



HAL
open science

A pivotal-based approach for enterprise business process and IS integration

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

► To cite this version:

Jean-Stéphane Ulmer, Jean-Pierre Belaud, Jean-Marc Le Lann. A pivotal-based approach for enterprise business process and IS integration. *Enterprise Information Systems*, 2012, 7 (1), pp.61-78. 10.1080/17517575.2012.700326 . hal-03608920

HAL Id: hal-03608920

<https://hal.science/hal-03608920>

Submitted on 15 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: <http://oatao.univ-toulouse.fr/>
Eprints ID: 8292

To link to this article: DOI:10.1080/17517575.2012.700326
<http://dx.doi.org/10.1080/17517575.2012.700326>

To cite this version : Ulmer, Jean-Stéphane and Belaud, Jean-Pierre and Le Lann, Jean-Marc *A pivotal-based approach for enterprise business process and IS integration*. (2013) Enterprise Information Systems, vol. 7 (n° 1). pp. 61-78. ISSN 1751-7575

Any correspondence concerning this service should be sent to the repository administrator: staff-oatao@inp-toulouse.fr

A pivotal-based approach for enterprise business process and IS integration

Jean-Stéphane ULMER^{a,b}, Jean-Pierre BELAUD^{a,b}, Jean-Marc LE LANN^{a,b}

^a Université de Toulouse ; INPT-ENSIACET ; 4, Allée Emile Monso, F-31432 Toulouse, FRANCE

^b CNRS ; Laboratoire de Génie Chimique ; F-31432 Toulouse, FRANCE

jeanstephane.ulmer@ensiacet.fr, jeanpierre.belaud@ensiacet.fr, jeanmarc.lelann@ensiacet.fr

A pivotal-based approach for enterprise business process and IS integration

A company must be able to describe and to react against any endogenous or exogenous event. Such flexibility can be achieved through Business Process Management (BPM). **Nevertheless a BPM approach highlights complex relations between business and IT domains.** A non-alignment is **exposed** between heterogeneous models: this is the “business-IT gap” as described in the literature. Through concepts from Business Engineering and Information Systems driven by models and IT, we define a generic approach ensuring **multi-view** consistency. Its role is to maintain and provide all information related to the structure and semantic of models. Allowing the full return of a transformed model in the sense of reverse engineering, our platform enables synchronization between analysis model and implementation model.

Keywords: operational alignment, functional integration, business process management

1. Introduction

The alignment between business strategy and the Information Technology (IT) exploitation have become an important business concern as it improves companies' agility. Indeed, to reach new opportunities, an enterprise needs business-domain flexibility. This flexibility is reached when using processes supported by IT which improves enterprise's agility. The growing importance of alignment between business strategy and IT has become, in twenty years, a major issue in management. If the interest of achieving such alignment is well-known, its implementation is in general limited. It is frequent that organization's actors ignore what alignment is for and how to obtain it.

1.1. Problematic

Alignment' definition and application, both between strategic (enterprise and IT management) and operational (processes and Information Systems) levels, are not well-defined (Avison et al. 2004). This lack of formal definition leads to query on its major steps and objectives.

Another difficulty encountered when applying alignment, is to consider unsynchronized lifecycles between applications and technologies. Organizational business needs distinguished themselves by their constant evolution. Their developments are due to a changing market, acquisitions and mergers between companies, implementation of new strategies and priorities in line with investors. The requirements related to business changes are unpredictable. In the opposite, IT associated products (for example Web-services or J2EE) are updated once to twice a year. And major tendencies change every two to five years. Thus, lifecycles are clearly different depending on business needs, IT requirements, related projects and products. This desynchronization creates a discontinuity between the as-wish process, the company process, and the as-is process, the technical process implemented by the information system. It reinforces the disjunction existing between business and IT domains. These domains are considered as two orthogonal viewpoints **(they consider different concerns but both address the process properties)**, as described by (Dijkman et al. 2008).

Finally, the last challenge when considering alignment is a corollary of the two previous ones: its conservation. In order to keep an operational alignment, semantic and structural consistency efforts have to be realized between heterogeneous models. **This operational alignment is also defined as the functional integration into by (Henderson et Venkatraman 1999) Strategic Alignment Model (SAM).**

During the manipulation of models from different levels of abstraction and/or granularity, a non-alignment is **revealed**: the business-IT gap as described in the literature.

1.2. Objectives

In this paper, we describe how we manage the “discontinuity” between the business and IT perspectives, the “business-IT gap”, as presented by (Ulmer 2011). To improve the current alignment strategies, we suggest a generic approach for modelling and implementing processes and establish a platform that supports such an approach. This approach allows the control of business processes, from their modelling to their implementation within an Information System (IS). Inspired by concepts from model-driven business and IS engineering, the defined approach ensures consistency between heterogeneous models by using a pivotal metamodel and model. Pivotal model’s role is to maintain and provide all information related to the structure and semantics of models. This intermediate format is considered as a reference model (Fettke et al. 2006), necessary to store and exchange model information between modelling environments and implementation.

This paper is organized as follows: Section 2 introduced some concepts used into our approach. In the Section 3, we discuss and develop the core of this approach: the pivotal model and its metamodel. Section 4 briefly demonstrates an application of this approach on a production process. Finally, the Section 5 concludes this paper and presents some future works.

2. Models, metamodels and transformation

In this section, we concisely present some underlying concepts and technologies related to our approach. We highlight differences between business and IT models and how an

operational alignment can be achieved.

2.1. Heterogeneous models

A business process can be interpreted by two different views during its lifecycle: a business view and an IT view. A modelled process must be understandable by both business actors and IT ones. Therefore, a business process is supported by two types of models during its lifecycle: analysis and implementation models.

Through the business view, the process is represented using business terms, generally expressed in natural language, as for example a BPMN¹ diagram (Figure 1 – analysis model). In most cases, it is a graphical representation which facilitates communication between actors in a project. The resulting analysis model respects conventional, but not necessarily formal, business rules. In contrast, IT view provides a technical interpretation of the process, like BPEL² (Figure 1 – Obtained implementation model). The resulting implementation model is precisely defined in a given context. It also meets requirements specified by developers and end-users. This allows the model to be correctly implemented on a target business execution engine.

[Figure 1 near here]

Figure 1. Domains, models and transformations

Each of those models offers a very distinct view and is manipulated by different actors. The goal of an analysis model is to represent a graphical aspect and flows of the process. Business analysts do not systematically try to formalize the various elements of the model with a well-known “technical” logic as we do with an implementation model. The representation of the analysis model may be inaccurate, ambiguous and subject to interpretation from an IT perspective. This is typically a non-formal business model

¹ Business Process Modelling Notation (Object Management Group 2009)

² Business Process Execution Language (OASIS 2007)

used for documentation and communication between actors: a “contemplative” model. The implementation step that follows is all the more complex. Thus, when the IS evolves or is changing, analysis models are no longer updated or in sync with implementation models. In addition, the obtained model from analysis model transformation may presents irregularities and requires modifications by IT experts. In the end, the expressed semantic differs according to models. The lack of synchronization and absence of semantic equivalence between models prevent us from obtaining what will call an “intermodel” consistency. This is the famous Business-IT gap found in the literature (Peppard et Ward 1999), (Grembergen 2004), (Stein et al. 2009), the gap between business and IT domains.

2.2. *Models and transformation*

During the transformation from an analysis model (BPMN) to an implementation model (BPEL), shown on Figure 1, we observed that several changes occur. First, we should note that graphical aspects of the BPMN model are lost: there is not graphical data expressed in the BPEL code (Figure 1 – Obtained implementation model). Then, the obtained BPEL model may require adjustments. For example, the variables contained within the activities A and B may need to be reported as assigned to parameters of web-services. As an example the information needed to execute the process model are added and represented in the implementation model, here the “PartnerLink” (Figure 1 – Modified implementation model).

During this “simple” transformation, data are lost or modified. However, an operational alignment means that a reverse transformation is possible. For this, we consider three properties to be achieved: models have to be synchronized, semantically equivalent and consistent with each other. Following paragraphs describe those properties.

2.2.1. Synchronisation

We consider two models, i and j to be synchronized if and only if significant changes made to a model i can be notified on a model j . Changes affecting the structure and / or behaviour of a model are considered to be significant. In Figure 1, we precise how the various activities are performed by the implementation model: these details alter the implementation model's behaviour. And these changes are not transcribed into an analysis model. Thereby, in this example, the models are no longer synchronized.

2.2.2. Semantic equivalence

(Guarino 1995) defines equivalence as follows: two concepts are equivalent if they represent a single concept. As it remains difficult to compare the semantic equivalence of syntactically different models belonging to different abstraction levels, we propose our own definition of semantic equivalence. Consider the models i and j and the respective set of elements constituting these models, E_i and E_j . The models i and j are semantically equivalent if for each element belonging to E_i , an element (or group of elements) belonging to E_j can be associated. And each of these associate elements follows the same orchestration³.

In the Figure 1 example, the models are semantically equivalent: the process flow remains the same, whether in the analysis model or the implementation model. However, models are no longer synchronized. A major change on the implementation model would not be passed on to the analysis model, making it obsolete and equivalence between models would not be longer maintained.

³ Orchestration defines the whole internal steps of a process, including conditions and exceptions. In our case, it can be assimilated to the control-flow of the process.

2.2.3. Intermodel consistency

For business analysts the process represented by the BPEL model may look different from the one defined by BPMN language. Indeed it may be complicated for business analysts to “trace” a BPMN model by reading lines of code of a model serialized in XML. However, BPMN or BPEL models can express the same thing. Here the term “representation” seems inadequate. Thus, we replace it by “intermodel coherence”, term defined by (Ulmer et al. 2011). We assume that such a link exists between two models if we observe a semantic equivalence and synchronization between them. We consider this intermodel coherence as a necessary and sufficient condition of an operational alignment as defined by (Henderson et Venkatraman 1999). By following the three definitions presented here, the proposed approach enhances the alignment between business & IT domains, due to a pivot.

Based on these properties, we detail the different elements of such an approach.

2.3. Transformation and metamodels

Transformation rules between analysis and implementation models are generally defined on a case by case basis, which made them less flexible. To overcome this issue, the Model-Driven Engineering (MDE) (Kent 2002), recommends the use of metamodels as support for model transformation. Using metamodels eases transformations and makes them more general and systematic. Furthermore utilization of metamodels provides a semantic and rigorous semantic enrichment for models of the company. A wide range of MDE tools has been developed to define these transformations. Some tools use model-oriented syntax: metamodeling tools. In this case, the model transformation is the execution of a metaprogram. It is able to manipulate model and metamodel inputs and outputs through a reflexive language (Seidewitz 2003). Kermeta

(J.-M. Jézéquel et al. 2011)⁴ is an example of those languages we use in our approach. The Kermeta language is an EMOF (Essential Meta-Object Facilities) extension, developed by the Triskell team. Kermeta is able to define operational and denotational semantic of metamodel as structure behaviour. Thus Kermeta can describe relations between different types of elements belonging to different formalisms (Muller et al. 2005). Also Kermeta defines dynamic linking and exception management mechanisms and conventional control structures with an imperative language. For Business Process Management approaches, analysis and implementation metamodels are generally neither explicit nor formalized. Accordingly, in our approach, we specially describe/define the metamodels helped with Kermeta and Ecore tools⁵. Figure 2 represents our “ideal” architecture for model transformation, based on MDE.

[Figure 2 near here]

Figure 2. “Ideal” transformation

A bidirectional transformation between analysis and implementation models can mathematically justifies model consistency. Therefore, these two models would be considered equivalent and establishing intermodel coherence. In the following section, we explain why obtaining this consistency is quite difficult. We also demonstrate the need of a reference model, called here the pivotal model, for better interoperability between different domains.

2.4. Bidirectional model transformation and notion of pivot

In the remainder of the paper, we use the following notations (Table 1):

[Table 1 near here]

Table 1. Notations

⁴ <http://www.kermeta.org/>

⁵ Ecore tools component provides a complete environment to create, edit and maintain metamodels on the Eclipse Modeling Framework: <http://www.eclipse.org/modeling/emft/>

Consider the bidirectional transformation f :

$$f : MM_A \mapsto MM_B \text{ and } f^{-1} : MM_B \mapsto MM_A$$

To obtain equivalence between models m_A and m_B , transformation f has to be a bijective-type:

$$\forall m_A \in MM_A, \exists! m_B \in MM_B, f(m_A) = m_B, f^{-1}(m_B) = m_A$$

In other words, for each model m_A compliant to its metamodel MM_A , a unique model m_B compliant to MM_B exists such as f transforms m_A to m_B and inversely. This transformation is too restrictive and impossible to perform if cardinalities between models are different (Stevens 2008). In business approaches, models from business and IT domains remain heterogeneous and belong to different abstraction levels. Thus their cardinalities are different, the transformation f is impossible to resolve. One approach is to modify one of those models and to convert it so it can consider all elements, concepts and information that models can represent. Let τ_A and τ_B be two transformations and MM_{Int} an intermediary metamodel:

- $\tau_A : MM_A \mapsto MM_{Int}$

$$\tau_A(m_A) = m'_A$$

- $\tau_B : MM_B \mapsto MM_{Int}$

$$\tau_B(m_B) = m'_B$$

Models m'_A and m'_B are considered as equivalent, as defined section 2.2.2, if and only if:

$$m'_A \equiv m'_B$$

To reach such a result, we consider that $MM_A \subseteq MM_{int}$ and $MM_B \subseteq MM_{int}$.

The bijective transformation can become a surjective one. By doing so, we do not modify how models are perceived by users: m_A and m_D remain unchanged. Our pivot approach intervenes at the following equivalence relation:

$$m_A \xrightarrow{\tau_A} m_A^e \quad m_D^e \xleftarrow{\tau_B} m_D$$

Based on the functions of constructive compliance τ_A and τ_B , the resulting model is the pivotal model. **The term “constructive compliance” is described section 3.1.**

3. Pivotal approach

The pivot role is to ensure semantic equivalence between a conceptual model and the corresponding implementation model. Our approach considers that the pivot metamodel should facilitate the transformation between models of different perspectives and abstraction, in accordance to MDE principles. The pivot model must be able to preserve information integrity and consistency. So this intermediate format is necessary to store and exchange model information between analysis and implementation environments⁶.

3.1. Functions of constructive compliance

Those functions guarantee the transformation from m_A (A for analysis), or m_I (I for implementation), to a pivotal model: m_{pivot} . Its behaviour is to ensure semantic equivalence between analysis and implementation models.

We establish functions of constructive compliance (Favre et al. 2006), fc_A and fc_I respectively define from MM_A and from MM_I to MM_{pivot} . By definition, a function of constructive compliance (fc) defines a set of operations *conform* such as:

⁶ An environment is constituted by actors, tools ... needed to design and/or manipulate a target model.

$$\forall m_x \in L(MM_x) \text{ conform}(m_x) \text{ is true}$$

A fc also defines equivalency relations between models according to their metamodels. Let $m_A \in L(MM_A)$ and $m_I \in L(MM_I)$, then:

$$m_A \text{ s EMBED Equation. 3 s } m_I \\ \Leftrightarrow fc_A(m_A) = fc_I(m_I)$$

The « s » means the relation is considered from a semantic point of view. In the end, fc s allow us to obtain the pivotal model:

$$\exists m_{pivot} \in L(MM_{pivot}), fc_x(m_x) = m_{pivot}$$

Thank to these functions, we deduce the existing link between analysis and implementation environments, according to their respective models m_A and m_I , considered as semantically equivalent. The figure 3 positions the different models, metamodels and relations linking them. We can observe that the pivotal model is a necessary and a reference model for an enhanced interoperability between heterogeneous environments and their respective models:

$$m_A \xrightarrow{fc_A} m_{pivot} \xleftarrow{fc_I} m_I$$

Note – Even if $fc_x(m_x) = m_{pivot}$ with $m_x \in L(MM_x)$ and $m_{pivot} \in L(MM_{pivot})$, we still have $MM_x \neq MM_{pivot}$. Indeed, each of these metamodels uses different views and considers different aspects of the process.

[Figure 3 near here]

Figure 3. Environments, metamodels and models relations

These three different aspects of the process and their related environments require specific skills to be fully exploited. Hence we consider a third actor is needed to perform a Business Process Management (BPM) lifecycle: the process architect.

3.2. *Actors and pivotal model*

We can easily identify two major players needed to perform a typical BPM lifecycle: the business analyst and IT expert. A business analyst is looking for new ways to improve “business” efficiency of his company. The IT expert transforms those business requests into IT operations. The relation between business requirement and IT requirements created during this transformation is a first step to the business-IT alignment (Engelsman et al. 2011). The figure 4 shows how data are transferred between business analyst and IT expert during a model transformation. We can distinguish three kinds of data:

- (a) Useful data for both models;
- (b) Data lost during the transformation because unusable for the target model;
- (c) Data informally created in order to get a functional target model.

As example, we can identify these data with the ones observed in section 2.1. A description of activities is present on both models; this is a (a) type data. Graphical data from the analysis model are not transcribed on the implementation model; these are (b) type data. Some information is added on the implementation model without consulting or notifies the business analyst; this is (c) type data.

[Figure 4 near here]

Figure 4. Typical BPM roles and data

In previous sections we have demonstrated that the use of a pivotal model can enhance intermodel coherence and thus enhance operational alignment. For the associated

approach, we consider that a third actor is necessary: the process architect. By combining the two levels of abstractions specific to the business analyst and IT expert, the “pivot” actor allows a better consistency between models. The role of the process architect is to determine what data are useful for both input and output models; these are the useful data (a). He must be able to provide the needed information to obtain a complete output model, by providing (c) type data. Finally, it should allow the preservation of the integrity of information within a pivotal model, especially by storing unused data (from the “output model” point of view): the (b) type data. As depicted by (Millet et al. 2009), the process architect brings the requisite “knowledge” level to manage and associate both design concepts and operational tools. The figure 5 gives a useful representation of all those data and roles.

[Figure 5 near here]

Figure 5. Pivotal approach’s roles and data

The example given in Figure 5 illustrates this pivotal approach, the passage of a conceptual process model to an implementation platform. In order to test this approach, we use our own developed framework: the SCALP⁷ platform.

3.3. Pivotal metamodel

Before presenting an implementation of the approach, we describe our pivotal metamodel and its elements. These elements are chosen according to two criteria: their use frequency in process models and their affiliation to the BPMN standard conformity classes. Thus, the pivotal metamodel is constituted of 18 elements, as shown Figure 6. They represent the whole BPMN Common Core set and the most of the BPMN Extended Core set. This is mainly the most used in companies, as demonstrated by (Zur

⁷ SCALP : Solution pour la Cohérence et l’ALignement des Processus, in french, which means « Solution for Process Consistency and Alignment ».

Muehlen et Recker 2008). For more clarity, these elements are categorized into five groups and are reported in Table 2.

[Table 2 near here]

Table 2. Elements of the pivotal metamodel

As BPMN standard is well-known, we won't give an exhaustive presentation for each of its elements. Instead, we focus on a particular class: *BPElement*.

BPElement class considers information that do not influence m_A or m_I , but deemed necessary for process reverse engineering purpose.

It keeps data relating to business concepts expressed in m_A but not considered in m_I , through its related attribute: *GraphicalAttributes*. Similarly in the reverse transformation, data unused by business analyst is stored through the attribute *ImplAttributes*. *BPElement*'s attributes are defined by the process architect according to the recommendations of business analyst and IT expert (Table 3). The presence of these two actors is required to determine useful and common data of both models but also data specific to each model.

[Table 3 near here]

Table 3. *BPElement*'s attributes

[Figure 6 near here]

Figure 6. Pivotal metamodel

4. SCALP platform

The pivotal approach we propose remains generic and independent of tools or technologies. However specific languages and standards are required for its application, as presented Figure 7. Indeed to test and validate our concepts and approach, a development project has been realized. The obtained dedicated platform is based on the three environments considered in our approach. In the next sections, we describe the three environments of this SCALP platform and how transformations operate to get an

operational alignment enhancement.

4.1. SCALP environments

The analysis environment, where a conceptual model is designed by a business analyst, is based on the modelling tool Intalio Designer 6.0.3⁸. This editor allows process modelling using a palette of BPMN symbols. This tool is quite respectful to the BPMN 1.2. language specification. Intalio Designer generates two usable files in XML format. The first file describes the logic of the process; the second contains the process graphical data. These two files are the starting point of our framework.

The implementation environment includes the target execution platform. Within this environment, the IT expert modifies obtained input models in order to make them executable. Here we use OpenERP 5.0.1.4⁹ as a process engine. An OpenERP module is structured by a folder containing files in python (.py) and XML format. Python files describe the ERP module and specific classes, as for example associated forms with the module. They also define the structure of the interface. XML files provide a description of the activities, sequencing of the module and its interface.

The last considered environment is the pivotal environment containing the pivot platform. This platform has two distinct features: the transformation of a model from one environment (the input model) to another (the output model) and the preservation of information integrity contained in the input model. To instantiate the pivot platform, we use EMF 1.4.0 (Eclipse Modelling Framework) from the IDE (Integrated Development Environment) Eclipse 3.4.2¹⁰. The objective of EMF is to manipulate models instead of behaviour by using provide sets of tools. The metamodeling part is defined using Ecore

⁸ <http://www.intalio.com/bpms/designer>

⁹ <http://www.openerp.com/>

¹⁰ <http://www.eclipse.org/>

tools from EMF. And it is completed by using the open-source metamodeling platform Kermeta 1.3.0. One of the key features of Kermeta is its ability to extend an existing metamodel with constraints, new structural elements (meta-classes, classes, properties and operations) or set of features from other languages. This is the aspect weaving (Klein et al. 2007). This weaving adds code (a single class or a component of a specific language) in a target metamodel without changing its structure, by using the Visitor pattern (Gamma et al. 2000).

[Figure 7 near here]

Figure 7. SCALP framework

Table 4 summarizes software applications used by our dedicated platform. There is not specific compatibility between them and it provides heterogeneous models.

[Table 4 near here]

Table 4. Applications and technologies used by SCALP

4.2. Mappings and transformations

Figure 8 illustrates the required operations for the passage from a conceptual model derived from the Intalio Designer modelling tool to a technical model and its corresponding OpenERP module.

[Figure 8 near here]

Figure 8. Detailed SCALP framework

m_1 and m_2 mappings represent operation mappings that we perform from the analysis metamodel, MM_A , to the pivotal one, MM_{Pivot} , and vice versa. Similarly, m_3 and m_4 mappings define mapping from MM_{Pivot} to the implementation metamodel, MM_I . The different mappings are done using Kermeta, where the Process Architect associates similar semantic elements (Rizopoulos et McBrien 2005), (Izza 2009).

The t_1 transformation combines the two files from Intalio Designer: “Logic Description” and “Graphical data” files. Their combination provides our analysis model, m_A . Conversely, t_1' splits information contained in m_A depending on whether the information are for “Logic Description” or “Graphical data” files. The m_1 mapping occurs after the use of a Visitor pattern which determines and combines elements of analysis and pivot metamodels. This allows the framework to perform the t_2 transformation, from m_A to the pivot model m_{Pivot} , using a Builder-Linker method. We operate the same way to transform m_{Pivot} to m_I (m_3 - t_3), or for reversal operations (m_2 - t_6 and m_4 - t_5). The t_4 transformation is atypical compared to other transformations. This transformation from m_I provides an OpenERP module files. This transformation is semi-automated as its inverse (t_4').

NOTE. — WE consider that a business process composed of n-pools is equivalent to n-OpenERP modules. In order to simplify the case-study shown in this paper, this example treats the case of a business process containing a single pool. Furthermore, we focus on control-flow oriented business processes. This allows us to obtain a description of a sequencing of activities within the ERP module.

4.3. Application of SCALP: typical cases

During our research works, several cases were experienced to test the usefulness of this approach. To achieve this, our platform is based on EMF and uses Kermeta language. Kermeta defines behaviour rules of our metamodels and uses the Visitor pattern. In this paper we present two of these case-studies. Before describe them, we have to keep in mind that the SCALP platform is clearly defined as for sets of transformations and mappings presented in the previous section.

4.3.1. Model transformation with a blank pivotal model

During a first transformation between analysis and implementation models, we start with a blank pivotal model: it does not contain any data. After importing the input model into our platform (transformation t_1 or t_4' depending on whether the input model is from Intalio Designer or OpenERP), a Kermeta program (t_2 or t_5) is performed from the obtained XMI model to get m_{Pivot} . Because mappings (m_1 - m_2 and m_3 - m_4) have already been done, this transformation can be realized, using the visitor paradigm. This Visitor file determines elements (and their attributes) belonging to the input metamodel. The transformation file associates them to the MM_{Pivot} components (elements and attributes).

At this point, the generated pivotal model is constituted by three different sets of data. It contains data from the input model: useful data for both models and specific data only usable by the input model and stocked for further use. The third kind of data has default value and is necessary in order to accomplish the next transformation t_3 (or t_6). Information they provide are required to make the output model fully exploitable by the target environment.

The t_3 (or t_6) transformation has the same mechanisms as the t_2 (or t_5) transformation. It requires a XMI model as input, m_{Pivot} , and provides a XMI model as output. From this model, we form the associated OpenERP module or Intalio diagram (t_1' , t_4).

4.3.2. Model modification and transformation, pivotal model already created

The OpenERP module and Intalio diagram obtained in the previous step may be modified by related actors and/or end-users. Domain-specific modifications can be done. For example, graphical aspect of the business diagram could be modified; logical

functions could be added into the ERP's module. But modifications impacting both models can also be made, for example structural modifications that result in a change of the control-flow. Then transformations are realized in the same way we described in the previous section, by using Kermeta and the visitor pattern paradigm.

However, two significant differences can be noticed. The first difference appears when we provide additional information to the pivotal model. This information, from the input environment, is notified through the adequate *BPElement*'s sub-element: *GraphicalAttributes* or *ImplAttributes*. If data already existed, they are replaced (deleted and updated). Therefore, by recovering data contained in sub-elements from the previous m_{pivot} , we generate a new pivotal model containing both specific data from m_A and m_I . The second difference occurs during t_3 (or t_6). We extract data contained in *BPElement* to generate an output model with information from previous transformations and stored into pivotal model, not with default data. From the obtained model, we form, as before, the associated OpenERP module or Intalio diagram (t_1' , t_4).

4.3.3. Approach's impact

During the first case, we performed a transformation between a conceptual model, a process diagram, and a technical model, an ERP module. At each step of this transformation, the conformity between the XMI model and its metamodel is validated using Ecore. Through this phase, we store the data used by the non-technical model into the pivotal model. This data is given back, in the second case, to obtain a complete model. And as before, data non-usable by the output model are stored by the pivotal model. This model does not require any input of information to be usable.

These different cases demonstrate the process and utility of our approach. It transforms heterogeneous models from different levels of abstractions. It also stores and/or adds needed data during the analysis-implementation transformation (and vice-

versa). The changes being propagated during these transformations and integrity of information being provided, our approach allows synchronization and a semantic equivalence between models. We obtain an inter-model consistency, an operational alignment.

5. Conclusion and future works

Through a process-lifecycle, consistency is difficult to be maintained 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 domain and IT domain 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. We demonstrate how the use of a pivotal model enhances the operational alignment and thus the agility of an enterprise. Its associated metamodel allows to define semantically rich models. We have also defined a new role, essential for the suitable functioning of our approach: the process architect. His role is to have a global vision of business and IT domain of the company. His function made him an essential intermediary for this pivot approach.

Able to represent most of processes encounter in industries, the pivotal model contains information related to heterogeneous enterprise domains and is strictly consistent with the pivotal metamodel. Even through various transformations, we are able to generate an output model which takes in account modification made on the input model. Our two models (input and output ones) are synchronized and in both cases the data are transcribed in the pivotal model: our models remain synchronized and are semantically equivalent. New experimentations are needed to test our approach and its

platform, by using more specific processes from SME or SMI. This will allow us to assess the sufficiency of the pivotal metamodel expressiveness.

Our future works also considers the versioning of the pivotal model. We have shown that when a pivotal model is generated from an existing one, obsolete data is deleted. However, errors can occur when modifying models. Once versioning solutions are can make our approach more flexible.

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.

6. References

- Avison, D. et al. 2004. Using and validating the strategic alignment model. *The Journal of Strategic Information Systems* 13(3), 223–246.
- Dijkman, R.M. et al. 2008. Consistency in multi-viewpoint design of enterprise information systems. *Information and Software Technology* 50, 737–752.
- Engelsman, W. et al. 2011. Extending enterprise architecture modelling with business goals and requirements. In: *Enterprise Information Systems*. 9-36.
- Favre, J.-M. et al. 2006. *L'ingénierie dirigée par les modèles - Au-delà du MDA*. Cachan, France: Hermes - Lavoisier.
- Fettke, P. et al. 2006. Business Process Reference Models: Survey and Classification Bussler, C. J. et Haller, A. éd. *Business Process Management Workshops* 3812, 469-483.
- Gamma, E. et al. 2000. *Design Patterns: Elements of Reusable Object-Oriented Software*. Canada: Addison-Wesley Professional.
- Grembergen, W.V. 2004. *Strategies for information technology governance*. USA: Idea Group Inc (IGI).

- Guarino, N. 1995. Formal ontology, conceptual analysis and knowledge representation. In: *International Journal of Human-Computer Studies*. Elsevier, 625-640.
- Henderson, J.C. et Venkatraman, N. 1999. Strategic Alignment: Leveraging information technology for transforming organizations. In: *IBM Systems Journal*. IBM, 472-484.
- Izza, S. 2009. Integration of industrial information systems: from syntactic to semantic integration approaches. *Enterprise Information Systems* 3, 1-57.
- Jézéquel, J.-M. et al. 2011. Model Driven Language Engineering with Kermeta
Fernandes, J. M. et al. éd. *Generative and Transformational Techniques in Software Engineering III* 6491, 201-221.
- Kent, S. 2002. Model Driven Engineering. In: Butler, M. et al. éd. *Integrated Formal Methods*. Berlin, Heidelberg: Springer Berlin Heidelberg, 286-298.
- Klein, J. et al. 2007. Weaving multiple aspects in sequence diagrams. *Transactions on aspect-oriented software development III* 4620, 167-199.
- Millet, P.-A. et al. 2009. The SCOR model for the alignment of business processes and information systems. *Enterprise Information Systems* 3, 393-407.
- Muller, P.-A. et al. 2005. Weaving Executability into Object-Oriented Meta-Languages. In: *Proceedings of MODELS/UML'2005*.
- OASIS 2007. WSBPEL - Web Service Business Process Execution Language. . Available at: [internal-pdf://OASIS - WS BPEL vers.2.0-1864916224/OASIS - WS BPEL vers.2.0.pdf](http://internal-pdf://OASIS%20-%20WS%20BPEL%20vers.2.0-1864916224/OASIS%20-%20WS%20BPEL%20vers.2.0.pdf).
- Object Management Group 2009. BPMN - Business Process Modeling Notation Specification 1.2. Available at: <http://www.omg.org/spec/BPMN/1.2>.
- Peppard, J. et Ward, J. 1999. « Mind the Gap »: diagnosing the relationship between the IT organisation and the rest of the business. *The Journal of Strategic Information Systems* 8(1), 29-60.
- Rizopoulos, N. et McBrien, P. 2005. A general approach to the generation of conceptual model transformations. In: *Advanced Information Systems Engineering*. 326-341.
- Seidewitz, E. 2003. What Models Mean. *IEEE Computer Society* 20(5), 26-32.

- Stein, S. et al. 2009. Business to IT Transformations Revisited. *Business Process Management Workshops*, 176-187.
- Stevens, P. 2008. A Landscape of Bidirectional Model Transformations. In: *Generative and Transformational Techniques in Software Engineering II*. Lecture Notes in Computer Science. 408-424.
- Ulmer, J.-S. 2011. Approche générique pour la modélisation et l'implémentation des processus. Thesis (PhD). Toulouse, France: Université de Toulouse - INPT ENSIACET.
- Ulmer, J.-S. et al. 2011. Towards a pivotal-based approach for business process alignment. *International Journal of Computer Integrated Manufacturing* 24(11).
- Zur Muehlen, M. et Recker, J. 2008. How Much Language is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. In: *20th International Conference on Advanced Information Systems Engineering (CAISE 2008)*.

| Symbol | Description |
|-----------|---|
| MM_x | Metamodel |
| m_x | Model |
| \equiv | Relation « isEquivalentTo » |
| μ | Relation « RepresentationOf », here considere in its raffined version « specify » |
| χ | Relation « conformTo ». |
| \mapsto | Relation « mapsTo » |

[Table 1]

| # | Element category | Type | # | Element category | Type | |
|----|------------------|---------|-------------|------------------|--------------|------------|
| 1 | Node | Event | Start | Edge | Uncontrolled | |
| 2 | | | End | | Conditional | |
| 3 | | Action | Task | | Default | |
| 4 | | | Activity | | Association | |
| 5 | | | Sub-Process | | Objet data | |
| 6 | | Logical | Exclusive | | Artefact | Annotation |
| 7 | | | Inclusive | | | BPElement |
| 8 | | | Parallal | | | Process |
| 9 | Swimlane | Pool | | | | |
| 10 | | Lane | | | | |

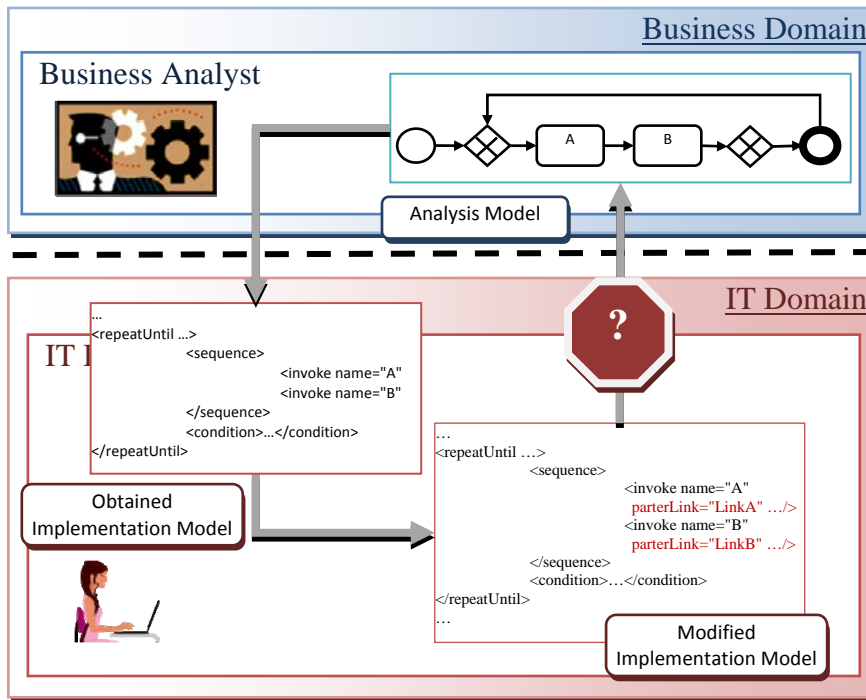
[Table 2]

| Name | Type | Description |
|---------------------|---------------------|--|
| isAGraphicalElement | GraphicalAttributes | Defines graphical elements (coordinates, shape, ...) |
| hasImplInformation | ImplAttributes | Defines needed elements for process execution (import, data type...) |

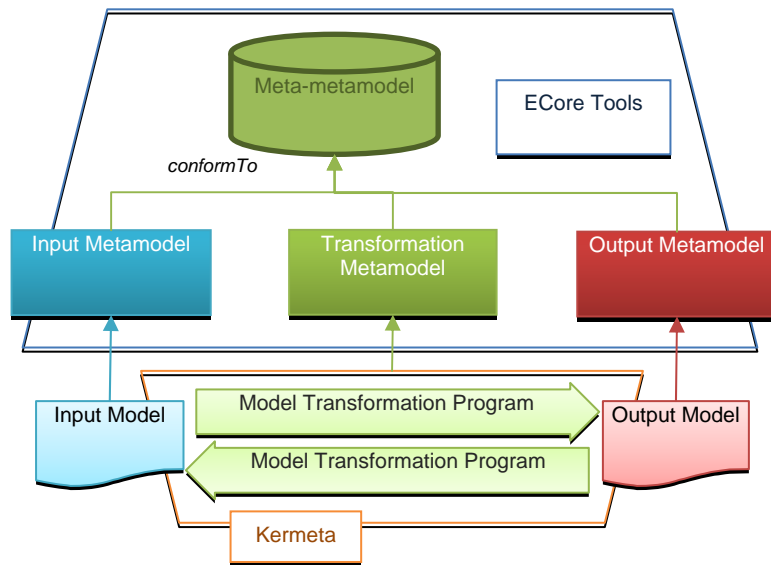
[Table 3]

| | Modelling tool | SCALP | ERP |
|-------------------------|-------------------------|---|-------------------------------|
| Software | Intalio Designer 6.0.2 | Ecore 0.7.0, EMF 1.4.0, XML JDom, Kermeta 1.3.0 | OpenERP 5.0.14 PGAdmin III |
| Technologies, languages | BPMN, XML | XML, XMI, Java | Python, PostgreSQL, XML |
| File formats | .bpmn, .bpmn_diagram | .xmi, .ecore, .java, .kmt | .xml, .py |

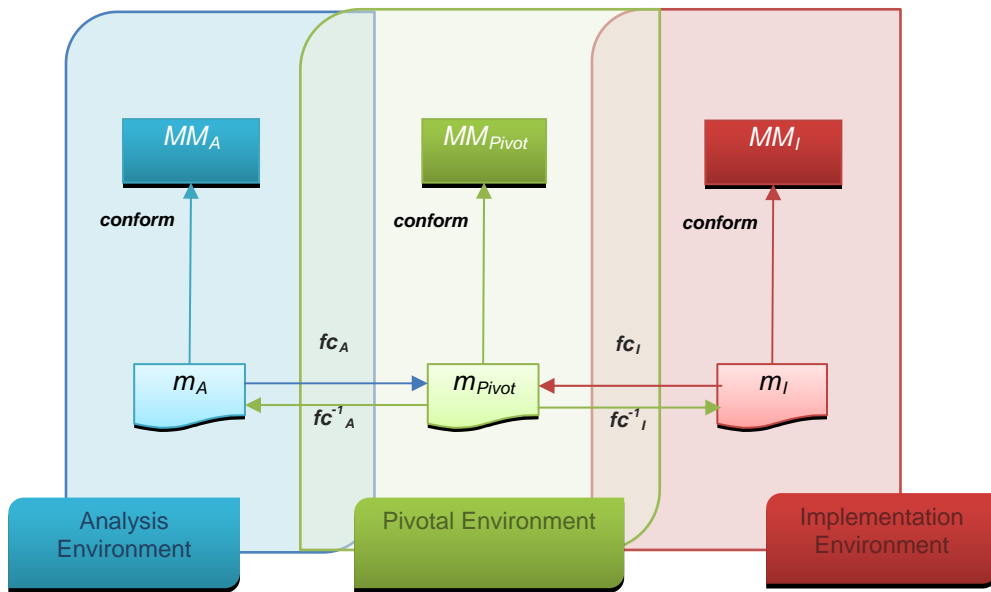
[Table 4]



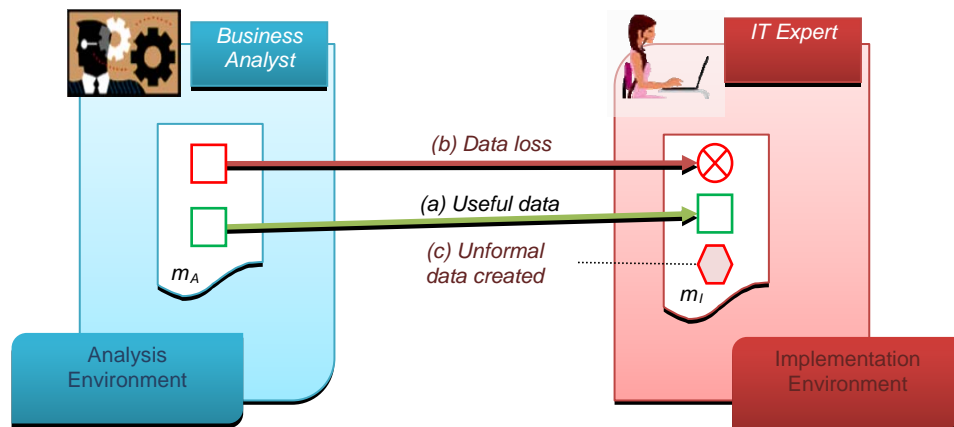
[Figure 1]



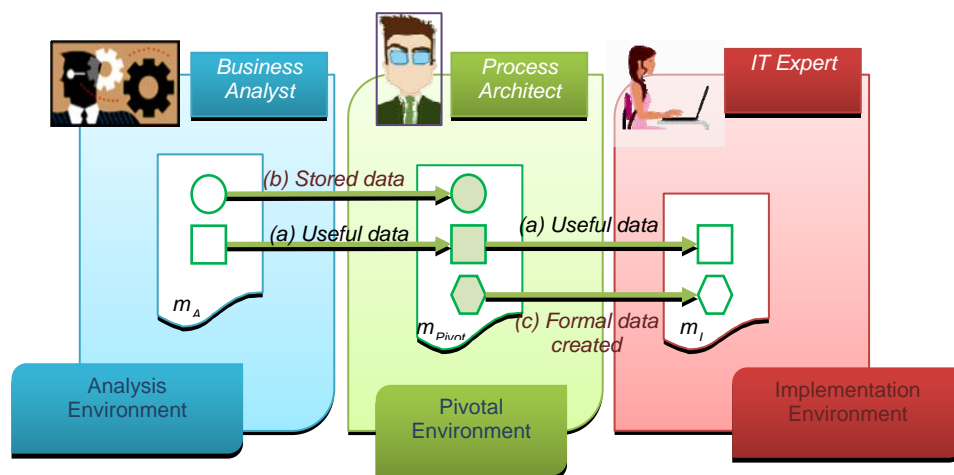
[Figure2]



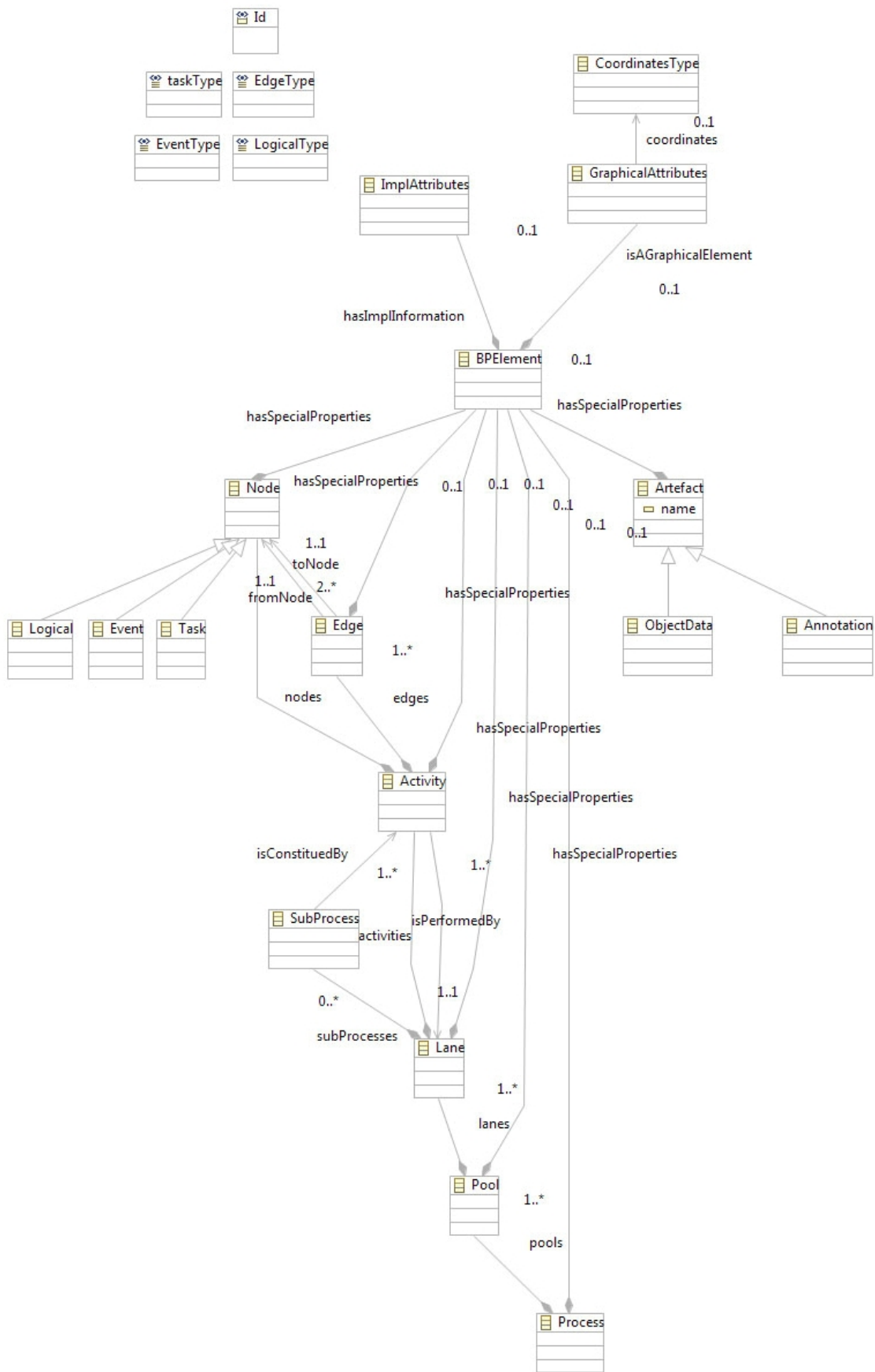
[Figure 3]



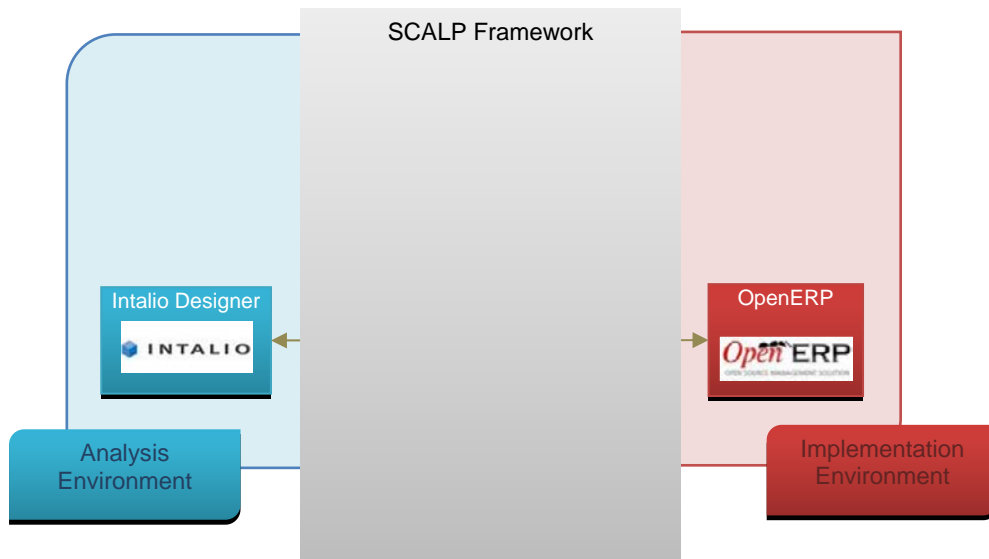
[Figure 4]



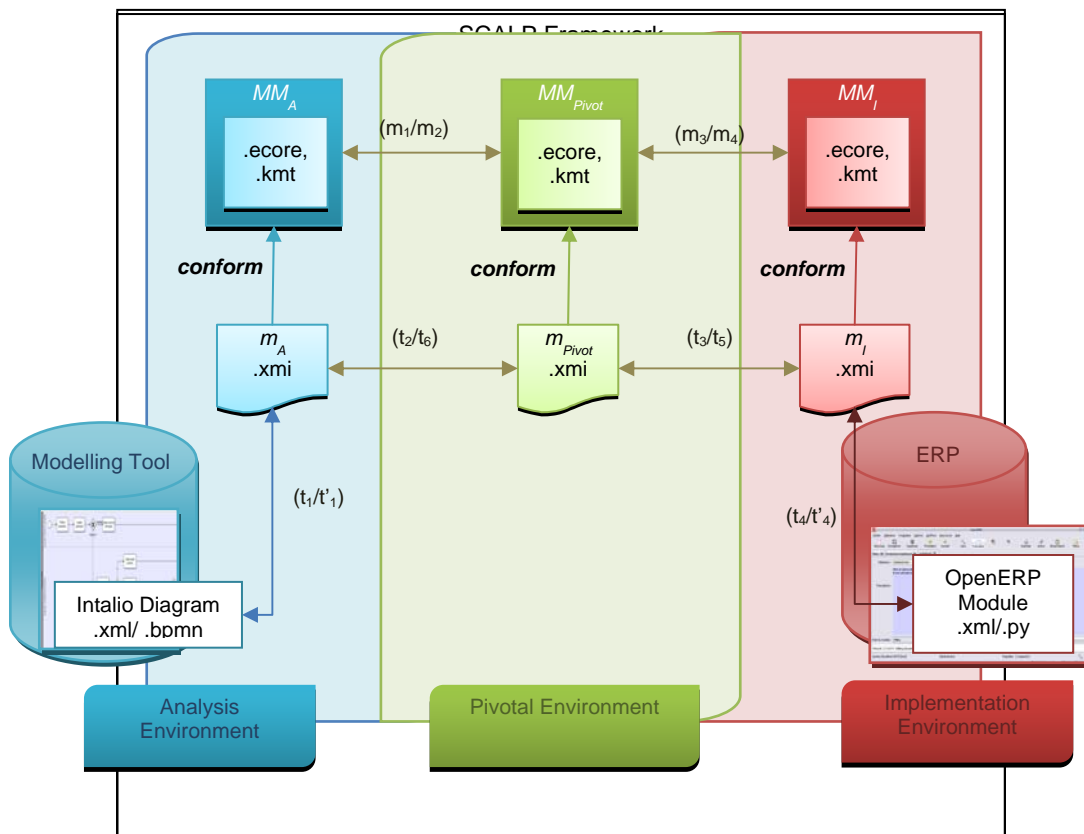
[Figure 5]



[Figure 6]



[Figure 7]



[Figure 8]