A Meta-Model Proposal For Graph Transformation as Part of Business Process Reengineering Projects

Leila Jamel Menzli a , Raja Hanafi b

a College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, KSA

 RIADI Laboratory- ENSI- TUNISIA

lmjamel@pnu.edu.sa

b Faculte des Sciences de Bizerte, TUNISIA

RIADI Laboratory- ENSI-TUNISIA

arbiahanafi@yahoo.fr

**Abstract**: The model-driven engineering (MDE) has several significant improvements in the development of complex systems to focus on a more abstract than conventional programming concern. It is a form of generative engineering in which all or part of an application is generated from the model. Substantial improvements are of highlighted in the development of complex software systems by providing the means to move from one abstraction level to another or from one workspace to another. However, the management models can be cumbersome and costly. To better meet the expectations of users, it is necessary to provide flexible and reliable tools for the automatic management of models and languages dedicated to their transformations. For this reason the transformation of models is at the heart of IDM, indeed several model transformations languages, approaches, and frameworks are generated in recent years specially graph model transformations. The aim of this paper is to propose a meta-model in order to integrate semantic knowledge about graph transformation in Business Process Reengineering Projects.

**Key words** :IDM, Model Transformation, Meta-model, graph transformation, BPR.

# 1 Introduction

In today’s ever-changing world, the three Cs: Customer, Competition and Change itself are forcing companies to continuously improve and innovate in terms of speed, flexibility, quality, service, cost and so on. To reach these improvements, companies are on the lookout for new solutions for their business problems. Since 1990, one of the more successful business initiatives in the world is: Business Process Reengineering (BPR). As defined by Hammer and Champy, BPR is the fundamental rethinking and radical redesign of business process to achieve dramatic improvements in critical, contemporary measures of performance such as cost, quality, service and speed (Hammer & Champy, 1993). Many BPR methodologies are defined in the literature, which share one common task: modeling existing and new company’ business processes (("is-a" modeling) respectively "to-be" modeling). In fact during the BPR life cycle, a lot of time is spent in the analysis of existing business processes (BPs) to identify which ones are functioning the worst, which ones are the most critical and which ones are the most likely to be successfully reengineered (Muthu, Whitman & Cheraghi, 1999). In an other hand, a lot of effort is expended in the proposition and the performance’s evaluation of new business processes.

A business process is a complex entity with a characteristic lifecycle. In our work we consider the approach defined by Weske (Weske, 2007) who organizes the lifecycle in a cyclic structure with logic dependences between the design and analysis, configuration, enactment, and evaluation stages. During these stages, many BP models are provided for existing ones or new defined BP during a given BPR project. Moreover, it is still arduous and confusing for users/modelers to select and choose the adequate and the best business process model(s) in the BPR project in hand (Chen,2004). To ensure the redesign of “as-is” BP models or the design of new BP models, a model transformation technique is required. At the moment much work is done in the area of model transformations (MTs). In the context of Model Driven Engineering, models are the main development artifacts and model transformations are among the most important operations applied to models. The main research interest lies on the technical aspects of MTs, for example transformation languages and verification of model transformations. A number of specialized languages have been proposed, aimed at specifying model transformations. There is also a wide variety of tools available, such as the Eclipse Generative Model Transformer (GMT) framework, the Generic Modeling Environment (GME) and more (Mens & Pieter, 2006). Additionally a class of MT approaches are proposed for a wide variety of purposes: vertical MT, horizontal MT, Endogenous versus exogenous transformations, syntactical versus semantical transformations, tree transformation or graph transformation, etc.

To the best of our knowledge there is no work on supporting model transformations for Business Process Reengineering purposes. The aim of this paper is to propose a meta-model for graph transformation for BPR projects. This proposal tries to help a designer, an analyst or a modeler to choose a particular MT approach, or MT tools that are best suited for his needs. Particularly, semantic knowledge about graph transformation approach are integrated with BPR context.

The paper is organized as follows. Section 2 presents the basic concepts related to BPR and BP modeling. Section 3 describes model transformation principles in general. In section 4 a particular focus on graph transformation approaches features is given. Section 5 puts our work in context of related work by presenting taxonomies and meta-models background survey. Section 6 presents a brief discussion. Section 7 explains the meta-model concepts and features. Finally, conclusion and future directions are ending this paper.

# 2. Business Process Reengineering and Business Process modelling

Nowadays several changing approaches are proposed to allow organizations to be more efficient, more effective and more readily adaptable to changes than traditional management. Most emerging approach is business process management, business process reengineering, etc. In fact organizations often have to change their processes at higher or lower frequencies, in order to improve and make them more efficient. Business Process Reengineering has become essential in these times because we cannot afford to have heavier and less agile process without losing any competitiveness. The risk increases even more when the processes are common and horizontal to several departments and this prevents to act quickly and effectively on the problems it causes.

## 2.1 Business Process Reengineering

BPR is described by Hammer and Champy as "questioning fundamental and radical redesign of organizational processes to achieve dramatic improvements in current performance on cost, service and speed” (Hammer & Champy, 1993), That means that existing processes are carefully reviewed and redesigned in order to improve and complete more effectively and efficiently the functions of the company. It must be supported by a modeling process because it was increasingly necessary to explicitly define the process. In the context of reengineering business processes, organizations compare business process models to identify operational commonalities and differences. Such comparisons are, for example, necessary when organizations merge and need to determine and resolve the differences between their operations, and when an organization needs to check whether its operations conform to a company-wide or industry-wide standard. This standard or these reference models are models that have proven their effectiveness and success. In such way companies will find improvements to remedy problems or gaps in its business process.

## 2.2. Business Process and Business Modeling

Over recent years the importance of business process is growing at a very rapid rate. There are several factors that can lead companies to change their business processes (Buller & Gerritz, 2001):

* They are threatened by competition.
* They need to develop new solutions to respond better to the customers’ needs and requirements.
* They must respond to organizational changes.
* They must respond to the emergence of new technologies

A business process can be defined as a related group of tasks that together create value for a (internal or external) customer. Examples of business processes are the processing of orders in a factory or the handling of insurance claims at an insurance company. Moreover, the business processes are defined by ISO as a "Set of interrelated or interacting activities which transform inputs into output". Many modeling notations are available to capture business processes, including Event-driven Process Chains (EPC), UML Activity Diagrams and the Business Process Modeling Notation (BPMN). Modeling BPs is very helpful for the success of BPR projects according to the several advantages it offers, such:

* understanding the existing BPs, identifying their weakness and problems,
* identifying areas of potential improvement and areas with a gap between existing BPs and the BPR objectives,
* representing new BPs in order to evaluate their performance
* increasing the speed and the quality of the implementation of BPR improvements
* being used for end-user training : all documents such as work instructions, user instructions, ISO documents, etc. are stored in the model that constitutes a single information source
* being used as “the best practices models”, BPs models can be used as start point in similar companies initiating BPR projects.

In this effect, modeling activity is primarily an activity of the brain. The meta-models, frameworks and methods are defined in this spirit. A process modeling tool is a software which allows representing and organizing BPs of a company. It has a graphical part to represent the diagram and a textual part to describe the graph and the associated data. The objective is to ensure that general tasks are described in one way and it is reused in multiple locations in order to facilitate updates.

# 3. Models and model transformations

## 3.1 MDE and Model Transformation

The MDE community has been using the concepts of model, meta-model, and meta-meta-model for quite some time. Models are considered as the unifying concept in IT engineering. Several sources acknowledge that a model is a simplified representation (or an abstract description) of a part of the world named the system (Mejri & Ghannouchi, 2013). Models have been used as initial design sketches mainly meant to communicate ideas among developers. A model is useful if it helps to gain a better understanding of the system, captures some characteristics of the system and provides knowledge about it. In an engineering context, a model is useful if it helps deciding the appropriate actions that need to be taken to reach and maintain the system’s goal (Mejri & Ghannouchi, 2013). In MDE interests are on models expressed in precise languages. When building modeling tools, one needs to model the structure and well-formed rules of the language in which the models are expressed. Such models are called meta-models (Jouault, Allilaire, Bézivin & Kurtev, 2008) : they are considered as definitions of the abstract syntax of modeling languages. The relation between a model expressed in a language and the meta-model of this language is called “conformsTo”. Meta- models are in turn expressed in a modeling language called meta-modeling language. Its conceptual foundation is captured in a model called meta-meta-model. Models, meta-models, and meta-meta-model form a three-level architecture with levels named M1, M2, and M3 respectively (Jouault, F & Bézivin,J, 2006).

Model transformations (MT) aims to provide a mean to produce target models from a number of source models in the scope of Model Driven Engineering (MDE). MDE is a software development methodology that is mainly concerned with the evolution of models as a means of developing software by focusing on models (Sawprakhon, P & Limpiyakorn, Y, 2014) MT follows a model transformation pattern shown in Fig. 1.



Fig1. Pattern of model transformations (Sawprakhon, P & Limpiyakorn, Y, 2014).

Mt is a transformation program which execution results in automatic creation of model Mb from Ma. These three entities (Mt, Mb, and Ma) are all models conforming to MMt, MMb, and MMa meta-models, respectively. MMt corresponds to the abstract syntax of the transformation language. The three meta-models conform to a meta-meta-model named MMM. In the context of OMG standards, the meta-meta-model MMM is the MOF. Meta-models are prerequisites for performing automated model transformations.

## 3.2 Graph transformation

In contrast to source code where a tree structure is appropriate (e.g., parse trees, abstract syntax trees), graphs are a natural representation of models as many models are intrinsically graph-based in nature (e.g., state charts, activity diagrams, collaboration diagrams, class diagrams, Petri nets). Model transformation problems could be formulated as graph transformation problems. Thus, a variety of tools choose this technique as the underlying mechanism for the transformation engine.

## 3.2.1. Graphs

Graphs are structures that consist of a set of vertices and a set of edges, that have as source and target the vertices of the graph. More formally, graphs are defined as follows:

A (*labeled*, *directed)* graph *G* = ( *NODES*, *EDGES*, *source, target, label*) consists of a finite set *NODES* of *nodes*, a finite set *EDGES* of *edges*, two mappings source and *target*, assigning a *source* and a *target* node to each edge, and a mapping *label*, assigning a labeling symbol from a given alphabet to each node and each edge.

A graph L is a subgraph of G, denoted by L ⊆G, if the node and edge sets of L are subsets of the respective sets of G, and the source, target and label mappings of L coincide with respective mappings of G. In figure2, the graph L is a subgraph of G. L has an occurrence in G, denoted L ⭢G, if there is a mapping *occ* which maps the nodes and the edges of L to the nodes and the edges of G, respectively, and preserves sources, targets and labeling, that is , for each edge e in L. The source of the image of e coincides with the image of the source of e, the target of the image of e coincides with the target of r, and for each item x in L, the label of the image of x coincides with the label of x. The occurrence of L in G given by *occ* is the subgraph og G consisting of the images of all nodes and edges. The graph L in figure 2 has occurrence in H in which the nodes 1 and 3 are identified ( denoted by 1=3) and node 2 in L corresponds to node 2 in H (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).



Fig 2. L is a subgraph of G and has an occurrence in H (Jouault,F & Bézivin,J, 2006).

A total homomorphism between graphs is a mapping of vertices and edges that is compatible with sources and targets of edges. Intuitively, a total homomorphism from graph G1 to graph G2 means that all items(vertices and edges) of G1 can be found in G2 (but distinct vertices/edges of G1 are not necessarily distinct in G2 ( Lumertz, Ribeiro & Duarte, 2106)

## 3.2.2. Graph variations and extensions

There are a variety number and extensions of the graph concept, such:

* Directed graphs: the edge set is given by a binary relation on the set of nodes. No parallel edges are allowed.
* Undirected graphs: they are a kind of direct graphs but undirected edges are replaced by two directed edges in opposite direction.
* Hypergraphs: are generalization of graphs in the sense that an edge has a sequence of source nodes and a sequence of targets nodes.
* Hierarchical graphs: where a subgraph can be abstracted to one and a bunch of edges between two abstracted subgraphs to one edge.
* Typed graphs: used for specification and programming. Labels are divided into classes, called types, and that edges of a certain type are restricted to be incident only to certain types of sources and target nodes.
* Attributed graphs: an attribute cam be a number, a text, an expression, a list or a graph too. Attributes could be of different types and attribute operations compatible with these types are available to manipulate the attributes (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).

## 3.2.3. Graph transformation process

Graph transformation is a graphical formal language for system modeling. The mathematical foundation of graph transformation systems returns to more than 40 years ago in response to shortcomings in the expressiveness of classical approaches to rewriting (e.g. Chomsky grammars) to deal with nonlinear grammars (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).One of the most fundamental features the Graph Transformation System (GTS) is its formal and accurate mathematical basic.

An attributed GTS is determined as a triple: AGT= (TG, HG, R) In which TG is the type graph, HG is the host graph, and R is the set of rules. Type graph is a tuple: TG= (TGN, TGE, src, trg), with two functions src: TG E → TGN and trg: TG E → TGN. TGN and TG E are a set of node types and a set of edge types, respectively. A host graph HG over TG, is a graph equipped with a graph morphism type G : HG → TG that assigns a type to every node and edge in HG (Youseﬁan, Rafe & Rahmani, 2014).

# 4. Graph transformation approaches

A graph transformation approach comprises a class of graphs, a class rules, a rule application operator, a class of graph class expressions and a class of control conditions. The objective of these later is to restrict the non-determinism of graph transformation.

## 4.1. Node label replacement approach

The node label replacement approach, mainly developed by Rozenberg, Engelfriet and Janssens, allows replacing a single node as the left hand side L by an arbitrary graph R, as shown in figure 3 (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).The connection of R with the context is determined by embedding rules depending on node labels.



Fig 3 : Node label replacement approach (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).

## 4.2. Hyperedge replacement approach

The hyperedge replacement approach, mainly developed by Habel, Kreowski and Drewes, has as the left hand side L a labeled hyperedge, which is replaced by an arbitrary hypergraph R with designated attachment nodes corresponding to the nodes of L. The gluing of R with the context at the corresponding attachment nodes leads to the target graph (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999). (cf. figure 4).

****

Fig 4 : Hyperedge replacement approach (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).

## 4.3. Algebraic approach

The algebraic approach is based on pushout constructions, where pushouts are used to model the gluing of graphs. In fact, there are two main variants of the algebraic approach, the double and the single pushout approach. The double pushout approach, mainly developed by Ehrig, Schneider and the Berlin- and Pisa-groups. This approach is also known as the double-pushout approach, short DPO approach, in contrast to the singlepushout (SPO) approach.

In the DPO approach a production is given by *p = (L, K, R),* where *L* and *R* are the left and right hand side graphs and *K* is the common interface of *L* and *R*. Given a production *p = (L, K, R)* and a context graph *D*, which includes also the interface *K*, the source graph *G* of a graph transformation *G ⇒ H* via p is given by the gluing of *L* and *D* via *K*, written *G = L +K D*, and the target graph *H* by the gluing of *R* and *D* via *K*, written *H = R +K D*. More precisely graph morphisms *K → L, K → R* and *K → D* are used to express how *K* is included in *L*, *R* and *D* respectively. This allows to define the gluing constructions *G = L +K D* and *H = R +K D* as pushout constructions (1) and (2) leading to a double pushout as shown in figure 5.



Fig 5. DPO graph transformation (Andries, Engels, Habel, Hoffman, Kreowski & Kuske, 1999).

## 4.4. Logical approach

The logical approach allows expressing graph transformation and graph properties in monadic second order logic.

# 5. Related works

In this section we discuss the background to our research. Firstly, we introduce taxonomies and classifications of graph transformation. Secondly we present the different models and meta-models for graph transformation as found in literature. A summary of the comparison is given in figure 6.

## 5.1. Graph transformation comparison with QVT

Each graph transformation approach comes along with an accompanying tool. A comparative study of the corresponding tools along main model transformation facilities with the Query/View/Transformation language is presented (Youseﬁan, Rafe & Rahmani, 2014) and this since is supposed to become the OMG standard language for model transformation (Query/View/Transformation, 2005). This comparison is based upon seven main criteria: typing Information, instances, pre-conditions, post-conditions, actions, control, correctness, and some additional features.

## 5.2. Comparison of Model Transformation Solutions

The criteria used in this comparison are: Persistent classes (attributes of primitive data type and attributes of persistent classes), Inheritance (transitive closure and attributes and associations of subclasses), Non-persistent classes (attributes and associations of non-persistent classes), deletion of source model and deletion of helper structure abilities (Query/View/Transformation, 2005).

## 5.3.Comparison of Accompanying Tools

A comparative study upon supporting tools functionalities: editors, simulation, compilation, debugging and validation is presented in (Query/View/Transformation, 2005).



Fig 6. Comparison of graph transformation approaches and QVT, solution and tools (Query/View/Transformation, 2005).

## 5.4. Mens model (Mens & Pieter, 2006)

Tom Mens et al. offer in (Mens & Pieter, 2006). another taxonomy of model transformation based on the discussions of a working group on model transformation of the Dagstuhl Seminar on Language Engineering for Model Driven Software Development. The two works are relatively similar since we find the same concepts. Mens et al. focuse on helping the developer choosing a particular transformation language by answering crucial questions for model transformation they proposes ( Czarnecki & Helsen, 2003).

## 5.5. Cezarnaki prototype ( Czarnecki, K & Helsen, S, 2003)

Based on the state of the art described , Czarnecki et al, proposes a possible taxonomy for the classification of several existing and proposed model transformation approaches ( Czarnecki & Helsen, 2003).This taxonomy is the result of applying domain analysis to existing model transformation approaches, and is shown in the top-level feature diagram, where each subnode represents a major point of variation in figure 7.



Fig 7. Feature diagram representing the top-level areas of variation ( Czarnecki & Helsen, 2003).

This taxonomy is not normative where used terms are:

* Transformation rule: it consists of two parts: a left-hand side (LHS) and a right-hand side (RHS). The LHS accesses the source model, whereas the RHS expands in the target model. Both LHS and RHS can be represented using any mixture of variables, Patterns and Logic expressions ( Czarnecki & Helsen, 2003), (Taentzer, Ehrig, Guerra, Lengyel, Levendovszk, Prange & Varro-Gyapay, 2005).
* Rule application scoping: which allows a transformation to restrict the parts of a model that participate in the transformation. Some approaches support flexible source model scoping (e.g. XDE and GreAT), where a scope smaller than the entire source model can be set. The latter can be important for performance reasons. The target scope is the scope of the target model, in which the RHS will be expanded (e.g., XDE) (Taentzer, Ehrig, Guerra, Lengyel, Levendovszk, Prange & Varro-Gyapay, 2005).
* Relationship between Source and Target: Some approaches mandate the creation of a new target model that has to be separate from the source . In some other approaches, source and target are always the same model, i.e., they only support in-place update (e.g., VIATRA, GreAT) ( Czarnecki & Helsen, 2003), (Taentzer, Ehrig, Guerra, Lengyel, Levendovszk, Prange & Varro-Gyapay, 2005).
* Rule application strategy: there may be more than one match for a rule within a given source scope, so an application strategy is needed. The strategy could be deterministic, non-deterministic or even interactive. For example, a deterministic strategy could exploit some standard traversal strategy (such as depth-first) over the containment hierarchy in the source ( Czarnecki & Helsen, 2003).
* Rule Scheduling: it determines the order in which individual rules are applied. The scheduling mechanism can vary in four main areas: form, rule selection, rule iteration, phasing.
* Rule Organization: it is concerned with composing and structuring multiple transformation rules. Three areas of variation are possible: Modularity mechanisms, Reuse mechanisms and Organizational structure.
* Traceability Links: links are between source and target transformations elements. These links are useful in performing impact analysis, synchronization between models, model-based debugging, and determining the target of a transformation ( Czarnecki & Helsen, 2003), (Taentzer, Ehrig, Guerra, Lengyel, Levendovszk, Prange & Varro-Gyapay, 2005).
* Directionality: transformations may be unidirectional or bidirectional. Unidirectional transformations can be executed in one direction only, in which case a target model is computed (or updated) based on a source model. Bidirectional transformations can be executed in both directions, which is useful in the context of synchronization between models ( Czarnecki & Helsen, 2003), (Taentzer, Ehrig, Guerra, Lengyel, Levendovszk, Prange & Varro-Gyapay, 2005).

# 6. Discussion

Graphs are particularly suited for modeling and representing many structured and dynamic contexts, for instance software architectures, distributed communications, call graphs and many more. Graph transformation systems are not only an intuitive way to represent the syntax of graphs, but also to formalize how graphs evolve. As stated by Ehrig et al.: ‘‘Graph transformation allows one to model the dynamics in all these descriptions, since it can describe the evolution of graphical structures” ( Maddeh,M, Romdhani, M & Ghedira,K, 2009). According to the examination of the literature, we notice that:

* model transformation is a very complicated but interesting field with several knowledge, notations and concepts,
* numerous taxonomies trying to classify model transformation operations according to various points of view, purposes ( abstraction level, meta-models, simple or multiple, preservation, etc.) are proposed in the literature,
* lots of model transformation tools are emerging, but most generate text or C, C++ or Java code,
* graph model transformation classifications or taxonomies are rare especially for BPR purposes.

As enterprises discovered the benefits of Business Process Modeling (BPM), the use of Business Process (BP) models moved from a ”luxury article” to an ”everyday necessity” in the last years. Meanwhile many companies own thousands of models which describe their business. Since business changes over the years, e.g., business to business interoperability came up with new inventions in communication and companies merge with others, there arises a need to keep existing business models up-to-date and to synchronize or translate them into a contemporary BPM language (Löwe, König, Schulz & Schultchen, 2015). To facilitate these scenarios a model transformation technique for BP models is needed. More precisely, the aim of this paper is to propose a metal-model which describes the most important concepts to be considered while transforming “as-is” BP models to “to-be” ones during a particular BPR project.

# 7. Proposed meta-model

The proposed solution presented in this work is based upon a meta-model integrating graph transformation knowledge in BPR context. Figure 8 describes the main concepts involved in this meta-model. This later is composed of two compartments: first compartment is related to BPR context. The second one reflects graph transformation features.

## 7.1. BPR level

The BPR level capitalizes most essential knowledge regarding a BPR project launched by a specific company. These concepts are:

* BPR operation: it reflects a BPR project launched for a fixed duration and by a given company,
* Action: it composes a BPR operation. Samples are defining, designing new BPs, redesigning existing BPs, etc. Each action is characterized by a noun, a duration and a location.
* BP: a company business process involved in the BPR project. It may be a new proposed process or an existing one which is candidate for BPR improvements.
* MT (Modeling Technique): it is used to model all BP views: activities, roles, input, output, data, objects, etc. ULM class diagrams, UML activity diagrams, Event Process Chain, Data flow Diagram, etc. are samples of MT.
* Concept: is a MT notation, such: activity, actor, class, association, multiplicity, use case, state, object, transition, etc.
* "Incites" association: links "Action" to “GMT operation”: A BPR action may cause an graph transformation operation for an existing BP model subject of a reengineering operation.
* "Present" association between "Existing BP" and "Source -Model" concepts: an existing BP in is modeled by a model (source), which can be a textual description or a graph or chart-based model. .
* "Is presented" association between "BP" and "Model" concepts: a BP my modeled by a model.

##

Fig 8. A meta-model proposal for graph transformation in BPR projects.

## 7.2. Graph transformation level

The second level of the proposed model involves knowledge regarding graph model transformation context. These data are depicted after literature examination. Related concepts are described below:

* “GMT operation”: it is a plot by transformation task for translating a source model into a target model. It is characterized by a name, a duration and a cost,. Examples of GMT operations are reverse engineering, migration, refactoring, etc.
* “Source- Model”, “intermediate-Model” and “target-Model” are respectively used models (such as ER model) operated (eg PDM) and product (such as script database) . It should be noted that an intermediate model is applied when it is difficult to obtain directly a target model, for example transforming a conceptual data model to a database script (code) requires conversion to an intermediate model that is the physical data model . The target respectively source model are consistent with their source meta-model respectively their target meta-model.
* "Model": is a representation of the real world (ER model, a class diagram, activity diagram, DFD, Petri network, etc.). Every model conforms to a meta-model.
* "Meta model" is a model of models, such as the UML meta-models.
* Rule: a data processing approach uses a set of rules to be applied to get,, from the components of a source model, a target items (examples of rules: "any activity is a transition of the Petri net," or "every class is transformed in a database table ") . Algebraic approach SPO (Single Push Out) and DPO (Double Push Out ) are also examples of GMT rules.
* GMT Approach: a graph-based processing approach reflects the mechanism adopted to actually perform the transformation of graph-based models. These approaches are based on the theory of graphs and transformations fall into logical transformation of sets and algebraic. A GT approach is characterized by its name, its complexity, its level of automation, and the type of transformation.
	+ - The transformation approach name such as pushout, pulback, etc.
		- The complexity degree : a GMT is called simple if the rules applied are of the form 'if ... then' (such : the transformation of a class diagram into a relational model using the rule 'if a class is persistent then apply the method "class to table ''). It is complicated when the GMT uses a mixture of grammar and rules, such the processing approach supported by the ATOM3 tool (Toffetti & Pezz, 2103) is called complex because it combines rules and tripled grammars.
		- The automation level (N. automation): an approach TPG can be automatic, semi-automatic or manual. A GMT approach is automatic when it is based on standard tools or programs in automating the generation of transformation rules. It is semi-automatic if it is described by a transformation process that is based primarily on matching and mapping These operations require subsequent human expert intervention to search for semantically equivalent elements between two meta-models (source and target).These mappings are stored in a mapping pattern conforms to a mapping meta-model (which defines the different types of links generated by the matching model). Eventually the human expert intervenes to adapt, complete or validate the results obtained. Finally a model of transformation is issued automatically from a model mapping (Hammoudi, 2008),
		- An approach is called manual if the mapping operations and processing are merged and performed in a single step by a human expert. For example the transformation of specifications to an analysis model is a so-called manual transformation.
		- Type: a GMT may be simple ("1 to 1") if a single source model is transformed to only one target model. For example the conversion of class diagram to a relational database. It could be multiple if the transformation is of type "M to N" when more source models are transformed into several target pattern. A GMT is a decomposition ("1 to M") when a single source is transformed to multiple targets models. A GMT is a fusion if several models sources are merged into a single target model.
* Set- approach : a GMT is a set-approach when it is based upon sets during the transformation operation. There are two types of set-approaches, namely:
	+ - replacing vertices between the source and target models.
		- converting, merging or eliminating of arcs during transformation.
* Logical approach: when a GMT is based on logic to formally write the properties of graphs (vertices, ridges and basic relationships (connectivity, etc.).) These different properties are rewritten using the first-order logic and second order logic.
* Algebraic approach: it is an approach based on the theory of class and topological operations. In category theory, it is a mathematical conceptual basis which is written in universal form "whatever ... there ... as". As against the conceptual topologies are used to locally modify a definite structure. As an example of algebraic approach we cite the "pushout" and "pulback".
* GMT Tool : each approach is supported by a tool that allows for the operation itself. Tools samples are: AGG VIATRA, FUJABA, ATL, etc. which differ in some characteristics such as format models, validation operations, etc.
* Format : GMT tools use different formats such as EMF (Eclipse Modeling Framework), UML, XSD (XML Schema Definition ) and GXL (Graph eXchange Language).
* Formalism : it deals with technical constraints used during a transformation operation. For example grammars triplets graphs used by ATOM3 is a technique grammar.
* Strategy: it is related to rules mechanism, as priority level strategy used in AGG tool.
* Validation: a GMT tool may allow validation of the transformed models. Some validation techniques may be used, such as:
	+ - Formal Validation: logical and mathematical concepts are used for this kind of validation. For example AGG and TIGER tools perform formal validation tasks (confluence, termination, analysis of critical rules pairs).
		- Unit- Testing : unit testing is to test each element of the target model developed separately. For example test each class separately in the Fujaba tool.
		- Model-checking: it deals with verifying that are no syntax errors in the accuracy of UML rule. This verification simply involves the decision of whether the left side conforms to the UML meta-model in the right. VIATRA tool offers the model checking “Check-VML”.
		- Technological space : each tool belongs to a space that reflects the technological framework respected during processing. For example XML space for AGG tool and MDA for VIATRA tool.

# 8. Conclusion and future work

In this paper we proposed a meta-model including graph model transformation knowledge in business process reengineering context. This meta-model assists designers, analyzers and mangers participating in BPR projects in redesigning new BPs models by transforming existing models.

In this work, the meta-model is a UML class diagram composed of two levels: BPR level and graph model transformation level. The first one capitalizes essential information regarding BPR operation, actions, BP and models. The second level contains information concerning graph transformation features: used formats, GTM approaches, GMT types, GMT tools, etc. The meta-model can be used to generate information for different GMT operations during BPR projects. We did not provide experimentation case study this is an ongoing work. a tool for instances generation,. We also plan to work on evolution, by providing an instance generation tool based upon the proposed meta-model. Furthermore, including semantic and completeness mechanisms for the proposed meta-model is subject of future work too.

## References

Andries, M, Engels,G, Habel,A, Hoffman, B, Kreowski, H & Kuske, S, (1999), Graph transformation for specification and programming, Science of Computer Programming, Vol. 34, pp. 1–54.

Buller.N & Gerritz.J, (20011), An Investigation into Techniques for Business Process Modeling and their Application to an Audit of Current Business Processes at the University of Natal. Part Two of a Report Prepared for the Information Technology Steering Committee of the University of Natal.

Czarnecki, K & Helsen, S, (2003), Classification of Model Transformation Approaches. In OOPSLA’03 Workshop on Generative Techniques in the Context of Model-Driven Architecture.

Chen. Y.C, (2001), Empirical Modeling for Participative Business Process Reengineering. Thesis. University of Warwick. Conventry, United Kingdom.

Hammer.M & Champy.J, (1993), Reengineering in the Corporation- ISBN 0-88730-640-3, Harper Business.

Hammoudi, (2008), Vers un processus semi-automatique de la transformation dans l’approche MDA, . s.l. : université de ESEO, de LERIA et de Université Fédérale du Maranhão, [www.idm08.uha.fr/actes/p14.pdf](http://www.idm08.uha.fr/actes/p14.pdf).

Jouault, F, & Bézivin,J, (2006), KM3: A DSL for metamodel specification FMOODS, Italy.

Jouault, F, Allilaire,F, & Bézivin, J I. Kurtev, (2008), ATL: A model transformation tool, Science of Computer Programming, 72.

Löwe, M, König,H, Schulz,C & Schultchen, M, (2015), “Algebraic graph transformations with inheritance and abstraction”, Science of Computer Programming 107–108 /2–18.

Lumertz,P.R, Ribeiro,L & Duarte, L.M, (2106), User interfaces metamodel based on graphs, Journal of Visual Languages and Computing 32/1–34.

Maddeh M., Romdhani,M & Ghedira,K, (2009), Classification of model refactoring approaches, JOURNAL OF OBJECT TECHNOLOGY, Vol.8 , No.6.

Mejri, M & Ghannouchi, S.A, (2013), Discovering reference process models in the context of BPM projects, P o c e d a T e c h n o l o g y , 9 , 4 8 9 – 4 9 7.

Mens, T & Pieter.V.G, (2006), A Taxonomy of Model Transformation and its Application to Graph Transformation,, 2006. s.l. : Universit´e de Mons-Hainaut,. p. p125\_142

Muthu.S, Whitman.L & Cheraghi, S.H,(1999), Business Process Reengineering: A Consolidated Methodology. The 4th Annual International Conference on Industrial Theory, Applications and Practice, San Antonio, Texas, USA. November 17-20.

Query/View/Transformation,(2005),QVT-Merge Group, version 2.0 (2005-03-02). <http://www.omg.org/cgi-bin/apps/doc?ad/05-03-02.pdf>

Sawprakhon,P & Limpiyakorn.Y, (2014), Sequence Diagram Generation with Model Transformation Technology, International MultiConference of Engineers and Computer Scientists 2014 Vol I, IMECS, Hong Kong.

Taentzer, G, Ehrig, K, Guerra, E , Lara,J, Lengyel, L , Levendovszky, T , Prange, U, Varro, D & Varro-Gyapay, S, (2005), Model Transformation by Graph Transformation: A Comparative Study, In Proceedings of the MTP workshop at MoDELS.

Toffetti,G & Pezz, M, (2103), Graph transformations and software engineering: Success stories and lost chances, Journal of Visual Languages and Computing 24/ 207–217.

Wesk. M, (2007), Business Process Management: Concepts, Languages, Architectures, 1st edn. Springer, Heidelberg.

Youseﬁan,R, Rafe,V & Rahmani,M, (2104), A heuristic solution for model checking graph transformation systems, Applied Soft Computing 24/ 169–180.

**Leila Jamel Menzli** is an assistant professor at Princess Nourah University,  Kingdom of Saudi Arabia (KSA). She received her Engineer and MS degrees in computer science from the University of Tunis. She obtained her PhD degree in computer science-software engineering from the National School of Computer Science Tunisia and the University of Nancy I- France. Her current research interests include business process reengineering and management, business process modeling, model transformation and model driven engineering. She is a member of the RIADI- Laboratory in National School of Computer Sciences- Tunisia.

**Raja Hanafi** obtained her MS degrees in computer science from the University of Carthage Tunisia. She is actually a PhD student in the National School of Computer Sciences- Tunisia.