Development Street: Automated testing

David Diks, Yannic Riesebos, Jandie Hendriks

March 14, 2018

1 Strategy

Automated deployment through multiple stages is the backbone of a successful project. The setup for this usually involved multiple different services each providing their individual piece of the puzzle, usually having the following components;

- Gitlab
- Jenkins
- Artifactory
- Sonarqube

However, a setup like this is error-prone due to the amount of moving necessary moving parts. A single component updating could break the entire chain and stop the deployment. This not only negatively impacts developer workflow but also the delivery of software. Therefore we should look into an approach that minimizes dependencies and single points of failure.

Another approach is the use of two different build systems. One of these systems is in charge of authoritative building (EG: Approving commits, checking merge requests) and another system is running as a deployment indicator (EG: Saving artifacts, managing production and qualification). Bonus points are scored if these services are externally hosted, thus reducing the amount of change management necessary on the DevOps end.[1]

A solution for github is called Travis[2], providing a public building service through a simple bash script. This build server is used as an authority, running unit tests and giving advice on commits and merge requests. It does not store artifacts.

For the artifact and deployment management we have looked at Gitlab CI/CD. Here, a daily mirror is made of the project and artifacts will be built. In the future, Gitlab will also be in charge of managing deployments.

[1] Using public services does mean that any confidential information will no longer be confidential. It is htus advised that for these issues a private setup is required

[2] https://travis-ci.org/