그리드 시스템에서 그룹유사함수를 이용한 확장성 있는 자원공유☆

Scalable Resource Sharing using Group Similarity Function in Grid System

마테오 로미오* 이 재 완** 이 말 레***
Romeo Mark A. Mateo Jaewan Lee Malrey Lee

요 약

그리드 시스템에서의 확장성은 점증하고 있는 그리드 이용자를 수용하기 위해 효율적으로 설계되어야 한다. 본 논문은 그리드환경에서 확장성 가상조직(SVO: Scalable Virtual Organization)을 사용하여 VO들 간의 확장형 자원 공유 기법을 제시한다. 제안하는 그리드 구조는 SVO를 관리하기 위한 그리드 시스템의 표준 서비스들로 구성되며, 통합할 VO들을 선택하기 위해, VO들 간의 유사도를 결정하는 데 사용하는 그룹 유사함수를 제시한다. 다른 유사 함수들을 제시한 기법(GSF)과 비교하고, 시뮬레이션을 통해 SVO의 성능을 평가하였다.

ABSTRACT

The scalability of a grid system should be efficiently designed to handle the increasing number of grid users. This paper presents a scalable resource sharing mechanism among virtual organizations (VO) in grid environment by the scalable virtual organizations (SVO). The proposed grid architecture is composed of standard services of a grid system to manage the SVO. We propose a group similarity function which is used to determine similarities among VOs to select the VOs to merge. We compared other similarity functions to the GSF and determined the throughput performance of the SVO using simulation for grid system.

☞ KeyWords : 유사함수, 가상조직, 그리드 시스템, 자원관리, Similarity Function, Virtual Organization, Grid System, Resource Management

1. Introduction

Grid system is a widely used architecture for sharing resources, computational and non-computational, within the Internet. This technology includes the concept of resource virtualization, on-demand provisioning, and service sharing between organizations [1]. Enhanced collaboration of researchers and scientists by sharing data or resources through different institutions is a well-known contribution of grid computing. In the business side, researchers already utilize grid technology [2]. Grid-enabled service implementation for e-business is the logical next step on the road of IT market to the ubiquitous connectivity, virtualization, service outsourcing, product commodization and globalization [3]. New paradigms like cloud computing emerged to adapt this business trend. Even though grid computing evolves in adapting a certain trend, the abstraction techniques from virtual organization are considered vital in resource provisioning.

Virtual organization usually spans across several

* 정 회 원 : 군산대학교 전자정보공학부 박사과정
rmmateo@kunsan.ac.kr
** 종신회원 : 군산대학교 정보통신공학과 교수
jwlee@kunsan.ac.kr (교신저자)
*** 정 회 원 : 전북대학교 영상정보신기술연구센터, 컴퓨터공학부 교수
mrlee@chonbuk.ac.kr

☆ This research was supported by grant R01-2006-000-10147-0 from the Basic Research Program of the Korea Science and Engineering Foundation
[2010/03/27 투고 - 2010/04/12 심사 - 2010/07/15 심사완료]

institutions or domains where resource providers and users join and leave a virtual organization on the fly. The technology behind this dynamic method is the grid middleware. Open Grid Service Architecture (OGSA), which is a standard specification of services for Grid and was initialized by the Global Grid Forum (GGF) [4], is mostly referenced in designing a grid system. Also, thanks to the Web Service Resource Framework (WSRF), grid services are easily implemented based on web service standards. These are already incorporated in the Globus toolkit [5]. The componentized structure of OGSA provides customized middleware services and efficient management of resources in grids. However, most designs of Grids limit its interactions inside a virtual organization and thus not aware of resources in other virtual organizations. In addition, the current design of grid middleware does not support the inter-joining of VOs to enable scalable resource sharing.

In this paper, we propose the scalable virtual organizations (SVO) in our design of a grid middleware to supports scalable resource sharing in virtual organizations. The proposed components are integrated into the services of OGSA to improve the Monitoring and Discovery System (MDS) and, Grid Resource Allocation and Management (GRAM). Our approach tackles the scalability of resource sharing in MDS using a proposed grouping service and includes the efficient distribution of loads in GRAM using a proposed load balancing service. In the process of SVO, the proposed group similarity function (GSF) is used to determine similar VOs to perform merging of resource information. The SVO is initialized by the grouping service which provides the GSF and then VO agents implement the SVO where it uses the function to request additional resources to other similar VOs. We evaluated the performance of the SVO using our simulator for dynamic virtual organizations in grid environment.

2. Related Works

2.1 Virtual Organization

Virtual organization (VO) is initially formed by grid users performing a specific goal. It is assumed that entities joining a VO will contribute by sharing their resources to all participants. The interactions from entities are hidden to users and constraints are set by an administrator. Most technologies in VO are focused on enhancing technical requirements and security issues [6, 7]. Another issue is grid users having different software used in the computer system which decreases the efficiency of using resources from among users. A virtual organization cluster model [8] is used to manage the isolation of VO software from the physical system. The dynamic characteristic of a VO is also integrated with multi-agent system [9, 10]. In providing a quality of service (QoS), the waiting time from task processes and latency from message sending should be minimized. This can be done by acquiring more resources and distributing the tasks efficiently within the Grid system. We propose the scalable virtual organizations (SVO) to search and to merge appropriate VOs for additional resources which minimizes the waiting time and increase the throughputs of a Grid system.

2.2 Similarity Measures

Similarity function is a method to calculate the similar features of objects being compared. The objects are provided with properties and the similar properties of two objects are processed in a distance function. After processing all properties, a score is

provided for objects that were compared. In Equation 1, the function f provides a score (s) between two objects based on their similar properties which is presented by vectors v_1 and v_2. A good review of the pairwise similarities between individual objects that belong to a taxonomy is given in [11]. This method is commonly used on classification and data matching [12].

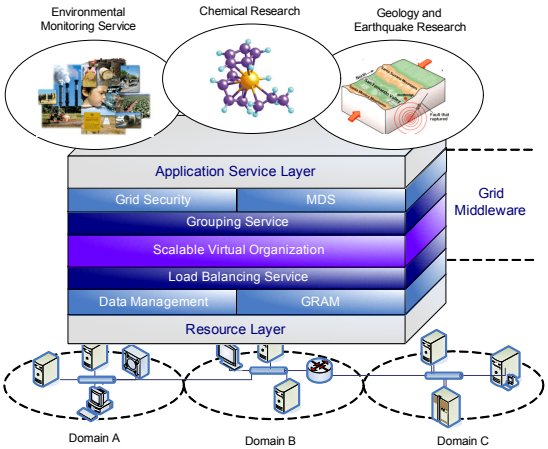
$$f(v_1, v_2) \rightarrow s \quad (1)$$

In the set-based approaches, the similarity is computed using set similarity measures such as Dice, Jaccard, or cosine [13]. Ganesan et al. [14] develop several similarity functions for information retrieval using different techniques. In the graph-based approaches, the objects in each set are considered as a graph that is a part of the original taxonomy. Also, this problem is encountered in many domains where the information can be represented as a graph, such as 2D shape [15], 3D structure matching [16], and MESH-based document retrieval [17]. A grid resource discovery is used to provide a set of candidate resources according to the given attribute set. The discovering strategies are well researched, however, little work has been done in using similarity measures to discover resources in Grid. In [18], a similarity measures is proposed for feature spaces. A grid resource model which is the Hyper Topology Space based Resource Model (HTSRM) is designed for effective discovering resources based on the proposed similarity measure [18]. In this paper, we proposed a group similarity function where we are not only specific on comparing two objects like in [18] but approximating the similarity of VOs using their frequency of use of resources.

3. Grid Architecture based on SVO

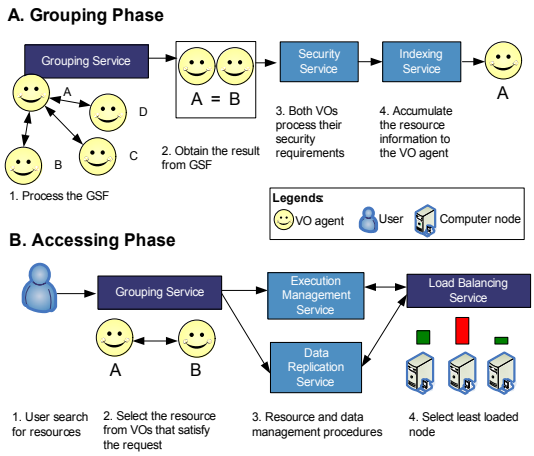
The current design of virtual organizations (VO) in grid does not consider the scalability where grid users are limited only in accessing resources within the VO. In our work, we define VO scaling as a procedure of finding VOs with similar interests and utilizing each other resource. By using this method, each VO can be aware of relevant resources in other VOs and use those resources. Figure 1 shows the scalable grouping that overlaps their resource providers with same interests on other groups by the same resources it provides. The proposed scalable virtual organization (SVO) is designed to find additional resources using similar extractions of interest among VOs. This also includes the dynamic method of joining and leaving of merged VO. Moreover, load balancing is also considered in the middleware design. In [19], the research study describes a similarity grouping to increase performance of a P2P system. Similar to our research, we used a group similarity function to find additional resources which increases the throughput of the Grid system. We propose a grid system based on the SVO which is an integration of a scalable scheme to the OGSA. Our SVO scheme is situated between the major components of the OGSA shown in Figure 1 which is supported by grouping and load balancing services. We identify these relationships; 1) grouping service works with Grid security and MDS to support the VO scaling while, 2) load balancing service works with Data Management and GRAM for the efficient distribution of tasks. The grouping service determines the security requirements of the grid system and initially registers the available resources from MDS. The load balancing service notifies both the GRAM and Data

Management about the load trends from the system.



(Fig 1) Grid architecture based on SVO is a layered approach of integrating the proposed scalable scheme with the services of OGSA.

The application layer represents different enhanced science or e-science applications like environmental monitoring, earthquake research and chemical research are grouped by VO. Users in each domain are separated by domains but they can share their resources to other domain via VO. MDS manages the indexing of resources and grid security does the authentication of users and delegation of resources for application layer. A resource provider after joining a VO, they can share and use resources base on the policies of a VO administrator. Rules and policies are implemented in a VO administrator server. A user can join multiple VOs after meeting their requirements. This procedure also includes the security and trust procedures but will not be focused in our work. Moreover, the grouping service is used for the VO scaling to enhance the scaling of resources. We use VO agent to find another VO that has similar interests and merge their resources. The resource layer shows the abstraction of computers in the network. These classifications of computers are highly computational-capable computers and connected on different locations. GRAM manages the scheduling of tasks and Data Management is responsible for file transfer and data replications. The load balancing service is working on the resource layer to support efficient load distribution generated by resource sharing of VOs.



(Fig 2) SVO two phases 1) grouping phase and 2) accessing phase. The components of SVO interact with the current services of OGSA to promote scalable resource sharing and efficient load distribution.

SVO is implemented in two phases shown in Figure 2. The grouping phase starts from the inquiry of a VO agent to other VO agents and calculate their similarities. The grouping service handles the group similarity procedure. After obtaining the result, VO agent starts the merging process which includes the security procedures. The technical details of the security and providing policies are not scope by this paper. All the resource information is gathered by each VO agent from the merged VOs after meeting such requirements. In the accessing phase, a user can request a resource found within the merged VOs which is handled by grouping service. In this work,

a resource can be a data or software service which is hosted by a node. The use of a resource is also utilizing the hardware resources of a node. VO agents use the merged resource information to find the appropriate resource for the request and if successful then the request is forwarded to the execution management service. This procedure is hidden to the clients and only the system is aware of it. Concurrently, execution management service communicates to the load balancing service to assist on deciding which node is appropriate for the task based on current loads. After this procedure, load balancing service returns the node address to the execution management service to execute the task. Also, the load balancing service verifies the need of additional data to request the data replication service for necessary data replication.

4. Scalable Virtual Organizations

Each resource in a VO is described by resource tags. A resource tag is provided with scores which is the resource use frequency. Similar VOs are determined by having near score values of resource tags. The proposed group similarity function (GSF) analyzes the similarity of VOs based on the mentioned assumption. First, the resources are labeled with tags which are configured by a resource provider. Before registering a resource in a VO, the resource provider is required to choose appropriate tags for their resources and if tags do not exist then the provider can create their own tags. A VO server contains information of resource tags where a resource provider can view and add their tags. New tags are stored in the VO server and these can be accessed by all resource providers to label their services, and thus called global tags. After labeling a resource with tags, the tags will be stored locally in a VO and these are only accessed by the local agents in the VO to perform the GSF, which are called local tags. Every time a resource is used, the associated tags on that VO are provided with scores. To verify other VOs that having same or related resources, the VO agent gathers similar resource tags to all other VO agents and process the GSF to each VO. After identifying and merging similar VOs, the VO agent can negotiate and request resources from those VOs. Also, the VO agent manages the authentication and authorization policies of the merged VOs based on their current security requirements. The common security management of VO stated in [6] is followed by this research.

4.1 Merging Virtual Organizations

We define g as a collection of VOs, G_i refers to a single VO and x is a service. Each G_i, has O_i which is the collection of services in a VO (O_i ={x_1, x_2,⋯,x_n | n is the number of x}). The services is labeled with resource tags (t) which are collected, x_n = {t_1, t_2,⋯,t_k | k is the number of t }. Each VO has resource tags, G_i = {t_1, t_2,⋯,t_k } that serves as an input for the GSF. GSF processes the resource tag values of G_i to other VOs in g and the VO that have similar resource tags to G_i is collected to g' where g' = {G_1, G_2,⋯,G_j |$G_j \neq G_i$}. The result from GSF in Equation 2 will be values from [0,1] which is a percentage of similarity between G_i and G_j.

$$gsf = f(G_i, g') = \begin{cases} if\ gsf = 1, equal \\ if\ gsf > 1, less \end{cases}, \quad (2)$$

Before calculating the GSF, we collect the similar attributes from G_i and G_j in Equation 3. s represents a tag element in G_i and t a tag element in G_j. Similar tags are collected in △ with scores. All these

are appended in *gs* shown in Equation 3 and will be used in calculating GSF.

$$gs = \{\Delta(s,t), \forall t_A = s_A \mid s_A \in G_i, t_A \in G_j \subset g'\} \quad (3)$$

In Equation 4, we calculate the mean of all similar tags of VOs. In every element in *gs*, the values of each score is mapped by *m()* function where *m(s)* is for G_i tag score and *m(t)* is for G_j score. After getting the values, a range function in Equation 5 is used to determine the similarity values.

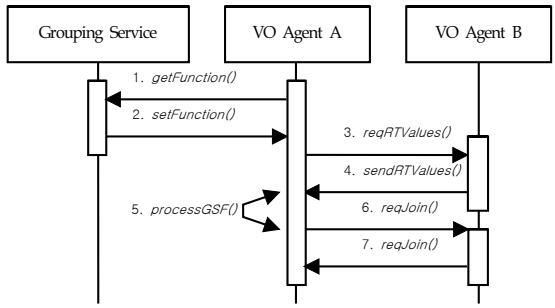
$$= \frac{1}{N} \sum_{m \in gs} r(m(s), m(t)) \quad (4)$$

The range function in Equation 5 calculates the Euclidian distance of *s* to *t* which is divided to the base value. The base value is determined by the distance of maximum (*max*) and minimum (*min*) score values. All similar tags use this range function and then calculate the mean in Equation 4.

$$r(s,t) = \left\{ \frac{|s-t|}{\max - \min} = [0,1] \right. \quad (5)$$

$$gsf = \{G_j \mid \forall f(G_i, G_j) > \Phi_G\} \quad (6)$$

After calculating all VOs in g', we collect VOs that satisfies the required GSF value for merging which is shown in Equation 6. The GSF value of each G_j is compared to a merging threshold (Φ_G). In our approach, similarity is determined by higher GSF because of the range function which is contrast to a distance function. The procedure collects more similar VOs if a smaller value was set on Φ_G. After gathering similar VOs, each VO processes their credentials and implements the scalable resource sharing.



(Fig 3) Interaction of VO agents and grouping service on processing the group similarity function and merging of VOs.

Figure 3 illustrates the interaction of VO agents on processing the GSF. 1) First, a VO agent determines the function that was configured from grouping service. 2) Then, the VO agent implements the grouping function that was set in the grouping service. The GSF is coded inside a module of grouping service and this module is extensible to new functions. In this paper, we only used the GSF to perform scalable resource sharing. 3) VO agent sends a request message to every VO agents for resource tag values (RTV). 4) Other VO agents reply with RTV to the VO agent and then 5) RTV are processed to the GSF procedure. After determining similar VO, they process their merging after meeting their requirements in steps 6 and 7 of figure 3.

4.2 Disjoining the Merged Virtual Organizations

The merging of VOs is based on properties of resource tags. A VO that was merged will share its resources to other VO. We define *gj* as GSF value of G_j that was merged on G_i. In our paper, we consider the contribution of each G_j based on forwarding the request to the appropriate node of G_j. We define G_i as the source of GSF process and G_j as sub VO that was merged from G_i. The contribution value of each sub VO is incremented

after forwarding a request to a node from other VO. In Equation 7, the calculation of a VO contribution ratio is shown. contrj is the incremented contribution value of G_j which is divided by the total contribution of all VOs in G_i.

$$vocontr_j = \frac{contr_j}{\sum_{k=1}^{N} contr_k} \quad (7)$$

A threshold value (Φ_j) determines the minimum contribution value for G_j to maintain its membership to G_i. Equation 8 is the ratio of GSF of gj to all GSF values in G_i. Equation 9 is the calculation of Φ_j where p is the ratio of contribution subtracted with a tolerated value. The tolerated value is a product of contribution ratio and a tolerance value t where t is manually configured.

$$p_j = \frac{g_j}{\sum_{k=1}^{N} g_k} \quad (8)$$

$$\Phi_j = p_j - (p_j \times t) \quad (9)$$

If $vocontr_j < \Phi_j$ then unmerge the G_j to G_i, otherwise, G_j maintains its membership to G_i. The condition also means that if a sub VO has less contribution then it will be unmerged. G_j can be merged again after processing the GSF function.

5. Experimental Evaluation

Our proposed grid architecture was simulated using the Globus toolkit [5]. The grouping and load balancing services were integrated in the toolkit. We simulated the dynamic VO using a given simulation environment in Section 5.1. The simulation performed the merging based on Φ_G and unmerging based on Φ_j using the interaction from the grouping service and VO agents. In this paper, we used our simulator for dynamic virtual organization which is the simVO [20] for the performance evaluation of SVO. Our simulator was developed using Java 2 SDK for easy handling of codes and extensible in adding new functions. The simVO implements message handlers for the communication of agents, and schedule manager to schedule events of message passing and task processing from the distributed nodes. The load balancing schemes were considered in our simulation and the load balancing service was configured to use least loaded selection method.

5.1 Environment Configurations

We provided 10 resources which are data mining services. The properties of each resource are described in Table 1. We used tags to classify the use of the data mining service and specified the processing speed (PT) of the service. In using a resource, a client sends raw data to process in data mining and the service returns the result to client. Each node contains 5 resources and these were randomly assigned in each node. The processing time of all resources has a mean of 500. Nodes were also identified using a resource shown in the last column of Table 1.

(Table 1) Resources represented by data mining services showing its properties (processing time and tags) and nodes where the resource was deployed.

Data mining	PT (ms)	Tags	Nodes
Apriori	550	association	a, d, e, i, j
kmeans	450	clustering	b, d, f, h, i
J48	300	classifier	a, d, f, j
fuzzyk	350	fuzzy, clustering	b, d, g, i, j
mlp	900	neuralnet, classifier	b, c, f, g, h, i
rbf	600	neuralnet, classifier	a, c, e, g, h, i
neurofuzzy	450	neuralnet, classifier, fuzzy	b, d, e, f, i
som	450	clustering, neuralnet	a, c, e, g,
art	350	classifier, neuralnet	a, c, e, g, h, j
svm	600	neuralnet, classifier	b, c, f, h

We set 10 nodes (*a,..,j*) and divided into two domains. Each node has a single processor and the incoming jobs were queued if the processor was busy. Loads were determined by total time of all waiting jobs in a queue of a node. Separated by domains, a node can access resources on other domains by the logical network of a VO. The 5 nodes were connected to a router with 10 ms of latency and the two routers are connected with 100 ms of latency. Nodes were joined in a VO shown in Table 2. Virtual organizations were labeled in the simulation; VO1=scientists, VO2=engineers, VO3=business and VO4=academy. Each VO agent collects tags from all nodes and these will be provided with scores when a resource on a node associated with a VO tag was used. These scores were then processed in the group similarity function. Table 2 shows the VO tag scores artificially configured and its node members.

(Table 2) Virtual organization's tag scores and node members

VO	classifier	clustering	association	fuzzy	neuralnet	Nodes
Scientist	8	7	4	3	8	a, b, c, d
Engineering	7	6	3	8	8	b, c, d, e
Business	7	9	9	0	2	f, g, h, i
Academy	7	7	6	8	8	h, i, j, a, e

5.2 Experimental Result

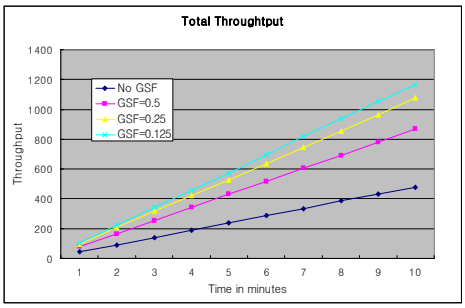
Initially, tag scores were artificially configured in Table 2 according to how much a VO used the services which was assumed in our simulation. For example, business mostly uses association mining and thus it has a higher score in association, but rarely uses fuzzy system which shows a lower score. We determined the initial value of GSF in each VO based on processing the tag scores in Table 2: VO1={2=0.25, 3=0.675, 4=0.23}, VO2={1=0.27, 3=0.671, 4=0.14}, VO3={1=0.43, 2=0.51, 4=0.42}, VO4={1=0.26, 2=0.13, 4=0.63}. We generated resource requests for the simulation and gathered the result using different values of Φ_G ={0.5, 0.25 and 0.125}. Each request contains information and these are source of VO source where a request takes place, resource type and arrival time which are randomly selected.

We also compared the GSF to a similarity measure which uses a Euclidean distance function to the resource count of VOs (σ) and to a method in merging all VOs (Σ) in the point of the average availability of a resource shared in a merged VO and the latency on accessing each resource.

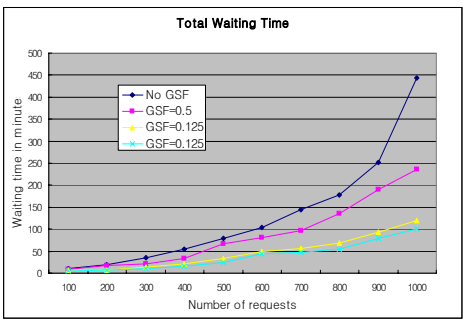
(Table 3) Results of the average availability of a resource and the average latency of accessing the resources using GSF, σ and Σ.

Measures	GSF	Nearest (σ)	All (Σ)
Average availability of resources	2~11	4.2	8.4
Average latency	20~120 ms	33.77 ms	60.642 ms

The first case (σ) was merging the nearest VOs with the least value in the function and the second case (Σ) was merging all VOs to append all resource information. The Σ showed an average of 8.2 resource availability but the latency to access the resource was also higher. The merging of GSF was dynamic where resource availability and latency were dependent on the trend of the frequent access of a resource. A more frequent access on a resource will increase the similarity among VO, but this was balanced by the Φ_j where it unmerged the VO which was not contributing. The availability of resources and latency on accessing the resources were adapting to the trend from the access of users. Figure 4 shows the result of simulating the generated requests.



(A)



(B)

(Fig 4) Total throughput (A) and waiting time (B) of SVO with given threshold values.

In measuring the throughput, we overloaded the VOs with requests. The contribution value for unmerging is also determined after a reply from a complete process was received. In the Figure 4a, throughputs were collected according to a period of time where millisecond was converted to minute. We overloaded the simulation with 10,000 requests and determined the throughputs on a period of time. In GSF=0.5, it merged two VOs and the increase of throughput was also twice. But observing in GSF=0.125, which merged 3 VOs, was less significant than GSF=0.5. The main reason was the nodes in other VO, in choosing GSF=0.125, were already subsets where VO2∪VO3=0, but VO2∪VO1=3 and VO2∪VO3=1. Figure 4b shows the total waiting time according to number of requests. The waiting time also indicates the quality of service (QoS) by faster job processing in grid. We also observed similar performance in throughput from the waiting time.

6. Conclusions and Future Work

Virtual organization (VO) is a fundamental on sharing resources in grid systems. The scalability of a grid system should be efficiently designed to handle the increasing number of grid users. This paper presented a scalable virtual organization for grids to perform scalable sharing of resource among VOs. Our approach tackled the scalable resource sharing using the grouping service. In the process of SVO, the proposed group similarity function (GSF) was used to determine similar VOs to perform merging of resource information. The SVO was initialized by the grouping service which provided the GSF method and then VO agents implemented the SVO where it used the grouping function to request additional resources to other similar VOs. We evaluated the performance of the SVO using our simulator for dynamic virtual organizations. The result of simulation shows that the SVO produces more throughputs by setting a smaller value of GSF which merges more VOs but the throughput increase is less significant if the same nodes in the VO are identified on the merging. The contribution of this paper is the enabling of the scalable resource sharing to the Grid system using our SVO approach.

References

[1] I. Foster, C. Kesselman, S. Tuecke, “The Anatomy of the Grid: Enabling Scalable Virtual Organizations.” International Journal of High Performance Computing Applications, Vol. 15 No. 3 (2001) pp. 200-222

[2] Business Experiment Grid Solution, " http://www. beingrid.eu /be.html"

[3] P. Plaszczak and R. Wellner, "Grid Computing, The Savvy Manager's Guide." Morgan Kaufman (2006) p. 25

[4] Open Grid Service Architecture, http://www.globus.org/ogsa/

[5] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems", Lecture Notes in Computer Science, 3779 (2005) pp. 2-13

[6] R. Alfieri, R. Cecchini, V. Ciaschini, Á. Frohner, A. Gianoli, K. Lőrentey, F. Spataro, "VOMS, an Authorization System for Virtual Organizations."In Proceedings of the 1st European Across Grids Conference (2003)

[7] R.O. Sinnott, D.W. Chadwick, J. Koetsier, O. Otenko, J. Watt and T. A. Nguyen, "Supporting Decentralized, Security Focused Dynamic Virtual Organizations across the Grid, International Conference on e-Science and Grid Computing (2006)

[8] M. Murphy, M. Fenn, and S. Goasguen, "Virtual OIrganization Clusters."Journal of Supercomputing Applications, vol. 15, no. 3 (2001) pp. 200 - 222.

[9] T.J. Norman, A. Preece, S. Chalmers, N.R. Jennings, M. Luck, V.D. Dang, T.D. Nguyen, V. Deora, J. Shao, W.A. Gray, N.J. Fiddian, "Agent-based Formation of Virtual Organisations", Knowledge-Based Systems, Vol. 17, No.2-4 (2004) pp. 103-111

[10] B. Shan, Y. Han, H. Wang, "An Agent-Mediated Service Framework Facilitating Virtual Organizations", Lecture Notes in Computer Science, 4402 (2007) pp.438-446

[11] J J Jiang, D W Conrath, "Semantic Similarity Based on Corpus Statistics and Lexical Ontology", The International Conference of Research on Computer Linguistics, 1997.

[12] R.D. Silva, R. Stasiu, V.M. Orengo, C.A. Heuser, "Measuring Quality of Similarity Functions in Approximate Data Matching", Journal of Informatics, Vol. 1, No. 1 (2007) pp. 35-46

[13] C D Manning, H Schutze, Foundations of Statistical Natural Language Processing, MIT Press, 2001.

[14] P Ganesan, H Garcia-Molina, J Widom, "Exploiting Hierarchical Domain Structure to Compute Similarity", ACM Transaction on Information Systems, Vol 21, No. 1, pp. 64-93.

[15] A Torsello, D Hidovic, M Pelillo, "Four Metrics for Efficiently Comparing Attributed Trees," The 17th International Conference of Pattern Recognition, Vol. 2, pp. 467-470, 2004

[16] K F Aoki, A Yamaguchi, Y Okuno et al, "Efficient Tree-Matching Methods for Accurate Carbohydrate Database Queries", Genome Informatics, Vol. 14, pp. 134-143, 2003.

[17] J Ontrup, T Nattkemper, O Gerstung et al, "A MeSH Term Based Distance Measure for Document Retrieval and Labeling Assistance", The 25th International Conference of IEEE English in Medical and Biological Societies, 2003.

[18] Bin Lu, Juan Chen, Uniform Similarity Measures in Grid Resource Discovery, Proc. of 7th International Conference on Machine Learning and Cybernetics, (2008) pp. 2678-2682

[19] J. Bourgeois, J.B. Ernst-Desmulier, F. Spies and J. Verbeke, "Using Similarity Groups to Increase Performance of P2P Computing," Springer-Verlag LNCS 3149, (2004)pp.1056-1059

[20] Simulator for Virtual Organization, http://dslab.kunsan.ac.kr/en/simvo/index.html

◐ 저 자 소 개 ◑

마테오 로미오(Romeo Mark A. Mateo)

2004 West Visayas State University, Philippines
BS in Information Technology
2007 Kunsan National University, South Korea
Master of Engineering major in Information and Telecommunications
2007 ~ current Kunsan National University, South Korea
Graduate student in Ph.D course
Research interest : Distributed systems, data mining, fuzzy systems, multi-agents, ubiquitous sensor networks, cloud computing
E-mail : rmmateo@kunsan.ac.kr

이 재 완(Jaewan Lee)

1984년 중앙대학교 이학사-전자계산학
1987년 중앙대학교 이학석사-전자계산학
1992년 중앙대학교 공학박사-전자계산학
1996년 3월~ 1998년 1월 한국학술진흥재단 전문위원
1992 ~ 현재 군산대학교 교수
관심분야 : 분산 시스템, 운영체제, 실시간 시스템, 컴퓨터 네트워크 등
E-mail: jwlee@kunsan.ac.kr

이 말 례(Malrey Lee)

1998 년 중앙대학교 컴퓨터공학과 박사
1999~2003: 전남대학교 멀티미디어학과조교수
2003~현재: 전북대학교 전자정보공학부 부교수
관심분야 : 인공지능, 로보틱스, 컴퓨터게임,
멀티미디어, 유비쿼터스 컴퓨팅, 헬스케어응용 등
E-mail: mrlee@chonbuk.ac.kr