The Use of Open Source Tools in PHENIX Code Development

M. Velkovsky[1], *D. Morrison*[2]

[1] University at Stony Brook, Stony Brook, NY 11794, USA
[2] Brookhaven National Laboratory, Upton, NY 11973, USA

Abstract

At present, the PHENIX Experiment at RHIC has approximately fifty collaborators actively developing code for both online and offline uses. If anything, this number is expected to increase in the near future as experimental data becomes available. PHENIX relies heavily on several open-source tools to help manage the code development problems presents by this large and varied group. These tools include ones long familiar to the HE/NP community, such as the use of CVS to for version control, as well as some that are not as widely used. These newer tools include some adopted from the Mozilla project, such as Bonsai which provides a query interface to CVS, Tinderbox for managing software builds, and Bugzilla for tracking problem reports. PHENIX also uses CVSWEB, LXR and DOC++ to provide source code navigation and documentation. The HtDig package is used to search the PHENIX web. This talk will discuss our experiences, both good and bad, with adapting these tools to PHENIX use.

Keywords: open source, GNU, cvs, mozilla, PHENIX, RHIC

1 The Need for Open Source Tools

The PHENIX collaboration consists of about 400 participants of which no less than 75-100 are involved in the development and the testing of the software. Some of these people are abroad, others travel frequently and theothennely3n people(are)]TJ0 -1.243 TD[assignsedtothespepordof the softwareothenca for softwar,s the with varoursfomrs and oporating ystemrs and

PHENIXsoftwareto

2 The standard UNIX tools

We have decided to restrict the portability of our software to several flavors of UNIX, like Linux for Intel, Solaris and Linux for Alpha, but other UNIX versions can be added. The key tools for the software development are

- CVS, the Concurrent Versions System is a ”Source Control” or ”Revision Control” tool designed to keep track of source changes made by groups of developers working on the same files, allowing them to stay in sync with each other as each individual chooses.
- The GNU C/C++ compilers gcc (egcs) and debugger gdb are some of the most sophisticated and tested ones. One considerable problem that emerges from the usage of commercial software like the object oriented database Objectivity is that its binaries work only with libraries produced by particular versions (usually outdated) of the C++ compilers.
- Arla - an open source AFS client/server, which although initially posed some problems, was more flexible than the commercial AFS server by Transarch, which was not working with the latest Linux kernels.
- The main piece in this collection is the GNU compile/build system, which includes Libtool/Automake/Autoconf It tremendously simplifies the task of building libraries and executables from a huge collection of sources, allows more consistent behavior of the build process, make it easy to automatize by using Perl scripts and conceptually, by introducing multiple layers of abstraction, it forces the developers to think at a higher level.

3 The build process

At the center of the PHENIX software build process is a Perl script, which checks out the software from the CVS repository in a source directory, uses the GNU Libtool/Automake/Autoconf tools to produce Makefiles in a build directory, and finally, by invoking the GNU make and ginstall, it builds the libraries and installs them, together with some header files in one of the several AFS areas: old, pro and new. From there the software is world wide available via AFS. The new area is rebuild nightly by a crontab job, while the pro area corresponds to named releases of the PHENIX code, done at weekly or monthly intervals. At present, the rebuild script contains only a rudimentary runtime (regression) testing, namely it checks whether the rebuilt libraries can load into Root - the object oriented data analysis framework that is adopted in PHENIX.

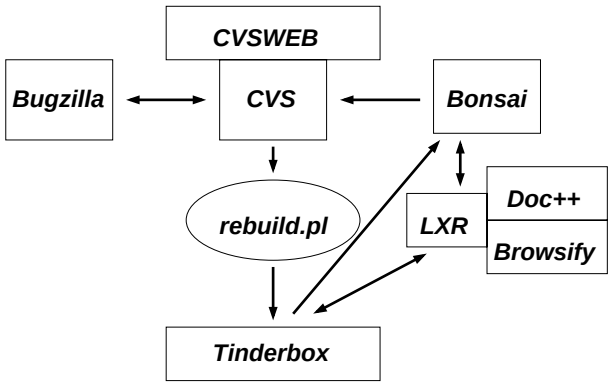

Figure 1: Interaction of the rebuild script with the other development tools

4 Some new development tools

We have installed some neat new development tools that help us a lot. They include:

- **HtDig**, a local search engine, allows searches of documents that might be behind the PHENIX firewall, because it goes along the file system.
- **CVSWEB** is a CVS repository browser, that uses the web.

Particularly nice are the tools developed by Mozilla - the open source browser group that spearheads the development of Netscape.

- **Bonsai** – tree (package) control. Queries of a database for all CVS activities.
- **Tinderbox** – monitors the rebuild process. Shows the status, the person to blame for a failure and the rebuild logfile. Links the source code via the massively cross-referenced **LXR** - the source code code browser.
- **Bugzilla** – the tool for listing, classifying, discussing and assigning responsibility for bugs. It may be rather annoying to the person responsible for fixing the bug, because it automatically sends him reminders by e-mail.

Together with **LXR** we use two other source code browsers and documentation tools: **Doc++** and **Browsify**. The latter is written by Matthias Messer from our tiny offline group.

5 Outlook

The adaptation of some of these tools required some efforts, because they were not particularly well adapted for installation in environments different than their native one (Mozilla). However one big step remains to be completed - the inclusion of some automatic regression testing framework like **greg** or**dejagnu** into the rebuild script. This will allow much stricter and systematic checks of the PHENIX software quality on a regular (nightly) basis.