ISSN 1870-4069

Semantic Formalism for Modelling the Group Interaction

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, Puebla, México

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

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

Abstract. The process of groupware development can be founded on the conceptual modeling of group interaction, since the interaction determines how the group members communicate, collaborate, and coordinate in order to perform some tasks-in accordance with the roles; users can play-to achieve a common goal. Therefore, in this paper a formalism to model the group interaction is proposed, this approach is inspired by formalisms that have been developed within this context: an ontology of the session management policy, which establishes the group organizational structure, in terms of the roles that users (group members) will play; an Model-View-Controller architectural pattern, which establishes a set of recommendations to facilitate the process of groupware development; and a Methodology that supports the process of ontologies development, by using a set of tasks, allowing us to simplify this process. The formalism to analyze and design the interaction in a shared workspace, is composed by the following modeling: 1) Role Modeling; 2) Interaction Modeling; and 3) User Interface Modeling. Finally, a proof of concept based on a case study is presented.

Keywords: Semantic formalism, group interaction, ontology, methontology, model-view-controller architectural pattern.

1 Introduction

The group interaction is a key aspect of the groupware, which is a computer-based system that supports groups of people who are engaged in a common task (or goal), and it provides an interface to a shared environment [1]. Thus, the development of this kind of applications, must be focused on modeled group interaction. In according to Molina [2], four forms of groupware development have been established:

1. *Ad-hoc:* The application is built in a completely adapted way to the specific problem to which it is intended to support.
2. *Use of toolkits:* These provide a higher level of programming abstraction by using functions and API (Application Programming Interfaces).

3. *Use of components:* They allow the construction of groupware by using predefined building blocks.
4. *Use of conceptual modeling:* The process of collaborative environment development is based on conceptual modeling.

With regard to conceptual modeling, some proposals have been made, such as: Coordination Theory [3] supplies a theoretical framework for analyzing coordination; Conceptual Model [4] characterizes the groupware from users' view point with three complementary model: ontological, coordination, and user interface; AMENITIES [5] is based on models of tasks and provides dynamic aspects, using an extension of UML notations called COMO-UML; TOUCHE [6] manages the interaction among the users through UML notations; CIAM [7] supports the user interface design of groupware enabling integration with software processes through UML notation (that it has called CIAN); and Interaction Modeling [8] proposes a framework for analyzing and designing virtual spaces oriented to collaborative work. However, these cannot be considered formal, since they lack the necessary expressivity and formality to specify the group work interaction. On the other hand, several authors [2, 9, 10, 11] have established limitations about conceptual modeling of the work group:

1. Lacking of theoretical and computational models that allow to adequately specify the group activities mediated by information technology.
2. Difficulties for addressing the integral modeling of interactive aspects among individuals and task aspects of group work.
3. Lacking of adequate conceptual specification artifacts for modeling collaborative tasks which have to be mediated by CSCW systems.

Therefore, in this work, a semantic formalism to model group interaction is proposed, which is based on: 1) An ontological model [12, 13, 14] for group organizational structure (which supplies a formal and explicit specification of this structure); 2) A Model-View-Controller (MVC) architectural model [15, 16] to develop groupware (which offers a set of templates that serves as a guideline to analyze, design, and implement groupwork); and 3) Methontology [17, 18] for building ontologies (which uses a set of intermediate representations, based on tabular and graphical notations). This formalism is composed of the specification of: 1) the division of labor in accordance with the established roles (Role Modeling); 2) the group interaction with respect to the defined task type (Interaction Modeling); and 3) the presented Information, the Participant and/or Context views regarding the collaboration carried out by users performing a role (User Interface Modeling).

The paper is organized as follows: The models supporting the formalism are described in Section 2. The background of this conceptual formalism is explained in Section 3. The formalism development is detailed in Section 4. The case study is defined in Section 5. Finally, conclusions and future works are presented in Section 6.

2 Models Supporting the Proposed Formalism

Three are the models aiding the suggested formalism: ontological model, MVC architectural pattern, and methontology.

2.1 Ontological Model

An ontology is presented as an organization's resource and knowledge representation through an abstract model. This representation model provides a common vocabulary of a domain and defines the meaning of the terms and the relations among them. The ontology supplies a set of *concepts* or classes, *relations*, *axioms*, and *instances* to describe a domain in a formal and explicit way [19]. In the groupware domain, the ontologies have mainly been used to model tasks or sessions, by defining concepts and terms, such as group, role, actor, task, etc.. Moreover, semiformal methods (e.g. UML class diagrams, use cases, activity graphs, transition graphs, etc.) and formal ones (such as algebraic expressions) have also been applied to model the sessions.

The ontologies can be implemented in various kinds of languages [20]. Some based on First-order (predicate) logic, other Frames-based languages with more expressive power but less inference capability; others based on descriptive logic [21] that are more robust in the power of reasoning as OWL [22, 23]. On the other hand, the Description Logic provides readily available reasoners such as Pellet [24] and HermiT [25]. OWL ontologies can also be combined with rules using the new W3C Rule Interchange Format (RIF) standard [26]. For developing ontologies are used tools, which provide graphical interfaces that facilitate the knowledge representation and reasoning. This article focuses on Protégé [27], which is an engineering tool open source ontology and a knowledge-based framework. Ontologies in Protégé can be developed in a variety of formats, including OWL, RDF (S), and XML Schema.

2.2 MVC Architectural Pattern

An architectural pattern captures the essence of a successful solution to commonly occurring problems in software design. Thus, a pattern can be seen as a clear and generic set of instructions, ensuring to use a solution that has been proven in countless software design problems with excellent results, allowing customize the pattern to solve specific problems. The importance in the architectural pattern approach is its potential to bridge the gap between high-level requirements and design.

MVC improves modularity by encapsulating volatile implementation details behind stable interfaces that reduce the effort required to understand and maintain existing software. In such way, it reduces the cost and improves the quality of software. The Model characterizes unique forms of data in an application; it will notify to its Views that a change has occurred in the Model, so that they may react suitably. View is a (visual) representation of its model. A view typically has associated a model and is notified when the model (or a part of it) changes, allowing the view to update itself accordingly. All these notifications must be in the model terminology. Users are able to interact with views, and this includes the capacity to access and modify the model. Controller is the link between a user and the application. It provides the user with input by arranging for relevant views to present themselves in suitable places on the screen. It receives user output, translates it into the appropriate messages and passes these messages to one or more views. In groupware, MVC has been used to manage the interaction among user's groups [17, 18, 19, 20, 21].

2.3 Methontology

Methontology is a methodology that supports the ontology construction process, from scratch or the reuse of existing ontologies. It defines common and structured guidelines that establish a set of principles, design criteria and phases for building the ontology. This methodology organizes and converts an informally perceived view of a domain into a semi-formal specification using a set of intermediate representations based on tabular and graphical notations that can be understood by domain experts and ontology developers. All this provides the necessary flexibility and simplicity in the ontology construction process. Methontology includes a set of eleven tasks for structuring knowledge within the conceptualization activity [17]: 1) to build the glossary of terms; 2) to build concept taxonomies; 3) to build ad hoc binary relation diagrams; 4) to build the concept dictionary; 5) to define ad hoc binary relations in detail; 6) to define instance attributes in detail; 7) to define class attributes in detail; 8) to define constants in detail; 9) to define formal axioms; 10) to define rules; and 11) to define instances.

3 Background of the Semantic Formalism

The proposed formalism is derived from created models to manage group interaction: Group Organizational Structure Ontology, and customized MVC Architectural Pattern.

3.1 Ontology- Based Group Organizational Structure

This ontology (see Fig. 1) establishes the *Group Organizational Structure* (GOS) [12, 13, 14] that is governed by a specific *policy*, which determines how the group is organized. This structure is made up of users. *Policy* (Pol) defines a configuration of the group organizational structure accord to each role established. *Users* can be people, either individuals or groups, although they may also refer to systems playing one or more roles. *Role* (R) is responsible for the tasks that users can perform, and provides one status as well as one right/obligation in the application. *Status* (S) describes the role hierarchy. *Rigth/obligation* (R/O) constrains the user actions in the shared workspace. *Task (*T) is made up of one or more activities, allowing users to achieve a given goal in a certain moment.

An *Event* (E) triggers a task. *Activities* (A) are actions that allow a role to execute a set of operations by using resources; which represent the resources used to carry out the activities. *Tasks-Precedence (*TP*)* indicates the order that tasks may have. A Task cab be *Sequential, Parallel, Partially-Concurrent,* and *Fully-Concurrent* kind. *Sequential-Task* (ST) specifies one activity follows the other.

Parallel-Task (PT) happens at the same time, but they use different objects, and no interference between them can occur. *Partially-Concurrent-Task* (PCT) refers to tasks that can be active at the same time but there is no simultaneous modification of any object.

Fully-Concurrent-Task (FCT) occurs when two or more simultaneous tasks to modify rights to same set of objects. Stage (g) reflects each of the collaboration moments and is composed by a set of Tasks. Stage-*Precedence* (SP) indicates the execution order of the stage.

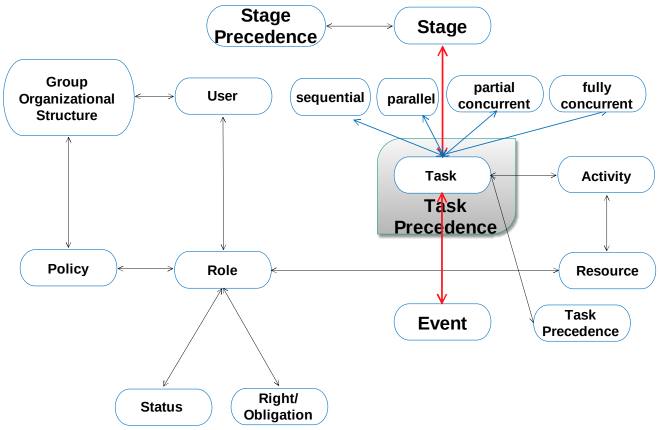

Fig. 1. Group Organizational Struture Ontology [12].

3.2 Customized MVC Architectural Model

The MVC architectural pattern [15, 16] offers a way to simplify the groupware development; providing the necessary flexibility and responsiveness to adjust to the changing needs within the group. This model is customized for characterizing and developing groupware (see Fig. 2).

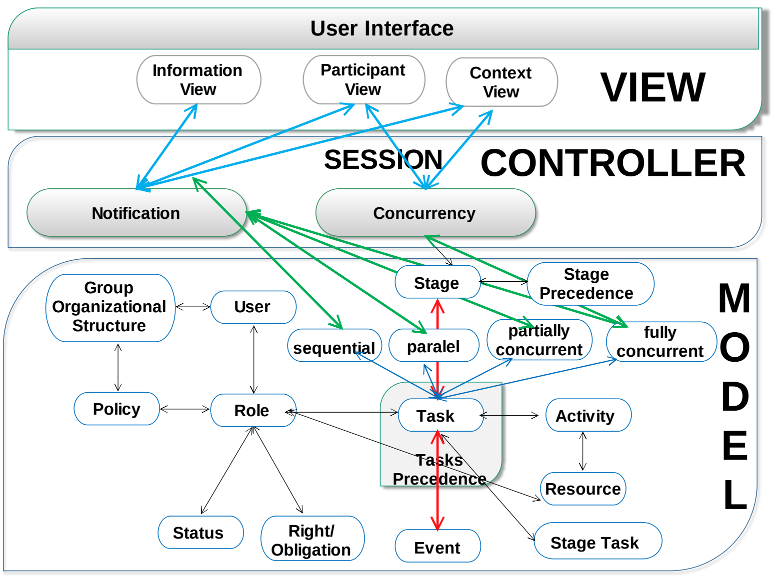

Fig. 2. Layered architectural pattern for building groupware [15].

In the which: the Model is group organizational structure ontology described in the section 3.1. The View is user interface, which is symbolized by the Information View (that provides all the information that helps the user to interact with the application), the Participant View (that allows to each user to be aware of what other's users are doing), and the Context View (that represents workspace where all information of shared resources is shown; which is named memory or history group).

The Controller establishes the notification and concurrency to manage and control appropriately group interaction.

4 The semantic Formalism Development

Group organizational structure ontology is considered the principal base for defining the three models that constituent the semantic formalism. Furthermore, some entities of the Interaction and User Interface Modeling are added by using MVC architectural pattern. The usage of rules, and tabular notation is taken from methontology.

4.1 Role Modeling

The group organizational structure ontology stipulates the entities, relations, and rules that determine the division of labor, for the roles to perform the group work in an appropriate manner. So, the Role Modelling can be specified by defining at what moment each task is executed, and by whom. Consequently, this model must define the Stages (g) in which the group work will be carried out, the order (Precedence Stage - SP) in which they will be made, the Tasks (T) and its order will be executed in each Stage, and the role that will perform them.

4.2 Interaction Modeling

The way in which users interact depends on the type of task they perform according to the role they play. Since: A sequential task establishes that a role should expect to be notified that another finishes his task so he can start his.

A parallel task determines that two or more roles can perform tasks that are different, at the same time; notifying to the roles and entities corresponding the carried-out modifications. A partial concurrent task stipulates that two or more roles perform the same task at the same time but modify different resources, therefore, it should only notify to the appropriate roles and entities the made adjustments. A complete concurrent task indicates that two or more roles perform the same task at the same time using the same resources, therefore, the notification and concurrence should be implemented.

Consequently, in all tasks, the communication (Cm) and collaboration (Cb) between users are used, only in the last task the coordination (Cr) is done. In such a way that the template (see Table 2) correspondent to the Interaction Modelling should contain the Role with the Tasks carried out by he/she, the Task Type (TT), and used mechanism (Notification-N-and Concurrency-C). The elements of this modeling are based on the analyzed ontology and MVC. The columns and rules are founded on methontology.

4.3 User Interface Modeling

In accordance with the customized MVC architectural pattern to develop groupware, the user interface is structured with respect to presented views, which can be: information view, participant view, and/or context view. The view shown depends on the task type performed. The view content be subject to used resources to execute the task. The information view is displayed when any task type has occurred. The participant view or context view are exhibited when a Partial Concurrent or Fully Concurrent has happened. On the other hand, the information view implicates individual actions, while the other views represent collaborative work. The participant view generates the group awareness. The context view produces the group memory. Therefore, the template of the User interface modeling (see Table 3) presents User Interface (UI), the three view types and the task type that produces them. The three views allow the communication; while the participants and the context view facilitate collaboration and coordination.

5 Proof Conceptual of the Semantic Formalism

The case study is an Academic Virtual Space (AVS), which provides a shared workspace to simplify student's access through the Internet to the course material imparted by the teachers. AVS presents a simple stage called Academic Collaboration (AcC), so that the column of the stage, and stage precedence are omitted in the paper rest. It includes two roles: Teacher (Tc), and Student (St). The Table 1 presents the AVS description.

Table 1. Description of the application AVS.

R	E	T	TP	A	R
Tc	Access to AVS	Registering	1	Enter data	labels, box text, bottom
Tc	Starting Session	Login	2	Enter data	labels, box text, bottom
Tc	Logged	Creating Profile	3	Enter data	labels, box text, bottom
Tc	Logged	Creating Course	4	Enter data	labels, box text, bottom
Tc	Access to Course	Publishing HomeWork	5	Enter data	labels, box text, bottom, file
Tc	Homework	Creating deadline	6	Enter data	labels, box text, bottom
Tc	Creating deadline	Download St HW	10	Enter data	labels, box text, bottom, file
Tc	Download HW	Upload Reviews	11	Enter data	labels, box text, bottom, file
Tc		Send Messages (Mss)		Write Mss	labels, box text, bottom
St	Access to AVS	Registering	1	Enter data	labels, box text, bottom
St	Starting Session	Login	2	Enter data	labels, box text, bottom
St	Logged	Creating Profile	3	Enter data	labels, box text, bottom
St	Created Course	Registering Course	7	Enter data	labels, box text, bottom
St	Register Course	Download Tc HW	8	Enter data	labels, box text, bottom, file
St	Download HW	Upload HW	9	Enter data	labels, box text, bottom, file
St		Send Messages (Mss)		Write Mss	labels, box text, bottom

The Template of the Role Modeling (see Table 2) is gotten in accordance with the explicated in the section 4.1. In this template is possible to see the role that participates and in what moment does this. Furthermore, the task called "Send Messages" can be performed when the role (Teacher or Student) requires it.

The Template of the Interaction Modeling (see Table 3) is developed with respect to the explained in the section 4.2. In this template, the group interaction is visualized through the performed task type, which determine the required aspect (communication, collaboration, or coordination) to support the group interaction. In addition, a set of rules is added for establishing and controlling the users' participation in this interaction.

The Template of the User Interface Modeling (see Table 4) is acquired regarding with the clarified in the section 4.3. This template presents the views' resultants from task and synchronization type used. In addition, a set of rules to determine the displayed view type is presented in the template.

Table 2. Template of the Role Modeling.

Role	Event	Task	TP
Tc	Access to AVS	Registering	1
Tc	Starting Session	Login	2
Tc	Logged	Creating Profile	3
St	Access to AVS	Registering	1
St	Starting Session	Login	2
St	Logged	Creating Profile	3
Tc	Logged	Creating Course	4
Tc	Access to Course	Publishing HomeWork	5
Tc	Homework	Creating deadline	6
St	Created Course	Registering Course	7
St	Register Course	Download Tc HW	8
St	Download HW	Upload HW	9
Tc	Creating deadline	Download St HW	10
Tc	Download HW	Upload Reviews	11
Tc		Send Messages (Mss)	
St		Send Messages (Mss)	

Table 3. Template of the Interaction Modeling.

R	Task	PT	TT	N	C	Aspect	Rule
Tc	Registering	1	ST, PT	X		Cm	if [[Tsk](?X) &
Tc	Login	2	ST, PT	X		Cm	[Act](?Y)](?X,?Y)] then
Tc	Creating Profile	3	ST, PT	X		Cm	[composited Act] (?X,?Y)
St	Registering	1	ST, PT	X		Cm	if [[A](?X) and
St	Login	2	ST, PT	X		Cm	[R](?Y)](?X,?Y)] then [has R]
St	Creating Profile	3	ST, PT	X		Cm	(?X,?Y)
Tc	Creating Course	4	ST	X		Cm, Cb	if [[T](?X) and
Tc	Publishing HomeWork	5	ST	X		Cm, Cb	[PCT](?Y)](?X,?Y)] then is_a
Tc	Creating deadline	6	ST	X		Cm, Cb	PCT] (?X,?Y)
St	Registering Course	7	ST	X		Cm, Cb	if [[T](?X) and [PCT](?Y)] and
St	Download Tc HW	8	PCT	X	X	Cm, Cb	[N](?Z)] (?X,?Y, ?Z)] then
St	Upload HW	9	PCT	X	X	Cm, Cb	actives N] (?X,?Y, ?Z)
Tc	Download St HW	10	PCT	X	X	Cm, Cb	if [[T](?X) and [FCT](?Y)] and
Tc	Upload Reviews	11	ST	X		Cm, Cb	[N](?Z)] and [C](?W)
Tc	Send Messages (Mss)		FCT	X	X	Cm, Cb, Cr	(?X,?Y,?Z,?W)] then actives N
St	Send Messages (Mss)		FCT	X	X	Cm, Cb, Cr	and C] (?X,?Y,?Z,?W)

6 Conclusions and Future Work

In this paper, a semantic formalism for modeling the group interaction in groupware has been established. As a result, this approach is constituted by three models: role modeling (the roles are the actives participants of the interaction), interaction modeling (the task type determines how the users will interact), and user interface modeling (the interaction is performed in the shared workspace, which is presented in the user interfaces). This formalism is based on the group organizational structure ontology; customized MVC architectural model, and methontology, which supply a set of elements to model the interaction of group through representations based on tabular notations.

The future work is orientated to specify a methodology to develop groupware, which is founded in the formalism here proposed.

Table 4. Template of the User Interface Modeling.

R	Task	TT	N	C	IV	PV	CV	Aspect
Tc	Registering	ST, PT	X		X			if [[T](?X) and [N](?Y)] and
Tc	Login	ST, PT	X		X			[IV] (?Z)(?X,?Y,?Z)] then
Tc	Creating Profile	ST, PT	X		X			actives IV] (?X,?Y, ?Z)
St	Registering	ST, PT	X		X			if [[T](?X) and [PCT](?Y)] and
St	Login	ST, PT	X		X			[PV](?Z)] and [CV](?W)
St	Creating Profile	ST, PT	X		X			(?X,?Y,?Z,?W)] then actives PV
Tc	Creating Course	ST	X		X			and CV] (?X,?Y,?Z,?W)
Tc	Publishing HomeWork	ST	X		X			if [[T](?X) and [FCT](?Y)] and
Tc	Creating deadline	ST	X		X			[PV](?Z)] and [CV](?W)
St	Registering Course	ST	X		X			(?X,?Y,?Z,?W)] then actives PV
St	Download Tc HW	PCT	X	X	X	X	X	and CV] (?X,?Y,?Z,?W)
St	Upload HW	PCT	X	X	X	X	X	
Tc	Download St HW	PCT	X	X	X	X	X	
Tc	Upload Reviews	ST	X		X			
Tc	Send Messages (Mss)	FCT	X	X	X	X	X	
St	Send Messages (Mss)	FCT	X	X	X	X	X	

References

1. Ellis, C.A., Gibas, S.J., Rein, G.L.: Groupware: Some Issues and Experiences. Communications of the ACM, Vol. 34-1, pp. 39–58 (1991)
2. Molina, A., Redondo, M., Ortega. M.: A Review of Notations for Conceptual Modeling of Groupware Systems. In New Trends on Human-Computer Interaction (Eds. J. Macías, A. Granollers, P. Latorre). pp. 1–12 (2009)
3. Crowston, K., Rubleske J., Howison J.: Coordination theory: A ten-year retrospective. Human-Comput. Interaction in Management Information Systems. M. E. Sharpe, Inc. (2006)
4. Ellis, C., Wainer, J.: A conceptual model of groupware. Proceedings of the 1994 ACM Conference on CSCW, pp. 79–88 (1994)
5. Garrido, J. L., Gea, M., Padilla, N., Canas, J.J., Waern Y.: AMENITIES: Modelo de entornos cooperativos. Actas del III Congreso Internacional Interacción Persona-Ordenador, pp. 97–104 (2003)
6. Ruiz Penichet., V.M.: Task-Oriented and User-Centred Process Model for Developing Interfaces for Human-Computer-Human Environments. Ph.D. dissertation, Universidad de Castilla-La Mancha (2007)
7. Molina, A.I., Redondo, M.A., Ortega, M., Hope, U.: ClAM. A methodology for the development of groupware user interfaces, Journal of Universal Computer Science, vol. 14-9, pp. 1435–1446 (2008)

8. Rodríguez, D., García-Martínez, R.: Modeling the Interactions in Virtual Spaces Oriented to Collaborative Work. Chapter 10, Carlos Mario Zapata, Guillermo González, Roberto Manjarrés, Fabio Alberto Vargas, and Wiliam Arévalo (Eds.), Software Engineering: Methods, modeling, and Teaching, Vol. 1, Lima, Perú (2012)
9. Rodríguez, D., García-Martínez, R.: Modeling the Interactions in Virtual Spaces Oriented to Collaborative Work. Chapter 10, Carlos Mario Zapata, Guillermo González, Roberto Manjarrés, Fabio Alberto Vargas, and Wiliam Arévalo (Eds.), Software Engineering: Methods, modeling, and Teaching, Vol. 1, Lima, Perú (2012)
10. Sosa, M., Zarco, R., Postiglioni, A.: Modelando Aspectos de Grupo en Entornos Colaborativos para Proyectos de Investigación. Revista de Informática Educativa y Medios Audiovisuales Vol. 3, pp. 22–31 (2006)
11. Giraldo, W., Molina, A., Collazos, C., Ortega, M., Redondo, M.: Taxonomy for Integrating Models in the Development of Interactive Groupware Systems. Journal of Universal Computer Science, Vol. 14-19, pp. 3142–3159 (2008)
12. Anzures-Garcia, M., Sanchez-Galvez, L.A., Hornos, M.J., P. Paderewski: A Knowledge Base for the Development of Collaborative Applications. Engineering Letters, vol. 23-2, International Association of Engineers, Hong Kong, pp. 65–71 (2015)
13. 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), pp. 807–812 (2008)
14. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., Paderewski-Rodríguez, P.: Ontology-Based Modeling of Session Management Policies for Groupware Applications. Lecture Notes in Computer Science, Vol. 4739, pp. 57–64, Springer, Heidelberg (2007)
15. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., Paderewski, P.: Facilitating the development of Collaborative Applications with the MVC Architectural Pattern. Chapter 19. Software Engineering: Methods, Modeling, and Teaching, Vol. 4, In press (2016)
16. Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J., Paderewski-Rodríguez, P.: A software architecture for defining a methodologic approach to develop Collaborative Applications. Research in Computing Science: Advances in Computer Science and Engineering, Vol. 105, pp. 9–20 (2015)
17. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: From Ontological Art towards Ontological Engineering. In: Spring Symposium on Ontological Engineering of AAAI. Stanford University, California, pp. 33–40 (1997)
18. Fernández-López, M., Gómez-Pérez, A., Pazos, A., Pazos, J.: Building a Chemical Ontology Using Methontology and the Ontology Design Environment. IEEE Intelligent Systems & their Applications. Vol. 4-1, pp. 37–46 (1999)
19. Gómez-Pérez, A., Fernández-López, M., and Corcho, O.: Ontological Engineering with Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web. Springer (2004)
20. Uschold, M., Grüninger, M.: Ontologies: Principles, Methods and Applications. Knowledge Engineering Review 11(2), pp. 93–155 (1996)
21. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. F. editors: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
22. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language semantics and abstract syntax. W3C Recommendation, 10 February (2004)
23. 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, Vol. 1-1, pp. 7–26 (2003)
24. Pellet OWL reasoner. Maryland Information and Network Dynamics Lab. http://www.mindswap.org/2003/pellet/index.shtml (2003)

25. 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), volume 4603 of Lecture Notes in Artificial Intelligence, Springer pp. 67–83 (2007)
26. RIF RDF and OWL Compatibility. W3C Recommendation, Available at http://www.w3.org/TR/rif-rdf-owl/ (2010)
27. Protégé Ontology Editor and Knowledge Acquisition System, http://protege.stanford.edu