Deploying Applications to caGrid/TeraGrid

Developer's Cookbook

NATIONAL® CANCER INSTITUTE

Center for Bioinformatics

This is a U.S. Government work.

October 17, 2008

Model caBIG™ Open Source Software License

v.2

Release Date: January 7, 2008

Copyright Notice. Copyright 2008 The Ohio State University Research Foundation (OSURF), Argonne National Labs (ANL), SemanticBits LLC (SemanticBits), and Ekagra Software Technologies Ltd. (Ekagra) ("caBIG™ Participant"). The caGrid 1.2 software was created with NCI funding and is part of the caBIG™ initiative. The software subject to this notice and license includes both human readable source code form and machine readable, binary, object code form (the "caBIG™ Software").

This caBIG™ Software License (the "License") is between caBIG™ Participant and You. "You (or "Your") shall mean a person or an entity, and all other entities that control, are controlled by, or are under common control with the entity. "Control" for purposes of this definition means (i) the direct or indirect power to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

License. Provided that You agree to the conditions described below, caBIG™ Participant grants You a non-exclusive, worldwide, perpetual, fully-paid-up, no-charge, irrevocable, transferable and royalty-free right and license in its rights in the caBIG™ Software, including any copyright or patent rights therein, to (i) use, install, disclose, access, operate, execute, reproduce, copy, modify, translate, market, publicly display, publicly perform, and prepare derivative works of the caBIG™ Software in any manner and for any purpose, and to have or permit others to do so; (ii) make, have made, use, practice, sell, and offer for sale, import, and/or otherwise dispose of caBIG™ Software (or portions thereof); (iii) distribute and have distributed to and by third parties the caBIG™ Software and any modifications and derivative works thereof; and (iv) sublicense the foregoing rights set out in (i), (ii) and (iii) to third parties, including the right to license such rights to further third parties. For sake of clarity, and not by way of limitation, caBIG™ Participant shall have no right of accounting or right of payment from You or Your sublicensees for the rights granted under this License. This License is granted at no charge to You. Your downloading, copying, modifying, displaying, distributing or use of caBIG™ Software constitutes acceptance of all of the terms and conditions of this Agreement. If you do not agree to such terms and conditions, you have no right to download, copy, modify, display, distribute or use the caBIG™ Software.

1. Your redistributions of the source code for the caBIG™ Software must retain the above copyright notice, this list of conditions and the disclaimer and limitation of liability of Article 6 below. Your redistributions in object code form must reproduce the above copyright notice, this list of conditions and the disclaimer of Article 6 in the documentation and/or other materials provided with the distribution, if any.
2. Your end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Ohio State University Research Foundation (OSURF), Argonne National Labs (ANL), SemanticBits LLC (SemanticBits), and Ekagra Software Technologies Ltd. (Ekagra)." If You do not include such end-user

documentation, You shall include this acknowledgment in the caBIG™ Software itself, wherever such third-party acknowledgments normally appear.

3. You may not use the names “The Ohio State University Research Foundation”, "OSURF", “Argonne National Labs”, “ANL”, “SemanticBits LLC”, "SemanticBits", “Ekagra Software Technologies Ltd.”, “Ekagra”, “The National Cancer Institute”, “NCI”, “Cancer Bioinformatics Grid” or “caBIG™” to endorse or promote products derived from this caBIG™ Software. This License does not authorize You to use any trademarks, service marks, trade names, logos or product names of either caBIG™ Participant, NCI or caBIG™, except as required to comply with the terms of this License.

4. For sake of clarity, and not by way of limitation, You may incorporate this caBIG™ Software into Your proprietary programs and into any third party proprietary programs. However, if You incorporate the caBIG™ Software into third party proprietary programs, You agree that You are solely responsible for obtaining any permission from such third parties required to incorporate the caBIG™ Software into such third party proprietary programs and for informing Your sublicensees, including without limitation Your end-users, of their obligation to secure any required permissions from such third parties before incorporating the caBIG™ Software into such third party proprietary software programs. In the event that You fail to obtain such permissions, You agree to indemnify caBIG™ Participant for any claims against caBIG™ Participant by such third parties, except to the extent prohibited by law, resulting from Your failure to obtain such permissions.

5. For sake of clarity, and not by way of limitation, You may add Your own copyright statement to Your modifications and to the derivative works, and You may provide additional or different license terms and conditions in Your sublicenses of modifications of the caBIG™ Software, or any derivative works of the caBIG™ Software as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

6. THIS caBIG™ SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES (INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE) ARE DISCLAIMED. IN NO EVENT SHALL THE OHIO STATE UNIVERSITY RESEARCH FOUNDATION ("OSURF"), ARGONNE NATIONAL LABS (“ANL”), SEMANTICBITS LLC ("SEMANTICBITS"), AND EKAGRA SOFTWARE TECHNOLOGIES LTD. (EKAGRA) OR ITS AFFILIATES BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS caBIG™ SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Revision History

The following is the revision history for this document.

Date	*Version*	*Description*	*Revised By*
10/17/08	1.0	Initial Release of CookBook	Bronwyn Gagne

Contents

About This Guide

This section introduces you to the *Deploying Applications to caGrid/TeraGrid - Developer's Cookbook*. Topics in this section include

Purpose

The cancer Biomedical Informatics Grid, or caBIG™, is a voluntary virtual informatics infrastructure that connects data, research tools, scientists, and organizations to leverage their combined strengths and expertise in an open environment with common standards and shared tools. The current test bed architecture of caBIG™, is dubbed caGrid. The software embodiment and corresponding documentation of this architecture constitute the caGrid 1.2 release.

The primary goal of the *Deploying Applications to caGrid/TeraGrid - Developer's Cookbook* is to provide users with instructions for deploying their caGrid service and for creating a caGrid service that is TeraGrid-aware. Figure 1-1 below provides a diagram of a deployed application.

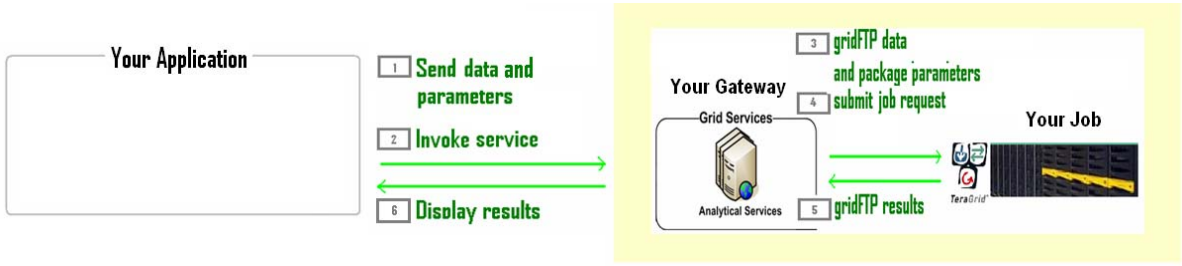

Figure 1-1: Diagram showing caGrid Deployment

The secondary goal of this guide is to provide a portable version of information that already resides on the Internet. The information used for this guide appears on two wiki pages created and hosted by the Informatics team at Columbia University: http://wiki.c2b2.columbia.edu/informatics/index.php/Cook_Book and http://wiki.c2b2.columbia.edu/informatics/index.php/GeWorkbench_Example.

The caGrid team determined that this information would also be well served in a printable, portable version made available for users.

Audience

This guide is designed for developers who want to create and deploy a TeraGrid-aware service either on caGrid or TeraGrid. This guide provides both generic instructions for performing this deployment, and a specific example of an application deployment on TeraGrid.

This guide assumes that you are familiar with caGrid, the Java programming language and/or other programming languages, database concepts, and the Internet. If you intend to use caGrid resources in software applications, it assumes that you have experience with building and using complex data systems.

To get the most out of this guide, be sure to review the next two sections of this chapter which list the tools used for the project and the security services used to set up the project prior to deployment. Each of these sections provide links to websites that contain more detailed information regarding each of the tools or applications. Click these links to review this information as necessary.

Topics Covered

This section provides a brief overview of what you will find in each chapter of this guide.

- *About This Guide*, this chapter, provides information regarding the guide as well as information and links for the tools used to develop the procedures provided in the guide.
- *Chapter 1, Configuring and Deploying a caGrid Service* beginning on page 9, provides generic instructions for configuring and deploying a service on the grid.
- *Chapter 2, Deploying the geWorkbench Application* beginning on page 29, provides the specific procedures used to deploy the geWorkbench application to the grid.
- *Appendix A References* on page 74 provides references relevant to caGrid.
- *Appendix B Glossary* on page 79, defines acronyms, objects, tools and other terms related to caGrid.

Tools Used for the Project

Below is a list of links to all the tools used in this project. If you are not familiar with these tools we strongly recommend reading at least the entire first chapter to get an idea of how these tools were used before coming back for more information on the tools themselves.

caGrid Tool Links

General

- Overview — This link directs you to the main caGrid wiki page, which provides an overview of the caGrid project and links to further information regarding caGrid.

- How-To — This link directs you to the "How To" caGrid wiki page, which provides links to instructions for installing caGrid, development of caGrid software, and links to both basic and advanced service development information.

Security

- Grid Authentication and Authorization with Reliably Distributed Services (GAARDS) — This link directs you to the main caGrid GAARDS wiki page, which provides an overview of how GAARDS works and links to related informational sites.
- Dorian — This link directs you to the main caGrid Dorian wiki page, which provides an overview of how Dorian works.
- Grid Trust Service (GTS) — This link directs you to the main caGrid GTS wiki page, which provides an overview of how GTS works.
- Credential Delegation Service (CDS) — This link directs you to the main caGrid CDE wiki page, which provides an overview of how CDS works.
- GridGrouper — This link directs you to the main caGrid GridGrouper wiki page, which provides an overview of how GridGrouper works.

Deploying a Secured Container

- Configure Tomcat to be a Secured Container — This link directs you to the caGrid Configure Tomcat wiki page, which provides instructions for manually configuring tomcat to host grid services.

or

- Install Globus as the Secured Container — This link directs you to the specific section of the caGrid Installer wiki page that contains instructions for creating a secure container using the caGrid Installer.

Communicating with TeraGrid

- Introduce with gRAVI Plug-In — This link directs you to the Globus gRAVI Users Guide wiki page.

TeraGrid Tool Links

General

- TeraGrid Overview — This link directs you to the TeraGrid website, which provides an overview of the TeraGrid project and links to further information regarding TeraGrid.
- TeraGrid User Support — This link directs you to the User Support page of the TeraGrid website, which provides a wide range of data, including first-time user information, documentation and knowledge base links, latest user news, and Help Desk contact information.

Logging On

- Logging in/Getting Started — This link directs you to the Start Using My Account page of the TeraGrid Getting Started Guide on the TeraGrid website. This page provides information for getting started with TeraGrid and for logging onto a TeraGrid resource for the first time.
- Logging in (Windows) — This link directs you to the SSO From Your Desktop (Windows) page of the Access section of the TeraGrid website. This page provides information for how to configure GSI-SSHTerm to connect to multiple resources on the TeraGrid after logging in once.
- Single Sign On Proxy — This link directs you to the Proxy Certificates page of the Access section of the TeraGrid website. This page provides instructions for creating a temporary credential called a certificate proxy. The proxy confirms that you are authorized by a trusted authority to access grid resources.
- Single Sign On Quick Start — This link directs you to the SSO Quick start page of the Access section of the TeraGrid website. This page provides basic quick reference information regarding the different methods for setting up SSO access to TeraGrid.

Submitting Jobs

- Manually Submitting Jobs — This link directs you to the main page of the Globus Toolkit 4.0 WS Gram User's Guide website.
- Submitting Jobs Using Java — This link directs you to the Submitting a job in Java using WS GRAM page Globus Toolkit website.
- Running Jobs — This link directs you to the Computing & Running Jobs Overview page of the TeraGrid website. This page provides basic information about job submission on TeraGrid resources and links to other related TeraGrid pages.
- TeraGrid GRAM Resources — This link directs you to the Jobs:Gram page of the TeraGrid website. This page provides information regarding using GRAM to remotely run jobs on TeraGrid systems.

Moving Files

- gridFTP — This link directs you to the GridFTP Clients page of the TeraGrid website. The page provides information on using the GridFTP protocol to transfer files.
- Data Staging — This link directs you to the Start Using My Account page of the TeraGrid Getting Started Guide on the TeraGrid website. This page contains information and guidelines for transferring data and files between sites.
- Plain Old Moving Files — This link directs you to the page of the TeraGrid website that provides information and instructions for moving local files to the TeraGrid.

geWorkbench Links

- GUI — This link directs you to the home page for the geWorkbench project. This page provides basic overview information for geWorkbench and links to further information.
- Dispatcher — This link directs you to the main Dispatcher web page, which provides links to further information about Dispatcher.

Additional Information

- WS-GRAM — Grid Resource Allocation and Management. This is a **Globus protocol** used for remote job submission and control. This link directs you to the main documentation site for WS GRAM.
- TeraGrid GRAM Resources — This link directs you to the main GRAM page of the TeraGrid website. GRAM is a component of the Globus Toolkit 4.0.
- Condor -- There is talk of using Condor to submit jobs to TeraGrid. Condor will use the GRAM protocol for this job submission. Condor has support for java jobs. This link directs you to a webpage that describes Condor and provides links to further information about the Condor project.

caGrid Services Used For Setup

Prior to the service deployment steps, each chapter contains setup instructions and procedures that use the caGrid services listed below. Use the website links below if you need further information regarding how to use any of these services.

You can set up your own Dorian, GTS, CDS, and Grid Grouper services using the instructions contained in the below links.

- Grid Authentication and Authorization with Reliably Distributed Services (GAARDS)
- Dorian
- Grid Trust Service (GTS)
- Credential Delegation Service (CDS)
- Grid Grouper

The procedures contained in this guide use the services available on the caGrid training grid.

Document Text Conventions

The following table shows how text conventions are represented in this guide. The various typefaces differentiate between regular text and menu commands, keyboard keys, and text that you type.

Convention	***Description***	***Example***
Bold	Highlights names of option buttons, check boxes, drop-down menus, menu commands, command buttons, or icons.	Click **Search**.
URL	Indicates a Web address.	http://domain.com
text in Small Caps	Indicates a keyboard shortcut.	Press Enter.
text in Small Caps + text in Small Caps	Indicates keys that are pressed simultaneously.	Press Shift + Ctrl.
Italics	Highlights references to other documents, sections, figures, and tables.	See *Figure 4.5*.
Italic boldface monospace type	Represents text that you type.	In the **New Subset** text box, enter ***Proprietary Proteins***.
Note:	Highlights information of particular importance.	**Note:** This concept is used throughout this document.
{ }	Surrounds replaceable items.	Replace {last name, first name} with the Principal Investigator's name.

Table 1-1 Document Conventions

Credits and Resources

The caGrid team would like to thank the author(s) and editor(s) of the wiki pages that contain the original procedures copied to and provided in this guide. The caGrid team would also like to thank the Columbia University Informatics team for allowing the transcription of their published information into the caBIG format for distribution to users.

For reference, the wiki pages from which the information in this guide was taken are located at the following URLs:

- http://wiki.c2b2.columbia.edu/informatics/index.php/Cook_Book
- http://wiki.c2b2.columbia.edu/informatics/index.php/GeWorkbench_Example.

The following people contributed to the development of this document:

Deploying Applications to caGrid/TeraGrid - Developer's Cookbook Development and Management Teams		
Development	***Documentation Support***	***Management***
Ravi Madduri[2]	Bronwyn Gagne[4]	Avinash Shanbhag (Product Manager)[1]
	Jill Hadfield[1]	John Eisenschmidt[3]
1. NCI - Center for Biomedical Informatics and Information Technology (CBIIT)	2. University of Chicago/Argonne National Laboratory	3. 5AM Solutions
4. Lockheed Martin Management System Designers.		

Other Acknowledgements
geWorkbench – Columbia University
GeneConnect – Project - Washington University
GridIMAGE – Project - Ohio State University
caBIO – Project - National Cancer Institute Center for Bioinformatics (NCICB)
caArray – Project - National Cancer Institute Center for Bioinformatics (NCICB)
caTRIP – Project – Duke Comprehensive Cancer Center

Contacts and Support	
NCICB Application Support	http://ncicb.nci.nih.gov/NCICB/support Telephone: 301-451-4384 Toll free: 888-478-4423

Chapter 1 Configuring and Deploying a caGrid Service

This chapter provides instructions for how to set up your application, configure security, and create and deploy a gateway service. The information in this chapter is designed to be generic enough to provide a customizable framework of procedures that you can use to deploy your own application onto caGrid.

Chapter 2, Deploying the geWorkbench Application, beginning on page 29 provides a specific example of the use of the instructions in this chapter, as they apply to deployment of the geWorkbench application on the grid.

NOTE: Throughout this chapter you will find active hyperlinks to websites and wiki sites that contain additional information regarding the task for which they appear. Click these links to access the site information and whatever additional instructions are being referenced.

Security for caGrid/TeraGrid Communication

This section focuses on configuring the proper security so that when finished, your configuration resembles the configuration displayed in the figure below.

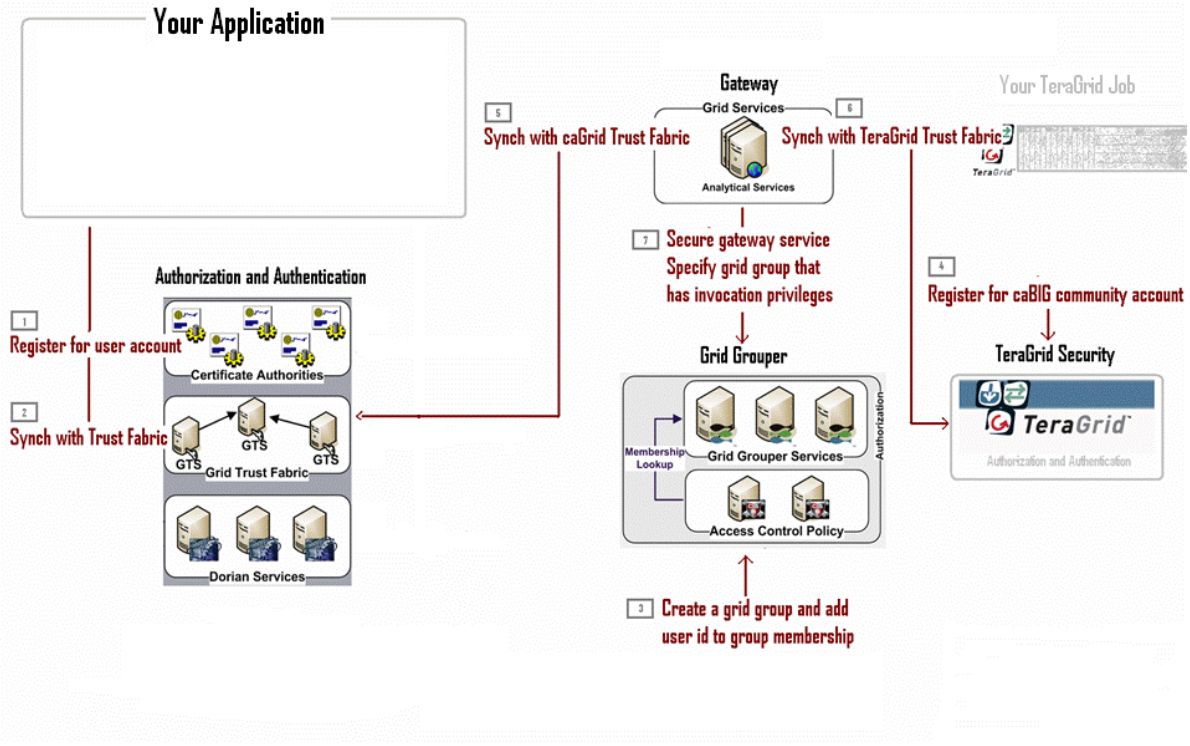

Figure 1-1: Security setup for caGrid deployment

Before diving into the setup process, bring up the caGrid GAARDS user interface (UI). If you do not have caGrid 1.1 installed, please do so using the following link: http://wiki.c2b2.columbia.edu/informatics/index.php/CaGrid_1.1-final.

NOTE: Throughout this document the location CAGRID_HOME refers to the directory where caGrid 1.1 is installed. For example c:\java\apps\caGrid\.

To open the GAARDS interface:

1. Navigate to CAGRID_HOME.
2. Type: `ant security`. The GAARDS window appears.

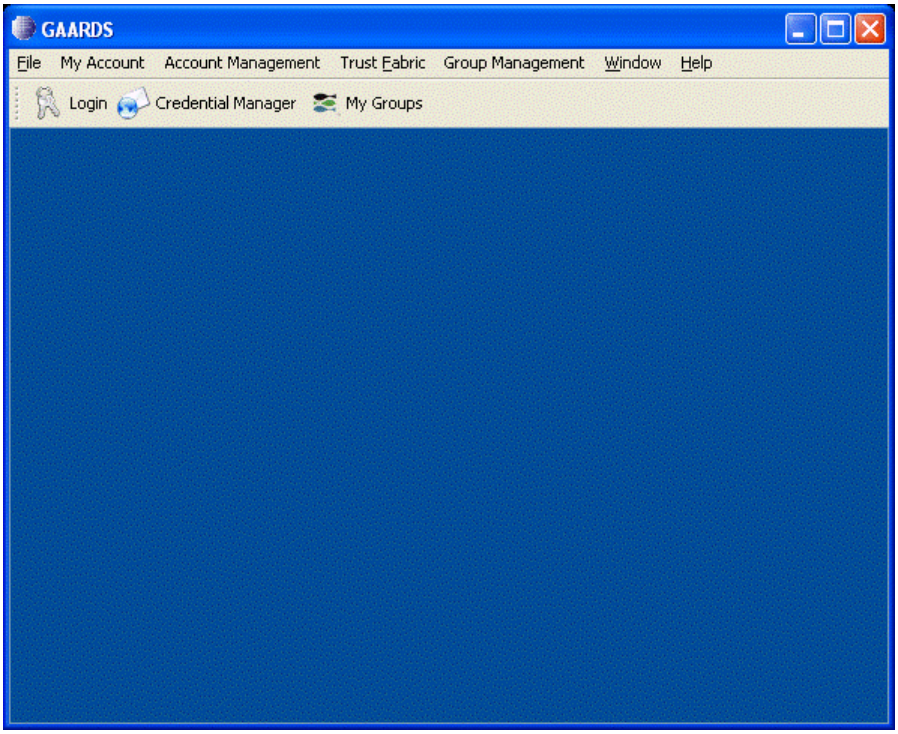

Figure 1-2: GAARDS Window

The sections that follow provide instructions for setting up proper security for caGrid-TeraGrid communications.

Configuring and Synchronizing Accounts

In this section you will create a caGrid account in case you don't have one already and after that you will configure other security configurations that are needed.

Step 1—Obtain caGrid User Account for the Application User

This step should be performed on the machine where you expect the application to be used and where GAARDS is open.

1. In GAARDS, open the local account registration form by selecting **Account Management > Local Accounts > Registration**. The Registration form appears.

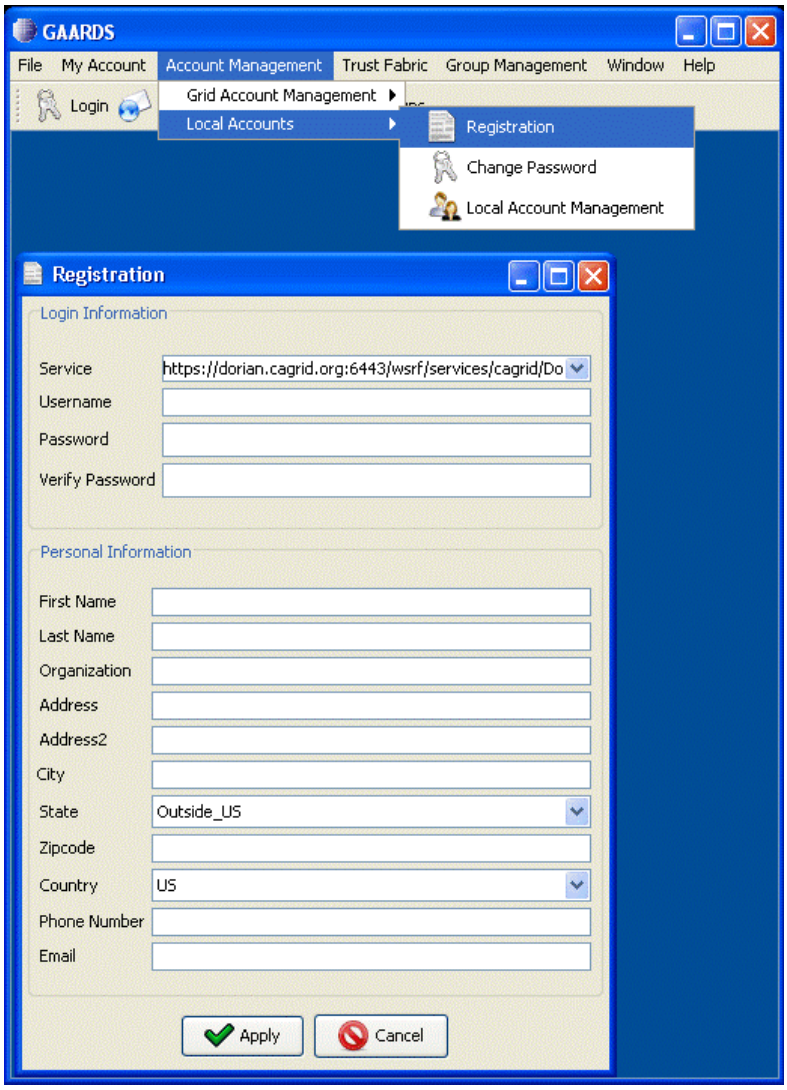

Figure 1-3: GAARDS Local Account Registration Form

2. Complete the fields in the form and click **Apply**. Be sure to remember the username and password entered for the account. You will need that information later.

Step 2—Synchronize User Credentials with caGrid Trust Fabric

This step should be performed on the machine where you expect the application to be used.

The procedure below allows you to first specify the caGrid with which you want to work, and then to synchronize the application's user credentials with the Trust Fabric of the selected caGrid.

1. Navigate to `CAGRID_HOME` and type:

```
ant -Dtarget.grid=<grid name> configure
```

   Possible grids names include: `nci_prod`, `nci_qa`, `nci_stage`, `nci_dev`, `osu_dev`, `training`, `custom_grid`.

2. Next, go to `CAGRID_HOME/projects/syncgts` and type:

```
ant syncWithTrustFabric
```

For more information on configuring caGrids, see how to change target grids. For more information on synchronizing with caGrid Trust Fabric, see caGrid Wiki on GTS.

Step 3—Create Grid Group and Include User ID in the caGrid Grid Grouper

This step should be performed on the machine the application is installed and where GAARDS is open.

The procedure in this section allows you to first obtain the user proxy and then include the user ID in the caGrid Grid Grouper.

To obtain the application user proxy:

1. In GAARDS, click **Login**.
2. Enter the appropriate URL into the **Dorian Service** text box. For example, if you obtained your user account on the Training grid, you must get its proxy from the Training Dorian.
3. Set **Lifetime** to 4 hours.

   NOTE: The 4 hours of proxy lifetime is used here because it coordinates with the delegation lifetime programmatically set when the geWorkbench application is deployed. See Step 2—Obtain the Delegation Reference on page 59 for those details.

4. Set the **Delegation Path Length** to 2.
5. Enter your User ID and Password. These are the application user account credentials registered in Step 1—Obtain caGrid User Account for the Application User above.

   When complete, the Login dialog box should appear similar to Figure 1-4 below.

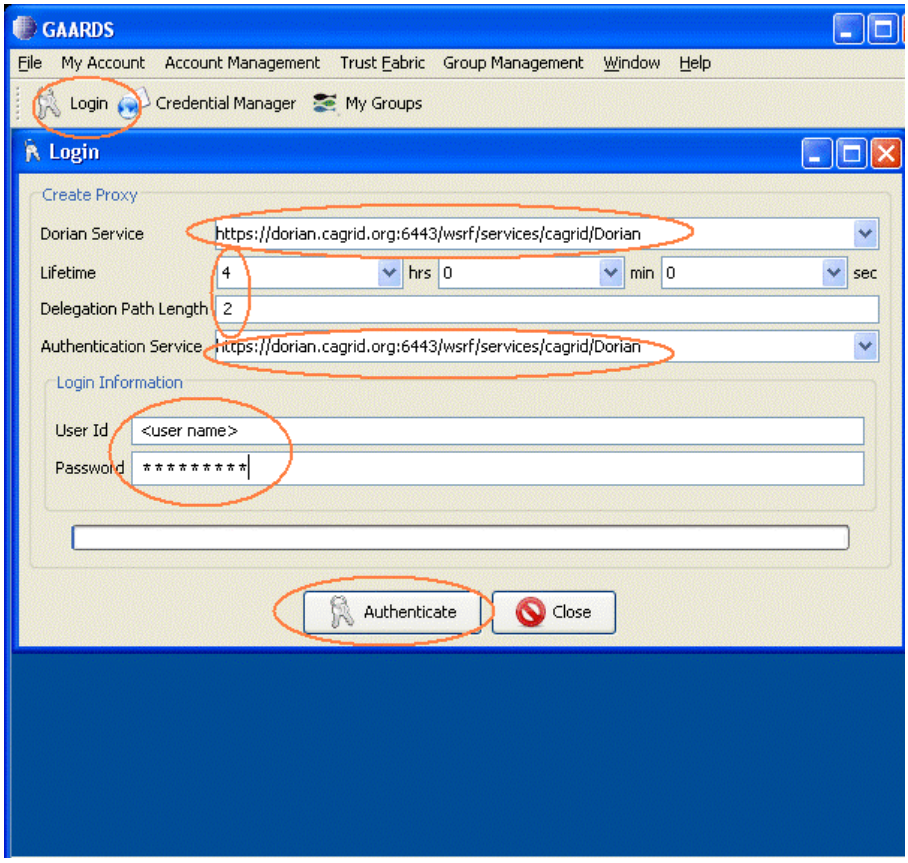

Figure 1-4: Completed Login dialog box

6. Click **Authenticate** in the completed Login dialog box. A Proxy Manager dialog box appears.

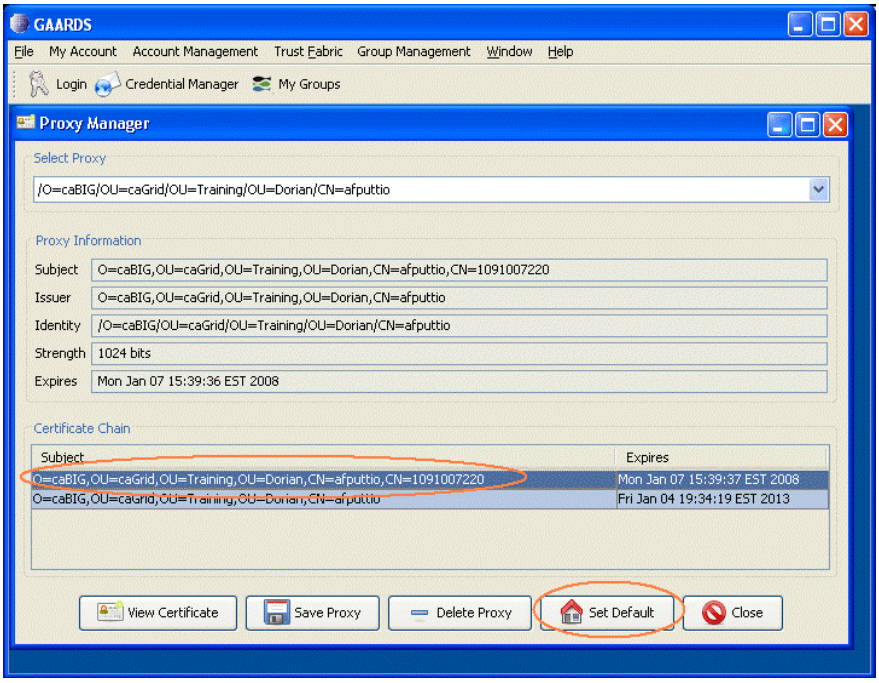

Figure 1-5: GAARDS Proxy Manager dialog box

7. In the Certificate Chain list, click on the certificate you just generated to highlight it.
8. Click **Set Default**.

To add the application User ID to the Grid Grouper:

1. From the main menu in GAARDS, select **Group Management > Group Browser**. The Group Browser window appears.
2. Click **Add Grid Grouper** located in the bottom-right of the Group Browser window.
3. In the Add Grid Grouper dialog box, use the drop-down lists to specify the appropriate Grid Grouper URL and the application user Credentials.
4. Click **Add**.

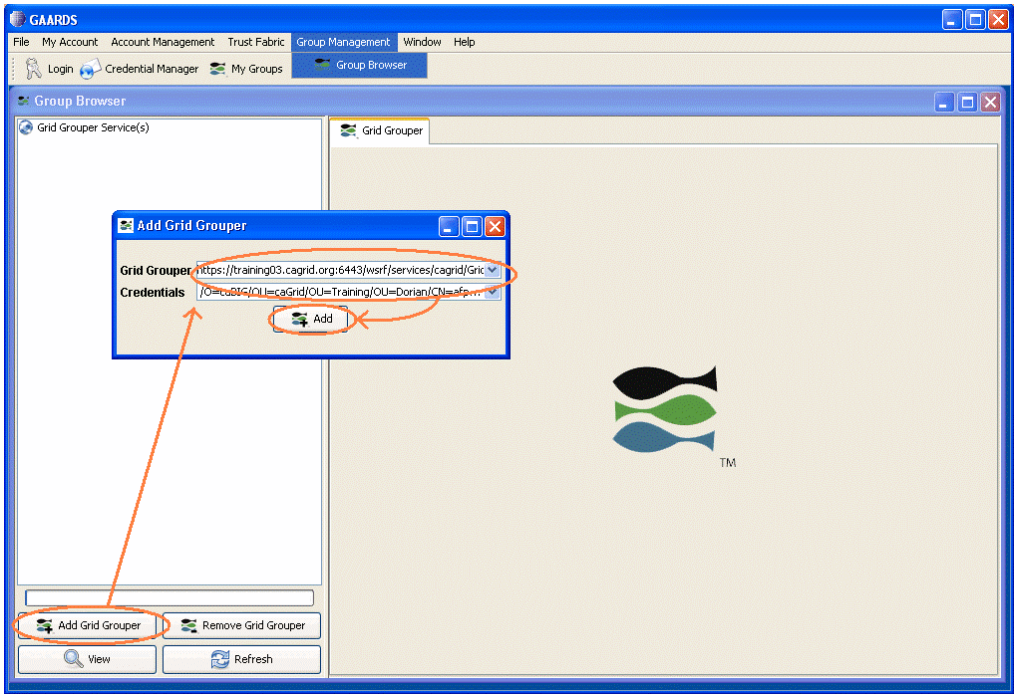

Figure 1-6: Group Browser window showing Add Grid Grouper dialog box

After the grid grouper service loads, it should appear in the Grid Grouper service explorer pane, located on the left side of the Group Browser window.

5. In the service explorer pane, find the grid group for this project, and double-click on the group name. This opens detailed information for the group in the right-side of the Group Browser window. The Details tab is active by default.

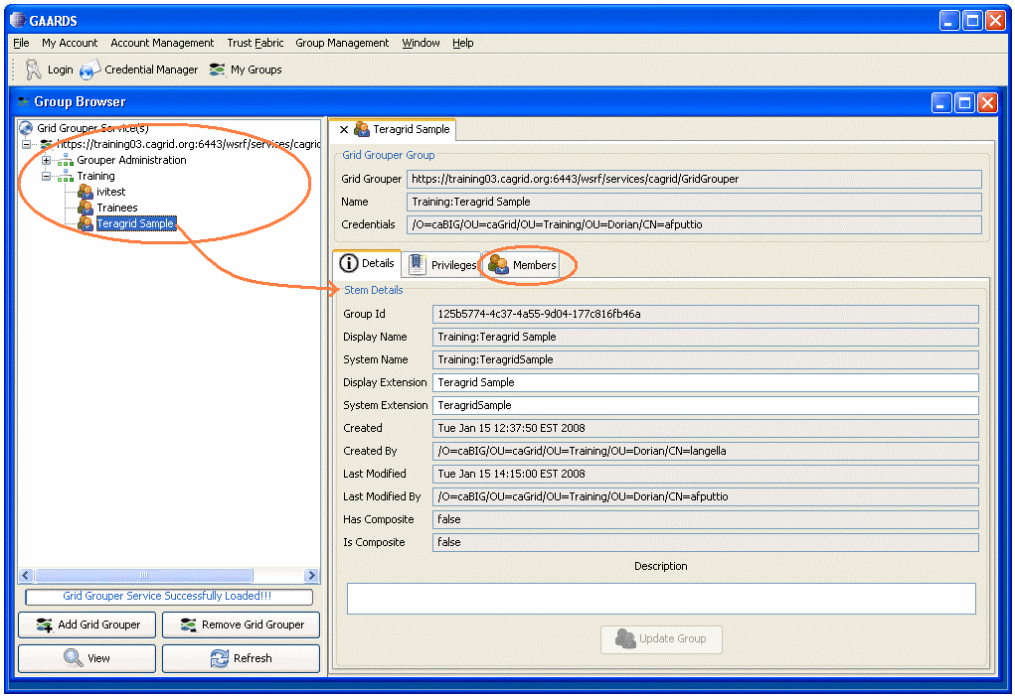

Figure 1-7: Group Browser window showing selected group details

6. Click on the Members tab to activate it.
7. Click **Add Member** at the bottom of the Members tab. The Add Member dialog box appears.

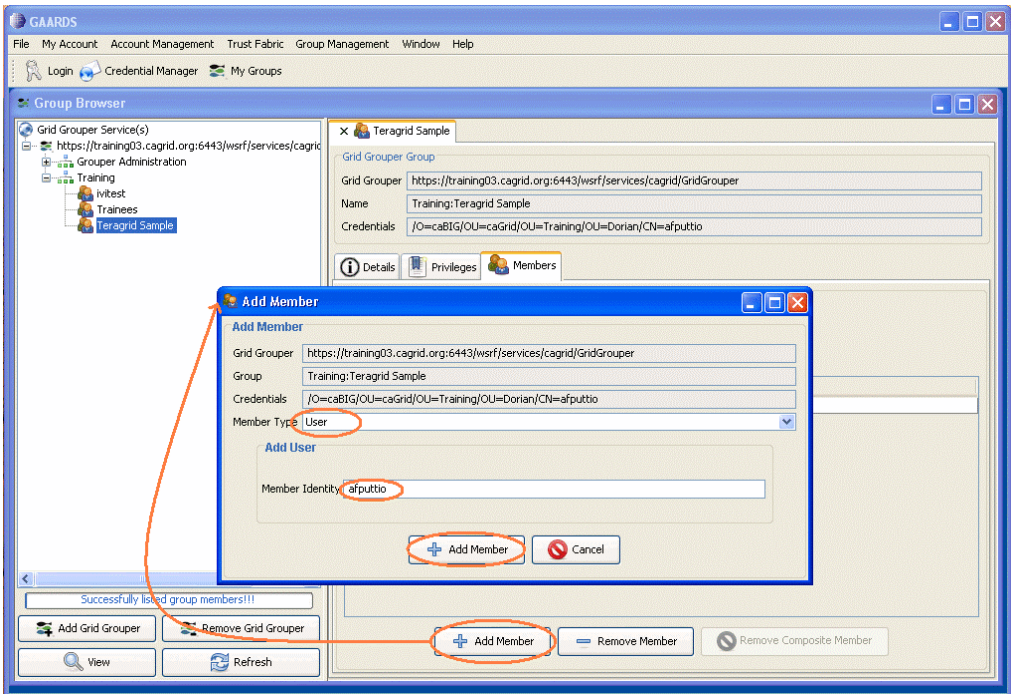

Figure 1-8: Add Member dialog box

8. From the Member Type drop-down list, select **User**.
9. In the Member Identity text box, type in the username of the member you want to add.
10. Click **Add Member**.

Once added, the user name appears in the Members Name list on the Members tab for the selected group.

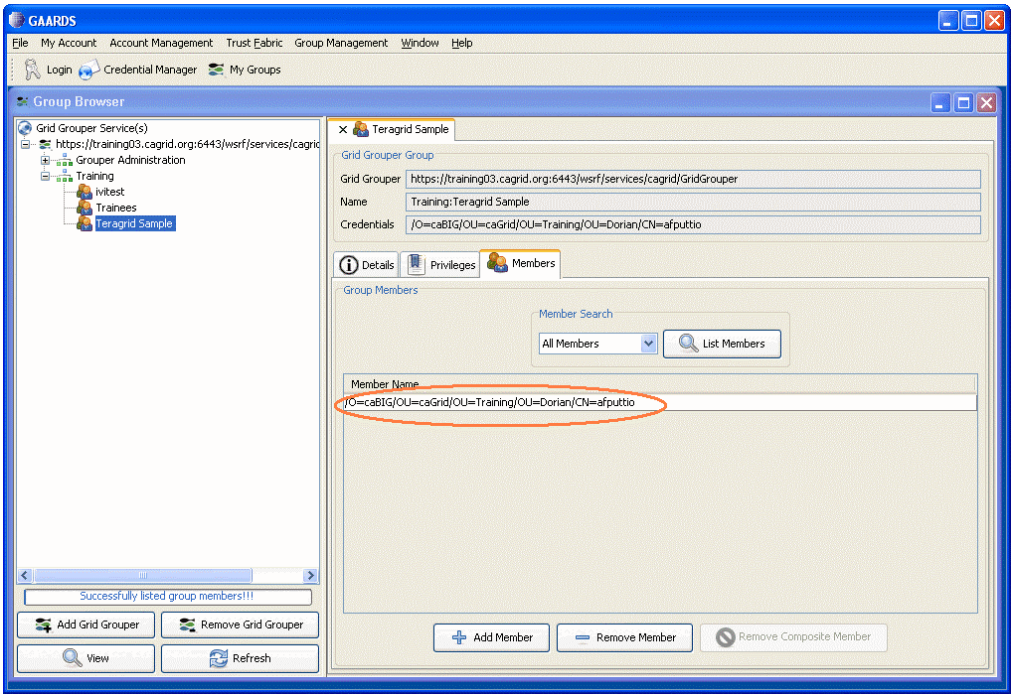

Figure 1-9: Group Browser Members tab showing added member

Step 4--Confirm caBIG community account for TeraGrid

Developers of gateway services can use the caBIG Community account to access TeraGrid resources; they do not have to obtain individual TeraGrid accounts. However if you are interested in obtaining a TeraGrid account, the following links provide further information and procedures:

- Apply for a TeraGrid account
- Set up community software area

Verify that you have access to the caBIG community account. You can do this by logging onto a login node. The example below shows the command for logging onto the San Diego cluster login node:

```
ssh -X <userNameForSanDiego>@dslogin.sdsc.edu
password:<SanDiegoPassword>
```

Once you have verified that you have access to the caBIG community account, use the sets of commands below to perform Single Sign on:

```
<userNameForSanDiego>@tg-grid1:~> myproxy-logon -T -l <userNameForUserPortal>
Enter MyProxy pass phrase:<userPortalPassword>
A credential has been received for user <userNameForUserPortal> in /tmp/x509up_u510.
Trust roots have been installed in /home/<userNameForUserPortal>/.globus/certificates/.
```

```
<userNameForUserPortal>@tg-grid1:~> grid-proxy-info
subject  : /C=US/O=National Center for Supercomputing Applications/ CN=<Name of User>
issuer   : /C=US/O=National Center for Supercomputing Applications/
OU=Certificate Authorities/CN=MyProxy
identity : /C=US/O=National Center for Supercomputing Applications/ CN=<Name of User>
type     : end entity credential
strength : 1024 bits
path     : /tmp/x509up_u510
timeleft : 11:59:37
```

```
<userNameForUserPortal>@tg-grid1:~> gsissh tg-login.ncsa.teragrid.org NCSA Teragrid Cluster
(MERCURY) --In Production with 868 nodes--
```

For additional information on available clusters and nodes, see TeraGrid Resources. (Click on the Resources tab and Systems Monitor sub-menu to get a list of host nodes.)

Step 5 -- Synchronize caBIG community credentials for TeraGrid with caGrid Trust Fabric

Use the username and password credentials established for your caBIG community account, and follow the basic steps outlined in Step 2—Synchronize User Credentials with caGrid Trust Fabric on page 12.

Step 6-- Synchronize caBIG community credentials for TeraGrid with TeraGrid Trust Fabric

Use the username and password credentials established for your caBIG community account, and follow the basic steps outlined in Step 2—Synchronize User Credentials with caGrid Trust Fabric on page 12, selecting a TeraGrid instead of a caGrid as the original instructions state.

Step 7-- Associate the Grid Group with the Gateway Service

Assuming the gateway service has been created, load it into Introduce via the **Modify Service** button. Then click the Security tab to activate it and show the sub-tabs available.

The Authorization sub-tab on the Security tab allows you to specify which grid grouper membership is allowed to access this gateway service.

NOTE: If the gateway service has not been created, please skip this step and follow the instructions located at Creating the caGrid Gateway Service on page 20.

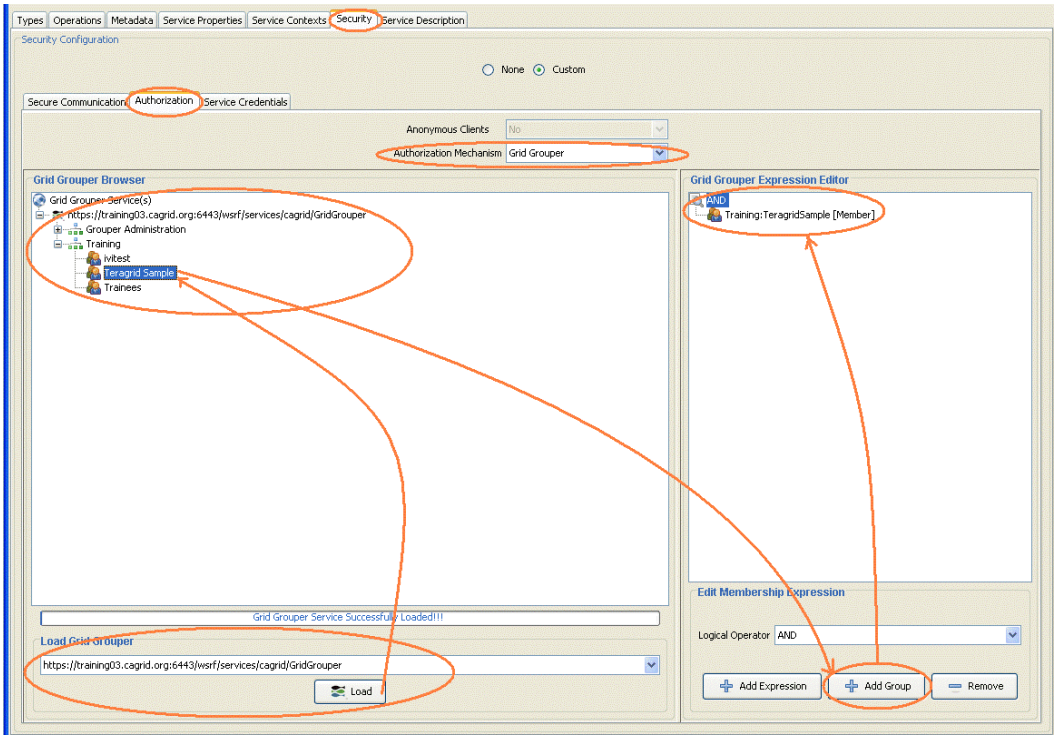

Figure 1-10: Introduce interface showing TeraGrid authorization

To authorize grid group access to the gateway service:

1. Click the Security tab to activate it, then click the Authorization sub-tab to display the Authorization options.
2. In the Authorization tab, select **Grid Grouper** from the Authorization Mechanism drop-down list.
3. Specify the grid grouper service URL from the Load Grid Grouper drop-down list and then click **Load** to bring up the list of available groups in the right side of the window.
4. Select the **TeragridSample** group from the list and then click **Add Group**.

The TeragridSample group is now added to the list of users authorized to invoke the gateway service.

For more information, see caGrid Grid Grouper.

Binary Staging on TeraGrid

Make sure you have access to the caBIG community account for TeraGrid or some other valid TeraGrid account. If not, see Step 4--Confirm caBIG community account for TeraGrid on page 17.

Stage your binary in the community software area. The University of Chicago is a good machine to use. To transfer your binary files, log in using the caBIG account, and use SCP to transfer the jar file.

To stage your binary in the community software area:

1. Make sure the node you plan to stage the binaries on contains the correct version of any software your binaries may need. For example, if your binaries use JDK 1.5 and you are using the University of Chicago node for staging, be sure it also contains JDK 1.5.
2. Create a command line interface and test your binary locally.
3. Test your binary in the community software area using the following steps:
   a. Log onto to a login node (e.g. sdsc) as detailed in Step 4--Confirm caBIG community account for TeraGrid on page 17.
   b. SCP your binary to the login machine.
   c. Perform single sign on as detailed in Step 4--Confirm caBIG community account for TeraGrid on page 17.
   d. SCP your binary from the login machine to your chosen cluster machine (e.g. tg-grid1.uc.teragrid.org).
   e. Verify that the necessary software is available to you.
   f. Run your binary again.

The log output should be the same as the local run.

Creating the caGrid Gateway Service

This section details how to create a caGrid gateway service.

Introduce and gRAVI

Use Introduce with the gRAVI plug-in to create the basis for the gateway service -- TeragridSample. The generated code should have the following structure:

Figure 1-11: Introduce-generated code structure (example)

In Introduce, the data types imported are:

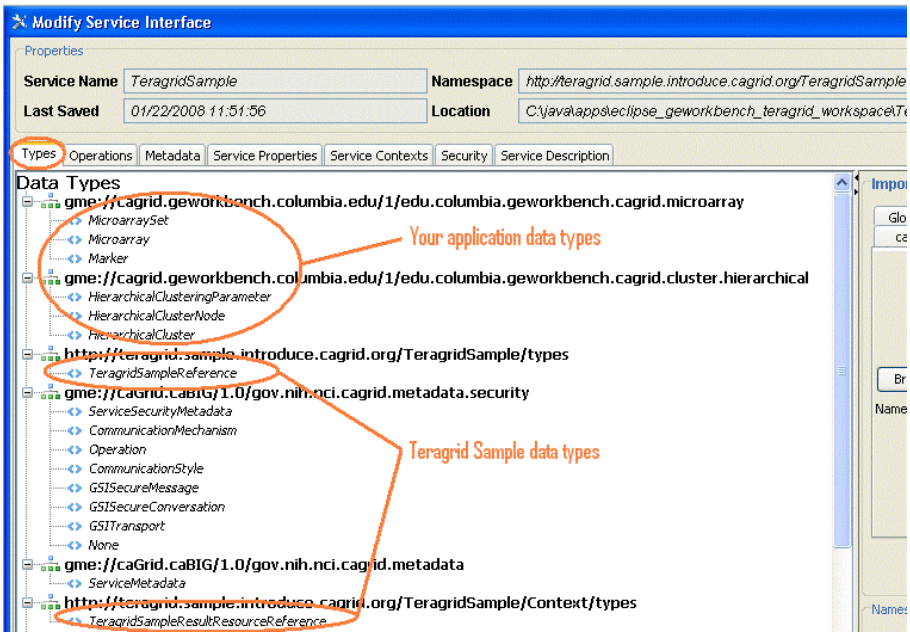

Figure 1-12: Introduce-imported data types (TeraGrid example)

The specified operations method is:

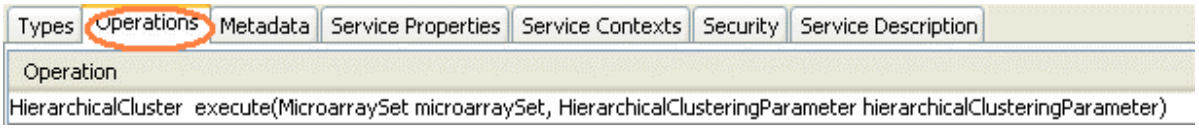

Figure 1-13: Introduce specified operations method (TeraGrid example)

In this case the one method the gateway client calls is `execute()`.

The Security tab in Introduce should be configured as shown in Figure 1-14, with the **Custom** option selected.

In the **Secure Communication** sub-tab, all three communications security options must be checked and all three Communication Method text boxes should show the value **Privacy**.

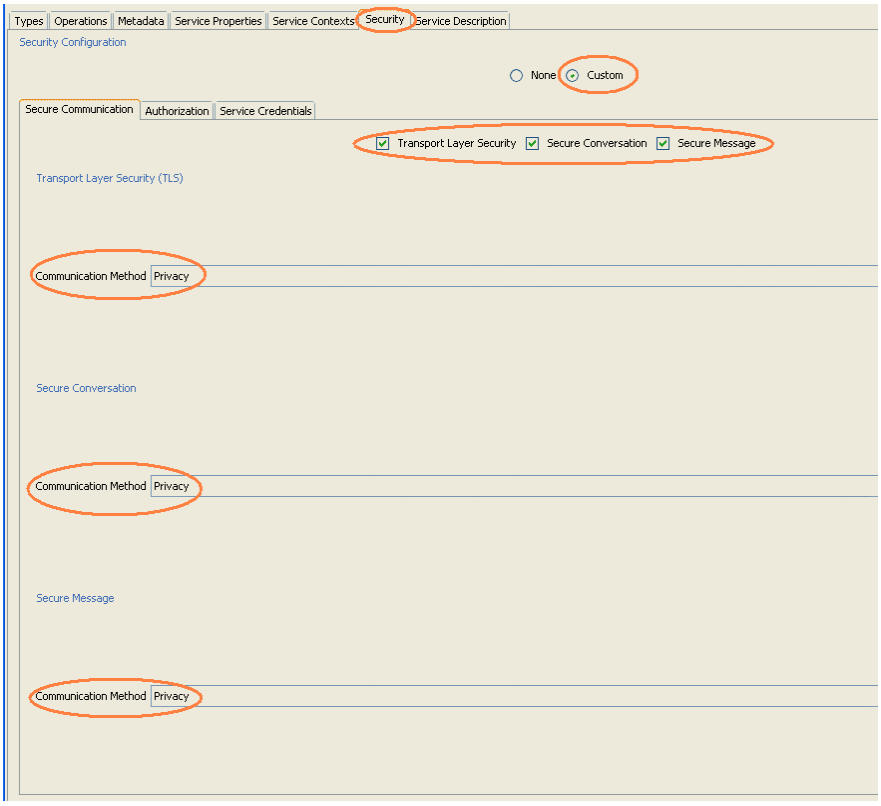

Figure 1-14: Introduce security options settings (TeraGrid example)

The **Authorization** sub-tab on the Security tab allows you to specify which grid grouper membership is allowed to access this gateway service.

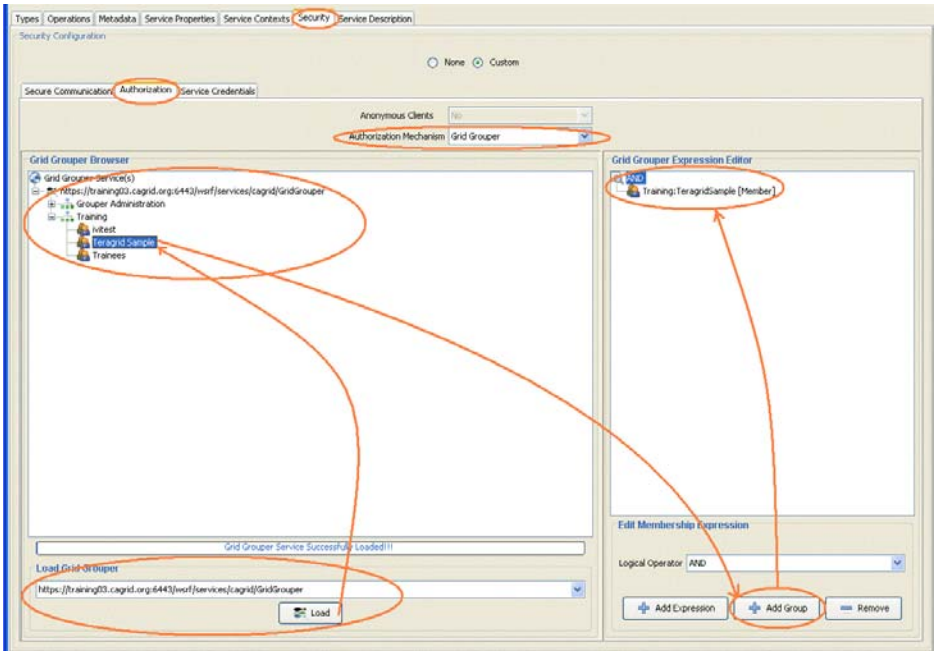

Figure 1-15: Introduce Authorization tab (TeraGrid example)

To authorize grid group access to the gateway service:

1. On the Security tab, click the Authorization sub-tab to show the Authorization options.
2. In the Authorization tab, select **Grid Grouper** from the Authorization Mechanism drop-down list.
3. Specify the Grid Grouper service URL from the Load Grid Grouper drop-down list and then click **Load** to bring up the list of available groups in the right side of the window.
4. Select the **TeragridSample** group from the list and then click **Add Group**.

The TeragridSample group is now added to the list of users authorized to invoke the gateway service. For more information, see caGrid Grid Grouper.

On the **Service Credentials** sub-tab of the Security tab, use the Import Credentials field to identify that the credentials are to be a Certificate/Private Key from the file system. Then enter System into the Run As field to run as the system.

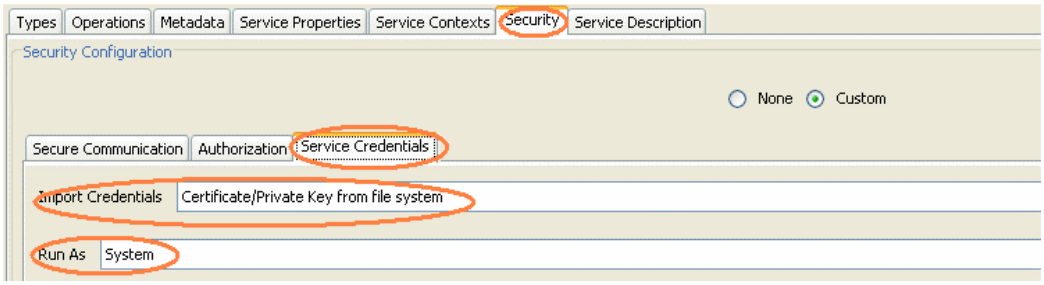

Figure 1-16: Introduce Service Credentials (TeraGrid example)

Gateway Client

The generated Gateway Client code is:
org.cagrid.introduce.sample.teragrid.client.TeragridSampleClient.java

The gateway client passes the application data, and any user selected parameters, to the gateway service hosted at `GATEWAY_SERVICE_URL`. Both the data and the parameters are should be caDSR compliant data structures.

```
TeragridSampleClient client = new TeragridSampleClient(GATEWAY_SERVICE_URL);
<ApplicationResult app> = client.execute(getData(), getParameters());
```

Gateway Service

The generated Gateway Service code is:
org.cagrid.introduce.sample.teragrid.context.service.TeragridSampleImpl.java

Make sure the following jars are on the classpath:

```
GLOBUS_LOCATION/lib/axis.jar
GLOBUS_LOCATION/lib/gram-client.jar
GLOBUS_LOCATION/lib/gram-stubs.jar
GLOBUS_LOCATION/lib/gram-util.jar
GLOBUS_LOCATION/lib/gram-monitoring.jar
GLOBUS_LOCATION/lib/wsrf-core.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
GLOBUS_LOCATION/lib/jgss.jar
```

Import the following classes:

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.RandomAccessFile;
import java.net.URL;
import java.rmi.RemoteException;
import java.util.Date;
import org.apache.axis.components.uuid.UUIDGen;
import org.apache.axis.components.uuid.UUIDGenFactory;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.globus.exec.client.GramJob;
import org.globus.exec.generated.JobDescriptionType;
import org.globus.exec.generated.StateEnumeration;
import org.globus.exec.utils.ManagedJobFactoryConstants;
import org.globus.exec.utils.client.ManagedJobFactoryClientHelper;
import org.globus.wsrf.impl.security.authentication.Constants;
import org.globus.wsrf.impl.security.authorization.Authorization;
import org.globus.wsrf.impl.security.authorization.HostAuthorization;
import org.gridforum.jgss.ExtendedGSSCredential;
import org.gridforum.jgss.ExtendedGSSManager;
import org.ietf.jgss.GSSCredential;
import org.globus.exec.client.GramJobListener;
import org.globus.ftp.DataChannelAuthentication;
import org.globus.ftp.DataSink;
import org.globus.ftp.DataSource;
import org.globus.ftp.FileRandomIO;
import org.globus.ftp.GridFTPClient;
import org.globus.ftp.GridFTPSession;
import org.globus.gsi.GlobusCredential;
import org.globus.gsi.GlobusCredentialException;
import org.globus.gsi.gssapi.GlobusGSSCredentialImpl;
```

Make sure the class implements GramJobListener interface:

```
public class TeragridSampleImpl extends TeragridSampleImplBase  implements GramJobListener {
```

The GramJobListener interface has one method:

```
public void stateChanged(GramJob job) {
StateEnumeration jobState = job.getState();
boolean holding = job.isHolding();
printMessage("========= State Notification =========");
printJobState(jobState, holding);
printMessage("====================================");
synchronized (this) {
if ( jobState.equals(StateEnumeration.Done)|| jobState.equals(StateEnumeration.Failed)) {
printMessage("Exit Code: " + Integer.toString(job.getExitCode()));
// also call code to start processing results, for example gridFTP them from TeraGrid back to
gateway service
this.jobCompleted = true;
}
notifyAll();
}
}
```

Authenticate with TeraGrid using caBIG community credentials. In the following sample, `PROXY_LOCATION` is the location of the TeraGrid user certificate.

```
GlobusCredential globusCred = new GlobusCredential(PROXY_LOCATION);
GlobusGSSCredentialImpl cred = new GlobusGSSCredentialImpl(globusCred,
GSSCredential.INITIATE_AND_ACCEPT);
```

Set up gridFTP info:

```
String DEST_HOST = <name of the host in TeraGrid with the gridFTP service>
int GRIDFTP_PORT = 2811;
```

GridFTP input data and parameters:

```
String FILE_TO_FTP = <full path to the input data file/parameter file>;
```

```
GridFTPClient client = new GridFTPClient(DEST_HOST, GRIDFTP_PORT);
client.authenticate(creds);
client.setProtectionBufferSize(16384);
client.setType(GridFTPSession.TYPE_IMAGE);
client.setMode(GridFTPSession.MODE_EBLOCK);
client.setDataChannelAuthentication(DataChannelAuthentication.NONE);
client.setDataChannelProtection(GridFTPSession.PROTECTION_SAFE);
client.setPassive();
client.setLocalActive();
DataSource source = new FileRandomIO(new RandomAccessFile(new File(FILE_TO_FTP), "r"));
client.extendedPut(FILE_TO_FTP, source, null);
client.close();
```

Submit the TeraGrid Job

See also TeraGrid wiki on submitting GRAM jobs via Java.

Break up the original command line to invoke the staged binary to:

```
String [] arguments = new String [...];
arguments [0] = "...";
arguments [1] = "...";
...
```

Allow delegation from client -> gRAVi service -> GRAM

```
boolean limitedDelegation = true;
boolean delegationEnabled = true;
```

Set service dates:

```
Date serviceDuration = <set service duration>
Date serviceTermination= <set service termination date>
```

Submit the job:

```
JobDescriptionType jobDescription = createJobDescription(PATH_TO_JAVA_BIN +
"/java", arguments);
GramJob job = new GramJob(jobDescription);
job.setTimeOut(GramJob.DEFAULT_TIMEOUT);
job.setAuthorization(HostAuthorization.getInstance());
job.setMessageProtectionType(Constants.ENCRYPTION);
job.setDelegationEnabled(delegationEnabled);
job.setDuration(serviceDuration);
job.setTerminationTime(serviceTermination);
// GRAMContact and GRAMType are both specified in rave.properties
EndpointReferenceType factoryEndpoint = getFactoryEPR(GRAMContact,
GRAMType);
ExtendedGSSManager manager = (ExtendedGSSManager)
ExtendedGSSManager.getInstance();
// proxyPath is specified in rave.properties
String handle = "X509_USER_PROXY=" + proxyPath.toString();
GSSCredential proxy = manager.createCredential(handle.getBytes(),
ExtendedGSSCredential.IMPEXP_MECH_SPECIFIC, GSSCredential.DEFAULT_LIFETIME,
null, GSSCredential.INITIATE_AND_ACCEPT);
job.setCredentials(proxy);
String submissionID = "uuid:" + uuidgen.nextUUID();
job.addListener(this);
job.submit(factoryEndpoint, false, limitedDelegation, submissionID);
```

For details on `rave.properties`, see the wiki on the gRAVI Plug-in.

GridFTP to Retrieve Results:

```
String FILE_TO_FTP = <full path to the result data file>;
```

```
GridFTPClient client = new GridFTPClient(DEST_HOST, GRIDFTP_PORT);
client.authenticate(cred);
client.setProtectionBufferSize(16384);
client.setType(GridFTPSession.TYPE_IMAGE);
client.setMode(GridFTPSession.MODE_EBLOCK);
client.setDataChannelAuthentication(DataChannelAuthentication.SELF);
client.setDataChannelProtection(GridFTPSession.PROTECTION_SAFE);
client.setLocalPassive();
client.setActive();
DataSink sink = new FileRandomIO(new RandomAccessFile(new File(FILE_TO_FTP), "rw"));
client.get(FILE_TO_FTP, sink, null);
client.close();;
```

Deploying the Gateway Service

A gateway service must be deployed in a secured caGrid container. You can either configure tomcat to be a secured container or install globus as the secured container.

In Introduce's Deploy Grid Service dialog box, click **Deploy**.

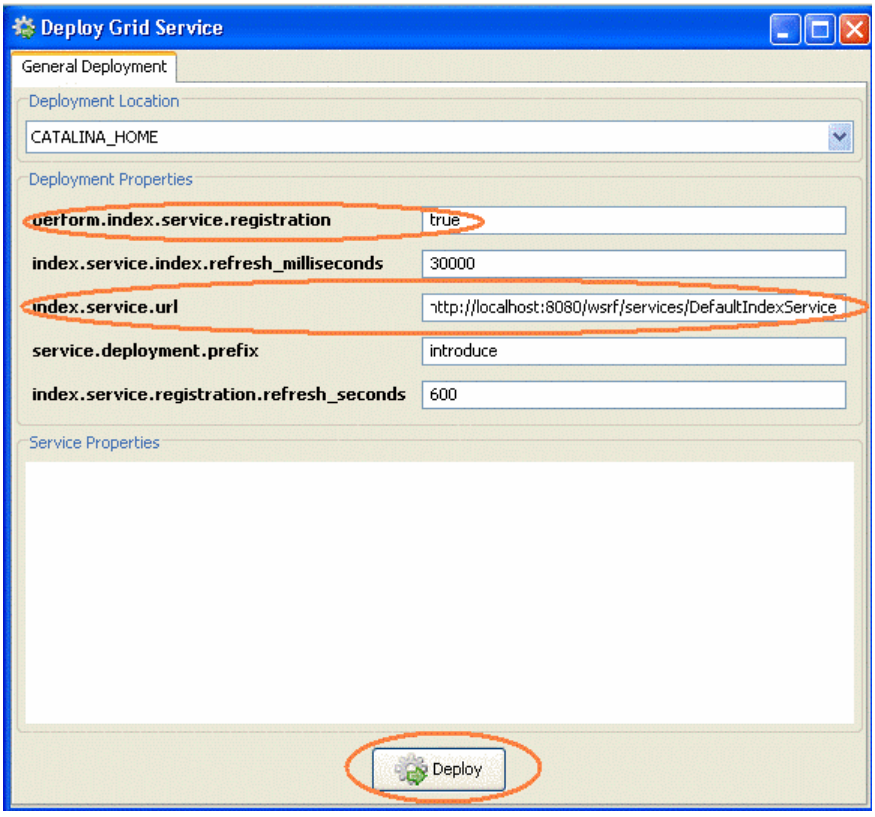

Figure 1-17: Introduce Deploy Grid Service dialog box

Use the URL to the gateway service the when you instantiate your gateway client. For specifics, see Gateway Client on page 23.

caGrid-TeraGrid Security

Review the following bulleted items. If the procedures provided above have been performed properly, the execution of these operations should be automatic.

- **Verify application user membership** – As long as the Introduce and gRAVI step (detailed on page 20) was performed appropriately, specifically with respect to Introduce's **Modify Service > Security > Authorization Tab**, this step is automatic when the gateway client invokes the gateway service.
- **Obtain proxy for caBIG community account** – As long as the Gateway Service step (beginning on page 24) includes code to authenticate either the caBIG community account or some other valid TeraGrid account, this step is automatic when the gateway service is invoked.
- **Use proxy for caBIG community account** – As long as the Gateway Service step (beginning on page 24) includes code to obtain proxy after authentication with TeraGrid, this step is automatic when the gateway service is invoked.

You may also want to refer to Security for caGrid/TeraGrid Communication on page 9.

Chapter 2 Deploying the geWorkbench Application

This chapter provides the instructions used to deploy the geWorkbench application to caGrid. Specifically the purpose of this chapter is to:

- Demonstrate how to submit jobs to TeraGrid via caGrid gateway services. This configuration allows computationally intensive analysis to run on resources that cater to heavy computation.
- Document best practices on setting up a TeraGrid-Aware caGrid gateway service. (See Deliverables for more information.)
- Demonstrate the configuration process using the geWorkbench Hierarchical Clustering analysis component.

Although the new service is TeraGrid-aware, the perspective from geWorkbench does not change. As far as geWorkbench is concerned, it is still connecting to a Hierarchical Clustering caGrid service.

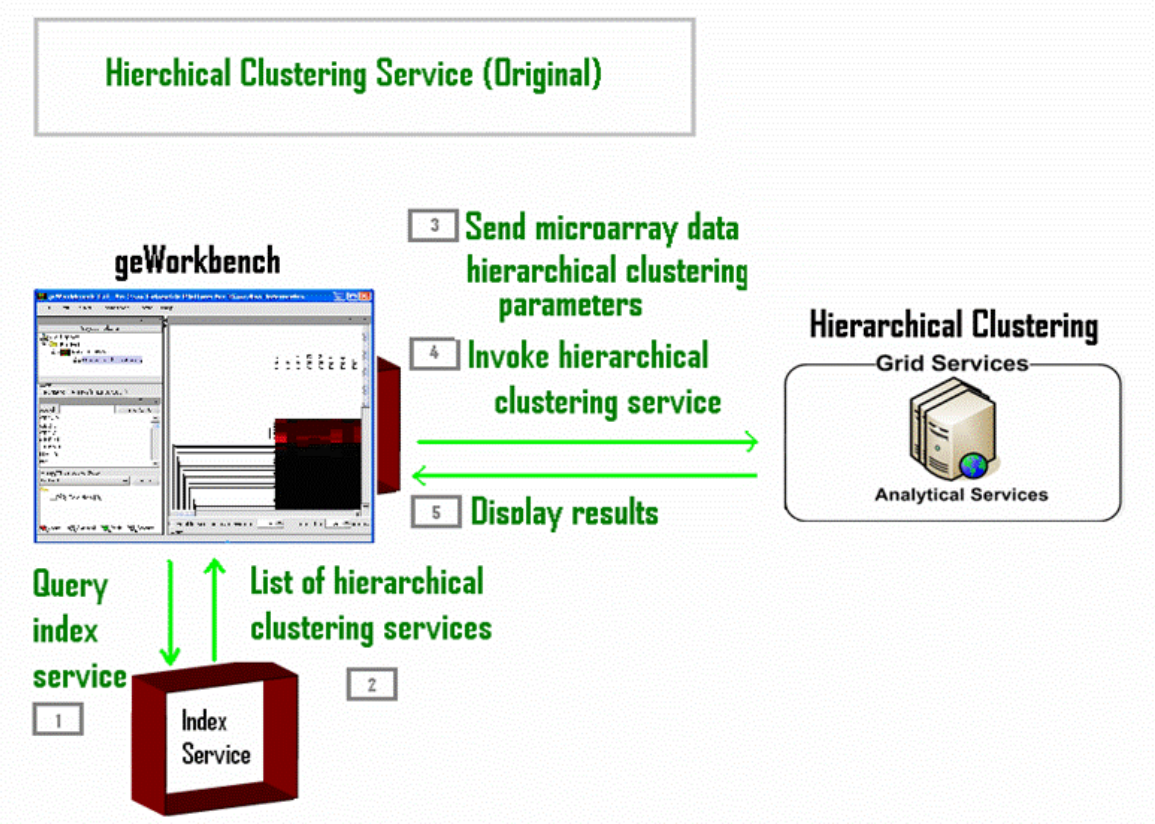

Figure 2-1: Diagram of a hierarchical clustering service using geWorkbench

The difference is now the caGrid service is a gateway service that submits a TeraGrid job on behalf of geWorkbench; geWorkbench, however, does not notice this difference.

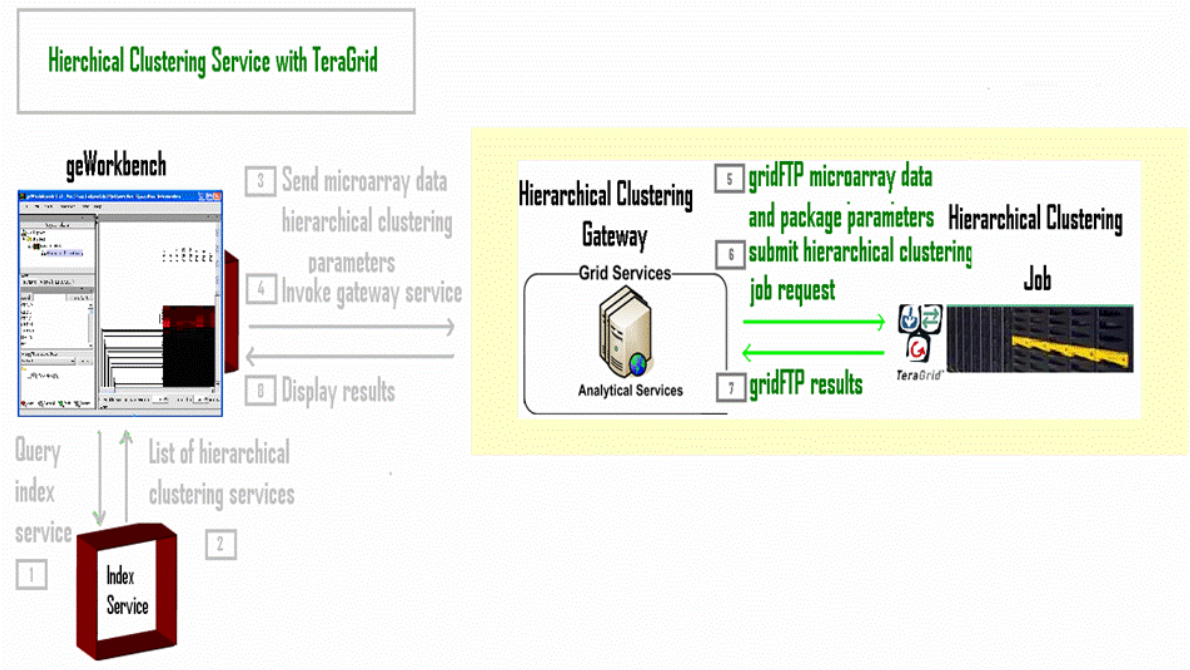

Figure 2-2: Diagram of a hierarchical clustering service with TeraGrid

For more generic information and instructions as they apply to the deployment of an application to caGrid/TeraGrid, see *Chapter 1,* Configuring and Deploying a caGrid Service beginning on page 9.

Security for caGrid-TeraGrid Communication

This section focuses on configuring the proper security so that your configuration resembles the configuration displayed in the figure below.

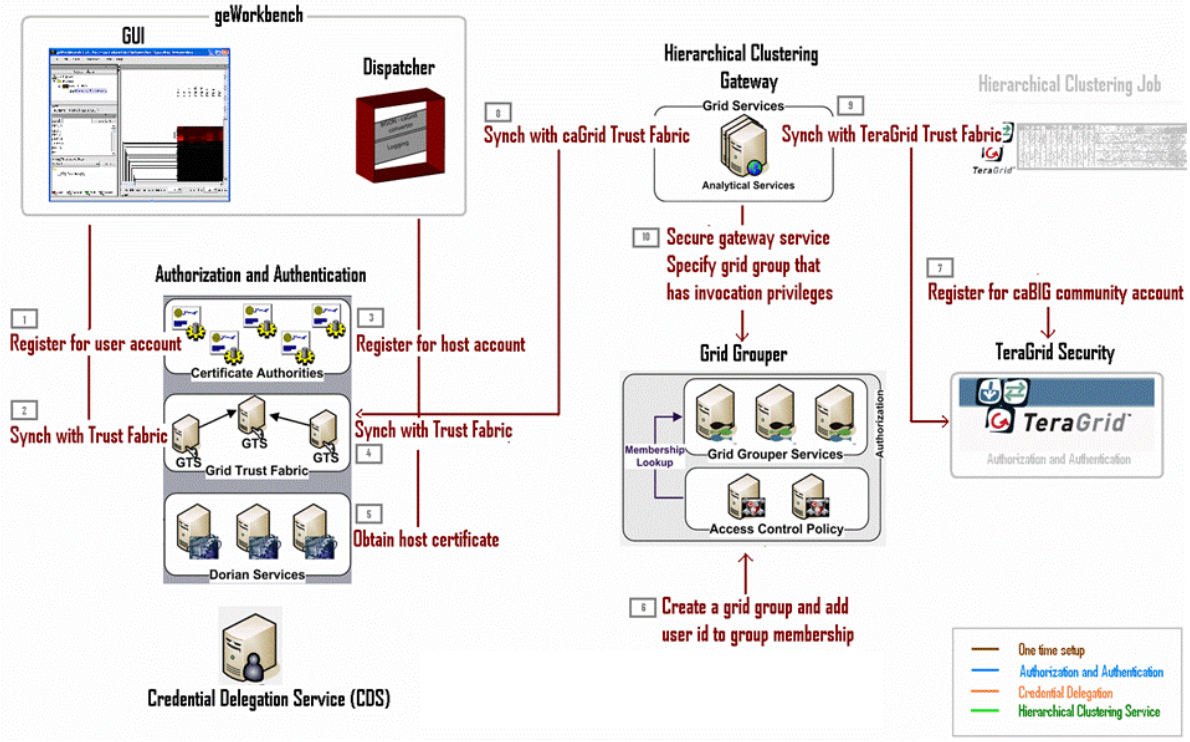

Figure 2-3: Security setup for caGrid deployment of geWorkbench

Before diving into the setup process, bring up the caGrid GAARDS user interface (UI). If you do not have caGrid 1.1 installed, please do so using the following link: http://wiki.c2b2.columbia.edu/informatics/index.php/CaGrid_1.1-final.

NOTE: Throughout this document the location CAGRID_HOME refers to the directory where caGrid 1.1 is installed. For example c:\java\apps\caGrid\.

To open the GAARDS interface:

1. Navigate to `CAGRID_HOME`.
2. Type: `ant security`. The GAARDS window appears.

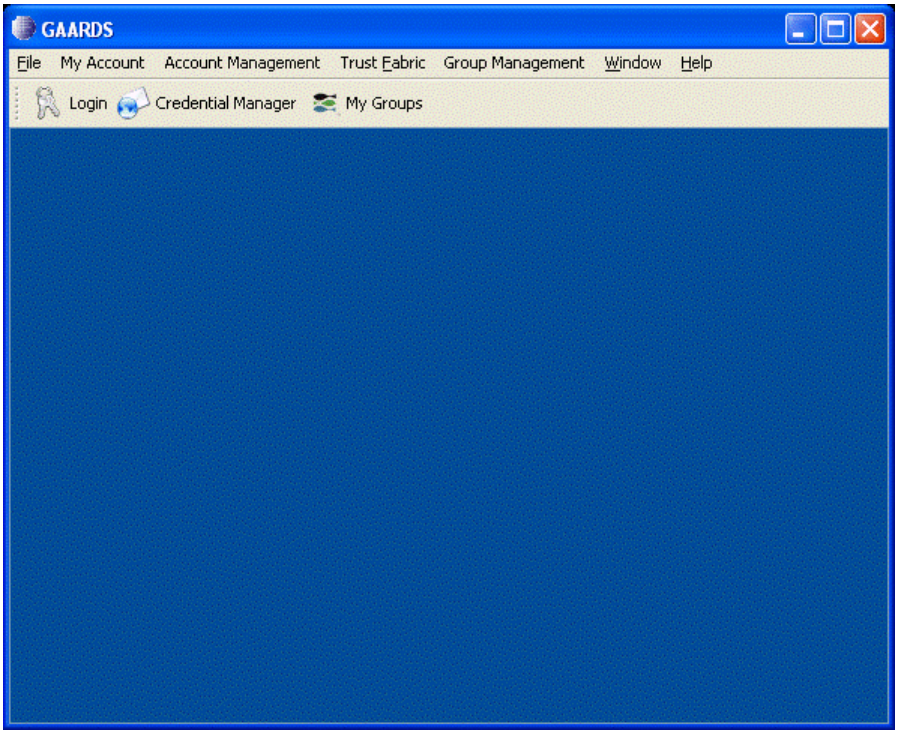

Figure 2-4: GAARDS Interface

The sections that follow include the procedures used for setting up proper security for caGrid-TeraGrid communications with geWorkbench.

Configuring and Synchronizing Accounts

In this section you will create a caGrid account in case you don't have one already and after that you will configure other security configurations that are needed.

Step 1—Obtain caGrid User Account for the Application User

This step should be performed on the machine where you expect the application to be used and where GAARDS is open.

1. In GAARDS, open the local account registration form by selecting **Account Management > Local Accounts > Registration**. The Registration form appears.

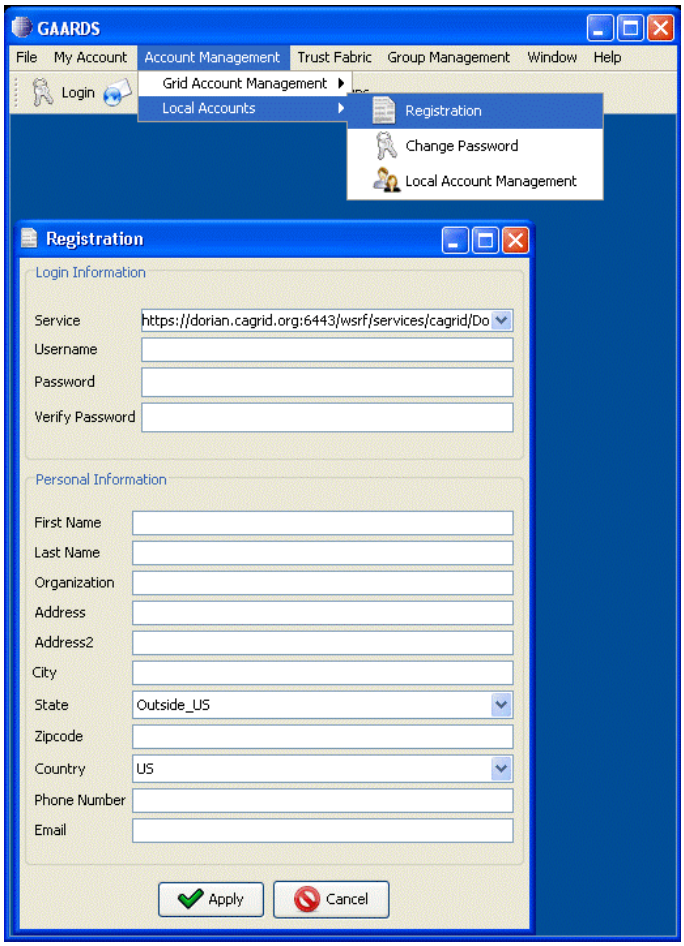

Figure 2-5: GAARDS Local Account Registration Form

2. Complete the fields in the form and click **Apply**. Be sure to remember the username and password entered for the account. You will need that information later.

Step 2—Synchronize User Credentials with caGrid Trust Fabric

This step should be performed on the machine where you expect the geWorkbench GUI to be used.

The procedure below allows you to first specify the caGrid with which you want to work, and then to synchronize the geWorkbench user credentials with the Trust Fabric of the selected caGrid.

1. Navigate to `CAGRID_HOME` and type:

```
ant -Dtarget.grid=<grid name> configure
```

   Possible grids names include: `nci_prod`, `nci_qa`, `nci_stage`, `nci_dev`, `osu_dev`, `training`, `custom_grid`.

2. Next, go to `CAGRID_HOME/projects/syncgts` and type:

```
ant syncWithTrustFabric
```

For more information on configuring caGrids, see how to change target grids. For more information on synchronizing with caGrid Trust Fabric, see caGrid Wiki on GTS.

Step 3—Obtain caGrid Host Account For geWorkbench Dispatcher

This step is the same as Step 1 above but is instead performed on the machine that hosts the geWorkbench dispatcher.

1. In GAARDS, open the local account registration form by selecting **Account Management > Local Accounts > Registration**. The Registration form appears.

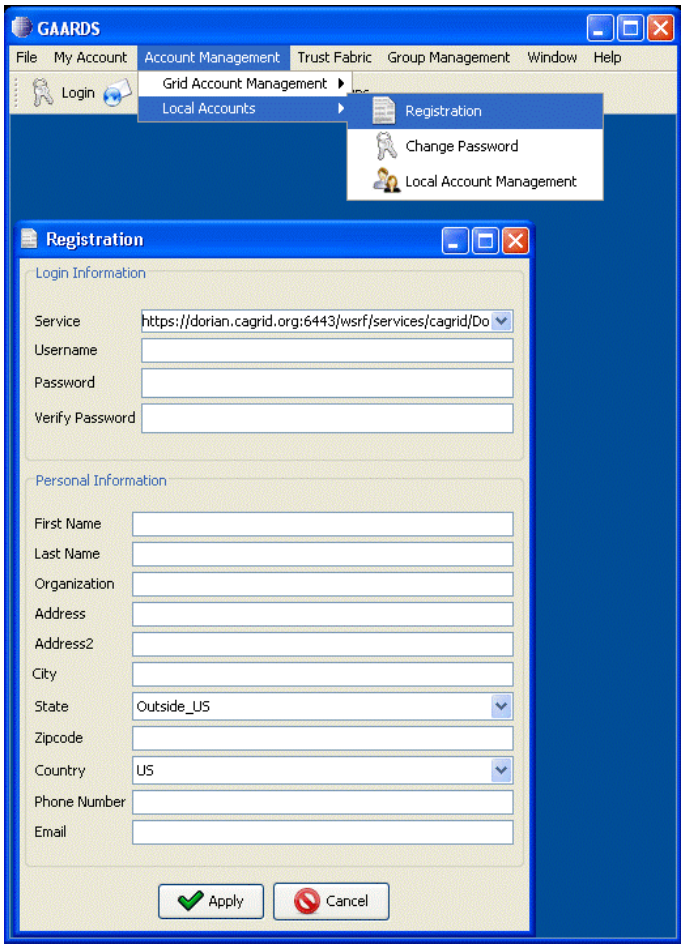

Figure 2-6: GAARDS Local Account Registration Form

2. Complete the fields in the form and click **Apply**. Be sure to remember the username and password entered for the account. You will need that information later.

Step 4—Synchronize geWorkbench Dispatcher User Credentials with caGrid Trust Fabric

This step is the same as Step 2 above but is instead performed on the machine that hosts the geWorkbench dispatcher.

The procedure below allows you to first specify the caGrid with which you want to work, and then to synchronize the geWorkbench dispatcher user credentials with the Trust Fabric of the selected caGrid.

1. Navigate to `CAGRID_HOME` and type:

```
ant -Dtarget.grid=<grid name> configure
```

Possible grids names include: `nci_prod`, `nci_qa`, `nci_stage`, `nci_dev`, `osu_dev`, `training`, `custom_grid`. Next, go to `CAGRID_HOME/projects/syncgts` and type:

```
ant syncWithTrustFabric
```

For more information on configuring caGrids, see how to change target grids. For more information on synchronizing with caGrid Trust Fabric, see caGrid Wiki on GTS.

Step 5—Obtain Host Proxy for geWorkbench Dispatcher

This step is performed on the machine that hosts geWorkbench dispatcher.

1. In GAARDS, open the Request Host Certificate form by selecting **My Account >Request Host Certificate**. The request form appears.

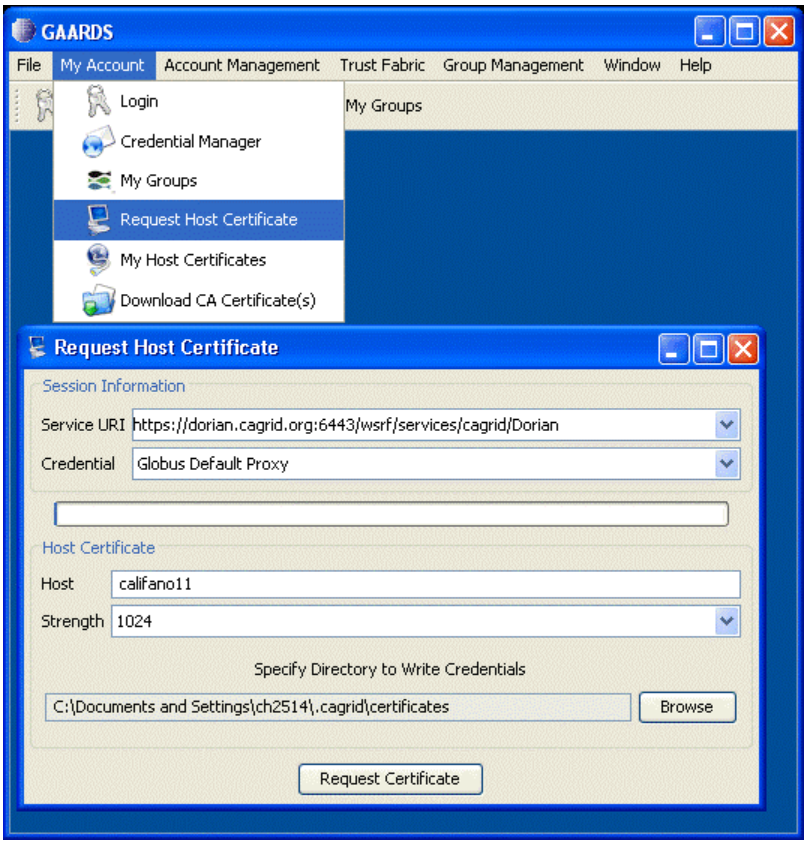

Figure 2-7: GAARDS Request Host Certificate Form

2. From the Service URI drop-down list, select the service address used for registering the geWorkbench dispatcher account.
3. Select Globus Default Proxy from the Credential drop-down list.
4. If necessary, enter the Host Certificate information.

5. Click **Browse** to navigate to the directory where you want the certificate credentials to be written.
6. Click **Request Certificate**.

Step 6—Include User ID in the caGrid Grid Grouper

This step should be performed on the machine where geWorkbench is expected to be used.

The procedure in this section allows you to first obtain the user proxy and then include the user ID in the caGrid Grid Grouper.

To obtain the application user proxy:

1. In GAARDS, click **Login**.
2. Enter the appropriate URI into the **Dorian Service** text box. For example, if you obtained your user account on the Training grid, you must get its proxy from the Training Dorian.
3. Set **Lifetime** to 4 hours.

   NOTE: The 4 hours of proxy lifetime is required to coordinate with the delegation lifetime that is set programmatically in Step 2—Obtain the Delegation Reference detailed on page 59.
4. Set the **Delegation Path Length** to 2.
5. Enter your User ID and Password. These are the geWorkbench application user account credentials registered in Step 1—Obtain caGrid User Account for the Application User on page *33*.

   When complete, the Login dialog box should appear similar to Figure 2-8 below.

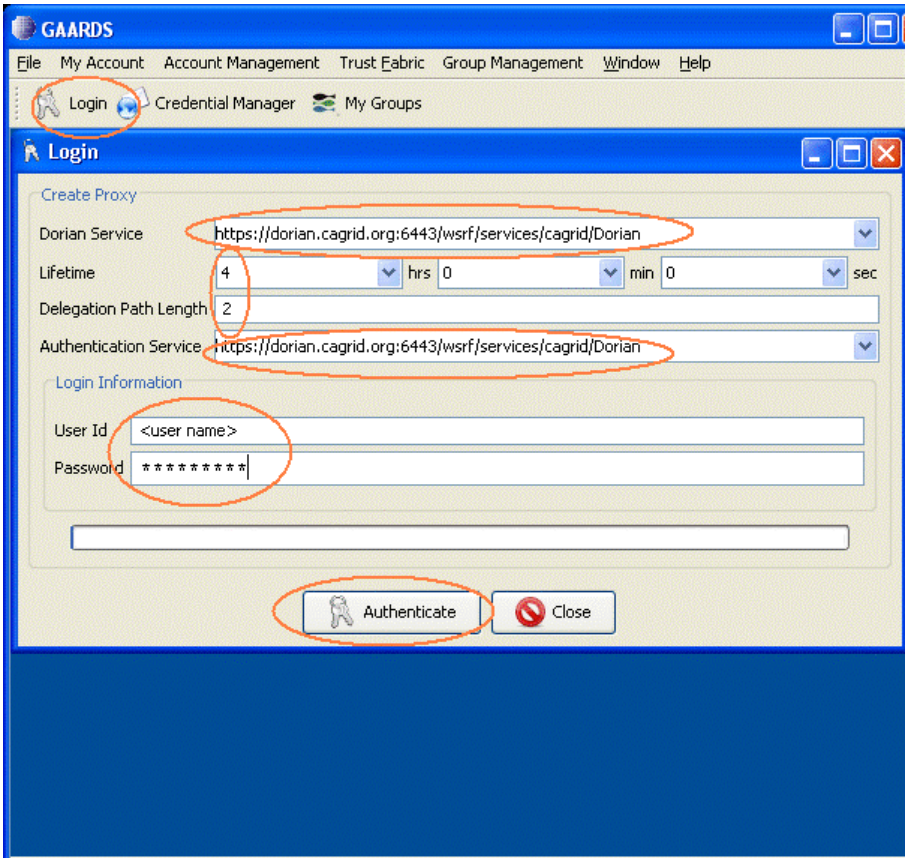

Figure 2-8: Completed Login dialog box

6. Click **Authenticate** in the completed Login dialog box. A Proxy Manager dialog box appears.

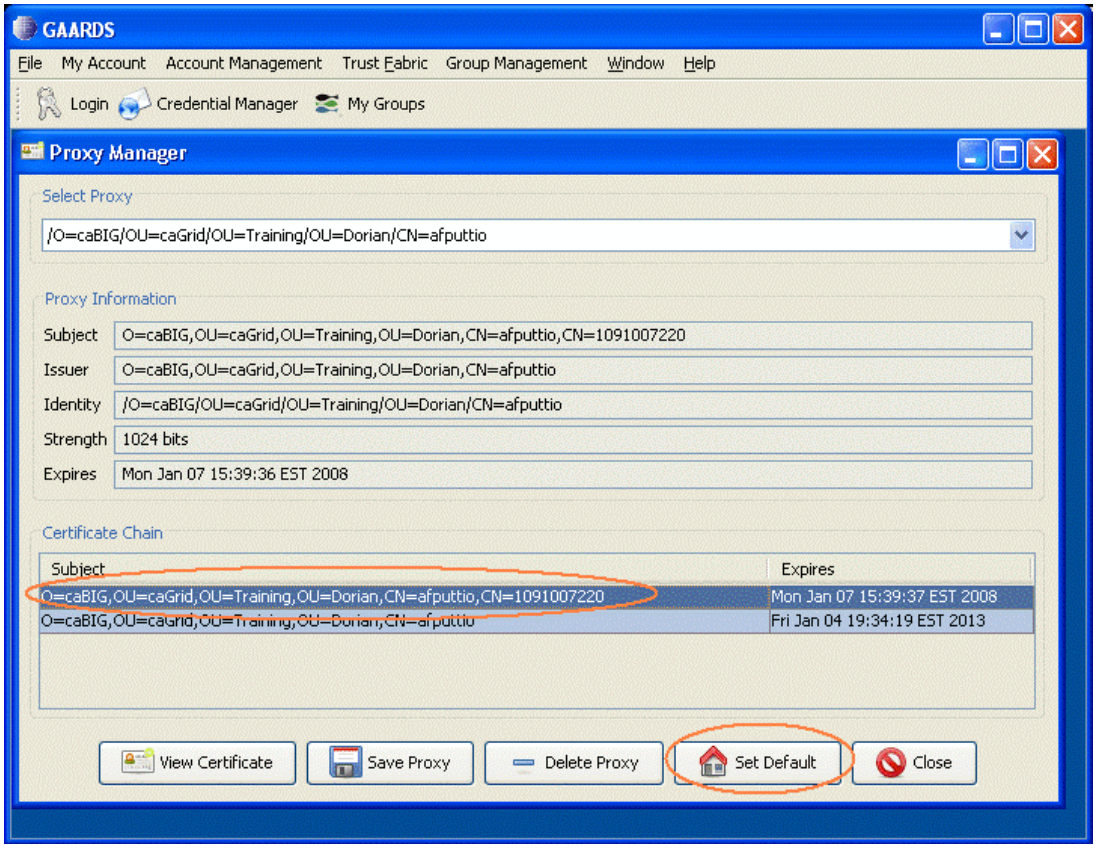

Figure 2-9: GAARDS Proxy Manager dialog box

7. In the Certificate Chain list, click on the certificate you just generated to highlight it.
8. Click **Set Default**.

To add the geWorkbench User ID to the Grid Grouper:

1. From the main menu in GAARDS, select **Group Management > Group Browser**. The Group Browser window appears.
2. Click **Add Grid Grouper** located in the bottom-right of the Group Browser window.
3. In the Add Grid Grouper dialog box, use the drop-down lists to specify the appropriate Grid Grouper URL and the geWorkbench user credentials.
4. Click **Add**.

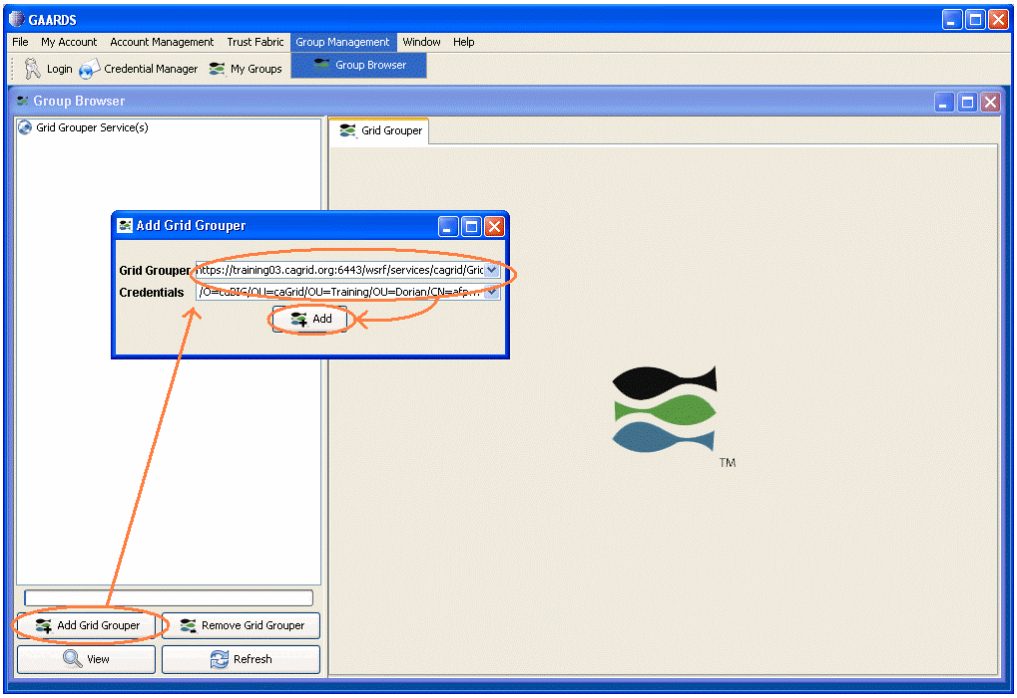

Figure 2-10: Group Browser window showing Add Grid Grouper dialog box

After the grid grouper service loads, it should appear in the Grid Grouper Service explorer pane located on the left side of the Group Browser window.

5. In the service explorer pane, find the grid group for this project, and double-click on the group name. This opens detailed information for the group in the right-side of the Group Browser window. The Details tab is active by default.

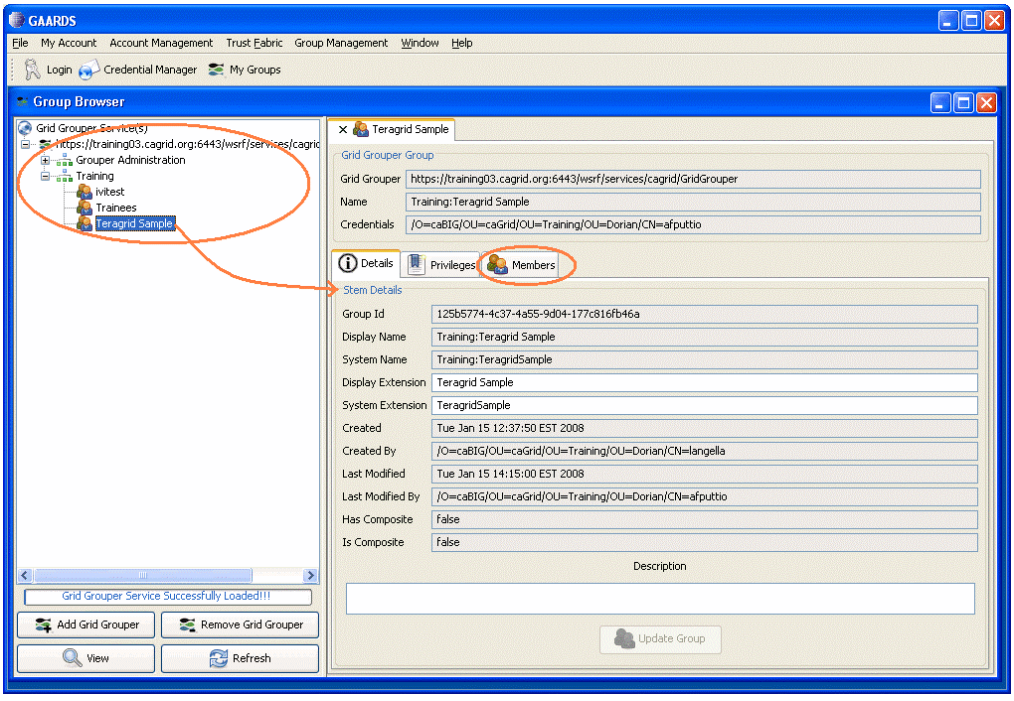

Figure 2-11: Group Browser window showing selected group details

6. Click on the **Members** tab to activate it.
7. Click **Add Member** at the bottom of the Members tab. The Add Member dialog box appears.

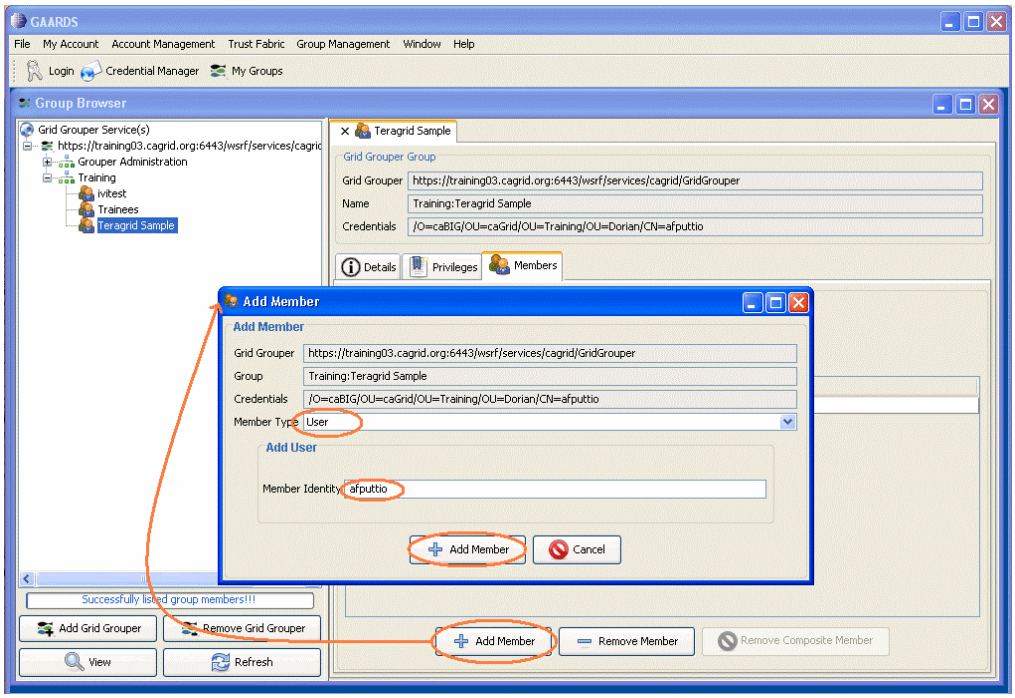

Figure 2-12: Add Member dialog box

8. From the Member Type drop-down list, select **User**.
9. In the Member Identity text box, type in the username of the member you want to add.
10. Click **Add Member**.

Once added, the user name will appear in the Members Name list on the Members tab for the selected group.

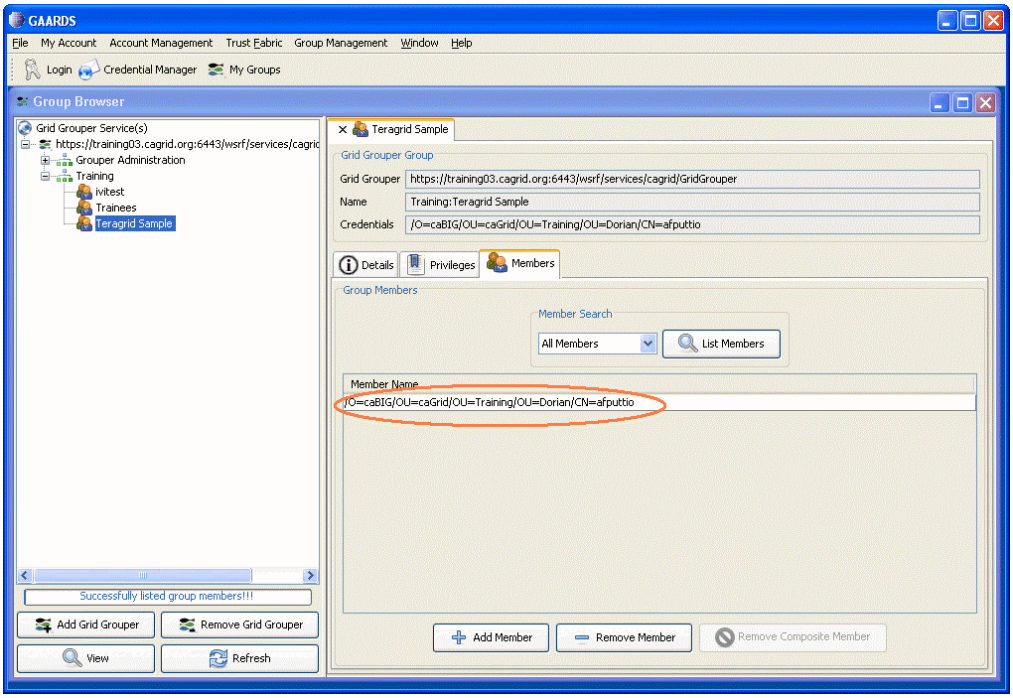

Figure 2-13: Group Browser Members tab showing added member

Step 7—Confirm caBIG community account for TeraGrid

Developers of gateway services can use the caBIG Community account to access TeraGrid resources; they do not have to obtain individual TeraGrid accounts. However if you are interested in obtaining a TeraGrid account, the following links provide further information and procedures:

- Apply for a TeraGrid account
- Set up community software area

Verify that you have access to the caBIG community account. You can do this by logging onto a login node. The example below shows the command for logging onto the San Diego cluster login node:

```
ssh -X <userNameForSanDiego>@dslogin.sdsc.edu
password:<SanDiegoPassword>
```

Once you have verified that you have access to the caBIG community account, use the sets of commands below to perform Single Sign on:

```
<userNameForSanDiego>@tg-grid1:~> myproxy-logon -T -l <userNameForUserPortal>
Enter MyProxy pass phrase:<userPortalPassword>
A credential has been received for user <userNameForUserPortal> in /tmp/x509up_u510.
Trust roots have been installed in /home/<userNameForUserPortal>/.globus/certificates/.
```

```
<userNameForUserPortal>@tg-grid1:~> grid-proxy-info
subject  : /C=US/O=National Center for Supercomputing Applications/ CN=<Name of User>
issuer   : /C=US/O=National Center for Supercomputing Applications/
OU=Certificate Authorities/CN=MyProxy
identity : /C=US/O=National Center for Supercomputing Applications/ CN=<Name of User>
type     : end entity credential
strength : 1024 bits
path     : /tmp/x509up_u510
timeleft : 11:59:37
```

```
<userNameForUserPortal>@tg-grid1:~> gsissh tg-login.ncsa.teragrid.org NCSA Teragrid Cluster
(MERCURY) --In Production with 868 nodes--
```

For additional information on available clusters and nodes, see TeraGrid Resources. (Click on the Resources tab and Systems Monitor sub-menu to get a list of host nodes.)

Step 8—Synchronize caBIG community credentials for TeraGrid with caGrid Trust Fabric

Use the username and password credentials established for your caBIG community account, and follow the basic steps outlined in Step 2—Synchronize User Credentials with caGrid Trust Fabric on page 34.

Step 9—Synchronize caBIG community credentials for TeraGrid with TeraGrid Trust Fabric

Use the username and password credentials established for your caBIG community account, and follow the basic steps outlined in Step 2—Synchronize User Credentials with caGrid Trust Fabric on page 34. However, instead of identifying a caGrid, substitute TeraGrid in the command.

Step 10—Associate the appropriate grid group with the Gateway Service

Assuming the gateway service has been created, load it into Introduce via the **Modify Service** button. Then click the Security tab to activate it and show the sub-tabs available.

The Authorization sub-tab on the Security tab allows you to specify which grid grouper membership is allowed to access this gateway service.

NOTE: If the gateway service has not been created, please skip this step and follow the instructions located at Creating the caGrid Gateway Service on page 46.

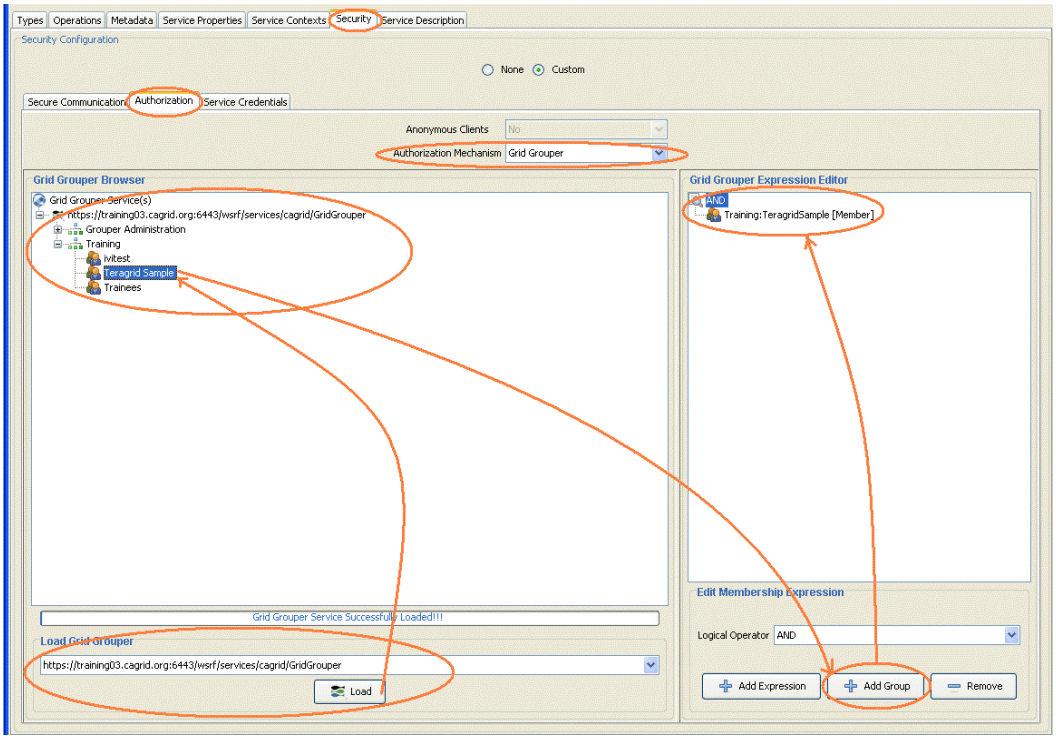

Figure 2-14: Introduce interface showing TeraGrid authorization

To authorize grid group access to the gateway service:

1. Click the Security tab to activate it, then click the Authorization sub-tab to display the Authorization options.
2. In the Authorization tab, select **Grid Grouper** from the Authorization Mechanism drop-down list.
3. Specify the grid grouper service URL from the Load Grid Grouper drop-down list and then click **Load** to bring up the list of available groups in the right side of the window.
4. Select the **TeragridSample** group from the list and then click **Add Group**.

The TeragridSample group is now added to the list of users authorized to invoke the gateway service.

For more information, see caGrid Grid Grouper.

Binary Staging on TeraGrid

Make sure you have access to the caBIG community account for TeraGrid or some other valid TeraGrid account. If not, see Step 7—Confirm caBIG community account for TeraGrid on page 42.

NOTE: For illustration purposes, the steps below use the fat jar file for the hierarchical clustering gateway. The specific file and cluster machine used are indicated in parentheses.

To stage your binary on caGrid:

1. Stage the jar files in the community software area. The University of Chicago is a good machine to use. To transfer your binary files, log in using the caBIG account, and use SCP to transfer the jar file.
2. Make sure the node you plan to stage the binaries on contains the correct version of any software your binaries may need. For example, if your binaries use JDK 1.5 and you are using the University of Chicago node for staging, be sure it also contains JDK 1.5.
3. Create a command line interface and test the file locally. This should take an input directory to read the serialized objects from and an output directory to serialize the results to. Running this would look like the following:

```
java -Xmx900M -jar HierarchicalClusteringCli_fat.jar -i my/input/dir o-
my/output/dir
```

4. Next, test your binary (`HierarchicalClusteringCli_fat.jar`) on the University of Chicago cluster using the following steps:
   a. Log onto to a login node (e.g. sdsc) as detailed in Step 7—Confirm caBIG community account for TeraGrid on page 42.
   b. SCP your binary (`HierarchicalClusteringCli_fat.jar`) to the login machine.
   c. Perform single sign on as detailed in Step 7—Confirm caBIG community account for TeraGrid on page 42.
   d. SCP your binary (`HierarchicalClusteringCli_fat.jar`) from the login machine to your chosen cluster machine (`tg-grid1.uc.teragrid.org`).
   e. Verify that the necessary software is available to you. For example, to run `HierarchicalClusteringCli_fat.jar`, type ***`java –version`*** to verify that `tg-grid1` is running java 1.5.*.)
   f. Run your binary again. For the example provided here using `HierarchicalClusteringCli_fat.jar`, re-run the java command indicated above.

The log output should be the same as the local run.

Creating the caGrid Gateway Service

This section details how to create a caGrid gateway service. For illustration purposes and to provide a working example to follow, the sections below use the code for the hierarchical clustering gateway.

Introduce and gRAVI

Use Introduce with the gRAVI plug-in to create the basis for the gateway service -- TeragridSample. The generated code should have the following structure:

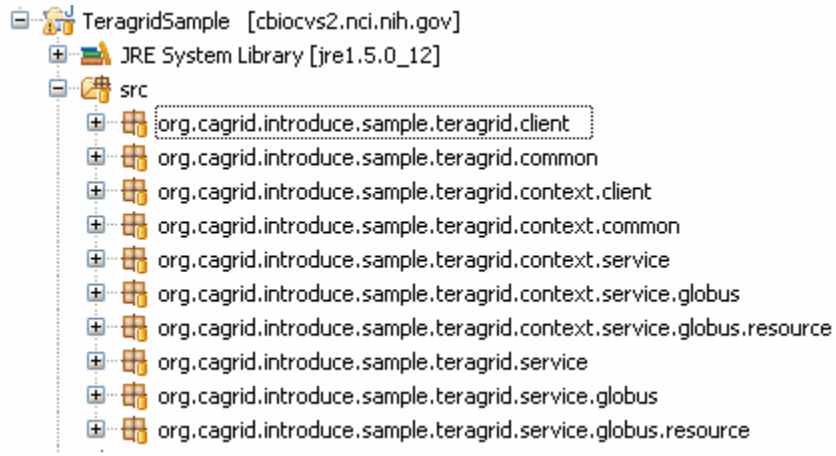

Figure 2-15: Introduce-generated code structure (example)

In Introduce, the data types imported are:

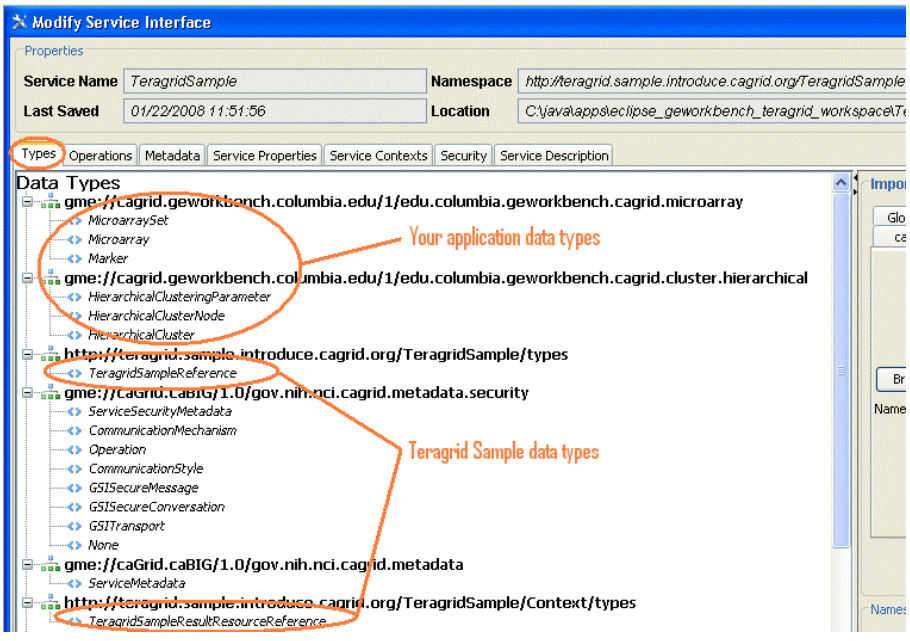

Figure 2-16: Introduce-imported data types (TeraGrid example)

The specified operations method is:

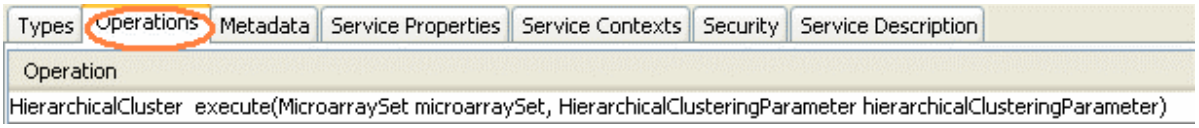

Figure 2-17: Introduce specified operations method (TeraGrid example)

In this case the one method the gateway client calls is `execute()`.

The Security tab in Introduce should be configured as shown in Figure 2-18, with the **Custom** option selected.

In the **Secure Communication** sub-tab, all three communications security options must be checked and all three Communication Method text boxes should show the value **Privacy**.

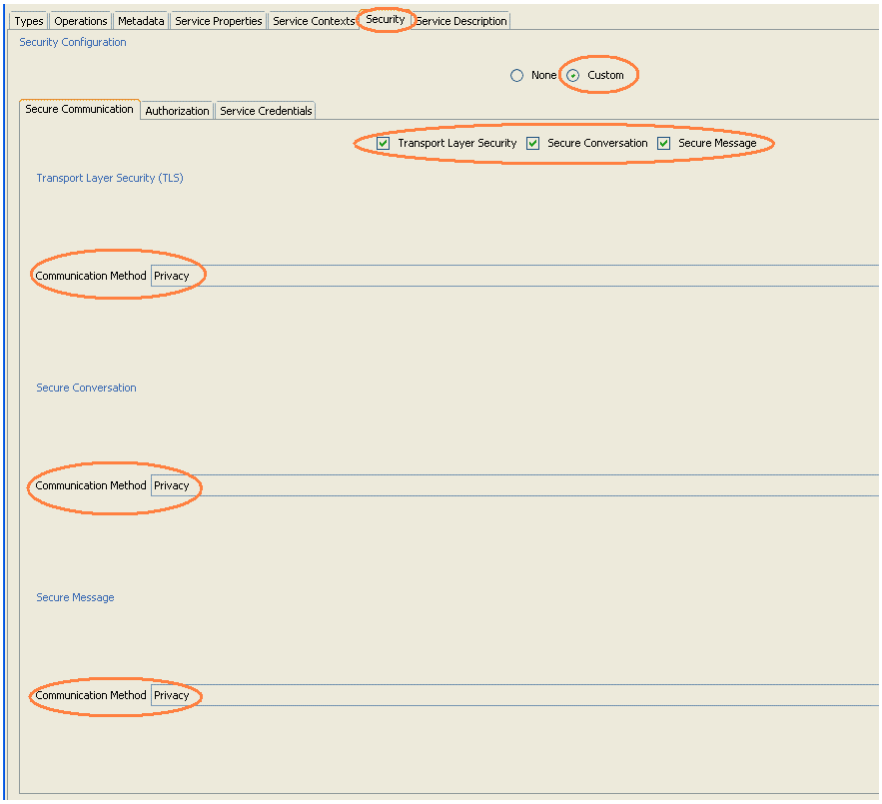

Figure 2-18: Introduce security options settings (TeraGrid example)

The Authorization sub-tab on the Security tab allows you to specify which grid grouper membership is allowed to access this gateway service.

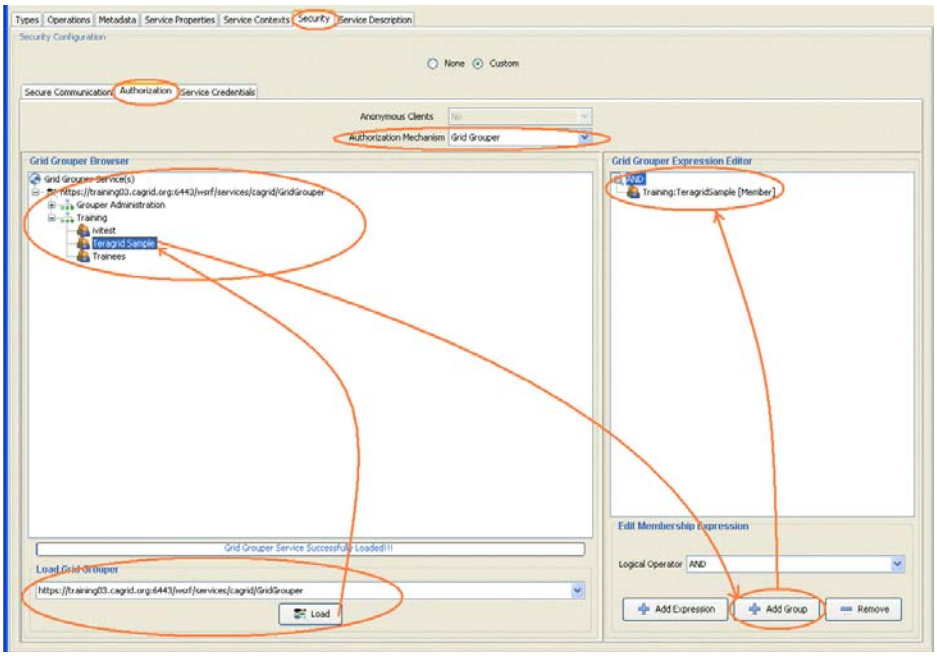

Figure 2-19: Introduce Authorization tab (TeraGrid example)

To authorize grid group access to the gateway service:

1. On the Security tab, click the Authorization sub-tab to show the Authorization options.
2. In the Authorization tab, select **Grid Grouper** from the Authorization Mechanism drop-down list.
3. Specify the grid grouper service URL from the Load Grid Grouper drop-down list and then click **Load** to bring up the list of available groups in the right side of the window.
4. Select the **TeragridSample** group from the list and then click **Add Group**.

The TeragridSample group is now added to the list of users authorized to invoke the gateway service. For more information, see caGrid Grid Grouper.

On the **Service Credentials** sub-tab of the Security tab, use the Import Credentials field to identify that the credentials are to be a Certificate/Private Key from the file system. Then enter **System** into the Run As field to run as the system.

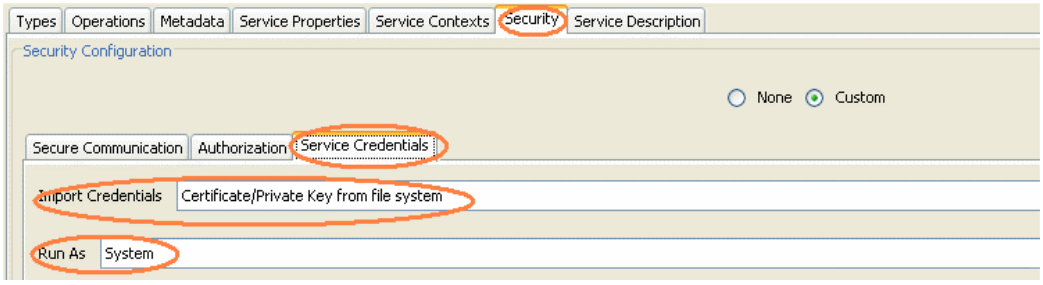

Figure 2-20: Introduce Service Credentials (TeraGrid example)

Gateway Client

- *org.cagrid.introduce.sample.teragrid.client.TeragridSampleClient.java*

The gateway client passes the data `getMicroArraySet()`, and the user selected parameters `getHierarchicalClusteringParameter()`, to the gateway service hosted at `GATEWAY_SERVICE_URL`.

Both the microarray set and parameters are caDSR-compliant data structures used by geWorkbench.

```
TeragridSampleClient client = new TeragridSampleClient(GATEWAY_SERVICE_URL);
HierarchicalCluster hc = client.execute(getMicroArraySet(),
getHierarchicalClusteringParameter());
```

Gateway Service

- *org.cagrid.introduce.sample.teragrid.context.service.TeragridSampleImpl.java*

Make sure the following jars are on the classpath:

```
GLOBUS_LOCATION/lib/axis.jar
GLOBUS_LOCATION/lib/gram-client.jar
GLOBUS_LOCATION/lib/gram-stubs.jar
GLOBUS_LOCATION/lib/gram-util.jar
GLOBUS_LOCATION/lib/gram-monitoring.jar
GLOBUS_LOCATION/lib/wsrf-core.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
GLOBUS_LOCATION/lib/jgss.jar
```

Import the following classes:

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.RandomAccessFile;
import java.net.URL;
import java.rmi.RemoteException;
import java.util.Date;
import org.apache.axis.components.uuid.UUIDGen;
import org.apache.axis.components.uuid.UUIDGenFactory;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.globus.exec.client.GramJob;
import org.globus.exec.generated.JobDescriptionType;
import org.globus.exec.generated.StateEnumeration;
import org.globus.exec.utils.ManagedJobFactoryConstants;
import org.globus.exec.utils.client.ManagedJobFactoryClientHelper;
import org.globus.wsrf.impl.security.authentication.Constants;
import org.globus.wsrf.impl.security.authorization.Authorization;
import org.globus.wsrf.impl.security.authorization.HostAuthorization;
import org.gridforum.jgss.ExtendedGSSCredential;
import org.gridforum.jgss.ExtendedGSSManager;
import org.ietf.jgss.GSSCredential;
import org.globus.exec.client.GramJobListener;
import org.globus.ftp.DataChannelAuthentication;
import org.globus.ftp.DataSink;
```

```
import org.globus.ftp.DataSource;
import org.globus.ftp.FileRandomIO;
import org.globus.ftp.GridFTPClient;
import org.globus.ftp.GridFTPSession;
import org.globus.gsi.GlobusCredential;
import org.globus.gsi.GlobusCredentialException;
import org.globus.gsi.gssapi.GlobusGSSCredentialImpl;import org.globus.ftp.DataSource;
import org.globus.ftp.FileRandomIO;
import org.globus.ftp.GridFTPClient;
import org.globus.ftp.GridFTPSession;
import org.globus.gsi.GlobusCredential;
import org.globus.gsi.GlobusCredentialException;
import org.globus.gsi.gssapi.GlobusGSSCredentialImpl;
```

Make sure the class implements GramJobListener interface:

```
public class TeragridSampleImpl extends TeragridSampleImplBase  implements GramJobListener {
```

The GramJobListener interface has one method:

```
public void stateChanged(GramJob job) {
StateEnumeration jobState = job.getState();
boolean holding = job.isHolding();
printMessage("========= State Notification =========");
printJobState(jobState, holding);
printMessage("====================================");
synchronized (this) {
if ( jobState.equals(StateEnumeration.Done)|| jobState.equals(StateEnumeration.Failed)) {
printMessage("Exit Code: " + Integer.toString(job.getExitCode()));
// also call code to start processing results, for example gridFTP them from TeraGrid back to
gateway service
this.jobCompleted = true;
}
notifyAll();
}
}
```

Authenticate with TeraGrid using caBIG community credentials. In the following command, the `PROXY_LOCATION` is the location of the TeraGrid user certificate.

```
GlobusCredential globusCred = new GlobusCredential(PROXY_LOCATION);
GlobusGSSCredentialImpl cred = new GlobusGSSCredentialImpl(globusCred,
GSSCredential.INITIATE_AND_ACCEPT);
```

Set up gridFTP info:

```
String DEST_HOST = <name of the host in TeraGrid with the gridFTP service>
int GRIDFTP_PORT = 2811;
```

GridFTP input data and parameters:

```
String FILE_TO_FTP = <full path to the input data file/parameter file>;
```

```
GridFTPClient client = new GridFTPClient(DEST_HOST, GRIDFTP_PORT);
client.authenticate(creds);
client.setProtectionBufferSize(16384);
client.setType(GridFTPSession.TYPE_IMAGE);
client.setMode(GridFTPSession.MODE_EBLOCK);
client.setDataChannelAuthentication(DataChannelAuthentication.NONE);
client.setDataChannelProtection(GridFTPSession.PROTECTION_SAFE);
client.setPassive();
client.setLocalActive();
DataSource source = new FileRandomIO(new RandomAccessFile(new File(FILE_TO_FTP), "r"));
client.extendedPut(FILE_TO_FTP, source, null);
client.close();
```

Submit the TeraGrid Job

See also TeraGrid wiki on submitting GRAM jobs via Java.

Break up the original command line to invoke the staged hierarchical clustering jar from the following:

```
java -Xmx900M -jar HierarchicalClusteringCli_fat.jar -i my/input/dir o- my/output/dir
```

to become:

```
String [] arguments = new String [7];
arguments [0] = "-Xmx900M";
arguments [1] = "-jar";
arguments [2] = "<gateway_bin>/HierarchicalClusteringCli_fat.jar";
arguments [3] = "-i";
arguments [4] = "<gateway_input_directory>";
arguments [5] = "-o";
arguments [6] = "<gateway_output_directory>";
```

Allow delegation from client > gRAVI service > GRAM:

```
boolean limitedDelegation = true;
boolean delegationEnabled = true;
```

Set service dates:

```
Date serviceDuration = <set service duration>
Date serviceTermination= <set service termination date>
```

Submit the job:

```
JobDescriptionType jobDescription = createJobDescription(PATH_TO_JAVA_BIN + "/java",
arguments);
GramJob job = new GramJob(jobDescription);
job.setTimeOut(GramJob.DEFAULT_TIMEOUT);
job.setAuthorization(HostAuthorization.getInstance());
job.setMessageProtectionType(Constants.ENCRYPTION);
job.setDelegationEnabled(delegationEnabled);
job.setDuration(serviceDuration);
job.setTerminationTime(serviceTermination);
```

```
// GRAMContact and GRAMType are both specified in rave.properties
EndpointReferenceType factoryEndpoint = getFactoryEPR(GRAMContact, GRAMType);
ExtendedGSSManager manager = (ExtendedGSSManager) ExtendedGSSManager.getInstance();
// proxyPath is specified in rave.properties
String handle = "X509_USER_PROXY=" + proxyPath.toString();
GSSCredential proxy = manager.createCredential(handle.getBytes(),
ExtendedGSSCredential.IMPEXP_MECH_SPECIFIC, GSSCredential.DEFAULT_LIFETIME, null,
GSSCredential.INITIATE_AND_ACCEPT);
job.setCredentials(proxy);
String submissionID = "uuid:" + uuidgen.nextUUID();
job.addListener(this);
job.submit(factoryEndpoint, false, limitedDelegation, submissionID);
```

For details on `rave.properties`, see the wiki on the gRAVi Plug-in.

GridFTP to Retrieve Results:

```
String FILE_TO_FTP = <full path to the result data file>;
```

```
GridFTPClient client = new GridFTPClient(DEST_HOST, GRIDFTP_PORT);
client.authenticate(cred);
client.setProtectionBufferSize(16384);
client.setType(GridFTPSession.TYPE_IMAGE);
client.setMode(GridFTPSession.MODE_EBLOCK);
client.setDataChannelAuthentication(DataChannelAuthentication.SELF);
client.setDataChannelProtection(GridFTPSession.PROTECTION_SAFE);
client.setLocalPassive();
client.setActive();
DataSink sink = new FileRandomIO(new RandomAccessFile(new File(FILE_TO_FTP), "rw"));
client.get(FILE_TO_FTP, sink, null);
client.close();;
```

Deploying the Gateway Service

Gateway services must be deployed in a secured caGrid container. You can either configure tomcat to be a secured container or install globus as the secured container.

If you intend to connect your component to this gateway service from geWorkbench, you must register the gateway service with an index service that geWorkbench can reach.

To deploy the gateway service:

1. In Introduce, open the Deploy Grid Service dialog box.
2. In the **perform.index.service.registration** text box, enter the value **true**.
3. In the **index.service.url** text box, specify the appropriate index service URL.
4. Click **Deploy**.

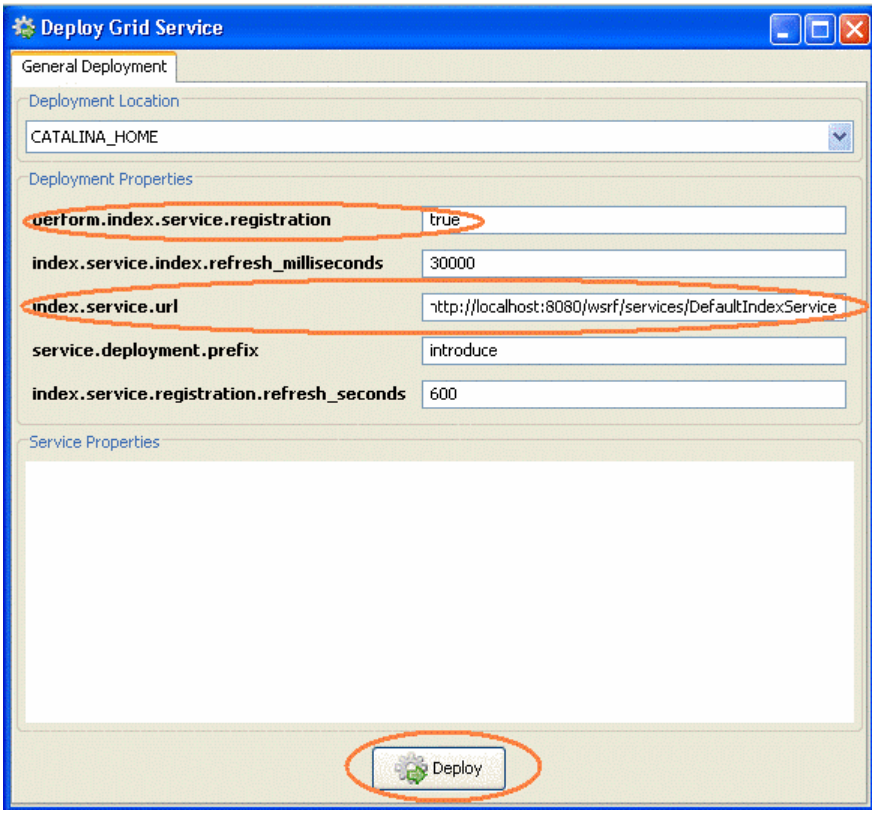

Figure 2-21: Introduce Deploy Grid Service dialog box

If you are not using geWorkbench, you can skip the index service registration and simply click **Deploy**. Then use the URL to the gateway service when you instantiate your gateway client. For more information, see Gateway Client on page 49.

caGrid-TeraGrid Security

Before continuing through this section, be sure that you have already performed the one-time security setups for both caGrid and TeraGrid. If not, please see Security for caGrid-TeraGrid Communication on page 31.

The geWorkbench Hierarchical Clustering example uses the services available on caGrid's training grid. These services are identified in caGrid Services Used For Setup on page 5.

Runtime Security Flow

The diagram below shows the flow of the security steps identified in this section. These steps are used to enable proper security of geWorkbench as it is deployed on caGrid.

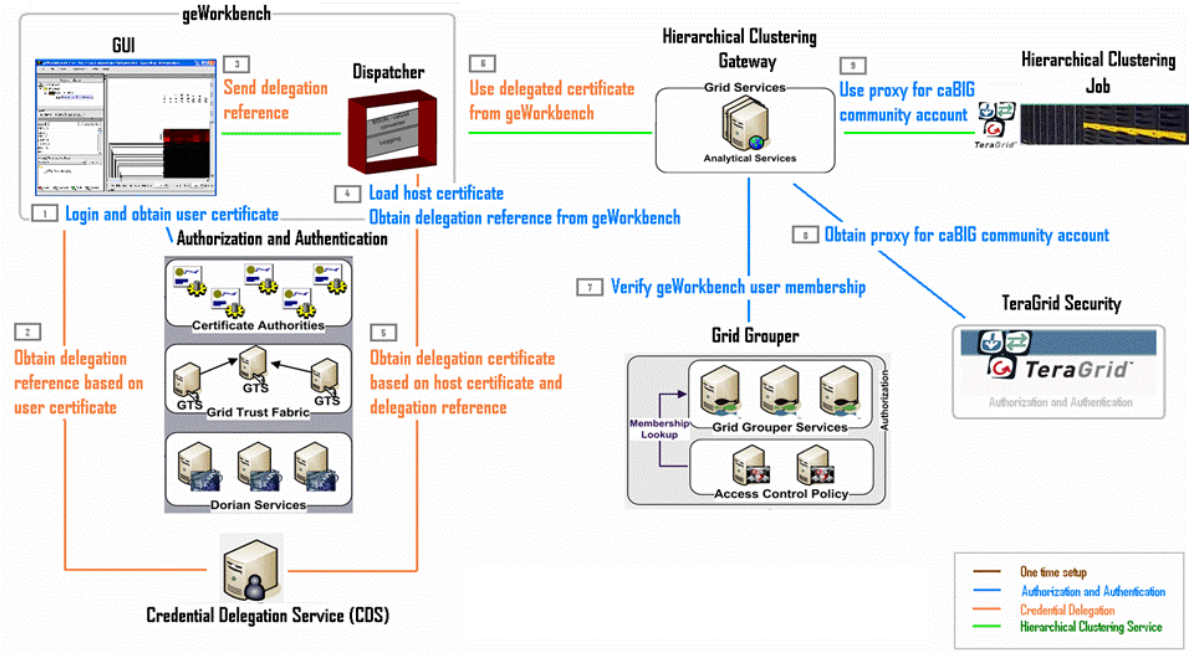

Figure 2-22: Diagram of Security setup at runtime

The sections that follow provide details regarding each of the numbered steps identified in the diagram above. Meaning that when you have completed the instructions outlined in the next sections, your geWorkbench deployment configuration should resemble that which appears in the diagram.

Delegating caGrid Proxy

With caGrid 1.2, an entity that successfully obtains a grid proxy can delegate the actual execution request to another entity. (In our case, we would be getting the grid proxy with the machine running geWorkbench and delegating the grid execution request to our dispatcher service.)

- caGrid Delegation Service Wiki
- Delegation Service Developer's Guide

While we are still working with caGrid 1.1, we can download the delegation service jars. A good place to put the `cds-client` folder is under `CAGRID_HOME/projects/`.

NOTE: Steps one through six below are completed before the Gateway Client invokes the Gateway Service. For more information, see the Gateway Client and Gateway Service sections beginning on page 49.

Step 1—Login and Obtain User Certificate

GeWorkbench stores the user's grid proxy generated from this step locally. There are two approaches for getting the certificate:

- Get the certificate manually and then programmatically delegate it;
- Programmatically both get the certificate and delegate it.

Both procedures are provided below. Choose the option that is appropriate.

To get the certificate manually then programmatically delegate it:

1. Bring up the GAARDS interface (CAGRID_HOME/ant security or with geWorkbench).
2. In GAARDS, click **Login**.
3. Enter the training Dorian URI into the **Dorian Service** text box, as shown in Figure 2-23 below.
4. Set **Lifetime** to 4 hours.

   NOTE: The 4 hours of proxy lifetime coordinates with the delegation lifetime programmatically set when the geWorkbench application is deployed. See Step 2—Obtain the Delegation Reference on page 59.
5. Set the **Delegation Path Length** to 2.
6. Enter the geWorkbench application User Id and Password.

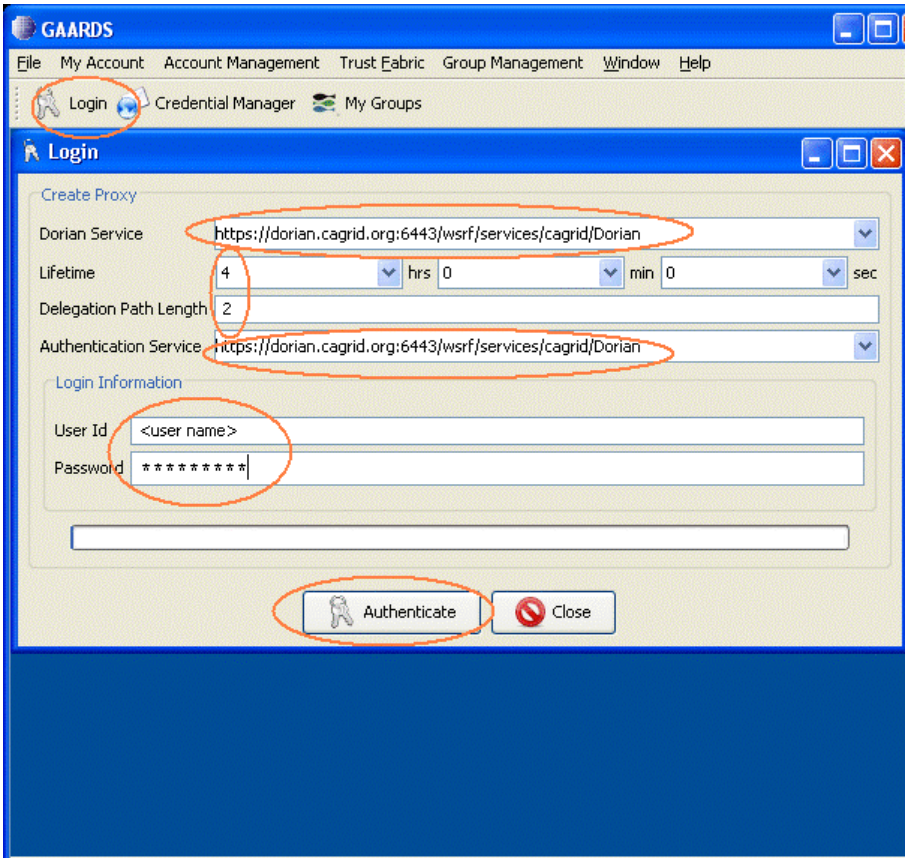

Figure 2-23: Completed Login dialog box

7. Click **Authenticate** in the completed Login dialog box. A Proxy Manager dialog box appears.

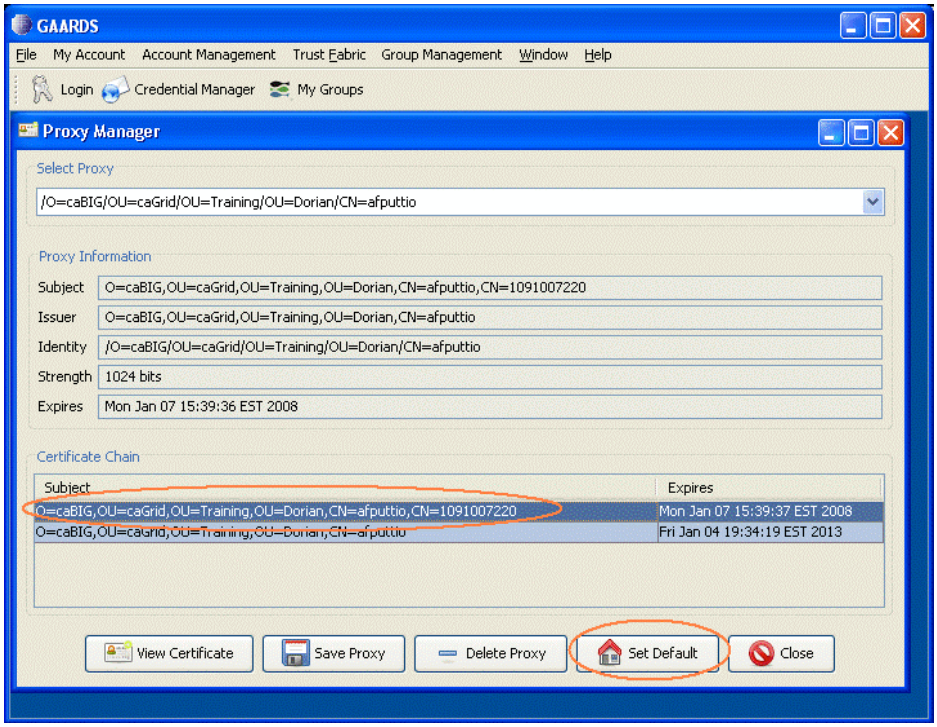

Figure 2-24: GAARDS Proxy Manager dialog box

8. In the Certificate Chain list, click on the certificate you just generated to highlight it.
9. Click **Set Default**.

To access the manually created default certificate in geWorkbench (or in any java program):

```
Put the following jars on the classpath:
CAGRID_HOME/projects/dorian/ext/lib/caGrid-1.1-core.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
```

```
import org.globus.gsi.GlobusCredential;
import org.cagrid.gaards.cds.common.core.ProxyUtil;
```

```
GlobusCredential proxy = ProxyUtil.getDefaultProxy();
```

If you print out your certificate information, it should appear as follows:

```
2007-12-20 10:02:40,685 INFO [edu.columbia.geworkbench.cagrid.security.SecurityTest] -
Testing authentication against:   https://dorian.cagrid.org:6443/wsrf/services/cagrid/Dorian
2007-12-20 10:02:41,622 WARN [org.apache.axis.utils.JavaUtils] - Unable to find required
classes (javax.activation.DataHandler and javax.mail.internet.MimeMultipart).  Attachment
support is disabled.
2007-12-20 10:02:46,169 INFO [edu.columbia.geworkbench.cagrid.security.SecurityTest] - id:
_7fca807b473d9f235c5575fe07e37fe6
2007-12-20 10:02:46,169 INFO [edu.columbia.geworkbench.cagrid.security.SecurityTest] -
issuer: O=caBIG,OU=caGrid,OU=Training,CN=Dorian IdP Authentication Asserter
2007-12-20 10:02:46,185 INFO [edu.columbia.geworkbench.cagrid.security.SecurityTest] - cert
type: sun.security.x509.X509CertImpl
2007-12-20 10:02:46,201 INFO [edu.columbia.geworkbench.cagrid.security.SecurityTest] - cert:
[
[
 Version: V3
 Subject: CN=Dorian IdP Authentication Asserter, OU=Training, OU=caGrid, O=caBIG
 Signature Algorithm: SHA1withRSA, OID = 1.2.840.113549.1.1.5
...
```

To both get the certificate and programmatically delegate it:

```
Put the following jars on the classpath:
CAGRID_HOME/projects/dorian/ext/lib/caGrid-1.1-authentication-service-stubs.jar
CAGRID_HOME/projects/dorian/ext/lib/caGrid-1.1-authentication-service-client.jar
CAGRID_HOME/projects/dorian/build/lib/caGrid-1.1-dorian-client.jar
CAGRID_HOME/projects/dorian/ext/lib/caGrid-1.1-opensaml-1.1.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
```

```
import gov.nih.nci.cagrid.authentication.bean.BasicAuthenticationCredential;
import gov.nih.nci.cagrid.authentication.bean.Credential;
import gov.nih.nci.cagrid.authentication.client.AuthenticationClient;
import gov.nih.nci.cagrid.dorian.client.IFSUserClient;
import gov.nih.nci.cagrid.opensaml.SAMLAssertion;
import org.globus.gsi.GlobusCredential;
```

```
BasicAuthenticationCredential userPass = new
BasicAuthenticationCredential(password, username);
Credential credential = new Credential();
credential.setBasicAuthenticationCredential(userPass);
AuthenticationClient authenticationClient = new
AuthenticationClient(dorianUri, credential);
SAMLAssertion samlAssertion = authenticationClient.authenticate();
assertNotNull(samlAssertion);
gov.nih.nci.cagrid.dorian.ifs.bean.ProxyLifetime lifetime = new
gov.nih.nci.cagrid.dorian.ifs.bean.ProxyLifetime();
lifetime.setHours(4);
lifetime.setMinutes(0);
lifetime.setSeconds(0);
IFSUserClient dorian = new IFSUserClient(dorianUri);
GlobusCredential proxy = dorian.createProxy(samlAssertion, lifetime, 2);
```

NOTE: There are various ProxyLifetime classes in different caGrid jar files. For this reason, it is best to use a fully qualified name for this class.

Step 2—Obtain the Delegation Reference

GeWorkbench uses the default proxy to contact the delegation service and delegate the credentials to the dispatcher service (returning a *delegationreference*).

This step identifies the procedures used to obtain the delegation reference and then pass it on to another service.

To obtain the delegation reference:

NOTE: The variable `proxy` used in the following commands is defined in Step 1—Login and Obtain User Certificate above.

```
Put the following jars on the classpath:
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-client.jar
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-common.jar
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-stubs.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
```

```
import org.cagrid.gaards.cds.client.ClientConstants;
import org.cagrid.gaards.cds.client.DelegationUserClient;
import org.cagrid.gaards.cds.common.IdentityDelegationPolicy;
import org.cagrid.gaards.cds.common.Utils;
import org.cagrid.gaards.cds.delegated.stubs.types.DelegatedCredentialReference;
import org.globus.gsi.GlobusCredential;
```

```
// Specify which host machine to delegate the user credentials.
// This is usually in the form of...
String party = "/O=caBIG/OU=caGrid/OU=Training/OU=Services/CN=host/<host name>";
// Specifies the path length of the credential being delegate the minumum is 1.
int delegationPathLength = 1;
// Specifies the how long credentials issued to allowed parties will be valid for.
org.cagrid.gaards.cds.common.ProxyLifetime issuedCredentialLifetime = new
org.cagrid.gaards.cds.common.ProxyLifetime();
issuedCredentialLifetime.setHours(4);
issuedCredentialLifetime.setMinutes(0);
issuedCredentialLifetime.setSeconds(0);
// Specifies how long the delegation service can delegated this credential to other parties.
org.cagrid.gaards.cds.common.ProxyLifetime delegationLifetime = new
org.cagrid.gaards.cds.common.ProxyLifetime();
delegationLifetime.setHours(1);
delegationLifetime.setMinutes(0);
delegationLifetime.setSeconds(0);
// Specifies the path length of the credentials issued to allowed parties. A path length of 0
means that the requesting party cannot further delegate the credential.
int issuedCredentialPathLength = 0;
// Specifies the key length of the delegated credential
int keySize = ClientConstants.DEFAULT_KEY_SIZE;
// The policy stating which parties will be allowed to obtain a delegated credential. The CDS
will only issue credentials to parties listed in this policy.
List<String> parties = new ArrayList<String>();
parties.add(party);
IdentityDelegationPolicy policy = Utils.createIdentityDelegationPolicy(parties);
```

```
// Create an instance of the delegation client, specifies the CDS Service URL and the
credential to be delegated.
DelegationUserClient client1 = new DelegationUserClient(cdsURL, proxy);
// Delegates the credential and returns a reference which can later be used by allowed
parties to obtain a credential.
DelegatedCredentialReference delegatedCredentialReference =
client1.delegateCredential(delegationLifetime, delegationPathLength,
policy,issuedCredentialLifetime, issuedCredentialPathLength, keySize);
```

NOTE: There are various ProxyLifetime classes in different caGrid jar files. For this reason, it is best to use a fully qualified name for this class.

To delegate to another service:

```
Put the following jars on the classpath:
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-client.jar
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-common.jar
CAGRID_HOME/projects/cds-client/ext/lib/caGrid-1.2M1-cds-stubs.jar
GLOBUS_LOCATION/lib/cog-jglobus.jar
```

```
import org.cagrid.gaards.cds.client.ClientConstants;
import org.cagrid.gaards.cds.client.DelegationUserClient;
import org.cagrid.gaards.cds.common.IdentityDelegationPolicy;
import org.cagrid.gaards.cds.common.Utils;
import org.cagrid.gaards.cds.delegated.stubs.types.DelegatedCredentialReference;
import org.globus.gsi.GlobusCredential;
```

As an example, the geWorkbench GUI delegates the delegatedCredentialReference to the geWorkbench dispatcher service.

Step 3— Send Delegation Reference

This step passes along the **DelegatedCredentialReference** *delegatedCredentialReference*, obtained through Step 2 above, to the process that is running the Gateway Client.

In the case of geWorkbench, the GUI must pass this reference to the dispatcher service.

```
dispatcherClient.submit(..., delegatedCredentialReference);
```

Step 4—Load Host Certificate and Obtain Delegation Reference

The machine hosting the service that is running the gateway client (in this case the geWorkbench dispatcher service), on behalf of the user (in this case geWorkbench GUI) should load its own certificates first before asking for the delegated certificate.

```
//Service certificate file
File certFile = new File(pathToHostCertificate);
```

```
File keyFile = new File(pathToHostKey);
//Load Dispatcher service's certificate
X509Certificate cert = CertUtil.loadCertificate(certFile);
//Load Dispatcher service's private key
PrivateKey key = KeyUtil.loadPrivateKey(keyFile, null);
X509Certificate[] chain = new X509Certificate[]{cert};
//Create Credential for dispatcher service
GlobusCredential credential = new GlobusCredential(key,chain);
```

Step 5—Obtain Delegation Certificate

The delegation certificate is obtained from the Credential Delegation Service (CDS) based on the host certificate and the delegation reference.

The service host would have its own credentials and would have obtained the delegated credential reference from the user. In our example the geWorkbench dispatcher service would have its own proxy and would have received the delegatedCredentialReference passed from the dispatcher client (the geWorkbench GUI).

```
//Create an instance of the delegate credential client, specifying the
//DelegatedCredentialReference and the credential of the delegatee.  The
//DelegatedCredentialReference specifies which credential to obtain.  The
//delegatee's credential is required to authenticate with the CDS such
//that the CDS may determing if the the delegatee has been granted access
//to the credential in which they wish to obtain.
DelegatedCredentialUserClient client = new
DelegatedCredentialUserClient(delegatedCredentialReference, credential);
//The get credential method obtains a signed delegated credential from the
CDS.
GlobusCredential delegatedCredential = client.getDelegatedCredential();
```

The result should appear as indicated in Figure 2-25 below:

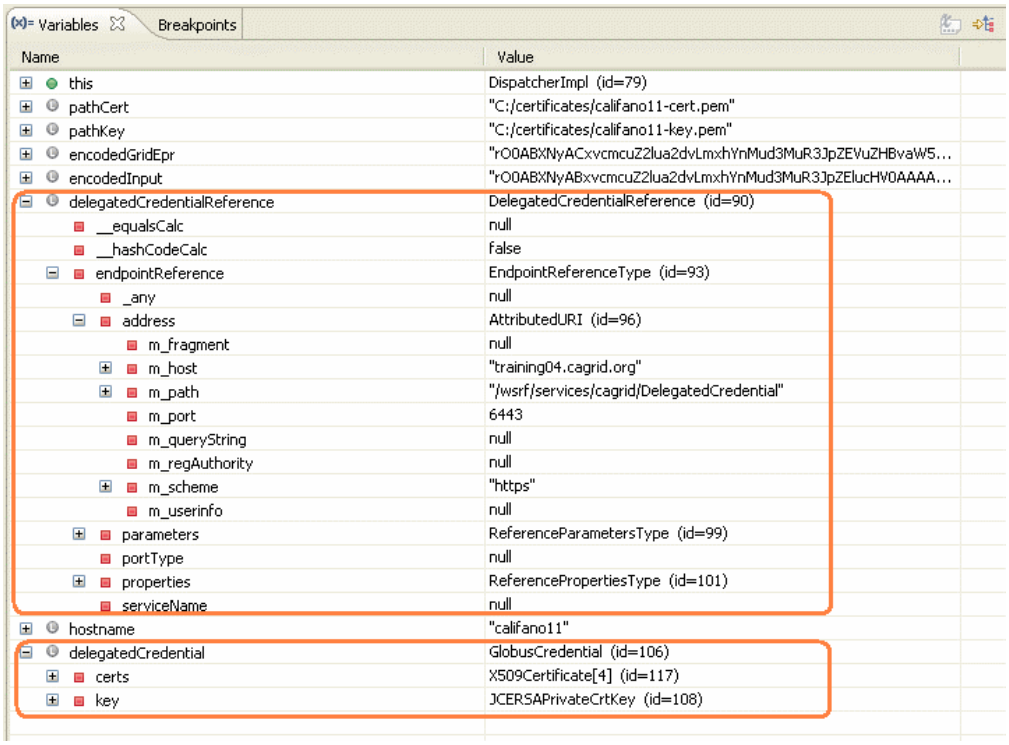

Figure 2-25: Delegated Credential Reference - desired results

Step 6—Use Delegated Certificate From geWorkbench

Set the delegated credential as the default proxy for the machine running the Gateway Client. When the Gateway Service is in turn invoked, the delegated credential is automatically checked against the grid grouper service.

```
ProxyUtil.saveProxyAsDefault(delegatedCredential);
```

TeraGrid Security

Step 7—Verify geWorkbench User Membership

As long as the Introduce and gRAVI step (detailed beginning on page 46) was performed appropriately, this step is automatic when the gateway client invokes the gateway service. This refers specifically to the configuration performed in Introduce through the **Modify Service > Security > Authorization tab**.

Step 8—Obtain Proxy for caBIG Community Account

As long as the Gateway Service step (detailed on page 49) was performed appropriately and includes code to authenticate either the caBIG community account or some other valid TeraGrid account, this step is automatic when the gateway service is invoked.

Step 9—Use Proxy for caBIG Community Account

As long as the Gateway Service step (detailed on page 49) was performed appropriately and includes code to obtain the proxy after authentication with TeraGrid, this step is automatic when the gateway service is invoked.

Running the geWorkbench-caGrid-TeraGrid Demo

The figure below shows the steps involved and the flow that occurs when running the geWorkbench caGrid/TeraGrid demonstration.

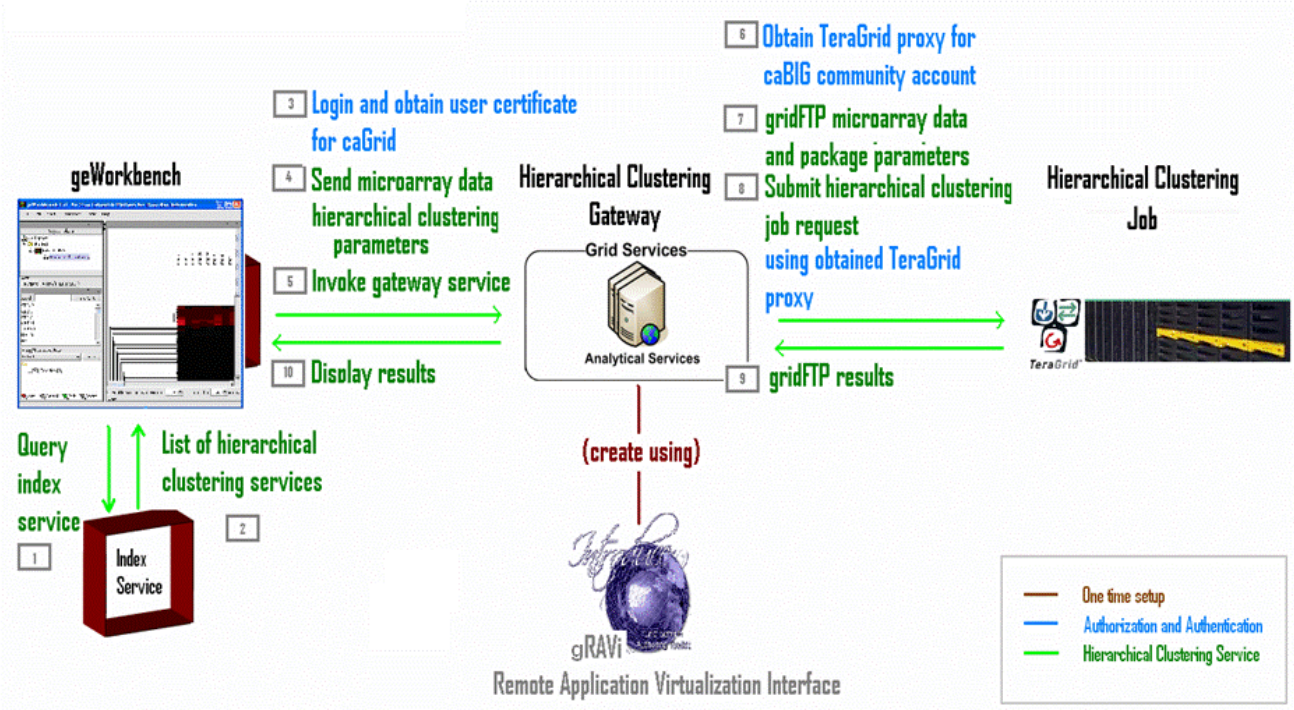

Figure 2-26: Diagram showing steps and flow of geWorkbench caGrid/TeraGrid Demo

The sections that follow provide the procedures necessary to perform the geWorkbench caGrid/TeraGrid demonstration.

Setting Up

If caGrid is not installed, please do so using the instructions located in the following link: http://wiki.c2b2.columbia.edu/informatics/index.php/CaGrid_1.1-final.

Download geWorkbench demo source, and then follow the instructions located on the geWorkbench getting started page.

Make sure to follow the setup instructions in the Security for caGrid-TeraGrid Communication section beginning on page 31 of this document.

In your environment variables, specify the home directories of Ant and caGrid, as shown in Figure 2-27 below.

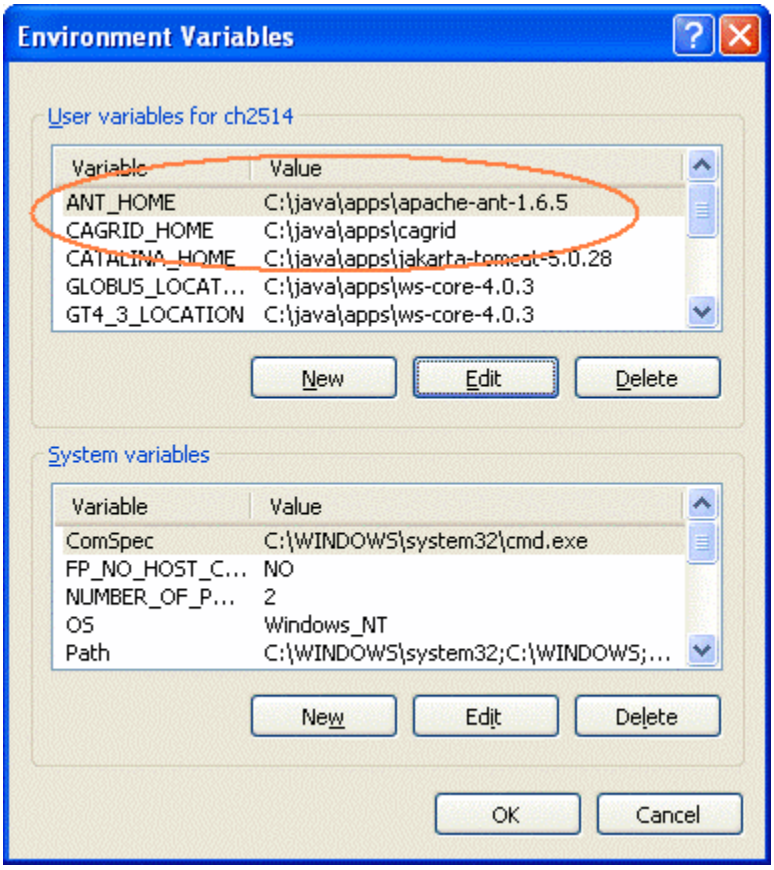

Figure 2-27: Environment variables showing home directories for Ant and caGrid

NOTE: After setting the environmental variables, you must restart the process used to run geWorkbench (i.e., the console, eclipse, etc.).

Bringing Up Hierarchical Clustering in geWorkbench

The following steps are performed in the geWorkbench application.

1. Open the geWorkbench GUI.
2. On the left side of the window, right-click on Workspace then select **New Project** from the shortcut menu.

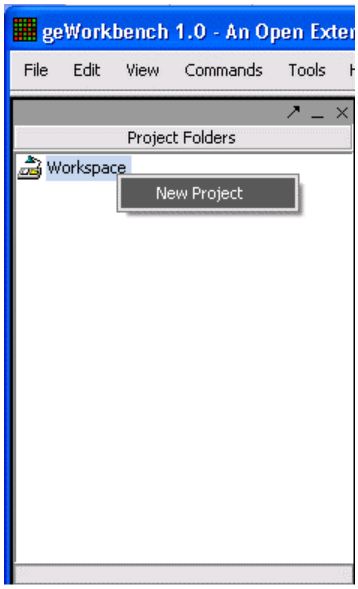

Figure 2-28: Create new project in geWorkbench GUI

3. When the Project folder appears, right-click on it and select **Open File(s)** from the shortcut menu.

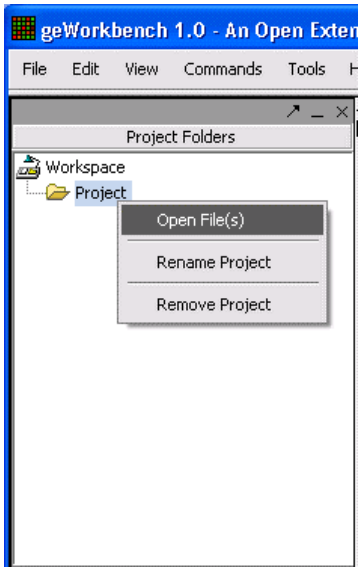

Figure 2-29: Open file in new project

4. From the Open File dialog box, find and select the appropriate `.exp` file and then click **Open**.

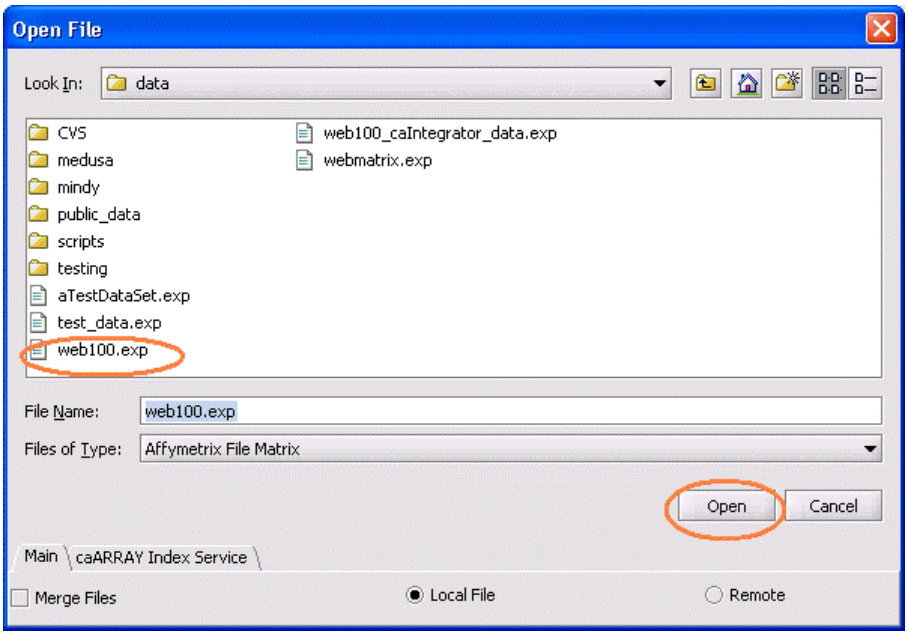

Figure 2-30: Find and select `.exp` file to open

5. An Annotations Information dialog box appears, showing licensing and registration information. Review the information, then enable the **Don't show this again** check box and click **Continue**.

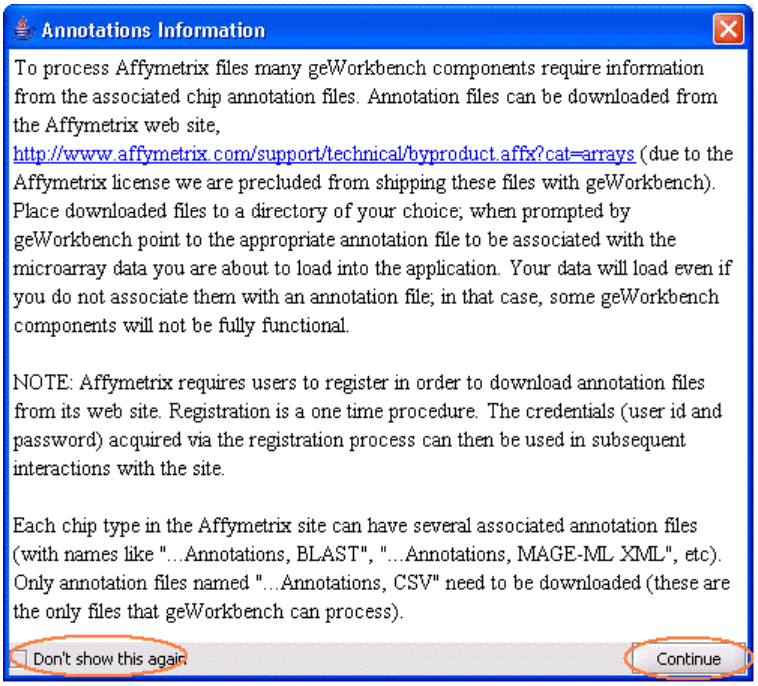

Figure 2-31:Annotations Information dialog box

6. In the annotation file selection dialog box, click **Cancel**.

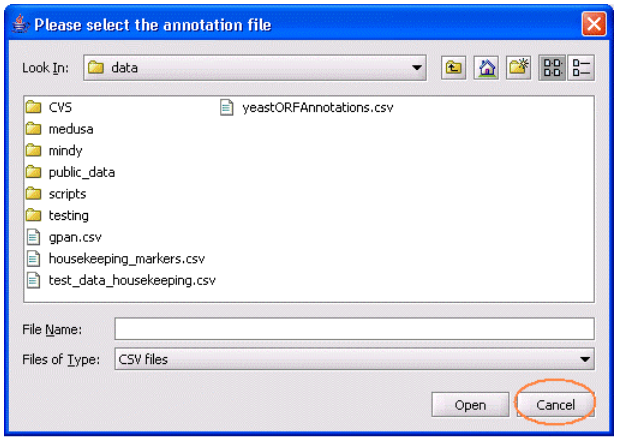

Figure 2-32: Select annotation file dialog box

The geWorkbench GUI returns. After the selected data file loads, if necessary click on it in the left tree-view to activate it, then click on **Fast Hierarchical Clustering Analysis** on the right side of the window. This displays the analysis component of the file.

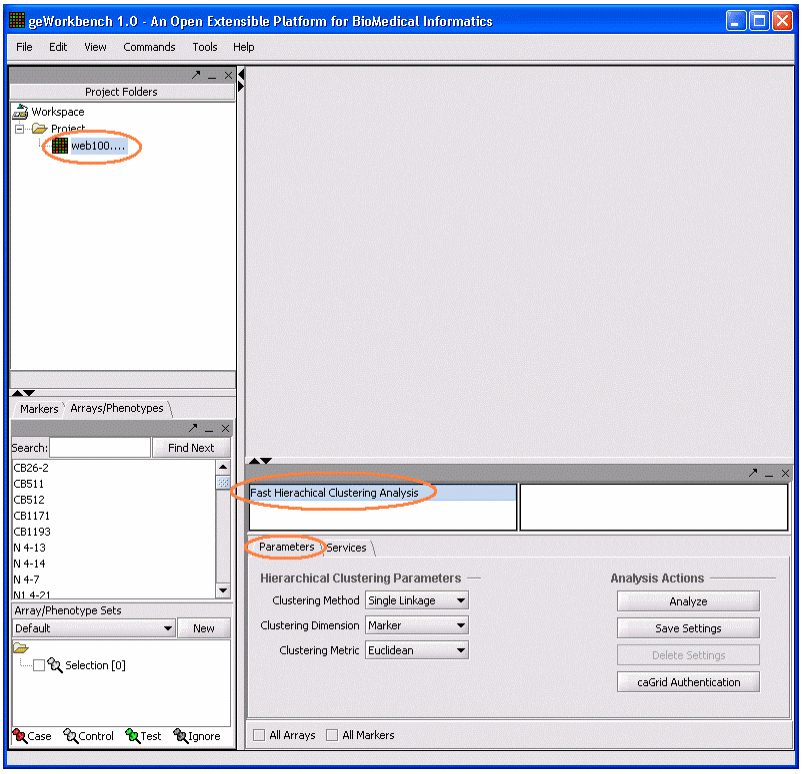

Figure 2-33: geWorkbench GUI with data file and hierarchical clustering shown

Accessing caGrid Authentication

Accessing the caGrid Authentication is done through GAARDS. The GAARDS interface can be accessed through geWorkbench by clicking the caGrid Authentication button.

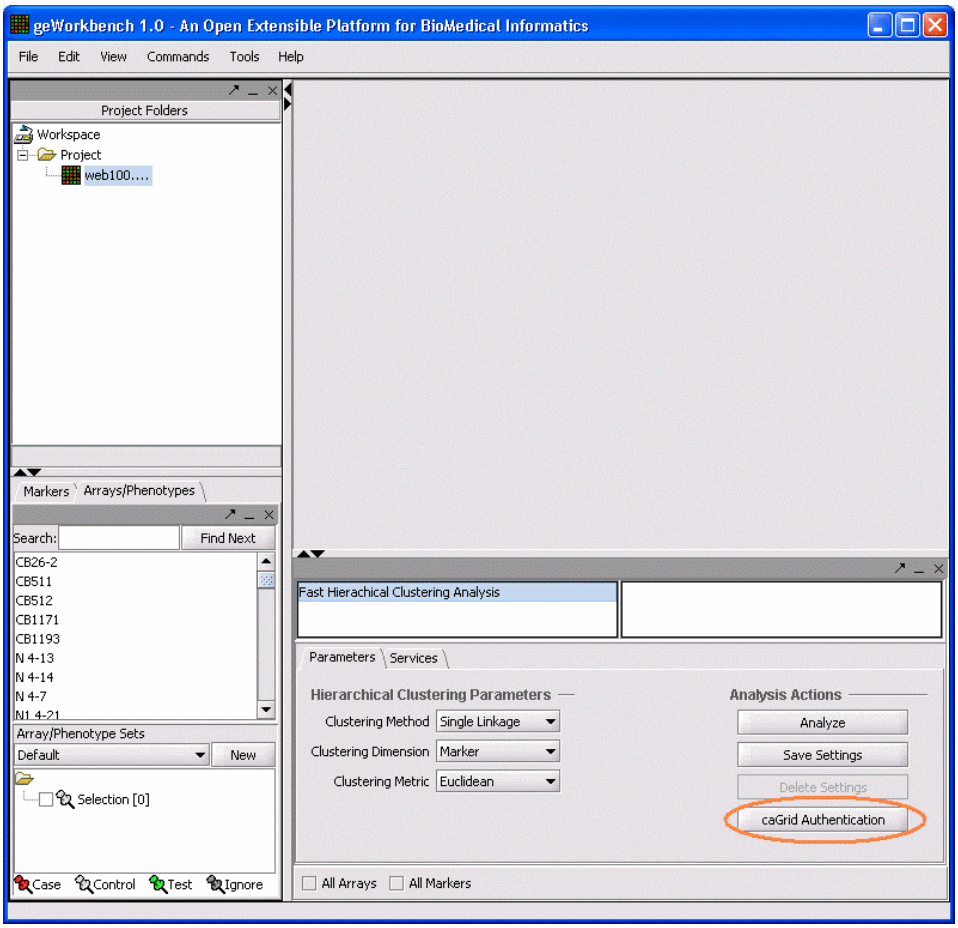

Figure 2-34: caGrid Authentication button in geWorkbench GUI

To access caGrid Authentication:

1. Open GAARDS by clicking the caGrid Authentication button in geWorkbench.

   NOTE: It may take a moment for the GAARDS UI to appear.

2. Once GAARDS appears, invoke your desired Dorian function. Assuming the user has a valid caGrid account, click **Login** to authenticate with that caGrid username and password.

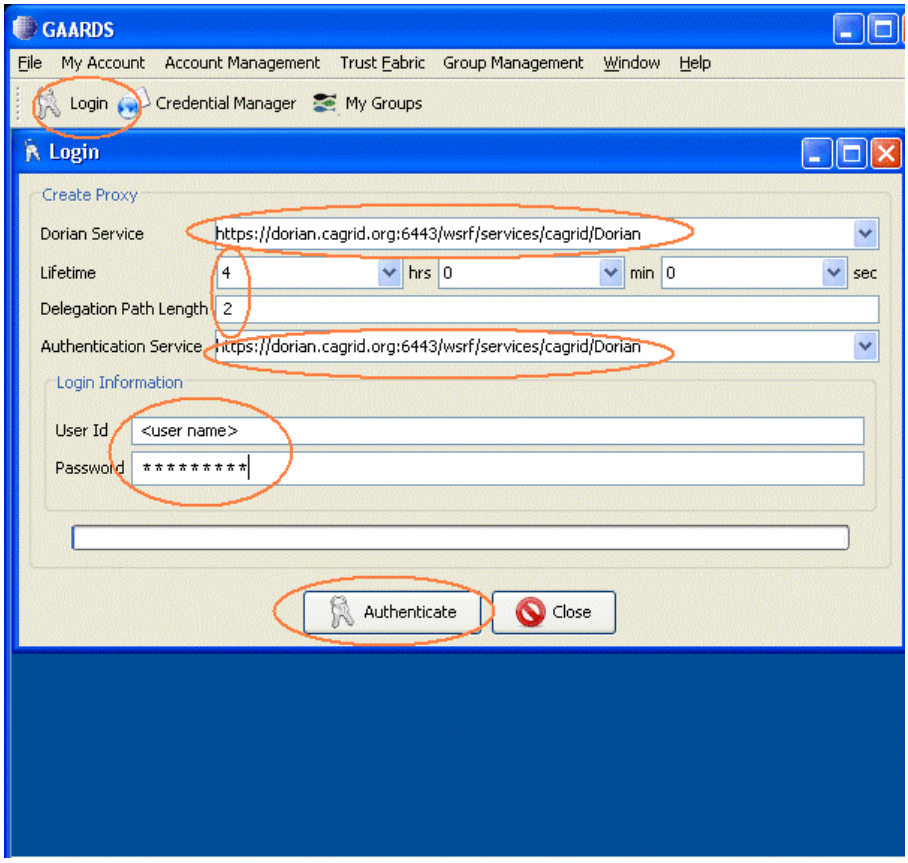

Figure 2-35:GAARDS Login dialog box with training Dorian URL

3. Using the training Dorian URL as shown in Figure 2-35 above, complete the Login dialog box and click **Authenticate**.
4. In the Proxy Manager dialog box that appears, in the Certificate Chain list, highlight the certificate you just generated and click **Set Default**.

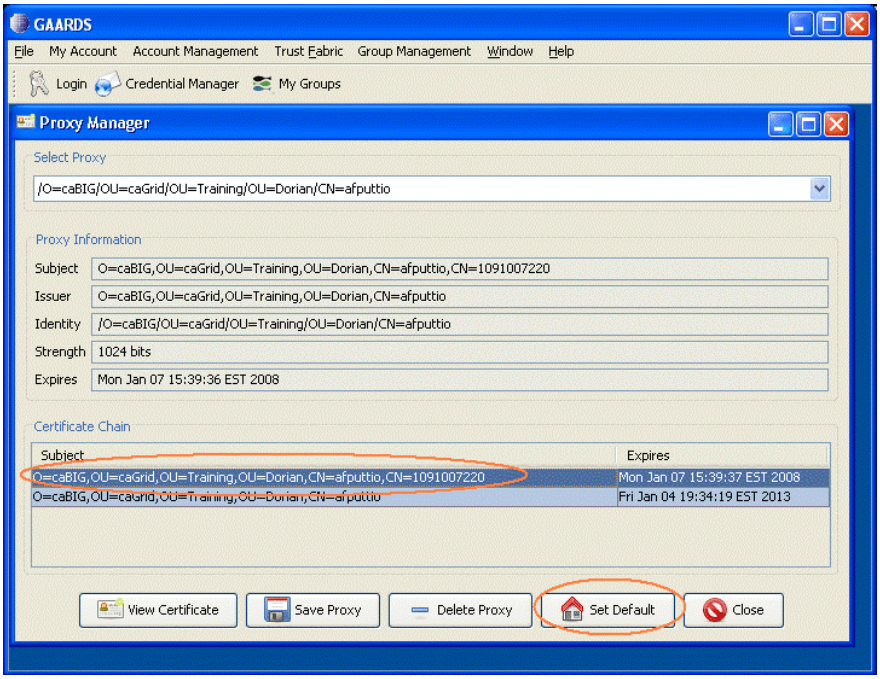

Figure 2-36:GAARDS Proxy Manager dialog box

Running the TeraGrid-Awareness Analysis

After authenticating and setting the default certificate, you can analyze whether your geWorkbench configuration is TeraGrid aware.

To run TeraGrid-Aware Analysis:

1. In geWorkbench, highlight the project file in the Project Folders list, and then click the **Services** tab in the bottom right pane to activate it.
2. On the Services tab, click the **Grid** radio button.
3. Click the **Change Index Service** link located to the right of the Grid radio button.
4. In the dialog box that appears, enter the **host** and **port** of the index service and click **Ok**.

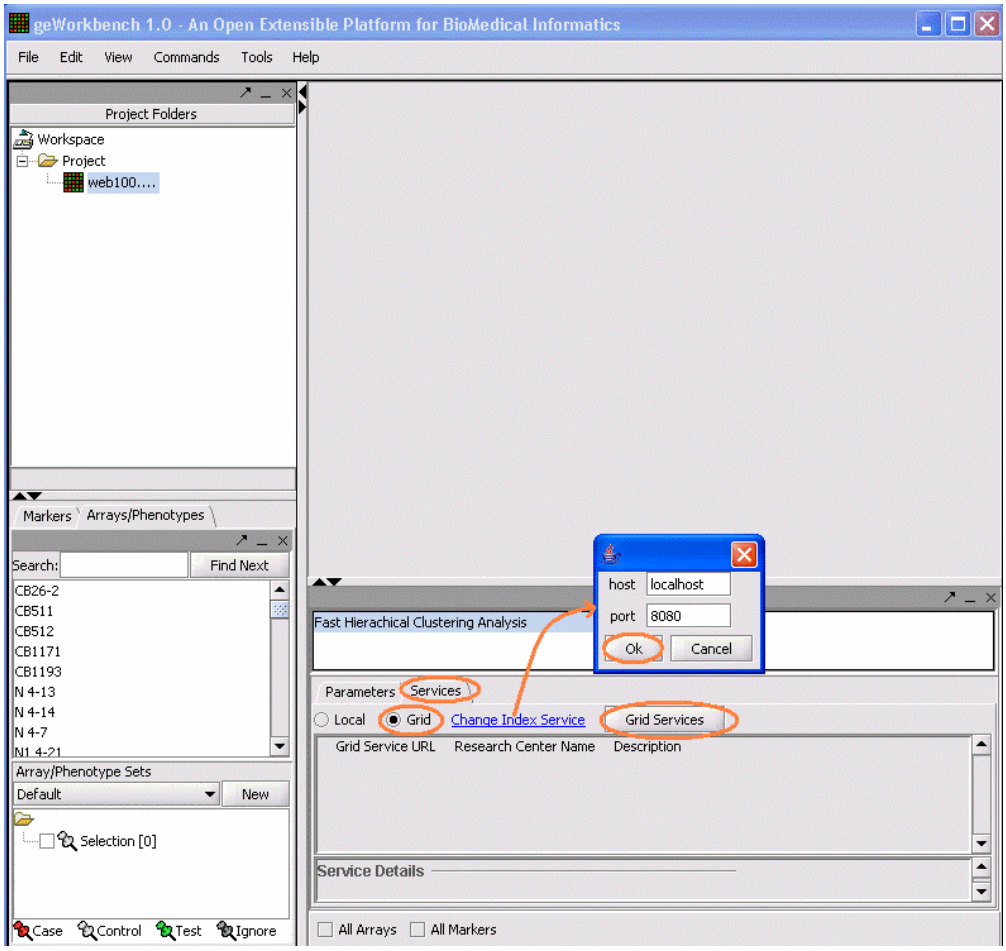

Figure 2-37:Services Tab showing configuration selections

5. Click **Grid Services** to find the registered hierarchical clustering services.
6. When the service you want appears in the list, click the radio button to the left of the appropriate grid service to designate it for use.

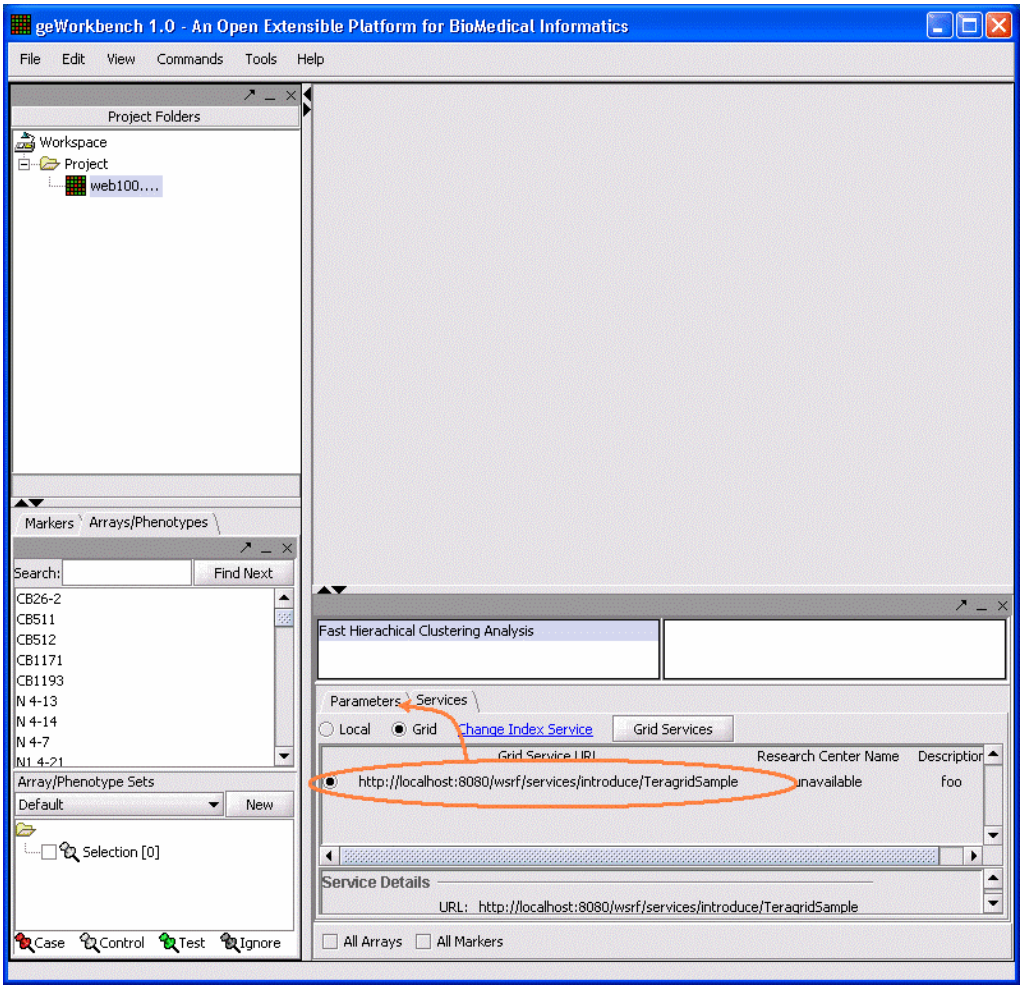

Figure 2-38:Services Tab with grid service designated

7. After selecting the grid service to use, click the **Parameters** tab to activate it.
8. In the Project Folders on the left side of the window, click the Project folder to highlight it.

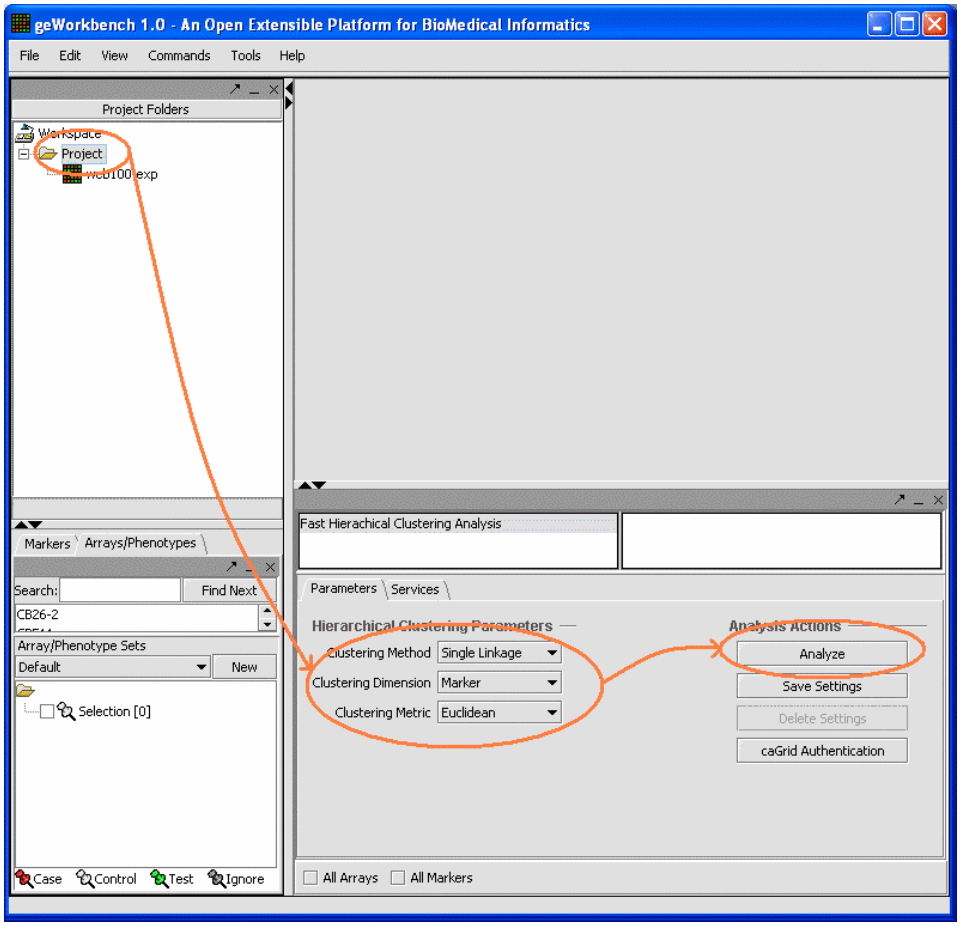

Figure 2-39:Parameters Tab after grid service designated, ready for analysis

9. In the Parameters tab, use the drop-down lists to specify the **Hierarchical Clustering Parameters** for the analysis.
10. Click **Analyze**.

Appendix A References

Scientific Publications

[1] B. Allcock, J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, and T. S., "Data Management and Transfer in High Performance Computational Grid Environments," *Parallel Computing Journal*, vol. 28, pp. 749-771, 2002.

[2] W. E. Allcock, I. Foster, and R. Madduri, "Reliable Data Transport: A Critical Service for the Grid.," in *Proceedings of Building Service Based Grids Workshop, Global Grid Forum 11*. Honolulu, Hawaii, USA, 2004.

[3] G. Allen, T. Dramlitsch, I. Foster, T. Goodale, N. Karonis, M. Ripeanu, E. Seidel, and B. Toonen, "Cactus-G Toolkit: Supporting Efficient Execution in Heterogeneous Distributed Computing Environments," in *Proceedings of the 4th Globus Retreat*. Pittsburg, PA, 2000.

[4] H. Andrade, T. Kurc, A. Sussman, and J. Saltz, "Active Proxy-G: Optimizing the Query Execution Process in the Grid," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2002)*. Baltimore, MD: ACM Press/IEEE Computer Society Press, 2002.

[5] J. Annis, Y. Zhao, J. Voeckler, M. Wilde, S. Kent, and I. Foster, "Applying Chimera Virtual Data Concepts to Cluster Finding in the Sloan Sky Survey," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2002)*. Baltimore, MD: ACM Press/IEEE Computer Society Press, 2002.

[6] M. P. Atkinson and et.al., "Grid Database Access and Integration: Requirements and Functionalities," Technical Document, Global Grid Forum. http://www.cs.man.ac.uk/grid-db/documents.html, 2002.

[7] F. Berman, H. Casanova, J. Dongarra, I. Foster, C. Kesselman, J. Saltz, and R. Wolski, "Retooling Middleware for Grid Computing," *NPACI & SDSC enVision*, vol. 18, 2002.

[8] M. Beynon, T. Kurc, A. Sussman, and J. Saltz, "Design of a Framework for Data-Intensive Wide-Area Applications," in *Proceedings of the 2000 Heterogeneous Computing Workshop (HCW2000)*. Cancun, Mexico, 2000.

[9] H. Casanova, O. Graziano, F. Berman, and R. Wolski, "The AppLeS Parameter Sweep Template: User-Level Middleware for the Grid," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2000)*: ACM Press/IEEE Computer Society Press, 2000.

[10] A. Chervenak, E. Deelman, I. Foster, L. Guy, W. Hoschek, A. Iamnitchi, C. Kesselman, P. Kunst, M. Ripeanu, B. Schwartzkopf, H. Stockinger, and B. Tierney, "Giggle: A Framework for Constructing Scalable Replica Location Services," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2002)*: ACM Press/IEEE Computer Computer Society Press, 2002, pp. 1-17.

[11] A. Chervenak, E. Deelman, C. Kesselman, B. Allcock, I. Foster, V. Nefedova, J. Lee, A. Sim, A. Shoshahi, B. Drach, D. Williams, and D. Middleton, "High-performance remote access to climate simulation data: a challenge problem for data grid technologies," *Parallel Computing*, vol. 29, pp. 1335-1356, 2003.

[12] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets," *Journal of Network and Computer Applications*, vol. 23, pp. 187-200, 2000.

[13] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh, and S. Koranda, "Mapping Abstract Complex Workflows onto Grid Environments," *Journal of Grid Computing*, vol. 1, pp. 25-39, 2003.

[14] E. Deelman, G. Singh, M. P. Atkinson, A. Chervenak, N. P. Chue Hong, C. Kesselman, S. Patil, L. Pearlman, and M. Su, "Grid-Based Metadata Services," in *Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM '04)*, 2004.

[15] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit.," *International Journal of High Performance Computing Applications*, vol. 11, pp. 115--128, 1997.

[16] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao, "Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation," in *Proceedings of the 14th Conference on Scientific and Statistical Database Management (SSDBM '02)*, 2002.

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

[18] N. Furmento, W. Lee, A. Mayer, S. Newhouse, and J. Darlington, "ICENI: An Open Grid Service Architecture Implemented with JINI," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2002)*. Baltimore, MD: ACM Press/IEEE Computer Society Press, 2002.

[19] A. S. Grimshaw and W. Wulf, "The Legion: Vision of a Worldwide Virtual Computer," *Communications of the ACM*, vol. 40, pp. 39--45, 1997.

[20] S. Hastings, S. Langella, S. Oster, and J. Saltz, "Distributed Data Management and Integration: The Mobius Project," *Proceedings of the Global Grid Forum 11 (GGF11) Semantic Grid Applications Workshop, Honolulu, Hawaii, USA.*, pp. 20-38, 2004.

[21] S. Langella, S. Oster, S. Hastings, F. Siebenlist, T. Kurc, and J. Saltz, "Dorian: Grid Service Infrastructure for Identity Management and Federation," presented at The 19th IEEE Symposium on Computer-Based Medical Systems, Special Track: Grids for Biomedical Informatics, Salt Lake City, Utah., 2006.

[22] R. Oldfield and D. Kotz, "Armada: A Parallel File System for Computational Grid," in *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid (CCGrid2001)*. Brisbane, Australia: IEEE Computer Society Press, 2001.

[23] M. Sato, H. Nakada, S. Sekiguchi, S. Matsuoka, U. Nagashima, and H. Takagi, "Ninf: A Network based Information Library for a Global World-Wide Computing Infrastructure," in *Proceedings of the Conference on High Performance Computing and Networking (HPCN '97) (LNCS-1225)*, 1997, pp. 491-502.

[24] G. Singh, S. Bharathi, A. Chervenak, E. Deelman, C. Kesselman, M. Mahohar, S. Pail, and L. Pearlman, "A Metadata Catalog Service for Data Intensive Applications," in *Proceedings of the ACM/IEEE Supercomputing Conference (SC2003)*, 2003.

[25] G. Singh, E. Deelman, G. Mehta, K. Vahi, M. Su, B. Berriman, J. Good, J. Jacob, D. Katz, A. Lazzarini, K. Blackburn, and S. Koranda, "The Pegasus Portal: Web Based Grid Computing," in *Proceedings of the 20th Annual ACM Symposium on Applied Computing*. Santa Fe, New Mexico, 2005.

[26] J. Smith, A. Gounaris, P. Watson, N. W. Paton, A. A. Fernandes, and R. Sakellariou, "Distributed Query Processing on the Grid.," presented at Proceedings of the Third Workshop on Grid Computing (GRID2002), Baltimore, MD, 2003.

[27] D. Thain, J. Basney, S. Son, and M. Livny, "Kangaroo Approach to Data Movement on the Grid," in *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC-10)*, 2001.

[28] L. Weng, G. Agrawal, U. Catalyurek, T. Kurc, S. Narayanan, and J. Saltz, "An Approach for Automatic Data Virtualization," in *Proceedings of the 13th IEEE International Symposium on High-Performance Distributed Computing (HPDC-13)*. Honolulu, Hawaii, 2004, pp. 24-33.

[29] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," Open Grid Service Infrastructure Working Group Technical Report, Global Grid Forum. http://www.globus.org/alliance/publications/papers/ogsa.pdf 2002.

[30] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations.," *International Journal of Supercomputer Applications*, vol. 15, pp. 200-222, 2001.

[31] E. Cerami, *Web Services Essentials*: O'Reilly & Associates Inc., 2002.

[32] S. Graham, S. Simeonov, T. Boubez, D. Davis, G. Daniels, Y. Nakamura, and R. Neyama, *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI*: SAMS Publishing, 2002.

[33] K. Czajkowski, D. F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe, "The WS-Resource Framework version 1.0," vol. 2004, 2004.

[34] J. Saltz, S. Oster, S. Hastings, T. Kurc, W. Sanchez, M. Kher, A. Manisundaram, K. Shanbhag, and P. Covitz, "caGrid: Design and Implementation of the Core Architecture of the Cancer Biomedical Informatics Grid," *Bioinformatics. (in press).* 2006.

[35] S. Langella, S. Hastings, S. Oster, T. Kurc, U. Catalyurek, and J. Saltz, "A Distributed Data Management Middleware for Data-Driven Application Systems," in *Proceedings of the 2004 IEEE International Conference on Cluster Computing (Cluster 2004)*, 2004.

[36] K. Bhatia, S. Chandra, and K. Mueller, "GAMA: Grid Account Management Architecture," San Diego Supercomputer Center (SDSC), UCSD Technical Report. #TR-2005-3, 2005.

[37] I. Foster, C. Kesselman, S. Tuecke, V. Volmer, V. Welch, R. Butler, and D. Engert, "A National Scale Authentication Infrastructure," *IEEE Computer*, vol. 33, pp. 60-66, 2000.

[38] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Czajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, and S. Tuecke, "Security for Grid Services," presented at 12th International Symposium on High Performance Distributed Computing (HPDC-12), 2003.

[39] H. Morohoshi and R. Huang, "A User-friendly Platform for Developing Grid Services over Globus Toolkit 3," presented at The 2005 11th International Conference on Parallel and Distributed Systems (ICPADS'05), 2005.

[40] S. Mizuta and R. Huang, "Automation of Grid Service Code Generation with AndroMDA for GT3," presented at The 19th International Conference on Advanced Information Networking and Applications (AINA'05), 2005.

[41] G. von Laszewski, I. Foster, J. Gawor, and P. Lane, "A Java Commodity Grid Kit," *Concurrency and Computation: Practice and Experience*, vol. 13, pp. 643-662, 2001.

[42] G. von Laszewski, I. Foster, J. Gawor, W. Smith, and S. Tuecke, "CoG Kits: A Bridge Between Commodity Distributed Computing and High Performance Grids," presented at ACM Java Grande 2000 Conference, 2000.

[43] R. Buyya and S. Venugopal, "The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report," presented at the First IEEE International Workshop on Grid Economics and Business Models (GECON 2004), New Jersey, USA, 2004.

[44] M. Humphrey and G. Wasson, "Architectural Foundations of WSRF.NET," *International Journal of Web Services Research*, vol. 2, pp. 83-97, 2005.

[45] M. Smith, T. Friese, and B. Freisleben, "Model Driven Development of Service Oriented Grid Applications," presented at Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW '06), 2006.

Technical Manuals/Articles

National Cancer Institute. "caCORE SDK 3.2.1 Programmer's Guide", ftp://ftp1.nci.nih.gov/pub/cacore/SDK/v3.2.1/caCORE_SDK_3.2.1_Programmers_Guide.pdf

National Cancer Institute. "caCORE 3.2 Technical Guide", ftp://ftp1.nci.nih.gov/pub/cacore/caCORE3.2_Tech_Guide.pdf

Java Bean Specification: http://java.sun.com/products/javabeans/docs/spec.html

Foundations of Object-Relational Mapping: http://www.chimu.com/publications/objectRelational/

Object-Relational Mapping articles and products:

http://www.service-architecture.com/object-relational-mapping/

Hibernate Reference Documentation: http://www.hibernate.org/hib_docs/reference/en/html/

Basic O/R Mapping: http://www.hibernate.org/hib_docs/reference/en/html/mapping.html

Java Programming: http://java.sun.com/learning/new2java/index.html

Javadoc tool: http://java.sun.com/j2se/javadoc/

JUnit: http://junit.sourceforge.net/

Extensible Markup Language: http://www.w3.org/TR/REC-xml/

XML Metadata Interchange: http://www.omg.org/technology/documents/formal/xmi.htm

Global Grid Forum: http://www.gridforum.org

Globus: http://www.globus.org

Mobius: http://www.projectmobius.org

W3C: http://www.w3c.org

OGSA-DAI: http://www.ogsadai.org

Apache: http://www.apache.org

Globus Toolkit 3 Programmer's Tutorial:

http://gdp.globus.org/gt3-tutorial/singlehtml/progtutorial_0.4.3.html

XPath tutorial: http://www.w3schools.com/xpath/xpath_syntax.asp

Globus Security Overview:

http://www.ogsadai.org.uk/docs/OtherDocs/SECURITY-FOR-DUMMIES.pdf

High level Overview of Grid:

http://gridcafe.web.cern.ch/gridcafe/index.html

Overview of Globus Toolkit 3 and the OGSI architecture :

http://www-128.ibm.com/developerworks/grid/library/gr-gt3/

caBIG Material

caBIG: http://cabig.nci.nih.gov/

caBIG Compatibility Guidelines: http://cabig.nci.nih.gov/guidelines_documentation

caCORE Material

caCORE: http://ncicb.nci.nih.gov/NCICB/infrastructure

caBIO: http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/caBIO

caDSR: http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/cadsr

EVS: http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/vocabulary

CSM: http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/csm

Appendix B Glossary

This glossary provides definitions for acronyms, objects, tools and other terms related to caGrid.

Term	Definition
API	Application Programming Interface
Authz	caGrid Authorization component
BPEL	Business Process Execution Language
CA	Certificate Authority
caArray	cancer Array Informatics
caBIG	cancer Biomedical Informatics Grid
caBIO	Cancer Bioinformatics Infrastructure Objects
caCORE	cancer Common Ontologic Representation Environment
caDSR	Cancer Data Standards Repository
caGrid	Current test bed architecture of caBIG
CRL	Certificate Revocation List
CSM	Common Security Module
CVS	Concurrent Versions System
DAO	Data Access Objects
DN	Distinguished Name
IdP	Identity Provider
EPR	End Point Reference
EVS	Enterprise Vocabulary Services
GAARDS	Grid Authentication and Authorization with Reliably Distributed Services
GDE	Introduce Graphical Development Environment
GForge	Primary site for collaborative project development for the NCI Center for Bioinformatics (NCICB) and for the NCI's Cancer Biomedical Informatics Grid™ (caBIG)
GGF	Global Grid Forum
GME	Mobius Global Model Exchange - DNS-like service for the universal creation, versioning, and sharing of data descriptions
Grid Service	Basically a Web Services with improved characteristics and standard services like stateful and potentially transient services, Service Data, Notifications, Service Groups, portType extension, and Lifecycle management.
GSH	Grid Service Handle
GSI	Grid Security Infrastructure - represents the latest evolution of the Grid Security Infrastructure. GSI in GT3 builds off of the functionality present in early GT2 toolkit releases - X.509 certificates, TLS/SSL for authentication

Term	Definition
	and message protection, X.509 Proxy Certificates for delegation and single sign-on.
GTS	Grid Trust Service - maintains a federated trust fabric of all the trusted digital signers in the grid
HTTP	Hypertext Transfer Protocol
ISO	International Organization for Standardization
JAAS	Java Authentication and Authorization Service
JAR	Java Archive
Javadoc	Tool for generating API documentation in HTML format from doc comments in source code (http://java.sun.com/j2se/javadoc/)
JDBC	Java Database Connectivity
JUnit	A simple framework to write repeatable tests (http://junit.sourceforge.net/)
LDAP	Lightweight Directory Access Protocol
MAGE	MicroArray and Gene Expression
MAGE-OM	MicroArray Gene Expression - Object Model
Metadata	Definitional data that provides information about or documentation of other data.
MGED	Microarray Gene Expression Data
Mobius	An array of tools and middleware components to coherently share and manage data and metadata in a Grid and/or distributed computing environment.
NCI	National Cancer Institute
NCICB	National Cancer Institute Center for Bioinformatics
OGSA	Open Grid Services Architecture - developed by the Global Grid Forum, aims to define a common, standard, and open architecture for grid-based applications.
OGSI	Open Grid Services Infrastructure -gives a formal and technical specification of what a Grid Service is. In other words, for a high-level architectural view of what Grid Services are, and how they fit into the next generation of grid applications
PKI	Public Key Cryptography
RDBMS	Relational Database Management System
SAML	Secure Access Markup Language
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
TRA	Trusted Registration Authority
UI	User Interface
UID	User Identification

Term	*Definition*
UML	Unified Modeling Language
UPT	User Provisioning Tool
URL	Uniform Resource Locators
Virtualization	Make a computational or data resource available to caBIG community - some people call "Gridification"
VO	Virtual Organization
WAR	Web Application Archive
Web Service	Application to application communication using web based service interfaces as describe by the Web Services 1.0 or 2.0 specification.
WSDD	Web Service Deployment Descriptor
WSDL	Web Services Description Language
WSRF	Web Services Resource Framework
X.509 Certificate	With its corresponding private key forms a unique credential or so-called “grid credential” within the grid
XMI	XML Metadata Interchange (http://www.omg.org/technology/documents/formal/xmi.htm) - The main purpose of XMI is to enable easy interchange of metadata between modeling tools (based on the OMG-UML) and metadata repositories (OMG-MOF) in distributed heterogeneous environments
XML	Extensible Markup Language (http://www.w3.org/TR/REC-xml/) - XML is a subset of Standard Generalized Markup Language (SGML). Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML
XPath	XML query/traversal language adhering to the XPath specification set forth by the W3C.

Index