Semantic Lifting of Business Process Models

Mario Lezoche, Antonio de Nicola, Tania Di Mascio, Francesco Taglino

Semantic Lifting of Business Process models

Mario Lezoche[1], Antonio De Nicola[1], Tania Di Mascio[1,2], Francesco Taglino[1]

[1]IASI-CNR, Viale Manzoni 30, Rome, I-00185, Italy
{denicola, taglino}@iasi.cnr.it
[2]Department of Electric and Information Engineering, University of L'Aquila
L'Aquila I-67040 Monteluco di Roio, Italy
tania@ing.univaq.it

Abstract. Business Process modeling is constantly acquiring attention in modern enterprises. Today, BP editor tools support modelers in building correct diagrams only from the syntactic point of view. Enriching them with ontologies may bring many advantages, that go from the possibility of applying advanced reasoning techniques, aimed at the identification of contradictions and mistakes in the model specification, to the possibility of organizing BP models repositories, with advanced search and retrieval facilities. Finally, semantic technologies can substantially help the solution of the Business/IT alignment problem. Semantic enrichment of a BP can be achieved by representing a BP, or part of it, with an ontology-oriented formalism (semantic lifting) and mapping it to a reference ontology. In this paper, we present the basic elements of a Business Process Ontology framework (OPAL+BPAL) and its concrete representation according to an OWL syntax. Finally, we show how it is possible to generate an OWL representation of a BPMN diagram.

Keywords: business process, ontology, semantic alignment, bp design.

1 Introduction

Traditionally, Business Process Management (BPM) has been perceived as divided into two distinct levels:

- **BPM as a management discipline** that supports business organizations in standardizing and continuously optimizing operational processes that have a large impact on achieving corporate performance goals;
- **BPM as a technology** (for software production) providing IT organizations with a set of tools to model, deploy and execute processes that include human and system tasks (as, e.g., workflows) or that span across different business applications and require a broad set of integration capabilities (as, e.g., messaging, transformation, adapter technology; known as EAI – Enterprise Application Integration.)

Both cases represent challenging activities that require highly skilled Business and IT experts, respectively. However, today in the two areas, the experts operate without a systematic interaction and cooperation, causing the well known problem of Business/IT alignment. In fact, one key problem is the alignment of different tools,

methods and sometimes even jargons used by the two communities (business and IT experts).

To reduce the gap between these two areas, BISOGNO DI UN SUPPORTO SOCIALE

ONTOLOGIA COME SUPPORTO SOCIALE in quanto prodotto risultante da una collaborazione di un gruppo multidisciplinare e mirato al raggiungimento di un consenso condiviso.

In our contest, ontology-based semantic technologies appear to be a valid option. In fact, such technologies are based on the usage of reference ontologies, which allow a social playground for the two expert communities to be built.

With respect to the IT community, the application of ontologies and semantics-based solutions is already a promising reality, since several initiatives are gaining consensus. They are mainly focused on web service discovery and composition (i.e., WSMO [17], SAWSDL [21], and OWL-S [16]). The business community is starting now to consider it.

In this paper we focus on the business level. We aim at supporting business process production, reengineering and maintenance with the adoption of SUPPORTO SOCIALE FORNITO DALL'ONTOLOGIA semantics-based technologies. The preconditions to our solution are the existence of a business ontology (e.g., [EO], [TOVE], [PHMIT]) gathering structural and procedural knowledge of an enterprise, and the possibility to represent a business process into an ontology language. In particular, in this paper we concentrate on this last aspect, referred to as *semantic lifting*, aiming at ALLINEARE UN DIAGRAMMA/PROCESSO BPMN RISPETTO AD UNA RAPPRESENTAZIONE ONTOLOGICA transforming a business process, modeled using BPMN (Business Process Modeling Notation) [2], in OWPAL. OWPAL is the OWL representation of the OPAL+BPAL [5] [3] ontology modeling framework.

In details, the envisaged advantages of the *semantic lifting* (SEMANTIC ALIGNMENT) are listed below.

- Support to *business process design* by verifying semantic alignment of a business process with respect to a reference ontology. The semantic alignment can be achieved by performing consistency checking through the use of a reasoning engine, like RACER or PELLET.
- Support to *business process reengineering* providing suggestions to business experts during the design phase of a BP. In fact a business expert can be supported in finding, for instance, alternative elements of a business process by performing semantic search and similarity reasoning over the business ontology.
- Support to *business process maintenance* by automatically checking the alignment between one of more business processes with the business ontology when the latter is modified. This provides strong benefits since, for instance, a change in the company organization, could affect many business processes that need to be manually checked. By using an ontology, the manual changes are limited to the business ontology since its alignment with business processes can be automatically verified by performing consistency checking.

Furthermore our proposal is not specific for a particular business process language or ontology language. In this paper we focus on BPMN and OWL but our approach can be easily extended to other languages.

The rest of the paper is organized as follows: In Sections 2 we present the related works. In Section 3 the main constructs of the BPMN modeling notation are presented. In Section 4, we focus on the OPAL+BPAL ontology modeling framework. In Section 5 the semantic lifting (SEMANTIC ALIGNMENT) from a BPMN diagram to OWPAL is described. Finally in Section 6 we present some conclusions.

2 Related works

In the last years there has been an intense research activity on methods for process ontologies. We here briefly report on: OWL-T, OWL-P, and oXPDL.
OWL-T [6] (T stands for Task) is a method for coding an ontology in OWL, expressing user demands (tasks) at a high-level of abstraction, without dealing with the technical details of the underlying infrastructure. The OWL-T meta-model is characterized by a hierarchy of task types: Atomic, Composite, Simple, Complex. Each task is described in terms of properties and components. In particular, functional properties allow inputs, outputs, preconditions, post-conditions, preferences and effects to be represented. For its characterization, OWL-T is particularly suitable for generation of executable processes. The OWL-T developers aim to transform tasks into executable processes by employing some automatic methods of service composition.
OWL-P [7] proposes an approach for business process modeling and enactment, based on a combination of protocols and policies. The key idea is to capture meaningful interactions as protocols. OWL-P is an ontology framework for protocols coded in the OWL. OWL-P describes concepts such as roles, messages exchanged between the roles, and declarative protocol rules. The main computational aspects of protocols are specified using the Semantic Web Rule language (SWRL) [8] for defining rules which allows implication rules over entities defined as OWL-P instances to be specified.
Both the two initiatives above are mainly concentrated on the generation of executable business processes and do not appear semantically aligned with BPMN as required for formal verification of the related BP.
oXPDL [9] is a process interchange ontology formalism based on the standardized XML Process Definition Language (XPDL) [10]. oXPDL explicitly represents the semantics of a process model defined according to XPDL in a Web ontology language. oXPDL also focuses on reusing and integration of existing standard and ontologies such as SUMO [11], eClassOWL [12], RosettaNet [13] and PSL [14]. With respect to OPAL+BPAL, oXPDL, as a process interchange formalism, is mainly for interoperability issues, e.g., integration of different BP management tools, than to actually support BP design activities.

3 Business Process modeling with BPMN

BPMN aims at standardizing business process modeling notations in order to simplify the process organization, and the communication among business users, customers, suppliers, and process implementers. To achieve this result is introduced a graphical notation with an intuitive semantics dedicated to business process modeling, and some basic syntactic rules, e.g., sequence and message flow rules.
Figure 1 shows the basic categories of BPMN constructs: flow objects, swim-lanes and connecting objects. In this section, we define the BPMN modeling constructs, categorized according to the BPMN Specification (see [2] for details), involved in the BPMN example process. We use this example as key reading of the transformation from BPMN schemas to OWPAL formalism in order to align

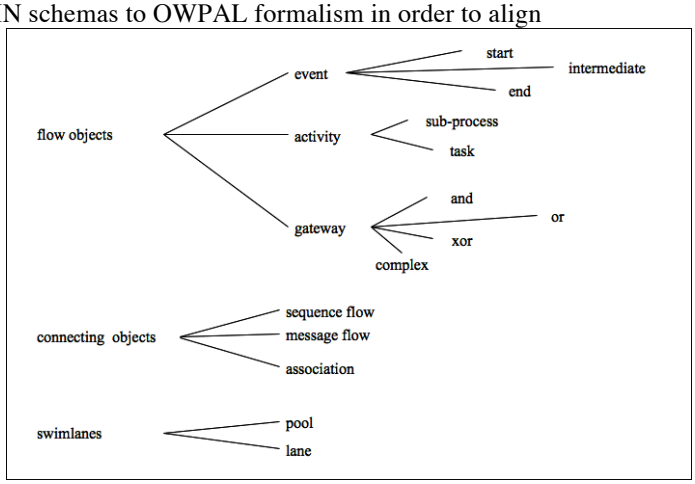


Figure 1: Hierarchy of BPMN modelling constructs.

It is worth noting that, from a graphical point of view, a *process* is not a graphical object. It is a composition of graphical objects representing BPMN constructs.

Flow Objects represent the business process behavior by using *event*, *activity*, and *gateway*. An *event* is something that "happens" during the course of a business process. An *activity* is a generic term representing the work performed within a company. In BPMN, an activity can be atomic or compound. In particular, BPMN activities are *sub-process* and *task*. A *gateway* is a modeling construct used to represent the interaction among different sequence flows; moreover they diverge and/or converge within a process. When sequence flows arrive at gateway, they can be merged together on input and/or split apart on output. Gateways can define all the types of business process sequence flow behaviour: *decisions/branching, merging, forking and joining*. In order to depict these behaviours, the BPMN uses the following gateway: *exclusive decision/merge XOR, inclusive decision/merge OR*, *parallel fork/join AND,* and *complex decision/merge*.
Swimlanes are used to model any relevant entity, able to activate or perform a process. Swimlanes aggregate organization or specific units, they group the

corresponding BP modeling constructs: pool and lane that allow partitioning activities according to performers.
Connecting Objects represent the possible connecting ways from flow objects to each other or to other information. Main connecting objects are *sequence flow*, *message flow* and *association*. The *sequence flow* is used to show the process activities order. Their source and target must be events, activities or gateways. The *message flow* is used to show the flow of messages between two participants (two different pools) of a process.

In order to illustrate how the BPMN diagram is transformed in OWPAL, we present (see Figure 2) the BPMN process example (built with the Intalio Editor Tool [4]). This process deals with a procurement scenario; the two pools are two different organizations performing the roles of Buyer and Supplier. The Buyer sends the Request for Quotation to the Supplier which replies sending back a Quotation. Afterwards, the Buyer analyzes the Quotation and, if satisfied, he/she sends the Purchase Order to the Supplier, otherwise, the Quotation is rejected. Invoicing from the Supplier and Payment from the Buyer concludes the process.

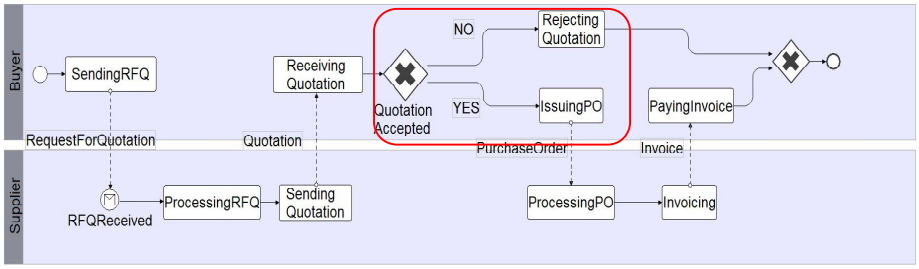


Figure 2: A Purchase Business Process realized with the Intalio Editor Tool.

4 An ontology-based framework for business modeling

As mentioned in the Section 1, our approach to validate BPMN diagram semantics entails a contrast between the BPMN diagrams in OWPAL format and the OPAL+BPAL ontology in the same OWPAL format. In this section we briefly describe the OPAL+BPAL ontology framework. In essence, OPAL provides designer with the constructs necessary to model a static view of the reality. Here also processes are modeled, but only from a structural point of view. BPAL is mainly concerned with operational aspects of processes. Starting from the OPAL knowledge, BPAL is able to model control and data flow in business processes.

4.1 OPAL: an ontology framework for structural modeling of business domains

OPAL (Object, Process, Actor modeling Language) [5] is an ontology framework aimed at supporting business experts in building a structural ontology. OPAL upper level concepts are functors (e.g., *Actor*), and arguments represent individual variables and constants (e.g., "Peter Smith"). OPAL atoms are unary and relational predicates. OPAL business ontology is built by defining concepts as specialization of unary predicates (e.g., *Organization isa Actor*) and relating the defined concepts by using relational predicates (e.g., *Employee works_in Organization*). In the following, the main OPAL predicates are presented.

Unary predicates (u_pre)

- *proc(_pr) - process; it* aims at modelling atomic or structured activities;
- *ar(_ar) - business actor*; it aims at modelling any relevant entity of the domain able to activate, monitor, or perform processes;
- *obj(_obj)* - a *business object*; it aims at modelling a passive entity, on which a *process* operates;
- *bod(_bod)* – a *business object document*; it is a specialization of *business object*;
- *aa(_aa) - atomic attribute*, for example *Street Name*;
- *ca(_ca) - complex attribute*; for example *Address*. It is defined as an aggregation of lower level *complex attributes* and/or *atomic attributes*.

Relational predicates

- *isa(_upre1[1], _upre2)* - *specialization* relation; it allows building specialization hierarchy, supporting a top-down refinement when applied to unary predicates (*_upre1* is a specialization of *_upre2*);
- *pof(_upre1, _upre2)* - *part-of* relation; it allows a top-down decomposition of concepts when applied to unary predicates (*_upre1* is part of *_upre2*);
- *ndrel(_upre, _upre, _relName)* - *named association*; it is defined between two unary predicates;
- *udrel(_upre, _upre)* - *unnamed association*; it is defined between two unary predicates;
- *pred(ca| aa, ar|obj)* - *association*; it is defined between an attribute and a concept.

OPAL example for the *purchasing* application

In the following we present a portion from the OPAL example about *purchasing* application (see the boxed fragment in Figure 1):

```
Organization(_x), ar(_x), isa(Organization, ar),
RejectingQuotation(_y), proc(_y), isa(RejectingQuotation, proc)
IssuingPO(_z), proc(_z), isa(IssuingPO, proc).
```

4.2 BPAL: an abstract language for BP Ontologies

The BPAL atoms [3] are predicates where functors represent ontological categories, and where arguments are typed variables representing concepts in the Core Business Ontology (CBO), built according to the OPAL framework [5]. In the following, we analyze in detail BPAL predicates.

Unary predicates

The BPAL unary predicates are:

- *bp(_bp)* - a business process;
- *bact(_act)* - a business activity, element of business process;
- *role(_rl)* - a business actor, involved with a given role in one or more activities;
- *dec(_bexp)* - a generic decision point; its argument is a boolean expression evaluated to {true, false}; it is used in the preliminary design phases when developing a BP with a stepwise refinement approach; in later phases, it will be substituted with one of the specific decision predicates (see below);

Relational predicates

The BPAL relational predicates are:

- *ev(_ev,_time,_bexp)* - an event; its arguments are a boolean expression and a timestamp;
- *msg(_msg,_obj,_source,_dest)* - a message, characterized by the content (*_obj*), a sending activity (*_source*), and a receiving activity (*_dest*);

[1] *_upre*. is a variable that refers to the generic unary predicate.

- *cxt(_listObj,_rl)* - a context, represented by a collection of information structures, it is related to a role;
- *adec(_dec,_bexp)*, *odec(_dec,_bexp)* - decision points representing a branching in the sequence flow, where the following paths will be executed in parallel or in alternative, respectively
- *xdec(_dec,_bexp,_act)* - a decision where the business activity *_act* will receive the control, depending if the boolean expression *_bexp* is true.
- *sort(_x, _y)* - "a sort of", it represent a bridge between OPAL and BPAL; if *_x* is a BPAL role, then *_y* is an OPAL actor; if *_x* is a BPAL activity or business process, then, *_y* is an OPAL process;
- *prec(_act|_dec|_ev,_act|_dec|_ev)* - a precedence relation among activities, decisions, events;
- *coop(_rl1,_rl2, …,_rln,_act)* - roles involved in performed activities;
- *perf*(*_rl*,*_act*) - a relation indicating which role is dedicated to which activity
- *gen*(*_act*,*_obj*), *upd*(*_act*,*_obj*), *arc*(*_act*,*_obj*) - indicating the business activity *_act* that can create, manipulate, archive the object *_obj*.

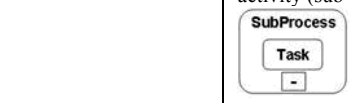
BPAL example for the *purchasing* application
In the following we present the BPAL example of the excerpt from the *purchasing* business process:

```
Buyer(_x), role(_x), isa(Buyer, role), sort(Buyer, Organization)
IssuingPO1(_x), bact(_x), isa(IssuingPO1, bact), sort(IssuingPO1,
    IssuingPO)
RejectingQuotation1(_x), bact(_x), sort(RejectingQuotation1,
    RejectingQuotion)
QuotationAccepted(_z, "YES",_IssuingPO), isa(QuotationAccepted, xdec)
prec(QuotationAccepted, RejectingQuotation)
prec(QuotationAccepted, IssuingPO)
perfBy(Buyer, RejectingQuotation)
perfBy(Buyer, IssuingPO)
```

5 Semantic lifting of BPMN constructs

In this section we analyze the main constructs of BPMN, providing for them a formal semantics based on the OPAL+BPAL ontology framework. This is referred to as *semantic lifting*. Table 1 presents an overview of the proposed semantic lifting of the main constructs of BPMN in terms of the OPAL+BPAL predicates. To actually build an ontology of a BP modelled in BPMN, we also need a concrete syntax. To this end, an OWL representation of the abstract OPAL+BPAL predicates is given. Such an OWL representation, named OWPAL, is reported in accordance with the N3 notation [18], which results more concise respect to the XML syntax of OWL. For the OWPAL representation, it is assumed that two namespaces, `opal` and `bpal`, exist where OPAL and BPAL keywords have been defined. The `xsd` namespace is where the XMLSchema types are defined.

Table 1: Semantic lifting of main BPMN constructs

BPMN Constructs	OPAL	BPAL	OWPAL (N3 notation)
Flow objects			
Process *e.g.: Purchasing*	*Purchasing(_x)* *proc(_x)* *isa(* *Purchasing,* *proc)*	*Purchasing1(_y)* *bp(_y)* *isa(* *Purchasing1, bp)* *sort(* *Purchasing1,* *Purchasing* *)*	*Purchasing* a owl:Class; rdfs:subClassOf opal:PROCESS *Purchasing1* a owl:Class; rdfs:subClassOf bpal:BUSINESS_PROCESS rdfs:subClassOf [a owl:Restriction; owl:onProperty bpal:SORT; owl:allValuesFrom *Purchasing*]
activity (task)  *e.g. IssuingPO* activity (sub-process)  	*IssuingPO(_x)* *proc(_x)* *isa(* *IssuingPO,* *proc)*	*IssuingPO1(_y)* *bact(_y)* *isa(* *IssuingPO1, bact)* *sort(* *IssuingPO1,* *IssuingPO* *)*	*IssuingPO* a owl:Class; rdfs:subClassOf opal: PROCESS *IssuingPO1* a owl:Class; rdfs:subClassOf bpal: BUSINESS_ACTIVITY rdfs:subClassOf [a owl:Restriction; owl:onProperty bpal:SORT; owl:allValuesFrom *IssuingPO*]
gateway (XOR/OR/AND) 		*dec(_dec,_bexp),* *adec(_adec,_bexp),* *odec(_odec,_bexp),* *xdec(_xdec_bexp,* *_act)* *QuotationAccepted(_* *xdec, "YES",* *bact(IssuingPO1))* *isa(QuotationAccepte* *d, xdec)*	*QuotationAccepted* a owl:Class; rdfs:subClassOf bpal:XDEC rdfs:subClassOf [a owl:Restriction; owl:onProperty bpal:BEXP; owl:allValuesFrom [a owl:Class owl:oneOf (:*YES*)]] rdfs:subClassOf [a owl:Restriction; owl:onProperty bpal:IFTRUE; owl:allValuesFrom *IssuingPO1*]
Connecting Objects			
sequence flow E.g., *Invoicing precedes PayingInvoice*		*prec(_act|_dec| _ev,* *_act|_dec|_ev)* *prec(Invoicing1,* *PayingInvoice1)*	*_PayingInvoice* a owl:Class rdfs:subClassOf bpal:BUSINESS_ACTIVITY *_Invoicing* a owl:Class rdfs:subClassOf bpal:BUSINESS_ACTIVITY rdfs:subClassOf [a owl:Restriction owl:onProperty bpal:PREC owl:allValuesFrom *PayingInvoice*]
Swimlanes			
pool  E.g. *Organization*	*Organization(* *_x)* *ar(_x)* *isa(* *Purchasing,* *ar)*	*Buyer(_y)* *role(_y)* *isa(* *Buyer, role)* *sort(* *Buyer,*	*Organization* a owl:Class; rdfs:subClassOf opal:BUSINESS_ACTOR *Buyer* a owl:Class; rdfs:subClassOf bpal: ROLE rdfs:subClassOf [a owl:Restriction; owl:onProperty bpal:SORT;

		Organization)	owl:allValuesFrom *Organization*]

According to the table above, using the OWL/N3 syntax, a fragment from the example extracted by the Figure 2 is now represented.

As Anticipated, this code can be enriched with the ontology knowledge (including axioms) and fed to a reasoner to check the semantic validity

6 Conclusions

In this paper, the semantic lifting of BPMN diagrams to OWPAL has been presented. OWPAL is the OWL representation of the OPAL+BPAL ontology framework. The lifting aims at representing a BPMN diagram by using an ontology-based formalism in order to use ontology reasoning techniques for several activities as, for example, business process design and consistency checking, automatic BP maintainance.
In particular, in the paper, the BPMN core construct and OPAL+BPAL ontology framework are presented and used to semantic enrich BPMN constructs.
The proposed formalization uses predicates and Horn Logic to compactly represent them introducing OWPAL, a homogeneous OWL representation with the BP coding. Finally the BPMN example process is introduced in order to well appreciate the semantic enrichment from the BPMN model to OWPAL format.

References

1. T. R. Gruber. A translation approach to portable ontologies. Knowledge Acquisition, 5(2):199-220, 1993.
2. Object Management Group. Business Process Modeling Notation Specification. Version 1.0. February 2006 [www.bpmn.org/Documents/OMG%20Final%20Adopted %20BPMN%2010%20Spec%2006-02-01.pdf]
3. A. De Nicola, M. Lezoche, M. Missikoff. An Ontological Approach to Business Process Modeling, IICAI-07. Pune, India, 17th-19th December, 2007.
4. Intalio Business Process Management Suite [http://bpms.intalio.com/]
5. F. D'Antonio, M. Missikoff, F. Taglino. Formalizing the OPAL eBusiness ontology de-sign patterns with OWL. I-ESA Conference 2007.
6. V. X. Tran, H. Tsuji. OWL-T: A Task Ontology Language for Automatic Service Composition, ICWS 07, pages 1164-1167, IEEE Computer Society.
7 N. Desai, A.U. Mallya, A.K. Chopra, M.P. Singh. OWL-P: A methodology for Business Process development. In proceeding of AOIS 2005. pp 79-94.
8 I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML, May, 2004 (W3C Submission). http://www.w3.org/Submission/2004/SUBMSWRL-20040521/
9 A. Haller, M. Marmolowski, E. Oren, W. Gaaloul. oXPDL: a Process Model Exchange Ontology. Technical report at DERI-TR-2007-12-12, 2007.

10 The Workflow Management Coalition. Workflow Standard Process Definition Interface – XML Process Definition Language. 2005.
11 I. Niles, A. Pease. Towards a standard upper ontology. In Proceedings of the International Conference on Formal Ontology in Information Systems, pp. 2–9. 2001.
12 M. Hepp. Products and services ontologies: A methodology for deriving owl ontologies from industrial categorization standards. International Journal on Semantic Web & Information Systems, 2(1):72–99, 2006.
13 A. Haller, J. Gontarczyk, and P. Kotinurmi. Towards a complete SCM Ontology - The Case of ontologising RosettaNet. In Proceedings of the 23rd ACM Symposium on Applied Computing. ACM, Fortaleza, Brazil, 2008.
14 M. Gruninger. Ontology of the process specification language. In S. Staab and R. Studer, (eds.) Handbook on Ontologies, pp. 575–592. Springer, 2004.
15 OWL. Web ontology language, Feb 2004. http://www.w3.org/TR/owl-features/.
16 OWL-S (Web Ontology Language for Web Service) 1.2 Release. Available from http://www.daml.org/services/owl-s/1.0/.
17 D. Roman, H. Lausen, and U. Keller. D2v1.3. Web Service Modelling Ontology (WSMO). Deliverable, http://www.wsmo.org/TR/d2/v1.3/, October 2006.
18 http://www.w3.org/DesignIssues/Notation3