UNIVERSITY OF PUERTO RICO

Mayagüez Campus

Development of Portlet-Based Sensor Management Interfaces

by

César Augusto Sandoval León

A thesis submitted in partial fulfillment for the degree of

MASTER OF ENGINEERING
in
ELECTRICAL ENGINEERING

Faculty of Engineering

Electrical and Computer Engineering Department

July 2008

Approved by:

Domingo Rodríguez, Ph.D.	Date
Member, Graduate Committee	
José G. Colom Ustáriz, Ph.D.	Date
Member, Graduate Committee	
Wilson Rivera, Ph.D.	Date
President, Graduate Committee	
Silvestre Colón, M.S.	Date
Representative of Graduate Studies	
Isidoro Couvertier, Ph.D.	Date
Chairperson of the Department	

Declaration of Authorship

I, Cesar Sandoval, declare that this thesis titled, "Development of Portlet-Based Sensor Management Interfaces" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated
- Where I have consulted the published work of others, this is always clearly attributed
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work
- I have acknowledged all main sources of help
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself

Signed:

Date:

"You push the button, we do the rest."

George Eastman

UNIVERSITY OF PUERTO RICO

Abstract

Faculty of Engineering

Electrical and Computer Engineering Department

Master of Engineering

by César Augusto Sandoval León

Efficient and fast management of complex systems requires the development of efficient interfaces. The purpose of this document is to describe the development of a set of interfaces to provide sensor management capabilities using grid portals. An automated installation package was develop to deploy portlet-based management interfaces in order to facilitate the installation, deployment and development of such management interfaces using grid portals. Two portlet-based interfaces were developed. The first interface allows control the features of the PR1 meteorological radar. The second interface allows the management of Gumstix sensors pertaining to acoustic sensing applications. Also, an application portlet was developed to manage PR1 data. Additionally, a simple rain data detector service that allows the automated storage of raw data from the PR1 radar was developed.

UNIVERSIDAD DE PUERTO RICO

Resumen

Facultad de Ingeniería
Departamento de Ingeniería Eléctrica y Computadoras

Maestría en Ingeniería

por César Augusto Sandoval León

El desarrollo de interfaces permite manipular sistemas complejos de una manera rápida y sencilla. Este reporte de proyecto describe el desarrollo de interfaces tipo portlet para manejo de sensores dentro de portales grid. Dada la complejidad de instalación de portales grid se construyo un paquete de instalación automática que provee un portal grid y permite una rápida implementación de interfaces de manejo de sensores como parte de este. Se desarrollaron dos interfaces: una para el manejo del radar meteorológico PR1 y otra para el manejo de sensores acústicos en malla para monitoreo ambiental. También se coloco en funcionamiento un servicio para acceder los datos meteorológicos del PR1, además de la detección automática de sus datos entre lluvia y no lluvia.

Acknowledgements

I would like to thank to my advisor Prof. Wilson Rivera for giving me the opportunity to work with him, for his support, patience, dedication and help in this thesis. Also, I would like to thank to the members of my graduate committee, Prof. José Colom and Prof. Domingo Rodríguez for their valuable help and advice. This work has been supported by the NSF Engineering Research Center for Collaborative and Adaptive Sensing of the Atmosphere (CASA) under Grant N. 0313747.

I also want to thanks all the team members of the ICIS and CLiMMATE Labs, you were more than friends; you became my family these years of my life. Thanks to Angel, Carlos, Cesar, Diego, Fernando, Gail, John, Jose, Juddy, Kennie, Maniel, Matthew, Mariana, Miguel, Oliver, Omar, Omar X, Osvaldo, Pablo, Rene, Ricardo, Yilda, Yuji, and all my special friends and coworkers for their company and advices.

I thank God for giving me the strength and health to accomplish this stage of my life.

Contents

List of Figures

List of Tables

Abbreviations

AJAX	**A**synchronous **JA**vaScript and **X**ML
CASA	**C**ollaborative **A**daptive **S**ensing **A**tmosphere
DCAS	**D**istributed **C**ollaborative **A**daptive **S**ensing
JSP	**J**ava**S**erver **P**ages
NEXRAD	**Nex**t-Generation **Rad**ar
STB	**S**tudent **T**est **B**ed
UPRM	**U**niversity of **P**uerto **R**ico at **M**ayagüez
WALSAIP	**W**ide **A**rea **L**arge **S**cale **A**utomated **I**nformation **P**rocessing
XML	E**x**tensible **M**arkup **L**anguage

Dedicated to my family.....

Chapter 1

Introduction

The National Science Foundation Engineering Research Center for Collaborative Adaptive Sensing of the Atmosphere (CASA) is focused on developing Distributed Collaborative Adaptive Sensing (DCAS) applications as a system technology to improve our ability to monitor the earth's lower atmosphere. Current approaches to sampling the first three kilometers of atmosphere are physically limited in their ability to provide the required resolution and coverage. For example, radar technology is currently limited by the focus on long range sensing by single instruments. Requiring radar to view distances up to 240km, as in the case of NEXRAD, introduces the problem of the earth's curvature [9]. As the range increases away from the radar, the earth's surface curves away under the radar beam. This causes the volume of atmosphere being observed to be located at an increasing height above the earth's surface. The radar is unable to observe the atmosphere close to the earth's surface where people live. DCAS aims to radically alter the radar paradigm. Rather than relying on single radar to provide long range (hundreds of kilometers) coverage, DCAS proposes to mosaic the output of lower power shorter range (tens of kilometers) radars.

The Student Test Bed (STB) is a CASA education project that is focused on constructing a network of radars to provide detailed Quantitative Precipitation Estimation (QPE) in the western area of Puerto Rico. The primary mission of the Puerto Rico STB is to validate the DCAS approach in tropical areas. In the test bed a customized version of the commercially available Raytheon MK2 marine radar [7] has been deployed.

In this work we have developed a portlet-based sensor management interface that can be used to configure and control sensor instruments. We have integrated this interface to the STB grid portal developed at UPRM. The portlet-based interface allows user with the appropriate authorization to control relevant functions of the radar from remote locations. We have extended this work to develop a portlet-based interface to manage gumstix sensors pertaining to acoustic sensing applications.

1.1 Project Contributions

The main contributions of this work are summarized as follows:

- An automated package builder to deploy portlet-based sensor management interfaces. The package provides Java, Ant, Tomcat, and the Gridsphere Framework. The development of portlet interfaces was improved with the use of AJAX and Java Scripts. This allows the portlets to be updated without refreshing the entire portal page.
- A portlet-based remote radar controller interface that allows users with appropriate authorization to control remotely a radar. This interface will help in the testing and validation of the PR1 radar.
- A simple rain data detector service that allows the automated storage of raw data from the PR1 radar. This mechanism reduces the storage of raw data. End-users can access the raw-data acquired from the PR1 radar through the grid portal interface.
- An acoustic sensor portlet-based interface for environmental surveillance monitoring. End-users can set the audio parameters at gumstix sensors, and manage audio files.

1.2 Project Report Organization

The remainder of this report is organized as follows: Chapter 2 presents a literature review of the most relevant research work conducted in the integration of sensors and grid computing. Chapter 3 discusses the implementation issues of the portlet-based sensor management interfaces. A summary of conclusions and future work are presented in Chapter 4. Finally, Appendix A provides an installation and administration guide for portlet-based sensor management interfaces, and Appendix B provides the setting up of public-key authentication with sensors.

Chapter 2

Background Work

This chapter survey previous work related to the integration of grid computing and sensors technologies.

2.1 Distributed Collaborative Adaptive Sensing (DCAS)

The Center for Collaborative Adaptive Sensing of the Atmosphere (CASA) is a joint project operated by the University of Massachusetts (lead university), the University of Oklahoma, Colorado State University, and the University of Puerto Rico at Mayagüez and in partnership with the private sector. CASA was established to develop a new generation of small, low-power, low-cost radars that could operate dynamically, adaptively, and collaboratively to detect hazardous weather, and thereby increase detection rates and warning time in order to save lives and properties.

The center is focused on developing Distributed Collaborative Adaptive Sensing (DCAS) as a system technology to improve our ability to monitor the earth's lower troposphere. The *Distributed* term refers to the use of small radars, located to sense close to the ground in spite of the Earth's curvature and avoid resolution degradation caused by radar beam spreading. The *Collaborative* operation improves the sensitivity, precision, and resolution with the coordination of the beams from multiple radars to view the same area. The *Adaptive* allows the dynamically reconfiguration of these radars and there computing infrastructures when changing the weather conditions.

The DCAS system has five principal components: (1) sensors as radars; (2) meteorological algorithms that detect, track, and predict hazards; (3) interfaces that enable end-users to access the system; (4) storage servers; and (5) an underlying substrate of distributed computation that dynamically process sensed data and manage system resources. The DCAS system is working on with a network of X-band radars increased resolution and volume coverage of observations in the lowest kilometers of the troposphere.

2.1.1 The NetRad System

In terms of networking, a current project under development in CASA is *NetRad*, a prototype DCAS system. The main goal of the project is detect and track tornados within 1 minute of formation with a temporal error no greater than 1 minute.

The NetRad system architecture is shown in Figure 2.1. The system operation cycle is divided in two parts of 30 seconds. In the first part, a set of scanning commands are transmitted to the radars. The radars are continuously collecting and sending data in real-time to the *System Operation Control Center (SOCC)* node. A data quality control mechanism is applied at the SOCC and the data is then sent to the *Meteorological Command and Control (MC&C)*, see Figure 2.2. The MC&C controls the system main loop analyzing data from remote radars. Also, it identifies in the data the meteorological features to determine the next radars scan strategy, and finally reports the features to end-users as described in [10].

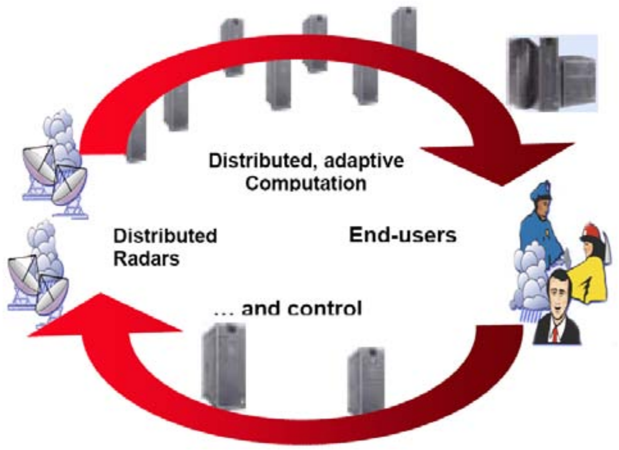


Figure 2.1: NetRad System Architecture [1]

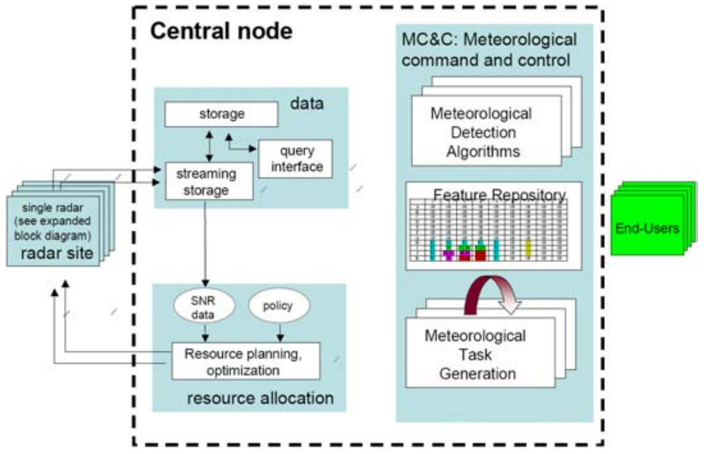

Central node
storage
data
query interface
streaming storage
single radar (see expanded block diagram)
radar site
SNR data
policy
Resource planning, optimization
resource allocation
MC&C: Meteorological command and control
Meteorological Detection Algorithms
Feature Repository
Meteorological Task Generation
End-Users

Figure 2.2: NetRad Data Flow [1]

An interesting feature of this module is the radar data filtering for particular hazardous weather features with the use of existing WDSS-II software. The observed data is summarized on a grid repository. In the second part of the cycle the radar network is optimized for scanning the particular features observed after applying cluster analysis techniques to the data.

2.2 What is Grid Computing?

Grid computing deals with the computing power that is supplied by a set of resources in a distributed environment. More formally the Open Grid Forum [11] defines a grid as a system concerned with the integration, virtualization, and management of services and resources in a distributed, heterogeneous environment that supports collections of users and resources *(virtual organizations)* across administrative and organizational domains *(real organizations)*. A grid system can be characterized by three important aspects [12]:

1. A *system that coordinates resources that are not subject to centralized control*: Sharing implies direct access on behalf of *virtual organizations (VOs)* [13] to computers, software, data, and other resources for collaborative problem-solving. VOs associate users, their requests, and the set of interacting resources. Sharing resources in a VO should be highly controlled, with resource providers and consumers

defining clearly and carefully what is shared, who is allowed to share, and the conditions under which sharing occurs.

2. A *system using standard, open, general-purpose protocols and interfaces*: To achieve desirable communication among those heterogeneous VOs, it is very important for both users and resources, to offer authentication, authorization, resource discovery, and resource access through standard protocols. The Open Grid Forum contributes to this grid computing aspect by developing such protocols. A *standard protocol* allows resource-sharing arrangements to be established with any interested party and thus creating a compatible and interoperable distributed system. A *general-purpose protocol* enables the implementation of general-purpose services and tools by means of this standardization.

3. A *system to deliver nontrivial qualities of service*: To supply the applications requirements the system must be able to coordinate the allocation of resources according to nontrivial quality of service criteria.

2.2.1 Grid Portals

Computational science portals are emerging as useful and necessary interfaces for performing operations on the Grid. A Grid Portal provides a customizable interface allowing scientists to perform a variety of Grid operations including remote program submission, file staging, and querying of information services from a single, secure gateway. A Grid portal is a web based application server enhanced with the necessary software to communicate to Grid services and resources. A Grid portal provides application scientists with a customized view of software and hardware resources specific to their particular problem domain and provides a single point of access to Grid resources they have already been authorized to use. A portal user is provided with a persistent, customizable profile that contains information that is stored securely on the portal and provides details on past jobs submitted, the set of computers they have access to, and any other information that is of interest to a particular user [14]. Therefore, Grid portals hide cyber-infrastructure complexity via easy-to-use interfaces, creating gateways to computing resources and data.

2.2.1.1 GridSphere Portal Framework

The primary goal of GridSphere portal framework [15] project is to develop a standards based portlet framework for building web portals, and a set of portlet web applications that work seemlessly with the GridSphere framework to provide a complete Grid portal development solution. The integration of the GridSphere portal framework with the collection of gridportlets provided as an add-on module forms a cohesive grid portal end-user environment for managing users, supporting remote job execution, and providing access to information services. The GridSphere portal framework provides an implementation of the JSR 168 portlet API standard. It supports the development of re-usable portlets and portlet services. It includes a set of core portlets and portlet services that provide the basic infrastructure required for developing and administering Web portals. A key feature of their design is that it builds upon the *web application repository (WAR)* deployment model to support third-party portlet web applications. In this way, portlet developers can easily distribute and share their work with other portal projects that use GridSphere to support their portal development.

2.2.2 Portlets

Portlets are a Java technology web components managed by a *portlet container*, which is in charge of process requests and generate dynamic content [16]. They are used in portals to present user interfaces. In a portal context, portlets represent the link between web users and services. Portlets also can be user-oriented in the way that a given portlet can present different information for different users. A portlet runs on the portal server and allows content to be embedded into portal pages.

The *Java Portlet Specification (JSR 168)* [17] standard establishes a standard API for creating portlets. Portlet specification is required to achieve interoperability between portlets and Java-based portal servers or other web applications that implement the specification. The goal is to allow portlets to be packaged into *Web Application aRchive (WAR)* files and deployed in a standard way on any server implementing the specification.

Java portlets are similar to java servlets, but in contrast, portlets can not send redirects or errors to browsers directly, forward requests or write arbitrary markup to an output stream.

Another difference with servlets is that portlets rely on portal specific infrastructure functions such as access to user profile information, and also they depend on an environment maintained by the portlet container.

A *portlet container* [16] is the server-side run-time environment. It calls the component and provides component-specific services (such as user information and persistence). In this environment portlets are instantiated, used and destroyed. The portlet container is not a stand-alone container like the servlet container, instead, it is implemented on top of it. It reuses the functionality provided by the servlet container.

2.3 Grid computing technologies

Grid computing technologies [18] involve coordination, storage and networking of resources across dynamic and geographically dispersed organizations in a transparent way for users. The Globus Toolkit [19] is a community-based, open-architecture, open-source set of services and software libraries that support Grid-based applications [20]. The toolkit includes software services and libraries for monitoring, discovery and management, of grid resources as well as services for security and data management.

Recent developments in Grid technologies have concentrated on providing batch access to distribute computational and storage resources, while the integration of sensor instruments to these resources has not been discussed widely. Some interesting projects addressing this problem include GridCC, SensorML, SensorGrid and CIMA.

2.3.1 Grid Enabled Remote Instrument with Distributed Control and Computation (GridCC) Project

The main goal of the Grid Enabled Remote Instrument with Distributed Control and Computation project (GridCC) [2] is to exploit grid opportunities to provide a secure and collaborative environment, where distributed teams make use of the grid massive memory and computing resources for storage and processing of data generated by scientific equipment; providing also the ability to remotely control and monitor those equipments.

The seven components of the GridCC architecture are shown in Figure 2.3: (1) *Virtual Control Room (VCR)*, (2) *Execution Service (ES)*, (3) *Compute and Storage Elements (CE & SE)*, (4) *Instrument Element (IE)*, (5) *Global Problem Solver (GPS)*, (6) *Information and Monitoring Service (IMS)*, and (7) *Security Service (SS)*.

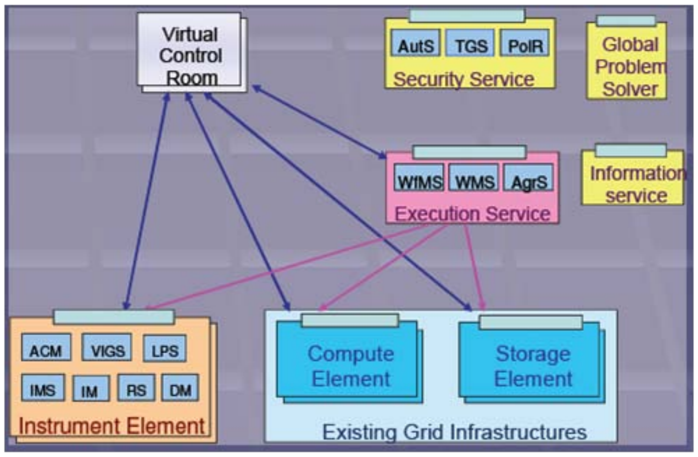


Figure 2.3: GridCC Architecture [2]

The user interface of the project is the *Virtual Control Room (VCR)*. It allows users to control instruments in real time and submit tasks to the *Execution Service (ES)* complex workflows.

Execution Service (ES) is the central component behind the GridCC. ES controls and maintains the status of the workflows executed by the user. The VCR can access such status information upon user request. ES also controls the quality of service (QoS) based on the resources currently available.

The GridCC executes the computation and store data in a group of resources called the *Compute and Storage Elements (CE & SE)*.

Another component is the *Instrument Element (IE)*. IE allows configure, partition, and control the physical instrument. A *Local Problem Solver (LPS)* is provided in each IE. The LPS manages and protects the instrument from harms with an automatic diagnosing.

An interested GridCC project feature is the two levels of automated problem solving: the local and the global level. The local level, called the *Local Problem Solver (LPS)*, solves problems in the instrument functions. The *Global Problem Solver (GPS)* solves problems related to the entire system.

The *Information and Monitoring Service (IMS)* distributes the monitoring data. The ES and VCR use the *Information Service (IS)* to consult the resources. The *Monitoring Service (MS)* disseminates the monitoring data with a publish system.

The GridCC project is described with more details in [21].

2.3.2 Sensor Model Language (SensorML) Project

Sensor Model Language (SensorML) [22] has been developed to provide standard models and XML encoding mechanisms for describing measurement process performed by sensors and instruments. The processes described by SensorML are discoverable and executable. All process must have defined their inputs, outputs and methods as well as any other relevant metadata.

SensorML provides a functional model for sensors, supports rigorous geolocation models, as well as mathematical models. SensorML can be applied virtually to any sensor, whether it is a in-situ or remote, and stationary or dynamic platform mount.

A grid sensor network project with this language is available in [23].

2.3.3 SensorGrid Project

The SensorGrid [24] project was developed to improve the Geographic Information Systems (GIS) applications with computing environment for real-time data sources. Based on *Service Oriented Architecture (SOA)* and High Performance Web Services principles, SensorGrid provides a set of standard interfaces for real-time data publishing with streaming and querying archived data. SensorGrid proposes a grid architecture called the *Scalable Proxy-based aRchItecture for seNsor Grid (SPRING)* [3], as shown in Figure 2.4.

The SPRING architecture has five components: (1) *Wireless Sensor Network (WSN)*, (2) WSN Proxy, (3) User Interface, (4) Computes Nodes, and the (5) Grid Network. The WSN Proxy, User Interface and Computes Nodes are connected to a central component called Grid Network. Also, it uses the proxy system as an interface between the WSNs and the grid. The SensorGrid project has delivered a prototype sensor grid testbed to

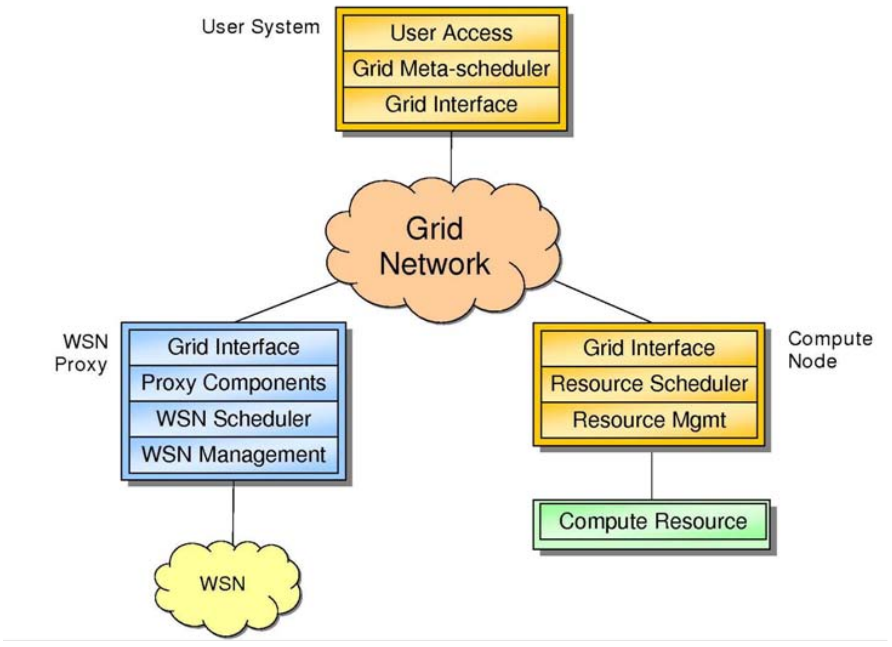


Figure 2.4: The SPRING Architecture [3]

implement the SPRING framework. For this testbed, the WSN was based on the Crossbow motes platform.

In this project, the principal component in the integration of WSNs with grid computing infrastructures is the WSN Proxy. It first receives the sensor jobs through the Grid Interface and then sends the jobs to the various Proxy Components. As shown in Figure 2.5, these components provide important services for the sensor grid including Data Management, Information Services, WSN Connectivity, Power Management, Security, Availability, and Quality of Service.

The SensorGrid project and the SPRING architecture are described with more details in [3].

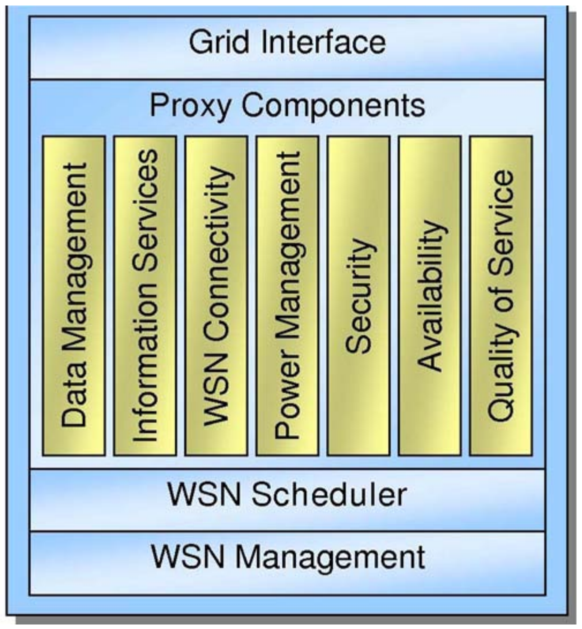


Figure 2.5: The Proxy Component Architecture [3]

2.3.4 The Common Instrument Middleware Architecture (CIMA) Project

The *Common Instrument Middleware Architecture (CIMA)* [4] project was developed to facilitate the instruments and grid integration improving the real-time data of the instruments. The project leverages on grid implementation standards such as the *Open Grid Services Architecture (OGSA)* and *Common Component Architecture (CCA)*. A CIMA goal is share instrument resources in geographically distributed Grids.

Figure 2.6 illustrates the CIMA architecture. It has two kinds of components: the instrument components (dark background) and the CIMA components (light background). The instrument components usually consist of an interface protocol (bottom) connected to the sensor hardware to control its basic functions. The CIMA components allow interact with the sensor or instruments through Web Services. Also, the Proteus communications library improves the data acquisition with the most appropriate communication protocols.

The general scheme of communication between sensor and consumer in the CIMA architecture is shown in Figure 2.7. A consumer can receive data from one or more sensors. Also, each sensor can be used for more than one consumer.

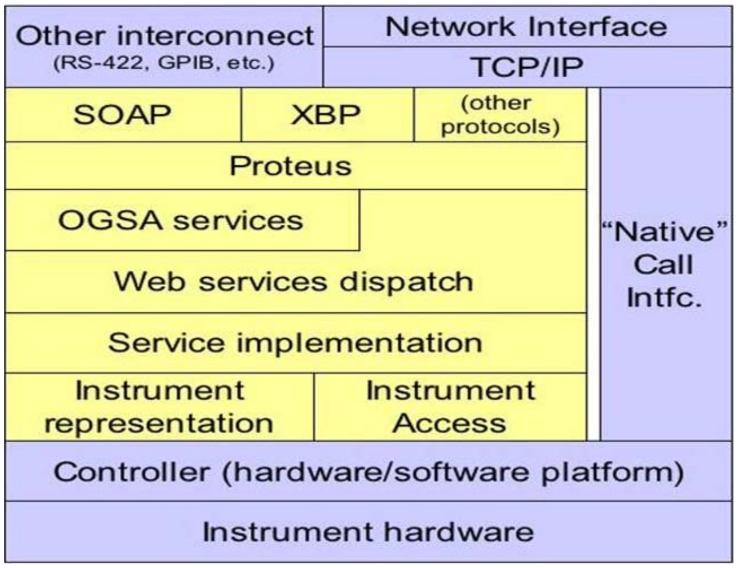


Figure 2.6: CIMA Architecture [4]

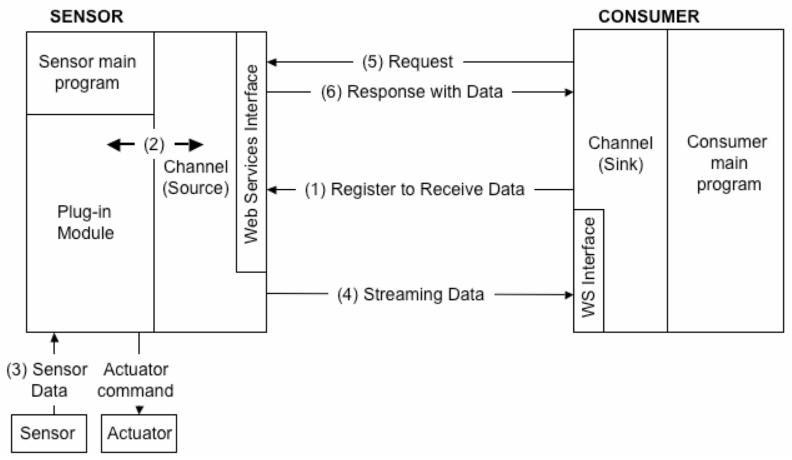


Figure 2.7: CIMA in Context [5]

The communication has the following sequence of steps: 1) user application registers to receive streaming data from a given sensor; 2) Channel service set up the appropriate data stream; 3) Sensor data is acquired; 4) Sensor data is sent to the user application. Also, the user application can make a sensor status request in any time (steps 5 and 6 in Figure 2.7). A problem with CIMA approach is that the middleware architecture might be too complex to be implemented on simple sensor devices with low computational and processing capability.

A CIMA approach with sensor and instrument for Crystallography is described with details in [25].

2.4 Review of Grid Computing and Sensor integration

The main research issues and challenges concerning the integration of conventional sensors and grid environment, arises due to the inherent limitations of sensors devices, such as energy management, coverage and communication, security [26–29]. Specific research challenges include interoperation of services, interconnection and networking, coordinated quality of service (QoS), and scalability [5, 30, 31].

A natural approach to integrate sensor nodes into the grid is to adopt the grid standards and APIs. The *Open Grid Services Architecture (OGSA)* is based on established web services standards and technologies like XML, SOAP, and WSDL. If sensor data were available in the OGSA framework, it would be easier to exchange and process the data on the grid. However, since sensor nodes have limited computational and processing capabilities, it may not be feasible for sensor data to be encoded in XML format within SOAP envelopes, and then transported using Internet protocols to applications. Grid services are also too complex to be implemented directly on most simple sensor nodes.

The network connections in conventional grids are typically fast and reasonably reliable. On the contrary, sensor nodes are usually connected via wireless and ad hoc networks which are generally low-bandwidth, with high-latency, and prone to interruptions or failures. Grid networking protocols are based on standard Internet protocols like TCP/IP, HTTP, FTP, etc. On the other hand, in wireless sensor networks the sensor nodes access a shared radio channel to communicate with neighbors are habitually based on proprietary protocols, especially for the *Medium Access Control (MAC)* protocol and routing protocol. Thus, techniques to interface sensor network protocols with grid networking protocols are necessary.

Scalability is the ability to a sensor grid to increase the capacity of sensor data collection. The sensors and grid integration architecture should allow multiple wireless sensor networks, possibly owned by different virtual organizations, to be easily integrated with compute and data grid resources. For these reason the integration must be scalable and support the agglomeration of sensor resources, without substantial changes to its software architecture.

Quality of Service (QoS) is a key issue. QoS determines whether the integration between sensors and grid environment can provide sensor resources on demand efficiently. QoS requirements of sensors applications must be described in a high-level manner. Mapping higher-level requirements into lower-level QoS required the development of a good mechanism. This mechanism must take into consideration a set of parameters that specify the amount of resources to be allocated, such as the amount of sensors, memory, and network bandwidth [32–34].

2.5 Portlet-based Interfaces

Assuming that you have in place an architecture to integrate sensor and grid infrastructures, a remain question is how to provide friendly easy-to-use/easy-to-manage interfaces to end users.

In grid environments, one approach is the usage of portlet-based interfaces. The portlet-based interfaces allow the management of complex system in a fast and easy way. The most important feature of this kind of interfaces is the possibility of hiding complex management processes behind the application interfaces. For example, providing file management in a portlet-based interface allows the users access to files, collections, and metadata for local and remote files, and also support third party file transfer. In the same way, the job management provides mechanisms for job execution and monitoring.

Porlet-based interfaces for sensor management allow users to control the relevant functions of the sensor from remote locations. Also, the interfaces inform users of the status of data and functions in the sensor. All these processes run transparently to end users.

Portlet-based interfaces also can be user-oriented in the way that a given application can present different information for different users. The interfaces establish different access rights depending on the user roles inside the application. Also, different applications and services can be added as components on the same portlet-based interface.

As a grid portal application, the portlet-based interfaces are accessed through a web browser, giving mobility to users, enabling them to work from anywhere without the need of installing additional third-party software on their machines. Therefore, knowledge of Linux environments or specific software are not necessary, as all the connections and interactions with the complex systems are transparent to the users.

Examples of portlet-based interfaces for sensor management are available in [35–37].

Chapter 3

Implementation of Portlet-Based Interfaces

This chapter presents a detailed description of the implementation of portlet-based management interfaces.

3.1 The STB Grid Architecture

The STB grid infrastructure illustrated in Figure 3.1 is an effort to integrate grid computing with radar technologies. It has three principal components: the PR1 radar, the CASA's server and the PDClab Grid Testbed. The radar is accessed and manipulated from the server, via a wireless link access point. The server is the host of the STB grid portal at UPRM. The portal interface provides a set of services to manipulate radars and its data, with transparent access to end-users. The STB portal allows user manipulation and storage of raw data and a service for the remote control of the PR1. Raw data from radars are sent to a CASA server via wireless communication. The PDCLab Grid Testbed provides resources required for storing and processing the data generated.

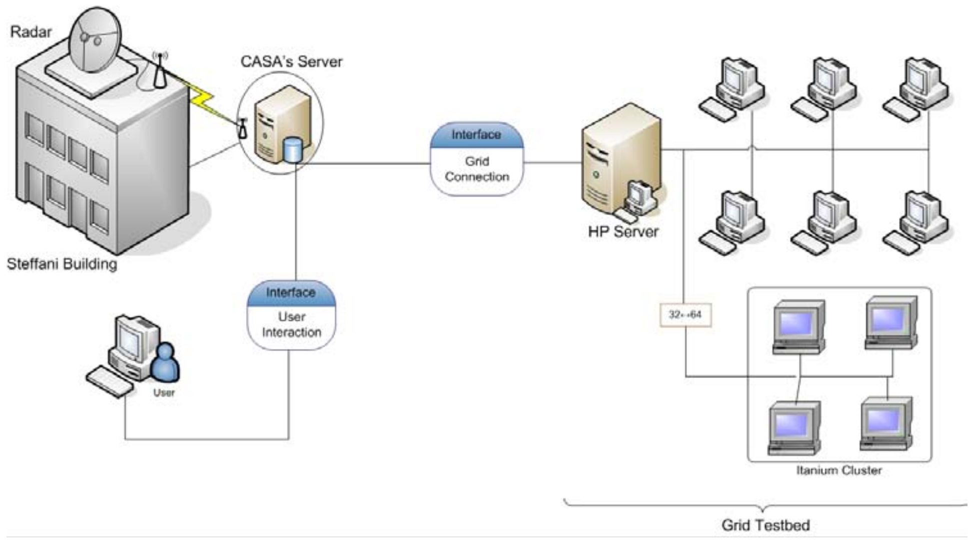


Figure 3.1: Mayagüez Radar Integration [6]

3.1.1 Radar Infrastructure

The first node (PR1) of the DCAS radar network was developed for CASA students of UPRM and is currently in the top of the Electrical and Computer Engineering building. The X-band radar has an IEEE 802.11b wireless link with the CASA server that is currently at the Atmospheric Phenomena Laboratory (APL) at the bottom of the ECE building. The Figure 3.2 shows the PR1 Radar System that is accessed and managed through the STB Grid Portal.

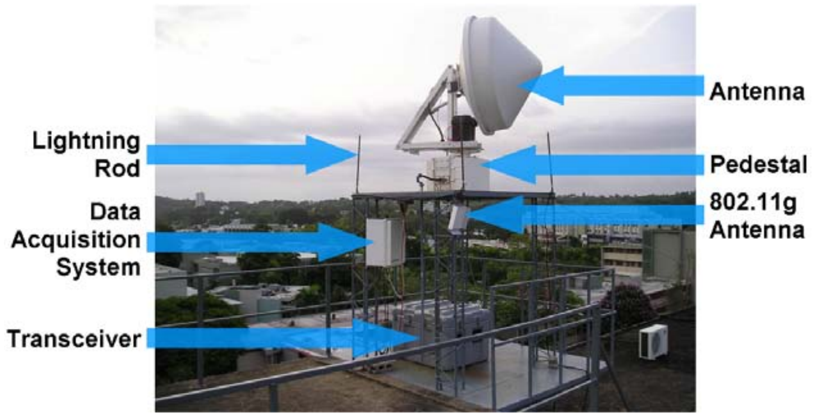


Figure 3.2: CASA STB's PR1 [7]

The radar system consist primary of a customized version of the commercially available Raytheon MK2 marine radar. Other PR1 components are the tower at the ECE building, wireless access point, transceiver, antenna and the Data Acquisition System. Figure 3.3 shows the internal parts of the Data Acquisition System of the radar.

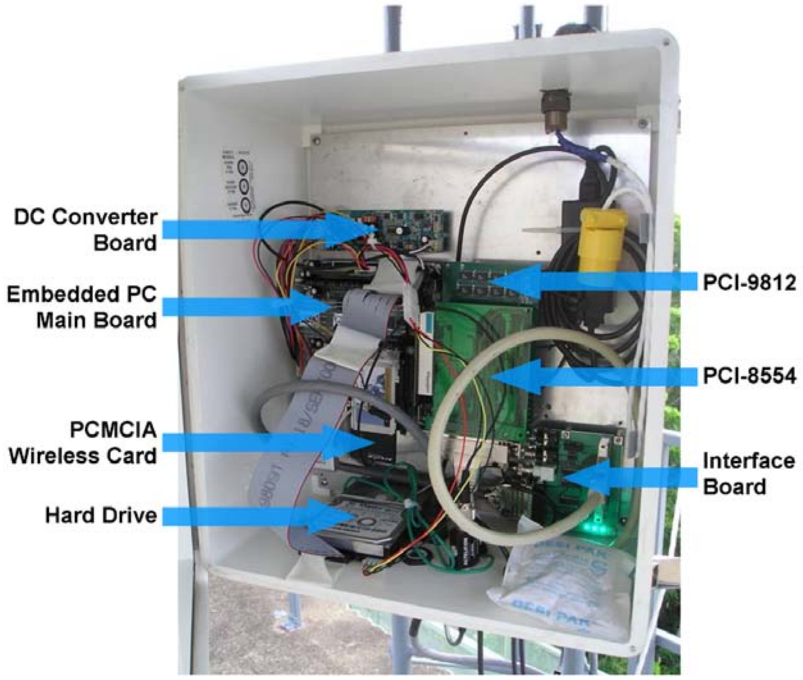


Figure 3.3: PR1's Data Acquisition System [7]

Radar control and data acquisition are handled by a Linux based VIA EPIA MII mini ITX embedded PC. It is equipped with an Adlink PCI-8554 counter/timer, PCI-9812 data acquisition card and a PCMCIA wireless card with 802.11g capabilities. Magnetron trigger signals are generated by the counter/timer card and fed into the high voltage modulator and calibration switch. Data from the antenna optical azimuth angle encoder is deciphered and sent through the PCI bus using the PCI-8554, while the video signal from the logarithmic amplifier is sampled by the PCI-9812, a Gage 12-bit data acquisition card. On the other hand, an inclinometer measures the antenna's elevation angle and transfers it to the embedded PC's serial port through a transparent Bluetooth wireless connection.

An interface board, between the transceiver and the embedded PC, was designed to handle all signal conditioning and provide software controllable transceiver and antenna rotation On/Off capabilities using the digital outputs of the PCI-8554 combined with relays. Furthermore, to synchronize the data acquisition with the magnetron trigger, the latter was split inside the interface board and fed into the PCI-9812. Thus, the PCI-9812 starts to acquire data every time the magnetron is triggered, [7].

3.1.2 Grid Testbed Infrastructure

The PDClab Grid Testbed, deployed at the University of Puerto Rico-Mayaguez, is an experimental grid designed to address research issues such as the effective integration of sensor and radar networks into grid infrastructures. The PDClab grid test-bed components run CentOS 4.2 and the Globus Toolkit 4.0.1. Figure 3.4 shows an overview of the Grid-Service based system structure. GridFTP can be used to improve data transport from the data server to the PDCLab Grid Testbed. Data exchange between server and Grid testbed is authenticated using Grid Security Infrastructure (GSI).

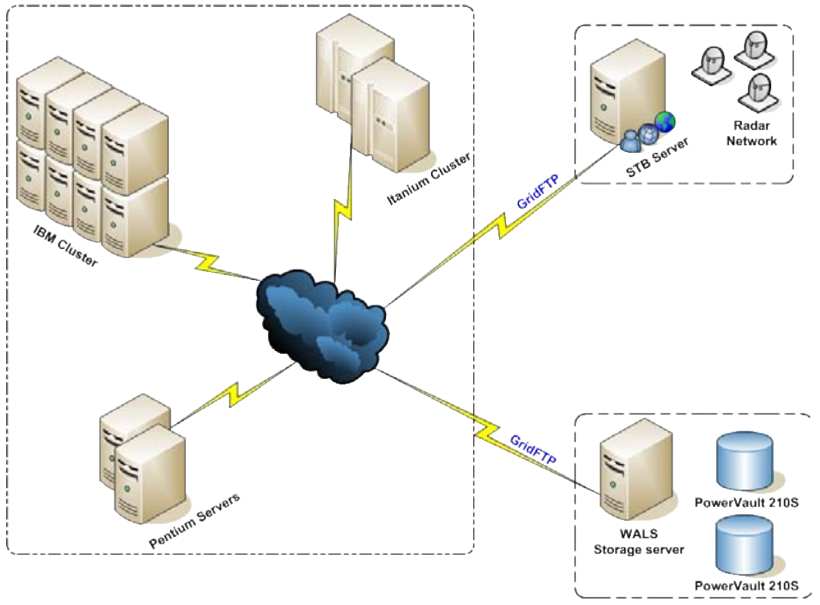


Figure 3.4: Grid-Service Based System Structure [6]

3.2 Automated Package to Grid Portals Deployment

Recently grid portals have became increasingly popular for creating customizable, Web-based interfaces to grid services and resources. The Gridsphere portal framework provides standards based portal for the easy development of modular web components, called portlets. Portlets are defined by a standard API and provide a model for developing new portal components that can be shared and exchanged by various portlet containers. Gridsphere provides both a portlet container, a collection of core portlets, and an advanced user interface library that makes developing new portlets easier for application developers. However, the installation and implementation of management interfaces to sensing instruments may be a very time consuming task. To overcome this problem, we have developed an automated package builder to deploy grid portals, specifically with the Gridsphere framework, and the development of portlet-based interfaces. The package provides the appropriate configuration of Java, Ant, Tomcat, and the Gridsphere Framework. The package starts with the installation of the software dependencies and the configuration of the Linux environment variables. The following code shows the environment variables used:

bashrc.entries

```
## Gridsphere Configuration by PDCLab
#
# General Settings for user environment
#
JAVA_HOME=/home/$USER/GridspherePDCLab/jdk1.5.0_15
JRE_HOME=$JAVA_HOME/jre
export JAVA_HOME
export JRE_HOME
PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$PATH:
ANT_HOME=/home/$USER/GridspherePDCLab/apache-ant-1.7.0
export ANT_HOME
PATH=$ANT_HOME/bin:$PATH:
GRIDSPHERE_HOME=/home/$USER/GridspherePDCLab/gridsphere-2.2.7
export GRIDSPHERE_HOME
CATALINA_HOME=/home/$USER/GridspherePDCLab/apache-tomcat-5.5.17
export CATALINA_HOME
#
## End of Gridsphere Configuration by PDCLab
```

Figure 3.5 shows the automated package components. The package has five components ready to work in a Linux distribution: (1) JDK, (2) Apache Ant, (3) Apache Tomcat, (4) Gridsphere, and (5) Portlet Application. *JDK* is the Java SE development kit. *Apache Ant* is used to compile and deploy the Gridsphere framework. *Apache Tomcat* is the servlet/JSP container. *Gridsphere* is the grid portal development framework that include the *Portlet Application* with the portlet-based interface.

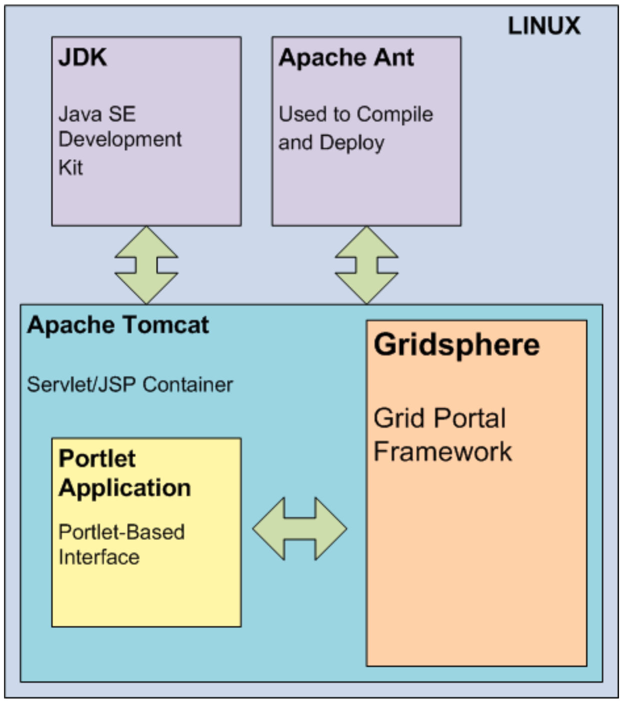


Figure 3.5: Automated Package Components

A complete guideline for installation and administration of the STB portal is provided in Appendix A.

3.3 Remote Radar Controller Interface

To facilitate testing and validation of the radar we have developed a portlet-based interface to control the radar. It allows users to control the relevant functions of the radar from remote locations. Therefore a user with appropriate certificates can manipulate the parameters of the radar for a weather acquisition data in any moment.

3.3.1 Local Radar Controller

The PR1 has a C program controller at its Linux mini ITX embedded PC. This program was created by the UPRM students Jose Ortiz and Manuel Vega. This program allows the user to start up or shut down the radar, enabling or disabling the trigger pulse with the configuration settings of frequency and pulse width. The trigger pulse is the electromagnetic pulse emitted in all directions from the radar, specifically from the magnetometer. Data acquisition is also another setting that can be managed from this controller system; acquisition cards can be activated and then data is processed and stored. The rotation of the radar structure can also be controlled.

3.3.2 Portlet-Based Remote Radar Controller Interface

The principal application portlet developed was a PR1 radar controller interface. This sensor management interface has a direct interaction with the C program controller located at the PR1 mini embedded PC and developed by UPRM CASA students. Figure 3.6 shows the portlet application developed.

The radar controller interface has two components "The Radar Controller Main" and the "Radar Camera". In the first area the users has all the controls and information about the radar status and, in the another area can see a live camera of the PR1. In this way some changes in the radar status can be seen in the live camera, e.g. the enable or disable of the radar rotation.

The "The Radar Controller Main" has four elements: User Online, Expert User, Controls, and Status Radar. *User Online* shows whether the radar is available or not. If busy, it shows who is the current user and her permission. *Expert User* has the fields where the users give her login and password to can control the radar, this login and password are different of the Gridsphere portal. *Controls* have the four identified parameters that can be used to control the radar. This section is only available after a successful expert user login. *Status Radar* shows a detailed history of all the operation in the radar with the date and time, user, and operation.

The radar controller interface has two access levels for users: the "Master" and the "Expert". When the section *User Online* says "Waiting User", as shown in Figure 3.7,

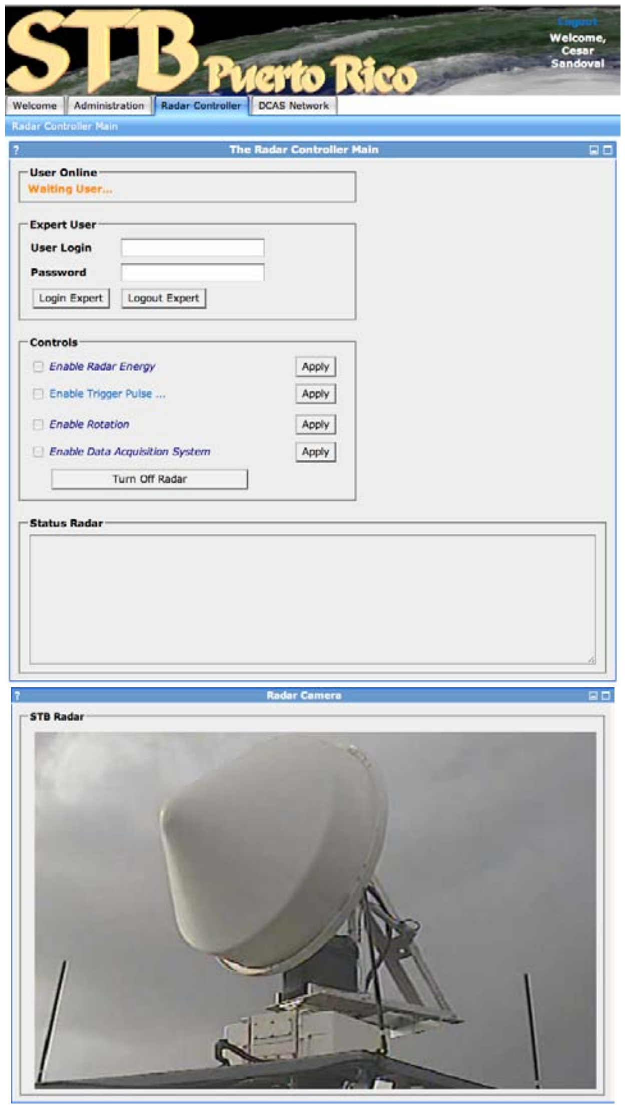
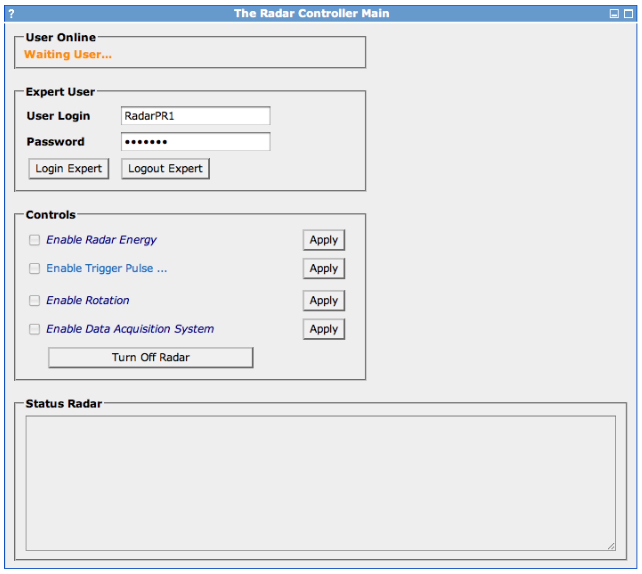


Figure 3.6: Remote Radar Controller Interface

the radar is available. After a successful login the *Controls* area is activated and a line is added in the *Status Radar* area, as shown in Figure 3.8. When the section *User Online* says that a user is online a Master user can login and take the control of the radar, therefore when a master user is online only other master user can take the control. Also, the *Expert User* area has a button to Logout. When it clicked the radar is turn off and available for a new user.

The section of *Controls* has four parameters to be controlled: (1) energy, (2) trigger pulse with its frequency and width, (3) rotation, and (4) data acquisition system. To enable a specific parameter the user only need to select the checkmark and click the respective



User Online: Waiting User...

Expert User

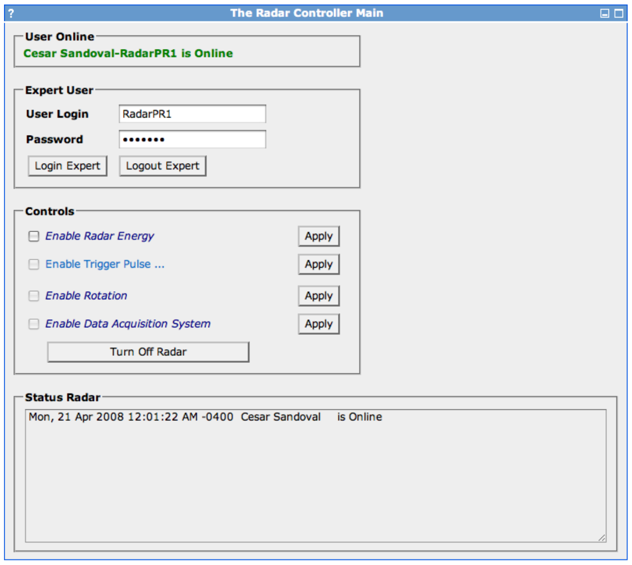
User Login	RadarPR1
Password	•••••••

Controls

- ☐ *Enable Radar Energy*
- ☐ Enable Trigger Pulse ...
- ☐ *Enable Rotation*
- ☐ *Enable Data Acquisition System*

Status Radar

Figure 3.7: Radar Controller Main



User Online: Cesar Sandoval-RadarPR1 is Online

Expert User

User Login	RadarPR1
Password	•••••••

Controls

- ☐ *Enable Radar Energy*
- ☐ Enable Trigger Pulse ...
- ☐ *Enable Rotation*
- ☐ *Enable Data Acquisition System*

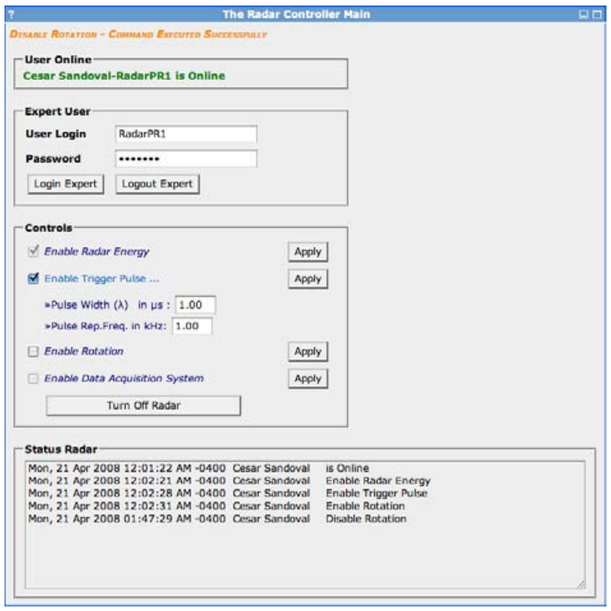
Status Radar

```
Mon, 21 Apr 2008 12:01:22 AM -0400  Cesar Sandoval    is Online
```

Figure 3.8: Radar Controller Main after Login

Apply button. The *Controls* area also has a turn off button that can be clicked at any moment, disabling the four parameters. When some action is clicked a line is added in the *Status Radar* area, furthermore a message of successful is showed at the top of the application. Figure 3.9 shows when some commands controls are executed.

The interaction with the local radar controller was improved forcing an initialization radar



DISABLE ROTATION - COMMAND EXECUTED SUCCESSFULLY

User Online

Cesar Sandoval-RadarPR1 is Online

Expert User

User Login RadarPR1

Password •••••••

Controls

✓ *Enable Radar Energy*

✓ Enable Trigger Pulse ...

»Pulse Width (λ) in μs : 1.00

»Pulse Rep.Freq. in kHz: 1.00

☐ *Enable Rotation*

☐ *Enable Data Acquisition System*

Status Radar

```
Mon, 21 Apr 2008 12:01:22 AM -0400  Cesar Sandoval   is Online
Mon, 21 Apr 2008 12:02:21 AM -0400  Cesar Sandoval   Enable Radar Energy
Mon, 21 Apr 2008 12:02:28 AM -0400  Cesar Sandoval   Enable Trigger Pulse
Mon, 21 Apr 2008 12:02:31 AM -0400  Cesar Sandoval   Enable Rotation
Mon, 21 Apr 2008 01:47:29 AM -0400  Cesar Sandoval   Disable Rotation
```

Figure 3.9: Executing controls command in the Radar Controller Main

sequence from the remote radar controller interface, making a better use of the radar, and preventing possible hardware damage. A states diagram of the radar controller interface is shown in Figure 3.10.

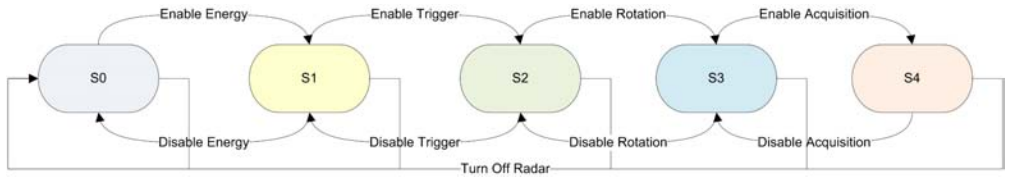


Figure 3.10: States Diagram of the Radar Controller Interface

The first state (S0) of the remote radar controller interface is when the user only can enable the energy of the radar, when the energy is enabled (S1) the user can enable the trigger pulse or disable the energy. Next, the trigger pulse can be activated (S2) but only if the parameter of width and frequency are valid. At this point (S2), the user can disable the trigger or enable the rotation of the radar. If the rotation is enabled (S3), the user can enable the data acquisition system or disable the rotation that represent come back to the before state (S2). Finally the data acquisition system can be enabled (S4) and the user can return to the before state (S3) disabling the data acquisition system. Also, at any state the user can turn off the radar and return to the initial state (S0).

3.3.3 Behind the Portlet-Based Interface

The portlet applications have two parts: the Java file and the JavaServer Pages (jsp) file. The jsp file includes the code to build the aspect graphics of the application, for example: buttons, text areas, check boxs, etc. The java file has the code to the initialization of the portlet and the components necessary to interact with the visual components at the jsp. The JSPs are interpreted by the Gridsphere container on-the-fly reducing the time taken to reload changes. With this features the development of portlet interfaces was improved with the use of AJAX and Java Scripts making more interactive the applications.

The following code shows a part of a jsp file in a Gridsphere environment with the use of AJAX and Java Scripts. The example code is a part of the remote radar controller interface.

RadarControllerMain.jsp

```
<%@ taglib uri="/portletUI" prefix="ui" %>
<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet" %>
<%@ taglib uri="/portletAPI" prefix="portletAPI" %>

<portletAPI:init/>
<portlet:defineObjects/>

<ui:messagebox beanId="messageBox"/>
.
.
.
  <ui:tablerow>
    <ui:tablecell width="10">
      <ui:checkbox beanId="checkbox2" disabled="true"/>
    </ui:tablecell>
    <ui:tablecell width="330">
      <a href="#"
       onclick="showHide('freqParams')">Enable Trigger Pulse ...</a>
    </ui:tablecell>
    <ui:tablecell width="60">
```

```
      <ui:actionsubmit beanId="apply2" action="apply2" key="Apply"/>
    </ui:tablecell>
    <ui:tablecell/>
  </ui:tablerow>

  <ui:tablerow>
    <ui:tablecell width="10">
      <ui:text value="" style="status"/>
    </ui:tablecell>
    <ui:tablecell width="330">
      <div id="freqParams" width="330"
       name="freqParams" style="display: none" >
      <FONT COLOR="#000080">&raquo;Pulse Width (&lambda;)
       &nbsp; in &micro;s : </FONT>
      <ui:textfield beanId="WidthTriguer"  value="1.00"
       size="5" maxlength="30"/><br>
      <FONT COLOR="#000080">&raquo;Pulse Rep.Freq. in kHz: </FONT>
      <ui:textfield beanId="PRFTriguer" value="1.00"
       size="5" maxlength="30"/>
      </div>
    </ui:tablecell>
  </ui:tablerow>
.
.
.
<script language="JavaScript" type="text/javascript">

  function showHide( whichLayer ) {
    var elem, vis;
    if( document.getElementById ) // standards work
      elem = document.getElementById( whichLayer );
    else if( document.all ) // old versions work
      elem = document.all[whichLayer];
    else if( document.layers ) // nn4 works
      elem = document.layers[whichLayer];
    vis = elem.style;
    if(vis.display==''&&elem.offsetWidth!=
     undefined&&elem.offsetHeight!=undefined)
    vis.display =
     (elem.offsetWidth!=0&&elem.offsetHeight!=0)?'block':'none';
    vis.display =
    (vis.display==''||vis.display=='block')?'none':'block';
  }

</script>
```

The Gridsphere framework has user interface (UI) tag and visual bean library to develop portlets. The previous code shows the libraries imported and the initialization tags. After that, we can start adding visual components. In order to have an accurate organization of the elements, these are positioned in a grid with the use of columns and rows. Next, check boxes, buttons and text fields elements are added with the UI tags `<ui:checkbox >`, `<ui:actionsubmit >`, and `<ui:textfield >`, respectively. Between these visual elements we can see components of HTML code allowing the AJAX calls. Finally the jsp file has the functions used in JavaScript code, these are inserted between the commands `<script language="JavaScript" type="text/javascript">` and `</script>`.

As said at the beginning of the section, the jsp file has a java file partner. The following code shows a java file partner for our jsp example file.

RadarControllerMain.java

```
package edu.uprm.radarcontroller.portlets;

import javax.portlet.*;
.
.
.
import org.gridlab.gridsphere.provider.portletui.beans.ActionSubmitBean;
import org.gridlab.gridsphere.provider.portletui.beans.CheckBoxBean;
.
.
.



/**
 * @author Cesar Sandoval Parallel and Distributed Computing Laboratory
 *         University of Puerto Rico at Mayaguez 2008
 */
public class RadarControllerMain extends ActionPortlet {

  private final static String DISPLAY_PAGE =
    "radarcontroller/RadarControllerMain.jsp";
  .
  .
  .

  public void init(PortletConfig config) throws PortletException {
    super.init(config);
    DEFAULT_VIEW_PAGE = "prepare";
    DEFAULT_HELP_PAGE =
      "radarcontroller/RadarControllerMainHelp.jsp";
  .
  .
  .
  }

  public void prepare(RenderFormEvent event) throws PortletException {
  .
  .
  .
  setNextState(event.getRenderRequest(), DISPLAY_PAGE);
  }

  public void apply2(ActionFormEvent event) throws PortletException {
  .
  .
  .
  CheckBoxBean checkbox2 = event.getCheckBoxBean("checkbox2");
  checkbox2.setDisabled(false);
  ActionSubmitBean apply2 = event.getActionSubmitBean("apply2");
  apply2.setDisabled(false);
  .
  .
  .
  }
.
.
.
}
```

This java file begin with the imports of the classes used in the application, the most important are the Gridsphere classes of the visual component used in the jsp file. Next, we have the main class called `RadarControllerMain` that extends the class of the Gridsphere portlets (`ActionPortlet`). The next lines have the global variables, as `DISPLAY_PAGE` that has the location of the jsp file partner. The Gridsphere portlet application always has the functions `init()` and `prepare()`. In the first function the initial parameter are defined and the second function has the commands that are executed every time that the portlet is refreshed. `prepare()` have the call to the jsp file. Finally, we can add the functions of the visual components calls. The previous java file shows as example the function `apply2()` that is called when the second button "Apply" is clicked. In this function we can call the bean libraries of the visual elements in order to interact with these.

An important feature in the development of sensor management interfaces is to allow the interface interact with the sensors. To achieve this we need to create a confidence among them. This confidence allows the interface execute remote instructions directly in the sensor. Therefore, because the communication between the elements are via secure shell (SSH) protocol, we need set up a SSH Public Key Authentication.

An SSH "key" is actually a matched pair of keys stored in two files. The private or secret key remains on the client machine, encrypted with a passphrase. The public key is copied to the remote machine (or sensor). When establishing a connection, the SSH client and server perform a complex negotiation based on the private and public key, and if they match (in a cryptographic sense), your identity is proven and the connection succeeds.

To set up a public-key authentication: first create an SSH key pair, next copy the public key to the sensor, and finally log into the sensor and install the public key. All this process is showed with more details in the Appendix B.

3.3.4 PR1 Live Radar Camera

To monitor the PR1, a camera was installed near to the radar. The camera needed have the featured of outdoor with weather proof and night vision. The camera acquired is shown in Figure 3.11.

The camera specifications are as follows:

Figure 3.11: Weatherproof High Resolution Night vision IP Network Color Camera [8]

- **Resolution:** 720x480 pixels
- **Video Coded:** M-JPEG4, 32 bit RSIC Embedded Processor
- **Audio:** External Audio Capable
- **Lens Type:** Sony 1/3" CCD, 8 mm fixed lens
- **Night Vision:** IR LED up to 100ft
- **Dimensions:** 143(w) x 78(h) x 56(d) mm
- **Case:** Weatherproof IP66 Casing
- **Power Source:** 12 VDC
- **Interface:** Lan or RS485 connection
- **Browser Compatibility:** Only Internet Explorer
- **Number of Client:** 5

Figure 3.12 shows a screenshots of the camera application.

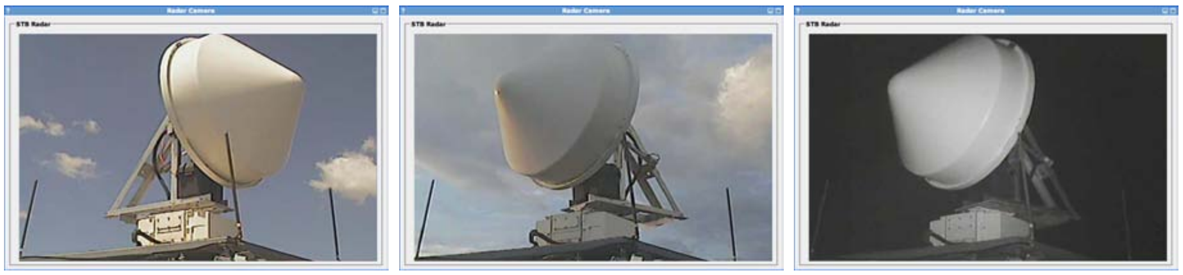


Figure 3.12: Day, Sunset and Night of the PR1 Live Camera

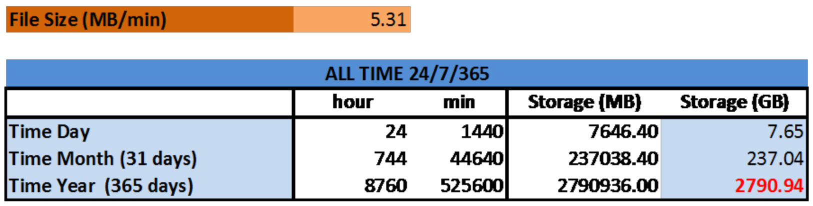
3.4 Rain Detector and Storage Services

The STB focuses on constructing a network of radars to provide precipitation estimations in the western area of Puerto Rico.

3.4.1 PR1 Data Storage

When the data acquisition is enabled, the information of each radar spin is saved in a .dat file, this file is sent to the CASA Server via wireless connection. The data is stored in a folder and the name has the date and time of the capture, the file `032808_132339_PR1.dat`, for instance, contains data captured on March 28, 2008 at 13:23:39.

The size of each file of radar data is around 5.31Mb. Table 3.1 shows the PR1 data storage operating 24/7/365. The average of data generated get the PR1 is 2 TB per year. This amount of raw data storage can be reduced using mechanism for data detection.



File Size (MB/min)	5.31

ALL TIME 24/7/365				
	hour	min	Storage (MB)	Storage (GB)
Time Day	24	1440	7646.40	7.65
Time Month (31 days)	744	44640	237038.40	237.04
Time Year (365 days)	8760	525600	2790936.00	2790.94

Table 3.1: PR1 Data Storage

3.4.2 Rain Data Detector and Automated Storage Services

To determine if the data acquired represents rain or not, and save storage in the process, a data detector has been developed. The simple rain data detector service allows the automated storage of raw data from the PR1 radar. The data is stored in CASA's server and automatically processed to determine if there was rain. If there was no rain the data can be deleted or moved to a trash folder, otherwise data is moved to an only rain data folder.

The algorithm was developed in `bash` code and has been tested and validated with PR1 radar data obtained from July 2007 until April 2008. We use 20 files to test and 100 files to validate, where the 50% was rain. Figure 3.13 and Figure 3.14 shows an examples of PR1 reflectivity plots with rain and without it, respectively.

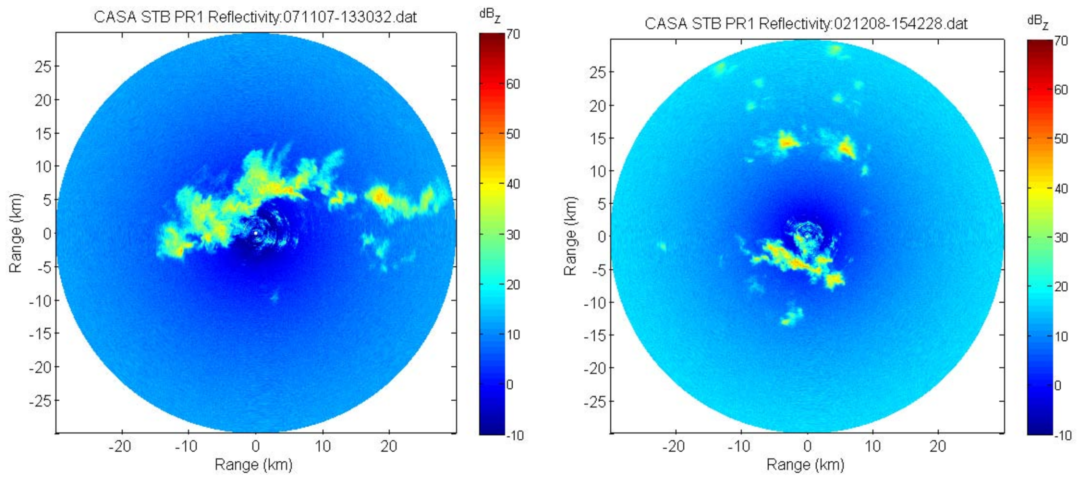


Figure 3.13: PR1 Reflectivity Plot with Rain

The pattern used to detect the rain was the power measurement. The files of the PR1 radar have two columns: one with the power measurement and other with the counter value of the radar angle position. The algorithm compute the average of the power values of each radar angle position and this average is compared with a power threshold. If the power average value of each radar angle position in a file is bigger that the power threshold we say that the data file is rain.

To found the appropriate threshold we analyze the data training. In the no rain data we found that the maximum power average was 0.418483 V although the rain data has a

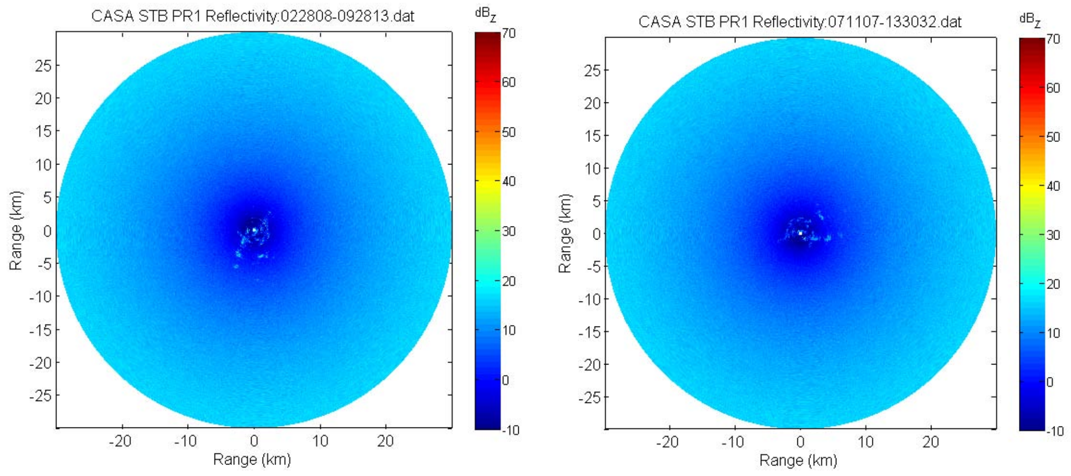


Figure 3.14: PR1 Reflectivity Plot without Rain

maximum power average around 0.6 V of even more, therefore we fix the threshold equal to 0.43 V. By using this threshold the detection with the training data was perfect. The detector was probed with this threshold in the validation data and the result is shown in Table 3.2. It shows a detection performance of 96% in rain data and 82% in no rain data. The data exhibit noise in the first kilometer range, that is produced for the near building to the radar (See Figure 3.14).

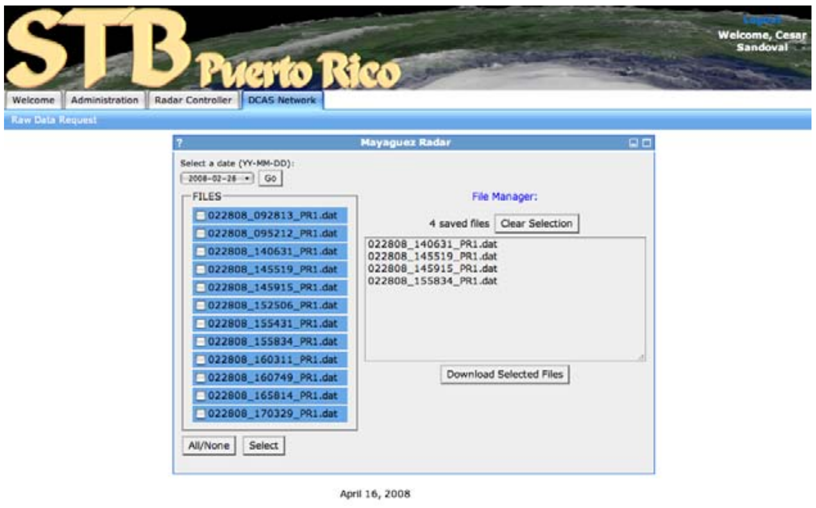
	Actual Data	
Classification	**Rain**	**No Rain**
Rain	**48** **96%**	**2** **4%**
No Rain	**9** **18%**	**41** **82%**

Table 3.2: Confusion Matrix for Rain Detection

3.4.3 PR1 Data Management

Users can access raw-data from PR1 radar through the grid portal. Currently, the files containing the raw-data are stored like text in a "dat" file. The data management portlets permit end-users to download the data.

Figure 3.15 shows the PR1 data request portlets. Once the user has been logged into the portal, the raw data request portlet is made available. The initial portlet shows a single selection form that permits the selection of the date of interest. Then, the data set selected can be downloaded as a compressed file. An important feature is that the user can download a data set of different days in a compressed file. This portlet is based on a replication data portlet proposed in [6].



Select a date (YY-MM-DD): 2008-02-26

FILES
- 022808_092813_PR1.dat
- 022808_095212_PR1.dat
- 022808_140631_PR1.dat
- 022808_145519_PR1.dat
- 022808_145915_PR1.dat
- 022808_152506_PR1.dat
- 022808_155431_PR1.dat
- 022808_155834_PR1.dat
- 022808_160311_PR1.dat
- 022808_160749_PR1.dat
- 022808_165814_PR1.dat
- 022808_170329_PR1.dat

4 saved files

```
022808_140631_PR1.dat
022808_145519_PR1.dat
022808_145915_PR1.dat
022808_155834_PR1.dat
```

April 16, 2008

Figure 3.15: PR1 Raw Data Request

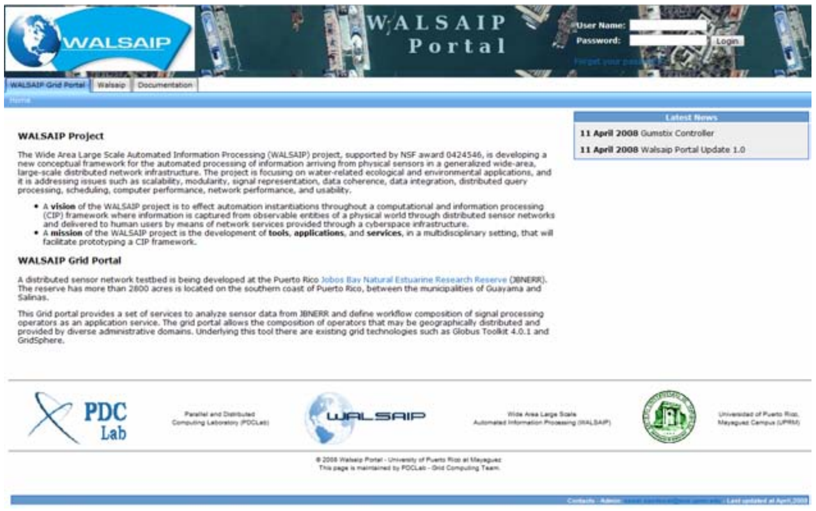
3.5 Acoustic Sensor Grid Interface

Another application portlet developed in this project was an acoustic sensor grid interface. This sensor manager interface controls gumstix sensors pertained to the environmental surveillance monitoring research of the WALSAIP (Wide Area Large Scale Automated Information Processing) team.

The particular application that the WALSAIP team is working on is to help biologists detect the endangered Puerto Rican Crested Toad *Peltophryne lemur*. The WALSAIP team has proposed an off-the-shelf distributed sensing infrastructure for the acoustic environmental surveillance monitoring problem based on the WALSAIP Sensor Grid (WSG). The acoustic-based WSG relies on a centralized architecture, i.e. a central or master node interacts with the lower nodes and is responsible for collecting, processing, and communicating the information observed to an end user or to a server.

For the problem of detecting the existence of *Peltophryne lemur* the WALSAIP team choose as the Master Sensor Node (MSN) module an embedded PC in mini-ITX, and as the Sensor Node modules Gumstix boards. The specifications of each module are described in detail in [38].

Currently the WALSAIP team is working on recording audio on the gumstix boards and managing the gumstix from the MSN. From this previous knowledge, the acoustic sensor grid interface was developed. First, a Gridsphere portal with the WALSAIP theme was developed and deployed on this MSN, as shown in the Figure 3.16. For this step, we use the automated package builder to deploy grid portals and development portlet-based interfaces previously described in Section 3.2. The automated package allowed us the development and deployment of the WALSAIP portal in an easy and fast way.



11 April 2008 Gumstix Controller

11 April 2008 Walsaip Portal Update 1.0

WALSAIP Project

The Wide Area Large Scale Automated Information Processing (WALSAIP) project, supported by NSF award 0424546, is developing a new conceptual framework for the automated processing of information arriving from physical sensors in a generalized wide-area, large-scale distributed network infrastructure. The project is focusing on water-related ecological and environmental applications, and it is addressing issues such as scalability, modularity, signal representation, data coherence, data integration, distributed query processing, scheduling, computer performance, network performance, and usability.

- A **vision** of the WALSAIP project is to effect automation instantiations throughout a computational and information processing (CIP) framework where information is captured from observable entities of a physical world through distributed sensor networks and delivered to human users by means of network services provided through a cyberspace infrastructure.
- A **mission** of the WALSAIP project is the development of **tools**, **applications**, and **services**, in a multidisciplinary setting, that will facilitate prototyping a CIP framework.

WALSAIP Grid Portal

A distributed sensor network testbed is being developed at the Puerto Rico Jobos Bay Natural Estuarine Research Reserve (JBNERR). The reserve has more than 2800 acres is located on the southern coast of Puerto Rico, between the municipalities of Guayama and Salinas.

This Grid portal provides a set of services to analyze sensor data from JBNERR and define workflow composition of signal processing operators as an application service. The grid portal allows the composition of operators that may be geographically distributed and provided by diverse administrative domains. Underlying this tool there are existing grid technologies such as Globus Toolkit 4.0.1 and GridSphere.

Parallel and Distributed Computing Laboratory (PDCLab)

Wide Area Large Scale Automated Information Processing (WALSAIP)

Universidad of Puerto Rico, Mayaguez Campus (UPRM)

© 2008 Walsaip Portal - University of Puerto Rico at Mayaguez
This page is maintained by PDCLab - Grid Computing Team.

Contacts - Admin: [illegible] - Last updated at April,2008

Figure 3.16: WALSAIP Grid Portal

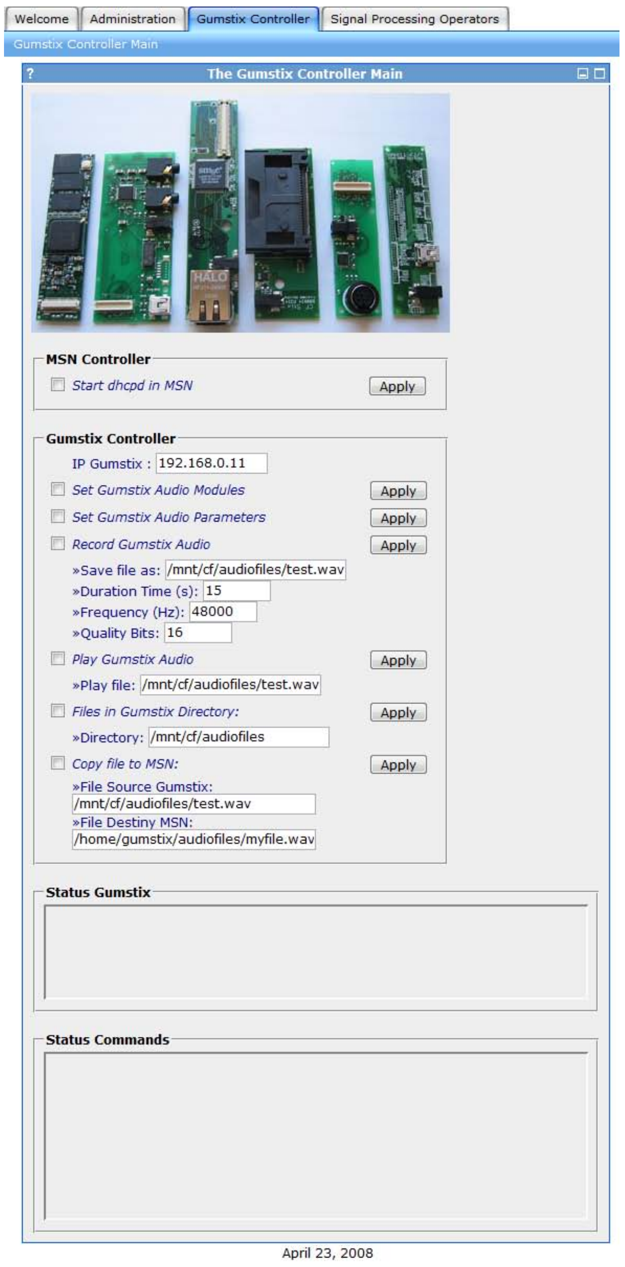
Next, we set up an SSH Public Key Authentication between the MSN and the gumstix nodes. The SSH authentication process allows the acoustic sensor grid interface to execute remote instructions directly in the sensor. To set up a public-key authentication: first we create an SSH key pair in the MSN, next we copy the public key to the gumstixs sensor, and finally we log into the gumstixs and installed the public key. The gumstixs sensor has an embedded Linux kernel version, therefore the installation of the SSH Public Key change compared to a non-embedded Linux kernel version such as the Radar Controller Interface case (See Section 3.3.3). The gumstixs authentication is described with more details in the Appendix B.

The portlet-based implementation was similar to the Radar Controller Interface described in Section 3.3.3. The users of this application can set the audio parameters of the gumstix sensors, and manage the audio files. Users that would like to use this gumstix interface only need a web browser. Knowledge of Linux environments or specific software are therefore not necessary, as all the connections and interactions with the gumstixs are transparent to the users. The developed Gumstix Controller Interface Portlet is illustrated in the Figure 3.17.

The interface developed allows the Gumstix setup in remote settings. Specifically the interface allows the user to set the Gumstix audio modules and its parameters. Therefore, each Gumstix can be activated to record audio data by setting the volume, sampling rate, duration of recording, bit resolution, and file format. Finally, the audio data can be displayed and transferred from each Gumstix to the MSN and stored in a local database. Two examples of the audio code executed directly in the gumstix sensor are shown below:

```
aumix -v100 -b0 -t100 -m100 -i100 -o100
brec -w -S s48000 b16 t15 audiofilename
```

The first line allows set the gumstix audio modules. The second line allows record an audio file for a sampling rate of 48000 Hz, 16 bit of resolution, recording length of 16 s, and the name of the audio file is `audiofilename`. Note that to execute this command first we need to provide a SSH connection between the MSN and the gumstix sensor. Therefore, the before commands can be executed for a user with a simple click on the portlet-based interface eliminating the knowledge of Linux commands.



MSN Controller

☐ *Start dhcpd in MSN*

Gumstix Controller

IP Gumstix : 192.168.0.11

☐ *Set Gumstix Audio Modules*

☐ *Set Gumstix Audio Parameters*

☐ *Record Gumstix Audio*

»Save file as: /mnt/cf/audiofiles/test.wav

»Duration Time (s): 15

»Frequency (Hz): 48000

»Quality Bits: 16

☐ *Play Gumstix Audio*

»Play file: /mnt/cf/audiofiles/test.wav

☐ *Files in Gumstix Directory:*

»Directory: /mnt/cf/audiofiles

☐ *Copy file to MSN:*

»File Source Gumstix: /mnt/cf/audiofiles/test.wav

»File Destiny MSN: /home/gumstix/audiofiles/myfile.wav

Status Gumstix

Status Commands

April 23, 2008

Figure 3.17: Gumstix Controller Interface Portlet

Chapter 4

Conclusions and Future Work

In this project report, we have presented the development of portlet-based sensor management interfaces that can be used to configure and control sensor instruments.

An automated package builder to deploy portlet-based sensor management interfaces was developed providing Java, Ant, Tomcat, and the Gridsphere Framework. The development of portlet interfaces was improved with the use of AJAX and Java Scripts. The components in the package can be used as a template to develop new grid portals and portlet-based interfaces.

Two portlet-based interfaces were developed. The first interface allows control the features of the PR1 radar. The second interface allows the management of Gumstix sensors pertaining to acoustic sensing applications.

A PR1 data detector was also develop with a 96% performance in rain data and 82% in no rain data. Also, an application portlet was developed to management PR1 data. End-users can access the raw-data acquired from the PR1 radar through the grid portal interface.

4.1 Future Work

The automated package builder to deploy sensor management interface can be improved with the develop of other interesting features to work on portlet-based interface with AJAX or Java Scripts as drag and drop features or request datasets from web services without refreshing the page. Furthermore, portlet applications can be developed and added to the grid portal to allow for example image manipulation, audio manipulation or maps location. Also, the portlet-based interface can be developed to manage other sensors as weather station or disdrometers.

The remote radar controller interface can be improved with the working on features of PR1 as the continuous acquisition data or with validation and calibration parameters.

The rain detector can be improved with others rain classes or by using other classificators such as neural networks, fuzzy classifier, nearest-neighbor classifiers or support vector machines.

The acoustic sensor grid interface can be improved with the management and control of the solar power supplies.

Appendix A

STB Installation and Administration Guide

A.1 Introduction

This guide shows how to use the GridSphere auto package builder and how administrate the STB Grid Portal developed under the PDC Lab at the University of Puerto Rico at Mayaguez. The package provides Java, Ant, Tomcat, and Gridsphere Framework. The current version of the STB portal is available at `http://stb.ece.uprm.edu`.

A.2 Software Dependencies

This auto-install package has the necessary software dependencies for the Gridsphere framework:

- **JDK 1.5.0.15:** The Java SE Development Kit.
- **Apache Ant 1.7.0:** Ant is used to compile and deploy the GridSphere framework.
- **Apache Tomcat 5.5.17:** The Servlet/JSP Container.
- **Gridshere 2.2.7-Release PDCLab:** The grid portal development framework that includes the STB service portlets.

A.3 Requirements

A Linux distribution. We recommend **CentOS** or **Fedora**.

A.4 Installation of GridSphere Auto Package Builder

1. Download the package `GridspherePDCLab.tar.gz`.

2. Extract the package at your `home` folder, e.g. `/home/myuser`. It needs to be installed only at this location.

3. Run the command `AutoInstallGridsphere.sh` at `/home/myuser/ GridspherePDCLab`, with the following syntax:

   ```
   ./GridspherePDCLab.sh
   ```

4. Then, read the Java License. The script will automatically make the installation of Java, Ant, and Tomcat, with their environment variables in the user system.

5. After a few minutes the installation completes and you will see the message **"Gridsphere Auto Installation Complete :)"**. After this message **close your terminal** and open a new one, in order to reload the new settings.

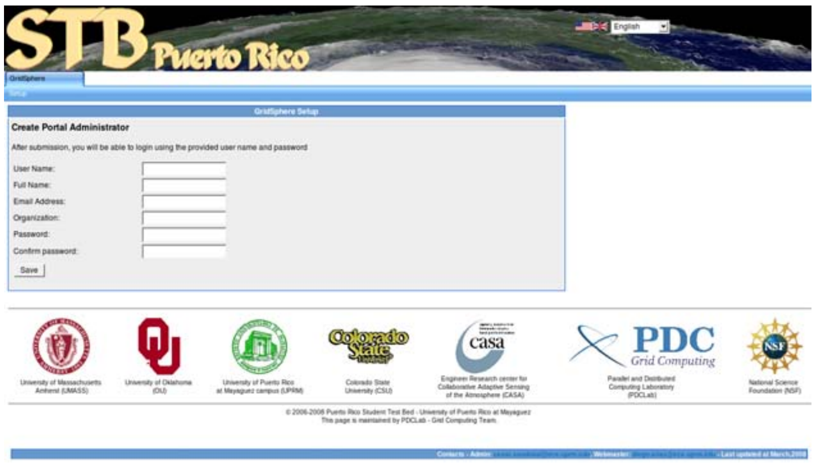
6. In the new terminal run the command `deployGridsphereAndPortlets.sh` at `/home /myuser/GridspherePDCLab`, with the following syntax:

   ```
   ./deployGridsphereAndPortlets.sh
   ```

   This script will automatically deploy the Gridsphere Framework and the provided Portlets at the Tomcat Container, and start up the Tomcat container.

7. Finally, the script will open your Web browser (deafault: Firefox) at the url `http: //localhost:8080/gridsphere` or `http://127.0.0.1:8080/gridsphere`. For the first time, you will see the Gridsphere Setup page as illustrated in Figure A.1.

   Review all required fields: user name, full name, email, organization (optional) and the password (the password must have at least 5 characters). Remember the entries for user and password because they will be assigned to the Gridsphere Admin user. Finally click at "save" and you will see the Gridsphere Portal Home Page, see Figure A.2.


STB Puerto Rico
English
GridSphere
Setup
Save
University of Massachusetts Amherst (UMASS)
University of Oklahoma (OU)
University of Puerto Rico at Mayaguez campus (UPRM)
Colorado State University (CSU)
Engineer Research center for Collaborative Adaptive Sensing of the Atmosphere (CASA)
Parallel and Distributed Computing Laboratory (PDCLab)
National Science Foundation (NSF)

GridSphere Setup

Create Portal Administrator

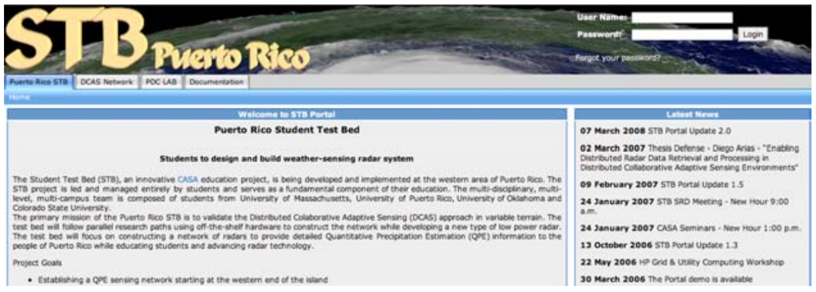
After submission, you will be able to login using the provided user name and password

User Name:	
Full Name:	
Email Address:	
Organization:	
Password:	
Confirm password:	

© 2006-2008 Puerto Rico Student Test Bed - University of Puerto Rico at Mayaguez
This page is maintained by PDCLab - Grid Computing Team.

Contacts - Admin: [illegible] - Webmaster: [illegible] - Last updated at March,2008

Figure A.1: STB Gridsphere Setup Page



Welcome to STB Portal

Puerto Rico Student Test Bed

Students to design and build weather-sensing radar system

The Student Test Bed (STB), an innovative CASA education project, is being developed and implemented at the western area of Puerto Rico. The STB project is led and managed entirely by students and serves as a fundamental component of their education. The multi-disciplinary, multi-level, multi-campus team is composed of students from University of Massachusetts, University of Puerto Rico, University of Oklahoma and Colorado State University.
The primary mission of the Puerto Rico STB is to validate the Distributed Colaborative Adaptive Sensing (DCAS) approach in variable terrain. The test bed will follow parallel research paths using off-the-shelf hardware to construct the network while developing a new type of low power radar. The test bed will focus on constructing a network of radars to provide detailed Quantitative Precipitation Estimation (QPE) information to the people of Puerto Rico while educating students and advancing radar technology.

Project Goals

- Establishing a QPE sensing network starting at the western end of the island

Latest News

07 March 2008 STB Portal Update 2.0

02 March 2007 Thesis Defense - Diego Arias - "Enabling Distributed Radar Data Retrieval and Processing in Distributed Collaborative Adaptive Sensing Environments"

09 February 2007 STB Portal Update 1.5

24 January 2007 STB SRD Meeting - New Hour 9:00 a.m.

24 January 2007 CASA Seminars - New Hour 1:00 p.m.

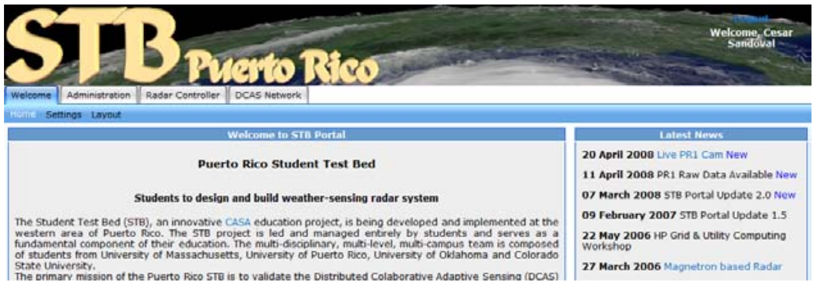
13 October 2006 STB Portal Update 1.3

22 May 2006 HP Grid & Utility Computing Workshop

30 March 2006 The Portal demo is available

Figure A.2: A part of the STB Gridsphere Portal Home Page

8. If the Gridsphere Portal is running, you can log in with your Admin user name, see Figure A.3.



Welcome to STB Portal

Puerto Rico Student Test Bed

Students to design and build weather-sensing radar system

The Student Test Bed (STB), an innovative CASA education project, is being developed and implemented at the western area of Puerto Rico. The STB project is led and managed entirely by students and serves as a fundamental component of their education. The multi-disciplinary, multi-level, multi-campus team is composed of students from University of Massachusetts, University of Puerto Rico, University of Oklahoma and Colorado State University.
The primary mission of the Puerto Rico STB is to validate the Distributed Colaborative Adaptive Sensing (DCAS)

Latest News

20 April 2008 Live PR1 Cam New

11 April 2008 PR1 Raw Data Available New

07 March 2008 STB Portal Update 2.0 New

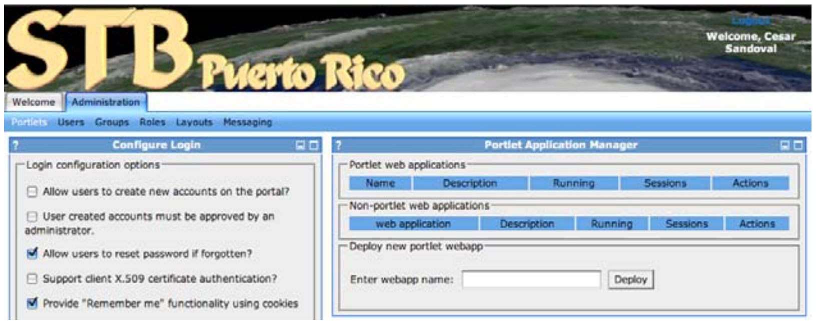
09 February 2007 STB Portal Update 1.5

22 May 2006 HP Grid & Utility Computing Workshop

27 March 2006 Magnetron based Radar

Figure A.3: A part of the STB Gridsphere Welcome Login Page

After logged in as the admin user, you can click the administration tab to access the administration portlet where you can configure the Gridsphere Framework, see Figure A.4.



Login configuration options

- ☐ Allow users to create new accounts on the portal?
- ☐ User created accounts must be approved by an administrator.
- ☑ Allow users to reset password if forgotten?
- ☐ Support client X.509 certificate authentication?
- ☑ Provide "Remember me" functionality using cookies

Portlet web applications

Non-portlet web applications

Deploy new portlet webapp

Enter webapp name:

Figure A.4: A part of the STB Gridsphere Administration Login Page

9. To access the portal from another computer in your web browser open the following URL: `http://<host-name>:8080/gridsphere`, where `<host-name>` is the host-name of the computer running the tomcat server. If any problem arises, first check your firewall configuration before debugging any further.

10. Starting and Stopping Tomcat Container

    - To start up the Tomcat (we provide a script at `GridspherePDCLab`) execute the following command:

      ```
      ./tomcat_start.sh
      ```

    - To stop Tomcat (we provide a script at `GridspherePDCLab`) execute the following command:

      ```
      ./tomcat_shutdown.sh
      ```

11. If you have any errors during the execution, please take a look at the Tomcat log messages at `$CATALINA_HOME/logs/`.

A.5 Starting and Stopping Tomcat Container at STB

The Tomcat container by default comes configured to run an HTTP server on port 8080. This means that every application in the container has the link `http://<host-name>:`

`8080/my-application`. In UNIX all the ports under 1024 are privileged ports and only the root may open these ports. For this reason the STB Grid Portal provide a script to startup and shutdown the tomcat container in the port 80, and thus removing the need for specifying the port in the URL. The script `jtomcat.sh` is located at `opt/tomcat/bin`. This script must be run as the root user in order to open port 80 and next spawn the tomcat container as the tomcat user.

- To start up the Tomcat in the STB Portal execute the following command:

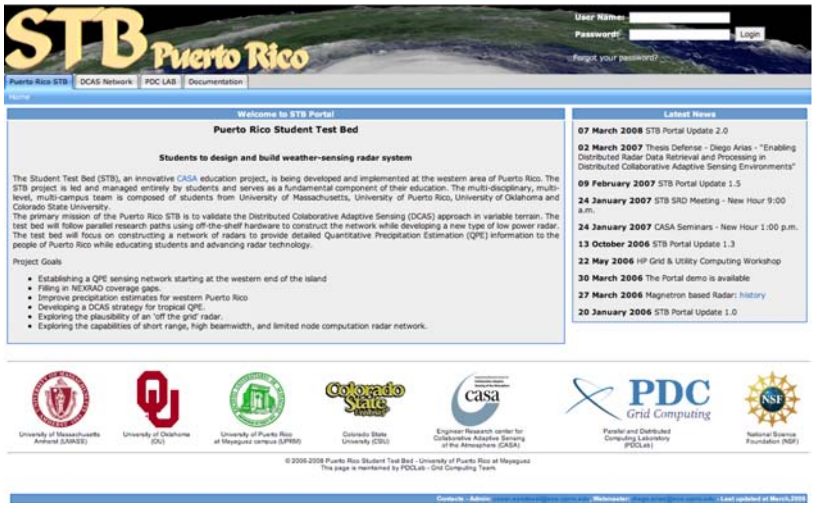
  ```
  ./jtomcat.sh start
  ```

- To stop Tomcat in the STB Portal execute the following command:

  ```
  ./jtomcat.sh stop
  ```

A.6 STB Portal

After starting the Tomcat container, you will be able to access the STB url of your container as `http://<host-name>`, see Figure A.5.


STB Puerto Rico
User Name:
Password:
Login
Forgot your password?
Puerto Rico STB
DCAS Network
PDC LAB
Documentation
Home
University of Massachusetts Amherst (UMASS)
University of Oklahoma (OU)
University of Puerto Rico at Mayaguez campus (UPRM)
Colorado State University (CSU)
Engineer Research center for Collaborative Adaptive Sensing of the Atmosphere (CASA)
Parallel and Distributed Computing Laboratory (PDCLab)
National Science Foundation (NSF)

Welcome to STB Portal

Puerto Rico Student Test Bed

Students to design and build weather-sensing radar system

The Student Test Bed (STB), an innovative CASA education project, is being developed and implemented at the western area of Puerto Rico. The STB project is led and managed entirely by students and serves as a fundamental component of their education. The multi-disciplinary, multi-level, multi-campus team is composed of students from University of Massachusetts, University of Puerto Rico, University of Oklahoma and Colorado State University.
The primary mission of the Puerto Rico STB is to validate the Distributed Colaborative Adaptive Sensing (DCAS) approach in variable terrain. The test bed will follow parallel research paths using off-the-shelf hardware to construct the network while developing a new type of low power radar. The test bed will focus on constructing a network of radars to provide detailed Quantitative Precipitation Estimation (QPE) information to the people of Puerto Rico while educating students and advancing radar technology.

Project Goals

- Establishing a QPE sensing network starting at the western end of the island
- Filling in NEXRAD coverage gaps.
- Improve precipitation estimates for western Puerto Rico
- Developing a DCAS strategy for tropical QPE.
- Exploring the plausibility of an 'off the grid' radar.
- Exploring the capabilities of short range, high beamwidth, and limited node computation radar network.

Latest News

07 March 2008 STB Portal Update 2.0

02 March 2007 Thesis Defense - Diego Arias - "Enabling Distributed Radar Data Retrieval and Processing in Distributed Collaborative Adaptive Sensing Environments"

09 February 2007 STB Portal Update 1.5

24 January 2007 STB SRD Meeting - New Hour 9:00 a.m.

24 January 2007 CASA Seminars - New Hour 1:00 p.m.

13 October 2006 STB Portal Update 1.3

22 May 2006 HP Grid & Utility Computing Workshop

30 March 2006 The Portal demo is available

27 March 2006 Magnetron based Radar: history

20 January 2006 STB Portal Update 1.0

© 2006-2008 Puerto Rico Student Test Bed - University of Puerto Rico at Mayaguez
This page is maintained by PDCLab - Grid Computing Team

Contacts - Admin: [illegible] | Webmaster: [illegible] - Last updated at March,2008

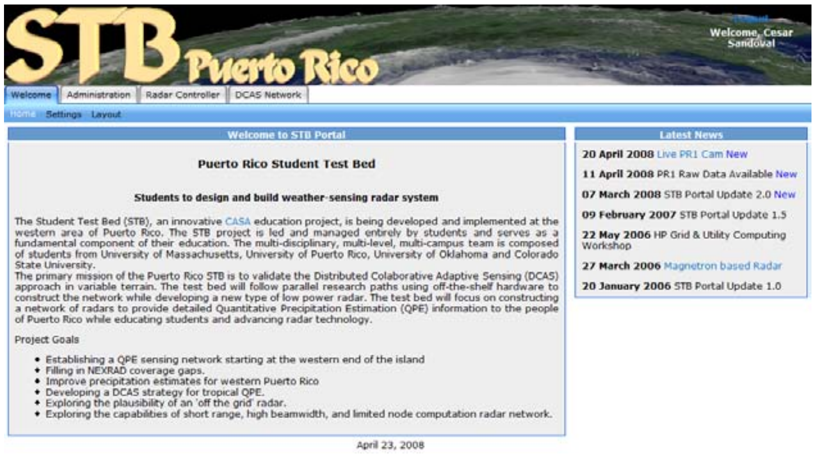
Figure A.5: STB Gridsphere Portal Home Page

A.7 STB Welcome Tab

When logged in with your user name and password you start in the "Welcome" tab. This provides three sub tabs: home, settings and, layout.

A.7.1 Home Sub tab

In this portlet we provide an abstract of the CASA project and show the last news about the portal, see Figure A.6.



Puerto Rico Student Test Bed

Students to design and build weather-sensing radar system

The Student Test Bed (STB), an innovative CASA education project, is being developed and implemented at the western area of Puerto Rico. The STB project is led and managed entirely by students and serves as a fundamental component of their education. The multi-disciplinary, multi-level, multi-campus team is composed of students from University of Massachusetts, University of Puerto Rico, University of Oklahoma and Colorado State University.
The primary mission of the Puerto Rico STB is to validate the Distributed Colaborative Adaptive Sensing (DCAS) approach in variable terrain. The test bed will follow parallel research paths using off-the-shelf hardware to construct the network while developing a new type of low power radar. The test bed will focus on constructing a network of radars to provide detailed Quantitative Precipitation Estimation (QPE) information to the people of Puerto Rico while educating students and advancing radar technology.

Project Goals

- Establishing a QPE sensing network starting at the western end of the island
- Filling in NEXRAD coverage gaps.
- Improve precipitation estimates for western Puerto Rico
- Developing a DCAS strategy for tropical QPE.
- Exploring the plausibility of an 'off the grid' radar.
- Exploring the capabilities of short range, high beamwidth, and limited node computation radar network.

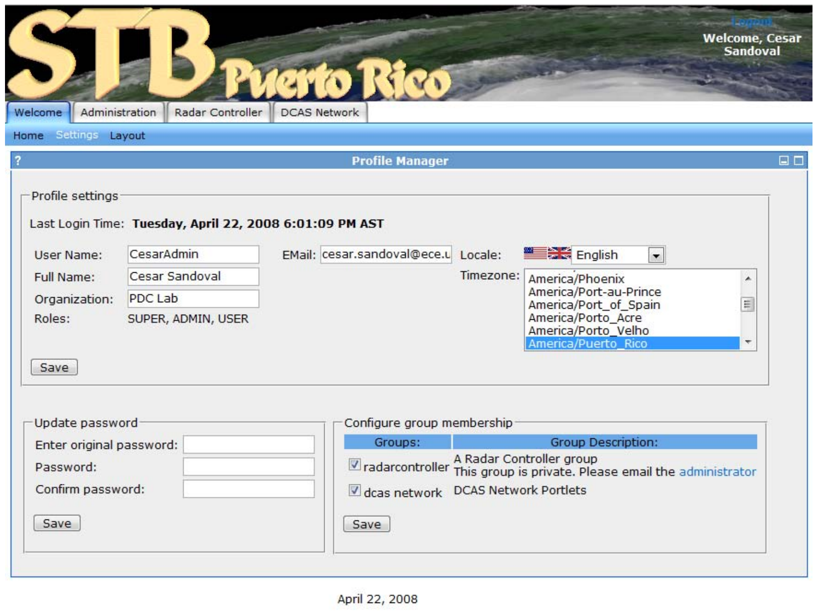
April 23, 2008

20 April 2008 Live PR1 Cam New
11 April 2008 PR1 Raw Data Available New
07 March 2008 STB Portal Update 2.0 New
09 February 2007 STB Portal Update 1.5
22 May 2006 HP Grid & Utility Computing Workshop
27 March 2006 Magnetron based Radar
20 January 2006 STB Portal Update 1.0

Figure A.6: STB Home Welcome portlet

A.7.2 Settings Subtab

The setting sub tab has a profile manager portlet. The Profile Manager portlet allows users to customize their settings including profile information, language preferred, time zone, and group membership. The view mode displays the existing user profile information that can be edited. The password of the user can be changed in this portlet. The configure group membership section show all the application portlets available, the user can select which applications would like for his profile, see Figure A.7.



Profile Manager

Profile settings

Last Login Time: **Tuesday, April 22, 2008 6:01:09 PM AST**

User Name:	CesarAdmin	EMail:	cesar.sandoval@ece.u	Locale:	English
Full Name:	Cesar Sandoval			Timezone:	America/Phoenix America/Port-au-Prince America/Port_of_Spain America/Porto_Acre America/Porto_Velho America/Puerto_Rico
Organization:	PDC Lab				
Roles:	SUPER, ADMIN, USER				

Save

Update password

Enter original password:

Password:

Confirm password:

Save

Configure group membership

Groups:	Group Description:
✓ radarcontroller	A Radar Controller group This group is private. Please email the administrator
✓ dcas network	DCAS Network Portlets

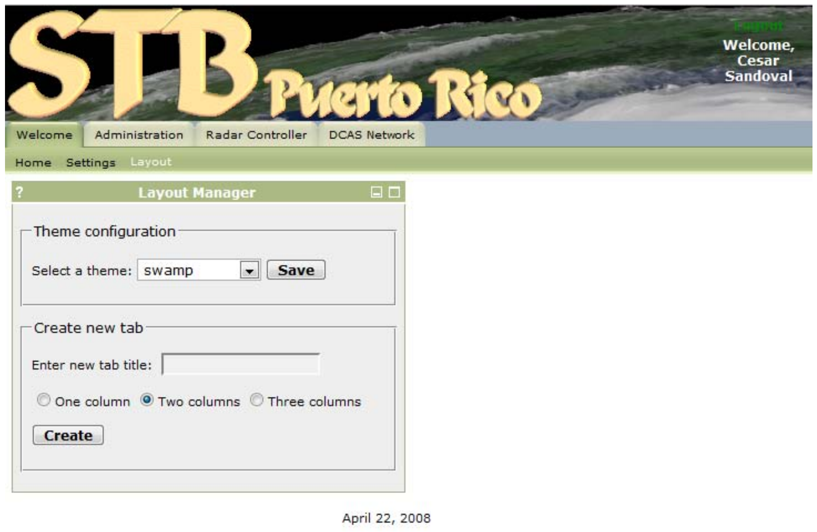
Save

April 22, 2008

Figure A.7: STB Home Settings portlet

A.7.3 Layout Subtab

The layout sub tab is a layout manager portlet. The Layout Manager portlet allows users to customize their profile theme and tabs. Various options are provided to allow for renaming, creating, and deleting portlet tab titles, see Figure A.8.



Layout Manager

Theme configuration

Select a theme: swamp Save

Create new tab

Enter new tab title:

One column [✓ Two columns] Three columns

Create

April 22, 2008

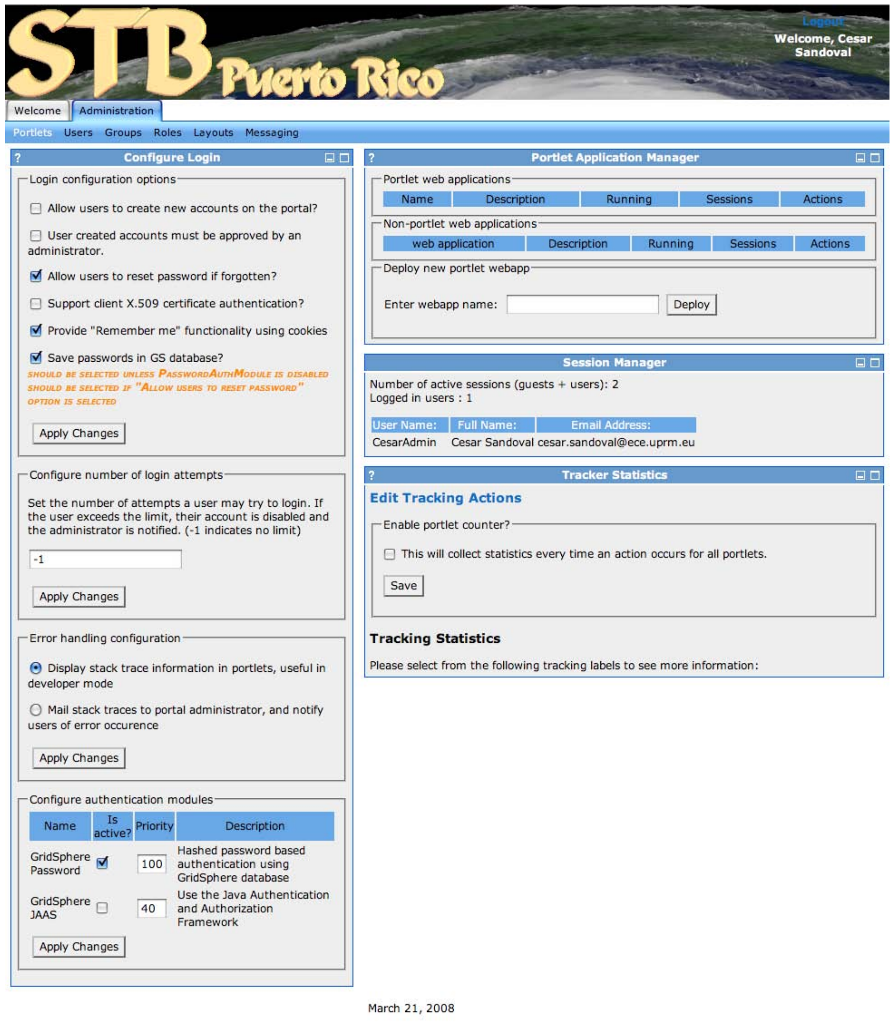
Figure A.8: STB Swamp Layout

A.8 STB Administration Tab

After login as admin user, you can click the administration tab to configure the Gridsphere Framework features. The Administration tab provides six sub tabs: portlets, users, groups, roles, layouts and, messaging.

A.8.1 Portlets Subtab

The Portlets sub tab has four sections: configure login, portlet application manager, session manager and, tracker statistics, see Figure A.9.



Configure Login

Login configuration options

- [ ] Allow users to create new accounts on the portal?
- [ ] User created accounts must be approved by an administrator.
- [✓] Allow users to reset password if forgotten?
- [ ] Support client X.509 certificate authentication?
- [✓] Provide "Remember me" functionality using cookies
- [✓] Save passwords in GS database?

SHOULD BE SELECTED UNLESS PASSWORDAUTHMODULE IS DISABLED
SHOULD BE SELECTED IF "ALLOW USERS TO RESET PASSWORD" OPTION IS SELECTED

Apply Changes

Configure number of login attempts

Set the number of attempts a user may try to login. If the user exceeds the limit, their account is disabled and the administrator is notified. (-1 indicates no limit)

-1

Apply Changes

Error handling configuration

- [✓] Display stack trace information in portlets, useful in developer mode
- [ ] Mail stack traces to portal administrator, and notify users of error occurence

Apply Changes

Configure authentication modules

Name	Is active?	Priority	Description
GridSphere Password	✓	100	Hashed password based authentication using GridSphere database
GridSphere JAAS		40	Use the Java Authentication and Authorization Framework

Apply Changes

Portlet Application Manager

Portlet web applications

Name	Description	Running	Sessions	Actions

Non-portlet web applications

web application	Description	Running	Sessions	Actions

Deploy new portlet webapp

Enter webapp name: Deploy

Session Manager

Number of active sessions (guests + users): 2
Logged in users : 1

User Name:	Full Name:	Email Address:
CesarAdmin	Cesar Sandoval	cesar.sandoval@ece.uprm.eu

Tracker Statistics

Edit Tracking Actions

Enable portlet counter?

- [ ] This will collect statistics every time an action occurs for all portlets.

Save

Tracking Statistics

Please select from the following tracking labels to see more information:

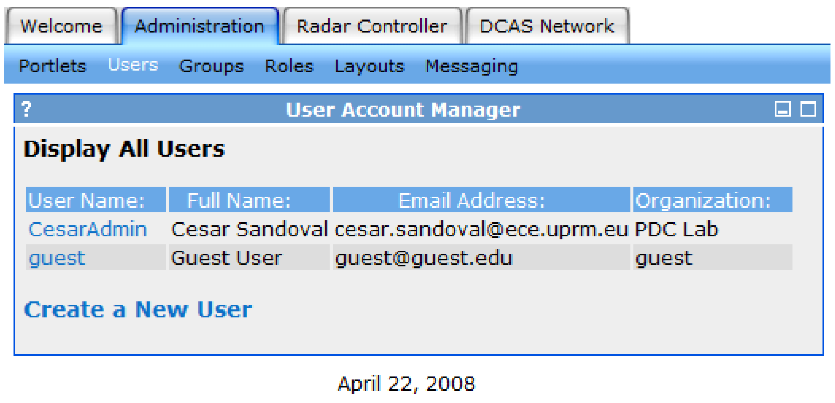
March 21, 2008

Figure A.9: STB Administration Portlets

The configure login section allows for creating new accounts, changing password, setting the number of user login in the portal. The Portlet Application Manager allows a portal administrator to stop, start and deploy portlet web applications to GridSphere. Once a portlet has been stopped it is no longer accessible. A new portlet web application may be deployed but it assumes that the web application archive (WAR) has already been placed into the Tomcat webapps directory. The session manager portlet shows all the information of current log users. Finally, the tracker statistics portlet allows for tracking the application portlets.

A.8.2 Users Subtab

The Users portlet is used for managing portal users. The portlet displays all the information of the portal users and allows editing this information, see Figure A.10.



Display All Users

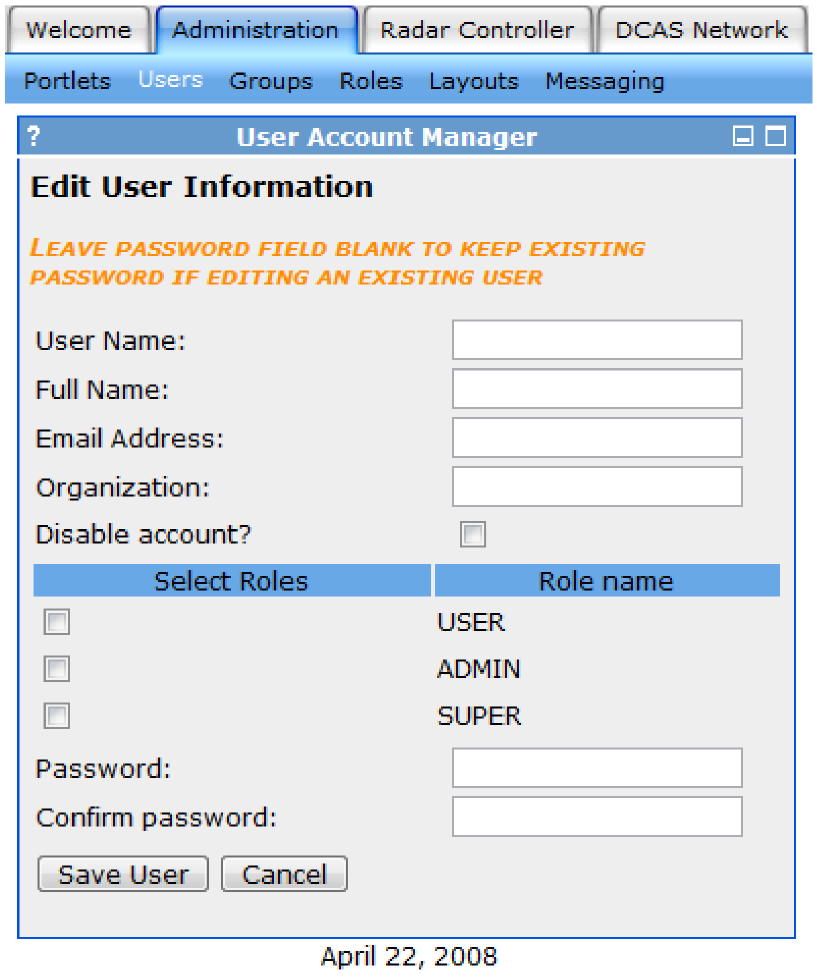
User Name:	Full Name:	Email Address:	Organization:
CesarAdmin	Cesar Sandoval	cesar.sandoval@ece.uprm.eu	PDC Lab
guest	Guest User	guest@guest.edu	guest

Create a New User

April 22, 2008

Figure A.10: STB Administration Users

The portlet also allows the creation of new users. Figure A.11 shows the form for a new user. Username, full name, email and password (must have at least 5 characters) are required fields, organization is optional. Admin must also select the roles for the new user. Once an account is created the new portal user may login with the provided password. Leave password field blank to keep existing password if editing an existing user.



Edit User Information

LEAVE PASSWORD FIELD BLANK TO KEEP EXISTING PASSWORD IF EDITING AN EXISTING USER

User Name:	
Full Name:	
Email Address:	
Organization:	
Disable account?	☐

Select Roles	Role name
☐	USER
☐	ADMIN
☐	SUPER

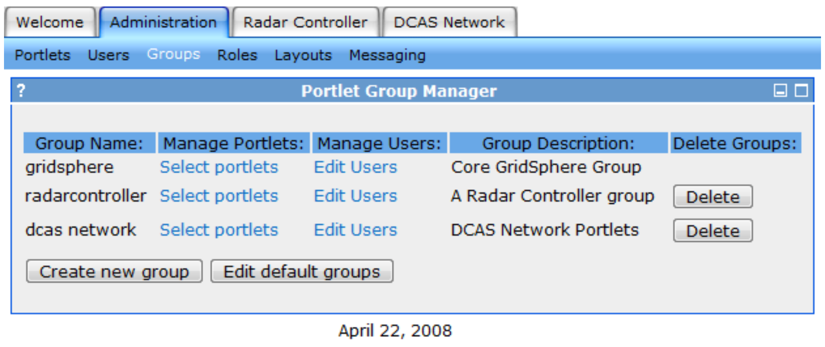
Password:	
Confirm password:	

April 22, 2008

Figure A.11: STB Administration New User Account

A.8.3 Groups Subtab

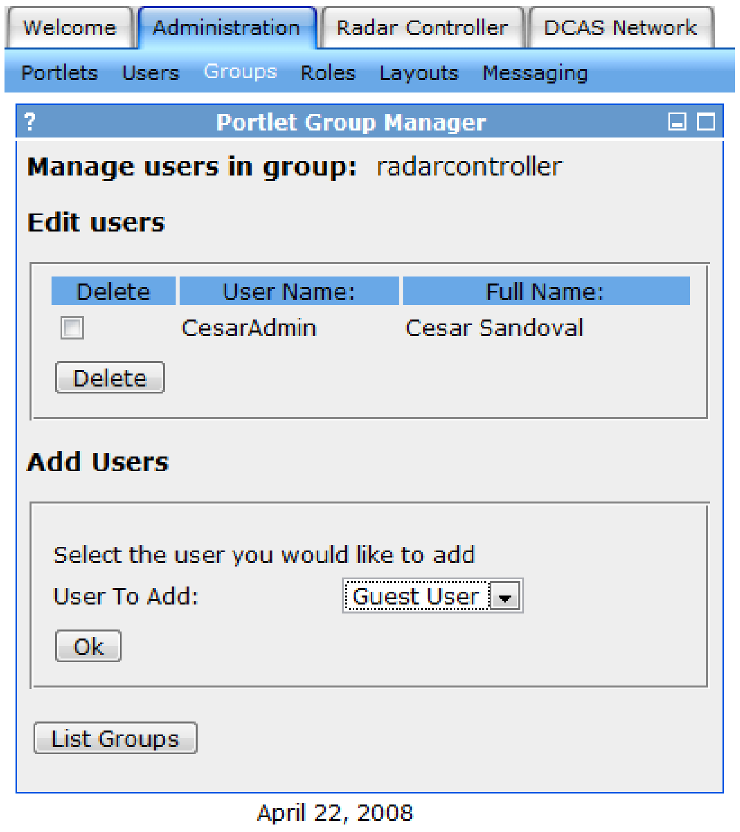
The Groups portlet displays all the groups available, see figure A.12. It is also used for adding and removing portal users to groups. A group is defined by a deployed portlet web application. New users are initially in the core GridSphere group but may be added to existing groups. Simply select the group to add or remove users from and then confirm changes, as shown in Figure A.13. Remember that the user can select in her profile the public applications (like DCAS Network) but the private application (like Radar Controller) only can be added by the administrator in this portlet. In this portlet the admin can create new groups with the portlet applications available in others groups.



Group Name:	Manage Portlets:	Manage Users:	Group Description:	Delete Groups:
gridsphere	Select portlets	Edit Users	Core GridSphere Group	
radarcontroller	Select portlets	Edit Users	A Radar Controller group	Delete
dcas network	Select portlets	Edit Users	DCAS Network Portlets	Delete

April 22, 2008

Figure A.12: STB Administration Groups



Manage users in group: radarcontroller

Edit users

Delete	User Name:	Full Name:
☐	CesarAdmin	Cesar Sandoval

Add Users

Select the user you would like to add

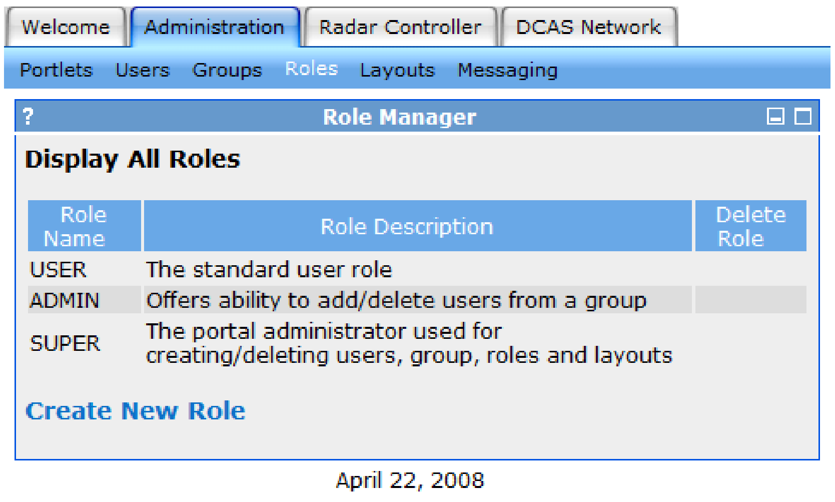
User To Add: Guest User

April 22, 2008

Figure A.13: STB Administration Users in Groups

A.8.4 Roles Subtab

The Roles portlet is used for managing roles in the portal. Roles can be added, modified and deleted except for the core GridSphere roles, USER, ADMIN and SUPER that also match the core role priority levels supported by the portal. A User role priority is used for any logged user, Admin is generally used for some kind of administrator functionality and super essentially has no access restrictions. Roles can also be defined by an individual webapp by specifying them in the role descriptor file at `WEB-INF/roles.xml`. Figure A.14 shows the administration roles portlet.



Display All Roles

Role Name	Role Description	Delete Role
USER	The standard user role	
ADMIN	Offers ability to add/delete users from a group	
SUPER	The portal administrator used for creating/deleting users, group, roles and layouts	

Create New Role

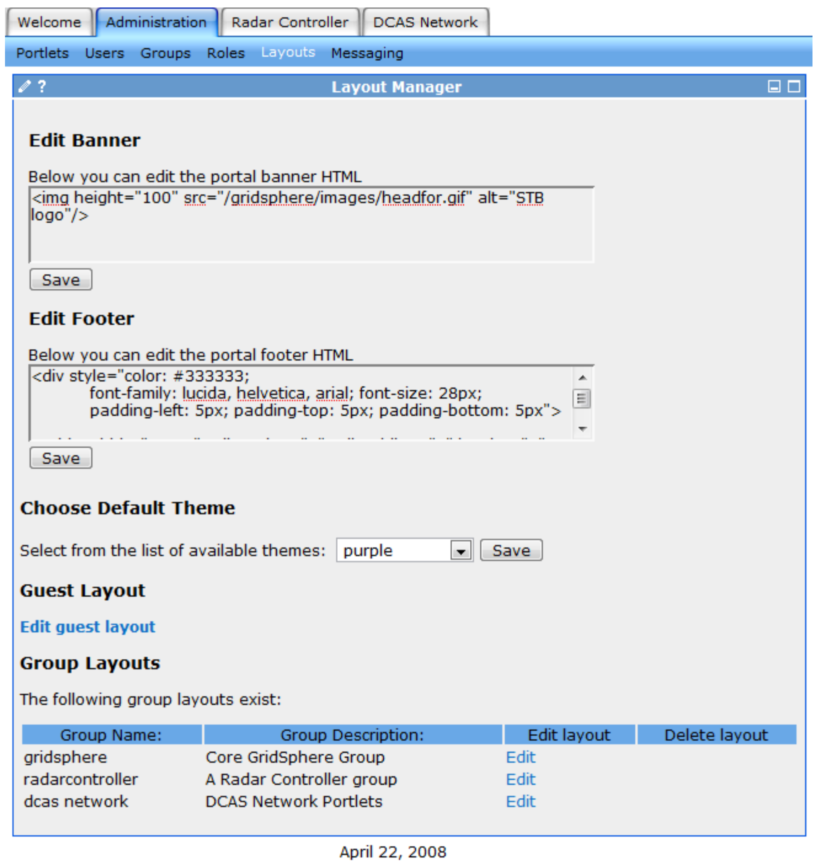
April 22, 2008

Figure A.14: STB Administration Roles

A.8.5 Layouts Subtab

The Layouts portlet allows customizing the theme and entire portal layout. Various options are provided to allow rename, create and delete portlet tab titles and sub tab titles. Portlets within a subtab can be arranged into table layouts with the desired columns and rows. Figure A.15 shows the layout manager portlet.

The layout manager portlet allows editting the banner and footer sections. It also provides a section to choose the default theme of the outside of the grid portal and edit the `GuestUserLayout.xml` file which has the configuration of the outside tabs, as shown in figure A.16.



Edit Banner

Below you can edit the portal banner HTML

```
<img height="100" src="/gridsphere/images/headfor.gif" alt="STB
logo"/>
```

Save

Edit Footer

Below you can edit the portal footer HTML

```
<div style="color: #333333;
        font-family: lucida, helvetica, arial; font-size: 28px;
        padding-left: 5px; padding-top: 5px; padding-bottom: 5px">
```

Save

Choose Default Theme

Select from the list of available themes: purple Save

Guest Layout

Edit guest layout

Group Layouts

The following group layouts exist:

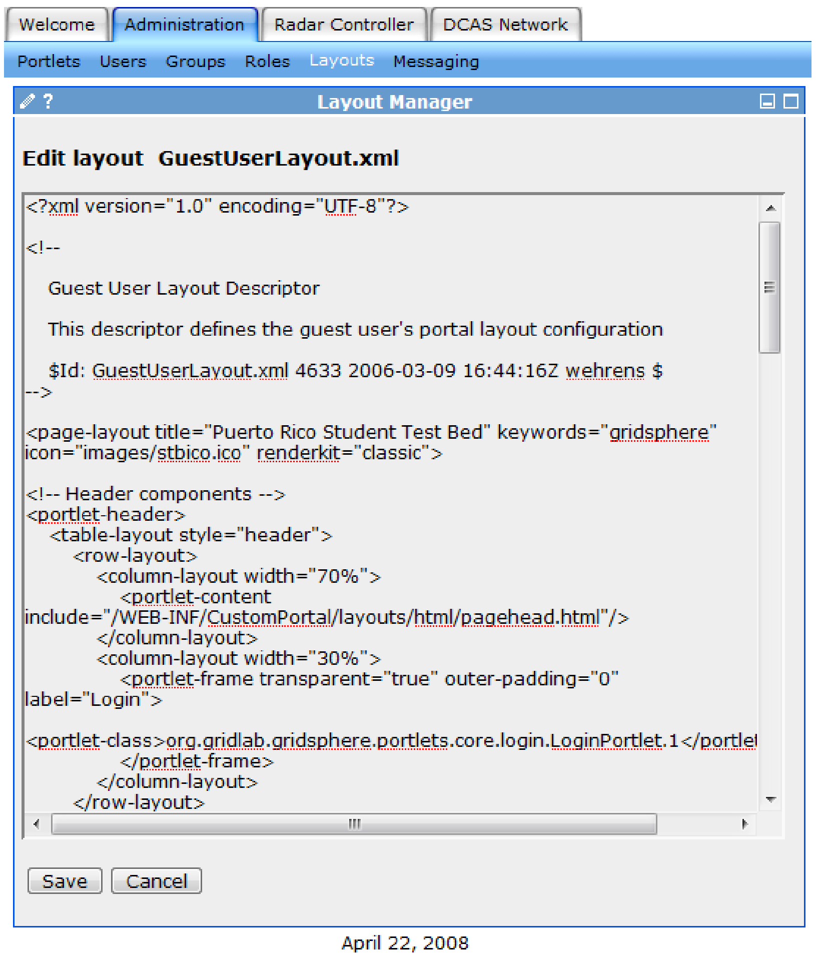
Group Name:	Group Description:	Edit layout	Delete layout
gridsphere	Core GridSphere Group	Edit	
radarcontroller	A Radar Controller group	Edit	
dcas network	DCAS Network Portlets	Edit	

April 22, 2008

Figure A.15: STB Administration Layouts

A.8.6 Messaging Subtab

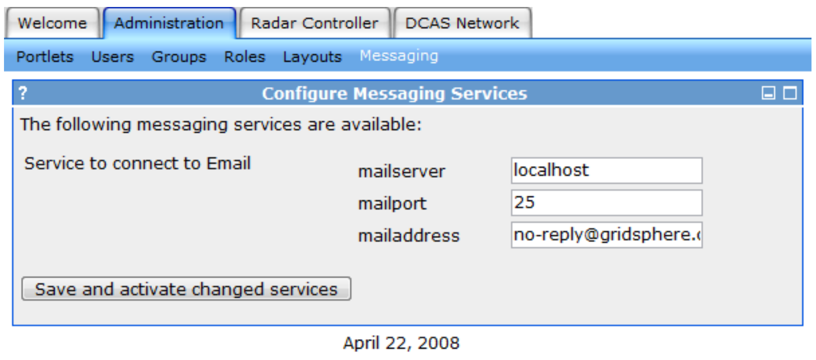
Messaging is the last sub tab of the administration option and allows the configuration of messaging services. This portlet allows editting server, port, and address of the mailing service. The information here concerning to grid portal alerts, for example when you forgot your password. Figure A.17 shows the configure messaging services portlet.



Edit layout GuestUserLayout.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!--

    Guest User Layout Descriptor

    This descriptor defines the guest user's portal layout configuration

    $Id: GuestUserLayout.xml 4633 2006-03-09 16:44:16Z wehrens $
-->

<page-layout title="Puerto Rico Student Test Bed" keywords="gridsphere"
icon="images/stbico.ico" renderkit="classic">

<!-- Header components -->
<portlet-header>
    <table-layout style="header">
        <row-layout>
            <column-layout width="70%">
                <portlet-content
include="/WEB-INF/CustomPortal/layouts/html/pagehead.html"/>
            </column-layout>
            <column-layout width="30%">
                <portlet-frame transparent="true" outer-padding="0"
label="Login">

<portlet-class>org.gridlab.gridsphere.portlets.core.login.LoginPortlet.1</portlet
                </portlet-frame>
            </column-layout>
        </row-layout>
```

Figure A.16: STB Administration GuestUserLayout



The following messaging services are available:

Service to connect to Email

mailserver	localhost
mailport	25
mailaddress	no-reply@gridsphere.

Figure A.17: STB Administration Messaging

Appendix B

Setting Up Public Key Authentication over SSH

This appendix shows how to use public-key authentication to prove your identity to a remote server instead of login and password.

1. **Generate key on local machine**

   To set up public-key authentication, first create an SSH key pair, executing the following command at `/home/myuser/.ssh`:

   ```
   ssh-keygen -b 1024 -t dsa
   ```

   After executing this command, make the `passphrase` empty in order to connect to the server instead password.

   This created two files in `.ssh` folder: `id_dsa` and `id_dsa.pub`. An example of a cryptographic key is following showed:

   id_dsa.pub

   ```
   ssh−dss AAAAB3NzaC1kc3MAAACBAJQte1m+ztwwQGk12J7Ud27JHw2gFC+8
   qJBCIgHCyiiLonvK/Uxjfd7sV272WDVgQ2BtB1KFZwLuZamdxXGVHj420oXQ
   6JWA69yYuoSLqskiVKjRpwJdt82uatGoxoIRsyc7+teksUNXqR9Ceds/wgBI
   unH+D5Df0v7Dto5zDJX7AAAAFQDCNjk2u+Vf90SPdzv9Gt+BzwrPiQAAAIAR
   5eWJWgiWOeBD5Z38v2nXmOP9jj+cYNLXBLSzHx6giUUeaF5bszy6168c0j71
   G2G+QJuop5pnu96JtcfIIM7RKxAVgmKUcSF/qGgvUIsC3eq4PzBPZy+I7ZW8
   jD2h23rsQZfzyKrJtL9gZHTLuiUFS+1NIBKPmu0bW9c4qrAjtwAAAIB1KQxi
   ```

```
0wmLAWy9WNOOBPerQOImsV8F7d/AvHCOfS17R/8UXUnk04y6GVaTMrTSxBVj
ALWuXewfSxojD1j6AISIPWHof1zwh6yNdySOrYcgqO+gVgp5fDpfJs6h6N0r
j22cPBr9IhPCrP45v9GfO27tBcAf62S50ir9QVYfsH9uUg
== myuser@<HOST-NAME>
```

2. **Copy the public key to the remote server**

   Copy the local public key to the remote server using password authentication executing the following command:

   ```
   scp id_dsa.pub remoteuser@remoteserver:
   ```

3. **Log into the remote server and install the public key**

   Log into the remote server using password authentication executing the following command:

   ```
   ssh remoteuser@remoteserver
   ```

   On the remote server, create the `/home/remoteuser/.ssh` directory if it does not already exist and set its mode appropriately, as follow:

   ```
   mkdir -p ~/.ssh
   chmod 700 ~/.ssh
   ```

   Then append the contents of `id_dsa.pub` to `~/.ssh/authorized_keys2` executing the following command:

   ```
   cat id_dsa.pub >> ~/.ssh/authorized_keys2    (Appending)
   chmod 600 ~/.ssh/authorized_keys2
   ```

   Note: *Always append the public key into* `authorized_keys2`, *because multiple people can be access the server. Each line in the* `authorized_keys2` *file on the server corresponds to a user who can get into the account from a remote host.*

   Note: *SSH public keys go into the file* `~/.ssh/authorized_keys2`. *Other versions of SSH, however, require SSH-1 protocol keys to be in* `~/.ssh/authorized_keys`, *like the* **gumstix boards.**

   Log out of the remote server and log back in.

4. **Now ssh to the remote server**

   Now you can ssh to the remote server without entering your password.

Bibliography

[1] J. Brotzge, D. Westbrook, K. Brewster, K. Hondl, and M. Zink, "The meteorological command and control structure of a dynamic, collaborative, automated radar network," 2006.

[2] "The GRIDCC project home page." [Online]. Available: http://www.gridcc.org

[3] H. B. Lim and D. Li, "Data management services for sensor grids," *Consumer Communications and Networking Conference, CCNC 2007*, pp. 609–613, 2006.

[4] "Instrument middleware project portal." [Online]. Available: http://www.instrumentmiddleware.org

[5] D. F. McMullen, I. M. Atkinson, K. Chiu, P. Turner, K. Huffman, R. Quilici, and M. Wyatt, "Toward standards for integration of instruments into grid computing environments," *2nd IEEE International Conference on e-Science and Grid Computing (e-Science 2006)*, 2006.

[6] D. Arias, *Enabling Distributed Radar Data Retrieval and Processing in Distributed Collaborative Adaptive Sensing Environments*. M.S. Thesis - University of Puerto Rico, Mayagüez, 2006.

[7] M. A. Vega, *Student Developed Meteorological Radar Network for the Western Part of Puerto Rico: First Node*. M.S. Thesis - University of Puerto Rico, Mayagüez, 2007.

[8] "Security Systems from Coastal Video Security, inc." http://www.covisec.com.

[9] "Collaborative adaptive sensing of the atmosphere (CASA) engineering research center - overview," 2003. [Online]. Available: http://www.casa.umass.edu

[10] M. Zink, D. Westbrook, S. Abdallah, B. Horling, V. Lakamraju, E. Lyons, V. Manfredi, J. Kurose, and K. Hondl, "Meteorological command and control: an end-to-end architecture for a hazardous weather detection sensor network," in *EESR '05: Proceedings of the 2005 workshop on End-to-end, sense-and-respond systems, applications and services*, 2005, pp. 37–42.

[11] "Open grid forum," http://www.ogf.org.

[12] I. Foster, "What is the grid? a three point checklist," *Grid Today*, vol. 1, no. 6, p. 22, 2002.

[13] J. Treadwell, "Open grid services architecture glossary of terms," in *Technical report, Global Grid Forum*, 2005.

[14] "Hewlett-Packard," http://www.hp.com.

[15] "The gridsphere portal framework." [Online]. Available: http://www.gridsphere.org

[16] C. Wege, "Portal server technology," *IEEE Internet Computing*, vol. 6, no. 3, pp. 73–77, 2002.

[17] "Jsr 168 portlet specification," http://jcp.org/en/jsr/detail?id=168.

[18] I. Foster and C. Kesselman, Eds., *The Grid: Blueprint for a Future Computing Infrastructure*. Morgan-Kaufmann, 1999.

[19] "The globus alliance." [Online]. Available: http://www.globus.org

[20] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," *Technical report, Open Grid Service Infrastructure WG, Global Grid Forum*, 2002.

[21] E. Frizziero, M. Gulmini, F. Lelli, G. Maron, A. Oh, S. Orlando, A. Petrucci, S. Squizzato, and S. Traldi, "Instrument element: a new grid component that enables the control of remote instrumentation," *Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops (CCGRIDW'06)*, 2006.

[22] "VAST - introduction to SensorML." [Online]. Available: http://vast.uah.edu/SensorML

[23] G. Aloisio, D. Conte, C. Elefante, G. P. Marra, G. Mastrantonio, and G. Quarta, "Globus monitoring and discovery service and SensorML for grid sensor networks," *Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'06)*, pp. 201–206, 2006.

[24] "SensorGrid." [Online]. Available: http://www.crisisgrid.org/html/sensorgrid.html

[25] I. Atkinson, D. du Boulay, C. Chee, K. Chiu, P. Coddington, A. Gerson, T. King, D. McMullen, R. Quilici, P. Turner, A. Wendelborn, M. Wyatt, , and D. Zhang, "Developing CIMA based remote access for collaborative e-research," *5th Australasian Symposium on Grid Computing and e-Research (AusGrid 2007)*, Ballarat. January, 2007.

[26] D. P. Zhang, L. S. He, and H. Yang, "Grid based integration technologies of virtual measurement system," *Journal of Physics: Conference Series 48*, pp. 1258–1263, 2006.

[27] T. Basten, L. Benini, A. Chandrakasan, M. Lindwer, J. Liu, R. Min, and F. Zhao, "Scaling into ambient intelligence," *Design, Automation and Test in Europe Conference and Exhibition (DATE'03)*, vol. 01, p. 10076, 2003.

[28] D. Estrin, "Wireless sensor networks: application driver for low power distributed systems," in *ISLPED '01: Proceedings of the 2001 international symposium on Low power electronics and design*, 2001, pp. 194–194.

[29] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-centric storage in sensornets," *ACM SIGCOMM Computer Communication*, vol. 33, no. 1, pp. 137–142, 2003.

[30] V. Groza, V. Cretu, M. Bogoevici, and E. Petriu, "Distributed virtual instrumentation architecture," *Sensors for Industry Conference (Sicon'01)*, pp. 177–181, 2001.

[31] B. R. Kumar, K. Sridharan, and K. Srinivasan, "The design and development of a web-based data acquisition system," *Instrumentation and Measurement*, vol. 51, no. 3, pp. 427–432, 2002.

[32] J. Novotny, M. Russell, and O. Wehrens, "Gridsphere: an advanced portal framework," *Euromicro Conference, 2004. Proceedings. 30th*, pp. 412–419, 2004.

[33] D. D. Vecchio, V. Hazlewood, and M. Humphrey, "Evaluating grid portal security," *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, pp. 1–14, 2006.

[34] H. B. Lim, Y. M. Teo, P. Mukherjee, V. T. Lam, W. F. Wong, and S. See, "Sensor grid: Integration of wireless sensor networks and the grid," in *LCN '05: Proceedings of the The IEEE Conference on Local Computer Networks 30th Anniversary*, 2005, pp. 91–99.

[35] D. du Boulay, C. Chee, K. Chiu, R. Leow, D. F. McMullen, R. Quilici, and P. Turner, "Remote instrument control with CIMA web services and web 2.0 technology," *Workshop on Remote control of Devices (WoRD) (in conjunction with ICDIM'07, and in Cooperation with ACM SIGAPP.fr)*, yon, France, October 28–31, 2007.

[36] I. Rao, E. Huh, and S. Lim, "An adaptive and efficient design and implementation for meteorology data grid using grid technology," in *WETICE '06: Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE Computer Society, 2006, pp. 239–246.

[37] S. Padmanabhuni, K. Kunti, L. Sahoo, and S. Bharti, "Coupling RDF/RSS, WSRP and AJAX for dynamic reusable portlets: An approach and a use case," in *IEEE SCW*, 2007, pp. 175–182.

[38] Y. Yunes, *Acoustic Signal Representation for Environmental Surveillance Monitoring (ESM)*. M.S. Thesis - University of Puerto Rico, Mayagüez, 2007.