An Architecture for Semi-Automatic Ontology Merging System

Siham Amrouch[1], Sihem Mostefai[2]
[1]Departement of Computer Science
Med Cherif Messadia University
Souk Ahras, Algeria
[2]MISC Laboratory, Departement of Computer Science
Université Mentoury
Constantine, Algeria
sihamamrouch@yahoo.fr, xmostefai@yahoo.com

ABSTRACT: *.In recent years, ontologies have played a key technology role for information sharing and agents interoperability in different information systems. But, it seems that there is always more than one conceptualization for the same domain or even for similar domains. In other words, it emerges every day, new different ontology to model the same domain. Therefore, to answer queries on the modeled domain, bridge the gaps between different ontologies is a key challenge for the researchers in the AI community by using ontology merging. In this paper, we propose an architecture for a semi-automatic ontology merging process. The semi-automatic character is handled by the human intervention where the knowledge engineer intervenes to validate the results provided by the similarity computation module. This later is based on a lexicosemantic algorithm that combines lexical and semantic measures to identify the similar concepts that have to be merged into a single one in the merged ontology, after human validation. The judged different concepts are directly copied to the merged ontology.*

Keywords: Semi-Automatic Ontology Merging, Semantic Integration, WordNet, Owl, Lexico- Semantic Similarity

Received: 2 March 2012, Revised 19 April 2012, Accepted 28 April 2012

1. Introduction

In the last decade, the aim of researchers in semantic web community was and still to bring the actual web to its full potential by considering ontologies as the best means to annotate the data on the web.[7]. In other words, ontologies are emerged as the best models for information storage and representation with preserving the semantics embedded in their application domains, in several areas such as semantic web and web services, industrial and e-technologies in general. According to T. Gruber[1], ontology is an explicit specification of a conceptualization. This structure can be cognitively semantic (ontology intended to be exploited by the user) or computationally semantic (ontology intended to be exploited by the machine), [2]. In general, an ontology is composed of a set of concepts described by a set of properties and related by a set of semantic relationships, to construct an hierarchy of classes, where each sub-class described a concept that is more specific then the concept described by the super-class. With several designers that appear every day, there is always more than one ontology that describes the same domain. In other words, it emerges every day, different ontologies developed by different developers with different viewpoints and in different goals of use. Hence, to create a common repository of knowledge base and to remove overlaps between existing ontologies, we go for ontology merging. This process is seen as the effort of building a single ontology from a set of source

ontologies that cover a wider scope. Several tools and algorithms for ontology merging, based on different criteria, exist in the literature such as: Prompt, Chimaera, ONION, FCA-Merge, etc. According to [4], these algorithms are generally based on: Names and descriptions of concepts in natural language, class hierarchy (the relations: subclass and superclass), setting properties (domain, co-domain and restrictions), classes' instances and classes' descriptions (Description Logic-based tools). The contribution outlined in this paper describes an architecture for a semi-automatic ontology merging system. In this later, and in order to identify similar concepts, the human intervention is necessary to validate the results obtained by combining the results of computation of lexical and semantic similarity measures. This paper is structured as follows: Section 2 briefly describes the ontology merging process. Sections 3 and 4 outline successively the semantic enrichment and mapping discoveryprocesses. Sections 5 surveys the literature of related works. Section 6 presents the general proposed architecture. In section 7 we compare our proposed process with the well known existing algorithms and we conclude by stating some important remarks and possible prospects in Section 8.

2. Ontology Merging Process

According to [3], ontology merging is seen as a complex process composed of three sub-processes: Firstly, ontology mapping and alignment. Then the normalization of source ontologies [5]. This later aims to reconcile the different choices of conceptual models and representation languages of the ontologies to be merged. Once these ontologies are sufficiently homogeneous, the final sub-process is to build the union of the controlled ontologies that can be constantly complex because of the potential complexity of interactions between the axioms of the input ontologies.

In this paper, ontology merging is seen as the process that creates a new and unique ontology that represents the union of two source ontologies and gathers all the knowledge contained in the two ontologies. In other words, all the similarities and dissimilarities presented by the two source ontologies must be reflected by the ontology resulting from this process. This later has three main stages: After the importation of the source ontologies (assumed sufficiently homogenous) the mapping discovery stage aims to identify similar concepts that will be merged into a single one in the final stage of merge. In this step, dissimilar concepts are directly copied into the resulting ontology. Figure 1 depicts the three main stages of ontology merging process:

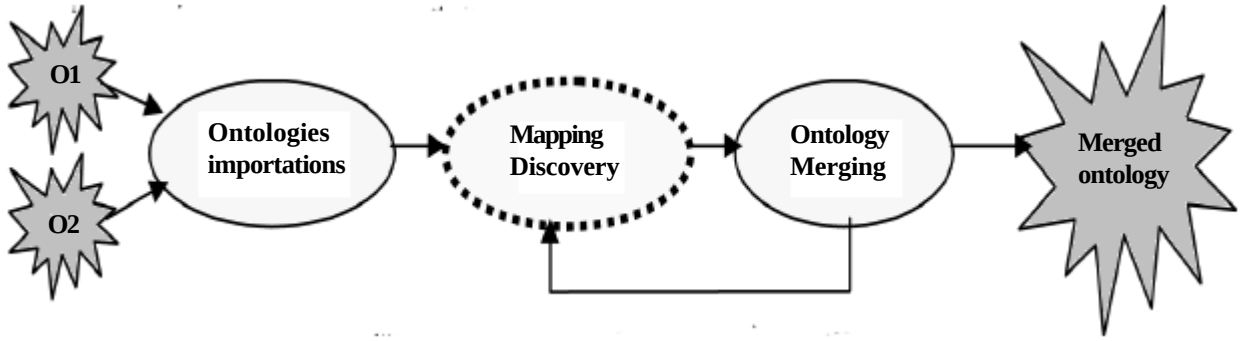

Figure 1. Ontology merging process

3. Semantic Enrichment

This is a substantial research field that serves the semantic web by facilitating interoperability between different applications and/or knowledge sources such as ontologies. In this paper, we will opt for the semantic enrichment from an external resource, wordNet, to avoid the limitations of the lexical aspect in the ontology merging process after their possible extensions according to their application domain. So, it is at this stage where acts the semantic aspect to support the ontology merging process. Herein, the more the extension of the source ontologies is close to the same shared ontology, the easier will be the similarity identification process. In addition, reasoning and inference processes handled by the ontology representation languages contribute in specifying the constraints of similar concepts merging. However, semantic integration process may be altered by several types of mismatches [17]. The first one is the language level mismatches or syntactic mismatches caused by the different ontology representation languages. In addition, even with ontologies represented in the same representation language, we can have ontology level mismatches or semantic mismatches, such as using the same term to describe different concepts (homonyms), use different terms to describe the same concept (synonyms), use different levels of granularity, etc.

4. Mapping Discovering

This is the most important step in the ontology merging process. It identifies similar concepts that have to be merged into a single one in the last step of the ontology merging process. The resulting single concept includes all characteristics of the input concepts. In [4], authors identified two main approaches to identify the mappings between similar concepts. The first approach is based on the idea that the ontologies are designed and constructed to support future semantic integration. In other words, an ontology is built to be shared by different systems. After that, knowledge engineers perform several extensions of this ontology with concepts and properties specific to their applications. The more these extensions are consistent with the definitions in the shared ontology, the easier will be the similar concepts identification.

The second approach is based on heuristics or machine learning techniques that use various ontology features, such as taxonomies, concept definitions, class instances, etc, to discover similar concepts. In this work we will opt for the second approach. We will base our method on linguistic analysis of concepts' names to compute the lexical and semantic similarities between them. These results will be accepted or rejected by a knowledge engineer before outlining the final judged similar concepts.

5. Related Works

Several tools for ontology Merging (and even ontology Mapping or Alignment) exist in the literature. Most of these tools are semi-automatic and the design of fully automatic tools is usually a delicate issue. In this section, we outline the well known and recent ones:

5.1 FCA-Merge [12]

It's a method for semi-automatic ontology merging. Its process is summarized as follows: First, from a set of input documents, popular ontologies (ontologies equipped by their instances) are extracted. Once the instances are extracted and the concept lattice is constructed, FCAtechniques are used to generate the formal context of each ontology. Using lexical analysis, FCA-techniques retreive specific information that combines a word or an expression to a concept if it has a similar concept in the other ontology. Then the two formal contexts are merged to generate the pruned concept lattice. Herein, the knowledge engineer may eventually intervene to resolve conflicts and eliminate duplications using his background about the domain. It should be mentioned that the major drawback of FCA-Merge is that it is based on instances to identify similar concepts, however, in most applications, there are no objects that are simultaneously instances in both source ontologies.

5.2 PROMPT [13]

This is an interactive ontology merging tool, it proposes a list of all possible merging actions (to-do list). After that, the knowledge engineer selects the appropriate proposals that go with his needs. Then, PROMPT automatically merges the selected pairs of concepts, provides the conflicts generated after merging (conflict-list) and proposes their appropriate solutions. Finally, the knowledge engineer selects the most suitable solutions.

5.3 Chimaera [14]

An interactive ontology merging tool, where the knowledge engineer is charged to make decisions that will affect the merging process. Chimaera analyzes the source Ontologies and if it finds linguistic matches the Merging is performed automatically, otherwise, the user is prompted for further action. Like PROMPT, Chimaera is an ontology editor plugin, namely Ontolingua, but they differ in the suggestions they make to their users with regard to the merging steps.

5.4 Glue [15]

To find mappings between two source ontologies O' and O", Glue uses machine learning techniques. So, for each concept of ontology O', Glue finds its most similar concept in ontology O" based on different practical similarity measures and several machine learning strategies. The authors also used a technique called "*relaxation labeling*" to map the two hierarchies of the two ontologies. This technique assigns a label to each node of a graph and uses a set of domain independent constraints, such as, two nodes of concepts c' and c" match if the nodes of their neighbourhood1 v(c') and v(c") also match, and a set of domain dependent constraints, such as, if X is an ascendent of Y and Y matches "direction" then X does not match "*sub-direction*".

5.5 ONION [16]

According to the authors, ontology Merging is inefficient because it is costly and not scalable. So, ONtology compositION system provides an articulation generator for resolving mismatches between different ontologies. The rules in the articulation generator express the relationship between two (or more) concepts belonging to the ontologies. Manual establishment of these rules is a very expensive and laborious task. And full automation is not feasible due to the inadequacy of natural language processing technology. The authors also elaborate on a generic relation for heuristic matches: Match gives a coarse relatedness measure and it is upon to the human expert to then refine it to something more semantic, if such refinement is required by the

application. In their system, and after validating the suggested matches by a human domain expert, a learning component is included in the system which uses the user's feedback to generate better articulation in the future when articulating similar ontologies.

6. General Proposed System

The aim of our work is to propose an architecture for a semiautomatic ontology merging system where the human intervention must not be avoided to ensure good performances. First, we import the two source ontologies that cover the same application domain (or related domains). Next, we identify their similar concepts. Here we will use an Information Retrieval (IR) technique, where each concept of the first ontology is compared with all concepts of the second one. The similarity identification module combines the results of two similarity measure techniques used in string comparison (concepts), one of them is lexical and the other one semantic. These results are accepted or rejected by a knowledge engineer using his background on the domain and oriented by his own needs. After that, the concepts accepted as similar are merged into one concept whereas the concepts judged dissimilar are directly copied to the resulting ontology. This later will be larger and more complete and will cover a wider application domain.

6.1 Lexical similarity

This technique is based on the computation of a distance between two strings describing the names of two concepts. Several measures of similarity or distances exist in the literature such as Levenstein distance [8], Hamming distance [11], Jaro distance [9], Jaro-winker distance [10], etc. All of these measures are based on the same assumption described by [6] which states that two strings are similar if they share enough important elements. We have chosen to use the Jaro distance as a similarity measure because it yields a value which is consistent with the value given by the semantic similarity measure that we have proposed (a value between 0 and 1) and therefore their combination is easier. The lexical similarity between the two concepts c1 and c2 is given by:

$$SIMlex(c1, c2) = Dj(s1, s2) \quad (1)$$

where Dj (s1,s2) is the Jaro distance between the two strings s1 and s2 labelling the two concepts c1 and c2 and which is defined by the equation :

$$Dj(s1, s2) = \left(\frac{1}{3} \quad \frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m - t}{m} \right) \quad (2)$$

Where: m: The number of matched characters. t = N / 2: the number of transpositions.

N: The number of pairs of matched characters that are not in the same order in their respective chains. Two identical characters of s1 and s2 (describing the concepts c1 and c2 respectively) are considered matched if their distance (i.e. the difference between their positions in their respective chains) does not exceed a certain value given by:

$$val = \left[\frac{max(|s1|, |s2|)}{2} \right] - 1 \quad (3)$$

The two concepts c1 and c2 are considered lexically similar if the distance between them exceeds a critical threshold to be determined empirically.

Example: Computation of lexical similarity between '*auto and automobile*' and between '*auto and car*':

SIMlex (auto, automobile) = Dj (auto, automobile)= ?

	a	u	t	o	m	o	b	i	l	e
a	1	0	0	0	0	0	0	0	0	0
u	0	1	0	0	0	0	0	0	0	0
t	0	0	1	0	0	0	0	0	0	0
o	0	0	0	1	0	1	0	0	0	0

m = 5 (number of 1 in the table), |s1| = 10, |s2| = 4, N = 1, t = 1/2,

$$SIMlex(auto, automobile) = \frac{1}{3}\left[\frac{5}{10} + \frac{5}{4} + \frac{5-0.5}{5}\right] = 0.883$$

Assuming that the threshold = 0.5, SIMlex = 0.883 ≥ 0.5 then uto and automobile are lexically similar.

Now, let's compare "*car*" and "*auto*", SIMlex (car, auto) = Dj(car, auto)= ?

	a	u	t	o
c	0	0	0	0
a	1	0	0	0
r	0	0	0	0

m =1, |s1| = 4, |s2| = 3, N =1, t =1/2,

$$SIMlex(auto, car) = \frac{1}{3}\left[\frac{1}{4} + \frac{1}{3} + \frac{1-0.5}{1}\right] = 0.36$$

SIMlex(auto, car) = 0.36 <0.5, then auto and car are lexically dissimilar.

SIMlex(car, plane)=0.34 <0.5, then plane and car are lexically dissimilar.

5.2 Semantic similarity

When the concepts are semantically similar but their names are different (synonyms) the null lexical similarity does not reflect the reality. To solve this problem, the integration of semantic similarity measure is crucial. To do this, we have begun with a semantic enrichment of the two source ontologies from wordNet[2]. It involves building a synonymy vector containing the synset elements for each concept.

We recall that WordNet is a computerized english dictionary where the basic unit is the concept. It uses two different means to define the meaning of a word, the synsets and the lexical relations. A word is then defined by a set of synonyms (synset) and a definition.

Example: Board: synset = {board, blank} Definition: A piece of wood.

For the computation of semantic similarity, we have used an information retrieval technique, which involves comparing each concept in the first ontology with all concepts of the second one to find out the most similar concept. We defined the semantic similarity between two concepts C1 and C2 as follows:

$$SIMsem(c1,c2) = 2 * \frac{Card(synset(c1) \cap synset(c1))}{Card(synset(c1)) + Card(synset(c2))} \quad (5)$$

$SIMsem(c1, c2) \in [0,1]$.

The two concepts c1 and c2 are judged similar if $SIMsem(c1, c2)$ is greater than a critical threshold which will be determined empirically. If the two concepts are exactly similar

$SIMsem(c1, c2) = 1$, in the opposite case $SIMsem(c1, c2) = 0$.

Example : Computation of lexical similarity between 'auto and car' and between 'car and plane':

Synset (*auto*)={*car, auto, automobile, machine, motocar*},

synset (*car*)={ *car, auto, automobile, machine, motocar* },

synset (*plane*)={*airplane, aeroplane, plane*}

$$\text{SIMsem (auto, car)} = 2*\frac{5}{10} = 1 \quad \text{and} \quad \text{SIMsem (auto, car)} = 2 * \frac{0}{8} = 0$$

[2] http://www.wordNet.princeton.edu/wordNet

Then auto and car are semantically similar but plane and car are semantically dissimilar.

Once the two similarity measures are computed, we compute the *lexico-semantic* similarity that combines the two results through the formula:

$$SIMlexsem\,(c1, c2) = \frac{SIMlex + 2 * SIMsem}{3} \qquad (6)$$

The two concepts are considered similar if *SIMlexSem* (*c*1, *c*2) reaches a critical threshold which will be determined empirically. Example :

$$\text{SIMlexsem (auto, car)} = \frac{0.36 + 2 * 1}{3} = 0.75 < 0.5$$

So, if the knowledge engineer accept or validate this similarity, the two concepts auto and car are similar and then will be merged into a single concept autocar.

$$\text{SIMlexsem (plane, car)} = \frac{0.34 + 2 * 1}{3} = 0.113 < 0.5$$

So, the two concepts plane and car are dissimilar and then will directly (without passing by the knowledge engineer) be separately copied in the resulting ontology.

4.2.1 How the merged ontology is constructed?

First, the merged ontology is initialized by the first source ontology. (All the concepts with all their properties of the first ontology are copied in the initial merged ontology). Then, each concept of the second source ontology is compared with all the concepts of the first one. If two concepts are judged as similar, we compare their properties. The ones of the second concept that does not exist (or have not similar ones) in the first concept (which has been copied in the initial resulting ontology) are added to the properties of this first concept. Hence the two similar concepts are merged into a single one without any omission of information. Else, if the two concepts are judged dissimilar, the most related concept (class) from the first ontology to the current concept of the second ontology is identified. It corresponds to the concept with the highest similarity measure between their properties. Finally, the one of these two related concepts, with the highest number of properties is the most specific, and hence will be copied (with all its properties) as the sub-concept (sub-class) of the other. This process is repeated for each concept of the second ontology. Hence, the whole merged ontology construction is accomplished.

The whole proposed architecture of the semi-automated ontology merging system is depicted by figure 2.

Properties	CHIMAERA	ONION	PROMPT	FCA-MERGE	GLUE	PROPOSED
1 Automation	Semi-automatic	Semi-automatic	Semi-automatic	Semi-automatic	Semi-automatic	Semi-automatic
2 Operation	Merge	Composition	Mapping+merge	Merge	Mapping	Merge
3 (In)dependence	Ontolingua	Independent	Protégé 2000	Independent	Independent	Independent
4 Representation langages	Ontologua	Labeled and oriented graphs + Horn rules	Rdfs – owl	Concepts taxonomies of populated ontologies	Taxonomies	Owl
5 External resources	No	WordNet	No	No	No	WordNet
6 Lexical matching	No	No	Yes	Yes	Yes	Yes
7 Semanticmatching	Yes	Yes	Yes	Yes	Yes	Yes
8 Instance matching	No	No	No	Yes	Yes	No
9 Structure matching	oui	No	Yes	Corrects conflicts	Yes	Yes
10 User role	Takes decisions affecting the merging process	Validates the proposed mappings	Selects appropriate mappings from todo list	and eliminates duplications	Selects the similarity computation function	Validates computed similarities

Table 1. Comparaison with existing algorithms

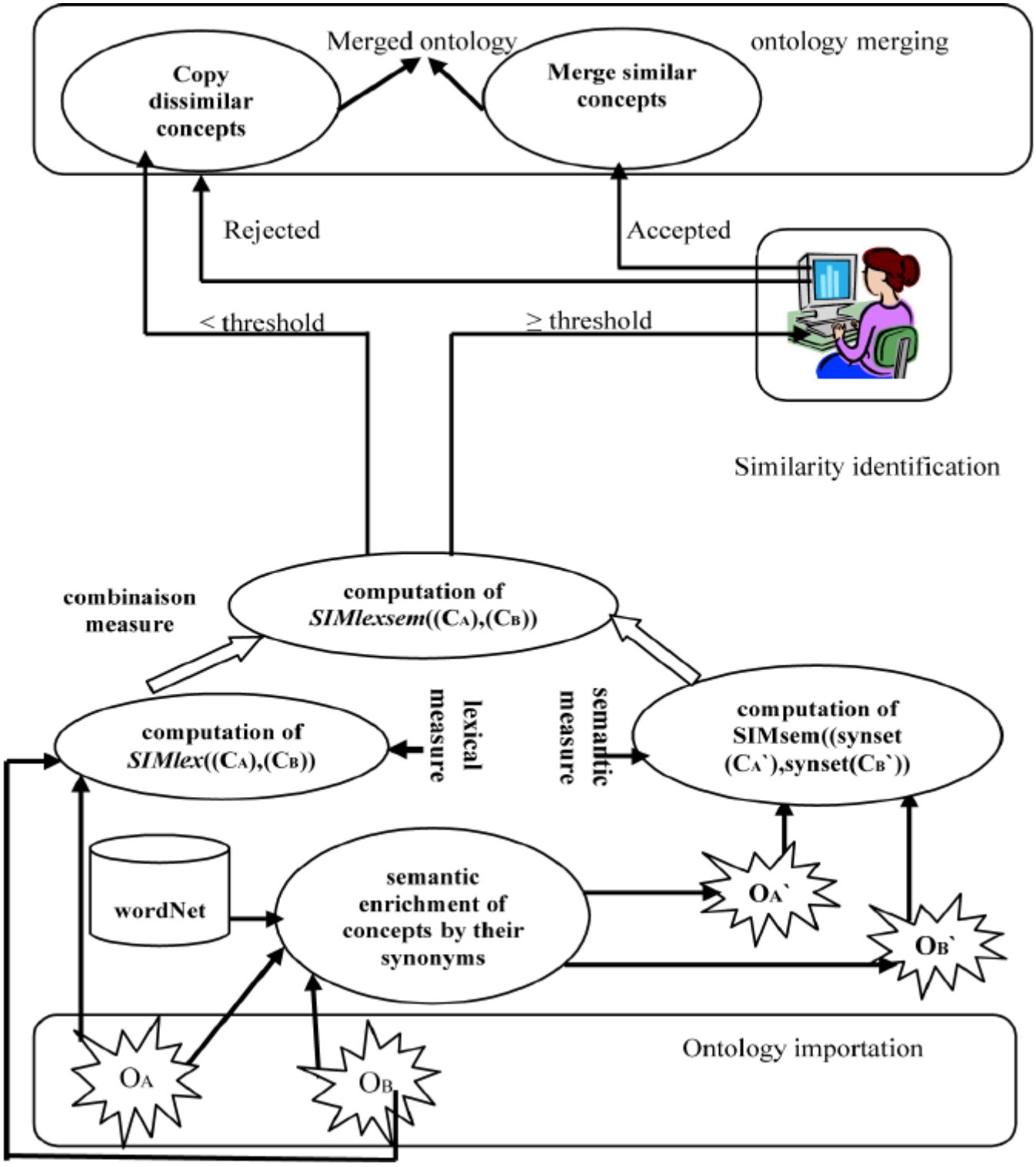

Figure 2. The proposed architecture for the semi-automatic ontology merging system

7. Comparaison With Existing Algorithms

Finally, we compare the whole proposed system with the most known ones that exist in the literature such as: CHIMAERA, ONION, PROMPT, FCA-MERGE and GLUE, throw a set of critical properties. Join Table 1 above:

8. Conclusion

Ontology merging process is a prominent technique to overcome the restrictions and specifications of information and knowledge when the application covers more than one domain. In this work, we have proposed an architecture for a semi-automatic ontology merging system. In this later, the human intervention is prominent to validate the results of similarity computation module. This later combines measures of lexical similarity, based on distance computation between two strings labelling two concepts in a universe of discourse, and semantic similarity, based on the semantic enrichment of the two source ontologies from an external resource "*wordNet*". So, the semantic similarity between the two input concepts is then measured. After that, the concepts, considered similar by combining the two previous results are handled by the knowledge engineer to validate them. If so, the two concepts under discussion are merged into a single concept. Whereas the concepts considered dissimilar or even similar (by the same combination) but their similarity is rejected by the knowledge engineer are directly copied in the resulting ontology. This provides an ontology that covers a hyperdomain of discourse. Our algorithm is far from complete, several improvements must be completed to make it more efficient. In future work, we aim to enhance the mapping discovery results by using other information retrieval techniques and elaborate and use a thesaurus of synonymy specific to the application domain, to enhance the results of the semantic similarity measures. Then, we will choose and study an appropriate application domain, on which we will apply our approach.

References

[1] Gruber, T. (1993). A translation approach to portable ontology specifications. Knowledge acquisition, 5, 199-220.

[2] Aubry, S. (2007). Annotations et gestion de connaissances en environnement virtuel collaboratif. Thèse de doctorat. Université de technologies de Compiegne.

[3] Graul, B. C., Lutz, C., Sattler, U., Turhan, A. Y. (2006). Tasks for ontology integration and merging. Delivrable TONES-D11, The University of Manchister.

[4] Noy, N. F. (2004). Semantic integration: A survey of ontology-based approaches. SIGMOD Rec., 33 (4) 65-70.

[5] Kalfouglou, Y., Schrolmmer, M. (2003). Ontology mapping: The state of the art. *The Knowledge Engineering Rreview*, 18 (1) 1-31.

[6] Maedche, A., Staab, S. (2002). Measuring similarity between ontologies. *In*: proceedings of the European conference on knowledge acquisition and management-EKAW, Madrid, Spain, October 1-4, LNCS/LNAI 2473, Springer, p. 251-263.

[7] Euzenat, J., Shvaiko, P. (2007). Ontology Matching, ISBN: 978-3 540- 49611-3, Springer Berlin Heidelberg, New York.

[8] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions insertions and reversals. Sov. Phys. Dokl., 6, 707-710.

[9] M. A. Jaro (1989). Advances in record linking methodology as applied to the 1985 census of Tampa Florida, dans *Journal of the American Statistical Society*, 84 (406) 414-420

[10] Winkler, W. E. (2006). Overview of Record Linkage and Current Research Directions, dans Research Report Series, RRS.

[11] Hamming Richard, W. (1950), Error detecting and error correcting codes, *Bell System Technical Journal* 29 (2)147–160.

[12] Stemme, G., Maedche, A. (2001). 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.

[13] Noy, N. F., Musen, M. (2000). PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. *In*: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00), Austin, TX, USA.

[14] McGuiness, D., Fikes, R., Rice, J., Wilder, S. (2000). An Environment for Merging and Testing Large Ontologies, *In*: Proceedings of the 17th International Conference on Principales of Knowledge Representation and Reasoning (KR-2000), Colorado, USA.

[15] Doan, A., Madhaven, J., Domingos, P., Halevy, A. (2002). Learning to map between ontologies on the semantic web. *In*: proceedings of the 11th International World Wide Web Conference (WWW 2002), Hawaii, USA.

[16] Mitra, P., Wiederhold, G. (2002). Resolving terminological heterogeneity in ontologies. ECAI'02 workshop on ontologies and semantic interoperability, Lyon, France.

[17] Klein, M., Fensel, D. (2001). Ontology versioning on the semantic web. In the first semantic web working symposium.