An Architecture for the interoperability of Workflow Models

Hamri Salah
Laboratory LIRE
Computer Science Department
Mentouri University of Constantine
Algeria, 25000
Tel /Fax: 213 31 63 90 10,
hamris@yahoo.fr

Boufaida Mahmoud
Laboratory LIRE
Computer Science Department
Mentouri University of Constantine
Algeria, 25000
Tel /Fax: 213 31 63 90 10
boufaida_mahmoud@yahoo.fr

Boudjlida Nacer
University of Nancy 1, LORIA
UHP, France
Tel: +33 3 83 59 30 76
Fax: + 33 3 83 27 83 19
Nacer.Boudjlida@loria.fr

ABSTRACT
The goal of this work is to contribute to the field of interoperability of Workflow models. To achieve this interoperability, we have built a generic architecture that addresses three levels of abstraction: the common meta-model that the Workflow models must share, the common model that they enact collectively, and the common data model whose management is shared. So, the approach we have adopted is based on a strategy of uniformity to solve the problems related to the semantic, syntactic and execution platform heterogeneity. The common meta-model gathers the common concepts that are shared between all these Workflow models and it defines their semantics. These concepts (activity, event, etc.) are extracted from different formalisms used in the field of business process (or Workflow). This latter allows us to instantiate a canonical model that describes only common parts (activities, artifacts). As for the common data model, it is instantiated by the common model. For the control interoperability, we have adopted an approach that deals with a more flexible connection mechanism based on events through a connection server. This approach can be implemented above any interoperability platform (CORBA, EJB, etc.). Also, the shared canonical model that we have proposed is generic, simple and re-usable.

Keywords
Interoperability, Workflow, semantic, common meta-model, canonical model, process, event.

1. INTRODUCTION
Nowadays, organizations (software companies, banks, office automation, hearth care services, etc.) rely on a wide variety of systems or applications which are generally autonomous, heterogeneous, and distributed to conduct their business process (or Workflows). Interoperability between such systems or applications is indispensable because sites are needed in their resources and therefore in the activities they can perform. It becomes vital for these organizations to develop their Workflow models based on a strategy of integration, i.e. with facilities to share data, electronic forms, etc., and interoperability, i.e. in addition to their integration, their capacity to cooperate in the execution of a process or an activity [2]. This need leads us to study the problems of integration and interoperability in the Workflow domain. Most explorations of interoperability have been in different domains: databases [5], software engineering [1], [2], Workflow systems [25], [8], [15], Web services composition/orchestration [10], [18], [17], [4] in the past few years. Although many research works have treated the interoperability of business process (Workflow models), there is no semantics for interoperability at higher levels of abstraction. Indeed, these works provide generally a canonical model, which is insufficient such as XPDL (XML - Process Language Definition) [24], WSFL (Web Services Flow Language) of IBM [6], XLANG of Microsoft [16], BPML (Business Process Modeling Language) of BPMI [10], [7], etc. However, there is a need of semantic interoperability, which ensures that "the requester and the provider have a common understanding of the 'meaning' of the requested services and data". Therefore, we have decided to contribute to this part and to investigate the concepts and mechanisms that enable the semantic, syntactic and control interoperability.

The remainder of this paper is organized as follows:

In section 2, we describe the different techniques of interoperability that exist and that we will encounter in a heterogeneous environment. Section 3 makes a brief comment on related works concerning interoperability in our domain. In section 4, we present an overview of general interoperability architecture. The fifth section gives the infrastructure that supports the interoperability of our Workflow models by defining the communication model and shows the feasibility of our approach by using for example a CORBA platform [9]. In Section 6, we present a concrete example in business process and in order to clarify ideas, we demonstrate our approach for interoperability by illustrating an execution scenario that concerns the purchase order process. Section 7 concludes the paper with a summary and the future work.

CIKM'05, October 31-November 4, 2005, Bremen, Germany.

2. TECHNIQUES OF INTEROPERABILITY

In this section, we outline the different levels of interoperability that exist and should be solved in order to favor the interoperability between different Workflow models. These levels are related to the used concepts, to the different formalisms of modeling, and to the several execution platforms [1], [13]. However, one can distinguish three techniques of interoperability:

- Semantic interoperability: it concerns the diversity of the used concepts. Each Workflow model can offer different concepts to manage the aspect of the process to which it is dedicated.
- Syntactic interoperability: it is related to the diversity of the formalisms that exist to describe the aspects of Workflow. The choice of formalism as compared to an other can have a considerable impact on the comprehension of the model.
- Control interoperability: it concerns the execution of Workflow that is distributed on several autonomous and different material platforms.

In the following section, we review some related works concerning the interoperability of Workflow models. The objective, here, is to inspire about these works for describing our approach.

3. RELATED WORK

Business process has received a lot of attention during the last decade. There has been extensive research in the Workflow field. These have been conducted separately with different researchers leading to multiple models and environment prototypes (WFMS i.e. Workflow Management Systems). The WfMC (Workflow Management Coalition) [20] that is an organization of standardization in the Workflow domain has defined five functional interfaces to a Workflow service as part of its standardization. 'The interface one' supports Process Definition Import and Export. It includes a common model (canonical model) for the process definition. Also, the WfMC has established a glossary [21] which contains a description and a common terminology for the basic concepts embodied within a process definition such as activity, transitions, etc. This canonical model [24] called XPDL, uses XML as a mechanism for process interchange. As for the 'interface four', the coalition has defined WAPI4 (Workflow Programming Interface/Interchange4) for supporting Workflow interoperability. In this context, WfMC has proposed [23], [25] the Wf-XML as an interoperability protocol based on XML. In parallel, the OMG (Object Management Group) has defined a Workflow interoperability protocol named jFlow [15]. The standard CORBA allows distributed objects to participate in the Workflow by means of an ORB (Object Request Broker). Other research group (HP, Netscape, and SUN) has proposed SWAP (Simple Workflow Access Protocol) as a solution to support the interoperability between Workflow execution servers. It uses the http protocol and XML for data interchange [8].

Therefore, XPDL has not an execution semantic and this latter is in charge of Workflow product. As for the OMG and the research group (HP, Netscape, SUN), they do not define the semantics aspects of the entities that are manipulated in the integration or in the interoperability.

There are other approaches of interoperability with same objectives, such as Web services composition / orchestration that use different languages like XLANG from Microsoft [16], WSFL (Web Services Flow Language) from IBM [6] WSCI (Web Services Choreography Interface) from SUN [19], BPEL4WS from IBM, Microsoft and BEA [17], [4], [18], [7]. These approaches are oriented toward web services and are usually called WSOA (Web Services Oriented Approach) [18].

Although these approaches have resolved the problem of interoperability, there is no semantics for the entities or concepts that are manipulated during the cooperation of business process, and no common standard has been agreed upon for these languages for web services composition. However, the approach that we adopt supports at the same time semantic, syntactic and control interoperability. Furthermore, it takes into account criteria of flexibility (opening for other Workflows to easily join the interoperability environment), distribution (among different sites) and autonomy of Workflow models (at the design and the execution level).

4. GENERAL INTEROPERABILITY ARCHITECTURE: AN OVERVIEW

The architecture that we propose as depicted in figure 1 forms an interconnecting network in which each node represents a Workflow model. Each model on the network is autonomous with its own meta-model, its own modeling language, and its own process engine. So, it has an interface and can be integrated with others by being registered on a common model and ontology. To interoperate, they must agree on the semantics of concepts that enable them to exchange with a common understanding. Therefore, we must investigate the shared concepts and mechanisms that enable the semantic interoperability. We call this set of concepts and their semantics the common meta-model.

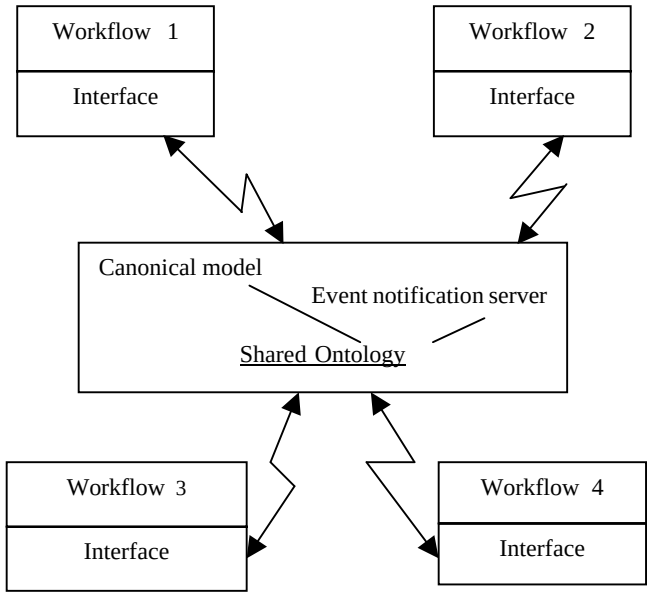

Figure 1. General architecture.

- ***The common meta-model***: This latter constitutes then, a common ontology that makes models to understand each other. The choice of these concepts is based on the different formalisms that are used in the Workflow field or business

process such as: XPDL [24], XLANG [16], WSFL [6], BPML [10], etc. We have compared the proposed concepts and have aligned them up according to their objectives as defined by their designers. Then, we have extracted a core of basic concepts that is common to the set of Workflow models. These concepts are generic such as: activity, event, resource, etc. The figure 2 shows the UML class diagram of our common meta-model.

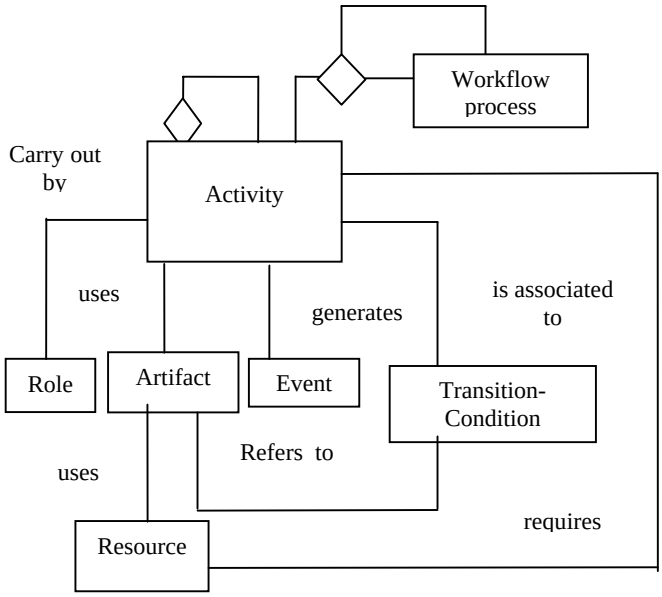

Figure 2. Common meta-model.

- ***The common model:*** this model is canonical and it is inspired by [1], [13]. It represents the objective of the integration. It contains only that parts that are common to the different Workflow models (activities, artifacts) and it neglects the aspects such as the resources and their assignment, the communication model between the actors of Workflows, etc. All these "contextual" aspects are uncoupled from the common model. This way allows to our model to be simple, generic, reusable in different contexts. For example, the same model might be used with various strategies of data sharing, different policies for allocating resources, different modes of communications between actors, etc.

The canonical model is obtained by instantiation from the common meta-model according to the principle of reification used in meta-types systems or in object-oriented languages [3]. The goal is to consider a class as an object making it possible to manipulate it as its instances (The modification of a class is made like the modification of the object's value). The approach of reification (figure 3) then consists in introducing a supplementary level above the level of the types. This level is called "meta" and it contains a set of special types called meta-types. The instances of a meta-type are types that can to be instantiated.

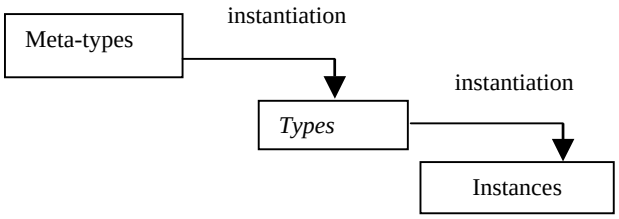

Figure 3. The level meta as a support of the reification.

For data integration, one single data model is used. In this case, all the Workflow models share a common data information description.

- ***The common data model:*** It is a common database obtained by instantiating the common model. Its construction is done from the reified canonical model according to the principle of reification quoted above. An other approach to instantiate the common data base is to apply the 'clone' operation that used in languages like java. For example, in java, the clone (instance) has the same structure, the same methods and the same value as the cloned object.

5. INFRASTRUCTURE FOR INTEROPERABILITY

We provide the necessary infrastructure that enables different Workflow models to communicate and to share data.

5.1 Event Based Communication Model

For the control interoperability, we adopt an approach that deals with a flexible connection mechanism based on events. The choice of event paradigm as a mechanism of interaction is probably more suitable for interoperability leading the different Workflows to interact by sending out notification about certain events, other Workflows may subscribe for notifications about events. Indeed, in the Workflow field, the concept of event is fundamental since the dynamic of a process is represented by the events, which make it possible to start the activities [12]. Therefore, event-based communication, in which Workflow interact with each other by raising and answering to events occurrences, is the most prevalent control integration. We adopt a control interoperability based on broadcast events and supported by a connection server. It receives the registration of Workflow models. These latter send their messages without specifying any receiver. The connection server then captures the events. Workflow models subscribe to a set of events, they define filters that specify what kind of events they want to receive. In response to an event, the receiver can take one or more several actions. Each action corresponds to an execution method.

To achieve the communication between the Workflow models, we rely on the pattern of communication that is described below.

```
Communication pattern <ident> :
    Sender (SenderId)  < == >  Receiver (ReceiverId)
    {  On SenderId.event  == > (parameters)
        ReceiverId.event  == > (parameters) } *
End  < ident>;
```

As mentioned above, our approach is based on events, therefore, we need tools for the management of these events such as a local event manager and a global event manager. The scenario of communication between Workflow models, is then as follows:

The Workflow engine executes its corresponding model and can notify the local event manager of a certain subset of events and parameters. These events are specified based on a common representation (canonical model) that is common to different Workflow models.
The local event manager notifies events to the connection server.
The connection server, referring to a strategy, assigns events to different Workflow models. The assignment is based on the specification defined in the connection server.
Once the event item is assigned to the most appropriate Workflow model, the local event manager propagates it to the Workflow engine for execution.

5.2 Example of an Interoperability Platform

In order to validate our approach, we used CORBA as a platform of interoperability [6] that presents advantages such as : abstraction, opening, flexibility, and programming languages independence. As depicted in figure 4, the Workflow models can interact thanks to the functionalities offered by an Object Request Broker (ORB services) [11], [9]. This latter provides the mechanisms enabling to send and receive requests and answers in a transparent way. In this context, we have used two ORB: Orbix that offers a binding to C++ for the implantation of the connection, the common model, etc., and OrbixWeb that offers a binding to java (in which the Workflow models can be implanted).

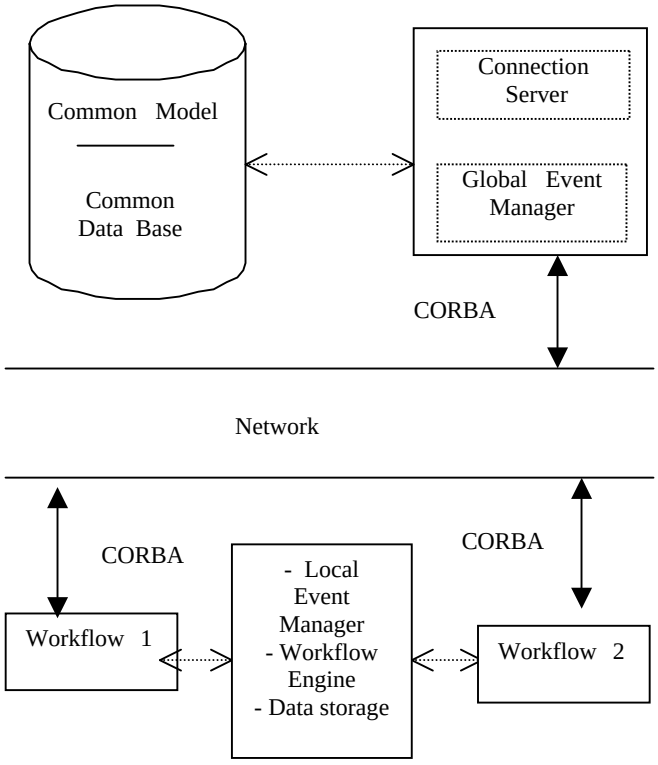

Figure 4. Example of CORBA based architecture.

CORBA manages requests between clients and servers. In our architecture, it is used by the Workflow models (the clients) to invoke the services of the global event manager and the connection server, which is implemented as a CORBA server.

In our case, CORBA intervenes in two ways:

1. The abstracted description through the IDL language of the common model and the common data model. This description is essential for our heterogeneous framework (syntactic interoperability).
2. The management of the interactions between the Workflow models (objects CORBA : clients) and the CORBA server (connection server, global event manager) through the ORB (Object Broker Request).

Therefore, the IDL description of the Workflow models (their interfaces) is presented in terms of attributes and methods.
The figure 5 shows an example of the IDL structure of these interfaces.

```
Interface Meta {
   String typeName;
   Sequence <string> supertypes;
   String instanceOf
      };
Interface Event: Meta {
   String nature;
   String entitytype;
   String methodname;
   String <param> parameters;
     };
Interface Activity: Meta {
    String activityId ;
    String description ;
   Boolean resourcesRequired;
   String  responsible;
    State state ;
    Sequence <activity> subActivities;
    Desktop desktop;.
    Sequence <Rule> rules;
              …… ……. };
 Interface Rule: Meta {
   Event event;
   Condition conditionTransition;
   sequence <Actions> actions;
      };
Interface ConditionTransition: Meta {
   String conditionTransition;    }.
Interface Action :Meta  {
   String  methodname;
   String  entityType;
   String <param> parameters;
   };
```

Figure 5. IDL interfaces of common meta-model entities.

6. A MOTIVATING EXAMPLE

In this section, we illustrate our approach giving a motivating example that is used in business process. It concerns the purchase order process.
A model commonly used in business process is derived from a generic negotiation protocol between a customer who sends a product purchase order to a company, A which sub-contracts with a company B.
The scenario of this product purchase order process is subject to the following steps:

- A client requests a product from the company A via his purchase order or via a telephone call.
- The company A checks its stock if it is sufficient. Otherwise, the company A creates a document (proposal order) and sends the proposal to the company B that is a sub-contractor.
- The company B verifies its stock (requested quantities) and asks for information and clarification about the customer's solvency, price reduction, etc.
- The company A receives the information, completes the clarificationand then sends the order proposal.
- The company B receives again the proposal, accepts the order, and then sends the approval reply.
- The company A receives the approval reply and waits for the invoice.

Such an example provides a good framework for using our approach of interoperability. The figure 6 shows the protocol. That is defined.

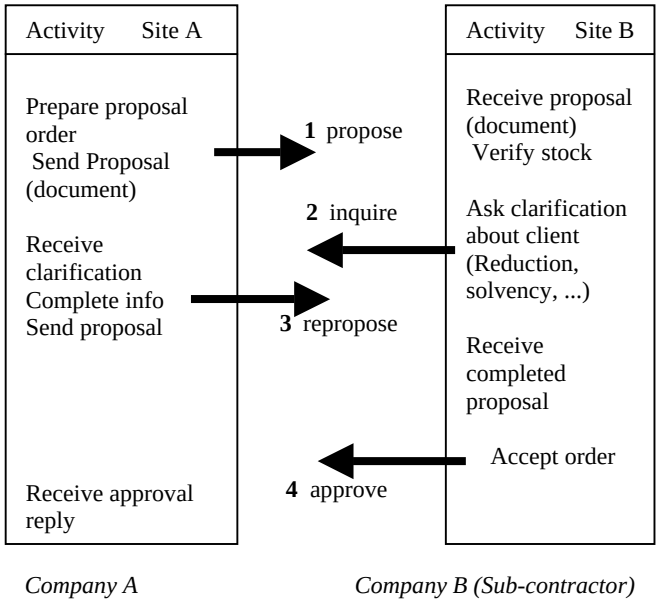

Figure 6. Simplified coordinating activities.

Therefore according to this protocol, the workflow models can directly communicate with other models by exchanging events i.e. messages.

6.1 The Communication Model

Referring to the previous example, we will achieve the communication according to the pattern of communication that was depicted in section 5.1.

The communication specification between the company A's sales manager (SLMA) and the company B's stock manager (SKMB) is as follows :

Communication Purchase_Order_Process :
SalesManager (SLMA) < == > StockManager (SKMB)
P: Proposalorder;
PO: PurchaseOrder;
On SLMA.Sendproposal (P, SKMB) == > **SKMB .Verify** (P) ;
On SKMB.Inquire (P, SLMA) == > **SLMA .Complete** (P) ;
On SLMA.Sendpurchaseorder (PO, SKMB) == >
SKMB .Approve(PO) ;
On SKMB. Accept (PO, SLMA) == >
SLMA.Approvalreply (PO) ;
End Purchase_Order_Process.

6.2 Execution Scenario of the Purchase Order Process

This section illustrates the execution scenario of a purchase order process. Let us assume we have two Companies A and B (B a Sub-contractor). We want these companies to cooperate in the execution of the purchase order process. The scenario starts when a customer sends a purchase order to the company A and this latter checks its stock and notices that the available quantity is not sufficient concerning one product. Therefore, it requests the company B for delivering the missing product.

To simplify, we present the Workflow model as composed of four activities : Verify stock, Validate order, Invoice, Deliver order.

The Company B executes the activity 'Verify stock' and the Company A executes the activity 'Validate order' by adding information about the customer (such as: is the client solvent or not, what percentage of reduction has to be applied, etc.) in order to validate the purchase order. The figure 7 shows only a limited part of this scenario.

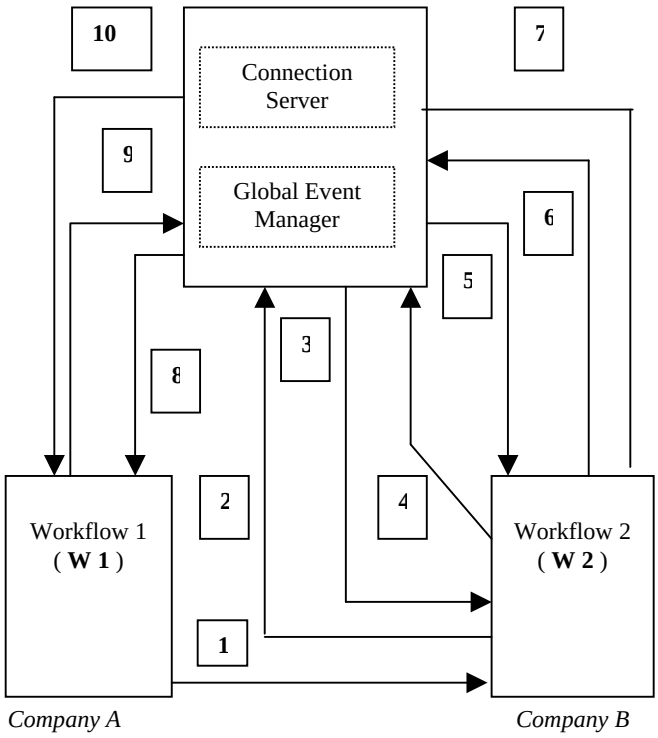

Figure 7. Execution scenario of purchase order process.

// Example of IDL interface of the Workflow1 (communication based on Corba)
interfaceW1For Workflow1 {
logon (in string Workflow1_1d);
logoff (in string Workflow1_Id);

Void sendRequestTOW1(in any request);
……}

The cooperating operations are depicted below :

(1) getRequestToW2 (in any request) ; // to request to W2 for delivering order.

(2) startProcess (Scenario_PO) ; // to instantiate the process (of Purchase Order).

(3) W2.getEevent (Creation_Verifystock1) ; // to notify the creation of the first activity that shall be performed by the Workflow 2 (W2).

(4) changeState (Verifystock1, Init, Active) ; // to change the state of this activity (Verifystock1 is an instance of the activity 'Verifystock'.

(5) W2.getEevent (Activation_Verifystock1) ; // to start this activity by notifying the Workflow 2 (W2) .

(6) changeState (Verifystock1, Active, Terminated); // the activity is terminated

(7) W2.getEevent (Fin_Verifystock1) ; // the execution end of the activity.

(8) W1.getEevent (Creation_ValidateOrder1) ; // to notify the creation of the second activity that shall be performed by the Workflow 1 (W2).

(**9)** changestate (Validateorder1, init, Active) ; // to change the state of this activity.

(10) W1.getEvent (Activation-Validateorder1) ; // to start the second activity.

……..……..etc.

The process is instantiated by (1) and creates four activities (instances) : Verifystock1, Validate_order1, Invoice1, Deliver_order1. Then it generates a set of events : Creation_Verifystock1,Activation_Verifystock1,Fin_Verifystok1, etc. These events are notified by the local event manager or by the global event manager.

7. CONCLUSION AND FUTURE WORK

In this paper, we have presented an approach for the interoperability of workflows model based on events above an interoperability platform such as CORBA. The paradigm of event is very adapted for the interaction of workflows. To achieve this interoperability, we have oriented our approach according to a strategy of uniformity allowing us to solve the problems that concern heterogeneity semantic, syntactic and execution platform, by presenting a generic architecture for interoperability of workflow models. This architecture addressed three levels of abstraction: the common meta-model that interoperating workflow models must share (by using the same semantics for the shared concepts), the common model that they enact collectively, and the common data base whose management is shared. Also, we have applied the principle of reification used in object-oriented languages to our domain. Furthermore, we used CORBA platforms for the experimentation. This choice is motivated by many advantages (flexibility, opening, etc.). Indeed, CORBA allows us to increase the degree of opening of our interoperability by enabling a broad category of workflow models to manipulate the common model.

In parallel, we are working on extending the interoperability architecture. As further work, we plan to improve our architecture by:

- Using the MDA (Model Driven Architecture) as an alternative approach for building a common meta-model that shall be based on the MOF (Meta-Object Facility). This latter is a standard formalism of a higher level that defines the structures and the operations to create meta-models
- Orienting our architecture to Web services technology. In this context, we expose the server CORBA as a web service, and we create a gateway between the protocol CORBA/ IIOP and the protocol SOAP/ http. So, we apply a mapping form CORBA/IDL to WSDL.. This perspective is currently under implementation.
- In future research, we envisage to introduce semantic aspects for the Web services in our field Workflow (or business process) that called by researchers the semantic Web services.

8. REFERENCES

[1] Amiour Mahfoud : « Vers une fédération de composants interopérables pour les environnements centrés procédé logiciels », thèse de Doctorat, Grenoble I, France, 17 juin 1999.

[2] Boudjlida Nacer : « Intégration et Interopérabilité dans les Environnements logiciels » , Loria, Université Henri Poincaré Nancy I, Projet Architecture. [http://www.aee.inria.fr/cederom/bilan/docbilan/interop.pdf

[3] Bounaas.F Mchabre-Peccoud, P.Ycumin, J.P Cumin, P.Morat an D.Rieu : « Objet et méta-modélisation », 1997.

[4] DAVID Lévy : «Coordination des Web services : langages de description et plates-formes d'exécution », XRCE Grenoble, Mémoire septembre 2002.

[5] Djamel Benslimane, Kouyou Yetongnon, : « Une architecture multi-agents pour l'interopérabilité des bases de données hétérogènes », CARI'98 Dakar, October 1998.

[6] Frank Leyman, Web Services Flow Language», IBM Software Group specification, Mai 2001.

[7] Gilles Bretin, Peter Dahne : « ISNET40 : la vague EAI , la gestion des processus, Décembre 2003. version1.1. ».

[8] Gregory Alan Bolcer and Gail Kaiser : « SWAP : Leveraging the Web to manage Workflow, IEEE Internet Computing, Janvier, Février 1999.

[9] http://www. omg.org.

[10] Jean-Marie Chauvet, : « Web Services avec SOAP,WSDL, UDDI, ebXML…», Editions Eyrolles 2002.

[11] Jean-Marie Geib, Christophe Gransart et Philippe Merle : « CORBA », des Concepts à la Pratique », http:// www.corbaweb.lifl.fr.

[12] Karim Saïkali : « Flexibilité des Workflows par l'Approche Objet : 2Flow, un Framework pour Worklows Flexibles », Thèse de Doctorat, Lyon, France, Sept 2000.

[13] Ma.Lizbeth Gallardo : « Une Approche à Base de Composants pour la Modélisation des Procédés Logiciels, DEA, Université Joseph Fourrier, Grenoble I, Septembre 2000.

[14] Nicolas Ploquin : « Etudes des différentes formes d'interopérabilité en Génie Logiciel « , DEA, Université de Nantes, France, Septembre 2001.

[15] Object Management Group, Workflow Facility Submission (jFlow) OMG Document Number bom/98-06-0, July 4, 1998

[16] Satish Thatte, « XLANG,Web Services for business process design », Microsoft, Mai 2001.

[17] Tanguy.C ; : « Business Process Management, De la modélisation à l'exécution, positionnement par rapport aux architectures orientées services, INTALLIO ».2003.

[18] Tran.P, « WSOA l'architecture orientée Service Web », Developer reference, 5 Septembre 2002, www.devreference.net.

[19] « Web Services Choreography Interface1.0», BEA systems INTALLIO, SAP, Sun Microsystems, June 2002.

[20] Workflow Management Coalition: The reference Model, WfMC Document Number TC-1003, issue 2.0, Jan 1995.

[21] Workflow Management Coalition, Terminology and Glossary, WfMC Document Number TC-1011, issue 2.0, June 1996.

[22] Workflow Management Coalition, Interface: Process Definition interchange, WfMC Document Number TC-1016, November 1998.

[23] Workflow Management Coalition, Interoperability Abstract Specification, WfMC Document Number TC-1012, new version, June 2000.

[24] Workflow Management Coalition, Workflow Process Definition Interface, XML Process Definition Language, Document Number TC-1025, May 22, 2001.

[25] Workflow Management Coalition, Proposal for an Asynchronous HTTP binding of Wf-XML,, June 1, 2000.