How to Design a Loose Inter-organizational Workflow? An Illustrative Case Study

Lotfi Bouzguenda

IRIT Laboratory, University Toulouse 1, Place Anatole France,
31042 Toulouse Cedex, France
lotfi.bouzguenda@univ-tlse1.fr

Abstract. This work deals with the design of Loose Inter-Organizational Workflow (IOW). Loose IOW refers to occasional cooperation, free of structural constraints, where the partners involved and their number are not pre defined. We show that the design of Loose IOW application is very complex due to three factors: (i) the heterogeneity and distribution of the component processes, the organizations and the information (ii) the autonomy of each partner, which must be preserved (iii) the need to integrate in a coherent framework the three dimensions of a workflow: process, information and organization. One possible way to deal with this complexity, and to ease loose IOW applications design, is to use a well known software engineering principle: *the separation of aspects*, which aims at decomposing a system in communicating sub systems, each one coping with a relevant abstraction that requires a model to be structured and described. Following this practice, a loose IOW application must be though as three communicating models: an informational model, an organizational model and a process model. The first two models are represented with UML class's diagram, while the last model is described with *Petri Nets with Objects* (PNO), which are a formal language, have a very adequate expressive power and make the glue between the three workflow dimensions. We illustrate our solution through the well-known "reviewing papers" case study.

1 Introduction

Inter-organizational Workflow Context. Inter-Organizational Workflow (IOW for short) is a current research problematic, which investigates the cooperation of several distributed, autonomous and heterogeneous business processes [1] [2]. We mean by cooperation the gathering of business processes and the sharing of resources (information, human and machine) between the component organizations in order to achieve a common global goal.

IOW can be studied in the context of two following distinctive scenarios: loose IOW and tight IOW [3]. In this work, we focus on loose IOW which refers to occasional cooperation between organizations, free of structural constraints, where the organizations involved and their number are not pre-defined but should be selected at run time in an opportunistic way.

K.V. Andersen, J. Debenham, and R. Wagner (Eds.): DEXA 2005, LNCS 3588, pp. 1 – 13, 2005.

The design of loose IOW application is very complex. This complexity is mainly due to three factors:

- The heterogeneity and distribution of the component processes, the organizations and the information since IOW supports the cooperation between business processes running in different organizations. Naturally, these organizations do not share the same information and do not have necessary the same capacities. Regarding the heterogeneity of processes, the same service can be provided by two different organizations according to processes which differ by their quality, their duration or the number of stages they require.
- The autonomy of each partner must be preserved. First, each partner participating in an IOW should be able to decide by itself, the conditions of the cooperation i.e. when, how and with whom it cooperates. Second, each partner may prefer publish the interface of its process rather than its detail (implementation).
- The need to integrate in a coherent framework the three related dimensions of a workflow (process, information and organization). Indeed, a workflow process is made of a set of coordinated tasks, each one uses and produces information and is performed by an actor (human or machine) of the organization.

Most of the works concerning IOW [1][2] only focus on the process dimension by providing interaction models to support distributed execution of component processes. These works do not make really the glue between the three-workflow dimensions.

The problem being addressed in this paper is "*how to design a loose IOW application considering the three main dimensions (organization, information and processes) in a coherent framework*". One possible way to take into account these different dimensions and to deal with their complexity is to use a well known software engineering principle [4]: the *separation of aspects*, which aims at decomposing a system in communicating sub systems, each one coping with a relevant abstraction that requires a model to be structured and described. Following this practice, a loose IOW application must be though as three communicating models: an informational model, an organizational model and a process model. They are described below.

- The *informational model (IM)* describes the forms, documents, data that are used and produced by a workflow.
- The *organizational model (OM)* has two objectives. First, it structures actors in classes sharing the same features. A class is called *role* when it comprises actors having the same capabilities, and an *organizational unit* for actors belonging to a some organization structure. Second the organizational model attributes to each actor authorization to perform tasks. Roles and organizational Units are abstraction that can be used to define business processes without referring explicitly to the individual actors in a workflow, but rather to the capacity they must have.

The *process model (PM)* defines the component tasks, their coordination, and the information and actors involved in each task. This model refers to both the organizational model, which defines and organizes the set of potential actors, and the informational model, which allows access to the objects to be processed. To describe a

process model we need a Process Description Language (PDL). Unfortunately, some PDLs define tasks at a low level detail, as the process to be executed, and do not provide abstractions to design and simulate the model. Conversely, other languages define tasks at a very high level, as a goal to be reached, and do not provide an operational semantics. The ideal language would be one with a very large expressive power to describe the three models (information, organization and process) in a uniform way, to provide an operational semantics and to define tasks at a very high level.

Our solution is based on the following principles:

- *The separation of aspects*, which introduces an original way to decompose a system in communicating sub systems, thus offering new reuse opportunities and easing software maintenance and evolution.
- *The use of PNO*, which is an appropriate PDL to formally describe processes referencing the organizational and informational models.

We illustrate our solution through the well-known "reviewing papers" case study (see table 1).

Table 1. The reviewing papers case study

We consider a distributed version of the well-known "reviewing papers" case study. The chairman receives papers from authors and then registers, codifies and classifies them by topics. According to this classification, he elaborates one or several call for reviewers in a public and well known electronic space. After receiving bids from potential reviewers, he selects some of them to constitute his Program Committee (PC). Then, he distributes the papers to be evaluated and the review form to the PC members. After receiving all the review reports from the PC members, the chairman synthesizes these reports and elaborates two lists, one for the accepted papers and the other one for the rejected papers and finally, he informs each author. This case study is inspired from the ACM Special Track on Coordination [5], and can be seen as a loose IOW since its actors (authors, chairman and reviewers) are distributed in different organizations (laboratories, enterprises or universities) and as we have described above, reviewers are recruited dynamically by the chairman. Moreover, each reviewer may have its own reviewing process. For example, some of them could delegate the reviewing to colleagues, while others will review all the papers by them self.

Organization of the paper. The remainder of this paper is organized as follows. Section 2 briefly introduces PNO formalism as an appropriate language for modeling processes and justifies why we use this formalism. Section 3 models the case study through three communicating models (OM, IM and PM). The IM is based on ontology to solve information heterogeneity. The OM is based on an original component "a Matchmaker" in charge of connecting dynamically partners. The PM is based on PNO formalism, which enable the description and the coordination of the component processes while referencing the two previous models. Section 4 briefly discuses the related works and concludes the paper.

2 Petri Nets with Objects

2.1 What Are Petri Nets with Objects?

Petri Nets with Objects (PNO) [6] are a formalism combining coherently Petri nets (PN) technology and Object-Oriented (OO) approach. While PN are very suitable to express the dynamic behavior of a system, OO approach enables the modeling and the structuring of its active (actor) and passive (information) entities. In a conventional PN, tokens are atomic and indissociable, whereas they are objects in a PNO. As any PN, a PNO is made up of places, arcs and transitions, but in PNO, they are labeled with inscriptions referring to the handled objects. More precisely, a PNO features the following additive characteristics:

- *Places* are typed. The type of a place is a (list of) type of some object-oriented sequential languages. A token is a value matching the type of a place such as a (list of) constant (e.g. 2 or 'hello'), an instance of an object class, or a reference towards such an instance. The value of a place is a set of tokens it contains. At any moment, the state of the net, or its marking is defined by the distribution of tokens onto places. A transition is connected to places by oriented arcs as it aims at changing the net state, i.e. the location and value of tokens.
- *Arcs are labeled with parameters*. Each arc is labeled with a (list of) variable of the same type, as the place the arc is connected to. The variables on the arcs surrounding a transition serve as formal parameters of that transition and define the flow of tokens from input to output places. *Arcs* from places to a transition determine the *enabling* condition of the transition: a transition *may occur* (or *is enabled*) if there exists a *binding* of its input variables with tokens lying in its input places. The *occurrence* of an enabled transition changes the marking of its surrounding places: tokens bound to input variables are removed from input places, and tokens are put into output places according to variables labeling output arcs.
- *Each Transition is a complex structure made up of three components*: a precondition, an action and emission rules. A transition may be guarded by a *precondition*, i.e. a side-effect free Boolean expression involving input variables. In this case, the transition is enabled by a binding only if this binding evaluates the precondition to true. Preconditions allow for the fact that the enabling of a transition depends on the location of tokens and also on their value. Most transitions also include *an action*, which consists in a piece of code in which transition's variables may appear and object methods be invoked. This action is executed at each occurrence of the transition and it processes the values of tokens. Finally, a transition may include a set of *emission rules* i.e. side-effect free Boolean expressions that determine the output arcs that are actually activated after the execution of the action.

Figure 1 gives an example of a PNO describing a simple task registering of paper, given the paper, call for paper, an available author and the Chairman in charge of registering the paper. This PNO is composed of a transition, four input places and two output places. Each place is typed with one of the four following object classes: <*Call for paper*>, <*Paper*>, <*Author*> and <*Chairman* >. Each input place contains a token

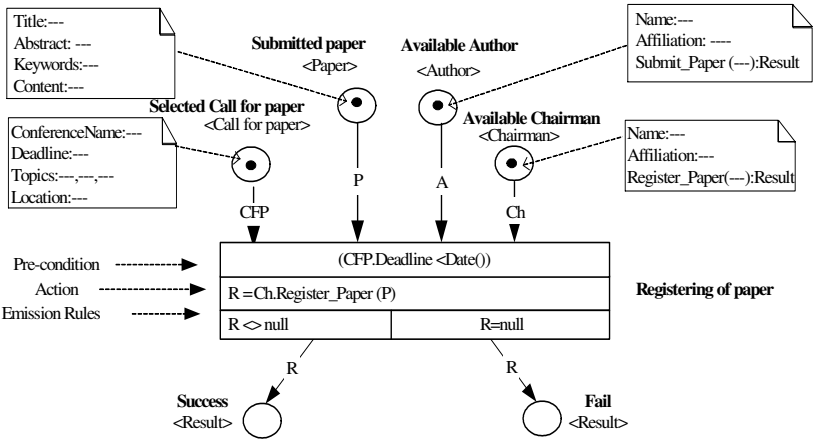

Fig. 1. Example of a PNO

whose value is indicated by a comment linked to it by an arrow. From left to right, the first two input places called *submitted paper* and *selected call for paper* contain one token corresponding respectively to a paper and a call for paper. The object class <*Paper*> has four attributes {Title, Abstract, Keywords, Content} and the object class <*Call for paper*> has four attributes {ConferenceName, Deadline, Topics, Location}. Let us also remark that the <Paper> and <Call for paper> object classes refer to informational model. The second two input places called *Available Chairman* and *Available Author* contain also one token corresponding respectively to a Chairman and an Author. The class object <*Chairman* > has two attributes {Name, Affiliation}, features a method {Register_Paper} and the <*Author*> object class, has two attributes {Name, Affiliation}, features a method {Submit_Paper}. Both object classes refer to organizational model. Now let us consider the transition registering of paper. It has a precondition *(CFP.Deadline<Date())*which indicates that the submission date must not exceed the actually date. If this precondition is satisfied, the action is executed and the Chairman is asked to execute the *Register_Paper* method. According to the result R, returned by this method, the emission rules will direct the process through one path or another. If the registering of paper is ok, the result R is not null and then a token is put in the *Success* output place. In the other case, a token is put in the *Fail* output place

2.2 Motivations for Using Petri Nets with Objects

Advantages of PN in Workflow Context. Petri Nets are widely used for workflow modeling [7]. Several good reasons justify their use:

- *An appropriate expressive power* that allows the clear and precise description of the different tasks involved in a process and their coordination. The main workflow control patterns [8] (e.g. sequence, parallel, split, join…) can be described by Petri Nets.
- *A graphical representation* that eases the process definition.

- *An operational semantics* enabling an easy mapping from specification to implementation.
- *Theoretical foundations* allowing analysis and verification of behavioral properties and performance evaluation. Numerous techniques with associated tools are available as varied as algebraic techniques, graph analysis and simulation.

Advantages of PNO in Workflow Context. Conventional Petri nets focus on the process definition and do not capture the organizational and the informational dimensions of a workflow. As we have mentioned it in the previous section, Petri nets with Objects extend classical Petri nets by integrating high-level data structure represented as objects and therefore provide the possibility to integrate in a coherent way the two dimensions missing in conventional Petri nets. Thus, using PNO, actors of the organizational model are directly represented as objects and they may be invoked through methods in the action part of a transition. In the same way, data and documents of the informational model are also represented by objects flowing in the PNO and transformed by transitions.

Advantages of PNO in IOW Context. PNO provides two mechanisms to support process interoperability. Interaction with other external processes can be modeled with additional (called connection) places. Input places can represent localizations where partners are asked to put typed information while output places represent localizations where typed information are made available for partners. This mechanism does not require to know the identity of the partner and the detail of their process. In this case, the autonomy of each partner is preserved. Regarding, the interaction with a priori known software components (matchmaker, …), it can be modeled by directly invoking them in the action part of a transition. The use of a matchmaker is very useful in the context of loose IOW since it helps to dynamically connect distributed partners.

3 Modelization of the Case Study

The purpose of this section is to present our solution of the well-known “reviewing papers” case study. Our solution is made of three communicating models, namely the informational, organizational and process models.

3.1 The Informational Model

As we have mentioned in section 1, the loose IOW context corresponds to a situation where the identities of partners and their processes are not known a priori and consequently the informational model can not be described fully. To solve this problem, we propose the use of an ontology. This ontology describes the common vocabulary (or main concepts) of the domain being considered, and partners (reviewers and chairman in our case) are supposed to adhere to this common ontology in order to cooperate. As shown by [9], an ontology can be support for solving data semantic interoperability between partners.

Our informational model describes the structure of two types of information: documents and data. The documents can be classified in two great classes: *Manuscript* and *Electronic*.

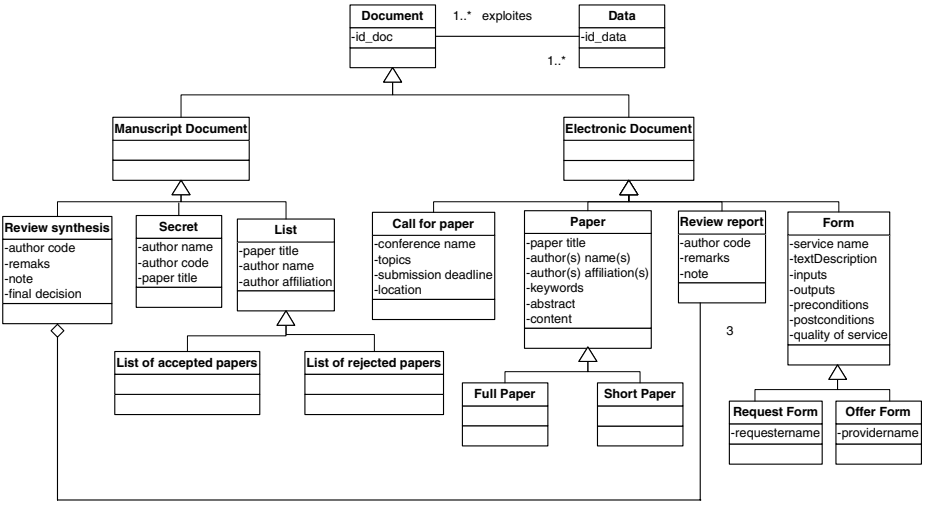

Fig. 2. Informational model: the ontology of the reviewing papers case study

The manuscripts documents are the following:

- *Secret*, which is produced by the chairman for codifying and decodifying papers when we consider anonymous authors;
- *List of accepted papers*, which is produced by the chairman for mentioning the accepted papers including the author's names and their affiliations.
- *List of rejected papers*, which is also produced by the chairman for mentioning the rejected papers including the author's names and their affiliations.
- *Review* synthesis, which is produced by the chairman and corresponds to an aggregation of a set of review reports.
- The electronics documents are the following:
- *Paper*, which is submitted by the author to the conference;
- *Review report,* which is filled by an anonymous reviewer containing his remarks and his evaluation note.
- *Call for paper*, which is produced by the Program Organization Chair and contains the necessary information about the conference such as submission deadline, categories of papers, topics and so on.
- *Request form (or call for reviewers),* which is used by the chairman in order to express its needs for reviewer recruiting;
- *Offer form (or bid)*, which is used by the reviewer in order to describe its capabilities for reviewing papers.

These documents also exploit data which can be structured in information sources. The following figure gives an overview of our informational model described by means of UML class's diagram (see figure 2).

3.2 The Organizational Model

Our organizational model is based on the Agent-Group-Role Meta model (AGR for short) suggested by [10]. This meta model is one of the frameworks proposed to

define the organizational dimension of a multi-agent system, and it is well appropriate to the IOW context. Several reasons justify the interest of this meta model: (i) it eases security: what happens in a group cannot be viewed from agents that do not belong to that group. (ii) adding dynamically a software component into the kernel of the application is easy because creating a new group or playing a new role may be seen as a plug-in process when a software component is integrated into an application. (iii) it supports coherent exchange because a role describes the constraints (obligations, requirements, skills) that an agent should satisfy to obtain a role. Moreover, our organizational model extends classical organizational models [11] by adding an original component called "Matchmaker" as it is presented in [12]. This component is very useful in the context of loose IOW since it helps to connect a requester (for instance chairman) to a provider one (for instance reviewer). More precisely, our organizational model is organized around the following components:

- Three types of groups: Program Committee, Authors and Matchmakers.
- Two types of agents: performer or non-performer.
- Four roles: Author (if the paper is co-authored, the corresponding author is the first in the list); Chairman; Reviewer and Matchmaker.

The following figure gives an overview of our organizational model by means of UML class's diagram (see figure 3).

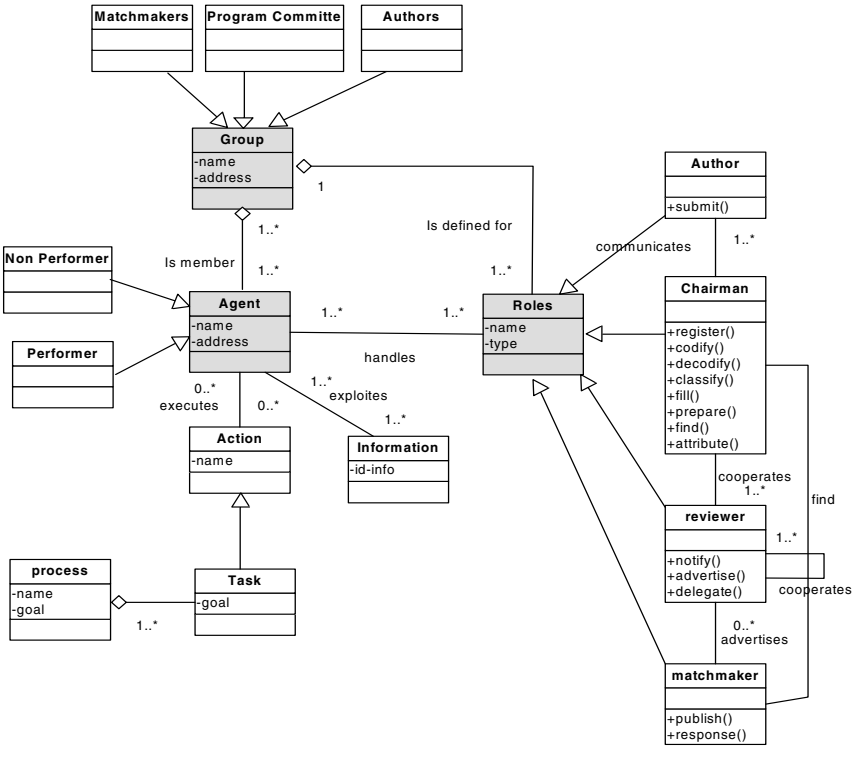

Fig. 3. Organizational model based on AGR meta model

Remark 1. Since we are in loose context, all potential partners are not known. To solve this difficulty, we propose the use of the notion of role seen as an abstraction that can be used without referring explicitly to the individual actors in a workflow but rather to the capacity they must have.

Remark 2. In our case study, we have only one matchmaker, which is specialized in the conference organization domain.

3.3 The Process Model

The reviewing papers process is made of several coordinated tasks and described with a PNO (see figure 4). In this figure, the left hand side net corresponds to the behavior of the chairman. In the middle, the transition in "grey" represents the behavior of the matchmaker, and on the right hand side net we have the behavior of a potential reviewer. Let us detail the tasks (transitions) composing the process:

1. *Registering of paper* by the chairman consists in entering and saving the different components of a paper (e.g. the author(s), received date, keywords etc) submitted by the author.
2. Codifying of paper by the chairman consists in deleting author (s) name (s) of already registered paper, and creating a secret document containing the attributed code for author(s), which helps the chairman after receiving review reports to decodify papers.
3. *Classifying of paper* by the chairman consists in gathering the anonymous paper by topic.
4. *Finding reviewers* consists in publishing requests (or call for reviewers) according to a precise and clear format. This publication by the chairman with destination for the matchmaker; each request form clearly describes the capabilities of the required reviewers. The role of the matchmaker is first to select the best partners and then to return the identities of partners to the chairman. We assume that the chairman and the reviewer share the same form (see the informational model in section 3.1) in order to facilitate the matching process.
5. *Attributing of paper* by the chairman consists in assigning a set of reviewers to a paper.
6. *Evaluating of papers* consists in judging papers by the assigned reviewers. Each evaluation is the review report filled by the corresponding reviewer. Moreover, each reviewer may have its own reviewing process as we have mentioned it in table 1. In this way, we represent the transition "Evaluating of papers" as a "black box" for the others partners.
7. *Collecting and summarizing review reports* by the chairman consists in erasing the anonymous mentioning off the review report and preparing a review synthesis.
8. *Preparing two lists* by the chairman consists in producing i) a document called "list of accepted papers" making appear the list of accepted papers as well as the authors and ii) a document called "list of rejected papers" containing the same information.

Remark 3. The places correspond to classes of the informational model and the organizational model.

Remark 4. To solve the distribution of component processes, we use two mechanisms: connection places (in black in the figure) and a Matchmaker in "grey".

Remark 5. For clarity reason, we do not give the detail of each transition.

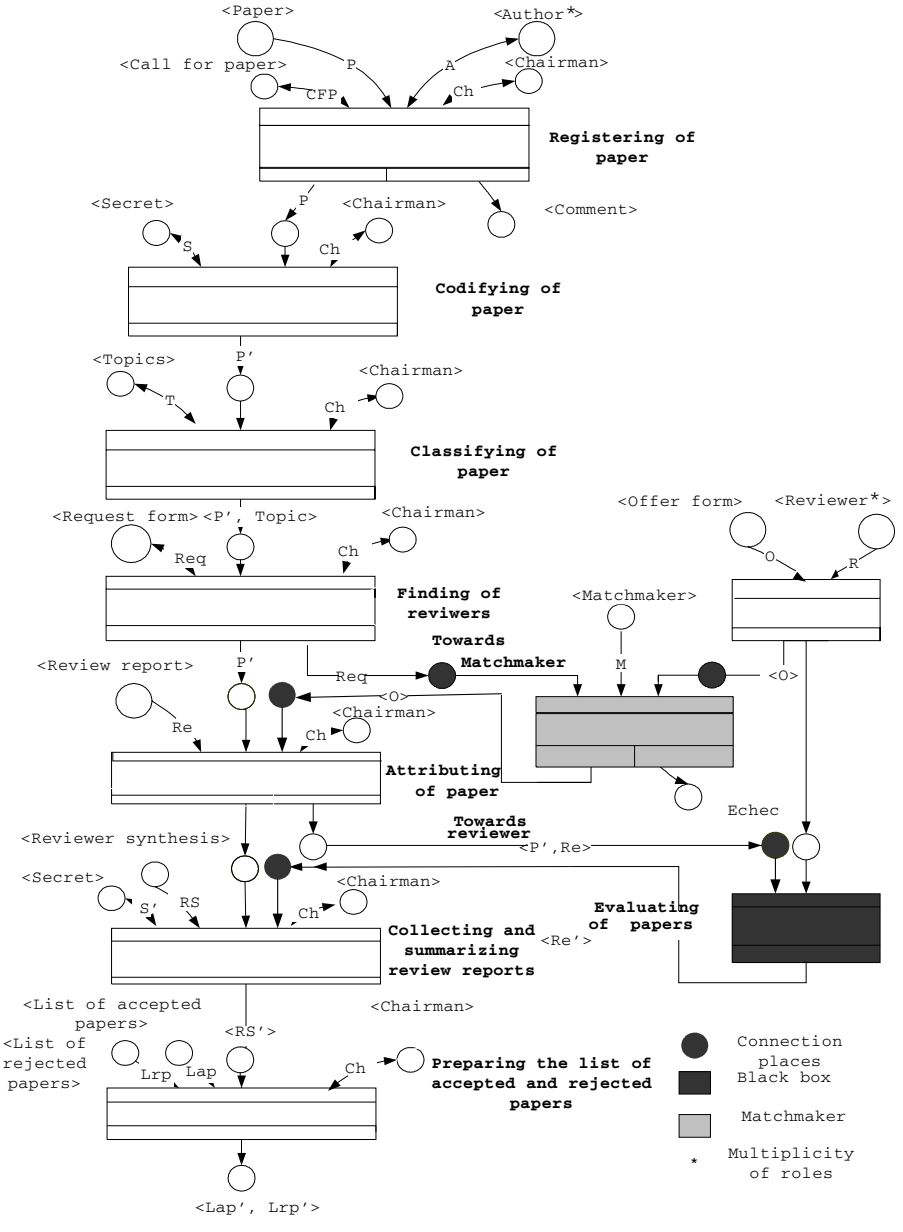

Fig. 4. Modelization of process model by means of Petri Nets with Objects

4 Discussion and Conclusion

The design of loose IOW remains insufficiently addressed. Existing propositions in the literature are rather dedicated to tight IOW ([13], [14] and [15]), and they do not really make the glue between the workflow dimensions i.e. information, organization and process. Theses works only focus on the process dimension by providing interaction models to support distributed execution of component processes. For instance, [13] only focuses on execution aspect of processes by proposing Web services based architecture to support dynamic inter-organizational business processes, and it does not concentrate on design aspect of processes. [14] only proposes a model supporting dynamic heterogeneous workflow process interconnection. Even if [15] deals with the design and execution aspects of processes, the resulting language "YAWL" -which extends Petri Nets with some additional patterns-, does not make the glue between the three-workflow dimensions. We believe our solution is currently unique in trying to take into account the three dimensions of a workflow in a coherent framework. This is made possible thanks to the use of Petri Nets with Objects formalism.

In this paper, we have presented a solution based on an approach the s*eparation of aspects,* and a formalism the *Petri Nets with Objects (PNO)* for the design of loose IOW. To better illustrate our solution, we have chosen the well-known "reviewing papers" case study. During the design of loose IOW, we have taken into account its three specifics features, namely the distribution, the autonomy and the heterogeneity. Regarding the *distribution*, we have integrated in the organizational and process models an original component called "Matchmaker" in charge of connecting dynamically distributed partners. Moreover, thanks to additional places (called connection places) provided by PNO formalism, it's possible to compose and coordinate components processes. Regarding the *autonomy*, we have added the concept of role in the organizational model, which can be considered as an abstraction, which does not refer explicitly to the individual actors but rather to the capacity they must have. Doing so, the workflow initiator does not have to know the potential partners and each partner can keep its internal structure private. Regarding the *heterogeneity*, we have used an ontology enabling the informational model description and data semantic conflict solving.

Our solution forms the basis of a method for the design of Loose IOW applications. It can be organized around three steps:

- Step1. *Creation of the informational model*. We must identify the universe of discourse i.e. the business domain. We use or we create an ontology of this domain to which the partner could adhere. Then the informational model can be built as it is a sub-set of this ontology.
- Step2. *Description of the organizational model.* We instantiate the AGR meta model which structures organizations participating in IOW in terms of Agents, Groups and Roles. To connect dynamically distributed partners, the organizational model must integrate mediator agents. The potential partners, not known at design time, are described through roles.
- Step3. *Description of the process model*. Once the informational and the organizational models are described, we describe the process model as a Petri-Net

with Objects. While some transitions correspond to local tasks, other transitions correspond to tasks to be sub-contracted. In this last case, the corresponding transitions must include an invocation method to call the mediator in charge of finding a partner, and input and output places to respectively provide information and receive result. The links with the two previous models are guaranteed by the two principles: i) The types of the places are classes of the informational or organizational models ii) actions inside transitions are methods of these classes. Once defined, the process model can be simulated, checked and validated. Our case study has been implemented in a simulator called MatchFlow [12] whose objective is to connect workflow service requesters (for instance chairman in our case) to workflow service providers (for instance reviewer in our case). MatchFlow implements the three-workflow dimensions. As future work, we plan to derive OWL-S specification [16] from PNO, which is considered as an appropriate language for Web Workflow Service description allowing providers to publish their capabilities and requesters to express their needs.

Acknowledgments

I would like to thank C. H. who made many suggestions and comments on the first draft of this paper.

References

1. Casati, F., Discenza, A.: Supporting Workflow Cooperation Within and Across Organizations. 15th Int. Symposium on Applied Computing (2000) Como (Italy) 196–202
2. van der Aalst, W.: Inter-Organizational Workflows: An Approach Based on Message Sequence Charts and Petri Nets. Int. Journal on SAMS, 34 (3) (1999) 335–367
3. Divitini, M., Hanachi, C., Sibertin-Blanc, C.: Inter Organizational Workflows for Enterprise Coordination. Chapter 15 of Coordination of Internet Agents, 2001: 369-398
4. 4 Hanachi, C., Sibertin-Blanc, C., Tout, H.: A Task Model for Cooperative Information Gathering. IEEE Int Conference on Systems, Man and Cybernitics, Hammamet, Tunisia, 2002
5. 19th ACM Symposium on Applied Computing, Special Track on Coordination Models, Languages and Applications, Web Site : http://www.cs.fit.edu/~rmenezes/sac04cm/
6. Sibertin-Blanc, C.: High Level Petri Nets with Data Structure. 6th Int. Workshop on PetriNets and Applications (1985) Espoo (Finland)
7. van der Aalst, W.: The application of Petri Nets to Workflow Management. Int. Journal on Circuits, Systems and Computers 8(1) (1998) 21–66
8. van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow Patterns. Int. Journal on Distributed and Parallel Databases 34(1) (2003) 5–51
9. Tatiana, A., Vieira, C., Marco Antonio, C., Luis Gustavo, F.: An Ontology-Driven Architecture for Flexible Workflow Execution. 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress (2004), Brazil, 70-77.
10. Ferber, J., Gutknecht, O., Michel, F.: From Agents to Organizations: an Organizational View of Multi-Agent Systems. AOSE 2003:214 -230
11. van der Aalst, W., Kumar, L., Verbeek, A.: Organizational Modeling in UML and XML in the Context of Workflow Systems. SAC 2003: 603-608

12. Andonoff, E., Bouzguenda, L., Hanachi, C., Sibertin-Blanc, C.: Finding Partners in the Coordination of Loose Inter-Organizational Workflow. COOP 2004: 147 -162
13. Schmidt. R.: Web Services Based Architectures to Support Dynamic Inter-organizational Business Processes. ICWS-Europe 2003: 123-136
14. Baïna, K., Benali, K., Godart. C.: Dynamic Interconnection of Heterogeneous Workflow Processes through Services. In CoopIS/DOA/ODBASE'03
15. van der Aalst, W., Alderd, L., Dumas, M., ter Hofstede., A.: Design and Implementation of the YAWL System. CAISE 2004: 142: 159
16. OWL Services Coalition: http://www.daml.org/services/owl-s/1.1/