Improving Software Installation Techniques at the National Center for Computational Sciences at Oak Ridge National Laboratory

Anthony DiGirolamo
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, Tennessee
Email: anthonyd@ornl.gov

Robert D. French
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, Tennessee
Email: frenchrd@ornl.gov

I. Introduction

Smithy is a software compilation and installation tool that borrows ideas heavily from SWTools[1] and the homebrew[2] package management system for Mac OS X. Smithy is designed to manage many software builds within an HPC Linux environment using modulefiles to load software into a user's shell. SWTools has set very good conventions for software installations at the NCCS. Smithy's goal is to make following the SWtools conventions easier and less error prone. Smithy improves upon SWTools by providing a simpler command line interface, modulefile generation and management, and installations via formulas written in Ruby.

II. Improving on SWTools

SWTools relied on build scripts to perform installations. The problem we encountered with build scripts is that they are duplicated for every software installation. This can make installing new software tedious since one has to go back and look at existing build scripts and copy relevant steps to a new build script. Frequently build scripts simply run the default ”configure, make, install” pattern and omit custom changes to Makefiles or source code. This knowledge is easily lost and can make future installs difficult to replicate and maintain.

III. Installation Formulas

The goal of installation formulas in Smithy is to consolidate all knowledge required to build a software package on any system into a single file. This removes the problem of duplicate build scripts and missed steps. Formulas can include: defining dependencies, loading or swapping modules, setting environment variables, applying patches, creating or changing makefiles, running the compilation, running tests, and defining a modulefile. It is recommended that formulas be stored in a version control system such as git or subversion. The NCCS uses github.com to store a repository with all its formulas. This allows for easy sharing and collaboration with the HPC community. New formulas or changing existing ones can be sent as a pull request between HPC centers.

IV. Documentation

Smithy provides detailed documentation in manpages and on its website. Additionally any Smithy subcommand can be prefixed with 'help' on the command line to show all relevant options and arguments. There is also command line tab completion for bash and zsh. Documentation for installed software can be now written in markdown or html. Under the SWTools scheme, installing the same package on multiple systems would require separate documentation for each installation. Smithy adds an option to use only one documentation file per software package across all systems.

V. Conclusion

Formulas are a way to programatically install software in such a way that is self documenting. We have been able to use them to great effect on systems at the NCCS. Our hope is that Smithy software installation formulas for different operating systems and software packages will be shared between HPC centers. This could greatly reduce the time required to install software.

References

[1] N. Jones, M. R. Fahey, ”Design, Implementation, and Experiences of Third-Party Software Administration at the ORNL NCCS,” Proceedings of the 50th Cray User Group (CUG08), Helsinki, May 2008.

[2] Homebrew. brew.sh. Retrieved December, 2013 from http://brew.sh/