# EVOLUTION MANAGEMENT FRAMEWORK FOR MULTI-DIMENSIONAL INFORMATION SYSTEMS

Nesrine Yahiaoui, Bruno Traverson
*EDF R&D, 1 avenue du Général de Gaulle, F-92140 Clamart, France*

Nicole Levy
*UVSQ PRiSM,45 avenue des Etats-Unis, F-78035 Versailles, France*

Keywords: Dynamic adaptation, Information Systems, RM-ODP, MDA.

Abstract: Information Systems (IS) have become key elements in enterprise activities and are now fully embedded into business units. Productivity gained by this proximity is to be balanced with more strategic requirements against the Information System. In particular, due to the proximity to business layers, adaptability property of IS is more than ever required. The framework we have developed aims to keep synchronized multiple descriptions of the same system in case of evolution. Its foundations are based on RM-ODP viewpoints and meta-modelling technology. A prototype tool to support the framework has been developed as an EMF/Eclipse plug-in.

## 1 INTRODUCTION

Information Systems (IS) have become key elements in enterprise activities and are now fully embedded into business units. Productivity gained by this proximity is to be balanced with more strategic requirements against the Information System. In particular, due to the proximity to business layers, adaptability property of IS is more than ever required.

Now-a-days, many software architects tend to agree that the design of sophisticated and distributed applications has to be performed according to different viewpoints (IEEE, 2000). This leads to multi-dimensional systems where each dimension describes a particular concern.

Evolution of multi-dimensional systems may appear tricky if no links are maintained between the various dimensions of the system. The purpose of our framework is to manage these links and to use them during evolution.

Part 2 gives the motivations of our work. Then, part 3 and part 4 respectively focuses on specification and implementation of the framework. Part 5 looks at some related work and part 6 concludes the paper.

## 2 CONTEXT AND OBJECTIVES

As mentioned in the introduction, adaptation to evolutions is critical for software-intensive systems. This is the major motivation for our framework. Also, our work has been based on two major standards: RM-ODP (Reference Model of Open Distributed Processing) standard (ISO/IEC, 1995) and MDA (Model Driven Architecture) approach (OMG, 2001).

This motivation section is broken up into three parts: overview of RM-ODP standard, overview of MDA approach and organization of our work.

### 2.1 RM-ODP Viewpoints

First of all, our framework is based on RM-ODP. This standard recommends the separation of stakeholders concerns using five viewpoints: Enterprise, Information, Computation, Engineering and Technology.

Identifying those viewpoints allows the system specification to express at the same time but distinctly the business the IS supports (Enterprise Viewpoint), the way it is modelled in the computer system regarding information and functions

Yahiaoui N., Traverson B. and Levy N. (2006).
EVOLUTION MANAGEMENT FRAMEWORK FOR MULTI-DIMENSIONAL INFORMATION SYSTEMS.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - ISAS*, pages 331-336
DOI: 10.5220/0002489803310336
Copyright © SciTePress

(Information Viewpoint, Computational Viewpoint, Engineering Viewpoint) and the technical choices of the computer system mapping user requirements (Engineering Viewpoint, Technology Viewpoint).

The key points of RM-ODP are the completeness of its concepts and structuring rules and the relevance of its abstraction levels.

## 2.2 MDA

MDA is an approach recently promoted by OMG (Object Management Group) which emphasizes the use of models and meta-models. This approach defines, on one hand, PIMs (Platform Independent Models) to specify business aspects independently from the development platform and, on the other hand, PSMs (Platform Specific Models) which describe the implementation of the IS on a specific platform.

If the PSM is based on a programming language, the transformation from PIM to PSM is called code generation.

The transformation from a PIM to a PSM may be based on a third kind of model called PDM (Platform Description Model) which contains the description of the platform the implementation should be based on.

The transformation is, in this case, a model transformation taking a PIM and a PDM as inputs and producing a PSM as output.

The key points of the MDA approach are code generation and model transformation, drastically reducing the development costs.

## 2.3 Adaptation to Evolutions

RM-ODP standard provides the concepts to specify the system. More recently, RM4ODP specification (ISO/IEC, 2005) is proposing UML profiles to express them. Besides, the separation into viewpoints allows the designers to manage the complexity of the development process, but there is a need to maintain correctness and consistency of the models during evolutions.

First of all, local consistency should be checked when a model describing one viewpoint of the system has changed: Are the constraints enforced by this viewpoint still verified? Have we the ability to go back and forth in the versions generated by the various evolutions?

Then, impact on the global system should be managed. Have the links between models been updated? Has a change in a model to be reflected into another model?
All these questions can only be answered by knowing the exact history of every model and the potentially complex relationships between models.

Our framework intends to help designers and architects by offering support tools. As evolution may be seen as model transformation, MDA technology constitutes its core.

## 3 SPECIFICATION

The specification of our framework will be presented in two parts. The first part will focus on local consistency problem. The key point here is to maintain the trace of all changes made on one model. The second part will describe the linking mechanisms that have been defined to handle global consistency. Links are especially used for impact management.

## 3.1 Modelling Evolution

Evolutions done on a model describing one viewpoint of a system may be gathered in an evolution scenario. This latter concept is defined as a sequence of actions and gives a textual representation of changes performed between two versions of the model.

Of course, evolutions may be performed using a graphical tool. But, this is always possible to translate significant graphical events into a textual representation. Moreover, the scenario permits to keep track of successive evolutions and also to analyse impacts of these evolutions.

Evolution scenarios are described using an action language and stored in a repository. This facility permits, for the designer, to explicitly manage them. Also, it is associated to impact management process that generates impact analyses that are written using the same action language. An action may be either a creation (**Create**), a modification (**Modify**), a suppression (**Delete**) or a replacement (**Replace**).

## 3.2 Linking Viewpoints

This section will not provide meta-models for viewpoints because they are given in UML4ODP specification (ISO/IEC 2005). We will concentrate on description of correspondence rules that are given in part 3 of RM-ODP standard (ISO/IEC 1995) and correspondence links that materialize the relationships between models based on different viewpoints. Then, we will present the complete meta-model.

### 3.2.1 Correspondence Rules

When an ODP system is fully described, models representing each viewpoint should be available. Also, designers should verify that they are consistent. As we have already said earlier, two consistency levels have to be considered.

Local consistency ensures that models are described according to their respective meta-models or viewpoints. Global consistency takes care of dependencies between views. This later level is based on correspondence rules. These rules describe constraints on concepts from two distinct viewpoints and concerned by a correspondence.

RM-ODP standard does not insist on mechanisms to handle consistency between models based on two different viewpoints. It only gives informally a set of rules linking viewpoint concepts. These are some rule examples also given in (Putman, 2001):

- Example of enterprise and computational viewpoint correspondence: an enterprise object and the role it assumes correspond to a computational object or a configuration of computational objects.
- Example of enterprise and engineering viewpoint correspondence: the enterprise policy corresponds to and determines the engineering transparency mechanisms and supporting engineering objects.

These rules describe correspondence between concepts from two distinct viewpoints together with their multiplicity. They should be formalized into an executable language in order to enable verification. This is under current work in our research team. But we can give a flavour of it, for instance, on the example of enterprise and computational viewpoint correspondence given just before:

```
Correspondence rule:
Enterprise:EnterpriseObject[1] ←→
Computational:ComputationalObject[1] OR
(Computational:ComputationalObjet[1..*]
AND
(Computational:PrimitiveBinding[1..*]
OR Computational:BindingObject[1..*]))
```

### 3.2.2 Correspondence Links

As shown in the previous section, correspondence rules express consistency constraints between two viewpoints. However, these rules are for general-purpose and do not designate specific instances. In other words, they do not give to the designer the ability to navigate through models to learn real relationships between model elements.

We propose correspondence links that introduce traceability information between model elements by linking elements belonging to distinct models. Traceability is an important property for impact management. Thus, the correspondence link permits to know what model elements are to be checked when there is an evolution.

### 3.2.3 Link Meta-Model

A correspondence link is established between model elements belonging to models related to two distinct viewpoints. It materializes a certain relationship between these model elements. The correspondence link should enforce correspondence rules expressed between viewpoints concerned.

This link is bi-directional. It is possible to navigate through it from any endpoint. Bi-directionality enables to relax any constraint on navigation through models of different viewpoints. Any model may be modified. Then, it is possible to retrieve correspondence of a model element in any other models.

Moreover, links established between two models may be of multiplicity of 1-*. Then, this should be possible to link a model element related to one viewpoint to one or more model elements related to another viewpoint. Multiplicity restriction may apply in a correspondence rule. We call this first kind of correspondence rule a structural rule (StucturalRule).

To manage impact of a model evolution on another model, we introduce also the active rules (ActiveRule). An active rule is directly associated to a link. It permits to drive evolution of model elements that are impacted due to correspondence with another model. An active rule is composed of three parameters: event, condition and action - also called ECA (Event/Condition/Action) rule.
The global semantic of an active rule is « When an event occurs, If the condition is satisfied Then action is performed.
Event may be triggered by a modification made on a model element. Condition is a predicate on the state of the model element or on linked elements. Action describes modifications to be applied on linked elements. These modifications are expressed using the action language already mentioned in section 3.1.

Thus, the Link Meta-Model is illustrated by figure 1 where Endpoint designates each extremity of the correspondence link and LinkModel aggregates all the correspondence links.

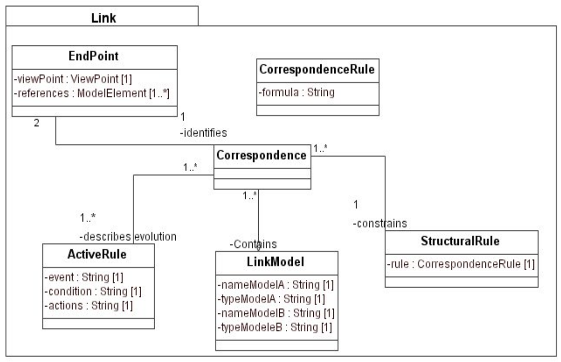

Figure 1: Link Meta-Model.

## 4 IMPLEMENTATION

Our architectural framework guides the designer in his or her various activities that may be gathered into three categories: description of models related to viewpoints, establishment of correspondence links and evolution of models. It is based on the link meta-model and the meta-models representing ODP viewpoints. Presently, we are working on the enterprise and the computational viewpoints.
First of all, we describe the three categories of activity, then the architecture and, finally, the Eclipse plug-in.

### 4.1 Activities Supported

- Description of models related to ODP viewpoints.

The designer is not constrained to follow a particular approach to construct his or her multi-dimensional system. The system may even be described by several designers; each one can build a model in accordance with their respective meta-model. The models and their meta-models are persistent; they are saved in repositories.

- Establishment of correspondence links.

Once all the models are completed, they are still independent. To establish correspondences, the designers have to link the models by using the link meta-model. The correspondence links are thus established between the models related to two different viewpoints. These links memorise and keep traces of the correspondence which can exist between the models and may be used to guide the evolution. The result of this activity is a link model that is saved with its meta-model in the repositories.

- Evolution of models.

The designer can modify any model of his or her multi-dimensional system. This evolution is either described in an evolution scenario or directly performed in a graphical way in the modelling tool. Our modelling framework includes impact management.

### 4.2 Architecture

This management is possible because our modelling framework is based on the following architecture, illustrated in figure 2.

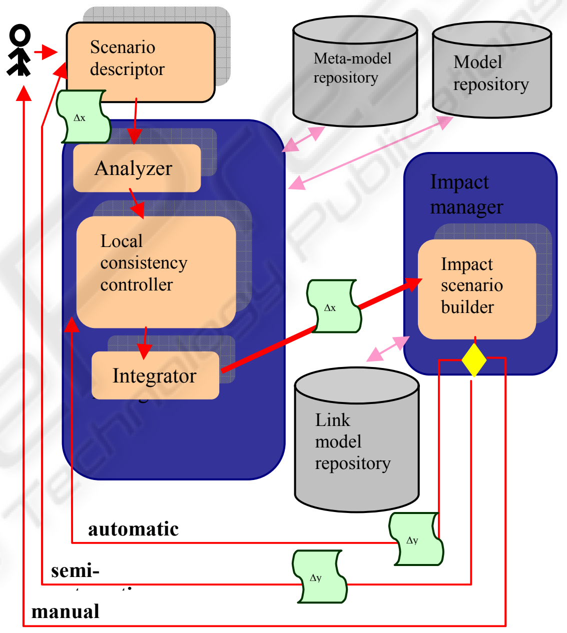

Figure 2: Framework architecture.

The architecture consists of several modules which collaborate. The designer writes the evolution scenario thanks to the **scenario descriptor** module. This module communicates with the **Analyser** which verifies the syntax and semantic of the scenario. The Local consistency controller verifies that the model evolution respects the constraint established by the respective meta-model. Once local consistency is enforced, the scenario is applied to the model by the **Integrator**. The Compiler, the Local consistency controller and the Integrator modules are included in the Evolution manager product.

If model evolution has impact on other models, the impact manager is triggered. This later builds the impact scenario (similar to an evolution scenario). It is mainly influenced by the applied evolution scenario and by the link model, especially on the active rules which drive the construction of the impact scenario. According to the adopted strategy, the impact scenario builder may interact with three different modules. These strategies are:

- Manual. The Impact scenario builder does not build the relevant impact scenario. It gives to the designer the list of model elements that must be modified in the models according to the other viewpoints. The designer will act on the other models according to his or her knowledge of the global system.
- Semi-automatic. The Impact scenario builder communicates with the scenario descriptor to edit the built scenario. This scenario is built from the active rules. The designer can validate it and/or improve it by adding or by modifying some actions.
- Automatic. The impact scenario builder communicates with the Local consistency controller to validate the constructed scenario. Then, the impact scenario will be applied by the Integrator.

### 4.3 Eclipse Plug-in

Our modelling and impact management tool is implemented as a plug-in in Eclipse EMF (Budinsky et al, 2004, Eclipse, 2006)

Eclipse is an open and extensible framework based on plug-in technology. It is dedicated for building integrated development environments (IDEs) that can be used to create applications as various as web sites, embedded Java programs, C++ programs.

EMF (Eclipse Modelling Framework) is a modelling framework for Eclipse, it offers a reflective API to manipulate the models and the meta-models which are built using the Ecore meta-meta-model. EMF can generate for each meta-model defined by Ecore a tree editor, that allows the instantiation of elements defined in a given meta-model.

To implement this tool, we supply two meta-models corresponding to enterprise and computational viewpoints.

Our plug-in uses other plug-ins, it uses EMF plug-in but also the generated plug-ins from the enterprise, computational, link and evolution scenario meta-models. Our plug-in thus enables to create new models: enterprise, computational and link. But also offers a graphical menu called **Evolution** which contains the following options:

- Textual editor. It opens a textual editor to write the evolution scenario. This editor permits to save and load a scenario.
- Graphic Editor. It opens a tree editor on a specific model.
- Validate Local Consistency. It verifies that the active model respects its meta-model.
- Validate Global Consistency. It verifies that the active model was adapted; otherwise it activates an order to the designer.

## 5 RELATED WORK

RM-ODP standard defines five viewpoints without giving a precise notation to describe the corresponding models. Furthermore, consistency among models is not handled in detail. It only describes some correspondence rules between viewpoint concepts.

Several works have been done around the formalization of viewpoints and the construction of consistent ODP systems.

The ODAC project (Open Distributed Applications Construction) (Gervais, 2003) carried out by the LIP6 laboratory and the DASIBAO project (Method based on ODP for the Architecture of Information Systems) (Picault et al, 2004) carried out by EDF R&D define, each one of both projects, an approach for building consistent ODP systems. The system is built in following steps and by applying transformation rules to the models. However, this consistency is lost if one of the models is modified. On the other hand, they impose a "top-down" approach which is not adapted when we consider that the systems can evolve according to any viewpoint.

Romeo's work (Romero et al, 2005) performed to the university of Malaga in Spain is mainly around the computational viewpoint. It describes a computational meta-model and proposes a UML 2.0 profile which bridges the gap between ODP and UML2.0 concepts. There are also works performed in Japan (Hashimoto et al, 2005) which propose UML 2.0 profiles for the of engineering and technology viewpoints. These works are named UML for ODP because it allows the designer to model according to ODP semantic ODP by using UML tools. However, they are not interested in the consistency of the systems nor in their evolutions.

Dijkman’s work (Dijkman et al, 2004) performed at the university of Twente in Netherlands is interested to define and to verify consistency relationships between the enterprise and computational viewpoints. He uses a generic framework to connect viewpoints and specifies reusable consistency rules. This framework also uses a basic viewpoint in which the two other viewpoints can be transformed. That is, the enterprise view (respectively. Computational) that respects the enterprise viewpoint (respectively. Computational) is transformed into a basic 'enterprise' view (respectively. Computational) that respects the basic viewpoint. The enterprise and computational views are consistent, if there is a abstraction relationship between the computational basic view and the enterprise basic view. However, this approach does not exploit the correspondence rules quoted in the standard which allows to put in relationships not only the models but also the model elements. Furthermore, this framework does not save the links and the relationships which can exist between the various model elements.

## 6 CONCLUSION

Models have long time been considered as a "documentation and discussion" tool. Recently, MDA initiative from the OMG consortium has pushed models as a more active part in Information Systems.

The framework, described in this paper, constitutes a proposal to support MDA approach in the specific case of multi-dimensional Information Systems based on RM-ODP standard.

The main contributions in this work are the specification of a Link Meta-Model and its implementation as an Eclipse plug-in. The Link Meta-Model goes beyond simple traceability because it contains active rules that permit impact management.

Short-term perspectives are to fully define the language used to express structural rules, to cover all the five viewpoints and to complete the implementation in order to release it as an open source.

Generalization of the framework to other kind of multi-dimensional systems sounds possible and could led us to compare our approach to other general-purpose model transformation tools. Another perspective is to use model weaving, transposition of AOP (Aspect-Oriented Programming) techniques on models to handle evolution and impact management.

## REFERENCES

Budinsky, F., Steinberg, D., Merks, E., Ellersick, R. and Grose, T., 2004. Eclipse Modeling Framework. Published by Addison Wesley Professionnal.

Dijkman, R. M., Quartel, D. A. C, Ferreira Pires L. and van Sinderen M. J., 2004. A Rigorous Approach to Relate Enterprise and Computational Viewpoints. In Proceding of the 8th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2004), Monterey, California.

Eclipse project, 2006. http://www.eclipse.org/

Gervais, M. P., 2003. ODAC: An Agent-Oriented Methodology Based on ODP. Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers.

Hashimoto, D., Miyazaki, H. and Tanaka, A., 2005. UML 2.0 Models for ODP Engineering/Technology Viewpoints. Workshop on ODP for Enterprise Computing (WODPEC 2005), in conjunction with the 9th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2005). Enschede, The Netherlands.

IEEE, 2000. IEEE Recommended practice for architectural description of software-intensive system. IEEE Std 1471–2000.

ISO/IEC, 2005. Information Technology Open Distributed Processing. Use of UML for ODP system specifications. Committee Draft.

ISO/IEC, 1995. Open Distributed Processing *Reference Model Part 1-4*, ITU-T Spec. ITU-T 901..4 and ISO/IEC Spec. ISO/IEC 10746-1..4.

OMG, 2001. Model Driven Architecture – Architecture board ORMSC – document number ormsc/2001-07-01 – OMG-2001.

Picault, A., Bedu, P., Le Delliou, J., Perrin, J. and Traverson, B., 2004. Specifying Information System Architectures with DASIBAO - A standard based method. 6$^{th}$ International Conference on Enterprise Information Systems. Porto, Portugal.

Putman, J. R., 2001. Architecting with RM-ODP, Prentice-Hall.

Romero, J. R. and Vallecillo, A.., 2005. Modelling the ODP Computational Viewpoint with UML 2.0. In Proceeding of the 9th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2005). Enschede, The Netherlands.