



Semantics Utilised for Process Management
within and between Enterprises



A Business Process Modelling Ontology

10th February 2009

Barry Norton, Open University



Introduction

- Previous talk on Semantic BPM centred on semantic extensions to the Business Process Execution Language (BPEL)
- Semantic BPEL, however, is oriented towards SOA-based implementation and there is a need for a representation which is more:
 - abstract;
 - graphically-represented;
 - contextualised in (cross-)organisational and domain models;
 - subject to automated reasoning;
 - subject to formal behavioural semantics.



Background 1

- Current favoured notations in the BPM community are:
 - BPMN (Business Process Modelling Notation)
 - an OMG specification
 - graphically very rich in control flow
 - weak in organisational context
 - semantically under-defined
 - increasingly used and tied to BPEL in cross-referential standards.
 - EPCs (Event-driven Process Chains)
 - of academic origin
 - popularised via ARIS Toolkit and in conjunction with SAP R/3
 - weak in control flow and semantics
 - relatively rich in domain and organisational modelling



Background 2

- We also mention for comparison
 - Workflow Nets
 - of academic origin and very heavily represented in academic BPM community
 - an extension of Petri nets and semantically well-defined
 - less of a direct notation (except for masochistic academics) and more of a common behavioural model translated from and to industry notations
 - Workflow Patterns
 - a notation-independent common vocabulary of control-flow (and later data-flow) idioms of workflow languages
 - given formal semantics in Petri nets
 - YAWL
 - an academic attempt to make a well-defined notation out of work in Workflow Patterns
 - not widely adopted



Motivation

- The motivation of BPMO is:
 - To be a common translation destination and source for industry notations
 - To provide such a common representation without being as low-level and cumbersome as Workflow Nets
 - To identify common control-flow features via Workflow Patterns vocabulary
 - To be represented in BPMN, rather than imposing a new notation (like YAWL)
 - To preserve and build on the organisational and domain modelling capabilities of EPCs, rather than concentrating only on behaviour
 - To be the basis of automated ontology-based reasoning
 - To have a formal connection to both an underlying behavioural semantics and to executable processes also via ontology-based reasoning



SUPER BPMO's top-level concepts

```
concept BusinessActivity subConceptOf upo#BusinessActivity
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string
  hasNonFunctionalProperties ofType (0 1) BusinessActivityNonFunc-
tionalProperties
  hasBusinessDomain ofType upo#BusinessDomain
  hasBusinessFunction ofType upo#BusinessFunction
  hasBusinessStrategy ofType upo#BusinessStrategy
  hasBusinessPolicy ofType upo#BusinessPolicy
  hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
  hasBusinessProcessGoal ofType upo#BusinessProcessGoal
  hasBusinessResource ofType upo#Resource

concept Process subConceptOf {BusinessActivity, upo# BusinessProcess-
Model}
  hasWSDescription ofType (0 1) SemanticCapability
  hasWorkflow ofType (0 1) Workflow

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1 1) WorkflowElement
```



SUPER BPMO's top-level concepts

```
concept BusinessActivity subConceptOf upo#BusinessActivity
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string
  hasNonFunctionalProperties ofType (0 1) BusinessActivityNonFunc-
tionalProperties
  hasBusinessDomain ofType upo#BusinessDomain
  hasBusinessFunction ofType upo#BusinessFunction
  hasBusinessStrategy ofType upo#BusinessStrategy
  hasBusinessPolicy ofType upo#BusinessPolicy
  hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
  hasBusinessProcessGoal ofType upo#BusinessProcessGoal
  hasBusinessResource ofType upo#Resource

concept Process subConceptOf {BusinessActivity, upo# BusinessProcess-
Model}
  hasWSDescription ofType (0 1) SemanticCapability
  hasWorkflow ofType (0 1) Workflow

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1 1) WorkflowElement
```

Dependence on external 'upper-level' ontology to ground abstract concepts and link to other (e.g. organisational) ontologies



SUPER BPMO's top-level concepts

```
concept BusinessActivity subConceptOf upo#BusinessActivity
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string
  hasNonFunctionalProperties ofType (0 1) BusinessActivityNonFunc-
tionalProperties
  hasBusinessDomain ofType upo#BusinessDomain
  hasBusinessFunction ofType upo#BusinessFunction
  hasBusinessStrategy ofType upo#BusinessStrategy
  hasBusinessPolicy ofType upo#BusinessPolicy
  hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
  hasBusinessProcessGoal ofType upo#BusinessProcessGoal
  hasBusinessResource ofType upo#Resource

concept Process subConceptOf {BusinessActivity, upo# BusinessProcess-
Model}
  hasWSDescription ofType (0 1) SemanticCapability
  hasWorkflow ofType (0 1) Workflow

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1 1) WorkflowElement
```

Meta-modelling confusions (note name is not pretty name, this is inherited from UPO)



SUPER BPMO's top-level concepts

```
concept BusinessActivity subConceptOf upo#BusinessActivity
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string
  hasNonFunctionalProperties ofType (0 1) BusinessActivityNonFunc-
tionalProperties
  hasBusinessDomain ofType upo#BusinessDomain
  hasBusinessFunction ofType upo#BusinessFunction
  hasBusinessStrategy ofType upo#BusinessStrategy
  hasBusinessPolicy ofType upo#BusinessPolicy
  hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
  hasBusinessProcessGoal ofType upo#BusinessProcessGoal
  hasBusinessResource ofType upo#Resource

concept Process subConceptOf {BusinessActivity, upo# BusinessProcess-
Model}
  hasWSDescription ofType (0 1) SemanticCapability
  hasWorkflow ofType (0 1) Workflow

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1 1) WorkflowElement
```

WSMO/L's own meta-modelling problems (as in Semantic BPEL a
WSML model of WSMO-Lite might rescue this)



SUPER BPMO's top-level concepts

```
concept BusinessActivity subConceptOf upo#BusinessActivity
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string
  hasNonFunctionalProperties ofType (0 1) BusinessActivityNonFunc-
tionalProperties
  hasBusinessDomain ofType upo#BusinessDomain
  hasBusinessFunction ofType upo#BusinessFunction
  hasBusinessStrategy ofType upo#BusinessStrategy
  hasBusinessPolicy ofType upo#BusinessPolicy
  hasBusinessProcessMetrics ofType upo#BusinessProcessMetrics
  hasBusinessProcessGoal ofType upo#BusinessProcessGoal
  hasBusinessResource ofType upo#Resource

concept Process subConceptOf {BusinessActivity, upo# BusinessProcess-
Model}
  hasWSDescription ofType (0 1) SemanticCapability
  hasWorkflow ofType (0 1) Workflow

concept Workflow subConceptOf upo#ProcessOrchestrationSpecification
  hasHomeProcess ofType (0 1) Process
  hasFirstWorkflowElement ofType (1 1) WorkflowElement
```

Under-axiomatised



Workflow Definition in SUPER's BPMO

The distinguishing feature of BPMO graphical elements are related to element containers (e.g. BPMO does not contain Pools or Lanes as in BPMN). The main container in BPMO is a Workflow as described below:

- **Workflow** – The business process container for Workflow Elements. The initial Workflow Element is a *Start event* or a block pattern, commonly a *Sequence* or *ParallelSplitSynchronise*. If the *Start* event is present, subsequent elements will be linked in graph style by *ControlflowConnectors*. If the Workflow Element is a *Sequence*, a sequence flow is implicit. If the Workflow Element is *ParallelSplitSynchronise*, a parallel flow is implicit.
- **Workflow Elements** – These are general elements that belong to a business process workflow, including *Processes*, *Tasks*, *Events*, block patterns and graph patterns;
- **Block Patterns** – These are structured control-flow elements representing workflow decision points (gateways), including *Sequence*, *ParallelSplitSynchronise*, *ExclusiveChoiceMerge*, *DeferredChoiceMerge*, *While* loops, and so on.
- **Graph patterns** – These are link based control-flow elements representing workflow decision points (gateways), including *ParallelSplit*, *ExclusiveChoice*, *DeferredChoice*, *SimpleMerge*, *Synchronise*, and so on.



SUPER BPMO's basic 'nodes'

```
concept GoalTask subConceptOf Task
  hasPartnerGoal ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  messageTo ofType (0 1) Receive
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept Send subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasReceiveCounterpart ofType (0 1) Receive
  messageTo ofType (0 1) Receive
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability

concept Receive subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasSendCounterpart ofType Send
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept ReceiveMessageEvent subConceptOf { IntermediateEvent, Receive }
```



SUPER BPMO's basic 'nodes'

```
concept GoalTask subConceptOf Task
  hasPartnerGoal ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  messageTo ofType (0 1) Receive
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept Send subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasReceiveCounterpart ofType (0 1) Receive
  messageTo ofType (0 1) Receive
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability

concept Receive subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasSendCounterpart ofType Send
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept ReceiveMessageEvent subConceptOf { IntermediateEvent, Receive }
```

Possible gap in using WSMO-Lite to address meta-modelling issue
(though potentially only a capability is needed)



SUPER BPMO's basic 'nodes'

```
concept GoalTask subConceptOf Task
  hasPartnerGoal ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  messageTo ofType (0 1) Receive
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept Send subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasReceiveCounterpart ofType (0 1) Receive
  messageTo ofType (0 1) Receive
  hasOutputDescription ofType SemanticCapability
  requestsCapability ofType (0 1) SemanticCapability

concept Receive subConceptOf Task
  hasPartnerWebService ofType (0 1) SemanticCapability
  hasPartnerRole ofType (0 1) BusinessRole
  hasSendCounterpart ofType Send
  messageFrom ofType (0 1) Send
  hasInputDescription ofType SemanticCapability
  providesCapability ofType (0 1) SemanticCapability

concept ReceiveMessageEvent subConceptOf { IntermediateEvent, Receive }
```

Arguable meta-modelling issue



Mediation in BPMO

```
concept MediationTask subConceptOf Task
  hasSourceTask ofType (0 1) Task
  hasTargetTask ofType (0 1) Task
  hasDataMediator ofType DataMediator

concept Mediator subConceptOf upo#BusinessProcessMediator
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string

concept ProcessMediator subConceptOf Mediator
  hasSourceProcess ofType (1 1) Process
  hasTargetProcess ofType (1 *) Process
  hasMediationProcess ofType (0 1) MediationProcess
  hasSWSMediator ofType (0 1) SemanticCapability

concept DataMediator subConceptOf Mediator
  hasMediator ofType SemanticCapability
  hasMediationService ofType SemanticCapability
  hasInputDescription ofType (0 1) SemanticCapability
  hasOutputDescription ofType (0 1) SemanticCapability

concept MediationProcess subConceptOf Process
```



Mediation in BPMO

```
concept MediationTask subConceptOf Task
  hasSourceTask ofType (0 1) Task
  hasTargetTask ofType (0 1) Task
  hasDataMediator ofType DataMediator

concept Mediator subConceptOf upo#BusinessProcessMediator
  hasName ofType (0 1) _string
  hasDescription ofType (0 1) _string

concept ProcessMediator subConceptOf Mediator
  hasSourceProcess ofType (1 1) Process
  hasTargetProcess ofType (1 *) Process
  hasMediationProcess ofType (0 1) MediationProcess
  hasSWSMediator ofType (0 1) SemanticCapability

concept DataMediator subConceptOf Mediator
  hasMediator ofType SemanticCapability
  hasMediationService ofType SemanticCapability
  hasInputDescription ofType (0 1) SemanticCapability
  hasOutputDescription ofType (0 1) SemanticCapability

concept MediationProcess subConceptOf Process
```

Meta-modelling issue in WSMO (can't subclass Mediator) disappears
in WSMO-Lite (no Mediator to subclass)



Plan for Action

- In order to adapt SUPER's BPMO to a CMS standard:
 - flatten UPO and organisational ontologies into it
 - restructure so that graph/container structure is reinstated
 - document intended notation via BPMN
 - complete axiomatisation
 - relate to Semantic BPEL ontology via rule-based axioms