Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A NEW DATABASE INTEGRATION MODEL USING AN ONTOLOGY-DRIVEN MEDIATED WAREHOUSING APPROACH

[1]HAFIZULLAH AMIN HASHIM, [2]ALI AHMED, [3]NAOMIE SALIM, [4]OH YU SHENG, [5]AHMAD OSMAN, [6]ALEX SIM, [7]ARYATI BAKRI, [8]NOR HAWANIAH ZAKARIA, [9]ROLIANA IBRAHIM, [10]SHAHIR SHAMSIR OMAR

[1,2,3,4,5,6,7,8,9]Faculty of Computing, Universiti Teknology Malaysia, Skudai,Johor Bharu, 81310, Malaysia
[2]faculty of Engineering, Karary University, 12304, Khartoum,Sudan
[10]Faculty of Biosciences and Bioengineering, Universiti Teknology Malaysia, Skudai,Johor Bharu, 81310, Malaysia
E-mail: [1]tringliserida@yahoo.com, [2]alikarary@gmail.com, [3]naomie@utm.my, [4]ysoh@mail.com, [5]ahmedagraa@hotmail.com, [6]alex@utm.my, [7]aryati@utm.my, [8]hawaniah@utm.my, [9]roliana@utm.my, [10]shahirshamsir@gmail.com

ABSTRACT

Database integration technology has been developed for more than 20 years. The difficulties of database integration are the integration of heterogeneous data sources, with respect to the schemas and their data, as well as the query processing time that can take longer than expected. In this study, we present a semantic database integration framework using an integrated mediated and data warehouse approach to search for query words or sentences in a database and determine the accuracy of the search results. This method exploits semantic annotation, which overcomes some of the traditional database integration problems such as syntactic heterogeneity, structural or schematic heterogeneity and semantic heterogeneity. In this approach, semantic annotation is extracted from two types of ontologies, local and global ontology. The former is used to provide semantic annotation of data sources, and the latter is used to provide the shared vocabulary of a particular domain. With the help of domain ontology, the searching process will be more meaningful as it caters for the semantic aspects of a search query. This approach can enhance the efficiency and effectiveness of the search for the desired information.

Keywords: *Semantic, Database Integration, Local Ontology, Global Ontology, WordNet, SPARQL Query, Warehouse, Mediated*

1. 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 big 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: (1) extracting data from multiple data sources, (2) transforming them to be compatible with the global schema defined in the data warehouse, and (3) 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 [1]. 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.

On the contrary, the mediation approach, proposed by Wiederhold [2], allows the data to be constructed directly from the original data sources. Two important components of this technique are the global schema construction and the mappings between the global schema and the local schemas of the data sources. At the user interface level, the query submitted to the global schema is decomposed and rewritten as local queries, addressed to data sources [4]. There are two principal methods for dealing with the mapping process, namely Global-as-View (GaV) and Local-as-View (LaV). In the GaV approach, every entity in the global schema is associated with one or more views over the local schemas of data sources.

Query processing is rather simple, but the evolution of the local source schemas is difficult to cater for. In contrast, the LaV approach permits changes to source schemas without affecting the global schema, because the local schemas are defined as views over the global schema, but the query processing can be very complex [3].

There are three kinds of data heterogeneity problems in data integration: (1) syntactic heterogeneity, which is caused by different languages and data models used for modelling the different data sources, (2) schematic heterogeneity, which is caused by different designed schemas in data sources, and (3) semantic heterogeneity, which is caused by different meanings or interpretations of data in various contexts.

To resolve these data heterogeneity problems, 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. According to Wache [4], three main ontology architectures are used for data integration: (1) a single ontology approach, where all source schemas are directly associated with the global ontology that provides a uniform interface to the user. However, since all sources have nearly the same view on a domain, this approach is susceptible to changes to data sources; (2) a multiple ontology approach, where each data source is described by its own ontology. Since there is no common ontology, local ontologies must be mapped to each other by using inter-ontology mappings; and (3) a hybrid ontology approach, which is a combination of the two former approaches. First, a local ontology is built for each source schema. However, instead of being mapped to other local ontologies, all local ontologies are mapped to a global ontology. New sources can be added easily without modifying the existing mappings and the global ontology. In this paper, we propose a new database integration model using an ontology-driven mediated warehousing approach that emphasizes four issues: (1) local ontology construction, (2) global ontology construction, (3) the mapping process, and (4) query processing. In this proposed approach, each data source has a specific local schema. Typically, these local schemas are inconsistent with each other. For the purpose of information integration, a system user would submit a query based on a global and mediated ontology instead of the multiple local ontologies. The mediated system would, in return, re-formulate the user query into sub-queries that refer directly to the local ontology of the data source. The differences between the proposed approach and the existing approach are the use of two types of ontology in two different parts; as a global ontology and local ontologies. A global ontology is deployed with WordNet and Mediator, whereas local ontologies are deployed with wrappers. The local ontology is generated from local data sources whereas the global ontology is derived from local ontologies by matching and merging processes.

The objective of this study is to present semantic database integration framework using an integrated mediated and data warehouse approach to search for query words or sentences in a database and determine the accuracy of the search results. The proposed approach exploits semantic annotation, which overcomes some of the traditional database integration problems such as syntactic heterogeneity, structural or schematic heterogeneity and semantic heterogeneity.

2. THE PROPOSED ONTOLOGY-DRIVEN MEDIATED-WAREHOUSING FRAMEWORK

Several researches have been performed in database integration over the last two decades. However, the ontology-based database integration approach is relatively new. The goal of database integration is to avoid the user having to search for the intended data from isolated sources and to manually combine the data from different sources [5]. In the area of ontology-based mediated database integration, this research [5] has applied the semantic of the ontology into a mixed sensor network. In this application, the ontology has been implemented to answer queries posted by the user. The role of the ontology is to increase the relevance of the query result and to recover tuples of the virtual relations from the source relations. In his study [6], the semantic of the ontology was applied during data extraction from data source to the warehouse. It was used to group the value of semantically related attributes and transform the data into the warehouse. However, the method is only used for data warehouse integration, contrary to the proposed method, which applies the semantic of the ontology in hybrid database integration. The main motivation for proposing the aforementioned framework is to speed up the query processing time and at the same time to reduce the heterogeneity of ambiguous data sources. Both are challenging issues of data integration, regardless of whether the materialized integration approach or virtual integration approach is used. In order to increase the query processing performance, we used the data warehouse method to avoid the need to search individual databases. But there are doubts pertaining to the accuracy of the intended results. This is due to the weakness of the data warehouse, which is inflexible to changes in the questions that can be asked [7]. Data not extracted into the warehouse cannot be queried conveniently. This weakness is rectified by integrating the mediated method into the proposed framework. The role of the mediated component is to provide a solution to interoperate the heterogeneous data source by transforming both the queries and the data transparently [8]. The process of reducing the heterogeneous data from disparate data sources is performed by applying the semantic component such as ontology. The proposed method applies the ontology in two portions; the data warehouse portion and the mediated portion, as shown in Figure 2. This proposed method is unique due to the introduction of hybrid database integration (mediated and warehouse) together with ontology implementation.

As shown in Figure 1, our proposed framework consists of five main components, namely a mediator with query processing and metadata, a data warehouse, global ontology WordNet mapping, a set of local data sources with wrappers, and local ontologies. The details of each component are described in the following subsections.

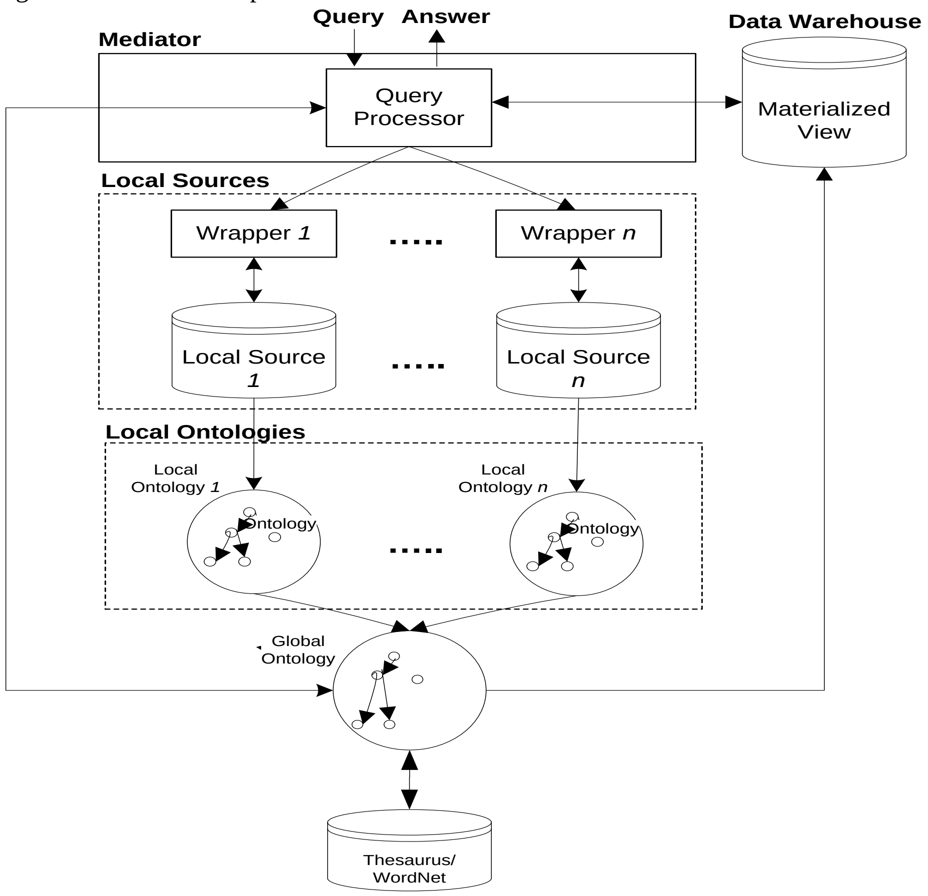


Figure 1: The Proposed Database Integration Using An Ontology-Driven Mediated-Warehousing Framework

2.1 Mediator

The mediator consists of two subcomponents; metadata, and query processing, as shown in Figure 1. The mediator has 3 roles; (1) It acts as a unified platform for query data from all sources of the database, (2) It serves as a shared vocabulary for wrappers by using the global ontology, and (3) It finds synonyms of the search query from global ontology. The Query Processor is used to determine whether the query can be answered by a materialized view or by a virtual view. The query processor receives a query from the user in SPARQL (SPARQL Protocol and RDF Query Language) format. It verifies the query according to the SPARQL syntax. The component of the query is then matched with the global ontology which has been extended using WordNet. The metadata will also contain the mapping of the global ontology to the local ontology, including the ontology of the content of the data warehouse. It then rewrites the query according to the local ontology and passes the resulting sub-queries to its respective wrapper. The result returned by the wrappers will be merged according to the global ontology. In the merging, only distinct results will be returned to users. Interline spacing is set sufficiently to prevent overlapping but without leaving too much space.

2.2 Data Warehouse

The data warehouse has been developed for the decision making process and traditional data analysis during the last decade. The data is collected from different operational data sources, transformed into information and stored in the warehouse so that the user can access the data in a consistent and useful manner [9].

In the proposed framework, the data warehouse is used to store a copy of data from several sources that are frequently accessed by the user, thus enabling those data to be queried through a unified query [7]. The role of the data warehouse in the proposed framework is to speed up the query response time if the intended result is available in the data warehouse. This is especially true for summarized data. The query will then be routed to the warehouse if the content of the warehouse can satisfy the query without using the local databases[7].

2.3 Global Ontology

Global ontology extracts data from local ontologies and extends it by finding the synonyms of the concepts, relationships, and relationships' values through WordNet. Once the synonyms of these attributes are found, it will update into the global ontology. Global ontology supports a high-level view. The user can formulate a query without specific knowledge of the different data sources. The query is then rewritten into a query over the sources through the mediator, based on semantic mapping between the global and local ontologies.

2.4 Local Data Sources

Local data sources consist of 2 subcomponents, namely local sources and wrappers. Local data sources provide the required local results of the query. Each of the local sources has dedicated wrappers and local ontologies. The wrappers have 2 roles; (1) to translate the SPARQL query to Structured Query Language (SQL), (2) to match and merge local ontologies to global ontologies.

In this framework, the GaV approach is followed to represent the mapping between concepts in the global ontology and local ontologies in the data sources. The wrappers will pass the query to the local databases. The query processor will determine if a query can be answered from either the materialized views or virtual views. The query is first addressed to the data warehouse, which caches, summarizes, and allocates frequently requested information. If the query matches the materialized views, the answer will be fetched directly from the data warehouse. Otherwise, the query will be decomposed and rewritten based on the mapping of the concepts between the global ontology and local ontologies. As soon as the query processor obtains the answers back from the data sources or the materialized views, it integrates the answer and returns it to the user.

2.5 Local Ontology

Local ontologies are extracted from local sources. Each local source has its own local ontology. In the proposed framework, local ontologies are then merged onto the global ontology to form a unified ontology, in order to find the semantic of the searched query. The purpose of local ontologies is to cater for the heterogeneity of the sources schemas and their respective data.

2.6 The Role of Local and Global Ontology in the Proposed Framework

The most important activities in database integration are performed in the identification schema-level correspondence and instance-level correspondence or tuple-level correspondence[10]. A schema-level correspondence is identified by mapping and merging database schemas from different operational data sources, whereas an instance-level schema is identified by mapping tuples or records from different operational data sources. The correspondence process is very critical, determining the accuracy of the query result.

Much research has been conducted in semantic correspondence from heterogeneous data sources. For example, in schema-level correspondence, [11] reported that schema mapping takes too much time due to communication between the domain expert and knowledge engineer. For example, in order to identify that mission start time and mission take-off have the same meaning and are located in different data sources, it took a week to get the verification.

The difficulty becomes greater when dealing with identification of instance-level correspondence. This is due to rapid proliferation of records updates compared to a database schema. Furthermore, the records are updated from multiple different data sources. Several names are given to the identification of instance-level correspondence, such as record linkage [12], approximate record matching [13], entity identification [14], merge/purge [15], and instance identification [16]. Research on schema- and instance-level correspondence across heterogeneous data sources has been performed before. There are four techniques to identify schema-level correspondence namely, the linguistic technique[5, 17-19], heuristic formulae [20-24], similarity measures [25], and cluster analysis [3, 26]. The proposed techniques do not use ontology to overcome heterogeneous data sources. Instead, they use a traditional method to detect semantic correspondence. For example, the similarity measure technique computes the similarity index of the database schemas from different sources using a technique developed in the information retrieval [24]. Several methods have also been proposed for identification of instance-level correspondences. A rule-based method is used to determine whether two database records represent the same object. It has also been used for modelling knowledge that was extracted from domain experts. Sometimes this approach becomes a nightmare for the knowledge engineer due to the difficulty of modeling domain expert knowledge using the rule-based method [12, 14, 27, 28]. Apart from this method, classification is also a technique for detecting the instance-level correspondence, such as the logistic regression technique [26] and decision tree technique [14]. A classification technique is used to classify the similarity of two tuples from heterogeneous databases. Neither technique (rule-based or classification) implements ontology as a tool for the identification of schema- and instance-level correspondence across different data sources.

In our proposed framework, a schema-level correspondence is identified by converting database schemas from different data sources to local ontologies. In other words, every data source has its dedicated ontology. The role of a local ontology is to provide a semantic environment for identification of schema-level correspondence. The mapping process begins with two local ontologies. Once the mapping is completed, it will be merged to the initial global ontology. We assume the global ontology already exists. The next local ontology is then mapped and merged to the global ontology. The mapping step only happens at the beginning level (mapping between two local ontologies). The next steps of merging will follow accordingly until the last local ontology. The global ontology plays a significant role in providing a unified semantic corresponding model which will overcome the schema-level and instance-level heterogeneity. For example, schema-level heterogeneity will be rectified by mapping the Concept/Class and Relationship/Property between local ontology S'_1 and local ontology S'_2, whereas instance-level heterogeneity will be rectified by mapping Individual/Instance between local ontology S'_1 and local ontology S'_2. The use of ontology in the proposed framework will overcome the identification of schema- and instance-level correspondence simultaneously rather than separately.

3. CONSTRUCTION OF LOCAL ONTOLOGIES

There are various methods for developing local ontologies. These may be manual or semi-automated approaches. In the manual approach, the ontology is developed from scratch or from the existing ontology. This approach requires more time to complete the ontology as the information is derived from domain experts. The knowledge engineer will then extract and convert the information into ontological form. Conversely, if the ontology is developed using the semi-automated approach, the knowledge engineer will only need to run programs such as D2RQ [29] for the ontology to be automatically built. The manual part is performed by domain experts who have to verify the accuracy of information that was recently modelled into ontological form.

In this paper, we will apply a semi-automated approach for the construction of local ontologies. This approach will then be deployed in the proposed framework. In the proposed framework, the homogeneous database which is relational databases is transformed into local ontologies. There are 16 rules [30]to follow in the construction of local ontologies from relational schema. The rules cater for the degree of relationships types determined by the number of distinct entities participating in the relationship. The relationships types are listed below:

i). Unary relationship mapping
ii). Binary relationship mapping
   a. 1:1 relationship mapping
   b. 1:M relationship mapping
   c. M:N relationship mapping
iii). N-ary or ternary relationship mapping
iv). Multi-valued attributes mapping
v). Specialization mapping
vi). Generalization mapping

In order to ensure that the transformation processes are well-organized, two algorithms, entity and attributes representation and relationship representation [31], are applied. These algorithms are written in Pseudo Code so that they can be implemented in any programming language. The first algorithm demonstrates all possible Class and Datatype Properties generated from Tables and Attributes. The second algorithm demonstrates all seven possible aforementioned conditions. This algorithm is mostly used in generating Classes and Object Properties from Tables and Relationship Constraints. For example, two relational schemas are provided from local source 1 (S_1) and local source 2 (S_2) as shown below:

Local Source	**Database Schema**
S_1	LECTURERS (staffID[PK], name, departID[FK]) DEPARTMENTS (departID[PK], address) STUDENTS (matricNo[PK], name, departID[FK])
S_2	PUBLICATIONS (pubID[PK], title, year) STUDENTS (matricNo[PK], name, pubID[FK])

We will show how the algorithm is used to generate classes and object properties from the database table and relationship constraint of schema S_2

S_2 { PUBLICATIONS(pubID[PK], title, year)
STUDENTS(matricNo[PK], name, pubID[FK])

where 'PK is Primary Key, and 'FK is Foreign Key.

The relational schemas above have a Referential Integrity Constraint (RIC) with a one-to-one relationship. 'pubID' is a foreign key belonging to STUDENTS and refers to the primary key of PUBLICATIONS. Rule_C1 below is executed in order to produce classes of ontology S'_2.

Rule_C1

Class Condition:

$$\exists x, y, z: \left(\begin{array}{l} (a)\ RIC(R_1, A_1, R_2, A_2) \wedge \\ (b)\ PK(x, R_1) \wedge PK(y, R_2) \wedge \\ (c)\ FK(z, R_2, x, R_1) \wedge \\ (d)\ PK(y, R_2) \neq FK(z, R_2) \end{array} \right)$$

Class Action:

$$\left((1)\ \ class(C_1) \leftarrow R_1 \right)$$

where:
C_1 and C_2 are not already created

Notation:
R: Relation
A: Attribute
^ : AND Operator
≠: Not Equal
⊆: Subset
col: Column
∃: There is
LA: Local Attribute
← *(e.g., R_1 is converted to be a C_1)*

Four class conditions should be followed during execution:

i). RIC must exist between two tables as a first condition to use Rule_C1. RIC tables are represented by quadruplet RIC(PUBLICATION, *{pubID, title, year}*, STUDENTS, *{matricNo, name, pubID)* where PUBLICATION *is* Table 1 and STUDENTS is Table 2, $A_1 \subseteq$ col (Table 1) and $A_2 \subseteq$ col (Table 2). In Rule_C1, Relation PUBLICATION is related to Relation STUDENTS by RIC whose ∃ LA(STUDENTS) = PK(PUBLICATIONS) and,
ii). *pubID* is the primary key of relation PUBLICATIONS and *matricNo* is the primary key of relation STUDENTS, and

iii). *pubID* is a foreign key in relation to STUDENTS and refers to primary key *pubID* of relation PUBLICATIONS, and

iv). *matricNo* is the primary key of relation STUDENTS and is not equal to the *pubID* foreign key belonging to relation STUDENTS. If the primary key and foreign key are the same element, then STUDENTS will be a subclass of PUBLICATIONS.

Once rules and the algorithm are executed, two classes are generated, namely PUBLICATIONS and STUDENTS corresponding to table PUBLICATIONS and table STUDENTS. The generation of both classes occurs if C_1 and C_2 do not yet exist in the ontology.

According to the algorithm [30], all entities/tables are transformed to classes and all attributes including primary key, except for composites belonging to each table, are transformed to Datatype Properties. Conversely, if a table contains composite attributes, it will be transformed to Class 'X' and one Object Property with the domain of the Class (transform from table contains the composite attributes) and the range Class 'X' is created. There is a problem in the algorithm [30]. Supposedly a primary key is transformed to Object Property instead of Datatype Property due to its capability to impose an Inverse Property restriction. In other words, to be an Inverse Property, the relationship must be a type of Object Property [32]. Therefore, in Function 2 of the algorithm [30], we modify the function as below. This algorithm will be used in the execution of Rule_P1

```
Input: Table name T and (primary key or alternative key A)
Output: Object Property (OP) creation
Steps:
        If (Pattr.name: Type or Alattr. name:
           Type ∈ Attr then let OP(has-A)
           cardinality =1, domain (T_2), range
           (T);
        Else
           Call function4;
        EndIf;
End;
```

Figure 2: Algorithm for Creating Primary Key or Alternative Key

Notation:
T: Table
Pattr: Primary Attributes
Alattr: Alternative Attributes

∈: Element of

Immediately after the class rule and class algorithm are executed, the property rule and relationship algorithm are executed to transform normal attributes and relationships of the tables into Datatype Properties and also possibly property restrictions such as Inverse Property, Functional Property, Transitive Property, etc. Below is the rule condition and subsequent action:

Rule_P1

Property Condition:

(a) $RIC(R_1, A_1, R_2, A_2) \wedge$
(b) $PK(x, R_1) \wedge PK(y, R_2) \wedge$
(c) $FK(z, R_2, x, R_1) \wedge$
(d) $\forall p: (Attr(p, R_1) \wedge NonFK(p, R_1) \wedge NonPK(p,R_1))$ v
(e) $\forall q: (Attr(q, R_2) \wedge NonFK(q, R_2) \wedge NonPK(q, R_2))$

Property Action:

(1) $OP(x, C_2, C_1)$
(2) $OP(y, C_1, C_2)$
(3) $IP(x, y)$
(4) $DP(p, C_1, type(p))$
(5) $DP(q, C_2, type(q))$

where:
x, y, p, and z are not already created,
class(C_1) « (R_1),
class(C_2) « (R_2)

Notation:
∀: All
V: OR Operator
NonPK: Non Primary Key
Attr: Attributes
OP: Object Property
IP: Inverse Property
DP: Datatype Property
C: Class
«: Asserted (e.g., R_1 is generated as relationship of C_1)

There are five property conditions demonstrated in Rule_P1. The first three conditions are similar to Rule_C1, whereas the rest of the conditions are clarified as below using the same local source S_2:

i). For all p which are attributes of R_1 but not a foreign key and not a primary key of relations R_1. p = {*pubID*, *title*, *year*} ∩

{*pubID, title, year*} ∩ {title, year}. Therefore p = {title, year}, and

ii). For all q which are attributes of R_2 but not a foreign key and not a primary key of relations R_2. q = {*matricNo, name, pubID*} ∩ {*matricNo, name*} ∩ {*name*}. Therefore q = {*name*}.

iii). Once the abovementioned conditions are complied with, Rule_P1 is then executed based on algorithm 7.1 and algorithm 7.2 [30]. According to Function 1of algorithm 7.1, normal attributes are transformed to Datatype Property, whereas an Object Property is generated base on the algorithm in Figure 2. Three Datatype Properties are generated, namely *hasTitle*, *hasName*, and *hasYear*. The domain of *hasName* and *hasYear* are Students Class and the range is an xml string value and xml year value respectively, whereas the domain of *hasTitle* is Publication Class with xml string value.

iv). A primary key of Publications; *pubID* and a primary key of Students; *matricNo* are transformed to Object Properties based on the algorithm in Figure 2. The output of the transformation is two Object Properties, namely *hasPubID* and *hasMatricNo*. Both Object Properties are executed as Inverse Property restrictions due to a one-to-one relationship using algorithm 7.2. The domain of *hasPubID* is Students and the range is Publications whereas the domain of *hasMatricNo* is Publications and the range is Students.

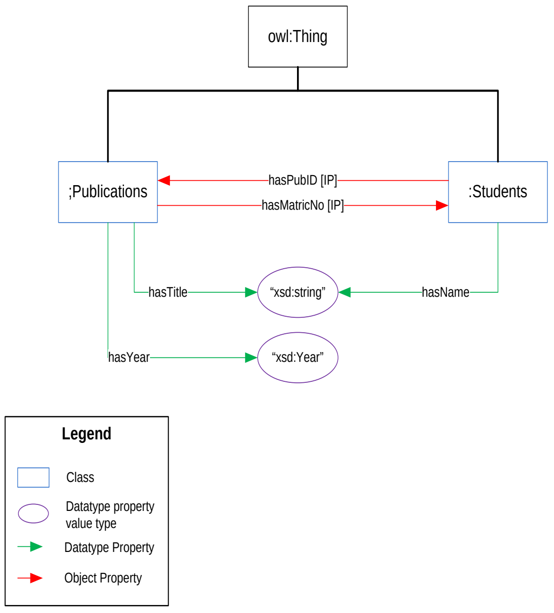


Fig. 3: Local Ontology S'$_2$

v). After the entire tables and attributes of the relational schemas of local source S_2 are transformed to local ontology S'_2, it is ready to be merged into the global ontology or target ontology. Figure 3 depicts the local ontology S'_2 and Table 1 shows the triples of local source S_2

Below are the following features of local ontology S'_2:

i). Two Classes, namely Students and Departments

ii). Two Object Properties:

a. *hasPubID* (domain: Students, range: Publication)

b. *hasMatricNo* (domain: Departments, range: Students

iii). Three Datatype Properties:

a. *hasName* (domain: Students, range: "xsd:string"

b. *hasTitle* (domain: Publications, range: "xsd:string"

c. *hasYear* (domain: Publications, range: "xsd:year"

Table 1: The Triples Generated From Local Schema S_2

Instance	Relationship	Value
Publications	hasMatricNo	Students
Students	hasPublicationID	Publication
Publication	hasTitle	"xsd:string"
Publications	hasYear	"xsd:year
Students	hasName	"xsd:string"

In order to show how the matching of functions and merging of local ontologies to the global ontology is done, we generate another local ontology S'_1 that was extracted from local schema S_1. Below is the local schema of S_1

S_1 { LECTURERS(staffID[PK], name, departID[FK])
DEPARTMENTS(departID[PK], address)
STUDENTS(matricNo[PK], name, departID[FK])

The process of transforming local schema S_1 to local ontology S'_1 is the same. Figure 4 and Table 2 show the local ontology S'_1 and the triples respectively with the following features:

i). Three Classes, namely Lecturers, Students, and Departments

ii). Three Object Properties:

a. *hasDepartID* (domain: Lecturers and Students, range: Departments)

   b. *hasStaffID* (domain: Departments, range: Lecturers)
   c. *hasMatricNo* (domain: Departments, range: Students)

iii). Two Datatype Properties:
   a. *hasName* (domain: Lecturers and Students, range: “xsd:string”)
   b. *hasAddress* (domain: Departments, range: “xsd:string”)

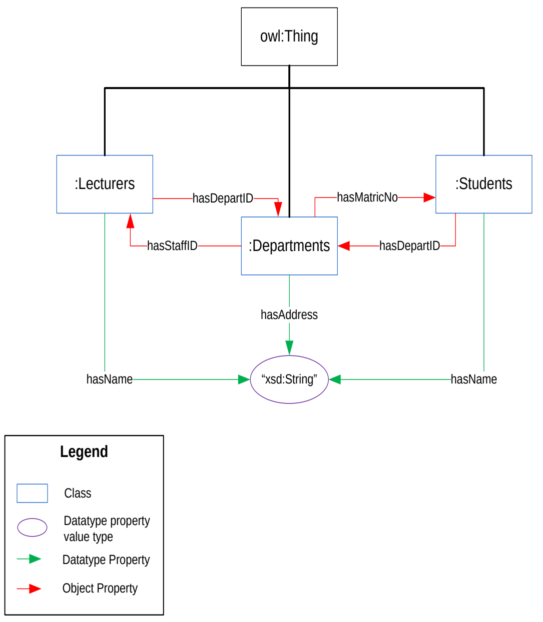


Figure 4: Local Ontology S’$_1$

Table 2: The Triples Generated From Local Schema S$_1$

Instance	Relationship	Value
Lecturers	hasDepartID	Departments
Departments	hasStaffID	Lecturers
Departments	hasMatricNo	Students
Student	hasDepartID	Departments
Lecturers	hasName	“xsd:string”
Students	hasName	“xsd:string”

4. 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 3 steps to create global ontologies from local ontologies [33]:

i). Matching classes and properties between local ontologies and the global ontology.
ii). Merging classes and properties from local ontology S’$_1$
iii). 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 WordNet 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.

The question arises how to limit the scope of the global ontology and how to classify whether the information remains as local ontology or should merge into the global ontology. The partitioning and granularity of the local and global ontologies are identified through three possibilities [34] where matching and merging ontology can be based on:

i). Organizational unit
ii). Topic area
iii). Specific and general information

The organizational unit corresponds to user groups. Anything required by user groups should be located in local ontology. A Topic Area possibility means that each user group is responsible for certain topics. Therefore, it should be located in local ontology. A local ontology stores more specific information whereas a global ontology stores more general information such as upper ontology.

In our approach, none of the aforementioned possibilities are followed. The local ontologies are matched and merged into the global ontology one by one based on algorithms proposed by Ellakwa et al. [35]. The algorithms consist of two parts, matching and merging. In the matching algorithm, local ontologies are matched to global ontology through 5 matching processes; equality string method, substring method, prefixes method, suffix method, and WordNet method. All 5 methods are used in 3 iterations; the first iteration for extracting matched concepts, the second iteration for extracting matched properties, and the third iteration for extracting matched values of the matched properties. Conversely, the merging algorithm demonstrates the process of merging classes, properties, values, and hierarchies into the global ontology.

To briefly show how the matching and merging activities are performed, we introduce an initial global ontology as described in Figure 5. Table 3 shows triples of the global ontology. Below are the features of the global ontology:

i). Two classes, namely Person and Faculty
ii). Two object properties:
   a. *workAt* (domain:Person, range:Faculty)
   b. *studyAt* (domain: Person, range: Faculty)
iii). Three datatype properties
   a. *hasHobby* (domain:Person, range:"xsd:string")
   b. *hasDOB* (domain:Person, range:"xsd:date")

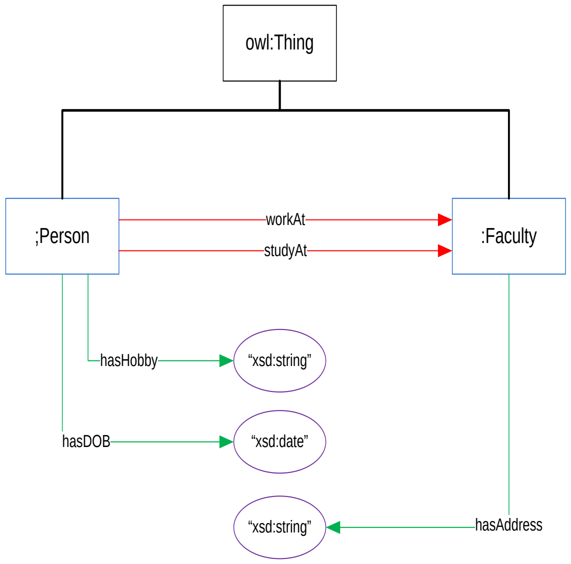


Figure 5: The Initial Global Ontology

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"

5. MATCHING AND MERGING LOCAL ONTOLOGIES TO GLOBAL ONTOLOGY

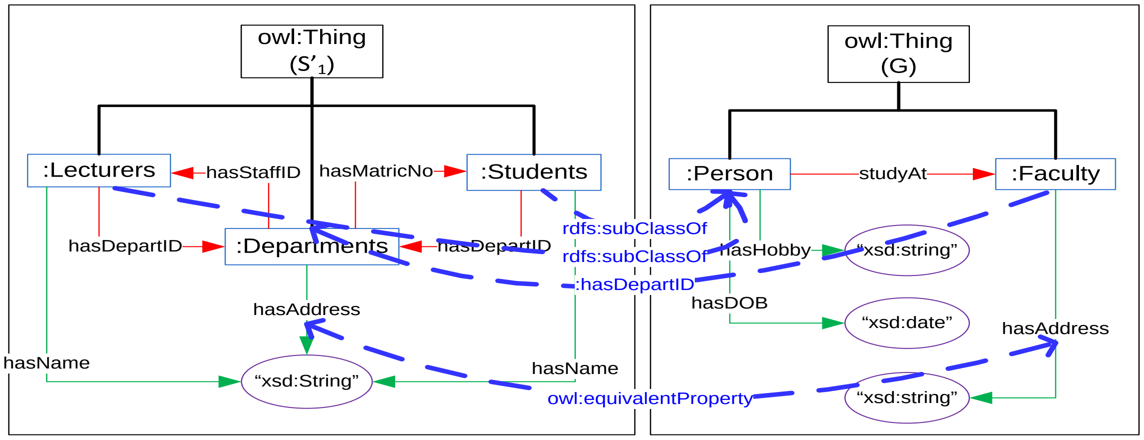
In this paper, we used OWL-DL to represent local and global ontology matching and merging. OWL-DL is based on Description Logic (DL) that contains a rich and highly expressive vocabulary for modeling ontology [36]. OWL ontology is represented by 3 fundamental resources; concept (in OWL named class), 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 this paper, we used equivalent class and equivalent property as an example to prove the proposed framework**.** Local ontologies are matched and merged onto global ontology one by one. Contrariwise, Susan Ellakwa et al. [35] match local-to-local ontology prior to merging to the global ontology. Based on our approach, maintenance of the global ontology is more efficient than Susan's approach. This is due to the global ontology construction process used in our approach. For example, in our approach, the matching and merging of local ontology is directly onto global, but in Ellakwa et al.'s approach [30], local ontologies have to be matched amongst them first before merging onto the global ontology.

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 Figure 6. The processes of matching and merging are as follows:

(i). 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 Datatype Property, namely *hasName* is copied into target ontology



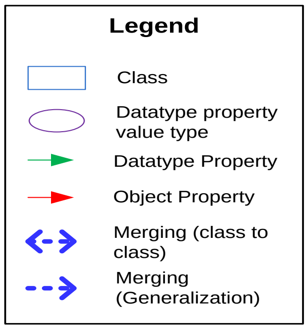


Figure 6: Process of Matching and Merging Local Ontology S_1 to Global ontology

(ii). Merging classes and properties from local ontology S'$_1$:

- No class is merged
- One Datatype Property, namely *hasAddress* from local ontology and target ontology are combined using *owl:equivalentProperty* restriction.

(iii). Class generalization:

- This step uses WordNet to search for the hyponym of Person. If WordNet 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.

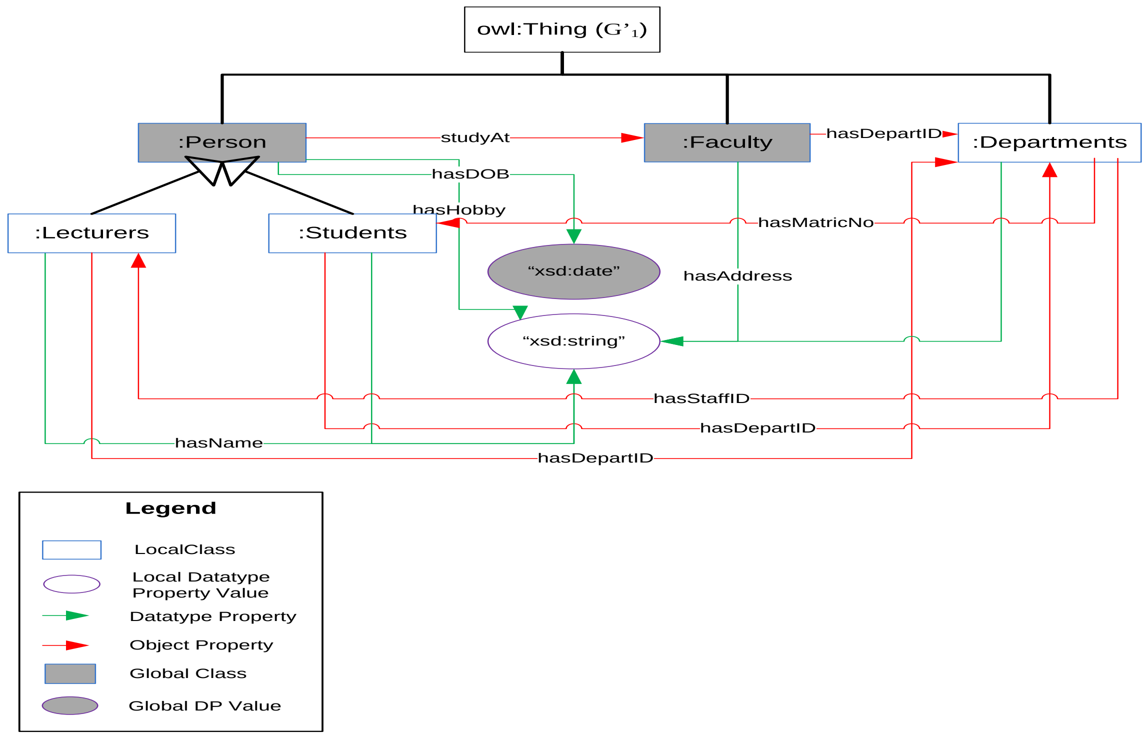


Figure. 7: The New Global Ontology (G'$_1$)

Figure 7 shows the result of the construction of the global ontology G'$_1$. We note that the global ontology G'$_1$ consists of the following new features:

(i). One Department Class is added. This class is the domain for two Object Properties, namely *hasStaffID* and *hasMatricNo* and one Datatype Property, namely *hasAddress*. The class is also the range for *hasDepartID*.

(ii). Lecturers and Students Class are subclasses of Person Class. Both subclasses will inherit Person Class attributes and relationships (*hasDOB, hasHobby, and workAt*). Therefore, when reasoning is 'on', the inferred triples are generated. For example, Figure 8 shows an example of the instance Lecturers (Lecturer_01) and its properties. Five triples of Lecturer_01 were generated: three are inferred triples from the initial global ontology (shown by a red line) and the rest are from local source S$_1$ (shown by a black line). Table 4 shows the triples of global ontology G'$_1$. In the 'type of triple' column, 'asserted' means the triple is generated manually or already exists while 'inferred' means the triple is generated when executing a reasoner or inference engine. Figure 8 shows an instance of Lecturers and its relationships in graph form.

Table 4: The Triples Of Global Ontology G'$_1$

Instance	Relationship	Value	Type of Triple
Lecturer_01	hasName	"Naomie Salim"	Asserted
Lecturer_01	hasDOB	"13-08-1966"	Inferred
Lecturer_01	hasHobby	"Writing"	Inferred
Lecturer_01	workAt	FSCIS	Inferred
Lecturer_01	hasDepartID	IS001	Asserted

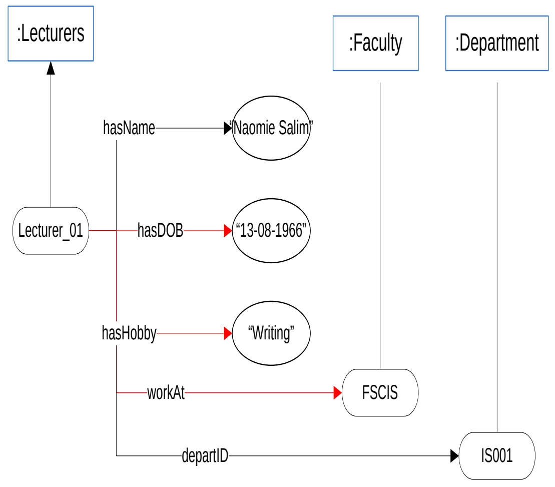


Figure 8: Example of Instance Lecturers

Once the local ontology S'$_1$ is merged into global ontology G'$_1$, the next local ontology S'$_2$ is ready to be matched and merged to global ontology G'$_1$. Figure 9 illustrates the process of updating global ontology G'$_1$ by matching and merging local ontology S'$_2$. The activities of matching and merging are as below:

(i). Matching classes and properties between local ontology S'$_2$ and global ontology G'$_1$

- Publication class is copied into target ontology (global ontology G'$_1$).
- One Object Property, namely *hasPubID* is copied into global ontology G'$_1$
- Two Datatype properties, namely *hasTitle* and *hasYear* are copied into global ontology G'$_1$.
- Note that the Students class *hasMatricNo* and *hasName* properties already existing in G'$_1$, so it will be combined and merged into G'$_1$.

(ii). Merging classes and properties from local ontology S'$_2$

- Students classes in the local and target ontology are combined into one class in the target ontology using *owl:equivalentClass* restriction
- Datatype Property *hasName* in the local and target ontology are combined into one Datatype Property using the *owl:equivalentProperty* restriction
- Object Property *hasMatricNo* in the local and target ontology are combined into one Object Property using the *owl:equivalentProperty* restriction

(iii). Class generalization

- No class generalized.

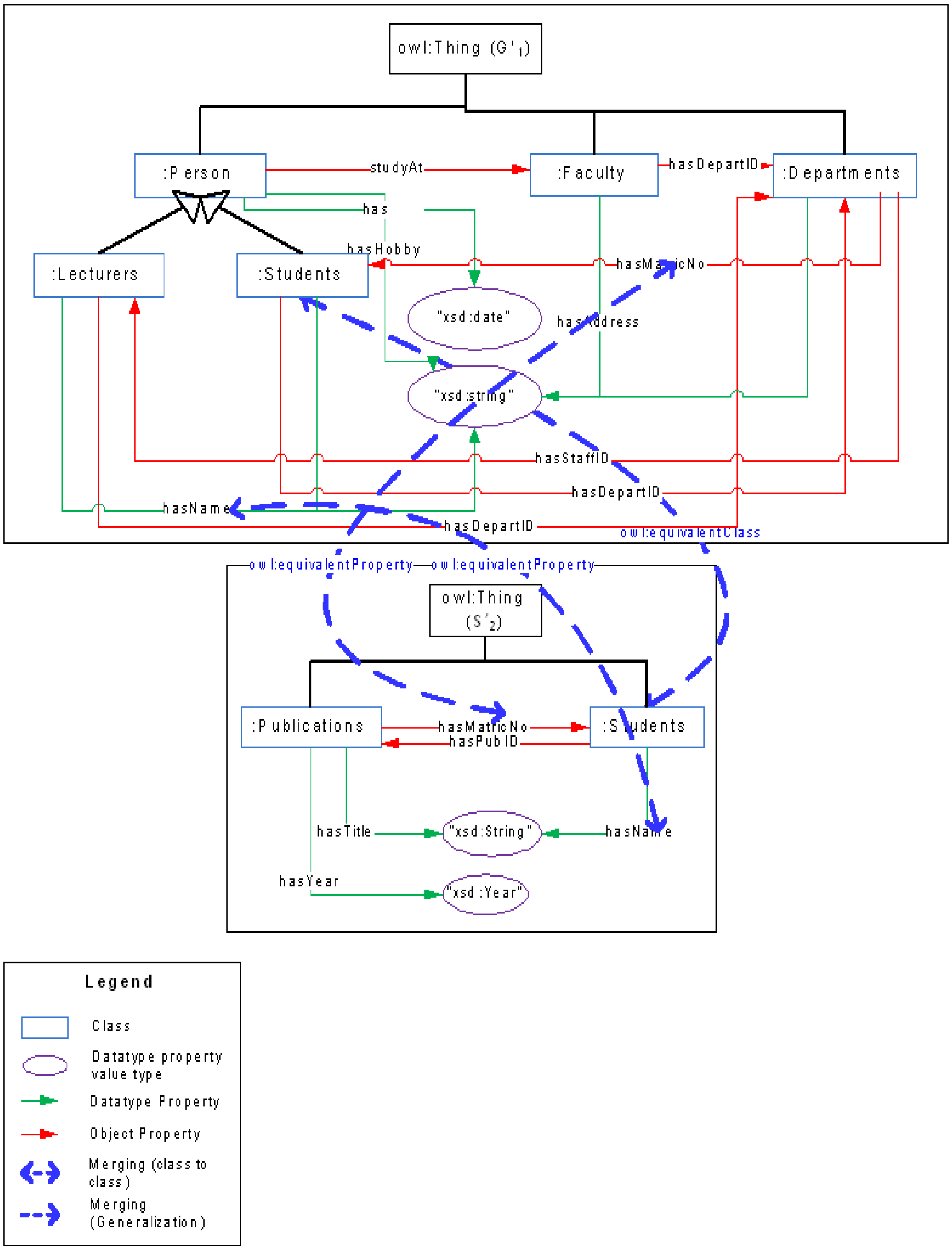


Figure 9: Process of Matching and Merging Local Ontology S_2 into Global Ontology G'_1

Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

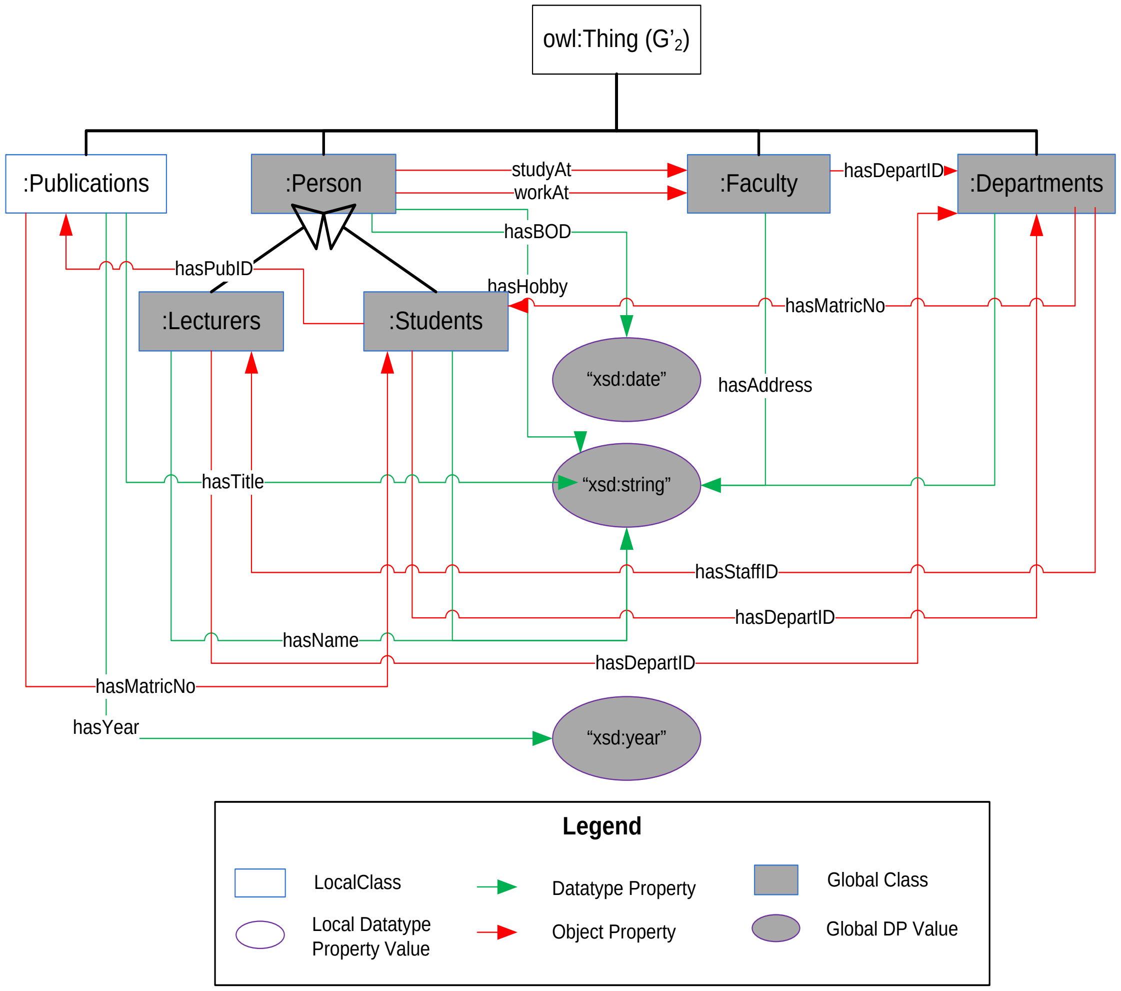


Figure 10: The Updated Global Ontology G'_2 from G'_1

From the diagram in Figure 10, we note that the global ontology G'_2 consists of the new features:

i). The Publication class is added. This is the domain for one Object Property, namely *hasMatricNo* and two Datatype Properties, namely *hasTitle* and *hasYear*. The Publication class also ranges for *hasPubID* Object Property.

ii). Table 5 shows the triples of global ontology G'_2, while Figure 11 illustrates an example of an instance of Students with six triples (three are inferred triples from global ontology G'_2, one is from local ontology S'_2, and two are from local ontology S'_1)

Table 5: The triples of global ontology G'_2

Instance	Relationship	Value	Type of Triple
Student_01	hasName	"Hafizullah Amin"	Asserted
Student_01	hasDOB	"19-06-1983"	Inferred
Student_01	hasHobby	"Riding Superbike"	Inferred
Student_01	studyAt	FSCIS	Inferred
Student_01	hasDepartID	IS001	Asserted
Student_01	hasPubID	ISBN123	Asserted

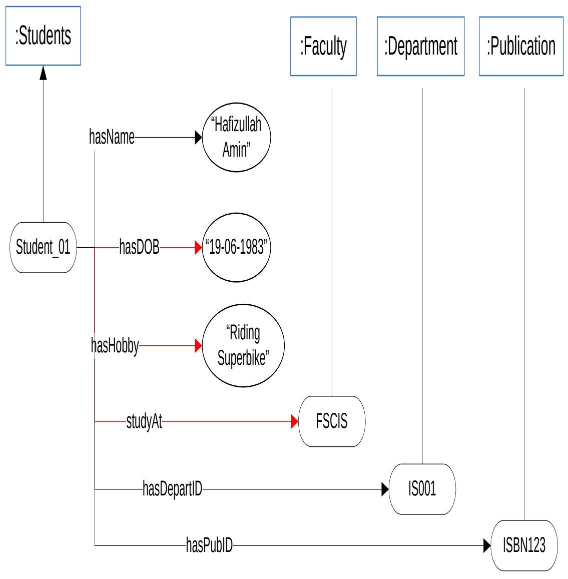


Figure. 11: Example of Instance Students

This section discussed the construction of local ontologies (S'$_1$ and S'$_2$) and global ontology G'$_1$ and G'$_2$. Global ontology G'$_1$ is generated when matching and merging local ontology S'$_1$, whereas global ontology G'$_2$ (final result) is generated once global ontology G'$_1$ is updated with local ontology S'$_2$. This section also discussed the significance of constructing global ontology by proving several examples of instances. The next section will discuss the query mechanism which retrieves relevant information from local sources using a SPARQL query through global ontology.

6. EXAMPLE OF SOURCE IDENTIFICATION IN QUERY PROCESSING

The proposed framework shows that the query inserted by the user is verified by the unified form of ontology which is the global ontology. The verified query is then passed to the mediator which will then pass it to the respective wrapper. The wrapper has to pass it to the local database. Prior to passing the query, the wrapper has to convert the ontology query (SPARQL) to a database query. The database passes back the result to the wrapper, which passes it to the mediator, and the mediator then passes it to the user. This process is clarified by the following example:

"Persons who have written journal ISBN123"
The above statement is an example of a question that might be asked by the user. The framework's input is in the form of words, not the complete statement as stated above. On the application interface, the user is required to enter the Journal number "ISBN123" in order to get the result of the person who wrote the Journal. Once the mediator receives the Journal number "ISBN123", it will be transformed into a SPARQL query. The SPARQL query is based on matching graph patterns. A graph pattern is composed of a Subject, Predicate and Object to form a triple. The triple is used to describe facts. A Subject can be either a Blank Node, or a Unified Resource Identifier (URI). A Predicate, also known as a property or slot, must always be a URI, whereas an Object can be a Blank Node, URI, or Literal [37]. Combining more triples can form more complex graph patterns.

```
PREFIX
global:<http://fsksm.utm.my/global/1.0
SELECT ?x
WHERE
{
  ?x   rdf:type   :Persons.
  ?x   :hasPubID     :P01.
}
```

The query variables will be searched for in the global ontology. If the components are in the data warehouse, then the mediator will search for the result through the data warehouse. If the result exists, then it will be passed back to the mediator, which subsequently passes it to the user. If there is no data related to the query, then the mediator will search for the result in local ontology through global ontology. From the example query, as the word "Journal" does not exist in the global ontology, the mapping between the global ontology and WordNet is executed in order to get the synonyms of 'journal'. The output of WordNet shows that "Journal" is a "Publication". The word 'publication' exists in the global ontology. The word 'journal' will be passed to the mediator, which will search for the query result through the data warehouse. If the data about the 'journal' word still does not exist in the data warehouse, then the mediator will search for it in the local data sources through wrappers.

Based on the SPARQL graph pattern, the result of x is either instances of Lecturers or instances of Students. This is because both classes are subclasses of Person, but in the global ontology no instances are migrated to the global ontology. It only consists of terminology of shared vocabulary. The mediator identifies x as belonging to whom. Once identified, it will pass the x value and variable "ISBN123" to the respective wrapper. In this example, the mediator accepts x values and "ISBN123" from the client, determines the sources needed to answer the x value and "ISBN123", and decomposes the x value and "ISBN123" into sub-queries to wrapper 2 and wrapper 3. This is because Lecturers and Students reside in both local databases S_1 and S_2. The wrapper then transforms the SPARQL query to the SQL format to query the data from the data source. If x is matched, the result is translated back to the SPARQL query by the wrapper. If not, the database returns 'null'. Finally, the wrapper passes the result to the mediator and returns the final answer to the user. Here the result is Students, which is "Hafizullah Amin". The mediator receives the value of x which is "Hafizullah Amin". Finally, the result will be passed to the user.

7. CONCLUSION

In this paper, we proposed a mediated-warehousing data integration framework driven by local and global ontology. The purpose of the proposed framework is to find the most relevant search result by catering for the semantic relationship between data sources and the user's query. This was executed by the local and global ontology. Local ontology is extracted from local sources and global ontology is generated by matching and merging local sources. Apart from using local and global ontology, our framework integrates WordNet as an agent for finding synonyms of words. In our framework, WordNet is used during matching and merging local ontologies to global ontologies and while accepting the result of the query from the user. Prior to decomposing the query into sub-queries, the mediator matches the query to WordNet in order to find any identical meaning before proceeding to wrappers. Further research should focus on the flexibility of the 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.

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.

REFRENCES:

[1] Bakhtouchi, A., L. Bellatreche, and A. Balla, Materializing Attributes Annotation: A hybrid approach for databases integration. 2009.

[2] Wiederhold, G., Mediators in the architecture of future information systems. Computer, 1992. 25(3): p. 38-49.

[3] Zhao, H. and S. Ram, Clustering similar schema elements across heterogeneous databases: a first step in database integration. Advanced Topics in Database Research, 2006. 5: p. 235-256.

[4] Wache, H., et al. Ontology-based integration of information-a survey of existing approaches. in IJCAI-01 workshop: ontologies and information sharing. 2001: Citeseer.

[5] Mirbel, I., Semantic integration of conceptual schemas. Data & Knowledge Engineering, 1997. 21(2): p. 183-195.

[6] Bergamaschi, S., et al., A semantic approach to ETL technologies. Data & Knowledge Engineering, 2011. 70(8): p. 717-731.

[7] Gardner, S.P., Ontologies and semantic data integration. Drug discovery today, 2005. 10(14): p. 1001-1007.

[8] Chirathamjaree, C. and S. Mukviboonchai. The mediated integration architecture for heterogeneous data integration. in TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering. 2002: IEEE.

[9] Nebot, V. and R. Berlanga, Building data warehouses with semantic web data. Decision Support Systems, 2012. 52(4): p. 853-868.

[10] Zhao, H. and S. Ram, Combining schema and instance information for integrating heterogeneous data sources. Data & Knowledge Engineering, 2007. 61(2): p. 281-303.

[11] Clifton, C., E. Housman, and A. Rosenthal, Experience with a combined approach to attribute-matching across heterogeneous databases, in Data Mining and Reverse Engineering. 1998, Springer. p. 428-451.

[12] Fellegi, I.P. and A.B. Sunter, A theory for record linkage. Journal of the American Statistical Association, 1969. 64(328): p. 1183-1210.

[13] Verykios, V.S., A.K. Elmagarmid, and E.N. Houstis, Automating the approximate

record-matching process. Information sciences, 2000. 126(1): p. 83-98.

[14] Ganesh, M., J. Srivastava, and T. Richardson. Mining entity-identification rules for database integration. in Proceedings of the Second International Conference on Data Mining and Knowledge Discovery. 1996.

[15] Hernández, M.A. and S.J. Stolfo, Real-world data is dirty: Data cleansing and the merge/purge problem. Data mining and knowledge discovery, 1998. 2(1): p. 9-37.

[16] Si, A., C.C. Ying, and D. McLeod. On using historical update information for instance identification in federated databases. in Cooperative Information Systems, 1996. Proceedings., First IFCIS International Conference on. 1996: IEEE.

[17] Bright, M., A.R. Hurson, and S. Pakzad, Automated resolution of semantic heterogeneity in multidatabases. ACM Transactions on Database Systems (TODS), 1994. 19(2): p. 212-253.

[18] Ambrosio, A.P., E. Métais, and J.-N. Meunier, The linguistic level: Contribution for conceptual design, view integration, reuse and documentation. Data & Knowledge Engineering, 1997. 21(2): p. 111-129.

[19] Johannesson, P., Supporting schema integration by linguistic instruments. Data & Knowledge Engineering, 1997. 21(2): p. 165-182.

[20] Hayne, S. and S. Ram. Multi-user view integration system (MUVIS): An expert system for view integration. in Data Engineering, 1990. Proceedings. Sixth International Conference on. 1990: IEEE.

[21] Madhavan, J., P.A. Bernstein, and E. Rahm. Generic schema matching with cupid. in Proceedings of the International Conference on Very Large Data Bases. 2001.

[22] Mangiameli, P., S.K. Chen, and D. West, A comparison of SOM neural network and hierarchical clustering methods. European Journal of Operational Research, 1996. 93(2): p. 402-417.

[23] Palopoli, L., et al., Intensional and extensional integration and abstraction of heterogeneous databases. Data & Knowledge Engineering, 2000. 35(3): p. 201-237.

[24] Rodríguez, M.A., M.J. Egenhofer, and R.D. Rugg, Assessing semantic similarities among geospatial feature class definitions, in Interoperating Geographic Information Systems. 1999, Springer. p. 189-202.

[25] S.S. Benkley, J.F. Fandozzi, E.M. Housman, G.M. Woodhouse, Data element tool-based analysis (DELTA), The MITRE Corporation, Bedford, MA, Technical Report, MTR 95B0000147, 1995.

[26] Zhao, H. and S. Ram, Clustering schema elements for semantic integration of heterogeneous data sources. Journal of Database Management (JDM), 2004. 15(4): p. 89-106.

[27] Segev, A. and A. Chatterjee, A framework for object matching in federated databases and its implementation. International Journal of Cooperative Information Systems, 1996. 5(01): p. 73-99.

[28] J.C. Pinheiro, D.X. Sun, Methods for linking and mining massive heterogeneous databases, in: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, 1998, pp. 309–313. (ref2)

[29] http://d2rq.org/

[30] Alalwan, N., H. Zedan, and F. Siewe. Generating OWL ontology for database integration. in Advances in Semantic Processing, 2009. SEMAPRO'09. Third International Conference on. 2009: IEEE.

[31] Alalwan, N.A., Ontological approach for database integration. 2011.

[32] www.w3.org/TR/owl-ref/

[33] Cruz, I.F. and H. Xiao, The role of ontologies in data integration. Engineering intelligent systems for electrical engineering and communications, 2005. 13(4): p. 245.

[34] Uschold, M. Creating, integrating and maintaining local and global ontologies. in Proceedings of the First Workshop on Ontology Learning (OL-2000) in conjunction with the 14th European Conference on Artificial Intelligence (ECAI-2000). 2000: Citeseer.

[35] Susan F Ellakwa, E.-s.E.-a.a.P.E.-k., Integrated Ontology for Agricultural Domain. International Journal of Computer Applications 54(2):46-53, 2012.

[36] Baader, F., The description logic handbook: theory, implementation, and applications. 2003: Cambridge: Cambridge University Press.

[37] Fabian, B., et al., Access control for semantic data federations in industrial product-lifecycle management. Computers in Industry, 2012.