Evolving ALICE Build Infrastructure

Giulio Eulisse & Dario Berzano

Goals

- Provide Continuous Integration (CI) of AliRoot, in an automated manner.
- Provide feedback about the Continuos Integration process
- Transparently integrate in the current system
- Be "as standard as possible", integrating in CERN/IT infrastructure for (static) resource provisioning via OpenStack, without sacrificing the option of being able to use own hardware, e.g. for Mac builds, larger tests.
- Provide clear information on what sources / recipes have been used for the official build.

Continuos process

Daily build

Deployment on CVMFS

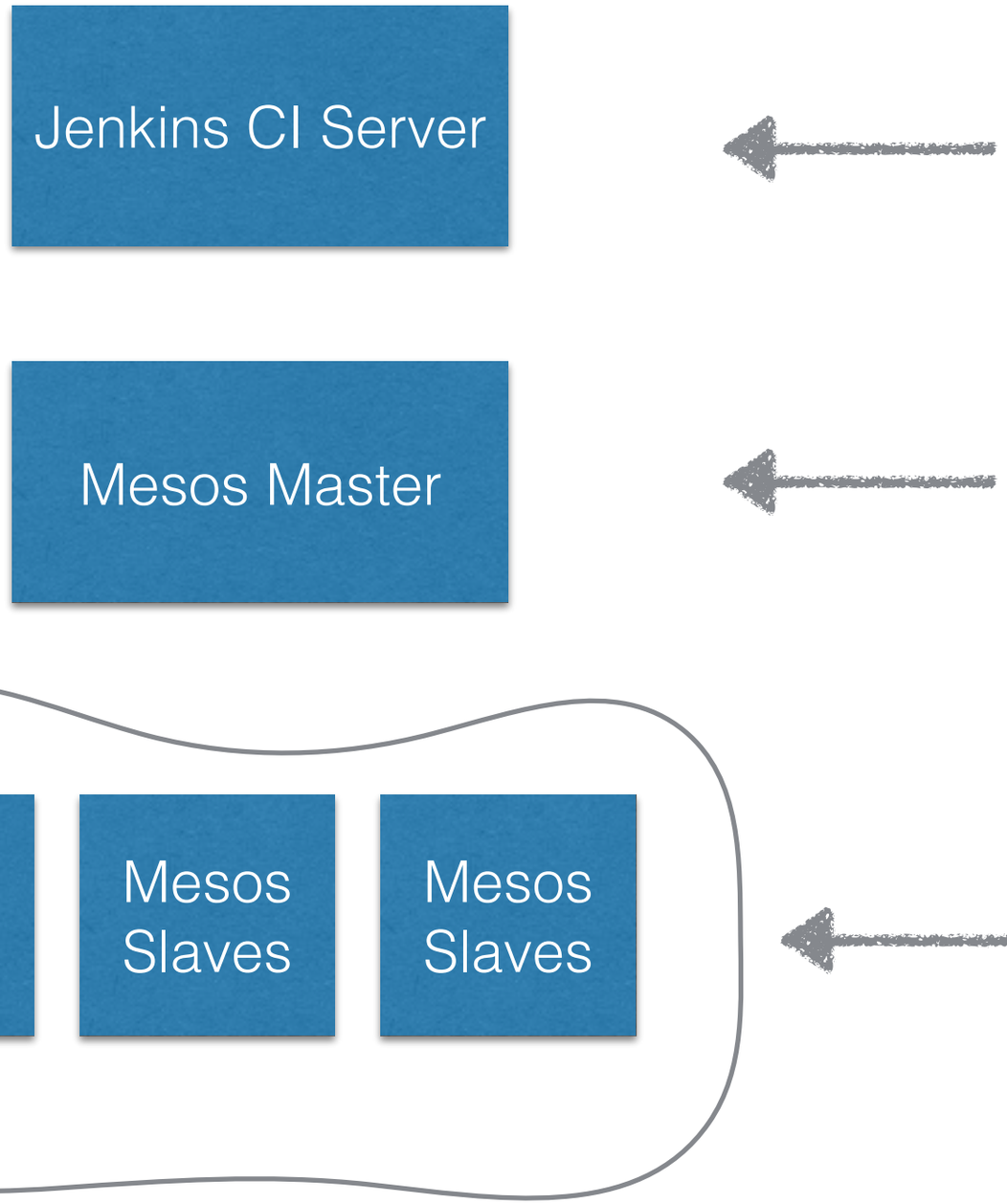
Deployment on Packman

Right now it simply means building and (unit) testing more often. In the future, if we decide to go to a GitHub / GitLab development model, it will mean that pull requests (PRs) can be checked before entering the release.

Requirements

- Multiplatform: slc5, slc6, ubuntu, centos7, MacosX
- Build (at least) the three major platforms once per day.
- Build and run test releases as commits arrive (without waiting for the daily tag) to spot issues early.
- Provide feedback in form of web pages informing what is going on and root files with the results of the tests to be downloaded.

Infrastructure



Industry standard, opensource, CI web application

We use Mesos cluster management software, providing HA, dynamic setup.

Docker containers running on a Mesos Slaves. **No more need for specially installed machines (e.g. slc5)**

Infrastructure

Jenkins CI Server

Mesos Master

Mesos Slaves

Mesos Slaves

Mesos Slaves

Build artifacts (tarballs, test root files, build logs) get copied to a temporary artifact storage to be then propagated to final deployment destination (CVMFS, Packman, Monalisa, summary web pages, elasticsearch(?)) and eventually trigger further actions in the system.

Driving the builds

- Goal: to have each build step / external fully documented and described in a simple recipe, including relevant externals.

- Two projects on GitHub:
  - the tool itself (https://github.com/alisw/alibuild)
  - the recipes (https://github.com/alisw/alidist).

- Maintaining compatibility with the current tar-balls and deployment infrastructure obviously a requirement.

Build Tool

Features:

- Small, pure python script (~270 SLOCs).
- Simple and understandable build recipes, based on YAML and bash scripts with documented conventions rather than custom language or template magic.
- Does not rebuild what is already built, rebuilds packages when recipe changes, rebuilds dependent packages if dependencies is rebuilt.
- Sources by default are hosted in a git repository, so that we can easily keep track of dependencies.
- Support for reusing prebuilt tar-balls in case of no changes will be added soon.

Build Recipes

Example recipe:

```
package: aliroot
version: master
requires:
  - geant3
source: http://git.cern.ch/pub/AliRoot
tag: master
---
#!/bin/sh

cmake . -DCMAKE_INSTALL_PREFIX=$INSTALLROOT \
        -DROOTSYS=$ROOT_ROOT \
        -DCMAKE_SKIP_RPATH=TRUE
make ${JOBS+-j $JOBS}
make install
```

How to use it:

```
git clone https://github.com/alisw/alibuild
git clone https://github.com/alisw/alidist
alibuild/aliBuild -a osx_x86-64 -j 10 -d build aliroot
```

Current status & plan

Build Infrastructure V1:

- Initial setup of Jenkins + Mesos, HA mode (3 availability zones, one can go down without need for intervention), all running on OpenStack in CERN/IT, configured by Puppet, including SSO frontend. **DONE**

- Initial docker containers to build on slc5, ubuntu, slc6, slc7. **DONE**

- Initial set of recipes and associated tool to build AliRoot and its major externals. **DONE**

- Deployment of the end-to-end chain to demonstrate CI of AliRoot, including some simple tests. **IN PROGRESS**

- Running the containers on top of real Linux HW and native builds on Mac. **TODO**

- Aggregation and initial parsing of logs, metrics. **TODO**

- Adding more tests to the setup and improving result presentation. **ONGOING EFFORT.**