Continuous Integration

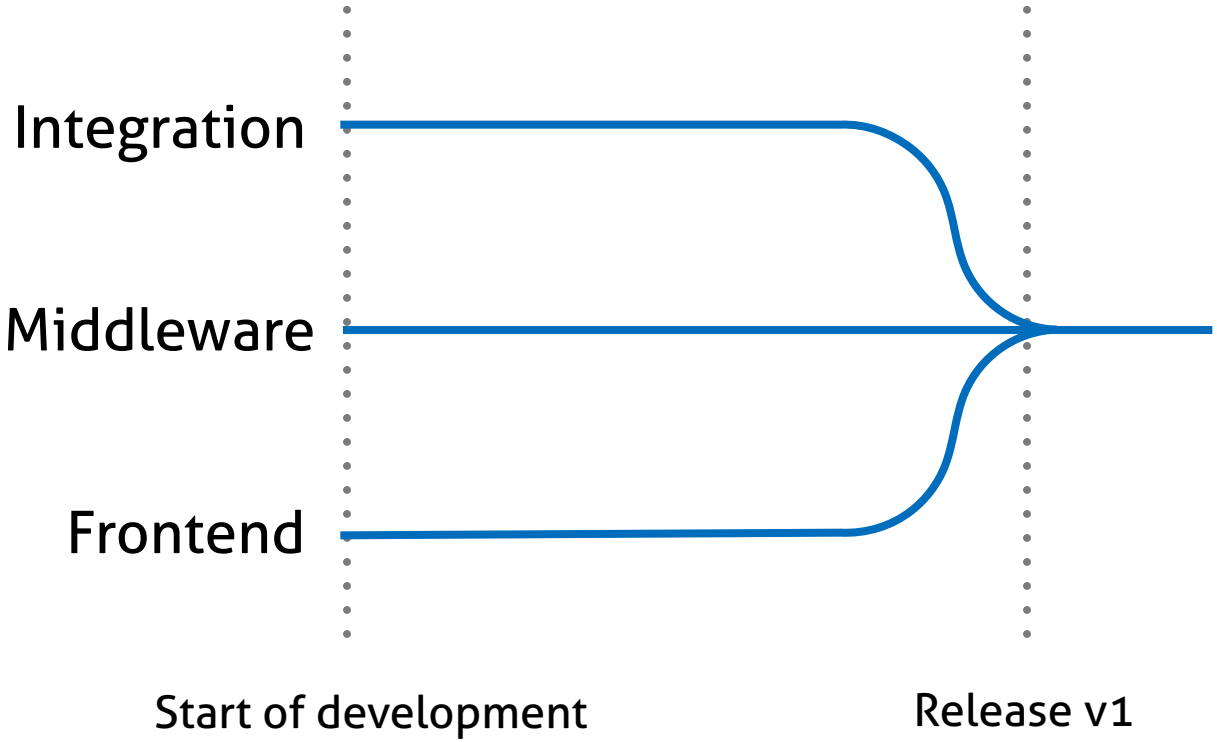
Stefan Sprenger (sprengsz@informatik.hu-berlin.de)
Semesterprojekt “Verteilte Echtzeitrecherche in Genomdaten”
15. Dezember 2015

Motivation

How was software developed before CI?

- Software development is done in teams
- Divide software in subparts, assign subparts to teams
- Work of multiple teams/persons has to be integrated
- Typically, integration is done at release time
- Typically, integration is done manually

How was software developed before CI?



Bug fixes?

New releases?

What's the issue?

- Bad software quality
- Integration tests can only be executed before a release
- Frustration rises towards the end of a project
- Manual (re-)execution of tasks
- No feedback possible before integration (release)

Obviously, we need to integrate more often and earlier.

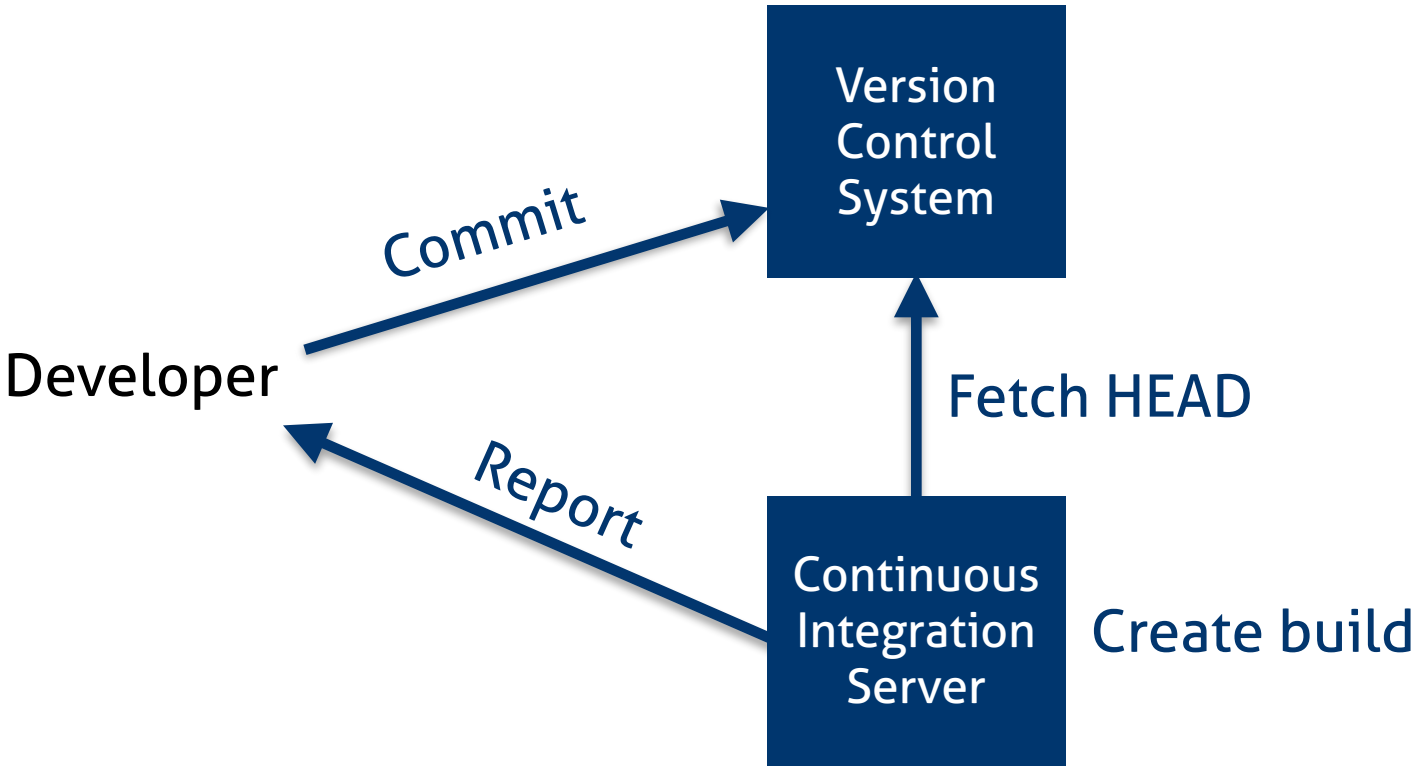
Continuous Integration

is a philosophy,
but not a tool.

Continuous Integration (CI)

- Automatic integration of a software project
- Every change in the software triggers a new build
- In a perfect world, the software's tests are executed to determine the success of a build
- Gives early feedback in form of reports
- A build can be successful or fail

CI in practice



How is a build created?

- Download dependencies
- Compile code
- Run tests
- Create build artifacts
- Create reports

A build can be successful or fail.

How does CI help us?

- We always know the latest stable version of our software
- We know if and which bugs currently exist
- We detect bugs earlier
- We can automatically test different setups
  - different databases
  - multiple versions of 3rd party libraries
  - different configurations

Good practices

- Always write tests for your software (unit, integration, ..)
- Commit frequently
- Small iterations
- Keep the build fast (keep your tests fast)
- Don’t commit when the build is broken
- Build system should be identical to production system
- Use build system for deployment

Software tests

Software tests

- No manual testing by {developer, manager, customer}
- Automatic testing using a test framework
- Test framework provides tools to define tests
- Usually, tests are defined in the same programming language as the tested software
- Tests check if the software meets a certain requirement
- Tests can be executed

Tests can be successful or fail.

Unit tests

- Test a certain unit of a software
- A unit may be a class in OOP
- The unit is tested isolated
- Interaction between different units is not tested
- Test cases are independent from each other
- Unit tests are written by software developers

How may a unit test look like?

```
class MailValidator {
  public boolean check(String mailAddress) {
     Pattern pattern    = Pattern.compile("[A-Z0-9._%+-]+@[A-Z0-9.-]+\\.[A-Z]{2,4}");
     Matcher matcher = pattern.matcher(mailAddress);

     return matcher.matches();
   }
}
```

How may a unit test look like?

```
import static org.junit.Assert.assertEquals;
import org.junit.Test;

class TestMailValidator {
   public void testValidMail() {
      MailValidator validator = new MailValidator();
      assert(validator.check("sprengsz@informatik.hu-berlin.de") == true);
   }
}
```

How may a unit test look like?

```
class TestMailValidator {
   public void testValidMail() {
      MailValidator validator = new MailValidator();
      assert(validator.check("sprengsz@informatik.hu-berlin.de") == true);
   }
}
```

How may a unit test look like?

```
class TestMailValidator {
   public void testValidMail() {
      MailValidator validator = new MailValidator();
      assert(validator.check("sprengsz@informatik.hu-berlin.de") == true);
   }

   public void testInvalidMail() {
      MailValidator validator = new MailValidator();
      assert(validator.check("sprengsz@informatik") == false);
   }
}
```

Integration tests

- Test multiple units combined
- Test interaction between units
- Ensures that integration of multiple subparts works
- Often done using same framework as for unit tests

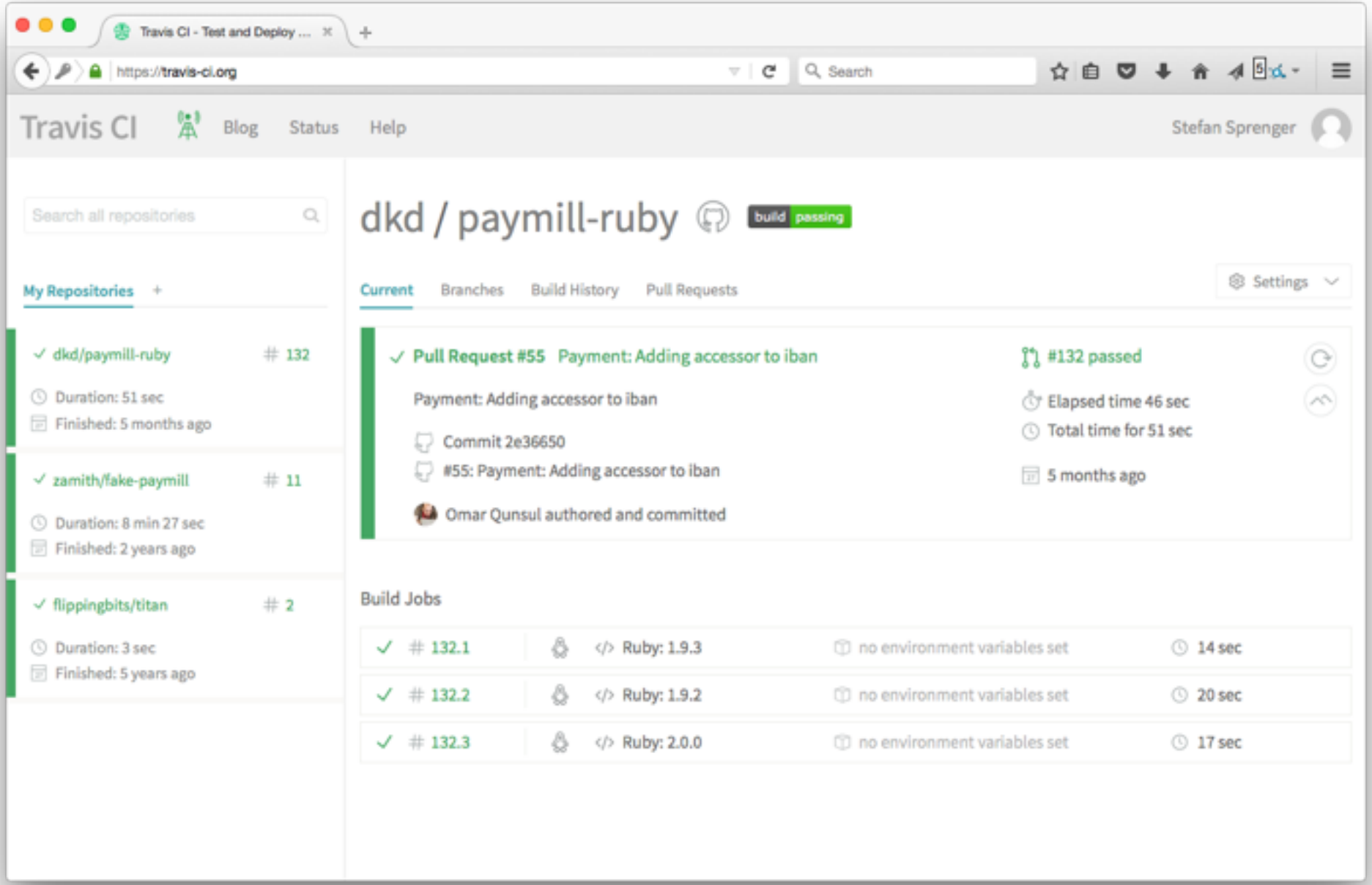
Tools

Travis CI https://travis-ci.org

- Open-source distributed build service
- Coupled to GitHub
- Setup:
  - 1) Sign in using your GitHub account
  - 2) Select repositories that Travis should build
- Build is configurable using a .travis.yml file
- Heavily used in the OSS community

Example .travis.yml

```
rvm:
  - 1.8
  - 1.9
env:
  -DB=mongodb
  -DB=redis
  -DB=mysql
before_script:
  - "mysql -e 'create database vanity_test;' > /dev/null"
```


Travis CI - Test and Deploy ...
https://travis-ci.org
Search
Travis CI
Blog
Status
Help
Stefan Sprenger
Search all repositories
My Repositories
dkd/paymill-ruby
132
Duration: 51 sec
Finished: 5 months ago
zamith/fake-paymill
11
Duration: 8 min 27 sec
Finished: 2 years ago
flippingbits/titan
2
Duration: 3 sec
Finished: 5 years ago
dkd / paymill-ruby
build passing
Current
Branches
Build History
Pull Requests
Settings
Pull Request #55 Payment: Adding accessor to iban
Payment: Adding accessor to iban
Commit 2e36650
#55: Payment: Adding accessor to iban
Omar Qunsul authored and committed
#132 passed
Elapsed time 46 sec
Total time for 51 sec
5 months ago
Build Jobs
132.1
Ruby: 1.9.3
no environment variables set
14 sec
132.2
Ruby: 1.9.2
no environment variables set
20 sec
132.3
Ruby: 2.0.0
no environment variables set
17 sec

Jenkins CI https://jenkins-ci.org/

- Open-source build system
- Is provided as Java application
- Can be hosted in your infrastructure
- More flexible than Travis CI
- Lots of plugins available

Questions?