Towards a pivotal-based approach for business process alignment

Jean-Philippe 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 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 2. MDA and BPM, adapted from Model-driven.org².

Figure 3. MDA and BPM.
standards are required for its use. The methodology is generic. However, specific languages and models are considered like a model (Mueller and Kienle 2010). This synchronisation is not contained in other, in particular with BPA models. A possible approach is to take of abstraction. Each model may have information that can be modified, in spite of the strong intrinsic reciprocal influence existing between the two models. That defines the underlying interaction between them. That defines the underlying relationship 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 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).

4.1. From bidirectional transformations to the notion of pivot
Consider \( \text{MM}_{\text{BPA}}, \text{MM}_{\text{BPI}}, \text{metamodels and } m_{\text{BPA}}, m_{\text{BPI}}, \text{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:}

\[
f : \text{MM}_{\text{BPA}} \rightarrow \text{MM}_{\text{BPI}} \text{ iff } \text{MM}_{\text{BPI}} \rightarrow \text{MM}_{\text{BPA}}
\]

And we obtain:

\[
\forall m_{\text{BPA}} \in \text{MM}_{\text{BPA}}, \exists m_{\text{BPI}} \in \text{MM}_{\text{BPI}}, f(m_{\text{BPA}}) = m_{\text{BPI}}; f^{-1}(m_{\text{BPI}}) = m_{\text{BPA}}
\]

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 particular 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 \( \tau_{\text{BPA}} \) and \( \tau_{\text{BPI}} \) two transformations and \( \text{MM}_{\text{Int}} \) an intermediary metamodel:

- \( \tau_{\text{BPA}} : \text{MM}_{\text{BPA}} \rightarrow \text{MM}_{\text{Int}} \)
- \( \tau_{\text{BPI}} : \text{MM}_{\text{BPI}} \rightarrow \text{MM}_{\text{Int}} \)

Models \( m_A \) and \( m_B \) are considered as equivalent if and only if:

\[
\tau_{\text{BPA}}(m_{\text{BPA}}) = m'_{\text{BPA}}
\]

So if we consider that \( \text{MM}_{\text{BPA}} \subseteq \text{MM}_{\text{Int}} \) and \( \text{MM}_{\text{BPI}} \subseteq \text{MM}_{\text{Int}} \), we obtain the following relationship:

\[
m_{\text{BPA}} \xrightarrow{\tau_{\text{BPA}}} m'_{\text{BPA}} \equiv m'_{\text{BPI}} \xrightarrow{\tau_{\text{BPI}}} m_{\text{BPI}}
\]

The bijective transformation can be made subjective, without changing the models’ appearance perceived by the user, and \( m_{\text{BPA}} \) and \( m_{\text{BPI}} \) remain unchanged. Our pivotal approach starts with this new equivalence and defines these transformations \( \tau_{\text{BPA}} \) and \( \tau_{\text{BPI}} \) 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 syntactical 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 syntactical 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.

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 \(m_{\text{Pivot}}\) stores specific data form the input model (here \(m_{\text{BPA}}\)). The \(m_{\text{Pivot}}\) also provides data with their default values used by the output model (here \(m_{\text{BPI}}\)). 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
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:
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.

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):

\[ \text{MM}_\text{rep} \mu \text{MM}_{\text{BPA}} \]  

\[ \text{RefMet} \mu \text{MM}_{\text{BPA}} \]  

In a same way, the BPA model, m_BPA, has to conform to (η) MM_rep and RefMet (Figure 8). Thus, we obtain the following relation:

\[ m_{\text{BPA}} \equiv \text{MM}_{\text{BPA}} \]  

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

\[
m_{\text{BPA}} \in L(MM_{\text{BPA}});
\]

\[
m_{\text{BPI}} \in L(MM_{\text{BPI}}).
\]

We obtain the following:

\[
m_{\text{BPA}} \not\equiv m_{\text{BPI}} \iff f_{\text{C}}(m_{\text{BPA}}) \not\equiv f_{\text{C}}(m_{\text{BPI}})
\]

(10)

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

\[
\exists m_{\text{pivot}} \in L(MM_{\text{pivot}}), f_{i}(m_{i}) = m_{\text{pivot}}
\]

(11)

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

### 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.

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).

<table>
<thead>
<tr>
<th>#</th>
<th>Class</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Node</td>
<td>Event</td>
</tr>
<tr>
<td>02</td>
<td>End</td>
<td>Start</td>
</tr>
<tr>
<td>03</td>
<td>Action</td>
<td>Task</td>
</tr>
<tr>
<td>04</td>
<td>Activity</td>
<td>Sub-process</td>
</tr>
<tr>
<td>05</td>
<td>Edge</td>
<td>Process</td>
</tr>
<tr>
<td>06</td>
<td>Logical</td>
<td>Exclusive</td>
</tr>
<tr>
<td>07</td>
<td>Inclusive</td>
<td>Inclusive</td>
</tr>
<tr>
<td>08</td>
<td>Parallel</td>
<td>Parallel</td>
</tr>
<tr>
<td>09</td>
<td>Pool</td>
<td>By default</td>
</tr>
<tr>
<td>10</td>
<td>Association</td>
<td>Link</td>
</tr>
<tr>
<td>11</td>
<td>Swimlanes</td>
<td>Uncontrolled</td>
</tr>
<tr>
<td>12</td>
<td>Lane</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Artifacts</td>
<td>Data object</td>
</tr>
<tr>
<td>14</td>
<td>Annotation</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Simple class set.
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 metamodeling 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).

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_{ep}, 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_{ep}, 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 visitElementName() 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
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 MM_{Pivot} (c) as the transformation t2 from an Enterprise_Simple model to a MM_{Pivot} (d) (Builder/linker method).

The transformation t2 of m_{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, m_{Pivot}, contains all data belonging to the analysis model m_{BPA}. It also contains elements necessary to establish complete and comprehensive implementation model m_{BPI}. Future works will demonstrate how modifications from a m_{BPI} to a m_{BPA}.
neither enhanced nor updated: studied before. Several issues arise if these models are 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 guarantees 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 webservices. 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).
3. RNTL FAROS project (a composition environment for reliable service-oriented architectures): http://www.lifl.fr/faros

References


