# Construction and Analysis of a Semantic Grid Service for Large-scale Environment

Toshihiro Uchibayashi†, Bernady O. Apduhan ‡, Norio Shiratori*

*Abstract*— **Building an ontology resource network using grid computing is one method to exploit the sparsely distributed ontology resources. Grid computing technology provides merit on security issues and ease of controlling the resources. However, problems on resource discovery and the effectiveness in managing and provisioning the various scattered grid resources exist. We addressed these problems with the construction of a semantic grid service which automatically provides the optimal required grid resources, and the adaption of the notion of Virtual Organization (VO). The semantic grid service consists of various VOs, and the sub-ontology extraction and tailoring application was used as a proof-of-concept. The processing of the application was analyzed in order to verify the workability of the system.**

*Index Terms*—**Semantic Grid, Sub-ontology Tailoring, Grid Application**

## I. Introduction

One method to exploit the sparsely distributed ontology resources includes the construction of a grid computing environment using these ontology resources. Grid computing enables to exploit systematically these distributed resources and to increase the processing throughput. Grid computing provides improvement in security and control of resources.

However, the problem of using a grid network and the problem of exploiting ontology resources exists. The first problem pertains to the resource discovery problem [1,2,3]. With the vast resources on the grid network, it becomes fairly difficult to grasp the required resources. The notion of Virtual Organization (VO) which bundles two or more resources (real or virtual) defines a group and is considered a viable approach. However, to manage the ontology resources using only the indexing service of the Globus Toolkit in the VO is insufficient. In grid computing, the scheduler describes the job for each node. If the information or composition about the nodes to be used is insufficient, it will become difficult to perform the required job and produce the expected results. Therefore, a means to provide detailed information about the nodes is sought.

The second problem pertains to the scattered and remote nodes on a grid network. If an application requires a collection of these nodes, which are also performing other processing at the same time, will result in slow or decrease performance and/or throughput. For one sub-ontology search, what can be done is to search for the required node, conduct the processing locally and then transmit the result to the user. However, when two or more required nodes are involved and connecting them altogether will incur large delay contributing to performance degradation.

† ‡ Faculty of Information Science, Kyushu Sangyo University, Fukuoka, Japan
* Research Institute of Electrical Communication, Tohoku University, Sendai, Japan

In this paper, we describe the construction of the semantic grid service in which a server automatically selects the optimal nodes even if the user did not fully describe the required nodes to the scheduler. The grid application using sub-ontology tailoring technology in extracting the required and relevant ontology information was developed.

We realized the above by building a semantic grid service which consists of two VOs and executed onto a grid application as a proof-of-concept.

In the following, section II describe some related work, while section III shows compositions and example of semantic grid service. Section IV describe the constructed environment with VOs, and analyze the constructed environment using sub-ontology extraction/tailoring application service in section V. Section VI describe our concluding remarks and future work.

## II. Related Works

Previous work on semantic based resource discovery and matching, i.e., A New Grid Resource Discovery Framework [4] proposed a novel semantic-based scalable decentralized grid resource discovery framework. It describes three related technologies, namely: JXTA, P2P, Ontology and Intelligent Agent. We adopted the same technologies in our service discovery. These technologies were further applied for resource discovery for large-scale semantic grid environment.

## III. Semantic Grid Service

### A. Grid Service

In this paper, a service which access the node on a grid network from a web site application, and enable it to perform jobs, such as file transmission and calculation processing, is defined as a grid service.

A user accesses a web application through a web browser, and can use the grid easily because beforehand, an administrator or a user defines the processing using the grid network.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

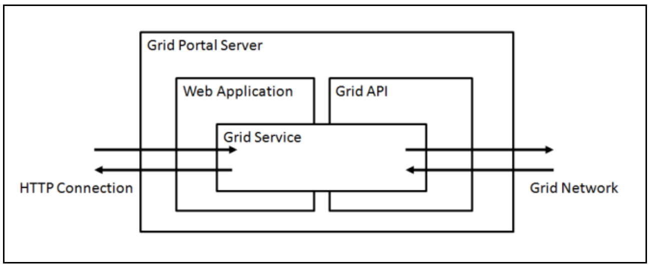

Figure 1 Composition of a typical Grid Service

Figure 1 shows the composition of the grid service on a grid portal server. Grid service contains the Web site application and the grid API on a grid portal server, and is the service which unites the grid with the web application. By registering the framework of a job into a Web site application beforehand, the processing which uses a web application and grid API can be performed. The grid service can be used through a web browser, and processing can be performed, without the user creating a new job, if it is the same processing.

### B. Semantic Grid Service

Generally, when using a grid, a user performs the scheduling and then uses the grid. However, it is difficult for the user to grasp which node can perform and what type of processing, the node information to perform in framing up the execution of process scheduling. In a semantic grid, the user should not be concerned or bothered about the job and resources scheduling.

In order to solve this problem, the grid has to perform the search, sorting, and the selection of nodes for every application processing, and provide this as a web service application. The detailed information of each node is recorded in the Resource Discovery Ontology and the selection process of nodes is done automatically.

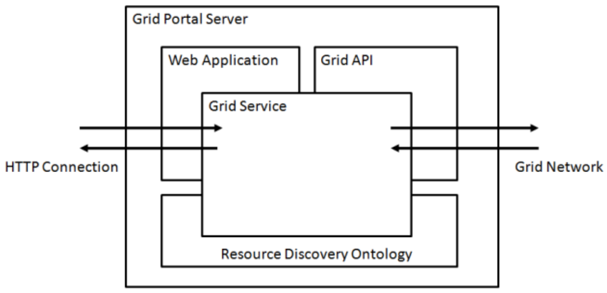

Figure 2 Composition of a Semantic Grid Service

Figure 2 shows the composition of a semantic grid service. The Resource Discovery Ontology for searching the node on a grid network is seen as a grid service from the grid portal server. This makes it possible to determine automatically which node is used by the server side by describing the processing which uses this ontology as a web application. Unlike the usual grid, it can be used without specifying a specific job and the user can process the application without specifying which nodes to use.

## IV. Virtual Organization with Virtual Machines

In this chapter, we explain an example of composing a semantic grid with virtual machines, as well as the construction of a semantic grid service.

### A. Semantic Grid Service Organization

Here, the difference between the conventional grid and Semantic Grid services is explained, and the advantage of Semantic Grid service is introduced.

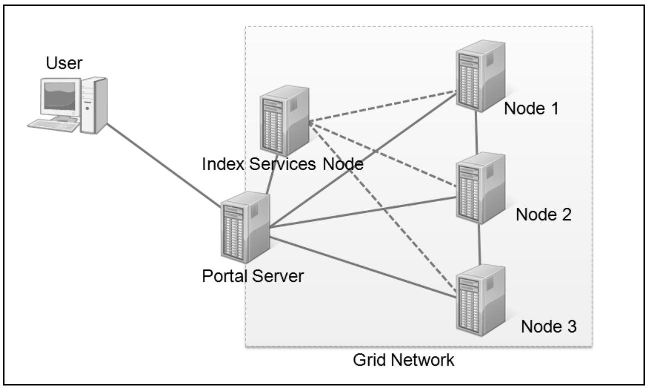

Figure 3 Example of Grid Using Index Service

Figure 3 shows an example of a grid which acquires the nodes information via the Index Service. The Index Service Node collects the static and dynamic information of the nodes on a grid network, and kept it as an index. By using the index through the Portal Server, the user can perform flexible processing. However, the user should decide beforehand which node to use based on the description of nodes' services in the Index Service Node. This procedure is very inefficient and the need to introduce a system arises, i.e., the Resource Discovery Ontology, which determines a node in a completely dynamic fashion.

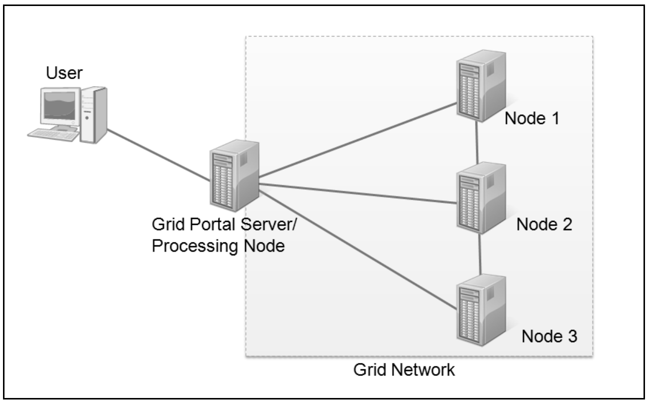

Figure 4 Example of Semantic Grid Service Composition

Figure 4 shows a composition example of a semantic grid. Our proposed semantic grid environment consists of a Grid Portal Server/ Processing Node and a number of nodes instead of Index Service Node and Portal Server. The Semantic grid service has been moved to the Grid Portal Server, and the Resource Discovery Ontology keeps statically the information of each node in more detail than the Index Services. The optimal node for processing can be extracted by performing guess search which is peculiar to ontology. Moreover, in order to perform local processing, the Grid Portal Server is also recognized as a Processing Node in the

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

grid network. The node in a semantic grid needs to be registered into the Resource Discovery Ontology of the Grid Portal Server. Unregistered nodes cannot be used by the semantic grid. This collection of the nodes centered on the Grid Portal Server functions as one group.

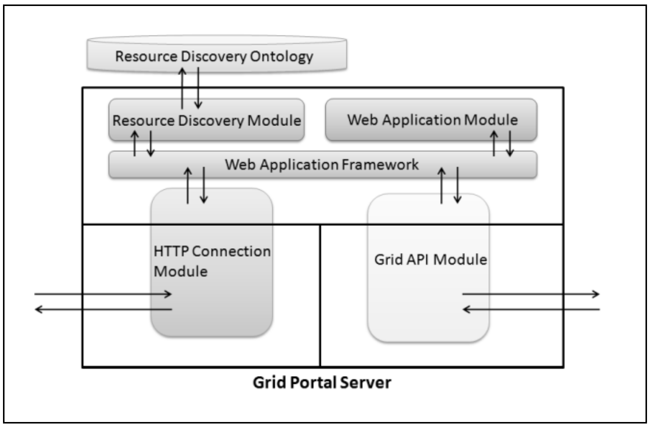


Figure 5 Structure of Grid Portal Server

Figure 5 shows the structure of a Grid Portal Server. It consists of HTTP Connection Module, Grid API Module, Web Application Framework and Web Application Module, and Resource Discovery Module. A user accesses to Grid Portal Server through HTTP Connection Module. Web Application Framework has generalized the whole processing and uses each module from this framework. The algorithm and node selection processing which are used to search for the required nodes is performed by the Resource Discovery and Resource Discovery Ontology modules. The application which will be executed on the semantic grid is registered into the Web Application Module, and the user uses this registered application. Grid API Module comprises the various APIs of the grid middleware which constitutes the grid network. To connect to the grid network, it is necessary to make the connection through these modules. The composition of Resource Discovery Ontology is shown below, Figure 6.

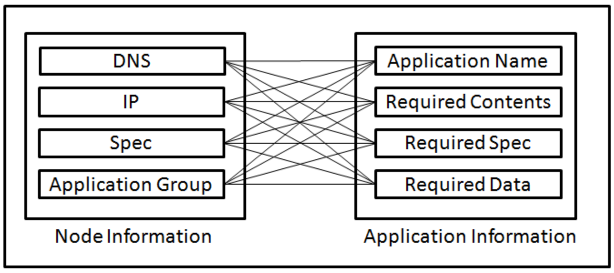


Figure 6 Composition of the Resource Discovery Ontology

The data in the Resource Discovery Ontology is mainly divided into two, i.e., the Node Information and Application Information. These pieces of information are hierarchical, and are recorded including its relevance. For example, even when the application of a web application module gives only the application name to search for available nodes, the guess search is performed, then the nodes are sorted out and the IPs of the optimal nodes comes out as a result. Thus, a service which can choose a node automatically can be built by using ontology instead of the Index Service.

*B. Environment Construction*

Here, we introduce the construction of the semantic grid service, as well as the mounting of a grid application.

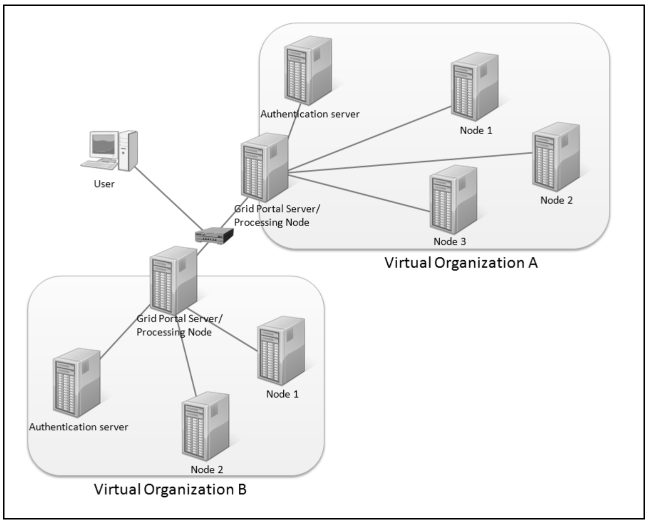


Figure 7 The structure of a semantic grid service

Figure 7 shows the constructed semantic grid service. It consists of two semantic grid groups. A group is called a VO (Virtual Organization), and hereon called as VO-A and VO-B in this environment. Grid Portal Server of each VO is cooperative, i.e., when a user uses VO-A and the required node is in VO-B, the node in VO-B is chosen automatically, and job is processed in VO-B.

Sub-ontology extraction/tailoring application was mounted on this semantic grid service. Globus Toolkit 4 and Grid Portal Server use the Tomcat6 for the grid middleware. Moreover, one VO was built in one server, creating Virtual Machines (VMs) and assigning each VM as a node. The simulation of a large-scale semantic grid can be performed using the ideas of VO.

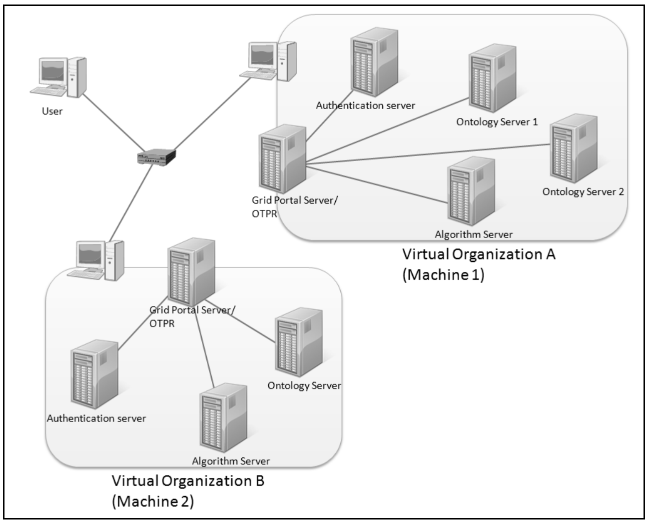


Figure 8 Mounting of sub-ontology extraction/tailoring application

Figure 8 shows the environment in executing the sub-ontology extraction/tailoring application. In sub-ontology extraction/tailoring, the OTPR (Ontology Tailoring Processing Resource), an ontology server, and an algorithm server are required. The OTPR performs the job management on the grid, assigning nodes, and also performs the ontology tailoring processing. In this environment, the Grid Portal Server performs the role of OTPR. Ontology data is stored in the ontology server and it is kept in data file form. The tagged ontology server transmits the ontology data to the OTPR which then extracts the sub-ontology and performs ontology tailoring. The scheme used in the sub-ontology extraction is kept at the algorithm server. It is assumed that different ontologies are kept by each ontology server, and the ontology tailoring is performed after bringing altogether in OTPR the required ontology/sub-ontology data from the participating servers.

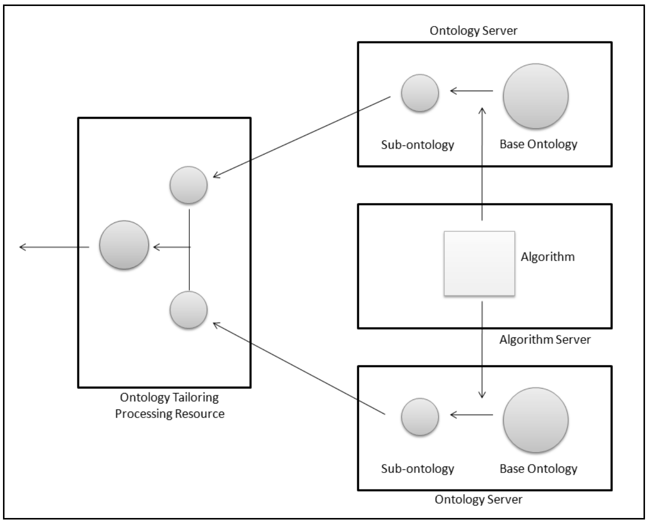


Figure 9 Flow of sub-ontology tailoring processing

Figure 9 shows the flow of the tailoring process on a semantic grid. Sub-ontology is extracted by each ontology server using the algorithm from the algorithm server. The sub-ontology extracted by each ontology server is brought together to OTPR, and tailoring is performed. Although Fig. 9 has only shows an example of tailoring of two ontologies, more number of ontologies can participate (as required), and the method in Figure 9 can be repeated.

## V. Analysis of the Processing Environment

In this section, we conducted the sub-ontology extraction/tailoring application mentioned in Section IV, and analyzed the constructed semantic grid. The relevant ontology with respect to VO-A and VO-B were arranged, conducted the sub-ontology processing, and investigate whether the sub-ontology is extracted. First, the UMLSSN-A ontology is put onto the ontology server in VO-A, and UMLSSN-B ontology is put onto the ontology server in VO-B (Figure 9). The Unified Medical Language System Semantic Network (UMLSSN) is a medical ontology that approaches to facilitate the development of computer systems that behave as if they "understand" the meaning of the language of biomedicine and health [8]. A user accesses to the Grid Portal Server in VO-B. Sub-ontology is extracted from the base ontology, and sends it to the Grid Portal Server in which the user is connected, and displayed on the screen.

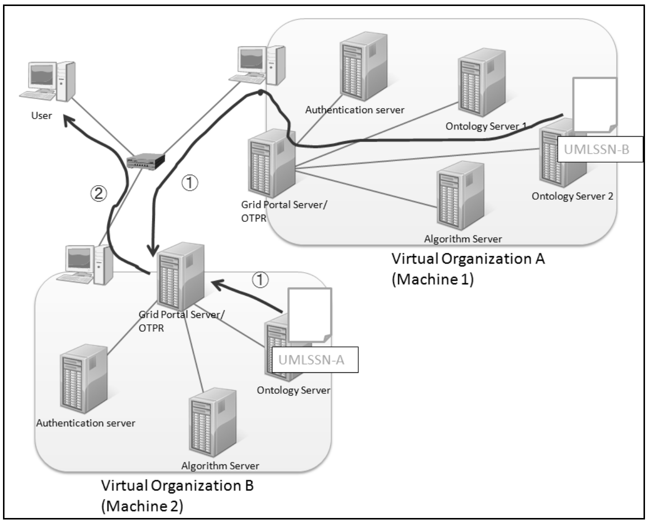


Figure 10 Arrangement and Flow of Ontologies

Figure 11 shows the ontology concept tree of UMLSSN-A onto the ontology server in VO-B, has 21 concepts and 65 data properties, with 4 concepts labeled as "selected (necessary)" and 2 concepts as "deselected (unnecessary)". As shown in Figure 12, UMLSSN-B has 27 concepts and 65 data properties, with 5 concepts labeled as "selected" and 2 concepts as "deselected". UMLSSN-A and UMLSSN-B (as we call it) are portions of the UMLSSN ontology. The sub-ontology tailoring processing is conducted in the OTPR in VO-B using the extracted sub-ontology data from ontology server. In the sub-ontology extraction, the Minimum method was used, as well as the CnV and CmS optimization schemes [5] storage in the algorithm server in the same VO location.

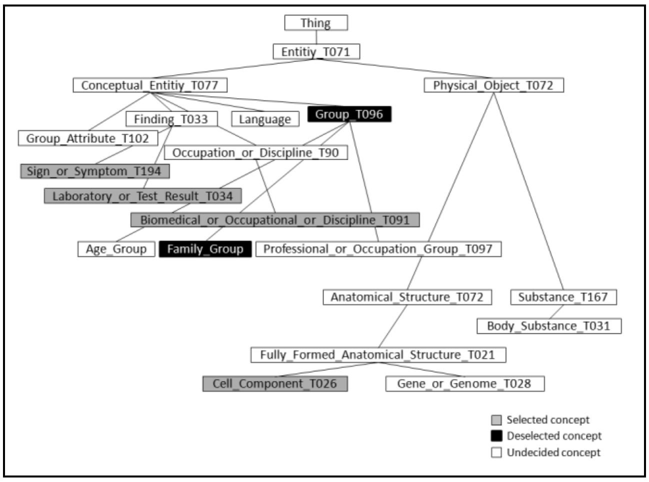


Figure 11 Ontology Elements of UMLSSN-A

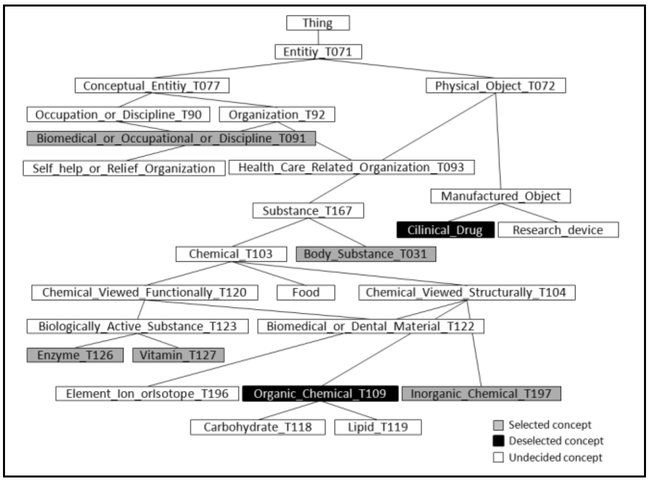


Figure 12 Ontology Elements of UMLSSN-B

The first step in sub-ontology tailoring is sub-ontology extraction from the ontology servers. Sub-UMLSSN-A, shown in Figure 13, is the sub-ontology of UMLSSN-A. Sub-UMLSSN-A has 14 concepts and 65 data properties, having 7 concepts less than its base ontology concepts. While Sub-UMLSSN-B, in Figure 14, has 11 concepts less than its base ontology concepts.

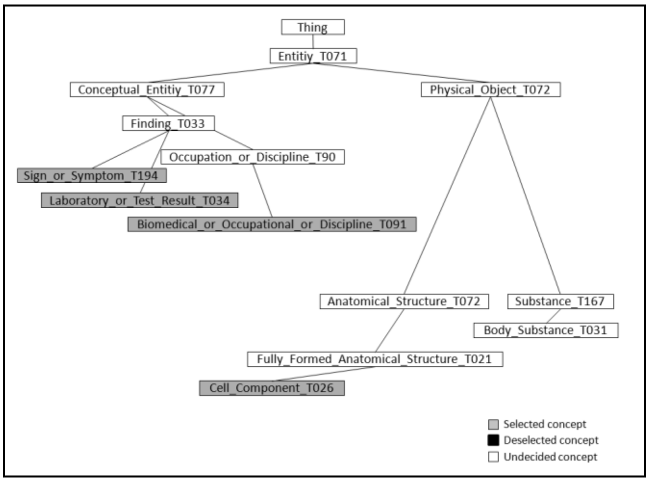


Figure 13 Sub-UMLSSN-A Ontology Elements

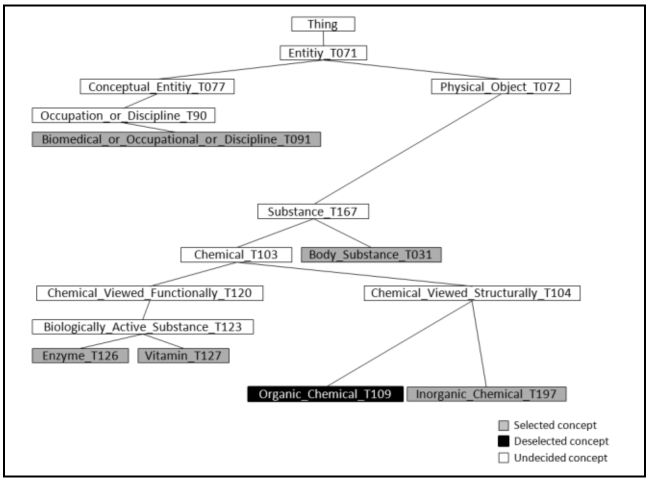


Figure 14 Sub-UMLSSN-B Ontology Elements

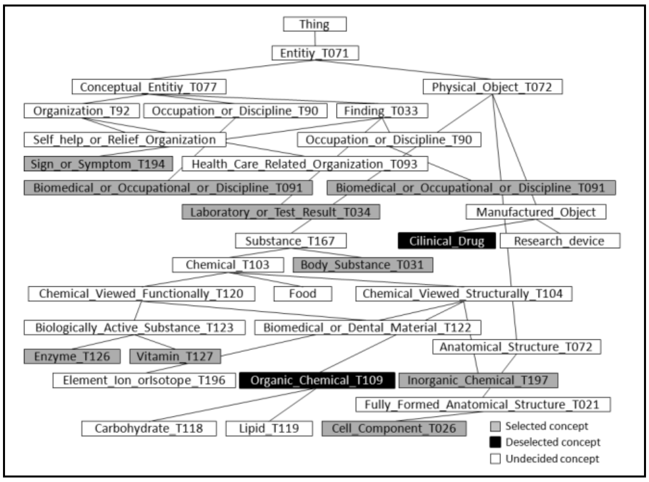


Figure 15 Sub-ontology Elements (add)

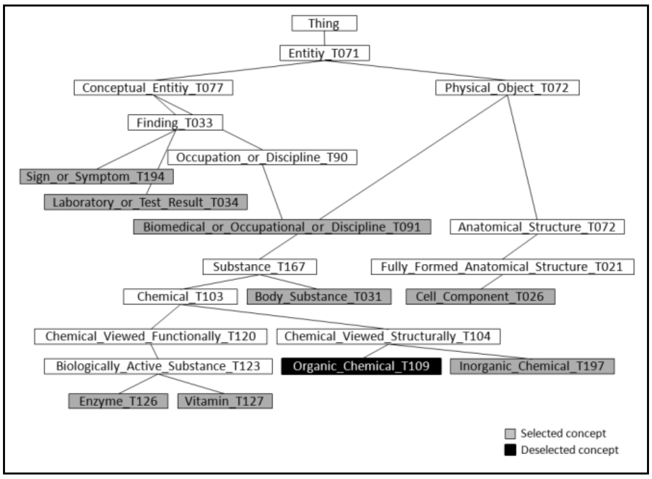


Figure 16 Sub-ontology Elements (merge)

The second step is to send the extracted sub-ontologies, i.e., Sub-UMLSSN-A and Sub-UMLSSN-B, to the OTPR in VO-B.

In the OTPR, the final step, i.e., the tailoring of extracted sub-ontologies is conducted. Figure 15 shows the tailored sub-ontology that utilizes the *Reuse, Extract, and Add* method [9]. That is, adding the extracted sub-ontology Sub-UMLSSN-A to ontology UMLSSN-B. The tailored (resulting) sub-ontology has 35 concepts and 65 data properties. Figure 16 shows the tailored sub-ontology that utilizes the *Reuse, Extract, and Merge* method. That is, adding the extracted sub-ontologies Sub-UMLSSN-A and Sub-UMLSSN-B. The tailored sub-ontology has 22 concepts and 65 data properties. These procedures exemplify some sub-ontology tailoring process.

These ontologies were extracted from one part of UMLSSN ontology and are many relevant with other ontologies Table 1 shows the data sizes of Sub-UMLSSN-A (76.1 KB), Sub-UMLSSN-B (72.8 KB), UMLSSN-A (114 KB), and UMLSSN-B (104 KB).

In using the *Reuse, Extract, and Add* tailoring method (Sub-UMLSSN-1 + UMLSSN-2), the resulting sub-ontology is 133KB. That is, the data size was reduced to about 61% compared to adding the two base ontology (UMLSSN-A + UMLSSN-B), and without adapting tailoring method. The consideration of using only the selected and undecided concepts and the tailoring method contributed to the reduced data size.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Likewise, in using the *Reuse, Extract, and Merge* tailoring method (Sub-UMLSSN-A + Sub-UMLSSN-B), the tailoring resulted to 80.3KB. That is about 54% compared to flatly merging two sub-ontologies. In the same manner, the consideration of using only the selected and undecided concepts and the adapted tailoring method contributed to the reduced data size.

Table 1. Data Size of Ontologies

| | |
|---|---|
| UMLSSN-A | 114KB |
| UMLSSN-B | 104KB |
| Sub-UMLSSN-A | 76.1KB |
| Sub-UMLSSN-B | 72.8KB |
| Sub-Ontology (using Add) | 133KB |
| Sub-Ontology (using Merge) | 80.3KB |

Sub-ontology tailoring processing which uses such ontology was performed 20 times on the same conditions. As a result, the sub-ontology is optimized without losing the relevance of the two ontologies. This implies that processing between VO(s) on a semantic grid was performed satisfactorily.

## VI. Concluding Remarks

We constructed the semantic grid service and its prototype environment with Virtual Organizations (VOs), and verified its workability by way of executing the sub-ontology extraction/tailoring application as a proof-of-concept. In the process, the sub-ontology extraction and sub-ontology tailoring were performed with different ontology data from different VO nodes. This demonstrates that a semantic grid service for large-scale environment can be realized with the creation and coordination with multiple VOs.

Future work includes experiments with more number of VOs with different processing capabilities, as well as thorough study and development of an automatic selection mechanism for optimal nodes.

## Acknowledgment

This work was supported in part by the Japan Society for the Promotion of Science Grants-in-Aid for Scientific Research, 21500087.

## References

[1] S.Jeffrey, J.Hunter, A Semantic Search Engine for the Storage Resource Broker, 3rd Semantic Grid Workshop, GGF16, Athens, Feb 15, 2006.

[2] Fouad Butt, Syed Saadat Bokhari, Abdolrezza Abhari, Alexander Ferworn, Scalable Grid Resource Discovery through Distributed Search, Signal & Image Processing, vol.2 Issue 3, pp1-19, 2011

[3] Hassan, M. I. and Azween, Abdullah,A Semantic Service Discovery Framework for Intergrid, 2nd International Conference on Software Engineering and Computer Systems, ICSECS 2011, pp27-29, 2011.

[4] Mahamat Issa Hassan, Azween Abdullah, A new grid resource discovery framework, The International Arab Journal of Information Technology, Vol.6, No.1, pp13-20 (2009).

[5] Toshihiro Uchibayashi, Bernady O. Apduhan, Wenny J. Rahayu, David Taniar, Norio Shiratori, On a Prototype Environment for Sub-ontology Extraction in Semantic Grid, The 10th International Conference on Computational Science and Its Applications (ICCSA 2010), pp.347-350, 2010.

[6] Toshihiro Uchibayashi, Bernady O. Apduhan, Wenny J. Rahayu, Norio Shiratori, A Grid Application Service Framework for Extracted Sub-ontology Update, Proc. International Symposium on Emerging Research Projects, Applications and Services (ERPAS/IiWAS 2010), 2010.

[7] Toshihiro Uchibayashi, Bernady O. Apduhan, Norio Shiratori, Experiments and Functional Analysis in Integrating Sub-ontology Extraction and Tailoring, Proceedings of the 2011 International Conference on Computational Science and Its Applications, pp143-149, 2011.

[8] The UMLS, Semantic Network, http://semanticnetwork.nlm.nih.gov/

[9] M. Bhatt, A. Flahive, C. Wouters, J.W. Rahayu and D. Taniar, MOVE: A Distributed Framework for Materialized Ontology View Extraction, Algorithmica 45(3), pp. 457-481, 2006.