



**THE AUSTRALIAN NATIONAL UNIVERSITY**

**FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY  
DEPARTMENT OF COMPUTER SCIENCE**

# **Web Services Architecture for a Fusion Data Grid**

**Client & Supervisor: Dr. Henry Gardner**

**Supervisor: Dr. Raju Karia**

**Student: Xiaobin Wang**

(u4266538)

March 2008

This is the final report for the COMP8770 eScience Project (Semester 1, 2008)

# TABLE OF CONTENTS

<b>ACKNOWLEDGMENTS.....</b>	<b>4</b>
<b>ABSTRACT.....</b>	<b>5</b>
<b>1. Introduction.....</b>	<b>6</b>
1.1 Overview.....	6
1.1.1 Background.....	6
1.1.2 Purpose.....	8
1.1.3 Statement of Scope.....	8
1.1.4 Target Results and Project Customers.....	9
1.1.4.1 Target Outcomes.....	9
1.1.4.2 Outputs.....	9
1.1.4.3 Deliverables.....	9
1.1.4.4 Customers.....	9
1.1.4.5 Stakeholders.....	10
1.2 Introduction to WebScope3.....	11
1.3 Introduction to WebScope4.....	13
1.3.1 Software structure evolvement.....	13
1.3.1.1 Implementing Jini in WebScope4.....	13
1.3.1.2 Implementing CAS in WebScope4.....	14
1.3.1.3 Separating the Metadata part from WebScope4.....	14
1.3.2 The main benefits of WebScope4.....	15
1.4 Report Overview.....	17
<b>2. Requirement Analysis.....</b>	<b>18</b>
2.1 Software Requirements.....	18
2.2 Operating System.....	18
2.3 Languages.....	18
2.4 Client Requirements.....	19
<b>3. Scheduling.....</b>	<b>20</b>
3.1 Planned Timetable.....	20
3.2 Actual Progress.....	21
3.3 Gantt Charts.....	22
<b>4. Modeling.....</b>	<b>23</b>
4.1 Structure of WebScope4.....	23
4.2 High Level Design.....	23
4.2.1 System Context.....	23
4.2.2 Domains.....	23
4.3 Detailed Design.....	27
4.3.1 Communication Between Packages.....	27
4.3.2 Database Design.....	27
4.3.3 WebScope4 Domain.....	28
4.3.4 dataAccessDomain Domain.....	30

4.3.5 dataServerDomain Domain.....	31
4.3.6 graphicsDomain Domain.....	31
4.3.7 Mapping Domain.....	33
4.3.8 Applet Domain.....	33
4.3.9 JiniServer Domain.....	34
4.3.10 metaJini Domain.....	35
<b>5. Implementation.....</b>	<b>36</b>
5.1 Sub phases of the project.....	36
5.2 Lines of code.....	36
5.3 Implementation issues.....	37
<b>6. Extensibility Study.....</b>	<b>40</b>
6.1 A description of the WebScope4 data grid.....	40
6.2 Including another dataset in the grid.....	41
6.2.1 Interface definition.....	41
6.2.2 Design pattern requirement for the new domain.....	43
6.3 Programming a customized client for the grid.....	43
6.3.1 MDSPlus Jini client specification.....	43
6.3.2 Metadata Jini client specification.....	44
<b>7. Testing.....</b>	<b>46</b>
7.1 Unit Test.....	46
7.2 Module Test.....	46
7.2.1 WebScope4 client.....	46
7.2.2 Jini-based MDSPlus Server.....	49
7.2.3 MetaScope.....	49
7.2.4 EScope Jini Client.....	50
7.3 Acceptance Test.....	50
<b>8. Summary of Contributions and Future Work.....</b>	<b>51</b>
8.1 Summary of Contributions.....	51
8.2 Future Work.....	52
<b>9. Reference.....</b>	<b>53</b>
<b>Appendix A: Deployment Guide.....</b>	<b>55</b>
<b>Appendix B: User Manual.....</b>	<b>60</b>

## **Acknowledgements**

During the entire project, I have received much invaluable help from my supervisor, my clients and my mates. I shall express my sincere gratefulness to the persons below:

Dr. Henry Gardner

Dr. Raju Karia

Mr. Ajith M. Jose

Mr. Ian Wood

I cannot imagine completing the project without their help.

I shall show my special gratitude to my supervisor, Dr. Henry Gardner, as he has been helping me in every possible way. He is always patient and ready to help, which enables me to get everything clear and find ways to resolve the problems and troubles I met during the project process.

## Abstract

This report outlines the development of a web services architecture for a fusion data grid for the eScience project course COMP8770, at the Australian National University.

The core idea behind this project is retrieving nuclear fusion datasets from an MDSPlus based data server using a web browser. The use of Jini technology to make the MDSPlus servers and WebScope clients find each other automatically is a highlight of this project.

The main achievements of this project are as follows:

- Jini technology has been implemented in the WebScope4 system. The WebScope4 clients can now find a MDSPlus server without knowing the server name and other details about the server.
- The CAS single sign-on system has been integrated into the WebScope4 system to authenticate users.
- The existing HSQLDB database has been migrated to a MySQL database, which is faster, more secure and more efficient.
- A Jini client version of EScope has been created to improve the flexibility of the Data Grid.
- A standalone metadata managing program MetaScope has been developed. The metadata database has been separated from the original database.
- A “collaboration space” which enables more than two users to work on the same dataset and experimental results at the same time has been added to the WebScope4 system.

# 1 Introduction

## 1.1 Overview

### 1.1.1 Background

Currently nuclear fusion research is being performed at various locations of the world. Millions of nuclear fusion datasets are generated as a result of various nuclear fusion experiments happening at different research organizations. It is very important for the researchers to share the datasets for mutual research benefits. The safe storage and retrieval of fusion datasets is currently done using a data storage and management system known as “MDSPlus” [4].

For research purposes, several students supervised by Dr. Henry Gardner have worked on “Scope projects” over the past two years. By the end of Semester 1 2007, there were two versions of Scope: “EScope” and “WebScope”. The latter is considered to be a web-based version of the former, with the intention of enhancing the universality of its usage.

#### **eScience Data Grid:**

In general terms, a Data Grid can be thought of as a loose federation of networked data stores which is supported by grid computing [1].

This concept has evolved because with the development of scientific and engineering applications, the need to access large amounts of distributed data has become more and more obvious.

#### **Previous WebScope Systems:**

The project “WebScope” is a project to resolve the existing issues with the retrieval of datasets from the MDSPlus server using EScope. Another highlight of “WebScope” project is the use of Java object relational mapping solution “Hibernate” to solve the caching issues with “EScope” data retrieval system and to enable metadata to be cached and saved dynamically.

At the time of the commencement of this project, there were three versions of WebScope available. Their genealogy is shown in the figure below (Figure 1.1):

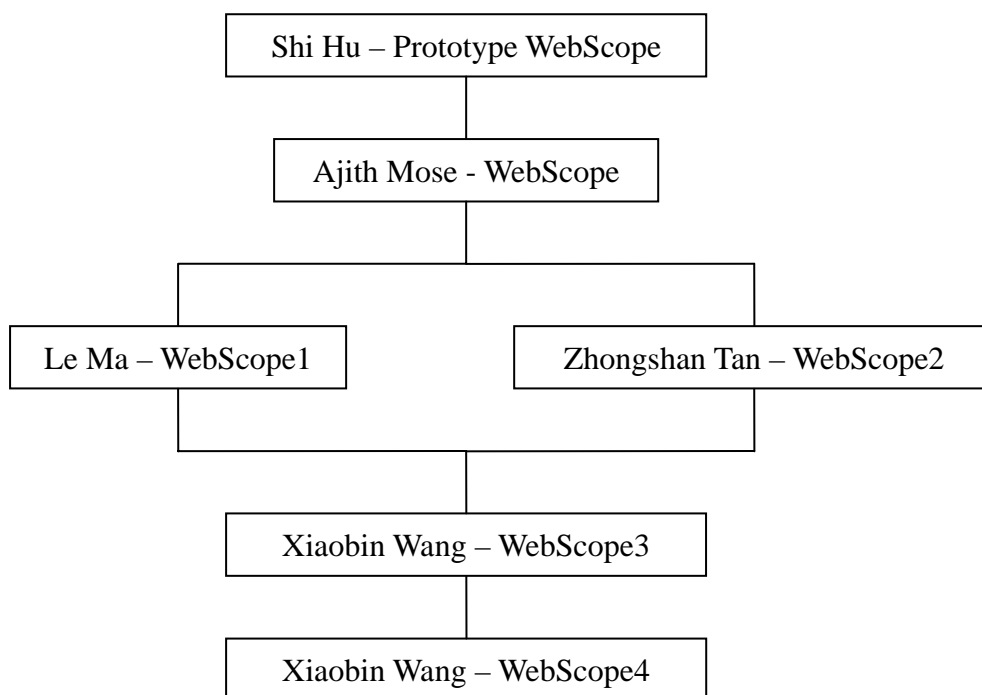


Figure 1.1 The Genealogy of WebScope

#### WebScope4

As the eScience community is growing, the need for making a web portal more available to scientists is obvious, which means that the web portal should make it very easy for a scientist to find required data and share his/her own data. Although the latest version of the WebScope system, which is WebScope3, is providing scientists a very convenient way to access data through web browsers, it still requires users to specify which MDSPlus server to connect and port, experiment, shot information about which scientists may not have any idea. In addition, the current WebScope systems are not providing users with any secure method to manage their user profiles.

What is more important is that users should be provided with a working environment other than just a web application. The environment should include everything users may need for his experiment or study. Users may also want to share their own data with others and collaborate with scientists all over the world. This environment is quoted as a WebScope grid here, which includes standalone MDSPlus servers, standalone Metadata servers, web clients, and locally install clients.

Having taken the reasoning above into account, Dr. Henry Gradner has decided to introduce an updated version of WebScope system with Jini & CAS, which is called WebScope4. I shall briefly describe Jini and CAS below:

**Jini:** Jini is a service oriented architecture that enables the construction of distributed systems consisting of network services. We can make almost everything on the web a Jini service. Jini services can automatically register themselves with Jini lookup services, which will be automatically found by Jini clients. [7]

**CAS:** CAS stands for Central Authentication Service. In my project, the main significance of CAS is that it can provide a means of centrally managing user profiles, which allows users to have access to all authorized data with just one sign-on process. [8]

### **1.1.2 Purpose**

The main purposes of the current project are as follows:

- Implement Jini in the WebScope4 system, including a Jini client for the WebScope4 client and a Jini server for the MDSPlus server. In addition, a Jini lookup service should also be set up.
- Implement CAS in the WebScope4 system to replace the existing logon sub-system. In addition, I will cooperate with Mr. Ajith Mose who is doing a project on CAS project designing.
- Replace the existing HSQLDB database with a MySQL database.
- Separate the metadata database from the existing database. Make a standalone user client to manage the metadata. WebScope4 clients are supposed to find the metadata servers with Jini technology.
- Enable users to create collaboration spaces which let them share some private experimental data and dynamic tables.

### **1.1.3 Statement of scope**

The scope of the current project is not limited to adding features on the basis of the current system. Some modification to the whole structure of the system may be necessary.

## **1.1.4 Target results and project customers**

### **1.1.4.1 Target outcomes**

The “Web services architecture for a Fusion DataGrid” project has five target outcomes:

- Implementing Jini in the current WebScope4 system.
- Implementing CAS in the WebScope4 system to improve the security.
- Migrating from the current HSQLDB database to a MySQL database.
- Creating a web service to centrally control the access and modifying of metadata.
- Collaboration space

### **1.1.4.2 Outputs**

The target outputs for this “Web Services Architecture for a Fusion DataGrid” project are the working “WebScope4” application and the final project report.

### **1.1.4.3 Deliverables**

Deliverables are a superset of the outputs obtained as a result of the project. The project deliverables are listed below:

- Source code of the WebScope4 application with all the intended changes
- Final report of the system
- All the other documents related to the project

### **1.1.4.4 Customers**

The project customers who are vital in ensuring the success of achieving target outcomes are nuclear fusion researchers or scientists from different parts of the globe.

#### 1.1.4.5 Stakeholders

The stakeholders of this project are described below.

<b>StakeHolder</b>	<b>Name</b>	<b>Contact Details</b>
Supervisor & Client	Dr. Henry Gardner	<a href="mailto:Henry@cs.anu.edu.au">Henry@cs.anu.edu.au</a>
Supervisor	Dr. Raju Karia	<a href="mailto:rjkaria@smartchat.net.au">rjkaria@smartchat.net.au</a>
Developer	Xiaobin Wang	<a href="mailto:tommywang1981@gmail.com">tommywang1981@gmail.com</a>

## 1.2 Introduction to WebScope3

“WebScope3” is the system proposed by Dr. Henry Gardner and realised by following the project I completed in S2/2007 at ANU. Figure 1.2 gives an overview of the WebScope3 system.

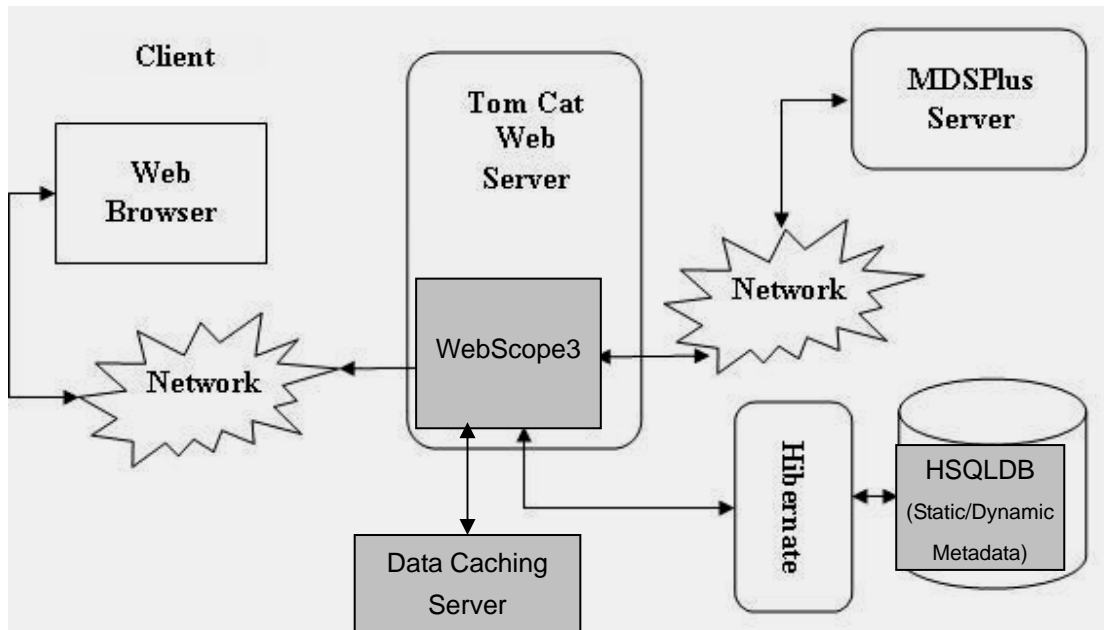


Figure 1.2 High level structure of WebScope3 system

“WebScope3” uses Java, Apache Tomcat and the Cooe graphical framework to develop web-based applications.

### The main benefits of using “WebScope3” are given below

- The interaction between the clients and servers are greatly improved because of the Cooe technology. We do not need to reload the whole page when users request a small change to a part of it. Because of the Ajax-based rendering engine, only the requested part of a page is updated per users’ requests.
- In the period of development, the developer does not have to think in terms of "page-based" applications and is able to develop applications using the conventional object-oriented and event-driven paradigm for user interface development.
- Users can not only read experiment data from the database, but also contribute static/dynamic metadata to the server, which makes the application more

interactive, flexible and functional. For the dynamic metadata feature, users can customize their own tables according to their requirements. New Java files and Hibernate mapping files are generated for the customized tables and then the Java files are compiled and the Java classes are loaded in runtime. Afterwards, the table content is saved into database. This gives users more control on the system.

- Users have two modes to plot the graph data now: the Image Map mode and the Java Applet mode. The former is quicker and simpler and the latter is more functional and interactive. It's up to the users to decide which mode suits them best. The two modes make the WebScope3 system more flexible.

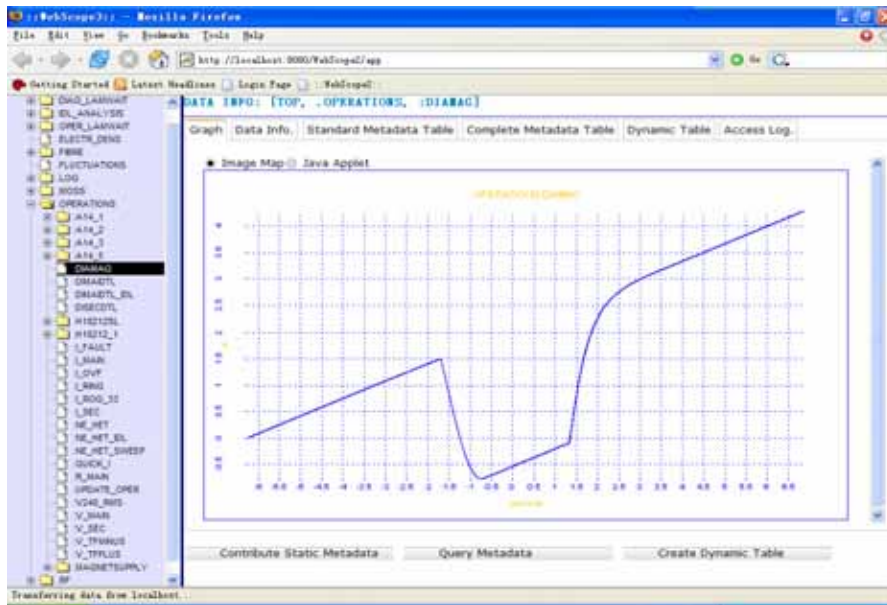


Figure 1.3 The interface of the WebScope3 system

## 1.3 Introduction to WebScope4

Although as of the end of 2007, the WebScope3 system could already provide users with a relatively easy portal to access eScience data, some user requirements had still not been addressed yet. For example, users will have to remember the address of the MDSPlus servers to obtain the required data. The authentication part of the WebScope3 system is not quite secure as user names and passwords were stored in plain text. In addition, the HSQLDB database was not efficient and as the database gets larger, it will be slower and slower to start up the database.

To resolve all these problems, Dr. Henry Gardner proposed the new WebScope4 system.

### 1.3.1 Software structure evolution

#### 1.3.1.1 Implementing Jini in WebScope4

The first step of the development of the WebScope4 system is to make the MDSPlus server a Jini server and the WebScope4 client a jinni client. They can communicate with each other via Jini Lookup Services.

The figure below (Figure 1.4) gives an overview of the new WebScope4 system after this step:

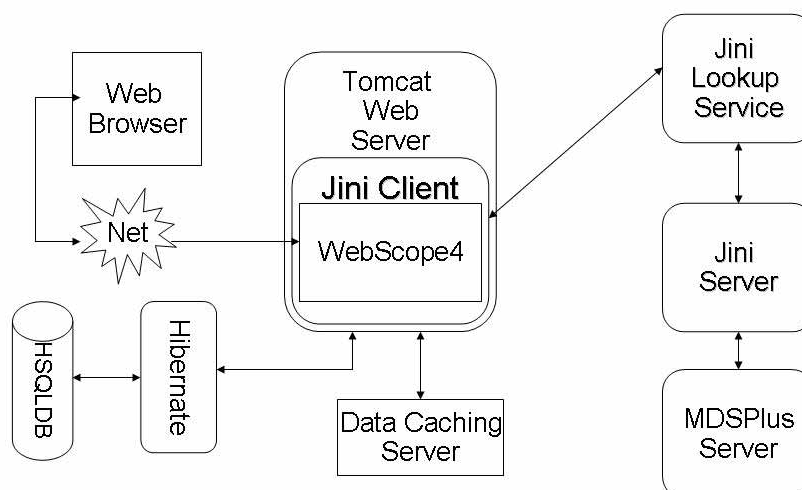


Figure 1.4 High level structure of WebScope4 after step 1

### 1.3.1.2 Implementing CAS in WebScope4

The second step of the development of the WebScope4 system is to use the CAS system to authenticate users. The figure below (Figure 1.5) illustrates the structure of the WebScope4 system after this step:

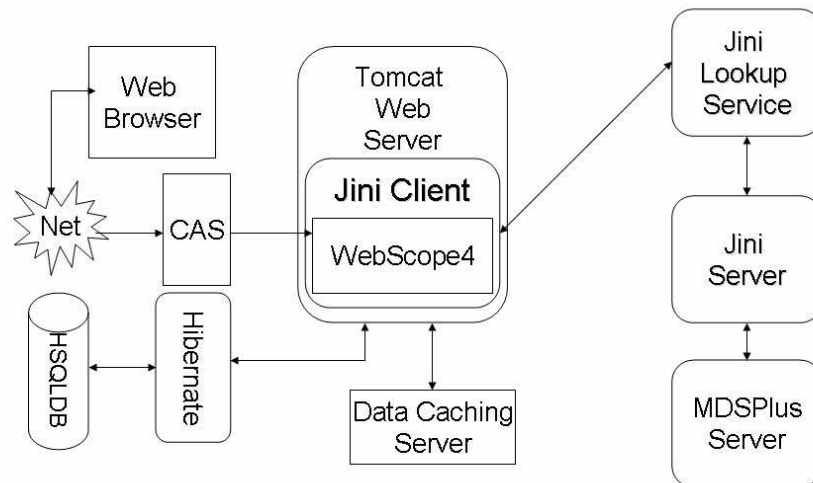


Figure 1.5 High level structure of WebScope4 after step 2

### 1.3.1.3 Separating the Metadata part from WebScope4

The third step is to make the Metadata part of the WebScope4 system a stand alone service. The new service, which is named MetaScope, communicates with the WebScope4 client using the Jini technology. In this step, the existing HSQldb database is replaced with a MySQL database, too. The figure below (Figure 1.6) illustrates the structure of the WebScope4 system after this step:

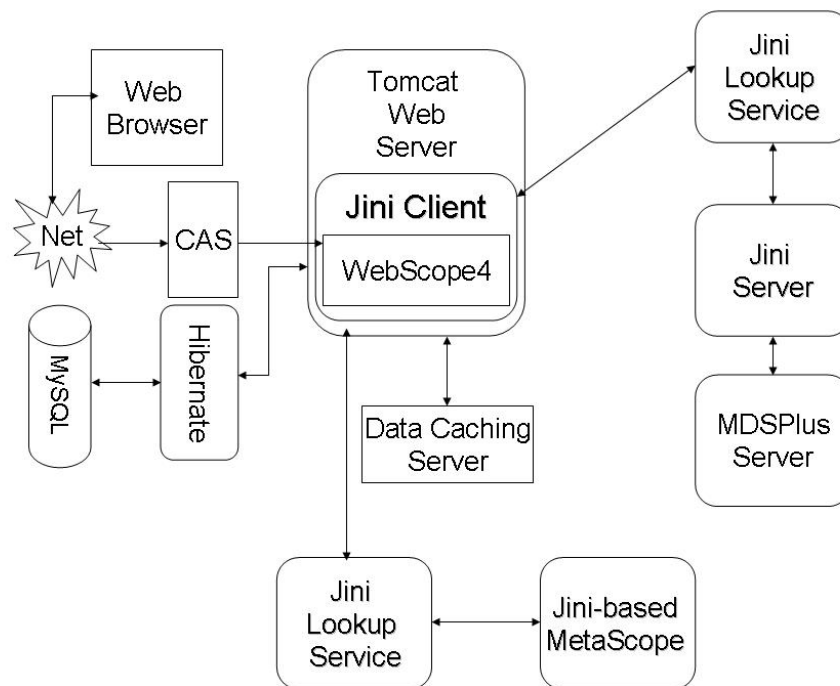


Figure 1.6 High level structure of WebScope4 after step 3

### 1.3.2 The main benefits of “WebScope4”

- All the benefits provided by the original WebScope3 system.
- WebScope clients can now find MDSPPlus servers automatically, without requiring users to input the server details.
- The CAS system provides users with the Single-Sign-On feature, which means that users only need to log on once during an attempt to access the WebScope system.
- As the HSQLDB database has been replaced with the MySQL database, the overall performance of database reading and writing has been greatly improved. In addition, the security and stability of the system has been improved as well.
- The metadata part of the WebScope4 system is now a standalone web service. This makes it easier to manage the security and authorization of metadata. In addition, this also makes the topology of the WebScope4 grid more flexible.
- Users can now create private working space, which is called Collaboration Space, in the WebScope4 system. This space allows more than two users to work on the same experimental data and results at the same time. The owner of the space can authorize other users to access the space. For those who are not authorized, the

space is not accessible. All the information in a Collaboration space can be synchronized to public Dynamic Metadata table.

- Please refer forward to Figure 6.1 for an overview of the WebScope4 data grid.

## 1.4 Report Overview

This report is divided into 11 major sections.

- Section 1, Introduction, gives an overview of the project, as well as the expected outcomes, outputs and the major deliverables of the project. Through this section the customers and major stakeholders of the project are identified.
- Section 2, Requirement Analysis, describes detailed user requirements, softwares and other tools needed to accomplish this project.
- Section 3, Scheduling, describes the various phases involved in this project and the number of days allotted for each phase.
- Section 4, Modelling, also known as Designing, uses UML model to design the structure of this software. It also defines the database structure.
- Section 5, Implementation, describes the ideas and methods used to achieve the functionalities of “WebScope” application with Jini & CAS.
- Section 6, Testing, tests the whole system to ensure that it’s working properly.
- Section 7, Future Work, gives suggestions on further development.
- Section 8, Conclusion, draws a conclusion from this project.
- Section 9, Reference List, shows the various references used for accomplishing the “WebScope” application with Jini & CAS.

## **2. Requirement Analysis**

### **2.1 Software Requirements**

The main software and tools used for the development of “Web services architecture for Fusion DataGrid” are given below:

- Apache TomCat 5.5
- Hibernate3.0
- JDK1.6
- MySQL
- MDSPlus server
- FireFox & Internet Explorer
- Eclipse 3.3.0
- Cooe Framework 1.0.2
- Jini 2.1
- CAS Server 3.2
- CAS Client 2.0.11

### **2.2 Operating system**

The WebScope4 application is platform-independent. All we need to get this application working is a browser with Java plug-in.

### **2.3 Languages**

- Programming Language: Java combined with Ajax, Java Servlet, Java Applet
- Database Querying Language: HQL (Hibernate Query Language)

## 2.4 Client Requirements

No.	Req. Name	Description	Priority
R1	Implement Jini technology in the WebScope system	“WebScope” client should be able to find MDSPlus servers automatically with the Jini technology.	High
R2	Implement CAS in the WebScope system	“WebScope” should use the CAS service to authenticate users.	High
R3	Migrate from HSQLDB to MySQL	“WebScope” should migrate from the current HSQLDB to a MySQL database.	High
R4	Creating a web service to centrally manage metadata	A standalone web service should be created to allow users manage metadata. The service should communicate with “WebScope” client with the Jini technology.	High
R5	Creating a Collaboration Space	The Collaboration Space should allow more than two users to work on the same experimental data and results at the same time. The owner of a space should be able to control the authorization and publication.	High

Table 2.1 Major requirements of “WebScope4”

### 3. Scheduling

#### 3.1 Planned timetable

The initial planned timetable is as below: (Table 3.1)

Phase	Date	Expected Duration(days)	Tasks	Notes
1	25 <sup>th</sup> Feb ~ 16 <sup>th</sup> Mar	21	<ul style="list-style-type: none"> <li>✓ Understanding requirements</li> <li>✓ Studying the new techniques</li> <li>✓ Studying the current Web Scope system</li> </ul>	<ul style="list-style-type: none"> <li>● Choose and meet supervisor</li> <li>● Understand requirements</li> <li>● Decide tools and install software on the computer</li> <li>● Study old reports</li> </ul>
2	17 <sup>th</sup> Mar ~ 6 <sup>th</sup> Apr	21	Modeling	<ul style="list-style-type: none"> <li>● Analyze existing project</li> <li>● Modeling and design architecture for the new project</li> <li>● Learn various software</li> </ul>
3	7 <sup>th</sup> Apr~ 11 <sup>th</sup> May	35	Implementation	<ul style="list-style-type: none"> <li>● Do coding, Testing and Debugging</li> </ul>
4	12 <sup>th</sup> May~ 25 <sup>th</sup> May	14	Documentation and final report creation	<ul style="list-style-type: none"> <li>● Complete all documents</li> <li>● Write the final report</li> </ul>
5	26 <sup>th</sup> May ~ 1 <sup>st</sup> Jun	7	Preparation for final presentation	<ul style="list-style-type: none"> <li>● Edit slides for final presentation</li> </ul>

Table 3.1 Planned Timetable for “Web services architecture for a Fusion DataGrid” project

### 3.2 Actual progress

The actual timetable is as below: (Table 3.2)

Phase	Date	Expected Duration(days)	Tasks	Notes
1	25 <sup>th</sup> Feb ~ 2 <sup>nd</sup> Mar	7	<ul style="list-style-type: none"> <li>✓ Understanding requirements</li> <li>✓ Studying the new techniques</li> <li>✓ Studying the current Web Scope system</li> </ul>	<ul style="list-style-type: none"> <li>● Choose and meet supervisor</li> <li>● Understand requirements</li> <li>● Decide tools and install software on the computer</li> <li>● Study old reports</li> </ul>
2	3 <sup>rd</sup> Mar ~ 16 <sup>th</sup> Mar	14	Modeling	<ul style="list-style-type: none"> <li>● Analyze existing project</li> <li>● Modeling and design architecture for the new project</li> <li>● Learn various software</li> </ul>
3	17 <sup>th</sup> Mar ~ 11 <sup>th</sup> May	56	Implementation	<ul style="list-style-type: none"> <li>● Do coding, Testing and Debugging</li> </ul>
4	12 <sup>th</sup> May~ 25 <sup>th</sup> May	14	Documentation and final report creation	<ul style="list-style-type: none"> <li>● Complete all documents</li> <li>● Write up the final report</li> </ul>
5	26 <sup>th</sup> May ~ 1 <sup>st</sup> Jun	7	Preparation for final presentation	<ul style="list-style-type: none"> <li>● Edit slides for final presentation</li> </ul>

Table 3.2 Actual Timetable for “Web services architecture for a Fusion DataGrid” project

### 3.3 Gantt charts

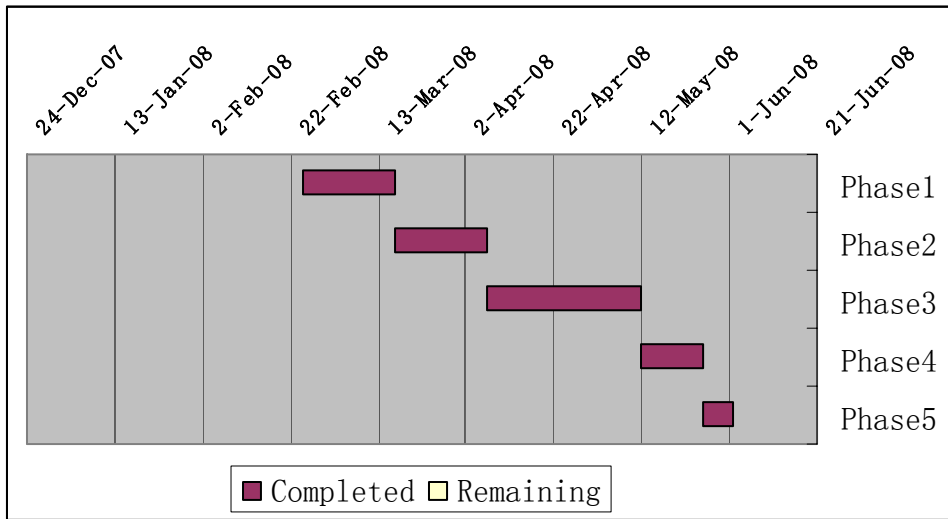


Figure 3.1 Planned Timetable Gantt Chart

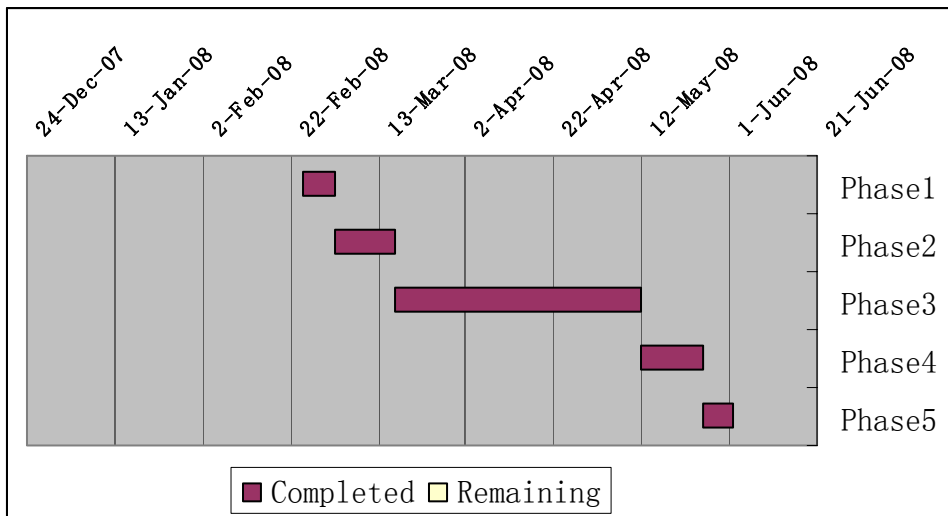


Figure 3.2 Actual Progress Gantt Chart

## 4. Modeling

### 4.1 Structure of WebScope4

The structure of WebScope4 is shown in Figure 1.4:

The structure of the WebScope4 system is largely the same as that of the WebScope3 system.

### 4.2 High Level Design

#### 4.2.1 System Context

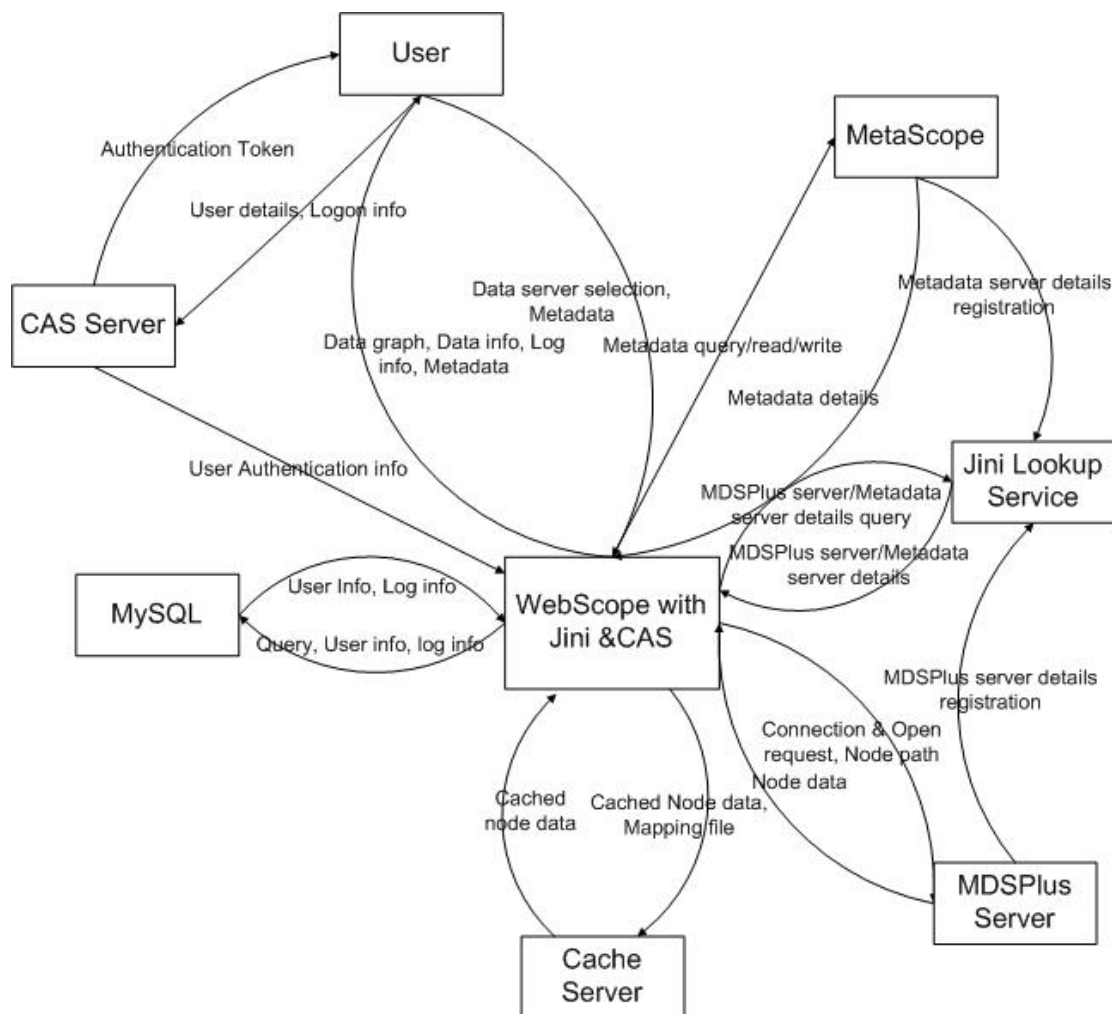


Figure 4.2 Context Diagram of “WebScope4”

There are six existing actors interacting with “WebScope4”.

- User:

1. A user inputs user login information (user email and password) at the login screen and this information is transferred to the CAS system. After authentication, the CAS system sends back a valid authentication token to the authenticated user.
2. A static metadata table will be shown to the user. The user specifies which data server to connect with server name, server port, experiment name and shot name. Then, the system sends data of the specified experiment, in the form of tree.
3. The user clicks to select a tree node that he/she wants to view. The system sends two type of expression of the tree node. One is the graph which is converted from binary code of the specified tree node. The other is data information such as numbers of point in the tree node and the text file which enables the user to download and analyze the data. The user can also contribute static/dynamic metadata to the database. In addition, the user can query metadata with different keywords.

- MDSPlus:

The WebScope4 system needs to connect to a MDSPlus server to work. A MDSPlus server with Jini automatically registers itself on the available or specified Jini Lookup Services as soon as it is loaded. WebScope4 clients can find the MDSPlus server via the Jini Lookup Services and then communicate with the MDSPlus Server directly. The MDSPlus server will send back the data required to the clients.

- Cache Server

1. The WebScope4 system sends data to the caching server in two possible formats: binary file and text file. The caching server stores the data.
2. When some data is required, the system sends a request to caching server to retrieve cached data. The caching server then returns the data to the system.

- MySQL

1. The WebScope4 system sends user information, metadata and log information to MySQL, where these kinds of information are saved.
2. When a certain kind of information is required, the WebScope4 system sends a request to the database to retrieve user information, metadata or log information.

The database then returns the required information to the system.

- CAS Server:

A CAS server is responsible for authenticating users in the WebScope4 system. It retrieves user details from user inputs and returns an authentication token if the user is proved to be valid. User authentication information is also passed to the WebScope4 system so that users can proceed with their operations.

- Jini Lookup Service:

Jini Lookup Service is responsible for MDSPlus server registering and responding to WebScope4 clients' queries.

- MetaScope:

MetaScope is a standalone web service which is responsible for metadata management. It registers itself on Jini Lookup Services. If found by a WebScope4 client, MetaScope then responds to the metadata read/write queries from the WebScope4 client.

#### 4.2.2 Domains

The WebScope4 system consists of seven main domains, which are described below:

Domain Name	Main Responsibility
WebScope4	Provides Ajax-based Web User Interface to users.
databaseAccessDomain	Provides an interface for the WebScope4 system to interact with the Hibernate mapping solution to read and write information from the MYSQL database.
dataServerDomain	Provides an interface for the WebScope4 system to retrieve datasets from MDSPlus server.
Graphics	Provides a series of classes to draw the data graph.
JiniServer	Provides an interface for the WebScope4 system to find Jini-based MDSPlus servers on Jini Lookup Services.
Web Server	Provides a container to run the WebScope4 system and the associated database.
Java	Provides the programming platform.
metaJini	Provides an interface for the WebScope4 system to find Jini-based MetaScope servers on Jini Lookup Services.

Table 4.2 Domain Description

The relationships between domains have been illustrated in the figure below (Figure 4.3):

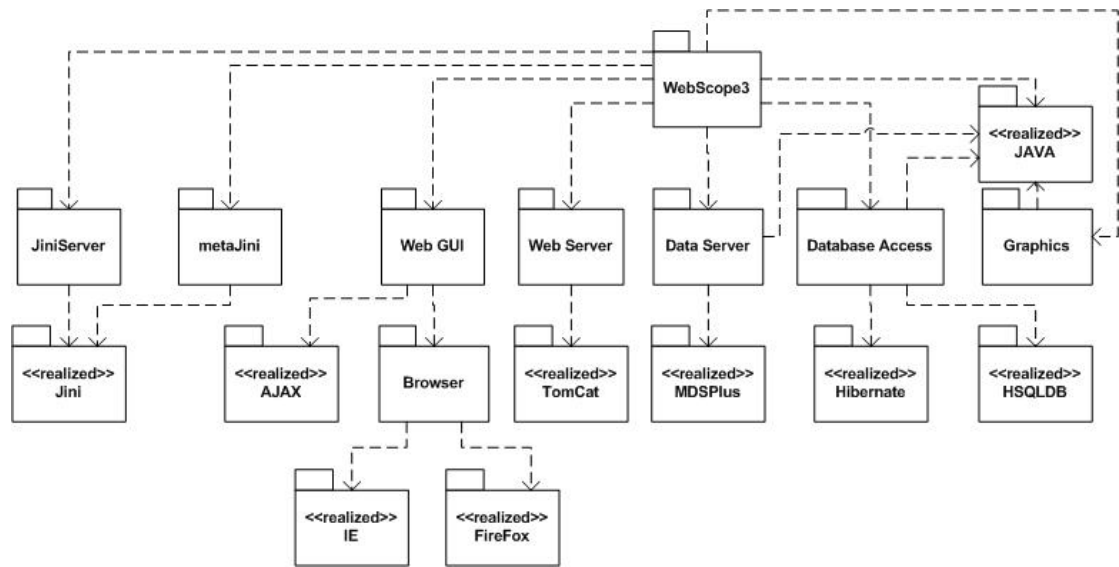


Figure 4.3 Domain Chart

## 4.3 Detailed Design

### 4.3.1 Communication between packages

The figure below (Figure 4.4) illustrates the communication between packages in the WebScope4 system:

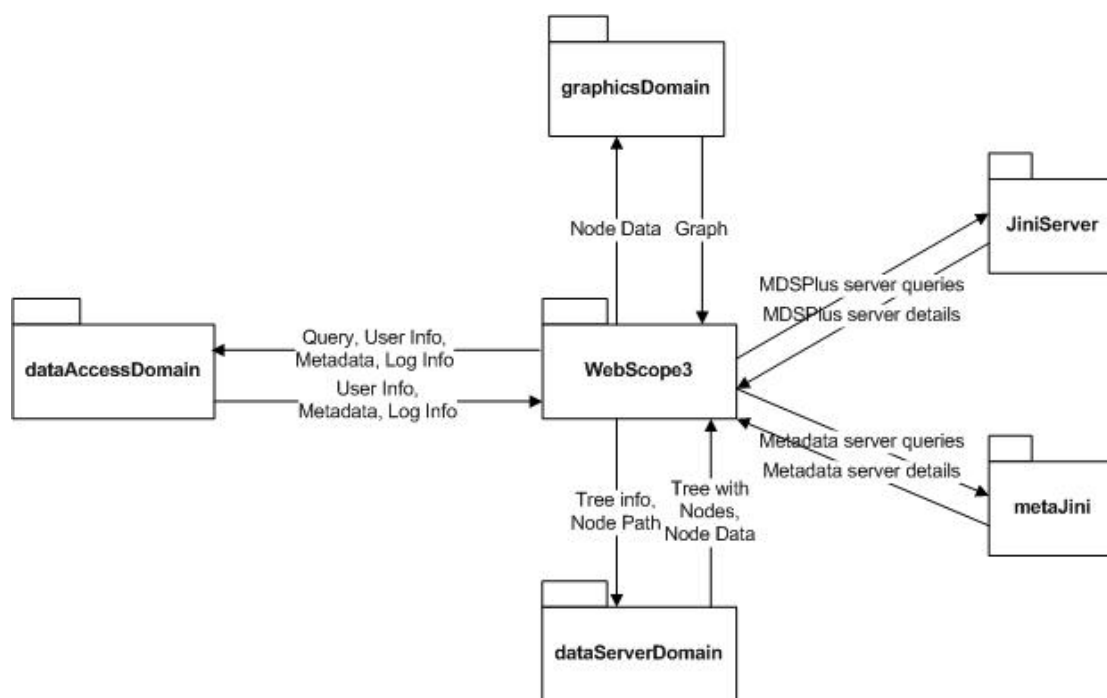


Figure 4.4 Communication between packages

### 4.3.2 Database Design

The database design for WebScope4 is based on the designs for the previous WebScope3 system. The detailed database design for WebScope4 is shown below:

Table Name	Explanation
USERS	Stores the personal details of users.
LOGINFO	Keeps a log of user activities.
METADATAINFO	Stores the static metadata information contributed by users. Metadata information is related to the experiment data to which the metadata information is added.
DYNAMICTABLEINFO	Keeps the dynamic metadata table

	information. Note that only the table names, column numbers and column names are stored. The column values are stored in the dynamic tables. This design is to facilitate the dynamic creation of java classes and Hibernate mapping files.
COLLABORATIONSPACEINFO	Stores the basic information of a Collaboration space, such as the space name and the name of the owner.
COLLABORATIONSPACECONTENT	Stores the information about the contents of Collaboration Spaces.
COLLABORATIONAUTHORIZATIONLIST	Stores the authorization information of Collaboration Spaces.

Table 4.1 Database Design

All interactions with the database are handled with the help of java object relational/mapping solution “Hibernate”. Java persistent classes were created to represent the database tables and the details of those classes were entered into the Hibernate mapping file (“Mapping.hbm.xml”). Hibernate dynamically creates the tables corresponding to the information provided in the mapping file, and performs various actions. The Appendix C shows the mapping file used by Hibernate package.

All the tables in the database use an Id field as the primary key and the values to these fields are dynamically generated by the Hibernate package.

The METADATAINFO table and LOGININFO table both have foreign keys referring to the USERS table.

### 4.3.3 WebScope4 Domain

The WebScope4 domain provides the Ajax-based Web User Interface to users. The structure of this domain is exactly the same as that of the WebScope2 domain in the WebScope2 system developed by Mr. Zhongshan Tan. However, the PlotDataServer class has been greatly changed to provide the newly required features.

The figure below (Figure 4.5) illustrates the classes in the WebScope4 domain and their relationships:

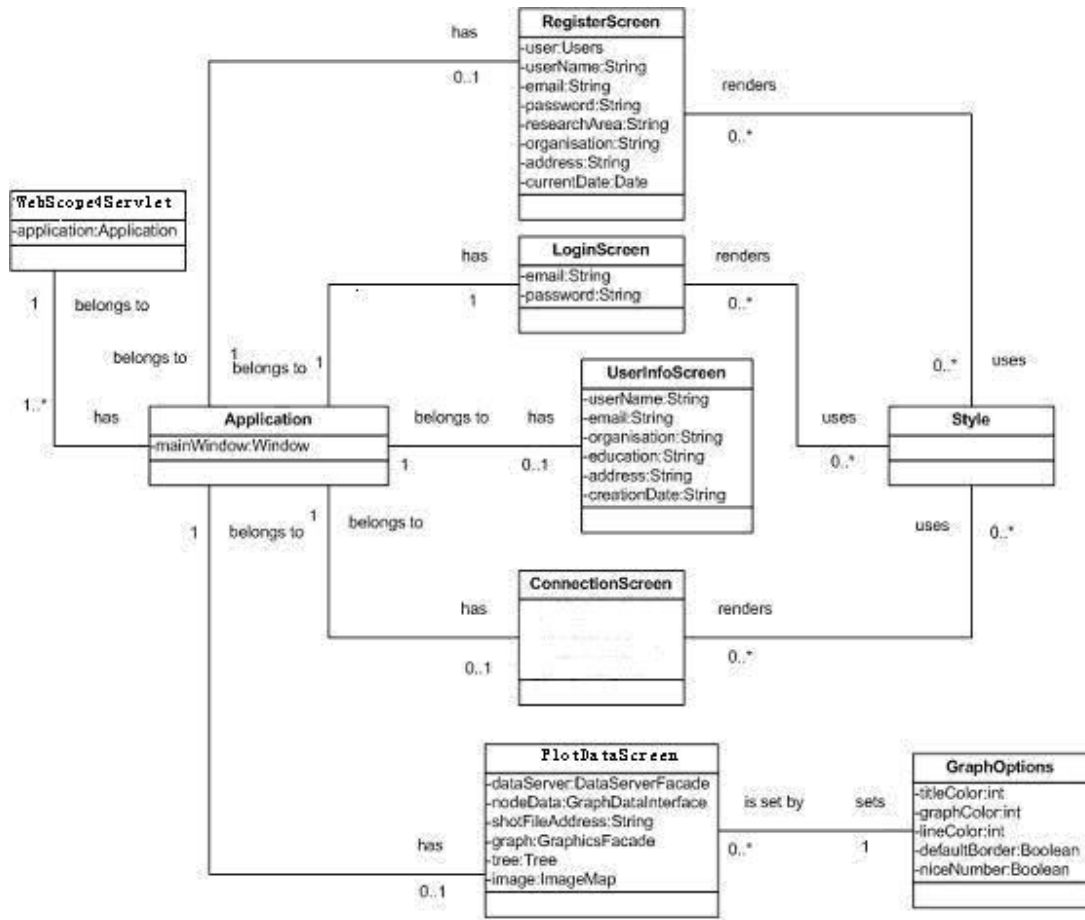


Figure 4.5 WebScope4 domain

The state machine of the WebScope4 domain is shown in the figure below (Figure 4.6):

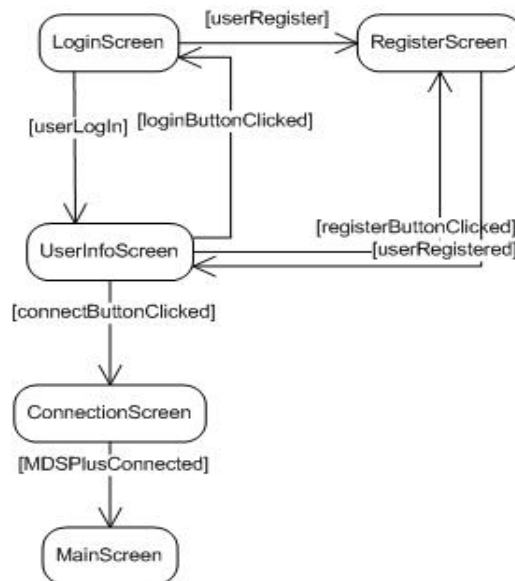


Figure 4.6 WebScope4 Domain State Chart

#### 4.3.4 dataAccessDomain Domain

In addition to the original User class in the dataAccessDomain domain of the WebScope2 system developed by Mr. Zhongshan Tan, 3 extra static classes, the DynamicTableInfo class, the LogInfo class, and the MetaDataInfo class, have been added.

The DynamicTableInfo class maps the DynamicTables table in the MySQL database via Hibernate. It stores the names of the dynamic tables, their number of columns and the names of the columns. However, the values of the columns are not stored here, but in the dynamically generated tables in the database. Therefore, the DynamicTableInfo class can be regarded as the manager or record of the dynamic tables.

The LogInfo class maps the LogInfo table in the MySQL database via Hibernate. It keeps a track of the access log of every user. Therefore, it has a Users foreign key.

The MetaDataInfo class maps the MetaDataInfo table in the MySQL database via Hibernate. It stores the static metadata information that users contribute to the database. It also has a Users foreign key together with a LogInfo foreign key.

The CollaborationSpaceInfo class maps the CollaborationSpaceInfo table in the MySQL database via Hibernate. It stores the basic information of Collaboration spaces such as the space name and owner. It has a Users foreign key.

The CollaborationSpaceContent class maps the CollaborationSpaceContent table in the MySQL database via Hibernate. It stores the information about the tables in Collaboration Spaces. It has a CollaborationSpaceInfo foreign key.

The CollaborationAuthorizationList class maps the CollaborationAuthorizationList table in the MySQL database via Hibernate. It stores the authorization information of Collaboration Spaces. It has a CollaborationSpaceInfo foreign key and a Users foreign key.

The figure below (Figure 4.7) illustrates the classes in the dataAccessDomain domain and their relationships:

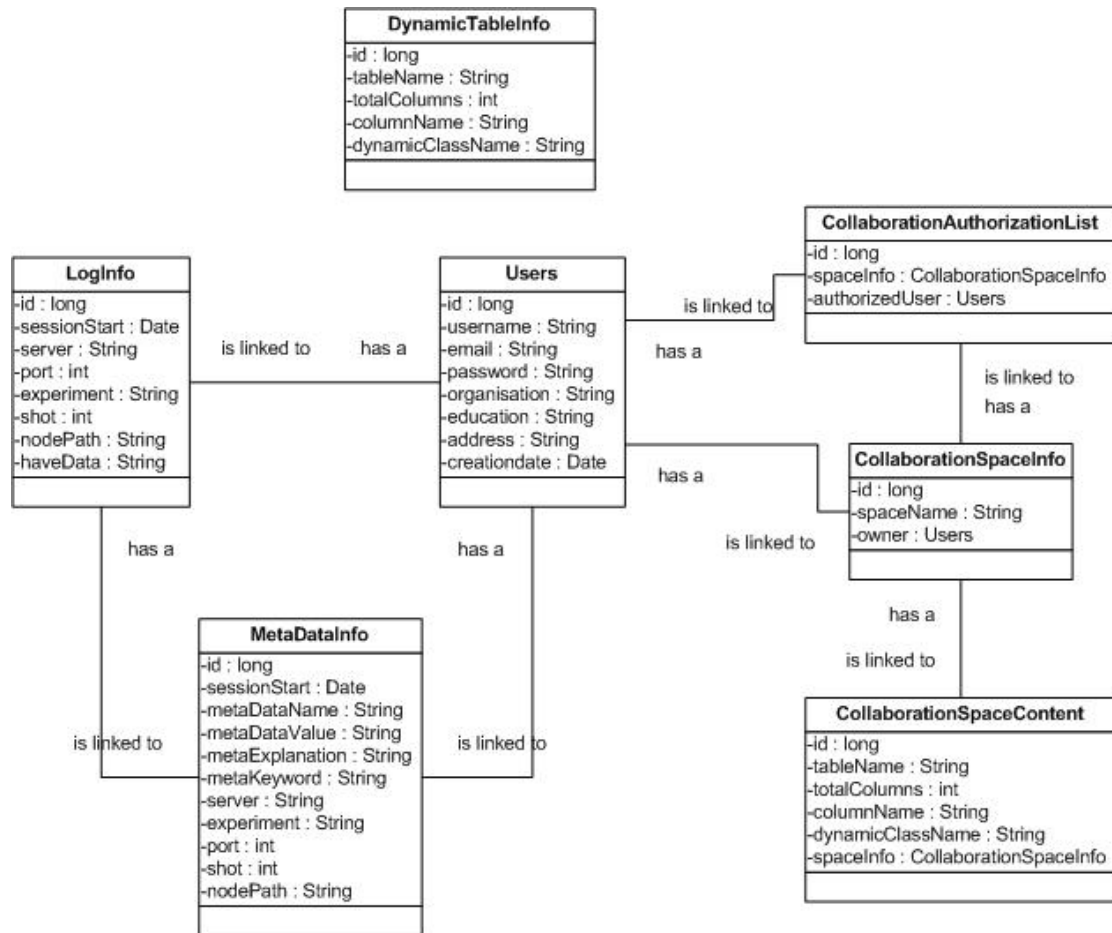


Figure 4.7 dataAccessDomain Domain

Moreover, classes are dynamically added to this domain at runtime if dynamic tables are contributed. The names of the classes and their attributes are consistent with the information stored in the DynamicTables table in MYSQL database.

### 4.3.5 dataServerDomain Domain

dataServerDomain domain here is inherited from the dataServerDomain domain of the WebScope4 system. It provides an interface for the WebScope4 system to retrieve datasets from the MDSPlus server.

Most classes in DataServer domain in WebScope2 system reuse dataServerDomain in EScope4 which is developed by Dr. Henry Gardner. Moreover, some additional classes are included in the domain. They are DataServerCache class, CacheThread class and TextFileMaker class.

- DataServerCache class is the class for caching data into local server rather other retrieving data from MDSPlus server every time.

- CacheThread class is the class in which a respective thread runs, which is on the purpose of enhance the performance and usability. Data can be downloaded while the system plot graph to user.
- TextFileMaker class converts binary file to text file, which enables users to download text file format of data to view and analysis off-line. Also, it has the caching function; any downloaded text file is cached for the later use.

The figure below (Figure 4.8) illustrates the classes in the dataServerDomain domain and their relationships:

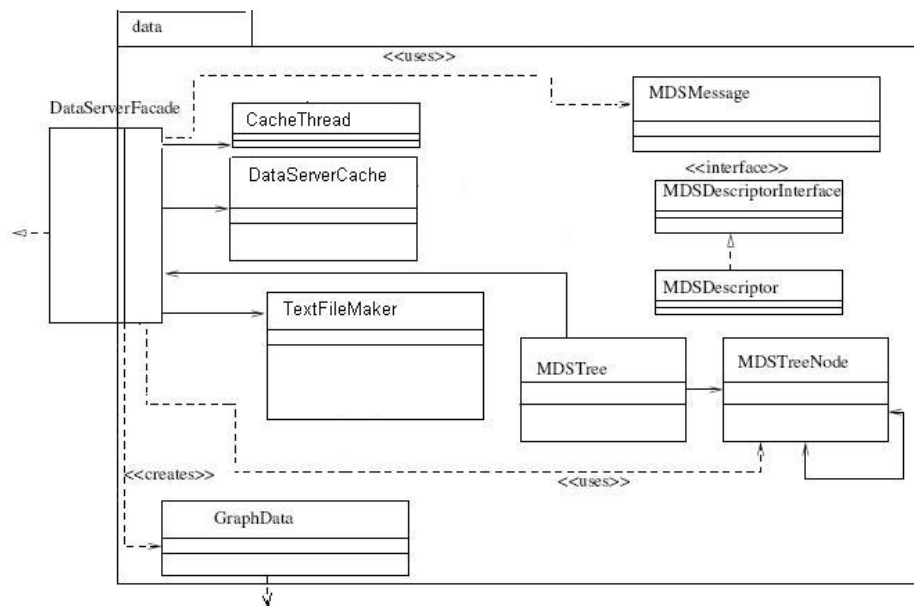


Figure 4.8 dataServerDomain Domain [1]

### 4.3.6 graphicsDomain Domain

The graphicsDomain Domain is currently inherited from the graphicsDomain domain of the WebScope4 system. However, this domain is being re-written. It provides a series of classes to draw the data graph.

The figure below (Figure 4.9) illustrates the classes in the current Graphics domain and their relationships:

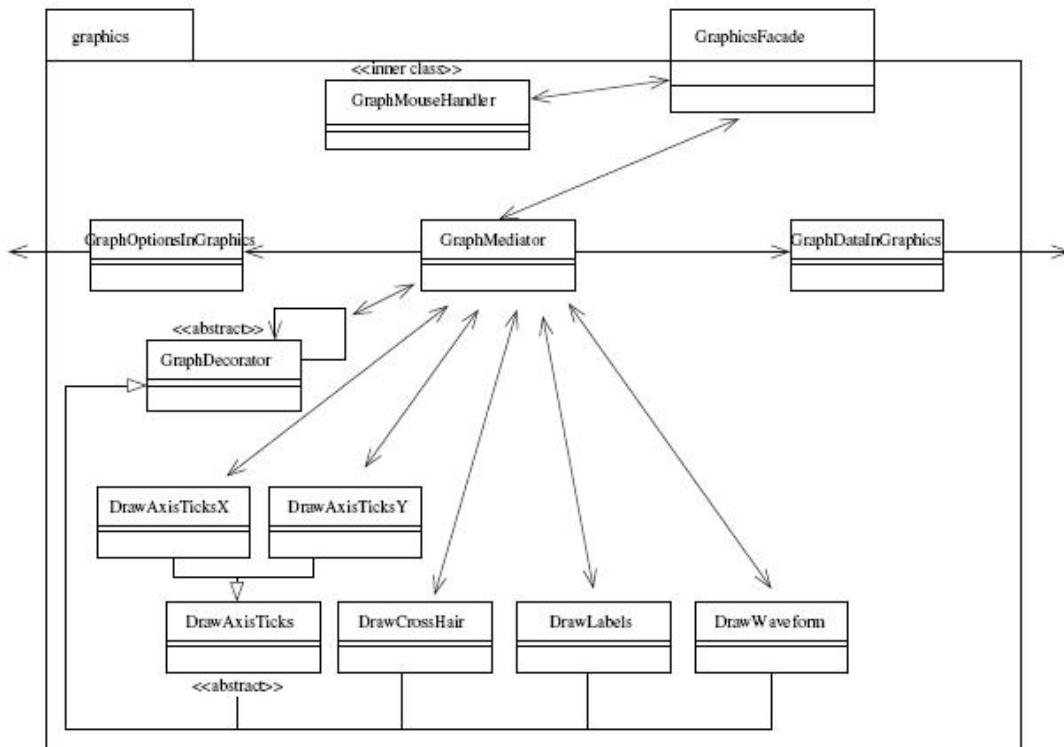


Figure 4.9 graphicsDomain Domain [1]

### 4.3.7 Mapping Domain

The Mapping domain is currently inherited from the Mapping domain of the WebScope4 system.

### 4.3.8 Applet Domain

The Applet domain is currently inherited from the Applet domain of the WebScope4 system. The figure below (Figure 4.10) illustrates the relationship between the 4 classes.

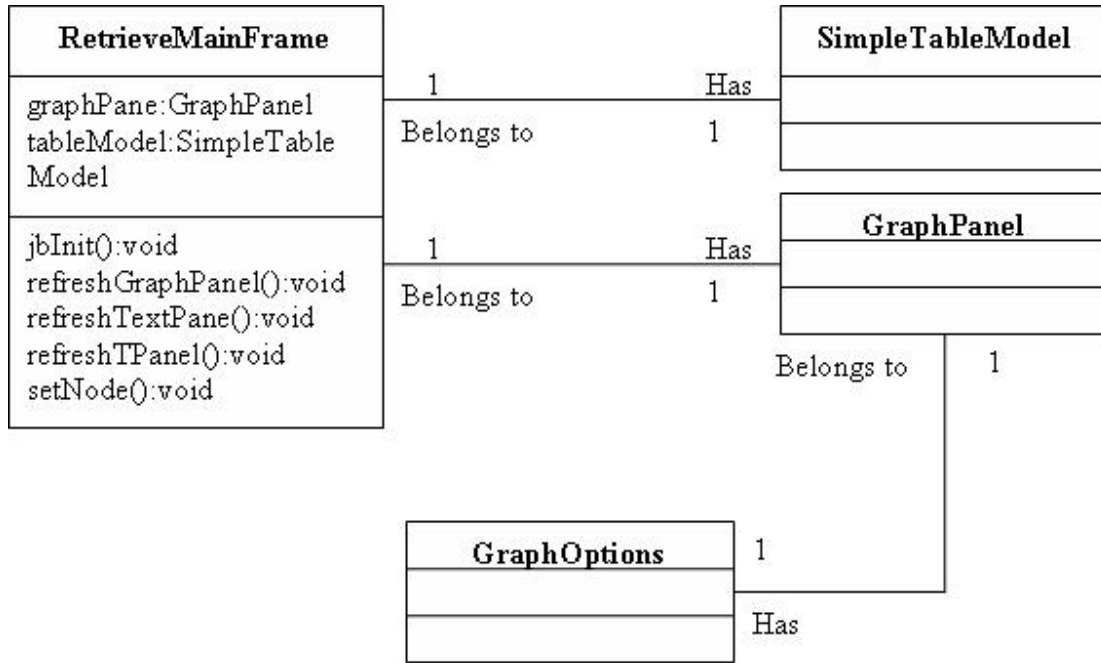


Figure 4.10 Applet Domain

### 4.3.9 JiniServer Domain

The JiniServer Domain is responsible for looking up and communicating with MDSPlus Jini servers. The figure below (Figure 4.11) illustrates the current structure of the JiniServer domain:

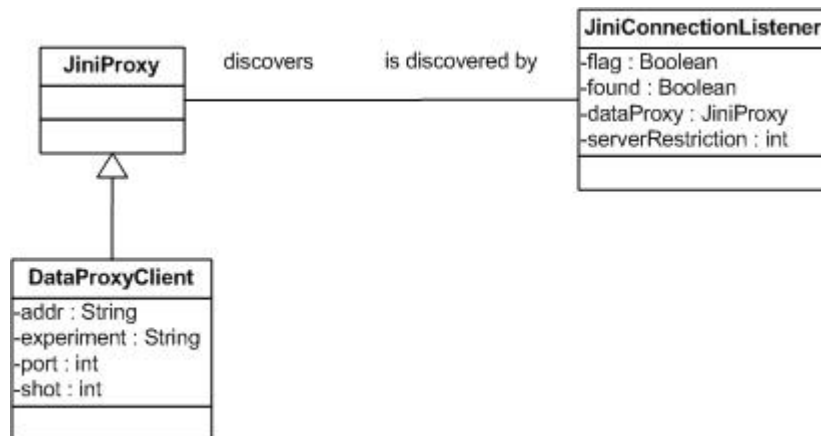


Figure 4.11 JiniServer Domain

### 4.3.10 metaJini Domain

The metaJini Domain is responsible for looking up and communicating with metadata Jini servers. The figure below (Figure 4.12) illustrates the current structure of the metaJini domain:

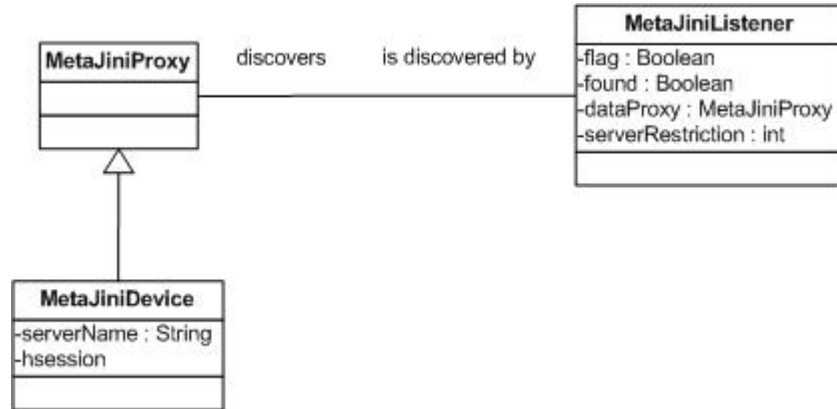


Figure 4.12 metaJini Domain

## 5. Implementation

### 5.1 Sub phases of the project

Since the number of technologies used for the development of “WebScope4” is high, the whole Implementation phase was divided into five sub phases. The details of these sub phases are given below:

Sub Phase Name	Activities Involved
1. Implementing Jini	Implementing the Jini technology in the existing WebScope4 system.
2. Implementing CAS	Implementing the CAS system in the existing WebScope4 system to authenticate users.
3. Migrating from HSQLDB to MySQL database	Replacing the current HSQLDB database with a MySQL database.
4. Creating a EScope12-based Jini client.	Making a EScope12 version of Jini client to improve the flexibility of the web portal.
5. Creating a standalone metadata server	Separating the metadata module from the original WebScope4 architecture and make it a standalone Jini service.
6. Creating a collaboration space for multiple users.	Letting users to share some metadata or information with other users so that they can work on the same data at the same time. This space is only accessible to users authorized by the space owner. The information included in spaces can be synchronized to the public metadata table.

Table 5.1 Sub-phases and activities involved

### 5.2 Lines of code

To measure the effort of the overall implementation phase of this project, the easiest way is to count the lines of code I have written for this project. However, this information doesn't reveal all the effort I have devoted to this project as the database configuration, CAS configuration, Jini configuration, and other debugging jobs are not calculated here.

Component	Lines of Code
WebScope4 Client	Original codes + 6500 lines
Jini-embedded MDSPlus Server	1000 lines
MetaScope	2600 lines
EScope12 Jini Client	Original codes + 500 lines

### 5.3 Implementation issues

I have met the following issues during the implementation phase of this project:

- **Issue:** MDSPlus Jini server could not register itself on the Jini Lookup Service. java.rmi.UnmarshalException: exception unmarshalling response is received.

**Resolution:** This issue is thrown because the Jini service could not find its code base, which should be the exported Jini Class. The code base should be obtained from an independent URL. Therefore, I created a file download page and put it on another Tomcat server. Afterwards, I added the following parameter when running the MDSPlus Jini server:

```
-Djava.rmi.server.codebase=http://192.168.102.135:8080/WebScope3/Download
```

The URL is the download link for the code base file, which has the file name JiniServer-dl.jar.

I had to add the parameter above on the Tomcat server which contained the WebScope4 client to make it find the MDSPlus Jini Servers through Jini Lookup Service.

- **Issue:** After HSQLDB was replaced with a MySQL database, Hibernate could not communicate with it. A “Cannot open connection” error message was received.

**Resolution:** This error message could appear for various reasons. I made the following changes to the Hibernate configuration file and mapping files to resolve this issue:

- Hibernate configuration file:

To make Hibernate communicate with a MySQL database, the following properties had to be configured in the Hibernate configuration file:

```
<property
name="connection.driver_class">com.mysql.jdbc.Driver</property>
<property
name="connection.url">jdbc:mysql://localhost:3306/webscope</property>
<property name="connection.username">root</property>
<property name="connection.password">1111</property>
<!--JDBC connection pool-->
<property name="connection.pool_size">1</property>
<!--SQL dialect-->
<property name="dialect">org.hibernate.dialect.MySQLDialect</property>
```

➤ Hibernate mapping files

In hibernate mapping files the type

- **Issue:** When trying to save an object to the CollaborationAuthorizationList table in the MySQL database via Hibernate, the error message below was received: org.hibernate.HibernateException: illegally attempted to associate a proxy with two open Sessions

**Resolution:** First of all, I tried using the same session whenever there was a need to access this table in the database. However, this did not work. After numerous attempts to resolve this issue, I finally got the resolution, which was to use the merge() method of the Session class instead of the save() method to save objects.

- **Issue:** By default, the CAS system does not read user name and password information from local database. It will authenticate users if we log on with same strings for user name and password. For example, we can log on with user name “tommy” and password “tommy”.

**Resolution:** The tutorial on the CAS home page is wrong and the official solution does not work. I had to find out my own resolution to this issue, which is as follows:

1. Open the file below with a text editor, such as UltraEdit32:  
%CATALINA\_HOME%/webapps/cas/WEB-INF/deployerConfigContext.xml
2. Find the line below in the file mentioned above and delete it:  
<bean  
class="org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler" />

3. Add the line below where the line above is found:

```
<bean
class="org.jasig.cas.adapters.jdbc.QueryDatabaseAuthenticationHandler">
<property name="sql" value="select password from app_user where
username=?" />
<property name="dataSource" ref="dataSource" />
</bean>
```

4. Add an independent bean as follows:

```
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource"
destroy-method="close">
<property
name="driverClassName"><value>com.mysql.jdbc.Driver</value>
</property>
<property
name="url"><value>jdbc:mysql://HOSTNAME:PORTNUMBER/DBNAME
</value></property>
<property name="username"><value>USERNAME</value></property>
<property name="password"><value>PASSWORD</value></property>
</bean>
```

NOTE: The HOSTNAME parameter should be changed to the host name of the MySQL database server. The PORTNUMBER parameter should be changed to the port number of the MySQL database server, which is usually 3306. The DBNAME parameter should be changed to the name of the database. The USERNAME parameter should be changed to the user name which has the read/write permission to the database and the PASSWORD parameter should be changed to the password of the corresponding user.

5. Copy the files below to the folder %CATALINA\_HOME%/webapps/cas/WEB-INF/lib:  
cas-server-jdbc-3.0.5-rc2.jar  
mysql-connector-java-3.1.12-bin.jar

6. Start Tomcat.

## 6. Extensibility Study

As described in the previous chapters, the WebScope4 system is currently getting dataset from MDSPlus servers. As the eScience community grows, the following problems have been brought to our attention:

- How easy is it to include another dataset in the grid?
- How easy is it to program a customized client for the grid?

This chapter will discuss the extensibility of the WebScope4 system and try to answer the questions above.

### 6.1 A description of the WebScope4 data grid

The WebScope4 data grid is the environment where all the web services and clients mentioned in this project work together. The grid includes the following components:

- MDSPlus servers with Jini embedded
- CAS servers
- Jini Lookup Services
- MetaScope
- Tomcat servers with WebScope4 clients deployed
- EScope 12 Jini client
- Users who access the grid via browsers

The figure below (Figure 6.1) illustrates an overview of the grid:

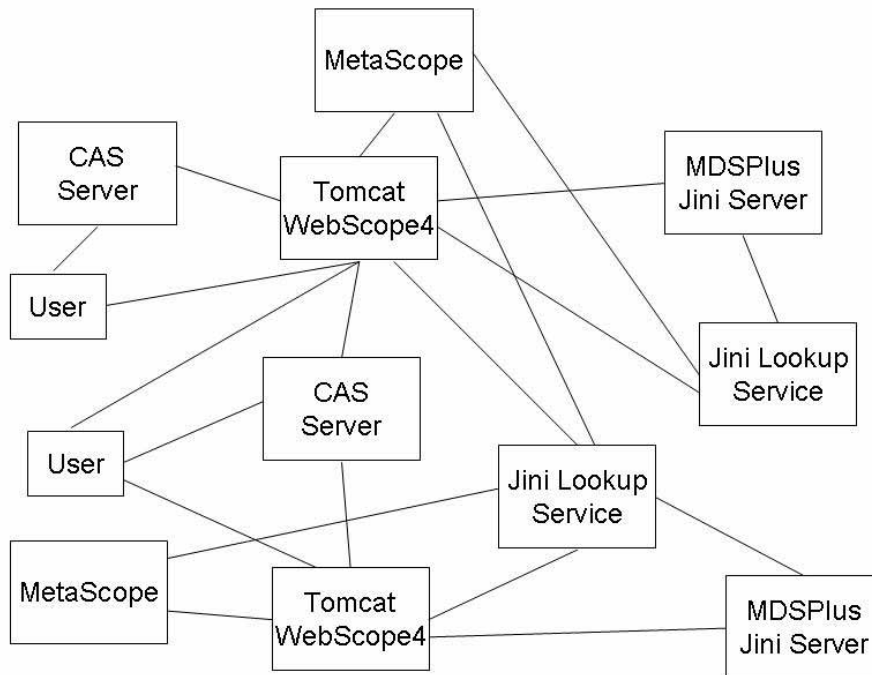


Figure 6.1 WebScope4 “Data Grid”

## 6.2 Including another dataset in the WebScope4 data grid

Although MDSPlus is the most widely used system for data management in the magnetic fusion energy program, it is possible that users may want to introduce another kind of data server into the current WebScope4 architecture.

According to the Modeling chapter above, the module of the WebScope4 system which is responsible for communicating with the MDSPlus server is the `dataServerDomain` domain. Therefore, to include another dataset in the grid, we need to swap the current `dataServerDomain` domain of the WebScope4 system with another one which can communicate with the new data server.

### 6.2.1 Interface definition

The new domain should provide the following interfaces so that it can communicate with the other existing part of the WebScope4 system:

- `public void connect(String serverAddrCPort ) throws IOException`

This function connects to the remote data server. The parameter should be the server address and port separated by a colon.

- `public void disconnect()` throws `IOException`

This function close the connection to the remote data server.

- `public void open(String experiment, int shot)` throws `IOException`

This function opens the experiment socket. The parameters are the name of the experiment and the shot number.

- `public void close()` throws `IOException`

This function closes the experiment socket.

- `public void constructTree(Tree tree)` throws `IOException`

This function constructs a tree of the experiment nodes. The tree will be

- `public void getDataDownload(ContentPane contentPane, GraphDataInterface graphData, String nodePath)`

This function prepares the data in plain text format so that they can be downloaded in a txt file. The `contentPane` parameter indicates the content pane which the download is invoked. The `graphData` parameter provides the function with the data to be packed. The `nodePath` parameter indicates the path to the node whose detailed data is going to be downloaded.

- `public String getSPES();`

This function returns server name, port number, experiment, and shot as a string of the form below:

Server name:port number||experiment name||shot number

- `public boolean isConnected()`

This function returns a Boolean variable indicating whether or not the data server is connected.

- `public boolean isOpen()`

This function returns a Boolean variable indicating whether or not the experiment socket is open.

- `public String getExperiment()`

This function returns the name of the open experiment as a string.

- `public int getShot()`

This function returns the shot number of the open experiment as an integer.

- `public GraphDataInterface getPlotData(String path)`

This function returns the data to be plotted. The parameter path indicates the path to the node to be plotted.

### **6.2.2 Design pattern requirement for the new domain**

As the Façade pattern has been currently implemented in the `dataServerDomain` domain, this design pattern is also required for the domain which is supposed to replace it.

The Façade class of the new domain should implement the `DataServerFacadeInterface` interface in the `sharedInterfaces` domain.

## **6.3 Programming a customized client for the grid**

Although we now have two types of clients for the grid, which are the `WebScope4` client and the `EScope12` client, it is possible that users may want to build their own customized clients for the grid so that they can add some specific functions or features to facilitate their study and research.

To achieve this, the new client will have to meet the following requirements:

- The client should be able to communicate with the `MDSPlus` server.
- The client should make itself a Jini client so that it can find Jini-based `MDSPlus` servers by querying Jini Lookup Service.
- If the client is supposed to be able to retrieve and contribute metadata from the Jini-based Metadata server, it should also be equipped another Jini client so that it can find the server automatically by querying Jini Lookup Service.

### **6.3.1 MDSPlus Jini client specification**

The object exported by the `MDSPlus` Jini server is as follows:

```

package JiniServer;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface JiniProxy extends Remote{

    public void setServer(String s) throws RemoteException;
    public String getServer() throws RemoteException;
    public void setPort(int p) throws RemoteException;
    public int getPort() throws RemoteException;
    public void setExperiment(String ex) throws RemoteException;
    public String getExperiment() throws RemoteException;
    public void setShot(int sh) throws RemoteException;
    public int getShot() throws RemoteException;
}

```

Therefore, a MDSPlus Jini client must be able to look up and discover the services which export the object listed above via Jini Lookup Services to find MDSPlus Jini servers.

### 6.3.2 Metadata Jini client specification

The object exported by the metadata Jini server is as follows:

```

package metaJini;

import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.List;

import databaseAccessDomain.*;

public interface MetaJiniProxy extends Remote{

    public List getAllMetadata() throws RemoteException;
    public void deleteRow(MetaDataInfo m) throws RemoteException;
    public List checkMetadataExistence(String s) throws RemoteException;
    public void saveMetaDataValues(String mName, String mValue, String mExplan,
String mKey, String mType, String mNodePath, String server, String experiment, int
shot, int port, Users user) throws RemoteException;
    public String getServerName() throws RemoteException;
    public void setServerName(String s) throws RemoteException;
}

```

Therefore, a metadata Jini client must be able to look up and discover the services which export the object listed above via Jini Lookup Services to find metadata Jini servers.

## **7. Testing**

Testing is always important as it can help to ensure the functionality and usability of a system. Three testing methods are carried out in this case, which are unit test, module test and acceptance test respectively.

### **7.1 Unit Test**

Unit test is a method used to check whether or not the individual units of a system are working properly. A unit is the smallest testable part of a system, which is a method in the WebScope4 system.

Unit tests are performed during the whole development process of this project. Whenever a new component or even several lines of code were added to the system, a unit test will be performed to ensure that the changes to the system will not cause problems.

### **7.2 Module Test**

Module test is a method used to check whether or not a collection of individual units of a system can work together properly. The collection of units can be a class or a package. Here, I describe a module test which was preferred on packages.

The WebScope4 system consists of 4 main components, which are as follows:

- WebScope4 client
- Jini-based MDSPlus Server
- MetaScope
- EScope12 client

Module tests will be performed on each of the four components above.

#### **7.2.1 WebScope4 client**

The WebScope4 client consists of packages, which are listed below:

- The databaseAccessDomain package
- The mapping package
- The dataServerDomain package
- The graphicsDomain package

- The sharedDataInterfaces package
- The sharedInterfaces package
- The webscope4 package
- The applet package
- The JiniServer package
- The metaJini package

Among the packages above, the following packages have not been changed:

- The mapping package
- The dataServerDomain package
- The graphicsDomain package
- The sharedDataInterfaces package
- The sharedInterfaces package
- The applet package

Therefore, the module tests for WebScope4 client focus on the packages below:

- The webscope4 package
- The JiniServer package
- The metaJini package
- The databaseAccessDomain package

The details of the module test have been included in the table below (Table 7.1~Table 7.5):

Test Plan	Test Result
<p><b>Plan 1: Logon on a User</b></p> <p>Step 1: Open a browser and open the link of the WebScope4 client.</p> <p>Step 2: Logon on a valid user at the CAS logon screen and check whether or not the CAS server can authenticate the user.</p> <p>Step 3: Logon on an invalid user at the CAS logon screen and check whether or not the CAS server can deny the access.</p> <p><b>Expected output:</b> The valid user should log on properly while the invalid user should not be logged on.</p>	<p>The valid user is logged on and the invalid user cannot be logged on</p> <p><b>Result: Test passed</b></p>

<p><b>Plan 2: Connect to a MetaScope server</b></p> <p>Step 1: Launch a WebScope4 client instance and logon a valid user. Step 2: Select a MetaScope server from the list of found MetaScope servers.</p> <p><b>Expected output:</b> The client should connect to the MetaScope server properly.</p>	<p>The MetaScope server is connected and a proper metadata table is listed on the user information screen.</p> <p><b>Result: Test passed</b></p>
<p><b>Plan 3: Connect to a Jini-based MDSPlus server</b></p> <p>Step 1: Launch a WebScope4 client instance and logon a valid user. Step 2: Select a Jini-based MDSPlus server from the list of found servers.</p> <p><b>Expected output:</b> The client should connect to the Jini-based MDSPlus server properly.</p>	<p>The Jini-based MDSPlus server is connected and data is retrieved from the MDSPlus server.</p> <p><b>Result: Test passed</b></p>
<p><b>Plan 4: Add a Collaboration Space</b></p> <p>Step 1: Launch a WebScope4 client instance and logon a valid user. Step 2: Connect to a MetaScope server, and then connect to a Jini-based MDSPlus server. Step 3: Create a Collaboration Space. Step 4: Authorize users to the new Collaboration Space. Step 5: Add tables to the new Collaboration Space.</p> <p><b>Expected output:</b> The new Collaboration Space should be created successfully. Authorization should be specified properly and new tables should be added to the space without problems.</p>	<p>The new Collaboration Space is created successfully. Authorization is specified properly and new tables can be added to the space.</p> <p><b>Result: Test passed</b></p>

Table 7.1 WebScope4 client Module Test

### 7.2.2 Jini-based MDSPlus Server

The main purpose of this component is to make the MDSPlus server a Jini service so that WebScope4 clients and EScope Jini clients can connect to it.

Test Plan	Test Result
<p><b>Plan: Start MDSPlus Server as a Jini service</b></p> <p>Step 1: Launch MDSPlus server.            Step 2: Start the Jini plug-in.            Step 3: Fill in the server details of the MDSPlus server and share it as a Jini service.</p> <p><b>Expected output:</b>            The service should be registered on Jini Lookup Service properly. WebScope4 clients and EScope Jini clients should be able to connect to it.</p>	<p>The server can be registered properly on Jini Lookup Service and WebScope4 clients and EScope Jini clients can find and connect to the MDSPlus Jini server properly.</p> <p><b>Result: Test passed</b></p>

Table 7.2 Jini-based MDSPlus Server Module Test

### 7.2.3 MetaScope

The main purpose of this component is to make metadata part of the WebScope4 system a standalone Jini service so that WebScope4 clients can connect to it.

Test Plan	Test Result
<p><b>Plan: Start MetaScope Server as a Jini service</b></p> <p>Step 1: Launch MetaScope.            Step 2: Click on the Start Jini Service button and fill in the server details of the MetaScope server to start it as a Jini service.</p> <p><b>Expected output:</b>            The service should be registered on Jini Lookup Service properly. WebScope4 clients should be able to connect to it.</p>	<p>The server can be registered properly on Jini Lookup Service and WebScope4 clients can find and connect to the MetaScope server properly.</p> <p><b>Result: Test passed</b></p>

Table 7.3 MetaScope Module Test

### 7.2.4 EScope Jini client

To make the WebScope4 system more flexible, I have converted a version EScope client, which is EScope12, to a Jini client so that it can be used in the WebScope4 system.

Test Plan	Test Result
<p><b>Plan: Connect to a Jini-based MDSPlus server</b></p> <p>Step 1: Start the EScope Jini Client.</p> <p>Step 2: Select a Jini-based MDSPlus server from the list of found servers.</p> <p><b>Expected output:</b> The list of available Jini-based MDSPlus servers should be retrieved and displayed. The client should be able to connect to the server and get experimental data.</p>	<p>The Jini-based MDSPlus server list can be retrieved from Jini Lookup Service properly. The server can be connected and experimental data can be got.</p> <p><b>Result: Test passed</b></p>

Table 7.4 applet Module Test

### 7.3 Acceptance Test

Acceptance test is a method used to check whether or not a system is acceptable to users and all the required features have been implemented.

No.	Req. Name	Priority	Implemented
R1	Implement Jini technology in the WebScope system	High	Yes
R2	Implement CAS in the WebScope system	High	Yes
R3	Migrate from HSQLDB to MySQL	High	Yes
R4	Creating a web service to centrally manage metadata	High	Yes
R5	Creating a Collaboration Space	High	Yes

Table 7.5 Acceptance Test

## 8. Summary of Contributions and Future Work

### 8.1 Summary of Contributions

The following contributions have been made to the WebScope4 system during this project:

- The concept of WebScope4 data grid has been built up. The grid has been designed to consist of the following components:
  - WebScope4 client
  - EScope12 client
  - Jini-based Metadata server
  - CAS server
  - Jini-based MDSPlus Server
  - Jini Lookup Service
- Jini technology has been integrated into the WebScope4 client, EScope12 client, Metadata server and MDSPlus server. This makes it much easier for servers and clients to find and serve each other.
- The CAS system has been involved to authenticate users in the WebScope4 system. This has largely improved the security of the data grid. It also makes the WebScope4 system more accessible by providing the Single-Sign-On feature.
- The original HSQLDB database has been replaced with a MySQL database, which is more stable, efficient and secure.
- The metadata database has been separated from the original WebScope3 database. A standalone central control system called “MetaScope” has been built up to maintain metadata and work as the metadata server. This gives us more options when designing the data grid and deploying the system.
- The “Collaboration Space” feature has been added to the WebScope4 client. This feature makes it possible for two or more users to share data and work on same data at the same time.

## 8.2 Future Work

The following topics should be considered as priorities for future development of the WebScope4 system:

- The WebScope4 client does not provide enough functions to users, such as multiple experiment plotting and video displaying. These functions might be added to the client in the future versions of WebScope4.
- Currently, users can access all the experiment data and metadata after logging on. Some secure authorization technology should be implemented in future WebScope systems to control the access to these data.
- The topology of CAS server federation should be well designed in the future so that it is more convenient and secure for users to be authenticated.
- The user interface of the components of the WebScope4 system may not be well designed and sometimes users may find some layout unclear or confusing. The interface should be improved in the future to improve the usability.
- The WebScope4 system is still running in an experimental environment and has not been deployed to real world nuclear labs. If possible, we shall deploy it and get feedback from the real users, which will help us to further improve the whole system.
- In order to have maximum impact, the WebScope4 system needs to be extensible by clients written in other popular languages and servers based on other database technology and protocols. The system design should be robust against these developments but this must be tested.

## 9. Reference

- [1] Henry J. Gardner, Raju Karia, Gabriele Manduchi, A Web-Based, Dynamic Metadata Interface to MDSplus, Fusion Engineering and Design, 83 (2008) 448-452.
- [2] Henry Gardner, Gabriele Manduchi, Design Patterns for e-Science, Springer Verlag, 2007, ISBN 978-3-540-68088-8.
- [3] Xiaobin Wang, Development of WebScope4, the final report for the COMP6703 eScience Project (Semester2, 2007) course,
- [4] MDSPLUS home page, <http://www.mdsplus.org/> . Last accessed 28 May 2008.
- [5] Clayton Lewis, John Rieman, Task-Centered User Interface Design, 1993.
- [6] Apache Tomcat Manual, <http://tomcat.apache.org/> . Last accessed 28 May 2008.
- [7] Jini home page, [http://www.jini.org/wiki/Main\\_Page](http://www.jini.org/wiki/Main_Page) . Last accessed 28 May 2008.
- [8] CAS home page, <http://www.ja-sig.org/products/cas/overview/background/index.html> . Last accessed 28 May 2008.
- [9] The HSQLDB official website, <http://hsqldb.org/> . Last accessed 28 May 2008.
- [10] Apache Ant 1.7.0 Manual, <http://ant.apache.org/manual/> . Last accessed 28 May 2008.
- [11] The java doc for the Karora Cooee project, <http://www.karora.org/projects/cooe/apidocs/> . Last accessed 28 May 2008.
- [12] The Hibernate official website, <http://www.hibernate.org/> . Last accessed 28 May 2008.
- [13] The MySQL official website, <http://dev.mysql.com/> . Last accessed 28 May 2008.
- [14] Java doc, <http://ephebe.anu.edu.au/doc/jdk1.5/api/index.html> . Last accessed 28 May 2008.

[15] The Wikipedia home page, <http://www.wikipedia.org/> . Last accessed 28 May 2008.

[16] Servlet Essentials, <http://www.novocode.com/doc/servlet-essentials/> . Last accessed 28 May 2008.

[17] Ajith Mannanakunnel Jose, Development of Web Scope, the final report for the COMP6703 eScience Project (Semester2, 2005) course, <http://escience/project/05S2/AjithJose/FinalReport.doc> . Last accessed 20 October 2007.

[18] Le Ma, Development of Web Scope, the final report for the COMP6703 eScience Project (Semester2, 2006) course, [http://escience/project/06S2/report/LeMa\\_report.pdf](http://escience/project/06S2/report/LeMa_report.pdf) . Last accessed 20 October 2007.

[19] Zhongshan Tan, Use of Echo2 in Web Scope, the final report for the COMP6703 eScience Project (Semester2, 2006) course, [http://escience/project/06S2/report/ZhongshanTan\\_report.pdf](http://escience/project/06S2/report/ZhongshanTan_report.pdf) . Last accessed 20 October 2007.

## Appendix A: Deployment Guide

As the WebScope4 system consists of the following components, this deployment guide will describe the detailed steps to install each component.

- MDSPlus server with Jini embedded
- CAS server
- Jini Lookup Service
- MetaScope
- WebScope4 client
- EScope12 client
- MySQL database

### A.1 Setup and configure software environment

The following table describes the dependency of the components mentioned above on the software environment:

Component	Dependency
MDSPlus server with Jini	Jini, JDK 1.6 or later, Apache Ant
CAS server	Apache Tomcat 5.5 or later, JDK 1.6 or later
Jini Lookup Service	Jini
MetaScope	Apache Tomcat 5.5 or later, JRE 6.0 or later, Jini
WebScope4 client	Apache Tomcat 5.5 or later, JRE 6.0 or later, Jini
EScope12 client	Jini, JDK 1.6 or later, Apache Ant

Table A.1 Components Dependency

#### A.1.1 Install JRE 6.0 or later versions/JDK 1.6 or later versions

1. Download Java 2 Standard Edition Runtime Environment (JRE), release version 6.0 or later/JDK 1.6 or later versions, from <http://java.sun.com/j2se> .
2. Install JRE/JDK according to the instructions shipped with the installation

package.

3. Set an environment variable named JAVA\_HOME to the path where JRE/JDK is installed. e.g. c:\Program Files\j2sdk6.0 (Windows) or /usr/local/java/j2sdk6.0 (Unix/Linux).

### **A.1.2 Install Apache Ant**

1. Download Apache Ant from the link below:

<http://ant.apache.org/bindownload.cgi>

2. Install Apache Ant according to the instructions shipped with the installation package.
3. Set an environment variable named ANT\_HOME to the path where Apache Ant is installed. Add the “bin” folder in the ANT\_HOME folder to the PATH system environment parameter.

### **A.1.3 Install Jini**

1. Download Jini Starter Kit from the link below:

[http://www.jini.org/wiki/Category:Jini\\_Starter\\_Kit](http://www.jini.org/wiki/Category:Jini_Starter_Kit)

2. Install Jini according to the instructions shipped with the installation package.
3. Set an environment variable named JINI\_HOME to the path where Jini is installed. Add the “jsk-lib.jar” and “jsk-platform.jar” files in the bin folder in the JINI\_HOME folder to the CLASSPATH system environment parameter.

### **A.1.4 Install Apache Tomcat**

1. Download Apache Tomcat 5.5 or later from the link below:

<http://tomcat.apache.org/>

2. Install Apache Tomcat 5.5 or later according to the instructions shipped with the installation package.
3. Set an environment variable named CATALINA\_HOME to the path where Apache Tomcat is installed.

## A.2 Install and configure the WebScope4 components

Before proceeding to the steps below, please check the dependency table A.1 and make sure that the dependency requirements have been met.

Please note that the following alias will be used in the rest part of this guide and in Appendix B:

Alias	Meaning
JAVA_HOME	The folder where JDK/JRE is installed
ANT_HOME	The folder where Apache Ant is installed
JINI_HOME	The folder where Jini Starter Kit is installed
CATALINA_HOME	The folder where Apache Tomcat is installed
WebScope4_HOME	The folder where the file WebScope4_all_in_one.rar is extracted to

Table A.2 Alias and the corresponding meaning

### A.2.1 Install MDSPlus server with Jini embedded

1. Download and install MDSPlus server from the link below:

<http://www.mdsplus.org/index.php?url=mdsplus/download.php&open=647955396387930143&page=Software%2FDownloads>

2. Copy the folder WebScope4\_HOME/WebScope4/JiniServer to local drive.
3. Refer to the user guide in Appendix B to run the MDSPlus Jini Server.

### A.2.2 Install CAS Server

1. Open a terminal and navigate to the bin folder under the JAVA\_HOME folder.
2. Type the commands below and press Enter after each command. Please note that each command consists of two lines:

```
keytool -genkey -alias tomcat-server -keyalg RSA -keypass changeit -storepass changeit -keystore server.keystore  
keytool -export -alias tomcat-server -storepass changeit -file server.cer -keystore
```

server.keystore

NOTE: after this command, you will be asked to input the password, which should be "changeit" (without the quotation marks).

```
keytool -genkey -alias tomcat-client -keyalg RSA -keypass changeit -storepass changeit -keystore client.keystore
```

```
keytool -export -alias tomcat-client -storepass changeit -file client.cer -keystore client.keystore
```

NOTE: after this command, you will be asked to input the password, which should be "changeit" (without the quotation marks).

```
keytool -import -trustcacerts -alias server -file server.cer -keystore cacerts -storepass changeit
```

```
keytool -import -trustcacerts -alias client -file client.cer -keystore cacerts -storepass changeit
```

3. Find the files cacerts which are generated by the commands above in the CATALINA\_HOME folder. Copy it to the folder JAVA\_HOME/jre/lib/security.
4. Locate the file server.xml in the folder CATALINA\_HOME/conf and open it with a text editor. Add the following paragraph in it:

```
<Connector port="8443" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS" keystoreFile="/server.keystore"
keystorePass="changeit"/>
```

5. Locate the file cas-server-3.2-release.zip in the folder WebScope4\_HOME/WebScope4/CAS. Extract it and find the file cas-server-3.2-release/cas-server-3.2/modules/cas-server-webapp-3.2.war in the extracted folder. Copy the file to CATALINA\_HOME/webapp
6. Run Apache Tomcat. Then stop it. This step will let Tomcat render the folder CATALINA\_HOME/webapp/cas-server-webapp-3.2 automatically.
7. Locate the file below:  
CATALINA\_HOME/webapps/cas-server-webapp-3.2/WEB-INF/deployerConfigContext.xml  
Open it with a text editor. Locate the line below in this file:  
<bean  
class="org.jasig.cas.authentication.handler.support.SimpleTestUsernamePasswordAuthenticationHandler" />

Delete or comment the bean above, and add the bean below to where the bean above is found:

```
<bean class="org.jasig.cas.adapters.jdbc.QueryDatabaseAuthenticationHandler">
<property name="sql" value="select password from app_user where username=?"
/>
<property name="dataSource" ref="dataSource" />
</bean>
```

Add the following independent bean afterwards:

```
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource"
destroy-method="close">
<property name="driverClassName">
<value>com.mysql.jdbc.Driver</value>
</property>
<property name="url">
<value>jdbc:mysql://localhost:3306/test</value>
</property>
<property name="username"><value>test</value></property>
<property name="password"><value>test</value></property>
</bean>
```

8. Copy the files `WebScope4_HOME/WebScope4/CAS/cas-server-jdbc-3.0.5.jar` and `WebScope4_HOME/WebScope4/MySQL/mysql-connector-java-3.1.14-bin.jar` to the folder `CATALINA_HOME/webapps/cas-server-webapp-3.2/WEB-INF/lib`.

### **A.2.3 Install MetaScope**

1. Locate the file `WebScope4_HOME/WebScope4/MetaScope/war/MetaScope4.war`
2. Copy it to the folder `CATALINA_HOME/webapp`

### **A.2.4 Install WebScope4 client**

1. Locate the file  
`WebScope4_HOME/WebScope4/WebScope4_src/war/WebScope4.war`
2. Copy it to the folder `CATALINA_HOME/webapp`

### **A.2.5 Install EScope12 client**

1. Open a terminal and navigate to the folder `WebScope4_HOME/WebScope4/escape12`.
2. Type the command “ant” (without quotation marks) and press Enter.

## Appendix B: User Manual

This section provides a detailed user manual for each component of the WebScope4 system.

### B.1 MDSPlus server with Jini

1. Install MDSPlus server with Jini as per the instructions in Appendix A. Run MDSPlus server
2. Open a terminal and navigate to the folder  
WebScope4\_HOME/WebScope4/JiniServer
3. Type the command “ant” (without quotation marks) and press Enter.
4. Click on the File menu. Click on Share... Then you will see a dialog box as shown in figure below:

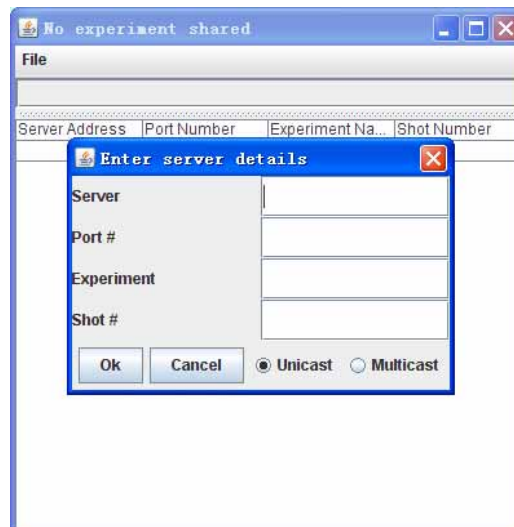


Figure B.1 Server Details Dialog Box

5. Fill in the server details, such as the server name, port number, experiment name and shot number. Choose the method to find the Jini Lookup Server (Unicast or Multicast). Click OK.
6. If the Unicast mode is selected. You will be required to input the address of the Jini Lookup Service. Please note that the address must be of the form `jini://hostname` or `jinni://hostname:port`. Click Ok.
7. The MDSPlus server with Jini will then be started.

## B.2 Jini Lookup Service

1. There should be a shortcut called Launch-All in the folder JINI\_HOME/installverify. Just run this shortcut and the Jini Lookup Service will be launched.

## B.3 MetaScope

1. Install MetaScope as per the instructions in Appendix A.
2. Start Apache Tomcat.
3. Launch a web browser and open the MetaScope main page with the link below:

<http://HOSTNAME:PORT/MetaScope/app>

Note: The HOSTNAME part should be changed to the host name or IP address of the server where MetaScope is installed on. The PORT part should be changed to the port number on which Tomcat is running, which is 8080 by default.

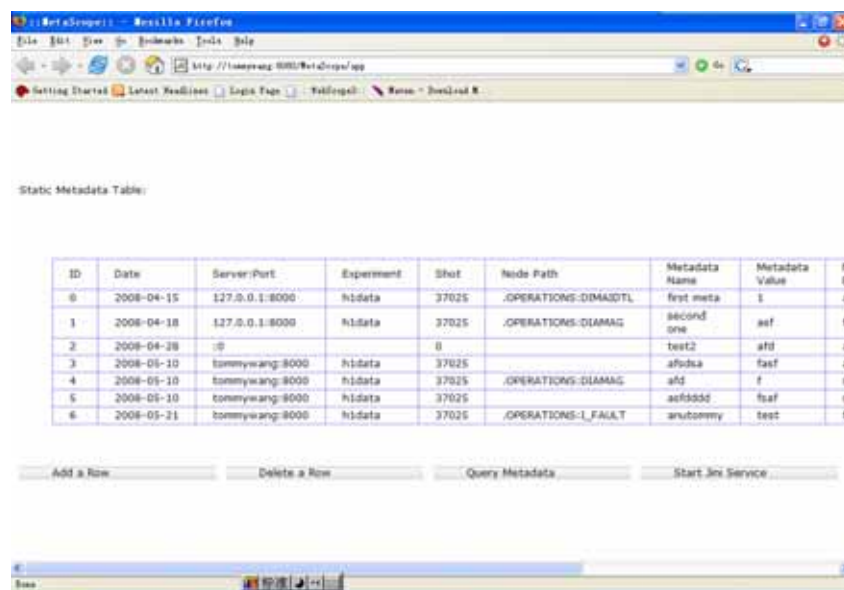


Figure B.2 Layout of MetaScope

4. The names of the four buttons on the main page of MetaScope describe exactly what they can do. Please make sure that the “Start Jini Service” button is clicked on as this will start a Jini service for the MetaScope instance.

## B.4 WebScope4 client

1. Install WebScope4 client as per the instructions in Appendix A.
2. Start Apache Tomcat.
3. Launch a web browser and open the WebScope4 client with the link below:

<http://HOSTNAME:PORT/WebScope4/app>

Note: The HOSTNAME part should be changed to the host name or IP address of the server where MetaScope is installed on. The PORT part should be changed to the port number on which Tomcat is running, which is 8080 by default.

4. Login with a user name and the corresponding password in the CAS page.
5. In the next page, click on the “Search for metadata servers” button. Then, select a Cast Mode to find the Jini Lookup Service. If Unicast is selected, you will be required to provide the address of the Jini Lookup Service of the form `jinni://hostname` or `jinni://hostname:port`
6. If metadata servers are available on the Jini Lookup Service, you will be provided with a list of the available servers. Select one of them and click on the Connect button to proceed.

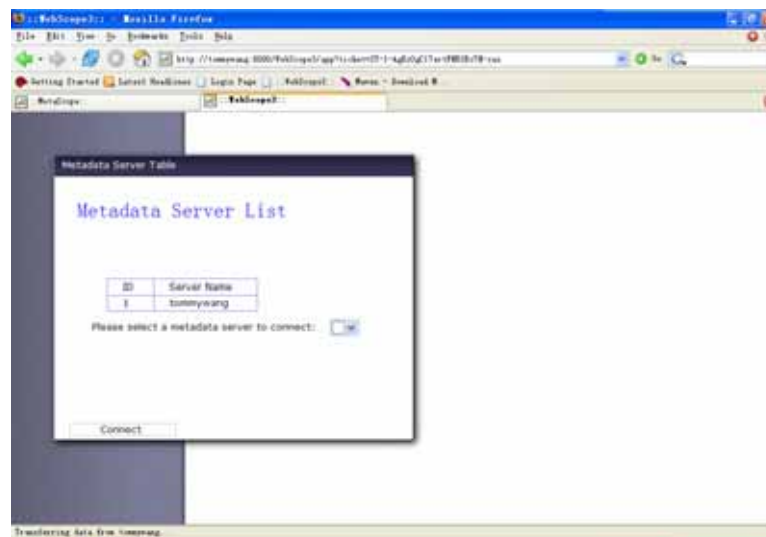


Figure B.3 Metadata Server List

7. You will see a screen with your user details listed. At this screen, you can register a new user or search for Jini-based MDSPlus servers.
8. If you click on the “Connect MDSPlus Servers” button, you will see a similar

dialog box as you get in Step 5. Do exactly what you did in Step 5.

9. You will be provided with a list of found MDSPlus servers. Select one of them and click on the Connect button.

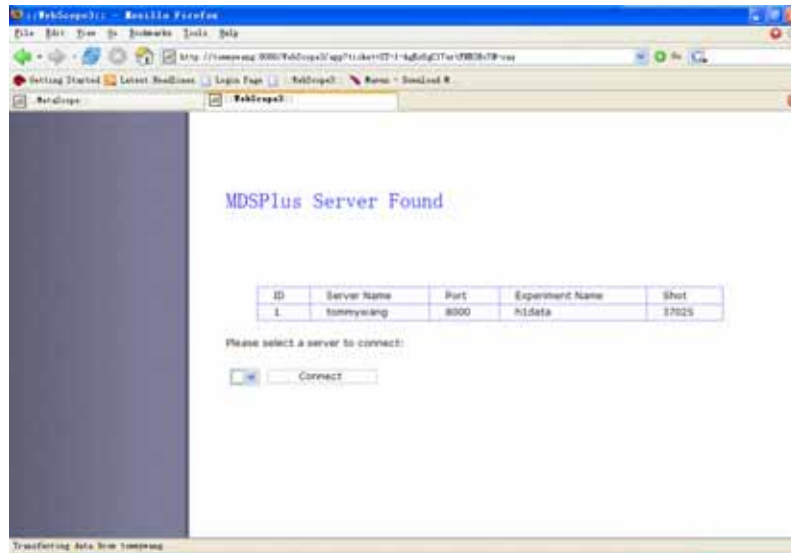


Figure B.3 MDSPlus Server List

10. The main screen of the WebScope4 client is quite similar to that of the WebScope3 client. However, the WebScope4 client main screen has two more buttons. One is the “Create Collaboration Space” button and the other is the “Manage Collaboration Spaces” button.
11. If you click on the “Create Collaboration Space” button, you will receive the dialog box as shown in the figure below (Figure B.4) which requires you to specify a name for the new collaboration space.

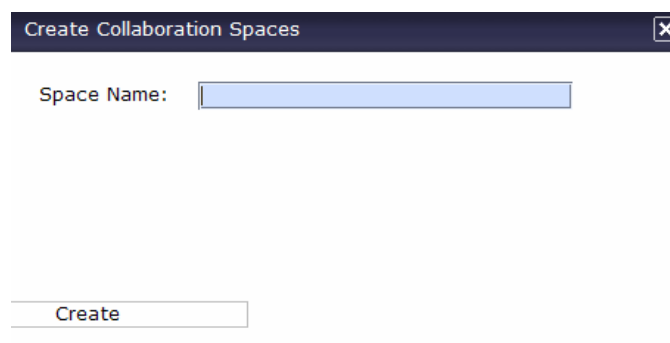


Figure B.4 Create Collaboration Spaces Dialog Box

12. If a valid space name is specified and the Create button is clicked on, you will receive the dialog box as shown in the figure below (Figure B.5) which requires you to authorize other users to visit the new space. After deciding which users you

would like to authorize, click on the Authorize button.

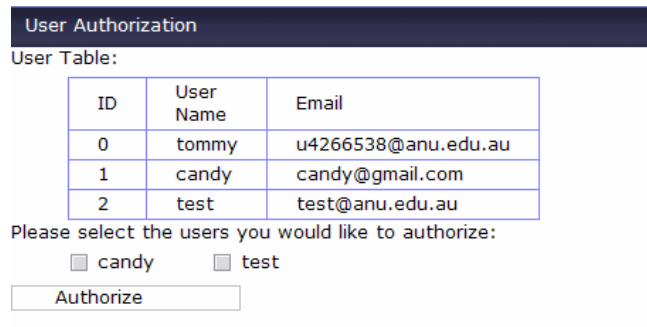


Figure B.5 User Authorization Dialog Box

13. On the next screen, you will have the option to add new tables to the space you have just created. Click on the "Add a new Table" button and follow the on screen instructions to add a table. After this process, the newly created table will be displayed in the Specify Space Content dialog box as shown in the figure below (Figure B.6).

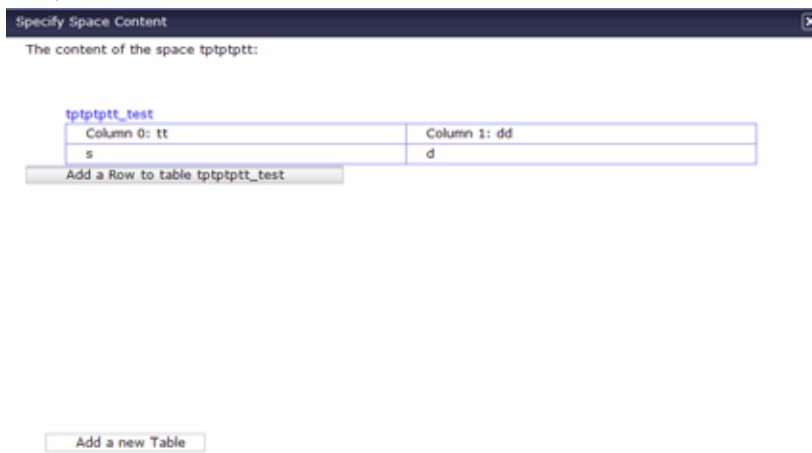


Figure B.6 Space Content Dialog Box

14. If the other new button on the WebScope4 main screen, which is the "Manage Collaboration Spaces" button, is clicked on, you will receive a list of all the available spaces, as shown in the Figure B.7. Select a space from the list and click on the Open button.

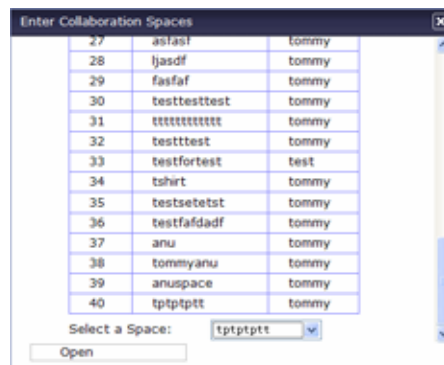


Figure B.7 Collaboration Space List

15. If you do not have the authority to access the selected space, you will receive the error message below:

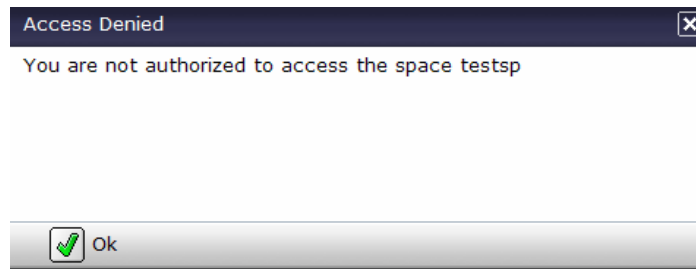


Figure B.8 Access Denied Message

Otherwise you will receive the dialog box below:

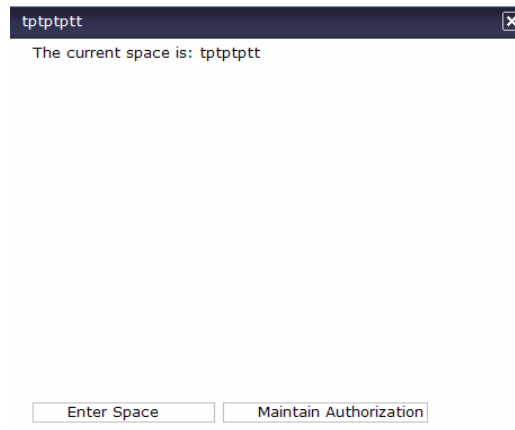


Figure B.9 Collaboration Space Dialog Box

16. If the Enter Space button is clicked on, you will receive the screen almost exactly the same as you did in Step 13. If the Maintain Authorization button is clicked on, you will get a list of authorized users as shown in the figure below:

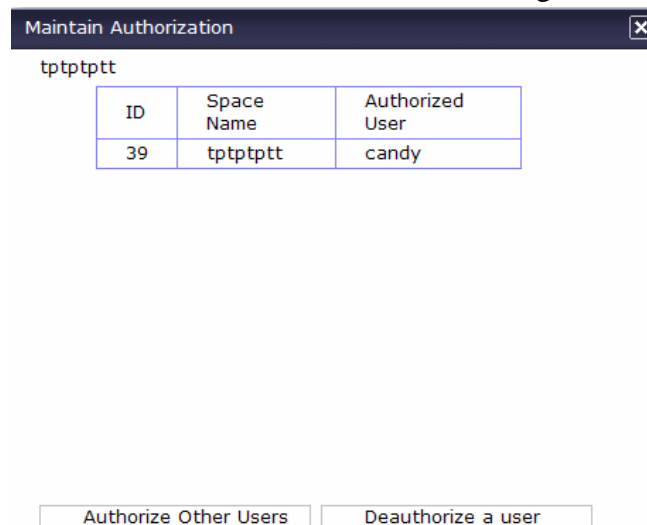


Figure B.10 Authorization List

17. Click on the “Authorize Other Users” button and you will get the dialog box similar to what you got in Step 12. Click on the “Deauthorize a user” button, and you will get the dialog box below:

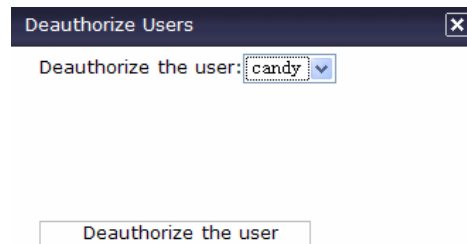


Figure B.11 Deauthorize Users Dialog Box

18. Select the user you would like to deauthorize, and click the “Deauthorize the user” button.