# The Mixed Ontology Building Methodology Using Database Information

Minyoung Ra, Donghee Yoo, Sungchun No, Jinhee Shin, and Changhee Han

***Abstract*—Recently, there has been a growing need for research to manage the knowledge of an organization effectively using ontology. To increase the effect of knowledge management, the development of a well-defined ontology using various concepts about the knowledge of an organization is needed. There are two approaches in the current methodology for ontology development. One approach is to develop ontology from database information, the other approach is to construct ontology using domain terms according to a top-down method or bottom-up method, and so on. In this paper, we propose a Mixed Ontology Building Methodology (MOBM) which combines the characteristics of both approaches to more effectively represent organizational knowledge on ontology. The proposed method first creates kernel ontology as the core, using various types of database information, including database schema, and then completes the additional ontology by applying the top-down method and the bottom-up method, respectively.**

***Index Terms*—ontology, ontology building methodology, kernel ontology, database schema**

## I. Introduction

Ontologies are formal and consensual specifications of conceptualizations that provide shared understanding of a domain [1]. Ontology has been utilized in knowledge management, natural languages processing, information retrieval and database integration, but recently it has been suggested as a promising method for the management of Internet resources in the new generation web environment called semantic web. Consequently, the number of ontologies is increasing rapidly. The development of ontology is becoming a crucial part of semantic web and knowledge management, and importance of ontology is increasing continuously. In order to develop a well-focused ontology that can manage an organization's knowledge and information effectively, the various concepts necessary for good knowledge management should be defined clearly in the ontology used.

The process of developing ontology suitable for an organization's knowledge management requires a lot of time and considerable cost [2]. Accordingly, it is important to reuse already pre-developed ontologies. In general, however, it is hard for an organization to find a pre-developed ontology that expresses that organization's information appropriately, so therefore, it is necessary to develop ontology that is customized to the specific organization. The core of ontology development is to define the key concepts necessary for the clear expression of an organization's knowledge and reduce the cost of development by simplifying that development process. One method is to utilize database information actually used in the organization to the fullest. A database is a depository of information, and in relational databases, the data are stored as tables. Such a database contains a large volume of information that is very important to the corresponding application domains. Thus, the utilization of well-organized information in a relational database will allow for quicker and more accurate collection of the core concepts required for the development of a precise ontology for each task.

Existing ontology development methodologies are largely divided into two groups. One lies in the direction of conceptualizing ontology and includes the bottom-up method [3,4], top-down method [5], middle-out method [6] and hybrid method [7]. The other methodology creates the ontology from an existing database schema [8, 9, 10, 11, 12]. However, there are many restrictions in terms of building ontology that accurately expresses an organization's knowledge and information when using just one method. That is, the methods in the former group do not deal with database information that expresses an organization's knowledge and information. In the latter method, the ontology expresses only the concepts in the database and therefore, only those limited terms necessary for knowledge expression are included in the ontology.

Thus, this paper proposes a Mixed Ontology Building Methodology (MOBM) that combines the characteristics of both approaches to more effectively represent an organization's knowledge as ontology. The proposed method first creates a kernel ontology as much as possible, which then becomes the core, using various types of database information, including database schema. The method completes the additional parts of the ontology by applying the top-down method and the bottom-up method respectively. This paper describes the process for the staged application of the proposed methodology based on a scenario for the virtual database company and evaluates that methodology.

The paper is structured as follows. In Chapter II, we review related works. In Chapter III, we present the overview of MOBM proposed in this paper and describe the details of each step of this approach. In Chapter IV, we introduce an exemplary scenario, using the MOBM, and in Chapter V we analyze lessons learned from this process. Lastly, in Chapter VI we draw conclusions and suggest future research.

## II. Related Works

Much work has been done on the issue of ontology building methodology. The related work can be divided into

Manuscript received November 29, 2011; revised January 12, 2012.

This research was supported by ADD (Agency for Defense Development). Contract Number UD110058MD.

Minyoung Ra, Donghee Yoo, Sungchun No, Jinhee Shin and Changhee Han are with the Faculty of Electronics Engineering & Information Science, Korea Military Academy, Seoul, Republic of Korea (e-mail: {myra, dhyoo, is695, suhacci, chhan}@kma.ac.kr).

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

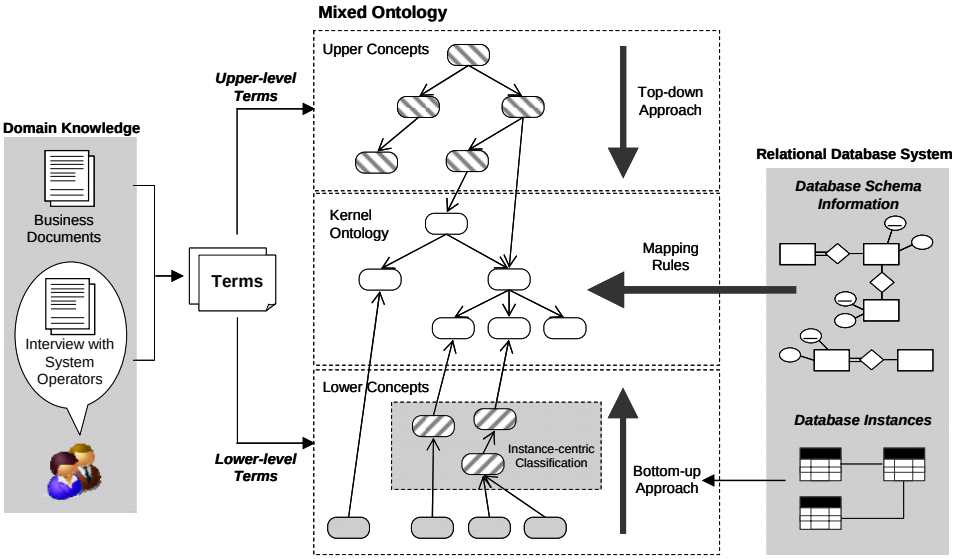


Fig. 1. Overview of the MOBM

two categories. The first category collects terminology, then builds the ontology by first analyzing concepts, then forming a hierarchy for the concepts, and defining the relations between the concepts and the rules for acquiring domain knowledge. According to the refinement process assigned to this task, the ontology is then completed. Several methods have been reported for this task. The bottom-up method starts from the most specific classes and then groups them into more general concepts [3, 4]. The top-down method starts with the definition of the most general concepts and then divides these into subsequent sub-concepts in detail [5]. The middle-out method starts with the certain middle level concepts and then applies the bottom-up method or the top-down method appropriately as needed [6]. The hybrid method merges ontologies developed from the bottom-up method and top-down method, respectively, into one ontology [7].

The second category of ontology building is to develop an ontology from database schemas. This work is studied by taking three directions: (1) Extract the ER model first from the database schema using reengineering, then from that model extract the ontology [8]; (2) Given the database schema and ontology, for semantic web applications, mapping rules between them are extracted [9, 10]; and (3) Generate the ontology structure itself from the relational database schema [11, 12].

In this paper, a mixed methodology is presented, which first generates a kernel ontology using database information as much as possible and then completes the ontology by applying the bottom-up method and the top-down method, respectively, to build additional parts of the ontology.

## III. The Mixed Ontology Building as Methodology

### A. Overview of the MOBM

Fig. 1 depicts the overview of the MOBM. In this methodology, mapping rules are defined to extract the main concepts and main relations of certain domain ontology from the target database schema. This kind of domain ontology is called kernel ontology. Kernel ontology is enhanced by adding upper-level terms and lower-level terms, which are collected from domain knowledge or instances of the target database because they may have new concepts or new relations which did not exist in the target database schema. Based on the top-down method, the upper-level terms are conceptualized into upper concepts. In the same way, the lower-level terms are conceptualized into lower concepts using the bottom-up method. Once the upper concepts and lower concepts are developed, they are then linked to the kernel ontology.

The MOBM has eight steps for building domain ontologies as follows:

- Step 1. Extraction of kernel ontology from database schema
- Step 2. Making class hierarchies from upper concepts
- Step 3. Making class hierarchies from lower concepts
- Step 4. Connecting of these class hierarchies into a kernel ontology
- Step 5. Enhancing the semantics between inter-terms
- Step 6. Enhancing any restrictions
- Step 7. Enhancing additional axioms and rules
- Step 8. Completion of the ontology

Fig. 2 illustrates the building sequence for each step. In Fig. 2, some steps do not occur sequentially. For example, Step 2 and Step 3 can be executed in parallel terms, regardless of their sequence, and in the same way as Step 5 to Step 7 are.

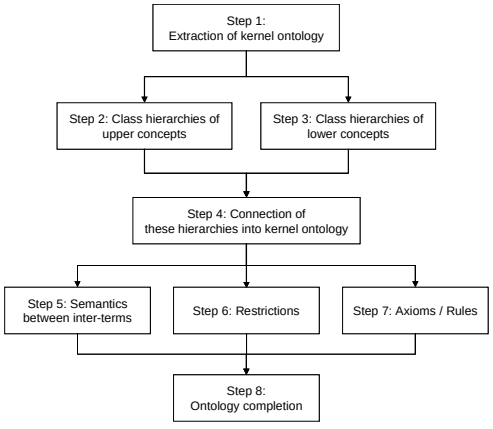


Fig. 2. Building Steps for the MOBM

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

### *B. Extraction of kernel ontology from database schemas*

Until now, many algorithms have been developed to extract a domain ontology from database information [8, 9, 10]. Among this database information, database schema is the most frequently used because it includes core concepts and relations for building a domain ontology properly. In this step, we present a set of fundamental mapping rules to extract components of the kernel ontology from the database schema. Fig. 3 lists the core database schema information used in the mapping rules.

| |
|---|
| Database names, Relation names, Attribute names, Primary keys, Foreign keys, Attribute data types, M:N relationship constraints, Integrity constraints, Multi-valued attributes |

Fig. 3. Core Information found in the Database Schema

In this paper, kernel ontology is represented in OWL (Web Ontology Language). Thus, the mapping rules should include how to transfer the core information of the database schema into the components of OWL, such as Class, Object property, and Datatype property. For this purpose, we use the following notations:

Suppose that the database schema (DS) has N tables. Then,

- $T_i$: the i-th table in DS where i = 1, 2, …, N
- $Att_{i,j}$: the j-th attribute in $T_i$ where j = 1, 2, …, $N_i$
- $PK_i$: the set of the primary keys of $T_i$
- $FK_i$: the set of the foreign keys of $T_i$
- CK: the set of composite keys

Then, we have the following equations formed from the definition.

$$DS = \bigcup_{i=1}^{N} T_i, \qquad T_i = \bigcup_{j=1}^{N_i} Att_{i,j} \text{ for all i}$$

Extracting the kernel ontology from DS, we have

- $C_k$: the k-th class in the kernel ontology where k = 1, 2, …, M ≤ N
- C: the set of classes in the kernel ontology, where

$$C = \bigcup_{k=1}^{M} C_k$$

Based on the notations developed above, the *mapping rules* are compiled as shown in Fig. 4.

**Rule 1**: Find $T_x$ for all x = 1, 2,…, N such that $T_x \notin C$. This rule has the following two cases.
(1) $T_x$ such that $PK_x \subset CK$ and $T_x - PK_x = FK_x$: This case corresponds to the M:N relationship.
(2) $T_x$ such that $PK_x \subset CK$, $\exists FK_x$, $FK_x \subset PK_x$ and $T_x - PK_x = \emptyset$: This case treats the multi-valued attribute.

**Rule 2**: Map all other tables onto the ontology classes except the tables corresponding to Rule 1 above.

**Rule 3**: Specify the properties between the classes. For some y = 1, 2,…, N,
(1) if $\exists FK_y$ and $FK_y = PK_y$, then set up the subclass-relation between those two classes.
(2) if $\exists FK_y$ and $FK_y \neq PK_y$, then establish the referential integrity constraint between those two classes.
(3) $PK_y \subset CK$, $\exists FK_y$, $FK_y \subset PK_y$ and $T_y - PK_y \neq \emptyset$ (case of weak entity), then set up the is-part-of Object property between those two classes.

**Rule 4**: If the M:N relationship exists, set up the inverse Object property between those two classes.

**Rule 5**: For the case of the table, which treats the multi-valued attribute, $T_z$ where z = 1, 2,…, N, $PK_z (\neq FK_z)$ is identified as the Datatype property of the referencing class, and the maximum cardinality of the property has to be considered.

**Rule 6**: Specify the Datatype property for the remaining columns that are non-FK attributes of the table.

Fig. 4. Mapping Rules for Extracting Kernel Ontology

### *C. Making class hierarchies from upper concepts*

In this step, the upper concepts of the kernel ontology are conceptualized based on domain knowledge, such as an interview with the system operator and business documents. To this end, we first selected new terms that did not exist in the target database schema from the domain knowledge. Among these selected terms, we identified upper-level terms that can be defined in the upper concepts of the kernel ontology. These upper-level terms are conceptualized into upper concepts using the top-down method.

### *D. Making class hierarchies from lower concepts*

This step specifies the lower concepts of the kernel ontology. The lower-level terms are collected from the instances of the target database and the domain knowledge. Then, the bottom-up method is adopted to build the lower concepts. To do this task, first the lower-level terms are identified as the most specific individuals, and then we generalize them into more abstract concepts. Therefore, some instances of the database can be defined in a concept.

### *E. Connection of these class hierarchies into a kernel ontology*

In this step, the upper concepts in section *C* of Chapter III and the lower concepts in section *D* of Chapter III are connected to the kernel ontology to integrate them into a single ontology. This ontology is called a mixed ontology. The connection methods are dynamically decided based on whether some concepts have the same name or do not. The former case is automatically recognized in that it has the same concepts, and the latter case is semi-automatically recognized using machine learning methods, such as lexical checking and semantic checking.

### *F. Enhancing the semantics between inter-terms*

The semantics between inter-terms can be obtained from the database schema or from domain knowledge. As earlier mentioned in section *B* of Chapter III, some semantics are identified when the kernel ontology is extracted from the database information. Thus, this step defines the enhanced semantics between inter-terms that are not included in the database information. There are two types of enhanced semantics; one is related to class hierarchies, and the other is related to the relationships between classes. In the former case, additional semantics for class hierarchies will be

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

defined not only as *subClassOf* but also as *equivalentClass*, *disjointWith*, *intersectionOf*, and so on. In the latter case, new relations will be specified between the upper concepts, lower concepts, and concepts of kernel ontology when the mixed ontology is generated. In addition, additional semantics, such as *inverseOf*, *symmetric*, *transitive,* can be defined in the mixed ontology when new semantics are discovered from the database information or the domain knowledge.

### G. Enhancing any restrictions

If some restrictions are identified from the domain knowledge in this step, mixed ontology will allow restrictions to be placed depending on how properties can be used by instances of a class. For example, we would say that a person has exactly one ID number, and that a seminar is presented by at least one presenter. Based on the restrictions, the logical errors in the facts and their relations using the mixed ontology will be verified.

### H. Enhancing additional axioms and rules

Either additional axioms or domain rules will be identified from the domain knowledge. In order to define formal concepts that are always true, a set of axioms (e.g., *subClassOf*, *equivalentClass*, *sameAs*, *differentFrom*, *TransitiveProperty*) are used in the mixed ontology. A new class of mixed ontology can be built from the existing components (class, properties, and individual) by fitting them together into the definitions. In the case of domain rules, they are represented in the form of an '*If-Then*' structure. Domain experts can define various domain rules depending on the types of domains. These additional axioms or domain rules are then utilized to check logical correctness and infer additional knowledge by reasoning.

### I. Completion of the Ontology

The final step is to build a domain ontology in an ontology language, such as OWL. To do this task we use Protégé, which is an ontology building tool that implements the conceptual models designed in the previous steps into OWL auto- matically.

## IV. Example Scenario for MOBM

In this section, we explain the process of constructing an ontology in an imaginary scenario through applying MOBM. The example company in our scenario is drawn up based on the well-known COMPANY database schema in the Elmasri/ Navathe Book [13]. Fig 5 shows the relational database schema that would be extracted into the kernel ontology.

**Step 1**: The following sub-steps summarize the process of extraction of the kernel ontology from the database schema information in Fig. 5 (this step corresponds to the mapping rules in Fig. 4).

(1) Recognize the tables in the database schema that cannot be a class.
  1.1 Recognize WORKS_ON which represents the M:N relationship.
  1.2 Recognize DEPT_LOCATIONS which treats a multi-valued attribute.

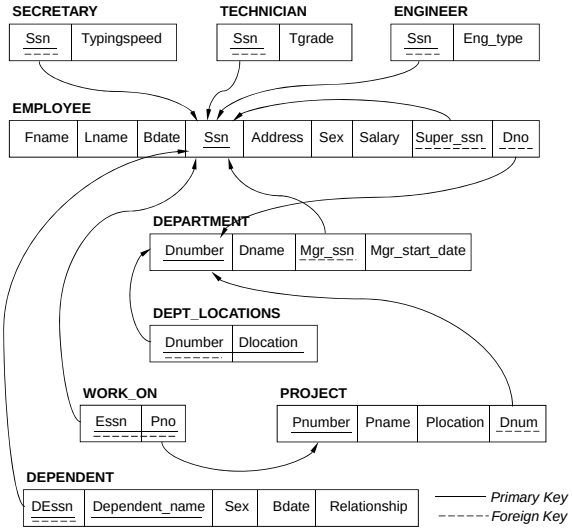


Fig. 5. The Relational Database Schema Diagram [13]

(2) Map the other 7 tables, EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT, SECRETARY, TECHNI CIAN and ENGINEER on to each ontology class.

(3) Set up the properties between the recognized classes as follows:
  3.1 Set up the subclass-relations.
    - EMPLOYEE-SECRETARY
    - EMPLOYEE-TECHNICIAN
    - EMPLOYEE-ENGINEER
  3.2 Set up the referential integrity Object properties and identify their domains and ranges for the ontology.
    - Super_ssn, Dno for EMPLOYEE
    - Mgrssn for DEPARTMENT
    - Dnum for PROJECT, DEssn for DEPENDENT
    For example, Object property Dno has EMPLOYEE as its domain and DEPARTMENT as its range.
  3.3 Set up the is_part_of the Object property between EMPLOYEE and DEPENDENT
    where DEPENDENT represents the weak entity.

(4) Set up the inverse Object property between EMPLOYEE and PROJECT that represents the M:N relationship.

(5) Identify Dlocation, which is not the FK of DEPT_ LOCATION, as the Datatype property of DEPART MENT.

(6) Identify the other non-FK attributes in each class as the Datatype properties of the ontology.
  - Ssn, Fname, Lname, Bdate, Address, Sex, Salary of EMPLOYEE
  - Dnumber, Dname, Mgr_start_date of DEPARTMENT
  - Pnumber, Pname, Plocation of PROJECT
  - Dependent_name, Sex, Bdate, Relationship of DEPENDENT
  - Typingspeed of SECRETARY
  - Tgrade of TECHNICIAN
  - Eng_type of ENGINEER

Fig. 6 describes the extracted kernel ontology based on the mapping rules.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

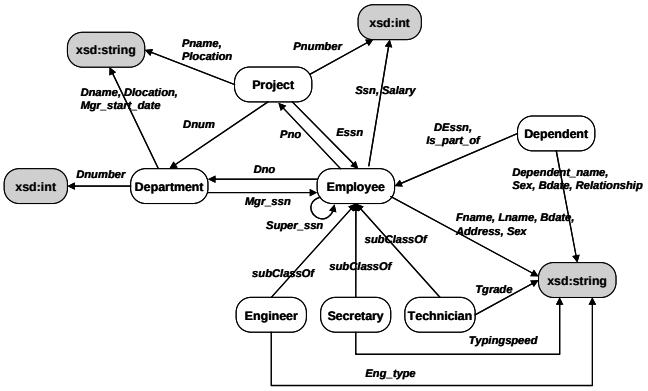


Fig. 6. The Kernel Ontology

**Step 2**: The terms that can be defined as the upper concepts of the kernel ontology are extracted from the domain knowledge. They are Company Entity, Task, People, Product, Owner, Project Member, and so forth. It is then verified that Company Entity includes the other terms from the extracted upper concepts. Hence, they are defined as the subclasses of *Company Entity*. As we can see from Part a) in Fig. 7, *People* is the subclass of *Company Entity*. In addition, *Employee* and *Dependent* in the kernel ontology, and the extracted terms, *Owner* and *Project Member* are defined as the subclasses of *People*. The subclasses of *People* are in a sibling relationship.

**Step 3**: After collecting the terms, defined as the lower concepts of the kernel ontology, from the domain knowledge and the instances of the database, we select only the terms that can be defined as a class. As you can see from Part b) in Fig. 7, *Headquarters*, *Research,* and *Administration* which are the attribute values of *Department* become a single class, respectively. This is because *accounting*, *marketing*, *finance* and *human_resource*, which were collected from domain knowledge, are identified as the individuals of *Administration.*

**Step 4**: In this step, we construct a mixed ontology by linking the classes developed in Step 2 and Step 3. Fig. 7 shows the parts of the mixed ontology built by connecting the upper ontology and the lower ontology to the kernel ontology.

**Step 5**: *Department*, *People*, *Product,* and *Task* which are the subclasses of *Company Entity*, are set up to be *disjoint-relation* as the semantics of the class hierarchy. We also indicate that the individuals in each class cannot be the same because the individuals might be instances of all the classes owing to Open World Assumption. The next semantics, additional relations, such as *Participate-relation* between class *Project Member* and class *Project*, are then added.

**Step 6**: The restrictions collected from the domain knowledge are added to the ontology. For example, the restriction saying that '*Department* has only and at least one *Mgrssn-relation* to *Employee*' can be expressed as follows.

$$Department \equiv \forall\ Mgrssn\ Employee \cap \exists\ Mgrssn\ Employee$$

**Step 7**: Additional axioms and rules for the mixed ontology are defined in this step. For instance, the axiom representing 'Project Member is People and has at least one Pno-relation to Project' can be expressed as follows.

$$Project\ Member \equiv People \cap \exists\ Pno\ Project$$

After defining axioms like this one, we are able to infer a new class hierarchy using the reasoner. In other words, *Project Member* can be inferred as a subclass of *Employee*, because *Employee* is defined as the domain of *Pno*.

**Step 8**: We use Protégé to realize the mixed ontology defined by MOBM in the OWL form. Using Protégé, we can easily define Class, Object property, Datatype property, Restriction, Axiom, and so on. We can also see the ontology information in a variety of forms through the plug-ins offered by Protégé. Fig. 8 shows the mixed ontology hierarchy using a plug-in called OntoGraf.

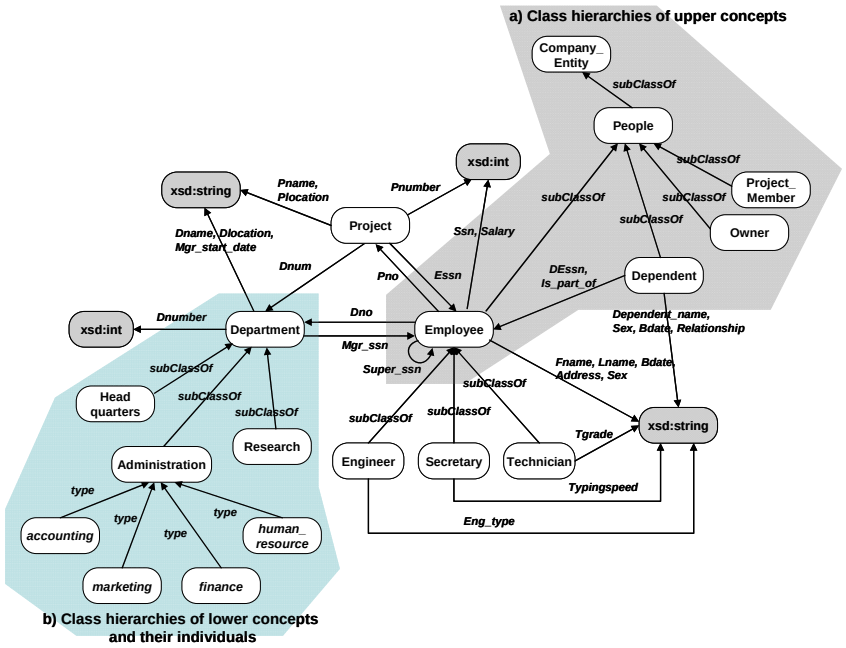


Fig. 7. The Mixed Ontology

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

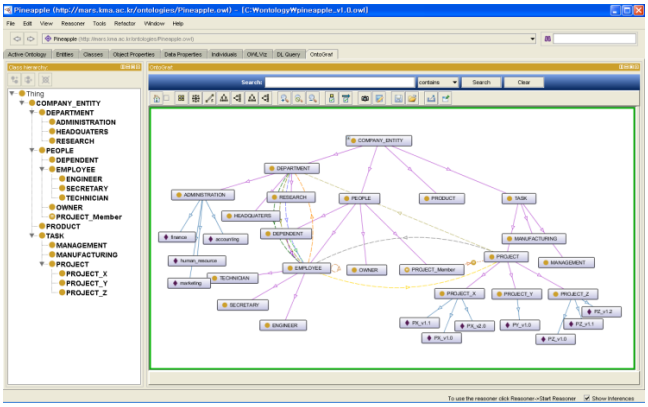

Fig. 8. Completion of Mixed Ontology using Protégé

## V. Lessons Learned

So far, we have applied the MOBM to the scenario for building an ontology, and the following advantages were analyzed.

- Core classes needed to build an ontology can be clearly extracted.
- Object properties that are the relation between classes can be easily defined by using the table relationship in database schema.
- Datatype properties also can be easily defined by using the attribute information of tables in the database schema.
- The method makes possible the gathering of information quickly on the core concepts and relations that ontologies consist of, so the time that takes to build the entire ontology can be reduced.
- In contrast to the ontology built with only database information, it is possible to build an ontology with richer semantics since the information collected from the domain knowledge is added to the kernel ontology.

However, MOBM has certain weaknesses still to improve, and they can be summarized as follows:

- If there is a domain that has a small number of tables in the database schema, the number of classes that the kernel ontology can have is limited. In this case, the effectiveness to be attained through building the initial ontology may be reduced. In MOBM, the greater number of the tables there are in the database produces more effectiveness.

## VI. Conclusion

In this paper we proposed the MOBM, which first extracts the kernel ontology by using database information as much as possible. It then completes the additional parts of the ontology by applying the top-down method and the bottom-up method respectively. In order to show the application possibilities, we applied this methodology to the example company database and explained the process for building ontology. From this application, we analyzed the advantages of the proposed methodology and identified the considerations for further application.

Future work will be continued in the direction of applying the proposed methodology to real world domains. We will develop a military ontology using MOBM and apply it to the Defense Information System, and investigate and conclude the degree to which this ontology is useful.

## References

[1] T. Gruber, "A Translation Approach to Portable Ontology Specifications," *Knowledge Acquisition*, Vol. 5, No. 2, 1993, pp. 199-220.

[2] D. Gašević, D. Djurić, V. Devedžić, "Model Driven Engineering and Ontology Development," Second Edition, Springer, Berlin, 2009.

[3] M. Grüninger, M. S. Fox, "Methodology for the design and evaluation of ontologies," *In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing held in conjunction with IJCAI-95*, Montreal, Canada, 1995.

[4] P. E. van der Vet, N. J. I. Mars, "Bottom-Up Construction of Ontologies," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 10, No. 4, 1998, pp. 513-526.

[5] G. Schreiber, B. Wielinga, W. Jansweijer, "The KACTUS view on the 'O' Word," *In IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Canada*, 1995, pp. 159-168.

[6] O. Corcho, M. Fernández-López, A. Gómez-Pérez, A. López-Cima, "Building legal ontologies with METHONTOLOGY and WebODE," *In Proceedings of Law and the Semantic Web*, 2003, pp. 142-157.

[7] F. J. Lopez-Pellicer, L. M. Vilches-Blázquez, J. Nogueras-Iso, O. Corcho, M. A. Bernabé, A. F. Rodríguez, "Using a hybrid approach for the development of an ontology in the hydrographical domain," *In Proceedings of 2nd Workshop Ontologies for Urban Development: Conceptual Models for Practitioners*, 2007.

[8] J. Trinkunas, O. Vasilecas, "Building Ontologies from Relatinoal Databases Using Reverse Engineering Methods," *In Proceedings of International Conference on Computer Systems and Technologies*, 2007.

[9] Z. Xu, S. Zhang, Y. Dong, "Mapping between Relational Database Schema and OWL Ontology for Deep Annotation," *In Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 2006, pp. 548-552.

[10] N. Konstantinou, D. E. Spanos, N. Mitrou, "Ontology and Database Mapping: A Survey of Current Implementations and Future Directions," *Journal of Web Engineering*, Vol. 7, No. 1, 2008, pp. 1-24.

[11] N. Cullot, R. Ghawi, K. Yétongnon, "DB2OWL: A Tool for Automatic Database-to-Ontology Mapping," *In Proceedings of the 15th Italian Symposium on Advanced Database Systems (SEBD 2007)*, Italy, pp. 491-494.

[12] S. S. Sane, A. Shirke, "Generating OWL Ontologies from a Relational Databases for the Semantic Web," In Proceedings of International Conference on Advances in Computing, Communication and Control, 2009, pp. 143-148.

[13] R. Elmasri, S. B. Navathe, "Fundamentals of Database Systems," Sixth Edition, Addison Wesley, 2010.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)