Explicitating semantics in Enterprise Information Systems Models

Mario Lezoche

Nancy-Université
Université Henri Poincaré

Faculté de Sciences et Technologies
Ecole Doctorale IAEM Lorraine

Centre de Recherche en Automatique de Nancy
CRAN

UMR 7039
NANCY-UNIVERSITE
CNRS

Laboratoire Lorrain de Recherche en Informatique et ses Applications
Loria

UMR 7503
NANCY-UNIVERSITE
CNRS

Rapport de Recherche

Présenté en vue de l'obtention du

Diplôme de Recherche Post Doctorale de l'Université Henri Poincaré, Nancy 1

par

Mario Lezoche

Docteur de l'Université Roma TRE (Italie)

Explicitating semantics in Enterprise Information Systems Models

Soutenance publique le 7 Novembre 2011 devant la commission d'examen :

Membres du Jury :

Président du Jury :	Thierry Divoux	Professeur à l'Université Henri Poincaré – Nancy 1
Directeur de Travaux :	Hervé Panetto	Professeur à l'Université Henri Poincaré – Nancy 1
Encadrant :	Alexis Aubry	Maitre de Conférences à l'Université Henri Poincaré – Nancy 1
Examinateur :	Nacer Boudjlida	Professeur à l'Université Henri Poincaré – Nancy 1

cnrs

E S I A L

To my beloved Bebaia and my faithful Chicca

Acknowledgement

This Post-Doctoral experience was firstly started under Interop VLab project and it was completed at the CRAN laboratory in the SYMPA team, SIO project, Université Henri Poincaré – Nancy 1 and at LORIA in the Score team. Therefore I would like to thank the CRAN laboratory director, Pr. Alain Richard, the SYMPA thematic manager, Pr. Thierry Divoux, the responsible of the SIO project, Pr. Benoit Iung, the LORIA director Françoise Simonot, the Score team manager Francois Charoy and Pr. Nacer Boudjlida, my guide in the Loria world, for welcoming me in such a warm way. I would like also to thank the Director of ESIAL School of engineering, Pr. André Schaff, who welcomed me during my pedagogic duties.

When I arrived in Nancy I found a weather that was quite different from that of my birth country. It was January and, I remember precisely, there were -18 Celsius degrees. I really hoped the people I would have met the next day would have been much warmer. My hopes were more than fulfilled. I am greatly attached to the concept of family and Life helped me letting meet people who became my new extended family.

Firstly, I would like to express my thankfulness to Professor Hervé Panetto who greeted me with an enormous smile. He guided me through the development of this research topic but also he let me acquire knowledge in different domains, better comprehension of research and pedagogic methodologies, involvement in laboratory and school life at all levels. It was really a diving experience in a new way to conceive this job, this passion. Our friendly relationship let me find the fire that warmed the cold distance to my family. I would like to thanks A/Professor Alexis Aubry, he is a special researcher and person. We had a lot of interesting and worthwhile discussions during this period and, from the moment we became roommates we deepened our knowledge and I find an all-accomplished person with whom I have many common interests. I like to thanks my colleague Esma Yahia for welcoming me in an enthusiastic way and for working hard together during all this time. Her feisty character

spurred me to always deepen the topics we discussed on. We rapidly became friends and knowing and helping each other made me feel at home.

Next to the core people teams, with whom I worked, there are all the laboratory people that enriched in an extraordinary way my work and private Nancy life. Pierre, Gabriela, Pascale and Chiara created a little family, my little Nancy family, who shared sadness and happiness, failures and victories… Life.

I would like to thanks all the other kind people of the laboratory and of the school who were always nice and wishful to help, David, William, Thomas, Alexandre, Fabien, Yongxin, Leila, Sylvain, Jéremy, Alex, Ludovic, Romain and all the other really kind people with whom I passed my time in and out the laboratory.

What I am living is a dream, one of my oldest dream, living an experience in a country I always wanted to know and performing a job I was eager to transform in my life.
The same life that changed three years ago when I met the person who transformed the way I felt my life. I would like, so, to thank my beloved Costanza who gave me the strength, the energy and a new perspective to look at the events, a new hope in our private spiritual road and who made me remember what the meaning of Life is.

1 Introduction

The present Post-Doctoral work was firstly financed with a grant offered by the University Henri Poincaré, Nancy I, and it was completed at the Research Centre for Automatic Control (Centre de Recherche en Automatique de Nancy - CRAN) in the Ambient Manufacturing System (SYMPA) team, Inter Operating System (SIO) project, and at the Lorraine Research Laboratory in Computer Science and its Applications (Laboratoire Lorrain de Recherche en Informatique et ses Applications – LORIA) in the Score team.
During this period, passed working as a researcher in these laboratories, I had the possibility to act as a teacher at the School of Engineering in Information Technology ("École Supérieure d'Informatique et Applications de Lorraine" - ESIAL). This experience helped me to better comprehend the different views of the research and teaching life.

1.1 Research domain context

In a present expanded market, enterprises are forced to become increasingly fast adapting and flexible in order to manage the rapid changing business conditions. Today's challenges mainly concern Enterprise Interoperability (EI) that focuses on removing organisational barriers and improving different type of interaction between people, systems and companies. EI passes, mainly, through their Enterprise Information Systems (EISs). They involve large number of ISs distributed over large, complex networked architecture. Such cooperative enterprise information systems (CEIS) have access to a large amount of information and have to interoperate to achieve their purpose. CEIS architects and developers have to face a hard problem: interoperability.

Interoperability can be defined as the ability of two or more systems to share, to understand and to consume information (IEEE, 1990). The work (Chen et al., 2006) in the INTEROP NoE project has identified three different levels of barriers for interoperability: technical, conceptual and organisational. Organisational barriers are still an important issue but they are out of scope of this paper. The technological barriers are strongly studied by researchers

in computer science and they are, in general, addressed by the models transformation (Frankel, 2003).

Enterprise Modelling (EM) plays a critical role in this interoperability action, enabling the capture of all the information and knowledge relevant for the enterprise operations and organisation (Vernadat, 1996; Panetto and Molina, 2004). The produced Enterprise Models must contain the necessary and sufficient semantics in order to be intelligible and then enabling the global Enterprise Interoperability. While studying an Information System (*IS*) model, we observe that its semantics is scrambled, due to the implementation requirements, and, more important, it is tacit.

Our research focuses on the conceptual level of interoperability, namely the ability to understand the exchanged information. Information may be defined as data linked to knowledge about this data. It is represented by so-called concepts. A concept is a cognitive unit of meaning (Vyvyan, 2006), an abstract idea, a mental symbol. It is created in the activity of conceptualisation, that is, a general and abstract mental representation of an object. During the history of human effort to model knowledge, different conceptualisation approaches regarding different application domains were developed (Aspray, 1985).

This research memory will show the results obtained during the Post Doc study referring to the published works. It deals with a first phase from our general research work that focuses on the study of the semantic loss that appears in the exchange of information about business concepts. **In order to quantify the semantic gap between interoperating ISs, their semantics needs to be enacted and structured by enriching, normalising and analysing their conceptual models.** We propose a conceptualisation approach for explicitation of the finest-grained semantics, embedded into conceptual models in order to facilitate the semantic matching between two different information systems that have to interoperate. The structure of the document represents the different steps and the research domain on which the study focused.

In section 2, we present the general context of our work, namely cooperative enterprise information systems. The following section, the 3rd section proposes a knowledge

explicitation process that transforms implemented relational model to a fact-oriented conceptual one. This process allows us to discover the finest-grained semantics that must be enacted to study semantics interoperability between collaborating ISs. Then, we will define First Order Logic formalisations of UML artefacts (classes, attributes, associations).

In the 4th section, the semantics structuring method is described. It assumes the definition of semantic aggregates that highlight the structure of the embedded semantics in the conceptual model obtained after the conceptualisation process. Each semantic aggregate (namely, each semantic block) is associated with a concept and defines the minimal mandatory semantics attached to this concept.

In order to illustrate the proposed approach, a case study is also presented in the section 5. This case study deals with B2M (Business to Manufacturing) interoperability requirements between an Enterprise Resource Planning (ERP) system and a Manufacturing Execution System (MES) applications and consists in applying our approach in order to extract the semantics embedded into those ISs. In Section 6 new study on Semantic annotation structure and use is presented in order to better explicitate the tacit knowledge hidden in the Enterprise Models. Moreover, enriching this semantics is still an open issue; we can for example quote those researches made by (Boudjlida and Panetto, 2008) in terms of process models annotations.

In section 7 we will discuss about further works concerning using the resulting semantic conceptual model and architecture for facilitating the assessment of the (non)-interoperation barriers between Enterprise Information Systems or some of their subsystems (identified, for instance, by the semantic blocks) as suggested in (Yahia, 2011). The resulting analysis, based on an interoperability measures map, can help information technology consulting companies for parameterising and integrating enterprise applications (ERP, MES…) taking into account interoperability constraints. The Appendix, containing all the published research works, completes this report.

2 Cooperative Information System

Information Systems are systems whose activities are devoted to capture, store and process data and to produce knowledge, used by any stakeholders within an enterprise or among different networked enterprises. It is commonly agreed that Cooperative Information Systems provide a backbone for the Integrated Information Infrastructure (Sheth, 1998). Fully understanding and exploiting the advances in computing is the only way to encompass the complexity of developing and maintaining such systems.

Although the progress made in Information Technology (IT) considerably improved the efficiency of software development, its drawbacks and limitations are obvious and serious. In fact, the models involved in a single application development are numerous and diverse, each coping only with particular and partial aspects of the overall task. Moreover, the components' technologies are diverse, platform- and machine-dependant. The above-mentioned limitations and barriers hinder the development and the maintenance process, significantly. *Though our knowledge has been enriched by such diversity, an ancillary consequence has been separate research conversations, hampering cross-pollination of ideas and findings and making it difficult for those working outside the area to understand what we have learned*. (Melville *et al*., 2004).

There is a growing demand for integrating such systems tightly with organizational work so that these information systems can be fully, directly and immediately exploited by the intra and inter-enterprise processes (Izza, 2009).

Here, the need of interoperation clearly appears. In fact, to achieve the purpose of the cooperation between the different Information Systems, information must be physically exchanged (technical interoperability), must be understood (conceptual interoperability) and must be used for the purpose for which it has been produced (conceptual and organisational interoperability). When trying to assess the understanding of an expression coming from a system to another system, several possible levels of interoperability can be identified (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).

This tiered structure is arguable in general; it is not as strict as it seems. In a way, it reflects maturity levels of the interoperability between the information systems, because each level cannot be achieved if the previous levels have not been completed (Euzenat, 2001).

The encoding, lexical and syntactic levels are the most effective solutions for removing technical barriers for interoperability, but they are not sufficient to achieve a practical interoperability between computerised systems. Enabling a seamless data and model exchange at the semantic and semiotic levels is still a big challenge which needs conceptual representation of the intended exchanged information and the definition of its pragmatic meaning in the context of the source and destination applications.

Different cooperation types have been investigated in ISO 14528 (ISO, 1999). In fact, this standard considers that models could be related in three ways:

(1) integration, when there exists a standard or pivotal format to represent these models;
(2) unification, when there exists a common meta-level structure establishing semantic equivalence between these models; and
(3) federation, when each model exists per se, but 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 (Panetto, 2007). Classification of the interoperability problems (Tursi, et al. 2009) may help in understanding the degree of development needed to solve, at least partially, these problems. However, conceptualisation and semantics extraction

is still an important issue because of the different, often contextual understanding of tacit knowledge embedded into those applications. This issue is typically driven by the misbalance of the needed ontological commitment and epistemological dimension in the conceptualisation process. In this sense, our task is not really to conceptualise the EIS models, but to make assumptions on the mental models of the information systems' designers, which they then expressed as Entity-Relationship models, and to introduce the ontological commitments by making those models fully or partially equivalent to the real world semantics. The main prerequisite for achievement of interoperability of information systems is to maximise the amount of semantics which can be used and to enact it by making it increasingly explicit (Obrst, 2003).

This section is derived from the following scientific publication:

Lezoche M., Panetto H., Aubry A., (2011). Conceptualisation approach for cooperative information systems interoperability, ACM. 13th International Conference on Enterprise Information Systems, ICEIS 2011, Jun 2011, Beijing, China. pp. 101-110

3 Our approach for semantics enactment in conceptual models

In order to cooperate, two (or more) Information Systems have to interoperate. As previously discussed, we focus our interest on the conceptual level of interoperability and on enabling different information systems to share and use knowledge models which they represent. In order to make this possible, we consider (Lezoche et al., 2011) two steps that need to be taken: first, we need to understand the conceptual relationships between those models in the context of their use; and second, we need to unhide the tacit knowledge buried inside them, by using conceptualisation.

Conceptualisation is a decision process (Guarino, 1998), a view, in which knowledge of the studied part of reality, typically available in an implicit and complex form, is reorganised and generalised in different aggregates. Conceptual models range in type from the more precise, such as the mental image of a familiar physical object, to the abstractness of mathematical models which cannot be visualized in mind. They can be developed in different levels of abstraction of a single domain (Zdravković et al., 2011). Conceptual models also range in terms of the scope of the subject matter that they are taken to represent. The variety and scope of conceptual models is due to the variety of purposes that people had while using them. The same applies for conceptualisation approaches, which are numerous and have been developed in different knowledge domains (LaOnsgri, 2009). According to (Engelbart, 1962), developing conceptual models means specifying the essential objects or components of the system to be studied, the relationships of the objects that are recognised, the types of changes in the objects or their relationships which affect the functioning of the system and the types of impact these changes have on the system. Similarly, Genesereth and Nilson (Genesereth and Nilson, 1987) define conceptualisation as “the objects, concepts and other entities that are assumed to exist in some area of interest and their inter-relationships”. Both definitions assume extensional character of the conceptualisation process, in the sense that they imply that the elements of the mental image of the specific domain are simply enumerated or listed. Some researchers (Guarino, 1997) argue that this contradicts to an intentional character of a human thinking, where the meaning of elements is constituted by

their necessary and sufficient conditions. These arguments are partially taken into account in our work by interpreting the semantics of the cardinality of relationships and existential constraints (mandatory elements).

Our contribution is to have at our disposal an approach which enables us to fragment knowledge through the transformation of attributes into entities and relationships, and thus to discover finest-grained knowledge atoms. In the proposed approach, presented in the Figure 1, different inputs can be used, such as an application, a data model, or a logical view. On this approach, the initial process (Step 1) is application of the reverse engineering methods, such as in (Fonkam, 1992) and in (Chiang, 1994), for delivering a conceptual model starting from the considered inputs. Then, the resulted initial model is enriched and validated through an Expert Knowledge Injection process (Step 2). In fact, the model is examined with the help of a domain expert or an end-user, who are the most qualified persons to describe the context of the particular domain and to affirm the conceptual model. According to the enterprise best practices and the associated data, they would clean and better organise the knowledge represented in the derived model. However, the obtained initial conceptual model, in the form of a UML class diagram, still has some major limitations from a semantic perspective. Indeed, for example, all the attributes are buried inside classes. Hence, their semantics is not explicit.

In order to overcome these limitations, in the next step of our approach (Step 3), namely a Fact-Oriented Transformation (Halpin, 1991), a set of rules for transforming the enriched conceptual model to a fact-oriented model (FOM) is applied. The core of this approach (FOM) is based on the so-called Lexical ObjecTs (LOTs) and Non-Lexical ObjecTs (NOLOTs). These artefacts are defined in (Meersman, 2003) as follows: *a lexical object (LOT), a term, is an object in a certain reality that can be written down. LOTs always consist of letters, numbers, symbols or other characters. They can be used as names for or references to other objects. A non-lexical object (NOLOT), a concept, is an object in a certain reality that cannot be written down. Non-lexical objects must be named by lexical objects or referred to by means of lexical objects.* In the outcome of the step 3, all the classes and their attributes are transformed into NOLOTs and LOTs respectively. The resulting fact-

oriented model, displaying the finest-grained semantic atoms, is then used as an input for the structuring process presented in section 4.

In the following sub-sections, we will discuss, in detail, the proposed 3 steps.

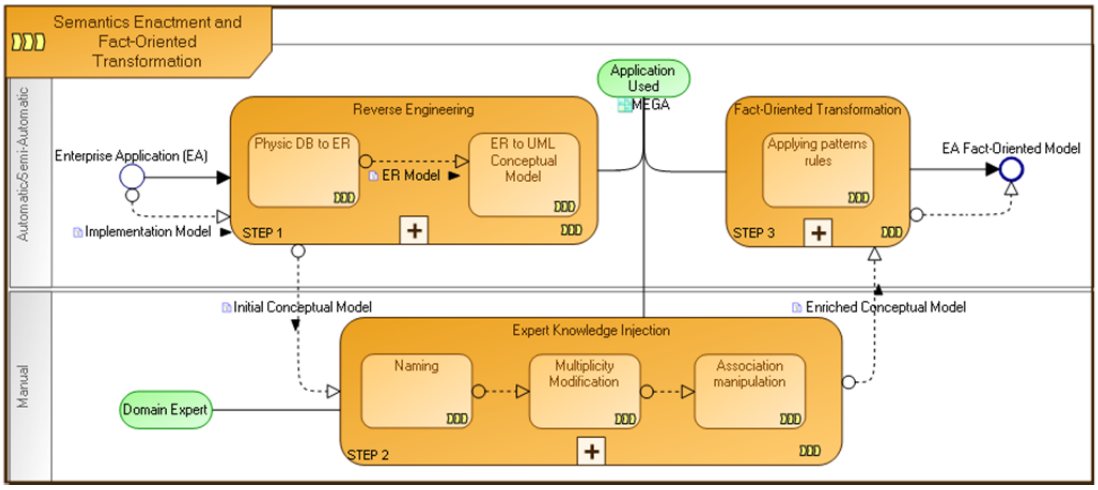


Figure 1 - Conceptualisation process

3.1 Step 1: Reverse Engineering

Our scenario assumes that we start from an enterprise application database. So, the first process is a reverse engineering. It is an approach to extract the domain semantics from the existing database structures.

Typically, the reverse engineering process concerns the application of transformation rules which transform logical to conceptual schema. In (Fonkam, 1992), the authors propose a general algorithm based on several old attempts to make explicit the logical structure buried into DB schemas, application programs and assumed intent of designers and developers. (Chiang, 1994) presents a methodology for extracting an extended Entity-Relationship model from a relational database, by using a combination of data schema and data instance analysis. In our study, we will consider and reuse the reverse engineering experiences developed in the past. These methods are, by now, adopted by the industry which produced a number of software tools. We choose MEGA Suite (http://www.mega.com), a modelling

management environment to transform relational models into conceptual ones. MEGA Suite implements a parameterised reverse engineering method coping with major existing approaches from direct database metadata analysis to a semi-automatic conceptual models building from existing database schemas.

3.2 Step 2: Expert Knowledge Injection

Although most of the reverse engineering approaches (Fonkam, 1992) (Chiang, 1994) produce the information structure, they deliver models without the explicitation of the tacit semantics. The ADM (Architecture-Driven Modernization) initiative (OMG, 2003) from OMG (Bézivin et al., 2005) is tackling this problem by implementing a common Knowledge Discovery Meta-model to facilitate discovery of the tacit knowledge embedded inside existing software. Sometimes, Entity-Relationship models, namely database schemas, do not capture the semantics of the application functionality and underlying data models; when information systems are highly generic, the application semantics is actually captured in the populated table rows. For example, in Business Process Management systems, the structure of the enterprise processes, namely activities, associated data structures (messages), compensation and error handling blocks, etc. are defined by a system user and are not expressed by the database schema. In these cases, the intervention of the domain expert in enriching the conceptual model may be useful. Some research is tackling this issue by providing the tools to automatically or semi-automatically discover the semantics buried into existing data patterns (Astrova, 2004). In our scenario, we are considering that enterprise applications store all their business knowledge into a DBMS. We can then extract, from each of them, some knowledge in a form of a conceptual model, by using reverse engineering approaches. Then, a domain expert has to enrich that model with enterprise best practices (knowledge coming from users). This is the goal of the current step.

After the reverse engineering process has produced a conceptual model, it is enriched by injecting some enterprise knowledge, expressed by the domain experts' or users' practices of using the corresponding enterprise application. These stakeholders know the domain peculiarities and they are capable to express the specific constraints that must be embedded

into the conceptual model. However, this phase must follow a structured process, in order to preserve the ontological commitment. This is particularly important when more experts are involved in the knowledge injection. In such cases, the approaches of setting up a collaborative conceptualisation processes (Guo, 2009) may be useful.

The first stage of the Step 2 is the renaming process. Usually, the database tables and columns (and consequently, the modelled concepts) do not have standard names. Thus, the renaming process is essential for bringing coherence and semantics to the lexical terms that otherwise would be very difficult to comprehend.

The next stage is the redefinition of the attributes and of the associations' roles multiplicities, according to the enterprise system users' practices. This step is fundamental for defining the real constraints which are not always explicit in the implementation model. For example, considering a particular attribute a_1, two possible redefinition cases are identified:

(1) a_1 is a non-mandatory attribute in the conceptual model but, as users are always requested to populate it with a specific value, the enriched model must formalise that this attribute a_1 has to be treated as mandatory;
(2) a_1 is defined as mandatory in the conceptual model but, in practice, the users never care about its value and generally fill it with some dummy one. In such case, the enriched model may formalise that this attribute is not mandatory.

The last stage concerns of making explicit some implicit associations. Those implicit associations relate some concepts but they are defined only by enterprise practices even if they are not expressed in the model itself. For example, let us consider, in a given enterprise, a good practice imposed for achieving information update traceability. When a user updates information concerning one product, the application must store, in dedicated fields, the date of the update and the name of the logged user. This feature is implemented directly into an application like the ERP Sage X3 but it is not reflected in the data model. Moreover, for the sake of simplifying the implementation, the developers did not set these attributes as mandatory. In order to consider this practice in the conceptual model, the constraint must

then be conceptualised as one mandatory association between the existing concepts *Product* and *Users* and by constraining the existing attribute *UpdateDate* in the previous association.

At this time, the enriched conceptual model formalises the whole application semantics (both the explicit ones and the users' implicit ones).

3.3 Step 3: Fact-Oriented Transformation

The quality of a conceptual model is often influenced by the conceptual language used for its specification. There are different approaches in conceptual modelling and these differences are reflected in the conceptual languages used for the modelling action. Entity-Relationship approaches (E-R) have been widely used and extended. They led to the development of different languages for data modelling (Barker, 1990), (Czejdo et al, 1990), (Hohenstein, 1991). Object-Oriented Modelling (OOM) (Rumbaugh et al, 1991) approach addresses the complexity of a problem domain by considering the problem as a set of related, interacting Objects. Entity-attribute-value model (EAV) (Chen et al, 2000) is a data modelling approach used to represent entities with a potentially vast number of attributes (properties, parameters).

However, the abstract semantics inherent to these approaches imposes the modeller to make subjective choices between entities, attributes and relationships artefacts for modelling a universe-of-discourse. Let us consider, for instance, the same concept modelled in two different ways (Figure 2). Intuitively, those concepts (represented as UML classes) represent out similar semantics (at least from a global point of view), but are modelled differently. For instance, the *WEIGHT* of a *PRODUCT* on the right side of the figure is represented by a single class due to, for example, an implementation constraint. When other classes are related to this class, a querying for specific values related to the weight is facilitated. In contrast, on the left side of the figure, the *WEIGHT* of a *PRODUCT* is modelled by two attributes (its value and its unit).

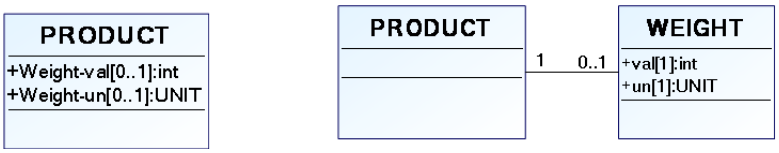


Figure 2 - Two choices of concept modelling

In order to cope with such heterogeneous modelling patterns, we focus our interest on approaches that enable their normalization to a fine-grained semantic model by fragmenting the represented knowledge into atoms. NIAM (Natural-language Information Analysis Method) (Nijssen et al, 1989) proposed to model the world in term of facts (either presenting terms (real things), or representing characteristics (attributes) of these real things), and relationships between facts. NIAM is attribute-free. We adapted this fact-oriented modelling approach idea to the UML (OMG, 2004) class notation representation of the conceptual models. Thus, we developed in (Lezoche et al., 2011) a set of transformation modelling rules, to be applied to selected UML patterns (Table1). In the resulting fact-oriented model, the semantics is preserved by adding annotations. The added annotations concern particular artefacts semantics such as generalisation, association class, aggregation and composite aggregation.

A set of transformation modelling rules, to be applied to selected UML patterns is presented on Table1.

Let us refer to the definitions of LOT and NOLOT facts given in the beginning of section 3. Transforming a particular conceptual model in a fact-oriented model must follow these rules:

(1) all classes are transformed into LOT facts. Using UML Class notation, a LOT fact is represented by a UML Class.
(2) all attributes are transformed into NOLOT facts. Using the UML Class notation, a NOLOT fact is represented as a UML Class.
(3) for each attribute *a* belonging to a UML Class *C*, an association is created between the corresponding LOT *a* and the corresponding NOLOT *C*, created by the two previous rules.

(4) the multiplicity associated to each attribute *a* is copied as the multiplicity of the role of the previous association (rule 3) attached to the NOLOT *a*. The opposite role of the same association must have a constraint multiplicity equal to one.

(5) all "simple" associations between classes are transformed into "simple" associations between NOLOTs.

(6) all generalisation relationships between classes are transformed into "simple" associations with a constraint multiplicity equal to one on the role attached to generalised NOLOT and a non-constraint multiplicity equal to * on the opposite role. In order to trace the fact that this association was derived from a generalisation, we annotate semantically the new corresponding association with a logical predicate. Moreover, the inheritance features of the generalisation association are mapped as new associations between LOTs representing the attributes of the generalised NOLOT, and all the specialised NOLOTs (sub-classes).

(7) composite aggregation and aggregation relationships are transformed into simple association (rule 3) that keep unchanged the existing roles' multiplicities but trace their specific semantics by an attached semantic annotation.

(8) association classes are transformed into a LOT fact with two associations linked to the corresponding initial LOT facts. The multiplicities of the roles of these two associations are determined by inverting the ones initially formalised on the roles of the previous association.

(9) any other specific constraints (generally modelled using OCL logical rules) are kept during the transformation process.

(10) we did not take into account the special cases of constraints in generalisations because they are not usually used in data conceptual modelling.

One of the conceptual modelling requirements is that a conceptual model must have formal foundations, which allow comparing that model with other conceptual models in a formal and exact way.

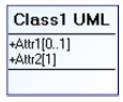
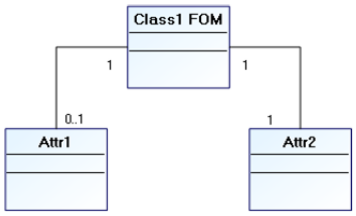
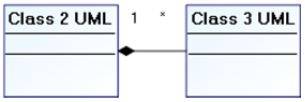
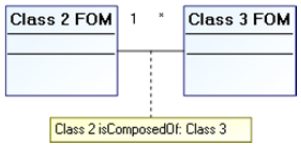
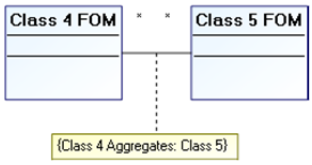
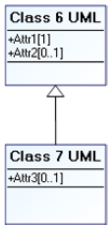
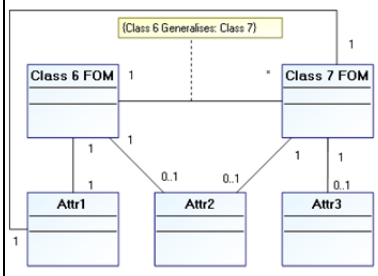
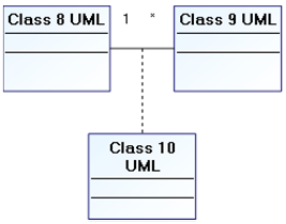
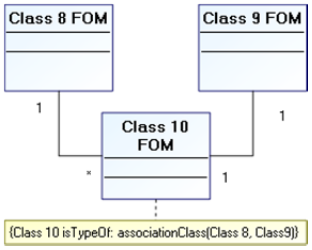
<table>
<tr><td colspan="2">Class and Attributes</td><td colspan="2">Composite aggregation</td></tr>
<tr><td>UML</td><td>FOM</td><td>UML</td><td>FOM</td></tr>
<tr><td> 
</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td colspan="2"></td><td colspan="2">Aggregation</td></tr>
<tr><td colspan="2"></td><td>UML</td><td>FOM</td></tr>
<tr><td colspan="2"></td><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Generalisation</td><td colspan="2">Association Class</td></tr>
<tr><td>UML</td><td>FOM</td><td>UML</td><td>FOM</td></tr>
<tr><td> 
</td><td> 
</td><td> 
</td><td> 
</td></tr>
</table>

Table 1 - Fact-Oriented modelling patterns using UML notation

3.4 FOL representation

A concern facing both developers and users of models is the degree of confidence in the model correctness. It is very easy to make errors, including errors in parameter estimations, in model assumptions and in programming. Verification methods are designed to address this question and have become important parts of model building process (Clarke et al., 1996), (Störrle, 2005).

As shown in Table1, a Fact-Oriented Model (FOM) uses semantic annotations to preserve coherence between the input and output models of transformation rules. Since those annotations are not formal objects, we need to verify the real semantics of the FOM model in comparison with the UML one.

(Berardi et al, 2005) and (Tursi et al, 2009) formalised UML class constructs semantics in First Order Logic (FOL) assertions. We propose to adapt and to extend these works to formalise the fact-oriented model patterns (presented in Table 1) in FOL assertions.

3.4.1 Translation of UML Artefacts and Semantic Annotations in FOL

1. UML artefacts

The conceptual models, produced after the first two steps of our approach, are representing concepts semantics using UML artefacts such as class, attribute, association, association class, aggregation, composite aggregation and generalisation. Let us now formalise each of these artefacts in FOL.

Class. A class in UML designates a set of object with common features (OMG, 2004). Formally a *class C* corresponds to a FOL *unary predicate C*.

$$\forall x\ C(x)\ . \qquad (1)$$

Attribute. An *attribute a* of type *T* for a *class C* associates to each instance of *C* a set of instances of *AttribType*, its multiplicity *[i..j]* specifies that *a* associates to each instance of *C* at least *i* and at most *j* instances of *AttribType*. Formally, an attribute *a* of type *AttribType* for *class C* corresponds to a binary predicate.

$$\forall x\ \forall y\ \big(C(x) \wedge a(x,y)\big) \rightarrow AttribType(y)\ . \qquad (2)$$

A multiplicity *[i..j]* is composed of two specifications: the minimal value (*i*) and the maximal value (*j*). Those minimal and maximal values are specified as “0” or “1” or “*” or

any positive integer. However, generally, in conceptual models, we do not specify positive integers for multiplicity because those numbers are embedded as constraints into the data processing of the software. We are then restraining our formalisation to the three generic cases: "1" (uniqueness), "0" (absence) and "*" (unlimited). This restraining postulate allows us to model each of the four general multiplicity cases (0..1, 1, 1..*, *) using a logical disjunction operator between the three logical expressions ((3), (4), (5)):

Uniqueness ("1"): for any instance *x* of class *C*, there is a unique value for the attribute *a* in the instance *x*.

$$\forall x\, C(x) \rightarrow \left(\exists y\, a(x,y) \wedge \forall z\, (a(x,z) \rightarrow y = z)\right). \quad (3)$$

Absence ("0"): for any instance *x* of class *C*, there is not any value for the attribute *a* in the instance *x*.

$$\forall x\, C(x) \rightarrow \forall y\, \neg a(x,y). \quad (4)$$

Unlimited (""):* for any instance *x* of class *C*, there is an unlimited number of values for the attribute *a* in the instance *x*.

$$\forall x\, C(x) \rightarrow \exists y\, a(x,y). \quad (5)$$

Association. An association in UML is a relation between two or more instances of classes. The multiplicity *[m..n]* attached to each role of an association specifies that each instance of the class *C* can participate at least *m* times and at most *n* times to the related association. The *n*-ary association construct may always be transformed into two or more binary associations. Thus an association A between two classes C_1 and C_2 is represented by a unary predicate A and two binary predicates r_1 and r_2, one for each role name, and can be formalised as the following set of FOL assertions:

$$\forall x\ \forall y\ A(x) \wedge\ r_1(x,y) \rightarrow C_1(y)\ . \quad (6)$$

$$\forall x\ \forall y\ A(x) \wedge\ r_2(x,y) \rightarrow C_2(y)\ . \quad (7)$$

$$\forall x\ A(x) \rightarrow \exists y\ r_1(x,y)\ . \quad (8)$$

$$\forall x\ A(x) \rightarrow \exists y\ r_2(x,y)\ . \quad (9)$$

$$\forall x\ \forall y\ \forall z\ A(x) \wedge r_1(x,y) \wedge r_1(x,z) \rightarrow y = \mathrm{z}\ . \quad (10)$$

$$\forall x\ \forall y\ \forall z\ A(x) \wedge r_2(x,y) \wedge r_2(x,z) \rightarrow y = \mathrm{z}\ . \quad (11)$$

$$\forall y_1\ \forall y_2\ \forall x\ \forall z\ A(x) \wedge A(z) \wedge_{i=1}^{2} \left(r_i(x,y_i) \wedge r_i(z,y_i)\right) \rightarrow x = \mathrm{z}\ . \quad (12)$$

Assertions (6) and (7) are typing the association. Assertions (8), (9), (10) and (11) are specifying that any association A has at least one role at each of its ends and that each role is unique; Assertion (12) imposes that each instance of an association A is unique.

The multiplicity constraints attached to association roles are formalised with logical expressions having the same structure as the previous ones ((3), (4), (5)) for the attributes.

Association Class. An association may have a related association class that describes properties of the association, such as attributes, operations, etc. It can be formalised in the same way of an association.

Aggregation. A particular kind of binary associations is aggregation, which plays an important role in UML class conceptual models. An aggregation is a binary relation between the instances of two classes C_1 and C_2, denoting a part-whole relation, i.e., a non-symmetric relation which specifies that each instance of a class (the containing class, C_1) contains a set of instances of another class (the contained class, C_2). The aggregation is represented by a unary predicate *Aggregation(x)* for which all the association assertions are valid. To complete its semantics, in order to formalise the fact that an instance cannot be its own aggregate, the following FOL assertion has to be added:

$$\forall x\ \forall y\ \forall z\ Aggregation(x) \wedge r_1(x,y) \wedge\ r_2(x,z) \wedge C_1(y) \wedge C_2(z) \rightarrow \neg(\ y = z)\ .$$

Moreover, in order to formalise the non-symmetry of the aggregation, the following FOL assertion has to be added:

$$\forall w\ \forall x\ \forall y\ \forall z\ Aggregation(x) \wedge Aggregation(w) \wedge \neg(\ x = w) \wedge r_1(x,y) \wedge r_2(x,z) \wedge C_1(y) \wedge C_2(z) \rightarrow \neg\ \big(r_1(w,z) \wedge\ r_2(w,y)\big)\ .$$

Composite aggregation. Composite aggregation is a strong form of aggregation that requires a component instance to be included in at most one composite at a time. The composite aggregation is represented by a unary predicate *Composition(x)* for which all the aggregation assertions are valid. To complete its semantics, in order to formalise the fact that a component cannot participate to more than one composite aggregation, the following FOL assertion has to be added:

$$\forall x\ \forall y\ \forall z\ Composition(x) \wedge r_1(x,y) \wedge\ r_2(x,z) \wedge C_1(y) \wedge C_2(z) \rightarrow (\forall w\ r_1(x,w) \rightarrow w = y)\ .$$

Moreover, in order to formalise the non-sharing property of the components, the following FOL assertion has to be added:

$$\forall u\ \forall w\ \forall x\ \forall y\ \forall z\ Composition(u) \wedge Composition(x) \wedge\ r_1(x,y) \wedge\ r_2(x,z) \wedge r_1(u,w) \wedge\ r_2(u,z) \wedge C_1(y) \wedge C_2(z) \wedge C_1(w) \rightarrow u = x\ .$$

Generalisation. In UML, one can use a generalisation 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 and the relationships of the parent class, but typically they also possess additional properties that do not hold for the parent class. Disjointness and completeness constraints can also be enforced on a class hierarchy. In this paper (and in our conceptualisation approach), we do not take into account

overlapping and incompleteness constraints over a class hierarchy because they generate hidden and poor semantics.

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

$$\forall x\ C_1(x) \rightarrow C(x)\ .$$

Disjointness among the child classes C_1 ... C_n is expressed by the additional FOL assertion:

$$\forall x\ C_i(x) \rightarrow \Lambda_{j=1,j\neq i}^{n} \neg C_j(x)\ for\ i = 1, \dots, n\ .$$

The completeness constraint, expressing that each instance of the parent class C is an instance of at least one of its child classes C_1 ... C_n, is formally defined by the additional assertion:

$$\forall x\ C(x) \rightarrow\ \vee_{i=1}^{n}\ C_i(x)\ .$$

2. Semantic Annotation in Fact-Oriented Model

In order to keep track of the initial model semantics, which may be lost sometimes after applying the FOM transformation rules on specific UML artefacts, we embed semantic annotations of some key modelling constructs (association class, generalisation, aggregation, composite aggregation) in resulting models. These annotations highlight, with the presented logic assertions, the specific semantics that can be lost. For the composite aggregation construct the semantic annotation brings also, in a textual form, the particular semantics of life cycle that relate to its instances.

This section is derived from the following scientific publications:

Lezoche M., Panetto H., Aubry A., (2011). Conceptualisation approach for cooperative information systems interoperability, ACM. 13th International Conference on Enterprise Information Systems, ICEIS 2011, Jun 2011, Beijing, China. pp. 101-110

Lezoche M., Panetto H., Aubry A., (2011). Formal Fact-Oriented model transformations for cooperative information systems semantic conceptualisation, Selected and extended version of ICEIS 2011. Lecture Notes in Business Information Processing, Proof read

4 A semantics structuring process

After conceptualising and enacting finest-grained semantics embedded into CISs models, resulting with a normalised FOM, the latter has to be structured into semantic aggregates (Yahia et al., 2011). Each of those identified aggregates represents a “semantic molecule”, composed of atomic concepts, with its own minimal mandatory semantics.

To build such aggregates, we propose a recursive approach for analysing the detailed semantics of the IS conceptual models obtained by the conceptualisation approach presented in section 3. We are considering that these models embed the whole explicited semantics of the associated IS.

Our structuring approach starts by identifying core atomic concepts and it ends by computing the semantic aggregates (namely, the semantic blocks) according to algorithms based on graph theory.

4.1 Core and extended semantics

When considering an available fact-oriented conceptual model from one IS (outputs from section 3), we can distinguish the mandatory (constrained) and non-mandatory (non-constrained) association roles, which represent mandatory and non-mandatory concepts expressing semantics.

The set of mandatory concepts represents all the necessary and sufficient elements which make the conceptual model semantically coherent and understandable. It comprises all the non-lexical and lexical concepts linked to constrained association roles with a multiplicity equal to 1 or 1..*. On the contrary, the non-mandatory concepts correspond to the non-mandatory roles (multiplicity equal to 0..1 or *) and are only enriching the semantics of those IS conceptual models.

To some extent, the set of mandatory concepts corresponds to the core semantics that is embedded into a given IS conceptual model. The extended semantics is defined by the set of mandatory and non-mandatory concepts.

4.2 Some mathematical definitions

We define, for each IS conceptual model, the following notations.

Definition 1. C_{IS} is the set of the identified lexical and non-lexical concepts, formally defined by

$$C_{IS} = \{c_i | c_i \text{ is a lexical or a non} - \text{lexical concept from the IS conceptual model}\}$$

Moreover, we define two subsets of C_{IS}as follows:

- NLC_{IS} is the subset of C_{IS} restricted to the non-lexical concepts and,
- LC_{IS} is the subset of C_{IS} restricted to the lexical concepts.

We can note that:

$$C_{IS} = NLC_{IS} \cup LC_{IS}$$

$$NLC_{IS} \cap LC_{IS} = \emptyset$$

Definition 2. Rel_{IS} is the set of the identified associations between concepts. Formally, it is defined by

$$Rel_{IS} = \{rel(c_i, c_j) | (c_i, c_j) \in (C_{IS})^2 \ \wedge \ c_i \ is \ associated \ to \ c_j \}$$

Definition 3. $Mult\left(rel(c_i, c_j)\right)$ is the multiplicity of the role of c_j when considering the association between c_i and c_j if it exists. For each $(c_i, c_j) \in (C_{IS})^2$, if $rel(c_i, c_j)$ exists then we have $Mult\left(rel(c_i, c_j)\right) \in \{*, 0..1, 1, 1..*\}$ and it is read c_j is associated to c_i with a multiplicity equal to $Mult\left(rel(c_i, c_j)\right)$.

Definition 4. MC_{IS} is the subset of C_{IS} restricted to mandatory concepts (the core semantics). It is formally defined by

$$MC_{IS} = \left\{ c_i \middle| \exists \left(c_i, rel(c_i, c_j) \right) \in C_{IS} \times Rel_{IS} \wedge Mult \left(rel(c_i, c_j) \right) \in \{1,1..*\} \right\}$$

Moreover, we define two subsets of C_{IS} as follows:

- $MNLC_{IS}$ is the subset of C_{IS} restricted to the mandatory non-lexical concepts and,
- MLC_{IS} is the subset of C_{IS} restricted to the mandatory lexical concepts.

We can note that:

$$MC_{IS} = MNLC_{IS} \cup MLC_{IS}$$

$$MNLC_{IS} \cap MLC_{IS} = \emptyset$$

$$MNLC_{IS} = MC_{IS} \cap NLC_{IS}$$

$$MLC_{IS} = MC_{IS} \cap LC_{IS}$$

Definition 5. For each non-lexical concept c_j, we can define the set of its associated mandatory lexical concepts as follows:

$$MLC(c_j) = \left\{ c_i \in LC_{IS} \middle| \left(\exists rel(c_j, c_i) \in Rel_{is} \middle| Mult \left(rel(c_j, c_i) \right) \in \{1,1..*\} \right) \right\}$$

Definition 6. For each non-lexical concept c_j, we can define the set of its associated mandatory non-lexical concepts as follows:

$$MNLC(c_j) = \left\{ c_i \in NLC_{IS} \middle| \left(\exists rel(c_j, c_i) \in Rel_{is} \middle| Mult \left(rel(c_j, c_i) \right) \in \{1,1..*\} \right) \right\}$$

If we consider a concept defined in the context of the IS core semantics, we notice that, in order to be semantically effective in the studied domain, this concept needs to be associated

on the one hand to its mandatory lexical concepts and on the other hand to other non-lexical concepts. This defines the notion of Semantic Block (SB).

4.3 Semantic blocks identification

1. *Definition*

Considering a particular non-lexical concept c_i from NLC_{IS}, a semantic block, denoted as $SB(c_i)$ and associated with the concept c_i, represents the set of the concepts necessary for the minimal semantics definition of the non-lexical concept c_i given by the conceptual model. Formally, $SB(c_i)$ is defined as follows:

$$SB(c_i) = \left\{ c_i \cup MLC(c_i) \bigcup_{c_j \in MNLC(c_i)} SB(c_j) \right\} \quad (1)$$

This definition, suggests that the notion of semantic block is recursive.

In the following, the meta-model of the semantic block is given and a procedure to compute all the semantic blocks of a conceptual model is proposed.

2. *Semantic block meta-model*

Here we propose to formalise the semantic block architecture through the meta-model represented on Figure 3. This meta-model is based on the composite pattern (Gamma et al, 1995). This meta-model defines an arborescence of components representing hierarchies of objects.

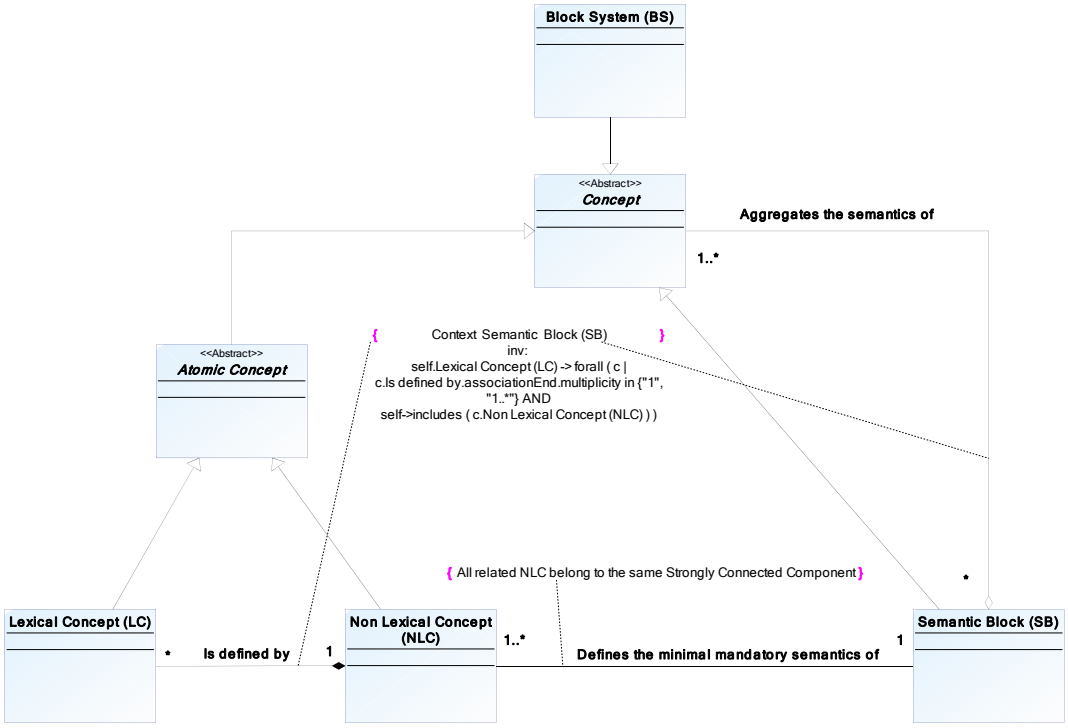


Figure 3 - Meta-model of the semantic block structure

A semantic block defines the minimal mandatory semantics of one or several non-lexical concepts such that these concepts are in the same strongly connected component[1]. Moreover, the semantics of one or several concepts can be aggregated into one or several semantic blocks. As the semantic block is a specialisation of the abstract class "Concept", its semantics can be aggregated into one or several semantic blocks of higher levels. The Block System represents the last level of aggregation and contains the minimal mandatory semantics of the studied IS conceptual model.

3. *How to build the Semantic blocks?*

Let us consider the conceptual model on Figure 4 and its transformation on Figure 5 obtained by applying the third step presented in section 3. Let us build the semantic block of the concept $C2$. The intrinsic mandatory semantics of the concept $C2$ is defined by the

[1] A strongly connected component of a directed graph is a maximal set of vertices such that for every pair of vertices *u* and *v*, there is a directed path from *u* to *v* and a directed path from *v* to *u*.

semantics of the mandatory lexical concepts that are associated to it, namely $A1C2$ and $A2C2$. Moreover, a given instance of the concept $C2$ exists only if it is associated to at least one instance of the concept $C5$. That means that $C5$ is mandatory for expressing the semantics of $C2$. Moreover, considering the roles of $C1$ and $C3$ in their association with $C2$, we can see that the minimal multiplicity is equal to 0. That means that the existence of any instance of $C2$ is not stipulated by the existence of one instance of $C1$ or $C3$. Finally, we find again $SB(C2) = \{C2 \cup \{A1C2, A2C2\} \cup SB(C5)\}$ as in equation (1).

Recursively, we can demonstrate that the intrinsic mandatory semantics of the concept $C5$ is defined by the semantics $A1C5$ and that a given instance of the concept $C5$ exists only if it is associated to exactly one instance of the concept $C8$ and exactly one instance of the concept $C2$. That means that $SB(C5) = \{C5 \cup \{A1C5\} \cup SB(C2) \cup SB(C8)\}$.

Applying the same reasoning, we can build $SB(C8)$ as follows: $SB(C8) = \{C8 \cup \{A1C8\}\}$. Finally we can deduce that: $SB(C2) = \{\{C2, C5, C8\} \cup \{A1C2, A2C2\} \cup \{A1C5\} \cup \{A1C8\}\}$.

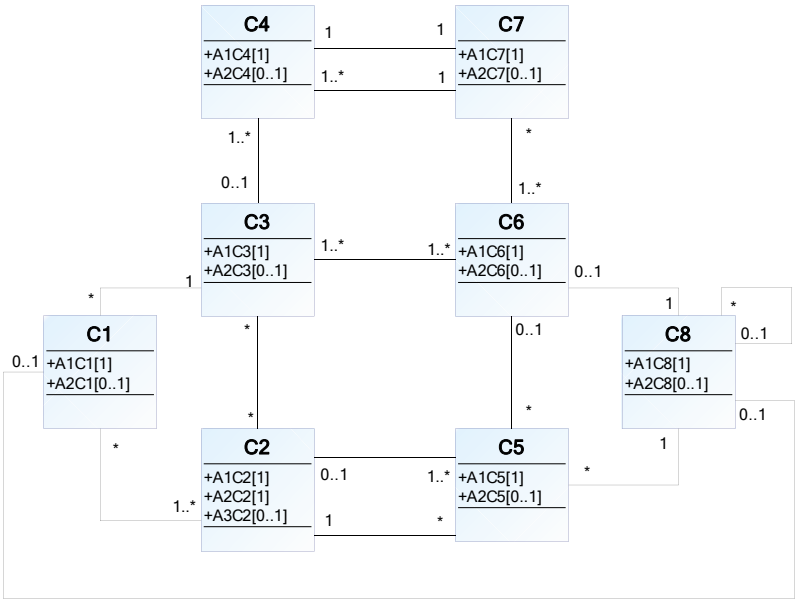


Figure 4 - An instance of conceptual model

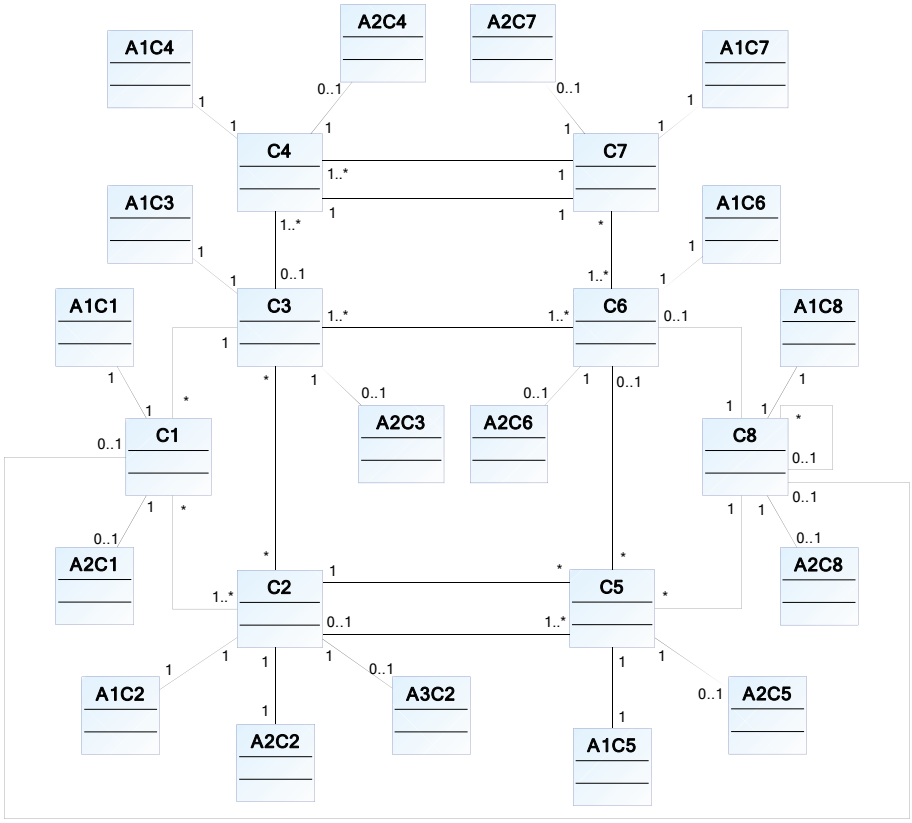


Figure 5 - "Fact-oriented modelling" transformation of the model of Figure 4

To simplify the computation of the semantic block of one concept c_i, we propose, first, to identify the set of non-lexical concepts that are included in the semantic block and, second, to add the associated mandatory lexical concepts. That means that $SB(c_i)$ is determined as follows: $SB(c_i) = SB_c(c_i) \cup SB_a(c_i)$ with

- $SB_c(c_i) = \left\{\{c_i\} \cup_{c_j \in MNLC(c_i)} SB_c(c_j)\right\}$ and,
- $SB_a(c_i) = \{MLC(c_j) | c_j \in SB_c(c_i)\}$

For instance, $SB_c(C2) = \{C2, C5, C8\}$ and $SB_a(C2) = \{A1C2, A2C2, A1C5, A1C8\}$.

4.4 Using graph theory for building $\mathbf{SB_c(c_i)}$

To facilitate the building of the semantic blocks, we propose, for each c_i from NLC_{IS}, to identify the associated set $SB_c(c_i)$ by using graph theory modelling and its associated mathematical tools.

Let us first define a semantic-dependency graph associated with a conceptual model. This semantic-dependency graph is a digraph $G = (V, E)$ where V is the set of nodes and E is the set of edges defined by a pair of nodes. Each node from V represents a non-lexical concept of the conceptual model. Each edge from E is built from the conceptual model as follows: the edge (c_i, c_j) exists if (i) there is an association between c_i and c_j in the conceptual model, and (ii) if the minimal multiplicity for the role of c_j is equal to 1 $\left(c_j \in MNLC(c_i)\right)$. That means that the existence of the edge (c_i, c_j) represents the fact that c_j is mandatory for expressing the semantics of c_i.

The Figure 6 shows the semantic-dependency graph associated with the conceptual model of the Figure 5.

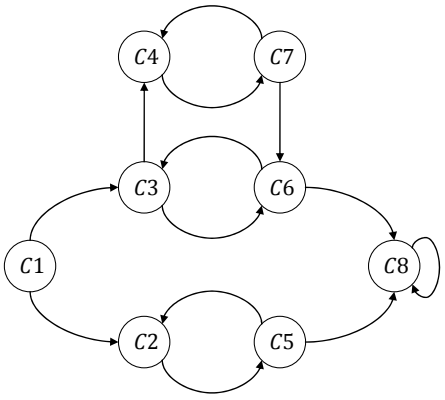


Figure 6 - Semantic-dependency graph associated with the conceptual model of Figure 5

Theorem 1. Given two particular concepts c_i and c_j, c_j belongs to $SB_c(c_i)$ if and only if there exists a directed path from c_i to c_j.

Proof. Let us consider the conceptual model on Figure 4. To build the semantic block of the concept c_i, we consider this concept as the starting point. This concept can thus be considered as the root in the associated semantic-dependency graph. Now we add in $SB_c(c_i)$ all the concepts c_k that must be instantiated to ensure the existence of a particular instance of c_i, i.e. all the concepts c_{1k} such that there is an association between c_i and c_{1k} in the conceptual model, and the minimal multiplicity for c_{1k}, considering this association, is equal to 1. This is the exact definition of all the successors of c_i in the semantic-dependency graph. Note that, by definition, there is a directed path from the concept c_i to these concepts c_{1k}. Iteratively, the only new concepts c_{2k} that can be added to $SB_c(c_i)$ are the successors of those first concepts c_{1k}. As successors of the concepts c_{1k}, there exists also a directed path from the concept c_i to the concepts c_{2k} (the path from c_i to c_{1k} plus the edge (c_{1k}, c_{2k})). Finally the semantic block of c_i contains exactly all the concepts c_j such that there exists a directed path from c_i to c_j. ■

Theorem 2. Given two particular concepts c_1 and c_2, if c_2 belongs to $SB_c(c_1)$ then $SB_c(c_2)$ is included in $SB_c(c_1)$.

Proof. c_2 belongs to $SB_c(c_1)$ means that there exists a path from c_1 to c_2 (see theorem 1). Let us now consider a particular concept from $SB_c(c_2)$ denoted as c. By definition of $SB_c(c_2)$, there exists a path from c_2 to c and then a path from c_1 to c (the path from c_1 to c_2 plus the path from c_2 to c). That means that c is in $SB_c(c_1)$. Finally $SB_c(c_2) \subseteq SB_c(c_1)$. ■

Theorem 3. All the concepts that are in the same cycle in the semantic-dependency graph are associated with the same unique semantic block.

Proof. A cycle is a closed path. Let us consider two particular concepts, denoted as c_i and c_j, which belong to a cycle. In particular there is a path from c_i to c_j. That means that c_j is in $SB_c(c_i)$. Following the theorem 2, we can also demonstrate that $SB_c(c_j) \subseteq SB_c(c_i)$. Moreover, there is a path from c_j to c_i. That means that c_i is in $SB_c(c_j)$. Following the theorem 2, that means that $SB_c(c_j) \supseteq SB_c(c_i)$. Finally, $SB_c(c_j) = SB_c(c_i)$. ■

The theorem 3 implies that there is one semantic block per strongly connected component of the semantic-dependency graph.

4.5 A procedure to compute the semantic blocks

Applying theorems 1 to 3, we propose the following procedure to compute all the semantic blocks of a given conceptual model:

i. Building the associated semantic-dependency graph.

ii. Building the graph of the strongly connected components based on the semantic-dependency graph.

iii. Computing the semantic blocks SB_c associated with each strongly connected component.

iv. Computing, for each SB_c, the semantic block SB_a by adding all the mandatory lexical concepts associated to each non-lexical concept from SB_c.

v. Computing $SB = SB_c \cup SB_a$.

These steps are detailed as follows.

4.5.1 Building the associated semantic-dependency graph

By definition of this graph, it can be easily obtained by considering each association between two concepts c_i and c_j and then building an edge from c_i to c_j if the minimal multiplicity for the role of c_j is equal to 1.

4.5.2 Building the graph of the strongly connected components

Theorem 3 implies that for building the semantic blocks, we can consider only one concept in a given strongly connected component (the other concepts share the same semantic block). That is the reason why we can simplify the semantic-dependency graph by considering only an equivalent graph where the nodes represent each strongly connected component of the former semantic-dependency graph, and where one of these nodes (e.g. SCC1) is connected

to another node (e.g. SCC2) if there exists at least one edge from a concept from SCC1 to a concept from SCC2.

Identifying all the strongly connected components of a graph is a well-known problem in graph theory that can be solved with polynomial effort by using Kosaraju-Sharir's algorithm (Sharir, 1981).

The graph of the strongly connected components related to the semantic-dependency graph of Figure 6 is given on Figure 7. On this graph, the strongly connected components are defined as follows $\mathrm{SCC1} = \{\mathrm{C1}\}$, $\mathrm{SCC2} = \{\mathrm{C2}, \mathrm{C5}\}$, $\mathrm{SCC3} = \{\mathrm{C3}, \mathrm{C4}, \mathrm{C6}, \mathrm{C7}\}$ and $\mathrm{SCC4} = \{\mathrm{C8}\}$.

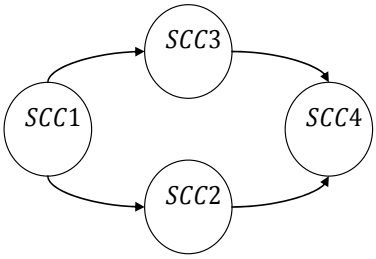


Figure 7 - Graph of the strongly connected components related to the graph of Figure 6

4.5.3 Computing SB_c associated with each strongly connected component

We propose now one algorithm for computing all the semantic blocks $\mathrm{SB_c}$ associated with each strongly connected component (see Algorithm 1 that invokes Algorithm 2). The algorithm 1 BuildSemBlocks is applied on the graph of the strongly connected components (denoted as $\mathrm{G_{SCC}}$).

Let us apply the algorithm BuildSemBlocks($\mathrm{G_{SCC}}$) on the graph of Figure 7. We obtain the following semantic blocks:

- $SB_c(SCC1) = SCC1 \cup SCC2 \cup SCC3 \cup SCC4$,
- $SB_c(SCC2) = SCC2 \cup SCC4$,
- $SB_c(SCC3) = SCC3 \cup SCC4$ and

- $SB_c(SCC4) = SCC4$.

And finally replacing the strongly connected components by their content we obtain the following semantic blocks:

- $SB_c(C1) = \{C1, C2, C3, C4, C5, C6, C7, C8\}$,
- $SB_c(C2, C5) = \{C2, C5, C8\}$,
- $SB_c(C3, C4, C6, C7) = \{C3, C4, C6, C7, C8\}$ and
- $SB_c(C8) = \{C8\}$.

Algorithm $BuildSemBlocks(G_{SCC})$

[Initialisation]
L: List of the strongly connected components in G_{SCC}
For each $SCC \in L$ **Do**
    $color(SCC) = -1$
    [color is an indicator that defines if a node SCC has already been visited or not]
    [-1 means not yet visited]
    [0 means being visited]
    [+1 means already visited]
Next SCC
For each $SCC \in L$ **Do**
    If $color(SCC) = -1$ **Then**
        [Building of the semantic block associated with SCC]
        $BuildSB(SCC)$*[calling Algorithm 2]*
    EndIf
Next SCC
Return

Algorithm 1. *BuildSemBlocks* algorithm

Algorithm $BuildSB(SCC)$

[Initialisation]
$SB_c(SCC) = SCC$ *[The semantic block associated with SCC initially contains all the concepts in the SCC]*
$color(SCC) = 0$ *[SCC is being visited]*
[Building]
[use of theorem 1]
For each SCC' successor from SCC in G_{SCC} **Do**
  If $color(SCC') = -1$ **Then**
    [Building of the semantic block associated with SCC']
    $BuildSB(SCC')$
  EndIf
  [Use of theorem 2]
  $SB_c(SCC) = SB_c(SCC) \cup SB_c(SCC')$
Next SCC' successor from SCC in G_{SCC}
Return $SB_c(SCC)$

Algorithm 2. *BuildSB* algorithm

4.5.4 Computing, for each SB_c, the semantic block SB_a

Each semantic block SB_a contains the mandatory lexical concepts associated to the non-lexical concepts in SB_c. By applying the definition of SB_a $\left(SB_a(c_i) = \left\{MLC\left(c_j\right) \middle| c_j \in SB_c(c_i)\right\}\right)$ on the instance of Figure 5 we obtain:

- $SB_a(\mathrm{C1}) = \{A1C1, A1C2, A2C2, A1C3, A1C4, A1C5, A1C6, A1C7, A1\mathrm{C8}\}$,
- $SB_a(\mathrm{C2}, \mathrm{C5}) = \{A1C2, A2C2, A1C5, A1\mathrm{C8}\}$,
- $SB_c(\mathrm{C3}, \mathrm{C4}, \mathrm{C6}, \mathrm{C7}) = \{A1C3, A1C4, A1C6, A1C7, A1\mathrm{C8}\}$ and
- $SB_c(\mathrm{C8}) = \{A1\mathrm{C8}\}$.

4.5.5 Computing each semantic block SB

Each semantic block SB is the union of SB_c and SB_a. By applying this definition on the instance of Figure 5 we obtain:

- $SB(C1) =$

$\{\{C1, C2, C3, C4, C5, C6, C7, C8\} \cup \{A1C1, A1C2, A2C2, A1C3, A1C4, A1C5, A1C6, A1C7, A1C8\}\}$,

- $SB(\mathrm{C2}, \mathrm{C5}) = \{\{C2, C5, C8\} \cup \{A1C2, A2C2, A1C5, A1\mathrm{C8}\}\}$,
- $SB(\mathrm{C3}, \mathrm{C4}, \mathrm{C6}, \mathrm{C7}) = \{\{C3, C4, C6, C7, C8\} \cup \{A1C3, A1C4, A1C6, A1C7, A1\mathrm{C8}\}\}$ and
- $SB(\mathrm{C8}) = \{C8\} \cup \{A1\mathrm{C8}\}$.

For validating our approach, next section will detail an industrial case study involving two enterprise information systems that need to interoperate: Sage X3 ERP and Flexnet MES.

This section is derived from the following scientific publication:

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Semantics enactment in Enterprise Information Systems, IFAC. 18th IFAC World Congress, IFAC WC'2011, Aug 2011, Milan, Italy. Elsevier - IFACPapersOnline, 18, 13064-13073

5 Case Study

Interoperability between organisational and manufacturing activities is crucial in manufacturing enterprises. Production services have to produce, quickly and efficiently, the right volume of the right product at the right moment. For this reason, they need real time information coming from others services, which need in return a precise and updated data on production. We propose here to study and present the first part of such a B2M interoperability issue by considering Sage X3 as an Enterprise Resource Planning (ERP) application and Flexnet as a Manufacturing Execution System (MES). Such interoperation process is based on a deep semantics analysis of their own models. In order to illustrate our approach, we will detail the conceptualisation process applied to a subset of the ERP information system model in section 5.1. Section 5.2 will detail the semantics structuration process (computing semantic blocks) applied to a subset of the MES information system model. For sake of readability, in the following, we will name each subset of models by the name of the related enterprise applications.

5.1 Conceptualisation of Sage X3 ERP model

An Enterprise Resource Planning (ERP) system is an integrated computer-based system which is used to manage internal and external resources including tangible assets, financial resources, materials, and human resources (Bidgol, 1997). Its purpose is to facilitate the flow of information between all business functions inside the boundaries of the organization, as well as to manage the connections to outside stakeholders. Built on a centralised database, ERP systems integrate all business operations into a uniform system environment. Sage X3 provides different enterprise management functions: finance, commercial, industrial and services.

The objectives of this case study are (i) to analyse how the manufacturing order process inside the Sage X3 application is modelled, (ii) to use the proposed modelling process for making the implicit knowledge explicit in the model structure.

The model depicted on Figure 8 is the output from the first two steps of our approach. This means that we have already completed the “Reverse Engineering” and the “Expert knowledge injection” steps. The “Manufacturing Order Heading” concept represents the management function of production orders and planned activities. This function allows the generation of a manufacturing order by variation of one or more classifications and a single production line. For each manufacturing order, the achievement of the material benefits and sequencing operations is possible. The function captures general information, such as planning and production facility and the status of the order. It allows entering general information about the production order. The availability of components is then checked through the information given by the bill of material related to the launched products. Once the above initial information is determined, the system updates the list of materials and operations of the created or modified orders.

Step 1: Reverse Engineering

All this information is stored in the Sage X3 application database. The first step of our method is to reverse-engineer the database, in order to extract the initial conceptual model by using standard tools in MEGA Suite. In this particular case, the resulting model is composed only of a set of classes and attributes without any associations. This is due to the fact that all relationships between concepts are directly implemented into the application software instead of in the database. We can note also that the implementation names of the entities (coming from tables and columns) are quite raw and not expressive. The bottom of Figure 9 shows two classes extracted from the reverse-engineered conceptual model. The objective of the next step is then to clean and enrich this model.

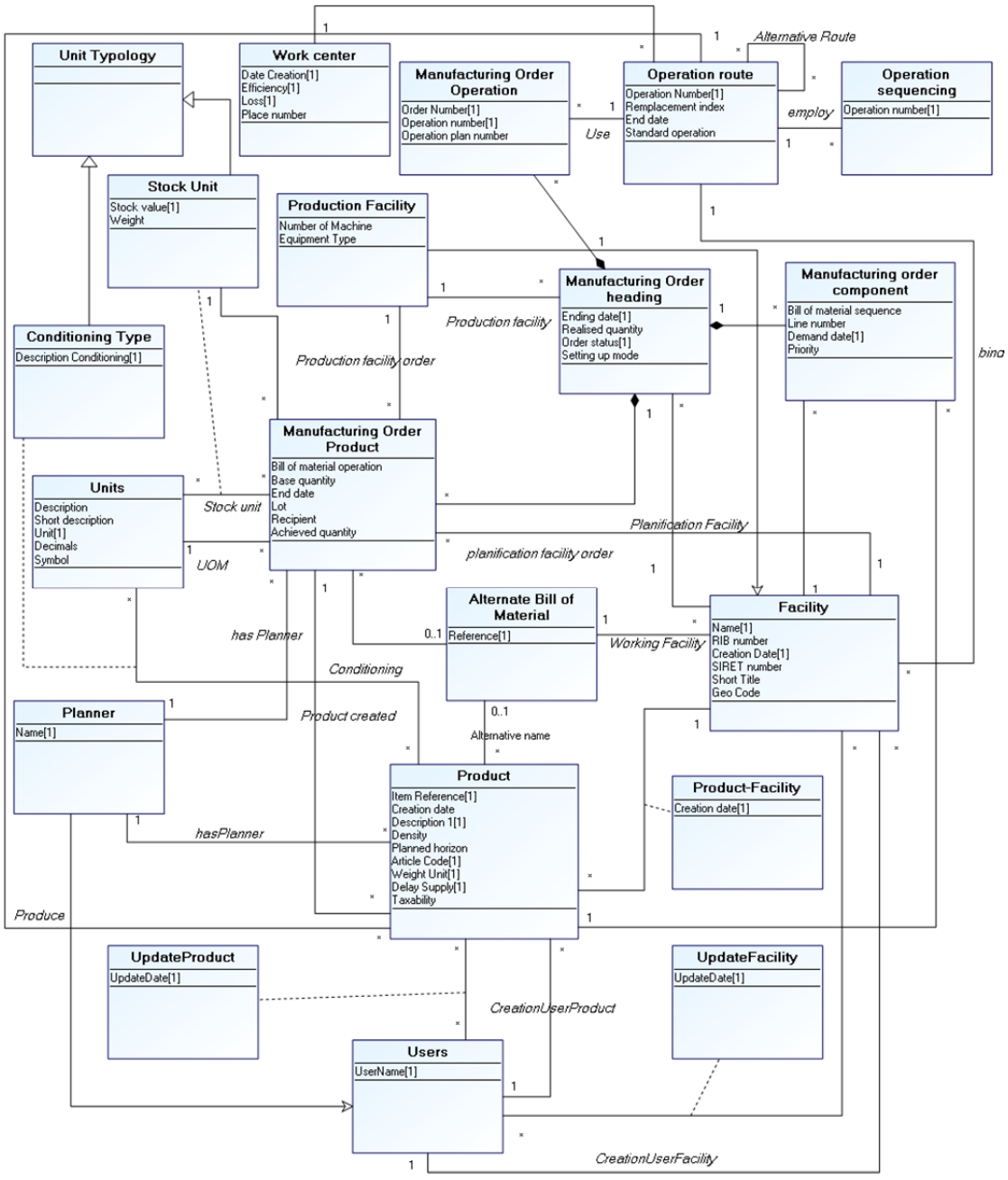


Figure 8 - Enriched Sage X3 manufacturing order process model

Step 2: Expert Knowledge Injection

The model depicted in Figure 8 is the result of the reverse engineering step and is enriched by a domain expert. In this case, the enrichment process involved a significant human effort

because the architecture of the Sage X3 ERP is built with all the database relationships implemented directly into the application layer and not in the database. The reverse engineering step results, as shown in the lower part of the Figure 9, in a model containing classes with coded names and no associations.

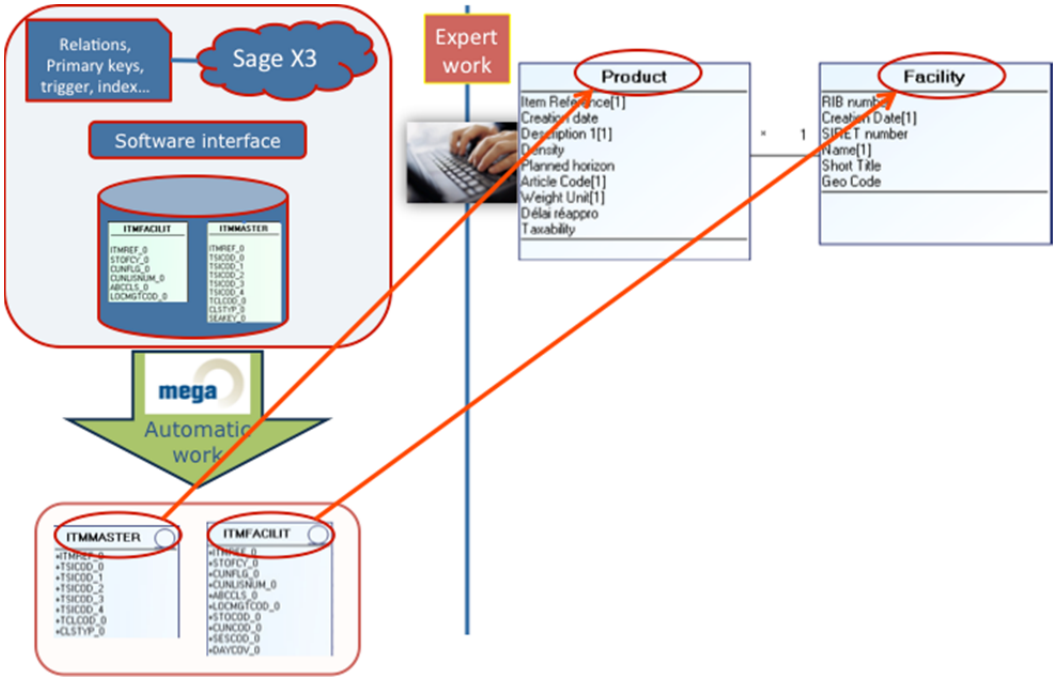


Figure 9 - Sage X3 architecture and expert knowledge injection

The expert work is about cleaning this conceptual model according to the best practices in the enterprise, modifying the attributes multiplicity (if needed), renaming the concepts, the attributes and the associations to fit the conceptual model to the "real" use of the Enterprise Information System. The typical case that requires the domain expert attention is the mandatory properties in forms' fields.

Let us consider the same example as in section 3.2. In the studied enterprise, a good practice is imposed for achieving information update traceability. When a user updates information concerning one product, the application must store, in dedicated fields, the date of the update and the name of the logged user. This feature is implemented directly into Sage X3 ERP but it is not reflected in the data model. Moreover, for the sake of simplifying the

implementation, the developers did not set these attributes as mandatory. In order to consider this practice in the conceptual model, the expert can conceptualise this constraint as one mandatory association between the existing concepts Product and Users and by constraining the existing attribute UpdateDate in the association class related to the previous association (as on Figure 9).

Step 3: Fact-Oriented Transformation

Applying the pattern transformation rules, presented in the previous section, classes and attributes are transformed into NOLOTs and LOTs respectively to increase the granularity of the knowledge embedded into the model. These rules have been coded by using a Mega Suite internal version of VBA scripting language and then automatically executed inside MEGA Suite.

Figure 10 shows the resulting FOM after applying our approach to the Sage X3 work order process.

At the first glance, it seems that the resulting model is much more complex than the initial one. This may looks true from a visual point of view, but it is false in terms of expressiveness of the model's semantics. Indeed, the finest-grained atoms of semantics are now made explicit, which helps any automatic computing. An important result is that using the model with such high level of granularity will facilitate automatic execution for semantic gap evaluation.

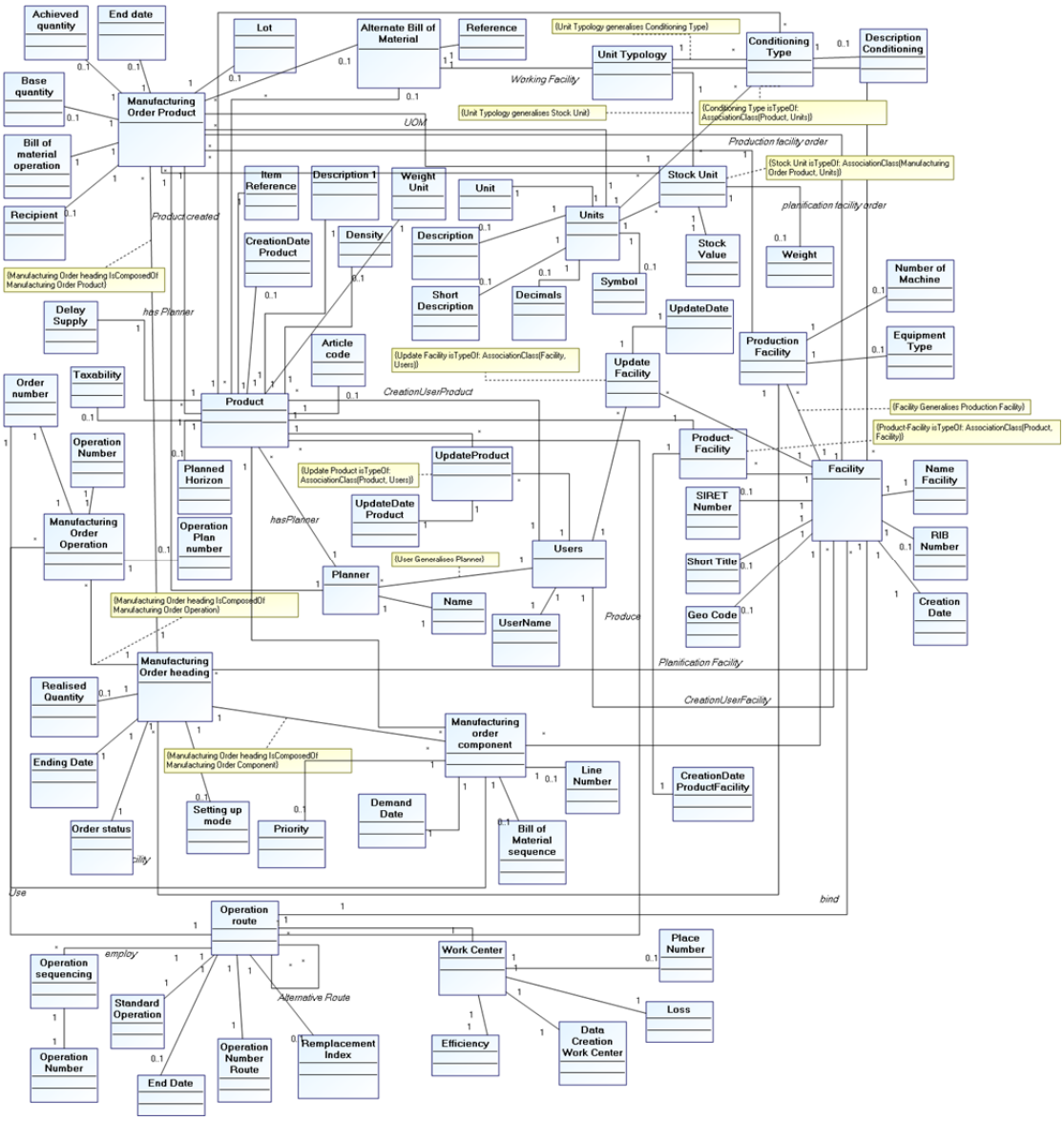


Figure 10 - Sage X3 manufacturing order process model - fact-oriented version

5.2 Semantics structuring of Flexnet MES model

Manufacturing Execution Systems (MES) are information technology systems that manage manufacturing operations in factories. Actually, a specific process implemented in Flexnet application has been chosen to support our validation process; it consists of the purchase order process. Figure 11 represents the enriched fact-oriented model of this process. Note that, in this model, classes with capital letters represent the non-lexical concepts. In order to

compute the semantic blocks for structuring the model semantics of this process, we apply the procedure presented in section 4.5.

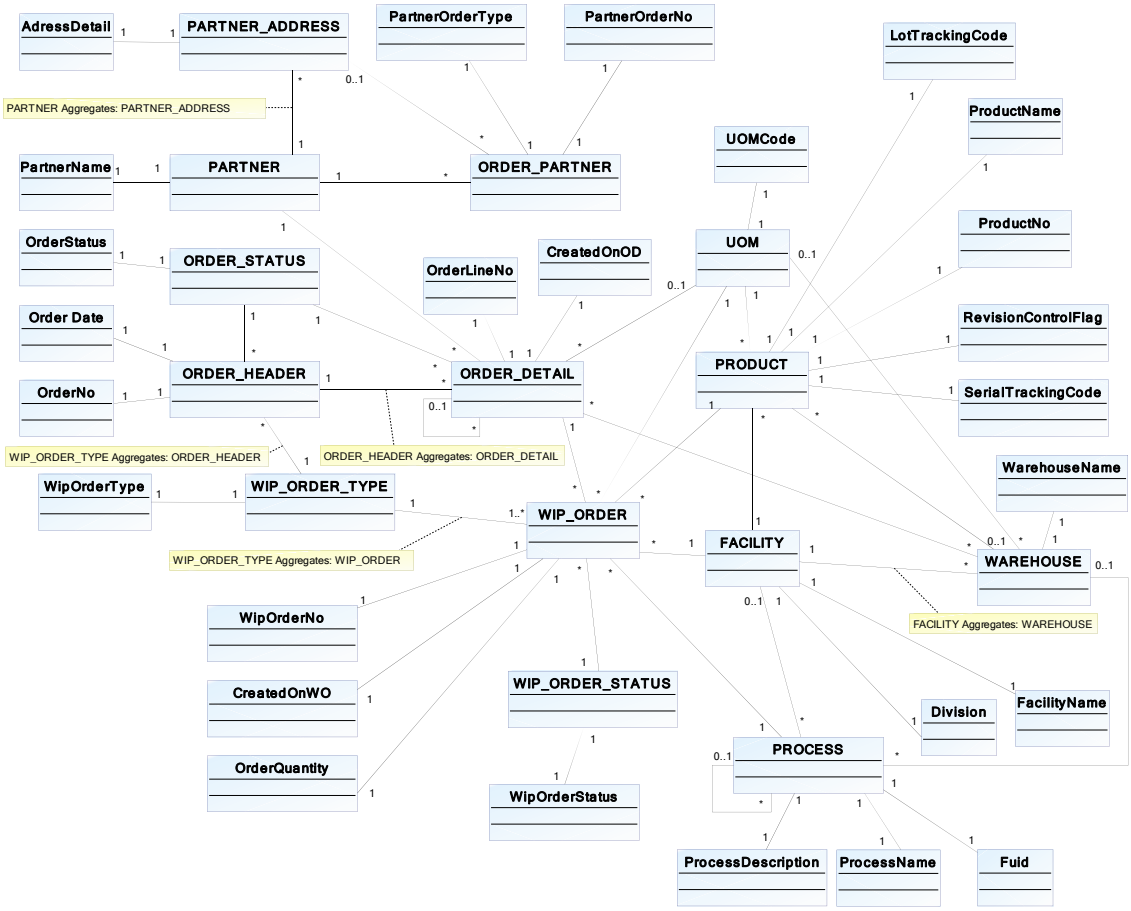


Figure 11 – Enriched fact-oriented model of the purchase order process in Flexnet application

1) Building the associated semantic-dependency graph

The semantic-dependency graph related to the conceptual model of Flexnet is given on Figure 12.

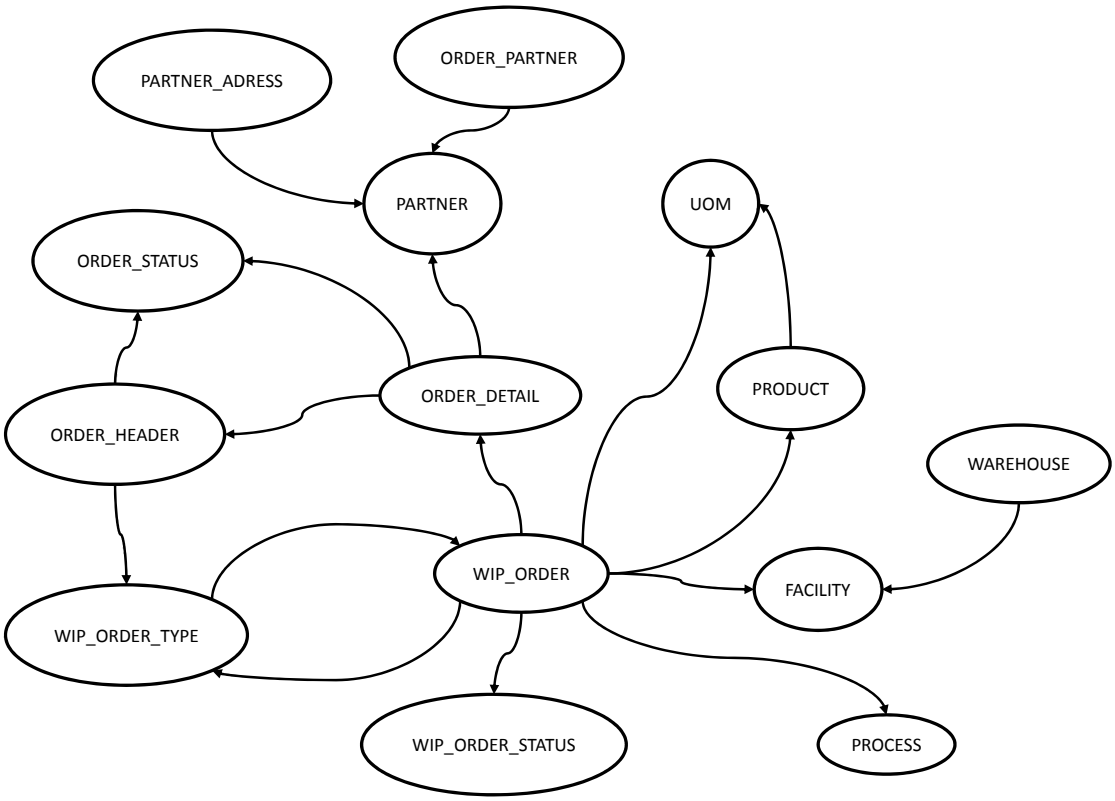


Figure 12 - Semantic-dependency graph related to the conceptual model of Flexnet MES

2) Building the graph of the strongly connected components based on the semantic-dependency graph

The graph of the strongly connected components related to the semantic-dependency graph of Flexnet MES is given on Figure 13. We can note that only one merged node has been built (namely SCC1) representing the strongly connected components of the concepts: *WIP_ORDER*, *WIP_ORDER_TYPE*, *ORDER_DETAIL* and *ORDER_HEADER*. All the other strongly connected components consist of only one concept.

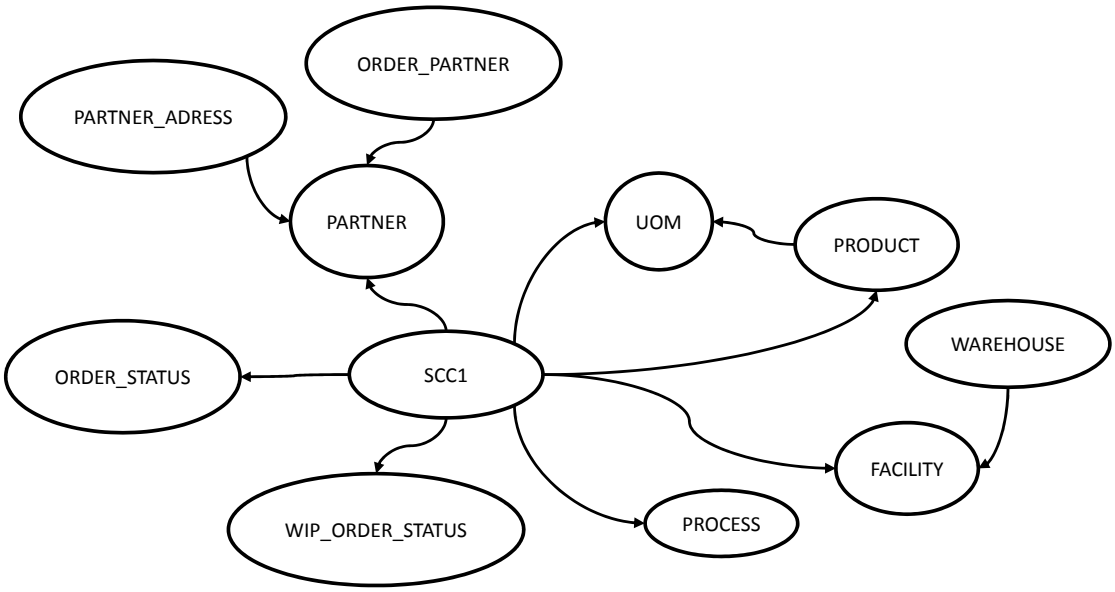


Figure 13 - Graph of the strongly connected components related to the semantic-dependency graph of Flexnet MES

3) Computing the semantic blocks SB_c associated with each strongly connected component

Table 2 lists the different semantic blocks SB_c related to Flexnet application after applying algorithm 1 (*BuildSemBlocks*) to the graph of the strongly connected components on Figure 13.

SB_c	***Concepts***
SB_c(WAREHOUSE)	WAREHOUSE, FACILITY
SB_c(ORDER_PARTNER)	ORDER_PARTNER, PARTNER
SB_c(PARTNER_ADDRESS)	PARTNER_ADDRESS, PARTNER
SB_c(PARTNER)	PARTNER

$SB_C(SCC1) = SB_c\begin{pmatrix}\text{WIP\_ORDER},\\ \text{ORDER\_DETAIL},\\ \text{ORDER\_HEADER},\\ \text{WIP\_ORDER\_TYPE}\end{pmatrix}$	WIP_ORDER, WIP_ORDER_TYPE,ORDER_DETAIL, ORDER_HEADER, WIP_ORDER_STATUS, PROCESS, FACILITY, PRODUCT, UOM, ORDER_STATUS, PARTNER
$SB_c(\text{PROCESS})$	PROCESS
$SB_c(\text{PRODUCT})$	PRODUCT, UOM, FACILITY
$SB_c(\text{UOM})$	UOM
$SB_c(\text{WIP\_ORDER\_STATUS})$	WIP_ORDER_STATUS
$SB_c(\text{FACILITY})$	FACILITY
$SB_c(\text{ORDER\_STATUS})$	ORDER_STATUS

Table 2 - Semantic Blocks (SB_c) of Flexnet MES

4) Computing, for each SB_c, the related semantic block SB_a

Table 3 lists the different semantic blocks SB_a related to Flexnet application.

SB_a	***Concepts***
$SB_a(\text{WAREHOUSE})$	WarehouseName, FacilityName, Division
$SB_a(\text{ORDER\_PARTNER})$	PartnerOrderNo, PartnerOrderType, PartnerName
$SB_a(\text{PARTNER\_ADDRESS})$	AdressDetail, PartnerName
$SB_a(\text{PARTNER})$	PartnerName

$SB_a\begin{pmatrix}\text{WIP\_ORDER},\\ \text{ORDER\_DETAIL},\\ \text{ORDER\_HEADER},\\ \text{WIP\_ORDER\_TYPE}\end{pmatrix}$	WipOrderNo, CreatedOnWO, OrderQuantity, WipOrderType , OrderLineNo, CreatedOnOD, OrderDate, OrderNo, WipOrderStatus, ProcessName, ProcessDescription, Fuid, FacilityName, Division, LotTrackingCode, ProductName, ProductNo, RevisionControlFlag, SerialTrackingCode, UOMCode, OrderStatus, PartnerName
$SB_a(\text{PROCESS})$	ProcessName, ProcessDescription, Fuid
$SB_a(\text{PRODUCT})$	LotTrackingCode, ProductName, ProductNo, RevisionControlFlag, SerialTrackingCode , UOMCode, FacilityName, Division
$SB_a(\text{UOM})$	UOMCode
$SB_a(\text{WIP\_ORDER\_STATUS})$	WipOrderStatus
$SB_a(\text{FACILITY})$	FacilityName, Division
$SB_a(\text{ORDER\_STATUS})$	OrderStatus

Table 3 - Semantic Blocks (SB_a) of Flexnet MES

5) Computing $SB = SB_c \cup SB_a$

Table 4 lists the different semantic blocks SB related to Flexnet application.

SB	***Concepts***
$SB(\text{WAREHOUSE})$	WAREHOUSE, WarehouseName, FACILITY, FacilityName, Division

$SB(\text{ORDER\_PARTNER})$	ORDER_PARTNER, PartnerOrderNo, PartnerOrderType, PARTNER, PartnerName
$SB(\text{PARTNER\_ADDRESS})$	PARTNER_ADDRESS, AdressDetail, PARTNER, PartnerName
$SB(\text{PARTNER})$	PARTNER, PartnerName
$SB\begin{pmatrix}\text{WIP\_ORDER},\\ \text{ORDER\_DETAIL},\\ \text{ORDER\_HEADER},\\ \text{WIP\_ORDER\_TYPE}\end{pmatrix}$	WIP_ORDER, WipOrderNo, CreatedOnWO, OrderQuantity, WIP_ORDER_TYPE, WipOrderType, ORDER_DETAIL, OrderLineNo, CreatedOnOD, ORDER_HEADER, OrderDate, OrderNo, WIP_ORDER_STATUS, WipOrderStatus, PROCESS, ProcessId, ProcessDescription, Fuid, FACILITY, FacilityName, Division, PRODUCT, LotTrackingCode, ProductName, ProductNo, RevisionControlFlag, SerialTrackingCode, UOM, UOMCode, ORDER_STATUS, OrderStatus, PARTNER, PartnerName
$SB(\text{PROCESS})$	PROCESS, ProcessName, ProcessDescription, Fuid
$SB(\text{PRODUCT})$	PRODUCT, LotTrackingCode, ProductName, ProductNo, RevisionControlFlag, SerialTrackingCode, UOM, UOMCode, FACILITY, FacilityName, Division
$SB(\text{UOM})$	UOM, UOMCode

SB(WIP_ORDER_STATUS)	WIP_ORDER_STATUS, WipOrderStatus
SB(FACILITY)	FACILITY, FacilityName, Division
SB(ORDER_STATUS)	ORDER_STATUS, OrderStatus

Table 4 - Semantic Blocks (SB) of Flexnet MES

The procedure presented in section 4.5 has been implemented in the MEGA Suite environment. MEGA Suite supports UML notations and allows building our own meta-model based on its ad-hoc MOF[2] meta-model. The meta-model presented on Figure 3 has been implemented in the MEGA Suite. In this implementation, the semantic block is conceptualised as a UML package and the lexical and non-lexical concepts are conceptualised as UML classes. The procedure presented in section 4.5 has been implemented taking advantage of MEGA programming facilities.

Figure 14 provides a model representing all the semantic blocks related to the Flexnet purchase order process and their inclusion relationships. Figure 15 provides the conceptual model associated to the semantic block SB(PRODUCT), and including all the mandatory concepts required to obtain the full semantics for the concept *PRODUCT*.

[2] OMG's MetaObject Facility: http://www.omg.org/mof/

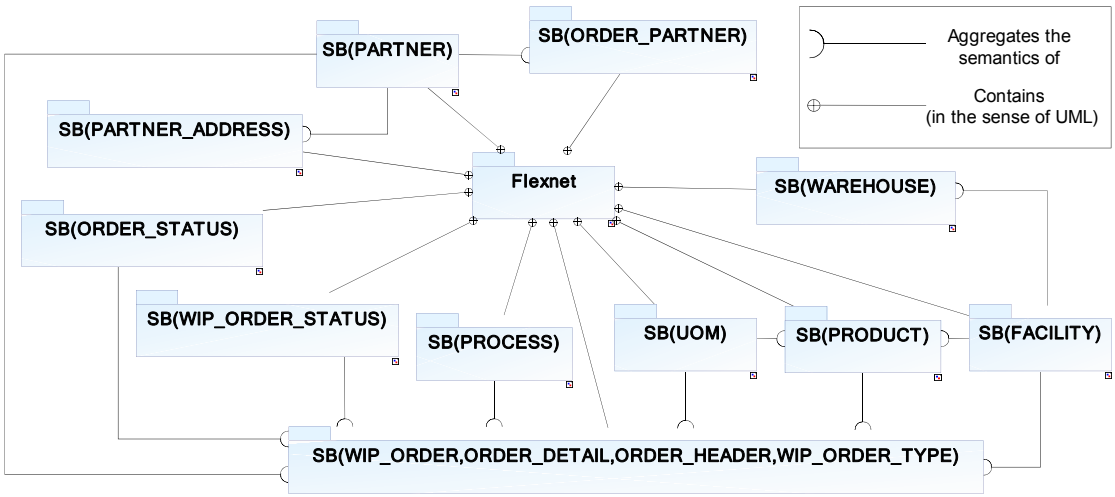


Figure 14 - The computed semantic blocks related to Flexnet MES

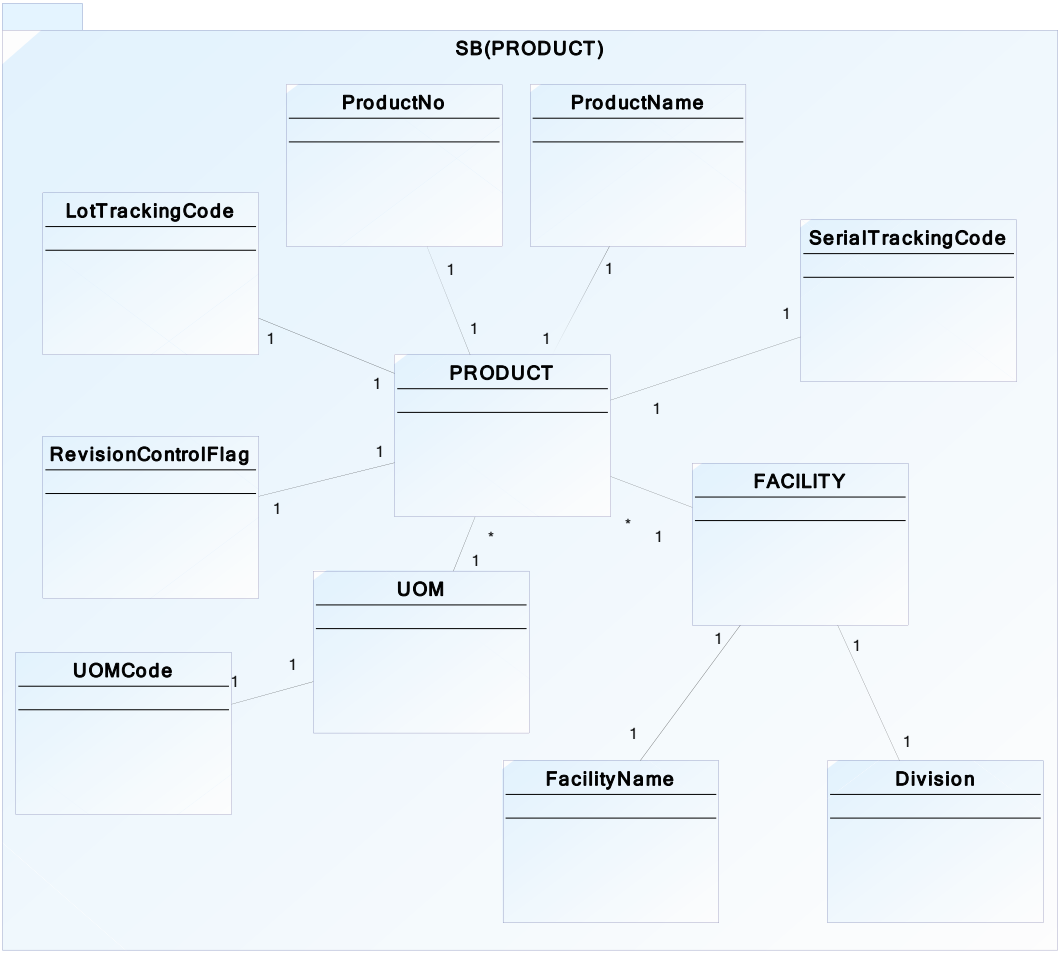


Figure 15 - The conceptual model associated to the semantic block *SB*(PRODUCT)

This section is derived from the following scientific publications:

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Semantics enactment in Enterprise Information Systems, IFAC. 18th IFAC World Congress, IFAC WC'2011, Aug 2011, Milan, Italy. Elsevier - IFACPapersOnline, 18, 13064-13073

Lezoche M., Panetto H., Aubry A., (2011). Formal Fact-Oriented model transformations for cooperative information systems semantic conceptualisation, Selected and extended version of ICEIS 2011. Lecture Notes in Business Information Processing, Proof read

6 Semantic Annotation Model Definition for Systems Interoperability

Nowadays, the need of systems collaboration across enterprises and through different domains has become more and more ubiquitous. But because the lack of standardized models or schemas, as well as semantic differences and inconsistencies problems, a series of research for data/model exchange, transformation, discovery and reuse are carried out in recent years. One of the main challenges in these researches is to overcome the gap among different data/model structures. Semantic annotation is not only just used for enriching the data/model's information, but also it can be one of the useful solutions for helping semi-automatic or even automatic systems interoperability.

This work is performed by Yongxin LIAO, a Ph.D. student at CRAN laboratory (financed by Fédération Charles Hermite and Region Lorraine) under the supervision of Pr. Hervé Panetto (CRAN) and Pr. Nacer Boudjlida (LORIA). I contributed to this work by helping Yongxin LIAO, during his first year, on the formalisation part of his work.

Our efforts focused on bridging the different knowledge representations through the use of semantic annotation. It can be widely used in many fields:

- It can be used to discover matching between models elements, which helps information systems integration (Agt, et al., 2010).
- It can semantically enhance XML-Schemas' information, which supports XML documents transformation (Köpke and Eder, 2010).
- It can describe web services in a semantic network, which is used for further discovery and composition (Talantikite, et al., 2009).
- It can support system modellers in reusing process models, detecting cross-process relations, facilitating change management and knowledge transfer (Bron, et al., 2007).
- It can link specific resources according to its domain ontologies.

The main contribution of our research work is identifying three main components of semantic annotation, gives a formal definition of semantic annotation and presenting a metamodel of the proposed semantic annotation structure model.

6.1 What is semantic annotation?

In Oxford Dictionary Online[3], the word "annotation" is defined as "*a note by way of explanation or comment added to a text or diagram*". It is used to enrich target object's information, which can be in the forms of text descriptions, underlines, highlights, images, links, etc. Annotation has special meanings and usages in different fields. In software programming, an annotation is represented as text comments embedded in the code to expand the program, which is being ignored when the program is running. In mechanical drawing, an annotation is a snippet of text or symbols with specific meanings. In Library Management, an annotation is written in a set form (numbers, letters, etc.), which helps the classification of books.

Further, different annotation types are identified by the following papers: Bechhofer, et al. (2002) and Boudjlida, et al. (2006) distinguished annotation as (i) *Textual annotation*: adding notes and comments to objects; (ii) *Link annotation*: linking objects to a readable content; (iii) *Semantic annotation*: that consists of semantic information which is machine-readable. Similarly, three types of annotation are described in the research of Oren, et al. (2006): (i) *Informal annotation*: notes that are not machine-readable; (ii) *Formal annotation*: notes that are formally defined and machine-readable (but it does not use ontology terms); (iii) *Ontological annotation*: notes that use only formally defined ontological terms that are commonly accepted and understood.

According to the above classification, semantic annotation can be considered as a kind of formal metadata, which is machine and human readable. This will be further discussed in the following sections.

[3]http://oxforddictionaries.com

6.1.1 Semantic annotation

The term "Semantic Annotation" is described as "*the action and results of describing (part of) an electronic knowledge by means of metadata whose meaning is formally specified in an ontology*" (electronic knowledge can be text contents, images, video, services, etc.) by Fernández (2010). Talantikite, et al. (2009) introduced it as "*An annotation assigns to an entity, which is in the text, a link to its semantic description. A semantic annotation is referent to an ontology*". In the research of Lin (2008), semantic annotation is concerned as "*an approach to link ontologies to the original information sources*". All above definitions from different papers show one thing in common: a semantic annotation is the process of linking electronic knowledge to a specific ontology. Ontology here is only one of the possible means to provide a formal semantic.

As it can be seen on Figure 16, the left side represents an Electronic Knowledge (EK) and on the right side, there are the three main components of semantic annotation: (1) *Ontology*, which defines the terms used to describe and represent a body of knowledge (Boyce, et al., 2007). It can be reused from existing ontologies or designed according to different requirements. (2) *Semantic Annotation Structure Model (SASM)*, which organizes the structure/schema of an annotation and describes the mappings between electronic knowledges and an ontology. (3) *Application*, which is designed to achieve the user's purposes (composition, sharing and reuse, integration, etc.) by using SASM. This figure also shows the three main steps on how to use semantic annotation, which is introduced in section 4: ontology (section 4.1), semantic annotation structure model (section 4.2) and application (section 4.3).

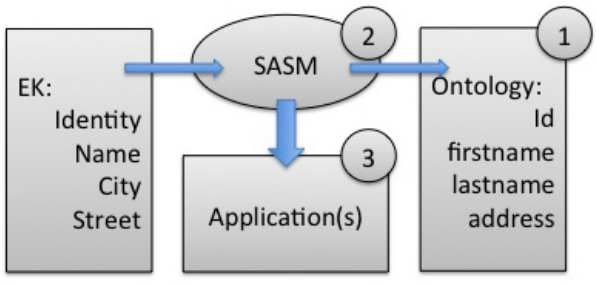


Figure 16 – Semantic annotation components

The following definition formally defines a semantic annotation: a *S*emantic Annotation $SA$is a tuple $(\mathcal{M}, \mathcal{A})$ consisting of the SASM $\mathcal{M}$and an application $\mathcal{A}$.

$$SA := \{\mathcal{M}(\mathcal{E}, \mathcal{P}(\mathcal{O})), \mathcal{A}\}$$

Where:

$\mathcal{O} = \{o_1, o_2, \ldots, o_n\}$, is the set of ontology o_i that bring some meaning to any annotated element.

An *Ontology* $o_j \epsilon\, \mathcal{O}$ is a 4-tuple (C_{o_j}, is_a, R_{o_j}, σ_{o_j}), where C_{o_j} is a set of *concepts*, *is_a* is a partial order relation on C_{o_j}, $R_{o_j}$is a set of relation names, andσ_{o_j}: $R_{o_j} \rightarrow (C^+)$is a function which defines each relation name with its arity (Stumme and Maedche, 2001a).

Formally, $\mathcal{M} = \{m_x: \langle e_i, p_j \rangle |\ e_i \epsilon\, \mathcal{E} \times p_j \epsilon\, \mathcal{P}(\mathcal{O})\}$ and represents the set of relationships between an element e_i of the set of electronic knowledge $\mathcal{E}$ and an element p_j of the powerset of the ontology set $\mathcal{O}$.

A mapping $m_x(e_i, p_j)$ may represent three different kinds of semantic relations:

(1) $m_{\sim}(e_i, p_j)$ is a binary *equivalence* relation. If $m_{\sim}(e_i, p_j)$ then an electronic knowledge e_i is semantically equivalent to p_j, an element of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

(2) $m_{\supset}(e_i, p_j)$ is a binary relation stating that the semantic of an electronic knowledge e_i *subsumes* the semantic of an element p_j of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

(3) $m_{\subset}(e_i, p_j)$: is a binary relation stating that the semantic of an electronic knowledge e_i *is subsumed* by the semantic of an element p_j of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

$\mathcal{M}$ can be further extended, including also some additional parameters or constraints c_k, generally expressed using, in the worst case, natural language, or, better, a formal logical expression. $\mathcal{M}$ is then defined as $\mathcal{M} := \{m_x, c_k\}$.

The main issue, related to mappings such as in (2) and in (3), is being able to measure the semantic gap (2) or the over semantic (3), brought by the semantic annotation. Such measures have been studied by researchers in the domain of information retrieval (Ellis, 1996) or in the domain of ontology matching (Maedche and Staab, 2002), mapping (Doan et al, 2002), merging (Stumme and Maedche, 2001b), alignment (Noy and Musen, 2000).

In addition, Peng, et al. (2004) also gave a very simple definition of semantic annotation in their paper, which is $SA := (R, O)$, where R is set of resources and O is an ontology. Furthermore, Luong and Dieng-Kuntz (2007) defined it as $SA := \{R_A, C_A, P_A, L, T_A\}$. In this definition, R_A is a set of resources; C_A is a set of concept names; P_A is a set of property names; L is a set of literal values; and T_A is a set of triple (s, p, v), where $s \in R_A, p \in P_A, v \in (R_A \cup L)$. To the best of our knowledge, T_A in this definition is duplicated.

6.2 Metamodel of Semantic Annotation Structure Model

After the identification of the three main components of semantic annotation and given its formal definition, in this section we can briefly present the metamodel of the Semantic Annotation Structure Model (Figure 17).

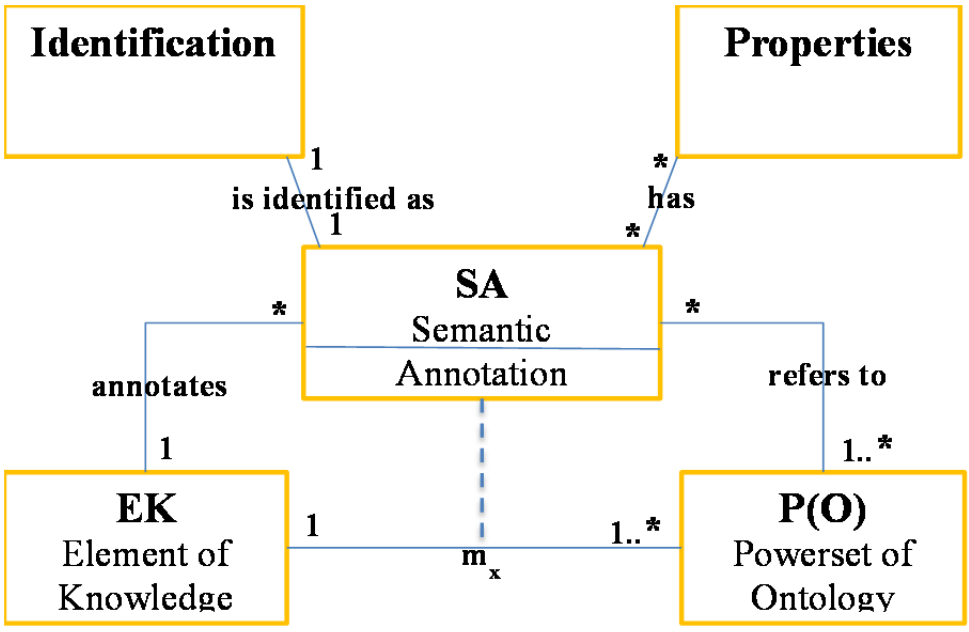


Figure 17 – Semantic annotation structure model metamodel

The metamodel presents five entities that represent the core of the structure. The Identification entity highlights the uniqueness of a semantic annotation, the Properties entity represents the constraints existing for a given semantic annotation, the Element of Knowledge entity shows the reference to a resource that has to be annotated and the powerset of Ontology entity refers to the relation between the semantic annotation and one or more Ontologies. The relation between the Powerset of Ontology and EK represents our studied relationship $\mathcal{M}$ that represents the set of relationships between the set of electronic knowledge and the powerset of the ontology set.

6.3 Conclusions

In this section we identify three main components of semantic annotations that are Ontology, Semantic Annotation Structure Model and Application. In addition, a formal definition of semantic annotation is proposed. It contributes to better understand what a semantic annotation is and then contributes to a common reference model. We presented a Metamodel explaining the formal links between the identified semantic annotation components.

But how to use semantic annotation? There are still many problems can be further discussed during the annotation process. For example, how to optimize ontology and an annotated model? How to solve the inconsistency or conflicts during the mapping? How to add consistent semantic on models in different levels of a system? How to achieve semi-automatic or automatic annotation?

We are currently investigating how semantic annotations can help collaborative actors (organizations, design teams, system developers, etc.) in co-designing, sharing, exchanging, aligning and transforming models. In particular, this research work will be based on general systems with several kinds of interactions. We can have interoperation between systems that with different versions (during many years, systems may have been modified or updated). We can also have systems with same functions but used by different enterprises. Semantic annotations can bridge this knowledge gap and identify differences in models, in schemas, etc. In some case, interoperation is a process between a set of related systems throughout a product lifecycle (Marketing, Design, Manufacture, Service, etc.), and semantic annotations

can influence the existing foundations and techniques which supports models reuse, semantic alignment and transformation, etc. Above all, our research work will focus on designing, and reusing appropriate ontologies in relationship with a formal semantic annotation structure model.

This section is derived from the following scientific publications:

Liao Y., Lezoche M., Panetto H., Boudjlida N., (2011). Why, Where and How to use Semantic Annotation for Systems Interoperability, 1st UNITE Doctoral Symposium, Jun 2011, Bucarest, Romania. pp. 71-78

Liao Y., Lezoche M., Panetto H., Boudjlida N., (2011). Semantic Annotation Model Definition for Systems Interoperability, OTM. OTM 2011 Workshops 2011 - 6th International Workshop on Enterprise Integration, Interoperability and Networking (EI2N), Oct 2011, Hersonissos, Crete, Greece. Springer-Verlag, LNCS 7046, pp. 61-70, Lecture Notes in Computer Science

7 Conclusions

In this research manuscript, we proposed a conceptualisation approach for enacting implicit semantics embedded into Enterprise Information System models by a deep analysis of existing data models enriched by users' and experts' knowledge. This approach is composed of 3 steps, staging from the traditional database reverse engineering process, through a knowledge elicitation and model enrichment by domain experts, to the application of formal fact-oriented modelling rules for externalising tacit semantics. Moreover, in order to structure the whole semantics into independent aggregates that may emphasize subsystems, we defined the concept of semantic block (SB) and we developed an automatized procedure to compute these SBs. The resulting semantics architecture allows the identification of semantically self-contained subsystems, facilitating further interoperation analysis.

The conceptualisation and structuring processes have been validated on a case study involving two industrial Enterprise Applications demonstrating the applicability of our approach.

Further works concern using the resulting semantic conceptual model and architecture for facilitating the assessment of the (non)-interoperation barriers between Enterprise Information Systems or some of their subsystems (identified, for instance, by the semantic blocks) as suggested in (Yahia, 2011). The resulting analysis, based on an interoperability measures map, can help information technology consulting companies for parameterising and integrating enterprise applications (ERP, MES…) taking into account interoperability constraints.

7.1 Scientific Contribution

This manuscript is the scientific product of our study on the applied semantic interoperability domain. During my Post-Doc period, I produced, with the laboratory team, the following scientific communications.

Peer-reviewed international journals (1 extended selected paper, 1 Under Review)

Lezoche M., Panetto H., Aubry A., (2011). Formal Fact-Oriented model transformations for cooperative information systems semantic conceptualisation, Lecture Notes in Business Information Processing (LNBIP), Selected extended version from ICEIS'2011, Proof read

Lezoche M., Panetto H., Aubry A., (2011). Conceptualising and structuring semantics in Cooperative Enterprise Information Systems Models, Enterprise Information System (EIS), Taylor & Francis edition, 2011. Under Review.

Peer-reviewed national journal (1 Under Review)

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Extraction de la sémantique dans les modèles de systèmes d'information d'entreprises collaboratives, Ingénierie des Systèmes d'Information, Hermès. Under Review

Peer-reviewed conference/proceedings (5)

Lezoche M., Panetto H., Aubry A., (2011). Conceptualisation approach for cooperative information systems interoperability, ACM. 13th International Conference on Enterprise Information Systems, ICEIS 2011, Jun 2011, Beijing, China. pp. 101-110

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Semantics enactment in Enterprise Information Systems, IFAC. 18th IFAC World Congress, IFAC WC'2011, Aug 2011, Milan, Italy. Elsevier - IFACPapersOnline, 18, 13064-13073, IFACPapersOnline

Liao Y., Lezoche M., Panetto H., Boudjlida N., (2011). Why, Where and How to use Semantic Annotation for Systems Interoperability, 1st UNITE Doctoral Symposium, Jun 2011, Bucarest, Romania. pp. 71-78

Zdravković M., Trajanović M., Panetto H., Aubry A., Lezoche M., (2011). Ontology-based supply chain process configuration, University of Nis. 34th International Conference on Production Engineering, ICPE 2011, Sep 2011, Nis, Serbia. pp. 399-402

Liao Y., Lezoche M., Panetto H., Boudjlida N., (2011). Semantic Annotation Model Definition for Systems Interoperability, OTM. OTM 2011 Workshops 2011 - 6th International Workshop on Enterprise Integration, Interoperability and Networking (EI2N), Oct 2011, Hersonissos, Crete, Greece. Springer-Verlag, LNCS 7046, pp. 61-70, Lecture Notes in Computer Science

Présentation without peer reviewing (1)

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Extraction de la sémantique dans les modèles de systèmes d'information d'entreprises collaboratifs, Journée Nationale du GT Easy-DIM, Apr 2011, Lyon, France.

8 References

Agt, H., Bauhoff, G., Kutsche, R., Milanovic, N., Widiker, J. 2010. Semantic Annotation and Conflict Analysis for Information System Integration. In: Proceedings of the 3rd Workshop on Model-Driven Tool & Process Integration. 7-18

Aspray, W. F. 1985. The Scientific Conceptualization of Information: A Survey. Annals of the History of Computing, 7: 117-40. IEEE

Astrova, I., 2004. Reverse Engineering of Relational Databases to Ontologies. In *Lecture Notes in Computer Science*, LNCS 3053, pp. 327-341, Springer.

Badia, A., 2002. Conceptual Modeling for Semistructured Data. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering Workshops* (WISE 2002 Workshops), p. 170-177. Singapore.

Barker, R. 1990, CASE* Method: Entity Relationship Modelling, Addison-Wesley, Wokingham, England.

Bechhofer, S., Carr, L., Goble, C., Kampa, S. and Miles-Board, T. 2002. The Semantics of Semantic Annotation. In: Proceedings of the 1st International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems. 1151-1167.

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

Bézivin, J., Kurtev, I. 2005. Model-based Technology Integration with the Technical Space Concept. Proceedings of the Metainformatics Symposium, Esbjerg, Denmark, November 8-11, 2005. Springer-Verlag

Bidgol, H., 2004. *The Internet Encyclopedia*, Volume 1, John Wiley & Sons, Inc. p. 707.

Born, M., Dorr, F., Weber, I. 2007. User-Friendly Semantic Annotation in Business Process Modeling. In: Proceedings of the 2007 international conference on Web information systems engineering.

Boudjlida, N., Dong, C., Baïna, S., Panetto, H., Krogstie, J., Hahn, A., Hausmann, K., Tomás, J.V., Poler, R., Abián, M.Á., Núñez, M.J., Zouggar, N., Diamantini, C., Tinella, S. 2006. A practical experiment on semantic enrichment of enterprise models in a homogeneous environment. INTEROP NoE Deliverable DTG4.1. INTEROP NoE IST 508011. http://www.interop-vlab.eu

Boudjlida, N., Panetto, H. 2007. Enterprise semantic modelling for interoperability. In: Proceedings of the 12th IEEE conference on emerging technologies and factory automation, Patras, Greece. 847–854.

Boyce, S., Pahl, C. 2007. Developing Domain Ontologies for Course Content. Educational Technology & Society 275-288.

Bunge, M. 1977. Treatise on Basic Philosophy (Vol 3): Ontology I: The Furniture of the World. D. Reidel Publishing Company, first edition

Carney, D., Fisher, D., Morris, E., Place P., 2005. Some current Approaches to Interoperability. In *technical note* CMU/SEI-2005-TN-033.

Chen D., Dassisti M., Elvesaeter B., Panetto H., Daclin N., Jaekel F.-W., Knothe T., Solberg A., Anaya V., Sanchis Gisbert R., Kalampoukas K., Pantelopoulos S., Bertoni M., Bordegoni M., Cugini U., Pulli M., Perjons E., Assogna P. 2006. In *DI.2: Enterprise Interoperability Framework and knowledge corpus*, Interoperability Research for Networked Enterprises Applications and Software Network of Excellence, n° IST 508-011.

Chen, R. S., Nadkarni, P., Marenco, L., Levin, F., Erdos, J., Miller, P.,. 2000. Exploring Performance Issues for a Clinical Database Organized Using an Entity-Attribute-Value Representation. J Am Med Inform Assoc, 475-487

Chiang Roger, H. L., Barron, T. M., Storey Veda, C., 1994. Reverse engineering of relational databases: Extraction of an EER model from a relational database. In *Data & Knowledge Engineering*. Volume 12, Number 2, pp. 107-142.

Clarke E., Wing J., 1996. Formal methods: state of the art and future directions. In ACM Computing Surveys (CSUR) - Special ACM 50th-anniversary issue: strategic directions in computing research. Volume 28, Number 4, pp. 626-643.

Czejdo, B., Elmasri, R., Rusinkiewicz, M. & Embley, D.W. 1990, 'A graphical data manipulation language for an extended entity-relationship model', IEEE Computer, March 1990, pp. 26-37.

De Bo, J., Spyns, P., and Meersman, R., 2003. Creating a "DOGMAtic" multilingual ontology infrastructure to support a semantic portal. In proceedings of the OTM confederated international workshops HCI-SWWA, IPW, JTRES, WORM, WMS, and MRSM. Catania , Italie: Springer-Verlag New York Inc, pp. 253-266.

Diamantini, C., Potena, D. 2008. Semantic enrichment of strategic datacubes. In: Proceedings of the ACM 11th International Workshop on Data Warehousing and OLAP. 81-88.

Ding, G., Xu, N. 2010. Automatic semantic annotation of images based on web data. In: Proceedings of the 6th international conference of Information Assurance and Security. 317-322

Doan, A., Madhavan, J., Domingos, P., Halevy, A.Y. 2002. Learning to map between ontologies on the semantic web. In Proceedings of the World Wide Web conference. 662-673.

Ellis, D. 1996. The Dilemma of Measurement in Information Retrieval Research. Journal of the American Society for Information. Vol. 47, N° 1. 23-36.

Engelbart, D.C., 1962. Augmenting human intellect: a conceptual framework. In *Menlo Park, CA*: Stanford Research Institute.

Euzenat, J., 2001. Towards a principled approach to semantic interoperability. In *CEUR Proceedings of the IJCAI-01* Workshop on Ontologies and Information Sharing, Seattle, USA, August 4-5, , ISSN 1613-0073, Vol. 47., 19-25.

Fernández, N. 2010. Semantic Annotation Introduction, [online] (Updated 14 Oct 2010) Available at <http://www.it.uc3m.es/labgimi/teoria/Module2/SA-Intro.pdf>

Fikes, R., Farquhar, A., Rice, J. 1997. Tools for Assembling Modular Ontologies in Ontolingua. In: Proceedings of the 14th national conference on artificial intelligence and 9th conference on Innovative applications of artificial intelligence. 436-441.

Fonkam, M.M., Gray, W.A., 1992. An Approach to Eliciting the Semantics of Relational Databases. In *CAiSE 1992*, Manchester, UK, May 12-15, 1992. Lecture Notes in Computer Science 593 Springer, ISBN 3-540-55481-5. 463-480 Manchester, UK.

Frankel D. S. 2003. Model Driven Architecture: Applying MDA to Enterprise Computing. John Wiley & Sons.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J., 1995. Design patterns: elements of reusable object-oriented software. Vol. 206. Addison-wesley Reading, MA.

Genesereth, M.R., Nilsson, N.J., 1987. Logical Foundation of Artificial Intelligence, Morgan Kauffman, Los Altos, CA

Guarino, N., 1998. *Formal Ontology in Information Systems* (Ed.). IOS Press.

Guarino, N. 1997. Understanding, building and using ontologies, International Journal of Human-Computer Studies 46 (2–3) 293–310.

Guo, J., 2009. Collaborative conceptualisation: towards a conceptual foundation of interoperable electronic product catalogue system design. Enterprise Information Systems, 3 (1) 59-94.

Halpin, T.A. 1991, 'A fact-oriented approach to schema transformation', Proceedings of MFDBS-91, Springer Lecture Notes in Computer Science, LNCS 495. Springer

Halpin, T., 1998. *Handbook on Architectures of Information Systems* Chapter 4, eds P. Bernus, K. Mertins & G. Schmidt, Springer-Verlag, Berlin.

Henrard, L., Hainaut, J.-L., 2001. Data Dependency Elicitation in Database Reverse Engineering Software Maintenance and Reengineering. In *Fifth European Conference on Software Maintenance and Reengineering*, pp. 11

Hohenstein, U. & Engels, G. 1991, 'Formal semantics of an entity-relationship-based query language', Entity-Relationship Approach: the core of conceptual modelling (Proc. 9th ER conf.), ed. H. Kangassalo, Elsevier Science Pub., Amsterdam

Horrocks, I., Patel-Schneider, P.F., Harmelen, F.V. 2003. From SHIQ and RDF to OWL: the making of a Web Ontology Language. Journal of Web Semantics. Vol 1, N° 1. 7-26.

IEEE: Standard Computer Dictionary, 1990. A Compilation of IEEE Standard Computer Glossaries. In *NY. 610-1990*. ISBN: 1559370793.

International Organization for Standardization, 1999. ISO 14528: Industrial Automation Systems – Concepts and rules for Enterprise Models, *TC 184/SC5/WG1*, Geneva, Switzerland.

International Organization for Standardization, 2002. ISO 16100: Manufacturing Software Capability Profiling for interoperability. In *Part 1: Framework, TC 184/SC5/WG4*, Geneva, Switzerland.

Irfanullah, I., Aslam, N., Loo, J., Loomes, M., Roohullah, R. 2010. In: Proceedings of the IEEE international symposium on Signal Processing and Information Technology. 491-495

Izza, S., 2009. Integration of industrial information systems: from syntactic to semantic integration approaches. *Enterprise Information Systems*. 3(1), pp. 1-57, Taylor & Francis.

LaOngsri, S. 2009, Semantic Extensions and a Novel Approach to Conceptual Modelling, Ph.D. Thesis, School of Computer Science, Engineering and Mathematics, The Flinders University of South Australia

Lezoche M., Panetto H., Aubry A. 2011. Conceptualisation approach for cooperative information systems interoperability, ACM. 13th International Conference on Enterprise Information Systems, ICEIS 2011, Jun 2011, Beijing, China. pp. 101-110

Lin, Y. 2008. Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability. PhD thesis, Norwegian University of Science and Technology, Trondheim, Norway.

Liao, Y., Romain, D., J. Berre, A. 2010. Model-driven Rule-based Mediation in XML Data Exchange. In: Proceedings of the 1st International Workshop on Model-Driven Interoperability. 89-97

Luong, P., Dieng-Kuntz, R. 2007. A Rule-based Approach for Semantic Annotation Evolution. In Computational Intelligence. Vol. 23, Issue 3, 320–338

Kifer, M., Lausen, G., Wu, J. 1995. Logical Foundations of Object-Oriented and Frame-Based Languages. Journal of the ACM. Vol.42, N°4. 741-843.

Kopecký, J., Vitvar, T., Bournez, C., Farrell, J. 2007. SAWSDL: Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing. Vol.11, N° 6. 60-67

Köpke, J., Eder, J. 2010. Semantic Annotation of XML-Schema for Document Transformations. In: Proceedings of the OTM Workshops. 5th International Workshop on Enterprise Integration, Interoperability and Networking. Lecture Notes in Computer Science, LNCS 6428. 219-228.

Maedche, A., Staab, S. 2002. Measuring Similarity between Ontologies. In: Proceeding of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web. 251-263.

Mellor, S.J., Scott, K., Uhl, A., Weise, D. 2002. Model-Driven Architecture. In: Proceedings of the Workshop at the 8th International Conference on Object-Oriented Information Systems. 290-297.

Mellor S.J., Kendall S., Uhl A. and Weise D. 2004. Model Driven Architecture, Addison-Wesley Pub Co.

Martin, D., Paolucci M., Wagner, M. 2007. Towards Semantic Annotations of Web Services: OWL-S from the SAWSDL Perspective. In: Proceedings of the OWL-S Experiences and Future Developments Workshop at ESWC 2007.

Mizoguchi, R. 2003. Tutorial on Ontological Engineering: Part 2: Ontology Development, Tools and Languages. New Generation Comput.

Mani, M.: EReX, 2004. A Conceptual Model for XML. In *Proceedings of the Second International XML Database Symposium* (XSym 2004), p. 128-142. Toronto, Canada.

Manola, F., Miller, E., 2004. RDF Primer. In *World Wide Web Consortium*, Recommendation REC-rdf-primer-20040210.

Melville, N., Kraemer, K., Gurbaxani, V., 2004. Information Technology and Organizational Performance: an Integrative Model of IT Business Value. In *MIS Quarterly*, Volume 28 Number 2, pp. 283-322.

Nijssen, G.M. & Halpin, T.A., 1989. Conceptual Schema and Relational Database Design, Prentice Hall, Sydney.

Noy, N.F., Musen, M.A. 2000. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: Proceedings of the 17th National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence. 450-455.

Obrst, L., 2003 Ontologies for semantically interoperable systems. In *Proceedings of the 12th International Conference on Information and Knowledge Management*. New Orleans, USA

OMG, 2003. Object Management Group. Architecture-Driven Modernization specification http://adm.omg.org

OMG, 2004. Object Management Group. UML 2.0 Superstructure Specification http://uml.omg.org

Oren, E., Hinnerk Möller, K., Scerri, S., Handschuh, S., Sintek, M. 2006. What are Semantic Annotations?. Technical report, DERI Galway

Patil, A., Oundhakar, S., Sheth, A., Verma, K. (2004). Meteor-S Web Service annotation framework. In: Proceedings of the 13th International Conference on the World Wide Web. 553-562.

Peng, W., Baowen, X., Jianjiang, L., Dazhou, K., Yanhui, L. 2004. A Novel Approach to Semantic Annotation Based on Multi-ontologies. In: Proceedings of the third International Conference on Machine Learning and Cybernetics. Vol. 3. 1452 - 1457

Reeve, L.H., Han, H. 2005. Survey of semantic annotation platforms. In: Proceedings of the ACM Symposium on Applied Computing. 1634-1638

Russel, S., Norvig, P. *Artificial Intelligence, A Modern Approach"*, Prentice-Hall. Inc. 1995

Seeley, R. S., 1997. Manufacturing execution systems in MED DEVICE DIAGN IND. Vol. 19, no. 11, pp. 64-68.

Sharir, M., 1981. A strong-connectivity algorithm and its applications in data flow analysis. In *Computers and Mathematics with Applications*. Volume 7, pp. 67-72.

Sheth, A., 1998. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. In M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman, editors. In *Interoperating Geographic Information Systems*, pp. 5– 30. Kluwer.

Smith, M.K., Welty, Ch., McGuinness, D.L., 2004. OWL Web Ontology Language Guide. In *World Wide Web Consortium*, Recommendation REC-owl-guide-20040210.

Störrle H., 2005. Semantics and Verification of Data Flow in UML 2.0 Activities. In Electronic Notes in Theoretical Computer Science, volume 276, pp. 35-52.

Stumme, G., Maedche, A. 2001a. Ontology Merging for Federated Ontologies on the Semantic Web. In: Proceedings of the International Workshop for Foundations of Models for Information Integration.

Stumme, G., Maedche, A. 2001b. FCA-MERGE: Bottom-Up Merging of Ontologies. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence. Seattle, Washington, USA. 225-234.

Talantikite, H.N., Aïssani, D., Boudjlida, N. (2009). Semantic annotations for web services discovery and composition. Computer Standards & Interfaces. Vol. 31, N°6. 1108-1117.

Tolk, A., Diallo, S. Y., Turnitsa, C. D., 2007. Applying the Levels of Conceptual Interoperability Modelling Support of Integrability, Interoperability, and Composability for System-of-Systems Engineering. In *Journal of Systemics, Cybernetics and Informatics*, Volume 5 Number 5, pp. 65-74.

Tursi, A., Panetto, H., Morel, G., Dassisti, M., 2009 Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. In *IFAC Annual Reviews in Control*. 33/2, 238-245, Elsevier, ISSN: 1367-5788.

Vyvyan, E., 2006. Lexical Concepts, Cognitive Models and Meaning-Construction. In *Cognitive Linguistics* 17 (4): 491-534.

Wand, Y., Weber, R. 1993. On the ontological expressiveness of information systems analysis and design grammars. Information System Journal Vol 3. N°4. 217-237.

Yahia E., Lezoche M., Aubry A., Panetto H. 2011. Semantics enactment in Enterprise Information Systems, IFAC. 18th IFAC World Congress, IFAC WC'2011, Aug 2011, Milan, Italy. Elsevier - IFACPapersOnline, 18, 13064-13073, IFACPapersOnline

Yahia, E., 2011. Contribution à l'Evaluation de l'Interopérabilité Sémantique entre Systèmes d'Information d'Entreprises : Application aux Systèmes d'Information de Pilotage de la Production. PhD Thesis, Université Henri Poincaré, Nancy I (in French).

Zdravković. M., Panetto, H., Trajanović, M., Aubry, A. 2011. An approach for formalising the supply chain operations, Enterprise Information Systems, 5/4, 401-421, Taylor & Francis, ISSN 1751-7575. DOI:10.1080/17517575.2011.593104.

Appendix

In the appendix will be presented a selection of the published papers.

Appendix A:

Lezoche M., Panetto H., Aubry A., (2011). Conceptualisation approach for cooperative information systems interoperability, ACM. 13th International Conference on Enterprise Information Systems, ICEIS 2011, Jun 2011, Beijing, China. pp. 101-110

Appendix B:

Lezoche M., Panetto H., Aubry A., (2011). Formal Fact-Oriented model transformations for cooperative information systems semantic conceptualisation, Selected extended version of ICEIS 2011. Lecture Notes in Business Information Processing (LNBIP), Proof read

Appendix C:

Yahia E., Lezoche M., Aubry A., Panetto H., (2011). Semantics enactment in Enterprise Information Systems, IFAC. 18th IFAC World Congress, IFAC WC'2011, Aug 2011, Milan, Italy. Elsevier - IFACPapersOnline, 18, 13064-13073, IFACPapersOnline

Appendix D:

Liao Y., Lezoche M., Panetto H., Boudjlida N., (2011). Why, Where and How to use Semantic Annotation for Systems Interoperability, 1st UNITE Doctoral Symposium, Jun 2011, Bucarest, Romania. pp. 71-78

9 Appendix A

Conceptualisation approach for Cooperative Information Systems interoperability

Mario Lezoche, Hervé Panetto, Alexis Aubry
Research Centre for Automatic Control (CRAN), Nancy-University, CNRS, Campus
Scientifique, Faculté des Sciences et Technologies, BP 70239, 54506 Vandoeuvre-lès-Nancy Cedex, France
{mario.lezoche, herve.panetto, alexis.aubry}@cran.uhp-nancy.fr

Keywords: **Conceptual modelling, cooperative information systems, semantic interoperability, data model conceptualisation**

Abstract: **In order to increase enterprise performance, economics paradigms focus, now more than ever, on how to better manage information. The modern architecture of information systems is based on distributed networks with a grand challenge representing and sharing knowledge managed by those ISs. One of the main issues in making such heterogeneous Cooperative Information Systems (CIS) working together is to remove semantics interoperability barriers. This paper firstly analyses interoperability issues between CISs and then proposes patterns for data models conceptualisation for knowledge explicitation, based on expert knowledge injection rules and a fact-oriented approach. A case study is proposed related to a work order process in Sage X3, an Enterprise Resource Planning application.**

1 Introduction

The actual archetype for the Information Systems (ISs) involves large number of ISs distributed over large, complex computer/ communication networks. Such cooperative information systems (CIS) have access to large amount of information and have to interoperate to achieve their purpose. The cooperative information systems architects and developers have to face a hard problem: interoperability.

Interoperability can be defined as the ability for two or more systems to share, to understand and to consume information (IEEE, 1990). Some work (Chen et al., 2006) in the INTEROP NoE project has identified three different levels of barriers for interoperability: technical, conceptual and organisational. Organisational barriers are still an important issue but out of scope of this paper. The technological barriers are strongly studied by researchers in computer science and are generally based on models transformation (Frankel, 2003).

Our research focuses on the conceptual level of interoperability that is the ability to understand the exchanged information. A concept is a cognition unit of meaning (Vyvyan, 2006), an abstract idea, a mental symbol. It is created through the action of conceptualisation, that is, a general and abstract mental representation of an object. During the history of human effort to model knowledge, different conceptualisation

approaches regarding different application domains were developed (Aspray, 1985).
This paper is dealing with a first step from a more general work focusing on the study of the semantic loss during the exchange of information representing business concepts. Quantifying the semantic gap between interoperating ISs implies enacting their semantics through their normalized conceptual models. Indeed, in this context, the starting point for semantics interoperability is related to models conceptualisation.

We will present a conceptualisation approach to make explicit the finest-grained semantics embedded into conceptual models for finally enabling two different information systems seamlessly interoperating.

Next section presents the general context of our work. Then, the following section details the fundamental pillars of our conceptualisation process. Then, we will propose a knowledge explicitation process starting from an implemented relational model to a fact-oriented conceptual one. This process allows us emphasizing the finest-grained semantics that must be enacted to study semantics interoperability between collaborating ISs.

Finally, to validate our proposal, a practical case study is presented based on an Enterprise Resource Planning application involved in a B2M (Business to Manufacturing) interoperation process.

2 Cooperative Information systems

Information Systems are systems whose activities are devoted to capture and to store data, to process them and produce knowledge, used by any stakeholders within an enterprise or among different networked enterprises. It is commonly agreed that Cooperative Information Systems provide a backbone for the Integrated Information Infrastructure (Sheth, 1998). Fully understanding and exploiting the advances in computing is the only way to encompass the complexity of constructing and maintaining such systems.

Although the progress made in information technology considerably improved the efficiency of applications development, its drawbacks and limitations are obvious and serious. In fact, the application models involved in a single application are numerous and different, each coping only with particular and partial aspects of the overall task. Moreover, the components technologies are heterogeneous, platform- and machine-dependant. The above-mentioned limitations and barriers measurably hinder the development and the maintenance process.

There is a growing demand to integrate such systems tightly with organizational work so that these information systems can be directly and immediately used by the business activity.
Here, the need of interoperation clearly appears. In fact, to achieve the purpose of the cooperation between the different Information Systems, information must be physically exchanged (technical interoperability), must be understood (conceptual interoperability) and must be used for the purpose that they have been produced (conceptual and organisational interoperability). When trying to assess the understanding of an expression coming from a system to another system, 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).

This tiered structure is arguable in general; it is not as strict as it seems. It makes sense because each level cannot be achieved if the previous levels have not been completed (Euzenat, 2001).

The encoding, lexical and syntactic levels are the most effective solutions for removing technical barriers for interoperability, but not sufficient, to achieve a practical interoperability between computerised systems. Dealing with trying to enable a seamless data and model exchange at the semantic level is still a big issue that needs conceptual representation of the intended exchanged information and the definition of the pragmatic meaning of that exchanged information in the context of the source and destination applications.

Different cooperation types have been investigated in ISO 14528 (ISO, 1999). In fact, this standard considers that models could be related in three ways:

(1) integration when there exists a standard or pivotal format to represent these models;
(2) unification when there exists a common meta-level structure establishing semantic equivalence between these models; and
(3) federation when each model exists per se, but 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 (Panetto, 2007). Classifying interoperability problems (Tursi, et al. 2009) may help in understanding the degree of development needed to solve, at least partially, these problems but conceptualisation and semantics extraction is still an important issue because of the various contextual understanding of tacit knowledge embedded into those applications. The main prerequisite for achievement of interoperability of information systems is to maximise the amount of semantics which can be used and make it increasingly explicit (Obrst, 2003), and consequently, to make the systems semantically interoperable. To highlight this issue, the paper is based on a referenced scenario involving enterprise systems applications.

Most of reverse engineering approaches (Fonkam, 1992) (Chiang, 1994) return the information structure but present a model with tacit semantics. The ADM (Architecture-Driven Modernization) initiative (OMG, 2003) from OMG (Bézivin et al., 2005) is tackling this problem by promoting a common Knowledge Discovery Meta-model to facilitate discovering tacit knowledge embedded inside existing software. In our scenario, those applications are still implemented and running using databases. We can extract, from them, by using reverse engineering approaches, some knowledge in a form of a conceptual model. We have then to enrich that model with enterprise applications best practices (knowledge coming from users). Finally, we make explicit all disclosed knowledge hidden in the resulting model.

3 Our approach for Semantics Enactment In Conceptual Models

In order to cooperate, two (or more) Information Systems have to interoperate. As previously discussed, we focus our interest on the conceptual level of interoperability letting different information systems to share and use knowledge models that they represent. Our principal issues are, therefore, first to understand the conceptual relationships between those models in the context of their use and secondly how, through conceptualisation, to unhide the tacit knowledge buried inside them. A usual approach for making explicit the tacit knowledge, concealed in attributes and classes, is the relationships-oriented perspective composed of a set of transformation rules. In that transformation method, an attribute a1 of type T1 pertaining to class C1 is modelled as a relationship between the class C1 and a standard type T1. This approach does not resolve entirely the semantics elicitation problem because it focuses its point of view on the values instead of on the concepts. The attribute semantics is somewhat yet hidden in the relationship just created.

In literature, (Meersman, 2003) presented the definition of two different objects types, *a lexical object (LOT),* a term*, is an object in a certain reality that can be written down. LOTs always consist of letters, numbers, symbols or other characters. They can be used as names for or references to other objects. A non-lexical object (NOLOT),* a concept, *is an object in a certain reality that cannot be written down. Non-lexical objects must be named by lexical objects or referred to by means of lexical objects*.

Applying these definitions, we can flatten the nested knowledge embedded in a model to simplify semantic enactment resulting from a set of modelling transformations. Our contribution is to have at our disposal an approach letting us to fragment knowledge through the transformation of attributes into entities and relationships, and thus to emphasize some fine-grained knowledge atoms. In the proposed approach, that is the first part (Figure 1) of our general methodology, the starting point can be various: an application, a data model, a logical view, a model. We have already mentioned that there are several reverse engineering methods, such as in (Fonkam, 1992) and in (Chiang, 1994), through which a model from the application or schema level can be derived (Step 1). Then, the resulted initial model is enriched and corrected through an Expert Knowledge Injection step (Step 2). In fact, the model is examined with the help of a domain expert or an end-user, who are the most qualified persons to describe the context of the peculiar domain and to put in evidence the contextual knowledge. According to the enterprise best practices and its data, they would clean and better organise the knowledge represented in the derived model. However, the obtained initial conceptual model, in the form of a UML class diagram, has yet a major limit. In fact, its semantics is in a tacit form because all the attributes are buried inside single classes and it is then difficult to make their semantics explicit.

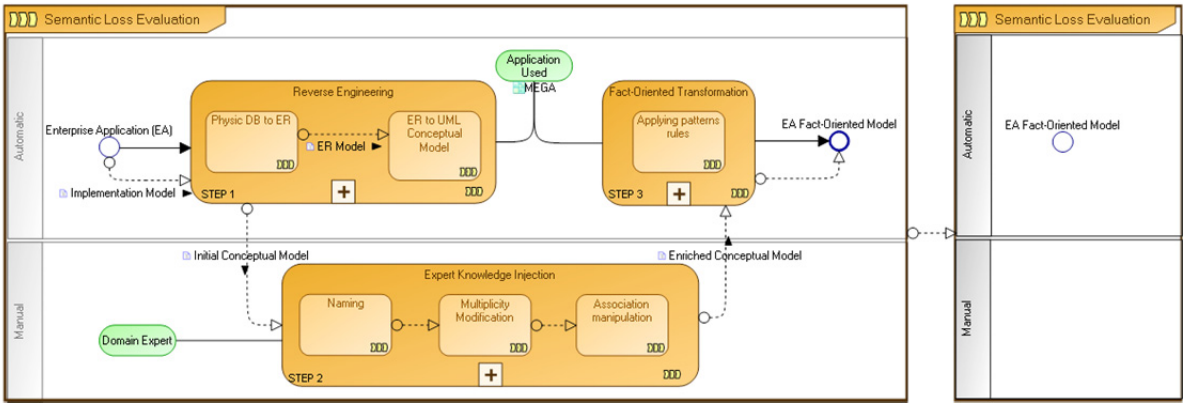


Figure 1 – Conceptualisation approach

Thus, the next step of our approach (Step 3) is a Fact-Oriented Transformation (Halpin, 1991) through the application of a set of patterns rules for transforming the enriched conceptual model to a fact-oriented model (FOM) with its semantics completely displayed. The consequence is that all the classes and their attributes are transformed into respectively LOTs and NOLOTs objects. The resulting fact-oriented model, displaying the finest-grained semantic atoms, is then used as an input for the second part of our methodology for semantic loss evaluation (not presented in this paper).

In the following sub-sections, we will discuss, in detail, the proposed 3 steps.

3.1 Step 1: Reverse Engineering

Conceptualisation is a decision process (Guarino, 1998), a view, in which studied part of reality knowledge, usually in an implicit and complex form, is reorganised in different aggregates usually simpler to be represented.

According to (Engelbart, 1962), developing conceptual models means specifying the essential objects or components of the system to be studied, the relationships of the objects that are recognised and what kinds of changes in the objects or their relationships affect the functioning of the system and in which ways.

Conceptual models range in type from the more precise, such as the mental image of a familiar physical object, to the abstractness of mathematical models that do not appear to the mind as an image. Conceptual models also range in terms of the scope of the subject matter that they are taken to represent. The variety and scope of conceptual models is due to the variety of purposes that people had while using them.

Conceptualisation approaches are numerous and have been developed in different knowledge domains (LaOnsgri, 2009).

Our scenario assumes that we start from enterprise application database. So, the first studied approach is the Reverse engineering. It is, in database (DB) community, an approach to extract the domain semantics from the existing database structures. Typically, it concerns making the reverse transformation from logical to conceptual schema. In (Fonkam, 1992), the authors propose a general algorithm based on several old attempts to make explicit the logical structure buried into DB schemas, application programs and in the minds of designers and developers. (Chiang, 1994) presents a methodology for extracting an extended Entity-Relationship model from a relational database, through a

combination of data schema and data instance analysis. In our study we will consider at profit the reverse engineering experiences developed in the past. These methods are, by now, acquired by the software industry that produces countless tools. We choose MEGA Suite (http://www.mega.com), a modelling management environment to transform relational models into conceptual ones.

3.2 Step 2: Expert Knowledge Injection

After the reverse engineering process has created a conceptual model, the current step concerns enriching it by injecting the enterprise knowledge, expressed by users' best practices or experts. These stakeholders know the domain peculiarities and they are capable to embed specific constraints into the new conceptual model. The first stage is the renaming process. Usually the database tables, and the derived concepts, have not standard names. The renaming process is essential to bring coherence and semantics in concepts that otherwise would be of very difficult comprehension. The following stage is the redefinition of the attributes and of the associations' roles multiplicities according to the enterprise users' best practices. This step is fundamental to define the real constraints that are not always made explicit into the implementation model. As an example, considering a particular attribute a_1, two cases can be considered:
1) a_1 is a non-mandatory attribute in the conceptual model but, as users are requested to always fill it with a specific value, the enriched model must formalise that this attribute a_1 is to be treated as mandatory;
2) a_1 is defined as mandatory in the conceptual model but, by practice, the users never care about its value and fill it with some dummy one. In such case, the enriched model may formalise that this attribute is not mandatory.
Note that the same cases may happen also to the roles of associations.
The last stage concerns of making explicit some implicit associations. Those implicit associations relate some concepts but they are defined only by enterprise practices even if they are not expressed in the model itself.
At this time, the enriched conceptual model formalises the whole application semantics (both the explicit one and the users' implicit one).

3.3 Step 3: Fact-Oriented Transformation

The quality of a conceptual model is often influenced by the conceptual language used for its specification. Most conceptual languages for data modelling are based on a version of Entity-Relationship modelling (E-R) (Barke, 1990) (Czejdo et al., 1990) (Hohenstein et al., 1991). However, these modelling languages are making a distinction between entities, attributes and relationships. On the contrary, in order to normalise the way that knowledge is represented, NIAM (Natural-language Information Analysis Method) (Nijssen & Halpin 1989) proposed to model the world in term of facts (either presenting terms (real things), or representing characteristics (attributes) of these real things), and relationships between facts. NIAM is attribute-free, it does not use explicitly the notion of attribute, treating all elementary facts as relationships. Some authors have extended the concepts and notations developed by NIAM with object orientation. It is the case of ORM (Object Role Modelling) (Halpin, 1998). Our purpose is to adapt this fact-oriented modelling approach to enriched conceptual models represented using the UML (OMG,

2004) class notation. Thus, we developed a set of transformation modelling rules, to be applied to selected UML patterns (Table 1). Let us refer to the definitions of LOT and NOLOT facts given in the beginning of section 3. Transforming a particular conceptual model in a fact-oriented model must follow these rules:

1. all classes are transformed into LOT facts. Using UML Class notation, a LOT fact is represented by a UML Class.
2. all attributes are transformed into NOLOT facts. Using the UML Class notation, a NOLOT fact is represented as a UML Class.
3. for each attribute *a* belonging to a UML Class *C*, an association is created between the corresponding LOT *a* and the corresponding NOLOT *C*, created by the two previous rules.
4. the multiplicity associated to each attribute *a* is copied as the multiplicity of the role of the previous (rule 3) association attached to the NOLOT *a*. The opposite role of the same association must have a constraint multiplicity equal to one.
5. all "simple" associations between classes are transformed into "simple" associations between NOLOTs.
6. all generalisation relationships between classes are transformed into "simple" associations with a constraint multiplicity equal to one on the role attached to generalised NOLOT and a non constraint multiplicity equal to * on the opposite role.
   In order to trace the fact that this association was coming from a generalisation, we annotate semantically the new corresponding association with a logical rule using OCL (Object Contraint Language) notation.
   Moreover, the inheritance feature of the generalisation association is mapped as new associations between LOTs representing the attributes of the generalised NOLOT, and all the specialised NOLOTs (sub-classes).
7. composition and aggregation relationships are transformed into simple association (rule 3) that keep unchanged the existing roles' multiplicities but trace their specific semantics through an attached semantic annotation formalised with an OCL logical rule.
8. association classes are transformed into a LOT fact with two associations linked to the corresponding initial LOT facts. The multiplicities of the roles of these two associations are determined inverting the ones initially formalised on the roles of the previous association.
9. any other specific constraints (generally modelled using OCL logical rules) are kept during the transformation process.
10. we did not take into account special cases of constraints in generalisations because they are not usually used in data conceptual modelling.

One of the conceptual modelling requirements is that a conceptual model must have formal foundations, which allow comparing that model with other conceptual models in a formal and exact way.

3.4 Patterns represented in FOL

(Berardi et al, 2005) and (Tursi, 2009) formalise UML class constructs semantics in First Order Language (FOL) axioms. We

propose to adapt these works to formalise the fact-oriented model patterns (presented previously) in FOL axioms.
Due to the lack of space we will present only one pattern rule formalisation in FOL: the "Class and Attributes" as reported in Table 5.

A class in UML designates a set of object with common features. Formally a *class C* corresponds to a FOL *unary predicate C*.

An *attribute a* of type *T* for a *class C* associates to each instance of *C* a set of instance of *T*, its multiplicity *[i..j]* specifies that *a* associates to each instance of *C* at least *i* and at most *j* instances of *T*. Formally, an attribute *a* of type *T* for *class C* corresponds to a binary predicate.

An association in UML is a relation between the instances of two or more classes. The multiplicity *[m..n]* attached to the role of a binary association specifies that each instance of the class *C* can participate at least *m* times and at most *n* times to the

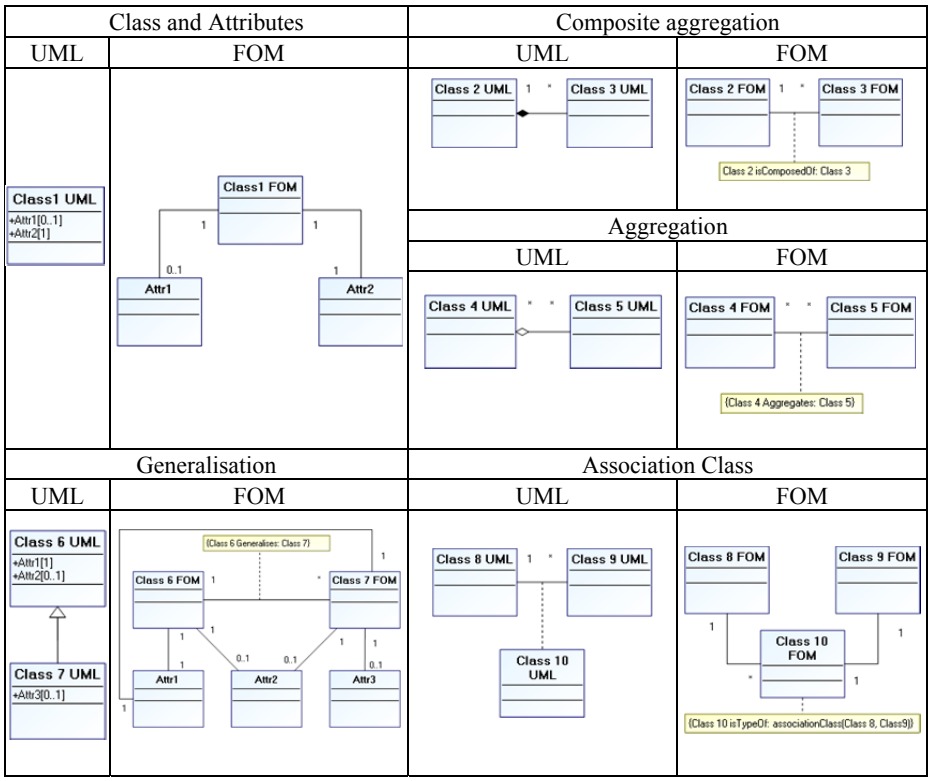


Table 5 - Fact-Oriented modelling patterns using UML notation

related association. An association *A* between two classes can be formalised as a binary predicate. In the studied pattern, we formalise a class C_1 containing two attributes A_1 and A_2 with respectively a multiplicity of 1 and [0..1], and with associated types respectively, A_1Type and A_2Type.

Its formalisation in FOL assertions is the following:

$$\forall x, y, \left(C_1(x) \wedge A_1(x, y)\right) \supset A_1 Type(y)$$
$$\forall x, z, \left(C_1(x) \wedge A_2(x, z)\right) \supset A_2 Type(z)$$

$$\forall x, C_1(x) \supset (1\ \{y|A_1(x,z)\})$$
$$\forall x, C_1(x) \supset (0 \leq 1\ \{y|A_1(x,y)\})$$

Applying the transformation rule, presented in 3.3, to the class C_1, and to the two attributes A_1 and A_2, we will obtain the Fact-Oriented Model (FOM) in UML notation as shown in Table 5 "Class and Attributes".

Its formalisation in FOL assertions is the following:

$$\forall x_1, x_2, Assoc_1(x_1, x_2) \supset C_1(x_1) \wedge A_1(x_2)$$
$$\forall x_1, C_1(x_1) \supset (1\ \{y|Assoc_1(x_1, y)\})$$
$$\forall x_2, A_1(x_2) \supset (1\ \{y|Assoc_1(x_2, y)\})$$
$$\forall x_1, x_2, Assoc_2(x_1, x_2) \supset C_1(x_1) \wedge A_2(x_2)$$
$$\forall x_1, C_1(x_1) \supset (0 \leq 1\ \{y|Assoc_2(x_1, y)\})$$
$$\forall x_2, A_2(x_2) \supset (1\ \{y|Assoc_2(x_2, y)\})$$

Using a FOL engine such as the Haskel engine (http://www.cs.yale.edu/homes/cc392/node1.html), based on Russel and Norvig algorithms (Russel and al, 1995), we are able to demonstrate that the semantics formalised in the initial conceptual model is equivalent or included into the one transformed in FOM.

4 Case study

Interoperability between organisational and manufacturing activities is crucial in manufacturing enterprises. Production services have to produce, quickly and efficiently, the good product at the right moment. For this reason, they need at time information coming from others services, which need in return precise and update data on production.

We propose here to study and present the first part of such a B2M interoperability issue by considering a particular IS implemented in a real manufacturing environment: Sage X3 as an Enterprise Resource Planning (ERP) application.

4.1 Specific analysed Enterprise Information System: Sage X3 ERP

An Enterprise Resource Planning (ERP) is an integrated computer-based system used to manage internal and external resources including tangible assets, financial resources, materials, and human resources (Bidgol, 1997). Its purpose is to facilitate the flow of information between all business functions inside the boundaries of the organization and manage the connections to outside stakeholders. Built on a centralized database, ERP systems centralise all business operations into a uniform system environment. Sage X3 presents different enterprise management functions: finance, commercial, industrial and services.

The focus for this case study is (i) to analyse how the work order process inside the Sage X3 application is modelled, (ii) to use the proposed modelling process to externalise the implicit knowledge in the model structure.

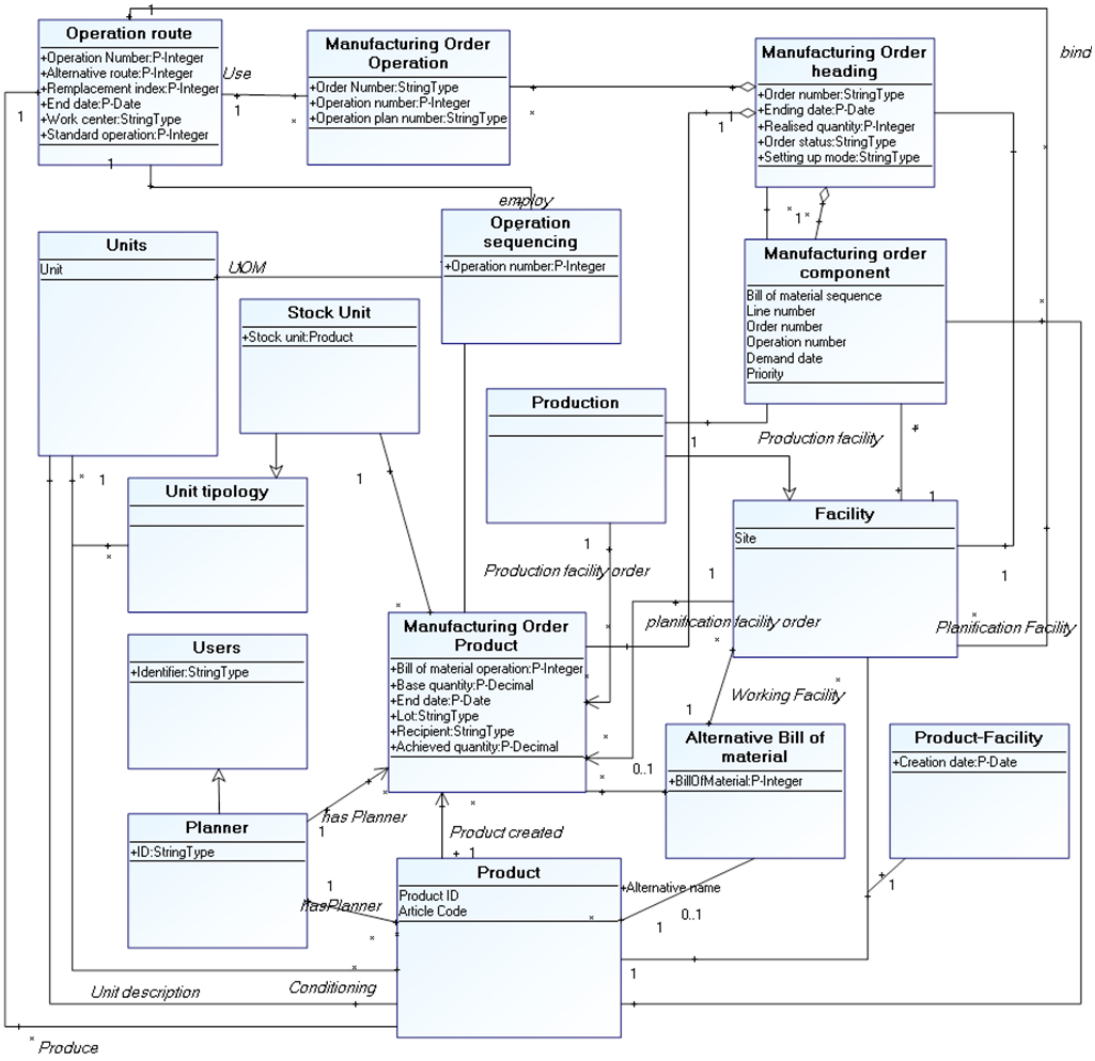


Figure 2 – Sage X3 work order enriched process model

The model depicted in figure 2 is already the result from the two first steps of our approach. This means that we have already passed the "Reverse Engineering" and the "Expert knowledge injection" stages. The "Manufacturing Order Heading" concept is the management function of production orders and planned activities. It allows the generation of a production order by variation of one or more classifications and a single production line. For each production order, the achievement of the material benefits and sequencing operations is possible. This block captures general information about the work order, such as, planning facility and facility of production, status of the order (manufacturing order product). It allows entering general information about the production order. The availability of components is checked through the information given by the bill of material related with the launched products. Once that initial information is determined, the system updates the list of materials and operations of the created or modified orders.

Step 1: Reverse Engineering
All these information are coded in the Sage application database. The first step of our method is the reverse engineering to extract the initial conceptual model.

Step 2: Expert Knowledge Injection
Currently the model depicted in Figure 2 is the result of the reverse engineering step enriched by a domain expert because the architecture of the Sage X3 ERP is built with all the database relationships implemented directly into the application layer and not in the database. The reverse engineering result, as shown in the lower part of the Figure 3, creates a model containing unlinked classes with coded names.

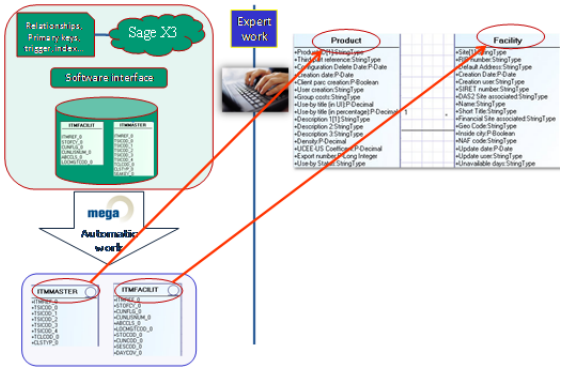


Figure 3 – Sage X3 architecture and expert knowledge injection

The expert work was about cleaning this conceptual model according to the best practices in the enterprise, modifying the attributes multiplicity, adding explicit names to the concepts, the attributes and the associations and others operations to fit the conceptual model to the "real" use of the Enterprise Information System. A usual case that requests the domain expert attention is about the mandatory properties in forms' attributes.

Step 3: Fact-Oriented Transformation
Applying the pattern transformation rules, presented in the previous section, class attributes are transformed into NOLOTs to increase the atomic representation of the knowledge embedded into the model. These rules have been coded using a programming language and then automatically executed inside MEGA Suite.

Figure 4 shows an extract of the resulting FOM after applying our approach to the Sage X3 work order process. The resulting full FOM is composed of 23 NOLOTs, 56 LOTs and 46 associations.
It seems then that the resulting model is much more complex than the initial one, which it is true in a visual point of view but it is false in term of expressiveness of its semantics. Indeed, the fine-grained atoms of semantics are now made explicit, which helps any automatic computing. An important result is that such semantically detailed model will help automating the next part of our methodology for semantic gap evaluation, as explained in section 3.

5 Conclusions

In this article, a conceptualisation approach for enacting implicit semantics from Enterprise Information Systems is proposed. Our approach if divided into 3 steps from the traditional reverse engineering process, through a knowledge elicitation and model enrichment by domain experts, till making use of fact-oriented modelling patterns to externalise tacit knowledge. These patterns have been formalised in FOL axioms to verify their semantic coherence. Our contribution can be assimilated to a reverse engineering methodology. However, the main objective is to formalize the whole semantics of such models in order to help automatic knowledge computing. An industrial case

study, related to an enterprise information system implemented into an ERP system demonstrates the applicability of our approach.

Our current work concerns applying this approach for evaluating the (non)-interoperation through the measurement of the semantic gap occurring between CISs interoperability (Yahia, 2011).

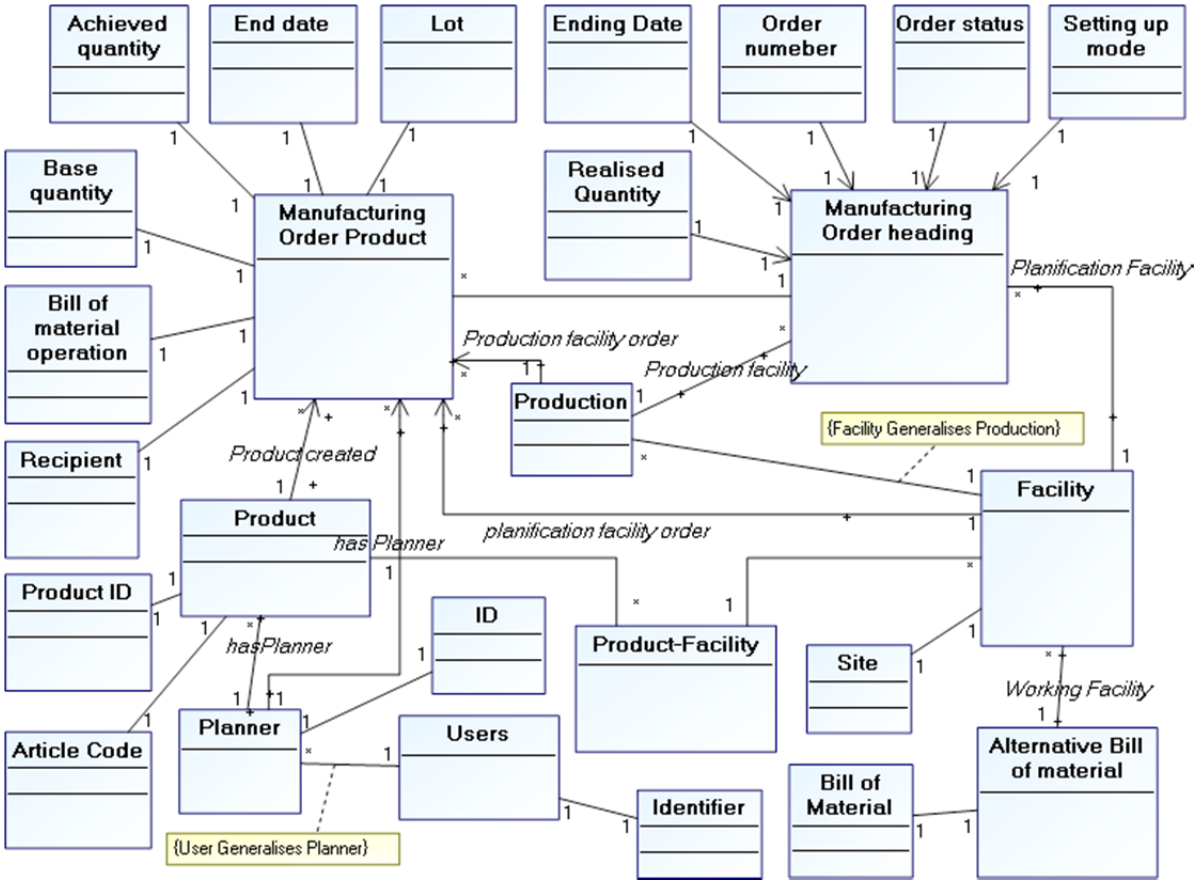


Figure 4 – Sage X3 work order process model part transformed with fact-oriented approach

REFERENCES

Aspray, W. F. 1985. The Scientific Conceptualization of Information: A Survey. Annals of the History of Computing, 7: 117-40. IEEE

Badia, A., 2002. Conceptual Modeling for Semistructured Data. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering Workshops* (WISE 2002 Workshops), p. 170-177. Singapore.

Barker, R. 1990, CASE* Method: Entity Relationship Modelling, Addison-Wesley, Wokingham, England.

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

Bézivin, J., Kurtev, I. 2005. Model-based Technology Integration with the Technical Space Concept. Proceedings of the Metainformatics Symposium, Esbjerg, Denmark, November 8-11, 2005. Springer-Verlag

Bidgol, H., 2004. *The Internet Encyclopedia*, Volume 1, John Wiley & Sons, Inc. p. 707.

Carney, D., Fisher, D., Morris, E., Place P., 2005. Some current Approaches to Interoperability. In *technical note* CMU/SEI-2005-TN-033.

Chen, D., Dassisti, M., Elvesaeter, B., Panetto, H., et al., 2006. In *DI.2: Enterprise Interoperability Framework and knowledge corpus*, Interoperability Research for Networked Enterprises Applications and Software Network of Excellence, n° IST 508-011.

Chiang Roger, H. L., Barron, T. M., Storey Veda, C., 1994. Reverse engineering of relational databases: Extraction of an EER model from a relational database. In *Data & Knowledge Engineering*. Vol. 12, Issue 2, 107-142.

Czejdo, B., Elmasri, R., Rusinkiewicz, M. & Embley, D.W. 1990, 'A graphical data manipulation language for an extended entity-relationship model', IEEE Computer, March 1990, pp. 26-37.

Engelbart, D.C., 1962. Augmenting human intellect: a conceptual framework. In *Menlo Park, CA*: Stanford Research Institute.

Euzenat, J., 2001. Towards a principled approach to semantic interoperability. In *CEUR Proceedings of the IJCAI-01* Workshop on Ontologies and Information Sharing, Seattle, USA, August 4-5, , ISSN 1613-0073, Vol. 47., 19-25.

Fonkam, M.M., Gray, W.A., 1992. An Approach to Eliciting the Semantics of Relational Databases. In *CAiSE 1992*, Manchester, UK, May 12-15, 1992. Lecture Notes in Computer Science 593 Springer, ISBN 3-540-55481-5. 463-480 Manchester, UK.

Frankel D. S. 2003. Model Driven Architecture: Applying MDA to Enterprise Computing. John Wiley & Sons.

Guarino, N., 1998. *Formal Ontology in Information Systems* (Ed.) IOS Press.

Halpin, T.A. 1991, 'A fact-oriented approach to schema transformation', Proc. MFDBS-91, Springer Lecture Notes in Computer Science, no. 495, Rostock.

Halpin, T., 1998. *Handbook on Architectures of Information Systems* Chapter 4, eds P. Bernus, K. Mertins & G. Schmidt, Springer-Verlag, Berlin.

Henrard, L., Hainaut, J.-L., 2001. Data Dependency Elicitation in Database Reverse Engineering Software Maintenance and Reengineering. In *Fifth European Conference on Software Maintenance and Reengineering*, pp. 11

Hohenstein, U. & Engels, G. 1991, 'Formal semantics of an entity-relationship-based query language', Entity-Relationship Approach: the core of conceptual modelling (Proc. 9th ER conf.), ed. H. Kangassalo, Elsevier Science Pub., Amsterdam

IEEE: Standard Computer Dictionary, 1990. A Compilation of IEEE Standard Computer Glossaries. In *NY. 610-1990*. ISBN: 1559370793.

International Organization for Standardization, 1999. ISO 14528: Industrial Automation Systems – Concepts and rules for Enterprise Models, *TC 184/SC5/WG1*, Geneva, Switzerland.

International Organization for Standardization, 2002. ISO 16100: Manufacturing Software Capability Profiling for interoperability. In *Part 1: Framework, TC 184/SC5/WG4*, Geneva, Switzerland.

LaOngsri, S. 2009, Semantic Extensions and a Novel Approach to Conceptual Modelling, Ph.D. Thesis, School of Computer Science, Engineering and Mathematics, The Flinders University of South Australia

Mani, M.: EReX, 2004. A Conceptual Model for XML. In *Proceedings of the Second International XML Database Symposium* (XSym 2004), p. 128-142. Toronto, Canada.

Manola, F., Miller, E., 2004. RDF Primer. In *World Wide Web Consortium*, Recommendation REC-rdf-primer-20040210.

Meersman, R., Tari, Z., 2003. On The Move to Meaningful Internet Systems 2003. In *OTM 2003 Workshops OTM Confederated International Workshops*, Lecture Notes in Computer Science, Vol. 2889. ISBN: 978-3-540-20494-7, Catania, Italy.

Nijssen, G.M. & Halpin, T.A. 1989, Conceptual Schema and Relational Database Design, Prentice Hall, Sydney.

Obrst, L., 2003 Ontologies for semantically interoperable systems. In *Proceedings of the 12th International Conference on Information and Knowledge Management*. New Orleans, USA

OMG, 2003. Object Management Group. Architecture-Driven Modernization specification http://adm.omg.org

OMG, 2004. Object Management Group. UML 2.0 Superstructure Specification http://uml.omg.org

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen W., 1991. Object Oriented Modeling and Design. *Prentice Hall*, Book Distribution Center, New York, USA.

Russel, S., Norvig, P. *Artificial Intelligence, A Modern Approach"*, Prentice-Hall. Inc. 1995

Seeley, R. S., 1997. Manufacturing execution systems in *MED DEVICE DIAGN IND.* Vol. 19, no. 11, pp. 64-68.

Sheth, A., 1998. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. In M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman, editors. In *Interoperating Geographic Information Systems*, pp. 5– 30. Kluwer.

Smith, M.K., Welty, Ch., McGuinness, D.L., 2004. OWL Web Ontology Language Guide. In *World Wide Web Consortium*, Recommendation REC-owl-guide-20040210.

Tolk, A., Diallo, S. Y., Turnitsa, C. D., 2007. Applying the Levels of Conceptual Interoperability Modelling Support of Integrability, Interoperability, and Composability for System-of-Systems Engineering. In *Journal of Systemics, Cybernetics and Informatics*, Volume 5 Number 5, pp. 65-74.

Tursi, A., Panetto, H., Morel, G., Dassisti, M., 2009 Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. In *IFAC Annual Reviews in Control*. 33/2, 238-245, Elsevier, ISSN: 1367-5788.

Vyvyan, E., 2006. Lexical Concepts, Cognitive Models and Meaning-Construction. In *Cognitive Linguistics* 17 (4): 491-534.

Yahia E., Yang J., Aubry A., Panetto H., 2009. On the use of Description Logic for Semantic Interoperability of Enterprise Systems. In *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, Springer Verlag, Lecture Notes in Computer Science, LNCS 5872, ISBN 978-3-540-88874-1.

Yahia E., Lezoche M., Aubry A., Panetto H. 2011. Semantics enactment in Enterprise Information Systems. 18th IFAC World Congress. Milan, Italy

10 Appendix B

Formal Fact-Oriented model transformations for cooperative information systems semantic conceptualisation

Mario Lezoche, Alexis Aubry, Hervé Panetto

Research Centre for Automatic Control (CRAN), Nancy-University, CNRS, Campus *Scientifique, Faculté des Sciences et Technologies, BP 70239, 54506 Vandoeuvre-lès-Nancy Cedex, France*
{mario.lezoche, alexis.aubry, herve.panetto}@cran.uhp-nancy.fr

Abstract: **Information in enterprise is, now more than ever, a fundamental resource. In order to increase enterprise performance, economics paradigms focus on how to better manage it. Information Systems (IS) are systems whose activities are devoted to capture and to store data, to process them and produce knowledge, used by any stakeholders within an enterprise or among different networked enterprises. The modern architecture of information systems is based on distributed networks. An important challenge, to reach higher performance, is to represent and share knowledge managed by those ISs. One of the main issues in making such heterogeneous Cooperative Information Systems (CIS) working together is to remove semantics interoperability barriers. This paper firstly analyses interoperability issues between CISs and then proposes a systematic approach for data models conceptualisation for knowledge explicitation, based on initial conceptual model cleaning rules, expert knowledge injection rules and finally fact-oriented transformation rules. A case study is proposed, related to a work order process in an Enterprise Resource Planning application, Sage X3.**

Keywords: **Conceptual modelling, cooperative information systems, formal verification, data model conceptualisation**

1 Introduction

The actual archetype for the Information Systems (ISs) involves large number of ISs distributed over large, complex computer/ communication networks. Such cooperative information systems (CIS) have access to large amount of information and have to interoperate to achieve their purpose. The cooperative information systems architects and developers have to face a hard problem: interoperability.
Interoperability can be defined as the ability for two or more systems to share, to understand and to consume information [20]. Some work [8] in the INTEROP NoE project has identified three different levels of barriers for interoperability: technical, conceptual and organisational. Organisational barriers are still an important issue but out of scope of this paper. The technological barriers are strongly studied by researchers in computer science and the solution is generally based on models transformation [14].
Our research [24] focuses on the conceptual level of interoperability that is the ability to understand the exchanged information. A concept is a cognition unit of meaning [39], an abstract idea, a mental symbol. It is created through the action of conceptualisation, that is, a general and abstract mental representation of an object. During the history of human effort to model knowledge, different conceptualisation approaches regarding different application domains were developed [1].
This paper is dealing with a first step from a more general work focusing on the study of the semantic loss during the exchange of information representing business concepts. Quantifying the semantic gap between interoperating ISs implies enacting their semantics through their normalized conceptual models. Indeed, in this context, the starting point for semantics interoperability is related to models conceptualisation.

We will present a conceptualisation approach to make explicit the finest-grained semantics embedded into conceptual models for finally enabling two different information systems seamlessly interoperating.

Next section presents the general context of our work. Then, the following section details the fundamental pillars of our conceptualisation process. Then, we will propose a knowledge explicitation process starting from an implemented relational model to a fact-oriented conceptual one. This process allows us emphasizing the finest-grained semantics that must be enacted to study semantics interoperability between collaborating ISs.

Finally, to validate our proposal, a practical case study is presented based on an Enterprise Resource Planning application involved in a B2M (Business to Manufacturing) interoperation process.

2 Cooperative Information systems

Information Systems are systems whose activities are devoted to capture and to store data, to process them and produce knowledge, used by any stakeholders within an enterprise or among different networked enterprises. It is commonly agreed that Cooperative Information Systems provide a backbone for the Integrated Information Infrastructure [35]. Fully

understanding and exploiting the advances in computing is the only way to encompass the complexity of constructing and maintaining such systems.

Although the progress made in information technology considerably improved the efficiency of applications development, its drawbacks and limitations are obvious and serious. In fact, the application models involved in a single application are numerous and different, each coping only with particular and partial aspects of the overall task. Moreover, the components technologies are heterogeneous, platform- and machine-dependant. The above-mentioned limitations and barriers measurably hinder the development and the maintenance process.

There is a growing demand to integrate such systems tightly with organizational work so that these information systems can be directly and immediately used by the business activity.

Here, the need of interoperation clearly appears. In fact, to achieve the purpose of the cooperation between the different Information Systems, information must be physically exchanged (technical interoperability), must be understood (conceptual interoperability) and must be used for the purpose that they have been produced (conceptual and organisational interoperability). When trying to assess the understanding of an expression coming from a system to another system, there are several possible levels of interoperability [12]:

- *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).

This tiered structure is arguable in general; it is not as strict as it seems. It makes sense because each level cannot be achieved if the previous levels have not been completed [12].

The encoding, lexical and syntactic levels are the most effective solutions for removing technical barriers for interoperability, but not sufficient, to achieve a practical interoperability between computerised systems. Dealing with trying to enable a seamless data and model exchange at the semantic level is still a big issue that needs conceptual representation of the intended exchanged information and the definition of the pragmatic meaning of that exchanged information in the context of the source and destination applications.

Different cooperation types have been investigated in ISO 14528 [21]. In fact, this standard considers that models could be related in three ways:

(4) integration when there exists a standard or pivotal format to represent these models;

(5) unification when there exists a common meta-level structure establishing semantic equivalence between these models; and

(6) federation when each model exists per se, but 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. Classifying interoperability problems [38] may help in understanding the degree of development needed to solve, at least partially, these problems but conceptualisation and semantics extraction is still an important issue because of the various contextual understanding of tacit knowledge embedded into those applications. The main prerequisite for achievement of interoperability of information systems is to maximise the amount of semantics which can be used and make it increasingly explicit [29], and consequently, to make the systems semantically interoperable. To highlight this issue, the paper is based on a referenced scenario involving enterprise systems applications.

Most of reverse engineering approaches [13] [9] return the information structure but present a model with tacit semantics. The ADM (Architecture-Driven Modernization) initiative [30] from OMG [5] is tackling this problem by promoting a common Knowledge Discovery Meta-model to facilitate discovering tacit knowledge embedded inside existing software. In our scenario, those applications are still implemented and running using databases. We can extract, from them, by using reverse engineering approaches, some knowledge in a form of a conceptual model. We have then to enrich that model with enterprise applications best practices (knowledge coming from users). Finally, we make explicit all disclosed knowledge hidden in the resulting model.

3 Our approach for Semantics Enactment In Conceptual Models

In order to cooperate, two (or more) Information Systems have to interoperate. As previously discussed, we focus our interest on the conceptual level of interoperability letting different information systems to share and use knowledge models that they represent. Our principal issues are, therefore, first to understand the conceptual relationships between those models in the context of their use and secondly how, through conceptualisation, to unhide the tacit knowledge buried inside them. A usual approach for making explicit the tacit knowledge, concealed in attributes and classes, is the relationships-oriented perspective composed of a set of transformation rules. In that transformation method, an attribute a_1 of type T_1 pertaining to a class C_1 is

modelled as a relationship between the class C_1 and a standard type T_1. This approach does not resolve entirely the semantics elicitation problem because it focuses its point of view on the values instead of on the concepts. The attribute semantics is somewhat yet hidden in the relationship just created.
In literature, [27] presented the definition of two different objects types, *a lexical object (LOT),* a term*, is an object in a certain reality that can be written down. LOTs always consist of letters, numbers, symbols or other characters. They can be used as names for or references to other objects. A non-lexical object (NOLOT),* a concept, *is an object in a certain reality that cannot be written down. Non-lexical objects must be named by lexical objects or referred to by means of lexical objects*.
Applying these definitions, we can flatten the nested knowledge embedded in a model to simplify semantic enactment resulting from a set of modelling transformations. Our contribution is to have at our disposal an approach letting us to fragment knowledge through the transformation of attributes into entities and relationships, and thus to emphasize some fine-grained knowledge atoms. In the proposed approach, that is the first part (Figure 1) of our general methodology, the starting point can be various: an application, a data model, a logical view, a model. We have already mentioned that there are several reverse engineering methods, such as in [13] and in [9], through which a model from the application or schema level can be derived (Step 1). Then, the resulted initial model is enriched and corrected through an Expert Knowledge Injection step (Step 2). In fact, the model is examined with the help of a domain expert or an end-user, who are the most qualified persons to describe the context of the peculiar domain and to put in evidence the contextual knowledge. According to the enterprise best practices and its data, they would clean and better organise the knowledge represented in the derived model. However, the obtained initial conceptual model, in the form of a UML class diagram, has yet a major limit. In fact, its semantics is in a tacit form because all the attributes are buried inside single classes and it is then difficult to make their semantics explicit.

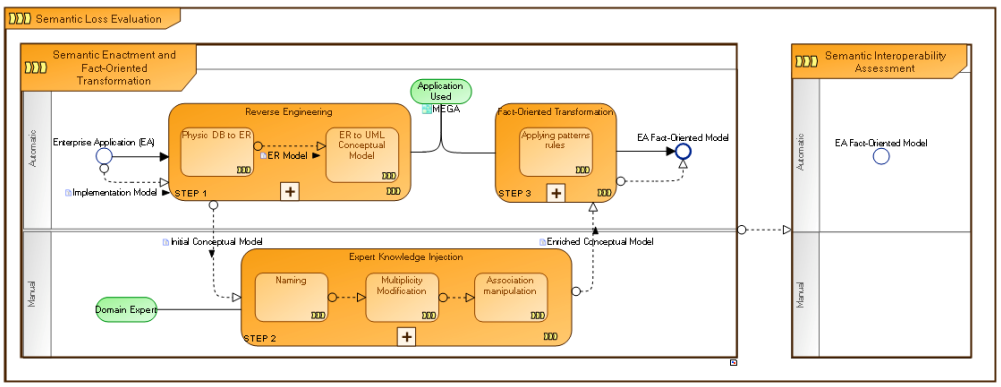


Figure 18 - Conceptualisation approach

Thus, the next step of our approach (Step 3) is a Fact-Oriented Transformation [16] through the application of a set of rules for transforming the enriched conceptual model to a fact-oriented model (FOM) with its

semantics completely displayed. The consequence is that all the classes and their attributes are transformed into respectively LOTs and NOLOTs objects. The resulting fact-oriented model, displaying the finest-grained semantic atoms, is then used as an input for the second part of our methodology for semantic loss evaluation (not presented in this paper).
In the following sub-sections, we will discuss, in detail, the proposed 3 steps.

3.1 Step 1: Reverse Engineering

Conceptualisation is a decision process [15], a view, in which studied part of reality knowledge, usually in an implicit and complex form, is reorganised in different aggregates usually simpler to be represented.
According to [11], developing conceptual models means specifying the essential objects or components of the system to be studied, the relationships of the objects that are recognised and what kinds of changes in the objects or their relationships affect the functioning of the system and in which ways.
Conceptual models range in type from the more precise, such as the mental image of a familiar physical object, to the abstractness of mathematical models that do not appear to the mind as an image. Conceptual models also range in terms of the scope of the subject matter that they are taken to represent. The variety and scope of conceptual models is due to the variety of purposes that people had while using them.
Conceptualisation approaches are numerous and have been developed in different knowledge domains [23].
Our scenario assumes that we start from enterprise application database. So, the first studied approach is the Reverse engineering. It is, in database (DB) community, an approach to extract the domain semantics from the existing database structures. Typically, it concerns making the reverse transformation from logical to conceptual schema. In [13], the authors propose a general algorithm based on several old attempts to make explicit the logical structure buried into DB schemas, application programs and in the minds of designers and developers. [9] presents a methodology for extracting an extended Entity-Relationship model from a relational database, through a combination of data schema and data instance analysis. In our study we will consider at profit the reverse engineering experiences developed in the past. These methods are, by now, acquired by the software industry that produces countless tools. We choose MEGA Suite (http://www.mega.com), a modelling management environment to transform relational models into conceptual ones. MEGA Suite implements a parameterised reverse engineering method coping with major existing approaches from direct database metadata analysis to a semi-automatic conceptual models building from existing database schemas.

3.2 Step 2: Expert Knowledge Injection

After the reverse engineering process has created a conceptual model, the current step concerns enriching it by injecting the enterprise knowledge, expressed by users' best practices or experts. These stakeholders know the domain peculiarities and they are capable to embed specific constraints into the new conceptual model. The first stage is the renaming process. Usually

the database tables, and the derived concepts, have not standard names. The renaming process is essential to bring coherence and semantics in concepts that otherwise would be of very difficult comprehension. The following stage is the redefinition of the attributes and of the associations' roles multiplicities according to the enterprise users' best practices. This step is fundamental to define the real constraints that are not always made explicit into the implementation model. As an example, considering a particular attribute a_1, two cases can be considered:
1) a_1 is a non-mandatory attribute in the conceptual model but, as users are requested to always fill it with a specific value, the enriched model must formalise that this attribute a_1 is to be treated as mandatory;
2) a_1 is defined as mandatory in the conceptual model but, by practice, the users never care about its value and fill it with some dummy one. In such case, the enriched model may formalise that this attribute is not mandatory.
Note that the same cases may happen also to the roles of associations.
The last stage concerns of making explicit some implicit associations. Those implicit associations relate some concepts but they are defined only by enterprise practices even if they are not expressed in the model itself.
At this time, the enriched conceptual model formalises the whole application semantics (both the explicit one and the users' implicit one).

3.3 Step 3: Fact-Oriented Transformation

The quality of a conceptual model is often influenced by the conceptual language used for its specification. Most conceptual languages for data modelling are based on a version of Entity-Relationship modelling (E-R) [3] [10] [19]. However, these modelling languages are making a distinction between entities, attributes and relationships. On the contrary, in order to normalise the way that knowledge is represented, NIAM (Natural-language Information Analysis Method) [28] proposed to model the world in term of facts (either presenting terms (real things), or representing characteristics (attributes) of these real things), and relationships between facts. NIAM is attribute-free, it does not use explicitly the notion of attribute, treating all elementary facts as relationships. Some authors have extended the concepts and notations developed by NIAM with object orientation. It is the case of ORM (Object Role Modelling) [17]. Our purpose is to adapt this fact-oriented modelling approach to enriched conceptual models represented using the UML [31] class notation. Thus, we developed a set of transformation modelling rules, to be applied to selected UML patterns (Table 1).
Let us refer to the definitions of LOT and NOLOT facts given in the beginning of section 3. Transforming a particular conceptual model in a fact-oriented model must follow these rules:

1. all classes are transformed into LOT facts. Using UML Class notation, a LOT fact is represented by a UML Class.
2. all attributes are transformed into NOLOT facts. Using the UML Class notation, a NOLOT fact is represented as a UML Class.
3. for each attribute *a* belonging to a UML Class *C*, an association is

created between the corresponding LOT *a* and the corresponding NOLOT *C*, created by the two previous rules.

4. the multiplicity associated to each attribute *a* is copied as the multiplicity of the role of the previous (rule 3) association attached to the NOLOT *a*. The opposite role of the same association must have a constraint multiplicity equal to one.
5. all "simple" associations between classes are transformed into "simple" associations between NOLOTs.
6. all generalisation relationships between classes are transformed into "simple" associations with a constraint multiplicity equal to one on the role attached to generalised NOLOT and a non constraint multiplicity equal to * on the opposite role. In order to trace the fact that this association was coming from a generalisation, we annotate semantically the new corresponding association with a logical rule using OCL (Object Constraint Language) notation. Moreover, the inheritance feature of the generalisation association is mapped as new associations between LOTs representing the attributes of the generalised NOLOT, and all the specialised NOLOTs (sub-classes).
7. composition and aggregation relationships are transformed into simple association (rule 3) that keep unchanged the existing roles' multiplicities but trace their specific semantics through an attached semantic annotation formalised with an OCL logical rule.
8. association classes are transformed into a LOT fact with two associations linked to the corresponding initial LOT facts. The multiplicities of the roles of these two associations are determined inverting the ones initially formalised on the roles of the previous association.
9. any other specific constraints (generally modelled using OCL logical rules) are kept during the transformation process.
10. we did not take into account special cases of constraints in generalisations because they are not usually used in data conceptual modelling.

One of the conceptual modelling requirements is that a conceptual model must have formal foundations, which allow comparing that model with other conceptual models in a formal and exact way.

3.4 Patterns represented in FOL

[4] and [38] formalise UML class constructs semantics in First Order Language (FOL) axioms. We propose to adapt these works to formalise the fact-oriented model patterns (presented previously) in FOL axioms.

We will present only one pattern rule formalisation in FOL: the "Class and Attributes" rule as presented in Table 1.

A class in UML designates a set of object with common features. Formally a *class C* corresponds to a FOL *unary predicate C*.

An *attribute a* of type *T* for a *class C* associates to each instance of *C* a set of instance of *T*, its multiplicity *[i..j]* specifies that *a* associates to each instance of *C* at least *i* and at most *j* instances of *T*. Formally, an attribute *a* of type *T* for *class C* corresponds to a binary predicate.

An association in UML is a relation between the instances of two or more classes. The multiplicity *[m..n]* attached to the role of a binary association specifies that each instance of the class *C* can participate at least *m* times and at most *n* times to the

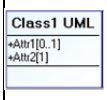
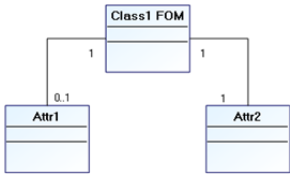
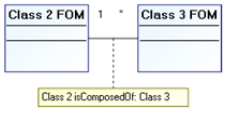
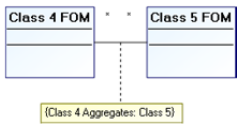
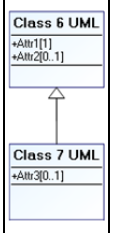
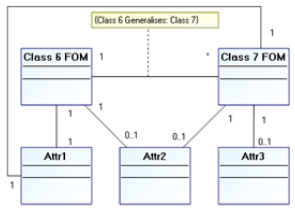
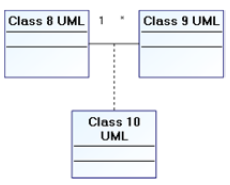
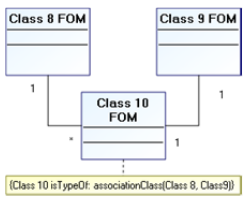
<table>
<tr><td colspan="2">Class and Attributes</td><td colspan="2">Composite aggregation</td></tr>
<tr><td>UML</td><td>FOM</td><td>UML</td><td>FOM</td></tr>
<tr><td rowspan="2"> 
</td><td rowspan="2"> 
</td><td> 
</td><td> 
</td></tr>
<tr><td colspan="2">Aggregation</td></tr>
<tr><td colspan="2">UML</td><td colspan="2">FOM</td></tr>
<tr><td colspan="2"> 
</td><td colspan="2"> 
</td></tr>
<tr><td colspan="2">Generalisation</td><td colspan="2">Association Class</td></tr>
<tr><td>UML</td><td>FOM</td><td>UML</td><td>FOM</td></tr>
<tr><td> 
</td><td> 
</td><td> 
</td><td> 
</td></tr>
</table>

Table 6 - Fact-Oriented modelling patterns using UML notation

related association. An association A between two classes can be formalised as a binary predicate. In the studied pattern, we formalise a class C_1 containing two attributes A_1 and A_2 with respectively a multiplicity of 1 and [0..1], and with associated types respectively, A_1Type and A_2Type.

Its formalisation in FOL assertions is the following:

$$\forall x, y, \left(C_1(x) \wedge A_1(x,y)\right) \supset A_1Type(y)$$

$$\forall x, z, \left(C_1(x) \wedge A_2(x,z)\right) \supset A_2Type(z)$$

$$\forall x, C_1(x) \supset (1\ \{y|A_1(x,z)\})$$

$$\forall x, C_1(x) \supset (0 \leq 1\ \{y|A_1(x,y)\})$$

Applying the transformation rule, presented in section 3.3, to the class C_1, and to the two attributes A_1 and A_2, we will obtain the Fact-Oriented Model (FOM) in UML notation as shown in Table 1 “Class and Attributes”.

Its formalisation in FOL assertions is the following:

$$\forall x_1, x_2, Assoc_1(x_1,x_2) \supset C_1(x_1) \wedge A_1(x_2)$$

$$\forall x_1, C_1(x_1) \supset (1\ \{y|Assoc_1(x_1,y)\})$$

$$\forall x_2, A_1(x_2) \supset (1\ \{y|Assoc_1(x_2,y)\})$$

$$\forall x_1, x_2, Assoc_2(x_1,x_2) \supset C_1(x_1) \wedge A_2(x_2)$$

$$\forall x_1, C_1(x_1) \supset (0 \leq 1\ \{y|Assoc_2(x_1,y)\})$$

$$\forall x_2, A_2(x_2) \supset (1\ \{y|Assoc_2(x_2,y)\})$$

We used as demonstration tools Prover9, an automated theorem prover for first-order and equational logic, and Mace4 searcher for finite models and counterexamples. We implemented our models in prover9 syntax to verify them in FOL formalised transformation models rules.

These artefacts are the constituent basis to represent the transformation rules in First Order Logic. After the translation, we own two sets of FOL expression facts. We named them A and B. They represent the formalised semantics of UML and FOM models. They can be used to verify the models semantic equality. We will use a standard verification algorithm based on set theory. The verification algorithm takes as input assumptions the two expression sets, A and B.

The goal of the verification task is to demonstrate $\forall x\ A(x) \wedge \neg B(x) \rightarrow \emptyset$.

With this evaluation method we are able to demonstrate that the semantics formalised in the initial conceptual model is equivalent or included into the one transformed in FOM and so it is sufficient for our verification purpose.

4 Case study

Interoperability between organisational and manufacturing activities is crucial in manufacturing enterprises. Production services have to produce, quickly and efficiently, the good product at the right moment. For this reason, they need at time information coming from others services, which need in return precise and update data on production.
We propose here to study and present the first part of such a B2M interoperability issue by considering a particular IS implemented in a real manufacturing environment: Sage X3 as an Enterprise Resource Planning (ERP) application.

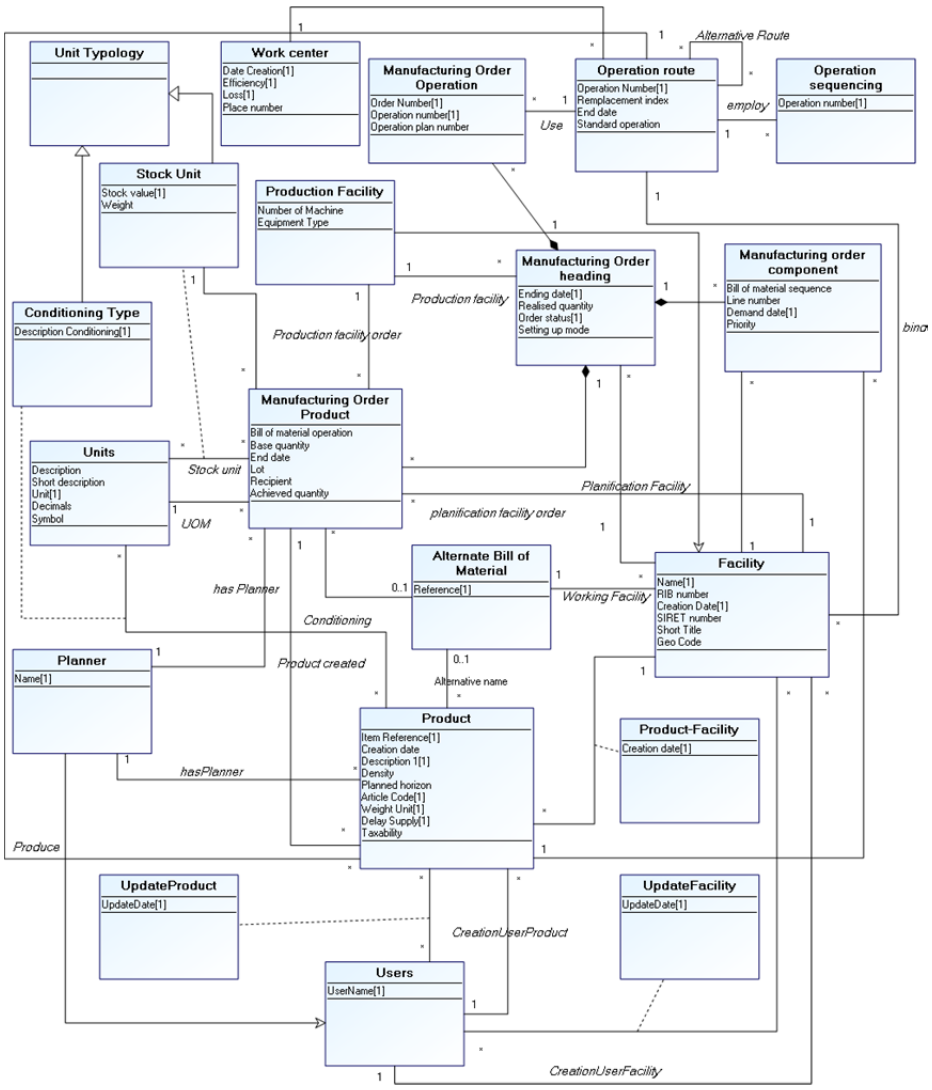


Figure 19 - Sage X3 work order enriched process model

4.1 Specific analysed Enterprise Information System: Sage X3 ERP

An Enterprise Resource Planning (ERP) is an integrated computer-based system used to manage internal and external resources including tangible assets, financial resources, materials, and human resources [6]. Its purpose is to facilitate the flow of information between all business functions inside the boundaries of the organization and manage the connections to outside stakeholders. Built on a centralized database, ERP systems centralise all business operations into a uniform system environment. Sage X3 presents different enterprise management functions: finance, commercial, industrial and services.
The focus for this case study is (i) to analyse how the work order process inside the Sage X3 application is modelled, (ii) to use the proposed modelling process to externalise the implicit knowledge in the model structure.

The model depicted in figure 2 is already the result from the two first steps of our approach. This means that we have already passed the "Reverse Engineering" and the "Expert knowledge injection" stages. The "Manufacturing Order Heading" concept is the management function of production orders and planned activities. It allows the generation of a production order by variation of one or more classifications and a single production line. For each production order, the achievement of the material benefits and sequencing operations is possible. This block captures general information about the work order, such as, planning facility and facility of production, status of the order (manufacturing order product). It allows entering general information about the production order. The availability of components is checked through the information given by the bill of material related with the launched products.
Once that initial information is determined, the system updates the list of materials and operations of the created or modified orders.

Step 1: Reverse Engineering
All these information are coded in the Sage application database. The first step of our method is the reverse engineering to extract the initial conceptual model.

Step 2: Expert Knowledge Injection
Currently the model depicted in Figure 2 is the result of the reverse engineering step enriched by a domain expert because the architecture of the Sage X3 ERP is built with all the database relationships implemented directly into the application layer and not in the database. The reverse engineering result, as shown in the lower part of the Figure 3, creates a model containing unlinked classes with coded names.

The expert work was about cleaning this conceptual model according to the best practices in the enterprise, modifying the attributes multiplicity, adding explicit names to the concepts, the attributes and the associations and others operations to fit the conceptual model to the "real" use of the Enterprise

Information System. A usual case that requests the domain expert attention is about the mandatory properties in forms' attributes.

Step 3: Fact-Oriented Transformation
Applying the pattern transformation rules, presented in the previous section, class attributes are transformed into NOLOTs to increase the atomic representation of the knowledge embedded into the model. These rules have been coded using a programming language and then automatically executed inside MEGA Suite.
Figure 4 shows an extract of the resulting FOM after applying our approach to the Sage X3 work order process. The resulting full FOM is composed of 23 NOLOTs, 56 LOTs and 46 associations. It seems then that the resulting model is much more complex than the initial one, which it is true in a visual point of view but it is false in term of expressiveness of its semantics. Indeed, the fine-grained atoms of semantics are now made explicit, which helps any automatic computing. An important result is that such semantically detailed model will help automating the next part of our methodology for semantic gap evaluation, as presented in [41].

5 Conclusions

In this article, a conceptualisation approach for enacting implicit semantics from Enterprise Information Systems is proposed. Our approach if divided into 3 steps from the traditional reverse engineering process, through a knowledge elicitation and model enrichment by domain experts, till making use of fact-oriented modelling patterns to externalise tacit knowledge. These patterns have been formalised in FOL axioms to verify their semantic coherence. Our contribution can be assimilated to a reverse engineering methodology. However, the main objective is to formalize the whole semantics of such models in order to help automatic knowledge computing. An industrial case study, related to an enterprise information system implemented into an ERP system demonstrates the applicability of our approach.
Our current work concerns applying this approach for evaluating the (non)-interoperation through the measurement of the semantic gap occurring between CISs interoperability [40] and [41].

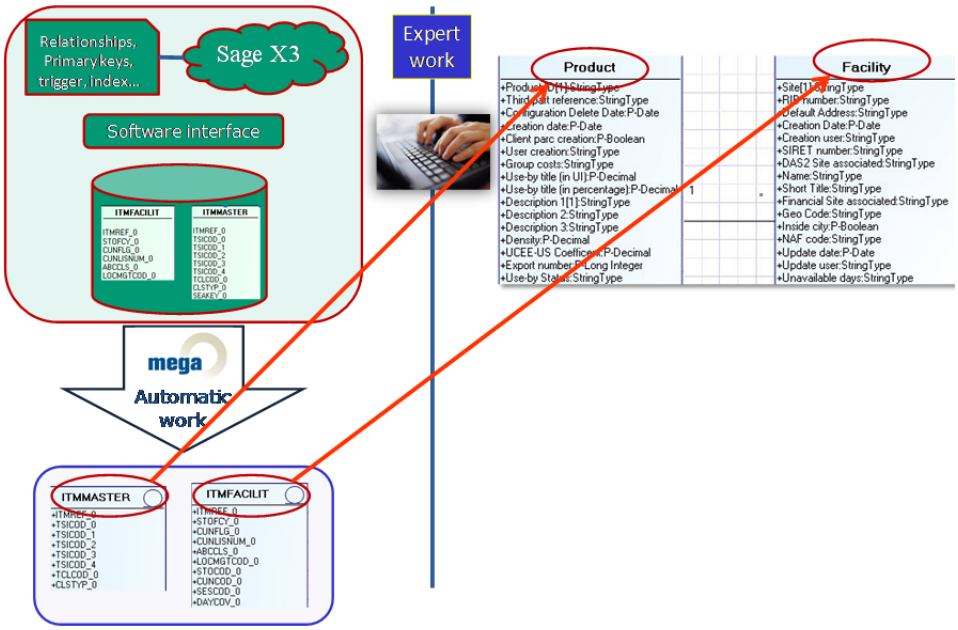


Figure 3 – Sage X3 architecture and expert knowledge injection

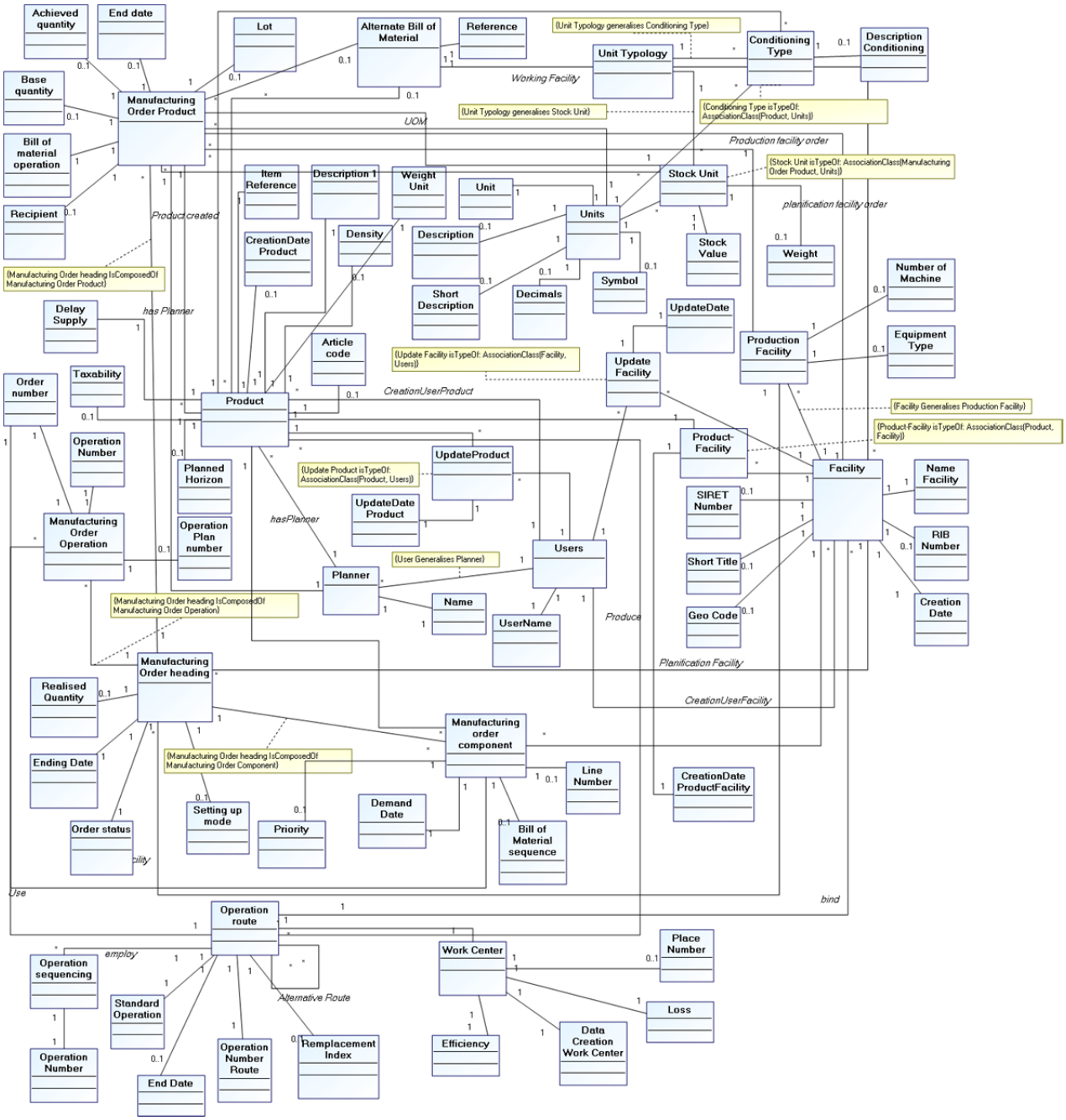


Figure 4 – Sage X3 work order process model part transformed with fact-oriented approach

REFERENCES

[1] Aspray, W. F. 1985. The Scientific Conceptualization of Information: A Survey. Annals of the History of Computing, 7: 117-40. IEEE

[2] Badia, A., 2002. Conceptual Modeling for Semistructured Data. In *Proceedings of the 3rd International Conference on Web Information Systems Engineering Workshops* (WISE 2002 Workshops), p. 170-177. Singapore.

[3] Barker, R. 1990, CASE* Method: Entity Relationship Modelling, Addison-Wesley, Wokingham, England.

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

[5] Bézivin, J., Kurtev, I. 2005. Model-based Technology Integration with the Technical Space Concept. Proceedings of the Metainformatics Symposium, Esbjerg, Denmark, November 8-11, 2005. Springer-Verlag

[6] Bidgol, H., 2004. *The Internet Encyclopedia*, Volume 1, John Wiley & Sons, Inc. p. 707.

[7] Carney, D., Fisher, D., Morris, E., Place P., 2005. Some current Approaches to Interoperability. In *technical note* CMU/SEI-2005-TN-033.

[8] Chen, D., Dassisti, M., Elvesaeter, B., Panetto, H., et al., 2006. In *DI.2: Enterprise Interoperability Framework and knowledge corpus*, Interoperability Research for Networked Enterprises Applications and Software Network of Excellence, n° IST 508-011.

[9] Chiang Roger, H. L., Barron, T. M., Storey Veda, C., 1994. Reverse engineering of relational databases: Extraction of an EER model from a relational database. In *Data & Knowledge Engineering*. Vol. 12, Issue 2, 107-142.

[10] Czejdo, B., Elmasri, R., Rusinkiewicz, M. & Embley, D.W. 1990, 'A graphical data manipulation language for an extended entity-relationship model', IEEE Computer, March 1990, pp. 26-37.

[11] Engelbart, D.C., 1962. Augmenting human intellect: a conceptual framework. In *Menlo Park, CA*: Stanford Research Institute.

[12] Euzenat, J., 2001. Towards a principled approach to semantic interoperability. In *CEUR Proceedings of the IJCAI-01* Workshop on Ontologies and Information Sharing, Seattle, USA, August 4-5, , ISSN 1613-0073, Vol. 47., 19-25.

[13] Fonkam, M.M., Gray, W.A., 1992. An Approach to Eliciting the Semantics of Relational Databases. In *CAiSE 1992*, Manchester, UK, May 12-15, 1992. Lecture Notes in Computer Science 593 Springer, ISBN 3-540-55481-5. 463-480 Manchester, UK.

[14] Frankel D. S. 2003. Model Driven Architecture: Applying MDA to Enterprise Computing. John Wiley & Sons.

[15] Guarino, N., 1998. *Formal Ontology in Information Systems* (Ed.) IOS Press.

[16] Halpin, T.A. 1991, 'A fact-oriented approach to schema transformation', Proc. MFDBS-91, Springer Lecture Notes in Computer Science, no. 495, Rostock.

[17] Halpin, T., 1998. *Handbook on Architectures of Information Systems* Chapter 4, eds P. Bernus, K. Mertins & G. Schmidt, Springer-Verlag, Berlin.

[18] Henrard, L., Hainaut, J.-L., 2001. Data Dependency Elicitation in Database Reverse Engineering Software Maintenance and Reengineering. In *Fifth European Conference on Software Maintenance and Reengineering*, pp. 11

[19] Hohenstein, U. & Engels, G. 1991, 'Formal semantics of an entity-relationship-based query language', Entity-Relationship Approach: the core of conceptual modelling (Proc. 9th ER conf.), ed. H. Kangassalo, Elsevier Science Pub., Amsterdam

[20] IEEE: Standard Computer Dictionary, 1990. A Compilation of IEEE Standard Computer Glossaries. In *NY. 610-1990*. ISBN: 1559370793.

[21] International Organization for Standardization, 1999. ISO 14528: Industrial Automation Systems – Concepts and rules for Enterprise Models, *TC 184/SC5/WG1*, Geneva, Switzerland.

[22] International Organization for Standardization, 2002. ISO 16100: Manufacturing Software Capability Profiling for interoperability. In *Part 1: Framework, TC 184/SC5/WG4*, Geneva, Switzerland.

[23] LaOngsri, S. 2009, Semantic Extensions and a Novel Approach to Conceptual Modelling, Ph.D. Thesis, School of Computer Science, Engineering and Mathematics, The Flinders University of South Australia

[24] Lezoche M., Panetto H., Aubry A., 2011, Conceptualisation approach for Cooperative Information Systems interoperability. 13th International Conference on Enterprise Information Systems ICEIS 101-110, Beijing, P. R. China, DOI: 10.5220/0003508401010110.

[25] Mani, M.: EReX, 2004. A Conceptual Model for XML. In *Proceedings of the Second International XML Database Symposium* (XSym 2004), 128-142. Toronto, Canada.

[26] Manola, F., Miller, E., 2004. RDF Primer. In *World Wide Web Consortium*, Recommendation REC-rdf-primer-20040210.

[27] Meersman, R., Tari, Z., 2003. Creating a "DOGMAtic" Multilingual Ontology Infrastructure to Support a Semantic Portal On The Move to Meaningful Internet Systems 2003. In OTM 2003 Workshops OTM Confederated International Workshops, Lecture Notes in Computer Science, Vol. 2889. ISBN: 978-3-540-20494-7, Catania, Italy.

[28] Nijssen, G.M. & Halpin, T.A. 1989, Conceptual Schema and Relational Database Design, Prentice Hall, Sydney.

[29] Obrst, L., 2003 Ontologies for semantically interoperable systems. In *Proceedings of the 12th International Conference on Information and Knowledge Management*. New Orleans, USA

[30] OMG, 2003. Object Management Group. Architecture-Driven Modernization specification http://adm.omg.org

[31] OMG, 2004. Object Management Group. UML 2.0 Superstructure Specification http://uml.omg.org

[32] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen W., 1991. Object Oriented Modeling and Design. *Prentice Hall*, Book Distribution Center, New York, USA.

[33] Russel, S., Norvig, P. *Artificial Intelligence, A Modern Approach"*, Prentice-Hall. Inc. 1995

[34] Seeley, R. S., 1997. Manufacturing execution systems in *MED DEVICE DIAGN IND.* Vol. 19, no. 11, pp. 64-68.

[35] Sheth, A., 1998. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. In M. Goodchild, M. Egenhofer, R. Fegeas, and C. Kottman, editors. In *Interoperating Geographic Information Systems*, pp. 5– 30. Kluwer.

[36] Smith, M.K., Welty, Ch., McGuinness, D.L., 2004. OWL Web Ontology Language Guide. In *World Wide Web Consortium*, Recommendation REC-owl-guide-20040210.

[37] Tolk, A., Diallo, S. Y., Turnitsa, C. D., 2007. Applying the Levels of Conceptual Interoperability Modelling Support of Integrability, Interoperability, and Composability for System-of-Systems Engineering. In *Journal of Systemics, Cybernetics and Informatics*, Volume 5 Number 5, pp. 65-74.

[38] Tursi, A., Panetto, H., Morel, G., Dassisti, M., 2009 Ontological approach for Products-Centric Information System Interoperability in Networked Manufacturing Enterprises. In *IFAC Annual Reviews in Control*. 33/2, 238-245, Elsevier, ISSN: 1367-5788.

[39] Vyvyan, E., 2006. Lexical Concepts, Cognitive Models and Meaning-Construction. In *Cognitive Linguistics* 17 (4): 491-534.

[40] Yahia E., Lezoche M., Aubry A., Panetto H. 2011. Semantics enactment in Enterprise Information Systems. 18th IFAC World Congress. Milan, Italy. IFAC PapersOnline.

[41] Yahia E., 2011. *Contribution à l'Evaluation de l'Interopérabilité Sémantique entre Systèmes d'Information d'Entreprises : Application aux Systèmes d'Information de Pilotage de la Production*. PhD Thesis, Université Henri Poincaré, Nancy I (in French).

11 Appendix C

Semantics enactment in Enterprise Information Systems

Esma Yahia, Mario Lezoche, Alexis Aubry, Hervé Panetto

Research Centre for Automatic Control (CRAN), Nancy-University, CNRS, Campus Scientifique, Faculté des Sciences et Techniques, BP 70239, 54506 Vandoeuvre-lès-Nancy Cedex, France
(e-mail: {Esma.Yahia, Mario.Lezoche, Alexis.Aubry, Herve.Panetto}@cran.uhp-nancy.fr)

Abstract: The grown complexity of the modern enterprise poses a series of challenges, among them keeping competitiveness in the fast changing environment in which the enterprise evolves. Addressing Enterprise Integration is considered as a key to achieve the goal of any enterprise either it is a single or a networked enterprise. Enterprise Modelling is a prerequisite to enable the common understanding of the enterprises and its various interactions in order to "provide the right information, at the right time, at the right place". However, problems often emerge from a lack of understanding of the semantics of the elaborated models resulting from various modelling experience based on different methods and tools. In this paper, we describe the challenges associated to semantics enactment in Information Systems models. To facilitate this enactment, we propose an approach based on a fact-oriented modelling perspective. Then, we also provide an algorithm to automatically build semantic aggregates that help in highlighting Enterprise Models core embedded semantics. A case study on the field of B2M interoperability is performed in order to illustrate the application of the presented approach.

Keywords: Enterprise Models, Information Systems, Semantics Enactment

1. INTRODUCTION

When evolving in a competitive global market, enterprises are forced to become increasingly agile and flexible in order to manage the fast changing business conditions. Today's challenges mainly concern Enterprise Integration (EI). Indeed, EI *deals with removing organisational barriers and/or improving interactions among people, systems, applications, departments, and companies (in terms of material, informational, decision and workflows)* (Vernadat, 2009)

Enterprise Modelling (EM) plays a critical role in this integration, enabling the capture of all the information and knowledge relevant for the enterprise operations and organisation (Vernadat, 1996; Panetto and Molina, 2004)

The produced Enterprise Models are mainly related to artefacts such as processes, behaviours, activities, information, resources, objects/material flows, goals, systems infrastructure and architectures... Those Enterprise Models must contain the necessary and sufficient semantics in order to be intelligible and then enabling the global Enterprise Integration.

For instance, if we consider the process model, its business semantics is mainly brought along by languages such as the Business Process Modelling Notation (BPMN[4]). Moreover, enriching this semantics is still an open issue; we can for example quote those researches made by (Boudjlida and Panetto, 2008) in terms of process models annotations.

Among all Enterprise Models, Information Systems (IS) models are considered as the core models of the enterprise. Concretely, the complexity of EI relies on the fact that an enterprise (a single or a networked enterprise) comprises numerous and heterogeneous Information Systems either at the business or manufacturing level such as ERP (Enterprise Resource Planning), MES (Manufacturing Execution System), SCM (Supply Chain Management), PDM (Product Data Management) and CRM (Customer Relationship Management). Those *IS*s need i) to share specified information and ii) to operate on that information according

[4] http://www.bpmn.org

to a shared operational semantics iii) in order to realise a specified purpose in a given context. Achieving these actions is commonly called *interoperation* (Whitman et al., 2006).

While studying an Information System (*IS*) model, we observe that its semantics is tacit as it is scrambled due to the implementation requirements.

The intent of this paper is to define a method for semantics enactment in IS. This allows bringing out the tacit semantics in order to get explicit semantics required when studying and using Enterprise Models.

In section 2, we present a modelling approach called fact-oriented modelling that allows releasing all the entities within the ISs conceptual models.

A recursive approach is thus proposed, in section 3, to analyse the detailed semantics of those ISs conceptual model. This approach starts by representing the basic concepts and ends by building semantic aggregates (so-called semantic blocks) according to predefined rules.

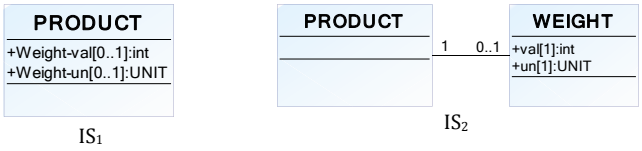
In order to illustrate the proposed approach, a case study is presented in the section 4. This case study deals with B2M (Business to Manufacturing) interoperability requirements between an Enterprise Resource Planning (ERP) system and a Manufacturing Execution System (MES) applications and consists in applying our approach in order to extract the semantics embedded into those ISs.

Finally, we conclude this paper with some remarks and perspectives for ongoing research.

2. FACT-ORIENTED PERSPECTIVE FOR SEMANTICS MODELLING

2.1 Fact-oriented modelling

The difficulty of operating with the various Enterprise Models comes out from the fact that the majority of those models have been made by different experts with several modelling experiences. That has led, for instance, to various conceptual representations for the same semantics. Since the majority of conceptual models have been fulfilled a posteriori and not a priori, implementation-based functionalities and constraints can cause interferences in the semantics understanding of those models. Let us consider, for instance, the extract of two different conceptual models in figure 1. Intuitively, those classes carry the same semantics, but are modelled differently. For instance, the *WEIGHT* of a *PRODUCT* on the right side of the figure is represented by a class due to an implementation constraint; when other classes are related to it, this facilitates querying for specific values related to the weight for example. While, on the left side of the figure, the *WEIGHT* of a *PRODUCT* is modelled by two attributes (its value and its unit).



IS1 IS2

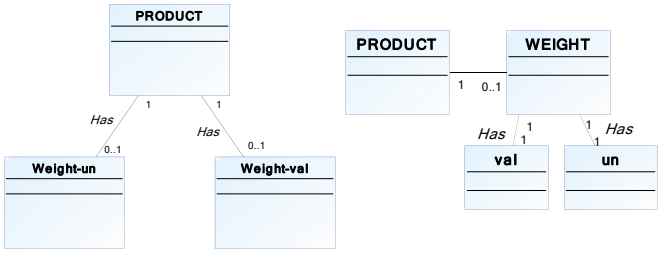
Fig. 1. Two extracts of conceptual models.

Fact-oriented modelling is a conceptual, natural language based approach avoiding such conflicting conceptual representations. It queries the information semantics of business domains in terms of the underlying facts of interest, where all facts and rules may be verbalised in a language readily understandable by users of those business domains (Halpin, 2007). Fact-oriented models are attribute-free, treating all elementary facts as relationships.

Object-Role Modelling (ORM) is the most popular fact-oriented approach. In fact, ORM makes no explicit use of attributes; instead it pictures the world in terms of lexical and non-lexical concepts that play roles (take part in relationships) (Halpin, 1998). This leads to a greater semantics stability and populatability, as well as facilitates natural verbalisation (Halpin, 2007).

In our work, we could use ORM as a modelling language. However, the existing conceptual models, in industrial context, are mainly represented with the UML notation. Hence getting a spread out of an attribute-free conceptualisation could be made using the UML notation but based on the ORM approach. Taking into account the ORM definitions, we will use the UML class diagram notation and we then call the UML concepts and the UML attributes as respectively non-lexical concepts and lexical concepts.

When applying the fact-oriented modelling on the examples of the figure 1, we obtain the following models (figure 2) that eases the semantics enactment.



IS1 IS2

Fig. 2. The conceptual models of Fig. 1 using the fact-oriented modelling perspective.

2.2 Core and extended knowledge

When considering an available fact-oriented conceptual model from one IS, we can distinguish the mandatory and non-mandatory "relationships", which represent mandatory and non-mandatory concepts expressing semantics.

In fact, the mandatory concepts contain all the necessary and sufficient elements to make the IS conceptual model semantically coherent and understandable. It comprises all the non-lexical and lexical concepts linked to constraint association roles with a multiplicity of 1 or 1..*. On the contrary, the non-mandatory concepts correspond to the non-mandatory roles (constraints 0..1 or *) and are only enriching the semantics of those IS conceptual models.

Somehow, the mandatory knowledge corresponds to the minimal semantics that should be contained in a given IS conceptual model. It could eventually represent the essence of the IS that is the core knowledge of the conceptual model. The extended knowledge includes the core and the non-mandatory knowledge.

Let us note that we consider that these models have made correct and that the implicit constraints are all represented explicitly in those models, that means that any constraint implemented into the software by developers have been reported in the models, themselves through roles multiplicities.

2.3 Some mathematical definitions

We define, for each *IS* conceptual model, the following notations.

Definition 1. A_{IS} is the set of the identified attributes or lexical concepts, formally defined by $A_{IS} = \{a_i | a_i$ is an attribute from the IS conceptual model$\}$

Definition 2. C_{IS} is the set of the identified concepts or non-lexical concepts, formally defined by

$$C_{IS} = \{c_i | c_i \text{ is a concept from the IS conceptual model}\}$$

Definition 3. Rel_{IS} is the set of the identified binary relationships between concepts such as hierarchy relationship and also between concepts and their related attributes. Formally, it is defined by

$$\text{Rel}_{\text{IS}} = \begin{Bmatrix} \{\text{rel}_a(c_j, a_i) | (c_j, a_i) \in C_{IS} \times A_{IS} \wedge c_j \text{ is related to } a_i\} \\ \cup \{\text{rel}_c(c_j, c_{j'}) | (c_j, c_{j'}) \in C_{IS}^2 \wedge c_j \text{ is related to } c_{j'}\} \end{Bmatrix}$$

Definition 4. Multiplicity is defined as Multiplicity = {*, 0..1, 1, 1..*} and serves to count the minimum and maximum number of instances when linking two given entities from the IS conceptual model. For each $(e_i, e_j) \in \{(C_{IS} \times A_{IS}), (C_{IS}^2)\}$, we have $Mult(e_i, e_j) \in \{*, 0..1, 1, 1..*\}$ and it is read e_i is related to e_j with a multiplicity $\in \{*, 0..1, 1, 1..*\}$.

If we consider a concept defined in the context of the IS core knowledge, we notice that in order to be ***semantically effective*** in the studied domain, this concept needs to be related on the one hand to its mandatory attributes and on the other hand to other concepts. This defines the notion of ***Semantic Block***.

3. SEMANTIC BLOCK IDENTIFICATION

3.1 Definition of a semantic block

Considering a particular concept c from C_{IS}, a semantic block, denoted as $B(c)$ and associated with the concept c, represents the minimal set of non-lexical concepts necessary for the minimal semantics definition of the concept c given by the conceptual model.

Let us consider the conceptual model on figure 3.

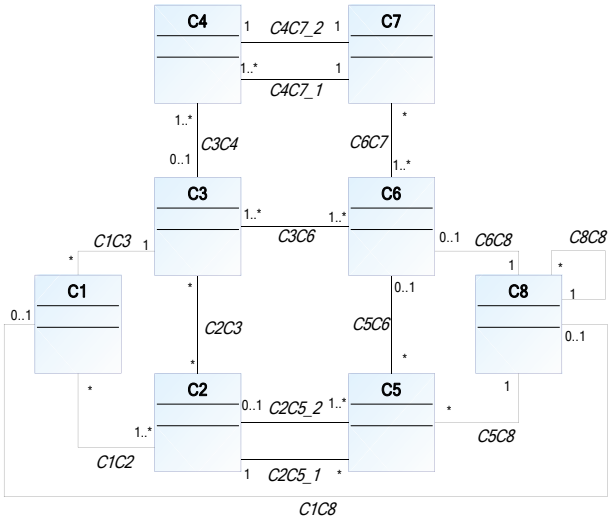


Fig. 3. A conceptual model

A given instance of the concept $C1$ exists only if it is associated with exactly one instance of the concept $C3$ and at least one instance of the concept $C2$. That means that $C2$ and $C3$ are mandatory for expressing the semantics of $C1$. Moreover an instance of $C3$ exists only if it is associated with at least one instance of $C4$ and at least one instance of $C6$. On the contrary, as the

minimal multiplicity is 0 for role $C5$ when considering the association between $C6$ and $C5$, the existence of any instance of $C6$ is not conditioned by the existence of one instance of $C5$.

Finally, continuing the same reasoning step by step, we can demonstrate that all the concepts are mandatory for expressing the semantics of $C1$. That means that $B(C1) = \{C1, C2, C3, C4, C5, C6, C7, C8\}$. The semantic block of a concept c finally contains all the concepts that must be instantiated for ensuring the existence of one instance of c.

3.2 A semantic-relationships graph

To facilitate the building of the semantic blocks, we propose to use graph theory modelling.

Let us define a semantic-relationships graph associated with a conceptual model. This semantic-relationships graph is a digraph $G = (V, E)$ where V is the set of nodes and E is the set of edges defined by a pair of nodes. Each node from V represents a non-lexical concept of the conceptual model. Each edge from E is built from the conceptual model as follows: the edge (c_i, c_j) exists if (i) there is an association between c_i and c_j in the conceptual model, and (ii) if the minimal multiplicity for the role c_j, considering the existing association between c_i and c_j, is equal to 1. That means that the existence of the edge (c_i, c_j) represents the fact that c_j is mandatory for expressing the semantics of c_i.

For each $e \in E$, we define the function $I: E \to V$ that gives the initial node of the edge e and we define the function $T: E \to V$ that gives the terminal node of the edge e.

The figure 4 shows the semantic-relationships graph associated with the conceptual model of the figure 3.

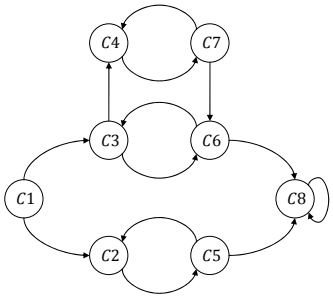


Fig. 4. Semantic-relationships graph associated with the conceptual model of figure 3.

3.3 Some Properties

Theorem 1. Given two particular concepts c_i and c_j, c_j belongs to $B(c_i)$ if and only if there exists a directed path from c_i to c_j.

Proof. Let us consider the conceptual model on figure 3. To build the semantic block of the concept c_i, we consider this concept as the starting point. This concept can thus be considered as the root in the associated semantic-relationships graph. Now we add in $B(c_i)$ all the concepts c_k that must be instantiated to ensure the existence of a particular instance of c_i, i.e. all the concepts c_{1k} such that there is an association between c_i and c_{1k} in the conceptual model, and the minimal multiplicity for c_{1k}, considering this association, is equal to 1. This is the exact definition of all the successors of c_i in the semantic-relationships graph. Note that, by definition, there is a directed path from the concept c_i to these concepts c_{1k}. Iteratively, the only new concepts c_{2k} that can be added to $B(c_i)$ are the successors of those first concepts c_{1k}. As successors of the concepts c_{1k}, there exists also a directed path from the concept c_i to the concepts c_{2k} (the path from c_i to c_{1k} plus the edge (c_{1k}, c_{2k})). Finally the semantic block of c_i contains exactly all the concepts c_j such that there exists a directed path from c_i to c_j. ■

Theorem 2. Given two particular concepts c_1 and c_2, if c_2 belongs to $B(c_1)$ then $B(c_2)$ is included in $B(c_1)$.

Proof. c_2 belongs to $B(c_1)$ means that there exists a path from c_1 to c_2 (see theorem 1). Let us now consider a particular concept from $B(c_2)$ denoted as c. By definition of $B(c_2)$, there exists a path from c_2 to c and then a path from c_1 to c (the path from c_1 to c_2 plus the path from c_2 to c). That means that c is in $B(c_1)$. Finally $B(c_2) \subseteq B(c_1)$. ■

Theorem 3. All the concepts that are in the same cycle in the semantic-relationships graph are associated with the same unique semantic block.

Proof. A cycle is a closed path. Let us consider two particular concepts, denoted as c_i and c_j, which belong to a cycle. In particular there is a

path from c_i to c_j. That means that c_j is in $B(c_i)$. Following the theorem 2, we can also demonstrate that $B(c_j) \subseteq B(c_i)$. Moreover, there is a path from c_j to c_i. That means that c_i is in $B(c_j)$. Following the theorem 2, that means that $B(c_j) \supseteq B(c_i)$. Finally, $B(c_j) = B(c_i)$. ■

For each cycle of the semantic-relationships graph, the theorem 3 implies that there is one shared semantic block associated with all the concepts that are in the same cycle, i.e. a strongly connected component of the semantic-relationships graph. Thus there is one semantic block per strongly connected component of the semantic-relationships graph.

3.4 Building the semantic blocks

Applying the theorems 1 to 3, we propose the following procedure to build all the semantic blocks of a given conceptual model:

Building the associated semantic-relationships graph, based on this associated semantic-relationships graph, building the graph of the strongly connected components,

And finally, building the semantic block associated with each strongly connected component.

3.4.1 Building the associated semantic-relationships graph

Following theorem 1, the semantic block of a concept c contains all the concepts c' such that it exists a directed path from c to c' in the associated semantic-relationships graph. This graph can be easily obtained by considering each association between two concepts c_i and c_j and then building an edge from c_i to c_j if the minimal multiplicity for c_j is equal to 1.

3.4.2 Building the graph of the strongly connected components

Theorem 3 implies that for building the semantic blocks, we can consider only one concept in a given strongly connected component (the other concepts have the same semantic block), that is the reason why we can simplify the semantic-relationships graph by considering only a graph where the nodes are the strongly connected components of the semantic-relationships graph and where an edge from one strongly connected component $SCC1$ to a second strongly connected component $SCC2$ exists if there exists at least one edge from a concept from $SCC1$ to a concept from $SCC2$.

Identifying all the strongly connected components of a graph is an easy problem that can be solved with polynomial effort by using Kosaraju-Sharir's algorithm (Sharir, 1981).

The graph of the strongly connected components associated with the semantic-relationships graph of figure 4 is given on figure 5. On this graph, the strongly connected components are defined as follows $SCC1 = \{C1\}$, $SCC2 = \{C2, C5\}$, $SCC3 = \{C3, C4, C6, C7\}$ and $SCC4 = \{C8\}$.

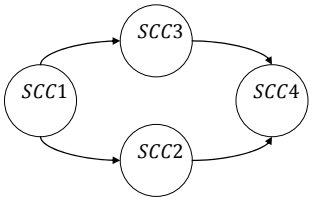


Fig. 5. Graph of the strongly connected components associated with the Semantic-relationships graph of figure 4.

3.4.3 Building the semantic block associated with each strongly connected component

We propose now a set of 2 algorithms to build all the semantic blocks associated with each strongly connected component (see Algo 1 and 2). The algorithm *BuildSemBlocks* is applied on the graph of the strongly connected components (denoted as G_{SCC}).

Let us apply the algorithm $BuildSemBlocks(G_{SCC})$ on the graph of figure 4. We obtain the following semantic blocks: $B(SCC1) = SCC1 \cup SCC2 \cup SCC3 \cup SCC4$,

$B(SCC2) = SCC2 \cup SCC4$,

$B(SCC3) = SCC3 \cup SCC4$ and

$B(SCC4) = SCC4$.

And finally replacing the strongly connected components by their content we obtain the following semantic blocks:

$B(C1) = \{C1, C2, C3, C4, C5, C6, C7, C8\}$,

$B(C2, C5) = \{C2, C5, C8\}$,

$B(C3, C4, C6, C7) = \{C3, C4, C6, C7, C8\}$ and

$B(C8) = \{C8\}$.

Algorithm $BuildSemBlocks(G_{SCC})$

[Initialisation]
L: List of the strongly connected components in G_{SCC}
For each $SCC \in L$ **Do**
  $color(SCC) = -1$
  [color is an indicator that defines if a node SCC has already been visited or not]
  [-1 means not yet visited]
  [0 means being visited]
  [+1 means already visited]
Next SCC
For each $SCC \in L$ **Do**
  If $color(SCC) = -1$ **Then**
    [Building of the semantic block associated with SCC]
    $BuildSB(SCC)$
  EndIf
Next SCC
Return

Algo. 1. BuildSemBlocks algorithm

Algorithm $BuildSB(SCC)$

[Initialisation]
$B(SCC) = SCC$ *[The semantic block associated with SCC initially contains all the concepts in the SCC]*
$color(SCC) = 0$ *[SCC is being visited]*

[Building]
[use of theorem 1]
For each SCC' successor from SCC in G_{SCC} **Do**
  If $color(SCC') = -1$ **Then**
    [Building of the semantic block associated with SCC']
    $BuildSB(SCC')$
  EndIf
  [Use of theorem 2]
  $B(SCC) = B(SCC) \cup B(SCC')$
Next SCC' successor from SCC in G_{SCC}

Return *B*(*SCC*)

Algo. 2. BuildSB algorithm

3.4.4 The semantic block architecture Meta-model

In this section, we propose to formalise the semantic block architecture by the meta-model represented on the figure 6. This meta-model uses the pattern composite (Gamma et al., 1995). The intent of this pattern is to compose the elements of the model into tree structures to represent part-whole hierarchies. In fact, each concept defined in a given context details its semantics and can be afterwards subsumed by a Semantic Block. Besides, the semantics of a set of aggregated concepts could be defined by a Semantic Block. Thus the Semantic Block properties emerge from the fact that those concepts are related to each others. Moreover, a Semantic Block could contain other blocks; this means that a given semantics could be subsumed by low level semantics. The "Block System" represents the last level of the Semantic Block aggregation, it could be interpreted as the semantics of the IS itself.

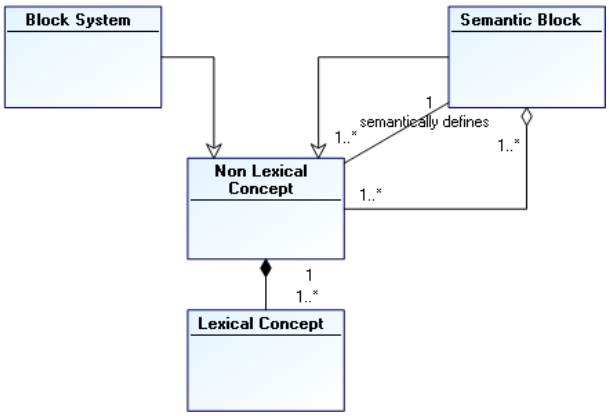


Fig. 6. The semantic block architecture meta-model

3.4.5 The automatic elaboration of semantic blocks

In this part, we present the tool prototype that automatically computes the semantic blocks for a given IS conceptual model. In fact the procedure presented in section 3.4 has been implemented into the MEGA EA Suite. *The MEGA EA Suite provides repository-based modelling tools to support projects ranging from capability mapping, to process analysis and application analysis and design. Covering all layers and phases of Enterprise Architecture, all modules are integrated*

into a consistent end to end solution[5]. Moreover the MEGA EA Suite supports UML notations and allows building our own meta-model based on its ad-hoc meta-model. The meta-model presented on figure 6 has been implemented into the MEGA EA Suite. In this implementation, the semantic block is conceptualised as an UML package and the lexical and non-lexical concepts are conceptualised as UML classes.

The procedure presented in section 3.4 has been implemented taking advantage of MEGA programming facilities.

Figure 7 shows an extract of the result after computing the implemented algorithm on the conceptual model presented in figure 3. This figure shows the semantic block $B(C2, C5)$ denoted as SB(2) (represented as an UML package) associated with $C2$ and $C5$, and including the concept $C8$. Figure 8 shows the sub-model formalising the semantics of $B(C2, C5)$. This sub-model is the extract of the complete conceptual model given on figure 3 by conserving only the concepts contained in $B(C2, C5)$ and the associations concerning these concepts.

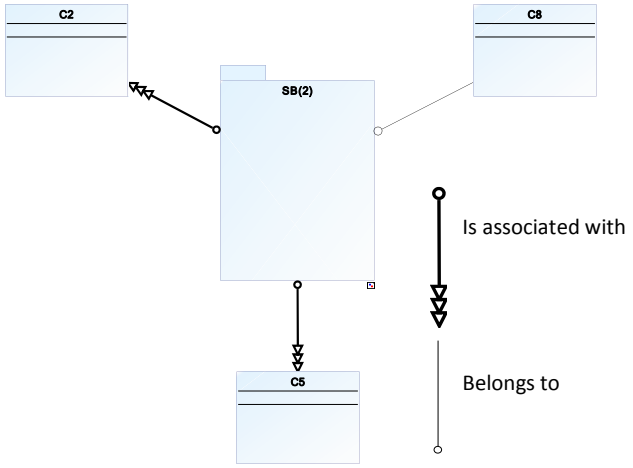


Fig. 7. The semantic block $B(C2, C5)$ for the conceptual model of figure 3

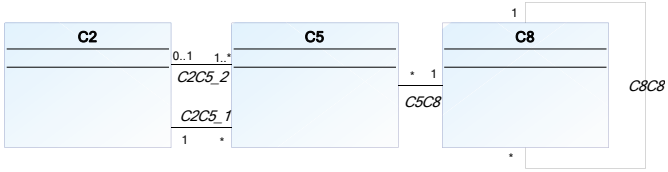


Fig. 8. The conceptual model of SB(2) = $B(C2, C5)$

[5] http://www.mega.com/uk/p/product/p2/enterprise-architecture

4. CASE STUDY: RAW MATERIAL PURCHASE

In order to illustrate the proposed approach of ISs semantics enactment, we choose the following case study that consists of two ISs dealing with B2M interoperability requirement. These ISs have been provided by a local technical centre: the AIPL-PRIMECA [6] (Atelier Inter-établissements de Productique Lorrain) in which the ERP Sage X3 application is cooperating with the MES Flexnet application in order to insure the manufacturing of a certain family of products. In such industrial large scale Enterprise Information Systems, applications comprise a multitude of tables and relations. Flexnet (a MES application) has around 800 tables with 300 relations, once we conceptualise its model, we get about 600 concepts and 500 associations. SAGE X3 has around 1600 tables with 900 relations, and when it is conceptualised, 1200 concepts and 1000 associations can be highlighted.

Actually, a specified process has been chosen to support our research; it consists of the Raw Material Purchase. For instance, Figure 9 represents the conceptual model for the purchase order process related to Flexnet.

When considering the long term planning, the ERP computes, for a given period, its needs in term of raw materials and then launches some purchase orders. Hence, those purchase orders have to be exported from the ERP to the MES that have to bring backward the ERP with the stock state and the purchase order status.

Once we apply the fact-oriented modelling with the UML notation, the tool generates the normalised conceptual models of Flexnet on figure 10 and the conceptual model of Sage X3 on the figure 11.

In order to extract the semantics from MES and ERP conceptual models, we compute the implemented algorithm for Flexnet and Sage X3 conceptual model.

Table 1 and 2 lists the different semantic blocks related to respectively Sage X3 and Flexnet applications, for purchase order process.

[6] AIPL-PRIMECA, www.aip-primeca.net/lorraine/

Figure 12 shows all the semantic blocks related to the Flexnet Purchase order.

Figure 13 shows the conceptual model associated to the semantic block $B(PRODUCT)$, and including all the mandatory concepts required to obtain the full semantics for the concept $PRODUCT$.

Table 7. Sage X3 semantic blocks

Semantic Block	Concepts
Block system 1= B^1(Purchase order) = B^1(Purchase order quantity)= B^1(PurchaseRequestDetail) = B^1(PurchaseRequest)=	Purchase order, Purchase order quantity, PurchaseRequestDetail, PurchaseRequest, Supplier, BusinessPartner, Facility, Units, Product, ProductFacility, Command number, Command Date, Command Line, Command Type, Stock Unit, Total Included Taxes, QuantityOrdred, PurchaseRequestQuantity, PurchaseRequestLine, Customer, RequestNo, RequestDate
B^1(Supplier)	Supplier, BusinessPartner, CorporateName, SupplierDescription, Tiers, Interfacility
B^1 (Facility)	Facility, Adress, SIRETNumber, FacitityType, Country, GeoCode, FacilityID, NAFcode
B^1(Units)	Units, UnitDescription, Unit, Symbol
B^1(Product) = B^1(ProductFacility)	Product, ProductFacility, Supplier, BusinessPartner, Facility, Units, ProductNo, ProductDescription, Article Code, CreationDate

Table 2. Flexnet semantic blocks

Semantic Block	***Concepts***
Block system 2	All the concepts
B^2 (WAREHOUSE)	WAREHOUSE, WarehouseID, FACILITY, FacilityID, Division
B^2(ORDER_PARTNER)	ORDER_PARTNER, PartnerOrderNo, PartnerOrderType, PARTNER, PartnerID
B^2(PARTNER_ADDRESS)	PARTNER_ADDRESS, AdressID, PARTNER, PartnerID
B^2(PARTNER)	PARTNER, PartnerID
B^2(WIP_ORDER, ORDER_DETAIL, ORDER_HEADER, WIP_ORDER_TYPE)	WIP_ORDER, WipOrderNo, CreatedOn, OrderQuantity, WIP_ORDER_TYPE, WipOrderType, ORDER_DETAIL, OrderLineNo, CreatedOn, ORDER_HEADER, OrderDate, OrderNo, WIP_ORDER_STATUS, WipOrderStatus, PROCESS, ProcessId, ProcessDescription, FUID, FACILITY, FacilityId, Division, PRODUCT, LotTrackingCode, ProductId, ProductNo, RevisionControlFlag, SerialTrackingCode, UOM, UOMCode, ORDER_STATUS,

	OrderStatus
B^2 (PROCESS)	PROCESS, ProcessId, ProcessDescription, FUID
B^2 (PRODUCT)	PRODUCT, LotTrackingCode, ProductId, ProductNo, RevisionControlFlag, SerialTrackingCode, UOM, UOMCode, FACILITY, FacilityId, Division,
B^2 (UOM)	UOM, UOMCode
B^2(WIP_ORDER_STATUS)	WIP_ORDER_STATUS , WipOrderStatus
B^2 (FACILITY)	FACILITY, FacilityId, Division,
B^2(ORDER_STATUS)	ORDER_STATUS, OrderStatus

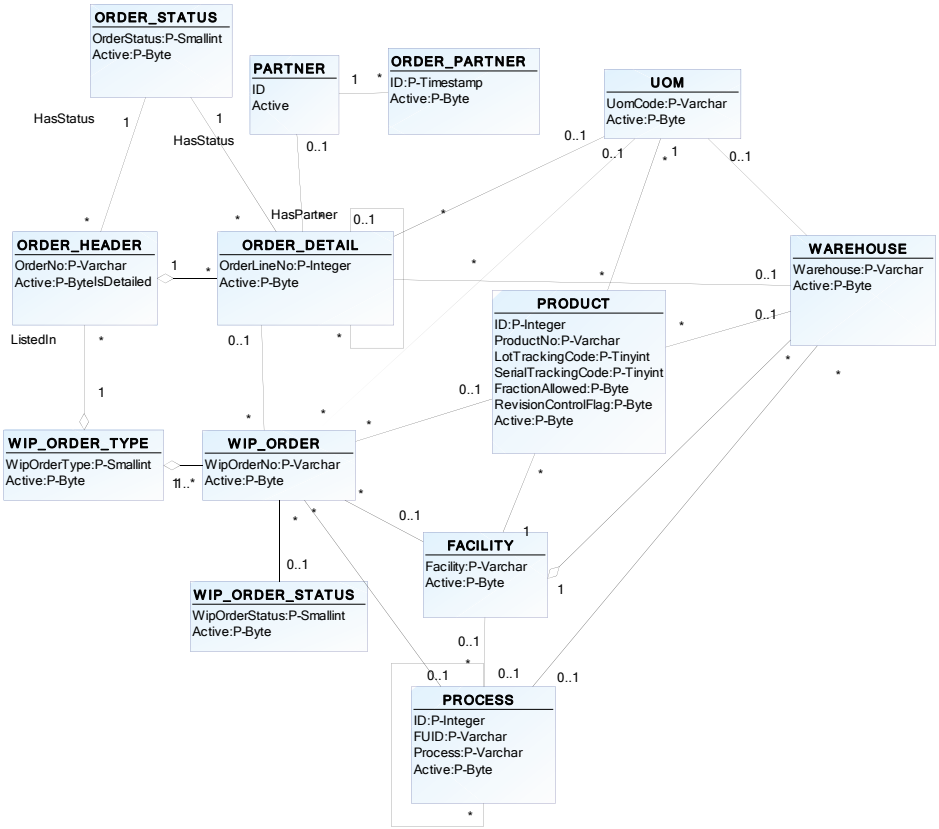


Fig. 9**.** Conceptual model for the purchase order process from Flexnet

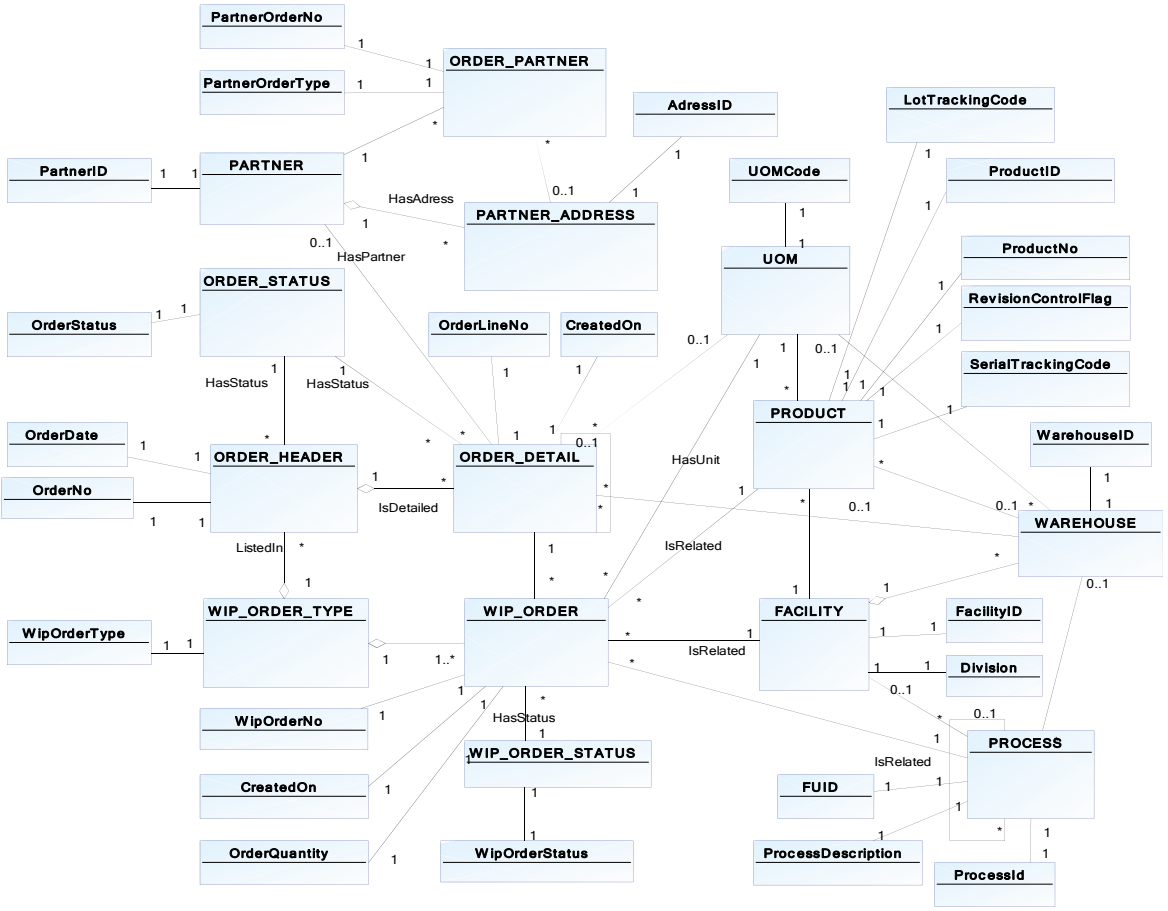


Fig. 10. The conceptual model of a purchase order in Flexnet application: fact-oriented model

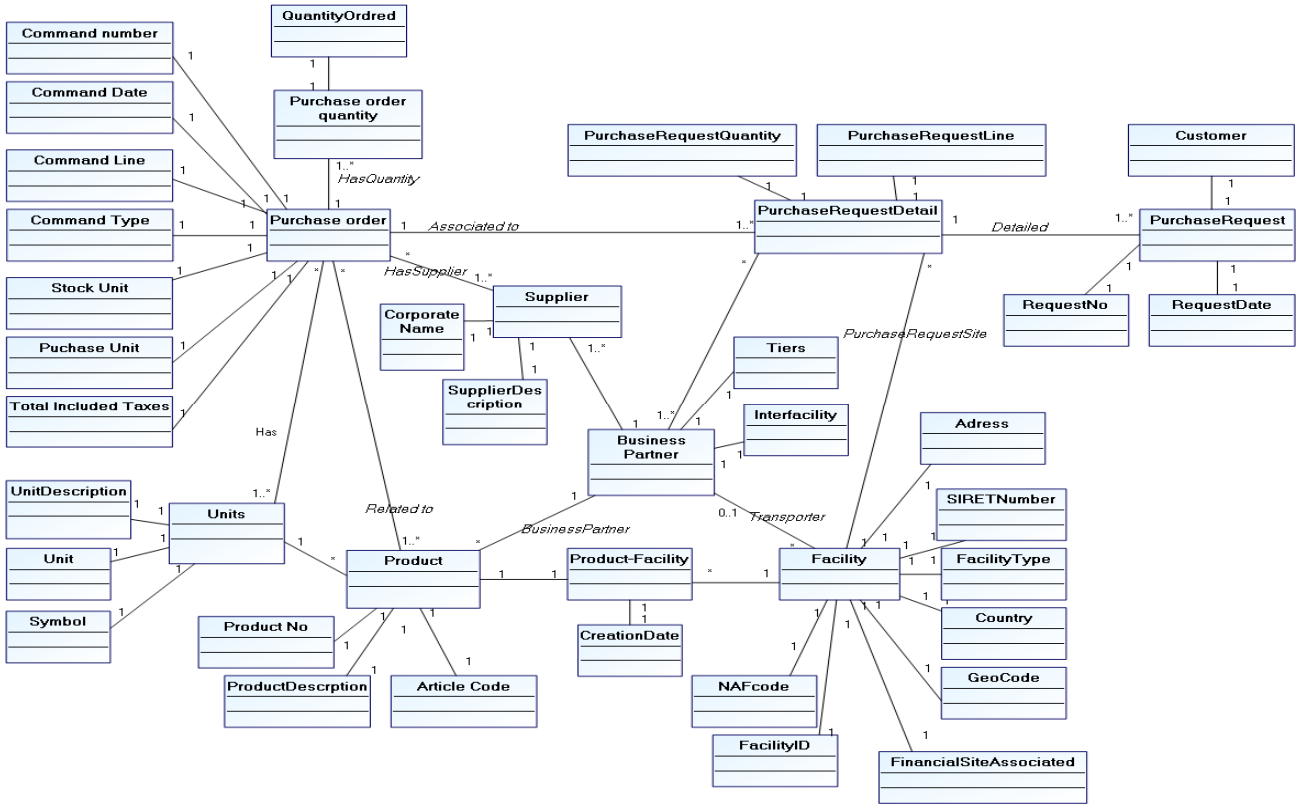


Fig. 11. The conceptual model of a purchase order in SAGE X3 application: fact-oriented model

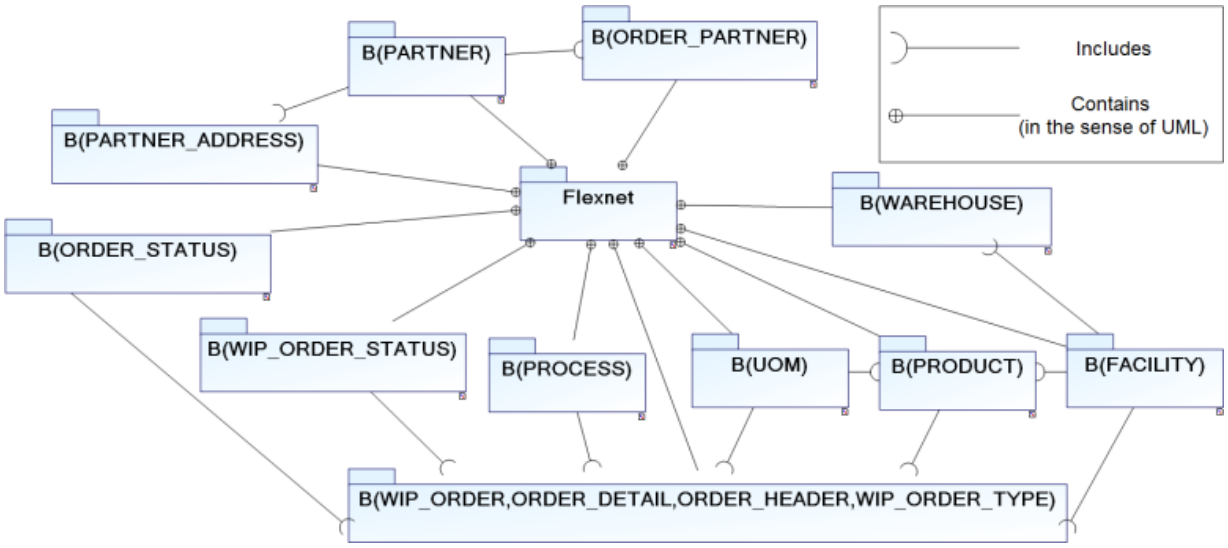


Fig. 12. The computed semantic blocks related to the Flexnet Purchase order

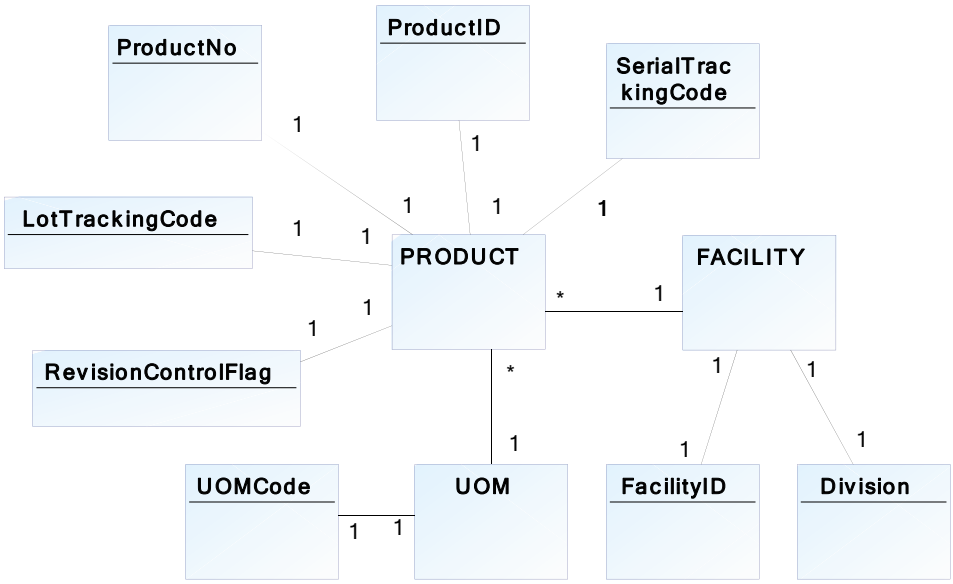


Fig. 13. The conceptual model associated to the semantic block B(PRODUCT)

6. CONCLUSION

Semantics enactment among *IS*s conceptual models is a critical issue in the context of Enterprise Models. Indeed, extracting these semantics has the advantage to ease the understanding and then the use of the exchanged information among heterogeneous information systems (In single or distributed Enterprises)

We proposed in this paper the fact-oriented modelling to get a spread out representation for ISs conceptual models. This has allowed us to identify the Core and the extended Knowledge for a given *IS*, respectively composed by the mandatory and non mandatory concepts.

The originality of this paper lies on the elaboration of the semantic blocs for enacting Enterprise Models semantics embedded and, often hidden, in complex Information Systems models. Moreover, each semantic block identifies and emphasises the border of one sub-system model with its own core semantics. It focuses on "what is important" in the system without taking care on implementation artefacts.

We illustrate the semantics blocks identification in a use case based on existing B2M applications: the ERP Sage X3 and the MES Flexnet enterprise software applications, which have to interoperate in order to achieve a global process performance.

Future work aims at using the semantic blocks formalisation in order to facilitate models matching and concepts mapping when formalise and evaluate the interoperability process between enterprise applications in a virtual networked enterprises environment.

REFERENCES

Boudjlida N., and Panetto, H. (2008). Annotation of enterprise models for interoperability purposes. In *Proceedings of the IWAISE 2008*.

Gamma E., Helm R., Johnson R., and Vlissides J. (1995). *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley, Reading, Springer, Heidelberg.

Halpin T. (1998). Object-Role Modeling (ORM/NIAM), Handbook on Architectures of Information.

Halpin, T. (2007). Fact-oriented modeling: Past, present and future. In J. Krogstie, A. Opdahl & S. Brinkkemper (eds), *Conceptual Modelling in Information Systems Engineering,* pages 19–38. Berlin: Springer-Verlag,

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

Sharir, M. (1981). A Strong-Connectivity Algorithm and its Applications in Data Flow Analysis. *Computers and Mathematics with Applications*, 7, 67-72.

Vernadat, F.B. (2009). Enterprise Integration and Interoperability. In *Springer Handbook of Automation*, pages 1529-1538.

Vernadat, F.B. (1996). *Enterprise Modelling and Integration: principles and applications*. Chapman and Hall, ISBN: 0 412 60550 3

Whitman, L., Santanu, D. and Panetto, H. (2006). An Enterprise Model of Interoperability. In: Proceeding of the 12th IFAC Symposium on Information Control Problems, in Manufacturing (INCOM'2006). Elsevier, Saint Etienne, France

12 Appendix D

Why, Where and How to use Semantic Annotation for Systems Interoperability

Yongxin Liao[1], Mario Lezoche[1, 2], Hervé Panetto[1], Nacer Boudjlida[2]
[1]*Research Centre for Automatic Control (CRAN), Nancy-University, CNRS, BP 70239, 54506 Vandoeuvre-lès-Nancy Cedex, France*
{Yongxin.Liao, Mario.Lezoche, Herve.Panetto }@cran.uhp-nancy.fr
[2]*LORIA, Nancy-University, CNRS, BP 70239, 54506 Vandoeuvre-lès-Nancy Cedex,France, Nacer.Boudjlida@loria.fr*

Abstract: Semantic annotation is one of the useful solutions to enrich target's (systems, models, meta-models, etc.) information. There are some papers which use semantic enrichment for different purposes (integration, composition, sharing and reuse, etc.) in several domains, but none of them provides a complete process of how to use semantic annotations. This paper identifies three main components of semantic annotation, gives a formal definition of semantic annotation method and presents a survey of current semantic annotation methods which include: languages and tools that can be used to develop ontology, the design of semantic annotation structure models and the corresponding applications. The survey presented in this paper will be the basis of our future research on models, semantics and architecture for systems interoperability.

Keywords: Semantic Annotation, Models, Ontology, Systems Interoperability.

1. INTRODUCTION

Nowadays, the need of systems collaboration across enterprises and through different domains has become more and more ubiquitous. But because the lack of standardized models or schemas, as well as semantic differences and inconsistencies problems, a series of research for data/model exchange, transformation, discovery and reuse are carried out in recent years. One of the main challenges in these researches is to overcome the gap among different data/model structures. Semantic annotation is not only just used for enriching the data/model's information, but also it can be one of the useful solutions for helping semi-automatic or even automatic systems interoperability.

Semantically annotating data/models can help to bridge the different knowledge representations. It can be used to discover matching between models elements, which helps information systems integration (Agt, et al., 2010). It can semantically enhance XML-Schemas' information, which supports XML documents transformation (Köpke and Eder, 2010). It can describe web services in a semantic network, which is used for further discovery and composition (Talantikite, et al., 2009). It can support system modellers in reusing process models, detecting cross-process relations, facilitating change management and knowledge transfer (Bron, et al., 2007). Semantic annotation can be widely used in many fields. It can link specific resources according to its domain ontologies.

The main contribution of this paper is identifying three main components of semantic annotation, gives a formal definition of semantic annotation and presenting a survey, based on the literature, of current semantic annotation methods that are applied for different purposes and domains. These annotation methods vary in their ontology (languages, tools and design), models and corresponding applications.

The remaining of this paper is organized as follows: Section 2 describes the definition of annotation and gives a formal definition of semantic annotations. Section 3 provides the answers to *why* and *where* to use semantic annotation. Section 4 first presents an introduction to ontologies and semantic annoation structure models, and then discusses the usage of semantic annotations. Section 5 concludes this paper, together with some related work and potential extensions.

2. WHAT IS SEMANTIC ANNOTATION?

In this section, we first illustrate the types of annotations from different papers (section 2.1), and then propose a formal definition of semantic annotation together with its three main components (section 2.2).

2.1 Definition and Types of annotation

In Oxford Dictionary Online[7], the word "annotation" is defined as "*a note by way of explanation or comment added to a text or diagram*". It is used to enrich target object's information, which can be in the forms of text descriptions, underlines, highlights, images, links, etc. Annotation has special meanings and usages in different fields. In software programming, an annotation is represented as text comments embedded in the code to expand the program, which is being ignored when the program is running. In mechanical drawing, an annotation is a snippet of text or symbols with specific meanings. In Library Management, an annotation is written in a set form (numbers, letters, etc.), which helps the classification of books.

Further, different annotation types are identified by the following papers: Bechhofer, et al. (2002) and Boudjlida, et al. (2006) distinguished annotation as (i) *Textual annotation*: adding notes and comments to objects; (ii) *Link annotation*: linking objects to a readable content; (iii) *Semantic annotation*: that consists of semantic information which is machine-readable. Similarly, three types of annotation are described in the research of Oren, et al. (2006): (i) *Informal annotation*: notes that are not machine-readable; (ii) *Formal annotation*: notes that are formally defined and machine-readable (but it does not use ontology terms); (iii) *Ontological annotation*: notes that use only formally defined ontological terms that are commonly accepted and understood.

Bechhofer, et al. (2002) further classified the annotation according to six possible uses that are not always clear and disjoint: (a) *Decoration*, comments on an object; (b) *Linking*, link anchors; (c) *Instances Identification*, strong assert that an object is an instance of a particular class. It may use a URI; (d) *Instance Reference*, less clear than instance identification, reference depending on background and world knowledge; (e) *Aboutness*, loose association of the object with a concept; (f) *Pertinence*, assertions about the concepts within an ontology without encoding that information.

[7]http://oxforddictionaries.com

According to the above classification, semantic annotation can be considered as a kind of formal metadata, which is machine and human readable. This will be further discussed in the following sections.

2.2 Semantic annotation

The term "Semantic Annotation" is described as "*the action and results of describing (part of) an electronic resource by means of metadata whose meaning is formally specified in an ontology*" (electronic resource can be text contents, images, video, services, etc.) by Fernández (2010). Talantikite, et al. (2009) introduced it as "*An annotation assigns to an entity, which is in the text, a link to its semantic description. A semantic annotation is referent to an ontology*". In the research of Lin (2008), semantic annotation is concerned as "*an approach to link ontologies to the original information sources*". All above definitions from different papers show one thing in common: a semantic annotation is the process of linking electronic resource to a specific ontology. Ontology here is only one of the possible means to provide a formal semantic.

As it can be seen on Figure 1, the left side represents an Electronic Resource (ER) and on the right side, there are the three main components of semantic annotation: (1) *Ontology*, which defines the terms used to describe and represent a body of knowledge (Boyce, et al., 2007). It can be reused from existing ontologies or designed according to different requirements. (2) *Semantic Annotation Structure Model (SASM)*, which organizes the structure/schema of an annotation and describes the mappings between electronic resources and an ontology. (3) *Application*, which is designed to achieve the user's purposes (composition, sharing and reuse, integration, etc.) by using SASM. This figure also shows the three main steps on how to use semantic annotation, which is introduced in section 4: ontology (section 4.1), semantic annotation structure model (section 4.2) and application (section 4.3).

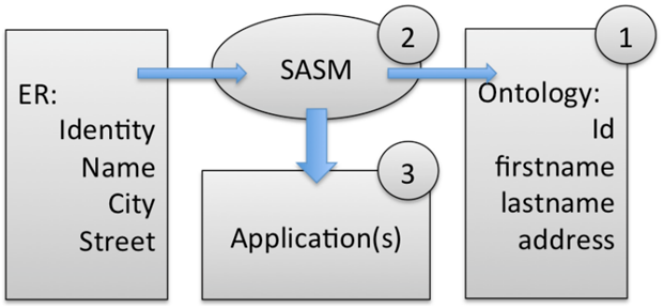


*Fig. 1. S*emantic Annotation components

The following definition formally defines a semantic annotation: a *S*emantic Annotation *SA*is a tuple $(\mathcal{M}, \mathcal{A})$ consisting of the SASM $\mathcal{M}$and an application $\mathcal{A}$.

$$SA := \{\mathcal{M}(\mathcal{E}, \mathcal{P}(\mathcal{O})), \mathcal{A}\}$$

Where:
$\mathcal{O} = \{o_1, o_2, \ldots, o_n\}$, is the set of ontology o_i that bring some meaning to any annotated element.

An *Ontology* $o_j \epsilon\, \mathcal{O}$ is a 4-tuple (C_{o_j}, is_a, R_{o_j}, σ_{o_j}), where C_{o_j} is a set of *concepts*, *is_a* is a partial order relation on C_{o_j}, $R_{o_j}$is a set of relation names, andσ_{o_j}: $R_{o_j} \rightarrow (C^+)$is a function which defines each relation name with its arity (Stumme and Maedche, 2001a).

Formally, $\mathcal{M} = \{m_x: \langle e_i, p_j \rangle |\ e_i \epsilon\, \mathcal{E} \times p_j \epsilon\, \mathcal{P}(\mathcal{O})\}$ and represents the set of relationships between an element e_i of the set of electronic resources $\mathcal{E}$ and an element p_j of the powerset of the ontology set $\mathcal{O}$.

A mapping $m_x(e_i, p_j)$ may represent three different kinds of semantic relations:

(4) $m_{\sim}(e_i, p_j)$ is a binary *equivalence* relation. If $m_{\sim}(e_i, p_j)$ then an electronic resource e_i is semantically equivalent to p_j, an element of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

(5) $m_{\supset}(e_i, p_j)$ is a binary relation stating that the semantic of an electronic resource e_i*subsumes* the semantic of an element p_j of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

(6) $m_{\subset}(e_i, p_j)$: is a binary relation stating that the semantic of an electronic resource e_i*is subsumed* by the semantic of an element p_j of the powerset $\mathcal{P}(\mathcal{O})$, in the context of an application $\mathcal{A}$.

$\mathcal{M}$ can be further extended, including also some additional parameters or constraints c_k, generally expressed using, in the worst case, natural language, or, better, a formal logical expression. $\mathcal{M}$ is then defined as $\mathcal{M} := \{m_x, c_k\}$.

The main issue, related to mappings such as in (2) and in (3), is being able to measure the semantic gap (2) or the over semantic (3), brought by the semantic annotation. Such measures have been studied by researchers in the domain of information retrieval (Ellis, 1996) or in the domain of ontology matching (Maedche and Staab, 2002), mapping (Doan et al, 2002), merging (Stumme and Maedche, 2001b), alignment (Noy and Musen, 2000).

In addition, Peng, et al. (2004) also gave a very simple definition of semantic annotation in their paper, which is $SA := (R, O)$, where R is set of resources and O is an ontology. Furthermore, Luong and Dieng-Kuntz (2007) defined it as $SA := \{R_A, C_A, P_A, L, T_A\}$. In this definition, R_A is a set of resources; C_A is a set of concept names; P_A is a set of property names; L is a set of literal values; and T_A is a set of triple (s, p, v), where $s \in R_A, p \in P_A, v \in (R_A \cup L)$. To the best of our knowledge, T_A in this definition is duplicated.

3. WHY, WHERE TO USE SEMANTIC ANNOTATION

A semantic annotation uses ontology objects for enriching resource's information that tells a computer the meanings and relations of the data terms. It can be used in many areas, such as Business Process Models, Web services, XML Schema, Strategic Data Models, Information Systems, etc. Several usages of semantic annotation are introduced:

Business Process Models: In the research of Lin (2008), a semantic annotation framework is designed to manage the semantic heterogeneity of process model, to solve the discovery and sharing of process models problems in/between enterprise(s). Born, et al. (2007) used semantic description of process artefacts to help a modeller in graphical modelling of business processes.

Web Services: Talantikite, et al. (2009) used a semantic annotation to represent web services as a semantic network. Based on the network and submitted requests, the composition algorithm produces the best composition plan. Patil, et al. (2004) proposed an annotation framework for semi-automatically marking up web service descriptions (WSDL files) with domain ontologies to help web services discovery and composition.

XML Schema: In the research of Köpke and Eder (2010), a path expression method is used to add annotation to XML-Schemas. Then they transform paths to ontology concepts and use them to create XML-Schema mappings that help XML document transformation.

Strategic Data Models: Diamantini and Potena (2008) presented a novel model that uses a mathematical ontology in semantic annotation to describe mathematical formulas in Data Warehouse schemas.

Information System: Agt, et al. (2010) used semantic annotations to help information system integration. They annotate the model/object at CIM (Computation Independent Model), PIM (Platform Independent Model) and PSM (Platform Specific Model) levels of the MDA approach (Mellor, et al. 2002; 2004), and then they discover some matching between model elements with respect to semantic process requirements.

In short, semantic annotation can be considered as a semantically enrichment of models or data, which may be widely used for many purposes. In business process models and Information system, it can be used to bridge the gap between two models. In Web service and Strategic Date Models, it can be used as additional information that helps description, discovery and composition. To the best of our knowledge, the path expression method in XML Schema will lead to lose information in Schema (e.g. restrictions of max-occur/min-occur, sequence or choice of elements, etc.), which still needs to be improved.

4. HOW TO USE SEMANTIC ANNOTATION

In this section, we present an introduction to the three main components of semantic annotation: the languages and tools which can be used in designing ontology; semantic annotation model's structure and mappings; and the applications of semantic annotation.

4.1 Introduction to Ontology

Designing an appropriate ontology for semantic annotations is the first step of the annotation process. Ontology has been actively studied for a long period of time, and there are many research works proposing ontology

engineering techniques. We are not going to give, here, a complete overview of every ontology languages, but we provide a brief introduction to the three more representative languages. We will show also some simple examples and typical development tools.

Ontolingua was developed by KSL (Knowledge Systems Lab, Stanford University) (Fikes, et al, 1997). It is an extension of KIF[8] (Knowledge Interchange Format) through adding frame-based representation and translation functionalities. But because of the newly development of semantic web ontology, Ontolingua is not frequently used recently. Figure 2-a) shows a simple Ontolingua example from Mizoguchi (2003). Ontolingua Server[9] provides an editor, which can be used to browse, create, edit, modify, and use Ontolingua ontologies.

F-Logic was presented by Michael Kifer (Stony Brook University) and Georg Lausen (University of Mannheim) (Kifer and Lausen, 1995). It is an object-oriented language that is frequently used for Semantic Web. It also can map straightforward to most frequent ontological constructs. Figure 2-b) shows a simple F-logic example from Liao, et al. (2010). Flora2[10] is an F-logic ontology development application, which extends F-logic with HiLog and Transaction Logic.

OWL (Web Ontology Language) was developed by World Wide Web Consortium, which shares many characteristics with RDF[11] (Resource Description Framework) and RDF Schema (Horrocks, et al., 2003). It is written using the XML syntax, and contains three sublanguages: OWL Lite, OWL DL and OWL Full. OWL is considered as a standard language for ontology representation for semantic web. Figure 2-c) shows a simple OWL example from OWL Guide[12]. Protégé[13] ontology editor is a Java-based tool that can export ontology into formats such as OWL, RDF and XML Schema. *OntoStudio*[14]supports the modelling of RDF(S), OWL and Object-Logic with possible transformation between them.

[8]http://www-ksl.stanford.edu/knowledge-sharing/kif/
[9]http://www.ksl.stanford.edu/software/ontolingua/
[10]http://flora.sourceforge.net/
[11]http://www.w3.org/RDF/
[12]http://www.w3.org/TR/2004/REC-owl-guide-20040210/
[13]http://protege.stanford.edu/
[14]http://www.ontoprise.de/en/products/ontostudio/

a) Ontolingua

```
(define-class Tutoring-objective (?t-obj)
"Attributes are also represented as slots"
:def (and (individual ?t-obj)
    (value-type ?t-objTuroring.policy Policy))
:axiom-def (subclass-partition Tutoring-objective
    (setof Transfer-ofknowledge Remedy)))
```

b) F-logic

```
General class information:
    person[name*=>string, children*=>person].
Database facts:
    John:person[name->'John Doe', children-> {Bob, Mary}].
Mary:person[name->'Mary Doe', ciildren->{Alice}]
Deductive rule:
    ?X:human:- ?X:person
Query:
    ?X:person[name->?Y, children->Mary]
```

c) OWL

```
Class and Individuals:
    <owl:Classrdf:ID="Wine">
<rdfs:subClassOfrdf:resource="&food;PotableLiquid"/>
<rdfs:labelxml:lang="en">wine</rdfs:label>
<rdfs:labelxml:lang="fr">vin</rdfs:label>
</owl:Class>
Properties:
<owl:ObjectPropertyrdf:ID="hasWineDescriptor">
<rdfs:domainrdf:resource="#Wine" />
<rdfs:rangerdf:resource="#WineDescriptor" />
    </owl:ObjectProperty>
```

Fig. 2. Examples of Ontolingua (a), F-logic (b) and OWL (c).

The design methods of ontology for annotations have their own purposes and structures.

Lin (2008) used Protégé OWL editor to design the ontology. In order to separately annotate meta-models (modelling language) and their process models, the author designs two ontologies: *General Process Ontology (GPO)* and *Domain Ontology*. The design of GPO is based on Bunge-Wand-Weber (BWW) Ontology (Bunge 1977; Wand and Weber, 1993). GPO contains nine main concepts: *Activity*, *Artifact*, *Actor-role*, *Input*, *Output*, *Precondition*, *Postcondition*, *Exception* and *WorkflowPattern*. Relations between above concepts are *has_actor-role*, *has_artifact*, *has_subActiviy*, *has_input*, *has_output*, *related_to*, *has_precondition*, *has_postcondition*, *has_exception*, *handled_by* (e.g. *Activity* uses *has_actor-role* relation to link *Actor-role*). The Domain ontology is formalized according to SCOR[15] specifications (Supply Chain Operations Reference-model).

Agt, et al. (2010) designed a semantic meta-model (SMM) to describe domain ontologies. Artefacts in ontology are castigated as DomainFunction and DomainObject. The relations (predicates) among Objects and Functions are defined as: *IsA*, *IsInputOf*, *IsOutputOf*, *Has*, *IsListOf*, *IsEquivalentTo*, etc. A RDF-like triple (e.g., *Tax Has TaxNumber*) is used as the statement in SMM.

Born, et al. (2007) used two kinds of ontologies: sBPMN[16] ontology and a domain ontology. The first ontology is used to represent BPMN process models. The second ontology defines domain objects, states and actions according to objects lifecycle, which is used to provide the user advices during the modelling process. More details of above ontologies can be found in references.

4.2 Introduction to Semantic Annotation Structure Model

[15]http://supply-chain.org/
[16]http://www.ip-super.org

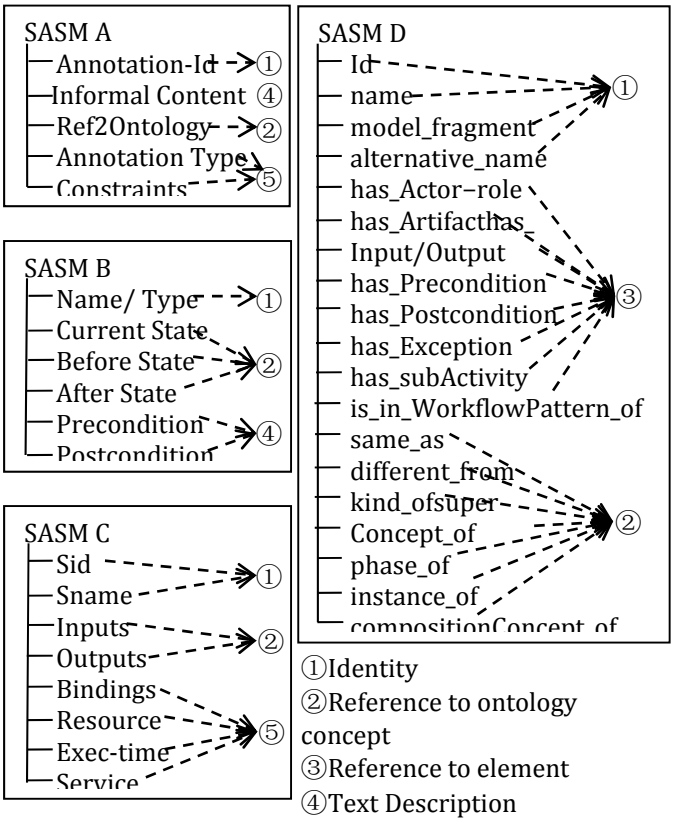
The second component of a semantic annotation is SASM. It is the connection between electronic resources and ontology concepts. A study in this direction is pursued by SAWSDL Working Group[17] that developed SAWSDL (Semantic Annotation for Web Services Definition Language) which provides two kinds of extension attributes as follow: (i) *modelReference*, to describe the association between a WSDL or XML Schema component and a semantic model concept; (ii) *liftingSchemaMapping* and *loweringSchema- Mapping*, to specify the mappings between semantic data and XML (Martin, et al., 2007; Kopecký, et al. 2007).

To be more specific, we analyse four SASMs that are designed for different requirements. Figure 3 below gives an overview of these four SASMs: Model A is the annotation schema for enterprise models from Boudjlida and Panetto (2007); Model B is designed to annotate the business process model from Born, et al. (2007); Model C is proposed to conceptually represent a web service from Talantikite, et al. (2009); and Model D is the annotation model for an activity element which is part of the Process Semantic Annotation Model (PSAM) from Lin (2008).

In order to compare above semantic annotation structure models, we identify five types for classifying the contents in SASM:

(1) *identity* of annotation (e.g. id, name, etc.);
(2) *reference to ontology concept* (e.g. element Customer has a reference "same_as" which is referenced to ontology concept Buyer);
(3) *reference to element* (represent the relationship between element themselves. e.g. element manufacture has a reference "has_input" which is referenced to element material);
(4) *text description*, the natural language definitions of annotation contents;
(5) *others* (extinction contents, such as: execution time, restriction, annotation types, etc.). The classification results of each SASM are described by linking model contents to type numbers

We can easily find that the basic components of SASMs are: *identity of annotation* and *reference to ontology concepts*; *reference to element*, *text description* and *others* are added for different usages. As an example, Lin (2008) adds "has_Actor−role" to denote the relationship between activity element and actor-role element; Boudjlida and Panetto (2007) added "Informal Content" for explaining the intent of the annotation; Talantikite, et al. (2009) added "exec-time" into SASM to record the execution time of a web service request. In the rest of this section, the discussion is focused on the design of *reference to ontology concepts*.



[17]http://www.w3.org/2002/ws/sawsdl/#Introduction

Fig. 3. Semantic Annotation Structure Model Examples.

As can be seen from above figure, *reference to ontology concepts* in model A is just a conceptual reference without meanings. Model B describes the references with meaning of states of objects (current, before and after). Model C uses inputs and outputs to represent the relationships. Model D gives more meanings to references like same_as, kind_of, phase_of, etc. Further, one to one mapping is not the only mapping type in SASM. For example, in Model C, there can be more than one input, which means the mapping between model content and ontology concept is one to many. Here, we analyses "*reference to ontology concepts*" according to mapping types and definitions of mappings.

Mappings are separated into two levels in the research of Lin (2008): meta-model level and model level. In the meta-model level, mapping direction is from ontology to model contents. The mappings are defined as: *Atomic Construct* (one to one. e.g. Activity is mapped to Task), *Enumerated Construct* (one to many. e.g. Artifact is mapped to Information or Material) and *Composed Construct* (one to combination. e.g. Workflow Pattern is mapped to a combination of Flow and Decision Point). In model level, semantic relationships are: *Synonym* (same_as, alternative_name), *Polysemy* (different_from), *Instance* (instance_of), *Hyponym* (superConcept_of), *Meronym* (part_of, member_of, phase_of and partial Effect_of), *Holonym* (composition Concept_of) *and Hypernym* (kind_of). (e.g. *Meronym*: *Airline* member_of *Air Alliance*). Agt, et al. (2010) described five mapping types in their work: *single representation* (one model element to one ontology concept), *containment* (one model element to multiple ontology concepts), *compositions* (multiple model elements to one ontology concepts), *multiple* and *alternative representation* (the mappings with AND and OR/XOR operators). Table 2 shows the comparison and classification of the mappings from Agt, et al. (2010) and Lin (2008). In order to classify those mappings, we assume the mapping direction in the table is from a model element to an ontology concept.

Table.1. Mappings from Model to Ontology

Types	Lin (2008)	Lin (2008)	Agt, et al.(2010)
1 to 1	Atomic Construct	Instance Synonym Polysemy Hyponym Hypernym	Single represent
1 to n			Containment, Multiple Alternative
n to 1	Enumerated Construct Composed Construct	Meronym Holonym	Composition

In our opinions, there are three high level mapping types: *1 to 1* mapping, *1 to n* mapping and *n to 1* mapping (n to n is a combination of 1 to n and n to 1). For each of the mapping, we can design different semantic relationships for further usages. Figure 4 shows the mapping types and semantic relationships for each kind of mapping. *1 to 1* means one element is annotated by one ontology concept. Semantic relationships can be: equal_to, similar_to, etc. *1 to n* means one element is annotated by the composition/aggregation of several ontology concepts. Semantic relationships can be: contains, has, etc. *n to 1* means the composition/aggregation of several elements are annotated by one ontology concept. Semantic relationships can be: part_of, member_of, etc. One element can have several semantic relationships, but for each relationship, they belong to one mapping type.

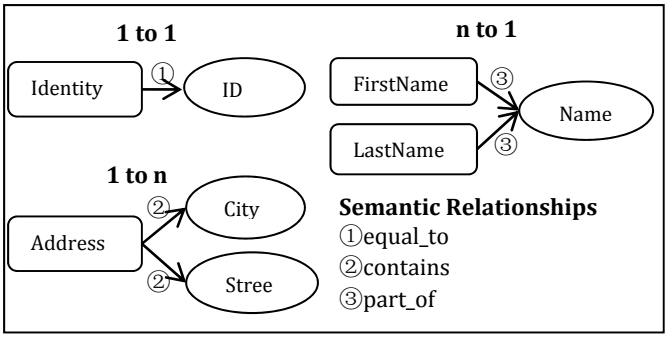


Fig. 4. Mapping types and semantic relationships

Since the structure and semantic relationships of SASM are designed, we should consider how to implement the annotation process. The annotation process can be performed manually, semi-automatically or automatically (Reeve and Han, 2005). In the research of Lin (2008), mapping is manually linking the process models to ontology. In the work of Patil, et al. (2004), mapping is semi-automatically computed. They developed algorithms to match and annotate WSDL files with relevant ontologies. Automatic mapping is, for the moment, restricted to some simple cases because of the impossibility to completely explicit knowledge from the different models.

4.3 Introduction to Application

Once the semantic annotation structure model is defined, designers can begin to design the application to achieve their purpose (composition, sharing and reuse, integration, etc.). Several applications of semantic annotation are introduced as follow:
Talantikite, et al. (2008) designed an application, which uses a matching algorithm to process the "input" and "output" (SASM model C, Figure 3) of elements, and builds a semantic network for web services. This semantic network is explored by a composition algorithm, which automatically finds a composite service to satisfy the request. Authors implement a prototype in java, which includes: Pellet[18]Reasoner (matching algorithm), RSsw (Réseau Sémantique des Services Web), Request and Composor (returns an optimal composite service for requesters).

Lin (2008) developed a prototype Process Semantic Annotation tool (Pro-SEAT) to describe the relationship between process models and ontologies. They use Metis[19] as a modelling environment integrating Protégé OWL API to provide the OWL ontology browser. Ontologies (GPO, Domain ontology, etc.) are stored on an ontology server, which can be loaded by annotators. The output of the annotation is an OWL instance file, which is used by a knowledge repository service to support the process knowledge query, discovery and navigation from users.

Born, et al. (2007) used Tensegrity Graph Framework[20] as environment to support graphical design functions. Name-base and Process Context-base matchmaking functionalities are designed to help user annotating process models. Name-base matching uses string distance metrics method for the matching between business process models and domain ontology, and it supports the user for specifying or refining the process. Process Context-base matching uses the lifecycle (state before, state after, etc.) in domain ontology for suggesting the next activity during modelling.

Indeed, there are many tools and technologies that enable designing applications in semantic annotation. The selections of tools are always depending on the design of semantic annotation structure models and ontologies. In any case, all three components of semantic annotation are closely related

5. CONCLUSIONS

In this paper, a brief survey of semantic annotation in different domains is presented. We identify three main components of semantic annotations that are Ontology, Semantic Annotation Structure Model and Application. In addition, a formal definition of semantic annotation is proposed. It contributes to better understand what a

[18]http://clarkparsia.com/pellet/
[19]http://www.troux.com/
[20]http://www.tensegrity-software.com/

semantic annotation is and then contributes to a common reference model. But how to use semantic annotation? There are still many problems can be further discussed during the annotation process. For example, how to optimize ontology and an annotated model? How to solve the inconsistency or conflicts during the mapping? How to add consistent semantic on models in different levels of a system? How to achieve semi-automatic or automatic annotation?

We are currently investigating how semantic annotations can help collaborative actors (organizations, design teams, system developers, etc.) in co-designing, sharing, exchanging, aligning and transforming models. In particular, this research work will be based on general systems with several kinds of interactions. We can have interoperation between systems that with different versions (during many years, systems may have been modified or updated). We can also have systems with same functions but used by different enterprises. Semantic annotations can bridge this knowledge gap and identify differences in models, in schemas, etc. In some case, interoperation is a process between a set of related systems throughout a product lifecycle (Marketing, Design, Manufacture, Service, etc.), and semantic annotations can influence the existing foundations and techniques which supports models reuse, semantic alignment and transformation, etc. Above all, our research work will focus on designing, and reusing appropriate ontologies in relationship with a formal semantic annotation structure model.

REFERENCES

Agt, H., Bauhoff, G., Kutsche, R., Milanovic, N., Widiker, J. (2010). Semantic Annotation and Conflict Analysis for Information System Integration. In: Proceedings of the 3rd Workshop on Model-Driven Tool & Process Integration. 7-18

Bechhofer, S., Carr, L., Goble, C., Kampa, S. and Miles-Board, T. (2002). The Semantics of Semantic Annotation. In: Proceedings of the 1st International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems. 1151-1167.

Born, M., Dorr, F., Weber, I. (2007). User-Friendly Semantic Annotation in Business Process Modeling. In: Proceedings of the 2007 international conference on Web information systems engineering.

Boudjlida, N., Dong, C., Baïna, S., Panetto, H., Krogstie, J., Hahn, A., Hausmann, K., Tomás, J.V., Poler, R., Abián, M.Á., Núñez, M.J., Zouggar, N., Diamantini, C., Tinella, S. (2006). A practical experiment on semantic enrichment of enterprise models in a homogeneous environment. INTEROP NoE Deliverable DTG4.1. INTEROP NoE IST 508011. http://www.interop-vlab.eu

Boudjlida, N., Panetto, H. (2007). Enterprise semantic modelling for interoperability. In: Proceedings of the 12th IEEE conference on emerging technologies and factory automation, Patras, Greece. 847–854.

Boyce, S., Pahl, C. (2007). Developing Domain Ontologies for Course Content. Educational Technology & Society 275-288.

Bunge, M. (1977): Treatise on Basic Philosophy (Vol 3): Ontology I: The Furniture of the World. D. Reidel Publishing Company, first edition

Diamantini, C., Potena, D. (2008). Semantic enrichment of strategic datacubes. In: Proceedings of the ACM 11th International Workshop on Data Warehousing and OLAP. 81-88.

Ding, G., Xu, N. (2010). Automatic semantic annotation of images based on web data. In: Proceedings of the 6th international conference of Information Assurance and Security. 317-322

Doan, A., Madhavan, J., Domingos, P., Halevy, A.Y. (2002). Learning to map between ontologies on the semantic web. In Proceedings of the World Wide Web conference. 662-673.

Ellis, D. (1996). The Dilemma of Measurement in Information Retrieval Research. Journal of the American Society for Information. Vol. 47, N° 1. 23-36.

Fernández, N. (2010). Semantic Annotation Introduction, [online] (Updated 14 Oct 2010) Available at <http://www.it.uc3m.es/labgimi/teoria/Module2/SA-Intro.pdf>

Fikes, R., Farquhar, A., Rice, J. (1997). Tools for Assembling Modular Ontologies in Ontolingua. In: Proceedings of the 14th national conference on artificial intelligence and 9th conference on Innovative applications of artificial intelligence. 436-441.

Horrocks, I., Patel-Schneider, P.F., Harmelen, F.V. (2003) From SHIQ and RDF to OWL: the making of a Web Ontology Language. Journal of Web Semantics. Vol 1, N° 1. 7-26.

Irfanullah, I., Aslam, N., Loo, J., Loomes, M., Roohullah, R. (2010). In: Proceedings of the IEEE international symposium on Signal Processing and Information Technology. 491-495

Lin, Y. (2008). Semantic Annotation for Process Models: Facilitating Process Knowledge Management via Semantic Interoperability. PhD thesis, Norwegian University of Science and Technology, Trondheim, Norway.

Liao, Y., Romain, D., J. Berre, A. (2010). Model-driven Rule-based Mediation in XML Data Exchange. In: Proceedings of the 1st International Workshop on Model-Driven Interoperability. 89-97

Luong, P., Dieng-Kuntz, R. (2007). A Rule-based Approach for Semantic Annotation Evolution. In Computational Intelligence. Vol. 23, Issue 3, 320–338

Kifer, M., Lausen, G., Wu, J. (1995). Logical Foundations of Object-Oriented and Frame-Based Languages. Journal of the ACM. Vol.42, N°4. 741-843.

Kopecký, J., Vitvar, T., Bournez, C., Farrell, J. (2007). SAWSDL: Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing. Vol.11, N° 6. 60-67

Köpke, J., Eder, J. (2010). Semantic Annotation of XML-Schema for Document Transformations. In: Proceedings of the OTM Workshops. 5th International Workshop on Enterprise Integration, Interoperability and Networking. Lecture Notes in Computer Science, Vol. 6428. 219-228.

Maedche, A., Staab, S. (2002). Measuring Similarity between Ontologies. In: Proceeding of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web. 251-263.

Mellor, S.J., Scott, K., Uhl, A., Weise, D. (2002). Model-Driven Architecture. In: Proceedings of the Workshop at the 8th International Conference on Object-Oriented Information Systems. 290-297.

Mellor S.J., Kendall S., Uhl A. and Weise D. (2004). Model Driven Architecture, Addison-Wesley Pub Co.

Martin, D., Paolucci M., Wagner, M. (2007). Towards Semantic Annotations of Web Services: OWL-S from the SAWSDL Perspective. In: Proceedings of the OWL-S Experiences and Future Developments Workshop at ESWC 2007.

Mizoguchi, R. (2003). Tutorial on Ontological Engineering: Part 2: Ontology Development, Tools and Languages. New Generation Comput.

Noy, N.F., Musen, M.A. (2000). PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: Proceedings of the 17th National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence. 450-455.

Oren, E., Hinnerk Möller, K., Scerri, S., Handschuh, S., Sintek, M.(2006). What are Semantic Annotations?. Technical report, DERI Galway

Patil, A., Oundhakar, S., Sheth, A., Verma, K. (2004). Meteor-S Web Service annotation framework. In: Proceedings of the 13th International Conference on the World Wide Web. 553-562.

Peng, W., Baowen, X., Jianjiang, L., Dazhou, K., Yanhui, L. (2004). A Novel Approach to Semantic Annotation Based on Multi-ontologies. In: Proceedings of the third International Conference on Machine Learning and Cybernetics. Vol. 3. 1452 - 1457

Reeve, L.H., Han, H. (2005). Survey of semantic annotation platforms. In: Proceedings of the ACM Symposium on Applied Computing. 1634-1638

Stumme, G., Maedche, A. (2001a). Ontology Merging for Federated Ontologies on the Semantic Web. In: Proceedings of the International Workshop for Foundations of Models for Information Integration.

Stumme, G., Maedche, A. (2001b). FCA-MERGE: Bottom-Up Merging of Ontologies. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence. Seattle, Washington, USA. 225-234.

Talantikite, H.N., Aïssani, D., Boudjlida, N. (2009). Semantic annotations for web services discovery and composition. Computer Standards & Interfaces. Vol. 31, N°6. 1108-1117.

Wand, Y., Weber, R. (1993). On the ontological expressiveness of information systems analysis and design grammars. Information System Journal Vol 3. N°4. 217-237.