Releasing the Monolith
On a daily basis

VINCENT KOK | ENGINEERING MANAGER, TRELLO | @VINCENTKOK
About me

Dutch
You’ve probably heard that already ;)

Trello
Engineering Manager on the Trello team

@vincentkok
Cool stories
Journey
Monolith
Agenda

- Confluence pipeline
- Culture
- Microservice envy
- Four problems
Confluence pipeline
15,000 builds per week
81,000
CI jobs
31,992 automated tests
1,000
build agents
3
size of the Pipeline team
72 steps to release Confluence :(
culture

/ˈkʌltrə/

noun
noun: culture

1. the arts and other manifestations of human intellectual achievement regarded collectively.
   "20th century popular culture"
synonyms: the arts, the humanities, intellectual achievement;  More
   • a refined understanding or appreciation of this.
     "men of culture"
synonyms: intellectual/artistic awareness, education, cultivation, enlightenment, discernment, discrimination, good taste, taste, refinement, polish, sophistication
     "a man of culture"

   • the customs, arts, social institutions, and achievements of a particular nation, people, or other social group.
     plural noun: cultures
     "Caribbean culture"
synonyms: civilization, society, way of life, lifestyle;  More

   • the attitudes and behavior characteristic of a particular social group.
     "the emerging drug culture"
Experience
The way you’ve been shaped

Beliefs
Based on your experience

Actions
Based on your beliefs

Results
The outcome feeding your experience
Theory in real life

- Clear goal
- Early results
- Momentum
Good intentions don’t work

ASANKA JAYASURIYA, HEAD OF ENGINEERING STRIDE
GreenDex
Mechanisms

- **Process**: What process can improve the situation?
- **Adoption**: Drive adoption by i.e. training people
- **Audit**: Does it move the needle?
Microservice envy
Microservice traits

- Small
- Clear ownership
- Independent lifecycle
Monolith
< 1 hour
< 1 day
Four problem areas

- Release cadence
- Ownership
- Test automation
- Complexity
#1: Complexity
A TYPICAL PIPELINE

Inner

Outer

Release

Publish

Deploy
SOME FLAWS

- Inner
- Outer
- Release

Staging deploy
- Publish
- Deploy

Manual release branch
Impact of complexity

Manual tasks  Hard to understand  Inconsistent feedback
Fix the model
FORK & JOIN

Inner

Outer

Release

Fork

Join

Publish

Deploy
FORK & JOIN

Inner

Outer

Release

Fork

Join

Publish

Deploy
Iterate

- Inner loop
- Build times
- Build stability
#2: Ownership

_Individually

Ownership

Everyone acts individually.
Getting the model wrong
Getting the model wrong
Impact

- Fire fighting
- Broken windows
- No improvements
Even more impact

Dev speed  Feedback loop  MTTR
OWNERSHIP

Pipeline team - BuildOps

Sustainable environment

Engineering teams  SRE
WHERE DOES YOUR TEAM SIT?

Busy work

Keeping the lights on

Improving team life
Enforce, don’t fix
Sustainable system
Early results

Visibility
Create awareness across the team

Measure
Gather data to allow data driven decisions

Move the needle
Don’t boil the ocean, focus on a set of smaller results
Plan statistics

1154 builds

58% successful

63m average duration
Define rules  |  Enforce by automation  |  Shared ownership
SHARED OWNERSHIP
## Project Issue Metrics

<table>
<thead>
<tr>
<th>Resolver Group</th>
<th>First Responder</th>
<th>Opened bugs in last 90 days</th>
<th>Needs Acknowledgement</th>
<th>Needs Resolution</th>
<th>Ack SLA Breaches</th>
<th>Resolution SLA Breaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Simplify</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Better Retros</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Webscale Lightning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Webscare Thunder</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>isadykow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Flaky Tests</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Mobile</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Galaxy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dragonbell</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hedgehogs</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Storm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Documentation</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Activ8</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Frontend Platform</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Ecosystem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Core</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>
#3: Testing
Unit
Service
Unit
Balance
Balance
Impact

- Slow
- Duplication
- Flakes
The impact of flakes

**Build times**
Restarts cause build times to increase drastically

**Culture**
The culture of green goes out of the window

**Broken windows**
It starts with one. Soon, there will be a lot
Retries as a painkiller

Automatic retries relieve some pain. This is an early result, so see it as a work-around.
Rerun Failing Tests

During development, you may re-run failing tests because they are flaky. To use this feature through Maven surefire, set the `rerunFailingTestsCount` property to be a value larger than 0. Tests will be run until they pass or the number of reruns has been exhausted.

**NOTE:** This feature is supported only for JUnit 4.x.

```
mvn -Dfailsafe.rerunFailingTestsCount=2 test
```

If `rerunFailingTestsCount` is set to a value smaller than or equal to 0, then it will be ignored.

Output flaky re-run information on the screen

When `rerunFailingTestsCount` is set to a value larger than 0 and the test fails, then it will be re-run and each run information will also be output. Each run with its number and trimmed stack trace will be output.

If the test passes in its first run, then the output on the screen will be identical to the case where `rerunFailingTestsCount` is not used.

If the test fails in the first run, then there are two possible cases:

1) The test passes in one of its re-runs: the last run will be marked as PASS

For example, a test passed in its second run will output on the screen:
Fix the root cause
Iterate

Test strategy

Flake policy
### CREATE VISIBILITY

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Status</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONF-45718</td>
<td>Flaky test - PermissionsXmlRpcAcceptanceTest testGetSinglePageHonoursPageLevelPermissions</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45713</td>
<td>Flaky test - MacroBrowserTest testEditMacroWithMultipleMacros</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45703</td>
<td>Flaky test - AutowatchTest testCreatorAlreadyUnwatchStillUnwatchIfBlogPostsModifiedByAnotherUser</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45692</td>
<td>Flaky test - KeyboardShortcutDialogTest testShortcutCheckboxForAnonymousUsers</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45691</td>
<td>Flaky test - EmbeddedImagesTest testCopyPasteImage</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45681</td>
<td>Flaky test - AddUserXmlRpcAcceptanceTest testSpecialCharacterUserNames</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45680</td>
<td>Flaky test - AddUserXmlRpcAcceptanceTest testCreateUser</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45674</td>
<td>Flaky test - UnicornReverseMigrateServletAcceptanceTest testissuerKeyMismatchAsapHeader</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45672</td>
<td>Flaky test - Atlassian Table Highlight tests CELL. Change colour; remove colour.</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
<tr>
<td>CONF-45665</td>
<td>Flaky test - AddPageAcceptanceTest testAddPageWithViewRestrictionsCorrectlySetsOwner</td>
<td>Unassigned</td>
<td>NEEDS VERIFICATION</td>
<td>Unresolved</td>
</tr>
</tbody>
</table>
Flake policy

- Detect
- Raise ticket
- Ignore
- Fix or remove
Deleting tests, seriously?
#4: Releases
RELEASE CYCLE

Latency

- New Daily Releases
- Legacy Once a Week Releases
Congestion
Impact

Ends at master

Big bang

Releases on hold

Dog-fooding becomes hard
For once a free lunch
Fork and Join solved most of our problems.
12 FACTOR CONFLUENCE

Expose a port
Only access to the service

Stateless*
Run as many nodes as you need

Health check
200 app is alive. 500 app is unhealthy, destroy the node
name: Confluence

description: Confluence Cloud

links:

  binary:
    type: docker
    name: docker.atlassian.io/confluence
    tag: latest

healthcheck:
  uri: /wiki/internal/healthcheck

deepcheck:
  uri: /wiki/internal/deepcheck

semanticCheck:
  dockerImage:
    name: docker.atlassian.io/confcloud/deployment-
resources:
  - type: sqs
    name: default
    attributes:
      MaxReceiveCount: 20
      VisibilityTimeout: 60
    scaling:
      instance: m3.xlarge
      min: 7
Reduce build times

Fix dog-fooding and the feedback loop

Iterate
Feedback

We loved it.
Soak time
Feature flags

Decouple code and features

Merge often

One process
Things to take home
Identify your problem areas

- Release cadence
- Ownership
- Test automation
- Complexity
Focus on momentum

Clear goal

Early results

Momentum
Good intentions don’t work.
Sustainable system
HAVING A GREEN HUDSON/JENKINS ... ... IS ... 

DIDN'T KNOW IT CAN CHANGE IT'S COLOR ... A GREAT EXPERIENCE
Thanks!