CHAPTER 4
ONTOLOGY-BASED METADATA DICTIONARY MODELING

This chapter proposes a metadata dictionary, the core component of SIGA, as a means for solving the semantic heterogeneity problem. First, the proposed metadata dictionary is modeled and designed based on the domain ontology (Gruber, 1993) to be a repository for storing conceptual level and physical level data descriptions to bridge the gap of heterogeneous sources into homogeneity. This work focuses on modeling and designing the domain ontology, which is the fundamental building block of the metadata dictionary instead of the straightforward ontology construction (Gruber, 1995; Jones, Bench-Capon and Visser, 1998; Uschold and Gruninger, 1996; Uschold, 1996). Next, the domain ontology which is an abstract representation of the proposed metadata dictionary structure has been extracted from the global conceptual schema to explicit representation by two levels, namely, the conceptual level of abstraction and the physical level of abstraction. The target output of the modeling is the proposed metadata dictionary that provides a mapping mechanism to associate user's requests posed at the conceptual level with the physical level, allowing direct access to stored information without loss of information in the query.

Finally, this chapter provides a means to manage the metadata dictionary contents to decrease the system development time. The design of metadata dictionary provides the scalability of the metadata dictionary when adding or dropping the physical source schemas.

4.1 Ontology-based Metadata Dictionary Modeling

Ontology-based metadata dictionary (Arch-Int, Sophatsathit and Li, 2003) has been modeled on the basis of a bottom-up design approach (Castano, Antonellis and Vimercati, 2001; Özsu and Valduriez, 1999, Vet and Mars, 1998). The objective of the modeling is to extract conceptual specifications from the existing physical information sources with the help of explicit representations. Extraction process of the ontology-based metadata dictionary by the domain ontology modeling is depicted in Figure 4.1.

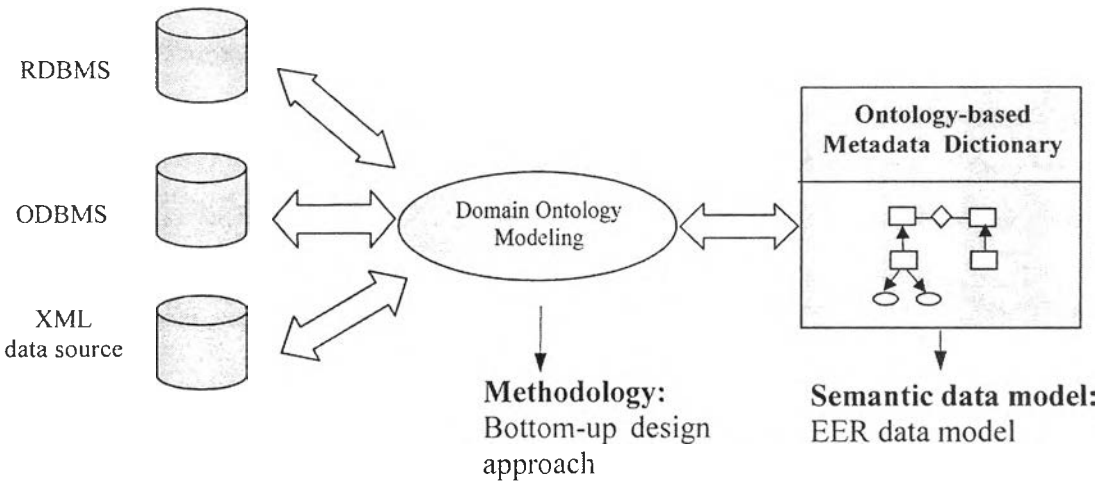


Figure 4.1 Extraction of the ontology-based metadata dictionary by domain ontology modeling.

The ontology modeling process consists of four steps, namely, schema translation, schema restructuring, schema integration, and ontology extraction. Details are elucidated in the sections that follow.

4.1.1 Schema Translation.

This step involves translating or mapping the underlying physical source schemas represented in various data models to intermediate schemas denoted by canonical data models. Due to the expressiveness of E-R model (Chen, 1976), this E-R model is adopted as the canonical model. In this canonical model, the physical source schemas are denoted by entities, attributes, relationships, and constraints. One difficulty is the determination of relations that represent entities and their relationships. One clue to identify the relationship of these entities is the presence of foreign key. Once the difficulty is solved, the relations that represent entities are modeled as entities, and relations that link these entities are modeled as relationships. Other modeling considerations are cardinality of the relationship (e.g., one-to-one, one-to-many, and many-to-many) and attributes such as primary keys and foreign keys.

4.1.2 Schema Restructuring

This step involves restructuring each intermediate schema to eliminate structural heterogeneity (Batini and Lenzirini, 1984; Batini, Lenzirini and Navathe, 1986; Özsu and Valduriez, 1999). This process employs the atomic conformation principles adapted from

Batini, Lenzirini and Navathe, 1986 to handle the structural heterogeneity. This technique transforms entities/attributes/relationships on an instance-by-instance basis. The atomic conformation principles are illustrated in Figure 4.2.

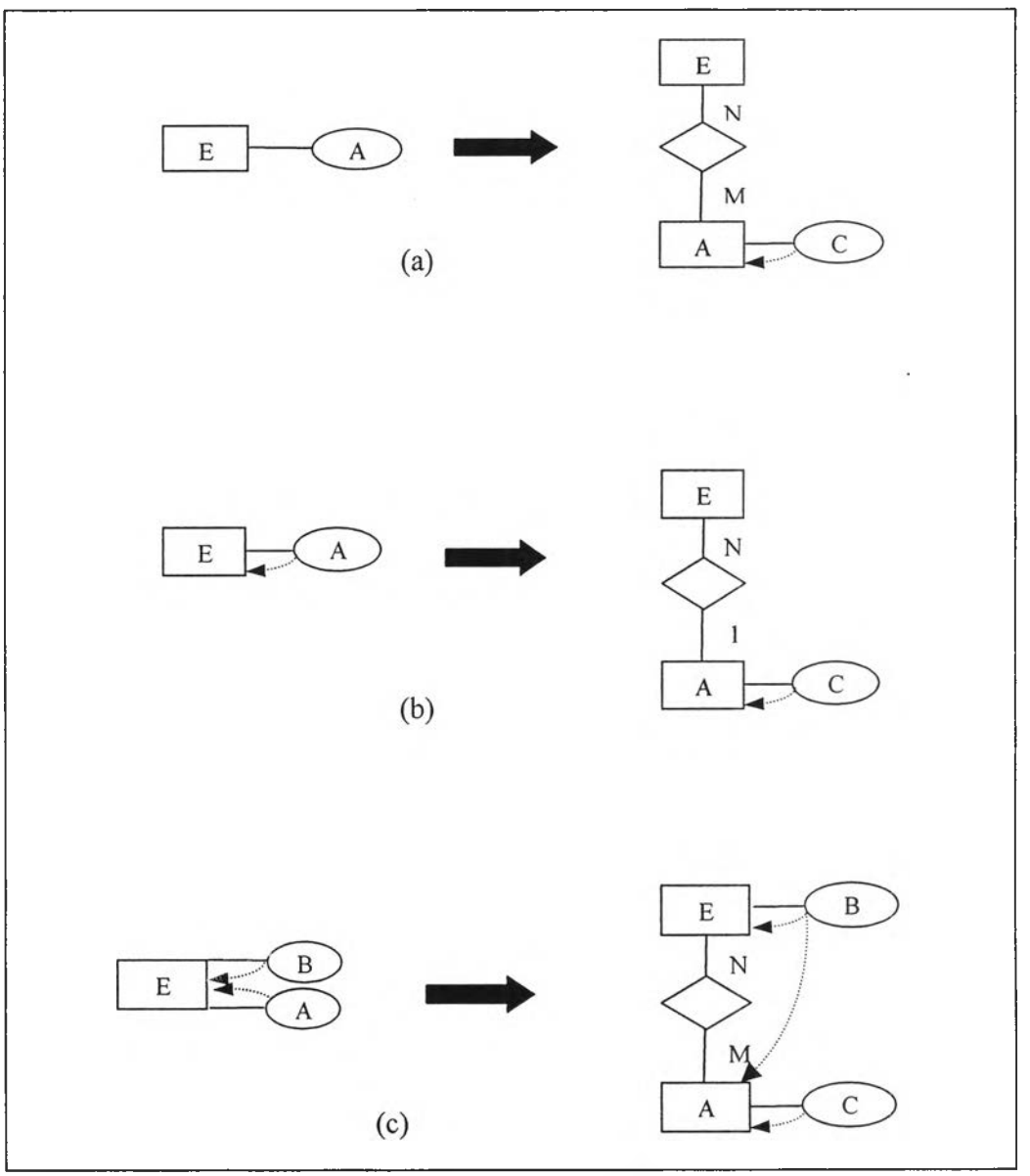


Figure 4.2 Atomic conformation principles (Özsu and Valduriez, 1999).

The dashed lines indicate that a given attribute is an identifier (key) of the associated entity. The results constitute the knowledge structure of the entire physical information sources. A non-key attribute can be transformed into an entity by creating an intermediate relationship connecting the new entity to a new attribute. Figure 4.2 (a) depicts such a

transformation of a non-key attribute A of entity E to a separate entity that is related to E by a many-to-many relationship which is uniquely identified by a new key attribute C. Figure 4.2 (b) illustrates a key attribute translation where a key attribute is transformed into an entity that has an identifier C. C becomes the identifier of both new entity A and the entity E because the relationship between E and A is many-to-one. Figure 4.2 (c) demonstrates the case where identifier A is only a part of the complete identifier, which requires the non-standard reference back to the originating entity.

4.1.3 Schema Integration

Schema integration combines all intermediate schemas into a global conceptual schema. This step is the process of identifying the components of an information source which are related to one another, selecting the best representation for the global conceptual schema, and integrating the components of each intermediate schema. Two components relate to each other as equivalent in which one contained in the other, or as disjoint (Özsu and Valduriez, 1999, Sheth, Larson, Cornellio And Navathe, 1988). The purpose of schema integration is to eliminate the generalization conflicts induced by the IS-A relationship between sub-type specific entities and the super-type general entity. The integration also applies to entities whose instances belong exclusively to an instance of another entity, that is, the component entities of an aggregate entity through the IS-PART-OF relationship.

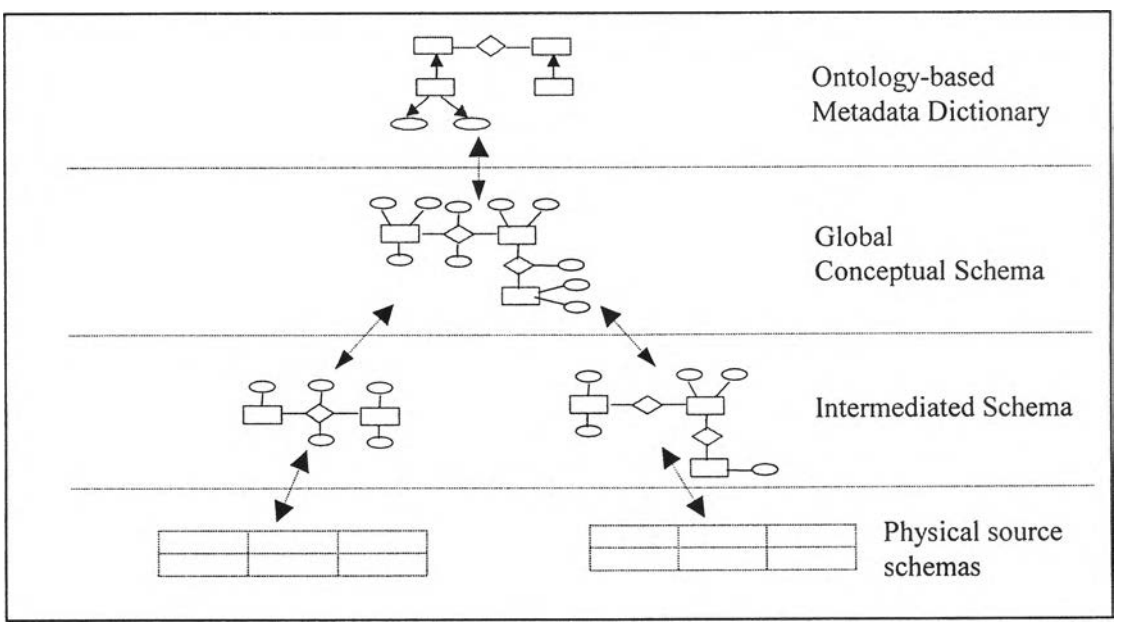


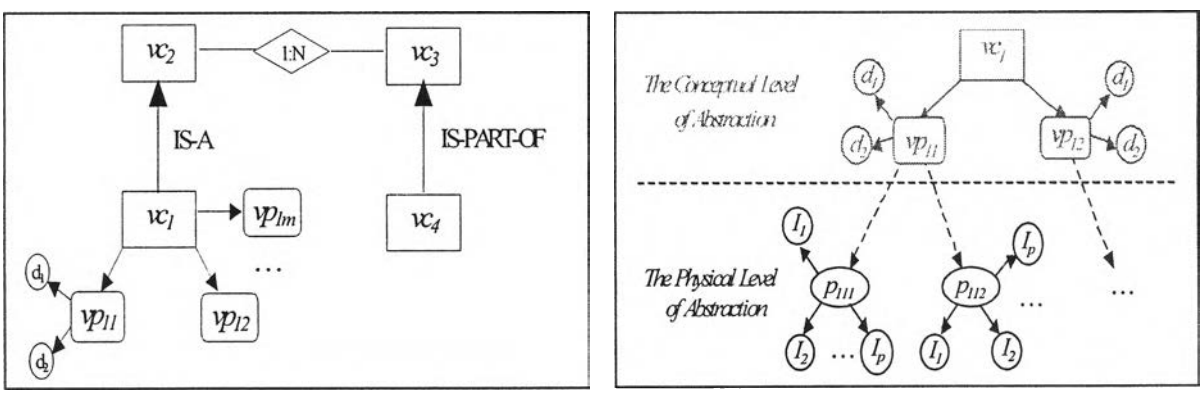
Figure 4.3 The relationship between ontology-based metadata dictionary and the underlying physical schemas.

The relationship between the ontology-based metadata dictionary and the underlying physical schemas is illustrated in Figure 4.3.

4.1.4 Ontology Extraction

This last step is the main contribution of our work by extracting ontology from the underlying global conceptual schema to obtain an explicit user-viewed representation. The ontology is systematically broken down explicitly to two levels of abstraction, namely, the conceptual level of abstraction and the physical level of abstraction.

(1) **The conceptual level of abstraction**. The global conceptual schema is restructured to virtual schema, which is an initial ontology represented by the Extended Entity-Relationship (EER) model encompassing virtual concepts (or entities), virtual properties (or attributes), relationships, and construction rules. The ontology conceptualized at this level abstracts the users from physical information sources. Users can pose their queries in the form of this ontology rather than dealing with real data. A partial internal structure of domain ontology at this level is depicted in Figure 4.4 (a).



(a) Domain ontology at the conceptual level of abstraction

(b) Domain ontology at the physical level of abstraction

Figure 4.4 Two levels of the domain ontology extracted from a global conceptual schema.

In this figure, boxes represent virtual concepts, whereas diamonds denote the relationships that hold among the virtual concepts. The virtual properties are shown as round-edged rectangles attached to each virtual concept. This level is designed to solve data type, scaling, and generalization conflicts. To eliminate data type and scaling

conflicts, a virtual property is designed as a class property which forms two domain properties, that is, the predefined type domain (e.g., integer, string, float, or char) and the scaling domain (or units of measure, e.g., kilogram, pound, US$, or AUS$). These domain properties are used to represent different physical data types and unit types from the HIS into a uniform format. Generalization conflicts are also eliminated through the IS-A relationships when connecting a specific concept to a general concept. The IS-PART-OF relationship is denoted by an arrow connecting a component concept to an aggregate concept. The construction rules are augmented from the diagram.

(2) **The physical level of abstraction**. This level provides a mapping mechanism to associate the virtual concepts and properties of a virtual schema with the corresponding physical concepts and properties of the global conceptual schema. A partial internal ontology structure is illustrated in Figure 4.4 (b). This level is designed to solve naming conflicts by designating each virtual property to hold its instances called physical instances, which is represented by ellipses. These physical instances store the synonymous physical property names of the physical concepts in a global conceptual schema. Each physical instance defines its own properties, denoted by circles that encompass other physical information corresponding to the physical instance, such as physical data type, unit type, concept, and source. The ontology on this level also holds physical source configurations describing the configurations of physical concepts in each physical source. These physical source configurations furnish necessary information to grant permission and knowledge for agents in accessing individual physical sources.

4.2 Metadata Dictionary Management

Currently, the metadata dictionary management involves the maintenance of the components of metadata dictionary and requires an understanding of the terminology and relationship between them. The management of metadata dictionary remains largely manual, that leads to both costly and laborious maintenance efforts. Beside, the matching of difference terminology by human may differ in interpretation.

In order to provide a means to manage metadata dictionary more efficiently and permit scalability when augmenting new schemas to the existing global conceptual schema,

the modeling process can be enhanced with the help of data management techniques to facilitate matching of different terminology between schemas and schema integration.

The logical architecture of metadata dictionary has been designed based on data management techniques consisting of five interacting components as illustrated in Figure 4.5. The responsibility of each component is described below.

(1) **Schema Engine.** The schema engine receives the existing signal in form of information package of new physical source from a new resource agent. The resource agent passes the schemas of augmented physical source to the schema engine where initial ontology extraction is initiated. The schema engine uses software reverse engineering techniques (Cui and O'Brien, 2000; Yang H. and Bennett, 1995; Yang, Cui and O'brien, 1999) to extract entities, attributes, and relationships between entities, as well as primary keys and foreign keys. The extracted information is then used as the ontology verification and refinement by the ontology administrator to ensure its validity with the help of ontology validation tools (Cui and O'Brien, 2000). A graphical user interface is provided to communicate with the ontology administrator for viewing, refinement, and confirmation of the extraction.

(2) **Ontology Integration Engine.** The ontology integration engine is used to load the current global conceptual schema in order to augment it with new schema derived from the schema engine. To facilitate the integration process, the ontology administrator employs the heterogeneous schema integration tool proposed by (Miller, Hernandez, Haas, Yan, Ho, Fagin, and Popa, 2001; Reddy, Prasad, Reddy and Gupta, 1994; Sheth, Larson, Cornellio and Navathe, 1988) to verify the schema representation. This tool is used to edit or further augment the generated global conceptual schema. In addition, the tool also provides a data view mode in which the administrator can browse through sample data from the schemas for better understanding of the schemas.

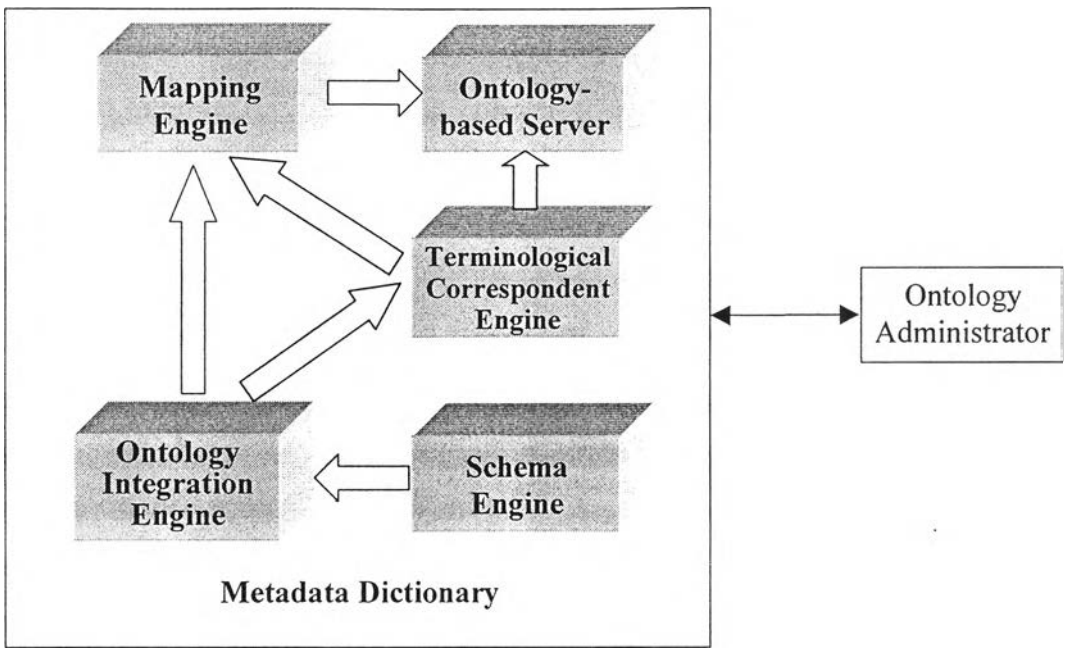


Figure 4.5. The metadata dictionary components.

(3) **The Terminological Correspondent Engine.** This engine generates and manages a set of correspondent terms to eliminate the problems of semantic mismatches. The engine makes use of an attribute classifier (Ho and Tian, 2001; Miller, Hernandez, Haas, Yan, Ho, Fagin, and Popa, 2001), dictionaries, and thesauri to learn possible correspondences. Since there must be human intervention in the process of identifying terminology correspondence between difference ontologies, the engine is imperative for suggestions of possible correspondences and validating human-specified correspondence. The ontology administrator can use the terminological correspondence tool (Miller, Hernandez, Haas, Yan, Ho, Fagin, and Popa, 2001) to view the representation of schemas to create the correspondent terms. This tool also provides a data view mode to display the correspondent terms of the sample data.

(4) **Mapping Engine.** Mapping engine records the mapping information between terms of the virtual schema at the conceptual level of the ontology to terms of the global conceptual schema at the physical level of the ontology. To facilitate the mapping process, the ontology editor (Cui and O'Brien, 2000; Farquhar, Fikes, Pratt and Rice, 1995; Sure, Erdmann, Angele, Staab, Studer and Wenke, 2002) allows the ontology administrator to visually manipulate the ontology constructed in a tree-like structure to

represent the ontology at the conceptual level and physical level. The ontology editor uses the terms defined in the terminological correspondent engine and the global conceptual schema of the ontology integration engine to generate the ontology at the physical level. In edit mode, the ontology administrator can add the global schema of the ontology at the conceptual level and map to global conceptual schema of the ontology at the physical level. In addition, the built-in visual capability supports data browsing of the ontology structure.

(5) **Ontology-based Server.** The ontology-based server stores the XML-based metadata dictionary that holds XML-DTD and XML documents. In order to support flexible update and augmentation capability of the XML-based metadata dictionary, an XML editor is incorporated (Altova, 2002; Jeuring, Meertens, Pemberton, Schrage, Steen and Swierstra, 2002). As such, this metadata dictionary can be scaled up without affecting the overall system configuration.