



ALICE

ALICE Software Deployment

HSF/WLCG pre-GDB

Nikola Hardi, for the ALICE collaboration

nikola.hardi@cern.ch

05-05-2020



OVERVIEW

1. **Build system and packaging**
2. **Containers perspective**
3. **Future development**

Software deployment pipeline


Key problems to address

- Setting up development environment
- Building and managing multiple software stacks and versions
- Continuous integration
- Providing similar development, testing and production environments
- Supporting multiple platforms
- Deploying and reusing compatible software components

	Development	CI	Deployment
Packaging	Build	Integrate	Multi-Platform
Containers	User machines	Jenkins	Grid jobs



Build system and packaging



	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

Zero to ALICE in three steps with aliBuild

1. Build the software

```
$ pip3 install alibuild
$ aliBuild build AliPhysics
```

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

2. Load the packages

```
$ alienv enter AliPhysics/latest
$ /cvmfs/alice.cern.ch/bin/alienv enter AliPhysics/vAN-20200504_ROOT6-1
```

3. Checkout source code and patch any package

```
$ aliBuild init XRootD [--defaults next-root6]
$ git clone https://github.com/xrootd/xroot # or fetch code manually
```

Some aliBuild features

Centralized dependency management with *alidist* (more details on the next slide)

- Repository with build recipes defining packages and their dependencies, and
- Alternative dependency trees defined in defaults configurations
- Build only what's needed
 - Select only missing dependencies
 - Use compatible system packages when possible
 - Detect which dependencies need rebuilding



Spack as alternative?

Multiple software versions installed side by side with *Modulefiles*

- Load packages with their dependencies, setup the environment
- Provide similar environment for development and production



ALICE

Keeping dependencies in check with alidist

alidist is git repository containing build recipes

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

Build recipe sections

1. List of build-time and run-time dependencies
2. A build script
3. Modulefile

Traceable dependency updates

- All dependency updates are visible in the git log
- The alidist repository is tagged and validated
- A stable alidist tag is used as base for daily builds
- Daily builds have fairly stable dependency tree, and only a couple of new packages



Users can build bleeding edge stack by checking out locally `alidist@master` ₇

Obtaining daily builds and reusing binaries

All daily builds are available in CVMFS

```
/cvmfs/alice.cern.ch/bin/alienv enter AliPhysics/vAN-20190505_ROOT6-1
```

Users can build any analysis framework version locally

- Rewind to corresponding alidist tag to pin dependency tree (build recipes)
- Checkout needed version of the analysis framework
- Build missing/new packages and use the full software stack locally

Common dependency binaries can be reused and shared in alicache

- The binaries can be reused, they are content addressable
- Only new packages and module files are installed in the CVMFS repository
- Individual users can also benefit from using alicache

Deployment targets and use cases

Three main deployment targets

1. User local builds
2. WLCG / CVMFS
3. O2 EPN Farm updateable RPMs (custom deployment)

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

Multiple platforms used in practice

- On the Grid, building for SLC6, running on SLC6 and CC7
- End users run wide range of platforms
 - Approximately equal share between CentOS, Ubuntu and macOS users
 - aliBuild toolset handles builds on all platforms and in multiple configurations

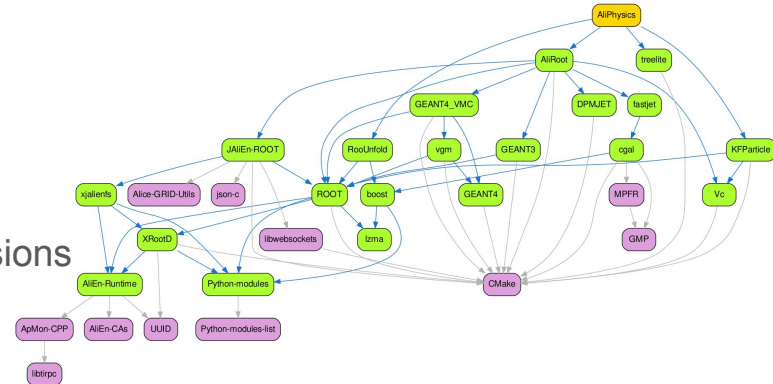
Managing dependencies on many platforms

Test system packages compatibility with *aliDoctor*

- `$ aliDoctor AliPhysics --defaults jalien`
- Describe the exact test that failed, point to version mismatch or missing OS package
- Print list of packages to be built, or to be picked up from the system


Inspect dependencies

- Draw dependency graph (graphviz, pdf)
- Detect circular dependencies
- Disable some system packages on a platform
- Multiple configuration, overriding dependencies and versions





Containers perspective



	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

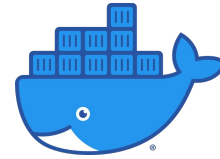
Development environment in a container

No officially supported development container

However, there is *alidock*

- A community project (collaboration with LHCb)
- Not actively maintained anymore
- Still popular with macOS users

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			



Alidock was designed to run a CentOS container with ALICE batteries included

- aliBuild preinstalled
- aliBuild workspace shared with host
- User friendly command line utility for starting and updating the container
- Additional features such as bind mounting CVMFS and GPU passthrough

AliCache is configured out-of-the box to fetch precompiled binaries.

CI, Jenkins and containers

Jenkins runs CI checks and builds in containers

Using the same aliBuild tools as for development

The container images are publicly available on Docker hub

Cluster of SLC6 and CC7 containers, plus physical machines for macOS

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

Development environment and CI share the same toolset and similar environment.

ALICE Containers on the Grid in Run 3

No support for user containers on the Grid

Common base image for all ALICE Grid jobs

- CERN CentOS 7 base image + hep-os-libs
- The image is stored in CVMFS but not yet in unpacked.cern.ch

	DEVELOPMENT	CI	DEPLOYMENT
PACKAGING			
CONTAINERS			

Singularity containers started by the pilot job

- Two layer pilot job
- Change identity and start a fresh Singularity container for each new Grid job
- Bind-mount CVMFS in the job container to load the full ALICE analysis framework

Requirements for running ALICE containers on the Grid

- Current implementation starts Singularity from host OS on the worker node
- Working towards using Singularity binaries from CVMFS
- Required CentOS 7 with *user namespaces* (Singularity in CVMFS)



ALICE

Future development

Future development

Deploy headers and use development packages from CVMFS

Improve support for sharing precompiled binaries

- Enable alicache service for more platforms, including Ubuntu and macOS

Vectorisation and AVX flavors in Run 3

- Analysis framework is designed to support vectorisation
- No dynamic AVX flavor detection yet

GPU resources on the Grid are becoming increasingly important

- ALICE O2 analysis framework for Run 3 extensively uses GPU algorithms



ALICE

Summary

Summary

ALICE is using a custom build tool, **aliBuild**

Spack is on the radar, but not for Run 3

- Routinely investigated as an alternative, but still lacking some critical features
- Missing support for multiple development packages
- Updatable RPMs

Singularity containers are planned to be used for running all Grid jobs in Run 3

- No plans for user containers on the Grid
- Development containers on user computers are not maintained at the moment
- Running GUI (X11) in development container can still be tricky
- “Notebooks” can be a good alternative (master classes, outreach)

***De facto* support for multiple platforms including: CC7, Ubuntu, macOS**



ALICE

Thank you!

Questions?