Continuous Drupal deployment with Aegir, Git, Fabric, and Jenkins

Zach Seifts
Appalachian State University
zach.seifts.us | @notzach

DrupalCamp Charlotte 2012

Who am I?

- Appalachian State University
- Over 300 instances hosted in a few Aegir instances
- Drupal, Django, Node.js, backbone, etc
- Working on the OpenBlog Project

Overview

- Why is this important?
- Tools
- Workflow
- Demo time!

What will Continuous Deployment do for me?

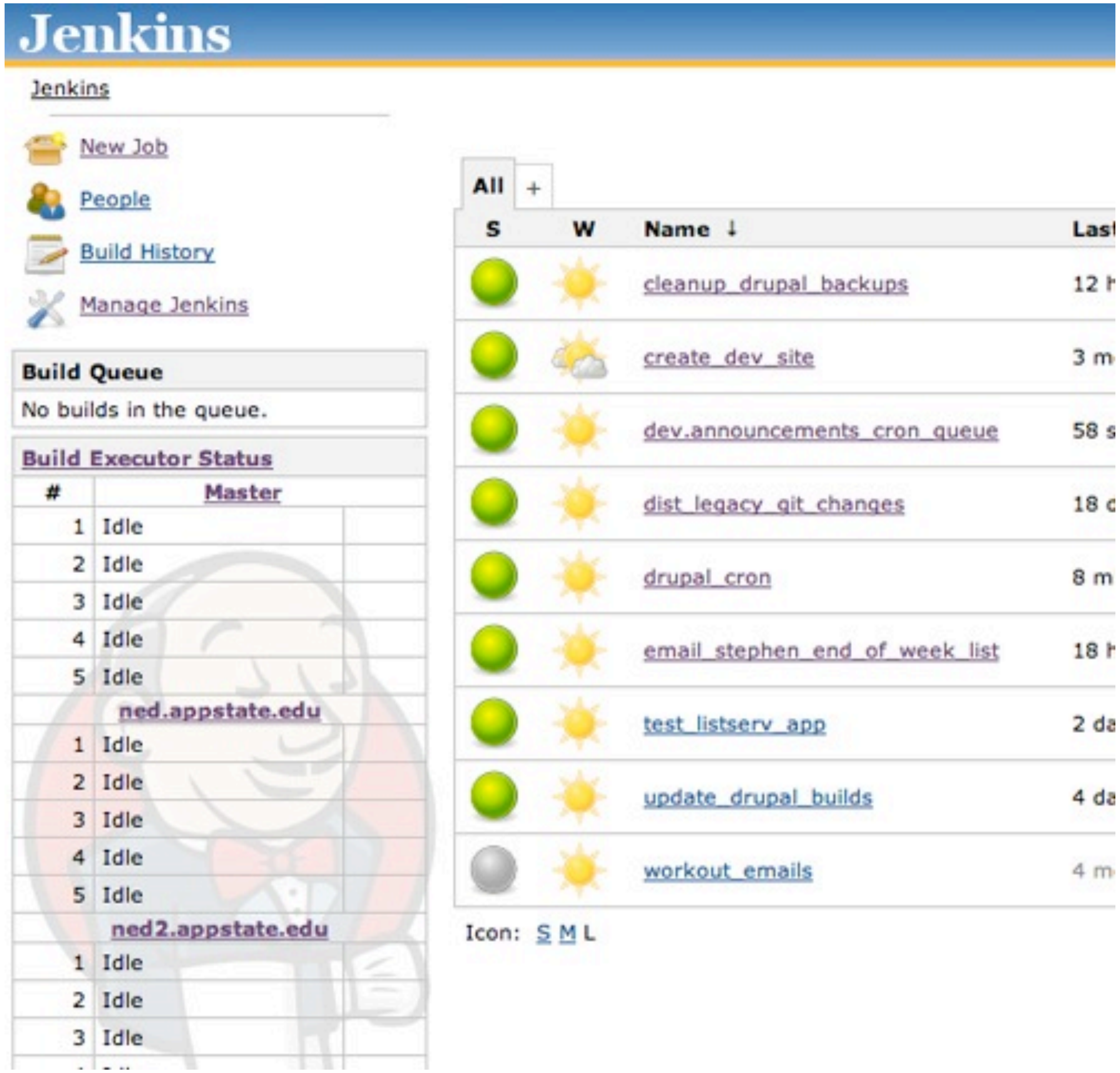
- Make sure changes don’t break existing functionality
- Automate workflows, testing, and deployment
- Keeps your production codebase current

Git

- A distributed version control system
- Every repo contains the full history of the project
- Workflows!
- Branching that works
- Sub-trees, remote origins, rebasing, and other cool things
- Great tools to manage your repos

Jenkins

- Java based CI server
- Large selection of plugins
- Remove instances
- Think of cron but with more options
- Awesome



Jenkins

Jenkins

New Job

People

Build History

Manage Jenkins

Build Queue

No builds in the queue.

Build Executor Status

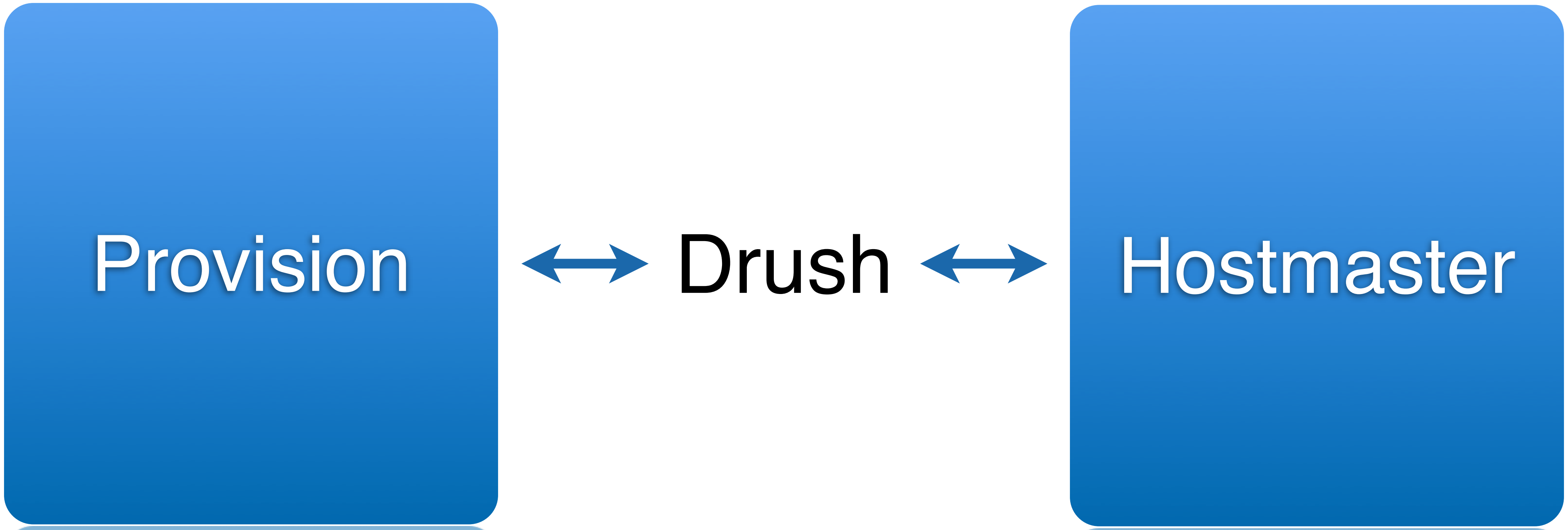
#	Master
1	Idle
2	Idle
3	Idle
4	Idle
5	Idle
	ned.appstate.edu
1	Idle
2	Idle
3	Idle
4	Idle
5	Idle
	ned2.appstate.edu
1	Idle
2	Idle
3	Idle

All +

S	W	Name ↓	Last
●	☀	cleanup_drupal_backups	12 h
●	⛅	create_dev_site	3 m
●	☀	dev.announcements_cron_queue	58 s
●	☀	dist_legacy_git_changes	18 d
●	☀	drupal_cron	8 m
●	☀	email_stephen_end_of_week_list	18 h
●	☀	test_listserv_app	2 da
●	☀	update_drupal_builds	4 da
○	☀	workout_emails	4 m

Icon: S M L

Aegir



Provision ↔ Drush ↔ Hostmaster

- Aegir is a Drupal based hosting environment for hosting Drupal sites
- Built around drush and Drupal multi-site
- Easily create, update, delete sites, platforms, and servers
- Easily extendable

Aegir

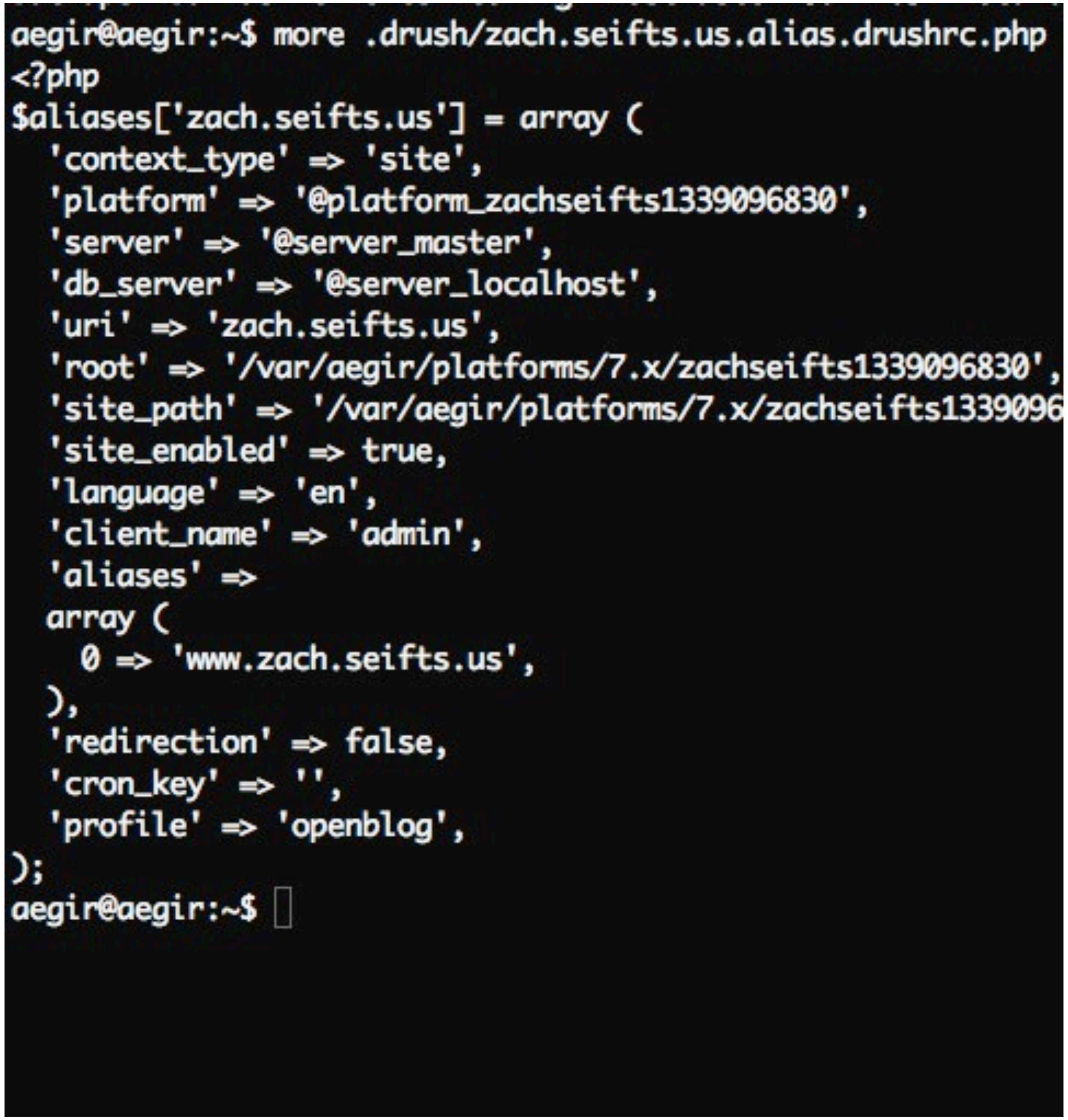
Provision

- Backend command line interface for managing Aegir
- Handles server config, apache or nginx config, databases, etc
- Represents the different entity types with named contexts, similar to site aliases

```
$ drush @site.example.com provision-verify
$ drush @sites.example.com provision-migrate @platform_newplatform
$ $(drush @site.example.com sql-connect)
```

Aegir

Provision



```
aegir@aegir:~$ more .drush/zach.seifts.us.alias.drushrc.php
<?php
$aliases['zach.seifts.us'] = array (
  'context_type' => 'site',
  'platform' => '@platform_zachseifts1339096830',
  'server' => '@server_master',
  'db_server' => '@server_localhost',
  'uri' => 'zach.seifts.us',
  'root' => '/var/aegir/platforms/7.x/zachseifts1339096830',
  'site_path' => '/var/aegir/platforms/7.x/zachseifts1339096
  'site_enabled' => true,
  'language' => 'en',
  'client_name' => 'admin',
  'aliases' =>
  array (
    0 => 'www.zach.seifts.us',
  ),
  'redirection' => false,
  'cron_key' => '',
  'profile' => 'openblog',
);
aegir@aegir:~$ 
```

Aegir

- The user interface to administrate sites
- Build around the hostmaster distro
- Communicates with the backend with named contexts
- Extendable

Hostmaster

Fabric

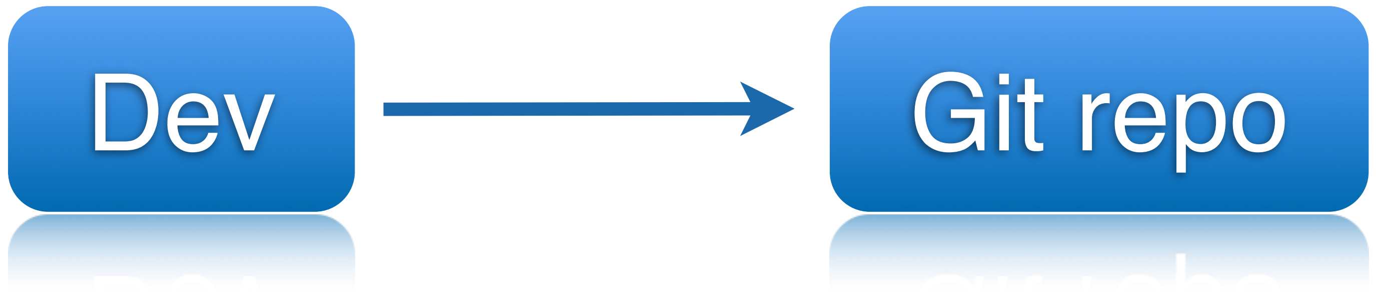
- A python library and command line tool for performing system admin tasks

```
from fabric.api import run, task

@task
def my_task(thing):
    ‘’’’ Docstring for my task.
    ‘’’’
    run(‘pwd’)
    run(‘echo %s’ % (thing,))
```

```
$ fab -H localhost,prod1 my_task:thing=foo
[localhost] run: pwd
[localhost] out: /Users/zach/dev
[localhost] run: echo foo
[localhost] out: foo
[prod1] run: pwd
[prod1] out: /home/www
[prod1] run: echo foo
[prod1] out: foo

Done.
Disconnecting from localhost... done.
Disconnecting from prod1... done.
```

The workflow

1 The developer merges changes into the production branch and pushes



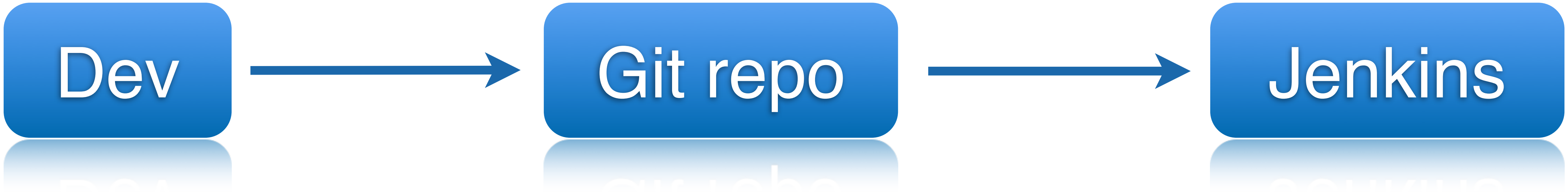
Dev

Git repo

The workflow

1 The developer merges changes into the production branch and pushes

2 Jenkins sees the changes, builds a testing instance and starts running jobs

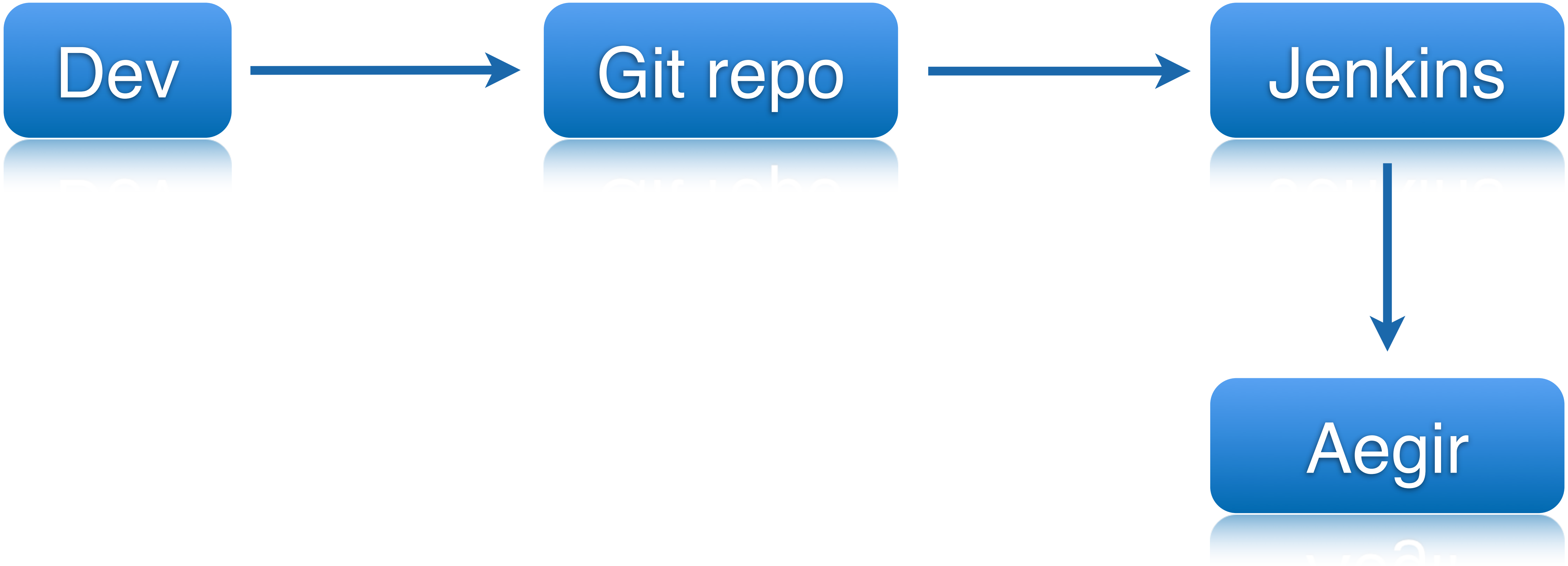


The workflow

1 The developer merges changes into the production branch and pushes

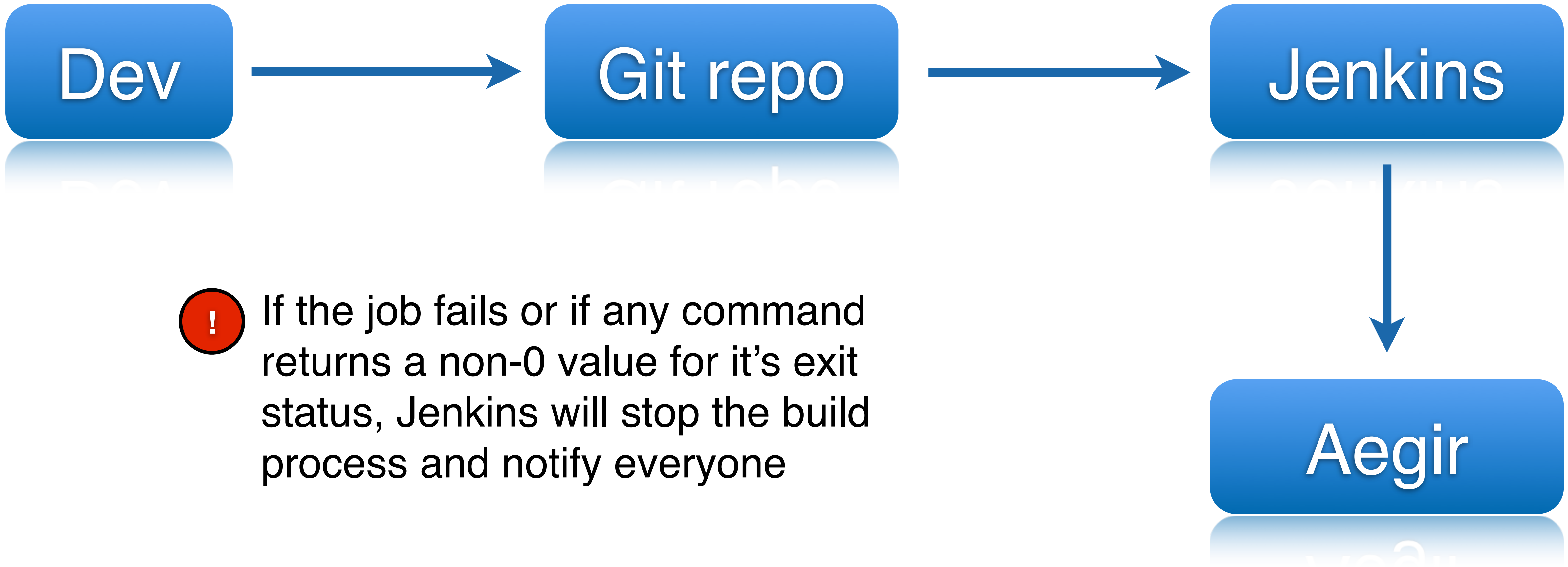
2 Jenkins sees the changes, builds a testing instance and starts running jobs

3 If the jobs succeed, Jenkins starts to deploy changes to production



The workflow

1. The developer merges changes into the production branch and pushes
2. Jenkins sees the changes, builds a testing instance and starts running jobs
3. If the jobs succeed, Jenkins starts to deploy changes to production



Dev

Git repo

Jenkins

Aegir

! If the job fails or if any command returns a non-0 value for it’s exit status, Jenkins will stop the build process and notify everyone

Demo time!

Learn more

- http://community.aegirproject.org
- http://jenkins-ci.org
- http://docs.fabfile.org
- http://git-scm.com

Questions?