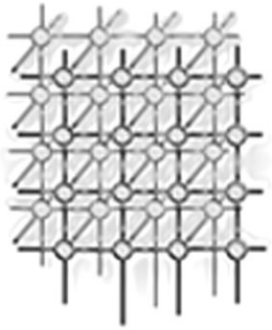
CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2004; **16**:1385–1413
Published online 14 September 2004 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe.867

Resource Space Grid: model, method and platform



Hai Zhuge*,†

China Knowledge Grid Research Group,
Key Lab of Intelligent Information Processing,
Institute of Computing Technology, Chinese Academy of Sciences,
Beijing 100080, People's Republic of China

SUMMARY

A Resource Space Grid is a virtual Grid that aims at effectively sharing, using and managing versatile resources across the Internet. The kernel of the Resource Space Grid includes a Resource Space Model (RSM) and a uniform Resource Using Mechanism (RUM). This paper presents the Resource Space Grid's core scientific issues and methodology, architecture, model and theory, design criteria and method, and practice. A normal form theory is proposed to normalize the resource space—a coordinate system for uniformly specifying and organizing resources. The RUM provides not only the end-users with an operable resource browser to operate resources using the built-in Resource Operation Language (ROL), but also the application developers with the ROL-based programming environment. The prototype platform based on the proposed model and method has been implemented and used for sharing and managing resources in distributed research teams. Operations on Resource Spaces can constitute the virtual communities of Resource Space Grids—a platform independent resource sharing environment. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: resource sharing; query language; Semantic Grid; Semantic Web; Web

1. INTRODUCTION

The World Wide Web—a global Web page repository—enables people to share information by means of HTML, HTTP, search engines and browsers. The major advantage and the success of the Web is easy—simple in architecture, usage mode and interface. The hyperlink enables 'anything' to link to 'anything' and leads to the scale-free characteristic [1]. However, Web pages are designed for humans to read, not for software to manipulate meaningfully [2]. So it is hard for the Web to support complex applications and effective information services. To overcome the shortcomings of the current Web,

*Correspondence to: Hai Zhuge, China Knowledge Grid Research Group, Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, People's Republic of China.
†E-mail: zhuge@ict.ac.cn

Received 4 June 2003
Revised 27 October 2003
Accepted 13 November 2003

people have made efforts towards the next-generation Web. The Semantic Web, the Web Service and the Grid are three of these efforts, which have attracted investment from many scientists and enterprises.

The main intention of the Semantic Web [3–6] is to enable the Web resources (e.g. Web pages) to be understood by the resource usage and service mechanisms such as the search engines and browsers. Current research on the Semantic Web focuses on ontology mechanisms and markup languages such as RDF (Resource Description Framework), OIL (Ontology Inference Layer), and DAML (DARPA Agent Markup Language). The XML-based RDF (http://www.w3c.org/rdf) defines the machine-understandable semantics of the Web resources by the object-attribute-value model [7,8]. The RDF schema (RDFS) enhances the representation ability of the RDF by providing the means to define the vocabulary, the class-based structure and the constraints for expressing the metadata about Web resources. As the extension of the markup language, the approach for representing knowledge in documents by extending the RDFS has been proposed [9]. OIL is the extension of RDFS through the well-defined XML syntax based on the document type definition [10]. DAML is the extension of the Web ontology to allow rules to be expressed within the languages [11]. Ontology can establish a certain common understanding between information-processing mechanisms. It usually contains a hierarchy of concepts of a domain and describes each concept's crucial properties through an attribute value. The WordNet (http://www.cogsci.princeton.edu/∼wn) is a kind of conceptual level ontology. Researchers have developed assistant tools for the creation and management of ontologies [10]. Approaches for representing and using knowledge in documents and the frame-based SHOE (Simple HTML Ontology Extensions) have been proposed [3,12]. Although these markup languages and ontology mechanisms can realize semantic-level resource sharing to some extent, so far we still do not have effective means to uniformly organize, accurately retrieve, and effectively manage globally distributed resources according to semantics.

The Web Services are application mechanisms that interact with each other using the Web standards for data definition and exchange (http://www.w3.org/XML/), message routing and remote method invocation (http://www.w3.org/TR/SOAP/), service description (http://www.w3.org/TR/wsdl), and services publication (http://www.uddi.org). The aim is to provide an open computing platform for the development, deployment, interaction and management of globally distributed e-services, rather than a passive repository where users primarily use it as the media of viewing and downloading content. Web Services provide the possibility of running an application on distributed machines, many of which the users neither own nor control. People can use Web Services as components to build their own applications and services. However, current Web Services still lack effective means to manage services.

Grid computing is the technology that enables a large-scale distributed computing system to carry out the controlled sharing of computing resources. The concept of Grid computing appeared in the 1995 I-WAY experiment, where high-speed networks were used to connect high-end resources at 17 sites across North America [13]. The development of Grid computing led to the establishment of the Grid forum in November 2000 (http://www.gridforum.org). Many systems that use the idea of Grid computing have been implemented [14]. Nowadays, Grid computing has become a hot research area and has attracted investment from many leading corporations.

From the architecture point of view, the computing Grid, the Web Service and the current Web belong to the tightly controlled paradigm in architecture, where strong servers play the key role. Peer-to-peer computing aims at the loosely controlled computing architecture, where a client can play the role of a server and a server can also play the role of a client. The notion of peer-to-peer also exists

at the resource-sharing and knowledge-sharing level. A model for realizing tightly coupled peer-to-peer team knowledge sharing and management was proposed in Zhuge [15].

The Semantic Web, the Web Service, the peer-to-peer, and the Grid focus on different aspects of application demands. They should converge on the same final target in the long run, and each area will benefit from the progress of other areas. For example, the extension of the Grid in semantics and the extension of the Semantic Web in computation ability converge at the Semantic Grid (http://www.semanticgrid.org). Actually, the current Grid architecture OGSA (Open Grid Service Architecture) has shifted the orientation from computing to service. The Semantic Grid has become the main architecture of the European e-science projects (http://mygrid.man.ac.uk/). However, the current Semantic Grid still lacks fundamental theory and methodology support.

China has launched a national research project on the Semantic and Knowledge Grid [16]. Its core technique is the Resource Space Grid—a virtual worldwide resource sharing and management environment.

2. CORE SCIENTIFIC ISSUES AND RESEARCH METHODOLOGY

The general research aim of the Resource Space Grid is to solve three core scientific issues: *normal organize*, *semantic interconnect* and *dynamic cluster* versatile resources across the Internet [16]. This paper focuses on the normal organization issue—how to abstract a disorder system and then content-equivalently reconstruct it in a normal space to guarantee the correctness of resource operations. An ideal approach is to seek a set of semantic normal forms to organize resources in the form that could normalize the resource organization and eliminate redundant, disorder and useless resources so as to guarantee the correctness and accuracy of resource operation, and realize complete and effective resource sharing.

Our research methodology contains the following four sets of key notions.

(1) *Virtualization, variety and uniformity*. Virtualization requires that the future interconnect environment should be platform independent and media independent. Variety has two meanings: the first is that our research objects are versatile Web resources such as information, knowledge, services and even Grid components; and, the second is the adoption of the principles and methods of multiple disciplines such as system methodology, psychology and ecology, and cross-disciplinary research. Research on future Web resources ferments a breakthrough in the background that important progress has been made in relevant disciplines. Uniformity means that versatile Web resources are managed and processed in a uniform and global view.
(2) *Enabling intelligence, personality and humanity*. The Grid platform should enable users and resources to use resources in an intelligent, in-depth, flexible, personalized and humanized way.
(3) *Global cooperation*. Resources can actively cluster together through understanding the semantics of demands and resources, and cooperate with each other to provide on-demand services.
(4) *Activity, autonomy and adaptability*. Resources can actively provide services. Communities in the future interconnection environment should be assigned the right to autonomously control the sharing process so as to raise the efficiency of sharing resources. Resource sharing and using can adapt to the change of requirement and environment.

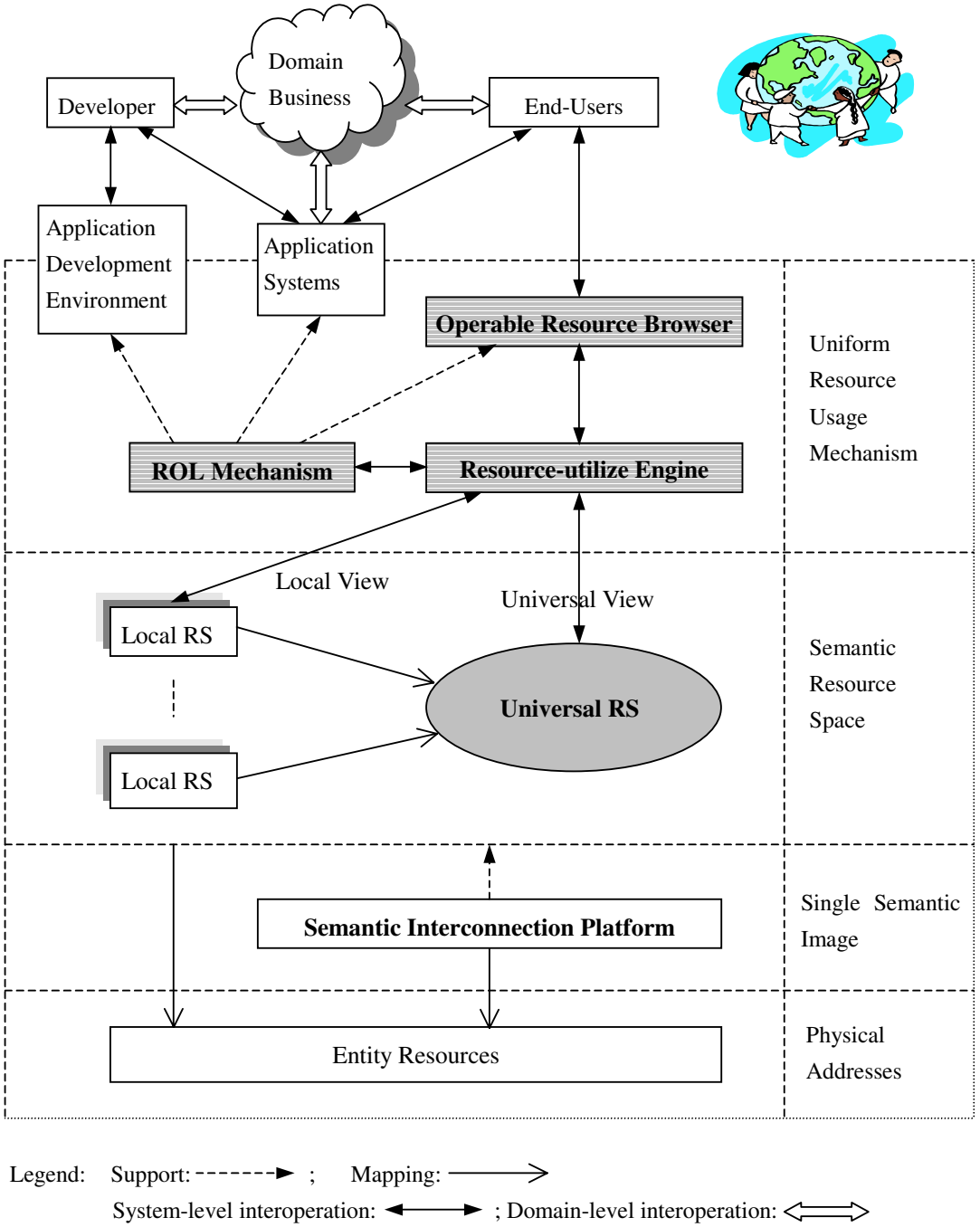
The rapid development of material science and communication technology enables the Internet to transmit information faster and faster. It is just a time issue that the future Internet will work at a speed several tens of thousand times the current speed, or much higher than that. This brings a scenario that a long waiting time (e.g. 10 min) for the same amount of information exchange will then be as short as several milliseconds or even shorter than that, so ordinary users will feel that information exchange across the Internet is being carried out as in a single PC or a local network. This provides us with the rationale to regard all the resources on the Internet as a huge resource repository. However, increasing the speed and bandwidth could never catch up with the exponentially growing resources, and does not solve semantic issues. If the semantics of versatile resources can be uniformly specified, then Internet users can share and manage the resources in the repository by using a uniform operational language, just as using SQL [17,18] to operate relational databases. Hence, we adopt the following three strategies.

(1) *Adopt the Semantic Web standard as the representation and interchange basis of resources*. This strategy enables resource sharing to be carried out on a mutually-understandable basis.
(2) *Refer to the relational data model and the SQL when developing the resource management model*. The relational data model has a firm theoretical basis and has proved to be successful since it was proposed in 1970. However, Web resources are complex, distributed, huge, heterogeneous, and exponentially growing. So the relational data model is not eligible for managing versatile resources on the Web, but we can learn from its theoretical structure. We also need to make use of the syntax and semantics of the SQL to define the operation language.
(3) *Make use of abstraction and analogy*. This strategy encourages the resource space designers to abstract common characteristics from versatile resources when organizing resources. It can also help designers to create new resource spaces for applications using past experience.

3. GENERAL ARCHITECTURE

The general architecture of the Resource Space Grid mainly consists of the Resource Using Mechanism (RUM) and the Resource Space Model (RSM), as shown in Figure 1. The RUM includes an operable resource browser, a resource-utilize engine, a Resource Operation Language (ROL), a ROL-enactment mechanism, and an application development environment. The operable resource browser provides an easy-to-use interface for the end-users to select the proper resources and the proper operation, to determine operation parameters, and then to submit the operation. The resource-utilize engine accepts the submitted operation and then performs the operations according to the types of the resources to be operated. The developers develop the application system with the help of the development environment, which is supported partially by the Resource Space Grid and partially by other application development tools. The ROL supports the end-users in carrying out one-step operation or supports application developers to compose a resource operation program to realize an application system. The ROL not only supports the application system to use resources but also helps the resource browser to operate resources. The end-users can either use the application system to process domain applications or use a resource browser to directly operate resources.

The universal resource space is the entire collection of local resource spaces. In order to correctly and efficiently operate resources, we propose a RSM—a uniform coordinate system with independent



Legend: Support: ----------► ; Mapping: ——►

System-level interoperation: ◄——► ; Domain-level interoperation: ⟺

Figure 1. Architecture of the Resource Space Grid.

coordinates and orthogonal axes for accurately identifying resources. A resource space has the following three schemas.

(1) A *user-view schema* (user level). A two- or three-dimensional resource space built in the resource browser, which enables the end-users to conveniently locate and operate resources because it is hard for end-users to deal with high-dimensional spaces.
(2) A *universal-view schema* (logical level). All the resources are located in an entire n-dimensional space. A user-view schema is a slice (or a subspace) of the universal-view schema.
(3) A *semantic-view* schema (*single semantic image* level [19]). Versatile resources are expressed in single semantic space by making use of the markup language of the Semantic Web (e.g. the Resource Description Framework (RDF)), ontology mapping, or the Semantic Link Network [20].

Every resource space has an identity and a logical location so that any client can easily access it. A resource's logical address can be a composition of its identity and coordinates. A resource has an entity located at a physical address. One-to-one and onto mapping between the logical address and the physical address enables the RUM to work at the semantic level.

4. RSM

4.1. Theory

The following common attributes can be abstracted from resources of versatile type.

(1) *Name*, an identifier for differentiating one resource from the other.
(2) *Author*, the creator's name.
(3) *Abstract*, the content abstraction of information or knowledge resources, or the function description of service resources. It can be in the form of natural language description, formal description, or template.
(4) *Version*, the version number.
(5) *Location*, the logical address in the virtual environment and the corresponding entity's residence in the Internet.
(6) *Privilege*, it has three values: (a) *public*, any user can access the resource; (b) *group*, only group members can access the resource; and (c) *private*, only the author can access the resource.
(7) *Access-approach*, the valid operations of the resource.
(8) *Effective-duration*, the life span of the resource.

Current research works on ontology support the reasonability of the following assumption used for developing the RSM.

Assumption 1. An ontology service mechanism is available: $Output = Ontology\text{-}Service(Input, k)$. The '*input*' parameter is a word or a word phrase. The second parameter is a variable with numerical type used for controlling the output. If $k = 0$, the ontology service outputs a word set with the following elements: the synonym(s), the abstract-concept(s), the specific-concept(s), and the instance(s) of the

input word. If $k = 1$, the ontology service outputs a set with one more element, the near-synonym of the input word.

Coordinates are the commonly used and efficient means to locate objects in a space. For example, people use latitudes and longitudes to accurately locate a geographical place on a map—a two-dimensional space. Establishing a resource space with the right axes and coordinates, we can accurately store and retrieve resources according to coordinates. To differentiate resources in different resource space, we use the space name to extend the resource name: Resource-name.Space-name.

Definition 1. A resource space is an n-dimensional space where every point uniquely determines one resource or a set of related resources. A resource space has a name, a type, a logical location, and the access privilege.

The uniqueness requires that the coordinate of each dimension should be defined independently. In the following discussion, we use the following representations and notations: (1) a resource space is represented by $RS(X_1, X_2, \ldots, X_n)$ or RS in simple terms, where RS is the name of the resource space and X_i is the name of an axis. We use $|RS|$ to denote the number of dimensions of the RS, i.e. $|RS| = n$; and (2) $X_i = \{C_{i1}, C_{i2}, \ldots, C_{in}\}$ represents an axis with its coordinates. C denotes the coordinate name in the form of a noun or a noun phrase. Any name corresponds to a formal or an informal semantic definition in its domain ontology, like the WordNet (http://www.cogsci.princeton.edu/∼wn/).

Definition 2. (1) Two axes are called the same if their names are the same and the names of all the corresponding coordinates are the same. (2) If two axes $X_1 = \{C_{11}, C_{12}, \ldots, C_{1n}\}$ and $X_2 = \{C_{21}, C_{22}, \ldots, C_{2m}\}$ have the same axis name but have different coordinates, they can be merged into one: $X = X_1 \cup X_2$, denoted as $X_1 \cup X_2 \Rightarrow X$.

Characteristic 1. 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 $X = X' \cup X''$.

A coordinate can be expanded as a coordinate hierarchy, where the low-level coordinates represent the subclasses of their common ancestor. We use $Sup(C)$ to denote the direct ancestor of C in a coordinate hierarchy. The name of each coordinate of the hierarchy can be differentiated from the others by connecting its name to all of its ancestors. Considering the hierarchical characteristic, the concept of coordinate is defined as follows.

Definition 3. A coordinate C represents a class of resources (denoted as $R(C)$) that satisfies the following: if $C = Sup(C')$ then $R(C') \subseteq R(C)$.

According to Definition 3, if $C = Sup(C')$ and $C' = Sup(C'')$ then $R(C'') \subseteq R(C)$.

An axis with hierarchical coordinates can be mapped onto an axis with flat coordinates. So this section focuses on the flat case. A good design for resource space should guarantee correct resource sharing and management. Synonyms like ‘teacher’, ‘instructor’ and ‘tutor’ are not eligible for being used as coordinates at the same axis because they lead to misunderstandings and incorrect operation when carrying out resource operations. The following four definitions define the semantic relationship between coordinates and the relationship between axes based on the assumption.

Definition 4. A coordinate C is independent from another coordinate C' if $C \notin Output = Ontology\text{-}Service(C', 1)$.

Definition 5. Let $X = (C_1, C_2, \ldots, C_n)$ be an axis and C'_i be a coordinate at another axis X'. We can say that X fine classifies C'_i (denoted as C'_i/X) if and only if

(1) $(R(C_k) \cap R(C'_i)) \cap (R(C_p) \cap R(C'_i)) = \phi$ ($k \neq p$, and k, $p \in [1, n]$); and
(2) $(R(C_1) \cap R(C'_i)) \cup (R(C_2) \cap R(C'_i)) \cup \cdots \cup (R(C_n) \cap R(C'_i)) = R(C'_i)$ hold.

As a 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 6. For two axes $X = \{C_1, C_2, \ldots, C_n\}$ and $X' = \{C'_1, C'_2, \ldots, C'_m\}$, we can say that X fine classifies X' (denoted as X'/X) if and only if X fine classifies $C'_1, C'_2, \ldots$, and C'_m.

Characteristic 2 (Transitivity of fine classification). If X''/X' and X'/X, then X''/X holds.

The proof of Characteristic 2 is presented in Appendix A.

Definition 7. Two axes X and X' are assumed to be 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.

For example, *KnowledgeLevel* = <*Concept*, *Axiom*, *Rule*, *Method*> $\perp$ *Discipline* = <*Computer*, *Communication*, *Ecology*, *Math*> because <*Concept*, *Axiom*, *Rule*, *Method*> constitutes a fine classification of every coordinate of the '*Discipline*' axis and the <*Computer*, *Communication*, *Ecology*, *Math*> constitutes a fine classification of every coordinate of the '*KnowledgeLevel*' axis. According to the transitivity of the fine classification, we can reach the following characteristic (proof is presented in Appendix B).

Characteristic 3. If $X \perp X'$ and $X' \perp X''$, then $X \perp X''$ holds.

To answer the question of what is a good design of the resource space, we need to define the normal forms of the resource space.

Definition 8. The first normal form of a resource space is a resource space, and there is no name duplication between coordinates at any axis.

Definition 9. The second normal form of a resource space is a first normal form, and any two coordinates are independent of each other.

The second normal form excludes the case that a coordinate includes another in semantics.

Definition 10. The third normal form of a resource space is a second normal form, and any two axes of it are orthogonal with each other.

The three normal forms provide the designers the guidelines to design a proper resource space. The first normal form is the weakest, which can avoid explicit coordinate duplication. The second normal form is stronger, which can avoid inexplicit coordinate duplication. The third normal form is the strongest, which guarantees resources are properly operated. The setting of the coordinates of a resource space is relevant to the type of resources contained.

Characteristic 4. 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 $RS_1 \cdot RS_2 \Rightarrow RS$.

Characteristic 5. 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 6. If two resource spaces RS_1 and RS_2 store the same type of resources and satisfy (1) $|RS_1| = |RS_2| = n$, and (2) they have $n - 1$ common axes, and 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 7. A resource space RS can be split into two resource spaces RS_1 and RS_2 that store the same type of resources as 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 $RS \Rightarrow RS_1 \cup RS_2$.

According to Definitions 8–10 and Characteristic 3, we can prove the following three lemmas. We omit the proofs for Lemmas 1 and 2 since they are fairly straightforward.

Lemma 1. *Let $RS_1 \cdot RS_2 \Rightarrow RS$, where RS is the first normal form if and only if both RS_1 and RS_2 are the first normal form.*

Lemma 2. *Let $RS_1 \cdot RS_2 \Rightarrow RS$, where RS is the second normal form if and only if both RS_1 and RS_2 are the second normal form.*

Lemma 3. *Let $RS_1 \cdot RS_2 \Rightarrow RS$, where RS is the third normal form if and only if both RS_1 and RS_2 are the third normal form.*

Proof.

(1) Let $RS_1(X_{11}, \ldots, X_{1n})$ and RS_2 $(X_{21}, \ldots, X_{2m})$ be two resource spaces satisfying the third normal form. Since $RS_1 \cdot RS_2 \Rightarrow RS$ holds, RS_1 and RS_2 have at least one common axis. Without losing generality, we assume $X_{1n} = X_{21} = X$. According to Lemmas 1 and 2, RS is the second normal form. According to Definition 10, we have $X_{11} \perp \cdots \perp X_{1n}$ and $X_{21} \perp \cdots \perp X_{2m}$. According to Characteristic 3 and the premise, we have $X_{11} \perp \cdots \perp X$ (X_{1n} or X_{21}) $\perp \cdots \perp X_{2m}$. Hence, RS is the third normal form.
(2) Let RS be the third normal form. According to Definition 10, Lemma 2 and Characteristic 2, We can simply say that both RS_1 and RS_2 are the third normal form. □

According to Characteristics 3–6, we have the following lemma.

Lemma 4. *We have:*

(1) *$RS \Rightarrow RS_1 \cdot RS_2$ if and only if $RS_1 \cdot RS_2 \Rightarrow RS$;*
(2) *$RS \Rightarrow RS_1 \cup RS_2$ if and only if $RS_1 \cup RS_2 \Rightarrow RS$.*

Similar to Lemmas 1–3, we can prove the following lemma. It enables a high-dimensional resource space to be disjoined into several low-dimensional resource spaces that keep the same normal form as the original resource space. For example, a five-dimensional resource space can be disjoined into two, three-dimensional resource spaces that share a common axis.

Lemma 5. *Let* $RS \Rightarrow RS_1 \cdot RS_2$.

(1) *If RS is the first/second/third normal form then both* RS_1 *and* RS_2 *are the first/second/third normal form.*
(2) *RS is the third normal form if and only if both* RS_1 *and* RS_2 *are the third normal form.*

According to the definition of the normal forms, we have the following two lemmas about the merge and split operations.

Lemma 6. *Let* $RS \Rightarrow RS_1 \cup RS_2$, *if RS is the first/second/third normal form, then* RS_1 *and* RS_2 *are the first/second/third normal form.*

Lemma 7. *Let* $RS_1 \cup RS_2 \Rightarrow RS$, *if* RS_1 *and* RS_2 *are the third normal forms, and RS is a second normal form RS, then RS is a third normal form.*

Semantic overlaps may exist between resource definitions in some applications. In this case, the resource space designers have to use near-synonyms as the coordinates at the same axis.

Definition 11. A coordinate C is called weak independent of another coordinate C' if $C \notin Output = Ontology\text{-}Service(C', 0)$.

According to Definition 9, we can relax the second normal form as the weak second normal form, based on which, we can further relax the third normal form as the weak third normal form. The weak third normal form can meet some application requirements.

Definition 12. The weak second normal form of the resource space is a first normal form and for any one of its axes, any two coordinates are weak independent of each other.

Definition 13. The weak third normal form of a resource space is a weak second normal form and any two of its axes are orthogonal with each other.

A third normal form resource space may contain empty points (an empty point specifies no resources). This forms negative impact on the efficiency of resource management. We can further normalize the resource space by ruling out the empty points. The following definition is an enhancement of Definition 5.

Definition 14. Let $X = (C_1, C_2, \ldots, C_n)$ be an axis and C'_i be a coordinate at another axis X'. We can say that X regularly fine classifies C'_i (denoted as C'_i / X) if and only if

(1) $R(C_1) \cap R(C'_i) \neq \phi$, $R(C_2) \cap R(C'_i) \neq \phi$, ..., and $R(C_n) \cap R(C'_i) \neq \phi$;
(2) $(R(C_k) \cap R(C'_i)) \cap (R(C_p) \cap R(C'_i)) = \phi$ ($k \neq p$, and $k, p \in [1, n]$); 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.

Similar to Definitions 6 and 7, we can define the regular fine classification between axes and the regular orthogonality between axes based on Definition 14. Thereafter we can give the following definition.

Definition 15. The fourth normal form of a resource space is a third normal form and any two of its axes are regularly orthogonal with each other.

4.2. Reference resource space

A generic *reference resource space* is a three-dimensional resource space: $RS = (category, level, location)$. The category dimension is a vertical classification of resources. A coordinate of the axis represents a certain type of category. A category coordinate can include several sub-categories, and each sub-category can further include several smaller sub-categories. A category together with its all-level sub-categories constitutes a category hierarchy. Coordinates of the category axis are scalable because people usually use resources across different levels. Except for the basic sub-categories, 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 up to its up-level coordinates. Name duplication can be avoided by denoting a sub-category as *up-level-category.sub-category*. A resource can be accurately located when its category and level information are given. The generic reference resource space can have different specializations according to the types of the resources that the resource space contains.

The *reference information space* is a three-dimensional resource space (*information-category*, *information-level*, *location*), where the former two dimensions identify information content, and the third dimension identifies the locations that store information. Each point in the space represents information at a certain information level of an information category and is stored at a certain location. Information category refers to the classification of the information content. Information can be leveled according to the scale of information granularity or the types of information. For example, information granularity can be defined as five scales from small to large: *data*, *paragraph*, *page*, *section*, and *book*.

The *reference service space* is a three-dimensional resource space (*service-category*, *service-level*, *location*), where the former two dimensions identify the service region, and the third dimension identifies the logical locations that store services. The coordinates of the service-category axis are the function classification of the services. The service-level axis contains four coordinates from low (close to hardware) to high (close to users): *system level*, *middleware level*, *application interface (API) level*, and *application level*. Each point in the space represents a set of services at a certain service level of a service category and is stored at a certain location.

The *reference knowledge space* is a three-dimensional resource space (*knowledge-category*, *knowledge-level*, *location*), where the knowledge-category dimension and the knowledge-level dimension of a knowledge space identify knowledge content at a certain knowledge level of a certain knowledge category, the location dimension identifies the locations that store knowledge. With reference to the knowledge levels of the axiom system, we can classify the knowledge space into four knowledge levels from low to high: *conceptual level*, *axiom level*, *rule level*, and *method level*. The conceptual level contains the basic concepts in the form of noun or noun phrases together with their explanations or definitions like dictionaries. The axiom level contains the commonsense knowledge of knowledge categories. An axiom takes the form of a natural language statement or a mathematical equation that describes the relationship between concepts. The rule level contains the basic rules and principles of knowledge categories. A rule reflects the logical relationship between axioms. It usually takes the form of production rules: *IF condition THEN conclusion*. The method level contains the problem-solving methods. A problem-solving method takes the form of the problem-solution pairs. The solution can be either a multi-step problem-solving process or simply a one-step solution. Each knowledge category can include several sub-categories, and each sub-category can further include several smaller sub-categories. A knowledge category together with its all-level sub-categories constitutes a *knowledge category hierarchy*.

5. CRITERIA FOR DESIGNING RESOURCE SPACE

The normal forms have answered the question of what is a good resource space. The purpose of setting the design criteria is to guide users to design a good resource space. Criteria 1–3 set the conditions for being a resource of the resource space. Criteria 4–6 set the conditions for the management mechanism of the resource space.

Criterion 1 (Semantic completeness criterion). Any resource of the resource space should have a complete semantics that can be described in natural languages, i.e. a resource space should not include the resources that cannot be described or have not been described.

Criterion 2 (Resource positioning criterion). Any resource of the resource space should belong to a set of resources that corresponds to a point in the resource space, i.e. any resource should be well positioned in the resource space.

Criterion 3 (Resource use criterion). The use of any resource in the resource space should obey the restriction of its usage and execution environment that is defined at its creation time.

For example, after we have got a MS Word format file from a resource space, the reading of this file should obey its use restriction, i.e. we need the corresponding software like MS Word to open the file and then to browse it.

A resource space has a logical representation layer and a physical storage layer. If the two layers do not co-exist with each other, e.g. a resource exists at one layer but is absent from another layer, then the management mechanism will fail to correctly operate it. To guarantee such a consistency, we set the following criterion.

Criterion 4 (Existence consistency criterion). Any resource in the resource space should guarantee the consistency of its existence in different levels of schemas.

Criterion 4 excludes the case that a resource exists in one schema but is absent from another schema. Resource operations should guarantee this consistency.

Criterion 5 (Pervasive residence criterion). The resources of a resource space should be allowed to reside on various hardware and software platforms.

Criterion 6 (Operable criterion). A resource space management mechanism should at least support three basic resource operations: get a resource from the resource space, put a resource into the resource space, and delete a resource in the resource space. These operations are also suitable for managing any view of the resource space.

6. APPROACH FOR DESIGNING RESOURCE SPACE

The logical level design consists of the following five steps.

(1) *Resource analysis.* Resource analysis determines the application scope, surveys the resources that need to be managed, and then specifies all the resources by using a Resource Dictionary (RD), which forms the sources of a local resource space for the application. The resources can

be described in XML, and the RD can be managed by using our newly developed ROL or any other XML query languages.

(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 achieve the uniform viewpoint is to unify the top-level resource partition. *Human*, *information*, and *natural* (*or artificial*) *object* are three key factors of human society, which also constitute 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*.
The next step is to refine each category top-down following the approach of the first step until the category is small enough to serve for domain applications.

(3) *Design two-dimensional resource spaces.* Ordinary 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 a high-dimensional resource space. The design process includes the following steps.

- *Determine axes' names*. An axis name reflects a category of the top-level partition of resources.
- *Determine the first-level coordinate names*. Each coordinate reflects a category of the super-category reflected by the axis name.
- *Determine the coordinate hierarchies*. For each first-level coordinate, determine its low-level coordinates until reaching the basic category according to the resource partition hierarchy.
- *Check independency between coordinates*. Check independency between coordinates at all coordinate levels. In case dependency happens, re-consider resource partition at this level and then adjust coordinates.
- *Check orthogonal relationship between axes*. In case orthogonality is not satisfied, re-consider the coordinate settings, and then adjust the relevant coordinates.

(4) *Join between spaces*. Check the join condition of these two-dimensional spaces to determine whether they can be joined into one uniform resource space.

(5) *Use abstraction and analogy in design process.* Making abstraction and analogy between the existing (or reference) resource spaces and the new resource space is an important way to design a proper resource space.

7. RESOURCE REPRESENTATION

The semantics of a resource can be represented by the black box—the semantics is reflected by the semantics of relevant resources, and the white box—the semantics is described by its features and functions. We can combine these two methods to achieve better effectiveness. A resource template represents the common features of a class of resources with the same type. Resources defined in a resource space need a set of templates, which can be organized into a template hierarchy, where the

low-level templates are an expansion of the high-level template. The root template takes the following form:

```
ResourceTemplate{
    Resource-name: <string>;
    Description: <abstract>;
    Related-materials: [
        Relationships: [  <SemanticLinkType1, Resource1>,
                                . . . ,
                          <SemanticLinkTypem, Resourcem>,];
        References: [  material-name1: <address1>,
                                . . . ,
                       material-namen:<addressn> ]
    Others ] }.
```

In the above template, the 'address' can be an URL, a book, or a paper. In the case of a book, the address takes the following form: (BookName: *String*; Author: *String*; Publisher: *String*; PublisherAddress: *String* or URL). In the case of a paper, the address takes the following form: (JournalName: *String*; Volume: *Number*; Issue: *Number*; PaperTitle: *String*; AuthorName: *String*; Publisher: *String*; PublisherAddress: *String* or URL). 'Related-materials' specifies the relationship between resources and the references of the resources. The relationship is a kind of semantic link that describes the semantic relationship between resources [20]. The 'Others' can be such relationships as the 'Peer-resources: <name-list>' and the 'Meta-resources: <name-list>'.

To satisfy the pervasive residence criterion, the XML (http://www.w3.org/TR/REC-xml) or the XML-based markup languages can be adopted to represent the worldwide resource space and local resource spaces. As an example, we represent a worldwide knowledge space and a local knowledge space in XML as follows, where the URSL represents the location of the resource space in the interconnection environment.

```
<WorldwideRS>
    <UserName1> URSL1: LocalRSName: Group </UserName1>
    . . .
    <UserNamen> URSLn: LocalRSName: Group </UserNamen>
</WorldwideRS>
<LocalRSName>
  <Public>
    <Method>
      <CategoryName> MethodSet </CategoryName>
        . . .
      <CategoryName> MethodSet </CategoryName>
    </Method>
    <Rule>
      <CategoryName> RuleSet </CategoryName>
        . . .
      <CategoryName> RuleSet </CategoryName>
```



```
        </Rule>
        <Axiom>
            <CategoryName> PrincipleSet </CategoryName>
                ...
            <CategoryName>PrincipleSet </CategoryName>
        </Axiom>
        <Concept>
            <CategoryName> ConceptSet </CategoryName>
                ...
            <CategoryName> ConceptSet </CategoryName>
        </Concept>
    </Public>
</LocalRSName>
```

8. RUM

The Resource Space Grid has two types of user: the end-user who hopes to use resources directly by operating an easy-to-use interface or indirectly through an application system, and the application developer who is responsible for building the complex application systems for the end-user with the support of the resource-utilize mechanism.

8.1. ROL

The ROL defines a set of basic operation statements for creating resource spaces and for sharing and managing resources. These statements enable the user to create a local resource space, to get the required resources from the universal view of all the local resource spaces, to put a set of resources into a local resource space, to delete the resources with the right privilege, to browse the required resources, to log the local resource space on or off the global view, to open a local resource space to a set of particular users, and to join several resource spaces into one resource space. The ROL includes the following statements, where '[...]' means by default.

(1) Create-statement. Grammar: CREATE $RS(X_1, X_2, \ldots, X_n)$ [AT *URSL*] WHERE $X_1 = \{C_{11}, \ldots, C_{1n}\}, \ldots, X_n = \{C_{n1}, \ldots, C_{nm}\}$. The create statement is to create a resource space with the axes and coordinates defined in the 'where' clause.
(2) Get-statement. Grammar: GET *R* FROM *RS* [AT *URSL*] WHERE $X_1 = C_{1i}, \ldots, X_n = C_{nj}$. The get-statement is to get the required resources from the specified point of the given resource space.
(3) Put-statement. Grammar: PUT *R* INTO *RS* [AT *URSL*] WHERE $X_1 = C_{1i}, \ldots, X_n = C_{nj}$. The put statement is to put resource *R* into the specified point of the given resource space.
(4) Delete-statement. Grammar: DELETE *R* IN *RS* [AT *URSL*] WHERE $X_1 = C_{1i}, \ldots, X_n = C_{nj}$. This statement means that if the specified resource exists at the specified point of the given resource space and the user has the operation privilege, then it can be deleted.

(5) Browse-statement. Grammar: BROWSE *RS* [AT *URSL*] [WHERE <*Condition*>]. The <*Condition*> can be either a subspace $(X_i, \ldots, X_j)$ or a set of points $(X_1 = \{C_{1\alpha}, \ldots, C_{1\beta}\}, \ldots, X_n = \{C_{n\mu}, \ldots, C_{nv}\})$. The browse statement is to browse the resources in the given resource space or the resources at the specified point.
(6) Log-statement. Grammar: LOG *RS* [AT *URSL*] ON/OFF [Community]. The log statement is to log a local resource space on or off the given community or the global resource space.
(7) Open-statement. Grammar: OPEN *RS* [AT *URSL*] TO $RS_1, \ldots, RS_m$ [AT $URSL_1, URSL_2, \ldots, URSL_m$]. The local resource spaces work in peer-to-peer mode. The open statement is to open a resource space to the specified resource spaces to make them accessible to its resources.
(8) *Join*-statement. Grammar: JOIN $RS_1, \ldots, RS_m$ [AT $URSL_1, URSL_2, \ldots, URSL_m$] INTO *RS* COMMON AXIS $(Axis_1, \ldots, Axis_n)$.
(9) *Disjoin*-statement. Grammar: DISJOIN *RS* [AT *URSL*] INTO $RS_1(X_{11}, \ldots, X_{1n}), \ldots, RS_m(X_{m1}, \ldots, X_{mn})$ [AT $URSL_1, URSL_2, \ldots, URSL_m$] COMMON AXIS $(Axis_1, \ldots, Axis_k)$. The disjoin-statement is to disjoin the *RS* at *URSL* into $RS_1, \ldots,$ and RS_m at $URSL_1, URSL_2, \ldots,$ and $URSL_m$, respectively, and satisfies certain conditions. The constraint between the axes of *RS* and RS_i are defined in the condition portion. The locations of the resource spaces are useful in deploying resource spaces.
(10) *Merge*-statement. Grammar: MERGE $RS_1, \ldots, RS_n$ [AT $URSL_1, URSL_2, \ldots, URSL_n$] INTO *RS* [AT *URSL*] WHERE *new-axis*(*RS*) $= X_{1\mu}(RS_1)$ & $\cdots$ & $X_{nv}(RS_n)$ COMMON AXIS $(Axis_1, \ldots, Axis_{n-1})$. The merge statement is to merge the resource spaces $RS_1, \ldots,$ and RS_n at $URSL_1, URSL_2, \ldots,$ and $URSL_n$, respectively, into a new *RS* and to place it at *URSL* under the condition of merge.
(11) *Split*-statement. Grammar: SPLIT *RS* [AT *URSL*] INTO $RS_1, \ldots, RS_n$ [AT $URSL_1, URSL_2, \ldots, URSL_n$] WHERE $X(RS) => X_{1\alpha}(RS_1) =$ <*CoordinateSet*$_1$> & $\cdots$ & $X_{n\beta}(RS_n) =$ <*CoordinateSet*$_n$>. The split statement is to split a resource space *RS* at the *URSL* into $RS_1, \ldots,$ and RS_n at $URSL_1, URSL_2, \ldots,$ and $URSL_n$, respectively. The axis X of *RS* will be split into $X_{1\alpha}(RS_1), \ldots,$ and $X_{n\beta}(RS_n)$.

The above operation statements can operate multiple resources by extending the expression of the resource and its location to a list of resources and a list of corresponding locations. The inexact resource retrieval can be realized by adding some particular equations to the condition portion.

8.2. Operable resource browser

An operable resource browser is a user-friendly operation interface that is responsible for locating the resources according to the user's requirement then informs the user of the content of these resources represented in a template form. A resource browser consists of the following main functions.

(1) Provide an easy-to-use interface for users to input their operation requirements and to carry out resource operations.
(2) Provide either a local resource view or a universal resource view of all the resources.
(3) Check the grammar of the resource operation statements.
(4) Deliver the operation to the resource-utilize engine, and then get the result from the engine.
(5) Show the operation result.

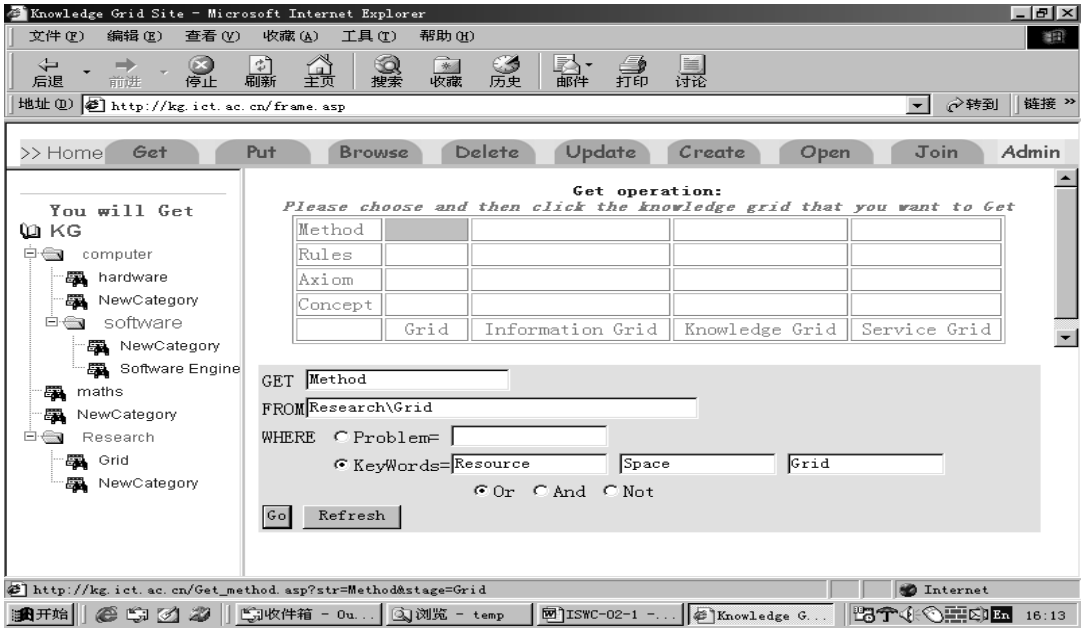


Figure 2. The browser interface for getting a method from a knowledge space.

Figure 2 shows an interface of the resource browser where the upper portion shows the operation buttons, the middle portion is the user-view schema of a resource space, the lower portion is the pattern of the operation that has been selected, and the left portion of the interface is a scalable coordinate hierarchy of the category axis. A cooperative team can unify the coordinates by editing the coordinate hierarchy. For example, a software development team can set the following five coordinates at the category axis: *analysis*, *design*, *programming*, *testing*, and *maintaining*, according to the stages of the structured software development methodology. They can also add such coordinates as 'team-affair' after the creation of the coordinates.

The browser supports the following resource operation process.

(1) Choose the operation to be operated by clicking an operation button.
(2) Choose the resources to be operated by determining the coordinates in a resource space (i.e. clicking the left resource tree then clicking the right rectangle representing a point of the resource space).
(3) Input the operation parameters to define the operation.

8.3. Cooperation between resource spaces and resource usage engine

The cooperation relationship between the service resources, the knowledge resources, and the information resources is described in Figure 3. Information is the origin of knowledge generation,

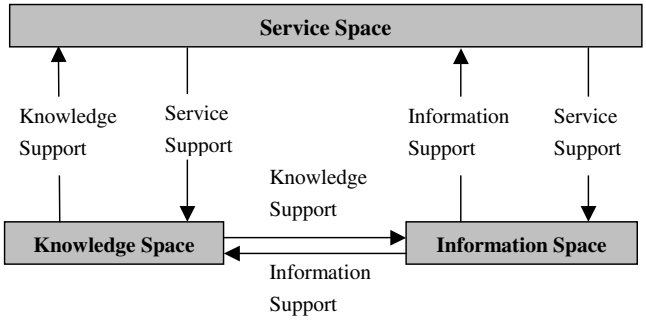


Figure 3. Cooperation between information space, knowledge space and service space.

and knowledge in turn supports information understanding. The information space provides both the knowledge space and the service space with information support. The knowledge resources provide both the service resources and the information resources with knowledge support. The service resources can get information from the information space or get knowledge from the knowledge space for processing information (like retrieving, computing or filtering) or processing knowledge (like extraction, refinement or reasoning), then put the processing outputs (knowledge or information) into the information space or the knowledge space.

The resource-utilize engine is responsible for enacting the resource operations determined by the browser and explaining the content of resources according to their type because different types of resources require different resource-utilize ways. The implementation concerns the following three issues.

(1) Guarantee a proper granularity of resource during operation. For example, rules should be used as a component because a single rule may not be meaningful and the deletion of a single rule may cause incompleteness of the component it belongs to.
(2) Explain the content of resources (e.g. convert the formal rules to informal) because the end-user may not be able to understand the formal expression of resources.
(3) Obtain the support from the other types of resources when using resources.

The resource-utilize engine can include the following advanced functions to realize in-depth resource utilization.

(1) Reasoning and explanation according to the knowledge in the resource space when seeking the answers to users' queries.
(2) Automatically acquire knowledge resources from information resources.
(3) Refine raw resources into fine resources, and eliminate inconsistency and redundancy.

8.4. Complex application: resource operation programming

The ROL statements can also be composed as a program to implement an application. Similar to any structural programming language, the control of the ROL is realized by the following statements.

(1) Sequential-process: ROL-Statement; ROL-Statement; . . . ; ROL-Statement.
(2) Begin–End-statement: BEGIN sequential-process END.
(3) Branch-statement: IF <Condition-Expression> THEN <ROL-Statement> ELSE <ROL-Statement>.
(4) Loop-statement: WHILE <Condition-Expression> DO <ROL-Statement>.
(5) Assignment: <Variable>=<ResourceId> | <Statement> | <Math-Expression> | <Boolean-Expression>.

We have implemented a prototype of the programming environment in the China VEGA-KG project.

9. IMPLEMENTATION

A prototype of the Resource Space Grid VEGA-KG has been implemented to support distributed resource sharing and management (http://kg.ict.ac.cn). It includes three types of Grids: a Knowledge Grid, an Information Grid and a Service Grid, which manage knowledge, information and services respectively in knowledge space, information space and service space. Any user can login as either a team member or a personal user to create a new local resource space or to operate resources in the existing resource spaces by clicking the operation button on the interface as shown in Figure 2 then selecting the suitable resources to be operated through clicking the rectangle representing the point of the resource space.

The VEGA-KG also supports flexible retrieval of resources. Users can express their inexact queries by specifying the following six kinds of inexact specialization relationships between resources: identical-specialization, extension-specialization, partial-specialization, revision-specialization, more-general-than and more-special-than. Figure 4 shows the browser interface for getting a service from a service space according to the inexact specialization relationship between services.

The VEGA-KG also includes a Knowledge Collection Board (KCB) that realizes knowledge collection through human–machine cooperation. The collected knowledge can be put into the knowledge space for public sharing after carrying out knowledge refinement. The KCB is available at http://kg.ict.ac.cn.

10. COMPARISONS

10.1. Comparison between RSM and RDBM

There are two common points between the proposed RSM and the RDBM [21,22]. The first is that operations are separated from the objects to be managed. The second is that their operation languages have similar forms—SQL-like [17]. This enables users to easily understand the syntax and semantics of the ROL. The major different points between the RSM and the RDBM concern six aspects, as shown in Table I.

The first difference is that the foundation of the RSM is the resource ontology, while the foundation of the RDBM is the relational algebra. The second difference is that the managed objects of the RSM are the structured or semi-structured information, knowledge, and resources, while the managed objects

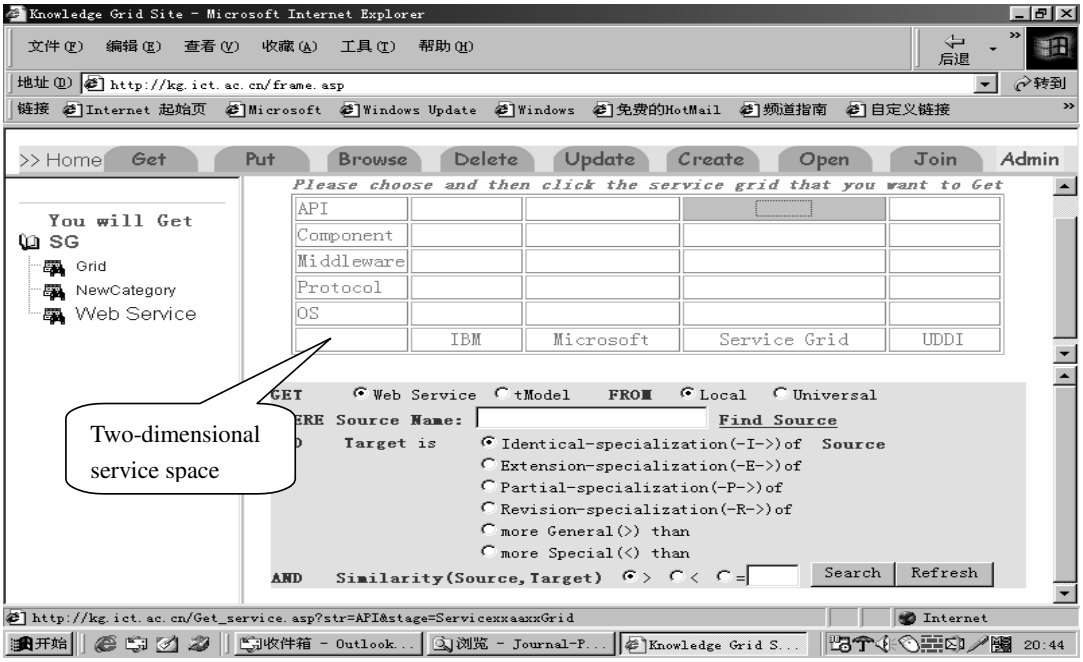


Figure 4. The browser interface for getting a service from a resource space.

Table I. Major differences between RSM and RDBM.

No.	Items	RSM	RDBM
1	Foundation	Resource ontology	Relational algebra
2	Managed objects	Structured/semi-structured resources	Atomic data
3	Data model	Uniform coordinate system	Relational table
4	Normalization basis	Independent coordinate and orthogonal axes	Function dependence
5	Operation feature	Uniform semantic view	Attribute view
6	Interchange basis	Semantic Web	ODBC (RDBMS)

of the RDBM are the atomic data. The third difference is that the data model of the RSM is a uniform coordinate system, while the data model of the RDBM is the relational table. The fourth difference is that the normalization basis of the RSM is the independent and orthogonal coordinate system, while the normalization basis of the RDBM is the function dependence relationship. The above differences determine that the RSM concerns the contents (semantics) of the managed objects (resources) and the content-based classification so it can carry out content-based operation and can accurately locate resources, but the RDBM concerns the attributes of the managed objects so it supports attribute-based operation. The fifth difference is that the RSM supports a uniform classification-based semantic view when operating resources, while the RDBM essentially supports the view of attributes. This feature enables the RSM to be suitable for uniformly sharing and managing the Internet resources. The sixth difference is that the interchange basis of the RSM is the Semantic Web, which provides the machine with an understandable semantic basis for resources, while the RDBM does not consider the interchange requirement. The ODBC is used as the interchange standard between different commercial RDBMS.

10.2. Comparison between RSM, multi-dimensional data model and object relation model

The differences between the OLTP and the OLAP were compared in [23]. The multidimensional data model is used for the data warehouse and OLAP. It is different from the RSM in the foundation, the managed objects, the normalization basis, the operation feature, and the interchange basis aspects.

The OO (object-oriented) methodology [24] provides the uniform method and mechanism for domain modeling and system implementation. It simplifies the complexity of objective systems by using the notions of class and object to abstract versatile entities, by encapsulating operations into each class, and by enabling the inheritance mechanism, etc. It realizes software reuse during the software development process. Object-relational databases (ORDBs) are a combination of object-orientation technology and the relational database technology [25]. In an ORDB, a table may not be in the first normal form of the relational data model, and a table can be nested in another table. Various nested normal forms are studied as an extension of traditional flat normal forms in the relational data model area [26–28], but these extensions are based on the relational algebra and relational data model. Similar to the comparison in Table I, the RSM is different from the ORDB in foundation, managed object, data model, normalization basis, operation feature, and exchange basis.

10.3. Comparison between the operable resource browser and the current Web browser

The operable resource browser enables users to actively operate resources. Table II shows five major differences between the resource browser and the current Web browser:

(1) the objects operated by the resource browser are versatile Internet resources, while the current Web browser only operates Web pages;
(2) the resource browser can accurately locate resources by coordinates—a kind of classification semantics, while the current Web browser locates Web pages by URL;
(3) the resource browser can accurately locate resources then operate them by setting proper parameters, but the current Web browser can only search Web pages according to keywords;

Table II. Comparison between the resource browser and the current Web browser.

No.	Items	Resource browser	Current Web browser
1	Operated object	Internet resources	HTML Web pages
2	Resource locating	Coordinate-based resource locating	URL + hyperlink
3	Operation	Coordinate-based resource locating + operation language	Keyword-based retrieval
4	View	Uniform resource semantic view	One HTML page
5	Support model	RSM	Without data model

Table III. 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 table operations
5	Reference model	Yes	No
6	Data organization	Hierarchy by top-down partitioning	Flat table
7	Schemas design	User-view level, logical level, Semantic Web level	Conceptual level, logical level, physical level

(4) the resource browser supports a uniform resource semantic view, while the current Web browser only supports a view of one Web page; and,
(5) the resource browser is supported by the RSM, while the current Web browser lacks the support from a complete data model.

10.4. Comparison between the design methods for RSM and for RDBM

The major differences between the design method for the relational database [22] and the design method for the RSM concern seven items, as compared in Table III. The design method for the RSM does not have a conceptual model, so experience and the reference model play a key role in 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 mode. The RSM is established on the Semantic Web level, so it does not have the physical level schema of the RDBM.

10.5. Comparison between ROL, LOREL, XSL and XQL

The ROL is not only an SQL-like language but also a programming environment based on a semi-structured data model combined with the XML syntax. According to the comparisons between five

Table IV. Comparison of ROL, XQL, LOREL and SQL.

Feature items	ROL	XQL	LOREL	SQL
Data format	XML Doc	XML Doc	XML Doc	Table
Operating object*	Resources (Information, Knowledge, Services, and Grid)	XML Doc	XML Doc	Table
Operation result	XML Doc	XML Doc	Set of OIDs	Table
Abstract data types	Yes	No	Yes	No
SQL-like	Yes	No	Yes	Yes
Specific data model	Yes	No	Yes	Yes
Different management of IDREFs	No	No	Yes	No
Document selection	Yes	Yes	Yes	No
Partial specified path expression	Yes	Yes	Yes	No
Nested queries*	Yes	No	Yes	Yes
Update language*	Yes	No	Yes	Yes
Loop statement*	Yes	No	No	No
Branch statement*	Yes	No	No	No
Set operations	Partially	Yes	Yes	Yes
Aggregates	Partially	Partially	Yes	Yes
Skelom function	Yes	No	Yes	No
Join operation*	Yes	No	Yes	Yes
Open operation*	Yes	No	No	No
Support of view*	Yes	No	No	Yes
Create new elements	Yes	No	Yes	Yes
Query structure	Yes	No	No	No
Query content	Yes	Yes	Yes	Yes
Support of RDF	No	No	No	No
Support of typed link*	Yes	No	No	No
Type coercion	No	Partially	Yes	No
Ordering the result	Yes	No	Yes	Yes
Universal resource view*	Yes	No	No	No

* Distinguishing features of the ROL.

types of XML query languages: LOREL, XML-QL, XML-GL, XSL and XQL made in Bonifati and Ceri [29], we compare the ROL with the XQL, the LOREL and the SQL in Table IV, where the distinguishing features of the ROL are denoted by an asterisk.

The XQL is a concise language and is developed as an extension of the XSL pattern language. It builds upon the capability of identifying classes of nodes by adding Boolean logic, filters, indexing into the collections of nodes [29]. The LOREL (http://www-db.stanford.edu/lore) is a friendly language in the SQL style. As Table IV shows, the ROL borrows the syntax and semantics from the standard SQL language. The statements of the ROL are SQL-like and have the SQL SELECT-FROM-WHERE pattern. The ROL can perform operations similar to the classical relational database operations, such as nested queries, aggregates, set operations, join and result ordering.

The ROL also borrows the following features from the XML query languages:

(1) the management of structured and semi-structured data;
(2) support abstract data types;
(3) the XML-based data format and the result semantics;
(4) support the skelom functions to associate a unique ID to a given Resource Space Grid;
(5) support document selection; and
(6) support partially specified path expression.

A distinguishing feature of the ROL is that it can operate various types of resources with hierarchical structure including information, knowledge, services and even Resource Space Grids with a universal semantic view supported by the RSM. But, the standard SQL only operates flat relational tables, and the XML query languages only manage XML documents.

In order to raise the effectiveness and to enhance the mobility of using resources, a set of typed links is established between resources in Resource Space Grid, like the abstraction and similar-to relationships. The function of the ID references (IDREFs) is interpreted as references between elements, not as strings in the ROL. The current version of the ROL does not support the Resource Definition Framework (RDF), the group-by clause, and the type coercion.

11. SUMMARY

11.1. Concluding remarks

As an effort towards the next-generation Web, we have proposed the model, method and platform for the Resource Space Grid. The kernel of the Resource Space Grid model includes the RSM and the RUM. The RSM organizes the versatile resources in semantic normal forms, which guarantee the correctness and the efficiency of resource operations. The Resource Space Grid model gives the Internet users a means to represent, store, accumulate, share, manage, and use versatile resources in a globally distributed application environment. The proposed criteria and design method can guide designers to design a proper resource space. The prototype platform based on the proposed model has been implemented and used in research teams and software development teams. Applications in geographically distributed research teams show that the platform can realize resource sharing between team members. The effectiveness and efficiency of teamwork can be raised due to the effect of resource sharing. Comparisons show that the kernel technologies used in the proposed approach (including the RSM model, the design method of the resource space, the operable browser and the ROL) are eligible for uniformly sharing and managing versatile resources in a future interconnection environment.

11.2. Discussion and ongoing work

The exponential growth of resources is an obstacle to realize effective management. A resource space with a fixed number of dimensions cannot avoid the exponential expansion of the resources in each point of resource space. This issue can be solved in theory by increasing the number of dimensions. Assuming that resources expand at the rate of e^n, we can use a resource space of n dimensions where each dimension has n coordinates to manage the resource explosion since $n^n > e^n$ when $n > e$.

In practice, resources are usually first classified by communities, and then classified by resource spaces on different topics. More importantly, the resources in a future interconnection environment should have life spans, which can avoid unlimited expansion of resources. A unified resource model named soft-device was proposed for modeling resources in future interconnection environments [19].

The orthogonal resource space only solves the normal organization issue. The ideal resource management approach still needs the mobility. For example, the ideal approach should still be able to accurately locate a resource even though it is put into the wrong place in the resource space. The Semantic Link Network model is proposed to establish the semantic links between resources [20]. The combination of the orthogonal resource space model and the Semantic Link Network model can establish a kind of *single semantic image* for versatile resources, which enables a resource to be efficiently retrieved no matter where it is placed in the space. Hence, an ideal resource space should be defined on the basis of the classification semantics and link semantics.

Ongoing work includes the following aspects.

(1) *Extension of RSM*. We are seeking the normal form between the second and the third normal forms to support a useful but not strict resource space. We are also seeking the higher normal forms.
(2) *Process Grid—enabling process management*. We are establishing knowledge management [30,31] functions above the Resource Space Grid for the purpose of raising the work efficiency of a cooperative team, inspiring innovation, and enabling the whole knowledge of a team to be accumulated and evolved during the cooperation process. A knowledge flow model has been proposed to realize effective knowledge sharing [15]. We are going to incorporate workflow management functions into the process management mechanism and to realize the coordination between the knowledge flow and the workflow in modeling business processes [15,32].
(3) *Resource coupling*—to give Internet users the intelligent and in-depth resource-utilize services through coupling different types of resources. As a special case of resource coupling, we have established the typed semantic links between text resources and realized the corresponding text browse mechanism.
(4) *Resource component and resource community*. Examining the case of people making use of the rules in a rule base, we find that knowledge usage activities are carried out at a certain granularity level, because the use of any single rule sometimes does not make any sense. Similarly, the resources are shared at a certain granularity level. A Grid can also be regarded as a special resource. A worldwide Grid can include a set of Grid components classified by geographical regions or specialization fields. A resource community is a kind of large granularity unit where resources are connected under the same or a similar topic.
(5) *Grid semantics and methodology*. We are investigating the semantic issues of the Resource Space Grid itself and establishing a new system modeling methodology that is suitable for establishing a future interconnect environment.

APPENDIX A. THE PROOF OF THE TRANSITIVITY OF FINE CLASSIFICATION

Let $X_1 = \{C_{11}, C_{12}, \ldots, C_{1k}\}$, $X_2 = \{C_{21}, C_{22}, \ldots, C_{2m}\}$ and $X_3 = \{C_{31}, C_{32}, \ldots, C_{3n}\}$ be three axes satisfying X_2/X_1 and X_3/X_2. We prove X_3/X_1 by the following two points.

(1) For $\forall C_{3p} \in X_3$, $\forall C_{1i} \in X_1$ and $\forall C_{1j} \in X_1$ $(C_{1i} \neq C_{1j})$, we prove $(R(C_{1i}) \cap R(C_{3p})) \cap (R(C_{1j}) \cap R(C_{3p})) = \varnothing$ as follows.

$(R(C_{1i}) \cap R(C_{3p})) \cap (R(C_{1j}) \cap R(C_{3p})) = R(C_{3p}) \cap (R(C_{1i}) \cap R(C_{1j}))$

$\because X_3 / X_2$

$\therefore C_{3p} / X_2$

$\therefore R(C_{3p}) = \bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{3p})) = R(C_{3p}) \cap \bigcup_{q=1}^{m} R(C_{2q})$

$\therefore R(C_{3p}) \cap (R(C_{1i}) \cap R(C_{1j}))$

$= \left(R(C_{3p}) \cap \bigcup_{q=1}^{m} R(C_{2q}) \right) \cap (R(C_{1i}) \cap R(C_{1j}))$

$= R(C_{3p}) \cap \bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{1i}) \cap R(C_{1j}))$

$= R(C_{3p}) \cap \bigcup_{q=1}^{m} ((R(C_{1i}) \cap R(C_{2q})) \cap (R(C_{1j}) \cap R(C_{2q})))$

$\because X_2 / X_1$

$\therefore$ for $\forall C_{2q} \in X_2$, C_{2q} / X_1

$\therefore (R(C_{1i}) \cap R(C_{2q})) \cap (R(C_{1j}) \cap R(C_{2q})) = \varnothing$

$\therefore R(C_{3p}) \cap \bigcup_{q=1}^{m} ((R(C_{1i}) \cap R(C_{2q})) \cap (R(C_{1j}) \cap R(C_{2q})))$

$= R(C_{3p}) \cap \bigcup_{q=1}^{m} \varnothing$

$= \varnothing$

$\therefore (R(C_{1i}) \cap R(C_{3p})) \cap (R(C_{1j}) \cap R(C_{3p})) = \varnothing.$

(2) For $\forall C_{3p} \in X_3$, we can prove $R(C_{3p}) = \bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{3p}))$

$\bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{3p})) = R(C_{3p}) \cap \bigcup_{i=1}^{k} R(C_{1i})$

$\because X_3 / X_2$

$\therefore R(C_{3p}) = \bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{3p}))$

$$\therefore R(C_{3p}) \cap \bigcup_{i=1}^{k} R(C_{1i}) = \left(\bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{3p})) \right) \cap \bigcup_{i=1}^{k} R(C_{1i})$$

$$= R(C_{3p}) \cap \left(\bigcup_{q=1}^{m} R(C_{2q}) \right) \cap \bigcup_{i=1}^{k} R(C_{1i})$$

$$= R(C_{3p}) \cap \left(\bigcup_{q=1}^{m} \bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{2q})) \right)$$

$\because X_2/X_1$

$$\therefore R(C_{2q}) = \bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{2q}))$$

$$\therefore R(C_{3p}) \cap \left(\bigcup_{q=1}^{m} \bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{2q})) \right) = R(C_{3p}) \cap \bigcup_{q=1}^{m} R(C_{2q})$$

$$= \bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{3p}))$$

$\because X_3/X_2$

$$\therefore R(C_{3p}) = \bigcup_{q=1}^{m} (R(C_{2q}) \cap R(C_{3p}))$$

$$\therefore R(C_{3p}) = \bigcup_{i=1}^{k} (R(C_{1i}) \cap R(C_{3p}))$$

According to (1) and (2), we have X_3/X_1.

APPENDIX B. THE PROOF OF THE TRANSITIVITY OF ORTHOGONALITY

Let $X_1 = \{C_{11}, C_{12}, \ldots, C_{1k}\}$, $X_2 = \{C_{21}, C_{22}, \ldots, C_{2m}\}$ and $X_3 = \{C_{31}, C_{32}, \ldots, C_{3n}\}$ be three axes satisfying $X_2 \perp X_1$ and $X_3 \perp X_2$. We can prove $X_3 \perp X_1$ as follows.

$\because X_3 \perp X_2$ and $X_2 \perp X_1$

$\therefore X_3/X_2$ and X_2/X_1

We have X_3/X_1 according to the transitivity of fine classification.

$\because X_2 \perp X_1$ and $X_3 \perp X_2$

$\therefore X_1 \perp X_2$ and $X_2 \perp X_3$

$\therefore X_1/X_2$ and X_2/X_3

We have X_1/X_3 according to the transitivity of fine classification. Hence we confirm $X_3 \perp X_1$.

APPENDIX C. RESOURCES IN A THIRD NORMAL FORM RESOURCE SPACE CAN BE ACCESSED FROM ANY AXIS

Let $X_i = \{C_{i1}, C_{i2}, \ldots, C_{ip}\}$, $1 \leqslant i \leqslant n$. If the *RS* satisfies the third normal form, then for any two axes $X_k = \{C_{k1}, C_{k2}, \ldots, C_{kl}\}$ and $X_j = \{C_{j1}, C_{j2}, \ldots, C_{jm}\}$ $(1 \leqslant k \neq j \leqslant n)$, $X_k \perp X_j$. We have C_{kq}/X_j for every C_{kq} $(1 \leqslant q \leqslant l)$.

According to the definition of the fine classification, we have

$$R(C_{kq}) = (R(C_{kq}) \cap R(C_{j1})) \cup (R(C_{kq}) \cap R(C_{j2})) \cup \cdots (R(C_{kq}) \cap R(C_{jm}))$$

$$= R(C_{kq}) \cap (R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm}))$$

then $R(C_{kq}) \subseteq (R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm}))$ hold $(1 \leqslant q \leqslant l)$. Hence, we have $(R(C_{k1}) \cup R(C_{k2}) \cup \cdots \cup R(C_{kl})) \subseteq (R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm}))$ $(1 \leqslant k \neq j \leqslant n)$. Similarly, we have $(R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm})) \subseteq (R(C_{k1}) \cup R(C_{k2}) \cup \cdots \cup R(C_{kl}))$ $(1 \leqslant k \neq j \leqslant n)$. And then we have $(R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm})) = (R(C_{k1}) \cup R(C_{k2}) \cup \cdots \cup R(C_{kl}))$ $(1 \leqslant k \neq j \leqslant n)$. So, if we suppose $R = (R(C_{j1}) \cup R(C_{j2}) \cup \cdots \cup R(C_{jm}))$, then for every axis $X_i = \{C_{i1}, C_{i2}, \ldots, C_{ip}\}$, $1 \leqslant i \leqslant n$, $(R(C_{i1}) \cup R(C_{i2}) \cup \cdots \cup R(C_{ip})) = R$. This means that the resources that are accessible from any axis are the same. Hence the resource retrieval algorithm does not need to consider the order of the axes.

ACKNOWLEDGEMENTS

The author thanks all the team members of the China Knowledge Grid Research Group (CKG), especially J. Liu, Y. Li, E. Yao, and Y. Xing for their diligent work on developing relevant software and helpful suggestions. Thanks also go to the editor and reviewers who provided helpful comments. The research work was supported by the National Science Foundation and the National Basic Research Plan of China.

REFERENCES

1. Barabási A-L, Bonabeau E. Scale-free networks. *Scientific American* 2003; **288**(5):60–69.
2. Berners-Lee T, Hendler J, Lassila O. Semantic Web. *Scientific American* 2001; **284**(5):34–43.
3. Heflin J, Hendler J. A portrait of the Semantic Web in action. *IEEE Intelligent Systems* 2001; **16**(2):54–59.
4. Hendler J. Agents and the Semantic Web. *IEEE Intelligent Systems* 2001; **16**(2):30–37.
5. Maedche A, Staab S. An ontology learning for the Semantic Web. *IEEE Intelligent Systems* 2001; **16**(2):72–79.
6. McHraith SA, Son TC, Zeng H. Semantic Web Services. *IEEE Intelligent Systems* 2001; **16**(2):46–53.
7. Decker S, Melnik S, Harmelen F, Fensel D, Klein M, Broekstra J, Erdmann M, Horrocks I. The Semantic Web: The roles of XML and RDF. *IEEE Internet Computing* 2000; **4**(5):63–74.
8. Klein M. XML, RDF, and relatives. *IEEE Intelligent Systems* 2001; **16**(2):26–28.
9. Broekstra J, Klein M, Decker S, Fensel D, Harmelen F, Horrocks I. Enabling knowledge representation on the Web by extending RDF schema. *Computer Networks* 2002; **39**(5):609–634.
10. Fensel D, Harmelen F, Horrocks I, McGuinness D, Patel-Schneider P. OIL: An ontology infrastructure for the Semantic Web. *IEEE Intelligent Systems* 2001; **16**(2):38–45.
11. Hendler J, McGuinness D. The DARPA Agent Markup Language. *IEEE Intelligent Systems* 2000; **15**(6):72–73.
12. Martin P, Eklund PW. Knowledge retrieval and the World Wide Web. *IEEE Intelligent Systems* 2000; **15**(3):18–25.
13. Foster I. Internet computing and the emerging Grid. http://www.nature.com/nature/webmatters/grid/grid.html [7 December 2000].

14. Frey J, Tannenbaum T, Livny M, Foster I, Tuecke S. Condor-G: A computation management agent for multi-institutional Grids. *Proceedings 10th IEEE Symposium on High Performance Distributed Computing (HPDC10)*, 2001. IEEE Computer Society Press: Los Alamitos, CA, 2001.
15. Zhuge H. A knowledge flow model for peer-to-peer team knowledge sharing and management. *Expert Systems with Applications* 2002; **23**(1):313–320.
16. Zhuge H. China e-science knowledge Grid environment. *IEEE Intelligent Systems* 2004; **19**(1):13–17.
17. ANSI. The database language SQL. *Document ANSI X3.315*, 1986.
18. Eisenberg A, Melton J. Sql:1999, formerly known as Sql3. *SIGMOD Record* 1999; **28**(1):131–138.
19. Zhuge H. Clustering soft-devices in Semantic Grid. *IEEE Computing in Science and Engineering* 2002; **4**(6):60–62.
20. Zhuge H. Active document framework: Model and tool. *Information and Management* 2003; **41**(1):87–97.
21. Bocy R, Chamberlin D, King W, Hammer M. Specifying queries as relational expressions. *Communications of the ACM* 1975; **18**(11):621–628.
22. Codd EF. A relational model of data for large shared data banks. *Communications of the ACM* 1970; **13**(6):377–387.
23. Han J, Kambr M. *Data Mining: Concepts and Techniques*. Morgan Kaufmann: San Francisco, CA, 2000.
24. Booch G, Rumbaugh J, Jacobson I. *The Unified Modeling Language: User Guide*. Addison-Wesley: Reading, MA, 1999.
25. Stonebraker M, Brown P, Moore D. *Object-Relational DBMSs: Tracking the Next Great Wave* (2nd edn). Morgan Kaufmann: San Francisco, CA, 1999.
26. Mok WY. A comparative study of various nested normal forms. *IEEE Transactions on Knowledge and Data Engineering* 2002; **14**(2):369–385.
27. Ozsoyoglu ZM, Yuan LY. A new normal form for nested relations. *ACM Transactions on Database Systems* 1987; **12**(1):111–136.
28. Tari Z, Stokes J, Spaccapietra S. Object normal forms and dependency constraints for object-oriented schemata. *ACM Transactions on Database Systems* 1997; **22**(4):513–569.
29. Bonifati A, Ceri S. Comparative analysis of five XML query languages. *SIGMOD Record* 2000; **29**(1):68–79.
30. Dieng R. Knowledge management and the Internet. *IEEE Intelligent Systems* 2000; **15**(3):14–17.
31. Drucker PF (ed.). *Harvard Business Review on Knowledge Management*. Harvard Business School Press: Boston, MA, 1998.
32. WfMC. The workflow reference model. http://www.wfmc.org [October 2000].

