Conceptual Ontology Intersection for Mapping and Alignment of Ontologies

Anne Håkansson and Dan Wu

Department of Software and Computer Systems
KTH Royal Institute of Technology
Forum 100
SE-164 40 Kista, Sweden
{annehak,dwu}@kth.se

Abstract. Combining ontologies can enrich knowledge within a domain and support the development and use of advanced services. This requires matching and combining the relevant ontologies for specific services, which can be supported by mapping and alignment of several ontologies. However, these techniques are not enough since the ontologies are often heterogeneous and difficult to combine. To overcome these problems, a conceptual ontology intersection is provided to map and align contents of the ontologies. This intersection is a conceptual ontology bridge between ontologies and contains parts from the involved ontologies. The contents are extracted by syntactic mapping and synonym alignment using an ontology repository, a rule base and a synonym lexicon using agents. The result is a set of concepts that together constitute the intersection, which is used for combining new incoming ontologies and, thereby, providing complex services.

1 Introduction

An ontology is a body of formal knowledge representation used in all kinds of domains. The ontology is a specification of a conceptualisation with objects, concepts and other entities and relations among them [1]. The ontology is used for enabling knowledge sharing and knowledge reuse, as well as, describing domain of discourse via systems that define a set of representational terms. The contents of the ontologies are human-defined texts describing the meaning of the names, the formal axioms that constrain the interpretation and the use of these terms. Hence, concepts of a domain is represented in a ontology as a model of the domain with which it is possible to reason about the entities within that domain.

Although, an ontology is a formal representation [2] that provides a semantic representation of a domain containing definitions of classes, relations and functions [3], the ontology cannot lend itself as an easy and direct solution to ontology mapping and integration. The problem is that using a well-formed description logic language for ontologies [4] is not enough since the semantic differences of ontologies are not reduced. The differences often depend on the ontology developers' perspectives and

A. Håkansson & R. Hartung (Eds.): *Agent & Multi-Agent Syst. in Distrib. Syst.*, SCI 462, pp. 105–124.
DOI: 10.1007/978-3-642-35208-9_6

terminologies. The developers of ontologies most certainly have divergent perspectives of a domain and various purposes with the developed ontologies. Moreover, the terminology, chosen by the different developers, can be distinct. Therefore, when matching and combining the ontologies, syntactically and semantically, all these challenges must be encountered.

To support tasks envisaged by a distributed environment, one single ontology is not enough. Instead multiple ontologies need to be accessed from several applications [5]. There exist a lot of ontologies that taken together can enrich knowledge within a domain. Beside supporting knowledge sharing and reuse, the ontologies support the development and use of advanced services. This requires matching and combining the relevant ontologies for the specific services, which can be carried out by mapping and alignment of these ontologies. Mapping can provide a meta-level from which several ontologies can be accessed, corresponding concepts mapped and alignment applied to establish relations between equivalent the vocabularies of two ontologies and, thereby, increase the set of concepts in the intersection.

When several ontologies are involved in reasoning on the semantic web, as well as, combining ontologies to provide services on the distributed systems, the heterogeneousness of the ontologies becomes a problem. The ontology heterogeneities occur on different levels, such as syntactic, terminological, conceptual and semiotic levels [6] but also at semantic and pragmatic levels. Some syntactic matching solutions have been promising but the semantic mapping among the ontologies is still a difficult issue. Moreover, mapping on the syntactic level is not enough to obtain a useful combination of several ontologies for advanced services.

The research, presented in this chapter, tackles the heterogeneous problem by introducing a conceptual ontology intersection that serve as a bridge between involved ontologies. The extraction of the ontologies' contents is performed in several steps. A syntactic mapping is carried out by extracting the syntax of each ontology, comparing those by matching between the ontologies and storing the result in an intersection meta-model. Then, this intersection meta-model is used for handling semantics by synonym alignment. Synonyms are collected, aligned and stored as conceptual ontology intersections [7]. The result is intersections with concepts and synonyms that correspond to the parts that are found in both ontologies, which can be used for providing advanced services.

The chapter is structured as follows: Section two gives a brief description of constituents of ontologies and Section three is about methods for combining several ontologies. Section four describes a conceptual ontology intersection resulting from mapping and aligning several ontologies and violation handling. Section five presents related work and section six conclude this chapter.

2 Constituents of Ontologies

An ontology is defined as a set of representational primitives with which it is possible to model a domain of knowledge or discourse [7]. In other words, ontology is "a catalog of the types of things that is assumed to exist in a domain of interest D from the

perspective of a person who uses a language L for the purpose of talking about D.", [8, 9]. Hence, the ontology states characteristics of a domain, which is known to be true about that domain [10].

To effectively use ontologies, a well-designed and well-defined ontology language should be used for development [10]. Some ontology languages, such as OWL 2, are based on description logics, which is monotonic and adhere to the open world assumption [11]. With the open world assumption and monotonic reasoning, reasoning can be used to derive implied relations in the ontology. With these assumptions, when adding anything during the reasoning, it does not reduce the set of consequences. Instead, the number of consequences will remain the same or increase. Applying the reasoning on the contents of the original ontology without changing anything is an important feature in the research of this paper since it ensures the quality of an ontology, which, for example, is avoiding non-contradictory concepts.

There is a set of constituents with which ontologies are built. These fundamental building blocks are individuals (objects), attributes (properties), classes, relations, function terms, axioms, restrictions, rules, and events [12]. The individuals are concrete objects in the domain and to describe these objects, they are related to attributes that define properties with features or characteristics. These attributes, themselves, can be objects or classes. Objects denote a concrete or abstract thing; class notation describes individuals as collections of objects. The classes are concepts describing abstract groups and sets of objects, which are defined by values of aspects constrained as being members of a specific class [13].

To specify how objects are related, relations are applied. These relations can be a of a particular type, or class, specifying in what sense objects are related. From relations, complex structures, called function terms, are formed to be used in place of an individual term in a statement.

The ontology contains a set of axioms and facts in the domain [14], which together comprise the overall theory that an ontology describes. These axioms apply constraints on sets of classes and the types of relations permitted between them. The vocabulary can be defined with class axioms, and properties, such as ObjectProperty, DataProperty, AnnotationProperty, Datatype, NamedIndividual and AnonymousIndividual. Class axioms allow establishing class relations, for example, the subClassof axiom states that instances in one class expression are also instances of another class. The classes can also express the semantically equivalences of the classes.

The ObjectProperty connects individuals in the ontology and the ObjectPropertyDomain expresses that the individuals are from the domain, specified by the class connected by that property. By using the object property axioms, domain and range properties can be described. For example, the ObjectPropertyRange expresses that individuals are from the range of the class connected by the property. This set of entities constitutes the signature of the ontology, which will be used for reasoning and mapping.

The rules are production rules in Horn-clause form that describe the logical inferences drawn from premisses in a particular form [15]. Rules are useful for representing contingent features, such as, the relations between preconditions and postconditions [5]. They can capture a significant fragment of ontologies, by capturing simple

frame axioms and more expressive property axioms. The fragments permit, for example, stating that a class is a subclass of a class, which is useful for the research presented in this chapter. Drawing conclusions from the classes assure the quality of the mapping result.

Moreover, the ontologies contain the representation of entities, ideas, and events, along with their properties and relations. The events can be, for example, people, organizations, places, addresses, artifacts, phone numbers, and dates but can also be transactions. Also these parts can be used in the inference process to make the reasoning more reliable.

3 Methods for combining Contents of the Ontologies

Many methods have been developed for extracting and combining ontologies. Some of the most common are matching, mapping, alignment, merging, and integrating. Depending on the expected result of the extraction and/or combination, these methods can be beneficial.

Ontology *matching* aims at finding correspondences between related entities of different ontologies, syntactically and semantically [16]. The matching process takes ontologies as input, which consist of a set of entities like classes, properties, and determines correspondences as output, which are relations holding between the entities [17]. These correspondences can be equivalence, consequence, subsumption, or disjointness between ontology entities [16]. The matching can involve matching of the labels of the ontologies measuring the similarities, or distances, among nodes using matching estimations, which estimate the distance differences [18]. Unfortunately, there are different heterogeneities, such as syntactic heterogeneity, terminological heterogeneity, conceptual heterogeneity and semiotic heterogeneity, and even minor name differences and small structure variations can lead to matching problems.

Ontology *mapping* handles a part of the more advanced tasks concerning the alignment and merging of ontologies [5]. Ontology mapping is the task of relating the vocabulary of two ontologies sharing the same domain of discourse [5]. The mapping is mostly concerned with the representation of correspondences [19] where the structure of ontologies and their interpretations are specified as ontological axioms, which are preserved although mapping. Mapping one ontology to another ontology means that there is a corresponding concept, for each concept in one ontology in the other ontology that has the same or similar semantics [17]. The mapping can be partial since there might be concepts in a ontology that have no equivalents in the other ontology [20]. The mapping needs to map the contents of the ontologies regardless of the format of the concepts.

Ontology *alignment* is the task of establishing binary relations between the vocabularies of two ontologies [5] but does not depend on the choices of names in either ontology [20]. Ontology alignment is the task of creating links between two original ontologies and they are equivalent if a concept or relation in one ontology maps to a concept or relation in the other ontology. Ontology alignment is usually carried out if the sources are consistent with each other but are kept separate [21] and when they

have complementary domains. Before two ontologies can be aligned, it may be necessary to introduce new subtypes or supertypes of concepts or relations in either one of the ontologies in order to provide suitable targets for alignment [20]. No other changes to the axioms, definitions, proofs, or computations in either ontology are made during the process of alignment. Alignment is the weakest form of integration since it requires minimal change [20]. Hence, it is useful for classification and information retrieval, but it does not support deep inferences [20].

Ontology *merging* is often used when the goal is to create a single coherent ontology that includes the information from all the sources and when the sources must be made consistent and coherent with one another but kept separately [21]. The merge is based on the discovery of the correspondences [19] by finding commonalities between two different ontologies and deriving a new ontology [20]. The new ontology may replace the merged ontologies but it can also be used as an intermediary. When performing ontology merging, a new ontology is created which is the union of the source ontologies, based on the correspondences between the ontologies [19]. The merged ontology captures all the knowledge from the original ontologies. The challenge in ontology merging is to ensure that all correspondences and differences between the ontologies are reflected in the merged ontology.

Ontology *integration* is the process of finding commonalities between two different ontologies and deriving a new ontology that facilitates the interoperability of the systems that are based on the integrated ontologies [22]. The term ontology integration has been used when the tasks are building new ontologies reusing other available ontologies, and integrating ontologies into applications but also merging ontologies into a single one that unifies them [24]. Integration is often used interchanged with ontology merging but a difference is that integration builds a new ontology whereas merging inserts one ontology into another ontology. The ontology integration is the process of building an ontology by assembling and extending other already existing ontologies, which becomes parts of the resulting ontology [24]. The new ontology may replace the other ontologies or become an intermediary part. There are three levels of integration: Alignment, mentioned above, Partial Compatibility and Total Compatibility (also called ontology merge). Partial Compatibility is an alignment of two ontologies that supports equivalent inferences and computations on all equivalent concepts and relations [25]. If two ontologies are partially compatible, any inference or computation, which can be expressed in one ontology by the aligned concepts and relations, can be translated to an equivalent inference or computation in the other ontology [25].

Since ontology mapping is the task of relating the vocabulary of two ontologies and concerned with the representation of correspondences specified as ontological axioms, mapping is used for finding the similar concepts. The corresponding concepts that have the same or similar syntax and semantics are recorded for further mapping tasks. When it comes to context for the ontologies, ontology alignment is applied because it is used to create links between two original ontologies, i.e., if they are equivalent. The ontologies are expected to be consistent and be in complementary domains but will be kept separately. To align the ontologies, new synonyms are introduced to provide ontologies as suitable targets for alignment.

4 The Conceptual Ontology Intersection

The conceptual ontology intersection, presented in this chapter, builds on the contents of the ontologies. The contents are extracted by syntactic mapping including syntactic matching, to build a intersection of all syntactic related parts, and synonym alignment to enrich the intersection with synonyms. This intersection becomes a conceptual ontology bridge between ontologies and contains related parts found in the involved ontologies.

A process for conceptual ontology intersection is used for mapping and alignment, see Figure 1. The process starts with fetching the ontologies from the web. Then, the syntactic mapping, with syntactic matching takes place. The result is a meta-model with syntactic parts found in the ontologies. Next step is a synonym alignment, which takes either ontologies or meta-model as input and expands the meta-model with more parts, this time with synonyms. Since mapping and alignment can introduce violations in the meta-model, violations are checked.

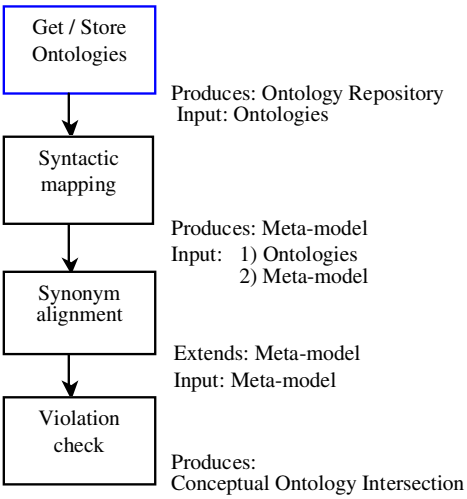

Fig. 1. The process

For each time a new ontology is introduced the meta-model will expand and the violation check needs to go through the whole meta-model.

4.1 The Syntactic Mapping

Mapping the ontologies syntactically is to match and map each concept in one ontology to each concept in the other ontology. The matching is carried out by picking up the different tags and words in the ontologies, using agents. The tags are the standard xml-tags that are used for, e.g., URL, texts, pictures, and signatures [26]; whereas the

words are concepts in a domain. The mapping is to reason with the contents of the ontologies to find relations and correspondances between the concepts.

To match and map parts in the ontologies, a rule base is used. The rule base includes parts that can be found in the ontologies. Some commonly used ontology parts are basic metadata, document metadata, RDF metadata, and description term:

- The basic metadata includes *url:, desc:*, *def:*, *ref:*, *pop:*, and *ns:*,
- The document metadata consists of *hasEncoding:*, *hasLenght:*, *hasMd5sum:*, *hasFiletype:*, *hasDateLastmodified:*, and *hasDataCache*.
- The RDF metadata has *hasGrammar:*, *hasCntTriple:*, *hasOntoRatio:*, *hasCntSwtDef:* and *hasCntInstance*.
- For the description term, there are other terms associated to it. For example, *log:forSome*, *title*, *s:label*, and *s:subClassOf*. To the term "Class", there are terms like *s:comment*, *s:label*, *ont:UniqueProperty*, *s:domain*, and *s:range*.

The terms are represented as facts and/or rules, in the rule base, to be able to compare the contents of the ontologies. In the rule base, there are single valued facts and compound terms, which are rules with different levels of complexity. Some terms can be matched directly to the facts whereas other terms must use rules to reach a conclusion, i.e., use reasoning to extract the contents of the ontology. By implementing all these terms in a rule base, the rule base will contain necessary knowledge and guidance to perform more efficiently.

Direct matching or *syntactic matching* is to match every line in ontology to the other ontology, which is to check that the tags and words are the same in both ontologies. Also words with different tags are matched between the ontologies. The process of direct matching is as follows: The process starts with one ontology and extracts the first tag and match it with the other ontology. Then, the process use the word, attached to the tag, and match it to the other ontology tags' word. For each match, the tags together with the words are stored in a meta-model as pair <Tag, Word>.

Then, the matching process proceeds with only the words, ignoring the tags, and matches each word in the ontologies. If the word appears in both ontologies, irrespectively of the tags, the tags and words are stored in the meta-model as <Tag1, Word>, <Tag2, Word>. The result from direct matching is a meta-model with same tags and words, or different tags and words, found in the ontologies. However, to know whether the words have correct connections to each other, reasoning with the contents is carried out. Reasoning with the ontologies capture context, to some extent, and might prevent that the mapping leads to wrong result.

For *syntactic mapping* with reasoning, the result, from direct matching in the meta-model, is used to reason with other ontologies, which is performed by using rules from the rule base and the ontology repository. These rules are parsing rules, which for each pair, stored in the meta-model, capture the surrounding environments in the ontologies by comparing the surrounding terms connected to the pairs, i.e., tags and words. The syntactic mapping use words in one ontology and rule base and, then, map with an other ontology to extracts concepts and relations with connections between classes, objects, and properties.

The rules for basic metadata, document metadata and RDF metadata, with axiom and entities, and description term are used to handle the syntactic contents of the ontology. The same types of entities and their contents are syntactically compared. For the types, the labels of the class are compared between the ontologies and, for the contents, the labels of the object property are compared. For example, a class have a subclass, and they share a property relation.

The syntactic mapping compares the concepts and relations in one ontology with the contents of another ontology by following strategies:

1) the letter cases are ignored. No matter the letters used in a word, it is consider to be identical in both ontologies, regardless the combination of the upper cases and lower cases.
2) only the letters are compared, and special characters are excluded.
3) grammatical forms are ignored, i.e., singular and plural of nouns are equal and all the forms of verbs are ignored.
4) nothing is excluded. As long as the definitions are not in conflict with each other, they can coexist and enrich the knowledge but only the same concepts and relations are checked and stored.
5) non-matched parts in the ontologies are left without consideration. These parts can still be reached by looking in the original ontologies.

The result from mapping is all the entities and the properties found in the ontologies that are comparable. The entities are stored as concepts in the meta-model and properties are stored as relations. Moreover, to keep the concepts and related relations connected to each other, these are connected by signatures, which are stored as rules, in the meta-model. Also, the relations are stored as rules in the meta-model.

To illustrate the syntactic matching and mapping with reasoning, an example is given below, see Figure 2. In the figure, classes, subclasses, objects and properties are presented. In Ontology 1, left hand in Figure 1, Document is a subclass of Root. The subclass connects to other parts by properties, which, in this case, are: “date-creation”, “name”, “has-author”, and “has-topic”. These properties connect the Document to several objects, which are Literal, Author, and Topic. Other connections, i.e., Journal, Publication, Book, Presentation, and Report are subclasses to Document. Ontology 2, on the right hand of the figure, shows that Document is a subclass of Source and contains the subclasses Website, Publication and Ontology. To the Document class, also the property “has-author” is connected, which is another way of connecting Document and Source.

The meta-model is expanded for every new axiom (fact) and entity (concept) and property (relation) the reasoner finds in the ontology. Also, the set of signatures, that is considered to be as being equivalent by the syntactic matching and syntactic mapping, are stored in the meta-model. A partial result of reasoning with Ontology 1 and Ontology 2, for the example above, is O1={(root, class), (document, subclass), (publication, subclass), (has_author, objectproperty)}, O2={(source, class),

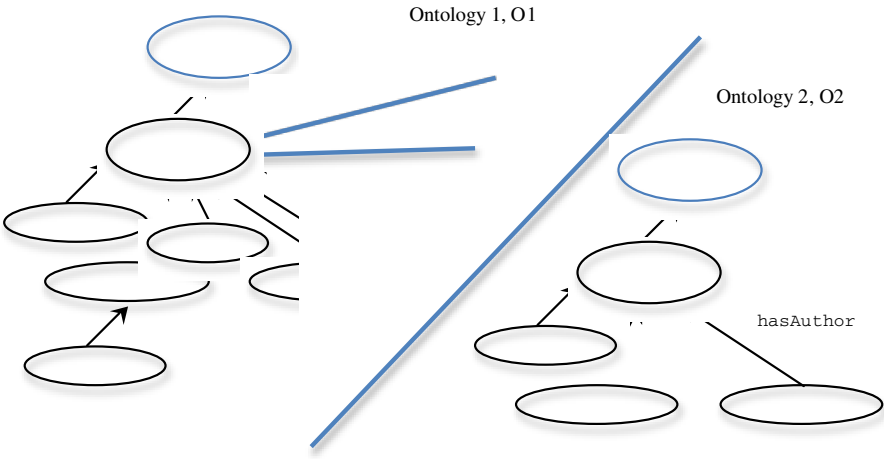

Fig. 2. The example ontologies

(document, subclass), (hasAuthor, objectproperty)}. These sets are stored as rules in the meta-model, as well as, the relations between them. The mapping produces following rules:

Class(X):- 'Root'.
Class(X):- 'Source'.
Class(X):- 'Publication'.
Class(X):- 'Paper'.
Subclass(X):- 'Document'.
Subclass(X):- 'Publication'.
Object(X):- 'Author'.
ObjectProperty(X):- has-author.
ObjectProperty(X):- hasAuthor.
SubClassOf(X, Y):- Subclass(X), Class(Y).
ObjectPropertyRange:- Subclass(X), ObjectProperty(X), Object(X).
ObjectPropertyRange:- Subclass(X), ObjectProperty(X), Class(X).

These rules expand the meta-model. One can argue that the rule base should be expanded for these new developed rules for matching and mapping. The argument for separating the rules for matching/mapping from rules for tags is that the rule base will be used for new comparing activities and the meta-model will be built from the beginning for each new comparison.

If a third ontology is compared with the other ontologies, the meta-model, and rule-base are used. However, since the third ontology might have more connections that can found in either of the already matched and mapped ontologies, also the ontology repository needs to be involved.

From this point there are three approaches provided:

1) expand the meta-model with rules, which are new connections provided by matching and mapping both ontologies but one at the time, Expanding the meta-model with new connections requires matching and mapping with first one ontology and then the other ontology giving the result {O1 Λ O2, O1 Λ O3, O2 Λ O3}
2) expand the meta-model with rules, which are connections found when matching and mapping with the meta-model. Expanding with connections is {O1 ⋂ O2, O1 ⋂ O3, O2 ⋂ O3}
3) limiting the contents in the meta-model. Limiting the meta-model means matching and mapping with the results from earlier matching and mapping to reduce the rules so the rules that only covers all three ontologies are left in the meta-model becoming the intersection of the ontologies, i.e., {O1 ⋂ O2 ⋂ O3}.

By using any of these three approaches, this meta-model becomes a bridge between the ontologies, containing parts found in all ontologies. However, all three approaches have advantages and disadvantages, and these depend on the situation in which the matching and mapping is performed. The first approach will include everything in the ontologies, i.e., not exclude anything, while the other two approaches are intersections of the ontologies with different sets as shown above. In the research in this chapter, the third approach is employed.

4.2 The Ontology Repository

The ontology repository is designed to store the ontologies. The ontologies are stored in their original form and the meta-model is used to sort the repository and retrieve the ontologies. Since the meta-model describes the ontologies, it forms context for the ontologies. The context also contains information about the ontology's annotation, i.e., the purpose, specific date, particular interest and the owner. The purpose of an ontology is the intention or goal given by the owner or developer of the ontology; the interest is the users, groups or organizations for which the ontology is developed. The owner of the ontology is the person or the organization that owns the copyright of the ontology. The date is the date of developing the ontology.

Consider the example ontologies in Figure 2, the Ontology 1 is about researchers and Ontology 2 is an ontology for inferring on the web. To store those ontologies in the repository, following parts can be used see Table 1.

Table 1. Example of the annotation

	Purpose	***Owner***	***Interest***	***Date***
Ontology1	Application	Karlsruhe		071010
Ontology2	Online course	Stanford	Graduate credits	100920

Using annotation can help to enrich the meaning of the integration. This can be applied to the meta-model and give more information about the integration.

4.3 The Synonym Mapping Process - Synonym Alignment

Synonym alignment is the process of finding synonyms for the words in the ontologies and checking whether or nor they match with the contents in the other ontologies. There are two slightly different processes for the alignment, which depend on the number of ontologies to align: 1) two ontologies are aligned and 2) three and more ontologies are aligned. The first process fetch synonyms from the contents of the ontologies and the second from the meta-model and the ontology to be aligned.

If two ontologies are aligned, the meta-model contains only the concepts that have syntactic match but there might be concepts that are similar in the original ontologies. Therefore, the whole contents of the two ontologies must be used for synonym alignment.

If three or more ontologies are aligned, it is enough to use the meta-model, instead of the two original ontologies because, this time, the meta-model contains both the syntactic match and synonym matching. This meta-model is used when the third or more ontology is being aligned.

When ontologies are aligned, the parts: classes, objects, and properties, in the ontologies, are used to find synonyms. For each of these parts, synonyms are searched and fetched from the online WordNet [14] giving back a set of words in the form {< , >,…,< , >}. The set of synonyms is used to check if the classes, objects and properties belong to the same set of synonyms and, if so, they are aligned. The alignment between the synonyms is creating links between two original ontologies. They synonyms are equivalent if a synonym of a concept or relation in one ontology map to 1) a concept or relation in the other ontology, or 2) a synonym of a concept or relation in the other ontology. If the synonyms are equivalent, they are stored in the meta-model, as the <concept, synonyms> or <relation, synonyms>. If synonyms are not equivalent, the synonyms are discharged.

When three or more ontologies are aligned, the process of finding and aligning the synonyms is identical but instead of using the original ontologies, that are already aligned, the meta-model with syntactic mapping and semantic alignment is used together with the third or more ontologies. The concepts and the relations in meta-model as well as the contents, i.e., classes, objects and properties, of the third or more ontologies are used to find the synonyms. Only the contents of the third or more ontologies are sent to WordNet [14] to find synonyms. The set of synonyms are aligned with the contents of the meta-model and if synonyms are equivalent, these synonyms are stored in the meta-model.

Commonly, words used in the ontologies are nouns and verbs. However, WordNet [14] generates both nouns and verbs as synonyms. That is, if a noun (or a verb) is sent to WordNet, it generats both nouns and verbs synonyms. This must be limited so noun give nouns as synonyms and verb gives verbs as synonyms. For synonyms to ontologies this means, the class and objects commonly are nouns, and hence, the synonyms, fetched from WordNet, must be a noun; the properties can be verbs and, therefore, the synonyms must be verbs. Limiting the synonyms to word classes can make the alignemnt more correct since, other word classes, might introduced missinterpretation. For example, the noun “document” and the verb “document” give different sets of synonyms.

To illustrate the synonym alignment, an example of synonyms for the Ontology 1 and Ontology 2 mentioned above, is given. O1 includes the word "Document" and when looking up synonyms in WordNet [14], it provides the synonyms: "written document", "papers", "text file" together with explanations, such as "writing that provides information (especially information of a official nature)", a "written account of ownership or obligation" and "(computer science) a computer file that contains text (and possibly formatting instructions) using seven-bit ASCII characters". It also provides synonyms for the verb "document" but only with the explanations: "record in detail", "support or supply with references". Since Document is a noun, the nouns are in focus when comparing (document, class) and (paper, class). Since WordNet [14] gives "paper" as a synonym to "Document", it is found to be an alignment between those words – even though the ontologies use two different words. Both these words are stored in the meta-model.

4.4 Violation Handling for Syntactic Mapping and Synonym Alignment

Although syntactic mapping and synonym alignment are made, the contents of the ontologies may not harmonize. To find contradictions or different implications among the parts in the ontologies, violation check for the syntactic mapping and synonym alignment is carried out. The concepts together with the relations are checked for violations to find how well they really match, map but also align.

Violation Check for Syntactic Mapping

For syntactic mapping, including syntactic matching, the combination of classes, objects and properties can mismatch between the meta-model and ontologies. The violation check for syntactic mapping controls if the concepts and relations really share corresponding concepts and relations between the mapped ontologies. The violation check also controls if relations are connecting the same concepts in the ontologies. The checking is both carried out by syntactic mapping, which controls if the concepts and relations are the same, as mentioned above, by following the five different strategies, such as, accepting different ways of writing the words but also the implications, i.e., O1A -> O1B $\wedge$ O2A -> O2B.

To illustrate contradictions, an example of the ontologies, O1 and O2 presented above, is given. The result of a syntactic mapping is that the relations "has-author" and "hasAuthor" are equal and the concepts Document, Author and Source are stored. In the Ontology 1, the subclass Document is connected to the class Author and in the Ontology 2, the subclass Document is connected to class Source. Hence, there is a contradiction between the classes Author and Source.

To handle the contradiction, rules from the rule base are used to solve the problem. The rules have three different outcomes and the possible results are checking the kind of violation and the outcomes are: 1) accept violation, 2) provide notification, or 3) confirm violation. Hence, the kind of violation found between the ontologies steers the outcome.

The first type of violation, accept violation, implies that relations are not violating anything between the ontologies, even though the concepts are the same same type of entity. This kind of violation implies that the relations in the ontologies are found and they are equal but the names of the classes or objects differ. The result is stored as an equivalent violation case in the conceptual intersection ontology, i.e., in the given example the subclass has the classes Source and Root, together with the related concepts, and relations. The format for the case is EquivalentViolation(Document, < Root, Source >) and the reason is that the synonym violation needs to check this equivalent violation case.

The second kind of violation, provide notification, also implies that relations are not violating anything between the ontologies, but, in this case, the concepts are of different same types of entities. This violation implies that the relations in the ontologies are equal but the types of entities differ. The result of this violation is stored as a notification in the conceptual intersection ontology with the concepts, relations, and the different types. In the example mentioned above, one entity is a class and the other entity is a subclass. The format for notifications is the NonEquivalentViolation(Document, <Publication, class ⋀ Publication, subclass>) where Publication is a subclass in O1 and a class in O2.

The third kind of violation, confirm violation, implies that the concepts and the classes violate too much to be stored in the conceptual intersection ontology. An example is that a concept and a relation have the same name. This case should not occur since the matching and mapping should avoid finding equivalence between concepts and relations from the tags and words. Nonetheless, this confirm violation case also comes into effect if none of the above cases can capture a violation. The confirmation of the violation is stored in the conceptual intersection ontology as ConfirmedViolation(Document, <HasAuthor, property ⋀ Author, class>)

The violation check process work through all of the concepts and relations in the meta-model and carries on until the whole meta-model is checked. The result with concepts, and relations that are acceptable and non-violating any of the syntactic mapping are stored in the conceptual ontology intersection.

Violation Check for Synonym Alignment

Also the synonyms are checked for violation problems. This is to check that the synonyms are comparable to concepts, and relations in the ontologies, and/or the meta-model. The violation check for synonyms is more difficult since it is difficult to detect synonym violations.

A violation check for synonyms is when syntactic mapping has found a violation, i.e., accept violation, where the relations in the ontologies are found to be equal but the names of the classes, or objects, differ. The synonym violation check uses the EquivalentViolation in the meta-model to look up the concepts in the WordNet and receive a set of the synonyms for these concepts. Then, the concepts are tested towards the set of synonyms to find equivalences. If succeeded, the result is stored in the conceptual ontology intersection as EquivalentCheck(concept, <concept, synonyms>) with the concepts and relations. In the case of the example, above, it can be

EquivalentViolation(Document, <Root, (beginning, origin, root, rootage, source) >, < Source, (beginning, origin, root, rootage) >).

Another violation that is when the meanings of the words differ. Meaning check requires semantic examination of the words to investigate the relations between the words. The violations can arise when a term, which is a concept or relation, in one ontology, *does not* align the same term, in the other ontology, even though surrounding concepts and relations *do* align. This means that almost the whole set of synonyms for one ontology, O1, is the same set of synonyms for other ontology, O2 and the aligned set is moved to the conceptual ontology intersection. The synonyms that do not align needs to be denoted and stored in conceptual ontology intersection by applying a notification to the synonym in the form of NonAlignedViolation(concept, synonym).

This violations can also arise when a term, , i.e., concept or relation, in one ontology, *does* align the same term, in the other ontology, but the surrounding concepts and relations *do not* align. Then, almost the whole set of synonyms for one ontology may be totally different from the other ontology's set. The synonym mapping can find a concept, or relation, in the involved ontologies, because it distinguishes itself from the set of the aligned concepts and relations, by using the surrounding concepts and relations to provide information about the domain and context. Again, the synonyms, that do not align, needs to be denoted and stored in conceptual ontology intersection by applying a notification to the synonym in the same form of earlier NonAlignedViolation(concept, synonym) together with the concept and/or relation that does align.

As result of the violation check or syntactic mapping and synonym alignment, a conceptual ontology intersection is produced with parts that are captured in the violation check.

4.5 Context of the Ontologies

As a result from mapping and alignment, the conceptual ontology intersection becomes a context for all compared ontologies. One entity is used to build context and a domain for that entity. All the ontologies that are using a specific entity can be fetched and incorporated in the system and in the intersection and, therefore, extend and strengthen the context for the entity. To use the context from different perspective, a perspective property is introduced as a relation in the ontology intersection. The perspective property is used to combine the parts of the ontologies that concerns the property. The perspective property is illustrated in Figure 3, where the conceptual ontology intersection is in the middle and the ontologies in the system is connected to the intersection by the relation "hasPerspective".

The contents in the ontologies can, together, give knowledge about a context in a domain. This useful facililty can provide information about, for example, a service and can, when combined with several different ontologies or similar intersections, provide complex services to the users. This requires that one ontology provides a service that can be combined with the services provided by the other ontologies.

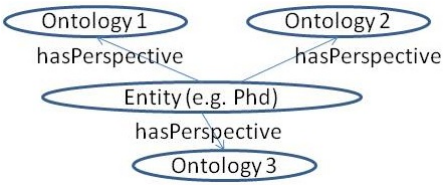

Fig. 3. Perspectives for the entities

For the context comparison, rules are used. The rules must prove that the entities of the ontologies match. The rules investigate the entities by checking that the contents of these entities belong to the same context. Thus, the rules check whether the contents of the entities are found in that domain or not, which is the context for the ontology. Conclusively, every ontology's entities are used to build the services.

Moreover, the rules can use other rules and facts to match the contents. If some information piece is missing, the system can turn to other sources, like the web or users to get additional information.

4.6 The Architecture

The architecture for the system distinguishes the ontology repository, rule base, meta-model and conceptual ontology intersection, see Figure 4. The ontologies, fecthed either from a file or an URL, are deposited into the ontology repository and the rules for matching, mapping, and alignment are stored in the rule base. These rules use, with the help of a rule engine, the ontologies to generate result for syntactic mapping. Syntactic mapping with matching applied the strategies 1) - 5) mentioned in 4.1 and the result is concepts and relations, which are stored in the meta-model.

The ontology repository is sorted accordingly to the contents of the ontolgies and with the help of the meta-model. The ontology repository stores signatures from the ontologies by adding the signatures as records in the table. These records are connections to the ontologies and are used for searching for the ontologies used in the system.

Synonym alignment provides function for synonym alignment, which uses the result in the meta-model to fetch synonyms in synonym lexicon and align the ontologies. The result from the synonym alignment is also stored in the meta-model.

The meta-model is checked for violations, syntactic and synonyms, and the concepts, relations and synonyms that passes the violation check are stored in the conceptual ontology intersection, as well as, the parts that are not contractions or wrong impliations. The conceptual ontology intersection constititues the context for the ontologies mapped and aligned in the system.

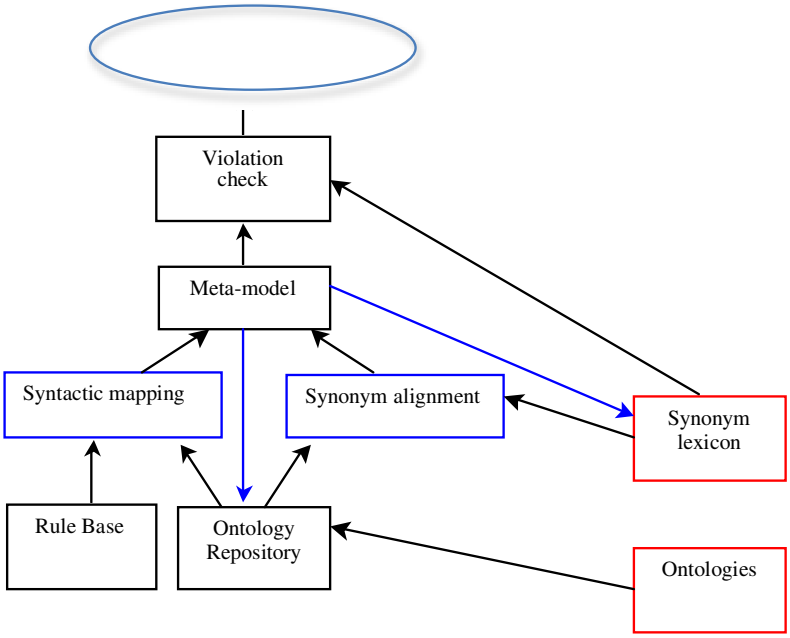

Fig. 4. Architecture

5 Related Work

A lot of semi-automatic and automatic methods have been developed for ontology ontology merging and mapping [27]. For example, FCA-Merge is a method for ontology merge [28] and IF-Map is a method for ontology mapping [29]. In the FCA-Merge method, natural language techniques are used to derive lattices of concepts. The lattices are explored, manually, by knowledge engineers who also build the merged ontology with the help of semi-automatic guidance of the FCA-Merge method [27]. Exploring the lattices and build merged ontologies can be a time-consuming and tedious job, if the ontologies are medium and larged sized.

IF-Map method is an automatic method for ontology mapping. IF-Map provides automated support in the alignment of ontologies by automatically generating mappings between a reference and various local ontologies. The method is a concept-to-concept and relation-to-relation mapping for local ontologies, which limits the result of the mapping.

Another automatic approach for mapping two ontologies is a process using weighted similarities [30]. The result is a set of statements that contains the semantic correspondences between similar elements in two ontologies with associated relations and a confidence of the mapping result. Confidences in results of mapping is an important feature and by applying rules for syntactic mapping, confidence can be upheld on a certified enough level [7].

Methods have been developed for ontology alignment. An example of ontology alignment is a general framework for alignment. This framework is data interlinking, which separate data interlinking and ontology matching activities to enhance data linking through ontology alignment by allowing linking specifications to reuse ontology alignments in a natural way [31]. Using specifications are interesting but should be extended together with the contents and relations found in the ontologies.

Another example of ontology alignment is an algorithm that uses a complete prover to decide subsumptions or equivalences between classes given initial equivalence of some classes and analysis of the relationships in the taxonomy [32, 33]. The complete prover, in its basic form, consists of ”testing each possible variables assignment. A backtrack is executed when an inconsistency is reached where certain rules can be used in order to reduce the search space ” [33].

Finding the equivalences and subsumptions is the main task for the research in this chapter but not from any initial equivalence between the ontologies, rather find all equivalences between the ontologies.

Another interesting method is semantic translations of the ontologies handling similar domains [34]. This method includes developing bridges for axioms in the ontologies to merge these ontologies. Ideas from this technique are used to build intersections of concepts and relations which become bridges between the ontologies. In the research, presented in this chapter, also synomyms are utilsed to expand the possibly to capture more bridges between the ontologies, that is when the ontologies use different vocabulary in the same domain.

6 Conclusions and Further Work

The main contribution of this chapter is conceptual ontology intersection which includes parts that are related in several different ontologies. Contents are extracted and compared by syntactic matching and syntactic mapping and synonym alignment. The syntactic mapping and synonym alignment are applied on ontologies fetched from the Web. These ontologies are stored in their original form in an ontology repository to be able to revisit and reuse them.

Contents are extracted by syntactic matching and mapping using a rule base, and the ontology repository. The rules are used for the matching and mapping process where the concepts and relations are extracted from the ontologies. During the mapping process, a meta-model of the ontologies is built. The meta-model contains the parts from the ontologies that have syntactically mapped and synonym matched. The meta-model is a conceptual ontology intersection, which is an ontology bridge containing syntactic similarities between the ontologies.

To enrich the connections between the ontologies, synonym alignment is used. The alignment is either aligning two ontologies or the result from syntactic matching and mapping ontologies and an ontology. To find the synonyms, the alignment uses a synonym lexicon and the result expands the meta-model with synonyms.

The result is a conceptual ontology intersection that contains parts from the ontologies. The ontology intersection provides contexts for the ontologies. Building contexts

for the ontologies can help to reuse the ontologies but also for further mapping and alignment.

The solution works for small and medium sized ontologies. For large sized ontologies, links to the ontologies' sites are stored instead. Also, the synonym lexicon is fetched from the Web. The lexicon is used for searching and finding synomyms for alignment. The lexicon itself will not be stored in the system but the synomyms, that is results from the alignment, are stored. The rule base stores rules for mapping and alignment in the system.

References

1. Duc, T.T., Haase, P., Motik, B., Grau, B.C., Horrocks, I.: Metalevel Information in Ontology-Based Applications. In: Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI 2008), Chicago, Illinois (2008)
2. W3C: OWL 2 Web Ontology Language, Structural Specification and Functional-Style Syntax, W3C Recommendation (October 2009)
3. Gruber, T.R.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing, Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University (1993)
4. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artificial Intelligence (2011)
5. Kalfoglou, Y., Schorlemmer, M.: Ontology Mapping: The State of the Art. Journal The Knowledge Engineering Review 18(1) (2003)
6. Bouquet, P., Ehrig, M., Euzenat, J., Franconi, E., Hitzler, P., Krötzsch, M., Serafinin, L., Stamou, G., Sure, Y., Tessaris, S.: D.2.2.1 Specification of a common framework for characterizing alignment, Knowledgeweb, realizing the semantic web (2005)
7. Wu, D., Håkansson, A.: Accepted in Proceedings of 8th International Conference on Web Information Systems and Technologies (WEBIST), April 18-21 (2012)
8. Gruber, T.: Ontology. In: Liu, L., Tamer Özsu, M. (eds.) Encyclopedia of Database Systems. Springer (2009), `http://tomgruber.org/writing/ontology-definition-2007.html`
9. Sowa, J.F.: Ontology (2011), `http://www.jfsowa.com/ontology/index.html`
10. Sowa, J.F.: Ontology, Metadata, and Semiotics. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS, vol. 1867, pp. 55–81. Springer, Heidelberg (2000)
11. McKeon, M.: Model-Theoretic Conceptions of Logical Consequence. Internect Encyclopedia of Philosopy. Michigan State University (2004), `http://www.iep.utm.edu/logcon-m/` (Originally published: June 16, 2004)
12. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artificial Intelligence (2011), doi:10.1016/j.artint. 2011.01.007
13. Gruber, T.R.: Ontolingua: A mechanism to support portable ontologies. Stanford University, Knowledge Systems Laboratory, Technical Report KSL-91-66 (March 1992)
14. Carlsson, C., Fullér, R.: Possibility for Decision. STUDFUZZ, vol. 270. Springer, Heidelberg (2011) ISBN 978-3-642-22641-0
15. Princeton University, WordNet, A lexical database for English, `http://wordnet.princeton.edu/wordnet/` (last update: June 21, 2011)

16. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programs: Combining Logic Programs with Description Logic. In: WWW 2003, May 20-24 (2003)
17. Jérôme, E., Pavel, S.: Ontology Matching. Springer, Heidelberg (2007) ISBN 3-540-49611-4
18. Ehrig, M., Sure, Y.: Ontology Mapping - An Integrated Approach. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 76–91. Springer, Heidelberg (2004)
19. Kotis, K., Vouros, G.A.: The HCONE Approach to Ontology Merging. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 137–151. Springer, Heidelberg (2004)
20. de Bruijn, J., Ehrig, M., Feier, C.: Ontology mediation, merging and aligning. In: Semantic Web Technologies Trends and Research in Ontologybased Systems, pp. 1–20 (2006)
21. Sowa, J.F.: Top-Level Ontological Categories'. International Journal of Human Computer Studies 43(5/6), 669–685 (1995)
22. Noy, N., Musen, M.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In: Proceedings of the National Conference on Artificial Intelligence, AAAI (2000)
23. Sowa, J.F.: Knowledge Representation: Logical, Philosophical and Computational Foundations. Brooks/Cole (2000)
24. Pinto, H.S., Gómez-Pérez, A., Martins, J.P.: Some Issues on Ontology Integration. In: Proc. of IJCAI 1999's Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends (1999)
25. Pinto, H.S., Gomez-Perez, A., Martins, J.P.: Some Issues on Ontology Integration. In: Proc. of IJCAI 1999's Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends (1999)
26. Sowa, J.F.: Principles of ontology, http://www-ksl.stanford.edu/onto-std/mailarchive/0136.html
27. Håkansson, A., Hartung, R., Moradian, E., Wu, D.: Comparing Ontologies Using Multi-agent System and Knowledge Base. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part IV. LNCS (LNAI), vol. 6279, pp. 124–134. Springer, Heidelberg (2010)
28. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review 18(1), 1–31 (2003)
29. Stumme, G., Maedche, A.: Ontology Merging for Federated Ontologies on the Semantic Web. In: Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII 2001), Viterbo, Italy (September 2001)
30. Kalfoglou, Y., Schorlemmer, M.: Information-Flow-Based Ontology Mapping. In: Meersman, R., Tari, Z. (eds.) CoopIS/DOA/ODBASE 2002. LNCS, vol. 2519, pp. 1132–1151. Springer, Heidelberg (2002)
31. Scharffe, F., Euzenat, J.: Linked Data Meets Ontology Matching - Enhancing Data Linking through Ontology Alignments. In: Filipe, J., Dietz, J.L.G. (eds.) KEOD 2011 - Proceedings of the International Conference on Knowledge Engineering and Ontology Development, Paris, France, October 26-29, pp. 279–284 (2011)
32. Dou, D., McDermott, D., Qi, P.: Ontology Translation on the Semantic Web. In: Spaccapietra, S., Bertino, E., Jajodia, S., King, R., McLeod, D., Orlowska, M.E., Strous, L. (eds.) Journal on Data Semantics II. LNCS, vol. 3360, pp. 35–57. Springer, Heidelberg (2005)

33. Euzenat, J., Valtchev, P.: Similarity-based ontology alignment in OWL-Lite. In: de Mantaras, R.L., Saitta, L. (eds.) Proc. 16th of European Conference on Artificial Intelligence (ECAI), Valencia, ES, pp. 333–337 (2004)
34. Giunchiglia, F., Shvaiko, P.: Semantic matching. In: Proc. IJCAI 2003 Workshop on Ontologies and Distributed Systems, Acapulco, MX, pp. 139–146 (2003)
35. Mao, M., Peng, Y., Spring, M.: An adaptive ontology mapping approach with neural network based constraint satisfaction. Web Semantics: Science, Services and Agents on the World Wide Web 8, 14–25 (2009)