CONVERTING RELATIONAL DATABASE INTO OWL ONTOLOGY BY MULTI-WAY SEMANTICS EXTRACTION

Sohee Jang, Insuk Park, Hoyun Cho And Soon Joo Hyun
School of Engineering, Information and Communications University
119, Munjiro, Yuseong-gu, Daejeon, 305-732, Korea

Keywords: Semantic Web, Relational Database, Web Ontology Language (OWL).

Abstract: Semantic Web provides means to share well-defined meaning of terms with semantically annotated information. In the current Web, most of the Web applications generate Web contents dynamically at the time of user request from underlying relational databases. To represent the relational data to the Semantic Web environment, the relational data should be transformed into the ontology form. In this paper, we propose a Semantic Web technique to convert relational database into ontology in OWL using multi-way semantics extraction technique. Extracted from E/R modeling components, schema descriptions and stored data, the generated ontology will provide application developers with rich semantics so as to quickly build a knowledge base for advanced Semantic Web services. Extracting the semantic information out of the traditional databases will provide enterprises with more opportunities for many value-added services.

1 INTRODUCTION

Exchanging machine-understandable data on the Web is an important research issue. In the current Web, contents are dynamically collected at the time of query requests typically from underlying databases, and the contents of the database-driven Web sites do not have the semantics that machines can understand. The Semantic Web framework and ontology are intended to give a solution to the problem (Berners-Lee et al., 2001).

Let us assume that a buying agent in e-businesses communicates with selling agents. The buying agent may only understand the values of merchandises by property of <*worth*>, whereas some other selling agents may use different properties, such as <*price*>. To share and exchange the merchandise information among those software agents, the information expressed in different terms in different databases need to be mutually understood and thus made transparent in a way through the Semantic Web using ontology expression. The proposed research in this paper was motivated by the fact that the structural model and semantic constraints, such as type information, cardinality, and uniqueness expressed in the ontology are closely compatible to those of relational database schema. So that semantics can be automatically extracted by well-defined conversion rules.

There are several researches in this track (Upadhyaya et al., 2005)(Buccella et al., 2004)(Laborda et al., 2005)(Bizer, 2003)(Korotkiy et al., 2004). Upadhyaya et al. have developed a tool for extracting OWL ontology from the Extended E/R models, named *ERONTO* (Upadhyaya et al., 2005). Although they considered the E/R models that are widely used for the semantic design of the databases, they did not consider semantics extractions from the relational schema and conversion of stored data in a database. Therefore, the data stored in the relational databases can not be automatically converted into ontology individuals. The mapping rules proposed by Buccella et al. take into account only the relational schema, e.g. SQL/DDL, for generating OWL ontology (Buccella et al, 2004). They also did not convert its stored data into ontology individuals. Laborda et al. introduced *Relational.OWL* ontology for the purpose of exchanging data among remote database systems (Laborda et al., 2005).

In this paper, we propose a multi-way semantics extraction strategy by converting a relational schema and its E/R components into OWL ontology and the stored data into ontology individuals. Our proposed strategy employs three different extraction paths: E/R to ontology, relational schema to ontology and

Jang S., Park I., Cho H. and Joo Hyun S. (2007).
CONVERTING RELATIONAL DATABASE INTO OWL ONTOLOGY BY MULTI-WAY SEMANTICS EXTRACTION.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - ISAS*, pages 563-568
DOI: 10.5220/0002383105630568

relational data to ontology individuals as depicted in Figure 1 of Section 2. Our approach uses pure OWL language. OWL can represent the meaning of a term which is machine understandable in the Semantic Web. OWL is used to define ontology and has more vocabularies for expressing meaning and semantics than XML, RDF, and RDF-S (Smith et al, 2004)(Manola et al, 2004). Therefore, the resulting ontology and ontology individuals can be used by Semantic Web applications and inference engines. In addition, the migration of stored data into ontology individuals will help Semantic Web application developers easily build the knowledge base for Semantic Web service environments.

This paper is organized as follows. Section 2 presents the details of the proposed conversion rules and procedures. Section 3 shows how the relational database is converted into ontology using a relational database example. Finally, Section 4 concludes the proposed work.

2 RDB-TO-OWL CONVERSION RULES

Different from the existing researches on this issue, the proposed strategy reported in this paper involves multi-way semantics extractions from the relational metadata, its E/R modeling components and stored relational data as shown in Figure 1. The E/R model implies more semantics than the relational model does (Chen, 2002). We can extract semantics of data from entities and relationships of E/R model. Such semantics cannot be extracted from relational model. The relations of the relational model are just tables and imply little semantics. Therefore, we considered both the relational schema and E/R model. In addition, the stored data in a database must be migrated to ontology individuals so that this converting work becomes useful and practical. By doing so, the resulting semantics in OWL ontology will be made more expressive and useful to the application developers during the course of service design.

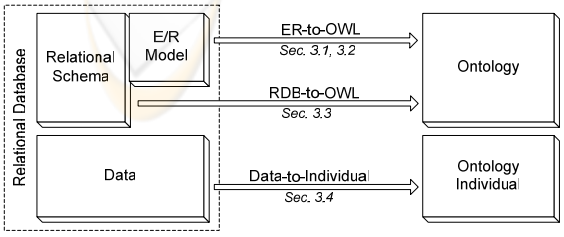

Figure 1: Multi-way Semantics Extraction Scheme.

The semantics extraction procedure is as follows. At first, ontology semantics are extracted from the E/R modeling components producing an *ER-to-OWL Map*. Next, some information of the relational schema that is not represented in the E/R model, such as the data types of attributes, is also converted into the ontology so as to produce an *RDB-to-OWL Map*. Finally, ontology individuals are generated from the stored data in the database by using *ER-to-OWL Map* and *RDB-to-OWL Map*. Table 1 shows the mapping rules from both E/R model and relational schema information to OWL ontology. In this paper, we consider the standard E/R model, not the Extended E/R model (Elmasri et al., 2003).

Table 1: Mapping E/R Model and Relational Schema to Ontology.

E/R or Relational Model	**OWL Ontology**
Entity	Class
Simple attribute	Functional data property
Composite attribute	Functional data properties
Multi-valued attribute	Non-functional data type property
Key attribute	Functional data property and inverse functional data property
1:1 relationship	A pair of two functional object properties
1:N relationship	A pair of functional object property and non-functional object property
Weak entity	A pair of functional object property with cardinality 1 and non-functional object property
M:N relationship	A pair of two non-functional object properties
N-ary relationship	A bridging class and *N* pairs of object properties
Role	Name of a property
Participation	Cardinality constraint
Data type of column	XML data type

The schema information of the E/R model and the relational model are listed in the first column of Table 1. The second column represents the corresponding of OWL ontology descriptions. The conversion rules indicate entity, attribute, relationship, role and participation of the E/R model, and data types in the relational database. There are detailed explanations and examples in Section 3.

3 MAPPINGS USING AN EXAMPLE *COMPANY* DATABASE

In this section, we demonstrate how the mapping rules defined in the previous section generate the corresponding OWL ontology expressions. Using a typical *COMPANY* database given in Figure 2,

Section 3.1, 3.2 and 3.3 illustrate how entities and attributes, relationships and other schema descriptions are expressed into OWL ontology, respectively. Section 3.4 shows how ontology individuals are extracted from the relational data.

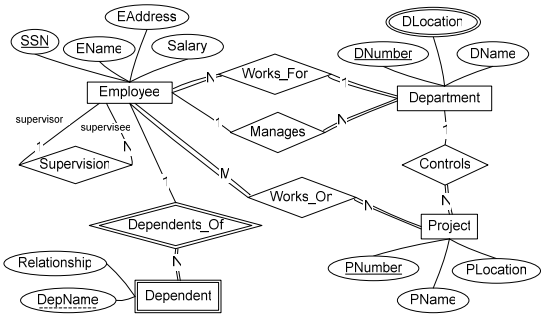

Figure 2. E/R Diagram of *COMPANY* Database.

3.1 Entities and Attributes

3.1.1 Entities

An *entity* is transformed into a *class* in OWL. The name of the entity is assigned to an *ID* of the OWL class. In the *COMPANY* database, *Department* entity is converted a class. Figure 3 shows how *Department* entity and its attributes are converted into OWL forms.

3.1.2 Attributes

There are three types of *attributes*: *simple attribute*, *composite attribute* and *multi-valued attribute*. First, the simple attribute is transformed into a *functional data type property* in OWL. Since an entity has a value for its simple attribute, the data type property of the simple attribute has to be the *functional property* in OWL. *DName* in Figure 3 is an example of a simple attribute. Second, each member attribute of a composite attribute in E/R model is transformed into a *functional data type property* in OWL. Finally, a multi-valued attribute is converted into a *non-functional data type property* in OWL. Since an entity has one or more values for the multi-valued attribute, the attribute is not the functional property in OWL. *DLocation* in Figure 3 is an example of a multi-valued attribute.

If an attribute is a *key attribute* of an entity, the property of the key attribute is also *inverse functional property*. The class of the entity has a *cardinality restriction* of *1* on the property of the key attribute. *DNumber* attribute in Figure 3 is an example of the key attribute of the entity.

```
<owl:Class rdf:ID="Department">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
rdf:resource="#DNumber" />
      <owl:cardinality
rdf:datatype="&xsd;nonNegativeInteger">1</ow
l:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty rdf:ID="DNumber">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdf:type
rdf:resource="&owl;InverseFunctionalProperty
" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="DName">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="DLocation" />
```

Figure 3: OWL Description of an Entity and its Attributes.

It is to be noted that not all the semantics can be transformed into OWL ontology due to the inherent expression gap between the languages of databases and ontology: SQL and OWL. There are a few mapping difficulties which have to be handled with some ad-hoc expressions in OWL after all. First, the scope of the uniqueness of the *inverse functional property* is different from the scope of the uniqueness of the *primary key*. The former is viewed in ontology and the latter is viewed within a table. Second, it is difficult to transform the *composite keys* to OWL forms. We use several *properties* to represent the attributes of the *composite keys* in OWL. The properties can be defined as *inverse functional properties.* However, it does not mean that the combination of their values is unique and not possible for the combination of all the properties to be defined by using one *inverse functional property*.

3.2 Relationship

Relationships are divided into two categories; *binary relationships* and *N-ary relationships*. Binary relationships are divided again into three: *1:1, 1:N and M:N relationships*. The *property* of the OWL describes directions unlike the relationships in E/R model. Therefore, we use a pair of *object properties* that are *inverse* to each other.

3.2.1 1:1 Relationship

This relationship is transformed into a pair of *functional object properties*. One of the two classes participating in the relationship is assigned to the *domain* of a property and the other class is assigned

to *range* of a property. One property is an *inverse of* the other property. Figure 4 shows 1:1 relationship between *Employee* and *Department* entities.

```
<owl:Class rdf:ID="Employee" />
<owl:Class rdf:ID="Department" />
<owl:ObjectProperty rdf:ID="Manages">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Employee" />
  <rdfs:range rdf:resource="#Department" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Has_Manager">
  <owl:inverseOf rdf:resource="#Manages" />
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Department" />
  <rdfs:range rdf:resource="#Employee" />
</owl:ObjectProperty>
```

Figure 4. OWL Description of 1:1 Relationship.

3.2.2 1:N Relationship

This relationship is the same as 1:1 relationship except that one of two object properties is *non-functional property*. The object property which has a class of N-side entity as *domain*, and a class of 1-side entity as *range* becomes a non-functional property as the N-side entity can participate in the relationship more than once.

The relationship between a *weak entity* and a *strong entity* is the special case of the 1:N relationship. It is the same as 1:N relationship except that the class of a weak entity has a *cardinality restriction* of *1* on the functional property. Since the key of strong entity also plays a role of a partial key of the weak entity, the weak entity must participate in the relationship, exactly once. For example, *Dependent* entity and *Employee* entity in the *COMPANY* database are a weak entity and a strong entity, respectively. Figure 5 shows an OWL description of weak entity and strong entity.

3.2.3 M:N Relationship

M:N relationship is converted into a pair of non-functional object properties in OWL. In relational databases, representation of the M:N relationships is a little tricky. A bridging table is used in order to describe the M:N relationship. However, OWL can represent M:N relationship like other binary relationships. In the *COMPANY* database, *Employee* entity and *Project* entity are participated in M:N relationship, through *Work_on* relationship. Figure 6 shows an OWL description of the M:N relationship.

```
<owl:Class rdf:ID="Employee" />
<owl:Class rdf:ID="Dependent">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
rdf:resource="#Dependents_Of" />
      <owl:minCardinality
rdf:datatype="&xsd;nonNegativeInteger">1</owl
:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="Has_Dependents">
  <rdfs:domain rdf:resource="#Employee" />
  <rdfs:range rdf:resource="#Dependent" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Dependents_Of">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <owl:inverseOf
rdf:resource="#Has_Dependents" />
  <rdfs:domain rdf:resource="#Dependent" />
  <rdfs:range rdf:resource="#Employee" />
</owl:ObjectProperty>
```

Figure 5. OWL Description of a Relationship between Strong Entity and Weak Entity.

```
<owl:Class rdf:ID="Employee" />
<owl:Class rdf:ID="Project" />
<owl:ObjectProperty rdf:ID="Works_On">
  <rdfs:domain rdf:resource="#Employee" />
  <rdfs:range rdf:resource="#Project" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Has_Members">
  <owl:inverseOf rdf:resource="#Works_On" />
  <rdfs:domain rdf:resource="#Project" />
  <rdfs:range rdf:resource="#Employee" />
</owl:ObjectProperty>
```

Figure 6. OWL Description of M:N Relationship.

3.2.4 N-ary Relationship

Since OWL language supports binary relationship, the binary relationship in a relational database can easily be represented in OWL. However, ternary or N-ary relationships cannot be converted by the mapping rules of the binary relationships. In order to convert N-ary relationships, a *bridging* class in OWL can be generated and then the *bridging* class and all the classes participating in the relationships are connected by OWL *object properties*.

3.2.5 Role

Each entity that participates in a relationship plays a particular *role* in the relationship. The role name signifies the role of participating entity in each relationship. In the *COMPANY* database, both participants in the *Supervision* relationship are *Employee* entities. One is a *Supervisor* and the other is a *Supervisee*. The role name is a good candidate for the name of *object properties* in OWL.

3.2.6 Participation constraints

There are two types of participation constraints in the E/R model: *total participation* and *partial participation*. If an entity has a total participation for a relationship, the OWL class of the entity must have a *min-cardinality restriction* of *1* on the object property of the relationship. For example, *Department* entity must have a manager in the *COMPANY* database. Therefore, the *Department* entity has the total participation for the *Manages* relationship. Figure 7 shows an OWL description of the *Department* entity and *Manages* relationship.

```
<owl:Class rdf:ID="Department">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
rdf:resource="#Has_Manager" />
      <owl:minCardinality
rdf:datatype="&xsd;nonNegativeInteger">1</owl:mi
nCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="Has_Manager">
  <owl:inverseOf rdf:resource="#Manages" />
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Department" />
  <rdfs:range rdf:resource="#Employee" />
</owl:ObjectProperty>
```

Figure 7. OWL Description of Binary Relationship with Total Participation.

3.3 Relational Schema

In the previous section, we discussed E/R-to-ontology conversion in an attempt to extract semantics from the E/R model. Semantics in a relational database can mostly be extracted through E/R-to-ontology conversion rules. However, some semantic information cannot be obtained from the E/R model, such as *data types* of attributes in the relational databases as they are not explicitly modeled. Data type property in OWL has a data type as *range* so that the data types of attributes of the relational database can be directly converted into data types of XML Schema. Table 2 shows the mappings between built-in database data types and XML Schema data types.

For example, *SSN* attribute and *EName* attribute of *Employee* entity in the *COMPANY* database have *varchar* and *integer* data types, respectively. In OWL, *data type properties* can be represented as shown in Figure 8.

Table 2. Mapping between Built-in Database Data Types and XML Schema Data Types

Database	XML Schema
bigint	*long*
decimal	*decimal*
char, nchar	*string*
text, ntext	
varchar, nvarchar	
datetime	*date*
smalldatetime	
int	*integer*
tinyint	

```
<owl:DatatypeProperty rdf:ID="SSN">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdf:type
rdf:resource="&owl;InverseFunctionalProperty" />
  <rdfs:domain rdf:resource="#Employee" />
  <rdfs:range  rdf:resource="&xsd;integer" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="EName">
  <rdf:type
rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Employee" />
  <rdfs:range  rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

Figure 8. OWL Description of Data Types of Attributes.

As the result of our proposed multi-way semantics extraction scheme, the ontology of the *COMPANY* database is generated as shown in Figure 9. Note that the structure of the resulting ontology in Figure 9 is very similar to the structure of the source E/R model in Figure 2.

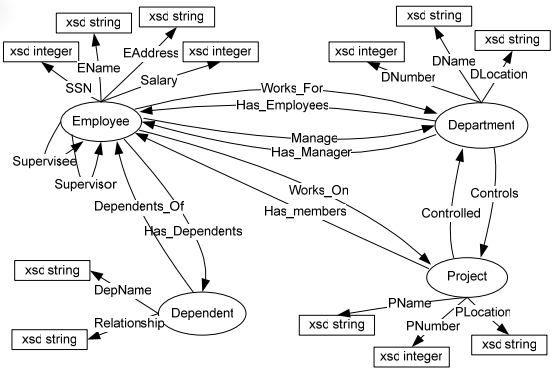

Figure 9. RDF Graph of Resulting Ontology for the *Company* Database.

3.4 Generating Ontology Individuals

To use the relational databases within the Semantic Web framework, the stored data of a database as well as schema information of the database must be taken into description for ontology individuals. After

the ontology is produced, the process of data migration transforms all the records in the database into ontology individuals.

For this, the ER-to-OWL map and RDB-to-OWL map obtained during the previous extraction procedure (refer back to Section 2) are utilized. All the tuples of the tables are transformed to ontology individuals and unique IDs are assigned to the ontology individuals. A good candidate for a unique ID is the value of the key attribute. Figure 10 shows an example ontology individual for *Employee* entity of the *COMPANY* database.

```
<Employee rdf:ID="1234567890">
 <SSN
rdf:datatype="&xsd;integer">1234567890</SSN>
 <EName
rdf:datatype="&xsd;string">John</EName>
 <EAddress rdf:datatype="&xsd;string">Apple
Street 192</EAddress>
 <Salary
rdf:datatype="&xsd;integer">32000000</Salary>
 <Works_For rdf:resource="#0123" />
 <Works_On rdf:resource="#20060732" />
 <Has_Dependents
rdf:resource="#1234567890_Nayoung" />
 <Has_Dependents
rdf:resource="#1234567890_Hoyun" />
</Employee>
```

Figure 10. An Ontology Individual for *Employee* Entity.

4 CONCLUSION

Semantic information extracted from the databases are useful to create knowledge services in addition to the conventional database services. In this line of effort, researches have been conducted to extract semantic information out of existing relational databases and the extracted semantic information is represented in the ontology framework using OWL standard expressions.

However, the semantics of data are not explicitly expressed in the relational model. Moreover the relational schema implies less semantic information than the E/R model does. In this paper, therefore, we have presented a multi-way semantics extraction scheme which extracts semantic information from both E/R modeling components and relational schema descriptions and then converts them into ontology in OWL. In addition, migrating data into ontology individuals allows application developers to quickly build the knowledge base in the Semantic Web environments. Although the resulting ontology cannot imply all semantics of the original database, the resulting semantics in OWL ontology of our work will be made much more expressive for the application developers than that of existing works. In addition, different from the existing approaches, the resulting ontology and ontology individuals converted by our work are fully compatible with OWL language, so that they can be easily used by OWL-based Semantic Web applications and inference engines.

As one of the future works, we will explore the ontology technique and develop various application services to show how the extracted ontology semantics are utilized. Using the extracted semantics, we can develop more sophisticated services on the Semantic Web than traditional database services. We will further explore the ontology techniques to be used on top of databases in an attempt to create knowledge-based application services for enterprises.

REFERENCES

Berners-Lee, T., Hendler, J., & Lassila, O., (2001, May). The Semantic Web. *Scientific American*. Vol. 284, no. 5, pp. 28-37.

Bizer, C., (2003). D2R MAP - A Database to RDF Mapping Language. *12th International World Wide Web Conference* – Posters.

Buccella, A., Penabad, M. R., Rodriguez, F. J., Faria, A., & Cechich, A., (2004). From Relational Databases to OWL Ontologies. *6th Russian Conference on Digital Libraries*, Pushchino, Russia.

Chen, P., (2002, June). Entity-Relationship Modeling: Historical Events, Future Trends, and Lessons Learned. Software *Pioneers: Contributions to Software Engineering*, Springer-Verlag, Berlin, LNCS, pp. 100-114

Elmasri, R., & Navathe, S. B., (2003). Fundamentals of Database Systems. 4th ed., *Addison-Wesley*.

Korotkiy, M., & Top, J. L., (2004). From Relational Data to RDFS models. *Proceedings of the International Conference on Web Engineering*, Munich.

Laborda, C. P., & Conrad, S., (2005). Relational.OWL - A Data and Schema Representation Format Based on OWL. *Proceedings of the 2nd Asia-Pacific conference on Conceptual modeling*, Vol. 43, pp. 89-96.

Manola, F., & Miller, E., (2004, February). RDF Primer, *W3C Recommendation*, from http://www.w3.org/TR/rdf-primer/

Smith, M. K., Welty, C., & McGuinness, D. L., (2004, February). OWL Web Ontology Language Guide. *W3C Recommendation*, from http://www.w3.org/TR/owl-guide/

Upadhyaya, S. P., & Kumar, P. S., (2005). ERONTO: A Tool for Extracting Ontologies from Extended E/R Diagrams. *Proceedings of the 2005 ACM symposium on Applied computing*, pp. 666-670.