Introduction to Continuous Delivery

Giovanni Toraldo & Tazio Ceri

Once upon a time

there was project management

-15 days to the release

Mr. Project Manager:«Hello mr. Tester, I need you to test everything again in two weeks. We need a release for the X certification process.

Mr. Tester:«Yes, sir.»

Two weeks of pain (1)

- The tester will have to manually test a huge number of new features...
- and old ones to avoid regressions.
- Developers will fix bugs as soon as the tester reports them, then he should have to start all tests from scratch to the new release candidate - but he won’t!

Two weeks of pain (2)

- The tester is on pressure, so he will work hard. His error rate will increase.
- The tester will have to make assumptions about the configuration that will be used for the certification process.
- The same for developers.
- In the shuffling madness of this project, everybody feels like the all-time loser, headlong to the project’s death.

Release!

What went wrong? (1)

- That system is too complex to work with!
- Software was never actually release ready.
- During these two weeks, the tester has filed several bug reports. Developers fixed bugs and made another release candidate, so the tester has actually tested the final release for few hours...

What went wrong?(2)

- Developers did not test very well their modification because installing the software was too hard and slow.
- So they always were working in an environment totally different from production.
- And anyway, they had no discipline to take track of what tests needed to do, just a fast smoke test before committing to SVN.

What went wrong? (3)

- The changelog was actually misleading the tester, making him test new features disregard regressions.
- So, developers were busy fixing bugs as they came out **during the certification phase!**
- After that, everybody enjoyed the art of avoiding responsability of failures.

It will be wrong again!

- After that, everybody enjoyed the art of avoiding responsibilities of failures.
- Some agreed it was the consultant’s fault. Yes, the one who left one year before.
- Some scheduled to work 12 hours everyday to fix everything they found before next certification phase.
- None of them understood the situation.

Problems to solve

- Complex installation!
- Complex development!
- Complex testing!
- **If you have to do many complex, error prone and repetitive task manually you are in trouble!**
- Continuous delivery is the solution.

Automate everything

- If something is hard and boring to do, do it more often. **Until you will automate it.**
- No exception to this rule!
- IT is about automation. You help other persons to automate their work and to increase productivity, but you don’t do that for yourself, do you?
- That does not sound very smart!

Automation needs care and control

- It’s funnier to write tests than test manually!
- More code to be written… but they will speed up maintenance.
- Nothing can’t be really tested: programs are useful as they produce output, not for their hidden status.
- Don’t forget to automate installation and database management.

The road to enlightenment

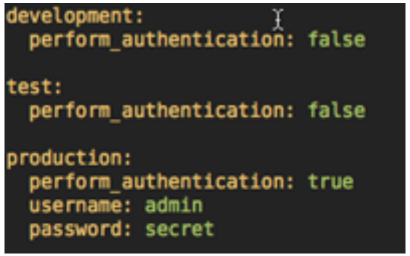
- Configuration Management
- Data management
- Continuous Integration
- Continuous Deployment

Configuration Management

- Keep *everything* in VCS.
- Managed software dependencies
- Your application should be configurable
- Avoid branches other than master

git

maven



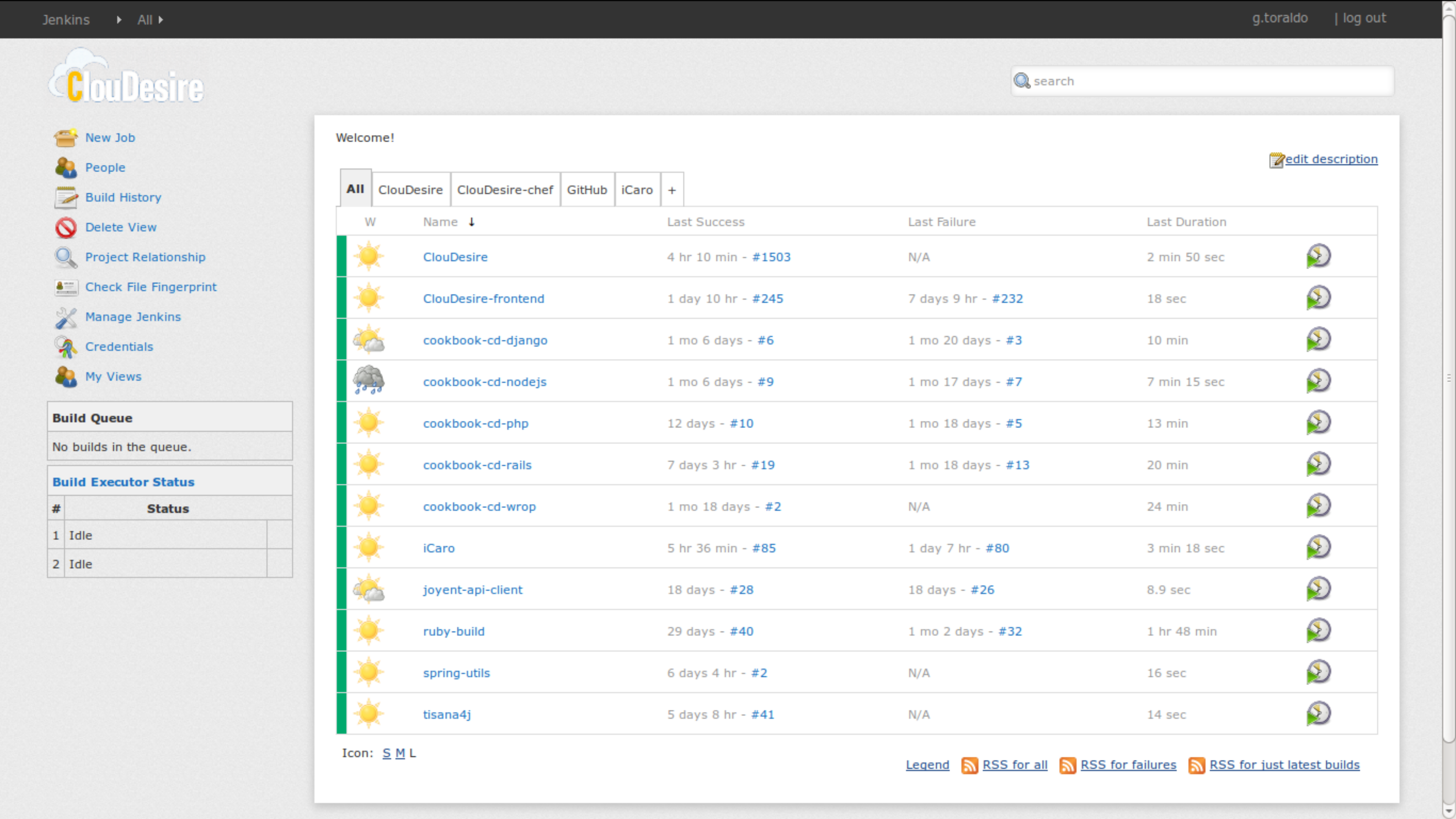
```
development:
  perform_authentication: false

test:
  perform_authentication: false

production:
  perform_authentication: true
  username: admin
  password: secret
```

Data management

- Scripted schema migrations
- Dataset for development
- Be backward and forward compatible (when possibile)

Continuous Integration

- Requirements:
  - Automated test and build tool
- Features:
  - Every push trigger a build
  - Feedbacks arrives in minutes
  - A broken build needs priority to be fixed
  - One does not simply @Ignore a failing test
  - Static code analysis (smells, duplicated lines, ...)
  - Trendy Graphs!



Jenkins ▸ All ▸ g.toraldo | log out

search

- New Job
- People
- Build History
- Delete View
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- Credentials
- My Views

Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Welcome!

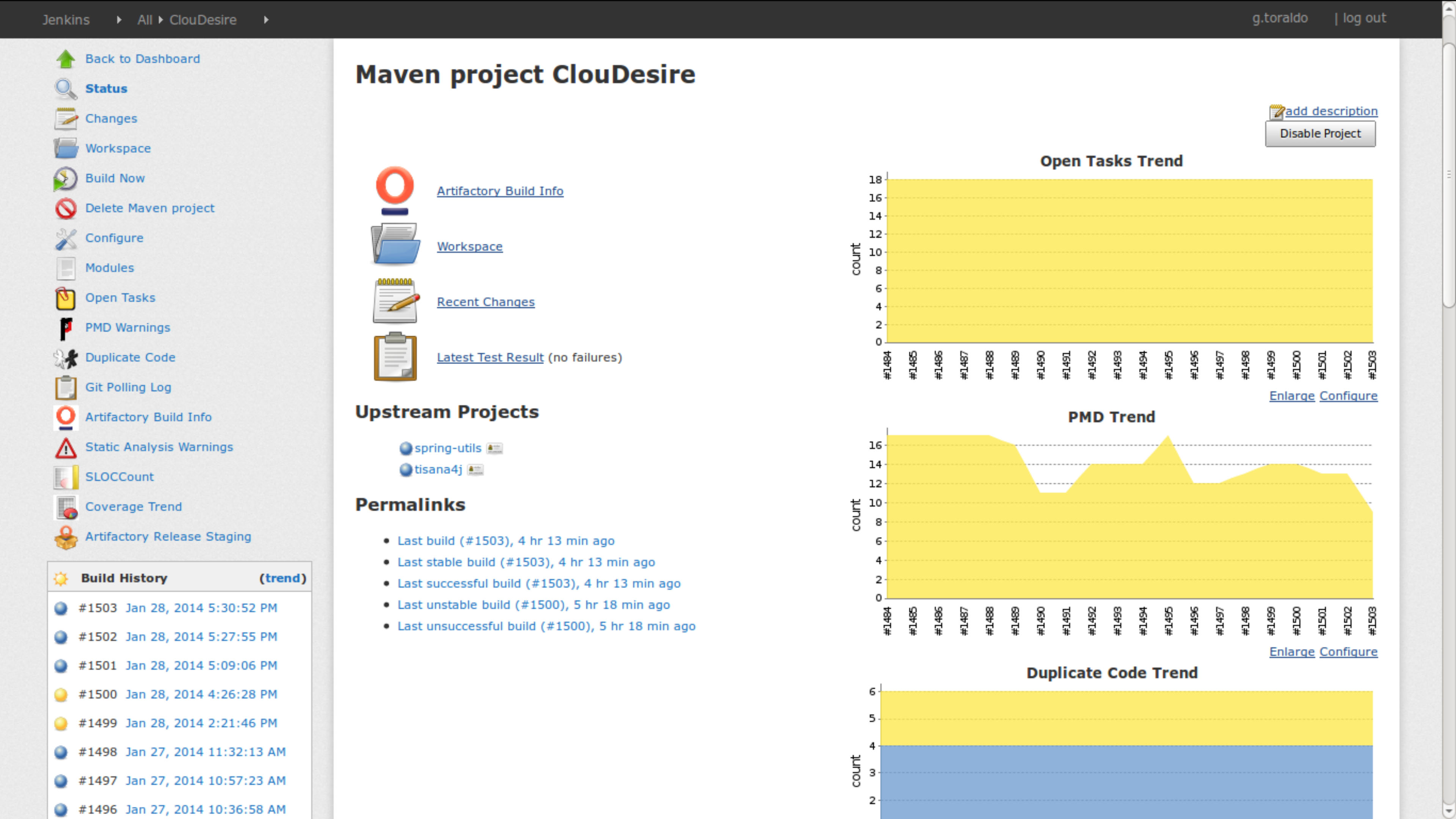
edit description

All | ClouDesire | ClouDesire-chef | GitHub | iCaro | +

W	Name ↓	Last Success	Last Failure	Last Duration
☀	ClouDesire	4 hr 10 min - #1503	N/A	2 min 50 sec
☀	ClouDesire-frontend	1 day 10 hr - #245	7 days 9 hr - #232	18 sec
⛅	cookbook-cd-django	1 mo 6 days - #6	1 mo 20 days - #3	10 min
🌧	cookbook-cd-nodejs	1 mo 6 days - #9	1 mo 17 days - #7	7 min 15 sec
☀	cookbook-cd-php	12 days - #10	1 mo 18 days - #5	13 min
☀	cookbook-cd-rails	7 days 3 hr - #19	1 mo 18 days - #13	20 min
☀	cookbook-cd-wrop	1 mo 18 days - #2	N/A	24 min
☀	iCaro	5 hr 36 min - #85	1 day 7 hr - #80	3 min 18 sec
⛅	joyent-api-client	18 days - #28	18 days - #26	8.9 sec
☀	ruby-build	29 days - #40	1 mo 2 days - #32	1 hr 48 min
☀	spring-utils	6 days 4 hr - #2	N/A	16 sec
☀	tisana4j	5 days 8 hr - #41	N/A	14 sec

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds



Jenkins ▸ All ▸ ClouDesire ▸ g.toraldo | log out

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now
- Delete Maven project
- Configure
- Modules
- Open Tasks
- PMD Warnings
- Duplicate Code
- Git Polling Log
- Artifactory Build Info
- Static Analysis Warnings
- SLOCCount
- Coverage Trend
- Artifactory Release Staging

Build History (trend)

#1503	Jan 28, 2014 5:30:52 PM
#1502	Jan 28, 2014 5:27:55 PM
#1501	Jan 28, 2014 5:09:06 PM
#1500	Jan 28, 2014 4:26:28 PM
#1499	Jan 28, 2014 2:21:46 PM
#1498	Jan 27, 2014 11:32:13 AM
#1497	Jan 27, 2014 10:57:23 AM
#1496	Jan 27, 2014 10:36:58 AM

Maven project ClouDesire

add description

Disable Project

Artifactory Build Info

Workspace

Recent Changes

Latest Test Result (no failures)

Upstream Projects

spring-utils

tisana4j

Permalinks

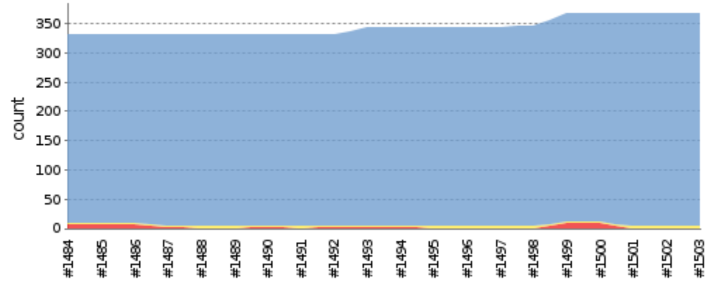
- Last build (#1503), 4 hr 13 min ago
- Last stable build (#1503), 4 hr 13 min ago
- Last successful build (#1503), 4 hr 13 min ago
- Last unstable build (#1500), 5 hr 18 min ago
- Last unsuccessful build (#1500), 5 hr 18 min ago

●	#1492	Jan 24, 2014 6:13:53 PM
●	#1491	Jan 24, 2014 3:50:17 PM
●	#1490	Jan 24, 2014 3:23:42 PM
●	#1489	Jan 24, 2014 3:10:47 PM
●	#1488	Jan 24, 2014 2:43:42 PM
●	#1487	Jan 24, 2014 2:28:32 PM
●	#1486	Jan 24, 2014 2:17:12 PM
●	#1485	Jan 24, 2014 2:04:47 PM
●	#1484	Jan 24, 2014 11:54:57 AM

RSS for all RSS for failures

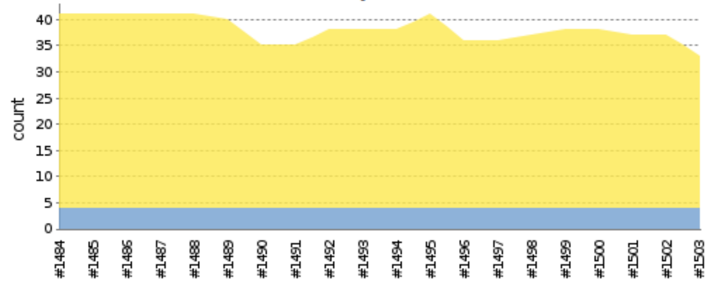
Enlarge Configure

Test Result Trend



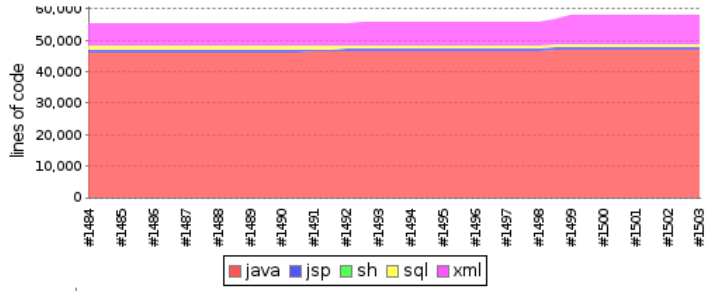
(just show failures) enlarge

Static Analysis Trend



Enlarge Configure

SLOCCount Trend



CI: general strategies

- Use dependency-injection (IoC)... better with a framework. Otherwise unit testing can be impossible.
- Component tests with database access.
- Integration tests againt *real* components.
- Everything must run in less than 8 minutes.
- Use a database configuration optimized for tests.

CI: unit tests

- Code coverage must be high, especially with dynamic languages.
- Test slices of code in isolation.
- Use mocking or stubs to simulate external interaction.
- Avoid the database!
- Must be very fast.

CI: component test

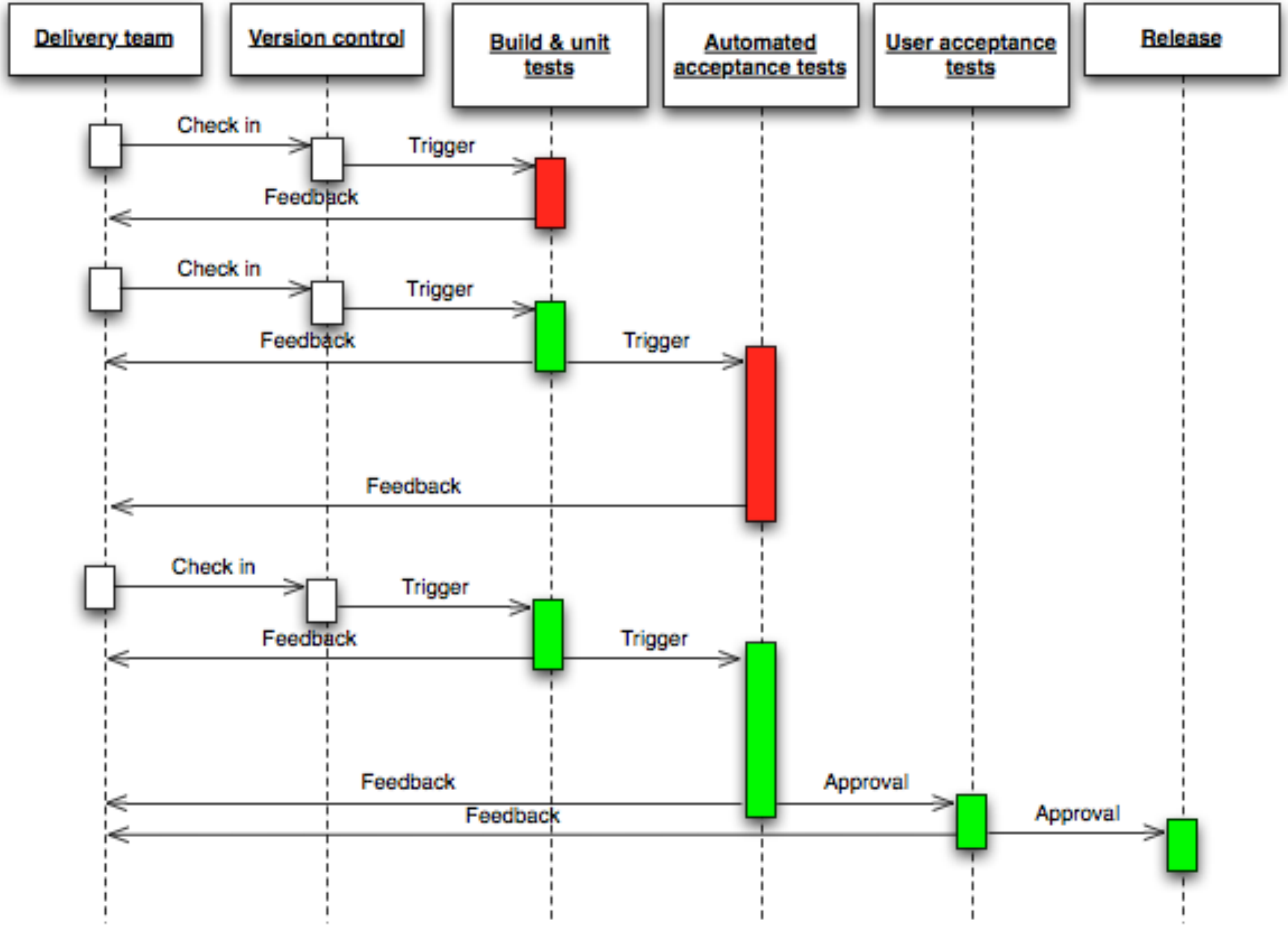
- ORM or SQL? You must test database access anyway. It’s where most bugs live!
- Use a meaningful small subset of your production database for each test.
- Isolate database between each test, transaction can be useful. (DBUnit)

CI: integration tests

- Whole modules of your system should interact together.
- Database and network access are a must.
- Slower and harder to write.
- A reason to keep interfaces between modules simple and extensible, and generally better.

Continuous Deployment

- Tag release and artifact build
- Push-the-button procedures for envs setup:
  - Development workstation
  - Staging/Acceptance
  - Production
- Fast rolling-back strategies
- Zero-downtime releases
- Infrastructure deserve tests too.



Deployment Pipeline

Questions?

Thanks!

Credits

The holy book:

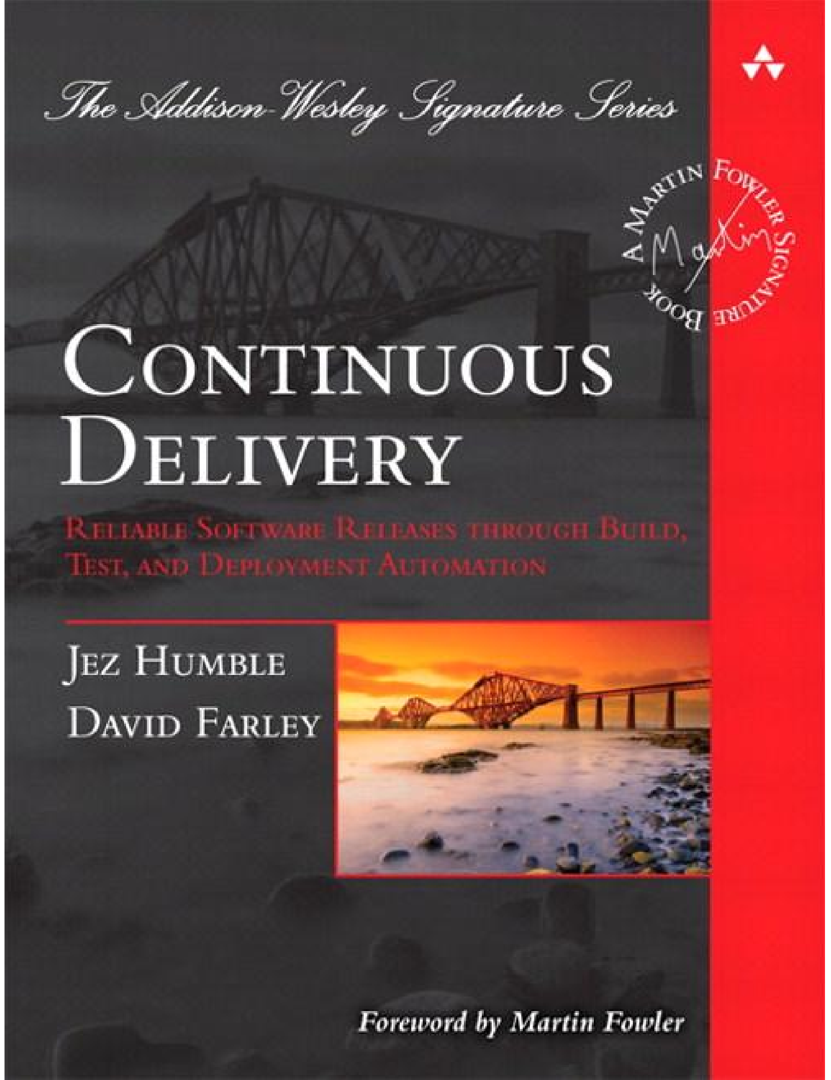
- Jez Humble & David Farley. Continuous delivery : reliable software releases through build, test, and deployment automation. ISBN 978-0-321-60191-9.

Images:

- http://en.wikipedia.org/wiki/File:Continuous_Delivery_process_diagram.png

GIFs:

- http://devopsreactions.tumblr.com
- http://thecodinglove.com



The Addison-Wesley Signature Series

A Martin Fowler Signature Book

CONTINUOUS DELIVERY

RELIABLE SOFTWARE RELEASES THROUGH BUILD, TEST, AND DEPLOYMENT AUTOMATION

JEZ HUMBLE

DAVID FARLEY

Foreword by Martin Fowler