A Classification of Ontology Change

Giorgos Flouris[1,2], Dimitris Plexousakis[2], and Grigoris Antoniou[2]
[1]Istituto della Scienza e delle Tecnologie della Informazione, C.N.R., Via G. Moruzzi, 1, 56124, Pisa, Italy
Email: georgios.flouris@isti.cnr.it
[2]Institute of Computer Science, FO.R.T.H., P.O. Box 1385, GR 71110, Heraklion, Greece
Email: {dp, antoniou}@ics.forth.gr

***Abstract*— The problem of modifying an ontology in response to a certain need for change is a complex and multifaceted one, being addressed by several different, but closely related and often overlapping research disciplines. Unfortunately, the boundaries of each such discipline are not clear, as certain terms are often used with different meanings in the relevant literature. The purpose of this paper is to identify the exact relationships, connections and overlaps between these research areas and determine the boundaries of each field, by performing a broad review of the relevant literature.**

I. Introduction

Originally introduced by Aristotle, *ontologies* are often viewed as the key means through which the Semantic Web vision [3] can be realized. Ontologies provide a means to formally define the basic terms and relations that comprise the vocabulary of a certain domain of interest [34], enabling machines to process information provided by human agents. As a result, they can help in the representation of the content of a web page in a formal manner so as to be suitable for use by an automated computer agent, search engine or other web service. The importance of ontologies in current AI research is also emphasized by the interest shown by both the research and the enterprise community to various problems related to ontologies and ontology manipulation [39].

Ontologies are often large structures, whose development and maintenance give rise to interesting research problems. One of the most important such problems is the problem of modifying an ontology in response to a certain need. In this paper, the term *ontology change* will be used to describe this problem; the term will be used in a broad sense, covering any type of change, including changes to the ontology in response to external events, changes dictated by the ontology engineer, changes forced by heterogeneity considerations and so on.

In order to cope with the complex problem of ontology change, several related research disciplines have emerged (such as ontology evolution, alignment, merging, mapping etc), each dealing with a different facet of the problem. These areas are greatly interlinked; as a result, several works and systems deal with more than one of these topics causing a certain confusion to a newcomer. This confusion is further increased by the fact that certain terms are often used with different meanings in the relevant literature, denoting similar, but not identical, research directions or concepts. For examples of such confusing and overused terms refer to [13], [51].

We believe that this lack of a standard terminology constitutes a major bottleneck for the ontology change community, causing unnecessary confusion as well as misunderstandings. The purpose of this paper is the introduction of a terminology which follows the most common uses of the various terms in the literature. Fixing this terminology will allow us to determine the boundaries of each field as well as to get a grip on their differences, overlaps, interactions and connections.

To do that, we perform a shallow, but broad, literature review on the field of ontology change, and introduce a broadly accepted terminology that will, hopefully, serve as a point of reference for the ontology change community. Our purpose is to give a clear overall picture of each relevant subfield and determine the boundaries, interactions and overlaps between the various areas; the interested reader is referred to the numerous bibliographic references that will appear throughout this paper for more details on each area or deeper results. A comprehensive summary of the results of our survey can be found in table I at the end of this paper.

II. Ontologies and Ontology Change

A. What is an Ontology?

The term ontology has come to refer to a wide range of formal representations, including taxonomies, hierarchical terminology vocabularies or detailed logical theories describing a domain [44]. For this reason, a precise definition of the term is rather difficult. A commonly used definition can be found in [21] where an ontology was defined to be a *specification of a shared conceptualization of a domain*.

A more formal, algebraic, approach, identifies an ontology as a pair $< S, A >$, where S is the *signature* of the ontology (being modeled by some mathematical structure, such as a lattice, a poset or an unstructured set) and A is the *set of ontological axioms*, which specify the intended interpretation of the signature in a given domain of discourse [27].

B. Ontology Change

Several reasons for changing an ontology have been identified in the literature. An ontology, just like any structure storing information, may need to change simply because the modeled domain has changed [55]; but even if we assume a static domain, which is a rather unrealistic assumption for most applications, we may need to change the perspective under which the domain is viewed [44], or we may discover a design flaw in our original conceptualization [52]; we may also wish to adapt to a change in users' needs or perspective and/or incorporate additional functionality [22]; new information, previously unknown, classified or otherwise unavailable may become available or different features of the domain may become important [25].

In addition, ontology development is becoming more and more a collaborative and parallelized process, whose subproducts need to be combined to produce the final ontology [32]; this process would require changes in each subontology to reach a consistent final state. But even then, the so-called "final" state is rarely final, as ontology development is usually an ongoing process [25].

The complex web of dependencies that is usually formed around an ontology is another common reason for change. The distributed nature of the Semantic Web implies that the knowledge engineer has no control over dependent and/or depending ontologies; if any of these ontologies change, the local ontology might also need to be modified [25]. In other cases, a certain agent, service or application may need to use an ontology whose terminology or representation is different from the one it can understand [9], so he needs to perform some kind of translation (change) in the imported ontology. Finally, we may need to merge or integrate information from two or more ontologies in order to produce a more appropriate one for some application [51].

Several philosophical problems related to knowledge update in general have been identified in the research area of *belief revision* [19], [20], [28]; many of them are also applicable to knowledge represented in ontologies [12], [13]. However, the problem is further complicated by the large size of modern day ontologies [39] and by the aforementioned ontology interdependencies; even subtle changes in an ontology may have unforeseeable effects in dependent and/or depending applications, services, data and other ontologies [54].

These facts raise the need to maintain different interoperable versions of the same ontology [25], [26], [31], a problem greatly interwoven with ontology change [30]. Moreover, heterogeneity leads to problems when an agent, service or application uses information from two different ontologies [9]. As ontologies often cover overlapping domains using different viewpoints and terminology, some kind of translation may be necessary in many practical applications.

All these arguments indicate the importance of the problem of ontology change and motivate us to use the term in order to cover all aspects of ontology modification, as well as the problems that are indirectly related to the change operation such as the maintenance of different versions of an ontology or the translation of ontological information in a common terminology. More specifically, we will use the term *ontology change to refer to the problem of deciding the modifications to perform upon an ontology in response to a certain need for change as well as the implementation of these modifications and the management of their effects in depending data, ontologies, services, applications, agents or other elements*.

In this definition, the need to change the ontology may take several different forms, including, but not limited to, the discovery of new information (some new instance data, another ontology, a new observation etc), a change in the focus or the viewpoint of the conceptualization, information received by some external source, a change in the domain, communication needs between heterogeneous sources of information or ontologies, the fusion of information from different ontologies and so on.

This definition covers several related research areas which are studied separately in the literature. In this paper, we identify nine such areas, namely *ontology mapping*, *morphism*, *alignment*, *articulation*, *translation*, *evolution*, *versioning*, *integration* and *merging*. Each of these areas deals with a certain facet of the problem from a different view or perspective, covering different application needs, change scenarios or needs for change (see table I for a comprehensive summary).

These fields are greatly interlinked, so several papers deal with more than one of these problems. In other cases, the same term is used in different papers to describe different research areas. This situation can easily lead to misunderstandings, confusion and unnecessary waste of effort, especially for a newcomer. In the following sections, we will attempt to precisely define the boundaries of each area and uncover their relations, overlaps and differences. This attempt will hopefully draw a fine line between the various research areas, allowing the clarification of the meaning of each term and making the differences and similarities between them explicit. The definitions provided here will not be arbitrary, but will be based on the most common uses of each term in the literature.

III. Ontology Evolution and Versioning

A. Disambiguating the Terms

Ontology versioning is often considered a stronger variant of ontology evolution [23]. Under that viewpoint, ontology evolution is the process of changing an ontology without losing data or negating its validity, whereas ontology versioning should additionally guarantee the validity, interoperability and management of all previous versions, including the current one, as well as transparent access to these versions.

This viewpoint is influenced by related research on relational and object-oriented database schema evolution and versioning [18], [29], [50]. A survey on the differences and similarities of ontologies and databases, as well as their impact with respect to evolution and versioning, can be found in [44]. In this paper it is argued that ontology evolution and versioning become indistinguishable under this understanding, because, due to the distributed nature of the Semantic Web, multiple versions of ontologies are bound to exist and must be supported. Furthermore, ontologies and dependent elements are likely to be owned by different parties; as a result, some parties may be unprepared to change and others may even be opposed to it [25]. All these facts force us to maintain and support different versions of ontologies, making ontology evolution (under this understanding) useless in practice.

We believe that the problem of modifying the ontology (ontology evolution) should be clearly separated from the problem of maintaining the interoperability of different versions of the ontology (ontology versioning). This distinction is not always clear in the literature, because the ontology dependencies and interrelationships force us to consider the issue of propagating the changes to dependent elements [37]. This tight coupling has caused ontology evolution algorithms to deal with these problems as well. For example, in [54], ontology evolution is defined as the timely adaptation of an ontology to changed business requirements, to trends in ontology instances and patterns of usage of the ontology-based application, as well as the consistent management and propagation of these changes to dependent elements.

On the contrary, here we define *ontology evolution to refer to the process of modifying an ontology in response to a certain change in the domain or its conceptualization* [13]; on the other hand, *ontology versioning refers to the ability to handle an evolving ontology by creating and managing different versions of it* [30]. Thus, ontology evolution is restricted to the process of modifying an ontology while maintaining its validity, whereas ontology versioning deals with the problem of managing different versions of an evolving ontology, maintaining interoperability between versions and providing transparent access to each version as required by the accessing element.

B. Ontology Evolution: General Discussion

Since an ontology is a specification of a shared conceptualization of a domain [21], a change may be caused by either a change in the domain, a change in the conceptualization or a change in the specification [30]. Changes in the specification refer to changes in the way the conceptualization is formally recorded, i.e., changes in the representation language. This type of change is dealt with in the field of ontology translation (see the next section and table I at the end of this paper). Thus, our definition of ontology evolution covers the first two types of change only (domain and conceptualization changes).

Both types of changes are not rare. The conceptualization of the domain may change because of a new observation or measurement, a change in the viewpoint or usage of the ontology, newly-gained access to information that was previously unknown, classified or otherwise unavailable and so on. The domain itself may also change, as the real world itself is generally not static but evolves over time. More examples of reasons initiating changes can be found in [30], [44].

C. Ontology Evolution Phases

In order to tame the complexity of the problem, six phases of ontology evolution have been identified, occurring in a cyclic loop [54]. Initially, we have the *change capturing* phase, where the changes to be performed are identified. Three types of change capturing have been distinguished: structure-driven, usage-driven and data-driven [23].

Once the changes have been determined, they have to be properly (and formally) represented during the *change representation phase*. There are two major types of changes, namely atomic and complex [56] (also called elementary and composite in [54]). Atomic changes represent simple, fine-grained changes such as the deletion of a concept. Complex changes represent more coarse-grained changes and can be replaced by a series of atomic changes. Even though possible, it is not generally appropriate to use a series of atomic changes to replace a complex change, as this might cause undesirable side-effects [54]; the proper level of granularity should be identified at each case. Unfortunately, there is no general consensus in the literature on the type and number of complex changes that are necessary. In [54], 12 different complex changes are identified; in [44], 22 such operations are listed; in [56] however, the authors mention that they have identified 120 different interesting complex operations and that the list is still growing! In fact, the number of definable complex operations can only be limited by setting a granularity threshold on the operations considered; if we allow unlimited granularity, we will be able to define more and more operations of coarser and coarser granularity, limited only by our imagination [32]. Thus, creating a complete list of complex operations is not possible, but, fortunately, it is not necessary either, since a complex operation can always be defined as a series of atomic operations [32].

The third phase is the *semantics of change* phase, in which we identify and address any problems that will be caused when the required changes are actually implemented, thus guaranteeing the validity of the ontology at the end of the process. For example, if a concept is deleted, we need (among other things) to determine what to do with its instances (e.g., delete them or re-classify them). In [54], it is suggested that the final decision should be made indirectly by the ontology engineer, through the selection of certain pre-determined evolution strategies, indicating the appropriate action in each case. Other (manual or semi-automatic) approaches are also possible (see [23]). This phase is probably the most crucial of ontology evolution, because during that phase the direct and indirect effects of a given change request are determined.

The *change implementation* phase follows, where the changes are physically applied to the ontology, using an appropriate tool, like, for example, the KAON API [54]. Such a tool should have transactional properties, based on the ACID model, i.e., guaranteeing Atomicity, Consistency, Isolation and Durability of changes [23]. It should also present the changes to the ontology engineer for final verification and keep a log of the implemented changes [23].

The implemented changes need to be propagated to all interested parties; this is the role of the *change propagation* phase. In [37], two different methods to address the problem are compared, namely push-based and pull-based approaches. Under a push-based approach, the changes are propagated to the dependent ontologies as they happen; in a pull-based approach, the propagation is initiated only after the explicit request of each of the dependent elements. In both [37] and [54] the push-based approach is favored. Alternatively, one could avoid this step altogether, by using an ontology versioning algorithm [31], allowing the interested parties to work with the original version of the ontology and update to the newer version at their own pace, if at all. This alternative is considered more realistic for practical purposes [25].

Finally, the *change validation* phase allows the ontology engineer to review the changes and possibly undo them. This phase may uncover further problems with the ontology, thus initiating new changes that need to be performed to improve the conceptualization; in this case, we need to start over by applying the change capturing phase of a new evolution process, closing the cyclic loop.

Notice that heterogeneity issues are not handled by the above ontology evolution model. Obviously, any approach to ontology evolution would collapse in the presence of heterogeneity, unless coupled with some algorithm that deals with heterogeneity (like the ones discussed in the next section). However, under the proposed model, this is not a problem, as the ontology engineer identifies the changes to be performed during the change representation phase, so it can be reasonably assumed that these changes will be represented in a suitable terminology. An alternative model of ontology evolution, involving five phases, has been proposed in [52].

D. The Current State of the Art in Ontology Evolution

The current state of the art in ontology evolution, as well as a list of relevant tools can be found in [23]. Some of these tools are simple ontology editors, whereas others provide more specialized features to the user. In some cases, the user can define some kind of pre-defined evolution strategies [54] that control how changes will be made, thus allowing the tool to perform some of the required changes automatically. Other tools allow collaborative edits, i.e., several users can work simultaneously on the same ontology [7], whereas others support transactional changes [23]. In other works, features related to ontology versioning, undo/redo operations and other helpful utilities are supported [7]. Some tools provide intuitive graphical interfaces that help the visualization of the process [33]. For more details on such systems refer to [7], [23].

A declarative language for changing the data portion of an RDF ontology appears in [38]. An alternative approach that uses belief revision techniques to handle ontology evolution has recently appeared [11], [14]–[16]; similar approaches, at a preliminary stage, appear in [35], [40]. An interesting variation of the problem appears in [17], [58], [59], where the evolving objects (and therefore the main objects of study) are the concepts; this viewpoint is quite different from the standard one, in which the evolving object is an ontology as a whole.

E. Ontology Versioning

Once the actual changes have been performed, ontology versioning comes into play. Ontology versioning typically involves the storage of both the old and the new version of the ontology and takes into account identification issues (i.e., how to identify the different versions of the ontology), the relation between different versions (i.e., a tree of versions resulting from the various ontology modifications) as well as compatibility information (i.e., information regarding the compatibility of any pair of ontology versions).

Several non-trivial problems are associated with this task. For example, any ontology versioning algorithm should be based on some type of identification mechanism to differentiate between various versions of an ontology, but it is not always clear when two ontologies constitute different versions. Should any change in the file that stores the ontology constitute the creation of a new version? When a concept specification changes, but the new specification is semantically equivalent to the original one, does this constitute a new version? More generally, when the ontology changes syntactically, but not semantically, does this constitute a new version? These and similar problems are dealt with in [25], [31].

Another desirable property of an ontology versioning system is the ability to allow transparent access to different versions of the ontology, by automatically relating versions with dependent elements [30]. Other issues involved is the so-called "packaging of changes" [31] as well as the different types of compatibility and how these are identified [30].

Another related problem is the introduction of a certain version relation between ontological elements (such as classes) that appear in different versions of the ontology and the properties that such a relation should have. This relation is called a change specification in [30] and its role is to make the relationship between different versions of ontological elements explicit. Using this relation, one can identify the changes that any given element went through between different versions; in addition, a version relation should include certain meta-data regarding these changes [31]. In [52] this relation is stored using a version log which is actually a specially designed ontology containing the different versions of each element, as well as the relation between them and some related meta-data. Similar considerations led to the definition of migration specifications [60], which associate concepts between different versions of an ontology after a change has been performed.

F. The Current State of the Art in Ontology Versioning

As an aid to the task of ontology versioning, certain tools have been developed which automatically identify the differences between ontology versions; unfortunately, most such tools provide information at the level of atomic changes [32]. PROMPTDIFF [45] uses certain heuristics to compare different versions of ontologies and outline their differences, by producing a structural diff between them. OntoView [31] contains a tool similar to PROMPTDIFF, whose output is a certain ontology of changes.

A survey on the different ways that can be used to represent a set of changes, as well as the relation and possible interactions between such representations can be found in [32]; in the same paper, another ontology of changes is proposed, containing both atomic and complex operations. A similar ontology of changes is proposed in [52], where the changes are identified through a version log stored in this ontology of changes.

A method to identify compatibility between versions is presented in [24], [25] where the SHOE language [36] is used to make backward compatibility between versions explicit and determinable by a computer agent. This is an indirect approach to the problem of ontology versioning, because it allows the computer agent to determine autonomously which version to use, as opposed to [30], [31], where a more direct and centralized path is taken. In [26], a temporal logic approach is used to allow access in different versions of an ontology.

IV. Ontology Mapping, Morphism, Alignment, Articulation and Translation

A. General Discussion

Work related to these areas tries to mitigate the problems caused by the heterogeneity of the Semantic Web. The general motivation for these research fields is that different ontologies (and sources of information based upon different ontologies) generally use different terminology, different representation languages and different syntax to refer to the same or similar concepts. A nice list of use cases where this heterogeneity may cause problems can be found in [9].

The obvious solution to this problem is the provision of a set of translation rules of some kind that will allow us to nullify these terminological differences. To put it simply, the goal of the whole process is to make two ontologies refer to same entities using the same name and to different entities using different names. For example, we should be able to identify that the concepts RESEARCHER and RESEARCH_STAFF_MEMBER that appear in two different ontologies refer to the same real-world concept, i.e., the class of researchers. We should also be able to differentiate between two different uses of the entity CHAIR, as it could refer to the class of chairs (as a furniture) in one ontology and to the people forming a Workshop's Chair in another.

Even though these research fields basically deal with the same problem (i.e., heterogeneity resolution), they can be identified based on the type of translation rules that is produced at the output. Due to the close relationship between these areas, sometimes the term ontology alignment (e.g., in [9]) or ontology mapping (e.g., in [27]) is used to refer collectively to all of them. In this section, we will try to disambiguate the situation; most of the material for this section is taken from [9] and [27].

B. Definitions

The term *ontology mapping refers to the task of relating the signatures of two ontologies that share the same domain of discourse in such a way that the mathematical structure of ontological signatures and their intended interpretations, as specified by the ontological axioms, are respected*. The result of an ontology mapping algorithm is a collection of functions on ontological signatures. A similar (and equivalent) definition appears in [4], where ontology mapping is defined as a (declarative) specification of the semantic overlap between two ontologies, which can be either one-way (injective) or two-way (bijective).

This definition restricts the mappings to ontological signatures. A more ambitious and interesting approach would be to create mappings that deal with both the signatures and the axioms of the ontologies. The term *ontology morphism* refers to that approach, i.e., *the development of a collection of functions that relate both ontological signatures and axioms*. Notice that ontology morphism, unlike the other fields discussed in this section, is not restricted to the ontology signature only, but covers the ontological axioms as well.

In ontology mapping and morphism the ontologies are related via functions; an interesting, and more general, alternative is by means of a relation. *The task of finding relationships between signature entities belonging to two different ontologies is called ontology alignment*. So the output of ontology alignment is a binary relationship between the ontological signatures. This approach is more liberal, allowing greater flexibility, so it is more commonly used in practice.

A binary relationship could be decomposed into a pair of total functions from a common intermediate source; therefore, the alignment of two ontologies could be described by means of a pair of ontology mappings from a common intermediate ontology. We use the term *ontology articulation* to refer to *the process of determining the intermediate ontology and the two mappings to the initial ontologies*.

Finally, the term *ontology translation* is used in the literature with two different meanings. Under one understanding, ontology translation refers to *the process of changing the formal representation of the ontology from one language to another*. This changes the syntactic (only) form of the axioms, but not the signature of the ontology. Under the second understanding, ontology translation refers to a *translation of the signature*, in a manner similar to that of ontology mapping. The difference between ontology mapping and ontology translation is that the former specifies the functions that relate the two ontologies' signatures, whereas the latter applies these functions to actually implement the mapping.

C. Methodology and the Current State of the Art

The methods commonly used to address the problem of heterogeneity include studying the taxonomic or mereological structure of the entities, evaluating name similarities (where the names are compared as strings) and so on. Other methods use a thesaurus to study the linguistic similarities of names, use semantic approaches, or determine the similarity based on the instances of each entity. The final similarity evaluation may also be affected by the evaluation of the similarity of the entities' neighborhood. In real systems, a combination of some of these approaches with some kind of human intervention usually works best. A detailed classification and description of these methods can be found in [9].

Two popular systems that deal with heterogeneity are PROMPT [46], [47] (originally called SMART [48]) and Chimaera [39]. In [10], the term ontology matching is used to refer to an ontology mapping algorithm based on the linguistic properties of terms, using a thesaurus based on WordNet [41]. In [53], a certain string metric is proposed to evaluate name similarities of elements in different ontologies, upon which an ontology alignment algorithm could be based. Some thoughts on the issue of heterogeneity in the context of the SHOE language can be found in [24], [25]. An interesting method of improving the results of an alignment process, which exploits user validation combined with machine learning techniques, can be found in [8].

In [42], a probabilistic technique is used towards this aim; the final similarity evaluation of this ontology mapping algorithm is affected by the similarity probabilities of each entity's neighborhood, improving the initial mapping result. Another method based on probabilistic analysis, which takes into account uncertainty issues in the mapped ontologies can be found in [49]. A general-purpose approach to the problem of translation is described in [6]. A much more extensive list of systems and works related to these research areas can be found in [4], [9], [27]; a relevant evaluation appears in [1].

Unfortunately, heterogeneity resolution in ontologies still relies on human intervention; however, the process has to be automatic in order to be practical [27]. In this direction, advances in the field of natural language processing will probably help researchers gain a better understanding on the processes behind automatic heterogeneity resolution [27].

D. Heterogeneity Resolution and Ontology Change

Notice that most of the fields studied in this section do not directly modify any ontology, but provide translation rules that relate ontologies. As a result, many would argue that these research areas should not be considered subfields of ontology change. We believe otherwise, for two reasons. First, heterogeneity resolution constitutes a prerequisite for successful ontology change, as it makes no sense to try to change an ontology in response to new information unless both the ontology and the new information are formulated using the same terminology, language and syntax. So, it makes practical sense to study these fields along with the problem of ontology change.

Second, heterogeneity resolution implicitly requires the modification of an ontology, so it is really a subfield of ontology change in the wide sense of the term used in this paper. Indeed, consider two agents with heterogeneous ontologies that need to communicate and some translation rules allowing this communication. In this particular example, the "need for change" is the need for communication. The rules produced do not directly modify any ontology; however, they allow each agent to change the other agent's ontology locally to fit his own terminology, language and syntax. So the change in this case is made on-the-fly by each agent. In this sense, we could consider ontology mapping and the other fields studied in this section to be subfields of ontology change that simply provide us with a method to change an ontology (even though no change is performed explicitly).

V. Ontology Integration and Merging

A. Discussion and Definitions

Both ontology integration and merging refer to the construction of a new ontology based on the information found in two or more source ontologies; yet, the two terms refer to slightly different research areas. Unfortunately, the exact meaning of each term is not clear in the literature, as they are often used interchangeably [51], causing a certain amount of confusion.

In [46], [47] ontology merging is defined as the process of creating a new, coherent ontology that includes information from two or more source ontologies; this is implicitly assumed to include the process of resolving any possible heterogeneities between the merged ontologies. In these papers, ontology merging and alignment are understood as variations of the same problem, the only difference being that ontology merging results in the creation of a new ontology, whereas in ontology alignment the merged ontologies persist, with links established between them.

A similar use of the term can be found in [39], whereas, in [25], the same research area is described using the term ontology integration. According to [35], ontology merging amounts to making sure that different agents use the same terms in identical ways (in a manner similar to ontology alignment). In [27] ontology integration is defined as the process of combining ontologies to build new ones, but whose respective signatures are usually not interpreted in the same domain of discourse. In [5] the same term is used to refer to the process of combining a number of local ontologies in order to build a global one, with the purpose of being able to answer queries over the local ontologies using the global one and the mappings between these ontologies.

Here, we will define these terms along the lines of [51], which was an attempt to disambiguate between different uses of the term ontology integration. Three different uses of the term were identified in that paper. The first refers to the *composition of ontologies covering loosely related (i.e., similar) domains*; this is mainly used when building a new ontology that covers all these domains. The term *ontology integration* has been reserved for this process.

The second use of the word refers to the *combination of ontologies covering highly overlapping or identical domains*; this process is used to fuse ontologies that contain information about the same subject into one large (and hopefully more accurate) ontology. The term *ontology merging* was attached to this interpretation.

Finally, the third use of the term integration refers to the *development of an application that uses one or more ontologies*; the more appropriate term *ontology use* was reserved for this process. In this paper, we focus on the first two research areas, namely ontology integration and merging.

B. Differences Between Integration and Merging

There are certain subtle differences between the processes of integration and merging. Ontology integration is mainly applied when the main concern is the reuse of other ontologies. The domain of discourse of the new ontology is usually more general than the domain of any of the source ontologies and integration often places the different (source) ontologies in different modules that comprise the resulting ontology.

On the other hand, in ontology merging, the focus is on creating an ontology that combines information on a given topic from different sources. In this case, the information from the source ontologies is greatly intermingled, so it is difficult to identify the part(s) of the final ontology that resulted from each source ontology. A more detailed discussion can be found in [51].

C. Integration, Merging and Heterogeneity Resolution

It is a common practice in the literature (e.g., [4], [25], [46], [51]), to consider heterogeneity resolution to be an internal part of ontology merging or integration. This is a reasonable choice, because in most cases the fused ontologies come from different sources, so they are generally heterogeneous in terms of vocabulary, syntax, representation etc. Therefore, the task of resolving any heterogeneities between the source ontologies constitutes a major part of the task of ontology merging (or integration). This is mostly true in merging, where the domain of discourse is (almost) identical. This has led to even more confusion on the exact meaning of the terms, as several researchers consider ontology merging (or integration) and alignment to be variations of the same problem (e.g., [35], [46]).

However, it should be clear that simply resolving the heterogeneity issues between two ontologies is not sufficient for successful integration (or merging); recall that different ontologies may encode different viewpoints regarding the real world, thus several conceptual differences are bound to exist, even if the same terminology is used. This is reminiscent of how beliefs held by different people are often different (and in some cases contradictory), even if a common terminology is agreed upon.

Similarly, modeling conventions and choices may be different; one example of modeling choice that often depends on personal taste or convention is whether to model a certain distinction between similar elements by introducing separate classes or by introducing a qualifying attribute relation in one class [6]. Such modeling differences need to be taken into account when selecting what to keep from each ontology during the integration or merging process. Reckless inclusion of ontology elements and axioms from the source ontologies (even when homogeneous) is likely to lead to a problematic, invalid or inconsistent ontology.

D. State of the Art in Ontology Integration and Merging

According to [6], the process of merging can be broken down in five steps. During the first step, we identify the semantic overlap between the source ontologies; during the second, we devise ways (transformations) to bring the sources into mutual agreement in terms of terminology, representation etc. In the third step, we apply these transformations, so we can now take the union of the sources (fourth step). The final step consists of evaluating the resulting ontology for consistency, uniformity, redundancy, quality of conceptualization etc; this evaluation might force us to repeat some or all of the above steps. The tool described in [6] facilitates the design and implementation of the transformations used in the merging process (second and third step).

The main tools used for ontology merging are PROMPT [46] and Chimaera [39]. These tools use a semi-automatic approach focused on suggesting how elements from the source ontologies should be merged in the resulting ontology. The final choice relies on the ontology engineer. Some ideas on ontology merging (called integration there) in the context of the SHOE language can be found in [25]; however, [25] is focused on the part of merging that deals with heterogeneity resolution. In [5], an interesting theoretical framework for ontology integration is defined, focusing on the creation of mappings between the source and the resulting ontologies and how these mappings can be exploited for query answering. An interesting theoretical approach to ontology merging can be found in [2], whereas in [47] some interesting connections of object-oriented programming with the problem of ontology merging are uncovered. The FCA-MERGE algorithm [57] performs ontology integration in a very efficient way, but is based on certain strong assumptions. A more detailed list of tools and systems related to the problem can be found in [4], [51].

Even though the problem of evaluating ontology merging techniques is still open in AI [57], certain comparison attempts have been made. In [34], the authors perform a comparison between PROMPT and Chimaera in the context of bioinformatics. In [46], the same two tools are compared with the generic Protégé-2000 [43]. Furthermore, [39] compares the efficiency of ontology merging with a simple plain-text editor, merging with the Ontolingua editor and merging with the specialized tool Chimaera, which is described in the same paper. These comparisons are made from a certain standpoint; a general, objective comparison is difficult, as it is not clear how the utility of such tools could be measured [39].

VI. Conclusion

In this paper, we performed a shallow, but broad literature review covering all the diverse types of ontology change. This allowed us to fix a terminology in an area that is plagued by underspecified and confusing terms which are used with different meanings by different researchers. This terminology was not introduced in an arbitrary manner, but was based on similar previous attempts (like [27], [51]) and on the most common uses of the terms in the literature. We hope that our work will prove helpful towards the clarification of the boundaries and relations between the various fields and will serve as a starting point for researchers interested in any of the many facets of ontology change. A summary of the results of our study can be found in table I at the end of this paper.

Acknowledgment

The authors are grateful to Panos Constantopoulos, Vassilis Christophides and Nicolas Spyratos for helpful comments in an earlier draft of this work. This work was carried out during the first author's tenure of an ERCIM "Alain Bensoussan" Fellowship Programme.

References

[1] P. Avesani, F. Giunchiglia, M. Yatskevich. A Large Scale Taxonomy Mapping Evaluation. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 67-81, 2005.

[2] T. Bench-Capon, G. Malcolm. Formalizing Ontologies and Their Relations. In Proceedings of the 16th International Conference on Database and Expert Systems Applications (DEXA-99), pp. 250-259, 1999.

[3] T. Berners-Lee, J. Hendler, O. Lassila. The Semantic Web. Scientific American, 284(5):34-43, 2001.

[4] J. de Bruijn, F. Martin-Recuerda, D. Manov, M. Ehrig. D4.2.1: State of the Art Survey on Ontology Merging and Aligning. SEKT Deliverable, 2004.

[5] D. Calvanese, G. De Giacomo, M. Lenzerini. A Framework for Ontology Integration. In I. Cruz, S. Decker, J. Euzenat, D. McGuinness (eds). The Emerging Semantic Web. Selected Papers from the First Semantic Web Working Symposium, pp. 201-214. IOS Press, 2002.

[6] H. Chalupsky. OntoMorph: A Translation System for Symbolic Knowledge. In Proceedings of the 7th International Conference on Knowledge Representation and Reasoning (KR-00), 2000.

[7] A.J. Duineveld, R. Stoter, M.R. Weiden, B. Kenepa, V.R. Benjamins. WonderTools? A Comparative Study of Ontological Engineering Tools. International Journal of Human-Computer Studies, 52(6):1111-1133, 2000.

[8] M. Ehrig, S. Staab, Y. Sure. Bootstrapping Ontology Alignment Methods with APFEL. In Proceedings of the 4th International Semantic Web Conference, 2005.

[9] J. Euzenat, T. Le Bach, J. Barrasa, P. Bouquet, J. de Bo, R. Dieng, M. Ehrig, M. Hauswirth, M. Jarrar, R. Lara, D. Maynard, A. Napoli, G. Stamou, H. Stuckeschmidt, P. Shvaiko, S. Tessaris, S. van Acker, I. Zaihrayeu. D2.2.3: State of the Art on Ontology Alignment. SEKT Deliverable, 2004.

[10] A. Ferrara. Methods and Techniques for Ontology Matching and Evolution in Open Distributed Systems. In Proceedings of the 16th International Conference on Advanced Information Systems Engineering, 2004.

[11] G. Flouris. On Belief Change and Ontology Evolution. Doctoral Dissertation, Department of Computer Science, Univ. of Crete, 2006.

[12] G. Flouris, D. Plexousakis. Bridging Ontology Evolution and Belief Change. In Proceedings of the 4th Hellenic Conference on Artificial Intelligence, 2006.

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

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

[15] G. Flouris, D. Plexousakis, G. Antoniou. Generalizing the AGM Postulates: Preliminary Results and Applications. In Proceedings of the 10th International Workshop on Non-Monotonic Reasoning, 2004.

[16] G. Flouris, D. Plexousakis, G. Antoniou. On Applying the AGM Theory to DLs and OWL. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 216-231, 2005.

[17] N. Foo. Ontology Revision. In G. Ellis, R. Levinson, W. Rich, J. Sowa (eds). Proceedings of the 3rd International Conference on Conceptual Structures (ICCS-95), Lecture Notes in Artificial Intelligence (LNAI), Volume 954, Springer-Verlag, pp. 16-31, 1995.

[18] E. Franconi, F. Grandi, F. Mandreoli. A Semantic Approach for Schema Evolution and Versioning in Object-Oriented Databases. In Proceedings of the 6th International Conference on Rules and Objects in Databases, 2000.

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

[20] P. Gärdenfors. The Dynamics of Belief Systems: Foundations Versus Coherence Theories. Revue Internationale de Philosophie 44, 1992.

[21] T.R. Gruber. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5(2):199-220, 1993.

[22] P. Haase, L. Stojanovic. Consistent Evolution of OWL Ontologies. In Proceedings of the 2nd European Semantic Web Conference (ESWC-05), 2005.

[23] P. Haase, Y. Sure. D3.1.1.b State of the Art on Ontology Evolution. SEKT Deliverable, 2004.

[24] J. Heflin, J. Hendler. Dynamic Ontologies on the Web. In Proceedings of the 17th National Conference on Artificial Intelligence, 2000.

[25] J. Heflin, J. Hendler, S. Luke. Coping with Changing Ontologies in a Distributed Environment. In Proceedings of the Workshop on Ontology Management of the 16th National Conference on Artificial Intelligence (AAAI-99), WS-99-13, AAAI Press, pp. 74-79, 1999.

[26] Z. Huang, H. Stuckenschmidt. Reasoning with Multi-version Ontologies: A Temporal Logic Approach. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 398-412, 2005.

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

[28] H. Katsuno, A.O. Mendelzon. On the Difference Between Updating a Knowledge Base and Revising It. Technical Report on Knowledge Representation and Reasoning, Univ. of Toronto, KRR-TR-90-6, 1990.

[29] W. Kim, H.T. Chou. Versions of Schema for Object-Oriented Databases. In Proceedings of the 14th International Conference on Very Large Data Bases (VLDB-88), pp. 148-159, 1988.

[30] M. Klein, D. Fensel. Ontology Versioning on the Semantic Web. In Proceedings of the International Semantic Web Working Symposium, 2001.

[31] M. Klein, D. Fensel, A. Kiryakov, D. Ognyanov. Ontology Versioning and Change Detection on the Web. In Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, 2002.

[32] M. Klein, N.F. Noy. A Component-Based Framework for Ontology Evolution. In Proceedings of the IJCAI-03 Workshop on Ontologies and Distributed Systems, CEUR-WS, vol. 71, 2003.

[33] S.C. Lam, D.H. Sleeman, W. Vasconselos. ReTAX++: A Tool for Browsing and Revising Ontologies. In Poster Proceedings of the 4th International Semantic Web Conference (ISWC-05), PID-33, 2005.

[34] P. Lambrix, A. Edberg. Evaluation of Ontology Merging Tools in Bioinformatics. In Proceedings of the 8th Pacific Symposium on Biocomputing, pp. 589-600, 2003.

[35] K. Lee, T. Meyer. A Classification of Ontology Modification. In Proceedings of the 17th Australian Joint Conference on Artificial Intelligence, 2004.

[36] S. Luke, L. Spector, D. Rager, J. Hendler. Ontology-based Web Agents. In Proceedings of the 1st International Conference on Autonomous Agents, 1997.

[37] A. Maedche, B. Motik, L. Stojanovic, R. Studer, R. Volz. An Infrastructure for Searching, Reusing and Evolving Distributed Ontologies. In Proceedings of the 12th International World Wide Web Conference, 2003.

[38] M. Magiridou, S. Sahtouris, V. Christophides, M. Koubarakis. RUL: A Declarative Update Language for RDF. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 506-521, 2005.

[39] D. McGuiness, R. Fikes, J. Rice, S. Wilder. An Environment for Merging and Testing Large Ontologies. In Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR-00), 2000.

[40] T. Meyer, K. Lee, R. Booth. Knowledge Integration for Description Logics. In Proceedings of the 7th International Symposium on Logical Formalizations of Commonsense Reasoning, 2005.

[41] G.A. Miller. WordNet: A Lexical Database for English. Communications of the ACM (CACM), 38(11):39-41, 1995.

[42] P. Mitra, N.F. Noy, A.R. Jaiswal. OMEN: A Probabilistic Ontology Mapping Tool. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 537-547, 2005.

[43] N.F. Noy, R. Fergerson, M. Musen. The Knowledge Model of Protégé-2000: Combining Interoperability and Flexibility. In Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management: Methods, Models, and Tools (EKAW-00), pp. 1732, 2000.

[44] N.F. Noy, M. Klein. Ontology Evolution: Not the Same as Schema Evolution. Knowledge and Information Systems, 6(4):428-440, 2004.

[45] N.F. Noy, S. Kunnatur, M. Klein, M.A. Musen. Tracking Changes During Ontology Evolution. In Proceedings of the 3rd International Semantic Web Conference (ISWC-04), 2004.

[46] N.F. Noy, M.A. Musen. Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00), 2000.

[47] N.F. Noy, M.A. Musen. An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support. In Proceedings of the Workshop on Ontology Management at 16th National Conference on Artificial Intelligence (AAAI-99), 1999.

[48] N.F. Noy, M.A. Musen. SMART: Automated Support for Ontology Merging and Alignment. In Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management, 1999.

[49] R. Pan, Z. Ding, Y. Yu, Y. Peng. A Bayesian Network Approach to Ontology Mapping. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 563-577, 2005.

[50] R.J. Peters, T. Ozsu. An Axiomatic Model of Dynamic Schema Evolution in Objectbase Systems. ACM Transactions on Database Systems, 22(1):75-114, 1997.

[51] H.S. Pinto, A. Gomez-Perez, J.P. Martins. Some Issues on Ontology Integration. In Proceedings of the Workshop on Ontologies and Problem-Solving Methods (KRR5), 1999.

[52] P. Plessers, O. De Troyer. Ontology Change Detection Using a Version Log. In Proceedings of the 4th International Semantic Web Conference, 2005.

[53] G. Stoilos, G. Stamou, S. Kollias. A String Metric for Ontology Alignment. In Proceedings of the 4th International Semantic Web Conference (ISWC-05), pp. 624-637, 2005.

[54] L. Stojanovic, A. Maedche, B. Motik, N. Stojanovic. User-driven Ontology Evolution Management. In Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, 2002.

[55] L. Stojanovic, A. Maedche, N. Stojanovic, R. Studer. Ontology Evolution as Reconfiguration-Design Problem Solving. In Proceedings of the 2nd International Conference on Knowledge Capture, 2003.

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

[57] G. Stumme, A. Maedche. Ontology Merging for Federated Ontologies on the Semantic Web. In Proceedings of the International Workshop on Foundations of Models for Information Integration (FMII-01), 2002.

[58] R. Wassermann. Revising Concepts. In Proceedings of the 5th Workshop on Logic, Language, Information and Communication (WoLLIC-98), 1998.

[59] R. Wassermann, E. Fermé. A Note on Prototype Revision. In Spinning Ideas, 1999.

[60] Z. Zhang, L. Zhang, C.X. Lin, Y. Zhao, Y. Yu. Data Migration for Ontology Evolution. In Poster Proceedings of the 2nd International Semantic Web Conference (ISWC-03), 2003.

TABLE I

SUMMARY OF THE VARIOUS SUBFIELDS OF ONTOLOGY CHANGE

Ontology Mapping	**Purpose:**	Heterogeneity resolution, interoperability
	Input:	Two (heterogeneous) ontologies
	Output:	A mapping between signatures
	Properties:	Output identifies related signature entities
Ontology Morphism	**Purpose:**	Heterogeneity resolution, interoperability
	Input:	Two (heterogeneous) ontologies
	Output:	Mappings between signatures and axioms
	Properties:	Output identifies related signature entities and axioms
Ontology Alignment	**Purpose:**	Heterogeneity resolution, interoperability
	Input:	Two (heterogeneous) ontologies
	Output:	A relation between signatures
	Properties:	Output identifies related signature entities
Ontology Articulation	**Purpose:**	Heterogeneity resolution, interoperability
	Input:	Two (heterogeneous) ontologies
	Output:	An intermediate ontology and mappings between the signatures of the intermediate ontology and each source
	Properties:	Output is equivalent to a relation identifying related signature entities
Ontology Translation (first reading)	**Purpose:**	Use a different representation language
	Input:	Ontology and target representation language
	Output:	Ontology expressed in the target language
	Properties:	Produces an equivalent ontology, if possible
Ontology Translation (second reading)	**Purpose:**	Implementation of a signature mapping
	Input:	An ontology and a mapping
	Output:	An ontology
	Properties:	Implements the mapping
Ontology Evolution	**Purpose:**	Apply changes (domain/conceptualization)
	Input:	Ontology and change operation(s)
	Output:	An ontology
	Properties:	Implements change(s) to the source ontology
Ontology Versioning	**Purpose:**	Transparent access to different versions
	Input:	Different versions of an ontology
	Output:	A versioning system
	Properties:	Version ids identify versions; transparent access to versions; compatibility determination
Ontology Integration	**Purpose:**	Fuse ontologies; similar domains
	Input:	Two ontologies (covering similar domains)
	Output:	An ontology
	Properties:	Fuses knowledge to cover a broader domain
Ontology Merging	**Purpose:**	Fuse ontologies; identical domains
	Input:	Two ontologies (covering identical domains)
	Output:	An ontology
	Properties:	Fuses knowledge to describe the domain more accurately