Software Packaging
Date: September 7, 2022
Presented by: David M. Rogers (Oak Ridge National Laboratory)
(The slides are available under “Materials from the Webinar” in the above link.)

Q. I have a stupid question. What do you mean by packaging? Does it mean everything you need is packaged together to the point of almost plug-and-play? What is annoying about research software sometimes is that people push to codes out there before compiling them–let alone testing them. Sometimes researchers push different versions of software out there and do not tell users which version is the working one. Sometimes the only version of code does not work and the person in charge does not answer questions.

A. I am using “package” to refer to a piece of software or data for which you have documented a process for users to download and use. I cover C++, fortran, and python here - which have conventions, tools, and repositories that make that documentation much simpler.

Q. It appears that exascale researchers are moving toward cmake. What is the main reason?

A. The main reason I see is that there are a lot of HPC users that cmake is working for. It supports CUDA and HIP compile options, documentation is good, and there are many examples. Still, other build tools and package managers are out there, and may be useful. Look for projects with similar requirements to yours.

Q. Is spack comparable to conda?

A. Spack builds its programs from source code, and offers much greater control (via https://spack.readthedocs.io/en/latest/build_settings.html) and build-specs to customize what system libraries will be used and what build options will be enabled. Building programs from source also enables hardware-specific optimizations (e.g. AVX512 or cuda_arch=72).

Q. I'd like to hear David's opinion/suggestions on single-sourcing a Python package's version from the source code manager (e.g., Git)

A. I’d always recommend manually setting your package’s semantic version. That way you can use major, minor, and patch levels to indicate whether the version is “new”, “incremental”, or just fixing bugs. With python, it’s sometimes annoying that both setup.cfg and __init__.py set the package version. You can get around that importlib.metadata (example) - as setup by pyscaffold.

Q. Do containers run slightly slower because of some overhead?

A. Usually no. Containers are lightweight, including only a few libraries and your program. In the case of python programs, containers can be faster because loading dependencies scans fewer files. Exception: If you have installed dependency libraries differently on the base system and your container, performance can vary. Note: on HPC systems with slurm, see --container= (https://slurm.schedmd.com/containers.html)