

Memo

Aan
Projectmedewerkers PMR VNC

Datum
16 september 2009

Aantal pagina's
12

Van
Loana Arentz

Doorkiesnummer
(088) 33 58 234

E-mail
loana.arentz@deltares.nl

Onderwerp
Getting started with the subversion repository.

The ideas behind a subversion repository in a nutshell

The risk

Multiple parties, multiple data, multiple purposes easily results in a chaotic database. This can be done better. Currently :

- All collaborating, but not enough
- Coordination needed
- But no overall boss
- Like wikipedia

This is illustrated in *Figure 1*.

The solution

Optimal is possible with a subversion REPOSITORY (see *Figure 2*).

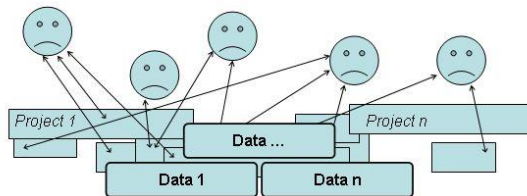


Figure 1. The risk: chaos

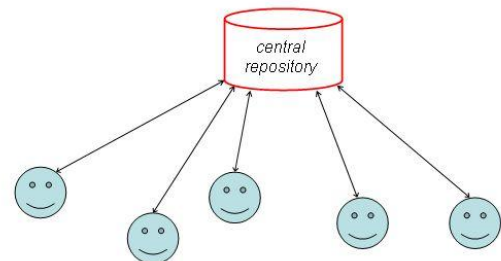


Figure 2. The solution: subversion repository
(image © G.J. de Boer)

The principle:

Subversion repository basics:

1. Get username and password.
2. For best quality, all actions are logged (see example below).
3. nothing can be lost, only temporarily disabled
4. so anyone can be allowed to join

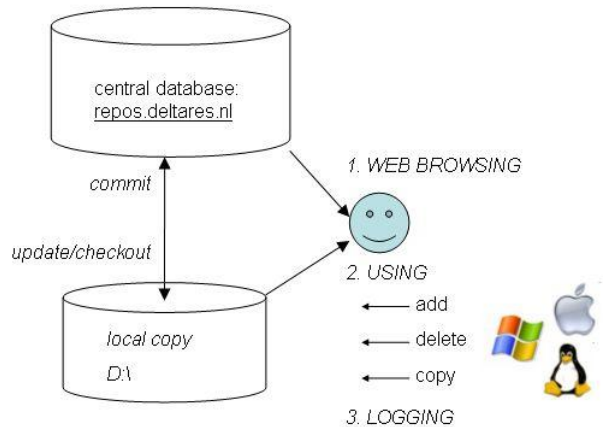


Figure 3. The principle (image © G.J. de Boer)

It is important to store all relevant data. Please note that Data = raw data + processing:

For example:

NASA satellite data with open source SeaDas processing toolkit (in IDL)

- L0: dump of recorded voltages, only averaged over 16 pixels
 - LAC: MLAC
 - GAC
- L1: voltages + satellite track
- L2 ~ physical quantities
- L3 ~ binned in space (1 grid instead of zillions of warped photos)
- L4 ~ binned in time (climatology)

We save all relevant sublevels.

A subversion repository: How does it work?

Deltares stelt een centraal 'repository' (uitwisselplatform en tevens archief) in waarin datasets voortschrijdend (d.w.z. in versies) kunnen worden beheerd (opgeslagen, benaderd, geupdate). Dit archief zal voor alle projectpartijen toegankelijk zijn: leesrechten voor de specifieke afnemers, schrijfrechten alleen voor de relevante partijen binnen het betreffende perceel. Uploads kunnen plaatsvinden zodra een partij zijn data kan en wil vrijgeven (na afronding van bepaalde campagne en QA, maar in ieder geval altijd voordat een volgende partij deze informatie nodig heeft). In de repository kunnen uiteraard naast de data files, ook gehele databases als werkdocumenten en afgeronde rapporten, wetenschappelijke papers en Powerpoint files opgeslagen en beheerd worden.

Structure of the repository

Here, we will illustrate the global structure of the repository. The structure was set-up for modifications in a.o. source code of Delates software. Branches and Tags are specifically designed for such software developments and have no purpose for data management. We only work in the trunk.

Trunk

The trunk contains the main version of the data. This is where all (short) developments by all users take place. As we are only dealing with short developments this is where we work

Branches

Separate directory e.g. for code developments. Not used here.

Tags

Also for code developments. Not used here.

Subversion (SVN)

Introduction

SVN is a versioning tool. It can be used for keeping track of the history of practically anything. We use it a.o. for the data in the project tree. SVN is especially useful when multiple users work on the same data. SVN makes sure that the modifications made by one person are not overwritten by those of another person, even when they are modifying at the same time. On a server machine, a source code repository has been created, which can be accessed by everyone with an internet connection and admission to the repository.

For PMR NCV the repository is <https://repos.deltares.nl/repos/PMR-NCV/>

Several clients are available to make use of the SVN. The one most frequently used (at Deltares, on Windows) is TortoiseSVN. Other SVN clients are also available, especially for use on UNIX. The remainder of this memo will focus on the use of TortoiseSVN.

Download and installing TortoiseSVN

<http://tortoisesvn.net> → download → 32 bit (for most regular desktops).

DEMONSTRATION

Now we will illustrate the use of SVN. We will illustrate this for the use on Windows, using TortoiseSVN. Be sure to have this installed on your PC.

An example is given on how to edit data in the repository. This can be done directly in the trunk. Below we shall illustrate the different steps to take and the possibilities SVN offers.

SVN Checkout: creating a local copy of (part of) the repository

First, we must obtain a local copy of the part of the directory, which we want to modify. We must therefore *checkout* that part of the trunk of the repository, which we are interested in. Execute the following steps:

- Create a directory where you want to checkout the source code, e.g. E:\Checkouts\
 - In this directory right click and select:
 - TortoiseSVN -> Checkout*
 - or (when available in the right-click menu):
 - SVN Checkout*
 - In the Checkout window:
 - Specify the location of the source code in the repository.
For the data-trunk this is:

<https://repos.deltares.nl/repos/PMR-NCV/trunk/>

or on a sublevel:

https://repos.deltares.nl/repos/PMR-NCV/trunk/004_Abiotiek/

- Make sure the Checkout directory is the directory where you want your copy of the source code
- Make sure the checkout depth is set to *Fully recursive*
- Determine the revision, which you want to checkout. The HEAD revision is the newest version available. Using the *Show log* button you can search for other (older) revisions.
- Click *OK* to checkout the source code to your local directory.
- Now you can modify the data. SVN will keep track of your changes locally on your PC. It stores information in subdirectories named “.svn”.
Please note that the checkout also creates a shadow image locally on your pc. If you checkout 20Mb, you need 40 Mb of free disk space.

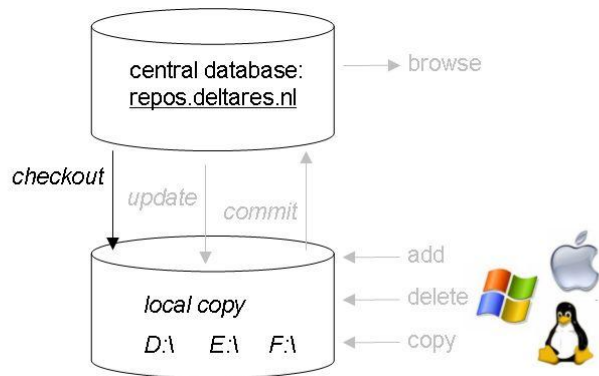


Figure 4 SVN checkout (image © G.J. de Boer)

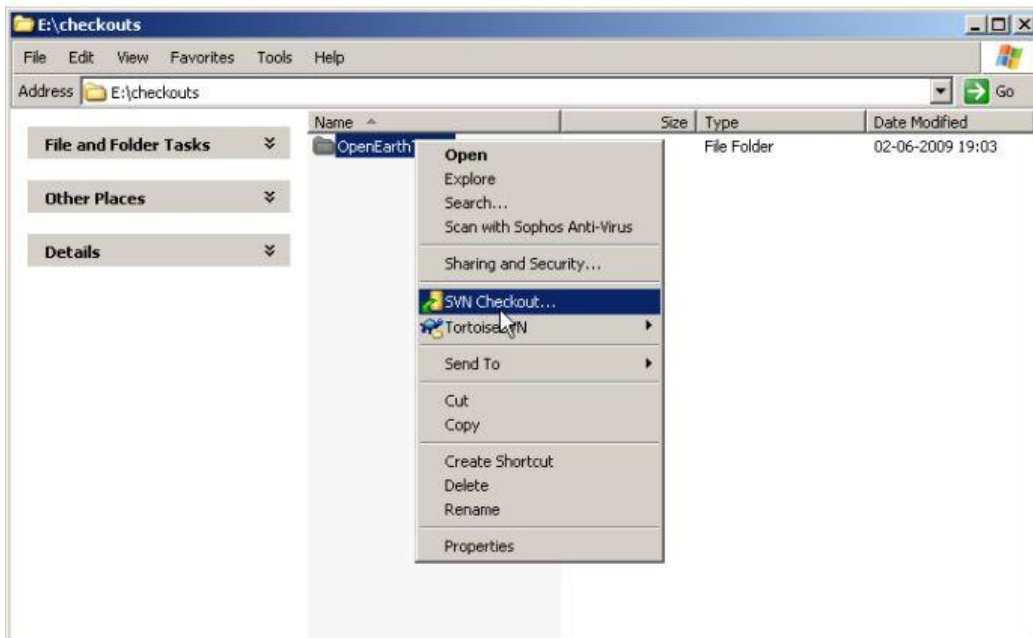
Demo

This demo is for the “OpenEarthTools” repository.

Step 1:

Not handy to get files one by one with browser; get them all at once with free program (SVN tortoise)

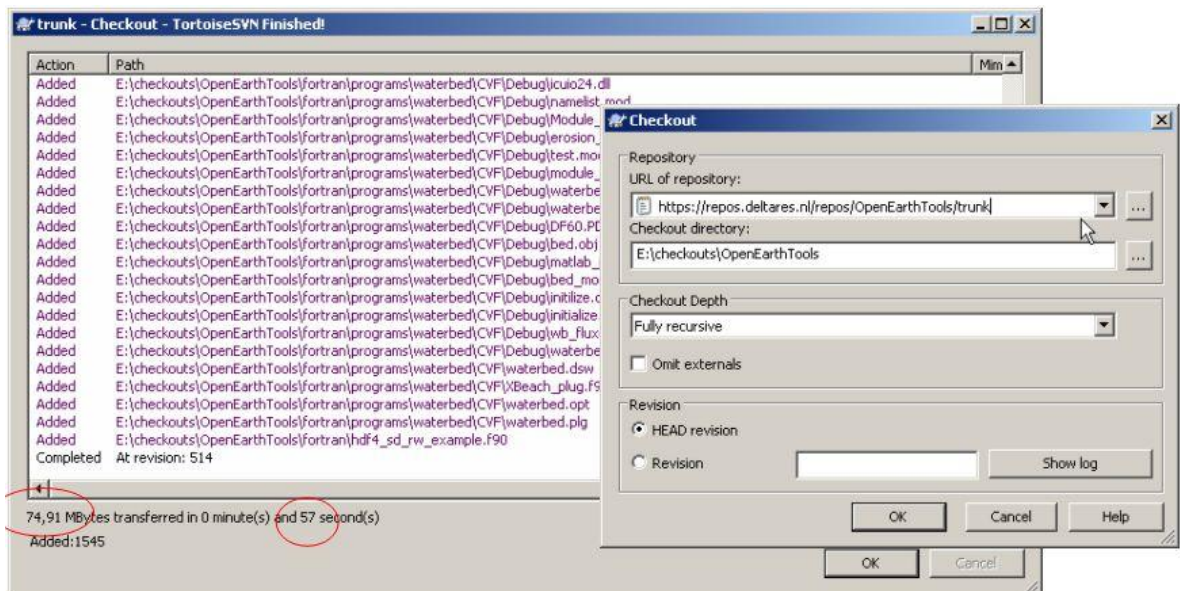
- Make a checkout in e.g. E:\checkouts\
• No need to back this up, it’s only a copy ...



(image © G.J. de Boer)

Step 2:

- Copy url from browser (case sensitive!)
- Make sure that tree of local copy resembles server



(image © G.J. de Boer)

Making adjustments in the checkout: editing the data

Important: modifying file or directory names requires special attention, see SVN rename. So does adding or removing files, see SVN Add/SVN delete.

SVN rename

This is done by right-clicking the file and selecting:

TortoiseSVN -> Rename

SVN rename is different from a simple rename. It makes sure that SVN sees the old file (with a different name) as the predecessor of the new file (with the new name). It makes sure that with the next commit, the file is also renamed in the repository, such that when other users of the repository checkout the source code or update their local copy, the same file is also renamed in their version of the source code.

Simply renaming files causes SVN to loose track of the history of that file. If a file is renamed (not using SVN rename) SVN will think it is missing.

SVN add

This is done by right-clicking in the containing directory and selecting:

TortoiseSVN -> Add

Using SVN add you can add new files to the repository. The same holds here as for the SVN rename command. Simply adding a file in a certain directory will not do the trick. You must tell SVN that there is a new file for which it will have to keep track of its history. This is done using SVN add. The file will get a '+' as Icon overlay (see Section Settings).

SVN delete

This is done by right-clicking in the containing directory and selecting:

TortoiseSVN -> Delete

For the SVN delete command the same restrictions hold as for SVN rename and SVN add. Simply deleting a file will not be enough for SVN. It will think the file is missing and put it back the next time you update. With an *SVN delete* you actually tell SVN to delete the file from its repository. Its history will be kept, but from this revision on the file will no longer be included in the source.

With an SVN delete the file will get an 'x' as Icon overlay (see Section Settings).

Repo-browser

This is done by right-clicking in the containing directory and selecting:

TortoiseSVN -> Repo-browser

The Repo-browser (Repository Browser) allows you to quickly browse through the repository. You can view the history (log) of the trunk and branches, you can search for certain revision numbers (top right) and you can checkout from the repository browser (by right-clicking).

Show log

This is done by right-clicking in the containing directory and selecting:

TortoiseSVN -> Show log

The log shows you the history of the source code. You can see which files have been modified, when the modifications were done, by whom and you can view possible comments that were added with the commit of that revision.

Comparing your local copy with the repository

Using *SVN diff* or *Check for modifications* you can investigate the status of your local copy, i.e. which files did you modify? Did you add/delete files? Are there any files, which have been modified in the repository (by other people) with respect to your local copy?

SVN Update

Using *SVN update* you can merge the HEAD revision of the repository into your local copy. Always perform an SVN Update before committing your changes. Updating is done as follows:

- In the directory where you have your local copy of the source code, which you wish to update, right click and select:
 - *TortoiseSVN -> Update*
 - Or (when available in the right-click menu):
SVN Update
 - SVN will now search all files (recursively down from the directory where you are) in the repository that have been changed with respect to your local version.
 - SVN will merge these changes into the local version.
 - SVN will also detect whether one of the files you have edited, has also been edited by someone else. SVN will try to merge the changes. If SVN does not succeed, it will report a conflict that you have to resolve. See *Resolving possible conflicts*.

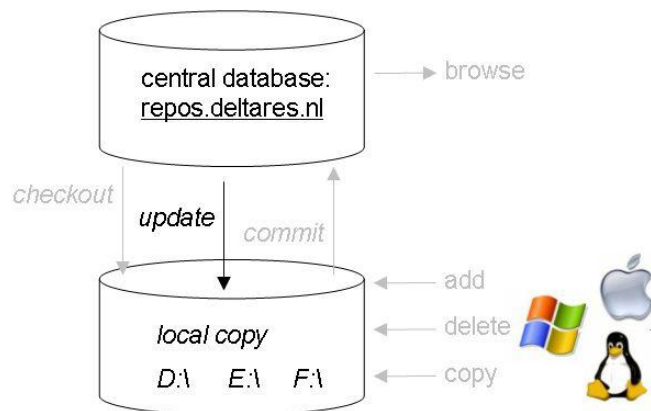


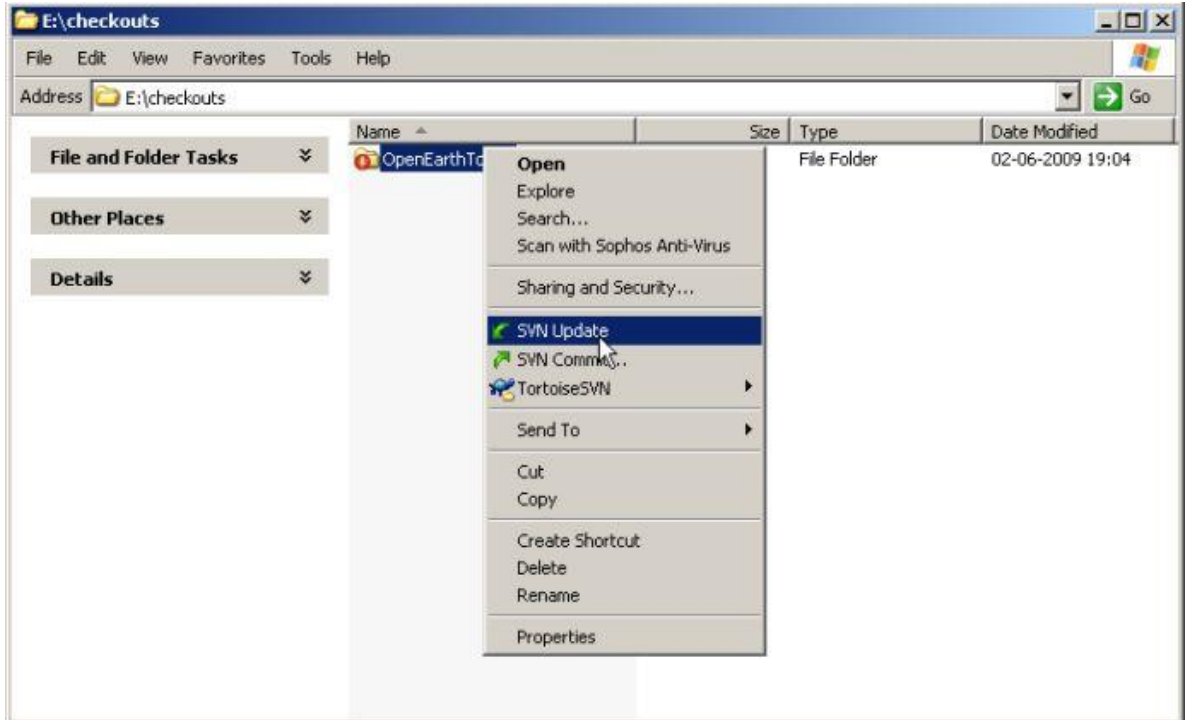
Figure 5 SVN update (image © G.J. de Boer)

Datum
16 september 2009

Pagina
8/12

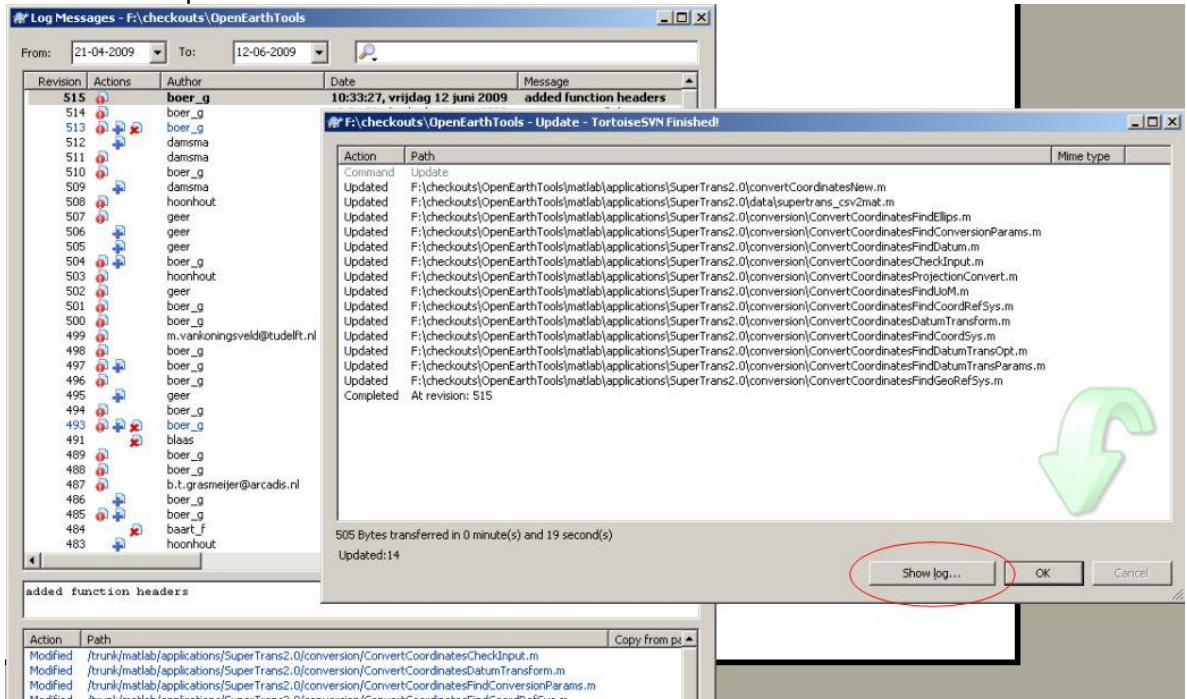
Demo

This demo is for the "OpenEarthTools" repository.



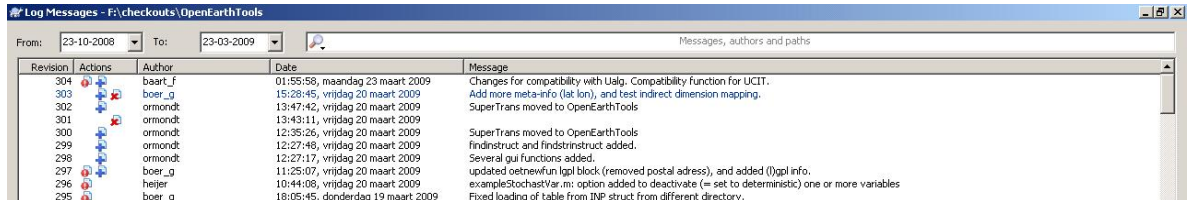
(image © G.J. de Boer)

Result of the update:



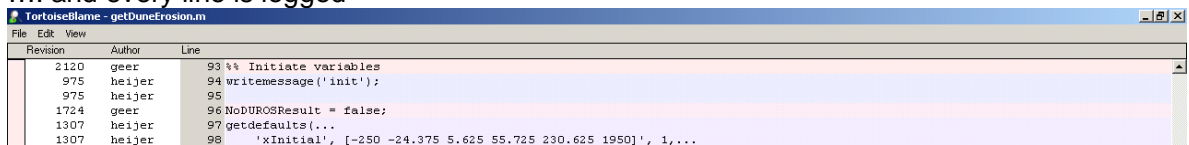
(image © G.J. de Boer)

Every file is logged ...



(image © G.J. de Boer)

.... and every line is logged



(image © G.J. de Boer)

SVN Commit

Using SVN commit you can feed your modifications to the repository on the server. After a Commit, your modifications are available to everyone using the same repository. Be sure to always do an update before committing. This will avoid conflicts when SVN tries to merge your local copy with the repository version. Committing is done as follows:

- In the directory where you have the modified source code you wish to feed to the repository, right click and select:
 - TortoiseSVN -> Commit
 - Or (when available in the right-click menu):
SVN Commit
 - SVN will now search all files (recursively down from the directory where you are) in the local version that you have changed and show them in a pop-up window. With check boxes, you can decide what changes have to be committed.
 - You can add a message (please try to add a message always)
 - When clicking OK, SVN sends your changes to the repository on the server machine and merges them.

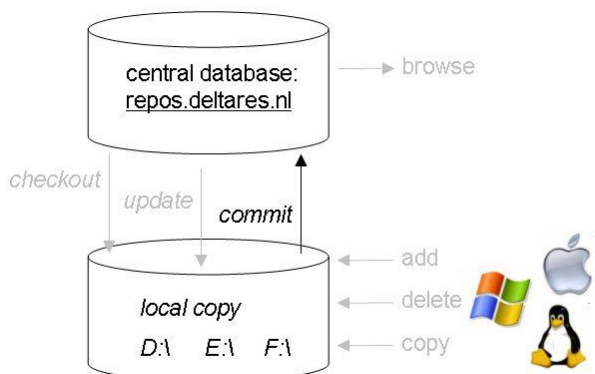
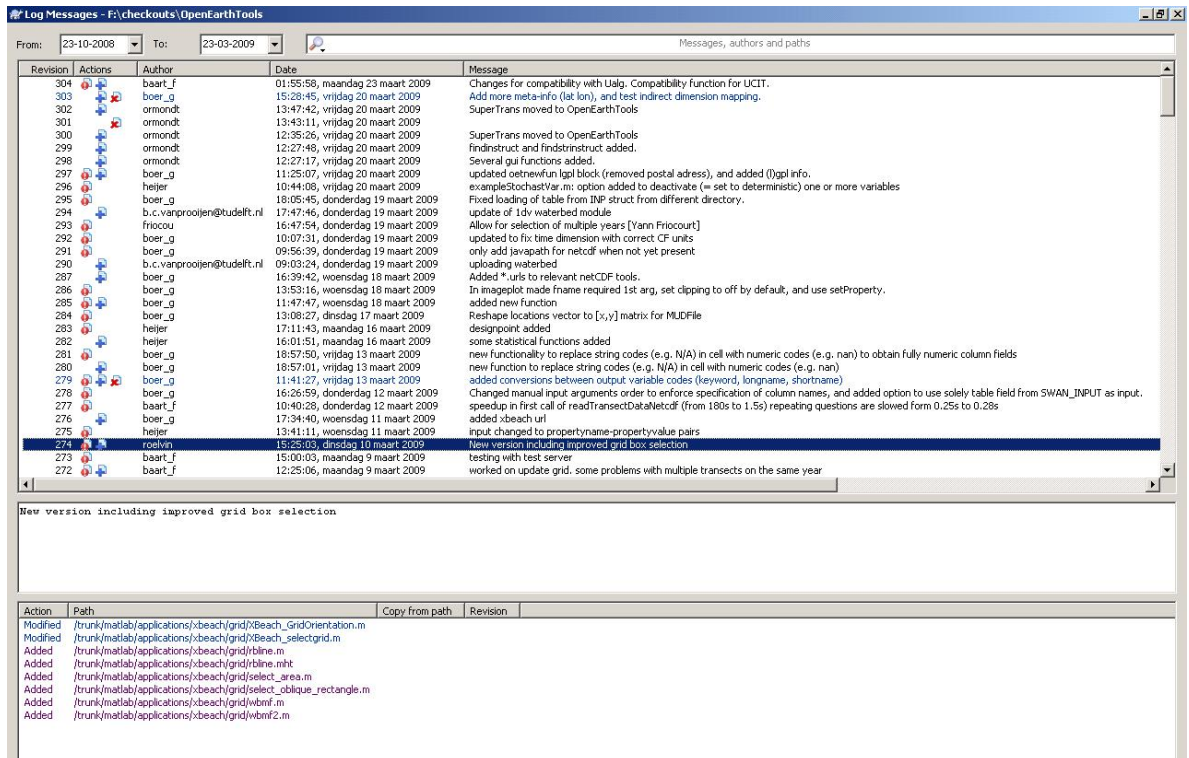


Figure 6 SVN commit (image © G.J. de Boer)

Users 'commit' their files in one central database (regular update local copy). Every commit gets a unique revision number. Per commit one can add a comment to indicate what was changed. *Please do add a line to describe what you did.*

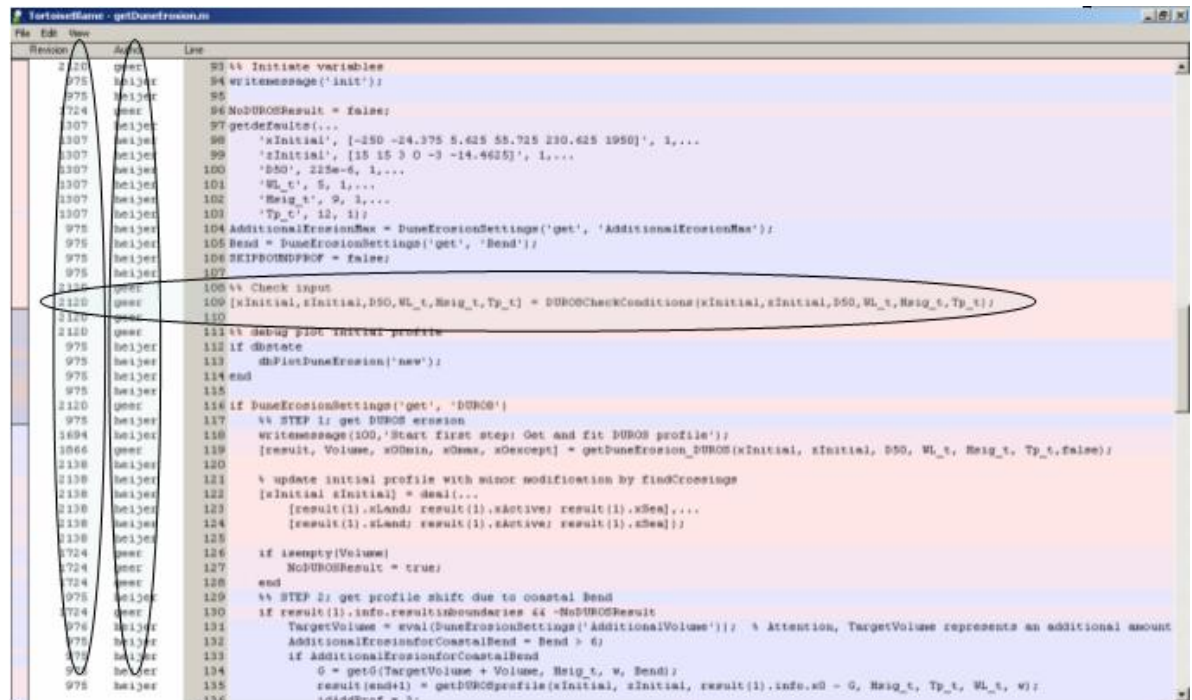


Revision	Actions	Author	Date	Message
304		baart_f	01:55:58, maandag 23 maart 2009	Changes for compatibility with Ualg. Compatibility function for UCLT.
303		boer_g	15:28:45, vrijdag 20 maart 2009	Add more meta-info (lat lon), and test indirect dimension mapping.
302		ormond	13:47:42, vrijdag 20 maart 2009	SuperTrans moved to OpenEarthTools
301		ormond	13:43:11, vrijdag 20 maart 2009	SuperTrans moved to OpenEarthTools
300		ormond	12:35:26, vrijdag 20 maart 2009	findinstruct and findinstruc added.
299		ormond	12:27:48, vrijdag 20 maart 2009	Several gui functions added.
298		ormond	12:27:17, vrijdag 20 maart 2009	updated oetnewfun lgpl block (removed postal address), and added (lgpl) info.
297		boer_g	11:25:07, vrijdag 20 maart 2009	example2tochastVar.m: option added to deactivate (= set to deterministic) one or more variables
296		heijer	10:44:05, vrijdag 20 maart 2009	Fixed loading of table from INP struct from different directory.
295		boer_g	18:05:45, donderdag 19 maart 2009	update of 1dv waterbed module
294		b.c.vanprooijen@tudelft.nl	17:47:46, donderdag 19 maart 2009	Allow for selection of multiple years [Yann Fricourt]
293		fricou	16:47:54, donderdag 19 maart 2009	updated to fix time dimension with correct CF units
292		boer_g	10:07:31, donderdag 19 maart 2009	only add javapath for netcdf when not yet present
291		boer_g	09:56:39, donderdag 19 maart 2009	uploading waterbed
290		b.c.vanprooijen@tudelft.nl	09:03:24, donderdag 19 maart 2009	Added *urls to relevant netCDF tools.
287		boer_g	16:39:42, woensdag 18 maart 2009	In imageplot made frame required 1st arg, set clipping to off by default, and use SetProperty.
286		boer_g	13:53:16, woensdag 18 maart 2009	added new function
285		boer_g	11:47:47, woensdag 18 maart 2009	Reshape locations vector to [x,y] matrix for MUDFile
284		boer_g	13:08:27, dinsdag 17 maart 2009	designpoint: added
283		heijer	17:11:43, maandag 16 maart 2009	some statistical functions added
282		heijer	16:01:51, maandag 16 maart 2009	new functionality to replace string codes (e.g. N/A) in cell with numeric codes (e.g. nan) to obtain fully numeric column files
281		boer_g	18:57:50, vrijdag 13 maart 2009	added conversions between output variable codes (keyword, longname, shortname)
280		boer_g	18:57:01, vrijdag 13 maart 2009	new function to replace string codes (e.g. N/A) in cell with numeric codes (e.g. nan)
279		boer_g	11:41:27, vrijdag 13 maart 2009	added manual input arguments order to enforce specification of column names, and added option to use solely table field from SWAN_INPUT as input.
278		boer_g	16:26:59, donderdag 12 maart 2009	speedup in first call of readTransectDataNetcdf (from 180s to 1.5s) repeating questions are slowed from 0.25s to 0.28s
277		baart_f	10:40:28, donderdag 12 maart 2009	added xbeach url
276		boer_g	17:34:40, woensdag 11 maart 2009	input changed to propertyname-propertyvalue pairs
275		heijer	13:41:11, woensdag 11 maart 2009	
274		rcalvyn	15:26:03, dinsdag 10 maart 2009	New version including improved grid box selection
273		baart_f	15:00:03, maandag 9 maart 2009	testing with test server
272		baart_f	12:25:06, maandag 9 maart 2009	worked on update grid. some problems with multiple transects on the same year

Action	Path	Copy from path	Revision
Modified	/trunk/matlab/applications/xbeach/grid/XBeach_GridOrientation.m		
Modified	/trunk/matlab/applications/xbeach/grid/XBeach_selectgrid.m		
Added	/trunk/matlab/applications/xbeach/grid/bline.m		
Added	/trunk/matlab/applications/xbeach/grid/bline.mht		
Added	/trunk/matlab/applications/xbeach/grid/select_area.m		
Added	/trunk/matlab/applications/xbeach/grid/select_oblique_rectangle.m		
Added	/trunk/matlab/applications/xbeach/grid/wbmf.m		
Added	/trunk/matlab/applications/xbeach/grid/wbmf2.m		

Figure 7 SVN commit: LOGGING (image © G.J. de Boer)

For every code line is it know who, when and under which revision number is was changed. Colors indicate the age of the code (more blue = older). Every change can be rolled back later on.



```

2120 user      93 %% Initiate variables
975 heljor    94 writemessage('init');
975 heljor    95
974 user      96 NoDUBOCResult = false;
1307 heljor    97 getDefaulter(...
1307 heljor    98   'xInitial', [-250 -24.375 5.625 55.725 230.625 1950]', 1,...
1307 heljor    99   'zInitial', [15 15 3 0 -3 -14.4625]', 1,...
1307 heljor   100   'D50', 225e-6, 1,...
1307 heljor   101   'Wl_t', 5, 1,...
1307 heljor   102   'Hsig_t', 9, 1,...
1307 heljor   103   'Tp_t', 12, 1);
975 heljor   104 AdditionalErosionMax = DuneErosionSettings('get', 'AdditionalErosionMax');
975 heljor   105 Bend = DuneErosionSettings('get', 'Bend');
975 heljor   106 SKIPBOUNDFFCF = false;
975 heljor   107
1125 user     108 %% Check input
1125 user     109 [xInitial, zInitial, D50, Wl_t, Hsig_t, Tp_t] = DUBOCCheckConditions(xInitial, zInitial, D50, Wl_t, Hsig_t, Tp_t);
1125 user     110
1120 user     111 %% debug plot initial profile
975 heljor   112 if @State
975 heljor   113   dbFindDuneErosion('new');
975 heljor   114 end
975 heljor   115
1120 user     116 if DuneErosionSettings('get', 'DUBOC')
975 heljor   117   %% STEP 1: get DUBOC erosion
1894 heljor   118   writemessage(100, 'Start first step: Get and fit DUBOC profile');
1866 user     119   [result, Volume, x0min, x0max, x0except] = getDuneErosion_DUBOC(xInitial, zInitial, D50, Wl_t, Hsig_t, Tp_t, false);
2138 heljor   120
2138 heljor   121   % update initial profile with minor modification by findCrossings
2138 heljor   122   [xInitial zInitial] = deal(...
2138 heljor   123     [result(1).xLands; result(1).xActive; result(1).xSea], ...
2138 heljor   124     [result(1).xLands; result(1).xActive; result(1).xSea]);
1724 user     125
1724 user     126   if isempty(Volume)
1724 user     127     NoDUBOCResult = true;
1724 user     128   end
975 heljor   129   %% STEP 2: get profile shift due to coastal bend
1724 user     130   if result(1).info.resultsInboundaries && ~NoDUBOCResult
975 heljor   131     TargetVolume = eval(DuneErosionSettings('AdditionalVolume')); % Attention, TargetVolume represents an additional amount
975 heljor   132     AdditionalErosionForCoastalBend = Bend > 0;
975 heljor   133     if AdditionalErosionForCoastalBend
135 user     134       G = getG(TargetVolume + Volume, Hsig_t, W, Bend);
975 heljor   135       result(end+1) = getDUBOCgetFile(xInitial, zInitial, result(1).info.x0 - G, Hsig_t, Tp_t, Wl_t, W);
136 user     136       result(end) = 3;

```

Figure 8 SVN commit: *BLAMING* (image © G.J. de Boer)

Resolving possible conflicts

When updating your source code or when committing to the repository, SVN will try to merge your local copy with its own version in the repository. Sometimes, when files have been edited by two different people in the same parts, SVN can not perform the merge. It will then yield conflicts. These must be resolved by hand. When finished resolving them, mark the related files as resolved via right click and select: *TortoiseSVN -> Resolved*.

Settings

When modifying a large number of files it can become difficult to keep an overview of which files you have edited. SVN has kept track of your modifications. Using *Icon overlays* you can easily detect which files have been modified, added, deleted or are in a conflicting state. The Icon overlays can be switched on as follows. In any directory, e.g. where you have your source code:

Right click and select *TortoiseSVN -> Settings*

In the Settings menu, select *Icon overlays* and make sure the checkbox *Show overlays and context menu only in explorer* is NOT checked (if you wish to see the overlays also in Total Commander).

Using *Exclude paths* and *Include paths* you can specify respectively which drives or directories SVN should not check or check explicitly. So if you do not want SVN to check your C: drive then write *C:* in the *Exclude paths*. If you however do want SVN to check the directory *C:\source*, then include *C:\source* in the *Include paths*.

up to date



modified

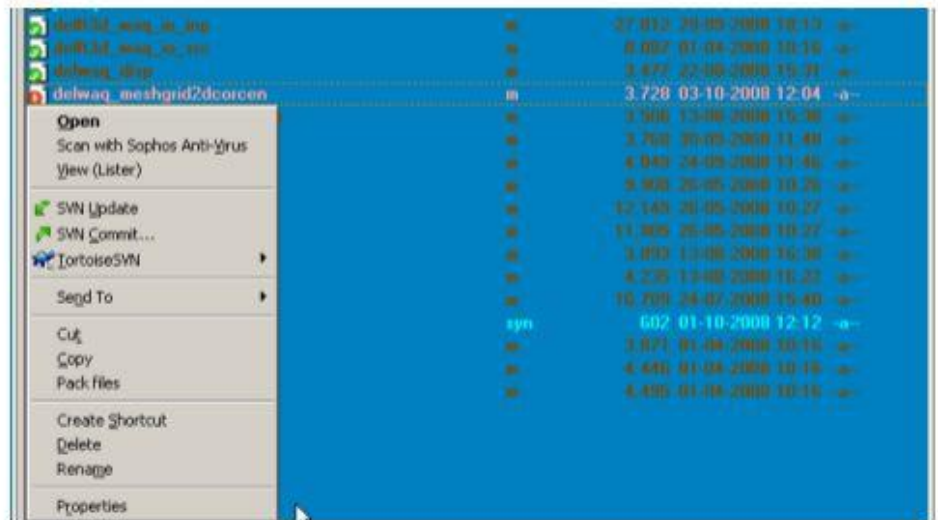


Figure 9 SVN example of icon overlays (image © G.J. de Boer)

TIPS

1. Maak checkout van alleen die subdirectories waar je iets mee doet (dus voor de meesten van ons niet op rootniveau in de directory-boom).
2. Het is belangrijk aanpassingen in checkout middels SVN Rename, SVN Add and SVN Delete uit te voeren.
3. Voer voor een commit altijd een update uit, zodat je aan kan vinken welke aanpassingen je wil committen naar de repository.
4. It is important to commit your changes as often as possible (only the relevant stuff).