On the Evolution of Ontological Signatures

Giorgos Flouris

ISTI-CNR
Via G. Moruzzi, 1,
56124, Pisa, Italy

flouris@isti.cnr.it

ABSTRACT

During ontology evolution, we are often faced with operations requiring the addition/removal of some ontological element (e.g., a concept) to/from the signature. Such operations deal with the ontological signature and are fundamentally different from operations that deal with the axiomatic part of the ontology, because they don't affect our knowledge on the domain but the non-logical symbols of the logic used to represent our knowledge on the domain. The consequences of this observation have been generally disregarded in the relevant literature. This paper attempts to fill this gap by introducing the concept of "change levels" and discussing the issues emerging from the different nature of the two types of operations. Furthermore, two alternative formalizations are described, which allow both types of operations to be represented at the same level, and, consequently, be considered of the same type.

1. INTRODUCTION

An *ontology* can be defined as a pair **<*S*,*A*>**, where ***S*** is the *vocabulary* (or *signature*) of the ontology and ***A*** is *the set of ontological axioms* [11]. The signature is usually modeled as a simple set containing the names of all concepts, properties or individuals that are relevant to the domain of discourse, while the axioms specify the intended interpretation of these symbols (names) in the given domain of discourse.

Given this definition, it seems reasonable that changes upon ontologies should affect both the signature and the ontological axioms. Indeed, ontology evolution has traditionally dealt with both types of changes and many works on ontology evolution handle both types of changes in a similar manner (e.g., [9], [16], [19], [20]).

However, the admittance of such operations is unique in the ontology evolution context; in the main research area studying changes upon a corpus of knowledge, namely *belief revision* [8] (also known as *belief change*), the signature (called *language* in that context) is considered static, so these types of changes are not considered.

As a result, the incorporation of signature changes in ontology evolution disallows the use of many of the formal tools provided by the related field of belief revision [7]. Thus, it is not surprising that many of the recent works in ontology evolution, especially the more theoretically-minded ones, do not consider such changes (e.g., [10], [13], [17], [18]).

In this paper we argue that treating both types of changes in the same manner is rather problematic from a methodological point of view, because the axioms and the signature each constitute a fundamentally different "knowledge level", so their respective change operations should be handled separately. This intuition is captured by introducing the concept of "change levels" (section 2), which allows the formal study of the two types of operations. In addition, two representation methodologies are introduced, which allow the incorporation of the signature information into the axiomatic part of the ontology, thus allowing a homogeneous treatment of both operation types (section 3).

Even though most of the results presented in this paper are applicable to many different kinds of representation formalisms and contexts, our focus will be the ontological context; standard logical Knowledge Bases (KBs) will be used for comparison. It will be assumed, for simplicity, that ontologies are represented using some Description Logic (DL) and logical KBs are represented using First-Order Logic (FOL), so the reader is assumed to have some basic familiarity with DLs [1] and FOL [4].

2. CHANGE LEVELS

2.1 Components of the Symbol Level

In his seminal work [15], Newell identified two major levels in every system (knowledge representation or other). The first, the *knowledge level*, contains all the abstractions that are used to describe a system's behavior and is independent of any implementation peculiarities; the second, the *symbol level*, contains the mechanisms (formalisms) that allow the system to operate.

Here, we focus on the symbol level; in the context of Knowledge Representation (KR), this level contains the axioms or formulas that describe system's knowledge (i.e., the KB). A KB is based on some logical formalism and uses various non-logical symbols (names) representing concepts, properties, predicates etc, depending on the context. The role of the KB is to capture the intended interpretation of the non-logical symbols in the domain of discourse using logical formulas; the semantics, syntax etc of these formulas is provided by the underlying logical formalism. This analysis motivates viewing the symbol level as being structured from these three clearly defined, but interrelated, components (levels): the *logic*, the *language* and the *knowledge base* (see table 1).

The first level (logic) is used to describe the logical elements (symbols) of the formalism that is used to represent our knowledge (e.g., connectives). Moreover, the semantics, syntax and inference mechanisms of the logic are all included in the logic level. In the ontological context, this level consists of the formal definition of the formalism used to formulate the axioms (e.g., DL [1], OWL [3], RDF [14] etc).

In the second level (language), the non-logical elements that are relevant to the domain are identified. These non-logical elements are, essentially, the (intuitive) names that we give to the various

-Table 1. Levels of Knowledge Representation (Components of the Symbol Level)

Components of the Symbol Level	Example: Knowledge Bases and Standard Logics	Example: Ontologies and Description Logics
Level 1: *Logic* Logical symbols, semantics, syntax, inference mechanism	FOL First-order connectives (e.g., ∀, ∃, ∧, …) Semantics of FOL Syntactical rules for FOL FOL inference rules	*ALC* *ALC* operators and connectives (e.g., ⊓, ¬, ⊑, …) *ALC* semantics Syntactical rules for *ALC* *ALC* inference rules
Level 2: *Language* Vocabulary and terminology of the domain	Non-logical symbols (names of predicates, functions etc)	Signature structure (names of concepts, properties etc)
Level 3: *Knowledge Base* Axioms, propositions	KB (set of FOL formulas)	Ontological axioms (set of *ALC* axioms)

relevant concepts, properties, predicates etc. This level corresponds to the signature of an ontology.

The third level (KB) is the actual embodiment of our knowledge on the domain. This level describes the interrelationships between the various elements of the language level; the types (and the semantics) of the allowed interrelationships are determined by the logic level. Obviously, the KB-level cannot be defined without an explicit and detailed description of the other two levels. In the ontological context, it is represented by the ontological axioms.

2.2 Language-level and KB-level Changes

The discrimination of the various components of the symbol level motivates a similar discrimination between the various types of changes on the basis of the component of the symbol level that they affect (see table 2).

In particular, the term *KB-level change* will be used to refer to change operations that directly affect the KB level of a KR system. Examples of KB-level changes in ontology evolution are the addition or removal of an IsA or a restriction upon the range of a property. An example of a KB-level operation in the standard logical setting (belief change) is contraction.

The term *language-level change* will be used to refer to change operators that directly affect the second level in table 1. Examples of language-level changes are the addition or removal of concepts, roles or individuals from the signature. In the standard logical setting, such operators are not considered, because the language is assumed to be static.

In principle, it is also possible to define *logic-level* changes, referring to changes that directly affect the logic itself. An example of such a change would be "remove the operator ⊓ from the underlying DL". However, the underlying logical formalism is usually considered static: neither belief revision nor ontology evolution deal with such operations.

Notice that the word "directly" is necessary in these definitions, because it is possible for a change to have side-effects affecting different levels. This is true because the three levels are not stand-alone entities but affect and depend on each other.

In particular, the removal of an element from the signature may have side-effects on the axiomatic part of the ontology; for example, if we are asked to remove a concept, then all axioms that refer to this concept (e.g., classification axioms) must be removed or otherwise amended so as not to involve the removed concept; all such amendments are KB-level changes.

A similar situation may occur when adding axioms; for example, if we are asked to add an IsA relation between concepts A and B and B does not exist in the ontological signature, then it should either be added (as a concept), or the operation should be rejected. In this case, a KB-level change may have a language-level side-effect.

On the other hand, removing an axiom from an ontology cannot cause any language-level changes. Some would argue that if, after the removal of an axiom, nothing is known regarding a certain element (e.g., a concept), then this element should be removed. This viewpoint is rather problematic. The fact that no interesting information regarding an element can be inferred from an ontology means that nothing is really known about this particular

Table 2. Change Levels and Their Support in Belief Change and Ontology Evolution

Change Levels	Belief Change	Ontology Evolution
Level 1: *Logic* Logic-level changes (affect the logic)	Does not support changes at this level	Does not support changes at this level
Level 2: *Language* Language-level changes (affect the language)	Does not support changes at this level	Supports changes at this level; changes may have side-effects in level 3
Level 3: *Knowledge Base* KB-level changes (affect the KB)	Supports changes at this level; changes cannot affect other levels; if they do, they are rejected as non-valid	Supports changes at this level; changes may have side-effects in level 2

element (yet). On the other hand, removing an element from the ontological signature implies that this element is irrelevant to the conceptualization of the domain described by the ontology; this statement is fundamentally different from the previous one. Therefore, it can be argued that, if the ontology engineer wishes to state that a particular element is irrelevant to the ontology, he should do so explicitly, by removing the element from the signature.

Similar arguments hold for the addition of ontological elements to the signature. Such elements are relevant to the domain conceptualized by the ontology at hand, since they are added to the signature, even if they do not (yet) appear in the axiomatic part. Thus, a language-level addition need not be coupled with a KB-level addition.

The identification of the exact side-effects of each operation in each level is irrelevant to this work and is omitted; the interested reader is referred to the standard ontology evolution literature (e.g., [9]) for a more detailed analysis of this issue.

2.3 Discussion on the Change Levels

As already mentioned, ontology evolution treats both language-level and KB-level operations in the same way. The analysis performed in the previous subsection implies that this approach may not be entirely correct from a methodological point of view, because it causes a mixture of effects upon both the axiomatic part of the ontology (KB-level) and the signature (language-level). The author argues that, even though both types of operations are useful, side-effects from one change level to the other should be avoided.

The argument can be stated more clearly with an example. Suppose that we attempt to develop an *ALC* ontology (see [1] for details on *ALC*), but later discover that we need more expressive power than the one provided by *ALC* for the particular domain. In that case, we are expected to switch to a new DL before adding any axiom types not supported by *ALC*. For example, if we want to add the axiom "$A \sqsubseteq B \sqcup \{x\}$" in the original ontology, we have to change the underlying DL first, then add the axiom.

If, instead, we attempted to add the new axiom directly, before changing (manually) the logic, that would not cause the introduction of the operator set-of ($\{\ldots\}$) into the underlying DL as a side-effect; no side-effect could cause a change in the underlying DL (logic-level change). On the contrary, the underlying ontology evolution system would not allow such a change (i.e., the addition of the axiom "$A \sqsubseteq B \sqcup \{x\}$" would be rejected as invalid).

What happens in this example is that a KB-level change is blocked (rejected) because it has a logic-level side-effect. This is considered intuitively adequate. But then, why should the addition of the axiom "$A \sqsubseteq B$" in an ontology whose signature does not contain B be allowed and cause the addition of B as a new concept (i.e., a KB-level change causing a language-level side-effect)?

Now consider a different case: suppose that the ontology engineer decides to switch logic by removing an operator (say $\sqcap$) from the DL. This, of course, should be made manually, as ontology evolution does not support logic-level changes. After such a change, much of the original ontology would be rendered invalid, as several axioms may use the removed operator. Nevertheless, we would expect the ontology engineer (rather than the ontology evolution system) to manually amend the axioms containing this operator so as to capture (as much as possible) the intended meaning of the axioms of the more expressive logic (the one containing $\sqcap$) using the axioms of the less expressive one (the one not containing $\sqcap$); this should be made before the removal of the operator $\sqcap$ from the logic.

On the contrary, we expect an ontology evolution algorithm to apply KB-level changes as side-effects in order to amend the axioms that are rendered invalid following the removal of a signature element (language-level change).

The conclusion from these examples is that there should exist clear boundaries between the various change levels disallowing the propagation of any side-effects from one level to the other. Should a change in one level cause changes in another level, it should be blocked or rejected until the knowledge engineer is given the chance to correct the problem(s) using change operations of the appropriate level.

This viewpoint is influenced by the viewpoint employed in standard logical formalisms. In belief change, only KB-level changes are considered: any changes that affect other levels, or that have side-effects in other levels, are rejected as non-valid. In fact, the operation "remove the predicate P from the language" would sound equally absurd to a logician as the operation "remove the operator $\sqcap$ from the DL" would sound to an ontology engineer.

The fact that belief change does not deal with language-level operations should not be viewed as a shortcoming of the field. If we confine each type of change to its own level only (by disallowing side-effects to other levels), then language-level operations become trivial to execute, because their language-level side-effects can be easily identified and resolved. Indeed, the removal of an element has no language-level side-effects, while the addition of an element could have, but only if the same name is already in use.

For example, if we are asked to add a class named P and there is already a property with that name, we should first remove the property before adding the class, as most formalisms (e.g., DLs) require the names used for classes, properties and individuals to be mutually disjoint. This side-effect would not exist in formalisms without this restriction, e.g., in RDF [14] or OWL Full [3]. In any case, such side-effects are trivial to identify, so belief change chose to ignore them. Of course, a language-level operation (in particular, a removal) could have a number of non-trivial KB-level side-effects, if such side-effects were allowed.

Another problem with language-level operations is that, unlike KB-level operations, it is not possible to formally describe a language-level operation using DL (or FOL) constructs. One of the consequences of this fact is that such operations render the recently proposed mapping of ontology evolution to belief change [7] unusable, since it is not possible to express a language-level change in the terminology used in belief change (even if it was, it wouldn't be of much use, as belief change does not provide any tools to handle such operations). A side-effect of this fact is that many formal approaches to ontology evolution (e.g., [10], [13], [17], [18]) do not consider language-level operations.

3. ALTERNATIVE REPRESENTATIONS

The previous section identified the need to keep operations affecting different levels separate and disallow side-effects from one level to affect the other. Even though such a rule is useful for the formal analysis of change operations, many existing methods do violate it.

In this section, we address this problem by describing two alternative techniques for representing ontologies. These representations allow the encapsulation of signature information into the axiomatic part of an ontology, which, in turn, confines both language-level and KB-level change operations (and side-effects) into the KB-level.

This way, we only need to consider KB-level operations which are well-studied and supported by both ontology evolution and belief revision, while still being able to perform changes (and side-effects) that would normally be classified as language-level ones. This allows us to enjoy the best of both worlds, since all useful operations and their side-effects can be addressed on the same level.

Applying these representation to ontologies has other advantages as well. First, it allows belief change techniques to be used to handle language-level operations; second, it makes the embedding of ontology evolution techniques into belief change methodologies (and vice-versa) possible; third, it allows a homogeneous treatment of all interesting operations; and, fourth, it allows methodologies originally designed to handle KB-level operations only to be used for language-level operations as well.

These representations should mainly take into account two important characteristics of signatures: first, there could be elements that are relevant to the ontological conceptualization (so they should appear in the signature in the standard approach), but for which no useful information is known (yet), so they don't appear in any of the "standard" DL axioms; second, the introduction of language-level assertions in the KB-level would inevitably introduce some non-standard KB-level information, whose semantics should be taken into account by the inference mechanism of the logic at hand.

Not surprisingly, the proposed alternative representations are not without problems of their own, discussed in the respective subsections. Such drawbacks are inherent in this approach, since this is actually an effort to model (represent) two intrinsically different types (levels) of information in the same representational level. Nevertheless, the proposed representations constitute interesting possible solutions to the problems described in the previous section because they allow the collapse of two representation levels into one. Both alternatives below will be described for DLs; however, they can be straightforwardly used for other logics as well, both in the logical and ontological setting.

3.1 First Alternative

This alternative originally appeared in earlier works by the author [5], [6], [7] in order to allow the representation of language-level ontology evolution operations using KB-level constructs. This was necessary to the end of being able to define the problem of ontology evolution in terms of the related field of belief change, which was one of the main objectives of the aforementioned works. Without the use of this alternative representation, only the part of ontology evolution dealing with KB-level changes can be described in terms of belief change.

Under this approach, the ontological signature is assumed static and the same for all ontologies; in particular, it is assumed that an ontological signature contains all possible element names (i.e., all strings of finite length). This deprives the signature from its original purpose of determining relevance of element names to the domain and raises the issue of how can one determine relevant and non-relevant element names.

There are two ways to resolve this problem. The first is to assume that there is no issue of relevance. All elements are, in principle, relevant to the domain of discourse, even though, for some of them, no information is known (yet), so they don't appear in any axiom. This approach was termed the *Open Vocabulary Assumption* (OVA) in [5]. Obviously, OVA causes the loss of all signature information and renders all language-level operators invalid, so it is not adequate for the purposes of this paper.

The second approach incorporates a new unary connective in the underlying DL to denote relevance; this connective is called the *Existence Assertion Connective* and is denoted by %. The semantics of % is that the axiom "%A" should be implied by the ontology if and only if the element A is relevant to the conceptualization of the ontology (i.e., it would have been part of the signature, if the standard approach was used). Using this connective, we can determine whether an element is relevant to the ontology or not, leading to what was termed the *Closed Vocabulary Assumption* (CVA) [5].

Of course, the standard DL inference mechanism should be amended in order to incorporate the semantics of the new connective. In [5] the proper amendments were described, which eventually boil down to two conditions: the first guarantees that whenever an element A appears in a "standard" DL axiom, then this DL axiom implies the "relevance" of the element (i.e., %A) but not the relevance of any elements not appearing in the axiom (e.g., %B); the second guarantees that axioms of the form "%A" do not imply any "useful" KB-level information, in the sense that no non-tautological "standard" axiom can be implied by any set of assertions of the form %A.

It is clear that the % connective "downgrades" language-level assertions into KB-level assertions, thus making possible the representation of what should be language-level change operations (and statements) using KB-level change operations (and statements). For example, the addition of an element A is now expressed as the addition of the axiom %A. The semantics of the inference relation dictate what the side-effects of such operations should be. For example, the removal of %A implies the removal of all axioms that include A (otherwise %A would re-emerge as an implication of such an axiom, due to the first amendment of the inference relation described above).

This fact implies that it is easy to adapt some standard belief change or ontology evolution algorithms so as to deal with language-level operations; all we have to do is replace the standard inference relation of the underlying logic/DL with the modified one. Of course, this technique may work only for the algorithms that are not tied to any particular logic/DL (and thus a particular inference relation).

The major disadvantage of this method is that it requires the addition of a non-standard connective in the logic, thus rendering standard inference algorithms non-sound for inferences that involve "fresh" elements, as well as non-complete for inferences

that involve the existence assertion connective. On the other hand, it is relatively easy to implement and it is applicable to any logic.

It is possible, even though not necessary, to refine the connective % so as to indicate whether an element is a class, role or individual (in effect introducing three different existence assertion connectives). Unfortunately, this refinement introduces an additional (and unnecessary) complexity in the approach so it will not be considered here. For a more detailed discussion on this refinement, as well as on the other issues raised in this subsection, see [5].

3.2 Second Alternative

This alternative maps DL information into FOL formulas, but, instead of using the standard mapping [2], it employs a twist in the way signature elements are viewed, resulting to a different mapping. This non-standard mapping has the advantage that it encapsulates the signature structure and allows it to be part of the resulting FOL KB. The final result is similar to the previous alternative: language-level assertions (change operations) can be expressed using KB-level assertions (change operations).

In order to implement this alternative, a FOL is defined whose language contains one predicate name for each connective appearing in the DL and one function name for each operator appearing in the DL. It also contains an infinite number of individual names (constants), which will be used to represent all possible element names that may appear in the ontological signature. To cover all cases, any finite-length string will be assumed to be a constant in said FOL (except, of course, from the symbols reserved for functions and predicates). In addition, the unary predicates Class(.), Property(.) and Instance(.) are included in order to capture language-level assertions, i.e., that a respective element name (a FOL constant in this representation) is a class, property or instance respectively in the DL ontology.

The mapping of a DL axiom into this FOL is made by rewriting the axiom using prefix (Polish) notation and then replacing each connective and operator with its respective predicate or function in the defined FOL. For example the axiom: "$\forall R.A \sqcap B \sqsubseteq C \sqcap A$" would be mapped into the FOL formula: "$Con_{\sqsubseteq}(Oper_{\sqcap}(Oper_{\forall}(R,A),B),Oper_{\sqcap}(C,A))$", where $Con_{\sqsubseteq}(.,.)$ is the binary predicate attached to the DL connective $\sqsubseteq$ and $Oper_{\sqcap}(.,.)$, $Oper_{\forall}(.,.)$ are the binary functions attached to the DL operators $\sqcap$, $\forall$ respectively. Language-level assertions are simpler to capture: Class(A), Property(A), Individual(A) imply that A is a class, property, individual respectively.

The mapping of axioms and signature assertions to FOL ground facts in the above manner is not enough, because the semantics of the connectives and operators are not carried over. To achieve this, the FOL KB should be coupled with a number of integrity constraints guaranteeing the intuitively expected behavior of the various FOL predicates and functions. For example, to guarantee the transitive semantics of the $Con_{\sqsubseteq}$ predicate, we need the constraint: "$\forall x,y,z\ Con_{\sqsubseteq}(x,y) \wedge Con_{\sqsubseteq}(y,z) \rightarrow Con_{\sqsubseteq}(x,y,z)$".

Similar constraints must be defined for the special predicates Class, Property and Instance as well; the general idea is the same as the one employed in order to amend the inference relation of the previous alternative. Unfortunately, the constraints in this case cannot be simplified by dropping the three predicates and keeping just one as was done in the previous subsection; such a change would not allow the detection of the invalidity of the statement "$Con_{\sqsubseteq}(Oper_{\forall}(A,A),A)$, as it would not be possible to determine that A in this statement is used both as a class and as a role.

It is clear by the above analysis that, for very expressive DLs, the task of defining all the necessary integrity constraints is very difficult; therefore, the difficulties involved in applying this method are depending on the logic's expressiveness (unlike the first alternative). This constitutes the most important drawback of this alternative, and makes it more adequate for less expressive logical formalisms.

The role of "downgrading" the language-level assertions into KB-level ones (undertaken by the % connective in the previous approach) is now performed by the three special predicates Class, Property, Instance. The same general comments on how this allows language-level changes and how existing (belief change or ontology evolution) algorithms could be used to address such changes apply here.

4. EPILOGUE

In this paper, three different representation levels were introduced (logic, language and KB) and an important distinction between changes affecting each level was introduced. This discussion is particularly relevant for the signature (language-level changes) and the axiomatic part of an ontology (KB-level changes); arguments were provided in favor of the discrimination of the two change types, as well as against allowing side-effects caused by a change to affect other levels.

Moreover, two alternative representation techniques were introduced that allow the collapse of the two lower levels (language and KB) into one (KB). These methodologies allow us to execute both language-level and KB-level changes at the same level (KB) and avoid the problem of side-effects caused from one level to affect another. In addition, these approaches facilitate the smoother integration of ontology evolution (dealing with language-level and KB-level changes) and belief change (dealing with KB-level changes only) approaches [7] and allow us to use methods originally designed to handle KB-level changes for language-level changes as well.

Even though these alternative representations suffer from various deficiencies, they could prove useful when the aforementioned collapse of the two levels into one is necessary. The deficiencies of the proposed alternatives show the inherent difficulty of this task and serve as an additional argument in favor of the proposed definition of representation and change levels.

The discrimination of the three representation levels is a known issue in the literature, but, to the best of the author's knowledge, the explicit classification of the various types of changes in three levels based on the representation level they affect was never considered before, except only superficially by earlier works of the author [5], [6], [7], as well as in [12], where a similar problem (variable forgetting) was addressed in the context of Propositional Logic.

5. ACKNOWLEDGMENTS

Significant fragments of this work have benefited from discussions with Carlo Meghini, Dimitris Plexousakis, Vassilis Christophides and Yannis Tzitzikas. This work was carried out during the author's tenure of an ERCIM "Alain Bensoussan" Fellowship Program.

6. REFERENCES

[1] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. (eds). *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2002.

[2] Borgida, A. On the Relative Expressiveness of Description Logics and Predicate Logics. *Artificial Intelligence, 82*, 1996, 353-367.

[3] Dean, M., Schreiber, G., Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuiness, D., Patel-Schneider, P., and Stein, L. A. OWL Web Ontology Language Reference. W3C Recommendation, 2004. Available at: http://www.w3.org/TR/owl-ref

[4] Enderton, H. B. *A Mathematical Introduction to Logic*. Academic Press, New York, 1972.

[5] Flouris, G. *On Belief Change and Ontology Evolution*. Ph.D. Thesis, Department of Computer Science, University of Crete, 2006.

[6] Flouris, G., Plexousakis, D., and Antoniou, G. Generalizing the AGM Postulates: Preliminary Results and Applications. In *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR-04)*, 2004, 171-179.

[7] Flouris, G., Plexousakis, D., and Antoniou, G. Evolving Ontology Evolution. In *Proceedings of the 32nd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM-06)*, Invited Talk, 2006.

[8] Gärdenfors, P. Belief Revision: An Introduction. In Gärdenfors, P. (ed). *Belief Revision*, Cambridge University Press, 1992, 1-20.

[9] Haase, P., and Sure, Y. D3.1.1.b State of the Art on Ontology Evolution. SEKT Deliverable, 2004. Available at: http://www.aifb.uni-karlsruhe.de/WBS/ysu/publications/SEKT-D3.1.1.b.pdf

[10] Halaschek-Wiener, C., and Katz, Y. Belief Base Revision For Expressive Description Logics. In *Proceedings of OWL: Experiences and Directions 2006 (OWLED-06)*, 2006.

[11] Kalfoglou, Y., and Schorlemmer, M. Ontology Mapping: the State of the Art. *Knowledge Engineering Review (KER), 18, 1*, 2003, 1-31.

[12] Lang, J., Liberatore, P., and Marquis, P. Propositional Independence: Formula-Variable Independence and Forgetting. *Journal of Artificial Intelligence Research (JAIR), 18*, 2003, 391-443.

[13] Meyer, T., Lee, K., and Booth, R. Knowledge Integration for Description Logics. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-05)*, 2005.

[14] Miller, E., Swick, R., and Brickley, D. Resource Description Framework (RDF) / W3C Semantic Web Activity. 2006. Available at: http://www.w3.org/RDF

[15] Newell, A. The Knowledge Level. *Artificial Intelligence, 18, 1*, 1982.

[16] Plessers, P., de Troyer, O., and Casteleyn, S. Event-based Modeling of Evolution for Semantic-driven Systems. In *Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE-05)*, 2005, 63-76.

[17] Qi, G., Liu, W., and Bell, D. A. A Revision-Based Approach for Handling Inconsistency in Description Logics. In *Proceedings of the 11th International Workshop on Non-Monotonic Reasoning (NMR-06)*, 2006.

[18] Qi, G., Liu, W., and Bell, D. A. Knowledge Base Revision in Description Logics. In *Proceedings of the 10th European Conference on Logics in Artificial Intelligence (JELIA-06)*, 2006.

[19] Stojanovic, L., Maedche, A., Motik, B., and Stojanovic, N. User-driven Ontology Evolution Management. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW-02)*, 2002, 285-300.

[20] Stuckenschmidt, H., and Klein, M. Integrity and Change in Modular Ontologies. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003.