



Migrate2SVN

Frequently Asked Questions

Table of Contents

- 1 General
 - 1.1 What information is migrated by Migrate2SVN?
 - 1.2 How can I migrate from ClearCase to Git?
- 2 Compatibility
 - 2.1 What ClearCase versions are supported?
 - 2.2 What Subversion releases are supported?
 - 2.3 Is Migrate2SVN compatible with ClearCase UCM
- 3 Troubleshooting
 - 3.1 Why does Migrate2SVN refetch some files into the cache on subsequent runs?
 - 3.2 How can I start from a clean slate and clean out the Migrate2SVN cache?
 - 3.3 Why are tags and branches not migrated to Git?
 - 3.4 Why do I have an unexpected file/folder on my main branch in Subversion; it was created on another branch in ClearCase?
 - 3.5 I created a folder in ClearCase but it is not migrated
 - 3.6 I have a ClearCase 'Evil Twin' file that has not been migrated correctly into my SVN repository
- 4 Infrastructure
 - 4.1 Does Migrate2SVN need to run on the ClearCase and/or Subversion servers?
 - 4.2 Do the ClearCase user names need to be replicated on Subversion?
 - 4.3 How is Migrate2SVN licensed?
 - 4.4 How do I continue with Development
 - 4.5 After exporting from CC to SVN I see white space diffs in some of my files

General

What information is migrated by Migrate2SVN?

Migrate2SVN migrates the complete ClearCase version history of each file, including checkin comments, authorship and checkin timestamps.

Tags and branches can also be migrated. However, this will require additional configuration to make sure the correct data set is transferred. Please see the Migrate2SVN User Guide for details.

How can I migrate from ClearCase to Git?

If you are moving from ClearCase to Git, you can use Migrate2SVN to migrate your data from ClearCase to Subversion first, followed by a migration to Git using Git's Subversion integration (`git svn`).

Please see the Migrate2SVN User Guide for details.

However, please note that tags and branches will not be migrated to Git from a Subversion repository created with Migrate2SVN. See section "Why are tags and branches not migrated to Git" for details.

Compatibility

What ClearCase versions are supported?

Migrate2SVN has been tested with ClearCase 7. However, as only standard calls to `cleartool` are used, it is expected to work with a wide range of ClearCase releases.

What Subversion releases are supported?

Migrate2SVN has been tested with Subversion 1.5 and 1.6. Since the Subversion dump file format is intended to be compatible with a wide range of Subversion releases, future versions of Subversion should be able to read the dump file generated by Migrate2SVN.

Is Migrate2SVN compatible with ClearCase UCM

Yes, Migrate2SVN can be used with ClearCase UCM data. UCM branches and labels are represented as normal branches and labels, which can be migrated with the Migrate2SVN tool.

Troubleshooting

Why does Migrate2SVN refetch some files into the cache on subsequent runs?

Migrate2SVN uses a cache directory to hold copies of the element versions to be migrated. If the config option `check_zero_size_cache_file` is switched on, Migrate2SVN will always refetch empty files. Under certain error conditions, ClearCase may fail and leave an empty file behind when fetching element versions. Always refetching files is a precaution to catch this error condition so that a failure from a previous run does not impact on subsequent runs.

You can switch this behaviour off by setting `check_zero_size_cache_file` to `false`. However, please make sure you check the cache for inconsistencies if you encounter ClearCase errors.

How can I start from a clean slate and clean out the Migrate2SVN cache?

Migrate2SVN fetches ClearCase element versions into a staging area (cache), which is reused on subsequent runs of the tool. Once a migration has been completed, you may want to reclaim the disc space used by cleaning out the cache directory.

The location of the cache directory location is specified in the Migrate2SVN config file with the `cache_dir` option. It is safe to delete this directory at any point, provided Migrate2SVN is not running at the time.

Why are tags and branches not migrated to Git?

Git's Subversion integration (git-svn) allows you to migrate tags and branches from a Subversion repository into Git. However, as the Git branching model is effectively represented by a pointer to a particular revision in the Git repository, git-svn needs to be able to trace back any branches or tags in the Subversion repository to a single revision on the Subversion trunk.

When migrating from a ClearCase repository, where tags and branches are made up of a combination of file revisions, tags and branches need to be built up incrementally, and whilst individual file revisions can be traced back to revisions on the trunk, there is no overall revision on the trunk that represents a tag or a branch in its entirety. As a result, git-svn will not be able to perform this mapping and tags and branches will not be visible within Git.

Please note that this limitation is due to the nature of a migration from a file-based configuration management system into a repository-wide-revision based system and cannot be overcome without introducing other, much more significant, limitations.

Why do I have an unexpected file/folder on my main branch in Subversion; it was created on another branch in ClearCase?

The following scenario is the most likely explanation of this behaviour.

- An element is created with a revision on main (in ClearCase) but the element was created under a folder on another branch.
- When migrating, Migrate2SVN will place the element under branches/main (in Subversion), rather than under the branch.

This is due to the fact that the element only has a revision on main, not on the branch.

In addition, if the element is subsequently deleted from the other branch (although it may not be visible on the main branch in ClearCase) it may still appear in the Subversion repository (on the branches/main folder).

Again, this is due to the fact that ClearCase removes the link to the element on the branch, but the element will still have a revision on main.

You can check this by navigating to the file in question within ClearCase checking that version is on main, not on your branch

I created a folder in ClearCase but it is not migrated

Migrate2SVN will not migrate folders from ClearCase that have never contained files.

It is possible to see empty folders in the SVN repository after migration; however this is likely to be because a file was created and also deleted from the folder

I have a ClearCase 'Evil Twin' file that has not been migrated correctly into my SVN repository

For an explanation of the Evil Twin files in ClearCase, please see <http://www-01.ibm.com/support/docview.wss?rs=984&uid=swg21125072>

When Evil Twin files exist in ClearCase, the behaviour of Migrate2SVN will be as follows

The 'original' file's revision (before being deleted) is at a greater revision than the Evil Twin

The file contents in the SVN repository will be the same as the file contents for the 'original' file at the latest revision of the Evil Twin.

The 'original' file's revision (before being deleted) is less than the revision of the Evil Twin

The file contents will be as expected (i.e the same as the file contents at the latest revision of the Evil Twin).

Infrastructure

Does Migrate2SVN need to run on the ClearCase and/or Subversion servers?

Migrate2SVN uses `cleartool` to interact with ClearCase, and it generates a dump file to be loaded into Subversion.

You can run Migrate2SVN on any machine that has access to the ClearCase VOB through a suitable view. There is no need for Subversion

access on the host that Migrate2SVN is executed on.

Once the Subversion dump file has been generated, it can be transferred to the Subversion server to be loaded with `svnadmin load`.

Do the ClearCase user names need to be replicated on Subversion?

Although Migrate2SVN will migrate the 'user' names (for element updates etc...) from ClearCase to Subversion, it is not necessary for those users to exist in Subversion. However if you need those users to have access to the migrated Subversion repository, you will need to add them.

How is Migrate2SVN licensed?

Migrate2SVN migration is a consulting service provided by Clearvision CM. Different consulting packages are available ranging from fully serviced migration and DIY migration following initial training and setup.

The License is a fixed term based license, with a number of options available. Please see the Migrate2SVN section on [our website](#) or contact [Clearvision Sales](#) for more details.

How do I continue with Development

Due to the main differences between ClearCase and Subversion, there are certain circumstances where development can continue in the new SVN repository.

The main points to note are as follows:

1. In the resulting SVN repository there will be a number of branches, such as
 - /branches/main
 - /branches/subBranch1
 - /branches/subBranch1_myPersonalBranch (in ClearCase this would relate to /main/subBranch1/myPersonalBranch/LATEST)
 - /tags/TAG1
 - /tags/TAG2

The rules of continuing development

Most importantly you should not merge any branches into their parents, i.e.

1. you should not merge /branches/subBranch1_myPersonalBranch/ into /branches/subBranch1
2. you should not merge /branches/subBranch1 into /branches/main
3. Going forward you can branch and merge into these branches, i.e.
4. you can create a branch from /branches/subBranch1 (e.g. /branches/subBranch1_development1), do some work and then merge it back into /branches/subBranch1

The likelihood is that for your ClearCase /main/subBranch1/myPersonalBranch/ you will have a view that looks similar to:

```
element * CHECKEDOUT
element * .../myPersonalBranch/LATEST
element * .../subBranch1/LATEST -mkbranch myPersonalBranch
element * /main/LATEST -mkbranch subBranch1
```

In order to replicate this in your SVN repository, you would do the following:

Recreating in SVN what you see in CC

1. Branch from the SVN /branches/subBranch1 HEAD revision to create a new myPersonalBranch in /branches/myPersonalBranch
2. Check out the new /branches/myPersonalBranch to your local file system (e.g. /path/to/my/development/space/myPersonalBranch)
3. SVN **export** /branches/subBranch1_myPersonalBranch over the top of /path/to/my/development/space **you will have to use the --force option in SVN**
4. Continue to work on your new branch and check in your changes when finished



Do not be surprised if you see some files in this new branch that you could not see in your ClearCase view. This is possible, due to

- a. In ClearCase, you may have used a different config spec that refers to your branch
 - when checking files using this alternate config spec, the files/folder elements will be branched onto your branch
 - when you switch back to your original config spec (above) you will not see those files

For your own piece of mind, after performing the SVN steps above, you can run a tree based diff on your new SVN branch and your config spec to locate these files

After exporting from CC to SVN I see white space diffs in some of my files

This is due to the fact that in CC the file line endings are not converted to the local system type.

For example if a user on windows checked in a file (in CC) and then a user on Linux checked out that file, the line endings are preserved and the Linux user would see the extra `\r` character.

In Subversion (through the use of the `svn:eol-style` property) the line endings of the files are automatically altered to match your OS.



The types of files that have their line endings modified between systems are dependant on what autoprops you set for the migration

When comparing file content, be sure to ignore white space. If keeping the eol is imperative, consult the Migrate2SVN Documentation on how to adjust the `svn auto` properties.