A Context-Specific Modularization for Ontology Change Management

Muhammad Raza Naqvi, Linda Elmhadhbi, Arkopaul Sakar, Da Xu, Mohammed Hedi Karray

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 207 (2022) 2578–2587

Procedia Computer Science

www.elsevier.com/locate/procedia

26th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2022)

A Context-Specific Modularization for Ontology Change Management

Muhammad Raza Naqvi [a,], Linda Elmhadhbi[b], Arkopaul Sakar[a], Da Xu[a], Mohammed Hedi Karray[a]

[a] *Laboratoire Génie de Production de l'École Nationale d'Ingénieurs de Tarbes (LGP-INP-ENIT), Université de Toulouse, CEDEX, 65016 Tarbes, France*

[b] *INSA Lyon, DISP EA4570, France*

Abstract

Knowledge engineering has a vital role in advancing the semantic web, in which ontologies play a key role in data interoperability and integration. One of the key issues in ontology engineering is how to handle the subsequent updates in the ontologies. A number of concerns need to be considered while working on the ontology change, such as, tracking ontology versions and heterogeneity issues. Ontology change management has been partially addressed by different researchers in overlapping research areas. However, a concrete description of the problem and its related concerns are still not available in the literature. Our work aims to present an overview of ontology change management and its concerns. We point up the need for modularization in ontology change management based on its advantages in the context of ontology reuse from different contextual viewpoints. For this purpose, we propose a protege plugin for reusing OWL modules, and allowing a safe/clean manual integration and reuse of different ontology modules.

Keywords: Ontologies; Modularization; Ontology Change; Semantic heterogeneity;

1. Introduction

As a branch of philosophy, ontology is the representation of "what exists", of the structures of the objects, events, properties, processes and their relations to all aspects of reality [1] [2]. In computer science, ontology is defined as a formal categorization of objects, events, properties, processes, and their relations. Ontology is a formal explicit specification of a shared conceptualization [3]. Explicit means that it is represented in machine-readable form with its classes, attributes, and relationships. The word conceptualization refers to an abstract model of a real-world phe-

* Corresponding author. Tel.: +3-376-833-6568.
E-mail address: syed_muhammad_raza.naqvi@enit.fr, razisyed4@gmail.com

10.1016/j.procs.2022.09.316

nomenon [4]. Ontology is a formal explanation of a common conceptual interest adapted for complex real-world conceptualization. Ontologies can be exploited by several logical reasoning mechanisms to deduce hidden knowledge and to provide machine-readable semantics so that they can be shared. Ontologies provide a common vocabulary to logical reasoning systems [5]. Reference ontologies are an emergent ontology kind that use to characterize deep knowledge of basic science in an upright way that allows them to be re-used in multiple ways, just as the basic sciences are re-used in different real-time applications. The position of upper-level reference ontologies with respect to other lower-level ontologies is shown in Fig.1

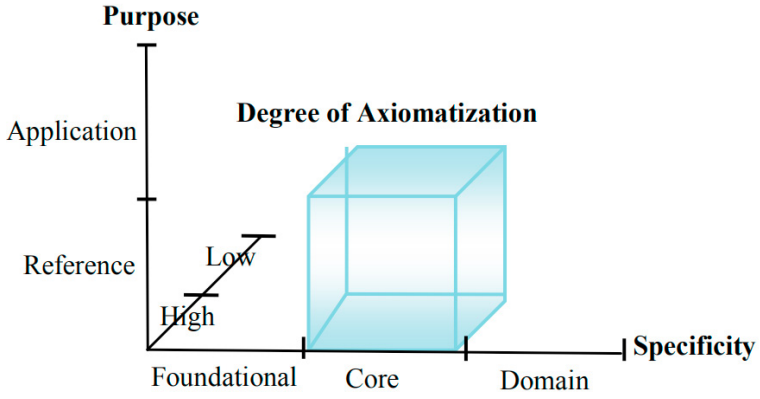

Fig. 1. Reference Ontology with other types.

There are different levels of abstractions of ontologies [6] in terms of their place in the reference hierarchy: top level ontology (e.g. Basic Formal Ontology[1] (BFO)), mid-level ontologies (e.g.Information Artifact Ontology, Ontology for Biomedical Investigations, and Spatial Ontology), and domain ontologies (e.g. Anatomy, Cell, Cellular component, Sub cellular, Sequence, Protein, Infectious Disease, Phenotypic Quality, Molecular Function, and Biological Process ontologies). Domain ontologies represent knowledge related to a particular domain that are generally used as metadata for information sources online. For example, WordNet is an online ontology developed by Princeton University and contains 100,000 words meanings in categories of nouns, verbs, adverbs, adjectives and function words and a relationship among these words [6] [7] [8] [9] [10]. These relationships can be based on similarity, differentiation, hierarchies, part-of, and morphological relationships.

Many practitioners of ontology engineering tried to formalize the development strategies of an ontology. The most common methodology for ontology development is 101 methodologies defined by Natalya et al. which include defining the scope of the ontology, considering the reuse of already existing ontologies and enumerating the essential terms in the ontology building taxonomy by creating classes and sub-classes [11]. The user plays different roles in the ontology development process including ontology reader, ontology editor, ontology checker and administrator.

In spite of the availability of numerous resources and methodologies for developing and maintaining ontologies, organisation and maintenance of ontologies in a distributed system is still problematic. One of the typical reason for such a problem is 'Ontology Proliferation',which is caused by the parallel development of too many ontologies in a single domain. This problem is exacerbated by the lack of quality standards and evaluation metrics for the ontologies. Furthermore, poor documentation of the ontologies often leads the users to interpret the terms of the ontologies differently from the original notion. However, all of these problems impact negatively ontologies' maintenance to tackle the changes in the ontology structure. Because ontologies are shared concepts, ontologies are built primarily to be reused by the community. Therefore, any change in one ontology affects many other ontologies in the network due to their internal dependencies. The detection of these interlinked parts that will require adjustment and managing the overall re-orientation task is cumbersome and demand an investigation in finding methodology and tools that may help developers manage them more effectively.

Our work focuses on identifying and analyzing the impact of ontology change at different levels and exploring the

[1] https://www.slideshare.net/BarrySmith3/the-six-category-ontology-basic-formal-ontology-and-its-applications

efficacy of modularization of ontologies in overcoming the challenges in ontology change management. In section II, we discuss ontology change by focusing on classification and coverage. Section III is about heterogeneity issues in the semantic web. Section IV discusses the theory of modularization. Finally, section V discusses modular ontology modelling and our vision regarding contextual integration of ontology modules.

2. Related Work

To manage the ontology change, various issues associated with ontology development, version, integration, planning or mapping may occur. These domains are interconnected and hold mutual component of sub-fields [12]. Thus, some exploration struggles and frameworks manage more than one subject of a statement and making it critical for beginners. This uncertainty is increased even more by the actuality that particular methods are normally used with different interpretations in related but not exactly equal research directions or conceptual frameworks [13] [14]. It describes the process of adapting the ontology to a certain requirement. Every kind of domain model including ontology like any structure which having information about a particular domain may require change as the understanding of the domain changes [15]. But, if we consider that the domain is static by most applications, this is an unrealistic assumption; we may have to determine the perception from which the domain is viewed [16], or might need to find the issues in the theory of the particular domain [17]. We might also wish to add new functionality according to the requirement of the users' change in requirement [18] [19], a discovery in the domain leading to the change in prior understanding of the domain, and some information, which was unavailable or new discovery in the certain domain that become known/available [20]. Due to all these inconsistencies that may occur, in this case, we need to take some action to deal with the inconsistency and in-coherency.

Furthermore, ontology engineering is becoming vastly a parallel and collaborative process. Due to that, some combinations may be needed to produce the final ontology [21] [22]. This collaborative ontology engineering process needs to be updated in every ontology version to reach the final and consistent version. There are also factors in relation to the distributed nature of knowledge technologies/semantic web ontologies. One ontology may depend on other ontologies and ontologists or knowledge engineers may not have complete control over these ontologies. If a remote ontology change due to issues discussed above, the dependent ontology may also need to be modified to accommodate the change in the remote ontology. In some cases, also the applications need to undergo changes to reflect these modifications, multiplying the effect of the change in the imported ontologies. Ontology change is the key issue in knowledge engineering as several philosophers discussed it by referring to such issues as belief change/revision in literature [23] [24] [25] [26] [27] [28]. Most of them can be considered as knowledge represented in ontologies [29] [30]. A large number of concepts, relations, constraints, and properties in modern-day ontology complicate the problem of ontology change even more [31].

One of the most critical problems is that once the ontology is publicly available and has been published, Furthermore, it is possible that after will have an impact on ontology-based services and applications.[32]. In some cases, the applications also need to undergo changes to reflect these modifications, multiplying the effect of the change in the imported ontologies ontologist/knowledge engineers may not understand the proper reason for such changes. It is also extremely difficult to build a system by predicting future changes in the used ontologies. On the other hand, developers of some ontologies cannot control the ways their ontology is imported and extended by other ontologies and thus cannot predict how any change in their ontology may affect the dependent ontologies [33]. Furthermore, due to heterogeneity, the lack of standard conceptualization or terminology of any given context, or domain, where users use two different ontologies, initiates the problem where service, application or multi-agent uses two dependent ontologies. Flouris et al. defined ontology changes in ten sub-fields. These sub-fields are grouped under the broad purposes of the changes (e.g. resolving heterogeneity of ontologies, modifying ontologies, and combining ontologies) [34].

2.1. Discussion on the heterogeneity of the semantic web

The issues presented by Flouris et al. require some types of translation that give us permission to accommodate the distinctions in syntax or terminology [34]. The main purpose of all these processes is to create ontologies is such a way that terms for similar concepts have some kind of mapping among them. For example, the ontology matching

algorithm [35] is able to recognize two terms like 'teacher' and 'teacher staff', which may show up in two distinct ontologies, actually refer to the same entity. Moreover, it should have the ability to differentiate this concept from other similar idea such as 'workshop chair'. Although these areas essentially manage similar issue (how to manage heterogeneous data), their solution techniques may vary from one domain to another. As a result, different study subjects may be formed based on the type of translation rules and the output provided by rules. Due to the strong link between these subjects, this technique is also called ontology alignment [36]. However, ontology alignment may be performed from many different point of views or requirements from the applications. Concept A from one ontology may be aligned to concept B from another ontology for a certain application, and to concept C for a different application. It is also often observed that strict alignment between ontologies may exacerbate the problem of ontology change management because the developer must resolve every discrepancy cascaded by these strong links to other ontologies. Therefore, ontologies must be partitioned in such a way that the changes are saved and can be easily identified.

3. Modularization

Modularization is highly popular in software engineering as a method of dividing a software system into separate and independent modules. Consequently, in the field of knowledge modeling, this technique has been receiving increasing attention. Some fundamental work on modularization is presented in form of a theorem. Farmer et al. encourage the use of combination of the "little theories" and present the particular mathematical equation to reason about the difficult problem [37]. The modular approach is beneficial if it is reusable and the struggle over modelling is decreasing. Lately, the knowledge engineering community have accepted the theory of reusing and integrating chunks of information instead of building information bases [38]. Amir et al. discuss that modularization of knowledge bases has also advantages for reasoning, even if the modularization is done a posteriori [39]. Algorithms are presented to break the current representation in different modules with insignificant connections and characterize reasoning methods for propositional and first-order logic [40] [41]. This work encourages by the posterior modularization technique where partitioning is performed after ontology is created so the time complexity is reduced [39].

Wang et al. propose a technique for modular execution of ontology utilising the description logics. This technique depends on different stages that are executed; after identifying an incoherence by a simple OWL reasoner, the origin of the incoherence can be identified [42]. Bucheit and others propose the same structure by distributing the terminology information based on taxonomies and also its view part [43]. They discuss that this differentiation could be utilized to accomplish the behaviour in better run-time and for difficult language views.

One common theme of these techniques is the use of a general ontology model to give a coherent representation of the world. However, this strategy fails to capture the pluralistic views of reality. Giunchiglia et al. propose radical techniques for distribution of representation, in which the local semantic models are proposed in addition to the global semantic one [44]. This set of local semantic models allows various modules to address different perspectives of the same reality while still maintaining a partial mappings among them. Previously, Serafini et al. characterized the distribution of the ontology versions on the basis of the local semantic model that is beneficial for contextual representation [45]. At present, in the field of Semantic Web, Ontologies are deployed in applications like search engines [46], integrated information systems and decision support systems[47] [48] [49][50].

To simplify the effort of building and maintaining ontology-driven systems through their life cycle, there is an unmistakable requirement for a representational and computational infrastructure and tools for creating and using Ontologies. Among many such concerns for ontology modularization, the following aspects demands special attention.

3.1. Distributed Systems

In distributed environments like the semantic web, the inquiry for modularization emerges normally. Ontologies built from different contexts lead to heterogeneity. Unhindered referencing of terms from remote Ontologies can be problematic without a clear understanding of the original elucidations for terms conceived by the authors of the remote ontologies.

3.2. Large Ontologies

Modularization isn't just attractive in dispersed conditions, it additionally assists in overseeing huge ontologies that we find in medication or science. These ontologies, which may contain more than a hundred thousand Concepts, are difficult to keep up with local changes that can influence a large part of the model.
Another contention for modularization of large ontologies is how to reuse only a part of the ontology without importing the entire ontology during development. Encounters from programming show that modules give a decent degree of abstraction to help in maintenance and reuse. The presentation of modules with local semantics can also assist with this issue.

3.3. Efficient Reasoning

A particular issue with distributed ontologies is the effectiveness of reasoning as the size of the ontologies increases computation time and the requirement of computing resources. Moreover, dependencies that are unseen and cyclic references can cause significant issues. The presentation of modules with local semantics and clear interfaces will assist with dissecting a distributed system to expose only smaller chunk of ontology as required.

4. Modular Ontology Modeling

Modular ontology modelling is a crucial task for many ontology-driven applications like multi-agent systems, decision support systems, and information retrieval systems. These latter are key areas of research in the field of ontologies. Various tools have been developed for creating ontologies, deploying, storing, and accessing from different environments like triple stores. Large portion of these tools and plugins treat ontologies as entities and help in categorizing them, however, very few ones focus on merging or integrating ontologies in a modular way:

- *CoMerger* technique works on merging multiple ontologies with adjustable Generic Merge Requirements (GMR).
- *iPrompt* provides guidance and recommendations to users during the merging process, recognizing inconsistencies, and problems.
- AnchorPrompt facilitates the iPrompt by graphical representation of correlations between the notions.
- *CoModIDE* is an interface feature that allows design patterns to express how they can be connected to one another.
- *XOD* eXtensible ontology development reuses existing terms to develop and apply well-established ontology design patterns (ODPs).
- *Pseudo-intent algorithm* features back-tracking-based FCA-Merge to perform a merger between identified relations.
- *Modular Ontology Design Library (MODL)* is a curated collection of well-documented ontology design patterns.

4.1. Contextual approach for reusing OWL Modules (Protégé Plugin)

As more ontology developers and users see the difficulties in reusing, maximizing, and maintaining large, and monolithic ontologies, there is a significant increase in interest in modularization strategies for ontologies. As previously stated, the concept of modularity is derived from software engineering. There is a huge potential and research gap to create a method that can work in a particular context to reuse two or more ontologies to form an ontology-driven system to work in a certain context rather than building an ontology from scratch. Moreover, there is a need of ontology visualization tool that ensure more efficient results. Due to this particular problem, we propose a solution called Contextual Integration of Modular Ontology (CIMOn), Which allows easy reuse and manual integration of various ontology components. Using this framework, knowledge will be only needed once distinctive ontologies will be incorporated around each other, such as additional relations, which can be saved independently. As a result, the

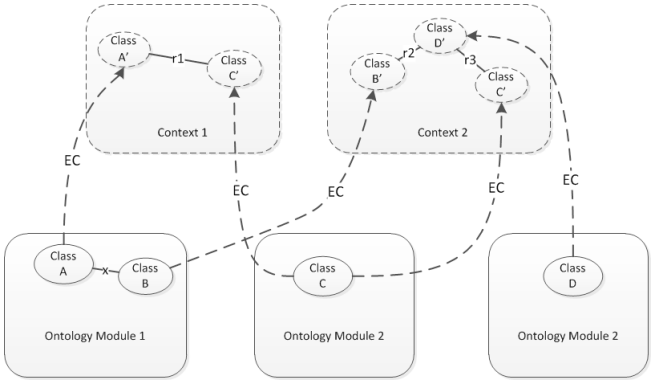

Fig. 2. By using context, multiple reuse at the same time without duplication is possible.

original ontology modules are preserved and can be reused for different purposes in multiple contexts at the same time.

Given the formal definition of the Context and Contextual Integration.

Context: Given a set of ontology $O = \{O_i\}_{i\in I}$, each O_i is expressed in Language L; S_i is the selected subset of the signature of each module O_i, i.e. $S_i \subseteq \mathrm{Sig}(O_i)$, $S = \{S_i\}_{i\in I}$ is the collection of all the selected signature of each module; Let r be an arbitrary relation expressed in L, r_{ij} associates two entities from $Sig(O_i)$ to $Sig(O_j)$, $R = \{r_{ij}\}_{i\neq j\in I}$ is the collection of such relations that happen between ontology pairs $\langle O_i, O_j\rangle_{i\neq j\in I}$. Then, for ontology set O and the selected signature collection S, we have $Context_O^S = R$.

Contextual Integration: Based on the definition of Context, let $M_i = O_i^{S_i}$ be the module extracted from O_i by signature S_i, $M_O^S = \{M_i\}_{i\in I}$, then the union of M_O^S and $Context_O^S$ is called the Contextual Integration of ontology set O based on signature collection S. i.e. $CI_O^S = \{M_O^S, Context_O^S\}$.

In more general ontology integration or composition scenario like Figure 2, it is not hard to see that one ontology module is reused multiple times. For instance, *ClassC* which is defined in *Module*2, this time it is referred in both *Context*1 and *Context*2. In different context, the use of the same entity could be totally different, e.g. *ClassC* could be the domain of property $r1$ in *Context*1, on the other hand, *ClassC* becomes the targeting range of property $r3$ in *Context*2. This polymorphism allows flexible contextual configuration. A big advantage of this scheme is that it allows using the same group ontology modules to construct various ontology integration or composition without ontology duplication and redundancy. In our research, an overlapping-oriented strategy is utilized.

Figure 3 is a preliminary implementation of our plug-in tool integrated into Protégé. By using this plug-in, we can edit the context of integrated ontologies. The Context tab is divided into three parts. The very left view is *SourceOntologyView*, the one next to it on the right is *TargetOntologyView*, and the main editing area on the rightmost is the *ContextEditorView*. A pair of source ontology module and a target ontology module should be specified for each *BinaryMappingBundle*, so two ontology view components are set on the left of the GUI. On the top side of each ontology view component, a combo box is set to allow user to switch between opened ontologies in Protégé

5. Case study and evaluation

In the following, we show how to apply CIMOn scheme to reuse and integrate ontology for a context using a case study in laundry detergent products of EU Eco-label[51]. Domain knowledge is stored in multiple ontology module files. It is a very common scenario that a product profile is required to be modeled by using multiple ontology modules [52]. So, let's suppose that, in some eco-labelling process or the evaluation task [53], knowledge engineers or domain experts want to use the knowledge in the ontology knowledge base to construct a cross-module product profile representation, detailed parameters of the product example are listed in Table 1.

In this product profile building scenario or modeling scenario, we should be aware that, first, we need the knowledge and information from different ontologies; second, we possibly only need some parts of certain ontologies. Higher level of flexibility is required, e.g. in the product profile modelling scenario that we have just presented. In this scenario

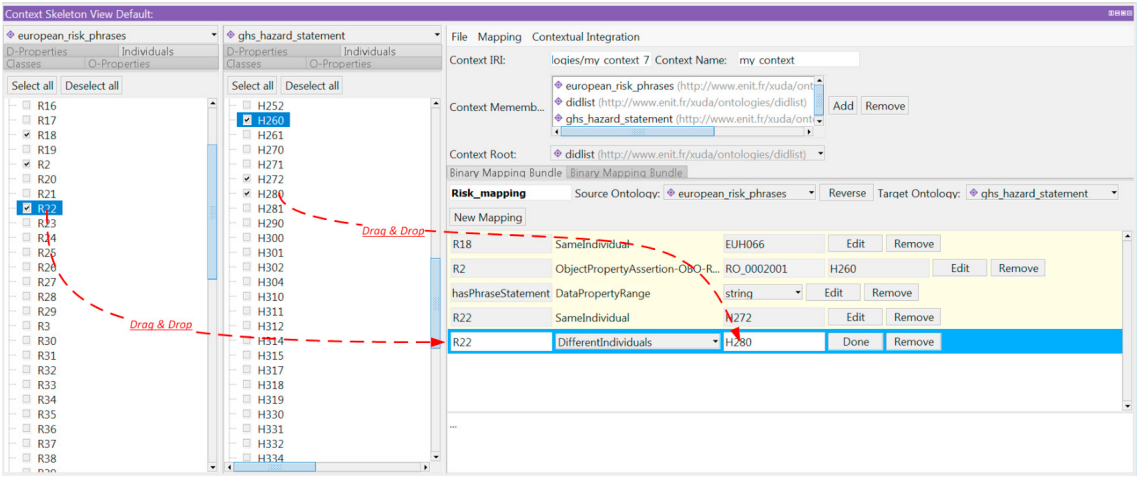

Fig. 3. User can directly drag and drop the entity he needs to each BinaryMapping item that is being edited.

Table 1. Detailed parameters of heavy-duty laundry detergent product profile example:

property parameter	value
product type	liquid
recommended dosage (reference dosage)	20.0 ml/kg wash
sales country	France
weight utility ratio (WUR)	2.0 g/kg wash
critical dilution volume (CDV)	30000.0 l/kg wash
aerobically non-biodegradability (aNBO)	0.5 g/kg wash
anaerobically non-biodegradability (anNBO)	0.5 g/kg wash
known ingredient:	C10-13 linear alkyl benzene sulphate; C8-12 Alkyl ether sulphate; Phosphonate; Sodium Lauroyl Methyl Isethionate; Benzisothiazol; Methylisothiazolinone.

or context, *Module Iso_standards*, *Module Regulation_european_commission* and *Module Commission_decision* can be excluded, because they are relevant as regards to the whole domain but not irrelevant for this specific scenario. More over, since there are only six ingredients that are explicitly listed, only a small part of *Module Ghs_hazard_statement*, *Module European_risk_phrases* and *Module Didlist* are needed. Figure 4 illustrates how we identify and form a context for this laundry detergent product profile modeling scenario. In the first step, relevant ontology modules are identified and chosen. This step requires that knowledge engineers have a basic understanding of the module's content. Second step, the dependencies between modules should be removed, e.g. owl:imports and sub-content or sub-module should be identified.

In our example, the whole *Module Laundry_detergent* is needed, thus it is kept entirely. The other three modules are tailored so that only the useful parts are kept. (The star symbol means that the module should be partly used and a sub-module is identified. In this scenario, rule modules are ignored.) At last, in the third step, a context involving the necessary content is formed. Complementary relations can be added and edited in the context. In our case, class with the same name "FunctionalUnit" are defined in both *Module Laundry_detergent* and *Module Didlist*. Axiom *Laundry_detergent:FunctionalUnit owl:equivalentClass Didlist:FunctionalUnit* and other mapping axioms are inserted in the context configuration.One of the significant benefits of this contextual integration or partly reuse method is that the size of the integration outcome is smaller than the one that uses owl:imports. A better reasoning performance is also obtained with regard to contextual integration.

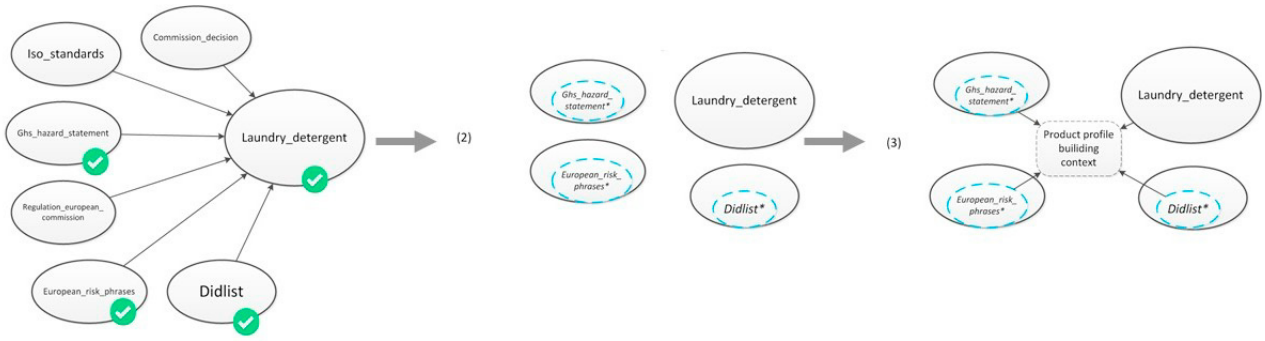

Fig. 4. A context is identified from modularized ontology knowledge base.

Table 2 is a comparison between traditional owl:imports method between CIMOn method.

Table 2. Detailed parameters of heavy-duty laundry detergent product profile integrated ontology

	integration by owl:imports	contextual integration by CIMOn
Logical axiom count	5049	425
Declaration axioms count	600	198
Class count	68	64
Object property count	52	44
Data property count	21	21
Individual count	459	69
Mean reasoning time (Hermit)	971ms	212ms

5.1. Ontology change management in context based modularization

The strategy of contextual modularization of ontology as described above helps in ontology change management in two primary ways. First, each context only adopts the useful part of the ontologies and therefore, greatly minimizes the effect of a change to be cascaded as opposed to the traditional need of importing the entire ontology however small the contribution of the imported ontology. It can be observed from the case study that *Module Iso_standards*, *Module Regulation_european_commission* and *Module Commission_decision* is excluded and only small parts of *Module Ghs_hazard_statement*, *Module European_risk_phrases* and *Module Didlist* are used in the context for the product profile context. Naturally, changes in any of the unused ontologies or their part will not affect the product profile context. Furthermore, if some change occurs in the used part, the context provides the related mappings to identify the affected concepts easily. Secondly, the parallel development of contexts reduces the interdependence among the ontologies in a network as separate applications may manage the changes locally within their contexts without affecting others. From the case study, it can be imagined that a company may need to create a number of product profiles. By using contexts for separate product profiles the company can restrict the change in one product profile from affecting the other profiles.

6. Conclusion

In this paper, we discussed the issues for ontology change, with a particular emphasis on heterogeneity methods that have been developed and the management of multiple versions, ontology merging, reusing and integration. Many of these aspects are connected to each other and therefore, resolution approaches to address more than one of these aspects. We also discussed the importance of modularization in the semantic web for re-using the existing ontologies for a certain context, minimizing the effects of changes in the original ontologies. It is evident from the literature review that a comprehensive method for handling modular ontology in contextual integration needs to be investigated.

Furthermore, a strategy for integrating different ontology modules based on different contexts is also presented as a potential solution for managing the ontology changes by reducing the effects of changes and providing ways to identify the areas for re-adjustments easily from context-specific mapping.

Acknowledgements

This research was partially financed by the OntoCommons project funded by the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 958371.

References

[1] Jacquette, D.(2014) "Ontology," *Central problems of philosophy*, .
[2] Smith, Barry, and Christopher Welty. (2001) "Ontology: Towards a new synthesis," *In Formal Ontology in Information Systems*, **10**, (3) 3–9. ACM Press.
[3] Fensel D. (2001) Ontologies.*In: Ontologies*. Springer, Berlin, Heidelberg,
[4] Gruber, Thomas R. (1993) "A translation approach to portable ontology specifications," *Knowledge acquisition*, **5**(2) 199–220.
[5] Guarino, Nicola, ed.(1998) Formal ontology in information systems:*Proceedings of the first international conference* **46**.
[6] Miller, George A.(1995) "WordNet: a lexical database for English," *Communications of the ACM* **38** (11) 39–41.
[7] Lenat, Douglas B., Ramanathan V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. (1990) "Cyc: toward programs with common sense," *Communications of the ACM*, **33** (8) 30–49.
[8] Oltramari, A.; Gangemi, A.; Guarino, N.; Masolo, C. (2002). "Restructuring WordNet's Top-Level: The OntoClean approach. OntoLex'2 Workshop," *Ontologies and Lexical Knowledge Bases (LREC 2002). Las Palmas, Spain* 17—26.
[9] Gangemi, A.; Navigli, R.; Velardi, P. (2003). The OntoWordNet Project: Extension and Axiomatization of Conceptual Relations in WordNet," *Proc. of International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE 2003). Catania, Sicily (Italy)*. pp. 820–838.
[10] CastaÃ±eda, VerÃ³nica, Luciana Ballejos, Ma Laura Caliusco, and Ma Rosa Galli. "The use of ontologies in requirements engineering,"*Global journal of research in engineering* 10, no. 6 (2010).
[11] Noy, N. F., & McGuinness, D. L. (2001). "Ontology development 101: A guide to creating your first ontology,"
[12] Noy, Natalya F., and Mark A. Musen. (2004). "Ontology versioning in an ontology management framework," *IEEE intelligent systems* **19** (4) 6–13.
[13] Flouris, Giorgos, Dimitris Plexousakis, and Grigoris Antoniou.(2005) "On applying the AGM theory to DLs and OWL," *In International Semantic Web Conference* 216–231. Springer, Berlin, Heidelberg.
[14] Pinto, H. Sofia, Asunción Gómez-Pérez, and João P. Martins. (1999) "Some issues on ontology integration," *In Proceedings of the International Joint Conference on Artificial Intelligence*, **99** 7–1
[15] Stojanovic, Ljiljana, Alexander Maedche, Nenad Stojanovic, and Rudi Studer. (2003) "Ontology evolution as reconfiguration-design problem solving "*In Proceedings of the 2nd international conference on Knowledge capture* 162–171
[16] Noy, Natalya F., and Michel Klein. (2004). "Ontology evolution: Not the same as schema evolution," *Knowledge and information systems* **6** (4) 428–440.
[17] Plessers, Peter, and Olga De Troyer. (2005). "Ontology change detection using a version log," *In International Semantic Web Conference* 578–592. Springer, Berlin, Heidelberg, .
[18] Haase, Peter, and Ljiljana Stojanovic. (2005)."Consistent evolution of OWL ontologies," *In European Semantic Web Conference*, 182–197. Springer, Berlin, Heidelberg.
[19] Flouris, Giorgos, and Dimitris Plexousakis. (2006) "Bridging ontology evolution and belief change,"*In Hellenic Conference on Artificial Intelligence* 486–489. Springer, Berlin, Heidelberg
[20] Heflin, Jeff, James Hendler, and Sean Luke. (1999)"Coping with changing ontologies in a distributed environment," *In AAAI-99 Workshop on Ontology Management* 74–79.
[21] Klein, Michel, and Natalya F. Noy. (2003) "A component-based framework for ontology evolution,"*In Workshop on Ontologies and Distributed Systems at International Joint Conference on Artificial Intelligence*, **3** (4).
[22] Noy, Natalya F., Abhita Chugh, William Liu, and Mark A. Musen. "A framework for ontology evolution in collaborative environments," *In International semantic web conference* 544–558. Springer, Berlin, Heidelberg, 2006.
[23] Alchourrón, Carlos E., Peter Gärdenfors, and David Makinson (1985). "On the logic of theory change: Partial meet contraction and revision functions," *Journal of symbolic logic* 510–530.
[24] Gärdenfors, Peter, ed. Belief revision. No. 29.*Cambridge University Press*, 2003.
[25] Gärdenfors, Peter. (1990) "The dynamics of belief systems: Foundations vs. coherence theories,"*Revue internationale de philosophie* 24-46.
[26] Hansson, Sven Ove. (1994)"Kernel contraction." *Journal of Symbolic Logic* 845–859.
[27] Hu, Wei, and Yuzhong Qu. (2006) "Block matching for ontologies," *In International Semantic Web Conference*, 300–313. Springer, Berlin, Heidelberg.
[28] Mendelzon, Alberto O. (1991) "On the difference between updating a knowledge base and revising it,"*KR proceedings*

[29] Qi, Guilin, Weiru Liu, and David A. Bell. (2006) "Knowledge base revision in description logics,"*In European Workshop on Logics in Artificial Intelligence* 386–398. Springer, Berlin, Heidelberg.

[30] Flouris, Giorgos, Dimitris Plexousakis, and Grigoris Antoniou.(2006) "Evolving ontology evolution." In *International Conference on Current Trends in Theory and Practice of Computer Science* 14–29. Springer, Berlin, Heidelberg.

[31] McGuinness, Deborah L., Richard Fikes, James Rice, and Steve Wilder.(2000) "An environment for merging and testing large ontologies." *In KR* 483–493

[32] Stojanovic, Ljiljana, Alexander Maedche, Boris Motik, and Nenad Stojanovic. (2002)"User-driven ontology evolution management," *In International Conference on Knowledge Engineering and Knowledge Management* 285–300. Springer, Berlin, Heidelberg.

[33] Huang, Zhisheng, and Heiner Stuckenschmidt.(2005) "Reasoning with multi-version ontologies: A temporal logic approach," *In International Semantic Web Conference* 398–412.

[34] Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., & Antoniou, G. (2008). "Ontology change: Classification and surveY," *The Knowledge Engineering Review* **23** (2) 117–152.

[35] Jiménez-Ruiz, Ernesto, Bernardo Cuenca Grau, Yujiao Zhou, and Ian Horrocks.(2012) "Large-scale Interactive Ontology Matching: Algorithms and Implementation," *In European Conference on Artificial Intelligence*, **242** 444–449.

[36] Kalfoglou, Y. & Schorlemmer, M. (2003). "Ontology Mapping: the State of the Art,"*Knowledge Engineering Review* **18** (1) 1–31.

[37] Farmer, William M.(1992) "Little theories," *In International Conference on Automated Deduction*, 567–581.

[38] Clark, Peter, John Thompson, Ken Barker, Bruce Porter, Vinay Chaudhri, Andres Rodriguez, Jerome Thomere et al. (2001) "Knowledge entry as the graphical assembly of components," *In Proceedings of the 1st international conference on Knowledge capture* 22–29. .

[39] Amir, Eyal, and Sheila McIlraith.(2005) "Partition-based logical reasoning for first-order and propositional theories,"*Artificial intelligence* **162** (1-2) 49–88.

[40] McIlraith, Sheila, and Eyal Amir. (2001) "Theorem proving with structured theories," *In International Joint Conference on Artificial Intelligence*, **1** 624–634.

[41] Lauritzen, Steffen L., and David J. Spiegelhalter.(1988) "Local computations with probabilities on graphical structures and their application to expert systems," *Journal of the Royal Statistical Society: Series B (Methodological)* **50** (2) 157–194.

[42] Wang, H., Horridge, M., Rector, A., Drummond, N. & Seidenberg, J. (2005) "Debugging OWL-DL ontologies: A heuristic approach," *Proceedings of the 4th International Semantic Web Conference (ISWC-05)*.

[43] M. Buchheit, F.D.W. Nutt, A. Schaerf,(1994) "Terminological systems revisited: Terminology = schema + views," *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*.

[44] Giunchiglia, F., Shvaiko, P. & Yatskevich, M. (2006) Discovering missing background knowledge in ontology matching," *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI-06)* 382-–386

[45] Serafini, L., Borgida, A., & Tamilin, A. (2005). Aspects of distributed and modular ontology reasoning," *International Joint Conference on Artificial Intelligence* **5** 570–575.

[46] Sadeeq, Mohammed J., and Subhi RM Zeebaree.(2021) "Semantic Search Engine Optimisation (SSEO) for Dynamic Websites: A Review," *International Journal of Science and Business* **5** (3) 148–158.

[47] M. Raza Naqvi, M. Waseem Iqbal, M. Usman Ashraf, S. Ahmad, A. T. Soliman et al. (2022)."Ontology driven testing strategies for iot applications,"*Computers, Materials & Continua*, **70** 5855—5869,

[48] Naz, T., Akhtar, M., Shahzad, S.K., Fasli, M., Iqbal, M.W. and Naqvi, M.R., 2020. "Ontology-driven advanced drug-drug interaction,"*Computers & Electrical Engineering*, **86** 106–695.

[49] Shahzad, S.K., Ahmed, D., Naqvi, M.R., Mushtaq, M.T., Iqbal, M.W. and Munir, F., (2021). "Ontology Driven Smart Health Service Integration," *Computer Methods and Programs in Biomedicine* **207** 106–146.

[50] Elmhadhbi, L., Karray, M.H., Archimède, B., Otte, J.N. and Smith, B., (2022)" Ontology-Driven Multicriteria Decision Support for Victim Evacuation," *International Journal of Information Technology* & Decision Making, **21** (01): 243–272.

[51] Xu Da, (2018)"A knowledge base with modularized ontologies for eco-labeling: Application for laundry detergents," *Computers in Industry* **98** 118—133.

[52] Xu, Da, Mohamed Hedi Karray, and Bernard Archim'ede.(2017)"A semantic- based decision support platform to assist products' eco-labeling process,"*Industrial Management and Data Systems*.

[53] Xu, Da, Hedi Karray, and Bernard Archim'ede.(2016)"Towards an inter-operable decision support platform for eco-labeling process," *In Enterprise Interop-erability VII,* 239–248.