Characterizing Modular Ontologies

Sarra Ben Abbès[1], Andreas Scheuermann[2], Thomas Meilender[3], and Mathieu d'Aquin[4]

[1]Laboratoire d'Informatique de Paris Nord (UMR 7030), CNRS, Paris 13 University, Sorbonne Paris Cité, France. E-mail: sarra.benabbes@lipn.univ-paris13.fr
[2]University of Hohenheim, Information Systems 2, 70599 Stuttgart, Germany. Email: andreas.scheuermann@uni-hohenheim.de
[3]UHP-Nancy 1, LORIA (UMR 7503 CNRS-INPL-INRIA-Nancy 2-UHP), France. Email: thomas.meilender@loria.fr
[4]Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes, MK7 6AA, UK. E-mail: m.daquin@open.ac.uk

Abstract. Since large monolithic ontologies are difficult to handle and reuse, ontology modularization has attracted increasing attention. Several approaches and tools have been developed to support ontology modularization. Despite these efforts, a lack of knowledge about characteristics of modularly organized ontologies prevents further development. This work aims at characterizing modular ontologies. Therefore, we analyze existing modular ontologies by applying selected metrics from software engineering in order to identify recurring structures, i.e. patterns in modularly organized ontologies. The contribution is a set of four patterns which characterize modularly organized ontologies.
Keywords: modularization, patterns, modules, ontology.

1 Introduction

Difficulties in reusing and maintaining large monolithic ontologies have resulted in an increasing interest in modularizing ontologies. In the past, several approaches and tools (e.g., SWOOP[1], NeOn Toolkit[2]) have been proposed to support the modularization of ontologies. Each of these approaches and tools respectively incorporates its own definition and notion of modular ontologies and criteria underpinning ontology modularization [5]. This proliferation is mainly due to the fact that the area of ontology modularization appears to be still in its infancy. Thus, it lacks the mature, well-defined, well-understood, and commonly agreed upon definitions and concepts as sociated with modularization in the area of software engineering [19]. This lack of knowledge about ontology modularization and, particularly, the lack of knowledge about characteristics of modular ontologies prevents its further development. Being able to characterize modular

[1] `http://www.mindswap.org/2004/`
[2] `http://neon-toolkit.org/wiki/Main_Page`

ontologies would allow for (1) comparing different modularization approaches, (2) assessing the quality of modular ontologies with respect to manually modularized ontologies, (3) customizing ontology modularization by providing modularization criteria, and (4) improving the area of ontology modularization as a whole.
The goal of this work is to characterize modularly organized ontologies to contribute to a better understanding of ontology modularization. For this purpose, we (1) extract a number of existing ontologies from the web, (2) select and adopt metrics from software engineering in order to both (3) identify recurring structures (patterns) in modularly organized ontologies and (4) characterize the identified patterns. The contribution is a set of four patterns, which characterize modularly organized ontologies.
The rest of this paper is organized as follows: Section 2 provides an introduction to ontology modularization and reviews related work. Section 3 reports on the research design whereas section 4 presents and discusses the results. Section 5 draws a conclusion and points to future avenues of research.

2 Ontology Modularization

2.1 Modular Ontologies

The main idea of modular ontologies originates from the general notion of modular software in the area of software engineering. Correspondingly, ontology modularization can be interpreted as decomposing potentially large and monolithic ontologies into (a set of) smaller and interlinked components (modules). Therefore, an ontology module can be considered as a loosely coupled and self-contained component of an ontology maintaining relationships to other ontology modules. Thereby, ontology modules are themselves ontologies [4].
In general, ontology modularization aims at providing users of ontologies with the knowledge they require, reducing the scope as much as possible to what is strictly necessary. In particular, ontology modules (1) facilitate knowledge reuse across various applications, (2) are easier to build, maintain, and replace, (3) enable distributed engineering of ontology modules over different locations and different areas of expertise, and (4) enable effective management and browsing of modules [12].

2.2 Approaches for Ontology Modularization

In recent years, the problem of ontology modularization has attracted more and more attention and, thus, several different approaches for modularizing ontologies appeared. These approaches can be classified in two main categories.
The first main category comprises approaches that focus on the composition of existing ontologies by means of integrating and mapping ontologies. On the one hand, approaches addressing integration of existing ontologies are owl:import, partial semantic import, e.g., [8, 5], package-based description logics, e.g., [2].

On the other hand, mapping approaches basically aim at (inter-)linking sets of ontology modules. The following approaches can be assigned to these two formalisms: distributed description logics, e.g., [17, 3], ε-connections, e.g., [15, 10]. Other approaches establish the relationship between various modular ontology formalisms [9] in order to have special syntax in the ontology languages for a modeling perspective.
The second main category comprises approaches for modularizing ontologies in terms of ontology partitioning and ontology module extraction. On the one hand, ontology partitioning aims at splitting up an existing ontology into a set of ontology modules. Approaches for partitioning ontologies are proposed by [13, 11] whereas [18] proposes a tool. On the other hand, ontology module extraction, which is also called segmentation [16] or traversal view extraction [14], aims at reducing an ontology to its relevant sub-parts. Approaches for ontology module extraction are the subject of [14, 16], and the PROMPT tool [14]. More details of this category of approaches are discussed in [5].

2.3 Criteria for Ontology Modularization

Criteria for modularizing ontologies generally aim at characterizing modular ontologies. To the best of our knowledge, only [5] explicitly deals with criteria for ontology modularization. Therefore, [5] distinguishes between criteria originating in software engineering, logical criteria, local criteria, structural criteria, quality of modules, and relations between modules. First, criteria from software engineering comprise encapsulation and coherence whereas logical criteria include local correctness and local completeness. Structural criteria, which are also discussed by [6], focus on size and intra-module coherence. It is proposed to determine the quality of modules in terms of module cohesion, richness of the representation, and domain coverage. At least, to assess the relation between modules the criteria of connectedness, redundancy, and inter-module distance can be applied. Against this background, the evaluation of ontology modularization respectively applies a subset of the proposed set of criteria with respect to different scenarios and ontology modularization techniques.
Based on best practices in Ontology Engineering, ontology design patterns (ODPs) simplify ontology design by providing a "modelling solution to solve recurrent ontology design problems" [1]. Several types of ODPs has already been identified, e.g., logical patterns that are used to solve problems of expressivity, or naming patterns that are conventions for naming elements. Among these types, architectural ontology design patterns (AODPs) aim at describing the overall shape of the ontology. More precisely, external AODPs describe the modular architecture of an ontology by considering a modular ontology as an ontology network. Involved ontologies are considered as modules and are connected by the import operation. A semantic web portal[3] has been proposed as a repository for ODPs. Unfortunately, to our knowledge, no work has been done on proposing and describing external AODPs.

[3] `http://ontologydesignpatterns.org`

3 Approach

In order to characterize reoccurring structures in modularly organized ontologies, the following approach establishes a methodological basis to guide the research program of this work. This approach comprises six subsequent steps:

1. **Search step**: the goal of the first step is to gather modularly organized ontologies. We use the Semantic Web gateway Watson[4] to search for available modular ontologies from the WWW. The search query focused on import-relationships between ontologies covering the same domain. The result is a set of 77 modularly organized ontologies.
2. **Cleaning up step**: the second step aims at cleaning up the initial search results in order to establish a thorough basis for further experiments. This is necessary because the set of 77 modular ontologies is afflicted with redundancies and incompleteness. This results in a set of 38 modular organized ontologies constituting a thorough basis for characterizing ontology modularization.
3. **Selection of metrics step**: the third step selects a set of appropriate metrics to characterize modularly organized ontologies. The modular ontologies could be described by various indicators such as the distribution of classes, the network of links between modules, the number of internal links in modules, etc. In general, the literature provides a plethora of various metrics, which could be applied for characterising modular ontologies. As a starting point, this work focuses on metrics originating in the area of software engineering, due to its maturity. In particular, this work adopts the following metrics from software engineering, which are easier to compute, in order to characterize modular ontologies [7]: (i) **size of the module**: the number of classes and properties (object and datatype properties), (ii) **cohesion of the module**: this metric is a value which is between 0 and 1 and is specified as follows:
   * Hierarchical Class Cohesion (HCC): the number of direct and indirect hierarchical class links.

     $$HCC = \frac{2*(NdHC+NidHC)}{NC^2-NC}$$

     where: $NdHC$: Number of direct Hierarchical relationships between Classes, $NidHC$: Number of indirect Hierarchical relationships between Classes, and NC: Number of Classes.
   * Role Cohesion (RC): the number of direct and indirect hierarchical role links.

     $$RC = \frac{2*(NdR+NidR)}{NRoles^2-NRoles}$$

     where: NdR: Number of direct roles between Classes, $NidR$: Number of indirect roles between Classes, and $NRoles$: Number of Roles.

[4] `http://watson.open.ac.uk/`

* Object Property Cohesion (OPC): the number of classes which have been associated through the particular object property (domain and range).

$$OPC = \frac{2*\sum_{i=1}^{NRoles} NdC(r_i)*NrC(r_i)}{NRoles*(Nc^2-NC)}$$

where: $NdC(r_i)$: Number of ontology Classes in the domain of the role r_i, $NrC(r_i)$: Number of ontology Classes in the range of the role r_i, NC: Number of Classes, and $NRoles$: Number of Roles.

The cohesion measure is computed as follows:

$$Cohesion = \frac{\alpha*HCC+\beta*RC+\delta*OPC}{\alpha+\beta+\delta}$$

where: α, β and δ specify the impact of each type of hierarchical class, role or object property cohesion. In our case, we choose $\alpha = \beta = \delta = 1$.

(iii) **coupling of the module**: it takes an estimation of the inter-dependency of different modules and is specified as follows:

* Hierarchical class dependency (HCD): the number of all direct and indirect hierarchical class relationships to foreign ontologies.

$$HCD = \frac{1}{2} * (\frac{NedHC}{NdHC} + \frac{NeidHC}{NidHC})$$

where: $NedHC$: Number of direct Hierarchical class dependencies between local classes and external classes, and $NeidHC$: Number of indirect Hierarchical class dependencies between local classes and external classes.

* Hierarchical role dependency (HRD): the number of all direct and indirect hierarchical role relationships to foreign ontologies.

$$HRD = \frac{1}{2} * (\frac{NdHR}{NdHR} + \frac{NeidHR}{NidHR})$$

where $NdHR$: Number of direct roles dependencies between local classes and external classes, and $NeidHR$: Number of indirect roles dependencies between local classes and external classes.

* Object property dependency (OPD): the number of roles that associate external classes to local ones.

$$OPD = \frac{NeRoles}{NRoles}$$

where: $NeRoles$: Number of all roles that have an external class in their domain or range, $NRoles$: Number of all existing roles in the ontology.

* Axiom dependency (AD) : a role or a class is associated to an external ontological element through an inclusion axiom.

$$AD = \frac{\sum_{i=1}^{NAxioms} externalAssociationNumber(axm_i)}{\sum_{i=1}^{NAxioms} LS(axm_i*RS(axm_i)}$$

where: $LS(axm)$: the size of the left sides of the axiom *axm*, $RS(axm)$: the size of the right sides of the axiom *axm*, $LSE(axm)$: the number of external elements in the left sides of the axiom *axm*, $RSE(axm)$: the number of external elements in the right sides of the axiom *axm*, and $externalAssociationNumber(axm)$: the number of all external ontological elements that have been associated through the axiom *axm* to internal elements. $externalAssociationNumber(axm) = LSE(axm_i)*RS(axm_i)$ + $LS(axm_i) * RSE(axm_i)$ - $LSE(axm_i) * RSE(axm_i)$.

The coupling measure is computed as follows:

$$Coupling = \frac{\alpha * HCD + \beta * HRD + \delta * OPC + \gamma * AD}{\alpha + \beta + \delta + \gamma}$$

where, in our case, $\alpha = \beta = \delta = \gamma = 1$

4. **Metrics implementation step**: the fourth step implements the selected metrics. The computation was performed by the OWL API[5] and the reasoner HermiT[6]. This step sets up the (technical) evaluation framework.
5. **Analysis step**: the fifth step is the analysis of the basic population of modularly organized ontologies.
6. **Result step**: the sixth step involves synthesis and discussion of the results from the analysis in order to characterize modular ontologies.

4 Results and Discussion

A set of four patterns, which characterize *M*odular *O*ntologies *MO*, are proposed using previous metrics (size, cohesion and coupling). This section presents and discusses the results and the characteristics of each kind of pattern.

4.1 Pattern type 1: 1 module importing n modules

Pattern 1 contains one module which imports n other modules. For instance (Figure 1), the module *WildNET.owl* imports several modules such as *Animal.owl*, *AnimalSighting.owl*, *BirdObservers.owl*, *Birds.owl*, etc. The pattern that we propose conforms to an **aggregation**. This pattern establishes a relationship between a single module and a set of modules in the same ontology. This link is unidirectional. Applying the size metric (Table 1), the first part of the ontol-

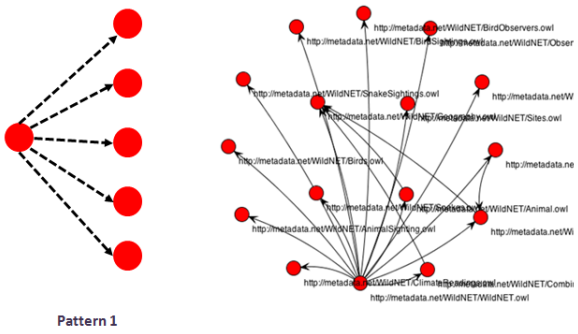

Fig. 1. 1 module importing n modules

ogy (one module) is very small (the module *WildNET.owl* contains 0 concepts)

[5] `http://owlapi.sourceforge.net/`
[6] `http://hermit-reasoner.com/`

and the second part (N modules) is not structured and is respectively bigger in size. Applying the cohesion and coupling metrics (see Table 1), Pattern 1 has a high cohesion compared to the coupling metric. We consider the pattern cohesion metric to be an indicator of the degree to which the elements in the module belong together. The idea of this pattern is that the concepts grouped in an ontology should be conceptually related for a particular domain in order to achieve common goals.

Metrics		Size	Cohesion	Coupling
MO_{11}	WildNET.owl	0	0,25	0
	SnakeSightings.owl	18331	1	0,25
	Snakes.owl	95	0,5	0,25
	Sites.owl	8	1	0,25
	Observers.owl	7	0,53	0,25
	Geography.owl	16	0,41	0,25
	Combined.owl	3497	0,67	0,12
	ClimateSensors.owl	255	0,61	0,6
	ClimateReadings.owl	63	0,5	0,25
	Climate.owl	24	0,61	0,15
	BirdSites.owl	437	1	0,25
	BirdSightings.owl	10401	1	0,5
	Birds.owl	1745	0,5	0,25
	BirdObservers.owl	303	0,5	0,25
	AnimalSighting.owl	26	0,66	0,25
	Animal.owl	9	0,82	0,11

Table 1. Results of Pattern 1 to n

4.2 Pattern type 2: n modules importing 1 module

Pattern 2 contains n modules, which respectively import one module. For instance (Figure 2), there are three independent modules importing one module, which containes general knowledge (*biopax-level1.owl*). The pattern that we propose corresponds to **inheritance**. This pattern establishes a correspondence between a set of modules and a single module in the same ontology. This correspondence is unidirectional. Applying the three metrics (Table 2), the first part of pattern 2 (n modules), *biopax-example-ecocyc-glycolysis.owl*, *biopax-example-Xwnt-b-catenin.owl* and *Xwnt-b-catenin_xref* have a high coupling metric with regard to the second part of the pattern (one module) *biopax-level1.owl*. This means that pattern 2 is characterized by the interconnections between modules. The degree of coupling depends on how complicated the connections are and on the type of connections between modules. As we can see, the second part of pattern 2 has a high cohesion because it encloses all other modules, which are strongly related.

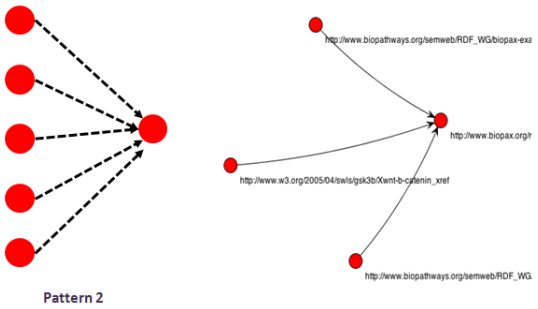

Fig. 2. n modules importing 1 module

Metrics		Size	Cohesion	Coupling
MO_{21}	biopax-example-ecocyc-glycolysis.owl	2139	0,20	0,72
	biopax-example-Xwnt-b-catenin.owl	236	0	0,2
	biopax-level1.owl	285	0,25	0,13
	Xwnt-b-catenin_xref	265	0	0,2

Table 2. Results of Pattern n to 1

4.3 Pattern type 3: n modules importing n-1 modules

Pattern 3 contains n modules, which import n-1 modules. For instance (Figure 3), we have three dependent modules: *dublincore.owl*, *terms.owl* and *dcmitype.owl*. The correspondence between modules is bidirectional. The distinguishing characteristic of Pattern 3 is that the n modules each import each other. Applying size,

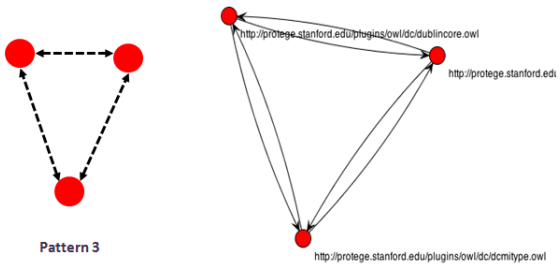

Fig. 3. n modules importing n-1 modules

cohesion and coupling metrics (Table 3), the module *dublincore.owl* has a small size (0 concepts) with regard to other modules *dcmitype.owl* and *terms.owl*. All modules have the same degree of relatedness of concepts (cohesion) 20%. The

coupling metric of the module *dublincore.owl* is null. In this case, pattern 3 is transformed to pattern 1 and has the same characteristics.

Metrics		Size	Cohesion	Coupling
MO_{32}	dcmitype.owl	21	0,20	0,02
	dublincore.owl	0	0,20	0
	terms.owl	47	0,20	0,25

Table 3. Results of Pattern n to n-1

4.4 Pattern type 4: Pattern mix

Pattern 4 combines all previous patterns (Patterns 1, 2 and 3). For instance (Figure 4), we find partterns 1 (5 * Pattern 1) and 2 (3 * Pattern 2). The proposed pattern is **pattern mix**. The correspondence can be undirectional and bidirectional. The major characteristic of this type of pattern is the highest

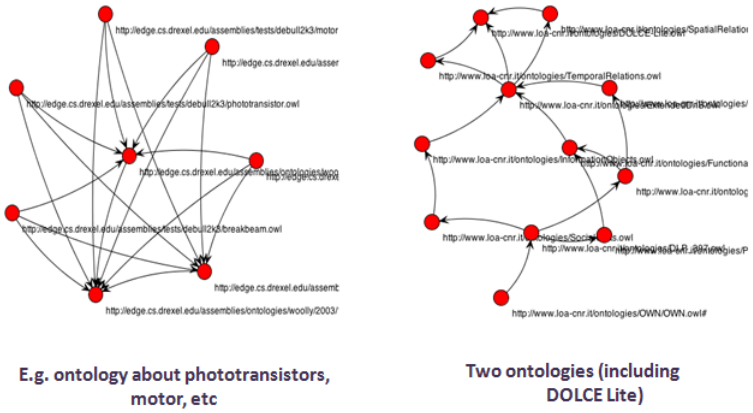

Fig. 4. Combination of patterns 1, 2, and 3

coupling metric with regard to the cohesion one. Two modular ontologies *iso-metadata* and *iso-19115*, have the same size, cohesion and coupling but they do not have a relationship like *import*.

4.5 Occurrence of Pattern Types

Having introduced and defined four types of patterns in order to characterize modularly organized ontologies, we consider how often these types of patterns

Metrics		Size	Cohesion	Coupling
MO_{41}	iso-metadata	2214	0,02	0,15
	iso-19108	159	0,08	0,16
	iso-19103	224	0,15	0,16
	iso-19115	2214	0,02	0,15

Table 4. Results of Pattern mix

respectively occur. Figure 5 provides an overview of the occurrence of the different types of patterns in a population of 38 modularly organized ontologies. It is

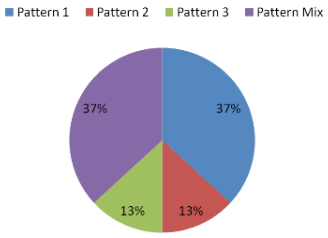

Fig. 5. Occurrence of Pattern Types

interesting to observe that pattern type 1 accounts for about 37%. The reason for this may be the fact that this type of pattern appears to be very intuitive. It could therefore be concluded that it implicitly constitutes the rationale underlying a large part of (semi-)automatic or manual approaches for modularizing ontologies. Similarly, pattern type 4 also accounts for about 37% of the basic population, i.e. 14 modularly organized ontologies. Pattern type 4 combines Pattern 1, Pattern 2, and Pattern 3. On the one hand, it is obvious that not all modularly organized ontologies have a rather straightforward structure, which could be easily characterized. This is especially true when assuming (semi-)automatic or manual modularizing approaches, which do not use clear and precisely defined criteria. And even when these criteria are clearly and precisely defined, the modularly organized ontologies could also have such a structure depending on the overall purpose of modularization. On the other hand, it is interesting to see that even more complex structures can (almost completely) be characterized by more simple and straightforward structural forms. Moreover, pattern type 2 and pattern type 3 equally account for about 13%, i.e. 5 modularly organized ontologies. This is particularly interesting because pattern type 2 is reasonable. This is due to the fact that it appears obvious that there exists an ontology that is of significance for several other ontologies. On the contrary, pattern type 3 is much less reasonable than pattern type 2. It is really hard to understand why ontology modules respectively import each other.

In this context, it can be observed that domain ontologies combine a clear structure and organization. This means that modularization of domain ontologies tend to rely on pattern type 1 or pattern type 2. In contrast, it appears that top-level ontologies (which represent relevant knowledge to a particular domain such as medical domain) have a less straightforward structure and organization particularly when compared to domain ontologies (which represent upper (generic) ontologies, covered the knowledge of many domain types such as Biomedical ontology, Dolce). An example for this is *dublincore.owl*, *terms.owl* and *dcmitype.owl*, which can be characterized by pattern type 3 (Figure 3).

5 Conclusion and Future Work

This work aims at characterizing modularly organized ontologies to contribute to a better understanding of ontology modularization. We introduced the notion of modular ontologies, reported on approaches for ontology modularization, and reviewed existing efforts to characterize modular ontologies. To characterize modular ontologies, we followed an approach comprising six consecutive steps. This approach mainly includes the extraction and selection of modular ontologies, the selection of a set of metrics from software engineering to analyse modular ontologies, and the evaluation of the analysis results. The evaluation results in a set of four patterns, which allow for characterizing the modular organization of ontologies. These patterns show amongst other things that modularly organized domain ontologies have a clear structure whereas top-level ontologies tend to have a rather confusing modular organisation.
In the future work, we aim at using firstly further Semantic Web gateways such as Falcons or Swoogle to identify and extract additional ontologies to gain a larger basic population. Second, extending the set of metrics and applying them to the ontologies should provide further insights to modular ontologies. Third, it would be interesting to create a comparison framework to conduct experiments with different modularization approaches, comparing them to each other or to manually modularized ontologies.

Acknowledgment

The authors would like to thank the organizers of Summer School on ontology engineering and the Semantic Web 2001 (SSSW'2011).

References

1. Gangemi A. and Presutti V. *Ontology Design Patterns*, pages 221–243. Springer, Berlin, 2009.
2. Jie Bao, George Voutsadakis, Giora Slutzki, and Vasant Honavar. Package-based description logics. In *Modular Ontologies*, pages 349–371. Springer, 2009.

3. Alexander Borgida and Luciano Serafini. Distributed description logics: Directed domain correspondences in federated information sources. In *On the Move to Meaningful Internet Systems*, pages 36–53, London, UK, 2002. Springer-Verlag.
4. Mathieu d'Aquin, Peter Haase, Sebastian Rudolph, Jérôme Euzenat, Antoine Zimmermann, Martin Dzbor, Marta Iglesias, Yves Jacques, Caterina Caracciolo, Carlos Buil Aranda, and Jose Manuel Gomez. D1.1.3: Neon formalisms for modularization: Syntax, semantics, algebra. deliverable 1.1.3, NeOn Integrated Project, 2008.
5. Mathieu d'Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Modular ontologies. chapter Criteria and Evaluation for Ontology Modularization Techniques, pages 67–89. Springer-Verlag, Berlin, Heidelberg, 2009.
6. Frederico Luiz Gonçalves de Freitas, Zacharias Candeias Jr., and Heiner Stuckenschmidt. Towards checking laws' consistency through ontology design: The case of brazilian vehicles' laws. *JTAER*, 6:112–126, 2011.
7. Faezeh Ensan and Weichang Du. *A Modular Approach to Scalable Ontology Development*, page 79. Springer Science+Business Media, 2010.
8. Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. A logical framework for modularity of ontologies. In *JCAI-2007*, 2007.
9. Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Extracting modules from ontologies: A logic-based approach. In *Modular Ontologies*, pages 159–186. 2009.
10. Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Working with multiple ontologies on the semantic web. In *International Semantic Web Conference*, pages 620–634. Springer, 2004.
11. Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Modularity and web ontologies. In *20th International Conference on Principles of Knowledge Representation and Reasonin*, pages 198–209. AAAI Press, 2006.
12. Mustafa Jarrar. *Towards Methodological Principles for Ontology Engineering*. PhD thesis, Vrije Universiteit Brussel, 2005.
13. Bill MacCartney, Sheila McIlraith, Eyal Amir, and Tomás E. Uribe. Practical partition-based theorem proving for large knowledge bases. In *Proceedings of the 18th international joint conference on Artificial intelligence*, pages 89–96, San Francisco, USA, 2003. Morgan Kaufmann Publishers Inc.
14. Natalya F. Noy and Mark A. Musen. Specifying ontology views by traversal. In *International Semantic Web Conference*, volume 3298/2004, pages 713–725. Springer Berlin, 2004.
15. Ana Armas Romero, Bernardo Cuenca Grau, and Ian Horrocks. Modular combination of reasoners for ontology classification. In *Description Logics*, 2012.
16. Julian Seidenberg and Alan Rector. Web ontology segmentation: analysis, classification and use. In *Proceedings of the 15th international conference on World Wide Web*, pages 13–22, New York, USA, 2006. ACM.
17. Luciano Serafini and Andrei Tamilin. Drago: Distributed reasoning architecture for the semantic web. In *ESWC*, pages 361–376. Springer, 2005.
18. Heiner Stuckenschmidt and Michel Klein. Structure-based partitioning of large concept hierarchies. In *International Semantic Web Conference*, pages 289–303, 2004.
19. Kevin J. Sullivan, William G. Griswold, Yuanfang Cai, and Ben Hallen. The structure and value of modularity in software design. In *Proceedings of the 8th European software engineering conference*, pages 99–108, New York, USA, 2001. ACM.