# Semantic integration of traditional and heterogeneous data sources (UML, XML and RDB) in OWL2 triplestore

Oussama El Hajjamy*, Hajar Khallouki, Larbi Alaoui and Mohamed Bahaj

Laboratory LITEN,
University Hassan 1,
26000 Settat, Morocco
and
TIC Research Laboratory,
International University of Rabat,
11100, Sala Al Jadida, Morocco
Email: elhajjamyoussama@gmail.com
Email: hajar.khallouki@gmail.com
Email: larbi.alaoui@hotmail.de
Email: mohamedbahaj@gmail.com
*Corresponding author

**Abstract:** With the success of the internet and the expansion of the amount of data in the web, the exchange of information from various heterogeneous and classical data sources becomes a critical need. In this context, researchers must propose integration solutions that allow applications to simultaneously access several data sources. In this perspective, we propose a semi-automatic integration approach of classical data sources via a global schema located in database management systems of RDF or OWL data, called triplestore. Our contribution is subdivided into three axes: 1) an automatic mapping solution that converts classical data sources such as UML, XML and RDB to local ontologies based on OWL2 language; 2) an alignment system of local ontologies based on syntactic, semantic and structural similarity measurement techniques in order to increase the probability of having real correspondences and real differences; 3) a fusion system of pre-existing local ontologies into a global ontology based on the alignment found in the previous step.

**Keywords:** semantic integration; UML; XML; RDB; semantic web; OWL2; RDF; triplestore; ontologies; mapping; alignment; fusion.

**Reference** to this paper should be made as follows: El Hajjamy, O., Khallouki. H., Alaoui, L. and Bahaj, M. (2021) 'Semantic integration of traditional and heterogeneous data sources (UML, XML and RDB) in OWL2 triplestore', *Int. J. Data Analysis Techniques and Strategies*, Vol. 13, Nos. 1/2, pp.36–58.

**Biographical notes:** Oussama El Hajjamy is an IT manager in G4S Company/Cash Solution. He received his PhD in Computer Science from the University Hassan I, Faculty of Sciences and Technology Settat, Morocco in 2019. His areas of interest include web ontologies and semantic data integration.

Hajar Khallouki is a PhD student in the Department of Mathematics and Computer Science, Faculty of Sciences and Technology, Hassan I University Settat, Morocco. Her actual main research interests concern multimedia documents adaptation, context awareness and semantic web.

Larbi Alaoui received his PhD in Mathematics and also a Master in Computer Sciences from the University of Tübingen in Germany. He is currently a Professor at the International University of Rabat. He taught various courses in many fields of mathematics and computer sciences. His research interest in mathematics focuses on the qualitative analysis of models of population dynamics within the framework of the theory he developed on the class of translation semi groups of operators associated to renewal equations. In computer sciences, his research interests include heterogeneous databases interoperability, data mining and big data analytics.

Mohamed Bahaj received his PhD in Applied Mathematics from the University of Pau, France, in 1993. He is currently working as a Professor at the Department of Mathematics and Computer Sciences, University Hassan I, Faculty of Sciences and Technology Settat, Morocco. His research interests include pattern recognition, semantic web and ontology in MAS, and controls of mobiles agents.

This paper is a revised and expanded version of a paper entitled 'Semantic integration of heterogeneous classical data sources in ontological data warehouse' presented at LOPAL'18, Rabat, Morocco, 2–5 May 2018.

## 1 Introduction

The number of information sources on the web and the number of users of these sources have increased exponentially in recent years. These sources are often heterogeneous, in other words they use different models for data representation, such as the relational model, the semi-structured model, the text files, etc. This heterogeneity reflects the diversity of designers' viewpoints of the various systems. As a result, the programming or querying languages of these data sources are different, which poses serious problems for users seeking to combine or integrate information from different sources. These problems have been tackled by data integration. Its main goal is to provide a single access interface or a uniform view for data stored in multiple data sources. These data sources are designed differently and do not use the same vocabulary, this leads to the following problems:

- *Mapping problem:* The UML model for conceptual modelling, the XML model for data exchange and the relational model for data management and storage are ubiquitous and adopted, hitherto, by a large majority of applications constituting the kernels of business information systems, in addition to their permanent presence in the background of the majority of websites. The problem here is the transformation of its different data sources into a common model. In this case, domain researchers encounter an important problem, because some types of reasoning and / or possible constraints in the source model may no longer be possible in the destination model. In addition to this, a model transformation to another leads information loss.

- *Alignment problem:* The models to be integrated were, a priori, built independently of each other; and each database designer uses his own vocabulary to express its needs. As a result, conflicts may arise during the integration process because of heterogeneities that may exist between model elements. These conflicts can be of different types:
  1. *Syntactic conflicts:* This conflict stems from the fact that each database designer uses terminologies that are specific to them. These terminologies may be identical or syntactically close (e.g., two concepts 'incid' and 'incident').
  2. *Semantic conflicts:* This type of conflict occurs when elements having the same meaning are represented differently. The different causes of this conflict are:
     - a The use of different natural languages (Arabic, English, French, etc.) to describe the same concept. For example, 'transport' in French and 'conveyance' in English.
     - b The use of different terms to describe the same concept (synonymy). For example, 'car' and 'vehicle'.
     - c The homonymy when different models use identical names for different concepts.
     - d The different contexts when an object of the real world is represented in the data sources by several representations according to a local context corresponding to each source.
  3. *Internal structural conflicts:* This type of conflict occurs when two distinct elements present the same information about their internal structure, i.e., their properties, domains, instances, etc. For example, two elements are considered similar, if they have proximity in the instances values. The calculation of this similarity is determined by the degree of information shared between two distinct elements.
  4. *External structural conflicts:* This type of conflict uses the hierarchical structure of the common model (OWL ontology in our case) to determine the similarity between concepts. The calculation of this similarity considers the structure of the ontology as a graph and examines only the relations of the type 'is-a'. For example if two elements are linked by such a relationship, they are considered similar then their neighbours can be considered as similar.
- *Fusion problem:* Consists in creating a single output model representing the target schema of the underlying data sources, so as to group all the similarities and dissimilarities and avoid their redundancy in the merge result.

In the field of data integration, OWL ontologies can contribute to solving the problems mentioned above. First, OWL provides a common vocabulary, a grammar for publishing data and a semantic description of data which can be reused and extended. Second, Ontology-based data integration systems benefit from the reasoning capabilities offered by the semantic web technologies. Lastly, OWL makes it possible to achieve a common and shared knowledge that can be transmitted between people and between application systems.

To answer these problems, we propose a semi-automatic integration approach, via a global schema located in a triplstore, integrating all aspects: semantic, syntactic, internal structural and external structural. Semi-automatic since our method requires human intervention to validate the results obtained by the similarity identification system on the base of its own needs. Our approach has three subsystems:

1 A mapping system: to convert the elements of classical data sources into local ontologies.

2 A similarity identification system: to identify similar elements that will be merged with the last subsystem.

3 A fusion system: to merge local ontologies into a global ontology based on typed graph grammars and Simple PushOut (SPO) algebraic approach.

Existing works on making classical data available as ontologies are not dealing with the integration of such data issued from various sources. Each of these works mainly deals separately, and not within a global integration framework, with a specific task in one of the various steps of the process of integration. Our aim is to tackle the aforementioned integration problem to come up with an approach leading to a system that is based on a uniform view of various data sources providing a single access interface for data stored in multiple data sources.

The rest of this paper is organised as follow. Section 2 present an overview of existing work that we consider to be major related to the mapping, alignment and fusion of ontological data. Section 3 describes our integration process; it is divided into three sub-parts describing the three subsystems of our integration model. The experimental part of our prototype is presented in Section 4. Finally, Section 5 concludes our work by summarising the main contributions and presenting a discussion of our perspectives.

## 2 Existing semantic integration approaches

During the last years, because of the importance of ontologies, many studies have been proposed to integrate classical data sources into ontologies to ensure interoperability between traditional information systems. These studies have separately studied the different tasks of an integration system (mapping, alignment and fusion) and until now there are not yet effective proposals that provide an end-to-end solution to this problem. We first addressed existing works related to the mapping task of one type of such data sources into ontologies. In a second step we give a discussion on relevant works on similarities between ontologies. Finally, we also give an overview of solutions existing for ontology fusion.

### *2.1 Mapping systems*

In order to evaluate the existing approaches, we highlight in this section, the different methods that were interested in the construction of ontologies from classical data sources:

- *UML-to-ontology:* Due to the widespread use of UML and OWL languages, it is no wonder that there are many works in the literature whose goal is to study the different relationships between UML and OWL and propose a transformation from UML to OWL. Khan and Porres (2015) present transformation rules of selected elements of UML diagrams to ontology. The rules are used to check the internal consistency between several UML diagrams. Gherabi and Bahj (2012) define a correspondence between the class diagrams of UML and OWL by using a mathematical representation of the class diagram. Zedlitz et al. (2011) considered the mapping between UML elements and OWL2 constructs such as disjoint and complete generalisation, generalisation between associations, composition and enumeration. However, we believe that our method UML2OWL2 (El Hajjamy et al., 2016) give a solution to all aforementioned limitations of existing approaches in order to provide the semantic world as complete as possible conversion technique that allow to easily and fully deduce all conceptual details of the considered UML specifications relative to the analysis, conception and design of the associated modelled systems.
- *XML-to-ontology:* We can found several approaches that deal with XML to OWL mapping: Huang et al. (2015) propose a template that can handle extremely large XML data and provides user friendly templates composed of RDF triple patterns including simplified XPath expressions. Kramer et al. (2015) describes a suite of domain-independent software tools that enable the completely automatic generation of OWL model files and instance files from XSDL model files and XML instance files. Breitling (2009) proposes a standard mapping method from XML to RDF via XSLT. It provides direct conversion and contains XPath information for retaining the hierarchical information of the XML data source. Bedini et al. (2010) propose a tool called 'Janus', this last provides automatic derivation of ontologies from XS files by applying a set of derivation rules. Then, the same group proposed a method based on patterns (Bedini et al., 2011), that deals with 40 patterns and convert each pattern to equivalent OWL ontology. All aforementioned ontology-based transformation present limitations in treating various important XSD elements related to the art of elements, relations or constraints. Our approach (El Hajjamy et al., 2017) aims at defining a correspondence between the xml schema and OWL2 ontology. It maintains the structure as well as the meaning of XML schema. Moreover, our mapping method provides more semantics for XML instances via adding more definitions for elements and their relationships in OWL ontology by using OWL2 functional style syntax.
- *RDB-to-ontology:* There are many researches that have been proposed to achieve RDB to OWL conversion (Ahmed et al., 2016; Alaoui et al., 2014a; Ling and Zhou, 2013) but most of them contain simple and limited cases, rules, and doesn't cover most complex relations and constraints. This has allowed us to build an associated general and complete mapping algorithm (Alaoui et al., 2014b) that covers different aspects of the relational model which are relevant for the mapping process. The algorithm deals among others with various multiplicities for relationships, relation transitivity, circular relationships, self referenced relationships, binary relations with additional attributes including many-to-many relations and constraints such as check constraints (check values, check in).

## 2.2 Alignment systems

In the literature, the alignment system of two or more ontologies is the ability to detect a set of correspondences between the concepts of these ontologies. We present the existing work according to the heterogeneity of data classification (Section 1) as follow:

- *Syntactic similarity:* Is based on the calculation of the distance between two characters. Different syntactic similarity distance calculation algorithms exist in the literature such as those of Levenshtein (1966), Hamming (Winkler, 2006), Jaro (Klein and Fensel, 2001), n-gram (Cavnar and Trenkle, 1994), etc. They are all based on the same hypothesis described by Maedche and Staab (2002) who states that two terms are similar if they share enough important elements.
- *Semantic similarity:* Is a human ability that machines can only reproduce very poorly. Researches on this subject are made in several fields: artificial intelligence, medicine, cognitive science, and this for many years. Various methods have been proposed for semantic similarity detection techniques: In Schadd and Roos (2012), virtual documents (context) represent the meaning of ontology entities and WordNet entries and entities are coupled according to their document similarities. The notion of context has also been exploited in (Wang, 2011), where semantic description documents refer to the information about concept hierarchies, related properties and instances. Another method is proposed by Leacock and Chodorow (1998) which is based on calculating the length of the shortest path between two synsets of WordNet. Amrouch and Mostefai (2012) used WordNet to construct a synonymy vector for each concept of the first ontology, and then compares it with all the concepts of the second to find the concept that is most similar to the concept in question.
- *Internal structural similarity:* Calculates the similarity between two elements by exploiting the information relating to their internal structure (restrictions and cardinalities on the properties, values of the instances, etc.). Most internal structural similarity identification systems compare ontologies from instances. Among the works in this field we can cite: Jaccard similarity measure (Smadja et al., 1996) which is defined by the number of common internal elements divided by the total number of elements minus the number of common elements. GLUE in (Doan et al., 2003) exploits multiple machines learning to find semantic mappings between concepts stored in distinct and autonomous ontologies. Its strategy uses the learning technique (such as the naive Bayes learning technique) to find matches between two ontologies. GLUE includes several learning modules (learners), which are entrained by instances of ontologies. Khiat and Benaissa (2015) developed an instance-based ontology alignment approach that shares common instances. This approach consists first to extract the argument and event structures using the generative lexicon, from each instance of a concept of the source ontology, and the common semantic features will be grouped in a document. Then this document will be compared with another document that contains the semantic features of instances of a target concept combined with WordNet in order to strengthen the similarity.
- *External structural similarity:* The objective of this technique is to obtain results for concepts related to each other by a subsumption relation. Lin (1998) performed a comparison between the methods of structural similarity measures. He deduced that

the technique proposed by Wu and Palmer (1994) has the advantage of being simple to compute and more efficient. However, it has a limit because with this measure it is possible to obtain a higher similarity between a concept and its surroundings with respect to this same concept and a child concept. To solve this problem Slimani et al. (2007) has developed a similarity measure extension based on the Wu and Palmer measurement that penalises the similarity of two distant concepts that are not located in the same hierarchy. (Ngo et al., 2012) propose a similarity propagation (SP) graph which uses fixed-point computation to determine corresponding nodes in the graphs. SP is effective and has been embedded in some outstanding ontology matching systems such as YAM++ (Ngo and Bellahsene, 2012).

### *2.3 Fusion systems*

Different ontology merge tools exist in literature. Most of these are semi-automatic and require the intervention of a knowledgeable engineer to validate the results obtained. The most known are:

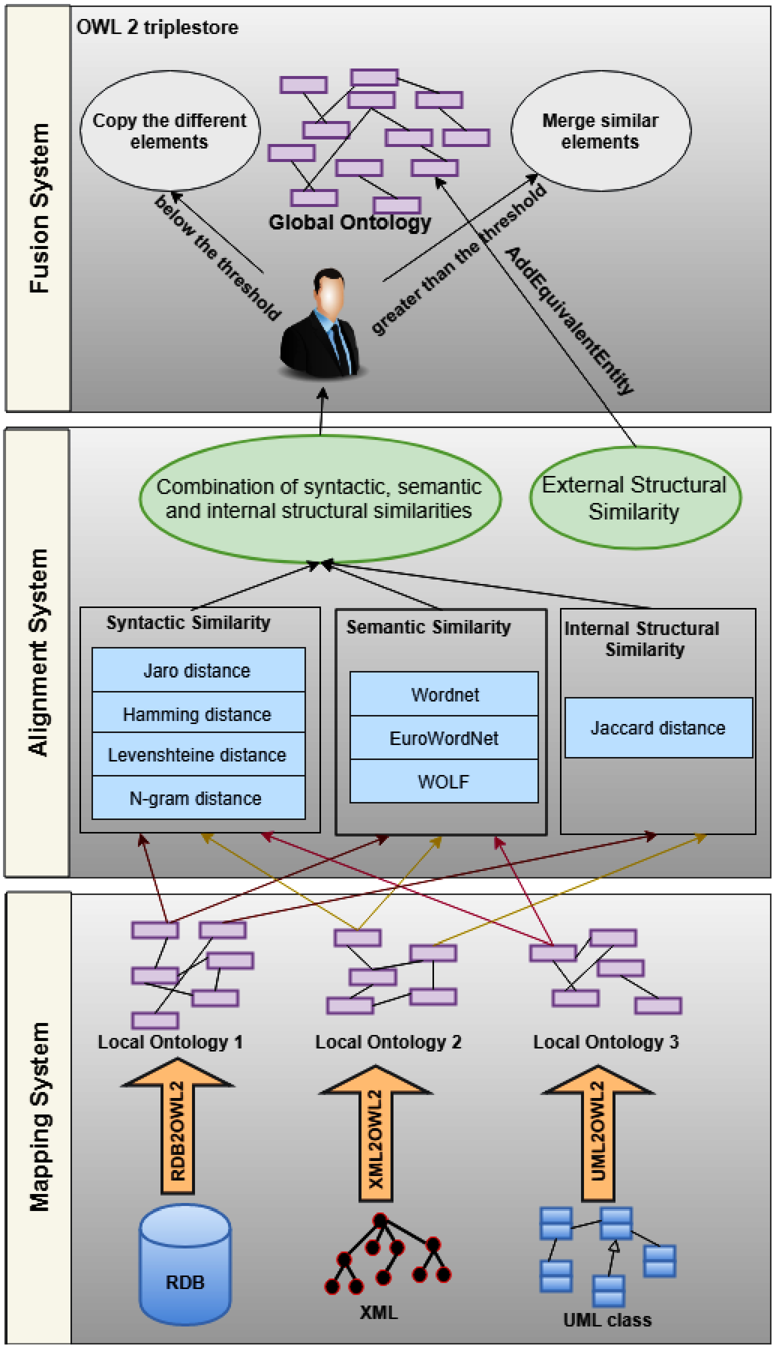
- *FCA-merge:* Is a symmetric approach proposed by Stumne and Maedche (2001) that allows merging ontologies based on the formal analysis of concepts. Its process is as follows: first, perform a linguistic analysis of the two ontologies and extract their instances. Once instances are retrieved, use FCA techniques to merge the two contexts and calculate the trellis. Then, generate the global ontology from the constructed trellis. Finally, to resolve conflicts and eliminate duplications, the user is invited through a 'question-and-answer' mechanism to choose the proposals that suit him the most.
- *PROMPT (Noy and Muzen, 2000):* Is a protégé plug-in for ontology merge. It looks for linguistic similarity points between the concepts of the two source ontologies and proposes a list of all the possible merging actions (to-do list). Then the user can choose the proposals that suit him the most.
- *MMOMS:* Framework proposed by Li et al. (2010) to merge OWL ontologies. It is based on learning machines, WordNet and structural techniques to look for similarity. It uses a merge algorithm that addresses the concepts, relationships, and attributes of both ontologies.

Raunich and Rahm (2011) have proposed a taxonomy merge approach that consists of two phases: the first phase generates a graph from two taxonomies and their corresponding result. The second phase adopts an asymmetric approach called ATOM to ensure the merge of the two source and destination taxonomies. This approach is capable of avoiding redundancies, but it only deals with ontologies composed by concepts and hierarchical relations (is-a).

## 3 Our semantic integration process

Our approach aims to provide a unique and transparent interface of classical data sources (UML, RDB, XML) via a global schema (OWL) located in ontological data warehouse. To deal with the heterogeneities of models and data, we have chosen ontologies as a common model. The latter ensures a semantic equivalence between the different models. Our strategy consists of three distinct phases, as shown in Figure 1.

**Figure 1** Proposed general approach (see online version for colours)



In the first step, the system extracts files from existing data sources, and applies our mapping algorithms ‘RDB2OWL2’, ‘UML2OWL2’ and ‘XSD2OWL2’ to create their OWL2 equivalents. It should be noted that the use of OWL2 to generate the resulting ontology allows us to benefit from a more powerful inference system, as well as OWL2 extends OWL1 with new features based on actual use in applications. It is indeed possible with OWL2 to define more constructs to express additional restrictions and obtain new characteristics on the properties of the modelled object.

In the second step, our tool imports the generated ontologies and uses Syntactic, Semantic and Structural Similarity techniques to determine the correspondences between the concepts of the ontologies to merge.

The final step is to merge the local ontologies based on the matches found in the previous step. We present ontologies with the formalism of typed graph grammars to merge ontologies using the SPO algebraic approach.

Our approach is asymmetrical; it requires the choice of the source ontology. The concepts of the source ontology will be preserved while the non-redundant concepts of the other ontologies will be added to the global ontology. On the other hand, the choice of the source ontology is based on several criteria (the size, the depth, the expressivity of the ontologies to merge). In this work, we are not interested in this type of comparison and we consider that it is up to the user to specify the source ontology.

## *3.1 Our mapping process from classic data sources to local ontologies*

This step consists of designing local ontological models from classical models, while keeping the operating principle of source models and while minimising the loss of information.

From the point of view of entity/association models for conceptual modelling, we use our UML2OWL2 method. This method aims to generate OWL ontologies from an existing UML class diagram. It is based on the XMI format, which provides a storage and knowledge exchange standard for UML model.

From the point of view of semi-structured models, we use our XSD2OWL2 approach. This solution takes an existing XML schema (XSD) as input, loads the XSD document and parses it using the DOM parser. Then, it extracts its elements with as many constraints as possible and applies our mapping algorithm to create the resulting OWL2 document. For a complete transformation the mapping of XML elements is added to our approach.

From the point of view of relational models, we use our approach RDB2OWL2, which makes it possible to automatically build OWL2 ontologies via a transformation process of relational databases. The goal of this solution is to provide a general transformation algorithm that covers all constraints, preserves the semantics of the source RDB, and maintains data consistency and integrity. This process operates on two levels: The schema level in which the terminology part or TBOX of the ontology is generated from a schema of the source RDB. The level of data instances in which data stored as records is converted to the factual level or ABOX of the ontology.

### 3.2 Our ontologies alignment process

Our objective is to design a semi-automatic local ontology fusion algorithm (generated in the previous step) based on a set of similarity search techniques. The alignment module covers all the elements of the comparison types in order to detect all the matches, and combines all the comparison types (syntactic, semantic and structural) in order to increase the probability of having real correspondences and real differences. The results obtained are accepted or rejected by the user according to his needs. In our alignment system, we only compare elements of the same nature (two classes, two data properties, two instances, etc.).

The process begins by comparing syntactically the ontology classes two by two. It tries to find out if there is a syntactic equivalence between them. We choose to compare semantically two classes only after verifying that they are not syntactically equivalent. Indeed, this allows reducing the search time of the equivalent classes.

Afterwards, a semantic comparison, two by two, of classes is performed by referring to the WordNet, Wolf or EuroWordNet lexical databases.

When the concepts of local ontologies are neither syntactically nor semantically equivalents and they share a large number of common instances (ABOX part), then they are necessarily equivalent. In this case, we compare their instances to identify their similarity.

If two classes are deemed equivalents, then we compare their internal and external structures. The comparison of the internal structure of two classes consists of comparing their 'DataProperties' syntactically two by two. If the 'DataProperties' are not syntactically equivalents, we compare them semantically. In the case where 'DataProperties' are syntactically or semantically equivalents, then we compare their internal structures 'types' and 'cardinalities'.

The comparison of the 'ObjectProperties' relations are carried out by comparing them syntactically, or semantically in the case where they are not syntactically equivalents. Note that we do not compare them in internal structure (maximal and minimal cardinality, navigability, transitivity, reflexivity, etc.). Indeed, we consider that two 'ObjectProperties' relations can be equivalent even if they are different in internal structure.

We have described so far, all the alignment process of the TBOX part of the local ontologies. Once this step is completed, the alignment process of the ABOX part will begin. For this, we compare syntactically then semantically the instances of the classes considered equivalents to avoid their redundancy in the resulting global ontology.

The last step is to enrich the global ontology by exploiting the subsumption relation 'is-a' of the global ontology entities, in order to determine the most similar pairs of entities based on the hierarchy or arcs distances in a hierarchical semantic network.

#### 3.2.1 Syntactic similarity measure

To measure the degree of syntactic equivalence, we compare the elements of the models syntactically. To do so, we use a combination of string-based techniques namely: Jaro, hamming, Levenshtein and N-gram measures. These measures identify the syntactic identity in the case where the measure is equal to 1, and the syntactic inclusion, in the case where the measure is strictly between 0 and 1. If the measure is equal to 0, then the two concepts are different.

The main advantage of this approach is the combination of several syntactic similarity detection strategies, which increase the chance of finding all the terminological correspondences between the ontologies elements to be aligned.

*Jaro distance*: is based on the number and order of common characters between two chains $C_1$ and $C_2$. It is defined as follows:

$$d_j(C_1, C_2) = \frac{1}{3}\left(\frac{m}{|C_1|} + \frac{m}{|C_2|} + \frac{m-t}{m}\right) \tag{1}$$

$m$ The number of corresponding characters. Two chains $C_1$ and $C_2$ are considered as corresponding if their distance does not exceed: $\left\lfloor \frac{\max(|C_1|, |C_2|)}{2} \right\rfloor - 1$.

$|C_1|$ The length of the chain $C_1$.

$t$ The number of transpositions. It is calculated by comparing the i-th corresponding character of $C_1$ with the i-th corresponding character of $C_2$. The number of times these characters are different, divided by two, gives the number of transpositions.

*Hamming distance:* Is used in computer science, telecommunication and signal processing. It plays an important role in the algebraic theory of the correcting codes and makes it possible to quantify the difference between two sequences of symbols. This distance counts the number of positions in which the two strings differ. It is defined by $F: S \times S \rightarrow [0, 1]$ such that:

$$F(x, y) = \left(\sum \min(|x|, |y|) x[i] \# y[i]) + (|x| - |y|)\right) \tag{2}$$

*Levenshtein distance:* It allows knowing the minimum number of characters edits (insertions, deletions, and substitutions) to transform one string into another. If the number of differences between the two chains is large then the transformation cost is important. This distance is robust against misspellings.

*N-gram distance:* it generates a similarity measure between 0 and 1 of two stings based on the calculation of the number of their identical substrings. This measure is effective when part of one element string, is included in the string of the other. For example: 'CodBanK' and 'CodB'. Let n-gram ($S$, $l$) be the set of all substrings of length 'l', the distance n-gram between two strings $S$ and $T$ is defined by the following similarity function:

$$\sigma = \left|\text{n-grams}(s, l) \cap \text{n-grams}(t, l) / l * \text{Min}(|s|; |t|)\right| \tag{3}$$

The combination of calculated similarities at the syntactic level is presented in our Algorithm 1 'SimSyntactic()'. The latter compares the elements of the same nature of the TBOX part of the local ontologies.

**Algorithm 1** Our SymSyntactic() algorithm

```
Input: E a set of ontology entities to align
Output: A alignment
Begin
For each (ei, ej) $\in$ E
```

```
If (SimJaro(ei, ej) < threshold) then
    SimJaro(ei, ej) = 0
If (SimHamming(ei, ej) < threshold) then
    SimHamming(ei, ej) = 0
If (SimLevenshteine(ei, ej) < threshold) then
    SimLevenshteine(ei, ej) = 0
If (SimN-gram(ei, ej) < threshold) then
    SimN-Gram(ei, ej) = 0
End Loop
For each (ei, ej) ∈ E
    SimSyntactic(ei, ej) = (SimJaro(ei, ej) + SimHamming(ei, ej) + SimLevenshteine(ei, ej)
    + SimN-gram(ei, ej) ) / 4
    If (SimSyntactic(ei, ej) > threshold) then
        A = A + (ei ≡ ej)
End Loop
Return A
End
```

Example: Calculate the syntax distance between ‘conveyance’ and ‘conv’, and ‘conveyance’ and ‘transport’. We assume that threshold = 0.5

- SimSyntactic(conveyance, conv) = 0.89 > 0.5. Then ‘conveyance’ and ‘conv’ are syntactically similar.
- SimSyntactic(conveyance, transport) = 0. Then ‘conveyance’ and ‘transport’ are syntactically different.

The SymSyntactic() algorithm is suitable when you want to compare two terms or two short strings but there are cases where you need to compare long texts. In this case, we chose to adapt the LIUPPA syntactic similarity metric proposed in (Nguyen et al., 2013). It is a hybrid approach of syntactic comparison of strings which gave very good results even in cases where labels to be compared are composed of several words in which the order is important. The LIUPPA method can be summarised in two steps:

Step 1 The two labels of concepts in question are considered as sequences of tokens this step is to use X a metric based on characters to compare pairs of tokens. Therefore, if two of tokens are equivalent (according to the metric and the similarity threshold used) they will be represented by the same symbol.

Step 2 This step is to use Y a second metric based on the characters to calculate the similarity between sequences of symbols. Note that the symbol sequences to be compared are also strings of characters in which each character is a symbol.

We adapted our SymSyntactic() algorithm for both X and Y metric, which yielded good results.

### *3.2.2 Semantic similarity measure*

When several symbolic names cover the same concept but their names are different (synonymy), the SimSyntactic < threshold does not reflect the reality. To solve this problem, semantic similarity measurement is essential (example: conveyance and transport). To do so, we use a lexical database (English WordNet dictionary, EuroWordNet multilingual dictionary or WOLF French WordNet dictionary) so that we can deduct the meaning of a word. By articulating on WordNet two concepts are equal if their synset overlap. For example: synset = {transport, conveyance}.

The measurement of semantic similarity between two concepts $C_1$ and $C_2$ is defined by calculating the number of common synonymy relations (synset) as follows:

$$SimSem(C_1, C_2) = \frac{2 \times card(synset(C_1)) \cap (synset(C_2))}{card(synset(C_1)) + card(synset(C_1))} \tag{4}$$

$C_1$ and $C_2$ are considered semantically similar if *SimSem()* is greater than a threshold that will be determined empirically. *SimSem*(transport, conveyance) = 2 × 2/4 = 1, then ‘transport’ and ‘conveyance’ are semantically similar.

### *3.2.3 Internal structural similarity measure*

When the local ontologies concepts are neither syntactically nor semantically equivalent and share a large number of common instances, then they are necessarily equivalent. In this case, we need to compare their instances to identify their similarity.

Instance-based similarity identification techniques are very useful when ontologies contain a large number of instances. Most of these techniques use sets metrics, such as the Jaccard index that allows us to evaluate the similarity between two sets.

$$Sim_{Jacc} = \frac{|A \cap B|}{|A \cup B|} \tag{5}$$

To calculate the intersection, we use our syntactic similarity measurement method mentioned above. $K$ denotes the intersection set satisfying the following formula:

$$K = (a, b) / a \in A, b \in B \; avec \; SimSyntactic(a, b) = 1$$

Then the instance-based similarity is calculated as follow:

$$Sim_{Ins} = \frac{|K|}{|A \cup B|} \tag{6}$$

### *3.2.4 External structural similarity measure*

Structural similarity identification methods use the hierarchical structure of the ontology and are based on arc counting techniques. We use it to enrich the global ontology.

The similarity between the entities is determined according to their positions in their hierarchies. It is calculated once for each pair of nodes. The nodes of the two ontologies are classified by category (or type). The method (Slimani et al., 2007) which inspires advantages of the work (Wu and Palmer (1994) is based on the following principle: let $C_1$ and $C_2$ two elements of the global ontology and $C$ their subsuming concept, the principle of calculating similarity is defined by the following formula:

$$SimStr(C_1, C_2) = \frac{2 \times depth(C)}{depth(C_1) + depth(C_2)} \times fp(C_1, C_2) \tag{7}$$

If $C_1$ and $C_2$ are not in the same path, then:

$$fp(C_1, C_2) = \frac{1}{|depth(C_1) - depth(C_2)| + 1} \tag{8}$$

Else if $C_1$ is ancestor of $C_2$ or the opposite, then: $fp(C_1, C_2) = 1$

The advantage of this measurement is that one can obtain a higher similarity between a concept and a child concept compared to this same concept and its surroundings.

## *3.3 Our local ontologies fusion process*

The ontology fusion is the creation of a global ontology from several existing ontologies. However, this step can cause many conflicts.

In order to resolve these conflicts, we have developed a set of guidelines based on our alignment process introduced in the previous Section. These directives indicate the actions to be applied to decide how the elements will appear in the result model, for example, the creation, the deletion and the renaming of the elements. Our fusion approach used TGGOnto(GTO,GO,RO) model (Mahfoudh et al., 2016): based on typed graph grammars with:

- GTO (NT, ET) is a type graph representing the OWL2 ontology meta-model and specifying the type of nodes and edges of the initial graph.
- GO is the initial graph representing the source ontology. It is defined as a system GO (N, E) where N, E correspond respectively to the sets of nodes and edges of the graph, and an application s: E → N × N which associates for each edge a source and target node.
- RO is the set of rewrite rules describing ontological changes. These rules make it possible to transform the initial graph GO and defined by RO(NAC, LHS, RHS, CHD) where: LHS and RHS respectively specify the left and right sides of a rule. The left side shows the structure that must be found in a host graph G to be able to apply the rule and the right part describes the rewriting rule that replaces L in G. NAC (Negative Application Conditions) defines the conditions that should not be checked for the rewriting rule to be applied. CHD presents the derived changes.

The classification of rewriting rules used in our method is presented in the Table 1.

*SPO:* is an algebraic method of graph transformation proposed by Löwe (1993). The stages of the transformation are as follows:

Identify the graph LHS in G according to a morphism m: LHS → G.

Remove from the graph G, the graph m(LHS) - m(LHS ∩ RHS) and delete all the suspended edges.

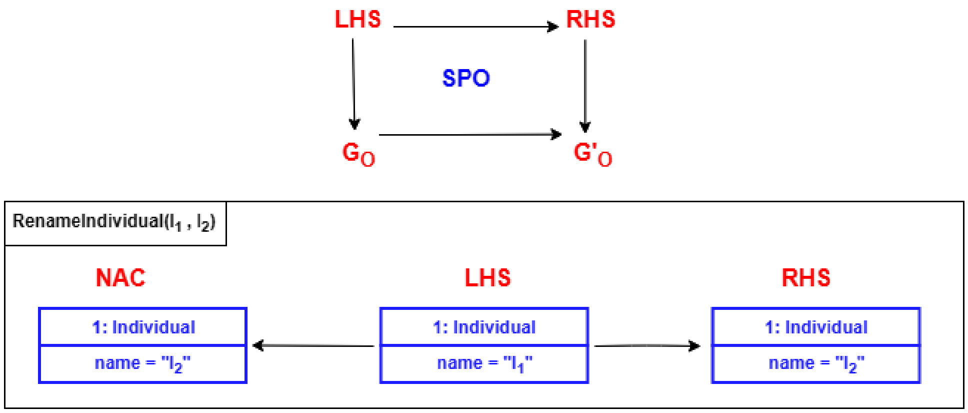
Add the graph m(RHS) - m(LHS ∩ RHS) to the initial graph G

Example: Rewriting rules of 'RenameIndividual($I_1$, $I_2$)' change with the SPO approach.

**Table 1** Formalising ontology changes

| *Add* | *Delete* | *Rename* |
|---|---|---|
| Class | Class | Class |
| Individual | Individual | Individual |
| ObjectProperty | ObjectProperty | ObjectProperty |
| DataProperty | DataProperty | DataProperty |
| SubClass | SubClass | |
| EquivalentClasses | EquivalentClasses | |
| DisjointClasses | DisjointClasses | |
| SubObjectProperty | SubObjectProperty | |
| EquivalentObjectProperties | EquivalentObjectProperties | |
| DisjointObjectProperties | DisjointObjectProperties | |
| ObjectPropertyAssertion | SubDataProperty | |
| SubDataProperty | EquivalentDataProperties | |
| EquivalentDataProperties | DisjointDataProperties | |
| DisjointDataProperties | CardinalityResriction | |
| DataPropertyAssertion | AllValuesFromRestriction | |
| CardinalityResriction | SomeValuesFromRestriction | |
| AllValuesFromRestriction | | |
| SomeValuesFromRestriction | | |
| HasValueFromRestriction | | |

**Figure 2** Rewriting rules of 'RenameIndividual' change with the SPO approach (see online version for colours)



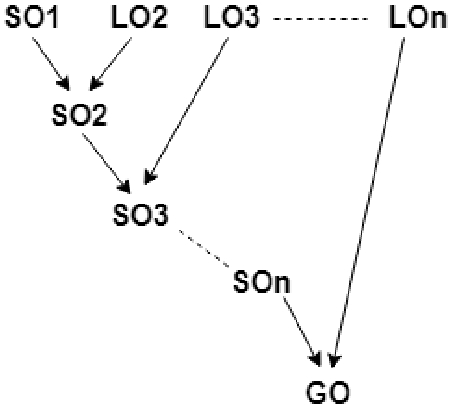
NAC {I₂}: To avoid data redundancy, this rule means that the sub-graph must not exist in the ontology graph for the change to be applied.

LHS {I₁}: specifies the sub-graph (of type 'IndividualName') that we want to rename.

RHS {I₂}: is the sub-graph who has to replace LHS.

Our approach is asymmetrical. On the one hand, for two ontologies $O_1$ and $O_2$ Merge($O_1$, $O_2$) $\neq$ Merge($O_2$, $O_1$). On the other hand, the fusion method adopts the 'one pair at time' strategy (Figure 3) and requires the definition of the source ontology which elements will be preserved and only the non-redundant elements of the other ontology will be added to the global ontology.

**Figure 3** One pair at time fusion strategy



Our system called MergeOnto (Algorithm 2) takes as input two ontologies SO and LO and return a third global ontology GO. This merge-based system is used to determine how the elements presented in the source models will appear in the result model. For equivalent elements that have:

- *Syntactic heterogeneity*: The name of the result element is the one with the most characters (the longest name). For example, the result DataPrperty of the two equivalent DataProperties 'CodBank' and 'CodB', will be 'CodBank'. This approach allows keeping the name providing the most information.
- *Semantic heterogeneity*: We use the solution of the source ontology. Indeed, one of the two local ontologies is chosen as source ontology, in order to take the names of the elements in case of differences between them. For example, considering O1 the source ontology, and 'Staff' a class of O1 equivalent to the class 'Personal' of O2, then the resulting class of the two merged classes will take the name 'Staff' of the class belonging to the source ontology.
- *External structural heterogeneity*: By applying the structural similarity measurement rule, we add 'EquivalentEntity' to elements deemed similar and accepted by the Knowledge Engineer.

**Algorithm 2** Our MergeOnto() algorithm

```
Input: SO, LO ontologies
Output: GO ontology
Begin
/* Syntactic an Semantic similarity
For each SimSyn(N, N′) > threshold do
```

```
If length(N) ≥ length(N′) then
    O′ ← SPO_AddEntity(N)
Else length(N) < length(N′) then
    N ← SPO_RenameEntity( SO, N, N′)
    O′ ← SPO_AddEntity(N)
End If
End Loop
For each SimSem(N, N′) > threshold do
    O′ ← SPO_AddEntity(N)
End Loop
/* Fusion function merge the similar entities and copy the different entities in SO
GO ← SPO_Fusion(O′, SO)
/*Structural Similarity:
For each element N′ ∈ N″Type{subsumption} in GO
O″ ← Entity(GO, N″)
End Loop
For each N″ in O″
For each Ni″ in O″
If (SimStr(N″, Ni″) > threshold) then
    GO ← SPO_AddEquivalentEntity(GO, N″, Ni″)
End If
End Loop
End Loop
End
```

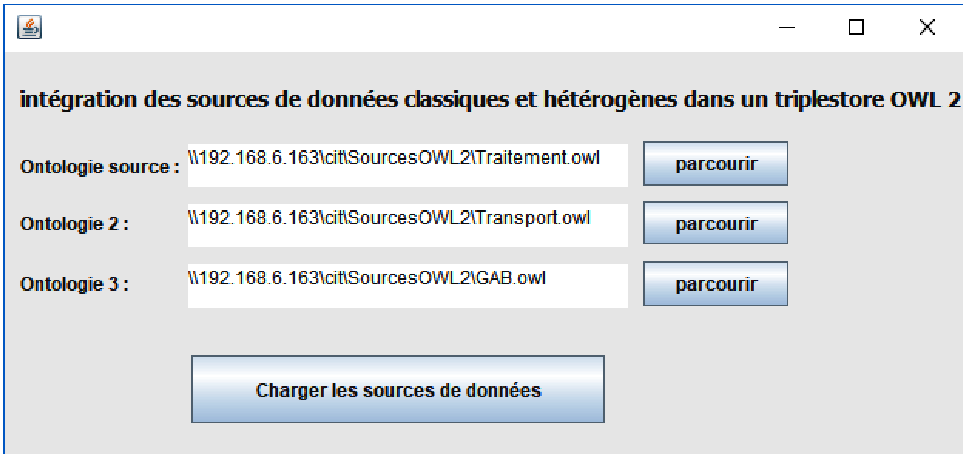
## 4 Implementation and validation

To evaluate our model a tool has been implemented and validated in a project in which we have participate. The purpose of this project is to develop a collaborative platform in the ‘Cash solution’ domain between the following applications: Cash processing, CIT Transport and ATM. For that, the users of the new platform want to ask questions that cover several data sources simultaneously.

Our prototype implements the three steps of the integration solution. The First generates three OWL2 ontologies (Transport.owl, Treatment.owl, and Gab.owl) from application databases, using our RDB2OWL2 algorithm. Then, our integration prototype allows to define the path of the OWL files and to choose the source ontology. We have chosen ‘Treatment.owl’ as the source model because it contains the largest number of classes. The first interface of our prototype is shown in Figure 4.

The execution of our alignment system is achieved by clicking on the button ‘Alignment des ontologies locales’. Once the system finishes its task, a new table appears, in which the result of the correspondences is displayed (Figure 5).

**Figure 4** Our prototype interface (see online version for colours)

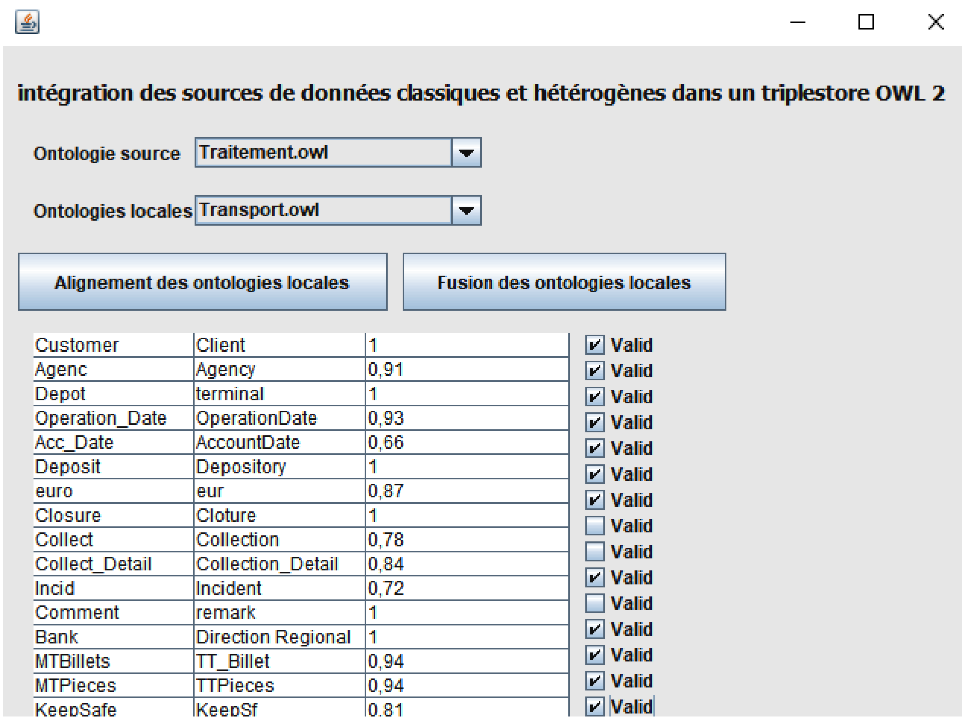

intégration des sources de données classiques et hétérogènes dans un triplestore OWL 2

Ontologie source : \\192.168.6.163\cit\SourcesOWL2\Traitement.owl

Ontologie 2 : \\192.168.6.163\cit\SourcesOWL2\Transport.owl

Ontologie 3 : \\192.168.6.163\cit\SourcesOWL2\GAB.owl

**Figure 5** The result of our alignment system (see online version for colours)


intégration des sources de données classiques et hétérogènes dans un triplestore OWL 2

Ontologie source Traitement.owl

Ontologies locales Transport.owl

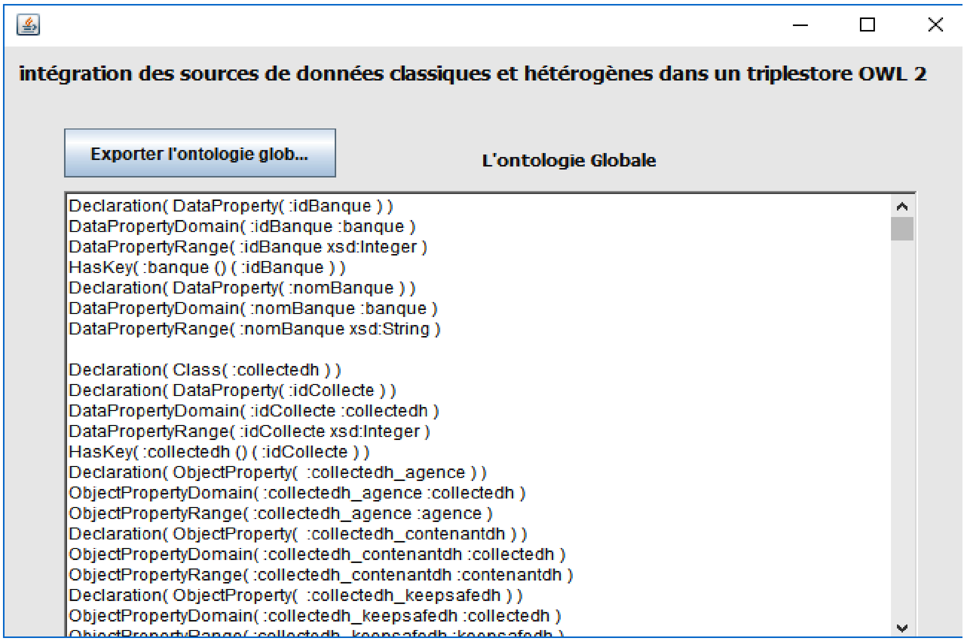
Alignement des ontologies locales
Fusion des ontologies locales

| | | | |
|---|---|---|---|
| Customer | Client | 1 | ✓ Valid |
| Agenc | Agency | 0,91 | ✓ Valid |
| Depot | terminal | 1 | ✓ Valid |
| Operation_Date | OperationDate | 0,93 | ✓ Valid |
| Acc_Date | AccountDate | 0,66 | ✓ Valid |
| Deposit | Depository | 1 | ✓ Valid |
| euro | eur | 0,87 | ✓ Valid |
| Closure | Cloture | 1 | ☐ Valid |
| Collect | Collection | 0,78 | ☐ Valid |
| Collect_Detail | Collection_Detail | 0,84 | ✓ Valid |
| Incid | Incident | 0,72 | ☐ Valid |
| Comment | remark | 1 | ✓ Valid |
| Bank | Direction Regional | 1 | ✓ Valid |
| MTBillets | TT_Billet | 0,94 | ✓ Valid |
| MTPieces | TTPieces | 0,94 | ✓ Valid |
| KeepSafe | KeepSf | 0,81 | ✓ Valid |

Our integration approach is semi-automatic, it offers to user the opportunity to validate the elements he deems equivalent (Figure 5). Then, the launch of our fusion system is achieved by clicking on the button ‘Fusion des ontologies locales’.

An overview of the interface developed for global ontology generation is given in Figure 6. The interface displays all the classes, properties, and OWL assertions.

**Figure 6** The result of our fusion system (see online version for colours)


intégration des sources de données classiques et hétérogènes dans un triplestore OWL 2
Exporter l'ontologie glob...
L'ontologie Globale

```
Declaration( DataProperty( :idBanque ) )
DataPropertyDomain( :idBanque :banque )
DataPropertyRange( :idBanque xsd:Integer )
HasKey( :banque () ( :idBanque ) )
Declaration( DataProperty( :nomBanque ) )
DataPropertyDomain( :nomBanque :banque )
DataPropertyRange( :nomBanque xsd:String )

Declaration( Class( :collectedh ) )
Declaration( DataProperty( :idCollecte ) )
DataPropertyDomain( :idCollecte :collectedh )
DataPropertyRange( :idCollecte xsd:Integer )
HasKey( :collectedh () ( :idCollecte ) )
Declaration( ObjectProperty(  :collectedh_agence ) )
ObjectPropertyDomain( :collectedh_agence :collectedh )
ObjectPropertyRange( :collectedh_agence :agence )
Declaration( ObjectProperty(  :collectedh_contenantdh ) )
ObjectPropertyDomain( :collectedh_contenantdh :collectedh )
ObjectPropertyRange( :collectedh_contenantdh :contenantdh )
Declaration( ObjectProperty(  :collectedh_keepsafedh ) )
ObjectPropertyDomain( :collectedh_keepsafedh :collectedh )
ObjectPropertyRange( :collectedh_keepsafedh :keepsafedh )
```

**Figure 7** Part of the global ontology (see online version for colours)

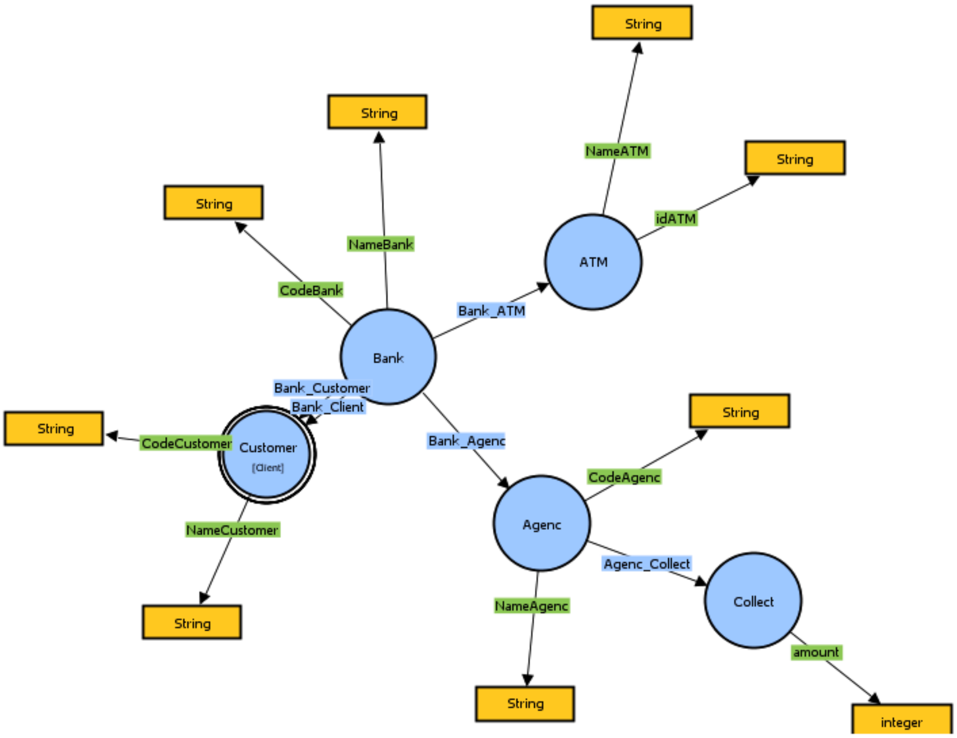


Figure 7 is obtained using the plug-in VOWL shows a part of the resulting global ontology obtained by our tool.

For example, to obtain the names of ATM who had 'BP' Bank, it is sufficient to apply an assertion of the 'Bank-ATM' Object Property, which ensures 'Junction' between classes. Thus, the SPARQL corresponding query is as follows:

```
Select ?NameATM ?NameBank
FROM <C:/Users/Servers/Glob/GlobalOntologyCashSolution.owl>
WHERE{
    ?NameBank Bank-ATM ?NameATM
    Filter(?NameBank='BP') }
```

To evaluate our alignment system, we compared our approach with the LogMap (Jimenez-Ruiz and Grau, 2011), YAM ++ (Ngo and Bellahsene, 2012) and AROMA (David, 2007) ontology alignment systems. The details of this experiment are given in Table 2.

**Table 2** Experimental results comparing our alignment system with other systems

| *Alignment systems* | *Runtime* | *Number of matches* |
|---|---|---|
| Notre approach | 54 | 1,473 |
| LogMap | 46 | 1,388 |
| YAM ++ | 91 | 1,402 |
| AROMA | 66 | 1,215 |

The analysis of the experimental results shows that our results are better in all the tests except for the execution time of the LogMap system which is better than our approach. It also shows that the use of combination similarity measures gives better results.

## 5 Conclusions and perspectives

The general context of this work is the integration of classical data sources into an ontological database. In order to address this issue, we have proposed a semi-automatic approach wherein human intervention assures the validation of results. This approach begins with a transformation of different classical sources (UML, XML and RDB) to local ontologies (OWL2). Thus, it combines:

a Syntactic similarity measures based on the computation of the distance between the characters describing the concepts.

b Semantics based on the semantic enrichment of local ontologies from WordNet.

c Internal structural based on the detection of common instances between two concepts and external structural between pairs of objects in a hierarchical network (subsumption relation) to identify real correspondences and real isolated elements.

Finally, it merges ontologies based on the result of similarity measures in the previous step and the algebraic approaches of graph transformations to generate the global ontology.

Our approach is far from being thorough, several improvements can be made. In future work, we intend to develop new modules to extract ontologies from other common types of data sources, such as NOSQL and Object databases. Besides, we aim to enhance the performance of the similarity identification module through the use of other information retrieval techniques. Finally, we plan to improve our alignment approach by considering other semantic relationships such as disjunction to discover new semantic correspondences.

## Acknowledgements

This work is within the framework of the 'Big data management and analytics – methods and applications' ongoing research project.

## References

Ahmed, A.K., Yasser, O. and Atef, Z.G. (2016) 'Simultaneous mapping of multi RDB to RDF', *7th International Conference on Computer Science and Information Technology*, 13–14 July.

Alaoui, L., El Hajjamy, O. and Bahaj, M. (2014a) 'Automatic mapping of relational databases to OWL ontology', *International Journal of Engineering Research and Technology (IJERT)*, April, Vol. 3, No. 4, pp.1988–1994.

Alaoui, L., El Hajjamy, O. and Bahaj, M. (2014b) 'RDB2OWL2: schema and data conversion from RDB into OWL2', *International Journal of Engineering Research and Technology (IJERT)*, November, Vol. 3, No. 11, pp.962–971.

Amrouch, S. and Mostefai, S. (2012) 'Un algorithme semi-automatique pour la fusion d'ontologies basé sur la combinaison de stratégies', in *International Conference on Education and e-Learning Innovations*.

Bedini, I., Benjamin, N. and Gardarin, G. (2010) 'Janus: automatic ontology builder from XSD files', *arXiv preprint,* arXiv: 1001.4892.

Bedini, I., Matheus, C. and Patel-Schneider, P.F. (2011) 'Transforming XML schema to OWL using patterns', in *2011 Fifth IEEE International Conference on Semantic Computing (ICSC)*, October.

Breitling, F. (2009) 'A standard transformation from XML to RDF via XSLT', in *Astronomische Nachrichten*, Vol. 330, No.7, pp.755–760.

Cavnar, B.W. and Trenkle, M.J. (1994) 'N-gram-based text categorization', in *3rd Annual Symposium on Document Analysis and Information Retrieval Proceedings of SDAIR-94*, pp.161–175.

David, J. (2007) *AROMA: Une Méthode Pour la Découverte D'alignements Orientés Entre Ontologies à Partir de Règles D'association*, PhD thesis, Université de Nantes.

Doan, A., Madhavan, J., Dhamankar, R., Domingos, P. and Halevy, A. (2003) 'Learning to match ontologies on the semantic web', *The VLDB Journal*, Vol. 12, No. 4, pp.303–319.

El Hajjamy, O., Alaoui, L. and Bahaj, M. (2016) 'Mapping UML to OWL2 Ontology', *Journal of Theoretical and Applied Information Technology (JATIT)*, August, Vol. 90, No. 1, pp.126–143.

El Hajjamy, O., Alaoui, L. and Bahaj, M. (2017) 'XSD2OWL2: automatic mapping from XML schema into OWL2 ontology', *Journal of Theoretical and Applied Information Technology (JATIT)*, April, Vol. 95, No. 8, pp.1781–1796.

Gherabi, N. and Bahaj, M. (2012) 'A new method for mapping UML class into OWL ontology', *Special Issue of International Journal of Computer Applications*, pp.0975–8887 on Software Engineering, Databases and Expert Systems – SEDEXS, September.

Huang, J.Y., Lange, C. and Auer, S. (2015) 'Streaming transformation of XML to RDF using XPathbased mappings', *Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS 2015*, Vienna, Austria, 15–17 September.

Jimenez-Ruiz, E. and Grau, B.C. (2011) 'LogMap: logic-based and scalable ontology matching', *ISWC 2011*.

Khan, A.H. and Porres, I. (2015) 'Consistency of uml class, object and state chart diagrams using ontology reasoners', *J. Vis. Lang. Comput.*, Vol. 26, pp.42–65.

Khiat, A. and Benaissa, M. (2015) 'A new instance-based approach for ontology alignment', *International Journal on Semantic Web and Information Systems (IJSWIS)*, Vol. 11, No. 3, pp.1–19.

Klein, M. and Fensel, D. (2001) 'Ontology versioning on the semantic web', in *The First Semantic Web Working Symposium*, Stanford, CA.

Kramer, T., Marks, B., Schlenoff, C., Balakirsky, B., Kootbally, Z. and Pietromartire, A. (2015) *Software Tools for XML to OWL Translation*, National Institute of Standards and Technology (U.S.), Engineering Laboratory, Intelligent Systems Division, June.

Leacock, C. and Chodorow, M. (1998) 'Combining local context and WordNet similarity for word sense identification', in Fellbaum, C. (Ed.): *WordNet: An Electronic Lexical Database*, Vol. 49, No. 2, pp.265–283, MIT Press.

Levenshtein, V.I. (1966) 'Binary codes capable of correcting deletions, insertions and reversals', *Sov. Phys. Dokl.*, Vol. 6, pp.707–710.

Li, G., Luo, Z. and Shao, J. (2010) 'Multi-mapping-based ontology merging system design', *2nd International Conference Advanced Computer Control (ICACC)*, June.

Lin, D. (1998) 'An information-theoretic definition of similarity', in *Proceedings of the Fifteenth International Conference on Machine Learning (ICML'98)*, MorganKaufmann, Madison, WI.

Ling, H. and Zhou, S. (2013) 'Mapping relational databases into OWL ontology', *International Journal of Engineering and Technology*, December, Vol. 5, No. 6, pp.4735–4740.

Löwe, M. (1993) 'Algebraic approach to single-pushout graph transformation', *Theoretical Computer Science*, March, Vol. 109, No.1, p.181.

Maedche, A. and Staab, S. (2002) 'Measuring similarity between ontologies', in *Proceedings of the European Conference on Knowledge Acquisition and Management-EKAW-2002*, LNCS/LNAI 2473, Madrid, Spain, Springer, 1–4 October, p.251.

Mahfoudh, M., Forestier, G. and Hassenforder, M. (2016) 'A benchmark for ontologies merging assessment', *International Conference on Knowledge Science, Engineering and Management*, October.

Ngo, D.H. and Bellahsene, Z. (2012) 'YAM++ results for OAEI 2012', in *Proceedings of the 7th International Workshop on Ontology Matching collocated with the 11th International Semantic Web Conference (ISWC 2012)*, Boston, USA.

Ngo, D.H., Bellahsene, Z. and Coletta, R. (2012) 'A flexible system for ontology matching', in *IS Olympics: Information Systems in a Diverse World*, Springer, pp.79–94.

Nguyen, V.T., Sallaberry, C. and Gaio, M. (2013) 'Mesure de la similarité entre termes et labels de concepts ontologiques', *CORIA 2013 Conférence en Recherche d'Information et Applications*.

Noy, N.F. and Muzen, N.A. (2000) *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*, Stanford University.

Raunich, S. and Rahm, E. (2011) 'ATOM: Automatic target-driven ontology merging', *IEEE 27th International Conference Data Engineering (ICDE)*, May.

Schadd, F.C. and Roos, N. (2012) 'Coupling of Wordnet entries for ontology mapping using virtual documents', in *Proc. of the 7th Conf. on Ontology Matching*, pp.25–36.

Slimani, T., Yaghlane, B. and Mellouli, K. (2007) 'Une extension de mesure de similarité entre les concepts d'une ontologie', in *4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications*.

Smadja, F., McKeown, K.R. and Hatzivassiloglou, V. (1996) 'Translating collocations for bilingual lexicons: a statistical approach', *Computational Linguistics*, Vol. 22, No. 1, pp.1–38.

Stumne, G. and Maedche, A. (2001) 'FCA-MERGE: bottom-up merging of ontologies', the *17th International Joint Conference on Artificial Intelligence*, August, Vol. 1, pp.225–230.

Wang, P. (2011) 'Lily results on SEALS platform for OAEI 2011', in *Proc. of the 6th Intern. Workshop on Ontology Matching*, pp.156–162.

Winkler, W.E (2006) *Overview of Record Linkage and Current Research Directions*, in Research Report Series, RRS.

Wu, Z. and Palmer, M. (1994) 'Verb semantics and lexical selection', in *Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics*, pp.133–138.

Zedlitz, J., Jorke, J. and Luttenberger, N. (2011) 'From UML to OWL2', *Knowledge Technology: Third Knowledge Technology Week*, KTW 2011, Kajang, Malaysia, 18–22 July.