824
doi:10.1017/S1431927613006119
Microsc. Microanal. 19 (Suppl 2), 2013

The Use of Revision Control to Implement Best Practices for Experimental Microanalysis

Nicholas W. M. Ritchie

National Institute of Standards and Technology, Materials Measurement Science Division, Gaithersburg, MD 20899, USA

Revision control is one of the most powerful tools in a programmer's arsenal. However, the utility of revision control goes beyond the programming realm and can be an important addition to every scientist's tool kit. Revision control can be thought of as a mechanism for tracking the changes in the documents representing a project. Most often, the project consists of the files necessary to fully describe a software program or a software library - the source code, the build files and other resources. However, most revision control systems are not limited to working with software projects. Any evolving type of documentary or data projects is suitable for revision control.

Revision control tools come in both commercial and open source flavors. Revision control tools, accessible from the command line or wrapped with a user friendly graphical user interface, are available for every major desktop operating system. Some of the more capable and more modern systems are Git[1], Mercurial[2] and Subversion[3]. All of these systems offer the ability to maintain a historical record of the files in a project and any modifications to these files. This includes the ability to recreate the state of the project at any point in time in the project's history. Revision control systems also facilitate collaboration by allowing multiple people/places to create replicas of the project and to integrate local changes back into the project. Modern revision control systems are typically built around either distributed or client-server models (like Subversion). In the client-server model, the server retains the master copy of the project. In a distributed model (like Git or Mercurial), there is no explicit master copy and each copy of the project retains a complete record of the project history. Changes are synchronized between users by exchanging patches, or descriptors of version-to-version revisions. By exchanging only patches, multiple copies of projects can be kept synchronized with a minimal amount of data exchange.

The utility of revision control for software projects is obvious. Revision control facilitates collaboration but even for the single developer it allows tracking the evolution of a project. Tracking changes reduces the risk of making experimental changes to source code since changes can always be undone if they prove unwise. It is even possible to try multiple different approaches and merge only the most successful one into the final project.

However, revision control is useful outside the programming domain and, in particular, in the experimental domain. As experimentalists, we have a responsibility for maintaining an accurate record of our experimental environment. Historically, this was done with laboratory notebooks. Today, the laboratory notebooks can only capture a portion of the data or an incomplete snapshot of the data. Revision control provides a mechanism to keep a historical record of instrument parameters, initialization files and experiment scripts and other pieces of information which may be important when asked to show how a measurement might be reproduced. Revision controlled projects can also be used to maintain data integrity such as required by 21 CFR Part 11, a pharmaceutical industry regulation to ensure the trustworthiness of data records.

One microanalytical application of revision control is to curate a historical archive of standard spectra. The highest accuracy quantification of energy dispersive x-ray spectra involves comparing a spectrum from an unknown material with *standard spectra* from materials of known composition. Rather than recollecting standard spectra for each measurement, it is often useful to keep a carefully curated set of common standard spectra. As time goes by these spectra must be updated as the efficiency and performance of the x-ray detector change. Simply overwriting older spectra with new ones discards critical historical data. In addition to being bad practice, there may be legal ramifications to discarding experimental data. With revision control, it is possible to retain a bit-perfect record of the standards archive at any point in time. With revision control, it is possible to keep only one standard spectrum archive but to restore the contents to any previous point in time. It is also possible for people to share the same basic standard library but to 'fork' the library for their own purposes and then to 'merge' their fork back into the main archive at a later date.

Introducing revision control into your experimental protocols will take a little investment in time to learn the concepts and to gain confidence in how your files are being managed. However, the initial investment will quickly pay off as you discover how easy revision control makes administering historical archives of experimental procedures and data.

[1] Git is available from http://git-scm.com
[2] Mercurial is available from http://mercurial.selenic.com/
[3] Subversion is available from https://subversion.apache.org/

https://doi.org/10.1017/S1431927613006119 Published online by Cambridge University Press