

# 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<sup>1</sup>). 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 ISs need i) to share specified information and ii) to operate on that information according 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

<sup>1</sup> <http://www.bpmn.org>



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).

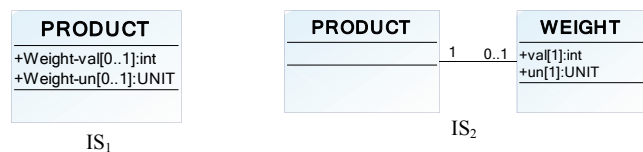


Fig. 1. Two extracts of conceptual models.

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.

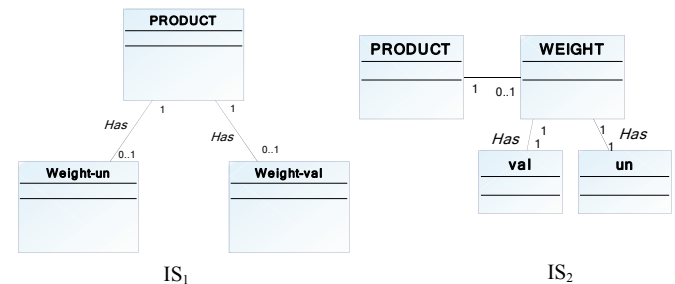


Fig. 2. The conceptual models of Fig. 1 using the fact-oriented modelling perspective.

### 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  $A_{IS} = \{a_i | a_i \text{ is an attribute from the IS conceptual model}\}$

**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

$$Rel_{IS} = \left\{ \begin{array}{l} \{rel_a(c_j, a_i) \mid (c_j, a_i) \in C_{IS} \times A_{IS} \wedge c_j \text{ is related to } a_i\} \\ \cup \{rel_c(c_j, c_j') \mid (c_j, c_j') \in C_{IS}^2 \wedge c_j \text{ is related to } c_j'\} \end{array} \right\}$$

**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.

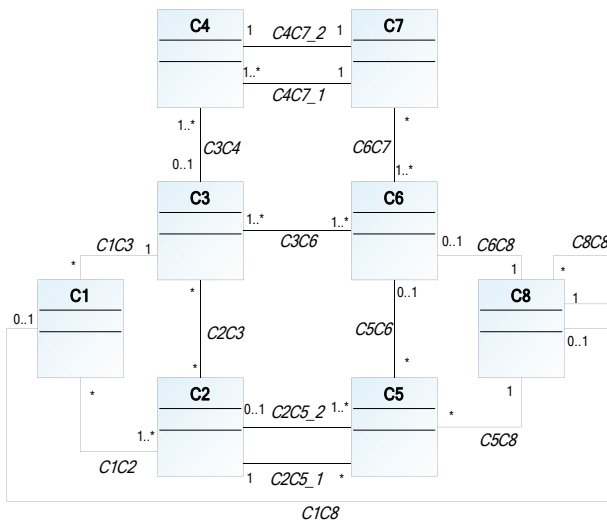


Fig. 3. A conceptual model

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  $I: E \rightarrow V$  that gives the initial node of the edge  $e$  and we define the function  $T: E \rightarrow V$  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.

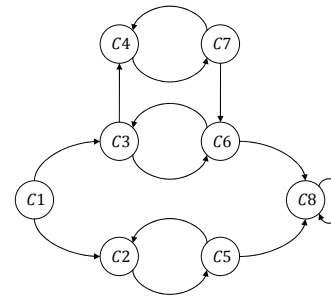


Fig. 4. Semantic-relationships graph associated with the conceptual model of figure 3.

#### 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  $SCC4 = \{C8\}$ .

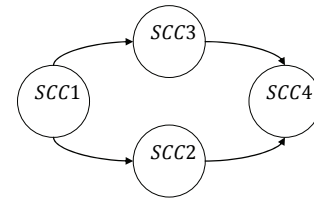


Fig. 5. Graph of the strongly connected components associated with the Semantic-relationships graph of figure 4.

#### 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:  $B(SCC1) = SCC1 \cup SCC2 \cup SCC3 \cup SCC4$ ,

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

$$B(SCC3) = SCC3 \cup SCC4 \text{ and}$$

$$B(SCC4) = SCC4.$$

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

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

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

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

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

---

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.

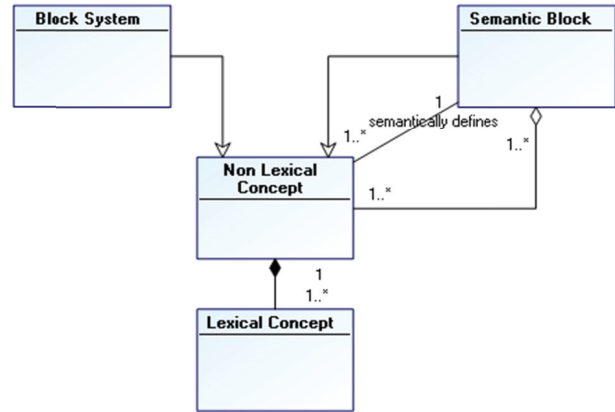


Fig. 6. The semantic block architecture meta-model

### 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<sup>2</sup>. 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

<sup>2</sup> <http://www.mega.com/uk/p/product/p2/enterprise-architecture>

conceptual model given on figure 3 by conserving only the concepts contained in  $B(C2, C5)$  and the associations concerning these concepts.

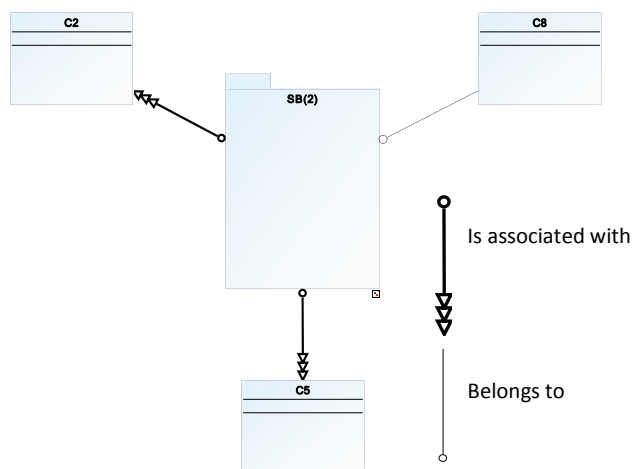


Fig. 7. The semantic block  $B(C2, C5)$  for the conceptual model of figure 3

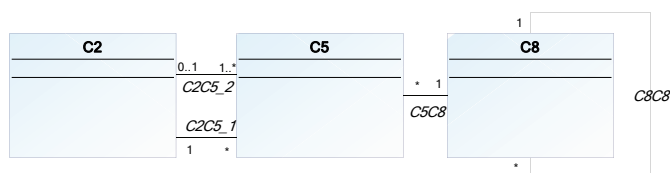


Fig. 8. The conceptual model of  $SB(2) = B(C2, C5)$

#### 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<sup>3</sup> (Atelier Inter-établissements de Productique Lorraine) 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.

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

Figure 13 shows the semantic block  $B(PRODUCT)$  denoted (represented as an UML package) associated with the concept  $PRODUCT$ , and including all the mandatory concepts required to obtain the full semantics for the concept  $PRODUCT$ .

Table 1. Sage X3 semantic blocks

Semantic Block	Concepts
Block system 1 = $B^1(\text{Purchase order}) = B^1(\text{Purchase order quantity}) = B^1(\text{PurchaseRequestDetail}) = B^1(\text{PurchaseRequest}) =$	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
$B^1(\text{Supplier})$	Supplier, BusinessPartner, CorporateName, SupplierDescription, Tiers, Interfacility
$B^1(\text{Facility})$	Facility, Adress, SIRETNumber, FacilityType, Country, GeoCode, FacilityID, NAFcode
$B^1(\text{Units})$	Units, UnitDescription, Unit, Symbol

<sup>3</sup> AIPL-PRIMECA, [www.aip-primeca.net/lorraine/](http://www.aip-primeca.net/lorraine/)

$B^1(\text{Product}) = B^1(\text{ProductFacility})$	Product, ProductFacility, Supplier, BusinessPartner, Facility, Units, ProductNo, ProductDescription, Article Code, CreationDate	$B^2(\text{PROCESS})$	PROCESS, ProcessId, ProcessDescription, FUID
		$B^2(\text{PRODUCT})$	PRODUCT, LotTrackingCode, ProductId, ProductNo, RevisionControlFlag, SerialTrackingCode, UOM, UOMCode, FACILITY, FacilityId, Division,

Table 2. Flexnet semantic blocks

<i>Semantic Block</i>	<i>Concepts</i>		
Block system 2	All the concepts	$B^2(\text{UOM})$	UOM, UOMCode
$B^2(\text{WAREHOUSE})$	WAREHOUSE, WarehouseID, FACILITY, FacilityID, Division	$B^2(\text{WIP\_ORDER\_STATUS})$	WIP_ORDER_STATUS, WipOrderStatus
$B^2(\text{ORDER\_PARTNER})$	ORDER_PARTNER, PartnerOrderNo, PartnerOrderType, PARTNER, PartnerID	$B^2(\text{FACILITY})$	FACILITY, FacilityId, Division,
$B^2(\text{PARTNER\_ADDRESS})$	PARTNER_ADDRESS, AdressID, PARTNER, PartnerID	$B^2(\text{ORDER\_STATUS})$	ORDER_STATUS, OrderStatus
$B^2(\text{PARTNER})$	PARTNER, PartnerID		
$B^2(\text{WIP\_ORDER, ORDER\_DETAIL, ORDER\_HEADER, WIP\_ORDER\_TYPE})$	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		

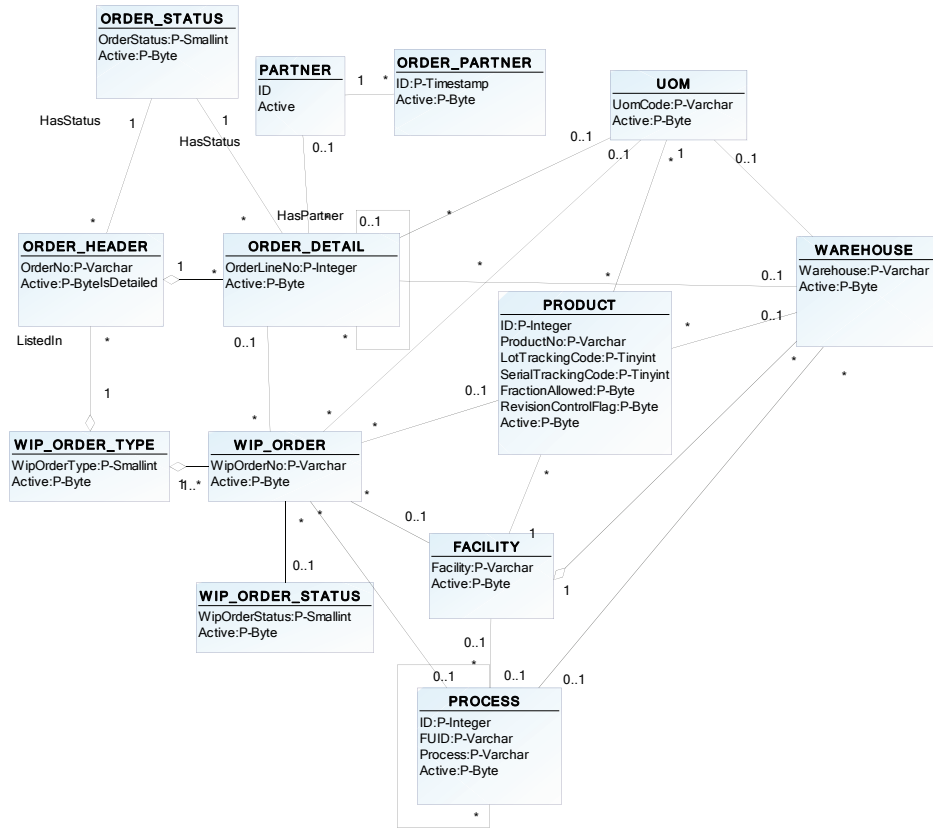


Fig. 9. Conceptual model for the purchase order process from Flexnet

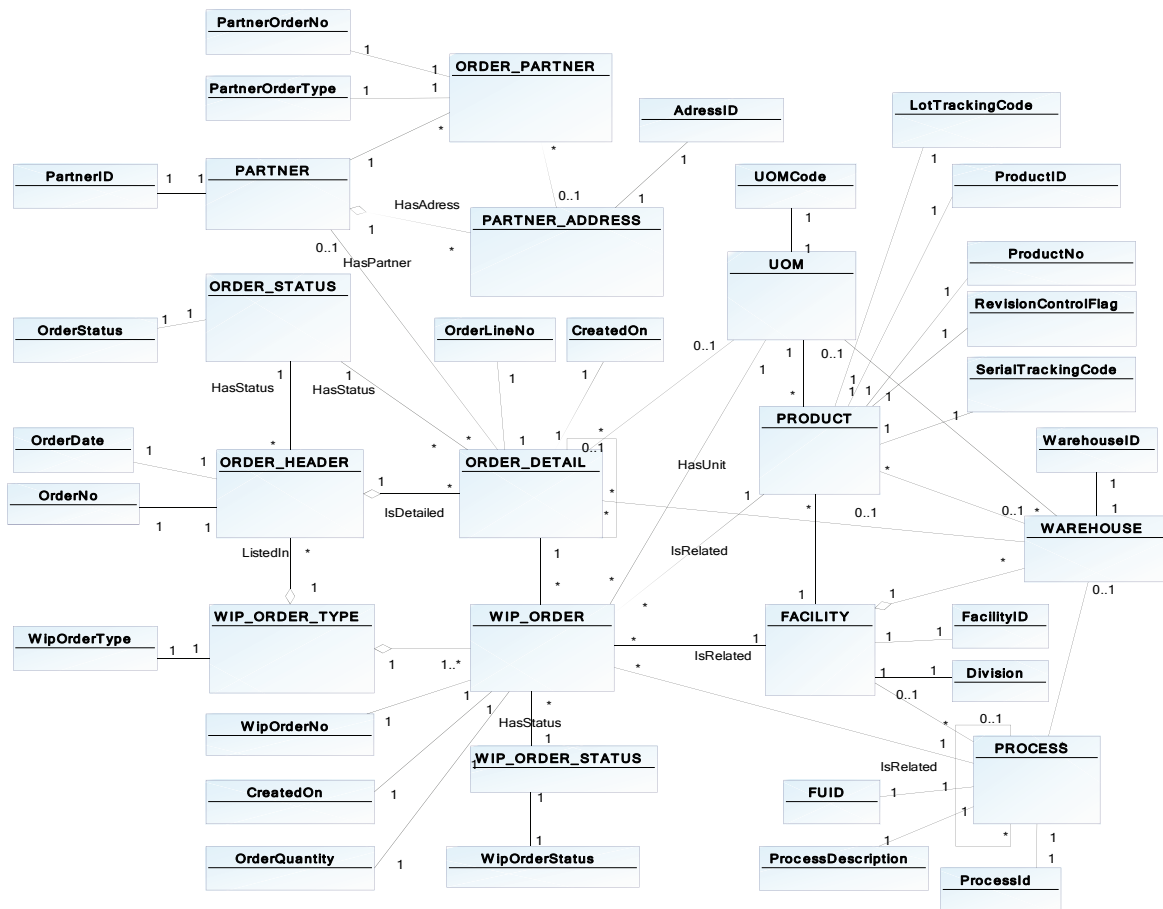


Fig. 10. The conceptual model of a purchase order in Flexnet application: fact-oriented model

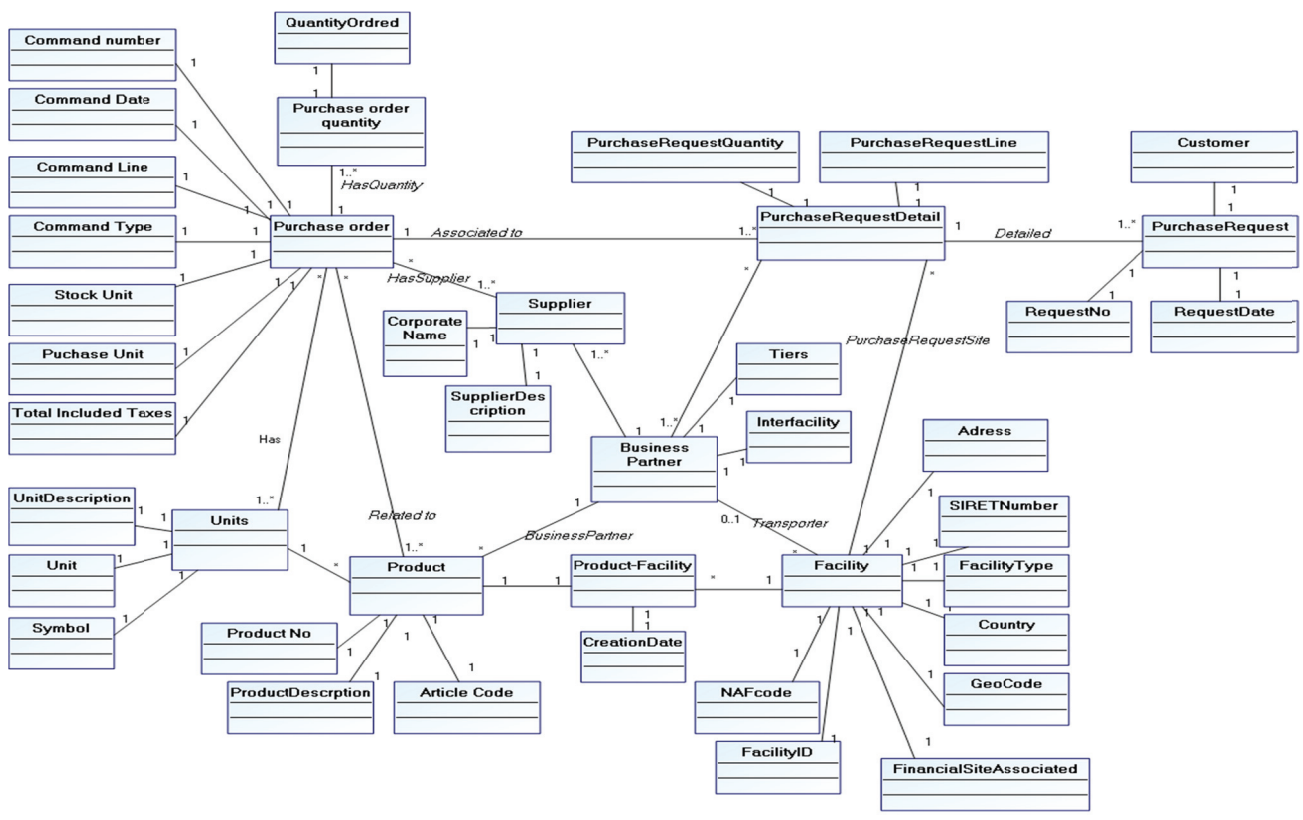


Fig. 11. The conceptual model of a purchase order in SAGE X3 application: fact-oriented model

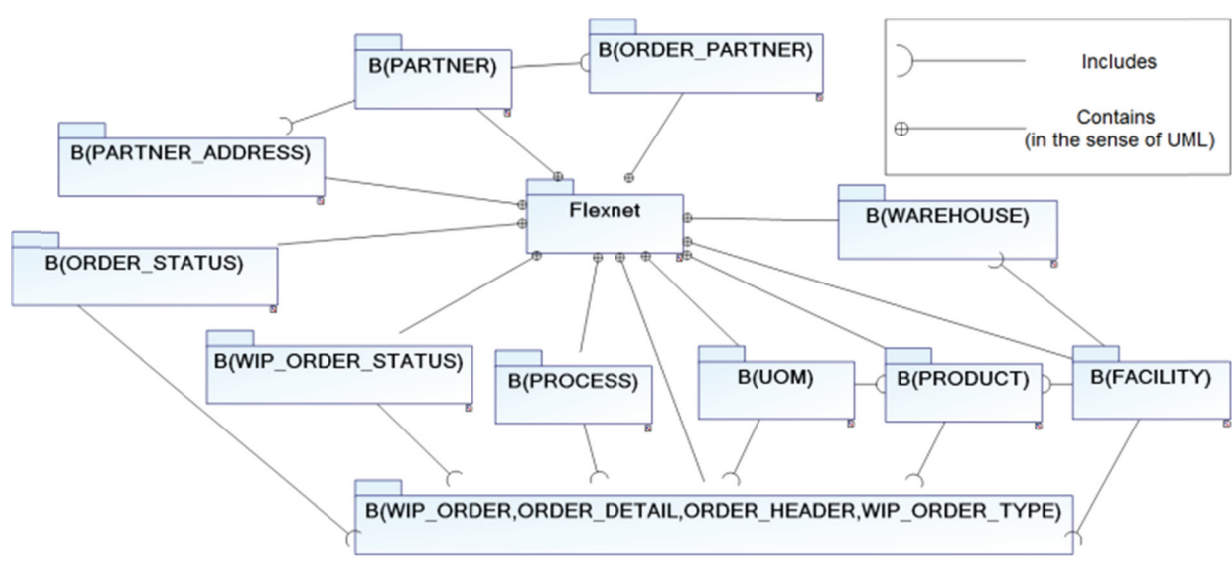


Fig. 12. The computed semantic blocks related to the Flexnet Purchase order

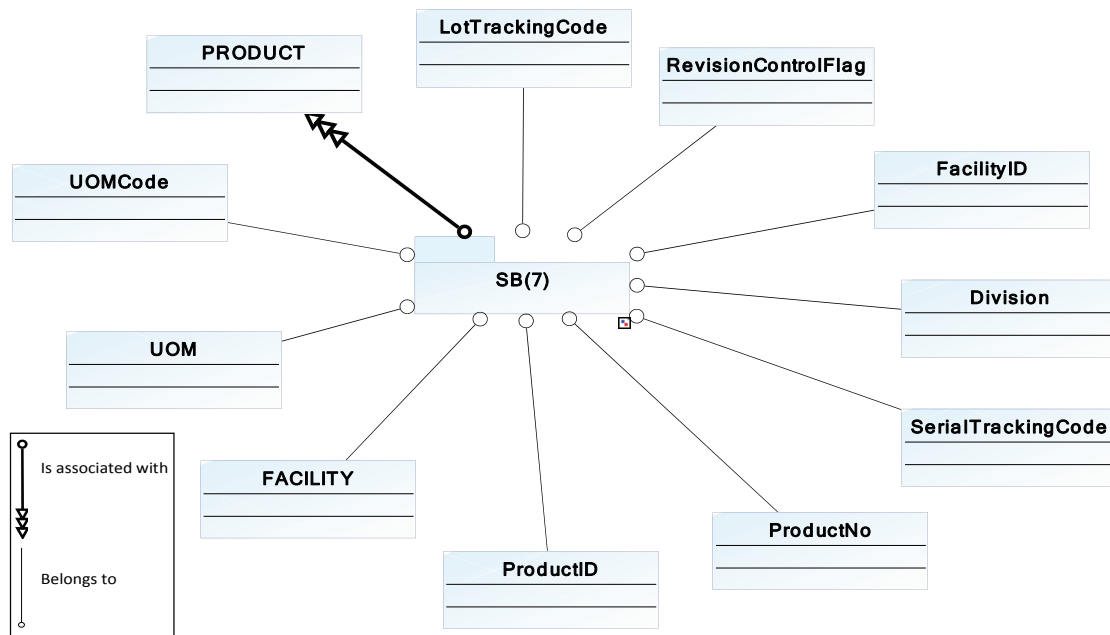


Fig. 13. The semantic block B(PRODUCT) for the conceptual model of figure 11

## 6. CONCLUSION

Semantics enactment among *ISs* 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