Journal of Theoretical and Applied Information Technology





www.jatit.org

# MODEL OBJECT OF RM-ODP STANDARD IN DYNAMIC DISTRIBUTED DATABASES

**[1]JALAL LAASSIRI, SAID EL HAJJI, YOUSSEF BALOUKI, GHIZLANE ORHANOU, MOHAMED BOUHDADI**

Department of Mathematics and Computer Science, University Mohammed V Faculty of Sciences,

BP 1014, Rabat Morocco

**ABSTRACT**

Distributed databases have invested considerable effort and system resources in the development and adoption of dynamic distributed databases services. In order to sustain the quality of their services, Business process management need to solve the problem of efficient and secure electronic exchange and processing of system database. A major difficulty in this distributed deployment is the fact that these interconnected systems are heterogeneous and they may operate in multiple organizational domains. This paper demonstrates how the ISO/RM-ODP standard offers a general framework to design and develop an open distributed system attuned to dynamic distributed databases environments.

The purpose of the RM-ODP is to define such a framework. The Reference Model for Open Distributed Processing (RM-ODP) provides a framework within which support of distribution, inter-working and portability can be integrated. It defines: an object model, architectural concepts and architecture for the development of ODP systems in terms of five viewpoints. However, RM-ODP is a meta-norm, and several ODP standards have to be defined. In this paper, we report on the definition and address of the syntax and semantics for a fragment of ODP object concepts defined in the RM-ODP foundations part and in the information language. These concepts are suitable for describing and constraining dynamic distributed databases information specifications.

**Keywords:** *RM-ODP, dynamic distributed database (DDB), Meta-modeling Semantics, UML/OCL.*

## 1. OVERVIEW OF THE RM-ODP STANDARD

Distributed systems are an important development in computing technology which is concerned with the delivery of constantly expanding data to points of query. Collections of data in the forms of partitions or fragments can be distributed or replicated over multiple physical locations. Local autonomy, synchronous and asynchronous data distributions are examples of distributed database design schemas which can be implemented depending on business needs and data Sensitivity/confidentiality.

Data reliability and availability are basic requirements for system design. Reliability is the possibility that a system is running at a certain point in time while availability is the probability that the system is continuously available during a time interval. Both data reliability and availability can be enhanced by distributing data and DDB software over several sites. The database administrator carries the responsibility of ensuring that the distributed nature of the system is transparent.

The rapid growth of distributed processing has led to a need of coordinating framework for the standardization of Open Distributed Processing (ODP).

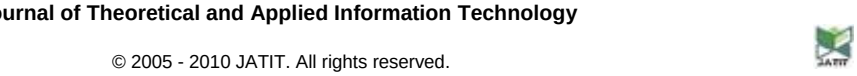
The open distributed processing (ODP) computational viewpoint describes the functionality of a system and its environment, in terms of a configuration of interacting objects at system interfaces, independently of their distribution. In addition, Quality of service (QoS) contracts and service level agreements are an integral part of any computational specification, which is specified in ODP in terms of environment contracts.

The Reference Model for ODP (RM-ODP) is a framework for the construction of open distributed systems [1]-[4]. It creates an architecture supporting distribution, networking and portability.

Vol. 16 No.2 June , 2010 pp [129 – 134]



**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615





The foundations part [2] contains the definition of concepts and analytical framework for normalized description of (arbitrary) distributed processing systems. These concepts are gathered in several categories including basic modeling concepts, specification concepts, organizational concepts, and structuring concepts.

The architecture part [3] contains specifications of the required characteristics that qualify distributed processing to be open. It defines a framework containing:

Five viewpoints called: enterprise, information, computation, engineering and technology; which provide a basis for the ODP systems specification.

A viewpoint language for each viewpoint, defining concepts and rules for specifying ODP systems from the corresponding viewpoint.

The ODP functions are required to support ODP systems. The transparency prescriptions show how to use the ODP functions to achieve distribution transparency.

In other words, the first three viewpoints points do not take into account neither distribution nor heterogeneity inherent problems. This principle corresponds closely to the concepts of PIM (Platform Independent Model) and PSM (Platform Specific Model) models in MDA (Model Driven Architecture) architecture [5]. However, RM-ODP is a meta-norm [6] and cannot be directly applied. Indeed, for instance, the viewpoint languages are abstract in sense that they define what concepts should be supported, not how these concepts should be represented [7]. It is important that RM-ODP does not use the term language in its largest sense: a set of terms and rules for the construction of statements from terms; it does not propose any notation for supporting viewpoint languages.

This approach provides a formalization of well-established design practices of abstraction and encapsulation. We define the meta-models for concepts DDB. Figure 1 defines the context free syntax for the core of DDB object concepts.

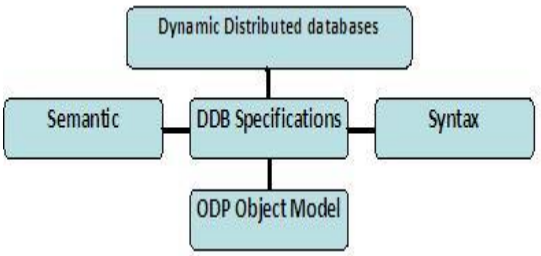



*Figure 1. Dynamic Distributed databases model*

Using the Unified Modeling Language (UML)/OCL (Object Constraints Language) [9], [10] we defined and explain the semantics of the generic model DDB Object more clearly, the Alloy [28], formalism was used. Alloy is a simple modeling language that allows a modeler to describe the conceptual space of a problem domain. Using Alloy, specifying the RM-ODP semantic domain can be obtained.

A part of UML meta-model itself has a precise semantic [12], [13] defined using denotational meta-modeling approach. A denotational approach [14] is realized by a definition of the form of an instance of every language element and a set of rules which determine which instances are denoted or not by a particular language element. For testing ODP systems [2], [3], the current testing techniques [15], [16] are not widely accepted. A new approach for testing, named agile programming [17] or test first approach [19], is being increasingly adopted. The principle is the integration of the system model and the testing model using UML meta-modeling approach [20], [21]. This approach [26].is based on the executable UML [27, 25].

RM-ODP conceptual elements from the semantic domain can be partitioned in the following way:

**Model RM-ODP** {

Domain {ODP_Concepts}

State {partition … BasicModellingConcepts, SpecificationConcepts : static ODP_Concepts

}

**Code Fragment 1. RM-ODP model Object**

## 2. DISTRIBUTED DATABASES SYNTAX

Basic DDB objects are modeled by atomic DDB objects. More complex information is modeled as composite DDB objects which, as any other ODP object, behavior, state, identity and encapsulation. Its type is a predicate characterizing a collection of DDB objects, which their class is the set of all DDB objects satisfying a given type.

DDB objects template specifies the common features of a DDB objects collection in sufficient detail that a DDB objects can be instantiated using it. It may reference static, invariant and dynamic schema.

An **invariant schema** is a set of predicates on one or more DDB objects which must always be true. The predicates constrain the possible states and state changes of the objects on which they





**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615

apply. ODP also notes that an invariant schema can specify the types of one or more DDB objects; that will always be satisfied by whatever behavior the objects might exhibit.

A **static schema** defines the state of one or more DDB objects, at some point in time, subject to the constraints of any invariant schema.

A dynamic schema is a requirement of the allowable state changes of one or more DDB objects, subject to the constraints of any **invariant schema**. A dynamic schema specifies how the information can evolve as the system operates. In addition to describing state changes, dynamic schema can also create and delete DDB objects, and allow reclassifications of instances from one type to another. Besides, in the DBDB challenges, a state change involving a set of objects can be seen as an interaction between those objects. Not all the objects involved in the interaction need to change state; some of the objects may be involved in a read-only manner [29].

Figure 2 defines the context free syntax for the information language of DDB object.

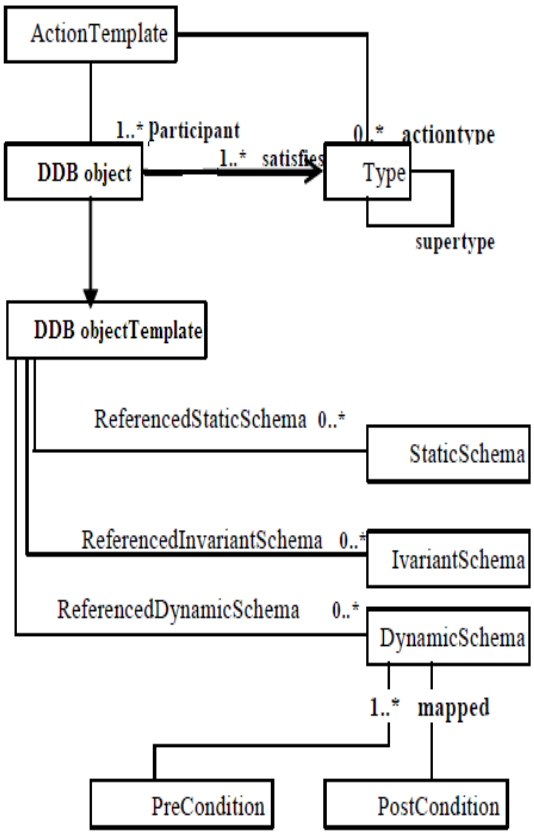



*Figure2. DDB Information concepts*

## 3. SEMANTICS DOMAIN

The semantics of a UML/OCL model is given by constraining the relationship between a model and possible instances of that model (see Figure 3). It means constraining the relationship between expressions of the UML abstract syntax for models and expressions of the UML abstract syntax for instances. We define a model to specify the ODP information viewpoint. That is, a set of DDB objects, their relationships and behaviors. This model defines Semantic Domain (figure 3).

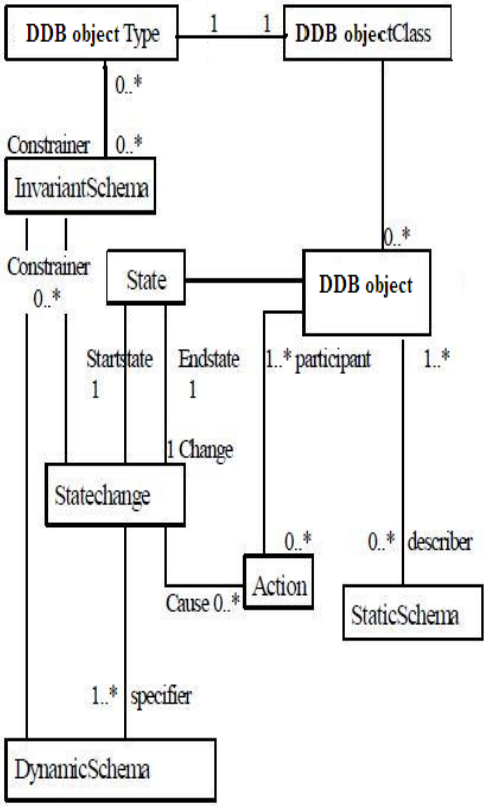



*Figure3. Semantic Domain*

A system can only be an instance of a single system model, because it is self contained and disjoint from other models. On the other side, objects are instances of one or more object templates; they may be of one or several types. With no further constraints, it is possible for an object to change the templates of which it is an instance; thus this meta-model supports dynamic types.

There is one well-formedness rule for instances, which are given bellow:





**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615

**Context Time inv:**

For all (o: DDBObject, t: Time | t.instant -> notEmpty implies o.state -> notEmpty)

**Context Precondition inv:**

For all (prec: Dynamic schema.Precondition , o : DDBObject |exists ( s : State) | o.mappedTo = prec and o.state_start = s)

**Context Postcondition inv:**

For all (postc: dynamic schema.Postcondition, o: DDBObject | exists(s: State) | o.mappedTo = postc and a.state_end = s)

## 4. MEANING FUNCTION

Other invariants are required to constraint the relationships between models and instances. These constitute the semantics which are the subject of this section. The semantics for the UML/OCL based language defined by the relationship between a system model and its possible instances (systems). The constraints are relatively simple, but they demonstrate the general principle. Firstly there are two constraints related to DDB information objects and links, respectively. The first shows how inheritance relationships can force a DDB information object to be of many DDB Information Object Template.

**Context o: object inv:**

The templates of o must be a single template and all the parents of that template

o.of->exists (t | o.of=t->union (t.parents))

The second ensures that a link connects objects of templates as dictated by its role.

**Context l: link inv:**

DDB Information Objects which are the source/target of link have templates which are the source/target of the corresponding roles.

(l.of.source)->intersection (l.source.of)-> notEmpty and (l.of.target)->intersection(l.target.of)->notEmpty

## 5. FUNCTION TRADER AND ACTIVITY DIAGRAM FOR THE DDB PROCESS

The use of RM-ODP as a standard for designing a distributed system enables and supports the development of systems with certain desired characteristics.

The DDB behavior of the class is described using an activity graph [8]. The activity graph for the DDB process is shown in figure 4. The activities, such as invoke, are shown as the rectangles with rounded corners. The actions to be performed are shown as Entry conditions to the activity [30]. For example, action constraint (a variable) is set to the result of the check service. The partners with which the process communicates are represented by the UML partitions (also known as swimlanes): Trader, Client and Server. The activities that involve a message send or receive operation to a partner appear in the corresponding partition. The arrows indicate the order in which the process performs the activities. Note that the assignment activity is not in a swimlane; it depicts an action that takes place within the process itself.

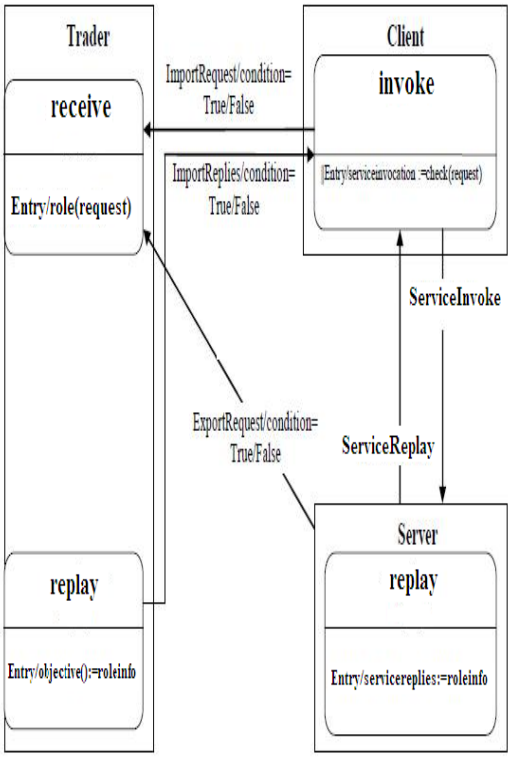



*Figure4. Activity Diagram for the DDB Process*

The reply activity returns a response back to the client, completing the execution of the process. Each activity has a descriptive name and an entry action detailing the work performed by the activity.





**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615

## 6. CONCLUSION

The Reference Model for Open Distributed Processing (RM-ODP) provides a framework which supports distribution, inter-working and portability can be integrated. However, the ODP viewpoint languages define what concepts should be supported, not how these concepts should be represented. We have applied RM-ODP concepts specifications in order to design a dynamic distributed databases platform. In addition, the UML standard has adopted a meta-modeling approach to define the abstract syntax and semantic domain of DDB. One approach to define the formal semantics of a language is denotational: essentially elaborating the value or instance denoted by an expression of the language in a particular context. However, when we use the denotational meta-modeling approach in this paper, we defined the UML/OCL based syntax and semantics of a language for a fragment of ODP object concepts described in the foundations part and in the information viewpoint language. Indeed, these concepts are suitable to define and constrain DDB information specifications. In parallel, we are applying the same approach to define a language of concepts characterizing dynamic behavior of dynamic distributed database.

## REFRENCES:


[1]. ISO/IEC, Basic Reference Model of Open Distributed Processing-Part1: Overview and Guide to Use, ISO/IEC CD 10746-1, July 1994.

[2]. ISO/IEC, RM-ODP-Part2: Descriptive Model, ISO/IEC DIS 10746-2, February 1994.

[3]. ISO/IEC, RM-ODP-Part3: Prescriptive Model, ISO/IEC DIS 10746-3, February 1994.

[4]. ISO/IEC, RM-ODP-Part4: Architectural Semantics, ISO/IEC DIS 10746-4, July 1994.

[5]. OMG, the Object Management Architecture, OMG, 1991. http://www.omg.org

[6]. M. Bouhdadi et al. A Methodology for the Development of Open Distributed Systems, Proc. JDIR'98, Paris France October 1998, pp. 200-208

[7]. ISO/IEC, ODP Type Repository Function, ISO/IEC JTC1/SC7 N2057, January 1999.

[8]. ISO/IEC, the ODP Trading Function, ISO/IEC JTC1/SC21, June 1995.

[9]. J.M. Spivey, The Z Reference manual, Prentice Hall, 1992.

[10]. IUT, SDL: Specification and Description Language, IUT-T-Rec. Z.100, 1992.

[11]. ISO and IUT-T, LOTOS: A Formal Description Technique Based on the Temporal Ordering of Observational Behavior, ISO/IEC 8807, August 1998.

[12]. H. Bowman et al. FDTs for ODP, Computer Standards & Interfaces Journal, Elsevier Science Publishers, Vol.17, No.5-6, 1995, pp. 457-479.

[13]. J. Rumbaugh et al., The Unified Modeling Language, Addison Wesley, 1999.

[14]. B. Rumpe, A Note on Semantics with an Emphasis on UML, Second ECOOP Workshop on Precise Behavioral Semantics, Technische Universitaty unchen publisher, 1998.

[15]. A. Evans et al., Making UML precise, OOPSLA'98, October 1998,Evans et al. The UML as a formal notation, UML'98, France June 1998, LNCS 1618, Springer Berlin, 1999, pp. 336-348.

[16]. J. Warner and A. Kleppe, the Object Constraint Language: Precise Modeling with UML, Addison Wesley, 1998.

[17]. M. Bouhdadi et al, An UML-based Meta-language for the QoS-aware Enterprise Specification of Open Distributed Systems, IFIP TC5/WG5.5 Third Working Conference on Infrastructures for Virtual Enterprises (PRO-VE'02), May 1-3 Sesimbra Portugal, Kluwer Vol. 213 (IFIP Conference Proceeding series), 2002. Collaborative Business Ecosystems & Virtual Enterprises IFIP Series Vol. 85 Springer Boston 2002.

[18]. S. Kent, S. Gaito, N. Ross. A meta-model semantics for structural constraints in UML,, In H. Kilov, B. Rumpe, and I. Simmonds, editors, Behavioral specifications for businesses and systems, chapter 9, pages 123-141. Kluwer Academic Publishers, Norwell, MA, September 1999.

[19]. D.A. Schmidt, Denotational semantics: A Methodology for Language Development, Allyn and Bacon, Massachusetts, 1986.

[20]. Myers, G. The art of Software Testing, John Wiley &Sons, New York, 1979






**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615

[21]. Binder, R. Testing Object Oriented Systems. Models. Patterns, and Tools, Addison-Wesley, 1999

[22]. Cockburn, A. Agile Software Development. Addison-Wesley, 2002.

[23]. Bernhard Rumpe. Agile Modeling with UML. Habilitation Thesis, Germany, 2003.

[24]. Beck K. Column on Test-First Approach. IEEE Software, 18(5):87-89, 2001

[25]. Wegmann, A. and A. Naumenko. Conceptual Modeling of Complex Systems Using an RM-ODP Based Ontology. in 5th IEEE International Enterprise Distributed Object Computing Conference - EDOC 2001. 2001.

[26]. Bernhard Rumpe, Executable Modeling with UML. A vision or a Nightmare? In Issues & Trends of Information Technology Management in Contemporary Associations, Seattle. Idea group Publishing, Hershey, London, pp. 697-7001. 2002 Author, Title of the Book, Publishing House, 200X.

[27]. A.Naumenko, A.Wegmann, “Proposal for a formal foundation of RM-ODP concepts » conference woodpecker 2001.

[28]. ISO/IEC, Basic Reference Model of Open Distributed Processing-Use of UML for ODP system specifications, May 2006.

[29]. Naumenko, A., et al. A Viewpoint on Formal Foundation of RM-ODP Conceptual Framework, Technical report No. DSC/2001/040, July 2001, EPFL-DSC ICA.

[30]. Briand L. and Labiche Y. A UML-based Approach to System testing. In M. Gogolla and C. Kobryn (eds): “UML” – The Unified Modeling Language, 4th Intl. Conference, LNCS 2185. Springer, 2001 pp. 194-208.



**ISSN**: 1817-3195 / **E-ISSN**: 1992-8615