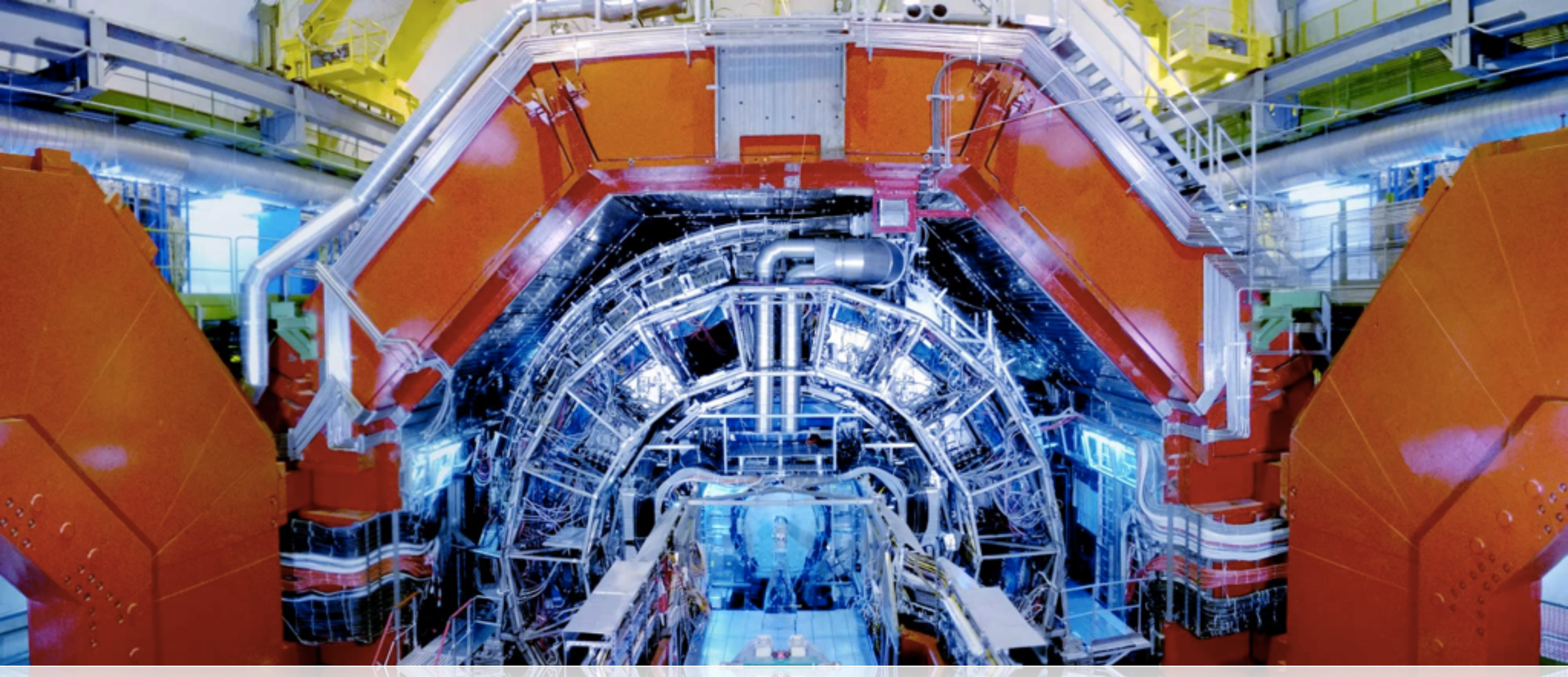


New packaging options, avoiding the dependency hell and custom compilers

Dario Berzano

ALICE Offline

ALICE Offline Week - Nov 24-27, 2015

The new ALICE build system

- The old ALICE build system could not satisfy the experiment's needs any longer
  - No efficient caching: slow time to production
  - Packages not treated equally: easy to modify AliRoot/AliPhysics recipes, difficult to add new ones
  - Unclear notifications: needed manual checks to verify that things worked, and what went wrong

We welcome the new build system

- The new ALICE build system enables us to address many long standing issues
  - Helped us separating the AliEn Runtime version without affecting Grid operations
  - Bring our own GCC to the Grid: more modern than SLC6's
  - Build and deploy event generators: EPOS, JEWEL, ThePEG…
- Most importantly: it streamlines many tasks and lets us concentrate on the important things

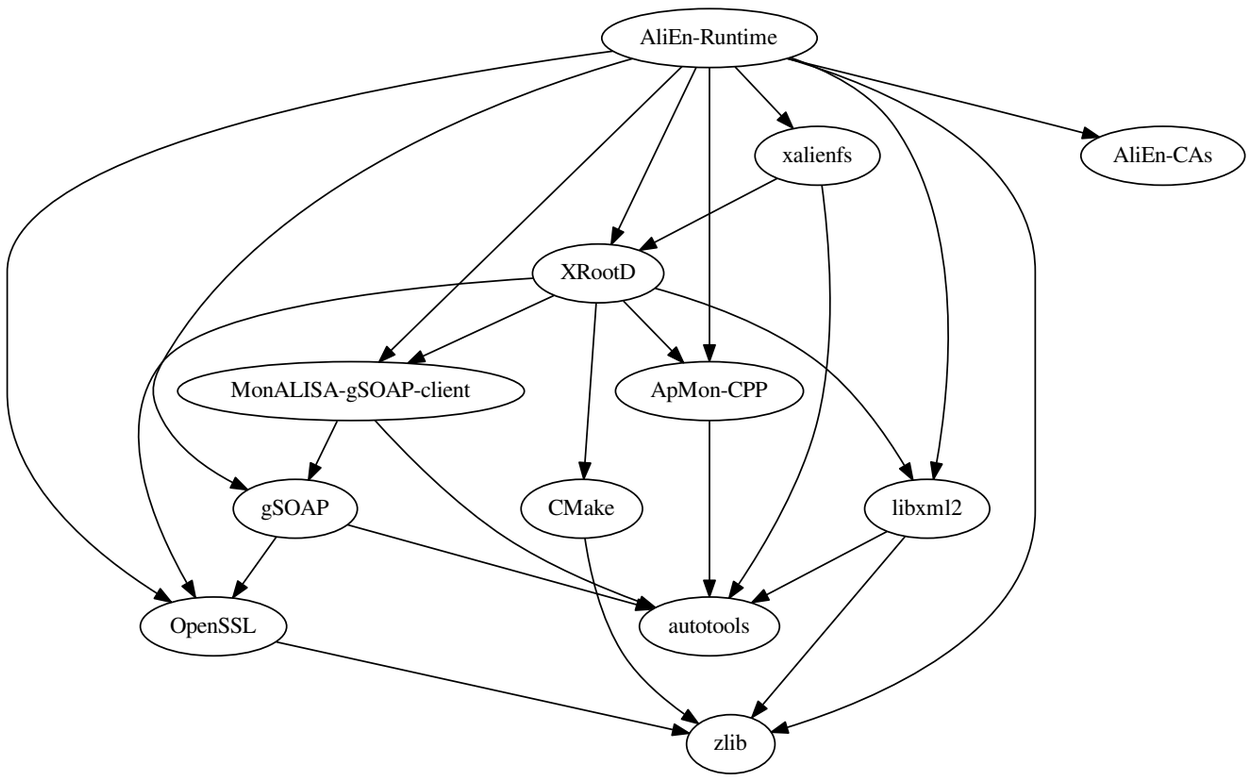
Decoupling AliEn-Runtime

AliEn Runtime

- We had a long standing ticket open: high failure rate observed with certain jobs
  - alice.its.cern.ch/jira/browse/ALIROOT-6222
- Turned out it was because of a 8-years old XRootD version shipped with AliEn

- Decoupled “services” AliEn and AliEn-Runtime seen by jobs
  - We can update (and fix!) it more frequently: simpler to test a fix with the aliBuild recipes than modifying the original ones!
  - Required if we want to bring our own compiler: we must compile AliEn with the same compiler

AliEn-Runtime



- Metapackage: painstakingly but easily converted to aliBuild recipes
- Success rate increased from 56% to almost 100%: a success
  - indico.cern.ch/event/364306 (thanks Friederike Bock!)
- Having an agile system is practically helping us to react quickly when we must provide a fix!

The dependency hell

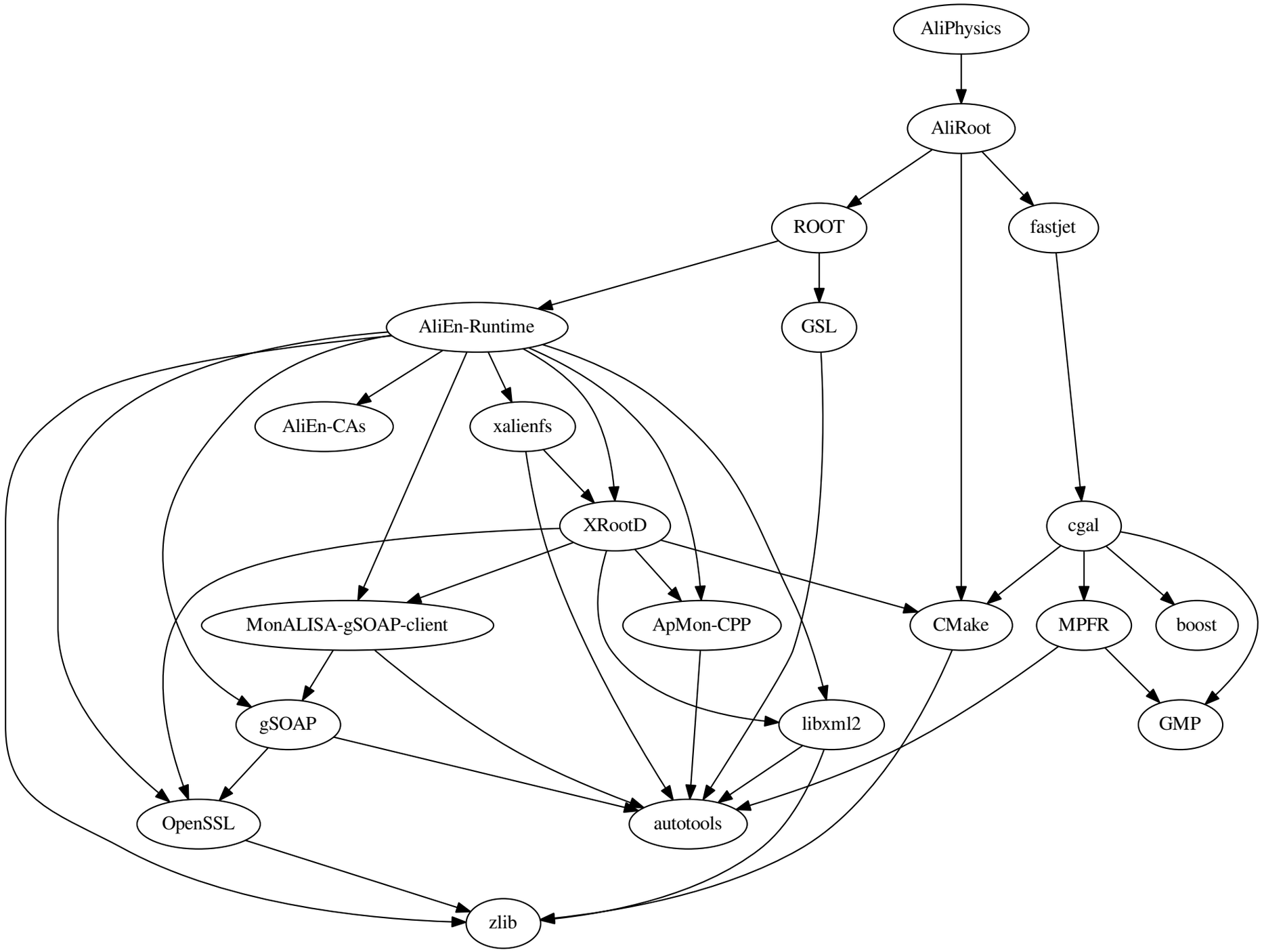
Showing dependencies

- Simple aliDeps tool in place. Requires Graphviz only
- Instant gratification:

```
# Show all recipes used in production
git clone https://github.com/alisw/alidist -b IB/v5-06/prod
git clone https://github.com/alisw/alibuild
alibuild/aliDeps all  # or packageName
```

- Doc: alisw.github.io/alibuild/extra.html

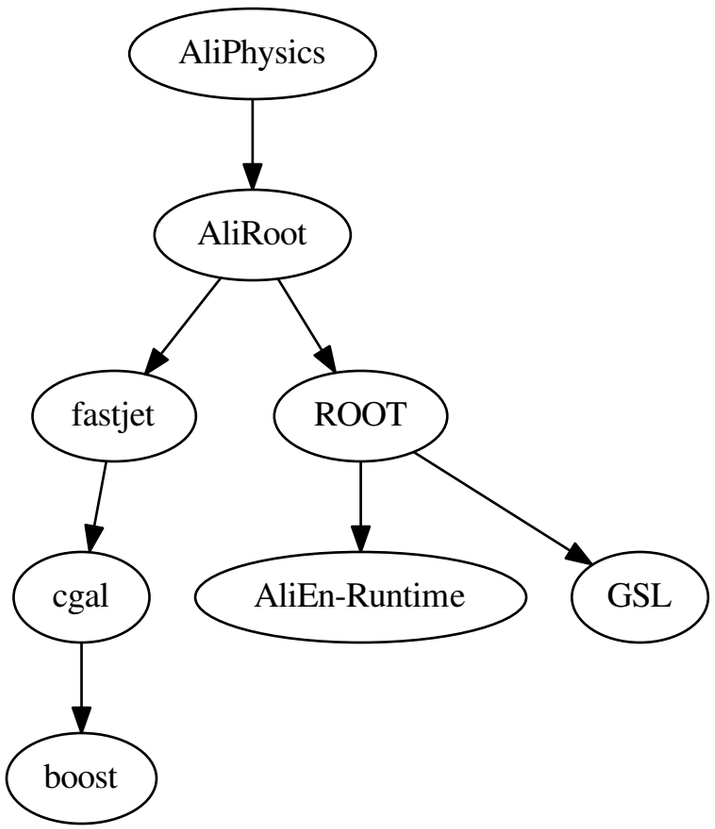
The dependency hell



All the packages required to build AliPhysics

`alibuild/aliDeps -b AliPhysics`

The dependency “purgatory”



All the packages required to run AliPhysics

`alibuild/aliDeps AliPhysics`

- Many packages were build-only requirements and they are not deployed in production: *e.g.* autotools, CMake
- Many packages are aggregated into one metapackage: *e.g.* AliEn-Runtime

Reducing the dependencies

- Can we avoid having so many dependencies? **No.**
  - Reference environment is SLC6: old, no updated stuff
  - On the Grid we must bring what we need with CVMFS

- We try to reduce the dependency hell with some techniques:
  - Do not deploy build-only requires
  - Metapackages: *e.g.* we can have one for all the generators
  - User selects only top level package and all the rest is loaded automatically

Handling dependencies

- Environment modules: modules.sourceforge.net
  - A “modulefile” per package describing environment and deps
- Used in many places:
  - Directly on CVMFS, *e.g.* from lxplus:

    ```
    alienv enter VO_ALICE@AliPhysics::vAN-20151124-1
    ```

  - Transparently on the Grid: JDL, Analysis Plugin…
  - With our RPMs (see later)

No mix and match!

- Loading the top level package only does the right thing
  - Easier and risk free
  - If we change a low level dependency you do not need to care
  - You do not risk mixing incompatible versions of dependencies
- No mix and match: an example.
  - AliRoot depends on a certain version of ROOT
  - If you load AliRoot, the correct ROOT is loaded automatically
  - If you specify manually the wrong ROOT version, things might break at runtime!
- **Always load the top level package only!**

What packages are available?

- Many versions of software are deployed: this is confusing
  - Packages list on alimonitor.cern.ch/packages is not enough
- Common user problems:
  - Which version is the right one?
  - What gets loaded if I select a certain version?
- In the next weeks we will release a new web page
  - Puts in evidence the latest versions for production and analysis
  - Clearly lists all the dependencies
- **Stay tuned: we will announce it and welcome any feedback**

Package deployment

Publishing the builds

- The build system creates generic relocatable tarballs in a repo
  - Used as a cache for the build system itself
  - Input for deploying software to many output formats
- aliPublish: publishing tool, see github.com/alisw/ali-bot
  - Periodically checks the repository for new packages
  - Include/exclude packages with rules
  - Dependencies automatically published too
  - Sends notifications when done
  - Self-heals and recovers automatically on failures
  - One tool, many output formats: CVMFS, AliEn, RPMs…

CVMFS and AliEn publishers/1

- They publish packages available on the Grid and CVMFS:
  - Results visible here: alimonitor.cern.ch/packages
- The CVMFS publisher scans for new packages every 20 minutes
  - Publishes the missing ones in a single transaction: reduces mirroring operations from Statum-0 to Stratum-1s
  - Package is relocated to the correct destination path
  - Does not step over manual CVMFS operations: checks for open transactions and exits if lock cannot be acquired

CVMFS and AliEn publishers/2

- The new publishing mechanism is very robust
  - Continuously checks if some packages are missing on the destination and tries to reach a consistent state
  - If a package is deleted by accident, it restores it: self healing
  - If it fails for temporary issues, it will recover at the next run
- This was not the case with the old mechanism
  - End of the build called back an AliEn registration which called back CVMFS in turn
  - If one of the callbacks failed, no retry and manual recovery
- **We have zero reports of "build completed but not on CVMFS"**

RPMs and other packages

- We have a publisher producing RPMs for all AliRoot and AliPhysics releases (no daily tags for the moment)
  - How to use: dberzano.github.io/alice/install-aliroot/rpms
  - Configure repo and install with yum: dependencies handled
  - Multiple versions of the same software can be installed, selected at runtime using Environment modules
- The publisher is capable of producing RPMs for every package
  - Uses fpm: github.com/jordansissel/fpm/wiki
- We are preparing the same for .deb and even OSX .pkg
  - Ready for testing before Christmas

Externals and recipe contributions

Adding new software to the system

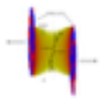
- github.com/alisw/alidist: we keep our recipes here
  - Branch IB/vX-XX/prod: recipes used in production
  - Branch IB/vX-XX/next: test branch periodically merged into prod
- New recipes are added into the next branch
  - You can fork the alidist repo and issue a pull request to the next branch if you are a power user and you want to contribute
- We require software to be on a Git repository, either the official one or a forked one with our patches on top
- As seen in Giulio's presentation: our build tool (aliBuild) can be used on your own computer, so you can test the recipe beforehand

Example: recipes for some generators

compileable versions of ThePEG, JEWEL, CRMC: #84

Merged **dberzano** merged 1 commit into `alisw:master` from `qgp:fixes_thepeg` on 15 Sep

Conversation 11 · Commits 1 · Files changed 6

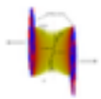


qgp commented on 10 Sep — Collaborator

- fixes in deps (capitalization)
- adding JEWEL
- adding CRMC

dberzano commented on 10 Sep — Owner

Hello **@qgp**, could you change lhapdf5 -> lhapdf? We use Git branches to distinguish between versions, for instance we have the master branch using ROOT 5 and a "ROOT 6" branch with the same recipes but ROOT version 6. Thanks!

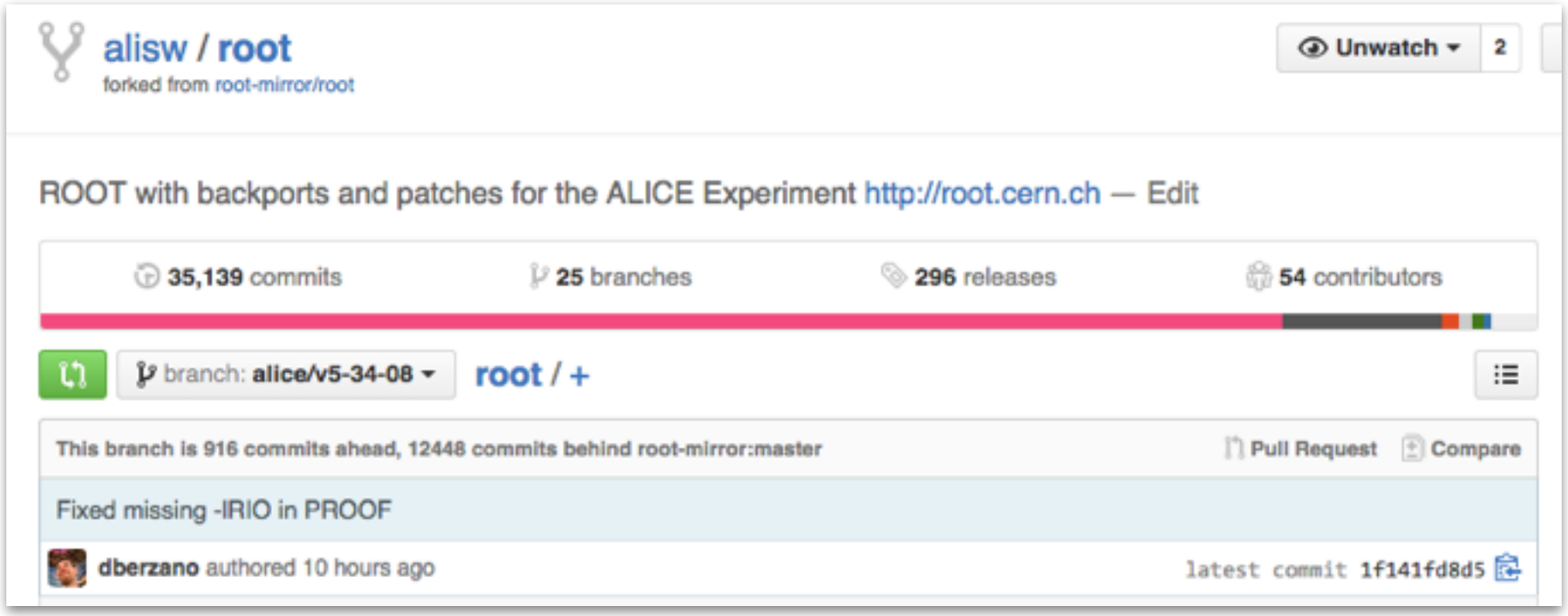


qgp commented on 10 Sep — Collaborator

The issue is depending on lhapdf v5, see alisw/alibuild#64 where **@ktf** suggested to use lhapdf and lhapdf5. The situtation is somewhat similar to pythia6 and pythia8. I can change this if there is another way to depend on a specific version of LHAPDF.

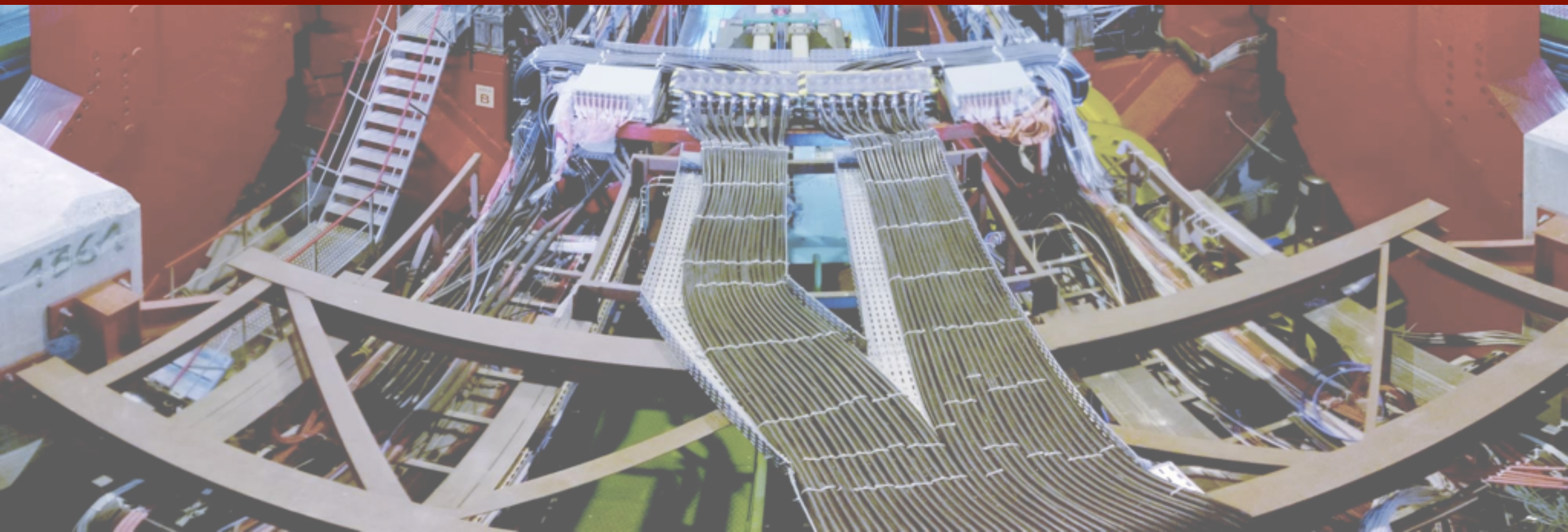
- Contributions from Jochen Klein (@qgp on GitHub)
- This speeded up the deployment and testing of generators

Externals and patches



- Fork original repo and patch in special alice/* branches
- Example: ROOT for ALICE has patches: github.com/alisw/root
  - Easily compare alice/* branches and upstream
  - Can backport minor fixes via cherry-picking
  - Users can download software already patched, no .patch files
- github.com/alisw/alidist#guidelines-for-handling-externals-sources

Bringing our own compiler



Why bringing our own compiler to the Grid?

- The new build system enables us to easily build our own compiler
  - Treated as a normal dependency
- Grid reference environment is SLC6 with GCC 4.4.7 from 2012
- Our build environment is SLC5 with GCC 4.1.2 from 2007…
- Latest GCC 4.x is 4.9.3, we want to use that one:
  - Faster build time and better optimizations (we expect better running times, we will benchmark it)
  - Full support for C++11
  - Required for migrating to ROOT 6

Current status and technicalities

- Along with GCC we also need to build Binutils (*e.g.* the linker), GDB, etc.: we are polishing the recipe and making sure it works
  - All included in a single metapackage for simplicity
- GCC very sensitive to libc: no build on SLC5 and run on SLC6
  - We need to build ROOT macros on the Grid
- Postponed the deployment after the Pb-Pb run
  - In January we will have for a period both builds with and without custom GCC for AliRoot/AliPhysics releases (non daily)
  - After validation we will build only against GCC
- Validation not an issue: we have been using modern native compilers since a long time (*e.g.* on Ubuntu, Fedora…)

Future migrations

- Having GCC as a normal recipe enables us to update it quickly
- We will no longer have to stay 8 years with the same old compiler
- Looking into GCC 4.9.3. Why not GCC 5 right away?
  - This is the next step, don’t worry!
  - GCC 5 changes the ABI for the first time in 10 years
  - Some issues here and there with ROOT 6
- Not a problem to try deploying Clang too

Will C++11 be enabled?

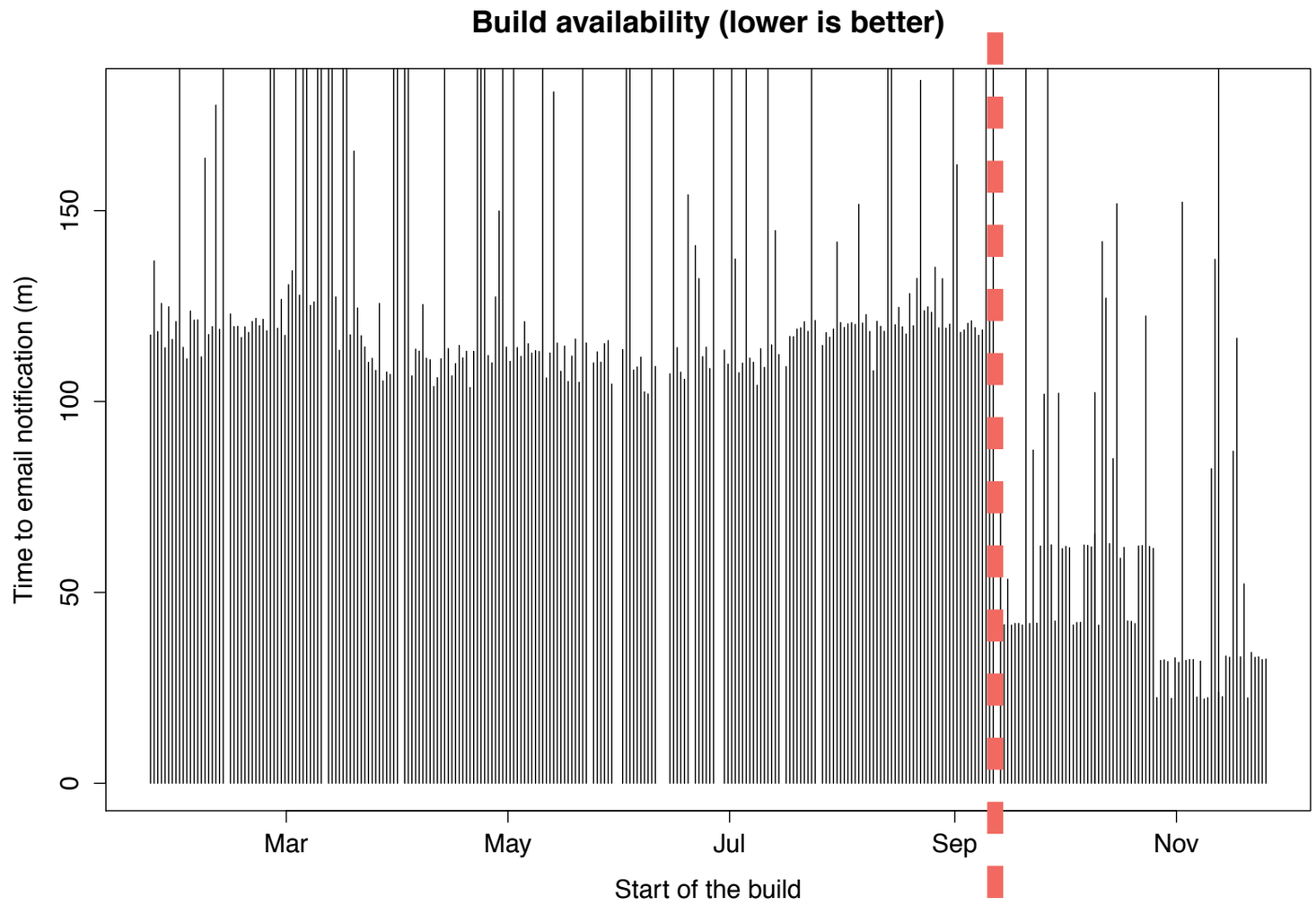
- **Yes,** but carefully.
- Online builds still use the native compiler of SLC6 and they will have to remain compatible with that
- In general, we'll be able to use it freely in AliPhysics: critical online parts are in AliRoot, which should compile with and without C++11
- We may have C++11 code in AliRoot if it is not relevant for the Online and exclude parts from the build using CMake
- We will always run integration builds on native SLC6 to spot such incompatibilities proactively

Daily tagging

New daily tagging procedure

- Before: tagging the HEAD of master at 4pm
  - Build tested on an Ubuntu machine but built on SLC5…
  - Actual build occurred on a SLC5 machine
  - Sometimes, inconsistencies and failures
- Now:
  - We first create a branch called rc/vAN-YYYYMMDD to checkpoint what we build, always at 4pm
  - We build it directly on the destination platform
  - If it works, we tag and we remove the branch
- **Tags are created only if they actually build**

Tags come much quicker



- Daily tags come much quicker than before: ~4.20pm
- If something goes wrong, we can fix it during working hours instead of discovering it at 6pm and reacting in a rush…