Storage of Ontology as Database and Representation of Existing Database as Ontology

Ivy Jane Thomas[1]
Amal Jyothi College of Engineering, Kerala
ivyjanethomas@gmail.com

Fr. Rubin Thottupuram
Amal Jyothi College of Engineering, Kerala
thottupuram@gmail.com

Abstract: Semantic Web, the next generation Web, stands out from the traditional Web by incorporating meaning to the information that is accessible to the users. Hence in effect a Web of Data is formed, represented through Ontologies. Most of the data in the traditional Web is being stored in the form of relational databases. Hence for the common man to start with ontologies, this paper tries to propose a mechanism that efficiently stores an entire ontology as a database. To move along with this transition from the traditional Web to Semantic Web, all data must be converted to a form that complies to the Semantic Web concepts. Hence this paper also proposes a mechanism to represent databases as ontology by determining the relationship between the various database components. The system proposed also tries to integrate knowledge of various databases and existing ontologies leading to a global ontology that can be used in various contexts.

Keywords: *database, ontology, storage, retrieval, integration*

*****

I. INTRODUCTION

The World Wide Web, was a system introduced to access information through a set of interlinked hypertext documents. It provided mechanisms to link users with sources of information residing worldwide. The World Wide Web enabled the spread of information over the Internet through an easy-to-use and flexible format.[1]. Searching the internet using WWW search engines mainly involved the lexical form of the words. Indexes were built by these engines based on the words of the document and also their location in that document. This reduced the accuracy of the searches being made as any document that contained the word would be displayed. It never considered the context that the user really wants in which the word was being used.

The Semantic Web concept introduced by Tim Berners-Lee, mainly was focused on making the Web meaningful by adding semantics to the data, programs and other web resources and thus making it a knowledge-centric Web. This will lead to more accurate searching of information. To make it possible, the system must be much more machine processable and intelligence must be incorporated to make searches more accurate and efficient. Hence Semantic Web tries to point out that the meaning of the information is more important than the textual structure of it. Semantic Web gave birth to what is known as `Web of Data' that makes automated interlinking of information from various sources possible increasing the relevancy of the information that is being displayed to the user.

Semantic Web represents knowledge in the form of Ontology that is composed of classes, properties describing classes,etc. Organizations of this era store their related information in the form of databases comprising a collection of tables. In order to evolve from the World Wide Web to Semantic Web there must be a mapping between the ontology concept and the relational database concept. It must be possible to decide what components of the database will become what of the ontology. Relational Databases, one of the main components of business computing, is a collection of data represented in tabular form where each table is composed of rows and columns. It is called as `Relational` because a table is referred as a `relation' that is composed of a collection of objects of the same type. Now, since the traditional Web is slowly evolving into Semantic Web, that stores data in the form of Ontology, hence in order for the transition to move on and to make business computing also more effective, there must be some mechanism to convert the relational databases to Ontology form.

II. LITERATURE SURVEY

Semantic Web[4], a concept introduced by Tim Berners Lee, is not a separate Web but an extension of the current one, in which information is given well defined meaning, better enabling computers and people to work in cooperation.[2]. Most of the information today simply focuses on how the data is presented to the user. It cannot be processed by the machines. Hence the users manually would have to browse through several pages and integrated the required knowledge. One of the major features is that the reuse of the already integrated knowledge, facilitating easier inter-operability.

Consider a scenario wherein we need to plan for a holiday trip to London. So we initially will have the official government site to learn about the visa processing procedures. Then we would have to check the site of various flights to get the flight schedules. Then we would have to browse for the various hotels in London and their locations. If the Semantic Web was being implemented, then if we specify that our domain of interest is `holiday trip', then computers would themselves process information about the visa procedures, flight schedules, hotels, etc and present it in an integrated form to the users.

An ontology [11] is a formal explicit description of concepts in a domain of concept (classes), properties of each concept describing various features and attributes of the concept and restrictions on properties [3]. Ontology can be considered as a composition of 5 main elements:

- **Class**: Classes represent an abstraction mechanism for grouping instances with similar characteristics (properties).
- **Properties:** These are binary relations connecting different concepts. They can be categorised as data properties and object properties. Data properties are those that represent relation between instances and literal values. Object properties on the other hand represent relations between instances.
- **Instances:** These are the actual entities of an ontology. They would belong to some particular class possessing the properties of that particular class.
- **Property Restrictions**: These are the constraints that are being applied to the value a property can hold. They can be value constraints that constrain the range of a property and cardinality constraints that constrain the number of values that the property can hold.
- **Facts:** These are statements that provide information of a instances.
- **Axioms:** These are statements that formally define the semantics of classes or properties (For e.g., subclass relationship).

Various semantic repositories have come into implementation. We can discuss some of them.

- **Sesame:** Sesame is an open source framework that can store ontology as RDF data. This repository provides in-memory storage as well as native Java store. In-memory storage store the complete RDF graph in memory and hence is the fastest. One limitation of this type is that the RDF graph size must not exceed the memory capacity. Native Java store stores the RDF graph on the disk and hence removes the limitation of in-memory storage.
- **RedLand:** This repository stores ontology in a triple storage and hence facilitating RDF parsing and query capabilities. It provides a command line tool that can convert the RDF graphs to a wide variety of formats including N triples, N3, Turtle,etc.
- **OpenLink Virtuoso Universal Server:** It is an object oriented relational database system that supports SQL queries as RDF/SPARQL operators. It actually combines the functionality of traditional RDBMS, ORDBMS, XML, RDF,etc into a single system. Hence a single instance of Virtuoso can provide a Linked Data view of the entire network that is concept oriented.

Initially the features of the Semantic Web were being studied and compared with the existing technologies.

TABLE 1 COMPARISON OF DATA MODELS

Features	Relational	Hierarchical	Graph
Example Format	MySQL, Oracle	XML	RDF/XML, turtle
Data	Table cell values	Tag/Attribute values	RDF
Metadata	Object Property Definitions	XSD/DTD	RDFS/OWL
Query Syntax	SQL	XPath	SPARQL
Identifier	Primary Key	Attribute Key value	URI
Semantics	No	No	Yes

As the details of these repositories were reviewed, it was being noticed that all of them follow the triple storage model. By using this model it is difficult to effectively express the hierarchical relationships. Humans will manually have to find what the hierarchies present in the ontology are. To avoid this problem, we try to propose a storage model that explicitly stores such relationships. Also, such repositories can only support the conversion of ontology to the database form. It cannot do the reverse process, hence if needed for future use, the ontology cannot be retrieved in its original format. To solve this limitation, the various databases to ontology conversion mechanisms were being analysed. Through these studies various mechanisms like D2RQ[3],RDOTE[14], DB2OWL[13]etc. were familiarised.

D2RQ provides a mapping language that generates a mapping between the relational databases and RDF graphs. This language creates RDF graphs of a database and is written as an N3 file. The generated ontology is read-only and no integration of multiple databases is possible. Also certain inferences cannot be made. RDOTE is another system to convert databases to ontology. In this queries are generated to extract RDBMS components that are then being converted to corresponding ontology components. It has the limitation that an automatic transformation is not possible as human intervention is required to define concepts like properties. DB2OWL analyses the database components as three table cases. Based on this a R2O document is generated. However this can also not integrate several databases to form a single ontology.

From these mechanisms it was noted that they express the databases as RDF graphs and complex relationships that can be expressed in OWL language will be missed out. Hence a mechanism is proposed that is being added to the proposed model in order to add on such relations. Moving on further, the idea of integrating the information from a database to some existing ontology came into mind. This had the advantage that the ontology can be gradually transformed into a global one that can be used in various contexts. Regarding integration, Ontograte[12] was a system that dealt with integration of two databases and

 ISSN: 2321-8169
2526 – 2534

then converting into an ontology. So this idea is also being included in the proposed method. As the various mechanisms were studied, a small comparison was generated as follows:

TABLE 2 COMPARISON OF DATABASE CONVERSION SYSTEMS

Features	D2RQ	RDOTE	Onto-Grate	DB2OWL
Mapping Language	Yes	No	No	No
Transformat-ion Engine	Yes	Yes	Yes	Yes
Direct Mapping	Yes	Yes	Yes	No
Query Based Access	Yes	Yes	No	No

III. EXISTING SYSTEM

As explained before most of the semantic repositories that is based on the relational database implement a monolithic schema wherein the ontology relations are expressed in a single table with in the form of triples. The proposed approach is mainly based on Jena, one of the commonly used semantic repositories.

A. Jena

Here Jena is being explained in detail. Jena is a Java application programming interface for Ontology storage. It mainly deals with the creation, manipulation and storage of RDF graphs. It supports both in-memory storage as well as persistent storage. An architecture of Jena can be depicted as follows:

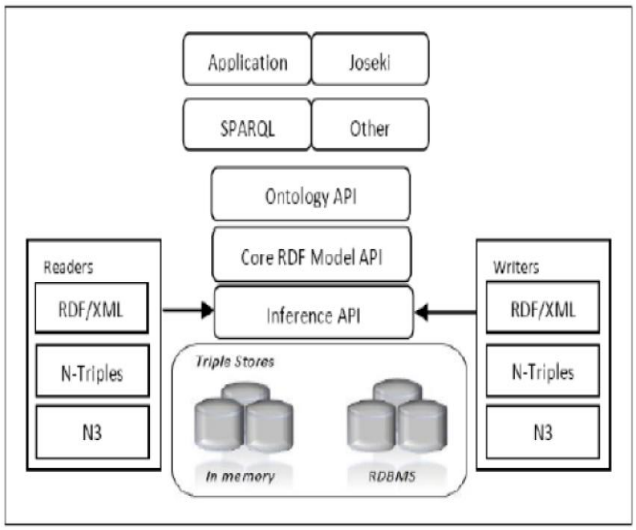

Figure 1: Architecture of Jena framework

As shown in Figure 1, Jena provides various tools, e.g., an RDF/XML parser, a query language, I/O modules for N3, N-triple and RDF/XML output. Underneath the API the user can choose to store RDF graphs in memory or in databases. It also provides additional functionality to support RDFS and OWL. Jena provides OWL storage based on relational databases and stores the ontology in:

- **Asserted Statement Table:** Once the OWL ontology is given, the real data for OWL documents (triples) is stored in this table. It consists of three columns entitled subject, property, and object.
- **Long Literals Table:** This table stores literals that are too long (more than 256 bytes) to store directly in a statement table. Each long literal is assigned a unique integer identifier and this identifier is used to reference the literal from a statement table.
- **Long Resources Table:** If the length of the URIs is too long (more than 256 bytes), then they are stored directly here.
- **Prefixes Table:** URIs often have common prefixes (e.g., http://www.example.org/prop1, http://www.example.org/prop2). There can be substantial space savings if prefixes are stored just once. Jena will optionally store common URI prefixes in the prefixes table.
- **System Statement Table:** The system statement table is used to store system metadata for the Jena storage subsystem such as configuration parameters, table names for graphs, etc. The metadata is expressed as RDF statements so this table looks very much like an asserted statement table**.**
- **Graph Table:** RDF data may also be stored in the form of graphs. The graph table stores the name and unique identifier for each user graph.
- **Reified Statement Table:** Reification in RDF and Jena is the ability to treat a Statement as a Resource, and hence to make assertions about that statement. Such reified statements are being stored here.

Jena supports both SPARQL1 and RDQL2 query languages to access RDF or OWL data. As Jena was being studied, it can be noted here also that there is no mention of the hierarchy. A user will have to check through all the tables to find any hierarchical relationships. So a model is proposed that can explicitly store such relations. Also such semantic repositories can simply store and at the most extent queries the ontology. However it is a fact that Semantic Web and its technologies are in the evolving stage. So it is required that ontologies must be created from scratch. In order to create such ontologies, the domain must be learnt, the context must be analyzed, then the various components of the ontology must be identified from the raw data. This consumes a lot of time. Most of the organizations store their data in relational databases. So if we could convert this database data to ontology, then the transition from the traditional Web to the Semantic Web can be done much faster. The paper discussed in the following section explains

 ISSN: 2321-8169
2526 – 2534

a mechanism wherein certain rules are defined that can convert such databases to ontology.

B. *Rule-Based Transformation of SQL Relational Databases to OWL Ontologies[2]*

This paper tries to propose an approach where relational databases are automatically converted to ontologies. A set of rules have been defined and according to these rules the conversion is performed. The rules can be explained as follows:

1. **Mapping tables:** All tables are mapped as classes.
2. **Mapping columns:** Each column can be mapped as a datatype property of the table it belongs to.
3. **Mapping row:** Each row of the table becomes an instance of the corresponding class having specific values for the corresponding datatype properties.
4. **Mapping datatypes:** The datatypes of the database is being mapped to XML Schema Datatypes. This datatype is being used by OWL since it does not have its own.
5. **Mapping constraints:** The constraints specified in the database is being analysed and is mapped to corresponding restrictions in OWL. For example, a not null constraint can be considered as minimum cardinality restriction of 1. Similarly a foreign key constraint can be considered as an object property, that relates two classes.

This approach can extract most of the components of the ontology. But the complex relationships of Web Ontology Language like subclass-superclass relationships, equivalent class relationships, etc cannot be extracted. This can lead to loss of information and affect the completeness of the ontology. To avoid this the proposed approach tries to derive rules that can include these information also.

IV. PROPOSED SYSTEM

As mentioned in the previous chapters, the thesis tries to propose a new approach that can overcome some of the limitations of the existing semantic repositories. In order to implement the inter-operability between relational database and ontology, there must be some mapping designed to store ontology as relational database and vice versa. As we have seen earlier, the existing semantic repositories cannot completely store the hierarchical relationships due to single storage model. This thesis tries not to only store the ontology as database but also retain the entire hierarchical relation. Also regarding the conversion of database to ontology, an approach is proposed that tries to design a mechanism that can convert without the need of a mapping language. It also tries to integrate the knowledge extracted from the database to existing ontologies thus adding to the knowledge base. A general architecture can be shown as follows:

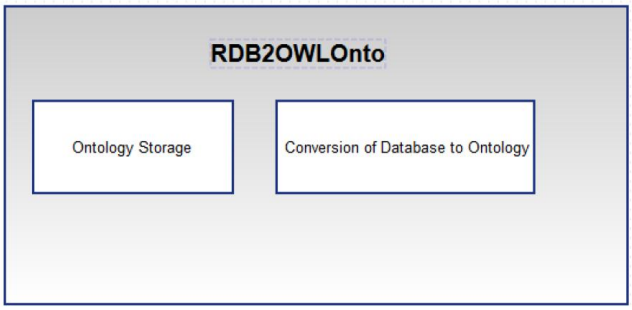

Figure 2: Architecture Overview

As shown in Figure 2, the system mainly consist of 2 modules: Ontology Storage and Database Conversion to Ontology. These modules are discussed in detail further.

A. *Ontology Storage*

As we have seen in earlier chapters, Ontology is a fast evolving concept and hence there must be some efficient mechanism to store these ontologies. Though there are several repositories that are being commonly used, but they still follow the single storage model wherein most of the data is being stored in tables in the triple format. The triple format means each ontology fact is being stored in the subject, object, predicate form where the subject and object could be instances and predicate will be any of the properties. As we saw in Jena, that follows this same model, it does not store any hierarchical relationships. Hence the ontology may not be complete that is being stored. So, here a new storage model is being proposed. Its architecture is as below:

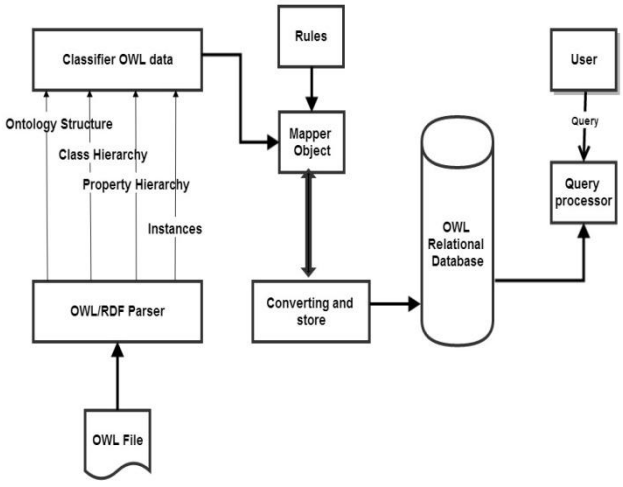

Figure 3 : Ontology Storage

Figure 3 shows that the system receives an OWL file containing the ontology. From this ontology, using OWL parsers, the structure of ontology is analysed and the various components of the ontology is being extracted. Depending on the components various tables are being created and the information related to these components are then inserted into the corresponding tables. Once all the tables are created, the users can now query the ontology using the traditional

 ISSN: 2321-8169
2526 – 2534

databases. The main advantage of this is that a person who does not know the syntax of ontology can get an overview of the various classes, properties, etc directly from the tables that are being created. Using one of the OWL parsers, the following components are being extracted:

1] Class: Classes represent an abstraction mechanism for grouping instances with similar characteristics. When all classes are being extracted the following details are being constructed as a table:

- **Class Details:** Information of each class like its name, ID, etc. are being stored.
- **Disjoint Classes:** Two classes are said to be disjoint when instances of one class can never be instances of the other. So information of disjoint classes, if any present in the ontology, is being stored as a table.
- **Equivalent Class:** Two classes are said to be equivalent when both the classes have exactly the same set of instances.
- **SubClass:** A class `A' is said to be a subclass of class `B' when instances of A are a subset of the instances of B.

2]Property: These represent the attributes that describe the instances of ontology. They could be either object properties(relating two instances) or data properties(relating an instance with a literal value). Once the properties are being extracted the following will be added on to tables:

- **Property details:** Information of each property like the name, ID, value type, etc will be stored.
- **SubProperties:** A property `A' is said to be a sub property of property `B' when values of A are a subset of the values of B.
- **Disjoint Properties:** 2 properties are said to be disjoint when instances that hold one property cannot hold the other.
- **Equivalent Properties:** Equivalent properties relate one individual to the same set of other individuals.
- **Property Characteristics:** A property maybe transitive, symmetric, etc. Such details are being analysed and stored in tables.
- **Inverse properties:** A property is said to be inverse of the other if one relates instances, then the other relates the same pair but in the reverse order.

3] Instances: Instances represent the actual entities of the ontology. Once these are extracted, the following is stored:

- **Instance details:** This includes the instance name, the class to which it belongs to, etc.
- **Same Individuals:** This indicates 2 URI references refer to the same thing.
- **Different Individuals:** This indicates that 2 URI references two different things.
- **Data Property Assertions:** These indicate the literal values that a particular individual can hold for a particular property.

4] Annotations: Annotations are metadata that maybe used to describe the ontology or a specific component of the ontology. It could be comments, description about authors, etc. Jena or other semantic repositories do not have option to store this annotation explicitly. Hence once the ontology is stored, we would find it difficult to retrieve the exact ontology that was being stored due to missing of such annotations. Hence a table is created for such ontologies are being constructed and the name, its type, its value,etc. are being stored.

5] Restrictions: Restrictions can be defined as the constraints on the values that a particular property can hold. There are mainly two type of restrictions: value constraints and cardinality constraints. Value constraints constrains the range of the property while cardinality constrains the number of values that a property can hold at the same time. These restrictions are also explicitly stored in this storage model.

B. *Conversion of Database to Ontology*

The next module of this system aims to convert the existing relational databases to an ontology form that includes the complete expressibility of OWL. The basic architecture can be shown as follows in Figure 4.

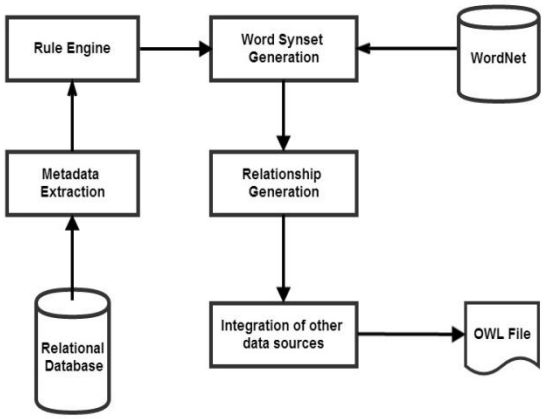

Figure 4: Database Conversion

Rules are designed, those that which when receives a database as input will extract the metadata and decides the classes, properties, etc. Also this mechanism tries to integrate concept knowledge base to the existing ontology store. It can be represented as an algorithm as:

Algorithm

Input: A relational Database
Output: OWL file of the same name as of the Database

- Step 1: User specifies the database to be converted
- Step 2: The database metadata is being extracted.
- Step 3: To the metadata the following rules are being applied.
    - Every relation in the database will become a class of the ontology
    - Every tuple of a relation will become instances of that class.
    - Each attribute of the relation, if it is not a primary or foreign key will become a datatype property with domain being the corresponding class and the range becomes a datatype of XML schema as per the mapping given in Figure 4.1
    - If an attribute is a primary key of the relation then it is represented as an InverseFunctional Property with domain, that of the class
    - If an attribute of a relation is a foreign key, then it is represented as a object property wherein the domain is the referencing relation and the range is the referenced table.
    - If an attribute has a `NOT NULL' constraint, then it is a property with MinCardinality 1
    - If an attribute has a `UNIQUE' constraint, then it is an inverse-functional property.
- Finally it is checked whether the knowledge extracted can be integrated to any of the existing ontologies, by comparison of the concepts in the newly created ontology and the existing ones.

1] Class Relationships: From a Relational Database of the traditional form, we cannot retrieve any relationships like subclass, superclass relationships from its structure. This is because the semantics is not being integrated into the databases. In order to retrieve such relationships we try to find the semantics [15] of the various concepts and then compare it with each other. For this purpose, WordNet is being used. Given a concept, it retrieves the synonyms, hyponyms (A specific term used to designate a member of a class. For instance, daisy and rose are hyponyms of flower), hypernyms(Word that names a broad category that includes other words) and antonyms. each other concept of the created ontology is being taken and then compared with the set of synonyms, hyponyms, hypernyms and antonyms. Then the following rule applied:

- If a concept C1 is in the synonym set of the input concept C2, then C1 and C2 are equivalent.
- If a concept C1 is in the hyponym set of the input concept C2, then C1 is the subclass of C2.
- If a concept C1 is in the hypernym set of the input concept C2, then C1 is the superclass of C2.
- If a concept C1 is in the antonym set of the input concept C2, then C1 and C2 are disjoint.

2] Integration of Knowledge to Existing Ontologies: One of the major limitations of existing conversion mechanism like D2RQ is that it cannot integrate the knowledge from various sources. Ontograte[12] is one system that tries to integrate multiple data sources. This paper tries to overcome this limitation. For that each ontology is retrieved from the ontology store and compared with the concepts of the newly created ontology using the synonym, hyponym, set as discussed above. If any match is found, then that relationships added it to the existing ontology. Most of the data nowadays are stored in the form of database, hence, instead of developing knowledge from scratch as ontology, it would be advantage if we can convert the knowledge in database as an ontology.

3]Altering the database: Another drawback of the D2RQ is that it creates a read only ontology. Hence the database cannot be edited. This paper provides facilities to the user to edit the ontology. For this purpose the newly created ontology is being loaded using the OWL API and then it provides the necessary functions to edit the ontology. After the ontology is being edited, the corresponding changes are made in the database by connecting to the corresponding database.
Consider for example a class is to be added, then the OWL API function will be:

```
OWLClass clsAMethodB = factory.getOWLClass("A", pm);

OWLDeclarationAxiom declarationAxiom =
factory.getOWLDeclarationAxiom(clsAMethodA);
manager.addAxiom(ontology, declarationAxiom);
```

This code adds a class named `A. The declaration is being done as an axiom. When such a class is created the database that has the same name as that of the ontology is connected and a table named `A' is being created.

V. IMPLEMENTATION AND RESULT

After deciding on the system that was proposed earlier , the next question was how to implement the system. It was a need to decide on what are the different programming tools that are needed in order to implement the system as discussed. Initially the question of which programming language must be used to implement the system came up. When the studies was done it was understood that Java is one of the languages that supported most of the projects of semantic web, since it has many application programming interfaces that supports the development of such projects. so this system was being implemented using Java.

Moving through each module, the first one was Storage of Ontology to a relational database. In this module, the input to be received was an ontology that was written in OWL. Hence in order to give the users an option to specify their desired file, it was understood that an interface was needed. If this interface was web based any user around the globe could use it thus spreading the range of such systems

 ISSN: 2321-8169
2526 – 2534

to people worldwide. It could be by such systems that the Semantic Web could gain popularity as our traditional Web. Hence to make this interface web based, Java Server Pages and Java Servlets were being used. Then the next step was to decide what all should be provided in the interface. Hence an interface was being designed that is being shown in Figure 5

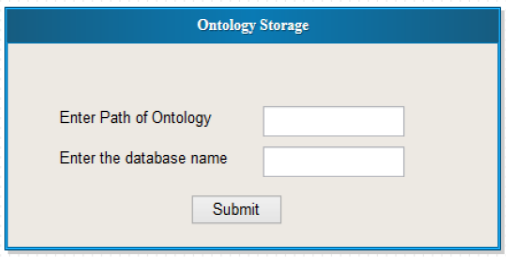

Figure 5: Interface Design for Ontology Storage

Once the interface was being designed, when the users have selected their required Ontology file, this must be parsed by some mechanism in order to extract the various components of the ontology. For this the various APIs were being considered. It was being identified Jena provides an API for this purpose and it is one of the most commonly used one. But when gone into detail, it was noticed that Jena API[16] deals ontology in the form of RDF graphs. Hence complex ontology relationships that is present in OWL and not in RDF like equivalent classes, disjoint classes, etc cannot be handled by Jena API. So this proved to be a limitation. To overcome, other APIs were looked into detail. And among them OWL API was the one found to be most suitable. So the parsing was done using the OWL API[10]. The OWL API initially started off by loading the ontology using a OWL Ontology Manager object. This manager is responsible for creating data factories and extracting the various components from the ontology into those data factories..

As the extraction was proceeded, it was realised that OWL API was able to extract various components of the ontology. But it was unable to extract the restrictions like cardinality constraints or the value constraints cannot be retrieved. Hence to do this there was the need of some other APIs else this data would have been lost. Hence it was decided that the Jena API be used for this purpose. So when we use the Jena API along with the OWL API. the limitations of Jena API and OWL API both are being hid.

Once the components are being extracted, according to the type of the components, the information is being stored in the corresponding tables. Hence it was required that a database management system had to be selected. For this purpose, MySQL was being used. The main reason why this was used is that the database could be easily connected to Java program through jdbc driver. Also the metadata of this database could easily be retrieved that will be used for future purposes. Therefore for each ontology a database was being created that holds the same name as the ontology. Then the components of this ontology was being stored in tables as being discussed earlier.

To make it easier for the user, instead of the user needing to manually access the database from the database server, this system provides an interface to view the database. Instead of displaying the entire database in a single page, it categorises the information. This means that once the user selects the required ontology, the various classes, instances and properties of the ontology is being displayed. Once the user selects any of them, then all information from all tables that relates to that particular entity is being displayed on the web page. Hence the development of this storage module was being completed.

The next step was the conversion of existing relational databases to ontology. As discussed before this was required to make the transition from traditional Web to Semantic Web possible. Again as before the users must be given the facility to select the database that must be converted to ontology. Once the user selects the database, its transformation must be done. For this the metadata of that particular database is being retrieved. This is being done in java using the jdbc driver. To this metadata, the rules are being applied. These rules are being coded using java itself. Depending on the rules, the components of the database are being mapped to ontology components and it is being written to into an OWL file. The rules have been discussed in the earlier chapter. So once this is complete, a OWL file describing the entire database as an ontology is being obtained.

When these rules were applied, the rules dealt only with basic things like classes, properties, instances,etc. It did not include complex relationships like equivalent classes, subclasses, etc within the database . If this has to be done we need to get the semantics of each words representing the classes. For including semantics, WordNet[17] dictionary was being used. WordNet describes about each word in the form of synset and this synset is used to obtain all the related words of that particular word. From these related words its been checked whether any of them is similar to other class names. If so, according to the relation type, the ontology relation is being decided.

Another functionality that was being added to the system was integrating the information obtained from the databases into existing ontologies if there are relating terms between the newly created ones and the existing ones in the ontology store. For doing this again the synset that was retrieved from the WordNet is again being used to check the similarity of terms between pair of ontologies. Likewise all ontologies are being considered. If any relation is identified then the relation is being added into our new ontology. Gradually this could be evolved in global ontology that can be used in different domains. This integration is not available in existing conversion mechanisms.

After the system was being developed, an analysis was being carried out in order to compare the existing systems with the new ones. The various features that was being considered are :

- Mapping Language - In order to transform the database to an Ontology some systems use a special mapping language. Eg:- R_2O
- Transformation Engine -This module will be responsible for applying rules to convert the databases.
- Direct Mapping - This, also known as automatic mapping involves conversion using simple rules.
- Query based Access - conversion of databases take place based on the query given by the user.
- User defined unique IDs- Some systems generate a unique ID during the conversion process rather than only using the primary key of the relation.
- Data types - this signifies whether there is a proper translation of SQL data types to XML data types.
- Integration with existing Ontologies - This indicates whether we can integrate any related information from the existing ontology to the newly created ontology.
- Integration of databases- This indicates whether two databases can be integrated into a single one.
- Editable Ontologies- Some systems create read-only ontologies.

Several ontology files were taken as input and was executed on D2RQ, DB2OWL, Ontograte and the new system(DB2OWLOnto). Based on the outputs obtained, the following were summarised:

TABLE 3 COMPARISON OF D2RQ, DB2OWL, OntoGrate AND DB2OWLOnto

Features	D2RQ	Onto-Grate	DB2OWL	DB2OWL-Onto
Mapping Language	Yes	No	No	NO
Transfor-mation Engine	Yes	Yes	Yes	Yes
Direct Mapping	Yes	No	Yes	Yes
Query Based Access	Yes	No	Yes	No
User Defined Unique IDs	Yes	No	No	Yes
Data Types	Yes	No	Yes	Yes
Integration with Existing Ontologies	No	No	Yes	Yes
Integration of Databases	No	Yes	No	Yes
OWL complex relation mapping	No	Yes	No	Yes

VI. CONCLUSION AND FUTURE WORK

A system that implements the mapping of ontology to relational database and vice versa is designed, thus facilitating their inter-operability. The system is much efficient in storing ontologies as databases as it maintains the hierarchical relations and other complex relations. This is different from the existing semantic repositories in the sense that they have a single storage model and hence cannot store hierarchical relationships. These relations are explicitly stored in the new model that is being proposed. Hence the complete ontology with no information lost can be stored as a database. Interfaces are also provided to the user so that the user can view the database with a better understanding of classes, properties, instances, etc.

Along with storage, conversion of existing relational databases to ontology is also included in the proposed system. By bringing in such a mechanism, it helps in actually accelerating the transition from the traditional Web to the Semantic Web and hence exploiting the complete usability of the benefits of the semantic concepts brought in by Semantic Web. The system not apply simple rules to convert the basic ontology components but also tries to discover the semantic relationship between various terms within the database itself, hence trying to explicitly define complex OWL relationships. Also to make use of this data, the integration of other related ontologies to the newly created one is also being done. Hence it can be gradually evolved into a global ontology that can be later reused in various other contexts.

This system is designed in such a way to support MySQL databases. This was so because it was easy to extract the metadata of the databases using java. It can be further extended to support heterogeneous databases. Also in the conversion mechanism, the direct mapping is being followed. In order to make it much more efficient, augmented direct mapping that tries to analyze the relational patterns of the database. This can convey information of the domain semantics. Another thought for improvement could be to use some other knowledge base other than WordNet to lead to more efficient knowledge base.

VII. REFERENCES

[1] McBride, B., "Jena: a semantic Web toolkit," Internet Computing, IEEE , vol.6, no.6, pp.55,59, Nov/Dec 2002
[2] Astrova, I., Korda, N., Kalja, A.: Rule-Based Transformation of SQL Relational Databases to OWL Ontologies. In: Proceedings

of the 2nd International Conference on Metadata & Semantics Research (October 2007)

[3] Christian Bizer, Andy Seaborne: D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs (Poster). 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan, November 2004.

[4] T. Berners-Lee, J. Hendler and O. Lassila, ``The Semantic Web'', Scientific American, 2001.

[5] Liu Baolin and Hu Bo. ``An evaluation of RDF storage systems for large data applications.'' In Semantics, Knowledge and Grid, 2005. SKG '05. First International Conference on, pages 59–59, 2005.

[6] RDF Working Group. ``RDF is a standard model for data interchange on the Web, http://www.w3.org/RDF/'', 25 February 2014.

[7] Jos de Bruijn, Axel Polleres, and Dieter Fensel. ``D20v0.1 OWL Lite-. http://www.wsmo.org/2004/d20/v0.1/20040629/'', June 2004.

[8] Chris Tipping. The Semantic Web: Layered Architecture, http://www.mctipit.com/computers/semantic-web-layered-architecture/

[9] S. Zhou, H. Ling, M. Han, "Ontology Generator from Relational Database Based on Jena". In Computer and Information Science. Vol. 3, No. 2; May 2010.

[10] Matthew Horridge, Sean Bechhofer. ``The OWL API: A Java API for OWL Ontologies Semantic Web Journal 2(1), Special Issue on Semantic Web Tools and Systems'', pp. 11-21, 2011.

[11] Noy, Natalya F., and Deborah L. McGuinness.``Ontology development 101: A guide to creating your first ontology.'' (2001).

[12] Dejing Dou, Han Qin and Paea Lependu. ``ONTOGRATE: Towards Automatic Integration for Relational Databases and the Semantic Web through an Ontology based Framework'', Advanced Integration and Mining Lab Computer and Information Science University of Oregon, Eugene, OR 97403, USA.

[13] Nadine Cullot, Raji Ghawi, and Kokou Yétongnon. ``DB2OWL: A Tool for Automatic Database-to-Ontology Mapping'', Laboratoire LE2I, Université de Bourgogne, Dijon, FRANCE.

[14] Vavliakis, K.N., Grollios, T.K., Mitkas, P.A.: RDOTE - transforming relational databases into semantic web data. In: Proc. 9th Int. Semantic Web Conf. (November 2010)

[15] Dixit, P.; Sethi, S.; Sharma, A K.; Dixit, A, "Design of an Automatic Ontology Construction Mechanism Using Semantic Analysis of the Documents," Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on , vol., no., pp.611,616, 3-5 Nov. 2012

[16] B. McBride, D. Boothby, C. Dollin, ``An Introduction to RDF and the Jena RDF API'', August, Vol. 1 (2004)

[17] George A. Miller, ``WordNet: a lexical database for English, Communications of the ACM'', v.38 n.11, p.39-41, Nov. 1995