HEP Software Foundation

HSF Packaging Group: R&D Ideas

Graeme Stewart and Ben Morgan

2018-06-13

LIM Workshop

- As already pointed out…
  - Great to have such a good survey of users and their needs
  - Much discussion around ‘snags’ and short term improvements
    - Very useful to have in this context
  - Not so much conclusive discussion about the future of packaging
    - That aligns more with the goals of this group
    - But interesting problems were raised and we can try to address them
- Good time to have a discussion about the long term goals of the group’s work and what we hope the outcomes will be

The Story So Far

- We prepared the Use Cases document
    - This has been well received
- We defined the Test Stack
    - Useful, but rather minimal
    - Clearly any packaging tool can build software, so this is a test that doesn’t really tell us much about the tool itself
    - What it can be useful for is to measure the availability of pre-defined build recipes for the software our community needs
- We started Test Drives
    - Again these are quite basic tests in themselves, experiments need a lot more (cf. use cases)
    - They do allow for useful comparisons between tools (look and feel tests, ease of use)
    - We’re happy to continue this as an activity, but not the top priority

R&D Questions

- How much to we rely on what the site has installed? Do we try and become independent of that?

- Tension between taking system packages and rebuilding everything
    - Sweet spot may actually be one of the *extreme* ends, rather than somewhere in the middle
    - That’s what a lot of stacks do today and it can very much seem like the worst of both worlds

Production: Bet on Containers

- HEP pushed more and more to use resources we do not “own”
- Much weaker influence on sites to customise themselves for our workflows
  - E.g., HPC centres
- The more customisation we ask for the more likely problems arise
  - Tension between site requirements for stability, our desire for latest and greatest versions of compilers and packages
    - Very problematic for C++ packages
  - Building a modern stack on an older base OS leads to a lot of work
- Looks like industry decided that this should be solved with containers
  - Ben and I think this is right
  - Experiments also moving this way too, e.g., CMS
  - N.B. This does not mean super-fat containers - many sites will have CVMFS in addition

Full Stack Depth

- Building an entire self-contained stack does have some significant advantages
  - Essentially we're rolling a HEP-OS Linux distribution
  - Ensures consistency
- However, it's a lot of work
  - Noted that for LCGCMake case it would be just unfeasible to do this within a small group
  - Minimum of 350 packages needed for functional development machine in CentOS7, Ubuntu 16
    - lxplus 7 has 2900 RPMs installed
  - Updates do need to be considered
- Only manageable by leveraging the work of a larger project
  - Take your pick of distribution or project: Portage, Nix, etc.
- Exploring this option matches the work of Chris Burr in LHCb
  - We think it's valuable and interesting to study and track the outcome

Modernise The Base

- Alternatively, can we save ourselves a lot of work by modernising the base?
- A lot of rebuilds being done because of an antediluvian base OS
  - SLC6 released 8 years ago
  - CentOS7 released 4 years ago
  - But many of the software versions within are several years older than that
    - Python, gcc, etc.
  - ABI changes, C++ standard changes affect compatibility
- What could we gain from a significant upgrade of the base OS?
  - Particularly if we use the toolchain from the underlying OS
- Advantage could be that we focus efforts on our core tasks - building HEP software
- Worries about longevity of release and support, of course - how much are these alleviated by the containerised deployment?

Developers

- Supporting our developers is paramount
- Containers are a much lighter weight affair than VMs
  - Developers do prefer to work “native”
  - Costs of supporting this?
- Alternative, for Linux, is a “prefix” environment, a la Nix or Portage
  - Gains independence from the underlying linux distribution, except for the kernel
    - This is like an “unboxed container”
- For OS X, Windows?
  - Base toolchains more effort to support in these cases
  - Not generally well supported by experiments
    - Maybe containers* are the most reasonable solution here
- Providing a solution here is explored by FNAL via SpackDev
  - Again, valuable and interesting to study and track the outcome

*Yes, there’s a light VM too!

An R&D Inventory

- Recognise the strengths of this group
    - Common point where we share ideas and experience
- Would be great to have a small amount of documentation on each of the ongoing projects
    - Very light “who, what, where” description, the desired outcomes and a few links
    - A markdown file in the packaging area or a section of the HSF webpage would do fine