# A Knowledge Base for the Development of Collaborative Applications

Mario Anzures-García, Luz A. Sánchez-Gálvez, Miguel J. Hornos, and Patricia Paderewski-Rodríguez

***Abstract*—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. In such a way, a knowledge base can be produced. 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. In this work, the model used is a workflow ontology. This proposal specifies the set of steps —along with their order of execution— performed by different symbols to develop collaborative applications. Ontology is one of the strategies for the structured representation of a chosen knowledge domain in a formal way, which removing ambiguity and redundancy, detecting errors, and allowing automated reasoning. A case study is presented to show the use of a workflow ontology as a knowledge base for the development of Collaborative Applications.**

***Index Terms*— collaborative application, knowledge base, knowledge representation, ontology-based knowledge, workflow ontology**

## I. Introduction

The logic is the dominant form of Knowledge Representation, since is used to formalize the main knowledge representation schemes such as: Semantic Networks, Frames, and Rules. Due to that it supplies a precise semantics, in such a way avoid its ambiguity and vagueness in order to computationally process them. This is because the logic provides an accurate formalization and axiomatization of problem domain, which is ideal for knowledge representing on computers in a meaningful way. Nowadays, all symbolic knowledge representation formalisms can be understood in their relation to First-order (predicate) logic. Descriptive logic [1] is essentially a set of decidable fragments of first-order logic, and is expressive enough as to become a key knowledge representation paradigm. Ontology provides an ideal solution, for it represents the domain knowledge using descriptive logic symbols, which allows specifying it in a simple, readable way for both humans and machines. It facilitates a knowledge base to provide semantic, common understanding, communication, and shared knowledge about the domain of interest.

Collaborative applications must provide an appropriate infrastructure to back up group work and support the dynamic structure of the organizations in runtime. In such a way that they can represent inherent knowledge in the applications that support groups of people engaged in a common goal, and provide an interface for a shared environment. This paper, tries to capture the resulting knowledge on the development of such applications so, it proposes a workflow ontology to create collaborative applications. This represents a knowledge base with all the necessary symbols to specify the elements for building these applications. Therefore, this work focuses both dynamism of group and users' interaction by providing a common and well defined vocabulary, a list of terms and meanings, which describes concepts and relationships among them, along with axioms in a formal way in order to provide to the collaborative applications a complete and coherent specification for its development.

The rest of the paper is organized as follows: Section 2, describes briefly the ontology-based knowledge; Section 3, explains the inherent knowledge in the collaborative applications; Section 4, presents the workflow ontology for collaborative applications and a case study focused on academic virtual space; Section 6, outlines the conclusions and future work.

## II. Ontology-based knowledge

In recent years, the use of ontologies has extended in diverse areas such as medicine [2]; bioinformatics [3]; groupware [4, 5]; mainly, because they allow a formal explicit specification of a shared conceptualization of certain domain of interest. Conceptualization refers to an abstract model of some knowledge of the world through the identification of relevant concepts of this. Explicit specification, means that the type of concepts used and the constraints on their use are explicitly defined. Formal, reflects the fact that the ontology should be machine-readable. Shared, represents the notion that an ontology captures consensual knowledge that is not reserved to some individual, but it is accepted by a group. So, it is said that ontology establishes the vocabulary used to describe and represent domain knowledge to facilitate machine reasoning. According to Gruber [6], domain knowledge in ontologies can be formalized using four components: concepts, relations, axioms and instances. Therefore, it can

Manuscript received December 15, 2014; revised January 2, 2015.

M. Anzures-García, and L. A. Sánchez-Gálvez are with the Faculty of Computer Science, Benemérita Universidad Autónoma de Puebla, Puebla, 52570 México e-mail: {mario.anzures, sanchez.galvez}@correo.buap.mx.

M. J. Hornos, and P. Paderewski-Rodríguez F. A. are with the Departamento de Lenguajes y Sistemas Informáticos, E.T.S.I. Informática y de Telecomunicación, Universidad de Granada, Granada, 18071 Spain e-mail: {patricia, mhornos}@ugr.es.

**(Advance online publication: 24 April 2015)**

be concluded that ontologies are an ideal solution for knowledge representation, since it provides a set of symbols through a formal and structured vocabulary.

Protégé provides graphical interfaces that facilitate the knowledge representation, and it is used to develop ontologies in Web Ontology language [7]. The OWL representation facilities are directly based on Description Logics. This basis confers OWL with a logical framework, including syntax and model-theoretical semantics, allowing a knowledge representation language capable of supporting a knowledge base.

## III. Knowledge Base of the Collaborative Applications

Information technologies are generating a necessity for collaborative applications, which were unthinkable only a few years ago. On the other hand, they allow collaborative applications to work appropriately. Therefore, there are a large number of ideas, methodologies, models, and technics in the area of collaborative applications investigation, and it seems this number will continue to rise. For example, tools or frameworks (such as Groupkit [14], ANTS [15], and SAGA [16]), architectures (e.g., Clock [17], and Clover [18]), and methodologies (AMENITIES [19], ClAM [3], and TOUCHE [4]) have been built to design and develop collaborative applications. Consequently, there are a large amount of terms defined —such as group, role, task, activity, resource, organizational structure, session, application— that can be considered the foundation for the development of these kinds of applications. However, these terms can have different meaning and connotation in according to the tool, architecture, and methodology where they were defined. On the other hand, these proposals lack of the necessary adaptability to conform naturally to the inherent dynamics or the distinct scenarios that this kind of applications often presents.

Therefore, this paper proposes a workflow ontology for having an appropriate knowledge base for the structured representation of the collaborative applications' development in a formal way, helping to remove ambiguity and redundancy of the terms used for this development. Furthermore, this ontology unlike the above mentioned methodologies, not only provides the static and dynamic structure of group's organization, but it also specifies the steps to develop this kind of applications and allows its adaptation. This workflow ontology was developed through an extended literature review, which included studies and surveys from multiple venues, such as journals, conferences, and workshops. From which, the above mentioned terms to the knowledge base were extracted. This proposal extends this base by adding other terms to existing, such as status, right/obligation, notification, concurrency, phase, adaptation, etc. As well as; the terms are classified into four aspects: group, interaction, application, and adaptation. The three first are commonly present in literature reviewed on collaborative applications, for example, in the most used definition of these applications. The fourth is applied for adapting them to new collaborative scenarios. So, the group aspect, facilitates the group organization allowing communication, coordination and collaboration of their members (users). The interaction aspect, provides a shared workspace and a mechanism to carry out and control the group interaction. The application aspect, manages the users' access to application, and shows the users' interaction. The adaptation aspect, offers a mechanism to adjust the application to the necessities of the group. From these aspects, a set of terms that will constitute the knowledge base to develop collaborative applications can be inferred. Therefore, the knowledge base for this kind of applications is established as follows.

### *A. Group*

It is responsible for achieving the common goal. The operation group depends on the creation of the **Group Organizational Structure** (**GOS**), which determines who will authorize the users' registration, how an interaction among users is carried out, how are the turns for user's participation defined. The GOS is ruled by the **Session Management Policy —SMP—** that the structure establishes [9]. The SMP defines a hierarchical or not-hierarchical GOS through **Roles** that **Users** can play. These **roles** establish the set of **Rights/Obligations (R/O)** and **Status (St)** within the group, as well as the **Tasks** that can be performed to accomplish a common goal. The **Task** is composed of **Activities,** which use the prevailing shared **Resources** [10]. The **GOS** specification provides the Knowledge on how the interaction among users is achieved; what **Roles** are involved in the group; what **Task** is performed by each **Role,** and what **Resources** are used to achieve a certain goal. Hence, this layer provides the following terms to the knowledge base that will support the development of collaborative applications: **GOS, SMP, Role, User, Rights/Obligations, Status, Tasks**, **Activity, and Resource.** Role, User, Task, Activity, and Resource are terms frequently presented in the collaborative domain. In this work, the Resource considers the resources used by the roles to carry out the tasks, including View elements, such as label, text box, button, etc.; Users can be both persons and systems that take part in the group work; Task is composed of Activities, which is the simpler action. On the other hand; GOS, SMP, Rights/Obligations, and Status are terms added to this base; as GOS can symbolize the static structure both group and organization, the SMP can be adjusted to group's necessities facilitating the representation of the dynamic structure, the Rights/Obligations and Status determine the roles' privileges —i.e. the tasks that a role can play— in the GOS leading to a hierarchical structure.

### *B. Interaction*

It is a key aspect of collaborative applications, as it allows the communication, collaboration, and coordination of the users. The interaction is subordinated to the GOS, and requires of a **Session, a Notification, and a Concurrency** [11] to operate appropriately**.** The first one is a shared workspace where a group will interact. The second one provides the users with the necessary information to support the *group awareness* —users are aware of another member's presence in the *session*, and of the actions that each one of them is carrying out— and supplies a common context on which the activities within the group are performed, and where the information of the shared resources is stored; thus, creating a *group memory* to provide understanding and reasoning around the collaborative process. The third one ensures the consistency of the data being shared; providing collaborating users with dynamically-generated temporary access and permissions, to reduce competing conditions, and to guarantee mutually exclusive resource usage. These permissions depend on the GOS established and on the lock mechanism. Accordingly, Session, Notification, and Concurrency are terms aggregated to the knowledge base. The first is commonly used in the collaborative domain, while Notification and Concurrency are not almost applied in this domain, although they are frequently mentioned. These two terms allow decreasing the probability of conflicts within the group by establishing a common context in order to facilitate the users' interaction.

### *C. Application*

It permits to control and show both the interaction among the users and of the latter with the shared resources. Hence, it presents the **Phases** and **Views**. The former is defined as a global description of the tasks that are active [12]; so it represents each collaborative moment in a collaborative application. The latter is a user interface that allows the interaction between the users and the application. There are three Views: Information **View** (**IV**) which is related to individual information; the **Participant View** (**PV**) that is associated with *group awareness*; and the **Context View** (**CV**) which is connected to group memory. Therefore, the knowledge base is increased with the terms; Phase, IV, PV, and CV. The first one controls the roles that can participate in it, which facilitates the authentication, interaction, and its adaptation among users in the shared workspace. IV, displays the user's information; the PV and the CV, illustrate the information about the group and organization; in addition, they contemplate two key aspects of all collaborative applications —*group awareness*, and *group memory*.

### *D. Adaptation*

It adjusts the Views in accordance to those changes triggered by *notification* and managed by concurrency in such a manner that the application shows the most recent updates while preserving its functionality. Accordingly, it is necessary to monitor the *changes* in the **session** using a **detection** process; in the case of an adaptable process (when the adaptation is carried out by direct intervention of the user) in a non-hierarchical GOS, the *pre-adaptation stage* is performed. This process accomplishes an *agreement,* where all users have to reach a consensus on whether an adaptation process should be performed by a Voting Tool, which offers several kinds of agreements such as the one based on the majority vote or the one based on a maximum or minimum value, etc. In the case of an adaptive process (when the adaptation is automatically performed) or the users have agreed to make an adaptation, an *adaptation flow* process is performed. When an *adaptation flow* cannot be completed, a *reparation* process is invoked —which returns each component to its previous state— then, the users are notified that this adaptation process cannot be achieved [13]. The terms of adaptation and change are added to the knowledge base to indicate which modifications were performed to adjust the application to the new features of the collaborative scenarios.

## IV. A Workflow Ontology for Collaborative Applications

The development of collaborative applications demands to execute a group of coordinated steps, which can be accomplished by means of a workflow. The term workflow typically refers to coordinated execution of multiple tasks or operations [20]. However, workflows lack the expressive power to represent the domain knowledge and the sequence of operations. On the other hand, ontology describes the knowledge domain by means of concepts, relations, axioms and instances; although ontology does not specify how these entities should be used and combined. Special attention has recently been paid to the development of workflow ontologies. The former defines in what way the tasks or operations should be used and combined, and once it has been represented, it may be considered a structure. The latter can symbolize this structure due to its expressive power. So ontology is an ideal solution for workflow representation. There are several examples of workflow ontologies: [21] presents a collaborative workflow for terminology extraction and collaborative modeling of formal ontologies using two tools: Protege and OntoLancs. In [22] allows the development of cooperative and distributed ontologies, based on dependencies management between ontologies modules.

In [23] shows an ontology-based workflows for ontology collaborative development in Protégé. In [24] presents the combination of workflows with ontologies to design formal protocols for laboratories. In [25] proposes a workflow ontology for the preservation of digital material produced by an organization or a file system. These works focus on building workflow ontologies to represent collaborative work in different areas; however, this paper presents a workflow ontology to develop collaborative applications (see Figure 1) using the knowledge base described in the section 3.

### *A. Workflow ontology*

This ontology —which can be used as a knowledge base— provides a common and well defined vocabulary, a list of terms and meanings, which describe concepts and relationships among them, along with axioms, in a formal way, in order to provide the collaborative applications with a complete and coherent specification for its development (see Table 1). The workflow ontology has been validated using the reasoner Pellet of Protégé. The workflow ontology enables the design and analysis for developing collaborative applications since it supplies a way of classifying, organizing, and representing the group organization; it provides the elements for simplifying and controlling the

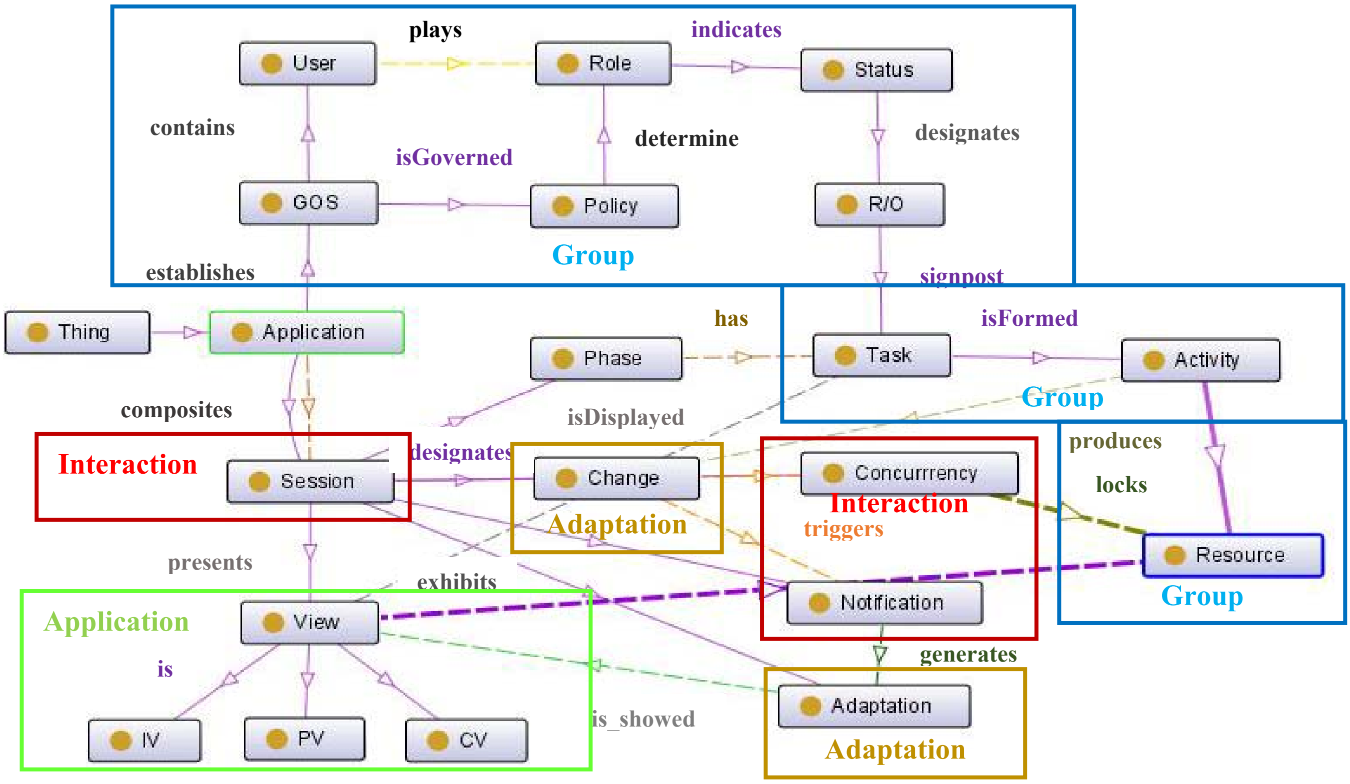


Fig. 1. Workflow Ontology —it is generated with Protégé— to develop groupware

users' interaction; it allows managing and presenting the users' interaction with the application —showing individual and group interaction— it allows adapting the application to the group's necessities or to new collaborative scenarios. In this manner, the ontology conveys a knowledge base that can be assembled by portions. The terms related to the Group aspect, constitute the first part of the knowledge base, allowing the designers to readily establish the components of the organizational structure. So, they know who is doing what, what kind of functions a particular user performs, or what status within the organization a role has. The terms associated to the Interaction aspect, make up the second part of the knowledge base, permitting the designers —from activities specified and resources used— inferring where a notification and concurrency process must be accomplished. The terms linked to the Application aspect, comprise the third part of the knowledge base, leading the designers to determine both the individual and the cooperative view, which must be shown and updated for the application's users. The terms connected to the Adaptation aspect, constitute the fourth part of the knowledge base, letting the designers to determine the changes required to adapt the application.

The workflow ontology specifies four aspects —that contemplate the above mentioned terms— in order to create a collaborative application.

First, the **Group** elements are detailed; thus, the Application´s GOS is described; defining the SMP that rules this GOS, the Roles of this SMP, the Status and the R/O of each Role, along with its Tasks, and for each Task, the Activities that constitute it are established; the Resources used in each Activity are also specified. The View —user interface— elements (such as: label, text box, combo box, button, lists, calendars, files, walls, and messages) are stated as the Resources of each Activity.

Second, the **Interaction** elements are indicated; consequently, the Sessions (Ss) that compose the Application should be determined; the Notification that will be executed should be described —if it were not present, it should be indicated by "Not" ("N"); in like manner, where the Concurrency process is produced —again, if it were not available it should be specified by an "N." To carry out the Notification, depends on the activities executed. The concurrency is activated according to the resources used by the users.

Third, the **Application** elements are designated; consequently, the application Phases and Views are established. The Phases are defined according to the tasks that allow to achieve a common goal; moreover, one or several Phases comprise a Session. The View can be IV, PV, and/or CV, which are made up with Resources used in the Tasks performed by users. Because the IV is always present, a "Yes" ("Y") is permanently exposed; while when the PV and CV are displayed when the Notification and Concurrency happen —in this case "Y" is exhibited, otherwise "N" is visible. The PV enables the group awareness, and CV supports the group memory.

Fourth, the **Adaptation** elements are selected; subsequently, the Changes that produce the Application Adaptation are specified. The Change is activated by the Notification, which indicates the Views —IV, PV, and CV— that should be adjusted to this Change. When an Adaptation (Ad) is carried out it is revealed, otherwise "N" is displayed. The Change is not shown due to the lack of space.

TABLE I
WORKFLOW ONTOLOGY COMPONENTS

| Relation | Domain (Concept) | Range (Concept) | Constrain |
|---|---|---|---|
| establishies inverse: isEstablished | Application | GOS | max cardinality=1 |
| contains inverse: IsContained | GOS | User | min cardinality=2 |
| isGoverned inverse: governs | GOS | Policy | max cardinality=1 |
| determines inverse: isDetermined | Policy | Role | min cardinality=1 |
| indicate inverse: isIndicated | Role | Status | max cardinality=1 |
| designates inverse: isDesignated | Status | R/O | min cardinality=1 |
| signpost inverse: isSignposted | R/O | Task | min cardinality=1 |
| isFormed inverse: formed | Task | Activity | min cardinality=1 |
| uses inverse: isUsed | Activity | Resource | min cardinality=1 |
| has inverse: isHave | Phase | Task | min cardinality=1 |
| isDisplayed inverse: display | Task | View | min cardinality=1 |
| exhibits inverse: isExhibited | View | Resource | min cardinality=1 |
| is | IV/PV/CV | View | min cardinality=1 |
| produces inverse: is Produced | Activity | Change | min cardinality=0 |
| triggers inverse: isTriggered | Change | Notification | min cardinality=0 |
| triggers inverse: isTriggered | Change | Concurrency | min cardinality=0 |
| locks inverse: isLocked | Concurrency | Resource | min cardinality=1 |
| generates inverse: isGenerated | Notification | Adaptation | min cardinality=1 |
| shows inverse: isShowed | View | Adaptation | min cardinality=1 |
| is_part_of | View | Session | min cardinality=1 |
| is_part_of | Phase | Session | min cardinality=1 |
| composite | Application | Session | min cardinality=1 |

*B. Case study*

In this paper, an Academic Virtual Space (AVS) is presented as a case study. AVS provides a shared workspace to simplify the access to the students to the course material (previously loaded by the professor) as well as the interchange of messages among the users for facilitating the feedback of the course's knowledge, thus, strengthening the students' learning. Only the workflow ontology instances (see Table 2) should be defined to create the collaborative application, called AVS. According to this ontology, first the **Group** elements are specified. Therefore, AVS presents two Roles (R): Professor (P) and Student (S). The former has mayor Status —1— that the latter —2. The professor R/O are to access to and authenticate on the AVS, manage the task —upload and download files— write messages, and create groups. The students R/O correspond to professor's first four, besides registering himself in the groups. The professor instances related to the R/O "accessing to AVS" (CA) are described. So, one of his/her tasks is "registering the user"" (RU), which is composed of the following activities: capturing login (cl), password (cp), first name (cfn), last name (cln), Facebook (cf), Twitter (ct), and e-mail (cem); as well as repeating password (rp), uploading picture (up), and sending data (sd). The first eight activities use the next resources: Register User Interface (RUI), Label (L), and Text Box (TB). The ninth activity requires the following resources: RUI, Browse Button (BB), and File (Fi). The tenth activity implicates the Button Register (BR) resource.

Second, the **Interaction** elements are stated (see Table 2). AVS offers one Session (Ss), called Academic Workspace (AW). Nine out of the ten activities of the "registering user" task, do not produce a Notification, so the notification and concurrency columns show an "N", while the tenth activity notifies when a new user has accessed the AVS, then the Concurrency locks the Database (DB) and the View (V); therefore, nobody else can write on both. Third, the **Application** elements are designated (see Table 2). Two Phases are considered, since the tasks focus on two goals: access to (AS) and work in AVS (WS). The first Phase has two Views: Register IU (RIU) and Access IU (AIU). The second Phase has three Views: Task IU (TIU), Messages IU (MIU), and Group IU (GIU). The IV column shows a "Y", while PV and CV display a "Y" whenever a notification is performed. Fourth, the **Adaptation** element is established (see Table 2).

The adaptation (Ad) column is described when a notification has happened. In the case of the Task "Registering the User", in the first nine activities, the Ad column exhibits an N, while on the tenth one; the executed adaptation is to show a new user. The remnants of the workflow ontology instances can be viewed in the Table 2.

When some instances of the workflow ontology are defined, it is possible to generate paths that can be viewed as a semantic network (see Figure 2), which can be made for each row of the Table 2, in such a way; several semantic networks for a collaborative application can be developed. Each semantic network can help to develop a collaborative application in a simple, easy manner. For example; Figure 2 shows that the AVS application is created, when AVS has the "GOS AVS" and it presents the Session "AW". The "GOS AVS" is governed by the "SMP AVS", this SMP defines the professor Role with Status 1 and R/O "CA", which allows performing the Task "RU". This Task is part of Phase "AS" —that comprises the Session "AW"— and it is composed by the Activity "sd", which uses the Resource "BS" and activates the Notification "ANU", therefore, the Concurrency locks the Resources: DB and RIU, until the Adaptation —which is produced by the Change— is executed. The Adaptation modifies the View, in such a way the IV, the PV, and the CV are updated. The View is part of Phase "AS", and this, in turn, of the Application, called AVS. The semantic network so described allows us to understand how the application depends on the GOS and on the Sessions, and how the Phase is related to the Task and View and, in like manner, in what way the Tasks include some activities, and so on. Consequently, this understanding simplifies the development of collaborative applications for the designers.

TABLE II
KNOWLEDGE BASE OF AVS

| R | St | R/O | Task | Activity | Resource | Ss | Notification | Cc | Phase | View | IV | PV | CV | Ad |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P, S | 1, 2 | Accessing to AVS | Registering User | Capturing Login | RUI, L, TB | AW | N | N | AS | RIU | Y | N | N | N |
| | | | | Capturing Password | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Repeat Password | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Capturing First Name | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Capturing Last Name | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Capturing Facebook | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Capturing Twitter | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Capturing e-mail | RUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Uploading Picture | RUI, BB, Fi | | N | N | | | Y | N | N | N |
| | | | | Send Data | RUI, BR | | Arriving New User (ANU) to another | DB & RUI | | | Y | Y | Y | Showing new user (SNU) |
| P, S | 1, 2 | Authenticating on the AVS | Authenticating User | Capturing Login | AUI, L, TB | | N | N | | AIU | Y | N | N | N |
| | | | | Capturing Password | AUI, L, TB | | N | N | | | Y | N | N | N |
| | | | | Send Data | AUI, BR | | User Access (UA) to another | DB & AUI | | | Y | Y | Y | UA |
| P, S | 1, 2 | Managing Task | Managing User Task | Review File | TUI, L, TB, Fi | | File Updating to another | DB & TUI | | TIU | Y | Y | Y | Showing File |
| | | | | Creating Delivery Date | TUI, Calendar | | Date Updating to another | DB & TUI | | | Y | Y | Y | Showing Date |
| | | | | Uploading Task | TUI, BB, Fi | | Task Updating to another | DB & TUI | | | Y | Y | Y | Showing Task |
| | | | | Downloading Task | TUI, BB, Fi | | Downloaded Task to another | DB & TUI | | | Y | Y | Y | Showing Task File |
| P, S | 1, 2 | Writing Message | Sending Message | Writing on the Wall | MUI, TB, BS | | Updating Message to another | DB & MUI | WS | MIU | Y | Y | Y | Showing Messages |
| P | 1 | Generate Group | Creating Group | Selecting Course | GUI, L, CL, TB, CB, BS | | Selected Course to another | DB & GUI | | GIU | Y | Y | Y | Showing Course |
| | | | | Describing Course | GUI, L, TB, BS | | Described Course to another | DB & GUI | | | Y | Y | Y | Showing description |
| | | | | Choosing Hours | GUI, CL, TB, CB, BS | | Chosen Hours to another | DB & GUI | | | Y | Y | Y | Showing Hours |
| | | | | Creating Group Password | GUI, L, TB, BS | | Created Group to another | DB & GUI | | | Y | Y | Y | Showing Group |
| S | 2 | Register in the Group | Enrolling in the Group | Select Teacher, Select Group, and Select Course | EGUI, L, TL, SL, GL, CL, BS | | Selected Teacher to another | DB & EGUI | | | Y | Y | Y | Showing Teacher |

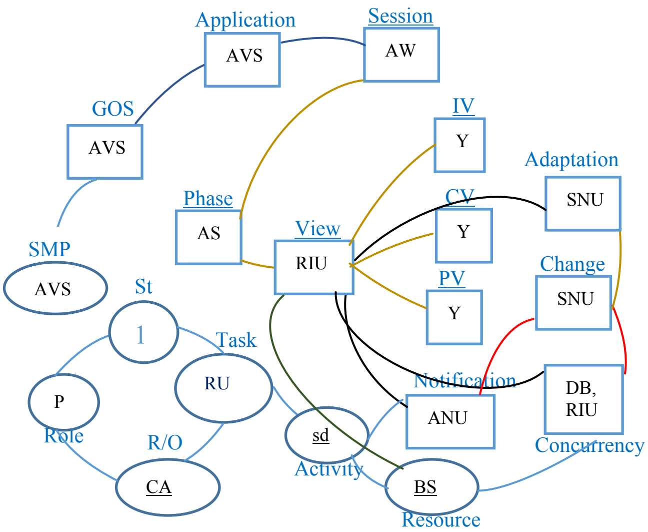


Fig. 2. A semantic network of the collaborative application, named AVS

## V. Conclusions and Future Work

This paper has presented a workflow ontology for developing collaborative applications, which makes it possible to represent the collaborative application to be built. Thus, the designers are provided with a knowledge base to avoid ambiguities, when they use certain terms in the modelling of collaborative applications. The ontology brings a knowledge base that can be assembled by portions, and that can be created by using an instance table and/or a set of semantic networks. Both of them allow simplifying the workflow ontology generation for developing a collaborative application. The future work will aim to specify a methodology to develop collaborative applications, starting with the workflow ontology described in this paper.

## References


[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider editors, *The Description Logic Handbook: Theory, Implementation and Applications.* Cambridge University Press, 2003.

[2] A. Jovic, M. Prcela, and D. Gamberger, "Ontologies in Medical Knowledge Representation," *in Proceedings of the 29th Int. Conf. on Inform. Technology Interfaces*, 2007, pp. 25-28.

[3] R. Stevens, C.A. Goble, and S. Bechhofer, "Ontology-based knowledge representation for bioinformatics," *Briefings in Bioinformatics*, vol. 1-4, 2000, pp. 398-414.

[4] A.I. Molina, M.A. Redondo, M. Ortega, and U. Hope, "CIAM. A methodology for the development of groupware user interfaces," *Journal of Universal Computer Science,* 2007.

[5] V. Penichet, M. Lozano, and J. Gallud. *An Ontology to Model Collaborative Organizational Structures in CSCW.* In Engineering the User Interface, Springer, 2008, pp.127-139.

[6] R. Gruber, "A translation approach to portable ontology specification," *Knowledge Acquisition,* vol. 5, 1993, pp. 199-220.

[7] I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen, "From SHIQ and RDF to OWL. The making of a web ontology language," *J. of Web Semantics*, vol. 1-1, 2003, pp. 7-26.

[8] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "Ontology-Based Modelling of Session Management Policies for Groupware Applications," *Lecture Notes in Computer Science*, vol. 4739, Springer, Heidelberg, 2007, pp. 57–64.

[9] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "SAMCA: Service-based architectural model for collaborative applications," *in Proceedings of 15TH International Conference on Software Engineering and Applications*, 2011.

[10] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "Methontology-based ontology representing a service-based architectural model for collaborative applications," *Advances in Soft Computing Algorithms. RCS,* vol. 54, 2011, pp. 77-90.

[11] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "Service-based access control using stages for collaborative systems," *Advances in Computer Science and Engineering. Research in Computing Science*, vol. 42, 2009, pp. 311-322.

[12] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "Security and adaptability to groupware applications using a set of SOA-based services," *Advances in Computer Science and Engineering. Research in Computing Science*, vol. 45, 2010, pp. 279-290.

[13] M. Roseman, and S. Greenberg, "Building Real-time Groupware with GroupKit, a Groupware ToolKit," *ACM Trans. Computer-Human-Interaction*, vol. 3, 1996, pp. 66-106.

[14] P. García, and A. Gómez, "ANTS Framework for Cooperative Work Environments," *IEEE Computer Society Press,* vol. 36-3, 2003, pp. 56-62.

[15] B. Fonseca, and E. Carrapatoso, "SAGA: A Web Services Architecture for Groupware Applications," *Springer-Verlag, LNCS,* vol. 4154, 2006, pp. 246-261.

[16] T.C.N. Graham, and T. Urnes, "Integrating Support for Temporal Media in to an Architecture for Graphical User Interfaces," *ACM Press,* 1997, pp. 172-182.

[17] Y. Laurillau, and L. Nigay, "Clover Architecture for Groupware," *in Proceedings of the ACM Conference on CSCW*, 2002, pp. 236-245.

[18] J.L. Garrido, M. Gea, N. Padilla, J.J. Canas, and Y. Waern, "AMENITIES: Modelo de entornos cooperativos," *in I. Aedo, P. Díaz & C. Fernández (eds.), Actas del III Congreso Internacional Interacción Persona-Ordenador*, 2002, pp. 97-104.

[19] L. Fischer, *Workflow Handbook. Future Strategeis Inc*., Lighthouse Point, FL, 2004.

[20] R. Gacitua, M. Arguello-Casteleiro, P. Sawyer, J. Des, R. Perez, M.J. Fernandez-Prieto, H. Paniagua, "A collaborative workflow for building ontologies: A case study in the biomedical field," *Research Challenges in Information Science*, 2009, pp.121-128.

[21] K. Kozaki, E. Sunagawa, Y. Kitamura, and R.A. Mizoguchi, "Framework for Cooperative Ontology Construction Based on Dependency Management of Modules," *CEUR Workshop Proceedings*, vol. 292, CEUR-WS.org, 2007, pp. 33-44.

[22] A. Sebastian, N.F. Noy, T. Tudorache, M.A. Musen, "A Generic Ontology for Collaborative Ontology-Development Workflows," *in Proceedings of the 16th international conference on Knowledge Engineering: Practice and Patterns,* 2008.

[23] A. Maccagnan, M. Riva, E. Feltrin, B. Simionati, T. Vardanega, G. Valle, N. Cannata, "Combining ontologies and workflows to design formal protocols for biological laboratories," *Automated Experimentation,* vol. 2-3, 2010.

[24] M. Mikelakis, C. Papatheodorou, "An ontology-based model for preservation workflows," *in Proceedings of the 9th International Conference on Digital Preservation,* 2012.