Resource Space Model, OWL and Database: Mapping and Integration

HAI ZHUGE, YUNPENG XING and PENG SHI
Chinese Academy of Sciences, Beijing, China

Semantics exhibits diversity in the real world, mental abstraction world, document world, and machine world. Studying mappings between different forms of semantics helps unveil the uniformity in the diversity. This article investigates the mappings between three typical semantic models: the Web ontology language (OWL), relational database model, and resource space model (a classification-based semantic model). By establishing mappings between the semantic primitives of the three models, we study the mapping from OWL description onto resource space and analyze the normal forms of the generated resource space. Mapping back from resource space onto OWL description is then discussed. Further, we investigate the mapping between OWL description and relational database, as well as the mapping between relational database and resource space. Normal forms of the generated relational tables are analyzed. To support advanced applications on the future Web, we suggest integrating the resource space, OWL, and databases to form a powerful semantic platform that enables different semantic models to enhance each other.

Categories and Subject Descriptors: H.1.1 [**Models and Principles**]: Systems and Information Theory—*General systems theory, information theory*; H.2.5 [**Database Management**]: Heterogeneous Databases—*Data translation*; I.7.2 [**Document and Text Processing**]: Document Preparation—*Markup languages*

General Terms: Algorithms, Design, Theory

Additional Key Words and Phrases: Integration, mapping, relational database model, resource space model, semantic web, semantic link network, web ontology language.

ACM Reference Format:

Zhuge, H., Xing, Y., and Shi, P. 2008. Resource space model, OWL and database: Mapping and integration. ACM Trans. Intern. Tech., 8, 4, Article 20 (September 2008), 31 pages. DOI = 10.1145/1391949.1391954 http://doi.acm.org/10.1145/1391949.1391954

This work is supported by the National Basic Research Program of China (973 Semantic Grid project No. 2003CB317000) and the International Cooperation Program of the Ministry of Science and Technology of China (Grant no.2006DFA11970).

Authors' addresses: H. Zhuge, Knowledge Grid Research Group, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China; email: zhuge@ict.ac.cn; Y. Xing, P. Shi, Chinese Academy of Sciences, Beijing, 100190, China.

1. INTRODUCTION

The Semantic Web is to support intelligent applications by enriching semantics within the World Wide Web [Berners-Lee et al. 2001]. Ontology is an important layer of the semantic Web stack [Berners-Lee et al. 2006]. The Web ontology language (OWL, www.w3.org/TR/owl-features/) is to facilitate publishing and sharing of ontologies on the Web based on the uniform resource identifier (URI), extensible markup language (XML, www.w3.org/XML/), and resource description framework (RDF, www.w3.org/RDF/). OWL is for use by applications that need to process the content of information, rather than just presenting information to humans. By providing additional vocabulary along with a kind of formal semantics, it facilitates stronger machine interpretability of Web content than that supported by XML, RDF, and RDF schema. It provides three increasingly expressive sublanguages for applications: OWL Lite for expressing classification hierarchy and simple constraint features, OWL DL for maximum expressiveness without losing computational completeness and decidability of reasoning systems, and OWL Full for maximum expressiveness and the syntactic freedom of RDF without computational guarantees.

Database technology is a milestone of information management. Among various data models [Abiteboul et al. 1995; Bachman 1969; Garcia-Molina et al. 2001], the relational database model (RDBM) is the most successful in theory, system, modeling method, and application [Batini et al. 1992; Chen 1976; Codd 1979, 1970]. Original RDBM requires the atomicity of data fields. For online analytical processing (OLAP), multidimensional databases were proposed to logically expose a multidimensional view of data with categorical attributes by read-only operations such as select, drill-down, roll-up, and pivot [Agrawal et al. 1997; Sarawagi 1999]. It helps analysts make decisions on historical transactional data. Database techniques have been used to manage data on the Web [Bohannon et al. 2002; Tatarinov et al. 2002; Vianu 2001] and keep evolving to meet the needs of new application requirements.

The Web is a huge evolving repository of Web pages linked with each other. Its sociality, heterogeneity, dynamicity, and decentralism challenge and push the development of data models.

Classification is a basic method for humans to know the world and manage versatile resources. The resource space model (RSM) is a semantic model for specifying, organizing, and retrieving versatile resources such as image, text, video, audio, Web page, and link by classifying their contents according to different partition methods, organizing them into a multidimensional classification space, and normalizing the resource space for effective management [Zhuge 2004a, 2004b, 2007; Zhuge et al. 2005a]. Every point in the resource space, determined by one coordinate at every axis of the resource space, represents resources of the same category. A point itself can be also a resource space. The normal-form theory and integrity theory of the RSM ensure the correctness of representation and operations on the resource space [Zhuge et al. 2005b]. The RSM is equipped with the SQL-like resource operation language (ROL) to implement such operations as join, disjoin, merge and split [Zhuge 2004a]. A high-dimensional resource space can be split into lower-dimensional resource

spaces by the disjoin operation, and several low-dimensional resource spaces can be joined into a higher-dimensional resource space by the join operation. Resource space, axis, coordinate, and point are sets in essence. Set and partition comprise the mathematical foundation of the RSM. More references on RSM are available at www.knowledgegrid.net.

A semantic model has its own semantic primitive: a set of basic semantic elements of the model. Studying the relationship between different semantic primitives is the basis of studying the mapping between different semantic models. So, we firstly establish the mapping between primitives of the resource space model, OWL, and relational database.

The design of a data model application relies on domain knowledge and application requirements. To relieve such reliance is an important issue in application. The development of domain ontology makes codified domain knowledge. It would be very useful if we could codify existing classification knowledge into an automatic mechanism for mapping domain ontology onto resource spaces or databases. So, we study the approach to automatically map a given OWL description onto a resource space. Mapping from OWL description onto resource space is a process of extracting classification semantics from OWL description and then constructing a resource space according to the definition of RSM and its normal-form theory [Zhuge et al. 2006]. Then, mapping back from resource space onto OWL description is studied.

Further, we investigate the mappings between OWL description and relational database and the mappings between resource space and relational database. OWL, RSM, and RDBM have their own advantages and limitations. The integration of the OWL, resource spaces, and databases allows taking advantage of these models and enabling them to enhance each other. So we propose a solution toword this kind of integration.

Related work concerns approaches to representing the taxonomic relationship of products and service categorization standards in an OWL Lite ontology [Hepp 2005], ontology edit tools and ontology mapping [Kalfoglou et al. 2003; Knublauch et al. 2004], transformation between OWL service and unified modeling language (UML) [Grønmo et al. 2005], from OWL ontology into UML [Gaševic et al. 2004], mapping between OWL DL and Attempto controlled english [Fuchs et al. 2006], transforming existing thesauri and related resources from native format into RDF(S) or OWL [Assem et al. 2004], and using RDBMS to support ontology-based semantic matching [Das et al. 2004]. A metamodel-driven model transformation approach is proposed to interchange rules between the semantic Web rule language and object constraint language [Milanović et al. 2006].

Related work also concerns the software engineering area. Sructured software development is a multistep transformation from a semantic specification on domain business into a semantic specification on software. Semantic specification tools like the entity-relationship (ER) model help developers to specify domain business in a form that can be transformed into a relational database model [Chen 1976; Ng 1981]. Transforming the object-oriented model into RDBM schema can help free application developers from the details of database structure during early development [Blaha et al. 1994]. Transforming

the ER model into the relational database [Batini et al. 1992; Embley 1997; Teorey et al. 1986] and transforming relational into conceptual schemas have also been investigated [Johannesson 1994]. The "transforming" and "converting" in previous work can be regarded as mappings.

2. MAPPING BETWEEN PRIMITIVES

2.1 Basic Semantic Elements of OWL and RDBMS

Ontology facilitates the uniformity and sharing of domain knowledge by four basic modeling primitives: concept, relation, instance, and axiom [Gruber 1993; Neches et al. 1991]. OWL uses the following basic semantic elements to define ontology.

(1) *Class* describes concept by a set of *individuals* (instances) or other existing classes. The *rdfs*:*subClassOf* defines a *subclass* of an existing class.
(2) *Property* describes a binary relation. It has two types: *ObjectProperty* specifying the relation between individuals of the same or different classes, and *DatatypeProperty*, indicating the relations between individuals of classes and RDF literals and XML schema datatypes. Specifically, *rdfs*:*domain* and *rdfs*:*range* restrict the anterior and posterior values of a property, while *rdfs*:*subPropertyOf* defines a subproperty of an existing property.
(3) *Restriction* and *characteristic* describe constraints on relations and axioms. This includes the following semantic elements: *allValuesFrom*, *someValuesFrom*, *Cardinality*, *hasValue*, *TransitiveProperty*, *SymmetricProperty*, *FunctionalProperty*, *InverseOf*, and *InverseFunctionalProperty*.
{*Class*, *property*, *individual*, *restriction*&*characterstic*} constitutes the primitive of OWL.

The basic semantic elements of a relational database are *table*, *tuple*, and *attribute* as well as *constraints* on them to ensure the integrity.

2.2 Basic Semantic Elements of the Resource Space Model

The resource space model is a classification-based semantic model for managing various resources. An n-dimensional resource space RS represents n kinds of classification method X_1,X_2, ..., and X_n on a set of resources denoted as $RS(X_1,X_2, \ldots,X_n)$. The resource set represented by dimension X_i $(1 \leq i \leq n)$ is the union of the resource sets represented by all of its coordinates, denoted as $R(X_i) = R(C_{i1}) \cup R(C_{i2}) \cup \ldots \cup R(C_{im})$, in simple $X_i = \{C_{i1},C_{i2}, \ldots,C_{im}\}$. Every point p in the space represents a set of resources determined by one coordinate at every axis, denoted as $p(C_{1,j1},C_{2,j2}, \ldots,C_{n,jn})$ or $(C_{1,j1},C_{2,j2}, \ldots,C_{n,jn})$. $R(p) = R(C_{1,j1}) \cap R(C_{1,j2}) \cap \ldots \cap R(C_{1,jn})$.

Figure 1 shows a three-dimensional resource space *Spec-Apart-Gen*(*Specialty*, *Apartment*, *Gender*) specifying student information of a college. The three axes are $Specialty = \{math, computer, physics\}$, $Apartment = \{1\#, 2\#, 3\#\}$, and $Gender = \{male, female\}$. Each point denotes a class of students, for example, the point (*math*, 1#, *male*) represents all the male

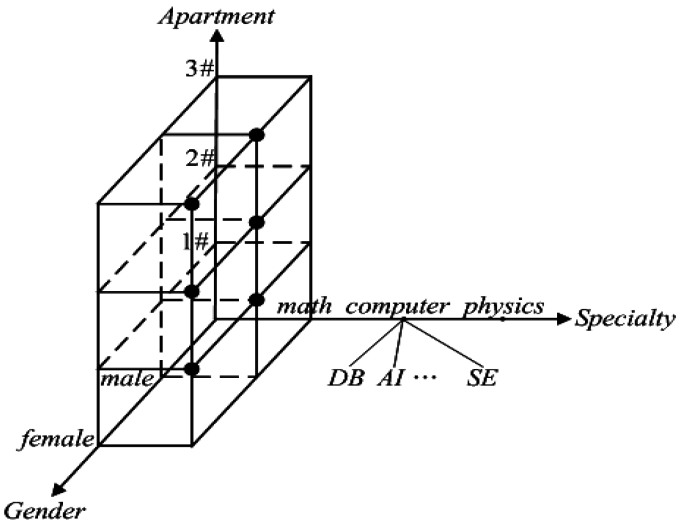


Fig. 1. A three-dimensional resource space.

students belonging to the department of mathematics and living in apartment no.1 of this college.

Coordinates directly residing at axis are called top-level coordinates. Each top-level coordinate can be refined top-down to form a coordinate hierarchy representing classifications of different levels and different granularities. Each node in the hierarchy can be named by the path from the root. For example, the top-level coordinate *computer* at axis *Specialty* shown in Figure 1 has a coordinate hierarchy *computer*(*DB*, *AI*, . . . , *SE*).

The primitive of the resource space model is {*resource space*, *resource*, *axis*, *coordinate*}, where each element is based on two basic mathematical concepts: *set* and *partition*.

2.3 Mapping Rules

Set and relation are two semantic bases independent of specific semantic models. A semantic model can be explained by the semantic bases.

Definition 1. A semantic element is defined by the following items.

(1) A semantic element has a name and a definition domain. A semantic base or semantic element can be the definition domain.
(2) Basic data types like real and integer are semantic elements.
(3) A set consisting of semantic elements is a semantic element.
(4) Set operations on semantic elements constitute semantic elements.

The following two mapping rules limit arbitrarily mapping between semantic elements.

Rule (*Set-Based Mapping*). Mapping α can be established between two semantic elements X and Y (denoted as α: $X \leftrightarrow Y$) if: (1) their definition domains

are the same, (2) their definition domains are based on the same semantic element, or (3) there exists a mapping between their definition domains.

Rule (*Relation-Based Mapping*). Mapping α can be established between two relational semantic elements R: $X \rightarrow Y$ and R': $X' \rightarrow Y'$ (denoted as α: (R: $X \rightarrow Y$) $\leftrightarrow$ (R': $X' \rightarrow Y'$)) if there exists two mappings between semantic elements: $X \leftrightarrow X'$ and $Y \leftrightarrow Y'$.

Definition 1 and the preceding two mapping rules can help establish automatic mapping between semantic models if the primitives of the models are given. According to previous definition, the following mappings between primitives of OWL, RDBM, and RSM can be established.

—Mapping between Primitives of RDBM and OWL $\varphi_{\text{RDBM}\leftrightarrow\text{OWL}}$: {$table_{\text{RDBM}}$, $tuple_{\text{RDBM}}$, $attribute_{\text{RDBM}}$, $constraint_{\text{RDBM}}$} $\leftrightarrow$ {$class_{\text{OWL}}$, $individual_{\text{OWL}}$, $property_{\text{OWL}}$, $restriction\&characteristic_{\text{OWL}}$}. This mapping is based on the following mappings between semantic elements: (1) $table_{\text{RDBM}} \leftrightarrow class_{\text{OWL}}$ (table and class are based on set and relation); (2) $tuple_{\text{RDBM}} \leftrightarrow individual_{\text{OWL}}$(tuple's definition domain is table, individual's definition domain is class, and a mapping can be established between table and class according to item (1)); (3) $attribute_{\text{RDBM}} \leftrightarrow property_{\text{OWL}}$(both attribute and property can be regarded as relation); (4) $attribute_{\text{RDBM}} \leftrightarrow class_{\text{OWL}}$(the projection of table on attributes can be regarded as set); and (5) $constraint_{\text{RDBM}} \leftrightarrow restriction\&characteristic_{\text{OWL}}$(constraint is regarded as relation).

—Mapping between Primitives of RDBM and RSM $\varphi_{\text{RDBM}\leftrightarrow\text{RSM}}$: {$table_{\text{RDBM}}$, $tuple_{\text{RDBM}}$, $attribute_{\text{RDBM}}$, $constraint_{\text{RDBM}}$} $\leftrightarrow$ {$ResourceSpace_{\text{RSM}}$, $resource_{\text{RSM}}$, $axis_{\text{RSM}}$, $coordinate_{\text{RSM}}$, $constraint_{\text{RSM}}$}. This mapping is based on the following mappings between semantic elements: (1) $table_{\text{RDBM}} \leftrightarrow ResourceSpace_{\text{RSM}}$ (resource space is based on set and relation); (2) $table_{\text{RDBM}} \leftrightarrow axis_{\text{RSM}}$ (axis is regarded as set); (3) $tuple_{\text{RDBM}} \leftrightarrow resource_{\text{RSM}}$(tuple's definition domain is table, resource's definition domain is resource space, and mapping between table and resource space can be established according to item (1)); (4) $table_{\text{RDBM}} \leftrightarrow coordinate_{\text{RSM}}$; (5) $attribute_{\text{RDBM}} \leftrightarrow coordinate_{\text{RSM}}$ (coordinate is regarded as set); (6) $attribute_{\text{RDBM}} \leftrightarrow axis_{\text{RSM}}$; and (7) $constraint_{\text{RDBM}} \leftrightarrow constraint_{\text{RSM}}$.

—Mapping between Primitives of OWL and RSM $\varphi_{\text{OWL}\leftrightarrow\text{RSM}}$: {$class_{\text{OWL}}$, $individual_{\text{OWL}}$, $property_{\text{OWL}}$, $restriction\&characteristic_{\text{OWL}}$} $\leftrightarrow$ {$ResourceSpace_{\text{RSM}}$, $resource_{\text{RSM}}$, $axis_{\text{RSM}}$, $coordinate_{\text{RSM}}$, $constraint_{\text{RSM}}$}. This mapping is based on the following mappings between semantic elements: $class_{\text{OWL}} \leftrightarrow ResourceSpace_{\text{RSM}}$, $class_{\text{OWL}} \leftrightarrow axis_{\text{RSM}}$, $class_{\text{OWL}} \leftrightarrow coordinate_{\text{RSM}}$, $individual_{\text{OWL}} \leftrightarrow resource_{\text{RSM}}$, $property_{\text{OWL}} \leftrightarrow axis_{\text{RSM}}$, $property_{\text{OWL}} \leftrightarrow coordinate_{\text{RSM}}$, and $restriction\&characteristic_{\text{OWL}} \leftrightarrow constraint_{\text{RSM}}$.

3. MAPPING FROM OWL DESCRIPTION ONTO RESOURCE SPACE

The inputs of the mapping from OWL description onto resource spaces, as shown in Figure 2 are the OWL file and the ancestor classes; the top-level

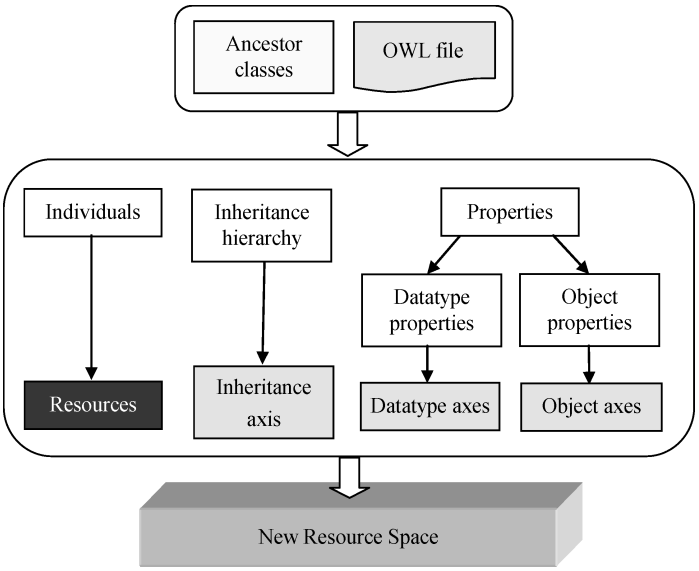


Fig. 2. The main process of mapping from OWL file onto resource space.

classification on resources. The class inheritance hierarchies and properties in OWL are mapped onto axes of new resource spaces. A parameter, that is, the number of dimensions, can be added to restrict the dimensions of the new resource space. Synonyms described by *equivalentClass*, *equivalentProperty*, and *sameAs* in OWL are replaced by complex names during preprocessing.

3.1 Mapping Inheritance Hierarchy onto Inheritance Axis

In OWL, the ancestor classes and their subclasses inherently represent classification hierarchies. This naturally corresponds to the set and partition of RSM, so the inheritance hierarchy of classes in OWL can be mapped onto an inheritance axis of resource space. The process of forming the inheritance axis consists of the following three steps.

(1) Parse the OWL file to find the subclasses and instances of an input ancestor class.
(2) According to the *rdfs*:*subClassOf* relations, construct the inheritance hierarchy structure, with the ancestor class as the root.
(3) Transform the structure into a tree, take its first-level children as the top-level coordinates of the inheritance axis, and name the axis after the root.

Figure 3 shows an example of mapping inheritance tree onto inheritance axis. The hierarchy of Figure 3(b), consisting of the ancestor class File, its subclasses, and individuals, is extracted from the OWL description of Figure 3(a). The inheritance tree of Figure 3(b) is mapped onto the inheritance axis shown in Figure 3(c).

In a multi-inheritance hierarchy, a class can be the subclass of more than one class, so the parent classes cannot classify resources independently. Process 1

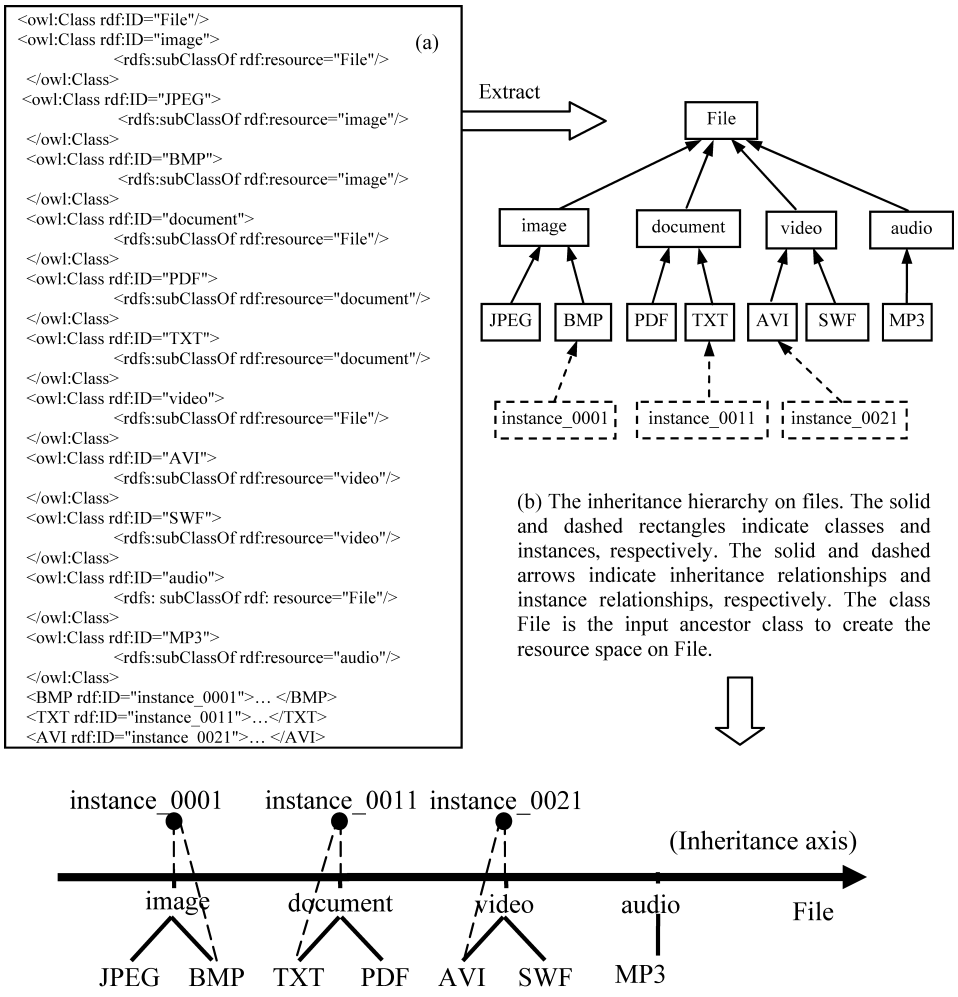


Fig. 3. Example of mapping inheritance tree onto inheritance axis.

converts an inheritance graph into a tree and guarantees that the output tree contains individuals and their classes [Zhuge et al. 2006].

OWL's concrete classes may own subclasses and individuals, but abstract classes can only have subclasses. The individuals of the concrete class may not be located by the subclass coordinates. For example, the concrete class Manager in Figure 4(a) has a subclass Director and three individuals Jane, Joe, and Mary. Director has its own individual Tim. If this structure is mapped onto the inheritance axis, the coordinates Manager and Director will be at different levels. The coordinate Director can specify Tim, but it cannot specify Jane, Joe, or Mary.

The following two strategies deal with the concrete classes: (1) Discard its subclasses and link their individuals directly to the parent class; and (2) add a

new subclass for the concrete class, and individuals of the concrete class can be identified as individuals of the added subclass. The former strategy weakens the classification semantics, but simplifies the process. The latter enriches the classification semantics.

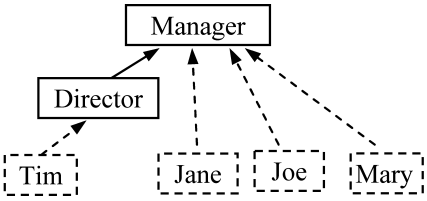
Process 2 in Figure 4 checks and deals with concrete classes [Zhuge et al. 2006]. If *bDiscard*=true, subclasses are discarded, otherwise a new subclass is added and named after the parameter *newClassName*. Figure 4(b) is the result of *bDiscard*=true. The subclass Director is deleted and its individual Tim becomes the individual of Manager. Figure 4(c) shows the result when *bDiscard*=false. A new subclass named General Director is added, with Jane, Joe and Mary as individuals. So far, the inheritance axis is created according to the ancestor class and inheritance hierarchy.

3.2 Mapping Properties onto Property Axes

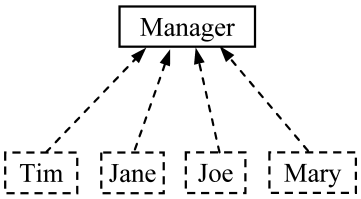
Properties in OWL are used to describe characteristics of classes and individuals. Properties whose domains include the ancestor class can be mapped onto a

Process. 1. void GraphToTree(Graph G, Tree T)

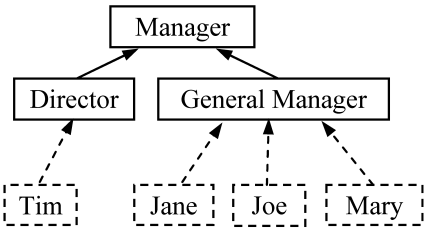
```
{/*convert a connected directed graph G into a tree(s) T*/
  For every node
  {/*treat from the bottom level*/
    If (indegree(node, G) = 0) {/*a bottom node*/
      Output node into T as a leaf;
      If (outdegree(node, G)= 0) {
        Show message "error: an individual hasn't class";
        Return;
      }
      Else if (outdegree(node, G)=1) {/*uni-inheritance*/
        setMark(node, T, treated);
        Output getParent(node,G) into T;
      }
    }
  }
  While (getUntreatedNumber(T)>0) {
    Get an untreated node from T;
    If (outdegree(node, G)=1) {/*qualified node*/
      If (getMark(getParent(node,G)) != deleted ) {
        Output getParent(node, G) as parent of node into T;
      }
    }
    Else if (outdegree(node, G)>1) {/*multi-inheritance*/
      For every ancestor of node in G {
        If (getMark(ancestor, G) != deleted) {
          setMark(ancestor, G, deleted);
          Delete ancestor from T;
        }
      }
    }
    setMark( node, T, treated ); /*mark treated node*/
  }
}
```



(a) concrete class Manager with both subclass and instances



(b) result of discarding subclasses



(c) result of adding subclass

Process. 2 Boolean Check&ChangeConcreteClass (Class conClass, Boolean bDiscard, String newClassName) {

```
If (conClass has both individuals and subclasses) {
  If (bDiscard ) { /*discard subclasses*/
    For every subclass of conClass {
      Move its individuals into conClass;
      Delete subclass;
    }
  }
  Else { /*add a new subclass*/
    Create a new class named newClassName;
    Get all individuals of conClass;
    Move the individuals into newClassName;
    Add newClassName as a subclass of conClass;
  }
  Return true;
}
Else {/*need not be modified*/
  Return false;
}
}
```

Fig. 4. An example of processing a concrete class.

property axis. The *owl*:*DatatypeProperty* declares property with data type coming from RDF literals and XML schema data types. A data-type property can be mapped onto a data-type axis in the new resource space, and named after the property name. Its coordinates include all values within the range of the property. In the example of Figure 5, the *unsignedInt* type property *CaveNo* specifies the cave where artifacts reside, and *xsp.owl* is imported to restrict

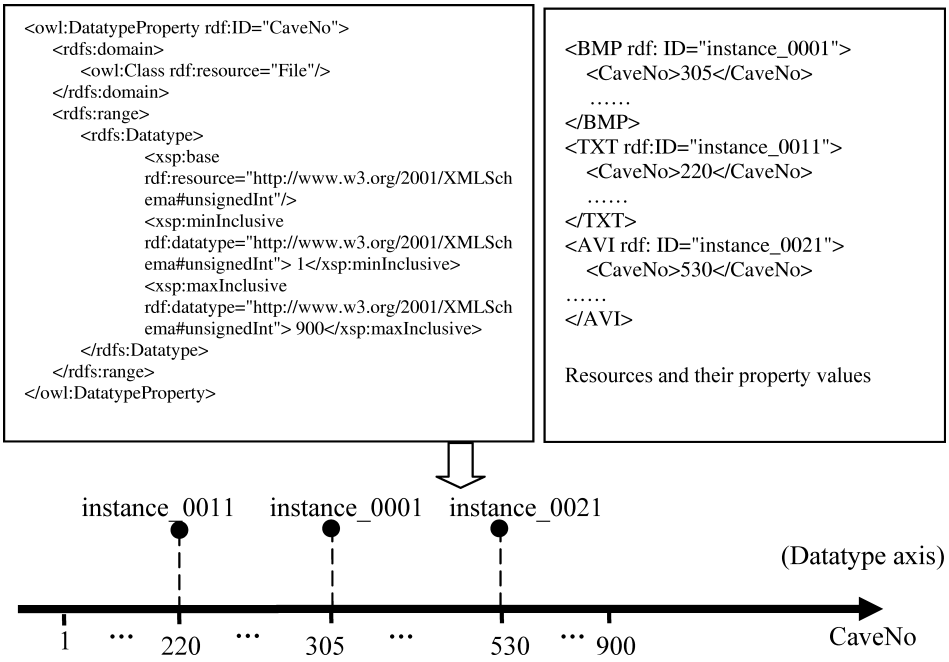


Fig. 5. The data-type axis transformed from data-type property. Property values are taken as the coordinates of the axis.

data-types. The range of *CaveNo* is from 1 to 900, and its domain is the class File. The property can be mapped onto an axis whose coordinates are elements within the property range.

Object property in OWL specifies the relation between objects by declaring *owl*:*ObjectProperty*. An object property can be mapped onto a homonymous axis, called the object axis. Its coordinates consist of the classes of those individuals within the property's range and they are usually in an inheritance hierarchy. All individuals within the property's range, together with their classes, form an inheritance hierarchy.

The procedure for creating an object axis is similar to that of creating inheritance axis. Processes 1 and 2 are used to get a directed tree or trees. The output tree structure is mapped onto coordinates on the object axis. The upper portion of Figure 6 shows an object property Content in OWL. The range of this property is declared as *ContentClass*, with subclasses painting, statue, and architecture. Its inheritance hierarchy and the corresponding object axis Content are shown in Figure 6(a) and (b).

3.3 Mapping Individuals onto Resources

An OWL description can include several ancestor classes. Individuals may belong to different ancestor classes, so only those individuals belonging to the ancestor class are mapped into resource space as resources [Zhuge et al. 2006].

The resources and axes mapped from an OWL description constitute a coordinate system. Every resource has a location determined by its classes and

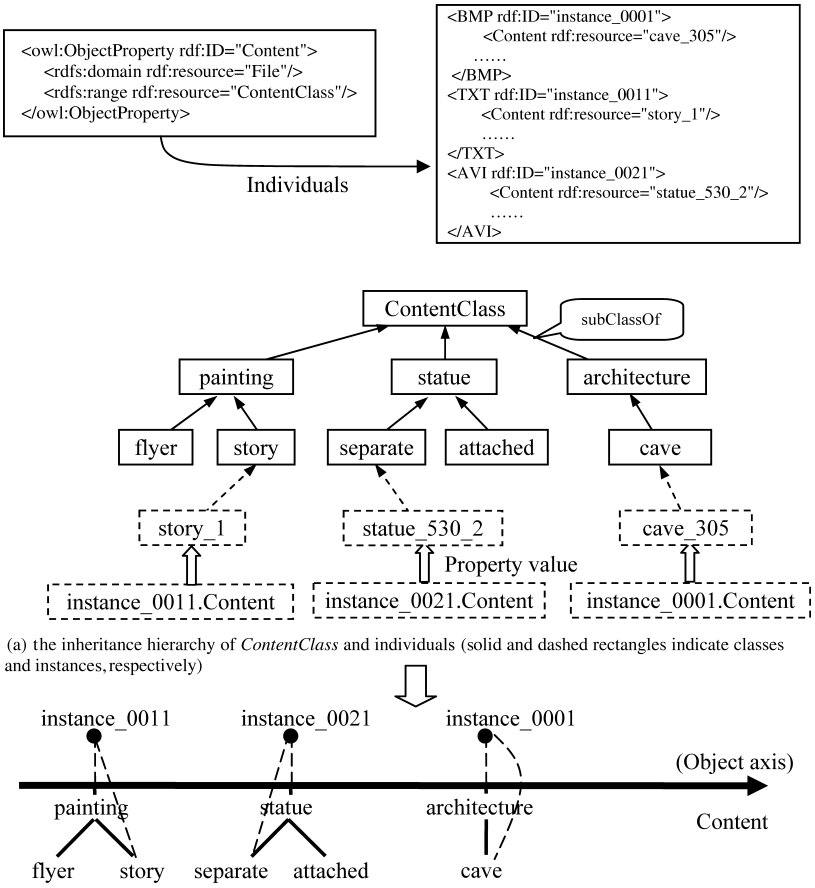


Fig. 6. Object axis mapped from object property.

properties' values. Resources are indexed in corresponding points in the resource space. This coordinate system constitutes a resource space [Zhuge 2004a], where a point uniquely represents a set of resources. Figure 7 shows the top-level structure of the generated resource space.

4. NORMAL-FORM ANALYSIS

Focusing on the mapping approach, we assume that the OWL description is well defined, that is, the resource space generated from a well-defined OWL description can represent correct classification semantics.

Definition 2. The first normal form (1NF) of resource space requires that there should be no name duplication between coordinates at every axis [Zhuge 2004a; Zhuge et al. 2005a].

The 1NF can be checked by comparing all of the coordinate names at one axis. It can be satisfied by combining the duplicate coordinates into one coordinate

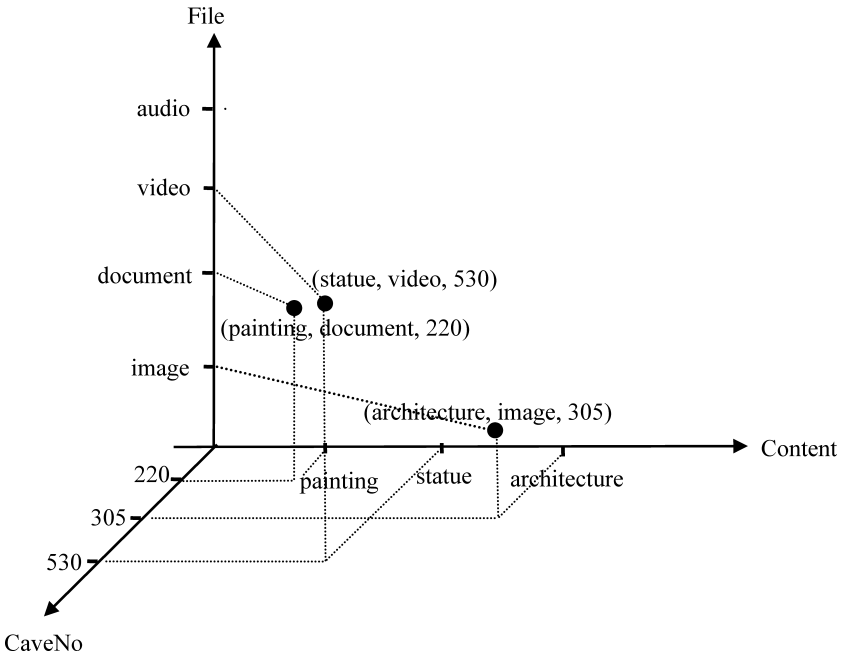


Fig. 7. The generated resource space for describing Dunhuang cave content. A point in the space is specified by the coordinates on every axis; for example, point (architecture, image, 305) represents a set of images describing the architecture of cave no. 305.

and grouping the corresponding resources into one set as the resources of the new coordinate. Automatically checking the normal form needs domain ontology.

A well-defined OWL description does not contain duplicated classes, individuals, and properties, so the coordinates mapped from classes and individuals at any axis should not be duplicated. Hence, the resource space generated from OWL description satisfies 1NF.

Definition 3. The second normal form (2NF) of RSM satisfies 1NF, and also that for any axis, any two coordinates are independent from each other [Zhuge 2004a; Zhuge et al. 2005a].

The 2NF avoids implicit coordinate duplication, and prevents one coordinate from semantically depending on another. Semantic independence here means that a coordinate is not the synonym, abstract concept, concrete concept, instance, or quasisynonym of another coordinate.

Since the synonymic classes, properties, and individuals are already combined by preprocessing, there are no synonymic coordinates at the inheritance axis and property axes. In a well-defined OWL file, the abstract concept of a coordinate should be declared as its parent class. Since the hierarchical structure of coordinates is based on the *subClassOf* relations, the abstract concept and the coordinates are at different levels. The concrete concept and instance of a coordinate should be its subclass and instance, respectively. They are also at different levels in the coordinate hierarchy. To avoid semantic confusion,

coordinates at different levels should not be used at the same time. Multiple inheritance is eliminated during the creation of an inheritance axis, so every resource has a certain value on the axis. The quasisynonymic classes do not influence classification. Since the coordinates at the data-type axis are values of one type or their classification, the quasisynonymic values cannot influence the classification. Similar to the inheritance axis, coordinates at the object axis are classes of their property values. They can avoid classification confusion. So, there are no influential quasisynonyms at any axis.

The 2NF avoids the intersection of classification at the same axis. In the resource space created from a well-defined OWL description, resources are classified clearly by the coordinates at any axis. So, the coordinates at every axis are semantically independent. Generally, classification confusion at an axis implies that the OWL description contains some confusing description. Hence, a well-defined OWL description can be mapped onto a 2NF resource space.

Definition 4. (Orthogonality of Axes [Zhuge 2004a; Zhuge et al. 2005a].)

(1) Let $X = (C_1, C_2, \ldots, C_n)$ be an axis and C_i' be a coordinate at another axis X'. We say that X finely classifies C_i' (denoted as C_i'/X) if and only if: (1) $(R(C_k) \cap R(C_i')) \cap (R(C_p) \cap R(C_i')) = \text{NULL}$ ($k \neq p$, and k, $p \in [1, n]$), and (2) $(R(C_1) \cap R(C_i')) \cup (R(C_2) \cap R(C_i')) \cup \ldots \cup (R(C_n) \cap R(C_i')) = R(C_i')$ hold. As the result of the fine classification, $R(C_i')$ is partitioned into n categories: $R(C_i'/X) = \{R(C_1) \cap R(C_i'), R(C_2) \cap R(C_i'), \ldots, R(C_n) \cap R(C_i')\}$.

(2) For two axes $X = \{C_1, C_2, \ldots, C_n\}$ and $X' = \{C_1', C_2', \ldots, C_m'\}$, we say that X finely classifies X' (denoted as X'/X) if and only if X finely classifies C_1', C_2' , ... , and C_m'.

(3) Two axes X and X' are called orthogonal with each other (denoted as $X \perp X'$) if X finely classifies X' and vice versa, that is, both X'/X and X/X' hold.

Characteristic 1. *Fine classification is transitive, that is, if X''/X' and X'/X, then X''/X holds.*

Definition 5. The third normal form (3NF) is a 2NF resource space and any two axes are orthogonal with each other [Zhuge 2004a].

The resource space generated from an OWL description contains one inheritance axis and several property axes. This implies the following lemmas.

Lemma 1. *In the resource space generated from OWL description, any two axes are orthogonal if and only if:* (1) *the inheritance axis is orthogonal with any property axes, and* (2) *any two property axes are orthogonal with each other.*

According to Definition 4 and Characteristic 1, we have Lemma 2.

Lemma 2. *The orthogonality between two axes is transitive, that is, if $X' \perp X$ and $X \perp X''$, then $X' \perp X''$* [*Zhuge et al.* 2005a].

Lemma 3. *In the resource space generated from OWL description, an arbitrary two property axes are orthogonal with each other if the inheritance axis is orthogonal to any property axes.*

PROOF. Let the inheritance axis be X^I, and X_1^P and X_2^P be arbitrary two property axes.

If the inheritance axis is orthogonal to any property axis, $X^I \perp X_1^P$ and $X^I \perp X_2^P$ hold. Because $X^I \perp X_1^P \Leftrightarrow X_1^P \perp X^I$ and, according to Lemma 2, $X_1^P \perp X_2^P$ holds, that is, an arbitrary two property axes are orthogonal with each other. □

THEOREM 1. *If the generated resource space is 2NF and the inheritance axis is orthogonal to any property axes, the resource space satisfies 3NF.*

PROOF. According to Lemma 3, if the inheritance axis is orthogonal to any property axes, we have that any two property axes are orthogonal to each other. From Lemma 1, we have that any two axes are orthogonal with each other in the generated resource space. According to the definition of 3NF, the resource space satisfies 3NF. □

LEMMA 4. *For any two axes X_i and X_j of a resource space, $X_j \perp X_i \Leftrightarrow R(X_j) = R(X_i)$ holds* [*Zhuge et al.* 2005*a*].

THEOREM 2. *Let the inheritance axis of the 2NF resource space generated from OWL description be X^I. If $R(X^I) = R(X^P)$ holds for any property axis X^P in the resource space, the resource space satisfies 3NF.*

PROOF. From Lemma 4, $R(X^I) = R(X^P) \Rightarrow X^I \perp X^P$. Then the inheritance axis is orthogonal to any property axis. According to Theorem 1, the resource space satisfies 3NF. □

LEMMA 5. *If a resource r owns property P, then r can be represented by the property axis X^P mapped from P, that is, $r \in R(X^P)$.*

PROOF. According to the process of mapping property onto a property axis, the coordinates at X^P originate from three kinds of elements: any value within the range, a classification on the range, and the classes of all individuals within the range. Because r owns property P, so P's value is within the range, and r has a coordinate on X^P. So $r \in R(X^P)$ holds. □

THEOREM 3. *If every property axis of the 2NF resource space RS is mapped from the common properties of all the subclasses of the ancestor class in the OWL description, the resource space RS satisfies 3NF.*

PROOF. Let E_R be all resources to be organized by RS, X^I be the inheritance axis, and X^P be an arbitrary property axis. We have $R(X^I) \subseteq E_R$, $R(X^P) \subseteq E_R$ and any resource $r \in E_R$.

(1) Since any resource r is an individual of a class, r has its class on X^I. Then $r \in R(X^I)$ and $E_R \subseteq R(X^I)$ hold. From $R(X^I) \subseteq E_R$, we get $R(X^I) = E_R$.

(2) Since r is an individual of a class, it has the same properties of its class. P is a common property owned by every class. Then, we have that r must own P as its property. From Lemma 5, $r \in R(X^P)$ holds. Since $r \in E_R$ holds, we have $E_R \subseteq R(X^P)$. From $R(X^P) \subseteq E_R$, we have that $R(X^P) = E_R$ holds.

From items (1) and (2), we get $R(X^I) = R(X^P)$. According to Theorem 2, RS satisfies 3NF. □

Theorem 3 shows that if every property axis in the generated resource space is created by the common properties, then the resource space satisfies 3NF. The mapping with this condition can generate a 3NF resource space.

5. STRATEGIES AND RULES OF MAPPING OWL DESCRIPTION ONTO RSM

5.1 Priority on Integrating OWL Descriptions and Integrating Resource Spaces

Since a large-scale ontology needs cooperative development, an integration of OWL descriptions developed by team members is very important in ontology engineering. The following characteristic unveils the relationship between the integration of OWL descriptions and the integration of the resource spaces generated from these OWL descriptions.

CHARACTERISTIC 2. *Let OWL-description=OWL-description*$_1$ $\cup$ *OWL-description*$_2$ *be the integration of OWL-description*$_1$ *and OWL-description*$_2$ *based on the union of graphs. RS, RS*$_1$*, and RS*$_2$ *are resource spaces created from OWL-description, OWL-description*$_1$ *and OWL-description*$_2$*, and they represent the same type of resources. Then, RS*$_1$ *is a subspace of RS (denoted as* $RS \supseteq RS_1$*), RS* $\supseteq RS_2$*, and* $RS \supseteq RS_1 \cdot RS_2$ *(i.e., the join of two spaces* $RS_1 \cdot RS_2$ *is the subspace of RS).*

PROOF. Since the integration operation $\cup$ is based on the union of graphs, the result of integration does not reduce individuals, properties, and classes. Therefore RS_1 and RS_2 are the subspaces of RS (i.e., all resources, axes, and coordinates in RS_1 or in RS_2 are also in RS). If there are common axes between RS_1 and RS_2, then RS_1 and RS_2 can be integrated by the RSM's join operation: $RS_1 \cdot RS_2$ [Zhuge 2004a; Zhuge et al. 2005a]. Since a join operation does not increase any new axis, coordinate, and resource, $RS_1 \cdot RS_2$ is also a subspace of RS. □

This characteristic implies the following strategy of mapping OWL description onto resource space.

STRATEGY 1. *Integrating OWL descriptions takes higher priority than integrating resource spaces*.

This strategy suggests that selecting the integrated OWL description for mapping can reserve more semantics than selecting separate OWL descriptions for mapping and then integrating the created resource spaces.

5.2 Diversity and Invariance of Mapping

Humans classify real-world resources according to epistemology, make consensus on classifications by sharing, and develop the classifications toward standards and taxonomy. Classification consensus, standards, and taxonomies keep updating with the deepening of human cognition. Different epistemologies lead to different classification hierarchies. Users of the resource space should know relevant classification standards and common sense. Figure 8 depicts the diversity of classification due to epistemological difference. The classification

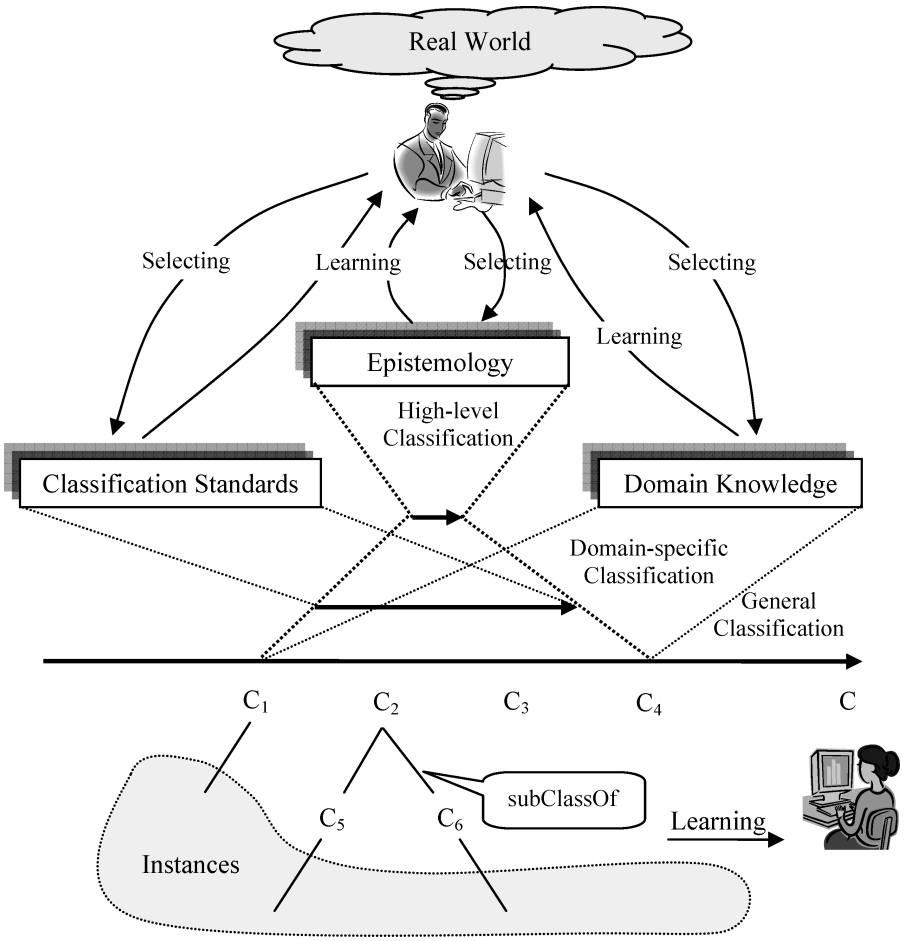


Fig. 8. Diversity of classification hierarchy.

hierarchy can help users learn details when they just know a part of the classification hierarchy.

Figure 9 shows the diversity of mapping. Given the ancestor class C defined with properties P_1, P_2, ..., and P_n in the OWL description, the following two methods can be used to create resource spaces.

(1) *Mapping* 1. Construct resource space RS $(X_1, X_2, \ldots, X_n)$ by mapping each property of class C onto the corresponding property axis (e.g., P_i corresponds to X_i), as shown in Figure 9.

(2) *Mapping* 2. Create resource space $RS'(H, X_1, X_2, \ldots, X_n)$ by not only mapping each property of class C onto the property axis in RS' as in item (1), but also mapping the inheritance hierarchy of class C onto inheritance axis H, as shown in Figure 9.

Moreover, the constructed resource space can be an embedded space; for example, a point can be another resource space if this point's corresponding

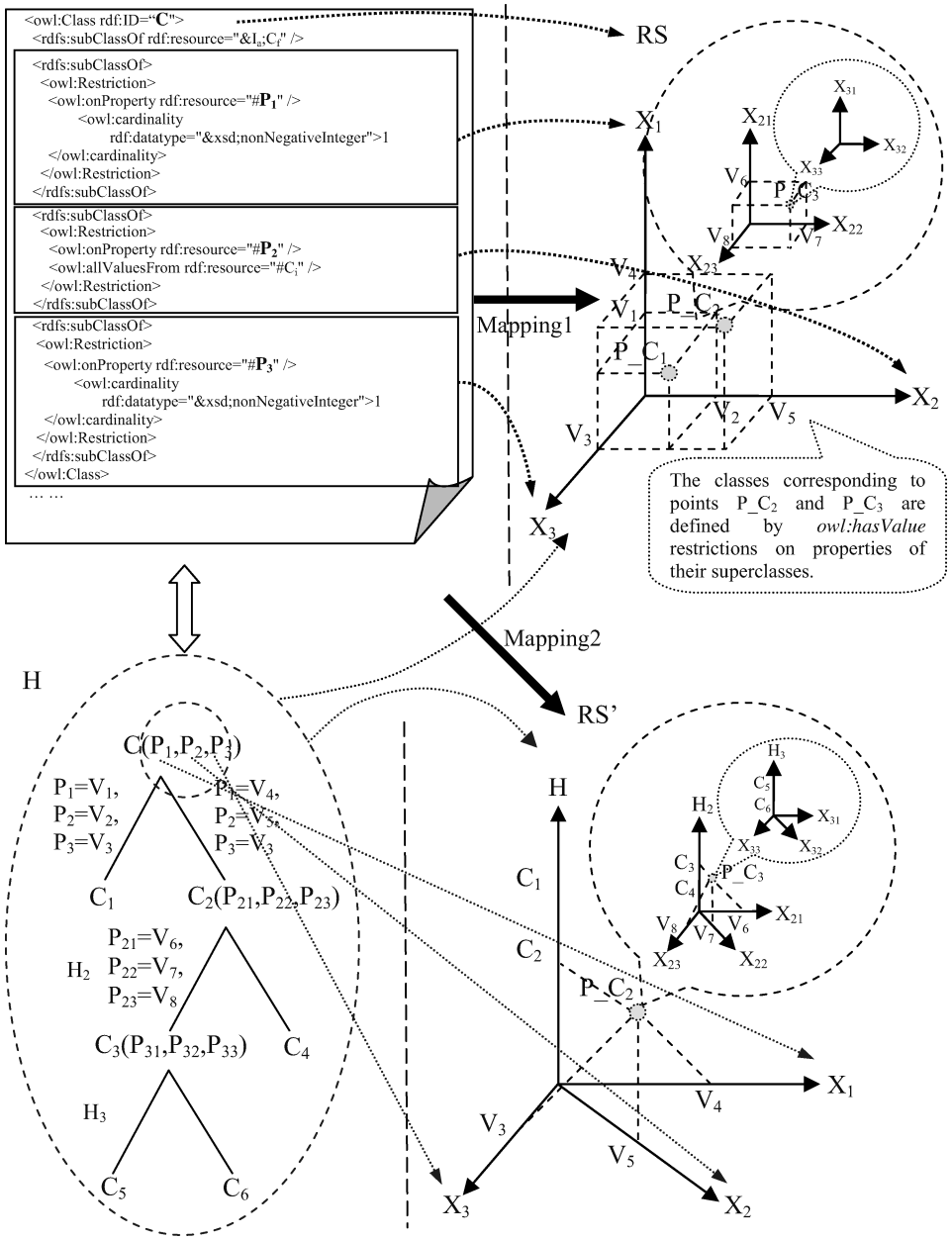


Fig. 9. Diversity of mapping from OWL description onto resource space.

class in OWL description is defined by *owl:hasValue* restrictions on properties of the superclass.

For an OWL description, given different ancestor classes, the mapping process will generate different resource spaces. Then, *what is the relationship between the generated resource spaces if we input related ancestor classes*? Given an ancestor class, the mapping can generate resource spaces of different dimensions. Then, *what is the resource space with the minimum number of dimensions*? *What is the resource space with the maximum number of dimensions*?

An OWL description can be mapped onto resource spaces of different dimensions. When describing, people might have determined what kind of *thing* needs to be expressed. The proposed mapping approach takes such a thing as the ancestor class and its class inheritance hierarchy under particular epistemology as the backbone. The inheritance axis represents all individuals in the application domain, while all common properties can also represent all the individuals. The following two lemmas reflect the invariance.

LEMMA 6. *Among all the possible 2NF resource spaces generated by the proposed mapping, the one with a single inheritance axis is the resource space with minimum dimensions (minimum resource space).*

The minimum resource space is a one-dimensional classification space, which does not make use of the orthogonal characteristic of a 3NF resource space.

LEMMA 7. *Among all possible 3NF resource spaces generated by the proposed mapping, the one with the inheritance axis and the axes of common properties of all classes is the resource space with the maximum number of dimensions (maximum resource space).*

In the same application domain, different forms of resource space can have the same expressive ability. For example, the three-dimensional resource spaces RS_1(*publisher*, *author*, *year*) and RS_2(*ISBN*, *discipline*, *price*) have the same expressive ability, two-dimensional resource spaces RS_3(*discipline*, *author*) and RS_4(*ISBN*, *price*) have the same expressive ability, and one-dimensional resource spaces RS_5(*author*) and RS_6(*ISBN*) have the same expressive ability. The expressiveness of the minimum resource space and that of the maximum resource space cover other forms of resource spaces generated by the proposed mapping approach. Based on the preceding discussion, we suggest the following strategy of mapping.

Strategy 2. Obtain a 3NF resource space by mapping an inheritance hierarchy onto the inheritance axis and mapping the common properties of classes onto property axes, then establishing views of different dimensions according to different application requirements.

CHARACTERISTIC 3. *For an OWL description, given two ancestor classes C and C′, the mapping process generates two resource spaces RS and RS′, respectively. If C is a subclass of C′, then RS, is a subspace of RS′.*

PROOF. We first check the case that RS and RS' comprise the minimum one-dimensional resource space with only one inheritance axis. Let X and X' be the inheritance axes mapped from C and C'. Since C is a subclass of C', according

to the mapping process, we have that any coordinate at X is a coordinate at X', so X' includes X. Therefore RS is a subspace of RS'. Then, we check the case that RS and RS' constitute the maximum resource space. Since C is a subclass of C', the axes mapped from the common properties of C and C' are the same. The only difference is the inheritance axis. Based on the first case, we have that RS is the subspace of RS'. Since the expressiveness of the minimum and maximum resource spaces cover other forms of resource spaces, this characteristic holds. □

6. MAPPING RESOURCE SPACE ONTO OWL DESCRIPTION

6.1 Mapping Process

Mappings between the primitives of RSM and OWL enable the following process to map a resource space onto an OWL description.

(1) Input the definition of a resource space $RS(X_1, \ldots, X_n)$;
(2) create class RS as well as classes $rangeX_1$, $rangeX_2$, ..., and $rangeX_n$, corresponding to X_1, X_2, ..., and X_n;
(3) create property X_i $(1 \leq i \leq n)$ with domain RS using *rdfs*:*domain* and range $rangeX_i$ using *rdfs*:*range*;
(4) for each coordinate C_k of $X_i (1 \leq i \leq n)$, make C_k a subclass of class $rangeX_i$ using *rdfs*:*subClassOf* (if C_k is a tree, create a subclass hierarchy);
(5) for each resource $r(C_1, \ldots, C_n)$ in resource space RS, create the individuals of class RS and class $rangeX_i$ $(1 \leq i \leq n)$; and
(6) create the individuals of property X_i, denoted by a pair (individual of RS, individual of $rangeX_i$).

A 2NF resource space can be mapped onto a well-defined OWL description without any name duplication and overlap of classes. A 3NF resource space requests each class X_i to have the same expression ability, so the implicit constrains of 3NF should be explicated in the generated OWL description.

6.2 Discussion on the Mappings between OWL Description and Resource Space

With the mappings between OWL description and resource space, the following two mapping connections can be made.

(1) Resource Space $RS \rightarrow$ OWL description $\rightarrow$ Resource Space RS'; and
(2) OWL description $D \rightarrow$ Resource Space $\rightarrow$ OWL description D'.

We hope to answer the following interesting questions: Is there any difference between RS and RS'? Is there any difference between D and D'? For an application, OWL can describe richer semantics than RSM, as the latter mainly describes classification semantics. So, the mapping from OWL description onto resource space could lose some of the semantic information described in OWL, but the mapping from resource space onto OWL description can reserve the semantics described by RSM, as the classification semantics can be fully defined in OWL. Hence, for mapping (1), according to the introduced mapping processes,

the only difference between RS and RS' is that RS' contains an empty axis, named after the resource space representing the whole space. Without changing the semantics of RS', removing this empty axis makes RS' the same as RS. For mapping (2), D can contain richer semantics than D'.

7. MAPPINGS BETWEEN OWL DESCRIPTION AND RELATIONAL DATABASE

7.1 Mapping Relational Database onto OWL Description

According to the mapping between primitives, a relational table can be regarded as a class. Suppose the primary key of 3NF relational table $T(A_1, A_2, \ldots, A_n)$ is $\{A_g, \ldots, A_m\}$ $(1 \leq g \leq m \leq n)$, the general idea of this mapping is to map T onto a class, map the primary key onto a class, map the foreign keys onto object properties, and map the primary attributes and nonforeign-key attributes onto data-type properties. The detailed mapping process is as follows.

(1) For each of the primary attributes and nonforeign-key attributes A_i $(1 \leq i \leq n)$, construct a functional data-type property P_i.
(2) The primary key $\{A_g, \ldots, A_m\}$ is mapped onto a class named k_C with properties $P_g, \ldots, P_m$. For each property $P_j (g \leq j \leq m)$, the value of *owl:cardinality* is set as 1 and the value of *owl:allValuesFrom* is set as the built-in OWL data type corresponding to the domain of attribute A_j in table T.
(3) Define a functional and inverse-functional object property *key*. Map relational table T onto class C with a set of restricted properties created in (1) as well as the property *key*. For the property *key*, the value of *owl:cardinality* is set as 1 and the value of *owl:allValuesFrom* is set as class k_C. For each property P_i, the value of *owl:allValuesFrom* is set as the built-in OWL datat-ype corresponding to the domain of attribute A_i. The value of *owl:cardinality* of each primary property P_j is set as 1.
(4) For each tuple $(V_1, V_2, \ldots, V_n)$ of T, construct an individual of class C named as $C\_V_g\_\ldots\_V_m$, and an individual of class k_C named as $k\_C\_V_g\_\ldots\_V_m$, then create $(C\_V_g\_\ldots\_V_m, V_i)$ as an individual of property P_i $(1 \leq i \leq n)$ and $(C\_V_g\_\ldots\_V_m, k\_C\_V_g\_\ldots\_V_m)$ as an individual of property *key*, respectively.
(5) If $T(A_1, A_2, \ldots, A_n)$ has foreign keys $\{A_{r1}, \ldots, A_{t1}\}$, $\{A_{r2}, \ldots, A_{t2}\}$, $\ldots$, and $\{A_{rs}, \ldots, A_{ts}\}$ $(1 \leq r_u \leq t_u \leq n, \text{x} \leq u \leq s)$, for each foreign key $\{A_{rv}, \ldots, A_{tv}\}$ $(1 \leq r_v \leq t_v \leq n, \text{x} \leq v \leq s)$ referring to table T_v, construct a functional object property p_C_v for class C, and the value of *owl:allValuesFrom* of p_C_v is set as class C_v.

The approach in Trinh et al. [2006] presents a method of constructing OWL ontologies from relational databases using the vocabularies and structural constraints defined in a shared relational database ontology. It mainly enhances the interoperability between relational database systems. Our approach focuses on the mapping from relational database onto OWL description by establishing the correspondence between the structures as well as the relevant constraints of the two models. This brings convenience for interoperation between existing OWL ontologies and the new OWL ontology generated from database.

7.2 Mapping OWL Description onto Relational Database

In OWL, a class can be regarded as a relational table. Properties of a class can be regarded as the attributes of a relational table. The inheritance (*subClassOf*) relation between classes can be realized by the foreign key between relational tables.

For each class C with properties $P_1, \ldots,$ and P_n in OWL description, the mapping process is as follows.

(1) Create a relational table T with *varchar* type attribute *ID* as its primary key, and map class C onto T.
(2) For each data-type property P_i $(1 \leq i \leq n)$ of class C, if it is functional or its cardinality is equal to 1, map property P_i onto attribute A_i with the data type corresponding to the range of property P_i, and make A_i the attribute of table T.
(3) For each object property $P_j (1 \leq j \leq n)$ of class C having class C' as its range, if it is functional or its cardinality is equal to 1, create *varchar* type attribute A_j for T as a foreign key referring to the attribute *ID* of another table T' corresponding to C'.
(4) If C is a subclass of C_1, define the attribute *ID* of T as the foreign key referring to the attribute *ID* of table T_1 corresponding to C_1.

The cardinality of relevant properties in OWL DL description is less than or equal to 1, and the *ID* in each table can uniquely determine other attributes, so the generated table satisfies 1NF and 2NF of RDBM. But we cannot guarantee 3NF if the functional dependence relationship between properties is not defined in the OWL description. If dependence relationships between properties have been defined in the given OWL description, then the functional dependence relationship between attributes of a table can be checked according to the definition of 3NF of RDBM.

8. MAPPINGS BETWEEN RELATIONAL DATABASE AND RESOURCE SPACE

A relational database focuses on relations on attributes of resources, while resource space focuses on classification. According to the correspondence between primitives of the two models, mappings between relational database and resource space can be established.

8.1 Mapping Relational Database onto Resource Space

Let p_1 and p_2 be two points containing resources in resource space $RS(X_1, X_2, \ldots, X_n)$. A subset of axes $\{X_1, X_2, \ldots, X_k\}$ is called the *key* of RS if $p_1.X_i = p_2.X_i$ (i.e., the projection of p_1 on X_i is equal to the projection of p_2 on X_i) for $1 \leq i \leq n$ can be derived from $p_1.X_j = p_2.X_j$ for $1 \leq j \leq k$ [Zhuge et al. 2005b]. The key of RSM is to locate resources without knowing all axes. The following is the mapping process from a relational table onto a resource space.

(1) For each table $T(A_1, A_2, \ldots, A_n)$ in the given relational database system, create a resource space $RS(X_1, X_2, \ldots, X_n)$ by naming resource space after the table name, establishing a one-to-one relationship between the axes of

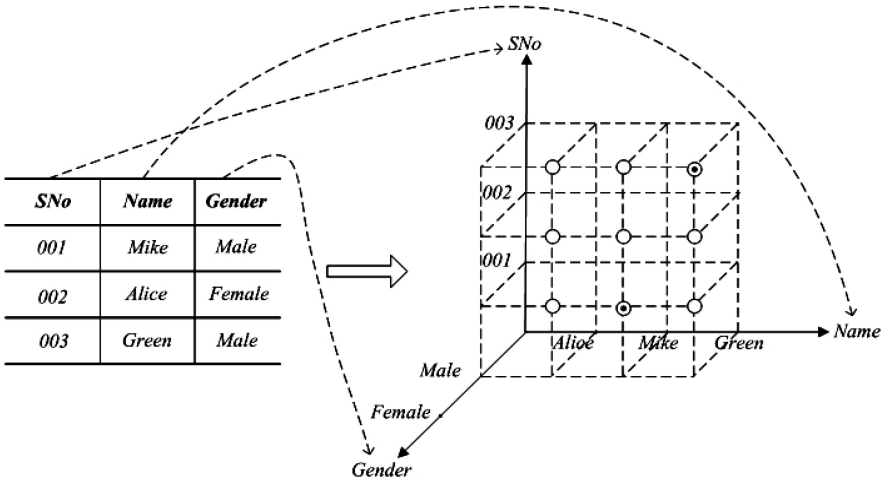


Fig. 10. An example of mapping relational table onto resource space.

the resource space and the attributes of the table (e.g., A_i corresponds to X_i), and naming each axis of this resource space after the corresponding attribute name. For each tuple $t(x_1, x_2, \ldots, x_n)$ in the table, insert x_i $(1 \leq i \leq n)$ as a coordinate into the axis X_i if no coordinate duplication exists, and then insert a resource into the point $p(x_1, x_2, \ldots, x_n)$ to represent the tuple $t(x_1, x_2, \ldots, x_n)$.

(2) For each table $T(A_1, A_2, \ldots, A_n)$, let $A_1 \ldots A_m$ $(1 \leq m \leq n)$ be the key of T. Set axes $X_1 \ldots X_m$ as the key of RS. There do not exist two different points $p(x_1 \ldots x_m, x_{m+1} \ldots x_n)$ and $p'(x_1 \ldots x_m, x'_{m+1} \ldots x'_n)$ such that both p and p' contain the resource simultaneously. Functional dependency in the relational database is represented by the classification relationship in the RSM.

Figure 10 is an example of mapping a relational table onto a resource space. A one-to-one correspondence between the basic operations {union, difference, join, cartesian product, selection and projection} of a relational database and the operations {merge, difference, join, cartesian product, selection and projection} of RSM can be established (here union corresponds to merge) [Zhuge 2004a]. Thus, any information represented by the relational database can be easily managed by the resource space.

THEOREM 4. *For any 1NF relational table without null information, the resource space generated by the mapping from relational table onto resource space satisfies the 1NF, 2NF, and 3NF of the resource space model.*

PROOF. Let $T(A_1, A_2, \ldots, A_n)$ be a relational table and $RS(X_1, X_2, \ldots, X_n)$ be the resource space generated from the mapping, which enables A_i to correspond to X_i $(1 \leq i \leq n)$. The mapping from relational table onto resource space determines that resources in the RS are tuples of T. Let $R(C)$ and $R(C')$ be the resources represented by coordinates C and C' at arbitrarily selected axis X_i. For any tuple t of T, $t \in R(C)$ if and only if $t[A_i] = C$. Since $t[A_i] = C$

and $t[A_i] = C'$ cannot hold simultaneously, $t \in R(C)$ implies $t \notin R(C')$ and vice versa. So $R(C) \cap R(C') = \emptyset$ holds. Therefore RS satisfies the 2NF. For any axis X_i and any tuple t in T, there exists a coordinate on X_i such that this coordinate is the projection of t on X_i. So $t \in R(X_i)$ holds. On the other hand, $R(X_i)$ only contains those resources derived from the tuples in T. Thus, for any two axes X_i and X_j, $R(X_i) = R(X_j)$ holds. Similar to Theorem 3, $X_i \perp X_j$ holds. So RS satisfies 3NF. □

8.2 Mapping Resource Space onto Relational Database

A given resource space $RS(X_1, X_2, \ldots, X_n)$ can be represented by three relational tables. The following two relational tables represent the schema information of the given resource space.

(1) Axis table *AT*(*AxisName*, *isKey*) is for recording all the axes' names of the given resource space, and it has two fields: *AxisName* (*varchar* type) and *isKey* (Boolean type). *AxisName* is the key of the table *AT* denoting the axis names (e.g., $X_1, X_2, \ldots$) and *isKey* can specify whether an axis is a primary axis.

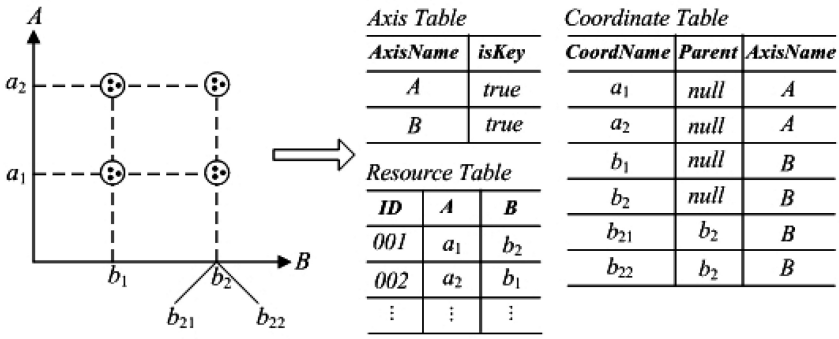
(2) Coordinate table *CT*(*CoordName*, *Parent*, *AxisName*) is for representing all the coordinates of the given resource space and includes three fields: *CoordName* (*varchar* type), *Parent* (*varchar* type), and *AxisName* (*varchar* type). *CoordName* represents the coordinate names appearing in *RS*, and *Parent* denotes the parent coordinate of a given coordinate. The *AxisName* represents where the given coordinate resides. The *CoordName* and *AxisName* comprise the key of *CT* and *AxisName* is the foreign key of *CT* referring to *AT*. Thus, once an axis in *AT* is deleted, all coordinates at this axis in *CT* will be deleted automatically, according to the reference integrity of the relational database.

The third relational table *RT* is used to represent all those resources appearing in the given resource space. For the resource space $RS(X_1, X_2, \ldots, X_n)$, we construct a resource table *RT* having n+1 fields: (*ID*, $X_1, X_2, \ldots, X_n$). Moreover, the field *ID* is the key of the table *RT*. Any resource appearing in *RS* can be inserted into the resource table.

Mapping from the given resource space onto relational tables is illustrated by Figure 11. In the following, we show that all the RSM operations (join, disjoin, merge, and split) can be simulated by the operations on the generated tables.

Let $RS_1(X_1, X_2, \ldots, X_n)$ and $RS_2(Y_1, Y_2, \ldots, Y_m)$ be two resource spaces containing the same type of resources. The relational tables corresponding to RS_1 are AT_1(*AxisName*, *isKey*), CT_1(*CoordName*, *Parent*, *AxisName*), and RT_1(*ID*, $X_1, X_2, \ldots, X_n$). The relational tables corresponding to RS_2 are AT_2(*AxisName*, *isKey*), CT_2(*CoordName*, *Parent*, *AxisName*), and RT_2(*ID*, $Y_1, Y_2, \ldots, Y_m$). The RSM operations can be simulated by the following operations on relational tables.

(1) *Join*. Suppose that RS_1 and RS_2 satisfy the join condition and $X_1 = Y_1, \ldots, X_p = Y_p$. Let RS be the join of RS_1 and RS_2, namely, $RS_1 \cdot RS_2 \Rightarrow RS$.



Axis Table

AxisName	isKey
A	true
B	true

Resource Table

ID	A	B
001	a_1	b_2
002	a_2	b_1
⋮	⋮	⋮

Coordinate Table

CoordName	Parent	AxisName
a_1	null	A
a_2	null	A
b_1	null	B
b_2	null	B
b_{21}	b_2	B
b_{22}	b_2	B

Fig. 11. An example of mapping from resource space onto relational tables.

Then, the relational tables corresponding to RS are: $AT(AxisName, isKey) = AT_1 \cup AT_2$, $CT(CoordName, Parent, AxisName) = CT_1 \cup CT_2$, and $RT(ID, X_1, \ldots, X_p, \ldots, X_n, Y_{p+1}, \ldots, Y_m) = RT_1 \bowtie RT_2$. Herein, $\cup$ and $\bowtie$ are the union and natural-join operations of the relational database, respectively.

(2) *Disjoin*. A resource space $RS(X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_m)$ can be represented by three tables $AT(AxisName, isKey)$, $CT(CoordName, Parent, AxisName)$, and $RT(ID, X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_m)$. Assume that a disjoin operation on RS generates resource space $RS_1(X_1, X_2, \ldots, X_n)$ and another resource space RS_2 sharing some axes with RS_1. Then, the relational tables corresponding to RS_1 are: $AT_1(AxisName, isKey) = \sigma_{AxisName=X1 \text{ or } \ldots \text{ or } AxisName=Xn}(AT)$, $CT_1(CoordName, Parent, AxisName) = \sigma_{AxisName=X1 \text{ or } \ldots \text{ or } AxisName=Xn}(CT)$, and $RT_1(ID, X_1, X_2, \ldots, X_n) = \pi_{ID,X1,X2,\ldots,Xn}(RT)$, where σ and π are the selection and projection operations of the relational database, respectively. RS_2 can be processed in the same way.

(3) *Merge*. Suppose that RS_1 and RS_2 satisfy the merge condition, namely, $n = m$ and $X_1 = Y_1, \ldots, X_n = Y_m$. Let RS be the merge of RS_1 and RS_2. Then, the relational tables corresponding to RS are: $AT(AxisName, isKey) = AT_1 \cup AT_2$, $CT(CoordName, Parent, AxisName) = CT_1 \cup CT_2$, and $RT(ID, X_1, \ldots, X_n) = RT_1 \cup RT_2$.

(4) *Split*. Let $RS(X_1, X_2, \ldots, X_n)$ be a resource space and its corresponding relational tables be $AT(AxisName, isKey)$, $CT(CoordName, Parent, AxisName)$, and $RT(ID, X_1, X_2, \ldots, X_n)$. Suppose that $RS_1(X_1, X_2, \ldots, X_n)$ is the result of splitting RS by deleting $C_{i1}, \ldots, C_{im}$ from X_i. Then, the relational tables corresponding to RS_1 are: $AT_1(AxisName, isKey) = AT$, $CT_1(CoordName, Parent, AxisName) = \sigma_{AxisName \neq Xi \text{ or } (CoordName \neq Ci1 \text{ and } \ldots \text{and } CoordName \neq Cim)}(CT)$, and $RT_1(ID, X_1, X_2, \ldots, X_n) = \sigma_{Xi \neq Ci1 \text{ and} \ldots \text{and } Xi \neq Cim}(RT)$.

9. INTEGRATION AND SYNERGY

9.1 Integration of Resource Space, OWL and Database

The relational database is based on functional dependence between attributes of entities and relationships between attributes and values. The RSM is based on

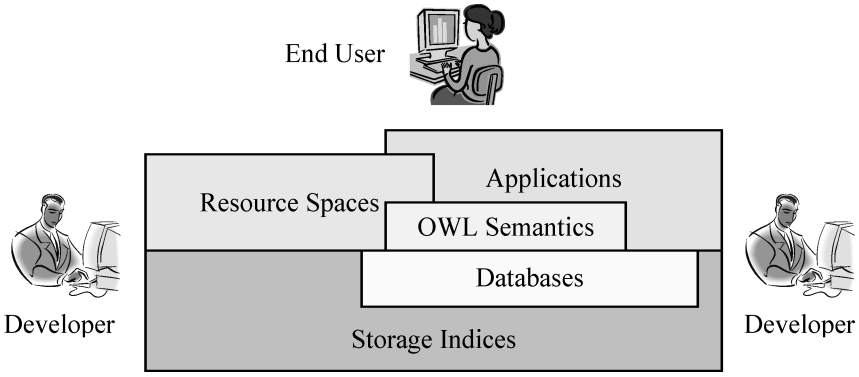


Fig. 12. A solution to integrate the resource space model, OWL, and databases.

classification and fine classification relationships. The two models use different normalization theories to ensure the correctness of the respective operations. Their common mathematical basis, namely algebra, enables one to be mapped onto the other.

As a W3C recommended standard, OWL plays an important role in developing application ontologies, which are the basis of automatically generating resource spaces and databases for applications. OWL supports relation description and reasoning, but lacks an efficient resource management mechanism. The RSM and RDBM can be used to help manage OWL files, especially large-scale OWL files.

Integrating OWL with the resource space and database can form a new semantic platform owning their advantages. Figure 12 is a general solution to this integration. OWL can represent the resource space and database in a standardized cross-platform sharable form. OWL provides fundamental semantics for resource spaces and relational databases to support autonomic normalization and advanced applications like explanation and reasoning [Codd 1970; Zhuge 2004a]. OWL also enables the integrated platform to be compatible with the standards of the semantic Web. The current approach to query over OWL description is based on graph matching (SPARQL, www.w3.org/TR/rdf-sparql-query/). For RSM, the lower bound of time complexity to search a point in a resource space is O(log$n$), where n is the number of points in the given resource space. The query efficiency can be improved by localizing queries in resource spaces. The resource space provides a normalized classification view on resources. It can be directly operated by both the end-users and the application systems. Relational databases provide a view of entity, attribute, and relation for application systems, as well as an efficient storage and management mechanism [Chen 1976]. Object-oriented databases support class-based abstraction mechanisms and modeling methodologies [Embley 1997]. The underlying storage indices realize efficient storage to support resource spaces, databases, and applications. They can be centralized or decentralized according to application requirements.

9.2 Synergy of Semantics in the Real World, Document World, and Abstraction World

Real-world semantics used by humans is hard to be understood by machines. People have developed modeling languages like UML to specify real-world semantics by standardized symbol systems like the object-oriented modeling method.

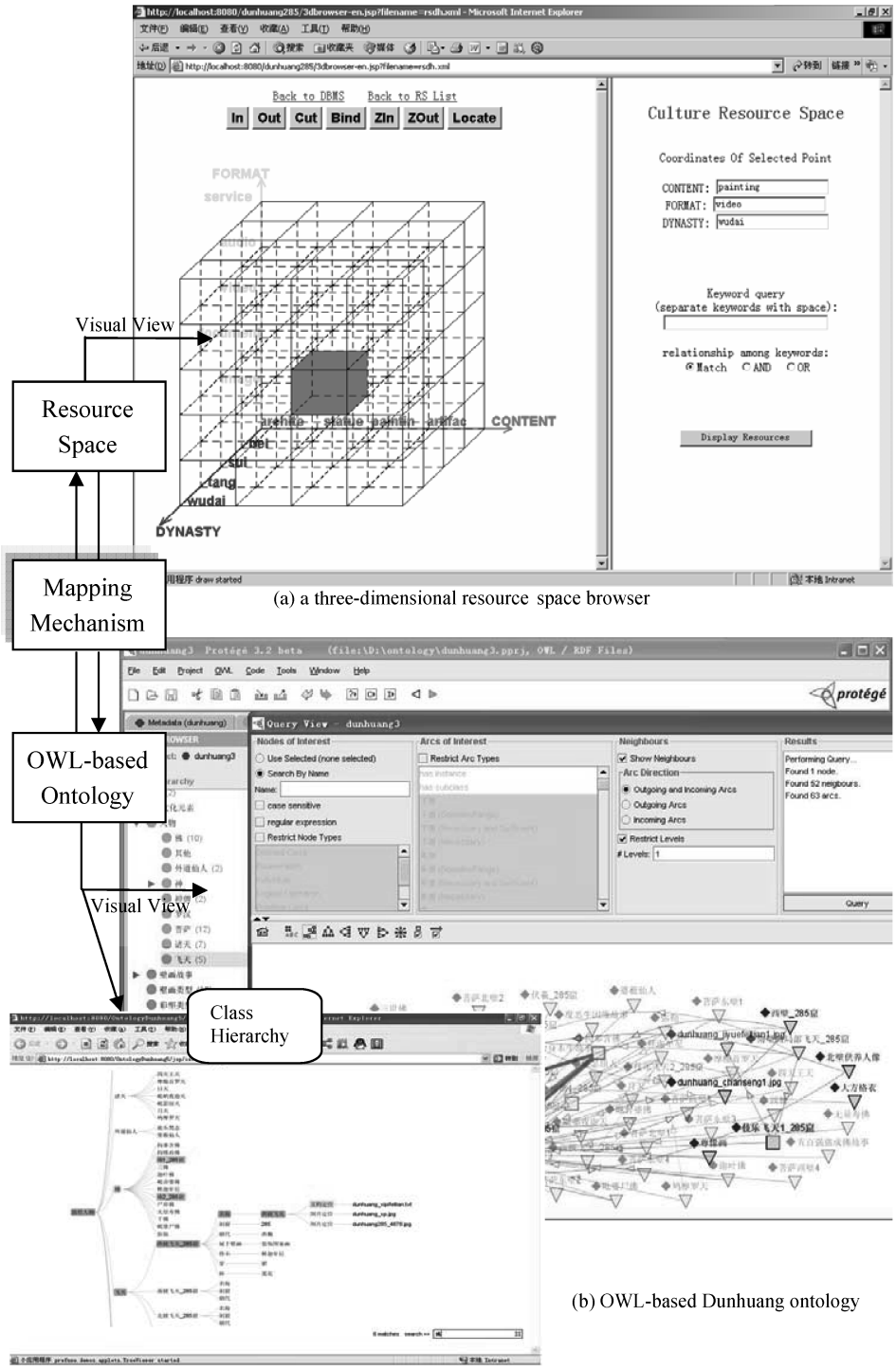
Semantics in the mental world can be intuitive or abstract. Abstract semantics takes symbolized and geometrical forms. Humans often use the classification method to recognize and manage objects in the real world. A resource space can be represented by data structures in the machine world for efficient storage and retrieval, by OWL in the document world, and by geometrical forms in the abstraction world. Figure 13(a) shows a three-dimensional resource-space browser for the Dunhuang culture exhibition. Resources can be located and manipulated by moving the black cube indicating a point in the space. The black cube can be located at any point in the space by using mouse and "In", "Out", "Cut", "Locate", "Zoom-in" and "Zoom-out" buttons. The displayed three-dimensional resource space is a geometrical view of the underlying logical higher-dimensional resource space. The proposed approach establishes the mapping between the OWL ontology and the resource space. Figure 13(b) shows the class hierarchy and the graphical expression of a part of the OWL-based Dunhuang ontology, developed by using Java and a protégé [Noy et al. 2001].

Semantics in the machine world is machine-understandable, but hard for people to understand. The Web standards XML, RDF, and OWL intermediate the machine world and the document world at different semantic levels.

Various semantics overlap and interact with each other to establish and develop the interconnection semantics, as depicted in Figure 14. The future interconnection environment needs the synergy of the diversity and uniformity of semantics in the real world, in the document world and in the mental abstraction world. Automatic mapping between semantics of different levels is an important issue.

The mappings between resource space, relational database, and OWL description support the synergy of the semantics in the machine world and in the document world. Since the resource space model is based on classification semantics, the resource space created from an OWL description does not keep all the semantics described in OWL. Mapping between UML and OWL provides ontologies for the understanding and sharing of UML and enables OWL to make use of the modeling function of UML, object-oriented methodology (OOM), and tools [Rumbaugh et al. 1991].

In the hyperlink network, we cannot derive out the hyperlink A$\rightarrow$C from hyperlinks A$\rightarrow$B and B$\rightarrow$C. The semantic link network (SLN) is an extension of the hyperlink network achieved by adding semantic factors to hyperlinks. We can derive out the semantic link A—$\alpha\rightarrow$C from semantic links A—$\beta\rightarrow$B and B—$\gamma\rightarrow$C in SLN according to some rules on α, β, and γ. SLN is rich in relational semantics and supports semantic-relation reasoning [Zhuge 2004b]. The following semantic relations can be defined as properties in OWL to realize the SLN: *causeEffect*, *similarTo*, *isPartOf*, *implication*, *sequential*, *reference*, and



(a) a three-dimensional resource space browser

(b) OWL-based Dunhuang ontology

Fig. 13. Graphical forms of semantic models.

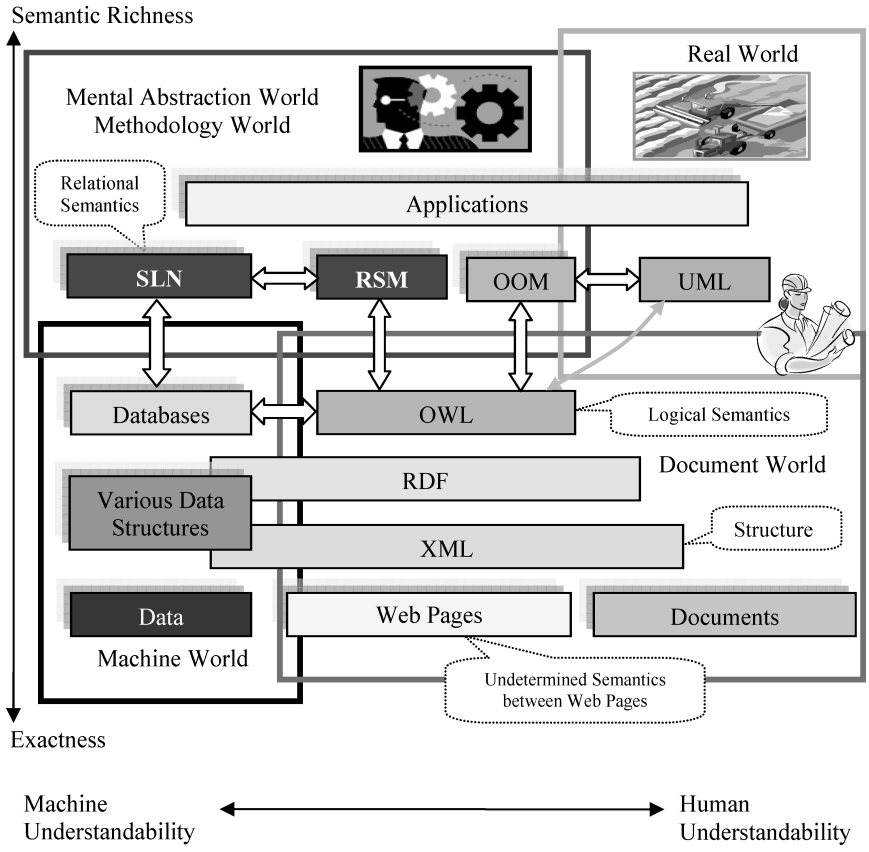


Fig. 14. Synergy semantics of four worlds in the future interconnection environment.

null. OWL enables the SLN to be machine-understandable and cross-platform sharable. Integrating the relational reasoning of the SLN and the description logics of the OWL DL can support more intelligent applications.

The relational database model, resource space model, and semantic link network are based on relational semantics. OWL includes simple classification semantics (OWL Lite) and description logics (OWL DL). The integration of these models can support intelligent applications by incorporating relational semantics with logical semantics.

10. CONCLUSION

Various semantic models have different characteristics and reflect epistemological differences. Studying the mapping between different semantic models unveils the uniformity in the diversity of semantics. The development of the Semantic Web provides a new condition for studying the mappings between various semantic models. The proposed mappings between resource space, OWL description, and relational database, as well as related strategies and theories, enable one to support the other. Investigating the relationships between

different semantic models helps people understand their differences, select appropriate semantic models for application, and make use of existing semantic models to develop new applications. Integration of resource space, OWL, and databases can form a new powerful semantic platform supporting advanced applications on the future Web. The platform can be extended by incorporating more semantic models like the semantic link network.

ACKNOWLEDGMENTS

The authors wish to thank the reviewers, who provided helpful comments on an earlier version of this article. Thanks also go to all team members of the China Knowledge Grid Research Group for their help and cooperation.

REFERENCES

ABITEBOUL, S., HULL R., AND VIANU, V. 1995. *Foundations of Databases*. Addison-Wesley, Reading, MA.

AGRAWAL, R., GUPTA, A., AND SARAWAGI, S. 1997. Modeling multidimensional databases. In *Proceedings of the 13th International Conference on Data Engineering*, IEEE Computer Society, Washington, DC, USA, 232–243.

ASSEM, M., MENKEN, M. R., SCHREIBER, G., WIELEMAKER, J., AND WIELINGA, B. J. 2004. A method for converting thesauri to RDF/OWL. In *Proceedings of the International Semantic Web Conference*, Hiroshima, Japan, 17–31.

BACHMAN, C. W. 1969. Data structure diagrams. *ACM SIGMIS Database 1*, 2, 4–10.

BATINI, C., CERI, S., AND NAVATHE, S. B. 1992. *Conceptual Database Design: An Entity-Relationship Approach*. Benjamin and Cummings, Menlo Park, CA.

BERNERS-LEE, T., HENDLER, J., AND LASSILA, O. 2001. The semantic web. *Sci. Amer. 284*, 5, 34–43.

BERNERS-LEE, T., HALL, W., HEND-LER, J. A., O'HARA, K., SHADBOLT, N., AND WEITZNER, D. J. 2006. A framework for web science. *Foundations and Trends @ in Web Science*, 1(1), 1–130.

BLAHA, M., PREMERLANI, W., AND SHEN, H. 1994. Converting OO models into RDBMS schema. *IEEE Software 11*, 3, 28–39.

BOHANNON, P., FREIRE, J., ROY, P., AND SIMEON, J. 2002. From XML schema to relations: A cost-based approach to XML storage. In *Proceedings of the 18th International Conference on Data Engineering*, San Jose, CA, 64–75.

CHEN, P. P. 1976. The entity-relationship model, towards a unified view of data. *ACM Trans. Database Syst. 1*, 1, 9–36.

CODD, E. F. 1970. A relational model of data for large shared data banks. *Commun. ACM 13*, 6, 377–387.

CODD, E. F. 1979. Extending the database relational model to capture more meaning. *ACM Trans. Database Syst. 4*, 4, 397–434.

DAS, S., CHONG, E. I., EADON, G., AND SRINIVASAN, J. 2004. Supporting ontology-based semantic matching in RDBMS. In *Proceedings of the 30th VLDB Conference*, Toronto, Canada, 1054–1065.

EMBLEY, D. W. 1997. *Object Database Development Concepts and Principles*. Addison Wesley, Reading, MA.

FUCHS, N. E., KALJURAND, K., AND SCHNEIDER, G. 2006. Bidirectional mapping between OWL DL and attempto controlled english. In *Proceedings of 4th Workshop on Principles and Practice of Semantic Web Reasoning*, Budva, Montenegro, Lecture Notes in Computer Science, Vol. 4187, Springer, 179–189.

GAŠEVIC, D., DJURIC, D., DEVEDŽIC, V., AND DAMJANOVIC, V. 2004. Converting UML to OWL ontologies. In *Proceedings of the 13th International World Wide Web Conference*, New York, 488–489.

GRØNMO, R., JAEGER, M. C., AND HOFF, H. 2005. Transformations between UML and OWL-S. In *Proceedings of the European Conference on Model Driven Architecture Foundations and Applications* (ECMDA-FA), Nuremberg, Germany. Springer.

GARCIA-MOLINA, H., ULLMAN, J. D., AND WIDOM, J. 2001. *Database Systems: The Complete Book*. Prentice-Hall.

GRUBER, T. R. 1993. A translation approach to portable ontology specifications. *Knowl. Acquisit. 5*, 2, 199–220.

HEPP, M. 2005. A methodology for deriving OWL ontologies from products and services categorization standards. In *Proceedings of the 13th European Conference on Information Systems (ECIS)*, Regensburg, Germany, 1–12.

JOHANNESSON, P. 1994. A method for transforming relational schemas into conceptual schemas. In *Proceedings of the 10th International Conference on Data Engineering,* TX, 190–201.

KALFOGLOU, Y. AND SCHORLEMMER, M. 2003. Ontology mapping: The state of the art. *The Knowl. Eng. Rev. 18*, 1–31.

KNUBLAUCH, H.,FERGERSON, R. W., NOY, N. F., AND MUSEN, M. A. 2004. The protégé OWL plugin: An open development environment for semantic web applications. In *Proceedings of 3rd International Semantic Web Conference*, Lecture Notes in Computer Science, vol. 3298, Springer, 229–243.

MARCA, D. AND MCGOWAN, C. 1987. *SADT Structured Analysis and Design Techniques*. McGraw-Hill.

MILANOVIĆ, M., GAŠEVIĆ, D., GIURCA, AND A. WAGNER G. 2006. On interchanging between OWL/SWRL and UML/OCL. In *Proceedings of the 6th OCL Workshop at the UML/MoDELS Conference (OCLApps)*, Genova, Italy.

NG, P. A. 1981. Further analysis of the entity-relationship approach to database design. *IEEE Trans. Softw. Eng. 7*, 1, 85–99.

NECHES, R., FIKES, R. E., GRUBER, T. R., PATIL, R., SENATOR, T., AND SWARTOUT, W. 1991. Enabling technology for knowledge sharing, *Artif. Intell. Mag. 12*, 3, 36–56.

NOY, N., SINTEK, M., DECKER, S., CRUBÉZY, M., FERGERSON, R. W., AND MUSEN, M. A. 2001. Creating semantic web contents with protege-2000. *IEEE Intelligent Systems 16*, 2, 60–71.

RUMBAUGH, J., BLAHA, M. R., LORENSFN, W., EDDY, F., AND PREMERIANI, W. 1991. *Object-Oriented Modeling and Design*. Prentice-Hall, NJ.

SARAWAGI, S. 1999. Explaining differences in multidimensional aggregates. In *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 42–53.

TATARINOV, I., VIGLAS, S. D., BEYER, K., SHANMUGASUNDARAM, J., SHEKITA, E., AND ZHANG, C. 2002. Storing and querying ordered XML using a relational database system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Madison, 204–215.

TEOREY, T., YANG, D., AND FRY, J. 1986. A logical design methodology for relational databases using the extended entity-relationship model. *ACM Comput. Surv. 18*, 2, 197–222.

TRINH, Q., BARKER, K., AND ALHAJJ, R. 2006. RDB2ONT: A tool for generating OWL ontologies from relational database systems. In *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW)*, Guadeloupe, French Caribbean.

VIANU, V. 2001. A Web odyssey: From Codd to XML. In *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Santa Barbara, CA, 1–15.

ZHUGE, H. 2004a. Resource space grid: Model, method and platform. *Concurrency Comput. Pract. Exper. 16*, 14, 1385–1413.

ZHUGE, H. 2004b. *The Knowledge Grid*. World Scientific, Singapore.

ZHUGE, H., YAO, E., XING, Y., AND LIU, J. 2005a. Extended resource space model. *Future Gen. Comput. Syst. 21*, 1, 189–198.

ZHUGE, H. AND XING, Y. 2005b. Integrity theory for resource space model and its application. In *Proceedings of the International Conference on Web Age Information Management (WAIM)*, Lecture Notes in Computer Science, vol. 3739, Springer, 8–24.

ZHUGE, H., SHI, P., XING, Y., AND HE, C. 2006. Transformation from OWL to RSM. In *Proceedings of the 1st Asia Semantic Web Conference (ASWC)*, Lecture Notes in Computer Science, vol. 4185, Springer, 4–23.

ZHUGE, H. 2007. *The Web Resource Space Model*. Springer.

Received October 2005; revised January 2007; accepted June 2007