Software Packaging

Elizabeth Sexton-Kennedy and Benedikt Hegner
for the HSF packaging working group

A Reminder: Motivations

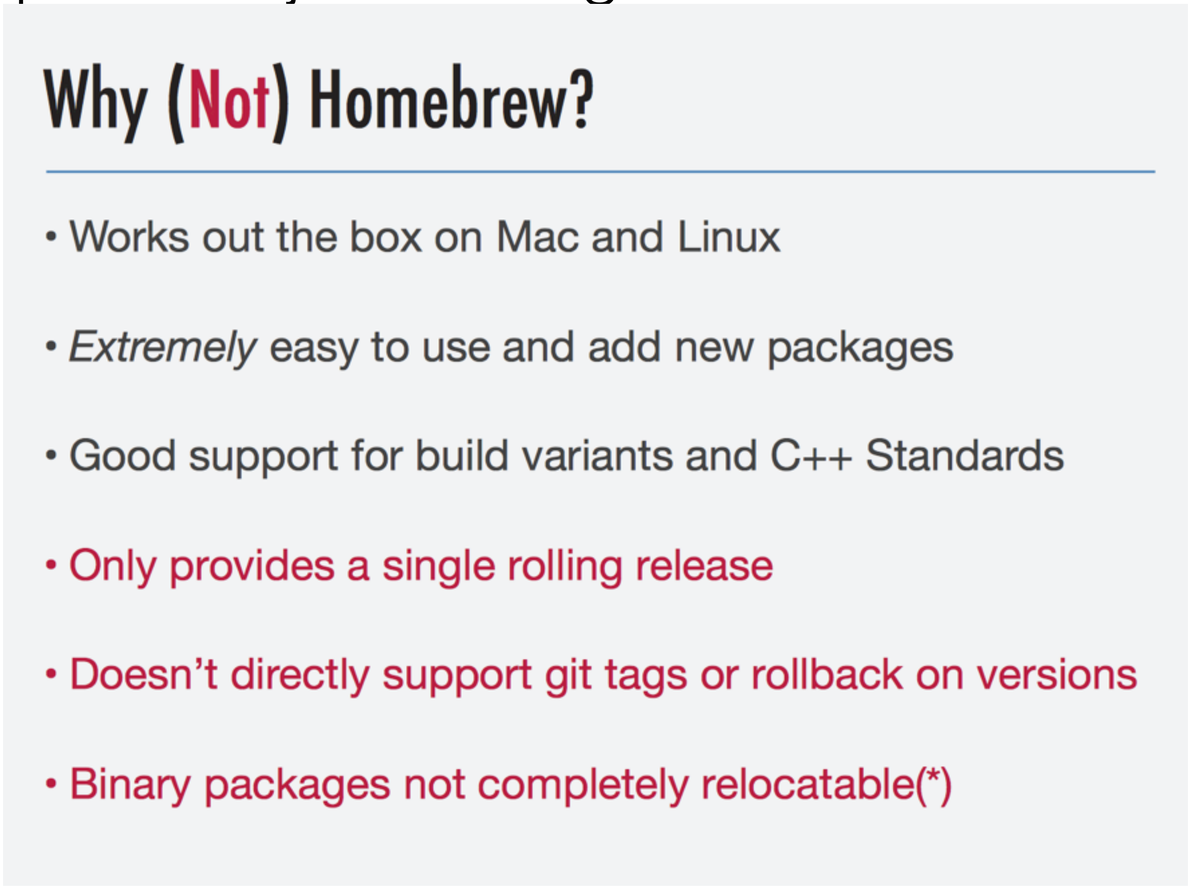
- Build tools are foundational. Packaging is an opportunity for improvement.
- Common enough problem, that duplication is not necessary.
  - example porting OSS to new platforms
- Most of our software stacks depend on O(100) external packages, maintenance of that stack requires lots of manpower that could be shared/ consolidated.
- Reducing effort in this area means it can be applied elsewhere.

Working Group Goals

- At CHEP we said:
  - Collect technical input from all communities on what they have and want in the future (use case style requirements gathering)
  - Distill it down to common language and features necessary to define some sort of packaging / building protocol
- The first of these bullets is DONE. The presentations where a mixture of two things and served two purposes:
  - gathering requirements
  - a taxonomy of existing solutions
- Will use a google doc to build consensus about what we have learned.

Packaging Discussion #1

- Homebrew as an Example - Benjamin Morgan



Packaging Discussion #2

- DUNE packaging ideas - Brett Viren

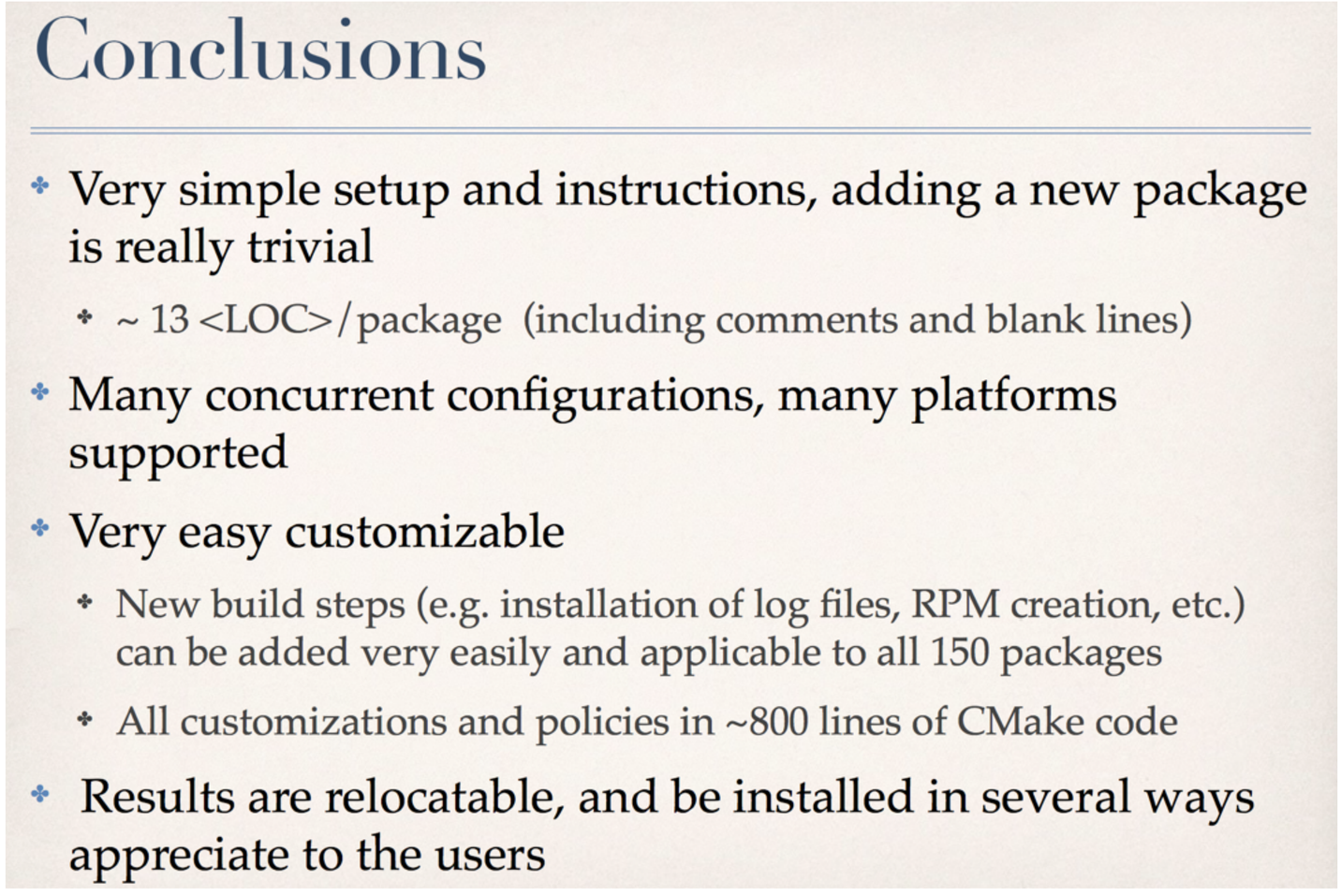
Worch Overview

Worch = **W**af + **Orch**estration:

- **software suite builder** used to build large suites of software composed of many packages from all different sources.
- **configuration manager** using a simple declarative language in order to precisely and concisely assert all build parameters.
- **workflow manager** using Waf to run interdependent tasks in parallel
- **software build features** "batteries included" for exercising many common package-level build methods
- **bootstrap aggregation** packaged using Python's `setuputils` with support for developing domain-specific extensions to easily create the build environment.
- **policy-free** leaving issues such as installation layout, target package formats, suite content, build parameters up to the end user.

Packaging Discussion #3

- LCG Packaging Infrastructure - Pere Mato Vila and Benedikt Hegner



Packaging Discussion #4

- cmsBuild - CMS SW Packaging tool, from Shahzad Malik MUZAFFAR

Summary

- Minimal standard system dependency
  - python, tar, zip, git, svn, perl, X11 ...
- Easy to build/install new package
- Multiple architectures
- Relocatable packages
- Installation of multiple versions of a package
- Multiple versions of a package within same configuration

Packaging Discussion #5

- Build and Release Issues in Accelerator Modeling - James Amundson
- Muon g-2 Development System - Adam Lyon

Contractor

Contractor: https://cdcvs.fnal.gov/redmine/projects/contractor/wiki
Remarkably similar to Worch (entirely unrelated, however)

Very small: < 2700 lines of Python
A few key differences:

- Package descriptions are Python code and can contain logic
- No external build system
  - nodes.py: 235 lines
- Has mechanism for discovering system packages
  - Used for defaults
  - Users can always choose which system packages to use and which to build

Pros & Cons

Pros:

The system works well and collaborators seem to like it

The system prevents you from making an inconsistent built

ups makes release version/release/qualifier management easy

Combine ups with CVMFS and the system is easily deployable

Cons:

Yet another different system

ups commands are strange (ups list -aK+); should rethink

Interpackage versioning is very strict and sometimes inconvenient

Summary and Outlook

- Very constructive discussions in the working group
- There is a broad range of solutions, but no controversies (yet) about the goal of the effort.
- We are now entering the analysis phase in which we will extract common requirements: for example “must be able to create relocatable packages”
- We will combine the both the taxonomies (rows) and features (columns) into a table and build consensus towards selecting a good candidate for a common tool.

https://groups.google.com/forum/#!forum/hep-sf-packaging-wg
https://indico.cern.ch/category/5816/