NISTIR 7274

Ontology Formalization of Product Semantics for Product Lifecycle Management

Lalit Patil
Debasish Dutta
Ram D. Sriram

National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

NISTIR 7274

Ontology Formalization of Product Semantics for Product Lifecycle Management

Lalit Patil
Debasish Dutta
Mechanical Engineering Department
University of Michigan
Ann Arbor, Michigan 48109

Ram D. Sriram
Manufacturing Systems Integration Division
National Institute of Standards and Technology
Gaithersburg, MD 20899-8260

November 2005

U.S. DEPARTMENT OF COMMERCE
Carlos M. Gutierrez, Secretary
TECHNOLOGY ADMINISTRATION
Michelle O'Neill, Acting Under Secretary of Commerce for Technology
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
William Jeffrey, Director

Ontology formalization of product semantics for Product Lifecycle Management

Lalit Patil and Debasish Dutta
Mechanical Engineering Department
University of Michigan
Ann Arbor, MI 48109
Email: [lpatil/dutta]@umich.edu

Ram Sriram
Group Leader, Design and Process Group
National Insitute of Standards and Technology
Manufacturing Systems Integration Division
Gaithersburg, MD 20899
Email: sriram@nist.gov

Abstract— **Product Lifecycle Management (PLM) is a concept that takes into account that the development of a product is influenced by knowledge from various stakeholders throughout its lifecycle. Computing environments in the PLM framework are expected to have several independent information resources. This requires a meaningful formal representation of product data semantics throughout the product's lifecycle. This paper presents an ontological approach to formalize product semantics into a Product Semantic Representation Language (PSRL). Building blocks to develop the explicit, extensible and comprehensive PSRL are described. The PSRL is open and based on standard W3C OWL constructs. The extensibility is demonstrated by considering an example product. The representation and the method of its development is expected to support several applications in the context of PLM. The use of OWL will enable the provision of the application software and information resources as Web services in the context of the Semantic Web.**

INTRODUCTION

Product Lifecycle Management (PLM) is a concept that takes into account that the development of a product is influenced by knowledge from various stakeholders throughout its lifecycle. Cross-functional, distributed teams use Computer Aided Design (CAD) and other tools along with available knowledge to develop the physical form, logic, specifications and all other information that defines a product. Additionally, multiple source vendors, contract manufacturers, distribution, and sales partners also add value to the product by using existing information and generating more knowledge [1].

Therefore, a broad spectrum of knowledge is associated with the product. Computing environments in the PLM framework are expected to have several independent information resources. These resources are typically of different types (databases, expert systems, application software, etc.) because they serve the needs of different domains. Thus, an essential feature of product information is the well-defined meaning (semantics) in a particular context. Further, growth in the use of the Internet has facilitated communication between the information resources. In this context, PLM views the extended value chain as one enterprise, not a set of silo-ed processes. Hence, every stakeholder in the product's lifecycle must have access to *the right information, in the right context, at the right time* [1].

As mentioned above, PLM needs the development of an architecture to support the integration of various information resources. The Automated Methods for Integrating Systems (AMIS) project [2] at the National Institute of Standards and Technology (NIST) identifies several technologies available to develop automated methods for integration of software systems. It identifies semantic conflicts as an important issue in solving an integration problem. Technologies and techniques that affect such an integration include technologies to formally capture and represent semantics of software systems, automated reasoning of tasks, and mechanisms to integrate semantic information from multiple sources.

Additionally, Semantic Web technologies have the potential to enable a service-oriented architecture. A service-oriented architecture is a collection of services that communicate with each other through some medium. A service is a self-contained system that provides a desired function. The Internet has a potential to provide robust connection among the services in service-oriented architectures. We expect that PLM tools will become service providers available through the Semantic Web.

The success of PLM in the context of the Internet requires reliable access to product information for stakeholders through a semantic web of the stakeholders' services whose knowledge is encoded in an unambiguous and computer-interpretable representation. Informal representations (unstructured, textual descriptions) are not effective because they can lead to ambiguities. Further, they cannot be used for automation because they are not entirely computer-interpretable. Therefore, formally defined representations are necessary for successful realization of the PLM architecture.

This paper outlines building blocks to formalize product semantics into a representation called Product Semantics Representation Language (PSRL). An earlier paper [3] described our approach to enable semantic interoperability based on the determination of semantic maps between ontological representations of product development systems. For this purpose, it discussed the development of shared semantics in the form of the PSRL using DAML+OIL [4]. This paper extracts the development of the PSRL from [3] and presents it as an ontological basis for semantic representations to potentially enable different applications in the context of PLM and the Semantic Web. The ontology development strategy and the example used in this paper are as mentioned in [3]. However, the PSRL is now encoded using Web Ontology Language (OWL) [5] which is an evolution over DAML+OIL. Corresponding modifications are described in this paper. Additionally, it also specifies details on the development and restriction of the ontology to the domain of OWL-based description logics.

The next section presents some requirements that the PSRL should satisfy. Later, the paper studies current commercial and research efforts relevant to product representation. Further sections explain development of the PSRL as an ontology and describe its capabilities through an example object. This is followed by a discussion on the potential applications for PLM that the PSRL can support. The paper concludes with a discussion of the features of the PSRL.

REQUIREMENTS FOR REPRESENTATION OF PRODUCT SEMANTICS

The following are requirements that the Product Semantic Representation Language (PSRL) must satisfy to form the basis of integration in PLM.

- **Application independence and dependence**
  The PSRL should be able to represent information that is common to all interacting applications[1]. In other words, the PSRL should be application independent.
  However, semantics of product information are relevant in a particular context (the application software). The number of software systems interacting in a PLM environment is not fixed. Further, there is an increasing number of new software that are being developed to support the activity. Therefore, the PSRL should support this dynamic evolution and capture the application-specific semantics of existing and new information resources.
- **Expressiveness**
  The PSRL must adequately express the meaning associated with a syntax. The primary requirement for automation is that the PSRL should provide a computer-interpretable representation of semantics associated with product data relevant throughout the product's lifecycle. It should not be restricted to represent information specific to only one application domain.
- **Unambiguity**
  Even if two systems use same phrases, they may differ in the meaning that each of them associates with that terminology. Alternately, two different terms may have similar meanings. A language that represents semantics successfully should be able to support automated reasoning to detect such unambiguities at least within itself.

[1]This paper uses *software systems* and *applications* to indicate information resources in the PLM environment

The next section analyzes efforts that are relevant to the development of such a product representation scheme.

RELEVANT WORK

ISO 10303 (also called STEP – STandard for the Exchange of Product model data) [6] is an international standard for product data representation. However, it only captures detailed geometry and related information. There are efforts to enhance ISO 10303 to enable the exchange of parametrization and constraint information associated with solid models [7]. However, ISO 10303 does not have the ability to model various information resources used throughout the product's lifecycle.

Several research efforts are focused on the creation of a product representation for collaborative product development. They are dedicated to answer questions at a higher level of product development. They try to consider not only the "what" but also the the "how" and the "why" of the design of an artifact. A detailed survey of such research efforts is provided in [8]. NIST's *Core Product Model* [9] and its extensions form a sound basis for its product information modeling framework [10] for PLM. However, this framework does not facilitate automated reasoning tasks. Further, it does not provide direct support to enable its implementation in the Semantic Web.

The intent of our work is to create a strong foundation for the formalization of product semantics. It does not focus on the development of a new product representation model, or new terminologies and semantics. Further, it does not create an exhaustive list of requirements necessary for every application in PLM.

In order to satisfy the reasoning requirements mentioned in the previous section, our research develops an ontology to formalize product semantics for PLM. Terminologies and their semantics are primarily based on concepts from NIST's *Core Product Model*. This ontology is called the Product Semantics Representation Language (PSRL).

PRODUCT SEMANTIC REPRESENTATION LANGUAGE (PSRL)

Ontologies have been found to facilitate representation for the purpose of integrating systems [11]. Typically, an ontology is defined as an explicit specification of a conceptualization [12]. An ontology language usually introduces concepts (entities), properties of concepts (attributes), relationships between concepts (associations), and additional constraints.

Literature documents several methods to build an ontology. A skeletal methodology to build ontologies is presented in [13]. It forms the basis of the ontology design for this work. This section describes the procedure of creating the PSRL by expressing engineering product knowledge into an ontology.

Identifying purpose and scope

The first phase in the development of an ontology is to identify its purpose and scope [13]. The primary purpose of the PSRL is to serve as an interlingua to enable integration in PLM.

As mentioned earlier, the scope of the PSRL is limited to terminologies and their semantics that are based on concepts from NIST's *Core Product Model*. CAD modeling concepts derived by a study of Unigraphics and Solidworks are used to demonstrate the extensibility of the PSRL to the CAD modeling domain.

After identifying the purpose and defining the scope of the ontology, the next phases in building the ontology involve capturing the ontology and coding it.

Capturing the ontology

This phase involves identification of key concepts and relationships in the domain of interest. It also involves identification of terms to refer to such concepts and relationships and providing textual definitions to them.

The *Core Product Model* is used to identify and classify a set of core concepts and relations required to define a product. It should be noted that the definitions mentioned in this paper assume certain knowledge of the domain of product development. Providing detailed and complete definitions is out of the scope of this paper. Later, certain application-specific definitions are developed as extensions to these core concepts and relations.

Core concepts in PSRL: The key concepts for the ontology can be briefly described as follows.

- Every concept is an *Object*. Thus, every *Assembly*, *Artifact*, *View* (*Front*, *Top*, etc.) is an *Object*.
- A *Specification* is information relevant to an *Artifact* based on customer needs and engineering requirements.
- An *Assembly* represents a collection of *Artifacts*.
- An *Artifact* represents a distinct entity in the design such that it has a *Form*, a *Function*, and a *Behavior*.
- The *Form* of the artifact is the physical design solution for the problem specified by a corresponding *Function*.
- The *Function* specifies what the *Artifact* is supposed to do.
- The *Behavior* represents how the *Artifact* implements the *Function*.
- Every *Form* is represented by its *Geometry* and *Material*.
- A *Feature* is a *Geometry* with other associated *Objects* that may lead to some knowledge about the *Form*'s *Function*.
- A *Constraint* is an *Object* that defines a shared property that must hold in all cases.

Core relationships in PSRL: The key relationships for the PSRL can be briefly described as follows:

- An *Object* may have other *Objects* associated with it.
- An *Object* may depend on (is a *child* of) one or more *Objects* for its existence.
- Correspondingly, an *Object* may be a *parent* of one or more *Objects*.

More detail on the textual descriptions of the terms defining these concepts and relationships can be found in [8]. These core concepts and relationships are used to explicitly encode the ontology to form a representation language (Product Semantic Representation Language - PSRL).

Coding the ontology

This section describes the lexicon for the intermediate language (PSRL). This is followed by describing axioms to provide semantics to the terminologies. Axioms also provide basic reasoning ability to the language. In order to keep the scope of the work feasible, not all the concepts and relations gathered in the previous phases are modeled completely. However, enough concepts and relations are included to provide an overview of a logical representation of product semantics.

Web Ontology Language (OWL): The PSRL is encoded in description logic [14]. Description logics (DL) are knowledge representation languages tailored for expressing knowledge about concepts and concept hierarchies. Most description logics are decidable subsets of first-order logic [15]. They are not as expressive as first-order logics. However, the decidability and tractability of reasoning services have made them a widely used tool for the representation of ontologies.

This research uses Protégé [16] as the ontology editor. Syntax for encoding the PSRL is based on the Web Ontology Language (OWL) [5], which is a recommended standard for the Semantic Web. OWL, which has an XML-based transfer syntax, provides a set of logical constructs to define ontologies. The underlying DL is obtained recursively by starting from a schema $S = (CN, RN, IN)$ of names of concepts names (CN), role names (RN) and individual names (IN). Concepts describe common properties of individuals. Roles are interpreted as binary relations between concepts. OWL has mathematical foundations in description logics. This allows the use of automatic reasoners to check the consistency of the ontology as it is being built.

Lexicon for the PSRL: Along with logical and non-logical symbols provided by OWL [5], the core of the PSRL consists of non-logical part of the lexicon (concepts and relations) that represents basic concepts in the PSRL ontology. In particular, these include the following:

- **Concepts:** *Object*, *Assembly*, *Artifact*, *Behavior*, *Constraint*, *Form*, *Function*, *Feature*, *Geometry*, *Material*, *Specification*
- **Relations:** *hasChild*, *hasParent*, *hasAttribute*

The intuitive semantics of these concepts and relations have been briefly described in the previous section. *Object* is the basic concept from which all other concepts are derived.

OWL syntax uses the term *owl:Class* to define a concept. For example, the term *Object* is a concept. This can be written as:

```
<owl:Class rdf:ID="Object"/>
```

The class hierarchy can be generated by using the construct *owl:subClassOf*. For example, *Artifact* is a *subClassOf Object*. This can be stated as:

```
<owl:Class rdf:ID="Artifact">
 <rdfs:subClassOf rdf:resource="#Object"/>
</owl:Class>
```

There are two types of properties in OWL that represent the domain of relations. Relations for which the range is an instance of an OWL class are classified as *owl:ObjectProperty*. Relations for which the range is a primitive data type such as integer, date and string are classified as *owl:DatatypeProperty* . For example, the relation *hasName* is defined as a datatype property with "string" as its range.

```
<owl:DatatypeProperty rdf:about="#hasName">
 <rdfs:range rdf:resource=
  "http://www.w3.org/2001/XMLSchema#string"/>
 <rdfs:domain rdf:resource="#Object"/>
</owl:DatatypeProperty>
```

ObjectProperties are defined similarly. Complete definitions for the ontology are specified by using restrictions and axioms in OWL.

Axioms for the core-PSRL: Axioms are used to capture basic properties of the ontology. They also provide semantics to the lexicon used in the PSRL. Precise definitions of some non-logical symbols and corresponding axioms are described in this section.

- ***Object***
  It is the most basic concept name in the PSRL. All concepts (and concept names) are subclasses of *Object*. The OWL property *owl:equivalentClass* is used for a complete definition of a concept (or a relation). In OWL, the concept *Object* is formally defined as follows:
  ```
  <owl:Class rdf:ID="Object">
   <owl:equivalentClass>
    <owl:Restriction>
     <owl:cardinality rdf:datatype=
      "http://www.w3.org/2001/XMLSchema#int">1<
     /owl:cardinality>
      <owl:onProperty>
       <owl:DatatypeProperty rdf:ID="hasName"/>
      </owl:onProperty>
    </owl:Restriction>
   </owl:equivalentClass>
  </owl:Class>
  ```
  Similarly, OWL constructs such as *owl:disjointWith, owl:unionOf, owl:inverseOf*, etc., are used to develop more definitions and axioms.
  However, for the purpose of this paper, we represent all other axioms using description logic syntax [17].
- ***hasChild***
  This relation represents the existence of a child object. It has the following axioms:
  Axiom 1: The *hasChild* relation is transitive.
  $$hasChild^{+} \sqsubseteq hasChild \tag{1}$$
  Axiom 2: The *hasChild* relation is an inverse of *hasParent*.
  $$hasChild \equiv hasParent^{-} \tag{2}$$
- ***hasParent***
  This represents the existence of a parent object. It has the following axioms:
  Axiom 3: The *hasParent* relation is transitive.
  $$hasParent^{+} \sqsubseteq hasParent \tag{3}$$
  Axiom 4: The *hasParent* relation is an inverse of *hasChild*.
  $$hasParent \equiv hasChild^{-} \tag{4}$$
- ***hasAttribute***
  The *hasAttribute* relation is used to interpret the role played by single *Objects* as attributes for describing another *Object*. Specific sub-properties such as *hasFunction*, *hasForm* and *hasConstraint* are created to define explicit relationships between various *Objects*. Each of these sub-properties may have more sub-properties, e.g., *hasDimensionalConstraint* is a sub-property of *hasConstraint*. Thus, there are no generic axioms associated with the *hasAttribute* relation. The *hasComponentArtifact* relation is a sub-property of *hasAttribute* relation. It is also a sub-property of the *hasChild* relation. It is used to represent the composition relationship between an *Assembly* and its *Artifacts*. Thus, an *Assembly* is described by its *Artifacts*, and the *Artifacts* are children of the *Assembly*.

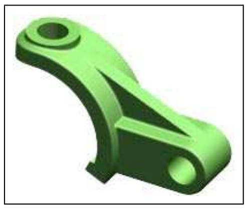
Fig. 1. Example product (bracket)

It should be noted that some of the above-mentioned axioms (For example, transitivity of relations) are represented in OWL as properties of relations (and concepts). However, they are only different representations of axioms in the language. Furthermore, the axioms mentioned above are not the only ones for the corresponding relations and concepts. There are more axioms and more representation is required for a complete definition of any concept or relation. For example, the range of *hasParent* is an *Object*. Similarly, its domain is an *Object*. Such details have not been provided in this paper.

Encoding the core constructs in the PSRL enables a generic formal representation of product information. More concepts need to be encoded within the PSRL. These provide more specific and instantiable constructs in the PSRL. The development of such new concepts utilizes the core constructs that have been mentioned in this section.

REPRESENTATION OF NEW CONCEPTS IN THE PSRL

One of the requirements for the PSRL is that it should be extensible to account for the dynamic evolution of a wide range of information resources in PLM.

This section demonstrates the ability to represent new concepts to represent product information in the PSRL. These concepts are developed onto the core-PSRL mentioned in the previous section. Only a few representative *Objects* are modeled here. Only geometric features are considered emphasizing the utility of semantic data representation even for geometric entities.

Example product

The bracket shown in Fig. 1 is an object from the National Design Repository [18]. It has several features characteristic of a prismatic part generated by milling process.

The taxonomy for some of the features in this example component is shown in Fig. 2. The complete detailed taxonomy for the example object is out of the scope of this paper.

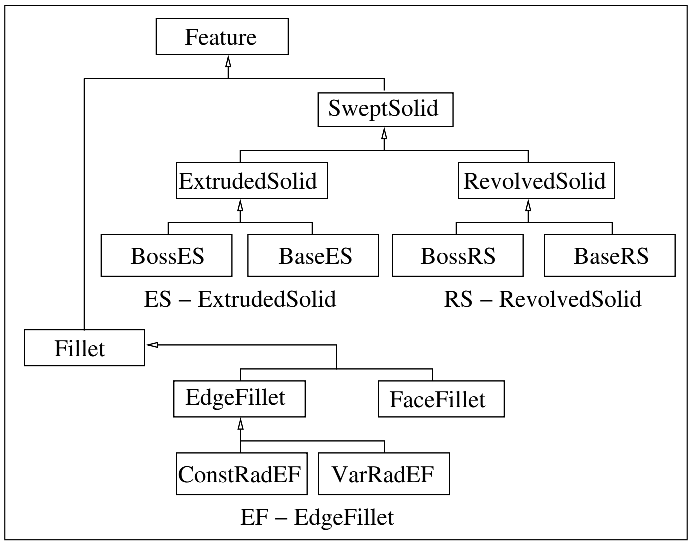

Fig. 2. Partial taxonomy of features of example product(Fig. 1)

The PSRL representation and involved axioms are stated as follows:

Axiom 5 Every *SweptSolid* is a *subclass* of a *Feature* that has a *SweptSketch* as at least one of its parents.

$$SweptSolid \sqsubseteq Feature \sqcap \exists hasParent.SweptSketch \quad (5)$$

Axiom 6 An *ExtrudedSolid* is defined as *SweptSolid* such that it has exactly one *Direction* and exactly one *depth* of extrusion. Further, the *SweptDirection* has the *Constraint* that the *value* of the *AngleBetweenDirectionAndSketchPlane* (measured in *Degrees*) is 0.

$$\begin{aligned} ExtrudedSolid \equiv\, & SweptSolid \\ & \sqcap =1 hasSweptDirection \sqcap =1 hasDepth \\ & \sqcap \forall hasSweptDirection.(Direction \\ & \sqcap hasConstraint.(\\ & AngleOfDirectionWithSketch \\ & \sqcap (Degrees \sqcap hasValue.\text{“0”}))) \end{aligned} \quad (6)$$

Axiom 7 A *BaseExtrudedSolid* is an *ExtrudedSolid* such that all its parents are a *SweptSketch*. Clearly, this means that the *BaseExtrudedSolid* does not have any *Feature* as its parent.

$$\begin{aligned} BaseExtrudedSolid \equiv\, & ExtrudedSolid \\ & \sqcap \forall hasParent.SweptSketch \end{aligned} \quad (7)$$

Axiom 8 Every *BossExtrudeSolid* is an *ExtrudedSolid* such that at least one of its parents is a *Feature*.

$$\begin{aligned} BossExtrudedSolid \equiv\, & ExtrudedSolid \\ & \sqcap \exists hasParent.Feature \end{aligned} \quad (8)$$

In order to distinguish between *BossExtrudedSolid* and *BaseExtrudedSolid* the reasoner should be able to differentiate between the concepts *SweptSketch* and *Feature*. Therefore, we explicitly state that an instance of the class *Feature* cannot simultaneously be an instance of the class *Sketch*. This is depicted in the next axiom.

Axiom 9 Every *Feature* is disjoint with a *Sketch*.

$$Feature \sqcap Sketch \equiv \bot \quad (9)$$

Axiom 10 *Feature* is an abstract concept that cannot be directly instantiated.

A concept cannot be directly instantiated if an instance of the class must also be an instance of its subclass. In OWL, this can be enforced by defining the parent class as a subclass of the union of all its subClasses. In this particular case, if *Hole*, *SweptSolid* and *Fillet* are the only subClasses of the class *Feature*, then abstractness can be enforced on *Feature* by stating

$$Feature \sqsubseteq Hole \sqcup SweptSolid \sqcup Fillet \quad (10)$$

The definitions and axioms mentioned in this section are derived from a study of the representations in SolidWorks and Unigraphics. Therefore, they represent a superset of the representations for the two application software. Similarly, axioms for the representation of *RevolvedSolid*, *Fillet* and *Hole* are stated in the PSRL.

As explained in this section, core concepts in the PSRL can be used along with the language constructs to form new concepts. Therefore, we can potentially model the full set of features that are encountered in a typical CAD system, such as SolidWorks and Unigraphics that have been studied in this work. New atomic concepts and relations can also be specified in the PSRL if the existing set of atomic entities is insufficient to model any particular feature. Therefore, for all practical purposes, all different types of features can be modeled within the PSRL.

Similarly, concepts from other application domains or information resources can be modeled to develop the application-specific elements of the PSRL. Such extensibility enables the PSRL to successfully represent dynamic changes in the information resources.

APPLICATIONS OF ONTOLOGICAL FORMALIZATION OF PRODUCT SEMANTICS

This paper focused on the development of ontologies for a formal representation of product semantics for PLM. The PSRL generates a consistent map of all knowledge that is available and has the ability to show all the information contained in different resources. It provides the means for a dynamic and flexible structuring of knowledge coupled with automated reasoning for effective browsing. In the context of PLM, this can potentially form the basis for several applications such as the following:

- **Standard for neutral representation**
  A unified view of the domain experts in product development can be presented in the form of the PSRL. The extensibility of the PSRL makes it ideal to form a superset of all interacting applications. It is also modifiable to account for new definitions, or changes in existing definitions. A formal logic base makes it easier to check the consistency of the evolving standard.

- **Semantic interoperability**
  Semantic translation involves determination of mappings between semantically equivalent terms between ontologies representing the interacting application domains. Using a logical base for the development of the product ontologies provides automated reasoning which can provide sound procedures to determine such semantic maps [3]. Additionally, using OWL to encode the ontology enables the usage of Semantic Web technologies being developed to support the determination of such mappings.
- **Semantic search**
  An explicit ontology can be used as a formal metadata for semantic searches on a repository of product models. Along with shape, all other information that is included in a product representation can be effectively utilized for better matches to a query. This will enable better re-use of previous designs and corresponding knowledge.

DISCUSSION

Description logics such as OWL may not be able to completely represent all information that is required for the complete representation of a product. First-order logic such as Knowledge Interchange Format (KIF) [19] is best suited for a complete representation, although at the expense of computational efficiency in reasoning [15]. Restriction to the domain of Description Logics may be impossible if a very low level (geometric entities such as points, lines, etc.) of abstraction is to be achieved. At this level, we encounter more types of restrictions (e.g., asymmetry, need to use variables) on concepts and relations that cannot be represented within the domain of DL. Efforts such as the proposed Semantic Web Rule Language [20] use additional rule layers on top of the description logics in order to enhance expressiveness. Such extra expressiveness, however, impacts on the characteristics of the languages.

We believe that OWL-based description logics is sufficient to represent information required for practical applications in PLM as mentioned in the previous section. We envision hybrid systems composed of a DL-based representation and existing well-defined standards. For example, a semi-automatic determination of semantic maps will provide a correct input to, and thus complement the use of well-developed translation standards (such as ISO 10303) for the physical translation of product model data.

CONCLUSION AND FUTURE WORK

Computing environments within a PLM framework are expected to have several independent information resources. This requires a meaningful representation of product data semantics across different application domains. This paper presented building blocks of the Product Semantics Representation Language (PSRL) for an intuitive and comprehensive formalization of product data semantics for PLM. Formal description logic is used to encode the PSRL. This provides the PSRL with the following features:

- **Application Independence and Dependence**. The PSRL is developed using a standards-based approach of analyzing application ontologies that need to exchange semantics. So, it is application-independent. The core PSRL can be updated for new application-specific features because it can incorporate new concepts and relations. This extensibility of the PSRL provides the potential to enable integration in a PLM environment where enterprises dynamically form temporary alliances.
- **Expressiveness** The PSRL is based on description logics and is therefore, fairly expressive. The formal language provides a computer-readable format and automated reasoning for applications relevant in the domain of PLM.

Additionally, the formal approach used to define the PSRL facilitates modifications to the representation while maintaining its consistency and preventing ambiguities.

Research is required to include additional and detailed concepts such as in [10] and more so that this formalization that will encompass all stakeholders in the product lifecycle. Further research is necessary to identify other components that will enable the usage of this representation language in the PLM framework.

We believe that such OWL-based ontological formalization of product semantics will enable better integration for PLM and will present the application software as services in the context of Semantic Web.

ACKNOWLEDGMENT

Patil and Dutta acknowledge the financial support (Grant #60NANB2D01017) from National Institute of Standards and Technology (http://www.nist.gov), and the University of Michigan's PLM Alliance (http://plm.engin.umich.edu).

DISCLAIMER

Certain commercial equipment, instruments, or materials are identified in this paper in order to facilitate understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, and the PLM Alliance nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

REFERENCES

[1] D. Dutta and J. P. Wolowicz, "An Introduction to Product Lifecycle Management (PLM)," in *Proceedings of the 12th ISPE International Conference on Concurrent Engineering: Research and Applications*, Fort Worth/Dallas, TX, USA, July 25-29 2005.

[2] E. J. Barkmeyer, A. B. Feeney, P. Denno, D. W. Flater, D. E. Libes, M. P. Steves, and E. K. Wallace, "Concepts for automating systems integration," National Institute of Standards and Technology (NIST), Gaithersburg, MD, Tech. Rep. NISTIR 6928, February 2003.

[3] L. Patil, D. Dutta, and R. Sriram, "Ontology-based exchange of product data semantics," *IEEE Transactions on Automation, Science and Engineering*, Accepted for publication.

[4] D. Connolly, F. van Harmelen, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, and L. A. Stein, *DAML+OIL Reference Description*, http://www.w3.org/TR/daml+oil-reference, 18 December 2001, w3C Note.

[5] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein, *OWL Web Ontology Language Reference*, http://www.w3.org/TR/owl-ref, 10 February 2004, w3C Recommendation.

[6] J. Owen, *STEP: An introduction*. Information Geometers Ltd., 1993.

[7] M. J. Pratt, "Extensions of the standard ISO 10303 (STEP) for the exchange of parametric and variational cad models," in *Globalization of manufacturing in the digital communications era of the 21st Century: Innovation, Agility, and the Virtual Enterprise: Proceedings of the tenth international IFIP WG5.2/5.3 PROLOMAT conference*, 1998.

[8] S. Szykman, R. Sriram, and W. Regli, "The role of knowledge in next-generation product development systems," *ASME Journal of Computing and Information Science in Engineering*, vol. 1, no. 3, pp. 3–11, March 2001.

[9] S. Szykman, S. Fenves, W. Keirouz, and S. Shooter, "A foundation for interoperability in next-generation product development systems," *Computer Aided Design*, vol. 33, no. 7, pp. 545–559, June 2001.

[10] S. Fenves, R. Sudarshan, R. Sriram, and F. Wang, "A product information modeling framework for product lifecycle management," in *First International Symposium on Product Lifecycle Management*. Bangalore, India: Indian Institute of Science, July 2003.

[11] M. Ciocoiu, D. Nau, and M. Gruninger, "Ontologies for integrating engineering applications," *ASME Journal of Computing and Information Science in Engineering*, vol. 1, no. 1, pp. 12–22, March 2001.

[12] T. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," in *Formal ontology in conceptual analysis and knowledge representation*, N. Guarino and R. Poli, Eds. Deventer, The Netherlands: Kluwer Academic Publishers, 1993.

[13] M. Uschold and M. Gruninger, "Ontologies: Principles, methods and application," The University of Edinburgh, UK, Tech. Rep. AIAI-TR-191, 1996.

[14] F. Baader and W. Nutt, "Basic description logics," in *The description logic handbook: Theory, implementation, and applications*, F. Baader, D. Calvanese, D. McGuiness, D. Nardi, and P. F. Patel-Schneider, Eds. Cambridge University Press, 2003, pp. 43–95.

[15] A. Borgida, "On the relative expressiveness of description logics and predicate logics," *Artificial Intelligence*, vol. 82, no. 1-2, pp. 353–367, 1996.

[16] *The Protégé Project*, Protégé, http://protege.stanford.edu, 2005.

[17] F. Baader, "Description logic terminology," in *The description logic handbook: Theory, implementation, and applications*, F. Baader, D. Calvanese, D. McGuiness, D. Nardi, and P. F. Patel-Schneider, Eds. Cambridge University Press, 2003, pp. 485–495.

[18] *National Design Repository*, National Institute of Standards and Technology, http://edge.mcs.drexel.edu/repository/frameset.html, 2005.

[19] M. Genesereth and R. Fikes, "Knowledge Interchange Format," Stanford University, Tech. Rep. 3.0 edition, 1992.

[20] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, *SWRL: A Semantic Web Rule Language combining OWL and RuleML*, http://www.w3.org/Submission/SWRL/, 21 May 2004, w3C Member Submission.