# Subversion

Eelco Dolstra

Center for Software Technology

Utrecht University

eelco@cs.uu.nl

October 31, 2003

## Overview

- CVS: Problems
- Subversion
- `svn.cs.uu.nl`

## CVS: Problems

- Directories are not versioned.
  - I.e., files cannot be renamed / moved / copied while retaining history.
  - No refactoring of the file system structure of a project.
- Networking was hacked onto it.
  - Originally assumed FS access to the repository.
  - For network access:
    - `pserver`: not very secure.
    - Tunnelling through `ssh`: secure, but requires system accounts (or at least SSH keys) for every user.

- Extension to RCS.
  - Per-file revision numbers are not meaningful.
  - No way to identify project revisions except by tagging.
- Commits are not atomic.
  - No ACID properties on commit: they should succeed entirely or not at all.
  - No consistency guarantees on checkout: the result might be mixed-revision.
- Branching / tagging.
  - Expensive ($\Theta(n)$ space and time complexity).
  - Not so intuitive (it's another dimension).
- Minor problems.
  - E.g., output of `cvs status` is useless (too verbose).

## Subversion


- A version management system intended to be a "compelling replacement for CVS in the open source community".
- Free software: `http://subversion.tigris.org/`.
- Fixes CVS's most painful problems.
- Model and command set similar to CVS.
- Still in "alpha" stage, but quite usable.

- Runs on Windows, Unix (Linux, *BSD, Mac OS X).
- Fairly easy to install.
- Excellent manual.
- Several GUI frontends.
  - RapidSVN (portable, wxWindows).
  - TortoiseSVN (Windows Explorer shell extension).
  - ...
- `cvs2svn` repository converter available.

## Basic model

- Like CVS: central repository, developers *check out* private workspaces (*working copies*).
- Repositories and paths within them are identified using URLs, e.g., `https://svn.cs.uu.nl:12443/repos/trace/nix/trunk/`.
- Obtaining a working copy:

```
$ svn checkout \
 https://svn.cs.uu.nl:12443/repos/trace/nix/trunk/ nix
A  nix/substitute.mk
A  nix/AUTHORS
...
Checked out revision 438.
```

## Common CVS-like commands

Just like CVS (well, almost):

- `svn commit`: commit changes.
- `svn update`: update working copy.
- `svn add`: schedule a file for addition.
- `svn remove`: schedule a file for addition.
- `svn log`: show log messages for a file.
- `svn diff`: show differences between files/revisions.
- `svn merge`: merge differences between files/revisions.
- Most commands work on working copies as well as URLs, e.g.,

```
$ svn log foo.c
$ svn log http://svn.cs.uu.nl.../foo.c
```

## Working copy status

`svn status` is much more useful than `cvs status`:

```
slides/subversion$ svn status
A       subversion
A       subversion/server-repos.png
A  +    subversion/repo-wcs.fig
A  +    subversion/repeated-merge.fig
A  +    subversion/Makefile
D       security
D       security/Makefile
```

## Versioned directories

- Files (including directories) can be moved / renamed / copied.
- `svn move` / `svn rename`

```
$ svn mv doc docs
A         docs
D         doc
```

- `svn copy`
  - Copies maintain ancestry information.
- Meta-information on files is also versioned (e.g., executable bit, MIME type).

## Branching / tagging

- Very different from CVS: branches and tags are created by making a *versioned copy*.
- Copying a directory takes $O(1)$ time / space.

## Repository organisation

- Subversion does not require any specific repository layout, but a common layout is like this:

```
/projectX/trunk/
/projectX/branches/
/projectX/branches/1.x-stable/
/projectX/branches/experimental/
/projectX/tags/
/projectX/tags/1.0/
/projectX/tags/1.1/
/projectX/tags/2.0-beta/
/projectX/tags/2.0/
/projectX/tags/2.0.1/
```

## Repository organisation (cont'd)

- The *trunk* is the main development branch.
- The *branches* directory contains non-mainline development (such as bug-fix-only development, or experimental changes). These are copies of the trunk or another branch at some point.
- The *tags* directory contains *releases*; these never change.
- I.e., the only difference between a tag and a branch is that you don't commit on a tag (*policy*).

## Branching / tagging (cont'd)

- Example: you have just released version 2.0.2. The trunk can be tagged by doing:

```
$ svn copy http://.../projectX/trunk \
           http://.../projectX/tags/2.0.2
```

- Branching is exactly the same:

```
$ svn copy http://.../projectX/trunk \
           http://.../projectX/branches/2.0.2-fixes
```

- Working copies can be *switched* to another branch:

```
$ svn switch http://.../projectX/branches/2.0.2-fixes
```

## Revision numbers

- Revision numbers apply to the repository as a whole; no per-file revisions.
- Every commit creates a new repository revision: every commit is uniquely identified. Thus, change sets are unambiguous (unlike CVS).
- E.g., to see the difference between two revisions:

```
$ svn diff -r1000:1001 \
 https://svn.cs.uu.nl:12443/repos/StrategoXT
===================================================================
--- trunk/StrategoXT/srts/configure.in  (revision 1000)
+++ trunk/StrategoXT/srts/configure.in  (revision 1001)
@@ -16,7 +16,7 @@
...
```

- To undo the most recent commit (through a working copy):

```
$ svn merge -r HEAD:PREV .
U  src/fix-ng/Makefile.am
D  src/fix-ng/parser.cc
...

$ svn commit
Sending        src/fix-ng
...
Transmitting file data ......
Committed revision 439.
```

## Repository access methods

- `http` / `https`: access a repository through WebDAV / HTTP.
  - The principal access method.
  - Server implemented as an Apache module.
  - Therefore has all Apache's features, e.g., authorisation / authentication / access control modules.
  - Also all of HTTP's features, e.g., encryption, compression, pipelining, caching, proxying.
  - WebDAV is a standard protocol supported by many other products (e.g., Mac OS X can mount a Subversion repository as a native file system).
  - Takes some effort to set up.

- `file`: access a repository in the local file system.
  - Trivial to set up:

```
$ svnadmin create /home/eelco/my_repo
$ svn co file:///home/eelco/my_repo
```

- `svn`: light-weight Subversion-specific protocol. Can be used anonymously (`svn://server/home/eelco/my_repo`) or tunnelled through SSH (`svn+ssh://server/home/eelco/my_repo`).

## Safety

- Built on top of Sleepycat's Berkeley DB, a transactioned embedded database library.
- ACID semantics: ensures atomicity of commits.
- Hot backups, incremental dumps.

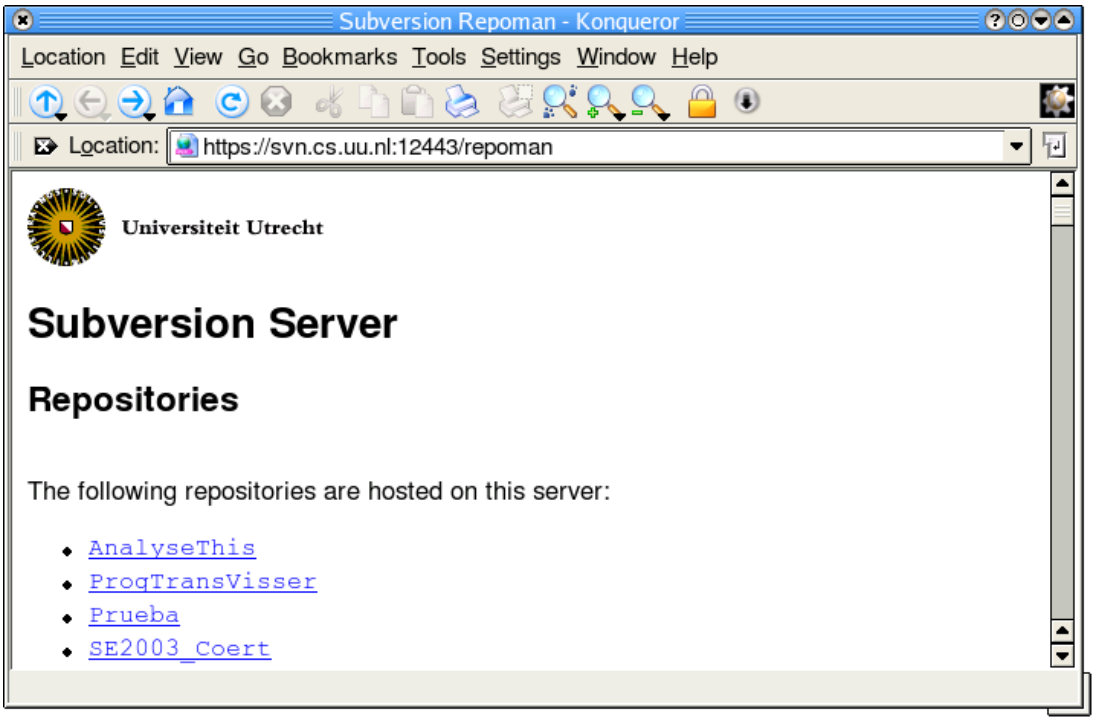
## What Subversion doesn't (yet) do

- No improved merge support (still has the "repeated merge" problem). Planned for post-1.0.
- No peer-to-peer architecture (like BitKeeper, Arch, darcs).

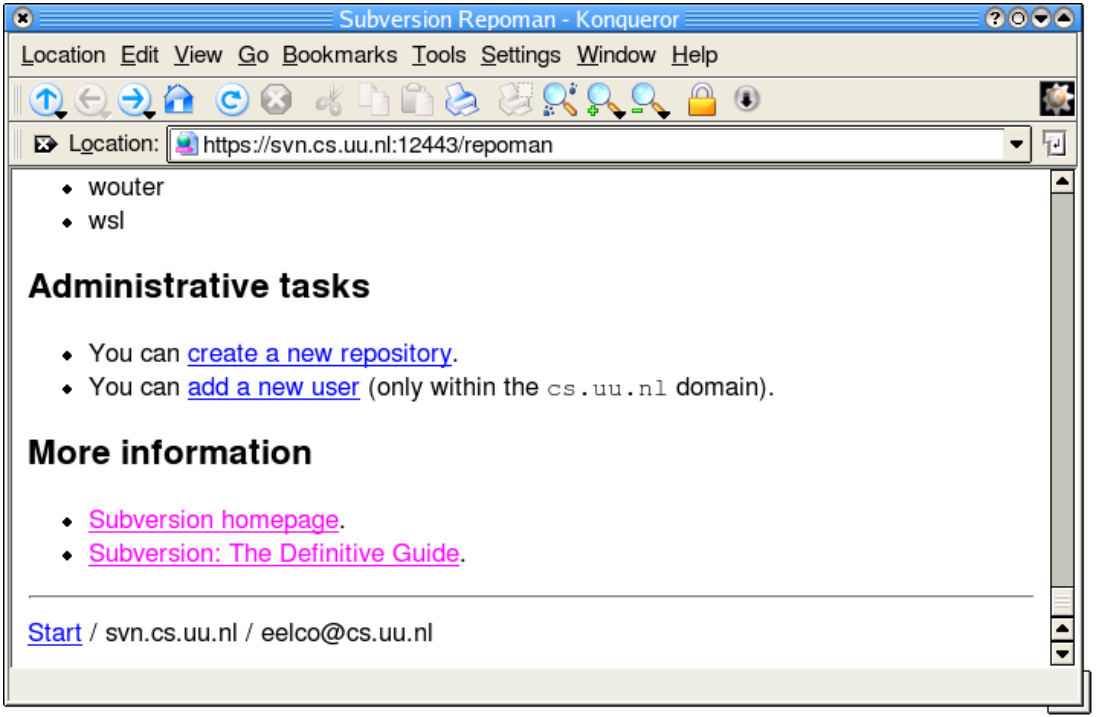
## svn.cs.uu.nl

- We have a Subversion server running at `http://svn.cs.uu.nl/`.
- Anyone within the `cs.uu.nl` domain can create user accounts / repositories.
- Repositories can be accessed (with proper access rights) from anywhere in the world.
- Simple web interface.

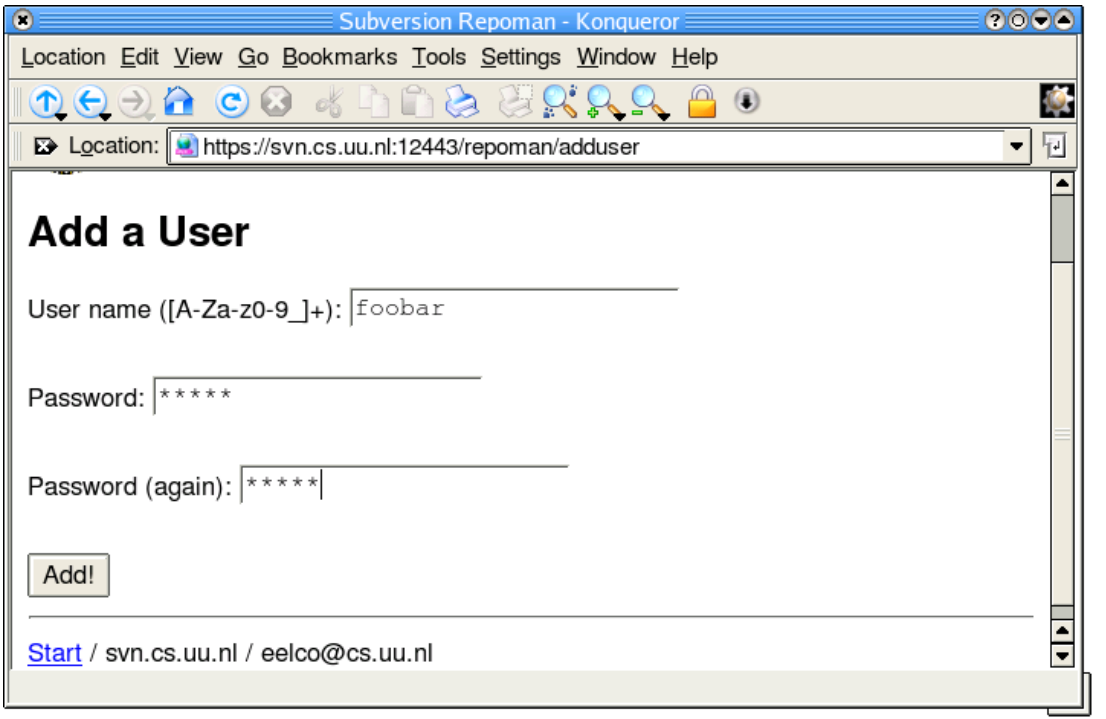
# Main page


Subversion Repoman - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: https://svn.cs.uu.nl:12443/repoman
Universiteit Utrecht
Subversion Server
Repositories
The following repositories are hosted on this server:
AnalyseThis
ProgTransVisser
Prueba
SE2003_Coert

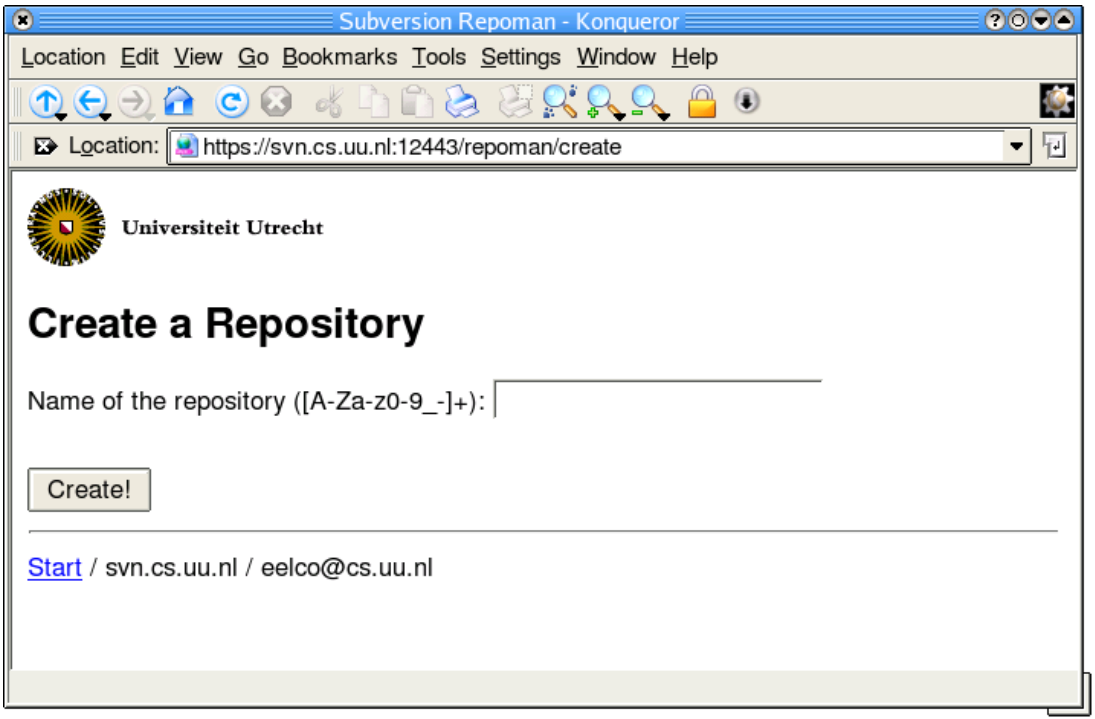
# Main page (cont'd)



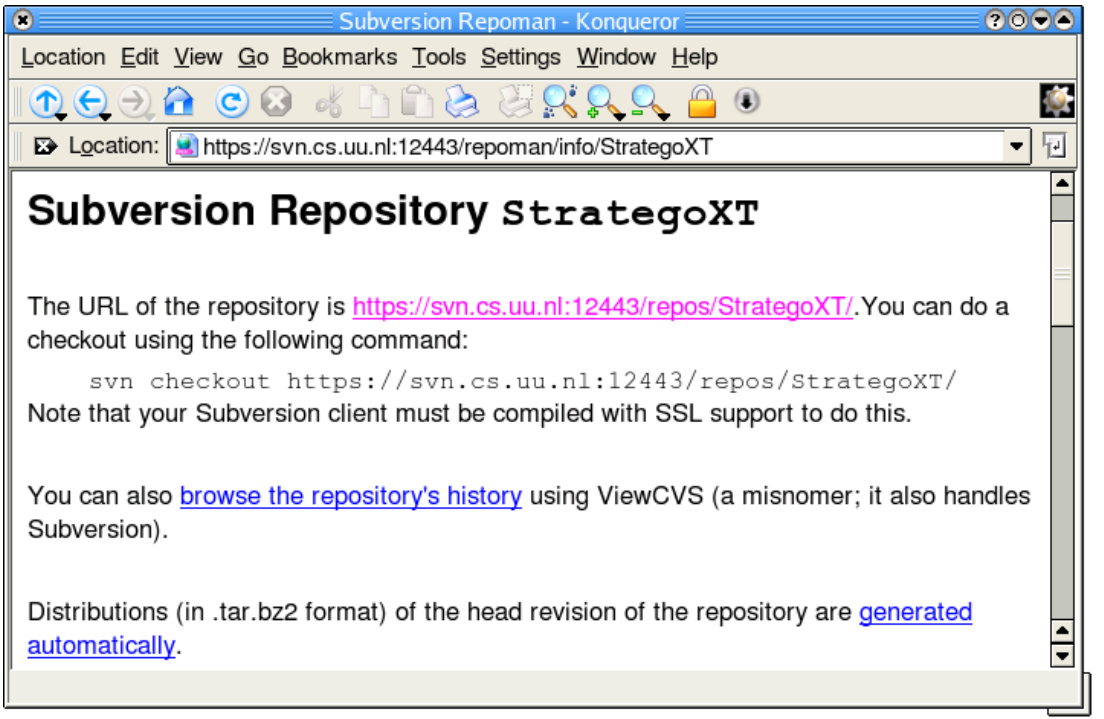
# Creating a new user



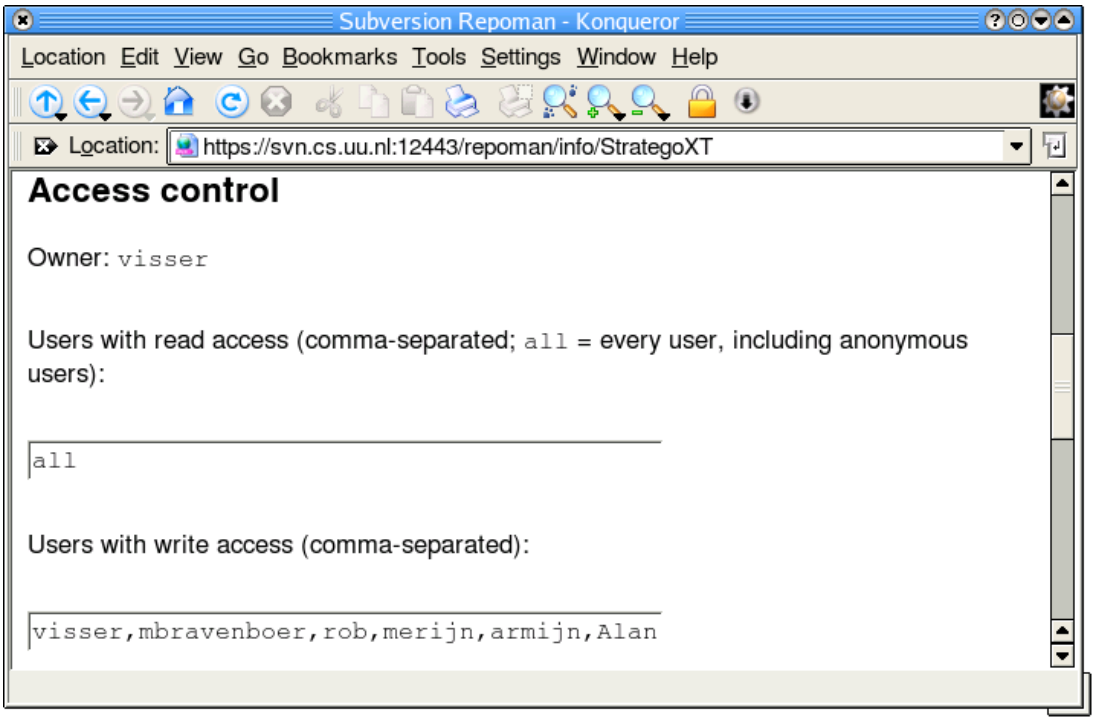
# Creating a new repository



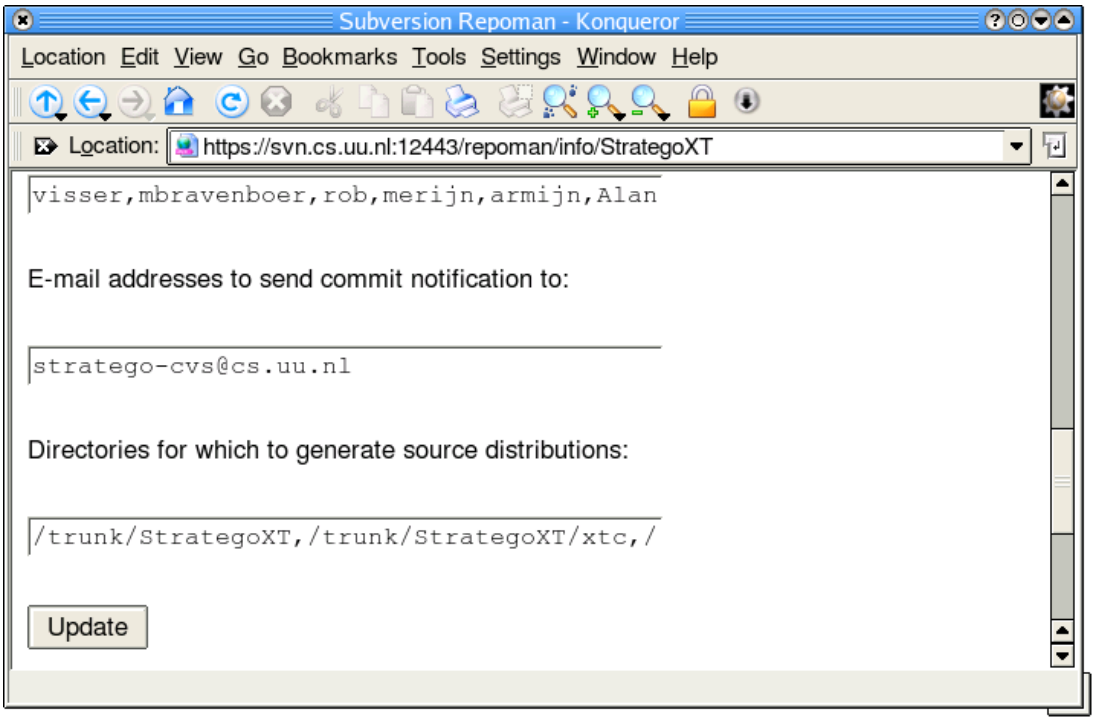
# Repository info



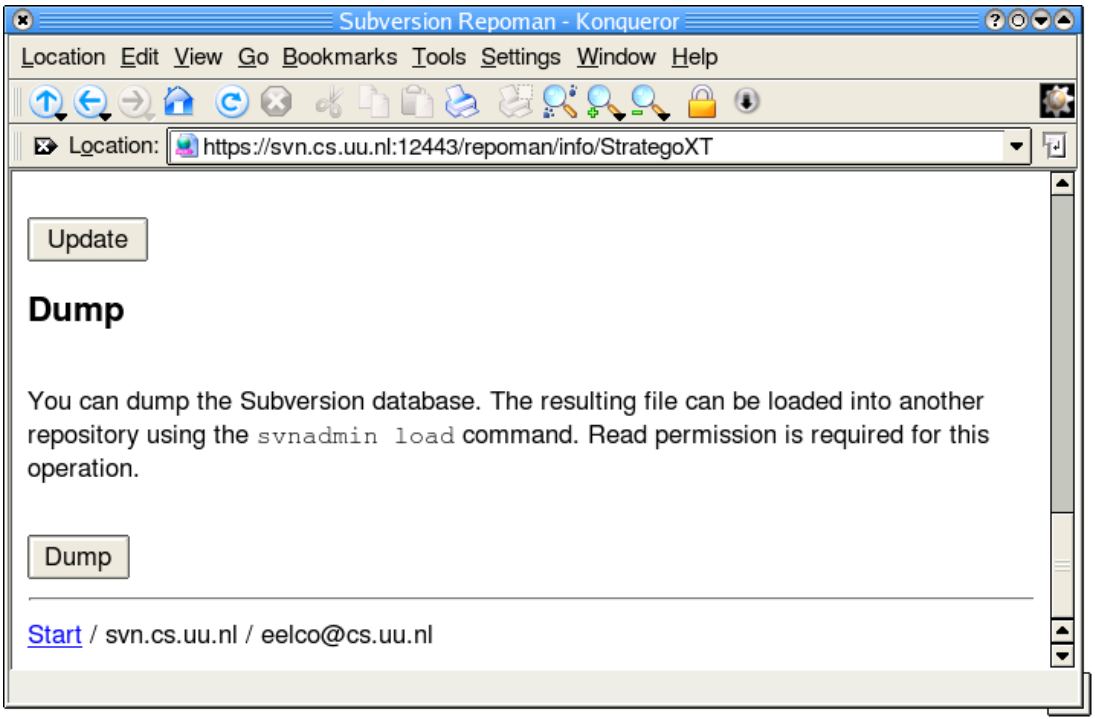
# Repository: access control



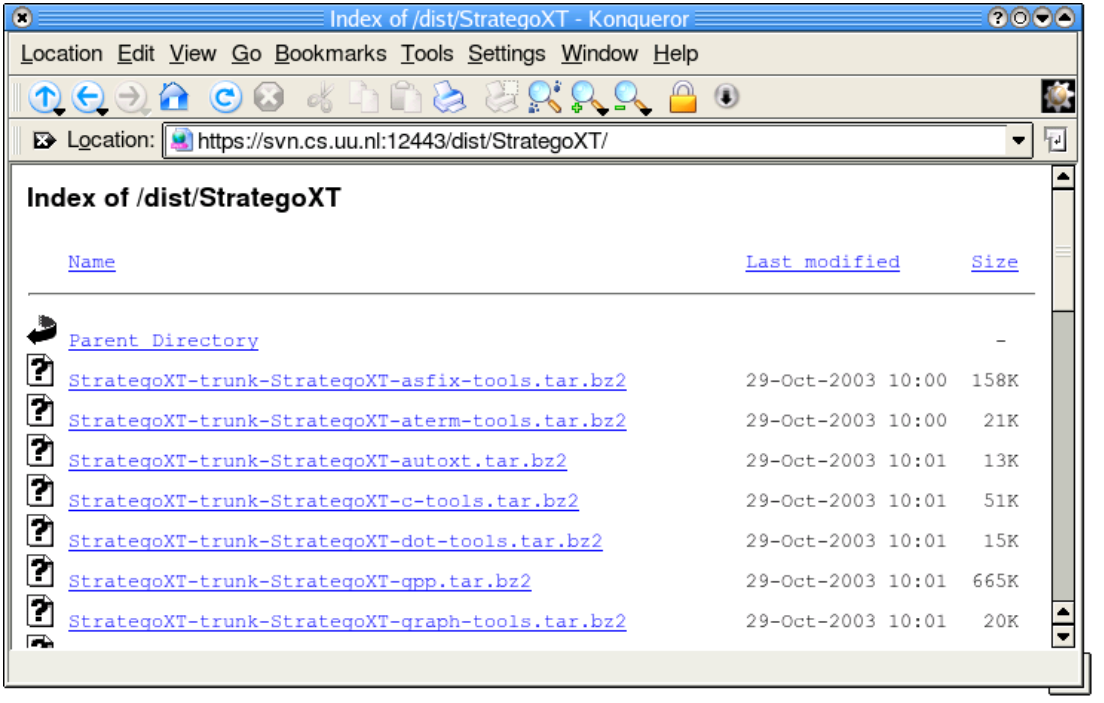
# Repository: commit notification



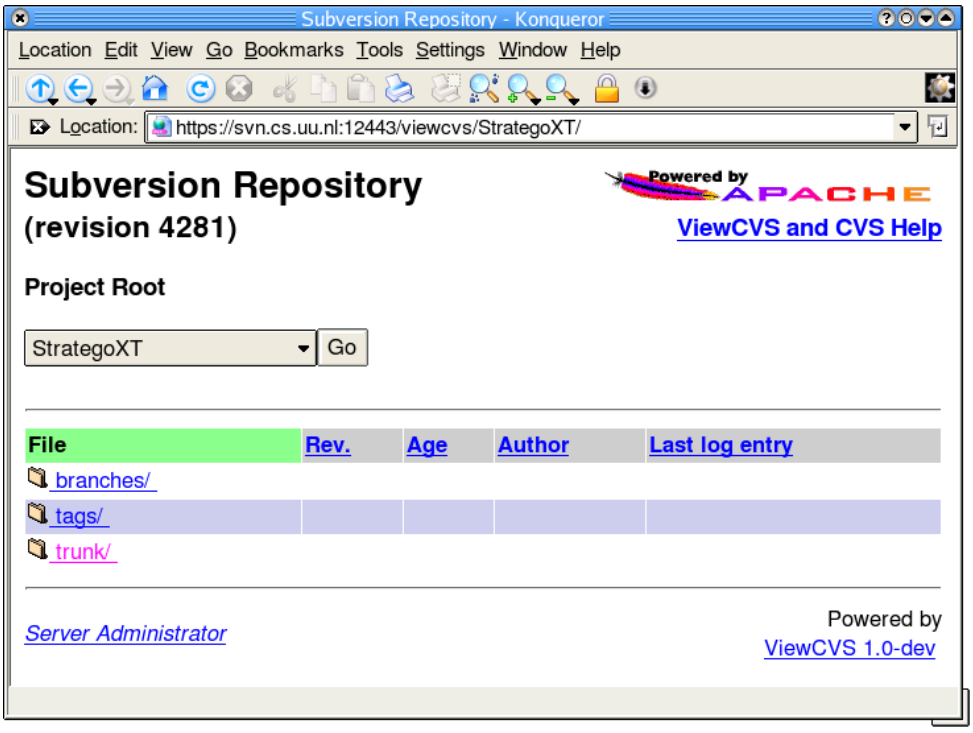
# Repository: dumping



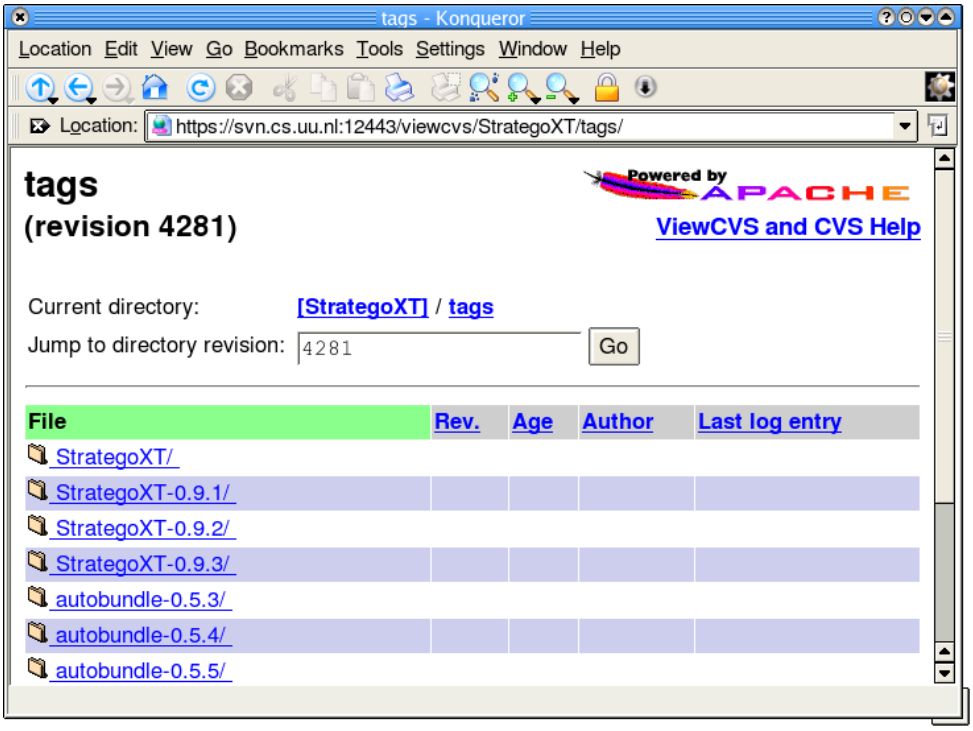
# Automatic source distributions



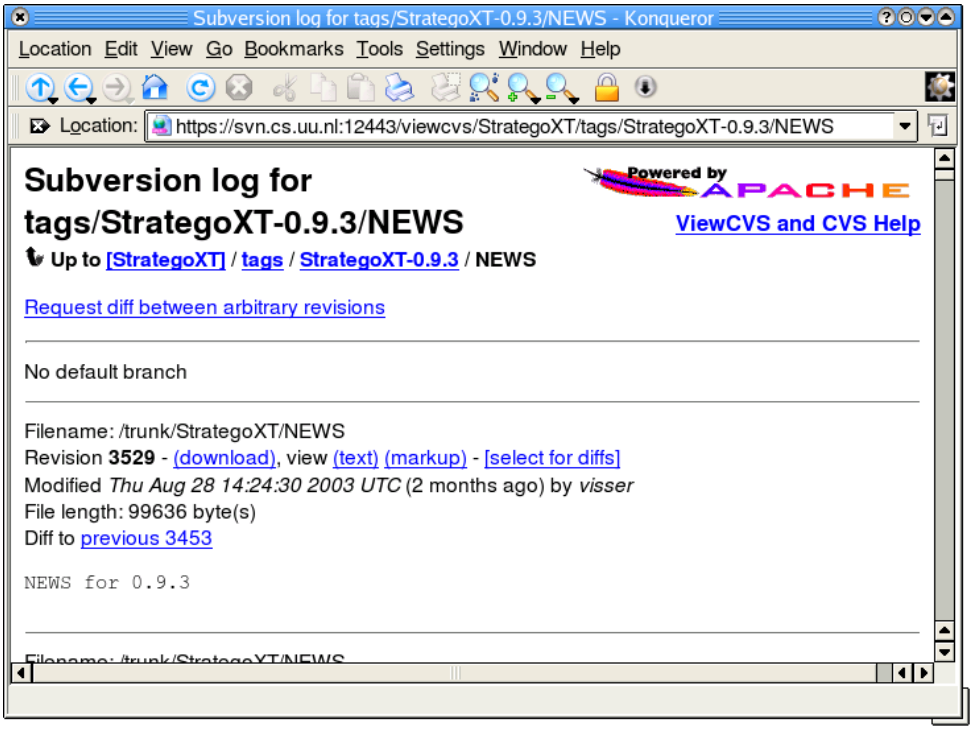
# ViewCVS


Subversion Repository - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: https://svn.cs.uu.nl:12443/viewcvs/StrategoXT/
Subversion Repository
(revision 4281)
Powered by APACHE
ViewCVS and CVS Help
Project Root
StrategoXT
Go
File
Rev.
Age
Author
Last log entry
branches/
tags/
trunk/
Server Administrator
Powered by
ViewCVS 1.0-dev

# ViewCVS (cont'd)



# ViewCVS (cont'd)



## Conclusion

- It's better.