An Approach to Constructing an Integrated Ontology for Integrating Object-Oriented Data Models

Manachaya Jamadhvaja* , Twittie Senivongse**

ABSTRACT

Semantically determining similarities and differences between data from multiple data sources plays an important role in the process of data integration. The similarities and differences identify the degree of relationships and are used for constructing the integrated data model. In structured data such as object-oriented database, their data models reveal only structure-based semantics without other additional semantics. This paper proposes an approach to integrate two object-oriented data models based on the integration of ontologies that describe their semantics. The ontology for an object model describes semantics based on the model structure and additional semantics, i.e. class and attribute semantics, and synonym and hypernym lists. The ontologies are analysed to devise an integrated data model that will be used to facilitate query on the integrated data.

KEYWORD : Ontology Integration, Data Model Integration, Ontology, Object-Oriented Data Model.

1. INTRODUCTION

The growth of information technology affects the use of data in information systems. In organisations, using a single data source is not much enough to run the business. Hence, data model integration is required for combining heterogeneous data from different data sources and for providing users with a single point of access to all data sources [9]. The data models that are integrated tend to be in the same context but are designed by different owners (e.g. the data models represent the population databases of different departments in a district). This will result in several kinds of semantic heterogeneity, for example, name conflict (the same information represented by different names or different information represented by the same name), structure conflict (e.g. same information represented by different data types), and scaling conflict (e.g. an attribute 'salary' with different currencies in different data models) [14]. To integrate two data models, a database integrator must understand well the meaning of both data models and can identify the similarities and differences in the semantics that underlies the data models.

For structured data such as object-oriented databases, structure-based semantics are represented as shown in UML class diagram in Figure 1. UML class diagram describes the semantics of an object-oriented data model in terms of classes and attributes that form the model as well as the relationships between them, but it does not reveal any other meaning inside the data.

The use of ontologies for describing the meaning inside the data is a practical method to solve the problem of semantic heterogeneity. Ontology, an explicit specification of a conceptualization [13], represents knowledge in a domain of interest in the form of a set of concepts (or terms) and describable relationships between them. W3C has recommended Web Ontology Language (OWL) for publishing and sharing ontologies [11]. OWL is developed as a vocabulary extension to RDF (Resource Description Framework), so one of the representations of OWL is RDF graph [1]. By using ontological concepts to represent semantics in the data models, shared vocabularies are defined to illustrate both structure-based semantics and other additional semantics in the models. As a result, ontologies will handle the heterogeneity and distribution across all data sources, and be the mediator between users and data sources. Ontologies will facilitate flexible semantics-based query on the data sources. Several research work deals with the use of ontologies for integrating heterogeneous data (e.g. SIMS [15], OBSERVER [5], COIN [2] and MECOTA [7]).

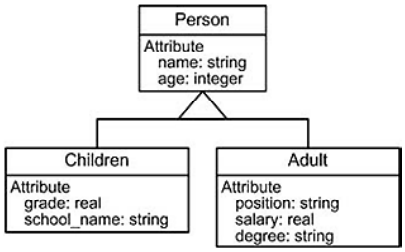

Fig. 1. Example of object-oriented data model

This paper proposes an approach to integrate two object-oriented data models whose semantics are described by ontologies. Since there is a mapping between object model and ontology, we follow such a suggestion from DSTC's Ontology Definition MetaModel [4] to develop an ontology integration approach by adapting from the methodology for integrating object-oriented data models in [12]. All classes from the two data models are analysed in order to determine the degree of relationships between them before the integration can be applied. Several kinds of relationships between ontological concepts within the models are considered, i.e. inheritance, aggregation, sibling, equivalence, synonym, and hypernym. The degree of similarities and differences between the ontologies are identified by heuristics-based scoring.

In Section 2, we review related work and discuss a comparison to our research. Our ontology integration method in Section 3 is described with an example of the integration of the object-oriented data models of two population databases. Finally, Section 4 discusses the use of this approach and concludes the paper.

2. RELATED WORK

2.1 Ontology-based information integration

Several research work suggests ways to use ontologies for heterogeneous data integration. In summary, all research can be categorised into three approaches [8]:

1. Single Ontology Approach: One global ontology is used to describe all data sources (Figure 2(a)), and query on the integrated data is via the global ontology. This approach

*Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University
**Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University

looks straightforward but needs an expert who knows the meaning of all data sources to define the global ontology (e.g. SIMS [15])

2. Multiple Ontology Approach: Each data source is conveniently described by its own local ontology but a mapping between the local ontologies has to be established (Figure 2(b)). Query on the integrated data is via a local ontology and the mapping is used to compose local queries on other local ontologies. This approach looks more adaptive but mapping local ontologies each time there is a query is an overhead (e.g. OBSERVER [5]).

3. Hybrid Ontology Approach: This combination of the first two approaches describes each data source with a local ontology and a shared ontology is built to for sharing vocabularies among local ontologies (Figure 2(c)). This approach takes the advantages of the two approaches: ease of defining ontologies locally and of querying via the shared ontology (e.g. COIN [2] and MECOTA [7]). Our research follows this hybrid approach.

2.2 Ontology integration

Most of research in ontology integration presents techniques for integrating ontologies which describe knowledge about unstructured or semi-structured data such as Web pages. By nature of such data, it is unavoidable that the proposed algorithms are complex (e.g. natural language processing is employed). For example, OBSERVER [1] presents a method that performs the translation from a local ontology into other local ontologies for cross-ontology query; no integrated ontology is actually created. On the contrary, FCA-MERGE [6], PROMPT [10], and Chimaera [3] propose ideas for constructing integrated ontology. FCA-MERGE [6] applies mathematical techniques to derive concepts of ontology, and produces suggestions for transforming into an integrated ontology. PROMPT [10] and Chimaera [3] are interactive tools that can perform some integration tasks automatically and guide the ontology integrator in performing other tasks.

Our research focuses on integrating two object-oriented data models, so it is assumed that the semantics of these models are more structured and the algorithm proposed here should be less complex.

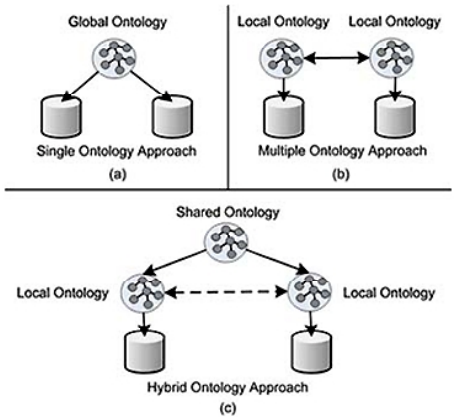

Fig. 2. Ontology-based approaches for information integration [8]

3. INTEGRATING ONTOLOGIES

For simplicity and due to limited space, the proposed approach to integrating ontologies for object-oriented data models will be explained through an example of the integration of two simple population data models in Figures 1 and 3. First, the semantics of these two models will be represented as two local ontologies based on the guideline in [4] (Section 3.1). Then they are compared in Section 3.2 and integrated in Section 3.3 using the methodology adapted from [12].

Our approach gains semantic richness of ontology and describe both the semantics that is already present in the data models (i.e. class, attribute, inheritance, aggregation, sibling) and the other additional semantics (i.e. other class and attribute semantics, equivalence, synonym, hypernym).

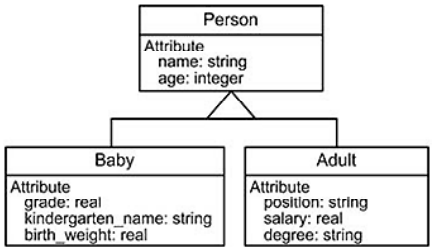

Fig. 3. Another example of object-oriented data model

3.1 Using upper ontology to derive local ontologies

Upper ontology (Figure 4) is a meta-ontology from which the local ontologies of the two object-oriented data models will be derived (OWL ontology here is represented in RDF graph). It explains what semantics should be described for the data models.

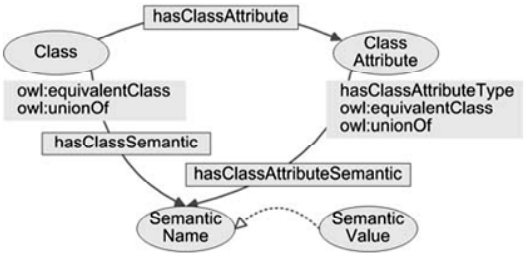

Fig. 4. Upper ontology

From the upper ontology, the semantics of the object-oriented data model are classified into two types: structure-based semantics and additional semantics. Structure-based semantics defines the semantics that is already present in the object-oriented data models. We use two ontology classes, i.e. "Class" and "ClassAttribute", to represent object-oriented class and attribute respectively. The ontology object property "hasClassAttribute" is used to describe the property of a class having an attribute, and the ontology datatype property "hasClassAttributeType" is used to describe data type of an attribute. Additional semantics defines other semantics that a data model integrator may add to the ontology for the purpose

of integration. According to the upper ontology, the additional semantics can be added to both OO classes and attributes in two ways:

1. By using "hasClassSemantic" or "hasClass Attribute Semantic" property for class semantics and attribute semantics, the semantics will be defined as "SemanticName" and "Semantic Value". For example, in Figure 5, we define semantics of Children as "ChildrenAgeSemantic" with value "≥ 0" and "≤ 20".

2. By using "owl:equivalentClass" and "owl:unionOf" property for both class and attribute, class semantics and attribute semantics will be defined as what term in a local ontology is a synonym or hypernym of what term in the other local ontology. For example, in Figure 5, we define a synonym of "ChildrenSchoolName" as "owl:equivalentClass BabyKindergartenName" to show that their meanings are the same.

The semantics of the two models in Figures 1 and 3 will be represented as local ontology 1 (upper side) and 2 (lower side) in Figure 5 respectively. They are derived from the upper ontology in the middle.

3.2 Ontology comparison

Ontology comparison is based on class comparison to find out the relationships between object-oriented classes, which are represented by the ontology class inherited from "Class" in the upper ontology. Class comparison (Section 3.2.3) is a bottom-up process, resulting from semantic comparison (Section 3.2.1) and ClassAttribute comparison (Section 3.2.2). Firstly, each pair of the classes from the two local ontologies is picked one by one. Secondly, class semantic comparison and ClassAttribute comparison are performed respectively. Finally, the relationship that this pair of classes has with each other is returned by class comparison. The process of ontology comparison is depicted in Figure 6.

3.2.1 Semantic comparison

Semantic comparison is based on some heuristics that is derived from an experiment on the integration of ten pairs of OO data models and from a data integration expert [11]. It gives a numeric value that tells the degree of relationship between two terms that are compared. Let T1 and T2 be the terms in local ontology 1 and 2 respectively whose semantics are to be compared. α is the number of SemanticName of T1, β is the number of SemanticName of T2, and $\alpha \leq \beta$.

Each pair of "SemanticName and SemanticValue" will be compared and the heuristic value will be returned according to Table 1. The result of each semantic pair will be gathered in a set $R = [r_1, r_2, \ldots, r_\alpha]$, and $Ms = \Sigma r_i$ where $r_i \in R$ and $i = 1, \ldots, \alpha$ is computed. Table 2 shows the kind of relationship that T1 has with T2.

3.2.2 ClassAttribute comparison

ClassAttribute comparison is based on the comparison of ClassAttribute and the comparison of the set of ClassAttribute.

1. ClassAttribute comparison is a process to compare each pair of ClassAttribute in a class. Each pair of Class Attribute will be compared and the heuristic value will be returned according to Figure 7.

Table 1. Result of comparison of each semantics of T1 with that of T2

Semantic name	Semantic values	r
Same	Same	1
Same	Subset	0.8
Same	Overlap	0.8
Same	Disjoint	0.5
Differrent	Any	0

Table 2. Relationships of Terms with Semantic Consideration

Ms	Result	Relationship	s
α ($\alpha = \beta$)	Equivalence	Equivalence	1
α ($\alpha < \beta$)	Inclusion	Superclass*	1
≥¾α ($\alpha = \beta$)	Equivalence	Equivalence	1
≥¾α ($\alpha < \beta$)	Inclusion	Superclass*	1
≥½α	Tight-intersection	Sibling	0.1
<½α	Lose-intersection	Disjoint	0
0	Disjoint	Disjoint	0

*T1 is Superclass of T2 = T2 is Subclass of T1

2. Set of ClassAttribute comparison is a process to compare a set of ClassAttribute of a class to another one. Set of ClassAttribute comparison is also based on heuristics scoring from [11]. The heuristics value tells the degree of relationship between two classes that are compared. Let C1 and C2 be the classes in local ontology 1 and 2 respectively whose ClassAttributes are to be compared. θ_{C1} is the number of ClassAttribute of C1, θ_{C2} is the number of ClassAttribute of C2. Let $\alpha = \theta_{C1}, \beta = \theta_{C2}$, and $\alpha \leq \beta$. If a ClassAttribute in C1 has "owl:unionOf" property with other group of ClassAttribute in C2, then $\alpha = \alpha - 1 +$ number of ClassAttribute in the group. And if a ClassAttribute in C2 has "owl:unionOf" property with other group of ClassAttribute in C1, $\beta = \beta - 1 +$ number of ClassAttribute in the group.

Each pair of ClassAttribute in C1 and C2 will be compared using the method in the ClassAttribute comparison process. The result of each ClassAttribute pair will be gathered in a set $S = [s_1, s_2, \ldots, s_\alpha]$, and $Ms = \sum s_i$ where $s_i \in S$ and $i = 1, \ldots, \alpha$ is computed. Table 3 shows the kind of relationship that C1 has with C2.

Table. 3 Relationships of classes C1 and C2, with consideration on the set of ClassAttribute

Ms	Result	Relationship
α ($\alpha = \beta$)	Equivalence	Equivalence
α ($\alpha < \beta$)	Inclusion	Superclass*
≥¾α ($\alpha = \beta$)	Equivalence	Equivalence
≥¾α ($\alpha < \beta$)	Inclusion	Superclass*
≥½α	Tight-intersection	Sibling
<½α	Lose-intersection	Disjoint
0	Disjoint	Disjoint

*C1 is Superclass of C2 = C2 is Subclass of C1

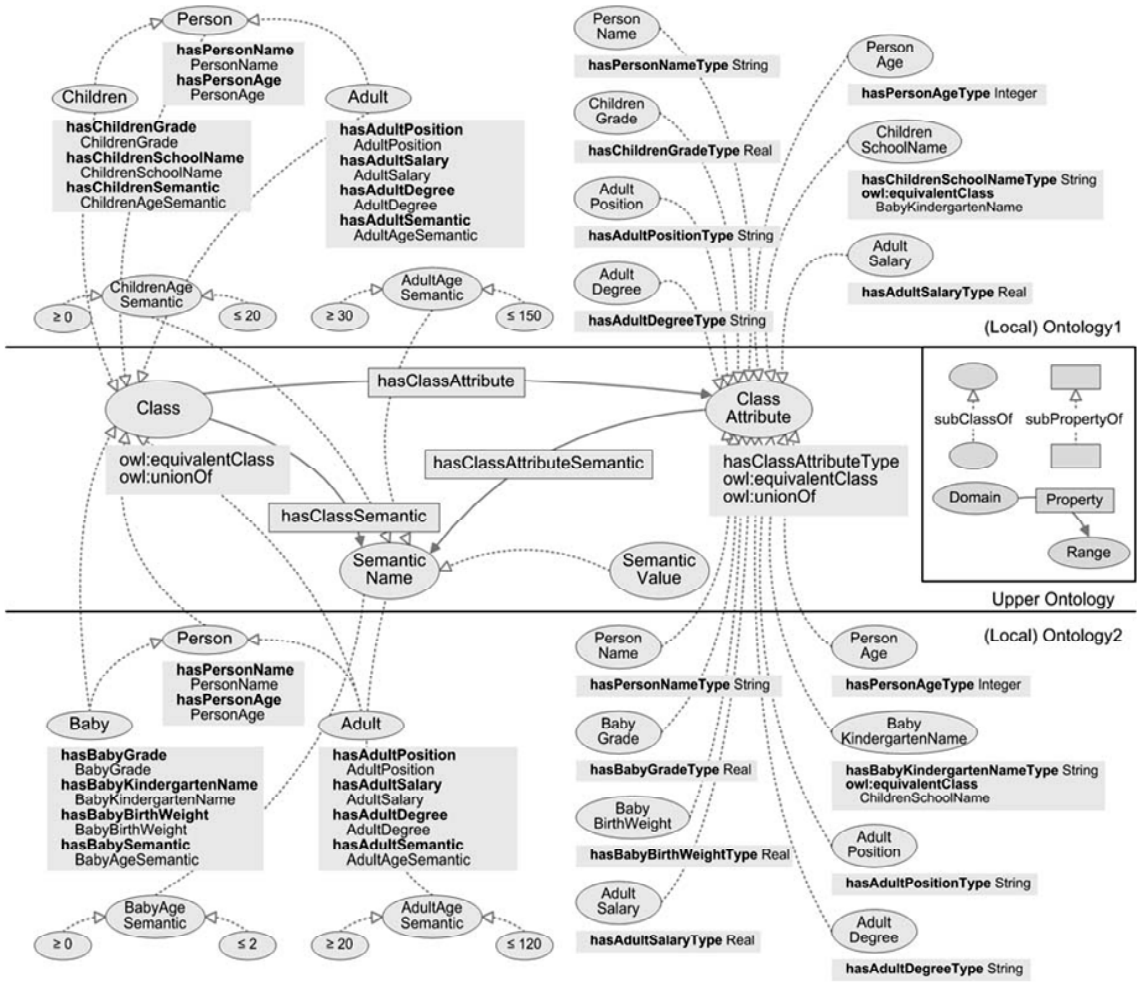

Fig. 5. Local ontologies

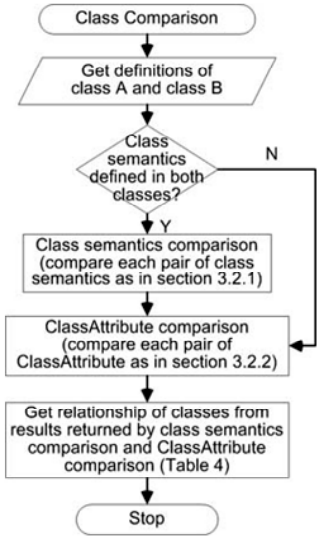

Fig. 6. Class comparison

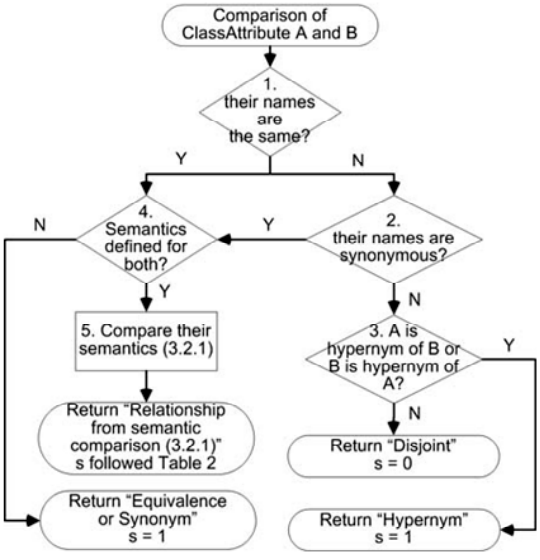

Fig. 7. ClassAttribute comparison

3.2.3 Class comparison

The relationship that a class C1 has with a class C2 is based on the relationship between their class semantics and the relationship between all of their ClassAttributes as a whole. From the method in section 3.2.1 and 3.2.2, we can obtain Relationship of ClassSemantic which indicates the kind of relationship between all of their class semantics and Relationship of the set of ClassAttribute which indicates the kind of relationship between a particular pair of their attributes. In this process, we can obtain the relationship that C1 has with C2 from Table 4.

Figure 8 shows the comparison of classes 'Children' and 'Baby' from Figure 5. The comparison follows the steps in section 3.2.1, 3.2.2 and 3.2.3. The result is that 'Children' has ClassSemantics and the set of ClassAttribute that are superclass of those of 'Baby', and therefore 'Children' is superclass of 'Baby'. Other pairs of classes in the two ontologies will be compared in a similar manner.

3.3 Ontology integration

Ontology comparison results in the relationship between each pair of the classes that are compared being identified (i.e. equivalence, superclass, subclass, or sibling). From the result, we can construct a new integrated ontology based on the resulting relationships. Table 5 shows the result of the comparison between pairs of the classes in our example. The class relationships in this table lead to the creation of the integrated ontology. Figure 9 presents the final integrated ontology of our population example which depicts 'Person' as the root class with 'Children' and 'Adult' as its subclass whereas 'Baby' also inherits from 'Children'.

Table 4: Summary of class relationships

Result of Class Semantic comparison	Result of set of Class Attributes comparison	Relationship
• Equivalence • Superclass • Subclass • Someone*	Equivalence	Equivalence
• Equivalence • Superclass • Subclass • Someone*	Superclass	Superclass
• Equivalence • Superclass • Subclass • Someone*	Subclass	Subclass
• Equivalence • Superclass • Subclass • Someone*	Sibling	Sibling
• Disjoint	Sibling	Sibling

*Either or none of the two classes have ClassSemantics

Semantic comparison

Ontology1.Children.Age has Ontology2.Baby.Age as a subset
$\therefore$ R = [0.8], Ms = 0.8, $\alpha = 1$, $\beta = 1$ Result = Equivalence **Relationship = Equivalence**

Class Attribute comparison

Ontology1.Children.PersonName has no additional semantics but has the same name as Ontology2.Baby.PersonName
$\Rightarrow$ return Equivalent s = 1
Ontology1.Children.PersonAge and Ontology2.Baby.PersonAge have class semantics that are subset and also have the same name
$\Rightarrow$ return Superclass s = 1
Ontology1.Children.ChildrenGrade has no additional semantics but has the same name as Ontology2.Baby.BabyGrade
$\Rightarrow$ return Equivalent s = 1
Ontology1.Children.ChildrenSchoolName and Ontology2.Baby.BabyKindergartenName have no additional semantics but they are synonymous
$\Rightarrow$ return Synonym s = 1
$\therefore$ $Ms_{SetOfClassAttribute}$ = 1+1+1+1 = 4, $\alpha = 4$, $\beta = 5$ Result = Inclusion **Relationship = Superclass**

Class comparison

$\therefore$ From Table 4, Relationship of ClassSemantic + Relationship of set of ClassAttribute
= Equivalence + Superclass = Superclass

Fig. 8. Comparison between class Children in local ontology 1 and class Baby in local ontology 2

Table. 5 Result of the comparison between local ontology 1 and local ontology 2

	Ontology1.Person	Ontology1.Children	Ontology1.Adult
Ontology2.Person	Equivalence	Superclass	Superclass
Ontology2.Baby	Subclass	Subclass	Sibling
Ontology2.Adult	Subclass	Sibling	Equivalence

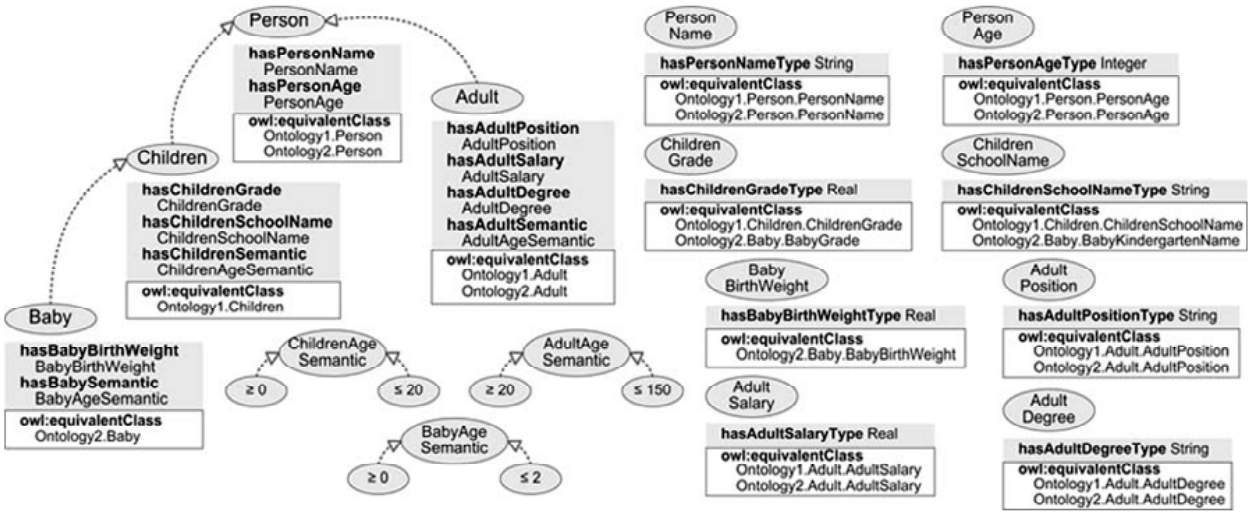

Fig. 9. Integrated ontology

4. CONCLUSION

From ontology integration, all resulting classes in the integrated ontology will be mapped to the local ontologies. A query through the integrated ontology will be translated as queries on the local ontologies of the underlying databases. Users can also benefit from describing semantics by ontology because query can then be semantics-based by the inference power of ontology. For example, a query for the data of all children will result in the data of children from data model 1 and of baby from data model 2 to be returned.

We are in the process of refining the ontology integration approach. We expect more kinds of semantics to be included in the local ontologies, such as integrity constraints on attributes. The prototype of the ontology integration and query framework is also in progress.

5. REFERENCE

[1] B. McBride, "An Introduction to RDF and the Jena RDF API," vol. 2005: http://jena.sourceforge.net/tutorial/RDF_API/ index.html, 2005.

[2] C. H. Goh, et al, "Context Interchange: New Features and Formalisms for the Intelligent Integration of Information," ACM Transactions on Information Systems (TOIS), vol. 17, pp. 270-293, 1999.

[3] D. L. McGuinness, et al, "An Environment for Merging and Testing Large Ontologies," Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, Colorado, 2000.

[4] DSTC, "Ontology Definition MetaModel – DSTC Initial Submission," submit to Object Management Group, 2003.

[5] E. Mena, et al, "OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies," Distributed and Parallel Databases, vol. 8, pp. 223-271, 2000.

[6] G. Stumme and A. Maedche, "FCA-MERGE: Bottom-Up Merging of Ontologies," Proceedings of 17th International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, 2001.

[7] H. Wache, et al, "An integration method for the specification of rule-oriented mediators," Proceedings of 1999 International Symposium on Database Applications in Non-Traditional Environments (DANTE '99), 1999.

[8] H. Wache, et al, "Ontology-Based Integration of Information — A Survey of Existing Approaches," Proceedings of 17th International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, 2001.

[9] M. Lenzerini, "Data Integration: A Theoretical Perspective," Proceedings of 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, Madison, Wisconsin, 2002.

[10] N. F. Noy and M. A. Musen, "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment," Proceedings of 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence, 2000.

[11] S. Bechhofer, et al, "OWL Web Ontology Language Reference," W3C Recommendation, vol. 2005, M. Dean and G. Schreiber, Eds., February 10, 2004.

[12] S. Swasrukkiet, "A Methodology for Integration of Object-Oriented Data Models Using Heuristics and Analysis of Relationships between Class," Master Thesis, Department of Computer Engineering, Chulalongkorn University, Bangkok, 1999, pp. 162.

[13] T. R. Gruber, "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition (1042-8143), vol. 5, pp. 199 - 220, 1993.

[14] W. Sull and R.L. Kashyap, "A Self-Organizing Knowledge Representation Scheme for Extensible Heterogeneous Information Environment", IEEE Transactions on Knowledge and Data Engineering, vol. 4 no. 2, pp. 185-191, 1992.

[15] Y. Arens, et al, "Query Processing in the SIMS Information Mediator," in Readings in agents: Morgan Kaufmann Publishers Inc., pp. 82 - 90, 1998.