A Hybrid Approach to Scientific Software Package Management on a High-Performance Computing Cluster

Qiyang Hu, Shao-Ching Huang

Institute for Digital Research and Education, University of California, Los Angeles (UCLA)

Abstract

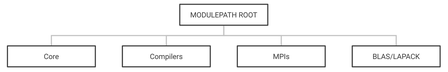
We present a practical approach for managing the scientific packages in the HPC cluster environment of a research university environment that has a diverse user base. The primary goal is to minimize the HPC operational team's burden of installing and maintaining a large number of software packages to support the broadband spectrum of the university's computational research. We propose a hybridizing management method that can harness the power of modern software management frameworks and the flexibility of in-house, or "manual", installation sources, and at the same time present a coherent view to the end users. The Spack [1] framework was used in this work. Our hybrid approach is applicable to using other framework tools. The manipulation of Spack-generated environment module files and the typical workflow are illustrated and discussed based on our use case.

Challenges

- Challenges when managing software manually from individual sources:
  - Combinatorial complexity is exploding: at UCLA > 10,000 combinations

- Challenges when using software management framework:
  - "Seemingly" duplicated copies of installed packages from Spack

  - Non-trivial to customize modulefile structures from Spack



  - People (sysadmins & users) are hesitated to move in
    - Overwhelmed by the complexity controlled in a "Spack-ish" way
    - Huge momentum to keep the existing scheme/work
    - Are cautiously waiting for further updates
    - OK to tolerate sub-optimal runtime performance in university environments

Our Working Goal

- To introduce a top layer to make the framework-based software management system and the manually-installed system **co-exist**
  - For sys admins: workflow scriptable as much as possible
  - For end users: package list presented as clean as possible

Modulefile Structure Customization

Adding a flexible management layer on top of framework requires traversing the module file structure generated by the software management framework. Based on our experience, also motivated by the operation of other large-scale HPC clusters (e.g. TACC Stampede), we first propose the module files to be organized in a simple structure as shown in Figure below, where the MPI level is nested in the compiler level. The versions of compilers and MPIs follow the convention in Spack.

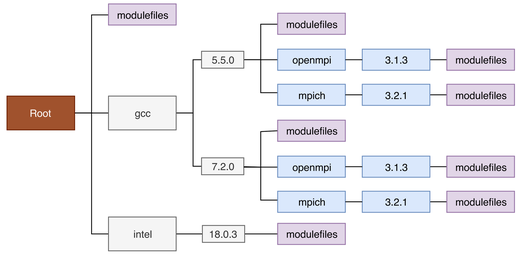


Figure: Newly defined hierarchical structure of module files in Hoffman2

Using JSON For Interfacing with Support

We use JSON to express the information of the packages that our HPC operational team will support, i.e. visible to end users. An example of a JSON file is listed below:

```
{ "bowtie2":  {
        "name_in_spack":  "bowtie2",
        "version":  [
            "2.3.4.1-gcc-7.2.0",
            "2.3.4.1-gcc-5.5.0"
        ],
        "default":  "2.3.4.1"
    },
},
```

where the main name in the json object pair is the application name preferred by system admins. The `name_in_spack` is the name used in Spack. It can be blank if there is no cooresponding one in Spack. The `version` keeps the application version information and the corresponding compilers. The default version that will be shown or selected by the `Lmod` system [3].

Hybridization of Modulefiles

Our approach takes advantage of Spack's capability of generating both flat and hierarchical module files by enabling them in modules.yaml configuration file. Figure below shows our module filtering process which will do a few additional steps of:

- Calling `tcl2lua.tcl` script
- Pre-pending `MODULEPATH`
- Adding `family`
- Generating `.version` file

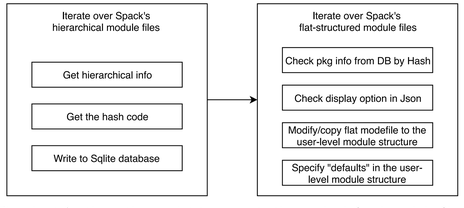


Figure: Flowchart for the python script to generate the newly defined module file structures

Operational Workflow

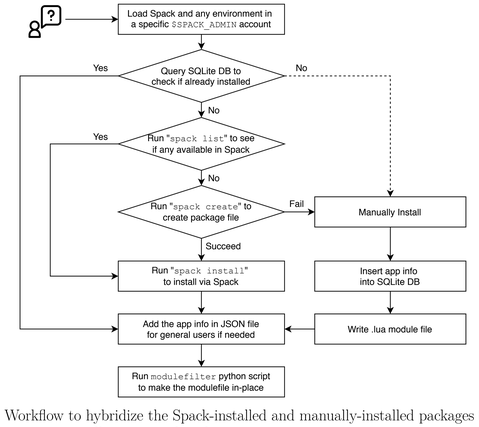


Figure: Workflow to hybridize the Spack-installed and manually-installed packages for an administrator to install a requested package

Summary

In this work we proposed an approach to hybridize the software management framework and the traditional manual package management. The design adds a layer on top of the standard package management framework expecting to take advantage of the powerfule framework while maintaining the management flexibility for manual installations and the customizable HPC user interface. With the definition of a module file structure in the HPC cluster environment, the workflow is able to process, filter and integrate the module files from two independent sources (i.e. framework-based and manual-based) to any target modulefile presentation.

Additional Information

- We will present our work in the "Software Provisioning" session in SC'20 State of Practice Talks on Nov 17, 2020 at 1:30PM:
  - https://sc20.supercomputing.org/presentation/?id=sotp110&sess=sess289
- Please feel free to check out our demo code in the gitlab repo:
  - https://gitlab.idre.ucla.edu/hoffman2/modulefile-processing

References

[1] Todd Gamblin, Matthew LeGendre, Michael R. Collette, Gregory L. Lee, Adam Moody, Bronis R. de Supinski, and Scott Futral. "The spack package manager: bringing order to hpc software chaos," SC '15 Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2015.

[2] Markus Geimer, Kenneth Hoste, Robert McLay. "Modern Scientific Software Management Using EasyBuild and Lmod" HUST '14: Proceedings of the First International Workshop on HPC User Support Tools, page 41-51, November 2014.

[3] R. McLay, K. W. Schulz, W. L. Barth, and T. Minyard. "Best practices for the deployment and management of production hpc clusters," In SC '11 In State of the Practice Reports, page 9:1–9:11, 2011.

[4] https://spack.readthedocs.io/en/latest

Contact Information

- Email: huqy@idre.ucla.edu, schuang@idre.ucla.edu
- Phone: +1-(310)-825-2011, +1-(310)-825-4874

UCLA idre