

A new viewpoint for change management in RM-ODP systems

Nesrine Yahiaoui^{1,2}, Bruno Traverson¹, Nicole Levy²

¹ EDF R&D 1 avenue du Général de Gaulle F-92140 Clamart France

² UVSQ PRiSM 45 avenue des Etats-Unis F-78035 Versailles France

nesrine.yahiaoui@prism.uvsq.fr, bruno.traverson@edf.fr, nicole.levy@prism.uvsq.fr

Abstract

RM-ODP (Reference Model - Open Distributed Processing) is a standardized modelling framework to describe a distributed application, according to different viewpoints. The standard has been published for about nearly ten years but in a quite abstract form and in leaving some issues pending (how to describe the viewpoints, management of consistency between views). Several academic and/or industrial projects, such as ODAC (LIP6) and DASIBAO (EDF R&D), have proposed various approaches to solve these problems.

Our proposal supplements these past or in progress works so that the evolution of distributed application is better taken into account. Indeed, because of the "one-shot" approach generally adopted, when a view evolves, it is necessary to rebuild manually consistency with the other views. In addition, the methodologies generally suggested impose a "top-down" approach which is not adapted when existing systems may evolve according to a viewpoint.

We define a modelling framework that maintains consistency using explicit correspondence links which are established between the various views of the distributed application. These links memorize the relationships between the views, thus guaranteeing traceability.

Keywords. Multi-view system, ODP, UML, change management.

1. Introduction

In the real world, a three dimensional object may be mapped according to several viewpoints (front face, back face, left face, right face). The result of the projection of an object according to a viewpoint is

called a view of the object. For example, if we project a cube on its six faces, we observe that each view is a square. The property of a cubic object is that the six squares that we obtained are identical. If we modify, for example, the size of one square, we must modify also the other views (squares), in order to preserve the constraint of the cube. The various views are not independent, since the global constraint must be enforced in the various views.

The concept of viewpoint also exists in the computer science world. A system may be described according to several viewpoints. A viewpoint allows to break up the system and to focus on a particular aspect of the system. A viewpoint introduces specific concepts which take into account the aspect considered by this viewpoint.

Thus, RM-ODP standard enables to describe systems according to five viewpoints: enterprise (objective, business rules), information (data), computational (functional decomposition), engineering (communication and deployment) and technology (hardware and software infrastructure). If we apply these five viewpoints to the same system, the obtained views must be consistent, i.e. the specification of a view should not conflict with the specification of another view. When a modification occurs in a view, it may involve a modification in other views in order to preserve the consistency of the system.

In our study, we are interested in the evolution of multi-view systems that are described according to RM-ODP viewpoints. The evolution means that each view may be modified for various reasons, for example: change of quality of service, environment, and objective. When a modification occurs in a view, several changes may be performed in other views, in order to maintain the consistency of the system.

In this article, we propose a modelling framework that maintains consistency using correspondence links, which are established between the various views of the system. These links memorize the relationships between the views, thus guaranteeing traceability.

This article is structured in five parts: The first part presents the RM-ODP standard that we use to describe multi-view systems. The second part describes some projects using and supplementing the RM-ODP standard. The third part introduces our modelling framework and insists on the interest to make explicit the correspondence links in a multi-view system. The fourth part gives some examples using our modelling framework for change management. Finally, we conclude in the fifth part.

2. RM-ODP reference model

RM-ODP reference model (Reference Model - Open Distributed Processing) is a standardized modelling framework to describe a distributed application according to five viewpoints [7] [8] [9] [13]. Each viewpoint gathers a set of concepts that take into account the concern associated with the viewpoint. Here are the five viewpoints of the standard:

1. **Enterprise.** It introduces the concepts necessary to represent a system in the context of an enterprise on which it operates. It is interested to the objective and the policies of a system. A system is then represented by a community which is a configuration of enterprise objects formed to achieve a goal.
2. **Information.** It is focused on the semantics of information and the treatment carried out on information. A system is then described by information objects, relationships and behavior. The description is expressed through the use of three diagrams named invariant, static and dynamic.
3. **Computational.** It allows a functional decomposition of the system. The various functions are fulfilled by objects that interact thanks to their interfaces. The basic concepts define the type of the interfaces which the computational objects support, the way in which the interfaces can be bound, and the forms of interaction which can take place.
4. **Engineering.** It is focused on deployment and communication mechanisms used in the system. It

defines communication concepts (channel, stub, skeleton) and deployment concepts (cluster, capsule,...).

5. **Technology.** It describes the implementation of a system in term of configuration of technical objects representing the hardware and software components used in the implementation. The goal of such a description is to provide additional information for the implementation and the test, by selecting standard solutions for the components and the communication mechanisms.

3. RM-ODP projects and standardization

The standard has been published for about nearly ten years but in a quite abstract form and in leaving many issues pending (how to describe the viewpoints, management of consistency between views). That explains why it is still little used in industrial applications. Nevertheless, several academic and/or industrial projects were carried out to solve these problems. They generally choose UML (Unified Modelling Language) [14] as modelling language and define one or more ways of guaranteeing consistency between views.

ODAC (Open Distributed Application Construction) is a project of the LIP6 laboratory. It defines a method for constructing systems according to an ODP semantic. It prescribes an order in the use of viewpoints [12]. This method proposes three UML profiles corresponding to enterprise, information and computational viewpoints. First, the designer describes the enterprise view, then the information view and after that the computational view, by complying to the correspondence rules specified by the method. These rules describe complementary and consistent views. The set of these three views are named behavioural specification because they describe the behaviour of the system independently of any platform. ODAC does not provide UML profiles for engineering and technology viewpoints, but it proposes an operational specification which results in the transformation of behavioural specification according to a target environment reflecting the real environment for execution, this environment being described by a UML profile. Two UML profiles exist for mobile agent environments [5] and for ANTS active networks [1].

DASIBAO (French acronym that means method for building Information System Architecture based on ODP) is a project from the EDF R&D (Electricity of

France Research and Development). It defines a method which is close to ODAC, because it constructs an ODP system by following a list of sequential steps [2]. Contrary to ODAC, it enables to build the five ODP views. First of all, the designer begins with the enterprise view, then the information view. The two obtained views are used to build the computational view, after that, she constructs the engineering view and finally the technical view. The transition from a view to another follows correspondence rules. The first three viewpoints are well described by UML profiles. However, the method proposes a graphical notation for the two last viewpoints, and a choice of UML profiles is not yet carried out.

In addition, OMG (Object Group Management) published in 2004 a set of UML profiles called EDOC (Enterprise Distributed Object Computing) [4]. In particular, ECA specification (Enterprise Collaboration Architecture) proposes profiles for the enterprise, information and computational viewpoints. The profiles corresponding to the engineering and technique viewpoints in other specifications and the rules of projection of the concepts of ECA are defined. Currently, the use of UML as a notation to describe the ODP viewpoints is in the process of being standardized by the ISO (International Organization of Standardization) [6].

4. Our modelling framework

The various views of an ODP system must be consistent between themselves. Consistency may be considered like a relationship between the views of the system in order to preserve the properties which are expressed in a different way according to the considered view.

In the methods which allow to build ODP systems according to a "top-down" approach [2] [12], the consistency of the system is verified at the time of the construction of the various views. The rules of transformation are applied to one or more views in order to construct one or more consistent views. Once the views are established, no information is saved on the relationships which exist between the any views, i.e. an element in a view does not have knowledge on the element or the elements from which it is build. However, even if these approaches build consistent systems, this consistency is lost as soon as one of the views is modified.

Our approach aims at providing a modelling framework, so that the designer can explicitly describe not only the views of his or her system (corresponding to the ODP viewpoints) but also the correspondences which exist between these views. More precisely, we want to allow the designer to connect the elements in the views which specify the same property of the system.

To illustrate our framework, we have chosen three viewpoints, the enterprise, the computational and the engineering viewpoints. We express for each viewpoint a meta-model that describes the appropriate concepts and the relationships that exist between them, the meta-models are inspired by [6]. In this article, we will not present the three meta-models but only the concepts that enable to connect the three views.

4.1 Correspondence rules

RM-ODP standard describes each concept defined in a viewpoint independently to other concept defined in other viewpoint, and does not stress on the concepts that allow to express the relationships which may exist between different viewpoints. We propose to extend ODP viewpoints to introduce a new concept named *correspondence rule*. Correspondence rule describes a constraint applied to concepts from two different viewpoints. A set of rules is thus already defined in an abstract way by the standard. These correspondence rules are between two viewpoints. Here are some correspondence rules:

1. Between enterprise and computational viewpoints, for instance, an enterprise object corresponds to computational object or a configuration of computational objects.
2. Between enterprise and engineering viewpoints, for instance, a policy may correspond to and determines transparency requirements that engineering objects support.
3. Between computational and engineering viewpoints, for instance, a computational binding, primitive or via a binding object, corresponds to a local or distributed binding (via a channel).

We propose to introduce these rules into the viewpoints in order to be able to connect ODP concepts defined in different viewpoints. The rules may be described as OCL constraints (Object Constraint Language) [11], they constrain the concepts having to be related and their multiplicity.

4.2 Correspondence links

In order to connect the views, we suggest to introduce a new viewpoint named link viewpoint. The link viewpoint has concepts that permit to connect the views two by two, in our case the enterprise view with the computational view, the computational view with engineering view. It is also possible to connect enterprise view with the engineering view but it represents a redundancy which we prefer to avoid, because these are implicitly linked by a transitivity relationship [Fig 1]:

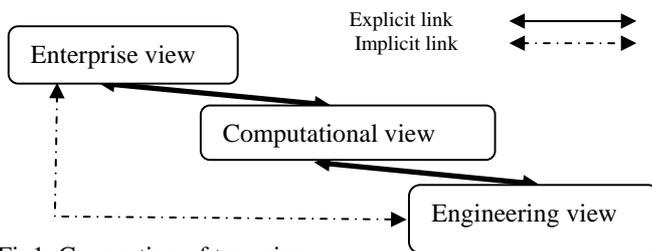


Fig1. Connection of two views

The relationship between two views of the application is called a correspondence link. A correspondence link is established between elements belonging to two different views to indicate that there is a certain relationship between them. The link must comply to the correspondence rules stated between the two considered viewpoints. The link may be governed by one or more correspondence rules.

The link must be bidirectional, i.e. it must be possible to navigate it from any of the two views. The bidirectionality enables us in particular to let no constraint on the way of navigating the system views. As we will see in the following part, each view may be modified. The interest of a bidirectional link is to find the correspondences with the other views, no matter which view is modified. Moreover, the links established between two ODP views has 1-N cardinality; indeed, it may be possible to connect an element in a view to one or more elements in another view. It is not a priori necessary to define links of N-M cardinality within the framework of our study because we did not find any use of this type of link in the ODP standard.

We propose the following link meta-model which specify the concepts that describe a correspondence between two views [Fig 2].

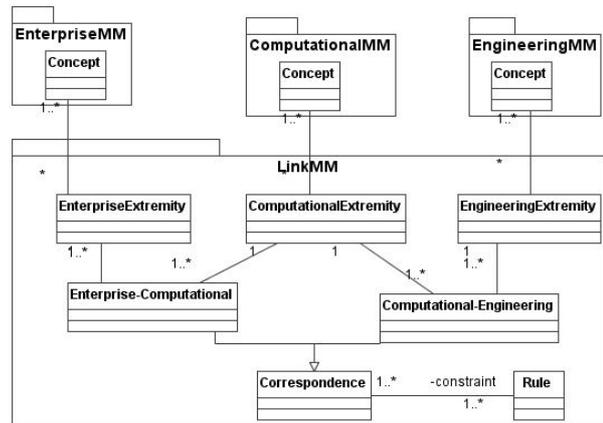


Fig 2. Link meta-model

The correspondence relationships are instances of the correspondence class of the link meta-model (*LinkMM*). The **correspondence** relationship is restricted by **rules**, these rules are those described previously.

The correspondence may be either a correspondence between enterprise and computational views (**enterprise –computational**), or between computational and engineering views (**computational-engineering**). Each extremity correspondence (**EnterpriseExtremity**, **ComputationalExtremity**, **EngineeringExtremity**) identifies elements belonging to the suitable view and having to be connected. Thus an instance of the correspondence class connects elements in a view with elements in another view.

The figure [Fig 3] illustrates an example of correspondence relationship. **R1** and **R2** are two correspondence links, **R1** connects the **EntC1** object of the enterprise view with the **ComC1**, **ComC2** and **ComC3** objects of the computational view, under the constraint of the **rule1** which establishes that EntC1 is an enterprise object and ComCi (i=1,2,3) are computational objects. **R2** connects the **ComC3** object of the computational view with the **EngC1** and **EngC2** objects of the engineering view, under the constraint of the **rule2** which establishes that ComC3 is computational object and that EngCi (i=1,2) are basic engineering objects.

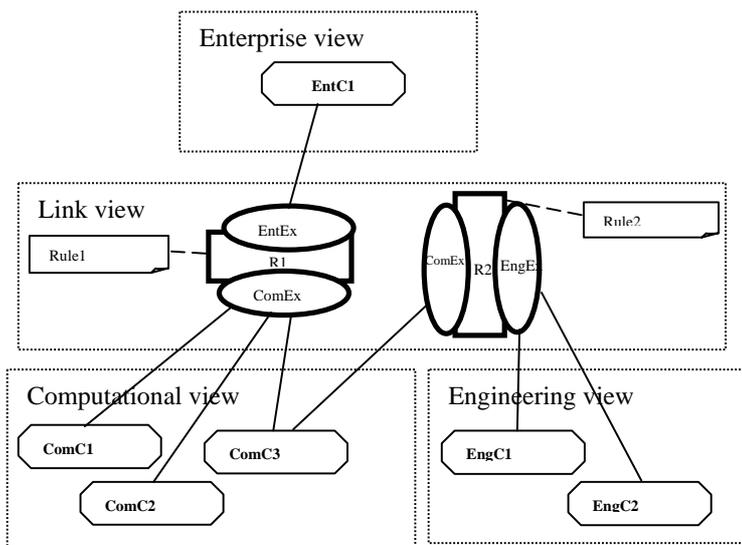


Fig3. Representation of correspondence link

5. Using an framework for change management

The specification of a system is not static and may evolve in order to answer to new user requirements or new constraints of the technical environment. The designer of the system takes into account these changes in the concerned views by adding, removing and/or modifying elements in the view.

In a RM-ODP vision, the system is described in different but dependent views; any action on a view may involve the same action or other actions in one or more other views. If the views are not explicitly linked, the designer does not have any means of knowing the elements in the other views which have a relationship with the modified element. Thus, change management is based only on his or her expertise and knowledge of the system.

Our modelling framework can facilitate the change management in a multi-view system thanks to the introduction of the link viewpoint. Contrary to the already existing UML profiles which allow to design views, according to viewpoints, that seem completely independent between them, we propose to extend ODP system descriptions, by adding a link viewpoint which permits to the designer to connect the views two by two. Moreover, our modelling framework guides the designer in his or her various activities. These activities fall in the three following categories.

1. Description of ODP system views.

The designer does not follow a particular method to build his multi-view system. The system may be even

described by a team of designers; each designer is responsible for the building of a view by modelling it according to the adequate UML profile. According to the considered view, the modelling framework prevents the designer to use other concepts not relevant for this view.

2. **Establishment of correspondence link** between the ODP system views. Once the modelling of the view completed, the obtained views are completely independent. The designer binds the views by using UML profile representing the link viewpoint. The correspondence relationships are thus established between elements in two different views. In our case, enterprise view with computational view, and computational view with engineering view. These links save and keep traces of the correspondence which may exist between the elements. As the correspondence links are constrained by rules, the modelling framework checks that these links verify the correspondence rules stated between the concerned views.

3. **Change management** in the various views. The designer may perform changes in the views by modifying, for example, one of their elements. The modelling framework indicates the elements which have a correspondence link with this element. Any action on an element generates an event which is delivered to a change manager. The event contains information about the element which emitted it (its identity and in which view it is). The change manager seeks whether there is correspondence links, in the link view, which connect this element. Once the correspondence links found, it is possible to know the other elements which are connected to the modified element. The change management can take one of the following forms:

- a. **Manual.** The change manager retrieves only the correspondence links referring to the modified element. It informs the designer by a message that points out the modified element, the action carried out and the elements having to be checked; the designer will act according to his or her knowledge of the system on the other views.
- b. **Semi-automatic.** The change manager can associate to the change a set of modification strategies which act in the other views in order to modify them. If a modification was not planned, the change manager informs the designer. A new strategy concerning this case could then be added

later to automate its management. If, for example, an action of deletion was performed in an element of the view, the associated strategy may specify that the elements having a correspondence link with this element must be removed from the other views, if there is not other correspondence links with other elements belonging to the modified view.

- c. **Automatic.** The change manager is autonomous and can act in the other views whatever the modification performed. This kind of manager is able to build new scenarios which were not planned at the time of its design.

6. Conclusion and perspectives

First of all, we have reminded the principles of the RM-ODP viewpoints and the various past and in progress projects and standardization efforts around it. Then, we presented our modelling framework that integrates, in particular, the management of correspondence links between various views of an application through the introduction of the link viewpoint. We showed its interest for the various activities of the designers of a multi-view system, in particular the activities relative to its evolution.

Currently, we have implemented the enterprise, computational, engineering and link meta-models. The first three meta-models were not presented in this article due to lack of space. This implementation is in the form of a "plug-in" in Eclipse [3].

In addition, we proposed a framework for change management in only one view. It has been validated by a prototype developed in the form of a "plug-in" [15] in MagicDraw [10]. A use case resulting from the context of the intelligent network for electric systems enabled us in parallel to better apprehend the requirements in term of business evolution.

The concept of correspondence link goes beyond the scope of ODP systems and may be generalized for any system which can be modelled according to several viewpoints such as, for example, PIM (Platform Independent Model) and PSM (Platform Specific Model) models with regards to the MDA (Model Driven Architecture) approach.

REFERENCES

- [1] S.Bouzitouna, M.Elias, P.Spathis, M-P.Gervais et K. Thai, Création de services actifs dans ANTS, 4ème Colloque francophone sur la gestion de réseau et de service (GRES'01), Maroc. 2001
- [2] A.Picault, P.Bedu, J.Le Delliou, J.Perrin, B Traverson. Specifying Information System Architectures with DASIBAO - A Standard Based Method. Proceedings of the 6th International Conference on Enterprise Information Systems. ICEIS 2004.
- [3] <http://www.eclipse.org/>
- [4] OMG. UML Profile for Enterprise Distributed Object Computing (EDOC). OMG Document. 2004. <http://www.omg.org/technology/documents/formal/edoc.htm>
- [5] M-P. Gervais and F. Muscutariu. A UML Profile for MASIF Compliant Mobile Agent Platform. OMG's Second Workshop on UML for Enterprise Applications: Model Driven Solutions for the Enterprise USA. 2001.
- [6] ISO. Information Technology – Open Distributed Processing – Use of UML for ODP system specifications. Committee Draft. 2005.
- [7] ISO/IEC 10746-2.Information technology - Open Distributed Processing – Reference Model: Foundations. 1996.
- [8] ISO/IEC 10746-3.Information technology - Open Distributed Processing – Reference Model: Architecture. 1996.
- [9] ISO/IEC 10746-1.Information technology - Open Distributed Processing – Reference Model: Overview. 1998
- [10] <http://www.magicdraw.com/>
- [11] OMG. UML 2 OCL Final Adopted Specification. <http://www.omg.org/cgi-bin/doc?ptc/2003-10-14.pdf>
- [12] M-P.Gervais, ODAC: An Agent-Oriented Methodology Based on ODP, Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers, Vol. 7, n°3, pp199-228, 2003
- [13] J-R.Putman. Architecting with RM-ODP. Prentice Hall PTR, Published October 2000.
- [14] OMG. Unified Modeling Language Specification. <http://www.omg.org/docs/formal/03-03-01.pdf>
- [15] N.Yahiaoui. Prototype using evolution contracts for reconfiguration. Research report EDF R&D. 2005