Semantic annotations for semantic interoperability in a product lifecycle management context

Yongxin Liao, Mario Lezoche, Hervé Panetto, Nacer Boudjlida

HAL Id: hal-01286475

https://hal.archives-ouvertes.fr/hal-01286475

Submitted on 10 Mar 2016

Semantic Annotations for Semantic Interoperability in a Product Lifecycle Management Context

Yongxin Liao[1], Mario Lezoche[2,3], Hervé Panetto[2,3], Nacer Boudjlida[4]

[1]*Graduate Program in Production Engineering and Systems (PPGEPS), Pontifícial Universidade Católica do Paraná (PUCPR), Rua Imaculada Conceição, 1155, Curitiba, Paraná, Brazil.*

CNRS, CRAN UMR 7039, France

[3]*Université de Lorraine, CRAN UMR 7039, Boulevard des Aiguillettes, B.P. 70239 F-54506 Vandœuvre-lès-Nancy, France*

[4]*Lorraine Research Laboratory in Computer Science and its Applications (LORIA), Université de Lorraine, UMR 7503,*

The interoperability among a variety of systems, in or across manufacturing enterprises, has been widely accepted as one of the important factors that affect the efficiency of production. Many research works, related to the interoperability at different levels, have been carried out to tackle the information exchange, transformation, discovery and reuse. One of the main challenges in these research works is to overcome the semantic heterogeneity in the exchanged information between enterprise applications along the life cycle of a product. As a possible solution to support the semantic interoperability issue, semantic annotations have gained more and more attention. This paper identifies several existing drawbacks and proposes a formal semantic annotation approach to support the semantic enrichment of models in a product lifecycle management context.

Keywords: Semantic Annotation; Formalisation; Product Lifecycle Management; Enterprise Models; Information Systems; Semantic Interoperability

1. Introduction

Product Lifecycle Management (PLM) aims at providing a solution to improve the product competitive ability (Gunendran and Young 2010). It offers a shared platform to support the process of capturing, representing, retrieving and reusing knowledge, which brings together different enterprise systems at each stage of a Product Life Cycle (PLC) (Ameri and Dutta 2005). The knowledge concerning a product along its life cycle, which is named as PLC-related knowledge, has become one of the essential concepts in a PLM solution.

As an awareness gained by experience of a fact or a situation, knowledge brings to its owner the capability of grasping the meanings (semantics) from the received information (Ackoff 1989). It is a kind of intangible thing, which has to be made perceptible and afterward to be expressed under various kinds of representations. In this paper, all the relevant resources that are produced by different stakeholders through a variety of enterprise systems are considered as knowledge representations. They act as the carriers of PLC-related knowledge and as the basis to support the collaboration activities along the life cycle of a product.

The interoperability action plays a foundational role in collaboration activities. It implies that two or more systems are able to exchange data, information and how to use the functionalities of one another (IEEE 1991; Vernadat 1996; Chen, Doumeingts, and Vernadat 2008). The semantic interoperability is the ability to ensure that the exchanged information has got the same meaning considering the point of view of both the senders and the receivers (Sowa 1999). In a PLM context, the different peculiarities (such as the different backgrounds, the heterogeneous expertise, the unique knowledge, the particular needs and the specific practices) of stakeholders, who operate on those information systems, are then over increasing the difficulty to achieve the semantic interoperability (Etienne et al. 2011; Wang, Wong, and Wang 2011).

Therefore, to achieve a mutual understanding during collaboration, one of the possible solutions is to apply the semantic explication (Zdravković et al. 2013) of the exchanged knowledge representation by means of an ontology (Gruber 1993). A complementary way to realize this enrichment is the application of the semantic annotation (Bechhofer et al. 2002; Euzenat 2002; Uren et al. 2006; Boudjlida and Panetto 2007; Boudjlida and Panetto 2008).. On one hand, it attaches the formal and

shared terms between the stakeholders to make semantics explicit and on the other hand, it brings the possibility to perform a semantic reasoning.

Concerning the research environment (a PLM context) and the focus (dealing with semantic interoperability issues), and also taking into account the research interests of this work and the domain of expertise, three hypotheses need to be made before the identification of the addressed problems and the proposition of the solution.

Hypothesis 1: *All the knowledge that is needed for the semantic enrichment of models has already been captured, represented and formalised into ontologies*. Above all, ontology is based on the open world assumption theory, which means the missing information in ontologies is treated as unknown (Antoniou and van Harmelen 2009). This hypothesis does not intend to state that all the knowledge present in the world must be captured, represented, or formalised. It only highlights that there is a minimum amount of knowledge whose presence is mandatory to achieve some specific goals. The research communities which worked on knowledge formalisation (Gruber 1993), discovery (Polanyi 1967), and conversion (Nonaka 1994), can provide supports for this hypothesis. Therefore, to carry out the process of semantic annotation, the knowledge that is required should be prepared (captured, represented and formalised) in advance.

Hypothesis 2: *The corresponding interconnections among all the used ontologies have already been prepared through appropriate methods*. An ontology provides the formalisation of certain knowledge that can exist for an agent or a community of agents (Gruber 1993). This characteristic also reflects that each ontology, which is created by different agents, might have its own way of formalisation and unique knowledge focus. To use two or more ontologies during the semantic annotation process, their interconnections should also be prepared (integrated or coordinated) in advance. Taking advantages from the research works about ontology matching (Euzenat

and Shvaiko 2007), mapping (Doan et al. 2003), and merging (Stumme and Maedche 2001), this hypothesis can be achieved.

Hypothesis 3: *The semantic similarity between two objects can be compared through certain methods*. Ontology is originally a branch of philosophy study that seeks to provide a definitive and exhaustive classification of entities in all spheres of being (Smith 2003). Since 1980s, computer and information scientists began to adopt this term and represent it as a set of concepts, their relationships and the axioms on both objects to model the world (Gruber 2009). As the backbone of an ontology, the taxonomy classifies concepts based on their common features (Noy and McGuinness 2001). The classification is done depending on the similarity comparisons among those concepts (Markman and Gentner 1993). A number of research works have already contributed in the evaluation of the semantic similarities, such as (Rong et al. 2006; Schwering and Kuhn 2009). As it is stated in (Markman and Gentner 1996), the similarity comparisons are ubiquitous in cognitive processing. Even if there is no automatic similarity comparison, at least, this process can be done with the participation of domain experts.

Based on these three hypotheses, the objective of the paper is to provide an extension of previous publications of the authors (Liao et al. 2014a, 2014b), through a more complete description of the semantic annotation method and a more detailed presentation of the case study. Among all the improvements in each section, several major ones are listed as examples. In Section 3, the necessary remarks and improvements of the formal definitions about the semantic annotation are firstly provided. After that, one of the important mechanisms for the semantic annotation suggestion is described in details. In Section 4, the former case study in (Liao et al. 2014a) is rearranged to be more understandable and detailed in each step. The extended

contents in the case study includes: the pre-processing of the five employed ontologies, the data structure of the semantic annotation schema, the examples of the suggestion and verification rules, the graphical demonstrations of semantic annotations, and the walkthrough examples of the annotation suggestion and similarity comparison.

2. Related Works

The enterprise modelling is a process that focuses on capturing and representing knowledge from the perspective of a system of interest, which helps organisations to model and understand their key components and the relationships among them (Leondes and Jackson 1992; Vernadat 2002). In this research work, all the different types of models along a PLC are considered as a target of the semantic enrichment. For example, the data models (Feng 1995), the product design models (Bugtai and Young 1998), the process models (Curtis, Kellner, and Over 1992), and so on. The interoperations among the systems not only require that models can be exchanged and operated on, but also demand an unambiguous understanding of the semantics inside those models.

While a model is an abstract representation of a thing or a piece of reality, from the representation point of view "*is often presented as a combination of drawings and text.*" (OMG 2013). The mutual understanding of a model requires the semantics of both the "*combination of drawing*" and the "*text*". Consequently, there are two important aspects of the semantics that are needed to be made explicit through a semantic annotation: (1) The s*tructure semantics*, which describes the interrelations between an annotated element and the other elements that are related to it; (2) The *domain semantics*, which describes the context and the meaning of an annotated element in a specific domain. Therefore, two kinds of ontologies are categorized to cover both aspects: The meta-model ontologies and the PLC-related ontologies. The former category represents the knowledge of how models are constructed. The latter category

represents the knowledge that is related to a product during one or more stages of its life cycle. In this paper, the semantic annotation is considered as a means that employs the Meta-model and the PLC-related ontologies to make explicit both structure and domain semantics of annotated enterprise models.

After the investigation of 135 related works (Liao et al. 2015), despite lots of efforts have been made, at least three existing drawbacks should be quoted:

(1) Lack of the formalisation of semantic annotations which is suitable to be used in the semantic enrichment of different kinds of models along a PLC;

(2) Lack of mechanisms which can combine both structure semantics and domain semantics together to contribute in the reasoning phase;

(3) Lack of mechanisms to support the verification of the semantic consistency between two semantic annotations on a single annotated element.

To deal with these three identified drawbacks, a formal semantic annotation approach and its corresponding suggestion and verification mechanisms are proposed in the next section.

3. Proposed Solution

3.1 The Formalisation of The Semantic Annotation

This section extends the previous publication (Liao et al. 2014a) through a more complete semantic annotation method. In Section 3.1.1 and Section 3.1.2, the meta-model of the semantic annotation and the concept of "Semantic Block" are simply recalled to identify the essential elements in this research work. Then, in Section 3.1.3, the formal definitions with necessary remarks and improvements are given as part of the extension.

3.1.1 Meta-model of the Semantic Annotation

The models are always expressed in some kinds of modelling languages with designer's specific peculiarities. In this research work, enterprise models are named as Target Knowledge Representations (TKRs) for the semantic enrichment. The ontology formalises a real-world semantics that enables humans to use meaningful terminologies for contents that can be processed by machines. In this research work, the PLC-related ontologies and the Meta-model ontologies are considered as an Ontology-based Knowledge Representations (OKRs) to support the semantic enrichment of TKRs. The Semantic Annotation acts as an interface to formally describe the semantic relationships between TKRs and OKRs.

The meta-model of the semantic annotation is shown in Figure 1, which describes the main components of a semantic annotation and their relationships. Several important concepts that are used throughout this paper are described as follows:

(1) A "*Target Knowledge Representation*" *(TKR)* is the composition of one or more "*Element(s) of a TKR*";

(2) The "*Ontology-based Knowledge Representation*" *(OKR)* is the generalization of the "*Meta-model Ontology*" and the "*PLC-related Ontology*";

(3) A "*Meta-model Ontology*" is the composition of one or more "*Element(s) of a Meta-model Ontology*";

(4) A "*PLC-related Ontology*" is the composition of one or more "*Element(s) of a PLC-related Ontology*";

(5) An "*Element of a TKR*" can be annotated by zero or more "*Semantic Annotation(s)*";

(6) A "*Semantic Annotation*" contains one "*Structure Semantics*";

(7) A "*Semantic Annotation*" contains zero or more "*Domain Semantics*";

(8) A "*Structure Semantics*" is the aggregation of one "*Element of a Meta-model Ontology*";

(9) A "*Domain Semantics*" is the aggregation of one or more "*Element(s) of a PLC-related Ontology*".

3.1.2 The Concept of "Semantic Block"

The concept of "Semantic Block" is adopted from the research work of Yahia et al. (2012) in which, a semantic block is composed by a minimal number of mandatory concepts that are needed to express the full semantics of an appointed concept. In this research work, this definition is extended to cover also the relationships among those selected concepts.

Generally, both ontologies and enterprise models can be regarded as the composition of an entity set (such as concepts in the ontologies or elements in the models) and the corresponding explicit or implicit relations that bind those entities together. Two kinds of semantic blocks are categorized based on their objectives:

(1) *The Semantic Blocks for Semantics Description*: They are used for the creation of a "*Domain Semantics*" through the delimitation of one or more "*Element(s) of a PLC-related Ontology*" from one or more "*PLC-related Ontology (Ontologies)*". The generated semantic block can be used to describe the domain semantics of an "*Element of a TKR*";

(2) *The Semantic Blocks for Semantics Substitution*: They are used for the creation of a substitute through delimitating one or more "*Element(s) of a TKR*" from one "*Target Knowledge Representation*" based on the "*Structure Semantics*" that these elements express. These substitutes act as new entities or new relations to support the creation of inference rules and semantic reasoning.

The detailed mathematical expression of the semantic block delimitation methods can be found in (Liao et al. 2014a) and their corresponding examples (for both the semantics description and the semantic substitution) will be given in Section 3.2.1.

3.1.3 Formal Definitions about the Semantic Annotation

Based on the formalisation in (Liao et al. 2014a), the necessary remarks and improvements along with the formal definitions are presented in this section.

Let E be the set of elements in a TKR and e_i be one of the elements in E.

Definition 1. An ontology is a formal and shared understanding of a domain of interest. It specifies the concepts and the relationships that can exist for an agent or a community of agents (Gruber 1993; Uschold and Gruninger 1996). Let o_x represent an ontology, which is formalised by a triple:

$$o_x := (C_{o_x}, R_{o_x}, A_{o_x}),$$

where C_{o_x} is a set of concepts; R_{o_x} is a set of relationships; A_{o_x} is a set of rule axioms. Let $oall_{o_x}$ be the set that contains all the elements from the set C_{o_x} and R_{o_x}. An ontology element $oe_{o_{xy}}$ is represented as:

$$oall_{o_x} = \{oe_{o_{xy}} | oe_{o_{xy}} \in C_{o_x} \cup R_{o_x}\}.$$

Remark 1. The ontology elements that are used as part of annotation contents are the concepts and the relationships. The rule axioms only participate in the reasoning phase.

Definition 2. A meta-model is a model that specifies the concepts, the relationships and the rules on how to construct a model. Let mm_x denote a meta-model, which is defined as a triple:

$$mm_x := (C_{mm_x}, R_{mm_x}, RU_{mm_x}),$$

where C_{mm_x} is a set of concepts; R_{mm_x} is a set of relationships; RU_{mm_x} is a set of rules.

Let mmo_x be an ontology that represents the meta-model mm$_x$, which is defined as:

$$mmo_x := (C_{mmo_x}, R_{mmo_x}, A_{mmo_x}).$$

Remark 2. In the scientific literature, there are at least two different visions about whether a meta-model is an ontology or not. In this paper, only those elements (concepts and relationships) that are needed for describing the interrelations among the annotated elements are taken into account. Therefore a meta-model can be considered and represented as an ontology.

Definition 3. The domain semantics of a TKR is made explicit by one or more PLC-related ontology (ontologies). Let PO be the set of PLC-related ontologies and P be the set of selected ontology element sets from the powerset of all ontology elements of PO, which is defined as:

$$\bigcup_{o_x \in PO} oall_{o_x} = \{oe_{o_{x_y}} | (\exists o_x)(o_x \in PO \wedge oe_{o_{x_y}} \in oall_{o_x})\},$$

$$P \subseteq \mathcal{P}\left(\bigcup_{o_x \in PO} oall_{o_x}\right).\,^1$$

Remark 3. In a different way from other semantic annotation methods, each e_i from a TKR can be annotated by a set of ontology elements that are delimitated by a semantic block (for semantics description), which contains concepts and relationships from one or more PLC-related ontology (ontologies).

Definition 4. The structure semantics of a TKR is made explicit by a meta-model ontology mmo_x. Let MME be the set that contains all the elements from the set C_{mmo_x}. An ontology element mme_l is defined as:

---

[1] In the previous publication (Liao et al. 2014a), it is $P \subseteq \mathcal{P}\left(\bigcup_{o_x \in O} oall_{o_x}\right)$.

$$MME := \{mme_l \mid mme_l \in C_{mmo_x}\}.$$

Remark 4. Each e_i from a TKR can be annotated by one ontology element from the C_{mmo_x} of a meta-model ontology mmo_x. The relationships in R_{mmo_x} are used for defining the relationships between the annotated elements in the TKR.

Definition 5. Let A and B be two sets, any subset of $br \subseteq A \times B$ is a binary relation from A to B. Given $a \in A$ and $b \in B$, the br in the notation $a \; br \; b$ is defined as,

$$br := \{(a,b) \mid a \text{ is in the relation } br \text{ with } b\}.$$

Let $dom(br)$ represents the domain of the br and $ran(br)$ represents the range of the br, which are defined as

$$dom(br) := \{a \in A \mid \exists b \in B, (a,b) \in br\},$$

$$ran(br) := \{b \in B \mid \exists a \in A, (a,b) \in br\}.$$

Definition 6. $SR_{E,P}$ is a set of binary relations that describe the semantic relationships from E to P. Given, $e_i \in E$ and $p_j \in P$, and let $sem(e_i)$ represents the semantics of e_i and $sem(p_j)$ represents the semantics of p_j, five subsets of the $SR_{E,P}$ are defined as follows:

$$sr_{\sim} := \{(e_i, p_j) \mid sem(e_i) \text{ and } sem(p_j) \text{ are equivalent}\};$$

$$sr_{\supset} := \{(e_i, p_j) \mid sem(e_i) \text{ is more general than } sem(p_j)\};$$

$$sr_{\subset} := \{(e_i, p_j) \mid sem(e_i) \text{ is less general than } sem(p_j)\};$$

$$sr_{\cap} := \{(e_i, p_j) \mid e_i \text{ and } p_j \text{ have common semantics}, (e_i, p_j) \notin sr_{\sim} \cup sr_{\supset} \cup sr_{\subset}\};$$

$$sr_{\perp} := \{(e_i, p_j) \mid e_i \text{ and } p_j \text{ have not common semantics}\}.$$

Remark 6. Five semantic relationships are proposed to describe the relations from an "*Element of a TKR*" to a "*Domain Semantics*": "*is equivalent to*" ($sr_{\backsim}$), "*subsumes*" ($sr_{\supset}$), "*is subsumed by*" ($sr_{\subset}$), "*intersects*" ($sr_{\cap}$), and "*is disjoint with*" ($sr_{\perp}$).

Definition 7. $MR_{E,MME}$ is a set of binary relations that describe the semantic relations from E to MME. Given $e_i \in E$ and $mme_l \in MME$, one subset of $MR_{E,MME}$ is defined as follow:

$$mr_{io} := \{(e_i, mme_l)| \ e_i \ is \ an \ instance \ of \ mme_l\}.$$

Remark 7. One semantic relationship is proposed to describe the relation from an "*Element of a TKR*" to a "*Structure Semantics*": "*is instance of*" (mr_{io}).

Finally, with all the above-mentioned definitions, the formal definition of the semantic annotation can be presented.

Definition 8. Let TKR, PO and mmo_x be given, the semantic annotation SA that is associated to them is defined by a 5-tuple:

$$SA := (E, P, SR, MME, MR),$$

where

- E is a set of elements from a TKR;

- P is a set of selected ontology element sets from a set of PLC-related ontologies PO, which makes explicit the domain semantics aspect of E;

- MME is a set of ontology elements from a meta-model ontology mmo_x, which makes explicit the structure semantics aspect of E;

- $SR := SR_{E,P}$;

- $MR := MR_{E,MME}$.

These formal definitions, on the one hand, can be used as a basis to construct a semantic annotation schema. On the other hand, they also can be used to support the application of semantic reasoning.

3.2 Reasoning Mechanisms

In this work, the semantic reasoning process is decomposed into three main stages as follows: (1) The semantic annotation suggestion (Section 3.2.1, one of the important extensions); (2) The inconsistency detection between the semantic annotations on a common annotated element (Section 3.2.2); and (3) The conflict identification between annotated elements (Section 3.2.2).

3.2.1 The Semantic Annotation Suggestion

The concept "Inconsistency" refers to any potential contradiction among two or more facts that describe one common object. Therefore, the inconsistency detection needs the comparison of two or more semantic annotations that describe the semantics of one "Element of a TKR". To cope with this principle, two types of semantic annotations are classified:

(1) *The Initial Semantic Annotations*, which are directly annotated on an "*Element of a TKR*" by an annotator;

(2) *The Inferred Semantic Annotations*, which are suggested to annotate an "*Element of a TKR*" through an inference action, which is based on its related element's semantic annotations and suggestion rules.

Both "*Structure Semantics*" and "*Domain Semantics*" are contributing in the suggestion stage. The "*Structure Semantics*" is used to make explicit the implicit relations between an annotated "*Element of a TKR*" and its related elements. Let *E* be a

set of elements in the TKR and let mmo_x be the meta-model ontology. The following steps are used to create a semantic block for semantics substitution:

(1) Let the elements in E be annotated by the concepts in C_{mmo_x}. Through the semantic relationship mr_{io}, these annotated elements are treated as instances of their corresponding concepts. The interrelations between two related instances are made explicit through the corresponding relationships in R_{mmo_x}.

(2) The selection of two concepts, $moc_i, moc_j \in C_{mmo_x}$, for the creation of the semantic block SBR_{moc_i,moc_j}.

(3) The selection of a set of concepts $A_{moc_i,moc_j} = \{moc_1 \dots moc_{n'}\} \subset C_{mmo_x}$. The selection of a set of relationships $B_{moc_i,moc_j} = \{mor_1 \dots mor_{m'}\} \subseteq R_{mmo_x}$, which are the relationships among the selected concepts in A_{moc_i,moc_j}. This selection process needs to satisfy the following three conditions:

(a) $moc_i, moc_j \notin A_{moc_i,moc_j}$;

(b) $\forall moc_k \in A_{moc_i,moc_j}, \exists moc_l \in A_{moc_i,moc_j}, \exists mor_z \in B_{moc_i,moc_j},$

$s.t.\ (moc_k, moc_l) = mor_z$;

(c) $\forall mor_{z'} \in B_{moc_i,moc_j}, (moc_{k'}, moc_{l'}) = mor_{z'} \Rightarrow moc_{k'}, moc_{l'} \in A_{moc_i,moc_j} \cup \{moc_i\} \cup \{moc_j\}.$

(4) Finally, the rule to delimitate SBR_{moc_i,moc_j} can be created as follows:

$$mr_{io}(?\,a, moc_i), mr_{io}(?\,b, moc_1), \dots, mr_{io}(?\,c, moc_n), mr_{io}(?\,d, moc_j),$$

$$mor_1(?\,a, ?\,b), \dots, mor_m(?\,c, ?\,d) \rightarrow SBR_{moc_i,moc_j}(?\,a, ?\,d).$$

SBR_{moc_i,moc_j} acts as a new relation from certain instances of moc_i to certain instances of moc_j which fulfil all the conditions in the delimitation rule. Because there

is a possibility to have several different combinations of concepts and relationships, moc_i and moc_j might have multiple SBR_{moc_i,moc_j}. In this case, they should be named differently.

The "*Domain Semantics*" is used as the range of semantic block delimitation traversal in this stage. Let E be a set of elements in the TKR and let PO be the set of PLC-related ontologies for making explicit the domain semantics of the TKR. The procedure to suggest a semantic annotation is explained below:

(1) Let $e_x, e_y \in E$ be two elements in the TKR. e_x is an instance of moc_i and e_y is an instance of moc_j. Let's assume that besides the interrelations, which are made explicit by the corresponding relationships in R_{mmo_x}, e_x and e_y have a new relation SBR_{moc_i,moc_j} between them.

(2) Let $poc_{i'}$ be a concept from PO which acts as the main concept of the semantic block $SB_{poc_{i'}}$. Let $A_{poc_{i'}} = \{poc_1 \dots poc_{n'}\}$ be the set of selected concepts from in $SB_{poc_{i'}}$. Let $B_{poc_{i'}} = \{por_1 \dots por_{m'}\}$ be the set of relationships in $SB_{poc_{i'}}$.

(3) Let $e_x \in E$ be annotated by $SB_{poc_{i'}}$ through the semantic relationship $sr_\sim$ or $sr_\subset$.

(4) Selecting a relation $por_{z'} \in B_{poc_{i'}}$ and associating it with the SBR_{moc_i,moc_j}. If there is a concept $poc_{j'}$ that satisfies $(poc_{i'}, poc_{j'}) = por_{z'}$, a new semantic block $SB_{poc_{j'}}$ that takes $poc_{j'}$ as its main concept can be generated. The traversal, which builds $A_{poc_{j'}}$ is based on $A_{poc_{i'}}$ and $B_{poc_{i'}}$, is as follows:

$A_{poc_{j'},0} = \{poc_{j'}\};$

$$A_{poc_{j'},1} = \{a_{i_1} \in A_{poc_i'} \mid \exists b_{i_0,i_1} \in B_{poc_i'}, a_{i_0} \in A_{poc_{j'},0}, (a_{i_0}, a_{i_1}) = b_{i_0,i_1}\};$$

$$A_{poc_{j'},2} = \{a_{i_2} \in A_{poc_i'} \mid \exists b_{i_1,i_2} \in B_{poc_i'}, a_{i_1} \in A_{poc_{j'},1}, (a_{i_1}, a_{i_2}) = b_{i_1,i_2}\};$$

...

$$A_{poc_{j'},n} = \{a_{i_n} \in A_{poc_i'} \mid \exists b_{i_{n-1},i_n} \in B_{poc_i'}, a_{i_{n-1}} \in A_{poc_{j'},n-1}, (a_{i_{n-1}}, a_{i_n}) =$$

$$b_{i_{n-1},i_n}\};$$

$$A_{poc_{j'}} := \bigcup_n A_{poc_{j'},n}$$

Let $B_{poc_{j'}} \subseteq B_{poc_i'}$ be the set of relations that appear during the creation of

$A_{poc_{j'}}$, then $SB_{poc_{j'}}$ is created.

(5) Finally, $SB_{poc_{j'}}$ is suggested to annotate e_y with the semantic relationship $sr_{\subset}$.

In this work, the semantic blocks that are nested within each other are not taken into account.

3.2.2 The Inconsistency Detection and Conflict Identification

Once an annotated element receives two or more semantic annotations, the inconsistency detection can be performed. Let's take the case of inconsistency detection between two semantic annotations as an example. Let e_i be annotated by sa_x and sa_y, in which, p_x and p_y are used to make explicit the domain semantics of e_i. The semantic similarity comparison results between p_x and p_y are defined as follows:

Definition 9. PR is a binary relation that describes the semantic relationships from P to P. Given $p_x, p_y \in P$, and let $sem(p_y)$ represents the semantics of p_x and $sem(p_y)$ represents the semantics of p_y, five subsets of PR are defined as follows:

$$pr_{\sim} := \{(p_x, p_y) \mid sem(p_x) \text{ and } sem(p_y) \text{ are equivalent}\};$$

$$pr_{\supset} := \{(p_x, p_y) \mid sem(p_x) \text{ is more general than } sem(p_y)\};$$

$$pr_\subset := \{(p_x, p_y) | sem(p_x) \ is \ less \ general \ than \ sem(p_y)\};$$

$$pr_\cap := \{(p_x, p_y) | \ p_x \ and \ p_y \ have \ common \ semantics, (p_x, p_y) \notin pr_\sim \cup pr_\supset \cup pr_\subset\};$$

$$pr_\perp := \{(p_x, p_y) | \ p_x \ and \ p_y \ have \ not \ common \ semantics\}.$$

The results of the inconsistency detection method point out the (possible) inconsistencies between two semantic annotations of a common annotated element. Three types of possible results are suggested as follows: (1) Result one expresses that p_x and p_y are consistent with each other; (2) Result two expresses that p_x and p_y are possibly consistent with each other; and (3) Result three expresses that there is an inconsistency between p_x and p_y.

Based on the inconsistency detection results, the possible conflicts between those annotated elements in a TKR can also be identified. The detail strategies for these two mechanisms are presented in (Liao et al. 2014a).

4. Case Study

Based on the proposed formalisation, a prototype annotation tool, Semantic Annotation Plugin for Knowledge Management (SAP-KM), was developed to assist the annotation and reasoning process in the case study. This section presents the major part of the extension. A brief introduction of the case study background is firstly given in Section 4.1. Further, Section 4.2 presents the utilisation of formal semantic annotations in this application scenario with detailed and necessary explanations, which include: (1) The pre-processing of the five employed ontologies; (2) The data structure of the semantic annotation schema; (3) The examples of the suggestion and verification rules; (4) The graphical demonstrations of semantic annotations; and (5) The walkthrough examples of the annotation suggestion and similarity comparison.

4.1 The Case Study Context

The life cycle of an educational combination product (Gouyon 2004) that is made in a technical production centre, named AIPL[2], has been chosen as the context of this case study. The requirements of this product come from the needs of reusability of the educational materials. The mechanical engineers at AIPL designed this product through the CATIA Computer-Aided Design software[3] (CATIA), which generates the product technical information into a so-called Engineering Bill of Material (EBOM). However, the EBOM represents the product structure from the designer's point of view, which does not contain all the information that is needed by the systems at the production stage.

For this reason, a Bill of Process (BOP) needs to be combined together with EBOM. These processes are defined and modelled via the MEGA modelling environment[4] (MEGA), which contains: (1) The *Bases turning process*, which chips an aluminium bar into designed bases; (2) *The Discs cutting process*, which cuts galvanized plates and magnetic plates into a number of designed discs; (3) The *Parts sticking process*, which uses glues to stick the galvanized or magnetic discs to the corresponding bases for producing four kinds of designed parts; (4) The *Products assembling process*, which assembles those parts into six types of the designed products.

Then EBOM and BOP are used as basis to support the parameterization of the systems in the production stage. An example of that utilisation is the Sage X3 ERP system[5] (Sage X3). It takes customer orders as inputs and generates orders to support

---

[2] AIPL (Atelier Inter-Etablissements de Productique Lorrain): http://www.aip-primeca.net/
[3] CATIA Computer-Aided Design software: http://www.3ds.com/products-services/catia/
[4] MEGA modelling environment: http://www.mega.com/
[5] Sage X3: http://www.sage.com/

purchasing, outsourcing and manufacturing processes. At the end, after quality examinations, the qualified products are packed into boxes and dispatched to the production engineering teaching group.

To determine a clear-cut information flow and to show the interoperation between those systems, three kinds of systems are classified: (1) the c*urrent system*, which is being used in a selected point on that information flow; (2) the *upstream system* which is placed before the current system; (3) on the contrary, the *downstream system* which is placed after the current system. As it is shown in Figure 2, the MEGA application is chosen as the current system, together with its upstream system (CATIA) and downstream system (Sage X3) as an application scenario. They represent the *TKR creation and management* module. In this module, the owners of those TKRs are also assigned for annotating and managing the corresponding semantic annotations. For the other parts of the figure, the Protégé Ontology Editor[6] is used as the *OKR Creation and Management* module, which is in charge of capturing, formalising and managing the PLC-related knowledge and the model construction knowledge into a knowledge base, namely, *Knowledge Cloud*. The Microsoft windows folder system is used as the *Knowledge Cloud* module, which is used as a knowledge repository to store OKRs, semantic annotations, and reasoning rules. The SAP-KM acts as the *Semantic Annotation and Processing Agency*. The Jena Reasoner[7] is employed as the *Reasoning Engine* module to support the inferences on the OKRs and semantic annotations based on the reasoning rules. For the rest of the four modules, domain experts with knowledge formalisation and reasoning skills are required. Besides ensuring the normal functioning of each module, they are also assigned to the task of training the staffs in the *TKR*

---

[6] Protégé Ontology Editor: http://protege.stanford.edu
[7] Jena Reasoner: http://jena.apache.org/documentation/inference/

creation and management module. They should provide them with enough supports about how to apply the semantic annotations based on the knowledge that they captured, formalised, and managed.

In the current version of the SAP-KM, there are two developed interfaces. One is between the MEGA application and the SAP-KM to assist the annotation on the model diagram, and the other one is between the Jena Reasoner and SAP-KM to support the reasoning processes. SAP-KM can also communicate with the *Knowledge Cloud* module to perform operations on the rules and existing semantic annotations. Because this research work is not focusing on the creation of the OKRs, the interface between SAP-KM and Protégé is not considered as part of the implementation. Also, because several research literatures have already showed the possibility of developing annotation plug-ins for product the design models (Attene et al. 2009; Li 2012) and the data models (Song 2013; Bergamaschi et al. 2011). To avoid the unnecessary repetition with those works, the interfaces between CATIA/Sage X3 and SAP-KM are not developed. In the case study, the corresponding plug-ins for the upstream system and the downstream system are assumed existing and they follow the proposed formalisation.

4.2 The Application of Formal Semantic Annotations

4.2.1 The Preparation Phases

Concerning the step of the *creation of a TKR*, a process model created by the MEGA (current system) is considered as the target of the semantic enrichment. As stated in the previous section, the product model from CATIA (upstream system) is considered as a model that is passed to the current system and already contains semantic annotations. As it can be seen from Figure 3, the "*Bases Turning*" operation in the WorkCentre US,

which is in charge of machining the "*Aluminium Bars*" and producing two kinds of bases (P0110 and P0960), is chosen as the example in this case study.

Concerning the *Collection and Formalisation of OKRs* step, two domain level ontologies (the MSDL ontology (Ameri et al. 2011) and the BPMN ontology (Ghidini, Rospocher, and Serafini 2008)) are employed. Based on these ontologies, a top-level ontology (the general ontology) and two application-level ontologies (the AIPL product ontology and the MEGA BPMN ontology) are created to fulfil the needs of annotation from different levels. Meanwhile, the pre-process on these five ontologies is carried out as follows:

(1) The creation of interconnections among ontologies. For example, the Object Property "*hasShape*" is added from the Individual "*P0110*" (AIPL Ontology) to the Individual "*Cylinder*" (MSDL Ontology);

(2) The completion of the top-level hierarchy. For example, a set of "*subClassOf*" is added from the second-level classes to the top level Class "*Thing*", which is omitted by Protégé;

(3) Semantic enrichment of existing ontologies. For example, as it can be seen from Table 1, the embedded semantics of the concept "*Turning*" (MSDL Ontology) and the concept "*Bases Turning*" (AIPL Ontology) are enriched at both the domain level and the application level;

(4) The RDF/XML Reformation. To facilitate the ontology loading process in Jena Reasoner, these five ontologies are restored in RDF/XML Syntax (W3C 2004).

Concerning the *Customization of the SA solution* step, based on the proposed formalisation in Section 3, a schema is designed to store the annotation results. To use the existing reasoning engines, this schema is constructed in the ontological format,

named *Semantic Annotation Schema*. The E, P and *MME* in the formal definition of the semantic annotation are represented as three disjoint Classes, named "*E*", "*P*" and "*MME*" respectively. To be more specific, the data structure for each of them is defined as follows:

(1) e_i is represented as an Individual of the Class "*E*". An example of Class "*E*" and its Individuals is shown in Figure 4 (a).

(2) PO is a number of PLC-related ontologies from the knowledge cloud, which are imported into the *Semantic Annotation Schema*. p_j is represented as an Individual of the Class "*P*". An example of Class "*P*" and its Individuals is shown in Figure 4 (b). Its related concepts and relationships are listed as follows:

- The Object Property "*hasMainConcept*" is the relationship between a p_j and the selected main concept (a Class or an Individual) of a semantic block.

- The Object Property "*hasSBEntity*" is the relationship between a p_j and the rest of the selected concepts (Classes or Individuals) of a semantic block.

- The Class "*SBRelations*" is used to store all relations in the semantic blocks. Each relation in this Class describes a relationship between two selected concepts in a semantic block.

- The Object Property "*hasSBRelation*" is the relationship between a p_j and an Individual in the Class "*SBRelations*".

- The Class "*NSstore*" is used to store all the namespace abbreviations and the complete terms that they stand for.

- The Datatype Property "*hasLongNS*" is the relationship between a namespace abbreviation and its full ontology namespace.

(3) mmo_x is a meta-model ontology from the knowledge cloud, which is imported into the *Semantic Annotation Schema*. mme_l is a Class in the mmo_x and the Class "*MME*". The model instances of mme_l are Individuals in the Class "*E*". An example of the Class "*MME*" and its sub-classes is shown in Figure 4 (c).

The *SR* and *MR* in the formal definition of the semantic annotation are represented as two Object Properties, named "*SR*" and "*MR*" respectively. To be more specific, their data structure is shown as follows:

(1) The Class "*E*" and "*P*" are defined as the domain and range restrictions of the Object Property "*SR*" respectively. The Object Property "*SR_isEquivalentTo*" ($sr_\sim$), "*SR_subsumes*" ($sr_\supset$), "*SR_isSubsumedBy*" ($sr_\subset$), "*SR_intersects*" ($sr_\cap$), and "*SR_isDisjointWith*" ($sr_\perp$) are its sub properties and inherit its domain and range restrictions. One example of "*SR_isSubsumedBy*" is shown in Figure 4 (d).

(2) mr_{io} is supposed to be represented as a sub property of the Object Property "*MR*". However, for efficiency reason during the implementation, the default relationship "rdf:type" is employed to represent mr_{io}. It is used to describe the semantic relationship between an Individual of the Class "*E*" and a sub Class of the Class "*MME*". One example of mr_{io} is shown in the Figure 4 (e).

Furthermore, in order to assist the reasoning process and simplify the expression of reasoning rules, the corresponding concepts and relationships in the proposed reasoning mechanisms are also expressed in the *Semantic Annotation Schema*. The *PR* is represented as an Object Property "*PR*". The class "*P*" is defined as its domain and range restrictions. The Object Property "*PR_isEquivalentTo*" ($pr_\sim$), "*PR_subsumes*" ($pr_\supset$), "*PR_isSubsumedBy*" ($pr_\subset$), "*PR_intersects*" ($pr_\cap$) and "*PR_isDisjointWith*" ($pr_\perp$) are defined as its sub properties and they inherit its domain and range restrictions. The

inconsistency detection results are represented as the Object Property "*isConsistentWith*", "*isPosConsistentWith*" and "*isNotConsistentWith*". The Class "*P*" is defined as their domain and range restrictions. The conflict detection result is represented as an Object Property "*isConflictWith*". The Class "*E*" is defined as its domain and range restrictions. The Object Property "*isAnnotatedBy*" denotes that the Individual in its range annotates the Individual in its domain. The Class "*E*" and "*P*" are defined as its domain and the range restrictions respectively. The Object Property "*isInferredFrom*" denotes that the Individual in its domain, which is a semantic annotation, is inferred from the domain semantics of the Individual in its range. The Class "*P*" and "*E*" are defined as its domain and range restrictions respectively.

There is, of course, no limit to this data structure. Based on the proposed formalisation, the *Semantic Annotation Schema* can be designed differently for adopting different kinds of requirements.

4.2.2 The Annotation Phase

Once the TKR, OKRs and *Semantic Annotation Schema* are prepared, the annotation process can be performed. In this case study, semantic annotations are divided into two parts: the received semantic annotations from the upstream system and the created semantic annotations in the current system. Therefore, concerning the explicitation of the domain semantics, there is two possibilities: (1) To reuse the domain semantics from the imported semantic annotations through the element matching function, which declares that two matched elements have the same domain semantics; (2) To create the new domain semantics and the new structure semantics for the selected model elements.

To avoid the massive details for each semantic annotation and also to facilitate the explanation and reading, a semantic annotation is represented in the syntax of "namespace; local name". The ontology abbreviations are as follows: General Ontology

as &GO, MSDL Ontology as &MSDL, BPMN Ontology as &BPMN, AIPL Product Ontology as &AIPL, MEGA BPMN Ontology as &MEGA and *Semantic Annotation Schema* as &SANS. The "local name" can be the local name of a Class, an Individual or a Property.

The semantic annotations from the upstream system are imported into the *Semantic Annotation Schema*. To reuse them, the element matching process is carried out. For example, Figure 5 shows the domain semantics of the e_1 ("*bar*") in the product design model, which is then matched to the e_{11} ("*Aluminium Bars*") in the process model. After the matching process, the elements in the process model inherit the domain semantics of the matched elements from the product design model.

In the current system, as an example, concerning the explicitation of the domain semantics, as it is shown in Figure 6 (a), the domain semantics of the "*Bases Turning*" operation is made explicit thanks to the MSDL Ontology and the AIPL Product Ontology. Concerning the explicitation of the structure semantics, as it is shown in Figure 6 (b), the internal relations between three selected elements in the process model are made explicit thanks to the BPMN Ontology and the MEGA BPMN Ontology.

All the semantic annotations, which are recorded in the *Semantic Annotation Schema*, are used as the basis to support the suggestion and verification in the reasoning phase.

4.2.3 The Reasoning Phases

The reasoning phase is mainly in charge of suggesting inferred new semantic annotations, detecting inconsistencies between semantic annotations of a common annotated element, and identifying the possible conflicts between annotated elements.

In Figure 7 (a), a semantic block delimitation rule, which is used to make explicit the internal relations from an operation to the data objects that act as its inputs, is illustrated as an example. After the execution of this rule, as it is shown in Figure 8 (in its upper part), the Object Property "*SBR_Operation_hasinput_DataObject*" is added from the operation "*Bases Turning*" to the data object "*Aluminium Bars*". Then, this new added Object Property (from the structure semantics) is associated to the Object Property "*&MSDL;hasInput*" and the Object Property "*&AIPL;hasInput*" (from the domain semantics) respectively. The corresponding inferred semantic annotations (p_{18} and p_{17}) for the "*Aluminium Bars*" (e_{11}) are suggested after the association, as it is shown in Figure 8 (in its lower part), based on (1) the initial semantic annotations (p_9 and p_{10}) and on (2) the annotation suggestion rules.

The association between "*&AIPL;hasInput*" and "*&SANS;SBR_Operation_hasInput_DataObject*" is taken as an example. The SAP-KM performs the following steps:

(1) It queries the Class "*&SANS;E*" and acquires all Individuals that has the Object Property "*&SANS;SBR_Operation_hasInput_DataObject*" (e_9, e_{15}, and e_{21});

(2) For each acquired individual from Step 1 (e.g. e_9), it lists the objects that the Object Property "*&SANS;SBR_Operation_hasInput_DataObject*" points to (e_{11});

(3) For each acquired individual from Step 1 (e.g. e_9), it lists the objects the the Object Property "*&SANS;SR_isEquivalentTo*" or "*&SANS;SR_isSubsumedBy*" points to (e.g. p_9 and p_{10}, they are the domain semantics of e_9);

(4) For each acquired individual from Step 3 (e.g. p_9 and p_{10}), it lists the objects that the Object Property "*&SANS:hasMainConcept*" points to (e.g. "*&MSDL;Turning*" and "*&AIPL;BasesTurning*", they are the main concepts in

p_9 and p_{10} respectively). For each listed main concept, it lists the objects that the Object Property "*&AIPL;hasInput*" points to (e.g. "*&AIPL;TInputs*").

(5) It marks the acquired results from Step 4 (e.g. "*&AIPL;TInputs*") as the main concepts of a new domain semantics (e.g. p_{17}). The contents of this new domain semantics is created based on the traversal of the paths related to its main concept. Meanwhile, the range of the traversal is limited by the original domain semantics (e.g. p_{10}).

(6) Eventually, the new domain semantics from Step 5 (e.g. p_{17}) is suggested to annotate the acquired Individuals in Step 2 (e.g. e_{11}) through the Object Property "*&SANS:SR_isSubsumeBy*".

After the annotation suggestion, the similarity comparison between the two domain semantics of a common annotated element is performed. The SAP-KM queries all the individuals, which have both initial and inferred semantic annotations, and it generates all the possible comparison pairs between the initial one and the inferred one. Let's take the e_{11}= 'Aluminium Bars' as an example. The two comparison pairs that fulfil this condition (one is between p_1 and p_{17}, and the other one is between p_1 and p_{18}) are found. The semantic similarities between them are, for the current version, manually compared. To be more specific, the comparisons are described as follows:

- The first pair, $e_{11}sr_{\subset} p_1$ and $e_{11}sr_{\subset} p_{17}$ indicates that e_{11} inherits all the conditions described in p_1 and p_{17}. The domain semantics p_1 states that e_{11} is a kind of "*&AIPL;3mBar*", which has the length 3 meters and is made of the material "*&MSDL;aluminium*". The domain semantics p_{17} states that e_{11} is a kind of "&AIPL;Tinputs" that has a maximum length-limited to 1 meter. Because e_{11} is impossible to fulfil the condition "has the length 3 meters" and

the condition "has a maximum length-limited to 1 meter" at the same time. Therefore, p_1 has no common semantics with p_{17}. The comparison result is noted as $p_1 \, pr_\perp \, p_{17}$.

- The second pair, $e_{11} sr_\subset p_1$ and $e_{11} sr_\subset p_{18}$ indicates that e_{11} inherits all the conditions described in p_1 and p_{18}. The domain semantics p_{18} states that e_{11} is a kind of "&MSDL;Tinputs" that is made of the material either a kind of "&MSDL;Wood" or a kind of "&MSDL;Metal". Because the Individual "*&MSDL;aluminium*" is an instance of the Class "&MSDL;Non-Ferrous", which is a subclass of the Class "&MSDL;Metal". Therefore, the semantics of p_1 is less general than the semantics of p_{17}. The comparison result is noted as $p_1 \, pr_\subset \, p_{17}$.

The results of similarity comparison and the rules for detecting inconsistencies between semantic annotations (one example of the inconsistency detection rules is given in Figure 7 (b)) are used as inputs of the reasoning engine. Finally, the results of inconsistency detection are p_1 &SANS; isNotConsistentWith p_{17} and p_1 &SANS; isPosConsistentWith p_{18}.

The results of the inconsistency detection and the rules for identifying conflicts between annotated model elements (one example of the conflict identification rules is given in Figure 7 (c)) are used as inputs of the reasoning engine. The possible conflict between e_{11} and e_9 is identified by SAP-KM to draw attention of modellers for examining the correctness of the process model. Ideally, the conflict detection result is supposed to contain the reason why model elements are conflicting and how to solve it. For example, it can be "The input "*Aluminium Bars*" is 3 meters, which is beyond the range of the "*Bases Turning*" operation ($\leqslant$1meter). Therefore, one can either change the input or change the operation." However, because the similarity comparison

between two domain semantics is performed manually and the SAP-KM receives the comparison results without any explanations, in the current version of the SAP-KM, it is only able to provide a list of model elements that are in conflict with each other.

The process model and its semantic annotations are sent to Sage X3 (downstream system) to assist the parameterization. Giving the table of "process planning" in Sage X3 as an example, in which, the "process", "work centre", "preparation time" and "execution time" are four of its main elements. Taking the "*Bases Turning*" (e_9) operation as an example, as can be seen from Figure 9, the corresponding data that are needed by Sage X3 are already contained in its domain semantics p_{10}. Once the semantic annotations are created in the Sage X3's data model, the corresponding elements matching can be used to assist the stakeholder in filling the right data into the right fields in the "process planning" table.

After all, this case study shows how the formal semantic annotations are contributing in three following aspects: (1) Acquiring the initial semantics that the stakeholders who manipulate the upstream system wanted to express; (2) Verifying, semi-automatically, the semantic consistency between the contents in the received models and in the developing models; and (3) Guaranteeing the semantics correctness of the developing models for the stakeholders who manipulate the downstream system.

5. Conclusion

In an industry, the complex information flow in a network of systems leads to lots of discussions about the interoperability in different levels. From the semantic interoperability point of view, this research work provides contributions for all kind of model designers along the product lifecycle in the following aspects: (1) The semantic annotation meta-model that unambiguously describes the major components of a

semantic annotation and their interrelations; (2) The semantic blocks for the semantics description and substitution, which can be adapted by any research works that need the aggregation of semantics; (3) The formal definitions about the semantic annotations, which can be used as a basis to create semantic annotation schemas for the realization of semantic enrichment on any models; and (4) Three reasoning mechanisms that show the possible usability of semantic enrichments. Of course, besides these contributions, some limitations and perspectives also need to be noted.

In general, the discussion of the limitations should start from the three hypotheses of this research work, which are considered as three important factors that affect the semantic enrichment. Concerning the Hypothesis 1, the cost of the ontology creation and management needs lots of resources and might decrease the benefits of the related approach. However, in order to achieve the semantic interoperability, a common and shared knowledge base needs to be created. Concerning the Hypothesis 2, in order to use multiple ontologies in the process of semantic enrichment and reasoning, the interconnections among ontologies is an indispensable resource. Reasoning engines are not able to reason on concepts from different ontologies that have not relationships between each other. Concerning the Hypothesis 3, the method for the similarity comparison between two domain semantics is an important factor that might influence the result of the inconsistency detection. The more detailed explanations a comparison method can provide, the more precise results can be produced.

Furthermore, in the context of a PLM, besides the above-mentioned perspectives, four interesting directions can also be considered as future work: (1) *To enable the traceability of requirements*. With the assistance of semantics annotation, it is possible to trace the validation of each requirement in every stage of the PLC, from the initial design until the final deposit of; (2) *To make explicit the relations among*

TKRs. The semantic annotations can also be used to make explicit the hidden relations among all the disperse TKRs along the PLC; (3) *To address the versioning of models*. The issue about the versioning of models in a PLC is difficult to be avoided. Semantically enriching models gives the possibility to ensure that the updated model contents do not semantically conflict with existing ones; and (4) *To manage the semantic annotations*. Over time, more and more semantic annotations will be added on different models along the PLC. How to effectively manage those annotations will also become a thorny issue.

In a nutshell, despite some limitations, as discussed in this section, the proposed formalisation of semantic annotations is able to assist the semantic interoperability in a PLM context.

References

Ackoff, Russell L. 1989. "From Data to Wisdom." *Journal of Applied Systems Analysis* 16: 3–9.

Ameri, Farhad, and Deba Dutta. 2005. "Product Lifecycle Management: Closing the Knowledge Loops." *Computer-Aided Design and Applications* 2 (5): 577–590.

Ameri, Farhad, Christian McArthur, Bahram Asiabanpour, and Mohammad Hayasi. 2011. "A Web-Based Framework for Semantic Supplier Discovery for Discrete Part Manufacturing." In *Proceedings of the 39th North American Manufacturing Research Institution of SME (NAMRI/SME)*. Vol. 39.

Antoniou, Grigoris, and Frank van Harmelen. 2009. "Web Ontology Language: OWL." In *Handbook on Ontologies*, edited by Steffen Staab and Rudi Studer. Berlin, Heidelberg: Springer Berlin Heidelberg.

Attene, Marco, Francesco Robbiano, Michela Spagnuolo, and Bianca Falcidieno. 2009. "Characterization of 3D Shape Parts for Semantic Annotation." *Computer Aided Design* 41 (10): 756–763.

Bechhofer, Sean, Leslie Carr, Carole Goble, Simon Kampa, and Timothy Miles-Board. 2002. "The Semantics of Semantic Annotation." In *Proceedings of the 1st International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE)*, edited by Robert Meersman and Zahir Tari, 2519:1152–1167. Lecture Notes in Computer Science. Irvine, California: Springer Berlin Heidelberg.

Bergamaschi, Sonia, Domenico Beneventano, Alberto Corni, Entela Kazazi, Mirko Orsini, Laura Po, and Serena Sorrentino. 2011. "The Open Source Release of the MOMIS Data Integration System." In *Proceedings of the 19th Italian Symposium on Advanced Database Systems (SEBD)*, 175–186.

Boudjlida, Nacer, and Hervé Panetto. 2007. "Enterprise Semantic Modelling for Interoperability." In *2007 IEEE Conference on Emerging Technologies & Factory Automation (EFTA '07)*, 847–854.

Boudjlida, Nacer, and Hervé Panetto. 2008. "Annotation of Enterprise Models for Interoperability Purposes." In *Proceedings of the IEEE International Workshop on Advanced Information Systems for Enterprises*, 11–17.

Bugtai, N., and R.I.M. Young. 1998. "Information Models in an Integrated Fixture Decision Support Tool." *Journal of Materials Processing Technology* 76 (1-3): 29–35.

Curtis, Bill, Marc I. Kellner, and Jim Over. 1992. "Process Modeling." *Communications of the ACM* 35 (9). ACM: 75–90.

Doan, AnHai H., Jayant Madhavan, Robin Dhamankar, Pedro Domingos, and Alon Halevy. 2003. "Learning to Match Ontologies on the Semantic Web." *VLDB Journal* 12 (4): 303–319.

Etienne, Alain, Emmanuel Guyot, Dimitri Van Wijk, and Lionel Roucoules. 2011. "Specifications and Development of Interoperability Solution Dedicated to Multiple Expertise Collaboration in a Design Framework." *International Journal of Product Lifecycle Management* 5 (2): 272–294.

Euzenat, Jérôme. 2002. "Eight Questions about Semantic Web Annotations." *IEEE Intelligent Systems* 17 (2): 55–62.

Euzenat, Jérôme, and Pavel Shvaiko. 2007. Ontology Matching. Berlin, Heidelberg: Springer Berlin Heidelberg.

Feng, Shaw C. 1995. "A Dimension and Tolerance Data Model for Concurrent Design and Systems Integration." *Journal of Manufacturing Systems* 14 (6): 406–426.

Chen, David, Guy Doumeingts, and François Vernadat. 2008. "Architectures for Enterprise Integration and Interoperability: Past, Present and Future." *Computers in Industry* 59 (7): 647–659

Ghidini, Chiara, Marco Rospocher, and Luciano Serafini. 2008. *A Formalisation of BPMN in Description Logics*. Technical report TR 2008-06-004 (FBK-irst).

Gouyon, David. 2004. "Contrôle Par Le Produit Des Systèmes D'exécution de La Production: Apport Des Techniques de Synthèse,." PhD Diss., Université Henri Poincaré-Nancy I.

Gruber, Thomas R. 1993. "A Translation Approach to Portable Ontology Specifications." *Knowledge Acquisition* 5 (2): 199–220.

———. 2009. "Ontology." In *Encyclopedia of Database Systems*, edited by Ling Liu and M. Tamer Özsu, 1963–1965. Springer-Verlag.

Gunendran, A.G., and R.I.M. Young. 2010. "Methods for the Capture of Manufacture Best Practice in Product Lifecycle Management." *International Journal of Production Research* 48 (20): 5885–5904.

IEEE. 1991. *IEEE Standard Computer Dictionary. A Compilation of IEEE Standard Computer Glossaries. IEEE Std 610*. IEEE Press.

Leondes, Cornelius T., and Richard Henry Frymuth Jackson. 1992. *Manufacturing and Automation Systems: Techniques and Technologies*. London: Academic Press. Inc.

Li, Chun. 2012. "Ontology-Driven Semantic Annotations for Multiple Engineering Viewpoints in Computer Aided Design." PhD Diss., University of Bath.

Liao, Yongxin, Mario Lezoche, Eduardo Rocha Loures, Hervé Panetto, and Nacer Boudjlida. 2014a. "Formal Semantic Annotations for Models Interoperability in a PLM Environment." In *Proceedings of the 19th World Congress of the International Federation of Automatic Control (IFAC 2014)*, 2382-2393.

Liao, Yongxin, Mario Lezoche, Eduardo Rocha Loures, Hervé Panetto, and Nacer Boudjlida. 2014b. "A Semantic Annotation Framework to Assist the Knowledge Interoperability along a Product Life Cycle." *Advanced Materials Research* 945-949 (June): 424–429.

Liao, Yongxin, Mario Lezoche, Hervé Panetto, Nacer Boudjlida, and Eduardo Rocha Loures. 2015. "Semantic Annotation for Knowledge Explicitation in a Product Lifecycle Management Context: A Survey." *Computers in Industry* 71: 24–34.

Maedche, Alexander, and Steffen Staab. 2002. "Measuring Similarity between Ontologies." In *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, 2473:213–221. Springer.

Markman, Arthur B., and Dedre Gentner. 1996. "Commonalities and Differences in Similarity Comparisons." *Memory & Cognition* 24 (2): 235–249.

Nonaka, Ikujiro. 1994. "A Dynamic Theory of Organizational Knowledge Creation." *Organization Science* 5 (1): 14–37.

Noy, Natalya. F., and Deborah. L. McGuinness. 2001. *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford, USA.

OMG. 2013. "Semantics Of Business Vocabulary And Rules (SBVR)." http://www.omg.org/spec/SBVR/1.2/.

Pascot, Daniel, Faouzi Bouslama, and Sehl Mellouli. 2011. "Architecturing Large Integrated Complex Information Systems: An Application to Healthcare." *Knowledge and Information Systems* 27 (1): 115–140.

Polanyi, Michael. 1967. The Tacit Dimension. Garden City, N.Y.

Rong, Li, Cao Shunliang, Li Yuanyuan, Tan Hao, Zhu Yangyong, Zhong Yang, and Li Yixue. 2006. "A Measure of Semantic Similarity between Gene Ontology Terms Based on Semantic Pathway Covering." Progress in Natural Science 16 (7): 721–726.

Schwering, Angela, and Werner Kuhn. 2009. "A Hybrid Semantic Similarity Measure for Spatial Information Retrieval." Spatial Cognition & Computation 9 (1): 30–63.

Song, Fuqi. 2013. "Contribution of Ontology Alignment to Enterprise Interoperability." PhD Diss., University of Bordeaux 1.

Smith, Barry. 2003. "Ontology." In *Blackwell Guide to the Philosophy of Computing and Information*, edited by L. Floridi, 155–166. Wiley-Blackwell.

Sowa, John F. 1999. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole Publishing Co.

Stumme, Gerd, and Alexander Maedche. 2001. "Ontology Merging for Federated Ontologies on the Semantic Web." In *Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001)*, 413–418.

Uren, Victoria, Philipp Cimiano, José Iria, Siegfried Handschuh, Maria Vargas-Vera, Enrico Motta, and Fabio Ciravegna. 2006. "Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art." *Web Semantics: Science, Services and Agents on the World Wide Web* 4 (1): 14–28.

Uschold, Mike, and Michael Gruninger. 1996. "Ontologies: Principles, Methods and Applications." *Knowledge Engineering Review* 11 (2): 93–136.

Vernadat, François. 1996. *Enterprise Modelling and Integration*. London, UK: Chapman & Hall.

Vernadat, François. 2002. "UEML: Towards a Unified Enterprise Modelling Language." *International Journal of Production Research* 40 (17): 4309–4321.

Wang, X.H., T.N. Wong, and G. Wang. 2011. "Knowledge Representation for Multi-Agent Negotiations in Virtual Enterprises." *International Journal of Production Research* 49 (14): 4275–4297.

W3C. 2004. "RDF/XML Syntax Specification." http://www.w3.org/TR/REC-rdf-syntax/.

Yahia, Esma, Alexis Aubry, and Hervé Panetto. 2012. "Formal Measures for Semantic Interoperability Assessment in Cooperative Enterprise Information Systems." *Computers in Industry* 63 (5): 443–457.

Zdravković, Milan, Hervé Panetto, Miroslav Trajanović, and Alexis Aubry. 2013. "Explication and Semantic Querying of Enterprise Information Systems." *Knowledge and Information Systems* 40 (3): 679–724.

Table 1. The Ontological Representation of the Concept "Turning" and Concept "Bases Turning"

Ontologies	Concept	Relationships	Definitions from a Dictionary
MSDL Ontology	Turning	*subClassOf* Activity	**Turning**: the *activity* of shaping *something* on a lathe
		hasInput some Tinputs	
		hasOutput some Toutputs	**Turning**: the activity of *shaping* something on a lathe **Shaping**: the act of fabricating *something* in a particular shape
		isPerformedOn some Lathe	**Turning**: the activity of shaping something *on a lathe*
	Lathe	*subClassOf* MachineTool	**Lathe**: *the machine tool* for shaping metal or wood
	Tinputs	*hasMaterial* only (Metal or Wood)	**Turning**: the activity of shaping *something on a lathe* **Lathe**: the machine tool for shaping *metal or wood*
	Toutputs	*hasShape* some Shape	**Shaping**: the act of fabricating something *in a* particular *shape*
AIPL Ontology	BasesTurning	*isPerformedOn* TBI-450	The **Bases Turning** *is performed on* the lathe *TBI-450*, which has *an input* length limited (max 1 meter).
		hasInput some Tinputs	
	T01MaxLength	*type* LengthDescription	The Bases Turning is performed on the lathe TBI-450, which has **an input** *length limited (max 1 meter)*.
		meters 1	

| | Tinputs | *hasMaxLength* value T01MaxLength | |
| | TBI-450 | *type* Lathe | The Bases Turning is performed on **_the lathe TBI-450_**, which has an input length limited (max 1 meter). |

Figure 1. The Meta-model of the Semantic Annotation (Liao et al. 2014a)

Figure 2. The Application Scenario of the Case Study

Figure 3. A Part of the Process Model and the Selected Operation

Figure 4. A Semantic Annotation Example Represented in the Semantic Annotation Schema

Figure 5. The Domain Semantics of the "bar" from the Upstream System

Figure 6. The Domain Semantics (a) of the "Bases Turning" operation and the Structure Semantics (b) of three selected elements in the Process Model

Figure 7. The Example Rules for the Semantic Block Delimitation (a), the Inconsistency Detection (b), and the Conflict Identification (c)

Figure 8. The Domain Semantics of the Inferred Semantic Annotations

Figure 9. The SPARQL Query Results for the Downstream System