CI / CD USING JENKINS

Giulio Eulisse

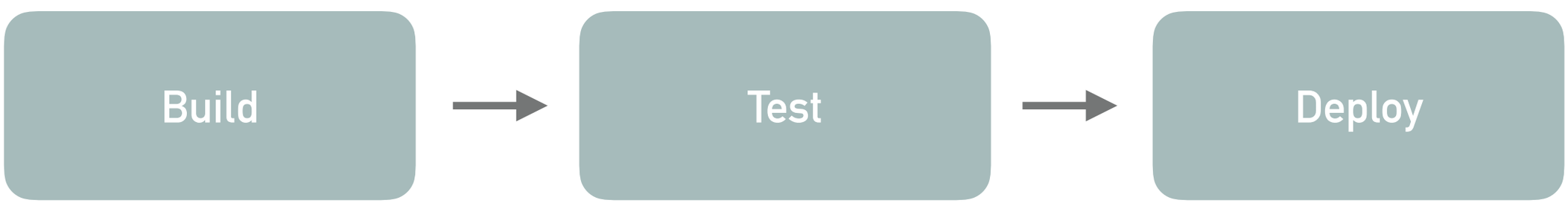
JENKINS PIPELINE PLUGIN

Towards Jenkins 2.0:

> *Jenkins is moving towards version 2.0. One of the core components (already available for our "old" 1.651 release) is the **Jenkins Pipeline plugin.***

From chained jobs to Pipelines

Jenkins Pipelines are defined in a programmatic way in so called "Jenkinsfile". This is a big difference compared to to the previous (GUI based) approach of chaining logically separated steps.



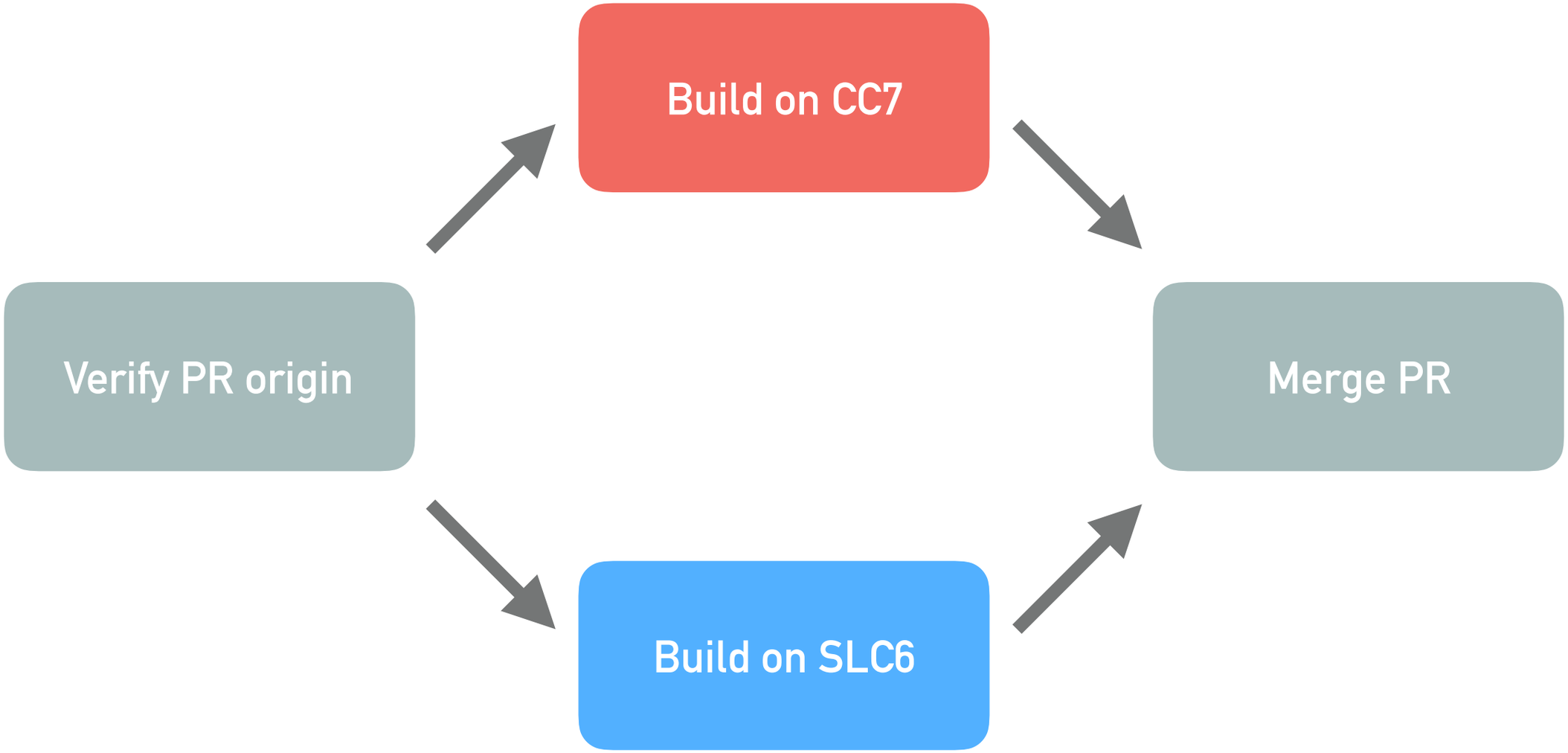
JENKINS PIPELINE PLUGIN

```
node {
    stage "Build"
    sh 'echo "A bash script"'

    stage "Test"
    sh 'echo "Another bash script"'

    stage "Deploy"
    sh 'echo "Another bash script"'
}
```

test-pipeline - Stage View

	Build	Test	Deploy
Average stage times: (Average full run time: ~2s)	424ms	1s	297ms
#8 Apr 21 22:47 — No Changes	424ms	1s	297ms
	mesos-jenkins-5eafc618854a40d3	mesos-jenkins-5eafc618854a40d3	mesos-jenkins-5eafc618854a40d3

PARALLEL PIPELINES



PARALLEL PIPELINES

```
node {
    stage "Do something"
    parallel (
      "slc7": node("slc7_x86-64-large") {
          sh "cat /etc/redhat-release"
      },
      "slc6": node("slc6_x86-64-large") {
          sh "cat /etc/redhat-release"
      }
    )
}
```

Tasks in a given stage can happen in parallel

JENKINS PIPELINE PLUGIN

```
node {
    stage "Do something"
    parallel (
      "slc7": node("slc7_x86-64-large") {
          sh "cat /etc/redhat-release"
      },
      "slc6": node("slc6_x86-64-large") {
          sh "cat /etc/redhat-release"
      }
    )
}
```

Nodes can be provisioned using Mesos + Docker

Recent Changes

Stage View

Average stage time:
(Average full run time: ~1m 13s)

	Apr 21 16:27	No Changes	1min 13s
#6			mesos-jenkins-198c57ab3bd3445
#5	Apr 21 16:25	No Changes	926ms
			mesos-jenkins-0d4c34f442c44421

Stage Logs (Do something parallel) ✖

Shell Script

```
[test-pipeline] Running shell script
+ cat /etc/redhat-release
Scientific Linux CERN SLC release 7.1 (Ptor)
```

Shell Script

JENKINS MULTIBRANCH PIPELINE

Multibranch pipeline

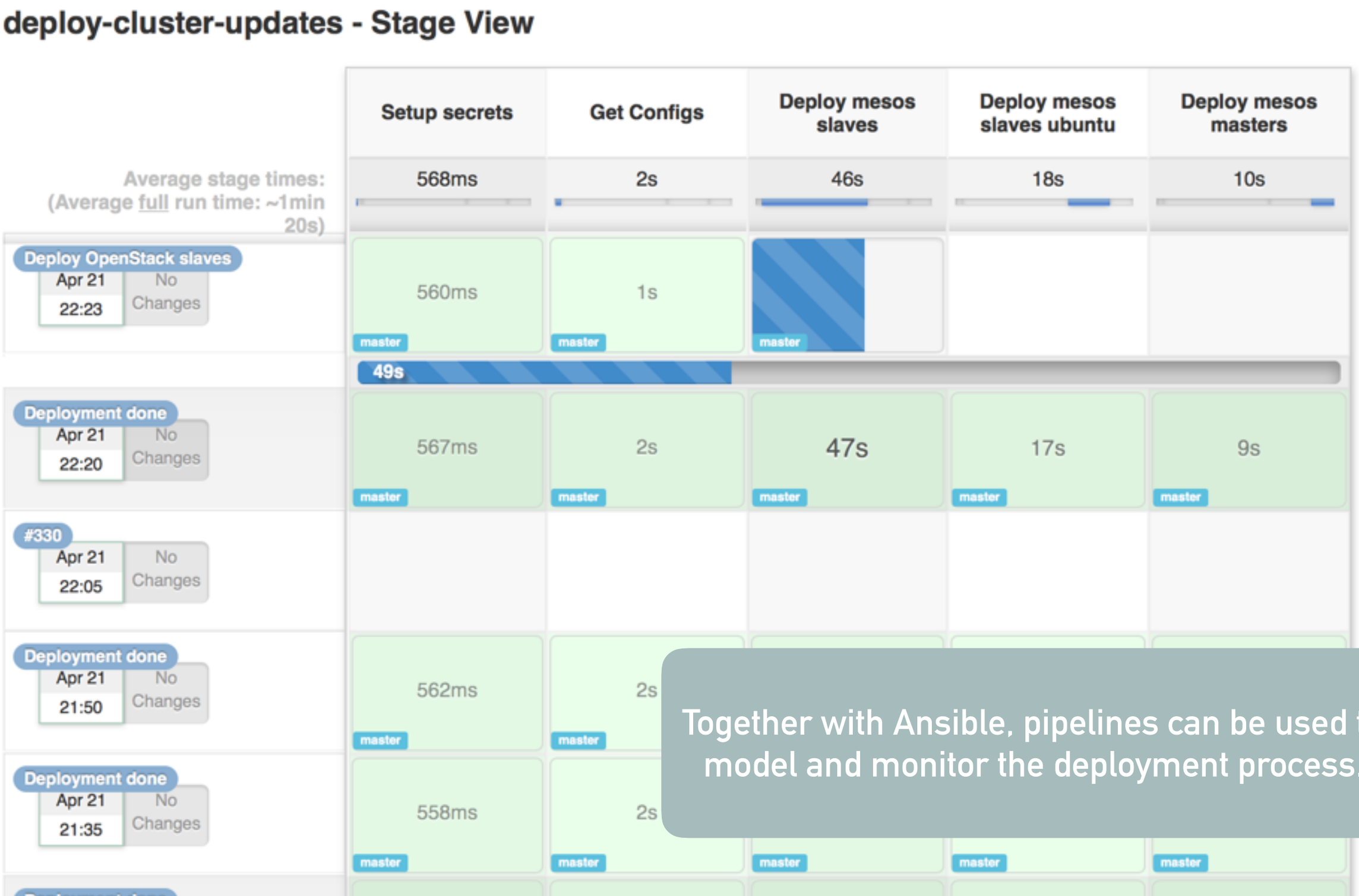
Pipelines can be triggered whenever there is a change on a branch (or on a pull request, which is simply a special kind of branch).

When this happens the pipeline defined in ***$PROJECTNAME/Jenkinsfile*** *is triggered.*

This is perfect to test Pull Requests or to take actions when a branch changes (and for example deploy its artefacts to production).

Pipeline has access to information on who did the change / what changed. This can be used, for example, to seek to "super-user" approval of PR testing when the committer is not a "usual suspect".

CONTINUOUS DEPLOYMENT



WRAPPING UP

Done:

Our Jenkins instance is now fully "Pipeline ready".

A first set of pipelines has been provided for alidist, alibuild, AliceO2 and for a number of internal tasks (e.g. deploying the Mesos cluster).

TODO:

Add more (and better) tests (e.g. fail quickly, fail early).

Add (dockerized) builders which reflect the "Online" (i.e. Barth and Vasco) setups.

Model the whole "Tag => RC => Validation => Release => Deployment" as a Pipeline.

Add more commits by pushing to the **`fix-missing-zmq-include-path`** branch on **ktf/AliceO2**.

All checks have passed

1 successful check

Hide all checks

✓ **Jenkins** — This commit looks good — Details

This branch has no conflicts with the base branch

Only those with write access to this repository can merge pull requests.