Karlsruhe Institute of Technology

Astroparticle Data Life Cycle

Astroparticle Package Manager (AstroPM)

Dmitriy Kostunin
March 5, 2018

INSTITUT FÜR KERNPHYSIK

Example of analysis in frame of KRAD

Tunka-Rex top-down analysis

Used software:

- `CERN ROOT`
- `AugerOffine-TRex`
- `SiMM`
- `CORSIKA` + hadronic models
- `EFieldFitter`
- + calibration, constants, etc

It is very hard to reproduce results without versioning!

- Hard to find proper revision in SVN/git/hg
- Hard to install some software (USE flags, libraries, etc)
- Hard to find proper calibration and supplementaries

Motivation

- Strict version control for simulation and analysis software
- Unified distributed storage for packages
- Easy to maintain and install

Existing solutions

- *nix package managers: too complicated, large overhead
- Auger Package Environment: proprietary(?), not distributed
- ???

Suggested solution

- Idea taken from gentoo portage
- Packages tree + distfiles
- Simplified buildfiles and maintainence
- Important cosmic-ray tools can be included immediatly at KIT: CORSIKA, KG, Tunka(Rex), Auger, etc.

Implementation

Packages tree

- Package naming following the directories structure: `<category>/<package>`
- Directory consists of few files:
  - Installer (bash) `<package>-<version>.apmi`
  - Index with versions `manifest`
  - List of distfiles with checksums `distfiles`
- Packages tree is maintainted with version control software (git/hg)

Distfiles is organized in a plain storage

Interface (.apmi)

```
astropm_pkg_install() {
        PACKAGE=$1
        VERSION=$2
        PACKAGE_DIR=$ASTROPM/ports/$PACKAGE
        BUILD_DIR=$ASTROPM/build/$PACKAGE
        astropm_pkg_check_dependencies && \
        astropm_pkg_check_files && \
        astropm_pkg_prepare_files && \
        astropm_pkg_compile_files && \
        astropm_pkg_install_files && \
        astropm_pkg_install_binaries && \
        astropm_pkg_clean && \
        return 0
}
```

Protocol

- The full revision tree is stored only on server side
- Maintainers and end-users syncronize packages via `rsync`

Mainteinence

- Package maintainer adds new installer (.apmi) and distfiles and sumbit them to repository: `astopm submit corsika`
- Submitted files stored in temporary buffer `rsync://packages.astroparticle.online/main-submit`
- Repository manager checks files in buffer and commit them to the main tree (via git/mercurial)

Syncronization

- End-user adds a repository specifying protocol and destination `rsync://packages.astroparticle.online/main-update`
- Multiple repositories are supported
- Repositories has to be syncronized (`astopm sync`)
- `USE=qgsjet astropm install simulation/corsika`

Conclusion

- It is neccessary to maintain the software envinroment in frame of KRAD/APPDS
- The standard successful approach is the package manager
- The question of choice: existing *nix, existing scientific (ape), ???
- Our own wheel/bicycle (astropm) is being developed, requires not much effort to produce and ready product