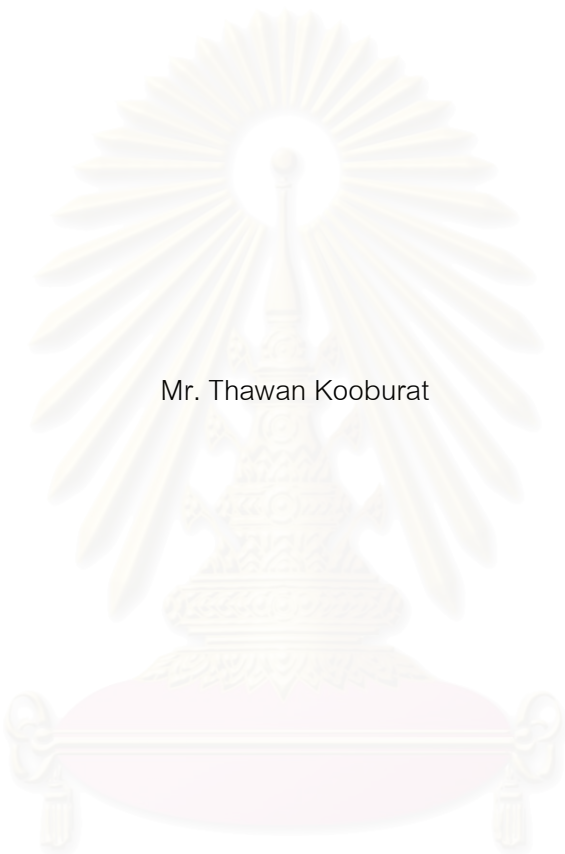


A GRID HOSTING SYSTEM FOR MULTIPLE VIRTUAL ORGANIZATIONS



Mr. Thawan Kooburat

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Computer Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2007

Copyright of Chulalongkorn University

ระบบการให้บริการกริดสำหรับหลายองค์กรเสมือน



นาย ธาวัน คูบุรีตถ์

สถาบันวิทยบริการ

จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2550

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

ธำวัน คูบุรีรัตน์ : ระบบการให้บริการกริดสำหรับหลายองค์กรเสมือน. (A GRID HOSTING SYSTEM FOR MULTIPLE VIRTUAL ORGANIZATIONS) อ. ที่ปรึกษา : อ. ดร. วีระ เหมืองสิน, 112 หน้า.

องค์กรเสมือน (Virtual Organization) เป็นแนวคิดที่เกิดขึ้นจากลักษณะการทำงานบนกริด ซึ่งเป็นรูปแบบขององค์กรที่เกิดจากการรวบรวมทรัพยากร เช่น บุคคลากร ข้อมูล และหน่วยประมวลผล จากหลากหลายหน่วยงานเข้ามาทำงานร่วมกัน ด้วยเหตุนี้องค์กรเสมือนจึงอาจมีการเปลี่ยนแปลงอยู่ตลอดเวลา และอาจจะมีทรัพยากรคาบเกี่ยวกันได้ ด้วยเทคโนโลยีกริดในปัจจุบัน การสร้างและใช้งานองค์กรเสมือนยังทำได้ลำบาก เพราะผู้ดูแลระบบไม่สามารถบริหารและจัดการองค์กรเสมือนที่เปลี่ยนแปลงอยู่ตลอดเวลาได้อย่างง่ายดาย รวมทั้งผู้ใช้อย่างมีความยุ่งยากหากต้องทำงานร่วมกับหลายองค์กรเสมือน

เพื่อทำการแก้ปัญหาเหล่านี้ งานวิทยานิพนธ์ชิ้นนี้จึงได้เสนอแนวคิดของการให้บริการกริดผ่านรูปแบบขององค์กรเสมือน และได้นำเสนอระบบการให้บริการกริด หรือมีชื่อเรียกว่าโกลด์ทั้น (Grid HOSTing System – GHOSTS) ระบบดังกล่าวใช้แนวคิดขององค์กรเสมือนอย่างง่ายในการให้บริการ และสามารถให้บริการหลายองค์กรเสมือนได้พร้อมๆกัน ซึ่งองค์กรเสมือนเหล่านี้อาจมีทรัพยากรที่คาบเกี่ยวกัน หรือมีความเปลี่ยนแปลงอยู่ตลอดเวลาด้วยก็ได้ โครงสร้างของระบบใช้การจัดการผ่านศูนย์กลางโดยการรวบรวมความสามารถในการบริหารงานองค์กรเสมือนและกริดพอร์ทัลแบบพอร์ทัลเล็ต (Portlet) เข้าไว้ด้วยกัน ด้วยเหตุนี้ระบบจึงสามารถซ่อนความซับซ้อนของการบริหารและการใช้งานองค์กรเสมือนไว้ได้ และให้สภาพแวดล้อมที่ผู้ใช้สามารถใช้งานกริดได้โดยง่าย

ภาควิชา วิศวกรรมคอมพิวเตอร์..... ลายมือชื่อนิสิต..... ๑๑๕๕ คูบุรีรัตน์
 สาขาวิชา วิศวกรรมคอมพิวเตอร์..... ลายมือชื่ออาจารย์ที่ปรึกษา..... วีระ เหมืองสิน
 ปีการศึกษา2550.....

4970355221 : MAJOR COMPUTER ENGINEERING

KEY WORD: GRID PORTAL / GRID HOSTING / VIRTUAL ORGANIZATION

THAWAN KOOBURAT : A GRID HOSTING SYSTEM FOR MULTIPLE
VIRTUAL ORGANIZATIONS. THESIS ADVISOR : VEERA MUANGSIN,
Ph.D., 112 pp.

Virtual Organizations (VOs) are dynamic and possibly overlapping groups of resources and users in the Grid. With the current development of Grid technologies, it is still a major challenge to build and run virtual organizations. Firstly, from an administrative perspective, it is difficult to manage dynamic and overlapping virtual organizations. Secondly, for users it is cumbersome to work in multiple virtual organizations.

To address these issues, this thesis proposes a concept of providing a VO hosting service and presents a software package called GHOSTS (Grid HOSTing System). The system employs a simple Virtual Organization model and it can host multiple VOs with dynamic and overlapping sets of resources and users. It is based on a centralized server that integrates VO management and Portlet-based Grid portals. This solution can hide complexity of VO management and also provides an easy-to-use working environment for using Grid resources.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Department ...Computer Engineering.... Student's signature.....
Field of study ...Computer Engineering.... Advisor's signature.....
Academic year2007.....

ACKNOWLEDGEMENTS

I would like to express my gratitude to all those who gave me the possibility to complete this thesis, especially, my advisor, Dr. Veera Muangsin, who gave me insightful advise and continuous support throughout my graduate study. I also appreciate constructive comments from all my thesis committee, Professor Prabhas Chongstitwatana, Ph.D., Assistance Professor Putchong Uthayopas, Ph.D., and Natawut Nupairoj, Ph.D.

Additionally, I would like to extend my personal thanks to all my friends within the Department of Computer Engineering, especially members of Scientific Parallel Computer Engineering Laboratory (SPACE Lab) who accompany me during the course of this work.

Finally, I would like to thank to my mother for their kindly support and love which make me accomplish this task.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CONTENTS

Page

ABSTRACT (THAI)	iv
ABSTRACT (ENGLISH)	v
ACKNOWLEDGEMENTS	vi
CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xi
CHAPTER I INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	2
1.2.1 VO Models.....	2
1.2.2 Enforcing VO Usage Policy.....	3
1.2.3 VO Usability	3
1.3 Objectives.....	4
1.4 Proposed Solution	4
1.5 Scope of Study	5
1.6 Research Process	5
1.7 Expected Benefits.....	6
1.8 Thesis Structure.....	6
1.9 Publications	6
CHAPTER II BACKGROUND THEORY AND RELATED WORK	8
2.1 Background Theory	8
2.1.1 Grid Computing	8
2.1.2 Virtual Organization (VO).....	10
2.1.3 Grid Architecture	11
2.1.4 Grid Security.....	14
2.1.5 Grid Portal.....	18
2.2 Related Work.....	22
2.2.1 VO Authorization.....	22
2.2.2 Virtual Accounts and Account Pool	23
2.2.3 Grid Portal Solutions.....	24
CHAPTER III GRID HOSTING MODEL	26
3.1 Overview	26
3.2 VO Model.....	26
3.3 Enabling Mechanisms	28
3.3.1 System Architecture.....	28
3.3.2 User Management	29

3.3.3	Credential Management	30
3.3.4	Access Control Mechanism	31
3.3.5	Resource Management	35
3.3.6	VO-aware Grid Portal	35
CHAPTER IV DESIGN AND IMPLEMENTATION		37
4.1	Grid Portal	37
4.1.1	GridSphere	37
4.1.2	GridPortlets	38
4.1.3	MyProxy	38
4.2	VO Hosting	39
4.2.1	VO Database	39
4.2.2	GHOST Portlets	42
4.2.3	GHOSTS Services	54
4.2.4	GHOSTS Client	57
4.3	Applications	59
4.3.1	General Purpose Job Profiles	60
4.3.2	Showcase Job Profiles.....	60
4.3.3	Scientific Applications Job Profiles.....	62
4.4	Additional Software	63
4.4.1	Google Analytics	64
4.4.2	Ganglia Monitoring System.....	64
CHAPTER V EVALUATION		65
5.1	System Features.....	65
5.2	Feature Comparison	66
5.3	Security Considerations.....	67
5.3.1	Server Security	67
5.3.2	Client Program Security.....	67
5.4	Reliability and Scalability Considerations	67
5.5	Usage Scenarios	68
5.5.1	ThaiGrid VO Creation	68
5.5.2	Department VO Deployment	69
5.6	Case Studies	69
5.6.1	ThaiGrid Integration	69
5.6.2	Class Room Teaching	71
CHAPTER VI CONCLUSION		75
6.1	Summary	75
6.2	Future Works.....	76

6.2.1	VO Authorization Integration	76
6.2.2	Web-service Components Compatibility	76
REFERENCES.....		77
APPENDICES		80
APPENDIX A GHOSTS USER MANUAL.....		81
A.1.	Introduction	81
A.2.	Architecture	81
A.3.	The Grid Portal.....	82
A.4.	Getting Started.....	82
A.5.	Activating a VO	83
A.6.	Overview of the Portal	84
A.7.	File Management.....	85
A.8.	Job Submission.....	87
A.8.1.	Shell Script Job Profile	89
A.8.2.	Java Application Job Profile	90
APPENDIX B USECASE SCENARIOS		92
B.1.	User Entry	92
B.2.	VO Creation	93
B.3.	Resource Admission.....	93
B.4.	Resource Membership.....	94
APPENDIX C EVALUATION RESULT.....		95
C.1.	Google Analytics Result.....	95
C.1.1.	Overall Statistic.....	95
C.1.2.	Visitors Overview	96
C.1.3.	Top Content	97
C.2.	Mandelbrot Picture Contest Result	98
BIOGRAPHY		99

LIST OF TABLES

	Page
Table 1: Mapping scheme benefits comparison.....	35
Table 2: Summary of GHOSTS portlets' function	42
Table 3: GHOSTS Portlets' access right matrix	43
Table 4: Features summary.....	65
Table 5: Feature comparison.....	66
Table 6: Network capacity of each site.....	70
Table 7: Activity statistic summary	73



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

LIST OF FIGURES

	Page
Figure 1: Grid Computing.....	9
Figure 2: Example application of Access Grid	10
Figure 3: Hourglass model.....	11
Figure 4: Globus Components	12
Figure 5: The UNICORE Architecture	13
Figure 6: OGSA capabilities.....	14
Figure 7: Secure communication using the public key cryptography.....	15
Figure 8: Digital signature using the public key cryptography ⁶	15
Figure 9: Digital certificate	16
Figure 10: Proxy certificate	17
Figure 11: Credential delegation in the job submission process	17
Figure 12: Example “ <i>grid-mapfile</i> ” entries.....	18
Figure 13: Portlet-based web portal.....	19
Figure 14: GridSphere architecture.....	20
Figure 15: Screenshots from GridPortlets.....	21
Figure 16: GridSphere/GridPortlets deployment.....	21
Figure 17: LCG networks	27
Figure 18: GHOSTS system diagram	29
Figure 19: Role relationships in GHOSTS	30
Figure 20: Automatic credential management.....	31
Figure 21: Mapping VO sub-groups structure to flat VO model.....	32
Figure 22: One-to-one account mapping	33
Figure 23: Account pool mapping	34
Figure 24: Shared account mapping	34
Figure 25: Software packages.....	37
Figure 26: First page of the Grid portal	38
Figure 27: VO structure class diagram	39
Figure 28: Request Class Diagram	41
Figure 29: Account Request Portlet.....	43
Figure 30: VO Activation Portlet.....	44
Figure 31: VO Activation Portlet.....	45

Figure 32: Joining normal VO	45
Figure 33: Certificate upload page.....	45
Figure 34: Accounting Portlet.....	46
Figure 35: VO Administrator Portlet	47
Figure 36: VO creation request page	47
Figure 37: Resource Management Portlet.....	48
Figure 38: VO Administrative Portlet.....	48
Figure 39: VO user management	49
Figure 40: Batch user addition feature	49
Figure 41: VO resource management	50
Figure 42: VO job profile select page.....	50
Figure 43: VO portal layout edit page	51
Figure 44: Host select page.....	52
Figure 45: Resource Administrative Portlet	52
Figure 46: Assigning a resource administrator to a resource.....	53
Figure 47: User Administrative Portlet.....	53
Figure 48: Request Administrative Portlet	54
Figure 49: Processing VO structure information	57
Figure 50: Format of “ <i>vomap-file</i> ”	58
Figure 51: Example “ <i>vomap-file</i> ” entries.....	58
Figure 52: Job profile sequence	60
Figure 53: Mandelbrot Job Profile	61
Figure 54: Selecting input file of Scilab	62
Figure 55: Input file is presented before submitting a job	63
Figure 56: Google Analytics’ report	64
Figure 57: Ganglia Monitoring System	64
Figure 58: VO resources diagram	70
Figure 59: Data transfer result	71
Figure 60: Usage statistic charts	73
Figure 61: Google Analytics’ graph report.....	74
Figure 62: GHOSTS system diagram	81

Figure 63: Multiple portlet displayed on the same page.....	82
Figure 64: GHOSTS front page	82
Figure 65: The Registration Portlet.....	83
Figure 66: The VO Activation Portlet	83
Figure 67: Use two file browser components to transfer file between hosts	85
Figure 68: Select a host and use the <i>List>></i> button to browse its files.....	85
Figure 69: Use the << <i>List</i> button and to list files from a parent directory.....	86
Figure 70: Use <i>Path</i> button to list files in a directory specified in path text box	86
Figure 71: Various file operations	86
Figure 72: Copy and Move can be used to manage file across hosts.	87
Figure 73: List of jobs previously submitted by a user.....	87
Figure 74: The job view page.	87
Figure 75: Select which type of jobs to start the job submission process.....	88
Figure 76: Shell script job submission interface.....	89
Figure 77: Java job submission interface.....	90
Figure 78: Confirm page shows all job parameters	91
Figure 79: User entry diagram	92
Figure 80: VO creation diagram	93
Figure 81: Resource admission diagram.....	93
Figure 82: Resource membership diagram	94
Figure 83: Overall statistic.....	95
Figure 84: Visitor overview	96
Figure 85: Top content.....	97
Figure 86: Contest result.....	98

CHAPTER I

INTRODUCTION

1.1 Introduction

A Virtual Organization (VO) is a result of multi-institute collaboration where each organization contributes its resource such as users, computing power and data to form a workforce. Its definition has been introduced since the beginning of the Grid [1]. However, it was inducted from real world scenarios rather than based on a theoretical definition. As a result, realizations of VOs in the Grid are different depending on the nature of each Grid project. There are many attempts to propose the complete model of VO [2, 3, 4] and define basic requirements for the operation of VO [5]. These proposed schemes are not easy to realize or some requirements are not crucial in an environment where Grid infrastructure is rudimentary and usability is more important.

Even so, there are some important characteristics that can be commonly found. Firstly, a VO is formed by collaboration in order to achieve certain goals, so its lifespan can be varied depending on the scale of the problem: from a year-long research project to a day-long training session, for example. Additionally, adjustment can be made anytime on VO resources to reflect the change of the associated collaboration. Secondly, multiple virtual organizations usually coexist in an environment where organizations must tackle multiple problems concurrently which lead to resource sharing among VOs. In such an environment, a user will typically participate in many VOs at the same time. These two characteristics result that the Grid will consist of lots of VOs of which resources and users are overlapping and dynamic.

From the implementation point of view, the first major problem in creating a VO is how to control usage policy in this new environment. The nature of VO introduces a new challenge in controlling the authority on Grid resources. The basic concept is that a resource has two governing entities. The first one is the resource administrator (or provider) who provides a certain amount of resources to the VO. The second one is the VO administrator who must have ability to control user access to these contributed resources. This topic has drawn much interest so there are many

answers to the problem such as VOMS [6] and CAS [7]. However, they tend to introduce more complexity and overhead in managing VOs.

For the users, working in a Grid environment currently requires much knowledge about the underlying Grid resources and tools, for example, credential management, job submission and file transfer utilities. Many technologies have been developed to hide the complexity of the Grid, increase the usability and provide a user-friendly problem solving environment. Grid Portal is probably the most widely-used solution to the problem. A Grid portal provides a familiar Web interface for the Grid. However, Grid Portal and other Grid user interfaces still have little support for a user who participates in multiple virtual organizations.

This thesis describes a design and implementation of a Grid hosting system for multiple virtual organizations. It offers a more complete solution to the difficulties in managing and accessing VO resources. Similar to a Web hosting service, Grid hosting allows administrators and users to build and manage their Grids and virtual organizations with minimum effort. By using a centralized architecture, a GHOSTS server can host multiple virtual organizations that may have overlapping sets of users and resources. A GHOSTS portal provides a single entry and working environment for the user to access all virtual organizations. Additionally, a client program which helps resource administrators manage their resources is also provided.

1.2 Problem Statement

In order to gain the benefits of applying VO concept to the current state of the Grid, there are many challenges that must be overcome before this can be achieved. Their details are discussed as follows.

1.2.1 VO Models

In one example [2] of proposed VO models, VO lifecycle starts as with the formation of a VO where partners identify goals, needs and offerings. Then, negotiation and agreements are made. The agreements are enforced onto VO activities, via usage policies of VO resources. When the goals are satisfied, hopefully, the agreements are terminated and the VO is dissolved.

However, many proposed VO models are based on theoretical foundation which is too complex to implement and not suitable for real world applications.

Therefore, this research tries to propose a VO model that strikes a balance between practicality and innovativeness. This model must be able to support some important characteristics such as the dynamic nature of VOs and the possibility of overlapping resources. On the other hand, it must be feasible to realize in an environment where the technologies and infrastructure are immature.

1.2.2 Enforcing VO Usage Policy

VOs are another level of abstraction that rest on top of the Grid infrastructure. They exist when each VO can define its own usage policy. In the present of VOs, these usage policies are enforced at two levels, i.e. VO-level and resource-level. The set of effective rights of a user to a resource is the intersection of the rights granted by the VO to the users and the rights granted by the resource to the VO (or specifically to the user). This concept can be realized in various ways. Globus Toolkit [8] supports only simple and static resource-level access control. Other products such as CAS and VOMS provide centralized control at VO-level but resource-level access control is managed by their compatible components. As a result, many Grid systems [5, 9] still rely on sophisticated ways of mapping Grid users to local accounts such as using a shared account and variations of account pool mechanisms.

The goal of these account mapping mechanisms is to reduce the number of inactive local accounts on each remote site. The account pool mechanism is chosen over the shared account mechanism when the separation of user space is required or user access must be auditable. However, each of these solutions is not a suitable for every scenario.

Both VO-level and resource-level access control must be achieved in order to enforce VO usage policy. Moreover, the authorization process at resource-level must be flexible enough to handle different requirement and dynamicity of VOs.

1.2.3 VO Usability

Currently, Grid portals are widely used to provide a familiar Web interface for Grid users to utilize Grid resources. There have been many Grid Portal builders, including GridSphere/GridPortlet [10, 11] and OGCE [12] that simplify the implementation and deployment of Grid portals.

However, most Grid portals do not support the VO concept. Therefore, a user cannot operate efficiently in case he is a member of multiple VOs. Especially, the credential management is much more complicated with multiple VOs. Therefore, ease of use should be a primary focus in designing the system, so that users easily exploit the benefits of the Grid.

Moreover, VO administration is also another factor that contributes to the VO usability issue. The complexity of creating and managing VO can be reduced by providing administration interface through the Grid portal. This will further reduce the administration cost of maintaining various VOs on the Grid.

1.3 Objectives

The objective of this research is to develop a Grid hosting solution using the Grid technologies. This research explores all aspects of VO deployment ranging from VO management, enforcing access policy, system usability, etc, in order to develop a system that can work with currently deployed technologies and infrastructure and also allows user to easily use the system. Finally, VOs should be managed with minimum efforts in order to facilitate the adoption of both VO concept and the Grid technologies.

1.4 Proposed Solution

In this research, we propose the Grid hosting model as a solution to this problem. This model is inspired by a web hosting solution where people can host their web on a web hosting provider. Therefore, VO should be hosted by a hosting provider with minimum effort so that various projects can gain the benefits of having their own VOs and utilize the power of Grid Computing.

As a result, the Grid HOSTing System (GHOSTS), which based on a Grid portal, is developed according to the proposed model. In order to achieve this, the system employs various mechanisms which are briefly discussed as follows.

1. The system is based on centralized architecture which reduces the overhead of Grid administration.
2. VO administration responsibility is delegate to various actors in the system. This means that each VO can manage certain aspects of it operation by themselves.

3. Automatic credential management is employed in order to shield users from the complexity of various credential operations. Users' credentials can be issued from GHOSTS CA or from other sources. The latter option allows user to use GHOSTS with resource from other Grid such as ThaiGrid [13].
4. GHOSTS adopts its own mechanism to enforce VO usage policy. This mechanism is aimed toward the compatibility with the existing Grid infrastructure. It can be configured to support different needs from various kinds of VOs.
5. A client programs are provided in order to help resource administrators manage their Grid nodes. One of the programs is a web service client that retrieves VO information from the server and performs account management operations on Grid nodes. The second program is a custom-developed VO authorization module that plugs in Globus Toolkit.
6. The Grid portal is based on a framework provided by GridSphere/GridPortlets. The Grid portal is modified to support the VO concept and also provides VO administration interface. Additionally, various add-ons have been developed or included in to the portal to increase its usability.

1.5 Scope of Study

1. The system is developed using the proposed model.
2. The Grid infrastructure used by this system is provided by the pre-web service components of Globus Toolkit.
3. The operating system for Grid nodes is Rock Cluster (version 4.2.1).
4. Only resources associated with the department will be used to evaluate the system.

1.6 Research Process

1. Study and evaluate various technologies relating to the topic of this research.
2. Design and develop a prototype to evaluate various concepts.
3. Design and implement the system according to the proposed model.

4. Evaluate the initial release by using a specify group of users.
5. Modify and optimize the system according to results and user feedbacks from the initial release.
6. Evaluate the final release using a large group of users.
7. Gather results and write thesis.

1.7 Expected Benefits

1. The system can be used as one of the solution for implementing VO concept on the Grid.
2. VOs can be deployed rapidly and with minimum effort by using the system.
3. The practicality of a centralized VO management solution is evaluated as part of this work.
4. The system provides easy-to-use tools which are suitable for new users of Grid system. This will facilitate the adoption of the Grid technologies.
5. The system can be used to support various types of researches and allows customized module to be hosted on the system.

1.8 Thesis Structure

The rest of the thesis is organized as follows. Chapter 2 presents the background theory and related work. In chapter 3, overview and enabling mechanisms regarding the VO hosting model are discussed. Chapter 4 describes the implementation of the system. Then, the evaluation result is shown in Chapter 5. Chapter 6 states the conclusion and future works. Finally, appendices provide user manual, use case scenarios and evaluation result.

1.9 Publications

Many parts of this thesis had been published as follows:

1. T. Kooburat and V. Muangsin, "*Building Virtual Organizations on Thai National Grid*", Thai Grid Computing Conference 2007 (TGCC2007), Bangkok, 23-24 August 2007. (Best Paper Award)

2. T. Kooburat and V. Muangsin "*The Experience in Deploying Multiple Virtual Organizations across Grid Community*" The 11th Annual National Symposium on Computational Science and Engineering (ANSCSE11), Phuket, 28-30 March 2007.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER II

BACKGROUND THEORY AND RELATED WORK

2.1 Background Theory

2.1.1 Grid Computing

Grid Computing, as shown in Figure 1, is an emerging technology built upon the foundation laid by High Performance Computing (HPC) and Distributed Computing. Therefore, the initial definition of Grid Computing is *“A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities”* [14]. This definition is based on the nature of the first generation on the Grid which is to connect high-performance computers across geographical boundary. However, as the subject evolves, the definition has changed to include other aspects which are important characteristics on the Grid such as resource sharing and virtual organizations. As a result, the latest and widely used definition of Grid Computing is *“coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations”* [1].

As the latest definition suggests, there are many facets of Grid Computing in the modern world usage. Firstly, by using the term “resource”, this means that we also take into account other type of components such as storage and network. So it is not just about the computational power alone that is important as mentioned by the first definition. Secondly, “sharing” also defines the nature of relationship of each entity within the Grid. This means that there are conditions on which resource is available to access. These conditions may be based on policies which are negotiated when the collaboration is formed. Thirdly, types of activities conducted on the Grid today also expand to distributed data analysis, multidisciplinary research, virtual classroom, etc. That is why the term “problem solving” is introduced in the definition. Finally, the result of collaboration in the Grid also creates a new type of organizations. A Virtual Organization is an organization structure that span across organizational boundary existed in the real world.

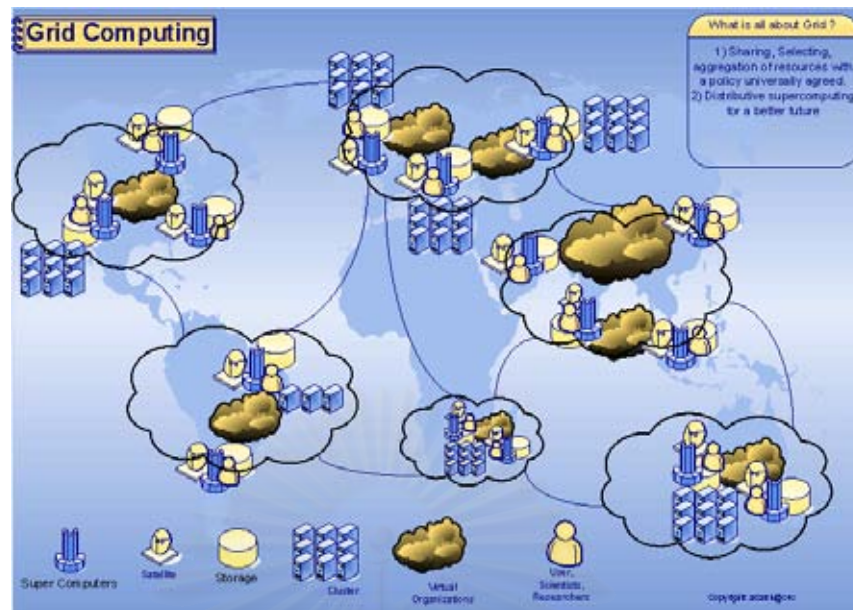


Figure 1: Grid Computing¹

The type of work conducted on the Grid also defines the nature of each Grid. Here, we will give examples of the Grid that is currently deployed.

- **Computational Grids** — Applications that run on this type of Grids usually need huge amount of computational power such as drug discovery, simulation and mathematical modeling. Thus, organizations must combine their computational resources in order to tackle these problems which result in Computational Grids
- **Data Grids** — Some types of experiments produce large amount of data and these data must be accessible by the community. Data Grids is built on conventional grids by including the data management features such as data replication and file access control. EGEE [15] is a well-known example of this type of Grids.
- **Collaborative Grids** — Multidisciplinary researches have lead to collaboration among various institutions. In order to facilitate this type of research, a special environment is needed so that researchers can effortlessly work together. Access Grid [16] is an example of tools that promote such collaboration within the Grid. Figure 2 shows the usage of one of the Access Grid nodes.

¹ From: <http://www.csa.com/discoveryguides/grid/reviewf.php>

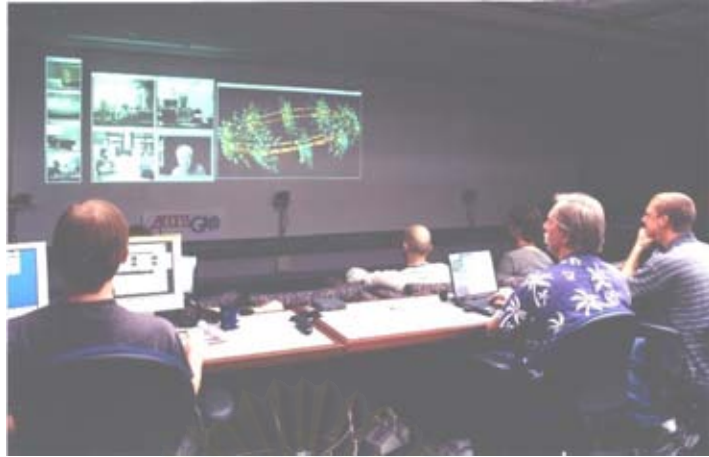


Figure 2: Example application of Access Grid²

2.1.2 Virtual Organization (VO)

Virtual Organization (VO) is a result of multi-institution collaboration where each institute allocates its resources including people, data and software to perform a collaborative problem solving. This type of cooperation results in a new organization called a Virtual Organization, which spans across conventional organization boundary. This kind of organization is good description of how various collaborations work together on the Grid infrastructure.

Today, there are many projects that can be considered as VOs. For instance, a development of a new aircraft which requires multidisciplinary researchers, a high energy physics research which involves hundreds of laboratories around the world. These examples come from a different domain of problems; however, they have a need for the same basic requirements in order to operate such as flexible resource sharing relationship and multi-stakeholder access control. As a result, satisfying these basic requirements is a primary objective in developing the Grid technologies because VOs allows diverse groups of institutions to share resources in a controlled manner, so that their members will be able to efficiently collaborate and achieve their objectives.

² From: <http://www.accessgrid.org/node/450>

2.1.3 Grid Architecture

2.1.3.1 The Hourglass Model

One of the proposed models of the Grid is the Hourglass model [1] depicted by Figure 3. In this model, all protocols and abstractions sit in the middle layer which is the neck of the hourglass. They can act as interfaces to various underlying technologies which are the base of the hourglass. Moreover, at the top of the hourglass, high-level applications can be developed by using these interfaces without having to interact directly with low-level resources.

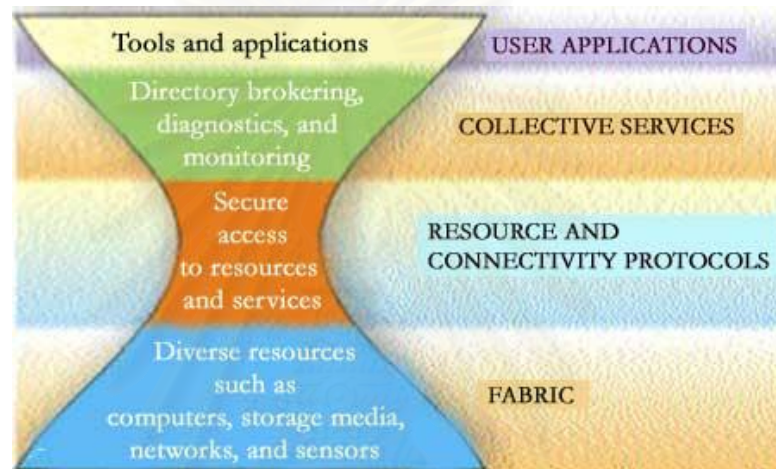


Figure 3: Hourglass model³

2.1.3.2 Grid Middleware

According to the hourglass model, a Grid middleware are the neck of the hourglass. It provides common services necessary for developing distributed applications according to the Grid paradigm. The complexity of developing such applications is reduced because developers need not learn how to communicate with low-level resources. Currently, there are two Grid middleware that receive popularity which are listed in the following section.

Globus Toolkit

Globus Toolkit [8] is an open source Grid middleware used for constructing a Grid. It provides common services which serve as “build blocks” for building Grid applications. These services are grouped in five domains as shown is Figure 4.

³ From: http://www.globus.org/grid_software/ecology.php

- **Security** — Provides security model based on Grid Security Infrastructure (GSI) [17]
- **Data Management** — Provides functions for managing large sets of data
- **Execution Management** — Handles the initiation, monitoring, management, scheduling and coordination of job
- **Information Services** — Provides a set of components for resource discovery and monitoring
- **Common Runtime** — Provides a collection of fundamental libraries and tools which are essential for building Grid services

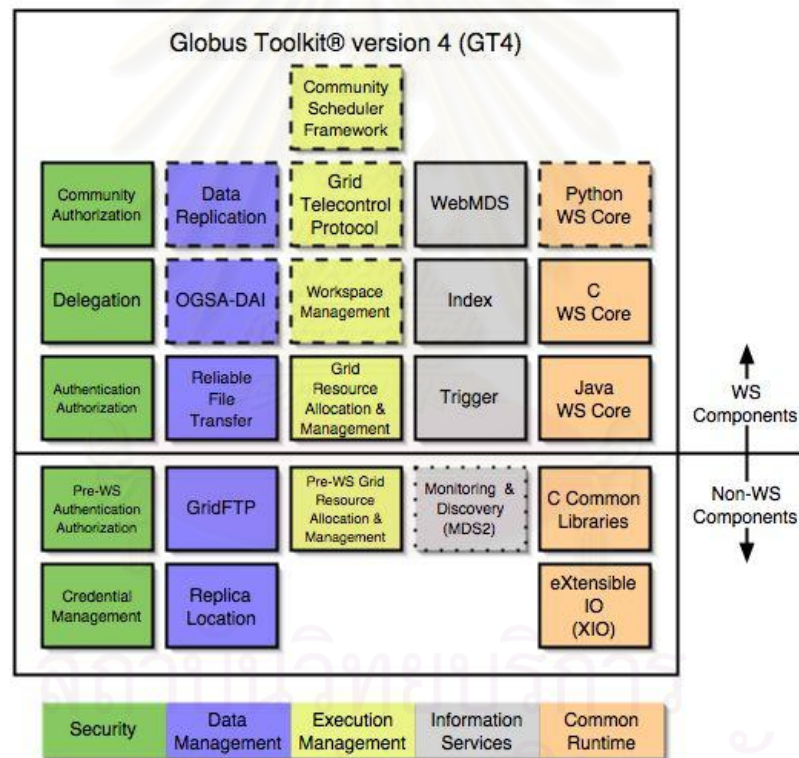


Figure 4: Globus Components⁴

The Globus Toolkit is chosen as a platform for build the Grid in this thesis because a wealth amount of software and support are available. Therefore, we are able to use various Globus-compatible components to construct an integrated system according to our model.

⁴ From: <http://www.globus.org/toolkit/about.html>

UNICORE

UNICORE (UNiform Interface to COmputing Resources) [18] is another alternative of building a Grid. It provides a vertically integrated solution by including various components ranging from graphic user interface to server components. Its architectural diagram is illustrated in the Figure 5.

UNICORE composed of three tiers: user, server and target system. Firstly, the user tier is a client program that provides Java-based GUI used for job submission and data management. Secondly, the server tier is responsible for authenticating user requests and dispatches them to specific targets. Finally, the target system tier is a daemon running on a target system where it acts as an interface to the local operating system. In addition, UNICORE can handle a Globus-enable machine as one of the target system.

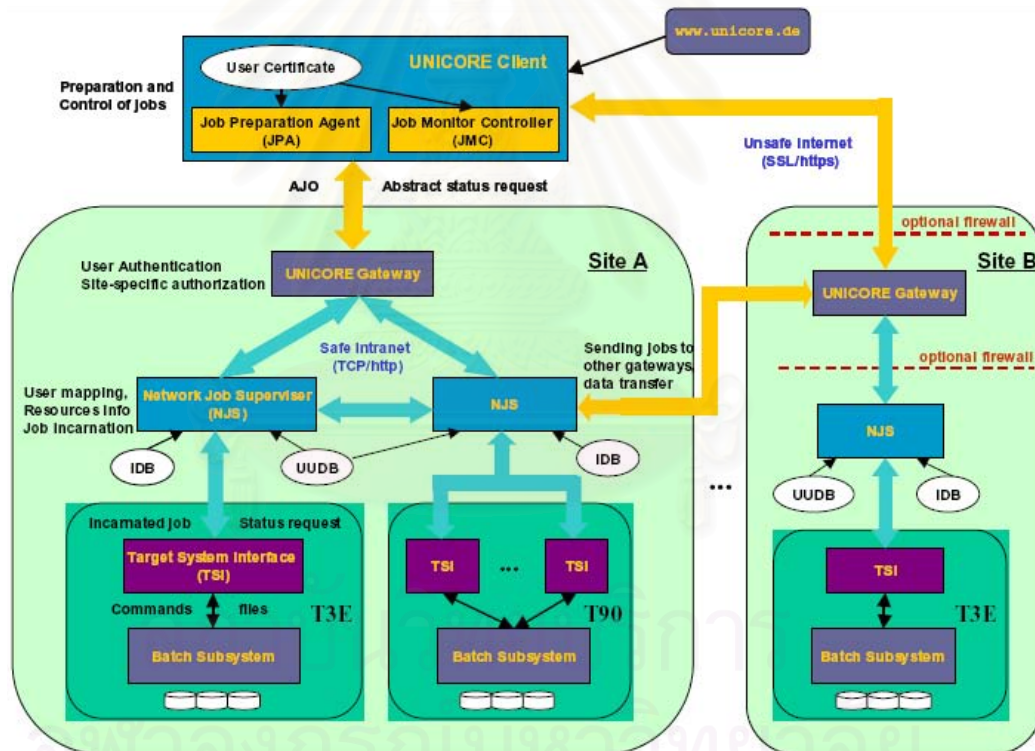


Figure 5: The UNICORE Architecture

2.1.3.3 OGSA

Open Grid Service Architecture (OGSA) [19] is an effort lead by the Open Grid Forum (OGF) [20] to increase interoperability among various Grid solutions by using the process of standardization. OGSA is drawn from a service-oriented architecture (SOA) and built on the web service infrastructure. OGSA defines a

collection of core capabilities and behaviors which are necessary for realizing Grid systems. These capabilities, depicted in Figure 6, are: Infrastructure, Execution Management, Data, Resource Management, Security, Self-Management, and Information services. In summary, OGSA can be considered as a refinement of Grid system requirement and also the method of building such system on the web service architecture.

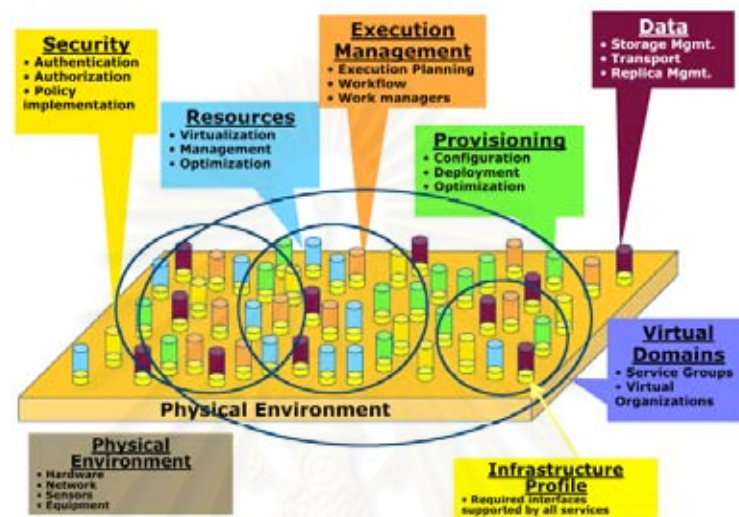


Figure 6: OGSA capabilities⁵

Since the introduction OGSA, the Grid community is moving toward the web service paradigm. However, many software components of the web service architecture have not yet achieved the production quality during the development of this thesis. As a result, our work is based on pre-web service components of the Grid.

2.1.4 Grid Security

Security is one of the most important aspects of Grid Computing. This is because the Grid must have an adequate safeguard to ensure that resources from various institutions are accessible only by those whom they trusted. We will explore the details of Grid security in the following topics.

2.1.4.1 Public Key Infrastructure

Public Key Infrastructure (PKI) is used at the core of Grid security. PKI provides a mechanism for many security features that are important for the operation of the Grid such as authentication and delegation. PKI uses a public key cryptography

⁵ From: The Open Grid Services Architecture, Version 1.5.

which employs two different keys – public and private key. Whichever key is used in the encryption process, the other key is used in the decryption process. As its name suggested, the private key is kept in secret and used by the sender to encrypt a message sent to the receiver. Then, he will use the sender's public key, which can be distributed without using a secure channel, to decrypt the message. This process is shown by Figure 7.

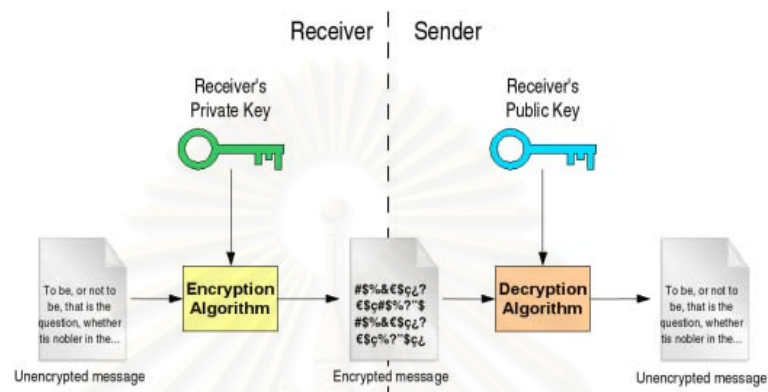


Figure 7: Secure communication using the public key cryptography⁶

Another important use for public key cryptography is the digital signature. With this process, illustrated in Figure 8, the message integrity is a primary goal. The message digest is created using certain hash algorithm, and then it is encrypted and sent to the receiver with the message. The receiver decrypts the message digest and compares it to the one he created from the received message. If the two matched, then the receiver can be sure that the message has not been tampered with.

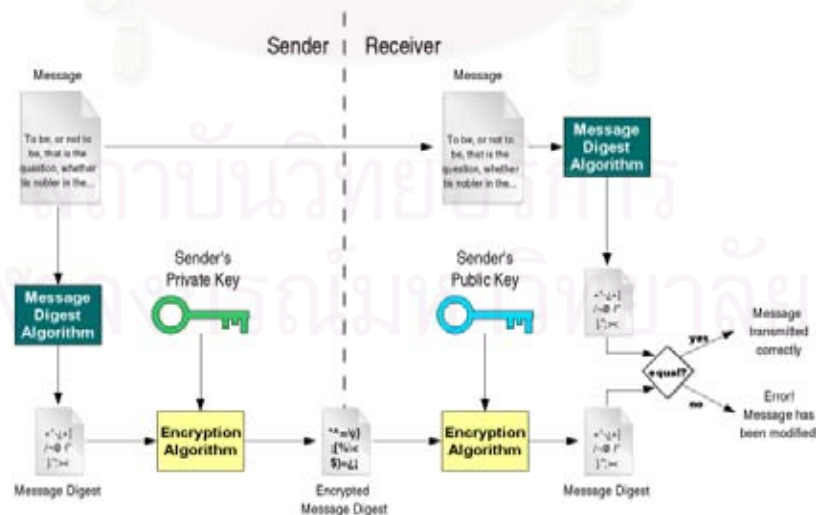


Figure 8: Digital signature using the public key cryptography⁶

⁶ From: http://gdp.globus.org/gt4-tutorial/singlehtml/progtutorial_0.2.1.html

However, using these mechanisms is not sufficient to create a trusted relationship between two entities because the public key distribution is subjected to compromise. In order to prevent this pitfall, PKI make use of various components which are explained in the subsequent sections.

Digital Certificate

Digital certificate is a digital document that binds the identity of the owner to his public key. X.509 is a format of a certificate which is used by the Grid community. In this standard, a certificate encodes its owner's name in a Distinguish Name (DN) entry and store the public key in hexadecimal numbers. Figure 9 depicts some sections of a digital certificate.

Certificate Authority

Certificate Authority (CA) is a certificate issuer which is trusted by the community. In order to prevent certificate forgery, CA inscribes a digital signature using its own certificate on every certificate it issued. As a result, certificate validation can be done by using CA's certificate and examining the digital signature.

Issuer:	O=CU, OU=ENG, OU=CP, CN=CP CA
Subject:	O=CU, OU=ENG, OU=CP, CN=Thawan
Validity:	Not Before: Jan 5 03:35:05 2007 GMT Not After : May 19 03:35:05 2008 GMT
Subject Public Key:	00:b2:69:00:9a:92:f0:1a:c6:57:5b:d4:2d:89:df: 5c:92:c1:8b:4a:84:48:62:16:00:ae:9b:35:b2:b9:
Signature:	8d:be:3f:b4:11:4a:d7:3f:65:91:d1:e1:f1:02:38:1b:ba:62: 5d:14:2b:ca:fc:95:ff:f1:51:b3:ce:64:68:7a:45:bd:70:2f:

Figure 9: Digital certificate

Proxy Credential

A pair of certificate (public key) and private key is called a credential. Normally, user credential will last for one year after it is issued. However, if the credential is stolen then it will create a major security threat because it can be used to identify as him as long as the credential is valid (or until it is revoked). Therefore, a short-lived credential – proxy credential is issued by using user certificate. This proxy credential can be used in the same way that a user uses his credential because it is also

bind to his identity. Moreover, the severity of threat is also reduced if compromised because it has a short life time.

Subject:	/O=CU/OU=ENG/OU=CP/CN=Thawan/CN=618872044
Issuer:	/O=CU/OU=ENG/OU=CP/CN=Thawan
Identity:	/O=CU/OU=ENG/OU=CP/CN=Thawan
Type:	Proxy draft (pre-RFC) compliant impersonation proxy
Timeleft:	11:59:59

Figure 10: Proxy certificate

2.1.4.2 Grid Security Infrastructure

Grid Security Infrastructure (GSI) is based on the PKI and it is a collection of tools, libraries and protocols used for operating a secured Grid system. For example, it contains command-line tools and API for managing certificates. Next, we will discuss some security features of provided by GSI.

Delegation

In a distributed system, a complex task can be divided and executed by other entity within the community. Delegation is a security feature that allows other entities to perform certain tasks on our behalf. By using a proxy credential, any entity which holds our credential will be able to identify itself as the owner of that credential.

Submitting a job to a remote host can be used to show how delegation works in the Grid. In such scenario which is presented by Figure 11, the job may need to read input file available on another remote host. The simple method would be that the user must transfer the input himself to the job's host. However, the job can be instructed to perform this task because it also has his credential which can be used to authenticate with another remote host and retrieve the file before the job execution.

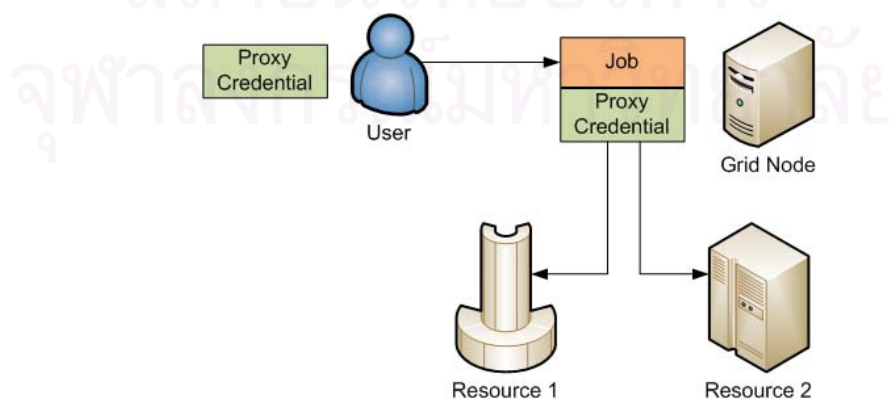


Figure 11: Credential delegation in the job submission process

Single Sign-on

From the internet user perspective, the term “single sign-on” means that we will be able to login to a set of web site by logging in to just one of the sites. However, when this concept is applied to the Grid, it means that a Grid user can login to any machine in a Grid system by logging in to just one machine. A user credential can be used to perform this task; however, it is usually stored in an encrypted format which requires him to supply the credential passphrase every time he wants to use the credential. This problem is solved by using the proxy credential instead of his permanent credential. Because the proxy’s life time is short, it can be stored in a plain text format.

Authentication and Authorization

Authentication is a process which determines if a person has a correct identity and authorization is a process which determines if a person is allowed to perform a particular task. In the Grid, these processes are performed when a Grid user is accessing any remote resource within his community. The authentication process will usually use the credential validation mechanism in order to determine its result. However, there are many solutions to authorization process which happens at the local resource. The goal of an authorization system is to enforce the local usage policy created by the resource provider.

The Grid Map authorization is a simple and default mechanism provided by the Globus Toolkit. It used a “*grid-mapfile*” (shown in Figure 12) which contains entries mapping Grid users’ DNs and local accounts. This means that a Grid user will have rights according to a local accounts associated with his credential’s DN.

/O=CU/OU=ENG/OU=CP/CN=Thawan	thawan
/O=Grid/OU=Thaigrid/CN=Thawan	thaigrid

Figure 12: Example “*grid-mapfile*” entries

2.1.5 Grid Portal

A Grid portal provides a web-based graphic user interface for interacting with Grid resources. It usually offers an easy-to-use interface for existing Grid services and allows user to work with Grid resources from any machine without installing Grid software. Current generation of Grid portals employed the portlet model [21] which is specified as a standard by JSR 168 [22]. GridSphere/Gridportlets

is an example of Grid portals that follows this specification. In the following section, we will describe the details of these topics.

2.1.5.1 Portlet

The portlet model can be considered as a component-based methodology for web application development. Each web application or service is encapsulated in a portlet which can be deployed on any portlet-based web portal. Code reusability is also increase by using this model because each portlet is not a stand-alone web application. It can use any existing services or functionality provided by the portlet container such as user authentication. Additionally, it is easy to customize a portlet-based web portal because a new service can be introduced easily by deploying a new portlet.



Figure 13: Portlet-based web portal

2.1.5.2 JSR168

Java Specification Request 168 (JSR168) is a Java portlet specification released in 2003. It defines the APIs for developing portlets and provides rules regarding user data, deployment, packing and etc. in order to accomplish interoperability among portlets and portlet containers.

Since it released, the specification has been implemented by many vendors; however, they have found various shortcomings. This leads to the developing of custom extension to the portlet API and other areas. As a result, the expert group was formed and they are developing the next iteration of the specification which is JSR 268 [23].

2.1.5.3 GridSphere

GridSphere Portlet Framework is an open source web portal which acts as a portlet container. It followed the JSR168 specification; however, it also introduced the portlet service layer to the portal architecture. A portlet service can be considered as an application logic layer of each portlet and it can be called by any portlet in the portal. GridSphere provides common services such as user management and database management through its portlet services. As a result, this feature facilitates the portlet development and the portal's behaviors can also be customized without modifying the portal's source code.

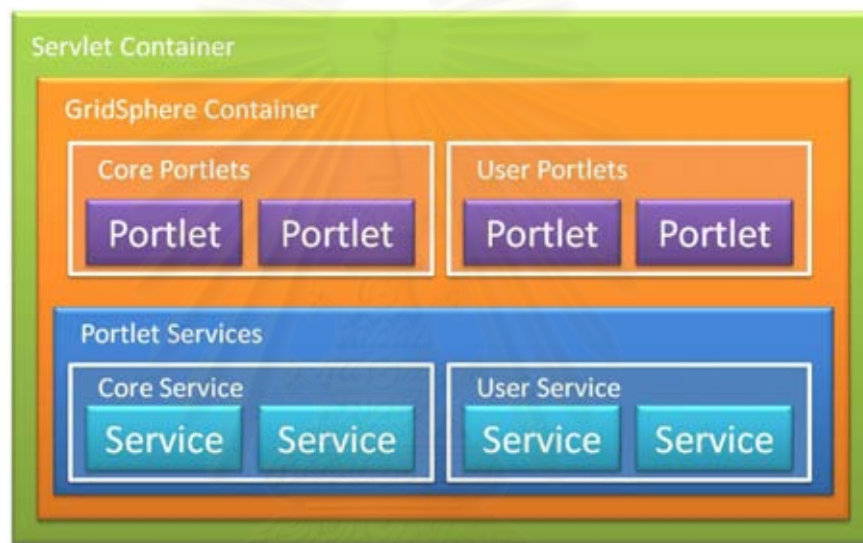


Figure 14: GridSphere architecture

2.1.5.4 GridPortlets

Despite its name, GridSphere does not come with the Grid functionalities. Instead, it is provided in a separate portlet application called GridPortlets. GridPortlets is a collection of portlets that provide Grid functionalities such as credential management, job submission, file transfer and resource browser. Besides, GridPortlets encapsulated all its functionalities in a set of portlet services. This allows developers to use GridSphere/GridPortlets as a framework for developing web-based Grid application.

GridPortlets also shipped with its own programming model – the Action Component Portlet. This model increases the granularity of the component of the web application by making each web page its own component. Moreover, these

components could be used other portlets. As a result, the code reusability is increase but at the cost of reduced portability because it deviates from the portlet specification.

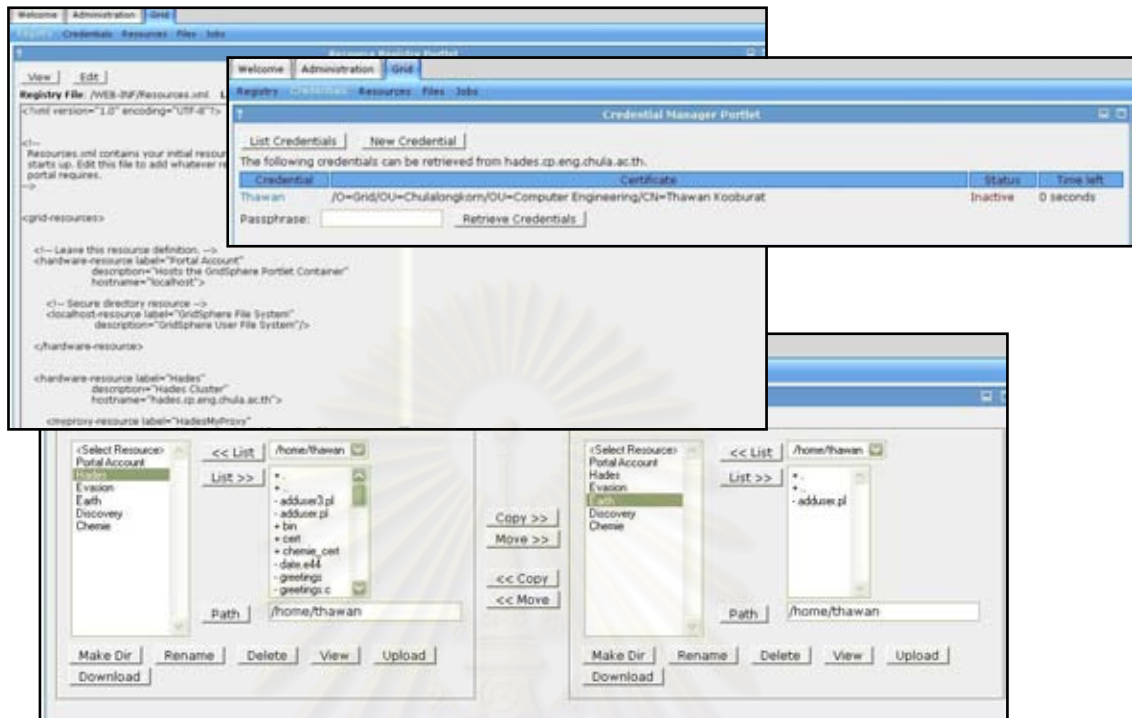


Figure 15: Screenshots from GridPortlets

In a common configuration, the GridSphere/GridPortlets grid portal is deployed along side MyProxy [24] which is a credential repository that stores user credentials. A user must delegate his credential to MyProxy and instruct the portal to retrieve his proxy credential. Thus, this will allow the portal to perform tasks on his behalf.

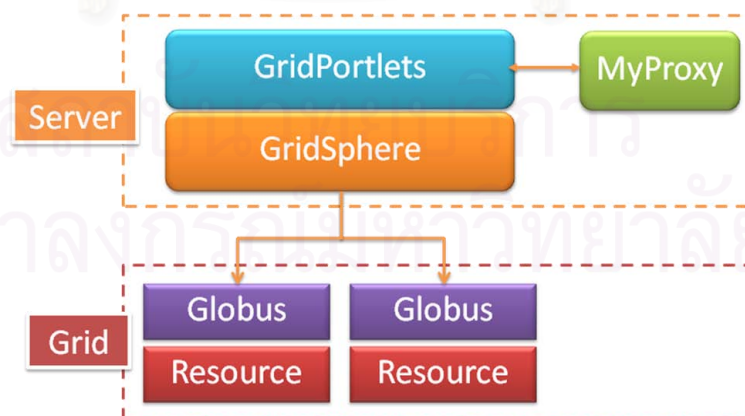


Figure 16: GridSphere/GridPortlets deployment

2.2 Related Work

Researches that are related to this thesis can be divided into three areas which are discussed in the followings

2.2.1 VO Authorization

This area addresses the problem of authorization mechanism need by the operation of VOs in the Grid environment. We will discuss two major works regarding this area which are CAS and VOMS.

2.2.1.1 Community Authorization Service

The Community Authorization Service (CAS) is now a part of the Globus Toolkit which provides an authorization mechanism that allows VOs to enforce their policies in a distributed environment. CAS proposed that VO should be able to manage all its rights given by its resource providers. As a result, a user must contact CAS server in order to retrieve his rights allocated to him by his VO. CAS inserts security assertions into his proxy credential that tell what can be accessed by him. Then, he will use this “restricted” proxy credential to contract a resource provider in order to perform his task. The resource provider inspects his request and security assertions in order to determine if he has enough right to perform the request.

CAS inserts fine-grained access control into users’ credentials. This means that CAS-aware services are needed in order to interpret these security assertions. Moreover, in one of CAS works [25], local policy enforcement is imposed by simply mapping different assertions to different pools of UNIX accounts. As a result, it is still not a complete solution for enforcing VO usage policy.

2.2.1.2 Virtual Organization Membership Service

The Virtual Organization Membership Service (VOMS) has a similar architecture to CAS, but with a different model. A VOMS server maintains only VO membership information including groups, roles and capabilities. A user must contract VOMS server in order to retrieve these attributes and includes them into his proxy certificate. When a resource provider received his request, it must retrieve membership information from his Attribute Certificate and translate this information according to local usage policy.

Therefore, VOMS-aware services are also required in order to utilize this authorization mechanism. VOMS solves this problem by using LCAS and LCMAPS [26]. The first one is a pluggable framework for authorization module, and the later one is also a pluggable framework that maps Grid credentials to UNIX accounts. This shows that enforcing local policy through UNIX accounts is still a necessary option especially when operating with non-Grid services.

2.2.2 Virtual Accounts and Account Pool

Dynamically create accounts or retrieving pre-generated accounts from account pool have been previously implemented by many projects. These two mechanisms are used in order to provide execution environment for Grid users on remote site. Examples of these works are discussed in the followings.

2.2.2.1 Dynamic Accounts

Dynamic Accounts [25, 26] (formerly known as Workspace Management Services) is a technical preview released as part of Globus Toolkit 4. This component allows client to dynamically retrieve UNIX accounts on a remote site. The account assignment can be performed by generating a new UNIX account or using an account pool supplied by LCMAPS. The access policy for these accounts is recorded into “*grid-mapfile*” and it is cleared when their time to lives (TTL) are expired. Moreover, it can also accept VOMS credentials and process authorization based on their attributes.

When using this mechanism, a user must contact the factory service first in order to retrieve an account on a remote site, only a modified version of GRAM is capable of automatically allocate an account on behalf of the user.

2.2.2.2 Virtual User System

Virtual User System (VUS) [29] maintains groups of virtual accounts that have different privileges. Therefore, access control is achieved by authorizing different users to different groups of accounts. This decision is made by pluggable authorization module which performs authorization based on VO membership. Therefore, it needs less configuration work than maintain one-to-one mapping information employed by the “*grid-mapfile*”. VUS also focus on providing

accounting and usage statistic information regarding user so that VO manager or resource provider can inspect this data.

VUS does not mention how it can be used to achieve the separation or preservation of user space. For the separation of user space, this means that multiple users might be able to authorize to the same account. For preservation of user space, this means that a user might be authorized to different accounts every time he access the remote site, so he cannot use data previously stored in his former virtual account.

2.2.3 Grid Portal Solutions

In this section, we will discuss other possible solutions that might be able to use in order to achieve the same goal as this thesis.

2.2.3.1 GAMA

The Grid Account Management Architecture (GAMA) [30] is a system that handles Grid account management for various types of applications such as Grid portals and rich-client applications. It comprises two components groups: a backend server that generates and manages user credentials, and a group of portlet interfaces for account management that can be deployed with GridSphere/GridPortlet portal. A user can request a Grid account from GAMA and use the generated certificates on other applications. For an administrator, he can generate rules for accepting or rejecting users based on their domain names. Moreover, GAMA provides a portlet for “satellite portals” which can be used to import Grid users from the GAMA server.

GHOSTS is similar to GAMA in the way that it provides the account management functionality where their mechanisms can be compared in the following points. Firstly, allowing other applications to retrieve generated user credentials is outside the scope of our system. However, we allow users to store their credentials on our server so that they could use it to interact with other Grid system, whereas GAMA relies solely on its CA. Secondly, CAS proxy credential can be retrieved from the GAMA server. In theory, this could be used to provide authorization mechanism for VO environment; however, it is only an experimental feature and it does not stated how it is going to integrate this mechanism with Grid portals in order to enable the VO concept. Finally, a generated Grid account can be fully utilized when all Grid resources modified their grid-map-files. However, GAMA only provides a basic

command line tool for creating a local account and modifying the mapping which must be executed on per user basis, whereas GHOSTS provides an automate tool for handle this task and also other account mapping mechanisms.

2.2.3.2 GENIUS

Grid Enabled web eNvironment for site Independent User job Submission (GENIUS) [29, 30] is a production-grade Grid portal released within the context of the EGEE. The portal is based on the EnginFrame framework and acts as an interface to various middleware such as Globus, EGEE LCG/gLite and Sun Grid Engine. It provides various services ranging from job submission, data management, and interactive services through TightVNC. Most of these services support the operation of VOs by using security features provided by VOMS. For example, its distributed data management allows user to publish or retrieve files shared within his VO. Moreover, it is also possible to develop complex wizard-like interface that interact with high-level Grid services by using its scripting language or Java programming.

GHOSTS is designed to support users whose knowledge of Grid technologies in minimum. For example, the credential management of GHOSTS completely shields its users from knowing how to operate his credentials, whereas, GENIUS lets users handle VOMS credentials by themselves. Moreover, GENIUS relies on matured EGEE infrastructure but GHOSTS has to manage and construct its own Grid.

CHAPTER III

GRID HOSTING MODEL

3.1 Overview

The key ideas of GHOSTS are motivated by Web hosting services and Web content management systems. A Web hosting service allows individuals and organizations to host their websites and users on the provider's server. The web host provides storage space, internet connectivity, database support, and application platforms. A Web content management system provides templates and customizable application modules to automate creating and managing websites and presentation of Web content. Both models of Web development significantly contribute to the growth of the Web today.

We believe that the same models that work for the Web can be naturally applied to the Grid. A virtual organization in the Grid is analogous to a website with some collaborative applications and a group of users. A major difference is that most resources and services of a VO are actually somewhere else in the Grid.

Therefore, we propose a model of Grid hosting system and have developed GHOSTS, a Grid hosting system that is designed to achieve the following objectives.

1. The system can host multiple dynamic VOs with overlapping resources.
2. VOs can be created rapidly and have their own management that can be done via a Grid portal.
3. A user can join multiple VOs and access Grid resources in those VOs via a Grid portal.
4. The system is easy to deploy and use.
5. The system is extensible to support specific requirement of each VO
6. The system can be used with existing Grid infrastructure and applications.

3.2 VO Model

Based on real world examples such as international high-energy physics collaboration, a national infrastructure project, a multi-institution drug design and a

training workshop, we have identified several types of VOs with distinct characteristics which are life-time, applications, dynamicity of resources and users. Their differences in these aspects should be addressed when designing a Grid system that supports the operation of these VOs.

1. **Infrastructure-oriented VOs** — This kind of VOs is created in conjunction with the infrastructure, so it spans across the whole system. It provides basic services and lives as long as these services are provided. Examples of such VO include EGEE [15] and ThaiGrid [13].
2. **Community-oriented VOs** — It can also be called Domain-specific VOs. This kind of VOs focuses on some specific application or subject domain. It often provides community-wide services which usually run on top of services provided by an infrastructure-oriented VO. An example of this kind of VO is the LHC Computing Grid Project (LCG) in EGEE.



Figure 17: LCG networks¹

3. **Project-oriented VOs** — Their collaboration exists to perform some specific task, e.g. a research project, for a specific period of time. The examples of these VOs are the LHC experiments which result in four separate Grid VOs.
4. **Ad hoc VOs** — This model includes short-term collaboration such as training courses and conferences. A system that can automatically generated VOs to provide on-demand services [2] can also facilitate the creation of these VOs.

¹ From: <http://www.sciencemag.org/cgi/content/full/313/5786/433/F1>

3.3 Enabling Mechanisms

In this section, we discuss important mechanisms that enable the Grid hosting functionality. Each topic is briefly described and listed in the following paragraph. Their details are discussed in the latter part of this section.

1. **System Architecture** — Centralized server is used to reduce the overheads of managing dynamic VOs.
2. **User Management** — Various roles within the system are responsible for different tasks of VO administration. Therefore, each VO has right to manage certain aspect of its operations.
3. **Credential Management** — GHOSTS automatically manage credentials for its users. This increase system usability especially when users have no familiarity with the Grid technologies.
4. **Access Control Mechanism** — Access control mechanism adopted by GHOSTS is a combination of a VO-level access control imposed by the Grid portal and a resource-level access control which use three methods of mapping user to different local UNIX accounts. These two levels of access control allow GHOSTS to deploy VO usage policy on existing Grid infrastructure.
5. **Resource Management** — Client programs are installed on GHOSTS-managed nodes. These programs help resource administrators manage their machine and insert VO authorization module into Globus Toolkit.
6. **VO-aware Grid Portal** — Grid portal based on GridSphere/GridPortlets is modified and enhanced to support the VO concept. This allows users to easily handle multiple VOs memberships. Moreover, VO management functionality is also provided through the portal interface.

3.3.1 System Architecture

GHOSTS is based on a centralized architecture. A central node called GHOSTS server is responsible for the management of Grid nodes and users. A GHOSTS server acts as certificate authority, certificate repository and Grid portal. This combination reduces the complexity of VO management behind a single Web-based portal and also improves the usability of the systems.

GHOSTS gathers all accessible resources into a single Grid. Each VO hosted by GHOSTS is a subset of this Grid. Moreover, its set of resources and users may overlap with that of other VOs. We can visualize these VOs as small Grids on top of a larger one. GHOSTS server uses a Grid portal as the main interface for management and accessing Grid resources. Figure 18 shows the system diagram.

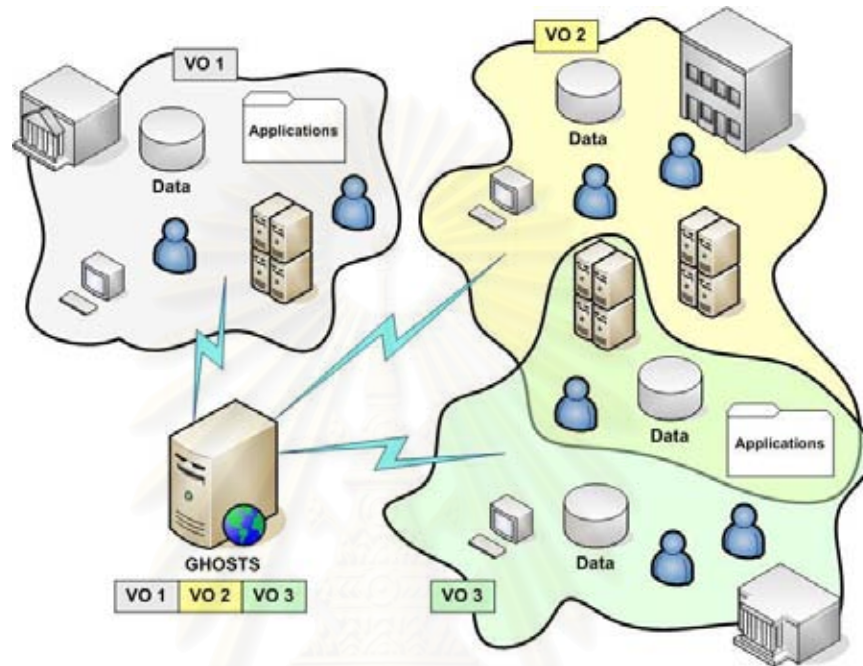


Figure 18: GHOSTS system diagram

3.3.2 User Management

The GHOSTS server maintains a user account database. Each GHOSTS user can join multiple VOs and may have different roles. There are four user roles in GHOSTS, namely User, VO Administrator, Resource Administrator, and Administrator. Figure 19 depicts relationships of various roles in GHOSTS and their responsibility is described as follows.

- **User** — A User is a person who have GHOSTS account; he can have any number of VO member including zero. Besides from utilizing his VO resources, a User can request for VO membership or a new VO to be created. The latter option will allows him to become a VO Administrator of that VO.
- **VO Administrator** — A VO Administrator is a member of VO who has right to access VO management of his VO. He can manage every aspect of

VO including approving various membership requests created from Users or Resource Administrators.

- **Resource Administrator** — A Resource Administrator is a User who has right to manage his resource on the portal. A User can be promoted to this role by submitting his resource to the portal.
- **Administrator** — An Administrator is responsible for managing the whole system. His main responsibility is admitting various entities to the system such as users, VOs and resources. Additionally, he can also manage all VO hosted on the system but this is not his main responsibility.

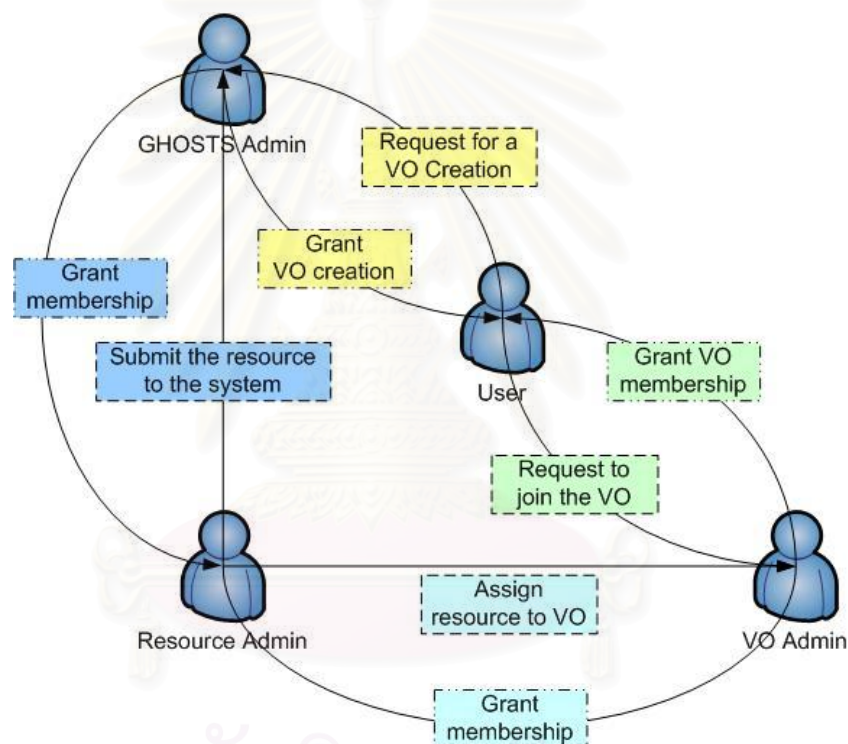


Figure 19: Role relationships in GHOSTS

3.3.3 Credential Management

For simplicity, the GHOSTS server is responsible for credential management on behalf of the users. The GHOSTS' Certificate Authority automatically generates a credential for a user for each VO that he joins. Moreover, a user may also upload his credentials to the portal repository. This will allow him to use the Grid portal as an interface to other resources which are not managed by GHOSTS such as resources from the Thai National Grid Project.

As a result, this leads to the fact that a user may have multiple credentials on the portal if he has multiple VO memberships. Thus, we also provide the automatic credential management in order to assist our users. A user only needs to specify which VO he is going to use, then the system will automatically initialize his credential that is associated with that VO. This process is illustrated in Figure 20.

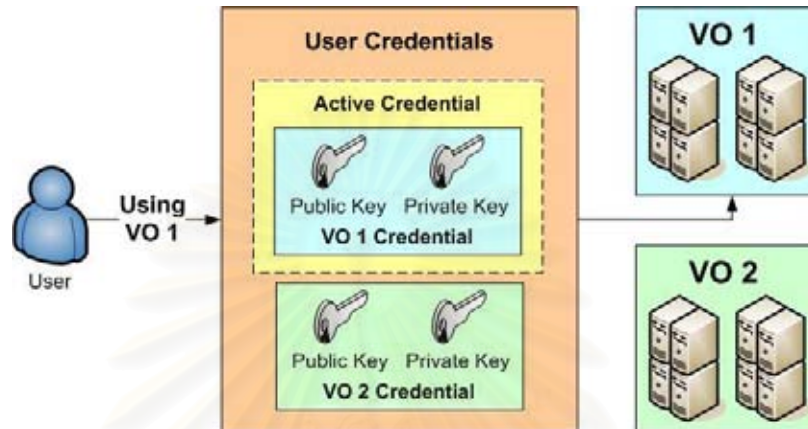


Figure 20: Automatic credential management

3.3.4 Access Control Mechanism

GHOSTS provides two levels of access control which are VO-level and resource-level.

3.3.4.1 VO-level Access Control

VO-level access control is a coarse-grained usage policy that expresses what resource is available to any given user within a particular VO. This level of access control is handled at the Grid portal level by restricting portal users from accessing resources that are not belong to his currently activated VO. Nevertheless, by switching between VOs, he will be able to access other resources that are associated with various VOs in which he has memberships

In this flat VO model, it is not possible to construct sub-groups within each VO. However, the same effect of having sub-groups can be achieved by creating a separate VO for each sub-group which is shown in Figure 21.

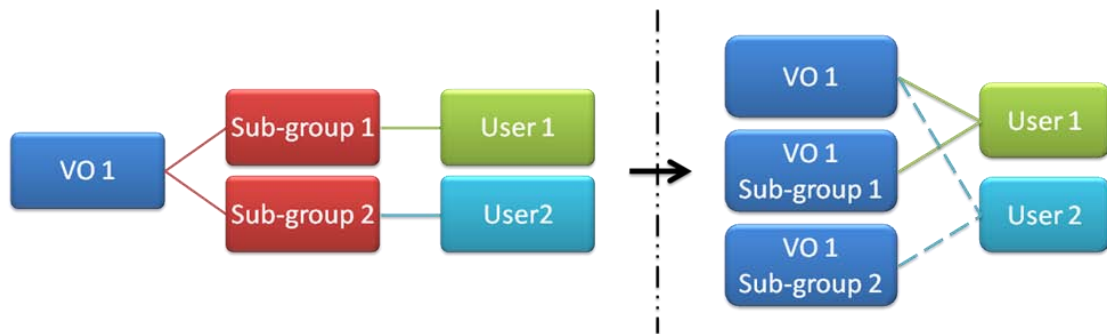


Figure 21: Mapping VO sub-groups structure to flat VO model

3.3.4.2 Resource-level Access Control

Resource-level access control is a fine-grained usage policy that governs how users access each resource. In order to reduce the complexity of management, GHOSTS rely on existing local access control mechanism on each resource to restrict user access which will allow resource administrator to use familiar tools to manage Grid users.

Moreover, GHOSTS provides three account mapping schemes which govern how Grid users should be mapped to local accounts. This facility further reduces the administration cost by reducing the number of local accounts that need to be accommodated by Grid users. Each scheme is suitable a specific type of VOs, so it can be adopted independently by each VO.

One-to-one Account Mapping

In this scheme, a Grid user has a separate account for each VO that he joins on each resource. These local accounts are created automatically by GHOSTS' client program. Besides, the resource administrator can impose VO-wide policy on his resource because these local accounts are members of UNIX's group associated with their VO. The mapping is suitable for VOs that want a complete separation of user space and preserve user data on its resources. Moreover, the generated account will be efficiently utilized if VO users are actively use it resources. Figure 22 shows the mechanism of one-to-one account mapping.

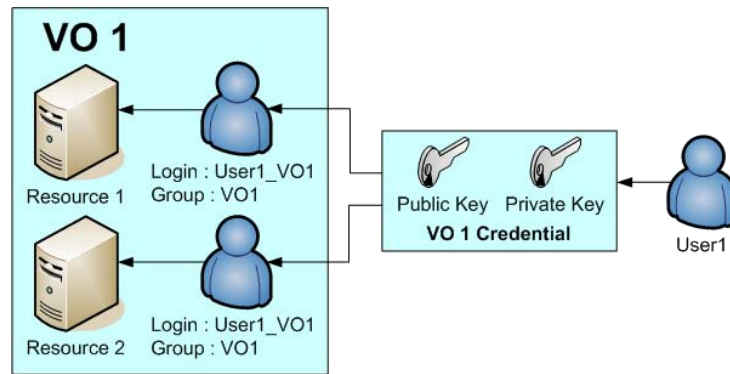


Figure 22: One-to-one account mapping

Additionally, this scheme also support mapping of existing accounts to Grid users. This feature provides a migration part for existing cluster users because they can quickly move their jobs onto the Grid.

Account Pool Mapping

This scheme is similar to DHCP leases mechanism where the resource administrator must create a set of accounts and assign them to each VO's pool. When a Grid user authenticate to the resource, an available account from his VO's pool will be given to him for a period of time. Therefore, a subsequent access by the same user during that period will grant him the same local account. Even after that period of time, if the same account is not given to other Grid user, he will be granted with that account. In case of a depleted pool, the resource will deny his access. In addition, the authorization mechanism in this scheme also authorizes users based on their VO's DNs. This means that a new VO user can instantly access the resource after he has a membership of this VO. The diagram for this mapping mechanism is depicted by Figure 23.

Therefore, this mapping mechanism is suitable for VO which has a noticeable amount of inactive users. As a result, creating an account for each user will result in many inactive local accounts. Therefore, this scheme reduces a number of active local accounts that the resource administrator needs to manage.

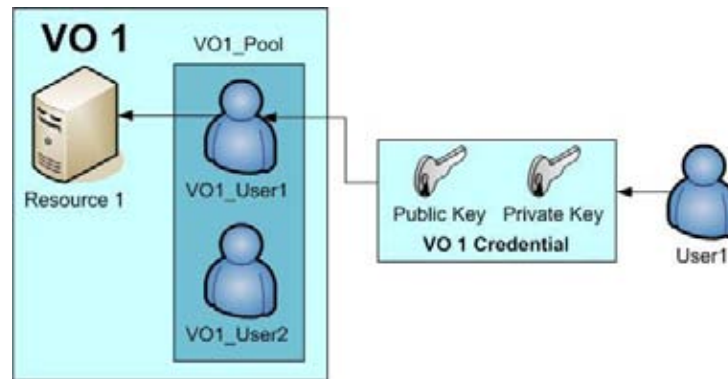


Figure 23: Account pool mapping

Shared Account Mapping

This can be considered as a special case of account pool mapping that has only one account in a pool. Every Grid user in a particular VO will be assigned to the same account whenever they access this resource. Additionally, it also employs VO authorization mechanism as found in the account pool mapping. As a result, this scheme is suitable for VO which has a large number of users and separation of user space is not required.

An additional benefit of using this scheme over the account pool is that an increase in the number of Grid users does not affect the availability of the resource. This is because every user will be able to access the system; whereas, the account pool mapping will reject an incoming user if the account pool is depleted.

The main objection regarding the use of shared account is the lack of the ability to audit user access. This can be solved easily by implement logging mechanism that logs users accessing the shared account.

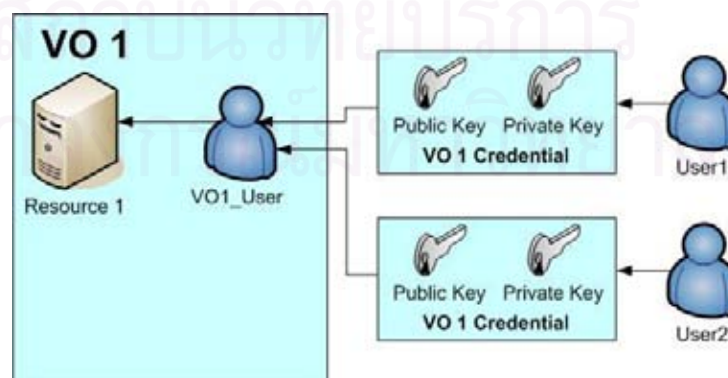


Figure 24: Shared account mapping

Finally, Table 1 compares the benefits of various mapping schemes and characteristics of suitable VOs for each scheme.

Table 1: Mapping scheme benefits comparison

Mapping Scheme	Number of Local Accounts / VO	Key Features	Suitable VO characteristics
One-to-one	Equal to number of total users in VO	<ul style="list-style-type: none"> - Separated user space - Allows usage of existing accounts 	small-size, preserve user data , mostly active user
Account Pool	Adjustable	<ul style="list-style-type: none"> - Separated user space - Adjustable pool size - VO DN authorization 	medium-size, moderate active user, session-based user access
Shared Account	1	<ul style="list-style-type: none"> - Shared user space - VO DN authorization 	large-size, variable active user, session-based user access

3.3.5 Resource Management

A client program must be installed on GHOSTS-managed nodes. This client program is responsible for handling administrative tasks by requesting VO structure information from GHOSTS server and execute operations such as add/remove user, add/remove group and modify “*grid-mapfile*” entries. It also provides an authorization module which is plugged into Globus Toolkit in order to override existing “*grid-mapfile*” mechanism and support other account mapping schemes used by GHOSTS.

3.3.6 VO-aware Grid Portal

A Grid portal provides a platform where a user can easily access Grid functionalities such as job submission and file transfer. By using the portal framework, the portal is also customizable and extensible because a portlet application can be developed and hosted on the portal to serve VO-specific needs.

To allow the portal to concurrently handle the existence multiple VOs, VO-level access control must be applied to various components of the portal. For example, a user can only invoke Grid services on his VO resources; he can only view jobs that are submitted through his currently active VO. Furthermore, a VO portal

which is a customizable page on the portal is also provided for each VO to host its portlets and other applications that are relevant to the operation of a particular VO.

Finally, VO management functionality must be provided via the portal interface so that changes can be made to VOs rapidly. The details of this capability are described in the next chapter.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER IV

DESIGN AND IMPLEMENTATION

The GHOSTS software package consists of three parts. The first part provides an extensible portal environment and Grid functionalities. The second part provides a VO hosting environment. The final part is the applications hosted on the portal. This chapter will give detail explanations of main software components in each part of GHOSTS.

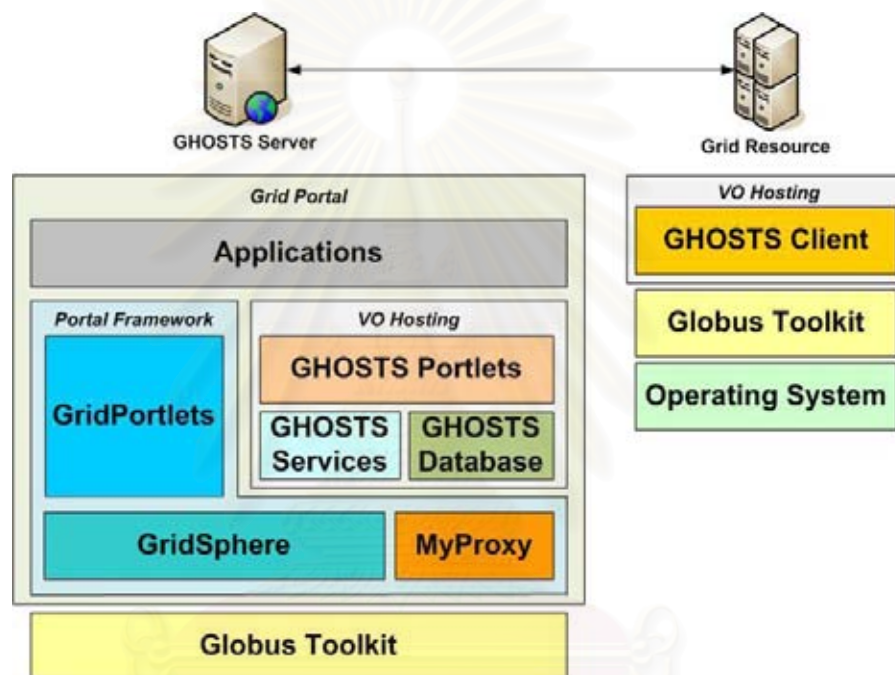


Figure 25: Software packages

4.1 Grid Portal

GHOSTS relied on existing software components to build its portal which are GridSphere, GridPortlets and MyProxy

4.1.1 GridSphere

GridSphere provides an application framework for other components of the system to build upon. We exploit various functionality of GridSphere without modifying its source code, except in one case which we have to port an important feature from GridSphere 3 back to our version which is GridSphere 2.2.8. Other modifications are done through configurations such as custom themes and custom layouts. Figure 26 depicts the system's Grid portal.



Figure 26: First page of the Grid portal

4.1.2 GridPortlets

The GridSphere portal can be considered as a Grid portal only when the GridPortlets is added to the portal. GridPortlets provides both user interface and portlet services which allow us to interact with various Grid resources ranging from job submission and file transfer. However, in order to fully exploit these functionalities we have to use GridPortlets' programming model which is the Action Component Portlet. This means that our developed components that rely on GridPortlets cannot be considered as JSR168-compatible

In order to create a VO-aware Grid portal which also allows user to instantaneously switch between VOs, we have to make various modifications to the core of GridPortlets. Mainly, this is done by creating VO-aware services. These services are adapter classes that have the same interface as their associated services. However, they will present only Grid resources that are available in a user's VO when he lists his available resources. As a result, he can interact only with Grid services on resources that are path of his VO. Additionally, some UI components have to be modified slightly in order to accommodate these changes.

4.1.3 MyProxy

MyProxy is deployed with our portal in order to serve as a credential repository. Instead of delegating short-lived proxy credential to MyProxy, the portal stores both portal-generated and user-supplied credentials permanently in this repository (they are removed when their users leave the system).

4.2 VO Hosting

This part comprises software components that we have developed in order to provide the VO hosting service.

4.2.1 VO Database

VO Database contain VO structure information and other related entities that involve in the operation of VO-aware Grid portal. The relationships between each entity are described using class diagrams. These classes are translated into relational database's tables using Hibernate 2 [33].

4.2.1.1 VO Structure

This package stores VO structure information that reflects various relationships between VO, user and resource. The detail of each class is discussed in the followings.

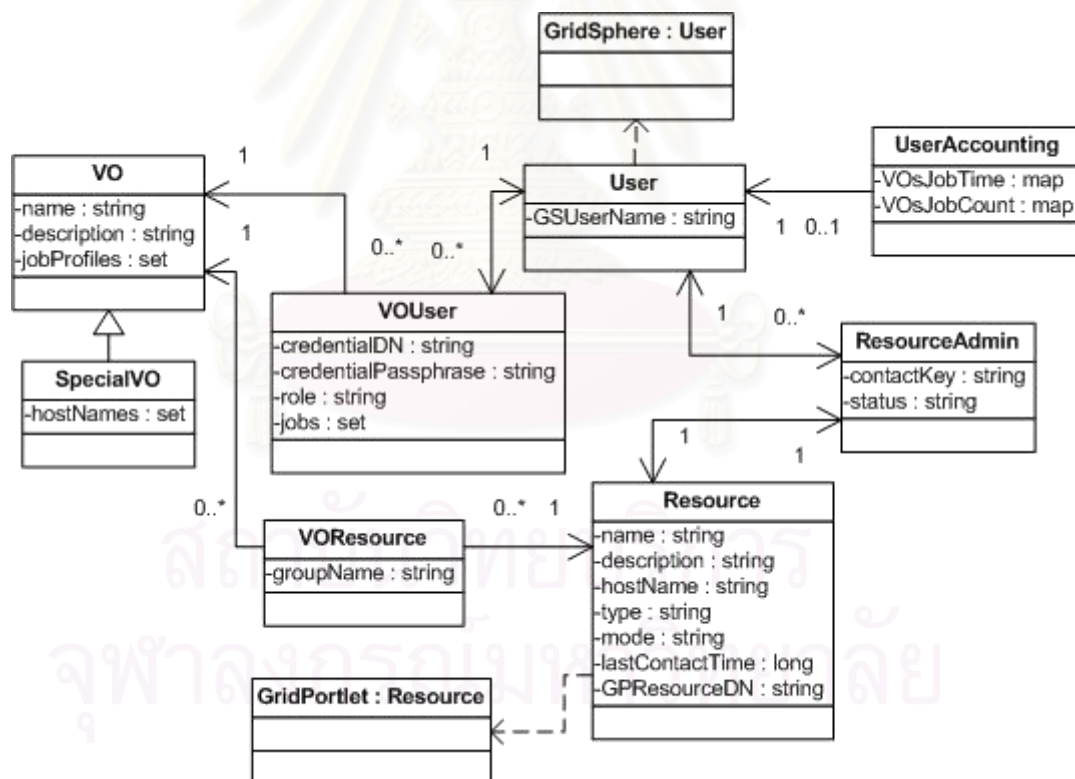


Figure 27: VO structure class diagram

- **VO** — The VO class represents a Normal VO (also referred as GHOSTS-managed VO) which is a type of VOs that can use all functionalities supported by GHOSTS. Additionally, users within a particular VO can

access job profiles that are included in a set of available job profiles maintained by every VO.

- **SpecialVO** — This subclass represents a Special VO which is type of VOs that allows users to use their own credentials instead of GHOSTS-generated credentials. This will allow user to access resources from VO that is not constructed by GHOSTS. As a result, it also maintains a separate list of available host because these hosts are not GHOST-managed resources.
- **Resource** — The Resource class represents hardware resources which are the Grid node machines. There are two types of resource; one for each type of VOs. For GHOSTS-managed resources, they also store the timestamp when the GHOSTS client program last contacts the server. Finally, every resource stores resourceDN which act as a reference to its corresponding GridPortlets' resources because it is required in order to operate GridPortlets' services.
- **VOResource** — The VOResource class is an associative class that represents VO-resource relationship.
- **User** — The User class represents GHOSTS user entity. Similar to the Resource class, it also acts as a reference to GridSphere's User class which represents all user accounts on the portal.
- **VOUser** — This class is also another associative class that represents the VO-user relationships. Information regarding credential and role is also stored in this entity. Additionally, it store a list of job submitted within this VO so that the system can construct a correct view of jobs within his active VO.
- **ResourceAdmin** — When a user becomes a Resource Administrator, it is represented by creating link between a user and a resource. This entity also maintains a contact key which is used to validate resource ownership when a user installs the GHOSTS client program on a Grid node.
- **UserAccounting** — This class maintains usage statistic for each user which includes number and execution time of jobs submitted on each VO. This will be used by the system to create various usage statistic charts.

4.2.1.2 Request

This package consists of various request types that express users' demand for certain actions. The detail of each request is shown in Figure 28 and in the following section.

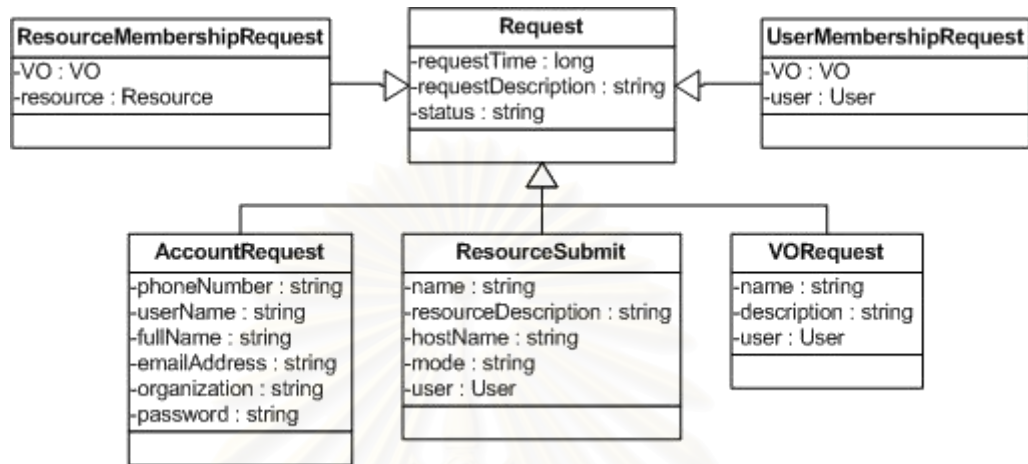


Figure 28: Request Class Diagram

- **Request** — This is a super class of all requests which stores general information such as request creation time.
- **AccountRequest** — This request is generated when a user want to create an account on the GHOSTS' Grid portal. The system uses information provided by this request to create account on the portal and also user entity in the VO Database.
- **ResourceSubmit** — The type of request is generated when a user want to submit his resource to the system. After approval, he will be promoted to the resource administrator role and will be able to manage his resource on the portal.
- **VORequest** — A VORequest is created when a user want to create a new VO. Similar to a ResourceSubmit, if this request is approved, he will be promoted to the VO administrator role and will be able to manages his newly created VO.
- **ResourceMembershipRequest** — Only resource administrators are allowed to create this type of request. When a resource administrator wants to include his resource into a particular VO, he needs to create this request. If it is

granted by the VO administrator, the system will use information provided by this request to create a VOResource entity

- **UserMembershipRequest** — Similar to the ResourceMembershipRequest, this request is created by a user when he wants to join a particular VO. When approved, a VOUser entity is created from information provided by this request.

4.2.2 GHOST Portlets

These portlets provides interface for various functionality of VO hosting in GHOSTS. Each user has access to different portlets depending on his role. Table 2 summarizes the functionality of each portlet and Table 3 provides their access right matrix.

Table 2: Summary of GHOSTS portlets' function

Portlet	Main Function
Account Request	Requesting a GHOSTS account
VO Activation	Activating a VO
VO Membership	Requesting a VO membership
Accounting	Showing usage statistic
VO Administrator	Managing VO (for the VO Administrators)
Resource Management	Managing resources (for the Resource Administrators)
VO Administrative	Creating and managing VOs (for the Administrator)
Resource Administrative	Creating and managing resources (for the Administrator)
User Administrative	Creating and managing users
Request Administrative	Managing various types of requests

Table 3: GHOSTS Portlets' access right matrix

Portlet	Admin	VO Admin	Resource Admin	User	Guest
Account Request					X
VO Activation	-	X	X	X	
VO Membership	-	X	X	X	
Accounting	X	X	X	X	
VO Administrator	-	X	x	x	
Resource Management	X	x	X	x	
VO Administrative	X				
Resource Administrative	X				
User Administrative	X				
Request Administrative	X				

Legends: **X** means full functionality access right
 x means partial functionality access right
 - means no relevant task associated with this portlet

4.2.2.1 Account Request Portlet

This portlet is accessible from the front page of the portal. Any user can request for a GHOSTS account by filling a request form. This request will be sent to administrator and the account will be created after request administrator approval.

Ghosts
Home Registration Tutorial

Account Request Portlet

Account Request
Please use the following form to request for a GHOSTS account.
The administrator will mail back to you as soon as your request is granted.

User Name * : Account name
Full Name * : Full name
Email Address * : Valid email address
Organization : Organization
Password * : Password must be at least 5 characters long
Confirm Password * : Retype password

Comments : Additional comments or specify a target VO

* : Required fields

Figure 29: Account Request Portlet

4.2.2.2 VO Activation Portlet

After login to the system, a user is brought to this portlet because he must choose which VO he wishes to activate. Therefore, this portlet displays a list of VOs in which he currently has memberships. By activating a VO, the system will initialize his proxy credential and will also mark this VO as his active VO. This information is used by other VO-aware portlets in order to limit which resources he is allowed to access.

A user must activate his VO periodically because it is associated with the life time of his proxy credential. The remaining life time of his proxy credential is represented as an activation time. By default, pressing an activate button of any particular VO will activate that VO for seven days. Nevertheless, he can use a list box to specify any values which range from 2 hours to 60 days as his activation time.

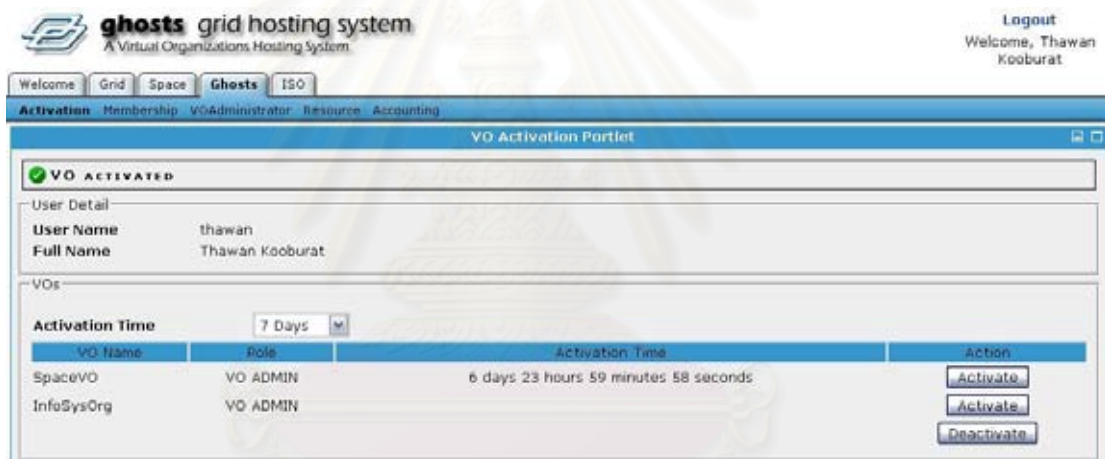


Figure 30: VO Activation Portlet

4.2.2.3 VO Membership Portlet

A user can use this portlet to manage his VO memberships. He can discard any of his current VO membership or request for a new one. Since there are two types of VOs existed in the system, two different interfaces are used for joining each type of VOs.



Figure 31: VO Activation Portlet

For a GHOSTS-managed VO (also called Normal VO), he only needs to specify his intention for joining the VO. Then, a request will be created and the administrators will inspect and grant him the VO membership. After approval, he will be able to activate this VO from the VO Activation Portlet.

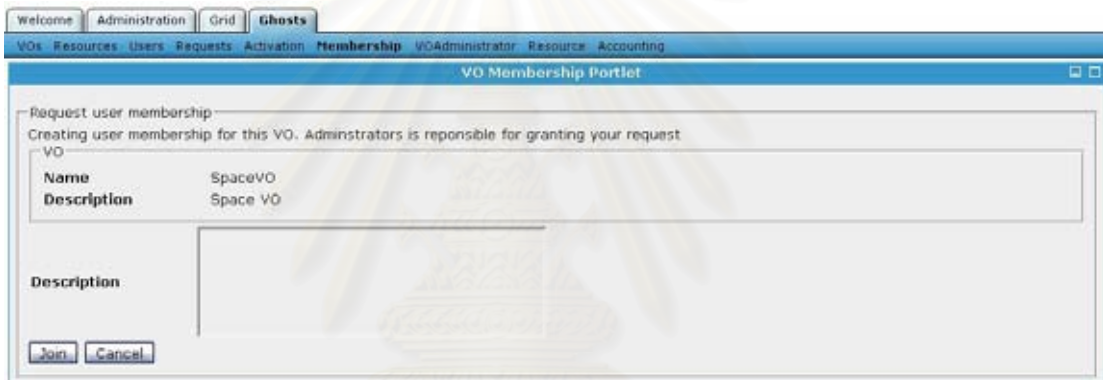


Figure 32: Joining normal VO

For a Special VO, he must supply his credential (public certificate and private key) issued by other CA. He also needs to provide a correct passphrase which is used to access his credential. If it is successful, he will be able to activate this VO immediately because his credential is proof of this VO membership so there is no need to wait for approval.

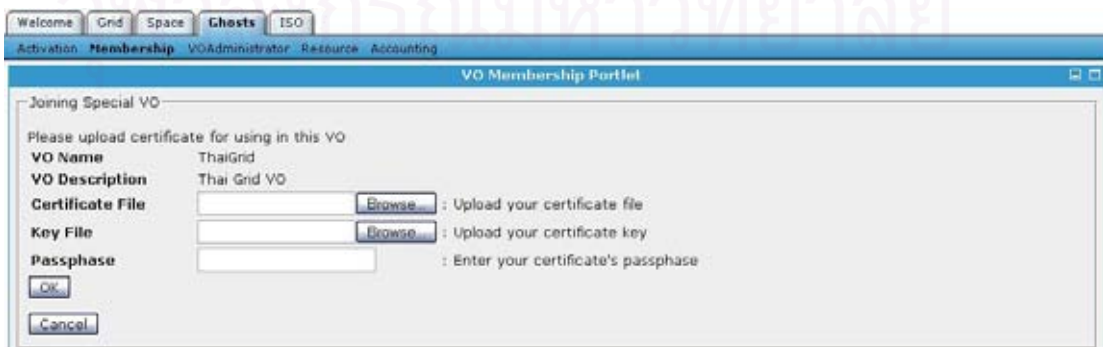


Figure 33: Certificate upload page

4.2.2.4 Accounting Portlet

Accounting Portlet provide site-wide statistic presented in various charts. Three charts have been implemented as an example to shows that the system is capable of correct various types of data. Firstly, the User Distribution chart depicts number of VO users in each VO using pie chart. Secondly, the CPU Usage chart shows how many CPU hours were consumed by all completed jobs for each user. A stacked bar chart is used to show the value distributed in each VO. Finally, the number of jobs submitted in each VO by top users is also provides in the Job Submission Statistic chart using a stacked bar chart. Additionally, every chart rendering is performed by using JFreeChart [34] library which is included in the GridSphere framework.

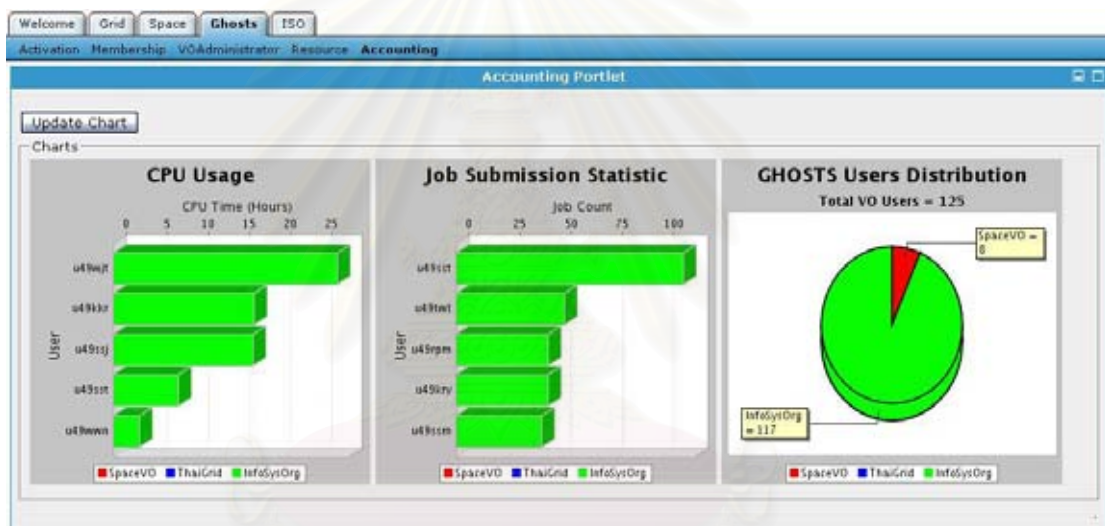


Figure 34: Accounting Portlet

4.2.2.5 VO Administrator Portlet

This is a strip-down version of VO Administrative portlet. It provides administrative functionality for VO administrators to manage their VO. However, some functionality is not presented in order to preserve how each role must interact with each other. For example, a VO administrator can add resource to his VO only when he receives a request from a resource administrator.



Figure 35: VO Administrator Portlet

For a normal user which does not have VO administrator role associated to any VO, he can request to create a new VO. If it is approved by the administrator, he will be granted a VO administrator role of his requested VO.

The screenshot shows the 'VO Creation Request' form. At the top, there are navigation tabs: 'Welcome', 'Grid', 'Space', 'Ghosts', and 'ISO'. Below these are sub-tabs: 'Activation', 'Membership', 'VOAdministrator', 'Resource', and 'Accounting'. The main content area is titled 'VO Administrator Portlet' and contains the following text:

VO Creation Request
Request for a new VO
If granted, you will become a VO Admin of this VO GHOSTS Administrator will inspect this request.

The form includes the following fields:

- VO Name ***: A text input field with a label ': Short VO name'.
- VO Description ***: A text input field with a label ': Long VO name'.
- Request Description**: A larger text area with a label ': Additional comments'.

At the bottom left, there is a legend: '* : Required fields'. At the bottom right, there are two buttons: 'Request' and 'Cancel'.

Figure 36: VO creation request page

4.2.2.6 Resource Management Portlet

Any user who which to add his resource to the portal can use this portlet to submit his resource information. This information is used by the administrator is add this resource to the portal. Moreover, he will have to install GHOST client program in order to allow the resource to join GHOSTS-managed VO. When all necessary steps are taken and the resource is added to the portal, he will become a resource administrator and be able to use other functionality of this portlet. Moreover, he will need to retrieve unique contact key from the portal to put it in the GHOSTS client program configuration. This contact key is used by the portal to validate resource ownership.



Figure 37: Resource Management Portlet

A resource administrator can have one or more resource under his administration. By using this portlet, he can inspect his resource information such as when is the last time that GHOSTS client program on his machine contract the portal. Additionally, he will be able to submit his resource to any VO on the portal and his request will be approved by the administrators.

4.2.2.7 VO Administrative Portlet

The VO Administrative Portlet allows the administrator to manage aspects of a VO. From the main page, the administrator can create/remove a VO. When creating a VO, the administrator must choose between two types of VOs (GHOSTS-managed or Special VO) which cannot be changed later. Other information such as users and resource can be filled after a VO is created. By selecting an edit label associated with each VO, the administrator will be able to manage various components of a VO which are discussed in the followings.

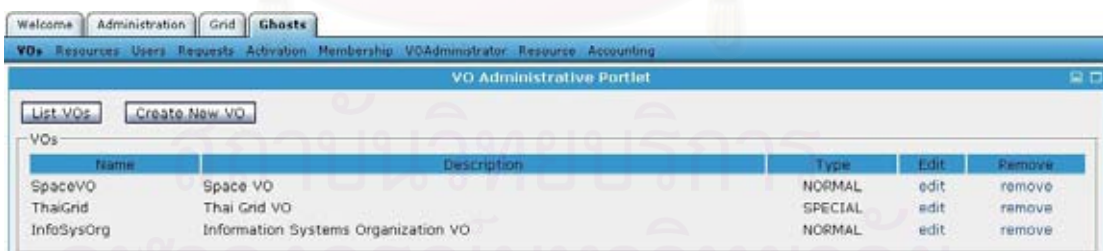


Figure 38: VO Administrative Portlet

Users

The administrator can add or remove a specific user to the VO by selecting from a list of GHOSTS user. Additionally, he can use the batch add feature which is suitable when adding large number of users.

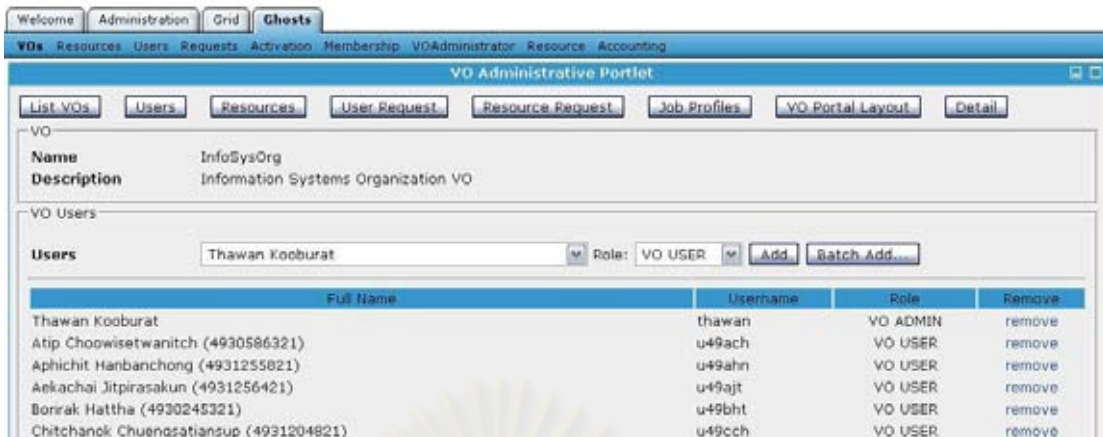


Figure 39: VO user management

When using the batch add feature, the administrator must specify users' information in XML. After that, the system will ask for his confirmation by presenting a list of users processed from the XML. Using this method, he can add new users to the VO directly because the system automatically add these user to GridSphere and GHOSTS

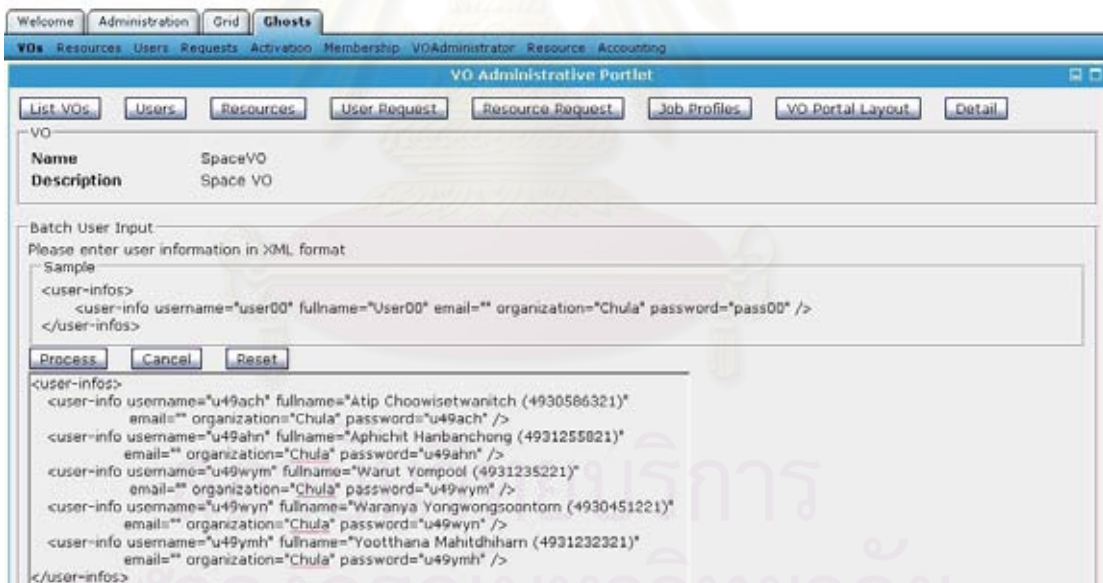


Figure 40: Batch user addition feature

Moreover, VO membership requests made by users are also presented when selecting the user request button. The user is automatically added to this VO when his request is granted by the administrator.

Resources

Managing VO resources is quite similar to managing VO users. The administrator can add/remove resource from the list of GHOSTS resources. He can also grant resource membership by inspecting a resource request page.

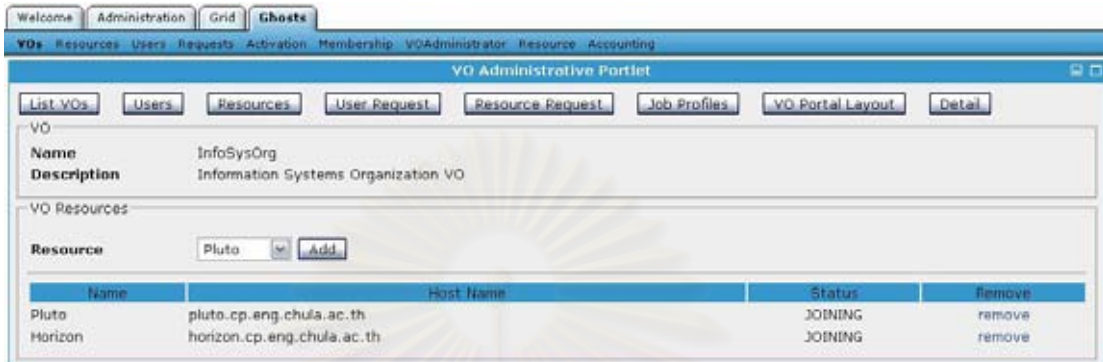


Figure 41: VO resource management

Job Profiles

The job submission functionality of GridPortlets introduces the concept of Job Profile. A job profile is a sequence of web pages that helps a user prepares his input for a specific job, thus serving as an interface of a specific application on the Grid. When submitting a job through the job submission portlet, a user must select a type of job by selecting a job profile.

This page allows the administrator to configure which job profiles are available on this VO. Every user has access to job submission functionality but the types of jobs that they can submit are limited by controlling the access to job profiles.

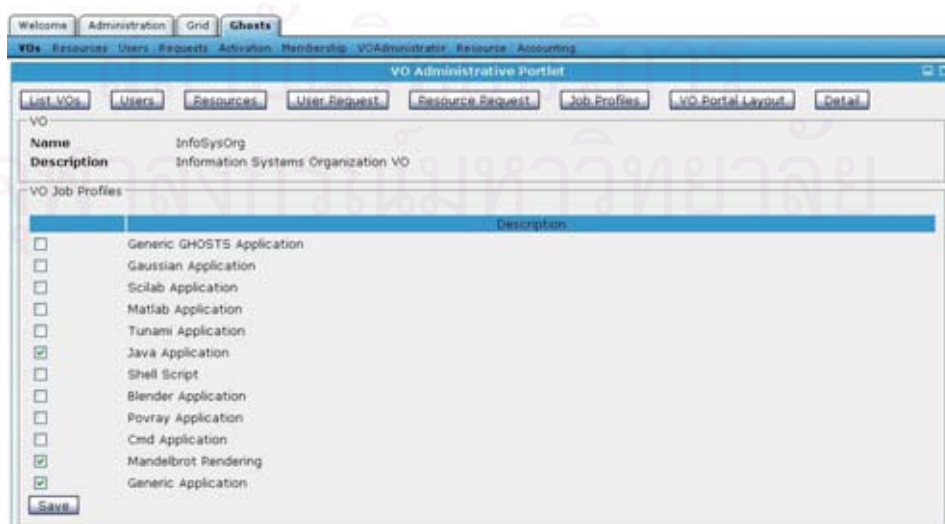


Figure 42: VO job profile select page

VO Portal

A VO portal is a page on a portal associated to a specific VO that can be used to host various portlets that are relevant to the operation of the VO. This feature is implemented on by creating a GridSphere's portlet group for each VO. Each VO portal will result in a new tab on the main page and user is allowed to access it only when he has that VO membership.

The administrator can use this page to specify the look of VO portal by using GridSphere's Portlet Layout Framework to describe the page. Each VO portal can have many sub-tabs and each sub-tab can host many portlets.

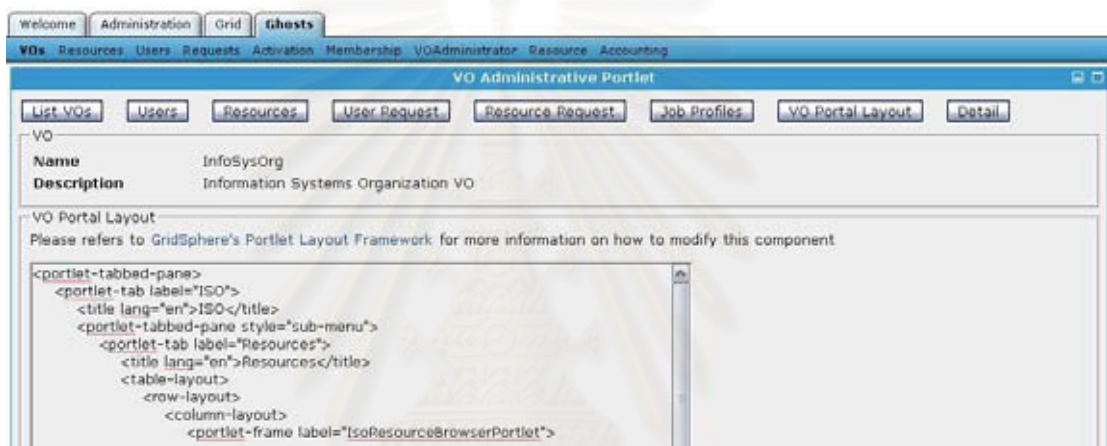


Figure 43: VO portal layout edit page

VO Detail

Only VO description is allows to be changed after the VO is created. This administrator can use this page to change a VO description

Hosts

This page is only available for Special VO type. The administrator can use his page to configure which hosts are allowed to access by this VO. This functionality replaces all user and resource management functionality which are not available in Special VO.

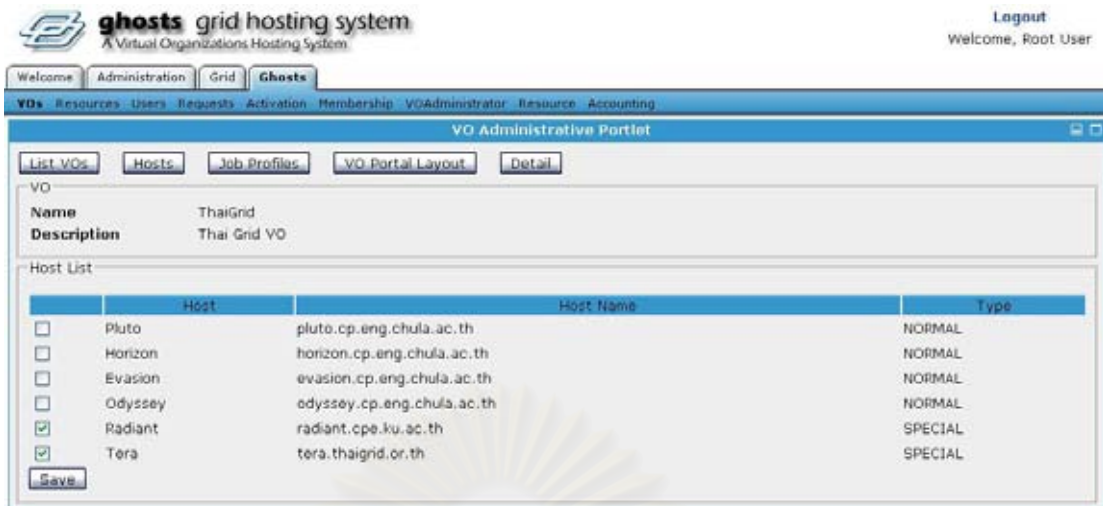


Figure 44: Host select page

4.2.2.8 Resource Administrative Portlet

GridPortlets uses the Resource Registry Portlet to maintain a site-wide list of available resources which can be modified through a XML configuration file. GHOSTS extends this model such that the GHOSTS portal has multiple views of resources associated with various VOs. This is done by requiring that all resources are added to the Resource Registry Portlet first, and then their corresponding entities which are used to store VO – resource relationships must be created on the VO database.

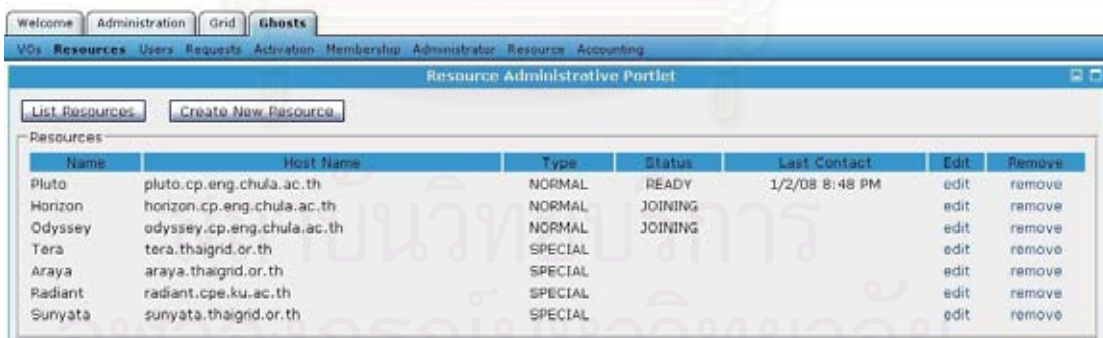


Figure 45: Resource Administrative Portlet

The administrator can use this portlet to add GridPortlets' resources to the VO database which will further allow them to be added to other VOs on the portal. After that, the administrator can manually choose a specific user to become a resource administrator of this resource. This will enable this user to access resource administrator functionality on this portal.

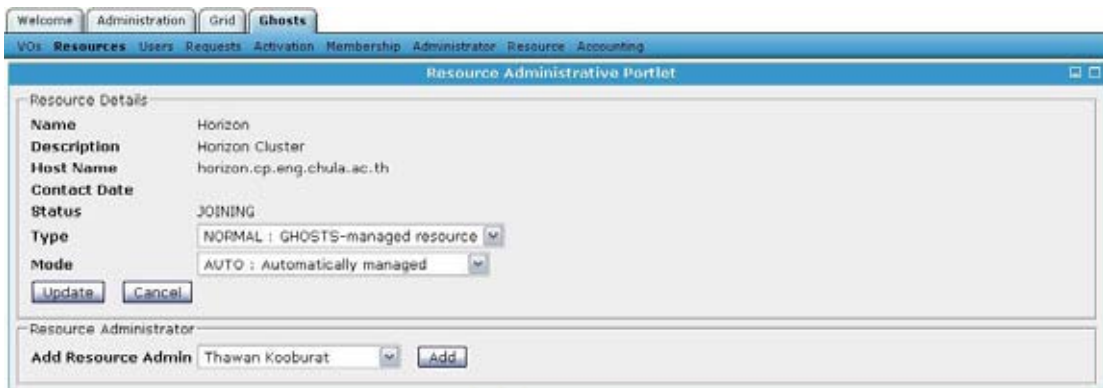


Figure 46: Assigning a resource administrator to a resource

4.2.2.9 User Administrative Portlet

Like how GHOSTS work with GridPortlets' resources, GHOSTS use the same mechanism to operate with GridSphere's users. Thus, GHOSTS maintains entities that are associated with GridSphere's user accounts.

This portlet allows the administrators to import GridSphere's user accounts to GHOSTS. This means that, in order to generate an account on the system, the administrator must create an account on GridSphere before adding it to GHOSTS. However, these steps are handled automatically if an account is generated through batch add or account request functionality.

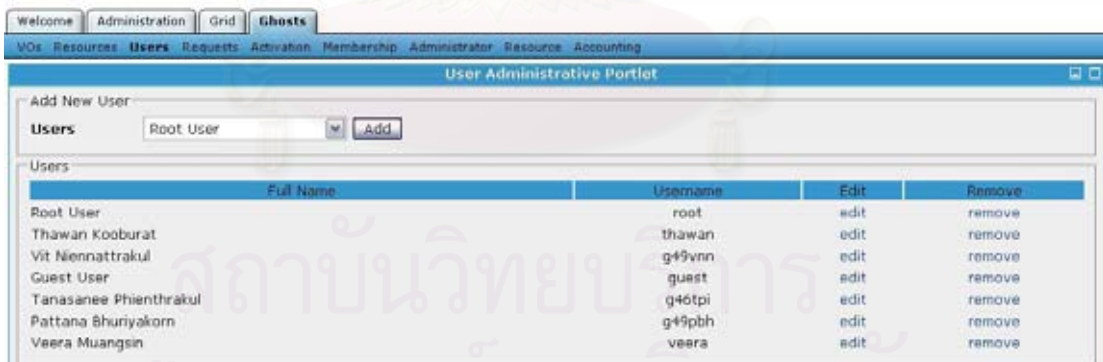


Figure 47: User Administrative Portlet

4.2.2.10 Request Administrative Portlet

All request-related pages have the same basic features where the administrator can inspect the information of the request by clicking on its name or take various actions on the request which are grant, consider and remove. There are three types of requests that are handled by this portlet and they are discussed as the followings.



Figure 48: Request Administrative Portlet

Firstly, the account request is generated when a user want to have an account on the GHOSTS system. When the administrator chooses to grant a particular account request, the system will automatically generate a GridSphere' account and add it GHOSTS. The result can be seen immediately on the User Administrative Portlet.

Secondly, the resource submission request is created by using the Resource Administrator Portlets when a user wants to submit his resource to GHOSTS. If the administrator approves a request, he must add this resource manually to GridPortlets' Resource Registry Portlets and also to GHOSTS' Resource Administrative Portlets.

Finally, the VO request is produced from the VO Administrator Portlets when a user wants to create his own VO. When a VO request is granted, the system will automatically generated a new VO and add this user to the VO as a VO administrator.

4.2.3 GHOSTS Services

GHOSTS services are the application logic of the system, they can be divided in to three categories which are described in the followings

4.2.3.1 GHOSTS Portlet Services

Most application logics of the GHOSTS system are encapsulated in these portlets. This paradigm follows the architectural model used by GridSphere and GridPortlets. The main benefit of this model is that it is easier to divide application logics from presentation logics. Moreover, other portlets can easily interact with GHOSTS's core system by invoking these portlet services. In the following section, we present the information regarding these services.

- **VODatabaseService** — This service encapsulate the GridSphere's PersistenceManagerRdbms which is responsible for retrieving and persisting objects in the database by relying on Hibernate [33]. It is used internally to by GHOSTS' portlets and services when they want to perform database transactions.
- **VORegistryService** — It is a primary service that handles application logics regarding VO management. It provides abstraction of various processes in form of VO operations such as create/remove a VO, add/remove a user to a VO and active a VO for a user. For example, adding a use to a VO involves creating a VO-user relationship in the database, creating a user credential, and etc.
- **AccountingChartService** — This service is responsible for generating various usage statistic charts.
- **AccountingService** — Updating usage statistic can be done by calling methods provided by this service. For example, a job listener is added to every job so that it will update user's total CPU usage time through this service when the job is finished.
- **CredentialService** — This service encapsulate various Globus's command-line interfaces that are used to generate and store credentials to MyProxy repository. It uses GHOSTS CA to generate user credentials which have two years lifetime and their proxy credentials can be instantiated for up to 60 days.
- **CredentialManager** — It is responsible for handling necessary operations related to credential management when a VO operations are performed. For example, when generating an account, it must create user credential by using CredentialService and also store this credential's information to credential-related services in GridPortlets.
- **GroupManager** — The VO portal functionality is achieved by using GridSphere's portlet group. This service handles various operations related to this area when VO operation are performed such as creating a portlet group that is associated with a VO.

4.2.3.2 VO-aware Portlet Services

These VO-aware services are adapter classes that implement or extend existing portlets service interfaces of GridPortlets. In most cases, these services operate by checking their query result returned from their counterpart services with the VO database. This will allow them to return their result based on a user's active VO. The details of each service are discussed in the followings.

- **VOJobProfileRegistryService** — Its main purpose is to return a list of available job profiles according to a user's current VO. It implements the JobProfileRegistryService interface.
- **VOJobSubmissionService** — This service which implements the GridPortlets' JobSubmissionService interface allows a user to see only a list of jobs that are submitted with his current VO. In order to achieve this, the service maintains job – VO relationships of every user in the VO database so that this information can be used by its query method. Additionally, it also records job submission statistics which are used to various charts.
- **VOResourceRegistryService** — This VO-aware service extends the ResourceRegistryService which is the main service of GridPortlets that maintain a list of available resources used by other portlets. It is called in various occasions such as when listing available hosts and listing available services. Finally, slight modifications have to be made to various portlets such as File Transfer, Job Submission and Resource Browser in order to replace their existing service with the VOResourceRegistryService.

4.2.3.3 Web Service

A web service based on Axis 2 web service engine [35] is used by the portal to communicate with its client programs. This web service publishes VO structure information described in XML which is used by the client programs to perform VO management on their resources. It will publish the information to those clients that present correct contact key to the service. This prevents malicious program from retrieving this important information. Additionally, all communication is encrypted and signed using WS-Security. Finally, it also keeps track of the status of these resources in order to present it to the resource administrators.

4.2.4 GHOSTS Client

GHOSTS client is software package that must be installed on every GHOSTS-managed resource. Its main purpose is to handle how grid credentials should be mapped to local accounts on each resource. GHOSTS supports three modes of account mapping which are one-to-one, account pool and shared account. These modes are achieved by using two different components and they are discussed in the followings.

4.2.4.1 GHOSTS Web Service Client

One-to-one account mapping relies on mapping each grid credential to a separate local account. This means that the client must be able to generate local accounts and know credentials' DNs in order to correctly modify the "grid-mapfile"

As a result, this component acts as a web service client which pulls VO structure information via an encrypted channel from the GHOSTS' web service. This program compares retrieved information with its local version in order to create a list of account management operations such as create/remove group and add/remove user. Figure 49 shows an example scenario where an existing user must be removed and a new VO together with its user must be added.

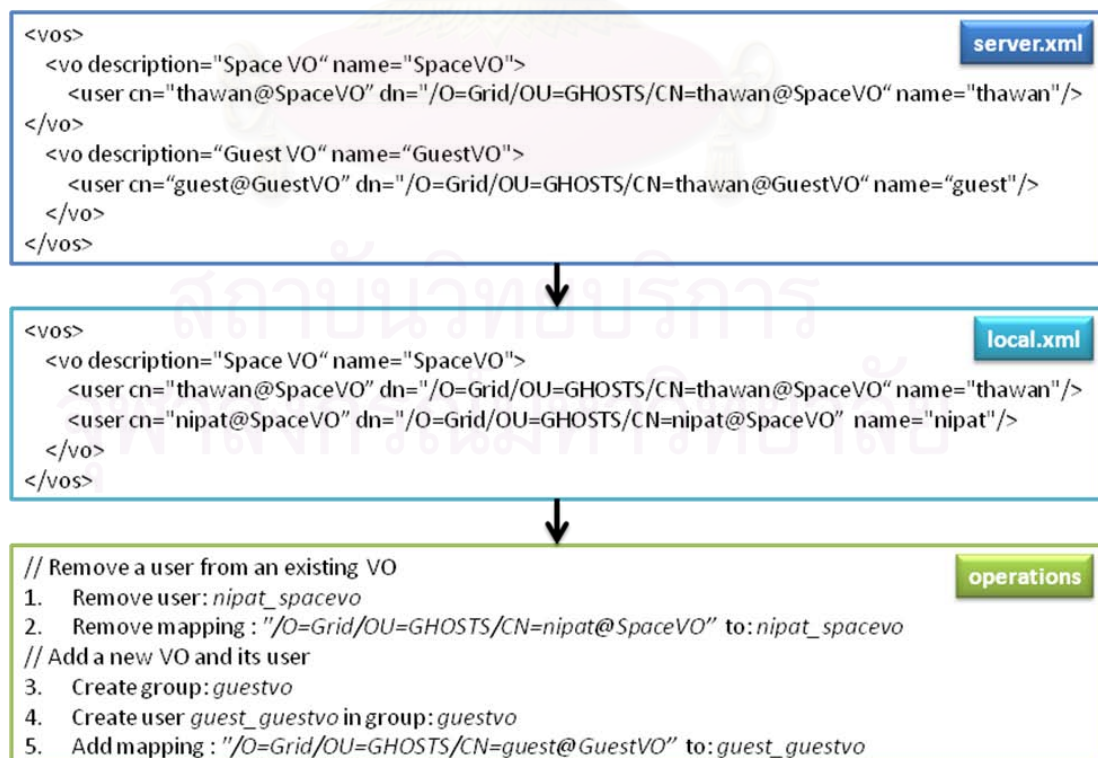


Figure 49: Processing VO structure information

Executing VO operations such as add/remove user may be different for each site. As a result, the client program executes these operations through “driver” which can be developed by using provided interface and plugs into the program by modifying the configuration file.

4.2.4.2 VO Map Authorization Module

The rest of the account mapping modes cannot be achieved efficiently without introduce a new authorization module which plugs directly into Globus. The pre-web services components of Globus toolkit support the development of authorization callouts [36] which can be used to perform custom authorization mechanism at various points.

VOMap module extends the basic Gridmap callout’s mechanism by introducing the “*vomap-file*”. Instead of authorizing users based on their unique DNs, it authorizes users based on their VO’s DNs which is one level higher. Based on this concept, VOMap module introduces two authorization mechanisms which are account pool and shared account.

```
# Comments
<VODN_PREFIX>                share_account
<VODN_PREFIX>*<VODN_SUFFIX> account (, account)* [lease_time]
```

Figure 50: Format of “*vomap-file*”

```
/O=Grid/OU=Thaigrid           thaigrid
/O=Grid/OU=GHOSTS/CN=*@SpaceVO space1, space2
/O=Grid/OU=CPGrid             cp1, cp2 [3600]
```

Figure 51: Example “*vomap-file*” entries

Figure 51 depicts two entries of “*vomap-file*” which shows it can be used to control VOMap module. The first entry illustrates an example of shared account mapping. The left portion of the first entry is used as a prefix to match all incoming credential DNs. If it is matched, then they will be mapping to the user account specified on the right portion of the entry.

For the second entry, this is an example of using account pool mapping. The left portion of the entry is used to match all incoming DNs but with more options. The star character is used to divide DN into prefix and suffix. Thus, a valid credential DN of this VO must match both parts. Then, a list of user accounts on the right portion is

used to construct a pool of accounts for each VO entry. This list can be modified at any time and the module will adjust a corresponding pool automatically.

VOMap module maintains “*accounts.leases*” file which stores lease information of every pools. Mapping mechanism is performed by searching for existing local account of the incoming credential. If no account found, an available account is searched by treating the account pool like a linear probing hash table using credential DN as a key. This means that the module is trying its best to map credential to its “default” account. The final result is that, a user is likely to use this local account whenever he returns to this resource. After the target account is found, it is reserved for that credential for a period time specified in a bracket. The third entry shows an account pool where 3600 seconds (one hour) is used as a lease period.

Finally, the target account returned by this module is used to map credential to local account in every pre-web service components of Globus such as GRAM and GridFTP. Additionally, if an incoming credential does not match any VO entry, the default “*grid-mapfile*” mechanism is used to perform the authorization decision. Moreover, every mapping decision is logged into */etc/vomap.log* so that all user access is auditable.

4.3 Applications

The combined platform of GridSphere and GridPortlets provides a framework for developing a web-based Grid application. GHOSTS support this idea by allowing each VO to host its custom-developed application on its VO portal. However, most of VO special requirements fall into developing a custom job submission that act as an interface to a specific application. These requirements can be achieved more efficiently by developing custom job profiles which are plugged into GridPortlets’ Job Submission Portlet.

From an implementation point of view, a job profile is a partial portlet which focuses on preparing various parameters of a certain job type, whereas other functionalities are provided by the Job Submission Portlet. This means that a job profile can take all the advantages of being a Grid portlet but takes lesser to develop. However, the job profile is only suitable for acting as an interface to command-line program with no interactive feature.

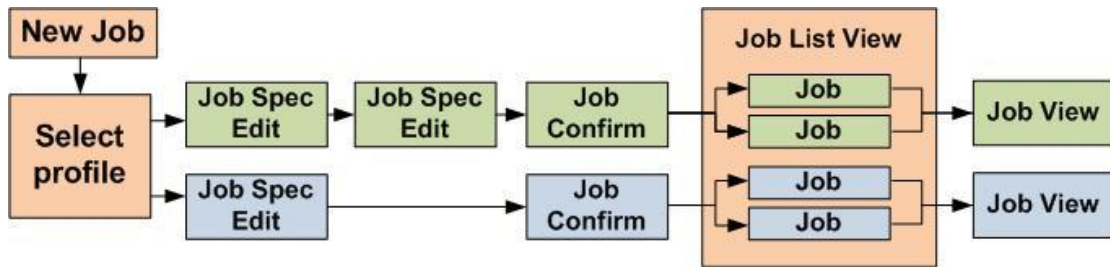


Figure 52: Job profile sequence

GHOSTS has develop various job profiles in order to answer specific requirement of each VO. The detail of each job profile is discussed in the followings.

4.3.1 General Purpose Job Profiles

This group of job profiles provides general purpose tools for using the job submission functionality.

4.3.1.1 Generic GHOSTS

This job profile is a modified version of Generic Job Profile provided by GridPortlets. This job profile exposes all parameters of the job submission functionality and it can be used to submit any types of jobs support by Globus Toolkit's interface. Finally, the modification is made in order to add support to SGE and P2P job scheduler.

4.3.1.2 Java Program

This job profile allows users to submit their Java programs to run on the Grid. Users can upload their programs achieved in Java' JAR format and submit them to any machines in their VOs.

4.3.1.3 Shell Script

This job profile allows users to use a shell script as a mean to submit their jobs. Users can create, modify and upload their shell script though the job profile's interface. Additionally, it also supports various type of shell script such as bash, csh, and ksh.

4.3.2 Showcase Job Profiles

These job profiles are developed in order to introduce users to Grid Computing. The usability and easy-to-understand output are primary focus of job

profiles within this group. As a result, these job profiles perform various type of image rendering and allow users to download the output image directly from their result page.

4.3.2.1 Mandelbrot

Mandelbrot Set is a collection of points on complex plane that create beautiful fractal images. A user can specify the area of interest by drawing a box on reference Mandelbrot image. This will be translated into coordinates by the JavaScript which will be used as parameters for the Mandelbrot program written in Java. Finally, when the job is finished, he can download and view the result image.

The screenshot shows a web-based job submission interface titled "Job Submission Portlet". It contains several sections for configuring a Mandelbrot set rendering job:

- Job Parameters:**
 - Job Description * (text input)
 - Host Name * (dropdown menu, value: pluto.cp.eng.chula.ac.th)
 - Scheduler Name * (dropdown menu, value: sge)
 - Standard Output * (radio buttons for Web or File)
- Mandelbrot Parameters:**
 - Gradient (dropdown menu, value: Adjusted Exponential Gradient)
 - Iteration * (text input, value: 1500)
- Coordinates and Dimension:**
 - Image Size * (Width: 1220, Height: 700)
 - X Range * (Min: -0.349, Max: -0.15075000000000016)
 - Y Range * (Min: 0.6266666666666667, Max: 0.7433333333333333)
- Parameter Chooser:**
 - Zoom * (text input, value: 20)

At the bottom of the interface is a preview window showing a colorful Mandelbrot set fractal image with a small white box indicating the selected area of interest.

Figure 53: Mandelbrot Job Profile

4.3.2.2 Blender

Blender [37] is a 3D rendering program which is installed on every Grid node in ThaiGrid. This job profile relies on ThaiGrid's Blender test script to render 3D image that contain hostname of the execution machine. A user will be able to download and view image directly from the job profile when the job is finished.

4.3.2.3 Povray

Povray [38] is also another 3D rendering program like Blender. The job profile provides a default 3D animation that a user can specify which frame range to render. The result images are packed into single archived and available for user to download.

4.3.3 Scientific Applications Job Profiles

This collection of job profiles serves as interfaces to scientific applications required by certain VOs.

4.3.3.1 Scilab

Scilab [39] is open-source scientific software for numerical computation. This job profile allows user to submit a job to Scilab via a script. A user can upload, create and modify the script through the job profile interface. Finally, the result of the computation is shown directly on the result page when the job is finished.

Figure 54: Selecting input file of Scilab



Figure 55: Input file is presented before submitting a job

4.3.3.2 Matlab

This job profile has the same set of functionalities as the Scilab Job Profile, but it uses Matlab [40] instead of Scilab as a target program.

4.3.3.3 Gaussian

Gaussian [41] is a computation chemistry program which can be used to evaluate a compound structure. This program allows a user to submit his input script via the job profile interface. Like the Scilab Job Profile, a user can upload a script and via the result computation directly from the portal.

4.3.3.4 Tunami

Tunami (Tohoku University's Numerical Analysis Model for Investigation of near field tsunamis) is a Tsunami simulation program written in FORTRAN. In the process of preparing a job, a user must create a job directory that contains various input files and source code used by the program. This job profile automatically creates a job directory and asks him to upload all input files according to their categories. When the job is finished, all outputs are achieved and stored in a special location (public_html) which will allow him to download the file directly from the execution machine.

4.4 Additional Software

In our production environment, GHOSTS is deployed along side other software which are not part of GHOSTS software package. These software components provide additional functionality that is important for the operation of Grid portal in production environment. Here is the list of software that we used

4.4.1 Google Analytics

Google Analytics [42] is used to provide extensive information about site statistics. GridSphere does provide basic site statistics but it is not adequate for the operation of production environment. Information from Google Analytics report allow administrator to track site usage and other interesting metrics.

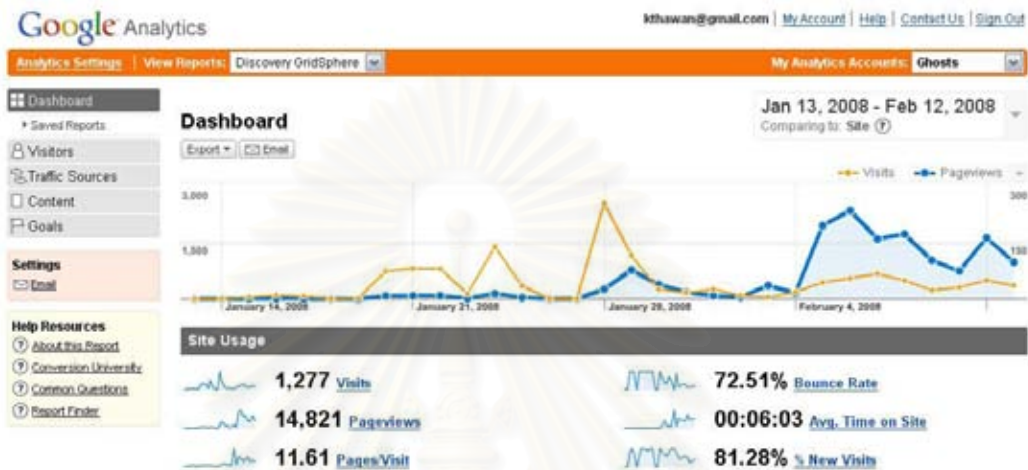


Figure 56: Google Analytics' report

4.4.2 Ganglia Monitoring System

Ganglia Monitoring System [43] is used as a main source for users to monitoring Grid resources. It provides extensive and real-time information which cannot be achieved by using GridPortlets' built-in resources monitoring. However, Ganglia is not a part of GHOSTS software package, so it does not inherited the VO concept used in GHOSTS. In current deployment which is shown in Figure 57, Ganglia's is used to monitor all Grid resource within the department.

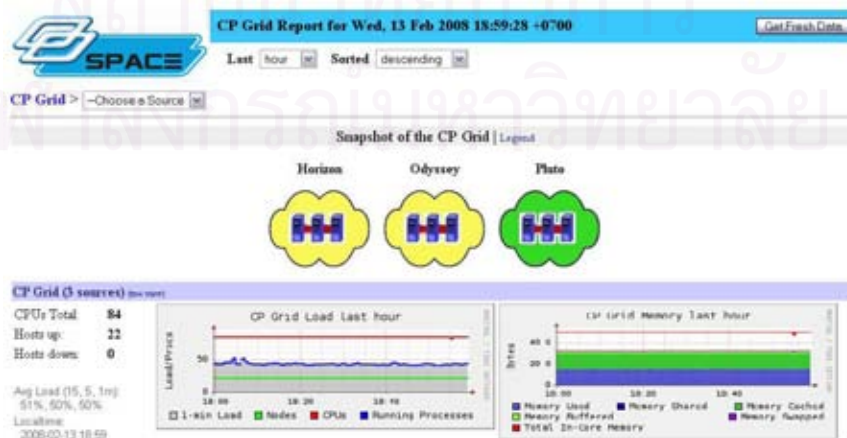


Figure 57: Ganglia Monitoring System

CHAPTER V

EVALUATION

5.1 System Features

This section summarizes system features according to their domains. Table 4 lists various features and their benefits.

Table 4: Features summary

Domain	Feature	Benefits
Architecture	Centralized server	Reduce overhead of dynamic VOs
VO Model	VO hosting	Allow VO to be hosted on the server with minimum effort
	Overlapping resources	Allow multiple VOs to coexist and share resources
	One credential per one VO	Compatible with existing Grid infrastructure and also easy to deploy
User Management	Roles	Various actors in each VO can participate in their VO administration process
	Requests	Administration task is distributed to many persons
Credential Management	Built-in certificate authority	Allow GHOSTS to issue user credentials by its own
	Built-in certificate repository	Securely store user credentials on the server
	Automatic credential management	Reduce the complexity of credential management for users
	Credential uploading	Allow users to use GHOSTS portal to interact with non GHOSTS-managed resources
Access Control	One-to-one account mapping	Automatically generate a new local account or using existing one
	Account pool	Reduce the number of local accounts for each VO
	Shared account	Support a large number of concurrent users
Resource Management	Web service client	Help perform account management on Grid nodes
	VO authorization	Enable authentication and authorization based on VO DN

Grid Portal	VO-aware	Allow users to efficiently use multiple VOs
	VO management	Allow administrators to manage VO from portal interfaces
	VO Portal	Allow VO to have its custom page on the portal

5.2 Feature Comparison

Table 5 compares features found in GHOSTS with other related work from section 2.2.3

Table 5: Feature comparison

Domain	Feature	GHOSTS	GAMA	GENIUS
VO	Multiple VOs	X		X
	Overlapping resources	X		X
	VO hosting	X		x
User management	Roles	X		x
	Request	X	x	x
Credential management	Built-in CA	X	X	
	Built-in repository	X	X	x
	Automatic credential	X	x	x
	Credential uploading	X		x
Access control	Mapping schemes	X		<u>X</u>
	VO DN Authorizaton	X		
	Access control granularity	x		<u>X</u>
Grid Portal	Job profiles	X	x	X
	VO portal	X		
	VO management	X	x	x
	Usage statistic	X		
	Application framework	Standard	Standard	Proprietary

Legends: **X** means full functionality
X means full functionality but provided by its infrastructure
x means partial functionality

5.3 Security Considerations

5.3.1 Server Security

A GHOSTS server is potentially a single point of failure and subjected to malicious attacks, like a MyProxy, VOMS or CAS server. However, periodic database backup and standard server replication techniques can be used to address this concern. Every credential stored in the MyProxy server is encrypted with a randomly-generated passphrase and only the portal is allowed to access these credentials. These credentials are also deleted when their users leave VOs or the system. Finally, the machine that runs GHOSTS server has no end-user access. This reduces the chance of the server being compromised.

5.3.2 Client Program Security

The idea of installing a client program that has a root privilege and handles user accounts on a resource's machine may make the administrator feel uncomfortable. However, similar mechanisms can be found in many projects such as `mkgridmap` in VOM and `GridMgr` [5]. Moreover we address this issue with many measures. Firstly, as mentioned earlier, it is the client who initiates the process of VO operations, not the server, and the connection is secured by using message encryption and digital signature. Secondly, generated accounts have no password set, so they can be accessed through Grid credentials only. Finally, resource administrators have options to monitor its operations or inspect the generated log file.

For the authorization module, the idea of authorizing users based on their VO DN is similar to the mechanism used by VUS. This is more scalable solution than dynamically modifying "*grid-mapfile*" on every time the VO structure is changed. Moreover, the components is extensive tested against memory leaks by using Valgrind [44] and deployed by using Globus-supplied package.

5.4 Reliability and Scalability Considerations

Multiple GHOSTS servers can be deployed concurrently. This feature is made possible by allowing GHOSTS client software to support multiple servers and maintain a separate state for each server. This feature is used during one of the case studies to reduce server's load by hosting a VO on a separate server. However, each server has no knowledge of each other, so they cannot seamlessly share their

resources. Nevertheless, the mechanism proves that GHOSTS can address certain issues regarding the scalability. Moreover, more work could be done in extending this idea such as using a shared database for multiple servers so that other scalability issues can be solved.

For the reliability issues, a routine database and file system backup can be used to solve a server failure problem. Moreover, the concept of multiple GHOSTS server using a shared database, if developed, can be used to address many reliability issues.

5.5 Usage Scenarios

In this section, we present some of the possible usage scenarios that show how various features of the system are utilized. Other scenarios and their diagrams are presented in Appendix B.

5.5.1 ThaiGrid VO Creation

In this scenario, we describe how GHOSTS can be used to support existing members of ThaiGrid.

Firstly, GHOSTS Administrator must construct ThaiGrid VO on the portal and add all ThaiGrid machines as its resource (see section 4.2.2.8). He can also prepare the VO portal so that it hosts applications that are relevant to the operation of ThaiGrid users.

Secondly, importing ThaiGrid users can be done by using batch user addition (see section 4.2.2.7). Users' information can easily be converted to GHOSTS-compatible XML by using program such as Microsoft Excel. GHOSTS Administrator can use this XML to add all users to ThaiGrid VO with minimum effort.

Thirdly, GHOSTS client programs and GHOSTS CA's certificate must be installed on every node. The first part is required because shared account mapping will be used in order to follow the method adopted by ThaiGrid. However, additional benefit of using GHOSTS *"vo-mapfile"* is that a new user can access all ThaiGrid resources instantly without waiting for *"grid-mapfile"* to be updated.

Finally, when users received their account information and password, they can login to GHOSTS and start accessing ThaiGrid resources. In fact, this is simpler and less error prone than the current practice of ThaiGrid.

5.5.2 Department VO Deployment

In this section, we discussed how GHOSTS is deployed to support researches with the department.

Firstly, the *SpaceVO* is constructed and three main production clusters are added as its resources. This VO is a GHOSTS-managed VO type and use one-to-one account mapping for resource-level access control.

Secondly, users' GHOSTS accounts are mapped to newly generated accounts unless they have existing local UNIX account on each cluster. The GHOSTS client program will help administration perform this process.

Thirdly, various job profiles that allow users to submit their jobs to applications such as Matlab, Scilab and Java is added to this VO. Moreover, its VO portal also contains tutorials, user manual and a link to department's Ganglia monitoring page.

5.6 Case Studies

5.6.1 ThaiGrid Integration

The goal of this case study is to assess the initial release of GHOSTS system and evaluate the service quality of the system when using ThaiGrid resources.

Firstly, a Special VO called *ThaiGrid* is created and a list of ThaiGrid's machines is registered as its resources. Figure 58 depict the testing environment. In order to use this VO, a user must upload his ThaiGrid-issued certificate to the credential repository.

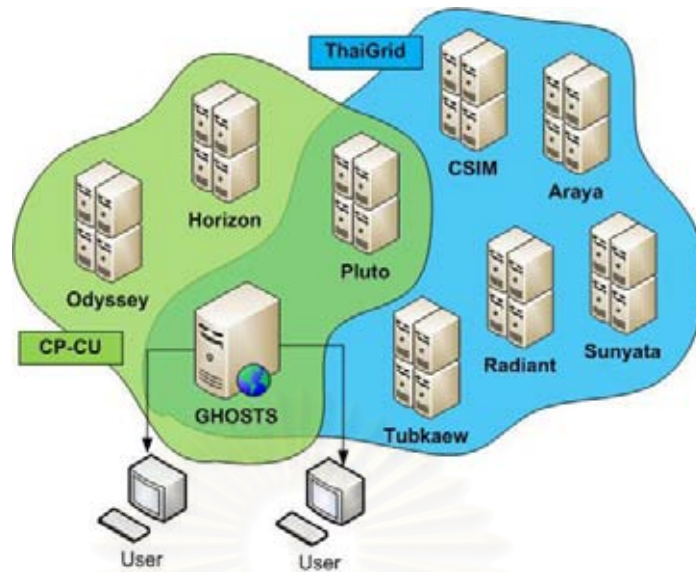


Figure 58: VO resources diagram

Secondly, some software and network configurations must be made to the initial release of the system in order to seamless access ThaiGrid resources. Mostly, they are firewall-related issues. The quality of the network during the testing period is shown in Table 6.

Table 6: Network capacity of each site

Host	Institute	Ping (ms)	Bandwidth (Mb/s)
Araya	Thaigrd	1.898	91.7
Sunyata	Thaigrd	1.942	85.4
Radiant	KU	1.885	63.0
Tubkaew	Silpakorn	-	11.4
CSIM	AIT	2.888	55.9
MU-HPC	Mahidol	5.547	-

*Measured on 27 June 2007

Thirdly, the experiment is conducted by performing data transfer using GridFTP. In this experiment, the amount of time used in downloading files via the portal and a GridFTP client (globus-url-copy) are compared. In both cases, only the time used in transferring a file from a remote site to the GHOSTS server is measured. The time spent on downloading the file from the portal to a user's desktop is neglected, because it depends on the user's connection capacity. However, with the 100 Mb/s connection used in our department, it took only 3-5 seconds to download a 50 MB file.

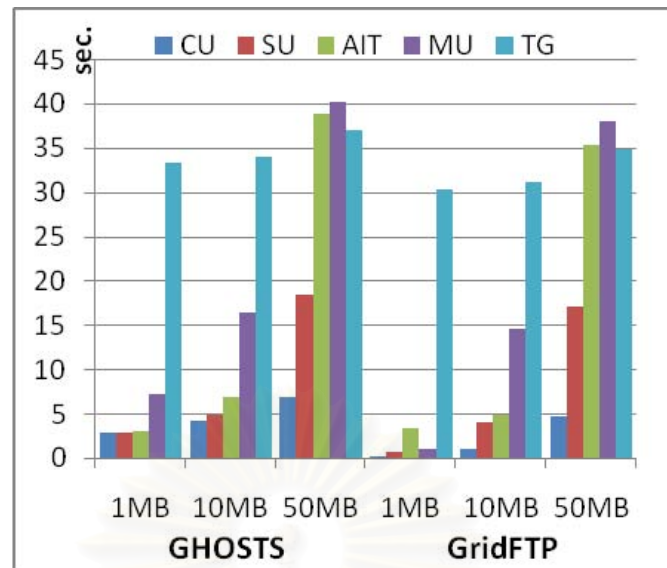


Figure 59: Data transfer result

As expected, the result shown in Figure 59 concludes that there is a slight overhead in transferring files with the Grid portal, because it takes more time to complete the operation than using Globus's command line. However, we have found an interesting issue regarding the ThaiGrid's main site. The experiment shows that any GridFTP activities take at least 30 seconds to complete on the ThaiGrid's main site which includes Araya, Radiant, Sunyata. Note that the 30 seconds response time is quite long for an interactive environment like a Grid portal. Additionally, we also confirm this result by conducting the same experiment on other university site. This concludes that there is a problem with the network infrastructure at the main site of ThaiGrid.

This study has shown that GHOSTS is capable of interacting with ThaiGrid resources. However, the result of the data transfer experiment shows that the interactive nature of Grid portal is suffered from network issue. This will also affect the job submission functionality of the portal as well because every job must perform file staging.

5.6.2 Class Room Teaching

The Department of Computer Engineering at Chulalongkorn University teaches Information Systems Organization to all of its Bachelor's students during their fourth semester. One section of this course introduces students to various applications of computer network including the High-Performance Computing

network. As a result, the students had been assigned with homework to study about Cluster and Grid Computing and GHOSTS was used as a tool for them to experiment with Grid Computing.

In this activity, the student had been given introductory handouts about the Cluster and Grid Computing and also the GHOSTS' user manual to study by themselves. The objectives of the assignment were that, during a two-week period, every student must access GHOSTS and uses the Mandelbrot Job Profile (see section 4.3.2.1) to generate their own images and post them on SPACE Lab's web site. Moreover, the best image voted by these students received an additional reward.

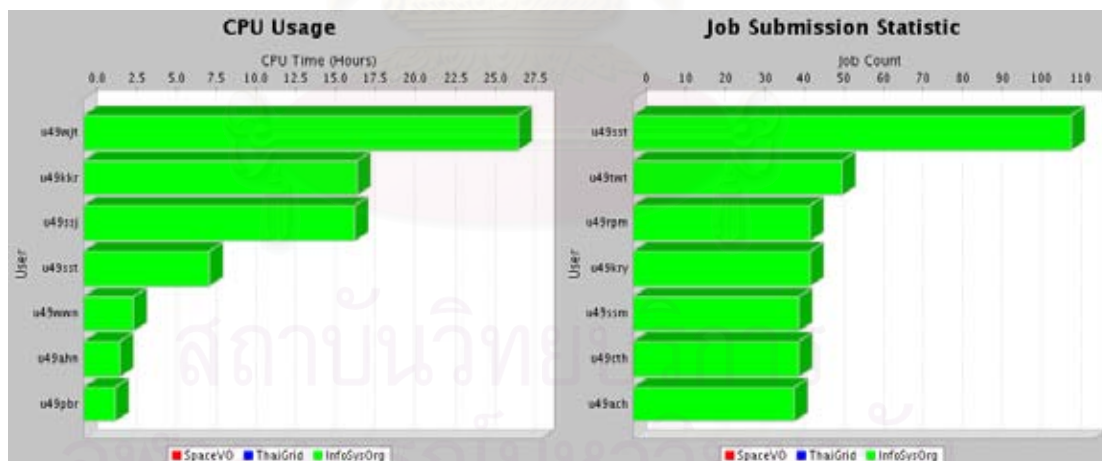
A new VO called *InfoSysOrg* was specially created for this purpose. It contained two production clusters of the department which are Pluto and Horizon cluster. They have a combined number of 48 job slots but only six are reserved for this activity because most of them were occupied by other researchers. A shared account mapping was used as the mapping scheme on both resources because large number of students (around 100 students) became a member of this VO, and no separation of user space was required because the students can download the output directly from the result page of the Mandelbrot Job Profile without searching through their shared home directory. Nevertheless, each of them was given a unique account on GHOSTS which was generated through the batch add feature.

Beside from basic tools such as file transfer and job submission, resource monitoring provided by Ganglia and various online videos tutorial were also provided on InfoSysOrg's VO portal. So they are equipped all tools necessary for using the system correctly and efficiently. Moreover, a lot of effort has been put into making the Mandelbrot Job Profile fun and easy to use so that they will enjoy using the system.

The summarized statistic result collected from both GHOSTS and Google Analytics presented by Table 7 shows that the system had been extensively used by the students. This result outlines the load capacity which can be achieved by the GHOSTS's portal. Moreover, the students were able to use the system without difficult despite the fact that they must learn how to use the system on their own through manual and tutorial. This confirms that GHOSTS' interface is easy to learn and use by users that have little knowledge of Grid Computing.

Table 7: Activity statistic summary

GHOSTS	
Number of active users	111
Jobs submitted	1,810
Average jobs per user (jobs/user)	16.5
Total job time (minutes)	6460.51
Average job time (minutes/user)	58.73
Google Analytics	
Total visits	875
Total page views (pages)	26,383
Average page views (pages/visit)	30.15
Average time on site (minutes)	20.17
Peak page views per day (pages/day)	3,733
Peak visits per day (visits/day)	100

**Figure 60: Usage statistic charts**

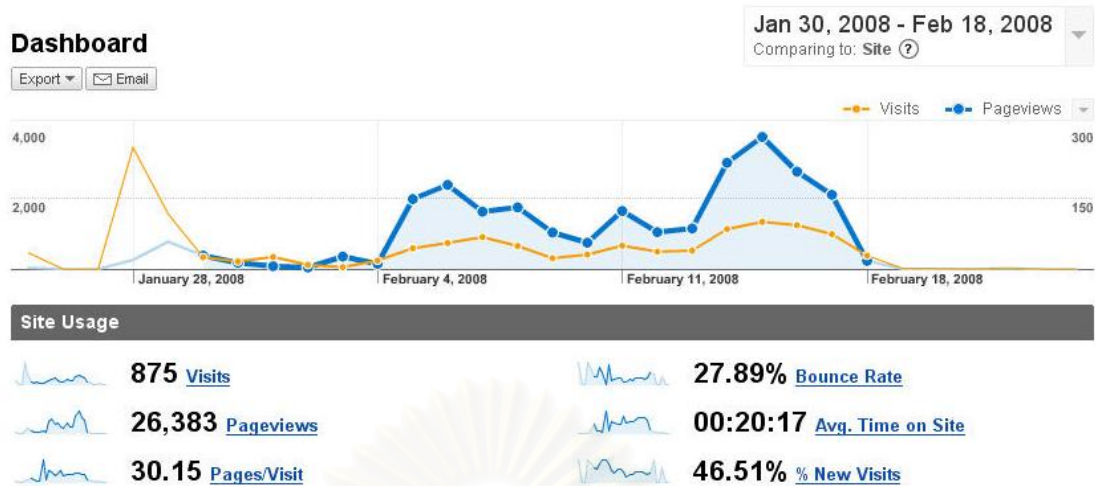


Figure 61: Google Analytics' graph report¹

From the administrator point of view, managing this activity on GHOSTS was also effortless because the new VO and its users were easily created through the portal. Since one-to-one account mapping was used, only one entry was required to add into “*vomap-file*” on each node in order to provide services to hundred users. Finally, these preparations including the Mandelbrot Job Profile development were completed in just one week, so it was also a proof that the system can rapidly deploy a VO.

The slight drawback observed from this case study was that users have to visit different site in order to access services provided by other tools. For example, Ganglia's website was used as a monitoring tool; SPACE Lab's web site constructed by using Drupal [45] was used to organize the picture contest where it provides features such as image uploading, voting and ranking. Please note that it is possible to seamlessly integrate or developing these features from scratch and includes them into GHOSTS' portal. However, these features are not part of the scope of this thesis and it is not possible to introduce production-grade services within a limited time without using these tools. Nevertheless, the degree of integration with other tools currently achieved by GHOSTS is quite satisfying.

¹ Detailed result can be found in Appendix C

CHAPTER VI

CONCLUSION

6.1 Summary

The initial idea behind this research is that Grid Computing has a potential to support various type of collaborative projects. However, the cost of building one own Grid is not feasible for some projects that are short-lived or has limited resources. The solution to this problem is to use the concept of Virtual Organizations. Each collaboration project can be visualized as a VO resting on shared Grid infrastructure. In this vision, multiple VOs may exist with overlapping resources. Moreover, their VOs are also dynamic because each project can change over time. However, such vision will be successfully if the overhead of creating and maintaining multiple VOs are low.

Therefore, a Grid hosting model, inspired by a web hosting solution, is developed as a solution to this problem. A Grid hosting should be able to host VO in the same way that a web hosting provider hosts a website. As a result, the Grid HOSTing System (GHOSTS) is developed according to this model. Many important design decisions have been made in order to realize this system.

Firstly, the centralized architecture is used in order to reduce the overhead of managing dynamic VOs. Secondly, a simple VO model is used so that it can be deployed efficiently on the existing Grid infrastructure. This is achieved by using Grid portal is control VO-level access and develop a VO authorization module that helps manage resource-level access control. The final result of this VO model allows different types of VO to be hosted by the system. Thirdly, VO usability is increased by adopting Grid portal as a main interface of the system. A VO-aware Grid portal is developed based on GridSphere/GridPortlet framework so that both users and administrator can easily operate their tasks. Finally, many add-on functionalities have been added to the system such as various job profiles, VO portal, etc. in order to demonstrate the capability of the system in handling different VO requirements.

Various evaluations have been performed on the system in order to prove its usability. The system has shown that it is capable of operation with ThaiGrid resources and suitable for a new user of Grid system. Moreover, the result from these

evaluations also shows that system's reliability and performance is quite satisfying. Additionally, security, reliability and scalability of the system are also discussed.

The main contribution of this thesis is the proposed model of Grid hosting solution and its implementation. Various design decision of the system can be used as an example of balancing practicality and innovativeness. There are many works that can be used to compare various part of this system but most of them rely on mature infrastructure which cannot be found in the environment of this study.

6.2 Future Works

6.2.1 VO Authorization Integration

Although GHOSTS chooses to adopt multiple certificate solutions to overcome the problem of VO authorization, it is possible to integrate other VO authorization services such as VOMS and CAS into GHOSTS.

6.2.2 Web-service Components Compatibility

The version of GridPortlets that GHOSTS uses can only interact with pre-web service components of Globus Toolkit. By porting GHOSTS to a new version of GridPortlets which is released as a Vine Toolkit [46], GHOSTS will be able to interact with web service components which will also allow it to expand its functionality.

REFERENCES

- [1] Foster, I., Kesselman, C., and Tuecke, S. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputer Applications 15: 200–222.
- [2] Nasser, B., Barrere, F., Benzekri, A., Laborde, R., and Kamel, M. (2006). Automated Creation of Inter-organizational Grid Virtual Organizations. Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP: 1-4.
- [3] Rodosek, G.D., Hegering, H.G., and Stiller, B. (2006). Dynamic Virtual Organizations as Enablers for Managed Invisible Grids. Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP: 1-4.
- [4] Demchenko, Y., de Laat, C., and Ciaschini, V. (2006). VO-based Dynamic Security Associations in Collaborative Grid Environment. Collaborative Technologies and Systems, 2006. CTS 2006. International Symposium on 38-47.
- [5] Lyle, J.W. (2005). A Simple Virtual Organisation Model and Practical Implementation. Proceedings of the 2005 Australasian workshop on Grid computing and e-research - Volume 44 Newcastle, New South Wales, Australia.
- [6] Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, A., Lorente, K., et al. (2005). From gridmap-file to VOMS: managing authorization in a Grid environment. Future Generation Computer Systems 21: 549-558.
- [7] Foster, I., Kesselman, C., Pearlman, L., Tuecke, S., and Welch, V. (2003). The Community Authorization Service: Status and future. Proceedings of Computing in High Energy Physics 03 (CHEP '03)
- [8] Foster, I. (2006). Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP International Conference on Network and Parallel Computing 2-13. Springer-Verlag
- [9] Cornwall, L., Jensen, J., Kelsey, D., and McNab, A. (2003). EU DataGrid and GridPP Authorization and Access Control. UK e-Science All Hands Conference.
- [10] Novotny, J., Russell, M., and Wehrens, O. (2004). GridSphere: an advanced portal framework. Euromicro Conference, 2004. Proceedings. 30th: 412-419.
- [11] Russell, M., Novotny, J., and Wehrens, O. (2005). The Grid Portlets Web Application: A Grid Portal Framework. Parallel Processing and Applied Mathematics (PPAM): 691-698.
- [12] Open Grid Computing Environments Collaboratory [Online]. Available from: www.collab-ogce.org
- [13] The Thai National Grid Project [Online]. Available from: <http://www.thaigrid.or.th/>
- [14] Foster, I., and Kesselman, C. (1998). The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
- [15] The Enabling Grids for E-science project [Online]. Available from: <http://www.eu-egee.org/>
- [16] The Access Grid [Online]. Available from: www.accessgrid.org
- [17] Welch, V., Siebenlist, F., Foster, I., Bresnahan, J.A.B.J., Czajkowski, K.A.C.K., Gawor, J.A.G.J., et al. (2003). Security for Grid services. High Performance Distributed Computing, 2003. Proceedings. 12th IEEE International Symposium on: 48-57.

- [18] Streit, A., Erwin, D., Lippert, T., Mallmann, D., Menday, R., Rambadt, M., et al. (2005). Unicore-From Project Results to Production Grids. Grid Computing: The New Frontiers of High Performance Processing 14: 357-376.
- [19] Foster, I., Kishimoto, H., Savva, A., Berry, D., Grimshaw, A., Horn, B., et al. (2006). The Open Grid Services Architecture. Version 1.5 [Online]. Available from: www.gridforum.org/gf/docs
- [20] The Open Grid Forum [Online]. Available from: www.ogf.org
- [21] Hepper, S., and Hesmer, S. (2003). Introducing the Portlet Specification [Online]. Available from: www.javaworld.com/javaworld/jw-08-2003/jw-0801-portlet.html
- [22] Introduction to JSR 168—The Java Portlet Specification [Online]. Sun Developer Network. Available from: http://developers.sun.com/prodtech/portalserver/reference/techart/jsr168/pb_w_hitepaper.pdf
- [23] Introducing Java Portlet Specifications: JSR 168 and JSR 286 [Online]. Available from: <http://developers.sun.com/portalserver/reference/techart/jsr168/>
- [24] Novotny, J., Tuecke, S., and Welch, V. (2001). An online credential repository for the Grid: MyProxy. High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on: 104-111.
- [25] Cannon, S., Chan, S., Olson, D., Tull, C., Welch, V., and Pearlman, L. (2003). Using CAS to Manage Role-Based VO Sub-Groups. Proceedings of Computing in High Energy Physics 03 (CHEP '03)
- [26] Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Gianoli, A., Spataro, F., et al. (2003). Managing Dynamic User Communities in a Grid of Autonomous Resources. Computing in High Energy and Nuclear Physics California.
- [27] Katarzyna, K., Karl, D., and Ian, F. (2004). From Sandbox to Playground: Dynamic Virtual Environments in the Grid. Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing.
- [28] Dynamic Accounts [Online]. Available from: http://dev.globus.org/wiki/Incubator/Dynamic_Accounts
- [29] Denmark, J., Jankowski, M., Krenek, A., Matyska, L., Meyer, N., Ruda, M., et al. (2005). Best Practices of User Account Management with Virtual Organization Based Access to Grid. Parallel Processing and Applied Mathematics, 6th International Conference, Poznan, Poland.
- [30] Bhatia, K., Chandra, S., and Mueller, K. (2005). GAMA: grid account management architecture. e-Science and Grid Computing, 2005. First International Conference on: 8 pp.
- [31] Andronico, G., Barbera, R., Falzone, A., Re, G.L., Pulvirenti, A., and Rodolico, A. (2003). The GENIUS Web portal: Grid computing made easy. Information Technology: Coding and Computing [Computers and Communications], 2003. Proceedings. ITCC 2003. International Conference on: 425-431.
- [32] Barbera, R., Falzone, A., Ardizzone, V., and Scardaci, D. (2007). The GENIUS Grid Portal: Its Architecture, Improvements of Features, and New Implementations about Authentication and Authorization. Proceedings of the 16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises.

- [33] Hibernate - Relational Persistence for Java and .NET [Online]. Available from: <http://www.hibernate.org/>
- [34] JFreeChart [Online]. Available from: <http://sourceforge.net/projects/jfreechart>
- [35] Apache Axis2 [Online]. Available from: <http://ws.apache.org/axis2/>
- [36] Globus Authorization Callouts (Pre-Web Services) [Online]. Available from: <http://www.globus.org/security/callouts/>
- [37] Blender [Online]. Available from: <http://www.blender.org/>
- [38] The Persistence of Vision Raytracer (Povray) [Online]. Available from: <http://www.povray.org/>
- [39] Scilab - The open source platform for numerical computation [Online]. Available from: <http://www.scilab.org/>
- [40] Matlab [Online]. Available from: <http://www.mathworks.com/>
- [41] Gaussian 03 [Online]. Available from: <http://www.gaussian.com/>
- [42] Google Analytics [Online]. Available from: <http://www.google.com/analytics/>
- [43] Ganglia [Online]. Available from: <http://ganglia.sourceforge.net/>
- [44] Valgrind [Online]. Available from: <http://www.valgrind.org/>
- [45] Drupal content management platform [Online]. Available from: <http://drupal.org/>
- [46] The Vine Toolkit [Online]. Available from: <http://www.gridisphere.org/gridisphere/gridisphere/guest/vine/r/>



APPENDICES

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDIX A

GHOSTS USER MANUAL

A.1. Introduction

Virtual Organization (VO) is a result of multi-institution collaboration where each organization contributes its resources such as users, machines to form a workforce. Grid technology can provide an infrastructure for accessing these geographically dispersed resources in a uniform interface. Setting up a VO that spans across organizational units impose technical challenges that required good understanding in Grid infrastructure and security. In order to overcome these challenges, we have developed the Grid Hosting System (GHOSTS), a system based on grid portal which provides access to Grid resources under the VO concept. It can host multiple VOs that comprise resources from various organizations.

A.2. Architecture

GHOSTS is based on a centralized architecture in which a GHOSTS server is responsible for the management of all Grid nodes and users.

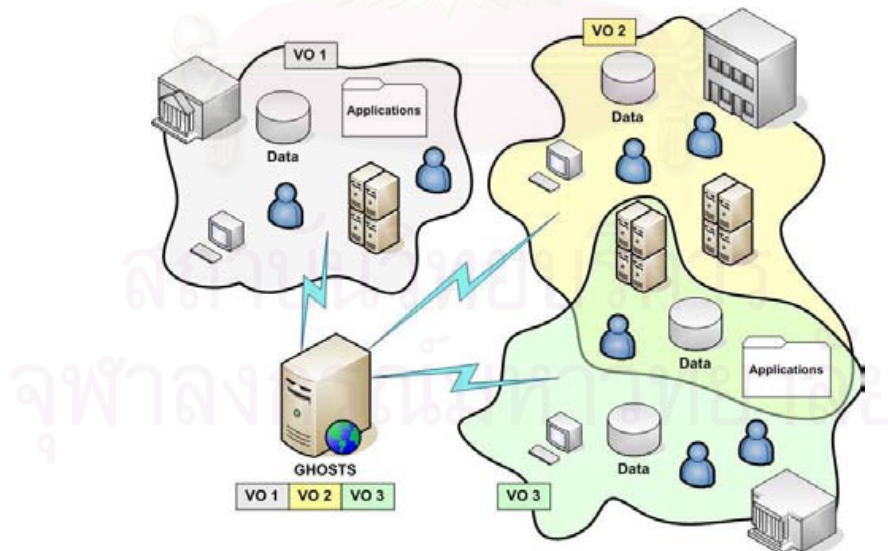


Figure 62: GHOSTS system diagram

A.3. The Grid Portal

A Grid portal provides a platform where a user can easily access Grid functionalities such as job submission and file transfer. GHOSTS' Grid Portal is based on the GridSphere Portlet Framework. The term "Portlet" can be described as a component-based programming model for web applications. As a result, a portlet-based Grid portal comprises of many portlets that provide specific functions for users.

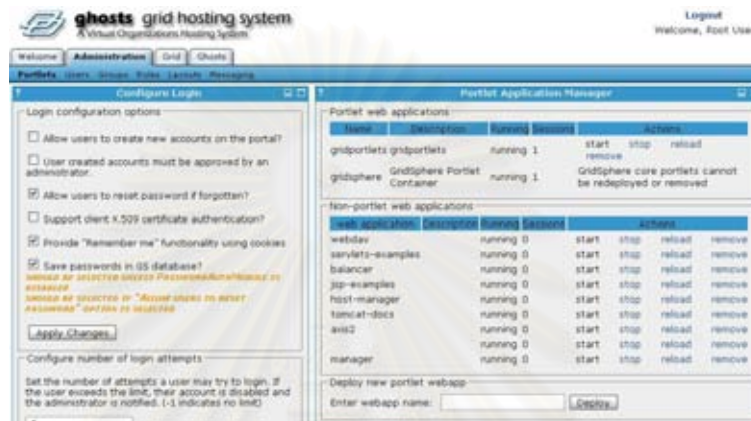


Figure 63: Multiple portlet displayed on the same page

A.4. Getting Started

Mozilla's Firefox is a preferred web browser for using the system (Microsoft's Internet Explorer work correctly except the file upload functionality). The front page of the system allows you to login to the system.

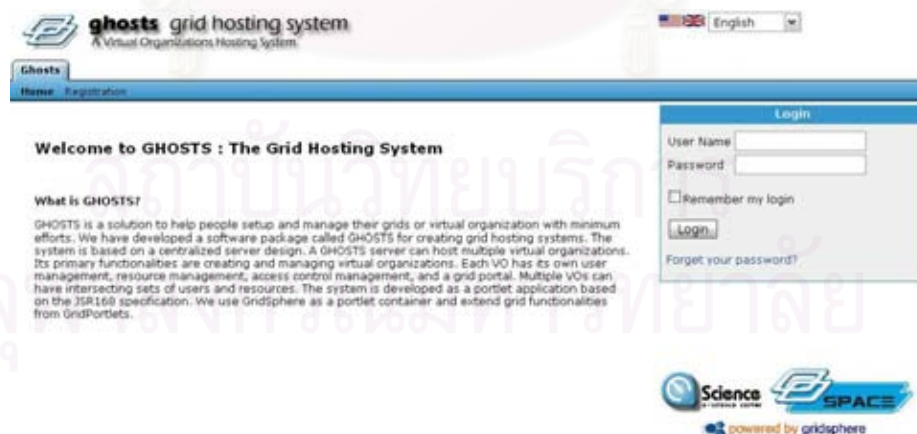


Figure 64: GHOSTS front page

Additionally, the Registration Portlet in the front page can be used to request for a GHOSTS account.

Figure 65: The Registration Portlet

A.5. Activating a VO

A user must select which VO he is going to work with. This step must be done before he can use other functionalities of the system. The Activation Portlet which can be accessed by going to the GHOSTS Tab and Activation, allows him to activate his VOs. By clicking the activate button, the system will handle all the complexity of Grid security for him. He can also switch between VOs which will allow him to access different resources or job profiles.

VO Name	Role	Activation Time	Action
SpaceVO	VO ADMIN	23 hours 59 minutes 59 seconds	<input type="button" value="Activate"/> <input type="button" value="Deactivate"/>

Figure 66: The VO Activation Portlet

In Figure 66, this user has only one VO membership. When pressing the Activate button, the default activation period is seven days. This value can be changed by selecting an option from a list box.

In the Grid, a user need to present a key when he which to do any action on Grid resources. We usually use a short live key in order to reduce the chance of the key being compromised. However, with a single click of VO activation, the system will handle all the underlying mechanisms related to the Grid.

IMPORTANT

A user should select an activation time that is greater than his job's runtime. Each job that he submits during that period will take the key with it. If the key is expired before a job is finished then he will lost contact with the job; however, his job will continue to run until it is finished.

Now the user is ready to uses other functionalities of the system, but let's begin by explaining various part of the portal before diving into usage details.

A.6. Overview of the Portal

The portal main interface is composed of various tabs. Each tab has sub-tabs that usually link to a page that display one or more portlets. There are three main tabs in the portal.

- **Welcome** — This tab allows a user to customize the portal and modify his profile.
- **Ghosts** — This tab provides GHOSTS VO management functionalities. For most users, they usually access this tab in order to access the Action portlet. This tab has many sub-tabs as follows
 - **Activation** — Provides VO activation
 - **Membership** — Users can request for other VO memberships from this portlet
 - **Administrator** — Allows VO administrator to modify his VOs (Only VO administrator can access its functions)
 - **Resource** — If a user wants to contribute a resource (machine) to the system, this portlet allows him to register his resource.
- **(VO Name)** — This is a VO portal page; each VO has its own tab for VO users. A user with multiple VO membership will have access to multiple VO tabs. Currently, there are at least two portlets is provided in each VO portal page.
 - **File** — Provides file management functionality

- **Jobs** — Provides job submission functionality

In the following sections we will focus on two main functionalities for users which are file management and job submission.

A.7. File Management

This tab provides two portlets: the File Browser Portlet which provides the main interface of file management and the File Activity Portlet which list all previous file operations.

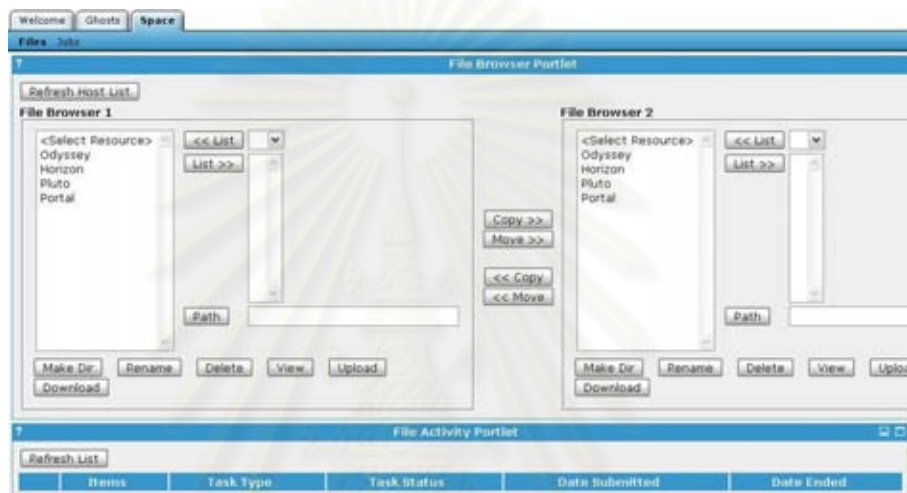


Figure 67: Use two file browser components to transfer file between hosts

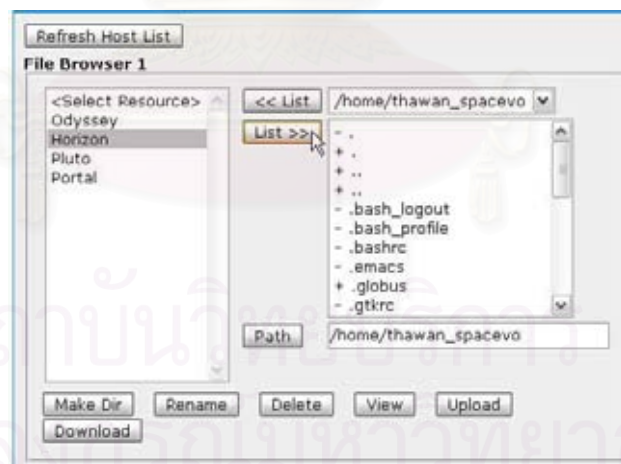


Figure 68: Select a host and use the *List >>* button to browse its files

Firstly, a list of available host will be presented on the left portion of each browser. In figure 9, there are 3 hosts and “Portal”. The last entry is a portal storage, this storage store file at the portal server. (Portal storage is shared among VOs, so it can be used as a intermediate storage for transferring files between VOs)

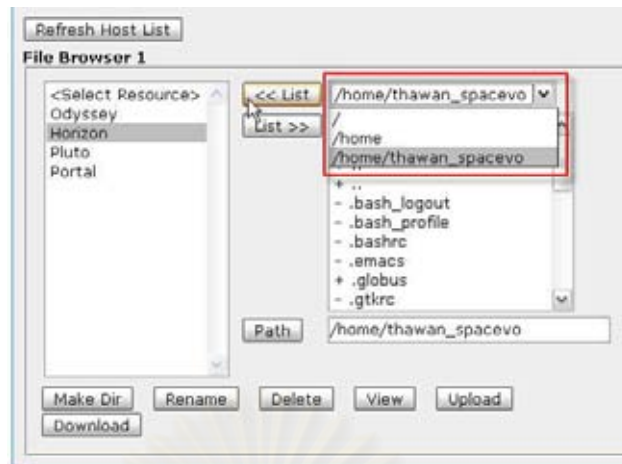


Figure 69: Use the <<List button and to list files from a parent directory

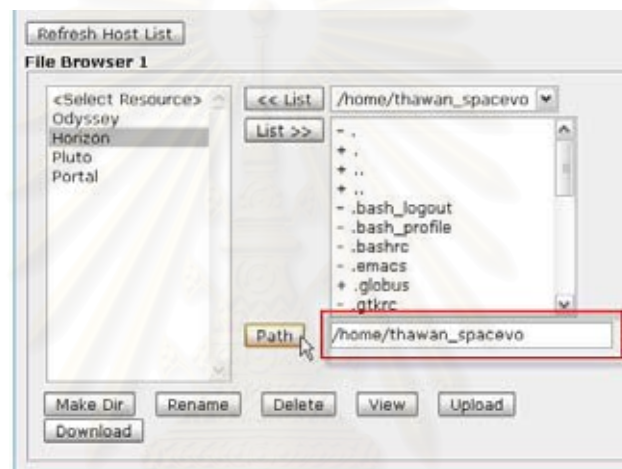


Figure 70: Use Path button to list files in a directory specified in path text box

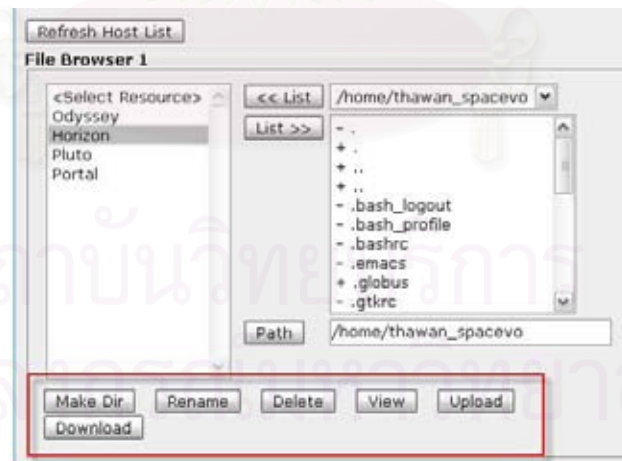


Figure 71: Various file operations

Figure 71 shows various file operations which can be implied from their names. The *View* button can be used to view and edit a text file.

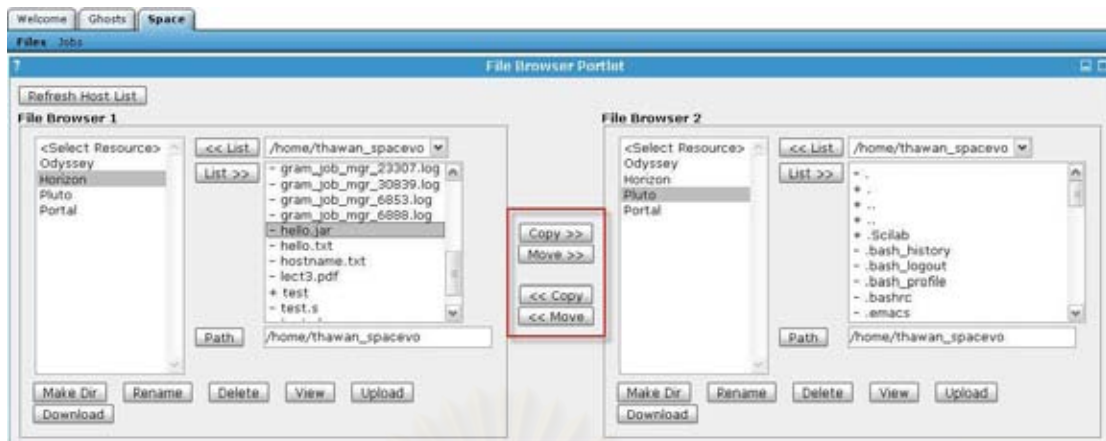


Figure 72: Copy and Move can be used to manage file across hosts.

A.8. Job Submission

The first page of the job submission portlet will list all jobs that a user has previously submitted through the portal. Here, he can click on each job to view its output and other information.

Job Id	Description	Job Type	Resource	Status	Date Submitted
<input type="checkbox"/> https://odyssey.cp.eng.chula.ac.th:51959/25267/1187170620/	Test	Generic GHOSTS Application	odyssey.cp.eng.chula.ac.th	Job completed with message success	Aug 15, 2007 4:37:01 PM

Figure 73: List of jobs previously submitted by a user.

Generic Application

Job Id	https://horizon.cp.eng.chula.ac.th:10002/7551/1187175084/	Job Resource	horizon.cp.eng.chula.ac.th	Date Submitted	Wednesday, August 15, 2007 5:51:24 PM ICT
Job Description	Test Script	Job Scheduler	sgc	Last Changed	Wednesday, August 15, 2007 5:51:35 PM ICT
Job Status	Job completed with message success	Job Queue		Date Ended	Wednesday, August 15, 2007 5:51:35 PM ICT

Job Spec: Job String Job Output Job Location

```

thawan spacevo
/home/thawan_spacevo
compute-0-1.local
Wed Aug 15 17:51:32 ICT 2007
Hello Thawan
  
```

Figure 74: The job view page.

To view the job output, select a job from the job list and click Job Output link to view the result. The *Refresh View* button is used to update the result in case of job is still running.

To submit a new job, press the *New Job* button. It will lead the job profile list page which can be used to select which type of jobs that is going to create. Each type of job has its associated job profile; it is a sequence of pages that helps a user set up various parameters of a particular job type.

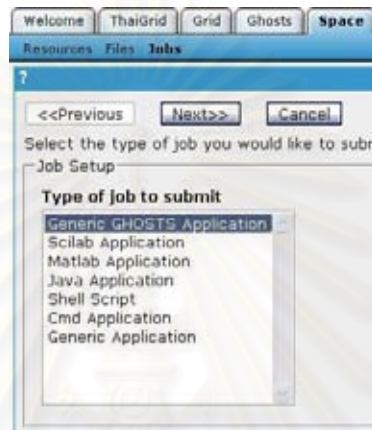


Figure 75: Select which type of jobs to start the job submission process

Here are the summary of each job type that is currently support in GHOSTS.

- **Generic GHOSTS Application** — This is a generic job profile for submitting any type of jobs. Many parameters can be guessed by those who familiar with UNIX system; however, some parameters are Grid-related.
- **Java Application** — Submit a job for running a Java program archived in JAR format.
- **Shell Script** — Submit a shell script. Users can upload, edit or create a new script through this job profile.
- **Scilab Application** — Submitting a job to Scilab by using a Scilab script.
- **Matlab Application** — Submitting a job to Matlab by using a Matlab script.
- **Cmd Application** — Executing simple UNIX commands.
- **Mandelbrot Rendering** — Rendering Mandelbrot Set's image.

Here, we will give detailed information about some job profiles.

A.8.1. Shell Script Job Profile

Figure 76: Shell script job submission interface

- **Job Description*** — The name of the job
- **Script Location*** — Absolute or relative path (from home directory) to the script. (absolute path: /home/thawan_spacevo/shell/test.sh, relative path: shell/test.sh)
 - **Browse** — This will bring up a file chooser dialog. It is similar to the file browser interface. It can be used to select an existing file or upload a new file by using upload button (After selecting a file, a host name list box will automatically changed to match the script location).
 - **Edit** — If a host name and a script location is correctly entered, this button can be used to modify the specified script.
 - **New** — By selecting a host name and a desired script destination, a new button can be used to create a new file (It will overwrite all previous content if the file is already existed).
- **Job Directory** — If not specified, the script will be executed at the user's home directory. Other directory can also be specified in this text box.
- **Host Name** —the execution host of this script is specified in this box. Moreover, the script must be store locally at this host too.
- **Scheduler** — Local scheduler at execution host.
 - **sge** — an equivalent of using qsub command (recommended)

- **fork** — a job will run on head node.
- **Shell Type*** — Specify the type of shell script that are going to execute. The following types are supported: bash, csh, ksh and ash.
- **Script Argument** — Specify script arguments in this text area.

A.8.2. Java Application Job Profile

Figure 77: Java job submission interface

- **Job Description*** — The name of the job
- **JAR File Location*** — Absolute or relative path (from home directory) to the JAR file.
 - **Browse** — This will bring up the file chooser dialog.
- **Host Name** — It will be automatically select the same host as the JAR file location
- **Scheduler** — Local scheduler at execution host.
 - **sge** — an equivalent of using qsub command (recommended)
 - **fork** — a job will run on head node.
- **Java Executable*** — The job profile will insert this value automatically if it knows the location. If not, the absolute path to the Java executable must be entered (Example: /usr/local/java/jdk-1.5.0_07/bin/java)
- **Standard Output** — Absolute or relative path (from home directory) to the file that will store the output. If not specified, the output will be stored at the

portal storage. (The latter option will failed if the activation time is expired before the job finished)

- **Program Arguments** — Specify the program arguments.
- **Java Arguments** — Specify the options for the java executable such as max memory.



Figure 78: Confirm page shows all job parameters

APPENDIX B

USECASE SCENARIOS

This section presents some use cases which depict how users of various roles take part in the VO management of the portal by using sequence diagram.

B.1. User Entry

This use case presents how a new user can create an account on GHOSTS and also become a new member of a particular VO.

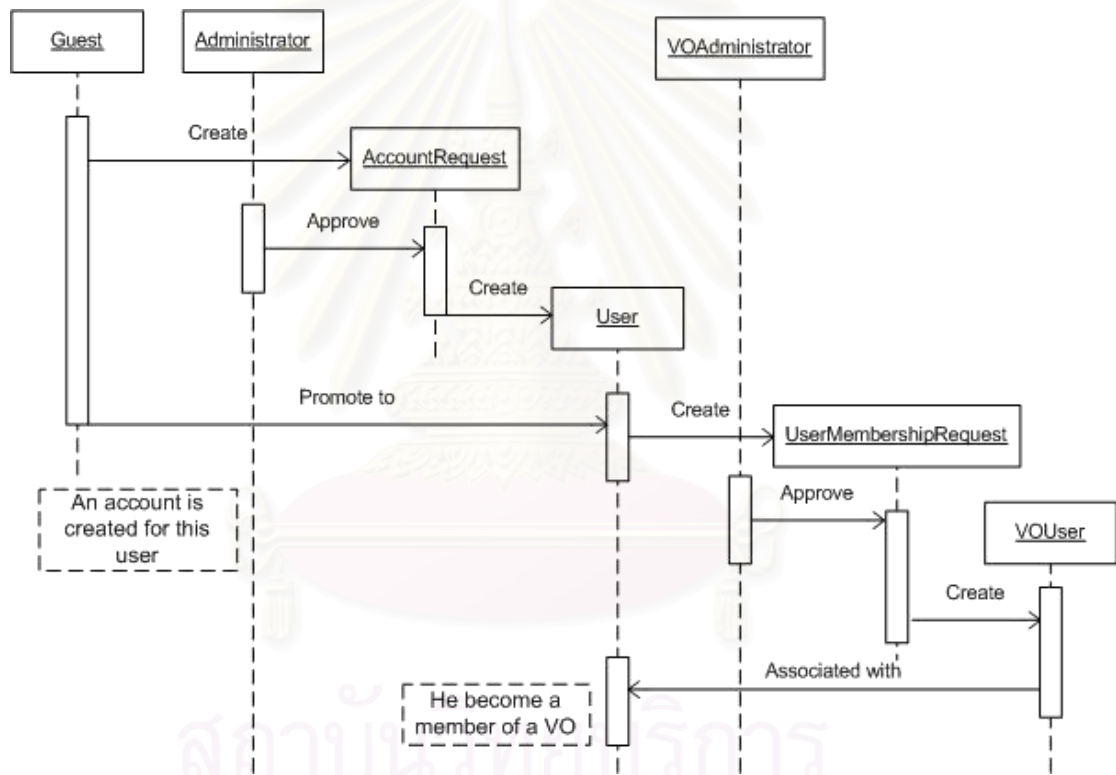


Figure 79: User entry diagram

B.2. VO Creation

This use case presents how VO is created from user's request.

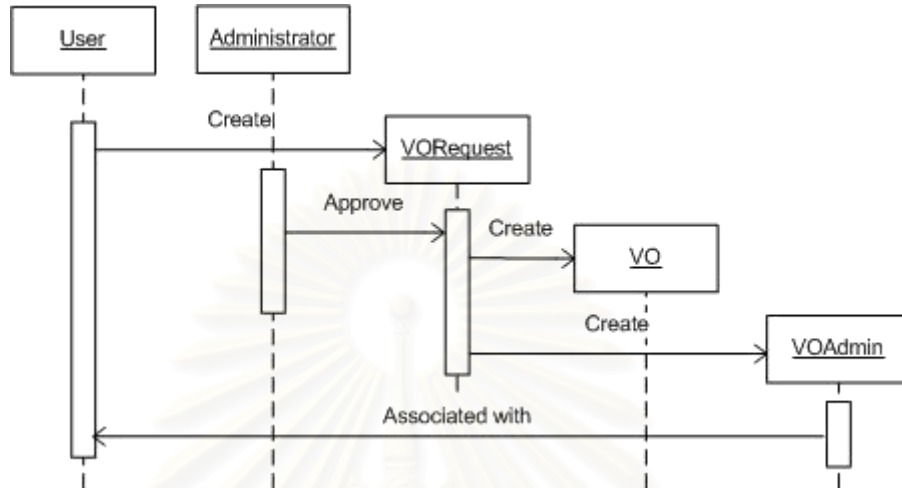


Figure 80: VO creation diagram

B.3. Resource Admission

This use case shows how a user can submit a new resource to the system and become a resource administrator.

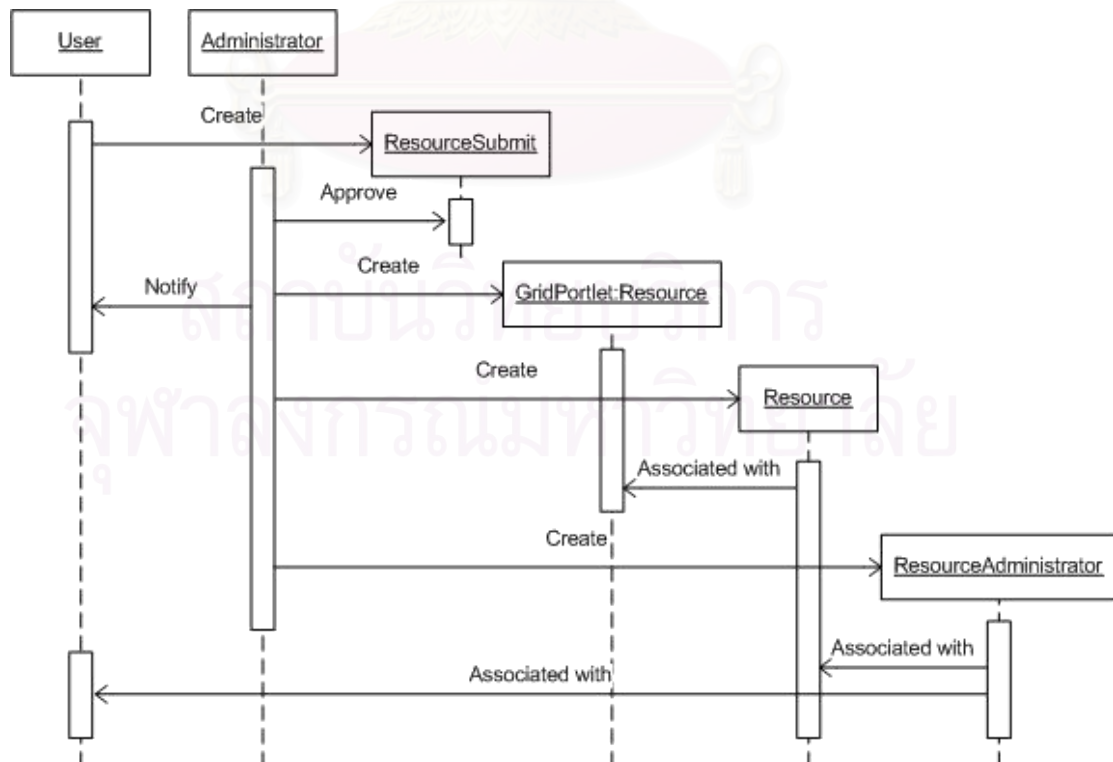


Figure 81: Resource admission diagram

B.4. Resource Membership

This use case depicts how a resource becomes a member of a particular VO.

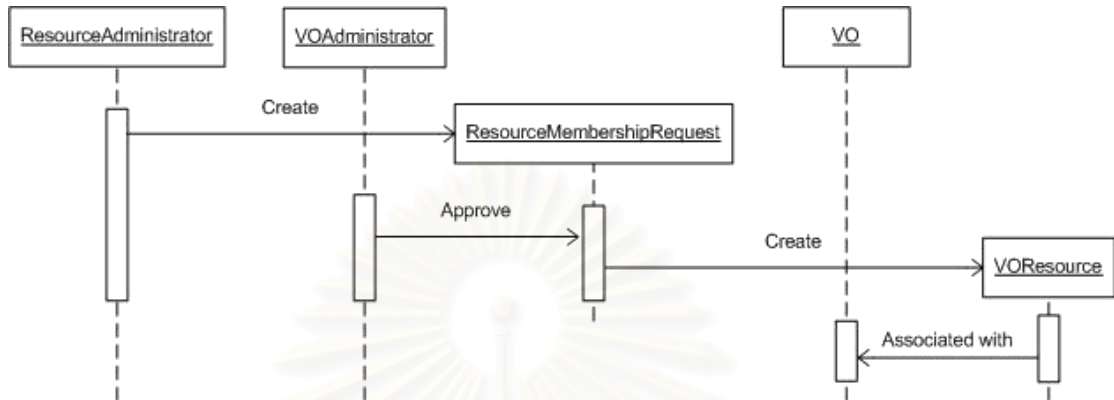


Figure 82: Resource membership diagram

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDIX C

EVALUATION RESULT

C.1. Google Analytics Result

C.1.1. Overall Statistic

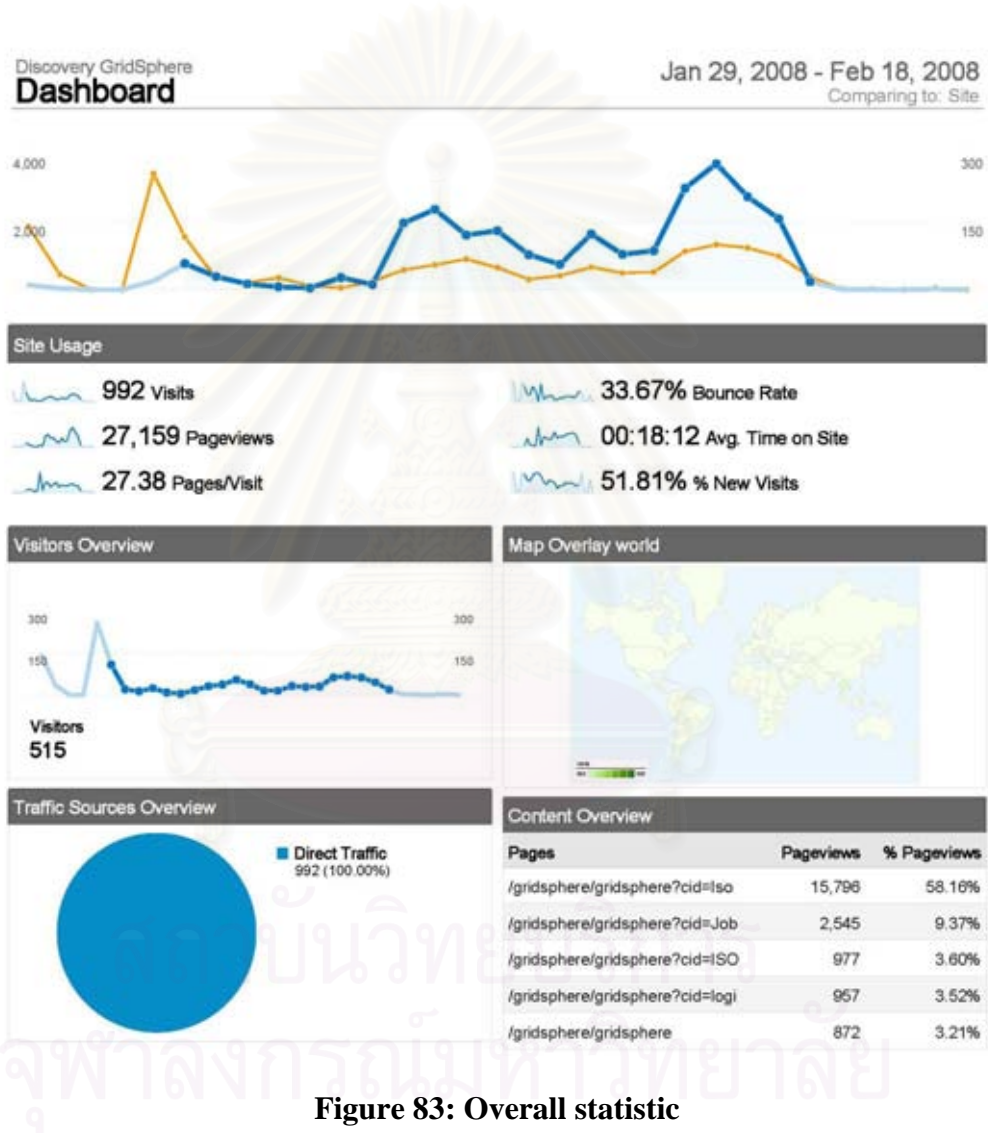


Figure 83: Overall statistic

C.1.2. Visitors Overview



Figure 84: Visitor overview

สถาบันวิทยบริการ
 จุฬาลงกรณ์มหาวิทยาลัย

C.1.3. Top Content

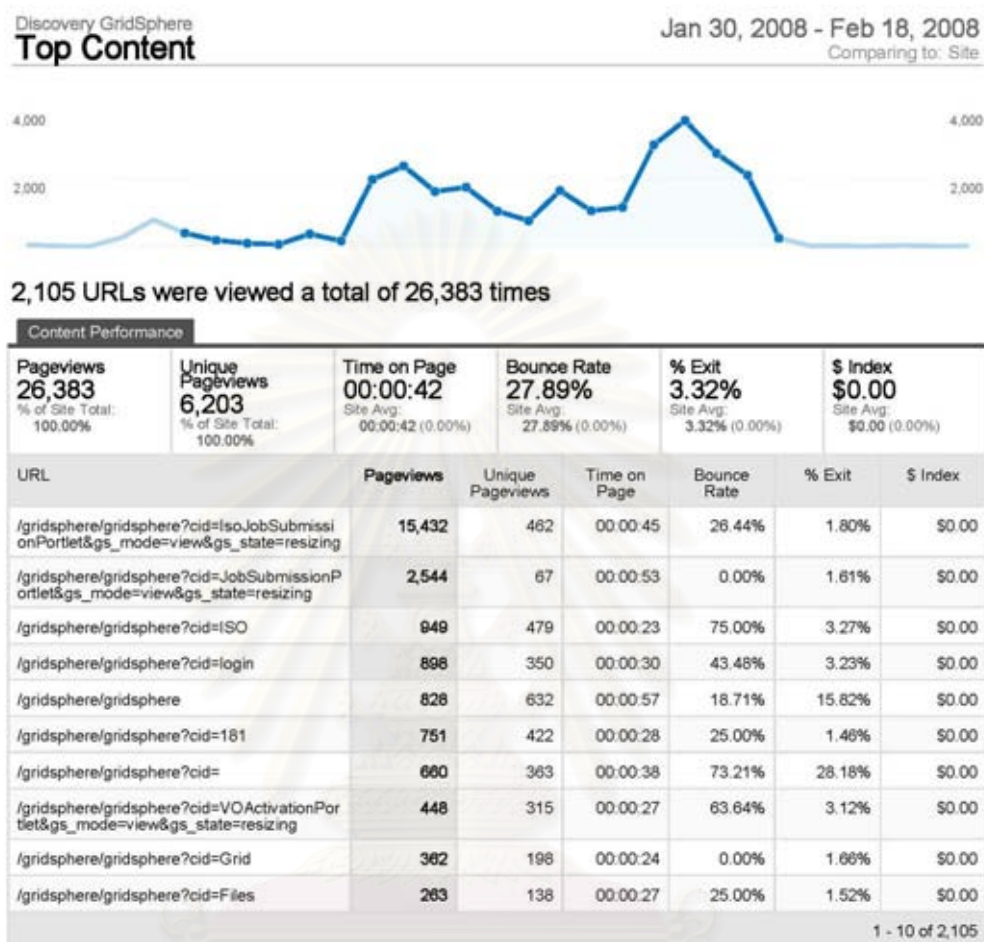


Figure 85: Top content

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

C.2. Mandelbrot Picture Contest Result

Mandelbrot Picture Contest

Add your own image (Require login)

[View gallery](#)

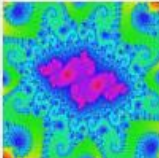
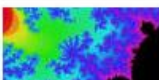
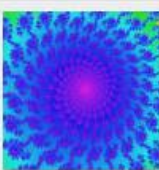
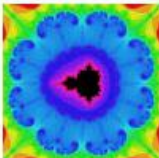
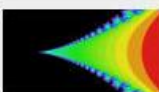
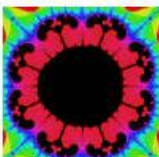

Image	Title	Vote
	Sukhum - Adjusted Exponential Gradient	Average: ★★★★★ Average: 4.8 (31 votes)
	Hero	Average: ★★★★★ Average: 4.7 (25 votes)
	Mystic Spiral	Average: ★★★★☆ Average: 4.4 (21 votes)
	Blue Flower	Average: ★★★★★ Average: 4.9 (14 votes)
	555	Average: ★★★☆☆ Average: 2.9 (12 votes)
	The born of Flower	Average: ★★★★★ Average: 4.8 (9 votes)
	peach::pic2 ~ Red Fire!!	Average: ★★★★★ Average: 4.8 (8 votes)

Figure 86: Contest result

BIOGRAPHY

Thawan Kooburat was born on 1 May 1984 and lived in Pattani until he came to study in high school at Bangkok. He received Degree in Bachelor of Engineering (Computer Engineering) from Chulalongkorn University in April, 2006. Moreover, he has been working in the area of Grid Computing since his undergraduate study. His current research is the Grid Hosting System which involves various aspects of Grid technologies such as Virtual Organization, Grid Infrastructure and Grid portal.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย