
pure::variants Transformer for CVS Manual

pure-systems GmbH

Copyright © 2003-2007 pure-systems GmbH

2007

Table of Contents

1. Synopsis	1
2. Name and Parameters	1
3. Supported Family Model Elements	2
3.1. ps:cvsfile	2
4. Synchronization Algorithm	2
5. CVS Client Programs	3
5.1. Tested clients	3
5.2. Recommended clients	3
5.3. Client Authorization	3
6. Supported pure::variants Editions	4
7. TODO	4
Index	4

1. Synopsis

This pure::variants transformer module is used to synchronize local files used in variant transformation with files from a CVS repository. It currently supports checking out files according to configurable repository, branch, tag, date, version, and working directory information.

A printable version of this document is [available](#).

2. Name and Parameters

The name of the module is `cvssync` and will appear in the list of available transformer modules if installed. It accepts three parameters:

<code>client</code>	Is an optional parameter specifying the full path to the CVS client program. It defaults to <code>cvs</code> .
<code>compression</code>	Specifies the compression level to use for the file transfer. It must be a value between 0 (no compression) and 9 (highest compression). The default is 0.
<code>working directory</code>	The full path to the local working directory, i.e. the directory containing the local copies of the files managed in the CVS repository. Before a file is checked out, it is first searched in this directory. This overrides any specification of working directories with attribute <code>workingdir</code> inside the used models.

The following two parameters are not yet supported:

check version mismatch	Check if the version of the repository file described in models matches the version of the corresponding file in the local CVS working directory. It defaults to <code>false</code> .
checkout modules	Checkout the sources of modules described in models if the local working directory does not contain it. It defaults to <code>true</code> .

3. Supported Family Model Elements

The following source elements are evaluated for the files to be checked out.

3.1. ps:cvsfile

This source element has the following attributes.

repository	[Mandatory] The CVS repository from which to check out.
dir	[Mandatory] The directory part of the file to check out relative to the corresponding module.
file	[Mandatory] The file name part of the file to check.
srcdir	[Optional] The directory part of the file to check out relative to the corresponding module. To be used if source directory location differs from the target directory location.
srcfile	[Optional] The file part of the file to check. To be used if source file name differs from the target file name.
module	[Optional] The name of the module in the CVS repository.
branch	[Optional] The name of the branch in which the file to check out is managed. If not specified the HEAD branch is used.
tag	[Optional] The tag of the file to check out. Has precedence over branch
version	[Optional] The version of the file to check out. Has precedence over tag and branch.
date	[Optional] The date of the file to check out. Can be used in conjunction with branch or tag.
workingdir	[Optional] The local working directory containing the local copies of the files managed in the repository. This does not overwrite the "working directory" setting given as module parameter (see above).

4. Synchronization Algorithm

The synchronization algorithm functions as follows.

First the attributes of the `ps:cvsfile` source element are used to build the name of the local file and the name of the source file. For the local file name the values of the attributes `dir` and `file` are combined to `<dir> '/' <file>`. For the source file the values of the

attributes `module`, `dir`, `file`, `srcdir`, and `srcfile` are combined to `<module>/'/'<srcdir>/'/'<srcfile>`. If `srcdir` resp. `srcfile` is not given, the value of `dir` resp. `file` is used.

Then, if a working directory is given, it is first checked whether the local file already exists in the working directory. For this purpose the local file name is combined with the working directory to `<workingdir>/'/'<module>/'/'<srcdir>/'/'<srcfile>`. If this file exists it is copied to the transformation output directory using the path `<outputdir>/'/'<dir>/'/'<file>`. The permission and access/modification time of the copied file are preserved.

If no working directory is given or the local file does not yet exist in the local working directory, the file is checked out from the CVS repository specified in the attribute `repository`. The file revision information given as values of the attributes `branch`, `tag`, `date` and/or `version` are used to specify the revision of the file in the CVS repository to check out. The checked out file is written to `<outputdir>/'/'<dir>/'/'<file>`.

5. CVS Client Programs

The `cvssync` module works with all command line clients which follow the original command line option syntax of `cv`s 1.11 from www.cvshome.org

5.1. Tested clients

<code>cv</code> s = 1.11.20	works (Win32/Cygwin/Linux/MacOS X)
<code>cv</code> s = 1.12.12	works (Win32/Cygwin/Linux/MacOS X)
<code>cv</code> s <= 1.11.17	under certain conditions checkouts for symbolic tags fail (see below)
<code>cv</code> snt 2.5.01.1976	- under certain conditions checkouts for symbolic tags fail (see below)

CVS Bug with symbolic tags

`cv`s in older version has a bug which may cause failed checkouts of files to standard out by giving symbolic tags. The command `"cv`s `co -p -r <sym_tag> <filepath>"` fails with an error message (`cv`snt) or with a failed assertion (`cv`s). This is reported at least for `cv`s as bugs [#186](#) and [#211](#). Checking out at least one tagged file with `"cv`s `up -r <sym_tag> <filepath>"` once into any working directory solves the issue. For `cv`snt it is also possible to manually edit the `CVSROOT/val-tags` file and put all relevant tags in this file.

5.2. Recommended clients

`cv`s 1.11.20 or newer from <http://www.cvshome.org>

5.3. Client Authorization

The `cvssync` plugin relies on external authorization of the client. It is therefor currently not possible to use `cvssync` when during the checkout if the used `cv`s client program requests a username and/or password interactively.

When connecting via a ssh encrypted connection, the use of public key authentication with

ssh-agent (OpenSSH) or plink/pageant (PuTTY) solves this issue. When using the `:pserver:` protocol, use `cvs -d <repository> login` to store the password in cvs's password cache.

6. Supported pure::variants Editions

This plugin works only in a full-featured pure::variants and pure::variants (Evaluation).

7. TODO

Update the `ps:cvsfile` source elements according the revision of the files checked out.

Find a suitable workaround for CVS bug described above.

Index