Semantic Web Services for Business Processes Management

ICIW 2007, May 13, Mauritius
Presenters
- Carlos Pedrinaci (KMI)
- Sebastian Stein (IDS Scheer)
- Michael Stollberg (DERI Austria)
Acknowledgement & Copyrights

• This material is based upon works supported by the EU under the SUPER project (FP6 - 026850)

• Material Preparation
  – KMI: John Domingue, Carlos Pedrinaci, Barry Norton
  – Poznan University: Agata Filipowska
  – IAAS, University of Stuttgart: Dimka Karastoyanova, Jörg Nitzsche, Tammo van Lessen, Zhilei Ma, Frank Leymann
  – IDS Scheer: Sebastian Stein
  – DERI Austria: Dumitru Roman, Michael Stollberg
  – DERI Ireland: Maciej Zaremba

© by the SUPER project consortium
1. Introduction: The Need of Semantics in BPM

2. Business Process Management
   ■ Introduction
   ■ BPEL

3. Semantic Web Services
   ■ Introduction
   ■ SWS Technologies

4. Integration: The SUPER Approach
Querying the Process Space

“In which of our food manufacturing machines are we processing meat or raw eggs?”

“How many inventory management methods are currently in use?”

“Do we have a cost approval process for items below $200?”
The Critical IT / Process Divide

Business Experts’ Perspective: Processes

Querying the Process Space

Manual Labor

Process Implementation

IT Implementation Perspective
The Critical IT / Process Divide

Business Experts’ Perspective: Processes

- Machine-Accessible Representation of Business Experts’ Requirements

SCOPE of SUPER

Mechanized Mediation based on Machine Reasoning

- Machine-Accessible Representation of Processes, Process Fragments, and IT Infrastructure as Semantic Web Services

IT Implementation Perspective

Querying the Process Space → Implementation Process
What Are My Services?

Here is my business process! I think this solves my business problem nicely...

A\textsuperscript{1} \rightarrow A\textsuperscript{2} \rightarrow A\textsuperscript{3} \rightarrow A\textsuperscript{4}

Nice try, but it won’t run... You need to specify the services that perform each step!
What Are My Services?

I don’t understand about these technical details! This is my view on the process...

A¹ → A² → A³ → A⁴

O.k. no problem, I will help you...
What are my services?

This is cumbersome! Why do I always need IT staff to solve my business problems?

It takes too long to get these folks, they use different terminology than I do...

I am happy to describe what the activities do in my terms. Can the system be smart enough and find the right services itself???
Matching Activities and Port Types Based on Semantics

Semantic Web Services
Supporting Business Users Better

Why do I have to draw everything?
Why do I have to use “expressions” and that technical stuff at all?
Why isn’t my description sufficient?
Matching Model Representations & Semantics

Here is my business process!

Business Representation

A → B → C

The reason of the order must be known in time.

Wow! This is perfect – nothing left to do for me!

IT Representation

α → β → χ

© SUPER 5/8/2007

Business Process Management

Introduction

Sebastian Stein, IDS Scheer
BPM: Introduction

• BPM’s Parents and Definition
• Enterprise Model
• Business Process Lifecycle
• BPM Applications
• Summary

© Sebastian Stein
**BPM’s Parents and Definition**

- **Business Process Management (BPM)**
- **Continuous Improvements** (since 1990)
- **Business Reengineering** (since 1990)
- **EAI** (since 1990)
- **Workflow Systems** (since 1985)
- **CSCW / Groupware / Workgroup Systems**
- **Office Automation** (since 1980)
- **SOA** (since 2000)
- **Business Process Mngt.** (since 2000)
- **Business Objects** (since 2000)

© Sebastian Stein
BPM’s Parents and Definition

• there are several competing definitions
• own focus coins BPM definition:
  – focus on documentation
  – focus on process and execution
  – focus on IT architecture
  – focus on costs and risks
  – focus on business strategy
  – etc.
• in SUPER we have a strong process and execution focus

© Sebastian Stein
Enterprise Model

- model of an enterprise
  - internal enterprise architecture
  - internal requirements
  - interfaces
  - business processes
  - external integration
  - external requirements
  - ...

- model is an abstraction of reality
- used by many different stakeholders
  - views needed
  - abstraction levels needed
  - lifecycle concept needed
  - different languages, notations and formalisms needed

© Sebastian Stein
Enterprise Model

Enterprise Model

Customer

License

Sales Department

License Service

License Available

Get License

Get PurchaseOrder

Purchase Order Received

SYS

Purchase Order Extracted

Content Identifier

Cell Phone Interface Service

IT Department

Load Content

Content Identifier

Digital Content

Content Library Service

Content Provider

Content Ready for Download

Send Content & License

Cell Phone Interface Service

IT Department

Digital Content

License

Purchase Order Satisfied

Cell Phone

Customer Purchase Order

Purchase Order

SYS

Extractor

SYS

SYS

Content Identifier

Content Identifier

Content Identifier

Sales Department

License Service

License

Cell Phone Interface Service

Content Library Service

Content Provider

Content

Content

Content

Sales Department

License Service

License

Cell Phone Interface Service

Content Library Service

Content Provider

Content

Content

Sales Department

License Service

License

Cell Phone Interface Service

Content Library Service

Content Provider

Content

Content

© Sebastian Stein
too complex
Enterprise Model

• possible abstraction layers are:
  – requirements definition
  – design specification
  – implementation specification
  – execution and run-time models

• possible views are:
  – organisational view
  – product view
  – data view (information architecture)
  – function and IT view
  – process view

© Sebastian Stein
Enterprise Model

- many different frameworks for enterprise architecture, e.g.:
  - Zachman Framework (very comprehensive)
  - ArchiMate (simplified version of Zachman)
  - ARIS (promoted by IDS Scheer)
  - TOGAF (strong IT focus)
  - IAF (promoted by Cap Gemini)

- currently many discussions around process design & execution, e.g.:
  - BPMN (notation for (IT oriented) business processes)
  - EPC (notation for business processes)
  - Petrinets (formalism often used for workflow modelling)
  - UML Activity diagrams
  - XPDL (execution language for process definitions)
  - BPEL (execution language for process definitions)
  - XLANG (execution language promoted by Microsoft)
  - ...

© Sebastian Stein
Process Lifecycle

- enterprise model evolves → lifecycle
- based on general Deming cycle for continuous process improvements
- sometimes also named Shewhart cycle

1. Plan
2. Do
3. Check
4. Act

© Sebastian Stein
process lifecycle:

1. Analysis
   - gather requirements
   - document current state (as-is)

2. Design
   - document to-be
   - specify how to get there

3. Implementation/Execution
   - implement to-be in organisation and IT
   - change management

4. Control and Monitoring
   - monitor execution
   - measure outcome and analyse it

© Sebastian Stein
Business Process Management (BPM)

- define enterprise vision
- define enterprise strategy
- business process design
- enterprise architecture (e.g. Zachman)
- IT management (e.g. ITIL)
- quality management
- corporate performance management
- process automation
- application integration
- risk management
- cost control and management
- compliance management

© Sebastian Stein
Business Experts’ Perspective: Processes

Querying the Process Space

Manual Labor

IT Implementation Perspective

Process Implementation
Summary

• BPM definition depends on your focus
• Enterprise Model describes all relevant aspects of your enterprise
• different stakeholders will have different views and information needs
• lifecycle for the different parts of the Enterprise Model
• BPM is done for many different purposes, but SUPER focus on:
  – business process design
  – business process execution
  – monitoring and analysis of execution

© Sebastian Stein
BPM Applications

Business Experts’ Perspective: Processes

Querying the Process Space

IT Implementation Perspective

Process Implementation

Manual Labor
Automating Business Processes

- automatic support of business processes has been enabled by Workflow Technology
- separation of control flow and business activities
Dimensions in Workflow

• Workflows have at least 3 dimensions (Leymann, Roller [Production Workflow]):
  ► What? = control logic dimension → what task should be executed
  ► Who? = Organization dimension who must execute a task? → a role or person in an organization
  ► What with? = infrastructure dimension → what program or tool must be used to execute a task

• agreement on the number of dimensions was never reached
• there are multiple notations and languages for workflows

© Dimka Karastoyanova
Workflow Languages

- Industry has tried to get an agreement on a common workflow language since the early 90’s
- Now the industry agreed on BPEL *(that’s why it’s interesting)*
  - Portability
  - Interoperability

© Frank Leymann, Joerg Nitzsche
WS-Flows are workflows that use only Web Services (as participants)

Complete utilization of the advantages of Web Services - integration

WS-Flows have only 2 dimensions
- Control Logic
- Infrastructure = Web Services

No direct support for people/organizational dimension yet

But efforts towards involving people in BPEL already exist: BPEL4People
1. Web service technology provides a “virtual component model“ for using components in a loosely coupled manner

2. When using a Web service the supporting container hides its “middleware idiosyncrasies“
(component model behind the implementation of the Web service, the invocation protocol etc.)

3. Web service technology does not provide a new component model for implementing components
(well, except for BPEL 😊)
Virtualizing Components

Virtual Component

Web Service

Concrete Component

implements (E)JB

implements StP

implements Assembly

J2EE

DBMS

.NET

© Frank Leymann
Ingredients Of WSDL

Interface ("What")

Port Type

Supports

Port

Operation

Input, Output & Faults

Message

Endpoints ("Where")

Binding

Transport: Formats & Protocols

How to invoke (doc exchange, rpc)

How to encode (serialization)

Access Specification ("How")

Hosts & Implements

Implements

Provides

Service

© Frank Leymann
The Role of Bindings

Client

Port

Type

Port 1

Port 2

Port 3

Port 4

Port k

Binding

SOAP/HTTP

POJO/JMS

ASCII/SMTP

RDF/TSpace

© Frank Leymann, Joerg Nitzsche
WS-Interoperability

- WS-I (Web Services Interoperability)
  - Consortium to ensure interoperability
    - Specification of so-called profiles ("regulation" of how and what to use)
    - Tools (test, analysis, monitor)
    - Sample implementation

- WS-I profile(s) compliance ensures common behavior
  - Specifications sometimes "terse"
    - Interpreted differently by different vendors
    - WS-I provides common interpretation resolving ambiguities
  - Subsetting of features of a specification

→ Interoperability is the result

© Frank Leymann, Joerg Nitzsche
Processes, Web Services andBindings

- WS-Flows maintain the two-level programming approach
- And comply with the life cycle of Workflows
- BUT an additional phase in the life cycle has been introduced
  - Deployment
- During deployment one specifies binding information for partner services
- Improved configurability and reusability
- Criteria for dynamic binding can also be provided in the deployment descriptor

© Dimka Karastoyanova
What is BPEL?

- A language to specify behaviour of business processes
  - Between Web services...
  - ...and as Web service
- Executable processes
  - Can be performed within all compliant environments (portability)
  - Interoperability between heterogeneous environments
- Abstract processes
  - Specify constraints of message exchange
  - Are “views” on internal processes
- Combination of graph-based language (IBM WSFL) and calculus-based language (Microsoft XLANG)
Business Processes Compose Web Services

- BPEL Process uses Web services
A BPEL Process is also a Web Service

- Provides functionality in terms of WSDL port types and operations
Aggregating Web services

BPEL provides a recursive aggregation model for Web services

© Frank Leymann
BPEL Elements

- Partner Links specify the roles of all external partners involved in the process as well as the role(s) of the process itself
- Variables can be defined either in the process or in a scope
  - They are used as input- and output-containers of interaction activities as well as assign activities
- Correlation Sets are used to correlate messages that belong to the same process instance
- Handlers can be used to define exception handling and compensation
- **The activities define the actual control logic**
partnerLinkTypes

- **Bi-directional typed connector**
- A mutual call-back dependency
- Specifies one or two roles; a port type per role
- Messages exchanged between partners
- A promise to playing a role is equal to a contract

© Frank Leymann, Joerg Nitzsche, Dimka Karastoyanova
Communication and Control Flow

• Elements:
  – Interaction activities
    • Receive, Reply, Invoke, Pick
  – Complex activities for control flow
    • Sequence, Flow, If, While, RepeatUntil, ForEach
  – Data manipulation
    • Assign
  – Exception handling
    • Throw, Rethrow, Fault Handlers, Compensation Handlers
  – Reaction to Events
    • Pick, Event Handlers

• Instantiation is implicit – use <receive> or <pick>
  – With the “createInstance” attribute set to "yes"
  – To instantiate a process using Pick
    • The events in the <pick> MUST all be <onMessage> events
    • Pick reacts on one <onMessage> event

© Zhilei Ma
Communication: Synchronous invocation

```
<invoke name=... partnerLink=... operation=... inputVariable=... outputVariable=... />
```

```
<receive name=... partnerLink=... operation=... variable=... />
```

```
<reply name=... partnerLink=... operation=... variable=... />
```
Communication: Asynchronous invocation (1)

```
<receive name=... partnerLink=... operation=... variable=... />
```

```
<invoke name=... partnerLink=... operation=... variable=... />
```
Communication: Asynchronous invocation (2)

<invoke name=... partnerLink=... operation=... variable=... />

<receive name=... partnerLink=... operation=... variable=... />

© Joerg Nitzsche
Control Logic

- Sequential execution – sequence
- Parallel execution of tasks → Flow, links
- Branching → if then else
- Loops: while, repeat until

© Zhilei Ma
Data Manipulation

• Data Flow in BPEL
  – No explicit data flow modelled
  – Data flow implicit via global/scoped variables, access via name
  – Variables can be defined as
    • WSDL message type
    • XML Schema type (simple or complex)
    • XML Schema element

• Data Manipulation
  – BPEL Assign activity
  – Allows copying (parts of) variables
    • XPath can be used to identify these parts

• Example

```xml
<assign>
  <copy>
    <from>$po/lineItem[@prodCode=$myProd]/amt * $exchangeRate</from>
    <to>$convertedPO/lineItem[@prodCode=$myProd]/amt</to>
  </copy>
</assign>
```
Exception Handling

- Scopes are constructs denoting
  - Units of functionality
  - Have all-or-nothing semantics

- Exception Handling
  - FaultHandler – provide an alternative path in a process, executed for a particular type of failure
  - CompensationHandler – reverse the effects of successfully executed activities
    - Can be called by compensation handlers
    - Default and custom-defined compensation handlers

- Reaction to external events
  - EventHandler – used to specify actions to be taken upon an external event
  - Executed in parallel to the process/activity/scope
Architecture of a BPEL Engine

- Process Engine
  - Deployment Component
  - Build Time Data (Process models)
  - Instance Data (Process Instances)
  - Navigation
  - Event Manager
  - Communication Manager
  - Invocation and Management Framework
  - Execution History
  - Monitoring API

© Dimka Karastoyanova, Tammo van Lessen, Joerg Nitzsche
* Receive may cause an Instantiation of a Process
Semantic Web Services

© Michael Stollberg
Outline

• Introduction
  – The need of semantics for realizing the SOA vision
  – Semantic Web – Web Services – SWS

• SWS Frameworks
  – Requirements
  – The WSMO Approach
    1. Ontologies
    2. Goals
    3. Web services
    4. Mediators

• SWS Techniques
  – Discovery
  – Composition
  – Mediation
The Idea – Realizing “Semantic SOA”

Dynamic

- Web Services
  - UDDI, WSDL, SOAP

Static

- WWW
  - URI, HTML, HTTP

→ Semantic Web
  - RDF, RDF(S), OWL
The Semantic Web

- next generation of the Internet (augmentation of the WWW)
- information has machine-processable and machine-understandable semantics
- ontologies as base technology for semantic interoperability
Ontology Definition

formal, explicit specification of a shared conceptualization

- unambiguous terminology definitions
- machine-readability with computational semantics
- commonly accepted understanding

conceptual model of a domain (ontological theory)
Ontology Example

Concept
categorical entity of the domain

Property
attribute describing a concept

Relation
relationship between concepts or properties

Axiom
coherency description between Concepts / Properties / Relations via logical expressions

Instance
individual in the domain

holds(Professor, Lecture) => Lecture.topic = Professor.researchField

Ann memberOf student
name = Ann Lee
studentID = 12345
Ontology Languages

Requirements
  – expressivity
  – reasoning support
  – web compliance

W3C Semantic Web Language Cake
revised version, Tim-Berners-Lee 2005
Ontology Technology

- **Ontology Reasoning**
  - advanced information processing
    - special requirements
      - large scale knowledge handling
      - fault-tolerant
      - stable & scalable inference machines

- **Ontology Management**
  - (collaborative) editing and browsing
  - storage and retrieval
  - versioning and evolution support
Ontology-Based Data Integration

- Ontology Integration Techniques
  - Ontology Mapping
    - Mapping Rules
  - Ontology Alignment
    - Ontology A is made compatible to ontology B
  - Ontology Merging
    - Ontology A is made compatible to ontology B

- integration on semantic level (domain independent)
- semi-automatic
  - human intervention needed for “integration decision
  - graphical support for ontology mapping as central technique
Web Services & SOA

• Web Service = program accessible over the Web
• Service-Oriented Architecture (SOA):
  dynamically find & invoke those Web services that allow to solve a particular request
• Web Service Technologies:
  1. **WSDL**  Web Service Description Language
     • in- and outgoing messages
     • technical access (port type, protocol, etc.)
  2. **SOAP**  XML data exchange protocol for the Web
  3. **UDDI**  registry for Web Services
The Web Service Usage Process

**web-based SOA as new system design paradigm**

- Repository
  - find usable Web Service
- Consumer
  - points to WSDL
- WSDL
  - describes Web Service
- SOAP
  - WS usage via message exchange
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

=> initial Web Service Technology Stack failed to realize the SOA Vision
Aim: Realize the SOA Vision

- automate Web Service technologies by
  1. rich, formal annotation of Web Services
  2. inference-based techniques for automated discovery, composition, mediation, execution of Web Services

- integration with the Semantic Web
  - ontologies as data model
  - Web Services as integral part

- semantic SOA
  - also semantically describe client requests
  - automate complete SOA process
  - semantically enhance SOA technology
Web Service Annotation

a) Web Service Description Structure

Interface

Web Service Implementation
(not of interest in Web Service Description)

XML

b) Semantic Web Service Description Structure

Non-functional

Functionality

Web Service Implementation
(not of interest in Web Service Description)

Interface

Aggregation

Ontology

Ontology

Ontology

a) Web Service Description Structure

b) Semantic Web Service Description Structure
Semantic Web Service Technologies

Request

Submit

Discoverer

if: directly usable
uses
if: composition needed
uses
Composer

Communication Conformance

Executor

if: compatible
uses
if: execution error
uses
Data Mediator

Process Mediator

else: try other WS

else: try other WS

matchmaking R with all WS

Service Repository

uses
if: successful
else: try other WS

uses
if: compatible
else: try other WS

uses
if: execution error
else: try other WS
Requirements & Frameworks

• Requirements for SWS Frameworks
  – cover all aspects relevant for enabling automated Web service usage
  – define conceptual model & axiomatization (= semantics)
  – provide formal language for semantic descriptions

• Approaches (W3C Member Submissions)
  1. WSMO: Ontologies, Goals, Web Services, Mediators
  2. OWL-S: WS Description Ontology (Profile, Service Model, Grounding)
  3. SWSF: Process-based Description Model & Language for WS
  4. WSDL-S: semantic annotation of WSDL descriptions
• **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

[www.wsmo.org](http://www.wsmo.org)
WSMO Top Level Notions

Objectives that a client wants to achieve by using Web Services

Formally specified terminology used by all other components

Connectors between components with mediation facilities for handling heterogeneities

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

*W3C submission 13 April 2005*
WSMO Web Service Description

- complete item description
- quality aspects
- Web Service Management

Non-functional Properties

- Advertising of Web Service
- Support for WS Discovery

Capability

Web Service Implementation
(not of interest in Web Service Description)

client-service interaction interface for consuming WS
- external visible behavior
- communication structure
- ‘grounding’

Non-functional Properties

DC + QoS + Version + financial

- complete item description
- quality aspects
- Web Service Management

Capability

Web Service Implementation
(not of interest in Web Service Description)

client-service interaction interface for consuming WS
- external visible behavior
- communication structure
- ‘grounding’

Non-functional Properties

DC + QoS + Version + financial

- Advertising of Web Service
- Support for WS Discovery

Capability

functional description

realization of functionality by aggregation
- 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)
• **Choreography** = how to interact with the service to consume its functionality

• **Orchestration** = how service functionality is achieved by aggregating other Web Services
• **External Visible Behavior**
  – those aspects of the workflow of a Web Service where Interaction is required
  – described by workflow constructs: sequence, split, loop, parallel

• **Communication Structure**
  – messages sent and received
  – their order (communicative behavior for service consumption)

• **Grounding**
  – executable communication technology for interaction
  – choreography related errors (e.g. input wrong, message timeout, etc.)

• **Formal Model**
  – reasoning on Web Service interfaces (service interoperability)
  – semantically enabled mediation on Web Service interfaces
Orchestration

interface for interaction with aggregated Web Services

- decomposition of service functionality
- other Web services consumed via their choreography interfaces

Web Service Business Logic

State in Orchestration
Control Flow
Data Flow
Service Interaction
WSMO Web Service Interfaces

- behavior interfaces of Web services and clients for “peer-2-peer” interaction

- Choreography and Orchestration as sub-concepts of Service Interface with common description language

Web Service Interface Description aspects:
1. represent the **dynamics** of information interchange during service consumption and interaction
2. support **ontologies** as the underlying data model
3. appropriate **communication technology** for information interchange
4. sound **formal model / semantics** of service interface specifications in order to allow advanced reasoning on them

=> “ontologized Abstract State Machines”
WSMO Goals

**Client-Side**

- **Goal Template**: generic objective description
- **Goal Instance**: concrete input

**Service-Side**

- **(Web) Service Implementation**
  - **functional**
  - **behavioral**

**Domain Knowledge**

- **Ontology**
- **Ontology**
- **Ontology**
- **Ontology**

- **service detection**
- **service usage**

Client defines Goal Template which instantiates Goal Instance. Domain Knowledge includes Ontologies supporting service usage and detection.
Goal Model

Composite Goal
- subGoals: goal
- workflow: orchestration

Goal
- domain: ontology
- objective: capability

automated Web service usage

WG Mediator
- source: goal
- target: webService
- usability: matchingdegree
- clientInterface: choreography

desired workflow

instantiation
Basic Goal

Goal

buy train ticket in Germany
- origin: o, destination: d
- date-time: dt

goal instance with inputs:
o = Munich, d = Berlin
dt = 20070319-1030

Client

instantiates

defines

WG Mediator

client interface

design time

runtime

DB Ticketing

executes
Composite Goal

Flight-hotel booking with desired workflow

- Flight Request
  - if hotel = Ø
  - flight.outwardArrival = hotel.arrival

- Hotel Request
  - if flight = Ø
  - hotel information

- Book Flight
  - flight information

- Book Hotel

Flight WS

Interface (Chor.)
1) get request
2) provide offer
3) receive selection
4) send confirmation

Hotel WS

Interface (Chor.)
1) get request
2) provide offer
3) receive selection
4) send confirmation
Web Service Discovery

detect directly usable Web services out of available ones

• Discovery Techniques (functional as primary focus)
  - 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
  - usage costs
  - ...
Exact Match:
\[ G, WS, O, M \models \forall x. (G(x) \iff WS(x)) \]

PlugIn Match:
\[ G, WS, O, M \models \forall x. (G(x) \implies WS(x)) \]

Subsumption Match:
\[ G, WS, O, M \models \forall x. (G(x) \subseteq WS(x)) \]

Intersection Match:
\[ G, WS, O, M \models \exists x. (G(x) \land WS(x)) \]

Non Match:
\[ G, WS, O, M \models \neg \exists x. (G(x) \land WS(x)) \]

WSMO Discovery Process

Goal-Repos.
- Predefined formal Goal

Available WS
- Abstract Capability

Requester Desire
- Goal Discovery
  - Requester Goal
  - Web Service Discovery
    - Concrete Capability (possibly dynamic)

Selected predefined Goal
- Goal refinement
  - Web Service (Service Discovery)
    - Service to be returned

Still relevant WS

Ease of description
- Efficient Filtering
- Accuracy
**Web Service Composition**

**combine several Web services for solving a request**

- composition of Web services is needed
  - a) a WS can satisfy goal, but goal cannot invoke WS
  - b) several WS need to be combined to achieve goal

- composition techniques:
  - functional = composition wrt *functionalties*
  - behavioral = composition wrt *behavioral interfaces*
  
  ⇒ *need to be integrated*:
  1. skeleton by functional composition
  2. refinement + executable code by behavioral composition

**Procedure:**

1. **directly usable WS (discovery)?**
   - yes
   - no
2. composition (functional)
   - a) a)
   - b) no
3. composition skeleton
   - yes
   - no
4. executable composition
   - abort
Choreography Discovery

**determine behavioral compatibility**

- **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 \( \emptyset \) then request

\[ \Omega_G(\omega 1) = \{\text{request(out)}\} \]

if \( \text{cnd1(offer)} \) then changeReq

\[ \Omega_G(\omega 2a) = \{\text{offer(in)}, \text{changeReq(out)}\} \]

if \( \text{cnd2(offer)} \) then order

\[ \Omega_G(\omega 2b) = \{\text{offer(in)}, \text{order(out)}\} \]

if \( \text{conf} \) then \( \emptyset \)

\[ \Omega_G(\omega 3) = \{\text{offer(in)}, \text{conf(in)}\} \]

WS Choreography Interface

\[ \Omega_{VTA}(\omega \emptyset) = \{\emptyset\} \]

if \( \text{request} \) then offer

\[ \Omega_{VTA}(\omega 1) = \{\text{request(in)}, \text{offer(out)}\} \]

if \( \text{changeReq} \) then offer

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

if \( \text{order} \) then conf

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

valid choreography existent
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:**
  1. **Representation Level:** heterogeneous Languages & Protocols
  2. **Data Level:** heterogeneous Data Sources
  3. **Functional Level:** heterogeneous Functionalities
  4. **Process Level:** heterogeneous Communication Processes
Data Mediation Techniques

- resolve semantic mismatches between terminologies
  - realized by ontology integration
    - mappings between heterogeneous ontologies (design time)
    - data transformation (runtime)
Ontology O1

Human
- name

Adult
Child

Ontology O2

Person
- name
- age

michael memberOf Person
- name = Michael Stollberg
- age = 28

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
Process Level Mediation

- not a priori compatible behavior interfaces for communication & information interchange => behavioral incompatibility

- partially resolvable by process mediation patterns
can resolve about 80% of process level mismatches
Unresolvable Process Mismatches
Process Mediation Example

REQUEST

itinerary[origin, destination, date] →

 Processes Mediator

→

SERVICE

origin

destination

itinerary[origin, destination]

date

itinerary [route, date, time, price]
Process Mediation Example

REQUEST

- itinerary [origin, destination, date]
- time
- price

Processes Mediator

SERVICE

- origin
- destination
- itinerary [origin, destination]
- date
- itinerary [route, date, time, price]
Process Mediation Example

**Request**: itinerary[origin, destination, date], time, price

**Processes Mediator**: itinerary[route, date, price]

**Service**: origin, destination, itinerary[route, date, price]
Process Mediation Example

Process Mediation Example

Processes Mediator

REQUEST

itinerary[origin, destination, date]

time

price

SERVICE

origin
destination

itinerary[origin, destination]
date

itinerary [route, date, time, price]
Process Mediation Example

REQUEST

itinerary[origin, destination, date]

time

price

Processes Mediator

SERVICE

origin

destination

itinerary[origin, destination]

date

itinerary [route, date, time, price]
WSMO Mediators Overview

- **OO Mediator**: O / G / WS / M
- **GG Mediator**: G
- **WG Mediator**: WS xor G
- **WW Mediator**: WS

**Legend**
- Technique used
- imports / reuses
- correlation

**Data level mediation**
- **terminology**
- **representation & protocol**

**Process Level**
- (Communication)
- (Cooperation)
The Web Service Execution Environment
WSMX (WSMO Reference Implementation)

Open source code base at SourceForge: http://sourceforge.net/projects/wsmx/
Other WSMO Tools
www.wsmo.org/tools

- WSML (Specification Language) www.wsmo.org/wsml
  - conceptual language for WSMO
  - ontology language with several variants

- WSMO Editors:
  - WSML editors + validation
  - WSMO Studio
  - WSMO Visualizer

- Ontology Technology:
  - WSML Reasoner (for DL and LP)
  - Ontology Management Suite
  - Data Mediator (incl. Abstract Mapping Language)

all: Eclipse plugins & open source (LGPL licence)
• Conceptual Model
  – A set of ontologies used to describe different aspects SWS

• Language: OWL

• Some OWL-S drawbacks
  – OWL not sufficiently expressive for all aspects of a service
    • more expressive languages have been syntactically integrated: SWRL, KIF, DRS, and PDDL – how do these languages interoperate?
  – Inherits some of the drawbacks of OWL (e.g. lack of proper layering, improper use of OWL for describing and reasoning about processes)
  – No explicit support for Mediation in the language
Semantic Web Services Framework (SWSF)

- Two major components: an ontology and a language used to axiomatize it
  - Semantic Web Services Ontology (SWSO) – an extension of OWL-S conceptual model, e.g. a rich behavioural process model based on PSL
    - FLOWS – First-Order Logic Ontology for Web Services
    - ROWS - Rule Ontology for Web Services
  - The Semantic Web Services Language (SWSL)
    - SWSL-FOL - based on First Order Logic; includes features from HiLog and F-Logic
    - SWSL-Rules - a logic programming language; includes features from Courteous logic programs, HiLog, and F-Logic
- Some SWSF drawbacks
  - unclear how all the paradigms part of this approach work together
  - first-order logic ontology for Web services, but not a Web language
Web Service Semantics - WSDL-S

- A mechanism to augment WSDL descriptions with semantics
  - a set of annotations can be created to semantically describe the inputs, outputs and operations of a Web service.
  - keeps the semantic model outside WSDL, making the approach agnostic to any ontology representation language.

- WSDL-S doesn’t provide a conceptual model and language for SWS
  - a bottom up approach to SWS (annotating existing standards with metadata)
- Could be used as a grounding mechanism for SWS
SWS Conclusions

- Semantic Web Services
  - Initial technical solutions existing
  - High potential in BPM, B2B, EAI, eCommerce, etc.
- The WSMO Approach to SWS looks promising
  - A unifying approach for semantic SOA
  - Top-level entities: Ontologies, Web Services, Goals, Mediators
- Standardization activities are emerging in this area
  - OWL-S, SWSF, WSDL-S, WSMO – submitted to W3C
  - OASIS SEE technical committee formed (based on WSMX)
  - W3C SAWSDL Working Group formed; closed to recommendation
- Future Aspects:
  - Apply & extend towards usage scenarios
  - Educate Ontology & SWS Engineers
Integration of SWS into BPM

[The SUPER Approach]
Modelling Stack

- Making sense of a domain problem
- Communication tool
- What is it all about?

- Visualizing/specifying business process
- Focus: Business Problem
- Who does what, when, how and why?
- Usually multiple layers

- Process execution specification
- Formal, clearly specified grammar
- Focus: Implementation
- Which component is called when, how, by whom with which data?

- Web service encapsulation
- Focus: Implementation
- Which components can and should be exposed as services?

- Implementation of components
- Solution maps
- Mind maps
- Ad-hoc modelling techniques
- Business Scenario Maps
- Event-driven process chains
- Flowchart techniques
- BPMN
- WS*
- Programming languages
Modelling Stack

- Making sense of a domain/problem
- Communication tool
- What is it all about?

- Visualizing/specifying business process
- Focus: Business Problem
- Who does what, when, how and why?
- Usually multiple layers

- Process execution specification
- Formal, clearly specified grammar
- Focus: Implementation
- Which component is called when, how, by whom with which data?

- Web service encapsulation
- Focus: Implementation
- Which components can and should be exposed as services?

- Implementation of components
- Solution maps
- Mind maps
- Ad-hoc modelling techniques
  - Business Scenario Maps
  - Event-driven process chains
  - Flowchart techniques
  - BPMN
  - ...

- BPEL
  - WS*
  - ...

- Programming languages
  - ...

Telecommunications Solution Map

Industry Value Chain

<table>
<thead>
<tr>
<th>Suppliers &amp; Partners</th>
<th>Design &amp; Build Infrastructure</th>
<th>Operate Infrastructure</th>
<th>Develop &amp; Promote Products &amp; Services</th>
<th>Sell &amp; Fulfill</th>
<th>Bill &amp; Collect</th>
<th>Assist Customers</th>
<th>Customers &amp; Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Network Life-Cycle Management</td>
<td></td>
<td></td>
<td>Sales &amp; Service Fulfillment</td>
<td>Loss and Opportunity Management</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Demand and Supply Planning</td>
<td></td>
<td></td>
<td>Sales and Order Management</td>
<td>Customer Field Service Management</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Investment Management</td>
<td></td>
<td></td>
<td>Logistics Management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Network Design and Build</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operations and Maintenance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Marketing Analytics & Product Management

Sales & Service Fulfillment
- Loss and Opportunity Management
- Sales and Order Management
- Customer Field Service Management
- Logistics Management

Dealer Management

Billing, Invoicing & Presentement

Customer Financial Management

Customer Service
- Customer Trouble Management
- Complaints and Returns Management
- Customer Field Service Management

Content On Demand
- Digital Rights Management & Content Procurement
- Content Distribution & Billing

Enterprise Management & Support

<process name="Mediation Example - Ordering B PEL Snippet - 1" suppressJoinFailure="yes" targetNamespace= "...">
  <sequence>
    <receive name="Initial_Receive" createInstance="true"/>
    <invoke name="Invoke_Check_Order_Consistency"/>
    <switch>
      <case condition= "bpws:getVariableData('consistency', ' ') = 'OK'">
        <flow>
          <invoke name="Invoke_Update_Provisioning_Systems_Subprocess"/>
          <invoke name="Invoke_CustomerReply_Confirmation_Note"/>
        </flow>
      </case>
      <otherwise>
        <invoke name= "Invoke_CustomerReply_Rejection_Note"/>
      </otherwise>
    </switch>
    <reply name="Final_Reply"/>
  </sequence>
</process>
Content on Demand

Digital Rights Management & Content Procurement

Content Distribution & Billing
Digital Rights Management & Content Procurement

Digital Rights Management & Content Procurement clusters activities related to the acquisition of licences of copyright material as well as the actual content itself. Licence providers and content providers are usually not identical.

![Diagram of Digital Rights Management & Content Procurement]

- **Content Provider**
  - Request Content from Provider
  - Request Licence for Content
  - Bundle Content & Licence
  - Deliver Content

- **Service Provider**
  - Invoke Check Order Consistency
  - Invoke Update Provisioning System Subprocess
  - Invoke Custom Reply Confirmation Note

- **License Provider**
  - Deliver Licence
  - Deliver Content to Customer

- **Message Flow**
  - Sequence
  - Receive (Initial_Receive)
  - Invoke (Invoke_Check_Order_Consistency)
  - Switch
  - Case (bpws:getVariableData('consistency', ' ') = 'OK ')
    - Flow
      - Invoke (Invoke_Update_Provisioning_System_Subprocess)
      - Invoke (Invoke_Custome Reply_Confirmation_Note)
  - Otherwise
    - Invoke (Invoke_CustomerReply_Rejection_Note)
  - Reply (Final_Reply)
### Modelling Stack

- **Making sense of a domain problem**
- **Communication tool**
- **What is it all about?**

- **Visualizing specifying business process**
- **Focus: Business Problem**
- **Who does what, when, how and why?**
- **Usually multiple layers**

<table>
<thead>
<tr>
<th>Process execution specification</th>
<th>Formal, clearly specified grammar</th>
<th>Focus: Implementation</th>
<th>Which component is called when, how, by whom with which data?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web service encapsulation</td>
<td>BPEL</td>
<td>WS*</td>
<td>...</td>
</tr>
<tr>
<td>Focus: Implementation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Which components can and should be exposed how as services?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Implementation of components</strong></td>
<td><strong>Programming languages</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Solution maps**
- **Mind maps**
- **Ad-hoc modelling techniques**
- **...**
- **Business Scenario Maps**
- **Event-driven process chains**
- **Flowchart techniques**
- **BPMN**
- **...**
Integration in the ARIS House

- Inquiry is received
- Inquiry is processed
- Quotation processing
- Customer quotation
- Customer order

- Inquiry
- Customer quotation
- Customer order

- Sales
- Sales processing
- Check credit rating
- Determine delivery date
- Quotation processing

- Customer
- Quotation
- Inquiry

- Scheduling
- Purchasing
- Materials administration
- Management

- Sales processing
- Inquiry processing
- Customer quotation
- Customer order
- Inquiry
- Quotation
- Inquiry is received
- Inquiry is processed
- Quotation processing
- Customer quotation
- Customer order

- Customer inquiry
- Quotation
- Inquiry
- Customer quotation
- Customer order

<process name="Mediation Example - Ordering BPEL Snippet - 1" suppressJoinFailure="yes" target Namespace=" ...">
  <sequence>
    <receive name="Initial_Receive" createInstance="true"/>
    <invoke name="Invoke_Check_Order_Consistency"/>
    <switch>
      <case condition="bpws:getVariableData('consistency', '') = 'OK'">
        <flow>
          <invoke name="Invoke_Update_Provisioning_Systems_Subprocess"/>
          <invoke name="Invoke_CustomerReply_Confirmation_Note"/>
        </flow>
      </case>
      <otherwise>
        <invoke name="Invoke_CustomerReply_Rejection_Note"/>
      </otherwise>
    </switch>
    <reply name="Final_Reply"/>
  </sequence>
</process>
<table>
<thead>
<tr>
<th>Modelling Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Making sense of a domain problem</td>
</tr>
<tr>
<td>- Communication tool</td>
</tr>
<tr>
<td>- What is it all about?</td>
</tr>
<tr>
<td>- Solution maps</td>
</tr>
<tr>
<td>- Mind maps</td>
</tr>
<tr>
<td>- Ad-hoc modelling techniques</td>
</tr>
<tr>
<td>- Visualizing/specifying business process</td>
</tr>
<tr>
<td>- Focus: Business Problem</td>
</tr>
<tr>
<td>- Who does what, when, how and why?</td>
</tr>
<tr>
<td>- Usually multiple layers</td>
</tr>
<tr>
<td>- Business Scenario Maps</td>
</tr>
<tr>
<td>- Event-driven process chains</td>
</tr>
<tr>
<td>- Flowchart techniques</td>
</tr>
<tr>
<td>- BPMN</td>
</tr>
<tr>
<td>- ...</td>
</tr>
<tr>
<td>- Process execution specification</td>
</tr>
<tr>
<td>- Formal, clearly specified grammar</td>
</tr>
<tr>
<td>- Focus: Implementation</td>
</tr>
<tr>
<td>- Which component is called when, how, by whom with which data?</td>
</tr>
<tr>
<td>- BPEL</td>
</tr>
<tr>
<td>- ...</td>
</tr>
<tr>
<td>- Web service encapsulation</td>
</tr>
<tr>
<td>- Focus: Implementation</td>
</tr>
<tr>
<td>- Which components can and should be exposed how as services?</td>
</tr>
<tr>
<td>- WS*</td>
</tr>
<tr>
<td>- ...</td>
</tr>
<tr>
<td>- Implementation of components</td>
</tr>
<tr>
<td>- Programming languages</td>
</tr>
<tr>
<td>- ...</td>
</tr>
</tbody>
</table>
Programming Model

Programming in the Large
Workflow System
Choreography/Flow/Process

Application Server
Programming in the Small
Web Services/Functions

Application
Deploying Applications

Process

A

B

C

D

E

Deployment Descriptors

Activity:
Partner
Role
Port Type
Operation...

EPR

Ports

؟

؟
Business Protocols

Partner Links

Partner 1
W
X
Y
Z

Partner 2
A
B
C
D

1
2
3
4

<process name="Mediation Example - Ordering BPEL Snippet - 1" suppressJoinFailure="yes" targetNamespace="...">
<sequence>
<receive name="Initial_Receive" createInstance="true" />
<invoke name="Invoke_Check_Order_Consistency" />
<switch>
<case condition="bpws:getVariableData('consistency', '') = 'OK'">
<flow>
<invoke name="Invoke_Update_Provisioning_Systems_Subprocess" />
<invoke name="Invoke_CustomerReply_Confirmation_Note" />
</flow>
</case>
<otherwise>
<invoke name="Invoke_CustomerReply_Rejection_Note" />
</otherwise>
</switch>
<reply name="Final_Reply" />
</sequence>
</process>
Modelling Stack

- Making sense of a domain problem
- Communication tool
- What is it all about?

- Visualizing/specifying business process
- Focus: Business Problem
- Who does what, when, how and why?
- Usually multiple layers

- Process execution specification
- Formal, clearly specified grammar
- Focus: Implementation
- Which component is called when, how, by whom with which data?

- Web service encapsulation
- Focus: Implementation
- Which components can and should be exposed how as services?

- Implementation of components
- Solution maps
- Mind maps
- Ad-hoc modelling techniques
- Business Scenario Maps
- Event-driven process chains
- Flowchart techniques
- BPMN
- BPEL
- WS*
- Programming languages
WSMO Top Level Notions

Objectives that a client wants to achieve by using Web Services

- Provide the formally specified terminology of the information used by all other components
- Connectors between components with mediation facilities for handling heterogeneities

Semantic description of Web Services:
- Capability (*functional*)
- Interfaces (*usage*)
SUPER Ontology Stack
Ontologies overview

- SUPER ontologies
  - Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
  - Imports
  - Maps to (unformalised)
  - Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

- DOLCE+DnS Plan (DDPO) Ontology

WSMO Ontology

- Events Ontology (EVO)
- Process Mining Ontology (PMO)
- Components Ontology (CONTO)

sBPEL
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
  - Maps to (unformalised)
  - Translates to

- Organisational-related Ontologies
  - Behavioural Reasoning Ontology (BRO)
  - sEPC Ontology
  - sBPMN Ontology
- Business Process Modelling Ontology (BPMO)
- Upper-Level Process Ontology (UPO)
- DOLCE+DnS Plan (DDPO) Ontology
- WSMO Ontology
- sBPEL
- Events Ontology (EVO)
- Process Mining Ontology (PMO)
- Components Ontology (CONTO)
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

DOLCE+DnS Plan (DDPO) Ontology

WSMO Ontology

sBPEL

Events Ontology (EVO)

Process Mining Ontology (PMO)

Components Ontology (CONTO)
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Business Process Modelling Ontology (BPMO)

- DOLCE+DnS Plan (DDPO) Ontology
- WSMO Ontology

Events Ontology (EVO)

Process Mining Ontology (PMO)

Components Ontology (CONTO)

sBPEL
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

Business Process Modelling Ontology (BPMO)

DOLCE+DnS Plan (DDPO) Ontology

WSMO Ontology

sBPEL

Events Ontology (EVO)

Process Mining Ontology (PMO)

Components Ontology (CONTO)
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- Business Process Modelling Ontology (BPMO)
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

- DOLCE+DnS Plan (DDPO) Ontology
- WSMO Ontology
- sBPEL
- Events Ontology (EVO)
- Process Mining Ontology (PMO)
- Components Ontology (CONTO)
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

- DOLCE+DnS Plan (DDPO) Ontology
- WSMO Ontology
- Events Ontology (EVO)
- Process Mining Ontology (PMO)
- Components Ontology (CONTO)
Ontologies overview

- SUPER ontologies
- Cluster of SUPER ontologies, some of which may reuse existing ontologies
- Pre-existing ontologies
- Imports
- Maps to (unformalised)
- Translates to

Organisational-related Ontologies

- Behavioural Reasoning Ontology (BRO)
- sEPC Ontology
- sBPMN Ontology

Upper-Level Process Ontology (UPO)

- DOLCE+DnS Plan (DDPO) Ontology
- WSMO Ontology

- Events Ontology (EVO)
- Process Mining Ontology (PMO)
- Components Ontology (CONTO)

sBPEL
SUPER Architecture and Scenarios
Architecture Static View

**SUPER Execution Environment**
- Semantic BPEL Execution Engine
- Semantic Execution Environment

**SUPER Tooling**
- Modelling Tool
- Monitoring Tool
- Mining Tool

**Deployment Component**

**Semantic Service Bus**
- Composition
- Discovery
- Mediation
- Translation

**SUPER platform services**

**SUPER Repositories**
- Semantic Web Services
- Execution History
- Business Process Library

**SUPER**
- Tooling
- Repositories
- Execution Environment

**SUPER**
- Platform services
- Deployment Component
- Deployment Component

**Semantic Service Bus**
- Composition
- Discovery
- Mediation
- Translation

**SUPER**
- Tooling
- Repositories
- Execution Environment

**SUPER**
- Platform services
- Deployment Component
- Deployment Component

**Semantic BPM Tutorial, ICIW 2007, Mauritius, 13 May 2007**

© SUPER 5/8/2007
Deployment Process: Semantic Process Artefacts Bundle (SPAB)

- BPEL4SWS
- WSDL
- WSMO
  - WSDL
- WSMO Mediators
- WSMO Goals
- Deployment descriptor
Deployment Process

- Semantic BPEL Execution Engine
- Semantic Execution Environment
- Modelling Tool
- Monitoring Tool
- Mining Tool

BPEL4SWS, WSDL

Semantic Service Bus

- WSMO
- WSDL

SUPER platform services

- Composition
- Discovery
- Mediation
- Translation

SUPER Repositories

- Semantic Web Services
- Execution History
- Business Process Library

SUPER Tooling

- Deployment Component

WSMO Mediators
- WSMO Goals

SUPER Tooling

SUPER 5/8/2007
Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service
4. Invoke Service
5. Return result to engine
6. Return Result

SUPER Execution Environment

SUPER Repositories
- Semantic Web Services
- Execution History

Mediation

Web Service

Semantic BPEL Execution Engine

Semantic Execution Environment

Semantic Business Process Execution

1. Request to Service

SUPER Execution Environment

Mediation

SUPER platform services

Semantic Execution Environment

Semantic BPEL Execution Engine

Web Service

SUPER Repositories

Semantic Web Services

Execution History

Semantic Business Process Execution
Semantic Business Process Execution

1. Request to Service

2. Achieve Goal

Semantic BPEL Execution Engine

Semantic Execution Environment

SUPER Execution Environment

SUPER Repositories

Semantic Web Services

Execution History

Web Service

SUPER platform services

Mediation

Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service

- SUPER Execution Environment
- Mediation
- Web Service
- SUPER Repositories
  - Semantic Web Services
  - Execution History

- Semantic BPEL Execution Engine
- Semantic Execution Environment

Achieve Goal
Discover Service

Semantic Business Process Execution

Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service
4. Invoke Service

SUPER Execution Environment

Semantic BPEL Execution Engine

Mediation

SUPER platform services

Web Service

Semantic Execution Environment

Semantic Web Services

SUPER Repositories

Execution History
Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service
4. Invoke Service
5. Return result to engine

SUPER Execution Environment

Mediation

SUPER platform services

Web Service

SUPER Repositories

Semantic Web Services

Execution History

Semantic BPEL Execution Engine

Semantic Execution Environment
Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service
4. Invoke Service
5. Return result to engine
6. Return Result

SUPER Execution Environment

Semantic BPEL Execution Engine

Mediation

SUPER platform services

Web Service

SUPER Repositories

Semantic Web Services

Execution History

Semantic Execution Environment

Semantic Business Process Execution Engine

Semantic Web Services

Return result to engine

Semantic Execution Environment

Semantic BPEL Execution Engine

Discover Service

Invoke Service

Return Result
Semantic Business Process Execution
on the SUPER Infrastructure

— DEMONSTRATION —
Semantic Business Process Execution

1. Request to Service
2. Achieve Goal
3. Discover Service
4. Invoke Service
5. Return result to engine
6. Return Result

SUPER Execution Environment

Mediation

SUPER Repositories

Semantic Execution Environment

Semantic BPEL Execution Engine

Web Service

Semantic Web Services

Execution History
Purchase Content Process

Customer

[Service Task]: Purchase content

Service provider

[Receive Task]: Receive content request

[Service Task]: Get content

License server

[Receive Task]: Receive license request

[Send Task]: Send license

Packager

[Receive Task]: Receive URL request

[Send Task]: Send URL

UserID and ContentID

URL and License

UserID and ContentID

License

UserID and ContentID

URL
The Content Purchase Process

BPEL4SWS Process (2)

- Sequence
- receiveContentRequest
- aggregateResult
- reply
- invokeGoalGenerateLicense
- invokeGoalGenerateURL
Demonstration - Process Client

Purchase Content

Please first provide login information and then choose the content you wish to download. You will be provided the URL and the licence.

Login information
Username:
Password:

Content
Please provide the Content-Id of the content you wish to purchase.

Content-Id:

Submit request
Result of the Execution of the Purchase Content Process

Thanks for your purchase!

Please find below the URL and the license of the requested content.

Content access information
- Content: http://youtube.com/watch?v=Dui5NfQjT
- License: "THE WORK IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE"
### Semantic Business Processes Monitor

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Event Information</th>
<th>Generated By</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Achieve Goal</td>
<td>Goal: goalGetLicense</td>
<td>IRS</td>
<td>Tue Jan 23 18:31:12 CET 2007</td>
</tr>
<tr>
<td>Start Activity Execution</td>
<td>invokeGoalGenerateURL (OExtensionActivity)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:11 CET 2007</td>
</tr>
<tr>
<td>End Activity Execution</td>
<td>ReceiveContentRequest (OPickReceive)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:11 CET 2007</td>
</tr>
</tbody>
</table>
### Semantic Business Processes Monitor

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Event Information</th>
<th>Generated By</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ContentProvision</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End Activity Execution</td>
<td>__unnamed:[<a href="http://schemas.xmisoap.org/ws/2004/03/business-process/sequence@45">http://schemas.xmisoap.org/ws/2004/03/business-process/sequence@45</a> (OSequence)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:24 CET 2007</td>
</tr>
<tr>
<td></td>
<td>reply (OReply)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Activity Execution</td>
<td>reply (OReply)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:24 CET 2007</td>
</tr>
<tr>
<td>End Activity Execution</td>
<td>aggregateResult (OAssign)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:24 CET 2007</td>
</tr>
<tr>
<td>Start Activity Execution</td>
<td>aggregateResult (OAssign)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:24 CET 2007</td>
</tr>
<tr>
<td>End Activity Execution</td>
<td>invokeGoalGenerateLicense (OExtensionActivity)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:24 CET 2007</td>
</tr>
<tr>
<td>End Achieve Goal</td>
<td>Goal: goalGetURL</td>
<td>WSMX</td>
<td>Tue Jan 23 17:31:23 CET 2007</td>
</tr>
<tr>
<td>Start Activity Execution</td>
<td>invokeGoalGenerateLicense (OExtensionActivity)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:14 CET 2007</td>
</tr>
<tr>
<td>End Activity Execution</td>
<td>invokeGoalGenerateURL (OExtensionActivity)</td>
<td>SBPELEE</td>
<td>Tue Jan 23 17:31:14 CET 2007</td>
</tr>
<tr>
<td>End Achieve Goal</td>
<td>Goal: goalGetLicense</td>
<td>IRS</td>
<td>Tue Jan 23 18:31:14 CET 2007</td>
</tr>
</tbody>
</table>
References
References BPM Foundations


References BPEL & BPM Technologies


References Semantic Web Services


References SWS: W3C Submissions

OWL-S

WSMO [see also www.wsmo.org]

SWSF

WSDL-S
References


References Mediation


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/


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