GEWORKBENCH V.1.0

USER MANUAL – SUPPLEMENT 1

(Advanced Services)

Document Change History

Version Number	Date	Contributor	Description
V1.0 draft	4/24/2007	Kenneth C. Smith	This supplement to the geWorkbench manual describes several new features, including caGrid Services, caScript, and a caArray query interface.
V1.0	5/1/2007	Kenneth C. Smith	Final version

caBIG™ cancer Biomedical Informatics Grid™

an initiative of the National Cancer Institute

Columbia University

Joint Centers for Systems Biology

caBIG^{tm}

National Cancer Institute Center for Bioinformatics

Copyright and License page

SOFTWARE LICENSE AGREEMENT

This software was developed by Columbia University in conjunction with First Genetic Trust and the National Cancer Institute, and so to the extent government employees are co-authors, any rights in such works shall be subject to Title 17 of the United States Code, section 105.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

2. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

"This product includes software developed by the Columbia University, First Genetic Trust and the National Cancer Institute."

If no such end-user documentation is to be included, this acknowledgment shall appear in the software itself, wherever such third-party acknowledgments normally appear.

3. This license does not authorize the incorporation of this software into any proprietary programs.

4. THIS SOFTWARE IS PROVIDED "AS IS," AND ANY EXPRESSED OR IMPLIED WARRANTIES, (INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE) ARE DISCLAIMED. IN NO EVENT SHALL THE COLUMBIA UNVIERSITY OR THEIR AFFILIATES BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, 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 SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5.Below is the list of all third party software used in geWorkbench and their license information.

This product includes software developed by the Apache Software Foundation. Batik, Xerces,and Xalan are part of Apache XML project. Byte Code Engineering Library, POI,

Members of the Development Team[1]		
Development	***User's Guide***	***Program Management***
Names of developers	*Names of technical writers and reviewers*	*Names of program managers*
Andrea Califano	Kenneth Smith	Aris Floratos
Aris Floratos	Eileen Daly	Kenneth Smith
Matt Hall		
Michael Honig		
Bernd Jagla		
Kiran Keshav		
Manjunath Kustagi		
John Watkinson		
Xiaoqing Zhang		

[1] All contributors are currently members of the Joint Centers for Systems Biology, Columbia University, New York, NY.

Contacts and Support	
Training contact	N/A
Support contact	http://gforge.nci.nih.gov/forum/?group_id=78

LISTSERV Facilities Pertinent to software teams

LISTSERV	***URL***	***Name***
geWorkbench	http://gforge.nci.nih.gov/forum/?group_id=78	geWorkbench Open Discussion Forum
caBIO_Users	https://list.nih.gov/archives/cabio_users.html	caBIO Users Discussion Forum
caBIO_Developers	https://list.nih.gov/archives/cabio_developers.html	caBIO Developers Discussion Forum
caDSR_Users	https://list.nih.gov/archives/cadsr_users.html	Cancer Data Standards Repository
NCIEVS-L Listserv	https://list.nih.gov/archives/ncievs-l.html	NCI Vocabulary Services Information
CAARRAY_DEVELOPERS-L	https://list.nih.gov/archives/caarray_developers-l.html	caARRAY Developers Forum
CAARRAY_MAGE-OM_API	https://list.nih.gov/archives/caarray_mage-om_api.html	caArray MAGE-OM API Forum
CAGRID_DEVELOPERS	https://list.nih.gov/archives/cagrid_developers.html	caGRID Developers Forum
CAGRID_USERS-L	https://list.nih.gov/archives/cagrid_users-l.html	caGRID Users Forum

Table of Contents

Figure Legends

Chapter 1 **Introduction to the Manual**

This manual is intended for users of geWorkbench who wish to experiment with newly added advanced features. It is directed at the bench scientist and bioinformatician. This manual does not provide installation instructions for geWorkbench nor for grid services.

Topics in this introductory chapter include:

- Topics covered in this supplement
- Organization of the Guide
- Getting Started with geWorkbench

Topics covered in this supplement

This manual is a supplement to the geWorkbench User Manual v.1.3, dated September 14, 2006. That manual explains the basic principles, design goals, and uses of geWorkbench, a software platform which is centered around the single or joint analysis of gene expression microarray and sequence data.

This supplement adds material covering newly developed features which provide advanced functionality. Some must still be regarded as experimental. As the software matures, final descriptions will be incorporated directly into the User Manual.

The new software modules described in this supplement relate to

1. use of geWorkbench within the context of the caGRID infrastructure. Several analytical routines already supported directly within geWorkbench have been developed as formal caGRID services, with an appropriate service interface present within geWorkbench. They are initially intended to be used in the analysis of microarray data. They are:

    a. Hierarchical Clustering

    b. SOM (Self-Organizing Maps)

    c. ARACNE (a gene network reverse-engineering tool

2. a query interface for caARRAY which allows searches on available annotation fields

3. use of the caSCRIPT scripting language developed specifically for geWorkbench to automate the running of repetitive or complex tasks.

This manual will provide detailed examples on how to use these new modules.

Organization of the Guide

Chapter in geWorkbench User Manual Supplement 1	***Chapter Contents***
Chapter 1	An introduction to using this Manual
Chapter 2	A general introduction to geWorkbench, and a high-level description of the advanced services covered in this manual.
Chapter 3	Description of the geWorkbench User Interface, including layout, basic file operations, and using Online Help
Chapter 4	Using caGRID-based remote analytical services
Chapter 5	Performing queries against a caARRAY database using MAGE-compliant data fields
Chapter 6	Using the java-like language scripting language caSCRIPT to automate routine or complex operations.
Chapter 7	Problems the user might encounter, with explanations.
Chapter 8	Glossary
Appendix A	References
Appendix B	Glossary

Getting Started with geWorkbench

To get started with geWorkbench you may refer to the following sections of this manual:

- This introduction provides an overview of the manual structure
- Review Chapter 2 for a brief overview of the software
- Review Chapter 3 to learn about the Graphical User Interface
- Refer to Chapters 4, 5, and 6 a description of how to use the advanced services covered in this supplement.

Please note that this manual is a supplement to the main geWorkbench User Manual. Additional information about running geWorkbench can be found there. Detailed instructions and step-by-step tutorials on how to install and run geWorkbench are available online at http://www.geworkbench.org/.

Document Text Conventions

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

Convention	***Description***	***Example***
Bold & Capitalized Command Capitalized command > Capitalized command	Indicates a Menu command Indicates Sequential Menu commands	**Admin > Refresh**
TEXT IN SMALL CAPS	Keyboard key that you press	Press ENTER.
TEXT IN SMALL CAPS + TEXT IN SMALL CAPS	Keyboard keys that you press simultaneously	Press SHIFT + CTRL and then release both.
Boldface type	Options that you select in dialog boxes or drop-down menus. Buttons or icons that you click.	In the Open dialog box, select the file and click the Open button.
Italics	Used to reference other documents, sections, figures, and tables.	*caCORE Software Development Kit 1.0 Programmer's Guide*
Italic boldface type	Text that you type	In the New Subset text box, enter *Proprietary Proteins.*
`Courier typestyle`	Used for filenames, directory names, commands, file listings, source code examples and anything that would appear in a Java program, such as methods, variables, and classes.	`URL_definition ::= url_string`
Note:	Highlights a concept of particular interest	Note: This concept is used throughout the installation manual.
Warning!	Highlights information of which you should be particularly aware.	Warning! Deleting an object will permanently delete it from the database.
{}	Curly brackets are used for replaceable items.	Replace {root directory} with its proper value such as c:\cabio

Table 1. 1 Document Conventions

Chapter 2 Overview of the Software

This chapter provides an overview of geWorkbench itself, and a description of the particular advanced services that will be covered in this manual: access to remote analytical services via caGRID, a query interface for caARRAY, and scripting with geWorkbench using caSCRIPT.

Topics in this chapter include:

- Introduction to geWorkbench
- Components of geWorkbench
- Features and Functions of the Software
- Brief Description of the User Interface*.*

Introduction to geWorkbench

geWorkbench is a framework for bioinformatics data analysis. It provides data management, visualization, analysis and retrieval capabilities. It has been primarily constructed for analysis of data derived from gene expression microarray experiements, and allows pulling in many different resources to this end, including sequence, gene ontology, promoter analysis, and standard analytic techniques such as the t-test, hierarchical clustering, and gene network reverse-engineering.

geWorkbench has a modular, component-based design. New modules can easily be written and added as the need arises. A primary aim is to allow easy integration of different forms of bioinformatic data analysis. Such integration of different software routines removes the common hinderance of needing to reformat data for each different type of analysis undertaken.

Extensive documentation and training material for geWorkbench can be found on its main website at http://www.geworkbench.org/. There are wiki-based tutorials there for almost all components of the application. These tutorials are more applied in nature than the material in the printed manual. The software can be downloaded via links found on the ‘Download’ section of that site. Those links refer to the actual archival location of the software, which is the GForge site maintained by the NCICB. All official releases of the software can be downloaded from that site.

Components of geWorkbench

geWorkbench: geWorkbench v.1.0 is a Java application which is run on the User's local Windows, Macintosh or Linux workstation. This main application also serves as a front-end client to a number of external computational and data services. Such services already present in geWorkbench include the ability to run BLAST jobs on NCBI servers, and to retrieve gene, pathway and sequence information from sources such as UC Santa Cruz and the NCICB. This supplement also describes caSCRIPT, which is a Java-like scripting language that can be used to automate tasks within geWorkbench. It is an interpreted language which runs within the geWorkbench client.

caGRID: The goal of caGRID is to provide standardized, reusable services. caGRID uses an enhanced variety of web-services to manage communications between the local client, in this case geWorkbench, and remote services. In implementing a framework that is intended to allow programs to interoperate, it provides a mechanism for all data and parameters passed on the grid to be of known, registered types.

geWorkbench is a consumer of grid services. As part of a caGRID proof-of-concept development project, the geWorkbench team has implemented three remote analytical grid services which have been approved at the caBIG silver-level: Hierarchical Clustering, SOM, and ARACNE. These remote services are accessible via caGRID through geWorkbench.

caArray: geWorkbench has previously supported data retrieval from instances of the caArray microarray gene expression database. The latest version of caArray now supports queries against annotations stored in the database concerning arrays and experiments. geWorkbench has implemented an interface based on the MAGE-OM API allowing such queries to be composed and dispatched.

Features and Functions of the Software

Three new sets of functionality are covered in this supplement to the User Manual. For all other topics please see the geWorkbench User Manual v.1.3. The new developemts covered here are:

1. use of geWorkbench within the context of the caGRID infrastructure to access remote analytical services.

2. development of a graphical user interface for MAGE-OM API-based queries of caARRAY, which allows searches on available annotation fields.

3. use of the caSCRIPT scripting language developed specifically for geWorkbench to automate the running of repetitive or complex tasks.

The new grid services covered in this supplement are Hierarchical Clustering, Self-Organizing Maps (SOM) and ARACNE. These are routines which can be used in the analysis of microarray data. These routines already existed in geWorkbench and function in the same manner as they did before. The difference is that now the user is provided with the option of discovering and choosing a caGRID node on which to run the computation. To this end, a **Service** tab has been added to the geWorkbench GUI for each routine.

Hierarchical Clustering is used to group a set of genes or microarrays based on similarities in the data. It can be used for example to find groups of genes with similar expression patterns across different tissue types or treatment conditions. Its output is a single connected graph of genes and/or arrays. A graphical viewer in geWorkbench displays the result and allows its manipulation, e.g. retrieving the list of genes from a particular branch of the graph.

SOM analysis divides a data set into a user-chosen number of discreet sets, again based on similarities in the data. Several parameters can be adjusted to determine how this grouping will be performed. Again, a graphical viewer is provided to inspect the results.

ARACNE can be used to find which gene is influencing the expression of another – it is a network reverse-engineering algorithm. It is based on an information-theory calculation known as mutual information. The result is believed to be more robust than a simple correlation calculation. It generally requires at least 100 array experiments for proper sensitivity.

The new geWorkbench query interface for caArray makes use of annotations stored in the database under the MAGE standard. It allows a query to be constructed using several standard fields, such as species, organ, or microarray platform.

caSCRIPT is a Java-like interpreted scripting language that can be used within geWorkbench. Scripts can be stored and reloaded for later execution. caSCRIPT can execute any geWorkbench analysis in the same way as it would be launched interactively from the geWorkbench GUI.

Brief Description of the User Interface

geWorkbench is a Java application. it is currently written to the Sun Java 1.5 (aka Java 5) specification and requires a Java 1.5 JRE to be installed on the client system in order to run. The user interface provides built-in areas for managing data, sub-setting data, viewing data and results, and finally a region for data analysis. There is a built-in **Online Help** system which covers most standard functionality which has been released as part of the formal release process. One hallmark of the geWorkbench GUI design is that it only displays those components relevant to the currently selected data type. That is, if a gene sequence object is selected in the data window, only components relevant to manipulating sequences will be displayed.

Chapter 3 The geWorkbench User Interface

This chapter provides a basic introduction to using the geWorkbench Graphical User Interface (GUI) as it relates to the advanced services described in this supplement. A more detailed description can also be found in the geWorkbench User Manual v.1.3 and in the online tutorials (http://www.geworkbench.org/).

Topics covered include:

- Basic Layout
- Online Help.
- Working with Data Files

Basic Layout

The graphical user interface for geWorkbench is divided into four major sections:

1. Data management - Workspace and Projects (upper left)
2. Marker and Array/Phenotype set selection and management (lower left)
3. Visualization tools (upper right)
4. Analytical tools (lower right)

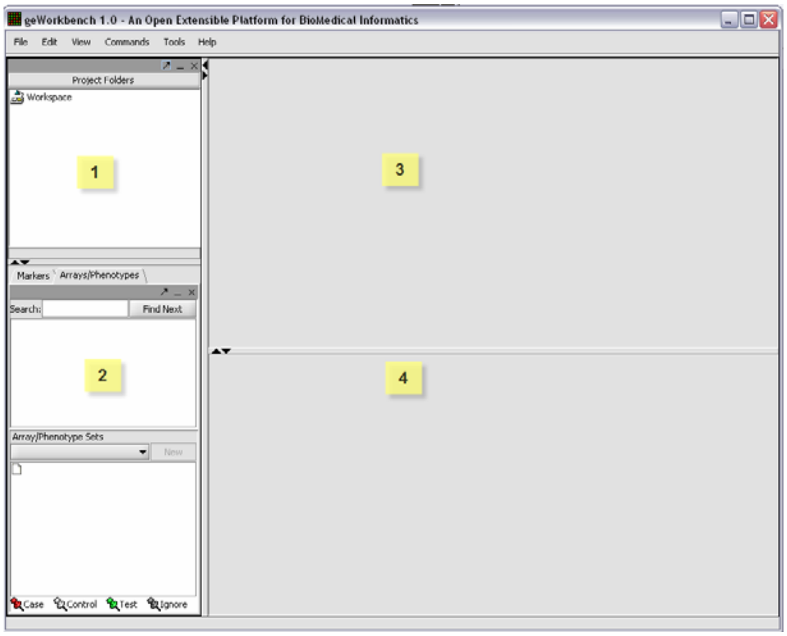


Figure 3-1 Basic geWorkbench GUI layout

Menu Bar

The GUI provides a menu bar at top with a standard choice of commands. Many commands that are available in the menu bar are also available by right-clicking on data objects.

Data management area (1)

Working with geWorkbench involves creating a project within the top-level workspace. Open data files and the results of data transformation or analysis are stored within a project. A workspace can contain more than one project at a time, allowing data to be organized as desired. A workspace and all the projects and data within it can be saved and later reloaded.

Set selection and management (2)

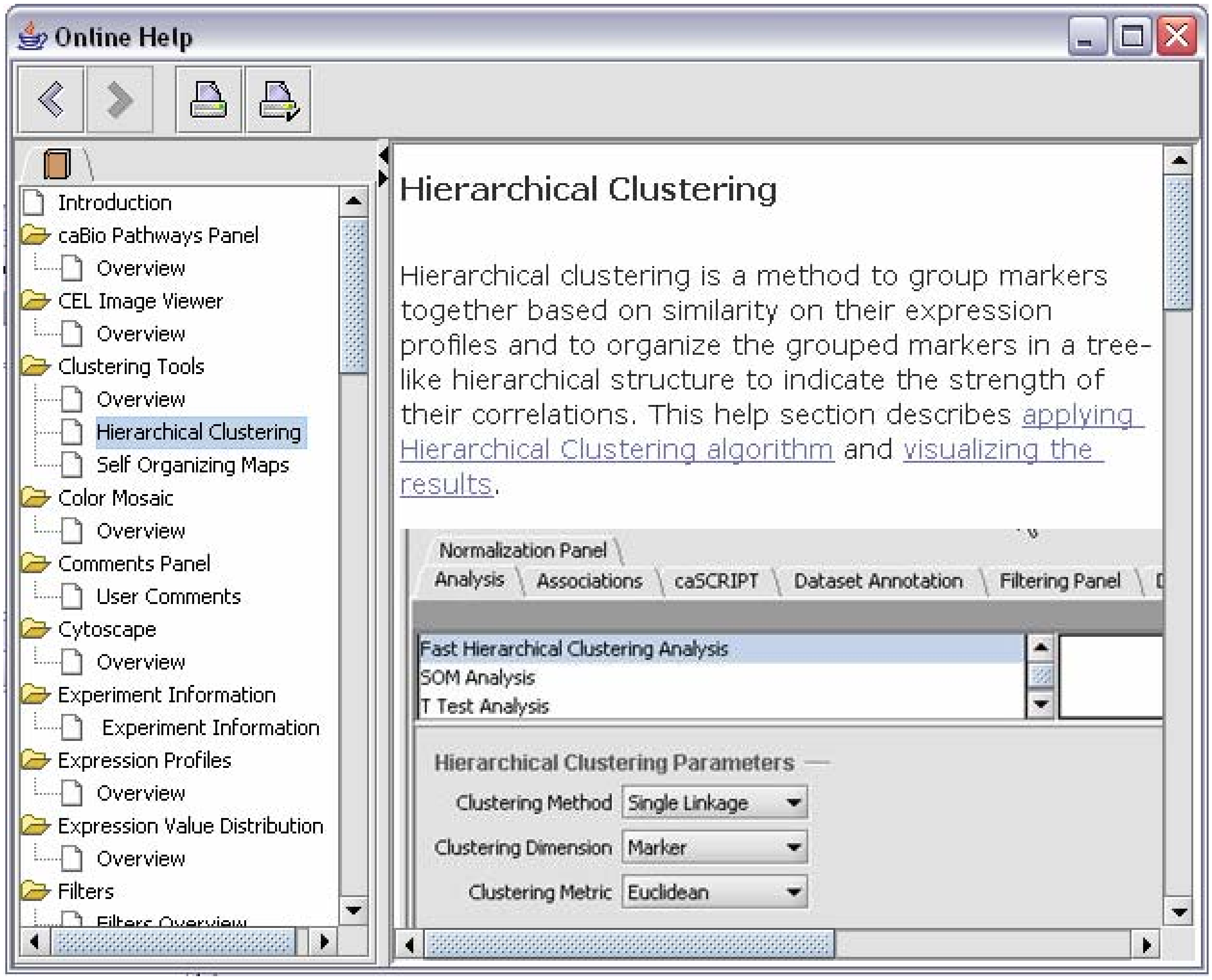
A key feature of geWorkbench is the ability to work with defined sets of markers or arrays. This allows subsets of data to be analyzed, and allows for passing of selected subsets of data between different components. For example, the t-test can be used to create a list of markers showing a significant difference in expression between two states, and this list can then be used to retrieve relevant sequences or annotations.

Visualization and Analysis tools (3 and 4)

geWorkbench works such that only the visualization and analysis components relevant to the type of dataset currently selected in the Project Folders area (1) are displayed through tabs in their respective areas (3 and 4). Thus choosing a microarray dataset will result in a different set of tabs being displayed as compared with those seen when a nucleotide sequence file is selected. When a new data file is loaded, or an analysis produces a new data set, not only is it added to the Project area (1), but an appropriate viewer in the Visualization area (3) is automatically selected.

Online Help

Figure 3-2 shows the **Online Help** interface. **Online Help** is found as a menu item under **Help** on the top menu bar. **Online Help** is provided for all geWorkbench modules which have been included in a formal release. They focus on the actual use of particular controls within a given module, e.g. button actions, definition of parameters etc.



Hierarchical Clustering

Hierarchical clustering is a method to group markers together based on similarity on their expression profiles and to organize the grouped markers in a tree-like hierarchical structure to indicate the strength of their correlations. This help section describes applying Hierarchical Clustering algorithm and visualizing the results.

Clustering Method: Single Linkage
Clustering Dimension: Marker
Clustering Metric: Euclidean

Figure 3-2 Online Help

Working with Data Files

The basics of opening a data file and the details of working with microarray data sets are covered in the geWorkbench User Manual and in the online tutorials (http://www.geworkbench.org/). To aid in the usability of this manual the background needed to work with files will be quickly reviewed.

Prerequisites

Much of the functionality of geWorkbench currently depends on annotation files supplied by Affymetrix for their microarray chips. Due to licensing restrictions, these files are not distributed with geWorkbench as part of formal releases. The examples in this supplement to the User Manual however do not depend on annotation information. If you nonetheless would like to work with the full functionality of geWorkbench, the relevant file for the dataset used in this manual can be downloaded from the Affymetrix.com support web site. The file is named “HG_U95Av2_annot.csv”.

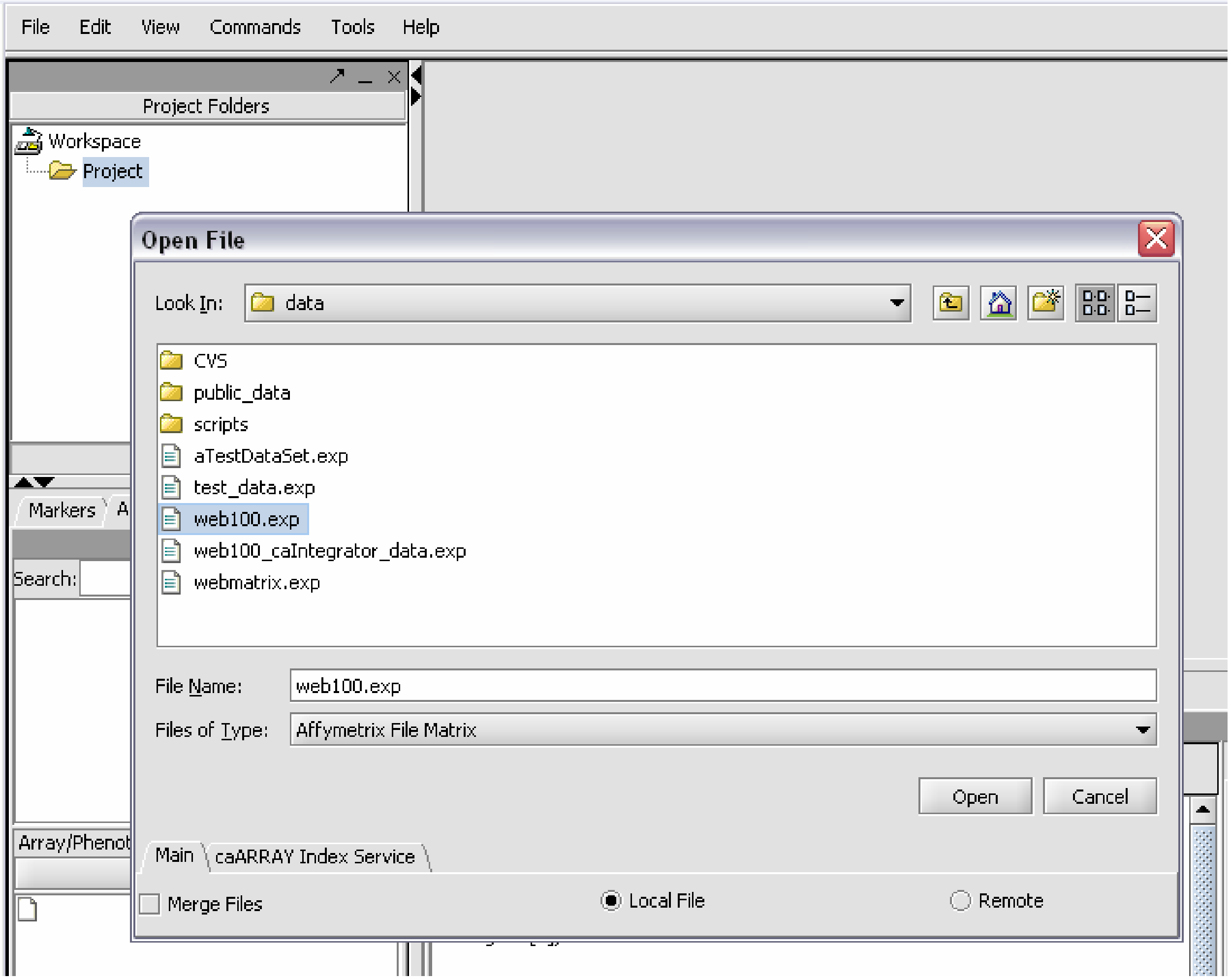
Workspaces, Projects, and Files

The top level of organization of data in geWorkbench is the Workspace. A Workspace can contain any number of Projects, which are used to organize data and results.

geWorkbench includes sample data files. In the example below we will open a small microarray data file, `web100.exp`. This file is in a custom format used by geWorkbench, which is termed the **Affymetrix File Matrix** format. It contains results from a number of different microarray chips that have been merged into one dataset.

Opening a file in a new Project (see Figure 3-3)

1. Right-click on Workspace and select **New Project**.
2. Right-click on **Project** and select **Open File(s)**.
3. By default, the file browser should open in the geWorkbench data directory. Select **File of Type** to be **Affymetrix File Matrix**.
4. Select the file `web100.exp`.



Look In: data

CVS
public_data
scripts
aTestDataSet.exp
test_data.exp
web100.exp
web100_caIntegrator_data.exp
webmatrix.exp

File Name: web100.exp

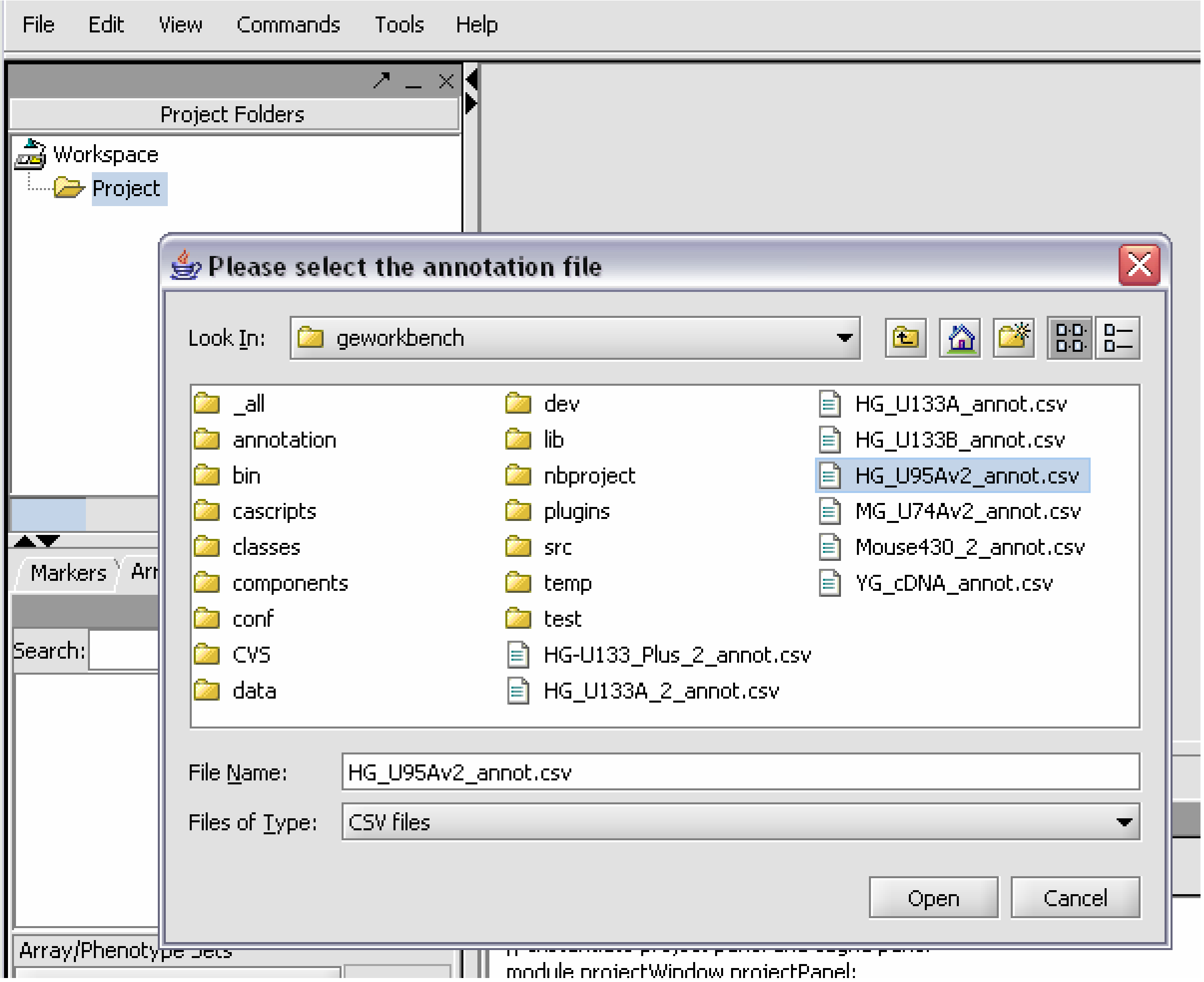
Files of Type: Affymetrix File Matrix

□ Merge Files ● Local File ○ Remote

Figure 3-3 Opening a file in a project

5. A box with information about annotation files will appear. Click **Continue**.

6. The file browser will open at the root of the geWorkbench installation directory (Figure 3-4). If the file `HG_U95Av2_annot.csv` is present, just press the **Open** button. If you have downloaded it to another directory, please navigate to that directory and open the file. If you do not have the file, just press **Cancel** and proceed without the annotation file.



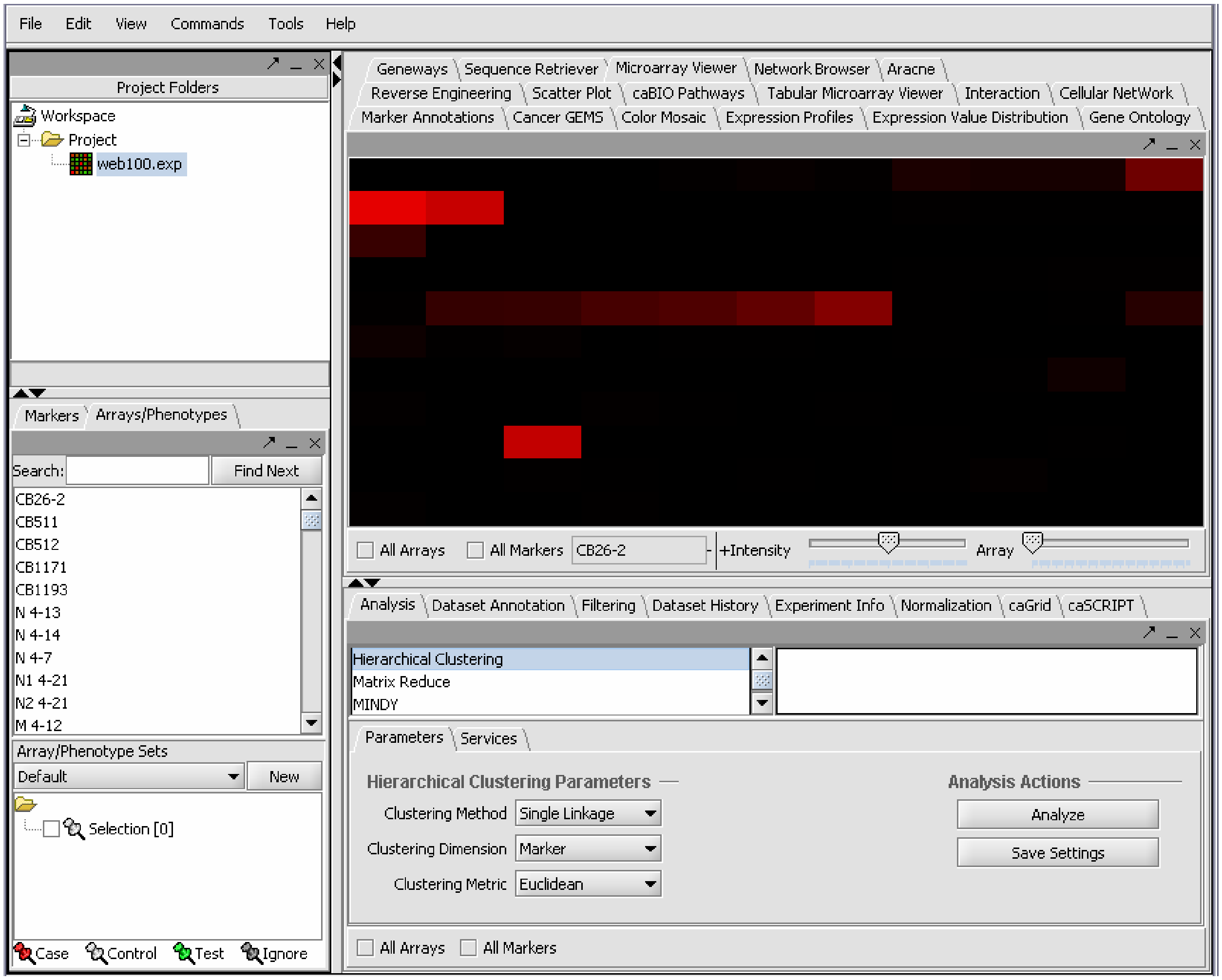
Look In: geworkbench

File Name: HG_U95Av2_annot.csv

Files of Type: CSV files

Figure 3-4 Opening the annotation file

The opened data file is now shown within the **Project Folders** area at upper left in the GUI. All components relevant to acting on microarray data have now appeared in the interface Figure 3-5.



Search:

CB26-2
CB511
CB512
CB1171
CB1193
N 4-13
N 4-14
N 4-7
N1 4-21
N2 4-21
M 4-12

Default

□ Selection [0]

□ All Arrays □ All Markers CB26-2

Hierarchical Clustering Parameters

Clustering Method: Single Linkage

Clustering Dimension: Marker

Clustering Metric: Euclidean

□ All Arrays □ All Markers

Figure 3-5 geWorkbench GUI showing all microarray-related modules

Chapter 4 Using caGRID Analytical Services

This chapter describes how geWorkbench can be used to dispatch jobs to remote servers using the caGRID infrastructure. Three services which have been implemented are Hierarchical Clustering, Self Organizing Maps, and ARACNE.

Topics covered in this chapter include:

- Using caGRID-based remote analytical services
- Hierarchical Clustering
- Self-Organizing Maps (SOM)
- ARACNE

Using caGRID-based remote analytical services

Three microarray analysis routines already available within geWorkbench have been deployed as caGRID analytical services: Hierarchical Clustering, Self-Organizing Maps (SOM), and ARACNE. The purpose is to remove large-scale calculations from the user's desktop machine, instead running them on appropriately scaled server systems. The remote systems could scale to cluster computers or other major hardware platforms as demand necessitates. This could be of particular interest for jobs requiring or benefiting from parallel programming, large memory, or which have long run-times. caGRID provides a standardized way to develop, deploy, and interact with these remote services.

Information about the hierarchical clustering and SOM routines is available in either or both of the geWorkbench User Manual and the tutorials available on http://www.geworkbench.org/. This material will not be repeated in detail in this supplement, as the only new feature is the availability of remote execution over caGRID. The material on ARACNE however is new.

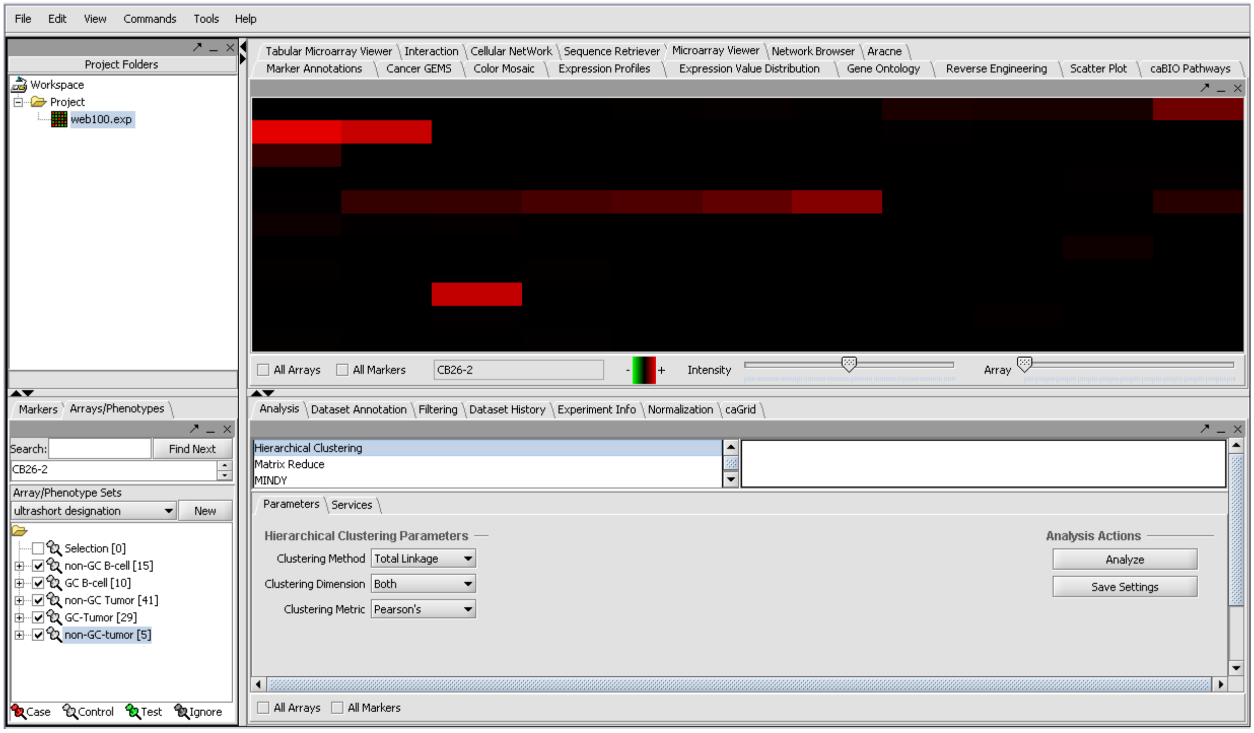
This chapter will primarily focus on invoking the grid-based services. Use of all three routines begins with loading a microarray dataset. An example of doing so has already been provided in Chapter 3, using the supplied data file `web100.exp`. Such datasets are two-dimensional, in that typically results from several experiments (chips) have been merged into a single data array, with genes on the vertical axis and the individual experiments on the horizontal axis when viewed in spreadsheet format.

Hierarchical Clustering

Hierarchical clustering is a method used to group data based on a measure of similarity. The two-dimensional microarray datasets used in geWorkbench can be clustered based on expression profiles for genes, for single arrays, or both. For example in clustering by gene, if the expression pattern for gene A across all experiments is very similar to that of gene B, then A and B will tend to cluster together.

Example

1. Load a microarray dataset, for example `web100.exp` as shown in Chapter 3 of this manual.
2. In the **Array/Phenotype** component at lower left in the GUI, select which arrays should be used in the analysis (Figure 4-1). Here we have activated all the arrays by checking the boxes next to each group. (The same could be accomplished by not checking any boxes, since the default is to include all arrays if nothing is selected).



Search: CB26-2

Array/Phenotype Sets: ultrashort designation

- □ Selection [0]
- ☑ non-GC B-cell [15]
- ☑ GC B-cell [10]
- ☑ non-GC Tumor [41]
- ☑ GC-Tumor [29]
- ☑ non-GC-tumor [5]

Hierarchical Clustering Parameters

- Clustering Method: Total Linkage
- Clustering Dimension: Both
- Clustering Metric: Pearson's

Figure 4-1 Activating the array sets

3. In the **Analysis** tab at lower right in the GUI, select **Hierarchical Clustering**. Select the **Services** tab (Figure 4-2). Click on the blue text **Change Index Service**. This will bring up a small input box.

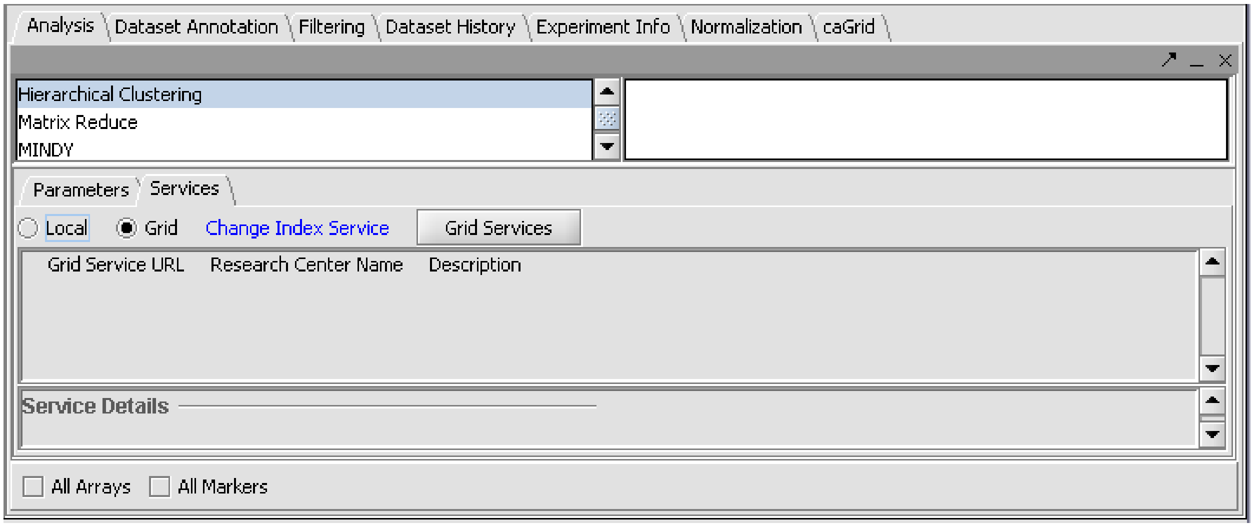
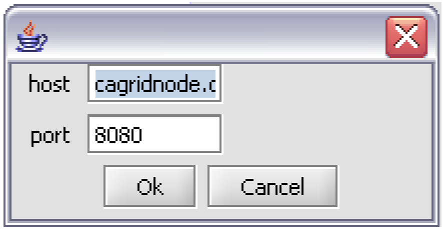


Figure 4-2 Setting the Grid Index Service

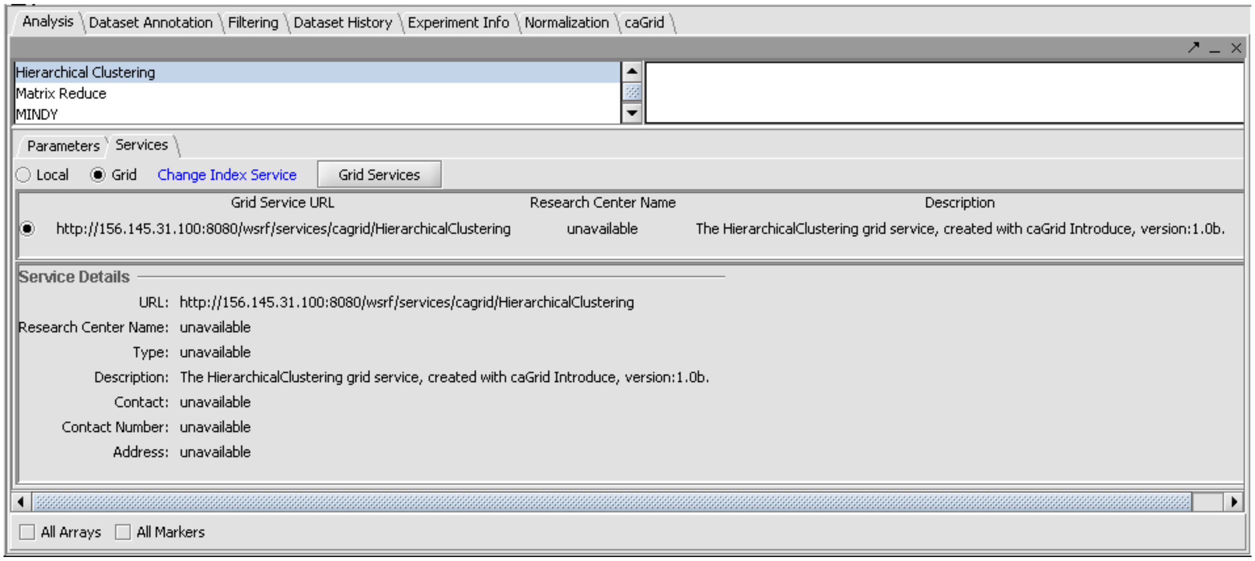
4. For **host**, (if it is not already present) enter `cagridnode.c2b2.columbia.edu` `(`Figure 4-3`)`. Leave the port set to `8080`. Press **OK**.


host cagridnode.c

port 8080

Figure 4-3 Adding a new grid node

5. Pressing **Grid Services** will display the available services (Figure 4-4). Select the available **Hierarchical Clustering** service. Once a service is selected, its details will display in the **Service Details** box below.



	Grid Service URL	Research Center Name	Description
●	http://156.145.31.100:8080/wsrf/services/cagrid/HierarchicalClustering	unavailable	The HierarchicalClustering grid service, created with caGrid Introduce, version:1.0b.

Service Details

URL:	http://156.145.31.100:8080/wsrf/services/cagrid/HierarchicalClustering
Research Center Name:	unavailable
Type:	unavailable
Description:	The HierarchicalClustering grid service, created with caGrid Introduce, version:1.0b.
Contact:	unavailable
Contact Number:	unavailable
Address:	unavailable

Figure 4-4 Selecting an available Hierarchical Clustering service

6. Now go back to the **Parameters** tab (see Figure 4-1 above) and set the following parameters for this example clustering operation:
    a. Clustering Method: **Total Linkage**
    b. Clustering Dimension: **Both**
    c. Clustering Metric: **Pearson's**
7. Click **Analyze**. The computation is carried out on the remote server and returned to geWorkbench, where the results are entered into the Project as a new data node (Figure 4-5), and displayed in the **Dendrogram Viewer** component (Figure 4-6).

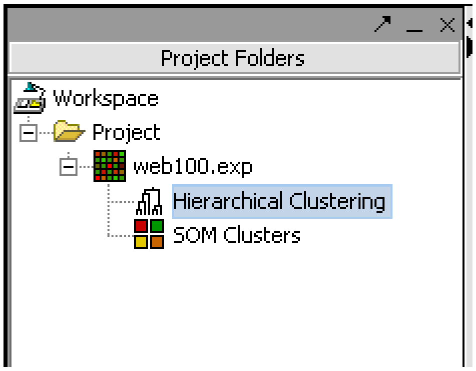
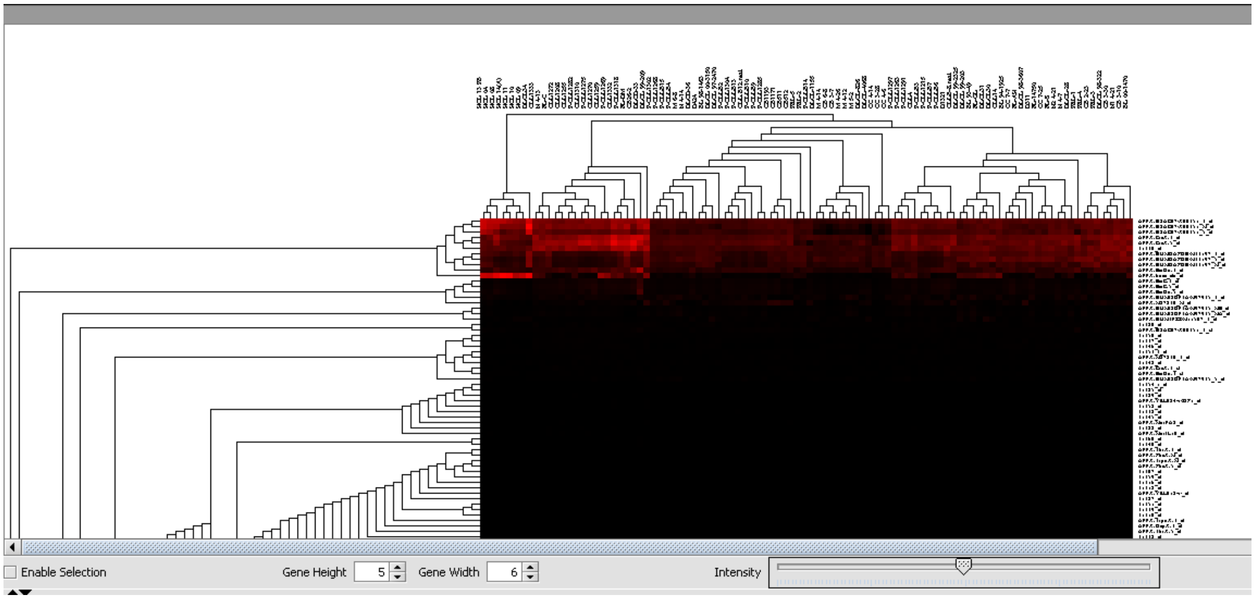


Figure 4-5 Result sets displayed in the Project Folder



□ Enable Selection

Gene Height 5

Gene Width 6

Figure 4-6 Hierarchical Clustering Dendrogram display

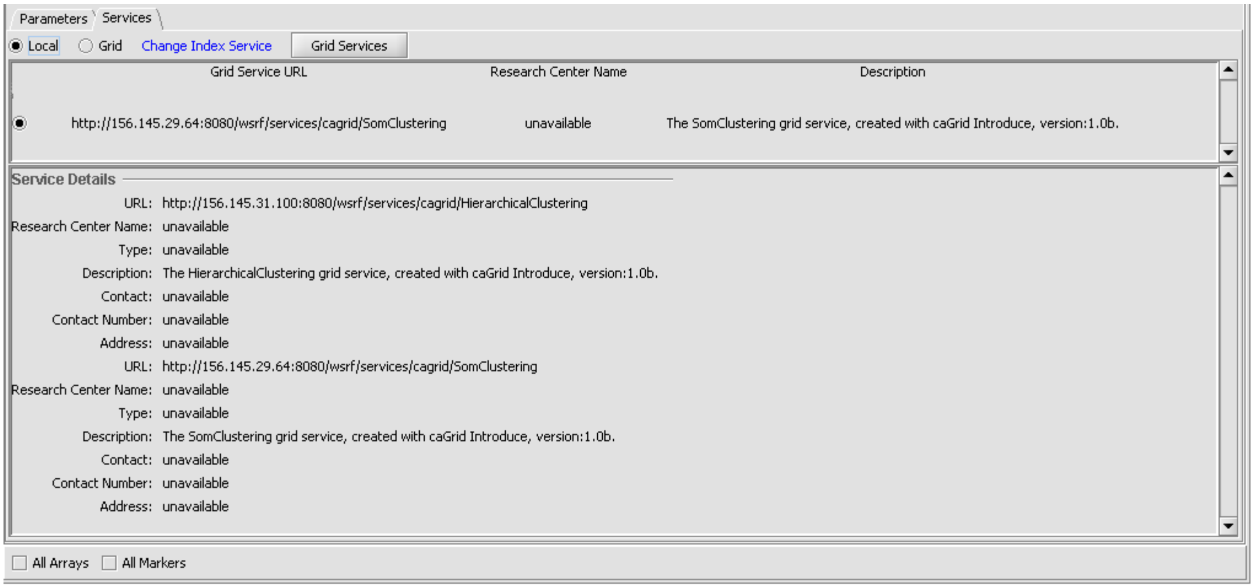
8. Within the Dendrogram display, the data can be manipulated in many ways, and clusters of genes can be selected and stored as named sets in the **Marker** component for further analysis (see the Clustering tutorial on http://www.geworkbench.org/ for details).

Self-Organizing Maps (SOM)

The SOM algorithm is used to divide a data set into a predetermined number groups based on similarity. Here we illustrate running SOM through the grid interface.

Example

1. Start as in the previous example by loading a microarray dataset, such as `web100.exp`. If it is already loaded, there is no need to reload it.
2. In the **Analysis** tab, select **SOM**.
3. Changing the Index Service, which should not be necessary, has been described above under Hierarchical Clustering.
4. Select an available **SOM** service. Once selected, the details will display in the area below it (Figure 4-7).



● Local ○ Grid

	Grid Service URL	Research Center Name	Description
●	http://156.145.29.64:8080/wsrf/services/cagrid/SomClustering	unavailable	The SomClustering grid service, created with caGrid Introduce, version:1.0b.

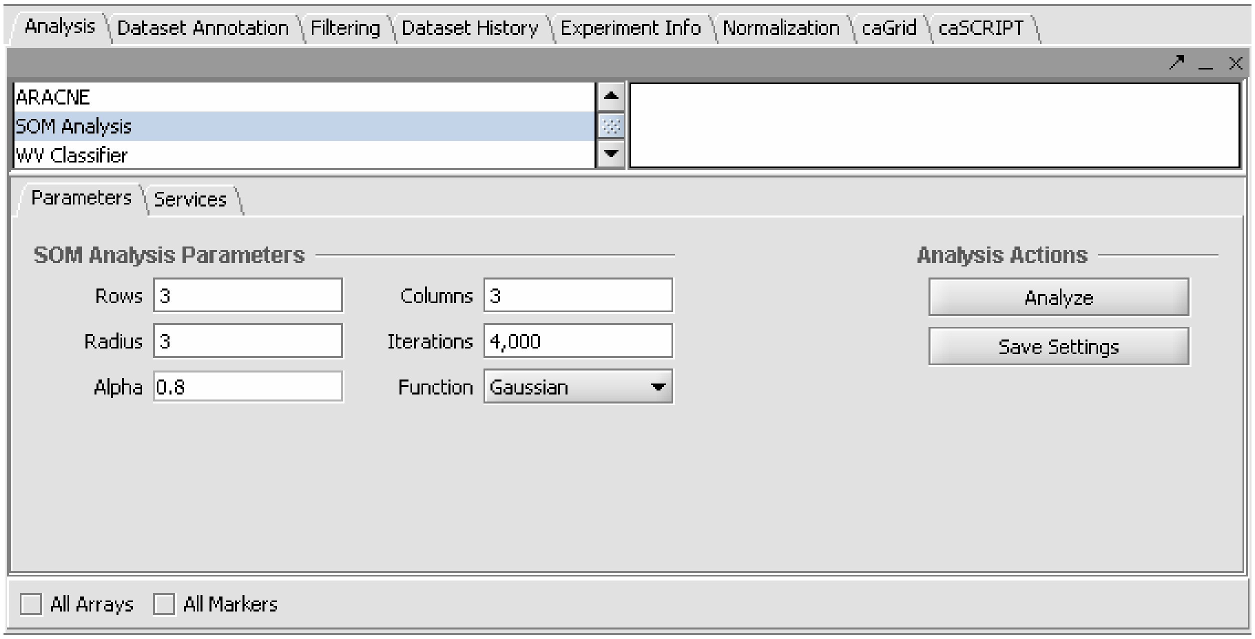
Service Details

URL: http://156.145.31.100:8080/wsrf/services/cagrid/HierarchicalClustering
Research Center Name: unavailable
Type: unavailable
Description: The HierarchicalClustering grid service, created with caGrid Introduce, version:1.0b.
Contact: unavailable
Contact Number: unavailable
Address: unavailable
URL: http://156.145.29.64:8080/wsrf/services/cagrid/SomClustering
Research Center Name: unavailable
Type: unavailable
Description: The SomClustering grid service, created with caGrid Introduce, version:1.0b.
Contact: unavailable
Contact Number: unavailable
Address: unavailable

□ All Arrays □ All Markers

Figure 4-7 SOM Grid Services

5. We will accept the default parameters, except setting the **Function** to **Gaussian** instead of Bubble (Figure 4-8):
   a. **Rows**: `3`
   b. **Columns**: `3`
   c. **Radius**: `3`
   d. **Iterations**: `4000`
   e. **Alpha**: `0.8`
   f. **Function**: **Gaussian**



SOM Analysis Parameters

Rows 3

Columns 3

Radius 3

Iterations 4,000

Alpha 0.8

Function Gaussian

□ All Arrays □ All Markers

Figure 4-8 Setting SOM parameters

6. Click **Analyze**. The result will be returned from the remote server and displayed in the SOM Viewer component as a 3x3 array of graphs. Each contains a set of expression profiles that are, within the boundaries established by the parameters chosen, similar to each other. Several clearly different groups can be distinguished(Figure 4-9)

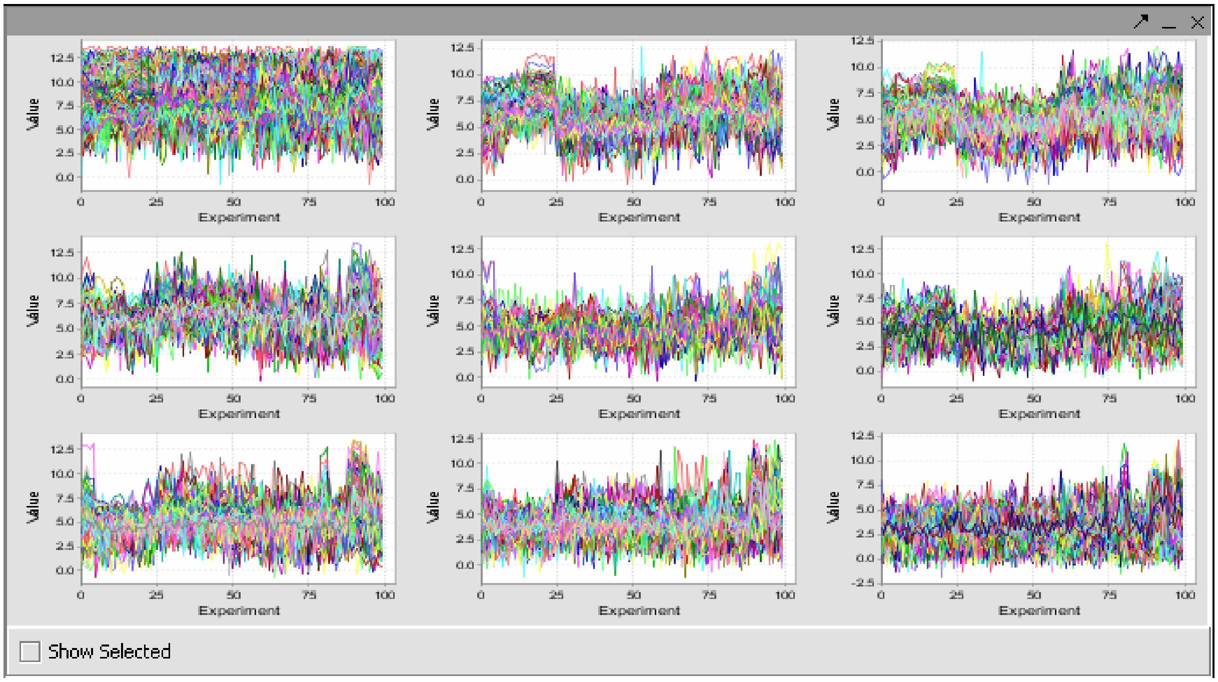


Figure 4-9 SOM Viewer

7. The genes represented in any given graph can be selected and returned to the Markers component for further analysis.
8. The result is also placed in the **Project Folders** component of the GUI beneath its parent data set. In this figure, both a hierarchical clustering dataset and a SOM result are present. Either can be viewed simply by selecting it (Figure 4-10) here.

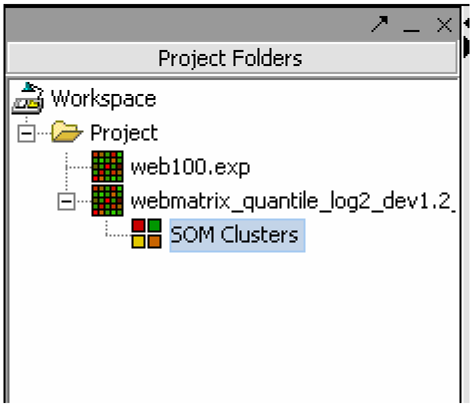


Figure 4-10 Project Folder with results nodes

ARACNE

Introduction

ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks) can be used to infer regulatory interactions within a set of microarray data. Its use has been described in detail in reference Margolin et al. 2006, #1.

The ARACNE algorithm is based on the calculation of mutual information (MI). It has been designed to overcome a number of problems with other methods, for example by allowing calculation on continuous-valued data rather than requiring discretized data, and needing no assumptions about the underlying network topology. The method is not without potential limitations, and the user should consult the references for more information on the usefulness of the algorithm in any particular type of investigation.

ARACNE calculates the mutual information between specified pairs of markers across a set of multiple microarray gene-expression experiments. In an optional second step, an information theoretic property known as the Data Processing Inequality (DPI) (Margolin et al. 2006, #2) is used to attempt to remove indirect interactions, that is, those mediated by another marker. The algorithm in principal could be used on any type of interaction data, not just gene expression. In geWorkbench however only gene expression data may currently be submitted to the algorithm. The output of ARACNE is an adjacency matrix, showing the strength of interaction for each calculated pair of markers.

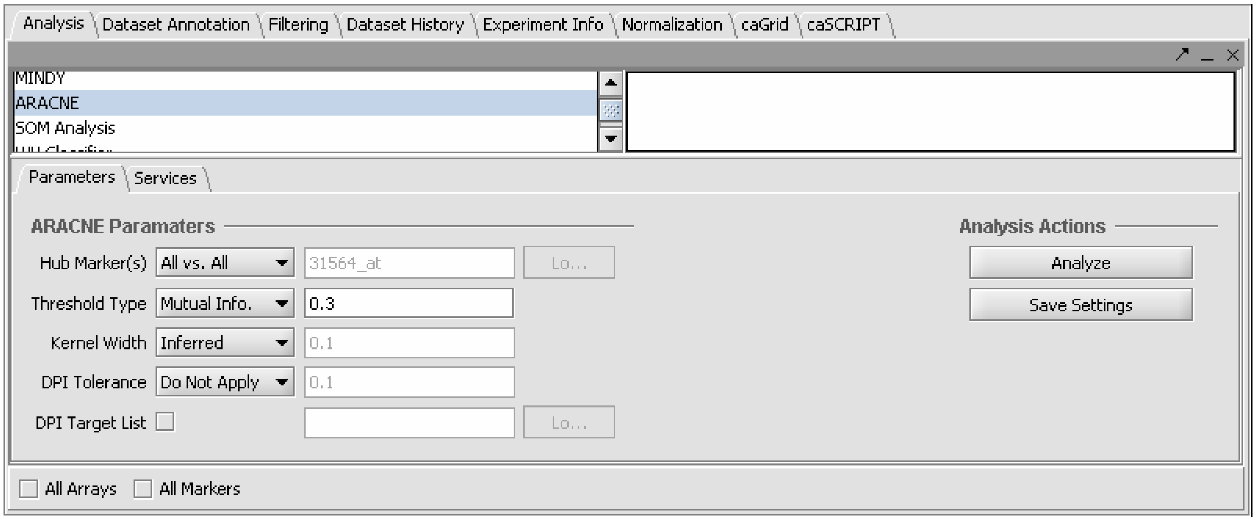
Prerequisites:

There should be at least 100 microarrays in the dataset to allow reliable calculation of the mutual information. However, please note than on such large datasets, only calculation of the mutual information of a particular gene marker with the rest of the dataset should be undertaken on a desktop-class machine. An all-against-all calculation generally requires use of a cluster computer, potentially via the grid service facility if such a service becomes available. Otherwise, ARACNE can be obtained as a stand-alone program for use on a local computational cluster.

The data used should show a significant dynamic range, for example through sampling multiple phenotypes, or through use of experimental perturbations. Uninformative genes, such as those with low mean expression, should likely be filtered out before running ARACNE.

Understanding the Parameters

There are several choices that can be made as to how ARACNE will be run on a given dataset. Here we will outline what these choices are, and provide an introduction on how to choose appropriate parameters for your own particular case. The figure below (Figure 4-11) shows the ARACNE component within geWorkbench and the parameters which can be set.



ARACNE Paramaters

Hub Marker(s): All vs. All — 31564_at

Threshold Type: Mutual Info. — 0.3

Kernel Width: Inferred — 0.1

DPI Tolerance: Do Not Apply — 0.1

DPI Target List □

□ All Arrays □ All Markers

Figure 4-11 The ARACNE component GUI

Hub Markers – Using a hub gene to limit calculations

An ARACNE run can either compute the MI values of every (active) marker against every other, or it can use a list of one or more hub genes to limit the number of calculations required. For large datasets, the all-against-all calculation is not feasible on an local desktop machine, and instead a computational cluster should be used if available as a grid service. If hub genes are used, the MI values are only calculated between each hub gene and all others (or all other markers activated in the Markers component).

- Options:
  - **All vs. All** – Compute the mutual information of each marker against all others (or all markers activated in the Markers component against each other).
  - **List** – Compute the mutual information of each marker in the list against all others (or all those activated in the Markers component).

Threshold Type – P-value or Mutual Information Score

The threshold value above which potentially interacting pairs will be reported can be set either as a p-value or directly as a mutual information score. The mutual information score is more useful when reevaluating an already calculated adjacency matrix, an option present in the standalone version of ARACNE. Margolin et al. 2006, #1 gives a suggested method of choosing a reasonable p-value:

> Divide ..” the desired number of false-positives (generally a small integer) by the number of tests performed, calculated as the number of distinct probe pairs. For example, a

threshold of 1e-7 will lead to about five expected false-positives for a data set with around 10,000 probes, because 10,000 choose 2 (i.e., about 5e7) probe pairs are tested".

- Options:
  - **Mutual Info** – use a specified mutual information value, ranging from from 0 to 1.
  - **P-Value** – specify a p-value. This will be converted internally to a mutual information threshold as described above and in Margolin et al. 2006, #1.

Kernel Width – a parameter for the Mutual Information calculation

The parameter used in the mutual information calculation is the kernel width of the Gaussian operator. ARACNE will calculate a default value based on the sample size. The user may also specify a kernel width; methods to calculate an optimal value are given in Margolin et al. 2006, #1. However, for most uses the default value should be adequate, as the algorithms has been shown to be robust with respect to this parameter.

- Options:
  - **Inferred** – accept the default value
  - **Specify** – enter an explicit value.

DPI Tolerance – Removing indirect interactions using DPI

As described in Margolin et al. 2006, #1,

> "Many statistical dependencies between gene expression profiles arise from cascades of transcriptional interactions that correlate the expressions of many genes that do not interact directly. ARACNE provides an option to eliminate interactions that are likely to be indirect by applying …the DPI (described in detail in Margolin et al. 2006, #2). The DPI requires accurate estimation of Mutual Information (MI) ranks; as MI values cannot be estimated exactly with finite data, a tolerance is used to compensate for errors in the estimate that might affect these ranks. Empirically, values between 0 (no tolerance) and 0.15 (15%) tolerance should be used, as larger values tend to cause high false-positive rates".

- Options:
  - **Do Not Apply** – do not run the DPI calculation
  - **Apply** – run the DPI calculation with the specified tolerance to remove potential indirect interactions.

DPI Target List (checkbox) – Generating a network of transcription factors

If you wish to reconstruct a network only involving transcription factors, you can include a list of such genes (which have been annotated as transcription factors (TFs)) whose interactions are not to be eliminated by DPI in favor of interactions consisting of two non-TFs. “This partially alleviates the problem associated with highly correlated non-interacting genes, such as those involved in stable complex formation, which violate some of the assumptions required for application of the DPI. This feature is described in greater detail in the online Supplementary Manual” (Margonlin et al. 2006, #1) (http://amdec-bioinfo.cu-genome.org/html/ARACNE.htm).

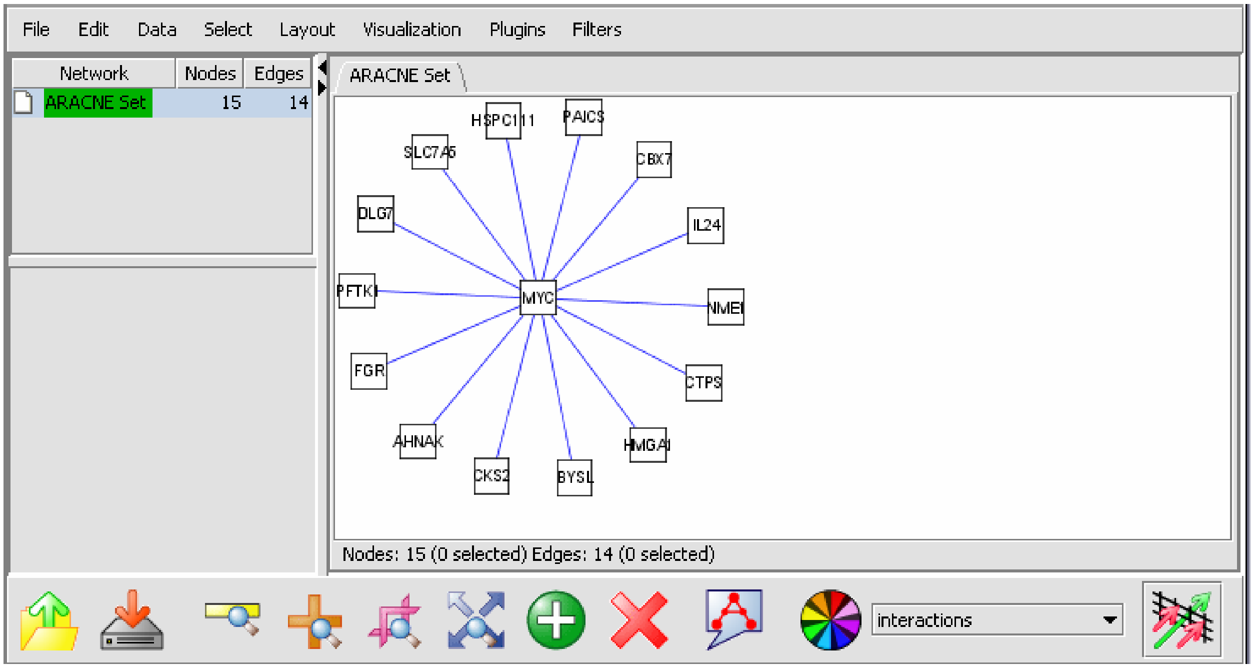
Running an ARACNE Calculation

Example

1. A microarray dataset must be loaded in the **Projects** component. Review the Prerequisites section above for details.
2. The **ARACNE** component can be found in the **Analysis** section (tab).
3. Set the parameters depending on the desired type of run as described above in “Understanding the Parameters”
4. If a remote ARACNE grid service is to be used, go to the **Services** tab in the **ARACNE** component and chose the desired service, e.g. as already described above for Hierarchical Clustering.
5. Push the **Analyze** button. When complete, the results will be displayed within geWorkbench in the **Cytoscape** component.

Viewing the Results

The figure below (Figure 4-12) shows an example of viewing the results of an ARACNE run using a single hub gene. The results are displayed in the **Cytoscape** component. This component has a number of options for controlling how the network graph will be displayed. It is an external component to geWorkbench and full documentation on its use can be found at http://www.cytoscape.org/.



Network	Nodes	Edges
ARACNE Set	15	14

Nodes: 15 (0 selected) Edges: 14 (0 selected)

Figure 4-12 Display of ARACNE-generated adjacency matrix in Cytoscape

References

(1) Reverse engineering cellular networks. Adam A Margolin' Kai Wang, Wei Keat Lim, Manjunath Kustagi, Ilya Nemenman & Andrea Califano. (2006) Nature Protocols **1**, pp 662-671

(2) ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. Margolin AA, Nemenman I, Basso K, Wiggins C, Stolovitzky G, Dalla Favera R, Califano A. (2006) BMC Bioinformatics 7,S7.

Chapter 5 Querying caARRAY

This chapter describes how geWorkbench can query remote instances of a caARRAY database. caARRAY uses a MAGE-OM compliant API which allows searching on a number of common annotation data fields, such as species, array type and tissue type.

This chapter covers the single topic:

- Searching caARRAY using MAGE annotations

Searching caARRAY using MAGE annotations

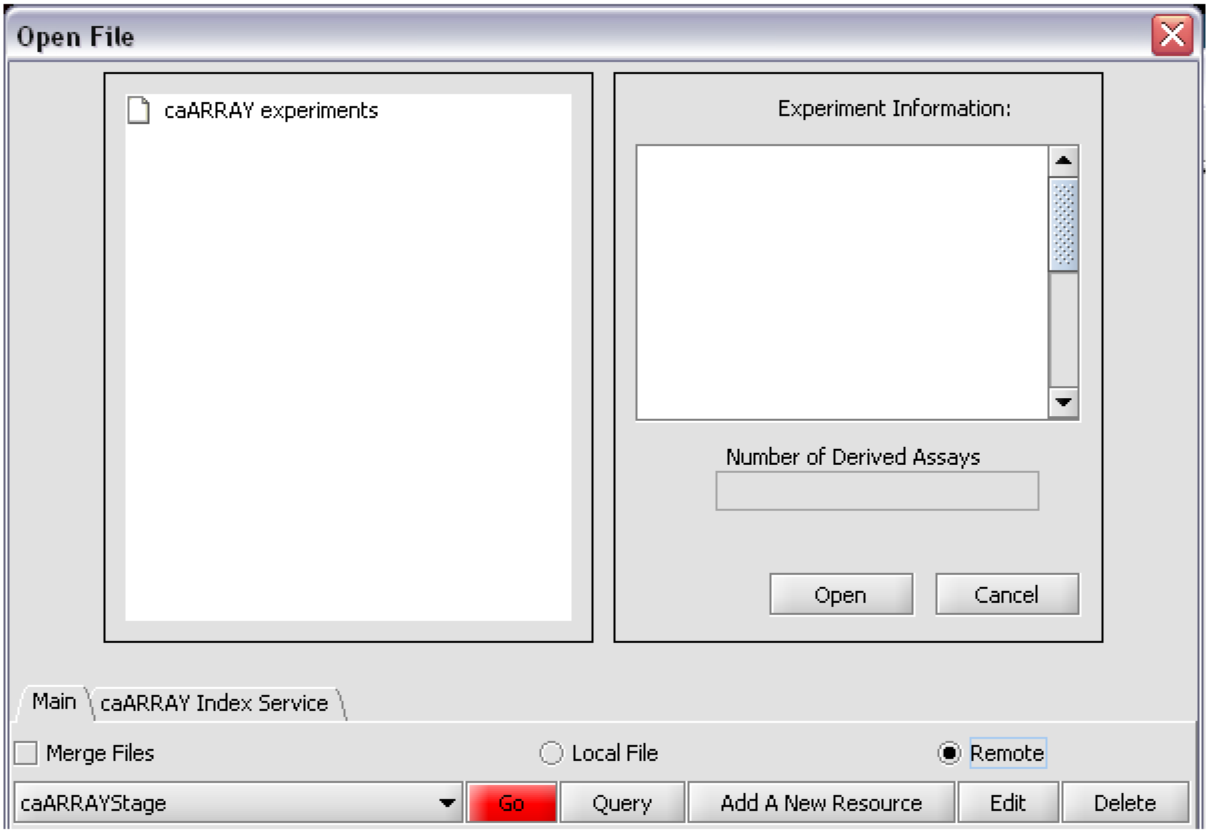
caARRAY is a microarray gene expression repository developed by the NCICB which supports storing and querying of annotated datasets. The annotations conform to the MAGE (Microarray Gene Expression) model. It should be noted that actual datasets may be only partially or sparsely annotated, but the model has the capability to store, in a rigorous fashion, almost any type of information about an experiment.

In the current implementation, geWorkbench can query against four types of annotations supported by caARRAY:

- Tissue type (available choices are displayed)
- Chip Platform (e.g. Affymetrix, Agilent etc.)
- Organism
- Principal Investigator

The following example illustrates constructing a query against caARRAY.

1. Create a new project as described previously.
2. Right-click on the **Project** entry and select **Open File(s)**.
3. Click the **Remote** radio button. This will cause the Open File popup to switch to the remote file interface (Figure 5-1).

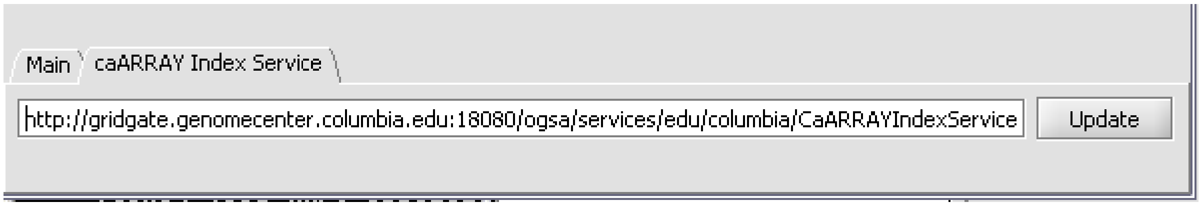


● Remote

caARRAYStage

Figure 5-1 The Remote Open File interface

You can inspect the **caARRAY Index Service** tab (Figure 5-2). As more services become available, the desired service can be entered here:



http://gridgate.genomecenter.columbia.edu:18080/ogsa/services/edu/columbia/CaARRAYIndexService

Figure 5-2 Inspecting the caARRAY Index Service entry

Clicking on the red **Go** button would retrieve all available experiments, Instead, we will construct a query.

4. Click on **Query** (see Figure 5-1 above) to build a MAGE keyword search. The query builder appears (Figure 5-3).

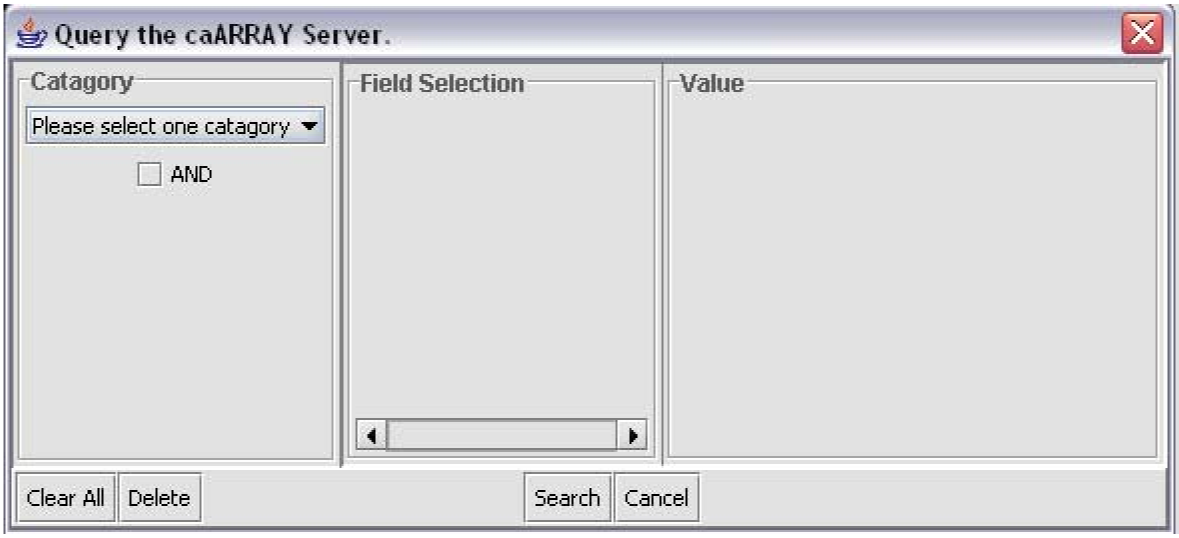
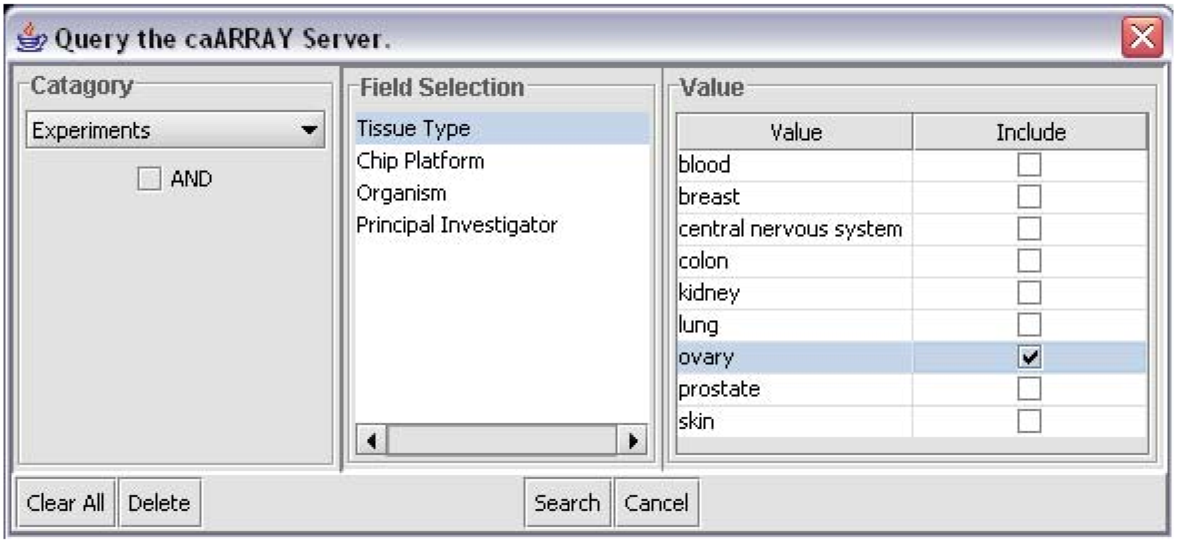


Figure 5-3 The caARRAY MAGE query interface

5. Under Category, select **Experiments** (Figure 5-4). The available search field types will be displayed. Here we will search on **Tissue Type**. Highlighting this field shows available tissue types.

6. Here we have selected the tissue type **ovary**.



Catagory: Experiments

□ AND

Field Selection:

- Tissue Type
- Chip Platform
- Organism
- Principal Investigator

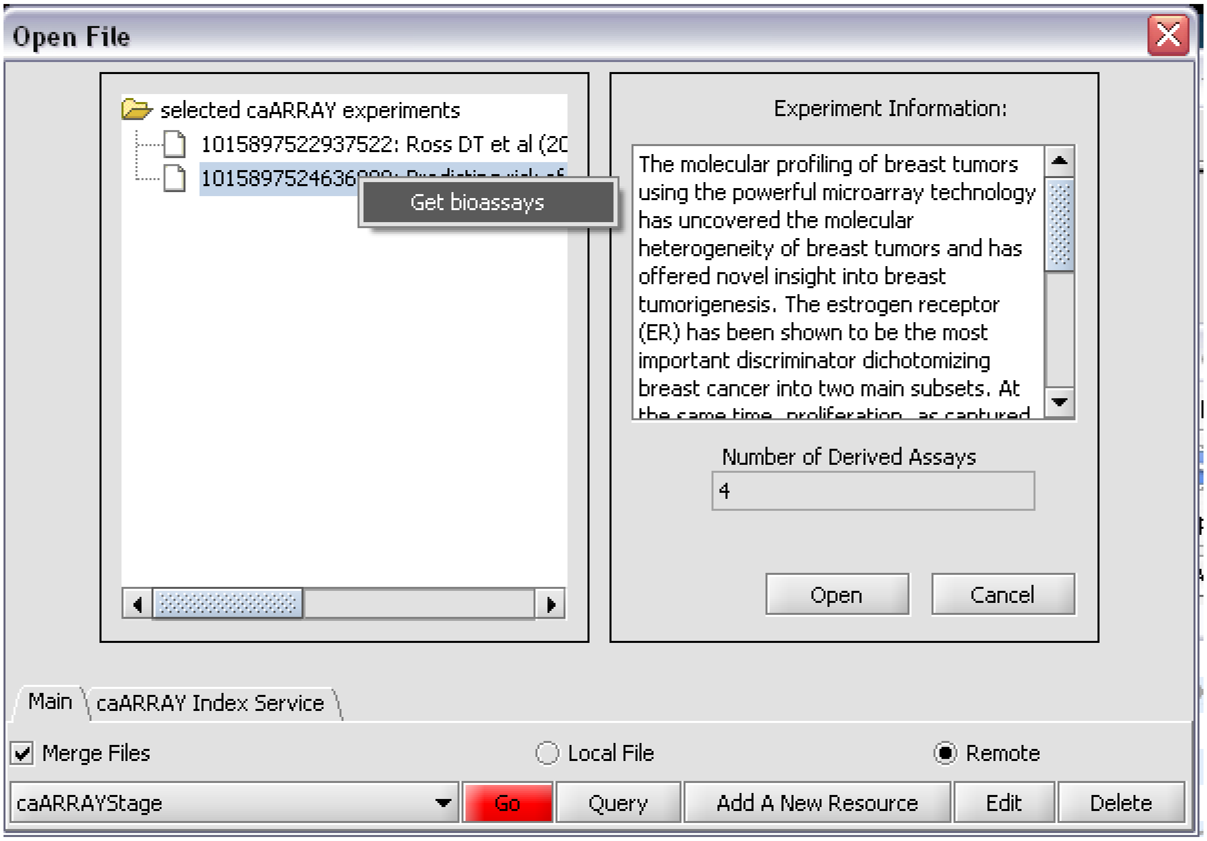
Value	Include
blood	□
breast	□
central nervous system	□
colon	□
kidney	□
lung	□
ovary	☑
prostate	□
skin	□

Figure 5-4 Constructing a new caARRAY query

Notes on query building –

- Use of the **AND** checkbox – As currently implemented in caARRAY, **AND** queries can only include items from different fields as defined above. That is, you could search for a **Tissue Type AND** a **Principal Investigator**". If the **AND** box is not checked, only the currently visible field will be used in the query.
- Implied **AND** within a field – If e.g. in Tissue Type you were to select two different tissues, the query would be for type A **AND** type B. This is a limitation of the current caARRAY implementation. No **OR** functionality is yet available.

7. Click **Search**. (see Figure 5-4 above) Experiments matching the search term are returned (Figure 5-5).
8. Selecting an experiment will display information about it in the boxes at right, as here we have selected the second entry.
9. After selecting an entry, right click on it – the **Get bioassays** button appears.



The molecular profiling of breast tumors using the powerful microarray technology has uncovered the molecular heterogeneity of breast tumors and has offered novel insight into breast tumorigenesis. The estrogen receptor (ER) has been shown to be the most important discriminator dichotomizing breast cancer into two main subsets. At the same time, proliferation, as captured

Number of Derived Assays: 4

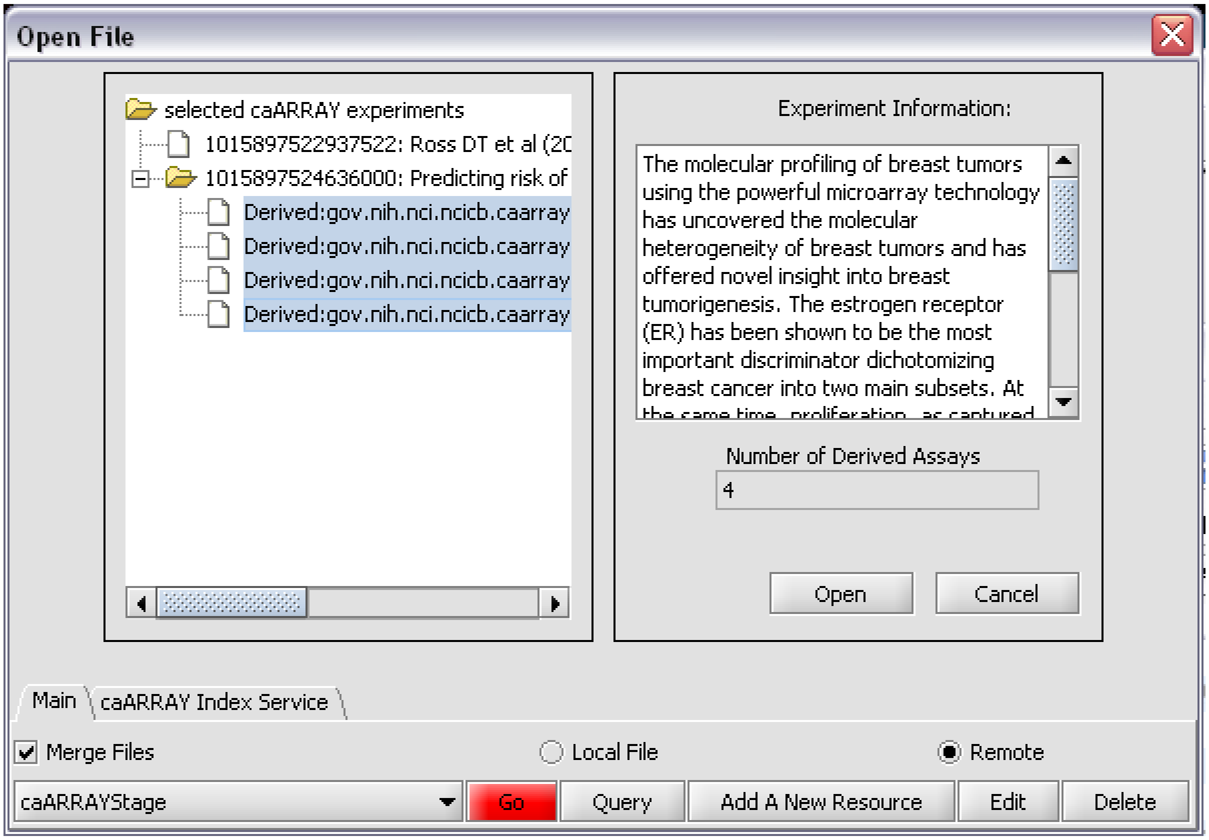
☑ Merge Files ○ Local File ● Remote

caARRAYStage ▾

Figure 5-5 Retrieving the list of bioassays for an experiment

10. Clicking on **Get bioassays** returns the available bioassays for this experiment. (This hidden step is the most easily overlooked one in geWorkbench). The

returned bioassays are shown in Figure 5-6.



selected caARRAY experiments
1015897522937522: Ross DT et al (2C
1015897524636000: Predicting risk of
Derived:gov.nih.nci.ncicb.caarray
Derived:gov.nih.nci.ncicb.caarray
Derived:gov.nih.nci.ncicb.caarray
Derived:gov.nih.nci.ncicb.caarray

Experiment Information:

The molecular profiling of breast tumors using the powerful microarray technology has uncovered the molecular heterogeneity of breast tumors and has offered novel insight into breast tumorigenesis. The estrogen receptor (ER) has been shown to be the most important discriminator dichotomizing breast cancer into two main subsets. At the same time, proliferation, as captured

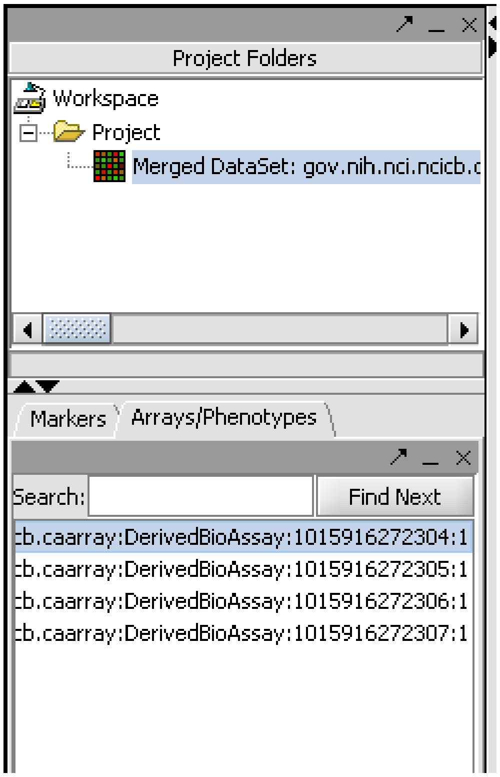
Number of Derived Assays
4

☑ Merge Files ○ Local File ● Remote

caARRAYStage

Figure 5-6 View of retrieved bioassays

11. Click on the **Merge Files** checkbox at lower left (Figure 5-6) so that the four files will be merged into a single geWorkbench dataset after they are downloaded. Highlight the desired bioassays.
12. Click **Open** (located in the **Experiment Information** box). The datasets are downloaded, merged, and inserted into the current **Project** (Figure 5-7). In the **Arrays/Phenotypes** component below, the four individual arrays can be seen.



cb.caarray:DerivedBioAssay:1015916272304:1
cb.caarray:DerivedBioAssay:1015916272305:1
cb.caarray:DerivedBioAssay:1015916272306:1
cb.caarray:DerivedBioAssay:1015916272307:1

Figure 5-7 The merged dataset retrieved from caARRAY as displayed in the Project Folder

13. The merged dataset can be renamed if desired by right-clicking on it and selecting **Rename**.
14. Did you forget to check the **Merge** checkbox before download? You can merge the files after download by selecting menu item **File > Merge Datasets**.

Chapter 6 Using caSCRIPT to automate actions

This chapter describes how caSCRIPT, a scripting language developed for geWorkbench, can be used to automate common or complex tasks in geWorkbench. The caSCRIPT environment includes a visual editor with access to the available methods and variables.

This chapter covers the single topic:

- Using caSCRIPT to automate tasks

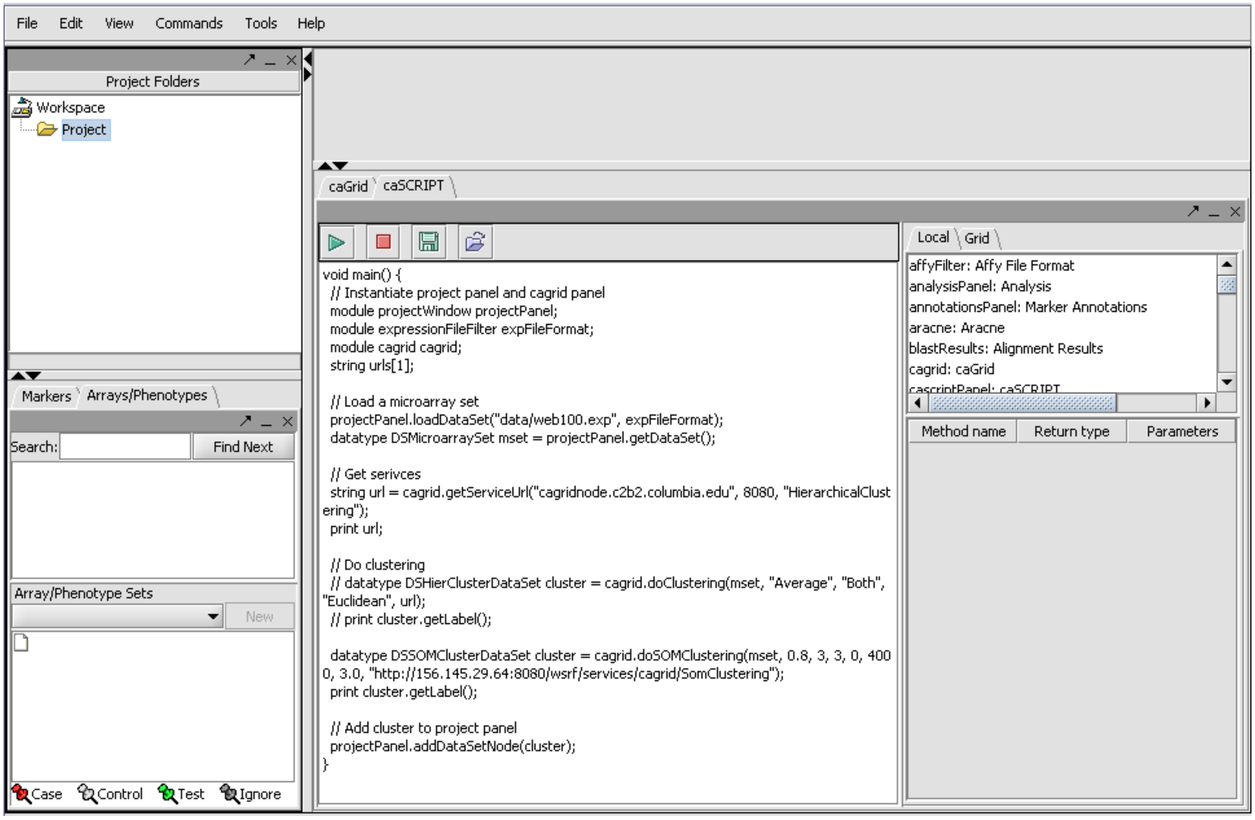
Using caSCRIPT to automate tasks

geWorkbench has a built-in scripting language, caSCRIPT. This language is similar to Java. The actual language is described in a separate technical document. It provides direct access to all of the modules in geWorkbench. Scripting allows any sequence of steps in a workflow to be automated, and allows them to be repeated as desired. The example presented here will illustrate how caSCRIPT can be used to execute a caGRID-based SOM calculation. The basics of running a SOM calculation on a caGRID node have already been covered above in Chapter 4.

The caSCRIPT component, shown below in Figure 6-1, contains a default demonstration script. It executes SOM clustering. A line to run Hierarchical Clustering has been commented out, but can be included by simply removing the comment symbol (//). For this example the user can paste in a script which will perform hierarchical clustering:

```
void main() {
  // Instantiate project panel and cagrid panel
  module projectWindow projectPanel;
  module expressionFileFilter expFileFormat;
  module cagrid cagrid;
  string urls[1];

  // Load a microarray set
  projectPanel.loadDataSet("data/web100.exp", expFileFormat);
  datatype DSMicroarraySet mset = projectPanel.getDataSet();

  // Get serivces
  string url =
cagrid.getServiceUrl("cagridnode.c2b2.columbia.edu", 8080,
"HierarchicalClustering");
  print url;
  // Do clustering
  datatype DSHierClusterDataSet cluster =
cagrid.doClustering(mset, "Total", "Both", "Pearson", url);
   print cluster.getLabel();

  // Add cluster to project panel
  projectPanel.addDataSetNode(cluster);
}
```

The caSCRIPT component contains three separate areas (Figure 6-1) – the script display at left, a list of available methods at top-right, and an area to display the details of any selected method at bottom right.   These aid in constructing new scripts (not covered in this manual).

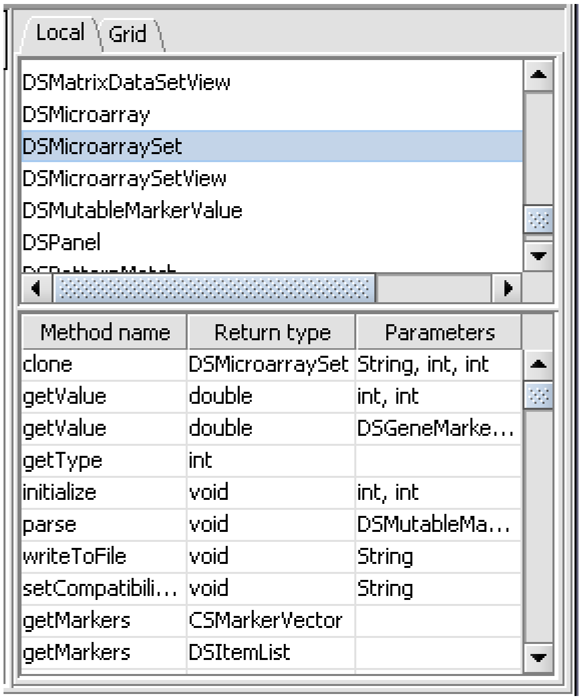


```
void main() {
  // Instantiate project panel and cagrid panel
  module projectWindow projectPanel;
  module expressionFileFilter expFileFormat;
  module cagrid cagrid;
  string urls[1];

  // Load a microarray set
  projectPanel.loadDataSet("data/web100.exp", expFileFormat);
  datatype DSMicroarraySet mset = projectPanel.getDataSet();

  // Get serivces
  string url = cagrid.getServiceUrl("cagridnode.c2b2.columbia.edu", 8080, "HierarchicalClust
ering");
  print url;

  // Do clustering
  // datatype DSHierClusterDataSet cluster = cagrid.doClustering(mset, "Average", "Both",
"Euclidean", url);
  // print cluster.getLabel();

  datatype DSSOMClusterDataSet cluster = cagrid.doSOMClustering(mset, 0.8, 3, 3, 0, 400
0, 3.0, "http://156.145.29.64:8080/wsrf/services/cagrid/SomClustering");
  print cluster.getLabel();

  // Add cluster to project panel
  projectPanel.addDataSetNode(cluster);
}
```

affyFilter: Affy File Format
analysisPanel: Analysis
annotationsPanel: Marker Annotations
aracne: Aracne
blastResults: Alignment Results
cagrid: caGrid
cascriptPanel: caSCRIPT

Figure 6-1 The caSCRIPT interface

Figure 6-2 below shows an example of listing a local method, which displays its properties just below it.

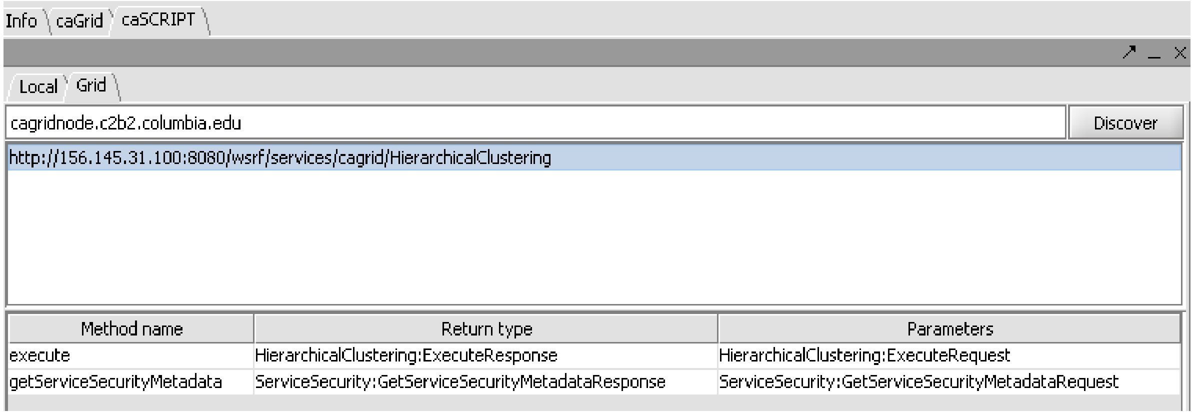


DSMatrixDataSetView
DSMicroarray
DSMicroarraySet
DSMicroarraySetView
DSMutableMarkerValue
DSPanel

Method name	Return type	Parameters
clone	DSMicroarraySet	String, int, int
getValue	double	int, int
getValue	double	DSGeneMarke...
getType	int	
initialize	void	int, int
parse	void	DSMutableMa...
writeToFile	void	String
setCompatibili...	void	String
getMarkers	CSMarkerVector	
getMarkers	DSItemList	

Figure 6-2 caSCRIPT methods selector

The **Grid** tab at right on the caSCRIPT component (Figure 6-3) reveals an interface for changing the **caGRID Index Service** and for discovering the analytical services the chosen node offers. The figure below shows that after the **Discover** button has been pushed, the **Hierarchical Clustering** service is offered. This information could be used in constructing a new script. When a discovered service is selected, its details are displayed in the area below.



cagridnode.c2b2.columbia.edu

http://156.145.31.100:8080/wsrf/services/cagrid/HierarchicalClustering

Method name	Return type	Parameters
execute	HierarchicalClustering:ExecuteResponse	HierarchicalClustering:ExecuteRequest
getServiceSecurityMetadata	ServiceSecurity:GetServiceSecurityMetadataResponse	ServiceSecurity:GetServiceSecurityMetadataRequest

Figure 6-3 Discovering available services

The **caSCRIPT** component has four main controls, shown below in Figure 6-4. The are from left:

Play – execute the current script.

Stop – stop execution of the script.

Save – save the current script to disk.

Open - opens a file browser to locate a saved script.

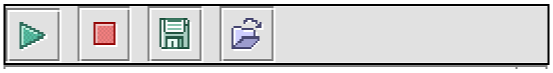
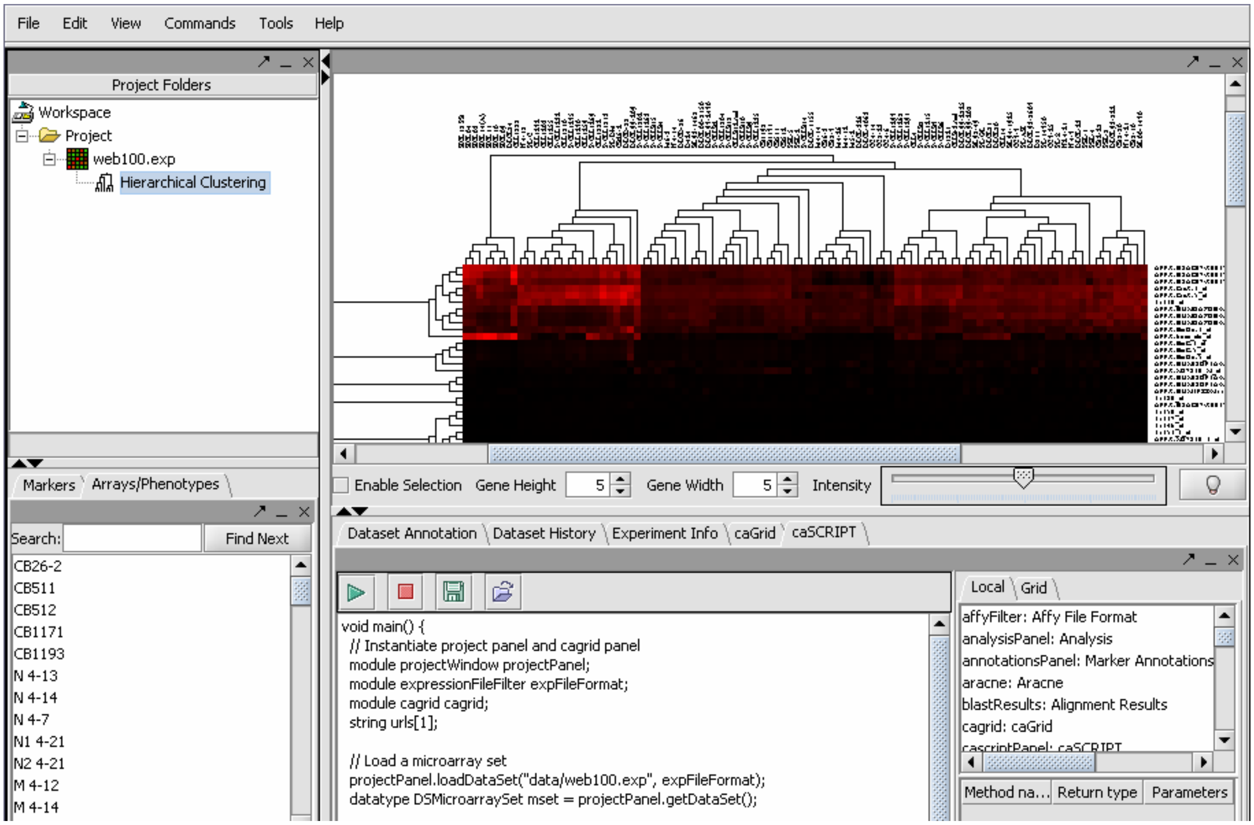


Figure 6-4 caSCRIPT controls

Example:

1. To run the hierarchical clustering example, select and copy the test script shown above into the **caSCRIPT** component.
2. Press the left-arrow shaped **Play** button on the **caSCRIPT** component. The task will be executed on the remote server and the results returned to the **Dendrogram** component, as shown in Figure 6-5 below.



CB26-2
CB511
CB512
CB1171
CB1193
N 4-13
N 4-14
N 4-7
N1 4-21
N2 4-21
M 4-12
M 4-14

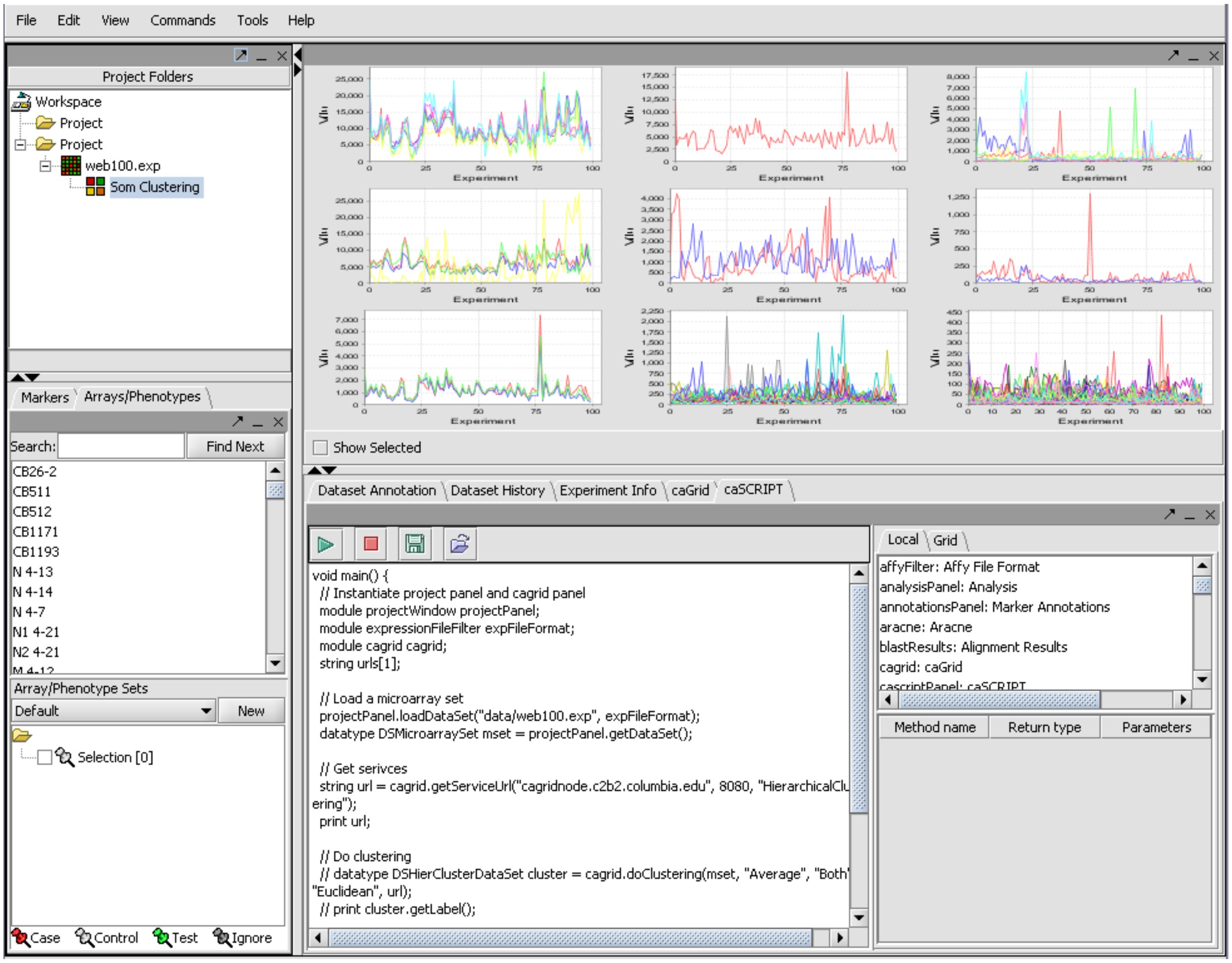
☐ Enable Selection Gene Height 5 Gene Width 5 Intensity

```
void main() {
  // Instantiate project panel and cagrid panel
  module projectWindow projectPanel;
  module expressionFileFilter expFileFormat;
  module cagrid cagrid;
  string urls[1];

  // Load a microarray set
  projectPanel.loadDataSet("data/web100.exp", expFileFormat);
  datatype DSMicroarraySet mset = projectPanel.getDataSet();
```

affyFilter: Affy File Format
analysisPanel: Analysis
annotationsPanel: Marker Annotations
aracne: Aracne
blastResults: Alignment Results
cagrid: caGrid
cascriptPanel: caSCRIPT

Figure 6-5 The Dendrogram component displaying results of hierarchical clustering executed using caSCRIPT

The result of running the default script, which executes SOM clustering, is shown in Figure 6-6 below:



CB26-2
CB511
CB512
CB1171
CB1193
N 4-13
N 4-14
N 4-7
N1 4-21
N2 4-21
M 4-12

```
void main() {
  // Instantiate project panel and cagrid panel
  module projectWindow projectPanel;
  module expressionFileFilter expFileFormat;
  module cagrid cagrid;
  string urls[1];

  // Load a microarray set
  projectPanel.loadDataSet("data/web100.exp", expFileFormat);
  datatype DSMicroarraySet mset = projectPanel.getDataSet();

  // Get serivces
  string url = cagrid.getServiceUrl("cagridnode.c2b2.columbia.edu", 8080, "HierarchicalClu
ering");
  print url;

  // Do clustering
  // datatype DSHierClusterDataSet cluster = cagrid.doClustering(mset, "Average", "Both"
"Euclidean", url);
  // print cluster.getLabel();
```

affyFilter: Affy File Format
analysisPanel: Analysis
annotationsPanel: Marker Annotations
aracne: Aracne
blastResults: Alignment Results
cagrid: caGrid
cascriptPanel: caSCRIPT

Figure 6-6 Results of SOM clustering example using caSCRIPT

Chapter 7 Error Messages/Indicators and Problem Resolutions

Why is the desired caGRID service not available?

Until services are installed in an official caBIG index service, their availability may vary.

After running a large local calculation, my Windows computer seems slow.

There are apparently some problems, at least with Windows XP, if the operating system is pushed into swapping – that is, some of the contents of memory are written to disk. Even after geWorkbench has been exited, the slow behavior may persist. In extreme cases, simply reboot the computer. We recommend running geWorkbench in a computer with at least 1 GB of memory, while 2 GB or more will greatly increase the size of calculations possible.

How do I increase the memory allocated to geWorkbench?

There is a file in the geWorkbench root directory called UILauncher.lax. There is a line there which specifies the Java heap size:

```
lax.nl.java.option.java.heap.size.max=640678989
```

Here it is shown set to about 640 MB. You can experiment with increasing this, subject to the amount of memory in your machine and demands on it from other applications.

(Note – this method applies to the packaged distribution version of geWorkbench)

Where else can I look for help?

Please see the main geWorkbench website at http://www.geworkbench.org/. Of particular interest will be the following sections:

1. FAQs

2. Known Issues

3. Tutorials

Appendix A References

Scientific Publications

1. Reverse engineering cellular networks. Adam A Margolin, Kai Wang, Wei Keat Lim, Manjunath Kustagi, Ilya Nemenman & Andrea Califano. (2006) Nature Protocols 1, pp 662-671
2. ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. Margolin AA, Nemenman I, Basso K, Wiggins C, Stolovitzky G, Dalla Favera R, Califano A. (2006) BMC Bioinformatics 7,S7.

Technical Manuals/Articles

1. National Cancer Institute. "caCORE 2.0 Technical Guide", ftp://ftp1.nci.nih.gov/pub/cacore/caCORE2.0_Tech_Guide.pdf
2. Java Programming: http://java.sun.com/learning/new2java/index.html
3. Extensible Markup Language: http://www.w3.org/TR/REC-xml/
4. XML Metadata Interchange: http://www.omg.org/technology/documents/formal/xmi.htm

caBIG Material

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

caCORE Material

1. **caCORE:** http://ncicb.nci.nih.gov/core
2. **caBIO:** http://ncicb.nci.nih.gov/core/caBIO
3. **caDSR:** http://ncicb.nci.nih.gov/core/caDSR
4. **EVS:** http://ncicb.nci.nih.gov/core/EVS
5. **CSM:** http://ncicb.nci.nih.gov/core/CSM

Appendix B Glossary

Following is a list of terms and their definitions.

Term	***Definition***
API	Application Programming Interface
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
caMOD	Cancer Models Database
CDE	Common Data Element
CGAP	Cancer Genome Anatomy Project
CMAP	Cancer Molecular Analysis Project
CVS	Concurrent Versions System
EVS	Enterprise Vocabulary Services
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
JAR	Java Archive
Javadoc	Tool for generating API documentation in HTML format from doc comments in source code (http://java.sun.com/j2se/javadoc/)
MAGE	MicroArray Gene Expression
MAGE-OM	MicroArray Gene Expression - Object Model
MGED	Microarray Gene Expression Data
MO	MGED Ontology
NCI	National Cancer Institute
NCICB	National Cancer Institute Center for Bioinformatics
SDK	Software Development Kit
SQL	Structured Query Language
UI	User Interface
URL	Uniform Resource Locators

INDEX