Volume 2 No.10, October 2012 ISSN 2224-3577
International Journal of Science and Technology



http://www.ejournalofsciences.org

# Modelling Distributed Computing Management through RM-ODP

**Laassiri Jalal[1], Krit Salahddine[2], El Hajji Said[3]**
[1] Department of informatics, Faculty of Sciences Kenitra, University Ibn Tofail, BP 33, Morocco
[2] Polydisciplinary Faculty of Ouarzazate, University Ibn Zohr, BP/638, Morocco
[3] Department of Mathematic Informatics, University Mohamed V-Agdal, BP 1040, Morocco

## ABSTRACT

Actually in distributed computing, data frequently resides on multiple sites inside an organization. This data might be managed by several Database Management Systems for multiple reasons such as performance, scalability, access and management. This paper describes specification and the core behavior concepts (time, action, state, behavior, interaction and the binding object). Secondly describes and specifies the behavior by the activity diagram, and proposes how the concepts defined in Part 2 and Part 3 of RM-ODP can be combined to develop EA models, especially on the aspects of business functions and information and computational processing, we present the syntax and the structure of a BPEL Behavior Processes of Working Object, we focus on behavioral interaction. The behavior of an ODP system is determined by collecting all possible actions in which the system (acting as a data object), or any of its constituent data objects, might take part, together with a set of constraints on when these actions can occur. In order to specify the executable behavior of a system and to make the processes of the management executable and controllable, the Reference Model for ODP RM-ODP can be used as a meta-model for behavioral specifications.

**Keywords:** *RM-ODP, distributed computing, Behavior Business Process Model, UML, BPEL*

## I. INTRODUCTION AND VIEWPOINTS MODELING

Most complex system specifications are so extensive that no single individual can fully comprehend all aspects of the specifications. Furthermore, we all have different interests in a given system and different reasons for examining the system's specifications. A business executive will ask different questions of a system make-up than would a system implementer. The concept of RM-ODP viewpoints framework [1]-[4], therefore, is to provide separate viewpoints into the specification of a given complex system. These viewpoints each satisfy an audience with interest in a particular set of aspects of the system. Associated with each viewpoint is a viewpoint language that optimizes the vocabulary and presentation for the audience of that viewpoint.

Viewpoint modeling has become an effective approach for dealing with the inherent complexity of large distributed systems. Current software architectural practices [4], as described in IEEE 1471, divide the design activity into several areas of concerns, each one focusing on a specific aspect of the system.

A viewpoint is a subdivision of the specification of a complete system, established to bring together those particular pieces of information relevant to some particular area of concern during the analysis or design of the system. Although separately specified, the viewpoints are not completely independent; key items in each are identified as related to items in the other viewpoints. Moreover, each viewpoint substantially uses the same foundational concepts (defined in Part 2 of RM-ODP) [2]. However, the viewpoints are sufficiently independent to simplify reasoning about the complete specification. The mutual consistency among the viewpoints is ensured by the architecture defined by RM-ODP [1]-[4], and the use of a common object model provides the glue that binds them all together.

More specifically, the RM-ODP framework provides five generic and complementary viewpoints on the system and its environment:

The enterprise viewpoint, which focuses on the purpose, scope and policies for the system.

The information viewpoint, which focuses on the semantics of the information and the information processing performed.

The computational viewpoint, which enables distribution through functional decomposition on the system into objects which interact at interfaces.

The engineering viewpoint, which focuses on the mechanisms and functions required to support distributed interactions between objects in the system.

The technology viewpoint, which focuses on the choice of technology of the system.

Each viewpoint language defines 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. The first three viewpoints do not take into account the distribution and heterogeneity inherent problems. This corresponds closely to the concepts of PIM (Platform Independent Model) and PSM (Platform Specific Model) models in the OMG MDA architecture. However, RM-ODP can not be directly applicable [5]. In fact, RM-ODP only provides a framework for the definition of new ODP standards. Which include standards for ODP functions [6-7]; standards for modeling and specifying ODP systems; standards for programming, implementing, and testing ODP systems.

We treated the need of formal notation for behavioral concepts in the Computational language [8]. Indeed, the viewpoint languages are abstract in the sense that they define what concepts should be

supported, not how these concepts should be represented. It is important to note that, RM-ODP uses the term language in its broadest sense: "a set of terms and rules for the construction of statements from the terms". It does not propose any notation to support the viewpoint languages. Using the Unified Modeling Language (UML)/OCL (Object Constraints Language) [9, 10] we defined a formal semantic for a fragment of ODP behavior concepts defined in the RM-ODP foundations part and in the enterprise language [11]. These concepts (time, action, behavior constraints and policies) are suitable for describing and constraining the behavior of ODP enterprise viewpoint specifications.

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 is based on the executable UML [22]. Executable UML is a major innovation in the field of software development. Use it to produce a comprehensive and understandable model of a solution independent of the organization of the software implementation. It is a highly abstract thinking tool that aids in the formalization of knowledge, and is also a way of describing the concepts that make up abstract solutions to software development problems.

In this context, BPEL (Business Process Execution Language for Web Services) (BPEL4WS or BPEL for short) to specify process behavior based on interaction and the binding object in the ODP systems. The BPEL is an XML-based standard for defining how you can combine Web services to implement business processes. It builds upon the Web Services Definition Language (WSDL) and XML Schema Definition (XSD). This article specifies the behavior processes by the activity diagrams, and generates the corresponding BPEL and computational files to implement that process. This capability is used to highlight some benefits of the Object Management Groups (OMG) Model Driven Architecture (MDA) initiative: raising the level of abstraction at which development occurs; which, in turn, will deliver greater productivity, better quality, and insulation from underlying changes in technology.

The paper is organized as follows, First introduction and viewpoints modeling, Section 2, we give both UML and the core behavior concepts (time, action, state, behavior, interaction and the binding object), Section 3 describes and specifies the behavior by the hierarchical diagrams and Section 4, proposes how the concepts defined in Part 2 and Part 3 of RM-ODP can be combined to develop EA models, especially on the aspects of business functions of information and computational processing. Section 5 We present the analyses for syntax and the structure of a BPEL Behavior Processes of Working Object, we focus on behavioral interaction ,Section 6 concludes the article and points out our future work.

## II. RM-ODP and UML for Representation of the Enterprise Model

### a. RM-ODP and UML

Currently there is growing interest in the use of UML for system modelling. However, there is no widely agreed approach to the structuring of such specifications. This adds to the cost of adopting the use of UML for system specification, hampers communication between system developers and makes it difficult to relate or merge system specifications where there is a need to integrate IT systems.

Although the ODP reference model provides abstract languages for the relevant concepts, it does not prescribe particular notations to be used in the individual viewpoints. The viewpoint languages defined in the reference model are abstract languages in the sense that they define what concepts should be used, not how they should be represented. It defines a set of UML Profiles, one for each viewpoint language and one to express the correspondences between viewpoints, and an approach for structuring them according to the RM-ODP principles. The purpose of "UML4ODP" to allow ODP modelers to use the UML notation for expressing their ODP specifications in a standard graphical way; to allow UML modelers to use the RM-ODP concepts and mechanisms to structure their large UML system specifications according to a mature and standard proposal; and to allow UML tools to be used to process viewpoint specifications, thus facilitating the software design process and the enterprise architecture specification of large software systems.

### b. Pharmacy Example

Let us consider an example of a pharmacy whose management decides to provide the company's services via the Internet. The management has a goal to specify the services that the pharmacy can provide its customers with and to describe how to implement them using business and IT resources. A medicine -selling market contains a MedicineValueNetwork and a Customer. The value network consists of three companies: a pharmacy company named pharmacyCo (responsible for the service of processing the orders placed by the customer), a shipping company called ShipCo (responsible for shipping the medicine ordered) and a publishing company PubCo (responsible for supplying the medicine that were ordered but not yet available in the inventory of the pharmacy company). The departmental structure of the pharmacy company shows two departments: one for coping with the purchasing data (PurchasingDep) and the other for managing an inventory of medicine (WarehouseDep). We might have an additional level showing the IT infrastructure of these departments.

Fig. 1 gives a simplified representation of the organizational structure and services in the pharmacy enterprise context using an ad-hoc notation. A regular rectangle represents a business entity or an IT system. A rounded rectangle can be attached to a

regular rectangle to represent the main service offered by the business entity or the IT system drawn under the regular rectangle. The smile symbol stands for people. The lines connecting these entities and people denote the containment hierarchy. In this paper, we needs to model the business entities, the IT systems (drawn under regular rectangles in Fig. 1) and their environment, the services offered to the customer by these entities, the company to company (and department to department) business processes, information flow and interaction between the IT system and an assistant who operates it and possibly the overall architecture of the IT system.

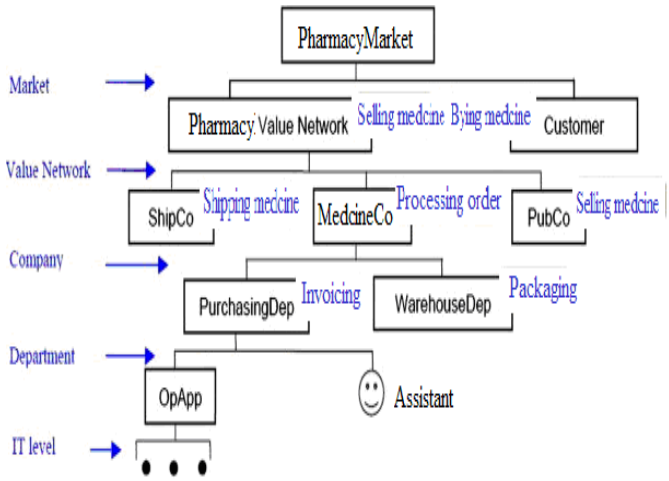


**Fig. 1: Representation of the Enterprise Model Object**

## III. EA MODELING FOR PHARMACY EXAMPLE

### a. Enterprise Specification

The RM-ODP Part 3 [3] specifies a viewpoint specification, namely the enterprise viewpoint specification or enterprise viewpoint for short that is dedicated to the enterprise level of ODP systems. This specification particularly matters if we explicitly represent the context where the ODP systems exist. Part 3 of RM-ODP also specifies four other viewpoint specifications with dedicated modeling concepts to target other aspects such as information processing, computing capabilities and technology choices, etc. of the ODP systems being modeled. For each of these specifications, RM-ODP defines a language that comprises concepts, rules and structures.

The enterprise language defined in the original version of RM-ODP has a few concepts that address the structure and policy in an enterprise specification [4]. However, viewpoints-specific concepts for behavior modeling are still missing in this language. Recently, a number of ISO/IEC international standards and recommendations came along with RM-ODP. Most notably, the Enterprise Language recommendation extends and refines the original RM-ODP enterprise language by introducing more concepts to better capture enterprise processes [19]. The information language provides us with concepts to model the semantics of information and information processing for enterprise objects that may have been represented in the enterprise specification. The focus here is to develop a commonly-agreed understanding of information exchanged when they collaborate.

In RM-ODP, one can build modeling constructs by applying concepts defined in Part 2 to the modeling concepts of Part 3.

We apply the interpretation concept of (RM-ODP concepts Part 2) Atomicity, Decomposition, Atomicity to viewpoint-specific concepts (RM-ODP concepts Part 3 , for Enterprise and information view points )(Enterprise Artifact, Enterprise Step , Enterprise Process, Information Object, Information Action, Dynamic Schema).

### b. Illustration of building blocks in the Pharmacy example

Here, in the figure 2 we represent by gray block working objects and blue rectangles – information objects; and rounded rectangles – localized actions.

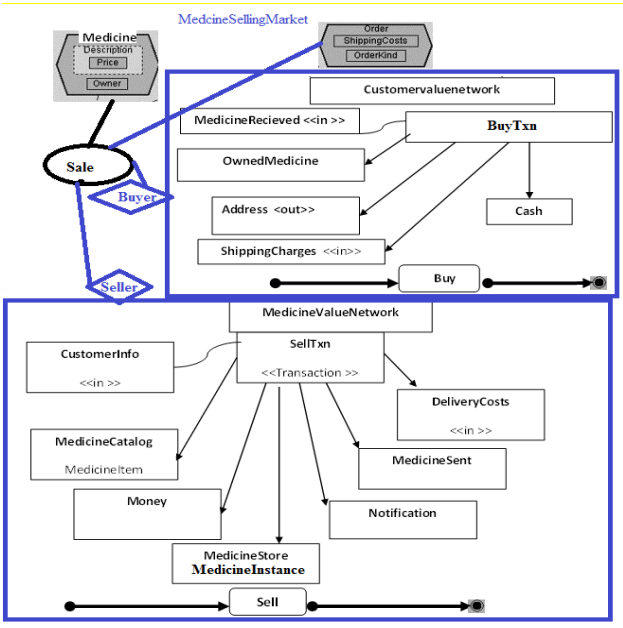


**Fig. 2. Hierarchical level of the pharmacy enterprise model**

Business collaboration sale takes place between business working objects MedicineValueNetwork and CustomerValueNetwork. It has Medicine and Order as parameter working objects. Being viewed as a whole, MedicineValueNetwork exhibits a localized action called Sell and a number of information objects that represent the money it has, the Pharmacy it operates, a catalog of Medicines it offers, and information about customers as well as the order they have placed. MedicineValueNetwork as a whole has a special information object called SellTxn,which represents information processing that is specific to the localized action Sell. It captures information items that are created and processed by localized action sell within MedicineValueNetwork. However, these

pieces of information, which may themselves be represented as component information objects of SellTxn, are not visible in this view because SellTxn is seen as a whole.

Similarly, CustomerValueNetwork is seen as a whole in this view. Intuitively, CustomerValueNetwork could be regarded as a group of people who share interests in purchasing medicines. As such, it exhibits a localized action called Buy and a number of information objects that represent the cash it possesses, the medicine it owns, information about the order placed and a special information object called BuyTxn that represents t Buy-specific information processing. Some information objects of MedicineValueNetwork and CustomerValueNetwork are marked with stereotypes < < in > > or < < out > > indicating they are information objects exchanged between the business working object in which they are defined and the environment.

### c. Second Level of Organizational Enterprise Model Object

In this level of specification, both *MedicineValueNetwork* and *CustomerValueNetwork* are seen as composites. *MedicineValueNetwork* has four component business working objects representing a total of 3 cooperating companies within this businessvalue network: MedicineCo (responsible for processing orders and inventory management), ShipCo (dealing with warehouse and shipping operations), PubCo (supplying Medicines) and *CreditCardAgent* (processing online payments). We also see a counterpart of the company level for *CustomerValueNetwork*. It has three human working objects patient, Businessperson and Doctors. Within *MedicineValueNetwork*, business collaborations, localized actions and information objects represent the information processing of and information exchanged between the companies.

## IV. COMPUTATIONAL VIEWPOINT AND DIAGRAMMATIC REPRESENTATIONS FOR EA

### a. Representations for Model Object and Software Architecture

We recommend the use existing ODP standards and recommendations to complement EA in making ODP computational viewpoints of an enterprise model. Fig. 3 gives a computational specification of the IT system that provides IT capabilities of business working object PurchaseDept. This CV specification is represented using the UML as recommended [20]. Computational objects AdminTouchpoint and AssistantTouchpoint stand for the user interface of the IT system. Computational objects MaintenanceModule and ProcessingModule deal with system maintenance and order/inventory management, respectively. AdminTouchpoint interacts with MaintenanceModule via an interface named ISystemAccess (likewise, we have interface IProcessingAccess). In this view, we also see component (computational) objects of MaintenanceModule and ProcessingModule. They manage data related to orders, customers, inventories and system troubleshooting. Some other computational objects (de-)serialize the data. Note that this representation could be considered a view showing an extended organizational level – the level of software architecture.

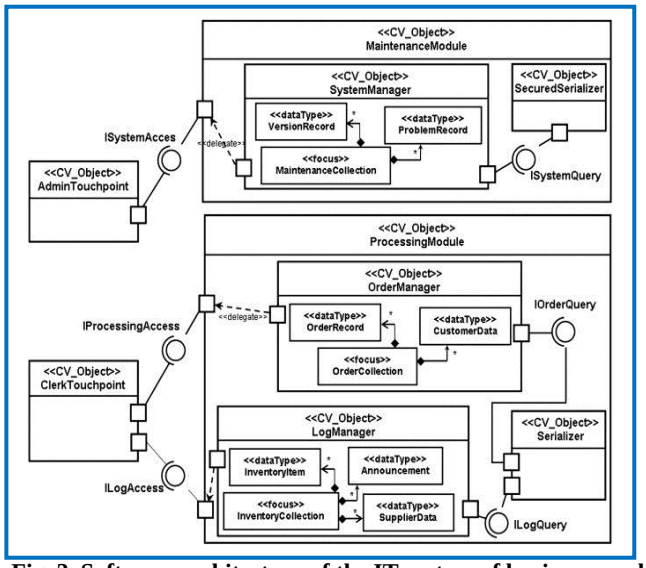


**Fig. 3. Software architecture of the IT system of business working objects PurchasingDept**

In this figure, we also see component (computational) objects of MaintenanceModule and ProcessingModule. They manage data related to orders, customers, inventories and system troubleshooting.

Now, after having given diagrammatic representations of business working objects and their collaboration, we proceed in precisely capturing the semantics of information processing for localized actions Sell and Buy as well as business collaboration sale that is regarded as dynamic schema in RM-ODP. We opt to use BPEL [24] for writing the spec of localized actions and business collaborations rather than Object Constraint Language (OCL) [9, 10] for the following two reasons.

## V. ANALYSIS APPROACH USING BPEL

Business Process Execution Language (BPEL) is a XML-based language for specifying services is used to define enterprise business processes within Web services. Every company has its unique way of defining its business process flow. The key objective of BPEL is to standardize the format of business process flow definition so companies can work together seamlessly using Web services [25, 28]. BPEL extends the Web services interaction model and enables it to support business transactions. BPEL is based on Web services in the sense that each of the business process involved is assumed to be implemented as a Web service. Processes written in BPEL can orchestrate interactions between Web services using XML documents in a standardized manner. These processes can be executed on any platform or product that complies with the BPEL specifications [23].

RM-ODP and EA are different architectures covering similar problem domain. There is an issue of interoperability or reuse of

models/specifications between the two. For instance, a Business Reference Model based business models will not be easily incorporated into ODP enterprise specifications, since their concerns and concepts are similar but not exactly the same.

### a. Execution of the Behavior Processes of Business Working Object

BPEL [section 2] is an XML representation of an executable process which can be deployed on any process motor. The atomic element of a process BPEL is an "activity", which can be the send of a message, the reception of a message, the call of an operation (sending of a message, makes an attempt of an answer), or a transformation of data.

A composite service in BPEL [23] is described in terms of a process. Each element in the process is called an activity. BPEL provides two kinds of activities: primitive activities and structured activities. Primitive activities perform simple operations such as receive (waiting for a message from an external partner), reply (reply a message to a partner), annonce (annonce ITsystem) maintain (maintain system "transaction", current version, Troubleshooting Record, Update, Backup), invoke (invoke a partner), assign (copying a value from one place to another), throw (generating a fault), terminate (stopping the entire process instance), wait (wait for a certain time) and empty (do nothing) [24].

Use case example: An Assistant is performing daily check of customer's orders for abnormality or doubtful payments. From time to time, he puts announcements on the web. An Admin is doing maintenance routines (i.e. troubleshooting, database backup, software update).

BPEL is an XML representation of an executable process which can be deployed on any process motor. The atomic element of a process BPEL is an "activity", which can be the send of a message, the reception of a message, the call of an operation (sending of a message, makes an attempt of an answer, troubleshooting, database backup, software update), or a transformation of data.

*< cv_behavior >*
*< working object Role />* → *definition of the working object role*
*<containers/>* → *definition of the containers of the data*
*<transitioncondition>*
*<constraints />* → *A set of rules related to a behaviour.*
*</transitioncondition>*
*<sequence/>*
*<receive />* → *reception of a request of process*
*<assign />* → *transformation of the data*
*<invocation />* → *call of an process*
*< maintain />* → *maintain system*
*<termination />* → *termination of process*
*<announcement />* → *interaction initiated by a client object*
*<reply />* → *sending of an answer to the process*
*</sequence>*
*</cv_behavior >*
*<process >*
*< roles />* → *definition of the roles*
*<containers/>* → *definition of the containers of the data*
*<sequence />*
*<receive />* → *reception of a request*
*<assign />* → *transformation of the data*
*<invoke />* → *call of an action*
*<reply />* → *sending of an answer*
*</sequence></process>*
*<messages> name = "namemessage"*
*<process name ="process"/>*
*< causality name = "causality"/>*
*<choice >*
*<message type =" Record "/>*
*< message type =" Update "/>*
*< message type =" Backup "/>*
*<message type ="invocations"/>*
*< message type ="terminations"/>*
*< message type ="annoucements"/>*
*</choice ></messages>*

**Fig. 3: Code Fragment that declares all Business and Parameter for Working Objects**

## VI. CONCLUSION AND FUTURE WORKS

This article has introduced a UML profile for automated hierarchical diagrams behavior processes with a UML to BPEL translator. The profile allows developers to use normal UML skills and tools to develop behavior Processes of business working object using BPEL. We leveraged the ODP modeling concepts to define a new approach for modeling an EA hierarchically. we centered around how to make RM-ODP, particularly the ODP languages of enterprise viewpoint [26] and information viewpoint [27], applicable in the context of multi-level EA modeling while adopting other ODP viewpoint languages computational viewpoint language to capture technical aspects in parallel with engineering viewpoint language for design of software and system architecture. And using of BPEL as an alternative to UML-OCL in capturing ODP dynamic schema.

Although we have only shown our method for the business functions behavior from the Enterprise Viewpoint, the method is generic enough to be applied in other viewpoints, such as business functions trader from the Information Viewpoint.This initial work brings many new and interesting questions in the general area of service development and management.

## REFERENCES

[1] ISO/IEC, ''Basic RM-ODP-Part1: Overview and Guide to Use, ''ISO/IEC CD 10746-1, 1994

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

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

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

[5] M. Bouhdadi, et al. ‘’ An UML-based Meta-language for the QoS-aware Enterprise Specification of Open Distributed Systems, ‘’ Collaborative Business Ecosystems & Virtual Enterprises, IFIP Series, Vol. 85, Springer Boston, pp.255-264, 2002.

[6] ISO/IEC, ‘’ODP Type Repository Function, ‘’ ISO/IEC JTC1/SC7 N2057, 1999.

[7] ISO/IEC, The ODP Trading Function, ISO/IEC JTC1/SC21 1995.

[8] ISO/IEC, ‘’ Use of UML for ODP system specifications’’ ISO/IEC 19793, May 2006.

[9] J. Rumbaugh, G. Booch, J. E. Jacobson, the Unified Modeling Language, Addison Wesley, 1999.

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

[11] J.Laassiri , Y.Balouki,H.Belhaj, S.El Hajji, M.Bouhdadi, "A Denotational Semantics of Concepts in ODP Information Language", Journal:Lecture Notes in Engineering and Computer Science ,2009

[12] 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, Kluwer Academic Publishers, Norwell, MA, September 1999. Chapter 9.

[13] E. Evans, R. France, K. lano, B. Rumpe, ‘’Meta-Modeling Semantics of UML, ‘’ In H. Kilov, B. Rumpe, and I. Simmonds, editors, Behavioral specifications for businesses and systems, Kluwer Academic Publishers, Norwell, MA, September 1999. chapter 4

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

[15] G. Myers, ‘’The art of Software Testing, ‘’, John Wiley &Sons, 1979

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

[17] J.Laassiri and al., “Specifying Behavioral Concepts by engineering language of RM-ODP", May 2010.

[18] Rumpe, ‘’ Agile Modeling with UML, ‘’ LNCS vol. 2941, Springer, 2004, pp. 297-309.

[19] K. Beck. Column on Test-First Approach. IEEE Software, vol. 18, no. 5, pp.87-89, 2001

[20] L. Briand , ‘’A UML-based Approach to System testing, ‘’ LNCS vol. 2185. Springer, 2001, pp. 194-208,

[21] B. Rumpe, ‘’ Model-Based Testing of Object-Oriented Systems; ‘’ LNCS vol.. 2852, Springer; 2003; pp. 380-402.

[22] B. Rumpe, Executable Modeling UML. A Vision or a Nightmare?, In: Issues and Trends of Information technology management in Contemporary Associations, Seattle, Idea Group, London, pp. 697-701.

[23] Duhang Zhong et al.”Reliability Prediction for BPEL-Based Composite Web Service” ,Proceedings of the first international conference On Research challenges in Information science, pages 265,270. Ouarzazate,Morocco,2007

[24] Dimitris Karagiannis et al. Business-oriented IT management developing e-business applications with E-BPMS,’’ ICEC 2007, 97-100

[25] keith Mantell,” From UML to BPEL Model Driven Architecture in a Web services world” ,Report IT Architect, IBM 2003.

[26] ISO/IEC,Information Technology - Open Distributed Processing - Use of UML for ODP system specifications | ITU-T Recommendation X.906 | ISO/IEC 19793, International standard, SC 7/WG19 and ITU-T (2009).

[27] Z. Milosevic, R.-G. Dromey, On Expressing and Monitoring Behaviour in Contracts, Proceedings of the 6th International Conference on Enterprise Distributed Object Computing (2002), pp. 3–14 Lausanne, Switzerland

[28] Business Process Execution Language for Web Services (version1.1),http://www.128.ibm.com/developerworks/library/specification/ws-bpel/, July 2002.