Model-Driven Semantic Service Matchmaking for Collaborative Business Processes

Matthias Klusch, Stefan Nesbigall, and Ingo Zinnikus

German Research Center for Artificial Intelligence (DFKI),
Stuhlsatzenhausweg 3, Saarbrücken
(klusch|stefan.nesbigall|ingo.zinnikus)@dfki.de

Abstract. Business process modelling and execution in a collaborative environment requires a set of methodologies and tools which support the transition from an analysis to an execution level. Integrating the process with a pre-existing IT infrastructure leads to typical interoperability problems. Service-oriented architectures are today's favorite answer to solve these interoperability issues. To tackle them, the recent trend is to use the principles of model driven-design. In this paper, we apply these principles to Semantic Web service technology to assist a business orchestrator finding suitable services at design time, and composing workflows for agent-based execution. We describe a formal approach to preserve the content of the semantic annotations in the model and code transformations.

1 Introduction

Service-oriented architectures (SOA) are today's favorite answer to realize the vision of seamless business interaction across organizational boundaries. It enables enterprises to offer selected functionalities of their business systems via standardized XML-based Web service interfaces (written in WSDL [5]). Complex business application processes can be implemented through appropriate Web service compositions in prior or on-demand each of which functionality is made available to the customer at the respective enterprise portal in the Web. The SOA principle provides a loosely coupled and standardized modular solution to enterprise business application landscapes. One recent trend of developing SOAs is to apply the principles of model-driven software development (MDD) by (i) modelling the overall business process workflows in a more abstract manner, and (ii) providing model transformations that define mappings between the abstract specification and the underlying platform-specific systems. According to [14], business process modelling and execution is commonly performed in a top-down fashion. Since existing standard Web services lack formal semantics, from the point of view of strong AI, the meaningful integration of services realizing the desired business processes exclusively relies on human business domain experts at design time. In contrast, Semantic Web service technology adds expressivity to existing Web service standards by introducing well-formed semantics that simple Web service

descriptions are lacking, and envisages intelligent agents to discover and compose complex business services through logic-based reasoning upon their semantic annotations. However, in many real-world cases of business process modelling among contracted and trusted business partners, the fully automated coordination of partly unknown business Web services is neither adequate nor efficient in practice. When service composition is concerned the Semantic Web service approach can be compared to planning from first principles in AI while the model-driven approach can be compared to planning from second principles if the platform-specific engine for executing the models is powerful enough. In this sense, both approaches model-driven process development (MDD) and Semantic Web services (SWS) have their pros and cons when used to integrate external, outsourced business services in SOAs. In the spirit of the model-driven approach, we introduce a metamodel for Semantic Web services, called PIM4SWS, which is an abstraction from most commonly used SWS description languages or so-called platform-specific models, that are OWL-S [18], WSML [26] and standard SAWSDL [22]. That renders semantic service selection and composition for implementing business process workflows in SOAs independent of these models. In particular, we envisage a model-driven Semantic Web service matchmaker, called MDSM (Model-Driven Service Matchmaker), to support human business domain experts and service orchestrators in finding suitable services for this purpose at design time. As a consequence, these experts only need knowledge about the common UML-based metamodel PIM4SWS but not the specific models like OWL-S, WSML or SAWSDL used by different business service developers to describe the semantics of their individual services that are potentiall relevant for implementing the collaborative business process workflow. Syntactic mapping from a metamodel in UML to parts of these specific models are proposed in [16, 11, 1] but without any formal grounding of their transformations. [21] provides a comparison between concepts in OWL-S and WSML. In contrast, we propose to use the formal specification language Z [23], respectively, Object-Z [8] as a common language for provably correct transformations between different SWS models.

The remainder of this paper is structured as follows. We outline the MDSM matchmaking process in section 2. In section 3 we describe the transformation of the service request from the platform-independent to the platform-specific level. Section 4 gives an example of the whole matchmaking process of MDSM, while section 5 concludes the paper.

2 MDSM Overview

The MDSM matchmaker is capable of automated, model-independent semantic service selection to assist business service orchestrators in finding suitable services to realize parts of collaborative business processes as adequate service orchestrations at design time. Consider, for example, the modelling of a complex travel planning process as depicted in figure 1. At a certain point of choice in the planning process the human user, that is the business orchestrator, needs

to select a flight booking Web service to realize the respective booking process in the overall workflow of travel planning. For this purpose, the orchestrator models her Web service request in the common metamodel she is familiar with only, that is the platform-independent metamodel PIM4SWS.

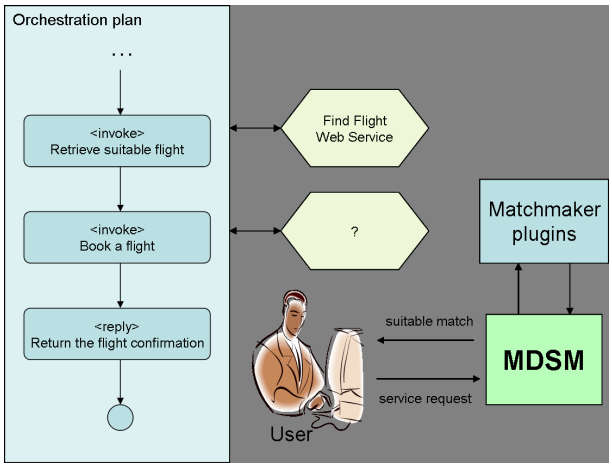

Fig. 1. Orchestration plan

The MDSM, in its first implementation, automatically transforms this request to semantically equivalent service requests in OWL-S, WSML and SAWSDL, and then issues them to relevant platform-specific matchmakers, that are for the implementation of MDSM 1.0, the matchmakers OWLS-MX [13], WSML-MX [12] and SAWSDL-MX. Eventually the MDSM provides the orchestrator with an aggregated and ranked answer set of relevant semantic services [3] together with their grounding in WSDL for invocation (cf. Fig. 2). The retransformation of platform-specific services to the common metamodel PIM4SWS by the MDSM is optional. One crucial step of this model-driven semantic service selection by the MDSM are the semantically equivalent model transformations which we discuss in the following section.

3 Transformation

In order to correctly compile a given service request in PIM4SWS to different platform-specific representations such as OWL-S and WSML, we differentiate between (a) structural transformation of the semantic service description as a whole, and (b) the semantic mapping between corresponding components of the information model of PIM4SWS such as its input, output, preconditions and effects that are described in specific ontology and rule languages like OWL [19], SWRL and WSML [27]. While structural transformations of PIM4SWS

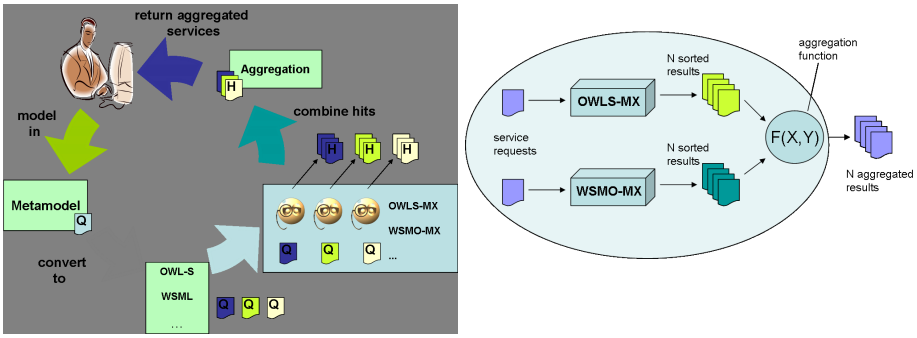

Fig. 2. Overview: Model-driven Semantic Web service matchmaking by the MDSM. (left); Aggregation of the answer sets of platform-specific service matchmakers. (right)

representations in UML to OWL-S and WSML are defined in terms of syntactic mappings between corresponding modelling concepts, we use the standardized formal specification language Z for the latter purpose (cf. Fig. 3).

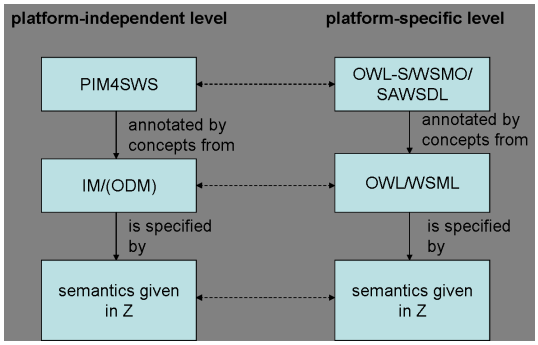

Fig. 3. Overview: Comparison of semantic services across platforms.

3.1 Structural Transformation

The metamodel PIM4SWS is designed as a core model to cover the common parts of the underlying semantic service description languages. It consists of three parts: *InformationModel* (IM; ODMNameSpace), *BlackBox* and *GlasBox* (cf. Fig. 4. The information model is the set service related ontologies described in the standard metamodel ODM [4, 9]) (also called ODMNameSpace), while both its functionality (*Functionals*) in terms of service signature, that are input and output parameters, and specification, that are preconditions and effects, and non-functional parameters (*NonFunctionals*) such as price, service name

and developer are described in the service profile or *BlackBox*. The *GlasBox* includes the description of the internal service process and is not considered in this paper.

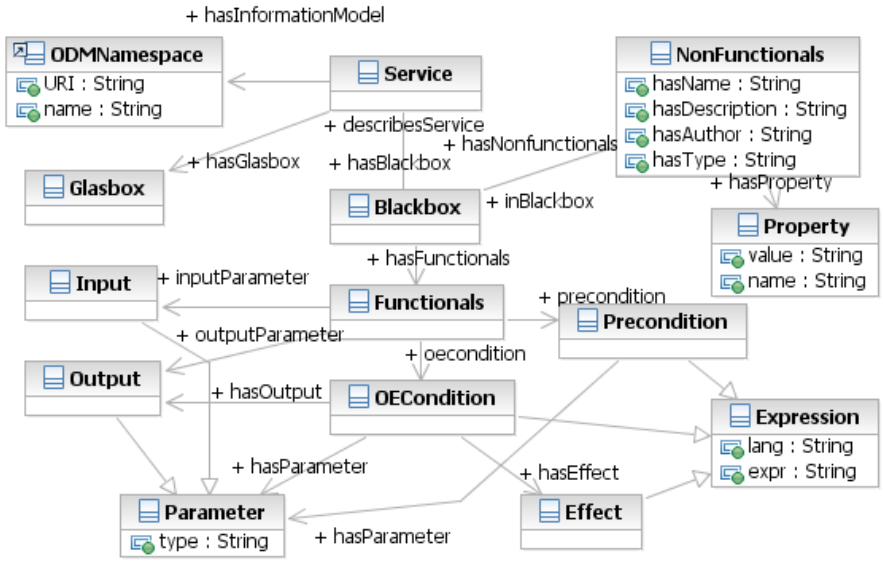

Fig. 4. Platform-independent metamodel for semantic Web services.

We acknowledge that the PIM4SWS metamodel is similar to the OWL-S model which can be to a large extent embedded into the WSML model[15] - which makes the structural transformations from PIM4SWS to both specific models or platforms straight-forward. In particular, the OWL-S service profile is generated from the *Functionals* and *NonFunctionals* of the given PIM4SWS service description, the OWL-S process model for atomic processes is given by the *Functionals* where the "hasResult" construct of the OWL-S service is extracted from the *OECondition*. For structural transformations from PIM4SWS to WSML, the following holds: (a) the *Service* class is related to any service component in WSML; (b) the *NonFunctionals* class is covered by annotations and non-functional properties of the considered service component; and (c) the *Functionals* class is mapped to the capability of the service. Since in PIM4SWS inputs and outputs describe information between a service provider and the requester, these classes are related to pre- and postcondition concepts in WSML. Furthermore, we map preconditions to assumptions. The WSML service result construct is resolved by an implication in the postcondition and effect axiom of WSML, where the antecedent is the semantically transformed *OECondition* expression, and the consequent is the transformed *hasEffect* expression for an effect, and the

transformed *hasOutput* for a postcondition in WSML. Each parameter is handled by initializing shared WSML variables. Due to space limitations, we omit further details of the structural transformation from PIM4SWS to WSML and SAWSDL, and rather focus on the semantic mapping between the PIM4SWS information model (in ODM) and different ontology languages (platforms) used for semantic annotation.

3.2 Semantic Transformation using Z

To achieve a verifiably correct mapping between the different DL-based ontology languages used for semantic annotation such as OWL-DL and WSML-DL in our case, we use the formal standard specification language Z as a common basis[1]. Based on [4, 9], our initial version of the information model IM of the PIM4SWS is the description logic *SHIN(D)* related part of the metamodel ODM written in UML; the metamodel of the PIM4SWS-IM is given in [4]. *SHIN(D)* is the intersection of the description logics underlying OWL-DL and WSML-DL, that are *SHOIN(D)*, respectively, *SHIQ(D)*. As such the (PIM4SWS-)IM does not support enumerated classes with nominals (O) nor qualified role cardinality restrictions (Q) for semantic annotations: While WSML-DL does not support the former, the latter cannot fully be covered by OWL-DL [20]. The standard for semantic Web services, SAWSDL, does not provide any specific ontology language, hence, for SAWSDL services, we assume model references to ontologies in OWL-DL and WSML-DL. As a consequence, each platform-specific matchmaker called by the model-driven MDSM matchmaker with service requests in PIM4SWS with annotations in *SHIN(D)* (subsumed by both *SHOIN(D)* and *SHIQ(D)*) is able to match these against any service in OWL-S, WSML and SAWSDL with annotations in OWL-DL or WSML-DL.

Why then using Z? In principle, the information model of the PIM4SWS is not restricted to our initial choice of a description logic (*SHIN(D)*) but shall cover different ontology and rule languages (in different notations) with first-order logic (FOL) semantics. For this purpose, we suggest to use the ISO standard specification language Z (semantically equivalent to FOL) as a common language for specifying semantic annotations of service requests in PIM4SWS by the orchestrator. Please note that the semantic equivalence of PIM4SWS service request annotations in *SHIN(D)* we proposed for our intial version of the PIM4SWS-IM with those in OWL-DL and WSML-DL of the request in OWL-S and WSML compiled by the MDSM matchmaker is trivial: It holds per definition of the PIM4SWS-IM as intersection of OWL-DL and WSML-DL both assumed as sole ontology languages for semantic annotations of service requests in PIM4SWS, OWL-S and WSML. In general, testing the semantic equivalence of pairs of platform-independent and platform-dependent semantic service request

[1] Z is based on Zermelo-Fraenkel set theory and first-order predicate logic. It is widely used by industry for system behaviour specification and verification of properties, and has undergone international ISO standardization. Various tools for formatting, type-checking and aiding proofs in Z are available, e.g. see http://vl.zuser.org/.

annotations in different first-order logic-based ontology languages in different syntactic representation is to convert them into equivalent FOL expressions and use a FOL theorem prover for checking the satisfiability of their mutual logic implication. This semantic equivalence can also be shown using Z as common specification language as shown in figure 5.

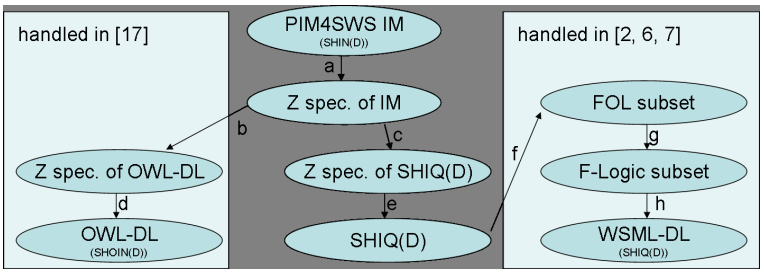

Fig. 5. Semantically equivalent compilation of annotations in PIM4SWS-IM to OWL-DL and WSML-DL using Z.

In particular, having a semantic annotation in the PIM4SWS-IM, we provide its transformation to Z (step a), which corresponds to the *SHIN(D)* related part of the specification of OWL-DL, respectively *SHOIN(D)*, in Z (steps b and d, as reported in [17] with proof of correctness and completeness). Likewise, the specification of the PIM4SWS service request annotation in Z corresponds to the *SHIN(D)* related part of the specification of WSML-DL, respectively *SHIQ(D)* in Z (steps c and d), which, in turn, is a FOL subset (step f, [2]) that can be written in F-Logic (step g) and as such syntactically transformed to an (WSML-DL equivalent) annotation of a WSML service request (step h, as reported in [7, 6]). Due to space limitations, we omit the full specification of the initial version of the PIM4SWS-IM, OWL-DL and WSML-DL in Z but a rough sketch only and show that the given transformations of IM and OWL-DL to Z are semantically equivalent according to the (FOL bases) Z semantics. The latter inherently holds since the IM is defined to be a subset of OWL-DL (and WSML-DL), but the principle of proving the semantic equivalence of a given pair of Z transformations is useful to apply also for cases where this is not the case, e.g., when it is not clear whether additional IM elements can be emulated by those of targeted platform-specific languages.

The universal signature of the (PIM4SWS-)IM is the set of OntologyElements with the interpretation $I_{IM} = (\mathit{IMIndividual}, \cdot^{I_{IM}})$ where IMIndividual is the domain of discourse Δ (according to the set-theoretic first-order semantics of description logics). The function $\cdot^{I_{IM}}$ maps an IMClass c to the subset of the domain (classInstances(c)) and an IMProperty p to a tuple (subject(propVals(p)), object(propVals(p))). The semantics of the initial IM, that is *SHIN(D)*, is then equivalently specified by the following Z axioms.

$[\mathit{OntologyElement}]$

An IMIndividual is modelled as a subset of OntologyElement. IMClass is another subset of OntologyElement disjoint from IMIndividual. The function classInstances maps an IMClass to the IMIndividual set of their instances.

$$
\begin{array}{|l}
IMIndividual, IMClass : \mathbb{P}\, OntologyElement \\
classInstances : IMClass \rightarrow \mathbb{P}\, IMIndividual \\
\hline
IMClass \cap IMIndividual = \varnothing
\end{array}
$$

IMProperty and IMPropertyValue are again subsets of OntologyElement and disjoint to each other and to the above subsets. An instance of an IMProperty is an IMPropertyValue. The function propVals maps an IMProperty to its set of instances.

$$
\begin{array}{|l}
IMProperty, IMPropertyValue : \mathbb{P}\, OntologyElement \\
propVals : IMProperty \rightarrow \mathbb{P}\, IMPropertyValue \\
\hline
IMProperty \cap IMClass = \varnothing \\
IMProperty \cap IMIndividual = \varnothing \\
IMPropertyValue \cap IMClass = \varnothing \\
IMPropertyValue \cap IMIndividual = \varnothing \\
IMPropertyValue \cap IMProperty = \varnothing
\end{array}
$$

Every IMPropertyValue has unary relations, subject and object, that returns two IMIndividuals which are related by the property.

$$
\begin{array}{|l}
subject : IMPropertyValue \leftrightarrow IMIndividual \\
object : IMPropertyValue \leftrightarrow IMIndividual
\end{array}
$$

To express a class hierarchy in the metamodel generalizations are used. A generalization relates two classes, where the first class is a subclass of the second.

$$
\begin{array}{|l}
imSubClassOf : IMClass \leftrightarrow IMClass \\
\hline
\forall\, c_1, c_2 : IMClass \bullet imSubClassOf(c_1) = c_2 \\
\Leftrightarrow classInstances(c_1) \subseteq classInstances(c_2)
\end{array}
$$

Universal restricted classes are a subset of an IMClass that are universal restricted to a target IMClass by a given IMProperty.

$$
\begin{array}{|l}
UniRestriction : \mathbb{P}\, IMClass \\
onProperty : UniRestriction \leftrightarrow IMProperty \\
toClass : UniRestriction \leftrightarrow IMClass \\
\hline
\forall\, r : UniRestriction;\ a_1, a_2 : IMIndividual \bullet a_1 \in classInstances(r) \\
\Leftrightarrow (\exists\, v : IMPropertyValue \mid v \in propVals(onProperty(r)) \bullet \\
(subject(v) = a_1 \wedge object(v) = a_2) \Rightarrow a_2 \in classInstances(toClass(r)))
\end{array}
$$

For the Z-specification of semantic annotations in OWL-DL, we refer to [17]. In the following, variables in platform-specific specifications are marked with a prime like x', y'. The interpretation of the Z-specification of OWL-DL expressions is defined as $I_{OWL} = (OWLIndividual, \cdot^{I_{OWL}})$ with the domain OWLIndividual and the interpretation function $\cdot^{I_{OWL}}$ mapping an OWLClass c' to a subset

(instances(c')) of the domain and an OWLProperty p' to a tuple (subVal(p')). The corresponding Z axioms for OWL-DL restricted to the IM specification in Z are as follows.

$[Resource]$

$$OWLIndividual, OWLClass, OWLProperty : \mathbb{P}\, Resource$$

$$OWLIndividual \cap OWLClass = \varnothing$$
$$OWLProperty \cap OWLClass = \varnothing$$
$$OWLProperty \cap OWLIndividual = \varnothing$$

$$instances : OWLClass \rightarrow \mathbb{P}\, OWLIndividual$$
$$subVal : OWLProperty \rightarrow (Resource \leftrightarrow Resource)$$

$$subClassOf : OWLClass \leftrightarrow OWLClass$$

$$\forall c_1', c_2' : OWLClass \bullet$$
$$subClassOf(c_1') = c_2' \Leftrightarrow$$
$$instances(c_1') \subseteq instances(c_2')$$

$$allValuesFrom : (OWLClass \times OWLProperty) \leftrightarrow OWLClass$$

$$\forall c_1' : OWLClass;\ p' : OWLProperty;\ c_2' : OWLClass \bullet$$
$$allValuesFrom(c_1', p') = c_2' \Leftrightarrow (\forall a_1', a_2' : OWLIndividual \bullet$$
$$a_1' \in instances(c_2') \Leftrightarrow ((a_1', a_2') \in subVal(p') \Rightarrow a_2' \in instances(c_1')))$$

Specifying *SHIQ(D)* of WSML-DL for its subset *SHIN(D)* of the IM in Z is the same as we showed for OWL-DL above. That allows to compare the respective elements of WSML-DL and OWL-DL by looking at their representation in Z (with differently renamed elements for better distinction between them) as shown in table 1. The semantically equivalent transformation from WSML-DL to the corresponding F-Logic fragment is given in [6, 7] which means that the Z specification of IM annotations in *SHIN(D)* can be equivalently transformed to F-Logic used to describe semantic annotations (concepts and constraints) in WSML services.

Table 1. Notation of some elements of WSML-DL and OWL-DL in Z

WSML-DL (*SHIQ(D)*) in Z	OWL-DL in Z	Remark
Δ^S	OWLIndividual	set of instances
AC	OWLClass	set of atomic concepts
AR	OWLProperty	set of atomic properties
$.^{I_S}$	instances / subVal	semantic mapping to the domain
subConcept	subClassOf	concept hierarchy
forall	allValuesFrom	universal quantifier

To verify whether a direct model transformation $t(D) = D'$ of a description D in the platform-independent model PIM4SWS-IM to a description D' in platform-specific model or ontology language is semantically correct, one can test whether the semantics of D and D' are equivalent in Z. We show this by example for a description D in IM and D' in OWL-DL both transformed to Z. In Zermelo-Fraenkel set theory, the axiom of extensionality defines the equality of two sets: $\forall A \forall B[\forall x(x \in A \Leftrightarrow x \in B) \Rightarrow A = B]$. Thus, descriptions D and D' are semantically equal ($sem(D) = sem(D')$) iff their interpretations in the domain are the same ($D^I = D'^I$). For reasons of comparability, the domains of discourse of considered ontology languages (models) have to be the same: $MIndividual = OWLIndividual = \Delta$. The element equality in Z is defined as follows: Let $x \in$ IMIndividual, $y' \in$ OWLIndividual/Δ, then elements x, y' are the same $eql(x, y') = true$ iff $sem(x) = sem(y')$. That allows us to compare the semantic equality of different language elements in Z as shown in table 2: Equality of facts (a), sets of instances (b), concepts (c), property values (d), sets of property values according to a given property (e), and properties (f). In fact, we can obtain the semantic equivalence between constructors of the description logics underlying the initial PIM4SWS-IM and those of OWL-DL, respectively WSML-DL denoted in Z. Due to space limitation, we provide only a selection of these Z-equality relations in the following.

Table 2. Equality of facts, concepts, and roles.

a)	**instance:** o
$x : IMIndividual$	$= y' : OWLIndividual/\Delta$
	$\Leftrightarrow eql(x, y')$
b)	**instances of class** C**:** C^I
$classInstances(x)$	$= instances(y')/y'^{I_S}$
	$\Leftrightarrow (\forall\, i \in classInstances(x) \exists\, i' \in instances(y')/y'^{I_S} \mid$
	$eql(i, i')) \wedge (\forall\, i' \in instances(y')/y'^{I_S}$
	$\exists\, i \in classInstances(x) \mid eql(i, i'))$
c)	**class** C**:** C
$x : IMClass$	$= y' : OWLClass/AC$
	$\Leftrightarrow classInstances(x) = instances(y')/y'^{I_S}$
d)	**role value:** $\langle o, o' \rangle$
$x : IMPropertyValue$	$= (a'_1, a'_2) : OWLIndividual/\Delta \times OWLIndividual/\Delta$
	$\Leftrightarrow eql(subject(x), a'_1) \wedge eql(object(x), a'_2)$
e)	**role values of** R**:** $\langle o, o' \rangle \in R^I$
$propVals(p)$	$= subVal(p')/p'^{I_S}$
	$\Leftrightarrow (\forall\, v \in propVals(p) \exists (a'_1, a'_2) \in subVal(p')/p'^{I_S} \mid$
	$v = (a'_1, a'_2)) \wedge (\forall (a'_1, a'_2) \in subVal(p')/p'^{I_S}$
	$\exists\, v \in propVals(p) \mid v = (a'_1, a'_2))$
f)	**role** R**:** R
$p : IMProperty$	$= p' : OWLProperty/AR$
	$\Leftrightarrow propVals(p) = subVal(p')/p'^{I_S}$

We use these elementary Z-equality relations recursively to prove (by structural induction) the semantic equality for any given DL axiom or expression. For example, consider the DL concept subsumption axiom $c_1 \sqsubseteq c_2$ for two concepts c_1, c_2. The equality of its description *imSubClassOf* (in PIM4SWS-IM) and *subClassOf* (in OWL-DL) can be shown by the equality of their transformation in Z. Let $c_1, c_2 \in$ IMClass, $c_1', c_2' \in$ OWLClass/AC, $c_3 \in$ UniRestriction, $p \in$ IMProperty, $p' \in$ OWLProperty with $c_1 = c_1'$, $c_2 = c_2'$, onProperty$(c_3) = p$, toClass$(c_3) = c_1$ and $p = p'$, then the following holds:

$$\begin{aligned} imSubClassOf(c_1) = c_2 &\Leftrightarrow classInstances(c_1) \subseteq classInstances(c_2) && (1)\\ &\Leftrightarrow instances(c_1') \subseteq instances(c_2') && (2)\\ &\Leftrightarrow subClassOf(c_1') = c_2' && (3) \end{aligned}$$

In line (2) we use the equality relation (b) in table 2 to translate the IM specification of imSubClassOf in Z to OWL-DL, which is the same as *subClassOf* in OWL-DL. Analogously, we provide the (Z-)equality relation for universal quantified role cardinality restrictions ($\forall R.C$ in DL syntax):

$$\begin{aligned} c_3 \Leftrightarrow \forall\, a_1, a_2 : IMIndividual \bullet\ & a_1 \in classInstances(c_3) \Leftrightarrow\\ & (\exists\, v : IMPropertyValue \mid v \in propVals(p) \bullet\\ & (subject(v) = a_1 \wedge object(v) = a_2) \Rightarrow a_2 \in classInstances(c_1) \end{aligned}$$

Thus, instances of c_3 are equal to instances of the allValuesFrom construct, and determined by the following expression:

$$\begin{aligned} classInstances(c_3) &\Leftrightarrow \{x : IMIndividual \mid \forall\, a_2 : IMIndividual \bullet\\ &\quad \exists\, v : IMPropertyValue \mid v \in propVals(p) \bullet (x = subject(v) \wedge\\ &\quad a_2 = object(v)) \Rightarrow a_2 \in classInstances(c_1)\}\\ &\Leftrightarrow \{x' : OWLIndividual \mid \forall\, a_2' : OWLIndividual \bullet\\ &\quad (x', a_2') \in subVal(p') \Rightarrow a_2' \in instances(c_1')\}\\ &\Leftrightarrow instances(allValuesFrom(c_1', p')) \end{aligned}$$

Based on these Z-equality relations, one can prove that the syntactic model transformation function ($t(D) = D'$) of the MDSM matchmaker from PIM4SWS-IM to OWL-DL and WSML-DL is semantically correct.

4 Example

In the following, we briefly illustrate the principle of model-driven service matchmaking by the MDSM matchmaker. Suppose that a business service orchestrator intends to integrate a flight-booking Web service into her business process implementation. The service shall book one ticket for a given flight and customer, and confirms the booking. This request is formulated in PIM4SWS by the orchestrator and passed to the MDSM which transforms the received request to specific description models, that are, in our case, OWL-S, WSML and SAWSDL.

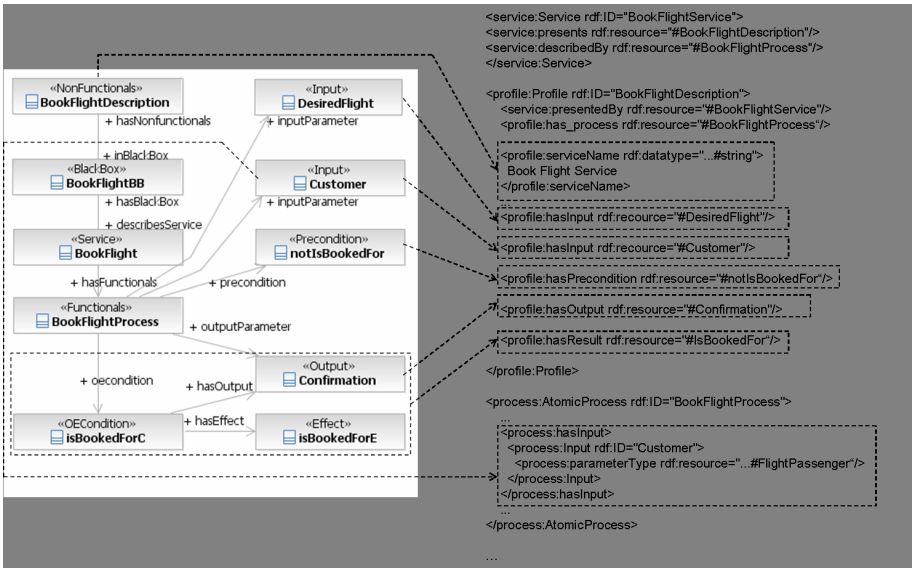

Fig. 6. Bookflight service request transformation from PIM4SWS to OWL-S.

The structural transformations to OWL-S and WSML are depicted in figures 6 and 7.

The semantic transformation of the request concerns all ontological concepts used to describe the service profile parameters (IOPE). We show this by example for the input concept Flight-Passenger (FP) which is described in the PIM4SWS-IM as shown in figure 8. In the following, we use the abbreviations P for Passenger, FP for FlightPassenger, AP for AirPlane, and V for Vehicle.

The MDSM transforms this description (D) directly into an equally named concept FP' described (D') in OWL-DL:

```
subClassOf(restriction travelsBy' allValuesFrom(AirPlane'), Passenger')
```

The semantic equivalence of this transformation ($t(D) = D'$) can be shown using Z as follows. Concept FP in the PIM4SWS-IM is specified in Z by

$$\frac{FP : UniRestriction}{imSubClassOf(FP) = P}$$

The denoted equality between concepts in this expression is checked by comparing their extensions: The set of instances of the UniRestriction class FP is given by the set of IMIndividual x which has to be a subset of the instances of concept P:

$$classInstances(FP) = \{x : IMIndividual \mid \forall\, a_2 : IMIndividual \bullet \exists\, v : IMPropertyValue \mid v \in propVals(travelsBy) \bullet x = subject(v) \land a_2 = object(v) \Rightarrow a_2 \in classInstances(AP)\} \subseteq classInstances(P)$$

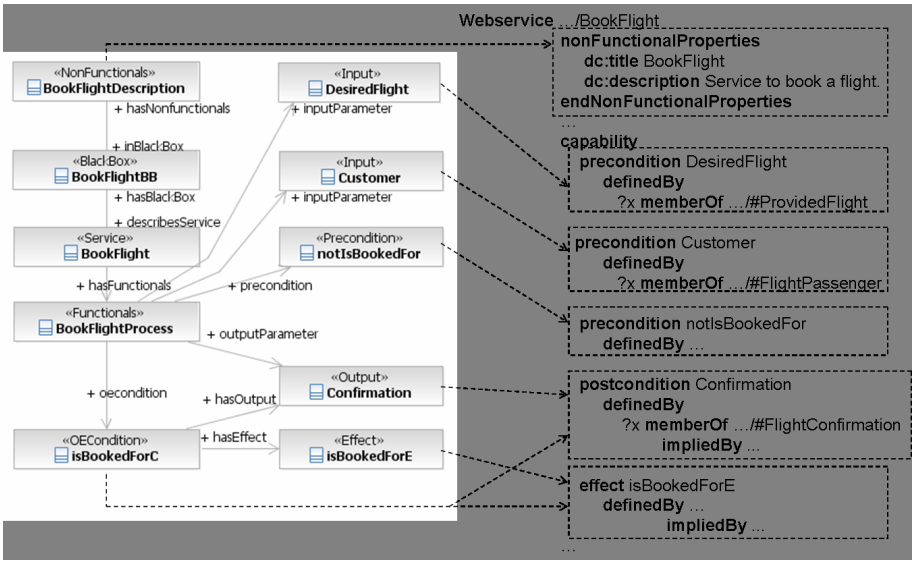

Fig. 7. Bookflight service request transformation from PIM4SWS to WSML.

The Z-specification of the concept FP' in the OWL-DL description D' produced by the MDSM is as follows [17]:

$$\frac{FP' : OWLClass}{subClassOf(allValuesFrom(AP', travelsBy')) = P'}$$

According to the Z-element equality definition above, and the semantic Z-equality relations (cf. table 2a-f) of the description logic operations above, the set of instances of FP (in Z) is equal to the OWLClass FP' (in Z):

$$instances(FP') = \{x' : OWLIndividual \mid \forall\, a_2' : OWLIndividual \bullet (x', a_2') \in subVal(travelsBy') \Rightarrow a_2' \in instances(AP')\} \subseteq instances(P')$$

Assuming that $P = P', AP = AP'$ and $travelsBy = travelsBy'$, the semantic equivalence of FP (in PIM4SWS-IM) and its transformation FP' (in OWL-DL) holds. The same is valid for FP' in WSML-DL. FP in *SHIQ(D)* can be specified in Z: `forall travelsBy'.AirPlane' ⊑ Passenger'` This expression can be equivalently written in FOL, F-Logic [6, 7] and WSML-DL style.
In FOL: $\forall\, x(\forall\, y(travelsBy'(x, y) \supset AirPlane'(y)) \supset Passenger'(x))$;
in F-Logic: $\forall\, x.(\forall\, y.x[travelsBy' \twoheadrightarrow y] \supset y : AirPlane') \supset x : Passenger')$;
in WSML-DL:

```
forall ?y( ?x [travelsBy' hasValue ?y] implies ?y memberOf AirPlane')
 implies ?x memberOf Passenger'
```

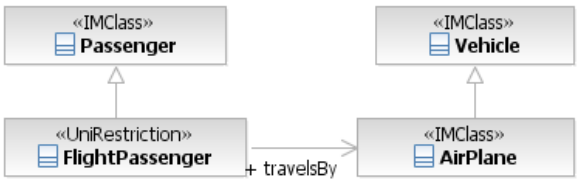

Fig. 8. Part of the PIM4SWS information model for service input concept FlightPassenger (FP).

The compiled service request in OWL-S, WSML and SAWSDL is passed by the MDSM to its integrated platform-specific matchmakers. Their ranked answer sets are aggregated and eventually presented by the MDSM to the orchestrator.

5 Conclusion

We provided a first approach to model-driven semantic Web service selection to support business process orchestrators at design time. In its inital version, our model-driven service matchmaker MDSM 1.0 is restricted to (a) specific matchmakers for OWL-S, WSML and SAWSDL, and (b) a platform-independent information model defined as intersection of OWL-DL and WSML-DL. However, the principle of model-driven semantic selection applies to other ontology languages and specific matchmakers to be plugged into the MDSM as well. Future work covers the extension of the PIM4SWS information model with OCL constraints, and transformations to SWRL and WSML-Rule [25, 24].

References

1. Acuna, C., Marcos, E.: Modeling Semantic Web Services: A Case Study. *Proc. of the 6th Intl. Conf. on Web Engineering*, ACM Intl. Conf. Proceeding Series; Vol. 263, Palo Alto, USA, 2006.
2. Borgida, A.: On the relative expressiveness of description logics and predicate logics. *Artificial Intelligence*, Vol.: 82, No.: 1-2, p.: 353-367, 1996.
3. Botelho, L., et al.: Service Discovery. *M. Schumacher, H. Helin, H. Schuldt (Eds.) CASCOM - Intelligent Service Coordination in the Semantic Web. Chapter 4.* Birkhuser Verlag, Springer, 2008.
4. Brockmans, S., et al.: Visual Modeling of OWL DL Ontologies Using UML. *In S.A. McIlraith et al., Proc. of the 3rd Intl. Semantic Web Conference*, Springer, Japan, 2004.
5. Chinnici, R., et al.: Web Services Description Language (WSDL) Version 2.0. [Online] Available: http://www.w3c.org/TR/wsdl20/, 2007.
6. de Bruijn, J., Heymans, S.: Translating Ontologies from Predicate-based to Frame-based Languages. *Proc. of the 2nd Intl. Conf. on Rules and Rule Markup Languages for the Semantic Web*, IEEE Computer Society, Washington, DC, USA, 2006.

7. de Bruijn, J., Heymans, S.: WSML Ontology Semantics. *WSML Final Draft*, Available: http://www.wsmo.org/TR/d28/d28.3/v0.1/20061218/, 2006.
8. Duke, R., Rose, G.: Formal Object Oriented Specification Using Object-Z. *Cornerstones of Computing*, Macmillan, 2000.
9. Duric, D.: MDA-based Ontology Infrastructure. *Intl. Journal on Computer Science and Information Systems*, Vol. 1, No. 1, 2004.
10. Gannod, G. C., et al.: Faciliating the Specification of Semantic Web Services Using Model-Driven Development. *Journal of Web Services Research*, 2006.
11. Grønmo, et al.: Transformations between UML and OWL-S. *European Conf. on Model Driven Architecture - Foundations and Applications (ECMDA-FA)*, Nürnberg, Germany, 2005.
12. Kaufer, F., Klusch, M.: WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. *Proc. of the IEEE Intl. Conf. on Systems, Man and Cybernetics*, Montreal, Canada, 2007.
13. Klusch, M., et al.: OWLS-MX: Hybrid Semantic Web Service Retrieval. *Proc. of the 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web*, Arlington VA, USA, 2005.
14. Koehler, J., et al.: The role of visual modeling and model transformations in business-driven development. *In 5th Intl. Workshop on Graph Transformation and Visual Modeling Techniques*, 2006.
15. Lara, R., Polleres, A.: Formal Mapping and Tool to OWL-S. Available: *http://www.wsmo.org/2004/d4/d4.2/v0.1/20041217/*
16. Lautenbacher, F., Bauer, B.: Creating a Meta-Model for Semantic Web Service Standards. *Proc. of the 3rd Intl. Conf. on Web Information System and Technologies (WEBIST) - Web Interfaces and Applications*, Barcelona, Spain, 2007.
17. Lucanu, D., et al.: Web Ontology Verification and Analysis in the Z Framework. TR 05-01, Faculty of Computer Science, University Alexandru Ioan Cuza, 2005.
18. OWL-S: Semantic Markup for Web Services, *W3C Member Submission*, Available: http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/, 2004.
19. OWL Web Ontology Language Reference, *W3C Recommendation*, Available: http://www.w3.org/TR/2004/REC-owl-ref-20040210/, 2004.
20. Rector, A., Schreiber, G.: Qualified cardinality restrictions (QCRs): Constraining the number of values of a particular type for a property. *W3C Editor's Draft*, Available: http://www.cs.vu.nl/ guus/public/qcr-20060508.html, 2006.
21. Scicluna, J.; et al. (2004): Formal Mapping and Tool to OWL-S. WSMO Working Draft 17 December 2004, Available: http://www.wsmo.org/TR/d4/d4.2/v0.1/.
22. Semantic Annotations for WSDL and XML Schema, *W3C Recommendation*, Available: http://www.w3.org/TR/2007/REC-sawsdl-20070828/, 2007.
23. Spivey, M.: The Z Notation: A Reference Manual, 2nd edition. *Prentice Hall International Series in Computer Science*, 1992.
24. Wang, H. H., et al.: A Formal Semantic Model of the Semantic Web Service Ontology (WSMO). *12th IEEE Intl. Conf. on Engineering of Complex Computer Systems*, Auckland, New Zealand, 2007.
25. Wang, H. H., et al.: Formal Specification of OWL-S with Object-Z. *The First ESWC Workshop on OWL-S: Experiences and Future Directions*, Austria, 2007.
26. Web Service Modeling Ontology (WSMO), *W3C Member Submission*, Available: http://www.w3.org/Submission/2005/SUBM-WSMO-20050603/, 2005.
27. Web Service Modeling Language (WSML), *W3C Member Submission*, Available: http://www.w3.org/Submission/2005/SUBM-WSML-20050603/, 2005.