U.F.R. Sciences & Techniques Mathématiques,
Informatique, Automatique
Ecole Doctorale IAEM Lorraine,
Département de Formation Doctorale Automatique

Scuola di Dottorato di Ricerca
Sistemi Avanzati di Produzione
Politecnico di Bari
XXI Ciclo

Thèse en co-tutelle
Tesi in cotutela

présentée pour l'obtention du titre de
presentata per l'ottenimento del titolo di

Docteur de l'Université Henri Poincaré, Nancy I
en Automatique, Traitement du signal et Génie Informatique

Dottore di Ricerca in Sistemi Avanzati di Produzione
del Politecnico di Bari

Angela TURSI

APPROCHE BASÉE SUR LES ONTOLOGIES POUR L'INTEROPÉRABILITÉ CENTRÉE SUR LE PRODUIT DES SYSTÈMES D'ENTREPRISE DE PRODUCTION

APPROCCIO BASATO SULL' ONTOLOGIA PER L'INTEROPERABILITÀ CENTRATA SUL PRODOTTO DEI SISTEMI AVANZATI DI PRODUZIONE

ONTOLOGY-BASED APPROACH FOR PRODUCT-DRIVEN INTEROPERABILITY OF ENTERPRISE PRODUCTION SYSTEMS

Public discussion on November, 13th 2009

Members of the jury:

President:	Gino DINI	Professor, Università di Pisa, Italy
Reviewers:	Bruno VALLESPIR	Professor, Université Bordeaux 1, France
	Jean-Pierre BOUREY	Professor, Ecole Centrale de Lille, France
Thesis supervisors:	Hervé PANETTO	Professor, Université Henri Poincaré - Nancy I, France
	Michele DASSISTI	Associate Professor, Politecnico di Bari, Italy
Examiners:	Gérard MOREL	Professor, Université Henri Poincaré - Nancy I, France
	Paolo CHIABERT	Professor, Politecnico di Torino, Italy
	Antonio Domenico GRIECO	Associate Professor, Università di Lecce, Italy

---

Centre de Recherche en Automatique de Nancy, UMR 7039, Nancy-Université, CNRS, France
Dipartimento di Ingegneria Meccanica e Gestionale, Politecnico di Bari, Italia

To my family:
mom, dad,
Pierangelo
and nanTitina

Acknowledgements

I want to thank my tutors: Professor Hervé PANETTO and Professor Michele DASSISTI, who have seen in me the skills of researcher and encouraged me along these years.

Then, I would like give my special recognition to Professor Gérard MOREL, who has advised me during the first PhD thesis period in France.

I have also to thank Professors Bruno VALLESPIR and Jean-Pierre BOUREY, who accepted to be the official reviewers of the thesis, and all the other members of the jury, Professors Paolo CHIABERT, Gino DINI and Antonio Domenico GRIECO for examining my work.

Also I have to thank Professor Giovanni MUMMOLO, the coordinator of the Italian PhD School in my department, in Politecnico di Bari, who has taken care of my learning course during these years. I would like to express my sincere gratitude and appreciation to Michele De Nicolò, for the support and help provided for this research work. I would also give my special attention to Professors Alain Richard and Thierry DIVOUX, who welcomed me during my period in France, at CRAN facilities.

At second, there are some guys who contributed to this thesis, in diverse way:

- Jean Philippe, Salah and Zied, who have made the French experience of my PhD thesis more pleasant and constructive both on a cultural and on a professional layer;
- Michela, who has been a valid support during moments of discouragement and nervousness, but who is also a friend: with her I have shared a desk and a meal for months;
- Monique and Daniel, Olly and Cesare, who have been as parents for me during the periods away from home.

Last, but not least, I have to thank who loves me: my mother, my father, Pierangelo, my grandmother Titina, Alberto, Eliana and Pasquina: they

have believed in me more than me and their love has help me to face the difficulties met in these last periods.

Index

Introduction: Problem statement, research objectives and methodologies

1. Problem statement

Managing distributed and delocalised productions is one of the strongest issues to address in the present era of market globalisation. Information management is considered as a main requirement for products development in such networked enterprises.

Heterogeneous enterprise applications, either at business or at manufacturing levels, either inside a single enterprise or among networked enterprises, need to share information and to cooperate, in order to optimise its performance. This information may be stored, processed and communicated in different ways by different applications, according to the scopes for which these have been collected and they will be used.

A problem of misunderstanding can occur when information is exchanged between enterprise applications, due to different view points, for which they have been developed and, consequently, a risk of loss of information semantics may arise when exchanging between those heterogeneous systems.

This “Babel tower effect”, induced by the heterogeneity of applications, of their domains and their users, may cause information understanding problems, leading applications systems to fail at collecting information from different and heterogeneous sources to effectively ensure their local objective. This problem of managing heterogeneous information coming from different systems, in order to achieve a unique comprehension, falls within the umbrella of *interoperability* problems.

1.1. The Interoperability Problem

Interoperability can be defined as the ability of two or more systems or components to exchange information and to use the information that has been exchanged (IEEE, 1990).

Vernadat (Vernadat, 1996) defines interoperability as the *ability to communicate with pier systems and access the functionality of the pier systems*. IEC TC65/290/DC (IEC TC65/290/DC, 2002) has characterized the concept of interoperability as a certain degree of compatibility: *The application data, their semantic and application related functionality of each device is so defined that, should any device be replaced with a similar one of different manufacturer, all distributed applications involving the replaced device will continue to operate as before the replacement, but with possible different dynamic responses.* If an (manufacturing) application is considered as a combination of a set of processes, a set of resources, and a set of information structures that are shared and exchanged among the resources (ISO 15745), this definition means that the interoperability is considered as achieved only if the interaction between two systems can, at least, take place at the three levels: *data, resource* and *business process* with the semantics defined in a business context.

The ISO 16100 standard (ISO 16100, 2002) defines the manufacturing software interoperability as *the ability to share and exchange information using common syntax and semantics to meet an application-specific functional relationship through the use of a common interface*.

These definitions focus on the technical side of interoperability: more precisely, *Interoperability* is the ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful manner (Panetto et al., 2007).

Interoperability is not only a problem of software and IT technologies: it emerged from proprietary development or extensions, unavailability or oversupply of standards, and heterogeneous hardware and software

platforms, but when extended enterprises and networked organisations became to require businesses in order to work together for achieving further benefits, the Interoperability became to be a support for communication and transactions between different organisations.

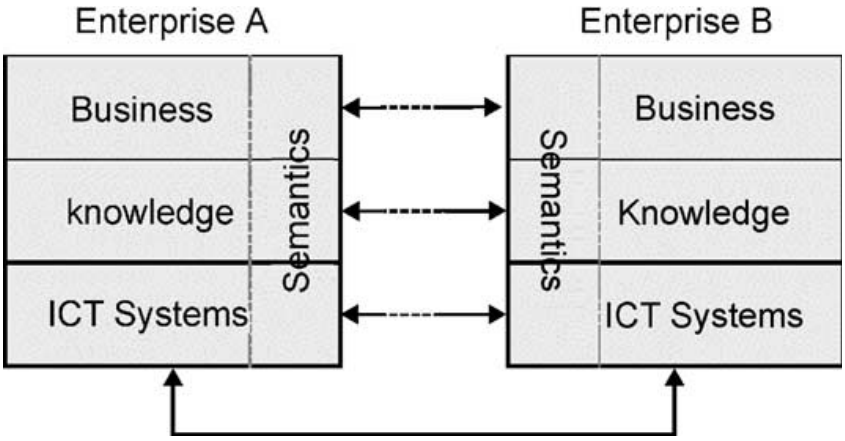


Figure 1 – Interoperability on all layers of an enterprise (Chen, 2003)

In order to achieve meaningful interoperation between enterprises, interoperability must be achieved on all layers of an enterprise. This includes the business environment and business processes on the business layer, the organisational roles, skills and competencies of employees and knowledge assets on the knowledge layer, and applications, data and communication components on the ICT layer. In additions, semantic descriptions can be used to get the necessary mutual understanding between enterprises that want to collaborate (Figure 1) (Chen, 2003).

Generally speaking, interoperability can be considered as that intrinsic characteristic of a generic entity (organization, system, process, model, …) allowing its interaction with other entities - to a different extent of simplicity - to cooperate for achieving a common goal within a definite interval of time, while pursuing its own specific goal.

There are three main research domains that address interoperability issues, identified by the expert group (INTEROP, 2003):

1) Enterprise modelling (EM) dealing with the representation of the inter-networked organisation to establish interoperability requirements;
2) Architecture & Platform (A&P) defining the implementation solution to achieve interoperability;
3) Ontologies (ONTO) addressing the semantics necessary to assure interoperability.

Enterprise modelling and Ontologies are contributing domains whose roles are to ensure that implemented applications correspond to user requirements, and the semantics used are understandable by two parties in interoperation. In other words, Enterprise modelling and Ontologies are to model the part of business and knowledge that have an impact on the interoperability of enterprise applications (Doumeingts and Chen, 2003).

According to the Enterprise modelling domain, in order to manage heterogeneous information, it is mandatory to develop models able to trace all relevant information related to the product lifecycle (design, manufacturing, sales, use and disposal). In fact, this information is quite often scattered within organizations: it is a matter of the materials adopted, of the applications used to manage technical data (e.g: Product Data Management systems (PDM)), of the applications that manage business information (e.g.: Enterprise Resource Planning (ERP)) and, finally, of the applications that manage manufacturing information (e.g.: Manufacturing Execution Systems (MES)).

Each enterprise application, in fact, uses an information repository, which refers to a Reference Information Model (RIM). A RIM specifies the structure and embeds the semantics of the information treated, in relation to the scope of the application to which it is devoted. This reference model may be either ad-hoc, thus developed specifically for and by any enterprise application or standard, when a consensus may take place among various key actors of the application domain. ISO 10303 and IEC 62264 are standards commonly accepted to allow information exchange

between ERP, CAD, PDM and MES applications, leading to an *application-driven interoperability* system (Figure 2).

Each enterprise application retrieves information from its repositories, according to the specific need during its operations and a negative effect may result in the case of exchange with different applications: the translation required can bring to significant loss of information, due to several causes (say, misinterpretation, misunderstanding,…) and this may have impact on its effectiveness.

Problems, then, can occur when there is a need to exchange information between enterprise applications. Firstly, a problem of misunderstanding, due to different view points for which each application has been developed: there is not an univocal way to express the same information. For example, in ERP application, the term *resource* refers indifferently to human and to machine resources, while in MES the term *equipment* refers to machine resources and the term *personnel* refers to human resources. Consequently, a risk of loss of information semantics may arise when effectively exchanging between heterogeneous systems (Tursi et al., 2007).

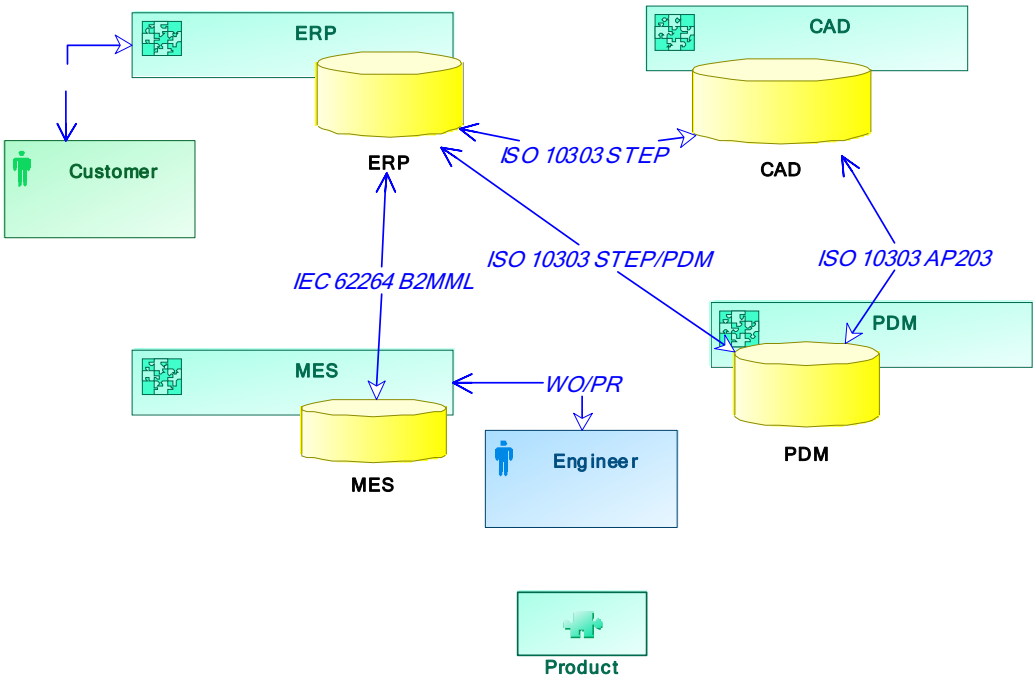


Figure 2 – Application-driven interoperability architecture

1.2. The Interoperability standards

Various standards have been developed and proposed for the area of enterprise systems interoperability. Generally, standards are developed in order to provide means and technology to integrate business management software among business partners.

In this domain, standards may be classified into two kinds: Portability Standards (which allow an executable program to run in different system contexts) and Interoperability Standards (which allow a program to communicate with another program without knowing its implementation or technology) (Bussler, 2003).

Looking to the literature, it is possible to identify three main categories of interoperability standards (Figure 3): standard covering the Product Development phase, standards covering the Product Production phase, standards covering the Product Use phase (Terzi, 2005).

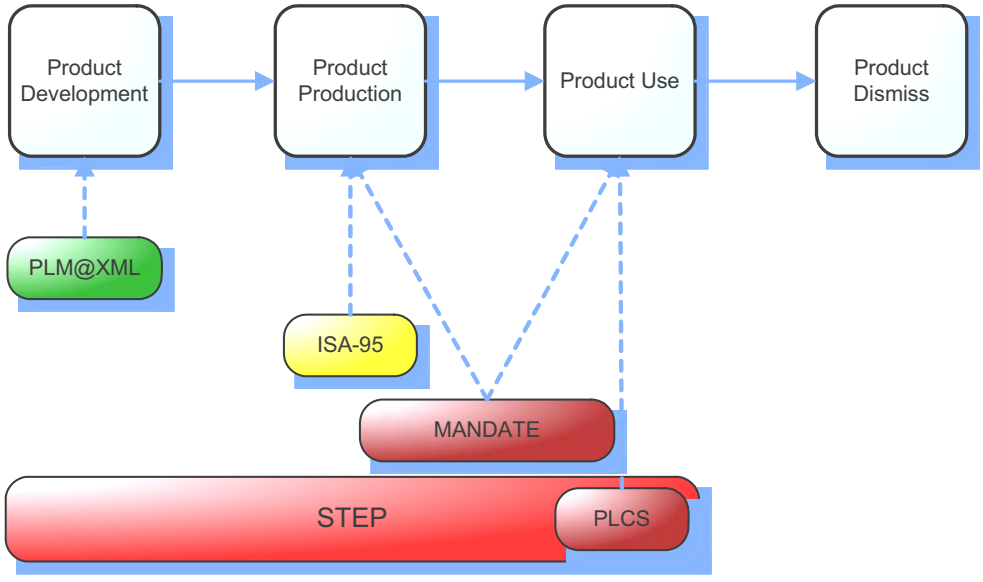


Figure 3 - Standards through product lifecycle (Terzi, 2005)

In the Product Development phase, standards interesting are ISO 10303 and PLM@XML (www.ugsplm.com). The most important and well-accepted standard in this phase is the STEP initiative (STandard for the Exchange of Product model data), which is an ISO (ISO 10303) standard

for the computer-interpretable representation and exchange of product definition data (ISO/TS 10303, 2004). PLM@XML derives partly from the STEP initiative, even if it is currently maintained by EDS (*Electronic Data Systems Corporation*) team in an open source way. PLM@XML provides a reference framework and a reference data format, based on XML, for the main sub-phases of Product Development, from Product Design to Plant Design and Process Design.

In the Product Production phase, two main “streams” of interoperability standards might be referred: (i) standards dealing with IT system supporting the Production Management, and (ii) standards dealing with ICT tools supporting the other activities of Operation Management.

Into the first classification, one of the most relevant standards, generally accepted by users and vendors, is ISO 62264 on Enterprise-Control-System Integration, developed with a joint effort spent by ISO and ISA organizations. ISO 62264 is a standard composed by four different parts designed for defining the interfaces between enterprise activities and control activities (IEC 62264, 2002).

Another interesting initiative is Mandate (MANufacturing DATa Exchange - ISO 15531). The Mandate scope is the representation of production information and resources information including capacity, monitoring, maintenance and control and the exchange and sharing of production information and resources information including storing, transferring, accessing and archiving. Mandate initiative aims to be compliant with STEP architecture, but on contrary of STEP, which takes a product-oriented view of manufacturing, Mandate is concerned with the processes of the organization which are used to produce the products.

Into the phase of Product Use, which deals with the day-by-day life of the product itself, interesting initiatives are (1) PLCS - Product Life-Cycle Support, standard based on ISO 10303 (STEP), which is an Application Protocol of STEP (AP 239) (www.plcs.org) and (2) PML - Physical Mark-up Language (McFarlane et al., 2003), developed by Auto-ID laboratories,

intended to be a general, standard means for describing the physical world for use in remote monitoring and control of the physical environment.

These standardisation initiatives share a common objective: trying to answer the information interoperability problem by formalising the knowledge related to products technical data along its lifecycle.

Between them, the most interesting ones are IEC 10303 and ISO 62264: they are universally well-accepted standards and they are able to model information regarding the product and its manufacturing, covering the product phases in which are more characterizing the questions of interoperability problem.

Both these standards will be studied in this thesis, because they try to solve the problem of managing heterogeneous information coming from different systems by formalising the knowledge related to Product Data Management at the business and the manufacturing levels of enterprises (B2M, Business to Manufacturing), in order to achieve the interoperability between systems. Actually, their models are used to allow the exchange of information between an ERP, PDM and MES within an application-driven system.

Nevertheless their approach is rather prescriptive, in the sense that it forces users to translate information from generic concepts to more practical and ad-hoc ones.

2. Research objectives

The integrated management of all the information regarding the product and its manufacturing is one of the more complex questions that characterize today's environment, defining a sort of "product-centric" or *product-driven* paradigm (Morel et al. 2003). In such a vision, the product itself becomes the medium of the data set, instantiating a kind of *active product* (Kärkkäinen et al., 2003), being able to interoperate in its environment, exchanging information (which is considered to be into the product itself) in real-time with different resources. The new

communication technologies, such as wireless technologies, RFID (Radio Frequency IDentification), etc., allow, from a technological point of view, to consider products as *active mobile objects*, embedding their own information structure, used and updated by the various actors during the product lifecycle.

New paradigms based on product oriented models also exist. The holonic-modelling paradigm proposes a structured method for designing product information, based on the synchronisation of product material flows and product information flows in a given manufacturing environment at modelling phase (Baîna, 2006). Holonic products integrating information from both the business and manufacturing point of view are able to contribute to interoperability issues (Baîna et al., 2009).

These paradigms share the consideration that, the product, along its life cycle, is the centred object from which all enterprise applications have a specific view that structures and embeds the semantics of information treated.

From the ICT point of view, a product-driven information management is no more than a “database” problem, which physically enables the previous business process modelling. Information about products are dispersed along a variety of information systems, which - until now - has been executed no more than “isolated islands” (e.g. PDM and ERP). From a structural point of view, the instantiation of a product-driven or product centric management approach means the product centric design and management of several elements:

1. An information infrastructure, which concerns with IT network establishment;
2. A resource infrastructure, which concerns with the design and the management of all physical elements involved along a product and production lifecycle (e.g. machines, plants, people, suppliers, warehouses…);

3. A product itself “infrastructure” where the same product has become a resource to be managed directly, traced into its same lifecycle (Terzi, 2005).

Starting form this, Morel and al (Morel, et al, 2003) postulate that it could be useful to define a common information model, to support information exchange between the many enterprise applications that interact between them, in order to solve the interoperability problem existing in the networked manufacturing enterprises.

Then, agreeing with this initial postulate, the main objective of our research activities is to define and possibly formalize the information model necessary to the product to become interoperable per se with the many applications involved in manufacturing enterprises and, as far as it embeds knowledge about itself, storing all its technical data, it will be able to act as a common source of understanding between enterprises applications. This results then to a so-called *product-centric interoperability* (Figure 4).

This model intends to specify an embedded *Product Ontology* that may be formed during the product life-cycle by the force of necessity of using it to communicate with the applications. The concept of embedding is related to the “pertinence” of the information structure: whenever related to the product information (technical, managerial, operational …) assumes a local (say embedded) meanly independently of the specific IT application requiring it (ERP, MES, PDM …).

In order to overcome questions pertinent to information exchange and its support, such as loss of information, problems of misunderstanding as well as redundant activities, it is necessary to define an *ontology-based* information model.

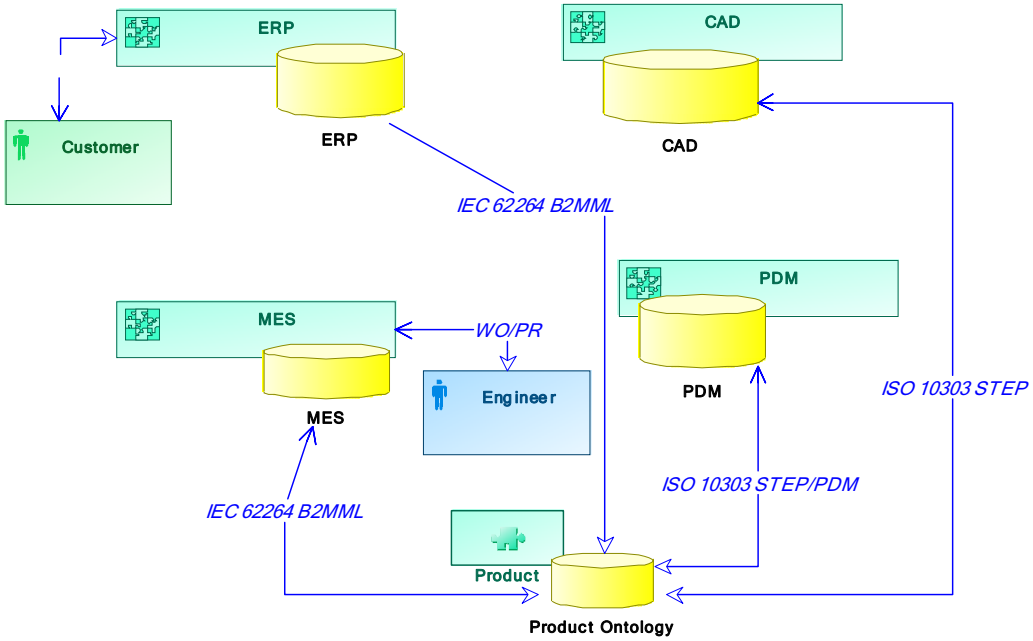


Figure 4 – Product-driven interoperability architecture

An ontology is an explicit specification of a shared conceptualization (Gruber, 1993), which allows the representation of domain's knowledge. It allows the formalization of the semantics of objects, and then it allows to formalize and to identify the modelling concepts and their dynamic behaviour, in order to express and to share this knowledge.

An *ontology* provides formal definitions of basic concepts in a domain and the relationships among them in a usually logic-based language (Gruninger and Lee, 2002). It is a specification of a conceptualization of an application domain of interest together with axioms that do constrain the possible interpretations for the defined concepts.

There have been, in many different sectors, some efforts examining the use of ontologies in supporting the semantic integration task (e.g. (Guo et al., 2003; Katranuschkov et al., 2003; Gehre et al., 2005; Lima et al., 2005; Patil et al., 2005; Terzi, 2005; Terzi et al.,2007). (Patil et al., 2005) is related to the NIST initiative on Product Engineering. (Terzi, 2005) has also been implemented in the frame of the PROMISE-PLM European project (www.promise.plm.com). An other interesting project of the European Community is PABADIS'PROMISE, called P2 project, which stipulates an innovative control and networking architecture across the

three levels of automation (ERP level, MES level, Field Control level) (www.pabadis-promise.org).

However, all these works have been either related to geometry data or have focused their study in the technology rather than to the conceptual view of product data exchange.

Aware of the efforts demonstrating the integration of models using ontologies, the matter of the approach, discussed in this thesis, is to formalise such *Product Ontology*, so it is feasible to embed information into the product and bringing and using them without further misunderstanding. Formalization means to provide a structured model of the information concepts and their semantics. Once the ontology is embedded, it represents a comprehensive structure of all the possible information pertinent to the product and genealogy.

As far as the product is processed, the related information can be "engraved" on it, with regards to the embedded ontology. This represents a significant advantage in terms of information retrieval and future use.

Through the formalization of this model, the product may be considered as interoperable per se as far as it embeds local information (knowledge about itself), as it stores all its technical data, provided that these are embedded on a common model, providing mappings from and to the enterprise applications (either inside a single enterprise or between networked enterprises) with respect to its life cycle.

In this direction, standards efforts, which represent a shared knowledge, developed by a group of expert, can be taken into account, in term of useful bases for the ontology of the domain and can represent a good starting point for the development of the ontological information model.

3. Research methodologies

The methodology applied to achieve the research objectives is constituted by two main steps. The first step is the study and analysis of existing standards related to product technical data modelling for the definition of

products information, allowing a non ambiguous model to represent knowledge and concepts, processable by the many enterprise applications adopted in manufacturing environment. The standard considered useful for this purpose are ISO10303 and IEC 62264.

The second step is to formalize this proposed “product-centric” information model as a Product Ontology, thus including domain rules, able to express and share product knowledge among systems (Tursi et al., 2009).

First of all, we focus mainly on mappings between both standards, in order to verify and to discover if they represent the same information, by instantiating the IEC 62264 and the corresponding STEP PDM modules on a particular example of product. This approach suffers of being not objective enough and being dependent on human interpretation, but it is the most pragmatic method for understanding the semantic of standards models information.

The first step of our methodology is based on a syntactical analysis whose aim is to compare the instances defined in both models and then based on semantics analysis, studying properties of the shared objects. Each relationship between different concepts will be studied and it will be possible to propose semantic correspondences between them (Baîna, 2006) in order to compare the contained information. For analyzing the semantics relationships between concepts, we choose FOL predicates: each predicate is formalizing mappings between STEP PDM concepts and IEC 62264 ones represented by a FOL axioms.

The final result of this work is then a contribution and prototype of a *Product Ontology* which, based on standard modelling concepts, intend to contribute to an interoperability solution between product views and enterprise applications that will manage them, formalising knowledge and skill around products.

4. Structure of the thesis

According to the presented research methodology, the thesis is structured as follows:

- Introduction about the problem statement, the research questions and methodologies.
- Chapter 1 describes the Interoperability problem and the existing initiatives and focuses on Interoperability problem in manufacturing system.
- Chapter 2 defines the Product Ontology: state of art, existing projects and tools and illustrates the state of art of Product Information Modelling domain: Interoperability Standards for Product Data are presented and ISO 10303 and IEC 62264 are described in particular.
- Chapter 3 illustrates the proposal of an ontological model for product-centric information systems interoperability.
- Chapter 4 deals with the validation of the proposed model.
- Finally, we conclude the thesis, summarizing the results and proposing some further researches for extending the current work.
- An annex is attached to the thesis in order to complete the relevant argument of First Order Logic.

5. References of the chapter

Baîna S. (2006). Interoperabilité Dirigée par les Modèles: Une Approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise. Thèse de doctorat de l'Universite Henri Poincare. December, Nancy, France (in french).

Baîna S., Panetto H., Morel G. (2009). New paradigms for a product oriented modelling: Case study for traceability, in: Special issue on Intelligent Products. Computers In Industry, 60/3, 172-183, April, Elsevier, ISSN: 0166-3615.

Baîna S., Morel, G. (2006). Product centric holons for synchronisation and interoperability in manufacturing environments, in: Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing. INCOM'06, May 17–19, 2006, St. Etienne, France.

Bussler C. (2003). B2B Integration: Concepts and Architecture, Springer.

Chen D. Doumeingts G., (2003). European initiatives to develop interoperability of enterprise applications—basic concepts, framework and roadmap, Annual Reviews in Control 27, 153–162.

Doumeingts G., Chen D. (2003). Interoperability of enterprise applications and software—An European IST Thematic Network Project: IDEAS e2003 eChallenges Conference, October 22–24, 2003, Bologna, Italy.

Gehre A., Katranuschkov P., Stankovski V., Scherer R.J. (2005). Towards semantic interoperability in virtual organisations, in: Proc. 22nd Conference of Information Technology in Construction, cibW78, July 18–21, Dresden, Germany.

Gruber T.R. (1993). 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.), Kluwer Academic Publishers.

Gruninger M., Lee J. (2002). Ontology — applications and design, in: Communications of the ACM 45/2, 39–41.

Guo M., Li S., Dong J., Fu X., Hu Y., Yin Q. (2003). Ontology-based product data integration, in: Proc. 17th International Conference on Advanced Information Networking and Applications;, 530–533, ISBN 0-7695-1906-7.

IEC 62264 Enterprise-control system integration, Part 1. Models and terminology, Part 2: Model object attributes. ISO/IEC, 2002, Geneva.

IEC TC65/290/DC. (2002, June 28). Device profile guideline, TC65: Industrial process measurement and control.

IEEE (1990), Standard Computer Dictionary- A Compilation of IEEE Standard Computer Glossaries. NY. 610-1990. ISBN: 155937079.

INTEROP. (2003, April 23). Interoperability research for networked enterprises applications and software, network of excellence. Proposal part B.

ISO 16100 (2002). *Manufacturing Software Capability Profiling for interoperability*, Part 1: Framework, TC 184/SC5/WG4, Geneva, Switzerland, ICS 25.040.01.

ISO/TS 10303 STEP modules related to Product Data Management. Industrial automation systems and integration — Product data representation and exchange, 2004, Geneva.

Kärkkäinen M., Ala-Risku T. and Främling K. (2003). The product centric approach: a solution to supply network information management problems?, *Computers in Industry* 52 (2), 147–159.

Katranuschkov P., Gehre A., Scherer R. (2003), An ontology framework to access IFC model data, in: ITcon; 8, 413–437.

Lima C., Silva C.F., Sousa P., Pimentao J.P. (2005), Interoperability among semantic resources in construction: is it feasible?, in: Proc.

22nd Conference of Information Technology in Construction, cibW78, July 18–21, Dresden, Germany.

McFarlane D., Sarma S., Chirn J.L., Wong C.Y., Ashton K. (2003). Auto ID System and Intelligent Manufacturing Control, Engineering Application of Artificial Intelligence, 16, 365-376.

Morel G., Panetto H., Zaremba M.B., Mayer F. (2003). Manufacturing Enterprise Control and Management System Engineering: paradigms and open issues, in: IFAC Annual Reviews in Control. 27/2, 199-209, Elsevier.

PABADIS'PROMISE, www.pabadis-promise.org.

Panetto H. (2007).Towards a Classification Framework for Interoperability of Enterprise Applications. International Journal of CIM, 20/8, 727-740, Taylor & Francis, December 2007, ISSN 0951-192X.

Patil L., Dutta D., Sriram R. (2005). Ontology-based exchange of product data semantics, in: IEEE Transactions of Automation Science and Engineering ; 2, 213–225.

PLCS, www.plcs.org

PLM XML, www.ugsplm.com

PROMISE PLM, www. promise.plm.com

Terzi S. (2005). Elements of Product Lifecycle Management: Definitions, Open Issues and Reference Models. PhD - University Henri Poincaré Nancy I and Politecnico di Milano, May.

Terzi S., Panetto H., Morel G., Garetti M. (2007). A holonic metamodel for product lifecycle management. International Journal of Product Lifecycle Management, 2/3, 253-289, Inderscience, December, ISBN 1743-5110.

Tursi A., Panetto H., Morel G., Dassisti M. (2009). Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. IFAC Annual Reviews in Control. 33/3, September, extended version from IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2–5, Monterrey, México, Elsevier, ISSN: 1367-5788 (in press).

Tursi A., Dassisti M., Panetto H. (2007). Products information interoperability in manufacturing systems. Ottavo Convegno AITeM (Associazione Italiana Tecnologia Meccanica) (AITEM'2007). September 10-11, Montecatini Terme, Italy.

Vernadat F. B. (1996). Enterprise modelling and integration: Principles *and applications*. London: Chapman & Hall.

Chapter 1: Interoperability in manufacturing systems

1. Introduction

One of the trends in the global market is the increasing collaboration among enterprises, during the entire product life cycle, in order to reduce time-to-market. Organisations have to flexibly react to changes in markets and trading partners. However, they have to cope with internal changes from both a technical (e.g. new software versions, new software and hardware technologies) and organisational points of view (e.g. merging, re-organisation).

Enterprise integration is that process which ensures the interaction between enterprise entities necessary to achieve domain objectives within value chain (EN/ISO 19439, 2003). Sharing product information must precede integration between enterprise entities. Interoperability of information among enterprises should be guaranteed in order that enterprises and their enterprise systems collaborate between them for integration of value chain. However, most enterprise systems and applications have different business experience and business domains. They store, process and communicate information in different ways because of the scope for which they are been developed. The heterogeneity of applications, of their domains and their users, may cause information understanding problems, leading applications systems to fail at collecting information from different and heterogeneous sources to effectively ensure a common objective. This is the major problem where interoperability is crucial.

This chapter proposes a relevant description on the predominant dimensions of interoperability domain, with a particular attention to enterprise interoperability, starting from a description of different levels and aspects of interoperability, mentioning the fundamental interoperability maturity models and concluding with a brief description of the existing

projects which deal with interoperability development. The problems of enterprise applications interoperability in manufacturing systems is explained in details and a description of the existing approaches to achieve the interoperability is given.

2. Interoperability: general issues

The interoperability problem is well known and recurring in many domains: database schema integration (Rahm and Bernstein, 2001), interoperability between modelling techniques (Domınguez and Zapata, 2000), interoperability in meta-modelling platforms (Kühn and Murzek, 2006), interoperability of ERP with other systems (Botta et al., 2005; Baîna, 2006), interoperability between heterogeneous information systems (Bermundez et al., 2007; Boulanger and Dubois, 1998) and in manufacturing (Xu and Newman, 2006). (Grangel et al., 2006) has worked on Model-driven based solutions for achieving interoperability in order to contribute solving the interoperability problems of Enterprise Software Applications (ESA) starting out from the enterprise models level and using an MDA-based approach. This method is called Model Driven Interoperability (MDI).

Interoperability is not only a problem concerning software and technologies. There are already various technologies to realise interoperability; some examples are TCP/IP, XML (W3C, 2004a), SOAP (W3C, 2003) , and web services are considered also potential technical solutions (Dogac et al., 2006).

Whitman and Panetto (Whitman and Panetto, 2006) refer to pragmatic interoperability as the willingness of all partners involved to participate in collaboration. This refers to the capacity of performing requested actions as well as the policies dictating them. Similarly, Mak and Ramaprasad (Mak and Ramaprasad, 2001) point out that, organisations must be able to contact each other using agreed protocols, share a common language, agree on goals and tasks, and have people assigned to complete these

tasks in order to achieve interoperability. That is, interoperability also concerns knowledge and business references that must be shared (Chen and Doumeingts, 2003).

Solving interoperability problems encompasses approaches to understand the technical, strategic and organisational behaviours from a holistic perspective (Wainwright and Waring, 2004). That is, organisations are complex and any effort has to handle all aspects in order to achieve interoperability between systems. Morel et al. (Morel et al., 2007) consider an enterprise as a SoS-like (System-of-Systems), because it is recursively composed of systems (its elements), and has a particular finality, related to its skill domain, resulting from the execution of enterprise applications.

Interoperability is a strategic issue; hence interoperability has to incorporate strategic planning for the entire system. This encompasses issues such as work practices, power and knowledge sharing which are all affected if enterprises are to be interoperable.

Interoperability between two organisations is a multifaceted problem since it concerns both technical and organisational issues, which are intertwined and complex to deal with.

Interoperability refers to the ability of two or more organisations to exchange and interpret all necessary information to collaborate. In order for organisations to be interoperable their strategies must cater for interoperation between business processes as well as ICT systems. The business view includes the strategic and operational aspects of the business. The ICT view includes the development and execution aspects. As interoperability problems can occur in any of these aspects, or in any combination of them, they have to be analysed with respect to all combinations.

2.1. Enterprise Interoperability

Enterprise integration is a domain of research developed since 1990s as the extension of Computer Integrated Manufacturing (CIM). Enterprise

integration research is mainly carried out within two distinct research communities: Enterprise Modelling and Information Technology (Panetto and Molina, 2008). In the context of Enterprise Modelling, the enterprise integration concerns the set of methods, models and tools that one can use to analyze, to design and to continually maintain an enterprise in an integrated state. An integrated state can be achieved ensuring constantly the interactions between enterprise entities necessary to achieve domain objectives. Enterprise interoperability refers to the ability of performing these interactions between enterprise systems (exchange of information and services). Then, it is a means to achieve integration (Chen and Vernadat, 2002; Panetto, 2007). ISO 14258 considers that interoperation between two (or more) entities can been achieved in three ways:

- Integrated: where there is a standard format for all constituent systems. Diverse enterprise models are interpreted in the standard format.
- Unified: where there is a common meta-level structure across constituent models, providing a means for establishing semantic equivalence.
- Federated: where models must be dynamically accommodated rather than having a predetermined meta-model: mapping between concepts could be done at an ontology level to formalise the interoperability semantics.

Integration is generally considered to go beyond mere interoperability to involve some degree of functional dependence. While interoperable systems can function independently, an integrated system loses significant functionality if the flow of services is interrupted. An integrated family of systems must, of necessity, be interoperable, but interoperable systems need not be integrated (Panetto, 2007). Integration also deals with organisational issues, in possibly a less formalised manner due to dealing with people, but integration is much more difficult to solve, while interoperability is more of a technical issue. Compatibility is something less than interoperability. It means that systems/units do not interfere with

each other's functioning. But it does not imply the ability to exchange services. Interoperable systems are by necessity compatible, but the converse is not necessarily true. To realize the power of networking through robust information exchange, one must go beyond compatibility.

In sum, interoperability lies in the middle of an "Integration Continuum" between compatibility and full integration. While compatibility is clearly a minimum requirement, the degree of interoperability/integration desired in a joint family of systems or units is driven by the underlying operational level of those systems.

Then, classifying interoperability problems may help in understanding the degree of development needed to solve, at least partially, these problems (Panetto, 2007).

2.2. Levels of interoperability

There are several possible levels of interoperability (Euzenat, 2001):

- encoding: being able to segment the representation in characters;
- lexical: being able to segment the representation in words (or symbols);
- syntactic: being able to structure the representation in structured sentences (or formulas or assertions);
- semantic: being able to construct the propositional meaning of the representation;
- semiotic: being able to construct the pragmatic meaning of the representation (or its meaning in context).

Each level cannot be achieved if the previous levels have not been completed. The encoding, lexical and syntactic levels are the most effective solutions, but not sufficient, to achieve a practical interoperability between computerized systems using existing technologies such as XML (eXtensible Mark-up Language) (W3C, 2004a), and its related applications (SOAP (Simple Object Access Protocol) (W3C, 2003), WSDL (Web Services Description Language) (W3C, 2004b), ebXML (Electronic

Business XML Initiative) (OASIS , 2002), to name a few). In that sense, standardisation initiatives (ISO 14528, 1999; IEC 62264, 2002; ISO EN DIS 19440, 2004) try to cope with this issue by defining generic constructs focusing on the domain concepts definitions. The semiotic level requires complex processing more related to artificial intelligence domain.

2.3. Aspects of Interoperability

Interoperability between two organisations is a multifaceted problem since it concerns both technical and organisational issues, which are intertwined and complex to deal with, but not only. According to the European Interoperability Framework (EIF, 2004), there are three aspects of interoperability:

1. *Organisational Interoperability*: This aspect of interoperability is concerned with defining business goals, modelling business processes and bringing about the collaboration of administrations that wish to exchange information and may have different internal structures and processes. Moreover, organisational interoperability aims at addressing the requirements of the user community by making services available, easily identifiable, accessible and user-oriented.
2. *Semantic Interoperability*: This aspect of interoperability is concerned with ensuring that the precise meaning of exchanged information is understandable by any other application that was not initially developed for this purpose. Semantic interoperability enables systems to combine received information with other information resources and to process it in a meaningful manner. Semantic interoperability is therefore a prerequisite for the front-end multilingual delivery of services to the user.
3. *Technical Interoperability*: This aspect of interoperability covers the technical issues of linking computer systems and services. It includes key aspects such as open interfaces, interconnection

services, data integration and middleware, data presentation and exchange, accessibility and security services.

2.4. Interoperability maturity models

The problems of enterprise interoperability can be defined according to various points of view and perspectives. Table 1 below shows the overlap and alignment between the various maturity models (Panetto, 2007). The main purpose of their framework is to provide an organized mechanism so that concepts, problems and knowledge on enterprise interoperability can be represented in a more structured way, in terms of diagrams, text and formal rules. They are not representation of operational processes, data, organizational structure, etc., but define the modelling constructs that are necessary to describe enterprise systems so that models achieved are consistent and easy integrated.

EIF					Organisational
			Semantic		
	Technical				
LISI	0 – Isolated	1 – Connected	2 – Functional Distributed	3 – Domain Integrated	4 – Enterprise Universal
OIM	0 – Independent	1 – Ad-hoc	2 – Collaborated	3 – Integrated	4 – Unified
LCIM	0 – System specific	1 – Documented	2 – Aligned static	3 – Aligned Dynamic	4 – Harmonised
NATO	1 – Unstructured data	2 – Structured data	3 - Seamless data sharing	4 – Seamless information sharing	

Table 1 – The maturity models (Panetto, 2007)

All these aspects correspond to modelling frameworks and enterprise architecture, with, as a common point, an implicit or explicit perspective of evolution according to a linear scale the more an application is interoperable with another and more it relates to a high level of abstraction of the models and their semantics (Panetto and Molina, 2008).

2.5. Existing projects

Two main initiatives relating to interoperability development exist: ATHENA Integrated Project (IP) (ATHENA, 2003) and INTEROP Network of Excellence (NoE) (INTEROP, 2003). The roadmap of these projects was defined by Interoperability Development of Enterprise Applications and Software (IDEAS) network, which was the first initiative carried out in Europe under the Fifth Framework Programme, to address enterprise and manufacturing interoperability.

Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications (ATHENA) is a programme that consists of a set of projects dealing with gaps-closing activities considered as priorities in IDEAS roadmaps and lead to prototypes, technical specifications, guidelines and best practices that form a common European repository of knowledge (ATHENA, 2003).

Interoperability Research for Networked Enterprises Applications and Software (INTEROP) aims at integrating expertise in relevant domains for sustainable structuration of European Research on Interoperability of Enterprise applications (INTEROP, 2003). More than 50 research entities (Universities, Institutes,…) and up to 150 researchers and 100 Doctorate students from 15 EU countries have worked within INTEROP NoE.

In other word, the gaps analysis and roadmaps resulted from IDEAS have led to the definition of R&D research projects to carry out by ATHENA. Dispersed and fragmented knowledge on interoperability and related research activities was integrated and restructured by INTEROP. The three initiatives form a coherent and complementary approach to enterprise interoperability.

This PhD thesis is developed in the frame of INTEROP NoE.

3. Interoperability in manufacturing systems

3.1. Enterprise Systems

In order to manage all information, enterprises are more and more equipped with *enterprise systems*. An enterprise system is a application, dedicated to specific tasks from resources planning to shop floor control.

The term 'application' is often misunderstood as a synonym of software. According to ISO 15745, an (manufacturing) application can be modelled as a combination of a set of processes, a set of resources, and a set of information structures that are shared and exchanged among the resources. In ENV 12204, three types of resource have been considered: machining, computing and human types.

In manufacturing enterprise (Figure 5), it is possible to identify a hierarchy, composed of three main levels (Baîna et al., 2009):

- The higher level represents management system level, responsible of the management of processes that handle all different informational aspects related to the enterprise (e.g. ERP systems)
- The lower level performs the processes that manage decision flows (e.g. Workflow systems) ad production flows (e.g. MES)
- The medium level is the process control level: contain all processes that perform physical transformations on the produced goods and services.

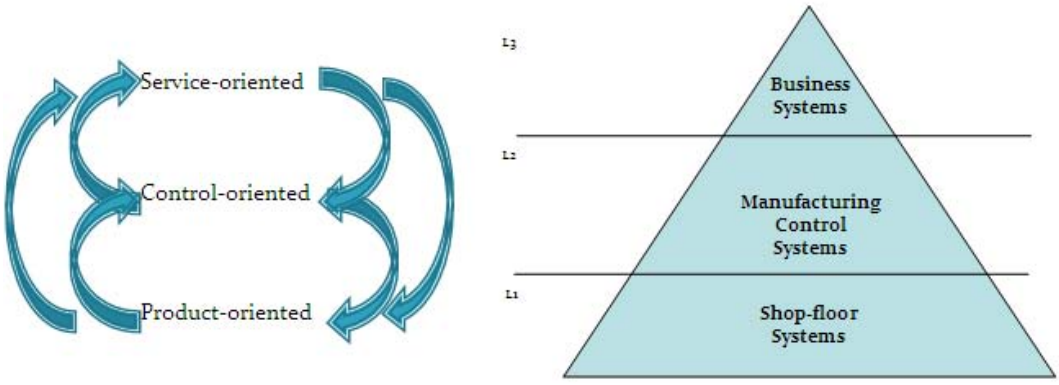


Figure 5 - The manufacturing enterprise model (Baîna et al., 2009)

(Baîna et al., 2009) consider that the manufacturing enterprise is composed of two separated worlds rather than a simple hierarchy of levels: on one hand, a world in which the product is mainly seen as a physical object. This world is called the *manufacturing world*: it handles systems that are tightly related to the shop-floor level. On the other hand, a world where the product is seen as a service released in the market. This world is called the *business world* (Figure 6).

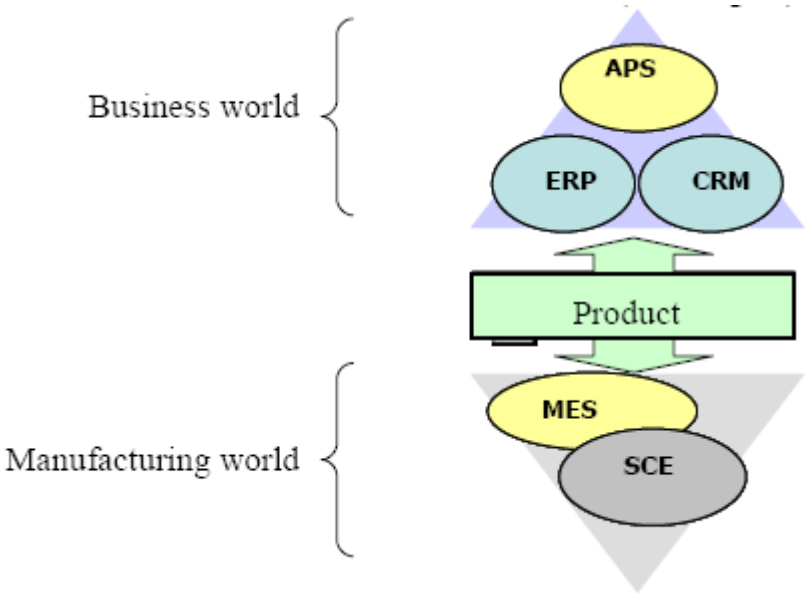


Figure 6 - Product centric approach (Baîna et al., 2009)

The enterprise systems are typically dedicated to a specific level inside an enterprise, working for different people with different skills.

In other words, in an enterprise there are Business Systems, that are more service oriented; Manufacturing Control Systems, that are oriented to control the production; and finally Shop-floor systems, closed to the real product to be produced.

These levels are not islands. Level 2 enterprise systems must communicate with level one systems for monitoring the manufacturing processes and resources. In the same way, service oriented enterprise systems may also need to communicate with either the level 2 or the level 1. All these communication channels are obviously difficult to put in place when applications are heterogeneous and not initially built to do so.

This is a major drawback in manufacturing systems, where interoperability is crucial.

3.2. Standard-based approach to enterprise interoperability

Because companies have been using heterogeneous information systems, they primarily have used standard-based approaches for large scale information sharing.

Various existing standards can be classified into two categories:

1. supporting infrastructures, architectures, and languages (e.g. CORBA, FIPA, KQML, and NIIIP, etc.);
2. standards for information exchange and sharing (e.g. STEP, KIF, and XML, etc.).

However, the standard-based approaches have raised several issues and problems, such as (Oh and Yee, 2008):

1. they force whole trading partners to follow a single unified standard, ignoring the heterogeneous nature inherent in business partners' environments;
2. it is significantly inefficient and difficult to fit, customize, and integrate complex industrial standards. Many enterprise applications schemas mismatches, such as terminology, structure, data organization, and data granularity, even though they share the same semantics at higher abstract level;
3. because these standards allow flexibility in terms of message contents and their processes composed, a significant effort is required to implement precisely business transactions (Kotinurmi, 2005), even though the partners agreed to use them;
4. an excessive lead-time is required to accept new partners and connect them to existing partners;
5. the traditional standardization process cannot manage semantics of messages effectively.

Most of the standard-based solutions only provide commonly agreed sets of labels, entity definitions and relationships for interchanging heterogeneous information or for defining project models. But they usually do not support broad ranges of explicit definitions for the terminologies and concepts used in their schemas.

As these standard schemas lack rich, formal and explicit semantic descriptions, they cannot ensure the consistent interpretation and understanding of application semantics across disciplines. Simply sharing the common labels and standard data structures is not sufficient to achieve *semantic interoperability*.

To address the issues and problems of the standard-based approaches, and to achieve semantic interoperability, different technologies have been introduced. Besides the interoperability standardization approach, ontology engineering is recognized as another key technology to deal with the semantic interoperability problems.

4. Ontology-based approach for semantic enterprise interoperability

Semantics can be broadly defined as the meaning associated with a terminology in a particular context (Patil et al., 2005). Semantic interoperability is the ability of enabling heterogeneous multi-disciplinary enterprise applications to understand and utilize semantics of enterprise systems and meanings of model data, and to map between commonly agreed concepts to establish a semantically compatible information interchange and sharing environment.

Ontologies are often considered to be a most powerful means to solve the problem of efficient storing and retrieving knowledge, because they are constructed to specify the conceptual model of an information and knowledge domain explicitly. For this reason, they can be used in supporting information and knowledge exchange between different

organizations and then they are very useful in solving semantic interoperability problem.

Ontologies specify the semantics of terminology systems in a well-defined and unambiguous manner (Guarino, 1998), by formally and explicitly representing shared understanding about domain concepts and relationships between concepts. In an ontology-based approach, intended meanings of terminologies and logical properties of relations are specified through ontological definitions and axioms in a formal language, such as OWL (Web Ontology Language) (W3C, 2005) or RDF (Resource Description Framework) Schema (www.w3.org/TR/2000/CR-rdf-schema-20000327).

Since information and knowledge domains are diverse and even evolve, different people and organizations tend to adopt different ontologies. As shown by Madnick (Madnick, 1995), we cannot hope that one universally accepted unchanging ontology, even for a small domain, would ever be created. Therefore, in order to achieve interoperability of information and knowledge among heterogeneous organizations, different ontologies must be reconciled.

The usage of appropriate structures and unified terms, defined by an ontology, is promising for the efficient exchange, re-use and further elaboration of innovations.

There exists a variety of alternative architectures to reconcile multiple ontologies for interoperability of heterogeneous organization networks. Hameed et al. (Hameed et al., 2003) suggests three architectures as shown in Figure 7.

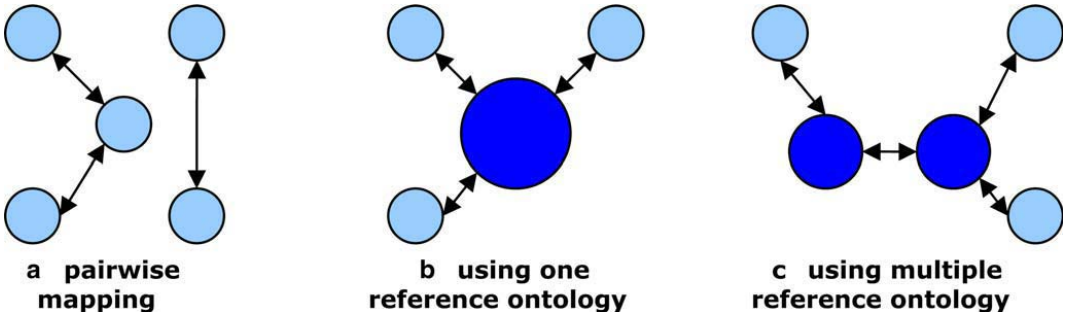


Figure 7 - Various ontology reconciliation architectures (Hameed et al., 2003)

Figure 7(a) shows the mapping which is used in the case when there is no need to reconcile all ontologies, but rather just interrelate individual ontologies as needed. While this approach gives great flexibility and simplicity, in the worst case, there will be many sets of mappings (n^2), if n individual ontologies are required to be mapped to the rest of others in a bidirectional way. Figure 7(b) depicts the mapping based on a single common ontology, when there is an attempt to reconcile individual ontologies in a principled, top-down fashion. This approach supposes that a common and standard conceptualization is identified and developed, whatever the cost of the development might be. It also loses some flexibility in the local management level, because all individual ontologies must follow a centralized standard. Figure 7(c) illustrates the case that uses multiple reference ontologies, forming clusters of interrelated ontologies. Each individual ontologies are mapped to the reference ontology for its cluster, and the reference ontologies are mapped to each other. This hybrid approach is to combine the advantages of Figure 7(a) and b—a reduced number of mapping using principled conceptualization, and yet also there is flexibility to extend interoperability through adding different clusters.

5. Conclusions

Nowadays, semantic interoperability constitutes an important approach to deal with heterogeneity within large and dynamic enterprises. Currently, the existing solutions are mainly based on the use of some standards and

also middleware in order to overcome the interoperability problem. These solutions generally fail as they do not scale to large number of applications and also fail as they do not provide more flexibility and agility. Here, solutions based on semantic web services are promising and they are being actively researched. These technologies can offer answers to some key challenges such as semantic mediation and interoperation. Nowadays, the ontologies are considered to be a most powerful means to solve the problem of semantic interoperability: an ontology defining concepts and properties of enterprise systems domain can be able to achieve communication and to share information between enterprise applications.

6. References of the chapter

ATHENA (2003, April). Advanced technologies for interoperability of heterogeneous enterprise networks and their applications. FP6-2002-IST-1, Integrated Project Proposal.

Baîna S. (2006). Interoperabilité Dirigée par les Modèles: Une Approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise. Thèse de doctorat de l'Universitè Henri Poincarè. December 2006, Nancy I, France (in french).

Baîna S., Panetto H., Morel G. (2009). New paradigms for a product oriented modelling: Case study for traceability, in: Special issue on Intelligent Products. Computers In Industry, 60/3, 172-183, April, Elsevier, ISSN: 0166-3615.

Bermundez J., Goni A., Illaramendi A., Bagues M.I. (2007). Interoperation among agent-based information systems through a communication acts ontology, Inf. Syst. 32 (8) ,1121–1144.

Botta V. -Genoulaz, P.- Millet A., Grabot B. (2005). A survey on the recent research literature on ERP systems, Comput. Ind. (56) , 510–522.

Boulanger D., Dubois G., (1998). An object approach for information system cooperation, Inf. Syst. 23 (6) , 383–399.

Brickley D, Guha RV. Resource Description Framework (RDF) Schema Specification 1.0. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

Chen D, Doumeingts G. (2003). Basic concepts and approaches to develop interoperability of enterprise applications, in: Camarinha - Matos LM, Afsarmanesh H, editors. In proceedings of IFIP TC5/WG5.5 fourth working conference PRO-VE'03, processes and foundations for virtual organisations, p. 323–30, October 29–31, 2003, Lugano, Switzerland, Dordecht: Kluwer; . ISBN 1-4020-7638-X.

Chen D., Vernadat F.B. (2002). Enterprise Interoperability: a standardisation view. IFIP International Conference on Enterprise Integration and Modelling Technology (ICEIMT'02), 273-282, April 24-26, 2002, Valencia, Spain, , Kluwer Academics Publisher, ISBN 1-4020-7277-5.

Chen D., Doumeingts G. (2003). European initiatives to develop interoperability of enterprise applications—basic concepts, Annu. Rev. Control (27) 153–162.

Dogac A., Laleci G.B., Kirbas S., Kabak Y., Sinir S.S., Yildiz A., Gurcan Y.(2006). Artemis: deploying semantically enriched Web services in the healthcare domain, Inf. Syst. 31, 321–339.

Domınguez E., Zapata M. A. (2000). Mappings and interoperability: a meta- modelling approach, in: T. Yakhno (Ed.), Proceedings of ADVIS 2000, LNCS 1909, 352–362, Springer, 2000.

EN/ISO 19439, Enterprise Integration—Framework for Enterprise Modelling, 2003.

EIF (2004), European Interoperability Framework for pan-European eGovernment Services, Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens (IDABC), November, Luxembourg.

Euzenat J. (2001). Towards a principled approach to semantic interoperability, CEUR Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing, Vol.47., 19-25, , August 4-5, 2001, Seattle, USA, http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-47, ISSN 1613-0073.

Grangel R., Bourey J. P., Berre A. (2006). Solving Problems in the Parametrisation of ERPs using a Model-Driven Approach, in: Proceedings of the 2nd IFAC/IFIP International conference on Interoperability for Enterprise Applications and Software (I-ESA'2006), Enterprise Interoperability, 461-472, March 22-24, 2006, Bordeaux, France, Springer Verlag, ISBN 978-1-84628-713-8.

Guarino N. (1998). Formal ontology and information systems. In: Guarino N, editor. Formal ontology and information systems. IOS Press.Hameed A, Preece A, Sleeman D (2003). Ontology

reconciliation, in: Staab S, Studer R (eds) Handbook on ontologies in information systems. Springer, Berlin Heidelberg New York, 231–250.

IEC 62264 (2002). IEC 62264-1:2002. Enterprise-control system integration, Part 1. Models and terminology, ISO/IEC, Geneva.

INTEROP. (2003, April 23). Interoperability research for networked enterprises applications and software, network of excellence. Proposal part B.

ISO 14528 (1999). Industrial Automation Systems – Concepts and rules for Enterprise Models, TC 184/SC5/WG1, Geneva, Switzerland.

ISO EN DIS 19440 (2004). Enterprise integration - Constructs of enterprise modelling, Draft version, TC 184/SC5/WG1, Geneva, Switzerland.

Kotinurmi P. (2005). Towards more intelligent business-to-business integration with semantic web service technologies. Proceedings of the CIMRU-DERI-HP research seminar, The Digital Enterprise Research Institute, Galway, Ireland, 33–35.

Kühn H., Murzek M (2006). Interoperability in metamodelling platforms, in: D. Konstantas, J.-P. Bourriéres, M. Lèonard and N. Boudjlida (eds.): 1st International Interoperability of Enterprise Software and Applications (I-ESA), 216-226, Springer, 2005, Geneva, Switzerland, Berlin heidelberg.

Madnick SE (1995). From VLDB to VMLDB (Very MANY Large Data Bases): dealing with large-scale semantic heterogeneity. Proceedings of the 21st very large data base conference, , 11–16, Zurich.

Mak K.-T., Ramaprasad A.(2001). An Interpretation of the Changing IS/IT-Standard Game, Circa 2001, Knowl. Technol. Policy 12 (14) , 20–30.

Morel G., Panetto H., Mayer F., Auzelle J.P. (2007). System of enterprise-Systems integration issues :an engineering perspective. Invited plenary paper. IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2 – 5, Monterrey, Mexico, IFAC Papersonline.

OASIS, (2002). Organization for the Advancement of Structured Information Standards. ebXML specification, February, http://www.oasis-open.org.

Oh S.C., Yee S-T (2008). Manufacturing interoperability using a semantic mediation, in: Int J Adv Manuf Technol 39:199–210 DOI 10.1007/s00170-007-1198-2.

Panetto H. (2007). Towards a Classification Framework for Interoperability of Enterprise Applications. International Journal of CIM, 20/8, 727-740, Taylor & Francis, December 2007, ISSN 0951-192X.

Panetto H., Molina A. (2008). Enterprise Integration and Interoperability in Manufacturing Systems: trends and issues. In: Special issue on Enterprise Integration and Interoperability in Manufacturing Systems, A. Molina and H. Panetto (Eds). Computers In Industry, 59/5, May, Elsevier, ISSN: 0166-3615.

Patil L., Dutta D., Sriram R. (2005), Ontology-based exchange of product data semantics, In: IEEE Transactions of Automation Science and Engineering ; 2, 213–225.

Rahm E., Bernstein P.A.(2001). A survey of approaches to automatic schema matching, VLDB J. 10, 334–350.

UML (2005). Unified Modeling Language. UML 2.0 Superstructure, v2.0 formal 05/07/04. OMG.

W3C (2003). Simple Object Access Protocol (SOAP) 1.2, W3C specification, February, http://www.w3c.org.

W3C (2004a), XML, Extensible Mark-up Language, W3C XML 1.1 recommendation, February, http://www.w3c.org.

W3C (2004b). Web Services Description Language (WSDL) 2.0, W3C specification. August, http://www.w3c.org.

W3C. OWL Web ontology language overview, http://www.w3.org/TR/2003/PR-owl-features-20031215/; 2005 [last accessed 10.05].

Wainwright D., Waring T.(2004). Three domains for implementing integrated information systems: redressing the balance between technology, strategic and organisational analysis, Int. J. Inf. Manag. 24, 329–346.

Whitman L.E., Panetto H.(2006). The missing link: culture and language barriers to interoperability, Annu. Rev. Control 30, 233–241.

Xu X.W., Newman S.T.(2006). Making CNC machine tools more open, interoperable and intelligent—a review of the technologies, Comput. Ind. 57 (2), 141–152.

Chapter 2: Product Ontology

1. Introduction

As enterprises are subject to cope with frequently changing business environment, enterprises should integrate value chains in order to reduce time-to-market. Sharing product information must precede for integration between enterprises that participate into a value chain. However, most of the participants have different business experience and business domains, interoperability of information among enterprises should be guaranteed in order that enterprises collaborate with other participants for integration of value chain. There are two main kinds of interoperability of information: syntactic and semantic. Syntactic interoperability can be achieved by defining electronically exchanged scheme, such as ebXML, while semantic interoperability which will enable machine-understandable data to be shared across the value chain can be achieved by ontological engineering process (Jeongsoo et al., 2009).

In an enterprise context, product information is the most basic information that is referring to not only systems and applications in the enterprise but also the related stakeholders out of the enterprise. For the semantic interoperability of product information, a product ontology which is commonly used by the related enterprises which participate in the value chain should be useful (Jeongsoo et al., 2009). An ontology, in fact, allows the formalization of the semantics of objects in order to express and to share knowledge about them.

The structure of the chapter is the following: in the first part a general description of the ontology domain will be given, describing types, language, tools, application area and open issues; in the second part a particular focus will be made on the Product Ontology, describing the related existing projects and work and exploiting the main product information and data standards.

2. Ontology: general issues

Ontologies were developed in Artificial Intelligence to facilitate knowledge sharing and reuse. The term “ontology” is borrowed from philosophy, where it is a systematic account of Existence. For AI systems, what “exists” is that which can be represented. When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the *universe of discourse*. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which a knowledge-based program represents knowledge. Thus, in the context of AI, it is possible describe the ontology of a program by defining a set of representational terms. In such ontology, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms. Formally, an ontology is the statement of a logical theory.

Since the beginning of the nineties, ontologies have become a popular research topic investigated by several Artificial Intelligence research communities, including Knowledge Engineering, natural-language processing and knowledge representation. More recently, the notion of ontology is also becoming widespread in fields such as intelligent information integration, cooperative information systems, information retrieval, electronic commerce, and knowledge management. The ontologies are becoming so popular, this in large part due to what they promise: a shared and common understanding of some domain that can be communicated between people and application systems. Because ontologies aim at consensual domain knowledge, their development is often a cooperative process involving different people, possibly at different locations. People who agree on a specific ontology are said to commit themselves to that ontology (Ding et al., 2002).

Many definitions of ontologies have been given in the last decade, but one that best characterizes the essence of an ontology is based on the related

definitions by Gruber (Gruber,1993): *An ontology is a formal, explicit specification of a shared conceptualization*. A 'conceptualization' refers to an abstract model of some phenomenon in the world which identifies the relevant concepts of that phenomenon. 'Explicit' means that the type of concepts used and the constraints on their use are explicitly defined. 'Formal' refers to the fact that the ontology should be machine understandable.

A conceptualization is the extraction of vocabularies from a domain and is an abstract, simplified view of the world that we wish to represent for some purpose. Through this conceptualization, concepts and their relations are extracted from the real world. Because ontologies consist of the shared vocabularies used to describe the concepts and the relations (Gruber, 1993), ontologies can be used as tools for specifying the semantics of terminology systems in a well defined and unambiguous manner (Guarino 1998; Gruber, 1993). Jasper and Uschold identified three major uses of ontologies (Jasper and Uschold, 1999):

(i) to assist in communication between human beings,

(ii) to achieve interoperability (communication) among software systems,

(iii) to improve the design and the quality of software systems.

We focus on (i) and (ii) from the viewpoint of communication (semantic interoperability). To guarantee semantic interoperability in a domain, a common ontology for the domain should exist. Otherwise, a new ontology should be built. If semantic interoperability across different domains is needed, a new temporal ontology for a virtual domain which includes all related domains should be built (Kalfoglou and Schorlemmer, 2002). A new ontology is built through the following procedure: identify purpose, ontology capture, ontology coding, integrating existing ontology, evaluation, and documentation (Uschold amd King, 1995).

There is an agreement in the ontology community that the integration of existing ontologies is the more beneficial way to eliminate time, cost, and effort for building a new ontology (Noy and Hafner, 1997). Guaranteeing

semantic interoperability among ontology for integration is a key factor for building ontologies efficiently and guaranteeing semantic interoperability across domains (Van Heijst et al., 1997).

2.1. Types of ontologies

Depending on their generality level and the domain in which they are applied, different types of ontologies may be identified that fulfil different roles in the process of building a knowledge-based system (Guarino, 1998; Van Heijst et al., 1997). Among others, we can distinguish the following ontology types:

- *Metadata ontologies* also called *Generic ontologies* or *Core ontologies* (Van Heijst, 1997) are reusable across domains. An example is Dublin Core (Weibel et al., 1995) that provide a vocabulary for describing the content of on-line information sources.
- *Generic* or *common sense ontologies* aim at capturing general knowledge about the world, providing basic notions and concepts for things like time, space, state, event etc. (Fridman-Noy and Hafner, 1997; Pirlein and Studer, 1997). As a consequence, they are valid across several domains.
- *Top-Level Ontologies* provide general notions under which with all the terms in existing ontologies are related.
- *Representational ontologies* do not commit themselves to any particular domain. Such ontologies provide representational entities without stating what should be represented. A well-known representational ontology is the *Frame Ontology* (Gruber, 1993), which defines concepts such as frames, slots, and slot constraints allowing the expression of knowledge in an object-oriented or frame-based way.
- *Domain ontologies* (Mizoguchi et al., 1995; Van Heijst et al., 1997) capture the knowledge valid for a particular type of domain (e.g. electronic, medical, mechanic, digital domain). They provide

vocabularies about the concepts within a domain and their relationships, about the activities that take place in that domain, and about the theories and elementary principles governing that domain. In the domain of enterprise modelling process, the Enterprise Ontology (Uschold amd King, 1995) is a collection of terms and definitions relevant to business enterprises.

- Other types of ontology are so-called *method* and *task ontologies* (Fensel and Grenboom, 1997; Studer et al., 1996). Task ontologies provide terms specific for particular tasks and method ontologies provide definitions of the relevant concepts and relations used to specify a reasoning process to achieve a particular task. Task and method ontologies provide a reasoning point of view on domain knowledge.
- *Application ontologies* (Van Heijst et al., 1997) contain the necessary knowledge for modelling a particular application.

Part of the research on ontologies is concerned with envisioning and building enabling technology for the large-scale reuse of ontologies at a world-wide level. In order to enable as much reuse as possible, ontologies should be small modules with a high internal coherence and a limited amount of interaction between the modules (Dieter, 2000).

Among these types of ontologies, we propose to develop a domain ontology for the proposed model, because it has the degree of generality suitable to formalize concepts related to enterprise applications, but which at the same time, belong to different domain, such as business and manufacturing ones. In fact, a Top/Upper Ontology is too generic for our scope: it formalizes general or high level concepts such as processes, time, region, physical objects, and the semantic relationships of these notions. An Application Ontology, instead, is too specific: it does not allow taking into account concepts from heterogeneous applications, such as an ERP, a PDM and a MES.

2.2. Ontology Languages

Ontologies are formal theories about a certain domain of discourse and require a formal logical language to express them. The languages for describing ontologies are:

1. First-Order Logic languages, such as CycL and KIF (Genesereth, 1991).
2. Frame-based approaches languages, such as *Ontolingua* (Farquhar et al., 1997) and *Frame Logic* (Kifer et al., 1995), which incorporate frame-based modelling primitives in a first-order logical framework.
3. Description Logics (Baader et al, 2003) that describe knowledge in terms of concepts and role restrictions used to automatically derive classification taxonomies.

Generally, ontology is expressed with logic based languages: the first-order logics, the rules Languages, the non-classical logics and the Description Logics. All these languages are characterized by a formal specification of the semantics that allows expressing structured knowledge in one hand and promotes the implementation of reasoning support in the other hand.

2.3. Application Area

The three main application areas of ontology technology are Knowledge Management, Web Commerce, and Electronic Business (Dieter, 2000).

Knowledge Management is concerned with acquiring, maintaining, and accessing knowledge of an organization. It aims to exploit an organisation's intellectual assets for greater productivity, new value, and increased competitiveness. Due to globalisation and the impact of the Internet, many organizations are increasingly geographically dispersed and organized around virtual teams.

There are severe weaknesses in this area:

· *Searching information*: searching information may cause missing of information meaning, because information can use different words in different contexts.

· *Extracting information*: extracting information may cause failure in integrating information spread over different sources, because it requires common sense knowledge for understanding.

· *Maintaining weakly structured text sources* is a difficult and time-consuming activity when such sources become large. Keeping such collections consistent, correct, and up-to-date requires a mechanized representation of semantics and constraints that help to detect anomalies.

· *Automatic document generation*: Adaptive web sites which enable a dynamic reconfiguration according to user profiles or other relevant aspects would be very useful. The generation of semi-structured information presentations from semi-structured data requires a machine-accessible representation of the semantics of these information sources.

Ontologies will allow structural and semantic definitions of documents providing completely new possibilities: intelligent search instead of keyword matching, query answering instead of information retrieval, document exchange between departments via ontology mappings, and definition of views on documents.

Web Commerce (B2C): Electronic Commerce is becoming an important and growing business area. This is happening for two reasons. First, electronic commerce is extending existing business models. It reduces costs and extends existing distribution channels and may even introduce new distribution possibilities. Second, it enables completely new business models or gives them a much greater importance than they had before. What has up to now been a peripheral aspect of a business field may suddenly receive its own important revenue flow. Examples of business field extensions are on-line stores, examples of new business fields are shopping agents, on-line marketplaces and auction houses that make comparison shopping or meditation of shopping processes into a business with its own significant revenue flow. The advantages of on-line stores and

the success story of many of them has led to a large number of such shopping pages. The new task for a customer is now to find a shop that sells the product he is looking for, getting it in the desired quality, quantity, and time, and paying as little as possible for it. Achieving these goals via browsing requires significant time and will only cover a small share of the actual offers. Very early, shopbots were developed that visit several stores, extract product information and present to the customer a instant market overview. Their functionality is provided via *wrappers* that, written for each on-line store, use a keyword search for finding the product information together with assumptions on regularities in the presentation format of stores and text extraction heuristics. This technology has two severe limitations:

· *Effort*: Writing a wrapper for each on-line store is a time-consuming activity and changes in the outfit of stores cause high maintenance efforts.

· *Quality:* The extracted product information is limited (mostly price information), error prone and incomplete.

These problems are caused by the fact that most product information is provided in natural language, and automatic text recognition is still a research area with significant unsolved problems. However, the situation will drastically change in the near future when standard representation formalisms for the structure and semantics of data are available. The low-level programming of wrappers based on text extraction and format heuristics will be replaced by ontology mappings, which translate different product descriptions into each other. An ontology describes the various products and can be used to navigate and search automatically for the required information.

Electronic Business (B2B): Electronic Commerce in the business to business field (B2B) is not a new phenomena. Initiatives to support electronic data exchange in business processes between different companies existed already in the sixties. In order to exchange business transactions sender and receiver have to agree on a common standard (a protocol for transmitting the content and a language for describing the

content). A number of standards arose for this purpose. Using the infrastructure of the Internet for business exchange will significantly improve this situation. Standard browsers can be used to render business transactions and these transactions are transparently integrated into other document exchange processes in intranet and Internet environments. However, this is currently hampered by the fact that HTML do not provide a means for presenting rich syntax and semantics of data. XML, which is designed to close this gap in current Internet technology, will therefore drastically change the situation. B2B communication and data exchange can then be modelled with the same means that are available for the other data exchange processes, transaction specifications can easily be rendered by standard browsers, maintenance will be cheap. XML will provide a standard serialized syntax for defining the structure and semantics of data. Still, it does not provide standard data structures and terminologies to describe business processes and exchanged products. Therefore, ontologies will have to play two important roles in XML based electronic commerce:

· *Standard ontologies* have to be developed covering the various business areas. In addition to official standards, vertical marketplaces (Internet portals) may generate de facto standards. Examples are: Dublin, Common Business Library (CBL), Commerce XML (cXML), ecl@ss, Open Applications Group Integration Specification (OAGIS), Open Catalog Format (OCF), Open Financial Exchange (OFX), Real Estate Transaction Markup Language (RETML), RosettaNet and UN/SPSC.5.

· *Ontology-based translation services* between different data structures in areas where standard ontologies do not exist or where a particular client wants to use his own terminology and needs translation service from his terminology into the standard. This translation service must cover structural and semantical as well as language differences.

Then, ontology-based trading will significantly extend the degree to which data exchange is automated and will create complete new business models in the participating market segments.

2.4. Tools

Effective and efficient work with the ontologies requires the following elements (Ding et al., 2002):

- *Editors and semi-automatic construction to build new ontologies*: Ontology editors help human knowledge engineers to build ontologies. Ontology editors support the definition of concept hierarchies, the definition attributes for concepts, and the definition of axioms and constraints. They must provide graphical interfaces and must confirm to existing standards in web-based software development. They enable inspecting, browsing, codifying and modifying ontologies and supports in this way the ontology development and maintenance task. An example system is *Protégé* (Grosso et al., 1999).
- *Reasoning Service*: Instance and Schema Inferences enable advanced query answering service, support ontology creation and help to map between different terminologies. Inference engines for ontologies can be used to reason about instances of an ontology or over ontology schemes. Reasoning over Instances of an ontology, for example, derive a certain value for an attribute applied to an object. Reasoning over Concepts of an ontology, for example, automatically derive the right position of a new concept in a give concept hierarchy. Racer, Pellet, FaCT++ are types of reasoners that help to build ontologies and to use them for advanced information access and navigation.
- *Reusing and Merging Ontologies*: Ontology library systems and Ontology Environments help to create new ontologies by reusing existing ones. Assuming that the world is full of well-designed modular ontologies, constructing a new ontology is a matter of assembling existing ones. Instead of building ontologies from scratch, one wants to reuse existing ontologies. This requires two types of tools: (1) tools to storage and retrieve existing Ontologies and (2) tools that help manipulate existing. A tool environment which can be able the union of two ontologies (ontology inclusion) is *Chimaera* (McGuinness et al., 2000), which provides support for two important tasks: (1) merging multiple ontologies and (2) diagnosing

(and evolving) ontologies. The PROMPT27 tool (Noy and Musen, 2000) is available as a plug-in for Protégé-2000 targeted to help the user in ontology merging. It takes two ontologies as input and guides the user in the creation of a single merged ontology as output.

For the purpose of this thesis, the logics on which reusing and merging ontologies tools are based will be useful for the development of the ontological information model, starting from existing standardization initiatives which can be considered “a sort of ontologies” that try to formalise the knowledge related to products technical data (STEP PDM, IEC 62264).

2.5. Open issues

As ontologies become more popular and are also used in real-life situations, new problems arise. Two important topics that the ontology research community is currently facing are (Ding et al., 2002):

• *Evolving ontologies*: how to manage ontologies that change over time. Ontologies are often not stable definitions that never change. One of the reasons for this is that a shared conceptualization of a domain has to be reached in a process of social communication. Other reason for modification of the ontology are changes in the domain and adaptation to a different task. The evolution of ontologies causes operability problems, which hamper their effective reuse. Solutions are required to allow changes to ontologies without making current use invalid.

• *Combining ontologies*: how to relate and align separately developed ontologies to use them together. Nowadays, people start annotating data with standard terminologies and other semantic data. This is providing us with a lot of freely accessible domain specific ontologies. However, to have a unique semantics which will allow to combine and infer implicit knowledge, those separate ontologies should be linked and related to each other. Adaptation of existing ontologies, and composition of new

ontologies from ontologies that are already around are important open issues today.

3. Product Ontologies

Increasing product complexity, growing competition, emerging globalization, and stronger customer focus force the majority of enterprises to network their own geographically dispersed sites and to extensively cooperate with customers or suppliers. Thus, products' data is spread among different areas of an organization and also may be distributed through different organizations. A lot of information is created in Engineering areas, but product information is also created and used in the Manufacturing, Marketing, Finance, Sales and Planning areas. These areas are often characterized by heterogeneous environments in which product data may be represented in different ways. When there is no standard definition of the data associated with a particular product or product component, each user (and application program) can have a different definition of the data, and all the definitions could be different. This leads to errors, as well as wasted time and money.

The latest developments of information and communication technologies establish a platform for worldwide cooperation and collaboration within engineering, since the problems of geographical and time related distance have nearly disappeared. However, these new technologies require intelligent integration between different and heterogeneous systems. "Intelligent Integration" implies three main aspects: technical, syntactical and semantics. Internet and Web technology give support to the two first aspects while the latter may be solved through the definition of domain ontologies.

Certain research efforts have focused on issues that are of relevance to the problem of semantic interoperability of product information.

Yoo and Kim (Yoo and Kim, 2002) have presented a Web-based knowledge management system for facilitating seamless sharing of

product data among application systems in virtual enterprises. Current research activities in this area are oriented towards the use of ontologies as a foundation for the ‘‘Semantic Web’’ (Berners-Lee et al., 2001). Vegetti et al. (Vegetti et al., 2005) have made a contribution, proposing an ontology called PRoduct ONTOlogy (PRONTO), which intend to provide a consensual knowledge model of the product modelling domain that can be used by all the stakeholders of extended supply chains involving industrial organizations. PRONTO presents concepts involved in the product modelling domain that are primarily related with the product structure. It considers distinct levels of abstraction in relation to the product concept: *Family* and *Variant*. These levels allow performing planning activities with different aggregation detail. The ontology also presents the composition and decomposition structures concepts enabling its use in different kinds of industries with complex product structures, not considered in traditional BOMs. Through the specification of inference rules the model allows doing the requirements explosion, both for products with composition structures as well as products with a hybrid structure (composition and decomposition) (Vegetti et al., 2008).

An effort of significant relevance is the development of Product Ontology at the National Institute of Standards and Technology (NIST). PSL (Process Specification Language) defines a neutral representation for interoperability of information relevant to manufacturing processes. It considers the representation of process data used throughout the life cycle of a product and an ontology is being developed to facilitate exchange of information among various manufacturing process related software. (Patil et al., 2005) proposes an ontology-based framework to enable the semantic interoperability across different application domains. Building blocks of an ontology (Product Semantic Representation Language, PSRL) are defined for an intuitive and comprehensive representation of product information. PRSL uses the *Core Product Model* (CPM) as a basis for the development of a formal representation of product information. The

Core Product Model presents a generic product representation scheme for the entire product development activity.

In (Kim et al., 2003), an approach toward the development of a product ontology and semantic mapping using first-order logic is presented. This effort proposes the development of a shared ontology.

In (Dartigues, 2003), an ontological approach is proposed to enable the exchange of features between application software. It uses the knowledge interchange format (KIF) (Genesereth and Fikes, 1992) to model participating ontologies and to create a common intermediate ontology. Rules are manually specified to enable mapping of concepts from one domain to another.

Some efforts in the same direction are made by PROMISE-PLM European project (Kiritsis et al., 2003; Jun and Kiritsis, 2007), whose objective is to develop a new generation of product information tracking and flow management system, with a particular focus on use, service and maintenance phase of product lifecycle. The approach used bases on (1) PEID (Product Embedded Information Devices), such as RFID; (2) Data, information and knowledge modelling and web-based programming, such as EPC code, Product Markup Language (PLM), Object naming service; (3) Web-enabled and embedded predictive e-service technologies, such as e-maintenance; (4) Distributed decision making logistics, e.g. by using multi-agent technologies; (5) innovative product life-cycle modelling and simulation technologies that allow the evaluation and validation of a product system through its whole life cycle.

Another interesting project of the European Community is PABADIS'PROMISE, called P2 project (www.pabadis-promise.org), which stipulates an innovative control and networking architecture across the three levels of automation. At ERP levels, functions and interfaces will enable to directly access from ERP level to the field control system following an ontology (the P2 Ontology), which is to provide a framework for product and production processes description and comparison. The

MES level and the Field Control level will be completely decentralized. In particular, the MES level will be decentralized into mobile software agents, which will be located in smart tags which will be attached directly on product (agent RFID). The scope of the P2 Ontology in the project is to define a manufacturing ontology allowing future P2 components and applications to become fully interoperable with each other throughout the manufacturing process life cycle. The P2 Ontology will provide formal and unambiguous definition of all the components and of their interactions with each other in an enterprise/industrial environment. The P2 Ontology aims to formalize conceptual information about:

- Each resource which can be used in a production line: machines, equipment, control systems, actuators, personnel, materials, etc.
- Each product which can be produced (i.e. transformed via a process) in this production line
- Each operation through the definition of each process (defined as a set of sequential or parallel operations): to drill, to move, to transport, to measure,…

Regarding the integration of the P2 Ontology in the P2 Architecture it possible to follow two approach: (a) a centralized Ontology Repository, which could provide a common semantic reference for all agents or (b) a distributed ontology repositories comprising parts of the P2 Ontology.

Although, all these works related to Product Ontology have the same final objective and they are based on the same logics, they have been either related to geometry data or they have focused only on generic product information (PRONTO and PSL) or they have focused their study in the technology rather than to the conceptual view of product data exchange (PROMISE-PLM and PABADIS'PROMISE projects). Our contribution to Product Ontology is rather a domain ontology, suitable for exchanging product technical data between enterprise applications.

4. Standards for Product Data

Recalling that the goal of ontologies is to facilitate knowledge sharing, ontologies are often developed with the explicit goal of providing the basis for future semantic integration. Then, an ontology is agreed upon by developers of different applications or systems, who integrate in a general ontology, concepts and properties specific to their applications. Finding correspondences between application models facilitate a common "grounding".

On the same process is based the creation of a standard. A standard represent a sharing knowledge, developed by a group of expert, who try an agreement on a specific domain. Interesting standardisation initiatives try to formalise the knowledge related to products technical data in order to solve the problem of managing heterogeneous information coming from different systems. They are related to Product Data Management at the business and the manufacturing levels of enterprises (B2M) and for these reasons they can be considered as a sort of "Product Ontology".

Diverse efforts spent in the area of formalization of product data and information had became (or are becoming) accepted standards. The "way" of standardization is a long trip and not all the standards defined by official organizations (e.g. ISO, ISA, CEN) are always accepted and adopted in the reality of the day-by-day product interoperability. On the contrary, diverse references are considered as de facto standards, even if normative offices do not already accept them.

Looking to the literature of official and de facto standards distributed along the product life-cycle, it is possible to identify three main categories of product information standard: standard covering the Product Development phase, standards covering the Product Production phase, standards covering the Product Use phase. Obviously, this is only a subjective categorization, and it might be observed that always product information standards stay in an overlapping stage (Terzi, 2005).

4.1. Product Development Standards

In the phase of Product Development exist several standards; the most important for the purpose of this thesis are described below.

ISO 10303

The most important and well-accepted standard in this phase is the mentioned STEP initiative (STandard for the Exchange of Product model data), which is an ISO (ISO 10303) standard for the computer-interpretable representation and exchange of product definition data. It was developed with the aim to provide a mechanism capable of describing product data throughout the life cycle of a product, independently from any particular system. Its natural implementation is that of computer system and CAD, CAM, CAE software for product design.

Nowadays, STEP has been recognized as appropriate to help in the integration of manufacturing systems in industries such as automotive, aircraft, shipbuilding, furniture, building and construction, gas and oil.

The way it was designed for describing product data makes it suitable for neutral file exchange among different software solution, also in a distributed engineering or manufacturing environment. It can also operate as a basis for implementing and sharing product databases and archiving. One of the most important aspects of STEP is its extensibility: STEP is based on a modular and reconfigurability structure, which uses Application Protocols (APs) to specify the representation of product information for one or more applications (Figure 8). Application Protocols are sub-sets of STEP, focused on specific issues or specific industrial sectors, which break the entire STEP standard into easily manageable views of quick implementation. STEP initiative adopts a strategy of specification into industrial context (e.g. APs for product design, for mechanical and electrical engineering, for sheet metal manufacturing, for product assembly, for automotive industry).

STEP uses the EXPRESS language for describing data type, constraints on data type and relationship between data type. However, Application Protocols are required to contain a representation of the information in

both EXPRESS and EXPRESS-G. EXPRESS-G is a diagramming technique supporting a subset of EXPRESS language.

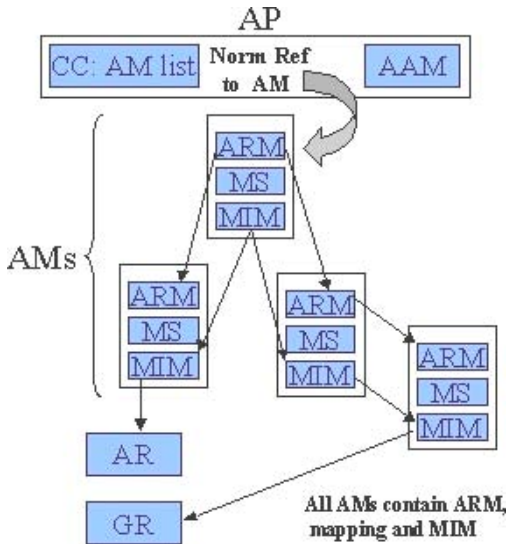


Figure 8 - Complex structure of an AP (ISO 10303)

A significant solution for PDM (Product Data Management) data exchange is the Unified PDM Schema, which is a basic specification for the exchange of administrative product definition data. It has been created by unifying all PDM data between all existing STEP Application Protocols, and allows the exchange of information that is stored in PDM systems. This information typically forms the metadata for any product. In order to deal with the increasing demands on product models exchange, the standard has specified a set of STEP reusable modules related to PDM. These modules are now published as technical specifications (TS) and concern all related information attached or describing products technical data such as product structure, configuration control, persons and organisations, etc. PDM systems maintain a single copy of the product master data in a secure vault; the data are then distributed to those departments requiring them: modified, updated design data are then resaved in the vault. Data integration ensures that the information describing product design, manufacturing and life cycle support is defined only once; STEP data integration eliminates redundancy and the problems caused by redundant information.

PLM XML

In the same phase, there is a de facto standard: PLM XML. PLM XML is an open standard proposed by EDS (currently UGS PLM Solutions) to facilitate high-content product lifecycle data sharing. PLM XML derives partly from the STEP initiative, even if it is currently maintained by EDS/UGS R&D team in an open source way (Figure 9). PLM XML provides a reference framework and a reference data format, based on XML, for the main sub-phases of Product Development, from Product Design to Plant Design and Process Design.

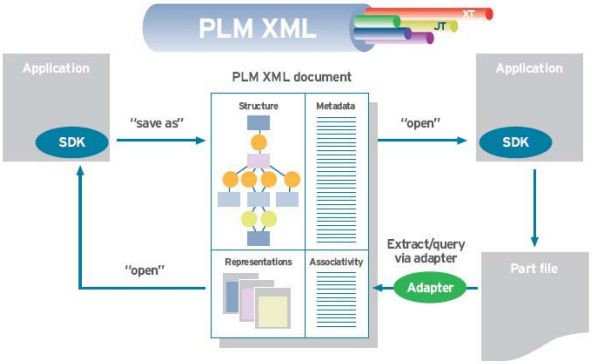


Figure 9 – PLM XML main functionalities (www.ugsplm.com)

4.2. Product Production Interoperability Standards

The Product Production phase deals with product manufacturing and distribution and all the related sub-activities. Into this phase, for a clear understanding are also considered all the activities acting at Operation Management level, like the relations with suppliers and customers, even if they are not directly related to the product itself.

ISO 62264

The IEC 62264 set of standards specify a set of reference models extending the ANSI/ISA S95 specifications, that defines an information exchange framework to facilitate the integration of business applications and manufacturing control applications, within an enterprise. It is composed by six different parts designed for defining the interfaces between enterprise activities and control activities. Among all its parts, part 1 describes the relevant functions within an enterprise and within the

control domain of an enterprise, stating which objects are normally exchanged between these domains (Figure 10) depicts the different levels of a functional hierarchy model: business planning and logistics, manufacturing operations and control, and batch, continuous, or discrete control.

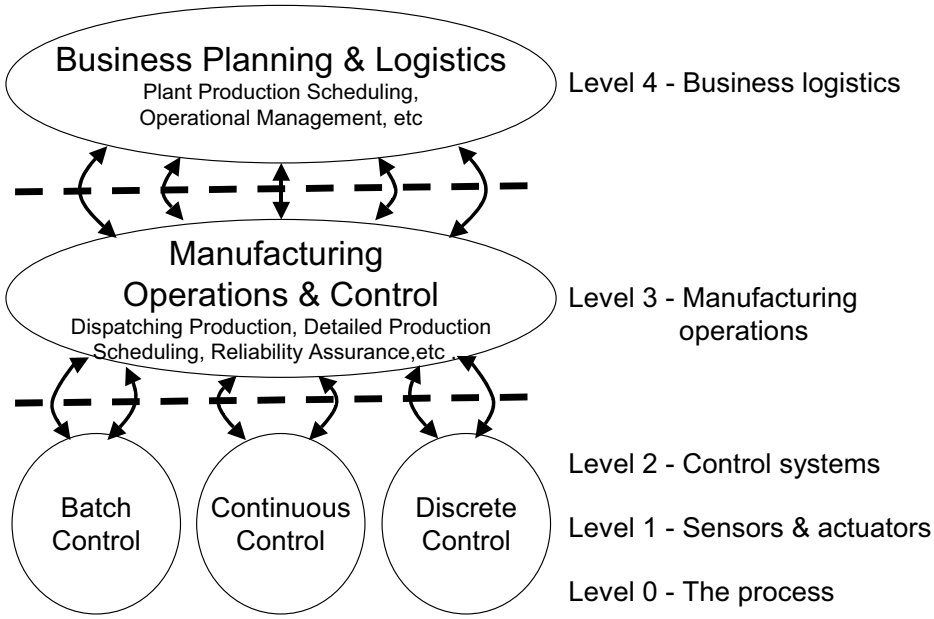


Figure 10 - Functional hierarchy as defined in IEC 62264 (IEC 62264, 2002)

The model shows the hierarchical levels at which decisions are made. The interface addressed in the standard is between Level 4 and Level 3 of the hierarchy model. This is generally the interface between plant production scheduling and operation management and plant floor coordination.

Levels 2, 1, and 0 present the cell or line supervision functions, operations functions, and process control functions, not addressed by this standard. The key aspects for integrating the business applications at Level 4 and the manufacturing operations and control applications at Level 2 (and below) are the information structures and exchanges managed by Level 3 activities, applications, processes, resources, and functions. Examples of Level 3 activities include the management of various manufacturing operations, such as: production, maintenance, product quality testing, and material handling.

The boundary between the enterprise manufacturing operations and control domains are signed by models: *hierarchy model* that describes the levels of functions and domains of control associated within manufacturing organizations; *data flow model* that describes the functional and data flows within manufacturing organizations; *object model* that describes the information that may cross the enterprise and control system boundary.

To take into account the various exchanged information, through the product representation, the standard defines a set of eight models that specifies all concepts for enterprise-control integration: three are related to the resource hierarchy (Personnel, Equipment, Material), the process hierarchy (Process Segments, Product Definition), and to the production (Production Schedule, Production Performance, Capability Definition).

Each model concerns a particular view of the integration problem. Those models show increasing detail level and are operational models or resource models.

The different models from IEC 62264 are linked together in a logical way in order to define a hierarchy of models (Figure 11):

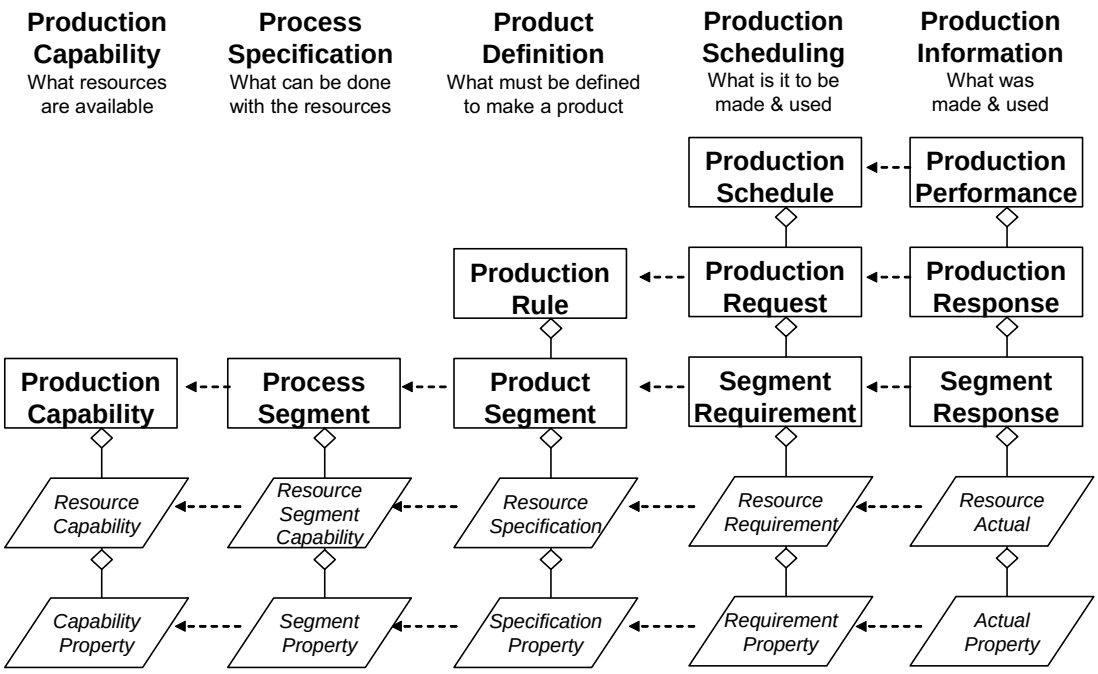


Figure 11 - The IEC 62264 models hierarchy (IEC 62264, 2002)

- The production information presents what was made and what was used. Its elements correspond to information in production scheduling that listed what to make and what to use.
- The production scheduling elements correspond to information in the product definition that shows what is specified to make a product.
- The product definition elements correspond to information in the process segment descriptions that present what can be done with the production resources.

IEC 62264 makes use of UML representation for displaying each “class” of information and its relations with other classes. Figure 12 depicts a UML diagram describing Production Capability class: this information-representing modelling class involves other information, such as those of personnel, materials or equipments capability (whose abstract UML representing elements are Personnel Capability class, etc.).

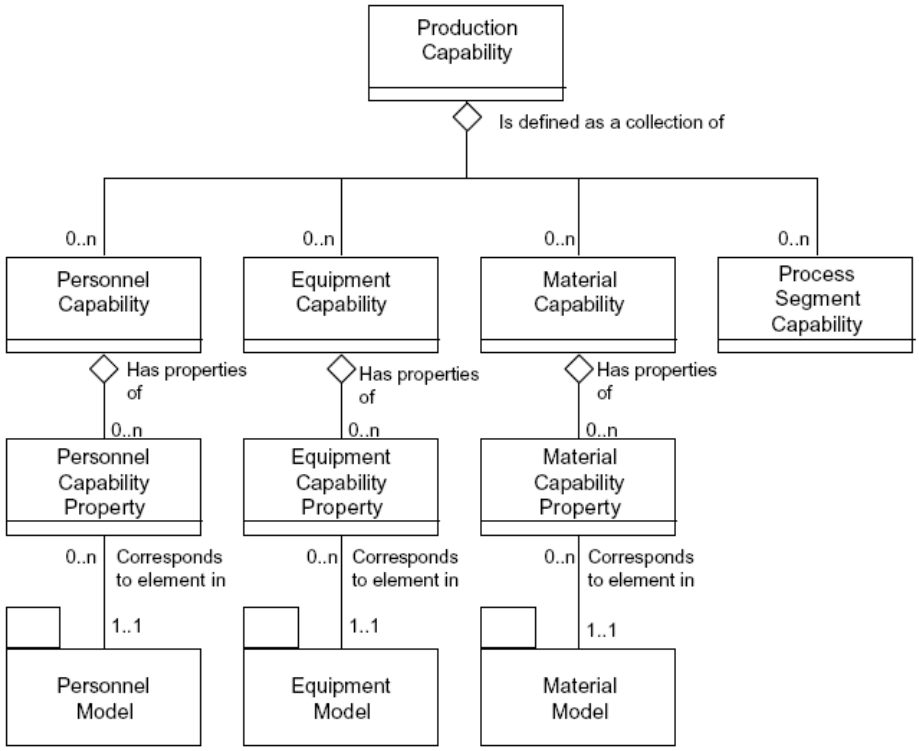


Figure 12 - Production capability model

B2MML (Business to Manufacturing Markup Language) is an XML implementation of the IEC 62264 part 1. It consists of a set of XML schemas, developed by the World Batch Forum, written using the World Wide Web Consortium's XML Schema language (XSD) that implements the standardised data models. B2MML is meant to be a common data format to link business enterprise applications (such as ERP systems) with manufacturing enterprise applications (such as MES). In particular, MES functions relate to production monitoring including materials (raw and finished) and resources (equipment and personnel) traceability information. Figure 13 shows the schemas definitions of B2MML using UML quotation for the Production Capability model.

```
<xsd:element name="ProductionCapability" type="ProductionCapabilityType" />
<!-- Simple & Complex Types -->
<xsd:complexType name="ProductionCapabilityType">
<xsd:sequence>
 <xsd:element name="ID" type="IDType" minOccurs="0" />
 <xsd:element name="Description" type="DescriptionType" minOccurs="0"
maxOccurs="unbounded" />
 <xsd:element name="Location" type="LocationType" minOccurs="0" />
 <xsd:element name="PublishedDate" type="PublishedDateType" minOccurs="0" /> .
```

Figure 13.- Example of an XSD in B2MML

Figure 14 shows the class diagram of Production Capability model, which for sake of visibility is split in two parts: the first part related to Material Capability and the second one related to Equipment and Personnel Capability. The classes that have relationships with equipment classes, personnel classes and material ones, such as LocationType, ProductionCapabilityType, EquipmentElementLevelType, ProcessSegmentCapabilityType and ProcessSegmentType are present in both the class diagrams.

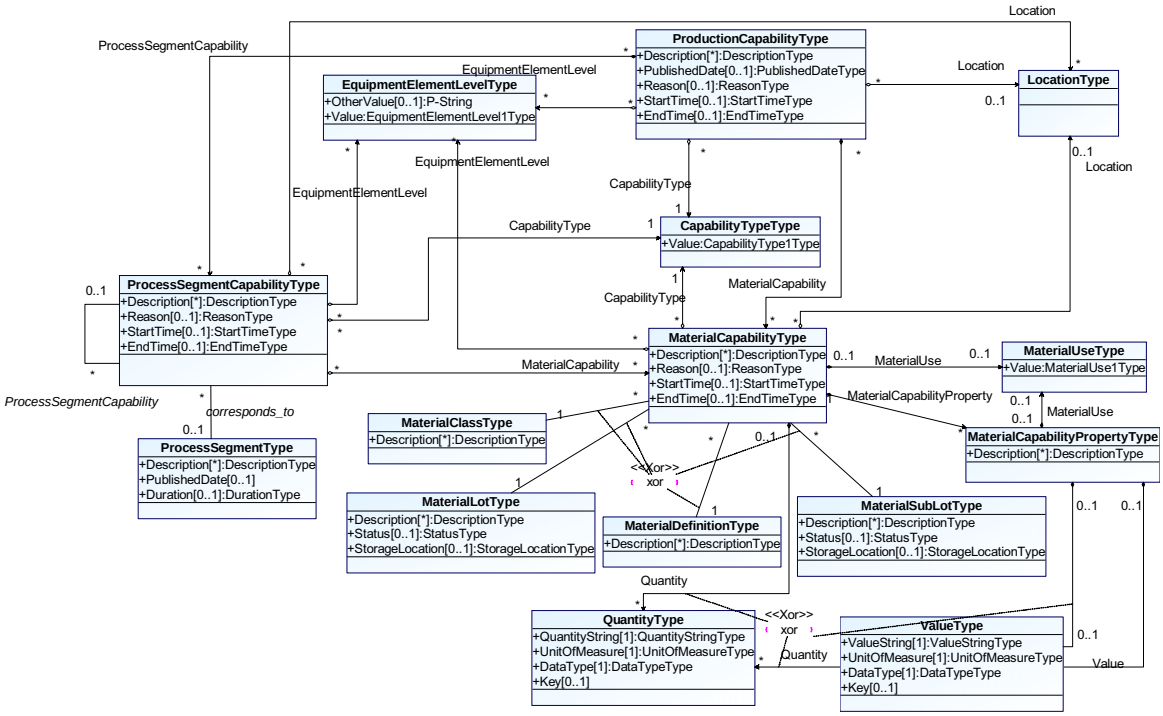


Figure 14 (a) - B2MML schemas definitions for production capability

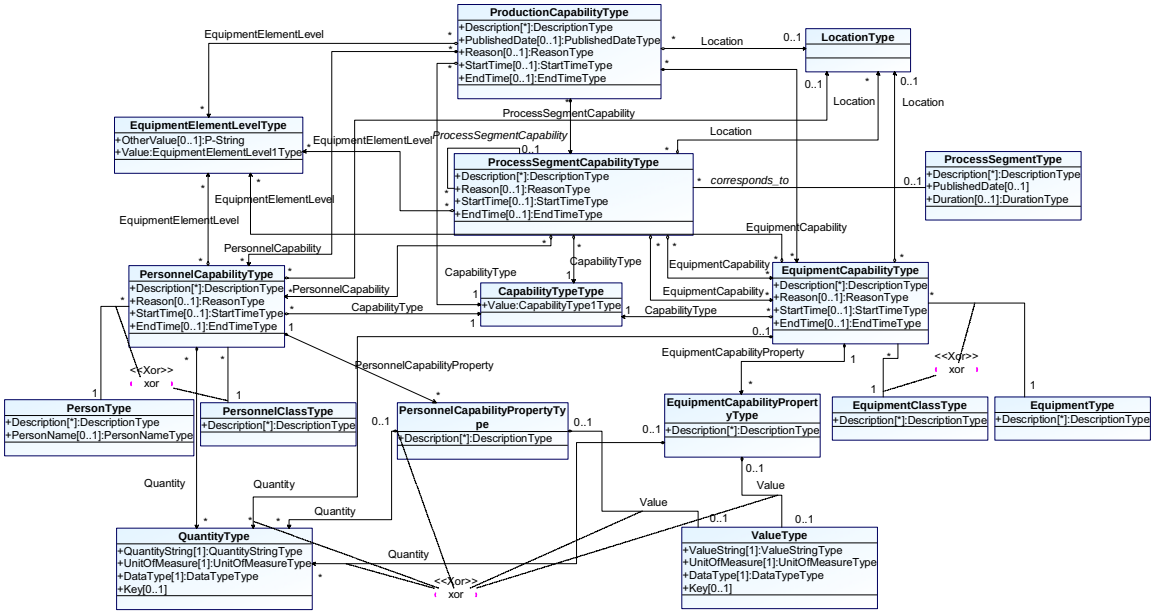


Figure 14 (b)- B2MML schemas definitions for production capability

MANDATE

Another interesting initiative is Mandate (MANufacturing DATa Exchange - ISO 15531), which is a part of the set of standards TC184/SC4. The Mandate scope is the representation of production information and

resources information including capacity, monitoring, maintenance and control and the exchange and sharing of production information and resources information including storing, transferring, accessing and archiving. Mandate is divided in three series of parts based on a common overview and fundamental:

- Parts 15531-2's series (Production data: external exchanges): those parts include all information and functions necessary to support quality, and order management, such as planning, executing, controlling and monitoring of product quality, orders and shipments.
- Parts 15531-3's series (Manufacturing Resources Management Data): those Parts refer to the resource usage management, such as resource configuration and capabilities, operation management of manufacturing devices, installation, quality features, maintenance-features (regarding the availability) and safety-features.
- Parts 15531-4's series (Manufacturing Flow Management Data): those parts refer to the flow material control, and intend to standardize data and elements, which support the control and monitoring of the flow of material in manufacturing or industrial processes.

Mandate initiative aims to be compliant with STEP architecture, but on contrary of STEP, which takes a product-oriented view of manufacturing, Mandate is concerned with the processes of the organization which are used to produce the products. By the contrary, parts 15531-3 aim to deal with aspects of “product” lifecycle (where the “product” is a machine), which more concern with Product Use phase (e.g. maintenance, installation). This aspect demonstrates how the desire of a comprehensive standardization along the whole product lifecycle (since to the product use itself) is highly considered.

4.3. Product Use Interoperability Standards

The phase of Product Use deals with the day-by-day life of the product itself.

PLCS

Another initiative is named PLCS- Product Life-Cycle Support (PLCS). PLCS is a standard based on ISO 10303 (STEP): furthermore, it is an Application Protocol of STEP (AP 239). It was born as an initiative supported by both industry and national governments with the aim to accelerate development of new standards for product support information. PLCS should be able to describe products needing support and the work required to sustain and maintain such product in operational conditions.

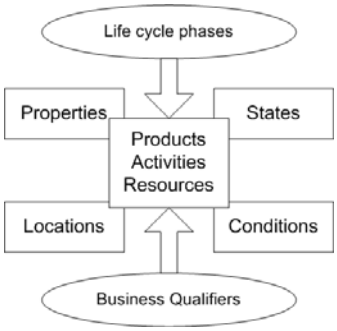


Figure 15 - PLCS concepts (www.plcs.org)

PLCS is based on three top-level concepts (Figure 15): Product, Activity and Resource. Each of these concepts is in relation with Properties, States or Locations and Conditions can be applied to their relationship.

PLCS uses the same ad-hoc developed language used for STEP (EXPRESS).

PML

The last interesting initiative is the Physical Mark-up Language (PML), developed by Auto-ID laboratories (McFarlane et al., 2003). PML is intended to be a general, standard means for describing the physical world. The objective of PML is a simple, general language for describing physical objects for use in remote monitoring and control of the physical environment. PLM was thought as a part of a wider structure whose purpose is that of linking physical objects to each other, people and information through the global Internet. This complex infrastructure is built around four major components: electronic tags, Electronic Product Code (EPC), Physical Mark-up Language (PML) and Object Naming Service (ONS).

Opposing to many standards and languages developed in specific application domains, PML was designed to provide broad definitions, describing those characteristics common to all physical objects. Furthermore, the need for a simple, reliable and effective framework for describing physical objects, processes and environments suggests avoiding use of complex and context-dependent standards. Many standards indeed are not adopted because of their inherent complexity in learning and implementation: this is the case, for example, of the Standard General Mark-up Language (SGML). Its derivative, the Hypertext Mark-up Language (HTML), has seen a wide spread growth, in part because of its simplicity and because of the tools and viewers available for the standard. The Extensible Mark-up Language (XML), also based on the Standard General Mark-up Language, has seen increasing growth as a tool for tagging data content.

The purpose of the core part of the PML is to provide a standardized format for the exchange of the data captured by the sensors in an Auto-ID infrastructure, e.g. RFID readers. PML core provides a set of schemas that define the interchange format for the transmission of the data captured.

Among the standardisation initiatives previously described, the most interesting ones are ISO 10303 and IEC 62264: they are universally well-accepted standards and they are able to model information regarding the product and its manufacturing. In fact, both these standards will be analyzed in this thesis, because they try to solve the problem of managing heterogeneous information coming from different systems: actually, their models are used to allow the exchange of information between an ERP, PDM (ISO 10303) and between an ERP and a MES (IEC 62264). Through the formalisation of the knowledge related to product data either at the business levels or at the manufacturing ones, they try to achieve the interoperability between systems: for this reasons they are very useful to our scope.

5. Conclusions

Product information of an enterprise is the most basic information which is referred to all systems and applications within an enterprise and to the other enterprises which collaborate between them in order to achieve common objectives. For the semantic interoperability of product information, a product ontology should be useful as a communication means between related enterprises which participate in the value chain.

Traditionally, product information is spread among several intra-organizational systems, especially ERP, PDM, and PLM systems, with many possibilities of data redundancies and inconsistencies.

As seen before, an ontology provides a conceptual framework for communicating in a given application domain. Consequently, ontologies for product data provide a framework for sharing a precise meaning of symbols exchanged during communication among the many enterprise and enterprise systems which demand accurate and reliable information of different granularity levels about products.

In all industrial organizations the available process and product knowledge must be maintained somehow. A company must register the products that it manufactures and the way they are produced, storage, sold and distributed. All this information is maintained in the so-called *Product Model*. A product model must represent, among other things, the way in which each product is manufactured by an industrial enterprise (Hegge, 1995).

Existing standardisation initiatives try to integrate enterprise product models by formalising the knowledge related to products technical data along its lifecycle, in order to answer the information interoperability problem.

Between them, the most interesting ones are IEC 10303 and ISO 62264: they are universally well-accepted standards and they are able to model information regarding the product and its manufacturing, covering the product phases in which are more characterizing the questions of interoperability problem.

Among the main standards discovered in literature, both these standards are been chosen in this thesis, because they, better than the other ones, put the basis to solve the problem of managing heterogeneous information coming from different systems. In fact, they try to formalise the knowledge related to product data management at the business and the manufacturing levels of enterprises (B2M, Business to Manufacturing), in order to achieve the interoperability between systems.

They can be considered a sort of Product Ontology, because they born by the agreement of a group of expert on the formalization of product information in order to be share by all enterprise application. Nevertheless, their approach is rather prescriptive, in sense that it forces users to translate information from generic concepts to more practical and ad-hoc ones. However, they cover different phases of product lifecycle (the Product Development phase and the Product Production one) and thus they are specific of a particular domain (the Engineering Domain and the Manufacturing one).

A Product Ontolgy that may be formed during the entire product life-cycle by the force of necessity of using it to communicate with the applications will be necessary to explicit and for this scope the product information standards above mentioned can represent a good stating point to build this ontology. As in analogy with the ontology community, in fact, it is possible to think that the integration of existing well-known and accepted models is more beneficial way to eliminate time, cost, and effort for building a new ontology.

6. References of the chapter

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. F. (2003). The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press.

Berners-Lee T., Hendler J., Lassila O..(2001), The Semantic Web, Scientific American, May.

Breitman K.K., Casanova M.A., Truszkowski W., (2007). Semantic Web: Concepts, Technologies and Applications. Springer Verlag.

Dartigues C. (2003). Product data exchange in a cooperative environment, Ph.D. dissertation, Univ. of Lyon 1, Lyon, France.

Dieter F.. (2000). Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. SpringerVerlag, Berlin.

Ding Y., Fensel D., Omelayenko B., Klein M.. (2002). The semantic web: yet another hip? DKE, 6(2-3):205–227.

Farquhar A., Fikes R., and Rice J., (1997). The Ontolingua Server: A Tool for Collaborative Ontology Construction, International Journal of Human-Computer Studies, 46:707-728.

Fensel D., Groenboom R. (1997). Specifying Knowledgebased Systems with Reusable Components. In Proceedings 9th International Conference on Software Engineering and Knowledge Engineering (SEKE ’97), Madrid.

Fridman-Noy N., Hafner C.D.. (1997). The State of the Art in Ontology Design, AI Magazine, 18(3):53-74.

Genesereth M., Fikes R. (1992) Knowledge Interchange Format (Tech. Rep. Logic-92-1). Stanford Univ., Stanford, CA. [Online]. Available: http://www-ksl.stanford.edu/knowledge-sharing/kif/#manual.

Genesereth M. R. (1991). Knowledge Interchange Format. In Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91), J. Allenet al., (eds), Morgan Kaufman Publishers, 238-249. See also http://logic.stanford.edu/kif/kif.html.

Grosso W. E., Eriksson H., Fergerson R. W., Gennari J. H., Tu S. W., Musen M. A. (1999). Knowledge Modeling at the Millennium (The Design and Evolution of Protégé-2000). In Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99), Banff, Alberta, Canada, October 16-21.

Gruber T.R. (1993). 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.), Kluwer Academic Publishers.

Guarino N., (1998). Formal ontology in information systems. In Proceedings of formal ontology in information systems (FOIS’98), 3–15,. Trento, Italy.

Hegge, H. (1995) Intelligent Product Descriptions for bussiness applications. Thesis Eindhoven. Eindhoven University of Technology.

IEC 62264 (2002). Enterprise-control system integration, Part 1. Models and terminology, Part 2: Model object attributes. ISO/IEC, 2002, Geneva.

ISO 10303 (STEP) task website, stepmod.sourceforge.net

Jasper R., Uschold, M. (1999). A framework for understanding and classifying ontology applications. In Proceedings of the IJCAI99 workshops on ontologies and problem-solving methods. Sweden: Stockholm.

Jeongsoo Lee J., Chae H., Kim C.H., Kim K.. (2009). Design of product ontology architecture for collaborative enterprises, Expert Systems with Applications 36, 2300–2309.

Jun H. B., Kiritsis D.. (2007). Research issues on closed-loop PLM, Computers in Industry 58, 866-868.

Kalfoglou Y., Schorlemmer M. (2002). Information-flow-based ontology mapping. Lecture Notes in Computer Science, 2519, 1132–1151.

Kifer M., Lausen G., Wu J., (1995). Logical Foundations of Object-Oriented and Frame-Based Languages, Journal of the ACM, 42.

Kim K. Y., Chae S. H., Suh H. W.. (2003). An approach to semantic mapping using product ontology for CPC environment, in Proceedings of the 10th ISPE International Conference on Concurrent Engineering Research and Applications, J. Cha, R. Jardim-Gonçalves, and A. Steiger- Garção, Eds. Madeira Island, Portugal: A. A. Balkema, Jul., 291–298.

Kiritsis D., Bufardi A., Xirouchakis P., (2003). Research issues on product lifecycle management and information tracking using smart embedded systems, Advanced Engineering Informatics 17, 189–202.

Lee J., Chae H., Kim K., Kim C. (2006). An ontology architecture for integration of ontologies. Lecture Notes in Computer Science, 4185, 205–211.

McFarlane D., Sarma S., Chirn J.L., Wong C.Y., Ashton K., (2003), Auto ID System and Intelligent Manufacturing Control, Engineering Application of Artificial Intelligence, 16, pp 365-376.

McGuinness D. L., Fikes R., Rice J., Wilder S.:, (2000). The Chimaera Ontology Environment. In the Proceedings of the The SeventeenthNational Conference on Artificial Intelligence (AAAI 2000), Austin, Texas, July 30 - August 3.

Mizoguchi R., Vanwelkenhuysen J., Ikeda M.. (1995). Task ontology for reuse of problem solving knowledge. In N. J. I. Mars, editor, Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing., pages 46–57. IOS Press, Amsterdam, NL.

Noy N., Musen M., (2000) PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of the Seventeenth

National Conference on Artificial Intelligence (AAAI-2000), Austin, Texas, July 30 - August 3.

Noy N., Hafner, C. (1997). The state of the art in ontology design – A survey and comparative review. AI Magazine, 36(3), 53–74.

Patil L., Dutta D., Sriram R. (2005), Ontology-based exchange of product data semantics, In: IEEE Transactions of Automation Science and Engineering; 2, 213–225.

Pirlein Th., Studer R., (1997). Integrating the Reuse of Commonsense Ontologies and Problem-Solving Methods, University of Karlsruhe, Institute AIFB, Research Report 354, May 1997.

PLCS, www.plcs.org

PLM XML, www.ugsplm.com

Studer R., Eriksson H., Gennari J. H., Tu S. W., Fensel D., and Musen M.. (1996). Ontologies and the Configuration of Problem-Solving Methods. In B. R. Gaines and M. A. Musen, (eds.), Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada.

Terzi S. (2005). Elements of Product Lifecycle Management: Definitions, Open Issues and Reference Models. PhD - University Henri Poincaré Nancy I and Politecnico di Milano, May

Uschold M., King, M. (1995). Towards a methodology for building ontologies. In Proceedings of IJCAI95 workshop on basic ontological issues in knowledge sharing. Montreal, Canada.

Van Heijst G., Schreiber A., and Wielinga B., (1997). Using Explicit Ontologies in KBS development, International Journal of Human Computer Studies, 46:183-292.

Vegetti M., Henning G.P., Leone H.P..(2005) PRoduct ONTOlogy. An ontology for complex product modelling domain, in: Proceedings of the ENPROMER, Rio de Janeiro.

Vegetti M.; Henning G. P.; Leone H. P., (2005). Product ontology: definition of an ontology for the complex product modelling domain. In: Mercosur Congress On Process Systems Engineering, 4.

Weibel S., Gridby J., Miller E., (1995). OCLC/NCSA Metadata Workshop Report, Dublin, EUA,. http://www.oclc.org:5046/oclc/research/conferences/metadata/dublin_core_report.html.

www.pabadis-promise.org

Yoo S.B., Kim Y..(2002). Web-based knowledge management for sharing product data in virtual enterprises, International Journal of Production Economics 75 173–183.

Chapter 3: Proposal of a ontological model for product-centric information systems interoperability

1. Introduction

This chapter deals with the proposal of an ontological model useful for product-centric information systems interoperability.

As point out in the precedent chapters of this thesis, we consider the standard models as a good starting point for the building of a common information model, to support information exchange between the product views and the many applications that interact with them. This model intends to specify an embedded *Product Ontology* that may be formed during the product life-cycle by the force of necessity of using it to communicate with the applications.

The standard chosen for this scope are the ISO 10303, and in particular STEP PDM, and IEC 62264, which have been presented in the chapter 2. In fact, the STEP PDM Schema deals with typical product-related information including geometry, engineering drawings, project plans, part files, assembly diagrams, product specifications, numerical control machine-tool programs, analysis results, correspondence, bills of material, engineering change orders, and many more. IEC 62264, instead, specify a set of reference models for information exchange between business applications and manufacturing control applications.

In order to use the product standard models as knowledge base for our ontology, first of all, it is necessary to analyse them, in order to deeply understand their semantics, in relation to technical product data. The chapter will follow describing the proposed methodology and the tools taken into account. Finally, the mapping between those concepts and its

formulation will be formalised and part of final ontological model will be shown.

2. Standard models analysis

ISO 10303 – STEP PDM

STEP is based on a modular and re-configurability structure, which uses Application Protocols (APs) to specify the representation of product information for one or more applications. Application Protocols are sub-sets of STEP, focused on specific issues or specific industrial sectors, which break the entire STEP standard into easily manageable views of quick implementation. STEP initiative adopts a strategy of specification into industrial context (e.g. APs for product design, for mechanical and electrical engineering, for sheet metal manufacturing, for product assembly, for automotive industry). Each AP is applicable to one or more life cycle stages of a particular product class. STEP methodology is currently migrating to a practice in which an AP is built from a collection of *Application Modules* (AMs). Such a module is a shareable set of closely related definitions that may be used by several different APs. This ensures that, in cases where there is technical overlap between the capabilities of those APs there is no inconsistency in the way that the information is represented. However, the AMs themselves are constructed on the basis of a set of Integrated Resources (IRs), defining fundamental constructs that can be specialised and applied for a wide variety of purposes.

We focus on STEP PDM (Product Data Management) Schema, which is a reference information model for the exchange of a central, common subset of the data being managed within a PDM system. It represents the intersection of requirements and data structures from a range of STEP Application Protocols, all generally within the domains of design and development of discrete electro/mechanical parts and assemblies.

The STEP PDM Schema is *not* a specification for the functionality required for the complete scope of all PDM system functionality – i.e., it is *not* the union, but the intersection, of functionality present in the set of STEP

Application Protocols. There exists functionality important for complete PDM functionality that is not represented in the PDM Schema, but is in other units of functionality present in STEP APs.

STEP uses the EXPRESS language for describing data type, constraints on data type and relationship between data type. EXPRESS is a modelling language combining ideas from the entity-attribute-relationship family of modelling languages with object modelling concepts.

Application Protocols are required to contain a representation of the information in both EXPRESS and EXPRESS-G. EXPRESS-G is a diagramming technique supporting a subset of the EXPRESS language.

We can provide an example of semantics of product data within STEP PDM, considering the *bill of material*. The *bill of material (BOM)* is one of the crucial product technical data in the production management domain as well as in the information technology that supports it (Xu et al, 2008): the BOM represents the base issue of integrating product design system with production planning system. The STEP PDM Schema supports hierarchical product structures representing assemblies and the constituents of those assemblies: this product structure corresponds to the traditional engineering and manufacturing bill of material indentured parts list.

The *Assembly_component_relationship* class represents the general relationship between two parts, one a definition of a component and the other a definition of the parent assembly. This entity is typically instantiated as the subtype *Next_assembly_usage*, which represents an unique individual occurrence of the component as used within the parent assembly. The subtype *Promissory_usage*, instead, represents the usage occurrence of a component within a higher-level assembly that is not the immediate parent. The subtype *Component_upper_level_identification* identifies a component of an assembly with respect to an upper level in the assembly structure (Tursi et al., 2009).

In the figures below , it is possible to see as this concepts are explained in STEP PDM in UML (Figure 16), in EXPRESS-G (Figure 17), and in EXPRESS (Figure 18).

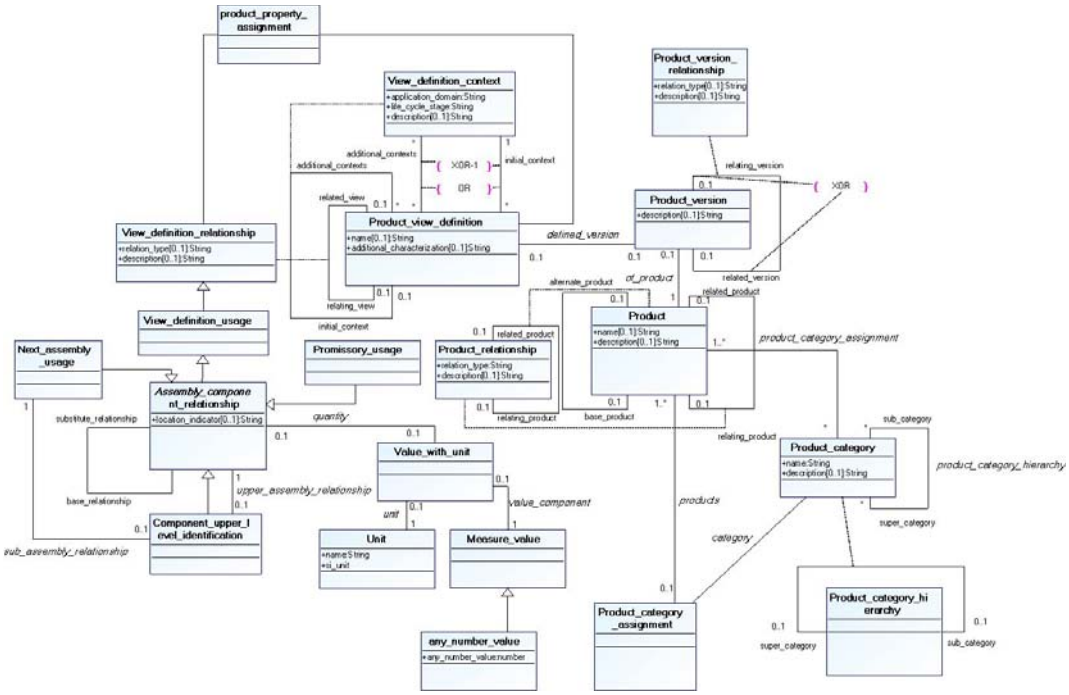


Figure 16 - Assembly structure module in UML (see annex III)

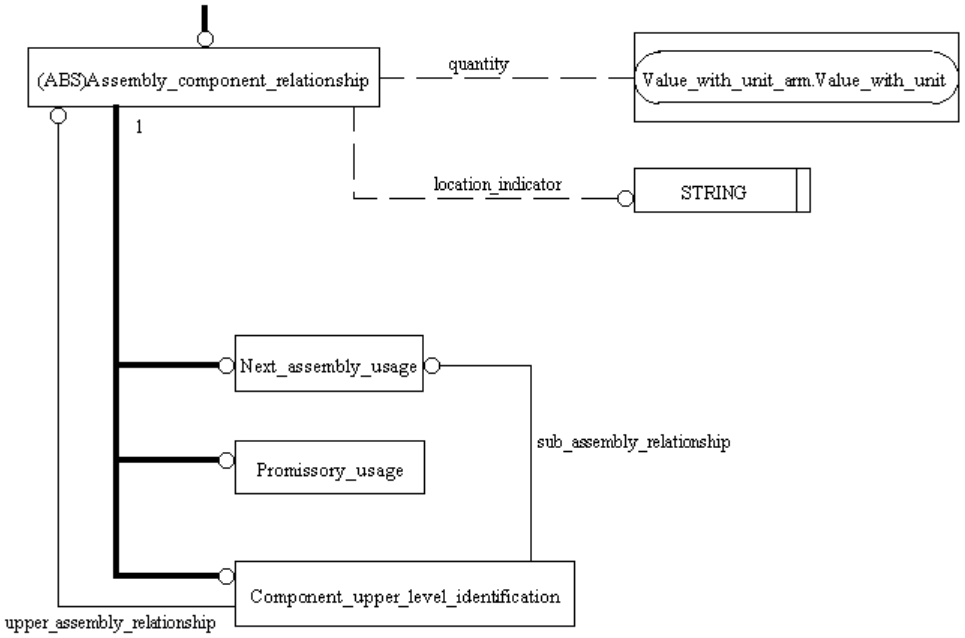


Figure 17 – Assembly structure in EXPRESS-G (see annex II)

```
*)
ENTITY Assembly_component_relationship
 ABSTRACT SUPERTYPE OF (ONEOF (Next_assembly_usage,
                Promissory_usage,
                Component_upper_level_identification))
 SUBTYPE OF (View_definition_usage);
 quantity : OPTIONAL Value_with_unit;
 location_indicator : OPTIONAL STRING;
WHERE
 WR1: NOT(EXISTS(quantity)) OR ((NOT ('NUMBER' IN
TYPEOF(quantity.value_component))) XOR
(quantity.value_component > 0));
END_ENTITY;
(*
```

Figure 18 - Assembly structure in EXPRESS

The *Assembly_component_relationship* is established between two instances of *Product_view_definition*: the relating view of *Product_version* of assembly and the related one of the Product_version which plays the role of component. A *Product_view_definition* is a collector of the properties that characterize the *Product_version* in the *initial_context* and possibly *additional_contexts*. A *Product_version* is a revision or a collector of the definitions of the revision of *Product.*

IEC 62264

The standard concern the information related to the interface between plant production scheduling and operation management and plant floor coordination. To take into account the various exchanged information, through the product representation, the standard defines a set of eight models that specifies all concepts for enterprise-control integration (Dassisti et al., 2008).

Product Definition Model: the product definition model is information shared between production rules, bill of material, and bill of resources. A product definition contains a listing of the exchanged information about a product. The information is used in a set of product segments that are the

values needed to quantify a segment for a specific product. A product segment identifies, references, or corresponds to a process segment. It is related to a specific product, while a process segment is product independent. The collection of product segments for a product gives the sequence and ordering of segments required to manufacture a product in sufficient detail for production planning and scheduling. The corresponding production rule presents the additional detail required for actual production.

Material Model: the material model defines the actual materials, material definitions, and information about classes of material definitions. Material information includes the inventory of raw, finished, and intermediate materials. Material classes are defined to organise materials. A Material definition is a means to describe goods with similar characteristics for purposes of scheduling and planning.

Equipment Model: the equipment model contains the information about specific equipment, the classes of equipment, equipment capability tests, and maintenance information associated with equipment.

Personnel Model: the personnel model contains the information about specific personnel, classes of personnel, and qualifications of personnel.

Process Segment Model: the process segment model contains process segments that list the classes of personnel, equipment, and material needed, and/or it may present specific resources, such as specific equipment needed. A process segment may list the quantity of the resource needed. A process segment is related to a product segment that can occur during production, as presented in the product definition model.

Production Schedule Model: a request for production shall be listed as a production schedule. A production schedule shall be made up of one or

more production requests. A request for production for a single product identified by a production rule shall be shown as a production request. A production request contains the information required by manufacturing to fulfil scheduled production. This may be a subset of the business production order information, or it may contain additional information not normally used by the business system. A production request may identify or reference the associated production rule. A production request shall contain at least one segment requirement, even if it spans all production of the product.

Production Capability Model: the production capability information is the collection of information about all resources for production for selected times. This is made up of information about equipment, material, personnel, and process segments. It describes the names, terms, statuses, and quantities of which the manufacturing control system has knowledge. The production capability information contains the vocabulary for capacity scheduling and maintenance information.

Production Performance Model: the performance of the requested manufacturing requests shall be listed as production performance. Production performance shall be a collection of production responses. The responses from manufacturing that are associated with a production request shall be used as production responses. There may be one or more production responses for a single production request if the production facility needs to split the production request into smaller elements of work. A production result may include the status of the request, such as the percentage complete, a finished status, or an aborted status.

IEC 62264 makes use of UML representation for displaying each “class” of information and its relations with other classes. Figure 19 depicts a UML class diagram representing a conceptualisation of Material Model classes.

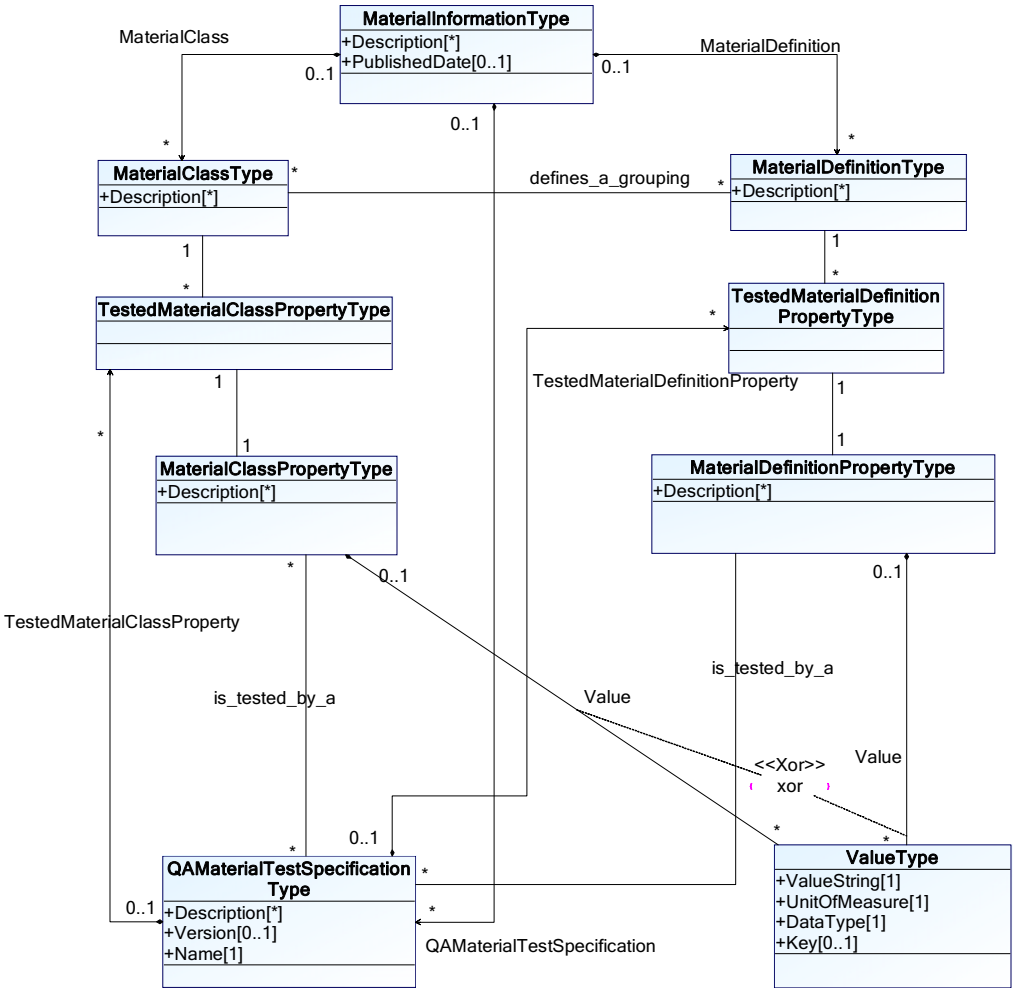


Figure 19 (a) – Conceptualised Material Model (Dassisti et al., 2008)

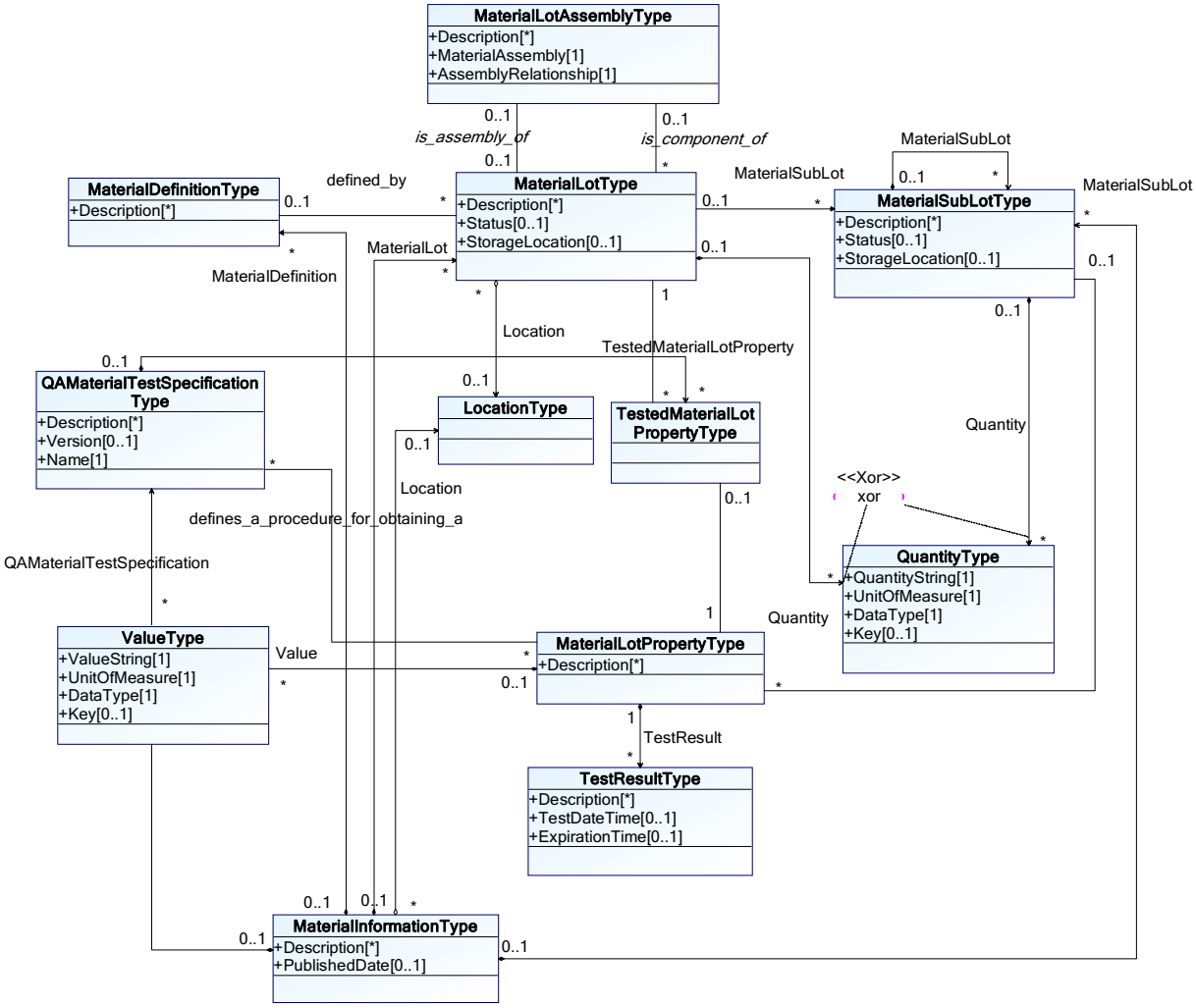


Figure 19 (b) – Conceptualised Material Model (Dassisti et al., 2008)

3. The methodology

3.1. The scenario

An illustrative scenario was defined to which to refer in order to provide a more familiar example of real production system. The proposed case study concerns the design and the production of a product. It is based on a set of enterprise systems, distributed on two production sites, one in Italy and one in France. The product is conceived and designed in the Department of Mechanical and Management Engineering of the Politecnico di Bari, in Bari, Italy. Technical and geometrical information,

joined to business information, are structured in the information model that implements the *Product Ontology*.

This digital product is sent to the *Atelier Inter-Établissements de Productique Lorrain* (AIPL-PRIMECA) of the Nancy-University, France.

AIPL has to manufacture the product on the base of information drawn from the *Product Ontology*.

At the end of production process, the manufactured product will be sent to DIMeG in order to be delivered to the customer.

Each enterprise of this case study is equipped with its enterprise systems (i.e. Windchill PDM and SAP R/3 for DIMeG or DS SmarTeam and Sage X3 for AIPL), dedicated to specific tasks (engineering tasks or manufacturing ones) and provided by a particular vendor. In this product-centric information system, these heterogeneous applications have to interoperate with the product, in order to store and to draw the pertinent product information (Tursi et al., 2009).

Step 1: Models transformation

In order to verify that the same information is modelled in different way by the two standards, we have de-normalised and conceptualised the PDM STEP Schema and IEC 62264 models and represented them using the UML class diagram notation. In this way, it is possible to have a common minimum semantics denominator which allows the matching and the mapping between the two standards. While IEC 62264 makes use of UML language for its models, STEP PDM is initially expressed in EXPRESS and then the two standards are not at the same abstraction level to be matched and mapped (Tursi et al., 2007). Different mapping methods have been proposed to implement the system integration within the product modelling area. A formal mapping notation is required for the definition of mappings on the conceptual level (Han and Suh, 2001). This notation provides a method to describe the correspondences between models.

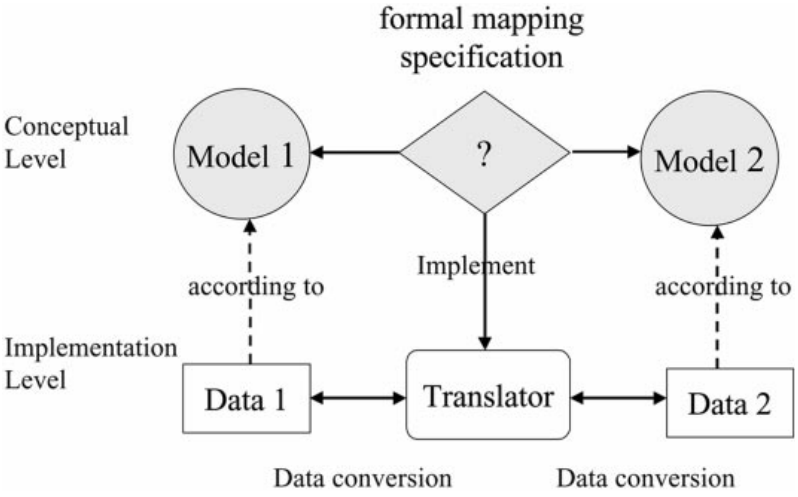


Figure 20 – A general mapping problem (Han and Suh, 2001)

Figure 20 shows the architecture of a general mapping problem (Liebich et al., 1995). There are two types of mapping problems. One is the transformation performed on two schemas specified by the same EXPRESS modelling language. EXPRESS-M, EXPRESS-V (ISO/TC184/SC4, 1992), and EXPRESS-X (ISO/TC184/SC4/WG11 N088, 1999) are technical solutions for this problem. The other one is the transformation performed on two schemas specified by different languages. This is a mapping problem between heterogeneous systems. The requirements for an EXPRESS mapping language such as human and computer interpretability, similarity to EXPRESS, formal specification, ARM to AIM mapping, and mapping between APs are specified by ISO (ISO TC/184/SC4/WG11 N013, 1997). In addition, there are more requirements for a general mapping language to map a legacy system to a STEP model. It should define the mapping between the heterogeneous models and support a graphical notation.

The UML is a visual modelling language for specifying, visualizing, and constructing software systems. It unifies the object-oriented methods of Booch, Rumbaugh, and Jacobson (Fowler, 1997). It can be easily applied to the development of a data translator, because it is made for the development of software systems. However, there is no mapping notation

in UML. The usage of UML diagrams for the STEP standards is also being discussed in ISO/TC184/SC4.

Using UML to formalize the concepts and axioms of IEC 62264 and ISO 10303 STEP-PDM can be done manually or semi-automatically.

In order to manually translate EXPRESS models into UML ones, some steps must be followed during its design. It is significant that we must firstly make a list of elements of the domain and then distinguish which will become these in the other language, by trying correspondence between elements of two languages. We use the Mega Suite[1] to develop UML class diagrams of IEC 62264 and ISO 10303 STEP-PDM.

The ISO STEP committee (TC184/SC4) is developing *ISO 10303-25, EXPRESS to OMG XMI binding* (Shin and Han, 1988; STEPTools, 1995) (also known as Part 25), for transforming EXPRESS schemas into UML models. This will enable developers to use their familiar UML tools to see the contents of STEP (EXPRESS) schemas and eventually to specify relationships between STEP information models and the other UML models that they use. A Part 25 mapping from our EXPRESS schema to the XML Metadata Interchange (XMI®) format (STEP PDM Schema, 1998) would produce a UML class diagram.

The current version of Part 25 (ISO/CD TS 10303-25) includes a mapping from EXPRESS to XMI that covers most of the basic EXPRESS concepts, with the exception of several of its more technical features that are commonly used to implement constraints (such as RULE, PROCEDURE, and FUNCTION declarations and UNIQUE rules). The mapping is also one-way only (i.e., from EXPRESS to XMI, but not yet from XMI to EXPRESS).

Concerning automatic transformation, until recently, there were few software tools for using STEP schemas and instance populations in the XML and UML worlds. There are now several promising development efforts underway to create such software that capitalizes on the popularity

[1] http://www.mega.com

of XML and UML, however one must develop, manually, the related rules to implement constraints.

The EXPRESS For Free (*exff*) project (http://exff.sourceforge.net) is developing tools to convert between EXPRESS and UML. The initial goal is to be able to employ UML-based code generation tools to help implement STEP. Future plans include supporting the use of UML modelling tools to build EXPRESS schemas. *exff* provides translators between XMI and EXPRESS marked up in XML using the STEP Module Repository Document Type Definition.

UNINOVA developed the STEP25 tool that translates EXPRESS-based models to XMI following the emerging ISO10303 part 25 directives. Using this tool, a mapping for two subsets of the model was implemented and validated grounded on the industrial scenarios. The respective XMI documents, were successfully imported in UML enabled application (i.e., Mega Suite), and the model was considered compliant with the specifications established in the part 25 of the ISO10303. This tool is the first that we know of that implements and proves this concept for EXPRESS to XMI binding, validating an ISO10303 application reference model.

The commercial Mega suite platform has then been used to import the ARM model, described in XMI, into UML, for obtaining an implementation model represented by a UML class diagram. However, this model is not at a semantic level and needs then to be conceptualised in order to keep only concepts and constructs representing domain knowledge.

Step 2: Models Formalization

The following step has been the formalization of UML standards models in First Order Logic language, which will be able to express the sharing knowledge of the standard (Tursi et al., 2007; Tursi et al., 2009).

According to Klein (Klein, 2001), there are two levels of mismatches between models. The first level is the *language* or *meta-model* level. Mismatches at this level are mismatches between the *mechanism* to

define classes, relations and so on. The languages can differ in their syntax, but, more important, constructs available in one language (e.g., stating that classes are disjoint) are not available in another. Even semantics of the same language primitives could be different (e.g., whether declarations of multiple ranges of a property have union or intersection semantics). The *normalization* process therefore often precedes models-matching (Kalfoglou and Schorlemmer, 2003) and translates source models to the same language, resolving these differences: this is what we have developed in the step 1. The second level is the *ontology* or *model* level. A mismatch at this level is a difference in the way the domain is modelled. A partial list of mismatches includes using the same linguistic terms to describe different concepts; using different terms to describe the same concept; using different modelling paradigms (e.g., using interval logic or points for temporal representation); using different modelling conventions and levels of granularity; having models with differing coverage of the domain, and so on.

The distinction between these two levels of differences is made very often. Kitakami *et al.* (Kitakami et al., 1996) and Visser *et al.* (Visser et al., 1997) call these kinds of differences respectively *non-semantic* and *semantic* differences

Step 3: Syntactical analysis

In order to demonstrate that the models describe the same information, our approach is based, firstly, on a syntactical analysis whose aim is to compare the instances defined in both models and then based on semantics analysis, studying properties of the shared objects. An example of the result of the syntactical analysis is presented in the Table 2.

AIPL objects	STEP PDM modules concepts	IEC 62264 concepts
Pxx	Pxx: Product	Pxx: MaterialClassType
P09	P09: Product	P09: MaterialDefinitionType P09_Lot: MaterialLotType
Diameter	Diameter: Independent_property	Diameter: MaterialClassPropertyType
Diameter_value_with_unit	Diameter_value_with_unit: Numerical_item_with_unit	Diameter_value_with_unit: MaterialLotPropertyType

Table 2 – Syntactical analysis of concepts (Tursi et al., 2007)

In order to build a knowledge representation of product information, in fact, two mechanisms are been followed: (1) *syntactical analysis* via instance-directed rewrite rules that allow the concise specification of concept-level transformations based on instance matching, and (2) *semantic analysis* which modulates syntactic writing via logical inference (Klein, 2001).

Step 4: Semantic analysis

The semantic analysis suggests the possibility to do a mapping between the instantiated concepts (Tursi et al. 2007, Tursi et al., 2009).

Many researchers agree that one of the major bottlenecks in semantic integration is mapping discovery. There are simply too many ontologies and database schemas available and they are too large to have manual definition of correspondences as the primary source of mapping discovery.

Hence, the task of finding mappings (semi-) automatically has been an active area of research in both database and ontology communities (Ramh and Bernstein, 2001; Kalfoglou and Schorlemmer, 2003).

In the ontology domain, to which we refer to, Noy (Noy, 2004) identifies two major architectures for mapping discovery between ontologies. For the first approach, we can recall that the goal of ontologies is to facilitate knowledge sharing. Here, the vision is that a general upper ontology is agreed upon by developers of different applications, who then extend this

general ontology with concepts and properties specific to their applications. As long as this extension is performed in a way consistent with the definitions in the shared ontology, finding correspondences between two extensions can be facilitated by this common “grounding”. The second set of approaches comprises *heuristics-based or machine learning techniques* that use various characteristics of ontologies, such as their structure, definitions of concepts, and instances of classes, to find mappings. These approaches are similar to approaches to mapping XML schemas or other structured data but tend to rely more heavily on features of concept definitions or on explicit semantics of these definitions.

Hovy (Hovy, 1998) describes a set of heuristics that researchers at ISI/USC used for semi-automatic alignment of domain ontologies to a large central ontology. Their techniques are based mainly on linguistic analysis of concept names and natural-language definitions of concepts.

The PROMPT system (Noy and Musen, 2003) uses a mixture of lexical and structural features, as well as input from the user during an interactive merging session to find the mappings. For instance, if a user said that two classes in two source ontologies are the same (should be merged), then PROMPT analyzed the properties of these classes, their subclasses and superclasses to look for similarities of their definitions and suggest additional correspondences.

Euzenat and Valtchev (Euzenat and Valtchev, 2004) developed a methodology based on a weighted combination of similarities of various features in OWL concept definitions: their labels, domains and ranges of properties, restrictions on properties (such as cardinality restrictions), types of concepts, subclasses and superclasses, and so on.

FCA-Merge (Stumme and Madche, 2001) is a method for comparing ontologies that have a set of shared instances or a shared set of documents annotated with concepts from source ontologies. Based on this information, FCA-Merge uses techniques from Formal Concept Analysis (Ganter and Wille, 1999) to produce a lattice of concepts which relates concepts from the source ontologies. The algorithm suggests equivalence

and subclass–superclass relations. An ontology engineer can then analyze the result and uses it as a guide for creating a merged ontology.

The IF-Map (Kalfoglou and Schorlemmer, 2003) system identifies mappings automatically based on the theory of information flow (Liebich et al.,1995). Given two ontologies, IF-Map generates a *logic infomorphism*— a mapping between ontologies that is based on the above conformance.

GLUE (Doan et al., 2002) is an example of a system that employs machine-learning techniques to find mappings. GLUE uses multiple learners exploiting information in concept instances and taxonomic structure of ontologies. GLUE uses a probabilistic model to combine results of different learners. To summarize, the tools for automatic and semi-automatic ontology alignment use the following features in ontology definitions (to various extent):

- concept names and natural-language descriptions
- class hierarchy (subclass–superclass relationships)
- property definitions (domains, ranges, restrictions)
- instances of classes
- class descriptions (as in DL-based tools).

All these techniques and methods are generally used when the two ontologies are defined in natural-language descriptions which are at the conceptual level. In our case, we are using a mix between class descriptions and instance of classes, with the aid of First Order Logic (FOL) formalization of the concepts, which allow expressing the knowledge and semantics contained in standard ontologies.

Step 5: Ontological model

The mapping rules allow to build a final model, given merging the two ontologies of standard, formalized in FOL. A detailed analysis of FOL, which describes syntax and semantics, is reported in the annex. This FOL formalization allows a verification of the coherence of the produced ontology using inference engines while a test case (chapter 4) can allow the validation the proposed model.

4. Mapping formalization

4.1. FOL formalisation of UML conceptual models

UML class diagrams allow for modelling, in a declarative way, the static structure of an application domain, in terms of concepts and relations between them. In the annex III, the main concepts of UML class diagram are represented.

We briefly describe UML class diagrams, and specify the semantics of the main constructs in terms of first-order logic (FOL) (Berardi et al. 2005; Calvanese et al., 1998). The semantics of each construct of UML class diagram will be formalized in FOL axioms, that we will call “patterns of formalization”.

Class

A class in a UML class diagram denotes a set of objects with common features. Names of classes are unique in UML Namespace. Formally, a *class* C corresponds to a FOL *unary predicate* C. Classes may have attributes and operations. For our scope, the operations of a class won’t be considered.

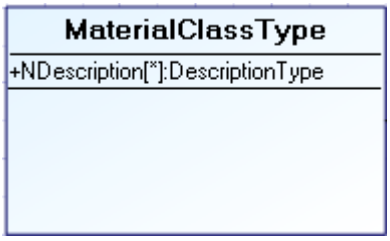


Figure 21 – A class of Material Model in IEC 62264

An *attribute a* of type *T* for a class *C* associates to each instance of *C* a set of instances of *T*. Attributes are unique within a class, but two classes may have two attributes with the same name, possibly of different types. An optional multiplicity [*i..j*] for *a* specifies that *a* associates to each instance of *C* at least *i* and most *j* instances of *T* . When there is no upper bound on the multiplicity, the symbol $*$ is used for *j* . When the multiplicity is missing, [1..1] is assumed, i.e., the attribute is *mandatory* and *single-valued*. For example, the attribute Description[*]: String in Figure 21

means that each instance of the class could have one, more o no one descriptions, and that each MaterialClass description is an instance of String (DescriptionType is an application datatype referring to the standard String type).

Formally, an attribute *a* of type *T* for class *C* corresponds to a binary predicate *a* for which the following FOL assertion holds:

$\forall x, y.\ (C(x) \wedge a(x, y)) \supset T(y)$	***1° pattern of formalization***

i.e., for each instance *x* of class *C*, an object *y* related to *x* by *a* is an instance of *T*.

In our case, for example:

$\forall$ x, y. (MaterialClassType(x) $\wedge$ Description(x,y)) $\supset$ String(y)

$\forall$ x. MaterialClassType(x) $\supset$ (0 $\leq$ # { y | Description(x,y) })

Associations and aggregations

An *association* in UML is a relation between the instances of two or more classes. Names of associations are unique in UML Namespace.

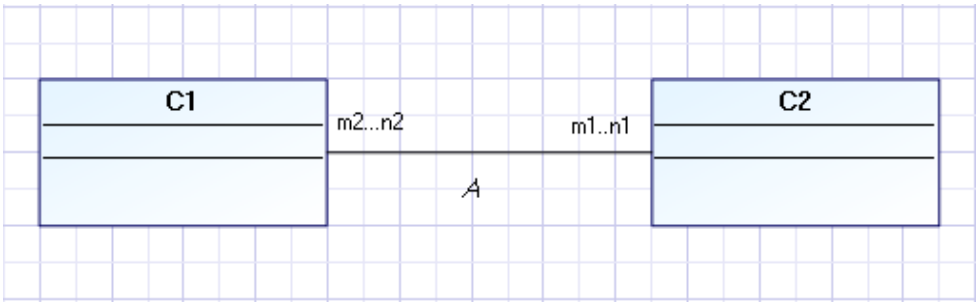


Figure 22 – Binary association in UML

The multiplicity $m_1..n_1$ on the binary association specifies that each instance of the class C_1 can participate at least m_1 times and at most n_1 times to A, similarly for C_2. When the multiplicity is omitted, it is intended to be 0..*.

An association A between the instances of classes C_1, . . . ,C_n, can be formalized as an n-ary predicate A that satisfies the following FOL assertion:

$\forall x_1, \ldots, x_n.\ A(x_1, \ldots, x_n) \supset C_1(x_1) \wedge \ldots \wedge C_n(x_n)$	***2° pattern of formalization***

For binary associations, multiplicities are formalized by the FOL assertions:

$\forall x.\ C_1(x) \supset (m_1 \leq \#\{y \mid A(x, y)\} \supset n_1)$	***3° pattern of formalization***
$\forall x.\ C_2(x) \supset (m_2 \leq \#\{y \mid A(x, y)\} \supset n_2)$	

where we have abbreviated FOL formulas expressing cardinality restrictions.

In semantic formalization the n-ary associations are not used because they are not expressing a simple semantics and may always be transformed into two or more binary associations.

In our case, for example:

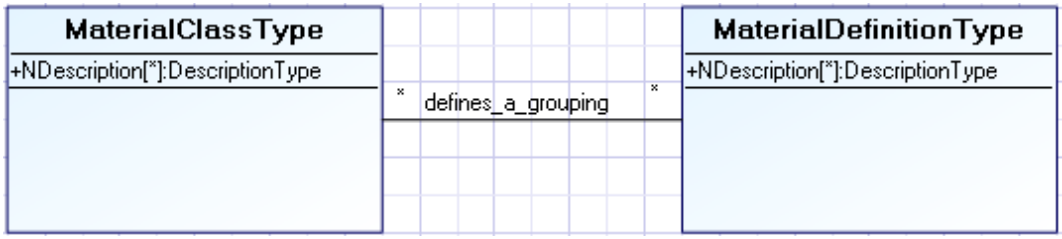


Figure 23 – An association between classes in Material Model in IEC 62264

$\forall\ x_1, x_2.$ defines_a_grouping $(x_1, x_2) \supset$ MaterialClassType$(x_1) \wedge$ MaterialDefinitionType(x_2)

$\forall$ x. MaterialClassType(x) $\supset (0 \leq \#\ \{\ y \mid$ defines_a_grouping $(x,y)\ \}$

Often, an association has a related *association class* that describes properties of the association, such as attributes, operations, etc. A binary association *A* between two classes C_1 and C_2 with an association class is graphically rendered as in Figure 24, where the class *A* is the association class related to the association, and r_1 and r_2 are the *role names* of C_1 and

C_2 respectively, which specify the role that each class plays within the association *A*.

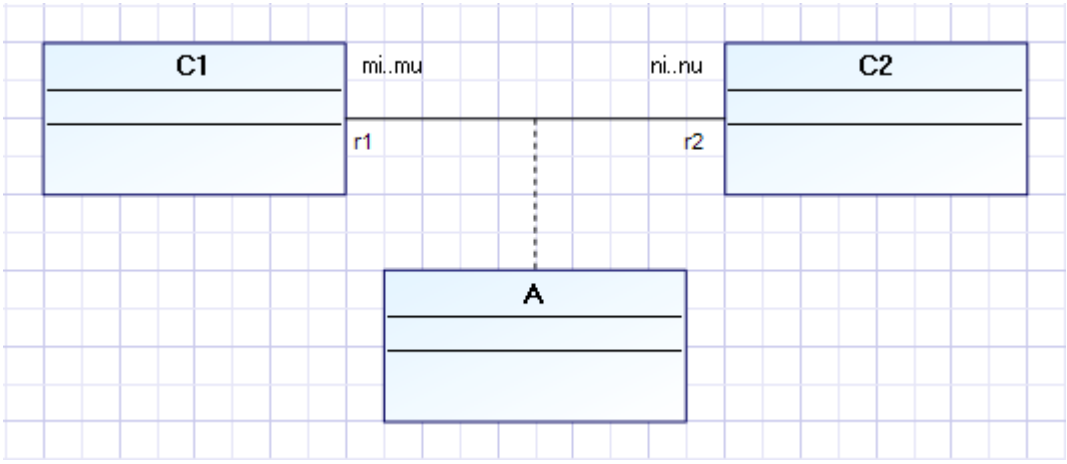


Figure 24 – Binary association with association class in UML

An association *A* between *n* classes $C_1, \ldots, C_n$ that has a related association class is represented by a unary predicate *A* and *n* binary predicates $r_1, \ldots, r_n$, one for each role name, for which the following FOL assertions hold:

$\forall x, y.\ A(x) \wedge ri(x, y) \supset Ci(y)$, for i = 1, . . . , n	***4° pattern of formalization***
$\forall x.\ A(x) \supset \exists y.\ ri(x, y)$, for i = 1, . . . , n	
$\forall x, y, y'.\ A(x) \wedge ri(x, y) \wedge ri(x, y') \supset y = y'$, for i = 1, . . . , n	
$\forall y_1, \ldots, y_n, x, x'.\ A(x) \wedge A(x') \wedge \Lambda_{i=1\ldots n} (r_i(x, y_i) \wedge r_i(x', y_i)) \supset x = x'$	

In our case, for example:

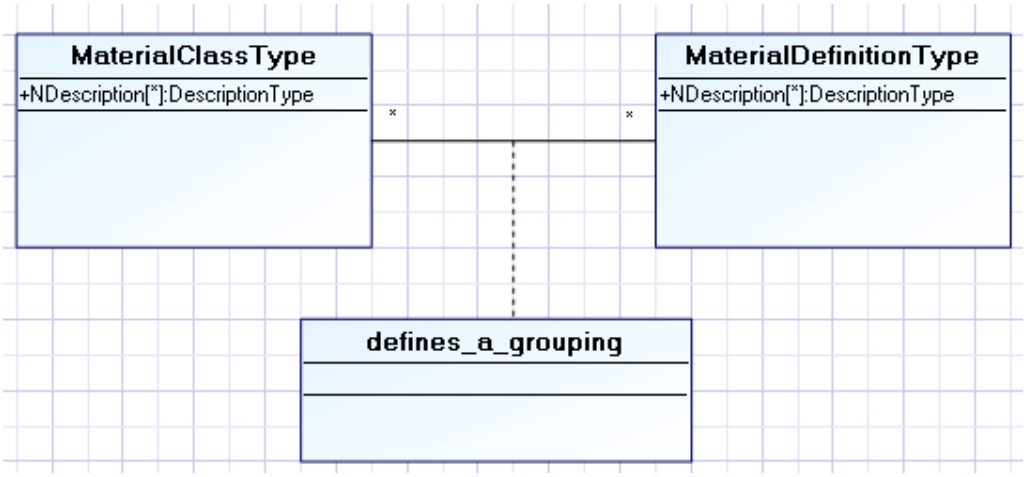


Figure 25 – An association class in Material Model in IEC 62264

$\forall$ x, y. defines_a_grouping(x) $\wedge$ r_1(x,y) $\supset$ MaterialClassType(x)

$\forall$ x, y. defines_a_grouping(x) $\wedge$ r_2(x,y) $\supset$ MaterialDefinitionType(x)

$\forall$ x. defines_a_grouping(x) $\supset$ $\exists$ y. r_1(x,y)

$\forall$ x. defines_a_grouping(x) $\supset$ $\exists$ y. r_2(x,y)

$\forall$ x, y, y'. defines_a_grouping(x) $\wedge$ r_1(x,y) $\wedge$ r_1(x, y') $\supset$ y = y'

$\forall$ x, y, y'. defines_a_grouping(x) $\wedge$ r_2(x,y) $\wedge$ r_2(x, y') $\supset$ y = y'

$\forall$ x, x', y_1, y_2. defines_a_grouping(x) $\wedge$ defines_a_grouping(x') $\wedge$ r_1(x, y_1) $\wedge$ r1(x', y_1)) $\wedge$ r_2(x, y_2) $\wedge$ r_2(x', y_2)) $\supset$ x= x'

A particular kind of binary associations are *aggregations*, which play an important role in UML class diagrams. An aggregation is a binary relation between the instances of two classes, denoting a part-whole relationship, i.e., a relationship that specifies that each instance of a class (the *containing class*) contains a set of instances of another class (the *contained class*). An aggregation is graphically rendered as shown in Figure 26, where the diamond indicates the containing class.

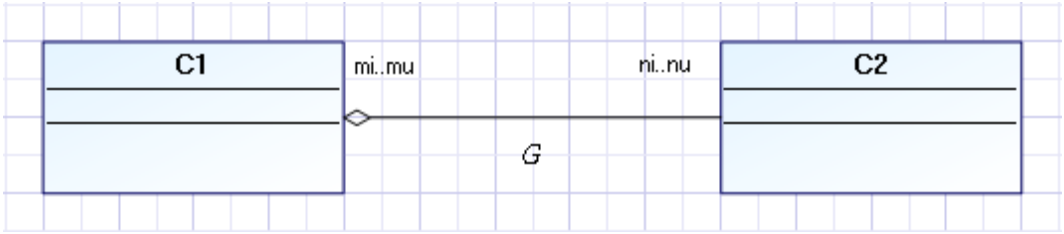


Figure 26 – Aggregation in UML

The aggregation of Figure 26 is represented by a binary predicate *G* for which the following FOL assertion holds:

$\forall x, y.\ G(x, y) \supset C_1(x) \wedge C_2(y)$	***5° pattern of formalization***

where we use the convention that the first argument of the predicate is the containing class.

Multiplicities are treated as for binary associations.

Generalization and hierarchies

In UML one can use a *generalization* between a parent class and a child class to specify that each instance of the child class is also an instance of the parent class. Hence, the instances of the child class inherit the properties of the parent class, but typically they satisfy additional properties that in general do not hold for the parent class. Several generalizations can be grouped together to form a class hierarchy (also called ISA hierarchy). Disjointness and completeness constraints can also be enforced on a class hierarchy (graphically, by adding suitable labels). A class hierarchy is said to be *disjoint* if no instance can belong to more than one derived class, and *complete* if any instance of the base class belongs also to some of the derived classes.

A UML class C generalizing a class C_1 can be formally captured by means of the FOL assertion:

$\forall x.\ C_1(x) \supset C(x)$	***6° pattern of formalization***

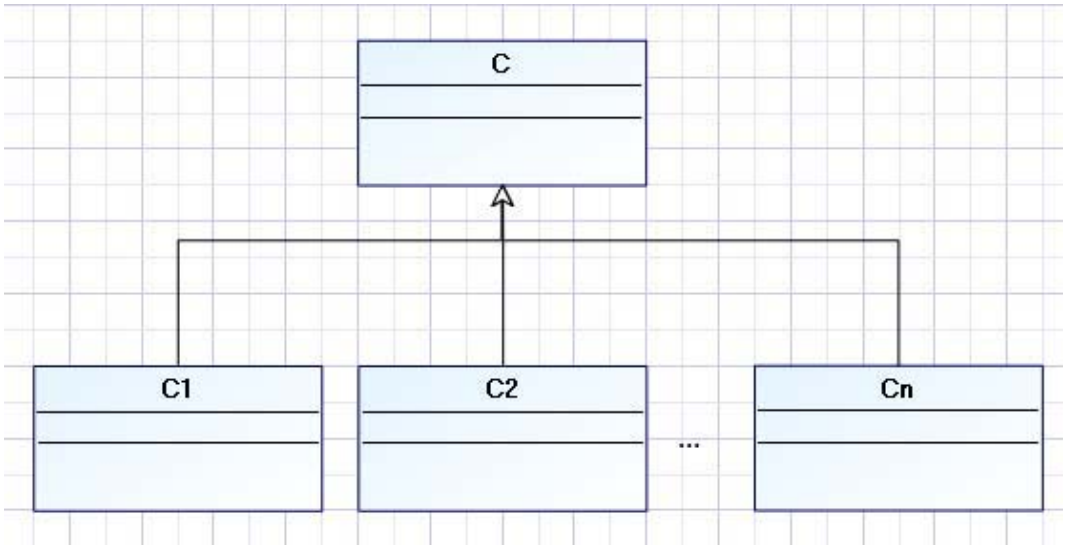


Figure 27 – A class hierarchy in UML

A class hierarchy as the one in Figure 27 is formally captured by means of the FOL assertions:

$\forall x.\ C_i(x) \supset C(x)$, for i = 1, . . . , n	***7° pattern of formalization***

Disjointness among $C_1, \ldots, C_n$ is expressed by the FOL assertions

$\forall x.\ C_i(x) \supset \Lambda^{n}_{j=i+1} \neg C_j\ (x)$, for i = 1, . . . , n − 1	***8° pattern of formalization***

The *completeness constraint* expressing that each instance of C is an instance of at least one of $C_1, \ldots, C_n$ is expressed by:

$\forall x.\ C(x) \supset V^{n}_{i=1}\ Ci(x)$	***9° pattern of formalization***

In our case, for example:

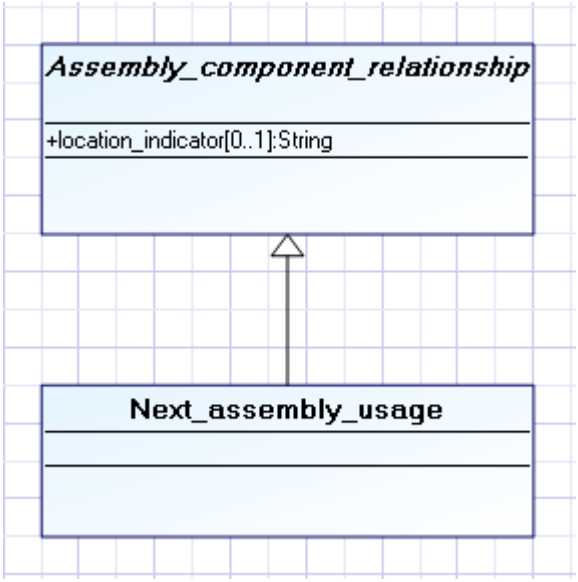


Figure 28 – A class hierarchy in STEP PDM

$\forall$ x. Next_assembly_usage(x) $\supset$ Assembly_component_relationship

In UML class diagrams, it is typically assumed that all classes not in the same hierarchy are a priori disjointed. Similarly, it is typically assumed that objects in a hierarchy must belong to a single most specific class. Hence, two classes in a hierarchy may have common instances only if they have a common subclass.

The semantics of other constructs of UML class diagram, such as n-ary associations or multiple generalization are not been formalized in FOL axioms, because they are not thought in the definition of standards semantical models.

5. Semantics of product data in standard models

Figure 29 shows a very small extract of the UML representation of the conceptualised Product Definition Model, from the IEC 62264. The

semantics of the modelling concepts, informally defined in the standard, have been formalized by FOL axioms as shown on Table 3.

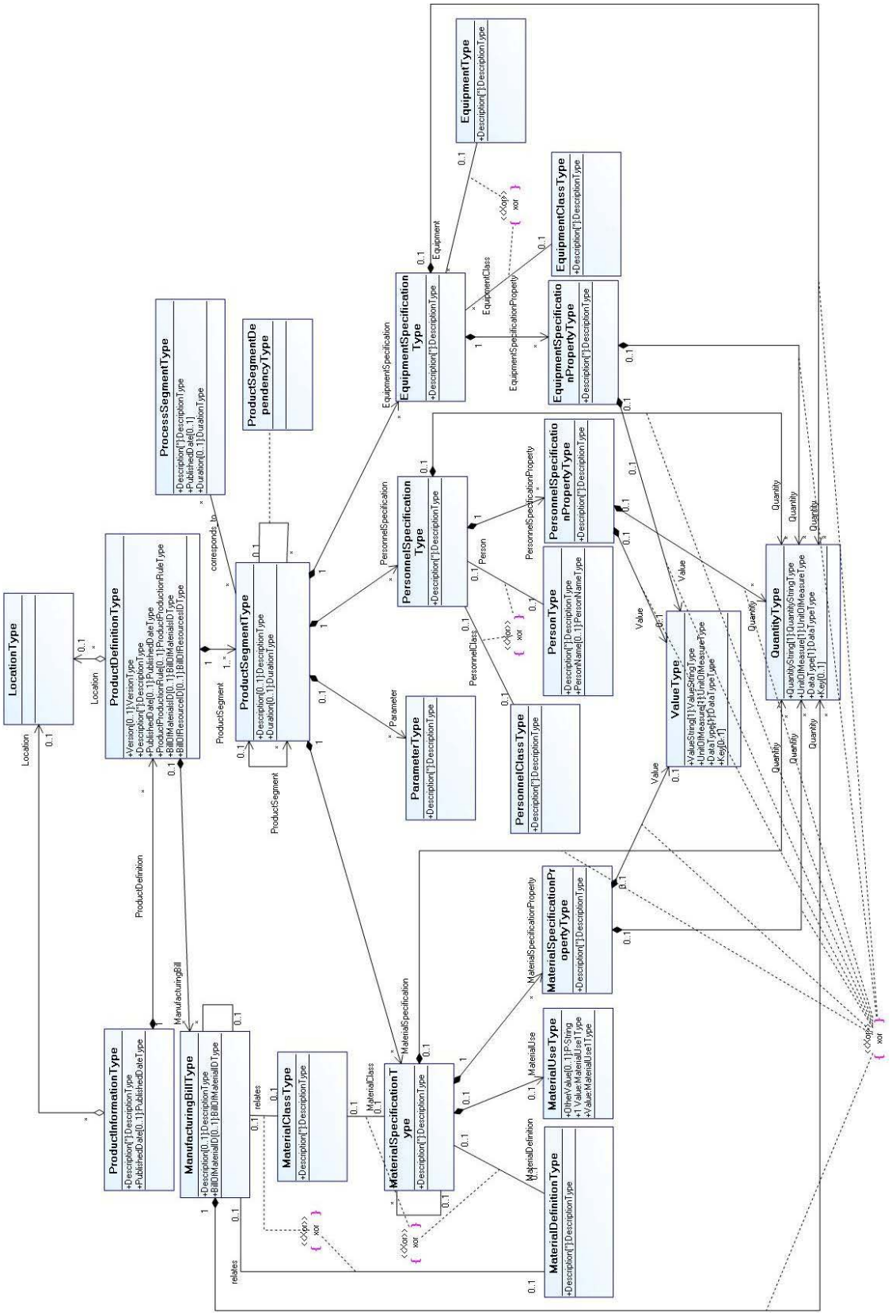


Figure 29 – Extract of UML formalization of Product Definition Model in IEC 62264

∀ x, y. (*ProductDefinitionType*(x) ∧ *Version*(x, y)) ⊃ VersionType (y)
∀ x. *ProductDefinitionType(x) ⊃ (0 ≤#{ y| Version(x, y)}* ⊃1)
∀ x, y. *(ProductDefinitionType(x) ∧ Description(x, y))* ⊃ DescriptionType(y)
∀ x. *ProductDefinitionType(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*})
∀ x, y. (*ProductDefinitionType(x) ∧ PublishedDate(x, y))* ⊃ PublishedDateType*(y)*
∀x. *ProductDefinitionType(x)* ⊃ (0 ≤#{ *y*| *PublishedDate(x, y)*} ⊃1)
∀ x, y. (*ManufacturingBillType*(x) ∧ *Description*(x, y)) ⊃ DescriptionType(y)
∀ x. *ManufacturingBillType(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*})
∀ x, y. *(ManufacturingBillType(x) ∧ BillOfMaterialType(x, y))* ⊃ BillOfMaterialType*(y)*
∀ x. *ManufacturingBillType(x)* ⊃ (0 ≤#{ *y*| *BillOfMaterialType(x, y)*} ⊃1)
∀x_1, x_2. ManufacturingBill(x_1, x_2) ⊃ *ProductDefinitionType*(x_1) ∧ *ManufacturingBillType*(x_2)
∀ x_1. *ProductDefinitionType*(x_1) ⊃ (0 ≤#{ *y*| *ManufacturingBill*(x_1, x_2)} ⊃1)
∀ x_2. *ManufacturingBillType*(x_2) ⊃ (0 ≤#{ *y*| *ManufacturingBill*(x_1, x_2)})
∀ *x, y.* (*ProductSegmentType(x)* ∧ *Description(x, y)*) ⊃ DescriptionType*(y)*
∀ x. *ProductSegmentType(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃1)
∀ x, y. (*MaterialSpecificationType(x)* ∧ *Description*(x, y)) ⊃ DescriptionType(y)
∀ x. *MaterialSpecificationType(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*})
∀x_1, x_2. MaterialSpecification(x_1, x_2) ⊃ *ProductSegmentType*(x_1) ∧ *MaterialSpecificationType*(x_2)
∀ x_1. *ProductSegmentType*(x_1) ⊃ (0 ≤#{ *y*| *MaterialSpecification*(x_1, x_2)})
∀ x_2. *MaterialSpecificationType*(x_2) ⊃ (0 ≤#{ *y*| *MaterialSpecification*(x_1, x_2)} ⊃ 1)
∀ x, y. (*MaterialDefinitionType(x)* ∧ *Description*(x, y)) ⊃ DescriptionType(y)
∀ x. *MaterialDefinitionType(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*})
∀x_1, x_2. MaterialDefinition(x_1, x_2) ⊃ *MaterialSpecificationType*(x_1) ∧ *MaterialDefinitionType*(x_2)
∀ x_1. *MaterialSpecificationType*(x_1) ⊃ (0 ≤#{ *y*| *MaterialDefinition*(x_1, x_2)} ⊃ 1)
∀ x_2. *MaterialDefinitionType*(x_2) ⊃ (0 ≤#{ *y*| *MaterialDefinition*(x_1, x_2)} ⊃ 1)
∀ x, y. (*MaterialUseType(x)* ∧ *OtherValue*(x, y)) ⊃ String(y)

$\forall$ x. *MaterialUseType(x)* $\supset$ (0 $\leq$#{ *y*| *OtherValue(x, y)*})
$\forall x_1, x_2$. MaterialUse(x_1, x_2) $\supset$ *MaterialSpecificationType*(x_1) $\wedge$ *MaterialUseType*(x_2)
$\forall$ x_1. *MaterialSpecificationType*(x_1) $\supset$ (0 $\leq$#{ *y*| *MaterialUse*(x_1, x_2)} $\supset$ 1)
$\forall$ x_2. *MaterialUseType*(x_2) $\supset$ (0 $\leq$#{ *y*| *MaterialUse*(x_1, x_2)} $\supset$ 1)
$\forall x_1, x_2$. relates(x_1, x_2) $\supset$ ManufacturingBillType(x_1) $\wedge$ MaterialDefinitionType(x_2)
$\forall$ x_1. *ManufacturingBillType*(x_1) $\supset$ (0 $\leq$#{ *y*| relates(x_1, x_2)} $\supset$ 1)
$\forall$ x_2. *MaterialDefinitionType*(x_2) $\supset$ (0 $\leq$#{ *y*| relates(x_1, x_2)} $\supset$ 1)
$\forall$ x, y. (*Quantity(x)* $\wedge$ *QuantityString*(x, y)) $\supset$ QuantityStringType(y)
$\forall$ x. *Quantity(x)* $\supset$ (1 $\leq$#{ *y*| *QuantityString(x, y)*} $\supset$ 1)
$\forall$ x, y. (*Quantity(x)* $\wedge$ *UnitOfMeasure*(x, y)) $\supset$ UnitOfMeasureType(y)
$\forall$ x. *Quantity(x)* $\supset$ (1 $\leq$#{ *y*| *UnitOfMeasure(x, y)*} $\supset$ 1)
$\forall$ x, y. (*Quantity(x)* $\wedge$ *DataType*(x, y)) $\supset$ DataTypeType(y)
$\forall$ x. *Quantity(x)* $\supset$ (1 $\leq$#{ *y*| *DataType(x, y)*} $\supset$ 1)
$\forall x_1, x_2$. *Quantity*(x_1, x_2) $\supset$ *ManufacturingBillType*(x_1) $\wedge$ *QuantityType*(x_2)
$\forall$ x_1. *ManufacturingBillType*(x_1) $\supset$ (0 $\leq$#{ *y*| relates(x_1, x_2)})
$\forall$ x_2. *QuantityType*(x_2) $\supset$ (1 $\leq$#{ *y*| relates(x_1, x_2)} $\supset$ 1)

Table 3 – FOL formalization of Product Definition Model

Figure 30 shows an a very small extract of UML representation of the conceptualized ISO STEP-PDM standard model, reduced to some concepts that may have a correspondence with those defined into the Product Definition model of IEC 62264. The semantics of the modelling concepts, informally defined in the standard, have been formalized by FOL axioms as shown on Table 4.

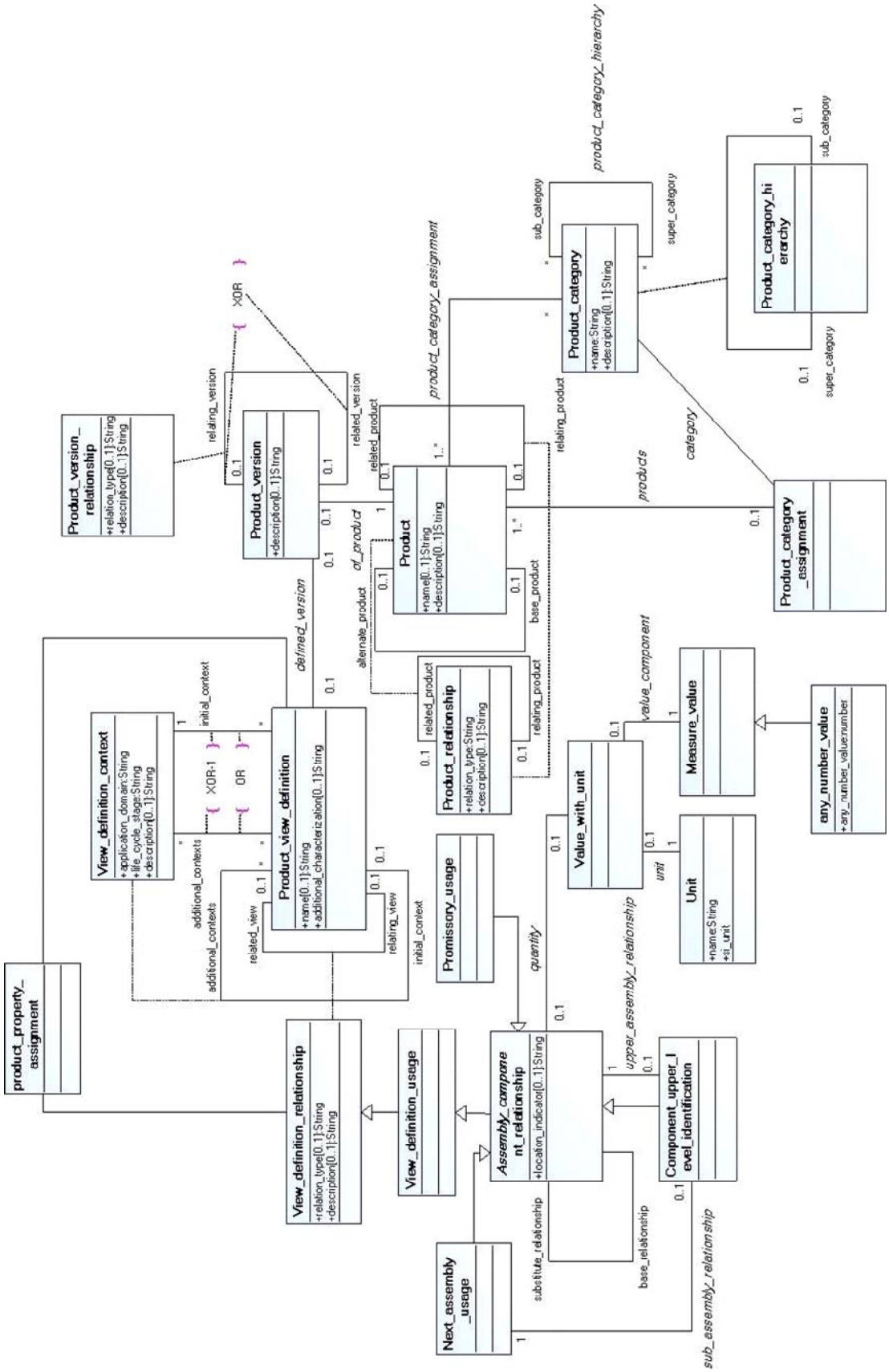


Figure 30 – Extract of concepts in STEP PDM (Tursi et al., 2009)

∀ x, y. (*Product(x)* ∧ *Name*(x, y)) ⊃ String(y)
∀ x. *Product(x)* ⊃ (0 ≤#{ *y*| *Name(x, y)*} ⊃ 1)
∀ x, y. (*Product(x)* ∧ *Description*(x, y)) ⊃ String(y)
∀ x. *Product(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
∀ x, y. (*Product_version(x)* ∧ *Description*(x, y)) ⊃ String(y)
∀ x. *Product_version(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
∀x_1, x_2. of_product(x_1, x_2) ⊃ *Product*(x_1) ∧*Product_version*(x_2)
∀ x_1. *Product*(x_1) ⊃ (0 ≤#{ *y*| of_product(x_1, x_2)} ⊃ 1)
∀ x_2. *Product_version*(x_2) ⊃ (1 ≤#{ *y*| of_product(x_1, x_2)} ⊃ 1)
∀ x, y. (*Product_view_definition(x)* ∧*Name*(x, y)) ⊃ String(y)
∀ x. *Product_view_definition(x)* ⊃ (0 ≤#{ *y*| *Name(x, y)*} ⊃ 1)
∀ x, y. (*Product_view_definition(x)* ∧Additional_characterization(x, y)) ⊃ String(y)
∀ x. *Product_view_definition(x)* ⊃ (0 ≤#{ *y*| Additional_characterization*(x, y)*} ⊃ 1)
∀x_1, x_2. defined_version(x_1, x_2) ⊃ *Product_version*(x_1) ∧*Product_view_definition* (x_2)
∀ x_1. *Product_version*(x_1) ⊃ (0 ≤#{ *y*| defined_version(x_1, x_2)} ⊃ 1)
∀ x_2. *Product_view_definition*(x_2) ⊃ (0 ≤#{ *y*| defined_version(x_1, x_2)} ⊃ 1)
∀ x, y. (*View_definition_context(x)* ∧ *Application_domain*(x, y)) ⊃ String(y)
∀ x. *View_definition_context(x)* ⊃ (1 ≤#{ *y*| *Application_domain(x, y)*} ⊃ 1)
∀ x, y. (*View_definition_context(x)* ∧ *Life_cycle_stage*(x, y)) ⊃ String(y)
∀ x. *View_definition_context(x)* ⊃ (1 ≤#{ *y*| *Life_cycle_stage(x, y)*} ⊃ 1)
∀ x, y. (*View_definition_context(x)* ∧ *Description*(x, y)) ⊃ String(y)
∀ x. *View_definition_context(x)* ⊃ (1 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
∀ x, y. *Product_view_definition*(x) ∧ initial_context(x,y) ⊃ View_definition_context(x)
∀ x, y. *Product_view_definition(*x) ∧ additional_contexts(x,y) ⊃ *View_definition_context(*x)
∀ x. *Product_view_definition*(x) ⊃ ∃ y. initial_context(x,y)
∀ x. *Product_view_definition*(x) ⊃ ∃ y. additional_contexts(x,y)

$\forall$ x, y, y'. *Product_view_definition*(x) $\wedge$ initial_context(x,y) $\wedge$ initial_context(x, y') $\supset$ y = y'
$\forall$ x, y, y'. *Product_view_definition*(x) $\wedge$ additional_contexts(x,y) $\wedge$ additional_contexts(x, y') $\supset$ y = y'
$\forall$ x, x', y_1, y_2. *Product_view_definition*(x) $\wedge$ *Product_view_definition(*x') $\wedge$ initial_context(x, y_1) $\wedge$ initial_context(x', y_1)) $\wedge$ additional_contexts(x, y_2) $\wedge$ additional_contexts(x', y_2)) $\supset$ x= x'
$\forall$ x, y. (*View_definition_relationship(x)* $\wedge$ *Relation_type*(x, y)) $\supset$ String(y)
$\forall$ x. *View_definition_relationship(x)* $\supset$ (0 $\leq$#{ *y*| *Relation_type(x, y)*} $\supset$ 1)
$\forall$ x, y. (*View_definition_relationship(x)* $\wedge$ *Description*(x, y)) $\supset$ String(y)
$\forall$ x. *View_definition_relationship(x)* $\supset$ (0 $\leq$#{ *y*| *Description(x, y)*} $\supset$ 1)
$\forall$ x, y. *View_definition_relationship*(x) $\wedge$ relating_view(x,y) $\supset$ *Product_view_definition*(x)
$\forall$ x, y. *View_definition_relationship*(x) $\wedge$ related_view(x,y) $\supset$ *Product_view_definition*(x)
$\forall$ x. *View_definition_relationship*(x) $\supset$ $\exists$ y. relating_view(x,y)
$\forall$ x. *View_definition_relationship*(x) $\supset$ $\exists$ y. related_view(x,y)
$\forall$ x, y, y'. *View_definition_relationship*(x) $\wedge$ relating_view(x,y) $\wedge$ relating_view(x, y') $\supset$ y = y'
$\forall$ x, y, y'. *View_definition_relationship*(x) $\wedge$ related_view(x,y) $\wedge$ related_view(x, y') $\supset$ y = y'
$\forall$ x, x', y_1, y_2. *View_definition_relationship*(x) $\wedge$ *View_definition_relationship*(x') $\wedge$ relating_view(x, y_1) $\wedge$ initial_context(x', y_1)) $\wedge$ additional_contexts(x, y_2) $\wedge$ additional_contexts(x', y_2)) $\supset$ x= x'
$\forall$ x. *View_definition_usage(x)* $\supset$ *View_definition_relationship*(x)
$\forall$ x, y. (*Assembly_component_relationship(x)* $\wedge$ *Location_indicator*(x, y)) $\supset$ String(y)
$\forall$ x. *Assembly_component_relationship(x)* $\supset$ (0 $\leq$#{ *y*| *Location_indicator(x, y)*} $\supset$ 1)
$\forall$ x. *Assembly_component_relationship(x)* $\supset$ *View_definition_usage*(x)
$\forall$ x. *Value_with_unit(x)*

$\forall$ x, y. (*Unit(x)* $\wedge$ *Name*(x, y)) $\supset$ String(y)
$\forall$ x. *Unit(x)* $\supset$ (1 $\leq$#{ *y*| *Name(x, y)*} $\supset$ 1)
$\forall x_1$, x_2. quantity(x_1, x_2) $\supset$ *Assembly_component_relationship*(x_1) $\wedge$ *Value_with_unit*(x_2)
$\forall$ x_1. *Assembly_component_relationship*(x_1) $\supset$ (0 $\leq$#{ *y*| quantity(x_1, x_2)} $\supset$ 1)
$\forall$ x_2. *Value_with_unit*(x_2) $\supset$ (0 $\leq$#{ *y*| quantity(x_1, x_2)} $\supset$ 1)
$\forall x_1$, x_2. unit(x_1, x_2) $\supset$ *Value_with_unit*(x_1) $\wedge$ *Unit*(x_2)
$\forall$ x_1. *Value_with_unit*(x_1) $\supset$ (1 $\leq$#{ *y*| of_product(x_1, x_2)} $\supset$ 1)
$\forall$ x_2. *Unit*(x_2) $\supset$ (0 $\leq$#{ *y*| unit(x_1, x_2)} $\supset$ 1)
$\forall$ x. *Measure_value(x)*
$\forall$ x, y. (*Any_number_value(x)* $\wedge$*any_number_value*(x, y)) $\supset$ Number(y)
$\forall$ x. *Any_number_value(x)* $\supset$ (1 $\leq$#{ *y*| *any_number_value(x, y)*} $\supset$ 1)
$\forall$ x. *Any_number_value(x)* $\supset$ *Measure_value*(x)
$\forall$ x. *Next_assembly_usage(x)* $\supset$ *Assembly_component_relationship*(x)

Table 4 – FOL formalization of STEP PDM model

For both the standards ISO and IEC, class, attributes, associations, multiplicity of attributes and associations, association classes and generalizations were been formalized using FOL. We have got two disjoined ontologies in term of concepts, but they are sharing common knowledge related to manufactured product. Among those top ontologies that contain highly abstract concepts, we propose to map common concepts in order to build a domain ontology referred as **Product Ontology**, which will be compatible with the standards IEC 62264 and ISO 10303 STEP PDM.

6. Mapping formalization

To overcome the problem of semantic interoperability, there already exist some techniques. The majority part of these refers to mapping between ontologies. To use ontology mapping consists in finding semantics correspondences between concepts from two given ontologies.

Considering the ontology as a model which formalizes shared knowledge, as well as standard models, a mapping is defined by (Su, 2002) in this way: given two ontologies O_1 and O_2, mapping one ontology with another means that for each concept (node) in ontology O_1, we try to find a corresponding concept (node), which has the same or similar semantics, in ontology O_2 and vice versa. Other but similar definitions are given by (Ding et al., 2001). Formally an ontology mapping function can be defined in the following way (Ehrig and Sure, 2004):

- map: $O_{i1} \rightarrow O_{i2}$

$map(e_{i1j1}) = e_{i2j2}$, if $sim(e_{i1j1}, e_{i2j2}) > t$ with t being the threshold

entity e_{i1j1} is mapped onto e_{i2j2}; they are semantically identical, each entity e_{i1j1} is mapped to at most one entity e_{i2j2}.

Where:

- O_i: ontology, with ontology index $_i \in N$
- *sim*(x, y): similarity function
- e_{ij}: entities of O_i, with $e_{ij} \in \{C_i, R_i, I_i\}$, entity index $j \in N$
- $sim(e_{i1j1}, e_{i2j2})$: similarity function between two entities e_{i1j1} and e_{i2j2} ($i_1 \neq i_2$).

Through semantics relationships between both concepts, we deduce one or more FOL predicates, which formalizes mapping between STEP PDM concepts and IEC 62264 ones. First of all, the standards concepts are compared with themselves, according to the subject they consider, as we can see below for the case of BOM:

$\forall$ x, y. (*MaterialDefinitionType(x)* $\wedge$ *Description(x, y)*) $\supset$ DescriptionType*(y)*	Contain a definition of a material definition	$\forall$ x, y. (*Product(x)* $\wedge$ *Name*(x, y)) $\supset$ String(y)
		$\forall$ x. *Product(x)* $\supset$ (0 $\leq$#{ *y*| *Name(x, y)*} $\supset$ 1)
$\forall$ x.		$\forall$ x, y. (*Product(x)* $\wedge$

MaterialDefinitionType(x) ⊃ (0 ≤#{ *y*| *Description(x, y)*})		*Description*(x, y)) ⊃ String(y)
		∀ x. *Product(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
No B2MML axioms contain this kind of information	Contain definition of revision of the Product.	∀ x, y. (*Product_version(x)* ∧ *Description*(x, y)) ⊃ String(y)
		∀ x. *Product_version(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
		∀x_1, x_2. of_product(x_1, x_2) ⊃ *Product*(x_1) ∧ *Product_version*(x_2)
		∀ x_1. *Product*(x_1) ⊃ (0 ≤#{ *y*| of_product(x_1, x_2)} ⊃ 1)
		∀ x_2. *Product_version*(x_2) ⊃ (1 ≤#{ *y*| of_product(x_1, x_2)} ⊃ 1)
No B2MML axioms contain this kind of information	Contain properties that characterize the Product_version in the initial_context and additional_contexts.	∀ x, y. (*Product_view_definition(x)* ∧ *Name*(x, y)) ⊃ String(y)
		∀ x. *Product_view_definition(x)* ⊃ (0 ≤#{ *y*| *Name(x, y)*} ⊃ 1)
		∀ x, y. (*Product_view_definition(x)* ∧ Additional_characterization(x, y)) ⊃ String(y)
		∀ x. *Product_view_definition(x)* ⊃ (0 ≤#{ *y*| Additional_characterization*(x, y)*} ⊃ 1)
		∀x_1, x_2. defined_version(x_1,

		$x_2) \supset$ *Product_version*$(x_1) \wedge$ *Product_view_definition* (x_2)
		$\forall\ x_1$. *Product_version*$(x_1) \supset$ $(0 \leq \#\{ y \mid$ defined_version$(x_1, x_2)\} \supset 1)$
		$\forall\ x_2$. *Product_view_definition*$(x_2) \supset$ $(0 \leq \#\{ y \mid$ defined_version$(x_1, x_2)\} \supset 1)$
No B2MML axioms contain this kind of information	Contain information about the application domain and the life cycle stage. it identifies a universe of discourse suitable for the description of products.	$\forall$ x, y. (*View_definition_context(x)* $\wedge$ *Application_domain*(x, y)) $\supset$ String(y)
		$\forall$ x. *View_definition_context(x)* $\supset$ $(1 \leq \#\{ y \mid$ *Application_domain(x, y)*$\} \supset 1)$
		$\forall$ x, y. (*View_definition_context(x)* $\wedge$ *Life_cycle_stage*(x, y)) $\supset$ String(y)
		$\forall$ x. *View_definition_context(x)* $\supset$ $(1 \leq \#\{ y \mid$ *Life_cycle_stage(x, y)*$\} \supset 1)$
		$\forall$ x, y. (*View_definition_context(x)* $\wedge$ *Description*(x, y)) $\supset$ String(y)
		$\forall$ x. *View_definition_context(x)* $\supset$

		(1 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
		∀ x, y. *Product_view_definition*(x) ∧ initial_context(x,y) ⊃ *View_definition_context*(x)
		∀ x, y. *Product_view_definition*(x) ∧ additional_contexts(x,y) ⊃ *View_definition_context*(x)

∀x_1, x_2. MaterialDefinition(x_1, x_2) ⊃ *MaterialSpecificationTyp e*(x_1) ∧ *MaterialDefinitionType*(x_2)	Include the identification of instance of resource	No STEP PDM axioms contain this kind of information
∀ x_1. *MaterialSpecificationTyp e*(x_1) ⊃ (0 ≤#{ *y*| MaterialDefinition(x_1, x_2)} ⊃ 1)		
∀ x_2. *MaterialDefinitionType*(x_2) ⊃ (0 ≤#{ *y*| MaterialDefinition(x_1, x_2)} ⊃ 1)		
∀ x, y. (*MaterialSpecificationTyp e(x)* ∧ *Description*(x, y)) ⊃ DescriptionType(y)		
∀ x.		

MaterialSpecificationTyp e(x) ⊃ (0 ≤#{ *y*| *Description(x, y)*})		

∀x_1, x_2. relates(x_1, x_2) ⊃ *ManufacturingBillType*(x_1) ∧ *MaterialDefinitionType*(x_2)	Include the identification of the material definition in the corresponding bill of material	No STEP PDM axioms contain this kind of information
∀ x_1. *ManufacturingBillType*(x_1) ⊃ (0 ≤#{ *y*| relates(x_1, x_2)} ⊃ 1)		
∀ x_2. *MaterialDefinitionType*(x_2) ⊃ (0 ≤#{ *y*| relates(x_1, x_2)} ⊃ 1)		

∀ x, y. (*MaterialUseType(x)* ∧ *OtherValue*(x, y)) ⊃ String(y)	Include information about use (consumed or produced) of the resource identified in the BOM of product. In other words, in a relationship type "product-component", identify the link between the related_view (the product) and the relating_view (the	∀ x, y. (*View_definition_relationship(x)* ∧ *Relation_type*(x, y)) ⊃ String(y)
∀ x. *MaterialUseType(x)* ⊃ (0 ≤#{ *y*| *OtherValue(x, y)*})		∀ x. *View_definition_relationship(x)* ⊃ (0 ≤#{ *y*| *Relation_type(x, y)*} ⊃ 1)
∀x_1, x_2. MaterialUse(x_1, x_2) ⊃ *MaterialSpecificationTyp e*(x_1) ∧ *MaterialUseType*(x_2)		∀ x, y. (*View_definition_relationship(x)* ∧ *Description*(x, y)) ⊃ String(y)
∀ x_1.		∀ x.

MaterialSpecificationTyp e(x_1) ⊃ (0 ≤#{ *y*| MaterialUse(x_1, x_2)} ⊃ 1)	component)	*View_definition_relationship(x)* ⊃ (0 ≤#{ *y*| *Description(x, y)*} ⊃ 1)
∀ x_2. *MaterialUseType*(x_2) ⊃ (0 ≤#{ *y*| MaterialUse(x_1, x_2)} ⊃ 1)		∀ x, y. *View_definition_relationship*(x) ∧ relating_view(x,y) ⊃ *Product_view_definition*(x)
		∀ x, y. *View_definition_relationship*(x) ∧ related_view(x,y) ⊃ *Product_view_definition*(x)
		∀ x. *View_definition_relationship*(x) ⊃ ∃ y. relating_view(x,y)
		∀ x. *View_definition_relationship*(x) ⊃ ∃ y. related_view(x,y)
		∀ x, y, y'. *View_definition_relationship*(x) ∧ relating_view(x,y) ∧ relating_view(x, y') ⊃ y = y'
		∀ x, y, y'. *View_definition_relationship*(x) ∧ related_view(x,y) ∧ related_view(x, y') ⊃ y = y'
		∀ x, x', y_1, y_2. *View_definition_relationship*(x) ∧ *View_definition_relationship*(x') ∧ relating_view(x, y_1) ∧ initial_context(x', y_1)) ∧ additional_contexts(x, y_2) ∧ additional_contexts(x', y_2)) ⊃

		x= x’
		∀ x. *View_definition_usage(x)* ⊃ *View_definition_relationship*(x)
		∀ x, y. (*Assembly_component_relati onship(x)* ∧ *Location_indicator*(x, y)) ⊃ String(y)
		∀ x. *Assembly_component_relatio nship(x)* ⊃ (0 ≤#{ *y*| *Location_indicator(x, y)*} ⊃ 1)
		∀ x. *Assembly_component_relatio nship(x)* ⊃ *View_definition_usage*(x)
		∀ x. *Next_assembly_usage(x)* ⊃ *Assembly_component_relatio nship*(x)

∀ x, y. (*Quantity(x)* ∧ *QuantityString*(x, y)) ⊃ QuantityStringType(y)	Include the quantity of the material needed	∀ x. *Value_with_unit(x)*
∀ x. *Quantity(x)* ⊃ (1 ≤#{ *y*| *QuantityString(x, y)*} ⊃ 1)		∀ x, y. (*Unit(x)* ∧ *Name*(x, y)) ⊃ String(y)
∀ x, y. (*Quantity(x)* ∧ *UnitOfMeasure*(x, y)) ⊃ UnitOfMeasureType(y)		∀ x. *Unit(x)* ⊃ (1 ≤#{ *y*| *Name(x, y)*} ⊃ 1)

$\forall$ x. *Quantity(x)* $\supset$ (1 $\leq$#{ *y*| *UnitOfMeasure(x, y)*} $\supset$ 1)		$\forall x_1, x_2$. quantity(x_1, x_2) $\supset$ *Assembly_component_relationship*(x_1) $\wedge$ *Value_with_unit*(x_2)
$\forall$ x, y. (*Quantity(x)* $\wedge$ *DataType*(x, y)) $\supset$ *DataTypeType*(y)		$\forall$ x_1. *Assembly_component_relationship*(x_1) $\supset$ (0 $\leq$#{ *y*| quantity(x_1, x_2)} $\supset$ 1)
$\forall$ x. *Quantity(x)* $\supset$ (1 $\leq$#{ *y*| *DataType(x, y)*} $\supset$ 1)		$\forall$ x_2. *Value_with_unit*(x_2) $\supset$ (0 $\leq$#{ *y*| quantity(x_1, x_2)} $\supset$ 1)
$\forall x_1, x_2$. Quantity(x_1, x_2) $\supset$ *ManufacturingBillType*(x_1) $\wedge$ QuantityType(x_2)		$\forall x_1, x_2$. unit(x_1, x_2) $\supset$ *Value_with_unit*(x_1) $\wedge$ *Unit*(x_2)
$\forall$ x_1. *ManufacturingBillType*(x_1) $\supset$ (0 $\leq$#{ *y*| relates(x_1, x_2)})		$\forall$ x_1. *Value_with_unit*(x_1) $\supset$ (1 $\leq$#{ *y*| of_product(x_1, x_2)} $\supset$ 1)
		$\forall$ x_2. *Unit*(x_2) $\supset$ (0 $\leq$#{ *y*| unit(x_1, x_2)} $\supset$ 1)
		$\forall$ x. *Measure_value(x)*
		$\forall$ x, y. (*Any_number_value(x)* $\wedge$ *any_number_value*(x, y)) $\supset$ Number(y)
		$\forall$ x. *Any_number_value(x)* $\supset$ (1 $\leq$#{ *y*| *any_number_value(x, y)*} $\supset$ 1)
		$\forall$ x. *Any_number_value(x)* $\supset$ *Measure_value*(x)

Each relation between different concepts can be studied and it is possible to define semantic correspondences between them (Baîna, 2006) in order

to compare the contained information. Different cases can occur: equivalence, represented by ≡ symbol (same definition for concepts semantics in the two standards), inclusion, represented by ⊂ symbol (a semantic concept includes the other one), and intersection, represented by ∩ symbol, (the concepts intersection defines the common sense of the two concepts). Finally, some concepts cannot have semantic correspondence (Tursi et al., 2007).

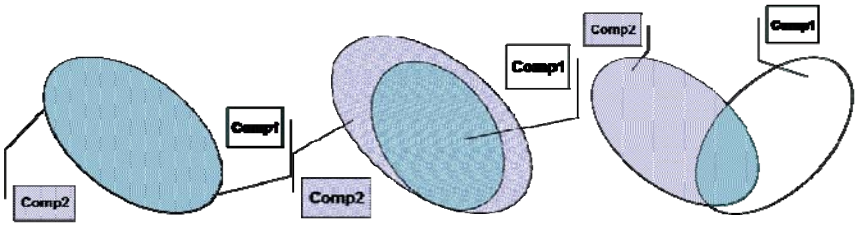


Figure 31 - Correspondences between the concepts semantics: (a) equivalence, (b) inclusion, (c) intersection

When different concepts are identified through syntactical analysis and compared with themselves, based on the subject they deal with, for each of them we value the contained information, formalized by FOL. For example, in the case of the first mapping rule of Table 6, we deduce that the class *Product* in STEP PDM express the same semantics of *MaterialDefinitionType* in IEC 62264. In fact, the formalisation FOL of *Product* says that the object *Product* can have a name (*Name* attribute) and one or more descriptions (*Description* attribute) that define the class. In the same way, the formalisation FOL of *MaterialDefinitionType* says that the object *MaterialDefinitionType* can have one or more descriptions. Because these heterogeneous classes consider the same subject and because the attribute *Name* of *Product* class is redundant, we can deduce that the two classes contain the same information. Similarly, if we compare the formalisation FOL of *Value_with_unit* class of STEP PDM with *QuantityType* one of IEC 62264 because they consider the same subject, we can read that *Value_with_unit* class is composed by *Measure_length*

class, which describes only the value of measure and by *Unit* class, which describes only the unit of measure used. *QuantityType* class, instead, contains information about data type (*Data Type* attribute), about measure value (*Value* attribute) and about unit of measure (*Unit_of_value* attribute). Then, information contained in *Unit* class and in *Measure_lenght* are contained in *QuantityType* (Table 5).

$\forall$ x. *Product(x)* $\equiv$ *MaterialDefinitionType(x)*
Product_version, Product_view_definition, *View_definition_context* are concepts, present in PDM STEP but without semantics equivalence in IEC 62264 Product Definition
MaterialSpecificationType, ManufacturingBillType are concepts, present in IEC 62264 Product Definition but without semantics equivalence in PDM STEP
$\forall$ x. *Assembly_component_relationship. relating_view(x)* $\equiv$ *MaterialUseType. OtherValue(x) ="consumed"*
$\forall$ x. *Assembly_component_relationship. related_view(x)* $\equiv$ *MaterialUseType. OtherValue(x) ="produced"*
$\forall$ x. *Unit(x)* $\subset$ *QuantityType(x)*
$\forall$ x. *Measure_value(x)* $\subset$ *QuantityType(x)*

Table 5 – Mapping rules

As shown, it is possible to find some information scattered in the IEC 62264 models and in the STEP PDM ones, even if they model the information in different way. The common minimum denominator between them will represent the core of our Product Ontology, to which it will be possible to add information specific of both standard, in order to have an ontological model, consistent with both IEC 62264 and STEP PDM.

7. Ontological Model

Taking into account the previous FOL axioms of standard models and the concepts mapping between them, the Product Ontology is proposed.

Starting from the realization, demonstrated by mapping, that the B2MML models contain the information included in STEP PDM, we merge the specific information of STEP PDM in B2MML ontology, in order to build an ontological model that will be able to store all product technical data and information, consistent with both standards. In other words, STEP PDM will extend the B2MML ontology. This common model will be able to provide mappings from and to the enterprise applications with respect to product life cycle.

A deductive system is used to demonstrate, on a purely syntactic basis, that one formula is a logical consequence of another formula. A rule of inference states that, given a particular (or a set) of FOL axioms, another one can be derived as a logical conclusion. In this way it is easy integrating information, in a common model.

For example: if the following FOL axioms are true:

$\forall$ x. *Product(x)* $\equiv$ *MaterialDefinitionType (x)*	Mapping rule

$\forall$ x, y. (*Product_version(x)* $\wedge$ *Description*(x, y)) $\supset$ String(y)	FOL axioms of STEP PDM
$\forall$ x. *Product_version(x)* $\supset$ (0 $\leq$#{ *y*| *Description(x, y)*} $\supset$ 1)	
$\forall x_1$, x_2. of_product(x_1, x_2) $\supset$ *Product*(x_1) $\wedge$ *Product_version*(x_2)	

Then, the following axioms are true:

$\forall$ x, y. (*Product_version(x)* $\wedge$ *Description*(x, y)) $\supset$ String(y)	New FOL axioms of Product Ontology
$\forall$ x. *Product_version(x)* $\supset$ (0 $\leq$#{ *y*| *Description(x, y)*} $\supset$ 1)	
$\forall x_1$, x_2. of_product(x_1, x_2) $\supset$ *MaterialDefinitionType*(x_1) $\wedge$	

Product_version(x_2)	

For uniformity of language, we can rename *Product_version* as *MaterialDefinitionVersionType,* in such way:

$\forall$ x, y. (*MaterialDefinitionVersionType(x)* $\wedge$ *Description*(x, y)) $\supset$ String(y)	New FOL axioms of Product Ontology
$\forall$ x. *MaterialDefinitionVersionType(x)* $\supset$ (0 $\leq \#\{ y \mid$ *Description(x, y)*$\}$ $\supset$ 1)	
$\forall x_1, x_2$. of_product(x_1, x_2) $\supset$ *MaterialDefinitionType*(x_1) $\wedge$ *MaterialDefinitionVersionType*(x_2)	

This step allows to integrate the concept “Product_version”, specific in STEP PDM, in IEC 62264.

The Product Definition Model of B2MML, integrated with STEP PDM concepts, is shown in Figures 32 (a)-(b).

In the same way, all models of B2MML are been extended with STEP PDM modules.

The ontological model that represents our Product Ontology is described firstly by UML class diagram formalism. *Mega* suite is the software tool useful for this scope.

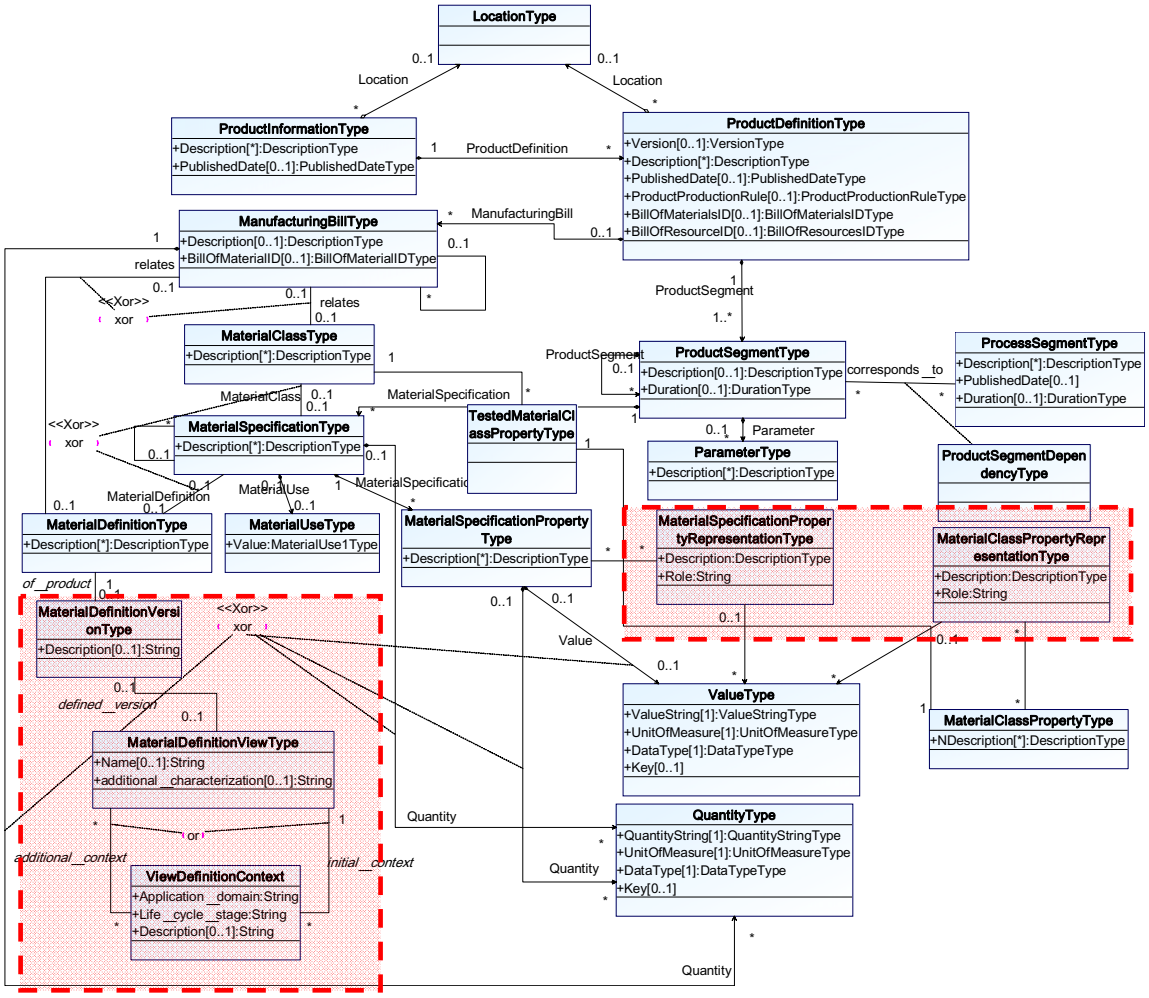


Figure 32 – (a) The Product Ontology (the red part is coming from STEP PDM)

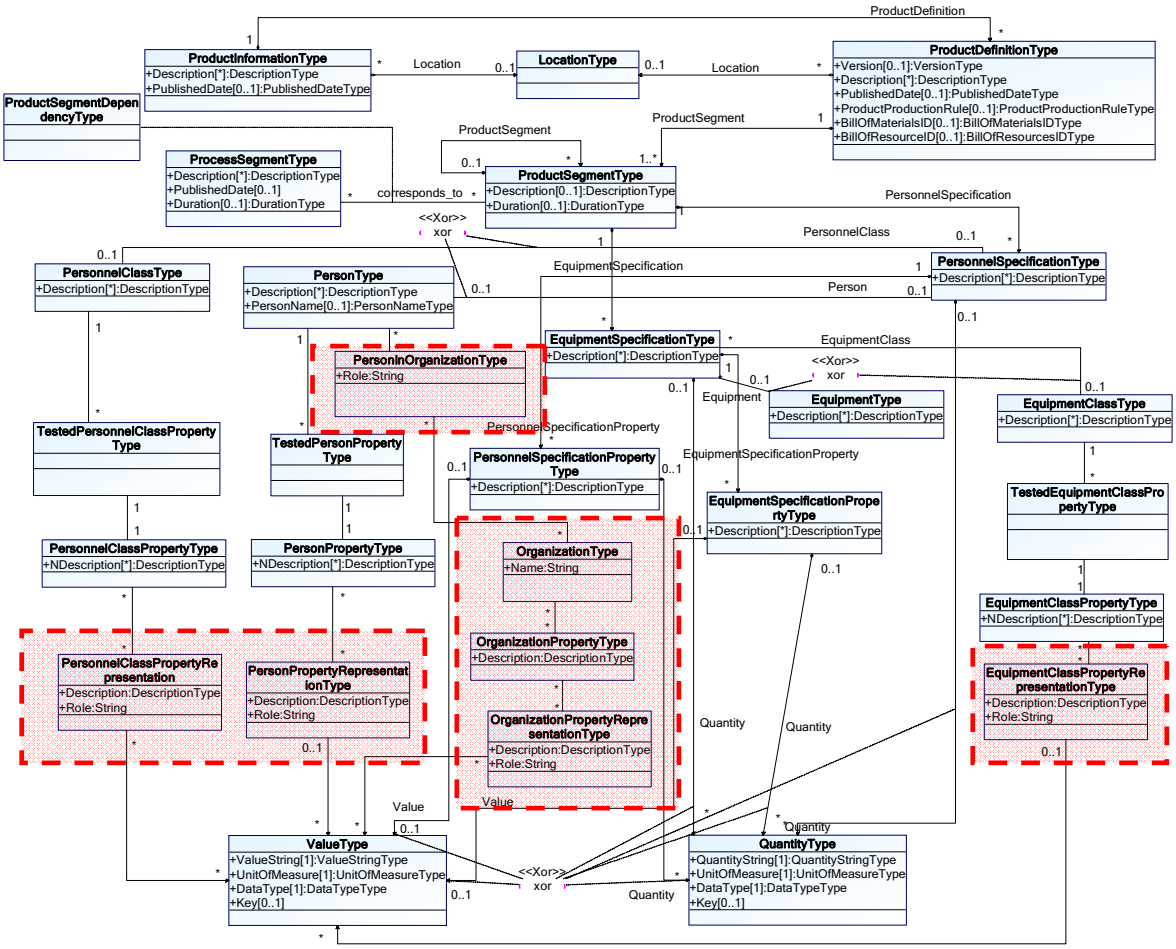


Figure 32 – (b) The Product Ontology (the red part is coming from STEP PDM)

8. Conclusion

In this chapter we describe and argue the proposed methodology for building a Product Ontology, which, including domain rules, is able to express and share product knowledge among systems. The Product Ontology, in fact, endeavouring existing standards related to product technical data modelling for the definition of product information, can allow a non ambiguous model to represent knowledge and concept, processable by many enterprise applications adopted in manufacturing environment.

We chose the First Order Logic (FOL) to express “patterns” of UML formalisation, in order to formalise the concepts semantics.

In the next chapter a test cases is described, analysed and schematised with the model, in order to validate the same model. The proposed

approach aims to foster interoperability along the diverse enterprise applications, during product lifecycle. This ontological model was established, re-using, at best, existing work around some standards: IEC 62264 and ISO 10303 STEP-PDM. The model is technology independent and fits to different application domains.

9. References of the chapter

Baîna S. (2006). Interoperabilité Dirigée par les Modèles: Une Approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise. Thèse de doctorat de l'Universitè Henri Poincarè. December 2006, Nancy I, France (in french).

Berardi, D., Calvanese, D., De Giacomo, G.(2005). Reasoning on UML class diagrams, Artificial Intelligence 168 (1–2) 70–118.

Calvanese, D., Lenzerini, M., and Nardi, D. (1998). Description logics for conceptual data modelling. In J. Chomicki and G. Saake, editors, Logics for Databases and Information Systems, pages 229–264. Kluwer Academic Publisher.

Dassisti M., Panetto H., Tursi A., De Nicolò M. (2008). Ontology-Based Model for Production-Control Systems Interoperability. Proceedings of the 5th International CIRP Conference on Digital Enterprise Technology, 22-24 October 2008, Nantes, France.

Ding Y., Fensel D., Klein M., Boris Omelayenko (2001). Ontology management: Survey, requirements and directions. Deliverable 4. IST Project IST-1999-10132.

Doan A., Madhavan J., Domingos P., Halevy A.. (2002). Learning to map between ontologies on the semantic web. In The Eleventh International WWW Conference, Hawaii, US.

Ehrig M., Sure Y. (2004). Ontology Mapping - An Integrated Approach. The Semantic Web: Research and Applications. Vol 3053, 76-91, ISBN978-3-540-21999-6.

Euzenat J., Valtchev P.(2004). Similarity-based ontology alignment in OWL-Lite. In The 16th EuropeanConference on Artificial Intelligence (ECAI-04), Valencia, Spain.

Fowler M. (1997). UML distilled. USA, Addison-Wesley, ISBN 0-201-32563-2.

Ganter B., Wille R. (1999) Formal Concept Analysis: Mathematical foundations. Springer, Berlin-Heidelberg.

Hovy E..(1998). Combining and standardizing largescale, practical ontologies for machine translation and other uses. In The First International Conference on Language Resources and Evaluation (LREC), pages 535–542, Granada, Spain.

Exff project (2003). The exff EXPRESS Guide. Release 0.1. http://exff.sourceforge.net

ISO TC/184/SC4/WG11 N013, Requirements specification for EXPRESS mapping language, (1997).

ISO/TC184/SC4/WG11 N088, EXPRESS-X Language Reference Manual (1999).

ISO/TC184/SC4, Industrial automation systems and integration—product data representation and exchange—Part 11:Descriptive Methods: The EXPRESS Language Reference Manual (1992).

Kalfoglou Y., Schorlemmer M.. (2003). IF-Map: an ontology mapping method based on information flow theory. Journal on Data Semantics, 1(1):98–127, Oct. 2003.

Kalfoglou, Y., Schorlemmer, M.. (2003). Ontology mapping: the state of the art. The Knowledge Engineering Review, 18(1):1–31, 2003.

Kitakami H., Mori Y., Arikawa, M..(1996) An intelligent system for integrating autonomous nomenclature databases in semantic heterogeneity. In Database and Expert System Applications, DEXA'96, number 1134 in Lecture Notes in Computer Science, 187–196, Zurich, Switzerland, 1996.

Klein M. (2001). Combining and relating ontologies: an analysis of problems and solutions. In: Proceedings of the IJCAI-20001 Workshop on Ontologies and Information Sharing, Seattle, WA.

Liebich T., Amor R., Verhoef M. (1995). A comparison of mapping methods available within the product modelling arena, Proceedings of the Fifth EXPRESS User Group Meeting, Grenoble, France, 21-22 October 1995.

Noy N. and Musen M. A. (2003). The PROMPT suite: Interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies, 59(6):983–1024.

Noy, N. (2004) Semantic Integration: A survey of ontology-based approaches. SIGMOD Record, 33(4):65–70.

Oh Y., Han S., Suh H.. (2001). Mapping product structures between CAD and PDM systems using UML, Computer-Aided Design, 33 , 521-529.,

Visser Pepijn R. S., Jones Dean M., Bench-Capon T. J. M., and Shave M. J. R.. (1997). An analysis of ontological mismatches: Heterogeneity versus interoperability. In AAAI 1997, Spring Symposium on Ontological Engineering, Stanford, USA.

Rahm E. and Bernstein P. A.. (2001). A survey of approaches to automatic schema matching. VLDB Journal, 10(4).

Shin Y, Han S-H.. (1988) Data enhancement for sharing of ship design models. Computer-Aided Design; 30(12):931±41.

ST-Developer User Guide, STEPTools, Inc., 1995.

STEP PDM Schema, PDMnet, http://www.pdmnet.org, 1998.

Stumme G. and Madche, A. (2001). FCA-Merge: Bottom-up merging of ontologies. In 7th Intl. Conf. on Artificial Intelligence (IJCAI ’01), pages 225–230, Seattle, WA.

Su X.M., (2002). A text categorization perspective for ontology mapping.Technical report, Department of Computer and Information Science. Norwegian University of Science and Technology, Norway

Tursi A., Panetto H., Morel G., Dassisti M. (2007). Ontology-based products information interoperability in networked manufacturing

enterprises. IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2 – 5, Monterrey, México, IFAC Papersonline.

Tursi A., Panetto H., Morel G., Dassisti M. (2009). Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. IFAC Annual Reviews in Control. 33/3, September, extended version from IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2–5, Monterrey, México, Elsevier, ISSN: 1367-5788 (in press).

Chapter 4: Validation of the ontological model

1. Introduction

This chapter shows how the ontological model presented and explained in chapter 4 can be instantiated on a real production system. A test case that shows how to represent product related information through its lifecycle is provided in order to give a validation to the model itself. This semi-industrial production system is provided by a local technical centre: the AIPL-PRIMECA[1] (*Atelier Inter-Établissements de Productique Lorrain*) while the design and selling centre is located at DIMeG, in Politecnico di Bari.

2. Use Case

2.1. The general context

Actually the increasing complexity on information flows on the one hand, and the distribution of the information in the whole supply chain on the other hand, had lead enterprises to use a lot of heterogeneous software applications like APS (Advanced Planning and Scheduling), ERP (Enterprise Resource Planning), MES (Manufacturing Execution System), SCM (Supply Chain Management), PDM (Product Data Management) and so on, to name only a few (Figure 33). Thus, all the enterprise systems have to interoperate to achieve global performances for the full manufacturing processes.

[1] www.aip-primeca.net/lorraine/

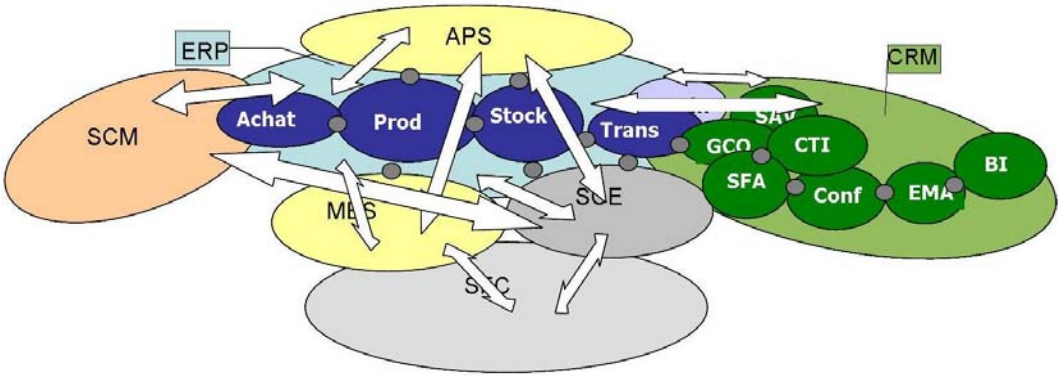


Figure 33 - Chaotic Integration for interoperability of enterprise systems (Baîna, 2006)

In (Morel et al., 2003), it is suggested and we agree that it is the customized product that must drive the interoperability relationship in the manufacturing process. In this paradigm, the product is seen as an information system that embeds the information about itself and that is able to communicate with the software applications in order to be manufactured.

2.2. The scenario

The proposed case study concerns the design and the production of a product. It is based on a set of enterprise systems, distributed on two production sites, one in Italy and one in France. The product is conceived and designed in the Department of Mechanical and Management Engineering of the Politecnico di Bari, in Bari, Italy. The definition of product is driven by market or by customer requirements and forecasting. technical and geometrical information, joined to business information, such as the required quantity of pieces, are stored in a memory chip (a RFID), and structured in the information model that implements the *Product Ontology*.

This digital product is sent to the *Atelier Inter-Établissements de Productique Lorrain* (AIPL-PRIMECA) of the Nancy-University, France (Figure 34).

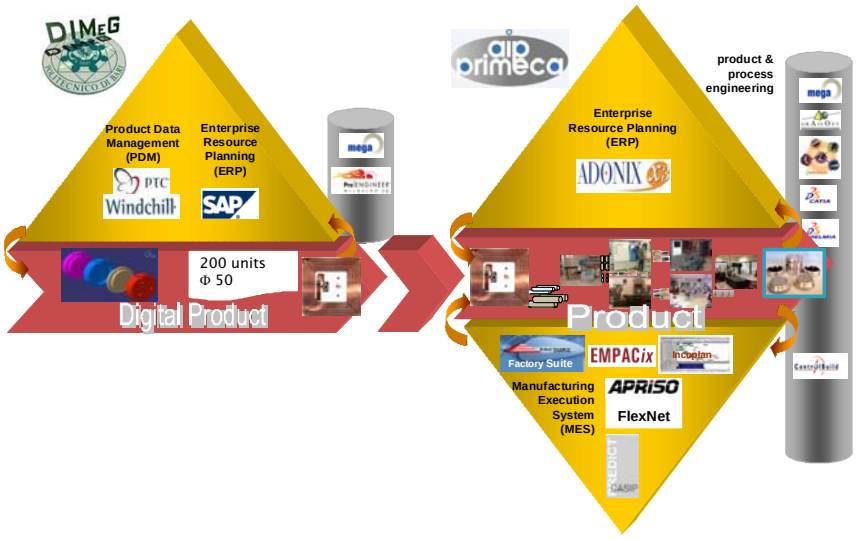


Figure 34 - The use case architecture

AIPL has to manufacture the product on the base of information drawn from the *Product Ontology*, retrieved from the chip. In this semi-industrial production system, it is possible to manufacture 4 types of base part from a product family and then assemble them in order to compose 6 types of product (Figure 35).

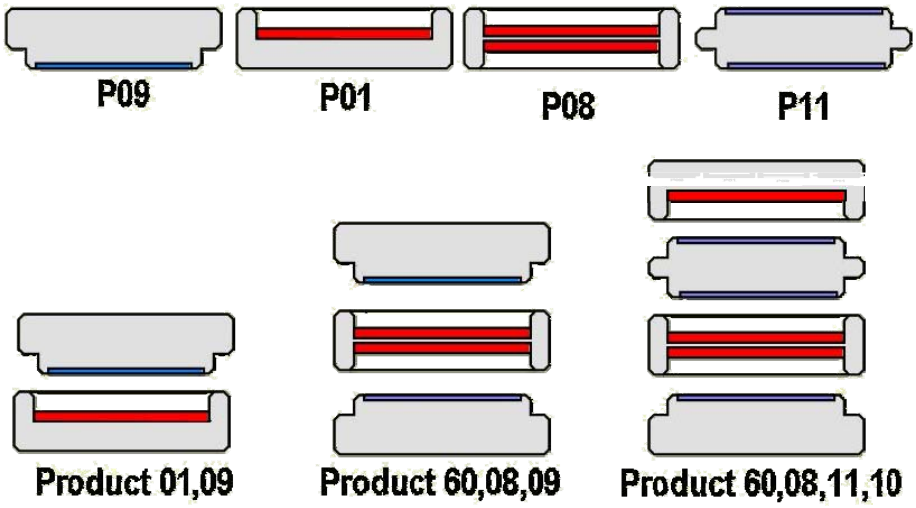


Figure 35 - Parts and some products produced at the AIPL

At the end of production process, the manufactured product will be sent to DiMeG in order to be delivered to the customer.

Each enterprise of this case study is equipped with its enterprise systems (i.e. Windchill PDM, ProEngineer CAD and SAP R/3 for DiMeG or Flexnet

MES and Sage ERP X3 for AIPL), dedicated to specific tasks (engineering tasks or manufacturing ones) and provided by a particular vendor. In this product-centric information system, these heterogeneous applications have to interoperate with the product, in order to store and to draw the pertinent product information.

Actually, the exchange of information between enterprise systems defines a sort of "*application-driven interoperability*", represented by the sequence diagram in Figure 36.

To support information exchange between the product and the many applications that interact with him, it is necessary to define a common information model, which intends to specify an embedded *Product Ontology*. The scenario in this case become "*product-centric*", as shown by the sequence diagram in Figure 37.

This common information model is following our Product ontology proposal.

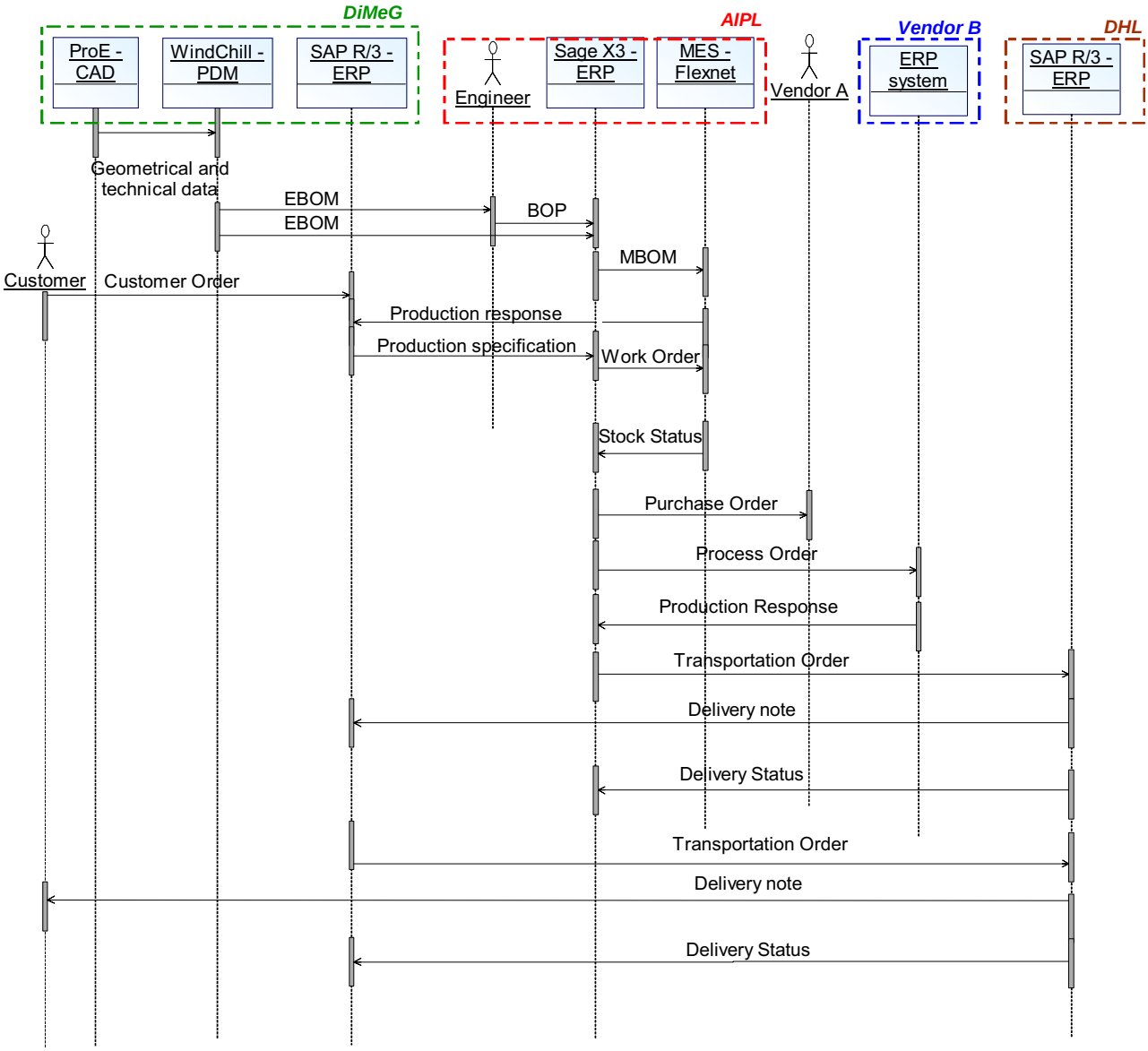


Figure 36 – Application driven interoperability scenario

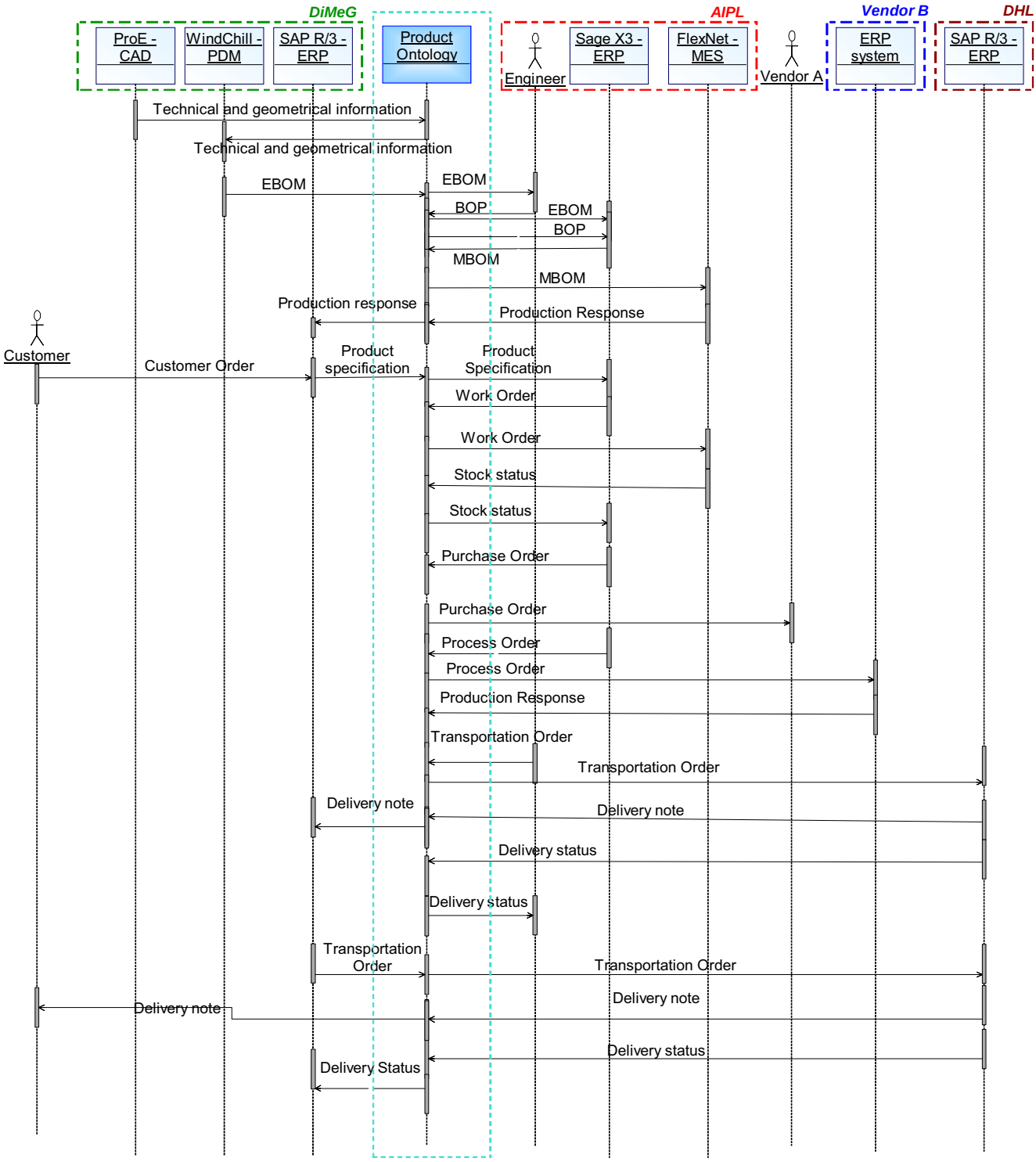


Figure 37 – Product-driven Interoperability scenario

We focus on a single part of the one of the AIPL products (Figure 35). We consider the production of P09 product and we implement the model, thus

validating that it allows the interoperability between the Product Ontology itself and the applications that interact with him.

P09 is composed by a base, which is obtained via turning of aluminium bar and by a disc, obtained via cutting a galvanized plate (Figure 38).

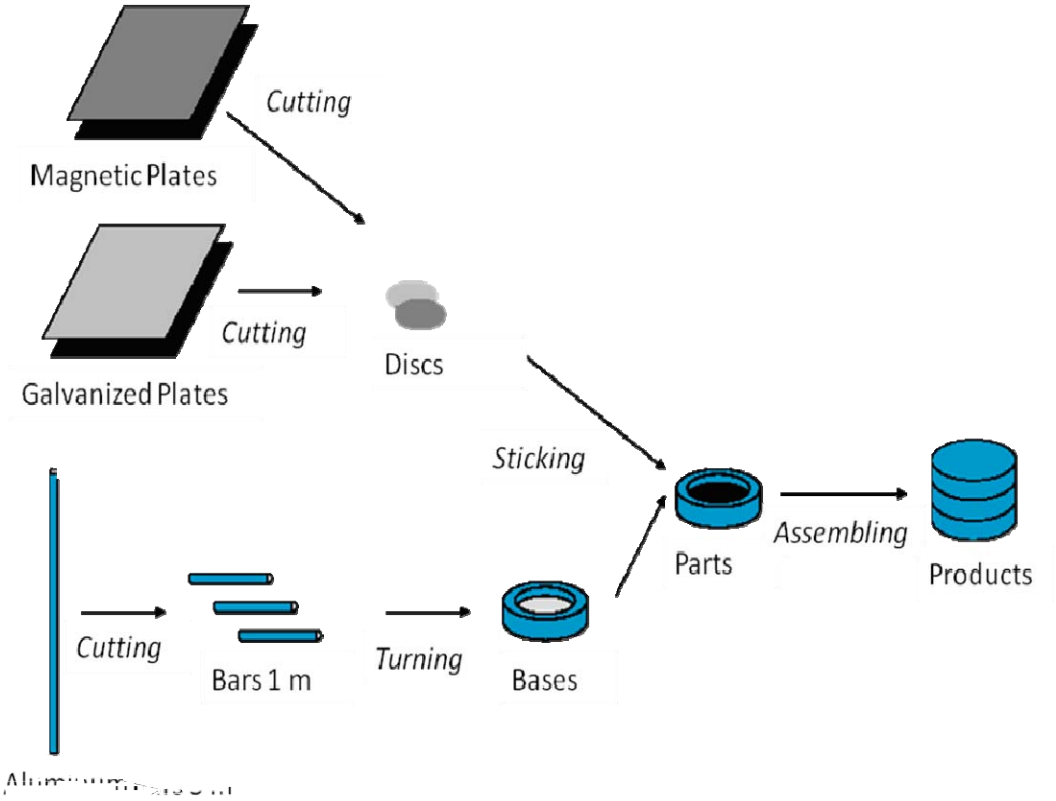


Figure 38 - Production process of P09 product

2.3. Application of proposed model

Step 1: EBOM of P09 at DIMEG

The product P09 is conceived and designed in the DIMEG. Initially, the definition of product is driven by market or forecasting by customers' needs. Technical and geometrical information will be produced in this phase, which will produce the EBOM (Engineering Bill of Material) of P09. This engineering BOM normally lists items according to their relationships with parent product as represented on assembly drawings. In practice, EBOM is usually produced automatically by CAD system: in the test case, we use ProEngineer application. The figure below represents the drawing resulting from the conception and design of all AIPL products.

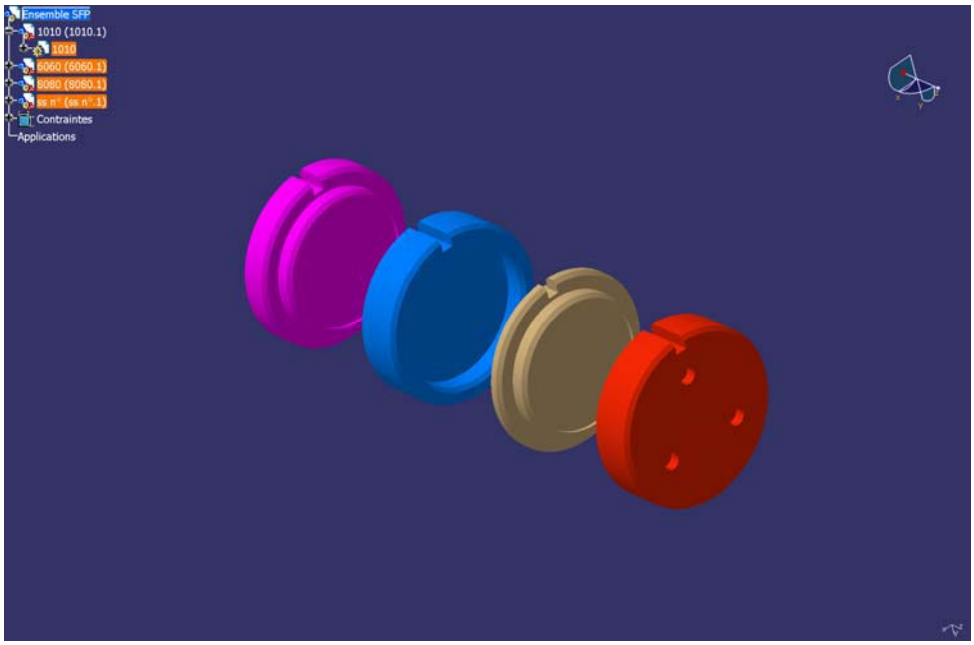


Figure 39 – AIPL products CAD model

In this phase P09 is a “Digital Product” and the technical and geometrical information about it are carried out by a PDM system, Windchild PDM application, which interfaces with ProE system. They are:

1. Relationships with component part: P09 Base and Galvanized Disc;
2. Diameter external (D) of P09 (coincident with diameter external of P09 Base)
3. Internal diameter (d) of P09 (coincident with diameter internal of P09 Base and the diameter of Galvanized Disc)
4. External height (H) of P09 (coincident with external height of P09 Base)
5. Internal Height (h) of P09 (coincident with internal height of P09 Base)

This information is stored in our Product Ontology, as shown in the Figure 40. In Figure 40, the relationships between P09 and its component parts, such as P09 Base and Galvanized Disc, are represented: one P09 product comprises one P09 Base and one Galvanized Disc. In the Figure 41, you can see part of the technical information about the galvanized

disc, such as the diameter and the material which is made up. In analogue way, the model contains the geometrical information about P09 Base.

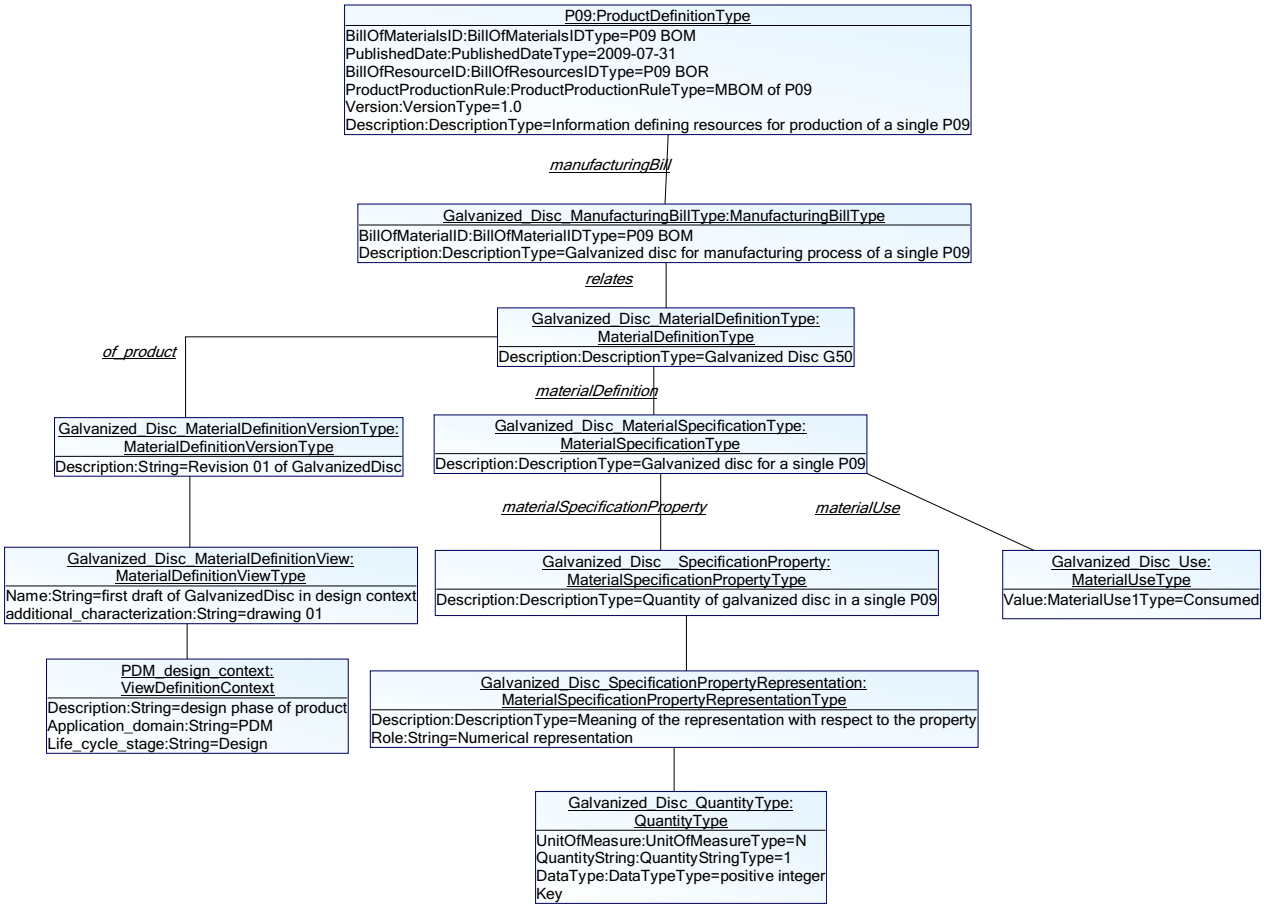


Figure 40 (a) – The EBOM in the Product Ontology

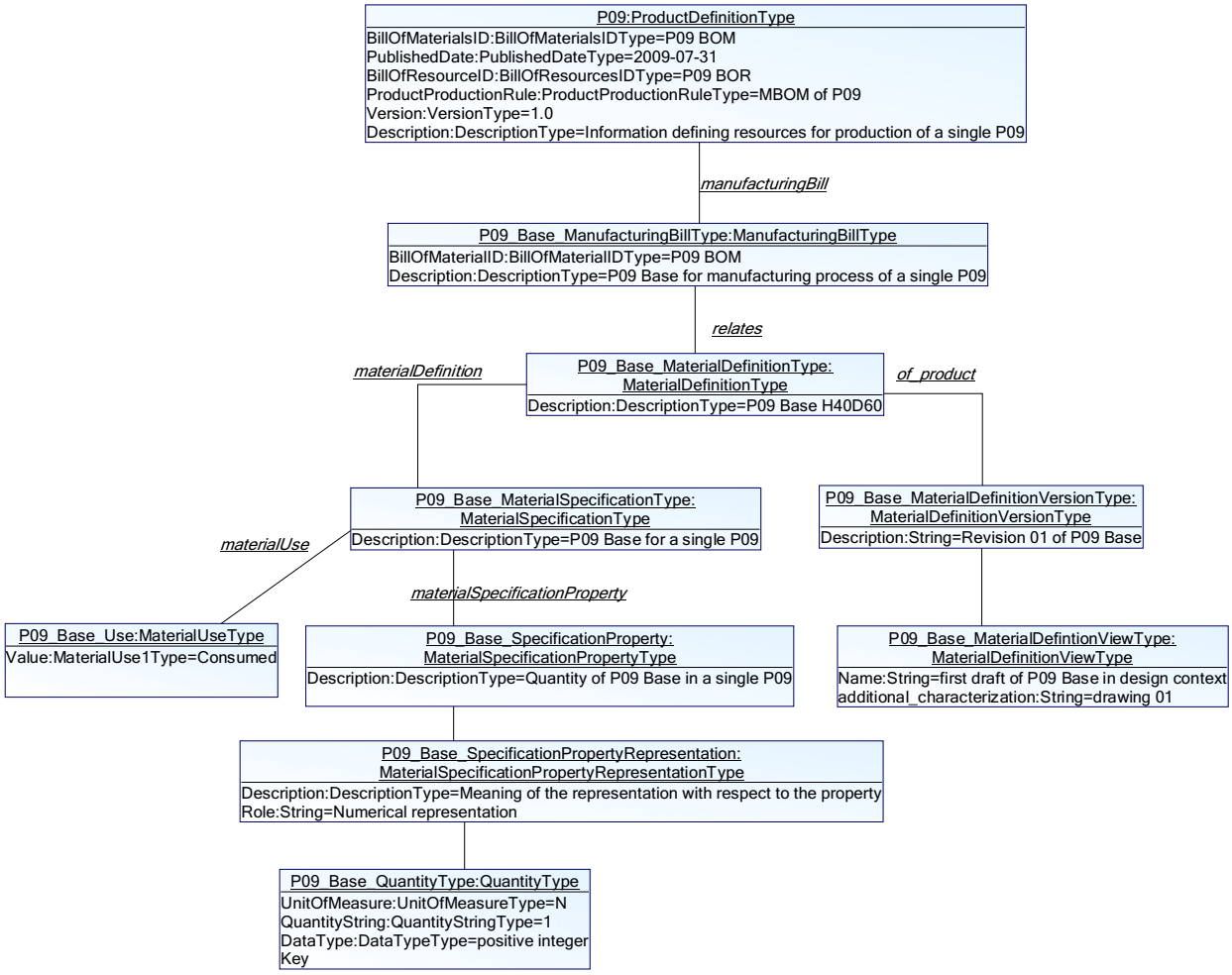


Figure 40 (b) - The EBOM in the Product Ontology

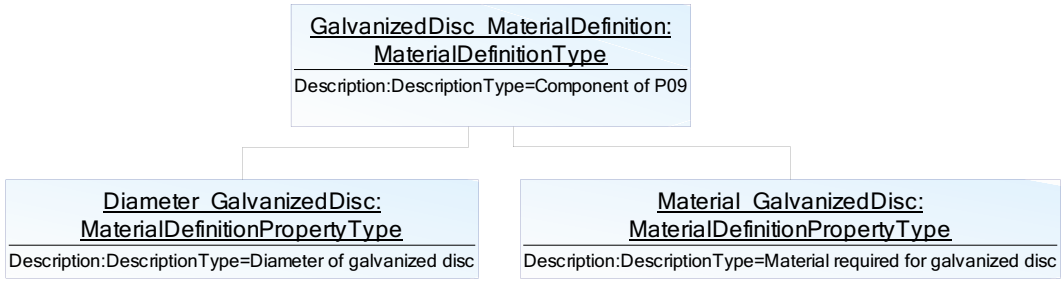


Figure 41 – Some technical information of galvanized disc in the Product Ontology

The EBOM information may not be sufficient to show the grouping of parts at each stage of the production process nor includes all of the data needed to support manufacturing or procurement. Thus, EBOM just

represents the product structure from the engineering point of view, not from the manufacturing viewpoint, and it cannot be directly used in the materials requirements planning (MRP) system. These requirements may force the arrangement of the product structure to be different in order to assure manufacturability and need a transformation from EBOM to MBOM (Manufacturing Bill of Material) (Xu et al., 2008).

Step 2: MBOM of P09 at AIPL

A manufacturing BOM (MBOM) represents the assembly build-up the way a product is manufactured. There is a close relationship between EBOM and MBOM. In practice EBOM is usually produced automatically by CAD system, and MBOM is generated with human intervention based on EBOM and complementing some manufacturing information from the *bill of process (BOP)*.

In this phase, the information related to the manufacturing process are added. To obtain the P09 product, this is the production cycle:

1. Cutting of 3 m aluminium bar in 1 m bar
2. Turning of first side of the bar
3. Turning of second side of the bar
4. Cutting galvanized plate in disc
5. Assembly (by sticking) of galvanized disc with the base

The segment processes may have dependences between them. For instance, the processes (1) and (4) are independent between them, instead (3) depends on (2) and (2) depends on (1); (5) depends on (3) and (4).

This information is also contained in the *Product Ontology*. You can see the information about the assembly process of P09 in the Figure 42. To make the assembly of P09, it is necessary that the cutting of Galvanized Disc and the turning of second side of P09 Base have to be finished.

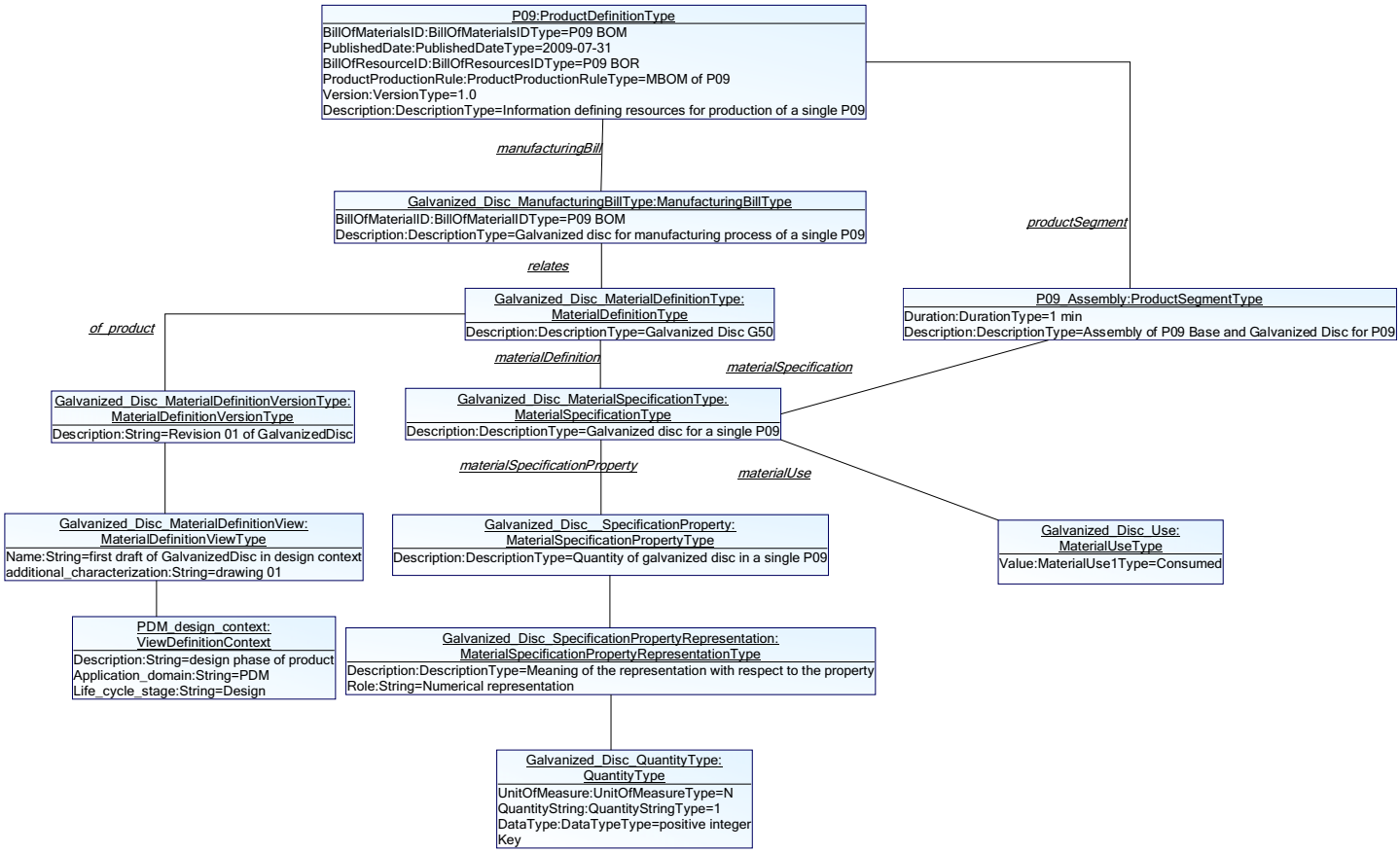


Figure 42 (a) - The MBOM in the Product Ontology

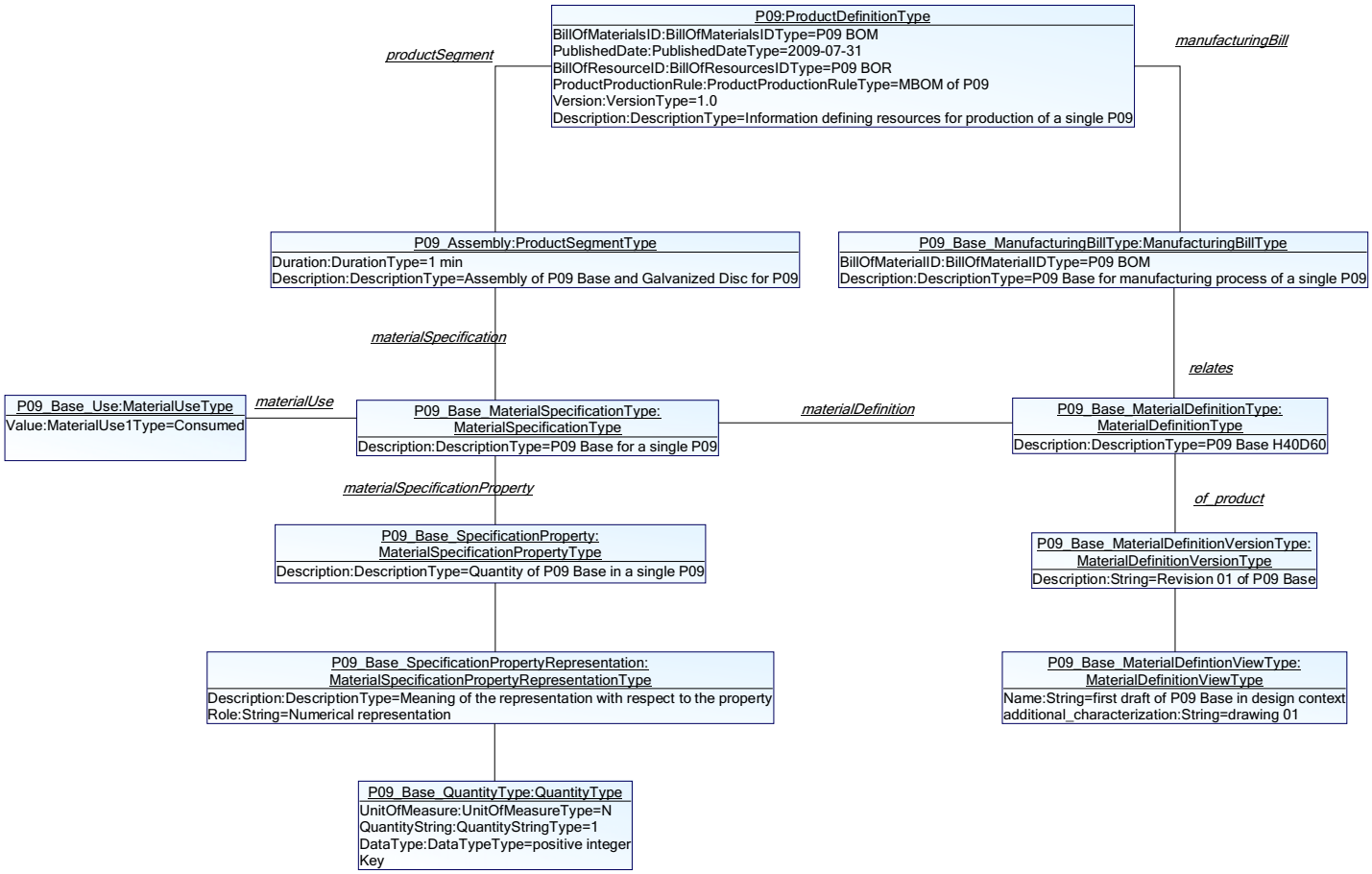


Figure 42 (b) - The MBOM in the Product Ontology

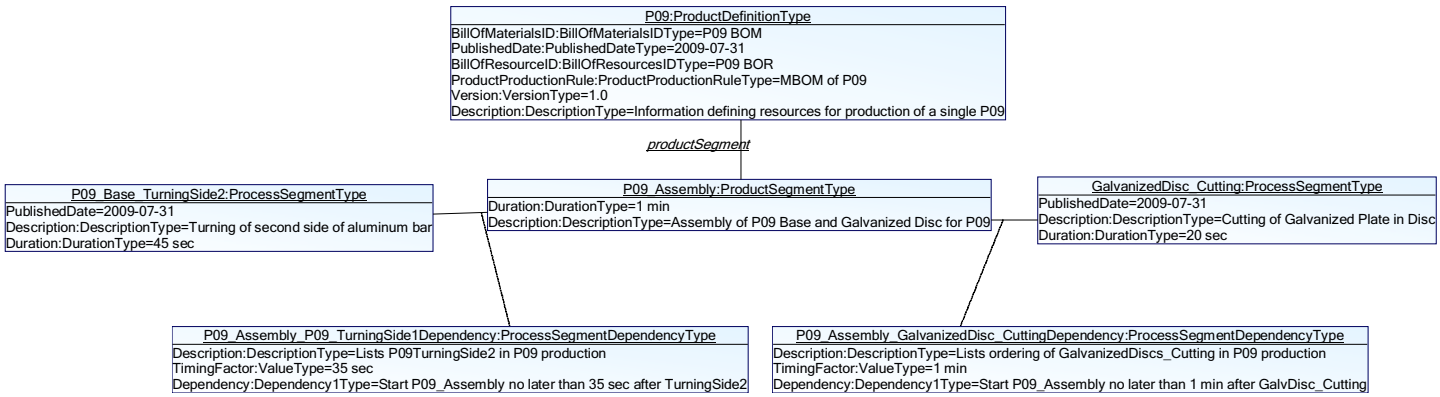


Figure 42 (c) – The MBOM in the Product Ontology

Also information about resource (machine, tool and personnel) will be defined in the model in this phase (Figure 43).

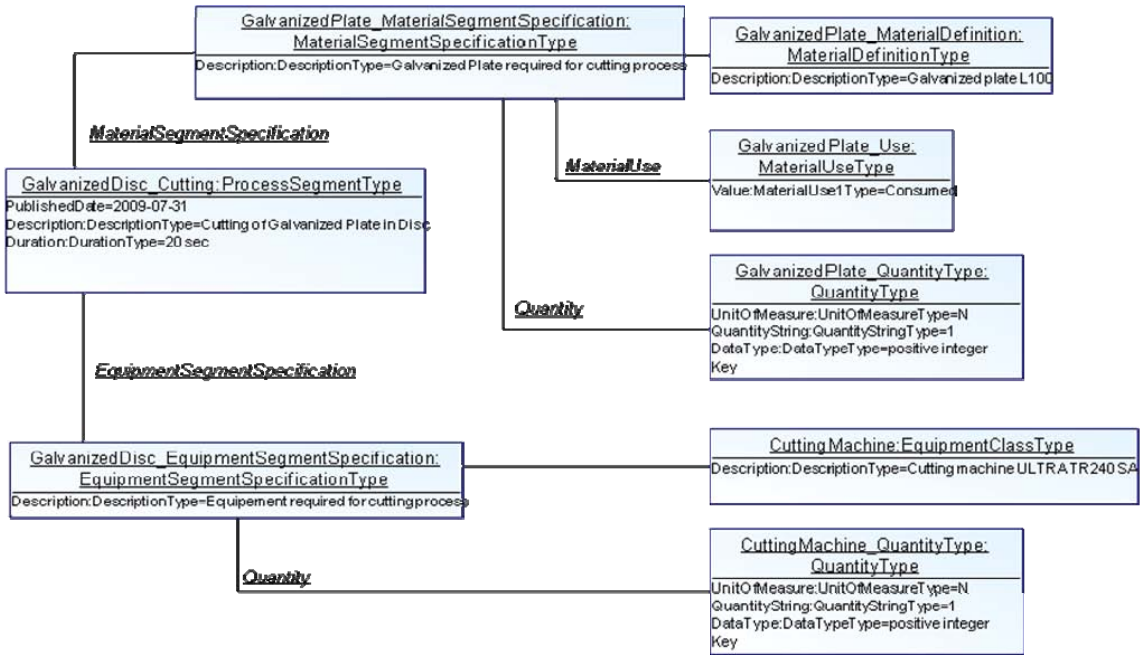


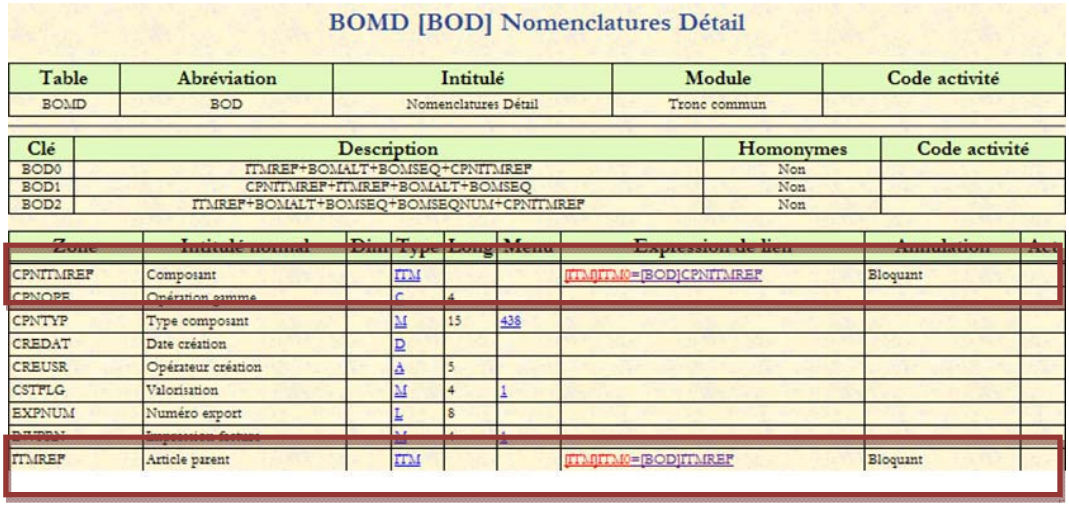
Figure 43 – The information about galvanized disc cutting process in the Product Ontology (materials and equipment)

All information are carried out by the ERP system at AIPL, the Sage ERP X3 application. They assure the integration of information flow between the applications of ERP and MES (Sage X3 and FlexNet): depending on this information, the P09 product will be manufactured.

As soon as FlexNet (the MES application) has verified the correctness of the MBOM, a production response will be registered to be retrieved back by Sage ERP X3.

In this step, the mapping between the application ERP and the Product Ontology and between the Product Ontology and the application MES can be shown, although a mediator will be useful. In fact, we can see how information about MBOM of P09, for example, is contained both in the applications models and the Product Ontology, in order to demonstrate that the mapping is possible. Indeed, in the Figure 44, you can see a data table of Sage ERP X3 data model. This table is extracted from the production module of the application and describes the details of manufactured product BOM. It is possible to remark the "*composant*"

(component) attribute that refers to the relationship with the component of P09, P09 Base and Galvanized Disc, and the "*article parent*" attribute that refers to the relationship with the product on superior level in the MBOM, whose P09 is component (see Figure 44).



BOMD [BOD] Nomenclatures Détail

Table	Abréviation	Intitulé	Module	Code activité
BOMD	BOD	Nomenclatures Détail	Tronc commun	

Clé	Description	Homonymes	Code activité
BOD0	ITMREF+BOMALT+BOMSEQ+CPNITMREF	Non	
BOD1	CPNITMREF+ITMREF+BOMALT+BOMSEQ	Non	
BOD2	ITMREF+BOMALT+BOMSEQ+BOMSEQNUM+CPNITMREF	Non	

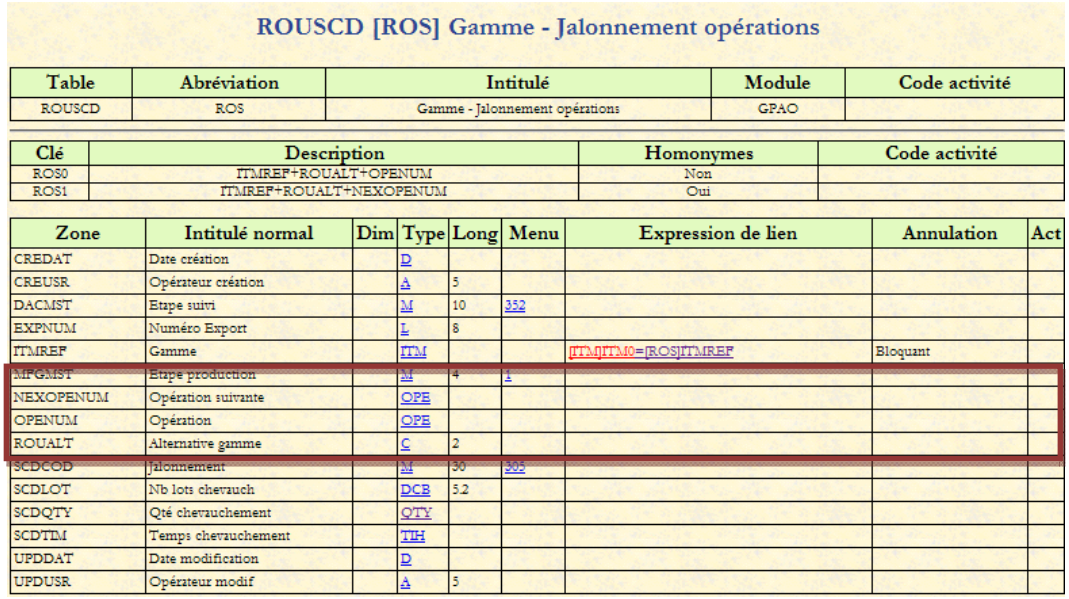
Zone	Intitulé normal	Dim	Type	Long	Menu	Expression de lien	Annulation	Act
CPNITMREF	Composant		ITM			ITM@ITM0=[BOD]CPNITMREF	Bloquant	
CPNOPE	Opération amme		C	4				
CPNTYP	Type composant		M	15	438			
CREDAT	Date création		D					
CREUSR	Opérateur création		A	5				
CSTFLG	Valorisation		M	4	1			
EXPNUM	Numéro export		L	8				
PUPDAT	Impression facture		M	4	4			
ITMREF	Article parent		ITM			ITM@ITM0=[BOD]ITMREF	Bloquant	

Figure 44 – An extract of the model MBOM implementation model in Sage X3 (1)

In the Figure 45, a table of Sage ERP X3 data model contains information about the production plan (operations needed to manufacture the product and the dependence between them).

Similarly, we can show the FlexNet data model (Figure 46), related to the Product to be manufactured. The class Product_Component related to Product expresses the relationship between P09 and its component: Galvanized Disc and P09 Base.

The Figure 47, instead, represents an extract of FlexNet data model related to the Work Order: in this model, Order_Detail contains information about the operations to follow and other details, Progress_status and Progress_Transition_status contain information about the dependence between operations.



ROUSCD [ROS] Gamme - Jalonnement opérations

Table	Abréviation	Intitulé	Module	Code activité
ROUSCD	ROS	Gamme - Jalonnement opérations	GPAO	

Clé	Description	Homonymes	Code activité
ROS0	ITMREF+ROUALT+OPENUM	Non	
ROS1	ITMREF+ROUALT+NEXOPENUM	Oui	

Zone	Intitulé normal	Dim	Type	Long	Menu	Expression de lien	Annulation	Act
CREDAT	Date création		D					
CREUSR	Opérateur création		A	5				
DACMST	Etape suivi		M	10	352			
EXPNUM	Numéro Export		L	8				
ITMREF	Gamme		ITM			[ITM]ITM0=[ROS]ITMREF	Bloquant	
MFGMST	Etape production		M	4	1			
NEXOPENUM	Opération suivante		OPE					
OPENUM	Opération		OPE					
ROUALT	Alternative gamme		C	2				
SCDCOD	Jalonnement		M	30	305			
SCDLOT	Nb lots chevauch		DCB	5.2				
SCDQTY	Qté chevauchement		QTY					
SCDTIM	Temps chevauchement		TIH					
UPDDAT	Date modification		D					
UPDUSR	Opérateur modif		A	5				

Figure 45 – An extract of MBOM implementation model in Sage X3 (2)

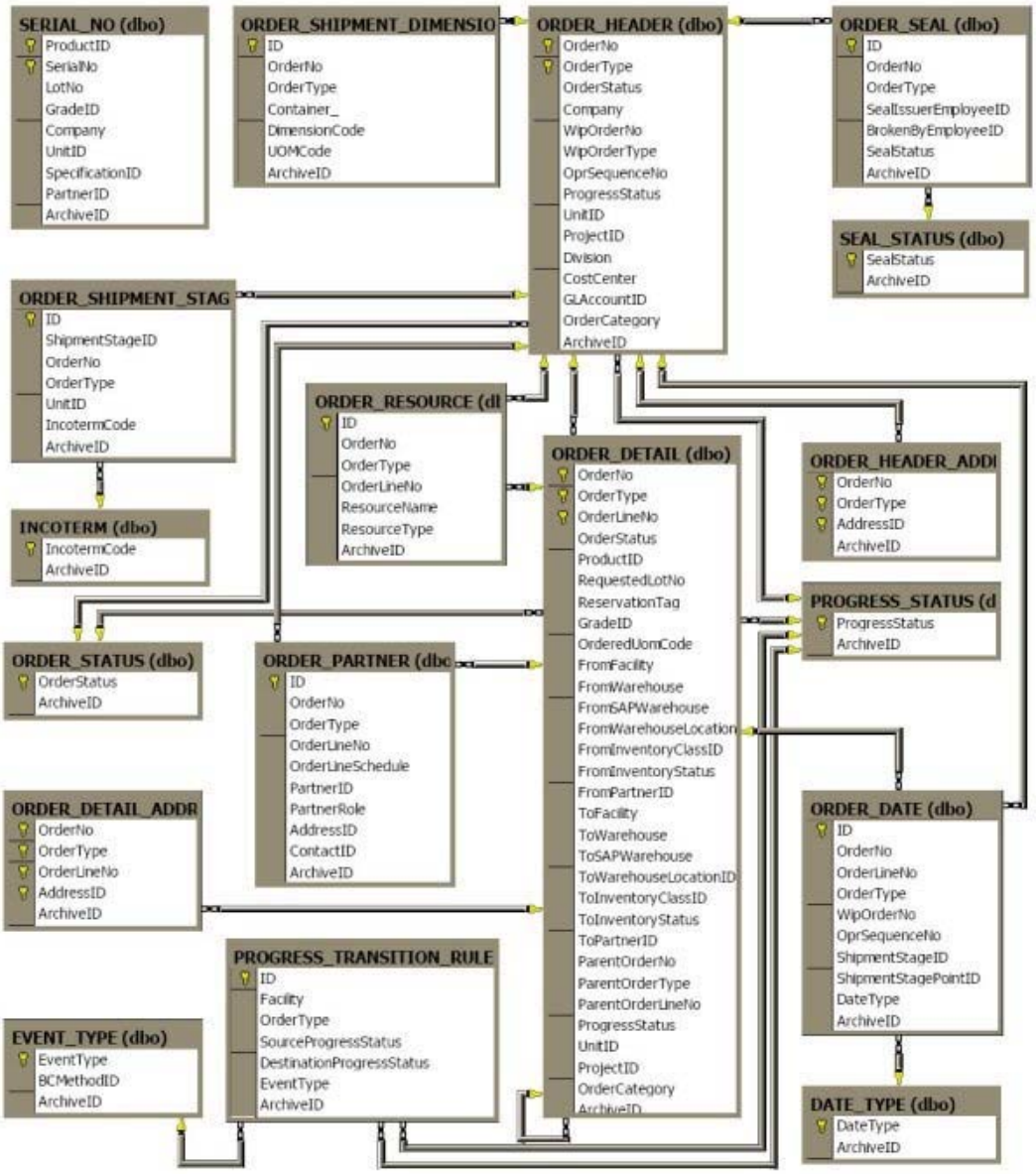


Figure 46 – An extract of the MBOM implementation model in FlexNet (1)

Thus, data models of ERP and MES applications show how the information formalized in Product Ontology is really stored in the applications and, through a mediator which allows the mapping between them, the interoperability between them may be assured.

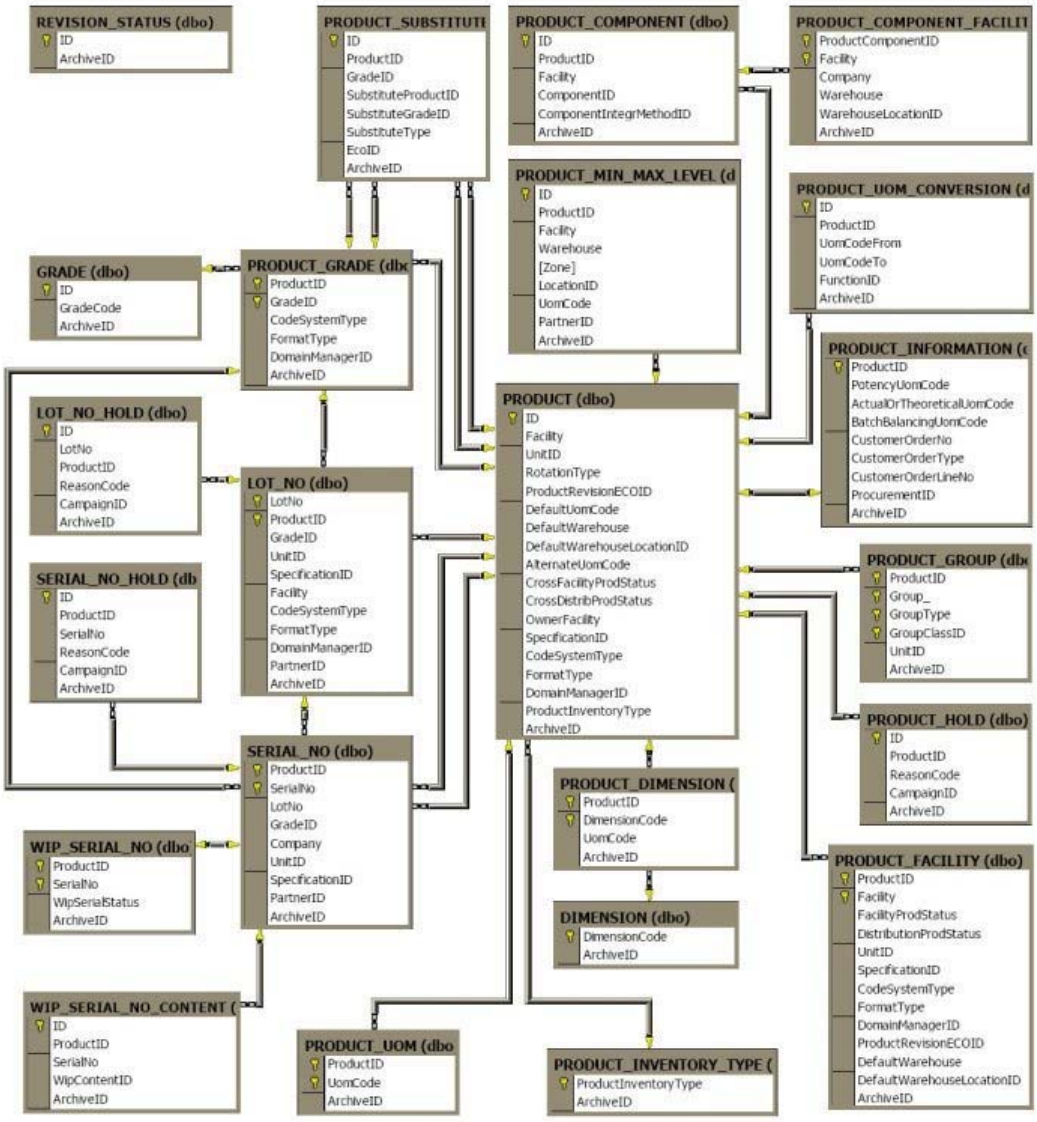


Figure 47 – An extract of the MBOM implementation model in FlexNet (2)

Step 3: Customer requirements

Until now, this exchange of information between DIMEG and AIPL systems assure the feasibility of product P09 in the manufacturing system. Let us suppose now that a customer requires P09 with specific technical characteristics and dimensions, in order to be used for a particular scope. This means that the product P09, which may be produced in various measures, have to be manufactured according to the customer requirements. The customer order is an input information for SAP R/3: all technical and geometrical values chosen by customer for the diameters and heights of P09 are stored in the *Product Ontology* by ERP SAP R/3

and are communicated to Sage ERP X3, responsible to realize the production plan.

In the Figure below (Figure 48), according to customer requirements, the galvanized disc diameter assumes as value 34 millimetres.

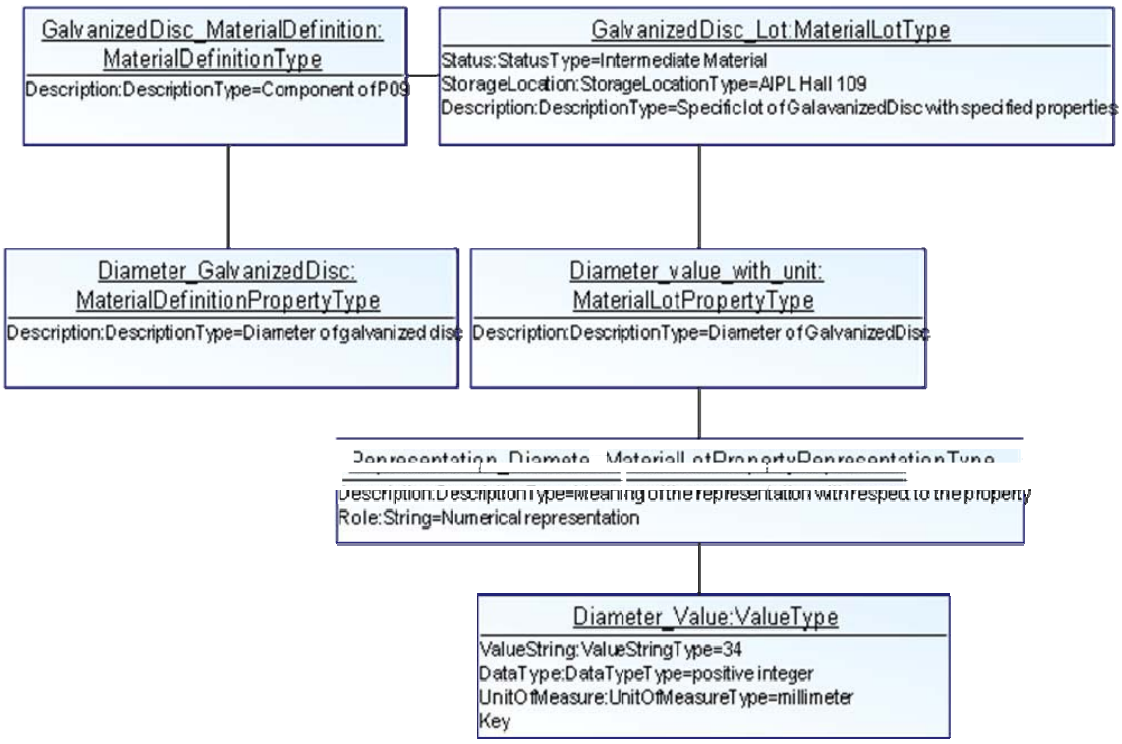


Figure 48 – Technical information about galvanized disc (diameter value)

Step 4: Supply, subcontract work and production

In the AIPL facility, the Sage ERP X3 has to communicate with the MES application, the Flexnet system. In fact, Sage ERP X3 have to communicate the Work Order to MES system, in order that this latter manufactures the product required by customer. Then, based on the stock status in the warehouse and availability of other indirect resources, communicated from FlexNet to Sage X3, the latter generates purchase requests. The purchase requests are then elaborated in purchase orders of materials and in purchase orders in subcontract work. The AIPL can have one or more vendors who realize only a phase of the production plan (for example, the galvanised plates are currently produced by a supplier

that has the needed equipment). In this latter case, the Sage ERP X3 has to interoperate with the ERP application of enterprise vendor, to give the information useful to realize the activity. For instance, the AIPL can order to another enterprise the aluminium bars of length 1 meter, useful to realize P09 base, supplying bars of length 3 meter. In other words, the AIPL provides to this enterprise vendor the materials and the production specifications to realize the cutting phase: we use the term "*process order*" to indicate this set of information.

Obviously, the purchase and work orders have to respect the delivery dates that Sap R/3 have established according to customer requirements.

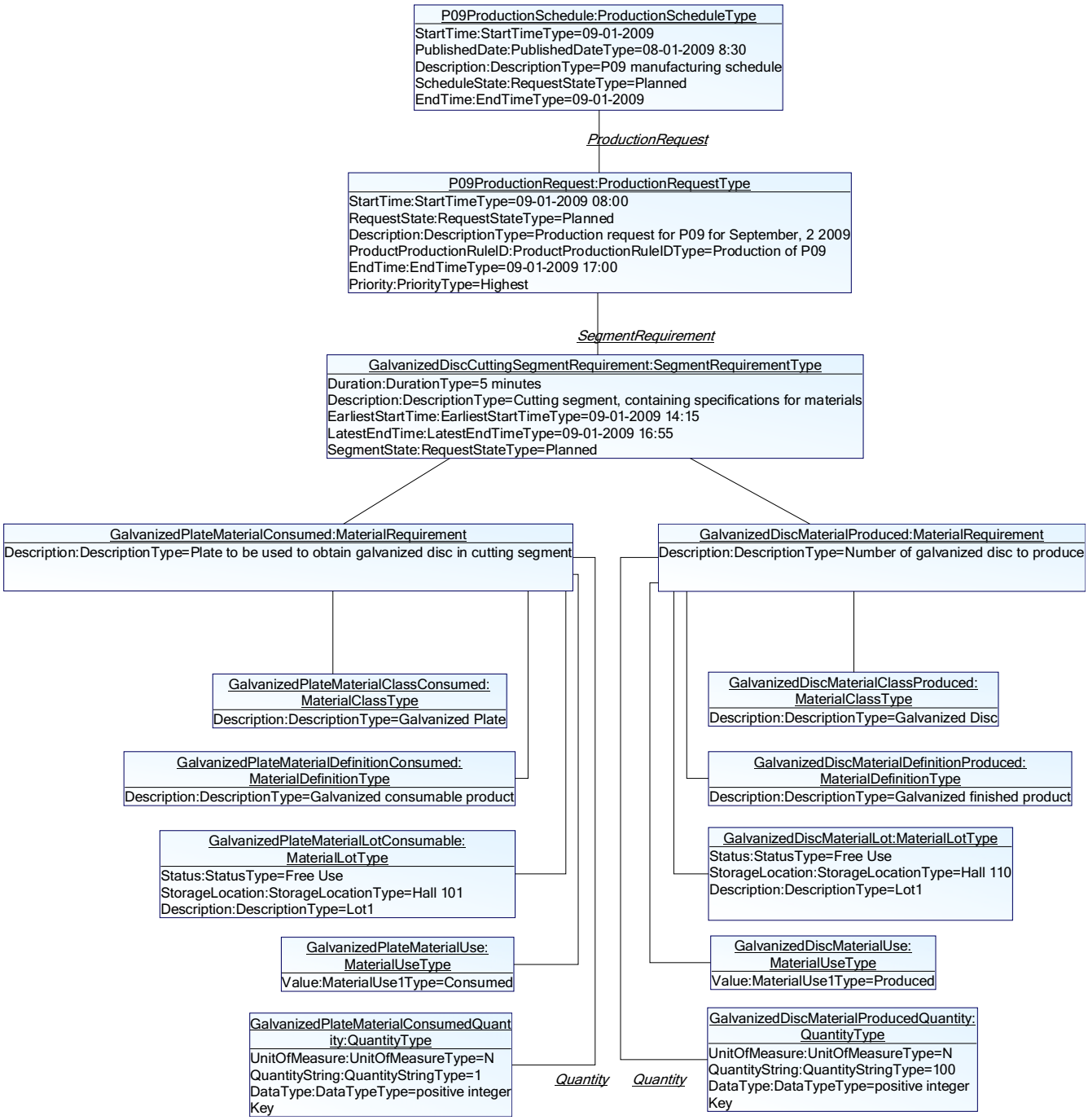


Figure 49 –Cutting segment schedule in the Product Ontology

In the Figure 49, the instantiated model contains the information about the schedule of a segment of P09 production, related to the cutting phase of galvanized plates in order to obtain the galvanized discs.

As shown previously, the information produced and exchanged in this phase are stored in *Product Ontology*. They allow to Sage ERP X3

application to interoperate with the MES one, belonging to AIPL itself and with the specific ERP system, belonging to the supplier.

In the Figure below (Figure 50), you can see the exchanged information between Sage ERP X3 and Flexnet, about the P09 material capability. Before planning the production and elaborating the purchase order, the ERP system has to know the availability of materials. In Figure 50, a lot of 100 pieces of P09, located in hall 109, is already committed for another customer order during September 2009. Depending on material capability of P09 product and its raw materials, the ERP system will issue purchase orders for raw materials.

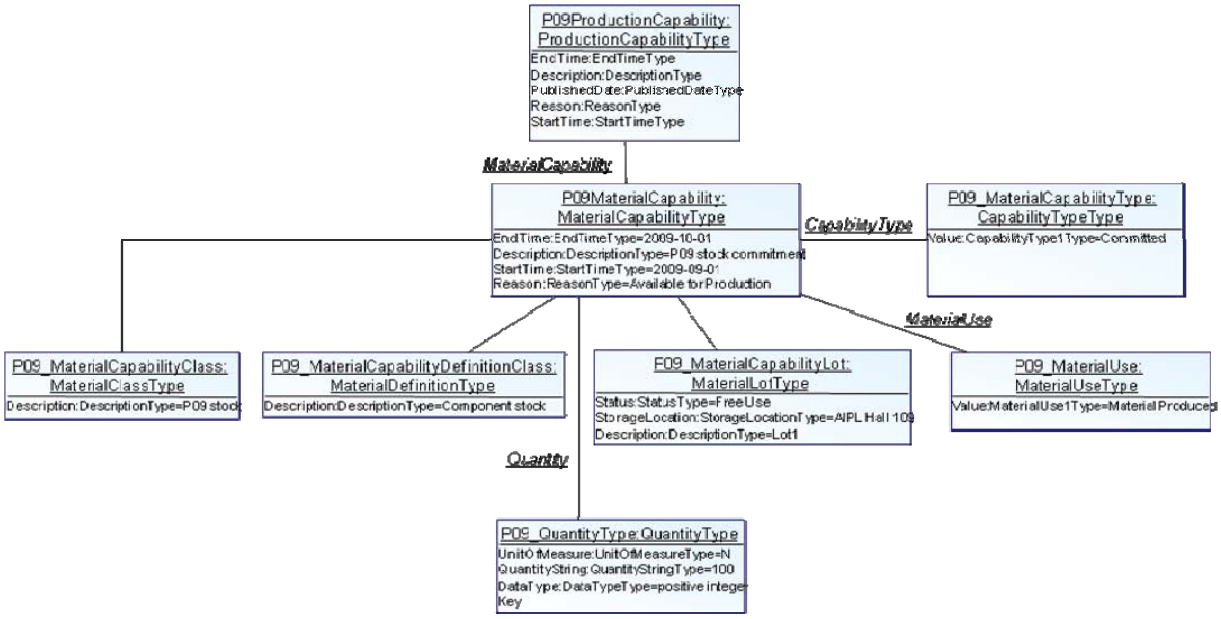


Figure 50 – Material capability for P09 in the Product Ontology

Step 5: Delivery of P09

When P09 is manufactured, it has to be delivered to the customer: the DHL enterprise, a logistic enterprise, is responsible of this delivery. At this stage, the Sage ERP X3 requests the transportation service to the SAP R/3 of DHL, in order to deliver the required lot of P09 to DiMeG site. When the delivery has been made, the DHL SAP R/3 communicates the delivery note to the DiMeG ERP and the delivery status (terminated) to the AIPL ERP. The last information flow is related to the delivery of product to the customer. The DiMeG SAP R/3 requires to DHL one a transportation

service from DiMeG site to the customer address. When the service is completed, DHL ERP communicates the delivery note to the customer, who receives his product, and the delivery status (terminated) to the DiMeG ERP.

The Figure 51 represents the information stored in the Product Ontology, related to the request of transportation service from AIPL warehouse to DiMeG warehouse through DHL facility. In this example, the transfer service is completed and actual data are also provided and stored in the Product Ontology.

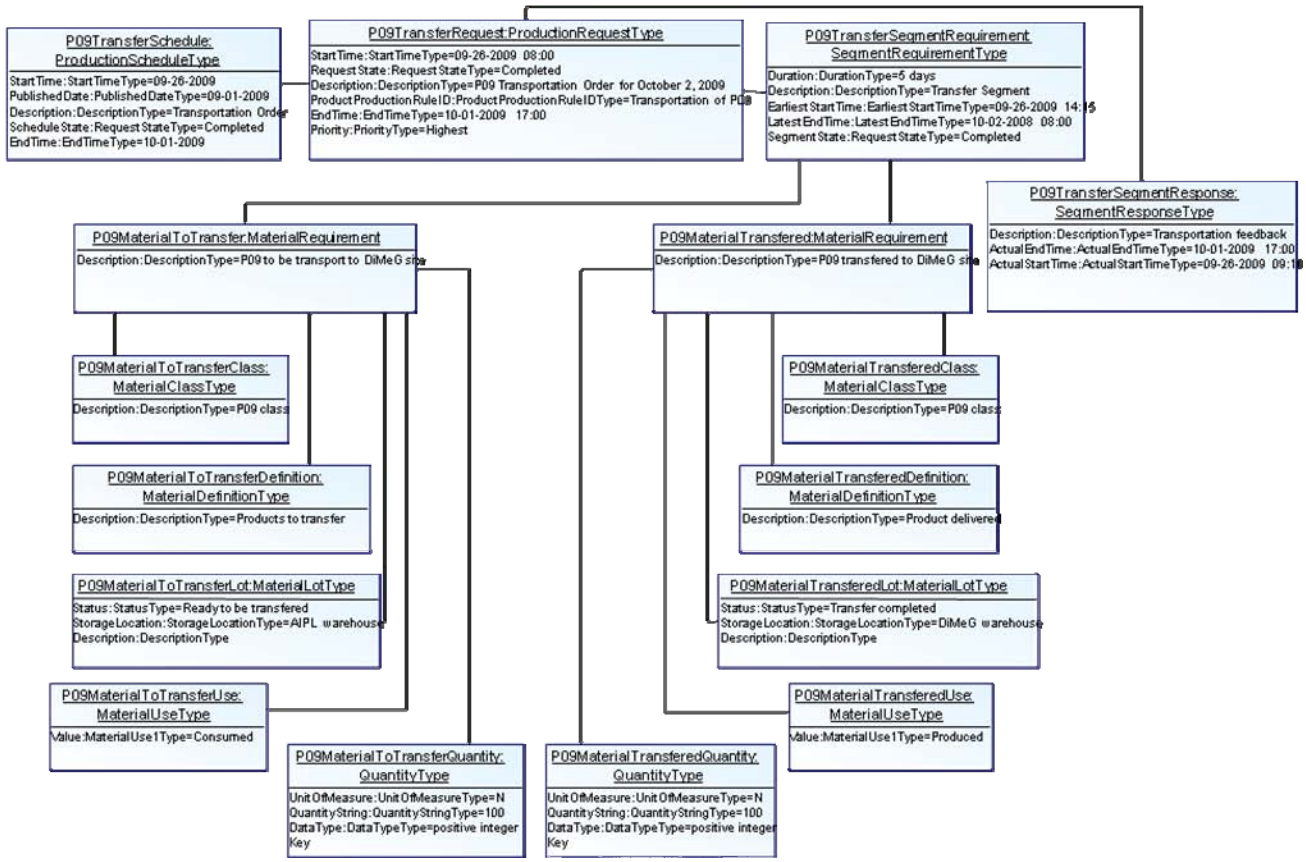


Figure 51 – Transportation information of P09 in Product Ontology

All these information, exchanged during the P09 lifecycle, are stored and are retrieved in the *Product Ontology*. Thus, the ontological model allows achieving a double objective:

1. the interoperability between enterprise systems
2. the traceability of product itself.

3. Conclusion

The chapter proposed an industrial test case to show an example of how the Product Ontology works and how it could implement the interoperation between enterprise systems. Obviously, the present use case is demonstrating the implementation of the Product Ontology, thus validating its applicability to an industrial context. However technological solutions for implementing such kind of vision might be considered but are not yet dealt in this work. Further research, mainly from a technical perspective, will be debated in the next final conclusion of the thesis.

4. References of the chapter

Morel G., Panetto H., Zaremba M.B., Mayer F.(2003): Manufacturing Enterprise Control and Management System Engineering paradigms and open issues. IFAC Annual Reviews in Control. 27/2, 199-209, Elsevier.

Xu H.C.; Xu X.F., He T., (2008). Research on Transformation Engineering BOM into Manufacturing BOM Based on BOP, Applied Mechanics and Materials. Special issue on e-Engineering & Digital Enterprise Technology. 10-12, 99-103.

Conclusions and future research

The final chapter of thesis elaborates the main conclusions of the work realized during the PhD period, in Italy and in France and the outcomes of discussions had with main experts in the frame of INTEROP-NoE Network of Excellence.

1. Summary

The aim of this thesis was to contribute to define problems of interoperability within the domain of manufacturing systems.

This kind of problems rise because of heterogeneous enterprise tools (CAD, CAM, PDM, etc.), either inside a single enterprise or among networked enterprises, due to the need to treat information in order to perform activities. This information may be stored, processed and communicated in different ways by different enterprise applications. Thus, a problem of misunderstanding can occur due to loss of information semantics may arise, when dealing with heterogeneous realities.

Starting from the consideration that the product is the common object, for which each part of the organization works, and it is the common element perceived in the same way by all manufacturing operators, it is possible to consider the product as truly interoperable per se, as far as it embeds all its technical data and information. This view reverse the common approach adopted to solve interoperability problems: if this information is structured in a common formal model, including domain rules, it can provide mappings from and to the enterprise applications, either inside a single enterprise or between networked enterprises, throughout all its life cycle.

The proposed approach in this thesis is the formalisation of knowledge and skill embedded in products and the related semantics of concepts, to support interoperability of enterprise applications.

The output is the definition of a *Product Ontology.* It contributes to solve interoperability problems between different enterprise applications.

The proposed model - which represent the Product Ontology - is based on the concepts mapping inherited from standardization initiatives related to product data management: the ISO 10303 and IEC 62264 initiatives. These standards formalise the knowledge related to products technical data and thus were selected as useful to structure the information model that can allow formalising product data and information.

The semantics of the modelling concepts, informally defined in the standards, have been formalized by First Order Logic (FOL) axioms, in order to provide a unambiguous representation of knowledge. The First Order Logic is a knowledge representation formalisms which allows modelling the application domain by defining the relevant concepts of the domain and then using these concepts to specify properties of objects and individuals occurring in the domain. The FOL is a language characterized by a formal specification of the semantics that allows expressing structured knowledge in one hand and promotes the implementation of reasoning support in the other hand. The FOL axioms, in fact, have allowed to define the mapping between concepts of STEP PDM and IEC 62264 models, in order to build a common ontological model of product information, which allow the interoperability between enterprise systems and allow to maintain the traceability of product during its lifecycle.

After the description and analysis of research domain of introductory chapters, the methodology applied to build the *Product Ontology* is explained in the chapter 3. A test case that shows how the model represents product related information through its lifecycle is provided in chapter 4: it gives a validation to the model itself and demonstrates how it can assure the interoperability between enterprise systems.

2. Limits and advantages of the proposed model

The potential main advantages of proposed model can be summarised in the following points:

1. ***Pertinence of the information structure***: The proposes model intends to specify an embedded *Product Ontology* that may be formed during the product life-cycle by the force of necessity of using it to communicate with the applications. The concept of embedding is related to the "*pertinence*" of the information structure: whenever related to the product information (technical, managerial, operational …) assumes a local (say embedded) meanly independently of the specific IT application requiring it (ERP, MES, PDM …).
2. ***Expressivity of sharing knowledge***: The *Product Ontology* provides a unique way to express sharing knowledge, in order to avoid problems of misunderstanding of information and risks of loss of information semantics.
3. ***Traceability of product***: The *Product Ontology* stores all data and information during product lifecycle, guaranteeing product traceability.
4. ***Interoperability of enterprise systems***: The main objective of our information model is to be able the product to become an active object. With such information, it may be interoperable per se with the many applications involved in manufacturing enterprises and, as far as it embeds knowledge about itself, storing all its technical data, it will be able to act as a common source of understanding between enterprises applications.

A limit of *Product Ontology* is represented by the fact that the representation mechanisms are not automatic and then can be influenced by human interpretation. It can be argued that, although it will be interesting to automate, it is still necessary the human rationality in order to take some decisions, such as the inferences and mappings.

Finally, the model is lacking of a validation in a real applications within networked enterprises, whose reality can be more complex than the test case proposed. This validation could give the cross-check of the utility of the *Product Ontology* for the interoperability of heterogeneous enterprise systems.

3. Further developments

As pointed out before, the research in this direction is still open, namely the roadmap to follow in order to achieve a full interoperability between enterprise systems.

Concerning the specific application proposed, it will be necessary to provide a formal verification of the mapping rules of *Product Ontology* (for instance using a suite of tools to construct domain ontology such as Protégé software): this can be done by applying skill-based axioms through an inference engine and using it with knowledge-based applications.

Then, other standardisation initiatives may be considered in order to have a full model that stores all technical data along the product lifecycle. We analysed standardization initiatives in the frame of ISO (ISO 10303) and IEC (IEC 62264). Applications interested by ISO 10303 standard are for example Product Data Management (PDM) systems or Computer Aided Design (CAD) systems, while applications interested by IEC 62264 standard are for example ERP systems at the business level and MES systems at the manufacturing level. Together, the two standards are covering most but not all information characterizing products and their related enterprise processes.

However, to consider products as *active objects* from a information point of view, new storing technologies, such as wireless technologies, RFID (Radio Frequency IDentification), etc., need to be developed to easily embed products information structure.

The *Product Ontology* developed proposes a centric view of the product information, acting as a pivotal element between all applications. Each

application has to be implemented, then, with a unique mediator from its own product model to the unified one (*Product Ontology*). We call this engineering process *Product-driven interoperability*.

The final objective of this approach will be to propose the standardization of the *Product Ontology.* It is a common trend of big enterprises to buy application solutions from a single software provider: they are naturally integrated. This is unfortunately not affordable for the majority of small or medium enterprise, due to increase of cost and time for the enterprise itself.

By identifying semantic gaps between information systems concepts and models and by measuring the degree of interoperability between them will help to add further hints in these directions.

Some works are currently ongoing to formalize and qualify the interoperability relationship (Yahia et al., 2009)

4. References of the chapter

Yahia E., Yang J., Aubry A., Panetto H. (2009). On the use of Description Logic for Semantic Interoperability of Enterprise Systems. On the Move to Meaningful Internet Systems: OTM 2009 Workshops (Meersman R., Tari Z., Henerro P. (Eds)), 4th IFAC/IFIP Workshop on Enterprise Integration, Interoperability and Networking (EI2N'2009), Vilamoura, Portugal, November 4-5, Springer Verlag, Lecture Notes in Computer Science, LNCS 5872, ISBN 978-3-540-88874-1.

Annex I: First Order Logic (FOL)

In this annex are reported the main aspects of First Order Logic, used in chapter 3 to formalize UML concepts of standards ontologies, in order to have the so-called “patterns of formalization”.

There are two key parts of first order logic: the syntax that determines which collections of symbols are legal expressions in first-order logic, and the semantics, that determines the meanings behind these expressions.

1. FOL: Syntax

The language of first-order logic is completely formal, so that it can be mechanically determined whether a given expression is legal. There are two key types of legal expressions: *terms*, which intuitively represent objects, and *formulas*, which intuitively express predicates that can be true or false. The terms and formulas of first-order logic are strings of symbols which together form the alphabet of the language.

It is common to divide the symbols of the alphabet into *logical symbols*, which always have the same meaning, and *non-logical symbols*, whose meaning varies by interpretation.

Logical symbols

There are several logical symbols in the alphabet, which usually include:

- The *quantifier symbols* $\forall$ and $\exists$
- The *logical connectives*: $\wedge$ for conjunction, $\vee$ for disjunction, $\rightarrow$ for implication, $\leftrightarrow$ for biconditional, $\neg$ for negation. Occasionally other logical connective symbols are included. Some authors use $\supset$ instead of $\rightarrow$ when this symbol is not available for technical reasons.
- Parentheses, brackets, and other punctuation symbols.

- An infinite set of *variables*, often denoted by lowercase letters at the end of the alphabet x, y, z, …. Subscripts are often used to distinguish variables: x_0, x_1, x_2, … .
- An *equality symbol* =

Though, it should be noted that not all of these symbols are required - only one of the quantifiers, negation and conjunction, variables, brackets and equality suffice. There are numerous minor variations that may define additional logical symbols. Sometimes the truth constants T or for "true" and F or for "false" are included.

Non-logical symbols

The non-logical symbols represent *predicates* (relations), functions and constants on the domain of discourse. It used to be standard practice to use a fixed, infinite set of non-logical symbols for all purposes.

Therefore it has become necessary to name the set of all non-logical symbols used in a particular application. This choice is made via a *signature*.

For every integer n >= 0 there is a collection of *n-ary*, or *n-place*, *predicate* symbols. Because they represent relations between n elements, they are also called *relation symbols*. For each arity n we have an infinite supply of them:

P^n_0, P^n_1, P^n_2, P^n_3, …

For every integer n>= 0 there are infinitely many *n-ary function symbols*:

f^n_0, f^n_1, f^n_2, f^n_3, …

Every non-logical symbol is of one of the following types:

1. A *predicate symbol* (or relation symbol) with some valence (or arity, number of arguments) greater than or equal to 0. These which are often denoted by uppercase letters P, Q, R,... Relations of valence 0 can be identified with propositional variables.
    a. For example, P which can stand for any statement. For example, P(x) is a predicate variable of valence 1. One possible interpretation is "x is a man".

b. Q(x, y) is a predicate variable of valence 2. Possible interpretations include "x is greater than y" and "x is the father of y".

2. A *function symbol*, with some valence greater than or equal to 0. These are often denoted by lowercase letters f, g, h ... Examples: f(x) may be interpreted as for "the father of x". In arithmetic, it may stand for "-x". Function symbols of valence 0 are called *constant symbols*, and are often denoted by lowercase letters at the beginning of the alphabet a, b, c ...

Formation rules

The formation rules define the terms and formulas of first order logic. When terms and formulas are represented as strings of symbols, these rules can be used to write a formal grammar for terms and formulas.

Terms

The set of terms is inductively defined by the following rules:

1. Variables: any variable is a term.
2. Functions: any expression $f(t_1,...,t_n)$ of n arguments (where each argument t_i is a term and f is a function symbol of valence n) is a term.

Only expressions which can be obtained by finitely many applications of rules 1 and 2 are terms. For example, no expression involving a predicate symbol is a term.

Formulas

The set of formulas (also called *well-formed formulas* or *wffs*) is inductively defined by the following rules:

1. Predicate symbols. If *P* is an n-ary predicate symbol and t_1, ..., t_n are terms then $P(t_1,...,t_n)$ is a formula.
2. Equality. If the equality symbol is considered part of logic, and t_1 and t_2 are terms, then $t_1 = t_2$ is a formula.
3. Negation. If φ is a formula, then $\neg \varphi$ is a formula.
4. Binary connectives. If φ and ψ are formulas, then $(\varphi \rightarrow \psi)$ is a formula. Similar rules apply to other binary logical connectives.

5. Quantifiers. If φ is a formula and x is a variable, then $\forall x\varphi$ and $\exists\varphi$ are formulas.

Only expressions which can be obtained by finitely many applications of rules 1–5 are formulas. The formulas obtained from the first two rules are said to be *atomic formulas*.

The role of the parentheses in the definition is to ensure that any formula can only be obtained in one way by following the inductive definition (in other words, there is a unique parse tree for each formula). This property is known as unique readability of formulas.

In a formula, a variable may occur *free* or *bound*. Intuitively, a variable is free in a formula if it is not quantified: in $\forall y.\ P(x, y)$, variable x is free while y is bound. A formula with no free variables is called a *sentence*.

2. FOL: Semantics

An *interpretation* of a first-order language assigns a denotation to all non-logical constants in that language. It also determines a domain of *discourse* that specifies the range of the quantifiers. The result is that each term is assigned an object that it represents, and each sentence is assigned a truth value. In this way, an interpretation provides semantic meaning to the terms and formulas of the language.

The domain of discourse D is a nonempty set of "objects" of some kind. Intuitively, a first-order formula is a statement about these objects; for example, $\exists x.\ P(x)$ states the existence of an object x such that the predicate P is true where referred to it. The domain of discourse is the set of considered objects. For example, one can take D to be the set of integer numbers.

The interpretation of a function symbol is a function. For example, if the domain of discourse consists of integers, a function symbol f of arity 2 can be interpreted as the function that gives the sum of its arguments.

In other words, the symbol f is associated with the function *I(f)* which, in this interpretation, is addition.

The interpretation of a constant symbol is a function from the one-element set D^0 to D, which can be simply identified with an object in D. For example, an interpretation may assign the value I(c) = 10 to the constant symbol c.

The interpretation of an n-ary predicate symbol is a set of n-tuples of elements of the domain of discourse.

This means that, given an interpretation, a predicate symbol, and n elements of the domain of discourse, one can tell whether the predicate is true of those elements according to the given interpretation. For example,

an interpretation *I(P)* of a binary predicate symbol P may be the set of pairs of integers such that the first one is less than the second. According to this interpretation, the predicate P would be true if its first argument is less than the second.

The most common way of specifying an interpretation is to specify a *structure* or *model*. The structure consists of a nonempty set D that forms the domain of discourse and an interpretation *I* of the non-logical terms of the signature. This interpretation is itself a function:

1. Each function symbol f of arity n is assigned a function *I(f)* from D^n to D. In particular, each constant symbol of the signature is assigned an *individual* in the domain of discourse.
2. Each predicate symbol P of arity n is assigned a relation *I(P)* over D^n or, equivalently, a function from D^n to {true, false}.

A formula evaluates to true or false given an interpretation, and a variable assignment μ that associates an element of the domain of discourse with each variable. The reason that a variable assignment is required is to give meanings to formulas with free variables, such as y = x. The truth value of this formula changes depending on whether x and y denote the same individual.

First, the variable assignment μ can be extended to all terms of the language, with the result that each term maps to a single element of the domain of discourse. The following rules are used to make this assignment:

1. Variables. Each variable x evaluates to μ(x)
2. Functions. Given terms t_1, …, t_n that have been evaluated to elements d_1, … d_n of the domain of discourse, and a n-ary function symbol f, the term $f(t_1, …, t_n)$ evaluates to $(I(f))(d_1, …, d_n)$

Next, each formula is assigned a truth value. The inductive definition used to make this assignment is called the T-schema.

1. Atomic formulas (1). A formula $P(t_1, …, t_n)$ is associated the value true or false depending on whether $(v_1, …, v_n) \in I(P)$, where are he evaluation of the terms t_1, …, t_n and I(P) is the interpretation of P, which by assumption is a subset of D^n.
2. Atomic formulas (2). A formula $t_1 = t_2$ is assigned true if t_1 and t_2 evaluate to the same object of the domain of discourse.
3. Logical connectives. A formula in the form $\neg\varphi$, $\varphi \rightarrow \psi$, etc. is evaluated according to the truth table for the connective in question, as in propositional logic.
4. Existential quantifiers. A formula $\exists x. \varphi(x)$ is true according to M and μ if there exists an evaluation μ' of the variables that only differs from μ regarding the evaluation of x and such that φ is true according to the interpretation M and the variable assignment μ'. This formal definition captures the idea that $\exists x. \varphi(x)$ is true if and only if there is a way to choose a value for x such that $\varphi(x)$ is satisfied.
5. Universal quantifiers. A formula $\forall x. \varphi(x)$ is true according to M and μ if $\varphi(x)$ is true for every pair composed by the interpretation M and some variable assignment μ' that differs from μ only on the value of x. This captures the idea that $\forall x. \varphi(x)$ is true if every possible choice of a value for x causes $\varphi(x)$ to be true.

If a formula does not contain free variables, and so is a sentence, then the initial variable assignment does not affect its truth value. In other words, a

sentence is true according to M and μ if and only if is true according to M and any other variable assignment μ'.

There is a second common approach to defining truth values that does not rely on variable assignment functions. Instead, given an interpretation M, one first adds to the signature a collection of constant symbols, one for each element of the domain of discourse in M; say that for each d in the domain the constant symbol c_d is fixed. The interpretation is extended so that each new constant symbol is assigned to its corresponding element of the domain. One now defines truth for quantified formulas syntactically, as follows:

1. Existential quantifiers (alternate). A formula ∃x. φ(x) is true according to M if there is some *d* in the domain of discourse such that φ(c_d) holds. Here φ(c_d) is the result of substituting c_d for every free occurrence of x in φ.
2. Universal quantifiers (alternate). A formula ∀x. φ(x) is true according to M if, for every *d* in the domain of discourse, T φ (c_d) is true according to M.

This alternate approach gives exactly the same truth values to all sentences as the approach via variable assignments.

If a sentence φ evaluates to True under a given interpretation M, one says that M satisfies φ; this is denoted M ⊨ φ. A sentence is *satisfiable* if there is some interpretation under which it is true.

A formula is logically *valid* (or simply valid) if it is true in every interpretation.

A formula φ is a logical consequence of a formula ψ if every interpretation that makes ψ true also makes φ true. In this case one says that φ is logically implied by ψ.

3. References of the chapter

Avigad J., Donnelly K., Gray D., Raff P. (2007). A formally verified proof of the prime number theorem, ACM Transactions on Computational Logic, v. 9 n. 1. doi:10.1145/1297658.1297660.

Barwise J. (1977). An introduction to first-order logic, in Barwise, Jon, ed. (1982), Handbook of Mathematical Logic, Studies in Logic and the Foundations of Mathematics, Amsterdam: North-Holland, ISBN 978-0-444-86388-1.

Barwise J., Etchemendy J. (2000). Language Proof and Logic. Stanford, CA: CSLI Publications (Distributed by the University of Chicago Press).

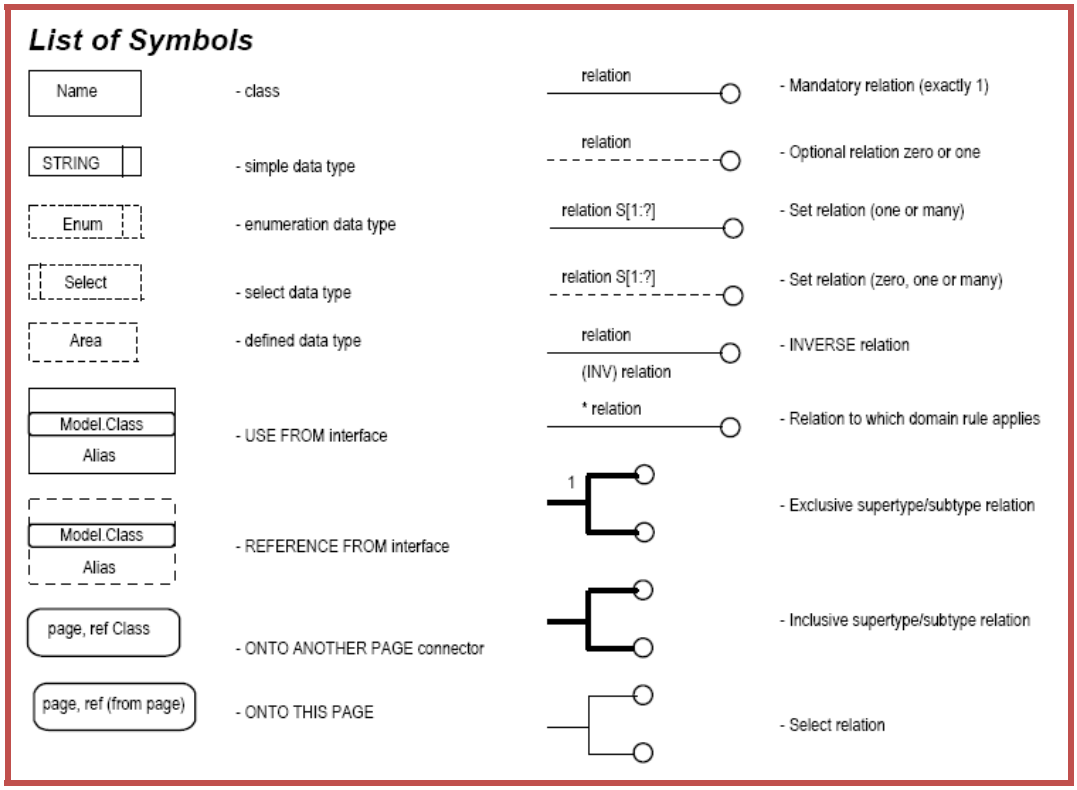
Ferreirós J. (2001). The Road to Modern Logic-An Interpretation, Bulletin of Symbolic Logic 7 (4): 441–484, doi:10.2307/2687794 . JStor.

Hilbert D., Ackermann W. (1950). Principles of Mathematical Logic (English translation). Chelsea.

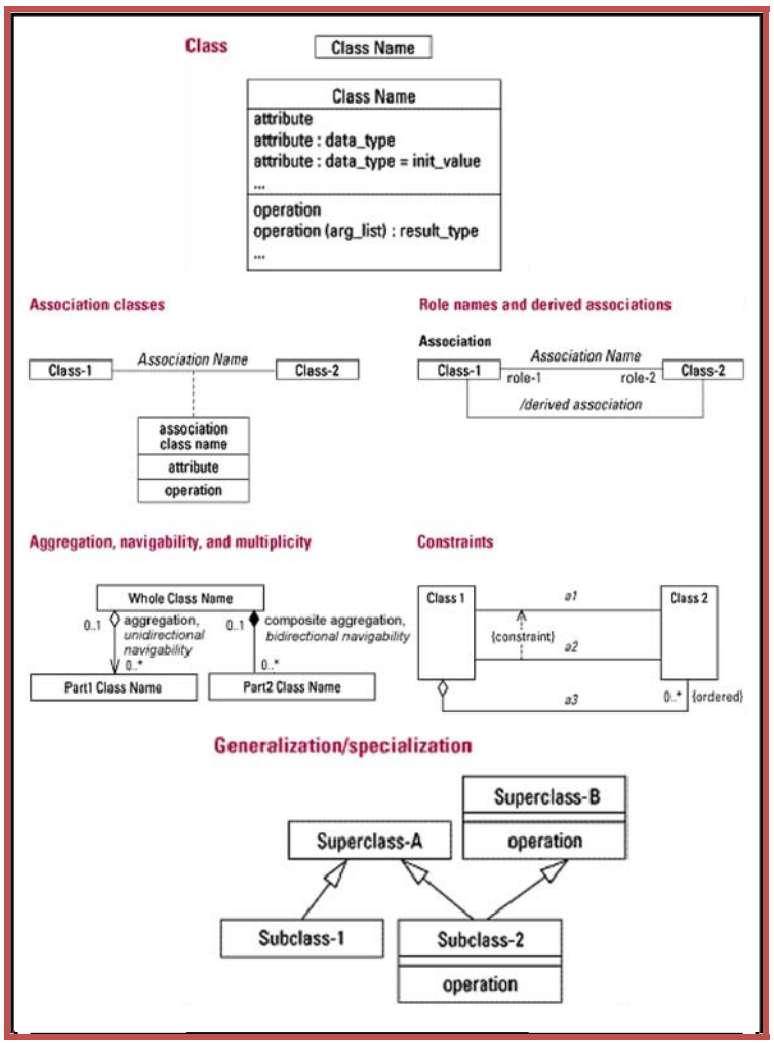
Hodges W. (2001). Classical Logic I: First Order Logic, in Lou Goble, ed., The Blackwell Guide to Philosophical Logic. Blackwell.

Ebbinghaus HD., Flum J., Thomas W. (1994), Mathematical Logic, Undergraduate Texts in Mathematics (2nd ed.), Berlin, New York: Springer-Verlag, ISBN 978-0-387-94258-2.

Annex II: Data Modelling Using EXPRESS-G



Annex III: Data Modelling Using UML



List of the references

Here all the consulted references are reported. Scientific references are publications like scientific books, journals, conferences. Technical references are guidebooks, standards and websites.

Scientific references

Avigad J., Donnelly K., Gray D., Raff P. (2007). A formally verified proof of the prime number theorem, ACM Transactions on Computational Logic, v. 9 n. 1. doi:10.1145/1297658.1297660.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. F. (2003) The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press.

Baîna S. (2006). Interoperabilité Dirigée par les Modèles: Une Approche Orientée Produit pour l'interopérabilité des systèmes d'entreprise. Thèse de doctorat de l'Universitè Henri Poincarè. December 2006, Nancy I, France (in french).

Baîna S., Morel, G. (2006). Product centric holons for synchronisation and interoperability in manufacturing environments, in: Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing. INCOM'06, May 17–19, 2006, St. Etienne, France.

Baîna S., Panetto H., Morel G. (2009). New paradigms for a product oriented modelling: Case study for traceability, in: Special issue on Intelligent Products. Computers In Industry, 60/3, 172-183, April, Elsevier, ISSN: 0166-3615.

Barwise J. (1977). An introduction to first-order logic, in Barwise, Jon, ed. (1982), Handbook of Mathematical Logic, Studies in Logic and the Foundations of Mathematics, Amsterdam: North-Holland, ISBN 978-0-444-86388-1.

Barwise J., Etchemendy J. (2000). Language Proof and Logic. Stanford, CA: CSLI Publications (Distributed by the University of Chicago Press).

Berardi, D., Calvanese, D., De Giacomo, G. (2005). Reasoning on UML class diagrams, Artificial Intelligence 168 (1–2) 70–118.

Bermundez J., Goni A., Illaramendi A., Bagues M.I. (2007). Interoperation among agent-based information systems through a communication acts ontology, Inf. Syst. 32 (8) ,1121–1144.

Berners-Lee T., Hendler J., Lassila O.(2001), The Semantic Web, Scientific American, May.

Botta V. -Genoulaz, P.- Millet A., Grabot B. (2005). A survey on the recent research literature on ERP systems, Comput. Ind. (56) , 510–522.

Boulanger D., Dubois G., (1998). An object approach for information system cooperation, Inf. Syst. 23 (6) , 383–399.

Breitman K.K., Casanova M.A., Truszkowski W., (2007). Semantic Web: Concepts, Technologies and Applications. Springer Verlag

Brickley D, Guha RV. Resource Description Framework (RDF) Schema Specification 1.0. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

Bussler C. (2003). B2B Integration: Concepts and Architecture, Springer.

Calvanese, D., Lenzerini, M., and Nardi, D. (1998). Description logics for conceptual data modelling. In J. Chomicki and G. Saake, editors, Logics for Databases and Information Systems, pages 229–264. Kluwer Academic Publisher.

Chen D, Doumeingts G. (2003). Basic concepts and approaches to develop interoperability of enterprise applications, in: Camarinha-Matos LM, Afsarmanesh H, editors. In proceedings of IFIP TC5/WG5.5 fourth working conference PRO-VE'03, processes and foundations for virtual organisations, p. 323–30, October 29–31, 2003, Lugano, Switzerland, Dordecht: Kluwer; . ISBN 1-4020-7638-X.

Chen D.v Vernadat F.B. (2002). Enterprise Interoperability: a standardisation view. IFIP International Conference on Enterprise Integration and Modelling Technology (ICEIMT'02), pp. 273-282,

Kluwer Academics Publisher, Valencia, Spain, 24-26 April 2002, ISBN 1-4020-7277-5.

Chen D., Doumeingts G., (2003). European initiatives to develop interoperability of enterprise applications—basic concepts, framework and roadmap, Annual Reviews in Control 27, 153–162.

Dartigues C. (2003). Product data exchange in a cooperative environment, Ph.D. dissertation, Univ. of Lyon 1, Lyon, France.

Dassisti M., Panetto H., Tursi A., De Nicolò M. (2008). Ontology-Based Model for Production-Control Systems Interoperability. Proceedings of the 5th International CIRP Conference on Digital Enterprise Technology, 22-24 October 2008, Nantes, France.

David Hilbert and Wilhelm Ackermann 1950. Principles of Mathematical Logic (English translation). Chelsea. The 1928 first German edition was titled Grundzüge der theoretischen Logik.

Dieter F.. (2000). Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. SpringerVerlag, Berlin

Ding Y., Fensel D., Klein M., Boris Omelayenko (2001). Ontology management: Survey, requirements and directions. Deliverable 4. IST Project IST-1999-10132.

Ding Y., Fensel D., Omelayenko B., Klein M. (2002). The semantic web: yet another hip? DKE, 6(2-3):205–227.

Doan A., Madhavan J., Domingos P., Halevy A.. (2002). Learning to map between ontologies on the semantic web. In The Eleventh International WWW Conference, Hawaii, US.

Dogac A., Laleci G.B., Kirbas S., Kabak Y., Sinir S.S., Yildiz A., Gurcan Y.(2006). Artemis: deploying semantically enriched Web services in the healthcare domain, Inf. Syst. 31, 321–339.

Domınguez E., Zapata M. A. (2000). Mappings and interoperability: a meta- odelling approach, in: T. Yakhno (Ed.), Proceedings of ADVIS 2000, LNCS 1909, 352–362, Springer, 2000.

Doumeingts G., Chen D. (2003). Interoperability of enterprise applications and software—An European IST Thematic Network Project: IDEAS e2003 eChallenges Conference, October 22–24, 2003, Bologna, Italy.

Ebbinghaus HD., Flum J., Thomas W. (1994), Mathematical Logic, Undergraduate Texts in Mathematics (2nd ed.), Berlin, New York: Springer-Verlag, ISBN 978-0-387-94258-2.

Ehrig M., Sure Y. (2004). Ontology Mapping - An Integrated Approach. The Semantic Web: Research and Applications. Vol 3053, 76-91, ISBN978-3-540-21999-6.

Euzenat J. (2001). Towards a principled approach to semantic interoperability, CEUR Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing, Vol.47., 19-25, , August 4-5, 2001, Seattle, USA, http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-47, ISSN 1613-0073.

Euzenat J., Valtchev P.(2004). Similarity-based ontology alignment in OWL-Lite. In The 16th EuropeanConference on Artificial Intelligence (ECAI-04), Valencia, Spain.

Exff project (2003). The exff EXPRESS Guide. Release 0.1. http://exff.sourceforge.net.

Farquhar A., Fikes R., and Rice J., (1997). The Ontolingua Server: A Tool for Collaborative Ontology Construction, International Journal of Human-Computer Studies, 46:707-728.

Fensel D., Groenboom R. (1997). Specifying Knowledgebased Systems with Reusable Components. In Proceedings 9th International Conference on Software Engineering and Knowledge Engineering (SEKE ’97), Madrid.

Ferreirós J.. (2001). The Road to Modern Logic-An Interpretation, Bulletin of Symbolic Logic 7 (4): 441–484, doi:10.2307/2687794 . JStor.

Fowler M. (1997). UML distilled. USA, Addison-Wesley, ISBN 0-201-32563-2.

Fridman-Noy N., Hafner C.D.. (1997). The State of the Art in Ontology Design, AI Magazine, 18(3):53-74.

Ganter B., Wille R. (1999) Formal Concept Analysis: Mathematical foundations. Springer, Berlin - Heidelberg.

Gehre A., Katranuschkov P., Stankovski V., Scherer R.J. (2005). Towards semantic interoperability in virtual organisations, in: Proc. 22nd Conference of Information Technology in Construction, cibW78, July 18–21, Dresden, Germany.

Genesereth M. R. (1991). Knowledge Interchange Format. In Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91), J. Allenet al., (eds), Morgan Kaufman Publishers, 238-249. See also http://logic.stanford.edu/kif/kif.html.

Genesereth M., Fikes R. (1992) Knowledge Interchange Format (Tech. Rep. Logic-92-1). Stanford Univ., Stanford, CA. [Online]. Available: http://www-ksl.stanford.edu/knowledge-sharing/kif/#manual

Grangel R., Bourey J. P., Berre A. (2006). Solving Problems in the Parametrisation of ERPs using a Model-Driven Approach, in: Proceedings of the 2nd IFAC/IFIP International conference on Interoperability for Enterprise Applications and Software (I-ESA'2006), Enterprise Interoperability, 461-472, March 22-24, 2006, Bordeaux, France, Springer Verlag, ISBN 978-1-84628-713-8.

Grosso W. E., Eriksson H., Fergerson R. W., Gennari J. H., Tu S. W., Musen M. A. (1999). Knowledge Modeling at the Millennium (The Design and Evolution of Protégé-2000). In Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99), Banff, Alberta, Canada, October 16-21.

.Gruber T.R. (1993). 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.), Kluwer Academic Publishers.

Gruninger M., Lee J. (2002). Ontology — applications and design, in: Communications of the ACM 45/2, 39–41.

Guarino N. (1998). Formal ontology and information systems. In: Guarino N, editor. Formal ontology and information systems. IOS Press.

Guarino N., (1998). Formal ontology in information systems. In Proceedings of formal ontology in information systems (FOIS'98) (pp. 3–15). Trento, Italy.

Guo M., Li S., Dong J., Fu X., Hu Y., Yin Q. (2003). Ontology-based product data integration, in: Proc. 17th International Conference on Advanced Information Networking and Applications;, 530–533, ISBN 0-7695-1906-7.

Hameed A, Preece A, Sleeman D (2003). Ontology reconciliation, in: Staab S, Studer R (eds) Handbook on ontologies in information systems. Springer, Berlin Heidelberg New York, 231–250.

Hegge, H. (1995) Intelligent Product Descriptions for bussiness applications. Thesis Eindhoven. Eindhoven University of Technology.

Hilbert D., Ackermann W. (1950). Principles of Mathematical Logic (English translation). Chelsea.

Hodges W. (2001). Classical Logic I: First Order Logic, in Lou Goble, ed., The Blackwell Guide to Philosophical Logic. Blackwell.

Hovy E..(1998). Combining and standardizing largescale, practical ontologies for machine translation and other uses. In The First International Conference on Language Resources and Evaluation (LREC), pages 535–542, Granada, Spain.

Jasper R., Uschold, M. (1999). A framework for understanding and classifying ontology applications. In Proceedings of the IJCAI99 workshops on ontologies and problem-solving methods. Sweden: Stockholm.

Jeongsoo Lee J., Chae H., Kim C.H., Kim K. (2009). Design of product ontology architecture for collaborative enterprises, Expert Systems with Applications 36, 2300–2309.

Jun H. B., Kiritsis D.. (2007). Research issues on closed-loop PLM, Computers in Industry 58, 866-868.

Kalfoglou Y., Schorlemmer M.. (2003). IF-Map: an ontology mapping method based on information flow theory. Journal on Data Semantics, 1(1):98–127, Oct. 2003.

Kalfoglou Y., Schorlemmer M. (2002). Information-flow-based ontology mapping. Lecture Notes in Computer Science, 2519, 1132–1151.

Kalfoglou, Y., Schorlemmer, M.. (2003). Ontology mapping: the state of the art. The Knowledge Engineering Review, 18(1):1–31, 2003.

Kärkkäinen M., Ala-Risku T. and Främling K. (2003). The product centric approach: a solution to supply network information management problems?, Computers in Industry 52 (2), 147–159.

Katranuschkov P., Gehre A., Scherer R. (2003), An ontology framework to access IFC model data, in: ITcon; 8, 413–437.

Kifer M., Lausen G., Wu J., (1995). Logical Foundations of Object-Oriented and Frame-Based Languages, Journal of the ACM, 42.

Kim K. Y., Chae S. H., Suh H. W. (2003). An approach to semantic mapping using product ontology for CPC environment, in Proceedings of the 10th ISPE International Conference on Concurrent Engineering Research and Applications, J. Cha, R. Jardim-Gonçalves, and A. Steiger- Garção, Eds. Madeira Island, Portugal: A. A. Balkema, Jul., 291–298.

Kiritsis D., Bufardi A., Xirouchakis P., (2003). Research issues on product lifecycle management and information tracking using smart embedded systems, Advanced Engineering Informatics 17, 189–202.

Kitakami H., Mori Y., Arikawa, M. (1996). An intelligent system for integrating autonomous nomenclature databases in semantic heterogeneity. In Database and Expert System Applications, DEXA'96, number 1134 in Lecture Notes in Computer Science, 187–196, Zurich, Switzerland, 1996.

Klein M. (2001). Combining and relating ontologies: an analysis of problems and solutions. In: Proceedings of the IJCAI-20001 Workshop on Ontologies and Information Sharing, Seattle, WA.

Kotinurmi P. (2005). Towards more intelligent business-to-business integration with semantic web service technologies. Proceedings of the CIMRU-DERI-HP research seminar, The Digital Enterprise Research Institute, Galway, Ireland, 33–35.

Kühn H., Murzek M (2006). Interoperability in metamodelling platforms, in: D. Konstantas, J.-P. Bourriéres, M. Lèonard and N. Boudjlida (eds.): 1st International Interoperability of Enterprise Software and Applications (I-ESA), 216-226, Springer, 2005, Geneva, Switzerland, Berlin Heidelberg.

Lee J., Chae, H., Kim, K., & Kim, C. (2006). An ontology architecture for integration of ontologies. Lecture Notes in Computer Science, 4185, 205–211.

.Liebich T., Amor R., Verhoef M. (1995). A comparison of mapping methods available within the product modelling arena, Proceedings of the Fifth EXPRESS User Group Meeting, Grenoble, France, 21-22 October 1995.

Lima C., Silva C.F., Sousa P., Pimentao J.P. (2005), Interoperability among semantic resources in construction: is it feasible?, in: Proc. 22nd Conference of Information Technology in Construction, cibW78, July 18–21, Dresden, Germany.

.Madnick SE (1995). From VLDB to VMLDB (Very MANY Large Data Bases): dealing with large-scale semantic heterogeneity. Proceedings of the 21st very large data base conference, , 11–16, Zurich.

Mak K.-T., Ramaprasad A.(2001). An Interpretation of the Changing IS/IT-Standard Game, Circa 2001, Knowl. Technol. Policy 12 (14) , 20–30.

McFarlane D., Sarma S., Chirn J.L., Wong C.Y., Ashton K. (2003). Auto ID System and Intelligent Manufacturing Control, Engineering Application of Artificial Intelligence, 16, 365-376.

.McGuinness D. L., Fikes R., Rice J., Wilder S.:, (2000). The Chimaera Ontology Environment. In the Proceedings of the The SeventeenthNational Conference on Artificial Intelligence (AAAI 2000), Austin, Texas, July 30 - August 3.

Mizoguchi R., Vanwelkenhuysen J., Ikeda M. (1995). Task ontology for reuse of problem solving knowledge. In N. J. I. Mars, editor, Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing., pages 46–57. IOS Press, Amsterdam, NL.

Morel G., Panetto H., Mayer F., Auzelle J.P. (2007). System of enterprise-Systems integration issues :an engineering perspective. Invited plenary paper. IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2 – 5, Monterrey, Mexico, IFAC Papersonline.

Morel G., Panetto H., Zaremba M.B., Mayer F. (2003). Manufacturing Enterprise Control and Management System Engineering: paradigms and open issues, in: IFAC Annual Reviews in Control. 27/2, 199-209, Elsevier.

Noy N. and Musen M. A. (2003). The PROMPT suite: Interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies, 59(6):983–1024.

Noy N., & Hafner, C. (1997). The state of the art in ontology design – A survey and comparative review. AI Magazine, 36(3), 53–74.

Noy N., Musen M., (2000) PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000), Austin, Texas, July 30 - August 3.

Noy, N. (2004) Semantic Integration: A survey of ontology-based approaches. SIGMOD Record, 33(4):65–70.

Noy, N. and Musen, M. A.. The PROMPT suite: Interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies, 59(6):983–1024, 2003.

Noy, N.. Semantic Integration: A survey of ontology-based approaches. SIGMOD Record, 33(4):65–70, 2004.

Oh S.C., Yee S-T (2008). Manufacturing interoperability using a semantic mediation,in: Int J Adv Manuf Technol 39:199–210 DOI 10.1007/s00170-007-1198-2.

Oh Y., Han S., Suh H.. (2001). Mapping product structures between CAD and PDM systems using UML, Computer-Aided Design, 33 , 521-529.,

Panetto H. (2007). Towards a Classification Framework for Interoperability of Enterprise Applications. International Journal of CIM, 20/8, 727-740, Taylor & Francis, December 2007, ISSN 0951-192X.

Panetto H., Molina A. (2008). Enterprise Integration and Interoperability in Manufacturing Systems: trends and issues. In: Special issue on Enterprise Integration and Interoperability in Manufacturing Systems, A. Molina and H. Panetto (Eds). Computers In Industry, 59/5, May, Elsevier, ISSN: 0166-3615.

Patil L., Dutta D., Sriram R. (2005), Ontology-based exchange of product data semantics, In: IEEE Transactions of Automation Science and Engineering ; 2, 213–225.

Pepijn R. S. Visser, Dean M. Jones, T. J. M. Bench-Capon, and M. J. R. Shave. An analysis of ontological mismatches: Heterogeneity versus interoperability. In AAAI 1997Spring Symposium on Ontological Engineering, Stanford, USA, 1997.

Pirlein Th., Studer R., (1997). Integrating the Reuse of Commonsense Ontologies and Problem-Solving Methods, University of Karlsruhe, Institute AIFB, Research Report 354, May 1997

Rahm E., Bernstein P.A.(2001). A survey of approaches to automatic schema matching, VLDB J. 10, 334–350.

Shin Y, Han S-H.. (1988) Data enhancement for sharing of ship design models. Computer-Aided Design; 30(12):931±41.

Studer R., Eriksson H., Gennari J. H., Tu S. W., Fensel D., and Musen M.. (1996). Ontologies and the Configuration of Problem-Solving Methods. In B. R. Gaines and M. A. Musen, (eds.), Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada.

Stumme G., Madche, A. (2001). FCA-Merge: Bottom-up merging of ontologies. In 7th Intl. Conf. on Artificial Intelligence (IJCAI ’01), pages 225–230, Seattle, WA.

Su X.M., (2002). A text categorization perspective for ontology mapping.Technical report, Department of Computer and Information Science. Norwegian University of Science and Technology, Norway.

Terzi S. (2005). Elements of Product Lifecycle Management: Definitions, Open Issues and Reference Models. PhD - University Henri Poincaré Nancy I and Politecnico di Milano, May.

Terzi S., Panetto H., Morel G., Garetti M. (2007). A holonic metamodel for product lifecycle management. International Journal of Product Lifecycle Management, 2/3, 253-289, Inderscience, December, ISBN 1743-5110.

Terzi, S. (2005). Elements of Product Lifecycle Management: Definitions, Open Issues and Reference Models. PhD - University Henri Poincaré Nancy I and Politecnico di Milano, May.

Tursi A., Dassisti M., Panetto H. (2007). Products information interoperability in manufacturing systems. Ottavo Convegno AITeM (Associazione Italiana Tecnologia Meccanica) (AITEM'2007). September 10-11, Montecatini Terme, Italy.

Tursi A., Panetto H., Morel G., Dassisti M. (2007). Ontology-based products information interoperability in networked manufacturing enterprises. IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2 – 5, Monterrey, México, IFAC Papersonline.

Tursi A., Panetto H., Morel G., Dassisti M. (2009). Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. IFAC Annual Reviews in Control. 33/3, September, extended version from IFAC Cost Effective Automation in Networked Product Development and Manufacturing, October 2–5, Monterrey, México, Elsevier, ISSN: 1367-5788 (in press).

Uschold M., & King, M. (1995). Towards a methodology for building ontologies. In Proceedings of IJCAI95 workshop on basic ontological issues in knowledge sharing. Montreal, Canada.

Van Heijst G., Schreiber A., Wielinga B. (1997). Using Explicit Ontologies in KBS development, International Journal of Human Computer Studies, 46:183-292.

Vegetti M., Henning G.P., Leone H.P. (2005). PRoduct ONTOlogy. An ontology for complex product modelling domain, in: Proceedings of the ENPROMER, Rio de Janeiro.

Vegetti, M.; Henning, G. P.; Leone, H. P. Product ontology: definition of an ontology for the complex product modelling domain. In: MERCOSUR CONGRESS ON PROCESS SYSTEMS ENGINEERING, 4., 2005. Proceedings…Costa Verde, RJ: UFRJ. Available from: http://www.enpromer2005.eq.ufrj.br/lng/en/index.php> Accessed on: Feb. 28th 2008.

Vernadat F. B. (1996). Enterprise modelling and integration: Principles and applications. London: Chapman & Hall.

Visser Pepijn R. S., Jones Dean M., Bench-Capon T. J. M., and Shave M. J. R.. (1997). An analysis of ontological mismatches: Heterogeneity versus interoperability. In AAAI 1997, Spring Symposium on Ontological Engineering, Stanford, USA.

Wainwright D., Waring T.(2004). Three domains for implementing integrated information systems: redressing the balance between technology, strategic and organisational analysis, Int. J. Inf. Manag. 24, 329–346.

Weibel S., Gridby J., Miller E., (1995). OCLC/NCSA Metadata Workshop Report, Dublin, EUA. http://www.oclc.org:5046/oclc/research/conferences/metadata/dublin_core_report.html.

Whitman L.E., Panetto H.(2006). The missing link: culture and language barriers to interoperability, Annu. Rev. Control 30, 233–241.

Wilfrid Hodges, 2001, "Classical Logic I: First Order Logic," in Lou Goble, ed., The Blackwell Guide to Philosophical Logic. Blackwell.

Xu X.W., Newman S.T.(2006). Making CNC machine tools more open, interoperable and intelligent—a review of the technologies, Comput. Ind. 57 (2), 141–152.

Xu, H.C.; Xu, X.F. and He, T. (2008). Research on Transformation Engineering BOM into Manufacturing BOM Based on BOP, Applied Mechanics and Materials. Special issue on e-Engineering & Digital Enterprise Technology. 10-12, 99-103.

Yahia E., Yang J., Aubry A., Panetto H. (2009). On the use of Description Logic for Semantic Interoperability of Enterprise Systems. On the Move to Meaningful Internet Systems: OTM 2009 Workshops (Meersman R., Tari Z., Henerro P. (Eds)), 4th IFAC/IFIP Workshop on Enterprise Integration, Interoperability and Networking (EI2N'2009), Vilamoura, Portugal, November 4-5, Springer Verlag, Lecture Notes in Computer Science, LNCS 5872, ISBN 978-3-540-88874-1.

Yoo S.B., Kim Y..(2002). Web-based knowledge management for sharing product data in virtual enterprises, International Journal of Production Economics 75 173–183.

Technical references

ATHENA. (2003, April). Advanced technologies for interoperability of heterogeneous enterprise networks and their applications. FP6-2002-IST-1, Integrated Project Proposal.

EIF (2004), European Interoperability Framework for pan-European eGovernment Services, Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens (IDABC), November, Luxembourg.

EN/ISO 19439, Enterprise Integration—Framework for Enterprise Modelling, 2003.

Exff project (2003). The exff EXPRESS Guide. Release 0.1. http://exff.sourceforge.net.

IEC 62264 (2002). Enterprise-control system integration, Part 1. Models and terminology, Part 2: Model object attributes. ISO/IEC, 2002, Geneva.

IEC TC65/290/DC. (2002, June 28). Device profile guideline, TC65: Industrial process measurement and control.

IEEE (1990), Standard Computer Dictionary- A Compilation of IEEE Standard Computer Glossaries. NY. 610-1990. ISBN: 155937079.

INTEROP. (2003, April 23). Interoperability research for networked enterprises applications and software, network of excellence. Proposal part B.

ISO 10303 (STEP) task website, stepmod.sourceforge.net.

ISO 14528 (1999). Industrial Automation Systems – Concepts and rules for Enterprise Models, TC 184/SC5/WG1, Geneva, Switzerland.

ISO 16100 (2002). Manufacturing Software Capability Profiling for interoperability, Part 1: Framework, TC 184/SC5/WG4, Geneva, Switzerland, ICS 25.040.01.

ISO EN DIS 19440 (2004). Enterprise integration - Constructs of enterprise modelling, Draft version, TC 184/SC5/WG1, Geneva, Switzerland.

ISO TC/184/SC4/WG11 N013, Requirements specification for EXPRESS mapping language, (1997).

ISO/TC184/SC4, Industrial automation systems and integration—product data representation and exchange—Part 11:Descriptive Methods: The EXPRESS Language Reference Manual (1992).

ISO/TC184/SC4/WG11 N088, EXPRESS-X Language Reference Manual (1999).

ISO/TS 10303 STEP modules related to Product Data Management. Industrial automation systems and integration — Product data representation and exchange, 2004, Geneva.

OASIS, (2002). Organization for the Advancement of Structured Information Standards. ebXML specification, February, http://www.oasis-open.org.

PABADIS'PROMISE, www.pabadis-promise.org

PLCS, www.plcs.org

PLM XML, www.ugsplm.com

PROMISE PLM, www. promise.plm.com

ST-Developer User Guide, STEPTools, Inc., 1995.

STEP PDM Schema, PDMnet, http://www.pdmnet.org, 1998.

UML (2005). Unified Modeling Language. UML 2.0 Superstructure, v2.0 formal 05/07/04. OMG

W3C (2003). Simple Object Access Protocol (SOAP) 1.2, W3C specification, February, http://www.w3c.org

W3C (2004a), XML, Extensible Mark-up Language, W3C XML 1.1 recommendation, February, http://www.w3c.org

W3C (2004b). Web Services Description Language (WSDL) 2.0, W3C specification. August, http://www.w3c.org

W3C. OWL Web ontology language overview, http://www.w3.org/TR/2003/PR-owl-features-20031215/; 2005 [last accessed 10.05].

List of the acronyms

AI	Artificial Intelligence
AIM	Application Integration Middleware
AM	Application Module
AP	Application Protocol
APS	Advanced Planning and Scheduling
ARM	Application Reference Model
B2B	Business to Business
B2C	Business to Consumer
B2M	Business to Manufacturing
B2MML	Business to Manufacturing Markup Language
BOM	Bill Of Material
BOP	Bill Of Process
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CEN	Comité Européen de Normalisation
CIM	Computer Integrated Manufacturing
CORBA	Common Object Request Broker Architecture
CPM	Core Product Model
CRM	Customer Relationship Management

EAI	Enterprise Application Integration
EAS	Enterprise Applications and Software
EBOM	Engineering Bill of Material
ebXML	Electronic Business using eXtensible Markup Language
EIF	European Interoperability Framework
EM	Enterprise Modelling
ENV	European Norm Vorubergehend
ERP	Enterprise Resource Planning
EPC	Electronic Product Code
FIPA	Foundation for Intelligent Physical Agent
FOL	First Order Logic
HMS	Holonic Manufacturing Systems
HTML	HyperText Markup Language
IEEE	Institute of Electrical & Electronics Engineers, Inc.
IEC	International Electrotechnical Commission
ISA	International Society of Automation
ISO	International Organisation for Standardisation
ICT	Information Communication Technology
IT	Information Technology
KIF	Knowledge Interchange Format
KQML	Knowledge Query and Manipulation Language
MES	Manufacturing Execution System
MBOM	Manufacturing Bill of Material

MRP	Materials Requirements Planning
NIIIP	National Industrial Information Infrastructure Protocols
ONS	Object Naming Service
OWL	Web Ontology Language
PDM	Product Data Management
PEID	Product Embedded Information Devices
PLCS	Product Life-Cycle Support
PLM	Product Lifecycle Management
PML	Physical Mark-up Language
PRM	Purdue Reference Model
PRSL	Product Semantic Representation Language
PSL	Process Specification Language
RDF	Resource Description Framework
RFID	Radio Frequency IDentification
RIM	Reference Information Model
SCM	Supply Chain Management
SGML	Standard General Mark-up Language
SOAP	Simple Object Access Protocol
SoS	System-of-Systems
STEP	STandard for the Exchange of Product model data
TCP/IP	Transmission Control Protocol/Internet Protocol
TS	Technical Specifications
UEML	Unified Enterprise Modelling Language

UML	Unified Modeling Language
WSDL	Web Services Description Language
W3C	World Wide Web Consortium
XSD	XML Schemas definition language
XMI	XML Metadata Interchange
XML	eXtensible Markup Language

Figure Index

Table Index

Author: Angela Tursi
Title: Ontology-Based approach for Product-Driven interoperability of enterprise production systems

RESUME

Depuis quelques années, l'interopérabilité des applications est devenue le leitmotiv des développeurs et concepteurs en ingénierie système. Cette importance a donné lieu à d'innombrables travaux de recherche avec chacun une définition particulière plus au moins formelle de l'interopérabilité entre applications. La plupart des approches pour l'interopérabilité existant dans l'entreprise ont pour objectif principal l'ajustement et l'adaptation des types et structures de données nécessaire à la mise en œuvre de collaboration entre entreprises. Dans le domaine des entreprises manufacturières, le produit est une composante centrale. Des travaux scientifiques proposent des solutions pour la prise en compte des systèmes d'information issus des produits, tout au long de leur cycle de vie. Mais ces informations sont souvent non corrélées. La gestion des données de produit (PDM) est couramment mise en œuvre pour gérer toute l'information relative aux produits durant tout leur cycle de vie. La modélisation des processus de fabrication et de gestion est largement répandue et appliquée tant aux produits physiques qu'aux services. Cependant, ces modèles sont généralement des "îlots" indépendants ne tenant pas compte de la problématique d'interopérabilité des applications supportant ces modèles. L'objectif de cette thèse est d'étudier cette problématique d'interopérabilité appliquée aux applications utilisées dans l'entreprise manufacturière et de définir un modèle ontologique de la connaissance des entreprises relatives aux produits qu'elles fabriquent, sur la base des données techniques de produits, pour assurer l'interopérabilité des systèmes d'entreprise supports, basés sur un échange d'information centrée sur le produit. Le résultat attendu de ce travail de recherche concerne la formalisation d'une méthodologie d'identification des informations de gestion techniques des produits, sous la forme d'une ontologie, pour l'interopérabilité des applications d'entreprises manufacturières, sur la base des standards existants tels que l'ISO 10303 et l'IEC 62264.

MOTS-CLES : Interopérabilité des systèmes d'entreprise, PDM, système d'information, IEC 62264, ISO 10303, Ontologie Produit

ABSTRACT

One of the emerging problems in manufacturing systems is the *information interoperability* problem: managing heterogeneous information coming from different systems, in order to achieve a unique comprehension when exchange is taking place. Information is required to be coherent and congruent with the specific use in interfacing enterprise applications, at any stage of the product lifecycle management. Most approaches to interoperability in the company have the primary objective of adjustment and adaptation of types and data structures necessary for the implementation of collaboration between companies. In the field of manufacturing, the product is a central component. Scientific works propose solutions taking into account information systems derived from products technical data throughout their life cycle. But this information is often uncorrelated. The management of product data (PDM) is commonly implemented to manage all information concerning products throughout their life cycle. However, these approaches are based on systems that generally are independent “islands” ignoring the problem of interoperability between applications that support information models. Standardisation initiatives (ISO and IEC) try to answer the problem of managing heterogeneous information scattered within organizations, by formalising the knowledge related to products technical data. The objective of this thesis is to study the problem of interoperability applied to applications used in the manufacturing environment and to define a model of the ontological knowledge of enterprises related to the products they manufacture, based on technical data, ensuring the interoperability of enterprise systems. The outcome of this research concerns the formalization of a methodology for identifying a product-centric information system in the form of an ontology, for the interoperability of applications in manufacturing companies, based on existing standard such as ISO 10303 and IEC 62264.

KEYWORDS: Enterprise systems interoperability, PDM, Information system, IEC 62264, ISO 10303, Product Ontology

SINTESI

Nell’attuale contesto manifatturiero, uno dei problemi emergenti è quello di gestire le informazioni eterogenee che si scambiano i diversi sistemi, al fine di avere un'unica interpretazione dell’informazione stessa, senza possibilità di incomprensioni: questo problema è noto come *interoperability problem*. L’informazione scambiata dalle diverse applicazioni aziendali durante tutte le fasi del ciclo di vita del prodotto deve essere coerente e congruente con l’uso che se ne fa della stessa. La maggior parte degli approcci all’interoperabilità nell’azienda hanno l’obiettivo primario di sistemare e adattare i tipi e le strutture di dati necessari per l’implementazione della collaborazione tra aziende. Nel campo manifatturiero, il prodotto è un componente centrale. Alcuni lavori scientifici propongono soluzioni che prendono in considerazione sistemi di informazione derivanti da dati tecnici del prodotto, attraverso il suo ciclo di vita. Questa informazione è spesso sconnessa. Il PDM (Product Data Management) è un approccio comunemente implementato in azienda per gestire tutte le informazioni che riguardano i prodotti attraverso il loro ciclo di vita. Tuttavia, tutti questi approcci si basano su sistemi che sono generalmente “isole” indipendenti che ignorano il problema della interoperabilità che c’è tra le applicazioni che supportano i diversi modelli di informazione. Le iniziative di standardizzazione esistenti (ISO e IEC) cercano di risolvere il problema di gestire le informazioni sparse all’interno dell’organizzazione nei diversi sistemi, formalizzando la conoscenza dei dati tecnici di prodotto. L’obiettivo di questa tesi è quello di studiare il problema di interoperabilità esistente tra applicazioni usate nell’ambiente manifatturiero e di definire un modello di conoscenza ontologica dell’azienda relativa ai prodotti che l’azienda stessa produce, basato su dati tecnici, assicurando l’interoperabiltà dei sistemi aziendali. Il risultato di questa ricerca riguarda la formalizzazione di una metodologia che identifichi un sistema di informazione “product-centric”, basato su standard esistenti, quali ISO 10303 e IEC 62264 ed espresso nella forma di un’ontologia, che favorisca l’interoperabilità delle applicazioni in aziende manifatturiere.

KEYWORDS: Interoperabilità tra sistemi aziendali, PDM, Sistemi di informazione, IEC 62264, ISO 10303, Ontologia del Prodotto