What is a Version Control System?

Learn what it means and the basic terminology used on this subject.

January 03, 2008
Ref: DOC-01-Resources

TDC software, Copenhagen, Denmark
www.tdcsoftware.com - info@tdcsoftware.com

INSIDE THIS ISSUE

"Your repository is more than just an archive of the current version of your code. Actually, it is an archive of every version of your code. Your repository contains history."

Basic concepts

Source control is an important practice for any software development team. The most basic element in software development is our source code. A source control tool offers a system for managing this source code.

There are many source control tools, and they are all different. However, regardless of which tool you use, it is likely that your source control tool provides some or all of the following basic features:

- It provides a place to store your source code.
- It provides a historical record of what you have done over time.
- It can provide a way for developers to work on separate tasks in parallel, merging their efforts later.
- It can provide a way for developers to work together without getting in each others' way.

Basics and terminology used in a normal Version Control System

In this section we want to explain some basic concepts about the VCS and the regular words used in this subject.

Let's start by defining two important terms: **repository** and **working folder**.

A VCS tool usually provides a place to store your source code. We call this place a **repository**. The repository exists on a server machine and is shared by everyone on your team.

Each individual developer does work in a **working folder**, which is located on a desktop machine and accessed using a client.

On **TDC** the repository of your source code and the important files and documents related with your projects will be located in a MS-SQL database on your server.
The first thing to do when you start using any VCS is to define and then create the structure of your projects (in a hierarchical structure maybe).

The workflow of a developer is an infinite loop which looks something like this:

- Copy the contents of the repository into a working folder.
- Make changes to the code in the working folder.
- Update the repository to incorporate those changes.
- Repeat.

With a VCS tool, working on a multi-person team is much simpler. Each developer has a working folder which is a private workspace. So the users can make changes to they working folder without adversely affecting the rest of the team.

Not all VCS tools use the exact terms we use on **TDC** and in this document. Many systems use the word "directory" instead of "folder". Some VCS tools use the word "database" instead of "repository". On **TDC** we use the terms **“working folder”** and **“database”**. We prefer to call the repository as “database” because in fact this tool uses MS-SQL database to keep all the information so using this word is much better.

The database exists on a server machine which is far away from the desktop machine containing the working folder where the developer does his work. The VCS tool provides the ability to communicate between the client and the server over TCP/IP, whether the network is a local Ethernet or an Internet connection to another continent.

Because of this separation between working folder and database, the most frequently used features of a VCS tool are the ones which help us move things back and forth between them. Let's define some terms:

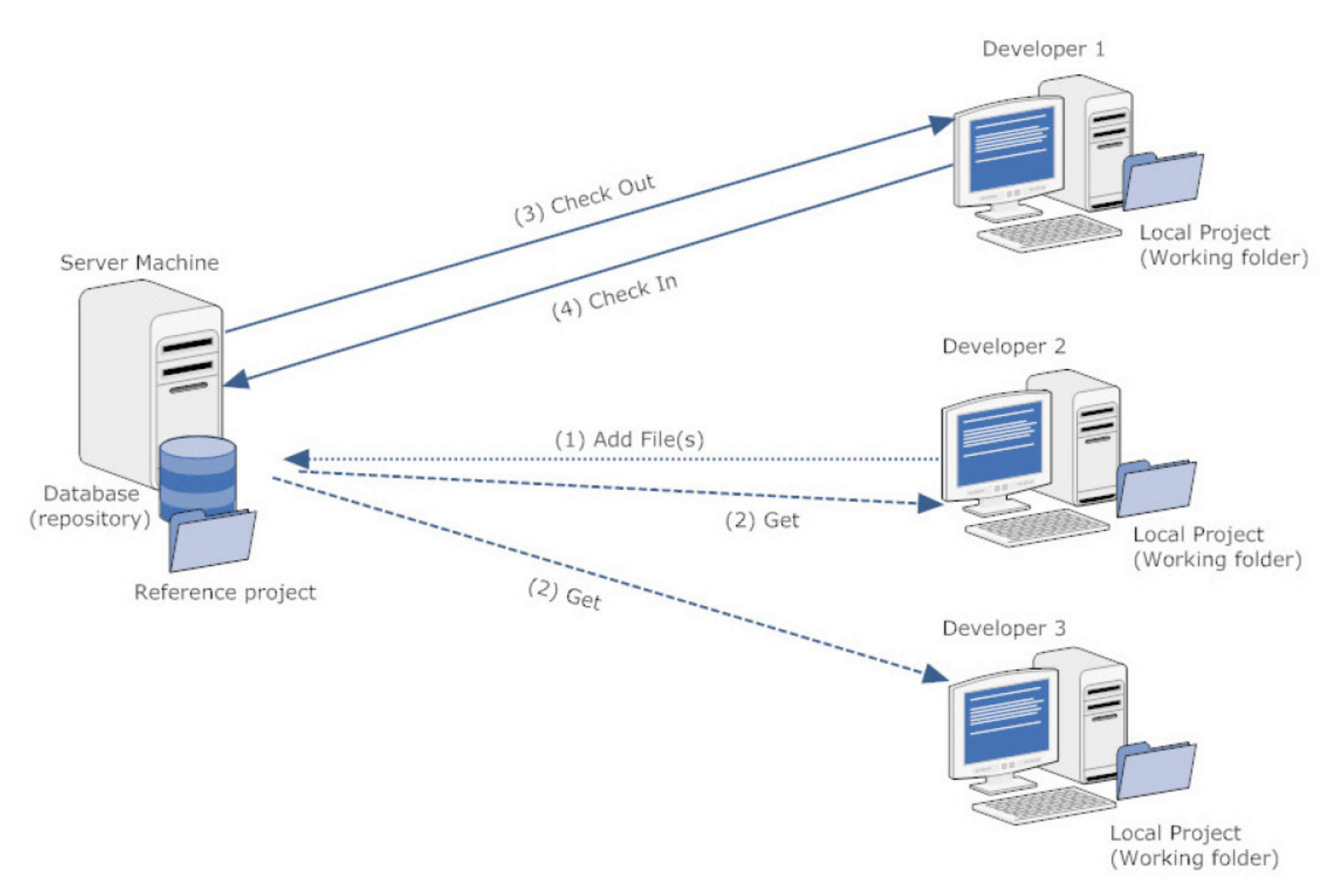
(1) Add: a repository database starts out completely empty, so we need to "Add" things to it. Using the "Add Files" command in **TDC** you can specify files or folders on your desktop machine which will be added to the database an in the selected project.

(2) Get: when we copy things from the repository to the working folder, we call that operation "Get". Note that this operation is usually used when retrieving files that we do not intend to edit. The files in the working folder will be read-only.

(3) Check-out: when we want to retrieve files for the purpose of modifying them, we call that operation "Check-out". Those files will be marked writable in our working folder. The VCS server will keep a record of our intent.

(4) Check-in: when we send changes back to the repository database, we call that operation "Check-in". Our working files will be marked back to read-only and the VCS server will update the repository database to contain new versions of the changed files.

In the next graphic you will be able to see in a visual way all the concepts explained before:



Graphic with the main concepts used in a VCS.

Your repository is more than just an archive of the current version of your code. Actually, it is an archive of every version of your code. Your repository contains history. It contains every version of every file that has ever been checked in to the repository. The ability to go back in time can be extremely useful for a software project. Suppose we need the ability to retrieve a copy of our source code exactly as it looked on June 28th, 2005. With a VCS tool this kind of thing are very easy to do.

An even more common case is the situation where a piece of code looks strange and nobody can figure out why. In this kind of situations it's handy to be able to look back at the history and understand when, who and why a certain change happened.

In the next resource document we will go thru other explanations more in deep about some important functions included in **TDC** tool.

Thanks for your time.

See you next time !

making tools for developers

TDC software
Copenhagen, Denmark

Web site:
www.tdcsoftware.com

E-mail:
info@tdcsoftware.com

Skype user:
tdcsoftware

Victoria Caballero
Mauricio Nicastro