CONGAS: a *CO*llaborative ontology development framework based on *N*amed *G*r*A*ph*S*

Daniele Bagni, Marco Cappella, Maria Teresa Pazienza, Armando Stellato

ART Group, Dept. of Computer Science, Systems and Production
University of Rome, Tor Vergata
Via del Politecnico 1, 00133 Rome, Italy
{daniele.bagni, marco.cappella}@gmail.com
{pazienza, stellato}@info.uniroma2.it

Abstract. The process of ontology development involves a range of skills and know-how often requiring team work of different people, each of them with his own way of contributing to the definition and formalization of the domain representation. For this reason, collaborative development is an important feature for ontology editing tools, and should take into account the different characteristics of team participants, provide them with a dedicated working environment allowing to express their ideas and creativity, still protecting integrity of the shared work. In this paper we present CONGAS, a collaborative version of the Knowledge Management and Acquisition platform Semantic Turkey which, exploiting the potentialities brought by recent introduction of context management into RDF triple graphs, offers a collaborative environment where proposals for ontology evolution can emerge and coexist, be evaluated by team users, trusted across different perspectives and eventually converged into the main development stream.

1. Introduction

The process of ontology development requires different skills and know-how, such as a deep understanding of the domain to be represented and proper expertise in domain modeling, which rarely can be found in one single person. Moreover, assessing the knowledge structure of an ontology typically involves different refinement steps, personal rethinking and discussion. For this reason, even the realization of medium-size ontologies often requires the collaboration of several experts, each of them bringing their own knowledge and skills towards the development of a consistent, hopefully stable, and potentially reusable domain representation.

As a natural consequence for the fact, and thanks to the maturity now reached by ontology development tools and to the proliferation of collaborative solutions brought by the advent of Web 2.0, we have seen in the last years an emerging interest in the research community towards the identification of requirements [1,2] and the proposal of innovative solutions [3,4,5] for collaborative development of ontologies.

The requirements and features which have emerged in these works mainly address the integration of tools supporting communication and discussion among users, the resolution of issues related to concurrent editing, and the definition of standard access control and contribution modalities. What lacks thereof is the ability for users to go beyond simple discussion or voting about round-the-corner ontology modifications, to follow or even create arbitrary evolution paths for the ontologies they are working on.

In our research work, we have tried to propose a novel approach to collaborative ontology development which would fill the above gap, by accounting the effort and results of the Semantic Web Interest Group on Named Graphs [6], and by exploiting the possibilities offered by their introduction.

In our approach, ontological knowledge is distributed across different *contexts* (identified by diverse named graphs), which identify the branched workspace of team members as well as the main development trunk shared by all of them. Users can thus freely work in their personal context (which is given by the merge of the named graph assigned to them and of the main development trunk), but can also inspect other *context*s, access their content, and *trust* (part of) the statements contained there, thus virtually importing them into their context. This poses unlimited possibilities to the creativity of each single team member, who can bring his work ahead and lately have it discussed through traditional communication gadgets or even implicitly promoted as he finds other users accepting and *trusting* his proposals.

Thanks to the introduction of Named Graphs into a few of the currently available triple store technologies, such as Jena [7] (through the NG4J [8] library extension), and Sesame 2.0 [9], we have also been able to develop and present here CONGAS, a novel system for collaborative editing of Semantic Web ontologies, which has been developed as a parallel collaborative version of the Knowledge Management and Acquisition platform Semantic Turkey [10].

2. Related Works

The first published result on Named Graphs dates back to the work of Carroll et al. [11], though other works have addressed the problem of data provenance and locality in the years before [12,13]. The introduction of Named Graphs has been a necessary step for the Semantic Web, their ability to express meta-information about RDF graphs is an important ingredient for addressing a range of its important requirements, such as Data syndication, Information Access, RDF Signature [14] and Trust [15], expressing propositional attitudes [16], scoping assertions and logic and managing ontology versioning and evolution. All of the above mainly account for one necessity: the ability to track provenance of single graphs merged into compound RDF repositories.

To our knowledge, no collaborative environment for development of knowledge graphs of the RDF family has widely exploited Named Graphs support, to introduce user spaces. There are however other aspects of collaborative ontology development which have been evidenced and widely experimented in several works. We mention a few of them according to their specific contributions:

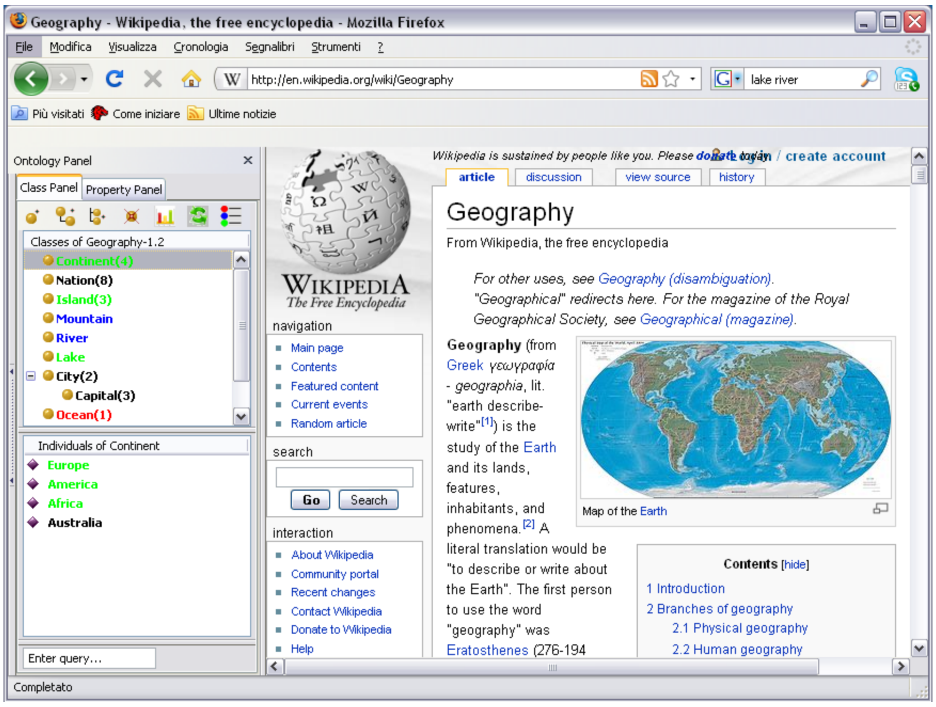
- *Wiki adoption*: [17] and [18] offer ontology development collaborative environments based on (modified versions of) wiki platforms. Both of them do not address the general target of ontology development, and are respectively oriented towards Enterprise Modeling and acquiring consensus over the definition of reusable Ontology Design Patterns [19]

– *Ontology modularization*: the Hozo ontology editor [3] enables asynchronous development of ontologies that are subdivided into multiple inter-connected modules. A developer checks out and locks a specific module, edits it locally, and then checks it back in. Each module are however still owned by all team members, thus limiting the freedom of action and rapid drafting of new ontology branches
– *Methodology*: The Cicero tool [20] implements the DILIGENT [21] methodology to for collaborative ontology development. The DILIGENT methodology focuses on the process of argumentation, thus supporting the generation of discussions about ontology maturing, both in general as well as for specific resources
– *Models*: in [22] the authors present an ontology supporting the definition of requirements for collaborative ontology development, while in [23] an workflow ontology can be used to describe different kind of workflows for the same task (they also experimented their model in expressing the DILIGENT methodology cited above).
– *Full integration into complete ontology development tools:* in [5], an extension for the popular Knowledge Management and Acquisition tool Protégé [24] is presented, perfectly integrating several contradistinguish features for collaborative ontology development, into the base tool (and thus beneficiating of all of its traditional editing facilities): these include *user management*, enabling *discussions/annotations*, *workflow support* (through the workflow ontology cited above) and *synchronous/asynchronous editing* of available data.

3. True Collaboration through Interwoven Ontology User Spaces

The objective which has been targeted in the design of CONGAS was to develop a completely new stereotype of collaboration, in which users could discuss, reject or approve modifications to existing ontology resources (from very common ontological axioms, such as classification and *is-a* organization, to detailed triple level analysis) as well as (and this is the novelty with respect to existing tools and methodologies) create and propose entirely new ontology branches, which can then be aligned/merged to the core ontology.

In our model, each user is assigned his dedicated *user space*, and can develop new extensions for the edited ontology without the need to propose them step-by-step on the main development trunk, nor the risk of a totally unrestricted editing, resulting in the production of noisy data which could be entropic for other users and thus for the whole collaborative process. By default, each user is able to view in his *space* the main development trunk and an ontology branch associated to him. The user has full read/write privileges over his personal development branch, while he can only propose changes to the main trunk and discuss them through ordinary argumentation/voting mechanisms (forum, polls etc…). Note that this limitation over the main trunk is confined to what is actually translated into deletion of triples (due to



Wikipedia is sustained by people like you. Please donate today.

Geography

From Wikipedia, the free encyclopedia

For other uses, see Geography (disambiguation).
"Geographical" redirects here. For the magazine of the Royal Geographical Society, see Geographical (magazine).

Geography (from Greek *γεωγραφία* - *geographia*, lit. "earth describe-write"[1]) is the study of the Earth and its lands, features, inhabitants, and phenomena.[2] A literal translation would be "to describe or write about the Earth". The first person to use the word "geography" was Eratosthenes (276-194

Fig. 1 A user viewing ontology contributions proposed by other team members

the monotonic discipline of RDF [25]), but addition of axioms referring to resources in the main trunk can be handled through the user space; for example, if the user is willing to add a rdfs:subClassOf relationship between two classes (namely: main:A and main:B) belonging to the main trunk, this is not considered a modification, since it involves the sole creation of the triple:

main:A rdfs:subClassOf main:B

which can thus be stored in his personal ontology branch (that is, in his *user space*). The component which is in charge of projecting the set of triples governed by the rdfs:subClassOf predicate into a tree visualization of ontology classes, will take into account this triple and show the tree accordingly, with class main:A arranged under class main:B.

What thus happens is that different users could even participate in suggesting changes to the main trunk (which is considered *frozen*) while they can freely contribute to the evolution of the ontology taking their way on extending the set of resources and their related axioms. *User spaces*, though assigned on a per-user basis and granting write-privileges only to their owner, can however be browsed by other users (see Fig. 1, where concepts and instances created by team members, and associated to them by different colors, are browsed by current user): the content of each space can be exported to the spaces of other users who decide to *trust* it and add it to their respective evolution branch. This way, it is easy for new knowledge to be produced by independent users, discussion is supported by forums/polls (which are also enabled for foreign user spaces), while convergence of the result is assured by the trust&import mechanism which allows several users to quickly share (portions of) proposed branches and thus promote them for the next main trunk release. Seen from the perspective of the triple store engine, the RDF repository is an aggregation of:

– several named graphs, representing foreign ontologies (which can only be accessed with *read* privileges by all kind of users) imported by the main trunk
– a core graph, containing data from the frozen main trunk. It is accessible by all users, and it is read-only for all standard users, though granting write permissions to ontology administrators
– a set of named graphs associated to user spaces. Each of them can be inspected by all users, has write permissions only for the owner of the space (though it can be entirely removed, but not modified, by ontology administrators)
– a set of named graphs associated to foreign ontologies imported by development branches from user spaces.

Management of ontologies to be visualized for each user is done by first importing all the first three set of graphs from the above list. Then, the list of owl:imports statements for the user development branch is inspected, and all named graphs from the fourth set which is cited in the object of these statements is added to the ontologies to be visualized for that user.

4. The Hosting Application: Semantic Turkey

CONGAS has been developed on top of Semantic Turkey (ST, from now on), a Knowledge Management and Acquisition System realized by the ART group of the University of Rome, Tor Vergata.

Developed as a Web Browser extension (available for the popular Web Browser Firefox), Semantic Turkey aims at reducing the impedance mismatch between domain experts and knowledge investigators on the one side, and knowledge engineers on the other, by providing them with a unifying platform for acquiring, building up, reorganizing and refining knowledge. Semantic Turkey offers in fact traditional menu options for ontology development, but it is also able to react to a range of several intuitive actions (such as drag'n'dropping objects – text in the standard version – from the web page to the ontology panels) with contextual outcomes (i.e. similar gestures may result in even long sequences of ontology modifications which depend on the kind of object dragged from the Web Page, on the type of resource where it is dragged etc…). ST deeper interaction with Web content is not only limited to the possibility of importing text and other sorts of media from Web Pages; it also features an extension mechanism which covers all of its aspects and technologies: from user interaction, through its UI extension points linked to the same Mozilla Extension mechanism which is at the base of its hosting web browser, to knowledge management and persistence (thorough standard OSGi service extendibility).

In realizing CONGAS, we have first examined the different possibilities which were available for converting the system into a distributed environment. These aspects will be discussed in the next section.

5. Architecture

For economy of space, we will limit ourselves here to describe those architectural changes in Semantic Turkey which have been introduced to realize its collaborative version CONGAS. For a detailed description of Semantic Turkey architecture and knowledge model, the reader may refer to [10], while [26] contains relevant updates related to the extension mechanism.

Semantic Turkey architecture is organized around a three-tier layering, with the presentation layer embodying the true Firefox extension and the other two layers built around java technologies for managing the business logic and data access.

Both the two interlayer interfaces could be, in principle, be separated in a distributed environment (http communication is already adopted between the presentation and middle layer, and data access interface can easily be implemented towards remote RDF repositories) so, when thinking about needed reengineering of this architecture for porting ST to a collaborative framework, we faced the following possibilities:

- Centralizing the sole Persistence layer and realize collaborative client applications with a rewritten business logic to support transaction based communication with the RDF service.
- Keeping the presentation layer for client applications, and move both server and persistency on the centralized collaborative server
- Split the middle layer into two components, one which is bundled with client applications and provides the required business logic for their operations, and the other one which coordinates collaboration between clients and manages all related services

The first one has been discarded, since it would have produced nothing more than a user interface for transaction-based knowledge repositories. The third option would have proven to be the best solution, though one consideration about client technology made us leaning towards the second one: what is reported as the presentation layer in ST architecture, is actually represented by the whole array of technologies supporting browser extendibility. For example, with respect to the current implementation available for Firefox, an extension of the JavaScript language is adopted to support business logic of extensions to be developed; it can thus be used to handle the minimal support required by user interaction, while demanding to the collaborative server most of the necessary computation. Analogous technologies satisfying the minimal computation requirements of user interaction are expected to be found for all classes of browsers, thus not invalidating the architecture on its own.

5.1. Coordination and Synchronization

Coordination between users is important in a collaborative environment, as well as keeping sync between what they see while editing the ontology, and changes to the model which can have been submitted by other users.

A refresh button is present in each client, which has the double function of activating (when depressed) a complete refresh of the graphic view over the ontology and of alerting (by blinking) users when a change has been made by another team member. A log of changes to the whole ontology repository is also available, so that the user can account for these changes in case they generate some conflict with or provide useful input for the work he is doing.

The process of convergence towards a shared evolution for the ontology (i.e. freezing proposed changes in the development *trunk*) is activated through different triggering events: roughly divided as *implicit triggers* (e.g., when a certain percentage of team members has reached consensus over a resource/statement, that is, is trusting, and thus importing, the given resource/statement in its user space) and *explicit* ones (e.g., by explicit approval, through argumentation services such as polls and forums, see next section for details).

Access management divides users according to three main categories:

– *Viewers* who can access the ontology, view resources, and comment or vote on the choices made (this role is usually assigned to domain experts). Those users, then, can "look and speak"
– *Users*: in addition to the rights of viewers, they own a dedicated user space, so that they can "look, speak, and propose".
– *Administrators*. An administrator has all the rights of a user, but it is also able to modify the main development trunk, as well as provide other coordination activity such as moderating forums and accepting proposals. Thus, an administrator can “look, speak, propose and validate ".

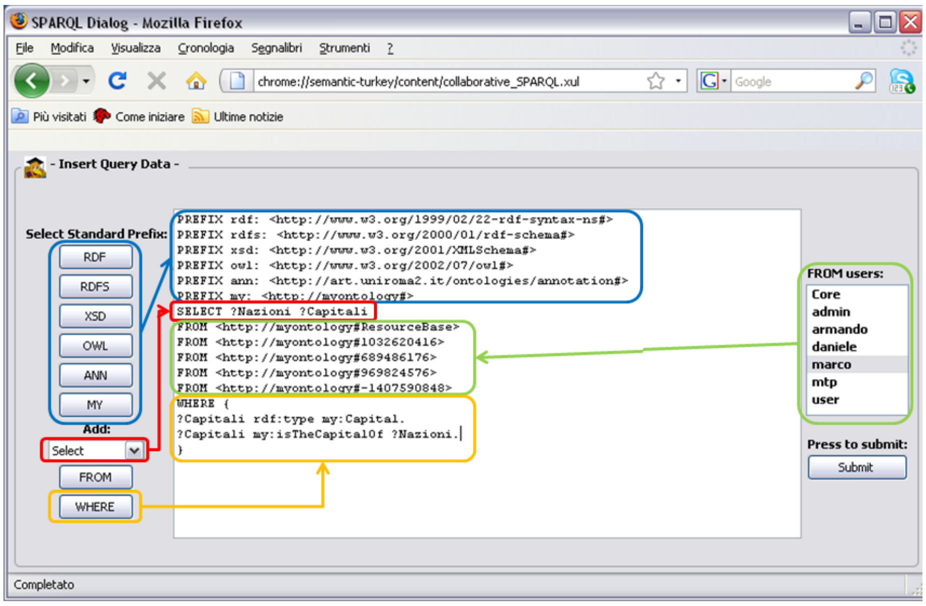
Finally, a simple versioning system allows administrators to freeze snapshots of developed ontologies at a certain time and store them with an assigned tag (usually, a version number). A version manager enables then users to retrieve them and inspect their content.

5.2. Services

In building CONGAS, we have tried to integrate several features and tools supporting collaborative development, together with the concept of *user space* and *model trusting* which pervade all of its architectural choices.

A *poll-based mechanism* allows users to express their opinions. They may choose on open arguments for discussion (*open polls*), where both the theme and options for polling are chosen by one of the users, as well as on validity of all statements available in development trunk and branches (*standard polls*). Standard polls are automatically associated by the system to their related statement, and can be easily accessed when inspecting the projection of that statement (a class/property in the tree, an individual in the list of instances of a class, or a valued property in a resource description).

With a similar approach, also a *forum* has been added to the system, enabling both the creation of free thematic discussions about ontology evolution, and of discussions focused on proposed resources and statements – e.g., whenever a user adds a new resource, it is easy to open a discussion on it: the thread is automatically matched by the resource URI and it can lately always be accessed from a context menu option for that resource. Emailing is also supported, by retrieving public mail addresses of registered users.


SPARQL Dialog - Mozilla Firefox
File Modifica Visualizza Cronologia Segnalibri Strumenti ?
chrome://semantic-turkey/content/collaborative_SPARQL.xul
Google
Più visitati Come iniziare Ultime notizie
- Insert Query Data -
Select Standard Prefix:
RDF
RDFS
XSD
OWL
ANN
MY
Add:
Select
FROM
WHERE
FROM users:
Core
admin
armando
daniele
marco
mtp
user
Press to submit:
Submit
Completato

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX ann: <http://art.uniroma2.it/ontologies/annotation#>
PREFIX my: <http://myontology#>
SELECT ?Nazioni ?Capitali
FROM <http://myontology#ResourceBase>
FROM <http://myontology#1032620416>
FROM <http://myontology#689486176>
FROM <http://myontology#969824576>
FROM <http://myontology#-1407590848>
WHERE {
?Capitali rdf:type my:Capital.
?Capitali my:isTheCapitalOf ?Nazioni.
}
```

Fig. 2 The SPARQL query panel, with facilities for restricting the domain of the query to specific user contexts

Finally, also query support has been integrated with the user spaces paradigm. A SPARQL interface (Fig. 2) allows users to select which user spaces will be considered (panel on the right) when retrieving tuples from the repository (the main development trunk is put by default in the query template, though it may be excluded by the user by manually changing the query code).

6. Conclusions

Apart from its services and functionalities for supporting collaboration, which are in line with state-of-art tools on collaborative editing (and improve them in some cases, as for the generation of forum threads and polls automatically-linked to proposed ontology resources), the main contribution of CONGAS to collaborative ontology development resides in its coordinated editing of evolution branches proposed by users. The possibilities offered by Named Graphs open up a completely novel scenario in which users may freely (and massively) contribute to the main trunk of development of an ontology, without the risk of over-generating undesired axioms, nor suffering from the impedance brought by a strictly disciplined update procedure. It is this aspect, stressing autonomy and independence of the user, which fills the gap between collaboration methodologies such as the already mentioned DILIGENT, and current implementations of collaboration tools: where these latter (coll. Protégé, or SWOOP [27]) provide support for collaboration and discussion on one single ontology (which may thus implement the **analysis** and **revision** steps of DILIGENT), CONGAS allows for a real **revision** and **local adaptation** phase, with distributed users improving their own branch of the ontology, according to their preferences (and/or needs), and then administrators (like the *board* in DILIGENT) participating to the final **revision** step. Finally, the strong interaction with the Web guaranteed by the fact of being hosted on a web browser, which has characterized its single-user predecessor, well matches with the needs and requirements of a collaborative environment, easing introduction of web-based collaboration facilities.

References

1. Noy, N., Chugh, A., Alani, H.: The CKC Challenge: Exploring tools for collaborative knowledge construction. IEEE Intelligent Systems 23(1), 64-68 (2008)
2. J., Seidenberg, Rector, A.: The state of multi-user ontology engineering. In : 2nd International Workshop on Modular Ontologies at KCAP 2007, Whistler, BC, Canada (2007)
3. Sunagawa, E., Kozaki, K., Kitamura, Y., Mizoguchi, R.: An environment for distributed ontology development based on dependency management. In : 2nd International SemanticWeb Conference (ISWC2003), Sanibel Island, FL, USA (2003)
4. Braun, Simone, Schmidt, Andreas, Zacharias, Valentin: Ontology Maturing with Lightweight Collaborative Ontology Editing Tools. In Gronau, Norbert, ed. : 4th Conference on Professional Knowledge Management - Experiences and Visions, Workshop on Productive Knowledge Work (ProKW 07), GITO, pp.217-226 (2007)
5. Tudorache, Tania, Noy, Natalya, Tu, Samson, Musen, Mark: Supporting Collaborative Ontology Development in Protégé. In : ISWC '08: Proceedings of the 7th International Conference on The Semantic Web, Berlin, Heidelberg, pp.17-32 (2008)
6. http://www.w3.org/2004/03/trix/: Named Graphs / Semantic Web Interest Group. In: World Wide Web Consortium - Web Standards. Available at: http://www.w3.org/2004/03/trix/
7. McBride, Brian: Jena: Implementing the RDF Model and Syntax Specification. In : Semantic Web Workshop, WWW2001 (2001)
8. NG4J - Named Graphs API for Jena. In: NG4J - Named Graphs API for Jena. Available at: http://www.wiwiss.fu-berlin.de/suhl/bizer/ng4j/
9. Broekstra, Jeen, Kampman, Arjohn, van Harmelen, Frank: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In : The Semantic Web - ISWC 2002: First International Semantic Web Conference, Sardinia, Italy, pp.54-68 (2002) June 9-12.
10. Griesi, Donato, Pazienza, Maria, Stellato, Armando: Semantic Turkey - a Semantic Bookmarking tool (System Description). In Franconi, Enrico, Kifer, Michael, May, Wolfgang, eds. : The Semantic Web: Research and Applications, 4th European Semantic Web Conference, ESWC 2007, Innsbruck, Austria, June 3-7, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4519, pp.779-788 (2007) Innsbruck, Austria, June 3-7.
11. Carroll, Jeremy, Bizer, Christian, Hayes, Pat, Stickler, Patrick: Named Graphs, Provenance and Trust. In : WWW '05: Proceedings of the 14th international conference on World Wide Web, New York, NY, USA, pp.613-622 (2005)
12. Sintek, M., Decker., S.: Triple - a query, inference, and transformation language for the semantic web. In : Proceedings of ISWC’2002, p.364–378 (2002)
13. R., Guha, Fikes, R.: Contexts for the semantic web. In : Proceedings of ISWC’2004 (2004)
14. Carroll, Jeremy: Signing RDF Graphs. In : 2nd International Semantic Web Conference

(ISWC03), volume 2870 of LNCS, pp.5-15 (2003)

15. Bizer, Christian, Oldakowski, Radoslaw: Using context- and content-based trust policies on the semantic web. In : WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, New York, NY, USA, pp.228-229 (2004)
16. Ibrahim, A.: Agent Communication Languages (ACL). (Accessed 2000)
17. Rospocher, Marco, Ghidini, Chiara, Serafini, Luciano, Kump, Barbara, Pammer, Viktoria, Lindstaedt, Stefanie, Faatz, Andreas, Ley, Tobias: Collaborative Enterprise Integrated Modelling. In : Semantic Web Applications and Perspectives, 5th Italian Semantic Web Workshop (SWAP 2008), FAO UN, Rome (2008)
18. Daga, Enrico, Presutti, Valentina, Salvati, Alberto: Ontologydesignpatterns.org and Evaluation WikiFlow. In : Semantic Web Applications and Perspectives, 5th Italian Semantic Web Workshop (SWAP 2008), FAO UN, Rome (2008)
19. Gangemi, Aldo: Ontology Design Patterns for Semantic Web Content. In Gil, Yolanda, Motta, Enrico, Benjamins, V., Musen, Mark, eds. : Proceedings of the Fourth International Semantic Web Conference, Galway, Ireland (2005)
20. Dellschaft, Klass., Engelbrecht, Hendrik, MonteBarreto, José, Rutenbeck, Sascha, Staab, Steffen: Cicero: Tracking design rationale in collaborative ontology engineering. In : Proceedings of the ESWC 2008 Demo Session (2008) pdf: http://www.uni-koblenz.de/~klaasd/Downloads/papers/Dellschaft2008CTD.pdf.
21. Tempich, C., Simperl, E., Luczak, M., Studer, R., Pinto, H.: Argumentation-based ontology engineering. IEEE Intelligent Systems 22(6), 52–59 (2007)
22. Gangemi, Aldo, Lehmann, J., Presutti, Valentina, Nissim, Malvina, Catenacci, C.: C-ODO: an OWL metamodel for collaborative ontology design. In Workshop on Social and Collaborative Construction of Structured Knowledge. In : WWW2007, Banff, Canada (2007)
23. Sebastian, A., Noy, N., Tudorache, T., Musen, M.: A generic ontology for collaborative ontology-development workflows. In : 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008), Catania, Italy (2008)
24. Gennari, John, Musen, Mark, Fergerson, Ray, Grosso, W, Crubézy, Monica, Eriksson, H., Noy, Natalya, Tu, Samson: The evolution of Protégé-2000: An environment for knowledge-based systems development. International Journal of Human-Computer Studies 58(1), 89–123 (2003) Protege.
25. Hayes, Patrick: RDF Semantics. In: World Wide Web Consortium - Web Standards. (Accessed 2009) Available at: http://www.w3.org/TR/rdf-mt/
26. Pazienza, Maria, Scarpato, Noemi, Stellato, Armando, Turbati, Andrea: Din din! The (Semantic) Turkey is served! In : Semantic Web Applications and Perspectives, Rome, Italy (2008)
27. Völkel, M., Enguix, C., Kruk, S., Zhdanova, A., Stevens, R., Sure, Y.: SemVersion - versioning RDF and ontologies. KnowledgeWeb Deliverable D2.3.3.v1, Institute AIFB, University of Karlsruhe (2005) Month: June.
28. Griesi, Donato, Pazienza, Maria, Stellato, Armando: Gobbleing over the Web with Semantic Turkey. In : Semantic Web Applications and Perspectives, 3rd Italian Semantic Web Workshop (SWAP2006), Scuola Normale Superiore, Pisa, Italy (2006) 18-20 December.
29. Kalyanpur, Aditya, Parsia, Bijan, Sirin, Evren, Grau, Bernardo, Hendler, James: Swoop: A web ontology editing browser. Web Semantics: Science, Services and Agents on the World Wide Web 4(2), 144-153 (2004)