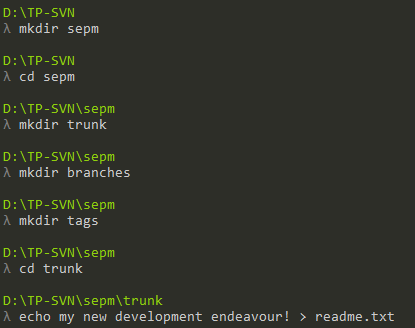
**Software Engineering and Project Management**

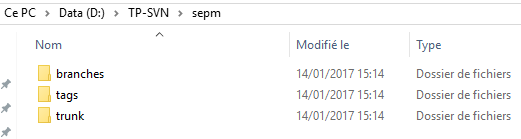
**SVN TP**

**I. New repository, adding, moving, deleting, resolving conflicts, tagging**

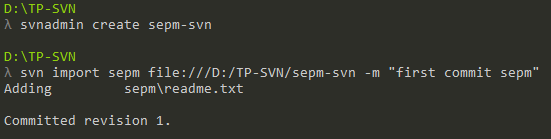
*1. Create a new directory project named sepm. In that project directory write a new readme.txt file containing some introduction information about your project, its name, developer names, date, company name etc. The file should start with the line “sepm – my new development endeavour”. This is the initial project directory.*

Let’s create the project using SVN’s conventions with three folders: trunk, branches and tags. They will be explained later.



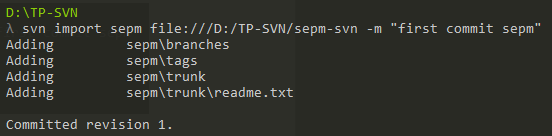


*2. Create a new local svn repository named sepm-svn.*

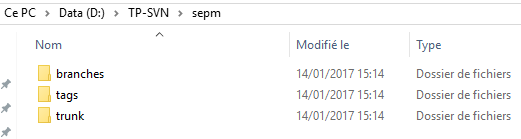


It creates the “sepm-svn” folder containing the files shown in the third question.

*3. Import your sepm project in the sepm-svn repository (trunk). Notice the difference between the initial project directory and the svn repository. Understand the role of the three folders (trunk, branches and tags).*

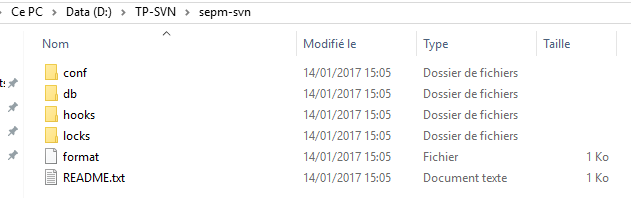


Initial project directory:

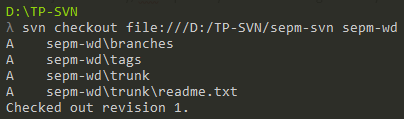


* **Trunk**: main development area. This is where the current source code lives.
* **Branches**: copy of code coming from a certain point in the trunk that is used for applying major changes to the code while preserving the original code in the trunk
* **Tags**: a point in time on the trunk or a branch that one wishes to preserve

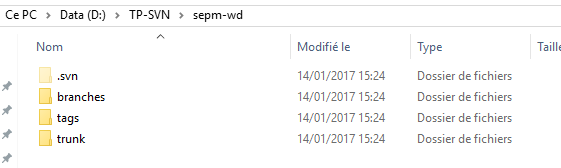
New local SVN repository:

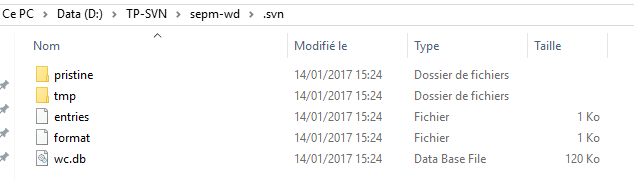


*4. Checkout a working copy of the newly created project in a new directory named sepm-wd. Notice the difference between initial project directory, svn repository and the working directory.*

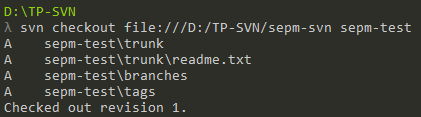


Both “sepm-svn” and “sepm” remain unchanged, however the new “sepm-wd” is like the original “sepm” with a new “.svn” hidden folder at the root:

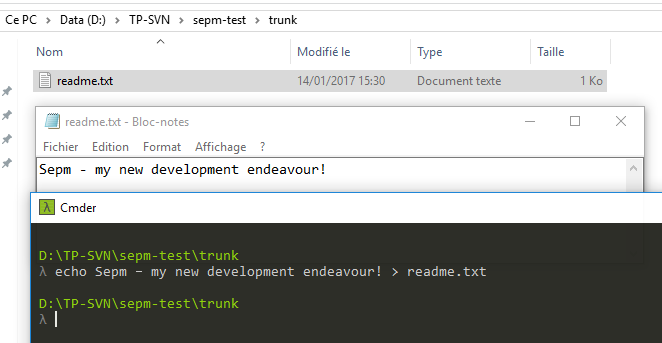




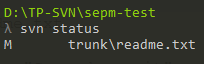
*5. Checkout a second working copy of the project in a different working directory, called sepm-test. Now you can work in two distinct working directories, simulating a two developers team using the same repository.*



*6. In the sepm-test working directory, modify the readme.txt file so that the first line looks like “****S****epm – my new development endeavour****!****” (differences are underlined).*

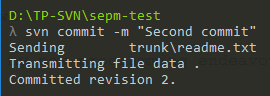


*7. Check the status of your working directory. What do you get?*

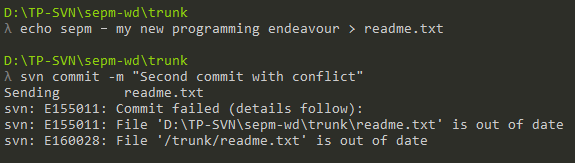


The file readme.txt is flagged as “Modified”.

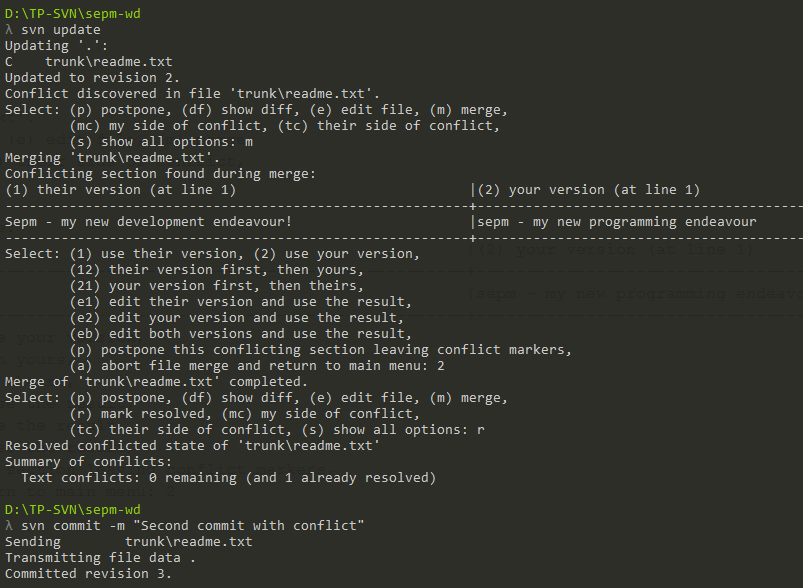
*8. Commit the change. Don’t forget to write an informative message!*



*9. In the first working directory (sepm-wd), change the first line to “sepm – my new programming endeavour”. Try to commit the change. Resolve the conflict.*



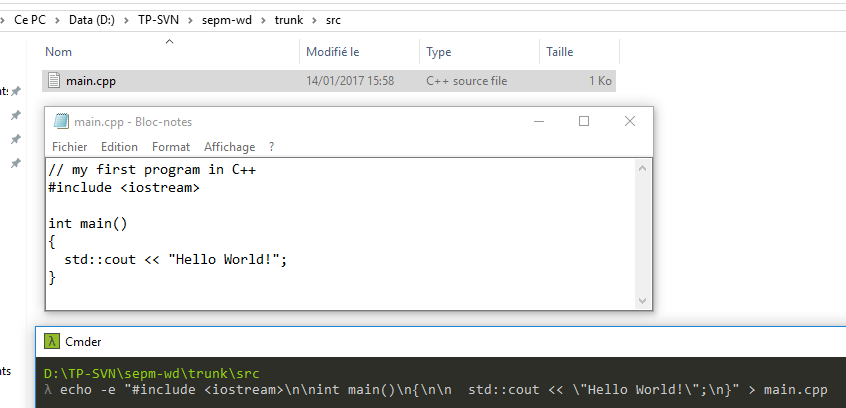
To solve the conflict, we will do a “svn update”:



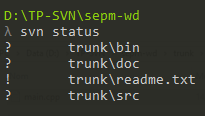
We are asked to choose a version to commit. Then, we can commit normally.

*10. In your current working directory (sepm-wd), create three new folders named src, bin and doc. Move the readme.txt file in the doc directory and create a new main.cpp file inside src (the main function should print a silly message in the console).*





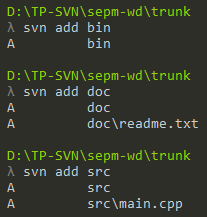
*11. Check the svn status. What happens?*



The folders bin, doc and src are flagged as “?”, which is untracked. The readme has a “!” because it has been deleted and is no longer tracked. The content of the new folders is not yet detected.

*12. Do what’s necessary to add/remove the new directories and files and commit successfully.*

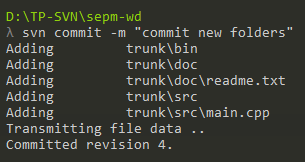
First, we must add the new folders:





The content of the new folders is now detected.

We can now commit:

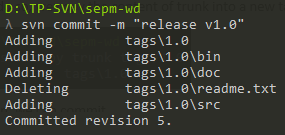


*13. Release version 1.0 of your project*

To do so, we copy the content of trunk into a new tag in the tags folder, 1.0.



Then, we can commit.



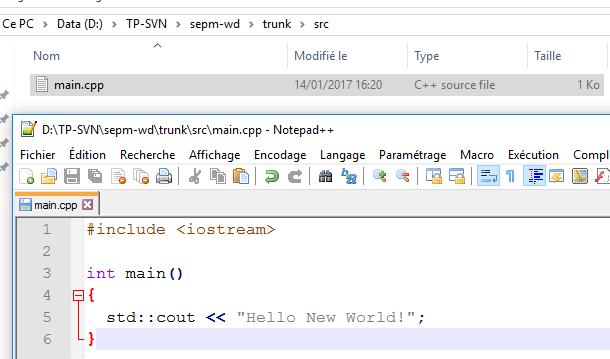
**II. Tagging, branching, merging**

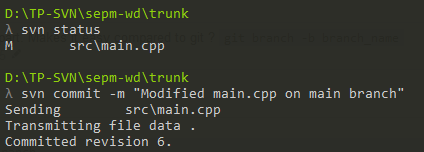
1. *Create an experimental branch named expe.*



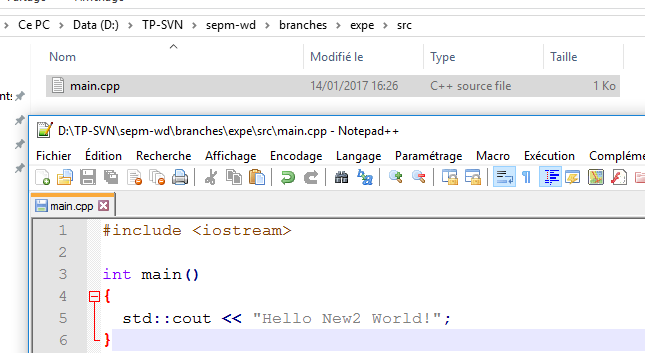
We do not commit it yet.

1. *On the main branch, update the main.cpp file to change the content of the first displayed silly message. Commit.*



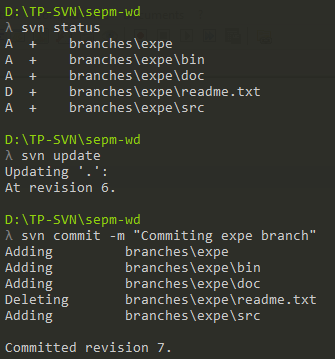


1. *On the experimental branch, update the main.cpp by adding a line of code to print the second silly message.*



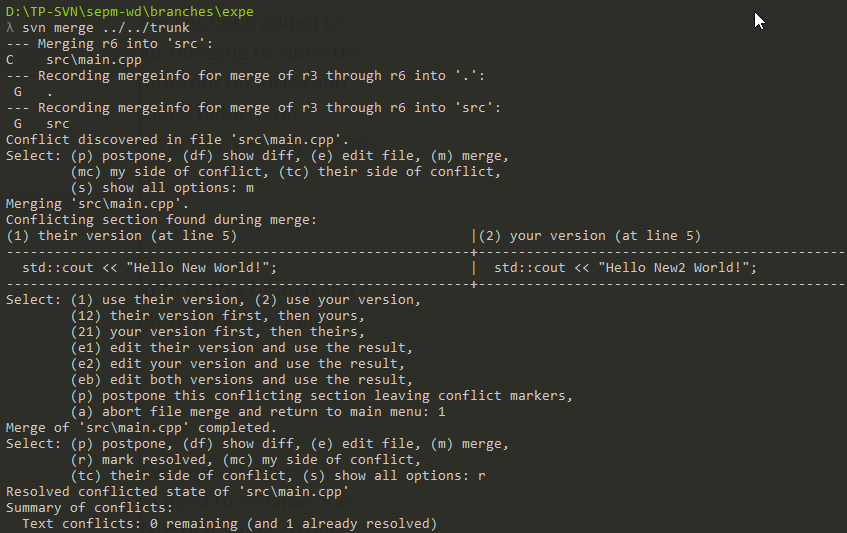
1. *Update the repository.*

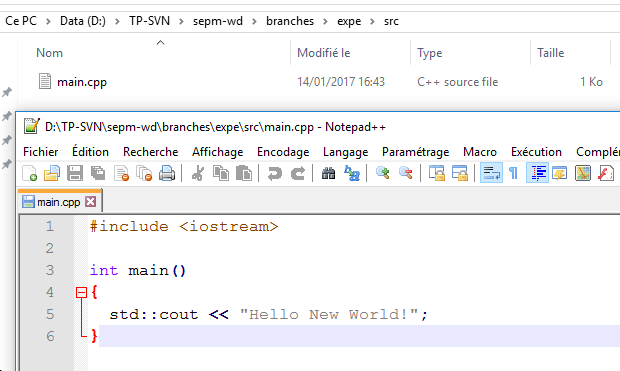
Now we can commit.



1. *You have heard that some important modifications have been made to the main file in the trunk branch that you should integrate in your experimental branch. Do what’s necessary to merge the main branch into the experimental branch.*

First, let’s cd into the branch directory. From there, request a merge from the trunk. Solve the conflict and we are done.





**References:**

<http://stackoverflow.com/questions/16142/what-do-branch-tag-and-trunk-mean-in-subversion-repositories>

<https://www.urbaninsight.com/2012/08/02/svn-merge-recent-change-trunk-production-branch>