GMIS Framework Architecture for Resource Management

This chapter presents the Grid Management Information Server (GMIS) architecture for resource management in grid environment. GMIS framework addresses the resource management requirements like resource discovery, monitoring and brokerage of resources in grid environment. It also discusses about distributed GMIS architecture.

3.1 GMIS Architecture Overview

The GMIS framework has been developed to address growing complexity and manageability in grids. Very few frameworks have been attempted to solve the grid management issues in real time. The architecture of GMIS framework described in this thesis is an attempt to provide solutions to the issues in resource management; discovery, monitoring and brokerage as mentioned in chapter 1, section 1.6.

The GMIS framework provides services for discovery, monitoring and brokerage of resources in a grid network. Discovering resources on a grid is an automated process which keeps track of addition and deletion of resources from the grid. To gather the information about resources, communicator server and communicator agents are used. Resources details are stored in a relational database on GMIS.

Monitoring service is used to monitor resources on grid level, cluster level and node level. Using this service, grid resources are visually represented as icons on a viewer with resource status. GMIS framework allows viewing of summary information of the individual node and summary view of cluster which is comparable and analyzable.

Brokerage service allocates the best possible resource or a combination of resources for a job execution based on set of attributes given by grid users.
GMIS architecture has been modeled as nine distinct components with specific functionalities and interfaces. The following figure 3.1 shows the architecture of GMIS.

![GMIS Architecture Diagram]

**Figure 3.1: GMIS Architecture**

1. **Discovery Server**: Discovery Server discovers resources on a grid. Resource discovery is at three levels: (i) Node level (ii) Cluster level and (iii) Grid level. In node level, an Internet protocol address or Fully Qualified Domain Name (FQDN) is used to discover resource information. In cluster level and grid level, a range of Internet protocol addresses are used to discover resources. User provides the information on Internet protocol addresses or FQDN, using user interface on Viewer. Discovery Server sends a request to resource agent software using communicator server. Upon receiving the request, resource discovery agents on resources gather resource information and sends to Discovery Server. The resource information is stored in the relational database through Topology Manager.

2. **Data Collection Manager**: Data Collection Manager provides the procedures to communicate with the agent software for information about the resources. It receives notifications from agents, and initiates actions accordingly. It polls resources for current status of resources and parses the poll responses. It updates the Topology Manager with the current resource information. It also handles events like addition of resource, deletion of resource and modification of resource.
(3) **Database:** Database contains information about the resources on grid. The data will get updated by Topology Manager and retrieved by other services like Monitor Service and Resource Selector.

(4) **Topology Manager:** Topology Manager maintains all resources information in a tree structure called Management Information Tree (MIT) as managed objects. Managed objects are created when there is a notification from the Discovery Server. Managed objects are modified or deleted based on the messages from Data Collection Manager. It also updates Database when ever there is a change in resource information.

(5) **Monitor Service:** Monitor Service provides information to the viewer by retrieving the resource information from Database. It controls and maintains the various users' access levels. The access information is persistently stored in the database. Each grid can have different or same access levels for the same user. Viewer provides interface for 'user access' configuration.

(6) **Resource Selector:** Resource Selector is responsible for generating a set of best possible resources for the given set of resource attributes. The entire resources information is in database and Topology Manager updates the information. Resource Selector takes inputs from user and searches in the database. If appropriate resources data is available in the database, those details will be provided to the user. Users can use these resources for their job execution. A user interface is provided on viewer to submit jobs. Key features of Resource Selector are:
   i. Providing Brokerage Service
   ii. Generate Activation Graph
   iii. Job execution

(7) **Communicator Server:** Communicator server establishes and maintains communication between the GMIS and resources agent software. Agent software on resources is designed to provide resource information and to communicate with GMIS on behalf of resource.

(8) **Grid Middleware:** Grid Middleware like schedulers, resource allocation managers interact with the GMIS services using Application Programming Interfaces (APIs). For instance, when a scheduler needs a resource for a particular application, it interacts with the Topology Manager through the resource selector, and a suitable resource is provided.
Viewer: Viewer interacts with the GMIS services through monitor service e.g., status information of resources. Viewer is a web based application uses Tomcat Apache as a Web Server. Icons are used to represent grid, cluster and node. Viewer gets resource information from database using Java Servlets. Viewer contains user interfaces for user management, discovery of resources, monitoring services, resource selection and job submission.

3.2 Discovery Server

Discovery server discovers all resources dynamically in a grid network using communicator agents. Discovery Server is an automated process to discover resources on a grid. Resource details are stored in relational database on GMIS through Topology Manager. The following figure 3.2 shows how the resources are discovered and stored in GMIS database and provides information to services like monitor service.

![Diagram of Discovery Server](image)

Figure 3.2: Discovery Server

Two types of discoveries are supported in GMIS system. One is hardware resources discovery and other one is services discovery. Hardware resource discovery is at three levels: (1) Node level (2) Cluster level and (3) Grid level. At node level, an Internet protocol address or Fully Qualified Domain Name (FQDN) is used to discover
resource information. At cluster level and grid level, a range of Internet protocol addresses are used to discover resources. A user interface is provided on Viewer to give Internet protocol addresses or FQDN. Discovery server process called Resource Discovery Manager (RDM) runs at the server which is part of GMIS. RDM requests communicator agents for the hardware resource information through communicator server. Resource discovery communicator agents running on resources will gather the resource information and sends it to RDM. The resource information is stored in the relational database through Topology Manager.

Similarly, hardware resource discovery methodology has been used for services discovery also. However, there is one limitation i.e., services should be in a particular directory, which is configurable, or services name should start with “UH_”.

Computing resources have two types of attributes: (i) static attributes such as type of operating system installed, network bandwidth, processor speed and storage capacity (including physical and secondary memory) and (ii) dynamic attributes such as processor utilization, physical memory utilization, free secondary memory size, and network bandwidth utilization; and applications have many types of attributes.

Static attributes information will be gathered only one time by Discovery Server, since this data will not change with time. At the time of initialization dynamic attributes information will be gathered by Discovery Server. As dynamic attributes information changes with time periodic information gathering is required. Data Collection Manager will collect the resource dynamic attributes information periodically.

3.3 Data Collection Manager

Data Collection Manager (DCM) sends request to communicator agents running at resources to collect resource information periodically. Communicator agents respond to the DCM request with the latest resource information. Data Collection Manager updates this information in topology and database through Topology Manager. A communicator agent, Critical Information Agent (CIA) sends notification events to DCM when there is a change in critical resource information. For example, if CPU usage exceeds 80% then CIA sends a critical event message to DCM. If a node is
added/modified/deleted then also an event message will come from CIA to DCM. The main functionalities of DCM are:

- Poll resources
- Parsing resource responses
- Event handling

### 3.3.1 DCM Polling and Parsing Mechanism

DCM periodically polls resources for current status of resource dynamic attributes. Communicator agents running at resources respond to DCM poll request. DCM parses the responses coming from communicator agents and updates Database through Topology Manager (TM). The following figure 3.3 shows the polling-parsing approach for CPU load which is a dynamic attribute of a computing resource.

**Figure 3.3: Polling and Parsing Mechanism in GMIS**

1. The DCM sends a request to the communicator agent for current CPU load using Communicator Server.
2. The communicator agent finds the current CPU load, responds with CPU load value.
3. DCM parses the response that comes from communicator agent and updates Database through Topology Manager.
4. DCM waits for a while before making another call according to the polling interval.
5. Steps 1-4 repeats for the next time interval.

3.3.2 DCM Event Handling

If any critical change like addition of resource, deletion of resource or modification of resource, occurs then a resource communicator agent called Critical Information Agent (CIA) sends a notification event to DCM through Communicator Server. The following figure 3.4 shows a notification event handling.

![Figure 3.4: Event Handling in GMIS](image)

1. If a resource ‘A’, CPU load exceeds 80%, then Critical Information Agent (CIA) which runs on resource sends a notification event to DCM through communicator server.
2. DCM updates the Database, through Topology Manager.
3. Once the resource ‘A’, CPU load drops below 80%, then again a notification event comes to DCM.
4. DCM updates the Database, through Topology Manager.
3.4 Database

GMIS framework database is a relational database that maintains resource information. It maintains resource information in a tree-like hierarchical representation of the relationships between the node, cluster, and grid. Nodes are physical computing resources. A cluster is a logical container that contains a group of nodes. Similarly, a grid is a logical container that contains a group of clusters. For example, in the following figure 3.5, node ‘n’ belongs to cluster ‘c’, so in node ‘n’ resource information parent of node ‘n’ is cluster ‘c’. Similarly, cluster ‘c’ belongs to grid ‘g’, so in cluster ‘c’ resource information parent of cluster ‘c’ is grid ‘g’.

![Figure 3.5: Relationship among Node – Cluster – Grid](image)

3.4.1 Data Handler

Data Handler is a single interface for all Database data handling. It acts as a wrapper for all SQL queries and updates to the database. Some of the handlers are:

- Handling the parameters associated with each resource:
  - Creating the instance of resource.
  - Retrieving the instance of resource.
• Modifying the parameters associated with each resource.
• Retrieving the parameters associated with each resource.

• Handling the hierarchy of the resources:
  • Traversing through the hierarchy
    • Retrieving the parent for a given resource.
    • Retrieving the child of a given resource.

### 3.5 Topology Manager

Topology Manager is backbone of the GMIS framework. It provides mechanism to create and maintain resource object hierarchy for the resources in grid network. Each resource is represented as a managed object. Topology manager stores, retrieves and manipulates data of managed objects based on updates from Discovery Server and Data Collection Manager.

Every resource on grid is considered as a Managed Object [RFC 1155] in Topology Manager. These objects are represented by one or more objects in the GMIS database. Communicator agent running at the resources is relied on to get the status of the resources.

Although a managed object need not be a physical object that can be seen, touched, and felt, it is convenient to use a physical representation to understand the characteristics and operations associated with a managed object.

Following figure 3.6 shows the topology manager associated with grid network configuration through Communicator Server, Discovery Server and Data Collection Manager. The topology manager has a database (DB) which contains the measured or administratively configured value of the resources of a grid network. This topology manager database is nothing but the GMIS database.
3.5.1 Topology Modeling

Topology is a central repository for all the resource managed objects in the grid network. During initialization topology is created based on inputs from discovery server or retrieving resource data from Database. Topology gets updated based on data collection manager or discovery server updates. Topology modeling provides a model to access, create, modify and manipulate the data in the topology. In topology modeling Management Information Base (MIB) is used to represent all managed objects.

An object identifier (or object ID) uniquely identifies a managed object in the MIB hierarchy. The MIB hierarchy can be depicted as a tree with a nameless root, the levels of which are assigned by different organizations.

Managed objects are uniquely defined by a tree structure specified by the Open Systems Interconnection (OSI) model and are used in the Internet model. The following figure 3.7 shows the generic representation of the Management Information Tree (MIT). There is a root node and well-defined nodes underneath each node at different levels. Each managed object occupies a node in the tree. In the OSI model, the managed objects are defined by a containment tree that represents the MIT.
The following figure 3.8 shows the internationally adopted OSI MIB tree. The root node does not have an explicit designation. There are three nodes in the layer beneath the root: iso, ccitt (itu), and iso-ccitt (iso-itu). The iso defines the International Standards Organization and itu defines the International Telecommunications Union (the old name is ccitt). The two standards organizations are on the first layer and define management of objects under them. The joint iso-itu node is for management objects jointly defined by the two organizations. The number in each circle identifies the designation of the object in each layer. Thus, iso is designated as 1 and org as 1.3, dod (Department of Defense) as 1.3.6 and the internet as 1.3.6.1. All Internet managed objects will be that number followed by more dots and numbers.

In this MIB tree, the top-level MIB object IDs belong to different standards organizations, while lower-level object IDs are allocated by associated organizations. Vendors can define private branches that include managed objects for their own products. MIBs that have not been standardized typically are positioned in the experimental branch.
The managed object at Input can be uniquely identified either by the object name—iso.identified-organization.dod.internet.private.enterprise.cisco.temporary variables.AppleTalk.atInput—or by the equivalent object descriptor: 1.3.6.1.4.1.9.3.3.1.

Figure 3.8: MIB Tree illustrates various hierarchies assigned by different organizations [Internetworking, 2001]
Topology model deals with the structure and the storage of information. The representation of objects (resources) and information relevant to their management form the topology model. As discussed in earlier sections, resource information is passed between the resource communicator agent and topology manager through GMIS components like Communicator Server, Discovery Server and Data Collection Manager. The topology model specifies the information base to describe managed objects and their relationships. The Structure of Management Information (SMI) defines the syntax and semantics of management information stored in the MIB. The MIB is used by both communicator agent and topology manager to store and exchange management information. The MIB associated with a communicator agent is called the agent MIB and the MIB associated with a topology manager is designated the manager MIB. A manager MIB consists of information on all the grid network resources that it manages, whereas an agent MIB needs to know only its local information.

Following figure 3.9 expands the topology manager configuration shown in figure 3.6, to include the MIB associated with the topology manager. Thus, the topology manager has both the database (DB) and the MIB. It is important to distinguish between the DB and MIB. The DB is a real database and contains the measured or administratively configured value of the resources of the grid network. On the other hand, the MIB is a virtual database and contains the information necessary for processes to exchange information.

**Figure 3.9: Topology Manager with MIB**
Let us illustrate the distinction between MIB and DB by considering the scenario of adding a new computing resource to the network. Assume all the computing nodes in the network are made by a single vendor, say SUN. In the above figure 3.9 the Topology manager’s knowledge about SUN computing nodes and their associated parameters are in its DB. For example, number of CPUs in the computing nodes is a parameter associated with the computing node (MIB information) and if they are dual-CPU nodes, the value associated with the number of CPUs is 2 (DB information). Suppose we added another SUN computing node to the grid network. The Topology manager would recognize the addition of resource to the grid network through Data Collection Manager. The new computing node is another instance of the computing node with a new IP address, and its MIB information is already in the manager's MIB. The node IP address and the number of CPUs associated with it are added to the DB by the topology manager.

Now, let us add an IBM computing node to the network. Let this be the first time that an IBM computing node is added to the network. The topology manager would recognize the addition of a new resource to the network through data collection manager. However, it would not know what component has been added until the MIB information on the computing node is added to the manager's MIB. This information is actually compiled into the manager’s MIB schema. After the information on the IBM computing node has been added to its MIB, then only data collection manager can communicate with the communicator agent residing in the IBM computing node. DCM then retrieves the values for the type of computing node, the number of CPUs, and so on, and adds them to its DB.

The MIB that contains data on managed objects need not be limited to physical elements. For example, in grid network management, management information extends beyond that associated with the description of network elements or objects. Here are some examples of information that can be stored in the MIB:

- **Grid Network Elements**: hubs, bridges, routers, transmission facilities
- **Software Processes**: programs, algorithms, protocol functions, database
- **Administrative Information**: contact person, account number
Host Resources MIB [RFC2790, 2000] has been used in the GMIS framework. The Host Resources MIB defines a uniform set of objects useful for the management of host computers. Host computers are independent of the operating system, network services, or any software application. The Host Resources MIB defines objects which are common across many computer system architectures.

### 3.6 Monitor Service

Monitor service provides resource data from the database to viewer as shown in the following figure 3.10. Monitor service has two functionalities, one is access control and other is status monitor.

Monitor Service controls and maintains the various users’ access levels. The access information is persistently stored in the database. Each grid can have different or same access levels for the same user. Viewer provides a user interface for ‘user access’ configuration. At the time of opening viewer a user interface obtains user’s ‘login’ and passes it to the Monitor Service for verification. If verification succeeds then only user is able to view the grid network information.

![Figure 3.10: Monitor Service in GMIS](image)
Monitor Service handles the viewer interface for resource status display. This keeps track of all the users that are registered with the GMIS and takes the responsibility of updating all the clients whenever there is change of resource state on the GMIS.

3.7 Resource Selector

Grid resource brokering is defined as the process of making scheduling decisions involving resources over multiple administrative domains [Schopf, 2002]. This includes searching multiple administrative domains to use a single machine or scheduling a single job to use multiple resources at a single site or multiple sites. A Grid broker must make resource selection decisions in an environment where it has no control over the local resources. The resources are distributed, and information about the resources is often limited.

The resource broker performs a number of basic functions. The first step is the identification and selection of resources that best fit the needs of the Grid application. The broker will then submit jobs in the application to the chosen machines. The broker thus handles submission of jobs but not how the job is actually executed on the resource, as that is part of the management system that resides on the resource involved. The resource broker functionality is done by the Resource Selector component of GMIS. Once jobs are being executed the broker for that application monitors the resources and the progression of the jobs through the Data Collection Manager. The following figure 3.11 shows Resource Selector component in GMIS.

Key features of Resource Selector are:

1. **Resource Selection**: Based on the user inputs, retrieve appropriate resource information from GMIS database and provide to the user. A user interface is developed on viewer to provide user inputs.

2. **Generate Activation Graph**: Activation graph is specification of the order of services to be executed. Activation graph will be created or modified using the user interface provided on viewer.
(3) **Job execution:** Grid users submit jobs using the interface provided on viewer. Resource Selector executes these jobs on behalf of the users.

![Diagram of Resource Selector Service in GMIS](image)

**Figure 3.11: Resource Selector Service in GMIS**

### 3.8 Communicator Server

A key component of the framework is Communicator Server (CS). The CS establishes and maintains communication between the GMIS and resources. At GMIS, Communicator Server runs along with GMIS, and communicator agents run on resources. Resources on grid network are classified as managed and unmanaged objects or elements. The managed elements have a management process running in them, called Communicator Agents. The unmanaged elements do not have a Communicator agent. The Communicator Server communicates with the agent in the managed element. The following figure 3.12 illustrates this.

For example, if an object that is managed by GMIS Communicator Server is selected on the viewer to submit a job. Communicator Server invokes a Communicator Agent on the corresponding object resource to execute the job.
The Communicator Server messages are exchanged using the connectionless UDP transport protocol in order to be consistent with simplicity of the model, as well as to reduce the traffic. This also avoids the tight linkage between the Communicator Server and agents. However, the mechanisms of the Communicator Server are suitable for a variety of protocols. Following figure 3.13 shows, how event handling is done between Communicator Agent and Communicator Server using UDP transport protocol.
The following figure 3.14 shows how data exchanged between communicator server and communicator agent using UDP transport.

![Figure 3.14: UDP Transport – Normal Send – Response exchange](image)

3.8.1 Packet Data Representation in Communicator Server

Management information communicated by Communicator server is represented according to the subset of the Abstract Syntax Notation One (ASN.1) language [ASN.1, 1987] that is specified for the definition of non-aggregate types in the Structure of Management Information (SMI). This extends this tradition by utilizing a moderately more complex subset of ASN.1 for describing managed objects and for describing the protocol data units used for managing those objects. In addition, the desire to ease eventual transition to OSI-based network management protocols led to the definition in the ASN.1 language of an Internet-standard Structure of Management Information (SMI) [RFC1065] and Management Information Base (MIB) [RFC1066].
The Communicator Server uses only a subset of the basic encoding rules of ASN.1. Namely, all encodings use the definite-length form. Further, whenever permissible, non-constructor encodings are used rather than constructor encodings. This restriction applies to all aspects of ASN.1 encoding, both for the top-level protocol data units and the data objects they contain.

ASN.1 is more than just syntax. It is a formal language developed jointly by CCITT (now ITU-T) and ISO for use with application layers for data transfer between systems. It is also applicable within the system for clearly separating the abstract syntax and the transfer syntax at the presentation layer. Abstract Syntax is set of rules used to specify data types and structures for storage of information. Transfer syntax represents the set of rules for communicating information between systems. Thus, abstract syntax would be applicable to the information model and transfer syntax to the communication model. The abstract syntax can be used with any presentation syntax, depending on the medium of presentation. The abstract syntax in ASN.1 makes it independent of the lower-layer protocols. ISO 8824/X.208 standards specify ASN.1. The algorithm to convert the textual ASN.1 syntax to machine readable code is called basic encoding rules (BER) and is defined in ISO-8825/X.209 standards.

ASN.1 is based on the Backus system and uses the formal syntax language and grammar of the Backus-Naur Form (BNF).

3.9 Grid Middleware

The GMIS framework is generic framework and can be integrated with other grid platforms like Globus [Foster, I, C. Kesselman, 1997] and other grid services like schedulers [Pramod Kumar, 2006]. GMIS provides Application Programming Interfaces (APIs) to access Topology manager and Database to access resource information.

3.10 Viewer

Viewer is a web based application. In the GMIS framework the web server chosen is Tomcat Apache web Server. Icons are used to represent grid, cluster and node. Viewer retrieves resource information through Monitoring Service.
Following are the main user interfaces provided in the viewer:

- **User Management** – used for authentication purpose and to register new users.
- **Discovery of resources** – to discover resources in grid by using Internet protocol addresses.
- **Resource Selector** – to select a resource based on the inputs from user, to generate activation graph and to execute job.
- **Monitoring of resources** – to monitor resources at grid level, cluster level and node level. This provides jobs status also.

### 3.11 Distributed GMIS Architecture

The distributed GMIS architecture provides the ability to distribute the Management Information Base (MIB) and resource information to multiple Grid Management Information Servers. Every GMIS allows users to have transparent access to management data irrespective of the GMIS on which the data is located. The data may reside in the local GMIS or on a remote GMIS in another geographical region. This architecture is centered on the concept of managed objects, and is shown in Figure 3.15.

![Figure 3.15: Distributed GMIS Architecture](image)

Access to all objects is achieved through a hierarchical structure called Management Information Tree (MIT). The MIT follows the globally defined object naming convention. When a connection is initiated by the user using Application Programming...
Interface (API) from GMIS A to GMIS B, the local MIT of GMIS B is mounted into GMIS A, and becomes visible in the viewer of GMIS A.

Distributed GMIS architecture contains the following components:

- **Resources** – Every physical resource on grid network is represented by one or more objects in GMIS topology manager, which we call them as managed objects. Performing an action like retrieving status information on a managed object is similar to performing that action on the corresponding resource.

- **Agent software** – This software provides objects level communication procedures between the managed resources and the GMIS. For instance, performing GET and SET procedures on managed resources.

- **Grid Management Information Server (GMIS)** – GMIS provides information services. It contains a relational database and a Management Information Tree (MIT) in topology manager.

  GMIS maintains resource information and relationships of all managed resources on a grid. Multiple GMIS connections are possible so that all Grid Management Information Servers information appear as one GMIS to any user. The GMIS supports dynamic creation, deletion and updating of objects in the MIT.

### 3.11.1 GMIS-GMIS Communication

The main function of the GMIS is to provide data about managed resources to its client services. The term managed resources includes both the physical resources connected to the network, such as nodes, as well as software resources such as applications.

Naming of managed resource instances is based on a containment relationship. Containment can be visualized as a directed graph with each directed edge pointing from a contained managed resource to a containing managed resource. Containment, as used for naming, implies a hierarchy.

A management system such as the GMIS provides services to a predetermined set of managed objects in the MIT. Since the objects for which the GMIS provides
services are a subset of the objects in the global name space, the GMIS must be able to act as a manager as well as an agent. The division of roles is determined by the configuration of the GMIS. When a request is made for an object which is non-local to the GMIS, it is forwarded to the appropriate agent. In this case the GMIS acts as a manager. If the object is local to the GMIS, the information is retrieved and returned to the requester. In this case, the GMIS acts as an agent. The manager role is achieved by allowing the user to configure parts of the MIT represented by the GMIS as being non-local. Thus, the requests to that part of the MIT are forwarded to the agent configured to manage that part of the tree.

GMIS-GMIS communication is achieved when two or more GMISs are connected and set up to forward requests for managed objects in manager and agent roles. The GMIS forwarding a request is acting in the manager role, while the GMIS receiving the request is acting in the agent role. To an application, it is transparent as to whether the object being accessed is local or non-local. The GMIS-GMIS communication (GGC) between GMISs makes it transparent.

3.12 Summary

This chapter discussed about the GMIS framework architecture and its components for resource management in grid environment. GMIS has been developed to address the resource management requirements like resource discovery, resource monitoring and brokerage. It also discussed how GMIS framework can be distributed in the grid network.

Next chapter discusses on the design aspects of GMIS framework along with data flow diagrams and data models.