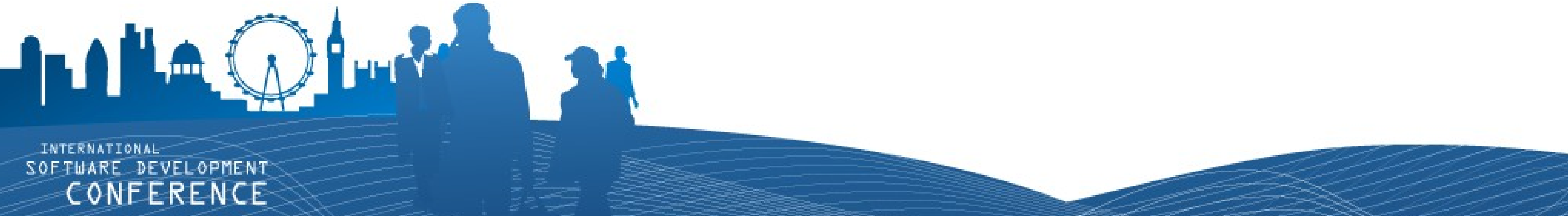
QCon

Please evaluate my talk via the mobile app!



The mean and lean pipeline

From code to production as fast and safe as possible

QCon London 2014

TRADESHIFT®

Who, what, and why?

Joakim Recht

Senior Code Monkey at Tradeshift – the platform for all your business interactions

“- Please don't do that”

TRADESHIFT®

The perfect build pipeline

Prevents bad code from entering production systems

Gives fast feedback

Ensures consistency

Quality becomes absolute

Is fully automated

Creates nice screens and buttons to click

You get to talk about it on conferences

Our build pipeline

Gives feedback

Prevents most bad code from entering production systems

Ensures some consistency

Increases quality

Is pretty automated

There are screens and buttons to click

You get to talk about it on conferences

Build pipeline stats (mid 2012+)

- Servers in Jenkins CI env: 28-52
- Backend builds: 18649
- Frontend builds: 10822
- PRs on Bob: 652
- Integration test subset runs: 61586
- Github PRs: 4781
- Number of integration test specs: 248
- Number of war files deployed during prod deploy: 20

The realities of a startup

- Not enough time
- Not enough people
- Not enough money
- Too many (crazy) ideas
- Too many requirements

TRADESHIFT®

2010: The beginning

The simple life

Java backend, Drupal frontend, REST API.

All code in Subversion. No branches.

Everything hosted on Amazon EC2.

Hudson CI to run tests. Green build policy.

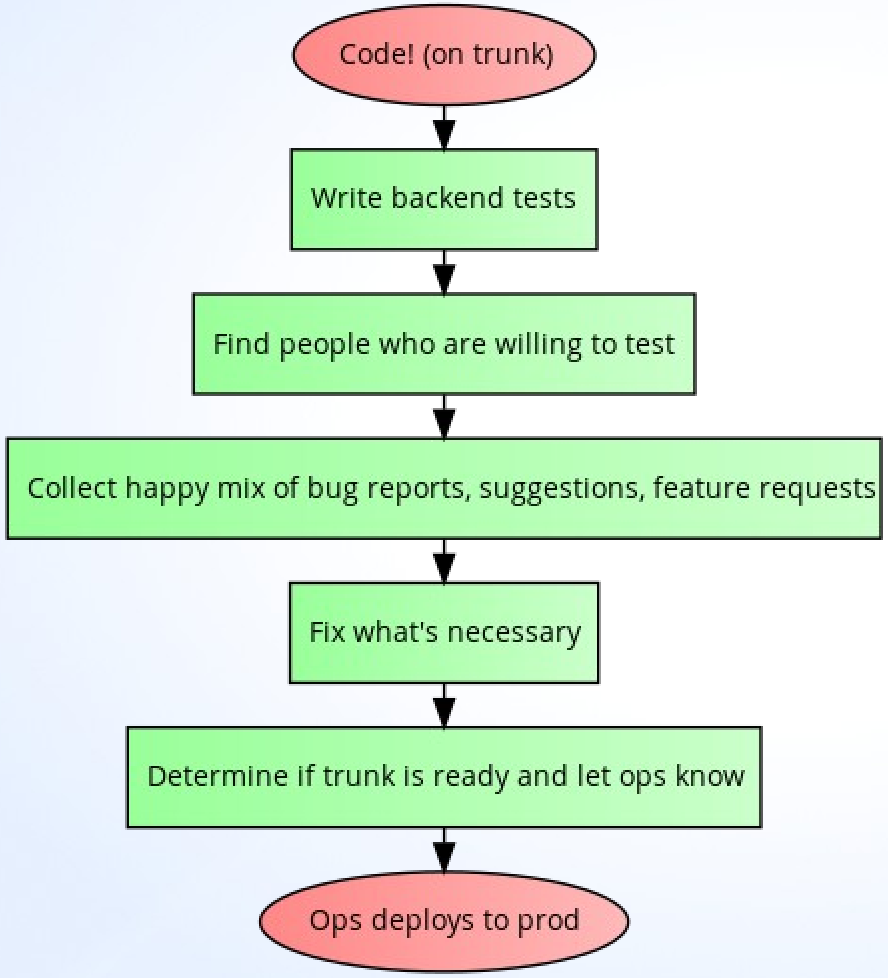
Tests written in jUnit, BDD style.

Deployment to test server: manual ssh, download war, restart Jetty.

Deployment to prod: semi-automated using custom scripts and AMI building.

TRADESHIFT®

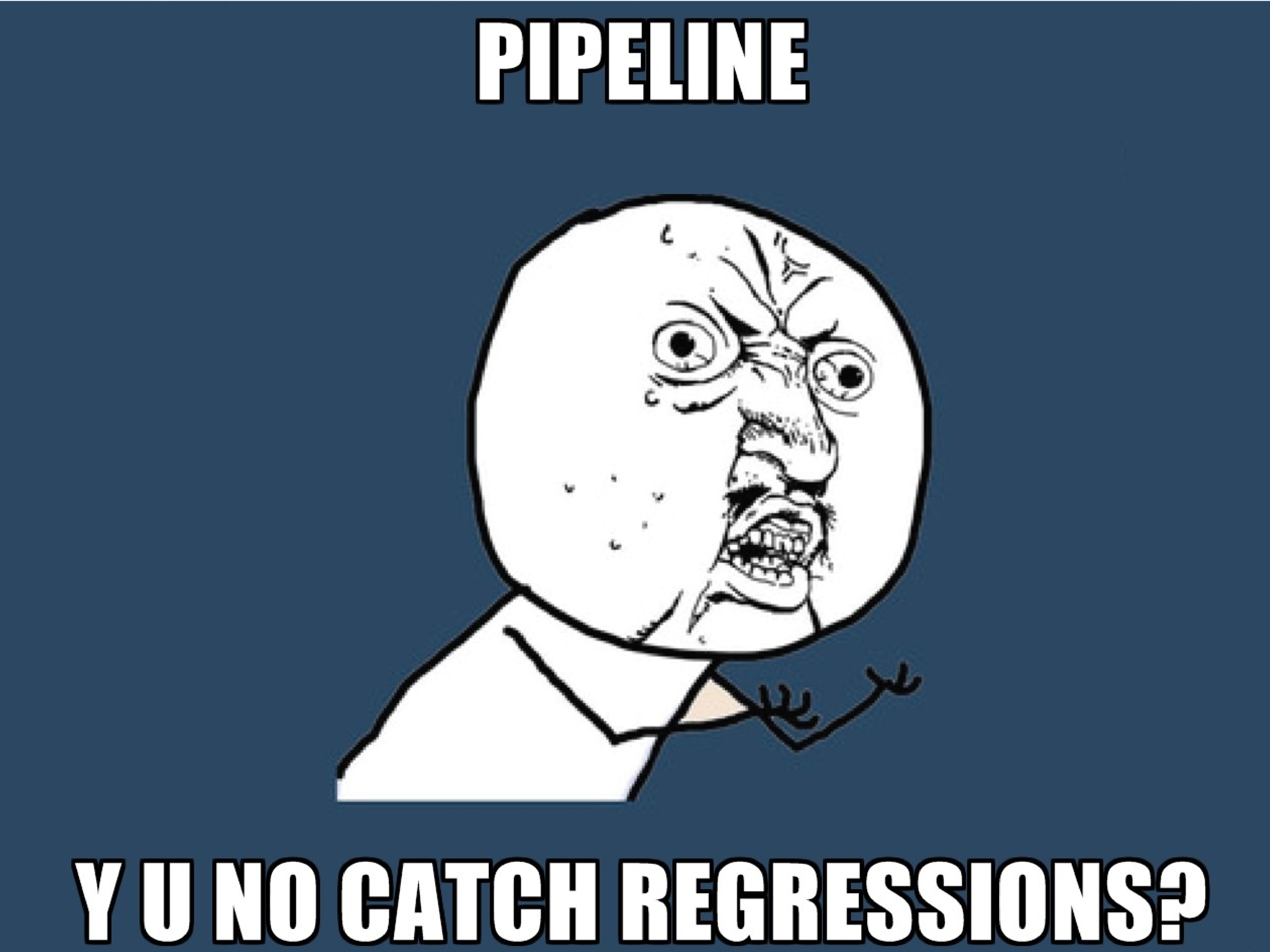
Getting a feature into prod


Code! (on trunk)
Write backend tests
Find people who are willing to test
Collect happy mix of bug reports, suggestions, feature requests
Fix what's necessary
Determine if trunk is ready and let ops know
Ops deploys to prod

Switching from SVN to Git

- Just working off trunk is easy, but release management is hard
- SVN branches (and merging) sucks
- Switched to Git in mid 2010
  - Offline support, team/personal branches, cherry-picking, history rewriting
  - No formal process yet, other than the introduction of a production branch. Manual merging all around. Based on social contract.

TRADESHIFT®


PIPELINE
Y U NO CATCH REGRESSIONS?

Selenium FTW. Or not.

- Manual testing is a pain
- There were no testers employed
- Regressions, esp. in UI, happened all the time
- Tried out Selenium UI tests
  - Manual maintenance
  - Not part of normal development process
  - Tests too fragile
- Selenium did not help.

TRADESHIFT®

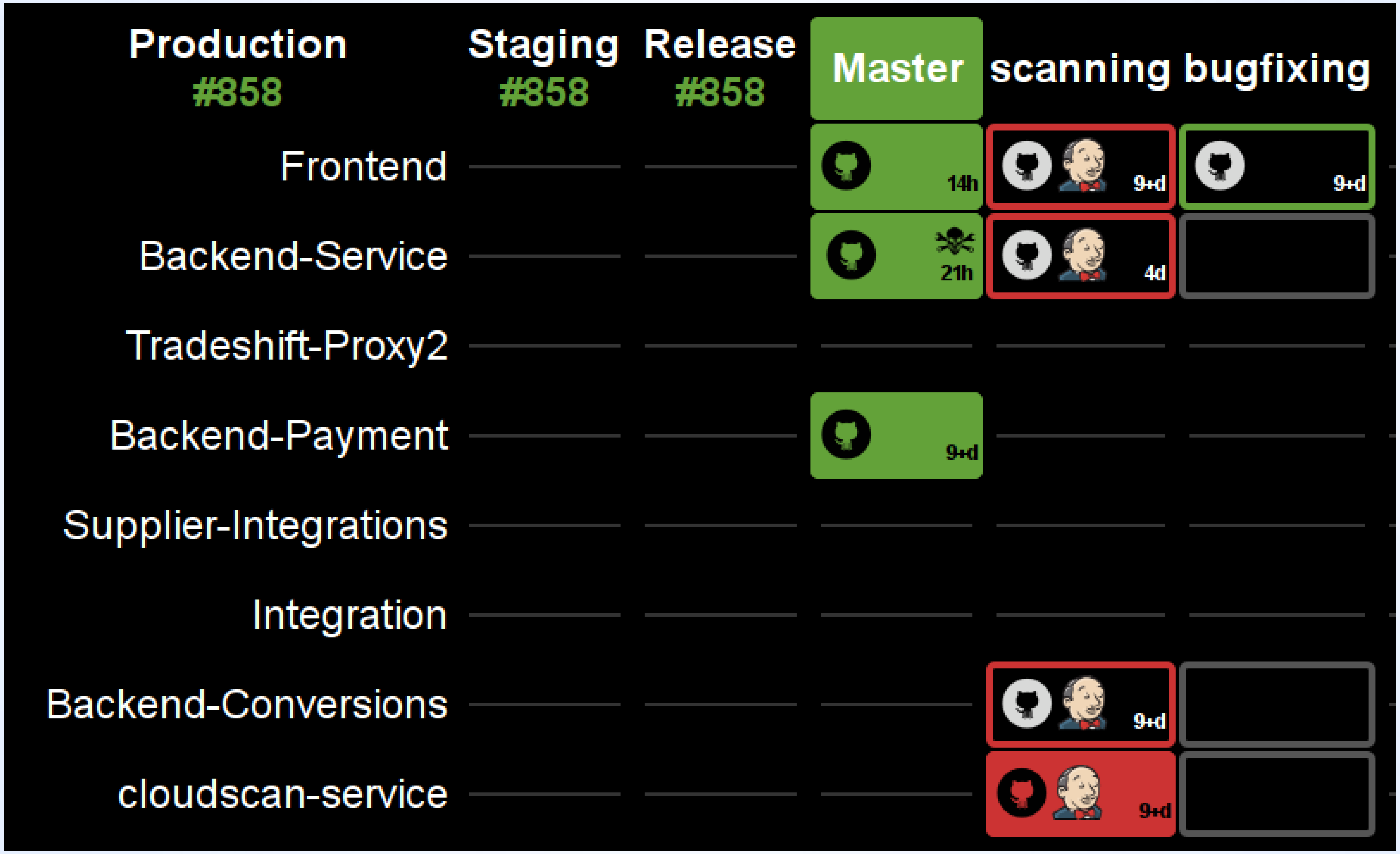
UI tests, take 2

Geb and Spock for UI tests

- Geb: Groovy framework on top of Webdriver (Selenium 2)
- Spock: Groovy-based BDD framework
- Writing tests become part of the development process
- Tests executed on the only physical server in order to show the runs on a big screen
- UI tests must be green

Introducing pipeline visualization

- By mid 2011 the number of teams had grown
- As had the number of components
- Keeping track of build status on all the branches was getting increasingly hard



	Production #858	Staging #858	Release #858	Master	scanning	bugfixing
Frontend				🟩 14h	🟥 9+d	🟩 9+d
Backend-Service				🟩 21h	🟥 4d	
Tradeshift-Proxy2						
Backend-Payment				🟩 9+d		
Supplier-Integrations						
Integration						
Backend-Conversions					🟥 9+d	
cloudscan-service					🟥 9+d	

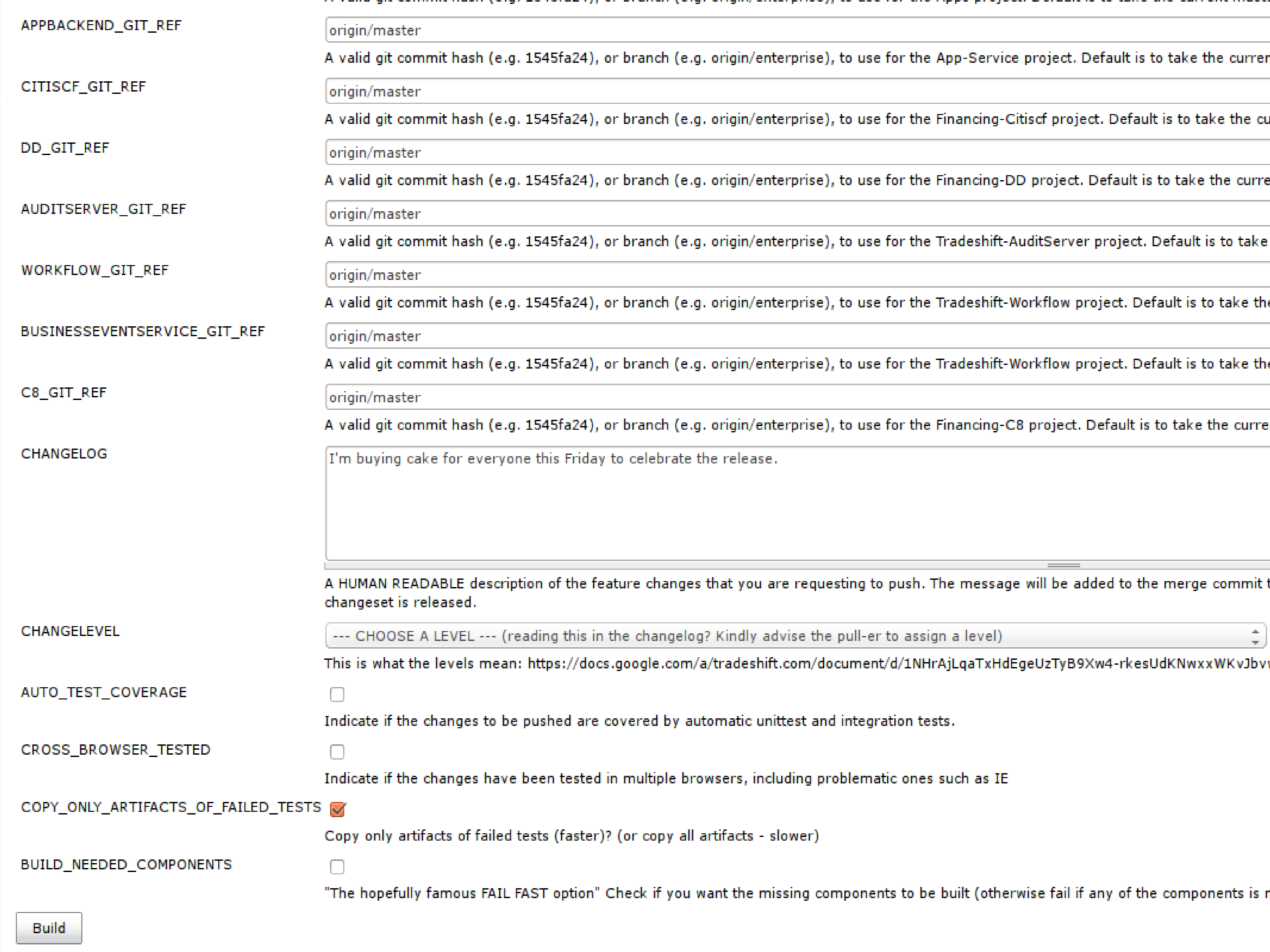
Starting a new office in SF

“For the n'th time since their arrival they are fixing broken stuff in master that was not committed by themselves. For the team that has no contact with the owner of the merged code for 7 hours or more, that could mean half a day of troubleshooting, missed sprint targets, convulsions and so on.” - Gert Sylvest, CTO

TRADESHIFT®

The Pull Request

- No manual pushes to master
- Anything going into master must be tested as a complete configuration
- All components must be green, all integration tests must run
- Ensures proper consistency
- Implemented by Jenkins jobs
  - Not Gerrit. Seemed too complicated.
- Physical build machine for Geb abandoned and replaced with Jenkins swarm on EC2



A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Apps project. Default is to take the current mast

APPBACKEND_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the App-Service project. Default is to take the curren
CITISCF_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Financing-Citiscf project. Default is to take the cu
DD_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Financing-DD project. Default is to take the curre
AUDITSERVER_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Tradeshift-AuditServer project. Default is to take
WORKFLOW_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Tradeshift-Workflow project. Default is to take the
BUSINESSEVENTSERVICE_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Tradeshift-Workflow project. Default is to take the
C8_GIT_REF	origin/master
	A valid git commit hash (e.g. 1545fa24), or branch (e.g. origin/enterprise), to use for the Financing-C8 project. Default is to take the curre
CHANGELOG	I'm buying cake for everyone this Friday to celebrate the release.
	A HUMAN READABLE description of the feature changes that you are requesting to push. The message will be added to the merge commit t changeset is released.
CHANGELEVEL	--- CHOOSE A LEVEL --- (reading this in the changelog? Kindly advise the pull-er to assign a level)
	This is what the levels mean: https://docs.google.com/a/tradeshift.com/document/d/1NHrAjLqaTxHdEgeUzTyB9Xw4-rkesUdKNwxxWKvJbv
AUTO_TEST_COVERAGE	☐
	Indicate if the changes to be pushed are covered by automatic unittest and integration tests.
CROSS_BROWSER_TESTED	☐
	Indicate if the changes have been tested in multiple browsers, including problematic ones such as IE
COPY_ONLY_ARTIFACTS_OF_FAILED_TESTS	☑
	Copy only artifacts of failed tests (faster)? (or copy all artifacts - slower)
BUILD_NEEDED_COMPONENTS	☐
	"The hopefully famous FAIL FAST option" Check if you want the missing components to be built (otherwise fail if any of the components is r

Build

YEAH I COMMENTED MY CODE
"UNTESTED. USE WITH CAUTION"
memegenerator.net

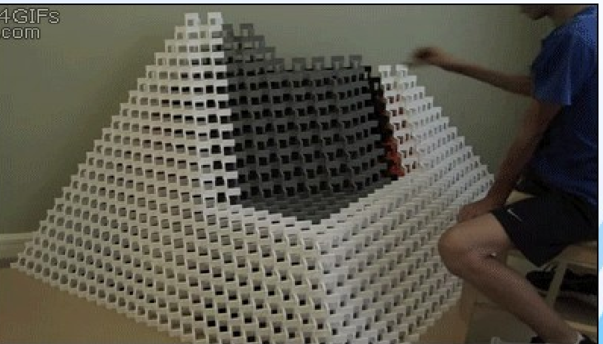
Developers, developers, developers

- Approaching 40 developers by 2013
- Keeping consistency and quality getting hard
- Knowledge sharing equally hard
- No written procedures or guides
  - Also, nobody wanted to write any or maintain them

Solution: Code reviews using Github PRs

- All code must be reviewed by at least one other dev
- DB migrations to be reviews by select group

More automation



Removing ops as bottleneck

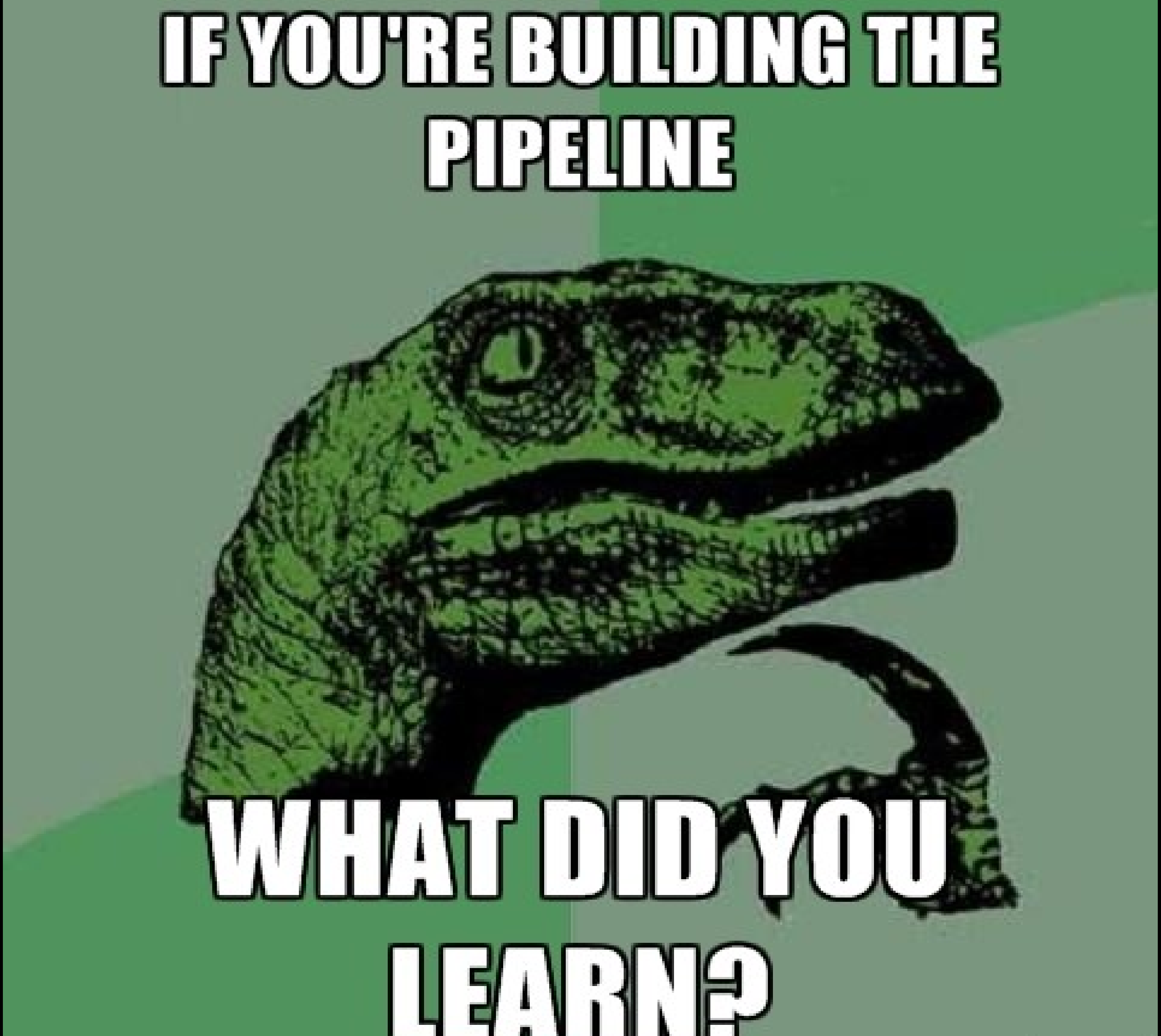
- In 2011 all environments (except local dev) were changed to use Puppet
- Release procedure a matter of starting a job on Jenkins (only available to Ops)
- 2013: Any team can deploy any configuration to any sandbox env by the click of a button
- Thread dumps for running env can be generated from Jenkins
- Sandbox availability times can be controlled

Recent issues

TRADESHIFT®

Regular annoyances

- Anybody can still push to master – Github cannot prevent this
  - Add comment to PR automatically if opened against master
- Build times
  - At the extremes: 40 minutes for backend, 45 minutes for integration tests
  - Cut down to 15 minutes each by optimizing tests and scaling out – see http://blog.tradeshift.com/just-add-servers/
  - Backend tests to be shortened more by splitting into other components
- Randomly failing tests
  - Esp. integration tests
  - Zero tolerance initiated


IF YOU'RE BUILDING THE PIPELINE
WHAT DID YOU LEARN?

Pivotal events

- Too many teams working concurrently on new features
- Too many regressions and bugs introduced into production
- Too many components to deploy
- Too many offices and timezones
- Pipeline throughput not sufficient
- SLAs being violated due to downtime

- Automation is king
  - But also quite expensive
- Don't be religious
  - But make sure to have an extensive test suite
    - That cannot be too slow
    - And with understandable tests

TRADESHIFT®

What we probably should have done earlier

- Use Git instead of Subversion
- Perform code reviews for all changes

- Using Geb/Spock would have been nice, but not stable enough at the time

- All of the above have had a significantly higher impact than anticipated

What we still don't have

- Full automation – also for dev envs
- Explicit code styles – just not painful enough without
- Naming conventions
- Framework versioning or policies
- Agreement on unit vs system vs integration vs mock vs UI tests
- Consistent use of Findbugs/Checkstyle/similar
- Testers

TRADESHIFT®

Thank you
Questions?

jre@tradeshift.com

@joakimrecht

https://plus.google.com/+JoakimRecht

http://tradeshift.com/blog/

TRADESHIFT®

QCon

Please evaluate my talk via the mobile app!