Implementation of Ontology Learning and Population System from Structured Data Sources: Standard-based Approach

Dongwon Jeong
Dept. of Statistics & Computer Science,
Kunsan National University
djeong@kunsan.ac.kr

Abstract

This paper proposes an ontology learning and population model from structured data sources. Recently various attempts have been made to harmonize Web 2.0 and the Semantic Web, named as Web 3.0 or Web 4.0. One of the most important issues for realization of the next Web platform is about how to make Web ontology rich as well as to address semantic interoperability between ontologies. To resolve those issues, Web ontology schemas should be precisely defined in semantic aspect and we should also develop a method for learning and population of ontology instances from diverse resources. This paper proposes a model which enables the ontology learning and population from structured data sources. The proposed model in this paper is a standard-based approach. In other words, our proposal is based on ISO/IEC 11179 - Metadata Registry, which is one of the international standards. This standard has been developed to support the interoperability between data by managing standardized common concepts. Therefore, our proposed model enhances the semantic interoperability between ontologies and enables the ontology enrichment from data sources.

Keywords: Ontology, Learning and population, Structured data, Metadata registry

1. Introduction

Web 3.0 harmonizing Web 2.0 and the Semantic Web is an attractive issue, and much research has been studied for realization of Web 3.0 [1]. The Semantic Web is one of key technologies for implementation of Web 3.0 and makes it possible for machines to semantically understand requests of human and use Web resources [2-3].

Building ontology is indispensible for realizing the Semantic Web, and thus many useful and powerful technologies have been developed to build, manage, and utilize ontology such as various editors (Protege, OntoStudio), Web ontology description languages (RDF, RDF Schema, OWL), storages and inference engines (Jena, Sesame, Bossam) [4-8].

For providing high quality services using ontology, the following two issues should be resolved.

• To define ontology schemas using qualified-common concepts and improve the interoperability of ontologies
• To develop an ontology population method for providing richer information

This paper proposes an ontology population model based on ISO/IEC 11179 – Metadata Registry (MDR) to address the aforementioned requirements. MDR is an international standard developed by ISO/IEC JTC 1/SC 32 [9] and various MDR management systems have been developed [10-17]. MDR provides to manage and utilize qualified semantics and supports a mechanism for exchanging and sharing data between databases. With the
mechanism of MDR, we can populate instances of an ontology from the databases. In other words, a database schema is defined by using common concepts in an MDR. An ontology schema is defined by using the concepts in the MDR and we can develop richer information services by inserting instances of databases into the defined ontology.

This paper defines the mapping relationships between key elements of MDR and Ontology. The overall framework and system architecture is described, and a prototype implementation is given with concrete examples. This paper performs evaluations to show the contributions of the proposed model.

2. Proposed Model

This section describes the overall framework of the proposed model. This paper aims to implement the proposed learning and population model. Therefore, this paper illustrates a system architecture and data model for the implementation.

2.1. Framework

Figure 1 shows the overall architecture for the proposed model. In this figure, the proposed model learns and populates ontology instances using common concepts based on MDR. The proposed model consists of four parts as described in Figure 1.

![Figure 1. Overall Architecture of the Proposed Model: Framework](image)

The first part builds an MDR management system for managing and registering common concepts, instances of the key MDR elements such as Data Element, Data Element Concept, Object Class, Property, Conceptual Domain, and so on. The MDR management system also includes the other classes specified in ISO/IEC 11179. In this part, a storage model for the MDR management system should be defined in accordance with the metamodel of ISO/IEC
The MDR management system consists of the key six components as follows:

- Authorization Management
- Metadata Registration Function
- User Management
- MDR Schema
- Metadata Instances (Common concepts)
- Database Connection Management

The second part creates database instances. The database schemas are defined based on the common concepts registered in the MDR management system. In a word, this part creates databases without semantic heterogeneity between fields at schema level.

The third part defines ontology schemas using common concepts, which are registered to the MDR management system. In this part, the MDR elements should be mapped to the generic ontology elements. The mapping rules can be defined as follows:

- Key MDR elements = \{Object Class, Property, Conceptual Domain, Value Domain, Data Element Concept, Data Element, Relationship\}
- Ontology elements = \{Concept, Property, Data Type, Relation\}
- Mapping relations = \{(Conceptual Domain, Concept), (Object Class, Concept), (Data Element Concept, Property), (Property, Property), (Data Element, Property), (Data Element, Data Type), (Relationship, Relation)\}

The final part populates ontology instances by inserting instances in databases into the created ontology. The ontology population is actually performed in this part. Additional functions for the accuracy improvement of instances are performed such as the review task and the checking task by a developer.

2.2. System Architecture and Mapping Model

Figure 2 illustrates the system architecture for the proposed model. The system architecture consists of four data processing layers – MDR Building & Access (MBA) layer, Database Schema Building (DSB) layer, Ontology Schema Building (OSB) layer, and Population layer.

![Figure 2. System Architecture for the Proposed Model](image-url)
The MBA layer has a role to generate, manage, access, and retrieve a metadata registry. To accomplish the role, the MBA layer includes many kinds of operations for typed table management, metadata generation, metadata selection, user-defined type management, and access methods to tables. The DSB layer creates and manages database schemas based on MDR and consists of a function set for element mapping, database schemas creation, metadata loading, and so on. The OSB layer is one the most important modules in this system. Its main role is to generate ontologies using a set of common concepts in a metadata registry. The layer is composed of metadata loader, data receiver, ontology element mapper, ontology schema creator, and graphic viewer. Finally, the Population layer executes the learning and population operation from database instances. This layer contains various components for proper database schema recommendation, instance saving, ontology connection, rule-based learning, and enrichment of ontology instances. The database schema recommender suggests a set of meaningful candidates by using the mapping relationships between the elements of MDR and ontology.

The conceptual mapping model between MDR and ontology is shown in Figure 3. The MDR elements, Concept Domain and Object Class are mapped to one of the ontology elements Class (Concept). Data Element Concept, Properties, and Data Element of MDR are mapped to Property, which is one of the ontology elements. In MDR, Data Element is the composition of Property and Value Domain, and thus Data Element is also mapped to DataType of ontology. Value Domain and Relationships are mapped to DataType and Relation respectively.

![Figure 3. Mapping Relations between Key Elements of MDR & Ontology](image)

**3. System Implementation**

This section shows the implementation result with a concrete example and we can recognize the contributions of the ontology population model proposed in this paper.

**3.1. Prototyping Environment**

For the system development, an MDR management system should be first implemented. The MDR management system in this paper is developed based on relational database model and the storage structure for the management system, S is composed of the following key relations.

\[
S = \{ \text{Data Element, Object Class Relationship, Object Class, Property, Conceptual Domain Relationship, Conceptual Domain, Data Element Concept, Value Domain, Registration Authority} \} 
\]
In this paper, the ontology Univ_bench is used as the example dataset for implementations and evaluations. This ontology has been defined for the SWAT (The Semantic Web and Agent Technologies) project from Lehigh University [18] and has been widely used for experiment and testing in many studies related with the Semantic Web. Therefore, this paper also uses the ontology Univ_bench to show the implementation. Figure 5 shows the ontology example Univ_bench in graphic form.

The metadata registry built in this paper contains the schemas and properties from the ontology Univ_bench. In other words, all concepts are registered into the metadata registry. The registered concepts are reused for defining databases and user-defined ontologies. Every database can be designed with different tables and fields reusing the concepts in the metadata registry, and thus a database includes a set of instances from the ontology Univ_bench. In any cases, a database can contain additional instances not from Univ_bench.

As aforementioned, the database schemas should be defined by using the qualified-common concepts registered in the metadata registry implementation. Before the creation of the database instances, the building process of the metadata registry implementation according to the ISO/IEC 11179 standard guideline should be preceded.

The examples of the definition of database schema field are summarized as follows:

Database Instances = \{DB1, DB2, DB3\}

- **DB1**: Defining the database schema field with all instances having Student as the value of the Object_Class field out of the instances of Data_Element table
- **DB2**: Defining the database schema field with several instances having Student as the value of the Object_Class field out of the instances of Data_Element table
- **DB3**: Defining the database schema field with several instances having Student and Professor as the value of Object_Class field out of the instances of Data_Element table

![Figure 4. Ontology Example: Graphical Presentation](image)
3.2. Implementation Results: Database/Ontology Schema Generation

This section describes the implementation results of the database and ontology schema generation with several snapshots.

3.2.1. Database Schema Generation: For the implementation of the proposed model, this paper creates a set of database schemas using the standardized-common concepts registered in the MDR system. Figure 5 shows the interface for the database schema generation. The common concepts stored in the MDR system can be retrieved by using this interface and a database schema including tables and fields can be defined with the retrieved result.

![Figure 5. Creating a Database Schema using Common Concepts](image)

3.2.2. Ontology Schema Generation: To generate an ontology schema, the following three steps are required: (1) ontology class generation; (2) property and data type definition; and (3) relation definition.
Figure 6 shows a snapshot for the ontology class generation. The concepts registered in the metadata registry are displayed and proper concepts can be chosen for defining ontology classes.

The second step is the property generation. As this paper intended to represent the ontology as OWL, the ontology properties have been classified into datatypes and object properties defined in OWL. Figure 7 shows the interface for the datatype properties and data type generation. To generate a datatype property, a user should first select a class. Then the user can choose datatype properties (e.g., age) and their corresponding data types (e.g., integer). For example, the class “Person” should be selected before defining the datatype property “age”. The generated data is stored in the triple model, i.e., stored into the table that consists of SUBJECT, PREDICATE, and OBJECT.
Figure 8 shows the interface for generating the object properties and relations. They are generated by selecting the domain and range for the object properties and by selecting the corresponding object properties or relations.

Figure 7. Ontology Generation: Creating Datatype Properties

Figure 8. Ontology Generation: Creating Object Properties

Through the aforementioned three steps, an ontology schema generation is completed. Figure 9 shows a part of the generated ontology schema in graphical form.
3.3. Implementation Results: Ontology Instance Population

This section describes the implementation of the ontology instance population with two snapshots. One is the snapshot of the ontology instance population operation and the other is the snapshot of the ontology population result. Figure 10 and Figure 11 illustrate the former and the latter respectively.

In Figure 10, the user interface is composed of three parts. The area “Ontology” first displays the ontology schema structure. A class is selected and one of the properties of the chosen class is selected. It intends to populate the chosen property.

Once determined a population target, the system recommends a set of candidates by performing the rule-based learning operation. The candidates are displayed on the area “Recommendation”. For this operation, the learning component uses the mapping relations between the elements of metadata registry and ontology. One of the candidates recommended is chosen and its corresponding data set is loaded. An example of the data set is shown in the area “Database”.

The data set is checked and determined by a user, and then the ontology property is populated with the qualified data set. The operation is semi-automatically executed to guarantee the high quality of the ontology instance population.

4. Evaluation

This paper compares of two models to show the contributions of the proposed model. One is the generic Web-based population model that populates from Web resources. The other is the population model based on MDR, i.e., the proposed model in this paper.

For the evaluation, this paper assumes the followings:

- The same ontology schema is used
- A set of databases using common concepts from a metadata registry management system is built
- The ontology schema is defined by using common concepts from a metadata registry management system
- Distribution of the same instances
Figure 10. Result of Ontology Population

Figure 11. Part of Ontology Instances Populated
The comparative item for the evaluation is processing time. For populating ontology, a lot of data are first extracted as candidates from databases or Web. The candidates should be checked to be added to instances of ontology. This work is required to guarantee accuracy of instances. Therefore, most cost is spent for the checking process. In this paper, the comparative evaluation is conducted based on the number of checking operation.

The two algorithms are defined in Table 1 and Table 2. The overall algorithm of the two models is basically similar. After retrieving a set of candidates for instances using keywords, the candidates are checked for their accuracy. The checking process is to filter valid instances. On the other hand, the two models have differences in their approach. The generic ontology population model based on Web checks instances at value level, but the proposed model conducts checking at schema level, exactly at attribute level.

<table>
<thead>
<tr>
<th>Table 1. The Learning and Population Model Proposed in this Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>K</strong>: INPUT; candidates, selectedInstances : LIST; <strong>c</strong> : STRING;</td>
</tr>
<tr>
<td><strong>BEGIN</strong><em>PROPOSED</em></td>
</tr>
<tr>
<td>candidates = getSchemaInfo(Ki);</td>
</tr>
<tr>
<td>WHILE(candidates.next() != NULL)</td>
</tr>
<tr>
<td>c = candidates.getMetadata();</td>
</tr>
<tr>
<td>IF (select(c) == TRUE))</td>
</tr>
<tr>
<td>selectedInstances.add(c);</td>
</tr>
<tr>
<td>END_IF</td>
</tr>
<tr>
<td>END_WHILE</td>
</tr>
<tr>
<td>populate(selectedInstances);</td>
</tr>
<tr>
<td><strong>END</strong><em>PROPOSED</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. Generic Web-based Learning and Population Model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>K</strong>: INPUT; candidates, selectedInstances : LIST; <strong>c</strong> : STRING;</td>
</tr>
<tr>
<td><strong>BEGIN</strong><em>PROPOSED</em></td>
</tr>
<tr>
<td>candidates = retrieveInfo(Ki);</td>
</tr>
<tr>
<td>WHILE(candidates.next() != NULL)</td>
</tr>
<tr>
<td>c = candidates.getLink();</td>
</tr>
<tr>
<td>visit(c);</td>
</tr>
<tr>
<td>IF (select(c) == TRUE))</td>
</tr>
<tr>
<td>selectedInstances.add(c);</td>
</tr>
<tr>
<td>END_IF</td>
</tr>
<tr>
<td>END_WHILE</td>
</tr>
<tr>
<td>populate(selectedInstances);</td>
</tr>
<tr>
<td><strong>END</strong><em>PROPOSED</em></td>
</tr>
</tbody>
</table>

With the algorithms, this paper defines the computational models. First **CMI**_PROPOSED_, the computational model for the proposed model, is defined as follows:

\[
\text{CMI}_{\text{PROPOSED}} = \text{Cal(getSchemaInfo(Ki))} + \text{Cal(candidates.getMetadata())} \tag{1}
\]

In case of the worst case of the proposed model, i.e., all instances are to be searched, **CMI**_PROPOSED_ is redefined as follows:

\[
\text{CMI}_{\text{PROPOSED}} = n(F_A) + n(F_A) = 2n(F_A) \tag{2}
\]
CMIGENERIC, the computational model for the generic web-based model, is defined as follows:
\[ CMIGENERIC = \text{Cal}(\text{retrieveInfo}(K_i)) + \text{Cal}(\text{Candidates.getLink()}) \]  
(3)

In here, if the generic web-based model is in the best case, CMIGENERIC can be redefined as follows:
\[ CMIGENERIC = n(I_S) + n(I_S) = 2n(I_S) \]  
(4)

However, if the general web-based model is in the worst case scenario, CMIGENERIC is as follows:
\[ CMIGENERIC(General) = n(I_A) + n(I_A) = 2n(I_A) \]  
(5)

Table 3 summarizes the key operations and conditions, and the number of instances to be checked according to the algorithms defined in Table 1 and Table 2.

<table>
<thead>
<tr>
<th>Operations &amp; conditions</th>
<th>Generic Web-based model</th>
<th>Proposed model</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of total instances</td>
<td>(n(I_A))</td>
<td>(n(I_A))</td>
</tr>
<tr>
<td>Candidate instances retrieved</td>
<td>(n(I_S) + \alpha) ((0 \leq \alpha \leq n(n(I_A) - n(I_S)))</td>
<td>(n(I_S) + \beta) ((0 \leq \beta \leq n(F_A - F_S))</td>
</tr>
<tr>
<td>Cost required for searching</td>
<td>In case the accuracy ratio is best, (\alpha = 0)</td>
<td>In case the accuracy ratio is worst, (\beta = n(F_A) - n(F_S))</td>
</tr>
<tr>
<td>The number of instances to be checked</td>
<td>In case the accuracy ratio is best, (n(I_S) + \alpha = n(I_S) + 0 = n(I_S))</td>
<td>In case the accuracy ratio is worst, (n(F_S) + \beta = n(F_S) + ((n(F_A) - n(F_S)) = n(F_A))</td>
</tr>
</tbody>
</table>

In the generic Web-based model, the number of candidates to insert instances of \(n(I_S)\) is \(n(I_S) + \alpha\). The best case in the generic Web-based model is when \(\alpha = 0\), because \(\alpha\) is larger than 0. Hence, when the accuracy is the best, the number of the served candidate instances is \(n(I_S)\).

On the other hand, when the accuracy is the worst case scenario in the proposed model, the number of the searched candidate instances is not \(n(I_A)\) but \(n(F_A)\), because the process for finding candidates is performed at field level not at value level.

The notations used in this paper are as follows:

- \(n(I_A)\): The number of all instances, it is fixed
- \(n(K)\): The number of keywords to insert the instances; The set of the numbers for experiment = \(\{1, 5, 10, 20, 50\}\)
- \(n(T_i,V)\): The number of each attribute, this paper supposes that it is same in all attributes, its set is \(\{100, 1000\}\)
- \(n(F)\): The number of attributes, \(n(F) \approx n(I_A) / n(T_i,V)\)
- \(n(I_S)\): The number of instances that users want, \(n(I_S) = 1000\)

Figure 11 shows the evaluation result on the candidate instances performed with different property numbers. In Figure 11, the proposed model is better than the generic Web-based model in most cases except the case in Figure 11-(b).
Figure 11. Evaluation Results on the Number of Candidates

The processing time for checking can be defined as the following computational models. The computational models for the processing time consider the operations to retrieve candidate instances. However, the other operations are excluded, because the effect from their costs is little.

This paper assumes that the operation time for each operation, t is 0<t<1. The computational model for the proposed model is defined as follows:

\[
CMT_{ \text{PROPOSED} } = T.getS + \Sigma_{(i=1}^{n FA)} (T.getM+T.selectI) \tag{6}
\]

, where

- \(T.getS\): The processing time for \(getSchemaInfo(K_i)\)
- \(T.getM\): The processing time for \(candidates.getMetadata()\)
- \(T.selectI\): The processing time for \(select(c)\)

The computational model for the general Web-based model is defined as follows:

\[
CMI_{ \text{GENERIC} } = T.getInfo + \Sigma_{(i=1}^{n IS)} (T.getL+T.visit+T.selectI) \tag{7}
\]

, where

- \(T.getInfo\): The processing time for \(retrieveInfo(K_i)\)
- \(T.getL\): The processing time for \(candidates.getLink()\)
- \(T.visit\): The processing time for \(visit(c)\)
- \(T.selectI\): The processing time for \(selected(c)\)
Figure 12 illustrates the comparative evaluation results on the processing time between the proposed model and the general Web-based model. In Figure 12, the proposed model shows better performance than the general Web-based model. Even though the generic Web-based model is the best case and the proposed model is the worst case, the proposed model is equal to or better than the generic Web-based model.

5. Related Work

This section introduces ISO/IEC 11179 – Metadata Registry (MDR) and describes the existing approaches for ontology learning and population approaches. ISO/IEC 11179 – Metadata Registry (MDR) has been developed for addressing the semantic interoperability between databases. This standard consists of six parts and its core part is the 3rd part. This part specifies the metamodel and basic attributes for managing metadata. The key classes of the metamodel are: Conceptual domain, Property, Object class, Data element concept, Value domain, and Data element. The 3rd edition of this standard has been currently developing to reflect the ontology concept, and the revision of the 3rd part has been published this year [9].

Much research on ontology population has been conducted and several studies have been focused on learning and population from relational databases. W3C surveyed and classified those studies into three categories [19]. As presented in the survey, the most critical issue is there is no standard method for populating RDF ontology from relational databases. Most of the existing approaches represent and map between RDF and RDB. In other words, they learn and populate their ontology with local semantics not common semantics. It makes many problems such as low semantic interoperability, high ontology generation cost using diverse resources, low semantics usability, and so on.
The standard, ISO/IEC 11179 provides the mechanism for managing and reusing semantics in a standardized way. It also specifies the registration procedure to qualify for quality of registered semantics. We can resolve several problems of the previous approach using the features of ISO/IEC 11179.

6. Conclusion

This paper proposed an ontology population model based on ISO/IEC 11179 – Metadata Registry (MDR). This paper also described a prototype system with an example including the comparative evaluations. With the evaluation results, the proposed model showed better performance than the generic Web-based model. The proposed model encourages the usability of MDR. In addition, the proposal provides the facility for building ontologies with common concepts and also guarantees the accuracy of ontology population result.

In future, this proposed model will be extended to resolve several limitations. The proposed model in this paper can be adopted for the restricted environment that databases are defined based on the standard, ISO/IEC 11179: MDR. Especially, this paper assumed that the database schemas are based on a metadata registry management system. However, in future, this model will be extended to consider two or more metadata registry management systems. Nowadays one is hot issues is about big data, linked data, and open data. Therefore, we need to extend the proposed model for such environments. In a word, a hybrid population model will be developed to facilitate usability of the proposed model.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NO.2012-0002439).

References


Author

Dongwon Jeong, received his Ph.D. degree in Computer Science from Korea University, Korea, 2004. He worked as a full-time instructor, 1999-2000 in the Advanced Institute of Information Technology, Korea. He was a Research Assistant Professor, Korea University, 2004-2005. He was a Visiting Research Scholar (PostDoc.), School of Information Sciences & Technology, Pennsylvania State University, USA, 2005. Now, he is a Professor in Dept. of Statistics & Computer Science, Kunsan National University, Korea from 2005. He is a Committee Member of Metadata Project Group (PG 606) and NGIS Project Group (PG 409), Telecommunications Technology Association (TTA), Korea, 2004-Present and 2009-Present. He is a Korean Mirror Committee Member, ISO/IEC JTC 1/SC 32 and ISO TC/211, Korean Agency for Technology and Standards (KATS), Korea, 2006-Present and 2009-Present respectively. He is now a Visiting Research Scholar, Department of Computer Science and Engineering, Oakland University, USA from 2013. He is interested in Data engineering, the Semantic Web, Smart mobile services, Ubiquitous computing, Cloud services, Big data, Open data, Linked data, Social Network Services, and Security.