Knowledge Representation for the Development of Collaborative Applications

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

[1] Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, Avenida San Claudio y 14 Sur, Ciudad Universitaria. 72570 Puebla, México

[2] Departamento de Lenguajes y Sistemas Informáticos, E.T.S.I. Informática y de Telecomunicación, Universidad de Granada, C/ Periodista Saucedo Aranda, s/n, 18071 Granada, Spain.

Abstract. The workflow to develop collaborative applications should be highly adaptable to frequent organizational changes. In order to increase the adaptability of workflow, an ontology-based workflow model is proposed. This model represents the set of steps —along with their order of execution— performed by different entities for developing this kind of applications. Ontology is one of the strategies for the structured representation of a chosen knowledge domain in a formal way, helping to remove ambiguity and redundancy, detecting errors, and allowing automated reasoning. In this work, the ontologies are considered as models to represent knowledge. A case study is presented in order to show the use of Knowledge-based Workflow Model for Collaborative Applications.

Palabras Clave: Ontology, Collaborative Application, Workflow Model, Knowledge Representation, Knowledge-based Workflow Model.

1 Introduction

Ontology provides an ideal solution, for it represents the domain knowledge using description logic symbols, which allows specifying it in a simple, readable way for both humans and machines; as well as performing a much deeper reasoning by means of machines. It facilitates a knowledge base to provide semantic, common understanding, communication, and shared knowledge on the domain of interest, and a knowledge reasoning carrying out an inference process to reach to conclusions on such a base, by means of a reasoner, inference rules, and query languages.

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 in the development of such applications so, it proposes an ontology-based workflow model named workflow ontology to develop collaborative applications. This workflow ontology allows representing all the necessary symbols in order to specify the elements for building this type of applications. These symbols make up a knowledge base, which allows to reason and draw conclusions; to generate new knowledge in collaborative

domain, using reasoner, inference rules and query languages. 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 model for collaborative applications and a case study focused on academic virtual space; Section 6, outlines the conclusions and future work.

2 Ontology-based knowledge

In recent years, the use of ontologies has extended in diverse areas such as medicine [1]; bioinformatics [2]; groupware [3, 4]; 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 in 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. The domain knowledge, describes the main static information and the objects of knowledge [5]. According to Gruber [6], domain knowledge in ontologies can be formalized using four kind of components: concepts, relations, axioms and instances.

The OWL representation [7] facilities are directly based on Description Logics [8]. This basis confers upon OWL a logical framework, including syntax and model-theoretical semantics, allowing a knowledge representation language capable of supporting a knowledge base, and a practical, effective reasoning. Protégé is used in order to develop ontologies with OWL to provide graphical interfaces that facilitate the knowledge representation and reasoning. Protégé is an engineering tool open source ontology and a knowledge-based framework, which is widely used due to its scalability and extensibility with lots of plugins; to facilitate inference knowledge through reasoners, query languages, and rules. Therefore, it can be concluded that ontologies are an ideal solution for knowledge representation and reasoning, since it provides a set of symbols through a formal and structured vocabulary.

3 Knowledge in collaborative applications

A collaborative application focuses on supporting group work to achieve a common goal by providing a shared workspace. For this reason, the knowledge is the result of a group interaction and its adaptation on application, i.e. depends on:

- **Group**, which carries out a set of tasks to achieve a common goal through a shared environment. Consequently, the group should have a suitable **Group Organizational Structure** (**GOS**). This is ruled by the **Session Management Policy —SMP—** that establishes the same [9]. The SMP defines a hierarchical or not-hierarchical GOS by means of **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 carried out to accomplish a common goal. The **Task** is composed of **Activities,** which use the prevailing shared **Resources** [10]. The **GOS** specification provides the Knowledge about how the interaction among users is carried out; what **Roles** are involved in the group; what **Task** is performed by each **Role,** and what **Resources** are used to achieve a goal. This Knowledge facilitates the basic elements of a collaborative application.

- **Interaction**, requires: A **session,** shared workspace where a group will interact**.** A **notification** process that 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 of the group are performed, and where the information of the shared resources is stored; thus, creating a *group memory* to provide understanding and reasoning about the collaborative process. A **concurrency** process that ensures the consistency of the data being shared; providing collaborating users with dynamically-generated temporary access and permissions, to reduce racing conditions, and to guarantee mutually exclusive resource usage. These permissions depend on the GOS established and on the lock mechanism. So, the interaction knowledge is supplied by concurrency, *awareness, and memory group,* in such a way that the common context helps avoiding surprises by reducing the probability of conflicts in the group established [11].

- **Application**, presents the **Views** that are user interfaces allowing 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.* The application is made up into **phases** [12], where each **phase** is defined as a global description of the tasks that are active. Therefore, the knowledge is supplied via the IV, PV, and CV, allowing to show the interaction between users and the application; as well as by means of phases which constrain the users who can participate in accordance with the roles they are playing.

- **Adaptation** [13]; adjusts the Views in accordance to changes triggered by *notification*, in such a manner that the application shows the most recent updates although 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 means of a Voting Tool, which offers several kinds of agreements such as the agreement based on the majority vote, 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 the 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, users are notified that this adaptation process cannot be achieved. Knowledge is related to modifying one or several application components, in order to change exclusively that part of the application which did not fit the features of the new scenario.

The aforementioned is derived from the architectural model (see Figure 1) presented in [10] that is the result of the study and review of several 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]). Where the different concepts and terms above mentioned (such as group, role, task, activity, resource, session, notification, concurrency, shared user interface, phase, adaptation, etc.), have been applied to design and develop collaborative applications. All these concepts and terms will be used as symbols, which are part of workflow ontology for the development of collaborative applications.

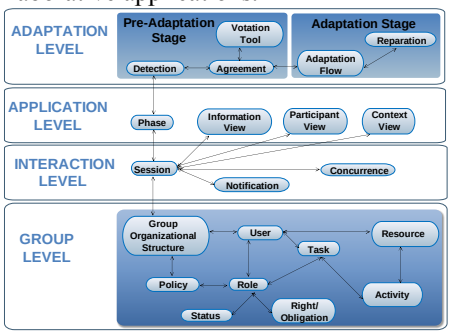

Figure 1. Layered architectural model for building groupware

4 A workflow model for collaborative applications

The development of collaborative applications needs to perform 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 knowledge domain through 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, as the workflow can be built on the ontology. The former defines how the tasks or operations should be used and combined, and once that it has been represented, it may be considered a static structure. The latter can symbolize this structure due to its expressive power. So the ontology is an ideal solution for the workflow representation. There are several examples of workflow ontologies: it presents a collaborative workflow for terminology extraction and collaborative modeling of formal ontologies using two tools Protege and OntoLancs [21]; it allows the development of cooperative and distributed ontologies, based on dependencies management between ontologies modules [22]; it shows an ontology-based workflows for ontology collaborative development in Protégé [23],it presents the combination of workflows with ontologies to design way formal protocols for laboratories [24], it proposes a workflow ontology for the preservation of digital material produced by an organization or a file system [25]. All these works are focused on building workflow ontologies to represent

collaborative work in different areas, however, this paper presents a workflow ontology to develop collaborative application (see Figure 2) using the architectural model elements as the ontology vocabulary.

The Table 1 shows the concepts, relations and axioms of the Workflow Ontology, which establishes; **first**, the Application is described; **second**, the SMP name is stated; **third**, the GOS together with all the elements that contain it are defined; **fourth**, the Tasks that are part of the Phase and the View are specified; **fifth**, the Resources that will be presented on the View; **sixth**, the IV, PV, and CV that will have the View; **seventh**, the Change produced by Activity; **eighth**, the Notification and Concurrency (Cc) triggered by Change; **ninth**, the Resource locked by means of the Cc; **tenth**, the Adaptation (Ad) generated through Notification; **eleventh**, the Adaptation is showed on View (IV, PV, and CV); **twelfth**, it outlines both Views and Phases that are part of the Session, and **finally** the Sessions that comprise the Application.

Table 1. 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

A case study is an Academic Virtual Space (AVS) which provides the students with a shared workspace to simplify their access to the course material previously loaded by the professor. The AVS includes two roles: 1) The professor can register himself, create groups, upload and download files both his and those of the students, publish and respond to those made by others; and 2) The students, can register themselves in the shared space and access courses, load and unload files (homework and course materials that have been uploaded by the teacher), make publications and to reply to them. Thus, in order to create this workspace only the workflow ontology instances should be defined (see Table 2). The instances are defined in according to the established steps in the workflow ontology. These constitute the knowledge base for developing collaborative applications. Therefore, **first**, the described Application is AVS; **second**, the stated SMP is SMP-AVS; **third**, the defined GOS is GOS-AVS and the elements

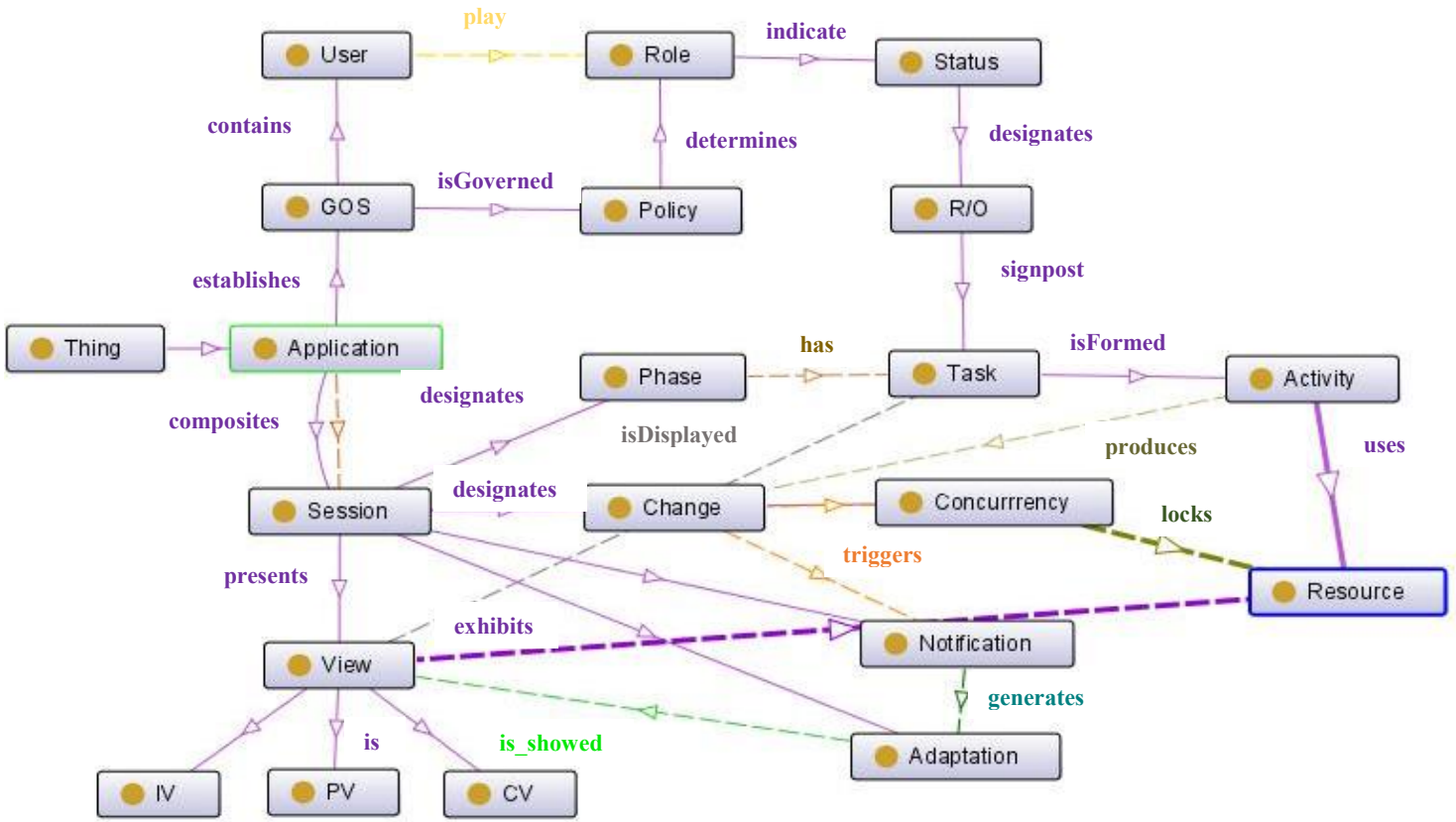

Figure 2. Workflow Ontology to develop groupware.

that contain the GOS-AVS are showed in the Table 2; the **fourth**, two phases are specified along with their tasks that are part of the Phase, as well as the Views are designated; **fifth**, the Resources on each View are presented; **sixth**, the IV, PV, and CV that will have the View are showed; **seventh**, the produced Change by Activity are revealed in the Notification Colum of the Table 2; **eighth**, the triggered Cc by Change is indicated by Yes in the Cc column (Y); **ninth**, the Cc column presents Y, when a Resource is locked; **tenth**, the generated Adaptation is displayed in the Ad column; **eleventh**, the showed Adaptation on View is specified with Y in IV, PV, and CV; **twelfth**, the Views and Phases of the Session are exhibited, and **finally** the Sessions that comprise the Application. When Table 2 is constructed, and the GOS elements are already defined, it is possible infer, for example: The tasks of each phase and view; the presented resources in each view, the produced changes by each activity, the activated concurrency by change, etc. This inference is very important when a collaborative application has been developed. Furthermore, the AVS knowledge base can be used in another applications, which focus in academic environment and requires two roles (Teacher and Student). In this a manner, it will be possible to reduce time and costs in the development of collaborative applications.

5 Conclusions and Future Work

This paper has presented a workflow model for developing collaborative applications using the inherent knowledge of the integrated elements —group, interaction, application and adaptation— necessary to build this applications. These elements are the ontology concepts, and present a set of constrains, that guide and facilitate the development of this applications. Developing a formal and structured knowledge base, which allows building any collaborative application, defining its instances on ontology. Furthermore, in order to simplify and facilitate an ontology description, a specification table is proposed, in such a manner that any individual may be able to develop a collaborative application by means on this table.

The future work will aim to specify a methodology to develop collaborative application, starting with the workflow ontology described in this article.

References

1. Jovic, A., Prcela, M, and Gamberger, D. Ontologies in Medical Knowledge Representation. In Proceedings of the 29th Int. Conf. on Information Technology Interfaces, pp. 25-28, (2007)
2. Stevens, R., Goble, C.A., and Bechhofer, S. Ontology-based knowledge representation for bioinformatics. Briefings in Bioinformatics, Vol. 1-4, pp. 398-414, (2000)
3. Molina, A.I., Redondo, M.A., Ortega, M., and Hope, U. ClAM. A methodology for the development of groupware user interfaces. Journal of Universal Computer Science (2007)
4. Penichet, V., Lozano, M., and Gallud. J. An Ontology to Model Collaborative Organizational Structures in CSCW. In Engineering the User Interface, Springer, pp.127-139, (2008)
5. Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R., Shadbolt, N., Van de Velde, W., Wielinga, B. Knowledge engineering and management. The KADS Methodology. (1999)

Tabla 2. Knowledge Base of AVS.

Rol	St	R/O	Task	Activity	Resource	Se_ ssion	View (V)	Phase	Notification	Cc	IV	PV	CV	Ad
Profe _ssor (P), Stu_ dent (S)	1, 2	Accessing the Application	Register User	Capturing Login	Label (L), Text Box (TB)	AVSS	Regis_ tering	Acce_ ssing	NOT (N)	N	YES (Y)	N	N	N
				Capturing Password	L, TB				N	N	Y	N	N	N
				Repeat Password	L, TB				N	N	Y	N	N	N
				Capturing First Name	L, TB				N	N	Y	N	N	N
				Capturing Last Name	L, TB				N	N	Y	N	N	N
				Capturing Facebook	L, TB				N	N	Y	N	N	N
				Capturing Twitter	L, TB				N	N	Y	N	N	N
				Capturing e-mail	L, TB				N	N	Y	N	N	N
				Uploading Picture	Browse Button(BB), File(Fi)				N	N	Y	N	N	N
				Send Data	Register Button (RB)				Arriving New User to another	DB & V	Y	Y	Y	TL, SL
P, S	1, 2	Authentica_ ting on the Application	Authentica _ting User	Capturing Login	L, TB		Identi_ fication		N	N	Y	N	N	N
				Capturing Password	L, TB				N	N	Y	N	N	N
				Send Data	RB				User Access (UA) to another	DB & V	Y	Y	Y	UA
P, S	1, 2	Managing Task	Managing User Task	Review File	L, TB, Fi		Task	Shared Work_ space	File Updating to another	DB & V	Y	Y	Y	Fi List
				Creating Delivery Date	Calendar				Date Updating to another	DB & V	Y	Y	Y	Date
				Uploading Task	BB, Fi				Task Updating to another	DB & V	Y	Y	Y	Task
				Downloading Task	BB, Fi				Downloaded Task to another	DB & V	Y	Y	Y	Task Fi
P, S	1, 2	Writing Comment	Sending Comment	Commenting on the Wall	TB, Button Send (BS)		Group		Updating Message to another	DB & V	Y	Y	Y	Messages
P	1	Generate Group	Creating Group	Selecting Course	L, Courses List(CL), TB, Combo Box (CB), BS				Selected Course to another	DB & V	Y	Y	Y	CL
				Describing Course	L. TB, BS				Described Course to another	DB & V	Y	Y	Y	Course
				Choosing Hours	CL, TB, CB, BS				Chosen Hours to another	DB & V	Y	Y	Y	Date
				Creating Group Password	L, TB, BS				Created Group to another	DB & V	Y	Y	Y	GL
S	2	Register in Group	Enroll in the Group	Select Teacher, Select Group, and Select Course	L, Teachers List(TL), Student List(SL), Groups List(GL), CL, BS				Selected Teacher to another	DB & V	Y	Y	Y	TL
									Selected Group to another	DB & V	Y	Y	Y	It shows G
									Enter of User to the Group	DB & V	Y	Y	Y	Group

6. Gruber, R. A translation approach to portable ontology specification. Knowledge Acquisition. Vol. 5, pp. 199-220, (1993)
7. Horrocks, I., Patel-Schneider, P.F., and van Harmelen, F. From SHIQ and RDF to OWL: The making of a web ontology language. J. of Web Semantics, Vol. 1-1, pp. 7-26, (2003)
8. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F. editors. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, (2003)
9. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., and Paderewski-Rodríguez, P. Ontology-Based Modelling of Session Management Policies for Groupware Applications. Lecture Notes in Computer Science, Vol. 4739, pp. 57–64, Springer, Heidelberg, (2007)
10. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., and Paderewski-Rodríguez, P. SAMCA: Service-based architectural model for collaborative applications. In Proc. 15TH International Conference on Software Engineering and Applications (2011)
11. Anzures-García M., Sánchez-Gálvez L.A., Hornos M.J. & Paderewski, P. Methontology-based ontology representing a service-based architectural model for collaborative applications. Advances in Soft Computing Algorithms. RCS, Vol. 54, pp. 77-90, (2011)
12. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., and Paderewski-Rodríguez, P. Service-based access control using stages for collaborative systems. Advances in Computer Science and Engineering. Research in Computing Science, Vol. 42, pp. 311-322, (2009)
13. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., and Paderewski, P. Security and adaptability to groupware applications using a set of SOA-based services. Advances in Computer Science and Engineering. RCS, Vol. 45, pp. 279-290, (2010)
14. Roseman, M., and Greenberg, S. Building Real-time Groupware with GroupKit, a Groupware ToolKit. ACM Trans. Computer-Human-Interaction, Vol. 3, 66-106, (1996).
15. García, P., and Gómez, A. ANTS Framework for Cooperative Work Environments. IEEE Computer Society Press, Vol. 36, 3 Los Alamitos, CA, USA, pp. 56-62, (2003).
16. Fonseca, B., and Carrapatoso, E. SAGA: A Web Services Architecture for Groupware Applications, In Proc. of the CRIWG, LNCS 4154, Springer-Verlag, pp. 246-261, (2006).
17. Graham, T.C.N., and Urnes, T. Integrating Support for Temporal Media in to an Architecture for Graphical User Interfaces. Proc. of the International Conference on Software Engineering (ICSE'97), ACM Press, Boston, USA, pp. 172-182, (1997).
18. Laurillau, Y., and Nigay, L. Clover Architecture for Groupware. Proc. of the ACM Conference on CSCW. New Orleans, Louisiana, USA, pp. 236-245, (2002)
19. Garrido, J. L., Gea, M., Padilla, N., Canas, J.J., and Waern, Y. AMENITIES: Modelo de entornos cooperativos. In I. Aedo, P. Díaz & C. Fernández (eds.), Actas del III Congreso Internacional Interacción Persona-Ordenador, pp. 97-104, (2002)
20. Fischer, L. Workflow Handbook. Future Strategeis Inc., Lighthouse Point, FL (2004)
21. 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 Scienc*. pp.121-128 (2009)
22. Kozaki, K., Sunagawa, E., Kitamura, Y. and Mizoguchi, R. A Framework for Cooperative Ontology Construction Based on Dependency Management of Modules. Vol. 292 of CEUR Workshop Proceedings, pp. 33-44. CEUR-WS.org,(2007)
23. Sebastian, A., Noy, N.F., Tudorache, T., Musen, M.A. A Generic Ontology for Collaborative Ontology-Development Workflows. Proceedings of the 16th international conference on Knowledge Engineering: Practice and Patterns, (2008)
24. 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, Vol. 2-3, (2010)
25. Mikelakis, M., Papatheodorou, C. An ontology-based model for preservation workflows. In Proceedings of the 9th International Conference on Digital Preservation, (2012)