Subversion in the Workflow

by Boyan Kostadinov

Using source control isn't just learning the software; it's acquiring work habits that may feel foreign to you or to your company. Making changes in how you work can seem threatening, but making the right changes can bring a wonderful boost of productivity to a company. In this part of our Subversion trilogy, Boyan Kostadinov tackles the question of how to incorporate Subversion into your company’s workflow.

It is very likely that you have heard of Subversion, commonly referred to as SVN. You might have been told how wonderful it is, how you need version control, and how Subversion will save you time and make you more organized and efficient. But you have not tried it. Maybe you have downloaded it, but it sits there, unused. Why? Maybe Subversion seems too complex, the idea of switching over scares you, or you have your own system of file organization and it is too hard to change. On top of that, there are all those strange terms: branches, trunks, tags. What do you do?

In this article, I am going to address all of those issues. We will look at the advantages of using Subversion, examine the terminology that seems confusing, and talk about using the tool in a real-life environment. This article is a companion to the “Getting Started with Subversion” article in this collection, so I will assume that you know what Subversion is and have already installed it as described in that article.

The Advantages of Subversion

Using a version control system will not write your project, make you dinner, or groom your cat. However, there are a few very good reasons for using a version control system:

- *Improve Productivity and Speed Up Your Development Life Cycle.* Once you start using a version control system, you can easily keep track of changes, Revert to previous versions and create an alternate code base. You will find it easier to create patches for previous and/or deployed versions.

- *A Distributed Development Environment.* If you work on the same project from two different locations (at the office and at home, for example), a common place for your source code will make working on your project much easier.

- *Clear Version History.* No more wondering who made a change, why, or when. SVN supports and even encourages log messages with each committed change.
CHAPTER 40 ■ SUBVERSION IN THE WORKFLOW

Terminology
Once you start working with SVN, you will need to be familiar with some of the terminology used in version control software. Here is a short description of the terms used:

- **branch** – a copy of the main development code that you made in order to make some change or to work on a new version of your application.
- **Commit** – the action of saving the latest changes in your project to the repository.
- **Checkout** – the action of getting an initial copy of your project from the repository.
- **locking** – the process of allowing only one user at a time access to a file inside the repository.
- **repository** – the location where SVN stores all information related to your project. The repository usually contains a trunk, one or more branches, and one or more tags.
- **revision** – a number by which SVN identifies a specific set of changes committed to the repository.
- **tag** – a snapshot of a certain trunk, most commonly created when you want to create a major or minor version of your application.
- **trunk** – the location where the main development for your project is stored.
- **Update** – the action of retrieving the latest changes to your project from the repository.
- **working copy** – the local development copy of your code that you have checked out from SVN.

Using Subversion
Before version control systems came along, it was somewhat difficult to work in more than one environment. If you wanted to work on the same code at the office and at home, you had to make a copy of the whole code base, take it with you, work on it at home, and remember to bring back the change. Furthermore, new functionality was developed by creating a copy of the code base, writing the new code, and eventually replacing the old code. Keeping track of what changed from one release to the next was nearly impossible.

SVN is an intermediate step between development and releasing your code to a production environment. The cycle goes from your development environment to the repository, possibly to a testing environment, and finally to production.

Subversion in Your Workflow This article assumes that you have a working copy of SVN already installed on your machine and that the main SVN directory is called Projects. There are various tools to interact with SVN, but the most common one is TortoiseSVN. For the sake of simplicity, this article also assumes that you have TortoiseSVN installed and working.

While the next few sections on creating a repository and importing code have been covered in greater detail in the “Getting Started with Subversion” article in this collection, I felt that they were