A Semantic Approach to Develop Groupware

Anzures-García Mario[1], Sánchez-Gálvez Luz[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
{mario.anzures, sanchez.galvez} @correo.buap.mx

[2] Departamento de Lenguajes y Sistemas Informáticos, ETS Ingeniería Informática, Universidad de Granada, C/ Periodista Saucedo Aranda, s/n, 18071, Granada, Spain
{patricia, mhornos}@ugr.es

Abstract. This paper presents an architectural model-based semantic approach to develop groupware. This approach facilitates a structured and formal representation of a knowledge domain, particularly, the groupware domain. The representation is carried out by using a workflow ontology, which simplifies the development of groupware and its adaptation. The knowledge base of the workflow ontology is comprised for the architectural model elements. The architectural model is provided in order to support the static structure and dynamic nature of this type of applications. In such a way, this model may be able to supply the group organizational structure that supports the group dynamic behavior and, its interaction; as well as, the manner how this behavior will be displayed on the application. Finally, a case study to show the semantic approach usefulness is presented.

Palabras Clave: Semantic Approach, Groupware, Architectural Model, Workflow Ontology, Ontology.

1 Introduction

Nowadays, many organizations require structures that facilitate an efficient communication, collaboration and coordination between the members of the group with independence of their geographical location. These facts have increased the interest in collaborative applications, which generally are known as groupware. This term is defined as: a computer-based system that supports groups of people engaged in a common task (or goal) and provides an interface to a shared environment [1]. The development of groupware is a complex process, since these applications have to consider activities and protocols social of the organizations; shared workspace; interaction among users and between them with shared resources; group awareness; group memory; shared and individual view; and adaptation. So groupware can be adapted to different scenarios using a structured and formal representation of a knowledge domain.

Several tools or frameworks (such as Groupkit [2], ANTS [3], and SAGA [4]), architectures (e.g., Clock [5], and Clover [6]), and methodologies (AMENITIES -A

MEthodology for aNalysis and desIgn of cooperaTIve systEmS [7], ClAM (Collaborative Interactive Applications Methodology) [8], y TOUCHE [9].) have been developed in order to carry out construction processes of groupware. However, these proposals lack of the necessary adaptability to conform naturally to the inherent dynamics or the different scenarios that groupware often present. And also, these show the absence of a formal and structured representation to help eliminate ambiguity and redundancy in the development of groupware. Although, TOUCHE, also, proposes an ontology to CSCW Systems, this focuses on collaborative organizational structures, which corresponds to part of the work presented here; since it also adds elements to control the interaction, application itself and adaptation. The TOUCHE ontology is semi-informal. Moreover, our group organizational structures includes the terms of Rights/Obligations and status that are important to determine the tasks that a role can play in the organization.

In this paper, a formal ontology to develop groupware is proposed. The ontology is based on architectural model that supplies the vocabulary terms that are used as knowledge base for the development of groupware. The ontology is one of the strategies to provide a semantic approach to domain knowledge, since establish a formal vocabulary of terms, and some specification of their meaning; including definitions and of how concepts are related.

The rest of the paper is organized as follows. Section 2 describes briefly the architectural model used as knowledge base. Section 3 introduces to ontology. Section 4 explains semantic approach to develop groupware. Section 5 details a case study. Finally, Section 6 outlines the conclusions.

2 Architectural model

As collaborative applications become more and more complex, a critical aspect of their design is the overall structure of the application, and the ways in which that structure provides conceptual integrity for the same. This level of system design has come to be known as software architecture [10] defined as the fundamental organization of a system, embodied in its components, their relationships to each other, to the environment, and the principles governing its design and evolution.

The Architectures are described with informal (by diagrams and words), or formals (by architecture description languages –ADL), representations. In addition, standard general-purpose modeling notations, such as UML, Z [11], or B [12], have been used. There are different architectural styles [10], this paper focuses a layered style, where the architecture components are grouped in layers. A layer is a software technique for structuring the software architecture that can be used to reflect different abstraction levels in the architecture. The architectural model was developed through an extended literature review, which included studies and surveys from multiple venues, such as journals, conferences, and workshops. This model presents a layered style, which contains four layers (see Figure 1).

The Group Layer, facilities the elements that allow the interaction on the application. This layer supplies the **Group Organizational Structure** (GOS) that *is governed by* the **Session Management Policy** (SMP), which determines how the users are organized. The SMP *defines* the **Roles** that **Users** can *play*. These Roles *establish* the set of **Rights/Obligations** and **Status** within the GOS, and **Tasks** that can be *carried out*. Each Task *is constituted* of **Activities** that *use* the prevailing shared **Resources**.

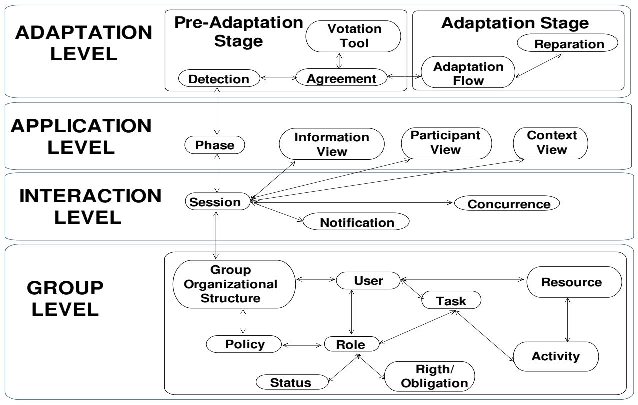


Figure 1. Layered architectural model for building groupware

The Interaction Layer, offers the elements to carry out the interaction among users and between them and application. So, it presents: **Session**, which establishes a shared workspace where a group will collaborate; **Notification**, informs to the users of every occurred change in the session, enabling the group awareness and group memory; and **Concurrency** ensures the consistency of the resources being shared, facilitating the manipulation of the users permissions, which are granted, in accordance with GOS and/or a lock mechanism established.

The Application Layer, allows controlling interaction by means of **phase** [13]—which in a coordination model is defined as each of the collaboration moments— and showing interaction through **views**. The views are user interfaces and there are three: **Information View** —IV, which displays the user information—; **Participant View** —PV, representing the shared view, displaying the changes in shared resources and therefore provide group awareness—; and **Context View** —CV, which displays the change history of shared resources thus providing group memory—.

The Adaptation Layer, facilitates application adaptation and presents two Stages [14] —Pre-Adaptation and Adaptation—. The former determines whether an Activity can adapt the application. For this, it contains a detection process that monitors the Activities in the Session; an agreement process —it is executed only if it is an adaptable process in a non-hierarchical organizational style— to reach a consensus on whether an adaptation process should be performed; and vote tool that allows several kinds of agreement, such as it based on the majority vote, the one based on a maximum or minimum value, etc. The latter performances the adaptation thus it presents the adaptation flow process and the one reparation, which returns each component to its previous state and notifies to users that adaptation cannot take place.

The elements of the first three layers constitute the knowledge base to develop groupware, along with the terms of change, adaptation, and application. These three terms are added, due to that a change in the resources requires to send notification, producing the views adaptation; so, the Application presents the latest changes.

3 Ontology

In recent years, the use of ontologies has extended in diverse areas as medicine [15], bioinformatics [16], groupware [9]; 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. According to Gruber [17], domain knowledge in ontologies can be formalized using four kind of components: concepts, relations, axioms, and instances.

Ontologies can be highly informal, if they are expressed in natural language; semi-informal, if they are expressed in a restricted and structured form of natural language; formal, if they are expressed in an artificial and formally defined language (i.e. Ontolingua, OWL -Web Ontology Language); and rigorously formal, if they meticulously provide terms defined with formal semantics, theorems and proofs of properties, such as soundness and completeness. Ontologies require a logical and formal language to be expressed. The OWL representation facilities are directly based on Description Logics. 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. The development of ontologies is a laborious and error-prone task, especially if done manually, so it is necessary to have tools that can automate some of this work and hide the features and formalisms for ontology specification languages. These tools provide graphical interfaces that facilitate the knowledge representation and reasoning. This paper uses Protégé, which 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.

4 Semantic approach

In this work, a semantic approach is proposed, the approach is based on the workflow ontology. On the one hand, the development of groupware needs to carry out a set of steps ordered, which can be made by means of a workflow. The term workflow typically refers to "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" [18]. 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, for example, it presents a collaborative workflow for terminology extraction and collaborative modeling of formal ontologies using two tools Protege and OntoLancs [19]; it allows the development of cooperatives and distributed ontologies. based on dependencies management between ontologies modules [20]; it shows an ontology-based workflows for ontology collaborative development in Protégé [21], it presents the combination of workflows with ontologies to design way formal protocols for laboratories [22], it proposes a workflow ontology for the preservation digital material produced by an organization or a file system [23]. All these works focused on building workflow ontologies to represent collaborative work in different areas, however, this paper presents a workflow ontology to develop groupware using the architectural model elements as the ontology vocabulary.

The Table 1 shows the concepts, relations and axioms of the Workflow Ontology, which stablishes that: first, the **Application GOS** should be established; second, the **SMP** should be determined, it will govern the GOS; third, the **Roles** should be defined, these are configured **SMP,** and played by one or more users; fourth, the **Users** should be indicated, they are part of GOS; fifth, the **Right/Obligation** (**R/O**) and the **Status** should be designated, the former points out that can do a role, and the latter is role position in the GOS; sixth, the **Tasks** should be mentioned, which are carried out by role, showed in **View**, and part of **Phase**; seventh, the **Activities** should be defined, these are part of a **Task**, and can generate changes**;** eighth, ninth, the **Resources** should be established, these are used in activities, presented on the View, and locked through the Concurrency; the **Notification** should be specified, it is triggered by a change, reflected in the View, and it, also, activate the Adaptation; tenth, the **Concurrency** should be defined, it is triggered by a **change, and** it lock resources to guarantee mutually exclusive usage of same; eleventh, the **IV, PV**, and **CV** should be determined, the three compose a View; twelfth, the **Adaptation** should be showed, it is produced by Notification and displayed on the View; finally, the **Sessions** should be indicated, these are part of an Application, and are composed by **Views** and **Phases**.

5 Case study

This paper presents an Academic Virtual Space (AVS, EVA in Spanish) as case study, which provides the students with a shared workspace to simplify their access to the course material previously loaded by the professor. So, the AVS allows to

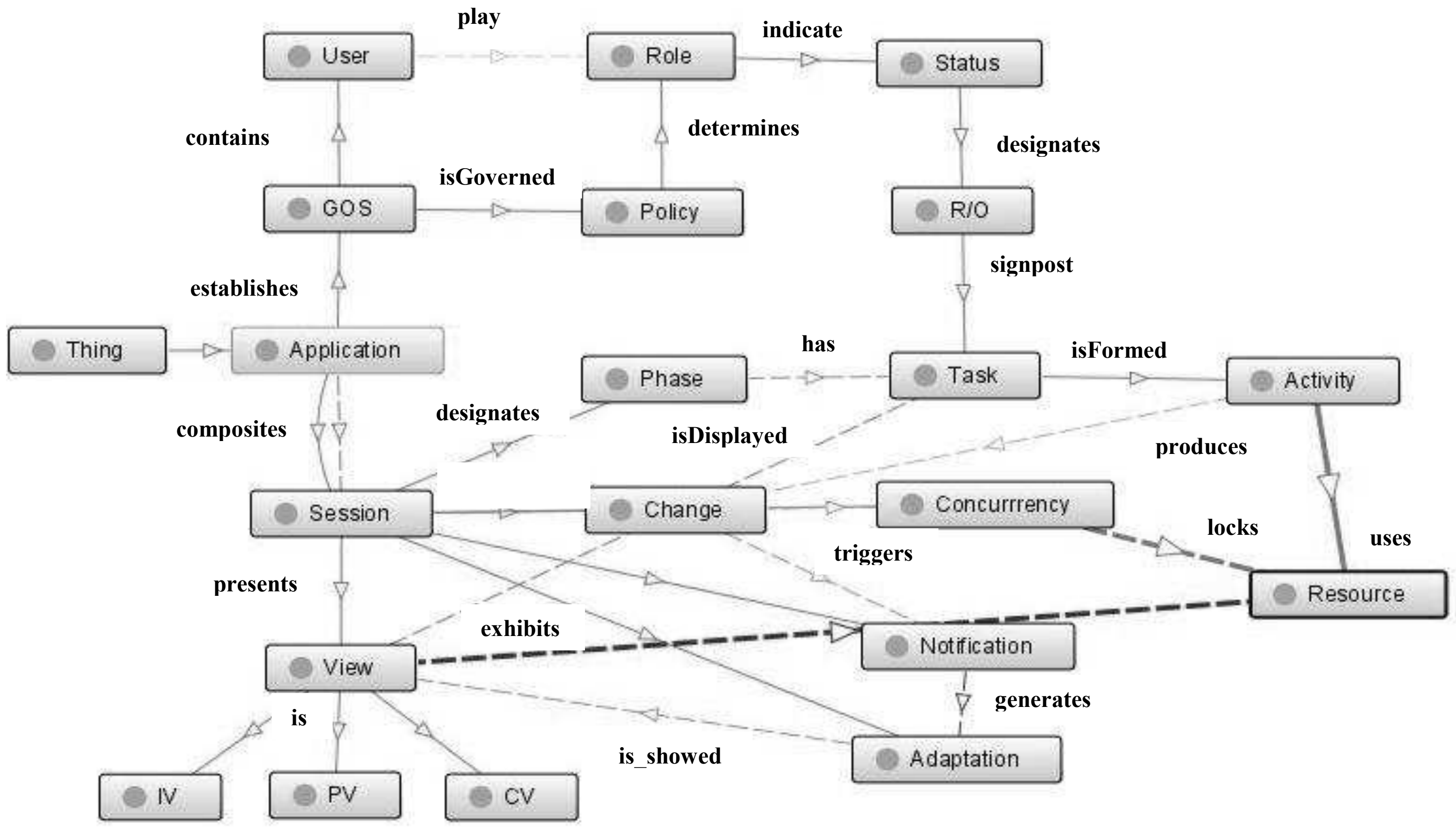


Figure 2. Workflow Ontology to develop groupware.

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	View	min cardinality=1
is	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

generate groups, where students and professors can share information, messages and files. According to workflow ontology carry out:

1. *GOS:* AVS-GOS.
2. *SMP:* AVS-P.
3. *Role:* It specifies two roles: Professor and Students.
4. Users: Professor Mario Anzures-García and Student Coral, Jesus, Ivan.
5. Professor with *status* 1. Student with *status* 2. The Professor and Student have the following RO in common: *accessing, and authenticating* the *Application, managing tasks and writing messages.* Furthermore, the Professor has the *generating group* R/O, and the Student of *Register in the Group.*
6. The Task are to *register user, to authenticate user, to manage user task, to send message, to generate group, and to enroll in the group. The two first Tasks compose the Access Phase, while the others comprise the Collaboration Phase.*

7, 8. Now, the Activities for each Task along with used Resources in they are defined:

   a) To *register user composites of the Activities* (see Figure 3):
      1. *Capturing login* using as Resources the Label, and Text Box

2. *Capturing password* using as Resources the Label, and Text Box
3. *Repeat password* using as Resources the Label, and Text Box.
4. *Capturing first name* with Resources the Label, and Text Box.
5. *Capturing last name* using as Resources the Label, and Text Box
6. *Capturing facebook* using as Resources the Label, and Text Box
7. *Capturing twitter* using as Resources the Label, and Text Box
8. *Capturing e-mail* using as Resources the Label, and Text Box.
9. *Uploading* Picture with Resources the Browse Button and File.
10. *Sending Data* using as Resources *the Register button.*

b) *To authenticate* user *composites of the Activities* (see Figure 3)*:*
1. *Capturing login* using as Resources the Label, and Text Box
2. *Capturing password* using as Resources the Label, and Text Box
3. *Sending Data* using as Resources *the button send.*

c) *To manage user task composites of the Activities:*
1. *It review file* using as Resources *the Label, Text Box, and file.*
2. *Creating Delivery Date* using as Resource the Calendar.
3. *Uploading Task* with Resources *the Label, Text Box, and file.*
4. *Downloading Task* with Resources *the* Browse Button, *and file.*

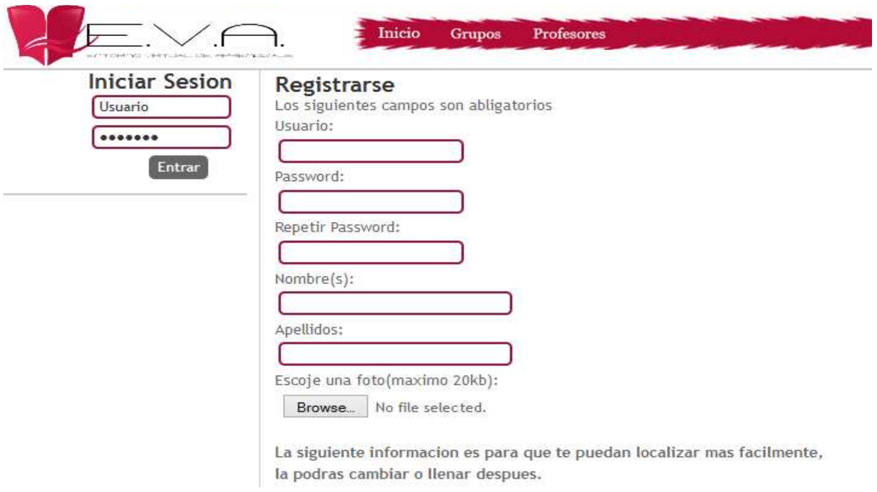
d) *To send message composites of the Activities:*
1. *Comment on the Wall* with Resources *Label, Text Box, and Button Publish.*

e) *To generate group composites of the Activities:*
1. *Selecting Course* using as Resources *the Label, Text Box, Combo Box, and Button Send.*
2. *Describing Course* with Resources *the Label, Text Box, and Button Send.*
3. *Choosing Hours* using as Resources *the Label, Text Box, Combo Box, and Button Send.*
4. *Creating Group Password* using as Resources *the Label, Text Box, and Button Send.*

f)To *enrol in the group with Activities:*
1. *Select Teacher using the* Resources: *Teachers List, Groups List, Courses List, and Button Send.*
2. *Select Group using the* Resources: *Teachers List, Groups List, Courses List, and Button Send.*
3. *Select Course with Resources of Teachers List, Groups List, Courses List, and Button Send.*
4. *Send Data with the Resource Button Send.*

9. The Notification is generated by the activities of *sending data, reviewing file, creating delivery date, uploading task, downloading task, comment on the wall, and creating group*.
10. The Concurrency is triggered by the activities of *sending data, reviewing file, creating delivery date, uploading task, downloading task, comment on the wall, and creating group*.
11. The AVS View presents IV, PV, and VC.

12. The Adaptation is activates by the activities of *sending data, reviewing file, creating delivery date, uploading task, downloading task, comment on the wall, and creating group*.
13. The session is AVS-Session.



Registrarse
Los siguientes campos son abligatorios
Usuario:
Password:
Repetir Password:
Nombre(s):
Apellidos:
Escoje una foto(maximo 20kb):

La siguiente informacion es para que te puedan localizar mas facilmente, la podras cambiar o llenar despues.

Figure 3. AVS (EVA) developed with workflow ontology.

6 Conclusions and Future Work

This paper has presented a semantic approach to develop groupware based on architectural model. This approach supplies knowledge base, which facilitates and simplifies the development of groupware. The knowledge base supplies a vocabulary that easiness the representation, specification, analysis, and design of this kind of applications. The workflow ontology indicates how users are distributed and organized —i.e. how they communicate, coordinate, collaborate—; how the interaction and adaptation are controlled, as well as how it is presented the interaction in the Views. The future work is orientated to specify a methodology to develop groupware.

Referencias

1. Ellis, C.A., Gibas, S.J., and Rein, G.L. Groupware: some issues and experiences. Communications of the ACM, Vol. 34-1, pp. 39-58, (1991).
2. Roseman, M., and Greenberg, S. Building Real-time Groupware with GroupKit, a Groupware ToolKit. ACM Trans. Computer-Human-Interaction, Vol. 3, 66-106, (1996).
3. 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).
4. 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).

5. 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).
6. Laurillau, Y., and Nigay, L. Clover Architecture for Groupware. Proc. of the ACM Conference on CSCW. New Orleans, Louisiana, USA, pp. 236-245, (2002).
7. Gea, M., Gutierrez, F.L., Garrido, J. L., and Canas, J.J. AMENITIES: Metodología de Modelado de Sistemas Cooperativos presentado en COLINE02.Workshop de Investigaci6n sobre nuevos paradigmas de interacción en entornos colaborativos aplicados a la gestión y difusión del Patrimonio cultural. Granada, Spain, (2002).
8. 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).
9. Penichet, V.M.R., Lozano, M.D., and Gallud. J.A. An Ontology to Model Collaborative Organizational Structures in CSCW Systems. In Engineering the User Interface, Springer, pp.127-139, (2008).
10. Garlan, D., and Shaw, M. An introduction to software architecture, Advances in Software Engineering and Knowledge Engineering, 1, pp. 1-39, (1994).
11. Spivey, J.M. The Z notation: A reference manual. Prentice Hall, (1989).
12. Abrial. J.R. The B-book: Assigning Programs to Meanings, Cambridge University Press.
13. 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)
14. 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)
15. 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).
16. Stevens, R., Goble, C.A., and Bechhofer, S. Ontology-based knowledge representation for bioinformatics. Henry Stewart Publications 1467 – 5463. Briefings in Bioinformatics, Vol. 1-4, pp. 398-414, (2000).
17. Gruber, R. A translation approach to portable ontology specification. Knowledge Acquisition. Vol. 5, pp. 199-220, (1993).
18. Allen, R. Workflow: An introduction. In Workflow Handbook, L. Fisher, Ed. Future Strategies, Lighthouse Point, FL, pp. 15–38, (2001).
19. 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*, pp.121-128.
20. Kozaki, K., Sunagawa, E., Kitamura, Y. and Mizoguchi, R. A Framework for Cooperative Ontology Construction Based on Dependency Management of Modules. ESOE, Vol. 292 of CEUR Workshop Proceedings, pp. 33-44. (2007)
21. 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) .
22. Maccagnan, A., Riva, M., Feltrin, E., Simionati, B., Vardanega, T., Valle, G., Cannata, N. Combining ontologies and workflows to design formal protocols for biological laboratorios, Automated Experimentation, Vol. 2-3, (2010).
23. Mikelakis, M., Papatheodorou, C. An ontology-based model for preservation workflows. In Proceedings of the 9th International Conference on Digital Preservation, (2012).