CEUR Workshop Proceedings
http://ceur-ws.org
ISSN 1613-0073

Vol-596
urn:nbn:de:0074-596-3

ORES-2010
Ontology Repositories and Editors for the Semantic Web

Proceedings of the 1st Workshop on Ontology Repositories and Editors for the Semantic Web

Hersonissos, Crete, Greece, May 31st, 2010.

Edited by

Mathieu d'Aquin, **The Open University, UK**
Alexander García Castro, **Universität Bremen, Germany**
Christoph Lange, **Jacobs University Bremen, Germany**
Kim Viljanen, **Aalto University, Helsinki, Finland**

10-Jun-2010: submitted by Christoph Lange
11-Jun-2010: published on CEUR-WS.org

CONSISTOLOGY: A SEMANTIC TOOL TO SUPPORT ONTOLOGY EVOLUTION AND CONSISTENCY

Najla SASSI, Miracl Laboratory, sassinajla@yahoo.fr
Wassim JAZIRI, Miracl Laboratory,, wassim.jaziri@isimsf.rnu.tn
Faiez GARGOURI, Miracl Laboratory, faiez.gargouri@isimsf.rnu.tn

Abstract

Ontologies recently have become a topic of interest in computer science since they are seen as a semantic support to explicit and enrich data-models as well as to ensure interoperability of data. Moreover, supporting ontology's evolution becomes essential and extremely important, mainly when using ontologies in changing environments. An important aspect in the evolution process is to guarantee the consistency of the ontology when changes occur, considering the changes semantics. This paper proposes the Consistology tool developed to assist users in expressing evolution requirements and generating coherent ontology versions. This tool, based on coherent kits of change, has been experimented to evolve an ontology of education.

Key Words: Consistology, Semantic tool, Evolution kits, Ontology, Consistency.

1. Introduction

Changing environments require ontologies adaptable to changes that occur over time. The adaption of an ontology is a complex process and several evolution problems must be treated, in particular maintaining the ontology consistency after changing.

The application of a change on ontological entities is a modification of a subset of knowledge represented in the ontology. Change management requires defining mechanisms specifying how knowledge can be changed and how to maintain the consistency of knowledge after each change. In addition, ontological entities are semantically and conceptually linked, the application of a change in some ontological entities may have effects on other entities.

We are interested in this paper in defining evolution kits to allow updating ontologies while preserving their consistency. We also developed an ontology evolution tool 'consistology' to assist users in expressing evolution requirements and generating coherent ontology versions.

This paper is structured as follows. Section 2 presents an overview about the most representative approaches and tools used in ontology evolution. In Section 3, we propose our approach to support ontology evolution and to anticipate inconsistencies. Sections 4, 5 and 6 present the Consistology tool and its application to the education domain. Section 7 concludes this work.

2. State of the art

Several application areas are especially concerned with evolution of data and users requirements, such as software development [RL05], temporal databases [BB08] and ontologies.

Software systems are rarely stable following initial implementations. They have complex structures which are likely to continually undergo changes during their lifetime. Temporal databases support time-varying information and maintain the history of the modelled data. They allow the maintenance of data histories through the support of time semantics at system level. We refer to [BB08] [SBJ+10] for further information about related work on software development and temporal databases.

Ontologies, like software development and temporal databases, need to change every time the modelled real world has changed. Ontology evolution is the process of adaptation of ontology to evolution changes and the consistent management of these changes to guarantee the consistency of ontology when changes occur [KF01] [NK04]. It encompasses the set of activities, both technical and managerial, which ensures that ontology continues to meet organizational objectives and users needs in an efficient and effective way [Sto04]. According to [Sto04], "Ontology Evolution is the timely adaptation of ontology to the arisen changes and the consistent propagation of these changes to dependent artifacts." It concerns different aspects: the needs to update and to evaluate data, the changes to apply in conformity with these needs, the management of inconsistencies in all parts of the ontology as well as in the dependent artifacts.

According to [MS03], two types of inconsistency can be identified:

- Structural inconsistency occurs when the constraints of the ontology model are invalid or if the semantics of the subjacent language of ontology is not respected.
- Semantic inconsistency occurs when the significance of the entities of ontology is changed.

An ontology is considered consistent if its axioms are respected and if it satisfies the whole of the invariants defined in the model of ontology [MS03].

Stojanovic et al. [SSG+03] proposed an approach for the management of evolution and the maintaining of consistency for KAON ontologies. The authors proposed the concept of strategies of evolution which allow choosing the most suitable solutions for the resolution of inconsistencies

Haase et al. [HS05] also used the concept of strategies of resolution based on the constraints of OWL-Lite for the detection and the resolution of inconsistencies in OWL ontologies. However, the resolution of inconsistencies is done after application of changes. It is ensured in two phases: the detection of inconsistencies which consists in finding the parts of ontology which do not satisfy the consistency conditions and

the generation of changes that allow ensuring the consistency of ontology by generating additional changes.

Flouris et al. [FP05] differentiate between a consistent ontology and a coherent ontology. Ontology is inconsistent if there is no interpretation which satisfies all the axioms of this ontology. It is incoherent if it does not satisfy some predefined constraints or the related invariants. The predefined constraints describe the consistent model of ontology. These authors consider the inconsistencies as sign of bad design and their correction does not relate to the ontology evolution but it is rather related to the ontology design.

Luong et al. [LD07] distinguish two levels of consistency for the model of ontology: structural consistency and logical consistency. Structural consistency relates to the constraints of consistency defined for an ontology model by ensuring a good organization of the ontological entities at the level of structure. Logical consistency checks if the elements of ontology remained "semantically correct" after their evolution.

In [KJL09], the authors investigate how ontologies developed for use in Semantic Web technology could be used in checking the consistency of requirements specifications. They use reasoning which is a part of ontology. The *TESSI* tool has been developed.

Djedidi et al. [DA10] proposed an approach of enrichment of ontology with an aim of optimizing and automating the management of changes while ensuring the consistency and the quality of ontology after evolution.

The maintenance of consistency is ensured through alternatives of resolution of inconsistency. A model of quality is defined and applied to guide the resolution of inconsistencies and to evaluate the impact of the suggested alternatives on the quality.

A prototype of the change management system was implemented to manage changes of OWL ontologies while maintaining their consistency and quality.

In addition, number of scientific and commercial tools for creating, managing and updating ontologies have been used to build applications in several domains such as KAON [OVM+04], OntoView [KFK+02], OntoManager [SSG+03], TextToOnto [MV01], SHOE [HH00], PromptDiff [NM02], Protégé[1], etc. Some tools dedicated to ontology debugging are also proposed, such as RADON [JHQ+09], SWOOP [KPS+05], DION [2], OntoClean [3], MUPSter [SC03] etc. Other tools, such as ConsVISor[4], do both consistency checking and debugging. A comprehensive survey on ontology editors and tools can be found in [Den09] [GM03].

---

[1] http://protege.stanford.edu/.
[2] http://wasp.cs.vu.nl/sekt/dion/.
[3] http://www.ontoclean.org/.
[4] http://projects.semwebcentral.org/projects/consvisor/.

The analysis of related work shows that no complete framework for managing ontology coherence is proposed since they do not take into account all steps of the ontology life cycle. The majority of works conducted so far in the field of ontologies has focused on ontology construction issues. These works assume that the domain knowledge encapsulated in ontology does not change over time. Indeed, in dynamic environments, the domain knowledge evolves continually due to: the evolution in the application domain, additional functionalities to add to the system, new requirements of users, needs to better organize and model the information system etc.

Most of existing systems related to the ontology evolution provide only one possibility for realizing a change, and this is usually the simplest one. For example, the deletion of a concept always causes the deletion of all its sub concepts. It means that users are not able to control the way changes are performed (supervision).

In this work, we aim to propose an evolution tool which allows taking into account all relationships and offers a great level of expression. In addition, the approaches proposed in the literature are based on the correction of inconsistencies after they occur. We propose in this paper an anticipatory approach to manage inconsistencies before they occur. We express the requirements of evolution using types of changes. For each type of change, we define corrective operations that must be applied in conjunction with this type of change in order to correct consistencies.

3. An approach based on coherent evolution kits

The identification of types of changes to apply on the ontology formally expresses the needs of evolution required by users. The types of changes allow users expressing the requirements of evolution. When they are applied, the ontology changes from a current version to another one. However, the application of a type of change can cause inconsistencies on the new ontology version. In fact, types of change ensure only the modification of ontology. They do not guarantee that the ontology remains coherent after modifications.

To ensure the consistency of an ontology after evolution, we propose to anticipate inconsistencies that can be generated by each type of change in order to propose alternatives to address these inconsistencies [Jaz09]. Thus, we defined coherent evolution kits. A coherent evolution kit is composed of a type of change and corrective operations that allow correcting the potential inconsistencies caused by the considered change. The role of corrective operations is to correct inconsistencies by proposing additional changes to be applied by the system in combination with the initial type of change required by users. If several possibilities exist, i.e., various corrective operations may be applied with different effects, the ontology engineer has to choose to implement the adequate corrective operation. Each type of change in addition to the corrective operations forms a "coherent evolution kit" that must be

applied in full. We refer to [JSG10] for more details about the evolution kits of change.

4. Consistology: a tool to ensure consistency of ontologies

In a collaborative setting, given some changes to do on the ontology, users must be able to: (1) apply changes on the ontology; (2) examine the effects of changes visually; and (3) accept or reject changes.

Due to the lack of tools providing an efficient automatic support for ontology evolution, the development of an automatic tool is very useful to maintain uniformity and consistency of ontologies. We developed the *Consistology* tool, based on Java and Eclipse, to serve as an efficient automatic support for ontology evolution. Changes on the ontology are performed using elementary and composite changes. The application of elementary and composite changes on the initial ontology allows generating a new ontology version (Figure 1).

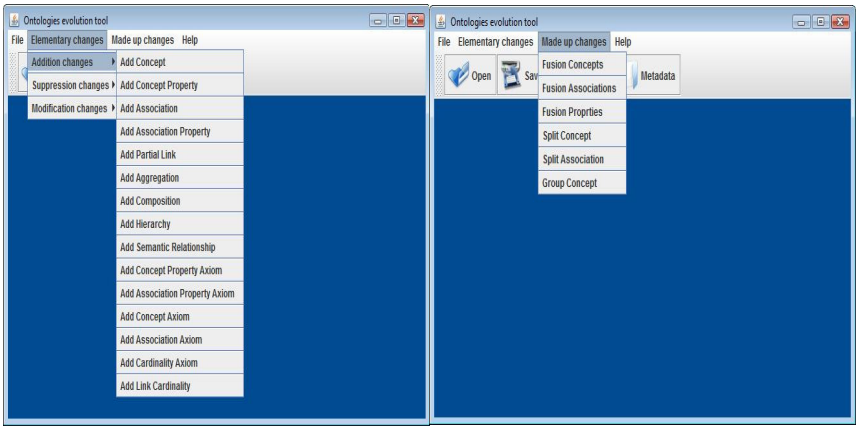

Figure 1. The Consistology tool allows applying elementary and composite changes.

The developed Consistology tool incorporates all actors (expert, ontology designer, system, user) in the evolution process. The ontology evolution process is initialized by the ontology designer and the expert, started by the user and guided by the system. The ontology designer initializes the process of evolution by introducing the ontology file and defining the metadata related to the semantic relationships. The expert defines the metadata related to the key concepts of the domain of study.

The user expresses evolution requirements using types of changes provided by the system which controls the required changes and applies the corresponding evolution kits of change in order to ensure the ontology consistency.

5. Application of Consistology to the Education domain

We present in this section an application of the developed Consistology tool to update an ontology of education related to the Tunisian higher education system.

The Tunisian higher education system is continually subject to changes to comply with social, economic and political strategies. Actually, it migrates from the old classical system toward a BMD (Bachelor's, Master's, Doctorate) system. The transition from the classical to the BMD system will certainly leave questions especially to students who followed their teachings within the old system. To provide satisfactory answers to these questions, it is necessary to understand and model the classical and the BMD systems as well as the transition between them. The modeling of this transition is also useful for the reuse of the current education system in case of future evolutions.

The modeling of the Tunisian education system requires a formal representation of knowledge. We use the ontology to explicit the semantics of the education domain and to model the classical and the BMD education systems [SJG09b]. The ontology of the BMD education system is an evolved version of the ontology related to the classical system. The evolution requires applying types of changes in order to adapt the old education ontology and to create a new ontology version adapted to the BMD system. We ensure the evolution of ontology based on primitive and complexes operators.

The acquisition of knowledge related to the education system is based on the analysis of technical documents and instruction manuals provided by the Ministry of higher education as well as interviews with experts of the domain. The ontology construction is done using *Protégé*.

We present in the following, an extract from the initial ontology of education according to the OWL syntax:

```
<owl:Ontology rdf:about=""/>
<owl:Class rdf:ID=" MEDICINE ">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="SECTOR"/>
  </rdfs:subClassOf>
 </owl:Class>
<owl:Class rdf:ID="TECHNOLOGY_SCIENCES ">
  <rdfs:subClassOf>
   <owl:Class rdf:ID=" SECTOR"/>
  </rdfs:subClassOf>
 </owl:Class>
<owl:Class rdf:ID="HUMAN_SCIENCES ">
  <rdfs:subClassOf>
   <owl:Class rdf:ID=" SECTOR"/>
  </rdfs:subClassOf>
 </owl:Class>
<owl:Class rdf:ID="ECONOMICS_MANAGEMENT ">
  <rdfs:subClassOf>
   <owl:Class rdf:ID=" SECTOR"/>
  </rdfs:subClassOf>
 </owl:Class>
<owl:Class rdf:ID=" PHYSICAL_EDUCATION ">
  <rdfs:subClassOf>
   <owl:Class rdf:ID=" SECTOR"/>
  </rdfs:subClassOf>
 </owl:Class>
. .
<owl:Class rdf:ID="DOCTORATE">
  <rdfs:subClassOf>
   <owl:Class rdf ID="DIPLOMA"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="PROFESIONAL_ MASTER ">
  <rdfs:subClassOf>
   <owl:Class rdf ID="MASTER"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="MAITRISE">
  <rdfs:subClassOf rdf:resource="#DIPLOMA"/>
 </owl:Class>
 <owl:Class rdf:ID="RESEARCH_MASTER">
  <rdfs:subClassOf>
   <owl:Class rdf about="#MASTER"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="TECHNICIAN">
  <rdfs:subClassOf rdf:resource="#DIPLOMA"/>
 </owl:Class>
 <owl:Class rdf:about="#MASTER">
  <rdfs:subClassOf rdf:resource="#DIPLOMA"/>
 </owl:Class>
. .
```

To express the evolution from the classical education system toward the BMD system, we apply operators of changes such as:

1. Add new concepts which exist only in the BMD system, such as: MENTION, OPTIONAL_UNIT, COURSE, OBLIGATORY_UNIT, LICENCE, MASTER1, MASTER2, EDUCATION_UNIT etc.
2. Add new relationships between concepts such as:
   - Equivalence: for example, an equivalence relationship is added between the concepts: TECHNICIAN and LICENCE, MAITRISE and MASTER1, etc.
   - Synonymy: for example, a synonymy relationship is added between the concepts: MODULE and EDUCATION_UNIT.

6. An illustrative example

We present in this section an example of application of an evolution kit: *Add_concept*. In this example, we aim to add a new concept '*LICENCE*' to a hierarchy of concepts in the ontology of the classical education system to evolve it towards to the BMD system.

The user introduces an initial ontology to update and selects the type of change to apply on the ontology, for example *Add_Concept* (Figures 2 and 3).

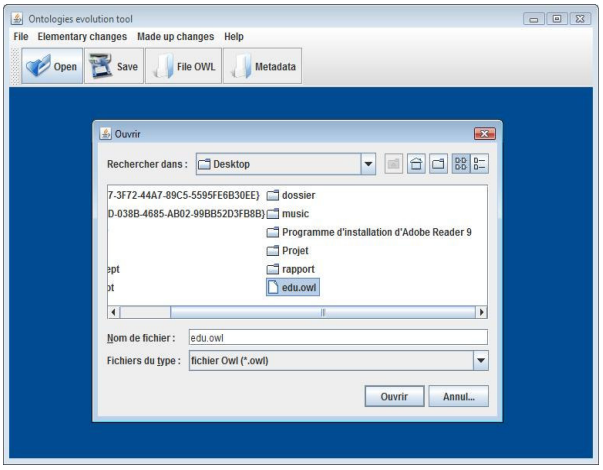

Figure 2. The input of Consistology is an ontology (e.g., owl file) to update.

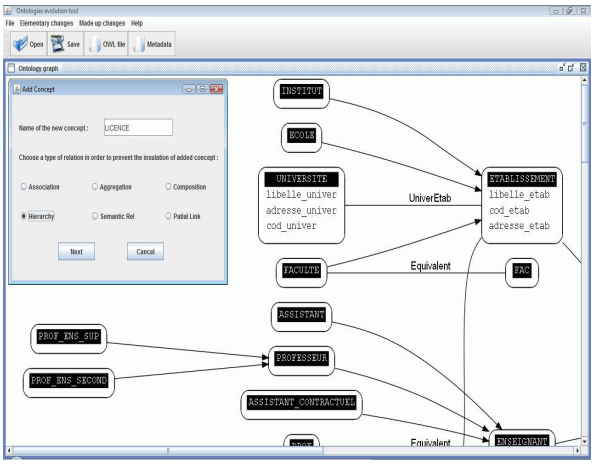

Figure 3. Add a new concept 'LICENCE' to the initial ontology.

The type of change *Add_Concept* generates inconsistencies related to an isolated and empty concept. To resolve the first inconsistency, the system automatically proposes to the user to add a new relationship between the added concept and another one in

the ontology. In this example, we chose to add a Hierarchy relationship between the concepts: *LICENCE* and *DIPLOMA* (Figure 4). Thus, since it is a hierarchy relationship, the concept *LICENCE* inherits the properties from the concept *DIPLOMA* and therefore the second inconsistency is resolved.

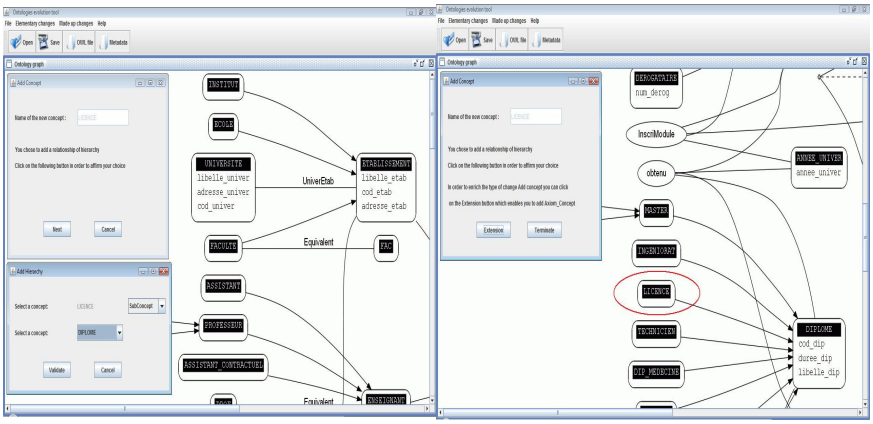

Figure 4. Add a new hierarchy relationship between the new concept LICENCE and another concept belonging to the ontology. The new concept LICENCE is added to the ontology as well as a hierarchy relationship between LICENCE and DIPLOMA.

In addition, the developed Consistology tool allows enriching the ontology by adding new axioms (Figure 5).

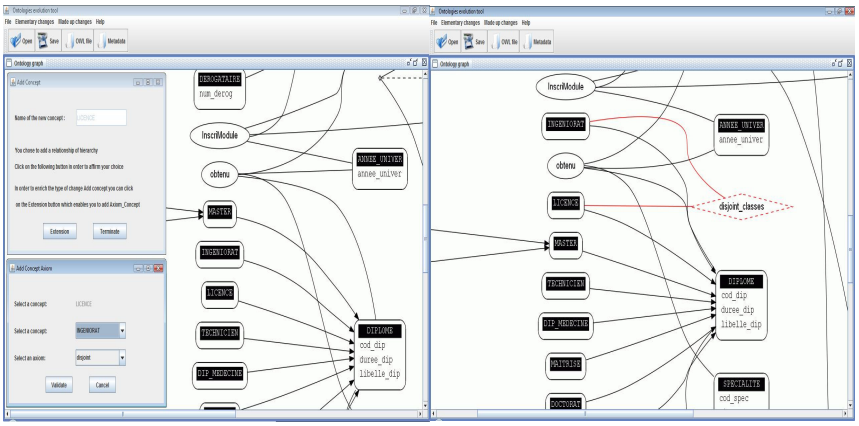

Figure 5. The tool allows enriching the ontology by adding new axioms (optional extensions). Example: A new axiom 'disjoint classes' is added between the concepts LICENCE and INGENIORAT.

The application of elementary and composite changes on the initial ontology allows generating a new ontology version (Figure 6). A historic file is created containing an ordered sequence of types of changes applied to the initial version.

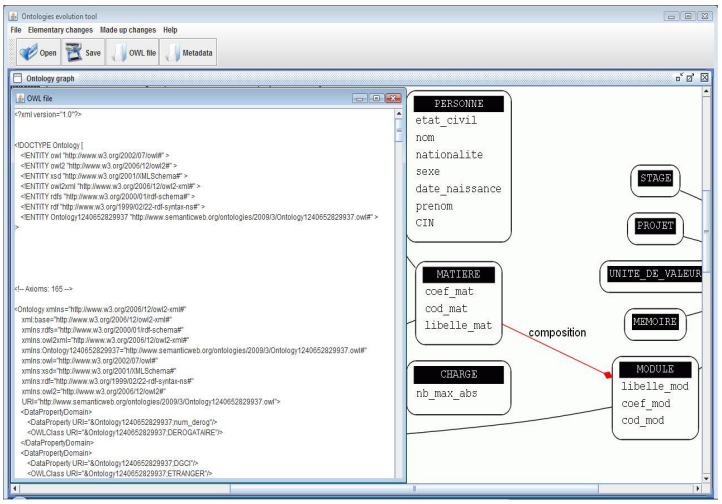

Figure 6. Consistology produces a new ontology version (OWL file) and a graph representing the ontology.

7. Conclusion and perspectives

Ontologies represent an explicit specification of a domain and serve as a support for providing and searching knowledge sources. They need to be modified to reflect new requirements and must remain coherent.

We express the requirements of evolution using types of changes. However, types of change allow updating ontology but do not ensure its consistency. The application of a type of change may produce inconsistencies on ontological entities. To correct them, corrective operations are defined and automatically done in addition to the type of changes.

An inconsistent ontology may be the consequence of a bad design or of the application of changes. We consider that the first case is rather a problem of ontology design and building. To maintain ontology consistency after applying types of changes, we developed a proactive approach to manage inconsistencies before they occur rather than managing them after evolution. This approach is based on evolution kits, defined to ensure the consistency of ontology after evolution. An evolution kit anticipates the inconsistencies that can generate each type of change in order to apply additional changes able to treat them. After the execution of a change, some corrective operations are automatically applied.

To implement types of changes, we developed the Consistology tool. Consistology is an ontology evolution support which allows users updating ontologies while preserving their consistency. It is based on elementary and composite changes that allow expressing the different possibilities of evolution requirements. Experimentation is presented, related to the evolution of the Tunisian higher education system. The *Consistology* tool is used to apply changes on the education ontology and to adapt it to new evolution requirements.

In future work, we aim to apply the developed system to other applications involving evolution changes. We will also add other functionalities to support versionning of ontology and to store and query various versions in an ontological database.

In fact, the problem of evolution and versioning is also present in other application areas, more especially in the context of databases systems. Dynamic schema evolution in databases is defined as managing schema changes in a timely manner without loss of existing data. Particular problems addressed are cascading changes (changes required to other parts of the schema as a result of a change), ensuring consistency of the schema, and propagation of the changes to the corresponding database.

Although there are significant differences between schema evolution and ontology evolution, many of the methods and technologies developed for schema evolution can be applied or adapted to ontology evolution. Our research in the ontology evolution can benefit from the many research works in database systems. Thus, we aim to exploit the techniques of databases to create versions of ontology and to incorporate additional functionalities in Consistology in order to allow representing, saving, evolving and accessing to ontology versions.

References

[BB08] Brahmia, Z., Bouaziz, R. (2008). Schema Versioning in Multi-Temporal XML Databases, Proceedings of the 7th IEE/ACIS International Conference on Computer and Information Science (IEEE/ACIS ICIS 2008), pages 158-164, Oregon, USA.

[DA10] Djedidi, R., Aufaure, M.A. (2010). ONTO-EVOAL an Ontology Evolution Approach Guided by Pattern Modeling and Quality Evaluation. FoIKS 2010: 286-305.

[Den09] Denny, M. (2009). Ontology Tools Survey. From http://www.xml.com/pub/a/2004/07/14/onto.html.

[FP05] Flouris, G., Plexousakis, D. (2005). Handling Ontology Change: Survey and Proposal for a Future Research Direction. Technical report FORTH-ICS/TR-362.

[GM03] Gómez-Pérez, A., Manzano-Macho, D. (2003). A survey of ontology learning methods and Techniques, Deliverable 1.5, Universidad Politécnica de Madrid.

[HH00] Heflin, J., Hendler, J. (2000). Dynamic Ontology on the Web, Proceedings of the Seventeenth National Conference on Artificial Intelligence AAAI/MIT, pages 443-449, CA.

[HS05] Haase, P., Stojanovic, L. (2005). Consistent Evolution of OWL Ontologies. In A. Gomez-Perez and J. Euzenat, editors, Proceedings of the 2nd European Semantic Web Conference (ESWC '05), volume 3532 of LCNS, pages 182–197. Springer.

[Jaz09] Jaziri, W. (2009). A methodology for ontology evolution and versioning, The Third International Conference on Advances in Semantic Processing (SEMAPRO'2009), pages 15-21, ISBN: 978-1-4244-5044-2, Sliema, Malta.

[JHQ+09] Ji, Q., Haase, P., Qi, G., Hitzler, P., Stadtmüller, S. (2009). RaDON: Repair and Diagnosis in Ontology Networks, Lecture Notes in Computer Science, pages 863-867, Volume 5554/2009, Springer Berlin-Heidelberg.

[JSG10] Jaziri, W., Sassi, N.,Gargouri F. (2010). Approach and tool to evolve ontology and maintain its coherence, International Journal of Metadata, Semantics and Ontologies, Inderscience Publishers, United Kingdom.

[KF01] Klein, M., Fensel, D. (2001). Ontology versioning on the Semantic Web, In Proceedings of the 1st Semantic Web Working Symposium, Stanford, CA, USA.

[KFK+02] Klein, M., Fensel, D., Kiryakov, A., Ognyanov, D. (2002). Ontology versioning and change detection on the web. Lecture Notes in Computer Science, pages 247-259, volume 2473, Springer.

[KJL09] Kroha, P., Janetzko, R., Labra, J.E. (2009). Ontologies in Checking for Inconsistency of Requirements Specification, The Third International Conference on Advances in Semantic Processing (SEMAPRO'2009), pages 32-37, Sliema, Malta.

[KPS+05] Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J. (2005). Debugging unsatisfiable classes in owl ontologies, Journal of Web Semantics, volume 3(4), pages 268-293.

[LD07] Luong, P-H., Dieng-Kuntz, R. (2007). A Rule-based Approach for Semantic Annotation Evolution, The Computational Intelligence Journal, 23(3): 320-338, USA.

[MS03] Maedche A., Staab S. (2003), Ontology Learning. In S. Staab & R. Studer (eds.), Handbook on Ontologies in Information Systems, pages 173-190, Springer.

[MV01] Maedche, A., Volz, R. (2001). The Text-To-Onto Ontology Extraction and Maintenance System, Workshop on Integrating Data Mining and Knowledge Management co-located with the first International Conference on Data Mining, San Jose, California, USA.

[NM02] Noy, N., Musen M. (2002). Promptdiff: a fixed-point algorithm for comparing ontology versions. In Proceedings of the 18th National Conference on Artificial Intelligence, pages 744-750, Canada.

[NK04] Noy, N., Klein, M. (2004). Ontology evolution: Not the same as schema evolution, Knowledge and Information Systems, 6(4):428-440.

[OVM+04] Oberle, D., Volz, R., Motik, B., Staab, S. (2004). An extensible ontology software environment, In: Steffen Staab, Rudi Studer (Eds.), Handbook on Ontologies, pages 311–333, Springer, Berlin.

[RL05] Robbes, R., Lanza, M. (2005). Versioning systems for evolution research, In Proceedings of the 8th International Workshop on Principles of Software Evolution (IWPSE 2005), pages 155-164, Lisbon, Portugal.

[SBJ+10] Sassi, N., Brahmia, Z., Jaziri, W., Bouaziz, R. (2010). From Temporal Databases to Ontology Versioning: An Approach for Ontology Evolution, Ontology Theory, Management and Design: Advanced Tools and Models, Ed. Faiez Gargouri and Wassim Jaziri, IGI-Global, USA (to appear, March 2010).

[SC03] Schlobach, S., Cornet, R. (2003). Non-standard reasoning services for the debugging of description logic terminologies. In Proceedings of the 18th International Joint Conference on Artificial Intelligence, pages 355-362, Acapulco, Mexico.

[SSG+03] Stojanovic, L., Stojanovic, N., Gonzalez, J., Studer, R. (2006). Ontomanager - a system for the usage-based ontology management", In Proceedings of ODBASE'2003, pages 858-875, Springer.

[Sto04] Stojanovic, L. (2004). Methods and Tools for Ontology Evolution. PhD Thesis, University of Karlsruhe.