CLC **Genomics** Server

Administrator Manual

Administrator Manual for

CLC Genomics Server 4.5

Windows, Mac OS X and Linux

August 8, 2012

This software is for research purposes only.

CLC bio
Finlandsgade 10-12
DK-8200 Aarhus N
Denmark

Quick installation guide

The following describes briefly the steps needed to set up a *CLC Genomics Server 4.5* with pointers to more detailed explanation of each step. If you are going to set up execution nodes as well, please read section 7 first.

1. Download and run the server installer. As part of the installation, choose to start the server (section 2.1).
2. Run the license download script that is part of the installation (section 2.5).
3. The script will automatically download a license file and place it in the server installation directory under `licenses`.
4. Restart the server (section 2.6).
5. Log in to the server using a web browser (note that the default port is 7777) with username **root** and password **default** (section 3).
6. Change the root password (section 4.1).
7. Configure authentication mechanism and optionally set up users and groups (section 4.2).
8. Add data locations (section 3.2).
9. Download and install plug-ins *in the Workbench* needed for the Workbench to contact the server (section 2.7).
10. Check your server set-up using the **Check set-up** link in the upper right corner as described in section B.1.
11. Your server is now ready for use.

Contents

Chapter 1

Introduction

The *CLC Server* is the central part of CLC bio's enterprise solutions. You can see an overview of the server solution in figure 1.1).

For documentation on the customization and integration, please see the developer kit for the server at `http://www.clcdeveloper.com`.

The *CLC Genomics Server* is shipped with the following tools and analyses that can be started from the *CLC Genomics Workbench*:

- Import
- Trimming
- Sequencing Data QC Report
- Map reads to reference
- De novo assembly
- All tools for re-sequencing analysis
- Download genomes
- All tools for track management and filtering
- Create detailed mapping report
- ChIP-Seq
- RNA-Seq
- Extract and count small RNA
- Annotate small RNA
- Process tagged sequences (de-multiplexing)
- External applications (integrating with 3rd party programs on the server)
- BLAST (creation of databases, download of databases and running BLAST against local data or NCBI)

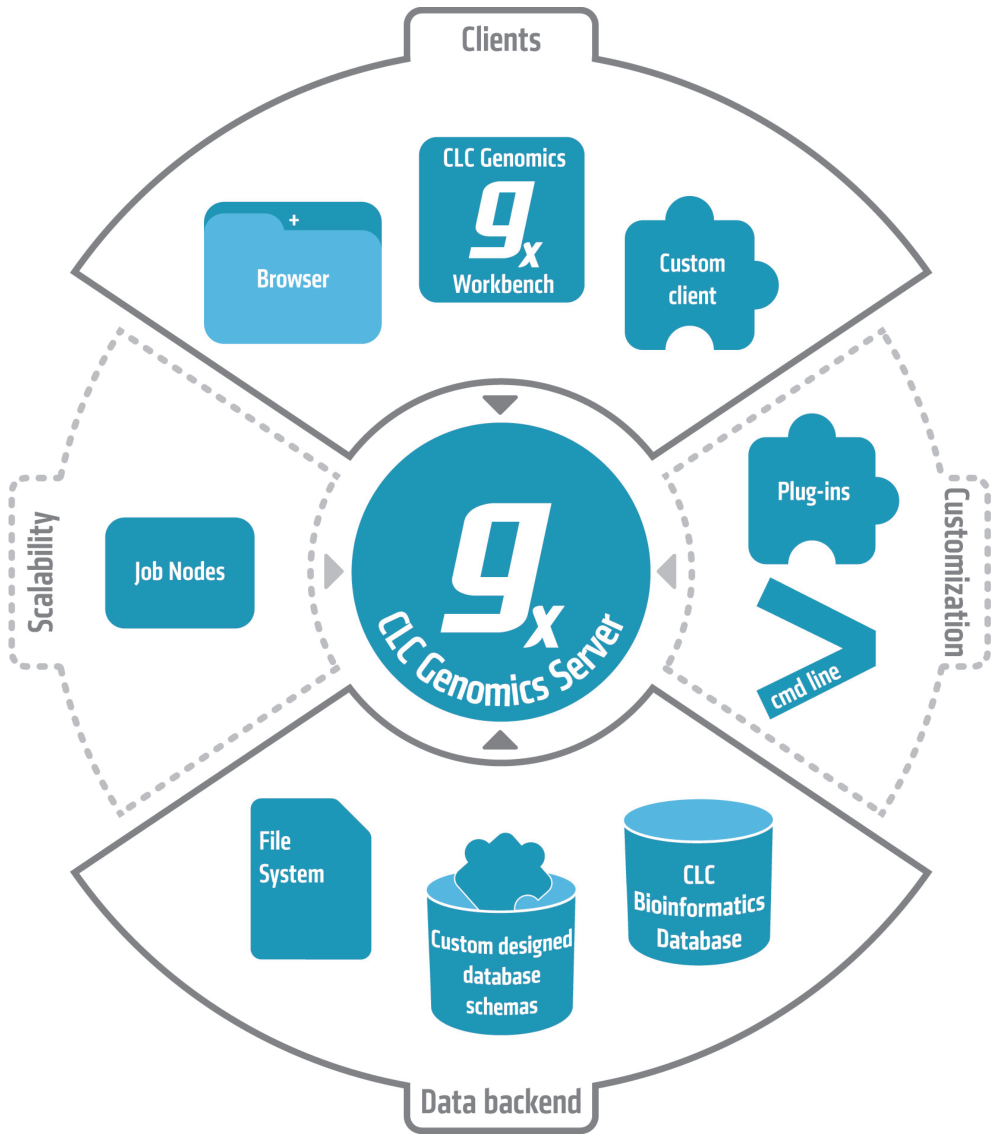


Figure 1.1: *An overview of the server solution from CLC bio. Note that not all features are included with all license models.*

- Find primer binding sites
- Annotate secondary peaks (Sanger data)
- Extract sequences
- Additional alignments (plug-in)

The functionality of the *CLC Genomics Server* can be extended by installation of Server plug-ins. The available plug-ins can be found at http://www.clcbio.com/server_plugins.

1.1 System requirements

The system requirements of *CLC Genomics Server* are:

- Windows XP, Windows Vista, or Windows 7, Windows Server 2003 or Windows Server 2008
- Mac OS X 10.6 or later. Intel CPU required. However, Mac OS X 10.5.8 is supported on 64-bit Intel systems.
- Linux: RedHat 5 or later. SuSE 10 or later.
- Intel or AMD CPU required
- Memory: The numbers below give minimum and recommended amounts for systems running mapping and analysis tasks. The requirements suggested are based on the genome size.
  - **E. coli K12 (4.6 megabases)**
    - Minimum: 2Gb RAM
    - Recommended: 4Gb RAM
  - **C. elegans (100 megabases)** and **Arabidopsis thaliana (120 megabases)**
    - Minimum: 4Gb RAM
    - Recommended: 8Gb RAM
  - **Zebrafish (1.5 gigabases)**
    - Minimum: 8Gb RAM
    - Recommended: 16Gb RAM
  - **Human (3.2 gigabases)** and **Mouse (2.7 gigabases)**
    - Minimum: 24Gb RAM
    - Recommended: 48Gb RAM
- **Special requirements for de novo assembly**. De novo assembly may need more memory than stated above - this depends both on the number of reads and the complexity and size of the genome. See `http://www.clcbio.com/white-paper` for examples of the memory usage of various data sets.
- A 64 bit computer and operating system is necessary if you wish to utilize more than 2GB RAM.

1.2 Licensing

There are three kinds of licenses needed for running the *CLC Genomics Server*:

- A license for the server. This is needed for running the server. The license will allow a certain number of sessions (i.e. number of log-ins from e.g. the web interface or the workbench). The number of sessions is part of the agreement with CLC bio when you purchase a license.
- A special license if you are running the server on a grid system (this is explained in detail in section 7.2.3)
- A license for the workbench. The Workbench is needed to start analyses on the server and view the results. Find the user manuals and deployment manual for the Workbenches at `http://www.clcbio.com/usermanuals`

The following chapter on installation will give you more information about how to obtain and deploy the license for the server.

Chapter 2

Installation

2.1 Installing and running the Server

Getting the *CLC Genomics Server* software installed and running involves, at minimum, these steps:

1. Install the software.
2. Ensure the necessary port in the firewall is open.
3. Download a license.
4. Start the Server and/or configure it as a service.

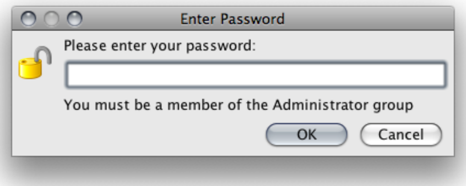
All these steps are covered in this section of the manual. Further configuration information, including for job nodes, grid nodes, and External Applications, are provided in later chapters.

Installing and running the *CLC Genomics Server* is straightforward. However, if you do run into troubles, please refer to the troubleshooting section in Appendix B, which provides tips on how to troubleshoot problems yourself, as well as how to get help.

2.1.1 Installing the Server software

The installation can only be performed by a user with administrative privileges. On some operating systems, you can double click on the installer file icon to begin installation. Depending on your operating system you may be prompted for your password (as shown in figure 2.1) or asked to allow the installation to be performed.

- On Windows 7 or Vista, you will need to right click on the installer file icon, and choose to **Run as administrator**.
- For the Linux-based package (rpm) installer, you can use your system's package tools, which generally require you to work as an administrative user.
- For the the Linux installation script (.sh), you will need to execute the script, either while working as an administrative user, or by prefacing the command with sudo.

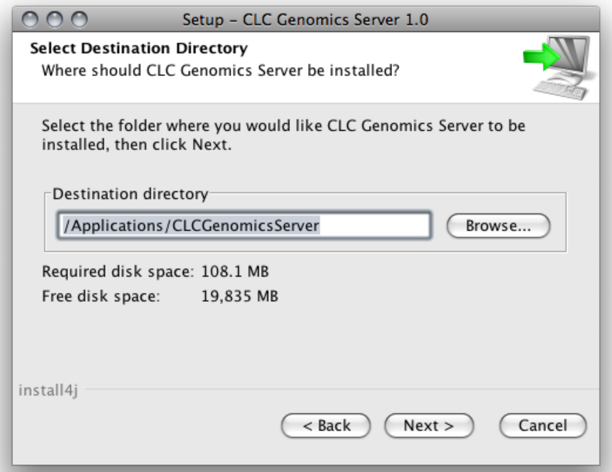

Enter Password

Please enter your password:

You must be a member of the Administrator group

Figure 2.1: *Enter your password.*

Next you will be asked where to install the server. If you do not have a particular reason to change this, simply leave it at the default setting. The chosen directory will be referred to as the *server installation directory* throughout the rest of this manual.


Setup – CLC Genomics Server 1.0

Select Destination Directory

Where should CLC Genomics Server be installed?

Select the folder where you would like CLC Genomics Server to be installed, then click Next.

Destination directory: /Applications/CLCGenomicsServer

Required disk space: 108.1 MB

Free disk space: 19,835 MB

Figure 2.2: *Choose where to install the server.*

The installer allows you to specify the maximum amount of memory the CLC Server will be able to utilize. The range of choice depends on the amount of memory installed on your system and on the type of machine used. On 32 bit machines you will not be able to utilize more than 2 GB of memory – on 64 bit machines there is no such limit.

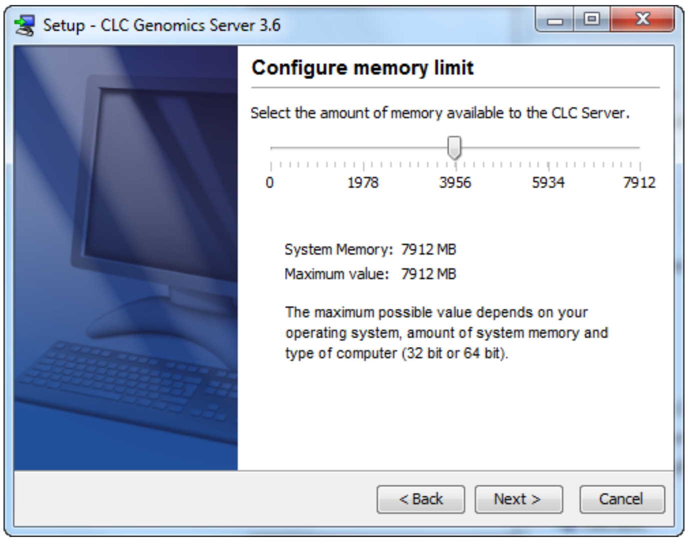
If you do not have a reason to change this value you should simply leave it at the default setting.

If you are installing the server on a Windows system you will be able to choose if the service is started manually or automatically by the system.

The installer will now extract the necessary files.

On a Windows system, if you have chosen that the service should be started automatically, the service should also start running at this point. Please note that if you do not already have a license file installed, then the *CLC Genomics Server* process will be running in a limited capacity at this point. Downloading a license is described in section 2.5.

Information on stopping and starting the *CLC Genomics Server* service is provided in section 2.6.



Configure memory limit

Select the amount of memory available to the CLC Server.

0 1978 3956 5934 7912

System Memory: 7912 MB
Maximum value: 7912 MB

The maximum possible value depends on your operating system, amount of system memory and type of computer (32 bit or 64 bit).

Figure 2.3: *Choose the maximum amount of memory used by the server.*

2.2 Silent installation

The installer also has a silent installation mode which is activated by the `-q` parameter when running the installer from a command line, e.g.

```
CLCGenomicsServer_3_0.exe -q
```

On Windows, if you wish to have console output, `-console` can be appended *as the second parameter* (this is only needed when running on Windows where there is no output per default):

```
CLCGenomicsServer_3_0.exe -q -console
```

You can also in silent mode define a different installation directory: `-dir`.

```
CLCGenomicsServer_3_0.exe -q -console -dir "c:\bioinformatics\clc"
```

Note! Both the `-console` and the `-dir` options only work when the installer is run in silent mode.

The `-q` and the `-console` options work for the Uninstall program as well.

2.3 Upgrading an existing installation

Upgrading an existing installation is very simple. For a single Genomics Server, the steps we recommend are:

- Make sure that nobody is using the server (see section 4.4). A standard procedure would be to give users advance notice that the system will be unavailable for maintenance.
- Install the server in the same installation directory as the one already installed. All settings will be maintained.
- Check that all plug-ins installed on the *CLC Genomics Server* are up to date, and that all client users are aware that they must upgrade their server connection plug-in (for Workbenches) or their software (CLC Command Line Tools).

If you have a CLC job node setup, you will also need to upgrade the *CLC Genomics Server* software on each job node. Upgrading the software itself on each node is all you need to do. Configurations and plug-ins for job nodes are pushed to them by the master node.

If you are running the CLC Grid Integration Tool, you will need to redeploy your grid worker script(s) after you upgrade the server.

For major versions a new license needs to be downloaded (see section 2.5).

2.4 Allowing access through your firewall

By default, the server listens for TCP-connections on port 7777 (See section 3.4 for info about changing this).

If you are running a firewall on your server system you will have to allow incoming TCP-connections on this port before your clients can contact the server from a Workbench or web browser. Consult the documentation of your firewall for information on how to do this.

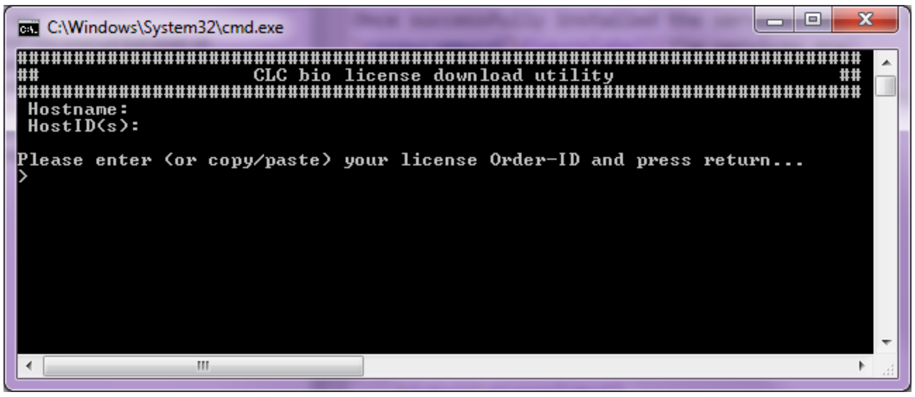
Besides the public port described above the server also uses an internal port on 7776. There is no need to allow incoming connections from client machines to this port.

2.5 Downloading a license

The license server will look for licenses in the `licenses` folder. This means that all license files should be located in this folder. Check the platform-specific instructions below to see how to download a license file.

2.5.1 Windows license download

License files are downloaded using the `licensedownload` script. To run the script, right-click on the file and choose **Run as administrator**. This will present a window as shown in figure 2.4.



```
##########################################################################
##                     CLC bio license download utility                 ##
##########################################################################
 Hostname:
 HostID(s):

Please enter (or copy/paste) your license Order-ID and press return...
>
```

Figure 2.4: *Download a license based on the Order ID.*

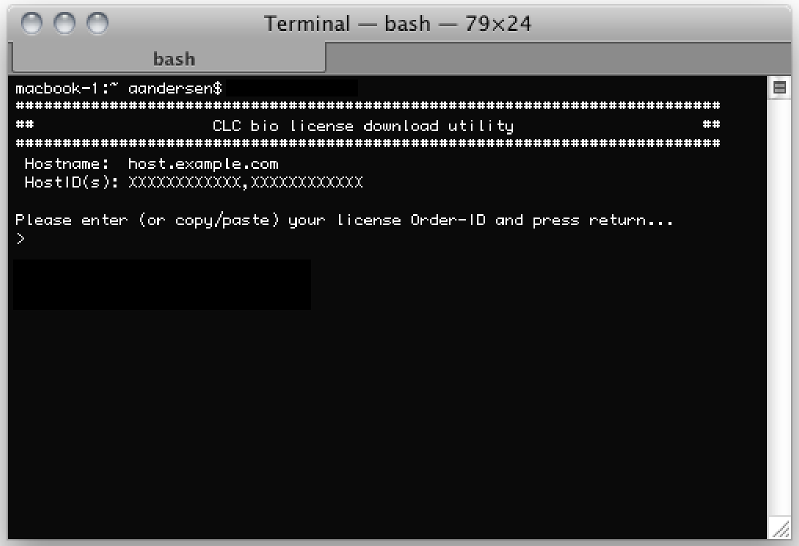
Paste the Order ID supplied by CLC bio (right-click to **Paste**) and press Enter. Please contact support@clcbio.com if you have not received an Order ID.

Note that if you are *upgrading* an existing license file, this needs to be deleted from the `licenses` folder. When you run the `licensedownload` script, it will create a new license file.

Restart the server for the new license to take effect (see how to restart the server in section 2.6.1).

2.5.2 Mac OS license download

License files are downloaded using the `downloadlicense.command` script. To run the script, double-click on the file. This will present a window as shown in figure 2.5.



```
macbook-1:~ aandersen$ [redacted]
###########################################################################
##                   CLC bio license download utility                    ##
###########################################################################
 Hostname:  host.example.com
 HostID(s): XXXXXXXXXXXX,XXXXXXXXXXXX

Please enter (or copy/paste) your license Order-ID and press return...
>

[redacted]
```

Figure 2.5: *Download a license based on the Order ID.*

Paste the Order ID supplied by CLC bio and press Enter. Please contact support@clcbio.com if you have not received an Order ID.

Note that if you are *upgrading* an existing license file, this needs to be deleted from the `licenses` folder. When you run the `downloadlicense.command` script, it will create a new license file.

Restart the server for the new license to take effect (see how to restart the server in section 2.6.2).

2.5.3 Linux license download

License files are downloaded using the `downloadlicense` script. Run the script and paste the Order ID supplied by CLC bio. Please contact support@clcbio.com if you have not received an Order ID.

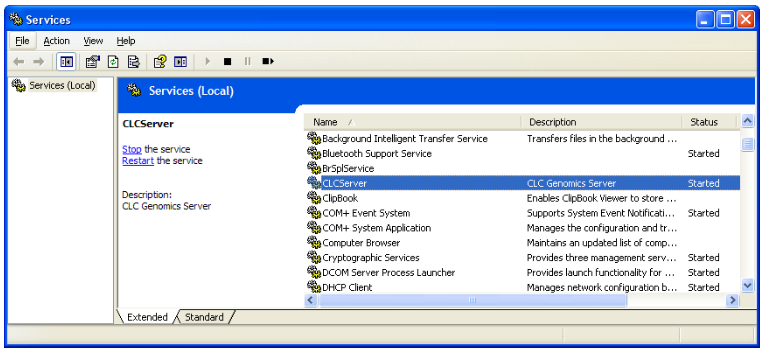
Note that if you are *upgrading* an existing license file, this needs to be deleted from the `licenses` folder. When you run the `downloadlicense` script, it will create a new license file.

Restart the server for the new license to take effect. Restarting the server is covered in in section 2.6.3.

2.6 Starting and stopping the server

2.6.1 Microsoft Windows

On Windows based systems the *CLC Genomics Server* can be controlled through the *Services* control panel. Choose the service called `CLCGenomicsServer` and click the start, stop or restart link as shown in figure 2.6.



CLCServer

Stop the service
Restart the service

Description:
CLC Genomics Server

Name	Description	Status
Background Intelligent Transfer Service	Transfers files in the background ...	
Bluetooth Support Service		Started
BrSplService		
CLCServer	CLC Genomics Server	Started
ClipBook	Enables ClipBook Viewer to store ...	
COM+ Event System	Supports System Event Notificati...	Started
COM+ System Application	Manages the configuration and tr...	
Computer Browser	Maintains an updated list of comp...	
Cryptographic Services	Provides three management serv...	Started
DCOM Server Process Launcher	Provides launch functionality for ...	Started
DHCP Client	Manages network configuration b...	Started

Figure 2.6: *Stopping and restarting the server on Windows by clicking the blue links.*

2.6.2 Mac OS X

On Mac OS X the server can be started and stopped from the command line.

Open a terminal and navigate to the CLC Server installation directory. Once there the server can be controlled with the following commands.

To start the server run the command:

```
sudo ./CLCGenomicsServer start
```

To stop the server run the command:

```
sudo ./CLCGenomicsServer stop
```

To view the current status of the server run the command:

```
sudo ./CLCGenomicsServer status
```

You will need to set this up as a service if you wish it to be run that way. Please refer to your operating system documentation if you are not sure how to do this.

2.6.3 Linux

You can start and stop the Genomics Server from the command line. You can also run it as a service.

Command Line:

Open a terminal and navigate to the CLC Server installation directory. Once there the server can be controlled with the following commands.

To start the server run the command:

```
./CLCGenomicsServer start
```

To stop the server run the command:

```
./CLCGenomicsServer stop
```

To view the current status of the server run the command:

```
./CLCGenomicsServer status
```

You can run these commands as any user, either directly on the command, or by prefacing the commands with sudo. e.g .

```
 sudo -u <username> ./CLCGenomicsServer status
```

Please note: To run the Genomics Server as a non-root user, you should first ensure that the files under your *CLC Genomics Server* installation directory have the appropriate permissions. The easiest way to do this is to recursively change the ownership of all the files in that directory to the user that will run the *CLC Genomics Server* process.

As a service:

On installation a link is created in /etc/init.d to the CLCGenomicsServer script. At this point, you need to configure it as a service. For example, on RedHat systems,

- service clcserver-startupscript start

We provide an example wrapper script for running the *CLC Genomics Server* process. You may find this convenient if you plan to run the process as a user other than the root user. The script is called clcserver-startupscript and can be found under the **conf** directory of the *CLC Genomics Server* installation area.

Please refer to your operating system documentation if you require further information about setting up services on your system.

2.7 Installing relevant plug-ins in the Workbench

In order to use the *CLC Genomics Server* from a CLC Workbench, you need to install a plug-in to the Workbench. If you connect to the server from a *CLC Genomics Workbench*, you should install the CLC Genomics Server Plug-in. If you are using e.g. *CLC Main Workbench*, you should install the generic CLC Server Plug-in. Both plug-ins allow you to log in to the server and access data from the server data locations, but the CLC Genomics Server Plug-in also includes the tools to access the analyses on the server (e.g. assembly, RNA-Seq).

The plug-ins can be found on the same web page as the server installer or at http://www.clcbio.com/plugins.

Plug-ins are installed using the plug-in manager[1]:

Help in the Menu Bar | Install Plug-ins ( )

or **Plug-ins ( ) in the Toolbar**

Install the plug-in by clicking the **Install from File** button at the bottom of the dialog. This will

[1]In order to install plug-ins on Windows Vista, the Workbench must be run in administrator mode: Right-click the program shortcut and choose "Run as Administrator". Then follow the procedure described below.

open a dialog where you can browse for the plug-in. The plug-in file is provided by CLC bio on the page where you downloaded the server installation.

You need to restart the Workbench before the plug-in is ready for use.

Note that if you want users to be able to use **External applications** (see chapter 9) on the server, there is a separate plug-in (CLC External Applications Plug-in) that needs to be installed in the Workbench the same way as described above.

Chapter 3

Configuring and administering the server

3.1 Logging into the administrative interface

Once the server is running, you can log into its administrative interface via a web browser. Most configuration occurs via this interface. Simply type the host name of the server machine you have installed the *CLC Genomics Server* software on, followed by the port it is listening on. Unless you change it, the port number is 7777. An example would be

```
http://clccomputer:7777/
```

The default administive user credentials are:

```
username: root
password: default
```

Use these details the first time you log in.

3.2 Adding locations for saving data

Before you can use the server for doing analyses you will need to add one or more locations for storing your data.

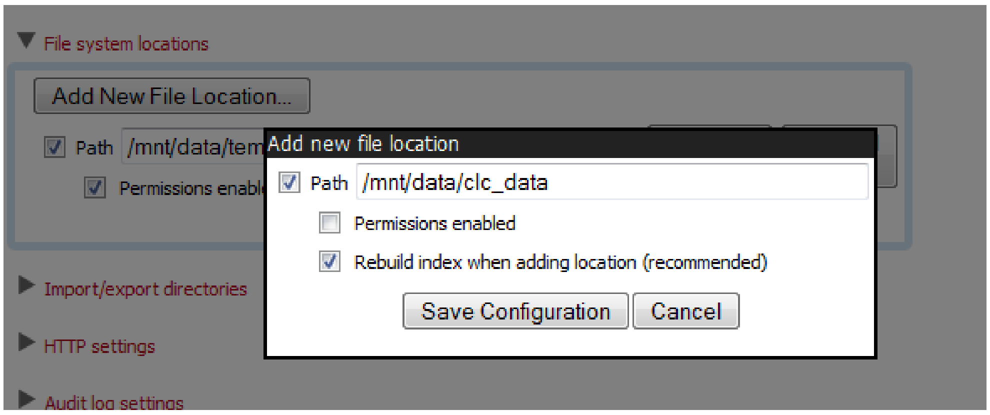
3.2.1 Adding a file system location

Under the **File system locations** heading, click the **Add New File Location** button to add a new file system location (see figure 3.1).

In this dialog, enter the path to the folder you want to use for storing the data. The path should point to an *existing* folder on the server machine, and the user *running the server process* needs to have read and write access to the folder. This is usually a dedicated user, or it may be the system's root user if you have not created a dedicated user for this purpose.

The file location(s) configured on the server will be accessible to those working using CLC Workbenches after they log into the server via their Workbench.

Once you have pressed **Save Configuration** (learn more about rebulding the index in section 3.2.2), this location will be added and it should now appear in the **Navigation Area** in the left



Add new file location

☑ Path /mnt/data/clc_data

☐ Permissions enabled

☑ Rebuild index when adding location (recommended)

Save Configuration | Cancel

Figure 3.1: *File system location settings.*

hand side of the window. By default it will also appear in the Workbench on next login. You can use the checkbox next to the location to indicate whether it should be visible to your users or not.

You can choose whether access control should be switched on and off. Please see section 5.1 for more information about enabling and setting permissions on *CLC Genomics Server* data folders.

Note that pressing **Remove Location** will only remove the location from this list - it will not delete the folder from your system or affect any data already stored in this folder. The data will be accessible again simply by adding the folder as a new location again.

Important points about the CLC Server data in the file system locations

Any file system locations added here should be folders **dedicated for use** by the *CLC Genomics Server*. Such areas should be directly accessed only by the *CLC Genomics Server*. In other words, files should **not** be moved into these folders, or their subfolders, manually, for example using your standard operating system's command tools, drag and drop, and so on. All the data stored in this areas will be in clc format and will be owned by the user that runs the *CLC Genomics Server* process.

File locations for job node set-ups

When you have a job node set-up, all the job node computers need to have access to the same data location folder. This is because the job nodes will write files directly to the folder rather than passing through the master node (which would be a bottleneck for big jobs). Furthermore, the user running the server must be the same for all the job nodes and it needs to act as the same user when accessing the folder no matter whether it is a job node or a master node.

The data location should be added **after** the job nodes have been configured and attached to the master node. In this way, all the job nodes will inherit the configurations made on the master node.

One relatively common problem faced in this regard is *root squashing* which often needs to be disabled, because it prevents the servers from writing and accessing the files as the same user - read more about this at `http://nfs.sourceforge.net/#faq_b11`.

You can read further about job node setups in section 7

3.2.2 Rebuilding the index

The server maintains an index of all the elements in the data locations. The index is used when searching for data. For all locations you can choose to **Rebuild Index**. This should be done only when a new location is added or if you experience problems while searching (e.g. something is missing from the search results). This operation can take a long time depending on how much data is stored in this location.

If you move the server from one computer to another, you need to move the index as well. Alternatively, you can re-build the index on the new server (this is the default option when you add a location). If the rebuild index operation takes too long and you would prefer to move the old index, simply copy the folder called `searchindex` from the old server installation folder to the new server.

The status of the index server can be seen in the **User Statistics** pane showing information on where the index server resides and the number of locations currently being serviced.

3.3 Accessing files on, and writing to, areas of the server filesystem

There are circumstance when it is beneficial to be able to interact with (non-CLC) files directly on your server filesystem. A common circumstance would be importing high-throughput sequencing data from folders where it is stored on the same system that your *CLC Genomics Server* is running on. This could eliminate the need for each user to copy large sequence data files onto the machine their CLC Workbench is running on before importing the data into a *CLC Genomics Server* CLC server data area. Another example is if you wished to export data from CLC format to other formats and save those files on your server machine's filesystem (as opposed to saving the files onto the system your Workbench is running on).

From the administrator's point of view, this is about configuring folders that are safe for the *CLC Genomics Server* to read and write to on the server machine' system.

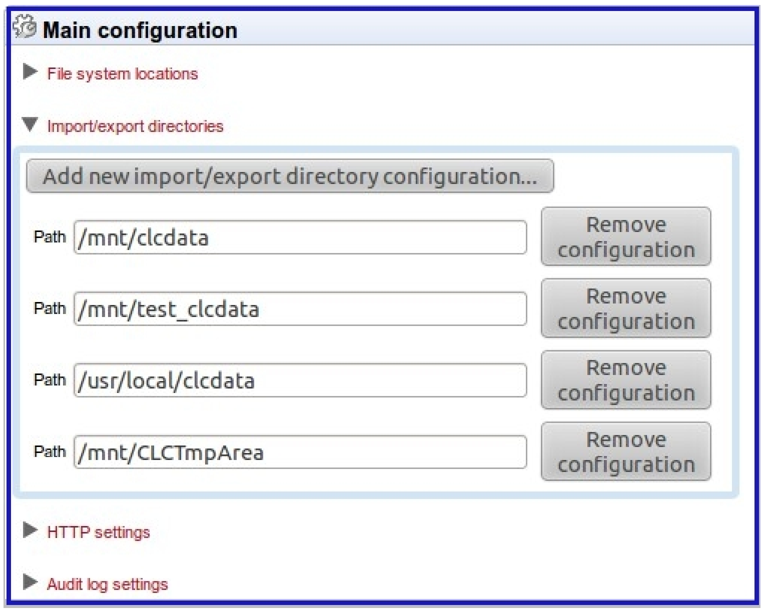
This means that users logged into the *CLC Genomics Server* from their Workbench will be able to access files in that area, and potentially write files to that area. Note that the *CLC Genomics Server* will be accessing the file system as the *user running the server process* - not as the user logged into the Workbench. This means that you should be careful when opening access to the server filesystem in this way. Certainly, only folders that do not contain sensitive information should be added.

Folders to be added for this type of access are configured in the web administration interface in the **Main configuration** tab under **Import/export directories** (see figure 3.2). (Earlier versions of the server software referred to these locations as High-throughput sequencing data import locations.)

Press the **Add new import/export directory location** button to specify a path to a folder on the server. This folder and all its subfolders will then be available for browsing in the Workbench for certain activities (e.g. importing data functions).

For example, in the case where a directory has been set up an Import/export location, and a user, logged into the *CLC Genomics Server* via their CLC Workbench wishes to import some high throughput sequencing data, they would see an option like that shown in figure 3.3.

On my local disk or a place I have access to means that the user will be able to select files

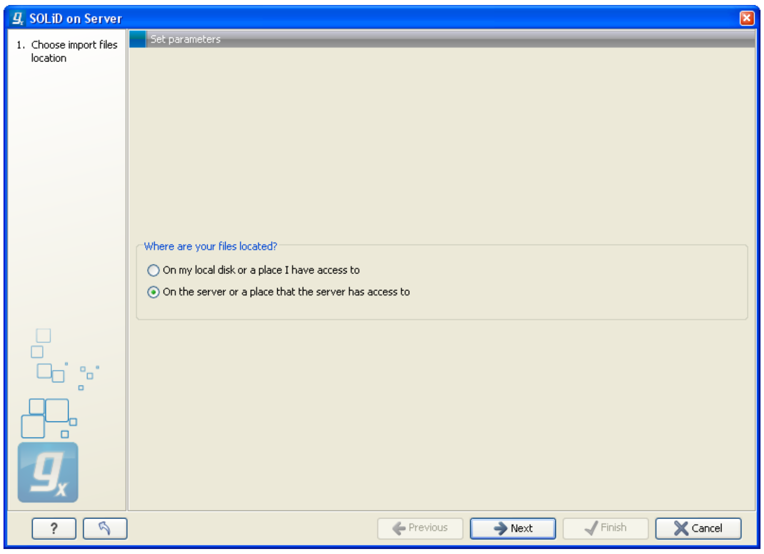


Import/export directories

Add new import/export directory configuration...

Path	/mnt/clcdata	Remove configuration
Path	/mnt/test_clcdata	Remove configuration
Path	/usr/local/clcdata	Remove configuration
Path	/mnt/CLCTmpArea	Remove configuration

Figure 3.2: *Defining source folders that should be available for browsing from the Workbench.*



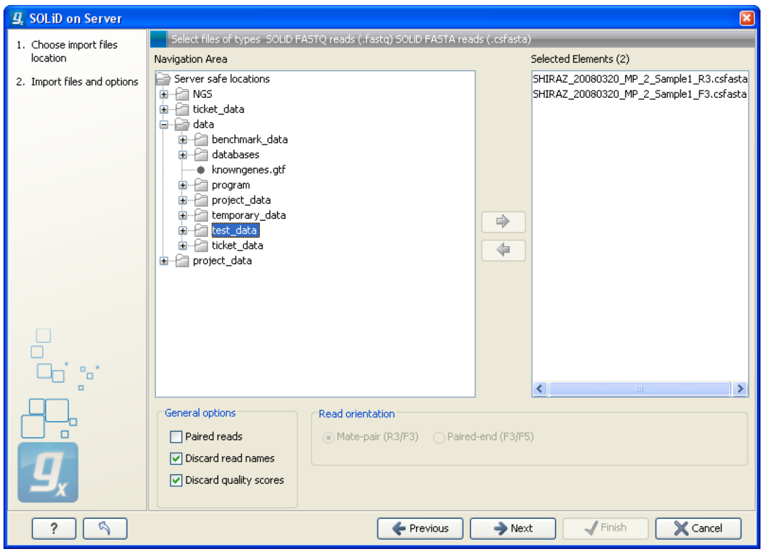
Where are your files located?

On my local disk or a place I have access to [✓ On the server or a place that the server has access to]

Figure 3.3: *Deciding source for high-throughput sequencing data files.*

from the file system of the machine their CLC Workbench is installed on. These files will then be transferred over the network to the server and placed as temporary files for importing. If the user chooses instead the option *On the server or a place the server has access to*, the user is presented with a file browser for the selected parts of the server file system that the administator has configured as an Import/export location. See figure 3.4.

Note: Import/Export locations should NOT be set to subfolders of any defined CLC file or data locations. CLC file and data locations should be used for CLC data, and data should only be added or removed from these areas by CLC tools. By definition, an Import/Export folder is meant for holding non-CLC data, for example, sequencing data that will be imported, data that you export from the Genomics Server, or blast databases.


SOLiD on Server

1. Choose import files location
2. Import files and options

Select files of types SOLiD FASTQ reads (.fastq) SOLiD FASTA reads (.csfasta)

Navigation Area

```
Server safe locations
  NGS
  ticket_data
  data
    benchmark_data
    databases
    knowngenes.gtf
    program
    project_data
    temporary_data
    test_data
    ticket_data
  project_data
```

Selected Elements (2)

SHIRAZ_20080320_MP_2_Sample1_R3.csfasta
SHIRAZ_20080320_MP_2_Sample1_F3.csfasta

General options
- ☐ Paired reads
- ✓ Discard read names
- ✓ Discard quality scores

Read orientation: [✓ Mate-pair (R3/F3)] Paired-end (F3/F5)

Figure 3.4: *Selecting files on server file system.*

3.4 Changing the listening port

The default listening port for the CLC Server is 7777. This has been chosen to minimize the risk of collisions with existing web-servers using the more familiar ports 80 and 8080. If you would like to have the server listening on port 80 in order to simplify the URL, this can be done in the following way.

- Navigate to the CLC Server installation directory.
- Locate the file called *server.xml* in the conf directory.
- Open the file in a text editor and locate the following section

```
<Connector port="7777" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

- Change the port value to desired listening port (80 in the example below)

```
<Connector port="80" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

- Restart the service for the change to take effect (see how to restart the server in section 2.6).

3.5 Changing the tmp directory

The *CLC Genomics Server* often uses a lot of disk space for temporary files. These are files needed during an analysis, and they are deleted when no longer needed. By default, these temporary files are written to your system default temporary directory. Due to the amount of space that can be required for temporary files, it can be useful to specify an alternative, larger, disk area where temporary files created by the CLC Genomics Server can be written.

In the *server installation directory* you will find a file called `CLCGenomicsServer.vmoptions`. Open this file in a text editor and add a new line like this: `-Djava.io.tmpdir=/path/to/tmp` with the path the new tmp directory. Restart the server for the change to take effect (see how to restart the server in section 2.6).

We highly recommend that the tmp area is set to a file system local to the server machine. Having tmp set to a file system on a network mounted drive can substantially affect the speed of performance.

3.5.1 Job node setups

The advice about having a tmp area being set on a local file system is true also for job nodes. Here, the tmp areas for nodes should **not** point to a shared folder. Rather, each node should have a tmp area with an identical name and path, but situated on a drive local to each node.

You will need to edit the `CLCGenomicsServer.vmoptions` file on each job node, as well as the master node, as described above. This setting is **not** pushed out from the master to the job nodes.

3.6 Limiting the number of cpus available for use

A number of the algorithms in the *CLC Genomics Server* will, in the case of large jobs, use all the cores available on your system to make the analysis as fast as possible. If you wish to restrict this to a predefined number of cores, this can be done with a properties file: Create a text file called *cpu.properties* and save it in the *settings* folder under the *CLC Genomics Server* installation directory.

The cpu.properties file should include one line like this:

maxcores = 1

Restart the *CLC Genomics Server* if you create or change this file for these settings to take effect.

Instead of 1 you write the maximum number of cores that the *CLC Genomics Server* is allowed to use. Please note that this is not a guarantee that the *CLC Genomics Server* will never use more cores than specified, but that will be for very brief and infrequent peaks and should not affect performance of other applications running on your system.

You can download a sample cpu.properties file at http://clcbio.com/files/deployment/cpu.properties.

3.7 Other configurations

3.7.1 HTTP settings

Under the **Admin** ( ) tab, click **Configuration**, and you will be able to specify HTTP settings. Here you can set the time out for the user HTTP session and the maximum upload size (when uploading files through the web interface).

3.7.2 Audit log settings

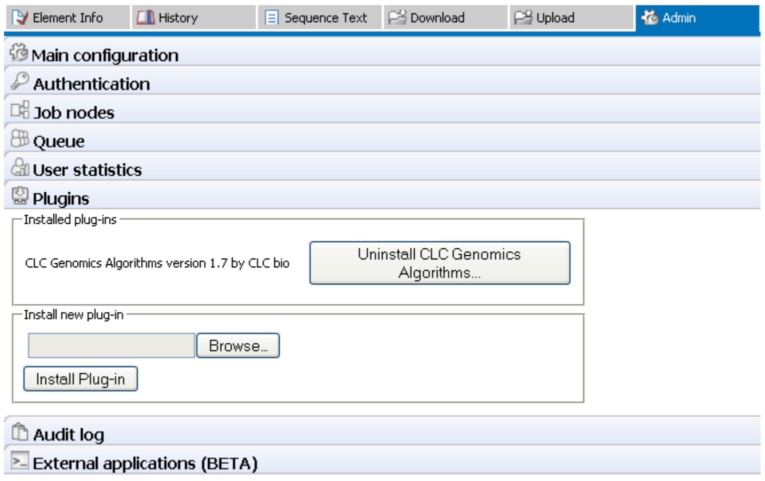
The audit log records all the actions performed in the web interface and through the Workbenches. Note that data management operations (copying, deleting and adding files) done through the Workbench are not recorded.

3.7.3 Deployment of server information to CLC Workbenches

See the *Deployment manual* at http://www.clcbio.com/usermanuals for information on pre-configuring the server log-in information when Workbench users log in for the first time.

3.8 Server plug-ins

You can install plug-ins on the server under the **Admin** ( ) tab (see figure 3.5).



CLC Genomics Algorithms version 1.7 by CLC bio

Figure 3.5: *Installing and uninstalling server plugins.*

Click the **Browse** button and locate the plug-in .cpa file to install a plug-in. To uninstall a plug-in, simply click the button next to the plug-in. The server does not need to be restarted after installation/uninstallation of plug-ins.

If you are running a job node setup, you **only** need to install the plug-in on the master server. However, you **need to check that the plugin is enabled for each job node** by going to the Job Distribution tab in the master nodes web administrative interface, and clicking on the link to each job node.

Read more about developing server plug-ins at http://www.clcdeveloper.com/.

3.9 Queue

Clicking the **Queue** panel will show a list of all the processes that are currently in the queue (including the one in progress). An example is shown in figure 3.6.

For each process, you are able to **Cancel** ( ) and re-prioritize the order of the processes by clicking the up and down arrows. Some processes also allow you to pause and resume. At the top, you can see the progress of the process that is currently running.

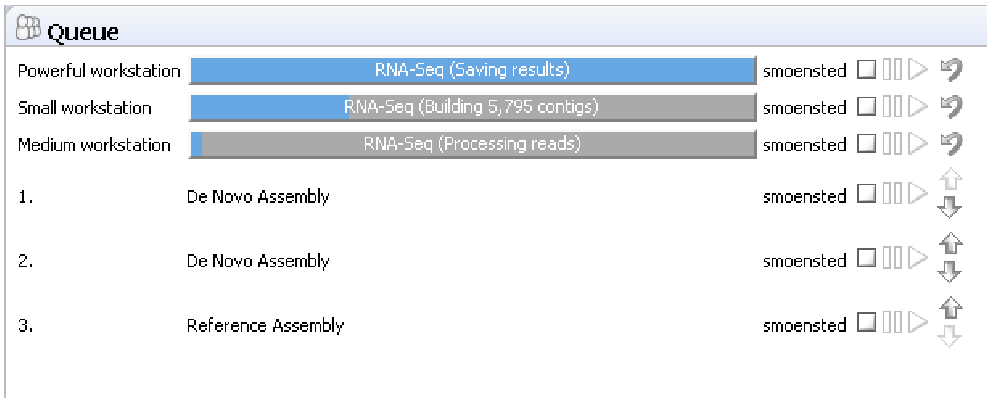


Figure 3.6: *The process queue.*

If you are running a CLC Server with execution nodes, you can also **Stop and requeue** ( ) a job that is currently being processed.

Chapter 4

Managing users and groups

4.1 Logging in the first time - root password

When the server is installed, you will be able to log in via the web interface using the following credentials:

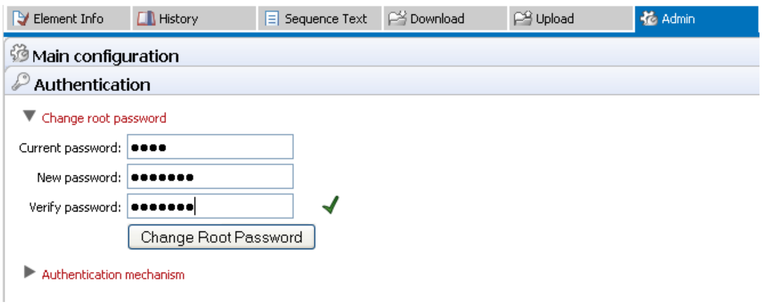
- **User name**: `root`
- **Password**: `default`

Once logged in, you should as a minimum set up user authentication (see section 4.2) and data locations (see section 3.2) before you can start using the server.

For security reasons, you should change the root password (see figure 4.1):

Admin ( ) | Authentication ( ) Change root password

Note that if you are going to use job nodes, it makes sense to set these up before changing the authentication mechanism and root password (see section 7).



Current password: ●●●●
New password: ●●●●●●●
Verify password: ●●●●●●● ✓

Figure 4.1: *We recommend changing the root password. The verification of the root password is shown with the green checkmark.*

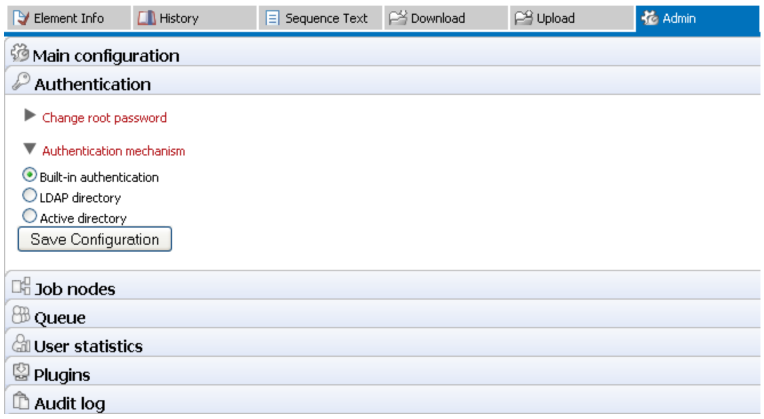
4.2 User authentication using the web interface

When the server is installed, you can log in using the default root password (username=root, password=default). Note that if you are going to use job nodes, it makes sense to set this up first before changing the authentication mechanism and root password (see section 7).

Once logged in, you can specify how the general user authentication should be done:

Admin ( ) | Authentication ( ) Authentication mechanism

This will reveal the three different modes of authentication as shown in figure 4.2.



[✓ Built-in authentication]
LDAP directory
Active directory

Figure 4.2: *Three modes of user authentication.*

The options are:

- **Built-in authentication**. This option will enable you to set up user authentication using the server's built-in user management system. This means that you create users, set passwords, assign users to groups and manage groups using the web interface (see section 4.2.1) or using the Workbench (see section 4.3.1). All the user information is stored on the server and is not accessible from other systems.
- **LDAP directory**. This option will allow you to use an existing LDAP directory. This means that all information needed during authentication and group memberships is retrieved from the LDAP directory.
- **Active directory**. This option will allow you to use an existing Active directory which is Microsoft's LDAP counterpart. This means that all information needed during authentication and group memberships is retrieved from the Active directory.

For the two last options, a settings panel will be revealed when the option is chosen, allowing you to specify the details of the integration.

Note that membership of an administrative group is used to control which users can access the admin part of the web interface. These users will also be able to set permissions on folders (see section 5). For the built-in authentication method, this means adding particular users to the built-in **admin** group. For Active Directory or LDAP, this means designating a group in the box labeled **Admin group name** and adding any users who should be administrators of the CLC Server to this group.

4.2.1 Managing users using the web interface

To create or remove users or change their password:

Admin ( ) | Users and groups ( ) Manage user accounts

This will display the panel shown in figure 4.3.

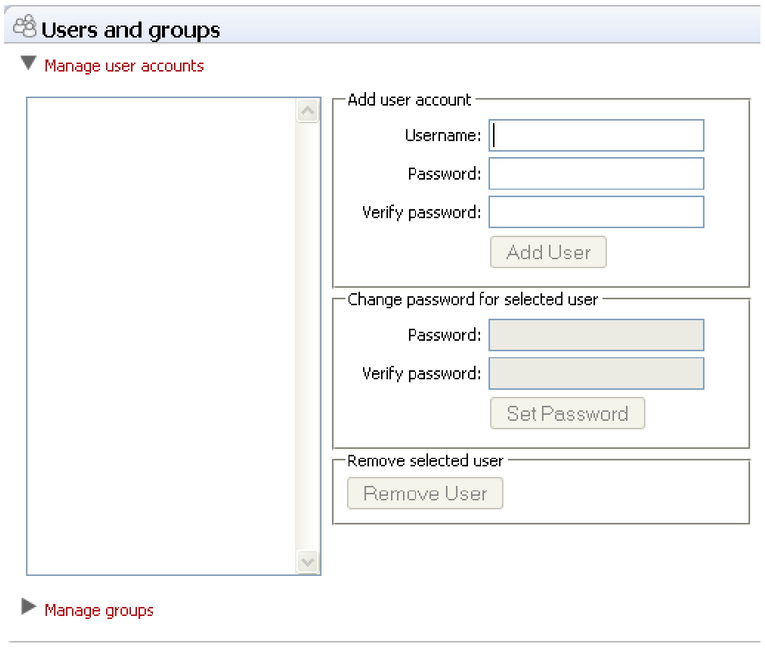


Figure 4.3: *Managing users.*

4.2.2 Managing groups using the web interface

To create or remove groups or change group membership for users:

Admin ( ) | Users and groups ( ) Manage groups

This will display the panel shown in figure 4.4.

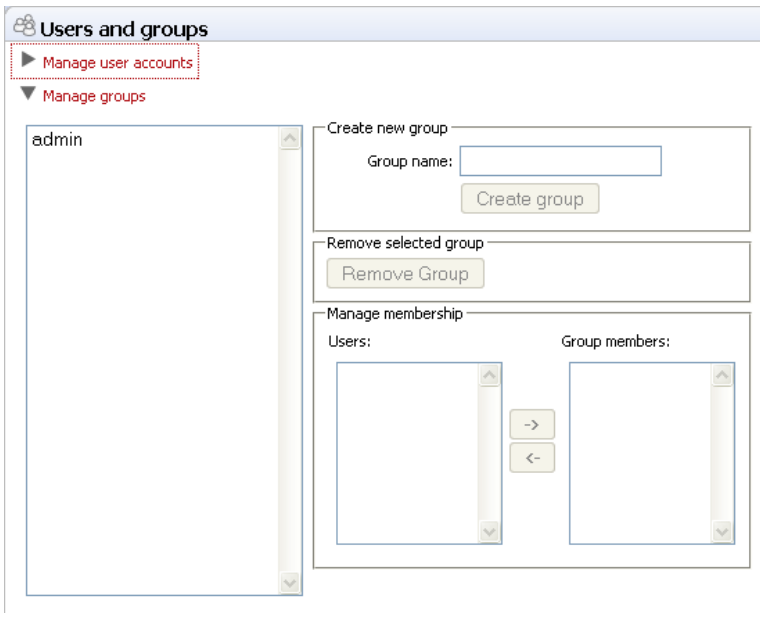


Figure 4.4: *Managing users.*

The same user can be a member of several groups.

Note that membership of the admin group is used for allowing users access to the admin part of the web interface. Users who should have access to the administrative part of the server should be part of the "admin" group which is the only special group (this group is already created for you).

Note that you will always be able to log as root with administrative access.

The functionality of this plug-in depends on the user authentication and management system: if the built-in system is used, all the functionality described below is relevant; if an external system is used for managing users and groups, the menus below will be disabled.

4.3 User authentication using the Workbench

Users and groups can also be managed through the Workbench (note that you need to set up the authentication mechanism as described in section 4.2):

File | Manage Users and Groups

This will display the dialog shown in figure 4.5.

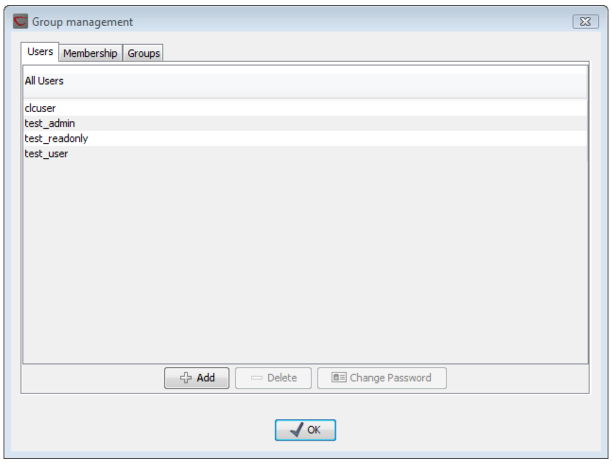


Figure 4.5: *Managing users.*

4.3.1 Managing users through the Workbench

Click the **Add** (✚) button to create a new user. Enter the name of the user and enter a password. You will be asked to re-type the password. If you wish to change the password at a later time, select the user in the list and click **Change password** ( ).

To delete a user, select the user in the list and click **Delete** (▭).

4.3.2 Managing groups through the Workbench

Access rights are granted to groups, not users, so a user has to be member of one or more groups to get access to the data location. Here you can see how to add and remove groups, and next you will see how to add users to a group.

Adding and removing groups is done in the **Groups** tab (see figure 4.6).

To create a new group, click the **Add** (✚) button and enter the name of the group. To delete a group, select the group in the list and click the **Delete** (▭) button.

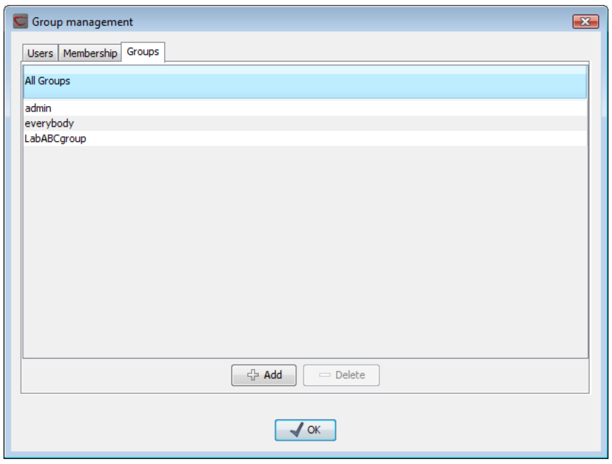


Figure 4.6: *Managing groups.*

4.3.3 Adding users to a group

When a new group is created, it is empty. To assign users to a group, click the **Membership** tab. In the **Selected group** box, you can choose among all the groups that have been created. When you select a group, you will see its members in the list below (see figure 4.7). The the left you see a list of all users.

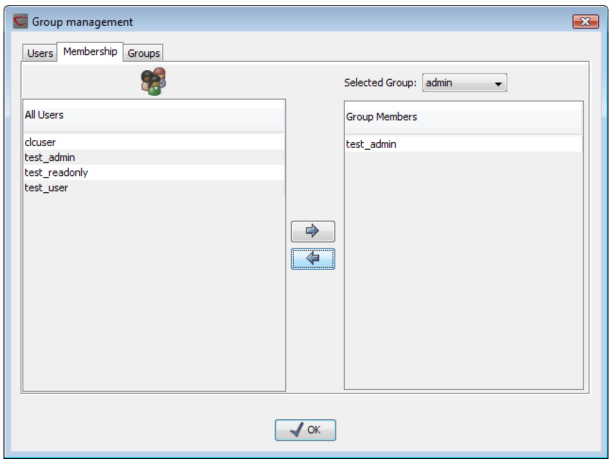


Figure 4.7: *Listing members of a group.*

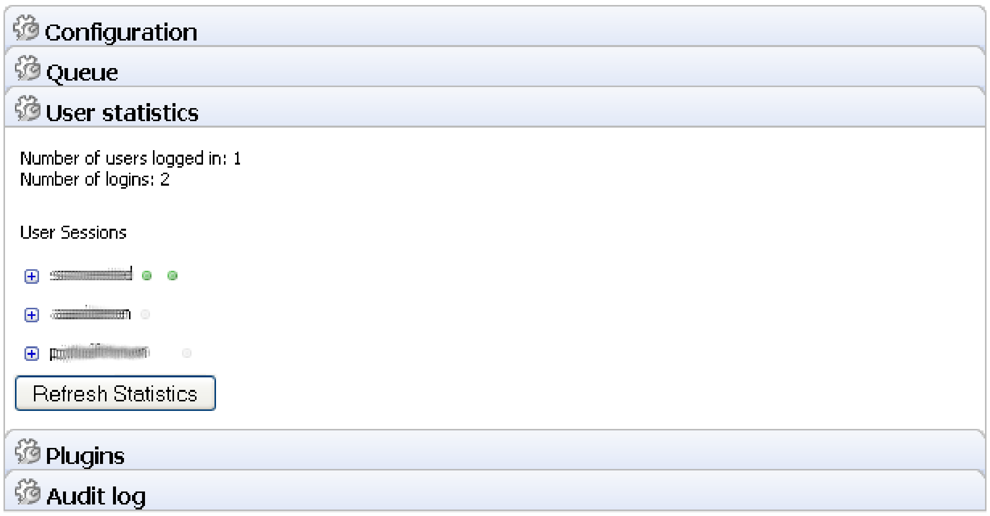
To add or remove users from a group, click the **Add** (➡) or **Remove** (⬅) buttons. To create new users, see section 4.3.1.

The same user can be a member of several groups.

4.4 User statistics

Clicking the **User statistics** panel will show a summary of the current usage of the server. An example is shown in figure 4.8.

You can see the number of users currently logged in, and you can see the number of sessions for each user. The two green dots indicate that this user is logged in twice (e.g. through the Workbench and through the web interface). The other two users have been logged in previously.



Number of users logged in: 1
Number of logins: 2

User Sessions

Figure 4.8: *The user statistics (user names have been blurred).*

You can also log users off by expanding the user sessions on the + sign and the click **Invalidate Session...**. This will open confirmation dialog where you can also write a message to the user that will displayed either in the Workbench or the browser.

Chapter 5

Access privileges and permissions

The *CLC Genomics Server* allows server administrators to control access to the server on several levels:

- Access to the data in the server's **data locations**. This is typically to allow groups of users to store data that cannot be accessed by other users, or to establish reference data sources that are "read-only" for most users.
- Access to **running jobs** on the server. Particular groups of users can be restricted to running only particular types of jobs on the server.
- Access to the **import/export directories**. The server administrator can give users access to browsing designated directories on the server file system. Thus it can be desirable to restrict this access to certain groups of users for security reasons.

The following sections describe each of these levels.

5.1 Controlling access to data

The *CLC Genomics Server* uses folders as the basic unit for controlling access to data, and access is granted (or denied) to groups of users.

On any folder within a location, you can grant two kinds of access to a group:

Read access This will make it possible for the users of the group to see the elements in this folder, to open them and to copy them. Access can be through any route, for example, browsing in the **Navigation Area**, searching or clicking "originates from" in the **History** ( ) of e.g. an alignment.

Write access Changes to an element can be saved **Save** ( ), and new elements and subfolders can be created.

For a user to be able to access a folder, there has to be at least read access to all the top folders. In the example shown in figure 5.1, to access the *Sequences* folder, the user must have at least read access to both the *Example Data* and *Protein* folders.

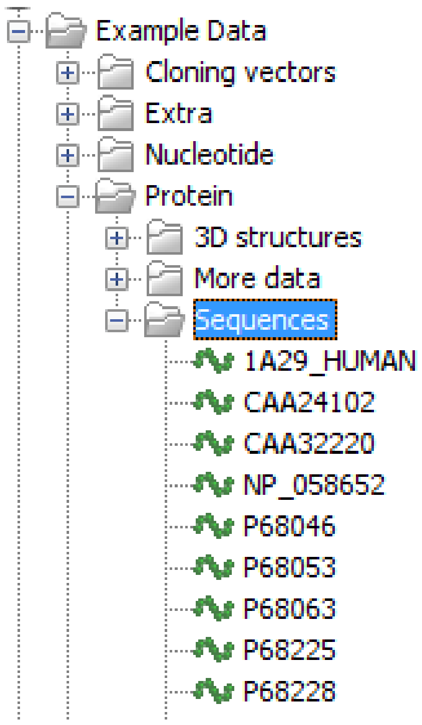


Figure 5.1: *A folder hierarchy on the server.*

However, you can grant read access to the *Example Data* and *Protein* folders and only grant write access to the *Sequences* folder.

Permissions on file system locations must be **explicitly enabled** if they are desired (see section 3.2.1). Please see 5.1.2 for further details about the system behaviour if permission are not enabled and configured.

If permissions are enabled on a file system location, then by default, no groups have read or write access to any area under this location until permissions are configured. Only the *CLC Genomics Server* root user will have access to the data held in the server at this point. In other words, you must set permissions on folders in this file system location before any users will be able to read from or write to it.

5.1.1 Setting permissions on a folder

This step is done from within a CLC Workbench. Start up a copy of a Workbench that has the CLC Server plugin installed. From within the Workbench, go to the File menu and choose the item **CLC Server Login**. Log into the CLC Server as an administrative user.

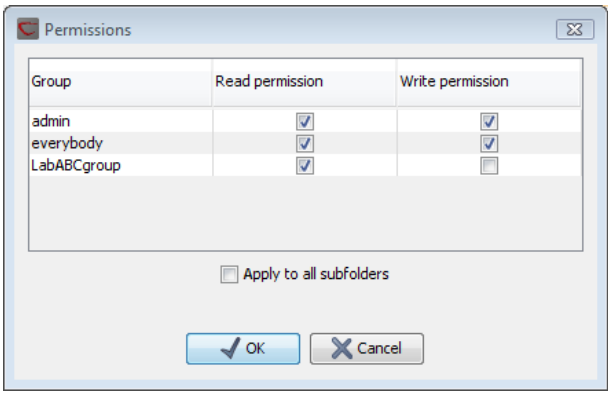
You can then set permissions on folders in your database, if you have one, or on folders within file system locations that have had permissions enabled.

right-click the folder ( ) | Permissions

This will open the dialog shown in figure 5.2.

Set the relevant permissions for each of the groups and click **OK**.

If you wish to apply the permissions recursively, that is to all subfolders, check **Apply to all subfolders** in the dialog shown in figure 5.2. **Note** that this operation is only relevant if you wish



Group	Read permission	Write permission
admin	✓	✓
everybody	✓	✓
LabABCgroup	✓	

Figure 5.2: *Setting permissions on a folder.*

to clean-up the permission structure of the subfolders. **It should be applied with caution**, since it can potentially destroy valuable permission settings in the subfolder structure.

Permissions on the recycle bin

The recycle bin is conceptually a folder like any else. It is special in the sense that all users must have write access (otherwise they will not be able to delete anything). Because there is one recycle bin for the data from all users, you should be careful granting everybody read access to the recycle bin, since they will then be able to see the data deleted by other users. We recommend only granting read access to administrators.

Only administrators are allowed to empty the recycle bin.

5.1.2 Technical notes about permissions and security

All data stored in *CLC Genomics Server* file system locations are owned by the user that runs the *CLC Genomics Server* process. Changing the ownership of the files using standard system tools is not recommended and will usually lead to serious problems with data indexing and hamper your work on the *CLC Genomics Server*.

One implication of the above ownership setup is that by default, (i.e. without permissions enabled), all users logging into the *CLC Genomics Server* are able to access all data within that file system location, and write data to that file system locations. All files created within such a file system location are then also accessible to all users of the *CLC Genomics Server*.

Group permissions on file system locations is an additional layer within the *CLC Genomics Server*, and is not part of your operating system's permission system. This means that enabling permissions, and setting access restrictions on CLC file system locations only affects users accessing data through CLC tools (e.g.using a Workbench, the CLC Command Line Tools, the *CLC Genomics Server* web interface or the Server API). If users have direct access to the data, using for example general system tools, the permissions set on the data in *CLC Genomics Server* has no effect.

5.2 Global permissions

In the server web interface, in the **Admin tab** you can set global permissions for the *CLC Genomics Server* as shown in figure 5.3.

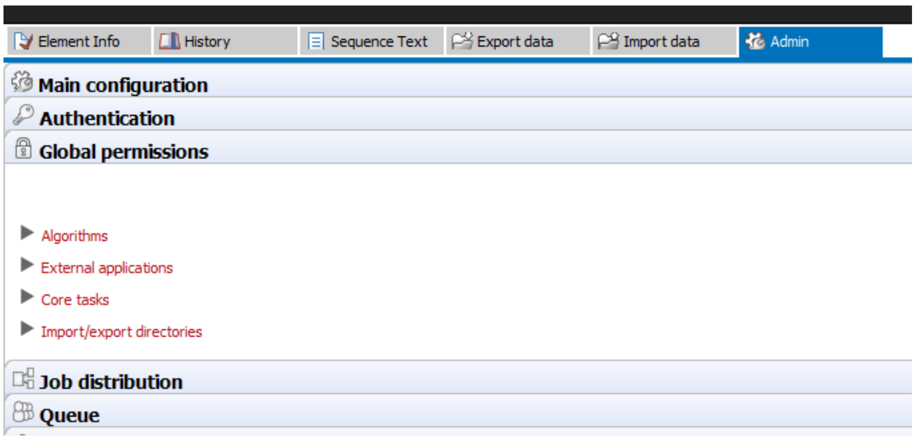
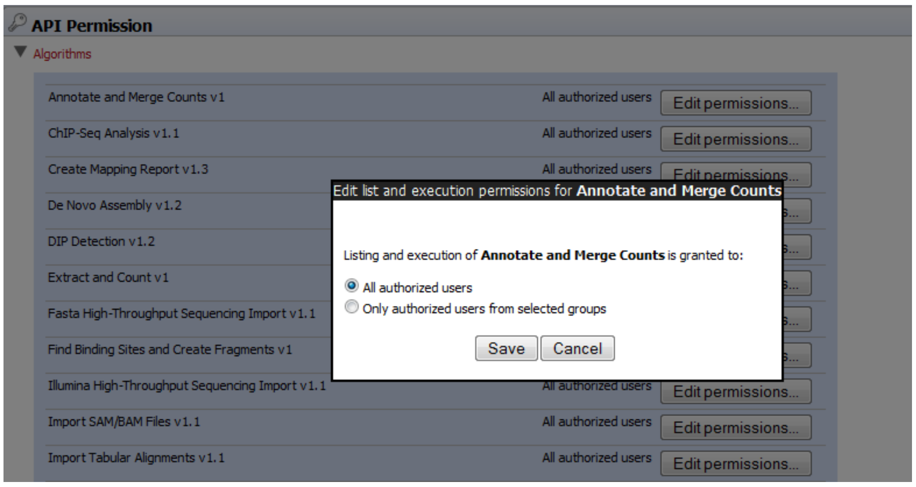


Figure 5.3: *Global permissions.*

Permissions can be set for:

- **Algorithms** All the algorithms running on the server are listed here.
- **External** applications. All the external applications configurations are listed here.
- **Core tasks** These are general import, export and maintenance tasks.
- **Import/export directories** Permissions can be set for each of the directories defined.

You can specify which groups should have access by clicking **Edit Permissions** button. A dialog will appear like that in figure 5.4) If you choose **Only authorized users from selected groups**, you will be offered a list of groups that you can select (or de-select) to give (or take away) access to that functionality.



Annotate and Merge Counts v1	All authorized users	Edit permissions...
ChIP-Seq Analysis v1.1	All authorized users	Edit permissions...
Create Mapping Report v1.3	All authorized users	Edit permissions...
De Novo Assembly v1.2		
DIP Detection v1.2		
Extract and Count v1		
Fasta High-Throughput Sequencing Import v1.1		
Find Binding Sites and Create Fragments v1		
Illumina High-Throughput Sequencing Import v1.1	All authorized users	Edit permissions...
Import SAM/BAM Files v1.1	All authorized users	Edit permissions...
Import Tabular Alignments v1.1	All authorized users	Edit permissions...

Edit list and execution permissions for **Annotate and Merge Counts**

Listing and execution of **Annotate and Merge Counts** is granted to:

[✓ All authorized users]
Only authorized users from selected groups

Figure 5.4: *Setting permissions for an alorithm.*

The default configuration is that all users have access to everything.

Chapter 6

Customized attributes on server data locations

The *CLC Genomics Server* makes it possible to define location-specific attributes on all elements stored in a data location. This could be company-specific information such as LIMS id, freezer position etc. Note that the attributes scheme belong to a location, so if a server has multiple locations, they will have their own separate set of attributes.

6.1 Configuring which fields should be available

To configure which fields that should be available, you need to log in as administrator via the Workbench.

Log in as administrator and:

right-click the data location ( ) | Location | Attribute Manager

This will display the dialog shown in figure 6.1.

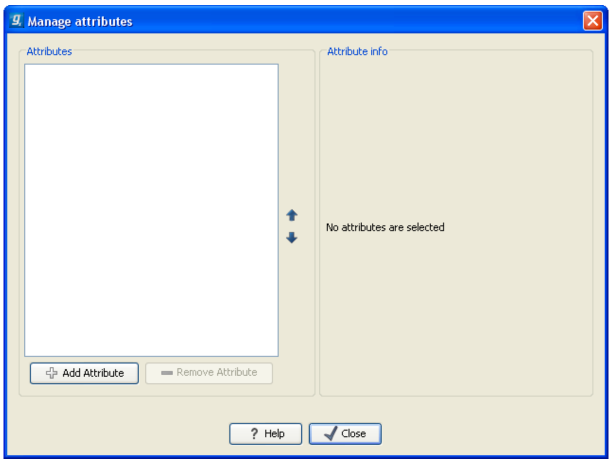


Figure 6.1: *Adding attributes.*

Click the **Add Attribute** ( ) button to create a new attribute. This will display the dialog shown in figure 6.2.

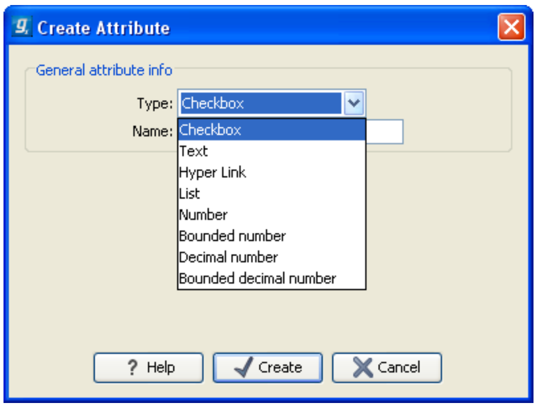


Figure 6.2: *The list of attribute types.*

First, select what kind of attribute you wish to create. This affects the type of information that can be entered by the end users, and it also affects the way the data can be searched. The following types are available:

- **Checkbox**. This is used for attributes that are binary (e.g. true/false, checked/unchecked and yes/no).
- **Text**. For simple text with no constraints on what can be entered.
- **List**. Lets you define a list of items that can be selected (explained in further detail below).
- **Number**. Any positive or negative integer.
- **Bounded number**. Same as number, but you can define the minimum and maximum values that should be accepted. If you designate some kind of ID to your sequences, you can use the bounded number to define that it should be at least 1 and max 99999 if that is the range of your IDs.
- **Decimal number**. Same as number, but it will also accept decimal numbers.
- **Bounded decimal number**. Same as bounded number, but it will also accept decimal numbers.
- **Hyper Link**. This can be used if the attribute is a reference to a web page. A value of this type will appear to the end user as a hyper link that can be clicked. Note that this attribute can only contain one hyper link. If you need more, you will have to create additional attributes.

When you click **OK**, the attribute will appear in the list to the left. Clicking the attribute will allow you to see information on it's type in the panel to the right.

6.1.1 Editing lists

Lists are a little special, since you have to define the items in the list. When you click a list in the left side of the dialog, you can define the items of the list in the panel to the right by clicking **Add Item** (➕) (see figure 6.3).

Remove items in the list by pressing **Remove Item** (➖).

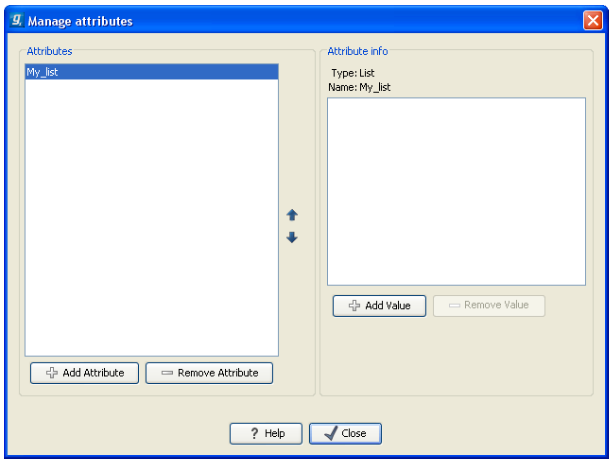


Figure 6.3: *Defining items in a list.*

6.1.2 Removing attributes

To remove an attribute, select the attribute in the list and click **Remove Attribute** (▭). This can be done without any further implications if the attribute has just been created, but if you remove an attribute where values have already been given for elements in the data location, it will have implications for these elements: The values will not be removed, but they will become static, which means that they cannot be edited anymore. They can only be removed (see more about how this looks in the user interface below).

If you accidentally removed an attribute and wish to restore it, this can be done by creating a new attribute of exactly the same name and type as the one you removed. All the "static" values will now become editable again.

When you remove an attribute, it will no longer be possible to search for it, even if there is "static" information on elements in the data location.

Renaming and changing the type of an attribute is not possible - you will have to create a new one.

6.1.3 Changing the order of the attributes

You can change the order of the attributes by selecting an attribute and click the **Up** and **Down** arrows in the dialog. This will affect the way the attributes are presented for the user as described below.

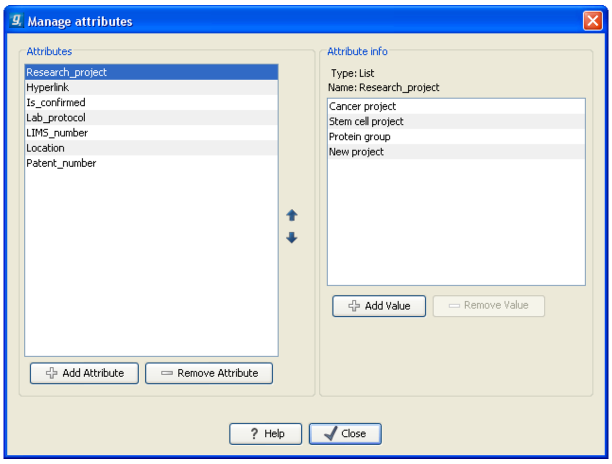
6.2 Filling in values

When a set of attributes has been created (as shown in 6.4), the end users can start filling in information.

This is done in the element info view:

> **select a sequence or another element in a server data location | Show ( ) in the Toolbar | Element info ( )**

This will open a view similar to the one shown in figure 6.5.

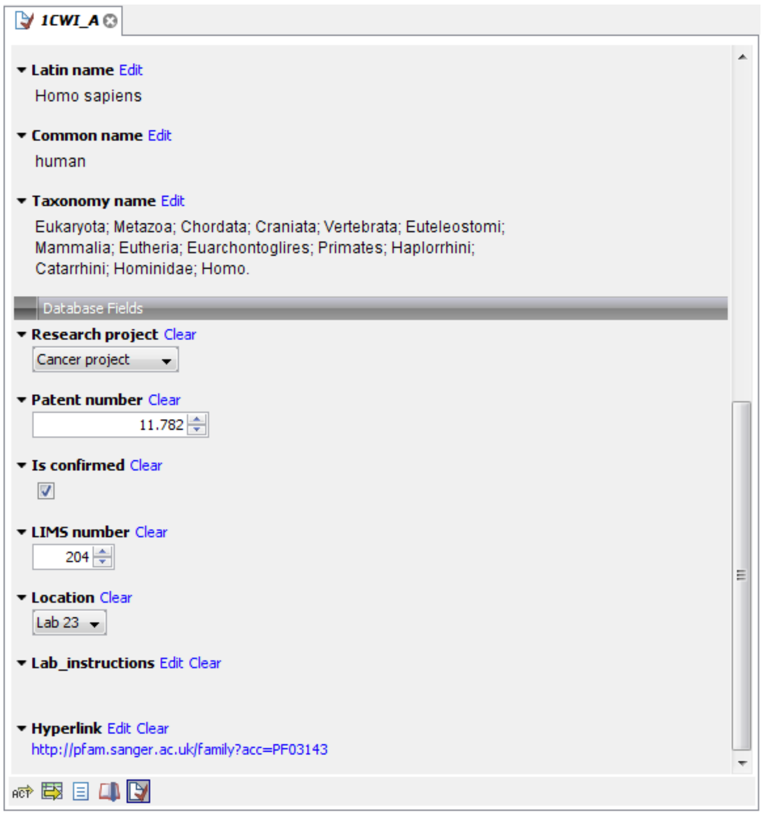


Research_project
Hyperlink
Is_confirmed
Lab_protocol
LIMS_number
Location
Patent_number

Type: List
Name: Research_project

Cancer project
Stem cell project
Protein group
New project

Figure 6.4: *A set of attributes defined in the attribute manager.*



Latin name Edit
Homo sapiens

Common name Edit
human

Taxonomy name Edit
Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Euarchontoglires; Primates; Haplorrhini; Catarrhini; Hominidae; Homo.

Research project Clear
Cancer project

Patent number Clear
11.782

Is confirmed Clear
✓

LIMS number Clear
204

Location Clear
Lab 23

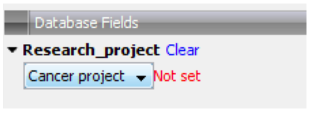
Lab_instructions Edit Clear

Hyperlink Edit Clear
http://pfam.sanger.ac.uk/family?acc=PF03143

Figure 6.5: *Adding values to the attributes.*

You can now enter the appropriate information and **Save**. When you have saved the information, you will be able to search for it (see below). Note that the sequence needs to be saved in the data location before you can edit the attribute values.

When nobody has entered information, the attribute will have a "Not set" written in red next to the attribute (see figure 6.6).



Research_project Clear
Cancer project Not set

Figure 6.6: *An attribute which has not been set.*

This is particularly useful for attribute types like checkboxes and lists where you cannot tell from the displayed value if it has been set or not. Note that when an attribute has not been set, you can't search for it, even if it looks like it has a value. In figure 6.6, you will *not* be able to find this sequence if you search for research projects with the value "Cancer project", because it has not been set. To set it, simply click in the list and you will see the red "Not set" disappear.

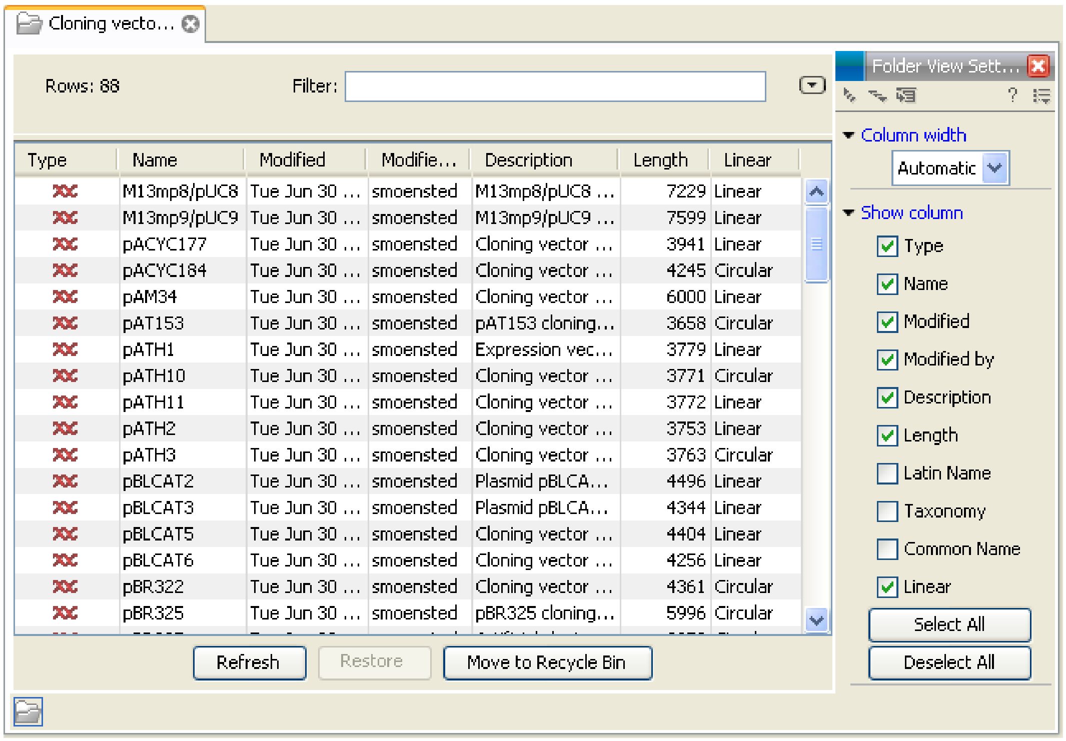
If you wish to reset the information that has been entered for an attribute, press "Clear" (written in blue next to the attribute). That will return it to the "Not set" state.

6.2.1 Show folder elements in a table

A location or a folder might contain large amounts of elements. It is possible to view their elements in the **View Area**:

select a folder or location | Show ( ) in the Toolbar | Contents ( )

An example is shown in figure 6.7.



Rows: 88 Filter:

Type	Name	Modified	Modifie...	Description	Length	Linear
	M13mp8/pUC8	Tue Jun 30 ...	smoensted	M13mp8/pUC8 ...	7229	Linear
	M13mp9/pUC9	Tue Jun 30 ...	smoensted	M13mp9/pUC9 ...	7599	Linear
	pACYC177	Tue Jun 30 ...	smoensted	Cloning vector ...	3941	Linear
	pACYC184	Tue Jun 30 ...	smoensted	Cloning vector ...	4245	Circular
	pAM34	Tue Jun 30 ...	smoensted	Cloning vector ...	6000	Linear
	pAT153	Tue Jun 30 ...	smoensted	pAT153 cloning...	3658	Circular
	pATH1	Tue Jun 30 ...	smoensted	Expression vec...	3779	Linear
	pATH10	Tue Jun 30 ...	smoensted	Cloning vector ...	3771	Circular
	pATH11	Tue Jun 30 ...	smoensted	Cloning vector ...	3772	Linear
	pATH2	Tue Jun 30 ...	smoensted	Cloning vector ...	3753	Linear
	pATH3	Tue Jun 30 ...	smoensted	Cloning vector ...	3763	Circular
	pBLCAT2	Tue Jun 30 ...	smoensted	Plasmid pBLCA...	4496	Linear
	pBLCAT3	Tue Jun 30 ...	smoensted	Plasmid pBLCA...	4344	Linear
	pBLCAT5	Tue Jun 30 ...	smoensted	Cloning vector ...	4404	Linear
	pBLCAT6	Tue Jun 30 ...	smoensted	Cloning vector ...	4256	Linear
	pBR322	Tue Jun 30 ...	smoensted	Cloning vector ...	4361	Circular
	pBR325	Tue Jun 30 ...	smoensted	pBR325 cloning...	5996	Circular

Column width: Automatic

Show column
- ✓ Type
- ✓ Name
- ✓ Modified
- ✓ Modified by
- ✓ Description
- ✓ Length
- ☐ Latin Name
- ☐ Taxonomy
- ☐ Common Name
- ✓ Linear

Figure 6.7: *Viewing the elements in a folder.*

When the elements are shown in the view, they can be sorted by clicking the heading of each of the columns. You can further refine the sorting by pressing Ctrl (⌘ on Mac) while clicking the heading of another column.

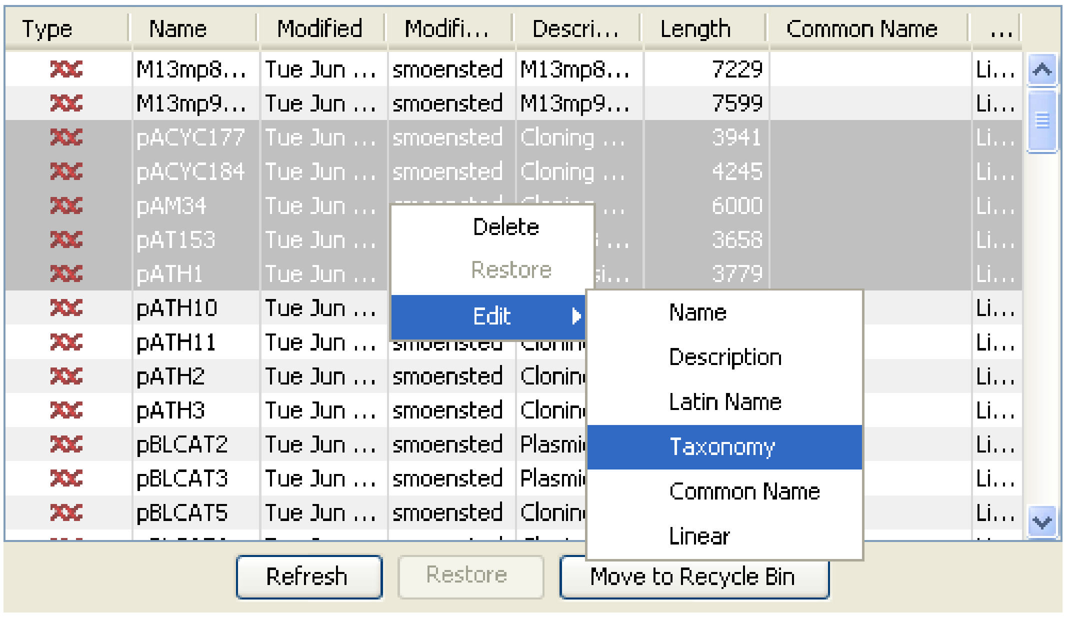
Sorting the elements in a view does not affect the ordering of the elements in the **Navigation Area**.

Note! The view only displays one "layer" at a time: the content of subfolders is not visible in this view. Also note that only sequences have the full span of information like organism etc.

Batch edit folder elements

You can select a number of elements in the table, right-click and choose **Edit** to batch edit the elements. In this way, you can change the e.g. the description or common name of several elements in one go.

In figure 6.8 you can see an example where the common name of five sequence are renamed in one go. In this example, a dialog with a text field will be shown, letting you enter a new common name for these five sequences. **Note!** This information is directly saved and you cannot



Type	Name	Modified	Modifi...	Descri...	Length	Common Name	...
	M13mp8...	Tue Jun ...	smoensted	M13mp8...	7229		Li...
	M13mp9...	Tue Jun ...	smoensted	M13mp9...	7599		Li...
	pACYC177	Tue Jun ...	smoensted	Cloning ...	3941		Li...
	pACYC184	Tue Jun ...	smoensted	Cloning ...	4245		Li...
	pAM34	Tue Jun ...		...	6000		Li...
	pAT153	Tue Jun ...		...	3658		Li...
	pATH1	Tue Jun ...		si...	3779		Li...
	pATH10	Tue Jun ...					Li...
	pATH11	Tue Jun ...					Li...
	pATH2	Tue Jun ...	smoensted	Clonin			Li...
	pATH3	Tue Jun ...	smoensted	Clonin			Li...
	pBLCAT2	Tue Jun ...	smoensted	Plasmi			Li...
	pBLCAT3	Tue Jun ...	smoensted	Plasmi			Li...
	pBLCAT5	Tue Jun ...	smoensted	Clonin			Li...

Figure 6.8: *Changing the common name of five sequences.*

undo.

6.2.2 What happens when the sequence gets outside the server data location?

Since the information entered in the **Element info** is very tightly connected to the attributes set in the data location, they will appear in a different way when it is stored in another location. When you save the sequence in another server data location with a different attribute set, or on the Workbench's own data location, the information will become "static" which means that it can't be changed and searched for. It can only be deleted. Note that attributes that were "Not set" will disappear when you go outside the data location.

If the sequence is moved back into the data location, the information will be available for editing and searching again.

6.3 Searching

When an attribute has been created, it will automatically be available for searching. This means that in the **Local Search** (), you can select the attribute in the list of search criteria (see figure 6.9).

It will also be available in the **Quick Search** below the **Navigation Area** (press Shift+F1 and it will be listed - see figure 6.10).

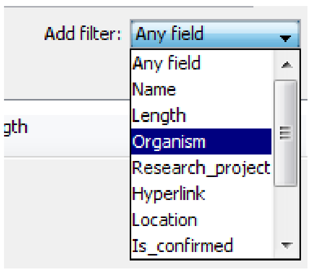


Figure 6.9: *The attributes from figure 6.4 are now listed in the search filter.*

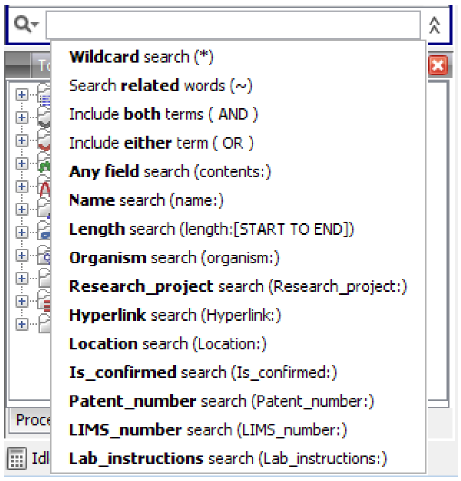

Wildcard search (*)
Search **related** words (~)
Include **both** terms (AND)
Include **either** term (OR)
Any field search (contents:)
Name search (name:)
Length search (length:[START TO END])
Organism search (organism:)
Research_project search (Research_project:)
Hyperlink search (Hyperlink:)
Location search (Location:)
Is_confirmed search (Is_confirmed:)
Patent_number search (Patent_number:)
LIMS_number search (LIMS_number:)
Lab_instructions search (Lab_instructions:)

Figure 6.10: *The attributes from figure 6.4 are now available in the Quick Search as well.*

Chapter 7

Job Distribution

The *CLC Genomics Server* has the concept of *distributing jobs to nodes*. This means that you can have a master server with the primary purpose of handling user access, serving data to users and starting jobs, and you have a number of nodes that will execute these jobs.

Two main models of this setup are available: a master server that submits tasks to dedicated execution nodes, and a master server that submits tasks to a local grid system. Figure 7.1 shows a schematic overview these possibilities, and they are described in text below.

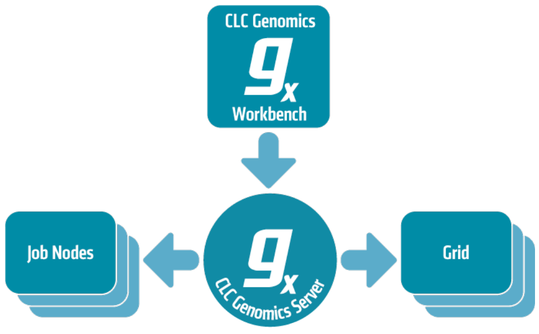


Figure 7.1: *An overview of the job distribution possibilities.*

- **Model I: Master server with execution nodes** - here, a master server submits CLC jobs directly to machines running the *CLC Genomics Server* for execution. In this setup, a group of machines (from two upwards) have the *CLC Genomics Server* software installed on them. The system administrator assigns one of them as the *master node*. This node controls the distribution of jobs. The other nodes are *execution nodes*, which carry out the computational tasks they are assigned. The execution nodes wait for the master node to assign them a job, and once the job is executed, they are available for the next job. With this set-up, the CLC master server controls the queue and the distribution of compute resources. This has the advantage of being simple to set up and maintain, with no other software required. However, it is not well suited to situations where the compute resources are shared with other systems because there is no mechanism for managing the load on the computer. This setup works best when the execute nodes are machines dedicated to running a *CLC Genomics Server*. Further details about this setup can be found in section 7.1
- **Model II: Master server submitting to grid nodes** - here the master server submits jobs to a third party job scheduler. That scheduler controls the resources on a local compuater

cluster (grid) where the job will be executed. This means that it is the responsibility of the native grid job scheduling system to start the job. When the job is started on one of the grid nodes, a **CLC Grid Worker**, which is a stand-alone executable including all the algorithms on the server, is started with a set of parameters specified by the user. Further details about this setup can be found in section 7.2.

7.1 Model I: Master server with execution nodes

The general steps to setting up this model are to:

1. Install the *CLC Genomics Server* software on all the machines involved. (See section 2.1.)
2. Start up the *CLC Genomics Server* software on all the machines involved. (See section 2.6.)
3. Install the license on the machine that will act as the master node. (See section 2.5.)
4. Log into the web adminstrative interface for the *CLC Genomics Server* of the machine that will act as the master node. (See section 3.1.)
5. Configure the Master node and the job nodes via the administrative interface on the Master node.

The only work you have to do directly on the machines that will run as job nodes is to install the *CLC Genomics Server* and start the software up on each of them. Almost all other configurations are done via the web adminstrative interface for the *CLC Genomics Server* of the master node. This includes the installation of plug-ins. See section section 7.1.4.

7.1.1 Master and job nodes explained

A machine running the *CLC Genomics Server* software can be considered to be of one of three types:

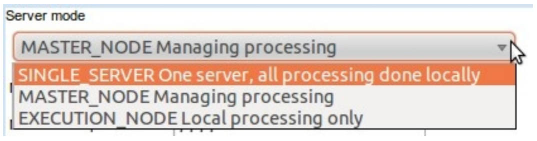
1. **Single server** - a server, to which users submit jobs, and which runs those jobs.
2. **Master node** - a machine that accepts jobs from users and then passes them to other systems - either to execution nodes (see below) or a local grid system.
3. **Execution node** - here used to describe a machine running the *CLC Genomics Server* that accepts jobs directly from a Master node.

Figure 7.2 shows the configuration options for the types of machines running the *CLC Genomics Server*.

7.1.2 User credentials on a master-job node setup

If you have a brand new installation, and you plan to use the default administrative login credentials (see section 3.1, you do not need to change anything.

If you wish to set other authentication details, then log into the web administration interface on each machine and set up *identical* authentication details on each one.



Server mode

MASTER_NODE Managing processing

SINGLE_SERVER One server, all processing done locally
MASTER_NODE Managing processing
EXECUTION_NODE Local processing only

Figure 7.2: *The configuration options for the types of machines running the* **CLC Genomics Server***. The choices of relevance under normal circumstances are Single_server and Master_node. An administrator will not usually need to manually choose the Execution Host option. This option is there primarily to allow for troubleshooting.*

You can now log out of the *CLC Genomics Server* web administrative interface on all the machines except the one that will be designated the master node. All further configuration will take place on the machine to be the master node.

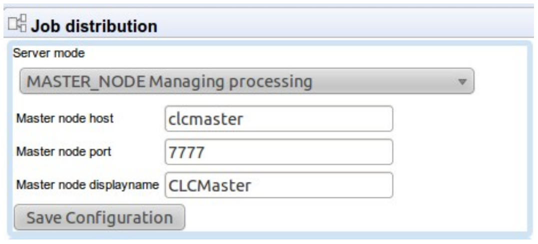
7.1.3 Configuring your setup

If you have not already, please download and install your license to the master node. (See section 2.5.) Do **not** install license files on the job nodes. The licensing information, including how many job nodes you are can run, are all included in the license on the master node.

To configure your master/execution node setup, navigate through these tabs in the web administrative interface on your master node:

Admin ( ) | Job distribution ( )

First, set the server mode to `MASTER_NODE` and provide the master node address, port and a human-readable name as shown in figure 7.3).



Job distribution

Server mode

MASTER_NODE Managing processing

Master node host	clcmaster
Master node port	7777
Master node displayname	CLCMaster

Save Configuration

Figure 7.3: *Setting up a master server.*

Next, click **Attach Node** to specify a job node. Fill in the appropriate information about the node (see figure 7.4).

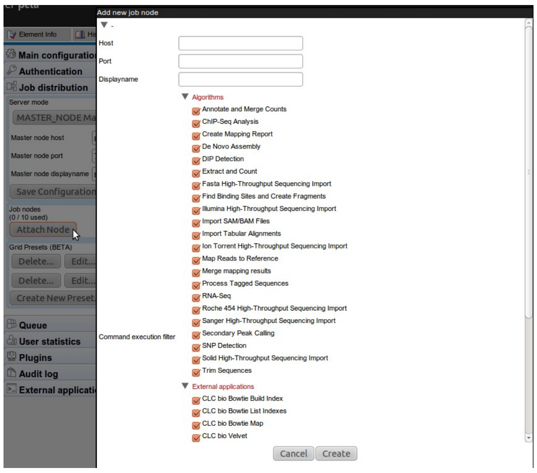
Besides information about the node hostname, port and displayname, you can also configure what kind of jobs that node is able to execute.

Repeat this process for each job node you wish to attach and click **Save Configuration** when you are done.

Once set up, the job nodes will automatically inherit all configurations made on the master node. At any time, you can log in to a job node itself via its Server administrative interface.

Note that you will get a warning dialog if there are types of jobs that are not enabled on any of the nodes.

Note that when a node has finished a job, it will take the first job in the queue that is of a type



Add new job node

Host
Port
Displayname

Command execution filter

- Algorithms
  - ✓ Annotate and Merge Counts
  - ✓ ChIP-Seq Analysis
  - ✓ Create Mapping Report
  - ✓ De Novo Assembly
  - ✓ DIP Detection
  - ✓ Extract and Count
  - ✓ Fasta High-Throughput Sequencing Import
  - ✓ Find Binding Sites and Create Fragments
  - ✓ Illumina High-Throughput Sequencing Import
  - ✓ Import SAM/BAM Files
  - ✓ Import Tabular Alignments
  - ✓ Ion Torrent High-Throughput Sequencing Import
  - ✓ Map Reads to Reference
  - ✓ Merge mapping results
  - ✓ Process Tagged Sequences
  - ✓ RNA-Seq
  - ✓ Roche 454 High-Throughput Sequencing Import
  - ✓ Sanger High-Throughput Sequencing Import
  - ✓ Secondary Peak Calling
  - ✓ SNP Detection
  - ✓ Solid High-Throughput Sequencing Import
  - ✓ Trim Sequences
- External applications
  - ✓ CLC bio Bowtie Build Index
  - ✓ CLC bio Bowtie List Indexes
  - ✓ CLC bio Bowtie Map
  - ✓ CLC bio Velvet

Figure 7.4: *Setting up a master server.*

the node is configured to process. This then means that, depending on how you have configurd your system, the job that is number one in the queue will not necessarily be processed first.

After your job nodes are all configured, file system locations should be added (see the section on Adding a file system location 3.2.1. We recommend this is done at this point as all job nodes will then inherit the configurations made on the master node.

In order to test that access works for both job nodes and the master node, you can perform the following check:

1. Create a new folder in the Workbench in the new data location (on disk, this will be done by the master node). Import some data into this folder.
2. In the Workbench, run one of the analyses in the server toolbox and choose to save the results in the new folder (this will be done by the job node)
3. Once finished, try to rename one of the result files in the Workbench (on disk, this will be done by the master node). If this works, it means that both master and job nodes are able to write to the same files.

One relatively common problem that can arise here is *root squashing*. This often needs to be disabled, because it prevents the servers from writing and accessing the files as the same user - read more about this at `http://nfs.sourceforge.net/#faq_b11`.

7.1.4 Installing Server plug-ins

Server plugin installation is described in section 2.1.

You **only** need to install the plug-in on the master server. Once installed, you should **check that the plugin is enabled** for **each job node** you wish to make available for users to run that particular task on. To do this:

- Go to the Job Distribution tab in the master nodes web administrative interface
- Click on the link to each job node
- Click in the box by the relevant task, marking it with a check mark if you wish to enable it.

7.2 Model II: Master server submitting to grid nodes

The CLC Grid Integration Tool allows jobs to be offloaded from a master server onto grid nodes using the local grid submission/queuing software to handle job scheduling and submission.

At the moment, all CLC algorithms run on a single machine; a single job is *not* run across nodes. Thus, grid nodes employed must have enough memory and space to carry out the entire requested task.

7.2.1 Requirements for CLC Grid Integration

- A functional grid submission system must already be in place. OGE and PBS Pro are the systems CLC have tested.
- The DRMAA library for the grid submission system to be used. Note that OGE comes with this library, while you will need to get the code for the PBS DRMAA library and compile this yourself. (Additional notes below.)
- The *CLC Genomics Server* must be installed on a Linux based system configured as a *submit host* in the grid environment.
- The user running the CLC Genomics Worker process is seen as the submitter of the grid job, and thus this user must exist on all the grid nodes.
- CLC Genomic Server file locations holding data that will be used must be mounted with the same path on the grid nodes as on the master Genomics Server and accessible to the user that runs the CLC Genomics Server process.
- If a CLC Bioinformatics Database is in use, all the grid nodes must be able to access that database using the user that runs the CLC Genomics Server process.
- A *CLC License Server* with one or more available *CLC Genomics Grid Worker* licenses must be reachable from the *execution hosts* in the grid setup.
- A SUN/Oracle Java Runtime environment 1.6 must be installed on all execution hosts that will be running CLC Grid Worker jobs.

DRMAA for PBS Pro: Code for this can be downloaded from http://sourceforge.net/projects/pbspro-drmaa/. When configuring the library, one has to pass an "–with-pbs=" argument, which points to the prefix of the PBS installation root. The configure script expects to be able to find lib/libpbs.a and include/pbs_ifl.h in the given root (amongst other files). SSL is needed. The configure script expects that linking with "ssl" will work, thus libssl.so must be present in one of the system's library paths. On Fedora and SuSE you will have to install openssl-devel packages to get that symlink (or create it yourself). The install procedure will install libdrmaa.so to the provided prefix (configure argument), which is the file the *CLC Genomics Server* needs to know about. The PBS DRMAA library can be configured to work in various modes as described in the README file of the pbs-drmaa source code. We have experienced the best performance, when the CLC Server has access to the PBS log files and pbs-drmaa is configured with wait_thread 1.

7.2.2 Technical overview

Figure 7.5 shows an overview of the communication involved in running a job on the grid, using OGE as the example submission system.

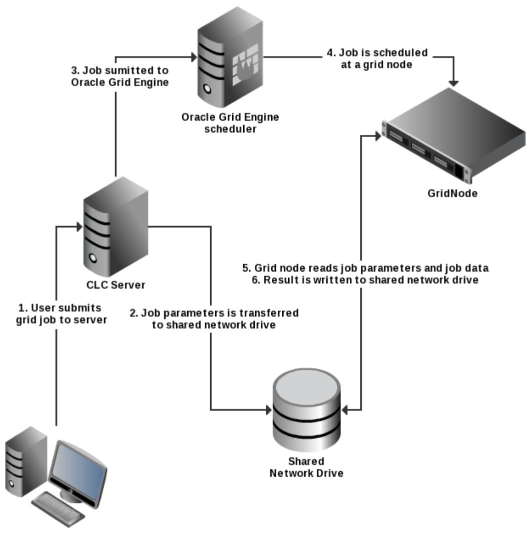

3. Job sumitted to Oracle Grid Engine
Oracle Grid Engine scheduler
4. Job is scheduled at a grid node
GridNode
CLC Server
5. Grid node reads job parameters and job data
6. Result is written to shared network drive
1. User submits grid job to server
2. Job parameters is transferred to shared network drive
Shared Network Drive

Figure 7.5: *An overview of grid integration, using OGE as the example submission system.*

The steps of this figure are in detail:

1. From the workbench the user invokes an algorithm to be run on the grid. This information is sent to the master server running the *CLC Genomics Server*.
2. The master server writes a file with job parameters to the filesystem shared with the grid execution nodes. The job parameters contain identifiers mapping to the job data placed in the CLC server data location. The job parameters file is automatically deleted when its no longer used by the grid node.
3. Now the server invokes *qsub* through the specified DRMAA native library. Then *qsub* transfers the job request to the grid scheduler. Since the user that runs the CLC Server process has invoked *qsub*, the grid node will run the job as this CLC-Server user.
4. The job scheduler will choose a grid node based on the parameters given to *qsub* and the user that invoked *qsub*.
5. The chosen grid node will retrieve the job parameters from the shared file system and start performing the given task.
6. After completion of the job, the grid node will write the results to the server's data location. After this step the result can be accessed by the Workbench user through the master server.

7.2.3 Setting up the grid integration

CLC jobs are submitted to a local grid via a special executable called **clcgridworker**. In the documentation, this executable is also referred to as the CLC Grid Worker. In general terms, four steps are taken to setup grid integration for CLC bio jobs.

- Install grid licenses
- Set up a consumable resource in the grid submissions system to record the number of available licenses
- Configure and deploy the CLC Grid Worker
- Configure the grid presets, which specify queues and other grid settings that will be applied when users submit to the grid from CLC software.

Licensing of grid workers

Generally, a pool of CLC grid licenses are purchased and are served by the license server as described below. One license is used for each CLC Grid Worker script launched. When the CLC Grid Worker starts running on a node, it will attempt to get a license from the license server. Once the job is complete, the license will be returned.

Configuring licenses as a consumable resource

Since there is a limitation on the number of licenses available, it is important that the local grid system is configured so that the number of CLC Grid Worker scripts launched is never higher than the maximum number of licenses installed. If the number of CLC Grid Worker scripts launched exceeds the number of licenses available, jobs unable to find a license will fail when they are executed.

Some grid systems support the concept of a *consumable resource*. Using this, you can set the number of CLC grid licenses available. This will restrict the number of CLC jobs launched to run on the grid at any one time to this number. Any job that is submitted while all licenses are already in use will sit in the queue until a license becomes available. **We highly recommend that CLC grid licenses are configured as a consumable resource on the local grid submission system.**

Deploy the Grid Worker

The CLC Grid Worker is a stand-alone executable that is used to perform the analyses. It is extracted from the *CLC Genomics Server* installation directory and will include any plug-ins installed on that server, as well as the built-in algorithms. The CLC Grid Worker should be deployed to a *shared file-system that is readable from all execution hosts*.

To deploy the Grid Worker, please run the following command from the *CLC Genomics Server* installation directory. The target directory must exist and be writable by the user running the deploy script.

```
./deploy-gridworker </path/to/gridworkerdir/>
```

The first time you deploy the Grid Worker you will have to setup the license server connection. Use a text editor and open the file: `settings/license.properties` in the Grid Worker directory.

The file will look something like this:

```
#License Settings
```

```
serverip=host.example.com
serverport=6200
disableborrow=false
autodiscover=false
useserver=true
```

You can leave `autodiscover=true` to use UDP-based auto discovery of the license server. However, for grid usage it is recommended that you set `autodiscover=false` and use the `serverip` property to specify the host name or IP-address of your CLC License Server.

The *CLC License Server* user manual can be found at http://clcbio.com/index.php?id=1392.

If you install additional plugins to your *CLC Genomics Server*, you will need to redploy your CLC Grid Workers to use the functionality of the new plugins via your grid.

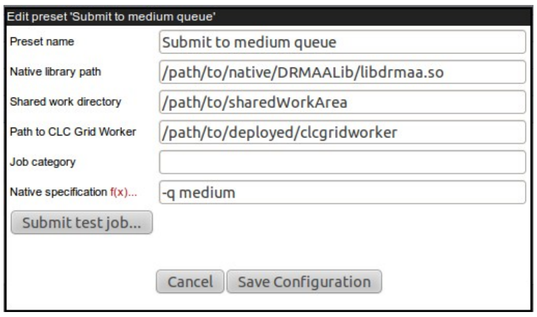
Configure grid presets

The details of the communication between the master server and the grid when submitting a job is configured using **grid presets**. The users selects a preset when starting the job as explained in section 7.2.5.

To configure the presets, log into the web-interface of the *CLC Genomics Server* on your master machine and navigate through these tabs in the web administrative interface:

Admin (⚙) | **Job distribution** (▭)

Choose the **Grid Presets** section and click the **Create New Preset** button.


Edit preset 'Submit to medium queue'

Preset name	Submit to medium queue
Native library path	/path/to/native/DRMAALib/libdrmaa.so
Shared work directory	/path/to/sharedWorkArea
Path to CLC Grid Worker	/path/to/deployed/clcgridworker
Job category	
Native specification f(x)...	-q medium

Figure 7.6: *Configuring presets.*

For each preset, the following information can be set:

Preset name The name of the preset as it will be presented in the Workbench when starting a job (see section 7.2.5) and as you will be able to refer to it when using the Command Line Tools.

Native library path The full path to the grid-specific DRMAA library.

Shared work directory The path to a directory that can be accessed by both the CLC Server and the Grid Workers. Temporary directories are created within this area during each job run. These temporary directories hold files used for communication between the CLC Server and Grid Worker.

Path to CLC Grid Worker The full path to the `clcgridworker` executable. That is, the path to the directory where you deployed the Grid Worker to, followed by the name of the script: `clcgridworker`. (See section 7.2.3).

Job category The name of the job category - a parameter passed to the underlying grid system.

Native specification List of native parameters to pass to the grid e.g. associations to specific grid queues or memory requirements (see below). Clicking on the f(x) next to Native Specification pops up a box allowing the insertion of particular variables into the Native Specification box. This is described futher below 7.2.3

Below are examples of OGE-specific arguments one might provide in the native specification field of a Grid Preset. Please see your grid scheduling documentation to determine what options are available for your scheduling system.

Example 1: To redirect standard output and error output, you might put the following in the Native specification field:

```
-o <path/to/standard_out> -e <path/to/error_out>
```

This corresponds to the following *qsub* command being generated:

```
qsub my_script -o <path/to/standard_out> -e <path/to/error_out>
```

Example 2: Use a specific OGE queue for all jobs.

```
-hard -l qname=<name_of_queue>
```

This corresponds to the following *qsub* command:

```
qsub my_script -q queue_name
```

f(x) - adding variables evaluated at run-time
Grid Presets are esentially static in nature, with most options being defined directly in the preset itself. In some cases though, it may be of interest to have variables that are evaluated at runtime. Currently, three such variables can be added to the Native Specification line:

USER_NAME The name of the user who is logged into the server and is submitting the analysis request. All grid jobs are submitted by the user that runs the CLC Server process, so this variable might be added to, for example, log usage statistics for actual users of the system, or to send an email to the an email account of a form that includes the contents of this variable. For example, the type of text that follows could be put into the Native specification field:

```
-M {USER_NAME}@yourmailserver.com
```

COMMAND_NAME The name of the *CLC Genomics Server* command to be executed on the grid by the clcgridworker executable.

COMMAND_ID The ID of the *CLC Genomics Server* command to be executed on the grid.

These variables can be added by the administrator directly into the Native Specification box, by surrounding the variable name with curly brackets. Alternatively, to ensure the proper syntax, you can click on the f(x) link and choose the variable to insert.

These variables can be used by the administrator in any way that fits with the native grid system, and that does not cause clashes with the way the CLC Server and Grid Workers communicate. For example, in cases where grid usage is closely monitored, it may be desirable to include the user name of the analysis job in metadata reports, so that computer resource time can be logged. Another example might be to set an option such as `-q {COMMAND_NAME}` if there were, for example, certain commands to be submitted to queues of the same name as the commands.

Controlling the number of cores utilized

In order to configure core usage, the native specification of the grid preset needs to be properly configured. This configuration depends on the grid system used. From version 4.01 of the CLC Genomics Server, all cores on an execution node will be used by default. Unless otherwise configured to limit the number of cores used for a job involving assembly or read mapping phases, a dedicated queue must then be setup, which only schedules a single job on any given machine at a time. Otherwise your CLC jobs may conflict with others running on the same execution host at the same time.

Configuration of OGE/SGE

1) CPU Core usage when not using parallel environment

By default the CLC Genomics Servers ignores the number of slots assigned to a grid job, and utilizes all cores of the execution host. That is, jobs will run on all cores of a execution host.

As of version 4.01 of the CLC Genomics Server, there is an environmental variable, which, when set to 1, will specify that the number of allocated slots should be interpreted as the maximum number of cores a job should be run on. To set this environmental variable, add the following to the native specification of the grid preset:

```
-v CLC_USE_OGE_SLOTS_AS_CORES=1
```

In this case, the number of utilized cores is equal to the number of slots allocated by OGE for the job.

2) Limiting CPU core usage by utilizing parallel environment

The parallel environment feature can be used to limit the number of cores used by the CLC Genomics Server, when running jobs on the grid. The syntax in the native specification for using parallel environments is:

```
-pe $PE_NAME $MIN_CORE-$MAX_CORE
```

When the parallel environments feature is used, the number of allocated slots is interpreted as the number of cores to be used. That is, the number of utilized cores is equal to the number of slots in this case.

The parallel environment, selected by its name, must be setup by the grid administrator (documentation provided by Oracle will cover this subject area), in such a way that the number of slots corresponds to the number of cores. `$MIN_CORE` and `$MAX_CORE` specify a range of cores, which the jobs submitted through this grid preset can run under. Care must be taken not to set `$MIN_CORE` too high, as the job might never be run (e.g. if there is no system with that many cores available), and the submitting user will not be warned by this fact.

An example of a native specification using parallel environments is the following:

`-l cfl=1 -l qname=32bit -pe clc 1-3.`

Here, the *clc* parallel environment is selected, and 1 to 3 cores are requested.

Older versions of the CLC Genomics Server

CLC Genomics Server version 4.0 and older utilize CPU cores equal to the number of allocated slots, unless a parallel environment is in use, in which case the behaviour is the same as described previously. In many situations the number of allocated slots is 1, effectively resulting in CLC jobs running on one core only.

Configuration of PBS Pro

With PBS Pro it is not possible to specify a range of cores (at least not to our knowledge). Here one specifies exactly how many cores are needed. This request can be granted (the process is scheduled) or denied (the process is not scheduled). It is thus very important to choose a realistic number. The number of cores are requested as a resource: `-l nodes=1:ppn=X`, where X is the number of cores. As this resource is also designed to work with parallel system, the number of nodes is allowed to be larger than 1. For the sake of scheduling cores, it is vital that this parameter is kept at 1. An example of a native specification is: `-q bit32 -l nodes=1:ppn=2`, which will request two cores and be queued in the *bit32* queue.

Other grid worker options

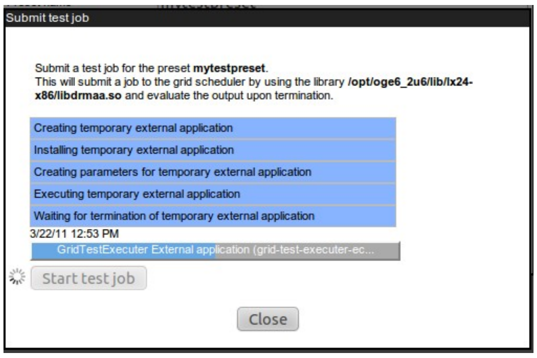
Additional java options can be set for grid workers by creating a file called `clcgridworker.vmoptions` in the same folder as the *deployed* `clcgridworker` script. For example, if such a file was created, containing the following two lines, it would set memory limits for the java process and a temporary directory for the grid nodes, overriding the defaults that would otherwise apply:

```
-Xmx1000m
-Djava.io.tmpdir=/path/to/tmp
```

If more than one clcgridworker is deployed, then a `clcgridworker.vmoptions` file can be created for each one.

7.2.4 Testing a Grid Preset

There are two types of tests that can be run to check a Grid Preset. The first runs automatically whenever the *Save Configuration* button in the Grid Preset configuration window is pressed. This is a basic test that looks for the native library you have specified. The second type of test is optional, and is launched if the *Submit test job...* button is pressed. This submits a small test job to your grid and the information returned is checked for things that might indicate problems with the configuration. While the job is running, a window is visible highlighting the jobs progression as shown in figure 7.7.



Submit a test job for the preset **mytestpreset**.
This will submit a job to the grid scheduler by using the library **/opt/oge6_2u6/lib/lx24-x86/libdrmaa.so** and evaluate the output upon termination.

```
Creating temporary external application
Installing temporary external application
Creating parameters for temporary external application
Executing temporary external application
Waiting for termination of temporary external application
3/22/11 12:53 PM
GridTestExecuter External application (grid-test-executer-ec...
```

Figure 7.7: *Testing a Grid Preset.*

7.2.5 Client-side: starting CLC jobs on the grid

Installing the CLC Grid Integration Client Plug-in

To submit jobs to the grid from within the *CLC Genomics Workbench*, users must first install the CLC Grid Integration Client Plug-in. This plug-in can be downloaded from the CLC Genomics Server download site: http://www.clcbio.com/download_genomics_server/.

Plug-ins are installed using the plug-in manager[1]:

Help in the Menu Bar | Install Plug-ins ( )

or **Plug-ins ( ) in the Toolbar**

After you have downloaded the CLC Grid Integration Client Plug-in, install it by clicking the **Install from File** button at the bottom of the dialog. This will open a dialog where you can browse for the plug-in file.

You need to restart the Workbench before the plug-in is ready for use.

Starting grid jobs

Once the server side is configured, and the CLC Grid Integration Client Plug-in has been installed in the *CLC Genomics Workbench*, an extra option will appear in the first dialogue presented when setting up a task that could be executed on the *CLC Genomics Server*. Users will be able to choose to execute such a task on their local machine, the CLC Server machine, or using any available grid presets. To submit to the grid is as simple as choosing from among the grid presets in the drop down box. See figure 7.8.

7.2.6 CLC Grid integration tool installation checklist

Below is a checklist of the configuration steps necessary to set up CLC grid integration:

1. Set up licensing of grid workers, section 7.2.3
2. Configure CLC grid licenses as a consumable resource in the local grid system.
3. Deploy the Grid Worker, section 7.2.3

[1]In order to install plug-ins on Windows Vista or Linux, the Workbench must be run in administrator mode. To do this in Windows Vista, right-click the program shortcut and choose "Run as Administrator". Then follow the procedure described. For Linux, run the Workbench as an administrative user, or using sudo.

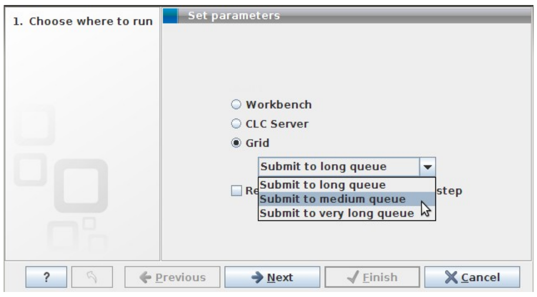


Figure 7.8: *Starting the job on the grid.*

4. Configure grid presets, section 7.2.3
5. Install the CLC Grid Integration Client Plug-in into the Workbench, section 7.2.5
6. Test your setup by submitting some small tasks to the grid via a *CLC Genomics Server* client such as the CLC Genomics Workbench or the Command Line Tools. Ideally, these would be tasks already known to run smoothly directly on the *CLC Genomics Server*.

7.2.7 Grid Integration Tips

If you are having problems with your CLC Grid Integration, please check the following points:

- Does your system meets the requirements of the CLC Grid Integration Tool 7.2.1? For example, please check that the machine the *CLC Genomics Server* is running on is configured as a submit host for your grid system, and please check that you are running Sun/Oracle Java 1.6 on all execution hosts.
- The user running the *CLC Genomics Server* process is the same user seen as the submitter of all jobs to the grid. Does this user exist on your grid nodes? Does it have permission to submit to the necessary queues, and to write to the shared directories identified in the Grid Preset(s) and any `clcgridworker.vmoptions` files?
- Are your CLC Genomic Server file locations mounted with the same path on the grid nodes as on the master Genomics Server and accessible to the user that runs the CLC Genomics Server process?
- If you store data in a database, are all the grid nodes able to access that database, using the user account of the user running the *CLC Genomics Server* process?
- If you store data in a database, did you enter a machine name in the Host box of the Database Location field when seeting up the Database Location using the *CLC Genomics Server* web administration form? In particular, a generic term such as `localhost` will not work, as the grid nodes will not be able to find the host with the database on it using that information.
- If you installed the *CLC Genomics Server* as root, and then later decided to run it as a non-privileged user, please ensure that you stop the server, recursively change ownership on the *CLC Genomics Server* installation directory and any data locations assigned to the CLC Server. Please restart the server as the new user. You may need to re-index your CLC data locations (section 3.2.2) after you restart the server.

- Did you remember to redeploy the grid worker after a plugin was added to the server.
- Is your java binary on the PATH? If not, then either add it to PATH, or edit the clcgridworker script that is used to deploy the grid workers, setting the JAVA variable to the full path of your java binary. This script will be under your *CLC Genomics Server* installation directory, with the relative path from this location: `gridres/dist/clcgridworker`.
- Is the `SGE_ROOT` variable set early enough in your system that it is included in the environment of services? Alternatively, did you edit the Genomics Server startup script to set this variable? If so, the script is overwritten on upgrade - you will need to re-add this variable setting, either to the startup script, or system wide in such a way that it is available in the environment of your services.
- Is your java 64 bit, while your DRMAA library is 32 bit, or vice versa? These two things must be either both for 64 bit systems or both for 32 bit systems.

7.2.8 Understanding memory settings

Most work done by the CLC Genomics Server is done via its java process. However, particular tools involving de novo assemlbly or mapping phases (e.g. read mappings, RNA-seq analyses, smallRNA analyses, etc.) use C binaries for the computational phases.

Java process

For the grid worker **java process**, if there is a memory limit set in your clcgridworker.vmoptions file, this is the value that will be used. See section 7.2.3.

If there is no memory setting in your grid worker's clcgridworker.vmoptions file, then the following sources are referred to, in the order stated. As soon as a valid setting is found, that is the one that will be used:

1. Any virtual memory settings given in the grid preset, or if that is not set, then
2. Any physical memory settings given in the grid preset, or if that is not set, then
3. Half of the total memory present, with 50GB being the maximum set in this circumstance.

Please note that any nodes running a 32 bit operating system will have a maximum memory allocation for the java process of 1.2GB by default.

C binaries

For the computationally intensive tools that include a phase using a C binary, (e.g. de novo assemlbly, and jobs involving mapping phases (e.g. read mappings, RNA-seq analyses, smallRNA analyses, etc.)), the C binary phase is not restricted by the amount of memory set for the java process. For this reason, we highly recommend caution if you plan to submit more jobs of these types to nodes that are being used simultaneously for other work.

Chapter 8

BLAST

The *CLC Genomics Server* supports running BLAST jobs. Users will be able to submit BLAST jobs to be run on the Genomics Server; they will be able to select data from Server **data locations** (see section 3.2.1) to search against other sequences held in Server data locations, or against BLAST databases stored in an area configured as an **import/export directory** (see section 3.3).

From a Workbench user's perspective, they will see two categories of data to search against when they are setting up their BLAST job. These are shown in figure 8.1:

- Select from among the databases located on the Server. (This option is described in detail below.)
- Select sequences from the Navigation Area to search against. Here, users choose from data stored in a Server data location. A **temporary database** is then created from that data in your **Server temp area** (see section 3.5). Once the BLAST job is complete, temporary blast databases are deleted.

8.1 Adding directories for storing BLAST databases

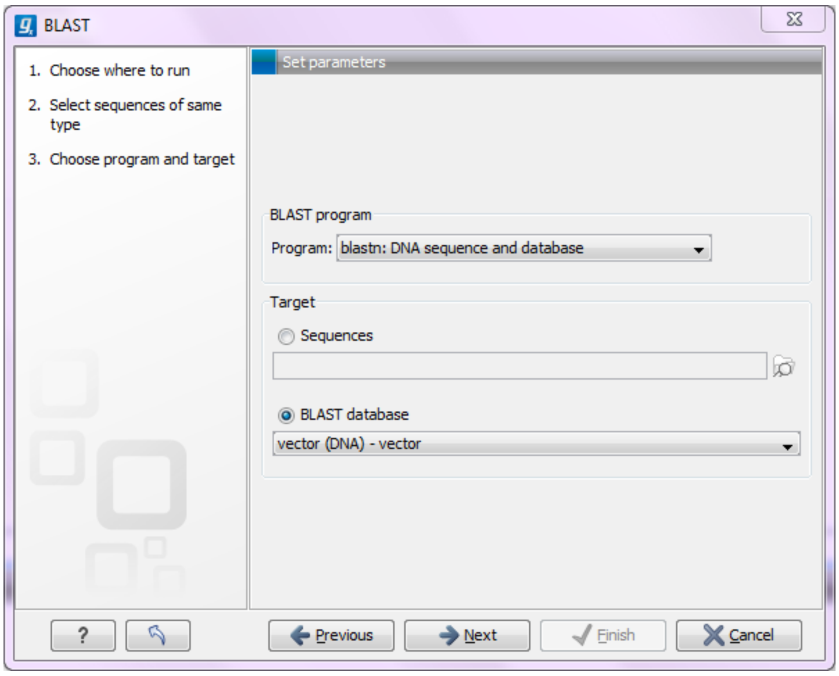
In the web interface of the server, you can configure your Server for BLAST databases:

Admin ( ) | BLAST ( )

Here, you can add a folder where you want the Server to look for BLAST databases. Any folder used for BLAST databases accessible to the *CLC Genomics Server* must already have been configured as an **import/export directory** (see section 3.3). This is a different situation than for other algorithms, where the data is stored in **data locations**. This difference is because BLAST databases are not truly CLC data, and thus are stored outside data locations specified for CLC data. They still need to be stored somewhere accessible to the Genomics Server process though, hence the need to put them in a directory configured as an import/export directory.

Clicking **Edit BLAST Database Locations** will bring up a dialog as shown in figure 8.2 where you can select which of the import/export directories you wish to use for storing BLAST databases.

Once added as a BLAST Database Location, the *CLC Genomics Server* will search this directory for any BLAST databases and list them under the BLAST tab in the web interface (see a section



BLAST program

Program: blastn: DNA sequence and database

Target

Sequences

BLAST database

vector (DNA) - vector

Figure 8.1: *Selecting database to BLAST against.*

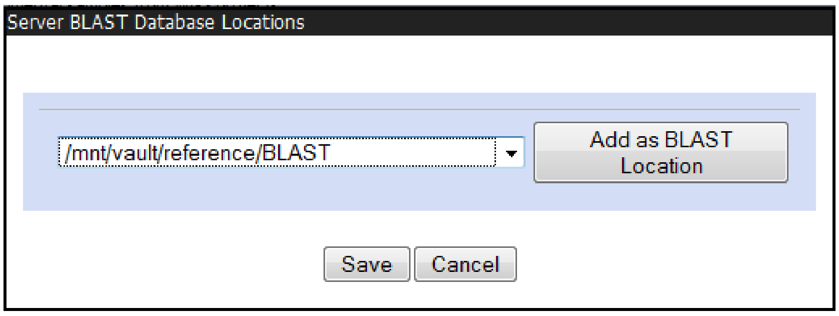
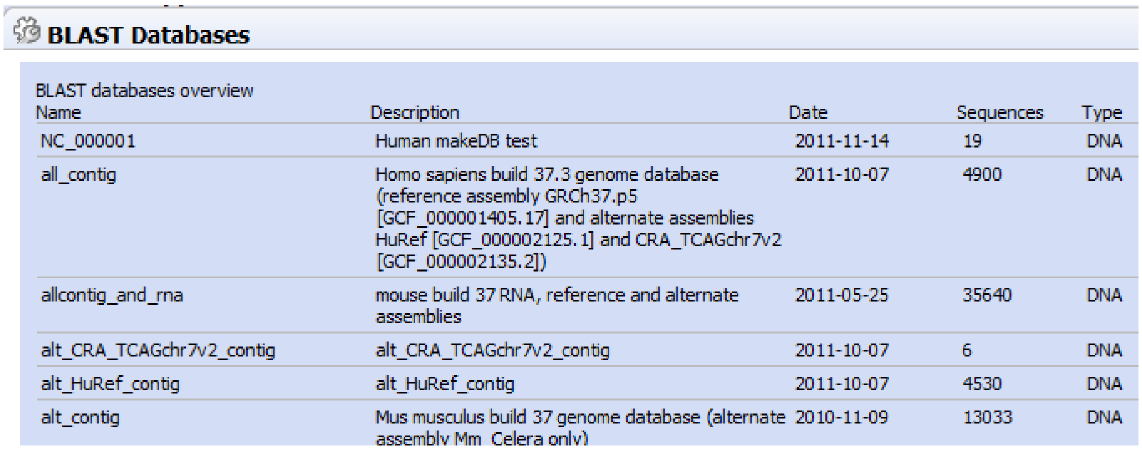


Figure 8.2: *Adding import/export directories as BLAST database locations.*

of this as an example in figure 8.3).

This overview is similar to the one you find in the Workbench BLAST manager for local databases including the following in formation:

- **Name.** The name of the BLAST database.
- **Description.** Detailed description of the contents of the database.
- **Date.** The date the database was created.
- **Sequences.** The number of sequences in the database.
- **Type.** The type can be either nucleotide (DNA) or protein.
- **Total size (1000 residues).** The number of residues in the database, either bases or amino acid.
- **Location.** The location of the database.



BLAST Databases

BLAST databases overview

Name	Description	Date	Sequences	Type
NC_000001	Human makeDB test	2011-11-14	19	DNA
all_contig	Homo sapiens build 37.3 genome database (reference assembly GRCh37.p5 [GCF_000001405.17] and alternate assemblies HuRef [GCF_000002125.1] and CRA_TCAGchr7v2 [GCF_000002135.2])	2011-10-07	4900	DNA
allcontig_and_rna	mouse build 37 RNA, reference and alternate assemblies	2011-05-25	35640	DNA
alt_CRA_TCAGchr7v2_contig	alt_CRA_TCAGchr7v2_contig	2011-10-07	6	DNA
alt_HuRef_contig	alt_HuRef_contig	2011-10-07	4530	DNA
alt_contig	Mus musculus build 37 genome database (alternate assembly Mm_Celera only)	2010-11-09	13033	DNA

Figure 8.3: *Selecting database to BLAST against.*

8.2 Adding and removing BLAST databases

Databases can be added in two ways:

- Place pre-formatted databases in the directory selected as BLAST database location on the server file system. The *CLC Genomics Server* will automatically detect the database files and list the database as target when running BLAST. You can download pre-formatted database from e.g. `ftp://ftp.ncbi.nih.gov/blast/db/`.
- Run the **Create BLAST Database** () tool via your Workbench, and choose to run the function on the Server when offered the option in the Workbench Wizard. You will get a list of the BLAST database locations that are configured on your Server. The final window of the wizard offers you a location to save the output to. The output referred to is the log file for the BLAST database creation. The BLAST databases themselves are stored in the designated BLAST database folder you chose earlier in the setup process.

A note on permissions: To create BLAST databases on the Server, using the Workbench interface, the user **running the CLC Genomics Server process** must have file system level write permission on the import/export directory that you have configured to hold BLAST database.

By default, if you do not change any permissions within the CLC Genomics Server, all users logging into the CLC Genomics Server (e.g. via their Workbench, or via the Command Line Tools), will be able to create BLAST databases in the areas you have configured to hold BLAST databases.

If you wish to restrict the ability to create BLAST databases to these areas completely, but still wish your users to be able to access the BLAST databases to search against, then set the file system level permissions on the import/export directory so they are read-only.

When listing the databases as shown in figure 8.3, it is possible to delete the databases by clicking the **Delete** link in the right-hand side.

Chapter 9

External applications

Command-line applications on the server machine can easily be made available via the graphical menu system of the Workbench. Such third-party applications can then be run via the normal graphical menu system of CLC Genomics Workbenches that are connected to the *CLC Genomics Server*. These tools can access data on the machine the CLC Workbench is installed on, data stored on the *CLC Genomics Server* or data stored in areas of the server accessible to the *CLC Genomics Server*, depending on choices made by the server administrator. The third party programs are executed on the server, not the local Workbench, giving the administrator full control over the execution environment.

The integration of third party external applications is configured in the *CLC Genomics Server* administrative web interface.

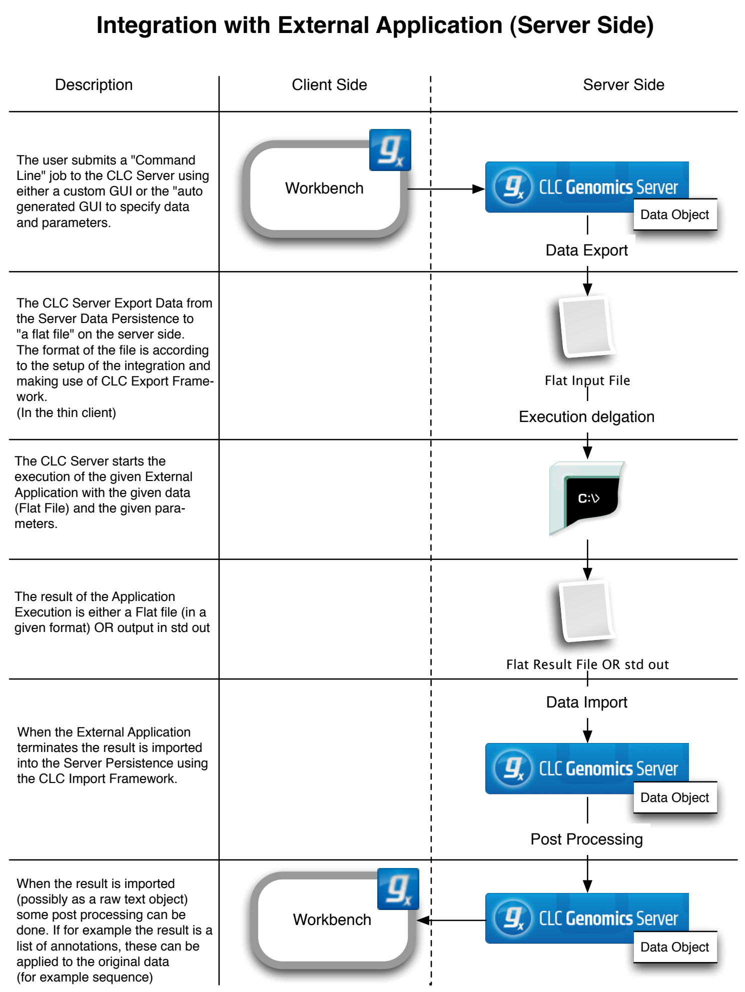
A special plug-in needs to be installed in the client Workbench to give the end-user access to the configured third-party tools. Please contact support@clcbio.com to get this Workbench plug-in.

Figure 9.1 shows an overview of the actions and data flow that occur when an integrated external applications is executed via the CLC Workbench.

In general terms the basic work flow is:

1. The user selects input data and parameters and starts the job from the Workbench
2. The server exports the input data to a temporary file
3. The server starts the command line application, using the parameters specified from the user and the temporary file as input
4. When the command line application is done, the server imports output data back into the CLC environment, saving it in the data location on the server
5. The user is notified that the job is done, and the results are available for viewing and further analysis in the Workbench

Note that all files used and files created and saved are within the CLC environment. Temporary files are created outside the CLC environment during the execution of a third party tool, but are deleted after the process runs under normal circumstances.



The user submits a "Command Line" job to the CLC Server using either a custom GUI or the "auto generated GUI to specify data and parameters.

The CLC Server Export Data from the Server Data Persistence to "a flat file" on the server side. The format of the file is according to the setup of the integration and making use of CLC Export Frame-work. (In the thin client)

The CLC Server starts the execution of the given External Application with the given data (Flat File) and the given para-meters.

The result of the Application Execution is either a Flat file (in a given format) OR output in std out

When the External Application terminates the result is imported into the Server Persistence using the CLC Import Framework.

When the result is imported (possibly as a raw text object) some post processing can be done. If for example the result is a list of annotations, these can be applied to the original data (for example sequence)

Figure 9.1: *An overview of the external applications integration.*

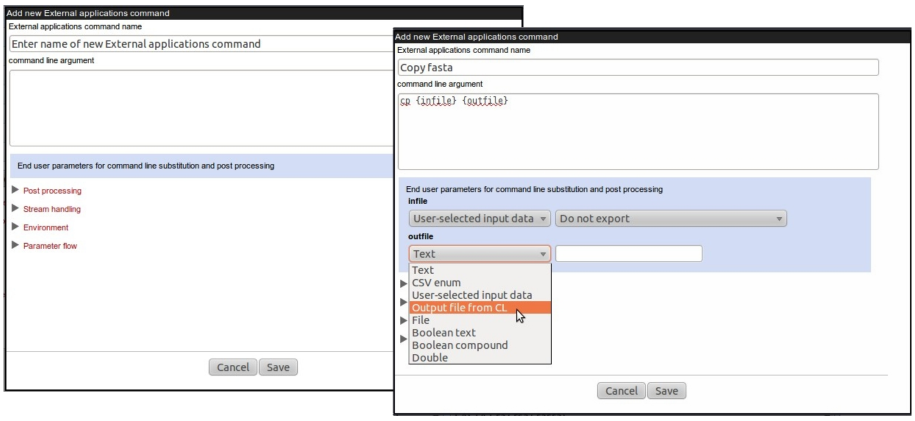
The best way to describe the integration of third party command lines tools is through a series of examples. We start with a very basic example and work towards more complex setups.

9.1 External applications integration: Basic configuration

Many aspects of configuring external tools in the *CLC Genomics Server* can be described as we set up a very simple command. We have chosen the *cp* command as will already be on your server.

The *cp* command requires at minimum two parameters, an input file and an output file. These parameters are positional: the first filename given after the command is the input file, the second is the output file. Here we will just copy a fasta file from one place in the *CLC Genomics Server* to another. This is a very inefficient way of doing this task, but it will illustrate how to integrate a command line tool without requring you to install additional software on your system.

Under the External Applications tab of the *CLC Genomics Server* administrative web interface, click on the *New configuration* button. This brings up a window like that shown at the left side of figure 9.2. In the text box labeled *External applications command name*, enter a name for this command. This will be what the end-user sees in the menu option they will be presented with via the Workbench. In the text box labeled *command line argument*, provide the command line. Start with the command, and within curly brackets, include any parameter that needs to be configured by the user. The names of the parameters inside the curly brackets will become the labels of the choices offered to the end-user when they start up this external application via their Workbench. In the right hand side of figure 9.2, we show how this looks if we give the *cp* command two parameters: *infile* and *outfile*.


Add new External applications command
External applications command name
Enter name of new External applications command
command line argument
End user parameters for command line substitution and post processing

Add new External applications command
External applications command name
Copy fasta
command line argument

```
cp {infile} {outfile}
```

End user parameters for command line substitution and post processing
infile
User-selected input data | Do not export
outfile
Text

Figure 9.2: *Setting up the cp command as an external application.*

Two drop-down menus have now appeared in the blue shaded area of the right hand window in 9.2. These are dynamically generated. Each parameter you enter in curly brackets in the command text box area will have a drop-down menu created for it. The text you entered within the curly brackets is used to label the entries in the administrative interface, and are also the labels used in the end-user interface presented via the Workbench.

The administrator now chooses the type of data each parameter will refer to. The options are:

- Text - the users is presented with a text box allowing them to enter a parameter value. The administrator can provide a default value if desired.
- CSV enum - this allows the administrator to set a drop down list of parameter choices for the user. For an example of this, please see section 9.6 on setting up Velvet as an external application.
- User-selected input data - users will be prompted to select an input file from those they have stored on the CLC Server.
- Output file from CL - users will be prompted for a location to store a file that is created by the third-party application. And extra text box is also provided in the configuration so the administrator can specify a default name for the re-imported file. If not filename is provided by the administrator, the basename of the file from the system is used.
- File - users can select an input file from their local machine's filesystem.
- Boolean text - This generates a checkbox in the end-user interface, labelled with the text you provide. If the user clicks in the box, the parameter is set to true; an empty box means the parameter is set to false.
- Boolean compound - this enables the creation of a checkbox, where if checked, the end-user is presented with another option (of your choice). If the check box is not checked, then

that option will be greyed out. Here, the administrator can also choose if the box is to be checked or unchecked by default in the Workbench interface.

- Double - Allows the user to enter a number. The administrator can choose a number this option should be set to by default. If none is set, then 0 is the default.

In the right hand side of figure 9.2 we set the parameters so that the input file to be sent to the system's copy command will be specified by the user, and we tell the system to export this file from the CLC Server as a fasta file. We then configure the import of output file from the copy command back into the CLC Server, and specify that we are importing a fasta file. When configuring standard bioinformatics third party applications, you are able to choose from many standard formats to export from, and to import back into, the *CLC Genomics Server*.

Once the configuration is complete and has been saved, the external application should now appear in the list in the administrative web interface.

The small checkbox to the left of the external application name should be checked. This means it is will be accessible to those with the Workbench plug-in installed. If a particular external application needs to be removed from end-user access for a while, this small box can just be unchecked.

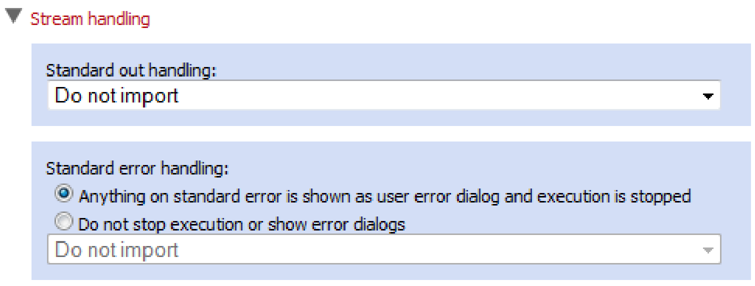
9.2 External applications integration: Post-processing

This section is still being written.

9.3 External applications integration: Stream handling

There is also a general configuration of stream handling available.

The stream handling shown in figure 9.3 allows you to specify where standard out and standard error for the external application should be handled.


Stream handling

Standard out handling:
Do not import

Standard error handling:
[✓ Anything on standard error is shown as user error dialog and execution is stopped]
Do not stop execution or show error dialogs
Do not import

Figure 9.3: *Stream handling.*

Basically, you can choose to ignore it, or you can import it using one of the importers available on the server. For some applications, standard out produces the main result, so here it makes sense to choose an appropriate importer. But also for debugging purposes it can be beneficial to import standard out and standard error as text so that you can see it in the Workbench after a run.

9.4 External applications integration: Environment

9.4.1 Environmenal Variables

This section is still being written.

9.4.2 Working directory

Define the area where temporary files will be stored. The *Default temp-dir* option uses the directory specified by the java.io.tmpdir setting for your system. The *Shared temp-dir* option allows you to set one of the directories you have already specified as an *Import/export directory* as the area to be used for temporary files created.

Choosing the Shared temp-dir option means that temporary files created will be accessible to all execution nodes and the master server, without having to move them between machines. In contrast, for a setup with CLC execution nodes that chooses the Default temp-dir setting, where such a directory is usually not shared between machines, files will be moved between the master and job node. The Default temp-dir setting will not work for any setup where the CLC Grid Worker will be used to submit jobs to a local grid.

If you work on a single server, then the Shared temp-dir setting can be used to specify a non-default area for temporary files.

If you work on a master-execution node setup, whether it be grid nodes or CLC execution nodes, the *Shared temp-dir* must be chosen, and this area must:

- Be configured in the Import/Export directories area under the Main Configuration tab
- Be a shared directory, accessible to your master server and all execution nodes

9.4.3 Execute as master process

The checkbox to **Execute as master process** can be checked if the process does not involve intensive processing on the server side. Setting this means that the process will always be run on the Master server. For a single server setup, this has no effect. However, in a system with execution nodes, checking this option results in the queue being effectively by-passed, as the job will be run directly on the master server. This choice will usually only make sense for tasks that require little RAM or cpu. Jobs run this way should will not actually block the queue - they just run as a master process. me.

9.5 External applications integration: Parameter flow

To aid in determining how the various parameters configured flow through the External Applications process, you can access small graphs showing how the parameters entered in the user parameters section will be used. You can see an example for the very simple copy command used as an example above in figure 9.4.

.

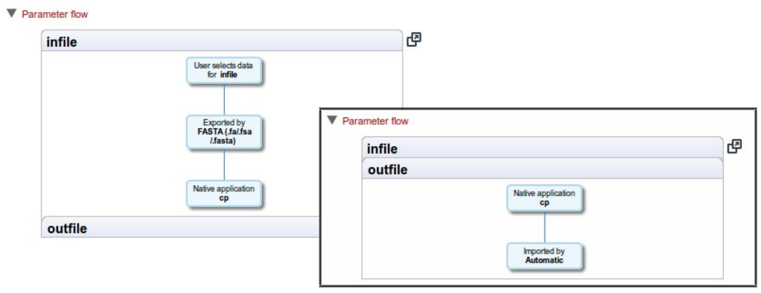


Figure 9.4: *An example of the parameter overview facility.*

9.6 External applications integration: Velvet

Velvet [Zerbino and Birney, 2008] is a popular de novo assembler for next-generation sequencing data. We have provided example scripts and configurations to set this up as an external application on *CLC Genomics Server*.

The velvet package includes two programs that need to be run consecutively. Because the external application on the *CLC Genomics Server* is designed to call one program, a script is needed to encapsulate this.

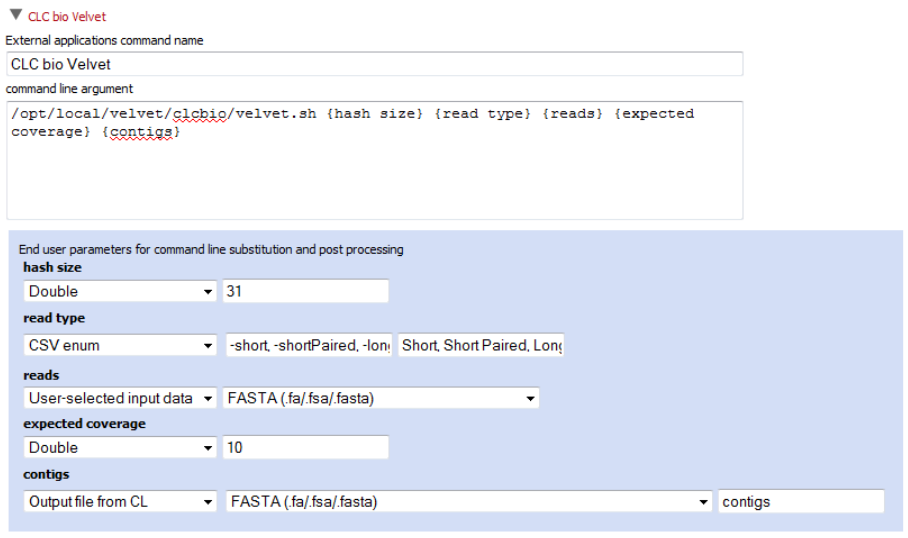
9.6.1 Installing Velvet

To get started, you need to do the following:

- Install Velvet on the server computer (download from `http://www.ebi.ac.uk/~zerbino/velvet/`). Note that if you have job nodes, it needs to be installed on all nodes that will be configured to run Velvet. We assume that Velvet is installed in `/usr/local/velvet` but you can just update the paths if it is placed elsewhere.
- Download the scripts and configuration files made by CLC bio from `http://www.clcbio.com/external-applications/velvet.zip`
- Unzip the file and place the `clcbio` folder and contents in the Velvet installation directory (this is the script tying the two Velvet program together). You need to edit the script if you did not place the Velvet binary files in `/usr/local/velvet`.
- Make sure execute permissions are set on the script and the executable files in the Velvet installation directory. Note that the user executing the files will be the user who started the Server process (if you are using the default start-up script, this will be *root*).
- Use the `velvet.xml` file as a new configuration on the server: Log in to the server via the web interface and go to the **External applications** (>_) tab under **Admin** (⚙) and click **Import Configuration**.

When the configuration has been imported, click the **CLC bio Velvet** header and you should see a configuration as shown in figure 9.5.

Update the path to the Velvet installation at the very top if necessary.



▼ CLC bio Velvet

External applications command name

CLC bio Velvet

command line argument

```
/opt/local/velvet/clcbio/velvet.sh {hash size} {read type} {reads} {expected
coverage} {contigs}
```

End user parameters for command line substitution and post processing

hash size		
Double	31	
read type		
CSV enum	-short, -shortPaired, -lon	Short, Short Paired, Lon
reads		
User-selected input data	FASTA (.fa/.fsa/.fasta)	
expected coverage		
Double	10	
contigs		
Output file from CL	FASTA (.fa/.fsa/.fasta)	contigs

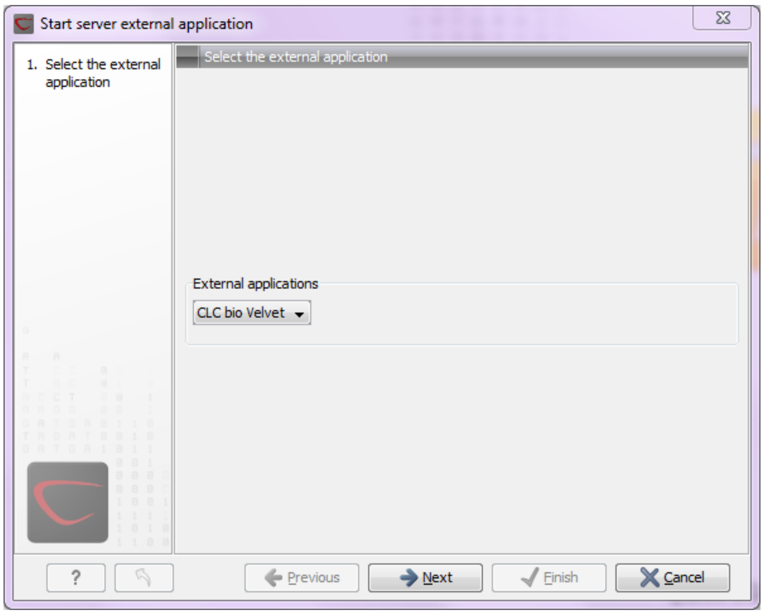
Figure 9.5: *The Velvet configuration has been imported.*

9.6.2 Running Velvet from the Workbench

Next step is to test if it can actually be executed. Open the Workbench with the **External Applications Client Plug-in** installed. Go to:

Toolbox | **CLC Server** () | **External Applications** ()

You will now see a list of all the applications that have been set up (figure 9.6).



Start server external application

1. Select the external application

Select the external application

External applications

CLC bio Velvet

Figure 9.6: *Running Velvet from the Workbench.*

In this case there is only one. When you click **Next**, you can select () some sequences, set a few parameters and click **Next** and **Finish**.

The process that follows has three steps:

1. The sequencing reads are exported by the server to a fasta file. The fasta file is a temporary

file that will be deleted when the process is done.

2. The velvet script is executed using this fasta file and the user-specified parameters as input.
3. The resulting output file is imported into the save location specified in the save step of the Workbench dialog, and the user is notified that the process is done.
4. All temporary files are deleted

9.6.3 Understanding the Velvet configuration

We will now explain how the configuration that we made actually works. And hopefully this will make it possible for you to design your own integrations.

Going back to figure 9.5, there is a text field at the top. This is where the command expression is created, in this case:

```
    /opt/local/velvet/clcbio/velvet.sh {hash size} {read type}
            {reads} {expected coverage} {contigs}
```

The first is the path to the script, and the following are parameters that are interpreted by the server when calling the script because they are surrounded by curly brackets { }. Note that each parameter entered in curly brackets gets an entry in the panel below the command line expression.

The first one, `hash size`, can be entered as a **Double** (which is a number in computer parlance) and it is thus up to the user to provide a value. A default value is entered here in the configuration (31).

The second one is the `read type` which has been configured as a **CSV enum** which is basically a list. The first part consists of the parameters to be used when calling the script (`-short, -shortPaired, -long, -longPaired`), and the second part is the more human-readable representation that is shown in the Workbench (`Short, Short Paired, Long, Long Paired`).

The third parameter is `reads` which is the input data. When the **User-selected input data** option is chosen, a list of all the available export formats is presented. In this case, Velvet expects a fasta file. When a user starts Velvet from the Workbench, the server starts exporting the selected input data to a temporary fasta file before running the script.

The `expected coverage` is similar to hash size.

The last parameter is `contigs` which represents the output file. This time, a list of import data formats is available used to import the data back into the folder that the user selected as save destination.

The rest of the configurations listed below are not used in this example, see the Bowtie example below 9.7.

9.7 External applications integration: Bowtie

Bowtie [Langmead et al., 2009] is a short-reads mapper that can map sequencing reads to a reference sequence. In this example, we show how to make an integration of the mapper where the users selects sequencing reads and sets a few parameters, and then Bowtie is executed using a pre-built index of the reference genome. Because Bowtie needs an indexed reference genome, the selection of a reference sequence as you know from CLC bio's own read mapper is not part of the integration described as an example here.

This means you can either make a list of pre-built index files, or you can use the integration of the indexing tool so that users will be able to work on new organisms that they provide a reference for themselves. These circumstances complicate the explanation a little bit, so for the first part we assume index files are in place.

9.7.1 Installing Bowtie

To get started:

- Install Bowtie from `http://bowtie-bio.sourceforge.net/index.shtml`. We assume that Bowtie is installed in `/usr/local/bowtie` but you can just update the paths if it is placed elsewhere.
- Download the scripts and configuration files made by CLC bio from `http://www.clcbio.com/external-applications/bowtie.zip`
- Place the `clcbio` folder and contents in the Bowtie installation directory. This is the script used to wrap the Bowtie functionality.
- Make sure execute permissions are set on the scripts and the executable files in the Bowtie installation directory. Note that the user executing the files will be the user who started the Server process (if you are using the default start-up script, this will be *root*).
- Use the `bowtie.xml` file as a new configuration on the server: Log in to the server via the web interface and go to the **External applications** (>_) tab under **Admin** () and click **Import Configuration**.

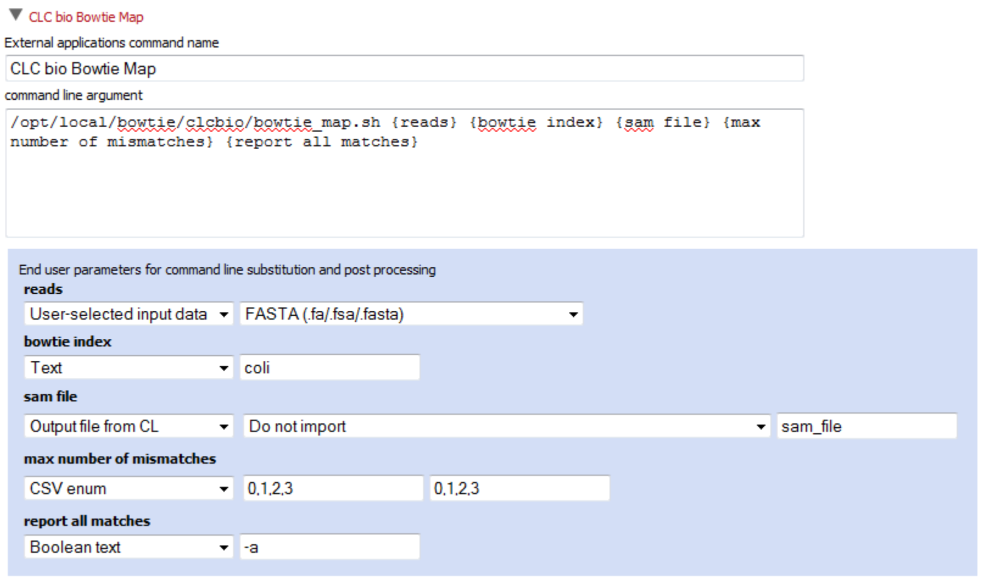
From `ftp://ftp.cbcb.umd.edu/pub/data/bowtie_indexes/` you can download pre-built index files of many model organisms. Download the index files relevant for you and extract them into the `indexes` folder in the Bowtie installation directory.

9.7.2 Understanding the Bowtie configuration

Once the `bowtie.xml` has been imported, you can click the **CLC bio Bowtie Map** header to see the configuration as shown figure 9.7.

The basic configuration is very much similar to the Velvet set-up (section 9.7). The `index` parameter is used to point to the relevant index file and could be substituted by a **CSV enum** to provide a fixed set of index files to be presented to the users.

There is one thing that complicates the Bowtie integration when compared to the relatively simple Velvet set-up. The result of Bowtie is a SAM or BAM file which cannot be imported using the



External applications command name: CLC bio Bowtie Map

command line argument:

```
/opt/local/bowtie/clcbio/bowtie_map.sh {reads} {bowtie index} {sam file} {max
number of mismatches} {report all matches}
```

End user parameters for command line substitution and post processing

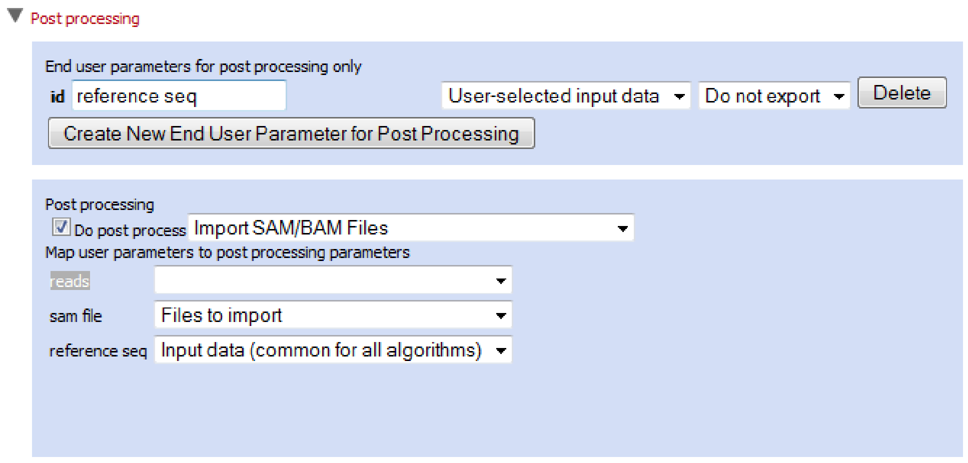
reads	User-selected input data	FASTA (.fa/.fsa/.fasta)	
bowtie index	Text	coli	
sam file	Output file from CL	Do not import	sam_file
max number of mismatches	CSV enum	0,1,2,3	0,1,2,3
report all matches	Boolean text	-a	

Figure 9.7: *The Bowtie configuration has been imported.*

standard import framework of *CLC Genomics Server*. This is because the file needs to be paired with a reference sequence. But SAM/BAM import is available on the server as an algorithm, similar to Trim, Read mapping and other tools.

In order to make use of this algorithm, the standard export-run-import flow cannot be used. Instead we use the concept of *post-processing* as an alternative to import.

The sam file parameter is set as the **Output file from C**L and the option selected is **Do not import**. If you expand the **Post-processing** panel, you can see the logic needed to handle the SAM file from Bowtie together with the reference sequence provided by the user (see figure 9.8).



End user parameters for post processing only

id reference seq | User-selected input data | Do not export

Post processing

☑ Do post process: Import SAM/BAM Files

Map user parameters to post processing parameters

reads	
sam file	Files to import
reference seq	Input data (common for all algorithms)

Figure 9.8: *The Bowtie post-processing set-up.*

At the top, there is a panel for specifying **End user parameters for post processing only** which in this case is the reference sequence. It is not needed by Bowtie which uses the index file as reference, so it is only needed for the SAM/BAM import post-processing.

Below the post-processing algorithm is specified, in this case **Import SAM/BAM Files**, and the input parameters are specified below. To the left, the various parameters (either from the command line expression at the top or from the **End user parameters for post processing only** panel) are listed and you can then map them to the corresponding input parameters for SAM/BAM import. In this case the `sam file` is mapped to **Files to import**, and the `reference seq` is mapped to **Input data**.

You may have noticed that the parameters that are available here are not all the ones from the top. This is because the server interprets the parameter types and only lists the ones that are suitable as input to parameters of the SAM/BAM import. In this case it is the output file from the command line program and sequence files selected by the user (this is the reason why `reads` is listed here, because it also represent sequences).

9.7.3 Parameter overview

Since the set-up and flow of parameters can be quite complex, there is a **Parameter flow** panel at the bottom of the configuration with a small graph for each parameter (see figure 9.9).

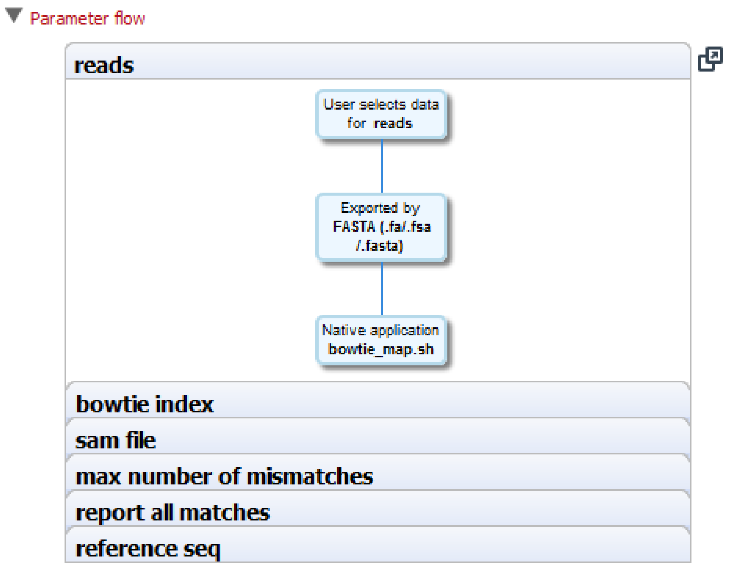

Parameter flow
reads
User selects data for reads
Exported by FASTA (.fa/.fsa /.fasta)
Native application bowtie_map.sh
bowtie index
sam file
max number of mismatches
report all matches
reference seq

Figure 9.9: *The reads parameter.*

The `reads` parameter starts with user selecting data, and the sequences are exported in fasta format and used as input for the Bowtie script.

Figure 9.10 shows the `max number of mismatches` parameter which also starts with the user selecting a value that is passed to the Bowtie script.

Figure 9.11 shows the reference seq parameter which also starts with the user selecting data which is passed on to the SAM/BAM import.

9.7.4 Setting path for temporary data

The **Environment** handling shown in figure 9.12 allows you to specify a folder for temporary data and add additional environment variables to be set when running the external application.

In Bowtie, the post-processing step needs to access the SAM file. Thus, the working directory you set must be the directory where this SAM file will be located, which will be in one of the

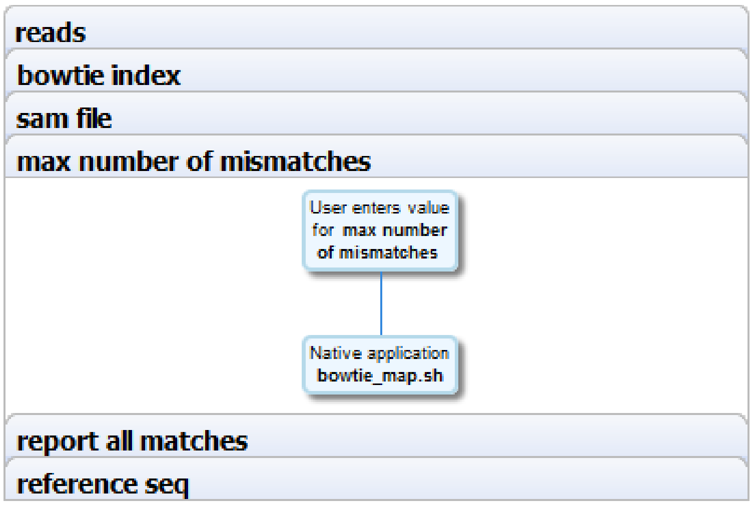


Figure 9.10: *The max number of mismatches parameter flow.*

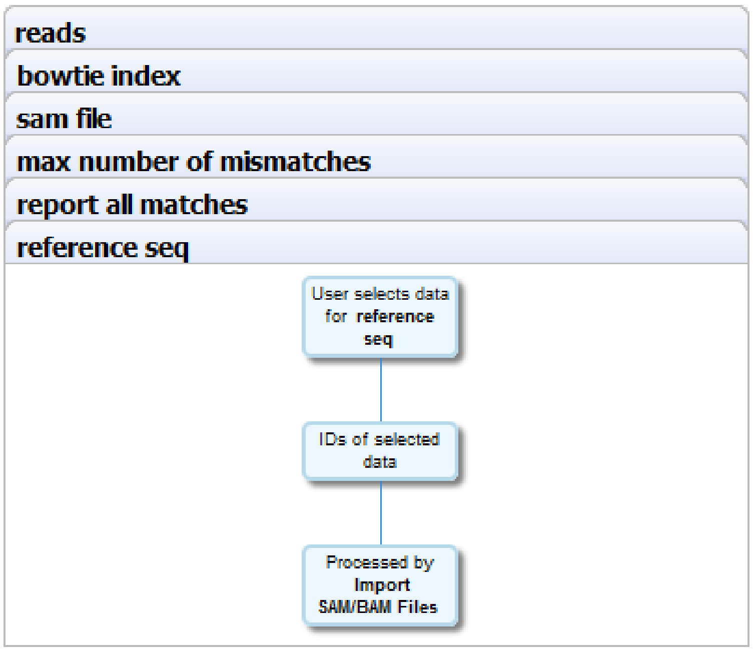


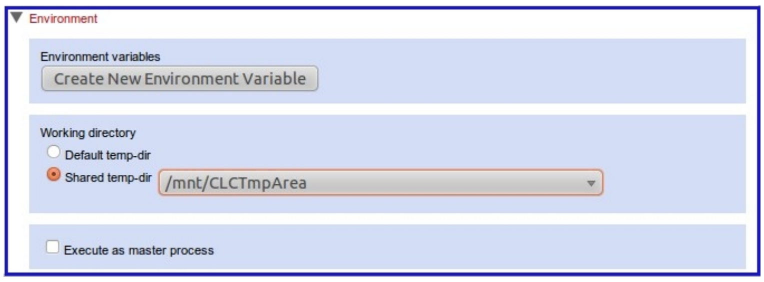
Figure 9.11: *The reference seq parameter flow.*

directories you have configured as an Import/Export directory.

If you are running on a master-node setup, the directory you choose must be shared, that is, accessible to all nodes you plan to have as execution nodes for this task. This is because different stages of your task could be run on different nodes. For example, the export process could run on a different node than the actual execution of the Bowtie script and the post-processing. Thus, in a master-node setup, be it using grid or CLC execution nodes, having this shared temporary area eliminates the overhead of transferring the temporary files between job nodes.

9.7.5 Tools for building index files

We have also included scripts and configurations for building index files using the external applications on *CLC Genomics Server*. This also includes the possibility of listing the index files



Working directory
Default temp-dir
[✓ Shared temp-dir] /mnt/CLCTmpArea
Execute as master process

Figure 9.12: *Temporary data should be defined for Bowtie.*

available. To get these to work, please make sure the path to the Bowtie installation directory is correct.

You should also note that the Bowtie distribution includes scripts to download index files of various organisms.

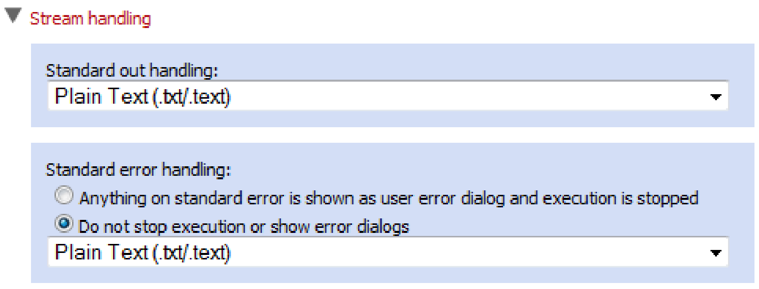
9.8 Troubleshooting

9.8.1 Checking the configuration

Since there is no check of consistency of the configuration when it has been set up, errors will only be seen on runtime when the application is executed. In order to help trouble-shooting in case of problems, there are a few things that can be done:

First, in the error dialog that will be presented in the workbench, you can see the actual command line call in the **Advanced** tab at the bottom. This can be a great help identifying syntax errors in the call.

Second, if you choose to import standard out and standard error as text, this will make it possible to check error messages posted by the external application (see figure 9.13).



Standard out handling: Plain Text (.txt/.text)

Standard error handling:
Anything on standard error is shown as user error dialog and execution is stopped
[✓ Do not stop execution or show error dialogs]
Plain Text (.txt/.text)

Figure 9.13: *Importing the result of standard error and standard out.*

Once the set-up is running and stable, you can deselect these options.

9.8.2 Check your third party application

- Is your third party application being found? Perhaps try giving the full path to it.
- If you are using someone else's configuration file, make sure the location to the third party application is correct for your system.

- If you are using someone else's wrapper scripts, make sure all locations referred to inside the script are correct for your system.
- Is your third party application executable?
- If there was a wrapper script being used to call the third party applciation, is that wrapper script executable?

9.8.3 Is your Import/Export directory configured?

For certain setups, you need to have Import/Export directories configured. Pleaes refer to section 9.4.2 for more details on this.

9.8.4 Check your naming

If your users will only access the External Applications via the Workbench, then you do not have to worry about what name you choose when setting up the configuration. However, if they plan to use the **clcserver** program, from the CLC Command Line Tools, to interact with your *CLC Genomics Server*, then please ensure that you do not use the same name as any of the internal commands available. You can get a list of these by running the clcserver command, with your *CLC Genomics Server* details, and using the -A flag with no argument.

Chapter 10

Workflows

The *CLC Genomics Server* supports workflows that are created with the CLC Workbenches. A work flow consists of a series of tools where the output of one tool is connected as the input to another tool. As an example, a workflow could pass data through read mapping, use the mapped reads as input for variant detection, and perform some filtering of the variant track. The workflow is created in the CLC Workbench and an installer file is created that can be installed on the *CLC Genomics Server*. For information about creating a workflow, please see the user manual of *CLC Genomics Workbench* or *CLC Main Workbench* at `http://www.clcbio.com/usermanuals`.

10.1 Installing and configuring workflows

Workflows can be installed from the server web interface:

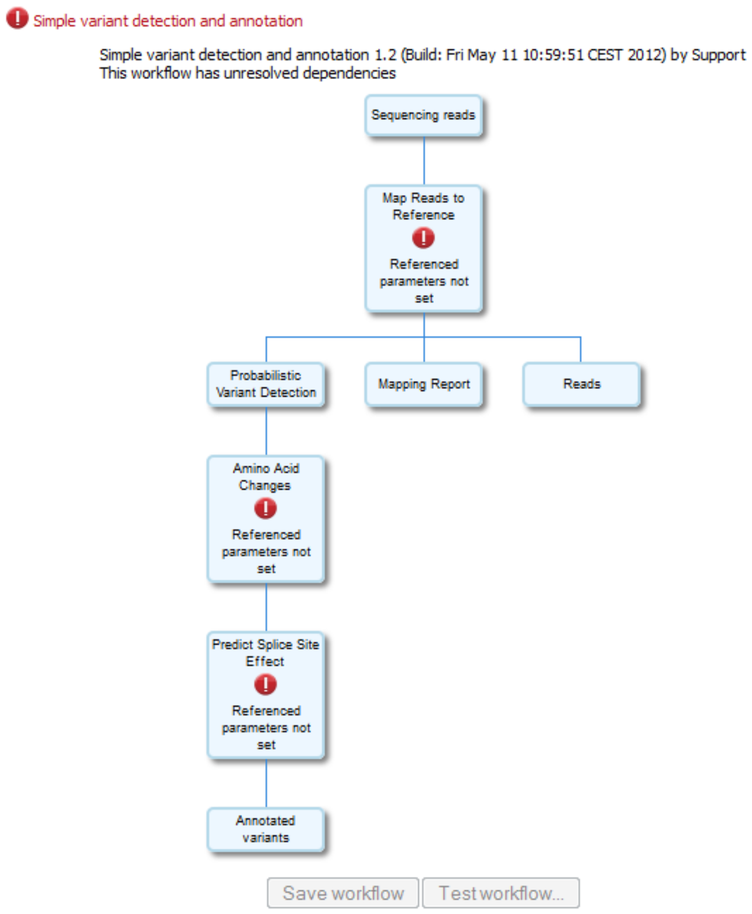
Admin ( ) | Workflows ( )

Click the **Install Workflow** button and select a workflow installer (for information about creating a workflow, please see the user manual of *CLC Genomics Workbench* or *CLC Main Workbench* at `http://www.clcbio.com/usermanuals`).

Once installed, the workflow is listed with a validated (✓) or attention (❗) status icon. When a workflow does not validate, it is usually because it depends on some reference data that needs to be specified. Alternatively, it can be because the tool specified in the workflow is not available on the server in the right version. When you click the name of the workflow, you will see a diagram with detailed information about which part of the workflow that validates as shown in figure 10.1).

In this example, the read mapping needs configuration. Simply click the read mapping box and you will see a dialog listing the parameters that need to be configured as well as an overview of all the parameters. An example is shown in figure 10.2.

In this case it is the read mapper that needs a reference genome which you have to select before the read mapper validates. Only the parameters that involve choosing some kind of reference data are open for configuration on the server. In order to change other parameters, you have to adjust in the original workflow definition.

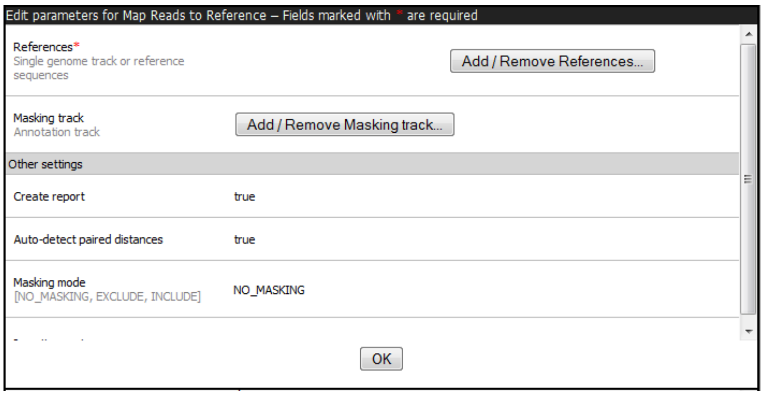


Simple variant detection and annotation

Simple variant detection and annotation 1.2 (Build: Fri May 11 10:59:51 CEST 2012) by Support
This workflow has unresolved dependencies

Sequencing reads
Map Reads to Reference
Referenced parameters not set
Probabilistic Variant Detection
Mapping Report
Reads
Amino Acid Changes
Referenced parameters not set
Predict Splice Site Effect
Referenced parameters not set
Annotated variants
Save workflow
Test workflow...

Figure 10.1: *A workflow is installed and needs configuration.*



Edit parameters for Map Reads to Reference – Fields marked with * are required

References* Single genome track or reference sequences	Add / Remove References...
Masking track Annotation track	Add / Remove Masking track...
Other settings	
Create report	true
Auto-detect paired distances	true
Masking mode [NO_MASKING, EXCLUDE, INCLUDE]	NO_MASKING

OK

Figure 10.2: *The read mapper needs a reference sequnce.*

10.2 Executing workflows

Once a workflow is installed and validated, it becomes available for execution. When you log in on the server using the CLC Workbench, workflows installed on the server automatically become available in the **Toolbox** (see figure 10.3).

When you select it, you will be presented with a dialog as shown in figure 10.4 with the options of where to run the workflow.

This means that workflows installed on the server can be executed either on the server or in

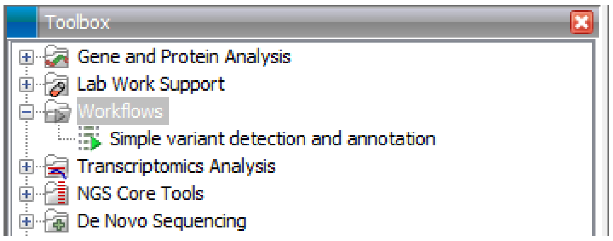
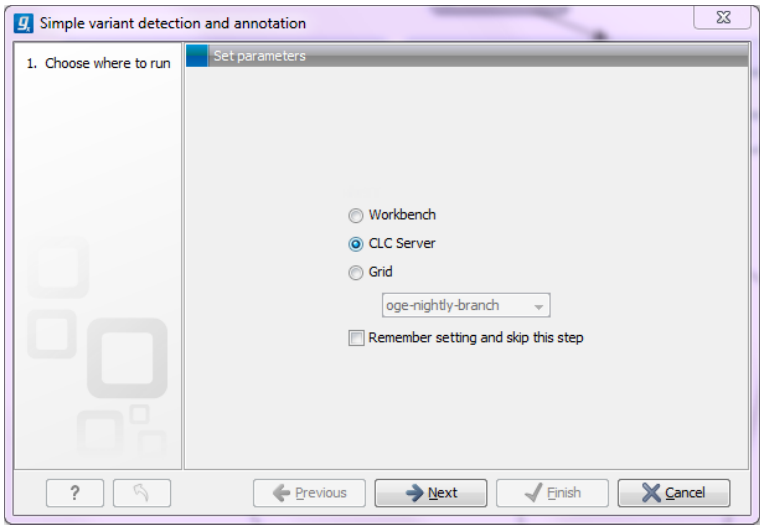


Figure 10.3: *A workflow is installed and ready to be used.*



( ) Workbench
(●) CLC Server
( ) Grid
oge-nightly-branch
☐ Remember setting and skip this step

Figure 10.4: *Selecting where to run the workflow.*

the workbench. In the same way, workflows installed on the workbench can be executed on the server as well as on the workbench. The only requirement is that both the tools that are part of the workflow and any reference data are available.

An important benefit of installing workflows in the server is that it provides the administrator an easy way to update and deploy new versions of the workflow, because any changes immediately take effect for all workbench users as well.

Appendix A

Use of multi-core computers

This section lists the tools that are making use of multi-core CPUs. This does not mean that they use all CPU cores available the whole time, but it means that they would benefit from running on computers with multiple CPU cores.

- Map Reads to Reference
- De Novo Assembly
- RNA-Seq Analysis
- Probabilistic Variant Detection
- Sequencing QC Report (will not scale well on more than four cores)
- BLAST (will not scale well on many cores)
- Large Gap Read Mapper (current in beta, part of the Transcript Discovery plug-in)

Appendix B

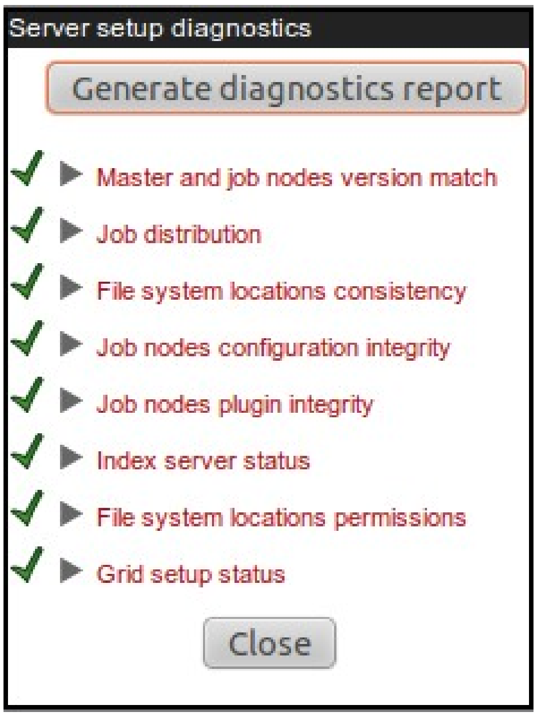
Troubleshooting

If there are problems regarding the installation and configuration of the server, please contact support@clcbio.com.

B.1 Check set-up

In order to check that your server has been set up correctly, you can run the **Check set-up** tool. Log in on the web interface of the server as an administrator and click the **Check Set-up** link at the upper right corner. This will show a dialog where you click **Generate Diagnostics Report**.

This will show a list of test that are performed on the system as shown in figure B.1.



✓ Master and job nodes version match
✓ Job distribution
✓ File system locations consistency
✓ Job nodes configuration integrity
✓ Job nodes plugin integrity
✓ Index server status
✓ File system locations permissions
✓ Grid setup status

Figure B.1: *Check system. Failed elements will be marked with a red X. If you have not configured your Server to submit jobs to a local Grid system, or if you have and your setup is configured correctly, you will see a green checkmark beside the Grid setup status item in the diagnostic report.*

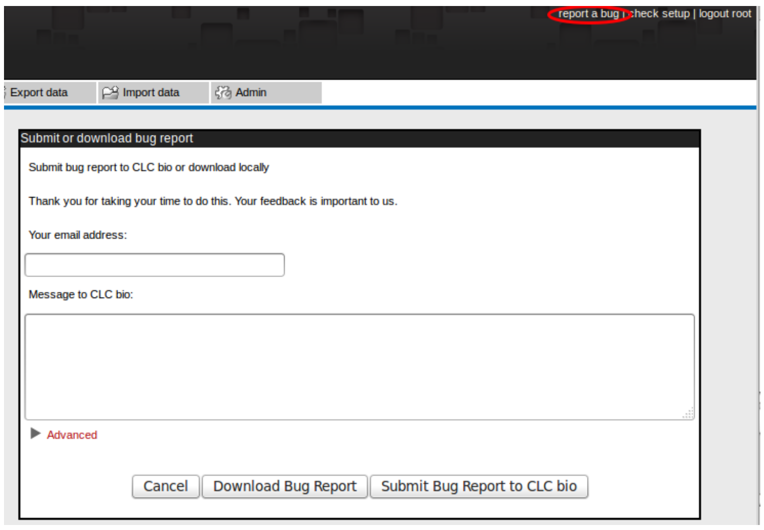
If any of the tests fail, it will be shown in the list. You can expand each of the tests to display more information about what the test is checking and information about the error if it fails.

B.2 Bug reporting

When contacting support@clcbio.com regarding problems on the server, you will often be asked for additional information about the server set-up etc. In this case, you can easily send the necessary information by submitting a bug report:

> **Log in to the web interface of the server as administrator | report a bug (at the top right corner) | Enter relevant information with as much detail as possible | Submit Bug Report to CLC bio**

You can see the bug report dialog in B.2.


Submit or download bug report

Submit bug report to CLC bio or download locally

Thank you for taking your time to do this. Your feedback is important to us.

Your email address:

Message to CLC bio:

report a bug | check setup | logout root
Export data
Import data
Admin
Advanced
Cancel
Download Bug Report
Submit Bug Report to CLC bio

Figure B.2: *Submitting a bug report to CLC bio.*

The bug report includes the following information:

- Log files
- A subset of the audit log showing the last events that happened on the server
- Configuration files of the server configuration

In a job node set-up you can include all this information from the job nodes as well by checking the **Include comprehensive job node info** checkbox in the **Advanced** part of the dialog.

If the server does not have access to the internet, you can **Download bug report**. This will create a zip file containing all the information and you can pass that on to CLC bio support. If the server has access to the internet, you can **Submit Bug Report to CLC bio**.

Note that the process of gathering the information for the bug report can take a while, especially for job node set-ups. If a Workbench user experiences a server-related error, it is also possible to submit a bug report from the Workbench error dialog. This report will include the same archive as when submitting a bug report from the web interface. All data sent to support@clcbio.com is treated confidentially.

No password information is included in the bug report.

Appendix C

Database configurations

C.1 Configurations for MySQL

For MySQL we recommend basing your configuration on the example configuration file `my-large.cnf` which is included in the MySQL distribution.

In addition the following changes should be made:

The `max_allowed_packet` should be increased to allow transferring large binary objects to an from the database. This is done by setting the option: `max_allowed_packet = 64M`

InnoDB must be available and configured for the MySQL instance to work properly as the CLC Database. You should enable the options in the InnoDB section of your configuration as suggested below:

```
# You can set .._buffer_pool_size up to 50 - 80 %
# of RAM but beware of setting memory usage too high
innodb_buffer_pool_size = 256M
innodb_additional_mem_pool_size = 20M
# Set .._log_file_size to 25 % of buffer pool size
innodb_log_file_size = 64M
innodb_log_buffer_size = 8M
innodb_flush_log_at_trx_commit = 1
innodb_lock_wait_timeout = 50
```

There appears to be a bug in certain versions of MySQL which can cause the cleanup of the query cache to take a very long time (some time many hours). If you experience this you should disable the query log by setting the following option: `query_cache_size= 0`

Appendix D

SSL and encryption

The *CLC Genomics Server* supports SSL communication between the server and its clients (e.g. Workbenches or the *CLC Server Command Line Tools*). This is particularly relevant if the server is accessible over the internet as well as on a local network.

The default configuration of the server does not use SSL.

D.1 Enabling SSL on the server

A **server certificate** is required before SSL can be enabled on the *CLC Genomics Server*. This is usually obtained from a *Certificate Authority* (CA) like Thawte or Verisign (see `http://en.wikipedia.org/wiki/Certificate_authorities`).

A **signed certificate** in a `pkcs12` keystore file is also needed. The keystore file is either provided by the CA or it can be generated from the private key used to request the certificate and the signed-certificate file from the CA (see section D.1.1).

Copy the keystore file to the conf subdirectory of the *CLC Genomics Server* installation folder.

Next, the `server.xml` file in the `conf` subdirectory of the *CLC Genomics Server* installation folder has to be edited to enable SSL-connections. Add text like the following text to the `server.xml` file:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS"
               keystoreFile="conf/keystore.pkcs12" keystorePass="tomcat"
               keystoreType="PKCS12"
/>
```

Replace `keystore.pkcs12` with the name of your keystore file, and replace `tomcat` with the password for your keystore.

The above settings make SSL available on port 8443. The stanadrd (non-SSL) port would still be 7777, or whatever you may have configured it to. If only SSL connection should be allowed, the connector entry for port 7777 can safely be removed from the `server.xml` file.

Self-signed certificates can be generated if only connection encryption is needed. See http://www.akadia.com/services/ssh_test_certificate.html for further details.

D.1.1 Creating a PKCS12 keystore file

If the certificate is not supplied in a pkcs12 keystore file, it can be put into one by combining the private key and the signed certificate obtained from the CA by using *openssl*:

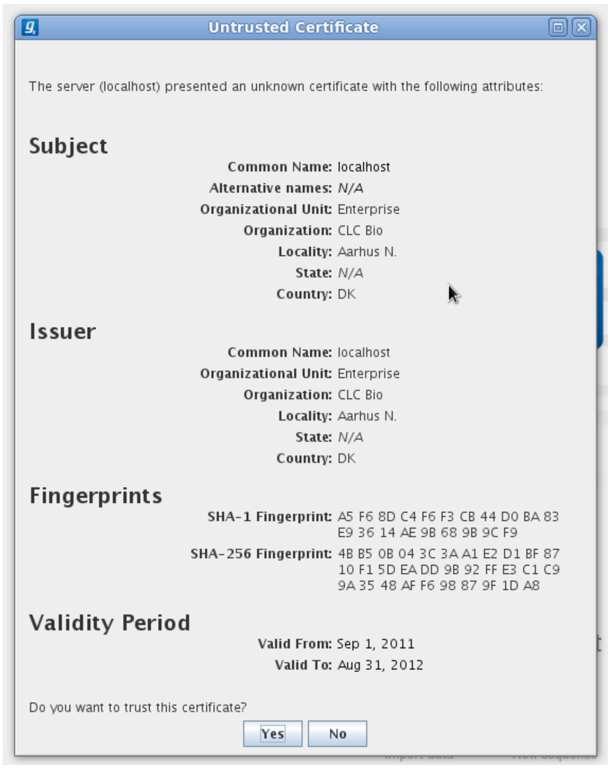
```
openssl pkcs12 -export -out keystore.pkcs12 -inkey private.key -in certificate.crt -name "tomcat"
```

This will take the private key from the file `private.key` and the signed certificate from `certificate.crt` and generate a pkcs12-store in the `keystore.pkcs12` file.

D.2 Logging in using SSL from the Workbench

When the Workbench connects to the *CLC Genomics Server* it automatically detects if Secure Socket Layer (SSL) should be used or not.

If SSL is detected, the server's certificate will be verified and a warning is displayed if the certificate is not signed by a recognized Certificate Authority (CA) as shown in figure D.1.



The server (localhost) presented an unknown certificate with the following attributes:

Subject

Common Name: localhost
Alternative names: *N/A*
Organizational Unit: Enterprise
Organization: CLC Bio
Locality: Aarhus N.
State: *N/A*
Country: DK

Issuer

Common Name: localhost
Organizational Unit: Enterprise
Organization: CLC Bio
Locality: Aarhus N.
State: *N/A*
Country: DK

Fingerprints

SHA-1 Fingerprint: A5 F6 8D C4 F6 F3 CB 44 D0 BA 83 E9 36 14 AE 9B 68 9B 9C F9
SHA-256 Fingerprint: 4B B5 0B 04 3C 3A A1 E2 D1 BF 87 10 F1 5D EA DD 9B 92 FF E3 C1 C9 9A 35 48 AF F6 98 87 9F 1D A8

Validity Period

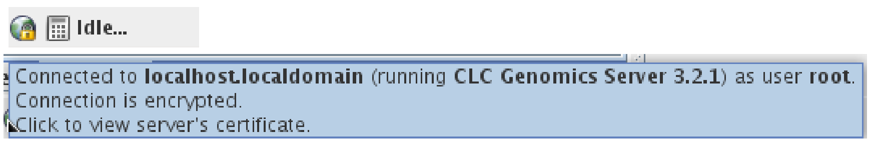
Valid From: Sep 1, 2011
Valid To: Aug 31, 2012

Do you want to trust this certificate?

Figure D.1: *A warning is shown when the certificate is not signed by a recognized CA.*

When such an "unknown" certificate has been accepted once, the warning will not appear again. It is necessary to log in again once the certificate has been accepted.

When logged into a server, information about the connection can be viewed by hovering the connection icon on the status-panel as shown in figure D.2.



Connected to **localhost.localdomain** (running **CLC Genomics Server 3.2.1**) as user **root**.
Connection is encrypted.
Click to view server's certificate.

Figure D.2: *Showing details on the server connection by placing the mouse on the globe.*

The icon is gray when the user is not logged in, and a pad lock is overlayed when the connection is encrypted via SSL.

D.3 Enabling redirection from non-ssl port to ssl-enabled port

If SSL is to be mandatory, it is possible to get the non-ssl port (7777) to forward to the ssl port (8443), rather than closing down the non-ssl port. This is done by adding the following section to the `conf/web.xml` file:

```
<security-constraint>
<display-name>Security Constraint</display-name>
<web-resource-collection>
      <web-resource-name>SSL-restricted Area</web-resource-name>
      <url-pattern>/*</url-pattern>
 </web-resource-collection>
<user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
</security-constraint>
```

The above section is commented out in the supplied `conf/web.xml` file, and just has to be commented in in order for redirection to work. Please note that this redirection only works for the browser accessing the web interface. When connecting from the Workbench, the correct port has to be specified.

D.4 Logging in using SSL from the *CLC Server Command Line Tools*

The *CLC Server Command Line Tools* will also automatically detect and use SSL if present on the port it connects to. If the certificate is untrusted the `clcserver` program will refuse to login:

```
./clcserver -S localhost -U root -W defaullt -P 7778
Message: Trying to log on to server
Error: SSL Handshake failed. Check certificate.
Option                                  Description
------                                  -----------
-A <Command>                            Command to run. If not specified the list of commands on the server will be returned.
-C <Integer>                            Specify column width of help output.
-D <Boolean>                            Enable debug mode (default: false)
-G <Grid Preset value>                  Specify to execute on grid.
-H                                      Display general help.
-I <Algorithm Command>                  Get information about an algorithm
-O <File>                               Output file.
-P <Integer>                            Server port number. (default: 7777)
-Q <Boolean>                            Quiet mode. No progress output. (default: false)
-S <String>                             Server hostname or IP-address of the CLC Server.
-U <String>                             Valid username for logging on to the CLC Server
-V                                      Display version.
-W <String>                             Clear text password or domain specific password token.
```

In order to trust the certificate the `sslStore` tool must be used:

```
./sslStore -S localhost -U root -W defaullt -P 7778
The server (localhost) presented an untrusted certificate with the following attributes:
SUBJECT
=======
Common Name        : localhost
Alternative Names  : N/A
Organizational Unit: Enterprise
Organization       : CLC Bio
Locality           : Aarhus N.
State              : N/A
Country            : DK

ISSUER
=======
Common Name        : localhost
Organizational Unit: Enterprise
Organization       : CLC Bio
Locality           : Aarhus N.
State              : N/A
Country            : DK

FINGERPRINTS
============
SHA-1              : A5 F6 8D C4 F6 F3 CB 44 D0 BA 83 E9 36 14 AE 9B 68 9B 9C F9
SHA-256            : 4B B5 0B 04 3C 3A A1 E2 D1 BF 87 10 F1 5D EA DD 9B 92 FF E3 C1 C9 9A 35 48 AF F6 98 87 9F 1D A8

VALIDITY PERIOD
===============
Valid From         : Sep 1, 2011
Valid To           : Aug 31, 2012
Trust this certificate? [yn]
```

Once the certificate has been accepted, the `clcserver` program is allowed to connect to the server.

Bibliography

[Langmead et al., 2009] Langmead, B., Trapnell, C., Pop, M., and Salzberg, S. L. (2009). Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biol*, 10(3):R25.

[Zerbino and Birney, 2008] Zerbino, D. R. and Birney, E. (2008). Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res*, 18(5):821–829.

Index