Automatic Build and Distribution of Software for MPD of the NICA Project

Ján Buša Jr.[1,2]
Slavomír Hnatič[1]
Oleg Vasilievich Rogachevsky[1]

[1]Joint Institute for Nuclear Research, Dubna, Russia

[2]A.I. Alikhanyan National Science Laboratory (Yerevan Physics Institute), Yerevan, Armenia

Yerevan
22. 10. 2024

Multi-Purpose Detector

- part of **N**uclotron-based **I**on **C**ollider f**A**cility
- **M**ulti **P**urpose **D**etector – main detector under costruction
- MPDRoot – library for reconstruction of tracks and physical analysis (http://mpdroot.jinr.ru/)
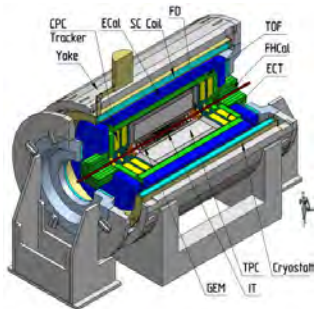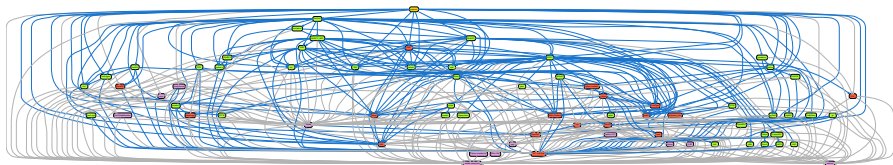
Just Install FairRoot and Build MPDRoot. . .

How to install MPDRoot (each user had to build own version):

- install all basic packages (about 70);
- download, patch, and build FairRoot;
- download and build mpdroot from sources. In-source build highly recomended.

It's really "easy":

- basic packages are often not available in compatible versions
- patching FairRoot can be tedious (during last minor version change more than 100 lines of code)
- building FairRoot takes about $2-4$ hours on a regular computer
- in-source build guides users to use improper paths when linking own code
- users tend to use same version of mpdroot for long time

59 packages are needed to run MPDRoot, about 70 to build it

Centralized Build of MPDRoot

Requirements:

- we want to deliver latest version of MPDRoot to the users
- all dependencies have to be statisfied and no version clashes are allowed
- results need to be repeatable

Solution (in cooperation with M. Vaľa, UPJŠ Košice, Slovakia):

- CVMFS – robust distribution of software not only between clusters/supercomuters
- aliBuild – set of tools for building software together with its dependencies
- toolbox – rootless wrapper for podman (docker)
- EnvironmentModules – environment enabling loading multiple versions of the same software with its dependencies

CernVM-FS
https://cernvm.cern.ch/fs/

- CMVMFS is a **read-only** file system
- anyone can mount the shared directories to its computer
- extensively uses caching of the files on local computer
- only used (accessed) files are downloaded. That makes first load slow but following are as fast as local usage
- since it is centralized, changes on server are directly available to all users

Toolbox
https://github.com/containers/toolbox

- wrapper for podman (rootless docker alternative) – ordinary user can use it
- allows user to "work" inside virtual machine – no connection to the users system
- we build primarily for Alma 9 version but still support CentOS7 due to our supercomputer Govorun requirements
- basic system image is fixed without any updates and therefore all results are reproducible
- supported on most linux-based systems. On Windows and MacOS one can use VMware (Fusion)

Environment Modules

- started as a project to separate various versions of same software
- simple "module files" contain paths to the software together with list of dependencies
- to use some software one has to load it first via command `module add mpdroot/latest`. Paths to executables as well as libraries are then added to the environment. All necessary dependencies are loaded as well.

Example of module file (excerpt):

```
module load GEANT4/$GEANT4_VERSION-$GEANT4_REVISION
module load ROOT/$ROOT_VERSION-$ROOT_REVISION
setenv MPDROOT \$PKG_ROOT
prepend-path ROOT_INCLUDE_PATH \$PKG_ROOT/include
prepend-path LD_LIBRARY_PATH \$PKG_ROOT/lib
```

aliBuild
https://alisw.github.io/alibuild/

- created as a solution to the ALICE experiment software distribution problem
- automatic build of software and all its dependencies
- reusing already built packages
- if some package was changed, all dependants are rebuild
- apropriate modules are created and numbered automatically
- can get messy if packages change frequently and are not cleared
- based on recipes – guides how to build some package

The MPDRoot Recipe

```
package: mpdroot
version: "%(commit_hash)s"
tag: "dev"
source: https://git.jinr.ru/nica/mpdroot.git
requires:
  - FairRoot
  - eigen3
  - nlohmann_json
  - VMC
  - GSL
---
unset SIMPATH
export MPDROOT=$INSTALLROOT

cmake $SOURCEDIR                                                                \
      ${BOOST_ROOT:+-DBOOST_ROOT=$BOOST_ROOT}                                  \
      ...
      ${LIBXML2_ROOT:+-DPC_LIBXML_LIBDIR=$LIBXML2_ROOT/lib}                    \
      ${NLOHMANN_JSON_ROOT:+-DNLOHMANN_JSON_ROOT=$NLOHMANN_JSON_ROOT}          \
      -DCMAKE_INSTALL_PREFIX=$MPDROOT
cmake --build .  -j$JOBS install

# Modulefile
MODULEDIR="$INSTALLROOT/etc/modulefiles"
MODULEFILE="$MODULEDIR/$PKGNAME"
mkdir -p "$MODULEDIR"
generate-module --project=NICA --bin --lib --path > $MODULEFILE
cat >> "$MODULEFILE" <<EoF
setenv MPDROOT \$PKG_ROOT
setenv MPDROOT_MACROS \$PKG_ROOT/macros
...
prepend-path ROOT_INCLUDE_PATH \$PKG_ROOT/include
EoF
```

Wrapping-Up

User's view:

- `toolbox enter a9-nica`
- `source /cvmfs/nica.jinr.ru/sw/login/login.sh`
- `module add mpdroot/v24.09.24-1`

Developer's view:

- `toolbox enter a9-nica`
- `source /cvmfs/nica.jinr.ru/sw/login/login.sh`
- `module add mpddev/latest`
- `export MPDROOT=/path/to/install`
- `git clone path_to_mpdroot && mkdir build && cd build && cmake .. && make install`

Conclusions

- for ordinary users build of mpdroot takes no time
- latest version of software is instantly available to everyone
- it is easy to return to previous versions (including dependencies)
- all results are repeatable since we store basic system image as well as all recipes
- most of the packages we build from scratch (including low level, s.a., CMake, GCC, OpenSSL)
- with recipes new dependencies are distributed among developers without necessity for them to install anything (it is done automatically)

Future Work

- follow updates of our dependencies and modify build accordingly during the run of experiment and after (30+ years)

- allow users to download "offline" version of the software – rpm
- support build for LTS versions Ubuntu (+ deb for offline)
- support MacOS

- find a way how to replace a broken package without rebuilding all