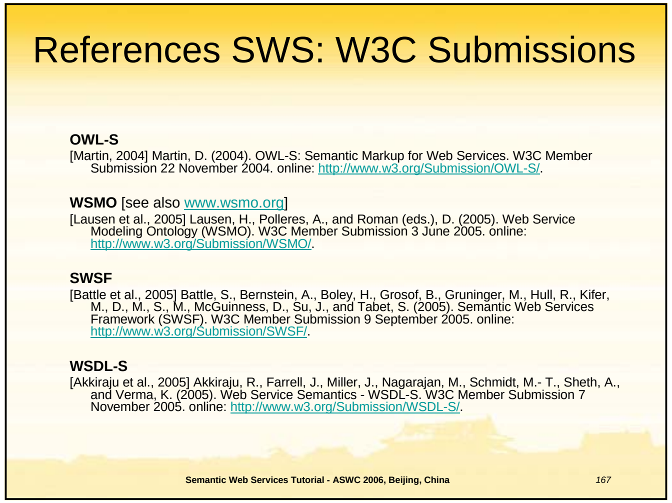
[McIlraith et al., 2001] McIlraith, S., Cao Son, T., and Zeng, H. (2001). Semantic Web Services. IEEE Intelligent Systems, Special Issue on the Semantic Web, 16(2):46-53.

[Preist, 2004] Preist, C. (2004). A Conceptual Architecture for Semantic Web Services. In Proc. of the Int. Semantic Web Conf. (ISWC 2004).

[Roman et al., 2005] Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C., and Fensel, D. (2005). Web Service Modeling Ontology. Applied Ontology, 1(1):77-106.

[Stollberg et al., 2006] Stollberg, M., Feier, C., Roman, D., and Fensel, D. (2006). Semantic Web Services - Concepts and Technology. In Ide, N., Cristea, D., and Tufis, D. (editors), Language Technology, Ontologies, and the Semantic Web. Kluwer Publishers.

[Sycara et al. 2003] Katia Sycara, Massimo Paolucci, Anupriya Ankolekar and Naveen Srinivasan, "Automated Discovery, Interaction and Composition of Semantic Web services," Journal of Web Semantics, Volume 1, Issue 1, September 2003, pp. 27-46



References SWS: W3C Submissions

OWL-S

[Martin, 2004] Martin, D. (2004). OWL-S: Semantic Markup for Web Services. W3C Member Submission 22 November 2004. online: http://www.w3.org/Submission/OWL-S/.

WSMO [see also www.wsmo.org]

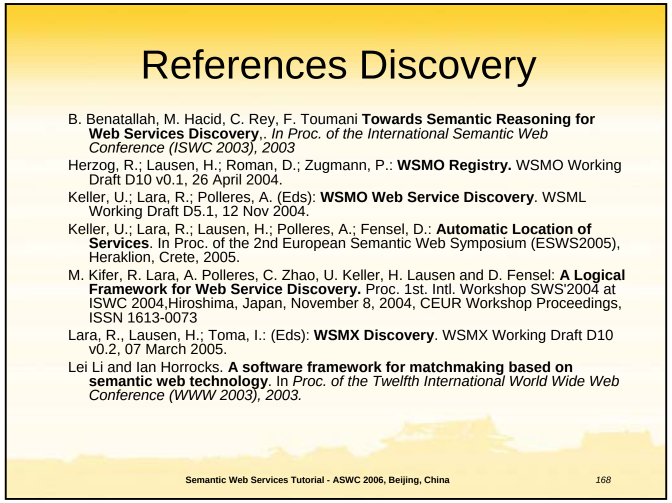
[Lausen et al., 2005] Lausen, H., Polleres, A., and Roman (eds.), D. (2005). Web Service Modeling Ontology (WSMO). W3C Member Submission 3 June 2005. online: http://www.w3.org/Submission/WSMO/.

SWSF

[Battle et al., 2005] Battle, S., Bernstein, A., Boley, H., Grosof, B., Gruninger, M., Hull, R., Kifer, M., D., M., S., M., McGuinness, D., Su, J., and Tabet, S. (2005). Semantic Web Services Framework (SWSF). W3C Member Submission 9 September 2005. online: http://www.w3.org/Submission/SWSF/.

WSDL-S

[Akkiraju et al., 2005] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.- T., Sheth, A., and Verma, K. (2005). Web Service Semantics - WSDL-S. W3C Member Submission 7 November 2005. online: http://www.w3.org/Submission/WSDL-S/.



References Discovery

B. Benatallah, M. Hacid, C. Rey, F. Toumani **Towards Semantic Reasoning for Web Services Discovery**,. *In Proc. of the International Semantic Web Conference (ISWC 2003), 2003*

Herzog, R.; Lausen, H.; Roman, D.; Zugmann, P.: **WSMO Registry.** WSMO Working Draft D10 v0.1, 26 April 2004.

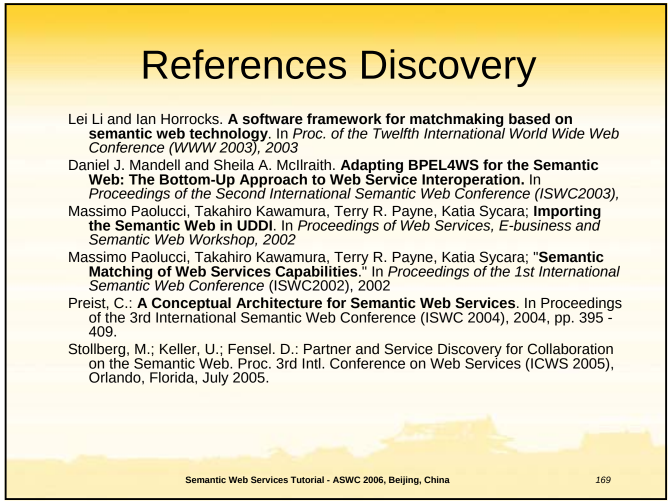
Keller, U.; Lara, R.; Polleres, A. (Eds): **WSMO Web Service Discovery**. WSML Working Draft D5.1, 12 Nov 2004.

Keller, U.; Lara, R.; Lausen, H.; Polleres, A.; Fensel, D.: **Automatic Location of Services**. In Proc. of the 2nd European Semantic Web Symposium (ESWS2005), Heraklion, Crete, 2005.

M. Kifer, R. Lara, A. Polleres, C. Zhao, U. Keller, H. Lausen and D. Fensel: **A Logical Framework for Web Service Discovery.** Proc. 1st. Intl. Workshop SWS'2004 at ISWC 2004,Hiroshima, Japan, November 8, 2004, CEUR Workshop Proceedings, ISSN 1613-0073

Lara, R., Lausen, H.; Toma, I.: (Eds): **WSMX Discovery**. WSMX Working Draft D10 v0.2, 07 March 2005.

Lei Li and Ian Horrocks. **A software framework for matchmaking based on semantic web technology**. In *Proc. of the Twelfth International World Wide Web Conference (WWW 2003), 2003.*



References Discovery

Lei Li and Ian Horrocks. **A software framework for matchmaking based on semantic web technology**. In *Proc. of the Twelfth International World Wide Web Conference (WWW 2003), 2003*

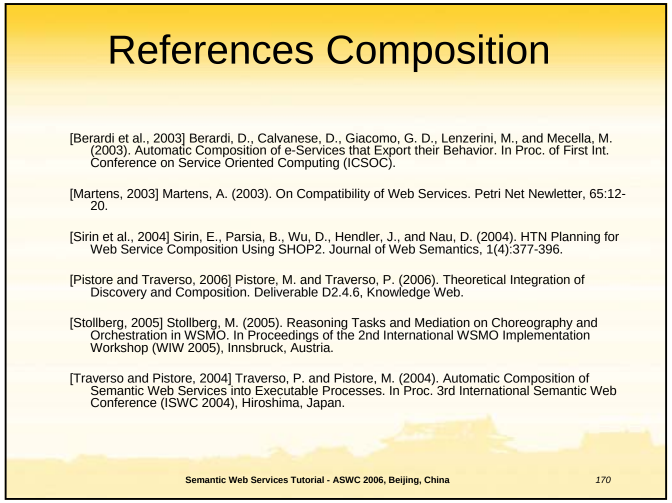
Daniel J. Mandell and Sheila A. McIlraith. **Adapting BPEL4WS for the Semantic Web: The Bottom-Up Approach to Web Service Interoperation.** In *Proceedings of the Second International Semantic Web Conference (ISWC2003),*

Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, Katia Sycara; **Importing the Semantic Web in UDDI**. In *Proceedings of Web Services, E-business and Semantic Web Workshop, 2002*

Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, Katia Sycara; "**Semantic Matching of Web Services Capabilities**." In *Proceedings of the 1st International Semantic Web Conference* (ISWC2002), 2002

Preist, C.: **A Conceptual Architecture for Semantic Web Services**. In Proceedings of the 3rd International Semantic Web Conference (ISWC 2004), 2004, pp. 395 - 409.

Stollberg, M.; Keller, U.; Fensel. D.: Partner and Service Discovery for Collaboration on the Semantic Web. Proc. 3rd Intl. Conference on Web Services (ICWS 2005), Orlando, Florida, July 2005.



References Composition

[Berardi et al., 2003] Berardi, D., Calvanese, D., Giacomo, G. D., Lenzerini, M., and Mecella, M. (2003). Automatic Composition of e-Services that Export their Behavior. In Proc. of First Int. Conference on Service Oriented Computing (ICSOC).

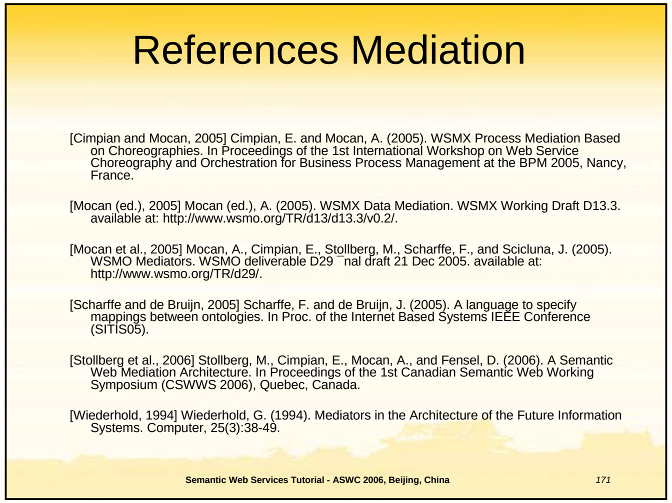
[Martens, 2003] Martens, A. (2003). On Compatibility of Web Services. Petri Net Newletter, 65:12-20.

[Sirin et al., 2004] Sirin, E., Parsia, B., Wu, D., Hendler, J., and Nau, D. (2004). HTN Planning for Web Service Composition Using SHOP2. Journal of Web Semantics, 1(4):377-396.

[Pistore and Traverso, 2006] Pistore, M. and Traverso, P. (2006). Theoretical Integration of Discovery and Composition. Deliverable D2.4.6, Knowledge Web.

[Stollberg, 2005] Stollberg, M. (2005). Reasoning Tasks and Mediation on Choreography and Orchestration in WSMO. In Proceedings of the 2nd International WSMO Implementation Workshop (WIW 2005), Innsbruck, Austria.

[Traverso and Pistore, 2004] Traverso, P. and Pistore, M. (2004). Automatic Composition of Semantic Web Services into Executable Processes. In Proc. 3rd International Semantic Web Conference (ISWC 2004), Hiroshima, Japan.



References Mediation

[Cimpian and Mocan, 2005] Cimpian, E. and Mocan, A. (2005). WSMX Process Mediation Based on Choreographies. In Proceedings of the 1st International Workshop on Web Service Choreography and Orchestration for Business Process Management at the BPM 2005, Nancy, France.

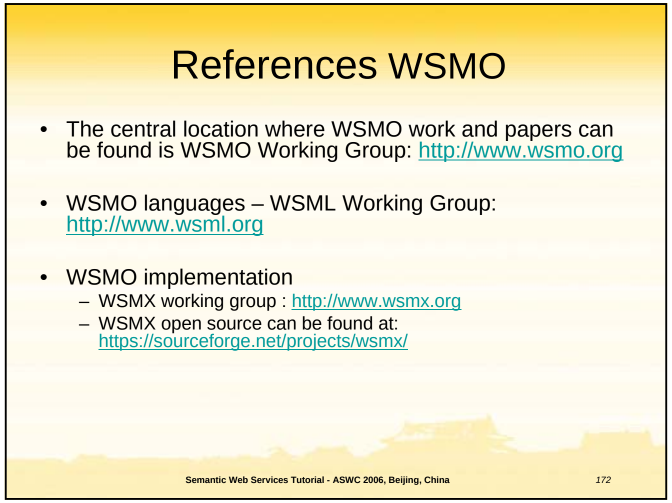
[Mocan (ed.), 2005] Mocan (ed.), A. (2005). WSMX Data Mediation. WSMX Working Draft D13.3. available at: http://www.wsmo.org/TR/d13/d13.3/v0.2/.

[Mocan et al., 2005] Mocan, A., Cimpian, E., Stollberg, M., Scharffe, F., and Scicluna, J. (2005). WSMO Mediators. WSMO deliverable D29 final draft 21 Dec 2005. available at: http://www.wsmo.org/TR/d29/.

[Scharffe and de Bruijn, 2005] Scharffe, F. and de Bruijn, J. (2005). A language to specify mappings between ontologies. In Proc. of the Internet Based Systems IEEE Conference (SITIS05).

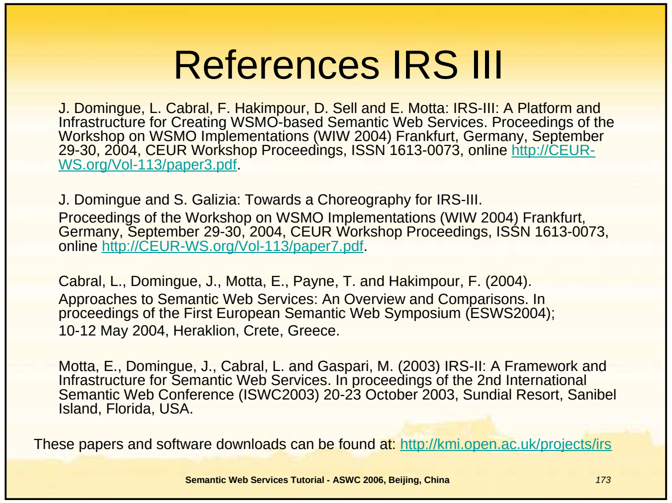
[Stollberg et al., 2006] Stollberg, M., Cimpian, E., Mocan, A., and Fensel, D. (2006). A Semantic Web Mediation Architecture. In Proceedings of the 1st Canadian Semantic Web Working Symposium (CSWWS 2006), Quebec, Canada.

[Wiederhold, 1994] Wiederhold, G. (1994). Mediators in the Architecture of the Future Information Systems. Computer, 25(3):38-49.



References WSMO

- The central location where WSMO work and papers can be found is WSMO Working Group: http://www.wsmo.org
- WSMO languages – WSML Working Group: http://www.wsml.org
- WSMO implementation
  - WSMX working group : http://www.wsmx.org
  - WSMX open source can be found at: https://sourceforge.net/projects/wsmx/



References IRS III

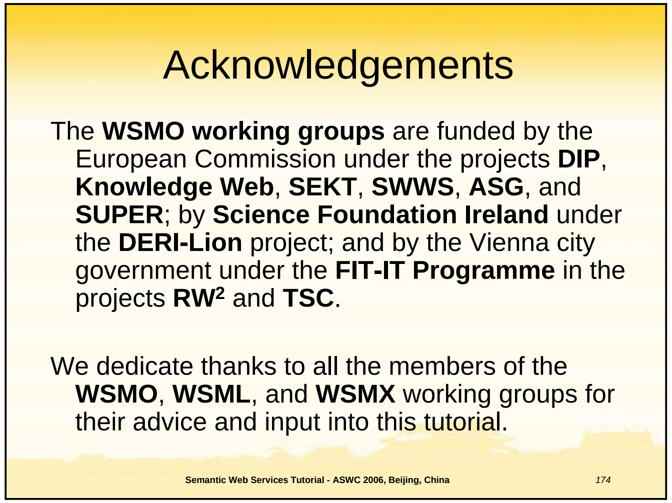
J. Domingue, L. Cabral, F. Hakimpour, D. Sell and E. Motta: IRS-III: A Platform and Infrastructure for Creating WSMO-based Semantic Web Services. Proceedings of the Workshop on WSMO Implementations (WIW 2004) Frankfurt, Germany, September 29-30, 2004, CEUR Workshop Proceedings, ISSN 1613-0073, online http://CEUR-WS.org/Vol-113/paper3.pdf.

J. Domingue and S. Galizia: Towards a Choreography for IRS-III.
Proceedings of the Workshop on WSMO Implementations (WIW 2004) Frankfurt, Germany, September 29-30, 2004, CEUR Workshop Proceedings, ISSN 1613-0073, online http://CEUR-WS.org/Vol-113/paper7.pdf.

Cabral, L., Domingue, J., Motta, E., Payne, T. and Hakimpour, F. (2004).
Approaches to Semantic Web Services: An Overview and Comparisons. In proceedings of the First European Semantic Web Symposium (ESWS2004);
10-12 May 2004, Heraklion, Crete, Greece.

Motta, E., Domingue, J., Cabral, L. and Gaspari, M. (2003) IRS-II: A Framework and Infrastructure for Semantic Web Services. In proceedings of the 2nd International Semantic Web Conference (ISWC2003) 20-23 October 2003, Sundial Resort, Sanibel Island, Florida, USA.

These papers and software downloads can be found at: http://kmi.open.ac.uk/projects/irs



Acknowledgements

The **WSMO working groups** are funded by the European Commission under the projects **DIP**, **Knowledge Web**, **SEKT**, **SWWS**, **ASG**, and **SUPER**; by **Science Foundation Ireland** under the **DERI-Lion** project; and by the Vienna city government under the **FIT-IT Programme** in the projects **RW^2** and **TSC**.

We dedicate thanks to all the members of the **WSMO**, **WSML**, and **WSMX** working groups for their advice and input into this tutorial.