Design and Development of A Grid Portal

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science & Engineering

By:
Virendra Kumar
(800832018)

Under the supervision of:
Dr. Inderveer Chana
Assistant Professor, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004
JULY - 2010

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Design and Development Of A Grid Portal"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Computer Science and Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Inderveer Chana** and refers other researcher's works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

V. Kumar

(Virendra Kumar)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Inderveer 12/7/2010

(Dr. Inderveer Chana)
Assistant Professor
CSED

Countersigned by

(Dr. Rajesh Bhatia) 12/07/10
Head (CSED)
Thapar University,
Patiala

(Dr. R. K. Sharma) 14.7.10
Dean (Academic Affairs)
Thapar University,
Patiala

Acknowledgement

No volume of words is enough to express my gratitude towards my guide **Dr. Inderveer Chana**, Department of Computer Science & Engineering, Thapar University, Patiala, who has been very concerned and has aided for all the materials essential for the preparation of this thesis report. She has helped me to explore this vast topic in an organized manner and provided me with all the ideas on how to work towards a research-oriented venture.

I am also thankful to **Dr. Rajesh Bhatia,** Head of Department, Computer Science and Engineering Department and **Dr. Inderveer Channa**, P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there at the need of the hour and provided with all the help and facilities, which I required, for the completion of my thesis work.

Most importantly, I would like to thank my parents and the almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

V. Kumar

Virendra Kumar

(800832018)

Abstract

The Grid has shifted its focus from the high performance aspects towards Virtual Organizations which provide flexible, secure, coordinated resources sharing among collections of individuals, institutions and resources. The world of Grid computing is so vast that it always poses constant challenges and high learning curves to researchers and scientists. An increased need for collaborative research, together with continuing advances in communication technology and computer hardware, has facilitated the development of distributed systems that can provide users access to geographically disperse computing resources that are administered in multiple computer domains. The term grid computing, or grids, is popularly used to refer to such distributed systems.

To access Grid resources an interface is needed. Grid portals are an increasingly popular mechanism for creating such customizable, Web-based interfaces to Grid services and resources. A Grid portal is a Web-based gateway that provides seamless access to heterogeneous backend resources. The main objective of Grid Portals is to provision Grid-based backend resources in transparent way. Portals are gaining attention among programmers due to their ease in development, richness in functionality, customization of interface and pluggable architecture. A collaborative portal brings together various resources, applications and services under one roof and provides a single point of entry and access to the users.

This Thesis addresses the requirements for building a high performance Grid Computing portal. It also presents a brief introduction of Grid Computing and analyzes and compares existing Grid Portal toolkits. A detailed description of design and development of a Grid Portal has been presented in this Thesis.

.

.

Contents **Page No.**

List of Figures

List of Tables

Chapter 1
Introduction

Grid Computing is problem solving environment that permits scientists and researchers the capability to solve program, access and implement distributed Grid applications with the help of a Web Browser and other desktop software. The main objective is to allow the scientists to focus completely on the problem by making the Grid as a transparent wing of the scientist's desktop computing environment. Today Grid has shifted its focus from the high performance aspects towards Virtual Organizations (VO) which grant flexible, secure and coordinated resources sharing among the organizations. Grid computing is an approach to distributed computing that spans not only locations but also organizations, machine architectures and software boundaries to provide unlimited power, collaboration and information access to everyone connected to Grid. A computational Grid environment behaves like a virtual organization consisting of distributed resources. A VO is a set of individuals and institutions organizations. Grid Computing also allows single large computation to be spread across several machines, each of which is executing some portion of the computation. Grid Computing focus on assemblage of distributed diverse computing resources used as a platform for high performance computing. Grid services utilize the available underutilize computational resources so that tasks can be run on any machine currently available. Grid is a software toolkit providing layers of services to access and manage distributed hardware and software resources [1]. Grid Computing has been delivering computing power to the users by sharing distributed computing power. To access Grid resources and services in Grid environment end user wants an easy and user friendly interface. Grid Portal is that interface. Grid Portals are an increasingly popular mechanism for creating customizable, web-based interfaces to Grid services and resources [3]. Grid Portal acts as an information aggregator which can access dissimilar information resources. The Grid Portal conceals the complexity of the Grid and offers user with a transparent accessing technique to the computational resources of the Grid. Grid Portals can provide an integrated platform for end users to access Grid services and resources via Web browsers without the need to download and install specialized software packages and

libraries [5]. It is an environment where the user can access Grid resources and services, execute and monitor Grid applications and collaborate with other users. The Grid Portal provides consistent "Grid context" for the user's Grid interactions [2].

1.1 Background

The preliminary idea of Grid computing was suggested by Len Kleinrock in 1969. He said "We will probably see the spread of computer utilities, which, like present electric and telephone utilities, will service individual homes and offices across the country" [4]. After that Ian Foster and Carl Kesselman, known as fathers of Grid computing, define Grid Computing as "A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities" [5].

Grid Computing is divided in three generation. First generation systems involved proprietary solutions for sharing high performance computing resources. Second generation systems introduced middleware to cope with scale and heterogeneity, with a focus on large-scale computational power and large volumes of data. Third generation systems are adopting a service-oriented approach, are metadata-enabled and may exhibit autonomic features [2].

1.2 Grid Computing

Recent changes in the nature of science are characterized by a significant class of contemporary scientific projects that require access to tremendous quantities of data, and Teraflops to Petaflops in computing power that are unavailable at any single research institute, and critically depend on collaborations of hundreds or thousands of researchers. Grid computing has the potential to provide users on-demand access to large amounts of computing power. The term "Grid computing" was itself preceded by the term metacomputing which also advocated transparent user access to distributed and heterogeneous computing resources by linking such resources by software and an underlying network [7]. The Grids gave evidence of how this new technology and distributed science infrastructure will contribute to scientific progress, how it will create a

whole new level of potential for breakthrough science advances. Figure 1.1 shown the simple structure of Grid environment in which a user connect with Grid environment through VO with the help of Grid Portal, an administrator will control Grid resource and Grid users. A computational Grid environment behaves like a VO which consists of distributed resources.

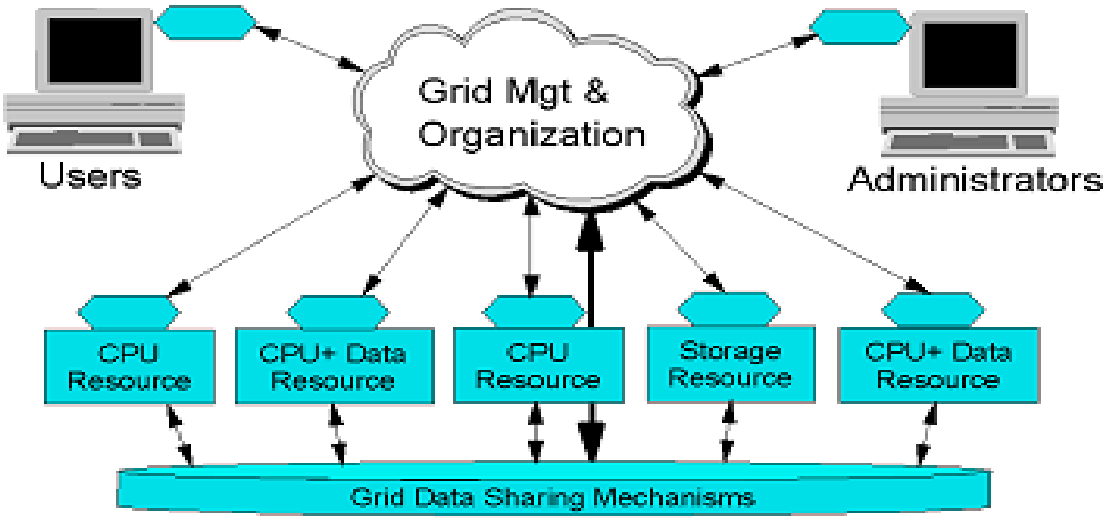

Figure 1.1 A Simple Grid [9]

1.2.1 Grid Services

Following services are popularly offered by Grid [6].

- Computational services: A computational Grid environment behaves like a virtual organization consisting of distributed resources. These are concerned with providing secure services for executing application jobs on distributed computational resources individually or collectively. Resources brokers provide the services for collective use of distributed resources.
- Data services: These services are concerned with proving secure access to distributed datasets and their management. To provide a scalable storage and

access to the data sets, they may be replicated, catalogued, and even different datasets stored in different locations to create and delusion of mass storage. The processing of datasets is carried out using computational Grid services and such combination is commonly called data Grids.

- Application services: Application services are concerned with application management and providing access to remote software and libraries transparently. The emerging technologies such as Web services are expected to play a leading role in defining application services of Grid Computing.
- Information services: These services are concerned with the extraction and presentation of data with meaning by using the services of computational, data, and application services. The low-level details handled by information services.
- Knowledge services: These are concerned with the way that knowledge is acquired, used, retrieved, published, and maintained to support users in achieving their particular goals and objectives. Knowledge services should be as information that is used to achieve a goal, solve a problem, or execute a decision.

1.2.2 General Principles for Grid Construction

Followings are the generic principles for Grid construction [10].

- Multiple administrative domains and autonomy: Grid resources are geographically distributed across multiple administrative domains and owned by different organizations. The autonomy of resource owners needs to be honoured along with their local resource management and usage policies.
- Heterogeneity: Grid involves a multiplicity of resources that are heterogeneous in nature and will cover a vast range of technologies. Grid provides seamless way to access different resources.
- Dynamicity or adaptability: In a Grid, resource failure is the rule rather than the exception. In fact, with so many resources in a Grid, the probability of some resource failing is high. Resource managers or applications must tailor their behaviour dynamically and use the available resources and services efficiently and effectively.

- Scalability: A Grid might grow from a few integrated resources to millions. This raises the problem of potential performance degradation as the size of Grids increases.

1.2.3 Grid Architecture

Grid architecture should be extensible and an open structure designed to solve key VO requirements. Foster and colleagues suggest the following Grid architecture, defined by a set of layers [11]. The Grid architecture is defined in such a manner that focuses on effective VO operations, which want the organization of sharing relationships among the participants of the Grid.. Grid architecture can be considered a layered architecture made up of layers of different widths arranged in an hourglass shape [11].

- Fabric: This is the bottom layer and consists of resource and connectivity protocols, which facilitate the sharing of individual resources for example, computational resources, storage systems, catalogs, network resources, and sensors [11]. This layer provides the resources shared access mediated by Grid protocols. A simpler fabric simplifies the deployment of Grid infrastructure. The fabric layer should implement enquiry mechanisms for discovery of their structure, state, and capabilities, and resource management mechanisms to deliver Quality of Service (QoS). The types of resources manipulated by the fabric can be computational, storage, network, code repositories, and databases. More prosperous fabric functionality provides more sophisticated sharing operations into Grid environment.
- Connectivity: This layer defines communication and authentication protocols for network transactions. The goal is to provide easy and secure communications. Communication protocols include Internet (IP), transport (TCP, UDP), and application (DNS, and so on), with space for new protocols as the need arises [12].

  Authentication protocols should be able to provide the following:

- Single sign on: Users must be able to “log on” (authenticate) just once and then have access to multiple Grid resources defined in the Fabric layer, without further user intervention [11].
- Delegation: To allow a program to run on the user’s behalf so it is able to access the resources on which the user is authorized [12].
- Integration with local security solutions: Each site or resource provider may employ any of a variety of local security solutions, including Kerberos and UNIX security [11]. Grid security solutions must be able to interoperate with these various local solutions.
- User-based trust relationships: In order for a user to use resources from multiple providers together, the security system must not require each of the resource providers to cooperate or interact with each other in configuring the security environment [11].
- Resource: This layer defines protocols for secure negotiation, initiation, monitoring, control, accounting, and payment of sharing operations on individual resources. These protocols deal with individual resources and ignore global state and atomic actions across distributed collections that are handled by the collections layer [12]. Resource layer protocols can be distinguished into two primary classes: Information protocols used to obtain information about configuration, load, or usage policies [11]. Management protocols that negotiate access to shared resources by handling resource requirements and operation(s) to be performed. Management protocols ensure consistency of operations for a given shared resource [11].
- Collective Layer: This layer defines protocols and services global in nature and captures interactions across collections of resources. Examples of collective protocols are the following[12]:

  Directory services: For resource properties discovery.

  Coallocation, scheduling, and brokering services: for allocation of one or more resources for a specific purpose and the scheduling of tasks.

Monitoring and diagnostics: For failure, intrusion detection, overload, and so on.

Data replication services: For storage management to maximize data access performance.

Grid-enabled programming systems: To provide a programming model for resource discovery, security, allocation, and others.

Workload management systems: For description and management of multi component workflows.

Software discovery services: For optimal software selection.

Community authorization servers: To enforce community policies governing resource access.

- Applications: This layer includes user applications that operate within a VO. Applications call on services defined at any layer resource management, data access, resource discovery and so on. Applications rely on application programming interfaces (APIs) implemented by software development kits (SDKs), which in turn call Grid protocols to interact The Roadmap to High-Performance Computing with network services that provide capabilities to the end user. Protocols may implement local functionality or interact with other protocols [11].

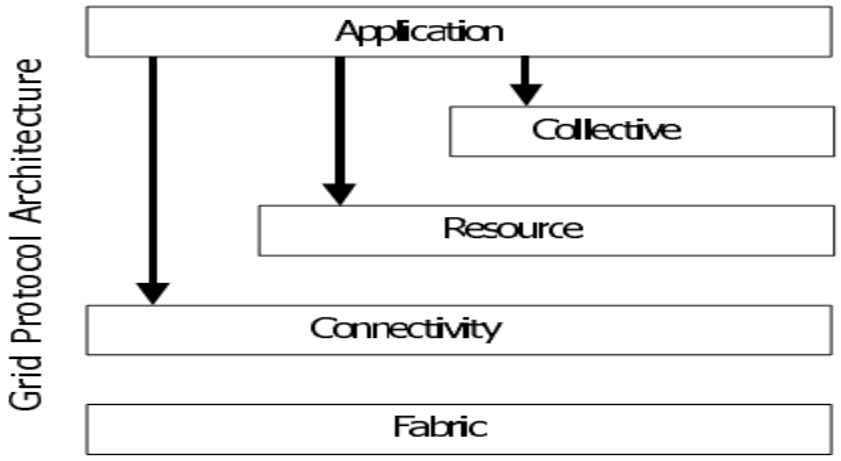

Figure 1.2: The layered Grid Architecture [11]

1.2.4 Virtual Organization

The combinations of two or more organizations that share resources become a VO. This environment effectively creates a dynamic sharing relationship between providers and consumers. These dynamic relationships may be defined by policies that govern access to resources. New organization can be added any time into the VO. The VO increases the use of underutilized resources. VOs enable disparate organizations or individuals to share resources in a controlled fashion to achieve a common goal [13]. The privileges the organization wants to grant the user, related to the tasks he is supposed to perform, are connected to user roles [6]. Authentication is used for the user's identity proof and is commonly encapsulated within X.509 public key certificates. Authorization to join a VO is done on the basis the policies of VO. The participants and users of the Grid can be members of several real and VO. The Grid can help in enforcing security rules among them and implement policies, which can resolve priorities for both resources and users [7].

1.2.5 Grid Middleware

Grid middlewares are distributed computing software that integrates network connected computing resources. Grid middlewares are fundamental part of a Grid infrastructure which is proposed to provide a layer between Grid applications and the low-level functionality of the Grid. Grid middleware makes Grid computing possible. Grid middleware offers services that combine users with remote resources through resource brokers. It is a set of protocols and core services used for resource access, data movement, name resolution, authentication, authorization, resource discovery and resource management etc. Middleware hides complexities of the Grid, enhances functionality and guarantees interoperability between different Grid domains and platforms. With multiple organizations involved in joint research collaborations, issues pertaining to security (authentication and authorization), resource management, job monitoring, secure file transfers, etc. are of paramount important issues [13]. Grid middleware is a mediator layer between user and Grid resource to facilitate a consistent and homogeneous access to resources managed locally with different syntax and access methods. The services offered by Grid middlewares include services for remote process

management, co-allocation of resources, storage access, information, security, authentication, and quality of service such as resource reservation and trading [24]. Grid middleware provides facility of the discovery and utilization of scientific data, computers, software, and instruments over the network in a controlled manner integrating remote resources and collaboration capabilities into local experimental, computational, and visualization environments; and managing, in a community setting, the authoring, publication, curation, and evolution of scientific data, programs, computations, and other products.

Varies Grid middlewares that are in use today given below -

1. GLOBUS TOOLKIT
2. UNICORE
3. CONDOR G

Grid middleware provides the following benefits:

- Resource allocation and process management.
- Authentication and security services.
- Resource information and discovery services.
- Remote access to data via sequential and parallel interfaces.
- Advanced resource reservation.

1.2.6 Grid Information Services

Grid information services (GIS) play an important role in the Grid infrastructure as repositories of resource information and information provider. A GIS is a large-scale distributed middleware that provides information about resources in the net for virtual organizations. GIS is a software component that maintains information about people, software, hardware and other services [1]. GIS provide binding, discovery and data protection information available on request. It provides scalable support for a large and a rapidly growing number of data objects and elegant representation of complex relationships between data objects. It provide high performance support for frequent and

complex updates to data objects, their attributes, and their relationships, including for objects that may belong to different administrative domains and high performance support for complex and evolving queries on data objects and their relationships, in particular for compositional queries. It uses two main functions Grid Resource Information Service (GRIS) and Grid Index Information Service (GIIS) to gather information. A Grid Portal should provide access to the GIS to enable users to publish and retrieve static and dynamic information about Grid resources such as the Globus Monitoring and Discovery services. Information providers form a common, VO-neutral infrastructure providing access to detailed, dynamic information about Grid entities. A large, distributed collection of generic information providers provide access to information about individual entities, via local operations or gateways to other information sources. Information is structured in term of a standard data model, taken from LDAP [29]. High performance execution in Grid environments relies on timely access to accurate and up-to-date information related to distributed resources and services: experience has shown that manual of default configuration hinders application performance [30]. Since Grid environments are increasingly adopting architectures that are distributed and rather dynamic collections of resources and services, discovery of both resources and services becomes a difficult and complex task on large-scale Grids. Thus, it is immediately evident the fundamental role of a GIS [30].

1.4 Thesis Organization

The objective of the Thesis is to reengineer the core functionality of an existing Web-based Grid front-end, Grid Portal based upon GridSphere. This Thesis addresses the requirements for developing a Portal. portlets are Created and deployed into the GridSphere Portal to make it more efficient and handle user problem easily.

This thesis has been organized as:

Chapter 2: A survey of varies first and second generation Grid Portal are given in this chapter.

Chapter 3: Chapter 3 gives a brief introduction of Center of Excellence Grid center of Taper University and need of a Grid Portal.

Chapter 4: Design aspects of a Grid Portal are mentioned in this chapter.

Chapter 5: In chapter 5 implementation of Grid Portal and creation and development of Resource discovery portlet and Job Submission portlet have been given.

Chapter 2

Literature Survey

Grid computing has the potential to provide users on-demand access to large amounts of computing power. To access a Grid resource a Grid Portal is needed. A Grid Portal is a Web-based gateway that provides seamless access to geographically dispersed backend resources. Computational Grids provide users with seamless access to computational power to solve a problem. It's necessary that user have a good knowledge of the Grid resource to use them in the best manner. But gathering information about the Grid resource is difficult task for user. The solution of this problem can be achieved by developing user friendly interfaces for the Grid environment. Such Interfaces are called Grid Portals. A Grid Portal acts as an information aggregator which can access dissimilar information about the resources. The Grid Portal conceals the complexity of the Grid and offer users with a transparent accessing technique to the computational resources of the Grid. It is an environment where the user can access Grid resources and services, execute and monitor Grid applications and collaborate with other users. The Portal provides consistent "Grid context" for the user's Grid interactions [2].

2.1 Grid Portal

A Grid Portal acts as a user's point of access to a Grid resource in Grid environment. The most important necessities for a Grid Portal system from a user's viewpoint involve access to Grid services. In order to identify the services a Grid Portal should provide, the potential user communities must first be defined and their requirements of the Portal understood. Grid Portals are the most popular means for delivering user interfaces to Grids. Grid Portals build upon the familiar Web Portal model to offer virtual communities of users a single point of access to computational resources [14]. A Grid Portal is a Web-based gateway that provides faultless access to a variety of backend resources. It also provides a single point of access to Grid-based resources that they have been authorized to use [1]. Grid Portals are the most popular means for delivering user interfaces to Grids. The GridSphere Portal Toolkit is a portlet JSR compliant

portlet container that offers a set of base classes and tools for developing Web ewapplications [14].

Mainly two types of Grid users are there: end users and system developers.

- End users: End users are the scientists and engineers who use the Grid to compute their domain-specific problems perhaps via a Portal [1]. A Grid Portal provides end users with a customized view of software and hardware resources specific to their particular problem domain. They use the Portal for problem solving and collaborate with community members to share data and information.

System developers are those who build Grid systems using middleware packages. System developers are of following type.

- Application Developers: Application Developers are individuals who use Grid services provided by a Grid Portal and the supporting software to develop Grid-enabled applications and components for end user.
- Portal Developers: They design and implement a Grid Portal satisfying requirements of the Portal's user communities. Portal developers also include new incorporating functionality into a Portal to improve the accessibility of Grid resource and service by the end user and providing Portal enhancement.
- Portal Administrators: Portal Administrators are responsible for managing and maintaining the Portal infrastructure. Their main task is to take care of technical issues, such as deployment and management of the Portal, configuration of the application server and installation of software and Portlet into the Grid Portals. New generation Portals have built-in Portal functionalities to support administrator's tasks.

The basic model of Grid Portal can be classified into three layers from the point view. The first layer is user layer; it can provide the interface for user. User layer is responsible for displaying the Portal content. It can be web browser, or other desktop tools. The second layer is service layer, including authentication, job management service, information service, file service, security service. The third layer is resource layer, including remote compute, data and application resources [15]. A Grid Portal provides a

customizable interface allowing scientists to perform a variety of Grid operations including remote job submission, file staging, and querying of information services from a single, secure gateway [16]. My Proxy and Globus GSI are used for user authentication and authorization. MyProxy allows a Portal to use the GSI to interact with resources in a standard, secure manner [17].

2.1.1 Grid Portal Architecture

Usually Portals are generally based on typical three-tier web architecture [1] [13]. Figure 2.1 has been shown three tier architecture of Grid Portal.

- Web browser or any device a user wishes to use to access the Portal. It will be transferred to Grid I/O operation.
- To obtain the information of all the heterogeneous resources, we added a layer between the Grid resources and Portal layer called Application server or Web server which can handle HTTP request from the client browser. The application server is responsible for handling HTTP/S requests generated from a client and is necessarily multi-threaded to allow multiple concurrent client connections.
- The bottom layer of Grid environment to access backend resources that include computing resources, databases, etc.

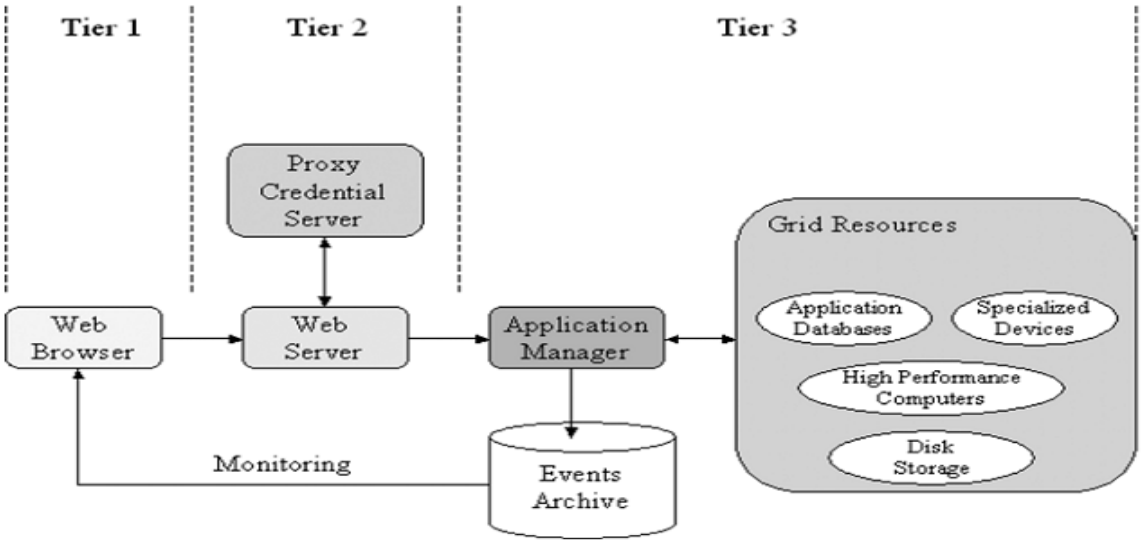

Figure 2.1: Grid Portal Architecture [11]

Grid Portal are mainly divided in two categories:-First-Generation Grid Portals and Second-Generation Grid Portals.

2.2 First Generation Grid Portals

These Grid Portals provide a uniform access to the Grid resources. They have a dynamic Graphical User Interface based on HTML pages, with JSP or JavaScript. Common Gateway Interface (CGI) and Perl are also used by some Portals. First-generation Grid Portals are tightly coupled with specific Grid middleware technologies such as Globus [1]. The first generation Grid Portals have some limitations, mainly lack of customization, Static and restricted Grid services. Access for end-users is static and it is almost impossible to dynamically customize a Portal to meet their special needs [19]. The first generation of Grid Portals relied heavily on the Globus middleware to provide Grid services. The main reasons for this are that Globus provides a complete package and a standard way in building Grid-enabled systems [20].
Following First Generation Grid Portals have been surveyed**:**

2.2.1 GridPort Toolkit 3.0

The GridPort is a Perl-based Grid Portal toolkit. The GridPort is used for easy development of application-specific Portals. GridPort is a collection of services, scripts and tools. The two key components of GridPort are the Web Portal services and the application APIs. GridPort provides both informational and interactive services for Portals and applications. It uses informational services to supply Grid related data gathered from numerous Grid technologies [1]. GridPort is a toolkit for developing Web-based Portals and applications for computational science on top of underlying distributed and Grid computing infrastructure. GridPort aggregates Grid services from popular Grid software packages and provides additional Grid capabilities, while presenting a simple, consistent API for Portal and applications developers [18]. Gridport's information services known as GPIR gather dynamic and static data about a Grid organized by VO. The back end infrastructure is a PostgreSQL database [1]. The basic architecture of GridPort has shown in Figure 2.2.

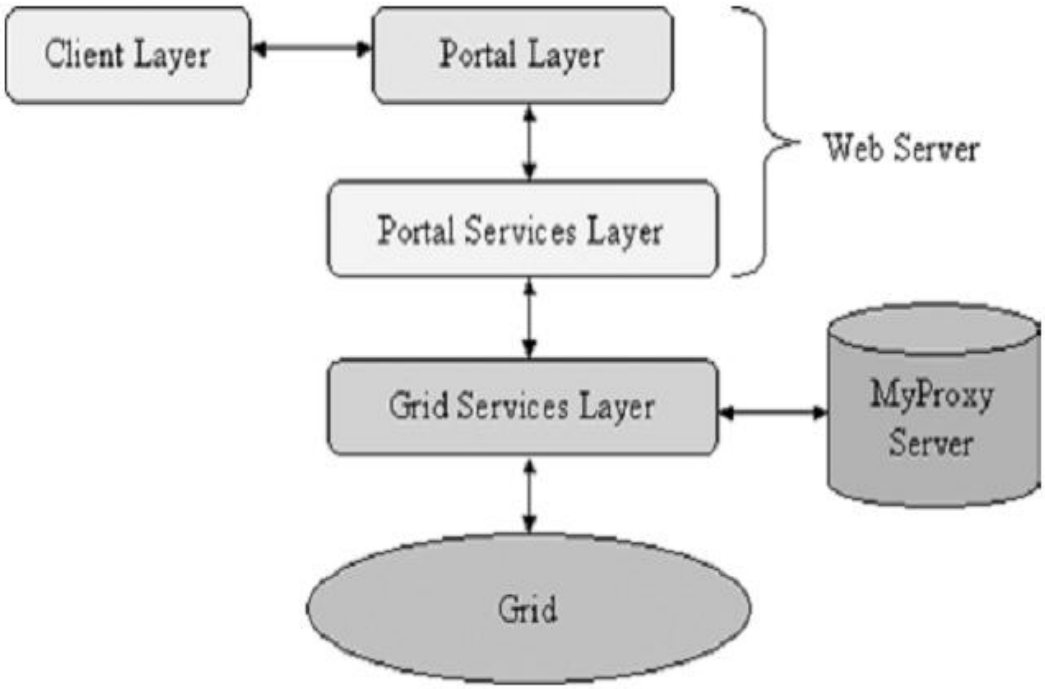

Figure 2.2: GridPort Architecture [1]

2.2.2 Grid Portal Development Kit

Grid Portal Development Kit (GPDK) is designed to provide a complete development environment for building customized application specific Portals that can take advantage of the core set of GPDK Grid service components. It provides a set of reusable components for accessing Globus based Grid services [16]. GPDK uses Java Server Pages (JSPs) for Portal presentation and JavaBeans to access backend Grid resources via Grid middleware [1]. The core of GPDK resides in a set of generic, reusable, common components used for accessing the various Grid services that are supported by the Globus toolkit [16]. A Portal user is provided with a persistent, customizable profile that contains information that is stored securely on the Portal and provides details on past jobs submitted, the set of computers they have access to, and any other information that is of interest to a particular user. The GPDK leverages off existing Globus/Grid middleware infrastructure as well as commodity web technology including Java Server Pages and servlets [16]. The basic architecture of GPDK has been shown in Figure 2.3.

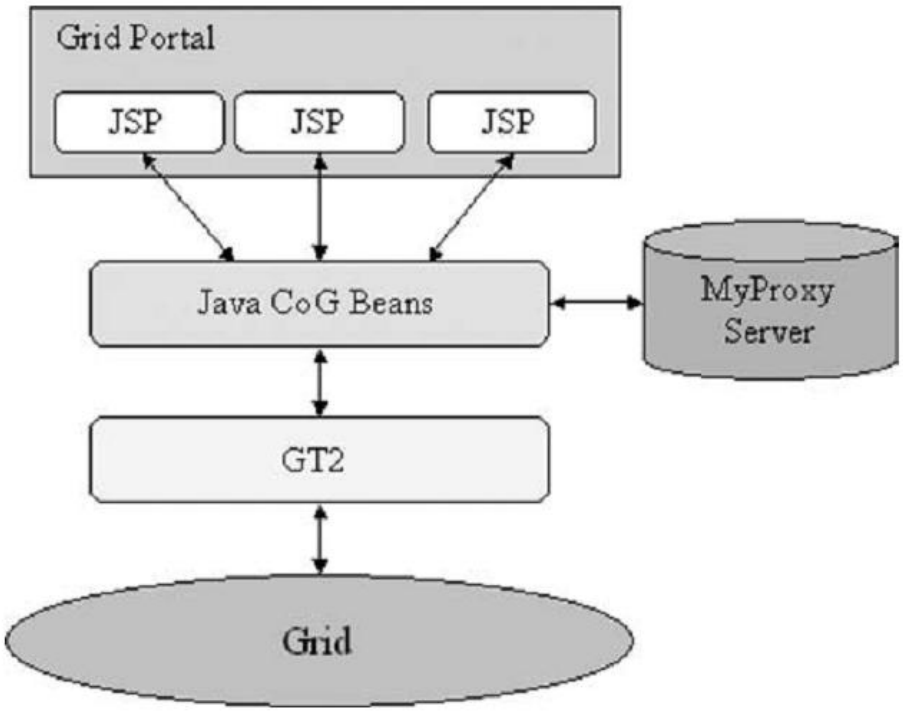

Figure 2.3: Grid Portal Development Kit [1]

2.2.3 The Ninf Portal

The Ninf Portal facilitates the development of Grid Portals by automatically generating a Portal front-end that consists of JSP and Java Servlets from a Grid application Interface Definition Language (IDL) defined in XML. Ninf-G is handled via the security mechanism of the Globus Toolkit [4]. The system alleviates the user's burden by automatically generating JSP system, such as Ninf-G to interact with backend Grid services [12]. The frontend consisting of a set of web pages, would be automatically generated from the interface information described in XML; the backend, a Grid application, would be easily developed using Ninf-G to allow users to write the Grid application if they make a local function call. In the Ninf Portal, the only task that Portal Developers should perform for obtaining the user interface is just describing an XML document, with which the Ninf Portal automatically generates both a JSP file that creates HTML pages on the fly and a general-purpose Java servlet that is capable of handling data [22].

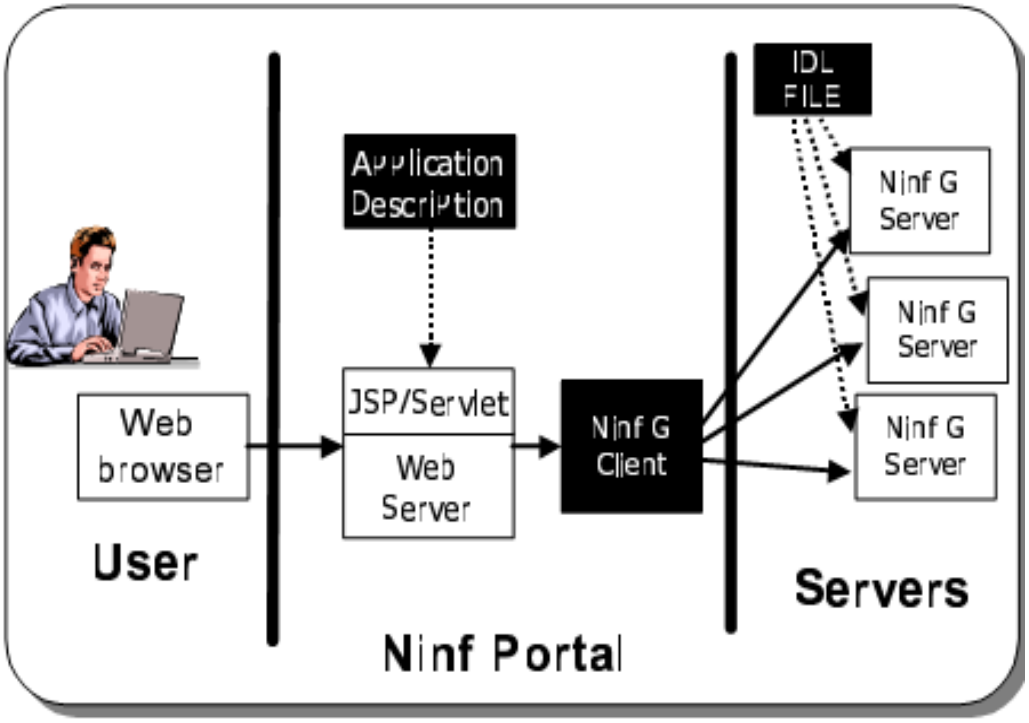

Figure 2.4: The Ninf Portal Architecture [30]

2.2.4 GridSpeed

GridSpeed is a Grid Portal hosting server that automatically generates and publishes a customized web interface to the Grid for applications, with minimal effort required from the user [21]. The main aim of GridSpeed is to hide the complexity of the underlying infrastructure from Grid users [1]. The main feature of GridSpeed is to provide users with capability of dynamically generate application Portals or instantiate and publish them without any deep knowledge of programming. GridSpeed provides a repository called the GridSpeed Portal Repository that allows Portal creators to publish their generated Application Portals and application users to search for their target Applications while specifying the name, category, manufacturer, Physical resources to be used [21]. GridSpeed has enough capabilities to handle real-world scientific applications. Applications run on the Grid in a distributed fashion not in centralized fashion.

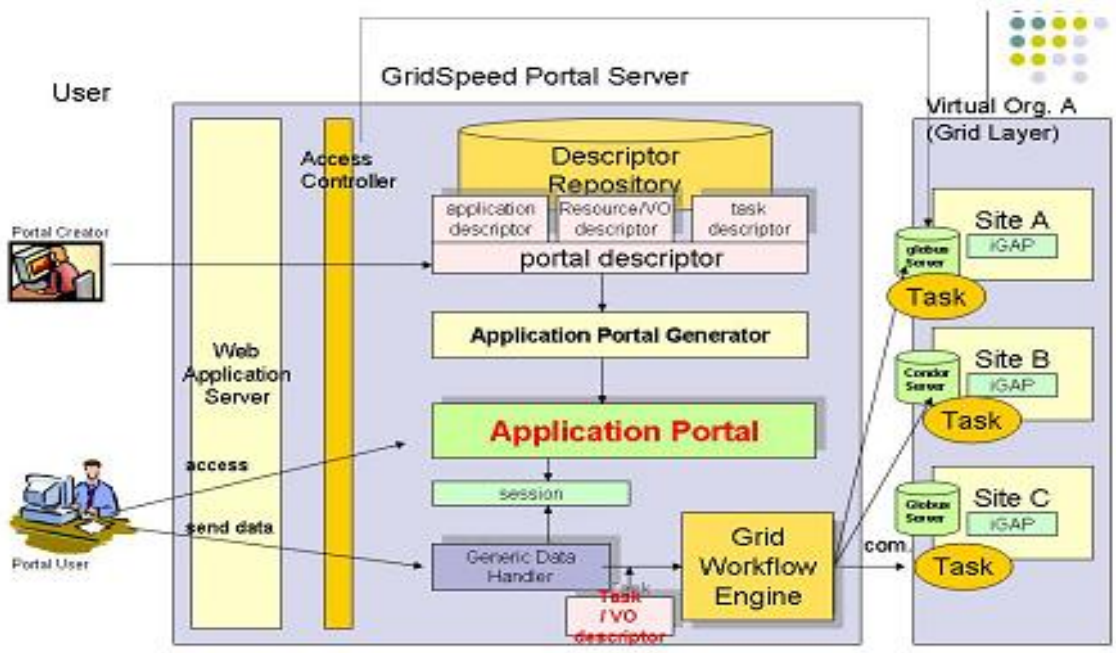

Figure 2.5: GridSpeed Portal Architecture [21]

2.2.5 Comparison of First Generation Grid Portals

On the basis of the above mentioned study the first generation Grid Portals have been compared on the basis of licence, pages used for interface, construction, descriptor, support, and Portal customization. The comparison is shown in table 2.1.

Table 2.1: Comparison of First Generation Grid Portals

Features	GridPort 2.0	GPDK	The Ninf Portal	GridSpeed
License	Open source	Open source	Commercial	Commercial
Portal pages	HTML	JSP	JSP	JSP
Portal Construction	Perl/CGI	JavaBeans	Portal JSP Generation	Portal Generation
Portal descriptor	Not Supported	Not Supported	Application Level	Application/resource/user level
Portal support	User Portal	User Portal	Application Portal	Application Portal
Portal Customization	No	No	No	Being Supported

2.3 Second Generation Grid Portals

The second generation Portal architecture is designed as a Service-Oriented Architecture (SOA) and consists of three layers which are Portal layer, service layer and resource layer [19]. To overcome from the limitation of first generation portlets have been introduced and promoted for use in building second-generation Grid Portals [1]. Portlets are pluggable applications that are designed to run inside the portlet container of a Portal. Portlet container manages and provides persistent storage mechanisms for portlets. Portlets are administered in a dynamical and flexible way. The Portals of second generation are JSR-168 compliant.

2.3.1 Portlet

Portlet is a mini-application that resides in the Portal. Multiple portlets can be composed in a Portal page. Portlet container provides the required runtime environment and manages the lifecycle of the portlet. Grid portlets offers developers a collection of "portlet services" for performing tasks on the Grid. The Grid portlets web application, released for the first time in June 2005, builds upon the core features in the GridSphere Portal toolkit to provide developers with a toolkit for developing Grid-enabled portlets [22]. From a user's perspective, a portlet is a window in a Portal that provides a specific service. From an application development perspective, a portlet is a software component written in Java, managed by a portlet container, which handles user requests and generates dynamic contents. Portlets, as a pluggable user interface components, can pass information to the presentation layer of the Portal system. Portlets in an application are installed as a single package. A portlet is a Java-based Web component that processes requests from a portlet container and generates dynamic content. The content of a portlet is normally aggregated with the content of other portlets to form the Portal page [1]. Portlets serve as the content that is plugged in to a Portal toolkit, and this content can conform to the JSR-168 specification [23].

2.3.1.1 Classification of Portlets

- Proxy Manager Portlet: They are used for retrieving credential from MyProxy servers.
- File Manager Portlet: They are using GridFTP to browse the contents of remote file system and upload/download files to/from the desktop.
- Job Submission Portlet: They are using GRAM to submit job for execution remotely.
- Information Services Portlet: They are using GPIR to display the current features and status of various storage and compute resources within or across organizations.

2.3.1.2 Key Portlet Services

Followings are the general services provided by the portlet [22].

- Resource Registry Service: The Resource Registry Service is responsible for making resources available to portlets and other portlet services, where resource descriptions are maintained in the Grid portlets database.
- Resource provider services: Grid portlets supports the ability to discover and monitor resources with resource provider services, where resource provider services are responsible for querying (or registering Grid portlets for notification) the various information resources that have been registered with Grid portlets.
- Credential Manager Service: The Credential Manager Service is responsible for managing credential contexts and making active GSS credentials that are registered for those contexts available to portlets and other portlet services.
- Credential Retrieval Service: Grid portlets supports the ability to retrieve GSS credentials from remote credential repository resources, such as MyProxy, with the Credential Retrieval Service. The Credential Retrieval Service is responsible for managing credential retrieval contexts. A credential retrieval context associates information for retrieving a credential with a particular DN to a Portal user.

- File Browser Service: The File Browser Service provides methods for creating file browsers to remote file resources. A file browser represents a stateful connection to a file resource and provides methods for listing files, changing directories, creating directories, transferring files between locations and other basic file operations.
- Logical File Browser Service: Grid portlets offers a Logical File Browser Service for creating logical file browsers to logical file resources. Logical File Browser Service naturally extends the File.
- Job Submission Service: The Job Submission Service, as described above, is a service for submitting jobs to remote job resources. Job resources are service resources that provide access to one or more job schedulers, where job schedulers define one or more job queues to which jobs can be submitted. The Job Submission Service can support job submission to multiple types of job resources at runtime.

2.2.1.3 Portlet Architecture

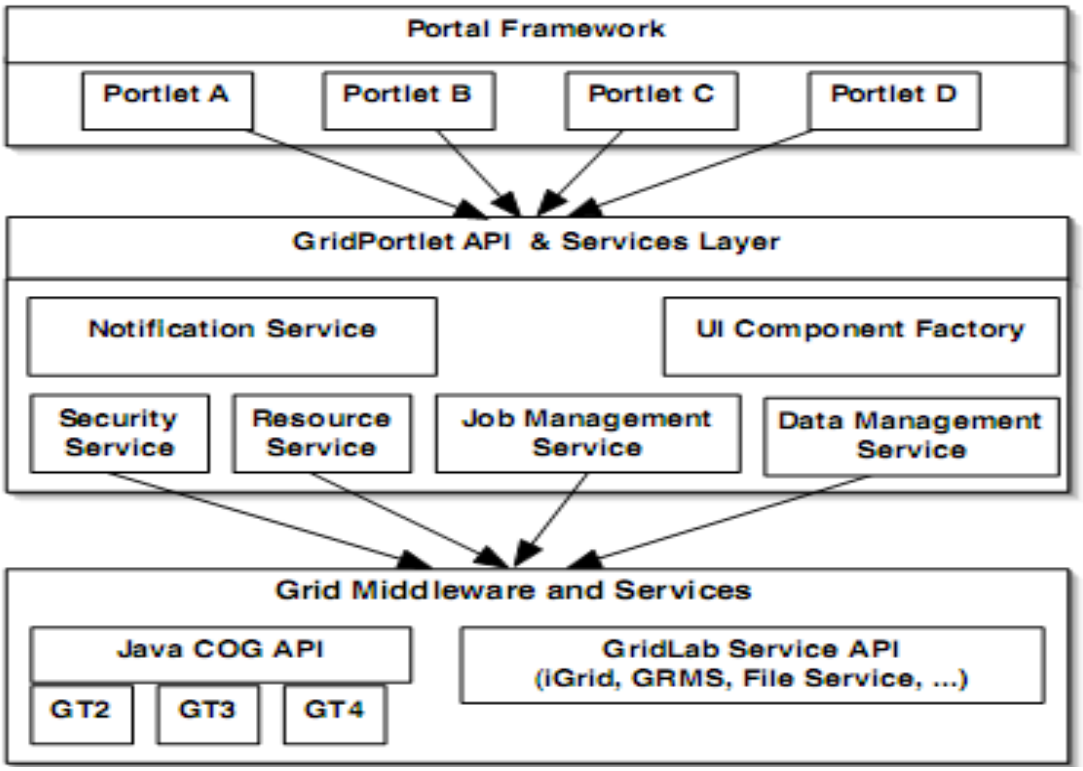

Figure 2.6: Grid Portlet Architecture [23]

Figure 2.6 shows the general architecture of Grid portlets, where a common and clearly defined API abstracts developers from underlying services and middleware. A simple execute task is constructed that does not require any details about the underlying implementation, which may be the Globus Resource Allocation Manager (GRAM) via the Java Commodity Grid Toolkit (Java CoG) or an advanced resource brokering system such as the GridLab Resource Management System (GRMS). Gridportlets contains many reusable User Interface (UI) components that can easily be exploited to develop other portlet based applications [23].

2.3.1.4 JSR-168 Standard

JSR 168 is a JCP portlet specification that provides a set of standerd Java APIs that permits interoperability between Portals and portlets. Java Specification Requests (JSRs) are the actual descriptions of proposed and final specifications for the Java platform. It defines run-time environment for the portlet container and mechanisms to store transient and persistent data for portlets [13]. The Java portlet Specification achieves interoperability among portlets and Portals by defining the APIs for portlets. The JSR-168 standardizes how components for Portal servers are to be developed. This standard has industry backing from major Portal server vendors. JSR-168 defines a common portlet API and infrastructure that provides facilities for personalization, presentation, and security [12].

2.3.1.4 Portlet Container

A portlet container provides a run-time environment in which portlets are instantiated, executed and finally destroyed. A portlet container manages and provides persistent storage mechanisms for portlets. A portlet container is not a standalone container like a Java Servlet container; instead, it is implemented as a layer on top of the Java Servlet container and reuses the functionality provided by the Servlet container [1]. The portlet container is responsible for the execution and lifecycle management of portlets. The controller Servlet, GridSphere Servlet, uses the portletManagerRegistry to handle the administration of portlets including installation, removal, initialization and shutdown. GridSphere application can be run and administered by portlet container [25].

Following Second Generation Grid Portals have been surveyed:

2.3.2 JetSpeed

JetSpeed is an open-source project from the Apache Software Foundation for building Portals with portlets in Java. JetSpeed is a Portal a well as a Portal Integration Toolkit and fully open and standards based, JSR-168 compliant [1]. A Portal user interface is composed of multiple portlets that each provides access to a particular service or set of services as well as providing content from information providers [26]. In JetSpeed, persistence services are available for all portlets to provide store state per user, page and portlet. JetSpeed is customizer for selecting portlets and defining layouts for individual pages [1]. The general architecture of JetSped is shown in Figure 2.7.

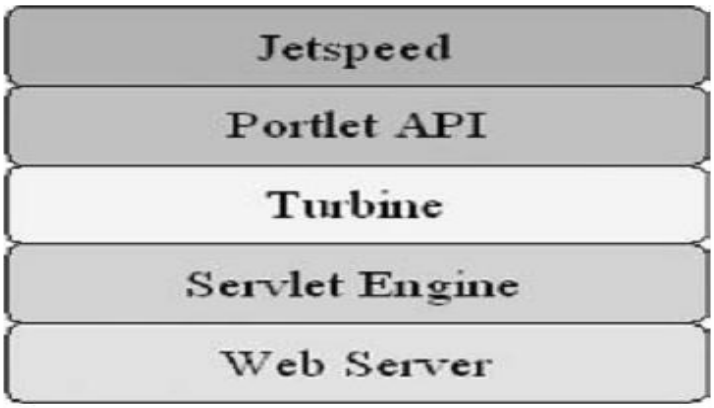

Figure 2.7: The Architecture of Jetspeed [1]

2.3.3 IBM WebSphere Portal

The WebSphere Portal is a J2EE application that runs on WebSphere application Server. Its main functions are to serve the WebSphere Portal toolkit to desktops and mobile devices of Portal users and allow a user to integrate applications and data sources, and also perform administrative tasks, such as controlling Portal membership [1]. The WebSphere Portal provides score Portal services including single sign-on, security, Web content publishing, search, personalization, collaboration services, enterprise application integration, support for mobile devices, and Site analytics [20]. The WebSphere Portal forms an environment that provides the connectivity, administration and presentation services required for the end user. It provides the versatile toolkit, content management,

collaboration and integration of application, customization and security by Portlats for Grid Environment. It supports a multi-user environment by managing workflow, security and administration. WebSphere is an extensible toolkit for building and delivering Portals providing the ability to access and interact with internal and external resources, which can include applications, content, people, and processes [20]. Presentation services are used to create Portal–user interface with a graphical user interface that can be customized to match the user's specific needs. A user can be configure Portal to render content on different client device types, such as laptops and mobile phones [1].

2.3.4 GridSphere

GridSphere is an open-source portlet-based Web Portal that can facilitate developers to rapidly develop and package third-party portlet web applications that can be run and administered within the GridSphere portlet container. One of the key elements in GridSphere is that it supports administrators and individual users to dynamically conFigure the Portal content based on their requirements [27]. GridSphere provides compatibility with the JSR-168 standard for portlet API and near full compatibility with the WebSphere. Application toolkits are designed to address this limitation by providing a semi-complete domain-specific application that can be specialized to produce custom applications. The goal of the GridSphere Portal toolkit is to make the development of new Portal interfaces and offer new functionality as easy as possible [28]. The primary core services of GridSphere are the LoginService, the UserManagerService and the AccessControlManagerService [13]. GridSphere use portlets for user authentication and account management, job management and other Grid service. The GridSphere portlet container is implemented as a web application and requires a hosting environment such as the Jakarta Tomcat container. Many additional libraries are used and deployed to the servlet container during installation [13]. The architecture diagram shows the primary components that combined with the servlet container comprise the web application server. A request to the Portal triggered by a user's web browser invokes the GridSphere servlet which acts as a controlling dispatcher to the layout engine responsible for rendering suitable output to the user's browser [13]. The GridSphere servlet and layout engine both make use of core services, including the portlet registry, to invoke the

appropriate set of portlets in a user's Portal page. Layouts in the Portal are defined as XML descriptor files. Each layout component defined in GridSphere e.g. portletFrame, portletTabbedPane, portletTab, PortetContent, etc itself adheres to a portletComponent interface which defines a set of lifecycle methods that closely maps to the lifecycle methods of portlets [13]. The Layout Engine provides vast flexibility and is entirely customizable. New components can simply be added by creating an XML descriptor element and a corresponding component class facilitate that implements the portletComponent interface.

2.3.4.1 GridSphere Architecture

The architecture of GridSphere is shown in Figure 2.8 shows the primary components that combined with the servlet container comprise the web application server. A request to the Portal triggered by a user's web browser invokes the GridSphere servlet which acts as a controlling dispatcher to the layout engine responsible for rendering suitable output to the user's browser [13]. Both the GridSphere servlet and layout engine make use of varies core services, including the portlet registry and other, to invoke the suitable set of portlets in a user's Portal page. The GridSphere architecture outlines both a general Portal toolkit used to assist virtual organizations comprised of scientists and project developers as well as architecture for the development of reusable, modular components that serve to access the services being developed within the GridLab project [28].

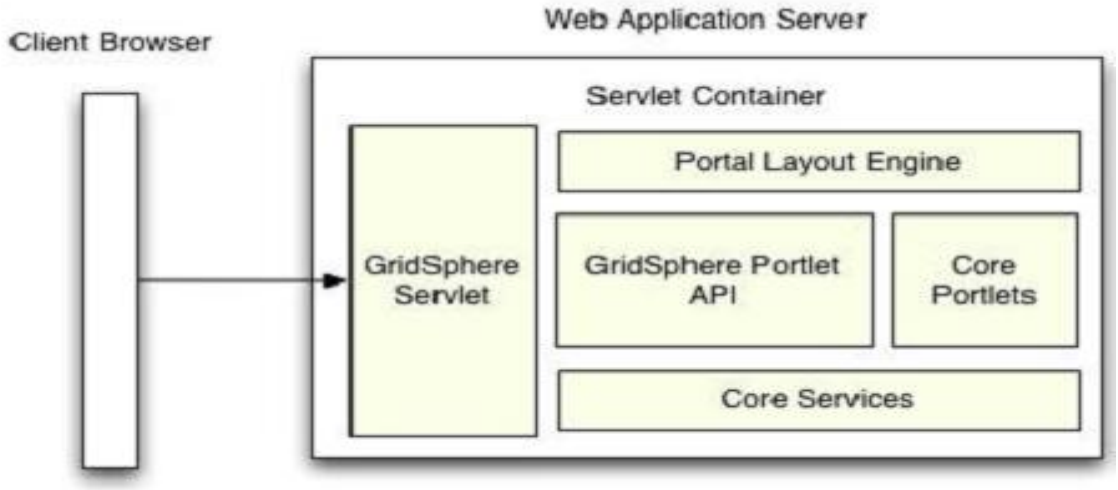

Figure 2.8: GridSphere Portal Architecture [13]

2.3.4.2 GridSphere Features

The portlet API implementation in GridSphere is almost fully compatible with IBM WebSphere Portal and all portlets are fully JSR complaint. GridSphere Support easy and fast development and integration of third-party portlets that can be plugged into the GridSphere portlet container on the user need. A high-level model for building complex portlets using visual beans and the GridSphere User Interface (UI) tag library and a flexible XML-based Portal presentation description that can be modified to create customized Portal layouts [1]. GridSphere provide an included support for Role-Based Access Control (RBAC) in which a user can be guest, user, administrator and super users. GridSphere is a portlet service model that permit developers creation of user services, these service methods can be limited based upon user rights. It provides Persistence of data provided using Hibernate for RDBMS database support. It has Integrated Junit and Cactus unit tests for complete server-side testing of portlet services including the generation of test reports [13]. GridSphere has in build core portlets that provide basic functionality such as including login, logout, user authentication, locale selection and access control management. Localization support in the portlet API implementation and GridSphere core portlets that support English, French, German, Czech, Polish, Hungarian and Italian [1]. GridSphere supports persistence of data using Hibernate and OQL for database support. It has sophisticated portlet service model that can encapsulate reusable portlet logic into services that may be shared between many portlets and support for portletizing Struts applications using the Portals Struts Bridge. GridSphere is Open-source and 100% free.

2.3.4.3 GridSphere Core Portlets

GridSphere comes with built-in basic portlets for carrying out administrative tasks such as management of user's account, portlets, layout and messaging.

Following core portlets are already included in the GridSphere [13].

- Login Portlet: This portlet is used to restrict access to the Portal and allows users to login based on a name and password.
- Logout: Logs a user out of the Portal.

- Locale Selection: A user can select from English, French, German, Italian, Hungarian, Czech and Polish etc language according to him.
- Account Request; Provides interface for a new Portal user to request an account and optionally choose groups to join.
- Account Management: Provides the Super user and Admin users the ability to assign/revoke users roles.
- User Profile: Provides the ability for users to edit personal information including name, email address and select portlet applications to join.
- User Management: Provides the Super user the ability to approve/deny account requests, and Admin users the ability to approve/deny group requests.
- Portlet Subscription: Provides the ability for users to add and remove portlets from their workspace.
- Layout Configuration: Users can configure their layout including placement of tabs and portlets within tabs.
- Local File Manager: Provides a virtual filesystem allowing users to edit, upload and download files to the Portal

2.3.5 Comparison of Second Generation Grid Portals

On the basis of the above mentioned study the Second Generation Grid Portals have been compared on the basis of licence, complexity, reusability, documentation, installation, hardware requirement, creating new portlet, stability and security. The comparison based on survey is shown in table 2.2.

Table 2.2: Comparison of Second Generation Grid Portals

Features	GridSphere	IBM WebSphere	JetSpeed
License	Open Source	Commercial Product	Open Source
Complexity	Low	High	Low
Re-use of Publicly Available Components	Good – deploys into Apache Tomcat and uses log4j	Poor proprietary implementations of application server, logging package and http server	Easily deploys into Apache Tomcat
Documentation	Good – about the right amount	Almost too much things are hard to find	Good and easy to find
Installation/Configuration	Easy & Quick	Cumbersome	Easy & Quick with Globus and tomcat
Special Hardware Requirements	No	At least 1 GB memory per processor and 4GB disk space	Not required
Debugging Your Own portlets	Easy – log4j can be used	Hard: in-code debugging	Yes with log4j
Stability	Not very stable, but getting better.	Fairly stable	Fairly good
JSR-168 Compliance	Full	Full from version 5.1 onwards	Full
Security	Basic Login & RBAC	Strong	use a role based entitlement

2.4 Conclusion

This chapter presented detailed survey of presently existing Grid Portal toolkits. Comparison of first generation grid Portal and second generation grid portals have been done. Next chapter will focus on the design of grid portal with the help of UML diagrams.

Chapter 3

Development Aspects of Grid Portal

This chapter describes the design aspects of a Grid Portal. The main aim of our Portal design will be providing user with a friendly and an easy interface for providing Resource discovery, Job submission and Job monitoring services. The design of the Grid Portal will offer flexibility and support for many different needs and requirements of the Portal users. This chapter seeks to provide the Software Requirements Specifications and UML diagram for the design of new Grid Portal.

3.1 Software Requirements Specification

3.1.1 Introduction

This SRS describes a specification required for a Portal development that is based on GridSphere. This document highlights the capabilities and requirements that are needed for standardized the Portal.

- Purpose

  This document seeks to provide the Software Requirements Specifications for the design of a new Grid Portal, and developed with the help of presently exist Grid Portal. This document will serve as a guideline for future Grid Portal related work. Furthermore, this document presents an initial description of the various functionalities and services that Grid Portal expects to invoke and/or incorporate from other Grid resource.

- Scope

  The proposed Grid Portal provides a user friendly abstract view to access Grid computational and storage resource, it is a perfect tool for anyone who would like solve computational and/or data intensive problems on large scale distributed Grid environments.

- Definition

  Grid Computing: Grid Computing has emerged as a promising next generation collaborative problem solving platform for industry, science, and engineering. Grid computing is defined as coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. Grids enable the sharing, selection, and aggregation of a wide variety of resources including supercomputers, storage systems, data sources, and specialized devices that are geographically distributed and owned by different organizations for solving large-scale computational and data intensive problems in science, engineering, and commerce.

  Grid Portal: Basically a Grid Portal acts as an information aggregator which can access different information resources. The Grid Portal hides the complexity of the Grid and provides users with a transparent accessing way to the computational resources of the Grid. A Grid Portal is a user's point of access to a Grid system.

  Portlet: From a user's perspective, a portlet is a window in a Portal that provides a specific service. From an application development perspective, a portlet is a software component written in Java, managed by a portlet container, which handles user requests and generates dynamic contents. Other term that are used in development of a Grid Portal are portlet Container, MyProxy, JSR-168, Grid information service, Grid Middleware and Virtual Organization etc.

- Overview

  This task will propose a Grid Portal which is used by the end user to access the backend resource of Grid. Such Portals are made to reduce the end user load and communicate with different Grid. The Proposed Grid Portal will be deployed on Apache Tomcat with the help of ant.

3.1.2 The Overall Description

The proposed Grid Portal is a web based, service rich environment for the development, execution and monitoring of workflows and workflow based parameter studies on various Grid platforms. It hides low-level Grid access mechanisms by high-level graphical interfaces.

- Product Perspective

  GridSphere is a portlet Container. It is JSR 168 and WebSphere portlet API compliant. The Grid Portal are being developed to provide a single point of access to computational resources like clusters, data servers, applications, scientific instruments and computing services on the Grid computing system. Grid-enabled computational Portals have become a widely-used way of accessing distributed heterogeneous resources and of supporting collaboration between project partners.

- Software Interface

  The other required software products for developing a Grid Portal are JDK, apache Ant, Apache Tomcat. The JDK minimum version is 1.4.2+ which is http://java.sun.com/j2se. Jakarta Ant minimum version is 1.6+ which is available on http://ant.apache.org. Ant is used to compile and deploy the GridSphere framework. Jakarta Tomcat Servlet Container minimum required version is 4.1+, which is available on http://tomcat.apache.org Tomcat 5.0.2x is best for use to deploy GridSphere.

- User Characteristic

  The users of the Grid Portal will be people interested in submitting jobs to Grid environment for computation. Users can access a Grid through the Grid Portal (web browser). User can be Scientist, Engineer or a researcher.

- Product Feature

  The Grid Portal will need to have an internet connection to communicate with Grid environment. The Grid Portal is a portlet JSR compliant portlet container that offers a set of base classes and tools for developing Web applications. It can be supported with Globus Toolkit and other service-oriented technologies.

  Resource Discovery Portlet: Grid resources are mainly connects with administrative domains. Grid resource discovery is not only based upon the core functionality of a resource or the service but also on the bases of the function they perform.

  Job Submission Portlet: A Portal provides users with the ability to submit a job, manage their job tasks, monitor the status of tasks and pausing or cancelling tasks if necessary.

3.1.3 Specific Requirements

Communication Interfaces: The internet connection is required for access the Grid environment and problem execution on distributed Grid resources. All communication between end user and the Job scheduler will be via Web Services protocols, second generation web services are required in order to maintain stateful information.

- External Interface

  Access to the Grid Service user will be provided through the Web-based User Interface. The users of the Grid Portal are not need to know more about backend resource, that are used to compute there problem. User Interface should be simple to understand by provide help on related functions and should be GUI.

- Software Components

  The Grid Portal will require several software packages/programs to be installed.

  - JDK 1.4.2+http://java.sun.com/j2se.

- Jakarta Ant 1.6+ Ant is used to compile and deploy the GridSphere framework.
- Jakarta Tomcat servlet Container Tomcat 4.1+, Tomcat 5.0.25+, is required.
- Grid Middleware Software Component.

Grid middleware will be required to have the distributed communication. The middleware must have security services and have second generation web interface capabilities.

- Software System Attributes

  Reliability: The proposed Grid Portal is based on Java language so it is reliable for the user and provides a seamless access to the Grid environment.

  Availability: The proposed Grid Portal is installed on server, so it is available 24 hours of a day and 7 day in a week.

  Security: Maintain an organizational MyProxy service to enable users to store & retrieve short- lived GSI proxy certificates with our Web services. General Security Infrastructure (GSI) based authentication and authorization should be used for the end user or developer. Maintain secure HTTP communication between user applications and Grid services.

  Maintainability: This Grid Portal is based upon GridSphere. GridSphere has capability of reuse existing portlet and quickly development of new portlet based upon requirement.

  Portability: The proposed Grid Portal is portable between different web browsers.

3.2 Grid Services

The main aim of our proposed work is to provide basic Grid services to the end user via Center of Excellence Grid Portal. The main portlets that are deployed into the Center of Excellence Grid Portal are Resource Discovery portlet, Job Submission portlet and Job monitoring. Every portlet provides some specific services. Portal must provide for

resource discovery, resource allocation, remote job submission and cancellation, job monitoring, job input/output, etc.

Authentication Services: When users access the Grid via a Portal, the Portal can authenticate users with their usernames and passwords.

Resource Discovery Services: Grid resources are mainly connects with administrative domains. Grid resource discovery is not only based upon the core functionality of a resource or the service but also on the bases of the function they perform.

Job management services: A Portal provides users with the ability to manage their job tasks, monitor the status of tasks and pausing or cancelling tasks if necessary.

Data transfer services: A Portal allows users to upload input data sets required by tasks that are to be executed on remote resources. Similarly the Portal allows results sets and other data to be downloaded via a Web browser to a local desktop.

Information services: A Portal uses discovery mechanisms to find the resources that are needed and available for a particular task. Information that can be collected about resources includes static and dynamic information such as OS or CPU type, current CPU load, free memory or file space and network status.

GridSphere provides so many services but the main aim of Center of Excellence Grid Portal is Resource Discovery service and Job Submission service.

3.2.1 Resource Discovery Service

Grid resources are mainly connects with administrative domains. Grid resource discovery is not only based upon the core functionality of a resource or the service but also on the bases of the function they perform. Resource discovery service occupies searching of resources that fulfill the user's application requirements. Flexible, safe and synchronized resource allocation among VOs needs the accessibility of a wealthy resource information environment to support easy resource discovery and decision making for end user. The resources are reserved in advance for future use. Usecase Diagram and Sequence Diagram of Resource Discovery are shown in Figure 3.1 and 3.2.

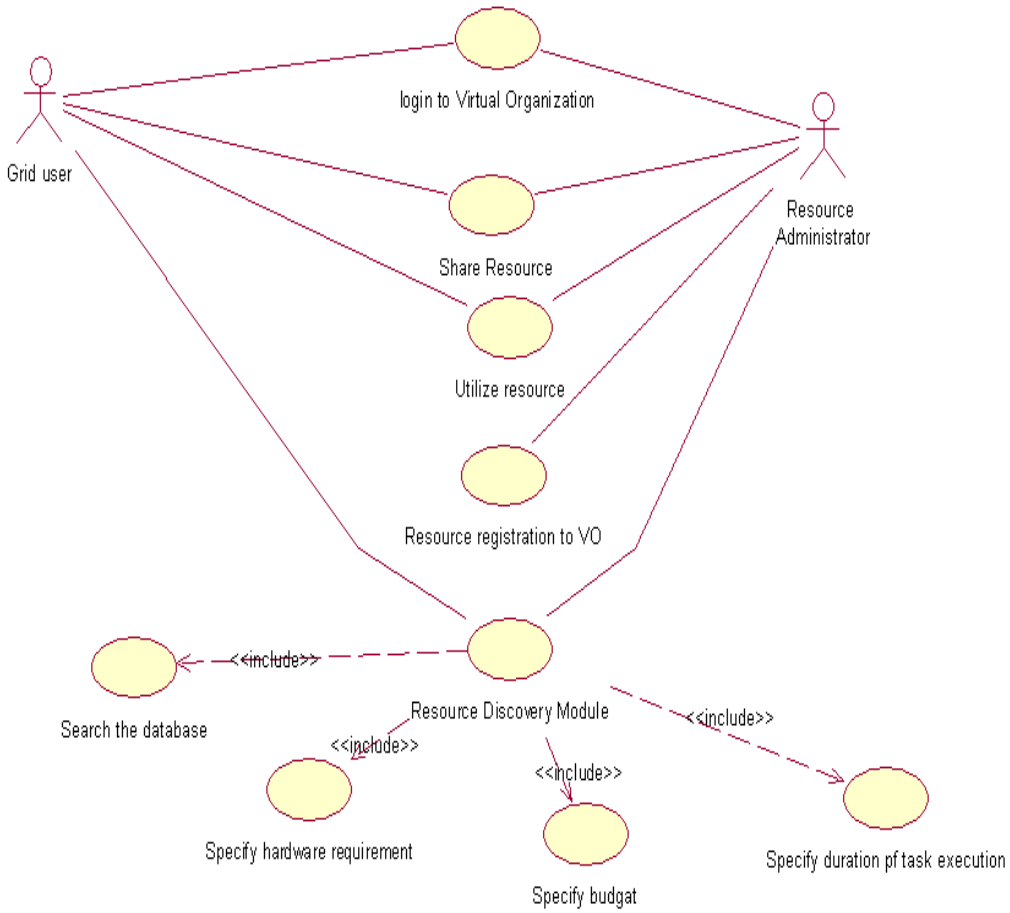

Figure 3.1: Use case Diagram of Resource Discovery

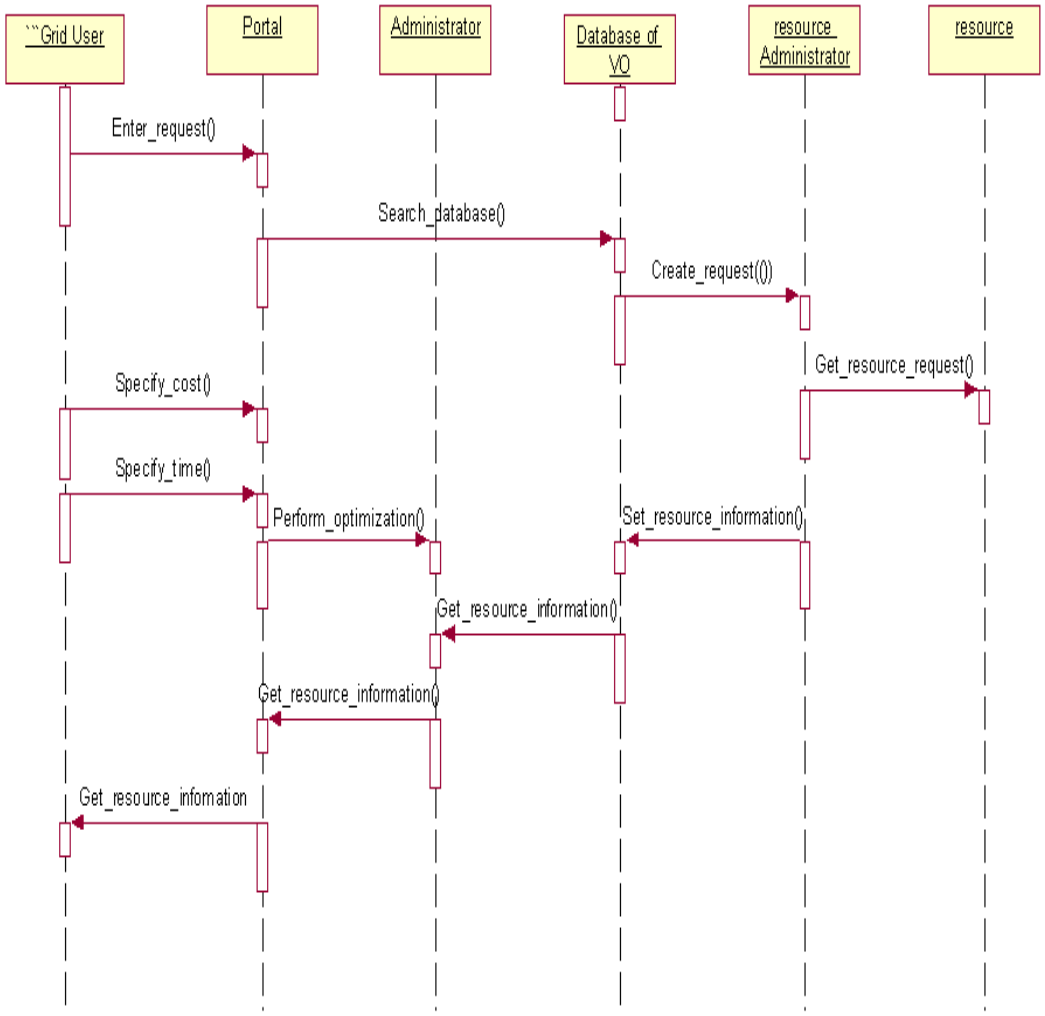

Figure 3.2: Sequence Diagram of Resource Discovery

3.2.2 Job Submission Service

The Job Submission service enables users to submit jobs to remote job schedulers and monitor the status of jobs remotely. It supports the ability to submit jobs to multiple types of job resources. The Job Submission service will basically present job profile when a user clicks the New Job button to submit a job. If there is more than one job available, then the Job Submission portlet will present a list box to allow users to select the appropriate job profile with which to edit jobs. Portal administrators can easily override this profile, or add new profiles, depending on the job profiles that are available in the web applications that have been deployed to the web application server. The usecase

diagram of Job Submission has shown in Figure 3.3. Sequence Diagram of job submission is shown in Figure 3.4.

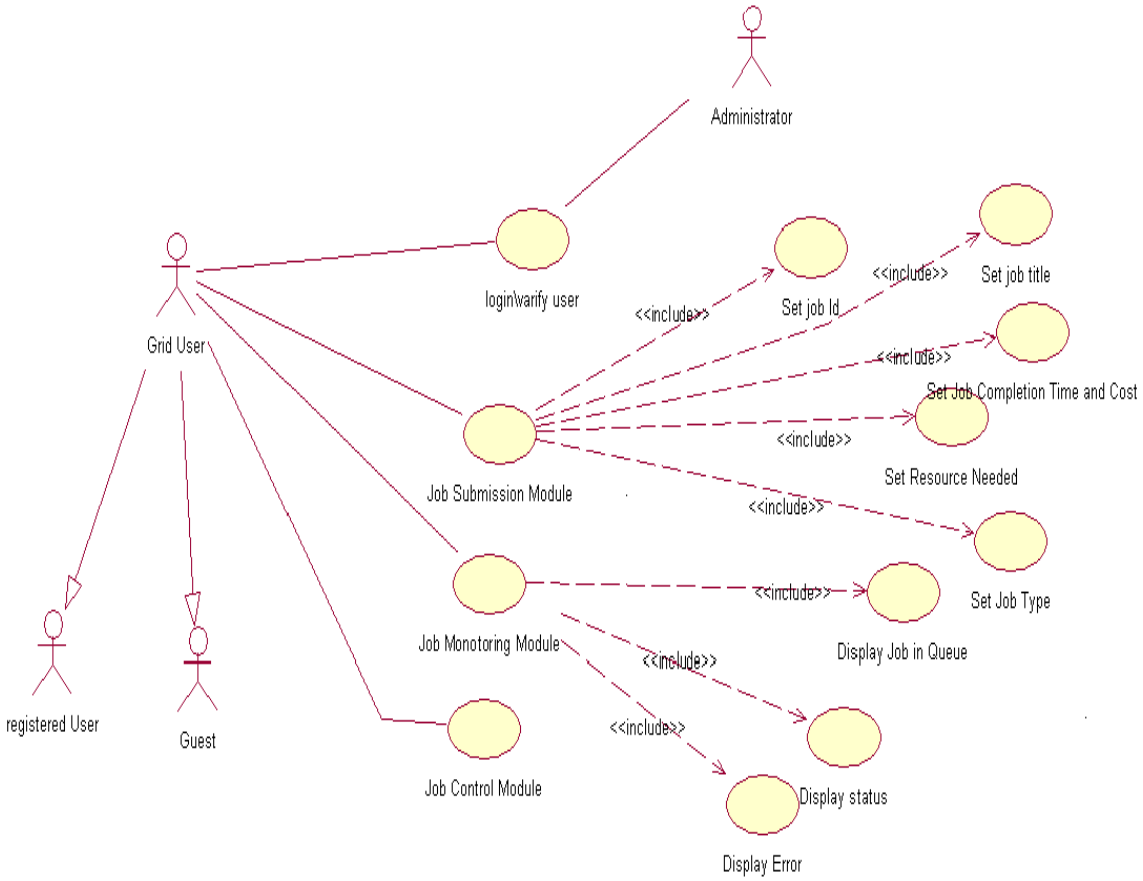

Figure 3.3: Use case Diagram for Job Submission

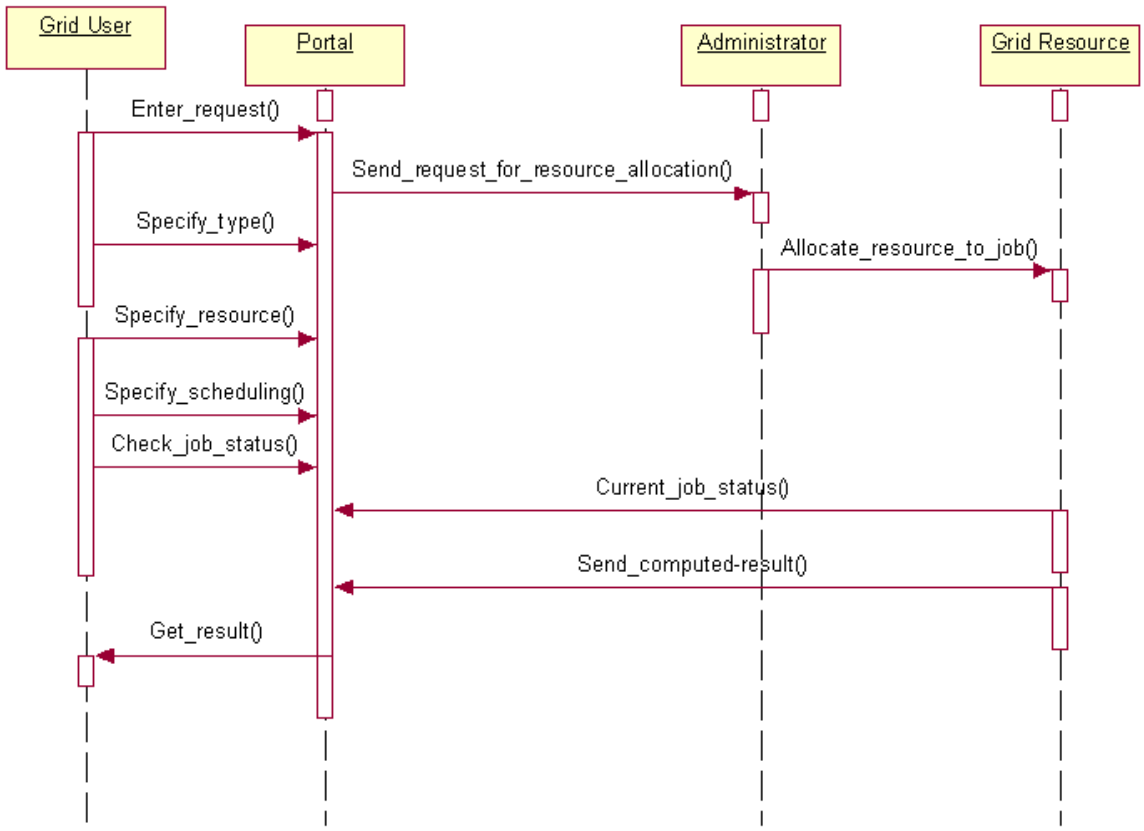

Figure 3.4: Sequence Diagram of Job Submission

3.3 Conclusion

This chapter presented a design overview of a Grid portal. Usecase diagram and Sequence diagram have been created for Resource Discovery portlet and Job Submission portlet.

Next chapter will focus on the installation on GridSphere and Creation and deployment of Resource Discovery portlet and Job Submission Portlet.

Chapter 4

Grid Portal Implementation

In this thesis work GridSphere, which is 100%, JSR-168 compliant and an open-source Portal framework has been chosen for implementation. The GridSphere Portal framework provides an open-source portlet based web Portal development environment. It provides the ability to plug-in custom user interfaces for each of its main page. GridSphere has many portlets in build such as login/logout, date, locale selection, layout management portlets etc. But the main feature of GridSphere is that it enables developers to quickly develop and package third party portlet web applications that can be run and administered within the GridSphere portlet container. In this work resource discovery and job submission portlets have been designed.

This chapter presents setup of Grid portal and implementation of portlets . Portlet API is an extension to the servlet API specification.

4.1 GridSphere Installation

The essential tools for installing and running GridSphere are the availability of Jakarta Ant and Apache Tomcat. GridSphere's source code is build into Tomcat by using Ant, an open-source compiler, a command-line tool, is used to run a build script, which is defined in an XML file. Ant informs the compiler what source code needs to be compiled and what Java doc documentation files need to be created. Figure 5.1 and Figure 5.2 show the deployment of GridSphere into Tomcat container. After the installation completes, "BUILD SUCCESFUL", is seen on command prompt.

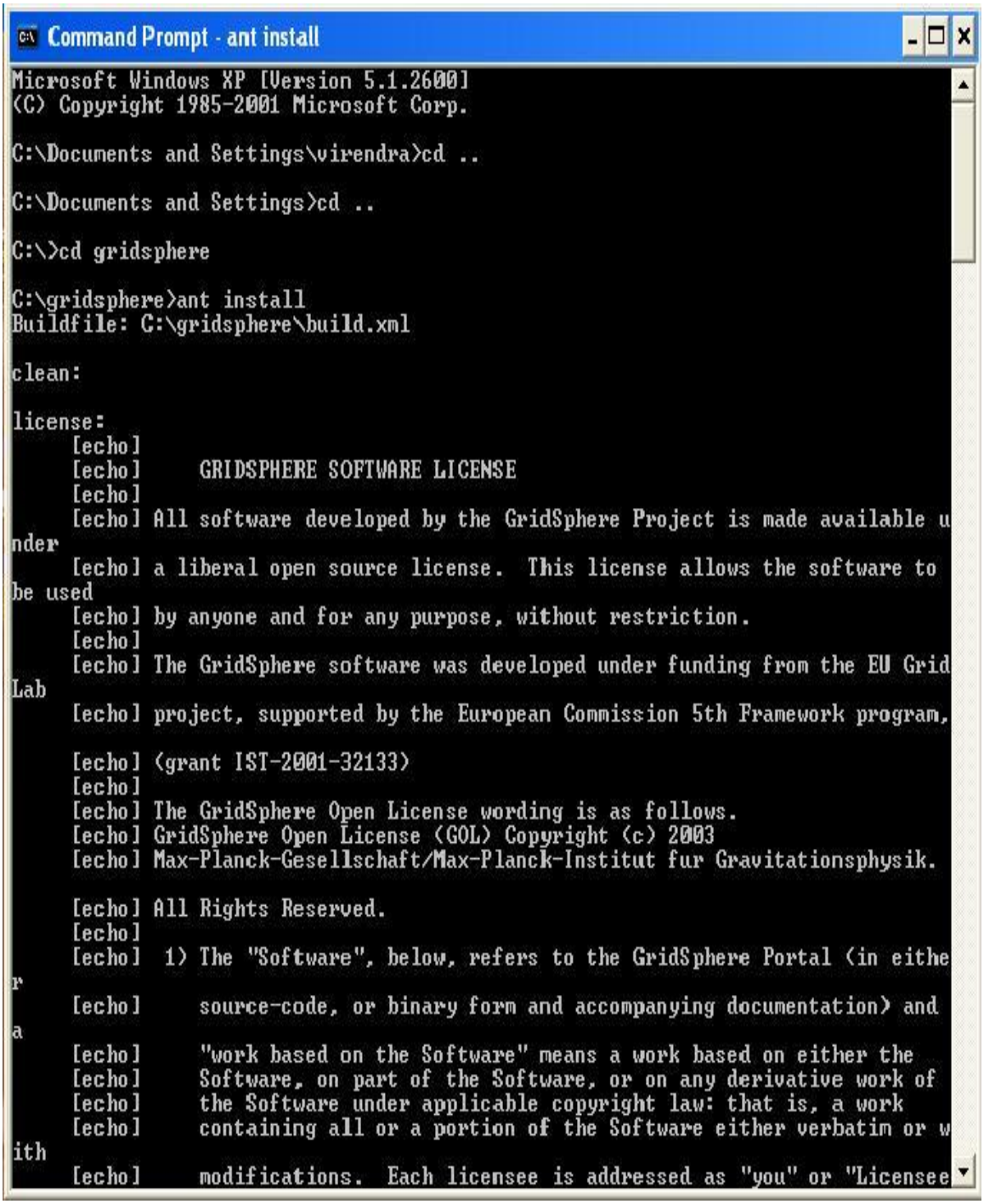

Figure 4.1: GridSphere Installation

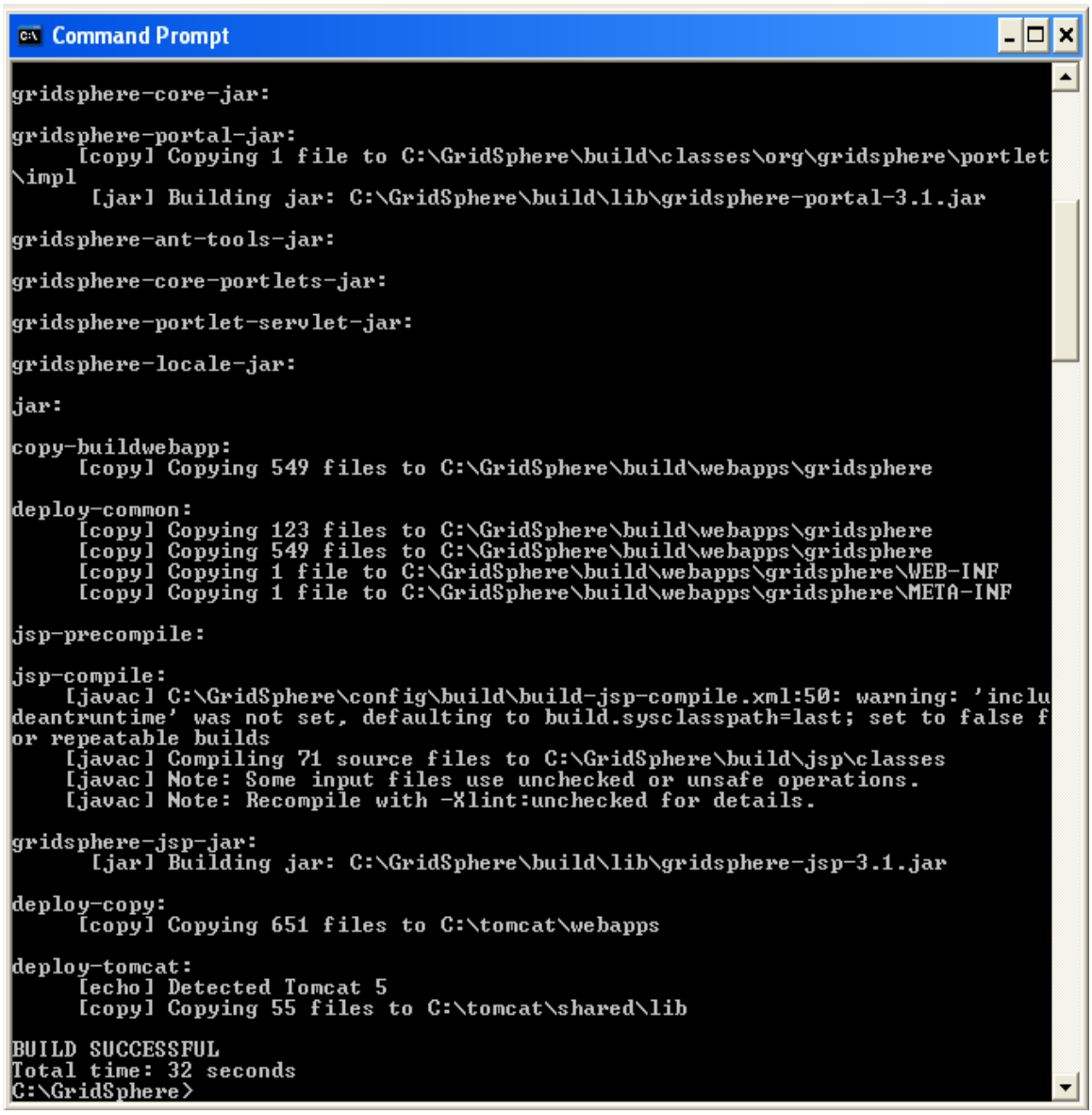

Figure 4.2: GridSphere Installation (Conttd.)

4.1.1 Starting GridSphere

After installation and successful build GridSphere is ready to be accessed at http://lacalhost:8080/GridSphere/GridSphere. To access GridSphere on Tomcat server, restart tomcat server. GridSphere starting page shown in Figure 5.3 is the first starting page. It is a portlet administrator creation form which requires information about the user with role administrator.

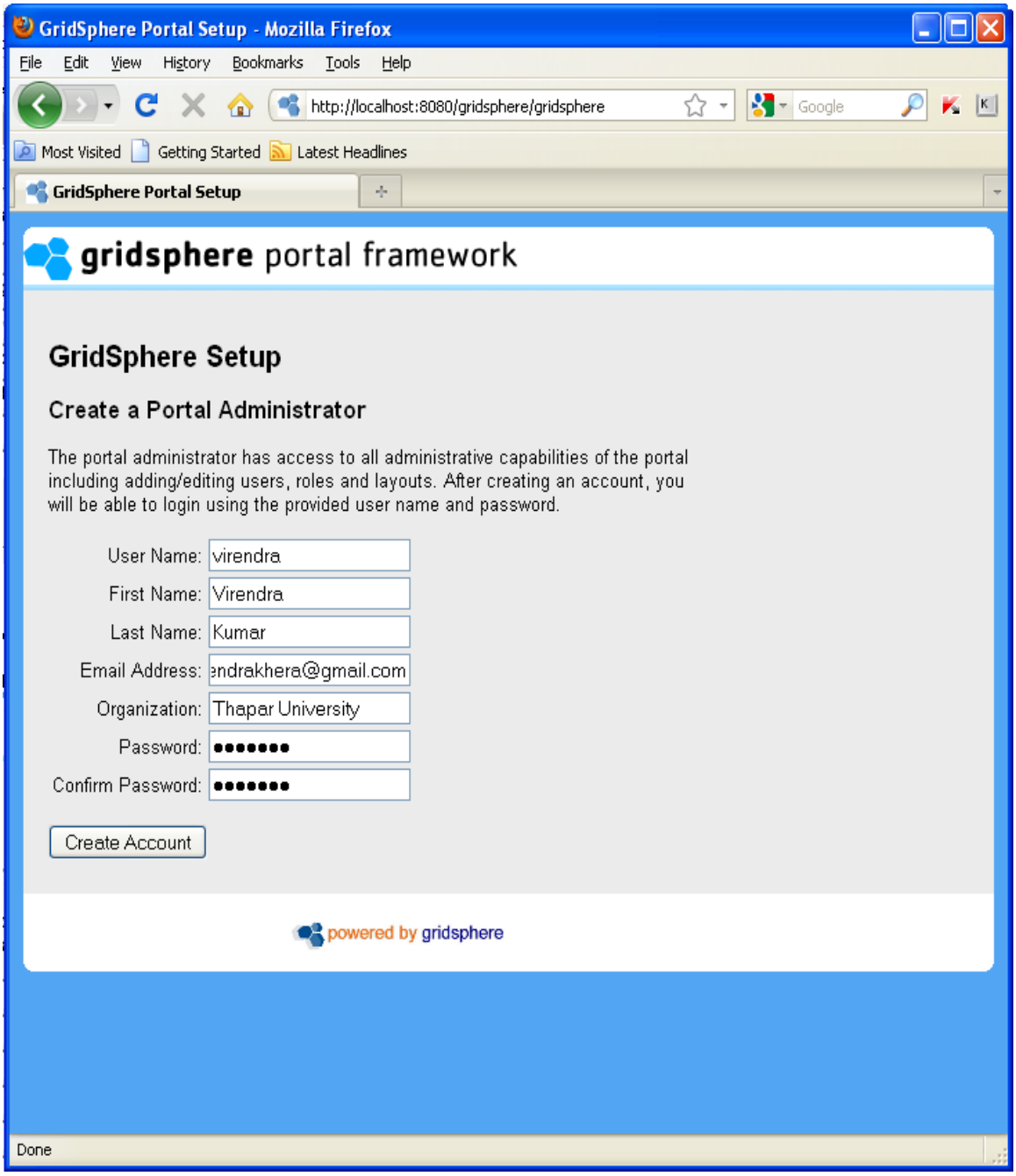

Figure 4.3: GridSphere Starting page

After creating account on GridSphere user has to login on Portal to access its in-build portlet ant deploy new portlet according to their need. Figure 5.4 show the login page of GridSphere Portal.

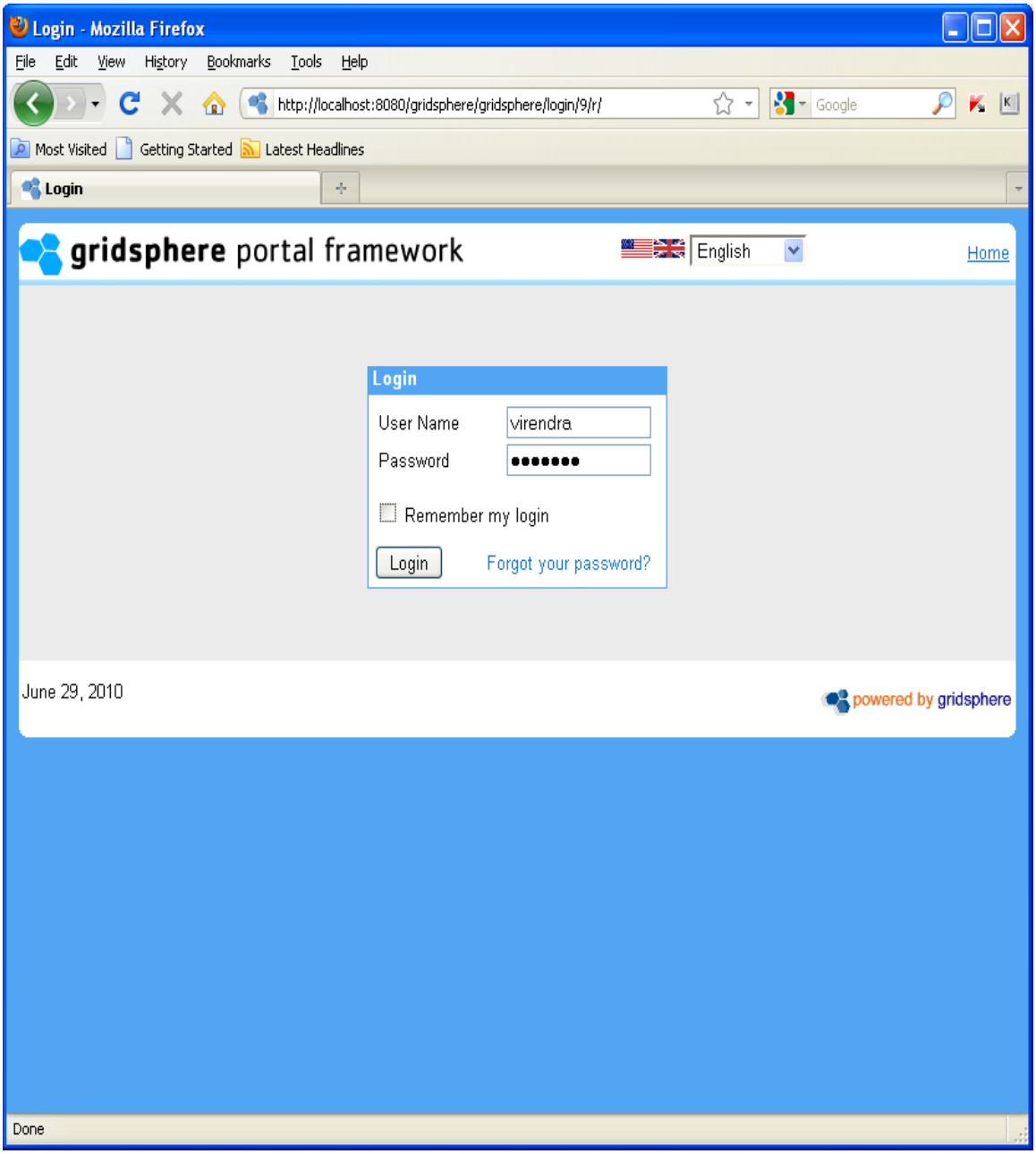

Figure 4.4: GridSphere Login page

4.2 Portlet Creation

Portlets are the pluggable user-interface components to a Grid Portal. Thesis work requires the design and implementation of the various Grid portlets into the GridSphere portlet toolkit. It is a small application that is plugged into or assembled in a Portal container to provide a specific service application into the Portal interface. The necessary

concept to understand is that, a portlet is combination of three different parts: portlet logic, portlet service and the visualization component.

Portlet logic is java source code that is responsible for the setting of variables and the service provided by the portlet. This logic normally provides one service, which is design for specific task, but it may provide multiple services.

The visualization components of the portlets are various JSP files. The portlet logic decides which JSP file is to be invoked.

Services are the main aim of Portal for which it is created. Different portlet are created to offer different services. Services are usually deal with the low-level detail.

4.3 Resource Discovery Portlet

The resource discovery portlet integrates all resources that are distributed geographically for end user into a Portal interface. Resource discovery Portal occupies searching of resources that fulfill the user's application requirements. The resources are reserved in advance for future use in a database where user can search resource according to their need. The resource discovery can be described in terms of required characteristics, to a set of resources that meet the expressed requirement. It takes as input a user request and returns a list of resources and services that can fulfill the user requires.

Resource Discovery portlet is first created and then deployed to the GridSphere portlet container. To create a new portlet following commands are recommended to run.

- Open command prompt and go to the GridSphere_HOME directory
- Run ant new-project command
- Set a project title, asked by the system Ex- Resource Discovery.
- Set project name asked by the system. This will be used for the web application name.

Project name is also used as directory name in the GridSphere/project directory. The directory is automatically created after the successful portlet creation. Ex- Resource

Discovery. Screen shot of creation Resource Discovery portlet is shown in Figure 5.4 and 4.5.

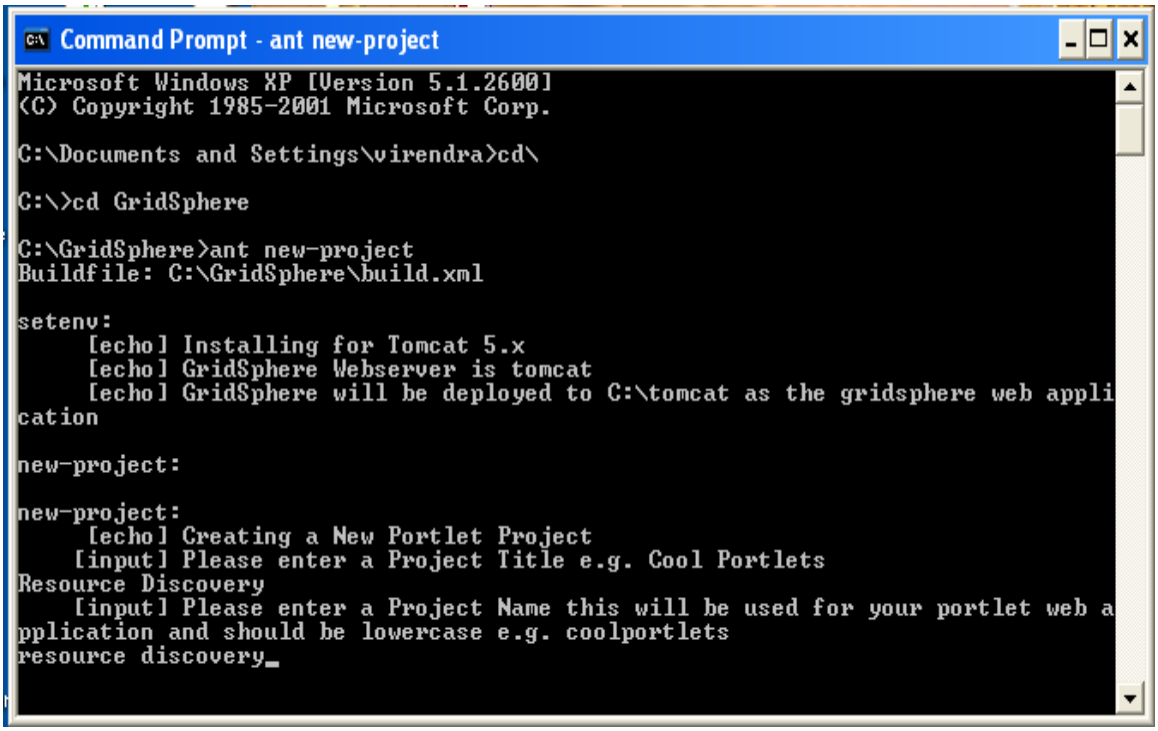

Figure 4.5: Creation of Resource Discovery Portlet into GridSphere

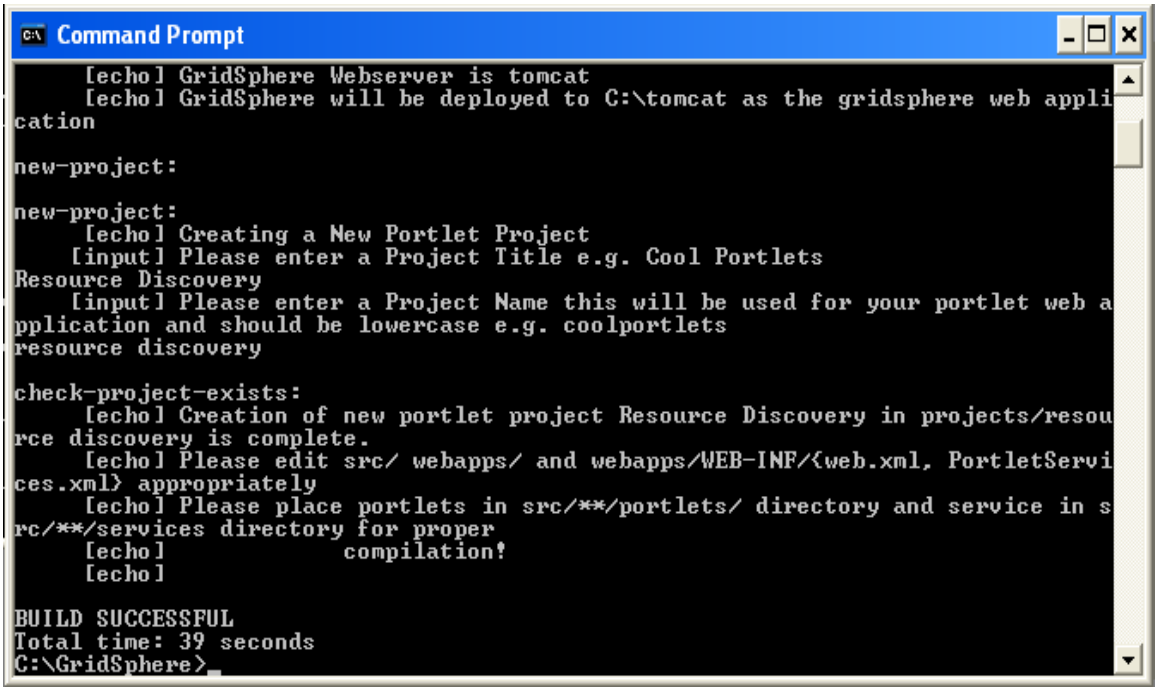

Figure 4.6: Creation of Resource Discovery Portlet into GridSphere (contdd.)

4.3.1 Deployment of Resource Discovery Portlet

To deploy a portlet into a portlet container, it must be packaged in the form of a Grid application archive or GAR file. Build.xml file contains all the recommended information about the portlet that are necessary to deploy a portlet into a Grid Portal. Figure 4.6 and 4.7 are showing snapshot of Resource Discovery portlet deployment into GridSphere. To deploy a new portlet into the GridSphere the recommended command to run are given below-

- Open command prompt and go to the "GridSphere/project/job submission" directory.
- Run 'ant install' at command prompt to deploy the portlet to GridSphere.

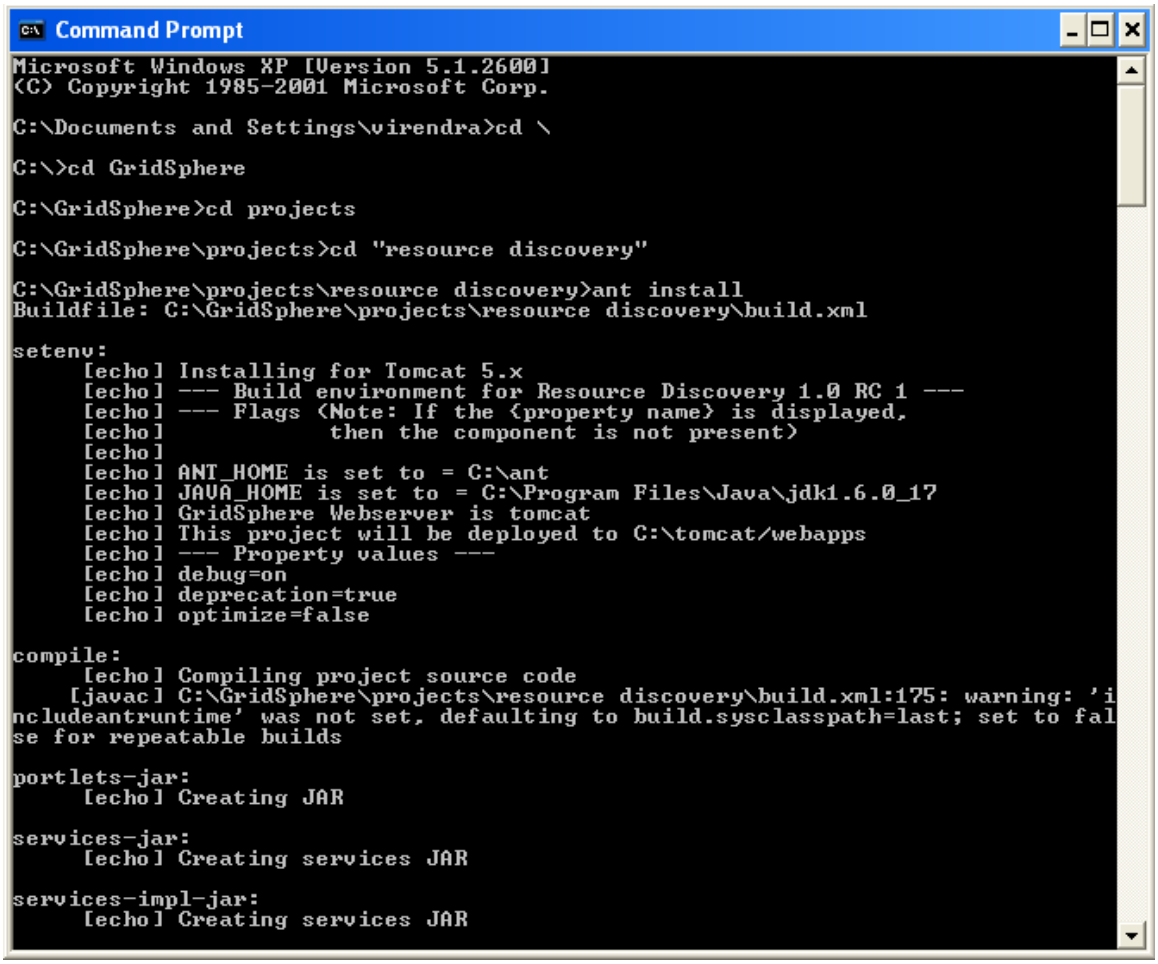

Figure 4.7: Deployment of Resource Discovery Portlet into GridSphere

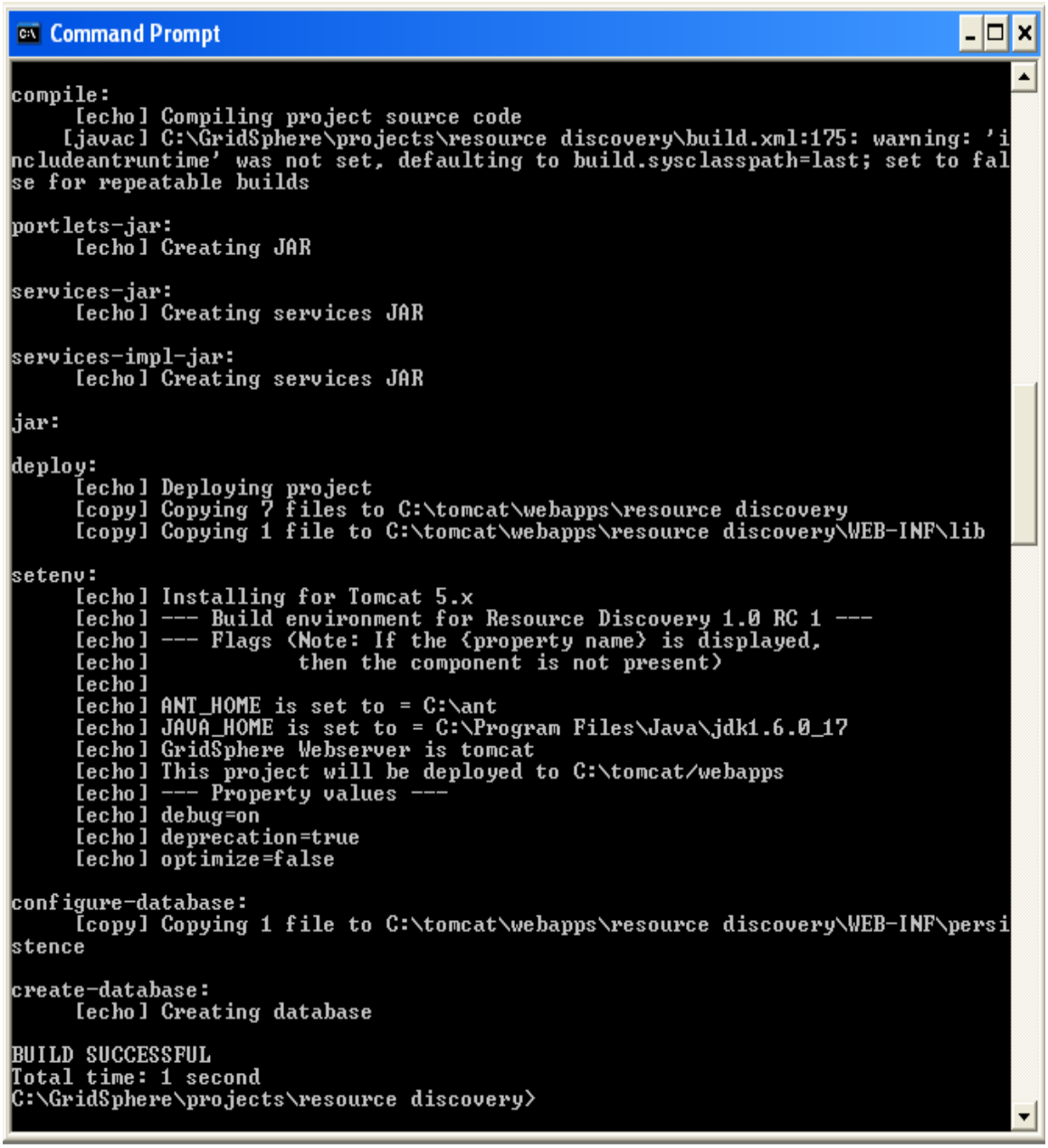

Figure 4.8: Deployment of Resource Discovery Portlet into GridSphere (contdd.)

4.3.2 Configuration Resource Discovery Portlet in Layout Manager

After restarting GridSphere go to the Layout manager. In Layout Manager first select layout customize as loggedin. After that, create a new tab named resource discovery. Now add resource discovery portlet to a named resource discovery tab. Now save resource discovery portlet and go to home tab of GridSphere. Figure 4.8 is showing Layout Manager configuration.

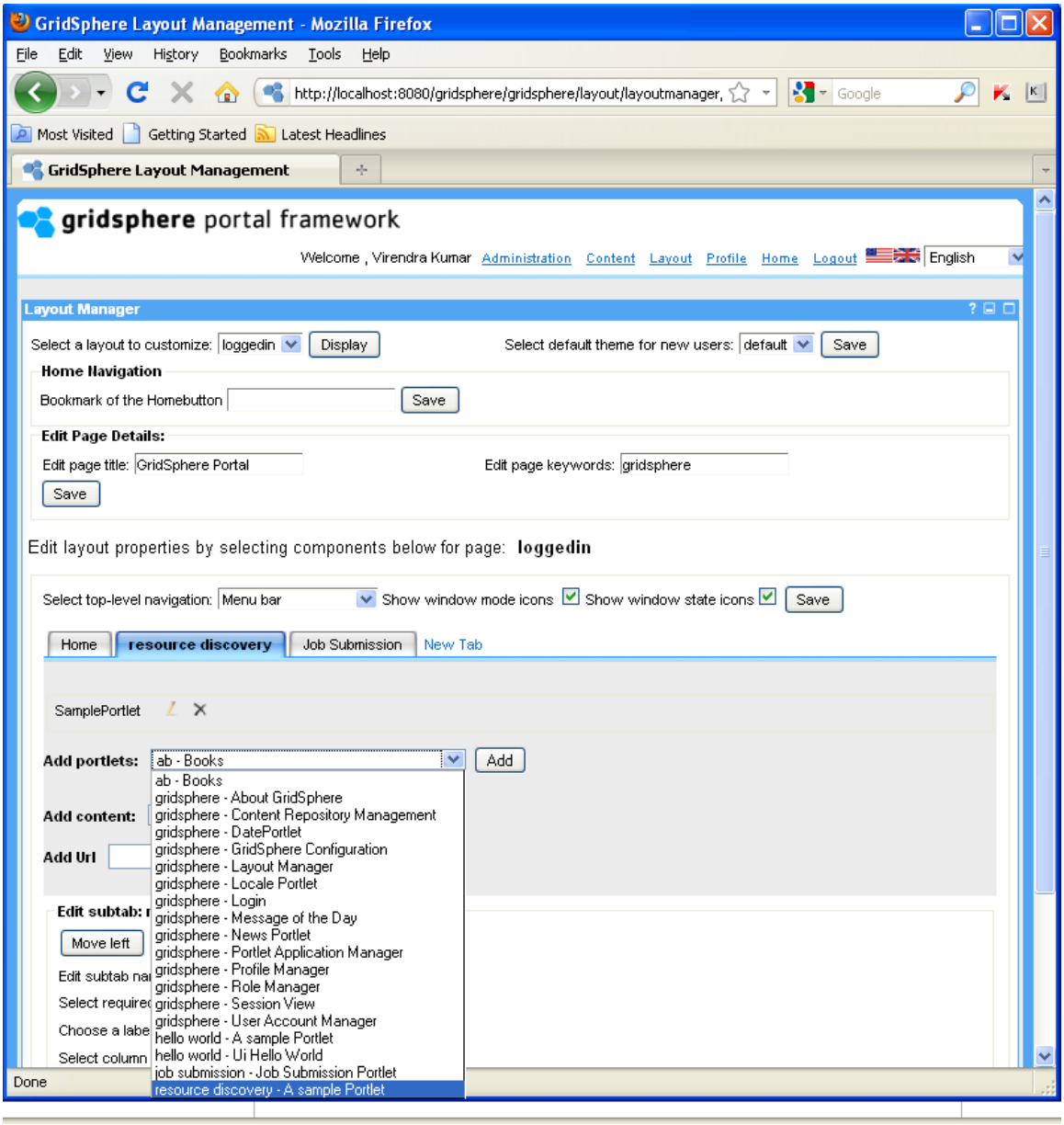

Figure 4.9: Set Resource Discovery Portlet into GridSphere Layout Manager

After doing that restart Tomcat and GridSphere to access the Resource Discovery portlet inside the GridSphere portlet container.

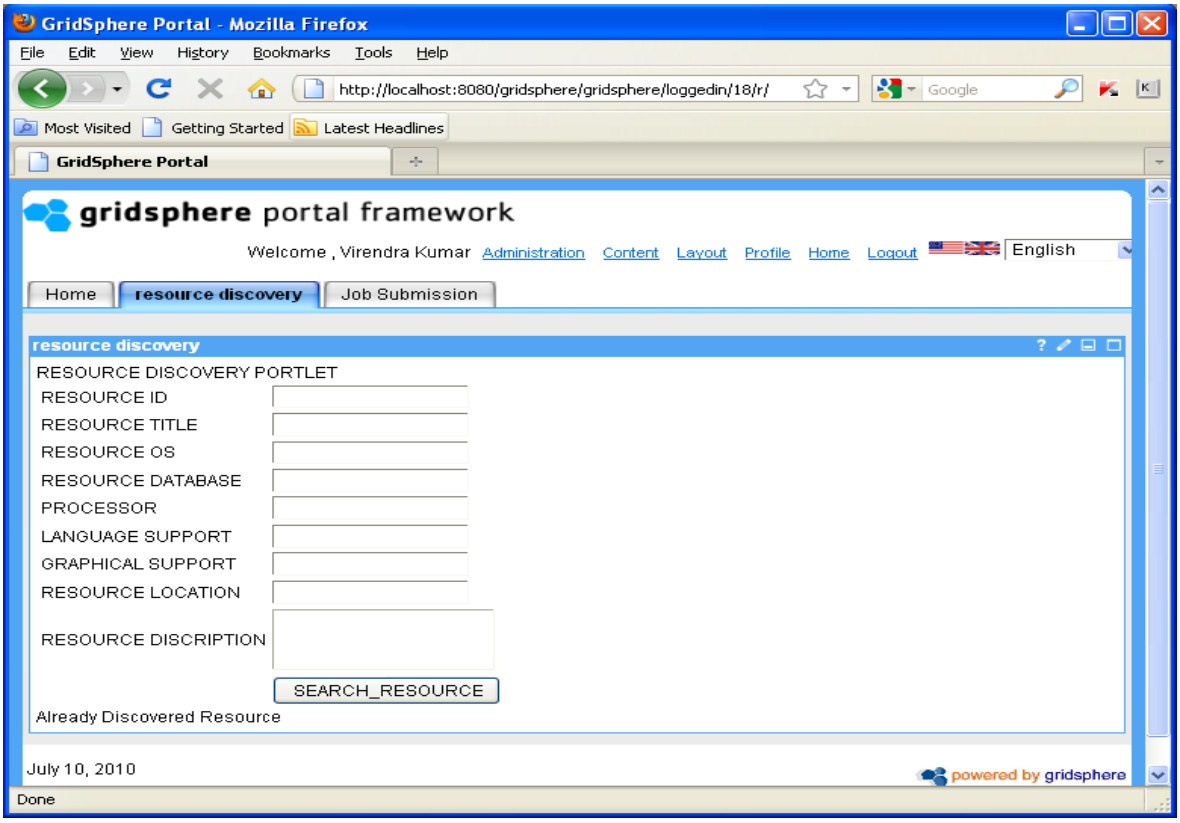

Figure 4.10: Resource Discovery portlet

4.4 Job Submission Portlet

The Job Submission portlet provides a general routing system for submitting, listing, specifying, viewing and managing jobs. A job submission portlet hides the complexity of the Grid environment and Grid protocols from the user. End user can submit the job and transfers the output files to the user upon job completion. To submit a job into Grid environment for computation user needs to provide some basic information about resource needed, time of computation, cost, type of job, specific scheduling and a brief description of the job.

Next step is creating a new portlet in the GridSphere for job submission. To create a new portlet same commands are recommended to run as described in section 4.3.2.

All template files are now created in the GridSphere\projects\job submission directory automatically. All the above steps have been demonstrated with the following screen shots to create a new portlet.

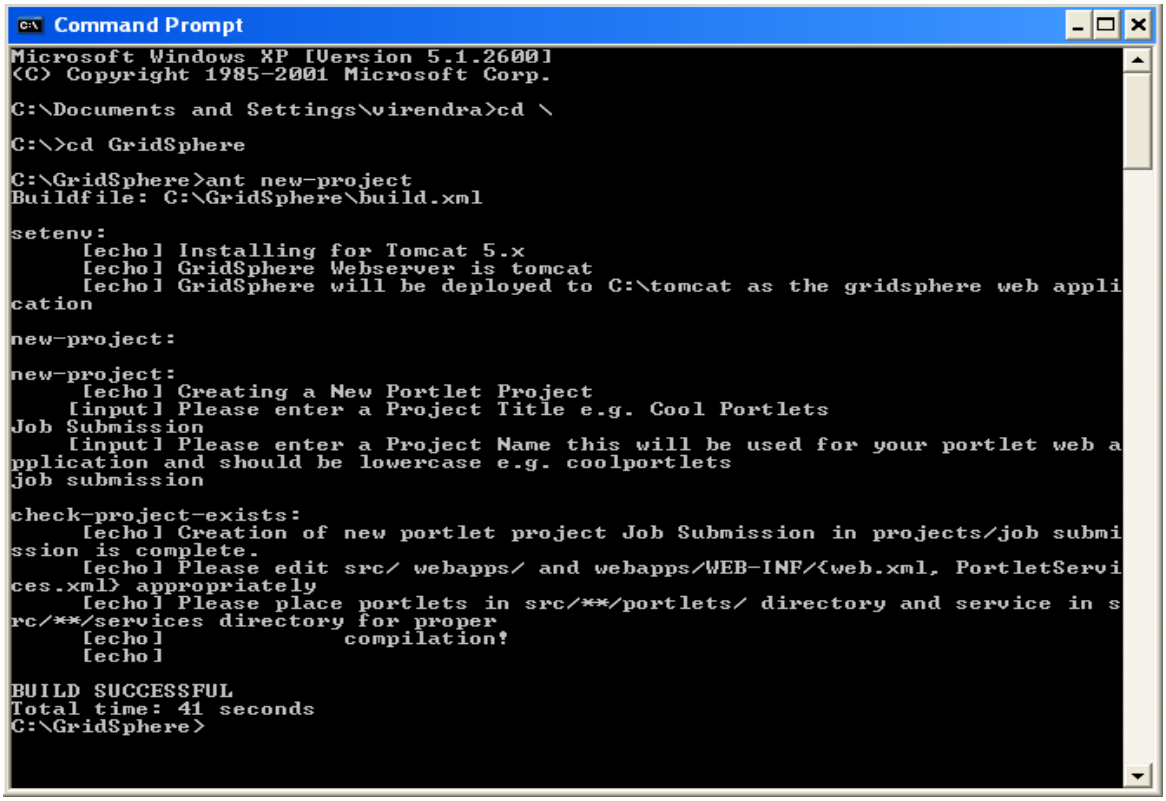

Figure 4.11: Creating Job Submission portlet into GridSphere

After the portlet template has been developed in the GridSphere\project\job submission directory, the next step is to change some descriptor file already generated by GridSphere Portal toolkit and write some source code file and JSP file to build the actual portlet according to the service that the portlet provide to the user. All descriptor files are located in GRIDSPHERE_HOME/projects/hello world/webapp/WEB-INF directory.

Following are the deployment descriptor files need to edit according to portlet.

portlet.xml:- Portlet are JSR-168 standard, describing the portlet. Edit portlet.xml shown below:

- Change portlet name according to your portlet.

  <portlet-name>Job Submission portlet</portlet-name>

  <display-name xml:lang="en">Job Submission</display-name>

- Change portlet-class path to the directory where the source code file is loaded.

<portlet-class>org.GridSphere.gsexamples.portlets.Jobportlet</portlet-class>

- Change title tag in portlet.xml file.

<portlet-info>
<title>Job Submission</title>
</portlet-info>

Creating JSP (Java Server Pages) file: JSP file are stored in webapp\jsp subdirectory of the just created in the job submission directory.

Creating portlet source code: Create a directory org\GridSphere\portlets inside the src directory. The developer java code for the portlet service should be included into this directory. The code created for the following portlet code is Jobportlet.java

4.4.1 Deployment of Job Submission Portlet

To deploy a portlet into a portlet container, it must be packaged in the form of a Grid application archive or GAR file. Build.xml file contains all the recommended information about the portlet that are necessary to deploy a portlet into a Grid Portal. To deploy a new portlet into the GridSphere the recommended command to run are given below-

- Open command prompt and go to the "GridSphere/project/job submission" directory.
- Run 'ant install' at command prompt to deploy the portlet to GridSphere.

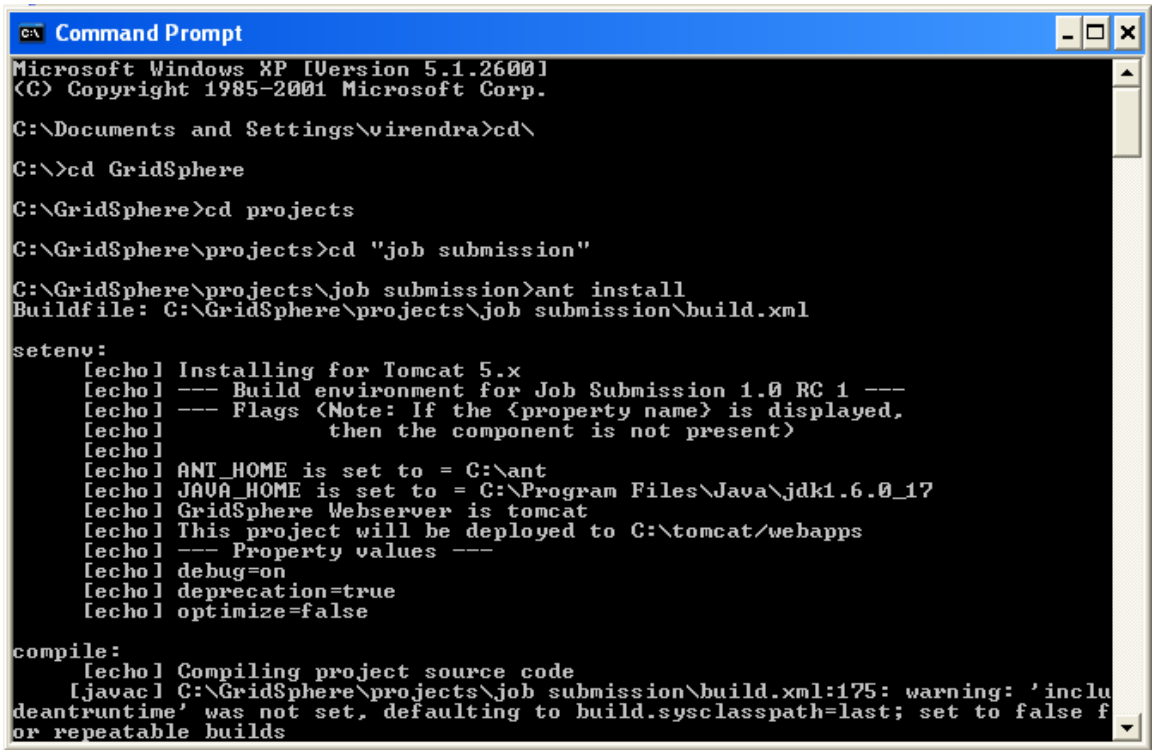

Figure 4.12: Job Submission Portlet Deployment in GridSphere

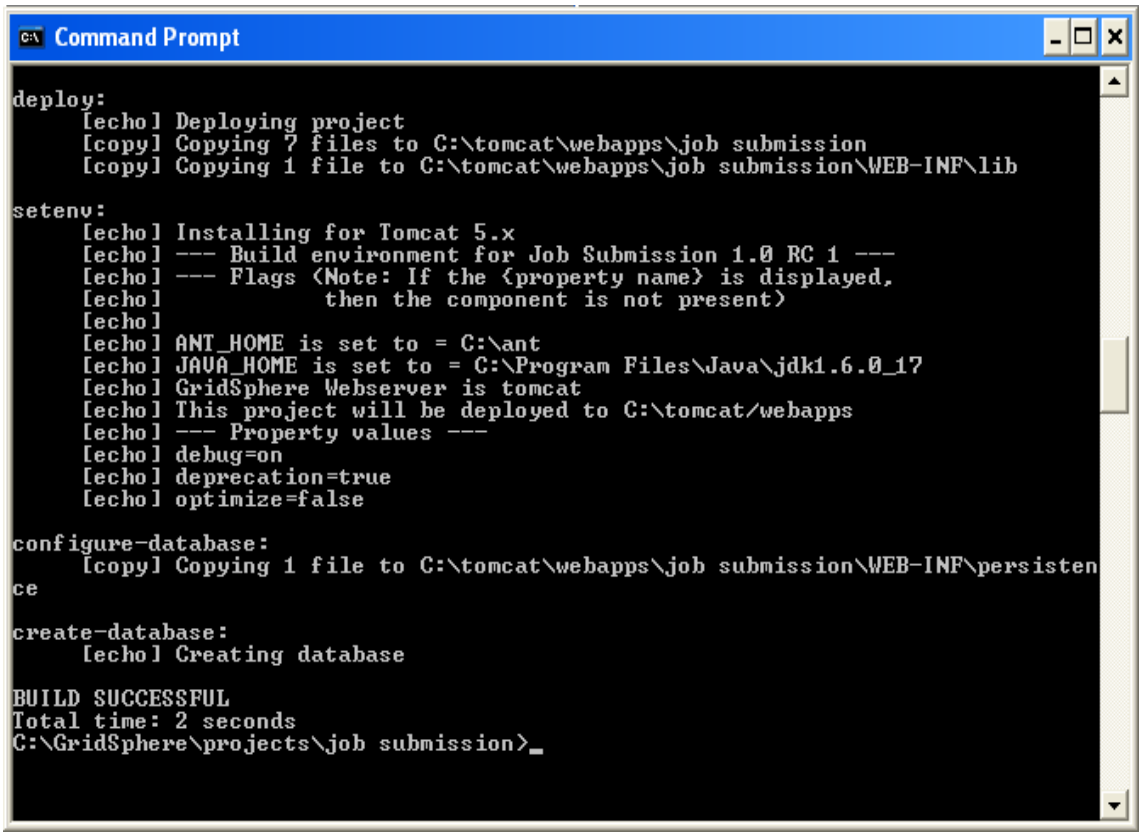

Figure 4.13: Job Submission Portlet Deployment in GridSphere (contdd)

After Deployment of Job Submission portlet, GridSphere requires access to deployed portlet web applications. Since portlets are packaged according to the web application repository (WAR) format defined in the Java 2.3 servlet Specification, the names of the WAR files or web applications needs to be added as an empty file whose filename is defined by the name of portlet web application to C:\.GridSphere directory.

4.4.2 Configuration Job Submission Portlet in Layout Manager

After restarting GridSphere go to the Layout manager. In Layout Manager first select layout customize as logged in. After that, create a new tab named job submission. Now add job submission portlet to a named job submission tab. Now save job submission portlet and go to home tab of GridSphere. Figure 4.12 is showing Layout Manager Configuration.

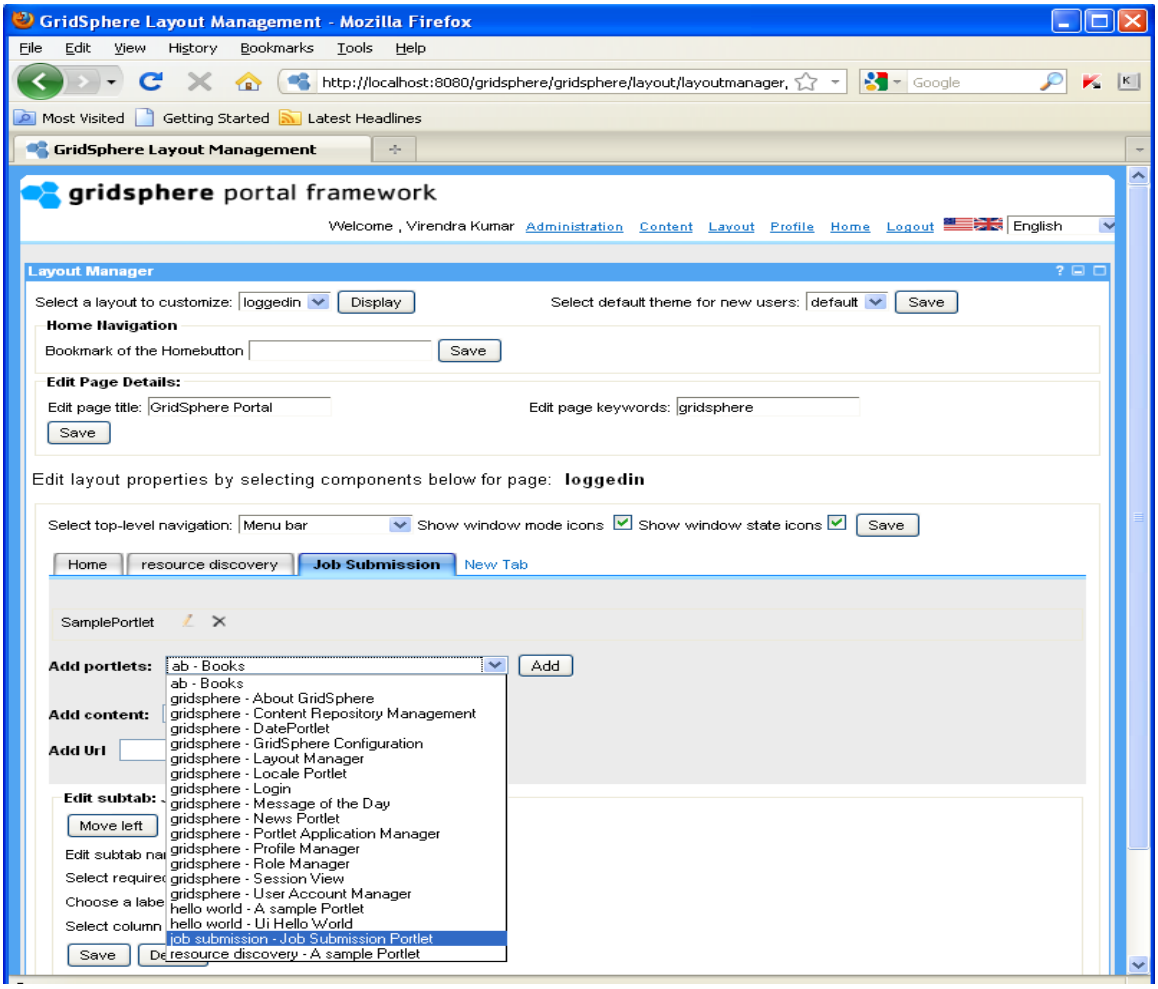

Figure 4.14: Set Job Submission Portlet into GridSphere Layout Manager

After doing that restart Tomcat and GridSphere to access the Job Submission portlet inside the GridSphere portlet container.

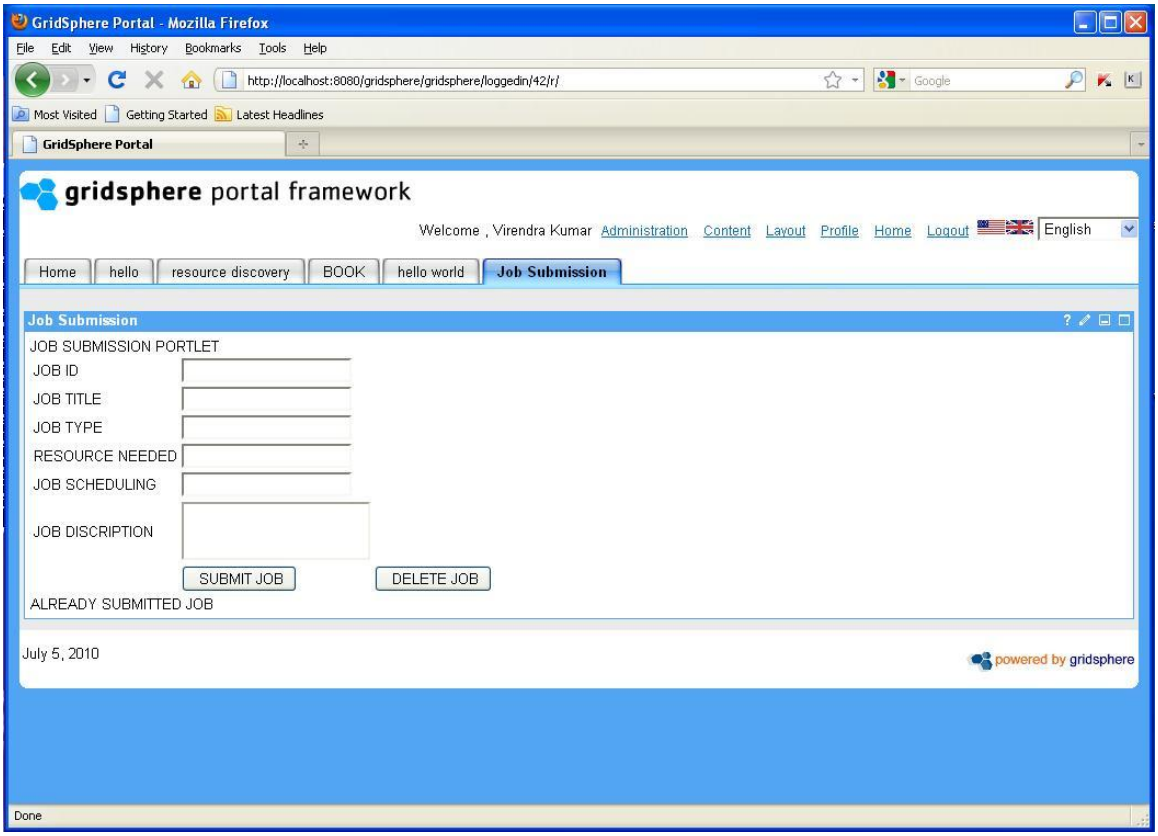

Figure 4.15: Job Submission Portlet

4.5 Conclusion

This chapter presented implementation of GridSphere, creation of Resource Discovery portlet and Job Submission portlet and deployment of Portlets into GridSphere. The Portlets and their execution have also been demonstrated in this chapter. Next chapter will focus on results comes after demonstration.

Chapter 5

Experimental Results

This chapter focuses on experimental results. Successful installation of GridSphere portal has been done on windows platform. Two main portlets Resource discovery Portlet and Job submission portlet have been deployed into GridSphere portal. With the help of this Portal user can submit request for resource discovery. This request goes to Grid middleware for discovering resource in the resource database. User also submits job for computation with the help of Job Submission portlet.

5.1 Case Study

For experimental result Center of Excellence in Grid Computing lab has been considered. Grid test bed has been established in the Computer Science and Engineering Department of Thapar University.

Grid test bed has been established using following middlewares

- GT4
- CONDOR 7.5.1

System configuration

Processor: core 2 duo 3.00 GHz.

RAM: 3 GB.

OS: Windows XP.

The basic structure of Center of Excellence is shown in Figure 5.1.

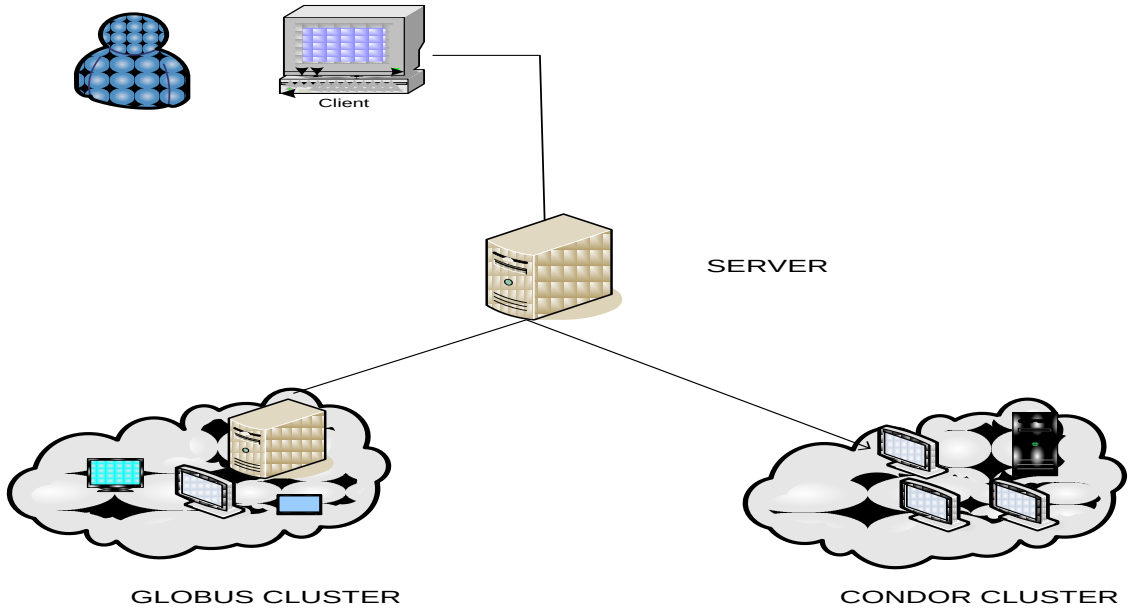

Figure 5.1: Center of Excellence Grid Computing Thapar University

5.2 Experimental Results

- Test case 1

  After the successful installation GridSphere is ready to be accessed at http://lacalhost:8080/gridsphere/gridsphere. Now user can login on the GridSphere and access its various Portlets as illustratted in Figure 4.3.

- Test case 2

  After starting GridSphere, user authentication is required.

  Figure 5.2 has shown the login of a registered user. After login the home page comes on interface which is shown in Figure 5.3.

  If user is not registered on GridSphere then GridSphere gives a message that user does not exist. Figure 5.3 and 5.4 show the login of an unauthorized user and response of GridSphere.

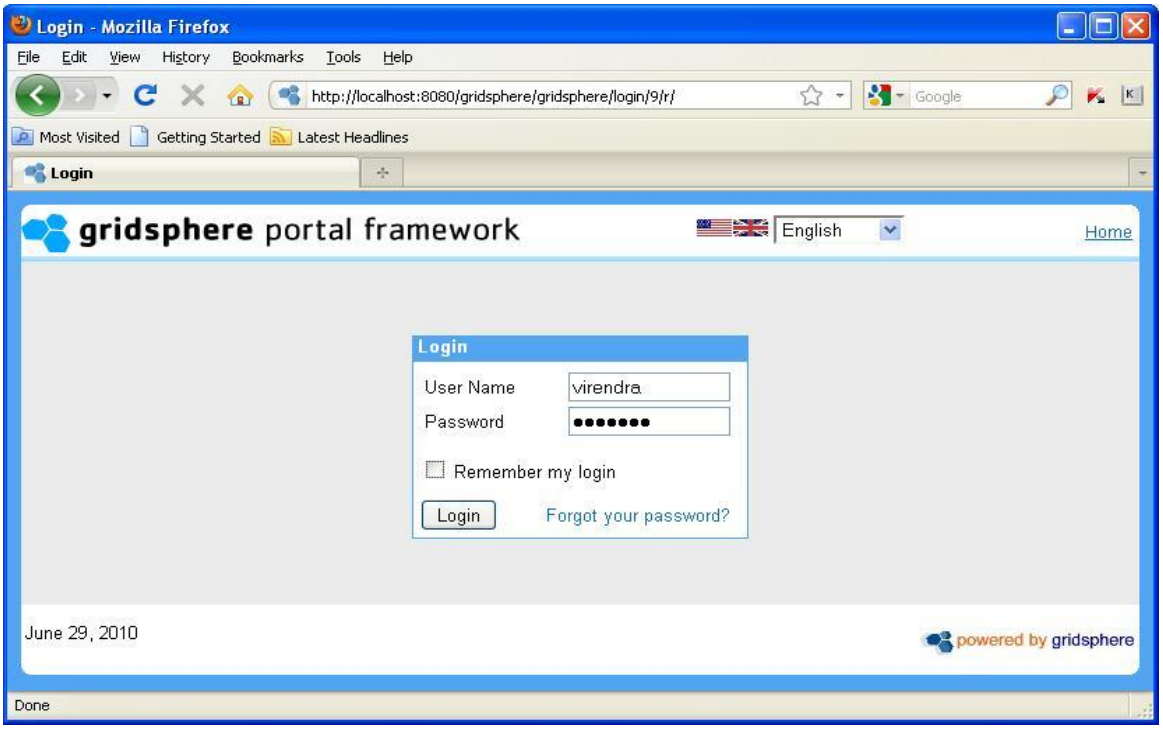

Figure 5.2: GridSphere Login Page

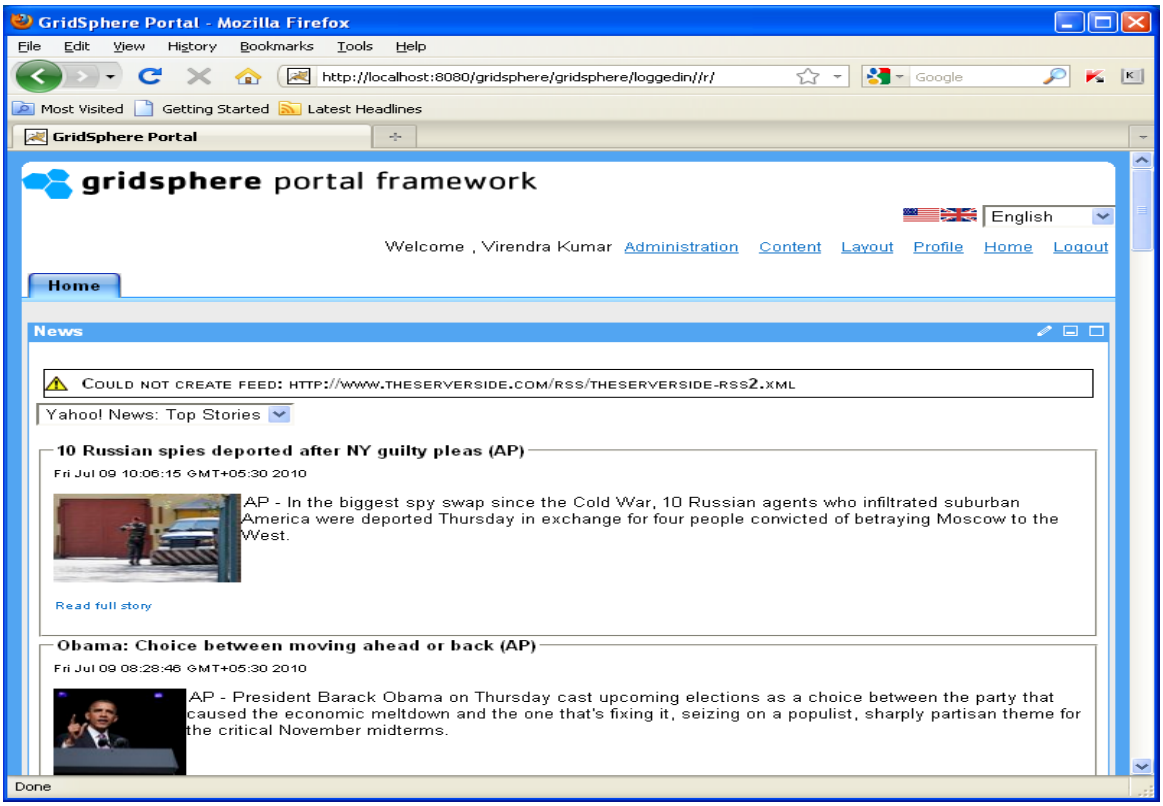

Figure 5.3: GridSphere Home Page

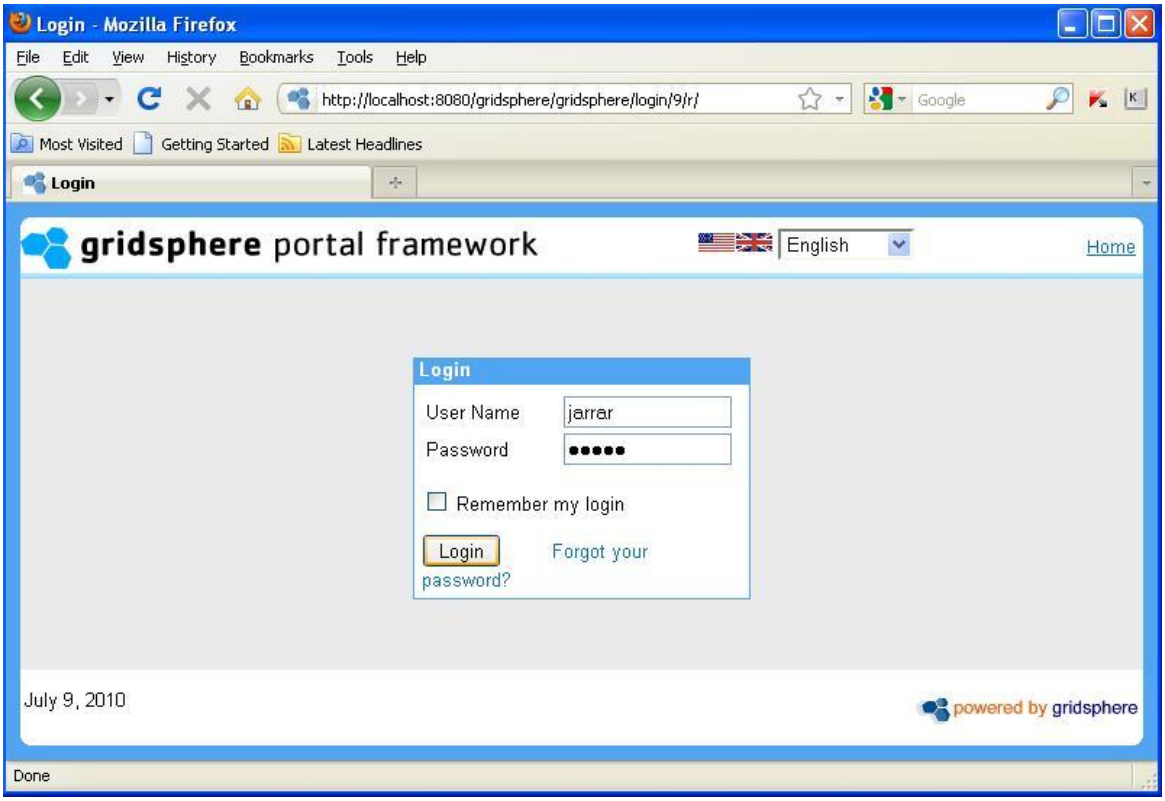

Figure 5.4: Login Without Account

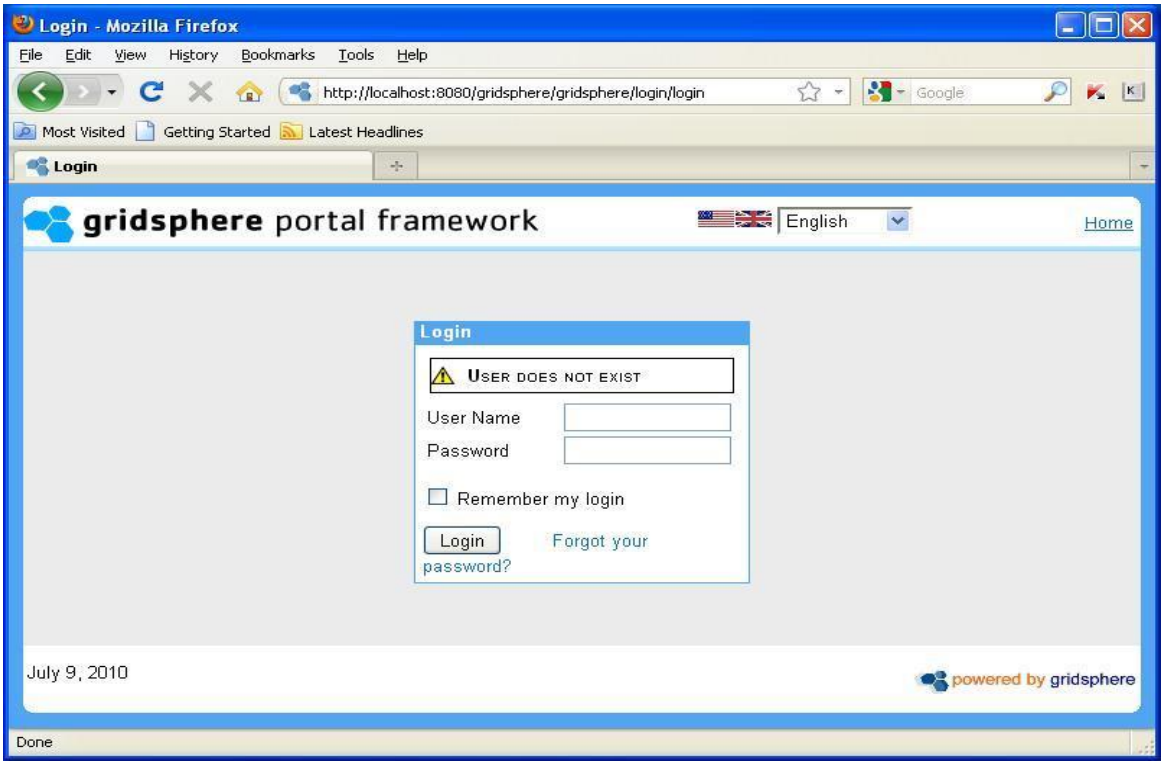

Figure 5.5: User Authentication checking

- Test case 3

  Resource Discovery Portlet has used for resource discovery. Figure 5.6 has shown the user request for a specific resource and Figure 5.7 shown the result of user request.

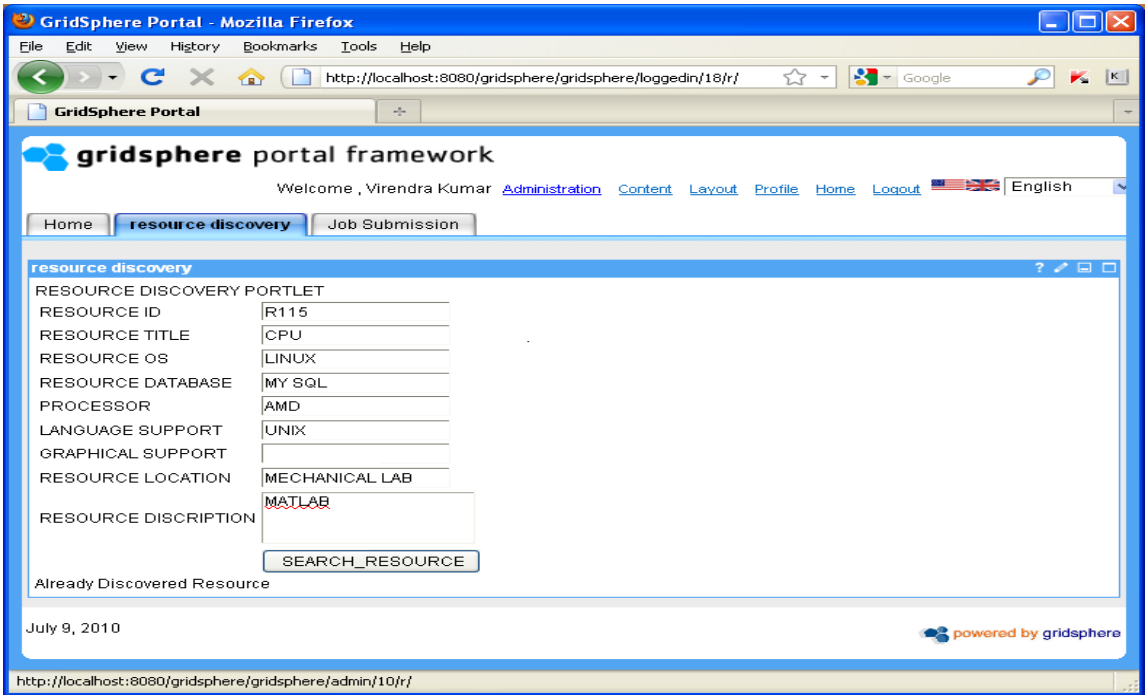

Figure 5.6: Search of Specific Resource

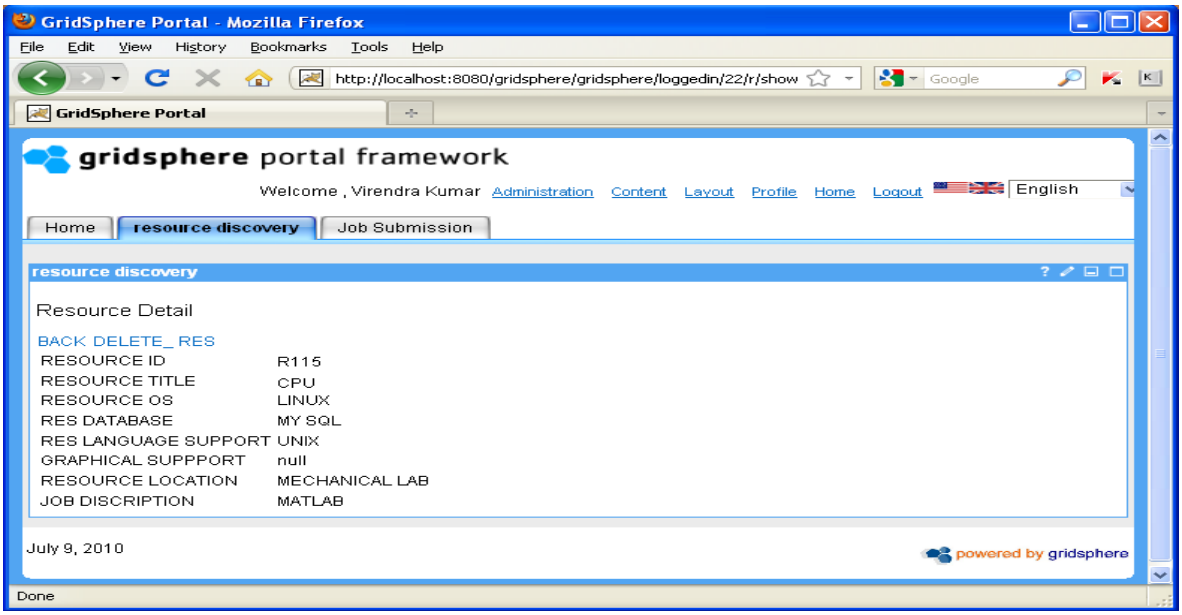

Figure 5.7: Result of Resource Discovery

- Test case 4

  Users can submit their job with help of Job Submission portlet. Figure 5.8 shown user request for job submission and Figure 5.9 shown the request result.

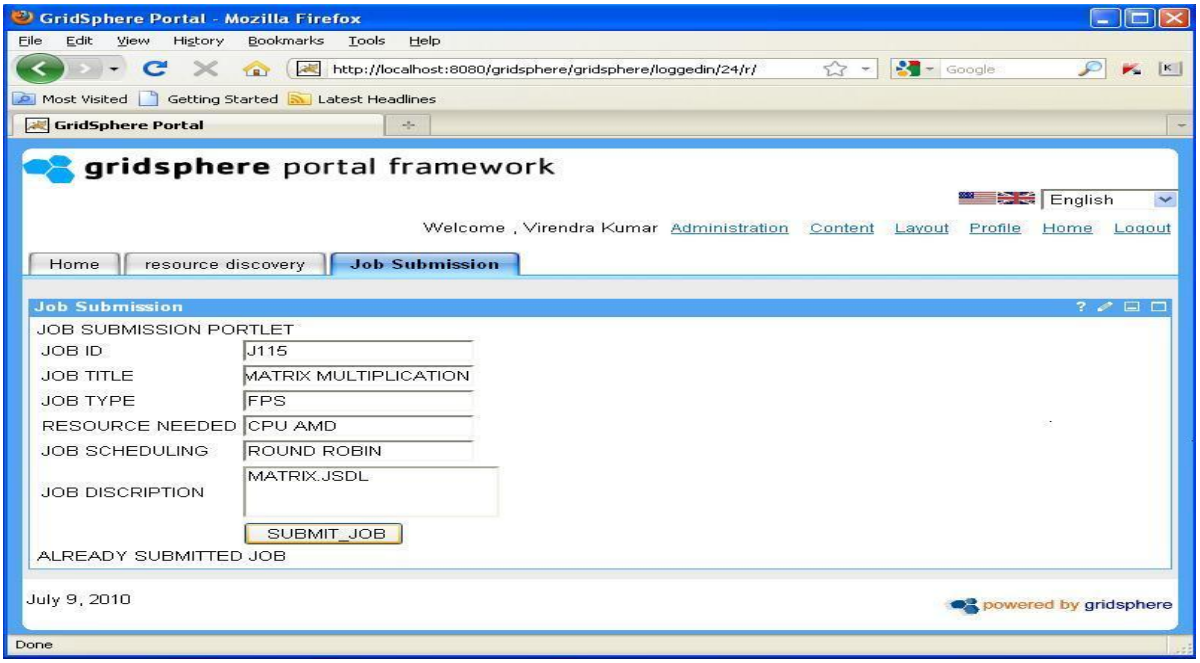

Figure 5.8: Submission of Job

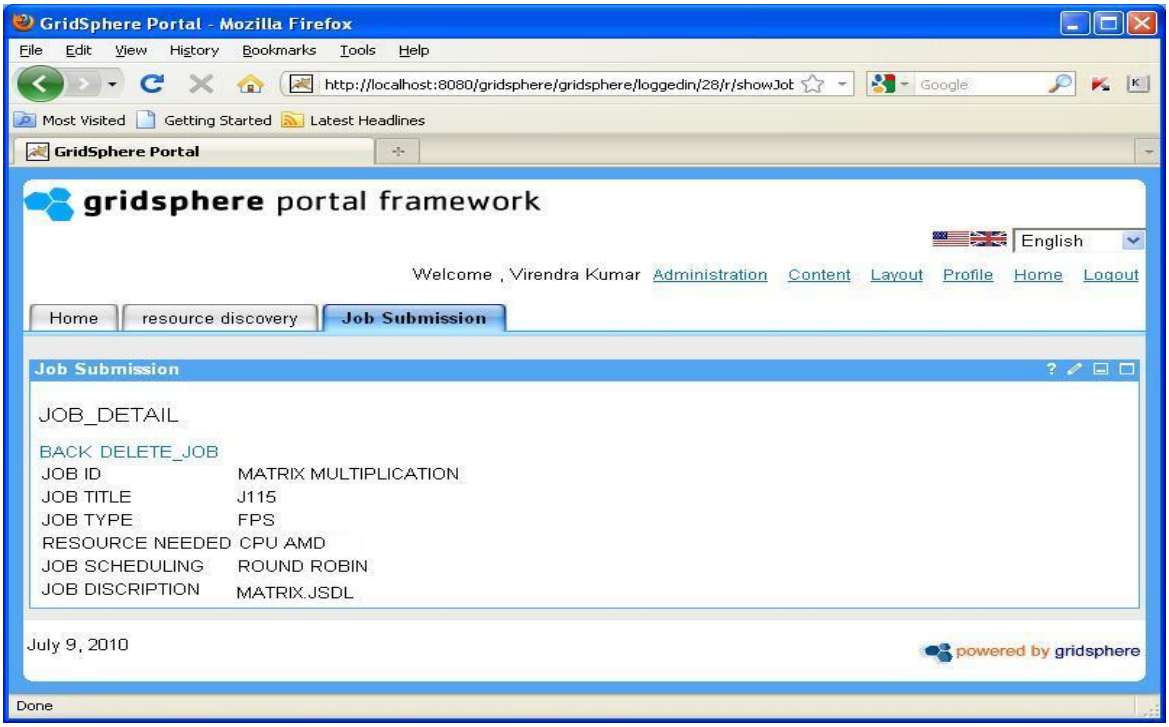

Figure 5.9: Detail of Submit Job

5.3 Conclusion

This chapter presented experimental results of Grid Portal. Test cases have been discussed for checking the authentication of user, resource discovery and Job Submission. Next Chapter will present the overall conclusion of the thesis and future scope in the field of Grid Portal.

Chapter 6

Conclusions and Future Scope

This thesis provides an introduction to the Grid computing. The basic fields such as Grid services, general principles of grid construction, grid architecture, VO, grid middleware and Grid information service of grid computing have been discussed. The main focus of the thesis is on the design and development aspects of a Grid Portal.

6.1 Conclusion

Surveys of variuos first and second generation grid portals have been done in this thesis. Table 2.1 has given comparisons of first generation grid portals and Table 2.2 has given comparisons of second generation grid portals. This thesis explores the usage of Grid Portlets and user interface components to Grid applications. Resource Discovery and Job submission Portlet have been developed and deployed into GridSphere.

Requirements of Grid Portlets have been analyzed through UML diagram. Installation of Gridsphere portal toolkit has been done on windows platform.

Thesis Contributions

- Literature survey of presently existing Grid Portals.
- Comparison of first and second generation Grid Portals.
- Design of grid portal.
- Implementation of Portlets in GridSphere.
- Demonstration of the implementation and deployment of the Portal and Portlets through experimental results.

6.2 Future Scope

The possible future extensions that can be more important are given below:

- In order to enhance the functionality of the portals addition of more new portlet can be done in future.
- In future the performance, availability and fault tolerance can be measured and monitored.
- Real-Time statistic can be introduced in Resource Discovery portlet and Job submission portlet.
- In future the Portal can be enhanced for Agent based applications.

References

[1]. M. Li and M. Baker," The Grid: Core Technologies", Wiley and Sons ltd, ISBN: 0-470-09417-6, published, April 2005.

[2]. M. Geldof "The Semantic Grid: will Semantic Web and Grid go hand in hand?" June, 2004.

[3]. D. D. Vecchio, V. Hazlewood and M. Humphrey "Grid networks and Portals - Eva Grid Portal security" Proceedings of the ACM/IEEE SC2006 Conference on High Performance Networking and Computing, November 11-17, 2006, Tampa, FL, USA 2006

[4]. I. foster "What is the Grid? A Three Point Checklist " GRIDToday, July 20, 2002.

[5]. X. Yang, M. T. Dove et al "Survey of Major Tools and Technologies for Grid-enabled Portal Development" Proceedings of the UK e-Science All Hands Meeting 2006, NeSC 2006.

[6]. J. Denemark, M. Jankowski, L Matyska, N, Meyer, M. Ruda and P. Wolniewicz "User Management for Virtual Organizations" CoreGRID Technical Report Number TR-0012, November 30, 2005

[7]. D. Gannon and R. Bramley" Middleware Technology to Support Science Portals: a Gateway to the Grid", march 2003.

[8]. K. Nadiminti and R. Buyya "Enterprise Grid computing: State-of-the-Art" Oct 2005.

[9]. L. Ferreira et al, "Introduction to Grid Computing With Globus" An IBM Redbooks publication Publish Date 01 October 2003

[10]. M. Baker, R. Buyya and D. Laforenza "Grids and Grid technologies for wide-area distributed computing", Software-Practice & Experience 2002.

[11]. I. Foster, C. Kesselman, and S. Tuecke "The Anatomy of the Grid." International Journal of Supercomputer Applications, 15(3), 200–222 , 2001.

[12]. V. Silva “Grid Computing for Developers “, Charles River Media, Inc., Hingham, Massachusetts March *16,* 2006**.**

[13]. J. Novotny, M. Russell and O. Wehrens “GridSphere: An Advanced Portal Framework”, August 31-2 September 2003.

[14]. M. Russell, J. Novotny, and O. Wehrens, (2006): “Gridsphere's Grid portlets.” Gridsphere Project. http://www.Gridsphere.org/. 2006.

[15]. F. Juan, G. Fox and M. Pierce “Grid Portal System Based on GPIR” Proceedings of the Second International Conference on Semantics, Knowledge, and Grid, Year of Publication: 2006 ISBN:0-7695-2673-X.

[16]. J. Novotny “The Grid Portal Development” Kit IEEE Concurrency: Practice and Experience, 2002.

[17]. J. Novotny , S. Tuecke and V. Welch “An Online Credential Repository for the Grid: MyProxy” Proceedings of the 10th. International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August, 2001.

[18]. M. Dahan, M. Thomas, E. Roberts and A. Seth, “Grid Portal Toolkit 3.0 (Gridport)” in Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing, 2004.

[19]. X. Wang, X. Yang, and R. Allan, “Top Ten Questions. *To* Design A Successful Grid Portal*”*, Proceedings of the Second International Conference on Semantics, Knowledge, and Grid, 2006.

[20]. O. F. Rana and J. C. Cunha “Grid Computing: Software Environments and Tools” Springer-Verlag January 2006.

[21]. T. Suzumura, H. Nakada, S. Matsuoka and H. Casanova, “GridSpeed: a Web-based Grid Portal generation server” in High Performance Computing and Grid in Asia Pacific Region, 2004. Proceedings. Seventh International Conference on 20-22 July 2000.

[22]. T. Suzumura et al "The Ninf Portal An Automatic Generation Tool for Grid Portals" Proceedings of the 2002 Joint ACM-ISCOPE Conference on Java Grande 2002, Seattle, Washington, USA, November 3-5, 2002.

[23]. C. Zhang, I. Kelley and G. Allen "Grid Portal Solutions: A Comparison of Gridportlets and OGCE" 27 june 2007.

[24]. R. Buyya, S. Chapin and D. DiNucci "Architectural Models for Resource Management" in First IEEE/ACM International Workshop on Grid Computing (GRID 2000)LNCS Series, Springer Verlag, Germany (2000).

[25]. X. D. Wang, X. Yang and R. Allan "Top Ten Questions To Design A Successful Grid Portal" International Conference on Semantics, Knowledge and Grid (SKG 2006), 1-3 November 2006, Guilin, China 2006.

[26]. I. Kelley, J. Novotny, M. Russell and O. Wehrens "JetSpeed Evaluation" WP4 Internal Document May,2002;

[27]. X. Yang, M. T. Dove et al "Survey of Major Tools and Technologies for Grid-enabled Portal Development" Proceedings of the UK e-Science All Hands Meeting 2006.

[28]. J. Novotny, M. Russell and O. Wehrens "GridSphere: A Portal Framework For Building Collaborations" International Middleware Conference, Workshop Proceedings, June 16-20, 2003, Rio de Janeiro, Brazil 2003.

[29]. K. Czajkowski, S. Fitzgerald, I. Foster and C. Kesselman "Grid Information Services for Distributed Resource Sharing." 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10 2001), 7-9 August 2001, San Francisco, CA, USA 2001

[30]. G. Aloisio, M. Cafaro et al "iGrid, a Novel Grid Information Service " Advances in Grid Computing - EGC 2005, European Grid Conference, Amsterdam, The Netherlands, February 14-16, 2005, Revised Selected Papers 2005.

List of Publications

[1]. Virendra Kumar and Dr. Indeerveer Chana, "Comparative Study of Grid Portals", National Conference on Emerging Trends in IT and Computing(ETIC-2010), March 27-28, 2010, GITM, Gurgaon, Hariyana.