Measuring Similarity between Semantic Business Process Models

Marc Ehrig **Agnes Koschmider** **Andreas Oberweis**

Institute of Applied Informatics and Formal Description Methods
Universität Karlsruhe (TH),
76187 Karlsruhe, Germany
Email: ehrig, koschmider, oberweis@aifb.uni-karlsruhe.de

Abstract

A business process may be modeled in different ways by different modelers even when utilizing the same modeling language. An appropriate method for solving ambiguity issues in process models caused by the use of synonyms, homonyms or different abstraction levels for process element names is the use of ontology-based descriptions of process models. So-called semantic business process models promise to support business process interoperability and interconnectivity. But, for (semi-) automatic process interoperability and interconnectivity two problems need to be solved. How can similar terms for process element names be automatically discovered and how can semantic business process composition be facilitated. In this paper we will present solutions for these problems based upon an OWL DL-based description of Petri nets.

Keywords: Petri nets, Ontologies, Semantic Business Process Models, Similarity Measures

1 Introduction

In E-Business, enterprises collaborate across organizational boundaries to perform common tasks. But, even when sharing similar demands, enterprises are using their specific vocabulary and structural representations for modeling business processes. By using formal languages such as Petri nets (Reisig & Rozenberg 1998) for modeling business processes, purely syntactic composition problems of interorganizational business environments may be solved (van der Aalst & Weske 2001), (Lenz & Oberweis 2003). However, a missing semantic representation of Petri net elements can hamper further interconnectivity of business processes. When enterprises decide to interconnect business processes, synonyms, homonyms or similar labeled process elements have to be identified to avoid misunderstandings. Furthermore, in order to understand business processes, significant experience in the field of business process engineering and effort to check differences between the respective business processes are required. By describing business process models in an unambiguous format which enables computer reasoning, the automation of process composition can be facilitated. These so-called semantic business process models promise appropriate business process discovery, interoperability and interconnectivity. The automation of process discovery can help to accelerate finding appropriate composable business process models faster than manually discovering business process models. Providing a (semi-)automatic approach for process interoperability and interconnectivity helps to save costs and time when establishing interorganizational business collaborations.

In this paper we will propose an approach for (semi-)automatic detection of synoyms and homonyms of process element names in order to support semantic process model interconnectivity and interoperability by measuring the similarity between business process models semantically modeled with the Web Ontology Language (OWL)(McGuinness & van Harmelen 2003). We describe traditional Petri nets with an Ontology Language-based format OWL (McGuinness & van Harmelen 2003). The translation of Petri nets into OWL makes it possible to automatically compute similarities between business process models. Figure 1 depicts two (simplified) business processes. Both processes perform *requests* and same terms are utilized (e.g. in both a *request* for something is issued). But the request is treated in different ways. In **Business Process I** the *request* is checked for completeness and will be accepted or rejected afterwards and in **Business Process II** the *request* is to be rejected or accepted without checking a selection criteria. Comparing these process models manually results in a high similarity degree (several element names occur in both processes). But, to determine differences between larger process models (>100 elements) requires time, effort and a couple of various modeling experiences.

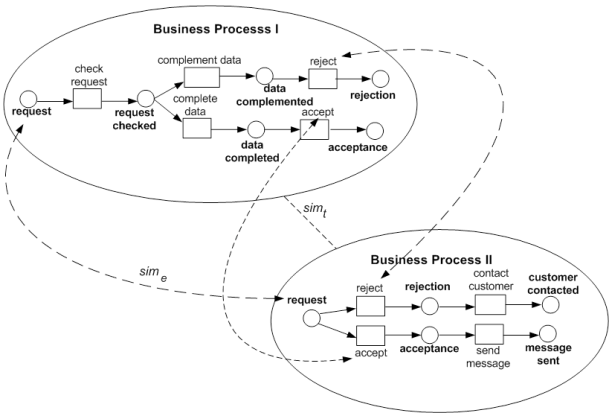

Figure 1: Example business processes with same terms for process element names

The degree of similarity between business process models correlates positively with the number of used synonyms and negatively with the number of used homonyms. In this paper we will demonstrate that by using the three similarity measures, *syntactic-*,

linguistic- and *structural* measure, we can compute similarity degrees between a pair of process element names (sim_e) and between a pair of business process models (sim_t). In order to compute the syntactic similarity degree we compare the number of common characters in the element names (e.g. *confirmation* vs. *verification*). The linguistic similarity degree relies on a dictionary to determine synonyms. However, syntactic- and linguistic similarity measures by themselves do not exploit the context of names. We do so with structural similarity measures, which helps to detect primarily homonyms. By extending existing ontology similarity approaches ((Ehrig, Haase, Stojanovic & Hefke 2005) and (Maedche & Staab 2002)) for semantic business process models we show possibilities to achieve improvements of business process interoperability and interconnectivity and ease of process composition when faciliating process similarity computation. Implementation and evaluation experiences with our approach are presented as well.

The structure of this paper is as follows. Firstly, we will recall the main notions of Petri nets, ontologies and a semantic description of Petri nets with OWL DL. In Section 3 we will describe an approach for measuring *syntactic-*, *linguistic-* and *structural* similarity between process element names. We aggregate these similarity measures to a combined similarity measure to determine similarities between two business process models. Application areas of our approach will be illustrated in Section 4. Practical experiences will be presented in Section 5. Section 6 surveys related work and Section 7 concludes the paper with an outlook on future research.

2 Foundations

Petri nets, ontologies and semantic business process models are introduced in the following subsections.

2.1 Petri nets

Petri nets (Reisig & Rozenberg 1998) are a widely accepted graphical language for the specification, simulation and verification of behaviour of information systems. Formally, a Petri net is a directed bipartite graph with two sets of nodes (places and transitions) and a set of arcs (flow relation). Numerous Petri net variants have been proposed, which can be subsumed in elementary or high-level Petri nets. In elementary Petri nets (place/transition nets), the flow of tokens representing anonymous objects defines the process flow. To describe objects with individual behavior, several variants of high-level Petri nets have been proposed ((Jensen 1994), (Genrich & Lautenbach 1981)). In (Genrich & Lautenbach 1981) predicate/transition nets (Pr/T nets) are introduced where places represent relation schemes (predicates). In Pr/T nets a function assigns to each place a marking, which is a relation of the respective type. The set of all place markings at a given time describes a certain global system state. A transition represents an operation on the relations in its input/output places. If a transition occurs, tuples are removed from the relations in its input places and are inserted into the relations of its output places. A logical expression, which may be assigned to a transition, makes it possible to specify certain conditions for the selection of tuples to be inserted or removed. Figure 2 shows two (simplified) Pr/T net descriptions of business processes.

Business Process I performs travel booking (where the booking *request* is described by its attributes *Name*, *Destination*, *Date*, and *Quantity*). After receiving a *request* this *request* will be checked for consistency (transition *check request*). In case of inconsistent data it needs to be corrected. If the request data is complete then travel availability will be checked. The travel booking is accepted (Condition $AQ - Q > 0$ of transition *book travel*) or rejected otherwise. Upon completion of the booking process a confirmation is sent to the customer (transition *send confirmation*). In **Business Process II** flight requests are processed where *flight* is defined by its attributes *Date*, *City*, *AvailableSeats* and *request* by *Name*, *City*, *Date* and *Amount*. With these data at hand the flight request will be rejected or accepted. In both cases the customer is informed by a respective message (in **Business Process I** by a confirmation and in **Business Process II** by a verification).

2.2 Ontologies

An ontology (Staab & Studer 2004) defines the relevant concepts of its domain (its terminology), its properties and instances (the world description). The following brief definition describes the term ontology as being used in our scenario. An ontology O is defined by a tuple as follows.

$$O := (C, H_C, P_C, I, A)$$

Concepts C of the schema are arranged in a subsumption hierarchy H_C. Concepts are defined by properties P_C (where properties can also be arranged in a hierarchy). By instatiating concepts each concept has a set of instances I. Additionally, an ontology might contain a set of axioms (denoted as A), which can be used to infer implicit knowledge from explicit one.

By standardizing the Web Ontology Language (OWL), the World Wide Web Consortium (W3C) laid the foundation for a wide-spread use of ontologies in business. OWL is syntactically layered on RDF. Therefore, the syntax of OWL is the syntax of RDF/XML (RDF/XML Syntax Specification). Figure 3 shows a simple example for OWL syntax. The concept *Place* is a subconcept of the concept *PetriNet* and where *Place* has a objectproperty *transRef* with the domain *Place* and the range *Transition*.

```
<owl:Class rdf:ID="Place">
 <rdfs:subClassOf rdf:resource="#PetriNet"/>
  <owl:ObjectProperty rdf:ID="transRef">
   <rdfs:domain rdf:resource="#Place"/>
   <rdfs:range  rdf:resource="#Transition"/>
  </owl:ObjectProperty>
  ...
</owl:Class>
```

Figure 3: Example for OWL syntax

OWL is given by three variants with an increasing degree of expressiveness: OWL Lite, OWL DL and OWL Full. For our work we will refer to OWL DL (Description Logic) in order to be able to use available off-the-shelf reasoning technologies.

2.3 Semantic Business Process Models

To solve ambiguity issues caused by the use of different terms for describing the same things and in order to support (semi-)automatic system collaboration, a machine readable and interpretable format, which might be used for ontological reasoning is required for Petri nets. Therefore, we have translated Petri net elements (i.e. Pr/T net elements) into OWL DL (Koschmider & Oberweis 2005). So-called semantic business process models (SBPM) consist of the

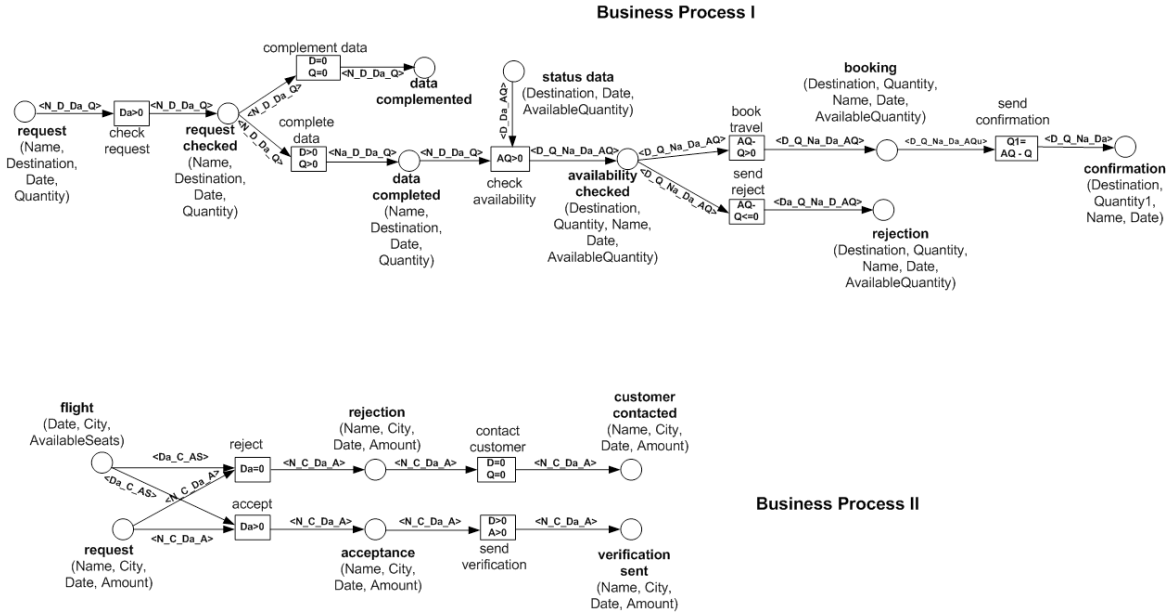

Figure 2: Business Processes modeled with Pr/T nets

concepts **Transition** (T), **Place** (P), **FromPlace** (F_r) (arcs running from a place to a transition) and **ToPlace** (F_w) (arcs running from a transition to a place). Besides, these concepts have specific attributes with ranges pointing to further concepts of the ontology. For example, the concept **Place** has the attribute **transRef** with the range **Transition**[1].

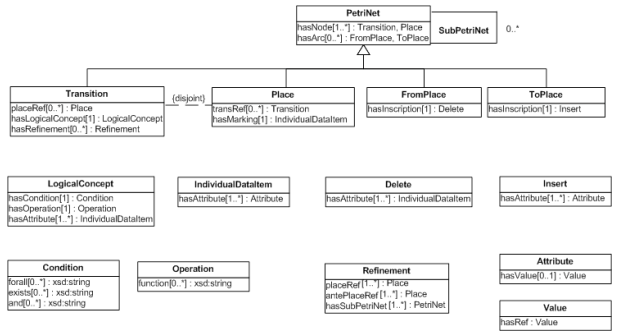

Figure 4: Pr/T net Ontology

OWL DL can be considered as a syntactic variant of the $\mathcal{SHOIN}(D)$ Description Logic which is known to be decidable (Horrocks 2005). The representation of Petri nets in Description Logic (Brockmans, Ehrig, Koschmider, Oberweis & Studer 2006) allows to reason about Petri net elements. A semantic business process model corresponds to the instantiation of the Pr/T net ontology, which can be represented in $\mathcal{SHOIN}(D)$ Description Logic or in OWL syntax. By representing semantic business process models in OWL syntax the corresponding files can be manipulated and enable to deduce new data being not directly modeled in business processes. An example is given in Figure 5.

For a detailed description of an OWL-based description of Petri nets we refer to (Koschmider & Oberweis 2005).

[1] Concepts such as LogicalConcept, IndividualDataItem, Delete, Insert, Condition, Operation, Attribute and Value are specific elements of Pr/T nets.

```
<petri:PetriNet rdf:ID="Perform-requests">
   <petri:hasNode rdf:resource="#request"/>
 <petri:Place rdf:ID="request">
  <petri:hasMarking>
   <petri:IndividualDataItem rdf:ID="R_request">
    <petri:hasAttribute rdf:resource="#Name"/>
    <petri:hasAttribute rdf:resource="#Destination"/>
    <petri:hasAttribute rdf:resource="#Date"/>
    <petri:hasAttribute rdf:resource="#Quantity"/>
   </petri:IndividualDataItem>
  </petri:hasMarking>
 </petri:Place>
  ....
</petri:PetriNet>
```

Figure 5: Representing Semantic Business Process Models in OWL syntax (excerpt)

3 Measuring Similarity between Semantic Business Process Models

In the following we consider two different application scenarios for measuring similarity between semantic business process models (SBPMs). One important issue is the semantic interconnectivity of business processes, which ensures flexible process interface composing. Supporting reusability of business process models is the other important issue. Given process templates (reference business process models) the system examines the processes and tries to find similarities between a modeling process and the process templates.

For these two scenarios we provide a solution based on measuring *syntactic-*, *linguistic-* and *structural* similarity as a way to evaluate the similiarity degree between two semantic process element names. By aggregating these similarity measures to a combined similarity measure we can compute the similarity between two SBPMs. In Section 5 we will explain our implemented system for similiarity calculation. To compute the syntactic similarity degree we compare the number of characters of concept instances' names (e.g. *confirmation* with *verification*). The linguistic similarity measure relies on a dictionary to determine synonyms. However, syntactic- and linguistic similarity measures alone do not make use of the context of concept instances. With structural similarity measures we consider the context of concept instances and makes it possible to detect homonyms.

In the following, we assume that process element names to be compared have the same level of detail. It is not reasonable to compute similarity between for instance *contract application* and *contract* where *contract application* can be regarded as a specialization of *contract*. Abstraction levels of element names can be calculated with the similarity metrics proposed by (Wu & Palmer 1994). These metrics take into account the depth of terms in lexical reference systems such as WordNet (Fellbaum 1998). The abstraction of terms correlates negatively with the depth of terms.

3.1 Properties of Similarity Measures

Similarity measures as explained in this paper must fulfill the properties *symmetry* and *reflexivity* as introducted in (Richter 1992). A real-valued function sim: S×S → [0,1] on a set S measuring the degree of similarity between two elements is called similarity measure if, ∀ x, y, ∈ S:

1. sim (x, y) = sim (y, x) (Symmetry)
2. sim (x, x) = 1 (Reflexivity)

In the literature a long debate on questions about whether similarity measures are symmetric or not can be found (Tversky 1977). Since in the context of business process models we use similarity measures to support (semi-)automatic comparing of business process models it is obvious that the returned similarity degrees must fulfil the symmetrical property.

3.2 Syntactic Similarity Measure

In order to measure similarity between two character strings, (Levenshtein 1966) proposed the edit distance method. The edit distance between two strings is defined by the number of changes (addition, deletion and replacement of characters) necessary to turn one string into another. For example, the edit distance between the strings "confirmation" and "verification" equals 6, because six substitutions are sufficient to transform "confirmation" to "verification". The greater the edit distance, the more different the strings are. Based on the edit distance method *(ed)* (Maedche & Staab 2002) have proposed a syntactic similarity measure sim_{syn} as shown below, which returns similarity degrees between 0 and 1, where 1 stands for perfect match and zero for bad match.

$$sim_{syn}(c_1, c_2) := max(0, \frac{min(|c_1|,|c_2|) - ed(c_1,c_2)}{min(|c_1|,|c_2|)})^2$$

In contrast to the edit distance method the syntactic similarity measure of (Maedche & Staab 2002) takes into account the number of characters and is inverse to the edit distance. This measure considers the number of changes being made to change one string into the other and weights the number of these changes against the length of the shortest string of these two strings ($min(|c_1|, |c_2|)$). Table 1 shows syntactic similarity degrees (sim_{syn}) for several concept instance names of Figure 2; e.g. sim_{syn} for *send confirmation* and *send verification* is 0.64.

Table 1: Results of syntactic similarity measure

c_i	c_j	sim_{syn}
request	request	1.0
send reject	reject	0.1667
booking	accept	0.0
...	...	...
send confirmation	send verification	0.64

[2] $|string|$ denotes for a given string the number of characters.

The computation of sim_{syn} for the pair *send confirmation* and *send notification* (not labeled in our process scenario in Figure 2) returns a syntactic similarity degree of 0.70. Considering only syntactic similarity measures would imply that the last pair would be added to the result list due to its higher similarity degree (0.70 > 0.64). Purely syntactical methods that treat elements isolated from their semantics are insufficient since misunderstandings may happen by expressing the same meaning in different terms. By considering the semantics of instance names with linguistic measures we will explain in the next subsection that the pair (*send confirmation* vs. *send verification*) is to a greater degree similar than (*send confirmation* vs. *send notification*).

3.3 Linguistic Similarity Measure

To exploit linguistic features we have utilized WordNet (Fellbaum 1998) (by using the JWNL API (Didion 2003)) and a specific UML Profile (Brockmans et al. 2006) as so-called background ontologies. JWNL is an API implemented in Java for accessing the WordNet dictionary. WordNet is an English online lexical reference system, which provides synonym, hyperonyms (generalization) and hyponyms (specialization) sets consisting of nouns, verbs, adjectives, and adverbs. To access the UML Profile vocabulary by using a specific API[3], a converting tool provides an automatic translation from the visual UML modeling to OWL DL syntax. WordNet is in contrast to the UML Profile fix and predefined.

To compute linguistic similarity degrees between process element names we are considering synonym sets proposed by WordNet. For instance, WordNet submits for the term *verification* two senses in a synonym relationship ordered by estimated frequency:

1. *confirmation*, *verification*, *check*, *substantiation* – (additional proof that something that was believed (some fact or hypothesis or theory) is correct; "fossils provided further confirmation of the evolutionary theory")
   → proof, cogent evidence – (any factual evidence that helps to establish the truth of something
2. *verification* – ((law) an affidavit attached to a statement confirming the truth of that statement)
   → affidavit – (written declaration made under oath; a written statement sworn to be true before someone legally authorized to administer an oath)

One synonym for *verification* is *confirmation* (sense 1). Thus, our implemented system indicates some linguistic relationships between these two instance names. Modeling business processes depends on the modeler and varies from one to another. One modeler might denote elements only with verbs while someone else might denominate elements with labels composed of nouns, verbs and adverbs (e.g. *send confirmation* or *print final deadline*). In case of composed process element names we calculate only the linguistic similarity for names that do not satisfy a sim_{syn} of 1.0. For instance, computing linguistic similarity for the pair (*send confirmation* vs. *send verification*) is restricted to the pair *confirmation* vs. *verification* due to identity of *send*. Otherwise, the linguistic similarity for *send employee* and *send software* would be > 0 due to the term send. This makes no sense from the linguistic point of view.

[3] this API is currently not available, but will be published in near future.

In order to compute linguistic similarity the function $\eta(c)$ retrieves all WordNet senses (all terms in a synonym relationship) for the phrasing of the given ontological concept instance c. Let $S = \eta(c_1) \cap \eta(c_2)$ be the set of common senses of the phrasings of two concept instances c_1 and c_2 to be compared. Then the cardinality $f(S)$ is defined as:

$$f(S) = \begin{cases} 1 & \text{iff} \quad \eta(c_1) \cap \eta(c_2) \neq \emptyset \\ 0 & \text{iff} \quad \eta(c_1) \cap \eta(c_2) = \emptyset \end{cases}$$

Let $max(|\eta(c_1)|, |\eta(c_2)|)$ be the maximum of the cardinalities of the two sets $\eta(c_1)$ and $\eta(c_2)$, then the linguistic similarity sim_{ling} between two concept instance names is defined as:

$$sim_{ling}(c_1, c_2) = \frac{f(S)}{max(|\eta(c_1)|, |\eta(c_2)|)}$$

This measure considers a synonym relationship of two instances, the number of synonyms that are proposed for one term by WordNet and weights the number of synonyms against the maximum sense cardinality of these two terms. This measure returns a similarity degree of 1.0 for c_1 vs. c_2 if c_1 is the only synonym for c_2 and vice versa.

A higher number of senses returns a lower linguistic similarity degree. But, as we will explain in the next section users can assign individual weights, which return higher linguistic similarity degrees. Table 2 shows several linguistic similarity degrees.

Table 2: Results of linguistic similarity degrees

c_i	c_j	sim_{ling}
request	request	1.0
confirmation	verification	0.5
...	...	...
send rejection	send acceptance	0.0

This linguistic measure returns another similarity degree for the same pair of process elements as the syntactical measure described above. Considering only linguistic features would return for the pair (*allocate three weights*, vs. *alocate tree wights*) a sim_{ling} of 0.0 due to spelling mistakes (allocate only with one "l", three without "h" and weight without "e"). The computation of syntactic similarity for this pair returns a degree of 0.842. Thus, the computation of both measures is necessary in our application scenarios.

In the following we will show that the computation of a so-called context of concept instance names uncovers homonyms, e.g. the word *organization* means once a social organization and once a governance. The linguistic and syntactic similarity would return a degree of 1.0 for (*organization* vs. *organization*).

3.4 Structural Similarity Measure

To make use of the hierarchical ontology structure for determining semantic similarities the context of concept instances should be considered. The structural similarity measure has as input two concept instances (c_1 and c_2) and their context. The context of a term is defined as the set of all elements which influence the similarity of the term. To elucidate the context of concept instances we have extracted concept instance names from $SBPM_1$ and $SBPM_2$ (SBPMs transformed of business processes in Figure 2) and have represented the instances as nodes and the properties as undirected arcs in a tree as shown in Figure 6.

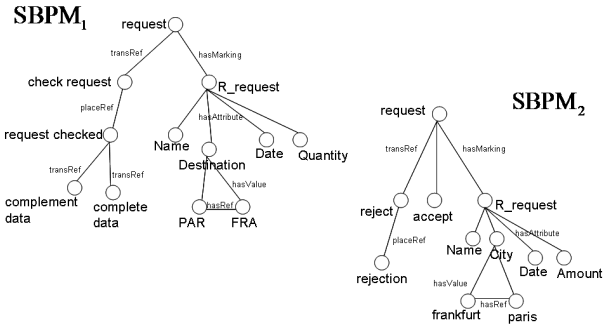

Figure 6: Tree representations of two Semantic Business Process Models

The upper level of the tree $SBPM_1$ represents the first element name of $SBPM_1$ (*request*) and the bottom of the tree (*complement data* and *complete data*) are the third and fourth transitions. The subsequent element of *request* is *check request* (second node on the left hand side) and the marking name is *R_request* which is defined by the attributes *Name*, *Destination*, *Date*, and *Quantity*. *Destination* has the values *PAR* and *FRA*. The structure of tree $SBPM_2$ is given analogously. In the tree representation the similarity degree of attribute names (i.e. Name, Destination) is influenced by the similarity of values. The similarity of place names is influenced by attributes and subsequent transition names and the similarity of subsequent transition names is influenced by subsequent place names.

Based upon this relationship we define the context for place names by a tuple $con_p =< a, v, sV, tR >$ with

- a = all attributes of a specific place,
- v = all values for each attribute,
- sV = sibling values,
- tR = all subsequent transition names linked via the transRef property.

The context of the concept instance name *request* in Figure 6 is defined by its attributes *Name*, *Destination*, *Date* and *Quantity*, the corresponding values (in Figure 6 we have only depicted *PAR* and *FRA*) and the subsequent instance name *check request*.

To define the context of attribute names we restrict con_a by a tuple $< sA, v, sV >$ with

- sA = sibling attributes,
- v = all values of the specific attribute,
- sV = sibling values.

The context for transition names is a tuple $con_t =< pR >$ with

- pR = all subsequent place names linked via the placeRef property.

The context of the concept instance name *check request* in Figure 6 is the subsequent instance name *request checked*.

But, each of the context elements influences the instance name with different weights. Attributes are significant and influence instance names more than values, which are more relevant to identify attributes. These different degrees of influence can be considered by instance weights which differ from process to process.

Table 3: Context and weights for concept instances

Comparing	Context	Measure	Weight
Places	Names	synt/ling sim.	0.2
	Attribute	synt/ling sim.	0.4
	Value	synt sim.	0.1
	successor (transRef)	synt/ling sim.	0.3
Attribute	Names	synt/ling sim.	0.2
	sibling Attribute	synt/ling sim.	0.4
	Values	synt sim.	0.4
Value	Names	synt/ling sim.	0.2
	Attribute	ling sim.	0.4
	Values Reference	ling sim.	0.4
Transition	Name	synt/ling sim.	0.2
	successor (placeRef)	synt sim.	0.8

In business process models with a lot of places, attributes and transitions weights play a less important role than in small process models with less elements. The more instances are modeled in a SBPM the more negligible the weights are. Our two processes modeled in Figure 2 have several places with their attributes and transitions. But, in order to make the business processes comprehensible we did not assign values for attributes or any refinements (subprocesses). For these two processes we have chosen the weights as depicted in Table 3[4]. With these weights the exact similarity degree has been computed as we will explain in Section 5.

To compute the structural similarity degree between two concept instances let c_1 be a particular concept instance of $SBPM_1$ and c_{2_j} be a set of concept instances of $SBPM_2$. Then $(sim_{k_i}(c_{1_i}, c_{2_j})))$ denotes the specific similarity measure used for the context elements of c_1 and c_2 which we multiply with individual weights as depicted in Table 3. In order to consider only reasonable pairs we filter only the maximum between c_1 and c_{2j} and c_{1_i} and c_2.

$$sim_{str}(c_1, c_2) := \frac{\sum_{i=1}^{n} \max_{j \in 1..m}(w_{k_i} * sim_{k_i}(c_{1_i}, c_{2_j}))}{\sum_{i=1}^{n} w_{k_i}}$$

This measure returns a similarity degree of 1.0 if the syntactical and/or linguistic similarity for the context elements equals 1.0. Some results of the structural calculation are shown in Table 4.

Table 4: Results of structural similarity degrees

c_i	c_j	sim_{str}
request	request	1.0
send reject	reject	0.9
request checked	acceptance	0.0
confirmation	customer contacted	0.0
...	...	...
send confirmation	send verification	1.0
confirmation	verification sent	0.8

The syntactic and linguistic similarity for the term organization is 1.0. Evaluating the context of the term *organization* would perhaps return a different similarity degree for sim_{str}. For instance, if the attributes of the concept instance name *organization* from $SBPM_1$ are *Room* and *Contact* and another instance name *organization* from $SBPM_2$ has *Name* and *City*, then a structural similarity of 0.0 is returned. The attributes (Room vs. Name) or (Room vs. City) have no synonym relationship and thus organization is unlike organization.

[4]for values we calculate only syntactical similarity as values are of datatype integer or string such as 11 or Smith

In the following section we will aggregate the three similarity measures syntactic, linguistic and structural similarity to a combined similarity measure.

3.5 Combined Similarity Measure

In order to compute the combined similarity sim_{com} between concept instance names c_1 and c_2 let c_1 be a particular concept instance name of $SBPM_1$ and c_2 be a concept instance name of $SBPM_2$. Then the combined similarity is an aggregation of the degrees returned from the syntactical, linguistic and structural similarity measures as explained above.

$$sim_{com}(c_1, c_2) :=$$

$$\frac{w_{c_{syn}} * sim_{syn}(c_1, c_2) + w_{c_{ling}} * sim_{ling}(c_1, c_2) + w_{c_{str}} * sim_{str}(c_1, c_2)}{w_{c_{syn}} + w_{c_{ling}} + w_{c_{str}}}$$

These similarity measures have particular weights w_{syn}, w_{ling} and w_{str} that can be individually assigned by users or learned e.g. using machine-learning-based approaches on a training set. The effects of different weights on the similarity results have been investigated by (Berkovsky, Eytani & Gal 2005). The combined similarity measure takes into account syntactical, linguistical and structural features of instance names.

The similarity between two semantic business process models $SBPM_1$ vs. $SBPM_2$ is defined by semantic relationships, which we consider by the two sets of concept instances C_1 and C_2 of $SBPM_1$ and $SBPM_2$.

- **equivalence**: sim $(SBPM_1, SBPM_2) = 1$ iff $C_1 = C_2$,
- **disjointness**: sim $(SBPM_1, SBPM_2) = 0$ iff $C_1 \cap C_2 = \emptyset$
- **intersection**: sim $(SBPM_1, SBPM_2) \in]0...1[$ iff $C_1 \cap C_2 = \{x | (x \in C_1) \wedge (x \in C_2)\} \wedge C_1 \neq C_2$.

Based on these semantic relationships we specify an overall similarity between two semantic business process models. Let c_1 be a particular concept instance of $SBPM_1$ and c_{2_j}, $j \in 1..m$, be a set of concept instances of $SBPM_2$. Then $\max_{j \in 1..m}(sim_{com}(c_1, c_{2_j}))$ denotes the maximum combined similarity between c_1 and concept instances c_{2_j} of $SBPM_2$.

$$sim_{SBPM}(C_1, C_2) := \frac{1}{n} \sum_{i=1}^{n} (\max_{j \in 1..m} (sim_{com}(c_{1_i}, c_{2_j})))$$

For the calculation of $SBPM_1$ and $SBPM_2$ the sim_{SBPM} equals 0.32. Experiments have shown that a value > 0.4 indicates a low business process similarity. A similarity degree ($SBPM_1$, $SBPM_2$) $= 0.4$ indicates an intersection relationship. This implies that only 40% of instances names (process element names) of $SBPM_2$ are similar to instance names in $SBPM_1$.

To make reasonable statements about the similarity degree returned for two SBPMs, neural nets can be used for example to learn a threshold θ for the combined similarity instead of using a fix threshold.

4 Application

Besides business process interconnectivity user support for business processes modeling is another promising application area for our approach (Betz, Klink, Koschmider & Oberweis 2006). Manual modeling of business processes is a time consuming task. Typos and structural modeling errors make it particularly error prone to model business processes manually. Users can be assisted in modeling business processes by providing an autocompletion mechanism during the modeling process. Before proposing appropriate process elements, a recommendation mechanism has to compare modeling elements with process templates and has to find similar elements, which will be proposed as fitting subsequent elements. Thus, the mechanism has to compare process templates with process elements, which are currently modeled, by computing their similarity degrees. Similarity computation of elements is being carried out during the modeling process and is not directly visible to the modeler. SBPMs as described in Section 2.3 promise to facilitate (semi-)automatic interconnectivity of business processes. To check if a process model meets certain properties we utilize in our recommendation system formal methods to validate that the insertion of the proposed process fragments does not cause deadlocks and synchronization errors.

Additionally, by utilizing our system for business process composition is helpful for process modelers if the system could indicate the position (at the beginning, in the middle, at the end of the process) of appropriate sequences of activities overlaping each other as shown in Table 5. In addition to the combined similarity degrees we specify the relative position number of the node and the element name (=Position) where P_1 is the first place in the business process, T_1 the first transition and so on. **Request**$_{SBPM_1}$ is the first place in **Business Process I** and the first place in $SBPM_2$. Both element names have a combined similarity of 1.0.

Table 5: Results of similarity measurement with position of concept instances

C_1	Position	C_2	Position	sim_{com}
request	P_1	request	P_1	1.0
send reject	T_6	reject	T_1	0.9
confirmation	T_7	verification	T_4	0.8
...	...	...	...	...

5 Implementation and Evaluation

We have implemented a Petri net editor called SemPeT[5] that offers semantic business process models export, which employs the Jena Semantic Web Framework API (HP 2003).

Furthermore, in order to support similarity measurement between semantic business process models we have integrated FOAM[6], an alignment and mapping framework for ontologies, into SemPeT. On top of FOAM we have implemented the features of measuring combined similarity between process element names and between two business process models. By choosing two semantic business process models (see Figure 7) to be compared our prototype computes sim_{com} and sim_{SBPM}. In the first iteration of similarity measurement instance names of the same concept are compared (e.g. the name *request* of the concept *place* from the left SBPM is compared with all instances of the concept *place* from the right SBPM). In this iteration *request* is compared with *request*, *rejection* and *verification*. After several iterations (by default the system iterates three times) the system displays only the instances with the highest similarity degree. In this example the pair (request, request) has the highest combined similarity. Transitions are handled analogously.

An excerpt of results from SemPeT is shown in Table 6. The second value is the result from the syntactic calculation, the third one from the lingustic measure and the fourth result returns from the structural measurement (sim_{str_P} corresponds to the structural similarity of places, sim_{str_A} to the structural similarity of attributes, and so on). The aggregation to a combined similarity is the first value.

In case of different grammatical usages of words we are restricting the suffixes for nouns, verbs and adjectives as shown below.

- name="noun" value="s=, ses=s, xes=x, zes=z, ches=ch, shes=sh, men=man, ies=y"
- name="verb" value="s=, ies=y, es=e, es=, ed=e, ed=, ing=e, ing="
- name="adjective" value="er=, est=, er=e, est=e"

Our implemented prototyp computes only the similarity between concept instance names (sim_{com}) and between semantic business process models (sim_{SBPM}) and does not consider the system behavior such as the control flow semantics (Hidders, Dumas, van der Aalst, ter Hofstede & Verelst 2005). By using our approach for modeling support (as explained in Section 4) our recommendation system validates only behavioral properties such as deadlocks or lacks of synchronisation.

We found out, that our results are downgraded if subsequent elements (for places transitions and for transitions places) are labeled completely different and have no relationship to its predecessor (e.g. a transition is denominated by *send confirmation* and a subsequent place by *printed* instead *confirmation sent* or *confirmation received*).

Our approach has been validated on small ($<$ 100 transitions) business process models. Our implemented system found nearly always correspondencies of instance names. This has been verified manually.

[5] http://aifbserver.aifb.uni-karlsruhe.de/sempet/index.htm
[6] http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/

Figure 7: Two Semantic Business Process Models

Table 6: Results of similarity measurement

instance name	sim_{com}	sim_{syn}	sim_{ling}	sim_{str_P}	sim_{str_I}	sim_{str_A}	sim_{str_V}	sim_{str_T}
(*confirmation* vs. *verification*)	0.8	0.64	0.5	0.95	0.0	0.0	0.0	0.0
(*verification* vs. *confirmation*)	0.8	0.64	0.5	0.95	0.0	0.0	0.0	0.0
(*Quantity* vs. *Amount*)	0.55	0.0	0.35	0.0	0.0	0.90	0.0	0.0
(*send request* vs. *send accept*)	0.2	0.55	0.0	0.0	0.0	0.0	0.0	0.0

6 Related Work

To the best of our knowledge, there is no other approach that describes high-level Petri nets in OWL. (Gasevic & Devedzic 2004) propose a Petri net ontology (for elementary Petri nets) that should enable sharing Petri nets on the Semantic Web and should make it possible to transform a specific XML-based Petri net format into OWL. The aim of our work is to reuse and analyze business process models not primary in the Semantic Web but in business environments. A domain ontology represents only a static structure of a domain; therefore our Pr/T net Ontology contains (in contrast to (Gasevic & Devedzic 2004)) only static elements.

In order to support process interconnectivity, several approches such as (Greco, Guzzo, Pontieri & Saccà 2004) have been proposed. In (Greco et al. 2004) a framework is proposed to enable ontology-driven process modeling. By utilizing the framework users can specify their business process models with semantically enriched processes. For process schema definition the authors propose a formal process model of their own and do not refer to existing business process modeling languages. Furthermore, the benefit of the framework is to provide manual support for business process designers, which means that the approach is not focussing on (semi-)automated interconnectivity of business process models. The benefit of our approach is that the user does not need to parameterize the similarity computation each time when computing similaritites.

A lot of research in service discovery is done in the field of Web services. The description and discovery of Web services can be seen as a prerequisite for Web service composition. (Cardoso & Sheth 2003) propose an algorithm to discover Web services and resolve heterogeneity among their interfaces in a workflow environment with the use of ontologies. Web services in contrast to business process models have only to interoperate at the input and output interface level without considering the context of process elements. To describe, match and compose Web services, (Pahl & Casey 2003) propose logical reasoning techniques based on a formal ontology framework. The prerequisite of this approach is the discovery of appropriate Web services which is a time-consuming task. The approach of (Chun, Atluri & Adam 2002) present a way to automatically compose workflows. Based upon an ontology of services an algorithm is provided that facilitates the workflow design and requires fewer evaluations at run time. The prerequisite for the use of this approach is large domain knowledge for modeling the ontologies. In contrast to these approaches we compute (semi-)automatically the similarity between process elements and process models and a so-called recommendation system proposes fitting subsequent elements.

As mentioned in Section 5 we have not yet considered the control flow semantics. The computation of equivalence of control flows is described in (Kiepuszewski, ter Hofstede, & van der Aalst 2003). In (Hidders et al. 2005) an approach is proposed to compute the relative expressiveness and variability of workflows. The similarity computation for elements meaning have not been regarded by these approaches.

A lot of research work has been done in the field of similarity measurement. In (Resnik 1999) an approach is presented for measuring syntactic- and linguistic similarity in a taxonomy. In a taxonomy relationships between concepts and properties are not considered. (Lin 1998) presents an information-theoretic definition of similarity that is applicable as long as there is a probabilistic model for terms.

Altogether, it does not exist an approach that measures syntactic and semantic similarity between business process models, particularly Petri net-based business process models.

7 Conclusion

In this paper we have presented an approach for measuring similarity between process element names and between semantic business process models. Consequently, semantic business process models are an instantiation of the Pr/T net ontology; thus our similiarity approach is feasible for measuring similarity between business process models. We have shown that calculating only syntactic- and linguistic similarity is insufficient since the instance context is not considered and homonyms can not be discovered. By taking into account structural aspects of instances, sufficient similarity degrees between element names and processes can be computed. Our similarity measurement approach provides a basis for future reseach.

Furthermore, we use the feasibility of semantic business process models for ontological reasoning to consider business rules in our similarity calculation. Usually, business process models are modeled according to specific business rules such as *all high loss estimations must include an expert's inspection* or *if more than 2 persons travel together, the third pays only half price*. Business rules represent enterprise policies, knowledge and expertise. The realization of an autocompletion mechanism requires to consider rules stated for business process models. The impact of such a mechanism is to decrease the amount of modeling time and to improve process model quality.

The simplicity of our approach makes it possible to apply our similarity calculation for other process modeling languages such as for the Business Process Execution Language for Web Services (Andrews, Curbera, Dholakia, Goland, Klein, Leymann, Liu, Roller, Smith, Thatte, Trickovic & Weerawarana 2003) or Event-driven Process Chains (Scheer 1998).

Besides semantic similarity computations as presented in this paper we are planning to integrate approaches for comparing processes based on their control flow semantics.

References

Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I. & Weerawarana, S. (2003), Business Process Execution Language for Web Services, Version 1.1, Specification, BEA Systems, IBM Corp., Microsoft Corp., SAP AG, Siebel Systems.

Berkovsky, S., Eytani, Y. & Gal, A. (2005), Measuring the relative performance of schema matchers., *in* 'Web Intelligence', IEEE Computer Society, pp. 366–371.

Betz, S., Klink, S., Koschmider, A. & Oberweis, A. (2006), Automatic user support for business process modeling, *in* K. Hinkelmann, D. Karagiannis, N. Stojanovic & G. Wagner, eds, 'Proceeding of the Workshop on Semantics for Business Process Management at the 3rd European Semantic Web Conference 2006', Budva, Montenegro, pp. 1–12.

Brockmans, S., Ehrig, M., Koschmider, A., Oberweis, A. & Studer, R. (2006), Semantic alignment of business processes, *in* Y. M. J. F. P. C. J. Cordeiro, ed., 'Proceedings of the Eighth International Conference on Enterprise Information Systems (ICEIS 2006)', INSTICC Press, Paphos, Cyprus, pp. 191–196.

Cardoso, J. & Sheth, A. (2003), 'Semantic e-workflow composition', *Journal of Intelligent Information Systems* **21**(3), 191–225.

Chun, S. A., Atluri, V. & Adam, N. R. (2002), Domain knowledge-based automatic workflow generation, *in* R. Cicchetti et al., eds, 'Proceedings of the 13th International Conference on Database and Expert Systems Applications', IEEE Computer Society, pp. 81–92.

Didion, J. (2003), *JWNL 1.3*. http://www.codezoo.com/pub/component/196?category=5.

Ehrig, M., Haase, P., Stojanovic, N. & Hefke, M. (2005), Similarity for ontologies - a comprehensive framework, *in* '13th European Conference on Information Systems', Regensburg.

Fellbaum, C., ed. (1998), *WordNet: An electronic lexical database*, MIT Press.

Gasevic, D. & Devedzic, V. (2004), Reusing petri nets through the semantic web., *in* C. Bussler, J. Davies & D. Fensel, eds, 'The Semantic Web: Research and Applications: First European Semantic Web Symposium', Vol. Lecture Notes in Computer Science, Springer, Heraklion, Crete, Greece, pp. 284–298.

Genrich, H. J. & Lautenbach, K. (1981), 'System modelling with high level petri nets', *Theoretical Computer Science* (13), 109–136.

Greco, G., Guzzo, A., Pontieri, L. & Saccà, D. (2004), An ontology-driven process modeling framework, *in* F. Galindo, M. Takizawa & R. Traunmüller, eds, '15th International Conference on Database and Expert Systems Applications', IEEE Computer Society, Zaragoza, Spain, pp. 13–23.

Hidders, J., Dumas, M., van der Aalst, W. M. P., ter Hofstede, A. H. & Verelst, J. (2005), When are two workflows the same?, *in* M. Atkinson & F. Dehne, eds, 'Proceedings of the 11th Australasian Theory Symposium', Vol. 41 of *Australian Computer Society*, Newcastle, Australia.

Horrocks, I. (2005), Applications of description logics: State of the art and research challenges., *in* 'Proceedings of the International Conference on Conceptual Structures', Lecture Notes in Computer Science, Springer, pp. 78–90.

HP (2003), *Jena 2.1 API*. Internet: http://jena.sourceforge.net/.

Jensen, K. (1994), An introduction to the theoretical aspects of coloured petri nets, *in* J. de Bakker, W. P. de Roever & G. Rozenberg, eds, 'A Decade of Concurrency – Reflections and Perspectives', Lecture Notes in Computer Science, Springer, pp. 230–272.

Kiepuszewski, B., ter Hofstede, A., & van der Aalst, W. (2003), 'Fundamentals of control flow in workflows', *Acta Informatica* **39**(3), 143–209.

Koschmider, A. & Oberweis, A. (2005), Ontology based business process description, *in* J. Castro & E. Teniente, eds, 'Proceedings of the CAiSE-05 Workshops', Lecture Notes in Computer Science, Springer, Porto, Portugal, pp. 321–333.

Lenz, K. & Oberweis, A. (2003), Interorganizational business process management with xml nets, *in* H. Ehrig, W. Reisig, G. Rozenberg & H. Weber, eds, 'Petri Net Technology for Communication-Based Systems, Advances in Petri Nets', Lecture Notes in Computer Science, Springer, pp. 243–263.

Levenshtein, I. (1966), 'Binary code capable of correcting deletions, insertions and reversals', *Cybernetics and Control Theory* **10**(8), 707–710.

Lin, D. (1998), An information-theoretic definition of similarity, *in* 'ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 296–304.

Maedche, A. & Staab, S. (2002), Measuring similarity between ontologies, *in* 'Proceedings of the European Conference on Knowledge Acquisition and Management', Lecture Notes in Computer Science, Springer.

McGuinness, D. L. & van Harmelen, F. (2003), OWL Web Ontology Language Overview, Technical report, World Wide Web Consortium (W3C). Internet: http://www.w3.org/TR/owl-features/.

Pahl, C. & Casey, M. (2003), Ontology support for web service processes, *in* 'Proceedings of the 9th European software engineering conference', ACM Press, Helsinki, Finland, pp. 208–216.

Reisig, W. & Rozenberg, G. (1998), *Lectures on Petri Nets: Basic Models*, Lecture Notes in Computer Science, Springer.

Resnik, P. (1999), 'Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language', *Journal of Artificial Intelligence Research* **11**.

Richter, M. M. (1992), Classification and learning of similarity measures, *in* Opitz, Lausen & Klar, eds, 'Proceedings der Jahrestagung der Gesellschaft für Klassifikation', Springer.

Scheer, A.-W. (1998), *ARIS - Business Process Modeling*, 2 edn, Springer, Berlin et al.

Staab, S. & Studer, R., eds (2004), *Handbook on Ontologies*, International Handbooks on Information Systems, Springer.

Tversky, A. (1977), 'Features of similarity', *Psychological Review* **84**(4), 327–352.

van der Aalst, W. M. P. & Weske, M. (2001), The P2P approach to interorganizational workflows, *in* 'Proceedings of the 13th International Conference on Advanced Information Systems Engineering', Lecture Notes in Computer Science, Springer, pp. 140–156.

Wu, Z. & Palmer, M. (1994), Verb semantics and lexical selection, *in* 'Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics', pp. 133–138.