Grid Resource Discovery Based on Semantic

YUEFENG FANG[1], KUN GAO[1], XIAOYONG WANG[2]
[1] Culture and Communication College
Zhejiang Wanli University
No. 8, South Qian Hu Road, 315100, Ningbo, Zhejiang
P. R. China
http://www.zwu.edu.cn
[2] Department of Computer and information
Zhejiang Wanli University
No. 8, South Qian Hu Road, 315100, Ningbo, Zhejiang
P. R. China
http://www.zwu.edu.cn

Abstract: An important aspect of Grid computing is publication resource service and discovery resource service. Grid computing is based on this kind of mechanism. A Grid service is an extended Web service that conforms to the Open Grid Service Infrastructure (OGSI) specification. The shortcoming of current Web service technology only provides syntactic description. In this paper, we analyze the limitations for current Web Service standards and point out that semantic description is the basis for automatic service discovery. We propose a novel semantics based Grid Service framework which support publishing and discovery of Grid Service very well. The experimental result demonstrates that the framework has good performance.

Key-Words: Grid Services, Web Service, Semantic Web, Grid Computation

1 Introduction

Grid is an emerging technology for enabling resource sharing and coordinated problem solving in dynamic multi-institutional virtual organizations. In the Grid environment, shared resources and users typically span different organizations. The resource publication and discovery problem in this environment involves assigning resources to tasks in order to satisfy task requirements and resource policies. These requirements and policies are often expressed in disjoint application and resource models, forcing a resource selector to perform semantic matching between the two.

Grids are used to join various geographically distributed computational and data resources, and deliver these resources to heterogeneous user communities. These resources may belong to different institutions, have different usage policies and pose different requirements on acceptable requests. Grid applications, at the same time, may have different constraints that can only be satisfied by certain types of resources with specific capabilities. Before resources can be allocated to run an application, a user or agent must select resources appropriate to the requirements of the application. In a dynamic Grid environment, where resources may come and go, it is desirable and sometimes necessary to automate the resource matching to robustly meet application requirements.

Existing resource description and resource selection in the Grid is highly constrained. Traditional resource matching, as exemplified by the Condor Matchmaker, is done based on symmetric, attribute-based matching. In these systems, the values of attributes advertised by resources are compared with those required by jobs. For the comparison to be meaningful and effective, the resource providers and consumers have to agree upon attribute names and values. The exact matching and coordination between providers and consumers make such systems inflexible and difficult to extend to new characteristics or concepts. Moreover, in a heterogeneous multi-institutional environment such as the Grid, it is difficult to enforce the syntax and semantics of resource descriptions.

In the ontology-based matchmaker, we can employ a flexible and extensible approach for performing Grid resource selection, which, unlike the traditional Grid resource selectors, uses separate ontologies to declaratively describe resources and job requests. Instead of exact syntax matching, the ontology-based matchmaker performs semantic matching using terms defined in ontologies. The loose coupling between resource and request descriptions removes the tight coordination requirement between resource providers and consumers. In addition, the matchmaker can be easily extended, by adding vocabularies and inference rules, to include new concepts about resources and applications and adapted the resource selection to changing policies.

Web service paradigm has emerged as an important mechanism for Web service discovery and publication in the dynamic environment. In typical Web service architecture, a public registry is used to store information about Web services description produced by the service providers and can be queried by the service requesters for specific application. However, the potential of a large scale growth of private and semiprivate registries is creating the need for an infrastructure which can support discovery and publication over a group of autonomous registries. Emerging PEER TO PEER solutions particularly suit for the increasingly decentralized application. They make it possible for different participants (organizations, individuals, or departments within an organization) to exchange information in a more flexible and scalable way. Recently a new generation of PEER TO PEER systems, offering distributed hash table functionality, has been proposed. These systems greatly improve the scalability and exact match accuracy of PEER TO PEER systems, but offer only the exact match query facility. Their applications are extremely limited. Semantic Web technologies have been shown to support all these missing types of functionality [1]. Thus the combination of concepts provided by Semantic Web and PEER TO PEER seems to be a good basis for the future of Web service discovery and publication.

Recently, the technology of Web service is widely used by researcher. In [20], the authors develop and evaluate parallel and distributed simulation using the web services technology for efficient simulation execution. A prototype framework is implemented using web services for a simple shop floor simulation, where the authors focus on web service based simulation optimization through the distribution of simulation replications across different servers. As an optimization algorithm, the ranking and selection procedure and the extended Optimal Computing Budget Allocation (OCBA) are employed. The development of parallel and distributed simulation using web services will help to solve large-scale problems efficiently and guarantee interoperability among heterogeneous networked systems.

Web searching is such an activity that its importance can just not be ignored in the current scenario. A number of public search engines are available for this purpose. Every day, hundreds of millions of requests are handled by a single search engine. Due to this reason, new and efficient but complex search systems are being expected in future. But, the problem of evaluating the quality of search results always remains there. In [21], the architecture of a comprehensive web search evaluation system is being proposed. The authors use a number of objective evaluation techniques to supplement the subjective evaluation based on user feedback. The user feedback is obtained implicitly by watching the actions of user on search results in response to his query. These techniques are combined using Modified Shimura technique of Rank aggregation. The aggregated ranking is then compared with the original ranking given by the search engine. The correlation coefficient thus obtained is averaged for a set of queries. The averaged correlation coefficient is thus a measure of web search quality of the search engine. The search engines then can be evaluated on the basis of the search quality by comparing the correlation coefficients.

At this moment web development is directed to Social Web which promotes collaborative tasks that needs to be considered for learning. Activities developed trough e-learning systems needs to be supported by some learning theories in order to promote their quality. The type of information to manage for e-learning is a topic that has led to the emergence of new concepts for resource development like Learning Object (LO) concept. [22] is an awareness of the elements that should be considered for quality e-activities taking into account LOs instructional design for e-learning systems in Social Web. According to this, the authors analyze the most important ideas from learning theories that needs to be rescued for quality e-activities and Social Web characteristics for its development. On this basis, authors propose a model to develop quality e-activities for e-learning systems in Social Web.

In our work, we use domain ontology for semantic Web service annotation. A DHT based catalog is used to store the semantic indexes for direct and flexible service publication and discovery. With a large number of electronic commerce application, each only focuses on publication or discovery of Web services with interest to their respective Virtual Organizations (VO). They are reluctant to partake irrelevant information of catalog services. Also, searching for a particular Web service would be very difficult and inefficient in an environment consisting of thousands of service providers from different Virtual Organizations. In our work, we organize domains into category ontology for partnership federation. This ontology maps each service to a specific domain thereby grouping them based on domains. In this way Web services publication and discovery could be limited to only the services in that specific Virtual Organizations.

The rest of this paper is structured as follows. Section 2 briefly lists the related works. Section 3 presents an ontology based Web service annotation method. The detailed explanation for distributed Web service organization model is discussed. In section 4, we have proposed the improved Web service organization model, which supports domain specific service publication and discovery. Section 5 presents an initial implementation. Section 6 gives a conclusion and some future work.

2 Related work

Currents approaches for Web service organization can be broadly classified as centralized or decentralized. The typical centralized approach includes UDDI [2], where central registry is used to store Web service descriptions. Three major operators, namely IBM, Microsoft, and ARIBA provide public UDDI service. The current UDDI attempts to alleviate the disadvantages of the centralized approach by replicating the entire information and putting them on different sites. Replication, however, may temporarily improve the performance if the number of UDDI users is limited. But, the more the replication sites, the less consistent the replicated data will be.

Having realized that replicating the UDDI data is not a scalable approach, several decentralized approaches have been proposed. [3] has proposed moving from a central design to a distributed approach by connecting private registries with PEER TO PEER technology, but it does not consider partner federations in a specific domain. [4] organizes peers into a hypercube. But maintaining the hypercube with large amount of peers is inefficient. It does not support domain specific service federation either. [5] uses a dimension reducing indexing scheme to map the multidimensional information space to physical peers, which supports complex queries containing partial keywords and wild-cards. But it does not utilize the semantics description. PEER TO PEER based Web service discovery is also discussed in [6, 7]. None of these works consider domain specific service publication and discovery. Our work is different as we use category ontology to categorize Web service on the basis of business domains. We also use domain ontology to capture domain knowledge and index the ontology information, store the index at peers in the PEER TO PEER system using a DHT approach.

3 DHTs Facilitated Decentralized Web Service Organization

Without a central services registry, a naïve way to discover a service in distributed system is to send the query to each of the participant, i.e. service provider. While this approach would work for a small number of service providers, it certainly does not scale to a large distributed system. Hence, when a system incorporates thousands of nodes, a facility is needed that allows the selection of the subset of nodes that will produce results, leaving out nodes that will definitely not produce results. Such a facility implies the deployment of catalog-like functionality.

Recently, a new generation of PEER TO PEER systems offering DHT functionality has been proposed. These systems greatly improve the scalability and exact-match accuracy of PEER TO PEER systems. This DHT functionality has proved to be a useful substrate for distributed systems. A DHT based catalog service can be used to determine which nodes should receive queries based on query content. Ideally, the PEER TO PEER service network will allow for issuing a query to be sent to exactly those peers that can potentially answer the query.

In this section, we will discuss DHT based decentralized Web services organization model. The DHT background and Chord protocol are first introduced. Based on this, we propose the system model for semantics indexing catalog service. Then the service discovery procedure is illustrated. At the end of this section, we have discussed the maintenance of system evolution.

3.1 DHT Background and Chord

PEER TO PEER is the decentralization from traditional central model to the decentralized service-to-service model. In this model no central index is required to span the network. Particular attention has been paid into making these systems scalable to large numbers of nodes, avoiding shortcomings of the early PEER TO PEER pioneers such as file sharing systems like Gnutella [8] and Napster [9]. Representatives of scalable location and routing protocols are CAN [10], Pastry [11], Chord [12] and Tapestry [13], henceforth referred to as Distributed Hash Tables (DHTs). Given a key, the corresponding data item can be efficiently located using only O(logn) network messages where n is the total number of nodes in the system [12]. Moreover, the distributed system evolves gracefully and can scale to very large numbers of nodes. Our work leverages this functionality to provide a scalable fully distributed catalog service. [14] show that Chord's

maintenance bandwidth to handle concurrent node arrivals and departures is near optimal. Thus we choose Chord as our DHT protocol.
Chord is a famous DHT protocol. Chord identifiers are structured in an identifier circle, which is called Chord ring. There is one basic operation in the Chord systems, lookup (key), which returns the identity of the node storing the corresponding data item with that key. This operation allows nodes to publish and query information based on their keys. The keys are strings of digits of some length. Nodes have identifiers, taken from the same space as the keys (i.e., same number of digits). Chord uses hashing to map both keys and node identifiers (such as IP address) onto the identifier ring (Fig. 1). Each key is assigned to its successor node, which is the nearest node traveling the ring clockwise. Depending on the application this node is responsible for associating the key with the corresponding data item. Thus it allows data request to be sent obliviously of where the corresponding items are stored. Nodes and keys may be added or removed at any time, while Chord maintains efficient lookups using just O(logn) state on each node in the system. For a detailed description of Chord and its algorithms, refer to [12]. Chord protocol is publicly available and has been successfully used in other projects such as CFS [15]. Nevertheless, our design does not depend on the specific DHT implementation and can work with any DHT protocols.

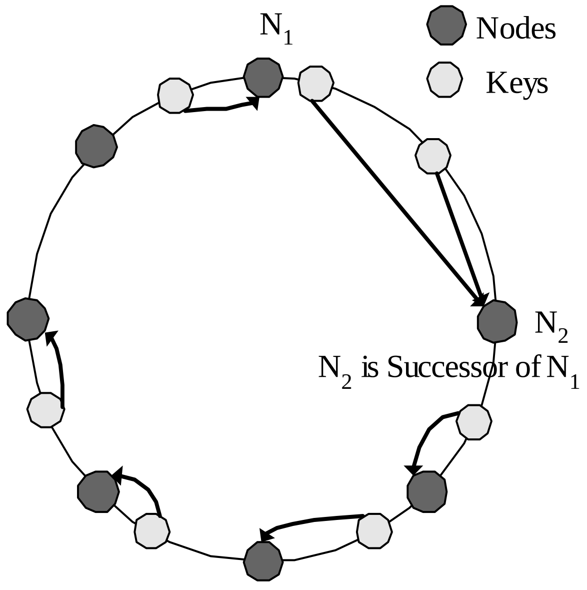


Fig. 1. The Chord identifier ring

3.2 Web service Semantic Annotation Based on Domain Ontology

Adding semantics to Web service descriptions can be achieved by using ontologies that support shared vocabularies and domain models for use in the service description [16]. We refer to related concepts in ontologies as annotation. While searching for Web services, relevant domain specific ontologies can be referred to, thus enabling semantic matching of services.

An ontology is a shared formalization of a conceptualization of a domain [17]. In our approach, we identify two aspects of semantics for Web service description elicited from object-oriented model. We also model concept from both attributes and behaviors aspects. As shown in Fig. 2, the concept Itinerary has attributes as start-Place, arrivalPlace and startTime etc. It is also associated with behaviors as search, book, cancel etc. The concept describes what functionality a Web service aims at. The behaviors aspect can be used to describe operations of a Web service. The attributes aspects can be used to describe the I/O interface of operations.

WSDL (Web Services Description Language) is the current standard of the description of Web service [18]. The syntax of WSDL is defined in XML Schema. From the left part of Fig. 2, we can see the simplified WSDL structure. We attempt to use minimum information to give Web service comprehensive annotations. Large distributed system can’t afford much detailed information although ontology can capture much information of real world knowledge and domain theory. This kind of information can be incorporate into WSDL or UDDI. By this method, Web service is semantically annotated in different granularity, which facilitates different service level discovery (see Fig.2). The following code exemplifies Web service annotation method.

```
<types>
  <xsd:element name="StartCity"
OntCocept="Itinerary:startPlace"
minOccurs="1" maxOccurs="1"
type="xsd:string"
  <xsd:element name="ArrivalCity"
OntConcept="Itinerary:arrivalPlace"
minOccurs="1" maxOc-curs="1"
type="xsd:string"
  <xsd:element name=" time"
OntConcept="Itinerary:startTime"
minOccurs="1" maxOccurs="1"
type="xsd:string"
  ……
</types>……
```

```
<portType>
  <operation          name="SearchItineary"
OntConcept="Itinerary:search">
  ……
  </operation>
</portType> ……
<service           name="ItineraryService"
OntCocept="AirTravel:Itinerary"
  ……
</service>
```

We add semantics to Web services by mapping service, operation, input and output in their descriptions to concepts in the domain specific ontologies. As the above WSDL code shows, the Web service is annotated in three aspects: service Itinerary-Service is annotated by AirTravel:Itinerary, operation SearchItineary is annotated by Itinerary:search and i/o element StartCity is annotated by Itinerary:startPlace etc.

3.3 DHT Based Web Services Publication and Discovery

Let $\{N_i\}$ denote the n providers (a provider is a node in the identifier ring), each of which publishes a set C_i of Web services. When a node N_i wants to publish Web services, it creates catalog information, which is the set $C_i = \{(k_j, S_{ij}) \mid S_{ij}$ is a summary of k_j on node $N_i\}$. In Sect. 3.2, we have discussed the semantic annotation of Web services. Thus the concept should be the key, i.e. k_j, as mapped by Chord protocol and *Behavior(Attributes Set)* should be the corresponding data item associated with the key, i.e. S_{ij}, the summary of k_j. As a Web service consists of several operations, there are sets of data summaries S_{ij}, and not just single data summary.

For a service discovery, the requestor should first choose appropriate concept to which the preferred service may refer. Then he (or she) decides the interested behaviors and attributes he would like to provide as input for the preferred Web service. As an example, a service requestor wants to inquire about the itinerary information from Shanghai to Beijing on 1 October 2004. Considering the ontology in Fig. 2, the requestor may choose *Itinerary* as the domain concept, *search* as behavior and *startPlace*, *arrivalPlace*, *startTime* as attributes. This is the first step for service discovery.

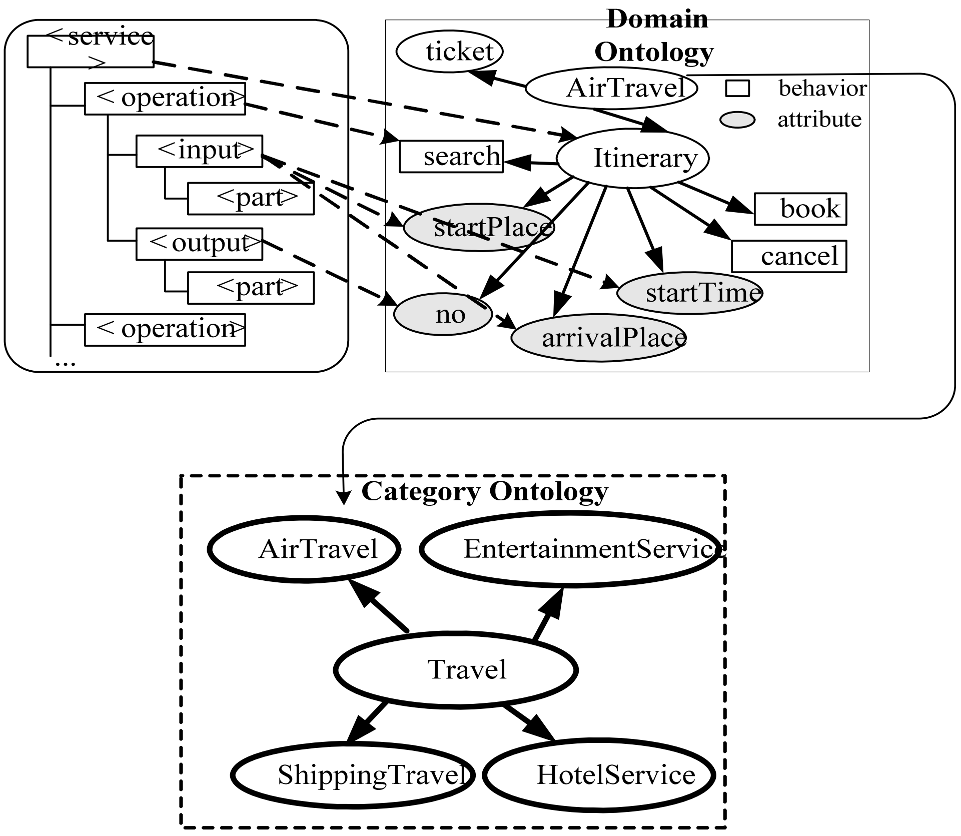
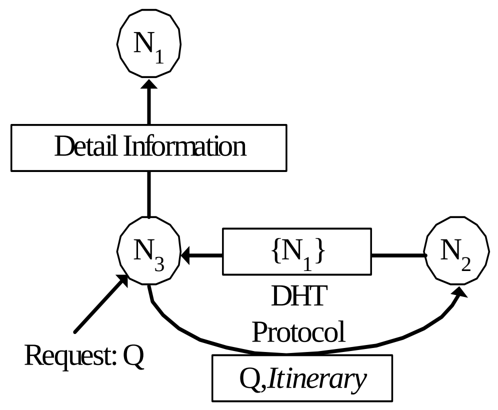
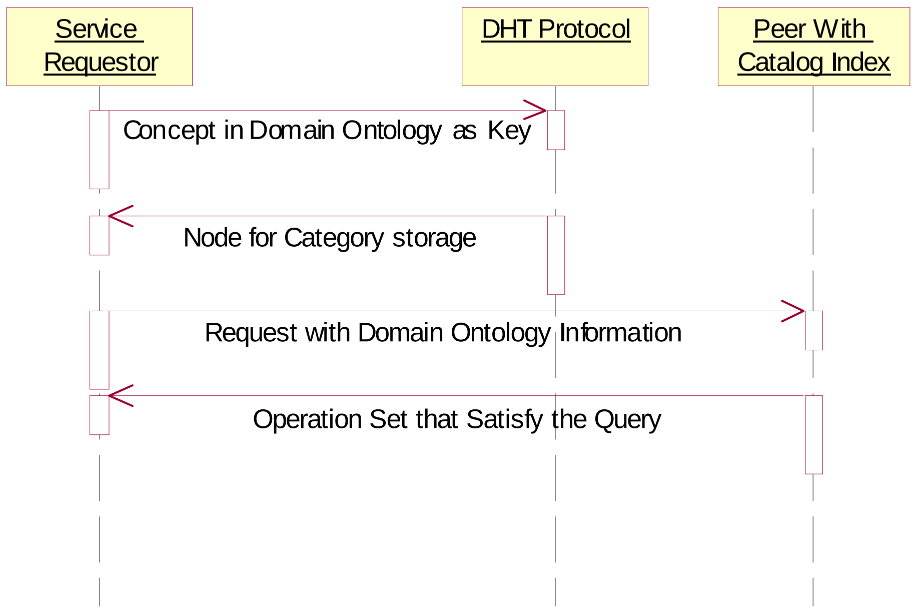


Fig.2 The left part illustrates the simplified WSDL structure. The right part is an ontology example. The arrow indicates semantic Web services annotation. The bottom part illustrates example of category ontology, which will be introduced in following sections.





Service Requestor
DHT Protocol
Peer With Catalog Index
Concept in Domain Ontology as Key
Node for Category storage
Request with Domain Ontology Information
Operation Set that Satisfy the Query

Fig. 3 The above part illustrates Web Service Discovery Procedure. The below part illustrates interaction of DHT facilitated Web services discovery process

An example illustrates services discovery procedure. We assume that there are four services providers: N1, N2, N3 and N4. The service request is submitted on N3. Itinerary serves as the DHT lookup key and the hash algorithm returns that the summary of Itinerary lies in N2, which stores part of the catalog that contains Itinerary information. Then the request with domain ontology information is sent to N2. This is the second step (see Fig. 3). On N2, the behaviors and attributes of the query is matched against Si, Itinerary, (1<i<3). N2 replies to N3 with the node set {N1}. Here we assume that only S1, Itinerary matches the given query and so N1 is the only node that satisfies the request. This is the third step. Finally, N3 contacts N1 for detail information, e.g. WSDL. Fig. 3 illustrates the procedure of Web services discovery and the interaction of DHT facili-tated Web services discovery process.

3.4 System Evolution

When a node joins/leaves the system, the affected data structure on some existing nodes must be updated accordingly to reflect the change. This section describes how the distributed catalog service evolves when nodes join, leave and update their data, and how objects, which are the sets of data summaries, are stored and accessed.

3.4.1 Nodes Joining

Each new node N_n (service provider) that joins the system creates the Service set C_n, which should be queryable by the nodes already in the system. First N_n contacts any node N_c in the system (Step 1, Fig. 4). Node and key have the same string space. N_n gets a position in the identifier ring by the same hash algorithm. Chord finds N_n's successor N_s in the identifier ring. The new node N_n, now part of the identifier ring, injects each $(k_{ni}, S_{ni}) \in C_n$ into the system and the Chord protocol decides which nodes should receive the new catalog information (Step 2). Additionally, N_n becomes part of the catalog infrastructure and should share the load of the catalog service by hosting parts of the summary sets already in the network. The Chord protocol will assign to N_n keys for which N_n should be the successor in the identifier ring. Therefore when N_n joins, k_1, k_2 and their corresponding summaries should be reallocated from N_s to N_n (Step 3), as shown in Fig. 4.

3.4.2U pdates and Departures

Catalog information stored in the system may need to be updated as the provider makes changes to their services. Update requests are handled in a similar way as insertion of information during nodes joining (Step 2, Fig. 4). Only the node that has created the data summary (the owner) is allowed to change its content. Nodes that store the data summaries are not allowed to alter their content, although they may alter the way they are stored. When a node N decides to leave the system, it must hand over the catalog information to its successor according to the Chord

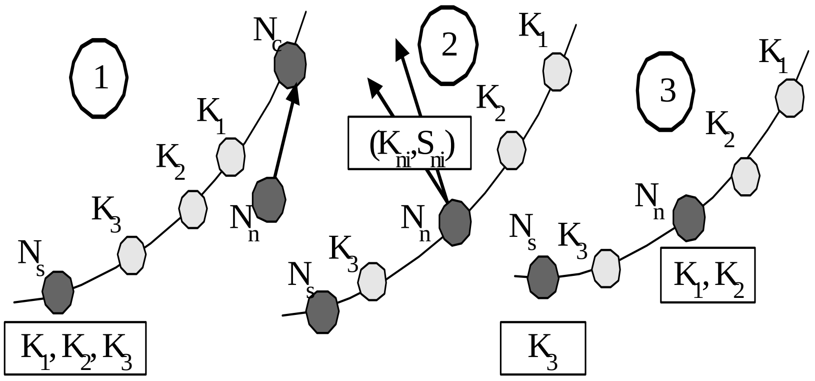


Fig. 4. When a Service provider joining the system, three steps is needed for system evolution

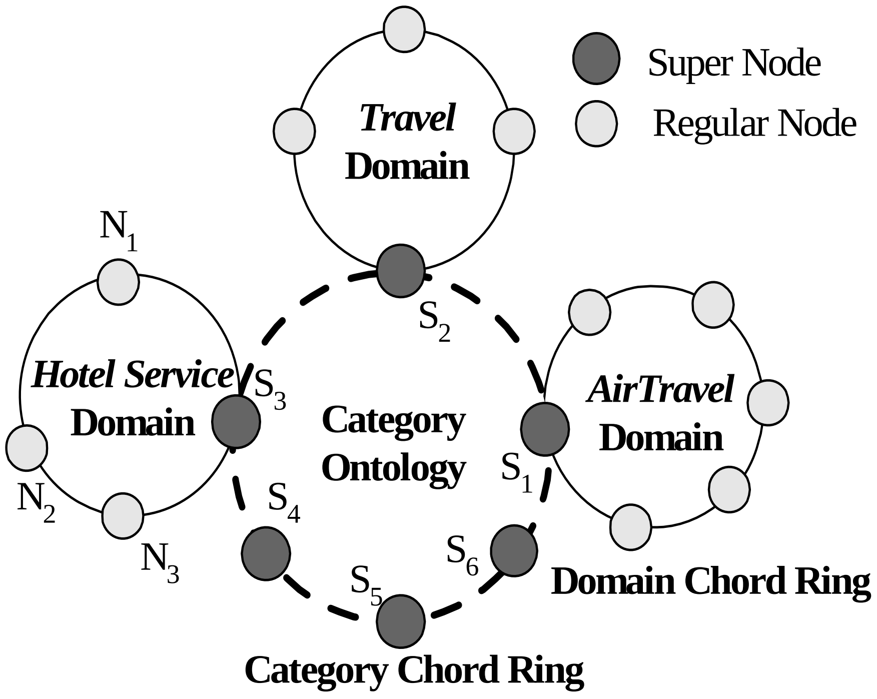


Fig.5 Category ontology facilitated Web service organization for partnership federation

protocol. Furthermore, it notifies the nodes that hold N's catalog data. To achieve this, N uses the keys it inserted into the system to find the nodes that currently hold N's catalog information.

4 Category Ontology Facilitated Web Service Organization for Partnership Federation

DHT based Web service discovery realizes the distributed service discovery. It doesn't require a public service registry. But with the large number of services, every peer shares much catalog information irrelevant to its interest. The challenge of dealing with thousands of services during service publication and discovery becomes critical. In this section, we propose an improved architecture which allows peers form federate partnership with common domain interests.

4.1 Organizing Domain into Category Ontology

If we could categorize all these distributed services based on different business do-mains, finding the right services would be easier. In our approach, we create another kind of ontology: category ontology. The category ontology is used to categorize Web service based on domains and it maintains relationship between domains. It maps each service to a specific domain thereby grouping all the Web services based on domains. This allows that domain related services are assembled together. Thus finding a specific Web service in a specific domain could be limited to only those domain related services. As shown in Fig. 2, the right part presents an example of category ontology, and the middle part is the domain ontology.

4.2 Improved Web Service Organization Architecture

The improved decentralized Web service organization architecture is shown in Fig. 5. There are two kinds of Chord ring for peer node organization. Domain Chord ring is used for organization of specific domain services. The category Chord ring links the separate domain specific services in their respective domain Chord ring together. The intention of this framework is to promise a pure PEER TO PEER network suit for distributed application. The node in category Chord ring is called super node. Other nodes are called regular nodes. The domain Chord ring functions the same as in Sect. 3 for domain specific services publication and discovery. The category Chord ring differs from domain Chord ring in the information being routed: the category Chord ring provides category ontology catalog service for each domain Chord ring. That is to say, the category Chord ring routes category information to each super node in its Chord ring. Thus the super node shoulders not only the Web service information in its do-main but also part of the category ontology information.

4.3 Web Services Publication and Discovery

When a service provider joins the infrastructure to publish his service, he has to choose an appropriate domain to which his business maps. A service provider is also allowed to map to multiple domains. If a new service provider wants to map to a do-main that does not exist in the category ontology, he is allowed to create a new do-main to map. Hence the service provider can either associate to an existing domain in the ontology or he can update the ontology with an appropriate domain and associate to that new domain.

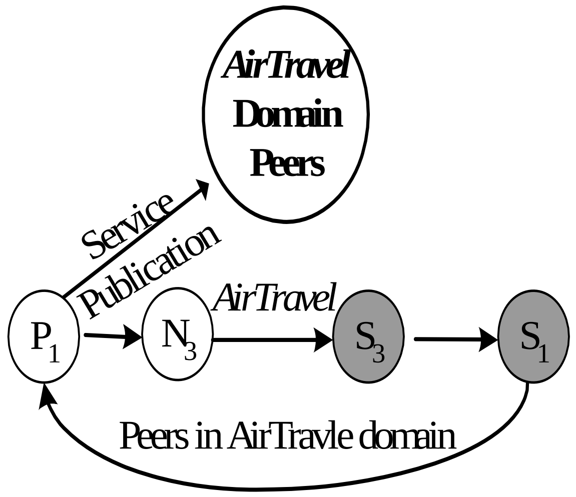


Fig. 6. Web service publication procedure combing domain ontology and category ontology

Considering the category ontology in Fig. 2, we assume that the DHT protocol routes Travel to node S2, AirTravel to S1 and Hotel Service to S3, as shown in Fig.5. If a service provider P1 in AirTravel domain wants to publish his service, we assume that he first contacts with node N3 in Hotel service Domain Chord ring. N3 contact the super node S3 in its Chord ring and issues the AirTravel category information to S3. Based on the DHT algorithm, S3 knows that S1 is the super node for AirTravel Do-main. Then S1 returns the peers in AirTravel domain Chord ring and P1 contacts with any peer in

AirTravel domain for service publication. The left service publication procedure acts as the same with in Sect. 3. Fig. 6 illustrates service publication process. Service discovery is similar to service publication. Service requestor should first refer to a domain in category ontology. The super node returns the domain specific peers and the requestor discovers appropriate services with the same procedure dis-cussed in Sect. 3.3.

5 Implementation and Experiments

We have implemented an initial prototype to illustrate the organization model we have proposed in Sect. 3. The prototype is based on the Chord protocol implementation found on the Chord project website which is linked as a library. WordNet 2.0, an on-line lexical database for the English language, is embedded into the system through APIs to further understand the semantics of Web services described [19]. As the over-lay network configuration and operations are based on Chord [12], its maintenance costs are of the same order as in Chord. An evaluation of the Web service discovery exactness and the system scalability is presented below.

5.1 Web Service Discovery Exactness Evaluation

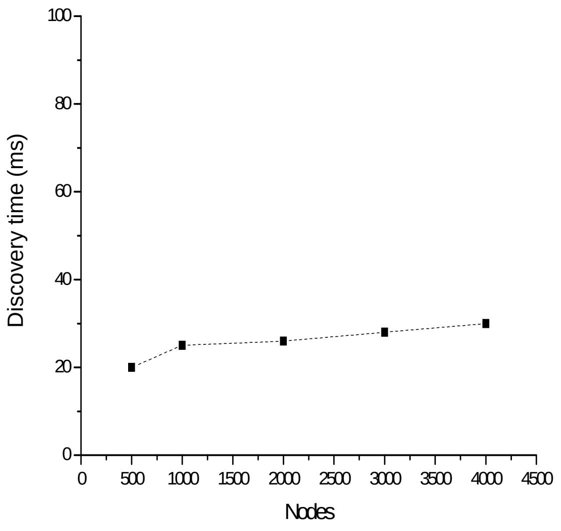
	Weather Domain		Geographical Domain	
	Correct Rate	Error Rate	Correct Rate	Error Rate
(a)	20%	0	17%	0
(b)	88%	0	91%	0

Table 1. Web service discovery exactness evaluation results: a collection of Web services in Weather and Geographical domains are used as samples

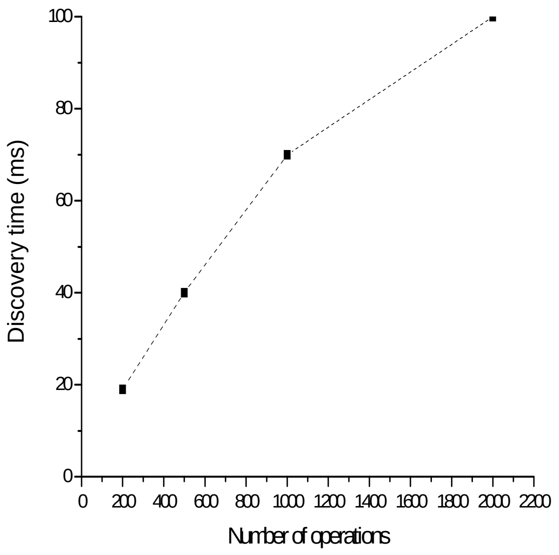
To test service discovery exactness we first obtain a corpus of Web services from SALCentral.org and XMethods.com. We have limited our testing to two domains, i.e. Weather and Geographical domains, due to lack of relevant domain specific ontologies. We compare the discovery results in two circumstances: (a) Web service is not annotated with ontology information. The service names are directly used as the key hashed to catalog indexes. (b) The Web services are manually annotated with domain ontology. From the results in Table 1, we can conclude that method (a) has low correct rate and error rate. Comparatively, method (b) gets most of the services satisfying the request.

5.2 System Scalability Evaluation

In this section, we will evaluate the system scalability by constantly changing the number of nodes and operations in Web service discovery.



(a)



(b)

Fig.7 System scalability evaluation results: we evaluate system scalability by evaluating the discovery time with number of nodes changes (a) and number of operations changes (b)

From (a) in Fig. 7, we can see that the discovery time changes steadily with peer nodes increase. This is because the discovery time mainly results from the matching of query against the summary in the catalog index. Thus it is not obviously irrelevant to the number of nodes. On the other hand, with the increasing of number of operations, the catalog index becomes larger, which results in longer discovery time as show in (b) of Fig. 7. Because the matching is processed only in one node, the increase ration of discovery time is not obvious. It inclines to stabilization. Thus this is not a critical problem for application in large distributed system. Also this can be overcome by improving the matching algorithm.

6 Conclusions and Future Work

This paper presents flexible Web service organization architecture for service publication and discovery by combining semantic Web service with PEER TO PEER networks. This system does not need a central registry for Web service discovery. We use an ontology-based approach to capture real world knowledge for semantic service annotation. A DHT based catalog service is used to store the semantic indexes for direct and flexible service publication and discovery. For partnership federation, we have made improvements for this architecture. We use category ontology to organize Web service into Virtual Organizations. Service publication and discovery is within a Virtual Organizations. Our initial experiments have shown that the semantic annotation approach suggested in this paper will significantly improve Web services discovery exactness. With Web services being as the enabling technology for next generation network, we believe that this service organization infrastructure will help Virtual Organizations in carrying out their goals in a more scalable environment.

Due to lack of relevant specific ontology, we have not conducted experiments about category organization. In the future work, we will put this Web service organization architecture on the grid, which will enable different organization and research workshop to test it. This will give more comments for us to improve this work.

References:

[1] R. Siebes: Peer-to-Peer solutions in the Semantic Web context: an overview. EU-IST Project IST-2001-34103 SWAP, Vrije Universiteit Amsterdam (2002)

[2] UDDI. UDDI white papers. http://www.uddi.org/whitepapers.html

[3] U. Thaden, W. Siberski, and W. Nejdl: A Semantic Web based Peer-to-Peer Service Registry Network. Technical Report, University of Hanover, Germany (2003)

[4] M. Schlosser, M. Sintek, S. Decker and W. Nejdl: A Scalable and Ontology-Based PEER TO PEER Infrastructure for Semantic Web Services. PEER TO PEER'02, Linkoping Sweden (2002)

[5] C. Schmidt, M. Parashar: A Peer to Peer Approach to Web Service Discovery. World Wide Web, Vol. 7, 2 (2003) 211–229

[6] M. Paolucci, K. Sycara, T. Nishimura, N. Srinivasan: Using DAML-S for PEER TO PEER Discovery. Proceedings of ICWS'03, Las Vegas USA (2003) 203–207

[7] A. Maedche, S. Staab: Services on the Move - Towards PEER TO PEER-Enabled Semantic Web Ser-vices. Proceedings of the 10th International Conference on Information Technology and Travel & Tourism, Helsinki Finland (2003)

[8] Gnutella Resources. http://gnutella.wego.com

[9] Napster. http://www.napster.com

[10] S. Ratnasamy, et al: A Scalable Content-Addressable Network. Proceedings of ACM SIGCOMM2001, San Diego CA USA (2001)

[11] A. Rowstron, et al: Scalable, distributed object location and routing for large-scale peer-to-peer systems. IFIP/ACM International Conference on Distributed Systems Platforms. Heidelberg Germany (2001) 329–350

[12] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan: Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking. Vol. 11, 1 (2003) 17–32

[13] B. Y. Zhao, et al: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. U. C. Berkeley Technical Report UCB/CSD-01-1141 (2001)

[14] D. Liben-nowell, H. Balakrishnan, D. Karger: Observations on the Dynamic Evolution of Peer-to-Peer Networks. Proceedings of IPTPS'02, Cambridge MA (2002) 22–33

[15] F. Dabek, et al: Wide-area cooperative storage with CFS. Proceedings of SOSP '01, Canada (2001)

[16] R.Akkiraju, et al: A Method for Semantically Enhancing the Service Discovery Capabilities of UDDI. Proceedings of IIWeb'03, Acapulco Mexico (2003)

[17] M. Uschold and M. Grüninger: Ontologies: Principles, Methods and Applications.

Knowl-edge Engineering Review, Vol. 11, 2 (1996)

[18] E. Christensen, et al: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. http://www.w3.org/TR/2004/WD-wsdl20-20040326/ (2004)

[19] G. Miller: Special Issue, WordNet: An on-line lexical database. International Journal of Lexicography. Vol. 3, 4 (1990) 235–312

[20] Yoo, T., Cho, H., Yucesan, E.,Web service based parallel and distributed simulation experience, WSEAS Transactions on Systems, Volume 5, Issue 5, May 2006, pages 973-980, ISSN 1109-2777

[21] Ali, R., Beg, M.M.S., A framework for evaluating web search systems, WSEAS Transactions on Systems, Volume 6, Issue 2, February 2007, pages 257-264, ISSN 1109-2777

[22] Garcia-Penalvo, F.J.; Morales, E.B., Angela, Learning objects for e-activities in social web, WSEAS Transactions on Systems, Volume 6, Issue 2, March 2007, pages 257-264, ISSN 1109-2777