MASON: A Proposal For An Ontology Of Manufacturing Domain

Séverin Lemaignan[1,2], Ali Siadat[1], Jean-Yves Dantan[1], Anatoli Semenenko[2]
[1]LGIPM - École Nationale Supérieure d'Arts et Métiers
4 rue Augustin Fresnel, 57000 Metz, France
severin.lemaignan@gadz.org, {ali.siadat, jean-yves.dantan}@metz.ensam.fr
[2]RPK -Universität Karlsruhe (TH)
Kaiserstr.12, 76131 Karlsruhe, Germany
semenenko@rpk.uni-karlsruhe.de

Abstract

This paper presents a proposal for a manufacturing upper ontology, aimed to draft a common semantic net in manufacturing domain. Usefulness of ontologies for data formalization and sharing, especially in a manufacturing environment, are first discussed. Details are given about the Web Ontology Language (OWL) and its adequation for ontologies in the manufacturing systems is shown. A concrete proposal named MASON (MAnufacturing's Semantics ONtology) is presented and two applications of this ontology are exposed: automatic cost estimation and semantic-aware multiagent system for manufacturing.

Keywords: *Ontologies, Intelligent Manufacturing, OWL*

1. Introduction and Related Work

As result of the development of Product Life Cycle management, the need for a common way of describing manufacturing processes emerges. Versatile manufacturing compagnies are required to dynamically adjust their production capacities in coordination with their suppliers, their customers and mainly with other departments inside the company.

Manufacturing domain has been described by Martin [11] as the sum of *product*, *process* and *resource* concepts. Thus, dealing with manufacturing means dealing with these concepts i.e., having control over them as well as the bindings occuring between them.

These bindings are driven by three main elements:

- An Information System
- Rules
- A common dictionary

Information Systems have nowadays well-known and well-mastered architectures. However rules, and above all, shared corpus of definitions, lack of a widely recognized formalism.

Ontologies are semantic tools that address that kind of issue. They are formalized descriptions of concepts and relationships (both taxinomic and semantic) that exist between concepts. Ontologies are often designed to be expressed through standardized formal languages (like XML), thus ensuring shareabily and interoperability. Relationships between concepts may be viewed as rules inside the corpus.

This paper presents a draft of an upper ontology in manufacturing domain, i.e. a common ancestor for more domain-specific ontologies.

This work addresses topics similar to earlier work by the National Institute of Standards and Technology (NIST) [14]. They proposed a standard for interoperability in process management chain (Process Specification Language, PSL). Although sharing same purposes, our work relies on up-to-date formalisms (OWL semantics) and tries to show concrete application of such ontologies.

In the following section, we expose in details the main intents and purposes of ontologies for manufacturing systems (section 2.1). We then describe the OWL formal ontology language (section 2.2). The next section specifies more precisely our proposal and presents an ontology architecture for manufacturing (section 3), and we present two applications of this ontology being worked at the École Nationale Supérieure d'Arts et Métiers (ENSAM, section 4).

2. Ontologies for Manufacturing Domain

2.1. Intents of Ontologies

Gruber [4] defines an ontology as both:

Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)
0-7695-2589-X/06 $20.00 © 2006 IEEE

- the *conceptualization* of a domain, i.e. a choice on a way to describe a domain.
- the *specification* of this conceptualization, i.e. its formal description.

Both these aspects are relevant for manufacturing domain but they raise distinct issues. The *conceptualization*'s one is bound to a design choice. Concerning our domain, it's a choice of representation and organization for manufacturing concepts: for instance, a *part* could be viewed as an abstract concept, as subset of an assembly or a system. But a *part* is also a concrete physical element, made of a raw material which has been tooled. Other definitions of a *part* could be found, depending of the context of use.

Conceptualization is the task to describe in a systematic way concepts inside their contexts unambiguously. This is the first step toward formalization of a system.

This choice of a design may look very context-depend. It's however interesting to identify and group the set of context-independent concepts for a given domain (i.e. the concepts common to every particular ontology which could be written on this domain) to draw up an *upper ontology*. The purpose of such an upper ontology is to allow the specific ontologies to fluently integrate on the same common cognitive architecture, thus enabling the effective share of data model between heterogenous environments (different department inside the same company, different companies, etc.). Upper ontologies address one of the major concern of ontologies, known as the ontology alignment issue, or ontology integration issue. It occurs when two different ontologies about the same domain have to be merged (e.g. two company with their own ontologies which want to exchange datas). That issue has been summarized and main approaches for ontologies' mapping have been presented in [9].

The second aspect, the *specification* of the ontology, is linked to the concrete ability to write and store knowledge in an optimal way. It's a tradeoff between several and sometimes contractidory aspects like: reusablity and accessibility, interoperability (which implies standardization), modularity and extendibility, univocality. The specification should futhermore grant a consistent use of the information, i.e. a reproductible representation of knowledge.

These two aspects lead to several consequences for manufacturing domain. Ontologies could be considered on two different levels. First at **conceptualization level**, then at **operational level**.

At conceptualization level, ontologies are a mean to investigate, clean and eventually formalize a company knowledge. Building a domain ontology for a company requires indeed not only an in-depth knowledge of resources and processes involved in the company's activity, but also a precise sketch of relationships between these resources and concepts. Ontologies offer a framework to represent this knowledge.

Modern tools for ontologies rely futhermore on standardized paradigms and they keep interoperability in sight. This grants easier data exchange, but also continuation of stored knowledge, which could be a critical point for industries, knowing the quick obsolescense threat in computer science. Continuation is granted both by a long-term recognized accessibility to the data model and by extensibility of ontologies (see below). It's an important aspect for complete integration of ontology paradigm into the company's knowledge life cycle.

At operational level, ontologies' main role is to allow a fluent dataflow between heterogenous environments. Ontologies are indeed the outcome of metadata-based approaches, giving not only a metadata-based description of objects and concept, but also a complete *context* for these objects.

For instance, let say my ontology defines the concept of `aluminium` as a subconcept of `raw material`. Concepts of `drill` as subconcept of `tool` and `drilling` as a kind of `operation` are also defined. Properties `drill diameter` and `drill speed` are added to these concepts to refine them. The ontology finally binds these concept through the relationships *`canBeMachinedBy`* (between a `raw material` and an `operation`) and *`uses`* (between an `operation` and a `tool`).

Once these concepts and relationships are established, rules may be written (like "for aluminium and drilling diameter smaller than 5mm, the drill speed should be 3000 rpm"). Let imagine a part made of aluminium. The ontology provides a context to this assertion (*"part is made of aluminium"*) through which futher informations can be infered (like the speed for the drill machine).

A common ontology would let interpret this assertion in an uniform way all along the product life. Each component of the manufacturing chain is required to be able to interface with it, but once this is achieved (and standard languages for ontologies facilitate this), the assertion *"part is made of aluminium"* will be understood everywhere in the same way, with *a priori* the same consequences, no matter if the part move from one workshop, department, plant to another.

2.2. OWL Ontologies

Today's most advanced language for ontologies representation is the Web Ontology Language (OWL). It's a W3C recommendation [1] since 2004. While primary intended for web applications (especially, Semantic Web), OWL offers a complete framework for ontologies writing.

In OWL, the ontology is a set of definitions of classes, properties, and constraints on the way those classes and

properties can be employed. The OWL ontology may include the following elements [1]:

- **classes** i.e. concepts of the domain,
- **taxonomic relations** between classes,
- **datatype properties** i.e. attributes of classes,
- **objects properties** i.e. relations between classes (besides taxonomic ones),
- **individuals** i.e. instances (elements) of classes and properties,
- **restrictions** i.e. constraints on properties.

All class elements in OWL are instances of `owl:Class` metaclass (itself a subclass of `rdfs:Class`). Classes can be refined by elements that include:

- a `rdfs:subClassOf` property asserting that a class is a subclass of another class expression.
- and objects and datatype properties (see below).

A class expression can be a class name, enumeration, property restriction on a class, or a boolean combination of class expressions.

For example, the `drilling` concept would be defined in that way, inheriting from a predefined `Operation` concept:

```
<owl:Class rdf:ID="Drilling">
  <rdfs:subClassOf>
   <owl:Class rdf:about="#Operation"/>
  </rdfs:subClassOf>
</owl:Class>
```

The other part of an ontology definition in OWL is the definition of properties of classes. Properties can be either object properties (instances of `owl:ObjectProperty`) that relate objects to other objects, or datatype properties (instances of `owl:DatatypeProperty`) that relate objects to datatype values. Datatype values are defined by XML Schema definitions. Similarly as in the definition of classes, a property can be refined by several elements, mostly:

- a `rdfs:domain` asserting that the property only applies to instances of the specified class expression
- and a `rdfs:range` asserting that the values of the property are only instances of the specified class expression

For instance, the `drilling` concept could have two properties: a datatype property (*`drill speed`*) and an object property (*`uses`*) which is inherited from the `operation` concept and which binds the `operation` concept with the `tool` concept:

```
<owl:DatatypeProperty rdf:ID="drill_speed">
  <rdfs:domain rdf:resource="#Drilling"/>
  <rdfs:range rdf:resource="xsd:int"/>
</owl:DatatypeProperty>

<owl:ObjectProperty rdf:ID="uses">
  <rdfs:domain rdf:resource="#Operation"/>
  <rdfs:range rdf:resource="#Tool"/>
</owl:ObjectProperty>
```

Finally we can add property restrictions to classes. These restrictions are a special kind of class expressions. They refine the classes by restricting possible values of properties of these classes, but do not restrict properties themselves.

For example, the following restriction declares that plasma welding can not be applied to aluminium, i.e. the allowed values for the property *`canBeMachinedBy`* have to belong to the complement of the set { `Plasma_Welding` } in the set `Operation`.

```
<owl:Class rdf:about="#Aluminium">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:allValuesFrom>
    <owl:Class>
     <owl:complementOf>
    <owl:Class rdf:about="#Plasma_Welding"/>
     </owl:complementOf>
    </owl:Class>
   </owl:allValuesFrom>
   <owl:onProperty>
    <owl:ObjectProperty
              rdf:about="#canBeMachinedBy"/>
   </owl:onProperty>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

Notice that one of the main issue for an effective use of ontologies is how to represent rules. Taxinomies or semantic networks carry out static relationships between concepts whereas rules are intented to allow a dynamic evaluation of relationships. Rules could be seen as meta-predicate for the ontology. OWL restrictions are usually not sufficient to express with enough freedom rules, and specific rules languages like SWRL [6] would better fit the needs of writing rules.

3. A Practical Implementation

We propose a draft of an upper ontology for manufacturing domain. We named it **MASON**, for *MAnufacturing's Semantics ONtology*. Source files of this ontology are available online on [12].

As shown in figure 1, our proposal is built upon three head concepts : *entities*, *operations* and *resources*. A cor-

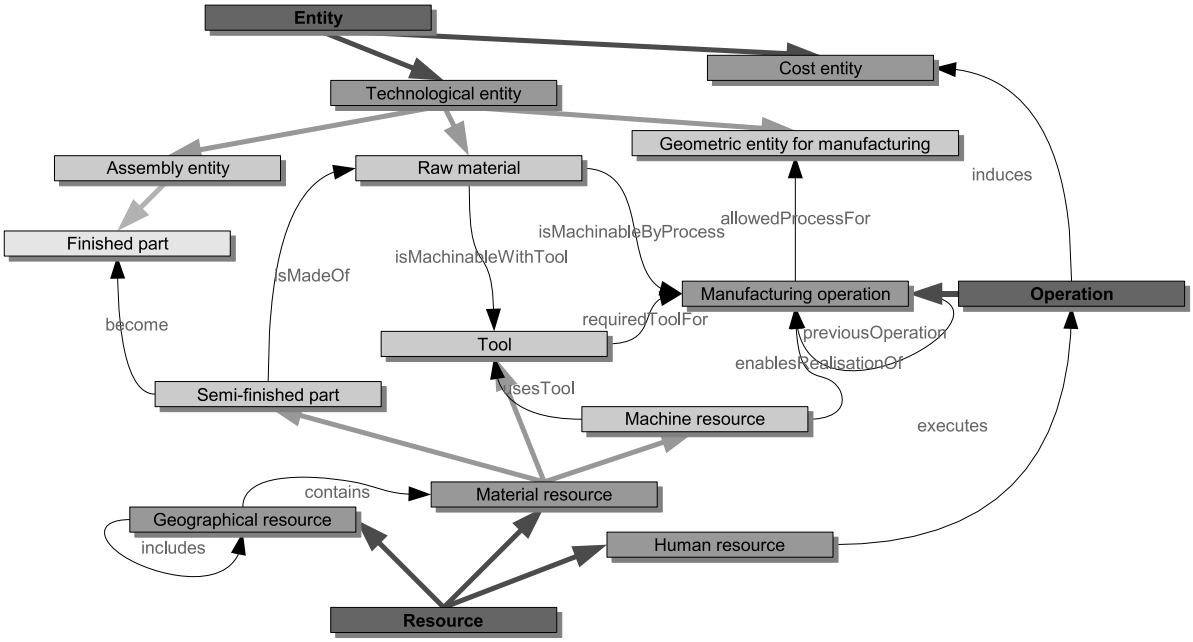

Figure 1. Overview of the ontology's main classes and object properties

respondance may be roughly established with Martin's decompositon of manufacturing in product, process and resource [11].

Figure 1 shows main subconcepts which inherite from head concepts as well as main relationships between these concepts. The figure is a small subset of the whole ontology which contains today up to 270 base concepts and 50 properties binding them.

3.1. Entities

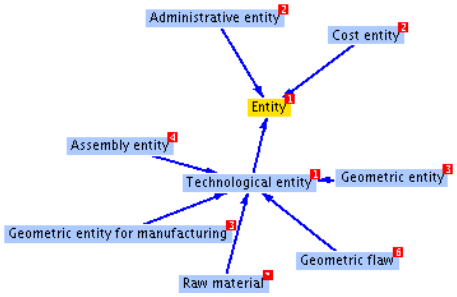

Figure 2. Entities' class hierarchy

Entities (figure 2) are all the common helper concepts. They aims to provide concepts to *specify* the product. It gives an abstract view on the product. The most important subconcepts amongst these entities are:

- *Geometric entities* and *Geometric entities for manufacturing* which represent respectively abstract (like `isTangentTo`) and concrete (like `Chamfer`) geometric features
- *Raw material*, actually viewed as abstract features of parts
- *Cost entities* which represent basic cost features as defined by H'Mida [5]

3.2. Operations

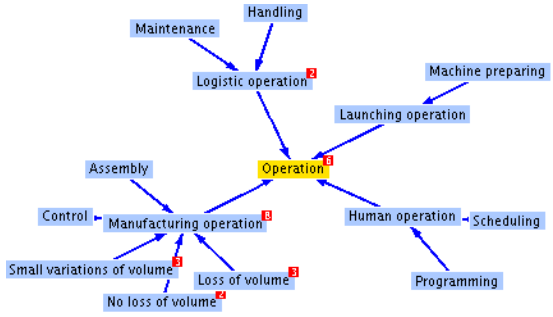

Figure 3. Operations' class hierarchy

Operations (figure 3) relate to process description. They cover all processes linked to manufacturing in a wide acceptation:

- *Manufacturing operations*, including machining operation as well as control or assembly. Machining operation are further classified according to their physical

Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)
0-7695-2589-X/06 $20.00 © 2006 IEEE

features (slow/brutal, hot/warm, with/without loss of volume...).

- but also *Logistic operations*,
- *Human operations*
- and *Launching operation*

3.3. Resources

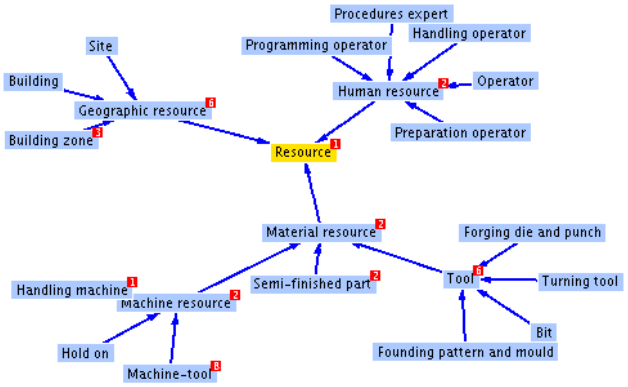

Figure 4. Resources' class hierarchy

Finally *resources* (figure 4) stand for the whole set of manufacturing linked resource, like:

- *Machine-tools*
- *Tools*,
- *Human resources*
- and *Geographic resources* (like plants, workshops...)

4. Applications

We present two applications which are recently our research topic. The first one aims to develop architectures and tools for automatic cost estimation. It relies on expert systems in conjunction with ontologies. The ontology we have presented in this paper has been actually drafted from this earlier research.

The second one try to fluently link a high level OWL ontology with a multiagent framework for manufacturing simulation.

4.1. Automatic Cost Estimation

As told by Kingsman [10], versatile manufacturing companies make mainly customised products, competing for each order with other supplier companies on the basis of price, technical expertise, delivery time and reliability in meeting due dates. They include engineer-to-order and make-to-order companies. Versatility is required in continually having to design and configure how to manufacture new or modified products, having continually to deal with varying production loads and having to deal with each customer order individually, even if it is for a very similar product to one sold earlier. A major problem is determining the cost of producing the order and then the price to be quoted.

The aim of this project is to provide a support for the manufacturing products cost estimation during design phase [7]. Such system needs a good and homogenous description of product, process plan and manufacturing resource to deliver an accurate cost. An expert system (Clips) based application was developed to estimate the cost of a mechanical part. All information and rules needed to expert system reasoning are collected from the MASON ontology. Using product information defined by its manufacturing concept, this application try to instantiate all activities needed to realize this product. In this way, the cost estimation could be evaluated by association a cost to each activity (cost entity concept, [5]).

4.2. Multiagent systems for Manufacturing

As already stated by Obitko [13], another domain of interest for ontologies are multiagent systems. As those architectures rely heavily on communication between agents, possibly in different locations and contexts, they require a robust language specification, i.e. a common vocabulary and syntax rules. Besides, cognitive agents [15] have to work with a world representation in order to interact with their environment. Both these aspects can be successfully addressed by using ontologies [8] and implementation of such models are already in use in multiagent frameworks like the Java Agent DEvelopement (JADE) framework [2].

The JADE framework use however a proprietary ontology model [3] which rely on Java inheritance schemes. While this approach is comfortable (especially in conjunction with ad-hoc Java classes which allow serialization and deserialization process to object level), it's hard to integrate the framework in a flawless flow of informations because of the non-trivial translation between standardized ontologies like OWL and JADE proprietary format. Moreover JADE only uses ontologies for vocabulary and syntax checking, while ontologies reveals usefull as cognitive model (knowledge base and reasonning capacities) as well.

We thus developped a mapper between OWL ontologies and JADE internal model (figure 5), to ensure at the same time functional JADE language operations (encoding/checking/decoding of messages), easy OWL integration and the use of domain ontology as a cognitive model for agents. This mapper is build around the Jena API (Ap-

plication Programming Interface) which provide extended access and manipulation tools for OWL ontologies, and is integrated in the JADE framework.

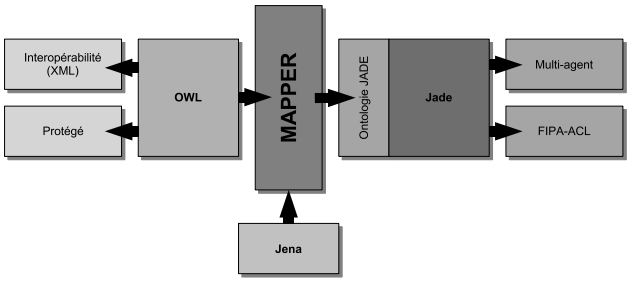

Figure 5. Mapper between OWL and Jade ontologies' model

This cognitive framework is currently under work at the ENSAM, with aim to set up a multiagent simulation for cost estimation in the manufacturing domain. The aim is to take a more global approach than in [7] with a logistic view of the manufacturing process. This application rely heavily on the manufacturing ontology we present in this paper, since interactions between the agents of the manufacturing chain derivate from the ontology.

5. Conclusion and future work

We've presented in this paper a proposal for a manufacturing upper ontology, aimed to draft a common semantic net in manufacturing domain. We first discussed the usefulness of ontologies for data formalization and share, especially in a open manufacturing environment. Details were then given about the Web Ontology Language (OWL) and we tried to show its adequation for ontologies in manufacturing systems. We presented a concrete approach and we exposed two applications (cost estimation and multiagent systems) which made use of the ontology. We finally discussed some points around ontology querying and interoperability between different ontologies.

We think ontologies have to play a central role in intelligent manufacturing: they enable fluent and consistent flows of data both inside and outside the compagny, they offer mature tools to deal with and XML definition language like OWL ease the integration with already implemented information systems as well as upcoming technologies like e-buisness or Web Services. However, ontologies are in no way a panacea: they don't avoid the necessary work to formalize compagny's knowledge, to determine and implement use cases, and first of all to examine in depth what kind of reasonning and inference are expected.

Within the scope of the european ICAMS project, the LGIPM laboratory of ENSAM will pursue investigations on the integration possibilities of technologies from artificial intelligence field in the manufacturing systems. Actual use cases should particulary be given attention in order to study concrete implementation opportunities.

References

[1] S. Bechhofer and al. Web ontology language (owl) reference. Technical report, W3C, 2005.

[2] F. Bellifemine and al. Jade: a fipa2000 compliant agent development environment. In *AGENTS '01*, pages 216–217, New York, NY, USA, 2001. ACM Press.

[3] G. Caire and D. Cabanillas. *Application-defined content languages and ontologies for the JADE framework*. JADE, 2004.

[4] T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. In *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers, 1993.

[5] F. H'Mida. *Contribution à l'estimation des coûts en production mécanique*. PhD thesis, ENSAM, 2002.

[6] I. Horrocks and al. Swrl: A semantic web rule language combining owl and ruleml. Technical report, W3C, 2004.

[7] X. Houin. Estimation des coûts et conceptual process planning. Master's thesis, ENSAM, 2005.

[8] M. N. Huhns and L. M. Stephens. *Multiagent systems: a modern approach to distributed artificial intelligence*, chapter Multiagent Systems and Societies of Agents, pages 79–120. MIT Press, Cambridge, MA, USA, 1999.

[9] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: The state of the art. *The Knowledge Engineering Review*, 18:1–31, 2003.

[10] d. S. A. Kingsman B.G. A knowledge-based decision support system for cost estimation and pricing decisions in versatile manufacturing companies. *International Journal of Production Economics*, 53:119–139, 1997.

[11] P. Martin and A. D'Acunto. Design of a production system: an application of integration product-process. *Int. J. Computer Integrated Manufacturing*, 16(7-8):509–516, 2003.

[12] MASON. Ontology public source code, 2005. `http://sourceforge.net/projects/mason-onto`.

[13] M. Obitko and V. Marík. Ontologies for multi-agent systems in manufacturing domain. In *DEXA '02: Proceedings of the 13th International Workshop on Database and Expert Systems Applications*, pages 597–602, Washington, DC, USA, 2002. IEEE Computer Society.

[14] C. Schlenoff, R. Ivester, and A. Knutilla. A robust process ontology for manufacturing systems integration. In *Proceedings of 2nd International Conference on Engineering Design and Automation*, 1998.

[15] M. Wooldridge. *Introduction to Multiagent Systems*. MIT Press, New York, NY, USA, 2001.