

HSF Packaging Meeting Notes

Agenda

<https://indico.cern.ch/event/869507/>

Participants: Ben Morgan, Patrick Gartung, Chris Green, Graeme Stewart, Martin Ritter, Serhan Mete, Marco Clemencic, Benedikt Hegner, Geri Ganis, Ivan , Dmitri, Pere, Attila, Richard? (SFT TS), Akanksha?

Introduction (Ben)

- Packaging WG merging with tools - to decide on topics
- Please send suggestions for Best Practice to Ben/Graeme
 - Can we relate this to the HSF project template?
 - Attila noted that project template for ATLAS tools was not that successful
 - To be discussed offline
- Lund registration will open in January

CHEP 2019 Report (Graeme)

- Four talks in Track 5, clearly packaging is still a topic of interest
 - Maybe goal of this group is to make packaging boring (i.e. simple!)
- Caveat emptor: views are own!
- Spack and SpackDev
 - Update from Sofia 2018 appreciated to illustrate complexity of problem
 - Graeme covered active work in FCC, SFT, SuperNEMO
 - Bonus: FAIR experiments moving to Spack
- SpackDev
 - Commitment to Spack in Exascale Computing Project underlined - reinforcing benefits of working with larger community
- Gentoo Prefix
 - From Benda Xu (Guilherme co-author)
 - Good to see tools covered/know by WG
 - Advantages of its deep build to libc/ld (almost container-like)
 - Discussion: can have lots of prefixes, but each only containers only one version at one time. Nix/Spack are more views on a set of variants
 - Graeme: powerful too, but not best fit to all HEP use cases
- Conda
 - From Chris Burr (also looked at in this WG)
 - Graeme: like that its widely used by users, as opposed by librarians/package managers
 - Ergonomics excellent

- Fit could be in delivery of flexible/up to date system to analysis users, without weight of whole production stack
 - Attila/Patrick: How to use conda to package e.g. ATLAS software? Patrick: example of this earlier this year for CMS FWlite
 - Pere: benefit over pip? Can get more than Python packages
- Missing slide!
 - On overview of track from Martin: Lots of interest in Spack

Spack at Supercomputing (Chris)

- Many Spack related events, inc. two round tables, tutorial
 - Had long one-to-one with core developer Peter Scheibel
- Concretizer was main news
 - New implementation based on constraint solver due in March
 - Much faster than current Python implementation
 - Very important for FNAL's use cases, but likely highly useful elsewhere
 - Preview available on a feature branch on GitHub
- Key concretizer improvements
 - Install of packages from older recipe versions (with hash change)
 - Explicit compiler dependence (inc. no compiler)
 - Language standard as a virtual dependency
 - Use of pre-built older versions over building latest with no version spec
 - Better reconciliation of compatible variant specs down dependency tree
 - Better handling of build/link/run/test time dependencies
- Developer discussions
 - Integration of SpackDev into Spack
 - Automatic detection/use of system packages
 - Spack environments in SpackDev
 - Expand concept of Spack extension commands (e.e.g add new subcommands to an existing spack command, extensions for build/module systems without needing upstreaming).
- Other highlights
 - FNAL contributions to Spack core noted in BoF
 - E4S (Extreme-Scale Scientific Software Stack): base stack of spack built/managed scientific software. Integration with Gitlab CI build pipelines to enable quick build of new software and/or new platforms/architectures.
 - Contents will evolve, should evaluate integration of HEP-centric software here, as would lose some fine control on version
 - Ben has experience with GitLab CI
 - Conceptually like LCG releases, but for a different community
 - Attila: ATLAS will be moving to GitLab CI in 2020, so need good integration here
 - Marco: Is Spack is a moving target in development? Chris: under active development, new concretizer should solve many of the issues

(e.g. hash changes). There is a regular release schedule (tags) to help with stability.

- Hope that automatic detection of system packages can be turned off!
Chris: “automatic: refers to option to generate packages.yaml file rather than hand-managing. Would be a spack command, so opt-in.

- Post-SC Spack article in HPC Wire (referenced SpackDev talk from CHEP 2018)

Spack Plans in SPI Project (Ivan)

- Interest across HEP community
- First step: use to build equivalent of LCG_96b
- LCG model:
 - Take as much from underlying OS as possible
 - Pre-built compilers
 - Will SPI provide compilers in a binary form?
 - Not yet decided how to do this, but need noted
 - Patrick: possible to build relocatable binary packages for compilers
 - Build and test entire stack (~460) every day (reusing previous ones if they match)
- Good:
 - Easier (python) syntax, but core code complicated
 - Already implements what are “hacks” in lcgcmake
 - Only missing a few package recipes (ML, Jupyter, GRID), not difficult to add, just need time
 - Attila: can you use the recipes as is? Yes, detailed comparison in slide link
- Not-so-good
 - Only one package built at a time
 - Some differences in recipes/style between lcgcmake/spack
 - Attila: how to handle differences in how we build Geant4 etc? Yes, if we supply/activate variants
 - Chris encourages common recipe, even if different experiments need to use different specific builds
 - Concretizer
 - Sometimes need to explicitly set versions
 - Some issues in resolving virtual packages
- Bad:
 - Hashes are very sensitive to changes
 - Chris: will be largely fixed by the new concretizer!
 - Patrick: some might say it’s a good thing! Comes down to what you treat as a significant change (and can identify that).
- Questions:
 - Views concept vs that in lcgcmake
 - Variants for Instruction sets. Chris: develop has support for microarchitecture builds

- RPATH/RUNPATH/LD_LIBRARY_PATH
 - Use of environment: modules vs views
- Next steps:
 - Dedicated spack fork on HSF GitHub
 - EOS area for sources/binaries
 - CVMFS test build (sw.hsf.org?)
 - Documentation under LCGDocs
 - SPI-JIRA Epic to track evolution and issues/wishes
- Attila: do we have enough FTEs to maintain? Chris: several from LLNL, rest unsure.
- RPMs not supported everywhere, e.g. OS X
- ATLAS looking forward to test the outcomes early next year
- LHCb would like to use the recipes and do the rebuild themselves

AOB

- Patrick: Current work on relocating buildcaches into non-default directory layout:
 - <https://github.com/spack/spack/pull/13797>