Research Journal of Applied Sciences, Engineering and Technology 7(13): 2747-2755, 2014
DOI:10.19026/rjaset.7.596
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2014 Maxwell Scientific Publication Corp.
Submitted: September 16, 2013 Accepted: September 28, 2013 Published: April 05, 2014

Research Article

Ontology Mapping for a New Database Integration Model Using an Ontology-driven Mediated Warehousing Approach

[1, 2]Ali Ahmed, [1]Hafizullah Amin Hashim, [1]Faisal Alsamt, [1]Naomie Salim, [2]Khalid Ahmed Ibrahim and [1]Ibrahim Almahy
[1]Soft Computing Research Group, Faculty of Computing, Universiti Teknologi Malaysia, 81310 Skudai, Malaysia
[2]Faculty of Engineering, Karary University, 12304, Khartoum, Sudan

Abstract: Ontology mapping is a technique that has become very useful for matching semantics between ontologies or schemas that were designed independently of each other. The main goal of the ontology mapping is to enable interoperability between applications in distributed information systems based on heterogeneous ontologies. To achieve this goal it is necessary to formally define mapping rules between local data sources and ontologies and the notion of a mapping between ontologies. In this study, the authors proposed a new mapping approach, so that the ontologies have to be linked to actual information sources in order to support the integration process. In this approach, first, for each incorporated information source, a local ontology is generated to describe its semantics as well as the resulting mappings between the source and the local ontology, then the local ontologies are mapped to a global ontology using the mapping rule.

Keywords: Data warehousing, database integration, global ontology, local ontology, ontology mapping

INTRODUCTION

Database integration is a multistep process of finding similar entities in two or more databases to create a non-redundant, unified view of all databases. Heterogeneous data integration is one of the major challenges in the database field. There are two principal techniques to integrate heterogeneous databases, namely the materialized approach (data warehousing) and the virtual approach (mediation).
The materialized technique involves three steps:

- Extracting data from multiple data sources
- Transforming them to be compatible with the global schema defined in the data warehouse
- Loading them into a single integrated system

The advantage of this technique is the reduction of query processing time, network bottlenecks, or the source's unavailability (Bakhtouchi *et al*., 2009). However, as the data in the warehouse is not regularly updated, the results of the queries might be retrieved from an outdated pool of data.

To resolve the data heterogeneity problems that will occur in the database integration process, ontology is proposed to homogenize data and their relationships via a formal machine-understandable language. In a data integration system, ontology is used as the global schema to reconcile the heterogeneities between different data sources (Shvaiko and Euzenat, 2005).

Currently, there are many approaches and tools to deal with database to ontology mapping. They can be classified into two main categories: approaches for creating a new ontology from a database and approaches for mapping a database to an already existing ontology. For our architecture, we suppose that the local ontology does not exist and may be created from the information source (Ghawi and Cullot, 2007). Figure 1 illustrates the correspondences between database components (table, column, constraint) and ontological components (concept, property).

In this study, a new mapping approach is proposed so that the ontologies have to be linked to actual information in order to support the integration process. For each incorporated information source, a local ontology is generated to describe its semantics as well as the resulting mappings between the source and the local ontology. Then the local ontologies are mapped to a global ontology using the mapping rule. The mapping approach proposed here will be used as mapping rule for the heterogeneous and distributed information source components shown in our previous framework in Fig. 2.

Corresponding Author: Ali Ahmed, Soft Computing Research Group, Faculty of Computing, Universiti Teknologi Malaysia, 81310 Skudai, Malaysia

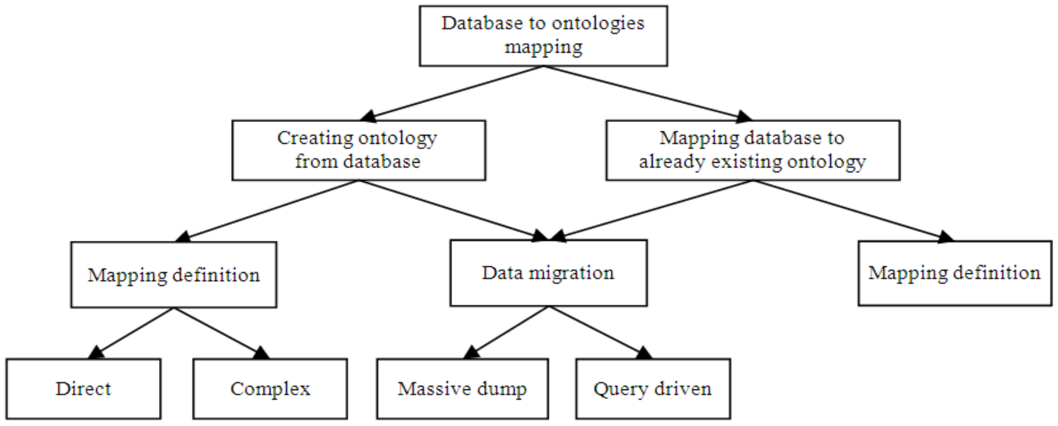
Database to ontologies mapping
Creating ontology from database
Mapping database to already existing ontology
Mapping definition
Data migration
Mapping definition
Direct
Complex
Massive dump
Query driven

Fig. 1: Classification of database-to-ontology mapping approaches

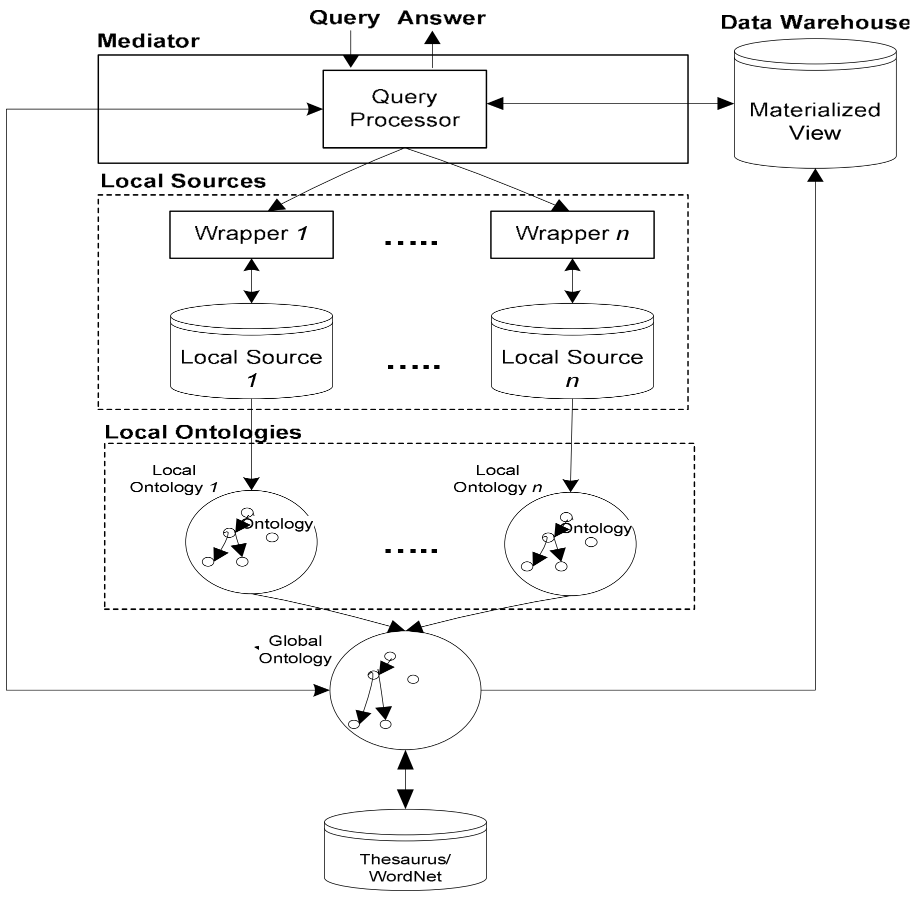

Fig. 2: The proposed database integration using an ontology-driven mediated-warehousing framework

LITERATURE REVIEW

Ontology mapping is a technique that has become very useful for matching semantics between ontologies or schemas that were designed independently of each other. When ontologies are mapped, applications can query data from a multitude of data sources transparently; applications can treat each data source the same irrespective of their differing underlying representations. Ontology mapping is done by analysing various properties of ontologies, such as syntax, semantics and structure, in order to deduce alternate semantics that may apply to other ontologies and therefore create a mapping (Godugula and Engels, 2008).

By this mapping, different ontologies can be automatically matched to each other in order to determine similar information with different structure or syntax. There are many studies on ontology mapping, such as Kalfoglou and Schorlemmer (2003), Noy (2004) and Shvaiko and Euzenat (2005). For instance, Noy and Musen (2001) showed that the results of machine-oriented mapping can produce approximately 75% accuracy.

According to Noy (2004), there are two distinct approaches for ontology mappings, which include the use of shared ontologies and the heuristic and machine learning approaches. The shared ontology is where a group of application developers discuss the meta-ontology, which is the high level ontology that can be used for a class of systems. When this high level ontology is defined, the developers can extend and adapt it for their applications. This approach assumes that there is enough information in the first ontology to make it worthwhile and the concern of each application developers is accounted for. Therefore, this situation is limited because it depends on the collaboration of many developers, which is difficult in many software organizations. The second approach is proposed for situations when a common ontology cannot be used and attempts through heuristic and machine learning to find commonalities between distinct ontologies (Godugula and Engels, 2008).

There are many ontology mapping methods in the literature. For instance, Castano and De Antonellis (2001) proposed a unification method (ARTEMIS method) for global schema construction. Their method is aimed at solving the problem of independent data stores and databases that are built on independent schemas and it was based on a theoretical framework where metadata can be used to automate various aspects of data integration.

In addition, Do and Rahm (2002) proposed the COMA method, which is a system for combining different ontology mapping approaches based on the situation or source schemas. They found that using a single matching approach will not always yield the best results, so combining multiple approaches can lead to better outcomes. Generally, there are two ways of combining matching approaches, which are hybrid and composite. The hybrid uses a combination of data types to match within the execution of an algorithm, while the composite separately executes different algorithms and then analyses the results of these executions.

Furthermore, Noy and Musen (2001) proposed the Anchor-PROMPT method, which is a traditional ontology mapper that attempts to automatically find semantically similar terms between two ontologies. This method takes a set of related terms pairs (anchors) from the source ontologies. These anchors are identified apriori; either the user enters them or the system generates them automatically. Based on these anchors, the method produces a set of new pairs of semantically close terms and it traverses the paths between the anchors in the corresponding ontologies. A path follows the link between classes defined by the hierarchical relations or by slots and their domains and ranges. Then, the terms are compared along with these paths to find similar terms.

Ehrig and Staab (2004) proposed another method, which is known as Quick Ontology Mapping (QOM). This method emphasizes the speed over the accuracy of results. The argument is that the loss in accuracy is quite marginal and the gain in efficiency is tremendous. This makes the QOM a more feasible method for the real-world practical applications when the ontology schemas can often become quite large.

Moreover, Giunchiglia *et al*. (2005) presented another ontology mapping method, which is known as S-Match. This method is proposed based on semantic integration of independently constructed schemas. In S-Match, the schema should be converted into a tree structure either automatically or manually. In these trees, a node is associated with a number and a label. The numbers are the unique identifiers for a node. Then, the “C” notation is used for concepts of nodes and labels. The output of this method will be a matching with different levels of strength between all possible concepts that are found in the input schemas.

PROPOSED MAPPING APPROACH

Using our previous proposed framework, users can query heterogeneous and distributed information sources simultaneously and combine the obtained results in order to gain information that may not be available directly, i.e., the user has the illusion that he queries a unique source. In order to bridge the gap of heterogeneity between information sources, ontologies are used to describe the semantics of the information sources and to make their contents explicit. The ontologies have to be linked to actual information in order to support the integration process. This is done via mappings between each information source and its ontology. For each incorporated information source, a local ontology is generated to describe its semantics as well as the resulting mappings between the source and the local ontology. Then the local ontologies are mapped to a global ontology using the mapping rule. The global ontology describes the semantics of the whole domain of interest. Users’ queries are submitted to the query processor or analyser that analyses the queries and decomposes them into sub-queries which are redelivered to the relevant data provider services. Data providers consists of database, local and global ontology and the mapping rule which is represented implicitly in the lowest part of our previous framework shown in Fig. 2, in more details. Figure 3 gives the description of each data provider services.

Construction and mapping of local ontology: In this study the authors applied four possible scenarios to

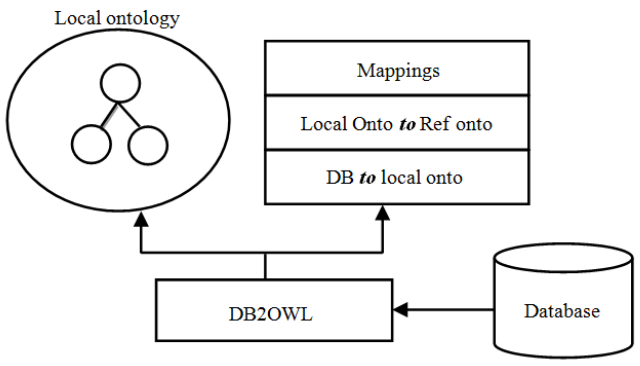

Fig. 3: The architecture of the data provider service

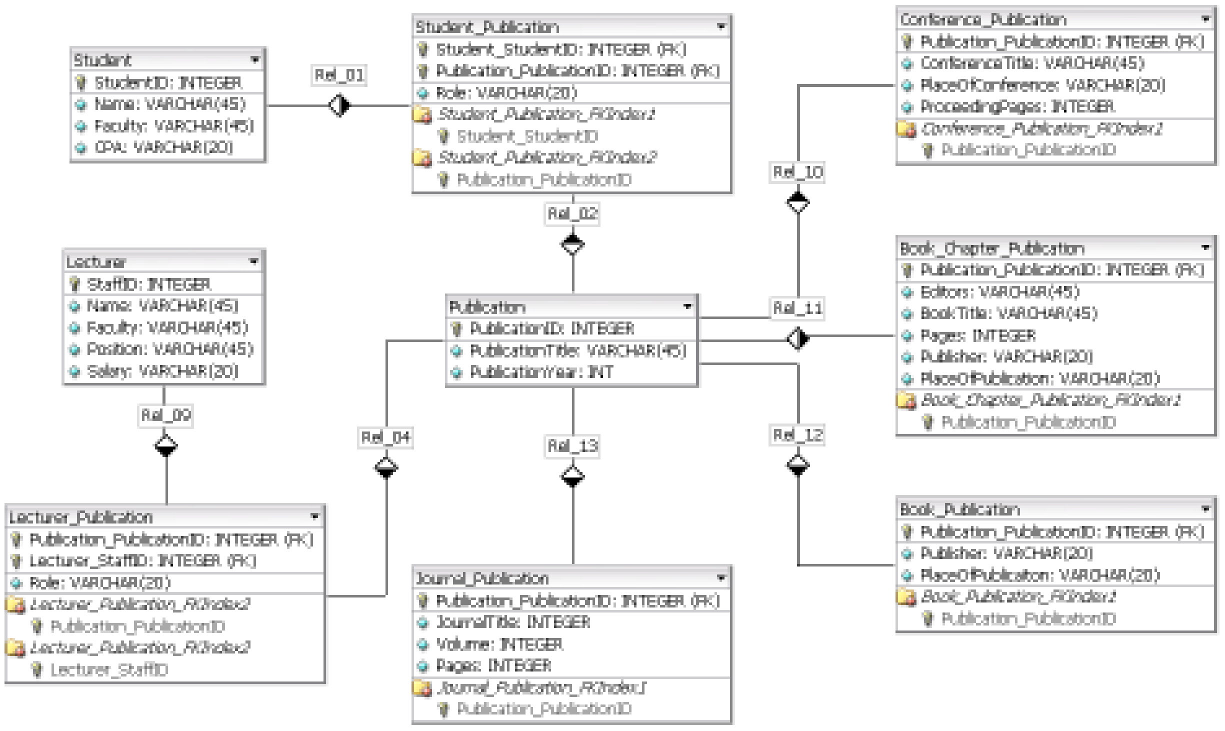
Student
StudentID: INTEGER
Name: VARCHAR(45)
Faculty: VARCHAR(45)
CPA: VARCHAR(20)
Rel_01
Student_Publication
Student_StudentID: INTEGER (FK)
Publication_PublicationID: INTEGER (FK)
Role: VARCHAR(20)
Student_Publication_FKIndex1
Student_StudentID
Student_Publication_FKIndex2
Publication_PublicationID
Conference_Publication
Publication_PublicationID: INTEGER (FK)
ConferenceTitle: VARCHAR(45)
PlaceOfConference: VARCHAR(20)
ProceedingPages: INTEGER
Conference_Publication_FKIndex1
Publication_PublicationID
Rel_10
Rel_02
Lecturer
StaffID: INTEGER
Name: VARCHAR(45)
Faculty: VARCHAR(45)
Position: VARCHAR(45)
Salary: VARCHAR(20)
Publication
PublicationID: INTEGER
PublicationTitle: VARCHAR(45)
PublicationYear: INT
Rel_11
Book_Chapter_Publication
Publication_PublicationID: INTEGER (FK)
Editors: VARCHAR(45)
BookTitle: VARCHAR(45)
Pages: INTEGER
Publisher: VARCHAR(20)
PlaceOfPublication: VARCHAR(20)
Book_Chapter_Publication_FKIndex1
Publication_PublicationID
Rel_09
Rel_04
Rel_13
Rel_12
Lecturer_Publication
Publication_PublicationID: INTEGER (FK)
Lecturer_StaffID: INTEGER (FK)
Role: VARCHAR(20)
Lecturer_Publication_FKIndex2
Publication_PublicationID
Lecturer_Publication_FKIndex2
Lecturer_StaffID
Journal_Publication
Publication_PublicationID: INTEGER (FK)
JournalTitle: INTEGER
Volume: INTEGER
Pages: INTEGER
Journal_Publication_FKIndex1
Publication_PublicationID
Book_Publication
Publication_PublicationID: INTEGER (FK)
Publisher: VARCHAR(20)
PlaceOfPublicaiton: VARCHAR(20)
Book_Publication_FKIndex1
Publication_PublicationID

Fig. 4: Database schema diagram

Student (Student ID, Name, Faculty$_i$, CPA)
Lecturer (Staff ID, Name, Faculty$_i$, Position, Salary)
Publication (Publication ID, Publication Title, Publication Year)
Book Publication (Publication ID, Publisher, Place of Publication)
Book Chapter Publication (Publication ID, Editors, Book Title, Publisher, Place of Publication, Pages)
Conference Publication (Publication ID, Conference Title, Place of Conference, Proceeding Pages)
Journal Publication (Publication ID, Journal Title, Volume, Pages)
Lecturer Publication (Staff ID, Publication ID, Role)
Student Publication (Student ID, Publication ID, Role)

Fig. 5: Database local scheme

illustrate the mapping rule of local database to local ontology. These scenarios are based on element of database and ontology. This includes RIC, object properties and special OWL properties of ontology. The publication database scheme diagram shown in Fig. 4 and 5 is used as an example to explain the mapping through this study.

First scenario: When a Table T is use only to relate two other tables T1, T2 in a many-to-many relationship, it can be divided into two disjointed subsets of columns A1, A2, each participating in a referential constraint with T1 and T2, respectively:

RIC (T) = {ric1, ric2}
ric1 (T, A1, T1, P (T1))
ric2 (T, A2, T2, P (T2))

Therefore, all T columns are foreign keys and they are primaries as well because their combination uniquely defines the rows of T. For example; col (T) = F (T) = P (T), so col (T) = PF (T).

Second scenario: This scenario occurs when a table T is related to more than one table (T2..Tn) by a referential integrity constraint whose local attributes are also primary keys. For example:

∃ ric ∃ RIC (T), LA (ric) = P (T), in other words:
ric (T, P (T), T1, P (T1))
RIC (T) = {ric1, ric2}
ric1 = (T, P (T), T1, P (T1))
ric2 = (T, P (T), T2, P (T2))

Therefore, all primary keys of T are foreign keys because they participate in a referential integrity constraint: P_ (T) = ∅.

Third scenario: This scenario occurs when a table T is related to another table T1 by a referential integrity constraint whose local attributes are also primary keys. For example:

∃ ric ∃ RIC(T), LA (ric) = P (T), in other words:
ric (T, P (T), T1, P (T1))

All primary keys of T are foreign keys because they participate in a referential integrity constraint: P_ (T) = ∅.

Fourth scenario: This scenario occurs when table T has not participated in a referential integrity constraint. Therefore, no foreign key is listed as a local attribute:

col (T) = {A, P (T)}

This scenario is the default scenario, it occurs when none of the previous scenarios occur. When these different scenarios are detected in the database, the mapping process can use them to appropriately map database components to suitable ontology components. Table 1 shows the summarization of these scenarios and condition of mapping process.

Local database to local ontology mapping algorithm: This algorithm is used to develop new ontology from the local database. It starts by mapping the tables to ontology concepts/classes and then mapping the columns to ontology properties. The mapping process should follow the conditions shown in Table 1. The mapping process consists of the following steps.

Table 1: Scenario and condition of mapping process

Scenario	Condition
Scenario 1	Col (T) = PF (T) and |RIC (T)| = 2
Scenario 2	∃ ric ∃ RIC (T), LA (ric1..n) = P (T)
Scenario 3	∃ ric ∃ RIC (T), LA (ric) = P (T)
Scenario 4	Col (T) = {A, P (T)}, F (T) = ∅

Table-class mapping:

- The database tables in scenario 4 are mapped to OWL classes.
- The tables in scenario 3 are mapped to subclasses of those classes corresponding to their related tables. For example:
  If T is in scenario 3 then there is a referential integrity constraint:

  ric ∈ RIC (T) where ric (T, P (T), T1, P (T1))

  so T is mapped to a subclass of the class corresponding to T1.
- The tables in scenario 2 are mapped to more than one subclass of those classes corresponding to their related tables. For example:
  If T is in scenario 2 then there are two referential integrity constraints:

  RIC (T) = {ric1, ric2}
  ric1 (T, A1, T1, P (T1))
  ric2 (T, A2, T2, P (T2))

  if c1, c2 are the two classes corresponding to T, then T is mapped to subclass c1 and c2.
- Each table in scenario 1 is not mapped to class, but the many-to-many relationship that it represents is expressed by object properties. Two object properties are added, one for each class whose corresponding table was related to the current table. For example:
  If T is in scenario 1, then there are two referential constraints:

  ric1 (T, A1, T1, P (T1))
  ric2 (T, A2, T2, P (T2))

  and if c1, c2 are the two classes corresponding to T1, T2 respectively:
  - c1 is assigned to op1 whose range is c2
  - c2 is assigned to op2 whose range is c1
  - op1 and op2 are inverse properties

Column-property mapping:

- For tables that are in scenario 4, we map their referential constraints to object properties whose ranges are classes corresponding to their related tables. For example, if a table T is in Scenario 3 has:

ric (T, A, T1, A1)

if c, c1 are the classes corresponding to T, T1 respectively:

- c is assigned to object property op whose range is c1.
- c1 is assigned to object property op1 whose range is c.
- op is set to functional property (to preserve the original direction of the referential constraint from T to T1.
  Therefore, it will have at most one value for the same instance; this characteristic is obvious because it comes from the uniqueness of key.

- For tables that are in scenario 3 and have other referential constraints than the one used to create the subclass, it is map to object properties as in the previous step.
- For attribute that exists in more than one table, we map the attribute to a data type property. The domain of the data type properties is more than one class. For example:

  If
  col (T1) = {A1, A2, B1, A3} and
  col (T2) = {B1, A2, B3, B2}
  Let's say, c1, c2 are the classes corresponding to T1, T2 respectively
  Then,
  A2 is assigned to data type property A2 whose range is XML schema data type and domain is c1 and c2
  B1 is assigned to data type property B1 whose range is XML schema data type and domain is c1 and c2

- Finally, all columns of tables that are candidate keys are assigned to a data type property. The range of a data type property is the XML schema data type equivalent to the data type of its original column.

Table 2: Implementation of database-ontology algorithm for publication database

Database	Ontology	Explanation
Algorithm 1		
Student (Student ID, Name, Facultyi, CPA)	Student rdf: typeowl: Thing	Table STUDENT consists of the following columns {Student ID, Name, Faculty, CPA} We note that P (T) = {Student ID} and F (T) = {}, So it is in Scenario 4
Lecturer (Staff ID, Name, Facultyi, Position, Salary)	Lecturer rdf: typeowl: Thing	Table LECTURER consists of the following columns {Staff ID, Name, Faculty, Position, Salary} We note that P (T) = {Staff ID} and F (T) = {}, So it is in Scenario 4
Publication (Publication ID, Publication Title, Publication Year)	Publication rdf: typeowl: Thing	Table PUBLICATION consists of the following columns {Publication ID, Publication Title, Publication Year} We note that P (T) = {Publication ID} and F (T) = {} So it is in Scenario 4
Algorithm 2		
Book Publication (Publication ID, Publisher, Place of Publication) Publication (Publication ID, Publication Title, Publication Year)	Book Publication rdfs: sub Class of: Production	Table BOOK PUBLICATION consists of the columns {Publication ID, Publisher, Place of Publication}. We find that P (T) = {Publication ID} and ric ∈ RIC (T) where ric (BOOKPUBLICATION, {Publication ID}, PUBLICATION, {Publication ID}) We note that LA (ric) = {Publication ID} = P (PUBLICATION), therefore BOOKP PUBLICATION is in Scenario 3
Book Chapter Publication (Publication ID, Editors, Book Title, Publisher, Place of Publication, Pages) Publication (Publication ID, Publication Title, Publication Year)	Book Chapter Publication rdfs: sub Class of: Production	Table BOOKCHAPTERPUBLICATION consists of the columns {Publication ID, Editors, Book Title, Publisher, Place of Publication, Pages}. We find that P (T) = {Publication ID} and ric ∈ RIC (T) where ric (BOOKCHAPTERPUBLICATION, {Publication ID}, PUBLICATION, {Publication ID}) We note that LA (ric) = {Publication ID} = P (PUBLICATION), therefore BOOK CHAPTERPUBLICATION is in Scenario 3
Conference Publication (Publication ID, Conference Title, Place of Conference, Proceeding Pages) Publication (Publication ID, Publication Title, Publication Year)	Conference Publication rdfs: sub Class of: Production	Table CONFERENCEPUBLICATION consists of the columns {Publication ID, Conference Title, Place of Conference, Proceeding Pages}. We find that P (T) = {Publication ID} and ric ∈ RIC (T) where ric (CONFERENCEPUBLICATION, {Publication ID}, PUBLICATION, {Publication ID}) We note that LA (ric) = {Publication ID} = P (PUBLICATION), therefore CONFERENCEPUBLICATION is in Scenario 3
Conference publication (Publication ID, Journal Title, Volume, Pages) Publication (Publication ID, Publication Title, Publication Year)	Journal Publication rdfs: sub Class of: Production	Table JOURNALPUBLICATION consists of the columns {Publication ID, Journal Title, Volume, Pages}. We find that P (T) = {Publication ID} and ric ∈ RIC (T) where ric (JOURNALPUBLICATION, {Publication ID}, PUBLICATION, {Publication ID}) We note that LA (ric) = {Publication ID} = P (PUBLICATION), therefore JOURNALPUBLICATION is in Scenario 3

Part of the implementation (first two algorithms) of the above steps for publication database examples is shown in Table 2.

Construction of global ontology: Global ontology plays an important role as a shared vocabulary. The shared vocabulary contains fundamental terms of specific domains. It also acts as an entry point to the query submitted by the user. In other words, it circumvents the need to search for information from local ontologies, one by one. Global ontology is generated based on matching and merging local ontologies.

There are three steps to create global ontologies from local ontologies (Cruz and Xiao, 2005):

- Matching classes and properties between local ontologies and the global ontology
- Merging classes and properties from local ontology S'1
- Class generalization

The process starts by copying all classes, their properties and values from local ontologies into the global ontology. Only the classes, properties and values that are not available in the global ontology are copied. If the class, property and values are already in the global ontology, they will be merged with the global ontology instead. In other words, identical classes, properties and values are combined to be one class, one property and one value respectively in the global ontology. Finally, the global ontology is checked with the Thesaurus or Word Net to find a compatible synonym or hierarchy for certain concepts/classes: for example, Lecturer class from local ontology S'1, Student from local ontology S'2 and Person from Global ontology. The Thesaurus is used to determine generalization of those classes. The result shows that Lecturer and Student are subclasses of Person.

To briefly show how the matching and merging activities are performed, we introduce an initial global ontology as described in Fig. 6. Table 3 shows triples of the global ontology. Below are the features of the global ontology:

- Two classes, namely Person and Faculty
- Two object properties:
  - WorkAt (domain: Person, range: Faculty)
  - StudyAt (domain: Person, range: Faculty)
- Three data type properties
  - Has Hobby (domain: Person, range: "xsd: string")
  - Has DOB (domain: Person, range: "xsd: date")

Matching and merging local ontologies to global ontology: In this study, we used OWL-DL to represent local and global ontology matching and merging.

Table 3: The triples generated in initial global ontology

Instance	Relationship	Value
Person	WorkAt	Faculty
Person	StudyAt	Faculty
Person	HasHobby	"xsd:string"
Publications	HasDOB	"xsd:year"
Faculty	HasAddress	"xsd:string"

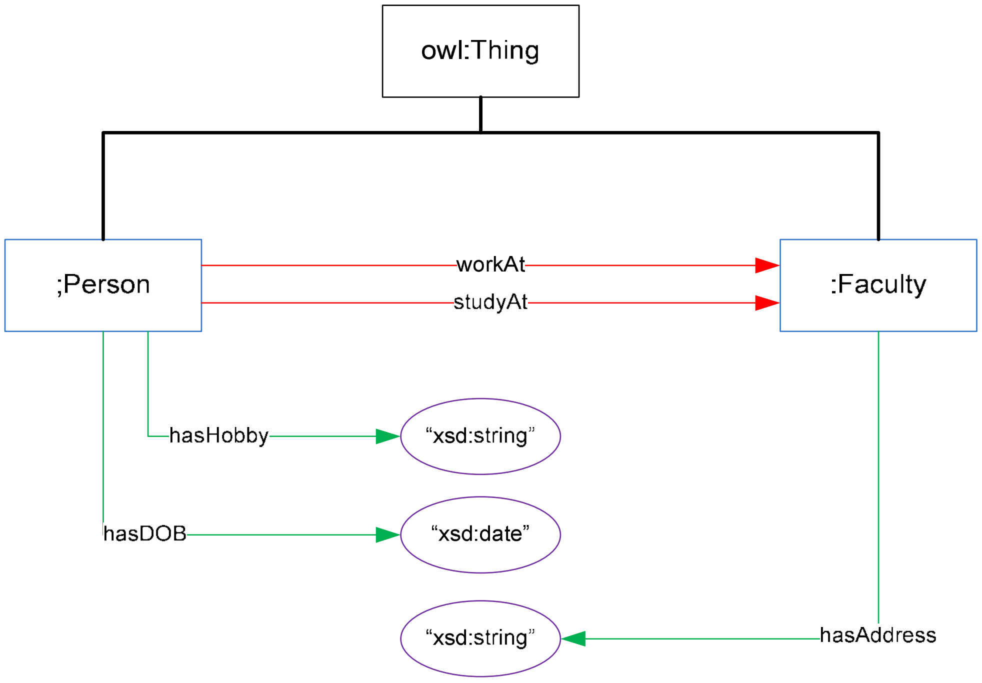

Fig. 6: The initial global ontology

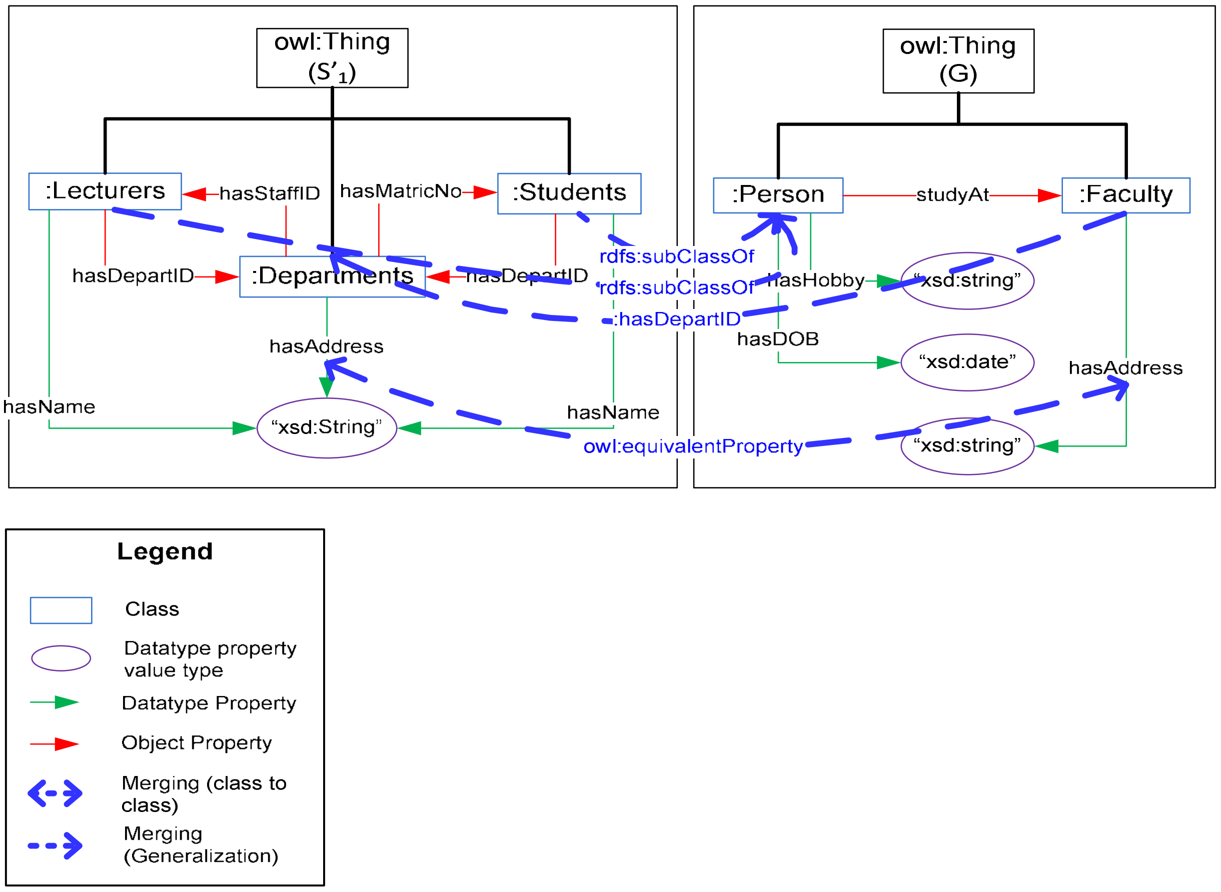

Fig. 7: Process of matching and merging local ontology S1 to global ontology

OWL-DL is based on Description Logic (DL) that contains a rich and highly expressive vocabulary for modelling ontology (Baader, 2003). OWL ontology is represented by three fundamental resources; concept (in OWL named class), role (in OWL named properties) and individuals (in OWL named instances). OWL is also enriched with restriction resources. A concept restriction can be created using a Boolean restriction (AND, OR, NOT), cardinality restriction (min, max), or local restriction (someValue, allValue, hasValue). OWL provides global property restrictions such as transitive property, inverse property, symmetric property, functional property, equivalent property, equivalent class and equivalent individual.

In order to describe the mechanism of our approach, the process begins with matching and merging local ontology S'1 to initial global ontology G, as shown in Fig. 7. The processes of matching and merging are as follows:

- **Matching classes and properties between local ontology S_1 and global ontology:**
  - Department class is copied into target ontology
  - Three Object Properties, namely hasDepartID, hasMatricNo and hasStaffID are copied into target ontology
  - The additional domain named Faculty class is added to hasDepartID
  - One Data type Property, namely hasName is copied into target ontology
- **Merging classes and properties from local ontology S'_1:**
  - No class is merged
  - One Data type Property, namely hasAddress from local ontology and target ontology are combined using owl: equivalent Property restriction
- **Class generalization:** This step uses Word Net to search for the hyponym of Person. If Word Net states that Lecturers and Students are hyponyms of Person, then the Lecturers and Students classes of local ontology will be classified as subclasses of the Person class of the target ontology

CONCLUSION AND RECOMMENDATIONS

In this study, we proposed a mapping approach for mapping the local database sources to the local ontologies. In the local scheme, for each incorporated information source, a local ontology was generated first to allow the ontologies to be linked to actual information resources in order to support the integration process. The local ontologies also mapped to a global

Table 4: Database-ontology mapping notations

Notation	Description
DB	Database
T	Table
col (T)	Set of columns
P (T)	Primary keys
F (T)	Foreign keys
PF (T)	Both primary and foreign keys
P_ (T)	Primary but not foreign keys
_ (T)	Not primary nor foreign keys
op	Object property
$\subseteq$	Is a subset of (every element of A is also an element of B)
$\forall$	For all
α	Alpha
$\in$	Is an element of
β	Beta
$\emptyset$	The empty set (the set with no element)
$\exists$	There is
{}	Set theory (the set with no elements)
Referential Integrity Constraint (RIC)	
(T_1, A_1, T_2, A_2)	T_1 and T_2 are tables $A_1 \subseteq$ col (T_1) - set of columns of the Table 1 $A_2 \subseteq$ col (T_2) - set of columns of the Table 2 Each element of A_1 is a foreign key referenced by an element of A_2 For example: $\forall \alpha_i \in A_1, \exists\ \beta_i \in A_2$ α_i is referenced by β_i So $A_1 \subseteq F\ (T_1)$ and $A_2 \subseteq P\ (T_2)$
LT	Local table
LA	Local attributes
RT	Referential table
RA	Referential attributes For example: LT (ric) = T_1 LA (ric) = A_1 RT (ric) = T_2 RA (ric) = A_2 For a table T, RIC is defined which returns the set of referential integrities whose local table is T For example: RIC: DB $\rightarrow$ P (RIC), RIC (T) = {ric $(T_1, A_1, T_2, A_2) \in$ RIC, LT (ric) = T}

ontology using the mapping rule. Local ontology is extracted from local sources while the global ontology is generated by matching and merging local sources. Apart from using local and global ontology, WordNet is used during matching and merging local ontologies to global ontologies. Further research should focus on the flexibility of our proposed framework by testing it with more types of domains and data sources. In addition, we will conduct further research on the construction of the local ontology algorithm and global ontology algorithm that satisfies all possible situations of database integrations.

Table 4 at the end shows the Database-Ontology mapping Notations.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of the Ministry of Higher Education (MOHE) and University Teknologi Malaysia. This research is partially funded by University research (4F011) of University Teknologi Malaysia.

REFERENCES

Baader, F., 2003. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge.

Bakhtouchi, A., L. Bellatreche and A. Balla, 2009. Materializing Attributes Annotation: A hybrid approach for databases integration. Retrieved from: http://www.univ-msila. dz/umvfr/ images/stories/ files/ manifestations/seminaires/STIC09/articles/ paper_47.pdf.

Castano, S. and V. De Antonellis, 2001. Global viewing of heterogeneous data sources. IEEE T. Knowl. Data Eng., 13(2): 277-297.

Cruz, I.F. and H. Xiao, 2005. The role of ontologies in data integration. Eng. Intell. Syst. Elect. Eng. Commun., 13(4): 245.

Do, H.H. and E. Rahm, 2002. COMA: A system for flexible combination of schema matching approaches. Proceedings of the 28th International Conference on Very Large Data Bases (VLDB), Endowment.

Ehrig, M. and S. Staab, 2004. QOM-quick Ontology Mapping. In: McIlraith, S.A., D. Plex- ousakis and F. Van Harmelen (Eds.), ISWC 2004. LNCS, Vol. 3298, Springer, Heidelberg, pp: 683-697.

Ghawi, R. and N. Cullot, 2007. Database-to-ontology mapping generation for semantic interoperability. Proceeding of the VDBL'07 Conference, VLDB Endowment, ACM, New York, pp: 1-8.

Giunchiglia, F., P. Shvaiko and M. Yatskevich, 2005. Semantic schema matching. Proceeding of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems 2005: CoopIS, DOA and ODBASE, Springer, 1: 347-365.

Godugula, S. and G. Engels, 2008. Survey of ontology mapping techniques. Software Quality and Assurance. Retrieved from: is.uni-paderborn. de/.../ Comparsion_of_Ontology_Matching_Techniques...

Kalfoglou, Y. and M. Schorlemmer, 2003. Ontology mapping: The state of the art. Knowl. Eng. Rev., 18(1): 1-31.

Noy, N.F., 2004. Semantic integration: A survey of ontology-based approaches. ACM Sigmod Record, 33(4): 65-70.

Noy, N.F. and M.A. Musen, 2001. Anchor-PROMPT: Using non-local context for semantic matching. Proceedings of the Workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI).

Shvaiko, P. and J. Euzenat, 2005. A survey of schema-based matching approaches. J. Data Semant., 4: 146-171.