ISSN 1870-4069

Knowledge-based Workflow Ontology for Group Organizational Structure

Mario Anzures-García[1], Luz A. Sánchez-Gálvez[1], Miguel J. Hornos[2], Patricia Paderewski-Rodríguez[2]

[1] Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Mexico

[2] Universidad de Granada, E.T.S.I. Informática y de Telecomunicación, Departamento de Lenguajes y Sistemas Informáticos, Spain

{mario.anzures, sanchez.galvez}@correo.buap.mx, {mhornos,patricia}@ugr.es

Abstract. The workflow model represents the set of steps performed by different entities and their execution ordering to control and manage the carried out process on organization. In which, the structure organizational determines the division of labor in order to persons perform the organization process in an appropriated manner. In collaborative applications, this structure determines how the interaction among users, as well as between the users and the application, are carried out. For this reason, a workflow to manage the group organizational structure will be ideal. Although, workflow lacks the expressive power to represent the domain knowledge and the sequence of operations. An ontology describes the knowledge domain through concepts, relations, axioms and instances, but ontology does not specify how these entities should be used and combined. Therefore, in this paper a workflow ontology to control the group organizational structure is proposed. A case study is presented to show the use of knowledge-based workflow ontology for this structure.

Keywords. Workflow model, group organizational structure, ontology, knowledge base, workflow ontology.

1 Introduction

Nowadays, knowledge must be processed computationally to be used not only as individuals but to groups. This leads to require a knowledge base to represent the problem domain. This base can aid to understand, manage and control every performed process by the organizations. In the which, the group work is determined by an organizational structure that is governed by a set of rules that establish its configuration. On the one hand, in the collaborative applications, this structure determines how the communication, collaboration, and coordination among the group members are performed. On the other hand, this configuration can change in accordance with tasks and group needs at a given moment. It is very important that the structure can adapt dynamically to cope with changing organization and own application conditions. For this reason, this structure is modeled by an ontology, since, it can adjust for the changes within the group and to the different working styles of several groups. Moreover, it is one of the strategies for the knowledge structured representation in a formal way, helping to remove ambiguity and redundancy, detecting errors and allowing automated reasoning [1, 2].

In order to manage the organizational structure, it is necessary to specify how the entities should be used and combined, which the ontology does not make. Consequently, a workflow can be used to this, because it refers to coordinated execution of multiple tasks or activities [3, 4]. Nevertheless, it lacks of necessary expressive to knowledge representation. So, a solution is a workflow ontology, which supplies a formal knowledge representation in order to specify the elements and control the steps ordered set of the organizational structure.

Therefore, in this paper, a knowledge-based workflow ontology along with a set of rules is proposed to manage the Group Organizational Structure. In such a way, the knowledge about this structure and special workflow, are formal, and explicitly modeled. Using this knowledge representation scheme and rules, the application can adapt to frequent changes in organizational structure, rules and procedures.

The rest of the paper is organized as follows. Section 2 describes briefly the schemas of knowledge representation. Section 3 explains the ontologies. Section 4 presents the workflow model, and workflow ontology. Section 5 details the workflow ontology of the group organizational structure, and conceptual proof in according to a case study focused on academic virtual space. Section 6 summaries the conceptual results obtained. Section 7 outlines the conclusions and future work.

2 Knowledge Base

Given that knowledge is a portion of all human activities, it is necessary to store it - seizing its meaning - organize it and make it available. So, it requires a representation scheme to provide a set of procedures, which allows the knowledge, to be stored, organized, and to represent the problem naturally. This leads to require a knowledge base to represent the problem domain, as well as can reason and draw conclusions through an inference mechanism for the contents of the knowledge base [5]. The representation scheme must be denoted by a model of some domain of interest in which symbols assist as substitutes for real world artifacts. These symbols must be stored as interest domain statements. The knowledge representation schemes are [6]:

- *Semantic Network* is appropriate for capturing the taxonomic structure of categories for domain objects and for expressing general statements about the domain of interest. Nevertheless, the representation of concrete individuals or even data values does not fit well the idea of semantic networks.

- *Frames* represent a concept, consisting of slots for which fillers are specified. The reasoning in frame-based systems involves both intentional and extensional knowledge contained in the knowledge base of the frame. However, the frames provide more expressive power but less capacity to infer.
- *Rules* come in the form of IF-THEN-constructs and allow to express various kinds of complex statements. Rule-based knowledge representation systems are especially suitable for reasoning about concrete instance data. Complex sets of rules can efficiently derive implicit such facts from explicitly given ones. They are problematic if more complex and general statements about the domain shall be derived, which do not fit a rule's head [6].
- *Logic* is the dominant form of knowledge representation, since is used to provide a precise formalization and axiomatization of problem domain, which is ideal for representing and processing knowledge within computers in a meaningful way. Nowadays, all symbolic knowledge representation and reasoning formalisms can be understood in their relation to First-order (predicate) logic, therefore, this is the prevalent and single most important knowledge representation and reasoning formalism. First-order logic allows one to describe the domain of interest as consisting of objects, i.e. things that have individual identity, and to construct logical formulas around these objects formed by predicates, functions, variables and logical connectives [7]. Description logic [8] is essentially a set of decidable fragments of first-order logic and is expressive enough such that it has become a major knowledge representation and reasoning paradigm. A description logic theory consists of statements about concepts, individuals, and their relations. Individuals correspond to constants in first-order logic, and concepts correspond to unary predicates. Concepts can be named concepts or anonymous (composite) concepts. Named concepts consist simply of a name, which will be mapped to a unary predicate in first-order logic. Composite concepts are formed from named concepts by using concept constructors, similar to the formation of complex formulas out of atomic formulas in first-order logic [9].

The ontology is an ideal solution to represent the knowledge domain using description logic symbols, which allow to specify it of a simple way, and readable for both human and machines; as well as perform much deeper reasoning through the machine. It facilitates a knowledge base in order to provide semantic, common understanding, communication and knowledge sharing on the domain of interest and a knowledge reasoning, carrying out an inference process to reach conclusions on the knowledge base by means on a reasoner, inference rules and query languages.

3 Ontologies

Ontology, according to Gruber, *is a formal and explicit specification of a shared conceptualization* [9]. *Conceptualization* refers to an abstract model of some phenomenon in the world by identifying the relevant concepts of this. *Explicit specification* means that the type of concepts used, and the constraints on their use are explicitly defined. Thus the ontology is a high level formal specification of a certain knowledge domain, providing a simplified and well defined view of domain.

3.1 Ontology Structure

The specification of the ontology is defined through of the following components [9]:

- *Classes*: Set of classes (or concepts) that belong to the ontology. They may contain individuals (or instances), other classes, or a combination of both with their correspondents attributes.
- *Relations*: These define interrelations between two or several classes (object properties) or a concept to a data type (data type properties).
- *Axioms*: These are used to impose constraints on the values of classes or instances. Axioms represent expressions in ontology (logical statement) and are always true if used inside the ontology.
- *Instances*: These represent the objects, elements or individuals of an ontology.

Nowadays, the ontologies (particularly in OWL - Ontology Web Language) have been extended with rules by Semantic Web Rule Language (SWRL), which use other predicates than just class or property names:

- *class expressions:* These are arbitrary class expressions, not just named classes.
- *property expressions:* The only operator available in OWL 2 for creating property expressions is inverse of object property; however, the same effect can be achieved by exchanging the property arguments, so there is no need to use property expressions in SWRL
- *data range restrictions*: They specify a type of data value, like integer, date, union of some XML Schema types, enumerated type.
- *sameIndividual and differentIndividuals*: These are used for specifying same and different individuals
- *core SWRL built-ins:* They are special predicates defined in SWRL proposal which can manipulate data values, for example, to add numbers custom SWRL built-ins —it can define own built-ins using Java code.

3.2 Ontology Languages

Like the knowledge representation and reasoning, ontologies require a logical and formal language to be expressed. In the area of Artificial Intelligence many languages have been developed for this purpose, some based on First-order (predicate) logic as KIF and Cycl providing modeling primitives and the possibility of redoing formulas that enable them to become in terms of other formulas. Other Frames-based languages with more expressive power but less inference capability as Ontolingua and F-Logic; others based on descriptive logic that are more robust in the power of reasoning as a Loom, OIL, DAML + OIL and OWL. OWL [10, 11] is an ontology language recommended by the W3C for use in the Semantic Web. The OWL representational facilities main are directly based on Description Logics. This basis confers upon

OWL a logical framework, including both syntax and model-theoretic semantics, allowing it is a knowledge representation language capable of supporting a knowledge base and a practical reasoning and effective. Moreover, the Description Logic provides readily available reasoners such as Pellet [12] and HermiT [13], both of which have been extended to handle all of OWL. OWL ontologies can also be combined with rules using the new W3C Rule Interchange Format (RIF) standard [14]. For the development of ontologies are used tools, which provide graphical interfaces that facilitate the knowledge representation and reasoning. This article focuses on Protégé, which is an engineering tool open source ontology and a knowledge-based framework. Protégé is widely used for the development of ontologies, due to the scalability and extensibility with lots of plugins; and by facilitate inference knowledge through reasoners, query languages and rules. Ontologies in Protégé can be developed in a variety of formats, including OWL, RDF (S), and XML Schema.

Summarizing, ontology establishes the vocabulary used to describe and represent knowledge, and to facilitate machine reasoning.

4 Workflow Model

Workflow is seen as an automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another, for action, according to a set of procedural rules [15, 16]. The workflow management, commonly, is achieved through three constructs: routes (it represents task sequences), rules (it defines routing and role constructs), and roles (it represents one who is responsible for a task). A suitable management of a workflow requires the following aspects [17]:

- *Expressiveness:* It should provide constructs to represent conditional mapping relationships between roles and actors based on the organizational model as well as complex business rules, including exceptional rules.
- *Model verification:* It should allow analysis that assures the correctness of workflow specification, together with checking the occurrence of inconsistent, redundant, and incomplete rules as well as non-terminality of processes.
- *Change management:* It should allow easy development of the propagation mechanisms against changes on the organizational structure and rules as well as organizational procedures to assure the correctness of a workflow model.

The ontology provides enough expressivity by the supplied structure (concepts, relations, instances, and axioms); the ontology verification is accomplished through reasoners (Pellet, and HermiT); and the change management on the ontology can be made modifying, adding, and/or deleting concepts, relations, and/or instances. Consequently, the ontology is ideal to workflow management. So, recently, it has paid special attention to the development of workflow ontologies. It presents a collaborative workflow for terminology extraction and collaborative modeling of formal ontologies using two tools Protege and OntoLancs [18]; it allows the development of ontology cooperatives and distributed based on dependencies management between ontologies modules [19]; it shows an ontology-based workflows for ontology collaborative development in Protégé [20], it presents the combination of workflows with ontologies to design way formal protocols for laboratories [21], and it proposes a workflow ontology for the preservation digital material produced by an organization or a file system [22].

All these works focused on building workflow ontologies to represent collaborative work in different areas, however, this paper presents a workflow ontology to manage the group structure organizational in the collaborative domain.

5 Workflow Ontology for Group Organizational Structure

In the collaborative applications, the shared work is supported for sessions, which denote the shared workspace. This type of applications typically provides a shared workspace by a session manager. On the one hand, this manager allows to establish the session (i.e., it permits to set up the connection, to create and manage meetings, and to enable a user to join and leave a session using a simple user interface). On the other hand, this manager allows defining the group organizational structure that states how sessions are organized to accomplish the shared work. In general, collaborative applications do not separate the mechanisms to establish the shared workspace from the group organizational structure. Therefore, in this paper, it is considered the proposed separation in [1]; because it allows us to support changes in the group at runtime and specify this organizational structure through a policy.

This structure can be hierarchical (one member has a greater status than other members, such as in meetings with department chief) or not hierarchical (the participants have equal status, for example, informal meetings of university professors). These structures are ruled by a policy, which determines how the group members will be organized. In groupware, this policy is called session management policy, which establishes the group organizational structure in terms of the functions that group members will carry out. This policy has been modeled by ontology [1, 2], adjusting to group dynamic nature and evolving needs of the same.

However, it is required the organizational structure management, thus, a workflow ontology of the group organizational structure is proposed.

5.1 Workflow Ontology Description

This ontology (see Fig. 1) defines that: the group organizational structure (GOS) is made up of users, and is governed by one policy (Pcy), which establishes a hierarchical organizational structure or not-hierarchical organizational structure by means of the roles (one or more - Rls) that users can play. Each role designing one status (Stt - which founds the role priority in the group), one right/obligation (R/O -set privileges for the user in the application), and tasks set (Tsk - which are role functions) and they can be composed of one or more activities (Atv - which are operations that allow users to achieve a given goal) that use resources (Rsc). For each task indicates the event (Evt) that triggers it, its precedence (PTk - i.e., tasks order), and its type (*Sequential-task - SqT;* one activity follows the other. *Parallel-Task -PrT*; these happen at the same time, but they use different objects, and no interference between them can occur. *Partially - Concurrent-Task - PCT;* it refers to tasks that can be active at the same time but there is no simultaneous modification of any object). *Fully-Concurrent-Task - FCT;* it occurs when two or more simultaneous tasks to modify rights to same set of objects). It establishes the application stages (Stg - it reflects each of the collaboration moments). For each stage determines the order of them (Stage Precedence - SPc), the tasks that correspond to these, and precedence of the tasks (STk) in the same.

The specification of the Workflow Ontology is carried out through of the following steps (WOS):

1. Starting Workflow (StW).
2. Defining the GOS name.
3. Determining the Policy name.
4. Establishing the Roles of the groupware.
   - 4.1. Designing a Status to role.
   - 4.2. Signalizing a Right/Obligation to role.
   - 4.3. Specifying the tasks that each role carries out in the groupware.
     - 4.3.1. Designating the event that triggers each Task.
     - 4.3.2. Indicating the task type (sequential, parallel, partially concurrent, and fully concurrent).
     - 4.3.3. Mark out the Activities of each Task.
       - 4.3.3.1. Defining the resources to the activity.
       - 4.3.3.2. If there are more resources go to step 4.3.3.1, else go to step 4.3.3.
     - 4.3.4. If there are more activities of one task go to step 4.3.3, else go to step 4.3.
   - 4.4. If there are more tasks for one role go to step 4.3, else go to step 4.
5. If there are more roles for the application go to step 4, else go to step 6.
6. Establishing the Stages of the collaborative application.
   - 6.1. Determining the order of the stage.
   - 6.2. Assigning tasks to a stage.
   - 6.3. Indicating the tasks' precedence in each stage.

5.2 Proof Conceptual of the Workflow Ontology

The study case consists in the development of a groupware for Managing Departmental Test (MDT) of the *Facultad de Ciencias de la Computación de la Universidad Autónoma de Puebla*. The Departmental Test (DET) homogenizes the teaching of a subject, i.e. it guarantees that all teachers encompass the same percentage of the academic program. For this reason, it requires a shared workspace that allows professors to manage and apply a DET.

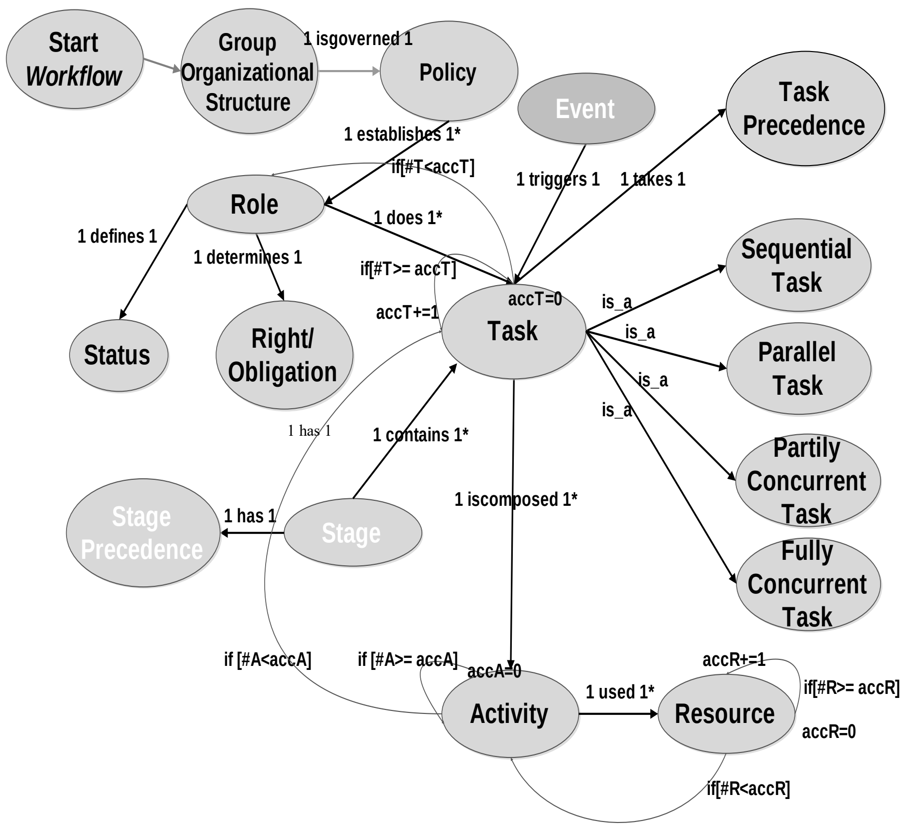

Fig. 1. Workflow ontology of the group organizational structure.

For reasons of space, the workflow ontology, that displays the knowledge representation in a conceptual, and formal manner; and a Table, that shows the workflow ontology elements, will be presented in a partial form.

Several roles are considered in MDT: The Manager (Mgr) who configures the application (CfA) and has status equal to 1, so, he/she registers the users, who play the other four roles, the knowledge areas, and the subjects that are a part of them. The Area Coordinator (ArC) with status 2, who manages the test (MaT), so, he/she registers the TeC and schedules the professors' meetings, related with the same subject. The Test Coordinator (TeC) with status 3, who organizes the test (OgT), so, he/she put in order the completion of each test, requesting and agreeing the number of tests to be applied, as well as on the dates and the number of questions, which will be included; then he/she will post the test and the classroom, where each Professor (Pfs) will apply it. The Pfs with status 4, who generates the test (GeT), thus, he/she will propose and vote the date when the test will be performed, so as the number of questions contained in the exam. The Students (Sds) with status 5, who views scores (ViS) of the test, so, he/she will look up the information about the date and classroom, where the test will be carried out, as well as to find the grades obtained for each subject. In general, the five roles must register to join at the session.

The collaborative application for managing the MDT has four stages (Stg):

1. Test Configuration (TCf) with stage precedence (SPc) equal to 1. In this stage only the role Mgr participates; executing the tasks of: Authenticate (Aut) him/herself (which is triggered by the Event accesses to the system), Create, Read, Update, and Delete (CRUD) for ArC (which are activated by the Event manages to ArC), Area (which is initiated by the Event manages to Area), and Subject (which is originated by the Event manages to Subject).
2. Test Preparation (TeP) with SPc equal to 2. In this stage, the roles Mgr and TeC enter to the same; the former performing the tasks of Aut, CRUD TeC, Proposing Meeting Date, and Setting Date (StD) with the activities of writing date (Wrd —that used the resources label, and calendar) and sending date (sed), these two tasks are triggered by the Event schedule meeting (ScM); the latter executing the tasks of Aut, and CRUD Pfs. The tasks Aut, Crud TeC and Pfs are triggered by the same Events of similar tasks corresponding to the role Mgr.
3. Test Elaborating (TeE) with SPc equal to 3, and two roles joining: 1) The role TcC carries out the tasks of: Aut him/herself with PTk equal to 1, Proposing Date (PD) with PTk equal to 2, Setting Date (SD) with PTk equal to 3, Proposing Number of Questions (PNQ) with PTk equal to 4, Setting Number of Questions (SNQ) with PTk equal to 5, and Posting Questions (PQ) with PTk equal to 6. The tasks 2 and 3 are activated by Event called defining the test date (DTD), while the 4, 5, and 6 by Event establishes test questions. On the other hand, the task 3 is composed of the activities: selecting date (ed), confirming date (CD), and submit date (uD). The first use the resource calendar; the second use the resource confirmation bottom, and thee third the acceptation bottom (AB). 2) The role Pfs executes the tasks: Aut him/herself, Consulting Proposals, Choosing Date, Loading Proposal of questions, Downloading Exercises of the test, Choosing Questions of the test, Posting Notice, and Posting Message.
4. Test Results (TeR) with SPc equal to 4. Four roles (ArC, TeC, Pfs, and Stu) participate in this Stg. The role ArC implements the tasks of Downloading Scores, Generating Reports, Loading Statistics, Posting Messages, Posting Notices, and Scheduling Test. The role TeC to performing the same tasks that the role ArC; in addition, he/she carries out the tasks of Loading DET and Posting classroom. The role Pfs effects the tasks of: Loading Scores, Download Scores, Register in Group, Loading Scores, Posting Messages, Posting Notices, and Scheduling Test. The role Stu carries out the task of Download Scores, Posting Messages, and Scheduling Test.

The Table 1 shows the workflow ontology elements that constitute the knowledge base and that are gotten of the case study with respect to the Stage TeE and the role TcC. This table is expressed in terms of the ontology specification, as well as, of the rules that determine the execution flow of the steps to be performed by this workflow.

Table 1. Workflow ontology specification.

WOS	Cpt	CoI	CAI	Relation	Cdy	TaC	TCI	TCA	TAI	Rule
1	STW									
2	GOS	GOS-MDT	1	isgoverned	1	Pcy	PMDT	NPcy	1	if [[GOS](?X) and [Pcy](?Y)] (?X,?Y)] then [isgoverned Pcy] (?X,?Y)
3	Pcy	PMDT	1	establishes	1*	Rls	Mgr, ArC, TcC, Pfs, Sds	#R, accR	5, 0	If [[Pcy](?X) and [Rls](?Y)] (?X,?Y)] then if [accR<=#R] then establishes Rls and accR+=1 else if [[Stg](?Z) and [Tsk](?W)](?Z,?W)] then contains Tsk] (?Z,?W)] (?X,?Y)
4	Rls	TcC	1	defines	1	Stt	3			if [[Rls](?X) and [Stt](?Y)] (?X,?Y)] then defines Stt] (?X,?Y) and con+=1;
4.1	Rls	TcC	1	determines	1	R/O	OgT			if [[Rls](?X) and [R/O](?Y)] (?X,?Y)] then determines R/O] (?X,?Y)
4.2	Rls	TcC	1	does	1*	Tsk	Aut, PD, SD, PNQ, SNQ, PQ	#T, accT	5, 0	if [[Rls](?X) and [Tsk](?Y)] (?X,?Y)] then if [[accT<=#T] then does Tsk and accT+=1; else if [[Pcy](?W) and [Rls](?Z)] (?W,?Z)] then establishes Rls] (?X,?Y).
4.2.1	Evt	ScM	1	triggers	1	Tsk	DTD			if [[Evt](?X) and [Tsk](?Y)] (?X,?Y)]then [trigger Tsk] (?X,?Y)
4.2.2	Tsk	SD	1	is_composed	1*	Atv	eD, CD, uD	#Atv, accA	3, 0	if [[Tsk](?X) and [Atv](?Y)] (?X,?Y)] then [is_composed Atv] (?X,?Y)
4.2.2	Atv	uD	1	uses	1*	Rsc	AB	#A, accC	1, 0	if [[Act](?X) and [Rsc](?Y)](?X,?Y)] then [uses Rsc and accA+=1] (?X,?Y); if [[Rsc](?Z) and [accC>=Rsc]](?Z) then [conA*=1 and goes [[Act](?X) and [Rsc](?Y)](?X,?Y)]
4.2.3	Tsk	SD	1	takes	1	PTk	3			if [[Tsk](?X) and [PTk](?Y)](?X,?Y)] then [takes PTk] (?X,?Y)
5	Stg	TeE	1	contains	1*	Tsk	Tcf, TeE, TcC, TEr	#S, accS	4,0	if [[Stg](?X) and [Tsk](?Y)](?X,?Y)] then contains Tsk] (?X,?Y)
6	Stg	TeE	1	supports	1	SPc	3			f [[Stg](?X) and [SPc](?Y)](?X,?Y)] then supports SPc] (?X,?Y)

Therefore, this table presents the following columns; allowing us to proof the ontology: Concepts (Cpt), Concept Instance (CoI), Concept Attribute (CAt), Concept Attribute Instance (CAI), Relation (Rel), Cardinality (Cdy), Target Concept (TaC), Target Concept Instance (TCI), Target Concept Attribute (TCA), Target concept Attribute Instance (TAI), and Rules (Rul).

6 Conceptual Results

Summarizing, the workflow ontology allows us to know: the roles participate in the interaction (fully concurrent, partially concurrent, parallel, and/or sequential) and in what order do; the resources used by each user for accomplishment each activity. This is possible, thanks that this ontology establishes:

1. The roles that access to each stage.
2. The priority of each stage.
3. The task executed in each stage and its priority on it.
4. The task carried out by each role.
5. The activities that compose each task.
6. The resources used in each activity.
7. Who performs each type task (SqT, PrT, PCT, and FCT).

7 Conclusions and Future Work

This paper has presented a workflow ontology to manage the group organizational structure. On the one hand, this ontology is created of the knowledge base (which aid to understand, manage and control every performed process by the organizations) provides by the session management policy ontology; allowing us to adjust for the changes within the group and to the different working styles of several groups, and helping to remove ambiguity and redundancy. On the other hand, a workflow is specified through this ontology in order to model how the organizational structure entities should be used and combined, as well as; it should be controlled the steps ordered set of the organizational structure.

The future work is orientated to specify a methodology to develop groupware, starting with workflow ontology described in this article.

References

1. Anzures-García, M., Sánchez-Gálvez, L.A.: Policy-based group organizational structure management using an ontological approach. In: Proc. International Conference on Availability, Reliability and Security (ARES), 807–812 (2008)
2. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., Paderewski-Rodríguez, P.: Ontology-Based Modelling of Session Management Policies for Groupware Applications. Lecture Notes in Computer Science, Springer, Heidelberg, 4739, 57–64 (2007)

3. Fischer, L.: Workflow Handbook. Future Strategies Inc., Lighthouse Point, FL (2004)
4. Marinescu, D.: Internet-Based Workflow Management: Toward a Semantic Web. Wiley, New York (2002)
5. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., Paderewski-Rodríguez, P.A.: Knowledge Base for the Development of Collaborative Applications. Engineering Letters, 23(2), 65–71 (2015)
6. Grimm, S., Hitzler, P., Abecker, A.: Knowledge Representation and Ontologies. In: Semantic Web Services: Concepts. Technologies and Applications, Springer-Verlag, 51–105 (2007)
7. Russel, S., Norvig, P.: Artificial Intelligence - A Modern Approach. Prentice-Hall (1995)
8. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. F.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
9. Gruber, R.: A translation approach to portable ontology specification. Knowledge Acquisition, 5, 199–220 (1993)
10. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language semantics and abstract syntax. W3C Recommendation (2004)
11. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From SHIQ and RDF to OWL: The making of a web ontology language. J. of Web Semantics, 1-1, 7–26 (2003)
12. Pellet: OWL reasoner. Maryland Information and Network Dynamics Lab, Available: http://www.mindswap.org/2003/pellet/index.shtml (2003)
13. Motik, B., Shearer, R., Horrocks, I.: Optimized reasoning in description logics using hypertableaux. In: Proc. of the 21st Int. Conf. on Automated Deduction (CADE-21), Lecture Notes in Artificial Intelligence, Springer, 4603, 67–83 (2007)
14. RIF RDF and OWL Compatibility. W3C Recommendation, Available: http://www.w3.org/TR/rif-rdf-owl/ (2010)
15. Allen, R., Workflow: An introduction. In Workflow Handbook, L. Fisher, Ed. Future Strategies, Lighthouse Point, FL, 15–38 (2001)
16. Marshak, R.T.: An overview of workflow structure. In: Proc. of Workflow. San Jose, USA: Future Strategies Inc, 13–42 (1994)
17. Lee, H.B., Kim, J.W., Park, S.J.: KWM: Knowledge-based Workflow Model for Agile Organization. Journal of Intelligent Information Systems 13(3), 261–278 (1999)
18. Gacitua, R., Arguello-Casteleiro, M., Sawyer, P., Des, J., Perez, R., Fernandez-Prieto, M.J., Paniagua, H.: A collaborative workflow for building ontologies: A case study in the biomedical field. Research Challenges in Information Science, 2009, RCIS 2009, Third International Conference, 121(128), 22–24 (2009)
19. Kozaki, K., Sunagawa, E., Kitamura, Y., Mizoguchi, R.: A Framework for Cooperative Ontology Construction Based on Dependency Management of Modules. In: ESOE, CEUR Workshop Proceedings, CEUR-WS.org, 292, 33–44 (2007)
20. Sebastian, A., Noy, N.F., Tudorache, T., Musen, M.A.: A Generic Ontology for Collaborative Ontology-Development Workflows. In: Proceedings of the 16th international conference on Knowledge Engineering: Practice and Patterns (2008)
21. Maccagnan, A., Riva, M., Feltrin, E., Simionati, B., Vardanega, T., Valle, G., Cannata, N.: Combining ontologies and workflows to design formal protocols for biological laboratories. Automated Experimentation, 2(3) (2010)
22. Mikelakis, M., Papatheodorou, C.: An ontology-based model for preservation workflows. In: Proceedings of the 9th International Conference on Digital Preservation (2012)