

# UCM4SVN

UNIFIED CHANGE MANAGEMENT FOR SUBVERSION



## administrationguide



*The content of this publication is the property of Clearvision-CM  
Reproduction of this content is strictly prohibited © 2009*

?

- 1 Introduction
- 2 Prerequisites
  - 2.1 Server Platform
  - 2.2 Client Platform
  - 2.3 License
    - 2.3.1 **Evaluation License**
    - 2.3.2 Full License
- 3 UCM4SVN Setup
  - 3.1 Create UCM4SVN Userid and Group
  - 3.2 Install Files
  - 3.3 Upgrading From A Previous Release

- 3.4 Password Encryption
- 3.5 Configure the Server Settings
  - 3.5.1.1 ucm4svn\_dir
  - 3.5.1.2 svn\_username/password
  - 3.5.1.3 ucm4svn\_workspace
  - 3.5.1.4 Email Notification Settings (SMTP)
  - 3.5.1.5 Activity Sources
  - 3.5.1.6 Baseline Status Settings
  - 3.5.2 License
  - 3.5.3 JIRA Settings
    - 3.5.3.1 Show Transition Links for Jira Issues
    - 3.5.3.2 Enable XML-RPC
    - 3.5.3.3 JIRA Workflow Matrix
  - 3.5.4 Trac Settings
    - 3.5.4.1 Install XML-RPC Plugin
    - 3.5.4.2 Adjust Permissions for XML-RPC Plugin
    - 3.5.4.3 Test XML-RPC Plugin
    - 3.5.4.4 Basic Configuration
    - 3.5.4.5 Trac Workflow
  - 3.5.5 ClearQuest Settings
    - 3.5.5.1 Basic Configuration
    - 3.5.5.2 ClearQuest Script Customisation
    - 3.5.5.3 get\_cq\_list.pl
    - 3.5.5.4 update\_cq.pl
    - 3.5.5.5 get\_cq\_url.pl
    - 3.5.5.6 get\_users.pl
    - 3.5.5.7 validate\_password.pl
  - 3.5.6 Logging
- 3.6 Starting the Server
- 4 Creating A Sample Database for Evaluation/Testing
- 5 How to restrict repository access
- 6 Troubleshooting
  - 6.1 Emergency Database Access
  - 6.2 Resetting UCM4SVN Data
  - 6.3 Running UCM4SVN As A Service
    - 6.3.1 On Windows
    - 6.3.2 On Linux

## Introduction

Welcome to Clearvision's UCM4SVN: Unified Change Management for Subversion

This is the Administration Guide for UCM4SVN, which explains how to install and configure UCM4SVN.

There is also a User Guide for a user's perspective, please refer to this for more detailed information about using UCM4SVN.

The latest versions of these documents are always available on

<http://www.clearvision-cm.com/products/unified-change-management-for-subversion.html>.

## Prerequisites

UCM4SVN delivers its service over the web, so any web browser can be used to interact with the application. The server is an executable, so you will need a version for your Operating System and Subversion release.

## Server Platform

The UCM4SVN *server* is tested and certified against a particular Subversion release and is currently distributed for specific platforms. (For more information on the product roadmap, please contact us at [sales@clearvision-cm.com](mailto:sales@clearvision-cm.com)).

v2.0 is currently built for (please check our website for newer platforms):

- CentOS 5.3
- Windows XP SP2

UCM4SVN has been tested against Subversion 1.6.5 and 1.5.5.

(For more information on the product roadmap, please contact us at [sales@clearvision-cm.com](mailto:sales@clearvision-cm.com)).

Note that the `svn` executable must be available on the server's path.

## Client Platform

UCM4SVN should be accessible with any modern web browser. It has been tested with the following browsers:

- Firefox 3
- MS Internet Explorer 7
- Google Chrome 1.0

The Java Checkout Tool, which is executed client-side, requires Java 1.5 or later and has been tested with Java 1.6.0\_14.

## License

### Evaluation License

Evaluation licenses can be requested from the clearvision website: <http://www.clearvision-cm.com>

A link to the license request form for a product can be found on the product's download page. To request a license we need some information about the machine that you will be running the server on. To get this information, you can run the `get_license_info` program on the machine you want to use as the server (the `get_license_info` program can be found within the `tools` directory of the installation directory). This will retrieve and present system information from the machine.

Once the license request form has been completed and submitted, an email will be sent to the registered email address containing the license information. This information will need to be inserted into the server config file.

Note that this license will only work on a particular server machine, on a particular port, and will allow a maximum number of users. The license is time-limited so it is recommended that Evaluation licenses are not requested until needed.

### Full License

Once a license has been purchased, you will receive an email containing a link to the online full license generator. A similar form will need to be filled in as for an evaluation license. Once the form has been completed and confirmed a full license will be emailed. Any evaluation license details in the config file will need to be replaced by the full license.

Licenses are tied to the hardware as well as the IP address of the machine used for the server. It is therefore necessary to ensure these details will not change as this will leave the license invalid.

```
License Request for Clearvision UCM4SVN:
  Evaluation: yes/no
  IP Address: 192.168.64.1
  Port: 54321
  MAC Address: 00-50-56-C0-00-05
  Users: 50
  Company: You Company name here
  Email Address: yourco@yourco.com
  Expiry Date: 2008-03-31
  Product: UCM4SVN
```

The `get_license_info` executable is run from a command prompt, eg on Windows:

---

## UCM4SVN Setup

On Windows, all of the configuration can be done via the installer. Make sure you have your licensing information handy as you will need this during the installation.

Once installed click on UCM4SVN Server in your Start Menu, located in the Clearvision folder. Then navigate to `http://<ip_address>:<port_number>` with your favourite browser as per your licensing information, logging in with the user details found below.

The instructions provided in this guide use Unix paths such as `/home/ucm4svn`. If you are installing and configuring UCM4SVN on Windows, you will need to adjust paths accordingly.

## Create UCM4SVN Userid and Group

We recommend that you create a user and group to run the server, e.g. user id 'ucm4svn', group name 'ucm4svn'. Note that for the instructions below, it is assumed that the home directory for the new user id is `/home/ucm4svn`.

## Install Files

If you have created a new user, you can install the UCM4SVN files within that user's home directory, eg:

```
/home/ucm4svn/ucm4svn/
```

and set the permissions so that the ucm4svn user can run the server (you may need to run the command below as ROOT).

```
chown -R ucm4svn:ucm4svn /home/ucm4svn/ucm4svn
```

Alternatively, if you are installing as a system service, you might prefer to install into a system folder, e.g.:

```
/opt/Clearvision/ucm4svn
```

(Note that we will use `/home/ucm4svn` to refer to this location in the remainder of this guide. Please substitute this with your installation directory).

## Upgrading From A Previous Release

If you are upgrading from a previous version of UCM4SVN (2.0 or later), you will need to upgrade your existing database for the new release.

Please note that databases from releases prior to UCM4SVN 2.0 cannot be upgraded and you will need to migrate your data as described in the UCM4SVN User Guide (Appendix A: Data Migration).

**BEFORE UPGRADING, PLEASE ENSURE THAT YOU FIRST BACK UP YOUR DATABASE. Clearvision are not responsible for any loss of data incurred as the result of a database upgrade.**

The database is upgraded using the `ucm4svn_update_db` tool. The `ucm4svn_update_db` tool will find the database file using the `ucm4svn_dir` setting in the config file and upgrade the database for the latest release.

The upgrade tool is invoked as follows:-

```
ucm4svn_update_db -f /path/to/ucm4svn.cfg
```

Note that if you do not specify the `-f` option, the upgrade tool will look for a config file called `{ucm4svn.cfg}` in the current working directory.

## Password Encryption

UCM4SVN requires a number of user names and passwords in its config file in order to be able to communicate with external systems such as Subversion, JIRA or ClearQuest.

If you provide passwords in plain text in the config file, users with sufficient permissions to read the config file can potentially gain access to your passwords.

In order to protect your passwords, you can encrypt them using the `ucm4svn_passwd` utility before placing the encrypted string into the config file.

To encrypt a password, execute the `ucm4svn_passwd` command, which lives in the tools directory of the UCM4SVN installation, and follow the instructions:-

```
> ucm4svn_passwd
UCM4SVN Password Encryption Utility
enter username> peter
enter password>
confirm password>
Encrypted password: Enc'e9b79876846668fd1b55222588af1d52'
Please place the complete string including the Enc and single quotes into your config file
```

Note that `ucm4svn_passwd` will not display the password being entered.

Please place the complete encrypted password string into your config file, e.g. using the above example of a Subversion user called 'peter', you would set up your Subversion settings as follows in the config file:

```
svn_username=peter
svn_password=Enc'e9b79876846668fd1b55222588af1d52'
```

## Configure the Server Settings

The server uses a configuration file to hold license information and other optional settings. A template is provided in `/home/ucm4svn/ucm4svn/ucm4svn.cfg`.

The UCM4SVN section of the config file contains the basic settings required for UCM4SVN to operate, which look

```
[UCM4SVN]
svn_username=svn_username
svn_password=svn_password
ucm4svn_workspace=/home/ucm4svn/workspace
ucm4svn_dir=/home/ucm4svn/ucm4svn

smtp_host=smtp.my-company.com
smtp_username=username
smtp_password=password
email_from=issue-tracker@no-reply.com

allow_multiple_activity_sources=1
```

## ucm4svn\_dir

This should be set to the directory where you installed UCM4SVN (eg /opt/Clearvision/ucm4svn or /home/ucm4svn/ucm4svn).

## svn\_username/password

The server must be able to write to any Subversion repositories that you are using. Use the `svn_username` and `svn_password` options to set these credentials.

## ucm4svn\_workspace

UCM4SVN needs an area to store temporary data. This option should be set to a suitable location, e.g. /home/ucm4svn/workspace. Please make sure that this directory has full write permissions for the ucm4svn server and has sufficient disk space available.

## Email Notification Settings (SMTP)

UCM4SVN can be configured to send emails on certain actions, e.g. to notify users that an activity has been assigned to them. To enable this feature, you need to set the `smtp_host`, `smtp_username` and `smtp_password` options in the config file. If you do not require this feature, please leave the smtp settings commented out.

Note that the host specified with `smtp_host` needs to be configured to accept SMTP (Simple Mail Transfer Protocol) connections on port 25 (the standard port for SMTP).

You can also configure the from-address of email notifications using the `email_from` option. Configuring the from-address is required if you want users to be able to reply to email notifications sent by UCM4SVN. In an environment where you are running multiple UCM4SVN servers, the from-address can also be used to distinguish between email notifications from the different servers.

```
[UCM4SVN]
smtp_host=smtp.your-server.com
smtp_username=your_username_here
smtp_password=your_password_here
email_from=issue-tracker@no-reply.com
```

## Activity Sources

An activity source is where an activity is created from. With UCM4SVN's [ClearQuest](#) and JIRA integrations, it is not only possible to create activities in UCM4SVN, but you can also create new activities by importing JIRA issues or [ClearQuest](#) records.

When you create a project, UCM4SVN allows you to configure which sources are allowable activity sources for new activities for the project. The three options are (provided both JIRA and [ClearQuest](#) integrations have been enabled):-

- UCM4SVN
- JIRA
- [ClearQuest](#)

By default, when you create a project you can specify more than one activity source. However, if you set the option `allow_multiple_activity_sources` to "0", then you restrict Project Managers to choose one of the options when a new project is created. This means that new activities for the project are then always created from the same activity source.

```
allow_multiple_activity_sources=0
```

## Baseline Status Settings

UCM4SVN allows you to label baselines in order to indicate their status. You can configure the names for labels using the following settings:-

```
baseline_good=Good
baseline_bad=Bad
baseline_untested=Untested
baseline_failed=Failed
baseline_obsolete=Obsolete
```

The value for each setting can be any text of your choice. However, it is recommended that you keep these relatively short and meaningful.

Note that the colours used in UCM4SVN are currently only configurable in the HTML stylesheet itself, which lives in the UCM4SVN installation under `media/css/ucm.css`.

## License

Set the License and Server sections of the config file as per your license email (  please note that all information is **case-sensitive**).

## JIRA Settings

You can configure UCM4SVN to interface to your JIRA server. To enable this feature, you need to set the `jira_server` option to point to your JIRA server location. Please specify the server location as `http://<host_name>:<port_number>`, or as `https://<host_name>:<port_number>`. If no protocol is defined, then `http://` is assumed.

For example, a valid JIRA server setting would be `http://our-jira-server.domain.com:8080`.

*Note: Do not forget the port number!*

For the UCM4SVN to be able to access the JIRA server it requires a valid administrator account. This does not have to be the main administrative account, but it does require full admin rights such that it can access user information and import ticket lists.

The JIRA user name and password are specified in the config file using the `jira_username` and `jira_password` options.

Example:-

```
[JIRA]
jira_server=http://our-jira-server.domain.com:8080
jira_username=myuser
jira_password=mypassword
```

## Show Transition Links for Jira Issues

The `jira_show_transition_links` option allows you to add links to the actual transition being performed to any error messages resulting from a JIRA issue transition failure. This is particularly useful if transitions in your JIRA environment require the user to enter additional information such as fill in mandatory fields or enter a comment. When a transition fails, the user can click on the direct link to the transition, and a new browser window will open, guiding the user through the transition.

```
[JIRA]
jira_show_transition_links=false
```

Please note that with JIRA 4.1 and later, you will need to disable from token checking in JIRA for direct links to function correctly. The details on how to disable form token checking can be found in the JIRA documentation: <http://confluence.atlassian.com/display/JIRA/Disabling+Form+Token+Checking>

## Enable XML-RPC

Enable the XML-RPC remote JIRA API as per the <http://confluence.atlassian.com/display/JIRA/JIRA+XML-RPC+Overview> , in summary:

- Go to System/Plugins in the Administration page and check that the RPC Jira Plugin is installed and that the 'System XML-RPC Services' are enabled
- Go to Global Settings/General Configuration set 'Accept remote API calls' to 'ON'.

## JIRA Workflow Matrix

UCM4SVN can, when integrated with JIRA, progress an issue through the relevant states. For example, when an issue is imported from JIRA, it remains in the Open state. Once an activity has been created from this issue in UCM4SVN and the developer has accepted this activity, it will move on to the 'In Progress' state. Then once the developer has finished his work and delivered it back to trunk (or has gotten an integrator to do so) the activity then progresses to the final state, 'Closed'.

*NOTE: If you enable this feature, you must specify a configuration that specifies which JIRA workflow action should be taken for each state.*

The example below shows a workflow matrix for the default JIRA workflow. If you use customised workflows in JIRA, you will need to go to the Administration tab and select the Workflows option from the menu on the left. Then click on Steps for your workflow. State ids are shown in parentheses behind each Step Name, and Action ids are shown in parentheses behind each Transition. The JIRA\_workflow settings need to be defined such that for each state it defines what action to take. The format is as follows:-

```
jira_accept=<state_id>:<workflow_action_id>
jira_deliver=<state_id>:<workflow_action_id>
jira_close=<state_id>:<workflow_action_id>
jira_integrate=<state_id>:<workflow_action_id>
```

The default workflow looks as follows. Note that if you are using the default JIRA workflow, you will not need to customise this.

```
jira_accept=1:4
jira_deliver=3:2
jira_close=3:5
jira_integrate=5:701
```

The behaviour of the default workflow is described in the following table:

UCM4SVN Action	Previous JIRA State	JIRA Workflow Action	New JIRA State
Accept	Open	Start Progress	In Progress
Deliver	In Progress	Close Issue	Closed
Close	In Progress	Resolve Issue	Resolved
Integrate	Resolved	Close Issue	Closed

If you need to customise this workflow, then please make sure that the action specified for Accept puts the JIRA issue into a state (e.g. In Progress) that makes it possible for the actions specified for Deliver and Close to progress the tickets further through the workflow. The same applies to the Close and Integrate actions. In other words, the New JIRA State for Accept needs to match the Previous JIRA State for Deliver and Close, and the New JIRA State for Close needs to match the Previous JIRA State for Integrate.

## Trac Settings

UCM4SVN can integrate with the Trac change management system. To enable this option, you must configure your Trac settings in the config file.

UCM4SVN requires the XML-RPC plugin to be installed on your Trac server, and permissions must be granted to use the plugin to all authenticated users.

### Install XML-RPC Plugin

UCM4SVN requires the Trac XML-RPC plugin. The installation is very straightforward, especially when using `easy_install`. Please see the XML-RPC plugin web site <http://trac-hacks.org/wiki/XmlRpcPlugin> for details on the installation.

As a quick guide, you can install the XML-RPC plugin as follows:-

#### Easy Install

Using Easy Install with Trac 0.11:-

```
easy_install -Z -U http://trac-hacks.org/svn/xmlrpcplugin/trunk
```

Using Easy Install with Trac 0.10:-

```
easy_install -Z -U http://trac-hacks.org/svn/xmlrpcplugin/0.10
```

#### Manual Install

- Download source, unpack, change into source directory, then:-

```
python setup.py bdist_egg
cp dist/*.egg <path_to_trac>/env/plugins
```

- Then edit `trac.ini` and add:-

```
[components]
tracrpc.* = enabled
```

## Adjust Permissions for XML-RPC Plugin

Once the XML-RPC plugin has been installed, you need to give all authenticated users access to the XML-RPC API. This can be done as follows:-

- Log in to Trac as an administrator
- Select the Admin tab
- In the Grant Permission box on the right, type authenticated in the Subject field and select XML\_RPC from the Action pull-down list.

**Grant Permission:**

Subject:

Action:

Grant permission for an action to a subject, which can be either a user or a group.

- Press Add

If you prefer, you can limit access to the XML-RPC API on a per-user basis. However, as a minimum requirement every UCM4SVN user must have access to the XML-RPC API.

## Test XML-RPC Plugin

If you wish, you can test that the XML-RPC plugin is operational with the following Python code:-

```
import xmlrpclib
username="user"
password="password"
server_address="trac.company.com/myproject"
proxy = xmlrpclib.ServerProxy("http://%s:%s@%s" % (username, password, server_address))
print proxy.system.getAPIVersion()
```

The result of this should be a list with three elements indicating the Trac API version, e.g. 1, 0, 6.

## Basic Configuration

UCM4SVN needs to know which Trac server to connect to. Use the configuration options in the TRAC section to configure your Trac server.

```
[TRAC]
trac_server=http://trac.mycompany.com:8000/
trac_username=trac_username
trac_password=trac_password
```

Specify the Trac server to connect to and provide the user name and password of an account with XML-RPC permissions.

## Trac Workflow

In order to move tickets to their next state, UCM4SVN needs to know the state changes you would like to be performed on UCM4SVN actions.

For each of the actions, specify the state it is coming from and the state you would like to move to. Original and new state names are separated by a colon.

```
trac_accept=new:assigned
trac_deliver=assigned:closed
trac_close=assigned:accepted
trac_integrate=accepted:closed
```

## ClearQuest Settings

If you are integrating with [ClearQuest](#), you will need to configure [ClearQuest](#)-related settings in the configuration file and customise the example CQPerl scripts that are delivered with UCM4SVN for your needs.

### Basic Configuration

[ClearQuest](#) settings are provided in the config file in the `CQ` section:

```
[CQ]
cq_username=user
cq_password=password
cq_db=db_name
cq_db_set=7.1
cq_perl=C:\Program Files\IBM\RationalSDLC\ClearQuest\CQperl.exe
cq_perl_scripts_dir=c:\Program Files\Clearvision\ucm4svn\scripts

cq_accept=open
cq_close=resolve
cq_deliver=resolve
cq_integrate=validate
```

The configuration needs to be customised as follows:-

- Set `cq_username` to the name of a suitable account with administrator privileges. The account used needs to be able retrieve information about user accounts from the server and run queries through CQPerl.
- Set `cq_password` to the password for the administrative account.
- Set `cq_db_name` to the database with which you would like to interface. If you need to be able to interface to multiple databases, you can ignore this setting and you will need to customise the `.pl` scripts in the `scripts` directory.
- Set `cq_db_set` to the name of the [ClearQuest](#) schema repository for the database specified in `cq_db_name`.
- Set `cq_perl` to the full path to the CQPerl executable on your system.
- Set `cq_perl_scripts_dir` to the `scripts` directory in the UCM4SVN installation. If you need to be able to maintain a number of different customised versions of the scripts for difference UCM4SVN servers, you can take a copy of the scripts for each server and customise the server's config file to point to the correct location.

The remaining four options specify the [ClearQuest](#) actions to perform on particular UCM4SVN actions. The actions are used to move [ClearQuest](#) records to the correct state on UCM4SVN actions.

- When a developer accepts an activity in the to-do list, UCM4SVN will perform the action specified in `cq_accept`.
- When a develop closes an activity in the to-do list, UCM4SVN will perform the action specified in `cq_accept`.
- When a developer delivers and activity in the to-do list, the action specified in `cq_deliver` will be applied to the associated [ClearQuest](#) record.
- And finally, when an integrator integrates an activity, the action specified in `cq_integrate` is applied.

The default actions specified in the config file are suitable for [ClearQuest](#)'s Defect tracking workflow.

### ClearQuest Script Customisation

UCM4SVN's [ClearQuest](#) integration uses CQPerl scripts to interface to [ClearQuest](#). As [ClearQuest](#) is very customisable, it is likely that you will need to customise the scripts provided for your needs. The following table gives an overview for the scripts called by UCM4SVN and their purpose:-

Script	Purpose
<code>validate_password.pl</code>	Called to validate the user credentials of a user against <a href="#">ClearQuest</a> . Users imported from <a href="#">ClearQuest</a> will automatically carry a user type of " <a href="#">ClearQuest</a> " to indicate that they are imported users who should be authenticated against the external database.
<code>get_cq_url.pl</code>	This script is called to retrieve a URL for a <a href="#">ClearQuest</a> record with a particular id.
<code>get_users.pl</code>	The purpose of this script is to retrieve a list of <a href="#">ClearQuest</a> users and return adequate information to make it possible to import the users into UCM4SVN.

get_cq_list.pl	This script is called in New Activity to retrieve a list of <a href="#">ClearQuest</a> records to import into UCM4SVN. May arguments are passed into the script (see the script itself for details). However, please note that one of the arguments is the value of the CQ Selector field for the project for which the script is called. By default the value of the CQ Selector field is used as filter against the Project field. However, you may want to change this to suit your needs.
----------------	---

When customising scripts, you can test your changes by running the script from the command in using the correct command line options. You can see the arguments passed into each script at the top of the script itself, e.g. for `get_cq_list.pl`:-

```
my $cq_superuser_id = @ARGV[0];      # cq_username from config file
my $cq_superuser_password = @ARGV[1]; # cq_password from config file
my $project_id = @ARGV[2];           # Numerical database project-table row id
my $project_selector = @ARGV[3];     # Cq Selector specified in Create/Edit Project
my $cq_db = @ARGV[4];               # cq_db from config file
my $cq_dbset = @ARGV[5];            # cq_db_set from config file
```

The following is an example command line for running `get_cq_list.pl` from the command line:-

```
cqperl "get_cq_list" "admin" "pass" "0" "selector" "CQ" "7.1"
```

## get\_cq\_list.pl

This script is very likely to require customisation to suit your needs.

Note that the `project_id` argument passed into the script is currently not used. Instead, the script currently uses the value of the CQ Selector field to generate a filter for the project of a record as follows:-

```
@project_test = $project_selector;
$rootfilternode->BuildFilter("Project", $CQPerlExt::CQ_COMP_OP_EQ, \@project_test);
```

However, if you prefer not to use the CQ Selector field, you could use the `project_id` to distinguish between different projects.

The other test that is of interest is the state test. The default script only returns records that are in "Assigned" state. This is achieved with the following lines of code:-

```
@state_test hl. "Assigned"; # AND (state= 'Assigned')
$rootfilternode->BuildFilter("State", $CQPerlExt::CQ_COMP_OP_EQ, \@state_test);
```

Whatever your customisations to the script may be, you must make sure that the result is printed on standard-out (stdout) in the following format:-

```
<id>:<headline>:<owner>:<priority>:<description>;[<id>:<headline>:<owner>:<priority>:<description>;]
```

The fields for a each record are separated by colons and the records themselves are separated by semicolons. Due to this, `get_cq_list.pl` must make sure that colons and semicolons in the headline and description fields are filtered not to contain these reserved characters. The default implementation of the script achieves this by filtering the fields and removing the reserved characters using a regular expression.

For example, the following would be a valid result:-

```
CQ00000011:A new ticket from ClearQuest:admin:Critical:This is a ticket imported from ClearQuest;
```

## update\_cq.pl

The purpose of this script is to attach the closing comment provided by the Developer to the [ClearQuest](#) record and move the record to the next state.

The state to move records to can be configured in the UCM4SVN config file. However, you may want to customise the field to which closing comments are attached.

When testing customisations to this script, please note that closing comments are expected to be provided in a file and that the file name is passed into the script.

Also, the username and password combination passed into the script is that of the individual user performing the Accept/Close/Deliver/Integrate action. If users do not have suitable permissions to perform these actions through the [ClearQuest](#) API, you may want to change the script to use an administrator's account to perform the modifications.

## get\_cq\_url.pl

The script provided is no more than an example, and it needs to be customised to return a URL that allows users to navigate directly to a [ClearQuest](#) record. This script is used to generate the URL displayed in Edit Activity for activities imported from ClearCase.

## get\_users.pl

The script is used for importing users into UCM4SVN. Things to consider for this script are the method of authentication in [ClearQuest](#). For example, you may want to customise the script to use an LDAP server.

## validate\_password.pl

This script is called when a user with User Type "ClearQuest" logs in to UCM4SVN. You should not need to customise this script unless individual users cannot authenticate against [ClearQuest](#) using the CQPerl API and you need to use an alternative method of authentication.

## Logging

The UCM4SVN server will keep a log file. You can configure the level of detail using the `log_level` option, using one of the following:

- INFO: output verbose (maximal) informational details about the operations of UCM4SVN. These include creating components, users, projects and activities.
- WARNING: show any warning messages. Show details of possible issues when operating UCM4SVN (these are not errors, but may be useful for administration purposes or highlight a potential problem)
- ERROR: show only error messages (minimal output)

## Starting the Server

The final step is to start the UCM4SVN server. You can run the server with your config file using the `-f` flag:

```
/home/ucm4svn/ucm4svn/ucm4svn -f /home/ucm4svn/ucm4svn.cfg &
```

This will start the server and make UCM4SVN available to web clients. Now you can connect to the UCM4SVN server in a web browser and login using the username `admin` and password `admin`. From here you can create users, components, projects and activities. See the user guide for more information on how to do this. 💡 We recommend that you change the passwords for the default users at this stage.

- **Username:** admin
  - **Password:** admin
- 

## Creating A Sample Database for Evaluation/Testing

During testing or evaluation, it can be useful to set up some sample data.

⚠️ This is a destructive operation, so please **DO NOT USE ON A LIVE SYSTEM!**

Create an empty subversion repository, and make sure you have no data stored in UCM4SVN (**all existing data will be erased**). You can set up the sample database with:

Linux:

```
/home/ucm4svn/ucm4svn/tools/ucm4svn_reset_db -f /home/ucm4svn/ucm4svn.cfg  
--create-sample-database=http://my-repo-server/svn/test
```

Windows:

```
"C:\Program Files\Clearvision\ucm4svn\ucm4svn_reset_db.exe" -f "C:\Program  
Files\Clearvision\ucm4svn\ucm4svn.cfg" --create-sample-database=http://my-repo-server/svn/test
```

This command will initialize UCM4SVN with a set of sample data using the `http://my-repo-server/svn/test` repository.

## How to restrict repository access

UCM4SVN also contains controls to restrict users access to a repository. To make proper you of this though, it must first be configured.

There are 2 options in which to do this.

**Local:** - This first is to create the authz files locally. This can be used when the subversion server is running on the same machine as the UCM4SVN server. To enable this, simply go to the repository page and choose the repository which is hosted locally. Then change it to local mode. Now whenever something happens which will affect the access restrictions of this repository, an authz file will be created in the UCM4SVN folder, under a sub folder of authz. Within apache (or whatever service you host your subversion repository with) you will need to configure it to use this authz file. An example configuration is shown below:

```
<Location /svn>
  Dav svn
  SVNParentPath C:\svn_repos
  Require valid-user
  AuthType Basic
  AuthName "By Invitation Only"
  AuthUserFile c:\password.htpasswd
  AuthzSVNAccessFile "C:\Program Files\Apache Software
Foundation\Apache2.2\htdocs\shared\ucm4svn-testing.authz"
</Location>
```

**Remote:** - The second option is to host the subversion server on a different machine. Now a problem such as this does not stop UCM4SVN from working correctly, however it does involve a special dav\_fs module to be enabled in apache. By enabling this, UCM4SVN will then 'PUT' the authz file on the remote machine via the dav\_fs connection. An example of how you might set this up in an apache conf file is shown below. Again once this link is set up, you will need to tell apache to use the authz files created there to restrict access to the repository correctly. Notice in the above example apache is set to use the authz file from this 'shared' location.

```
DavLockDB c:\DavLock.db

<Location /shared>
## Set up the shared directory to use WebDAV and authentication

  Dav On
  AuthName "WebDAV shared Login"
  AuthType Basic
  AuthUserFile c:\password.htpasswd
  ## Limit access for enhanced security
  require user admin
  # Uncomment the below to allow access from any user with a password. Not recommended
  #require valid-user

  Order allow,deny
  ## The line below is where to allow connections from, this is normally a list of IP
addresses
  Allow from all
</Location>
```

The events which trigger the creation/updating of these authz files are as followed:

```
Component Creation
Activity Acceptance
Activity Closure
Activity Delivery
```

An example of the authz file produced is as follows. This is based on 3 components all living under the same subversion repository.

```
# ----
# This is the UCM4SVN authz file
# Make sure you setup apache properly to include this file
# For full information check the documentation
# Clearvision 2010
# ----

# ----
# Setup UCM4SVN Groups
# ----

[groups]

developers = admin, dbadmin, developer
reviewers = admin, dbadmin, reviewer
integrators = admin, dbadmin, integrator
```

```
project_managers = admin, dbadmin, pm
database_admins = dbadmin, ucm4svn,

# ----
# Standard Repository Access Levels
# ----

[/]
@developers = r
@reviewers = r
@integrators = rw
@project_managers = rw
@database_admins = rw

# ----
# Make all tags read only
# ----

[ucm4svn-testing:/Component_3_3/tags]
@developers = r
@reviewers = r
@integrators = r
@project_managers = r
@database_admins = rw

[ucm4svn-testing:/Component_2_2/tags]
@developers = r
@reviewers = r
@integrators = r
@project_managers = r
@database_admins = rw

[ucm4svn-testing:/Component_1_1/tags]
@developers = r
@reviewers = r
@integrators = r
@project_managers = r
@database_admins = rw

# ----
# Make activities writable by their assigned developers
# ----

[ucm4svn-testing:/Component_1_1/branches/activities/Project_1_1/Internal_Activity_3_3]
admin = rw

# ----
# Make projects writable by their assigned developers if developers can deliver is turned on,
# system wide project managers and integrators already have permissions
# ----

[ucm4svn-testing:/Component_3_3/branches/projects/Project_3_3]
admin = rw
dbadmin = rw

[ucm4svn-testing:/Component_2_2/branches/projects/Project_2_2]
admin = rw
dbadmin = rw

[ucm4svn-testing:/Component_2_2/branches/projects/Project_3_3]
admin = rw
dbadmin = rw

[ucm4svn-testing:/Component_1_1/branches/projects/Project_1_1]
admin = rw
dbadmin = rw

[ucm4svn-testing:/Component_1_1/branches/projects/Project_3_3]
```

```
admin = rw
dbadmin = rw
```

**NOTE:** Remember to make sure that the created authz files have the correct permissions to be accessed by the apache user, else you may experience problems.

## Troubleshooting

### Emergency Database Access

Under exceptional circumstances, if you require access to the underlying database, please use the following login credentials:

- **Username:** dbadmin
- **Password:** clearvisionucm4svn

Note that this is for emergency use only, and will normally only be used by Clearvision Technical Support.

### Resetting UCM4SVN Data

 **Using this will DESTROY all of your UCM4SVN data!**

You can reset UCM4SVN by destroying all of its data. You might want to do this with a test or evaluation system, we do **NOT** recommend ever using this on a live system (please use with utmost caution)!

Run the reset command, and tell it which installation directory to reset.

```
/home/ucm4svn/ucm4svn/tools/ucm4svn_reset_db -l /home/ucm4svn/ucm4svn
```

This command will reset the internal UCM4SVN database to an empty state. Any existing data will be backed up as a new file <DATE>-ucm4svn.db.BAK in the same directory as the original database.

## Running UCM4SVN As A Service

### On Windows

Note: These instructions for running UCM4SVN as a service on Windows are intended for advanced users. And it of course needs full privileges to the machine.

This is using the srvany software from the Microsoft NT/win2k/XP resource kits. If you don't have srvany.exe download it from [ftp://ftp.microsoft.com/bussys/winnt/winnt-public/reskit/nt40/i386/srvany\\_x86.exe](ftp://ftp.microsoft.com/bussys/winnt/winnt-public/reskit/nt40/i386/srvany_x86.exe). instsrv.exe can be found in the Windows resource kit. Place these two files either in your main UCM4SVN folder, or in your %windir%. Open a command prompt (cmd.exe) and cd your way to your installed UCM4SVN folder. Execute the command-line:

```
instsrv.exe ucm4svn srvany.exe
```

You should get this as a result:

Create Service SUCCESS at creating: ucm4svn

If you got an error "instsrv.c: Error 1057 from CreateService? on line 103" it is because your Administrator account is not named "Administrator"; simply rename the account, create the service, then rename it back to it's original. Then in the service control panel applet, you will have to manually change the Log On credentials to that of the renamed administrator account (it will subsequently grant this account the "Logon As Service" right).

If you get an error "The fully qualified path name to the .exe must be given, and the drive letter must be for a fixed disk (e.g., not a net drive)", then you need to use:

```
instsrv.exe ucm4svn C:\Program Files\Clearvision\ucm4svn\srvany.exe
Launch the regedit.exe tool, go to the following key: HKEY_LOCAL_MACHINE\SYSTEM\ControlSet??001\Services\ucm4svn
```

Add a "Parameters" key with a string key called "Application" and a value of the path to your executable. For example, C:\ucm4svn\ucm4svn.exe

Optional:

Create a new sub-key under this key: AppParameters

In the (new) AppParameters key, you can specify a path to a config file if needed, for example:

```
-f c:\Program Files\Clearivision\ucm4svn\alt_config.cfg
```

Another string value in the same key: AppDirectory Set this to the ucm4svn folder.

Go into the control panel > admin tools > services and set the settings to ones which suite you (it should perhaps be set to not interact with desktop)

## On Linux

Make a file in /etc/rc.d/init.d and call it ucm4svn. Set its permissions to +x and edit its contents to look like this:

```
#!/bin/bash
# ucm4svn      Startup Script for UCM4SVN
# description: UCM4SVN Server basic start/shutdown script
# processname: ucm4svn
# pidfile:     /var/run/ucm4svn.pid

# Source function library.
. /etc/rc.d/init.d/functions

UCM4SVN_HOME=/opt/Clearvision/ucm4svn

start() {
    echo -n "Starting ucm4svn: "
    cd $UCM4SVN_HOME
    ./ucm4svn &
    echo "done."
}
case "$1" in
start)
start
;;
*)
echo "Usage: $0 {start}"
esac

exit 0
```

Don't forget to customise UCM4SVN\_HOME to your installed directory