To link to this article: DOI: 10.1080/0951192X.2011.597431
http://dx.doi.org/10.1080/0951192X.2011.597431

To cite this version: Ulmer, Jean-Stéphane and Belaud, Jean-Pierre and Le Lann, Jean-Marc *Towards a pivotal-based approach for business process alignment.* (2011) International Journal of Computer Integrated Manufacturing, vol. 24 (n° 11). pp. 1010-1021. ISSN 0951-192X

Towards a pivotal-based approach for business process alignment

Jean-Stéphane Ulmer*, Jean-Pierre Belaud and Jean-Marc Le Lann

Laboratoire de Génie Chimique, Institut National Polytechnique ENSIACET, Université de Toulouse,
CNRS UMR 5503, 4, allée Emile Monso – BP 44362, 31432 Toulouse Cedex 4, France

This article focuses on business process engineering, especially on alignment between business analysis and implementation. Through a business process management approach, different transformations interfere with process models in order to make them executable. To keep the consistency of process model from business model to IT model, we propose a pivotal metamodel-centric methodology. It aims at keeping or giving all requisite structural and semantic data needed to perform such transformations without loss of information. Through this we can ensure the alignment between business and IT. This article describes the concept of pivotal metamodel and proposes a methodology using such an approach. In addition, we present an example and the resulting benefits.

Keywords: business process engineering; metamodelling; transformation; alignment

1. Introduction

Adaptation to an unstable demand, ability to improve efficiency and bringing changes to its value chain are the challenges that companies have to constantly confront. In order to remain competitive, a company must be able to describe and remain reactive to an endogenic or exogenic event. Such flexibility can be obtained by using a process-oriented approach or BPM (business process management). BPM represents the business process engineering of the organisation using information technology (Smith *et al.* 2002, Malone *et al.* 2003). It is intended to model, deploy, execute and optimise business processes in an ongoing way. A BPM cycle consists of three major steps (Figure 1).

The first step is the business process analysis (BPA). During this step, process models including various views (functional, informational, organisational and resource) are constructed. The second step is intended to deploy and execute business process, the business process implementation (BPI). The third one relies on monitoring processes and analysing data. It gives scorecards and key performance indicators (Alfaro *et al.* 2009). This step, the business activity monitoring (BAM), is out of the scope for this article.

The BPM is most often seen and interpreted as a further step and a natural evolution of the workflow management. This could explain the attention given to the functional view and the associated control-flow during the BPA step.

Specialised editors, such as ARIS, IBM Telelogic, MEGA, W4 ... offer BPM-based services. These BPM tool suites provide some methods that guide the end-user through a BPM cycle. For example, the Architecture of Integrated Information Systems (ARIS) editor defines its BPM approach by 'ARIS Methodology', which contains four phases (BP strategy, BP design, BP implementation and BP controlling).

This article proposes a methodology[1] which enhances transformations between heterogeneous models, e.g. from analysis models to implementation models and vice versa. This enhancement is obtained, thanks to a systematic model formalisation and the use of a pivotal metamodel. By obtaining this formalisation, we allow reverse transformations. The models are synchronised and consistent with each other. This synchronisation and intermodel consistency reduce the gap between business domain and IT domain, which increase the Business-IT alignment.

The article is organised as follows. Section 2 reviews one of the main research areas about heterogeneous model alignment. In Section 3, we highlight the difficulties encountered in not only translating a BPA model to BPI model but also to maintain them structurally or semantically consistent. The use of a pivotal metamodel-centric approach is justified in Section 4. Section 5 presents the methodology used to formalise the relationships between models, metamodels and pivotal concepts. Section 6 describes the prototype framework, the standards, technologies and tools used to accomplish this purpose. This framework supports our methodology and reveals how pivotal model and metamodel are used to transform BPA and BPI models. An example illustrating these elements is also detailed in this section. Then the benefits of the approach are discussed in Section 7. Finally, conclusions and future outlook are presented in Section 8.

2. Model-driven architecture vs. BPM

Several research areas study on models as a solution for enterprise performance including the model-driven engineering (MDE) (Perez *et al*. 2006, Combemale 2009), and particularly through the model-driven architecture (MDA) approach (Kadima 2005).

MDE aims at generating whole or part of software, based on models and their metamodels, and facilitates the definition of domain-specific modelling languages (DSMLs). These languages provide a specific formalisation to the technical aspects used. MDA focuses on defining Platform Independent Models (PIMs), technically independent from execution platform (J2EE, .Net, PHP ...) and enables the automatic generation of a set of Platform Specific Models (PSMs) (Figure 2).

We can identify the different components of BPM with those of MDA. As described by OMG (2007), Computation Independent Models (CIM) are associated with system requirements and/or a business domain. A CIM describes the environment, business processes and other specific requirements of the system. It supports the definition of business rules and vocabulary and represents the organisational aspect of the system. So a CIM can be associated with a business model obtained during the BPA step. But this business model remains incomplete without the control-flow description of business processes. This information can be provided by PIM. According to Panetto (2007), PIM represents business functionalities and the system behaviour without worrying about technical details. The PIM is a conceptual model independent of any considerations related to the target platform, its language or used technologies. It captures the logical aspect of the business process and respects rules set by the CIM.

The association of CIM and PIM can be identified as the BPA model. PIMs are then 'technically enriched' to generate a PSM. The PSM may be related to a system, language or technology, unlike PIM. This step is critical to the generation of code and therefore the implementation of the target process, and it is typically the goal of the BPI's step. So PSM can be identified as an implementation model, the BPI model (Figure 3). However, the analogy has some limits (Smith 2003). BPM and MDA were not designed to achieve the same objectives. MDA had been designed to help software design and generation while BPM allows process engineering.

Concepts from MDE and the use of MDA provide a multi-domain management with/for models of different abstractions. Nevertheless, MDA is too restricted for the engineering of business process. MDA does not formally describe how business models are defined at a CIM level and how they are associated to PIMs. It becomes necessary to evolve from an MDA-approach to a BPM-approach, but

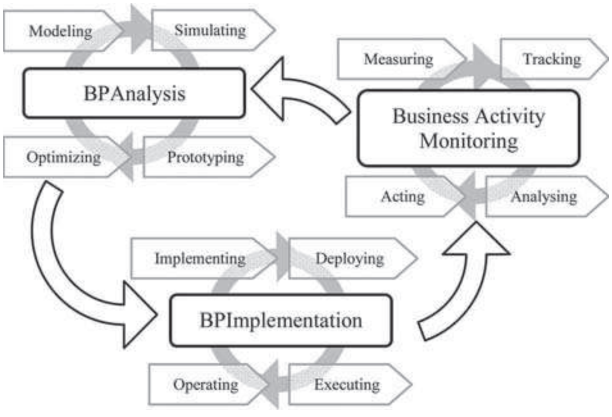


Figure 1. BPM Cycle (Debauche and Megard 2004).

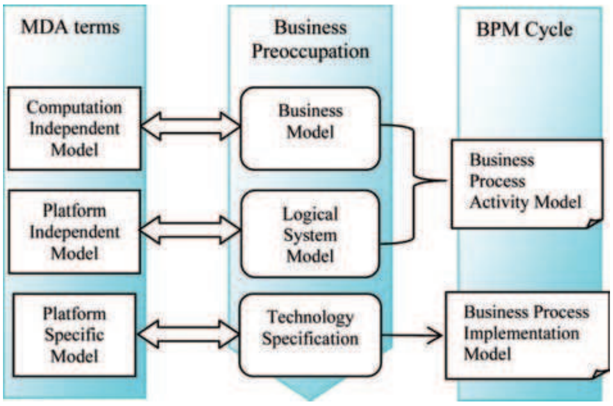

MDA terms
Business Preoccupation
BPM Cycle
Computation Independent Model
Business Model
Platform Independent Model
Logical System Model
Business Process Activity Model
Platform Specific Model
Technology Specification
Business Process Implementation Model

Figure 2. MDA and BPM, adapted from Model-driven.org[2].

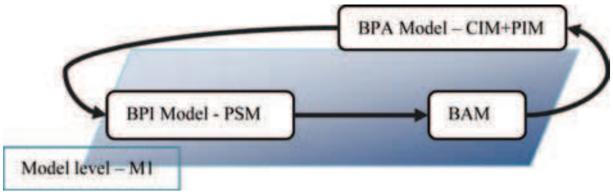


Figure 3. MDA and BPM.

concepts from MDE and MDA have to be taken into account.

3. Problem statement

The main issue encountered in a BPM cycle is the 'discontinuity' among business analysis and IT implementation views. For business and IT professionals, the inability to bridge the gap is mainly due to differences in model objectives. The analysis step (BPA) generates informal business process models and mostly interpretable by human beings. This contributes to complicate the implementation step (BPI). The lack of mutual understanding impedes the production of desired results. This is a 'Business-IT alignment' problem in the sense that the company is unable to use IT effectively to achieve its business objectives. Several transformations from a BPA model to a BPI model are required. It is necessary to emphasise that models will be modified and implemented processes might evolve. Thereby, synchronisation and model-consistency are imperative.

Another key issue that arises is the use of unique software platforms in integration-based approach which leads to an editor dependency. As processes and theirs models are evolving perpetually, an enterprise might have to change its software platform or adapt to new technologies. Hence, the ability to modify modelling tool or integration platform necessitates the use of 'loose coupling' between BPA and BPI models. Loose coupling means that these models should remain autonomous to each other and that their environments can be modified, in spite of the strong intrinsic interaction between them. That defines the underlying reciprocal influence existing between the two models (e.g. any modification on a BPA model could have repercussion on a BPI model and vice versa).

4. Proposed methodology

We propose a methodology for the BPM integrating concepts from MDE. This methodology allows business analysts to develop conceptual models, generally graphical ones, in accordance with a formalised metamodel. It also guarantees to IT specialist the conversion of a conceptual model to a block-structured one (technical model) in order to ease its implementation on an execution platform. And from an implementation model, our approach shall fully restore a graphical model, in the spirit of reverse engineering (Müller and Kienle 2010). This synchronisation between BPA and BPI models is considered like a necessary condition for model consistency. Our methodology is generic. However, specific languages and standards are required for its use.

4.1. From bidirectional transformations to the notion of pivot

Consider $\mathrm{MM_{BPA}}$, $\mathrm{MM_{BPI}}$, metamodels and $\mathrm{m_{BPA}}$, $\mathrm{m_{BPI}}$, their associated models. Bidirectional transformations between models help to ensure their maintenance and their consistency. This transformation is a way to algorithmically specify model-consistency. This transformation can be bijective:

$$f: \mathrm{MM_{BPA}} \rightarrow \mathrm{MM_{BPI}}\ et f^{-1}: \mathrm{MM_{BPI}} \rightarrow \mathrm{MM_{BPA}} \quad (1)$$

And we obtain:

$$\forall \mathrm{m_{BPA}} \in \mathrm{MM_{BPA}}, \exists ! \mathrm{m_{BPI}} \in \mathrm{MM_{BPI}}, f(\mathrm{m_{BPA}}) = \mathrm{m_{BPI}}, f^{-1}(\mathrm{m_{BPI}}) = \mathrm{m_{BPA}} \quad (2)$$

In most of cases, the bijective transformation is too restrictive and is impossible to get if models cardinalities are different (Stevens 2008). But, we consider models that can be heterogeneous and different levels of abstraction. Each model may have information that is not contained in other, in particularly with BPA models and BPI ones. A possible approach is to take one of the models and to modify, so that it contains all information from other models. The transformation is made subjective without modifying models appearance to the users. Consider τ_{BPA} and τ_{BPI} two transformations and $\mathrm{MM_{Int}}$ an intermediary metamodel:

- $\tau_{\mathrm{BPA}} : \mathrm{MM_{BPA}} \rightarrow \mathrm{MM_{Int}}$

$$\tau_{\mathrm{BPA}}(\mathrm{m_{BPA}}) = \mathrm{m'_{BPA}} \quad (3)$$

- $\tau_{\mathrm{BPI}} : \mathrm{MM_{BPI}} \rightarrow \mathrm{MM_{Int}}$

$$\tau_{\mathrm{BPI}}(\mathrm{m_{BPI}}) = \mathrm{m'_{BPI}} \quad (4)$$

Models $\mathrm{m_A}$ and $\mathrm{m_B}$ are considered as equivalent if and only if:

$$\mathrm{m'_{BPA}} \equiv \mathrm{m'_{BPI}} \quad (5)$$

So if we consider that $\mathrm{MM_{BPA}} \subseteq \mathrm{MM_{Int}}$ and $\mathrm{MM_{BPI}} \subseteq \mathrm{MM_{Int}}$, we obtain the following relationship

$$\mathrm{m_{BPA}} \xrightarrow{\tau_{\mathrm{BPA}}} \mathrm{m'_{BPA}} \equiv \mathrm{m'_{BPI}} \xrightarrow{\tau_{\mathrm{BPI}}} \mathrm{m_{BPI}} \quad (6)$$

The bijective transformation can be made subjective, without changing the models' appearance perceived by the user, and $\mathrm{m_{BPA}}$ and $\mathrm{m_{PBI}}$ remain unchanged. Our pivotal approach starts with this new equivalence and defines these transformations τ_{BPA} and τ_{BPI} as functions of constructive conformity, and the built model from these transformations is the pivotal model.

4.2. *Notion of pivot*

The concept of pivot has already been used, designed for example in database management systems (DBMS). The use of different models in DBMS gives some issues of syntaxical heterogeneity. A solution is obtained by translating all the schemas into a common model, the pivotal model. This pivotal concept can be found in computing research activities (such as MDE). For its model implementations, the Fiabilité d'ARchitectures Orientées Services (FAROS) project[3] (Blay-Fornarino *et al.* 2008) uses a similar concept. The transformation from business models to pivotal models eases transformation of business elements from pivotal models to technical ones.

Thus fulfilling the 'pivotal approach' commonly used in system interoperability (Meinadier 2002), we consider that a pivotal metamodel eases transformations between models, and a pivotal model is necessary to reduce issues of syntaxical heterogeneity issues. The pivot's role is to maintain a semantic equivalence between BPA and BPI models. For our approach, we are expanding its scope by adding the following:

- Since information gaps exist for implementing BPA models and for analysing BPI model, a pivotal metamodel must be able to strengthen them by adding necessary information, to preserve the information integrity and its consistency during the BPA–BPI transition (and vice versa).
- It must also allow autonomy between the target model and the initial model (e.g. be able to modify the BPA model without taking in consideration the BPI one) in order to have a loose-coupling between BPA and BPI models.
- This intermediate format (the model from the pivotal metamodel) becomes necessary to store and exchange information between the modelling and integration environments (Figure 4). Each metamodel focuses on different aspects of the same process, hence considering their relationships allow for a more in-depth comprehension of the process model (Saidani and Nurcan 2008). In our case, the relationships are established by the pivotal element.

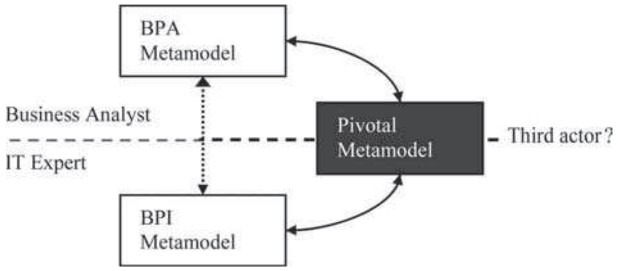


Figure 4. BPA, BPI and Pivotal metamodels.

These relationships between these two metamodels are determined according to the business domain. This involves the consideration of structural and semantic features of modelled processes. However, during a BPM approach, BPA and BPI metamodels are not always explicit and formalised. Thus, transformation rules between BPA and BPI models are not really flexible. According to our approach, the specification of the metamodels relies on the pivotal metamodel. In this way, we systematically provide formalised metamodels and ease mappings between them.

In a classic BPM-approach context, an enterprise has two main actors: business analyst and IT expert. According to (Various IIBA and Brennan 2008) a business analyst seeks new ways to improve business efficiency. This improvement can be done by increasing coordination between working teams changing tools or processes. The IT expert then addresses these business requirements and converts them into IT requirements. Due to the several issues as discussed before (Section 3), a third role is needed, the role of a process architect (Figure 5).

Concretely, the role of a process architect is to determine which data from the BPA model are used into the associated BPI model (Figure 5a) (e.g. control-flow data). The process architect must be able to provide the necessary information in order to complete the BPI model (Figure 5b) (e.g. details on roles and methods). Finally, he guarantees the preservation of the information model integrity from a BPA model to the resulting BPI one (Figure 5c) (e.g. graphical information, unspecific or irrelevant annotation). The reverse operation (a BPI model to a BPA model) is achieved in a similar way and requires a similar involvement of the process architect. Indeed a process architect is responsible of the technical strategy of the organisation, who must keep a global and complete vision of the BPM methodology.

The Figure 6 shows how the pivotal step provides an additional file containing information-type (b) through a BPM cycle. During the first transformation, the m_{Pivot} stores specific data form the input model (here m_{BPA}). The m_{Pivot} also provides data with their default values used by the output model (here m_{BPI}). Then these data are manipulated by the correct actor (here the IT expert). During the second transformation, the pivotal model is able to store data specific to the input model and to restitute data stored at the

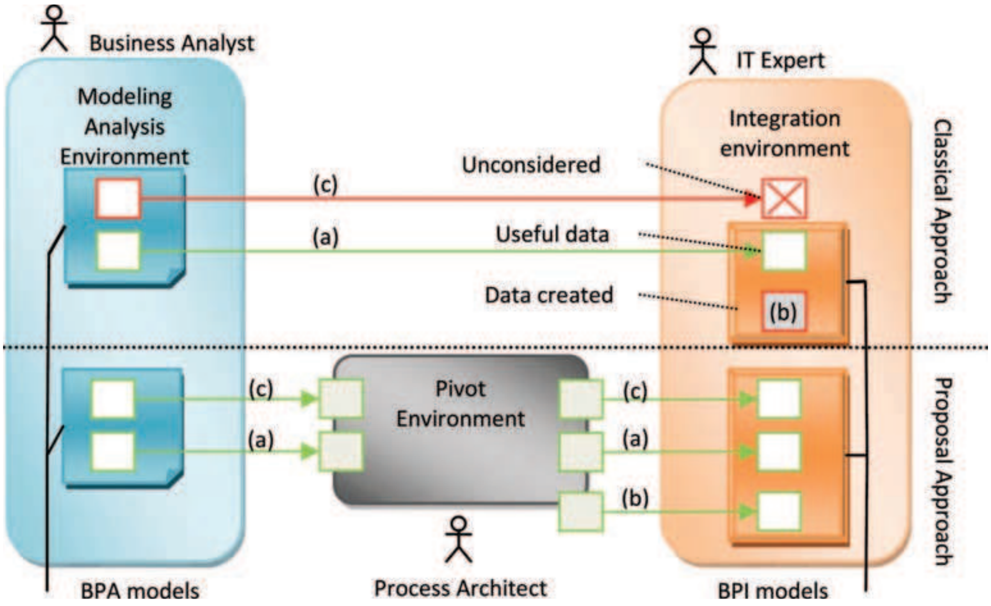


Figure 5. Roles, data and environments.

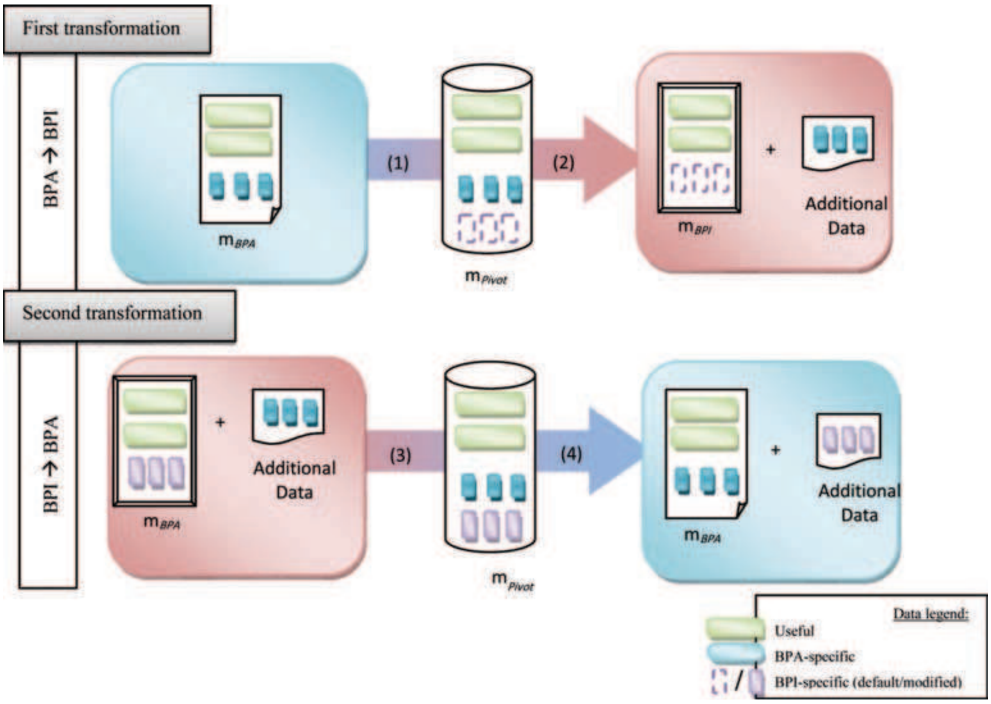


Figure 6. BPA to BPI with our pivot.

previous transformation. We obtain a complete output model without any data loss.

4.3. *Genericity concept*

Adaptability to all types of models, standard or specific languages, would make our pivotal metamodel complex, difficult to implement and to maintain, and in most cases infeasible to resolve. In order to reach a complete genericity, our pivot will become a ‘monster’. We must seek a compromise between absolute genericity and agility. In our case, the approach is restricted to ‘relative genericity’, i.e. our pivot is relative to:

- a business domain (banking, physico-chemical process . . .),
- a context of study and
- a desired level of abstraction.

Nevertheless, the overall methodology to build our pivot is generic and independent of business domain and selected technologies.

4.4. *Scope of our methodology*

The enterprise modelling consists of several views showed in Figure 7: functional view, informational view, organisational view and resource view (Vernadat 1996). Thus, our methodology focuses on the functional view and especially on the concept of activity/process as shown in (Vernadat 1999).

The concepts inherent to the other views may intervene, depending on modelling languages used or on the process architect modelling requirements. The pivotal metamodel can be given elements from other views than the functional one and that are considered necessary by the process architect. For example, a standard language such as BPMN focuses on the functional view but also allows:

- To model input/output documents/data (informational view);
- To define actions involved in products/information networks (resource view).

In the same way, our current pivotal metamodel contains the swimlane elements pool and lane, related to the organisational view, in order to describe the actor's roles performing the modelled process.

5. Methodology

In this section, we formally describe how our methodology can guarantee consistency links between its different elements. Then, the various steps constituting our framework are defined. A partially implemented case study on the proposed framework is discussed at the end of this section.

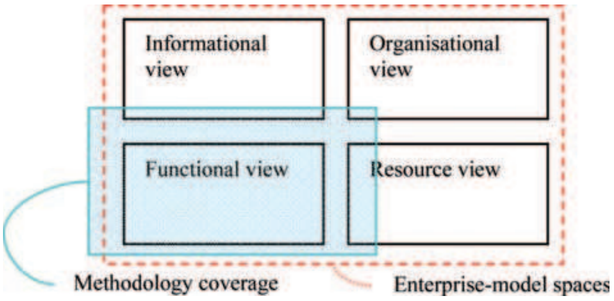


Figure 7. Methodology coverage and enterprise-model spaces (from (Touzi 2007)).

5.1. *Consistency between model, metamodel and pivotal metamodel*

We formally define a BPA metamodel, MM_{BPA}, by two elements. The first element is constituted by a representation standard or language, MM_{rep}, which is generally a business user-friendly graphical language. The second element is a set of business rules setting a business repository, RefMet. This repository can be obtained by using constraints or business rules as object constraint language (OCL) or semantic business vocabulary and business rules (SBVR).

The MM_{BPA} is specified as follows (μ):

$$MM_{rep}\,\mu MM_{BPA} \tag{7}$$

$$\mathrm{RefMet}\,\mu\, MM_{BPA} \tag{8}$$

In a same way, the BPA model, m_{BPA} has to conform to (χ) MM_{rep} and RefMet (Figure 8). Thus, we obtain the following relation:

$$m_{BPA}\chi MM_{BPA} \tag{9}$$

We can explain by analogy the different links existing between m_{BPI} and MM_{BPI} using their own MM_{rep} and RefMet. The relations of compliance between model and metamodel being established, we can specify the rules of transformations allowing converting a model m_{BPA} or m_{BPI} to a pivotal model

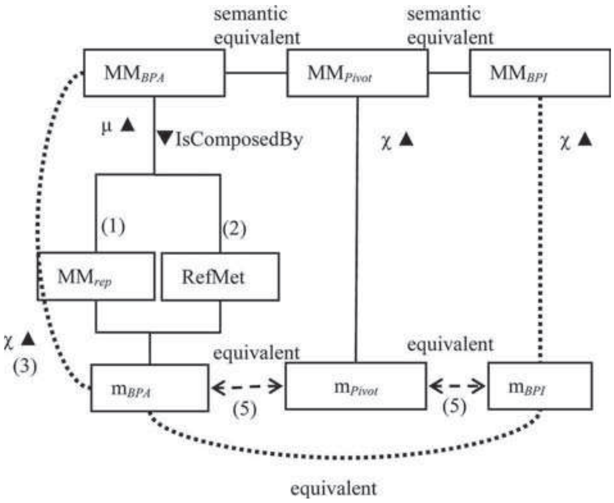


Figure 8. Relationships between models, metamodels and pivotal metamodel.

$\mathrm{m_{Pivot}}$. To ensure the consistency of models used in these transformations, we define functions of constructive conformity (Favre *et al.* 2006) $\mathrm{fc_{BPA}}$ and $\mathrm{fc_{BPI}}$, respectively defined from $\mathrm{MM_{BPA}}$ and $\mathrm{MM_{BPI}}$ to $\mathrm{MM_{Pivot}}$. Let us consider that $\mathrm{m_{BPA}}$ belongs to the space of models conforming to $\mathrm{MM_{BPA}}$ as for $\mathrm{m_{BPI}}$ and $\mathrm{MM_{BPI}}$:

$$\mathrm{m_{BPA}} \in L(\mathrm{MM_{BPA}});$$

$$\mathrm{m_{BPI}} \in L(\mathrm{MM_{BPI}}).$$

We obtain the following:

$$\mathrm{m_{BPA}} \overset{s}{\equiv} \mathrm{m_{BPI}} \Leftrightarrow fc_{\mathrm{BPA}}(\mathrm{m_{BPA}}) \equiv fc_{\mathrm{BPI}}(\mathrm{m_{BPI}}) \qquad (10)$$

In a general way, with i for BPA or BPI:

$$\exists\, m_{pivot} \in L\left(MM_{pivot}\right),\ fc_i(m_i) = m_{pivot} \qquad (11)$$

We obtain a link between BPA and BPI and their respective models $\mathrm{m_{BPA}}$ and $\mathrm{m_{BPI}}$, which can be considered as 'equivalent' (Figure 8). However, if we have $fc_i(m_i) = m_{pivot}$, we should keep in mind that $MM_i \neq MM_{pivot}$.

5.2. *Pivotal metamodel and semantic equivalences*

The functions of conformity allow producing a pivotal model equivalent to a BPA model or a BPI one (Figure 8). In this article, we focus particularly on the conformity of the elements modelled in the control flow.

5.2.1. *Elements of the pivotal metamodel*

In this section, we define the elements used in the different metamodels.

As a first step, our study is restricted to use only 17 objects (Table 1) at the stage of modelling. This reduces the expressiveness of the language by limiting the number of elements (Ulmer and Belaud 2008), in order to reduce the scope of our study and to ensure a model transformation from one kind of language (like a graphical one) to another one (like implementation language) unambiguously.

This set provides sufficiently expressive generic object-oriented concepts capable to model most of processes encountered in industrial companies (Zur Muehlen and Recker 2008). These elements are forming the simple model portability conformance class as defined by WfMC (2008). A modelling tool belonging to this class should be able to import and to understand each individual element of this class. We extend this definition to our study, in which a BPI tool can import and understand a BPA model and in the same way a BPA tool a BPI model. Each of these models must stay in accordance with our pivotal metamodel. We notice that even if the item 'pool' is present, this approach does not take in consideration, for now, the choreography of collaborative processes. In the same way, in order to ease metamodel transformations and manipulations, we limited our diagram hierarchy to three levels: process – subprocess – activity.

Table 1. Simple class set.

#	Class	Type	
01	Node	Event	Start
02			End
03		Action	Task
04			Activity
05			Sub-process
06			Process
07		Logical	Exclusive
08			Inclusive
09			Parallel
10	Edge	Link	Uncontrolled
11			Conditional
12			By default
13		Association	
14	Swimlanes	Pool	
15		Lane	
16	Artifacts	Data object	
17		Annotation	

Our metamodel, inspired by the XPDL specification (Morley *et al.* 2005, WfMC 2008), is activity-centric and is expressive enough to represent most business processes (Figure 9).

5.2.2. *Semantic equivalences*

To identify the semantic relationships between the elements of two models, we will handle the definitions of semantic equivalences of (Rizopoulos and Mçbrien 2005):

- Equivalence: the concepts of models A and B are equivalent, $A \overset{s}{=} B$,
- Subsumption: A is a subsumption of B, $B \overset{s}{\subset} A$,
- Intersection: the concepts of A intersect the B ones, $A \overset{s}{\cap} B$,
- Disjunction: A and B are disjointed if and only if $A \overset{s}{\not\cap} B$.

Thanks to these relationships we can identify the semantic links between our pivotal metamodel and the BPA and BPI metamodels.

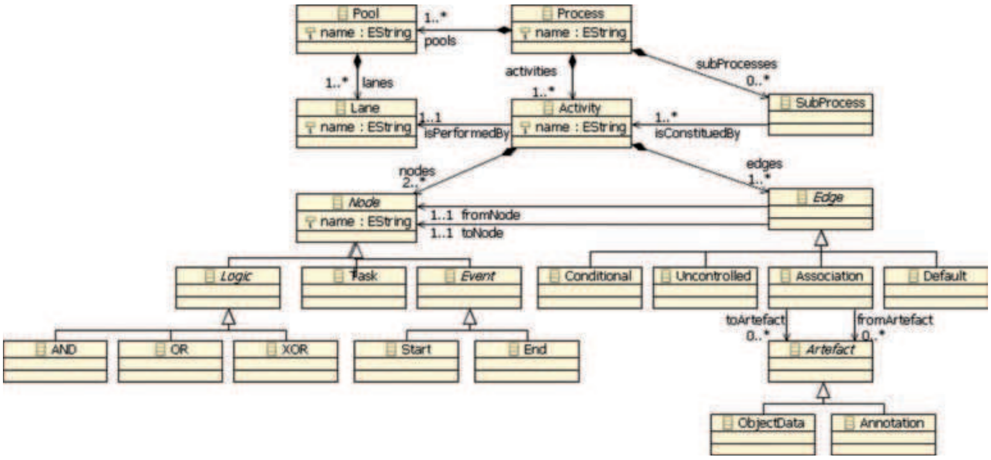


Figure 9. Pivotal metamodel.

6. Methodology implementation

The previous section proposed a model compliance methodology by analysing its elements. After formally defining our approach, we now apply it on a triplet (analysis environment, pivot environment and implementation environment).

A 'prototype software' framework is being developed that instantiated the methodology. It is based on the Eclipse platform, especially on eclipse modelling framework (EMF). The metamodelling part is realised by using Ecore tools. If the mapping between metamodels is determined by the process architect, it is assisted by the Kermeta (Kernel Metamodelling) language (Muller *et al.* 2005). After a detailed presentation of the framework development, the Section 6.2 presents a short demonstration of the software framework.

6.1. *Framework development*

Developed by the Triskell team, the Kermeta language is an extension of the Essential Meta-Object Facilities (EMOF) language. Languages like EMOF or Eclipse can only model structures with concepts like classes, attributes, associations. Kermeta enables the possibility to describe the semantic and the behaviour of these structures helped with its imperative action language. Kermeta also allows the writing of model transformation and model constraints. Business rules will be written according to the SBVR standard.

Figure 10 illustrates the passage from a BPA diagram to an implementation model and its code (transformations t1-t2-t3-t4), and vice versa (transformations t4-t5-t6-t1). In order to realise these transformations, we must first do mapping between MM_{BPA}, MM_{BPI} and MM_{Pivot} (m1, m2, m3, m4).

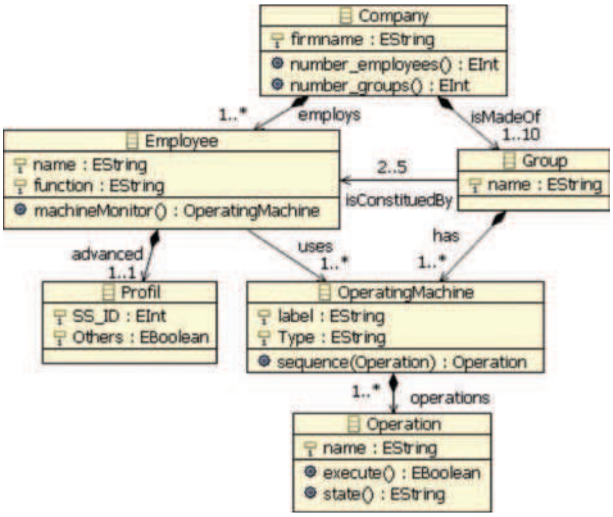


Figure 10. Implementation of the approach.

6.2. *From BPA to pivot*

To illustrate our approach, we describe the transformation from an organisational model m_{BPA} to the pivotal model m_{pivot}. Foremost, the respective metamodels are constituted. Elements of our metamodel shown in Figure 9 are identified with those of XPDL standard. Thus, we obtain our pivotal MM_{rep}, an altered form of the XPDL process definition metamodel. As we wish to highlight efforts to perform the mappings and transformations, we consider in this article the metamodel shown in Figure 11 in Ecore diagram format as our BPA MM_{rep}, SimpleCompany. The company model used is a generic one, in order to ease its representation and comprehension. The instantiated model m_{BPA} is partially shown in Figure 12. We do not consider, in this example, the business rules and the graphical aspects.

The next step in our approach is to realise the mapping (m1) between the two metamodels. We decide to use Kermeta as an aspects-weaver adapted to Ecore metamodels which is able to manipulate them without modification (Mosser and Blay-Fornarino 2009). Hence, defining a transformation using Kermeta is equivalent to implementing one (or more) visitor(s), within the meaning of visitor design pattern (Gamma *et al.* 1999). The visitor design pattern is applied as follows (Figure 13): each element to be visited has an accept() method (a) that takes 'visitorEntreprise' as an argument. The accept() method calls back the visit*ElementName*() with the visited element as argument (b). Thus, a visitor may be aware the reference of the element and calls its methods. Using this pattern is particularly advantageous, as it facilitates the addition of new operations that may be required during transformations. Indeed, a new operation on a metamodel is translated by adding a new visitor. Conversely, the addition of new elements is difficult: for each element, a *new operation in the visitor* has to be made. Nevertheless, if a certain level of maturity is reached by the process architect, we assume that within our approach, we more often modify/add/remove

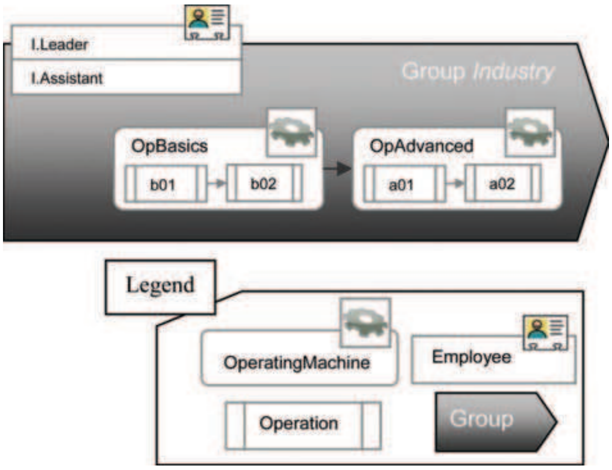


Figure 11. Studied metamodel, SimpleCompany.

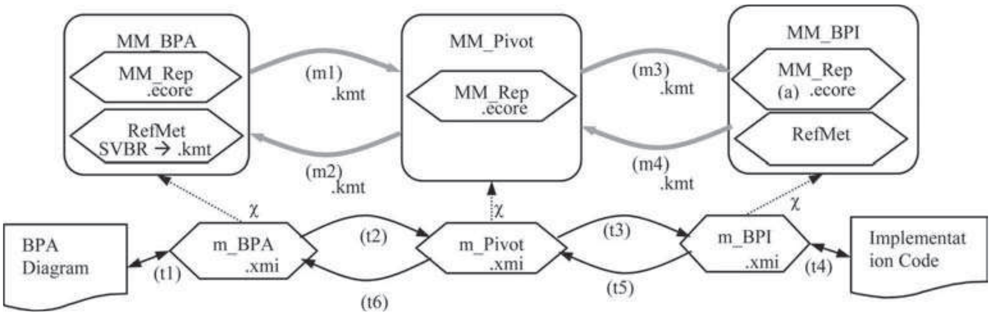
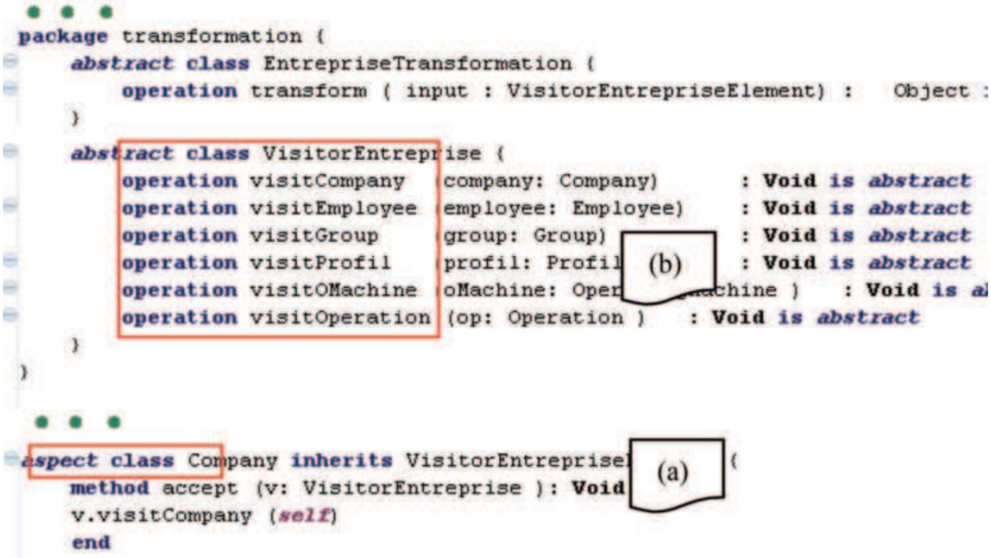


Figure 12. Extract from the SimpleCompany instantiated model.



```
package transformation {
    abstract class EntrepriseTransformation {
        operation transform ( input : VisitorEntrepriseElement) :   Object :
    }
    abstract class VisitorEntreprise {
        operation visitCompany    (company: Company)        : Void is abstract
        operation visitEmployee   (employee: Employee)      : Void is abstract
        operation visitGroup      (group: Group)            : Void is abstract
        operation visitProfil     (profil: Profil           : Void is abstract
        operation visitOMachine   (oMachine: Oper       chine )    : Void is al
        operation visitOperation  (op: Operation )    : Void is abstract
    }
}

aspect class Company inherits VisitorEntreprise        {
    method accept (v: VisitorEntreprise ): Void
    v.visitCompany (self)
    end
```

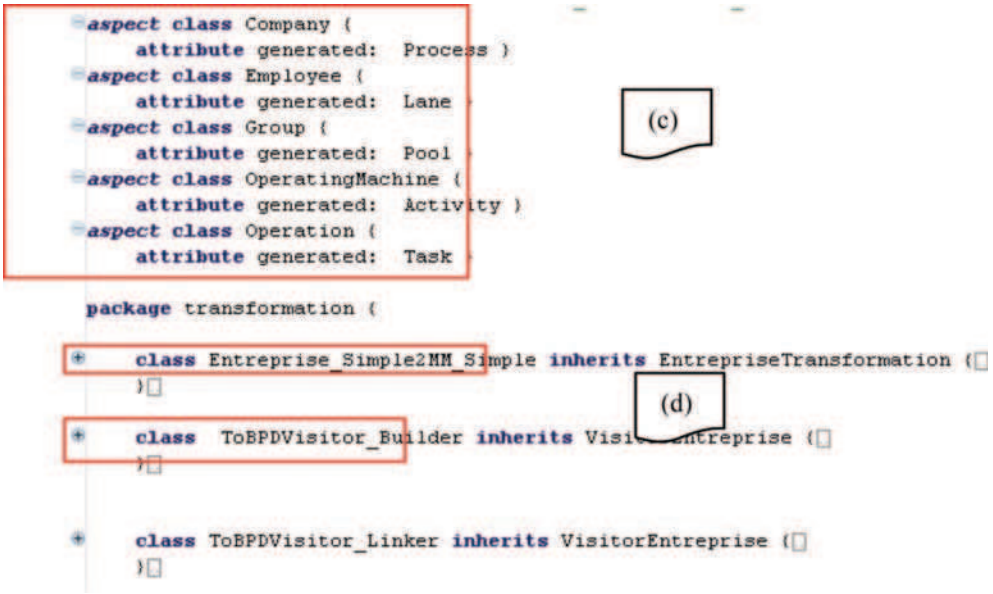
Figure 13. Extract from the enterprise visitor code (in Kermeta language).

operations performed on metamodels than change these metamodels.

Finally, we can use and manipulate concepts of the BPA metamodel under consideration (Figure 14) and realise the mapping m1 from the Enterprise_Simple to the $\mathrm{MM}_{\mathrm{Pivot}}$ (c) as the transformation t2 from an Enterprise_Simple model to a $\mathrm{m}_{\mathrm{Pivot}}$ (d) (Builder/linker method).

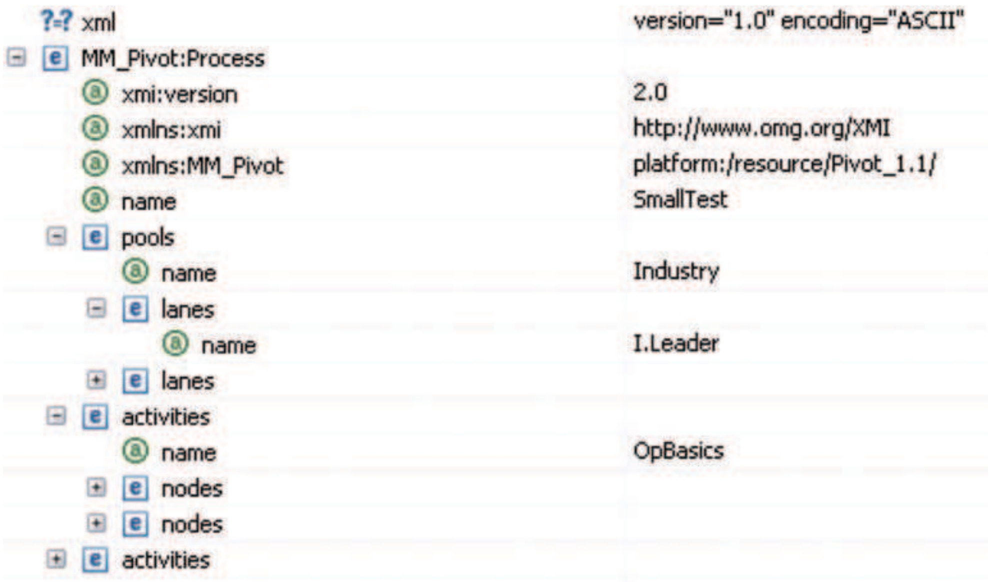
The transformation t2 of $\mathrm{m}_{\mathrm{BPA}}$ is shown in Figure 15. Therefore, at the current state of our framework prototype development, we succeed to execute the mappings m1 and m2, and the transformations t1, t2 and t6.

Unlike a usual analysis-implementation transformation (Grangel *et al.* 2010), the pivotal model obtained, $\mathrm{m}_{\mathrm{Pivot}}$, contains all data belonging to the analysis model $\mathrm{m}_{\mathrm{BPA}}$. It also contains elements necessary to establish complete and comprehensive implementation model $\mathrm{m}_{\mathrm{BPI}}$. Future works will demonstrate how modifications from a $\mathrm{m}_{\mathrm{BPI}}$ to a $\mathrm{m}_{\mathrm{BPA}}$,



```
aspect class Company {
    attribute generated:  Process }
aspect class Employee {
    attribute generated:  Lane }
aspect class Group {
    attribute generated:  Pool }
aspect class OperatingMachine {
    attribute generated:  Activity }
aspect class Operation {
    attribute generated:  Task }

package transformation {

    class Entreprise_Simple2MM_Simple inherits EntrepriseTransformation {
    }

    class  ToBPDVisitor_Builder inherits VisitorEntreprise {
    }


    class ToBPDVisitor_Linker inherits VisitorEntreprise {
    }
```

Figure 14. Extract from the “enterprise to pivot” transformation code.



```
?=? xml                                 version="1.0" encoding="ASCII"
⊟ e MM_Pivot:Process
    a xmi:version                       2.0
    a xmlns:xmi                         http://www.omg.org/XMI
    a xmlns:MM_Pivot                    platform:/resource/Pivot_1.1/
    a name                              SmallTest
  ⊟ e pools
      a name                            Industry
    ⊟ e lanes
        a name                          I.Leader
    ⊞ e lanes
  ⊟ e activities
      a name                            OpBasics
    ⊞ e nodes
    ⊞ e nodes
  ⊞ e activities
```

Figure 15. Extract from the $\mathrm{m}_{\mathrm{Pivot}}$ (in .xmi).

or vice versa, are propagated. Integrity of information being provided and modifications being propagated during transformations, our approach enables synchronisation and a semantic equivalence between models. We consider that these conditions improve the alignment between business and IT domains.

7. Pivotal-based approach's benefits

Architecture and business modelling software use several operations to transform a BPA model to a BPI one and vice versa. These tools and their underlying metamodels (when they exist) are usually opaque and slightly open. Thus, the result is frequently a unilateral change where the process model cannot be converted to other languages or to other editors. Furthermore consistency, inter-model compliance and alignment between models and metamodels are difficult to ensure.

For example ARIS contains various techniques for business process modelling. Every aspect of the modelled process is described by a metamodel. However, there is not any global metamodel ensuring consistency and a good visibility between these metamodels (Leist and Zellner 2006). In the 'good BPM architecture' proposed by Havey and Havey (2005), there is no feedback to the analysis models studied before. Several issues arise if these models are neither enhanced nor updated:

- models become 'contemplative',
- their relative documentation is difficult to use,
- and this results in a lag between BPI models and business process executed.

In order to resolve these issues, we explained the importance of a rigorous and semantic centred methodology. Our proposal for a pivotal approach guaranties a loose coupling between BPA and BPI, and consistency between models and metamodels. The methodology remains generic because of its language independency. It may use enterprise-specific or standard languages. Then we apply conformity and semantic equivalence rules to confirm intermodel consistency and bidirectional transformation.

However, using a pivotal metamodel complicates the transformation rules definition. We have to consider transformation problems between BPA, BPI and pivotal metamodels and not only between BPA and BPI ones. But as result of this approach, we ease transformations between models and obtain a better consistency between them, as explained in Section 4.

By providing a true 'communication' between analysis and IT models, this approach increases the business/IT alignment. Besides, we get semantically strong models, independent to modelling and integrating environments. Furthermore, we enable the exchange of implementation files among different implementation engines ($m_{BPI1} \rightarrow m_{Pivot} \rightarrow m_{BPI2}$).

8. Conclusion and outlook

During a process-lifecycle, consistency is difficult to maintain between models from different environments. The successive developments and changes made by different stakeholders lead to the development of inconsistencies between models. A discontinuity between business perspective and IT perspective appears.

In this article, we have proposed a semantic-oriented solution based on the concept of a pivotal metamodel, an essential element of our approach in business process engineering. Then we have defined how to formally establish relations of conformity between models and their metamodels as well as rules of semantic conformities between elements. The establishment of this pivot creates a loose coupling between process analysis and process integration.

Our generic approach was partially illustrated using the proposed framework through a simple example. Future research will allow us to validate and refine this approach. From these generic concepts, the prototype will therefore ensure the portability and validity of the analysis model, models' 'interoperability' and models' consistency. Another possible perspective of our work is to consider our approach in a service-oriented architecture context. Therefore, the pivot's role would be to divide the BPA model according to defined patterns associated with web-services. The relevance of such an approach is under study. In the near future, an industrial process from a SME will be studied in accordance with our approach, the target integration platform being an ERP software.

Notes

[1] We consider a methodology as 'a body of methods, rules, and postulates employed by a discipline: a particular procedure or set of procedures' as defined by Merriam-Webster dictionary (http://www.merriam-webster.com/dictionary/methodology).

[2] http://portal.modeldriven.org/

[3] RNTL FAROS project (a composition environment for reliable service-oriented architectures): http://www.lifl.fr/faros

References

Alfaro, J.J., *et al.*, 2009. Business process interoperability and collaborative performance measurement. *International Journal of Computer Integrated Manufacturing*, 22 (9), 877–899.

Blay-Fornarino, M., *et al.*, 2008. *Transformations depuis les modèles métier* [online]. Livrable F-2.4. Available from: http://www.lifl.fr/faros [Accessed 9 September 2010].

Combemale, B., 2009. Ingénierie Dirigée par les Modèles (IDM) – État de l'art. Unpublished work. Available from: http://hal.archives-ouvertes.fr/hal-00371565/en/ [Accessed 26 June 2010].

Debauche, B. and Megard, P., 2004. *BPM business process management: pilotage métier de l'entreprise*. Paris: Lavoisier.

Favre, J.-M., Estublier, J., and Blay-Fornarino, M., 2006. *L'ingénierie dirigée par les modèles: au-delà du MDA*. Paris: Hermès Science Publications.

Gamma, E., *et al.*, 1999. *Design patterns – Catalogue de modèles de conception réutilisables*. Paris: Vuibert.

Grangel, R., Bigand, M., and Bourey, J.-P., 2010. Transformation of decision models into UML: application to GRAI grids. *International Journal of Computer Integrated Manufacturing*, 23 (7), 655–672.

Havey, M.V. and Havey, M., 2005. *Essential business process modeling*. Sebastopol, CA: O'Reilly Media, Incorporated.

Kadima, H., 2005. *MDA: conception orientée objet guidée par les modèles*. Paris: Dunod.

Leist, S. and Zellner, G., 2006. Evaluation of current architecture frameworks. *In*: *2006 ACM symposium on applied computing*, 23–27 April 2006, Dijon. New York: ACM, 1546–1553.

Malone, T.W., Crowston, K., and Herman, G.A., 2003. *Organizing business knowledge – the MIT Process Handbook*. Cambridge, MA: MIT Press.

Meinadier, J.-P., 2002. *Le métier d'intégration de systèmes*. Paris: Hermès-Lavoisier.

Morley, C., *et al.*, 2005. *Processus métiers et S.I. – evaluation, modélisation, mise en œuvre*. Paris: Dunod.

Mosser, S. and Blay-Fornarino, M., 2009. Réflexions autour de la construction dirigée par les modèles d'un atelier de composition d'orchestrations. *In*: *15ème conférence francophone sur les Langages et Modèles à Objets (LMO'09)*. France: Cépadués, 139–152.

Müller, H.A. and Kienle, H.M., 2010. Reverse engineering. *Encyclopedia of Software Engineering*, 1 (1), 1016–1030.

Muller, P.-A., Fleurey, F., and Jézéquel, J.-M., 2005. Weaving executability into object-oriented meta-languages. *In*: S. Kent and L. Briand, eds. *Proceedings of MODELS/UML'2005, Lecture notes in computer science*. Vol. 3713. Berlin: Springer, 264–278.

OMG, 2007. *UML 2.1.2 Superstructure specification* [online]. Available from: http://www.omg.org/spec/UML/2.1.2/ [Accessed 26 August 2011].

Panetto, H., 2007. Towards a classification framework for interoperability of enterprise applications. *International Journal of Computer Integrated Manufacturing*, 20 (8), 727–740.

Perez, J.M., *et al.*, 2006. MDE for BPM: A systematic Review. *In*: *First international conference of software and data technologies, ICSOFT*, 11–14 September 2006, Setùbal, Portugal.

Rizopoulos, N. and Mçbrien, P., 2005. A general approach to the generation of conceptual model transformations. *In*: *17th Conference on advanced information systems engineering (CAiSE'05)*, Portugal. Berlin: Springer, 249–338.

Saidani, O. and Nurcan, S., 2008. Towards situational business process meta-modelling. *In*: *20th International conference on advanced information systems engineering (CAiSE'08)*, 16–18 June 2008, Montpellier, France, 93–96. Available from: CEUR-WS.org [Accessed 31 August 2011].

Smith, H., 2003. *BPM and MDA: competitors, alternatives or complementary* [online]. Business process trends. Available from: http://jobfunctions.bnet.com/abstract.aspx?docid=140700 [Accessed 31 August 2011].

Smith, H., *et al.*, 2002. *The emergence of business process management* [online]. CSC's Research Services. Available from: http://www.ebizq.net/topics/eai/features/1763.html [Accessed 31 August 2011].

Stevens, P., 2008. A landscape of bidirectional model transformations. *In*: R. Lämmel *et al.*, eds. *Lecture notes in computer science*. Vol. 5235. Berlin: Springer, 408–424.

Touzi, J., 2007. *Aide à la Conception de Système d'Information Collaboratif Support de l'Intéropérabilité des Entreprises*. Thesis (PhD). Ecole des Mines d'Albi Carmaux.

Ulmer, J.-S. and Belaud, J.-P., 2008. Analyse de l'expressivité des langages de modélisation de processus par les patterns. *In*: *Ingénierie d'Entreprise et des Systèmes d'Information*, Inforsid 2008, 27–30 May 2008, Fontainebleau, France.

Various, IIBA and Brennan, K., 2008. *A guide to the business analysis body of knowledge (BABOK guide)*. Marietta, GA: International Institute of Business Analysis.

Vernadat, F.B., 1996. *Enterprise modeling and integration: principles and applications*. London: Chapman and Hall.

Vernadat, F.B., 1999. *Techniques de modélisation en entreprise: applications aux processus opérationnels*. Paris: Economica.

WfMC, 2008. *XPDL 2.1. Process definition interface – XML process definition language* [online]. WFMC-TC-1025. Available from: http://www.wfmc.org/xpdl.html [Accessed 31 August 2011].

Zur Muehlen, M. and Recker, J., 2008. How much language is enough? Theoretical and practical use of the business process modeling notation. *In*: Z. Bellahsene and M. Leonard, eds. *Lecture notes in computer science*. Montpellier, France: Springerlink, 465–479.