





Multidimensional testing workflow before merge to master

@ygerasimov @podarok

<https://github.com/propeoplemd/cibox>



Introduction

Yuriy Gerasimov

Drupal Architect

<https://www.drupal.org/u/ygerasimov>

Andrii Podanenko

DevOps Architect

<https://www.drupal.org/u/podarok>





Development infrastructure

- single development environment
- separate company dev server (multiple vhosts one per project)
- SaaS solutions (Acquia, Pantheon, ...)





Coding process

- all commits directly to master
- master deployed to dev site (sometimes automatically)
- code review on dev site after deployment





Configuration process

- master database, get pulled to local dev environments (backup & migrate)
- changes to configuration happens on dev site manually





Problems

- shared resources on dev environment, dev conflicts (cache, solr)
- too much time to configure all services locally (varnish, solr)
- large distance between Dev and Ops (only one guru able to do production release)





Local development

- local virtualized environment (vagrant)
- started from puphpet.com but switched to ansible
- based on ubuntu 12.04 (upgrading to 14.04)





Database

- no configuration on demo/stage/prod sites
- code driven development
- database/profile workflow





Code process

- Github pull requests workflow
- code review before merge to master
- code style checks, test runs
- security tests
- QA on builds





Extra tools

- os monitoring to see resources consuming operations (multinode munin)
- visual regression testing
- automated complex deployments Acquia deployment
- urls health checks





Visual regression

- <http://backtrac.io> SaaS
- screenshots before / after release, diff
- automated scheduled screenshots / diffs
- authenticated user
- register and start using now!







URLs health checks

- <https://github.com/ygerasimov/website-size-scan>
- scans all URLs in the file, checks sizes of images
- logs 404s, 5xx, etc.





Welcome CIBox

<http://bit.ly/ffw-cibox>

FFWTM



CIBox code structure

CI for project and VM with Drupal initial codebase

Continuous Integration for a project (*jenkinsbox.yml*)

- Server preconfigured packages, swap, needed stuff
- Jenkins, sniffers, ansible
- Jenkins preconfigured jobs
- Apache, mysql, solr etc
- Optimized configs, vhosts etc

Project initial creation playbook (*github.yml*).

- Get latest Drupal
- Install basic profile
- Inject Vagrant VM config
- Inject DevOps scripts
 - reinstall.yml,
 - sniffers.yml,
 - tests.yml,
 - security.yml



How to deploy whole system

Steps for getting started

1. CI server

- Get virtual or real server from hosting provider (*Ubuntu LTS 64 bit only for now*)
- Set root password
- Make initial config changes to `jenkinsbox.yml` and inventory
- Run ***ansible-playbook jenkinsbox.yml*** from a shell
- Make changes to jenkins UI with credentials to github repo

2. Repository initialization

- Make needed changes to `github.yml`
- Run `ansible-playbook github.yml`
- push generated codebase folder to github repo
- check Pull Request builder with newly created change to `readme.md`
- profit



jenkinsbox.yml

ansible playbook for installing CI server













- Jenkins powered install
- Needed Jenkins's plugins
- LAMP stack + SSL
- PHP Code Sniffer, scss-lint
- Java JDK
- Jetty & Apache Solr
- Selenium & Behat packages
- Optimized and preconfigured configs

```
roles:
```

```
- { role: cibox-swap, tags: ["cibox-swap"] }
- { role: cibox-misc, tags: ["cibox-misc"] }
- { role: ansible-php-pear, tags: ["ansible-php-pear"] }
- { role: ansible-role-php-xhprof, tags: ["ansible-role-php-xhprof"] }
- { role: ansible-composer, tags: ["ansible-composer"] }
- { role: ansible-role-mysql, tags: ["ansible-role-mysql"] }
- { role: cibox-jenkins, tags: ["cibox-jenkins"] }
- { role: cibox-jetty-solr, tags: ["cibox-jetty-solr"] }
- { role: cibox-sniffers, tags: ["cibox-sniffers"] }
- { role: cibox-mysql-config, tags: ["cibox-mysql-config"] }
- { role: cibox-ssl-config, tags: ["cibox-ssl-config"] }
- { role: cibox-behat-selenium2, tags: ["cibox-behat-selenium2"] }
```

Preconfigured Jenkins

A bunch of jobs with scripts for running playbooks

All		
S	W	Name ↓
		ASC_DEMO_TESTS
		ASC_PR_BUILDER
		BACKUP_PROD_DB
		DEMO
		DISK_USAGE_TRIGGER
		SERVER_CLEANER

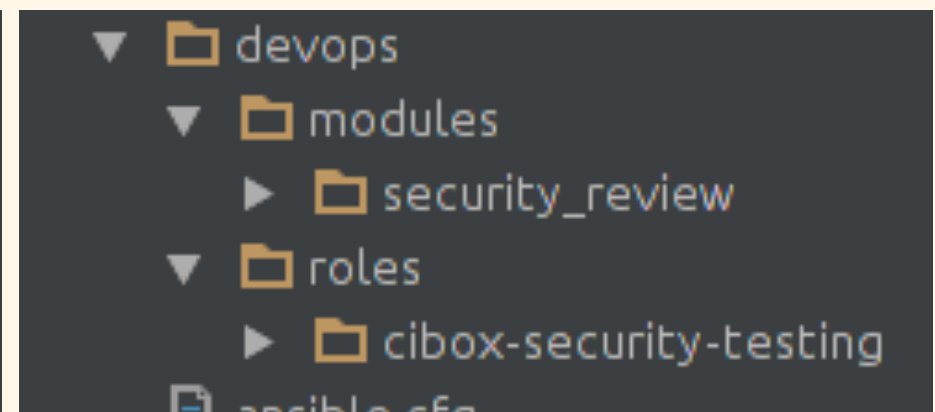
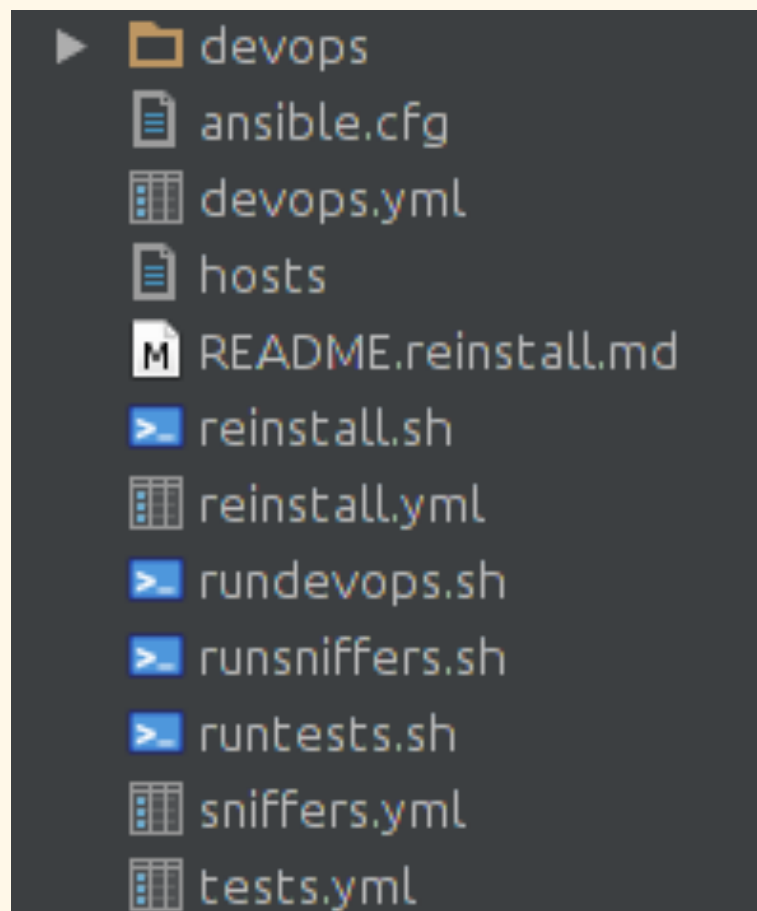
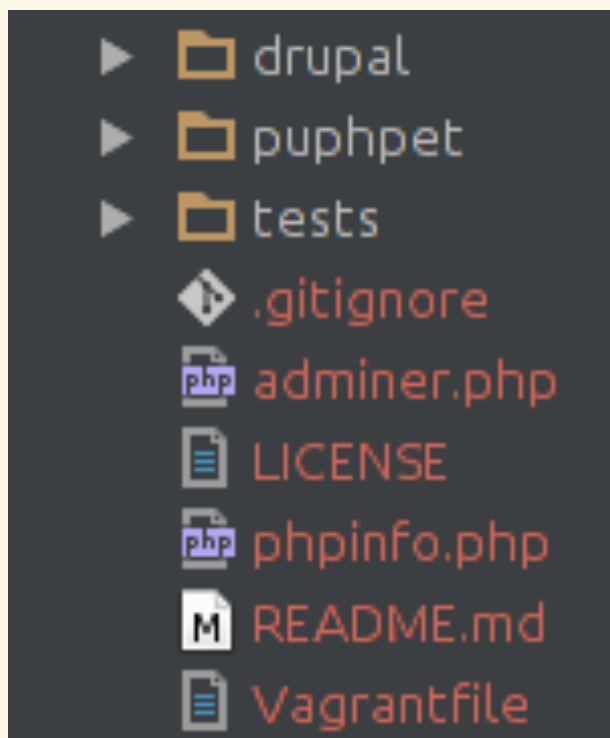
Значок: [S](#) [M](#) [L](#)

- Pull Request Builder
- Skeleton for Backup production database
- Demo site builder
- Disk space cleaner

Main project code structure

Latest drupal, adminer, devops scripts, basic profile

During run of ***github.yml*** you'll get a codebase that has latest drupal in drupal folder and scripts for future CI builds and tests.





Vagrant box

Vagrant + virtualbox (optional lxc) + ansible provisioner

We are using trick for sharing ansible roles between CI server and VM provisioning scripts for making sure we have equal environments for both.

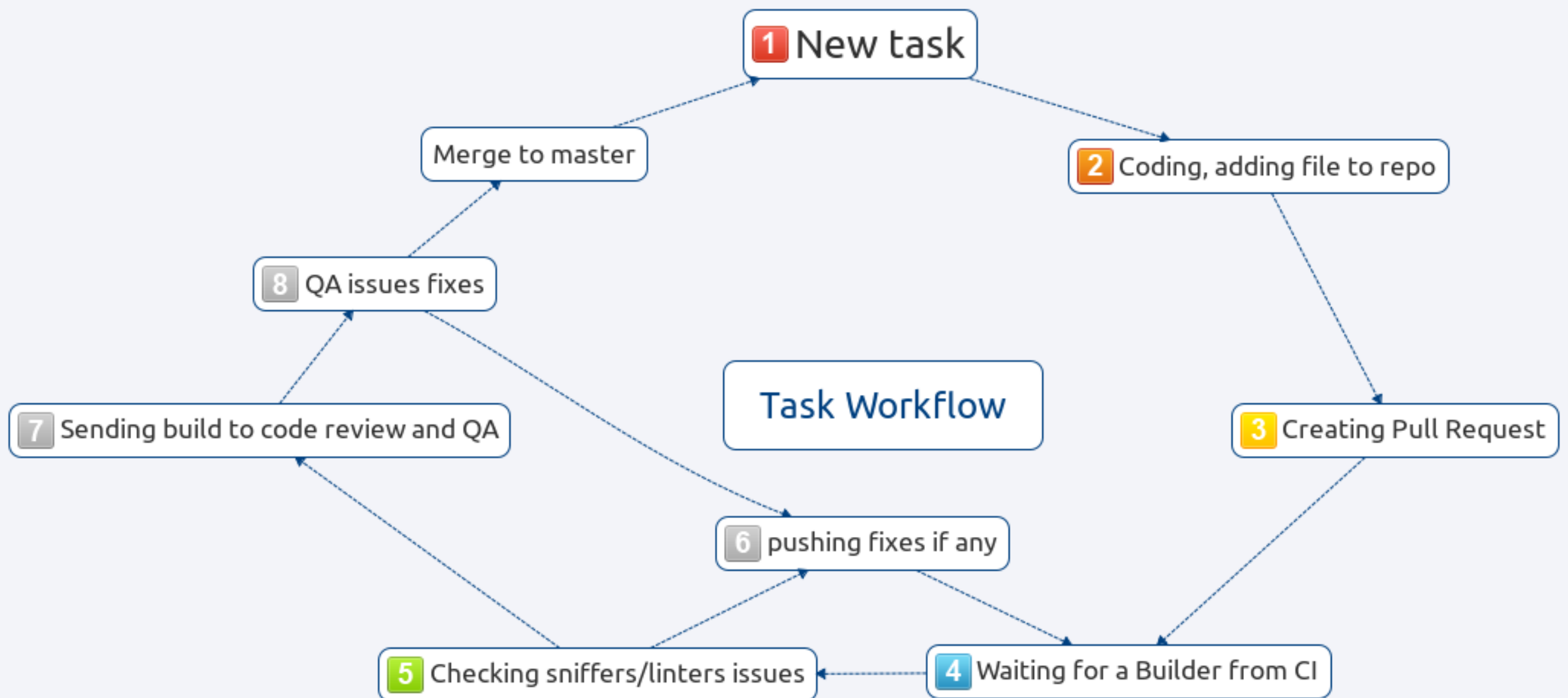
Basic stuff for now (all are inside splitted ansible roles):

composer, pear, ansible, apache, memcached, mysql, php, sdebug, shprof, selenium, behat, drush, jetty solr, phpdaemon, php codesniffer, apache ssl, custom swap.

Just **vagrant up** and you are ready to go coding.

Single task flow

Developer point of view



Comments by CI bot



ppbot commented on 1 Apr

Collaborator



Build comment file:

CodeSniffer: Drupal standard file <http://188.40.239.202/build59/Drupalsniff.txt>

CodeSniffer: DrupalPractice standard file <http://188.40.239.202/build59/DrupalPracticesniff.txt>

CodeSniffer: DrupalSecure standard file <http://188.40.239.202/build59/DrupalSecuresniff.txt>

CodeSniffer: Features Drupal standard file <http://188.40.239.202/build59/FeaturesDrupalsniff.txt>

CodeSniffer: Features DrupalPractice standard file

<http://188.40.239.202/build59/FeaturesDrupalPracticesniff.txt>

CodeSniffer: Features DrupalSecure standard file

<http://188.40.239.202/build59/FeaturesDrupalSecuresniff.txt>

JSHint: modules standard file <http://188.40.239.202/build59/modulesjshint.txt>

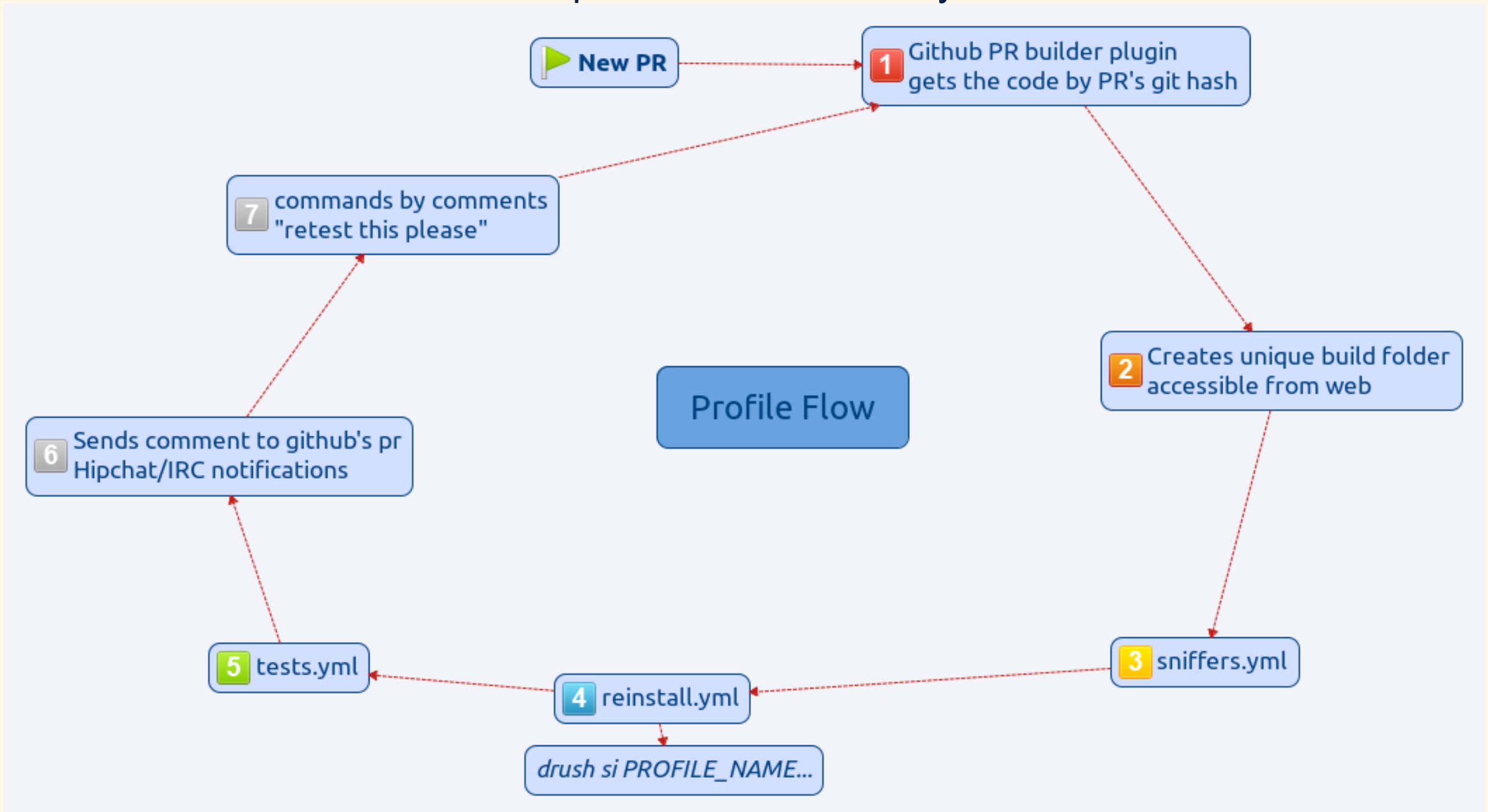
JSHint: themes standard file <http://188.40.239.202/build59/themesjshint.txt>

SCSS-lint: themes standard file <http://188.40.239.202/build59/scsslintthemes.txt>

Build site installed at <http://188.40.239.202/build59>

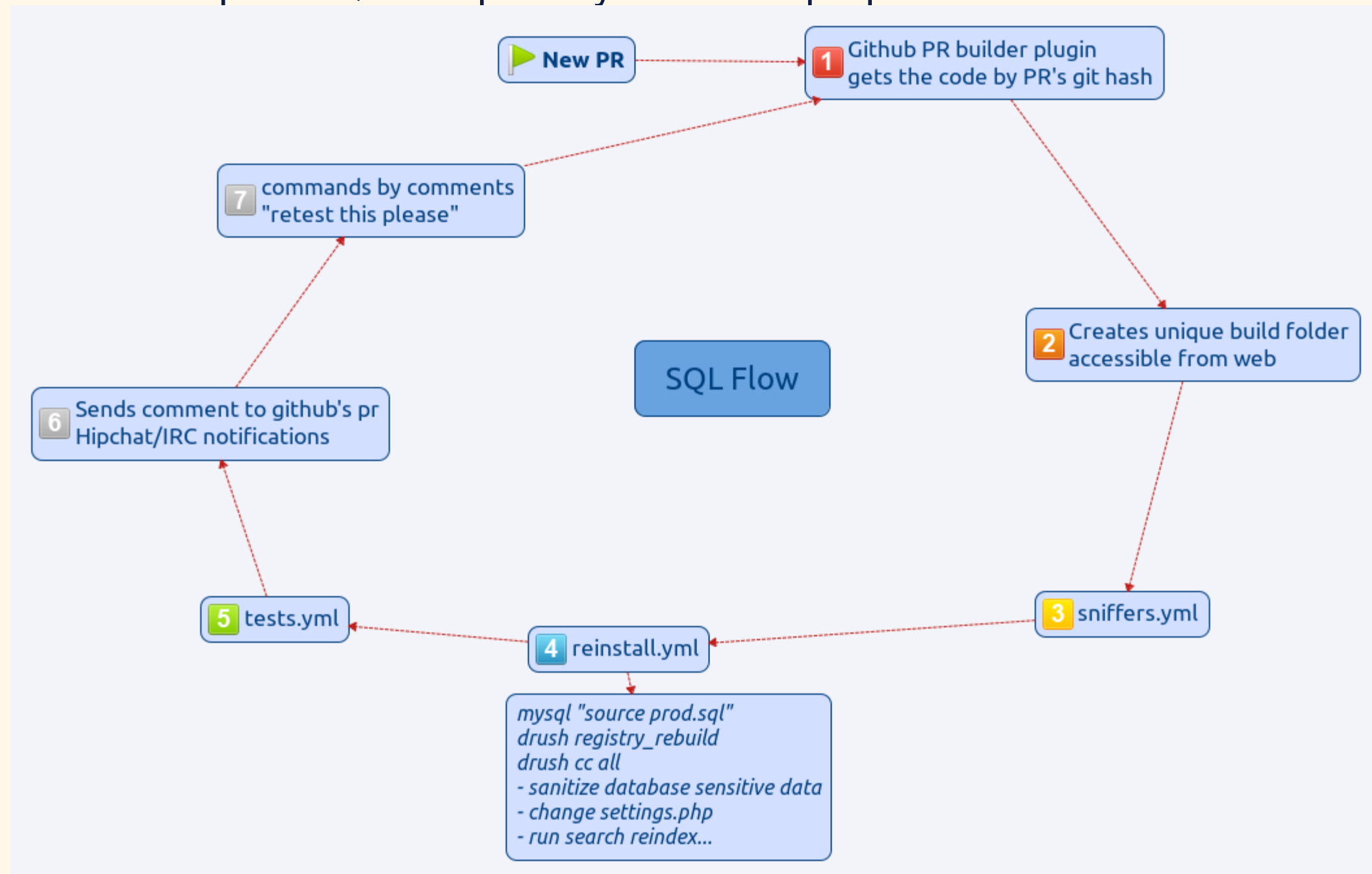
Profile based flow

Reinstall Drupal from scratch every builder time



SQL based flow

Import SQL dump every build and prepare it to codebase





Team rules

- Never merge own Pull Request
- Never push directly to main repo master branch
- ***master*** branch is stable
- There should be two siblings for every role (optional)
- Bugs introduced from specific PR would be nice to assign to its reviewer. (optional)



Development phases

1. Reinstall from scratch every build by `reinstall.yml` playbook
2. Update path, content can be edited at stage by `reinstall.yml` playbook and *pp_environment: staging*
3. SLA (production) update path with QA testing on stage



Responsibility

Due to the fact all DevOps scripts are in the same repo with a project itself - any developer can change workflow at any point.

Team does manage all the steps for DevOps scripts, no need to involve Ops into the team.



Flow Bottlenecks

- Dependency from github(gitlab, bitbucket)
- If CI server down - team get stopped on code review step
- New developers should follow new rules. (Coder is tough)
- DevOps must be a team member(s)
- Code Review get hurt
- Builds are slow on huge projects
- Decent desktops for a team (SSD is a must)
- Minimal task ≥ 1 hour



Usefull links

Documentation

- <https://github.com/propeoplemd/cibox>
- <https://github.com/propeoplemd/cibox/blob/master/README.md>
- <https://github.com/propeoplemd/cibox/blob/master/github/files/drupal7/scripts/README.reinstall.md>

Presentations

- <http://www.slideshare.net/podarok/drupal-continuous-integration-workflow>
- <http://www.slideshare.net/podarok/start-using-vagrant-now>
- <http://www.slideshare.net/podarok/live-deployment-ci-drupal>
- <http://www.slideshare.net/ygerasimov/ci-drupal-camp-berlin-2014>
- <http://www.slideshare.net/ygerasimov/vagrant-stanford-drupalcamp-2014>
- <http://www.slideshare.net/ygerasimov/continuous-integration-stanford-2014>

Blog posts

- <http://wearepropeople.com/blog/how-we-use-vagrant-in-our-drupal-development-workflow>
- <http://wearepropeople.com/blog/building-quality-into-drupal-development-workflow>



Big thanks to

Jeff Geerling
@geerlingguy





Evaluate Us Please

<http://bit.ly/ffw-cibox-evaluate>



WHAT DID YOU THINK?

EVAULATE THIS SESSION - [LOSANGELES2015.DRUPAL.ORG/SCHEDULE](https://losangeles2015.drupal.org/schedule)

Andriy Podanenko
DevOps Architect



Yuriy Gerasimov
Drupal Architect

THANK YOU!