A GLOBALLY DISTRIBUTED GRID MONITORING SYSTEM TO FACILITATE HIGH-PERFORMANCE COMPUTING AT DØ/SAM-GRID (DESIGN, DEVELOPMENT, IMPLEMENTATION AND DEPLOYMENT OF A PROTOTYPE)

by

ABHISHEK S. RANA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2002

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervising professor, David Levine for his constant encouragement and valuable guidance throughout my research. He has instilled in me a passion towards scientific research that will prove to be vital as I continue to pursue my academic interests. I am also grateful to the DØ/SAM-Grid project's technical leader, Igor Terekhov, of the Computing Division, Fermilab. He has been the main source of inspiration behind accomplishments of this work.

I would also like to thank Jaehoon Yu, Department of Physics, UT Arlington, for his excellent mentorship and suggestions while I was conducting this work. I am immensely thankful to Bob Weems, Department of Computer Science and Engineering, UT Arlington; he has been a role model for me in the realm of advanced distributed computing.

I am also grateful to Thomas Rethard and Behrooz Shirazi, of the Department of Computer Science and Engineering, UT Arlington, for their enthusiasm and interest in my research work throughout my graduate studies.

Most importantly, the credit for the achievement of this work goes to the DØ/SAM-Grid team, specifically Andrew Baranovski, Gabriele Garzoglio and Igor Terekhov; their numerous original ideas form part of this work. I am grateful to Hannu Koutaniemi and Siddharth Patil for their great help during the entire work. Above all, I would like to thank Lee Lueking for his highly efficient management of the project.

My heartfelt thanks go to my family, and to my friends. This work could not have been accomplished without their support.

November 12, 2002

ABSTRACT

A GLOBALLY DISTRIBUTED GRID MONITORING SYSTEM TO FACILITATE HIGH-PERFORMANCE COMPUTING AT DØ/SAM-GRID (DESIGN, DEVELOPMENT, IMPLEMENTATION AND DEPLOYMENT OF A PROTOTYPE)

Publication No. _____

Abhishek S. Rana, M.S.

The University of Texas at Arlington, 2002

Supervising Professor: David Levine

A grid environment involves large scale sharing of resources that are distributed from a geographical or an administrative perspective. There is a need for systems that enable continuous discovery and monitoring of the components of a grid. In this work, we discuss the development and deployment of a monitoring system that has been designed as a prototype for the DØ/SAM-Grid. We have developed a system that uses a layered architecture for information generation and processing, utilizes the various grid middleware tools, and implements Integration and Enquiry Protocols using existing Discovery Protocols to provide a user with a coherent view of all current activity in this grid - in the form of a web portal interface. The prototype system has been deployed for monitoring of 11 sites geographically distributed in 5 countries across 3 continents. This work focuses on the DØ/SAM-Grid, and is based on the SAM system developed at Fermilab.

TABLE OF CONTENTS

ACKNOWLEDGMENTS ii

ABSTRACT iii

LIST OF FIGURES v

Chapter

1 INTRODUCTION 1

2 GOAL OF THE THESIS 5

3 BACKGROUND AND PREVIOUS WORK 7

4 A GLOBALLY DISTRIBUTED GRID MONITORING SYSTEM PROTOTYPE FOR THE DØ/SAM-GRID 14

5 CURRENT UTILIZATION OF THE WORK 34

6 CONCLUSIONS AND FUTURE WORK 56

BIBLIOGRAPHY 58

BIOGRAPHICAL INFORMATION 60

LIST OF FIGURES

Figure Page

4.1 The basic Architecture of the Monitoring System for DØ/SAM-Grid 17

4.2 Partitioning of the grid into various sites 20

4.3 An example of a Distinguished Name 22

4.4 A representation of Directory Information Tree designed for the system 23

4.5 An example of object class definitions used in the monitoring system 25

4.6 An example of a Machine-Ad used in the monitoring system..... 26

4.7 The information processing layers defined by the monitoring system 27

4.8 A realistic snapshot of the *health* of a grid 30

4.9 The set of four protocols used by the system..... 31

4.10 Simplified logic of protocols 33

5.1 The web interface used for launching the monitoring system 35

5.2 The basic status indicators used by this monitoring system 36

5.3 Monitoring system shown as launched for 3 different monitoring sites..... 37

5.4 A single monitoring site with all its execution sites or SAM-Stations 38

5.5 The legend displaying all the status indicators 39

5.6 Monitoring the version, start time, and the status of a Station 40

5.7 Monitoring the general status of projects at a Station..... 41

5.8 Monitoring the progress status details for a specific project 42

5.9 Monitoring the disks of a Station..... 43

5.10 Monitoring the groups associated with a Station..... 44

5.11 Information about the authorized users at a monitoring site of the grid..... 45

5.12 The system as launched to monitor submission sites on the grid 46

5.13 Monitoring a known submission site 47

5.14 Monitoring the status of a submitted job at a remote execution site 48

5.15 Monitoring real-life *production* SAM-projects 49

5.16 Monitoring *production* SAM-disks..... 50

5.17(a) Progress details of processes of a *production* SAM-project 51

5.17(b) Progress summary of a *production* SAM-project 52

5.18 Information providers (grid sensors) developed for the system 54

CHAPTER 1

INTRODUCTION

Humans continuously keep developing new methodologies to solve the complex science and engineering problems. The requirements of these new scientific methods, however, tend to supersede capabilities of the contemporary underlying technology. High-energy physics is one such scientific domain; it has immensely helped in exploring the phenomena that range from the smallest particles of nature to the largest galaxies in the universe. Some of the greatest breakthroughs in high-energy physics rely on experiments that generate huge amounts of data. The analysis of this data can be facilitated by computing technologies like high-performance computing, simulations, data analysis and distributed computation – which require availability of enormous computing power. However, with the current pace of advancements in high-energy physics experiments, the available computing power may need to be harnessed much more effectively. A *solution* to this task of finding solutions to complex scientific problems is *grid computing*. The global scientific community is paying a lot of attention to this enabling technology. With global computing grids being setup all over the world by international collaboration, there is a need for the ubiquitous monitoring of this distributed computing power.

1.1 The Grid

'The Grid' as a term in computing world, was formulated in the last decade. It referred to an envisioned advanced distributed computing paradigm with capabilities to ultimately assist in solving complex science and engineering problems beyond the scope of

existing computing infrastructures [BLUEPRINT]. The concept has evolved considerably over these years. The growing popularity has also resulted in various kinds of 'grids', common ones being known as Data grids, Computational grids, Bio grids, Cluster grids, Science grids, among many others [FOSTERARTICLE]. Effort is in progress to converge the concepts related to the architecture, protocols, and applications of these grids to formulate a *single* paradigm – the Grid.

The article in [FOSTERARTICLE] lists various characteristics that may be significant in determining whether a distributed computing system meets the requirements to be a grid. According to [FOSTERARTICLE], such a system qualifies to be a grid, which:

a. coordinates resources that are not under centralized control.
b. utilizes standard, open, general-purpose protocols and interfaces.
c. promises to deliver non-trivial qualities of service.

Furthermore, these grids – if they follow common inter-grid protocols for authentication, authorization, access control, resource discovery, resource access, and resource sharing – will attempt to congregate into the Grid.

The grid community is working towards this goal, and standardized protocols are expected in near future, that may revolutionize the computing world; as Internet Protocol (IP) did with the Internet.

The author, when using the word 'grid' or 'Grid' in the remaining part of this document, implies *a* grid and not *the* Grid, the latter still being a vision that has not been attained yet. More appropriately, it is the DØ/SAM-Grid [SAM-2] that is the prime focus of this work.

1.2 Need for Information Services and Monitoring of a Grid

A grid environment involves large-scale sharing of resources within various virtual

organizations (called VO's in common grid terminology). These VO's can comprise of conglomerations of individuals and institutions, leading to a complex and widely diverse user-base [MDS-2] located all over the globe. There arises a need for mechanisms that enable continuous discovery and monitoring of grid-entities (resources, services and activity). This can be quite challenging owing to the dynamic and geographically-distributed nature of these entities. Information services thus form a critical part of such an infrastructure, and result in a more planned and regulated utilization of grid resources. Any grid infrastructure should therefore have a monitoring system dedicated to this task, which should provide at a minimum – dynamic resource discovery, information about resources, information about the grid activity, and performance diagnostics.

1.3 Importance of Distributed Monitoring

Monitoring of a grid can be a challenging task given the fact that restricted quality of service, low level of reliability, and network failure are the rule rather than the exception while dealing with globally distributed computing resources. For such a system to be robust there should not be a centralized control of information. The system should exhibit its minimalist behavior for a maximum possible subset of the grid-entities even if certain information access points fail to perform their function. This robustness can only be achieved if:

a. The information services are themselves distributed, and geographically reside as close to the individual components of a grid as possible.
b. The monitoring is performed in as decentralized a fashion as possible under the constraints of the underlying resources' architecture.

A grid monitoring system should therefore, make sure that unavailable and/or unreachable services and resources [MDS-2] do not interfere with its normal functionality. Also, if there

is a provision for the awareness of such defaulting entities, it makes the monitoring more effective in reducing the turnaround time for the grid to recover from failures of its participating entities.

1.4 Ubiquitous Grid-monitoring using the Internet

A grid monitoring system that is restricted to command-line interface may fail to provide easy accessibility to the monitoring information. It is beneficial for the front-end of any grid monitoring system to utilize the client-server model of the Internet, and be able to provide up-to-date information transparently and comprehensively to a client while performing all the major tasks at the various layers of the backend.

Conclusively, an effective model for a grid monitoring system should be globally distributed, as much de-centralized as possible, and must be capable of providing access to information in a ubiquitous manner. This work describes design, development and implementation of such a prototype grid monitoring system.

The rest of the thesis is organized as follows: Chapter 2 lists the particular Grid-effort this work is related to, and outlines the goals of this thesis. The background and previous work that forms an essential foundation for this work, is discussed in Chapter 3. The author describes the design and development of a Globally-distributed Grid Monitoring System in Chapter 4. The current utilization of this work is discussed in Chapter 5, followed by Conclusions in Chapter 6 and Future Enhancements in Chapter 7.

This work was partially supported by the Department of Computer Science and Engineering, Department of Physics at UT Arlington, TX, USA; and the Fermi National Accelerator Laboratory, Batavia, IL, USA under Contract No. 546763; and conducted with the United States Department of Energy SciDAC program, the Particle Physics Data Grid (PPDG), and the Grid for UK Particle Physics (GridPP).

CHAPTER 2

GOAL OF THE THESIS

This work concentrates on the architecture of a specific grid – the DØ/SAM-Grid, an introduction to which is provided in this chapter. More detailed information about the terminology used in this grid is provided in the later chapters.

2.1 The DØ/SAM-Grid

The SAM system (an acronym for Sequential Access to data via Metadata) has been developed at Fermilab [SAM-1] to handle the data management of the Run II experiments of high-energy physics. In this context, sequential refers to the layout of physics events stored within files, which in turn are stored sequentially on tapes within a Mass Storage System (MSS). SAM provides transparent delivering of files and managing of caches of data. This system is the sole data management system being used by the DØ experiment at Fermilab, and other experiments like CDF at Fermilab have also started using this system.

Based on the success and popularity of SAM, a larger architecture has been conceived that enables the existing system to fully evolve into a grid known as the DØ/SAM-Grid. This architecture incorporates grid level job submission and management, and grid information services [SAM-1].

2.2 High level components of the DØ/SAM-Grid architecture

The principal architecture [SAM-2] of DØ/SAM-Grid defines 3 high level components to realize a fully functional grid:

a. Job Definition and Management
b. Monitoring and Information Services
c. Data Handling

The Job Definition and Management component consists of the job submission user interfaces, formulation of Job Definition Language (JDL), the Request Broker, and reliable submission of jobs to remote resources.

The Monitoring and Information Services comprises of Resource Information Service, monitoring of job submission sites, monitoring of the job execution sites, information about the advertised resources, and related information about the participating virtual organizations.

The Data Handling is provided by the core functionality of the SAM system itself, which is distributed in nature, and well-suited for a grid-like environment [SAM-2].

2.3 Goal of the Thesis

The goal of this thesis can be summarized as "*Design, Development, Implementation, and Deployment of a Prototype for the Monitoring and Information Services component of the initial DØ/SAM-Grid architecture*". Additionally, owing to the fact that DØ/SAM-Grid will play a key role in facilitating high-energy physics experiments that are being undertaken currently on a very large scale worldwide, the prototypical Monitoring System should be as *scalable* and *robust* as possible, and must cater to the demands of a *real-life grid*.

The next chapter describes in greater detail the background and previous work related to the technologies that form an essential foundation towards realizing this goal.

CHAPTER 3

BACKGROUND AND PREVIOUS WORK

The development of the grid monitoring system pertinent to this work utilizes various existing grid computing technologies. This chapter contains a brief description of these technologies, along with an introduction to the particular high-energy physics experiment that has played an important role in laying a foundation for the DØ/SAM-Grid.

3.1 High Energy Physics Experiment DØ at Fermilab

The DØ experiment consists of worldwide collaboration of scientists conducting research on the fundamental nature of matter [D0-PAGE]. The experiment is located at the world's highest high-energy accelerator, the Tevatron proton-antiproton Collider, at the Fermi National Accelerator Laboratory (Fermilab), Batavia, Illinois, USA [D0-PAGE]. The DØ experiment was proposed for the Fermilab Tevatron Collider in 1983 and approved in 1984. The experiment recorded its first antiproton-proton interaction in mid-1992. The data-taking period referred to as 'Run I' lasted through the beginning of 1996 [D0-DOC]. The analysis of data produced by Run I led to "the discovery of the top quark and measurements of its mass and production cross section, the precise determination of the mass of W boson and the couplings of electroweak bosons (photon, W and Z), and numerous searches for new physics" according to the [D0-DOC].

The next phase of data collection 'Run II' has already started with an upgraded detector and improved accelerator. The analysis of the huge amount of data generated O(PB) is believed to bring about new discoveries and significant results in particle physics.

The DØ collaboration today comprises of more than 650 physicists, over 78 institutions from more than 18 countries [LEE-SLIDES].

3.2 SAM (Sequential Access to data via Metadata) system

The SAM project started in 1997 to handle DØ's needs for Run II data system. SAM is an acronym for 'Sequential Access to data via Metadata'. The term *sequential* refers to the layout of physics events stored within files, which are in turn stored sequentially on tapes within a Mass Storage System (MSS) [SAM-1]. SAM performs the task of transparently delivering files and managing caches of data. It is the sole data management system of the DØ experiment; other major experiments like CDF (Collision Detection at Fermilab) have also started using this system.

The information that follows in this section has been derived from [SAM-1] and [SAM-PPDG]; more elaborate details are available in [SAM-1], [SAM-PPDG] and [SAM-PAGE] about SAM.

SAM has been designed as a distributed system, using CORBA (Common Object Request Broker Architecture) as the underlying framework [SAM-1]. The system relies on compute systems and storage systems distributed over the world. Storage systems have disk storage elements at all locations and robotic tape libraries at select locations. All the storage elements support the basic functionalities of storing/retrieving a file. The tape storage management systems are also called Mass Storage Systems (MSS), the one at Fermilab being Enstore [ENSTORE-PAGE].

Metadata catalogs, Replica catalogs, data transformations, and databases of detector calibration and other parameters are implemented using Oracle relational databases. Other experiments like Babar and LHC (Large Hadron Collider) divide their architecture into centers at particular geographic locations (e.g., Tiers), the DØ architecture is organized by

physical groupings of compute, storage, network resources termed as ***Stations***. Certain Stations can directly access the tape storage; others utilize routes through the ones that provide caching and forwarding services. The disk storage elements can be managed either by a Station or externally, those managed by Stations together form logical ***disk*** caches which are administered for a particular ***group*** of physicists. There are provisions for the pinning of files residing at a Station's disk storage elements according to the customizable quotas and various policies for each group [SAM-PPDG]. Stations own the resource partitions, yet they do not have exclusive control over them. For instance, for the same set of compute resources, there can be two different Stations sharing the compute elements but with distinct and disjoint disk storage elements. This aids in running ***production*** and ***development*** stations sharing the compute elements and tape storage systems but with discrete sets of files, catalogs, and disks.

In SAM, *service registration* and *discovery* has been implemented using CORBA Naming Service, with namespace by Station Name [SAM-PPDG]. APIs to services in SAM are defined using CORBA IDL (Interface Definition Language) and can have multiple language bindings. UDP is used for event logging services and for certain Request Manager control messages. Each disk storage element has a *stager* associated that serves to transfer or erase a file by using the appropriate protocol for the source and destination storage elements. Rcp, kerberized rcp, bbftp, encp (and now gridftp too) provide the file transfer protocols.

Each Station has a Cache Manager and Job Manager implemented as a ***Station Master*** server. The Cache Manager provides caching services and also the policies for each group. The Job Manager provides services to execute a user application either interactively or by using a supported batch system like LSF, FBS, PBS, and schedulers like Condor. Request Managers, which are implemented as ***Project Master*** server, take care of the pre-staging of file replicas and the book-keeping about the file consumption. The project master executes

for each dataset to be delivered and consumed by a user job. Storage Manager services are provided by a Station's *file storage server* that lets a user store files in tape and disk storage elements.

SAM has been in successful use for handling the Monte Carlo data O(TB) produced off-site from Fermilab. The SAM system has been in continuous use for computational farm production processing on a multi-node Linux Farm System, for various analysis purposes on a large 176-processor SGI origin 2000 SMP called 'dØmino', and on several Linux workstations.

3.3 Globus

The Globus grid-middleware is provided by the Globus project [GLOBUS-PAGE] bundled as a set of tools called Globus Toolkit. The toolkit has 3 components known as *pillars*. These are:

a. Resource Management
b. Information Services
c. Data Management

The Globus Toolkit uses the GSI (Globus Security Infrastructure) to provide a common security protocol for each of the pillars. GSI is based on public key encryption, X.509 certificates and Secure Socket layer (SSL) protocol.

3.3.1 GRAM (Globus Resource Allocation Manager)

The Globus Resource Allocation Manager provides a standard interface to all the local resource management tools a site uses. The Globus resource management has the high-level global resource management services layered on top of local resource-allocation services [GLOBUS-PAGE]. The GRAM service is provided by a combination of the

gatekeeper and the ***jobmanager***. The gatekeeper performs the task of authenticating an inbound request using GSI, and mapping the user's global ID on the grid to a local username. The incoming request specifies a specific local service to be launched, the latter usually being a jobmanager. The user needs to compose the request in a Resource Specification Language (RSL) that is handed over to the jobmanager by the gatekeeper. After parsing the RSL, the jobmanager translates it into the local scheduler's language. The GRAM also provides the capability to stage in executables or data files, using Global Access to Secondary Storage (GASS). The jobmanager contacts the client before the job submission for retrieval of the staged in files.

3.3.2 MDS (Metacomputing and Directory Service)

The Globus Metacomputing and Directory Service provides an LDAP based information infrastructure, suited for grid environments. There are 2 components of MDS implementation – ***GRIS*** and ***GIIS***. The Grid Resource Information Service (GRIS) implements a uniform means to query resources on a grid for current status and configuration. The Grid Index Information Service (GIIS) component of MDS provides a framework to form an index over various GRIS's or other GIIS's. This combines the information of an entire system, thereby giving a method to explore a coherent system image. Both these components [GLOBUS-PAGE] are currently implemented using the *slapd* server provided by OpenLDAP and follow the Lightweight Directory Access Protocol (LDAP). MDS utilizes the concept of *information providers* – software programs that act as probe utilities or sensors to the smaller components of a grid. Since it is based on LDAP, MDS needs a schema to be built in that represents the hierarchy and rules of the information to be retrieved from the components of a grid. Caching mechanisms are used to make the retention of data more efficient, based on the time-sensitivity of a piece of information.

3.3.3 GridFTP

This is a data transfer protocol [GLOBUS-PAGE] based on FTP, highly optimized to give secure and reliable performance in a grid. Among the various features it provides, the important ones are GSI security, partial file transfers, authenticated data channels, third-party (direct server-to-server) transfers. The protocol also allows developers to add plug-ins for customized reliability and fault tolerance features.

3.4 Condor and Condor-G

Condor provides an efficient job scheduling system for distributed computing environments. It aids in harnessing idle CPU cycles of workstations in a transparent manner to the owner of the idle workstation being utilized. Among other job-scheduling functionalities, it implements ***check-pointing*** by saving the current state of a remotely executing job, when it is suspended, to restart it from the same stage [CONDOR].

The Condor-G system integrates the two technologies: Globus and Condor. It incorporates features of distributed resource access for multi-domain environments provided by Globus, and the benefits of distributed resource scheduling for a single administrative domain offered by Condor. Condor-G system utilizes various components, chiefly:

a. A Condor-G job ***scheduler*** at the submission end
b. A Condor-G ***grid manager***, that works together with GASS at the submission end
c. A Globus ***gatekeeper*** and Globus ***jobmanager*** at the execution end

The authentication and authorization is implemented using GSI mechanisms. Condor-G gives highly reliable performance; it provides precise levels of check-pointing to resume execution of the jobs from the stage of suspension in a highly transparent and reliable manner. More technical details about this system are available in [CONDOR-G], the paper also provides

illustrations of the procedure that is undertaken from the time of submission of a job to its execution at a remote machine.

The Condor-G core development team has incorporated various features in the system for the DØ/SAM-Grid, that facilitate ***Resource Broker*** and ***Negotiator*** components to match the submitted jobs using customized *ranking functions* built for this grid .

3.5 DØ/SAM-Grid and JIM

JIM system [SAM-2] is an acronym for '**J**ob Management Infrastructure and **I**nformation & **M**onitoring System' and is being developed by the DØ/SAM-Grid to enable fully-distributed computing for high-energy physics experiments. JIM also aims to enhance SAM – the distributed data handling system of DØ and CDF, by incorporating standard grid tools and protocols, and developing new grid-solutions for other major grids.

The JIM architecture [SAM-2] utilizes most of the above discussed technologies, and builds a robust system for a grid catering to the demands of the various high-energy physics experiments. Some details of this system will be provided in the next few chapters; the design of specifc components of JIM pertinent for this work is also described in greater detail in Chapter 4 and Chapter 5.

CHAPTER 4

A GLOBALLY DISTRIBUTED GRID MONITORING SYSTEM PROTOTYPE FOR THE DØ/SAM-GRID

The Information and Monitoring System component of JIM is also referred to as JIM-IM in this chapter; it applies to the globally-distributed grid monitoring system for the DØ/SAM-Grid. Its prototype release has been deployed and tested at various sites distributed over Europe and United States. The initial phase prototype is restricted to 'sam-analysis' type of jobs, that are executed on SAM-Stations. Hence, the scope of this work focuses on a subset of the final DØ/SAM-Grid architecture.

4.1 Architecture

This section describes the architectural model of the JIM-IM, also providing the various requirements and design issues important for defining the functionality of this grid monitoring system.

4.1.1 Design Issues

The task of retrieving, organizing, and providing information has been studied in detail by many researchers, and methodologies exist (e.g., Database Systems, Simple Network Management Protocol or SNMP, Directory Services) that act as viable approaches strictly depending on the requirements of a particular system.

Following are some requirements and design issues that played a key role in formulating the design and implementation of JIM-IM.

(i) Efficient monitoring from submission through execution of a job: At a minimum, the system must be sufficiently capable of monitoring a job submitted to the grid by various possible routes:

a. Monitoring the submission site
b. Tracking the progress of the submitted job from the known submission site
c. Reaching the *unknown* execution site from the *known* submission site
d. Monitoring directly the execution site

(ii) Information integration: It should be possible for the monitoring system to retrieve information from different realms (e.g., Condor, Globus MDS, SAM) and perform integration over all these sets to provide the user with meaningful and comprehensive monitoring data.

(iii) Information abstraction: Only the relevant information should be presented to the user, all the back-end mechanisms should be transparently abstracted from the front-end layer of the system (the only layer visible to the user).

(iv) Performance and robustness: The monitoring system should take into account the demands of a distributed environment, and therefore must be tolerant of failures that may inevitably occur in parts of the grid, the grid-entities, and components of the grid-entities for that matter. This implies that:

a. The unavailable, unreachable or under-performing entities (for instance, can be an entire site of the DØ/SAM-Grid or a single SAM-station) should not interfere with the performance of the system.
b. The system must provide access to the largest subset of information that can be made available to the user in the event of failure in parts of the grid.
c. There must be provision for the system to notify the users/administrators the points of failure, if possible.

(v) Scalability: The system must facilitate scalability due to ever-increasing numbers of the entities of the grid. It must be easy to configure and reconfigure it to reflect the new components of the grid as the need arises. The reconfiguration procedures must be automated and easy to perform using the system's built-in functionality.

(vi) Uniform data representation: There must be a standard way of representing data related to the information about the various entities of the grid, viz., SAM-stations, SAM-projects, SAM-groups, SAM-users. This standard representation, if followed by all the components of a grid, must allow seamless retrieval of information across the entire grid.

(vii) Dynamic retrieval of information: Most of the monitoring data in SAM is highly dynamic in nature. The monitoring system must have the capability to retrieve data that is up-to-date and current, if it does not adversely affect the performance of the system. This implies that the monitoring information should be reflective of the real-time state of the grid, at any given instant of time.

(viii) Ubiquitous presentation through the Internet: The front-end layer to the system must have the capability to present information in the form of a web portal. This feature would greatly enhance the ease of use and ubiquitous availability of the system. The information should be available anytime, anywhere to anyone by means of web browsers using the Internet.

(ix) Ease of deployment: The system, for it to be widely used, must be easy to deploy and maintain across the grid. The installation and configuration of the backend layers and maintenance of the front-end layer must not incur considerable overhead. Since the deployment of the back-end layers, in practical cases, is performed by a site's local administrators, the installation and configuration procedures must be simple to operate.

4.1.2 Architectural Model of the System

The monitoring system follows the architecture described in figure 4.1, the information is generated and collected from the grid, processed for integration, and finally presented at a user's browser in the form of a web portal.

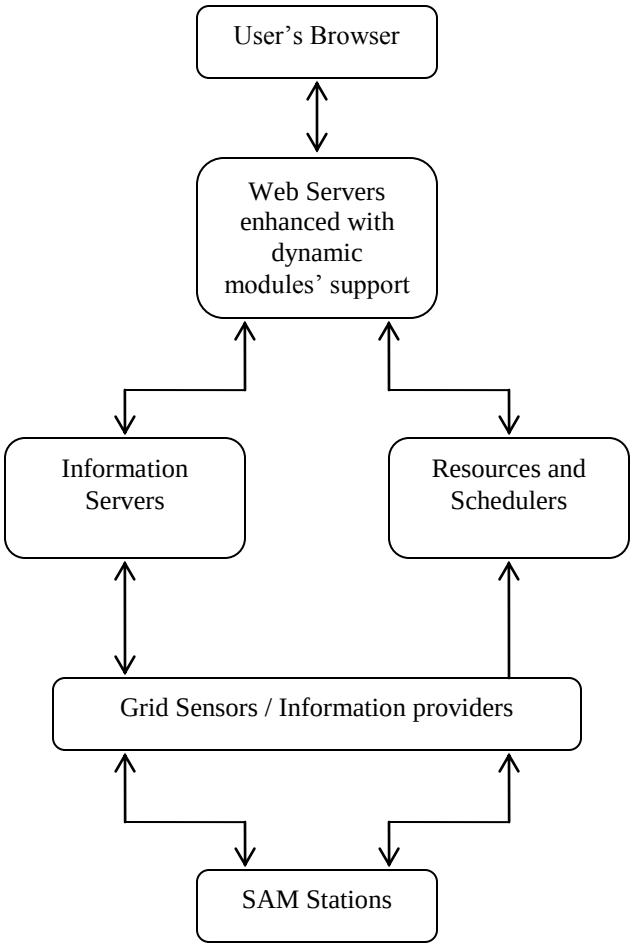

Figure 4.1 The basic architecture of the monitoring system for the DØ/SAM-Grid. The flow of information is depicted by the direction of arrows. All the bidirectional arrows represent an ***on-demand*** information retrieval. A Scheduler is not a SAM Station; it is a part of the grid-client package used for submission of jobs to the grid. Also, a Resource may not always be a SAM Station. Hence, the information retrieval using Grid-sensors is unidirectional in this case.

The architecture of this Monitoring System partitions the components of the grid according to the functionality exhibited by them as:

a. Submission Sites
b. Execution Sites
c. Monitoring Sites

The description of each of these categories is more evident in the following discussion of the common terminology used by JIM-IM system.

(i) Schedulers on the grid: A collection of grid-client packages that provides a user with an interface to authenticate and create a credential according to the security infrastructure, and thereafter submit jobs to the grid.

(ii) Submission sites: A subset of the schedulers on the grid. The user specifies the requirements of the job in the form of a Job Definition Language (JDL) that is parsed and the job passed on to the broker for match-making. After a successful match according to the user's requirements and the best available resource, the job is finally routed to the appropriate execution site.

(iii) Resources on the grid: These are the resources that have been configured to be utilized by the grid. A resource can be unavailable or available depending on its advertising itself to the grid at any given instant.

(iv) Execution sites: A subset of the resources on the grid. The jobs matched with the best available resource are routed to an execution site on the grid. For the scope of this work, it is assumed that an execution site is always a *SAM Station*.

(v) Monitoring site: This is a collection of submission sites and execution sites, grouped together within a single geographical or administrative domain. Commonly, information about all execution sites of a monitoring site is retrieved using different grid middleware than the one used to retrieve information about submission sites. Globus MDS is used to monitor

the execution sites, whereas Condor-G is utilized for monitoring of the submission sites. The SAM system is present beneath these different grid middleware realms. The monitoring system performs integration over information retrieved from these different software domains to provide a coherent view of the grid.

(vi) Grid sensors: A grid sensor is a service that is used to access current information about the individual grid-entities, and to notify the monitoring system of the availability of this information. Another common term for grid sensors is *information providers*. The sensors (information providers) need to be deployed at each monitoring site. The sensors have the capability to retrieve information from the grid components at various levels (monitoring site level, execution site level, and job level) and provide it to the backend layers of the monitoring system.

(vii) Information servers: Information servers are distributed over the grid, and are responsible for retrieving monitoring information from the grid using the grid sensors. The information servers serve as one of the backend layers to the monitoring system. The system utilizes the *slapd* servers of OpenLDAP software provided with the Globus MDS, as information servers.

(viii) Web server with JIM-IM engine: A collection of Linux Apache webserver (that has been configured to utilize dynamically loaded modules) and a set of PHP (Hypertext Preprocessor) scripts bundled as a software package – serve as an engine to the monitoring system. These PHP scripts process the information retrieved from the backend layers, and render dynamically produced web pages. The integration of information is also largely performed by this engine.

The partitioning of the entire grid into submission sites, execution sites and monitoring sites is represented in figure 4.2.

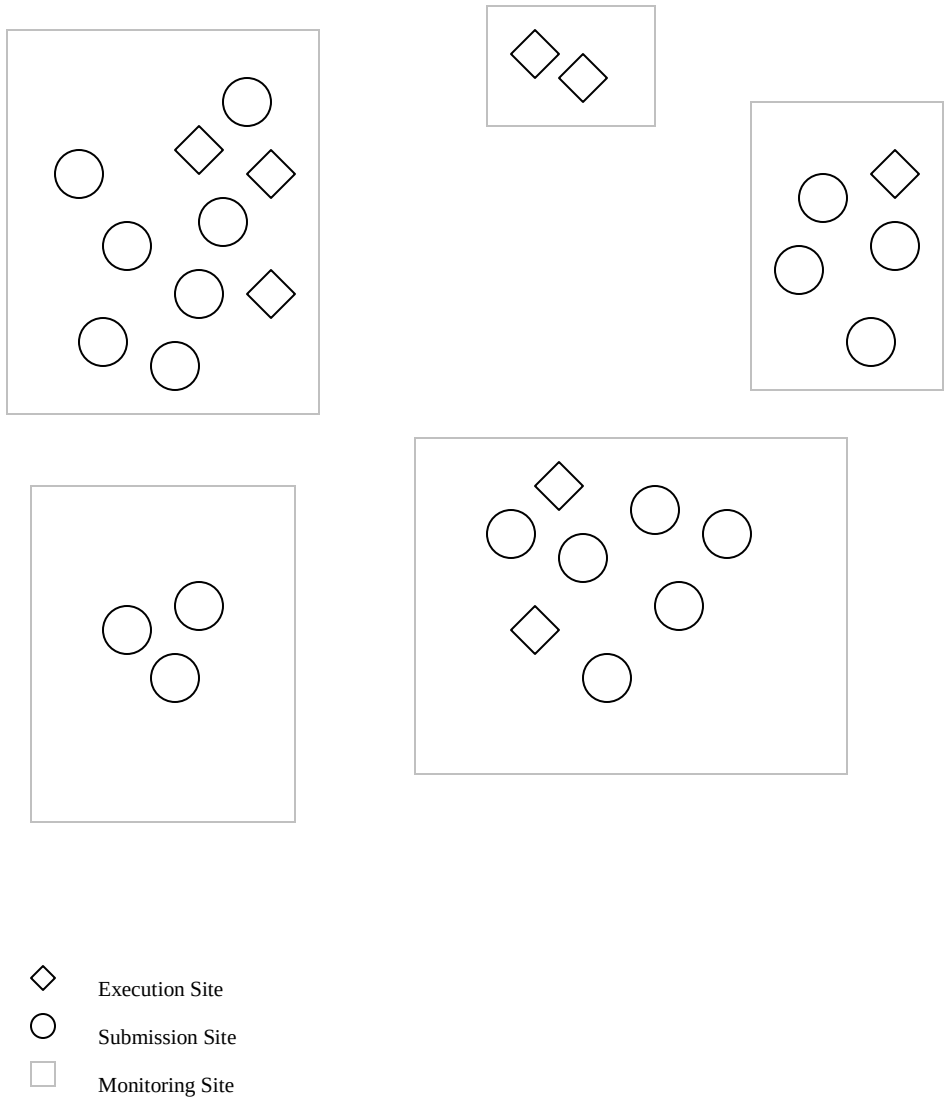

Figure 4.2 Partitioning of the grid into various sites. The monitoring system partitions the various components of the grid according to the functionality exhibited by them as: Execution Sites, Submission Sites, and Monitoring Sites. A Monitoring Site may comprise of multiple Execution Sites and/or Submission Sites grouped together within an administrative or geographical domain. For the purpose of the current prototype system, the monitoring of submission sites is performed discretely from the monitoring sites. However, in principle, the submission sites are geographically present within a monitoring site.

The architecture of the system is of an *On-Demand* nature (figure 4.1) for most of its components. Hence for those components, only an information request leads to its retrieval from the grid. This makes the monitoring system incur less cost on the system-resources being utilized for the information generation and processing. However, for monitoring the information from the submission sites, the architecture relies on periodic generation of data by Condor's internal mechanisms [CONDOR-ADS].

4.2 Design

The design of JIM-IM involves methodologies used for data representation; construction of data models; construction of information layers within the System; and formulation of several protocols that utilize several existing standards to facilitate efficient discovery of the grid-components and information retrieval in a fault-tolerant manner.

4.2.1 Representation of Information

Representation of information requires extensibility and flexibility. The Monitoring System utilizes for data representation:

a. Globus MDS [MDS-1], which in turn adopts the data representation and API defined by the LDAP [LDAP-1] Directory Service protocol based on X.500, and
b. Condor ClassAds [CONDOR-ADS], which provide information generated by Condor or Condor-G about the entities of the grid.

The MDS information model organizes related information into well-defined collections known as entries. MDS contains several entries; each represents an instance of a type of object. Information about an entry is represented by attributes, with name-value pairs. The attributes are associated with particular types of objects represented by an entry.

4.2.1.1 Naming Conventions for the Directory Information Tree

In order to identify an MDS entry uniquely, it needs to have a unique *Distinguished Name* (DN). All the entries (figure 4.4) form a hierarchical namespace called a *Directory Information Tree* (DIT). The DIT provides a faster and simpler way to search for a particular entry. The DN [RFC-2253] for a specific entry can be constructed using the entries on path from the DIT root to the node of the entry. For instance, the Distinguished Name shown in figure 4.3, lists an example of a generic DN to uniquely identify the SAM-Groups of a particular SAM-Station on the grid.

```
< Sam-View-Group-Name=groups,
  Sam-View-Group-Name=configuration,
  Mds-Software-deployment=SamStation-SAM_UNIVERSE-SAM_STATION,
  Sam-View-Group-Name=details,
  Mds-Software-deployment=Sam,
  Jim-Cluster=JIM_CLUSTER,
  Jim-Site=JIM_SITE,
  Mds-Vo-name=SAM_EXPERIMENT,
  o=samgrid >
```

Figure 4.3 An example of a Distinguished Name. A Distinguished Name is shown with its components listed in little-endian order. This DN uniquely identifies the SAM-Groups of a ***particular*** SAM-Station on the DØ/SAM-Grid. In order to locate this node on the Directory Information Tree, the path from the <o=samgrid> (the root of DIT listed at the end of the DN) to the experiment, followed by site, cluster, Station name and Universe, the particular entry corresponding to the configuration and disks – can be traversed.

4.2.1.2 Object Classes

Within the DIT as in figure 4.4, each entry is associated with a user-defined type, known as *Object Class*. The Monitoring System uses the extensions to the MDS/LDAP-defined [RFC-2256] standard object class definitions.

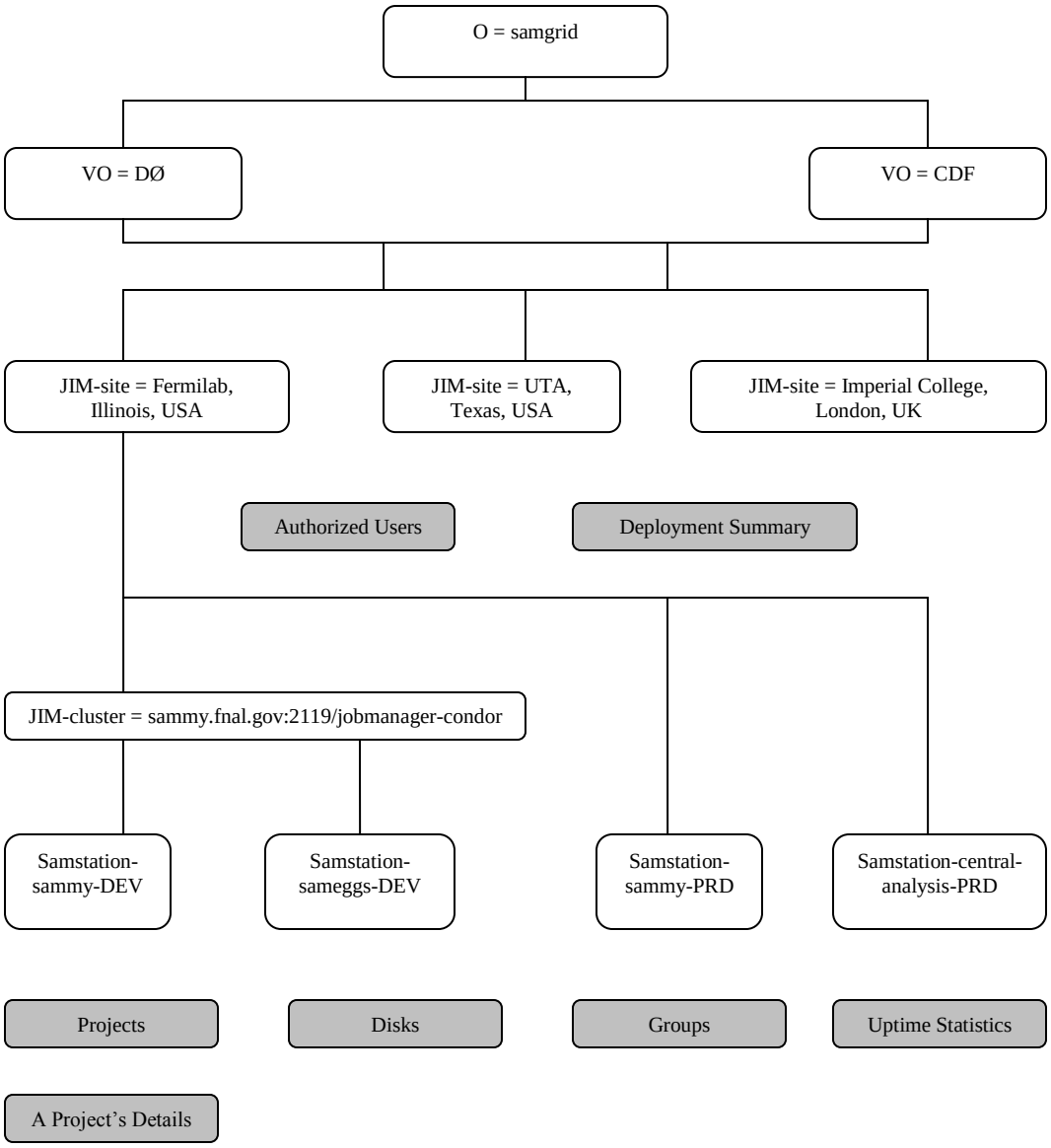

Figure 4.4 A representation of the ***Directory Information Tree*** designed for the system. The root specifies the *Organization*, each of its child nodes specify a *Virtual Organization*. Further, the *monitoring sites* are specified followed by the components within a site. Note that the shaded regions (***the invocation points on the tree for the Grid Sensors***) specify the location for the information related to the particular grid component one level higher. For brevity, the tree has been illustrated in a way so that the nodes and grid-sensor invocation points are shown for only a single node, related to a grid component, one level higher. The traversal of the DIT from the root node to a specific shaded region invokes the corresponding Grid Sensor. This leads to generation of information about the entry specified by the ***Distinguished Name*** formed by the *path* to this region.

All the attributes of an entry are characterized in the object class definition. An inheritance relation can also be made in this definition that extends an existing object class definition. An example of such a set of definitions is shown in figure 4.5.

In figure 4.5, the object class `'SamConfigurationGroups'` is a representation of a SAM-Group and inherits from the object class `'SamBase'`, so it is required to contain attributes `Sam-Server-Status`, `Sam-Group-Name`, and `Sam-Group-Administrators-Local-Id`. Other attributes are optional but provide a more descriptive view of the current status of Groups for a SAM-Station.

4.2.1.3 Class-Ad's from Condor

Condor [CONDOR] and Condor-G [CONDOR-G] have built-in mechanisms for representing resources, jobs, and schedulers. This information is used for brokering and match-making of jobs submitted to the grid. The JIM-IM makes use of this information and integrates it with the information received from other sources. Unique Global job-id's are generated by other packages of SAM-Grid, and also utilized for identifying jobs on the grid and monitoring them.

A simple Class-Ad is listed in figure 4.6, this is a type of Ad used for advertising a resource, evident from the value of the `MyType` attribute. This Ad seeks different types of Ads as defined by the `TargetType` attribute. The other attributes uniquely identify this entity (resource or job) amongst others of the same category.

Class-Ads for jobs are designed in a similar format, with a more complex representation. They are utilized to define the various requirements the job seeks to be met with a resource on the grid, the files involved, various arguments to be passed, and other platform/architecture details.

```
objectclass (
  NAME 'SamConfigurationGroups'
  SUBCLASS OF 'SamBase'
  MUST CONTAIN (
         Sam-Group-Name
         Sam-Group-Administrators-Local-Id
   )
   MAY CONTAIN (
         Sam-Group-Swap-Policy
         Sam-Group-Fair-Share
         Sam-Group-Quota-Projects-Current
         Sam-Group-Quota-Projects-Max
         Sam-Group-Quota-Disk-MB-Current
         Sam-Group-Quota-Disk-MB-Max
         Sam-Group-Quota-Locks-Current
         Sam-Group-Quota-Locks-Max
   )
 )
```

```
objectclass (
  NAME 'SamConfigurationGroupsViewGroup'
  SUBCLASS OF 'SamViewGroup'
  MUST CONTAIN (
         Sam-Group-Total
   )
 )
```

```
objectclass (
  NAME 'SamViewGroup'
  SUBCLASS OF 'SamBase'
  MUST CONTAIN (
         Sam-View-Group-Name
   )
 )
```

```
objectclass (
  NAME 'SamBase'
  SUBCLASS OF 'Mds'
  MAY CONTAIN (
         Sam-Server-Status
   )
 )
```

Figure 4.5 An example of object class definitions used in the monitoring system. These definitions conform to the LDAP standards (RFC 2256). `'SamConfigurationGroups'` and `'SamConfigurationGroupsViewGroup'` are object classes that extend other object class definitions. Each definition contains the attributes which must be always present in the information, along with other optional attributes. More complex structures can be defined using attribute names that are themselves Distinguished Names, to represent the graphical nature of entities in a grid environment.

```
MyType = "Machine"
TargetType = "Job"
Site = "IC"
Schema = "v0_7"
SamStationName = "imperial-test"
SamStationUniverse = "dev"
SamStationExperiment = "d0"
Name = "imperial-test.d0.dev"
Architecture = "Linux+2.4"
Gatekeeper_url = "sampc.hep.ph.ic.ac.uk:2119/jobmanager"
StartIpAddr = "<155.198.211.209:54496>"
LastHeardFrom = 1035560520
```

Figure 4.6 An example of a Machine-Ad used in the monitoring system. The format conforms to the Condor Class-Ad standards. The resource advertises this Ad periodically to the grid. As long as an Ad is alive, the resource is considered available for brokering, match-making and execution purposes. Similar but more complicated Ads are used for identifying Jobs and Schedulers on a grid.

4.2.2 Information Processing Layers of the System

The Monitoring System comprises of different layers that define the processing of information. A pictorial view of these layers is provided in figure 4.7 as:

Layer 1: Information Generation

Layer 2: Information Transformation

Layer 3: Information Integration & Filtering

Layer 4: Information Presentation

All the layers except the 4th have two parallel streams of data flow; these need integration for every desired monitoring *point* in the grid infrastructure. The stream 'A' builds upon the globally-distributed Information Servers, whereas the stream 'B' has the globally-distributed Condor Services as their foundation. Since the data streams originate and thereafter flow across different software realms, integration of all relevant information is performed beyond Layer 3. It is to be noted that the backend-layers Layer 1 and Layer 2 are highly distributed in nature.

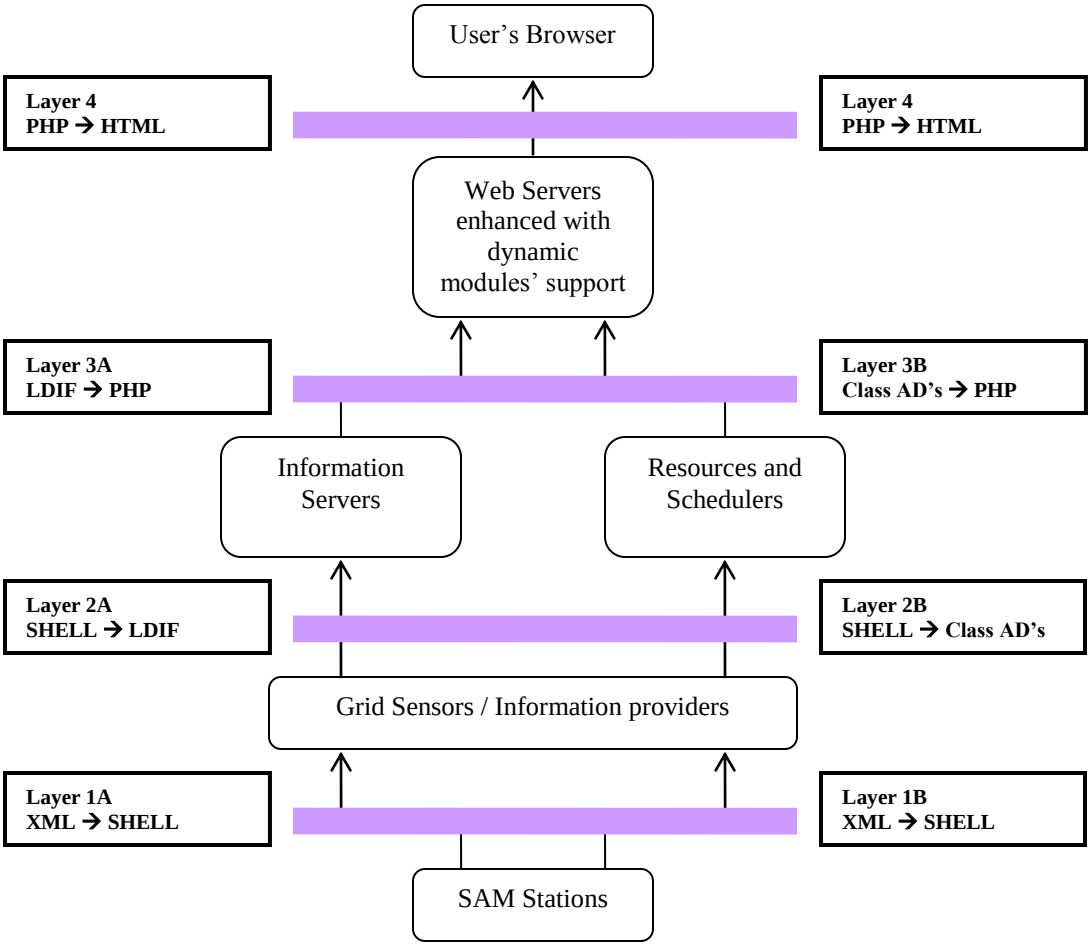

Figure 4.7 The Information Processing Layers defined by the monitoring system. The data stream 'A' and data stream 'B' are built on entirely different realms; hence require information to be integrated across all the desired monitoring junctures on the grid. Layer 1 defines *Information Generation*, Layer 2 defines *Information Transformation*, Layer 3 stresses on *Information Integration & Filtering*, and Layer 4 is the *Information Presentation* layer. The zone beyond Layer 3 serves as an Information Engine, driving the monitoring from the back-ends and upon completion of all processing/integration, finally sending it to a user in a dynamically-rendered 'html' format. The final format is both lightweight and hyperlinked when it reaches a user.

The zone beyond Layer 3 serves as an *information engine*, and plays an important role in extracting data from the lower layers. The information is processed, filtered and integrated in this zone. Thereafter, the monitoring data is prepared to be served to users in a dynamically rendered html format. The main advantage of choosing html to be served at a *user's end* is its being very lightweight, standard-form and widely adopted by most browsers. It also eliminates the need to download plug-ins at the user's end.

Following is a listing of these layers along with a brief description of the configuration features and processing of information that takes place in the monitoring system pertinent to the corresponding layer.

(i) Layer 1 A: At all the distributed monitoring/execution sites, the information is configured in XML (eXtensible Markup Language) at the initialization of the monitoring process. This XML formatted information for each grid-component is then processed and configured into shell scripts to be used by Information Services.

(ii) Layer 1 B: At all the distributed submission sites, the information is configured in XML at the initialization of the monitoring process. This XML formatted information for each grid-component is then processed and configured into shell scripts to be used by Condor and Condor-G Services.

(iii) Layer 2 A: Grid sensors (information providers) utilize the shell scripts, and extract information from the SAM interface and the general state of the site (e.g., the authorized users' details). This information is then delivered to the MDS/LDAP interface that provides with a coherent view of the site in conformance with the LDIF (LDAP Data Interchange Format) standard.

(iv) Layer 2 B: Grid sensors (information providers) utilize the shell scripts, and provide data to Condor interface, that delivers information in form of Class-Ad's. Note that this layer has not been fully implemented for the current prototype system discussed in Chapter 5.

(v) Layers 3 A and B: All the information available to the Layer 3 is extracted from Layer 1 and Layer 2. Loaded with Apache web services, PHP (Hypertext PreProcessor) scripts perform extensive processing of this data.

(vi) Layer 4: After all processing, the information is prepared to be served to a remote user. The user receives a coherent view of the entire grid, and can navigate through the various grid-components using a web interface.

4.2.3 Protocols

As shown in figure 4.2, the Monitoring System works on logically-partitioned and distributed subsets of the grid. These subsets comprise of monitoring sites, submission sites and execution sites.

However, it is true in most practical cases that a distributed environment may have individual components or entities (sub-components) not available to the grid temporarily, due to system or configuration problems; or an entire site may suffer network (or other) failure. Thus, a more realistic representation of the grid would be as shown in figure 4.8, where the grid has such problems manifested in various ways:

a. A monitoring site has few execution sites not available
b. A monitoring site has few execution sites as well as submission sites not available
c. An entire monitoring site has become unreachable from the rest of the grid

Three efficient design features of any Monitoring System would be to:

I. Be robust enough to notice such temporary failures
II. Provide information about which parts of grid have failed
III. Provide the reasons of failure (if possible)

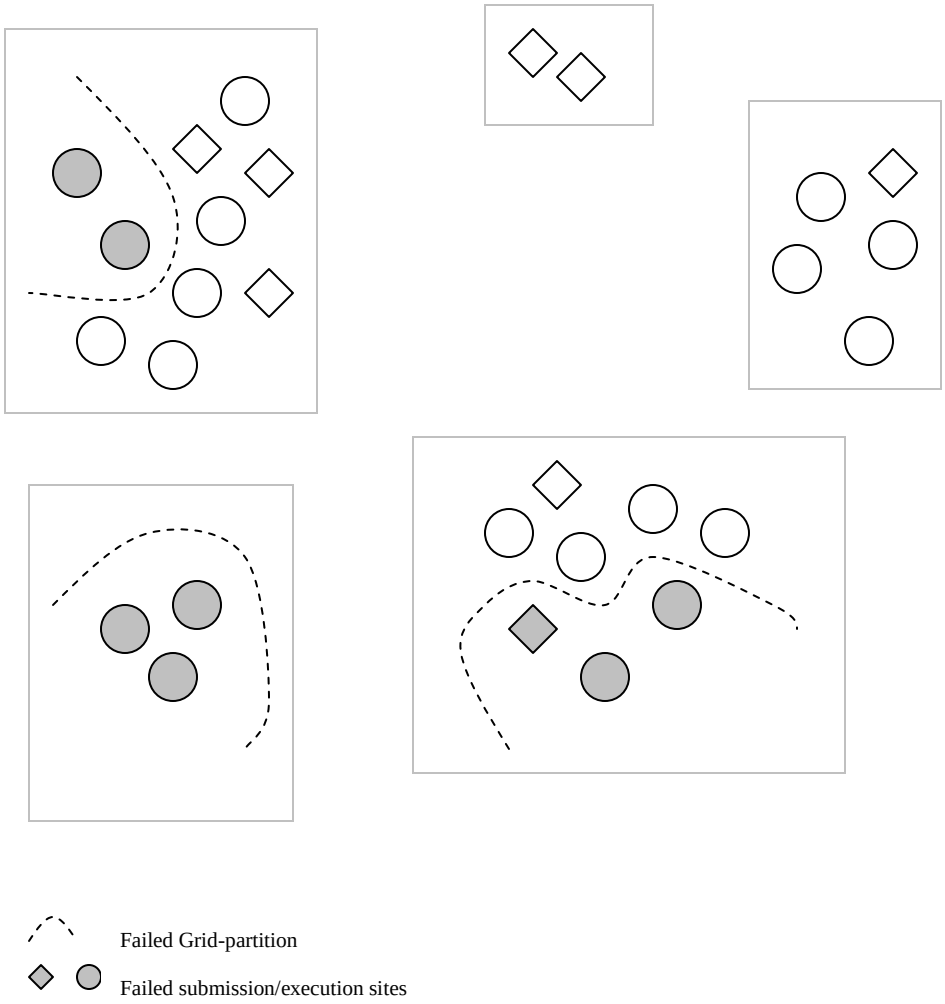

Figure 4.8 A realistic snapshot of the *health* of a grid. This snapshot is in contrast to the hypothetical scenario shown in figure 4.2. At a given time, individual components or entities (sub-components) – or an entire partition of the grid may become unavailable due to various kinds of failure. In this snapshot, a monitoring site has 2 unavailable execution sites, another monitoring site has 2 execution + 1 submission sites unavailable, and yet another has all the execution sites unavailable. In the context of this Monitoring system following a design for DØ/SAM-grid, an execution site can be called unavailable in 2 respects: (a) it is unavailable as a resource to the grid temporarily (b) it is unavailable to the SAM system temporarily.

The design of JIM-IM follows a set of 4 protocols, in order to reflect the real-life states of a grid, and to add robustness to the System.

The system also tries to incorporate all the above mentioned (I, II, and III) features.

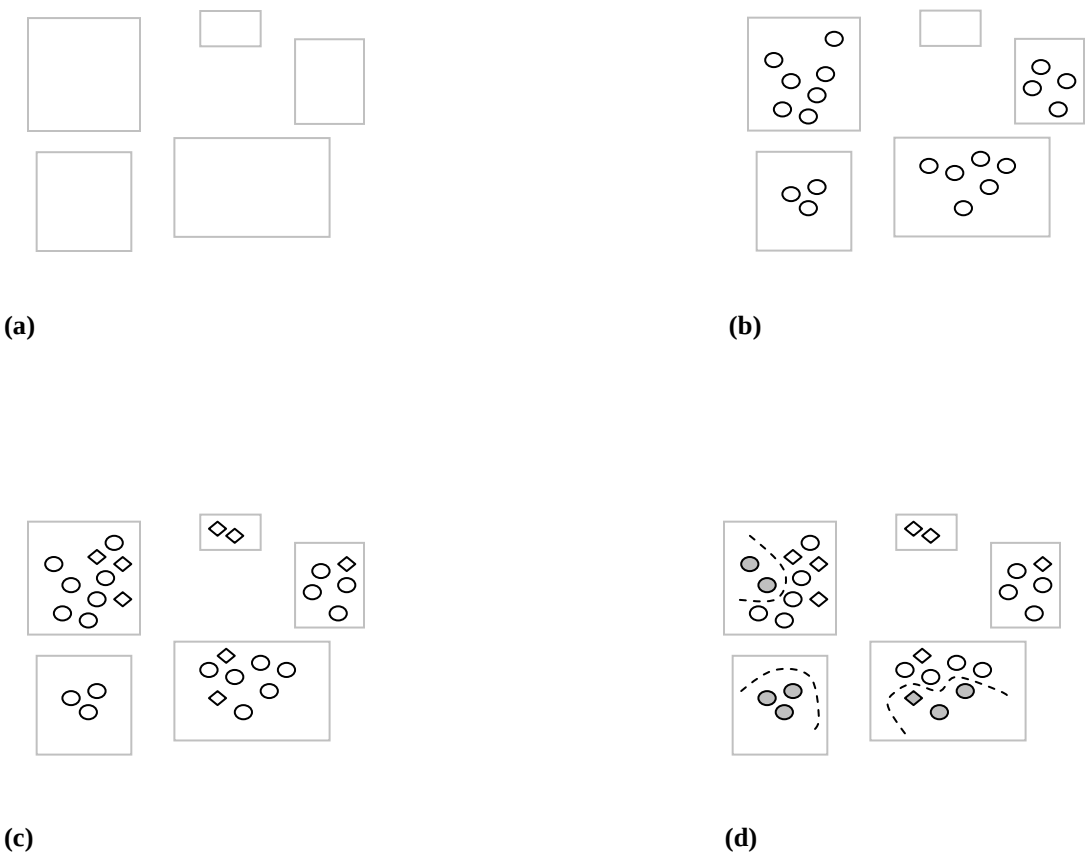

Figure 4.9 The set of four protocols used by the system. These add robustness to the system and enhance its ability to reflect the 'health' of a grid as shown in figure 4.8.
(a) Discovery Protocol I (LDAP)
(b) Discovery Protocol II (CORBA)
(c) Integration Protocol
(d) Enquiry Protocol

The design emphasizes on making a transition from the higher level details through the lower level details of a grid.

As depicted in figure 4.9, the Discovery Protocol I is utilized to find all the monitoring sites. This protocol is essentially LDAP v3. The Discovery Protocol II is responsible for finding all the stations/execution sites on a given monitoring site. This protocol is the built-in CORBA based discovery of SAM. The Integration Protocol gives a unified view of a monitoring site including the submission sites, with execution sites, as well. The Enquiry Protocol extracts all information to define '*a state of the grid*' that is reflective of the real-life snapshot as shown in figure 4.8. The Integration and Enquiry Protocols, combined together, overlap with the zone between Layer 3 and Layer 4 of the Information Processing Layers Model as listed in figure 4.7.

An example of the simplified logic of Integration and Enquiry Protocols combined together with the Discovery Protocols is listed in figure 4.10.

4.3 Implementation

The implementation of the System directly relies on GNU/Linux, Linux Apache/PHP, Shell-Awk scripts; and indirectly relies on C++, Python, XML, Condor, Globus-MDS, and OpenLDAP among other technologies.

4.4 Acknowledgments

The design and modeling of this monitoring system has benefited greatly from intellectual energy and sharp acumen provided by the DØ/SAM-Grid team – notably Igor Terekhov, Andrew Baranovski and Gabriele Garzoglio at FermiLab. The system's *Directory Information Tree* (figure 4.4) was independently designed by Igor Terekhov.

```
get information about all Schedulers;
integrate information;
get information about all Resources;
integrate information;

        for each Scheduler {
                get information about all Jobs;
                for each job {
                        match the pattern with an overlapping Resource;
                        if match successful {
                                retrieve information from the Resource;
                                integrate information with matched Resource;
                                if this integration successful {
                                        integrate information with Exec Site;
                                        discover the Exec Site;
                                        enquire the Exec Site;
                                } else {
                                        try integration later;
                                }
                        } else {
                                retrieve other information for this job;
                                try match later;
                        }
                }
                integrate information;
        }
        integrate information;
```

```
discover all Monitoring Sites;
get all bind details of each Monitoring Site;
get information about all Resources;
integrate information;

        for each Monitoring Site {
                try bind;
                if bind successful {
                        get information about all Exec Sites;
                        for each Exec Site {
                                discover the Exec Site;
                                enquire the Exec Site;
                                retrieve information;
                                integrate information with Resources;
                        }
                        integrate information;
                } else {
                        try bind unless success or all bind details exhausted;
                }
        }
        integrate information;
```

Figure 4.10 Simplified logic of protocols. Two examples showing simplified logic of the Integration & Enquiry Protocols combined together with the Discovery Protocols.

CHAPTER 5

CURRENT UTILIZATION OF THE WORK

A prototypical grid monitoring system has been developed based on the design and architecture discussed in Chapter 4. This chapter lists certain views from the front-end portal with the available monitoring information.

5.1 The current prototype system

A geographical map serves as an anchor to the execution sites as shown in Figure 5.1, there is also a hyperlink to monitor the submission sites on the grid. The monitoring system can be launched to monitor a particular site by clicking on the available hyperlink on the map. The information from the execution sites at a particular monitoring site is retrieved from the information servers deployed at the monitoring site itself. These information servers utilize the DØ/SAM-Grid's grid sensors (information providers) that are also deployed at the monitoring site. These grid sensors are responsible for generating the information required by the monitoring system, and also serve as an interface with the SAM system. The information about the submission sites and the currently available resources on the grid is retrieved from the Condor/Condor-G interfaces. The system attempts at providing the user with monitoring site level monitoring, the grid-job submission level monitoring, and the progress of execution of the job at the execution site – among other important information pertinent to the entire grid.

The execution sites in this prototype are actually SAM-Stations, each of these providing information about its SAM entities like projects, disks, and groups. The grid jobs in this prototype are synonymous with the sam-analysis projects of SAM.

The map shown in figure 5.1 shows 11 monitoring sites in 3 different continents on earth: North America, Europe and Asia, along with a link that is an anchor to all the submission sites. The system does not display the submission sites on the map, since their geographical distribution can not always be determined in advance.

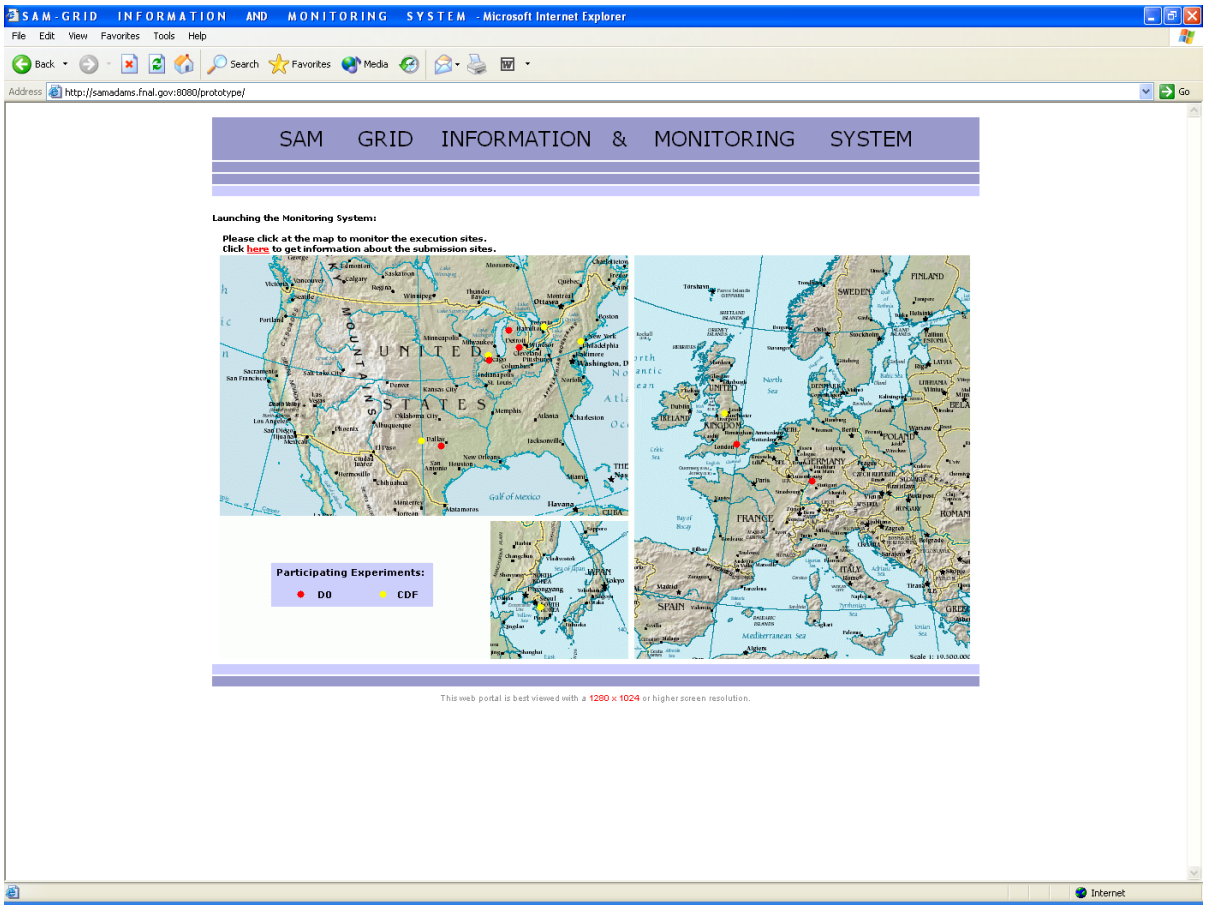

Figure 5.1 The web interface used for launching the monitoring system. A geographical map serves as an anchor to the execution sites at a particular monitoring site. There is also a hyperlink available that lets a user monitor the submission sites on the DØ/SAM-Grid. This map-based interface is an attempt to provide access to the entire grid information in a simple web-portal fashion. The map showed here displays 11 monitoring sites geographically distributed in 5 countries across 3 continents.

The monitoring system, as an attempt to be more informative, makes available a set of color indicators to monitor various entities of an execution site for the monitoring site level monitoring. The figure 5.2 displays these indicators and the corresponding inference that can be drawn by a user.

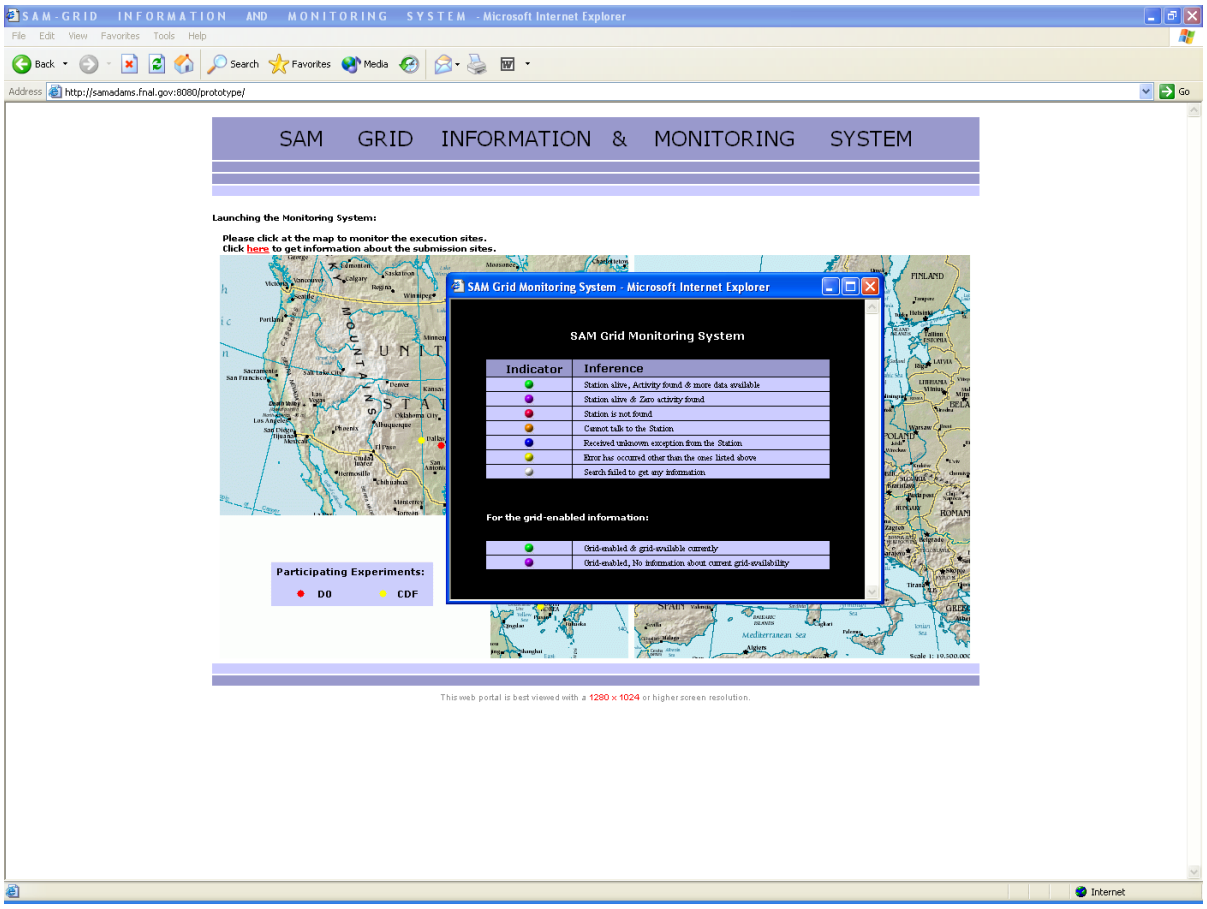

Figure 5.2 The basic status indicators used by this monitoring system. A set of indicators provides status information about a particular entity (projects, disks, groups) of an execution site at a monitoring site. A different set of indicators is available for the status information that is related to a particular execution site's current availability as a resource on the grid.

In figure 5.3, the monitoring system is shown launched for different monitoring sites with general site level status information. The system utilizes a de-centralized architecture. If the information servers and/or grid sensors deployed at a particular monitoring site are unavailable or fail to provide information, performance of the remainder of this monitoring system remains unaffected.

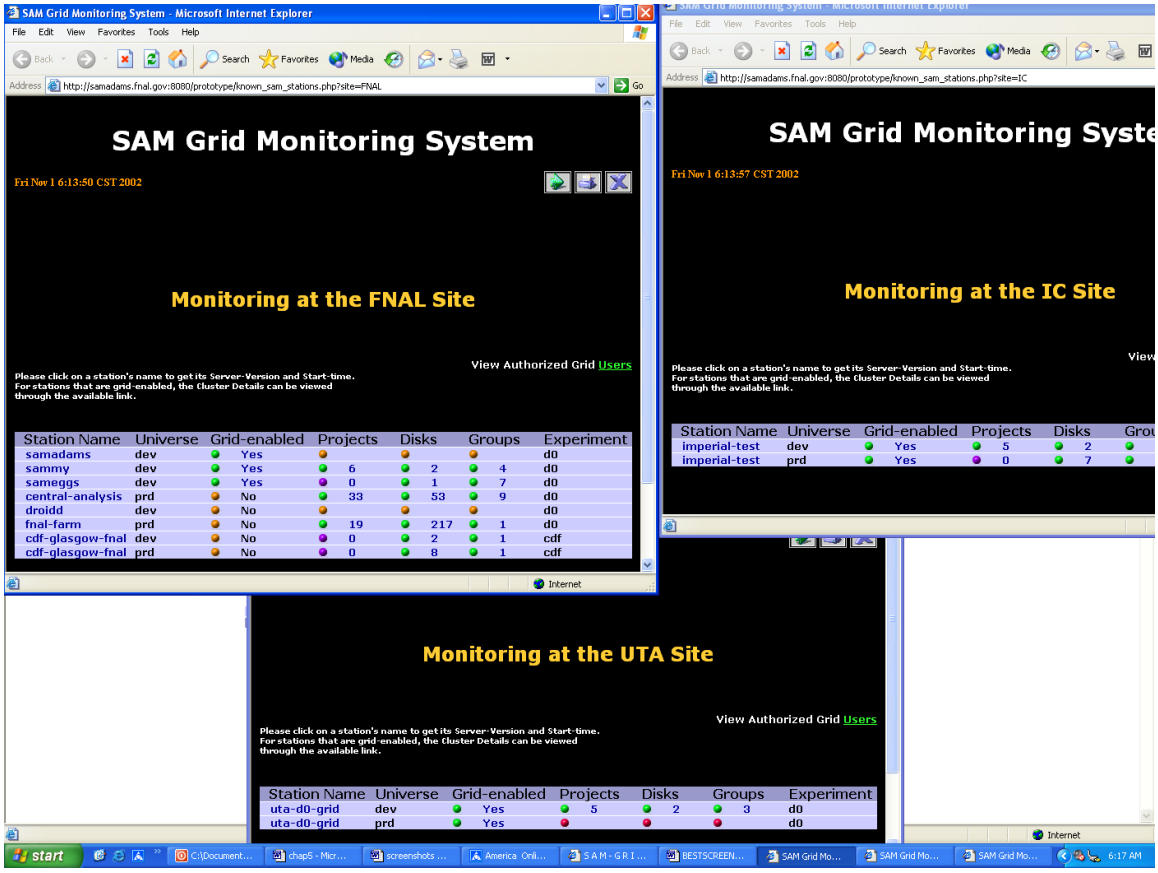

Figure 5.3 Monitoring system shown as launched for 3 different monitoring sites. The information for each monitoring site is retrieved from the information servers deployed locally at that site. The color indicators facilitate in providing an overall view of the grid with status information about the various execution sites.

The monitoring system focusing on a particular monitoring site is shown in Figure 5.4, with the color indicators providing site-level status information. This information is largely provided by the `sam-deployment-summary-reporter` information provider.

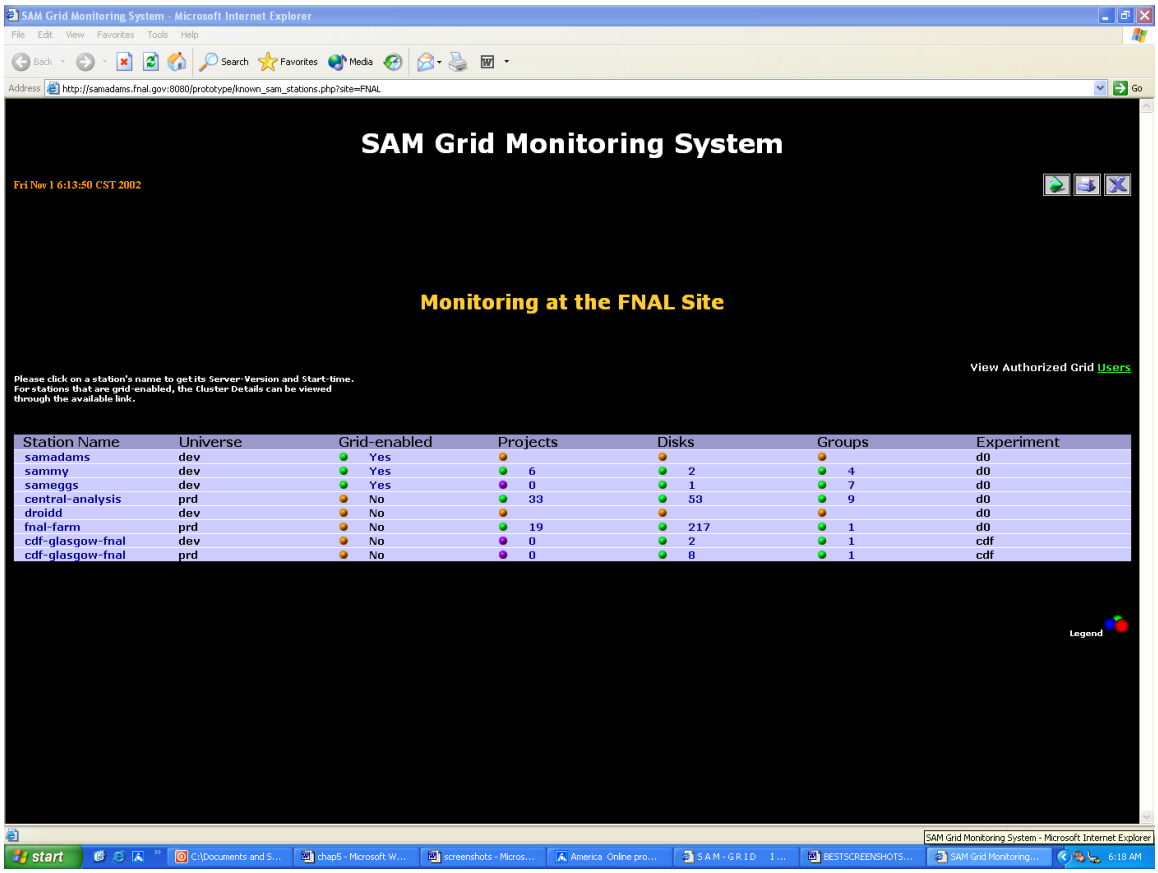

Figure 5.4 A single monitoring site with all its execution sites or SAM-Stations. There is provision to monitor:
(1) Uptime and SAM-software version of a particular Station.
(2) The SAM-universe allotted to a Station, this can be production (prd) or development (dev).
(3) The current grid-availabilty and related grid-specific details of a Station as a resource on the grid.
(4) The current projects on a Station.
(5) All disks available to a Station.
(6) The groups that have access to this Station.
(7) Physics experiment making use of this Station.
(8) The authorized grid-users for this particular monitoring site.

A description of the color indicators used for site level monitoring is given in figure 5.5, these also reflect that a failed search by the system does not interfere with the other searches.

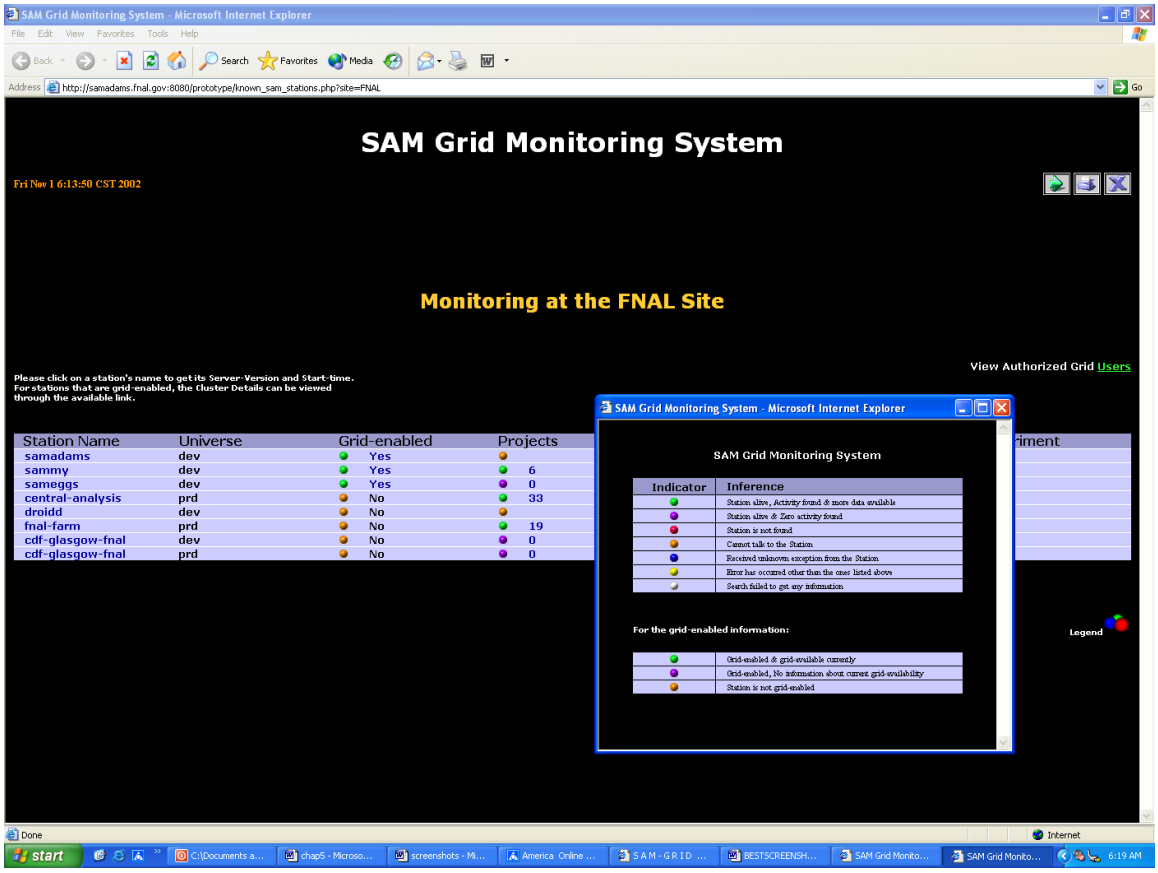

Figure 5.5 The legend displaying all the status indicators. Also shown are corresponding inferences that can be drawn, for site-level monitoring. The indicators provide one of the following status information:
(1) Station is alive, activity found, and more data available (hyperlink can be navigated further).
(2) Station is alive, but no activity found.
(3) Station is not found by the CORBA naming service.
(4) A communication error occurred while trying to reach Station by the CORBA naming service.
(5) The monitoring system received unknown exception from the Station.
(6) An error occurred that is unknown to the monitoring system.
(7) Search for information failed, and no information retrieved.
(8) Station is on the DØ/SAM-Grid and grid-available currently (hyperlink can be navigated further).
(9) Station is on the DØ/SAM-Grid but no information about its *current* grid-availability.
(10) Station is not on the DØ/SAM-Grid.

The system is shown in figure 5.6 with information about SAM-software version, uptime and grid-specific details for a particular Station. The information about the uptime data is provided by the `sam-deployment-reporter` information provider.

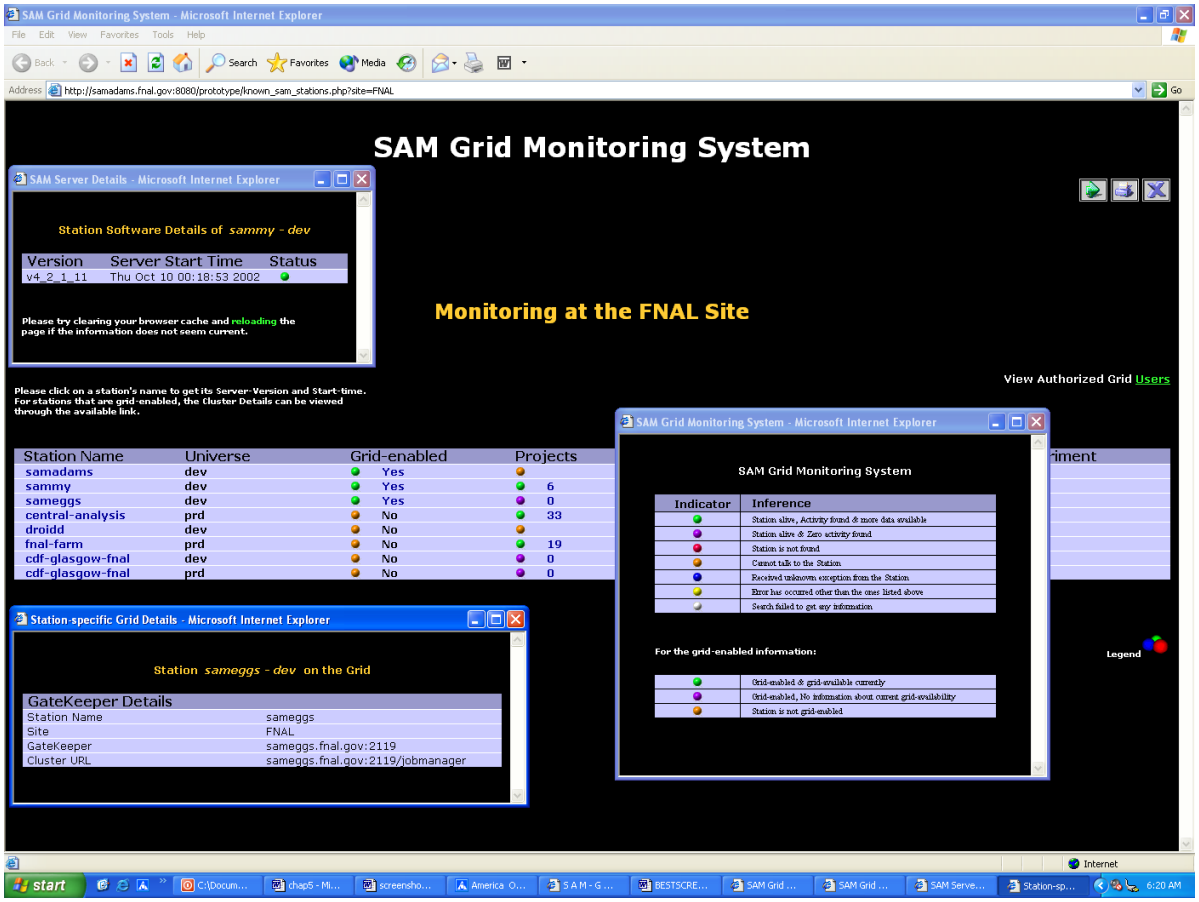

Figure 5.6 Monitoring the version, start time, and the status of a Station. The monitoring system with information about the SAM-software version, start time, and the status indicator – for a particular Station. If this information is not retrieved, the color of the indicator indicates the particular error that occurred. Also, the grid-details for a Station on the DØ/SAM-Grid are shown, that provide the gatekeeper and jobmanager URLs used by the grid to communicate with this Station.

The monitoring system with projects running on a particular Station, is shown in figure 5.7. The `sam-projects-reporter` information provider is used for this purpose. Information about the general status of each project is shown as provided by the SAM interface.

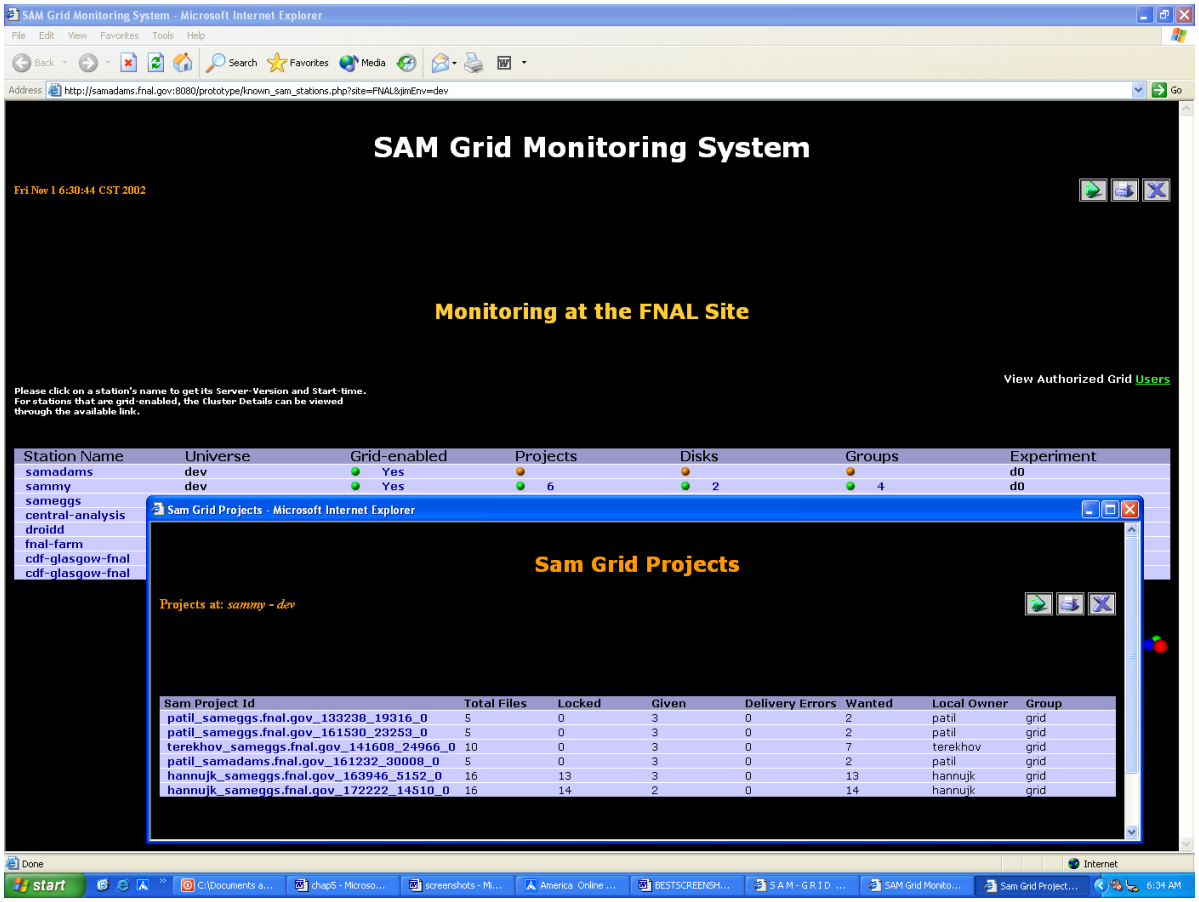

Figure 5.7 Monitoring the general status of projects at a Station. Information is provided by the SAM interface about the total files required, locked files, given files, delivery errors (if any), wanted files, the local owner of the project with the particular group, this project is associated with. Note that hyperlinks are present to further monitor the progress details of each project, using the corresponding SAM-project-master.

The details for a specific project can be monitored as shown in figure 5.8, with information about all the associated processes. The `sam-project-detail-reporter` information provider is used for this purpose. Information about the details of each project is shown as provided by the SAM interface.

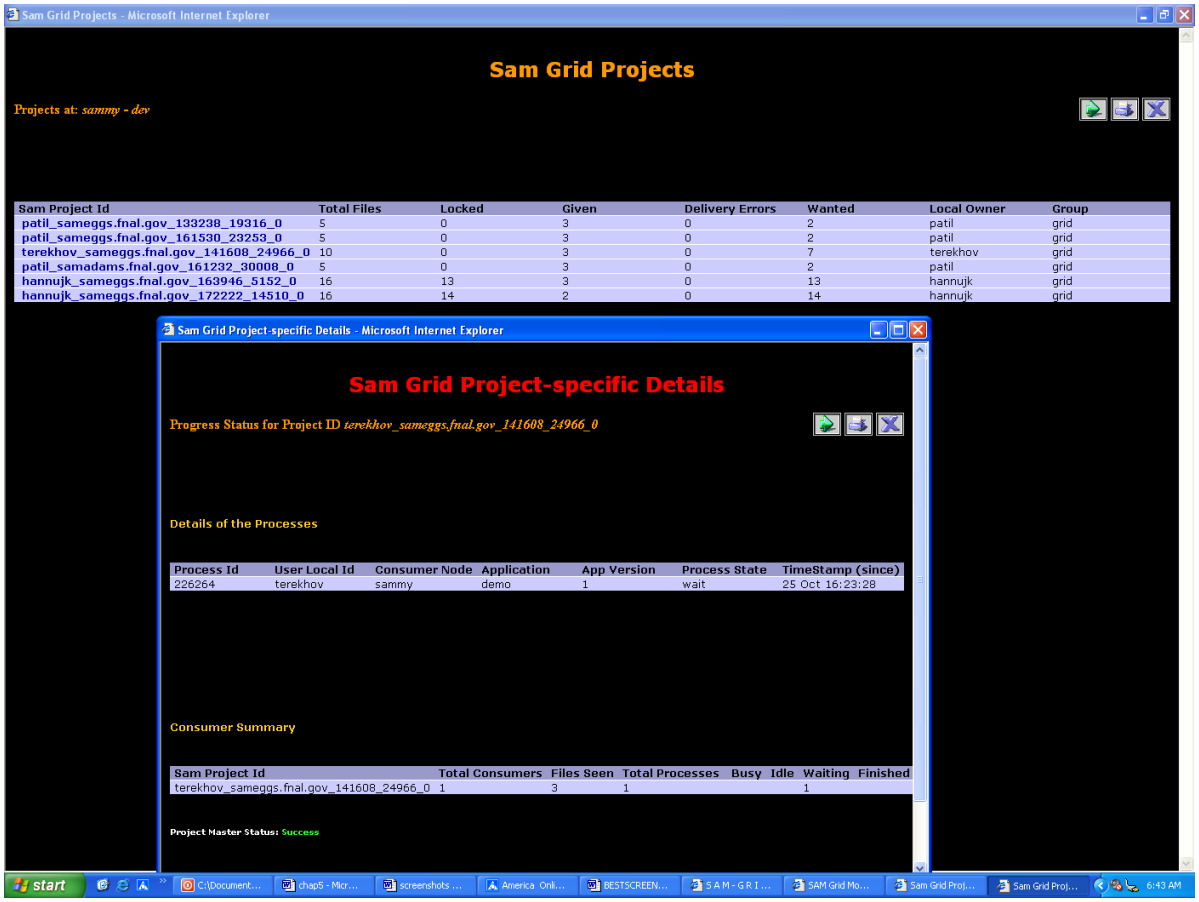

Figure 5.8 Monitoring the progress status details for a specific project. The details include information about all the associated processes of a project, and a summary for the consumer. Each process has its state (busy, idle, waiting, or finished) as provided by the SAM-interface.

The monitoring system with disks available to a particular Station is shown in figure 5.9. The `sam-config-disks-reporter` information provider is used for this purpose. A disks summary for this Station presents the total number of disks, and the inactive disks, with the total amount of available and free disk space in megabytes. Specific information is present for each disk regarding the disk location, disk space, and its status.

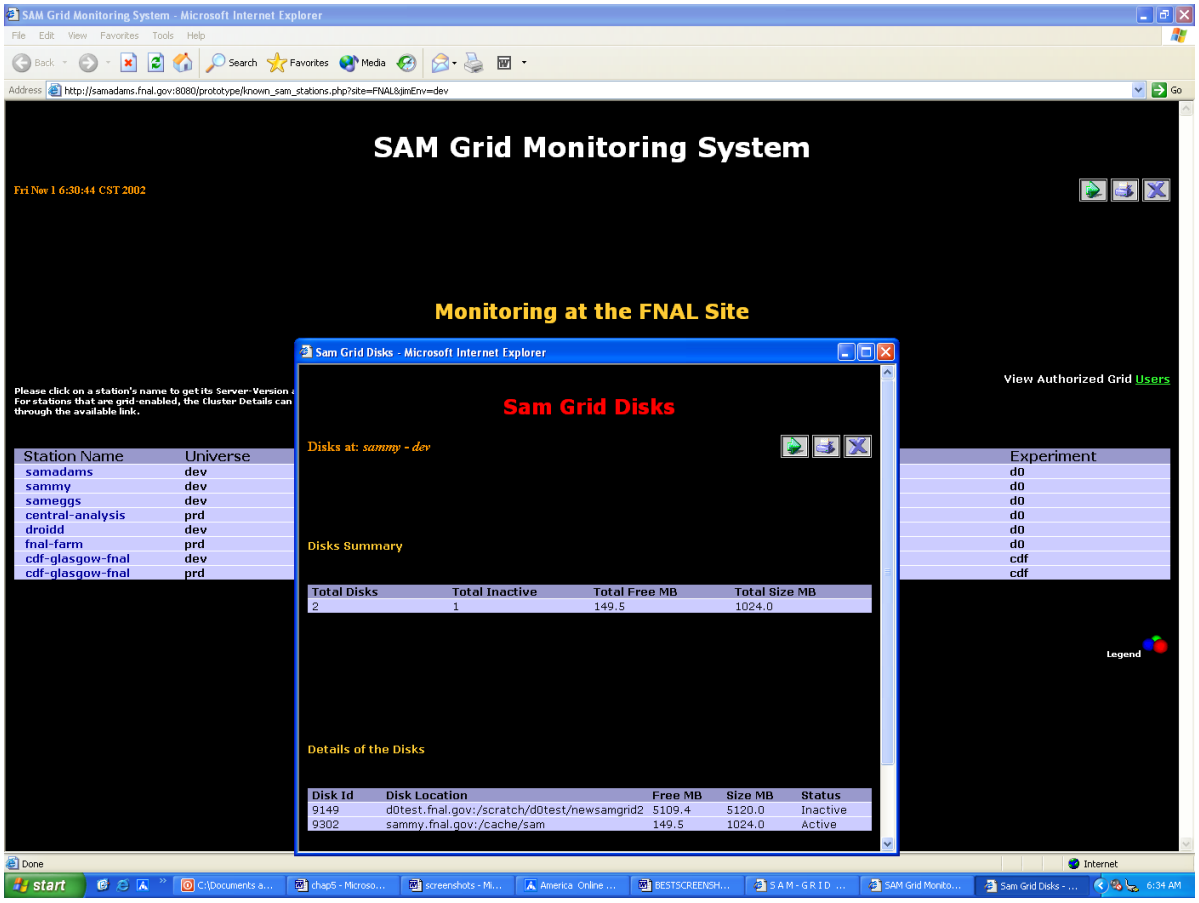

Figure 5.9 Monitoring the disks of a Station. Also shown is a summary for *all* the disks of the Station. The summary shoes the aggregated information about total disks, total inactive disks, total size and total free space in megabytes. Moreover, information is present for each disk on the Station. This information is extracted from the SAM-interface by an information provider.

The monitoring system with all the physics groups associated with a particular Station is shown in figure 5.10. The `sam-config-groups-reporter` information provider is used for this purpose.

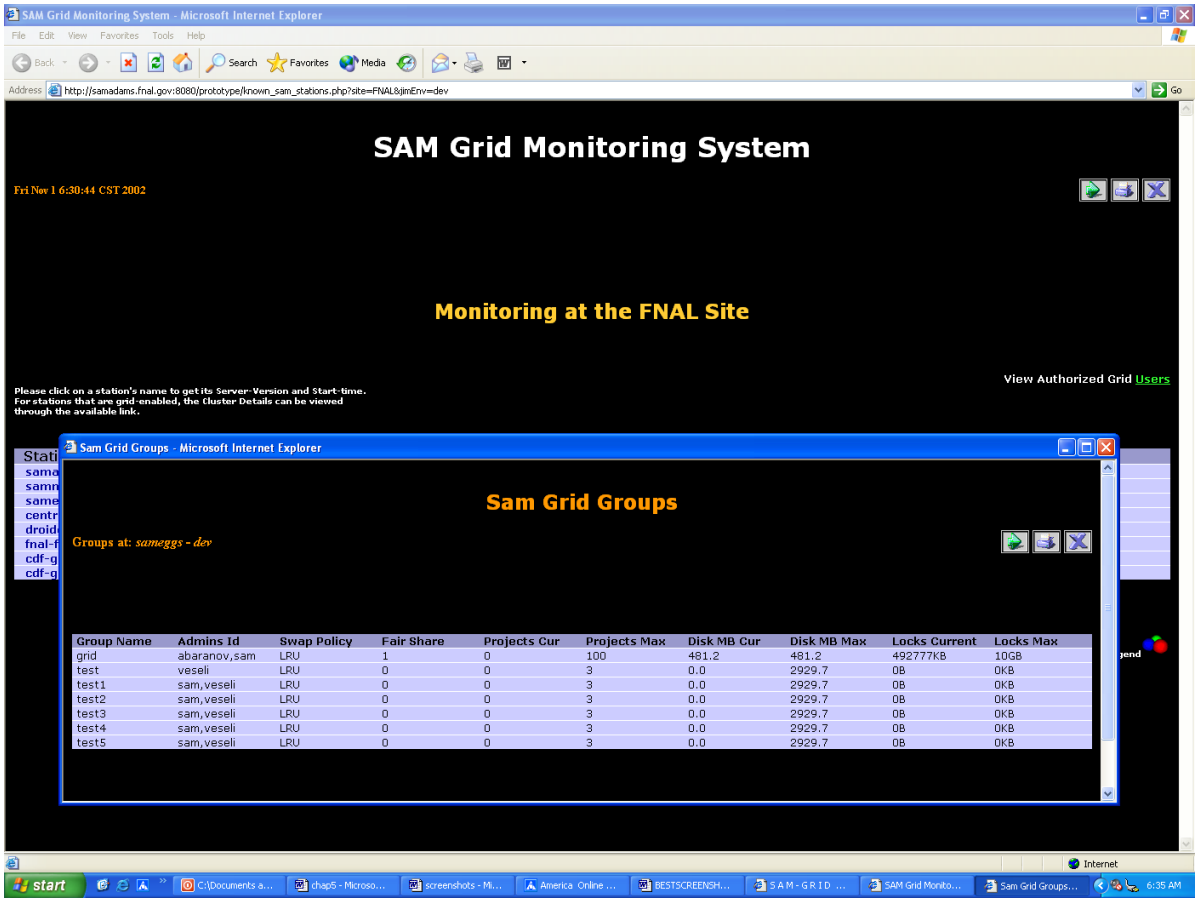

Figure 5.10 Monitoring the groups associated with a Station. Information is extracted from the SAM-interface by an information provider. Information is related to a particular group's administrators, the policies followed by the group, various quotas (projects, disks, locks) with the maximum and current values – for a Station.

The grid-authorized users for a monitoring site are shown in figure 5.11. The authorized users are listed in the grid-mapfile (a text file used as a standard repository of information, by Globus architecture, pertaining to the grid-users certified by a recognized certificate authority). An information provider sam-`authorized-users-reporter` is used to provide the monitoring system with this information. For known certificate authorities, the information provider also attempts to classify a user as *person* or *service* on the grid.

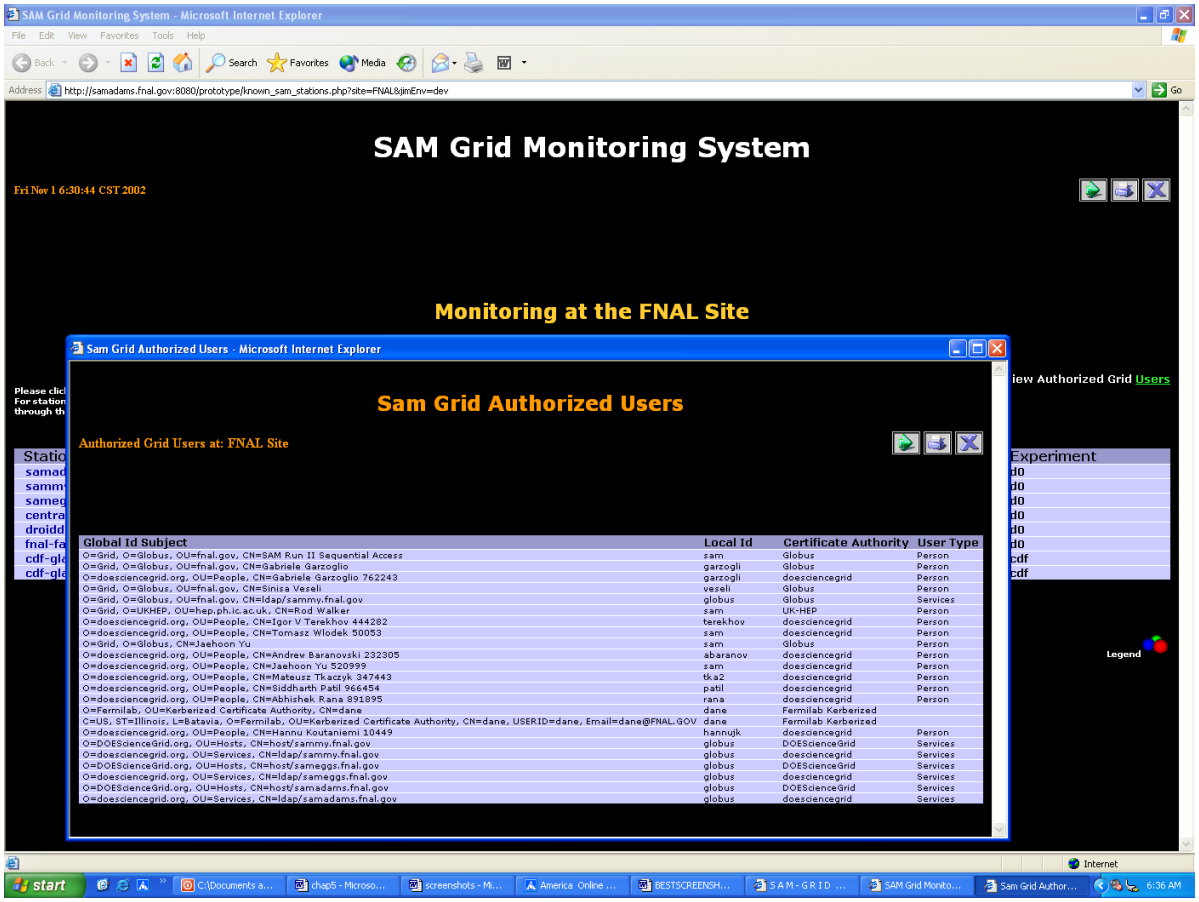

Figure 5.11 Information about the authorized users at a monitoring site of the grid. Such users have obtained a certificate from a recognized certificate authority with a global-id that uniquely identifies them on the grid. Information on the known certificate authorities is also provided. A user can be a human or a grid-service, the system attempts to classify the users for the known certificate authorities.

The monitoring system when launched from the anchor link on the web portal, to monitor the submission sites on the grid is shown in figure 5.12. The information can be used when a user wishes to submit a job to the DØ/SAM-Grid. A submission site is always known to the user, however the execution is site is unknown during job submission if the user relies on JIM's brokering services. Hence, the system attempts to facilitate tracking the progress of the submitted job even subsequent to the matching of job with a remote resource on the grid.

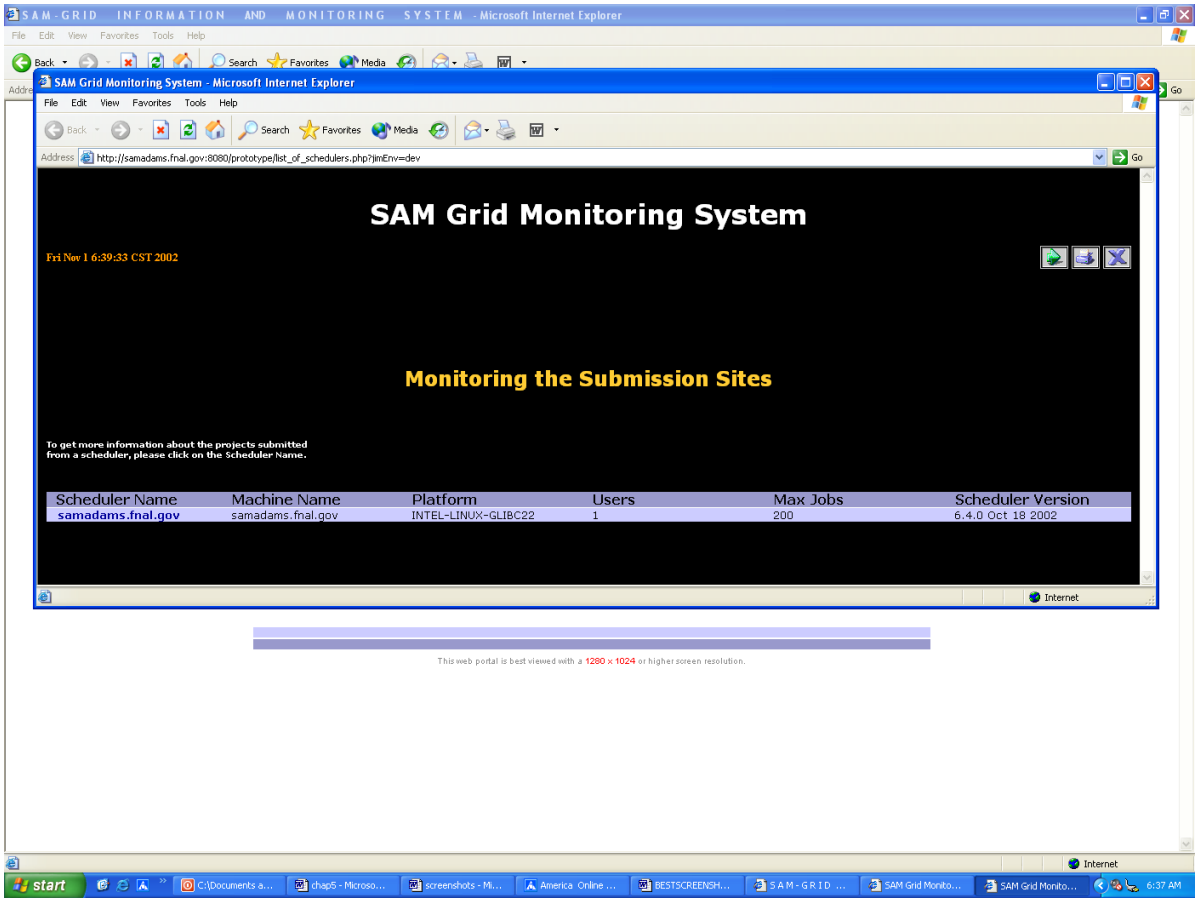

Figure 5.12 The system as launched to monitor the submission sites on the grid. The submission sites are also known as grid-clients, and utilize JIM's client software packages. This information is retrieved by the system from Condor Class-Ads.

Figure 5.13 describes the procedure a user can follow to monitor the progress of a job subsequent to its submission *to the grid* from a known submission site.

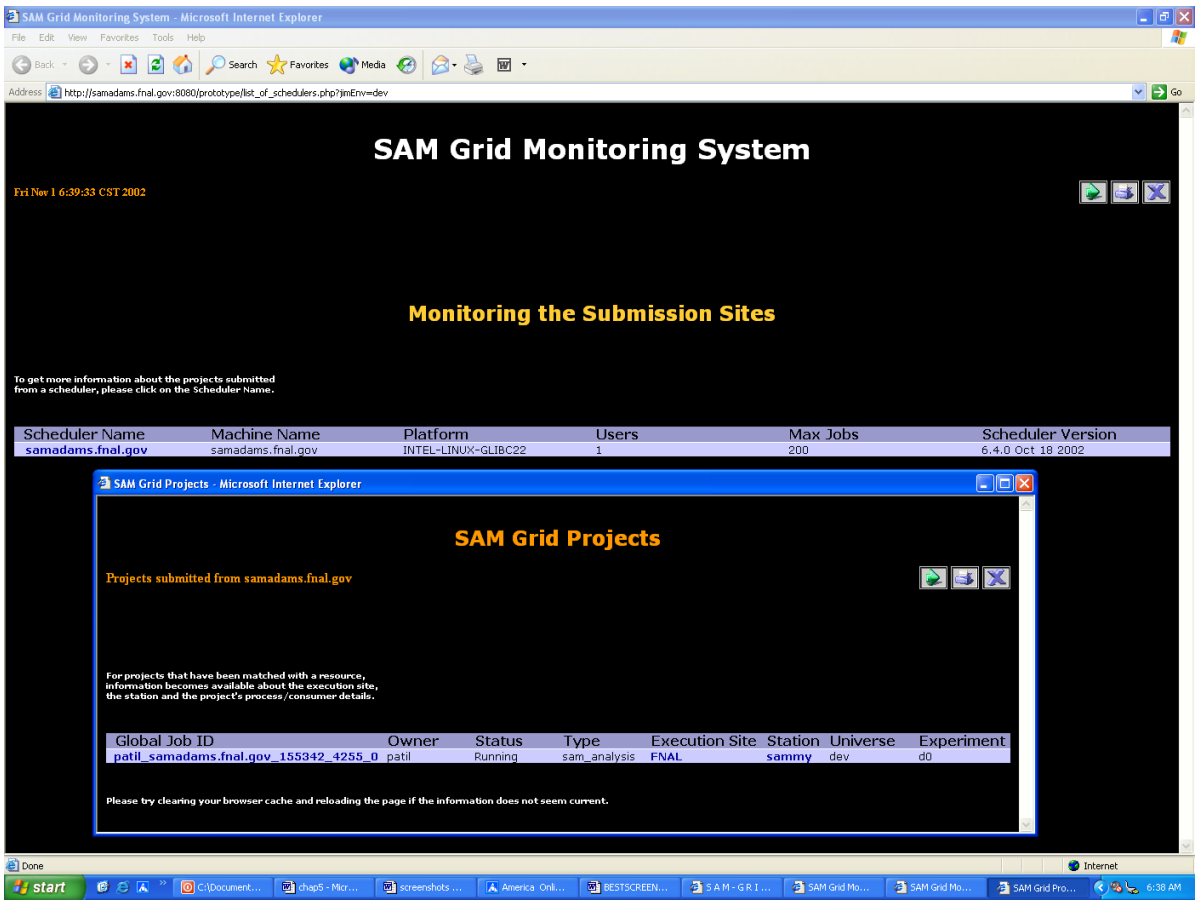

Figure 5.13 Monitoring a known submission site. This information is retrieved using the available hyperlink shown in figure 5.12. Information is available about all the jobs that have been submitted to the grid using the scheduler at this submission site. Note that immediately after a job's submission and prior to its being matched and routed to a remote resource on the grid, the monitoring system provides the user with the job's unique id, the owner's local id, and the status if available. However, at this stage, hyperlinks do not appear. Subsequent to the match, hyperlinks appear and information becomes available about the remote execution site (Station), the corresponding monitoring site, and the project's progress details as retrieved from its SAM-project-master. If the job (sam-analysis project) has not started execution at the remote station yet, although the hyperlinks are present, the information retrieved from clicking on the Job id displays the project-master status to be 'cannot talk to Pmaster' as shown in figure 5.14. As soon as the execution commences, the project-master status shows 'success' if no SAM-error occurs, and user can monitor detailed progress details of all the associated processes and consumers.

Monitoring the progress of a submitted job is shown in figure 5.14. A user monitors the progress of the related monitoring site, or directly the execution site (Station). Also, a user can use the system to monitor the detailed progress of the associated individual processes in a manner similar to the one shown in figure 5.8.

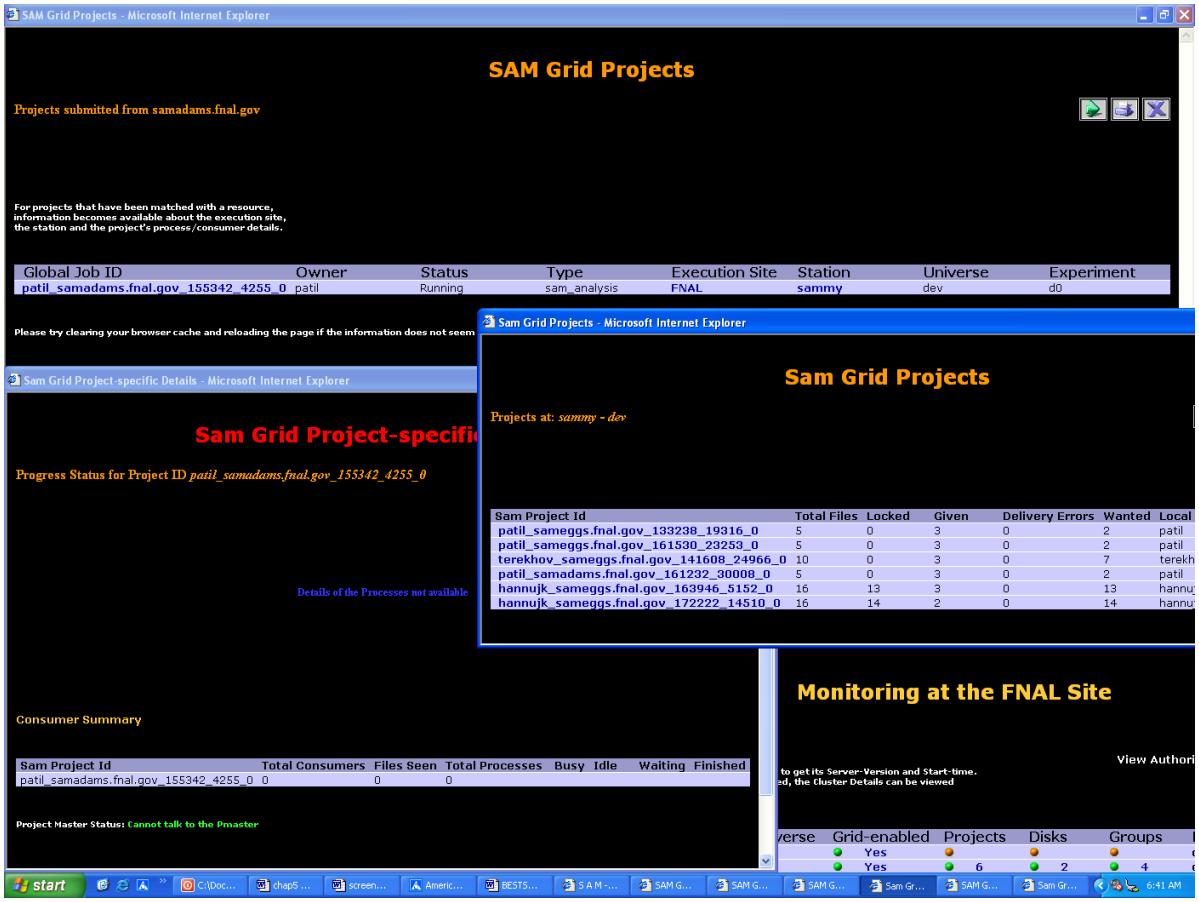

Figure 5.14 Monitoring the status of a submitted job at a remote execution site. Shown is the status of a job (sam-analysis project) submitted from a known scheduler after it has been matched with an available resource on the grid. From this stage, the system provides options to monitor the execution site directly, or the related monitoring site, or the detailed progress status directly form the job's project-master. The system also attempts to provide information about the general status of the job, as known to the scheduler. This status can be either: Idle, Running, Removed, Completed, or Held.

The following figures show the utilization of the monitoring system to get information about stations like *central-analysis* and *fnal-farm* in production environment (called 'prd' or 'production' universe) that are being used by the scientific community.

The figure 5.15 displays information about real-life sam-analysis projects running on *central-analysis* Station.

Sam Grid Projects - Microsoft Internet Explorer

Sam Grid Projects

Projects at: *central-analysis - prd*

Sam Project Id	Total Files	Locked	Given	Delivery Errors	Wanted	Local Owner	Group
filthaut_20021024042947	30	25	2	0	28	filthaut	dzero
malexeev_20021025091123	1123	641	0	0	1123	malexeev	dzero
malexeev_20021025113736	333	325	0	0	333	malexeev	dzero
malexeev_20021025113738	144	21	0	0	144	malexeev	dzero
malexeev_20021025113912	302	25	0	0	302	malexeev	dzero
malexeev_20021025114209	2	2	0	0	2	malexeev	dzero
malexeev_20021025114806	1	1	0	0	1	malexeev	dzero
duflot_20021028050002	358	28	0	0	358	duflot	dzero
omk_p11.11.00_210_28_02_18_06_52	203	9	0	0	203	kuznets	dzero
kuznets_20021029102125	184	183	0	0	184	kuznets	dzero
mikeh_20021029154959	22	22	0	0	22	mikeh	dzero
fblekman_20021030071515	116	105	0	0	116	fblekman	dzero
fblekman_20021030072931	854	763	0	0	854	fblekman	dzero
fblekman_20021030073644	612	589	0	0	612	fblekman	dzero
fblekman_20021030082643	633	577	0	0	633	fblekman	dzero
kuznets_20021030091755	1194	1192	0	0	1194	kuznets	dzero
kuznets_20021030093453	833	833	0	0	833	kuznets	dzero
fblekman_20021030090919	562	514	0	0	562	fblekman	dzero
zdrazil_20021030125535	792	156	0	0	792	zdrazil	dzero
zdrazil_20021030130717	647	11	0	0	647	zdrazil	dzero
zdrazil_20021030133616	632	29	0	0	632	zdrazil	dzero
zdrazil_20021030142703	253	23	0	0	253	zdrazil	dzero
zdrazil_20021030144139	0	0	0	0	0	zdrazil	dzero
chunhuih_20021030145822	565	555	8	0	557	chunhuih	algo
zdrazil_20021030161213	10	10	0	0	10	zdrazil	dzero
azabi_20021031100147	41	28	0	0	41	azabi	dzero
bandurin_20021031120329	62	0	33	0	29	bandurin	dzero
danielw_20021031131127	0	0	0	0	0	danielw	dzero
bandurin_20021031173031	781	0	32	1	748	bandurin	dzero
mikeh_20021031213905	59	9	6	0	53	mikeh	dzero
duflot_20021101035331	1820	1121	0	0	1820	duflot	dzero
duflot_20021101041623	1818	1101	18	0	1800	duflot	dzero
lumi1113-01Nov2002-05:52:58	5	0	5	0	0	mverzocc	dzero
mverzocc_20021101062234	29	0	24	0	5	mverzocc	dzero
fblekman_20021101062312	903	812	35	0	868	fblekman	dzero

Figure 5.15 Monitoring real-life *production* SAM-projects. An example of monitoring real-life SAM-projects on the Station *central-analysis (production universe)*.

Figure 5.16 shows the use of monitoring system to monitor numerous disks associated with *central-analysis* Station in *production universe*.

Sam Grid Disks - Microsoft Internet Explorer

Sam Grid Disks

Disks at: *central-analysis - prd*

Disks Summary

Total Disks	Total Inactive	Total Free MB	Total Size MB
53	1	190091.9	13492606.9

Details of the Disks

Disk Id	Disk Location	Free MB	Size MB	Status
22	d0mino.fnal.gov:/sam/cache4	2.3	276480.0	Active
23	d0mino.fnal.gov:/sam/cache5	6.3	276480.0	Active
24	d0mino.fnal.gov:/sam/cache6	20.9	276480.0	Active
25	d0mino.fnal.gov:/sam/cache7	11.3	276480.0	Active
26	d0mino.fnal.gov:/sam/cache8	4.5	276480.0	Active
27	d0mino.fnal.gov:/sam/cache9	3.1	276480.0	Active
28	d0mino.fnal.gov:/sam/cache10	13.2	276480.0	Active
29	d0mino.fnal.gov:/sam/cache11	7.5	276480.0	Active
30	d0mino.fnal.gov:/sam/cache12	2.2	276480.0	Active
31	d0mino.fnal.gov:/sam/cache13	3.3	276480.0	Active
32	d0mino.fnal.gov:/sam/cache14	1.9	276480.0	Active
164	d0mino.fnal.gov:/sam/cache15	0.9	99609.4	Active
165	d0mino.fnal.gov:/sam/cache22	0.3	99609.4	Active
166	d0mino.fnal.gov:/sam/cache23	0.1	99609.4	Active
167	d0mino.fnal.gov:/sam/cache24	5.2	99609.4	Active
168	d0mino.fnal.gov:/sam/cache25	0.8	99609.4	Active
182	d0mino.fnal.gov:/sam/cache26	1.7	276480.0	Active
183	d0mino.fnal.gov:/sam/cache27	0.2	276480.0	Active
184	d0mino.fnal.gov:/sam/cache28	0.5	276480.0	Active
185	d0mino.fnal.gov:/sam/cache29	3.0	276480.0	Active
186	d0mino.fnal.gov:/sam/cache30	6.2	276480.0	Active
187	d0mino.fnal.gov:/sam/cache31	1.0	276480.0	Active
188	d0mino.fnal.gov:/sam/cache32	1.9	276480.0	Active
189	d0mino.fnal.gov:/sam/cache33	4.4	276480.0	Active
190	d0mino.fnal.gov:/sam/cache34	0.5	276480.0	Active
191	d0mino.fnal.gov:/sam/cache35	4.5	276480.0	Active
192	d0mino.fnal.gov:/sam/cache36	0.9	276480.0	Active
193	d0mino.fnal.gov:/sam/cache37	2.8	276480.0	Active

Figure 5.16 Monitoring *production* SAM-disks. An example of monitoring SAM-disks available to the Station *central-analysis (production universe)*.

Figure 5.17 (a) displays the monitoring system with information about numerous processes of a real-life project running on *fnal-farm* Station in *production universe*. Each process has its state displayed with the latest timestamp since the instant this process witnessed activity.

Process Id	User Local Id	Consumer Node	Application	App Version	Process State	TimeStamp (since)
989159	d0farm	fnd0109	recon_root	x13.01.00	idle	31 Oct 23:10:36
989161	d0farm	fnd0214	recon_root	x13.01.00	idle	31 Oct 22:39:13
989169	d0farm	fnd0104	recon_root	x13.01.00	idle	31 Oct 22:35:54
989172	d0farm	fnd0215	recon_root	x13.01.00	idle	01 Nov 02:22:47
989173	d0farm	fnd0106	recon_root	x13.01.00	busy	31 Oct 20:04:27
989175	d0farm	fnd0121	recon_root	x13.01.00	busy	31 Oct 11:46:38
989176	d0farm	fnd0127	recon_root	x13.01.00	idle	01 Nov 03:08:50
989179	d0farm	fnd0191	recon_root	x13.01.00	busy	31 Oct 11:46:37
989187	d0farm	fnd0216	recon_root	x13.01.00	idle	01 Nov 00:45:20
989188	d0farm	fnd0118	recon_root	x13.01.00	busy	31 Oct 19:53:39
989193	d0farm	fnd0119	recon_root	x13.01.00	busy	31 Oct 11:49:14
989199	d0farm	fnd0108	recon_root	x13.01.00	busy	31 Oct 11:50:32
989208	d0farm	fnd0210	recon_root	x13.01.00	idle	01 Nov 01:17:24
989209	d0farm	fnd0130	recon_root	x13.01.00	busy	31 Oct 11:51:47
989211	d0farm	fnd0122	recon_root	x13.01.00	idle	01 Nov 04:50:24
989219	d0farm	fnd0198	recon_root	x13.01.00	busy	31 Oct 19:52:25
989221	d0farm	fnd0107	recon_root	x13.01.00	idle	01 Nov 00:28:20
989224	d0farm	fnd0131	recon_root	x13.01.00	busy	31 Oct 11:54:57
989225	d0farm	fnd0158	recon_root	x13.01.00	idle	31 Oct 21:06:32
989226	d0farm	fnd0199	recon_root	x13.01.00	idle	31 Oct 20:12:31
989228	d0farm	fnd0115	recon_root	x13.01.00	busy	31 Oct 11:58:34
989229	d0farm	fnd0184	recon_root	x13.01.00	idle	01 Nov 03:21:03
989230	d0farm	fnd0159	recon_root	x13.01.00	idle	31 Oct 23:27:25
989236	d0farm	fnd0112	recon_root	x13.01.00	busy	31 Oct 11:59:51
989237	d0farm	fnd0196	recon_root	x13.01.00	idle	01 Nov 00:13:40
989239	d0farm	fnd0208	recon_root	x13.01.00	idle	31 Oct 23:18:47
989240	d0farm	fnd0113	recon_root	x13.01.00	idle	01 Nov 02:18:26
989241	d0farm	fnd0197	recon_root	x13.01.00	busy	31 Oct 19:34:16
989244	d0farm	fnd0110	recon_root	x13.01.00	idle	01 Nov 03:14:07
989246	d0farm	fnd0202	recon_root	x13.01.00	idle	31 Oct 21:28:33
989247	d0farm	fnd0203	recon_root	x13.01.00	idle	31 Oct 21:19:41
989249	d0farm	fnd0194	recon_root	x13.01.00	idle	01 Nov 02:12:44
989250	d0farm	fnd0200	recon_root	x13.01.00	finished	31 Oct 20:04:27
989251	d0farm	fnd0195	recon_root	x13.01.00	idle	01 Nov 02:28:27
989253	d0farm	fnd0123	recon_root	x13.01.00	idle	31 Oct 21:07:14
989254	d0farm	fnd0111	recon_root	x13.01.00	busy	31 Oct 11:59:35
989255	d0farm	fnd0192	recon_root	x13.01.00	idle	31 Oct 20:39:36
989256	d0farm	fnd0201	recon_root	x13.01.00	idle	31 Oct 23:36:58
989258	d0farm	fnd0193	recon_root	x13.01.00	busy	31 Oct 17:37:24
989259	d0farm	fnd0114	recon_root	x13.01.00	idle	01 Nov 00:07:15
989260	d0farm	fnd0190	recon_root	x13.01.00	idle	31 Oct 21:30:56
989261	d0farm	fnd0206	recon_root	x13.01.00	busy	31 Oct 18:14:14

Figure 5.17(a) Progress details of processes of a *production* SAM-project. An example of monitoring detailed progress details of a SAM-project running on the Station *fnal-farm (production universe)*.

Figure 5.17 (b) shows the remaining part of the system's view in figure 5.17 (a) with the consumer summary having information about the count of all the processes, number of processes in each of the states: busy, idle, waiting, or finished. The project-master status is also shown, after the system has retrieved information from it.

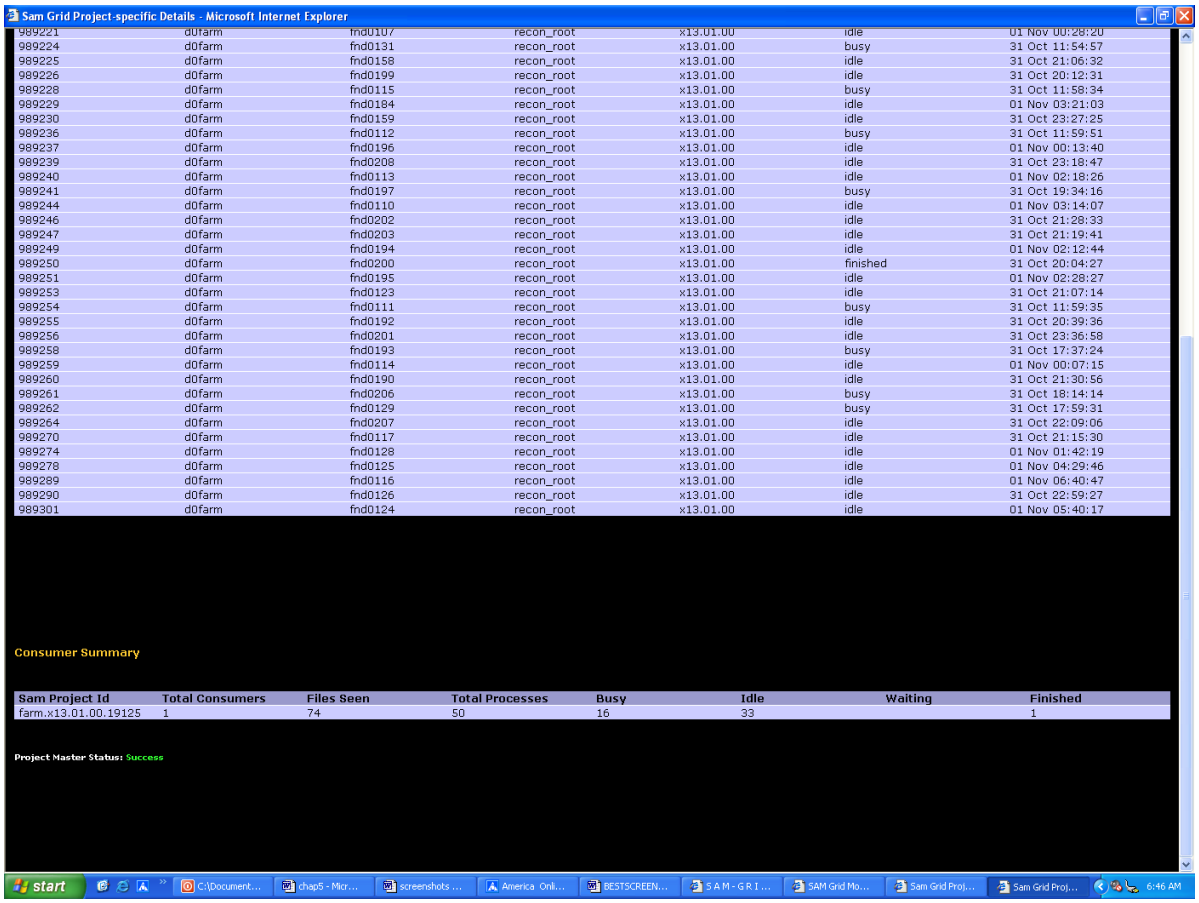

Figure 5.17(b) Progress summary of a *production* SAM-project. An example of monitoring detailed progress details of a SAM-project running on the Station *fnal-farm (production universe)*.

5.2 Salient novel features of the prototype

The prototype does not rely on any existing monitoring system or any existing LDAP browser, since:

1. The available browsers usually display *all* the information retrieved from Globus MDS, without much filtering of the irrelevant data. In contrast, this prototype displays only the pertinent information while hiding the unimportant details.
2. The available browsers display only LDIF data from the LDAP backend. In contrast, the prototype integrates information across various grid middleware domains, and is not restricted to information retrieval from a single source like MDS or LDAP.
3. The prototype transparently provides monitoring information from the known submission site to the unknown (at the job submission stage) execution site. This feature is not available in most available browsers.

The prototype does not depend on MDS's built-in schema since the SAM system needs monitoring based on Stations and not machines, the latter is assumed to be the default resource on the grid by MDS.

The prototype does not utilize MDS's built-in information providers (grid sensors) since they are limited to providing information about grid entities like processors, file systems, and operating systems among others. On the other hand, DØ/SAM-Grid's resources comprise of entities like Stations, projects, disks, and groups. Hence, the prototype utilizes its own set of information providers as shown in figure 5.18.

Different *DØ/SAM-Grid information providers* execute at different levels: Monitoring site level, Execution site level, and the Job (sam-analysis project) level. The `sam-deployment-summary-reporter` and `sam-authorized-users-reporter` are monitoring site level grid sensors. The `sam-deployment-summary-reporter` provides information about all the execution sites (Stations) at a monitoring site. The `sam-authorized-users-`

`reporter` provides information about the authorized users at a monitoring site. The execution site level grid sensors are `sam-deployment-reporter`, `sam-config-disks-reporter`, `sam-config-groups-reporter`, and `sam-projects-reporter`. These grid sensors provide information related to a particular execution site at this monitoring site. The system also has a job (sam-analysis project) level grid sensor `sam-project-details-reporter` which provides detailed progress information about a specific job running on a known execution site on this monitoring site.

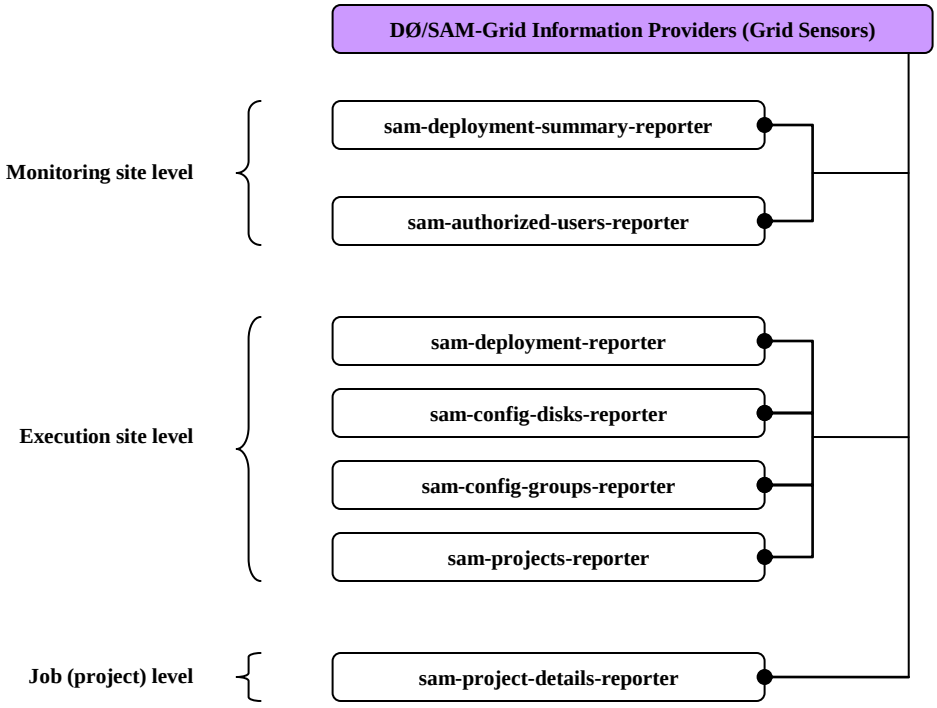

Figure 5.18 Information providers (grid sensors) developed for the system. Different sensors execute to provide information about different levels of the grid. Hence, the monitoring site level sensors need to be executed once for the monitoring site. The execution site level sensors need to be executed for each known execution site provided by the Level I sensor `sam-deployment-summary-reporter`. The Job (sam-analysis project) level sensor needs to be executed once for each project as provided by the Level II sensor `sam-projects-reporter`. The Level III sensor `sam-project-details-reporter` directly retrives information from a project's associated SAM-project-master.

The monitoring system has a provision for deploying multiple replicated information servers at each monitoring site. The system attempts to bind to the first available server so that if one of the servers fails, the monitoring shifts to the next available information sever. This feature enhances the fault tolerance of the system, and works transparently from a user.

5.3 Acknowledgments

The implementation of the prototype system has benefited greatly from regular meetings and discussions with the DØ/SAM-Grid team.

The fundamental model of the *DØ/SAM-Grid information providers* has been formulated by Igor Terekhov and Andrew Baranovski at FermiLab. Especially, the information providers like `sam-deployment-summary-reporter`, `sam-deployment-reporter`, `sam-projects-reporter` and `sam-project-details-reporter` have been originally created, and completely implemented by them.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

As new scientific methods that utilize data-intensive experiments are developed, their reliance on distributed high-performance computing will increase. Grid computing is an enabling technology that may perform a key role in harnessing the distributed computing power more effectively. There is a need to develop systems that assist in monitoring of this enormous distributed computing power.

6.1 Conclusions

DØ/SAM-Grid is being built on the SAM data-handling system and is expected to be one of the important grids used by major high-energy physics experiments. We have developed a prototype grid monitoring system that performs extensive monitoring of this grid, under certain assumptions like:

- All resources of the grid are SAM-Stations.
- All jobs in the grid are sam-analysis type of jobs (SAM projects).

Under these assumptions, the prototype monitoring system has been deployed and tested by monitoring 11 sites in 5 countries distributed across 3 continents, and has delivered the desired level of performance, with as much de-centralized control of information as possible. Robustness is a major feature of the system, with the performance unaffected by unavailable or unreachable parts of the grid. The system utilizes a layered architecture for information processing, with the sources of information distributed across the grid; and also integrates information generated using different grid middleware realms.

The system follows a novel design and information model developed in coherence with the SAM architecture. A web based front-end of the prototype system serves the need for ubiquitous monitoring of the grid. The usage of LDAP and CORBA at different levels provides more consistent discovery of resources, than the default LDAP based discovery mechanisms in contemporary grid middleware tools.

6.2 Future Work

The system needs to be evolved with the new releases of future versions of the SAM software, due to its being closely coupled with the SAM interfaces. Also, web-based job submission services need to be added to the monitoring system to converge the design into a complete grid-portal.

Monitoring of different types of jobs (for instance, *monte-carlo* jobs and *caf* jobs) that execute on a diverse range of resources needs to be taken into account in the future enhancements to this system.

A full-fledged MDS GRIS/GIIS discovery mechanism needs to be utilized to dynamically discover a newly added monitoring site on the grid. A possible methodology is to attach geographical co-ordinate information with each monitoring site, and converge it with the positional co-ordinates on the map at the web interface of the portal. This is, however, dependent on the improvements in performance of the new releases of MDS/OpenLDAP implementation.

The success of the prototype grid monitoring system under the current assumptions provides great motivation for this future work.

BIBLIOGRAPHY

[BLUEPRINT] Foster, I., and Kesselman, C. (eds.). *The Grid: BluePrint for a new Computing Infrastructure.* Morgan Kaufmann, 1999.

[FOSTERARTICLE] Foster, I. *What is the Grid?.* Grid Today, Vol.1 No. 6, July 22, 2002.

[MDS-2] Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C., *Grid Information Services for Distributed Resource Sharing.* Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, 2001.

[SAM-1] Baranovski, A., Bertram, I., Garzoglio, G., Lueking, L., Terekhov, I., Veseli, S., Walker, R. *SAM-Grid: Using SAM and Grid middleware to enable full function Grid Computing.*

[SAM-2] Baranovski, A., Garzoglio, G., Koutaniemi K., Lueking, L., Patil, S., Pordes, R., Rana, A., Terekhov, I., Veseli, S., Yu, J., Walker, R., White V. *The SAM-GRID project: architecture and plan.* Nuclear Instruments and Methods in Physics Research, Section A (Elsevier Science), Proceedings of ACAT'2002.

[D0-PAGE] http://www-d0.fnal.gov

[D0-DOC] *Physics Highlights from the D0 Experiment: 1992-1999.* Fermi National Accelerator Laboratory, Batavia, IL, USA. (File: *aboutd0__Highlights_v18_final.doc*)

[LEE-SLIDES] Lueking, L., *SAM: The D0 Data Grid.* Grid 2001, Denver, Colorado, 2001. (File: *sc2001_grid2001_presentation_20011112.ppt*)

[SAM-PPDG] Carpenter, L., Lueking, L., Moore, C., Pordes, R., Trumbo, J., Veseli, S., Terekhov, I., Vranicar, M., White, S., White, V. *SAM and the Particle Physics Data Grid*

[SAM-PAGE] http://www-d0db.fnal.gov/sam

[ENSTORE-PAGE] http://www-isd.fnal.gov/enstore

[GLOBUS-PAGE] http://www.globus.org

[GLOBUS-META] Foster, I., and Kesselman, C. *A Metacomputing Infrastructure Toolkit.* International Journal of Supercomputing Applications, 11(2):115-128, 1997.

[CONDOR] Litzkow, M., Livny, M., and Mutka, M. *Condor – A Hunter of Idle Workstations*. Proc. 8th Intl Conf. on Distributed Computing Systems, 1988, pp. 104-111.

[CONDOR-G] Frey, J., Tannenbaum, T., Livny, M., Foster, I., Tuecke, S. *Condor-G: A Computation Management Agent for Multi-Institutional Grids.* Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001

[LDAP-1] Wahl, M., Kille, S., Howes, T., *RFC 2251, Lightweight Directory Access Protocol (v3).* (Work in progress), IETF, July 1997

[CONDOR-ADS] Nicholas Coleman, *An Implementation of Matchmaking Analysis in Condor, Masters Project report*, University of Wisconsin, Madison, May 2001

[RFC-2253] Wahl, M., Kille, S., Howes, T. *RFC 2253, A String Representation of Distinguished Names*, (Work in progress), IETF, Dec 1997.

[RFC-2256] Wahl, M., *RFC 2256, LDAPv3 Schema*. (Work in progress), IETF, Dec 1997

[RFC-2849] Good, G., *RFC 2849, LDAP Data Interchange Format*. (Work in progress), IETF, June 2000

BIOGRAPHICAL INFORMATION

Abhishek S. Rana conducted Research and Development in Advanced Distributed Computing methodologies at the Fermi National Accelerator Laboratory in Batavia, Illinois, and received his M.S. in Computer Science and Engineering, at The University of Texas at Arlington, Texas, in December 2002. This work was supported with funds provided by the United States Department of Energy, and influenced the design and adoption of Globally Distributed Grid Computing at the Tevatron and at the Large Hadron Collider.