INTEGRATING BIG DATA... SMARTER

syncsort

Integrating DMExpress with Version Control

Table of Contents

1 What is Version Control

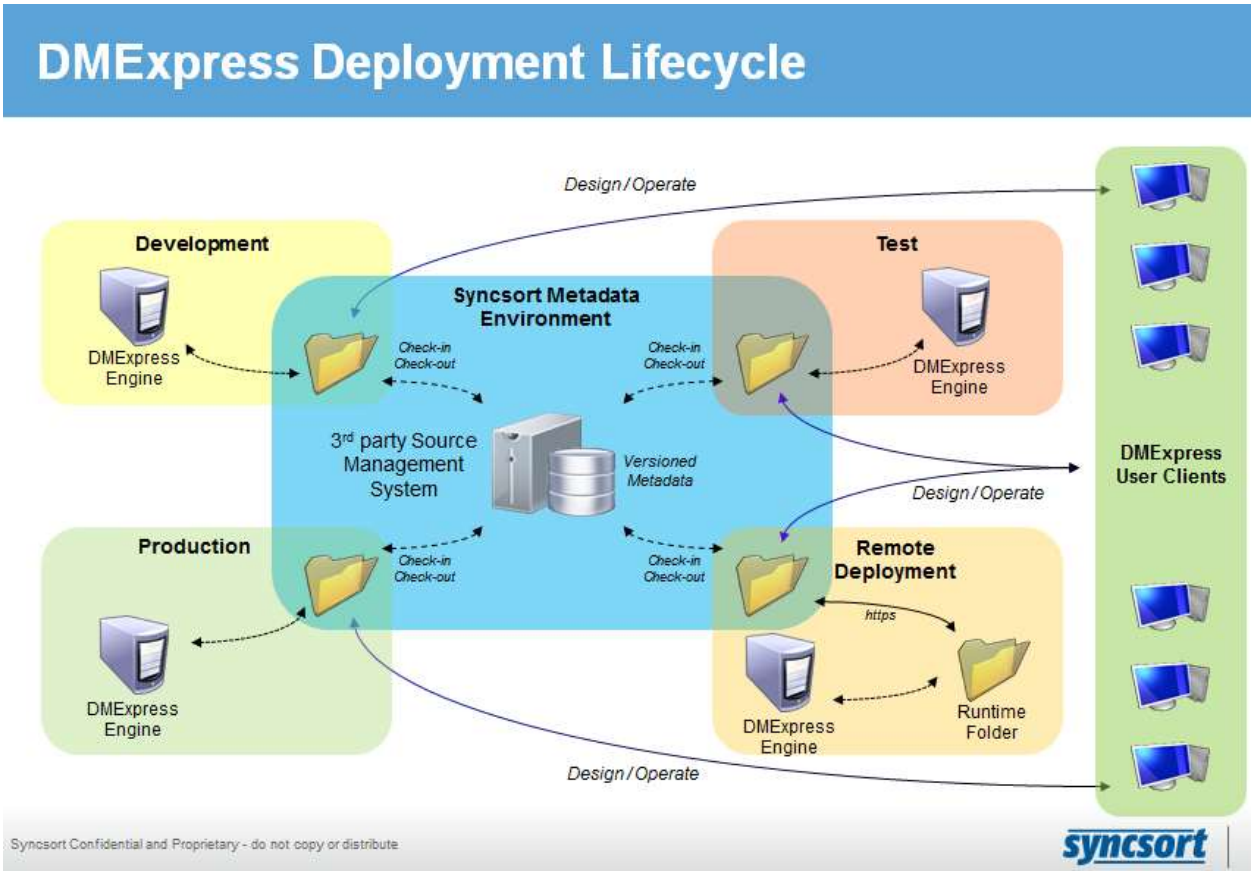
Version control is a way to manage changes to files and is most commonly used in software development, where a team of people may change the same files. Changes are usually identified by a number or letter code, termed the "revision number", "revision level", or, simply, "revision".

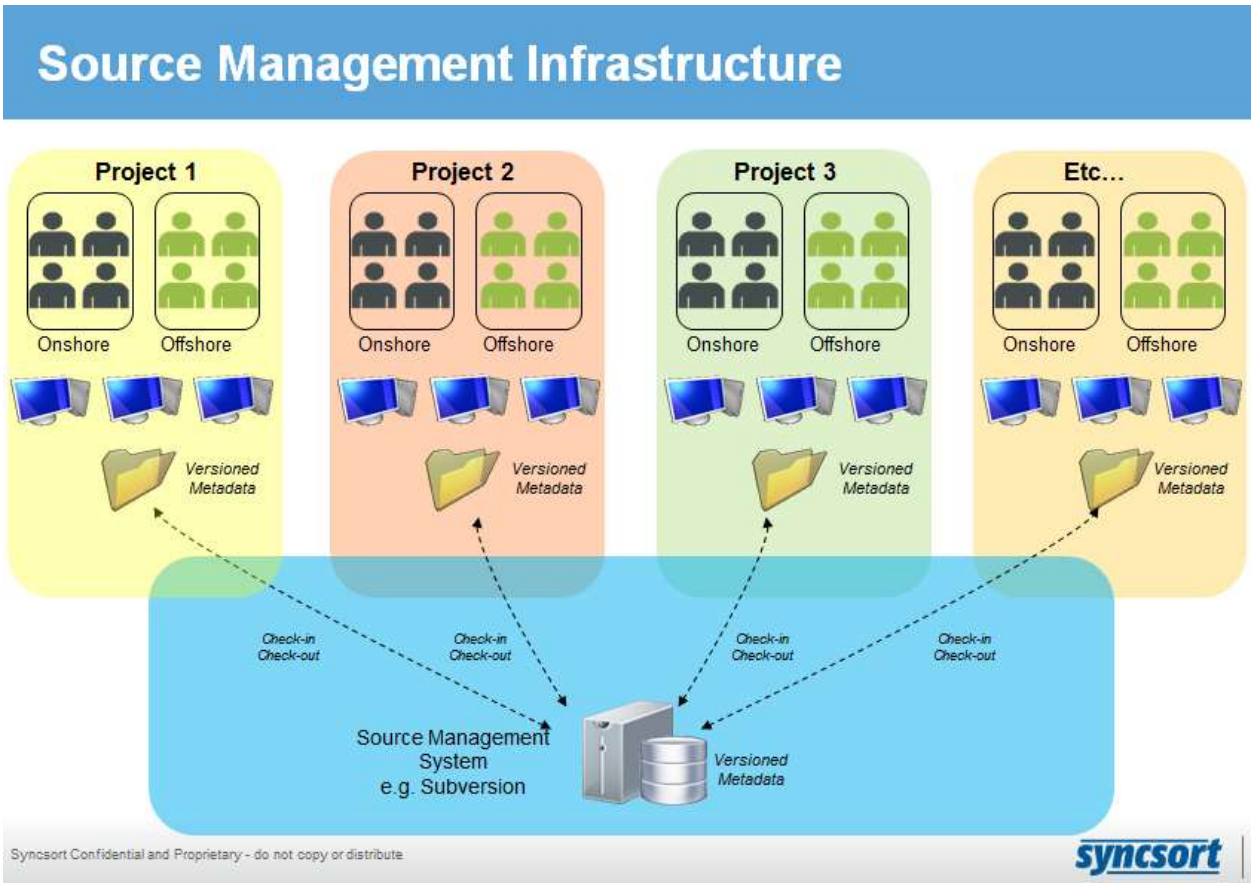
For example, an initial set of files is "revision 1". When the first change is made, the resulting set is "revision 2", and so on. Each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored, and, with some types of files, merged.

2 Why Use Version Control in an ETL Environment?

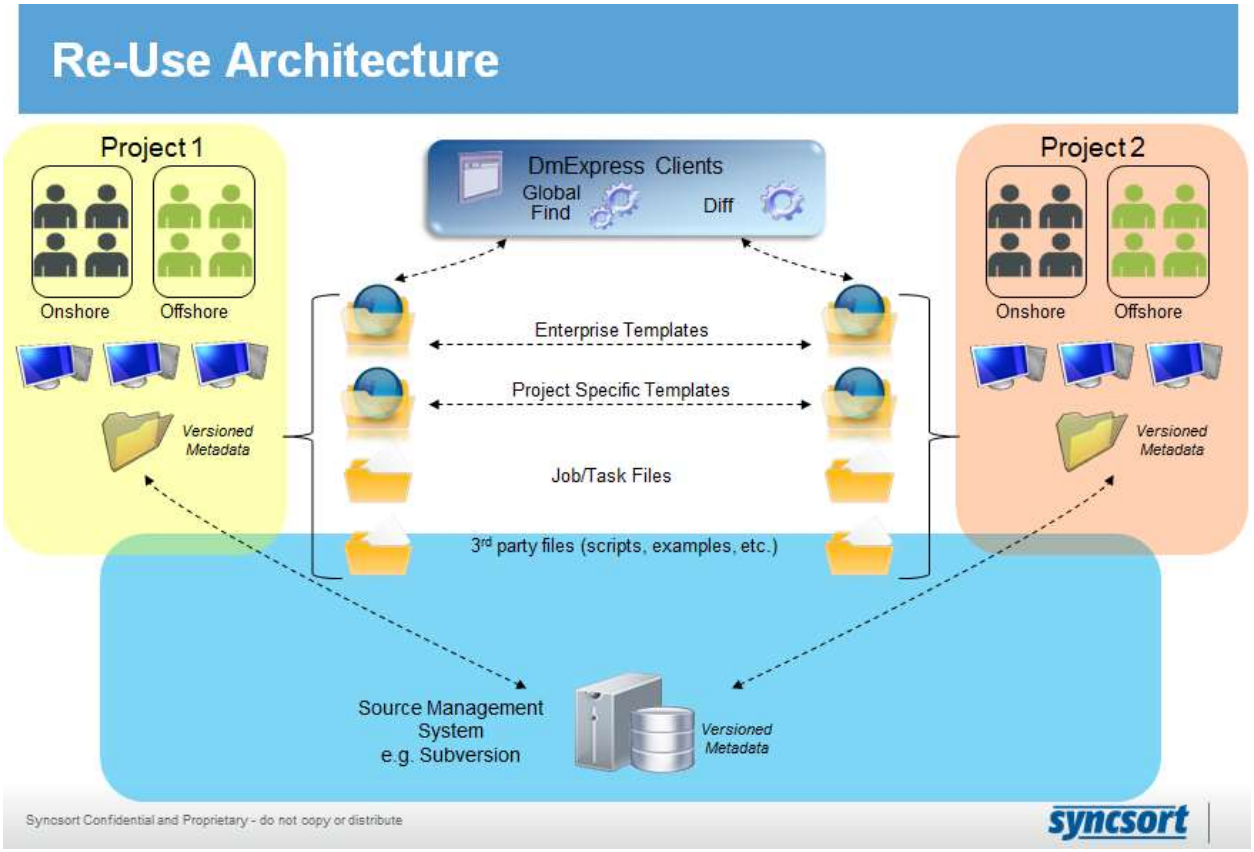
There are two key types of versioning needed in an ETL environment:

- Design-time versioning – required during the process of creating jobs/tasks, typically with multiple users and sometimes even across different countries/time zones.
- Deployment-time versioning – typically carried out by a specific operations team, and used when moving "packages" of jobs, tasks, scripts, and third party code between different environments as part of a "release".




Source Management Infrastructure
Project 1
Project 2
Project 3
Etc...
Onshore
Offshore
Versioned Metadata
Check-in
Check-out
Source Management System
e.g. Subversion
Syncsort Confidential and Proprietary - do not copy or distribute
syncsort

One important benefit of a version control system is that it enables “sharing” of common components between teams. This reduces the amount of new development required, and allows for version-specific components so you can track different releases using various iterations of these common components.



3 Advantages of Integrating DMExpress with Version Control

DMExpress works well with most file-based version control systems because it does not require a repository server (such as a database server) for its metadata, and all the DMExpress components (tasks, jobs, and external metadata files) are stored as files at the O/S level.

Using an access-controlled version control system in conjunction with the DMExpress development tools is recommended as a best practice due to the numerous benefits outlined in the following sections.

3.1 Concurrent Development

DMExpress development with version control allows for multiple developers to work on the same project concurrently. Instead of changing the name of the file when saving – which quickly leads to confusion and loss of accountability – the version control system can report conflicts and require the developer to resolve them by manually merging in his changes. This practice allows many developers to work on the project without locking files.

Each developer checks out a working directory of a project into their own "home" directory structure. The version control system tracks which developers make changes, and can provide reporting on activity by developer.

3.2 Access Control

Developer access control can be implemented using features of the version control system. For example, Subversion allows access control by repository path, SSH authentication, SSL authentication, and Windows Authentication.

3.3 Reuse of Development Objects

Reuse can be handled by defining directories in the source code repository which are to be included in multiple projects. In Subversion, for example, this inclusion is done using subversion external definitions.

3.4 Logging

Use of DMExpress with a version control system adds the advantage of detailed logs of every change made throughout the life of the project. Logging details includes user name, time and date of the commit, a change note entered by the developer, and the files affected.

3.5 View Developer Changes

Changes between revisions of non-binary file can be displayed using the version control system's `diff` (file difference) capability. For DMExpress job/task files, which are binary, you would need to use the Syncsort `dmxdiff` utility.

3.6 Tag Releases for Deployment

A project may be "tagged" with a release number prior to deployment, and any past revision or tagged version can be checked out. This capability assists deployment and technical support operations, since it provides a complete snapshot of any past state of the project.

3.7 Revert Recent Changes or Extract Previous Versions

Finally, changes may be reverted if it is found that a recent commit creates a "breaking change", i.e. a change that causes some unwanted behavior in the project.

4 A Practical Example – Using TortoiseSVN with DMExpress

TortoiseSVN is an open source Apache Subversion (SVN) version control system client. Implemented as a Windows shell extension, it integrates tightly with Windows Explorer, and is free, intuitive, and very easy to use. For more information on TortoiseSVN, see http://tortoisesvn.net/.

See DMExpress and DMX-h Implementation Best Practices for best practices when integrating DMExpress with a source control system such as Subversion.

4.1 Install TortoiseSVN on Windows

1. Download and install TortoiseSVN from: http://tortoisesvn.net/downloads.html.
2. Restart `explorer.exe` (or reboot) after installing; TortoiseSVN is a Windows Explorer plug-in.

4.2 Create a Repository

1. In Windows Explorer, create a folder to contain the repository, for example:

   ```
   C:\svn\svnrepo
   ```

2. Right-click on the created folder and select `TortoiseSVN->Create repository here`.

4.3 Integrate dmxdiff Utility with Subversion

DMExpress includes a utility, dmxdiff, that can be integrated with Subversion to compare revisions of DMExpress job and task files, generating a report in either text or HTML format. It can be found in the `<DMExpressInstallDir>\Programs` folder, where `<DMExpressInstallDir>` is typically `C:\Program Files\DMExpress`.

See "Integrating dmxdiff with Subversion" in the DMExpress Help for further details.

4.4 Import a Project's Initial Source into the Repository

1. In Windows Explorer, browse to the project directory containing your DMExpress project files.
2. Right-click and choose `TortoiseSVN->Import`
3. Add a new directory in the root level of the repository, for example:

   ```
   C:/svn/svnrepo/proj1-changedatacapture
   ```

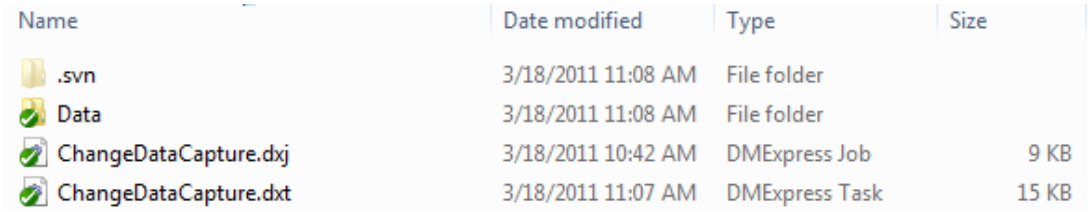
4. Add a note and click OK.
5. Once imported into Subversion, you can delete the original project directory.

Note that DMExpress job and task files should be managed as binary files in the source control system.

4.5 Check Out a Project

Check out the project directory from the repository by right-clicking on the folder, e.g. `c:/svn/svnrepo/proj1-changedatacapture`, and selecting `SVN CheckOut…`.

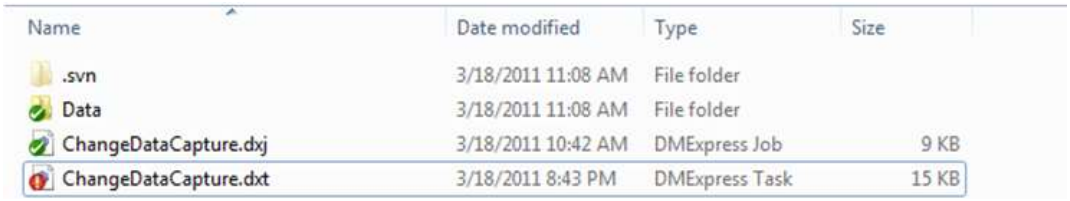
Note that the project folder now displays in Windows Explorer with a green checkmark, indicating that it is up-to-date.



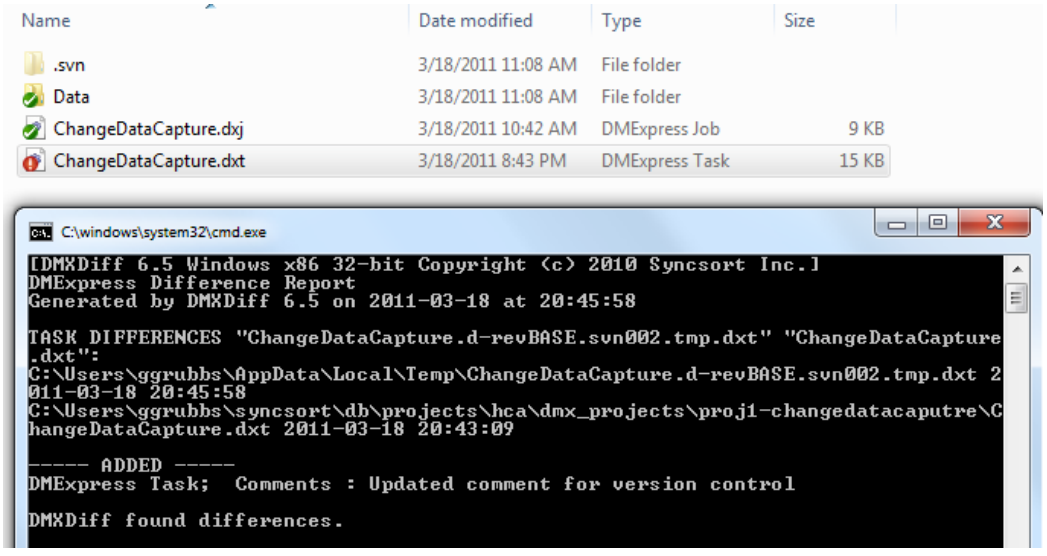
4.6 Edit Project Files

Edit one of the project files. For example, add a comment to a task.

Save the edited task, and note that the task file now displays with a red exclamation mark, indicating that the file has changes not yet checked in:



To view the difference between the changed file and the version last checked in, right click on the file and choose `TortoiseSVN->Diff`



4.7 Check in Changes

To check in changes, right click on the project folder in Explorer and select `SVN Commit…`.

Continue developing in the project as normal, periodically checking in changes. Any new files and directories will have to be added to the project.

4.8 View Log and Revert to Previous Versions

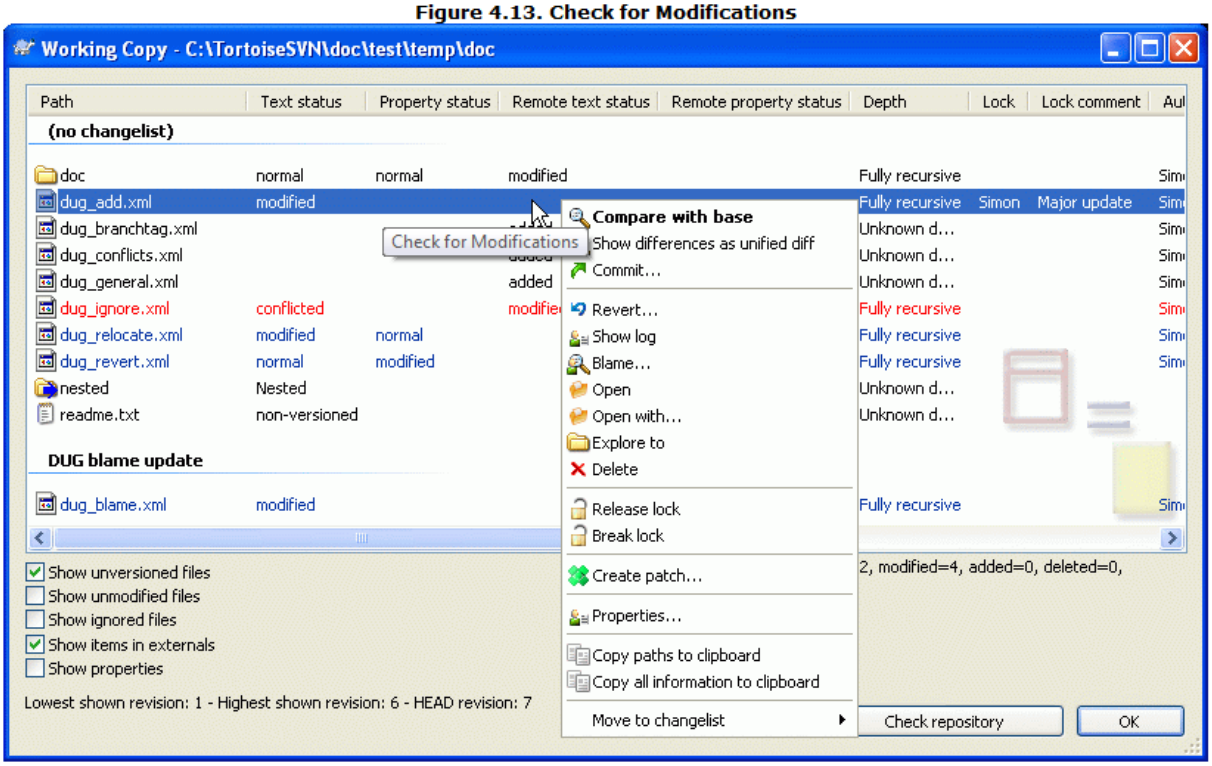
To view the log of changes, right click on the project folder and select `TortoiseSVN->Show log`.

To revert to an earlier version, right click on an item in the log and choose the appropriate action, such as `revert to version`.

4.9 Show Which Files are Out of Date with the Repository

To view the status of your files, right click on the project folder and select `TortoiseSVN->Check for modifications`.

Figure 4.13. Check for Modifications



4.10 Show How Developers are Contributing

To see how the developers are contributing to the project, right click on the project folder and select `TortoiseSVN->Show Log->Statistics`.

Figure 4.21. Commits-by-Author Pie Chart

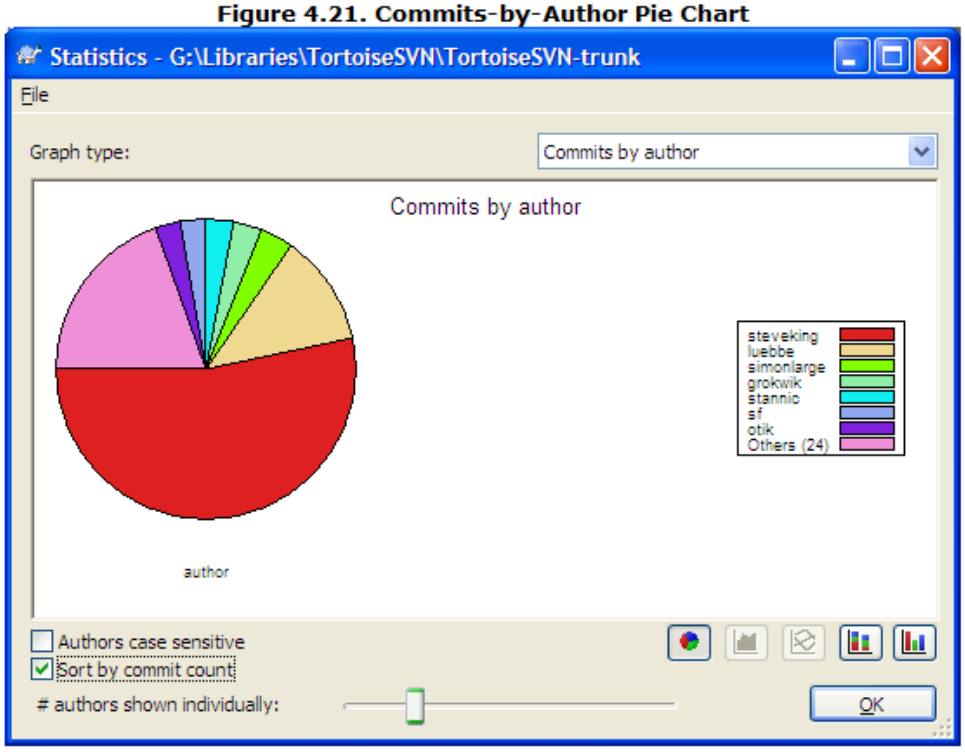
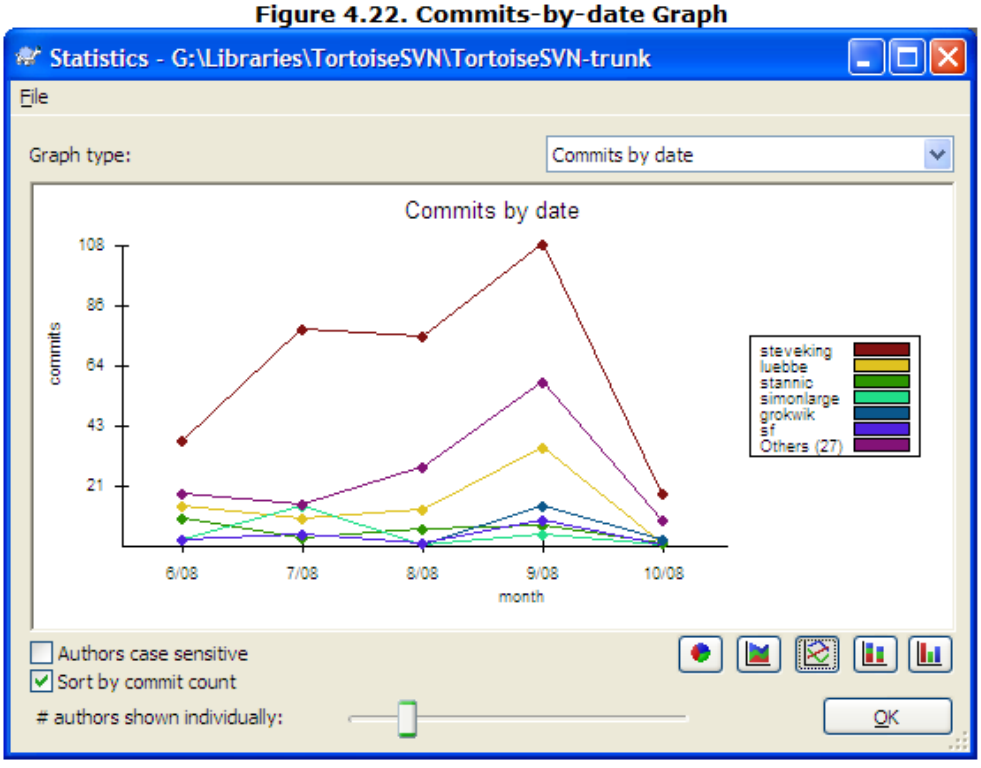


Figure 4.22. Commits-by-date Graph



About Syncsort

Syncsort provides data-intensive organizations across the big data continuum with a smarter way to collect and process the ever expanding data avalanche. With thousands of deployments across all major platforms, including mainframe, Syncsort helps customers around the world overcome the architectural limits of today's data integration and Hadoop environments, empowering their organizations to drive better business outcomes, in less time – with fewer resources and lower total cost of ownership. For more information, please visit www.syncsort.com.

Syncsort Inc. ***50 Tice Boulevard, Suite 250, Woodcliff Lake, NJ 07677*** ***201.930.8200***