Available online at www.sciencedirect.com

The Journal of Systems and Software 72 (2004) 71–81

The Journal of Systems and Software

www.elsevier.com/locate/jss

Resource space model, its design method and applications

Hai Zhuge [*]

Knowledge Grid Group, Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, P.O. Box 2704-28, 100080 Beijing, PR China

Received 3 May 2002; received in revised form 7 July 2002; accepted 8 August 2002

Abstract

A resource space model (RSM) is a model for specifying, sharing and managing versatile Web resources with a universal resource view. A normal resource space is a semantic coordinate system with independent coordinates and mutual-orthogonal axes. This paper first introduces the main viewpoint and basic content of the RSM, and then proposes a four-step method for designing the logical-level resource spaces: *resource analysis*, *top-down resource partition*, *design two-dimensional resource spaces*, and *join between resource spaces*. Design strategies and tools include: reference model, analogy and abstraction strategy, resource dictionary, independency checking tool, and orthogonality checking tool. The study on using the RSM to manage relational tables shows that the RSM is also suitable for managing structured resources. Applications show that the RSM together with the proposed development method is an applicable solution to realize normal and effective management of versatile web resources. Comparisons show the differences between the proposed model and the relational data model.

Keywords: Data model; Design method; Next-generation web; Resource space model; Semantic web; Web

1. Introduction

Database theories and systems have influenced the world for nearly 40 years (Bachman, 1974; Codd, 1970). Especially, the relational database theory, model and systems have gained a great success. Object-oriented databases and object-relational databases extended the application width of the relational databases by borrowing the advantages of the object-oriented methodologies and programming languages like inheritance and encapsulation to enable complex objects to be normally managed (Kim, 1990; Rumbaugh et al., 1991; Mok, 2002). But, their shortcomings have arisen in web-based applications, which require resources to be managed in an open, distributed, platform-irrelevant and content-based way. In data warehousing and OLAP area, the multi-dimensional data model was used (Han and Kambr, 2000), but it is a read-only model so could not meet the needs of most Internet applications where resources are frequently operated.

To uniformly, normally and effectively manage versatile Web resources has become one of the key issues of the next-generation Web. The Semantic Grid VEGA-KG is our newly proposed mechanism for sharing and managing versatile web resources (Zhuge, 2002a–d). It absorbs the ideal of the Grid (http://www.gridforum.org) and the standards of the semantic web (http://www.semanticweb.org; Hendler, 2001) and adopts the new resource management model and new communication platform. VEGA-KG has two key components: (1) a resource space model (RSM) that is used for uniformly specifying and organizing resources in normal forms; and (2) a uniform resource-using mechanism that enables users to conveniently use resources in the resource space.

The RSM can uniformly specifying and managing versatile Web resources in form of information, knowledge and services (Zhuge, 2002c). Information resources refer to various types of e-files that can be transmitted through the Internet, and can be read or felt directly or indirectly. Knowledge resources refer to the concepts, axioms, rules, or methods that can be represented in a certain machine-understandable form. Knowledge can be generated from understanding the

[*] Fax: +86-1062567724.
E-mail address: zhuge@ict.ac.cn (H. Zhuge).

0164-1212/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/S0164-1212(03)00058-X

information resources or generalizing human experience. Service resources refer to the re-usable processes for performing tasks, solving problems, or processing information or knowledge resources.

A RSM has three schemas: a *user-view schema*, a *logical-level schema* and a *semantic-web view schema*. The user-view schema is a two-dimensional space reflecting the users' view of the entire resource space. It can take the form of a "resource browser" (Zhuge, 2002d). The logical-level schema (i.e., the logical-level resource space) is an *n*-dimensional resource space, reflecting a universal view of the resource space. The semantic view schema is a semantic-based representation and organization mechanism of the resources.

The characteristics of the RSM require a special design method to assist designers to carry out resource space design. Before presenting the design method, we first introduce the main notion and basic content of the RSM.

2. Resource space model, RSM

The RSM is based on the following methods and viewpoints:

(1) *Uniform resource abstraction*. Mapping versatile resources (information, knowledge and service) into a uniform semantic space. The proposed method use a uniform set of attributes to informally describe resources.
(2) *Resource partition*. Resources could be partitioned in a given domain.
(3) *Uniform resource operation*. Resources can be operated by a uniform set of operations.
(4) *Universal resource view*. Users could operate any resources distributed on the whole Internet.

A set of common attributes can be generalized from versatile resources: {*name*, *author*, *owner*, *abstract*, *version*, *location*, *privilege*, *access-approach*, *effective-duration, related material*}, where the abstract means by the content abstraction of information and knowledge, or the function description of a service, it can be formal or informal. The access privilege includes three types: (a) *public*, any user can access to it; (b) *group*, only group members can access to it; and (c) *private*, only the author can access to it.

The basic concepts and notations are defined as follows:

1. A *resource space* is an *n*-dimensional space where every point uniquely determines one resource or a set of inter-related resources, denoted as $\mathrm{RS}(X_1, X_2, \ldots, X_n)$ or just by name RS in simple. X_i is the name of an axis. $X_i = \langle C_{i1}, C_{i2}, \ldots, C_{in}\rangle$ represents an axis with its coordinates and the order between them. C denotes the coordinate name in form of a noun or a noun phrase. Any name corresponds to a formal or an informal semantic definition in its domain ontology.
2. A coordinate C represents a class of resources, denoted as $R(C)$. A coordinate C is called independent from another coordinate C' if C is neither the synonym nor the near-synonym of C' in the discussion domain ontology.
3. Two axes are called the same if their names are the same and the names of all the corresponding coordinates are the same in a discussion domain ontology.
4. If two axes $X_1 = \langle C_{11}, C_{12}, \ldots, C_{1n}\rangle$ and $X_2 = \langle C_{21}, C_{22}, \ldots, C_{2m}\rangle$ have the same axis name but have different coordinates, then they can be merged into one: $X = X_1 \cup X_2 = \langle C_{11}, C_{12}, \ldots, C_{1n}, C_{21}, C_{22}, \ldots, C_{2m}\rangle$ whose order consists with the order of $\langle C_{11}, C_{12}, \ldots, C_{1n}\rangle$ and $\langle C_{21}, C_{22}, \ldots, C_{2m}\rangle$.
5. An axis X can be split into two axes X' and X'' by dividing the coordinate set of X into two: the coordinate set of X' and that of X'', such that $X = X' \cup X''$.

Definition 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 *fine classifies* C'_i (denoted as C'_i/X) if and only if:

1. $R(C_1) \cap R(C'_i) \neq \mathrm{NULL}$, $R(C_2) \cap R(C'_i) \neq \mathrm{NULL}, \ldots$, and $R(C_n) \cap R(C'_i) \neq \mathrm{NULL}$;
2. $(R(C_1) \cap R(C'_i)) \cap (R(C_2) \cap R(C'_i)) \cap \cdots \cap (R(C_n) \cap R(C'_i)) = \mathrm{NULL}$; and
3. $R(C_1) \cap R(C'_i) \cup R(C_2) \cap R(C'_i) \cup \cdots \cup R(C_n) \cap R(C'_i) = R(C'_i)$ hold.

As the result of the fine classification, $R(C')$ is classified into n categories: $R(C'_i/X) = \{R(C_1) \cap R(C'_i), R(C_2) \cap R(C'_i), \ldots, R(C_n) \cap R(C'_i)\}$.

Definition 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 fine classifies X' (denoted as X'/X) if and only if X fine classifies $C'_1, C'_2, \ldots, C'_m$.

Definition 3. Two axes X and X' are called *orthogonal* with each other (denoted as $X \perp X'$) if X fine classifies X' and vice versa, i.e., both X'/X and X/X' hold.

In order to answer the question of what is a good design of the resource space, we define the following three normal forms of the resource space.

Definition 4. The *first-normal-form* of a resource space is a resource space and there does not exist name duplication between coordinates at any axis. The *second-normal-form* of a resource space is a first-normal-form and for any axis, any two coordinates are independent each other. The *third-normal-form* of a resource space is a second-normal-form and any two axes of it are orthogonal with each other.

Based on the above definitions, operations on the resource spaces can be defined.

Characteristic 1. Let $|RS|$ be the number of the dimensions of the RS. If two resource spaces RS_1 and RS_2 store the same type of resources and they have n ($n \geqslant 1$) common axes, then they can be *joined* together as one RS such that RS_1 and RS_2 share these n common axes and $|RS| = |RS_1| + |RS_2| - n$. RS is called the join of RS_1 and RS_2, denoted as $RS_1 \cdot RS_2 \Rightarrow RS$.

If RS is formed by joining RS_1 and RS_2, then RS_1 and RS_2 are called the subspaces of RS. The join operation will generate some new subspaces whose axes are not orthogonal each other, i.e., there exist meaningless correspondence between the coordinates of two axes. So coordinate reinterpretation is required in these new subspaces in this case.

Definition 5. Assume $X = (C_1, \ldots, C_n)$ and $X' = (C'_1, \ldots, C'_m)$ are two axes of a new subspace that is generated by the join operation, and X is not orthogonal with X'. The *coordinate reinterpretation* is to find a new partition (i.e., a new set of coordinates $(C''_1, \ldots, C''_n)$ of X) on X such that $X = (C''_1, \ldots, C''_n) \perp X' = (C'_1, \ldots, C'_m)$.

The reinterpretation makes an axis to be the function of the subspaces: $X(\text{subspace}) = (C_1, \ldots, C_n)$, where the *subspace* is one of the subspaces that X participates in. If the coordinate reinterpretation does not carry out, the new resource space constructed by the join operation can also play the role of forming the universal view of resources. But only those meaningful subspaces could support correct resource retrieval.

Characteristic 2. A resource space RS can be *disjoined* into two resource spaces RS_1 and RS_2 (denoted as $RS \Rightarrow RS_1 \cdot RS_2$) that store the same type of resources as that of RS such that they have n ($1 \leqslant n \leqslant \min(|RS_1|, |RS_2|)$) common axes and $|RS| - n$ different axes, and $|RS| = |RS_1| + |RS_2| - n$.

Characteristic 3. If two resource spaces RS_1 and RS_2 store the same type of resources and satisfies: (1) $|RS_1| = |RS_2| = n$; and (2) they have $n-1$ common axes, and there exist two different axes X_1 and X_2 satisfy the merge condition, then they can be *merged* into one RS by retaining the $n-1$ common axes and adding a new axis $X = X_1 \cup X_2$. RS is called the merge of RS_1 and RS_2, denoted as $RS_1 \cup RS_2 \Rightarrow RS$, and $|RS| = n$.

Characteristic 4. A resource space RS can be *split* into two resource spaces RS_1 and RS_2 that store the same type of resources as that of RS and have $|RS| - 1$ common axes by splitting an axis X into two: X' and X'', such that $X = X' \cup X''$. This split operation is denoted as $RS \Rightarrow RS_1 \cup RS_2$.

Based on the proposed model, we have designed an SQL-like resource operation language as introduced in Zhuge (2002a). A prototype of the relevant programming environment has been implemented (http://kg.ict.ac.cn).

3. Method and strategy for resource space design

The design process of a resource space consists of the following four steps: *resource analysis*, *top-down resource partition*, *design two-dimensional resource spaces*, and *join between resource spaces*.

3.1. Resource analysis

Resource analysis is to determine the application scope, survey the resources need to be managed, and then specify all the resources in a *resource dictionary* (RD). Its basic functions are to help designers to specify resources, store resources, and enable designers to edit resources. Resources in the RD form the source of a local resource space in the discussed application. The XML is eligible for specifying the structural characteristic of resources. The following is a typical XML-based resource representation.

```
<RD>
   <resource1>
      <name>name1</name>
      <author>author1</author>
      <owner>owner1</owner>
      <abstract>abstract1</abstract>
      <version>version1</version>
      <location>location1</location>
      <privilege>privilege1</privilege>
      <access-approach>access-approach1<access-
      approach>
      <effective-duration>time-duration1</effective-
      duration>
      <related-material>related-material1</related-
      material>
   </resource1>
      ......
   <resourcen>
      <name>namen</name>
      <author>authorn</author>
      <owner>ownern</owner>
      <abstract>abstractn</abstract>
```

```
        <version>version_n</version>
        <location>location_n</location>
        <privilege>privilege_n</privilege>
        <access-approach>access-approach_n<access-
        approach>
        <effective-duration>time-duration_n</effective-
        duration>
        <related-material>related-material_n</related-
        material>
    </resource_n>
</RD>
```

The RD can be managed by using the XML query languages or the other query languages like the SQL (ANSI, 1986; Bocy, 1975). The repository techniques like inexact retrieval approaches can be used to enhance the effectiveness of resource management (Zhuge, 1998, 2000). The role of the RD is similar to that of the data dictionary used for establishing relational databases except for the following two aspects.

(1) The final aim of the RD is to form the axes of resource space through defining resource classification hierarchy.
(2) The resources in the RD are uniformly specified.

Fig. 1 shows the interface of a RD tool. The dictionary consists of a raw repository and a fine repository. The newly input resources are stored in the raw repository. The dictionary includes the following operations realized by the buttons on the top-portion:

(1) *Open*, create a new chapter or open an existing chapter of the given repository;
(2) *Append*, append a new resource to the given repository;
(3) *Edit*, edit the existing resources in the given repository;
(4) *Consistency checking*, check the semantic consistency among the resources (e.g., consistency among rules) in the given repository according to the resources in the fine repository and the designer's judgment;
(5) *Redundancy checking*, check the redundancy in the given repository according to the fine repository, ontology and the designer's judgment, and then delete the redundant resources;
(6) *Classification*, classify resources in the given repository according to the specialization relationship (e.g., the *is-part-of* relationship) between resources; and
(7) *Save*, store the current editing resources into the given repository. Only those resources that have passed the consistency checking and redundancy checking could be stored in the fine repository. The specialization relationship between resources can be determined according to: the existing taxonomy, the existing classification standard, the available domain ontology and the users' determination.

3.2. Top-down resource partition

Different designers may have different resource partition solutions, so a uniform viewpoint on resource partition is needed. The first step to reach a common viewpoint is to reach a common top-level resource partition agreement. *Human*, *information* and *natural* (*or artificial*) *object* are three key factors of human society, which can be regarded as the top-level resource partition of human society. The top-level resource partition of a domain can be regarded as a special case of the partition of human society. For example, the top-level resources of an institute's resources can be classified as three independent categories: *human resources*, *information resources*, and *service resources* (including facilitates). Each category can be refined top-down until the category is small enough to serve for domain applications. Fig. 2 shows an example of top-down resource partition.

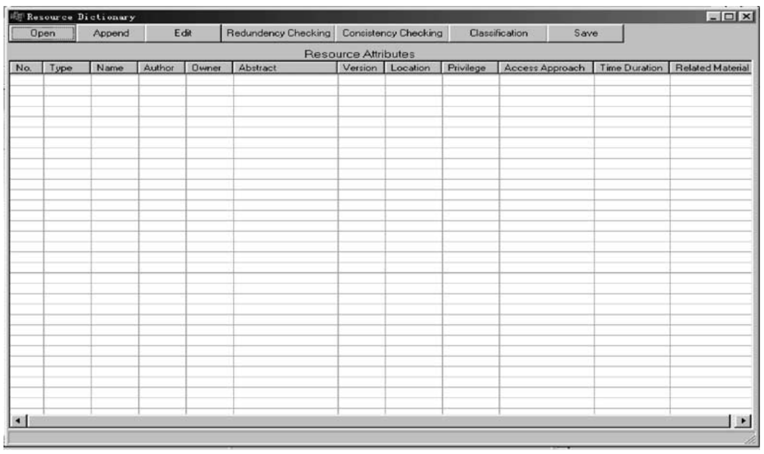


Fig. 1. Interface of the resource dictionary.

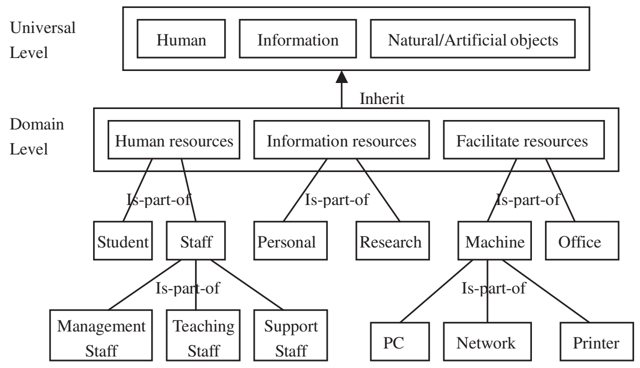


Fig. 2. An example of top-down resource partition.

3.3. Design two-dimensional resource spaces

People can better manage two-dimensional spaces than high-dimensional spaces. So we can first design a set of two-dimensional resource spaces then consider joining them into an entire resource space. The design process includes the following steps.

(1) *Determine the number of resource spaces*. The number of resource spaces can be determined according to the number of the top-level resource categories at the domain level.
(2) *Determine axes names*. The axes names can be determined according to the resource categories at the universal level or domain level. An axis name reflects a category of the domain-level partition of resources.
(3) *Determine the first-level coordinate names*. Each coordinate reflects one of the categories of the super-category determined by the axis.
(4) *Determine the coordinate hierarchies*. For each first-level coordinate, determine its low-level coordinates top-down until the basic category according to the resource partition hierarchy.
(5) *Check independency between coordinates*. Check independency between coordinates at all coordinate levels. In case the independency is not satisfied, reconsider resource partition at this level and then adjust coordinates.
(6) *Check orthogonal relationship between axes*. In case the orthogonality is not satisfied, re-consider the coordinate settings, and then adjust the relevant coordinates.

According to the above design process, we can construct two resource spaces as shown in Fig. 3 for the resource partition example of Fig. 2.

3.4. Join between spaces

In order to obtain the effect of the universal resource view, we need to join the generated two-dimensional spaces. The condition of the join needs to be checked according to Characteristic 1 introduced in Section 2. For example, the two resource spaces of Fig. 3 share a common axis. So they can be joined into a three-dimensional resource space as shown in Fig. 4. The join operation will generate new two-dimensional spaces. In the newly formed two-dimensional space: (*information*, *facilitate*), the coordinates of the information axis need to be reinterpreted according to the coordinates at the *facilitate* axis. For example, *PersonalInfor* and *ResearchInfor* can be reinterpreted as *upgrade-record* and *function-specification*. Therefore, the coordinates of the "information" axis are the function of two subspaces: (*human*, *information*) and (*information*, *facilitate*), i.e., *Information* ((*human*, *information*)) = (*PersonalInfor*, *ResearchInfor*) and *Information* ((*information*, *facilitate*)) = (*upgrading-record*, *function-specification*).

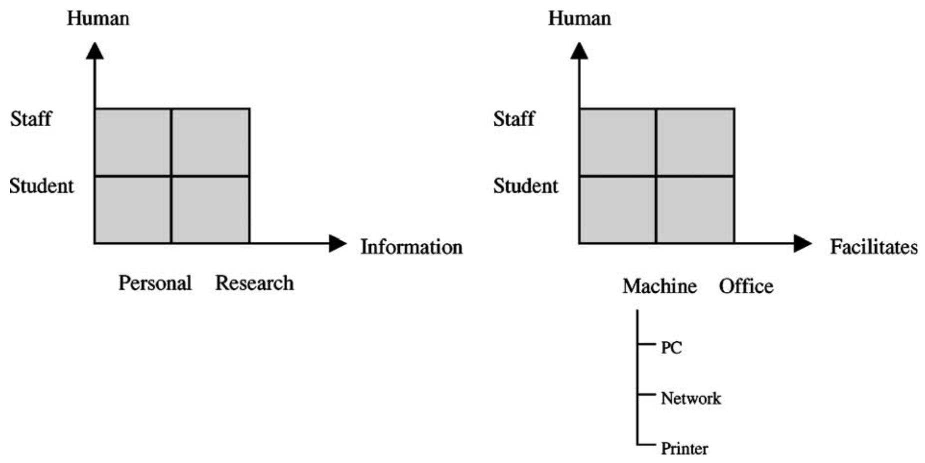


Fig. 3. Generate two-dimensional resource spaces.

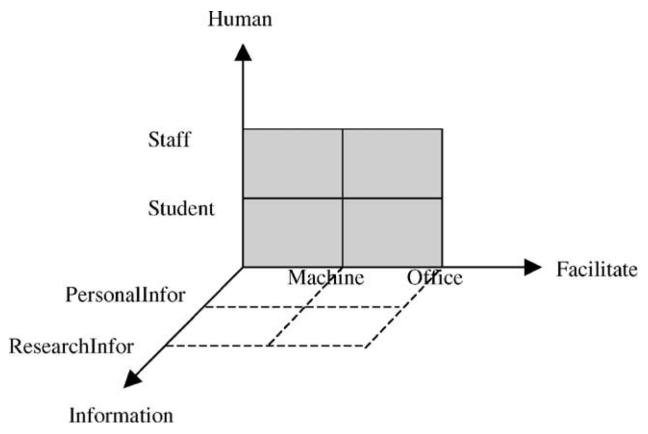


Fig. 4. A three-dimensional resource space constructed by joining two two-dimensional spaces.

3.5. Design strategy 1: make use of reference resource space

The reference resource space is two-dimensional: RS = $(category, level)$. The category dimension is the vertical partition of resources. A coordinate of the category axis represents a certain type of category, which can include several subcategories, and each subcategory can further include several smaller subcategories. A category together with its all-level subcategories constitutes a category hierarchy. So the coordinates of the category axis should be designed as scalable because people may concern resources across different categories. Except for the basic subcategories, each coordinate of the horizontal axis can be drilled down onto a set of low-level coordinates, which can be then drilled down again or rolled back up to its up-level coordinates. The top-level coordinates of the horizontal axis are the roots of all the category hierarchies.

The level dimension is a horizontal partition of resources. A coordinate of this dimension represents a level of a certain type, e.g., a *granularity level* or an *abstraction level*. Each level can be further a coordinate hierarchy like the category axis. The order between the levels can be designed as a bottom-up support relationship. For example, we have designed a knowledge space with the following four knowledge levels: *concept level*, *axiom level*, *rule level*, and *method level*, where the low-level knowledge resources can provide a certain support to the high-level knowledge resources (Zhuge, 2002a).

3.6. Design strategy 2: using abstraction and analogy strategy during design process

Abstraction and analogy are human problem-solving skills and strategies. Abstraction is usually used in combination with analogy. They can be applied to raise the efficiency of software process (Zhuge et al., 1997). Similarly, establishing abstraction and analogy relationship between the existing (reference) resource spaces and the new resource space is an experience-based way to design a new resource space. Fig. 5 graphically shows the method for designing a new resource space by analogous to the experience. The designers could get the support from the tool level, the method level, the model level, and the experience level when solving the problem of designing a new resource space for the new domain.

3.7. Design tools: independency checking tool and orthogonality checking tool

The *independency checking tool* (ICT) is to assist designers to check the independency between the coordinates of the resource space according to the synonym relationship between coordinates in the context of

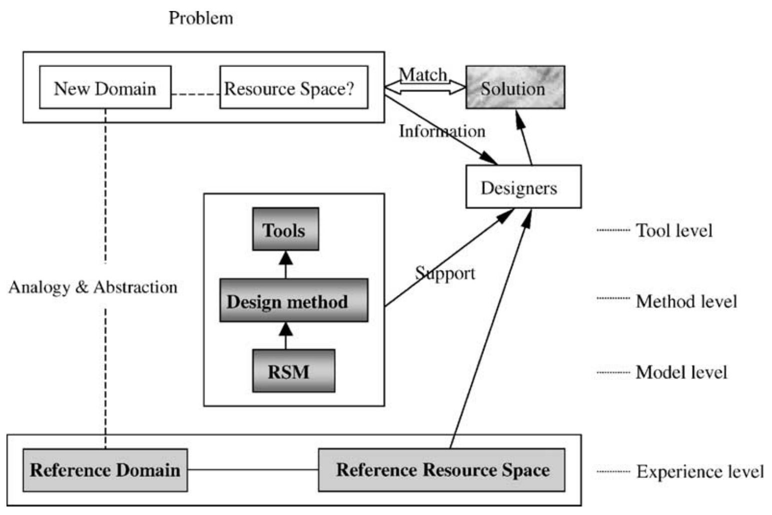


Fig. 5. The method for designing a new resource space.

certain domain ontology. The fine repository of the RD and domain ontology are the basis of determining the independency between two coordinates. Designers also need to participate the judgment in case automatic determination is difficult to carry out.

The *orthogonality checking tool* (OCT) is to check the orthogonal relationship between the axes of the resource space according to Definitions 1–3. Designers also need to participate the judgment in case automatic determination is hard. The independency checking should carry out before the checking of orthogonality.

4. Case study: use RSM to manage relational tables

A relational table can be transformed to a RSM. A relational table consists of one or several keys and a set of attributes dependent on the key(s). It can be transformed to a RSM with a key dimension and an attribute dimension denoted as follows: $\text{Table}_1(\text{Key}, A_1, A_2, \ldots, A_n) \Rightarrow \text{RSM}(\text{Key}, A(A_1, \ldots, A_n))$. The coordinates of the attribute dimension are the attributes, and the coordinates of the key dimension are the values of the key as shown in Fig. 6.

Let us first consider a first-normal-form table with one key: $\text{Table}_1(\text{Key}, A_1, A_2, \ldots, A_n)$, and we can transform it to $\text{RSM}(\text{Key}, A(A_1, \ldots, A_n))$. Since any attribute is atomic in a first-normal-form relational table, so there does not exist name duplication between $A_1, A_2, \ldots, A_n$, and they are independent each other, so it satisfies the first-normal-form and the second-normal-form of the corresponding RSM.

A second-formal-form relational table should guarantee any attribute completely depends on the Key, i.e., $\text{Key} \rightarrow A_1, A_2, \ldots, A_n$, this implies that the key's values constitute a classification of $A_i (i = 1, 2, \ldots, n)$ and A_i constitutes a classification of the *Key*, so the *key* is orthogonal with the attributes. Hence, it is the third-normal-form RSM.

In case a table has more than one key, it can be transformed into a RSM with an attribute dimension and the other dimensions for the keys. The coordinates of a key dimension are the key values. For example, a table with two keys can be transformed to a three-dimensional resource space: $\text{Table}_2(\text{Key}_1, \text{Key}_2, A_1, A_2, \ldots, A_n) \Rightarrow \text{RSM}(\text{Key}_1, \text{Key}_2, A(A_1, \ldots, A_n))$, as shown in Fig. 7. According to Characteristic 2, this three-dimensional space can be split into two-dimensional resource spaces each of which has only one key-dimension. According to the transitivity of orthogonal relationship between axes, we have the following conclusion.

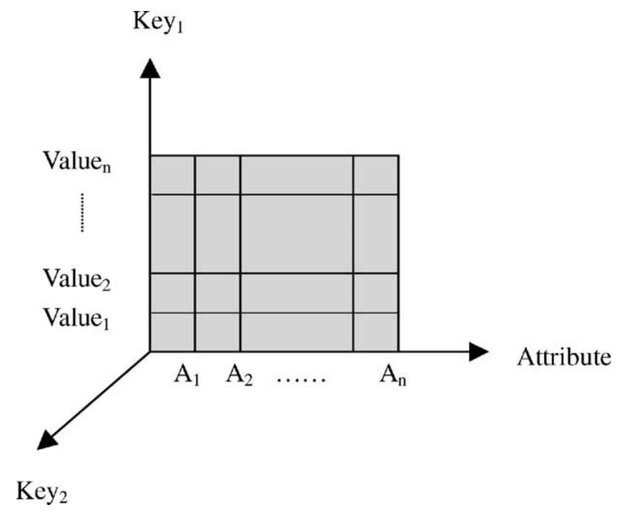


Fig. 7. A relational table with two keys can be transformed to a three-dimensional resource space.

Lemma 1. *If a table satisfies the first/second/third-normal-form of relational data model then it can be transformed to a RSM that satisfies the first/second/third-normal-form.*

The RSM can manage multiple relational tables by using a three-dimensional table as shown in Fig. 8, where the table-name dimension denotes all the tables that need to be managed. Each coordinate at the table-name dimension corresponds to a two-dimensional

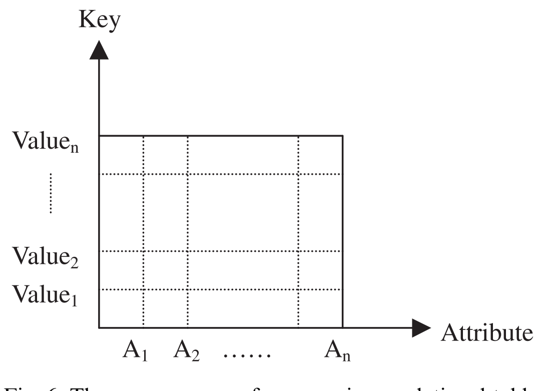


Fig. 6. The resource space for managing a relational table.

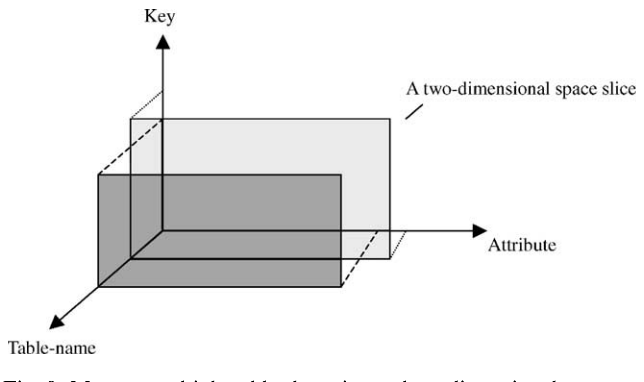


Fig. 8. Manage multiple tables by using a three-dimensional resource space.

space slice with a key dimension and an attribute dimension that represent a relational table.

5. Applications

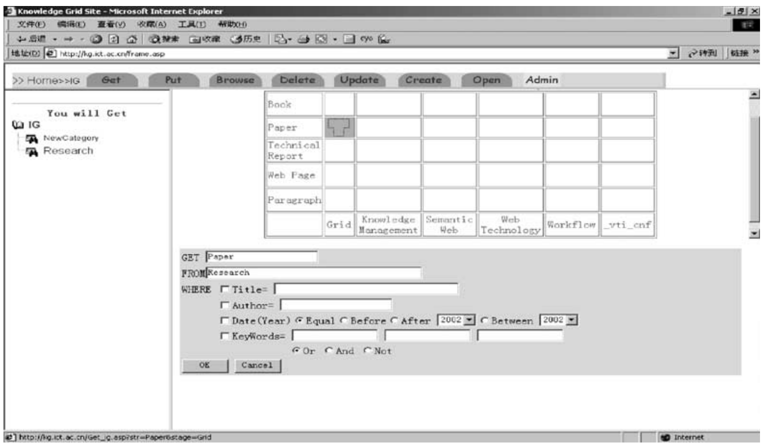
5.1. Application 1: design knowledge space, information space and service space

As an application of the proposed model and the design method, we have designed a knowledge space, an information space and a service space, and used them in the Knowledge Grid, the Information Grid and the Service Grid respectively for realizing knowledge sharing, information sharing and service sharing across the Internet. Currently these platforms are available for public use at http://kg.ict.ac.cn.

The operation interface of the Information Grid, the Knowledge Grid and the Service Grid are respectively shown in Figs. 9–11, where the middle-portions show the user-view schemas of the respective information space, knowledge space and service space, which herein is the same as the resource-space view schemas. The semantic-web view schemas of these spaces are realized by the XML. Information, knowledge and service resources can be uniformly and accurately identified by selecting the proper rectangle representing a point in the space when carrying out operations. The inexact retrieval of services has been realized in the Service Grid. The low-portion of Fig. 11 shows the interface for inexact service operations.

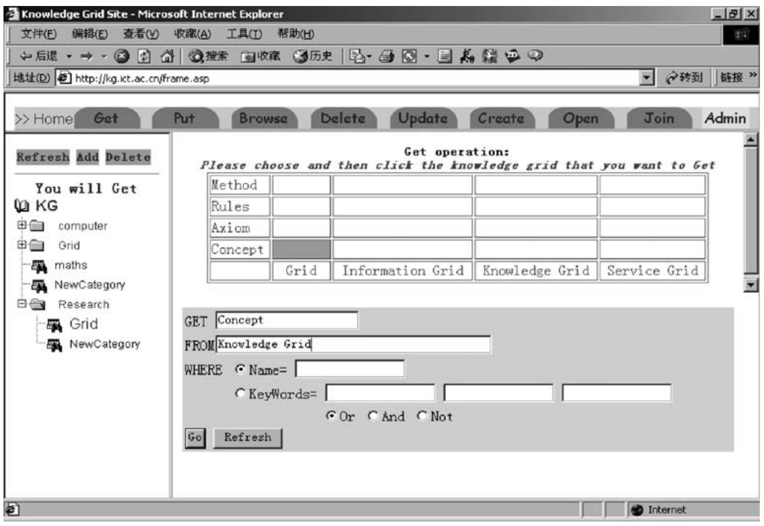
5.2. Application 2: reform "ACM Computing Classification System"

We have used the proposed approach to reform the HTML-based "ACM Computing Classification System" into a normalized three-dimensional information space: (*Category*, *Publication*, *Letter*), which is graphically shown in Fig. 12. The space satisfies the third-



GET Paper
FROM Research
WHERE ☐ Title=
☐ Author=
☐ Date(Year) [✓ Equal] Before After 2002 Between 2002
☐ KeyWords=
[✓ Or] And Not
OK Cancel

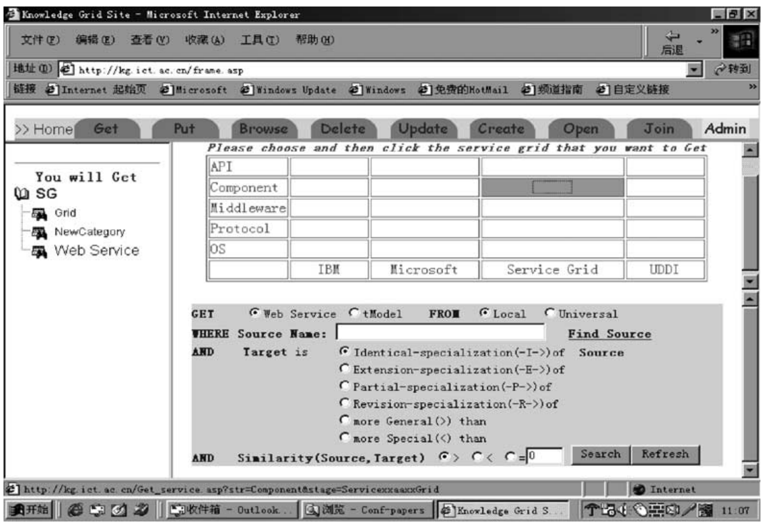
Fig. 9. Interface of Information Grid.



Get operation:
Please choose and then click the knowledge grid that you want to Get

GET Concept
FROM Knowledge Grid
WHERE [✓ Name=]
KeyWords=
[✓ Or] And Not
Go Refresh

Fig. 10. Interface of Knowledge Grid.


Knowledge Grid Site - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
地址(D) http://kg.ict.ac.cn/frame.asp
转到
链接 Internet 起始页 Microsoft Windows Update Windows 免费的HotMail 频道指南 自定义链接
>> Home
Get
Put
Browse
Delete
Update
Create
Open
Join
Admin
You will Get
SG
Grid
NewCategory
Web Service
开始
收件箱 - Outlook...
浏览 - Conf-papers
Knowledge Grid S...
Internet
11:07

Please choose and then click the service grid that you want to Get

API				
Component				
Middleware				
Protocol				
OS				
	IBM	Microsoft	Service Grid	UDDI

```
GET    (●) Web Service  ( ) tModel    FROM  (●) Local  ( ) Universal
WHERE  Source Name: [          ]              Find Source
AND    Target is  (●) Identical-specialization(-I->)of   Source
                  ( ) Extension-specialization(-E->)of
                  ( ) Partial-specialization(-P->)of
                  ( ) Revision-specialization(-R->)of
                  ( ) more General(>) than
                  ( ) more Special(<) than
AND    Similarity(Source,Target)  (●)>  ( )<  ( )=[0]   [Search] [Refresh]
```

http://kg.ict.ac.cn/Get_service.asp?str=Component&stage=ServicexxaaxxGrid

Fig. 11. Operation interface of Service Grid.

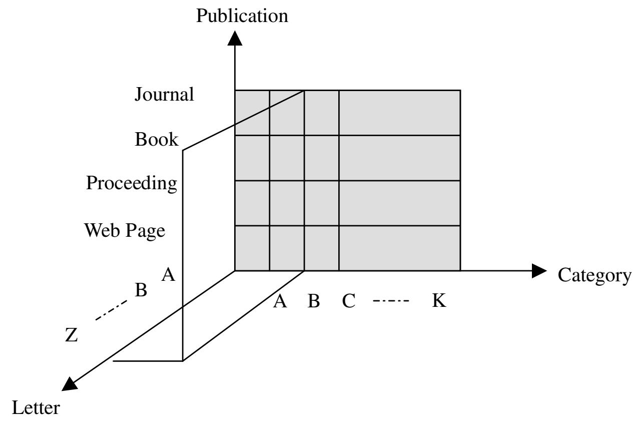


Fig. 12. A three-dimensional information space for the "ACM Computing Classification System".

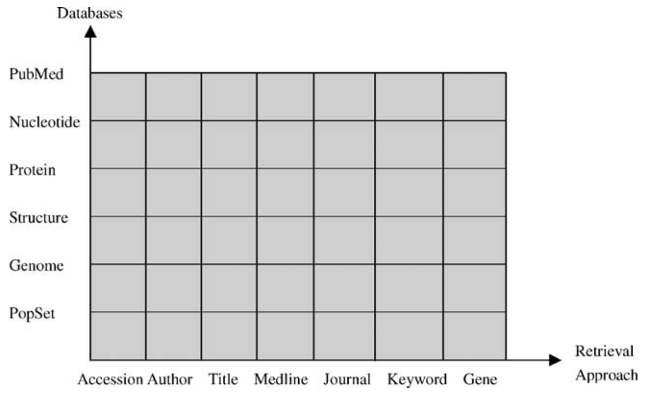


Fig. 13. A two-dimensional information space for uniformly and normally managing bio-information.

normal-form according to Definition 4. The category axis contains 11 categories marked by the letters from "A" to "K", each of which corresponds to a coordinate hierarchy. Each coordinate at the category axis corresponds to a two-dimensional slice (*Publication*, *Letter*), so users can retrieve the required information according to the publication types and/or the letter sequence in the given category. This feature is not provided by the existing classification system. The proposed approach enables information retrieval to carry out in a three-dimensional space, which could better meet users' needs. For the purpose of raising the retrieval efficiency, we can add a new axis: *Hot-topic* = (*methodology*, *theory*, *application*, *product*) to refine the space.

5.3. Application 3: realize uniform management of bio-information

We have also applied the proposed approach to reform the existing bio-information retrieval and management paradigm. Previously, the bio-information on the web is managed by versatile databases developed by different countries. Using the proposed method, all the bio-information databases can be uniformly and normally specified. Bio-information could be managed by a two-dimensional resource space as shown in Fig. 13, where "PubMed", "Structure", "Genome" and "PopSet" respectively stand for: biomedical literature, macromolecular structure, complete genome assemblies, and population study data sets. The introduced Information Grid platform enables the globally distributed bio-information to be uniformly, normally and effectively managed.

6. Comparisons

The differences between the RSM and the relational database consist of the following aspects.

(1) The objects managed by the RSM can be structured or semi-structured information, knowledge, and

Table 1
Design method comparison

No.	Items	Design method for RSM	Design method for RDB
1	Analyzed object	Resources	Entity and relationship
2	Conceptual model	No	ER model
3	Semantic basis	Partition	Function dependency
4	Purpose of normal forms	Raise preciseness of resource operations	Raise correctness of data operations
5	Reference model	Yes	No
6	Data organization	Hierarchy of top-down partitioning	Flat table
7	Schemas	User-view level, logical level, semantic-web view level	Conceptual level, logical level, physical level
8	Assistant tools	Resource dictionary, independency checking, orthogonal checking	Data dictionary

resources, while the RDBMS only manages the atomic data;

(2) The data model of the RSM is a uniform coordinate system, while the data model of the RDBMS is relational table;
(3) The normalization basis of the RSM is the independent and orthogonal coordinate system, while the normalization basis of the RDBMS is the function dependence relationship. The above three differences determine that the RSM concerns the contents (semantics) of resources and the content-based classification so supports content-based operation, but the RDBMS concerns the attributes of the objects being managed so supports attribute-based operation;
(4) The RSM enables a uniform and universal resource view when operating resources, while the RDBMS essentially supports the view of a single table. This feature enables the RSM to uniformly share and manage the Internet resources.

The major differences between the design methods for the relational databases and the RSM include eight items as shown in Table 1. The design method for the RSM does not have the conceptual model so the experience and reference model play an important role when designing a proper resource space. The hierarchical resource organization approach is in line with the top-down resource partition and the "from general to special" human thinking. The RSM is established at the semantic web level, so it does not have the physical level schema as the relational data model. The design of the RSM concerns RD, independency checking of coordinates, and orthogonal checking of axes. The design of the relational database concerns the data dictionary and the balance between the normal forms and the retrieval efficiency with respect to the application domain.

The normalization basis of the proposed RSM is based on the semantics of the resource classification, while the normalization basis of the relational database model is based on the semantics of the function dependency between data fields. This difference requires the RSM to have a special design method and tools. On the other hand, an effective and mobile resource management model should have multiple semantic levels. To investigate the resource management model at the other semantic levels is our ongoing work. The current XML-based resource space representation approach could be updated without affecting the proposed method with the evolution of the semantic web (Decker, 2000; Hendler, 2001; McHraith et al., 2001; Klein, 2001).

7. Concluding remarks

The paper first introduces the RSM for uniformly managing versatile web resources and then proposes the method for designing resource spaces. The design method integrates the assistant tools, the experience-based design process and strategy, and the reference model of the RSM. This investigation has also reached the following conclusions: (1) a relational table can be transformed to a resource space; (2) the transformation can keep the normal form correspondence between the relational model and the RSM; (3) a three-dimensional resource space can manage multiple relational tables; and (4) the application width of the RSM is wider than that of the relational data model. Applications in managing knowledge, information and service resources show that the proposed model and design method are feasible.

Acknowledgements

This work was supported by the National Science Foundation and the National Basic Research Plan of China.

References

ANSI, 1986. The Database Language SQL. Document ANSI X3.315.
Bachman, C., 1974. The data structure set model. In: Proceedings of the ACM SIGMOD, Debate on Data Model: Data Structure Set Versus Relation.
Bocy, R. et al., 1975. Specifying queries as relational expressions. Communications of the ACM 18, 621–628.

Codd, E.F., 1970. A relational model of data for large shared data banks. Communications of the ACM 13, 377–387.

Decker, S. et al., 2000. The semantic web: the roles of XML and RDF. IEEE Internet Computing (Sep/Oct), 63–74.

Han, J., Kambr, M., 2000. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers. Available from <http://www.mkp.com/>.

Hendler, J., 2001. Agents and the semantic web. IEEE Intelligent Systems 16 (2), 30–37.

Kim, W., 1990. Introduction to Object-oriented Databases. MIT Press, Cambridge.

Klein, M., 2001. XML, RDF, and relatives. IEEE Internet Computing (March/April), 26–28.

McHraith, S.A., Son, T.C., Zeng, H., 2001. Semantic web services. IEEE Intelligent Systems (March/April), 46–53.

Mok, W.Y., 2002. A comparative study of various nested normal forms. IEEE Transactions on Knowledge and Data Engineering 14, 369–385.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., 1991. Object-oriented Modelling and Design. Prentice-Hall.

Zhuge, H., 1998. Inheritance rules for flexible model retrieval. Decision Support Systems 22, 379–390.

Zhuge, H., 2000. A problem-oriented and rule-based component repository. Journal of Systems and Software 50, 201–208.

Zhuge, H., 2002a. A knowledge grid model and platform for global knowledge sharing. Expert Systems with Applications 22, 313–320.

Zhuge, H., 2002b. A knowledge flow model for peer-to-peer team knowledge sharing and management. Expert Systems with Applications 23, 23–30.

Zhuge, H., 2002c. VEGA-KG: A way to the knowledge web. In: Poster Proceedings of the 11th International World Wide Web Conference, Honolulu, Hawaii, USA, May 7–11. Available from <http://www2002.org>.

Zhuge, H., 2002d. Distributed team knowledge management by incorporating knowledge flow with knowledge grid. In: Proceedings of the Second International Conference on Knowledge Management, Graz, Austria, July, pp. 218–223.

Zhuge, H., Ma, J., Shi, X., 1997. Analogy and abstract in cognitive space: a software process model. Information and Software Technology 39, 463–468.

Hai Zhuge is a professor at the Institute of Computing Technology, Chinese Academy of Sciences. His research concerns the management and computing issues of the next-generation Web, including Semantic Grid, Knowledge Grid, Knowledge Flow, component-based system and process construction, cooperative and cognitive information systems and decision-making in the context of the next-generation Web. He early proposed the notions and models of the Knowledge Grid, Intelligent Semantic Grid, and Knowledge Flow Management. He is now the leader of the China Knowledge Grid project SVEGA-KG, which includes 20 team members. He is playing the editorial roles of several international journals and served as the program committee of a number of international conferences. He is the author of one book and over 50 papers appeared mainly in leading international conferences and international journals such as: *International World Wide Web Conference*, *Communications of the ACM*, *IEEE Intelligent Systems*, *IEEE Transactions on Systems, Man, and Cybernetics*, *Computing in Science and Engineering*, *Information and Management*, *Decision Support Systems*, *Journal of Systems and Software*, *Expert Systems with Applications*, *Knowledge-based Systems*, *Information and Software Technology*, and *Lecture Notes in Computer Science*.