Feature branches and toggles in a post-GitHub world

GOTO Chicago 2017

Sam Newman

ThoughtWorks®



ATOMIST

@gotochi

@samnewman

Sam Newman & Associates

@gotochi

@samnewman

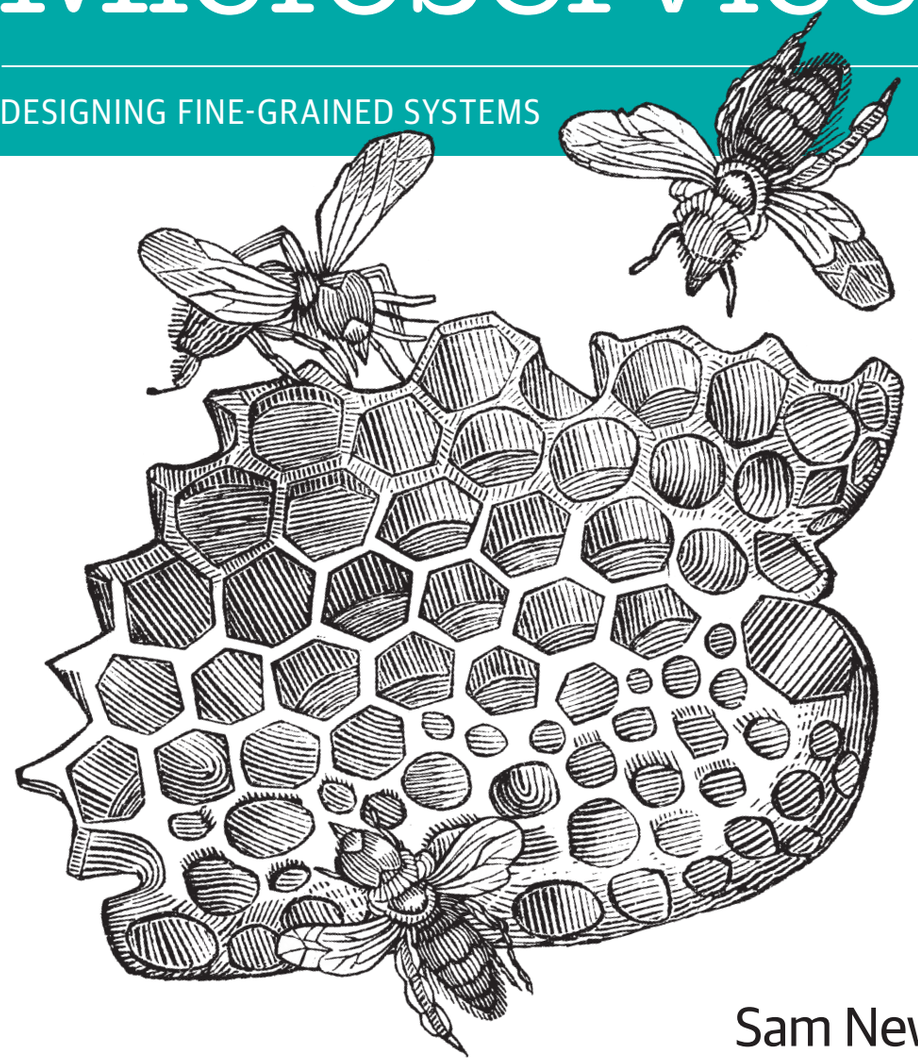
Sam Newman & Associates

(there are no associates)

O'REILLY®

Building Microservices

DESIGNING FINE-GRAINED SYSTEMS



Sam Newman

@gotochi

@samnewman

2004

Dixons travel

Service


16:40
16:40
9
10



@gotochi

@samnewman



@gotochi

@samnewman



@gotochi

@samnewman

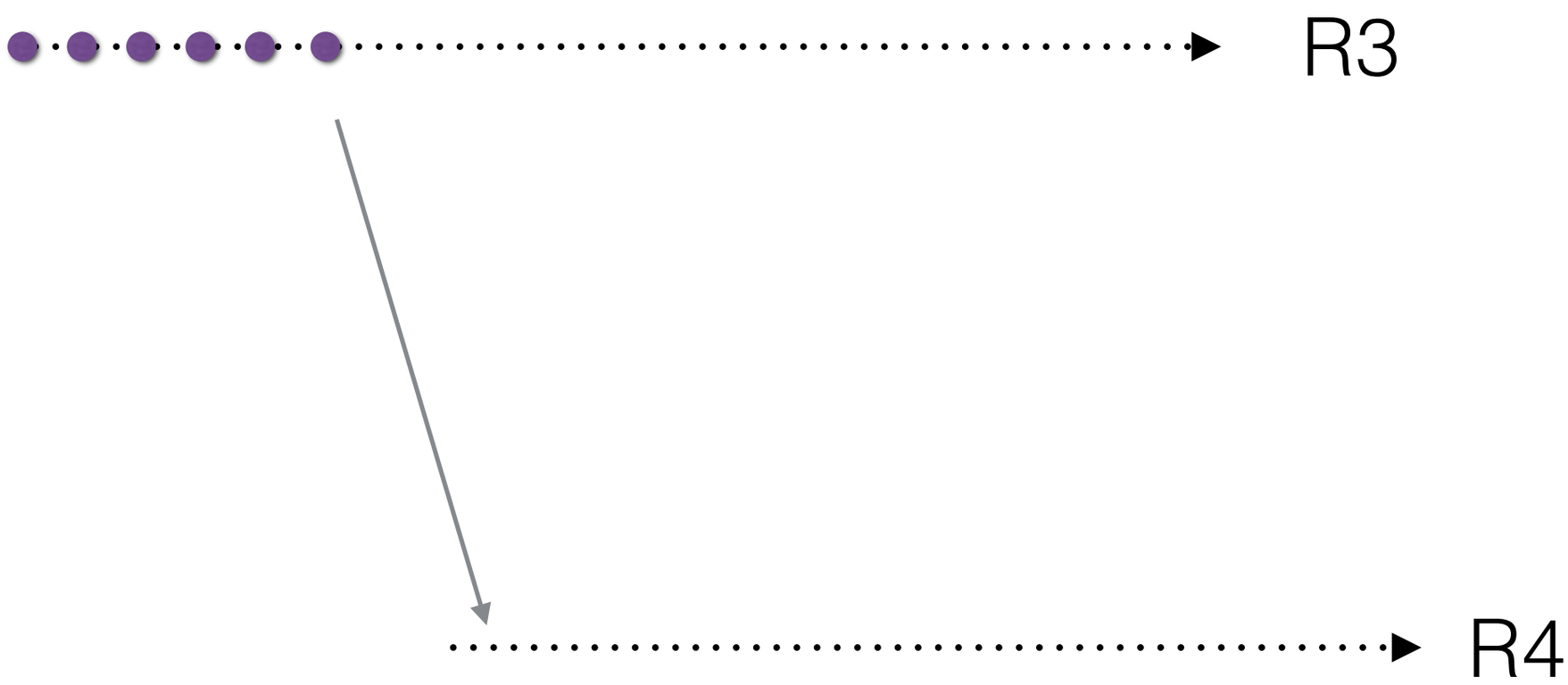


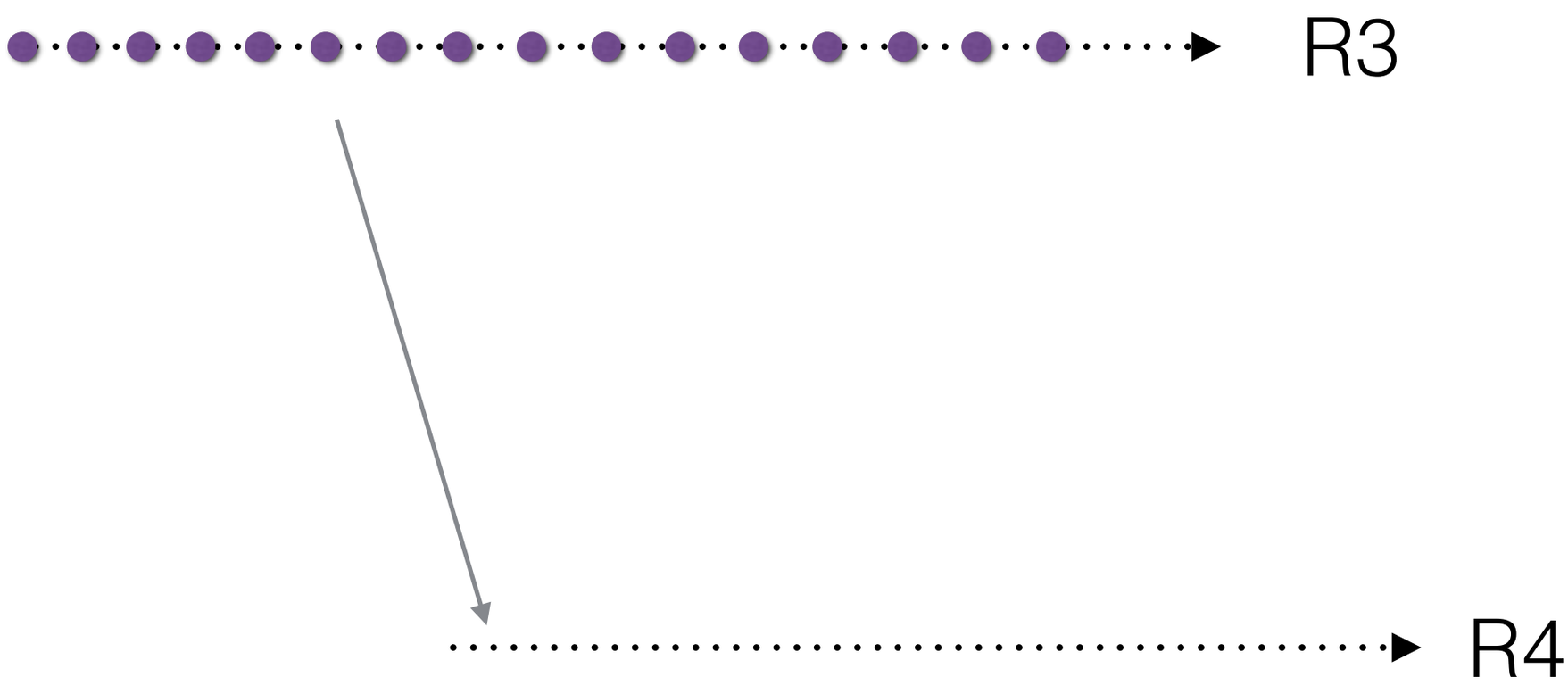
@gotochi

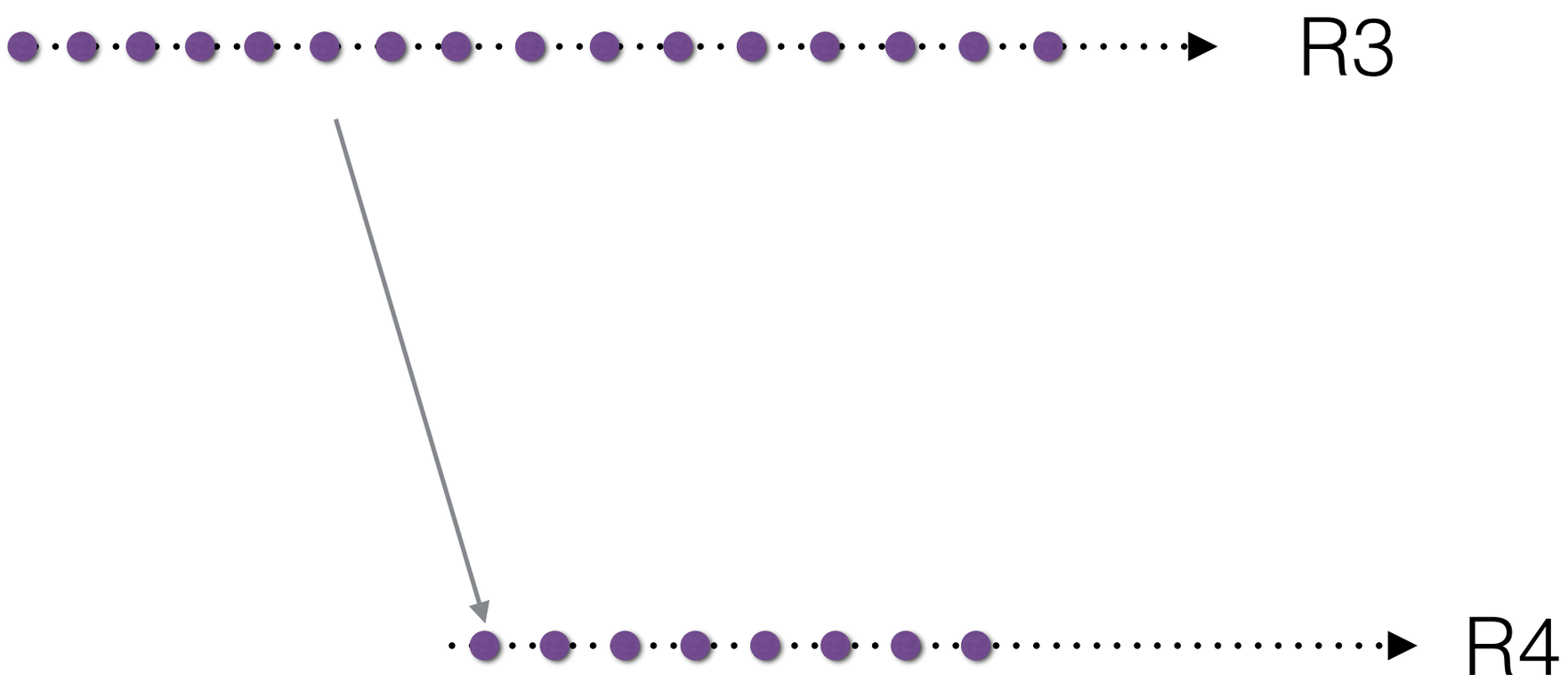
@samnewman

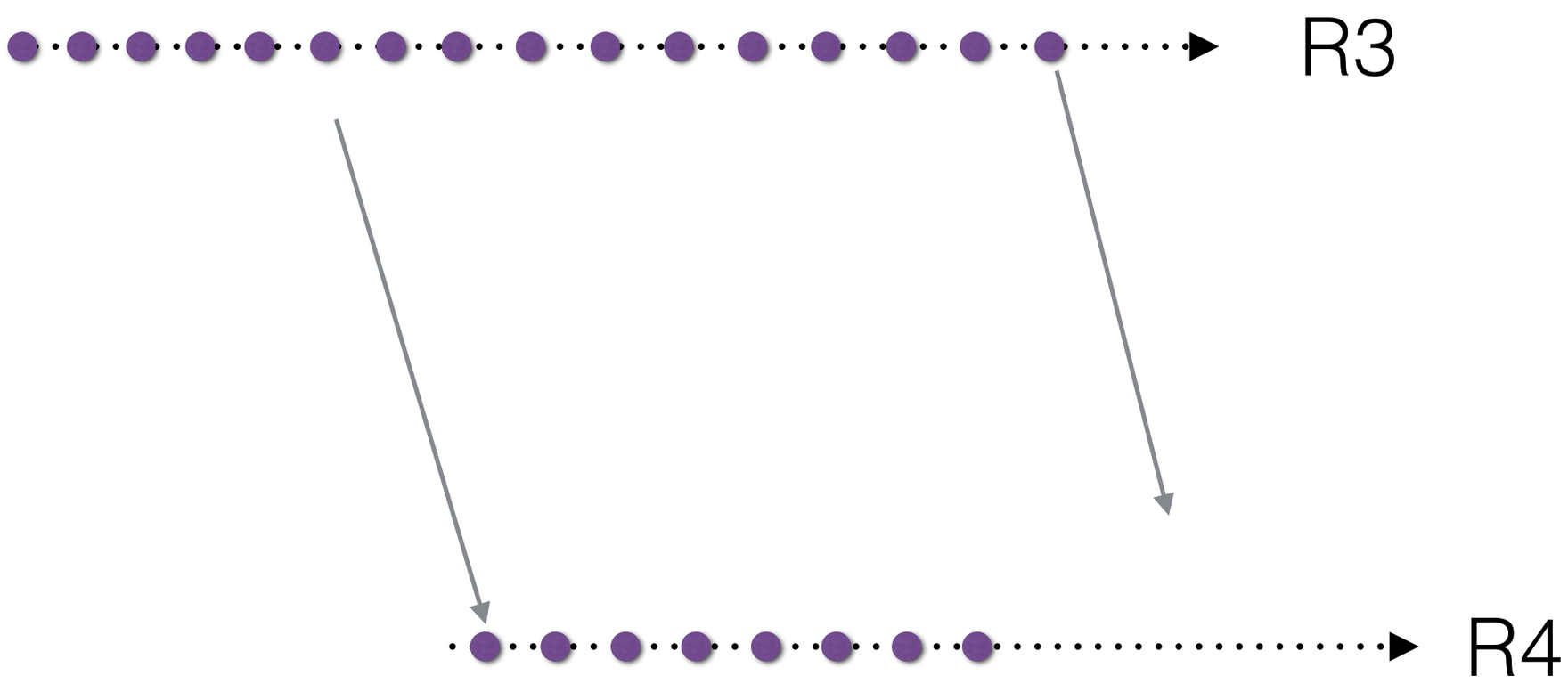

R3

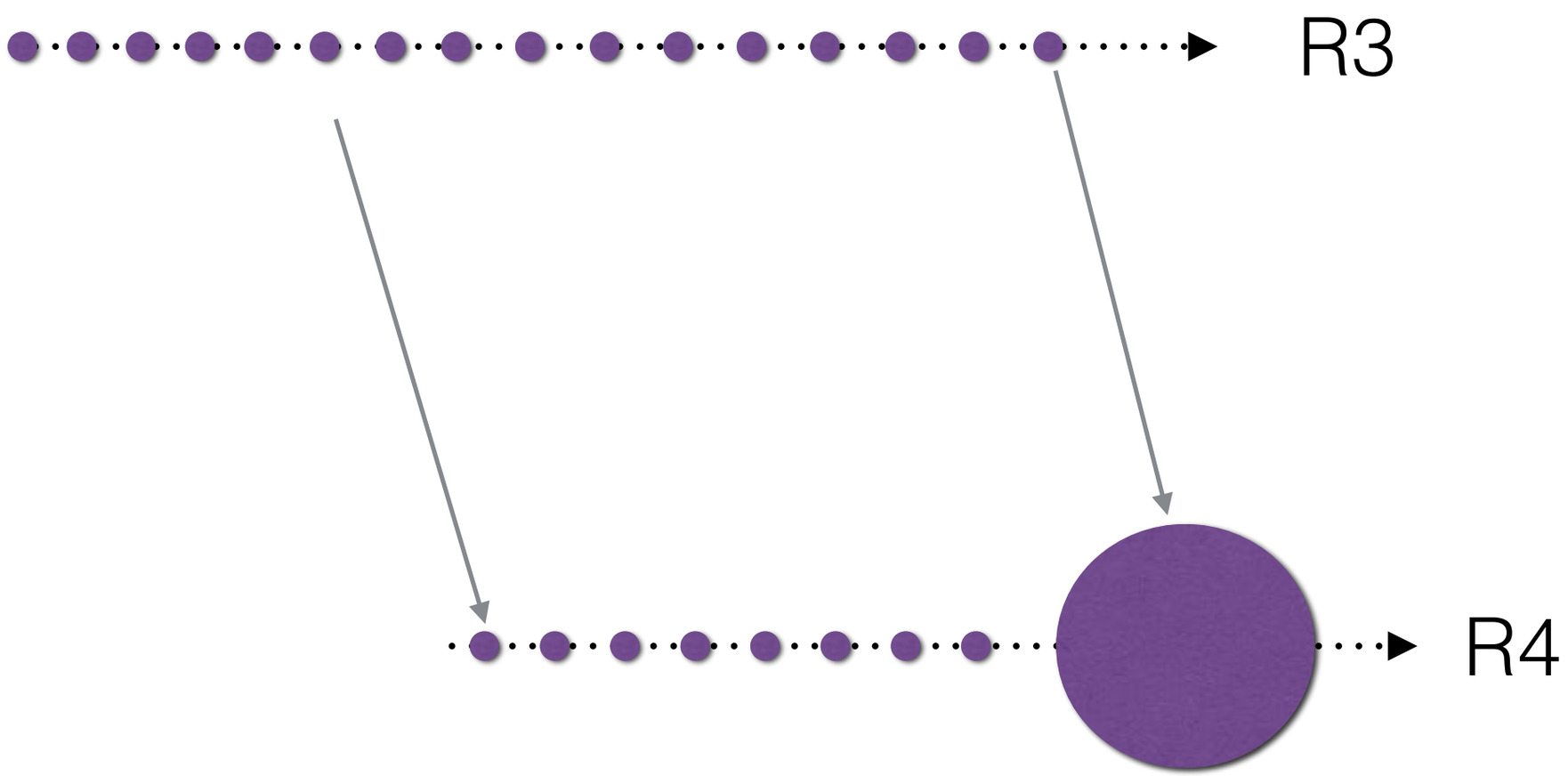

R3


R3
R4


R3
R4


R3
R4


R3
R4


R3
R4



https://www.flickr.com/photos/jubilo/6965626176/

R3-R4 Merge Bug Fix Team

@gotochi

@samnewman

R3-R4 Merge Bug Fix Team

yay

@gotochi

@samnewman



Continuous Integration

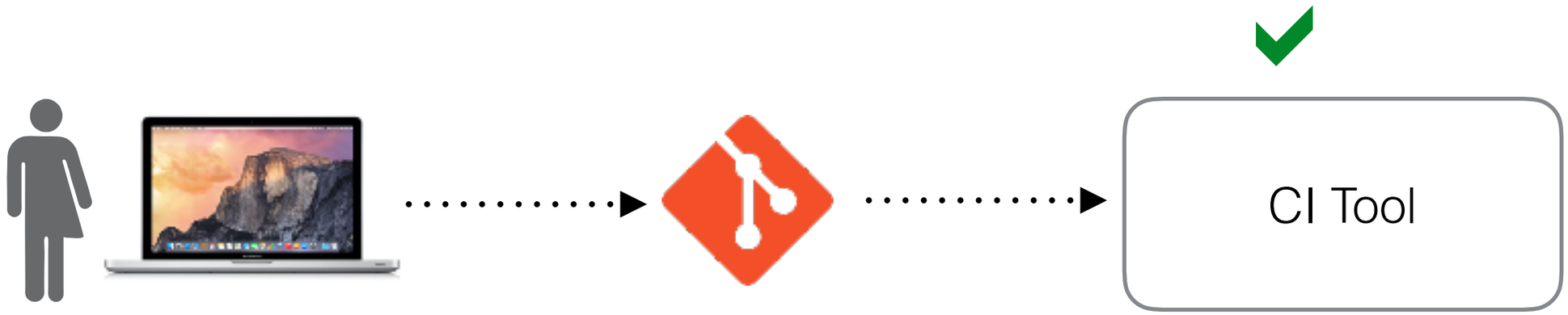
@gotochi

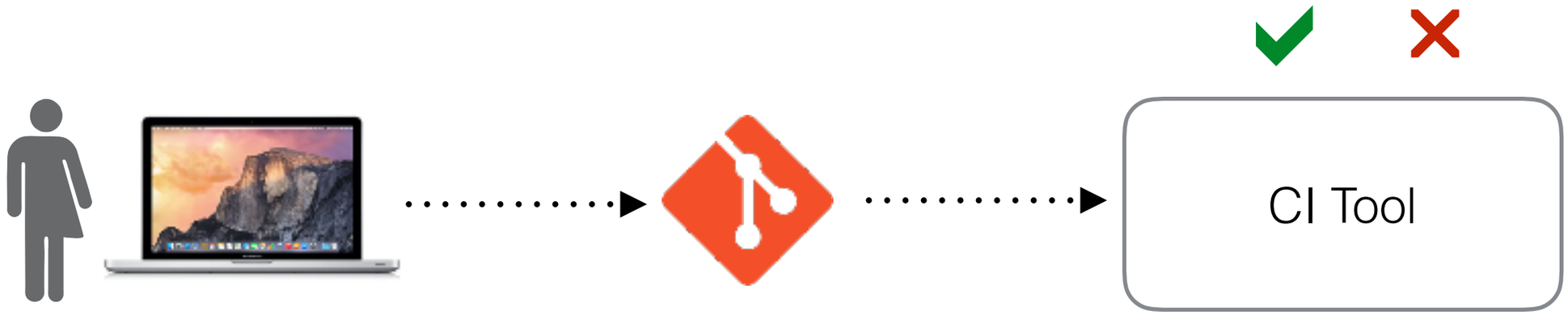
@samnewman






CI Tool


CI Tool

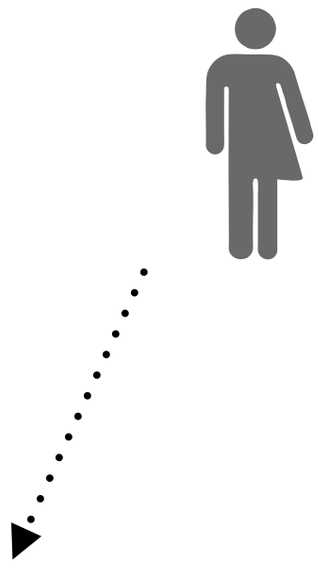

CI Tool





@gotochi

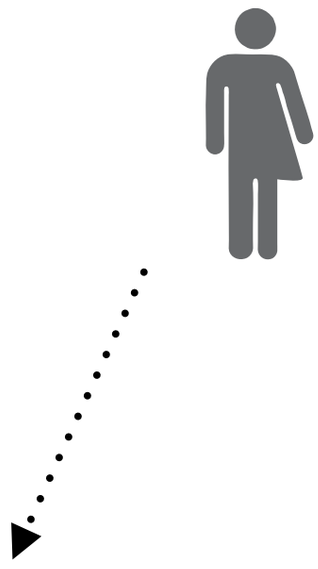
@samnewman





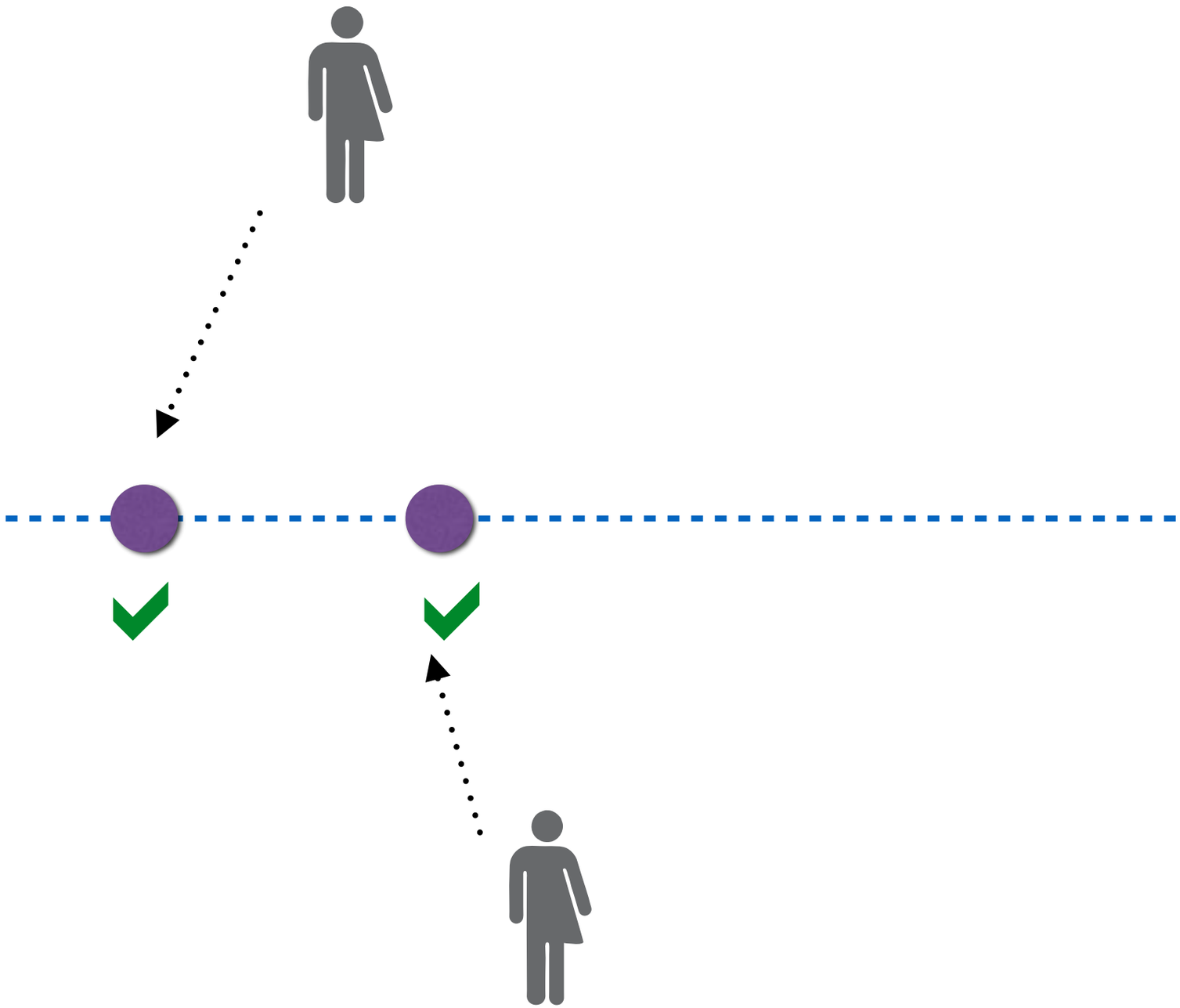
@gotochi

@samnewman

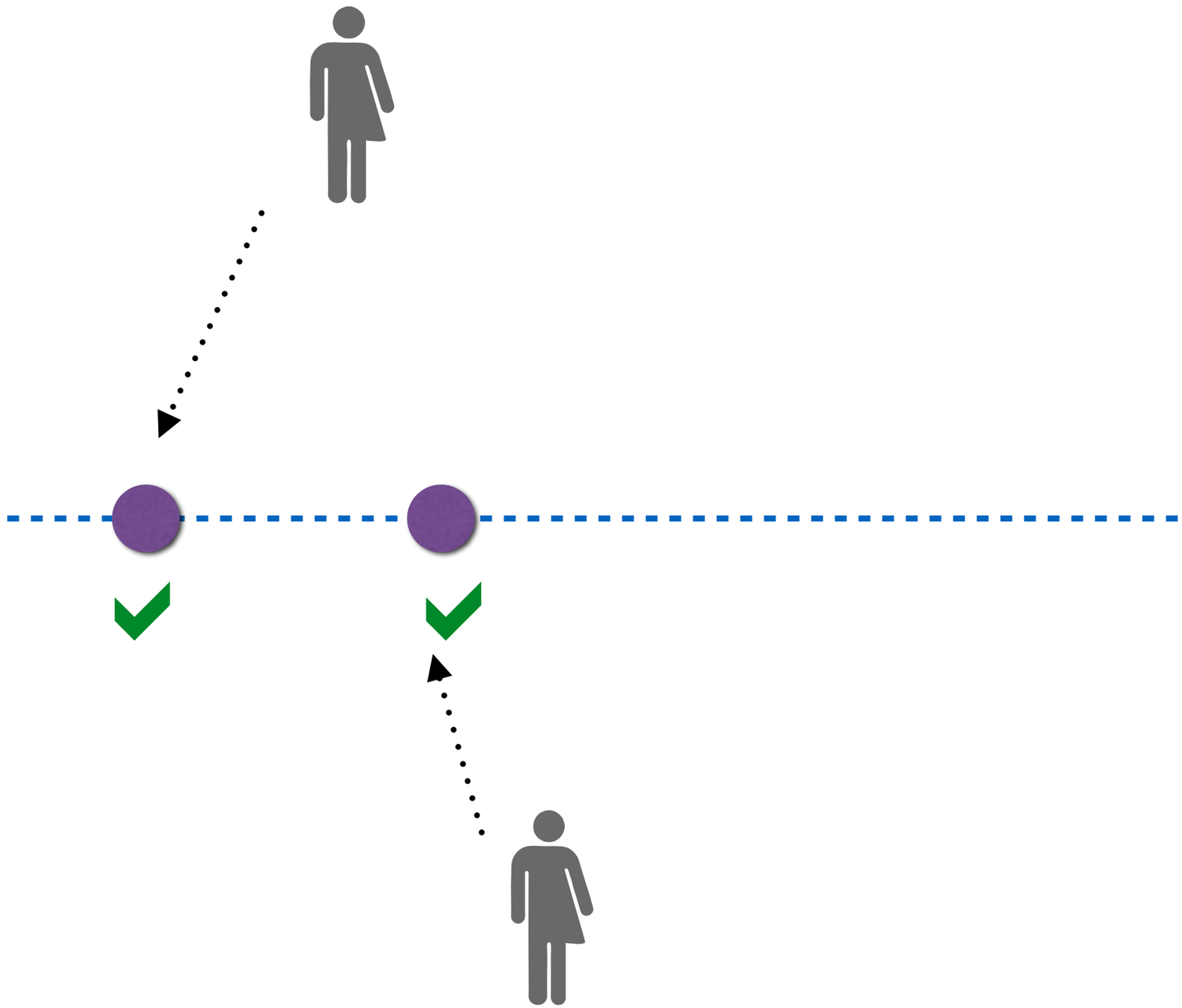








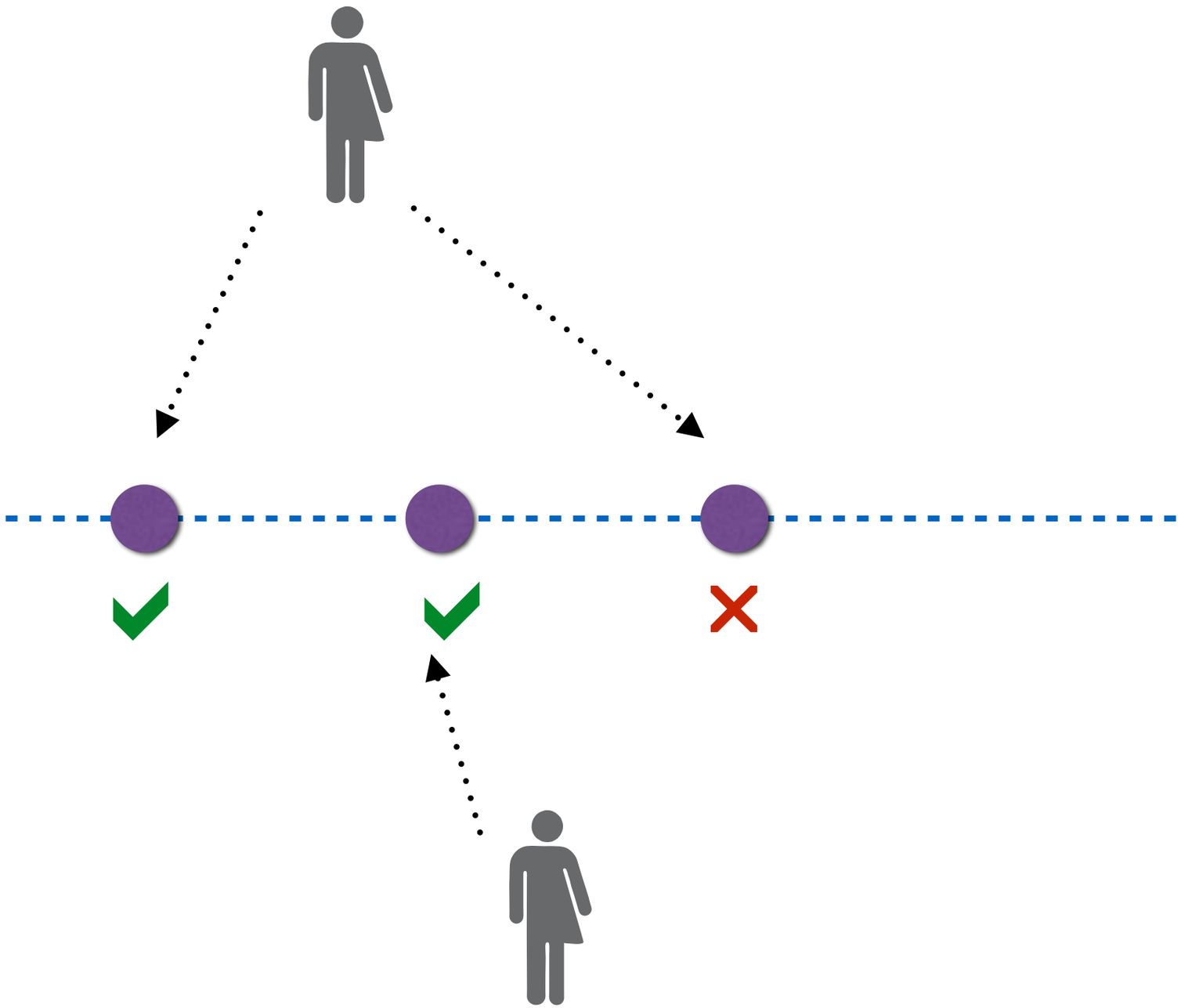
1. Validate the integration



@gotochi

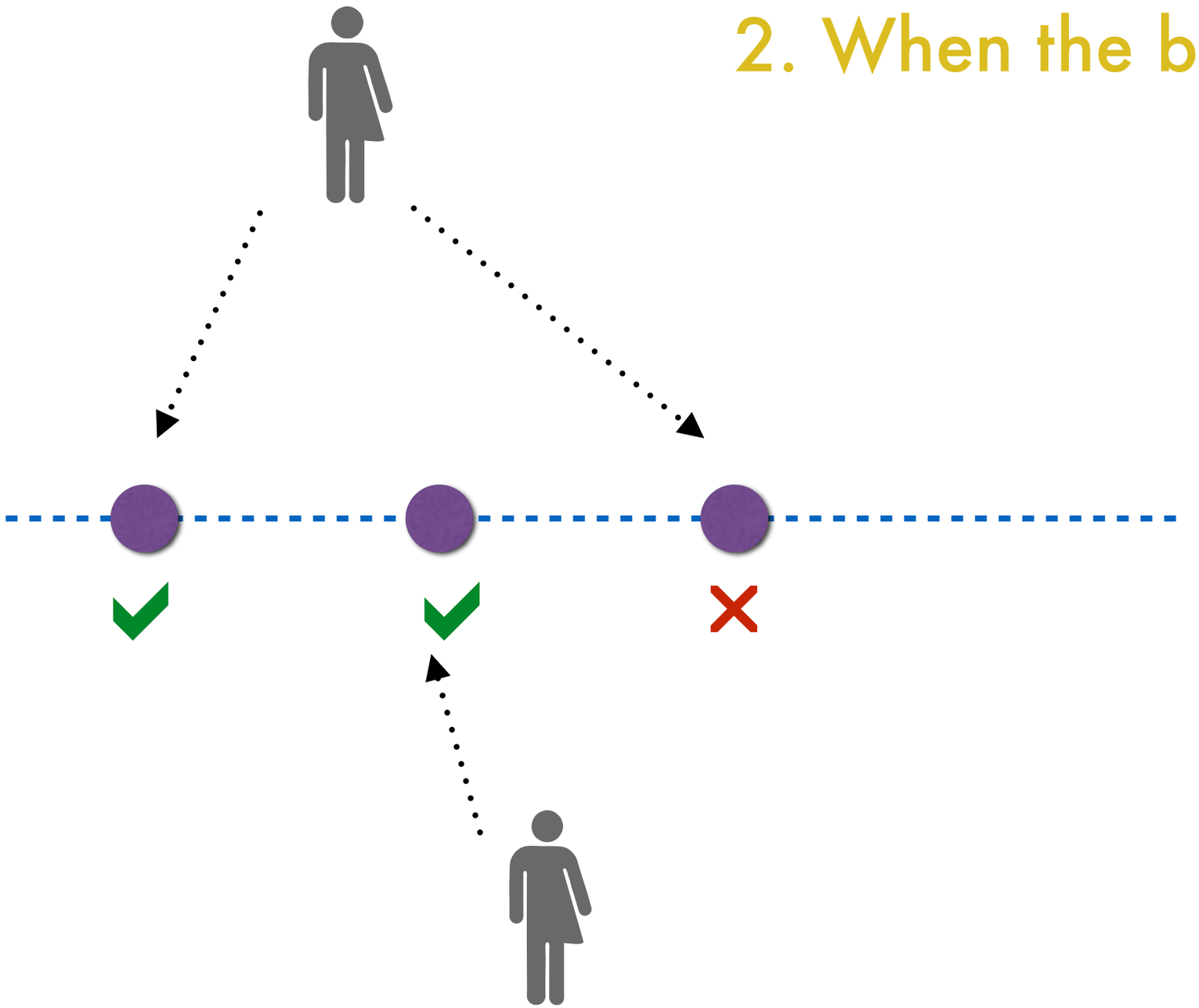
@samnewman

1. Validate the integration



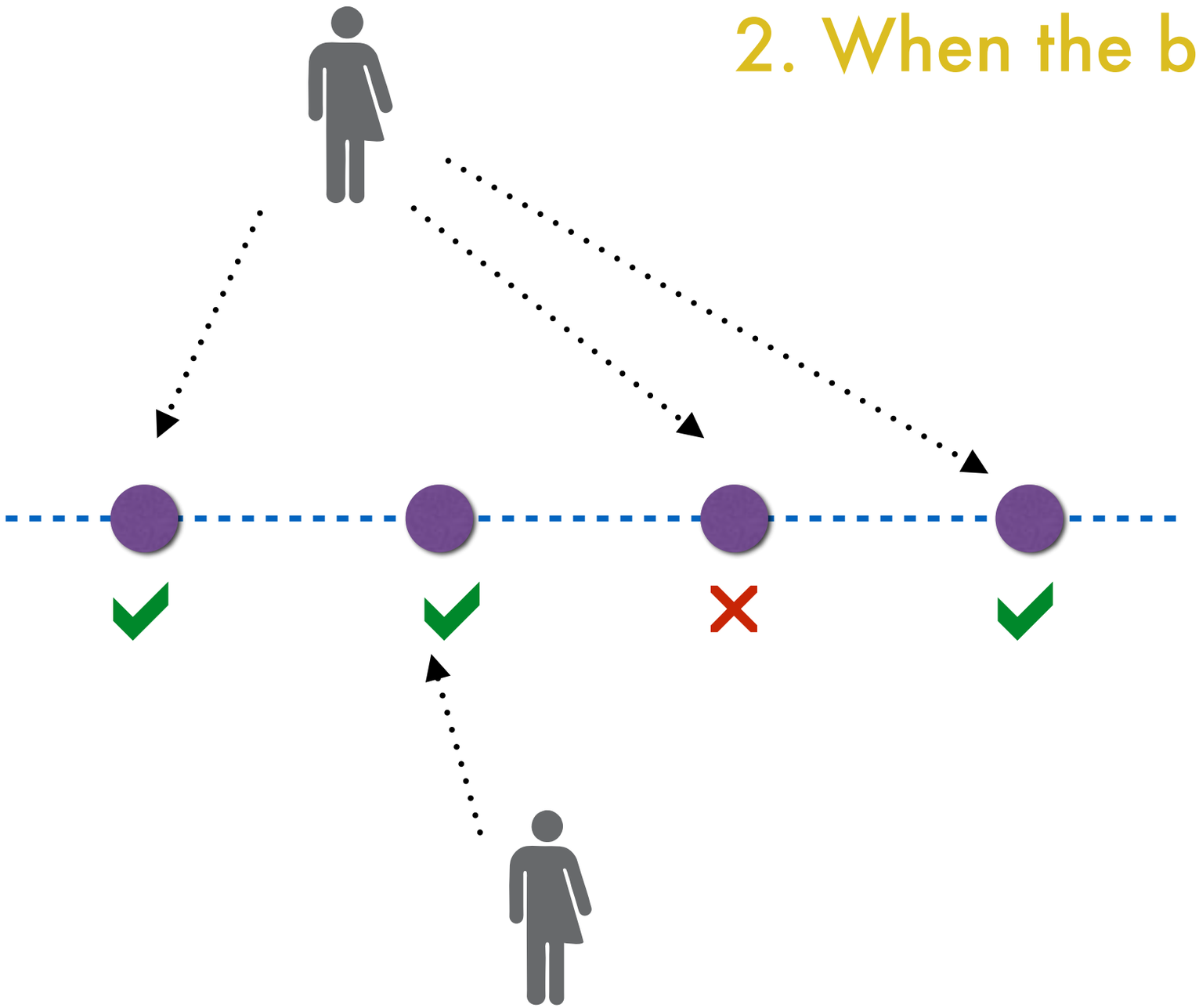
1. Validate the integration

2. When the build breaks, fix it!



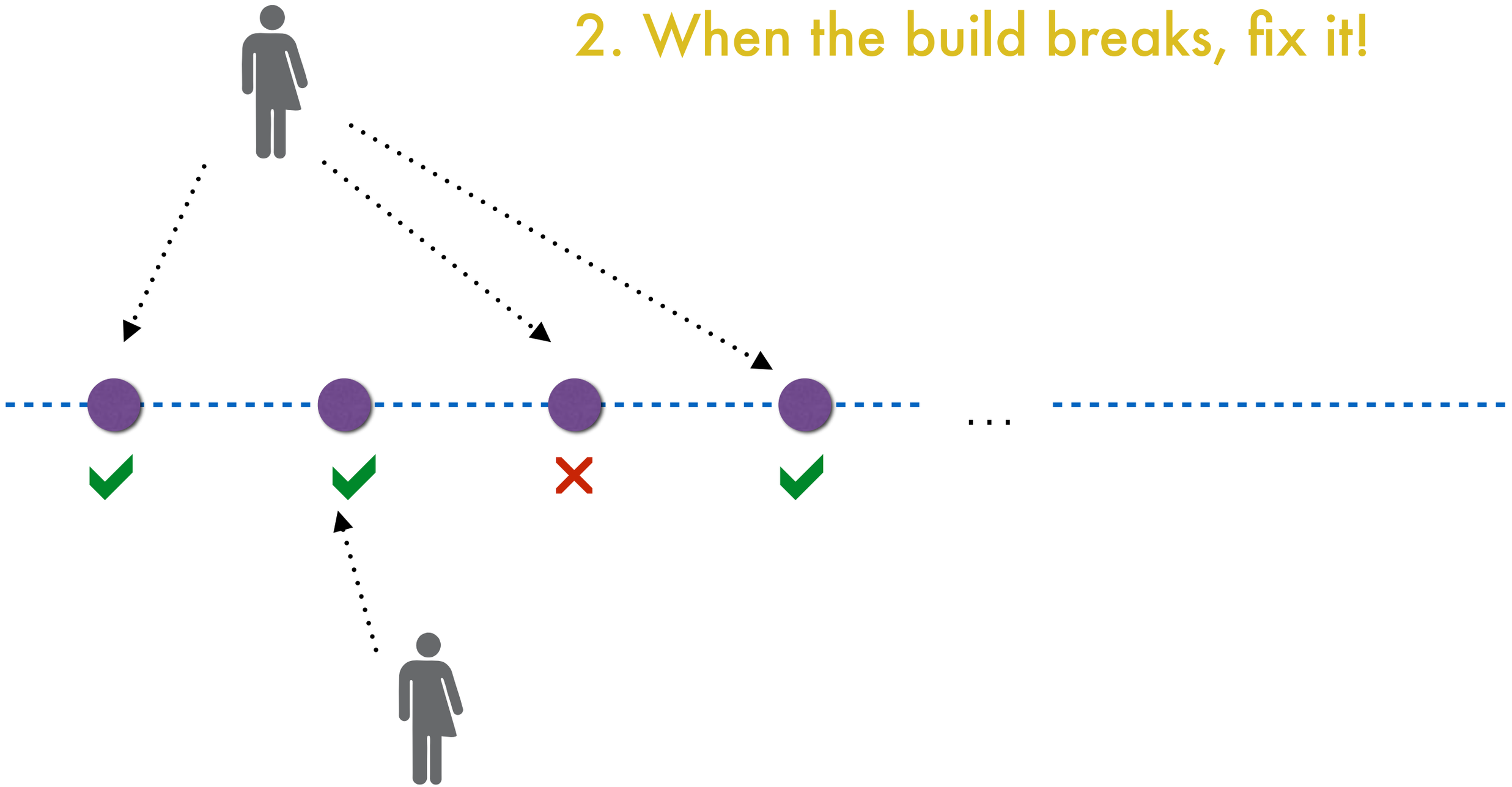
1. Validate the integration

2. When the build breaks, fix it!



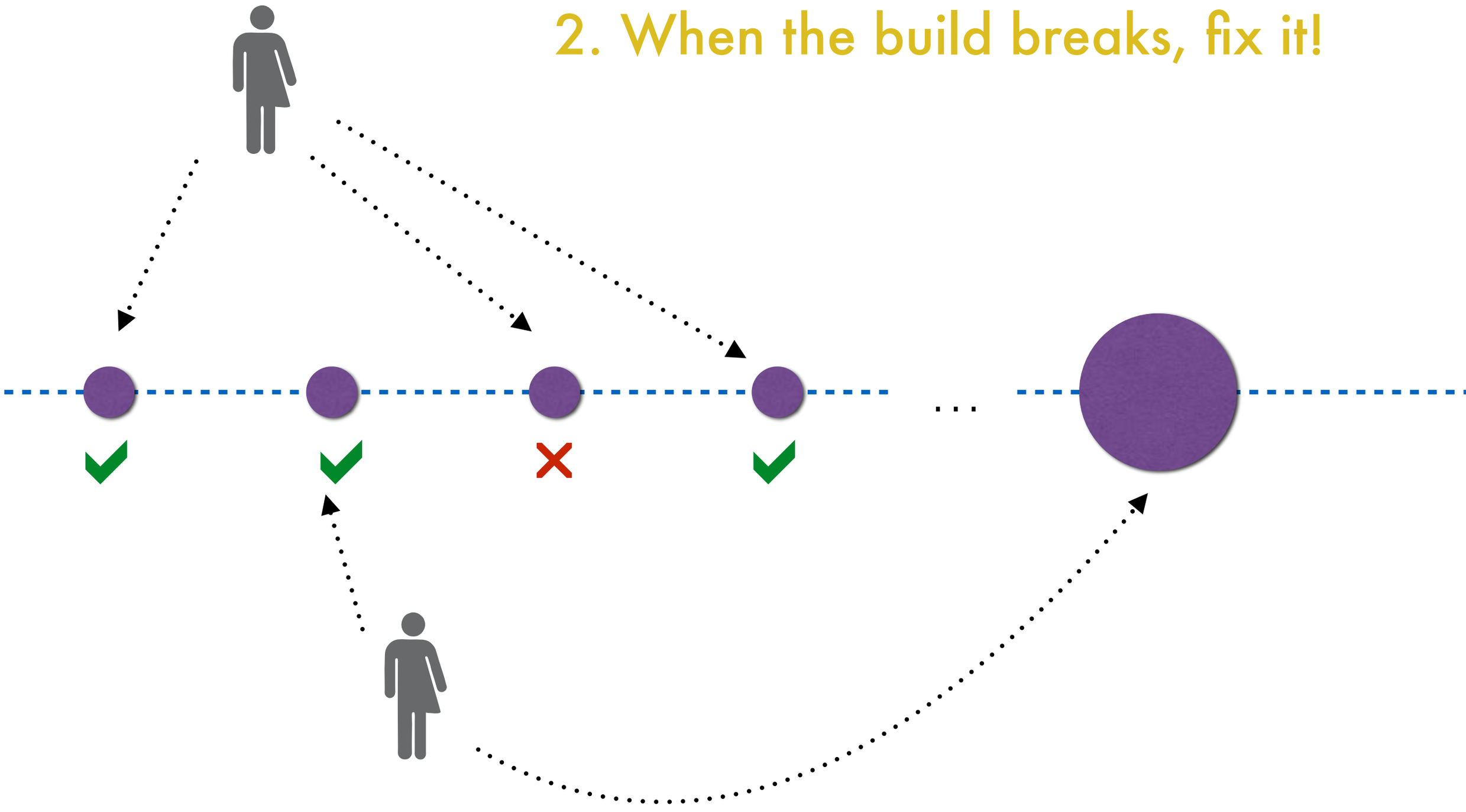
1. Validate the integration

2. When the build breaks, fix it!



1. Validate the integration

2. When the build breaks, fix it!

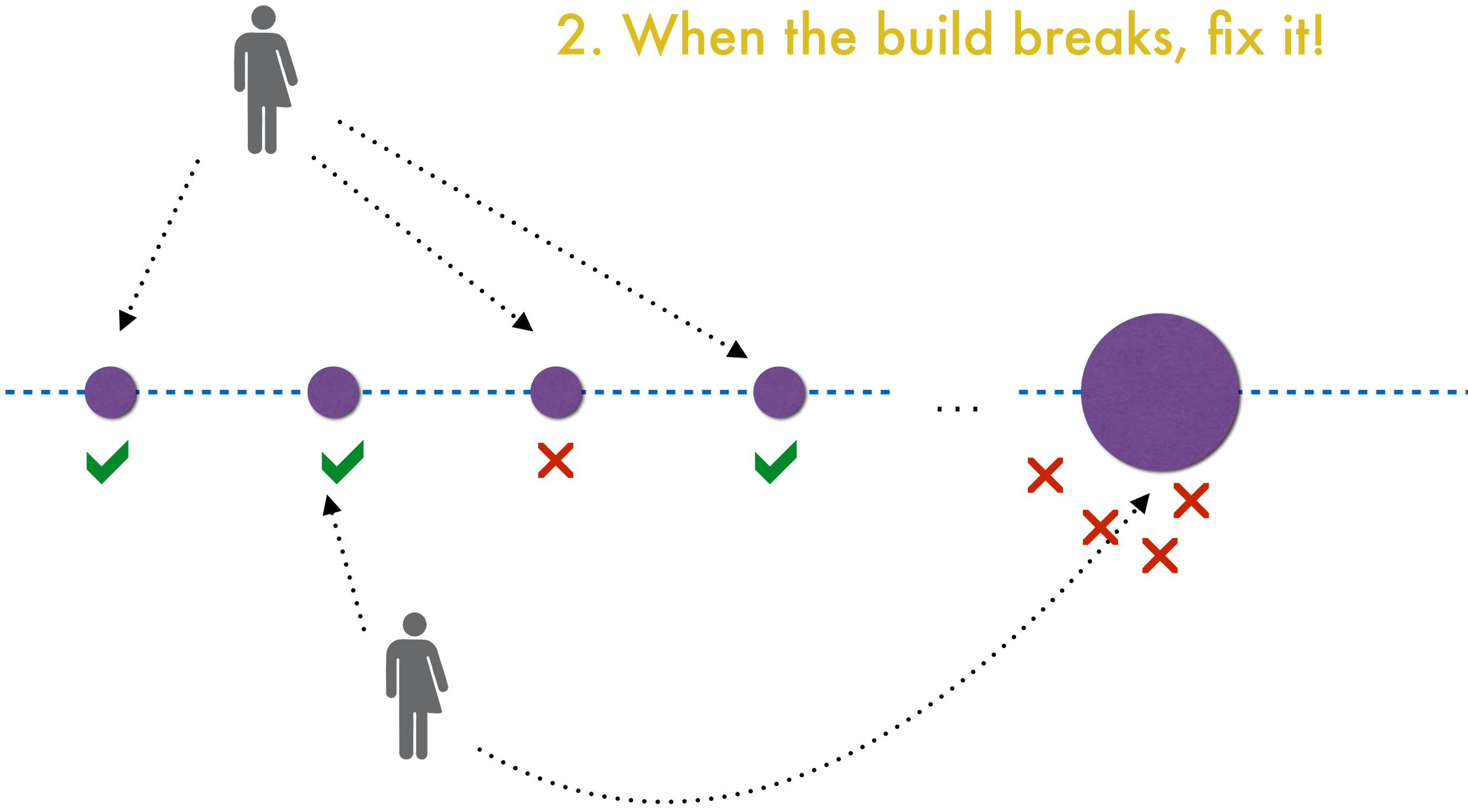


@gotochi

@samnewman

1. Validate the integration

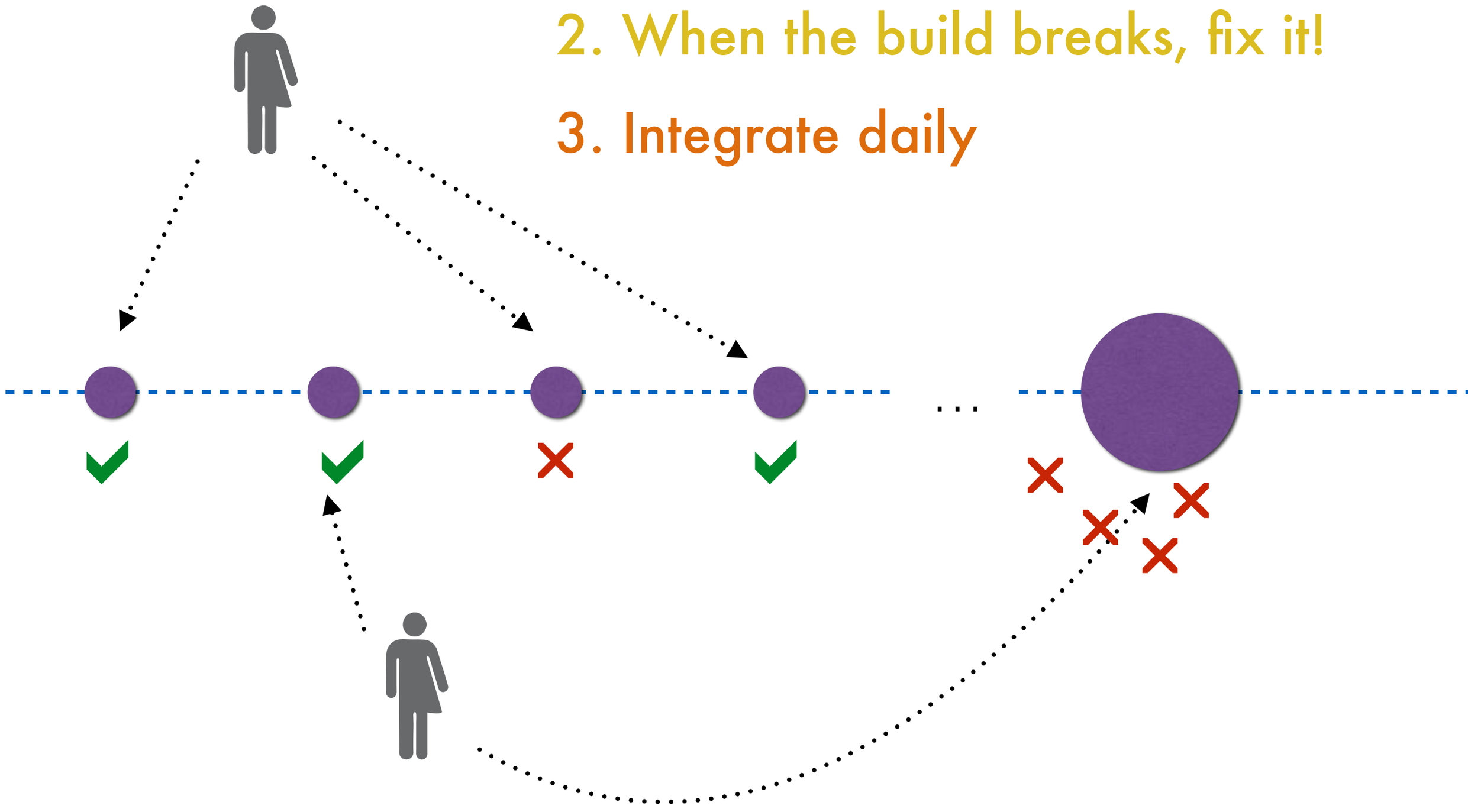
2. When the build breaks, fix it!



@gotochi

@samnewman

1. Validate the integration
2. When the build breaks, fix it!
3. Integrate daily



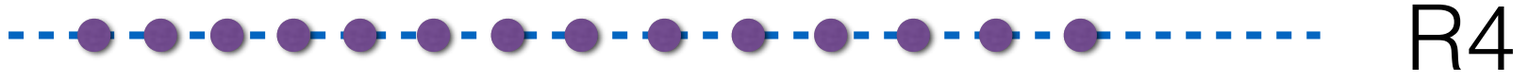
@gotochi

@samnewman



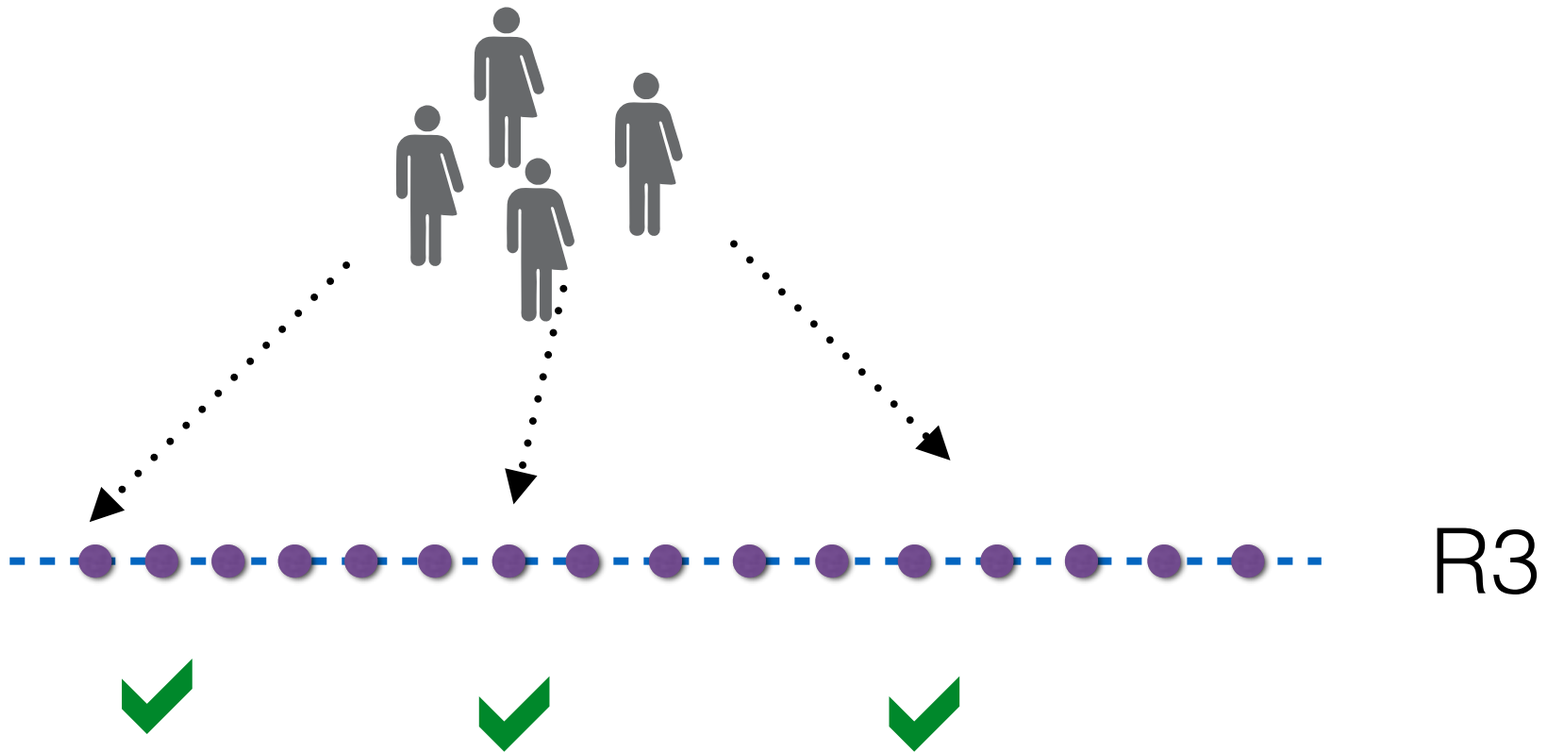


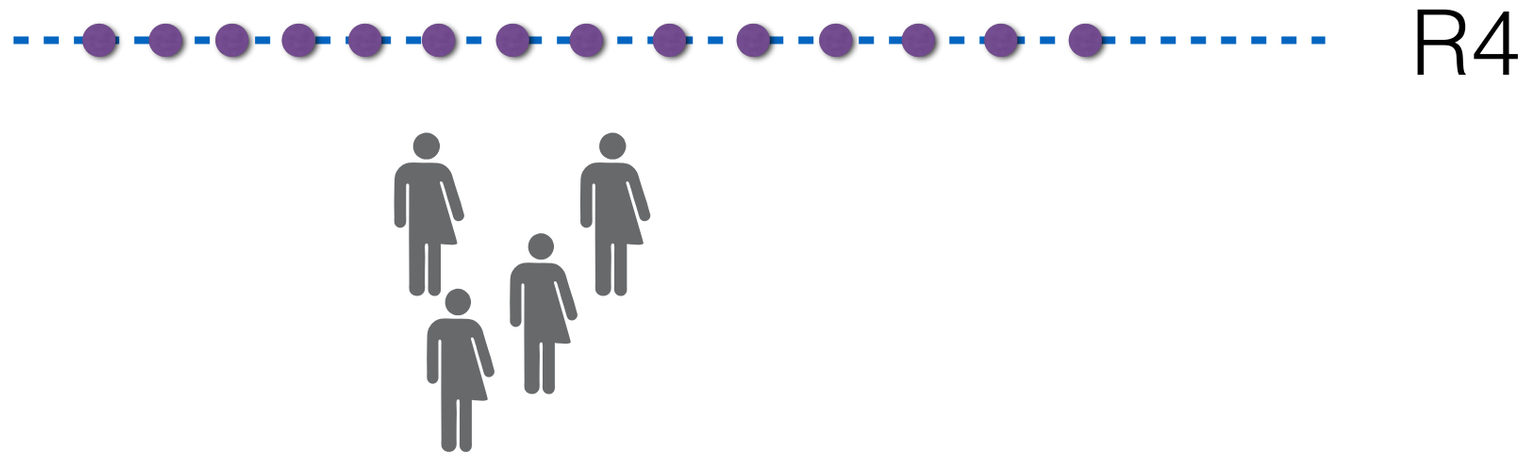
R3

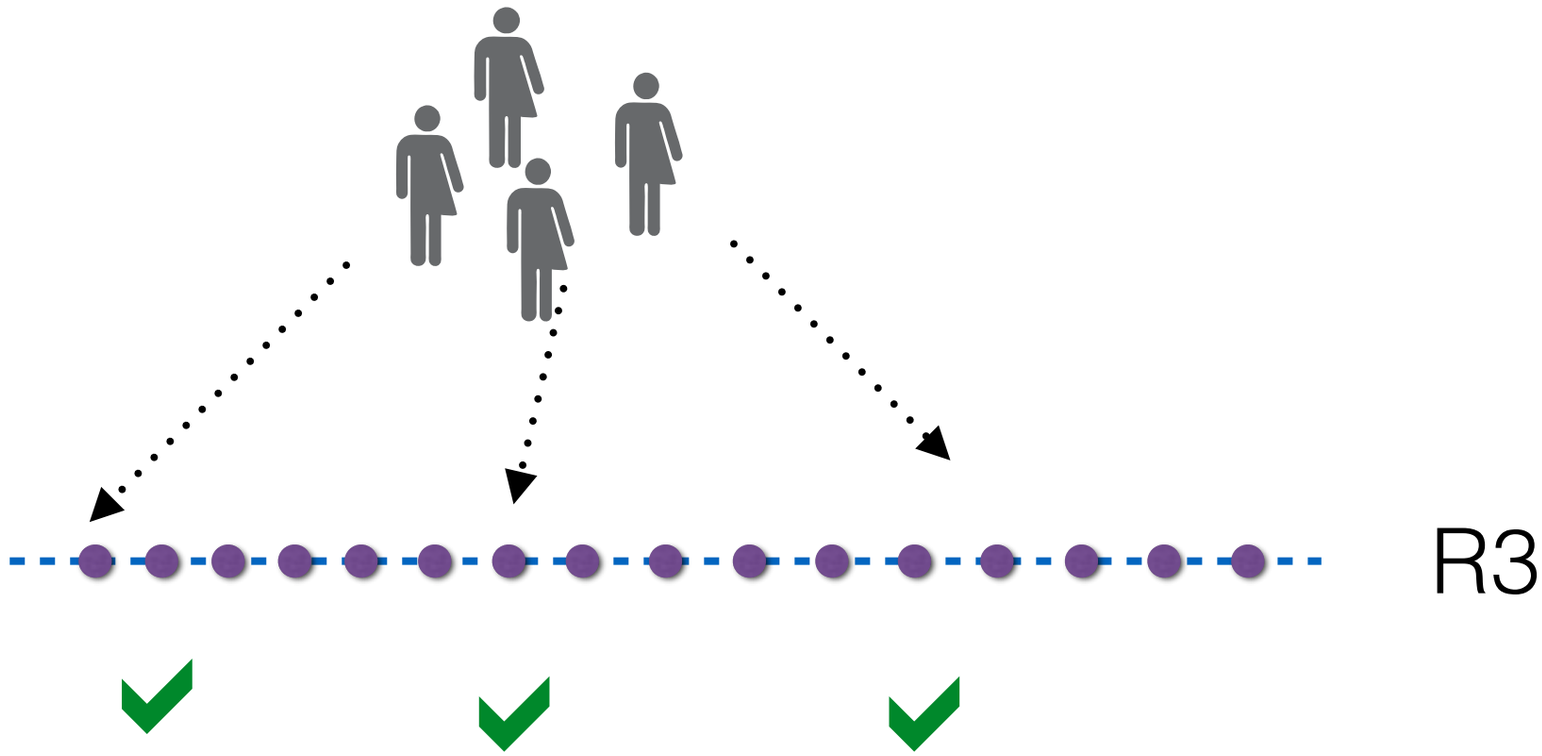


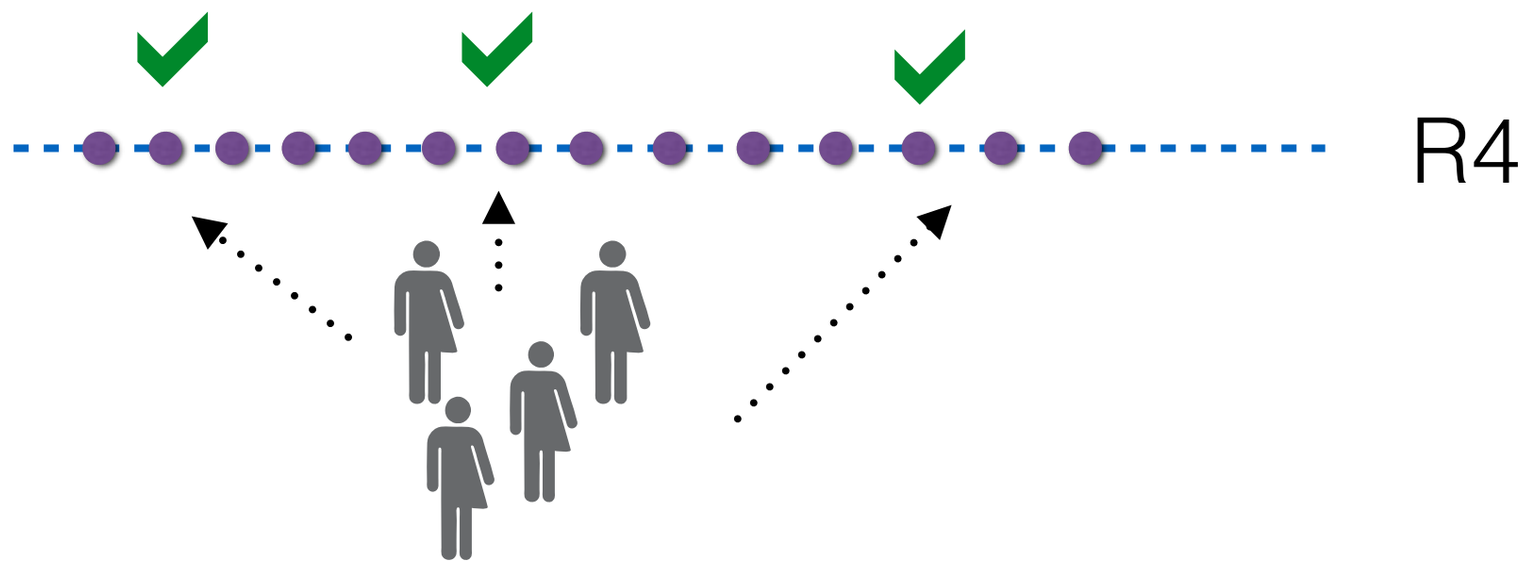
R4

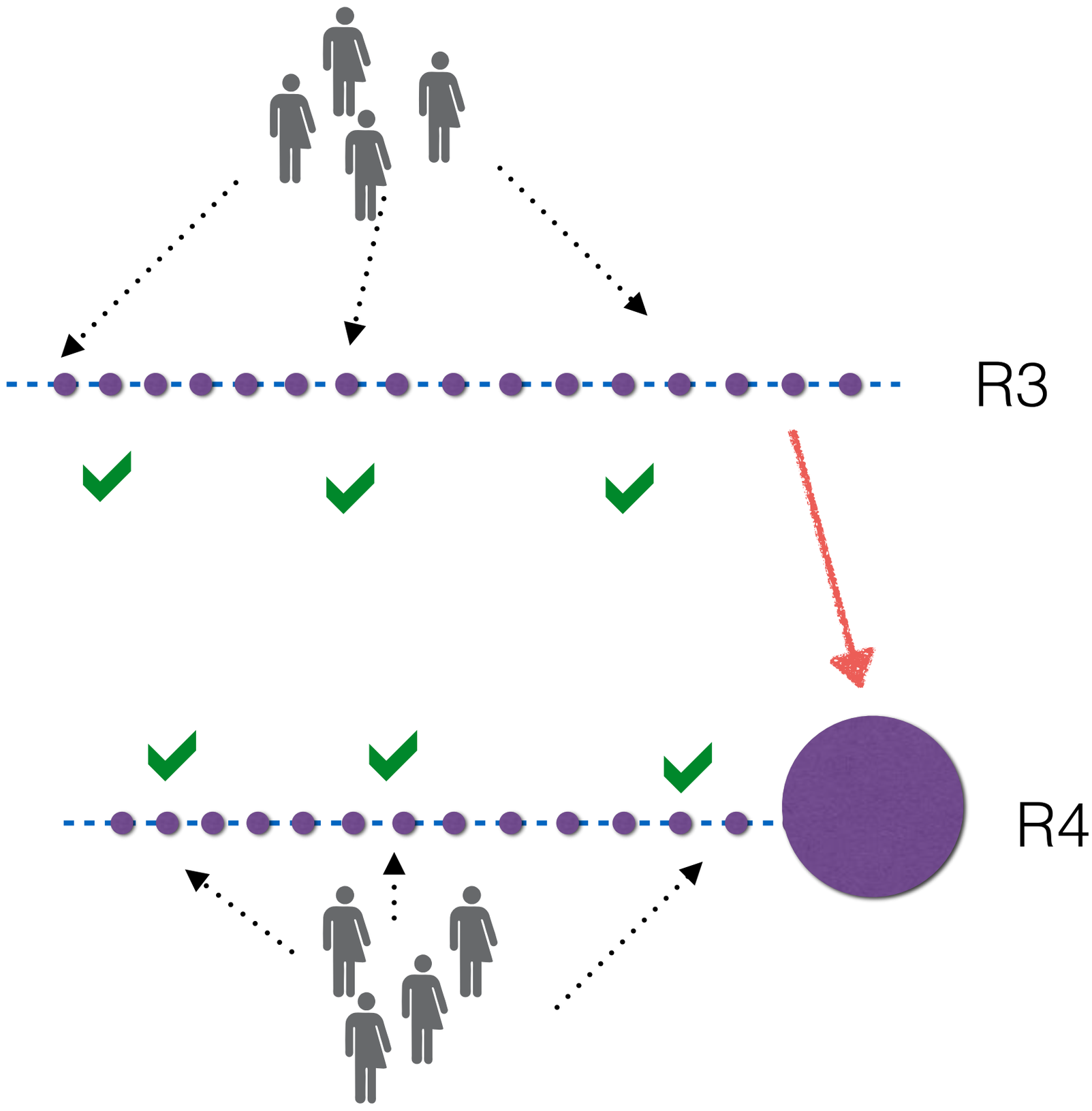



R3


R4


R3


R4


R3
R4

Integrate Once A Day

@gotochi

@samnewman

How can you deal with half-finished work?

@gotochi

@samnewman

Option 1:

@gotochi

@samnewman

Option 1:

Wait to check in

Option 2:

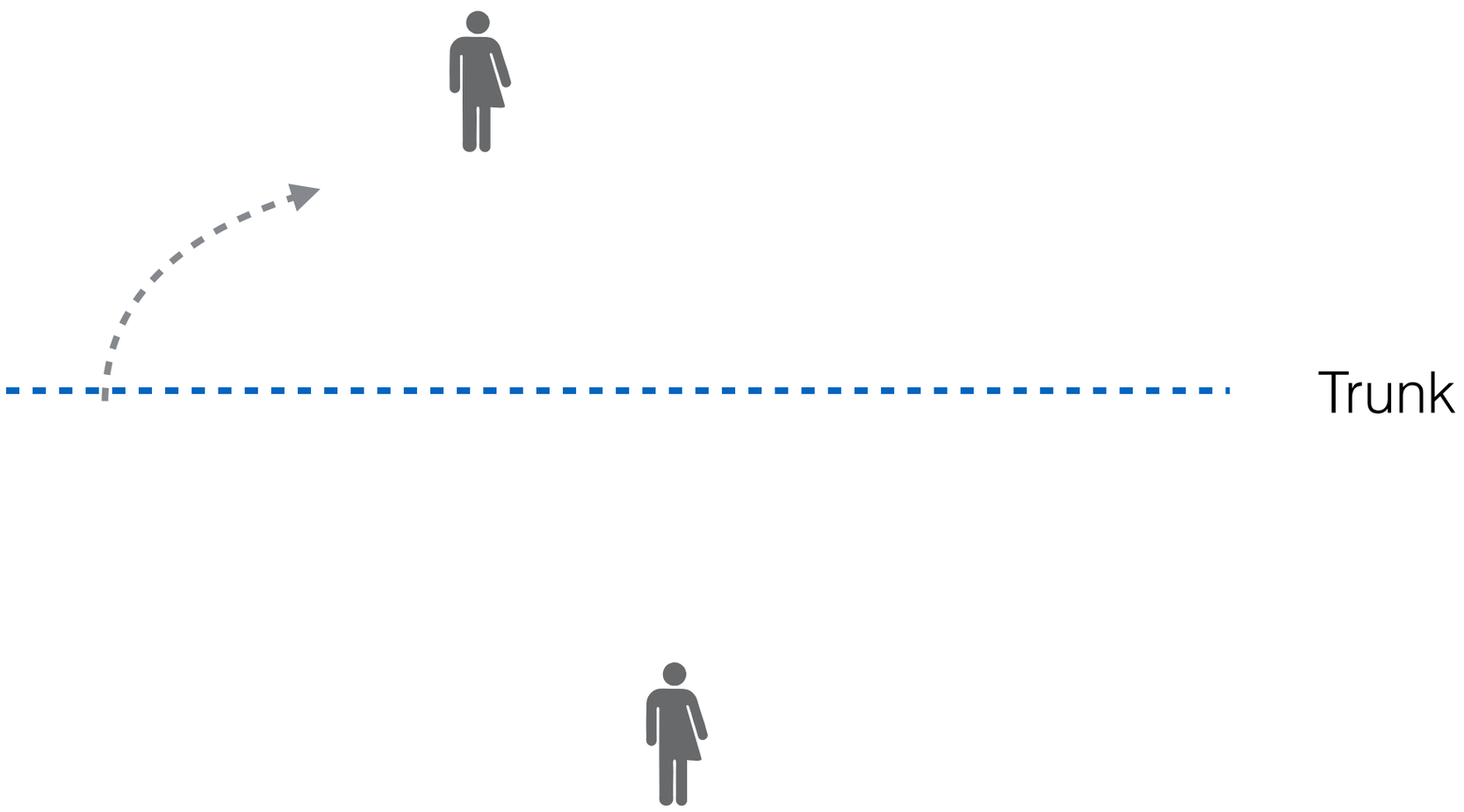
Option 2:

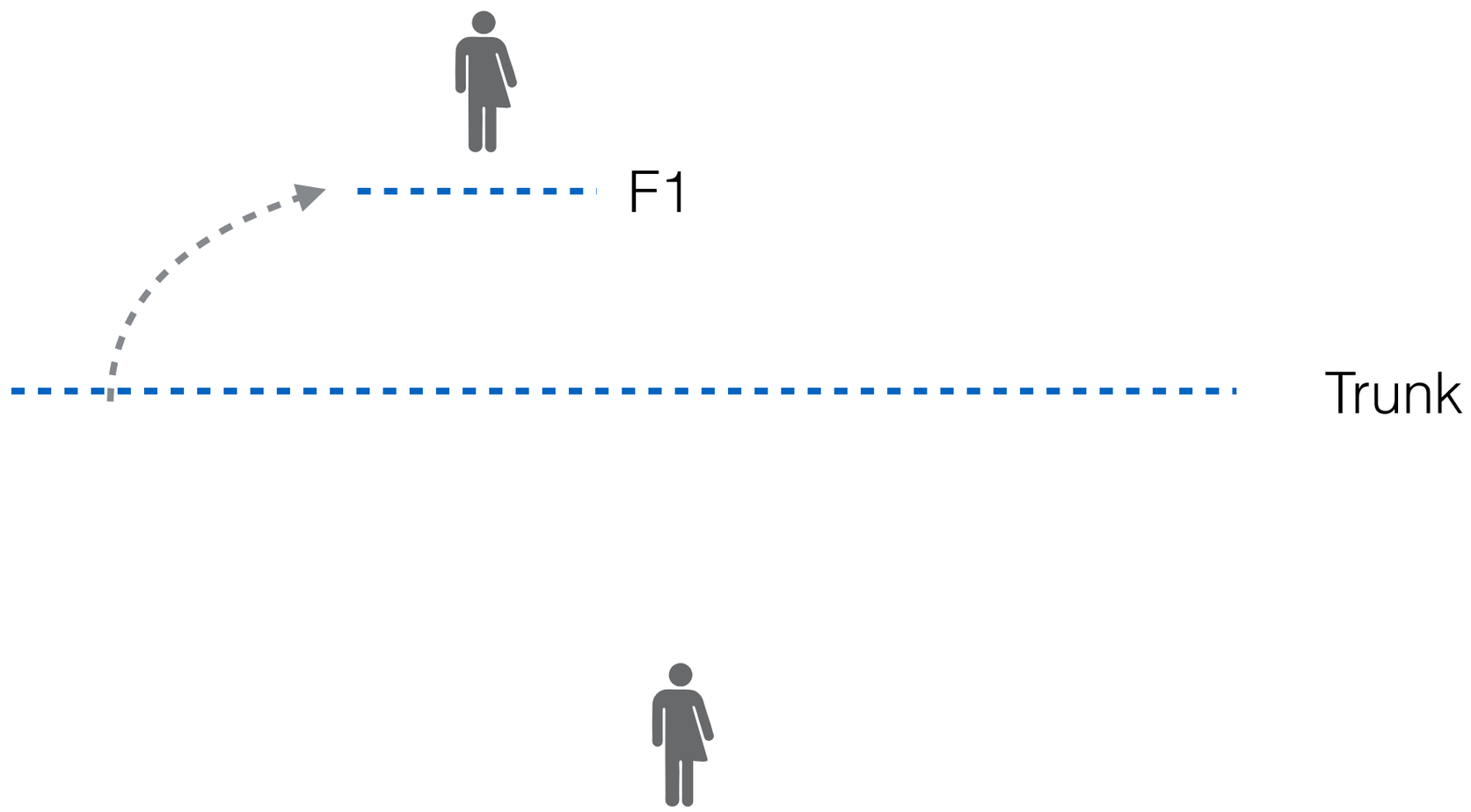
Make a branch!

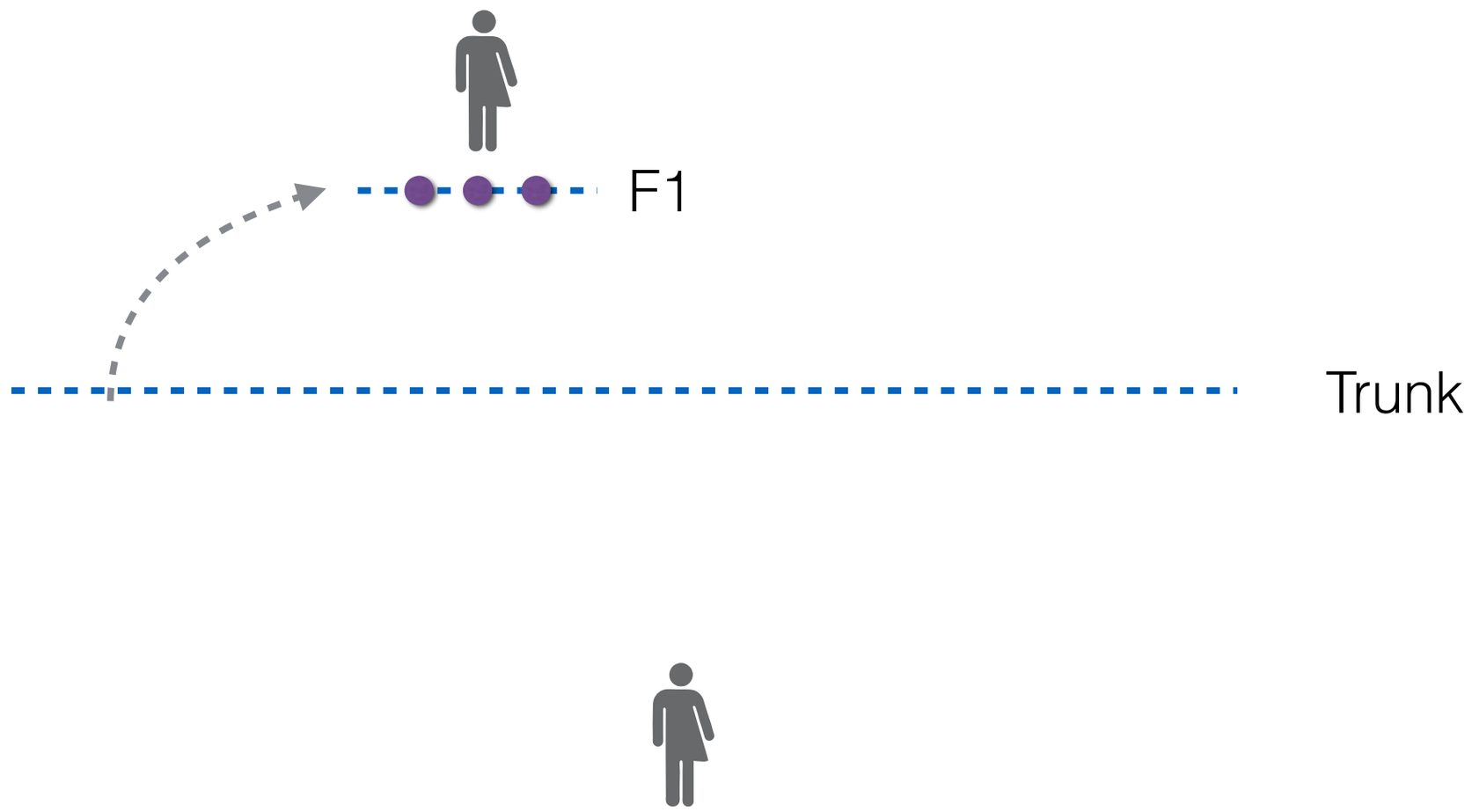


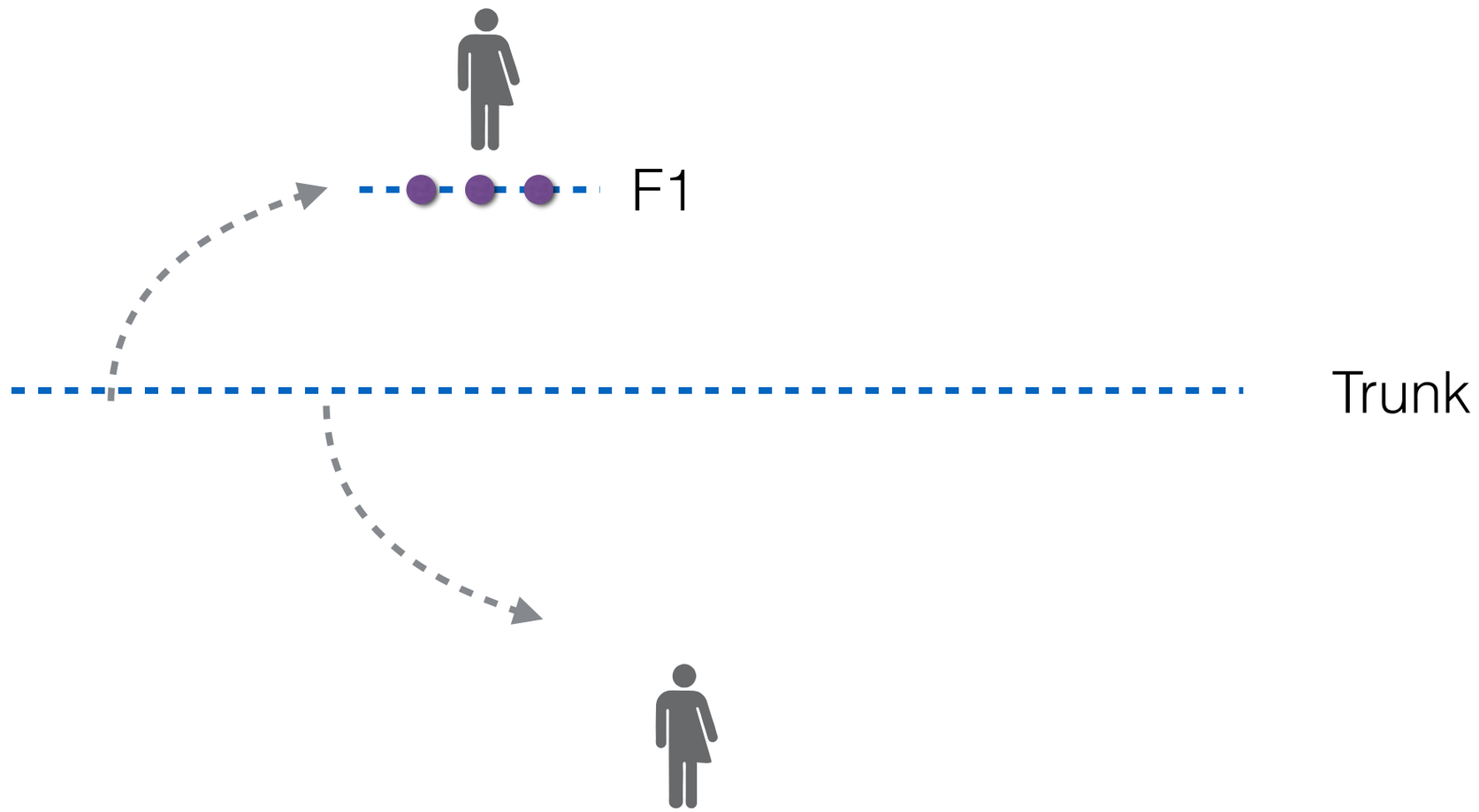
Trunk

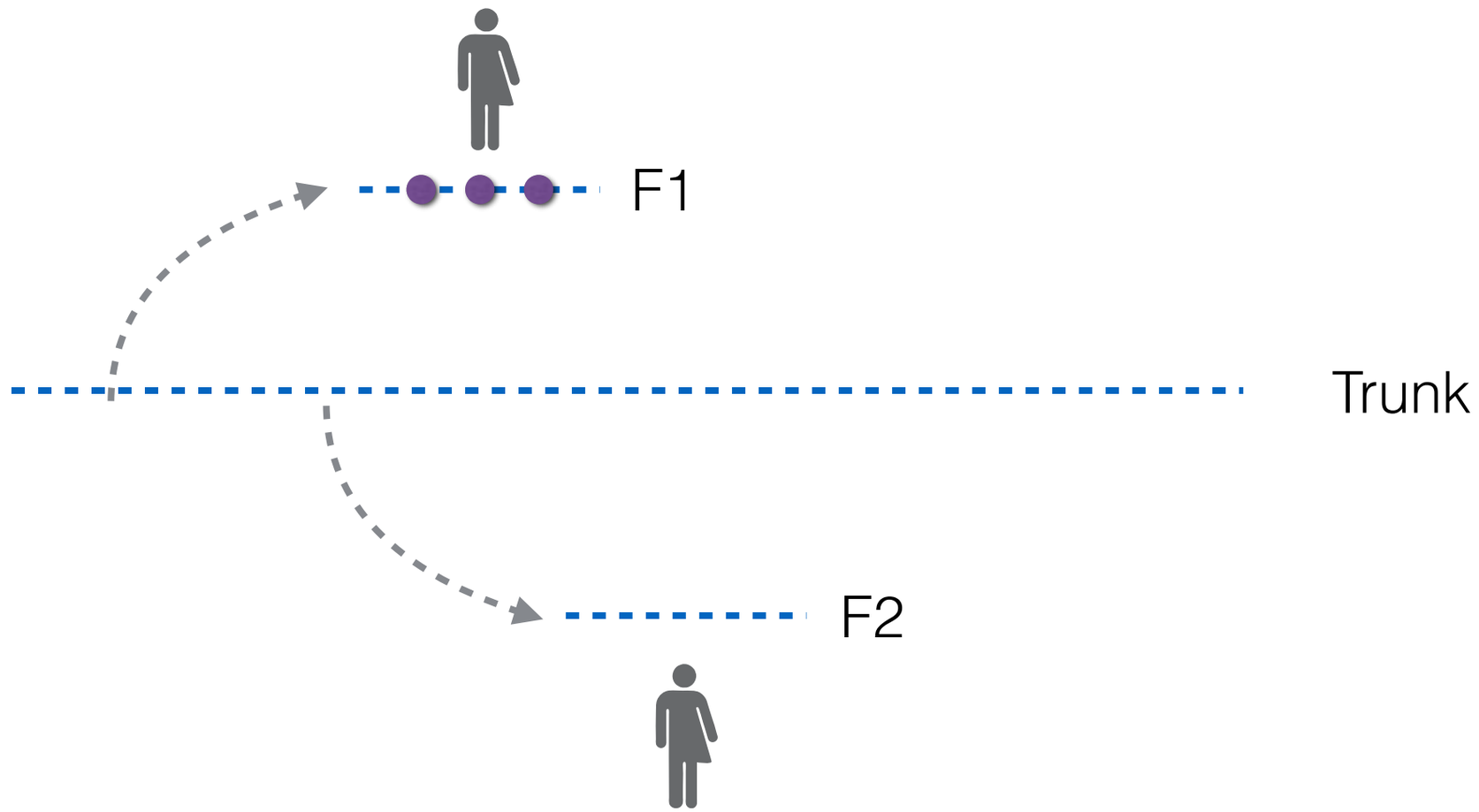


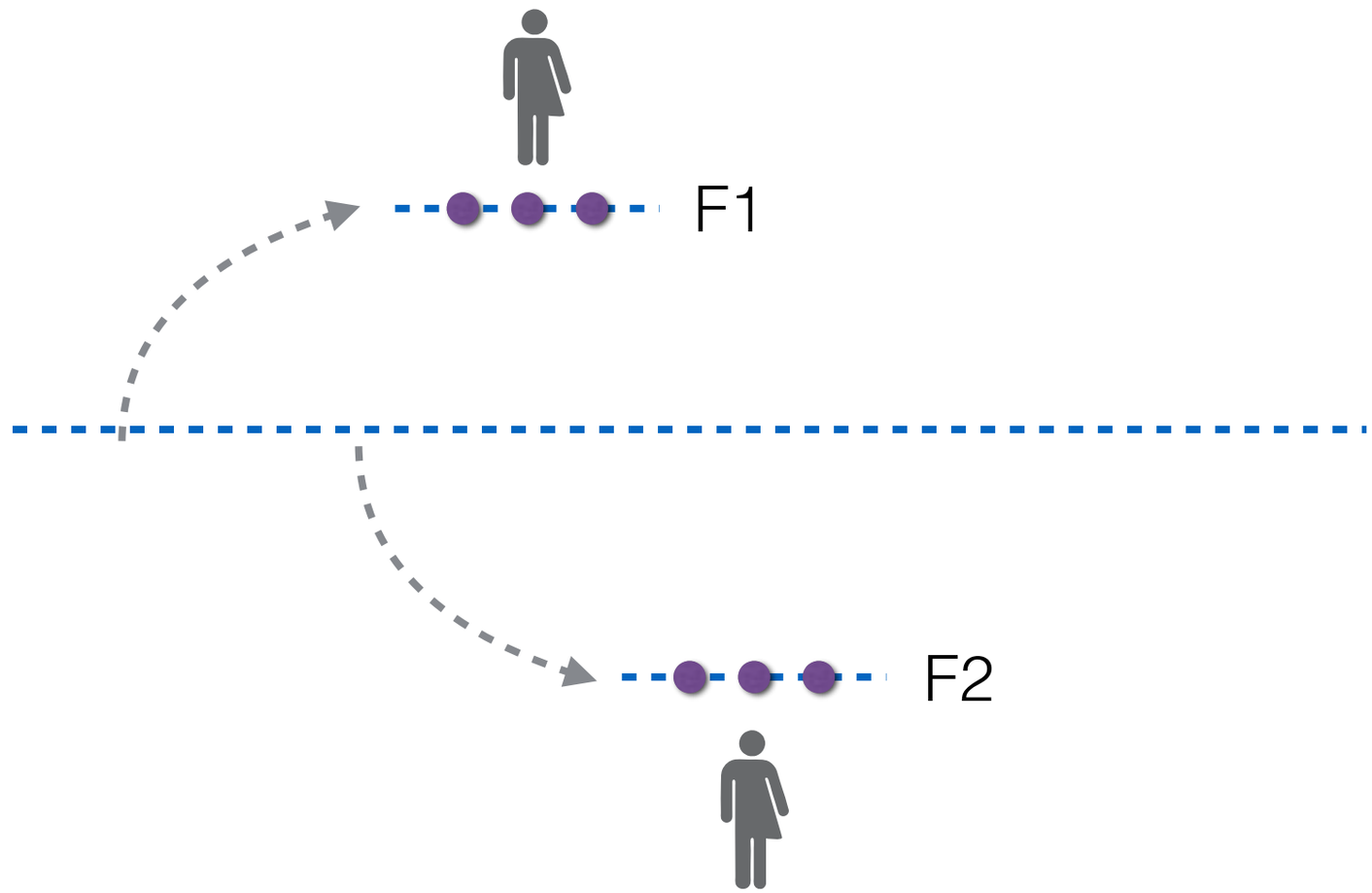

Trunk

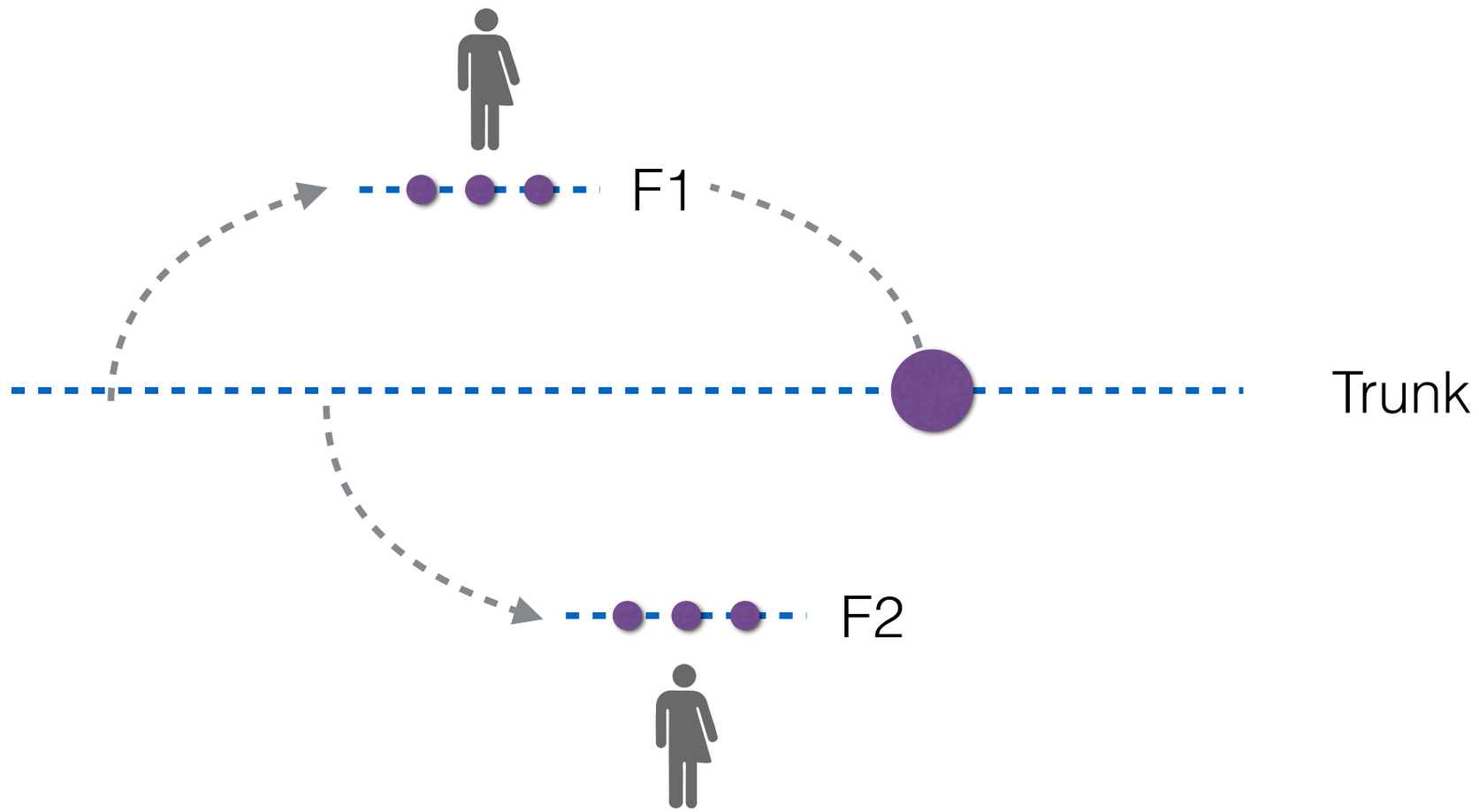

F1
Trunk

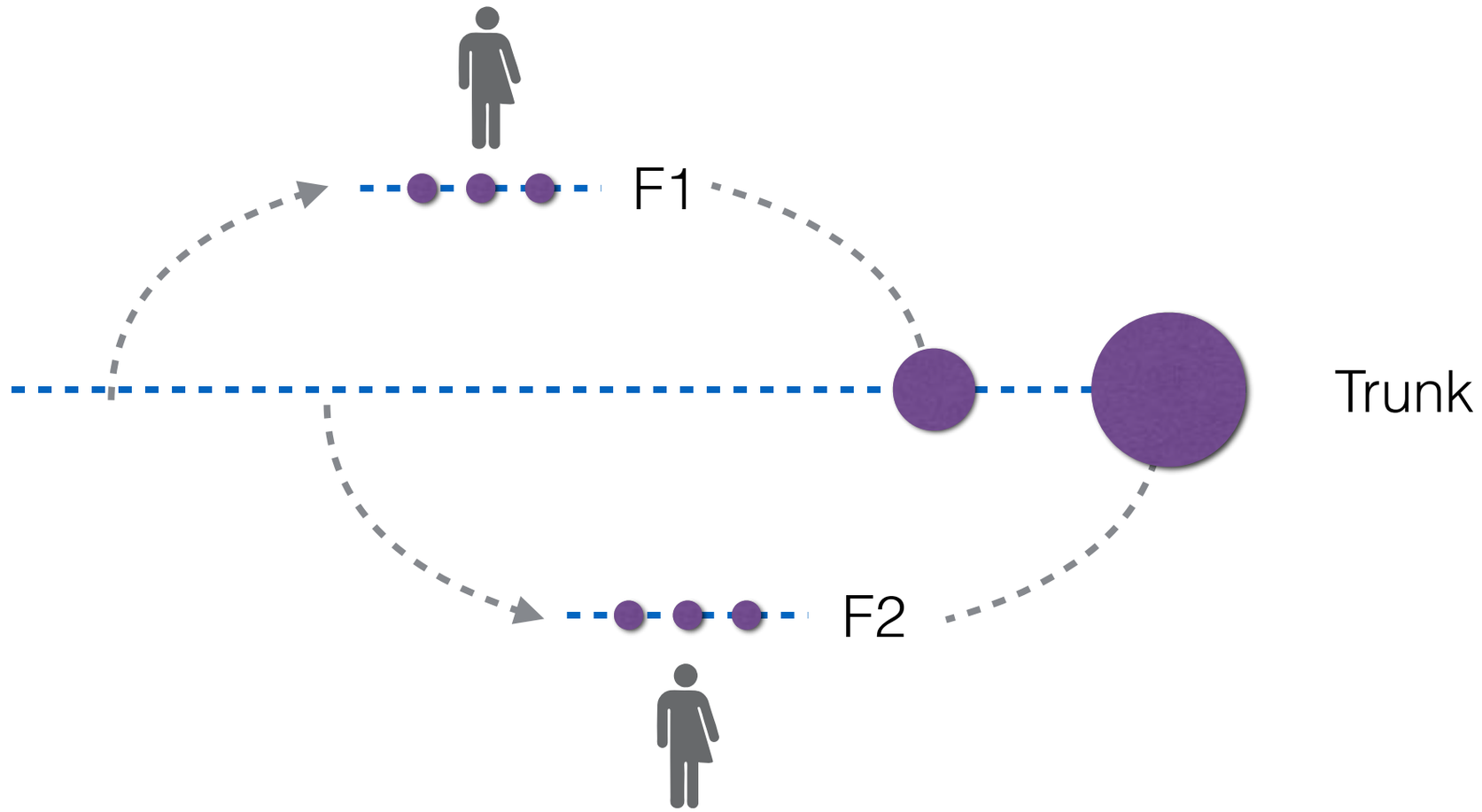

F1
Trunk

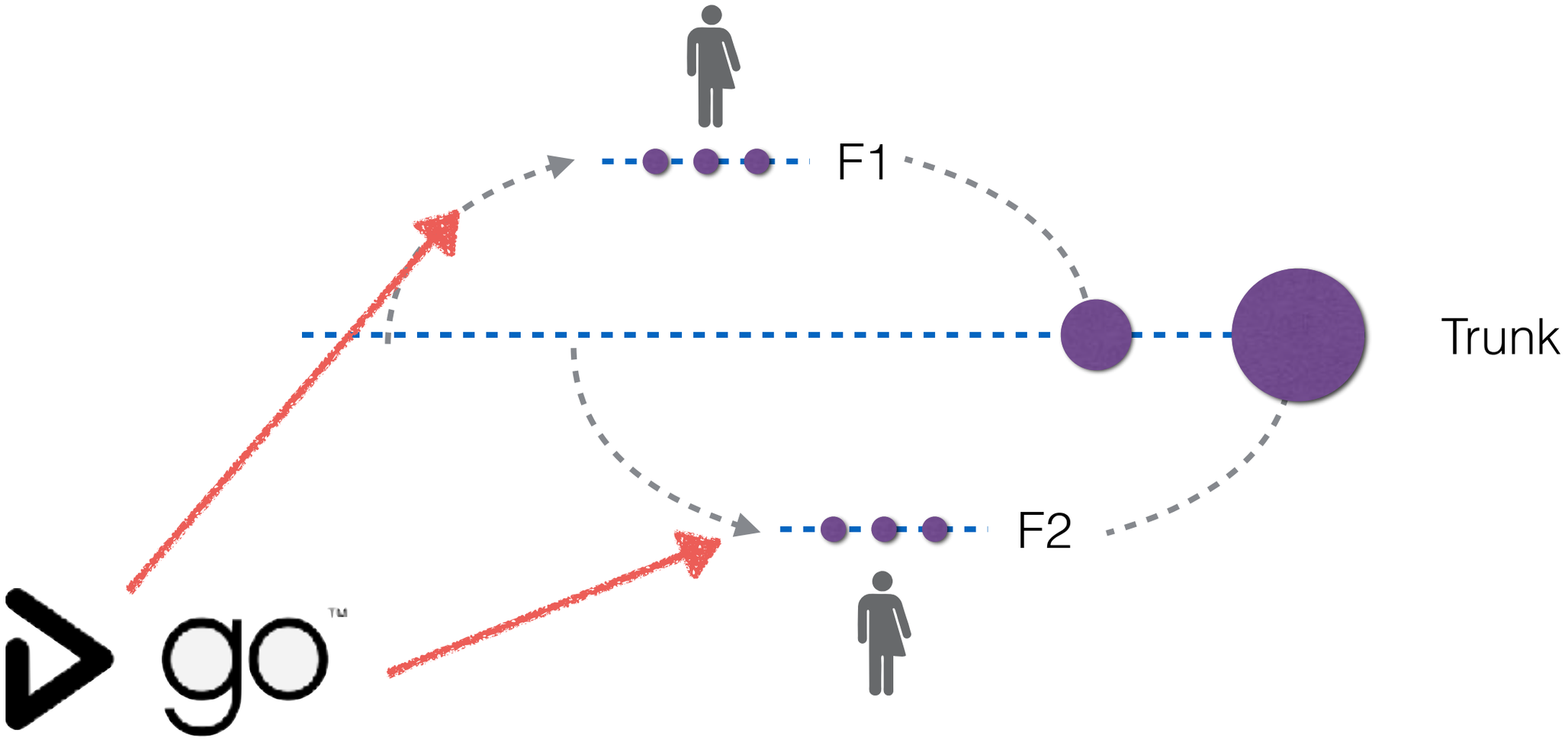

F1
Trunk

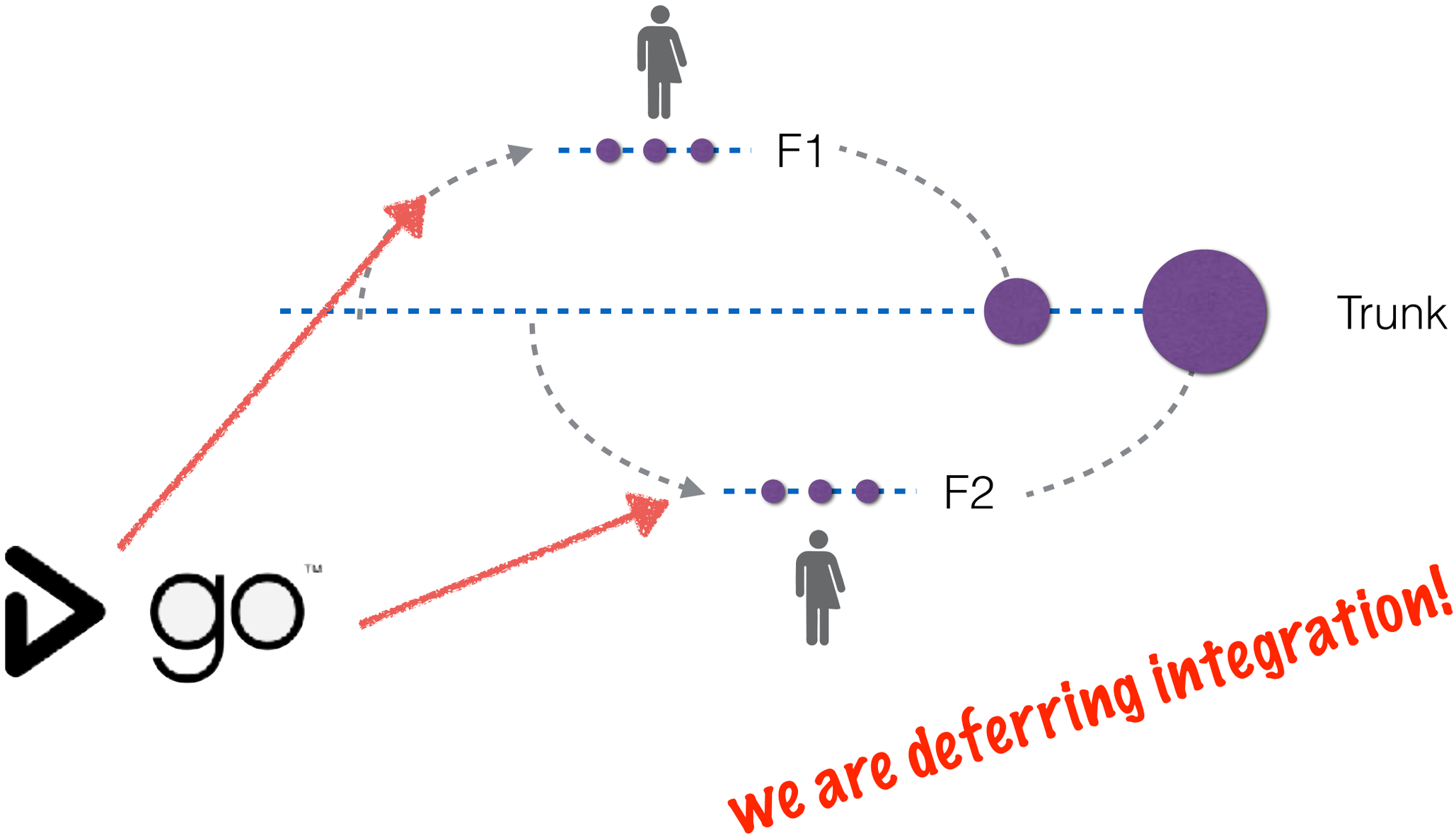

F1
Trunk
F2


F1
Trunk
F2


F1
Trunk
F2


F1
Trunk
F2


F1
Trunk
F2
go


F1
Trunk
F2
go
we are deferring integration!

Pain of merge = fn (
size_of_merge,
duration_since_last_merge)

Big merges = commit race!

Merging refactoring is *really* hard

@gotochi

@samnewman

Option 3:

Option 3:

Check in anyway

Option 3:

Check in anyway

err...wat?

Trunk-based development

@gotochi

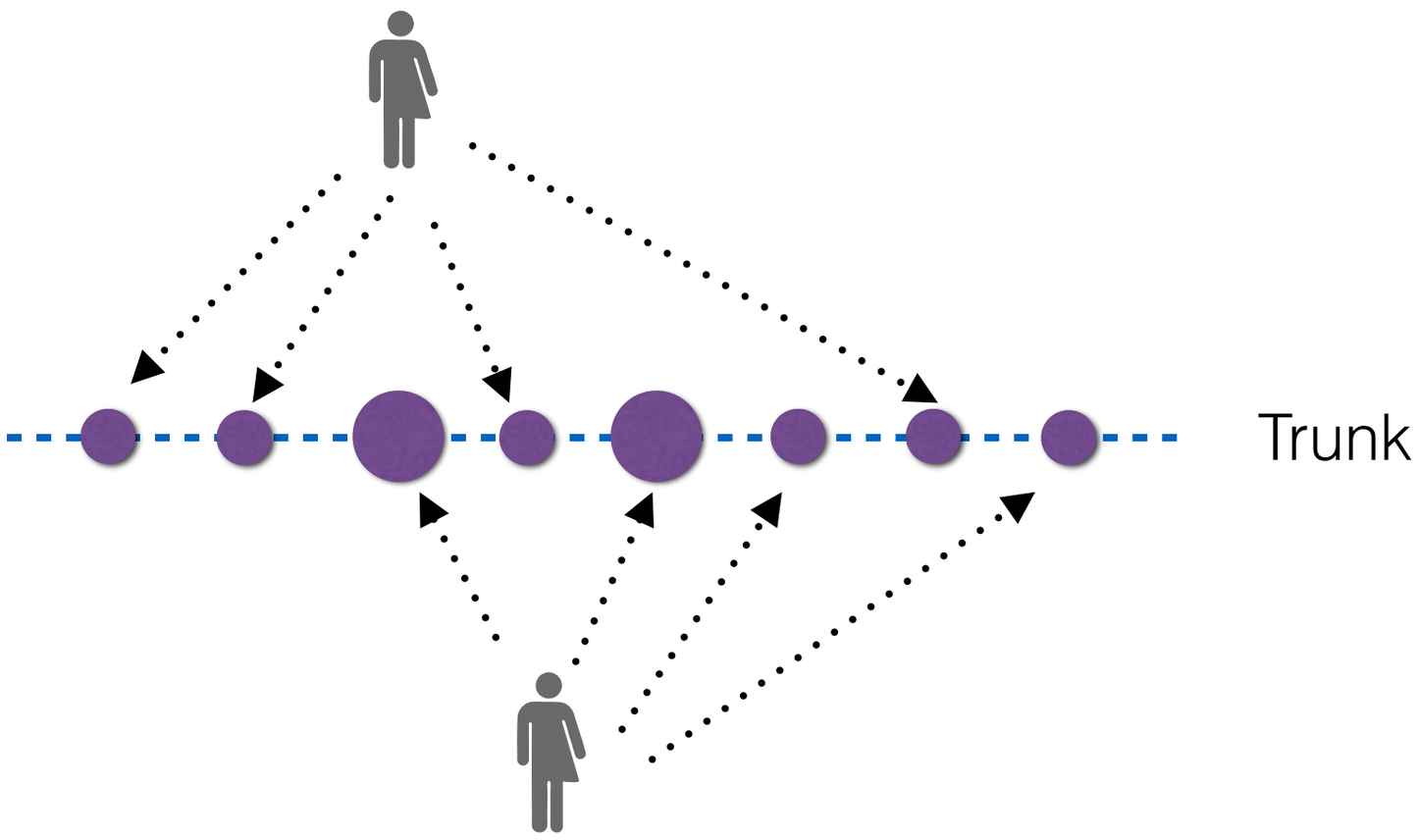
@samnewman

Everyone integrates into trunk

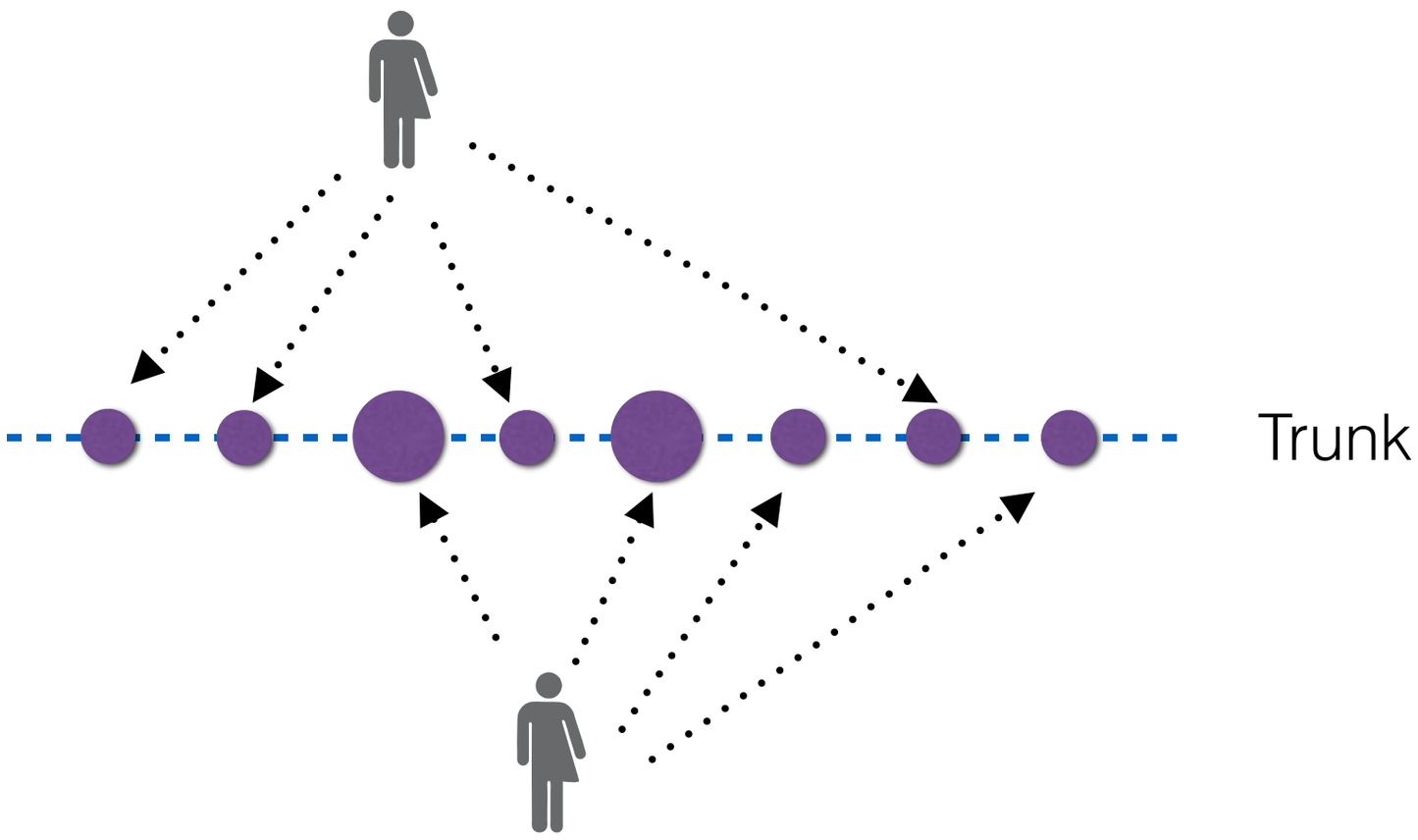


Trunk



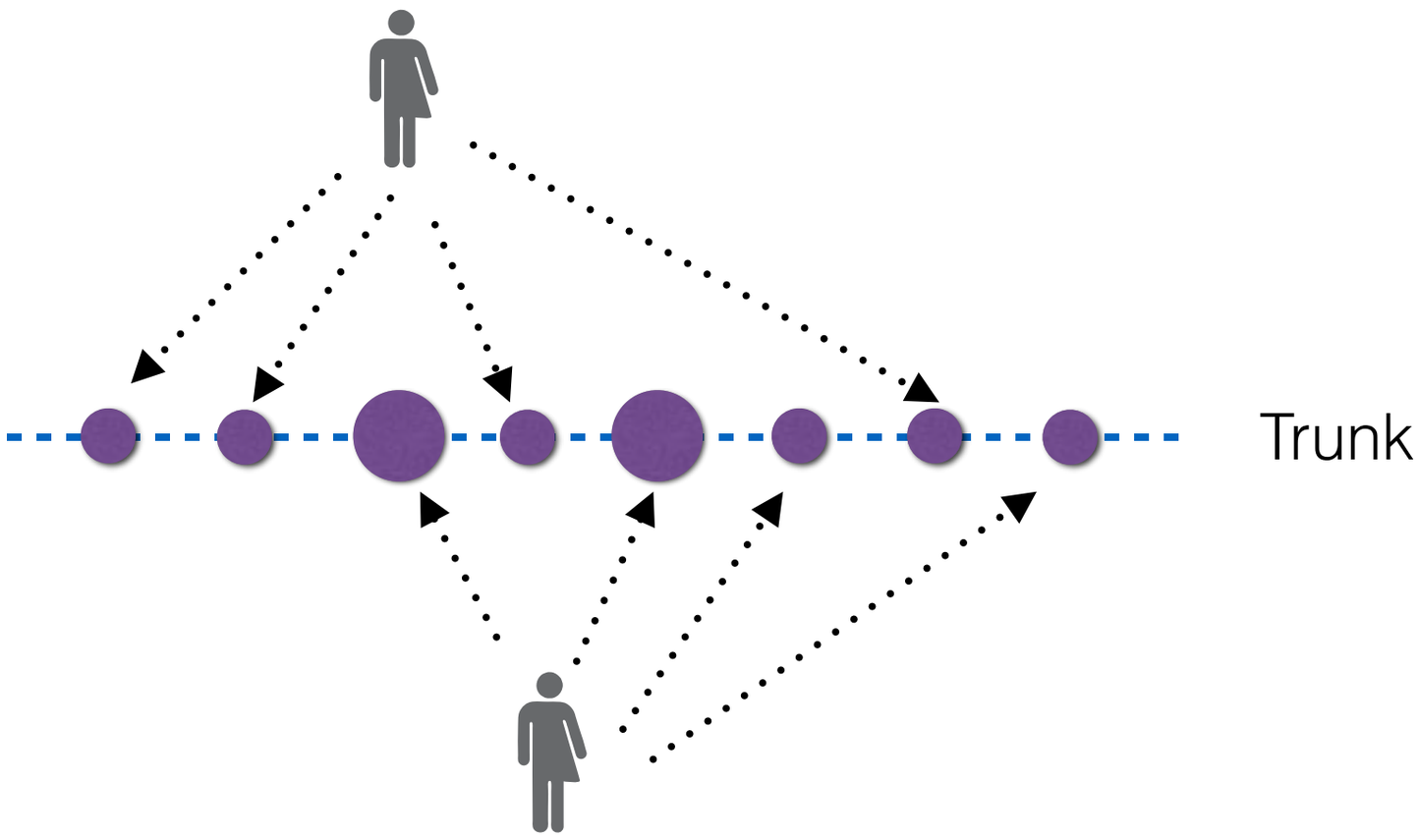

Trunk

integrate often...



integrate often...

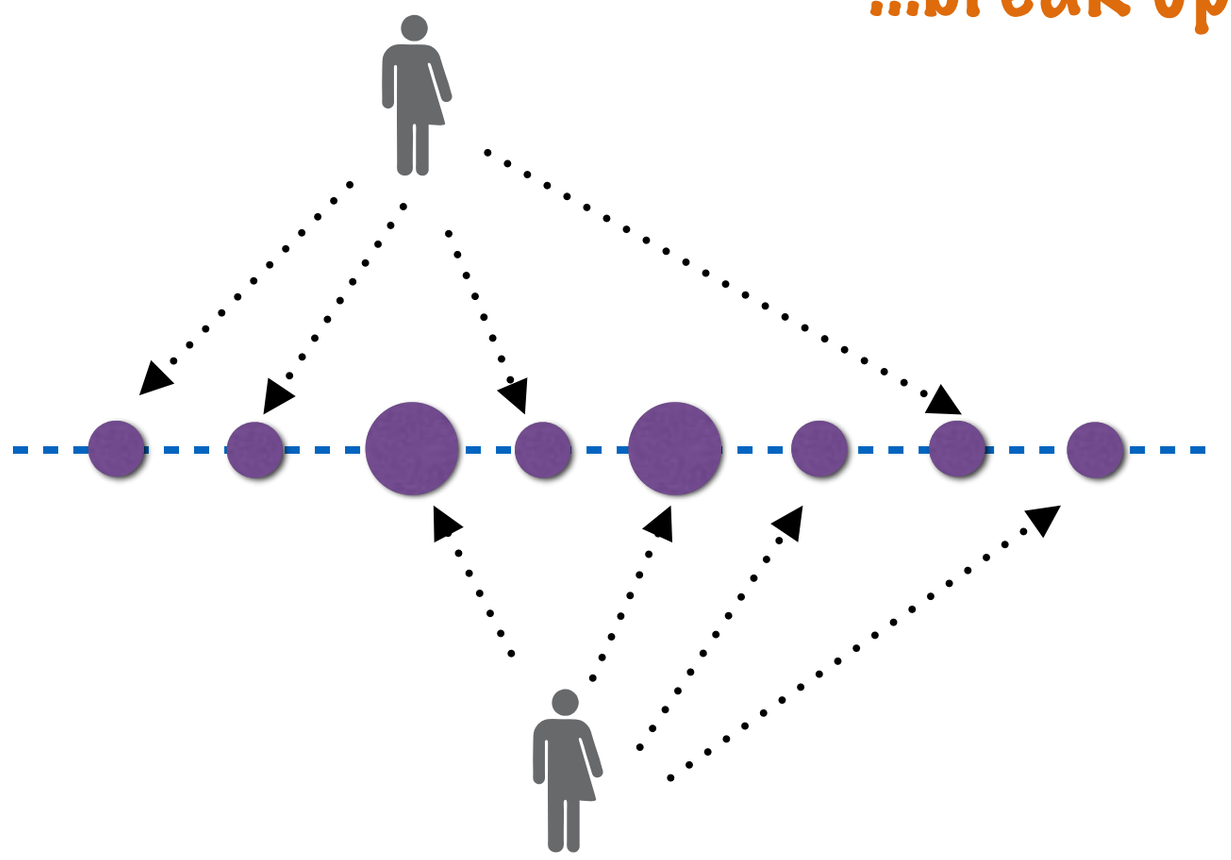
...fast feedback...



integrate often...

...fast feedback...

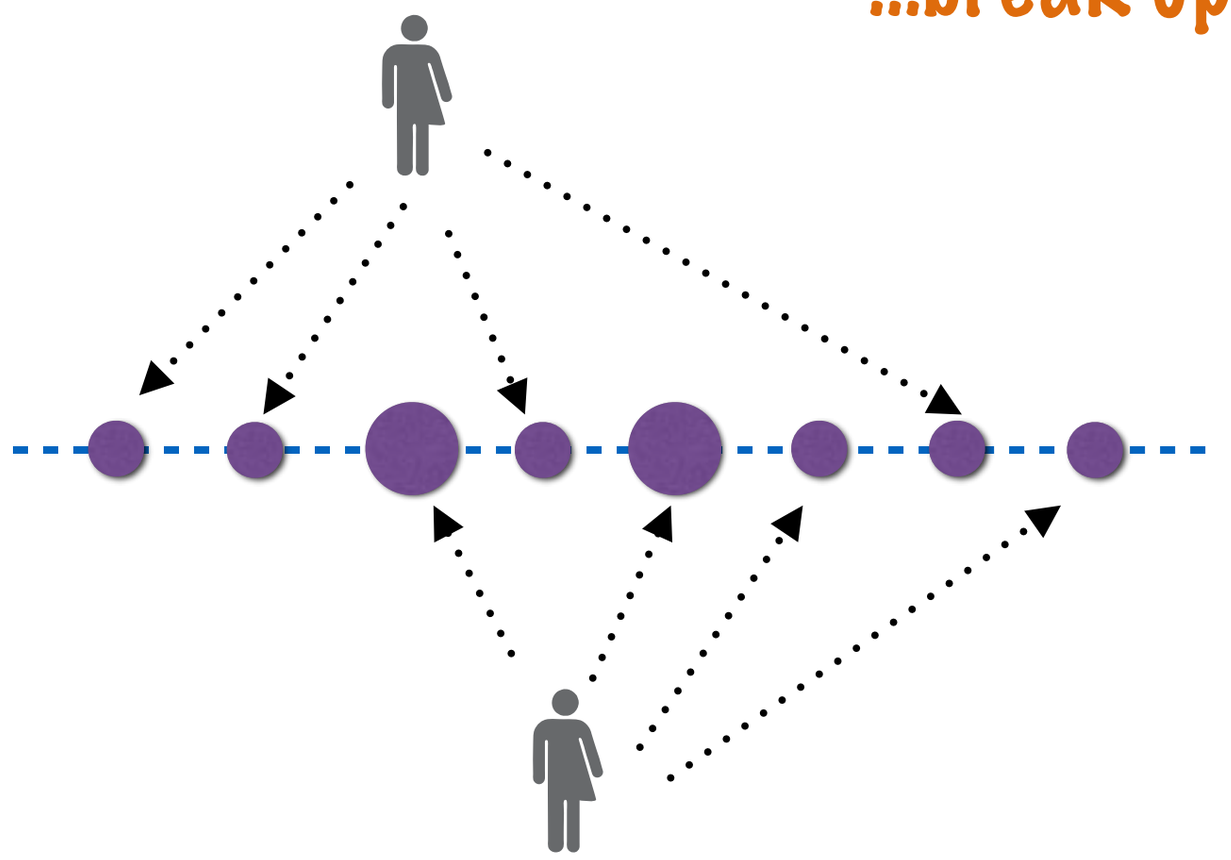
...break up merge pain



integrate often...

...fast feedback...

...break up merge pain



err...but what about half-finished features?

Feature Toggles

@gotochi

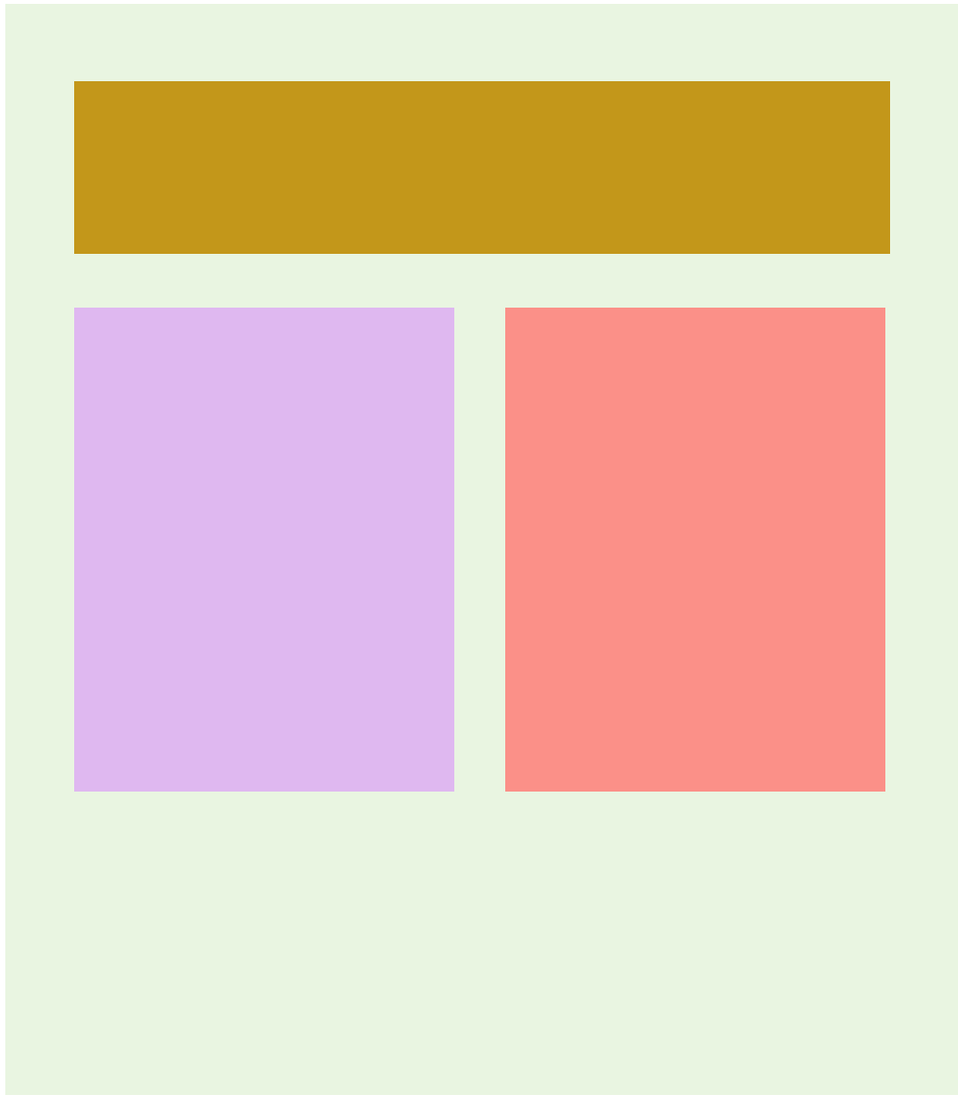
@samnewman

Feature Toggles

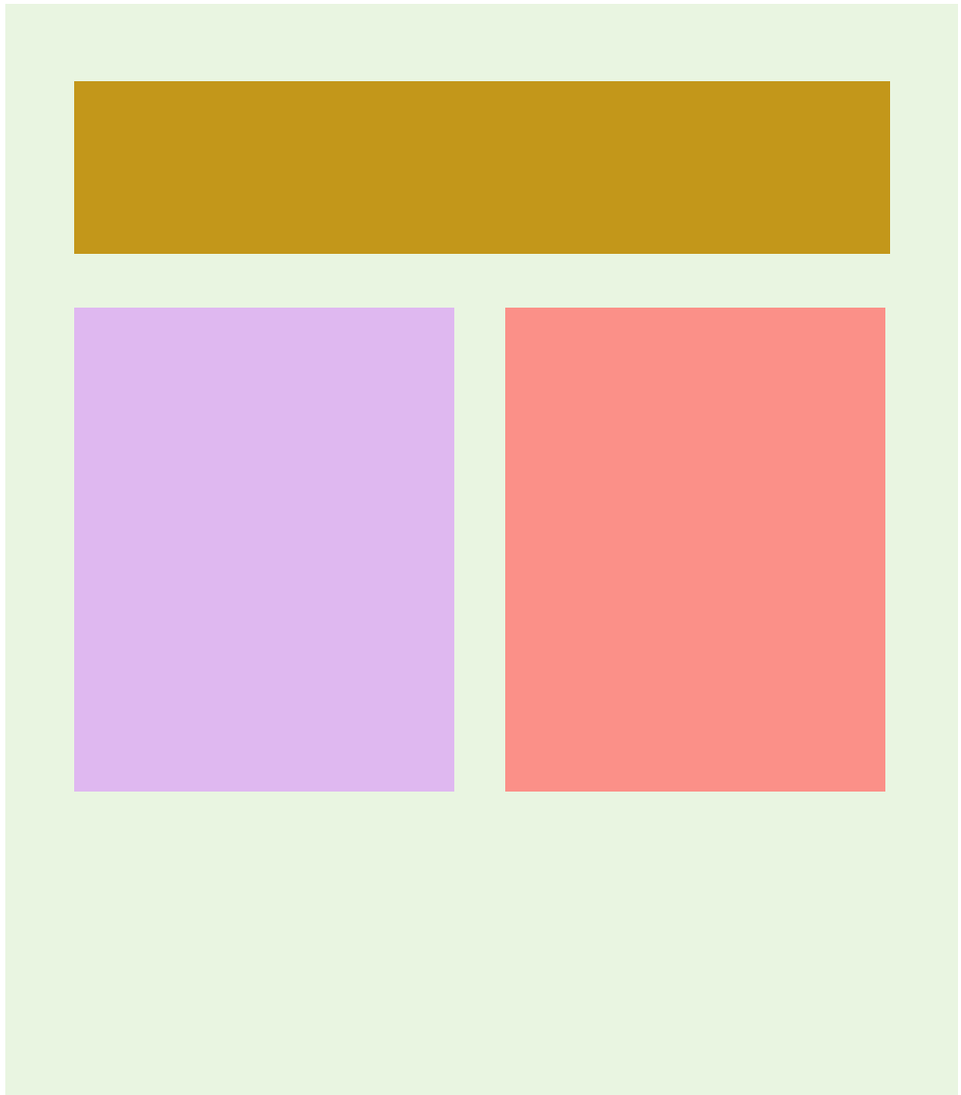
Hide the partially implemented feature in the running system

@gotochi

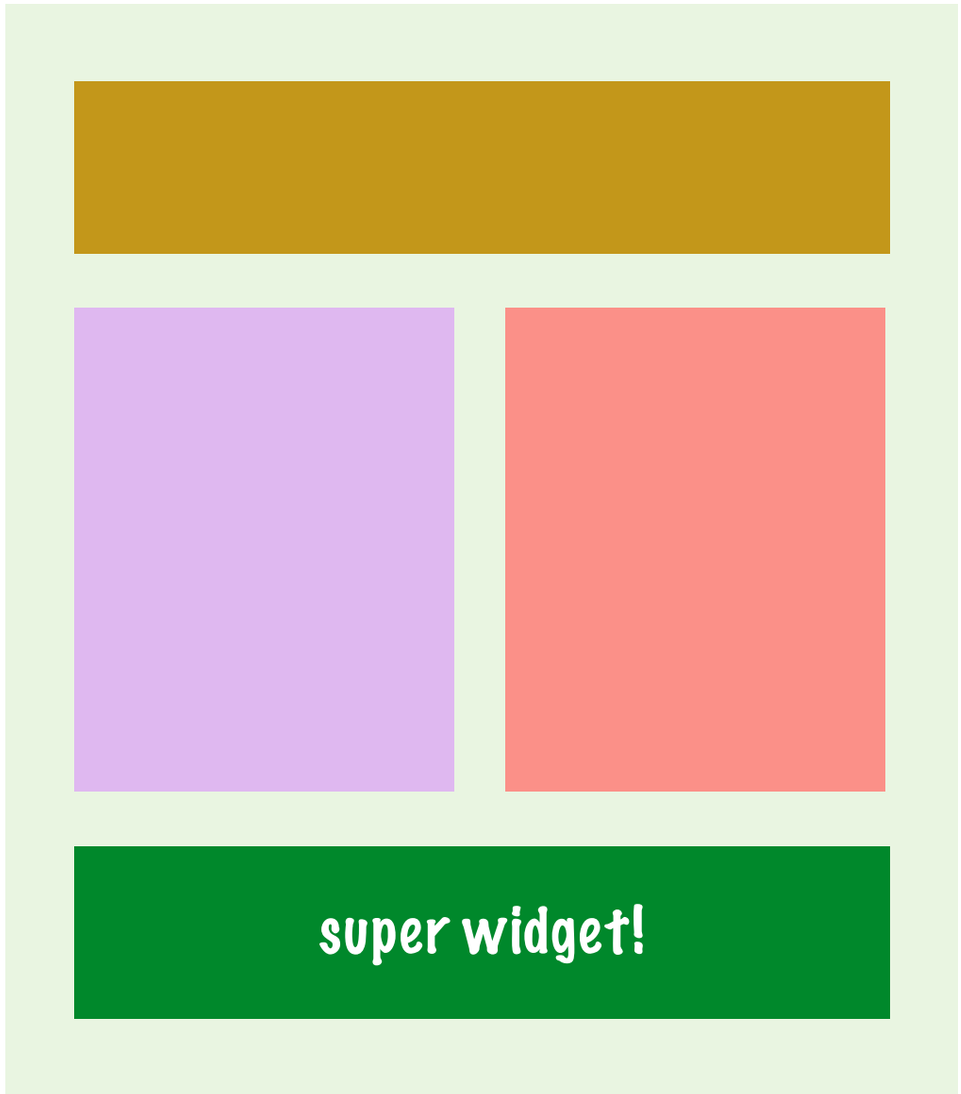
@samnewman



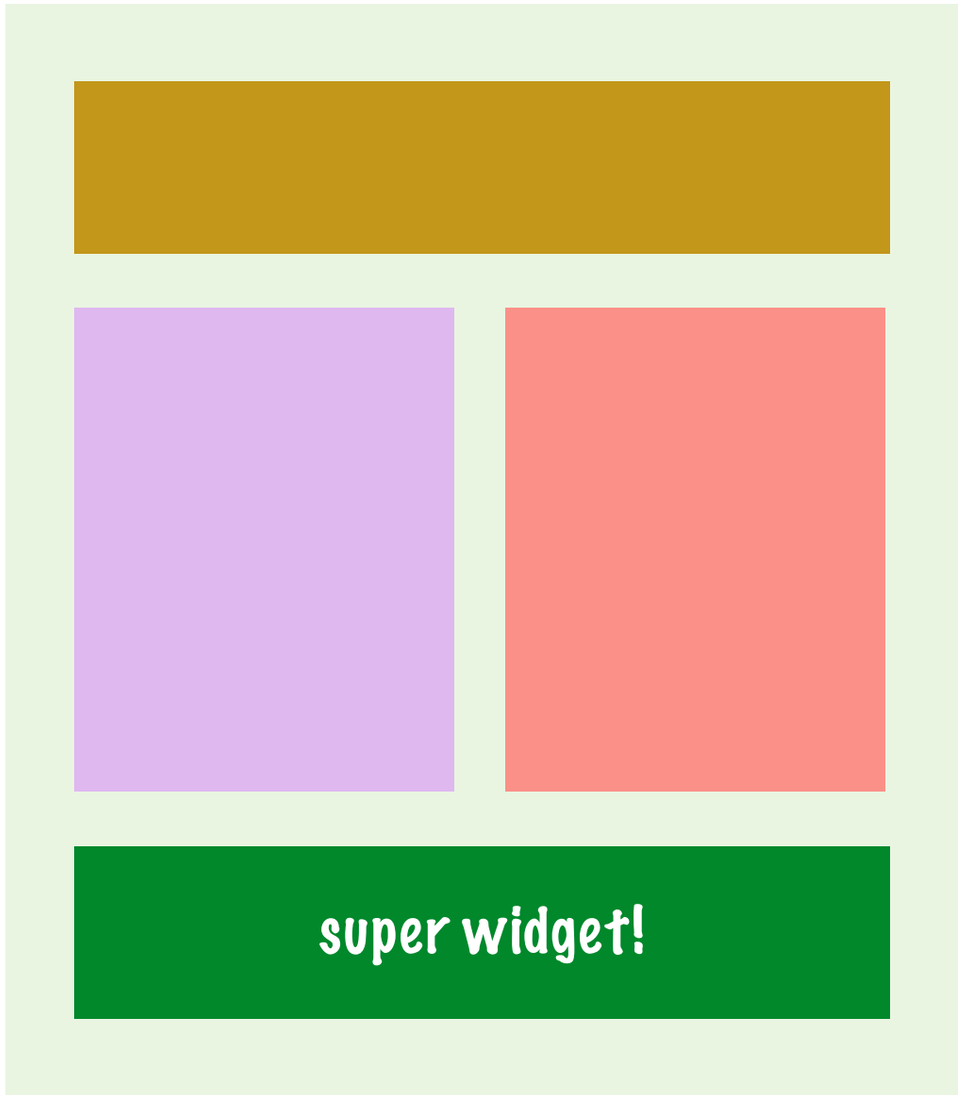
```
…
 super_widget = off
…
```



```
…
  super_widget = on
…
```

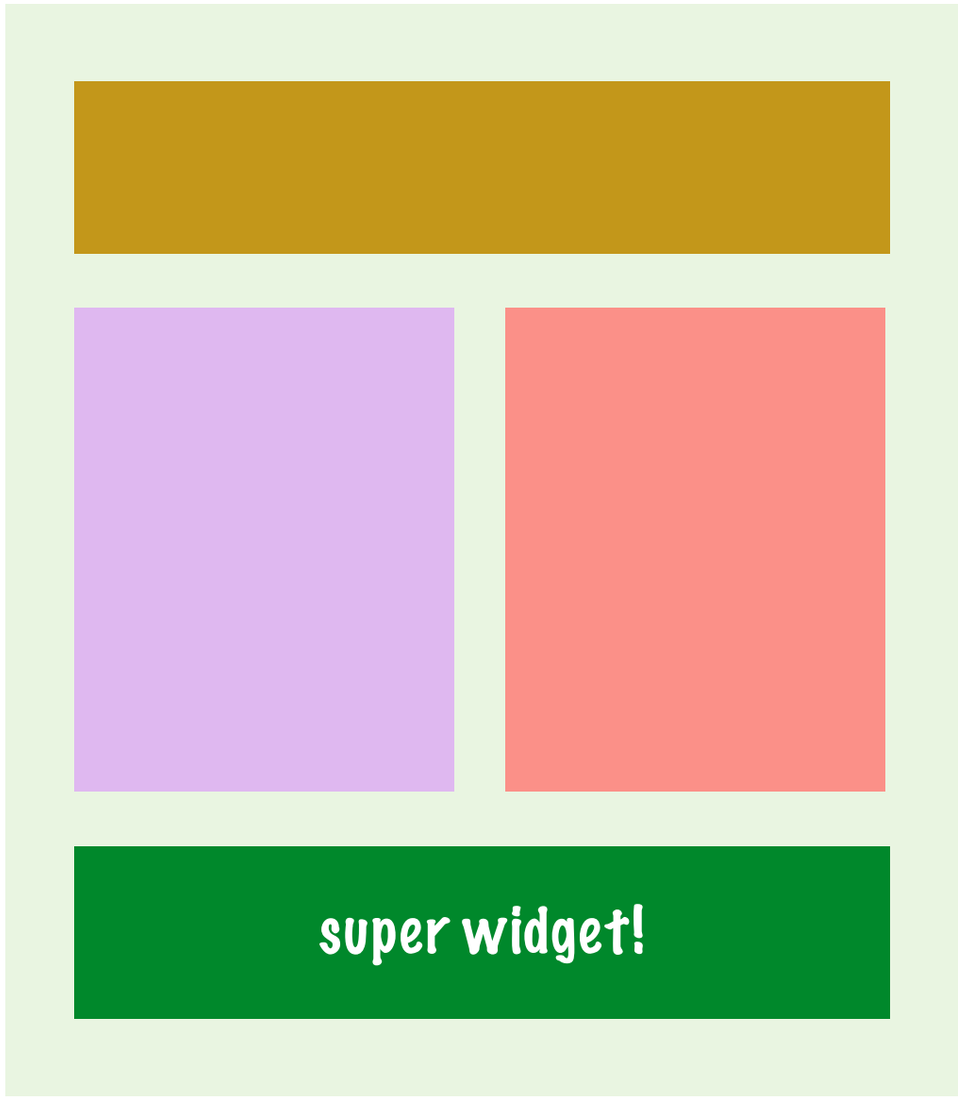



```
…
  super_widget = on
…
```




```
…
 super_widget = on
…
```

```
$. run -Dsuper_widget=on
```



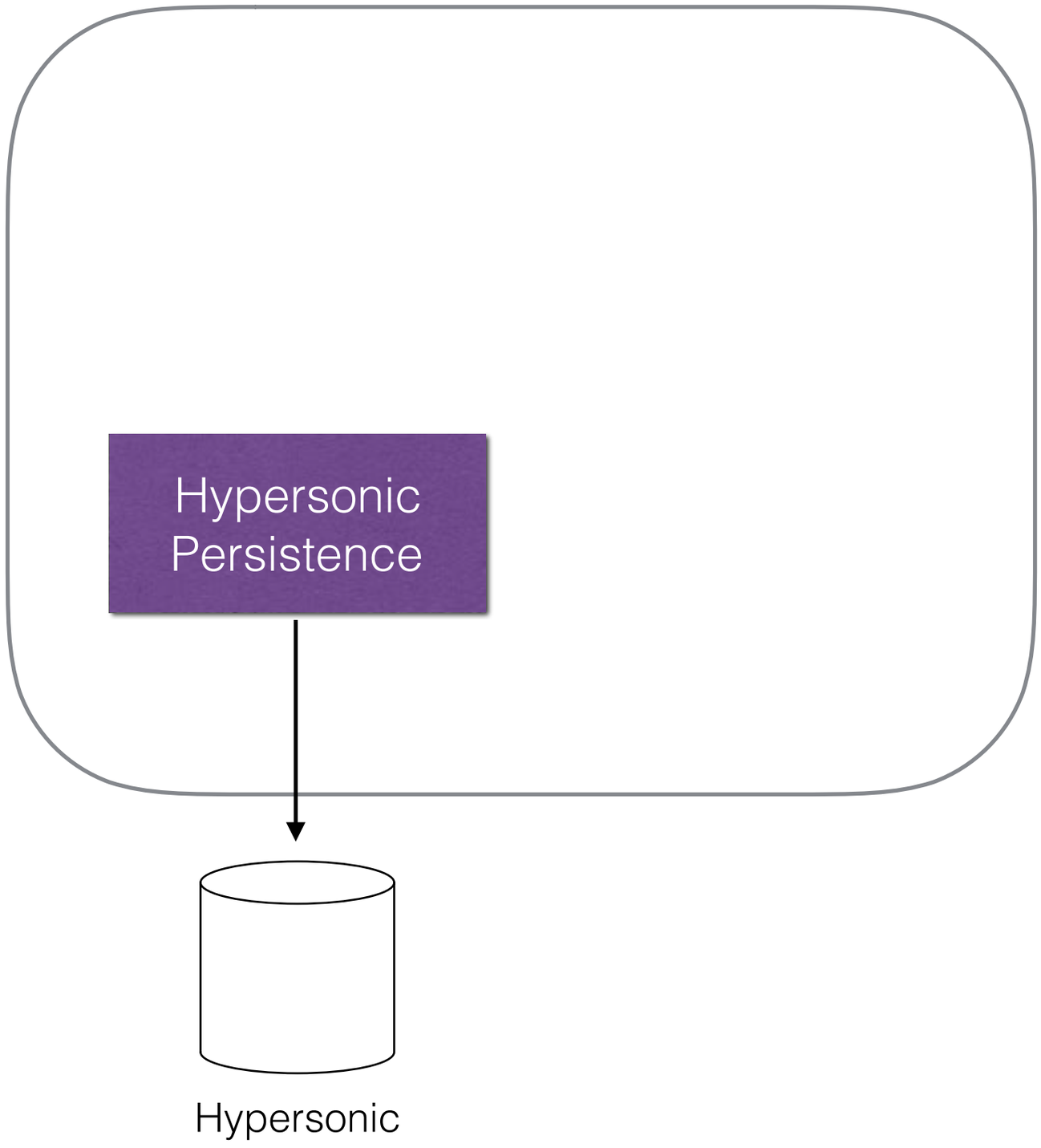

```
…
 super_widget = on
…
```

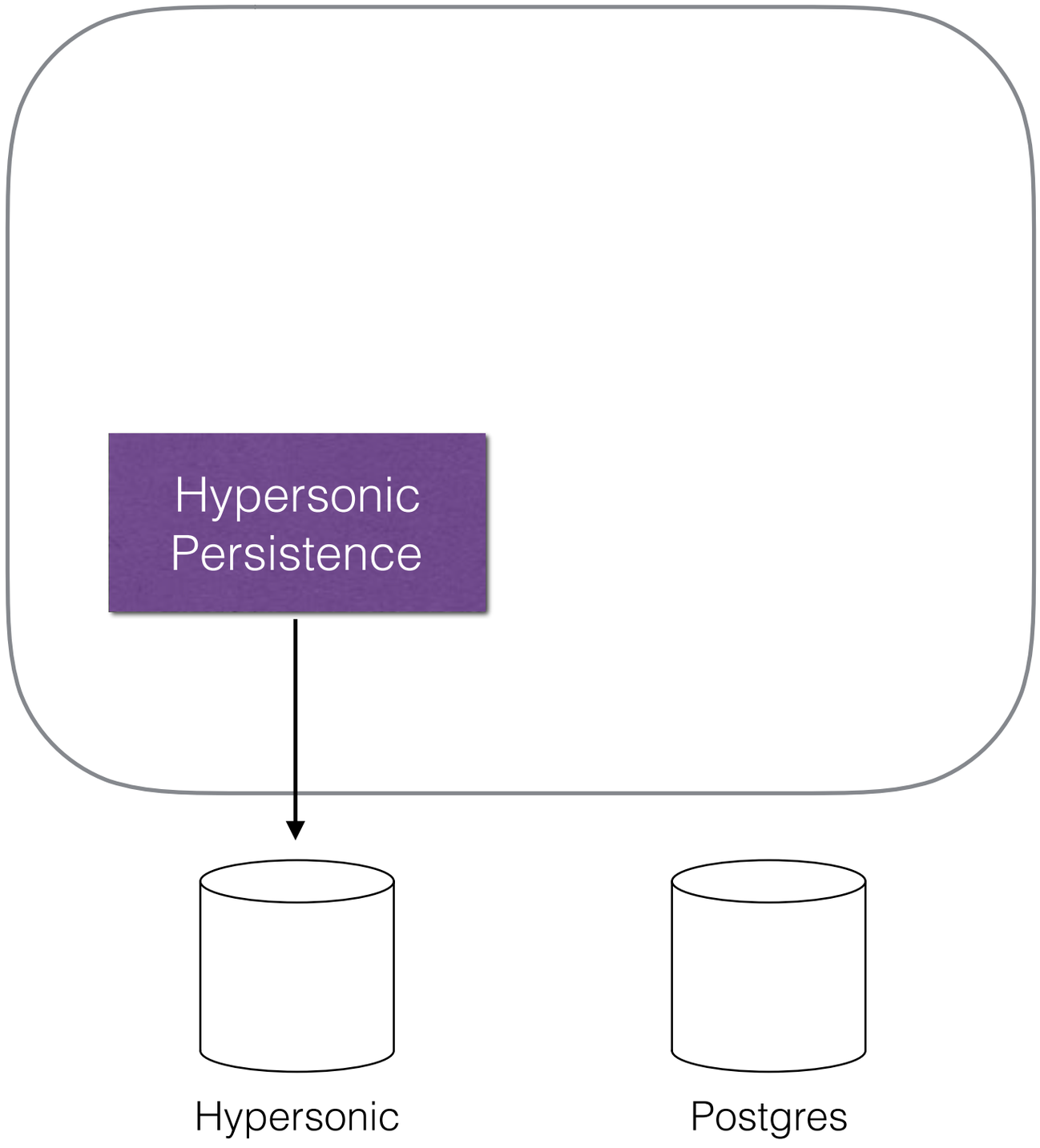
```
$. run -Dsuper_widget=on
```

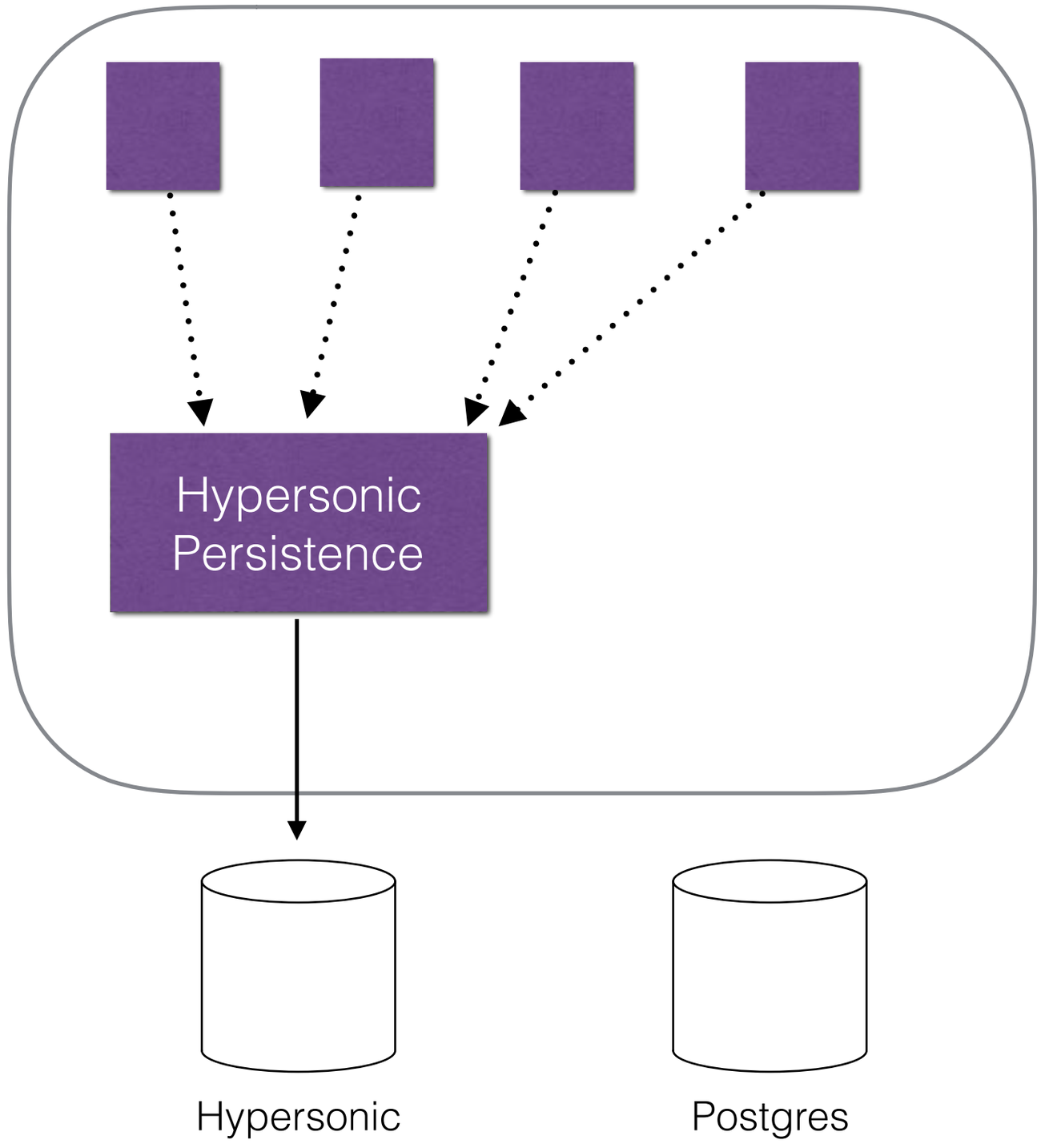


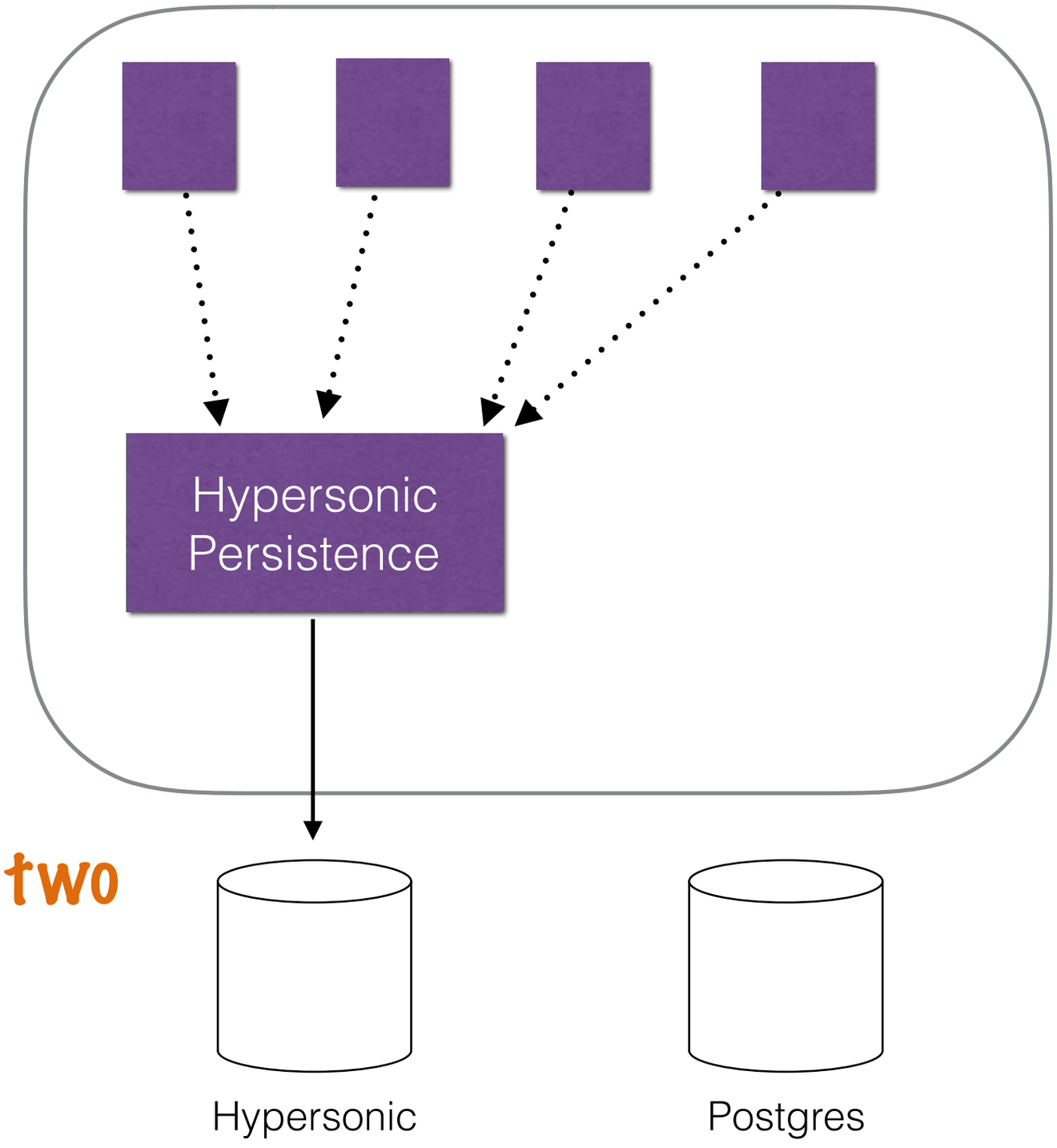
But what about changes to existing functionality?

Bigo™

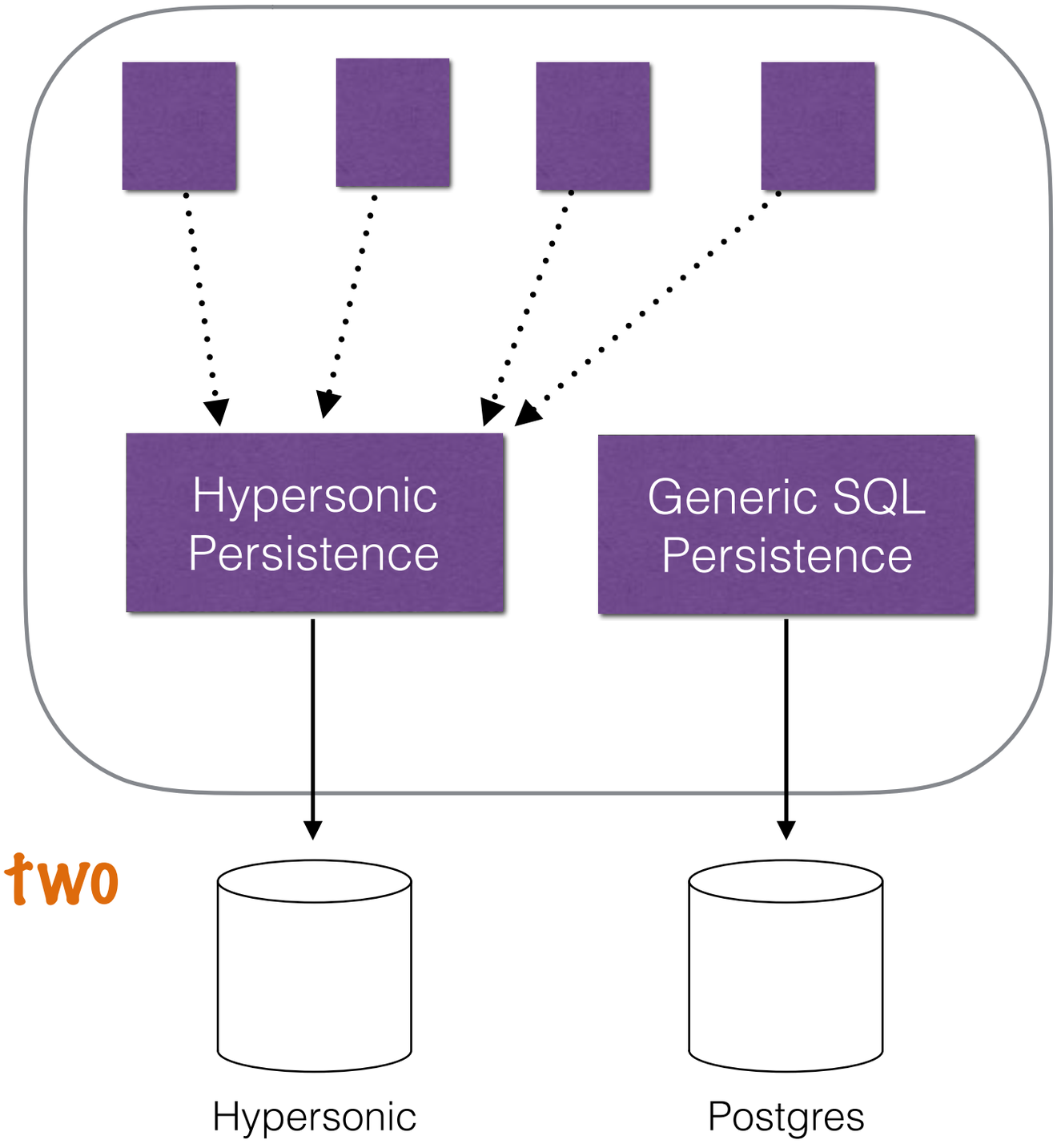

Hypersonic
Persistence
Hypersonic


Hypersonic
Persistence
Hypersonic
Postgres

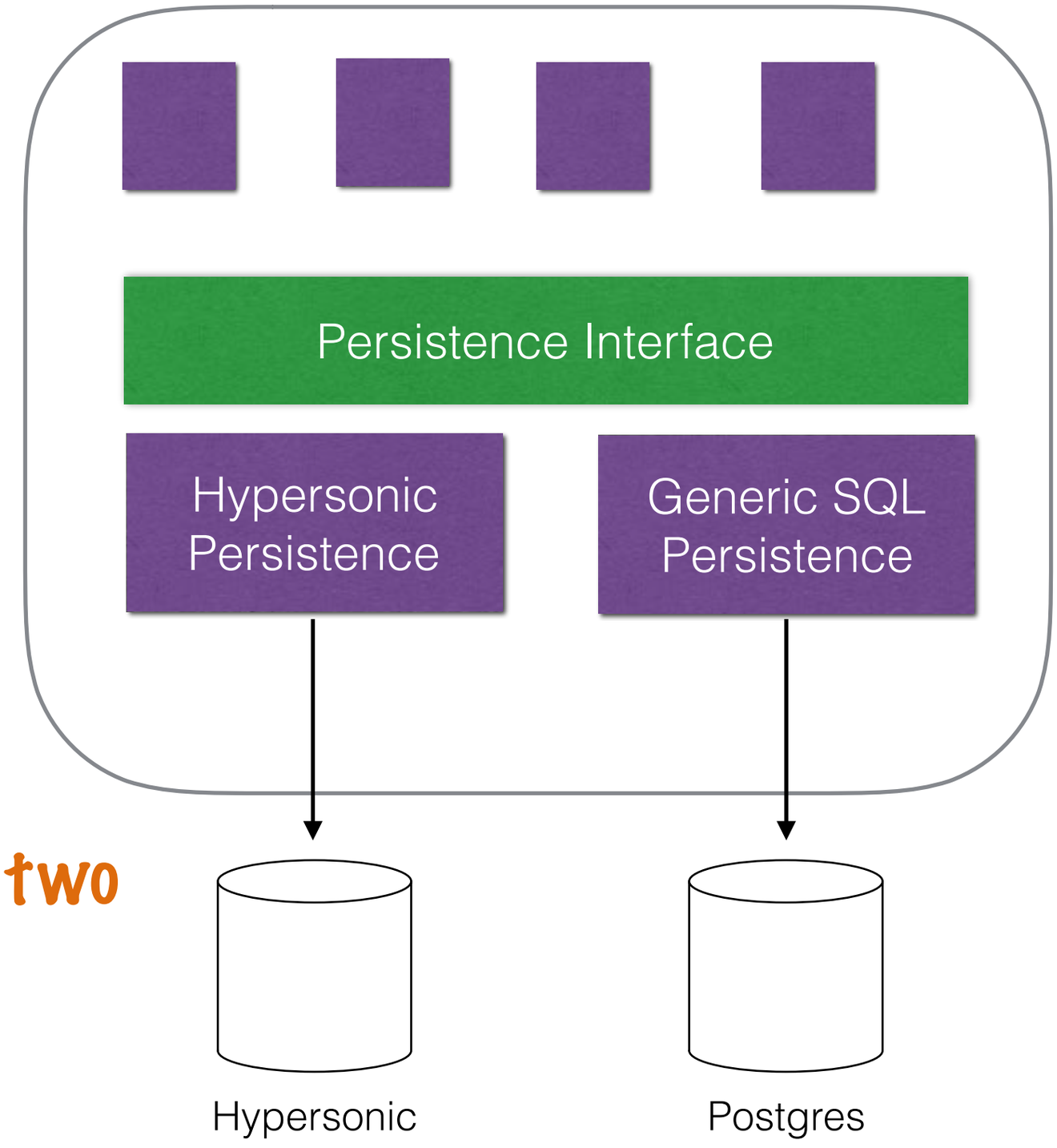

Hypersonic
Persistence
Hypersonic
Postgres



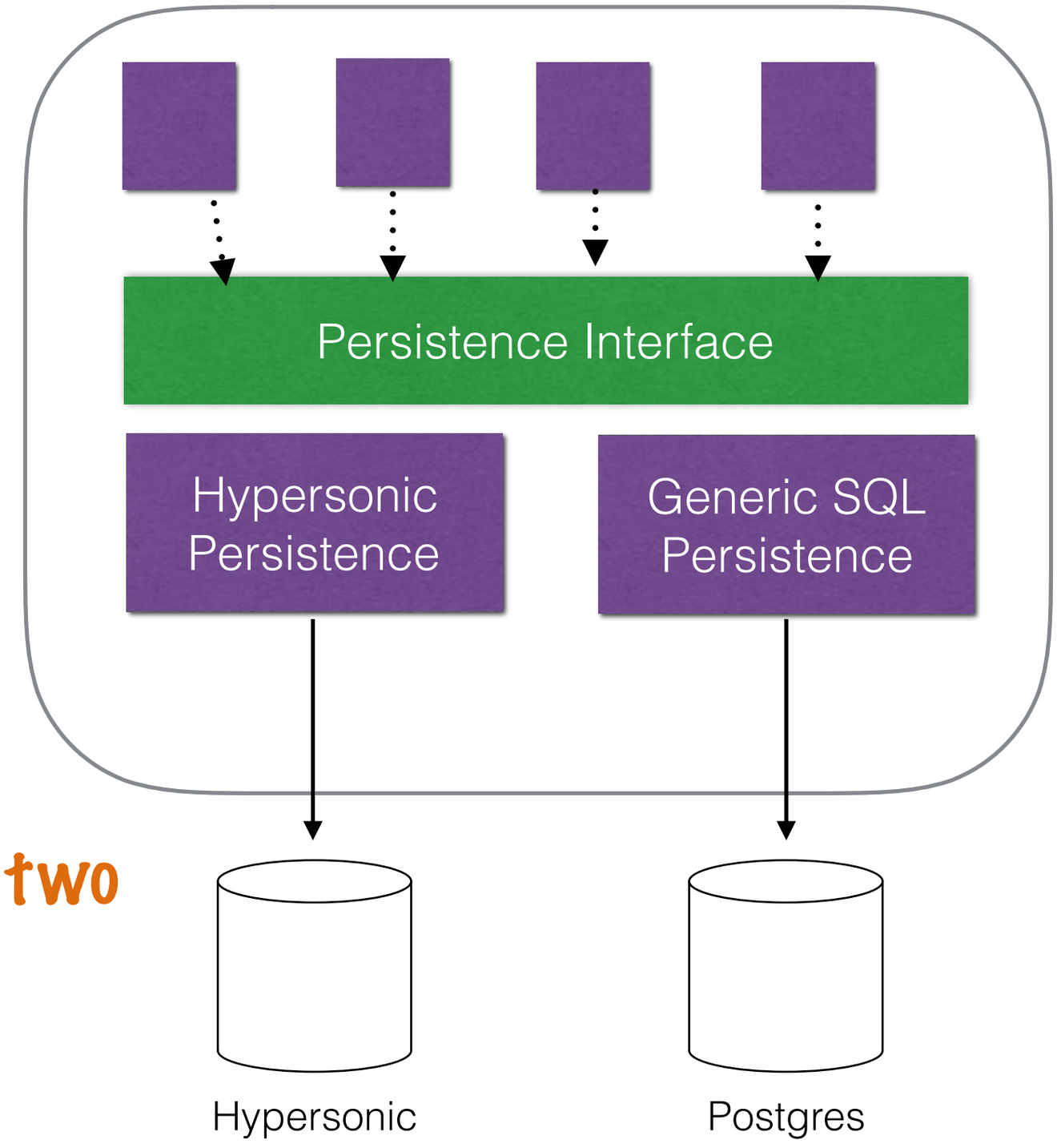
shipping every two weeks...



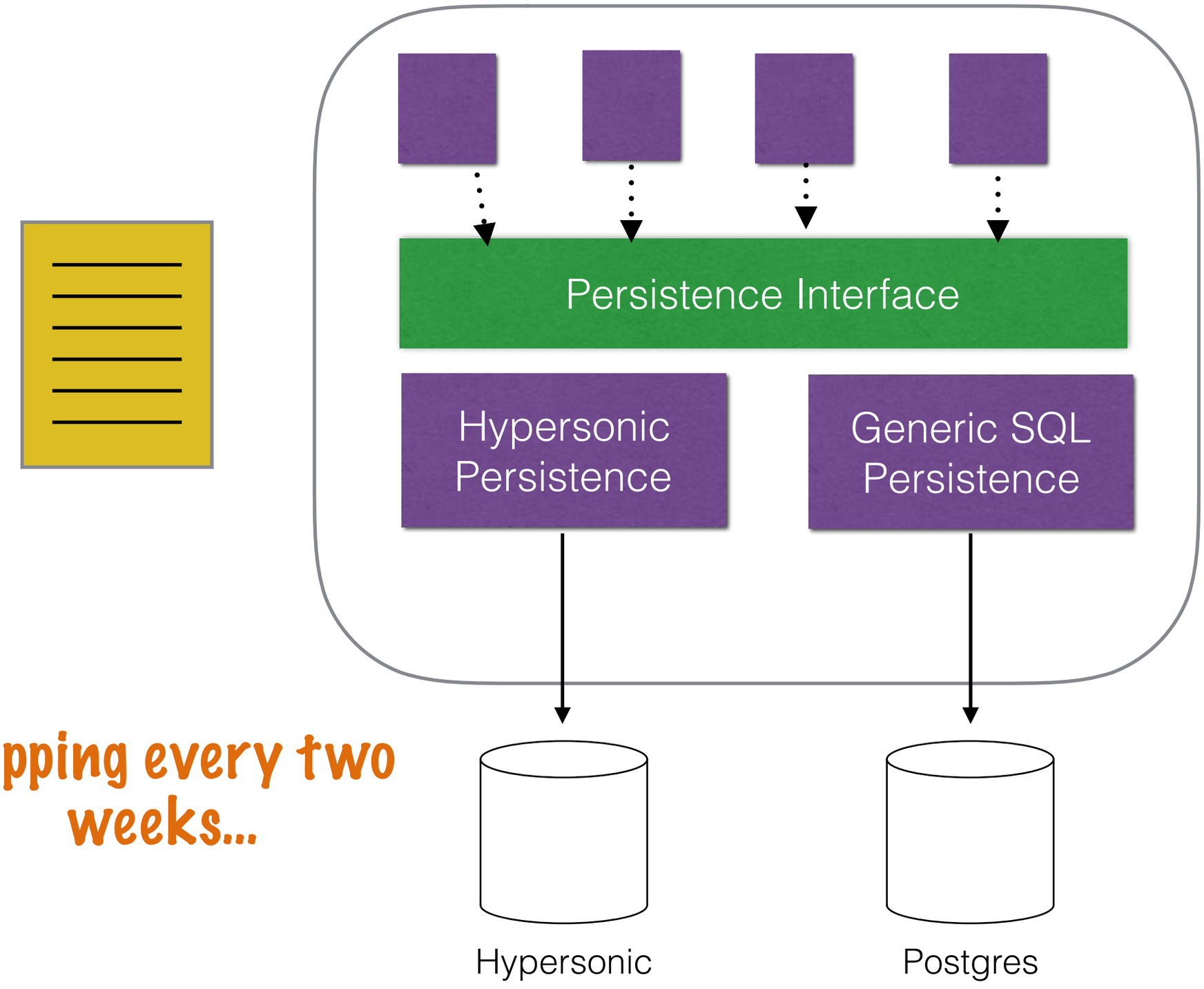
shipping every two weeks...



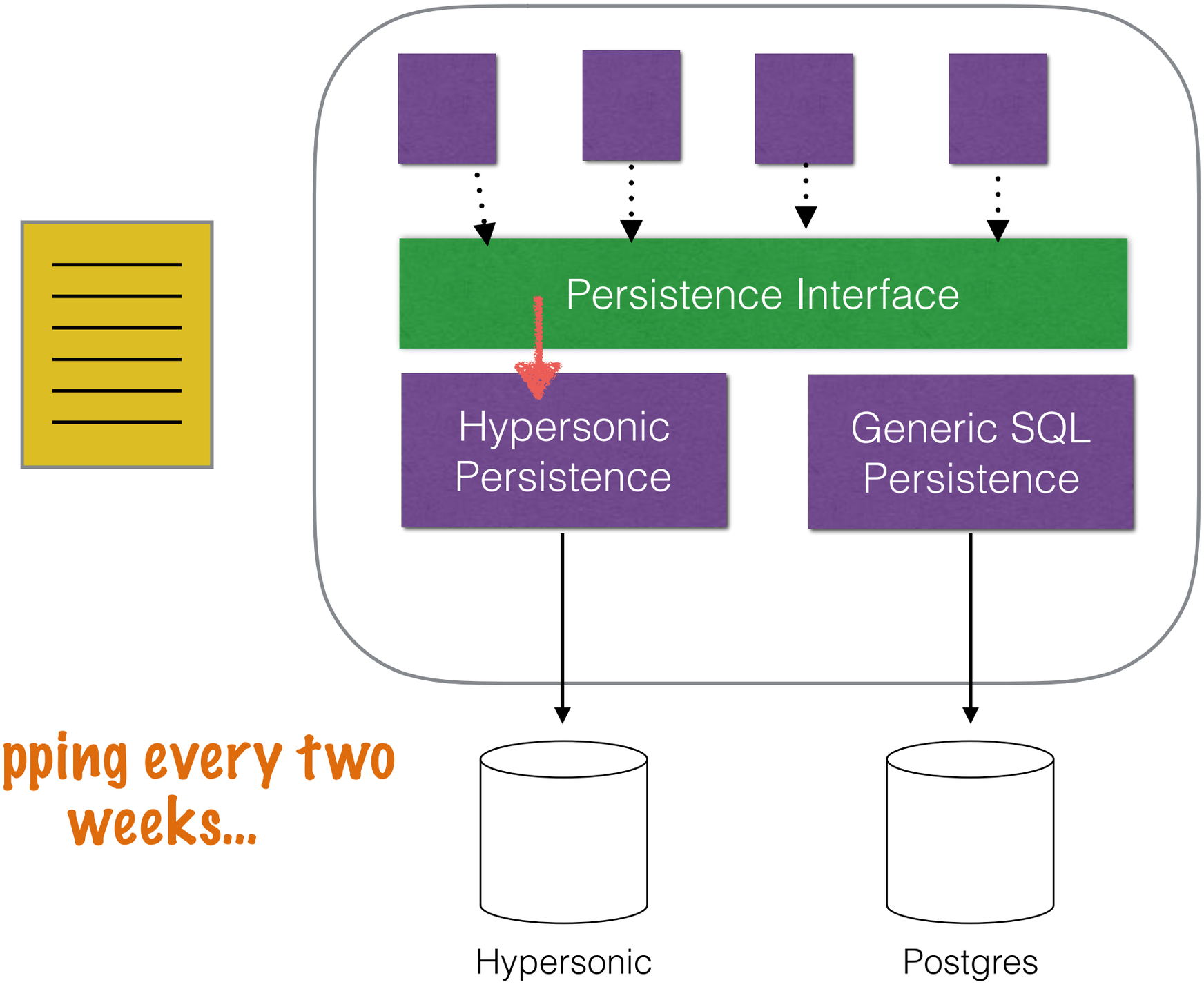
shipping every two weeks...



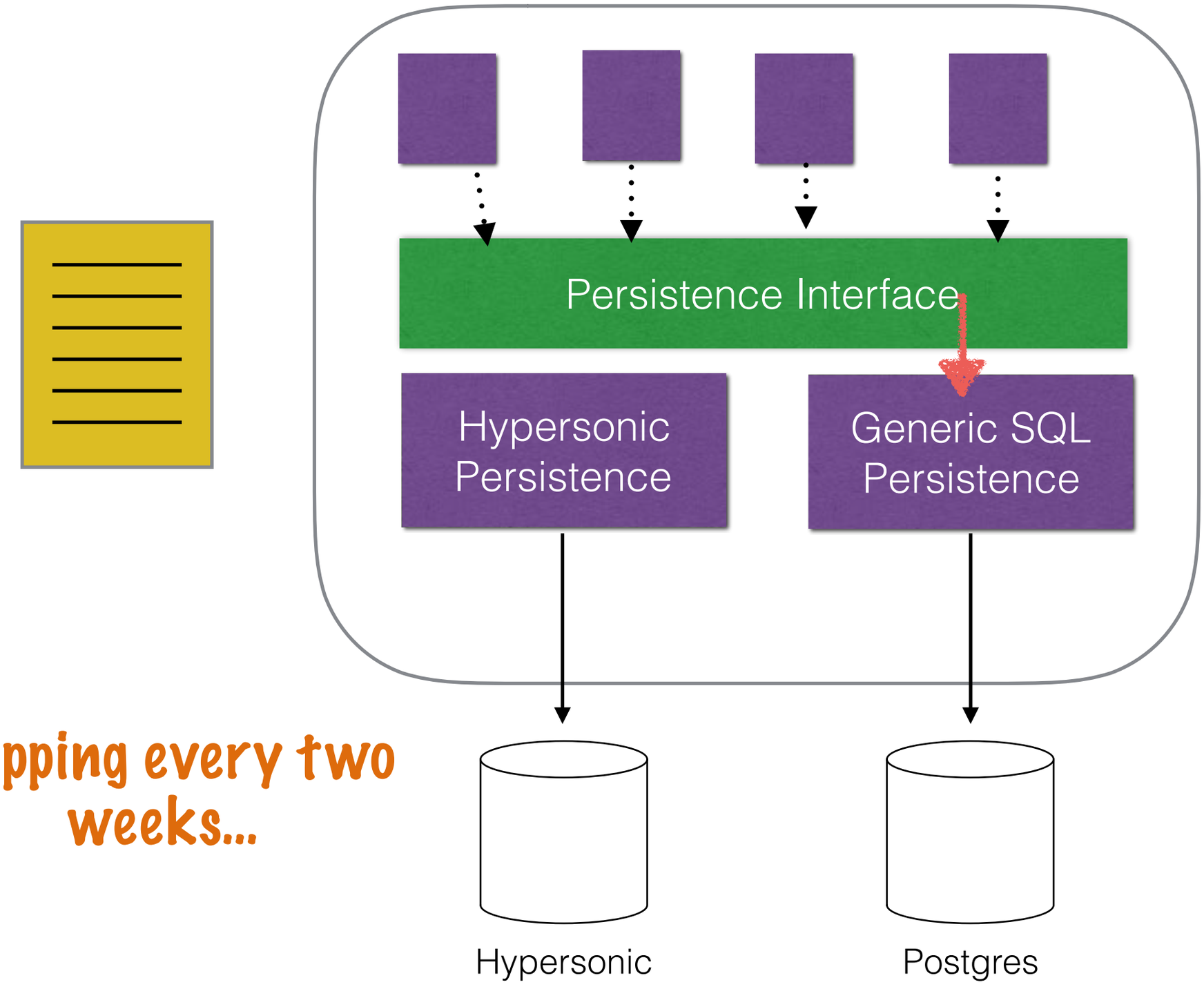
shipping every two weeks...



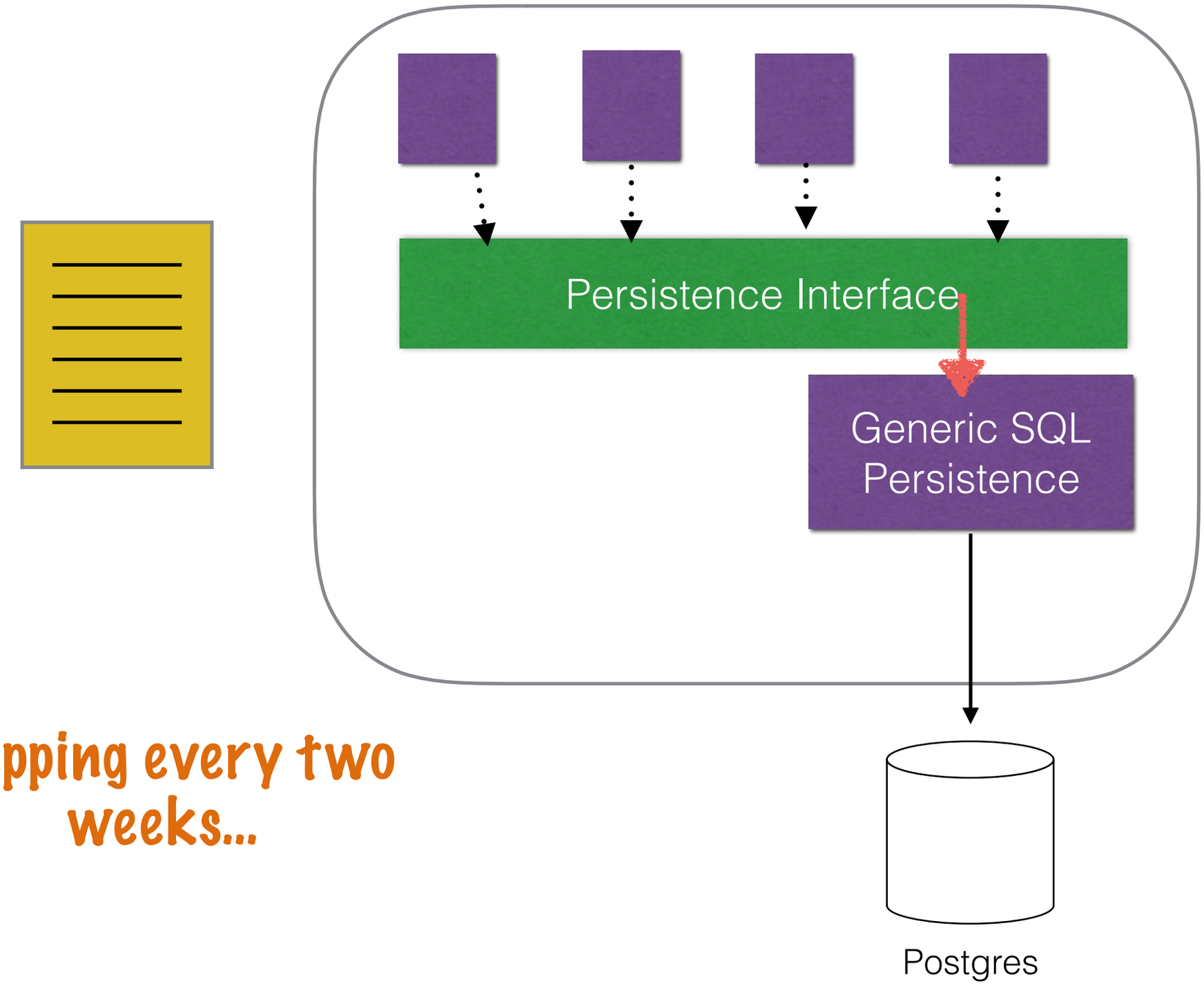
shipping every two weeks...



shipping every two weeks...



shipping every two weeks...



shipping every two weeks...

@gotochi

@samnewman

~~Feature Branching~~

~~Feature Branching~~

Branch-by abstraction

Important Rules:

@gotochi

@samnewman

Important Rules:

Use a flag in as few places as possible

Important Rules:

Use a flag in as few places as possible

Remove them once you're done

Side Benefits:

Side Benefits:

Can be used for A/B testing

Side Benefits:

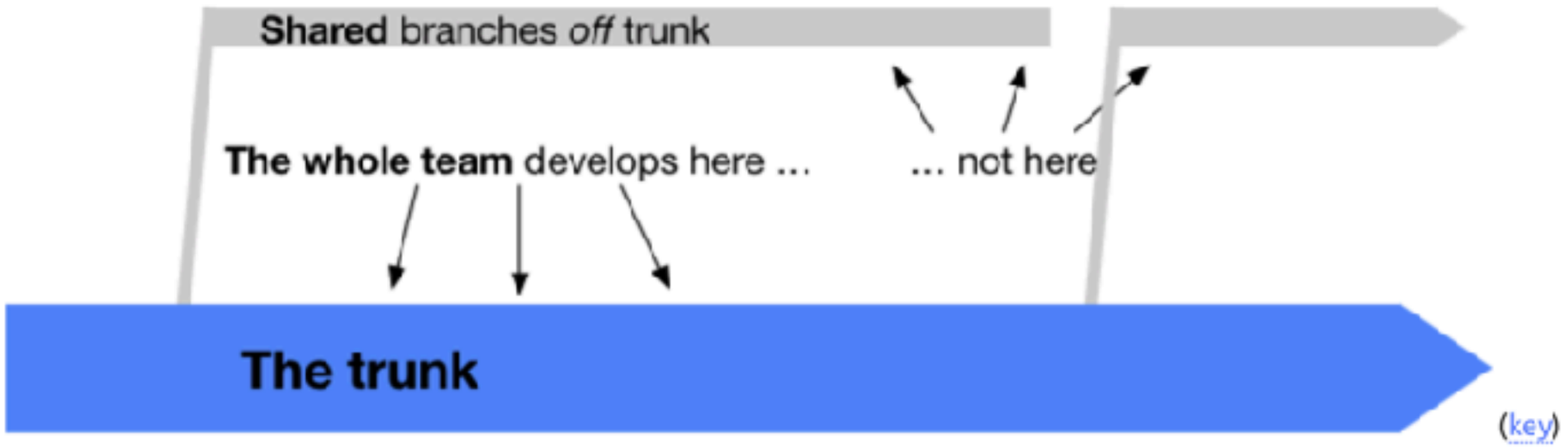
Can be used for A/B testing

And with some work, canary releasing

Introduction

One line summary

A source-control branching model, where developers collaborate on code in a single branch called 'trunk' *, resist any pressure to create other long-lived development branches by employing documented techniques, avoid merge hell, do not break the build, and live happily ever after.



* 'master', in Git nomenclature

Trunk-Based Development is a key enabler of Continuous Integration and by extension Continuous Delivery. When individuals on a team are committing their changes to the trunk multiple times a day it becomes easy to satisfy the core requirement of Continuous Integration that all team members commit to trunk at least once every 24 hours. This ensures the codebase is always releasable on demand and helps to make Continuous Delivery a reality.

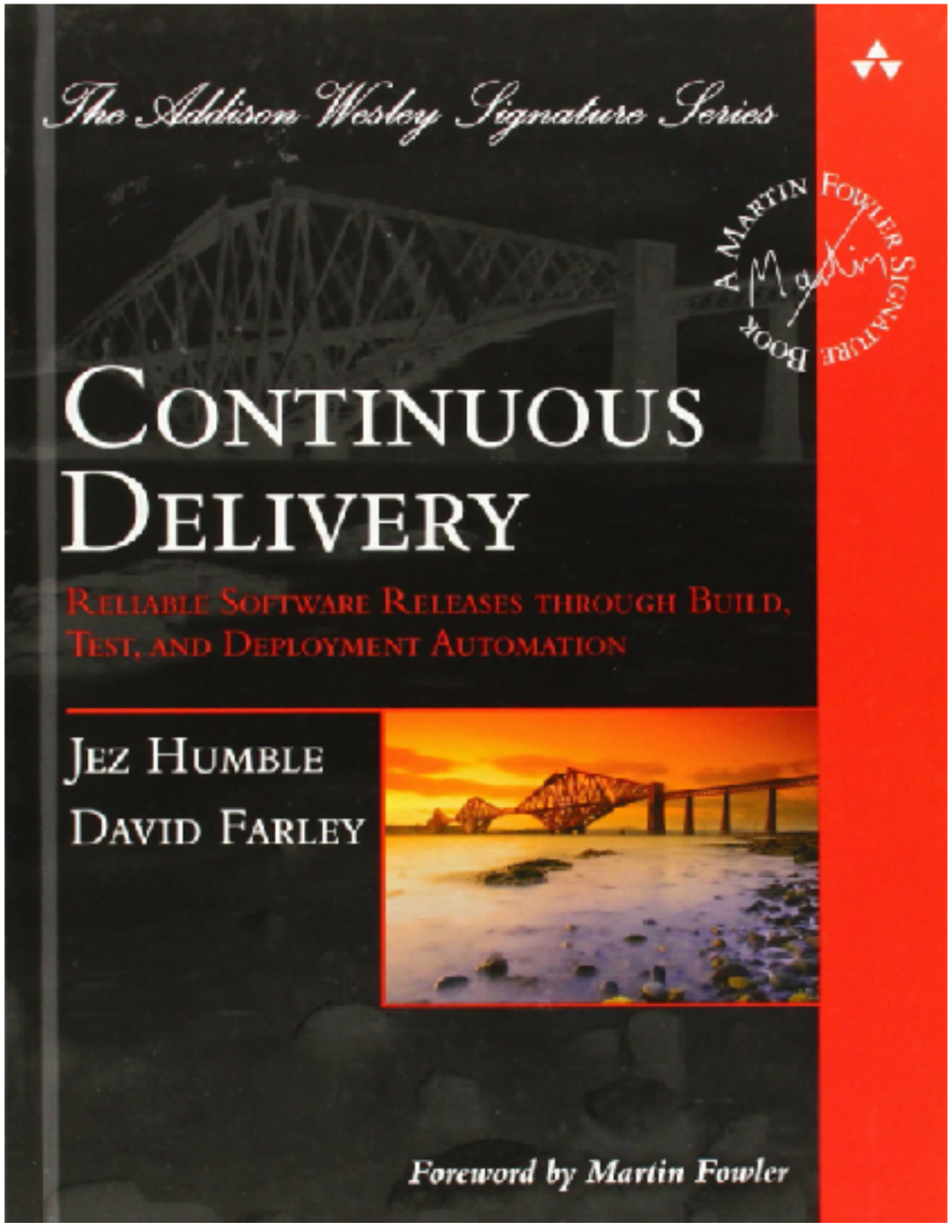
https://trunkbaseddevelopment.com

@gotochi @samnewman

Continuous Delivery Book...

@gotochi

@samnewman



Published in 2011

@gotochi

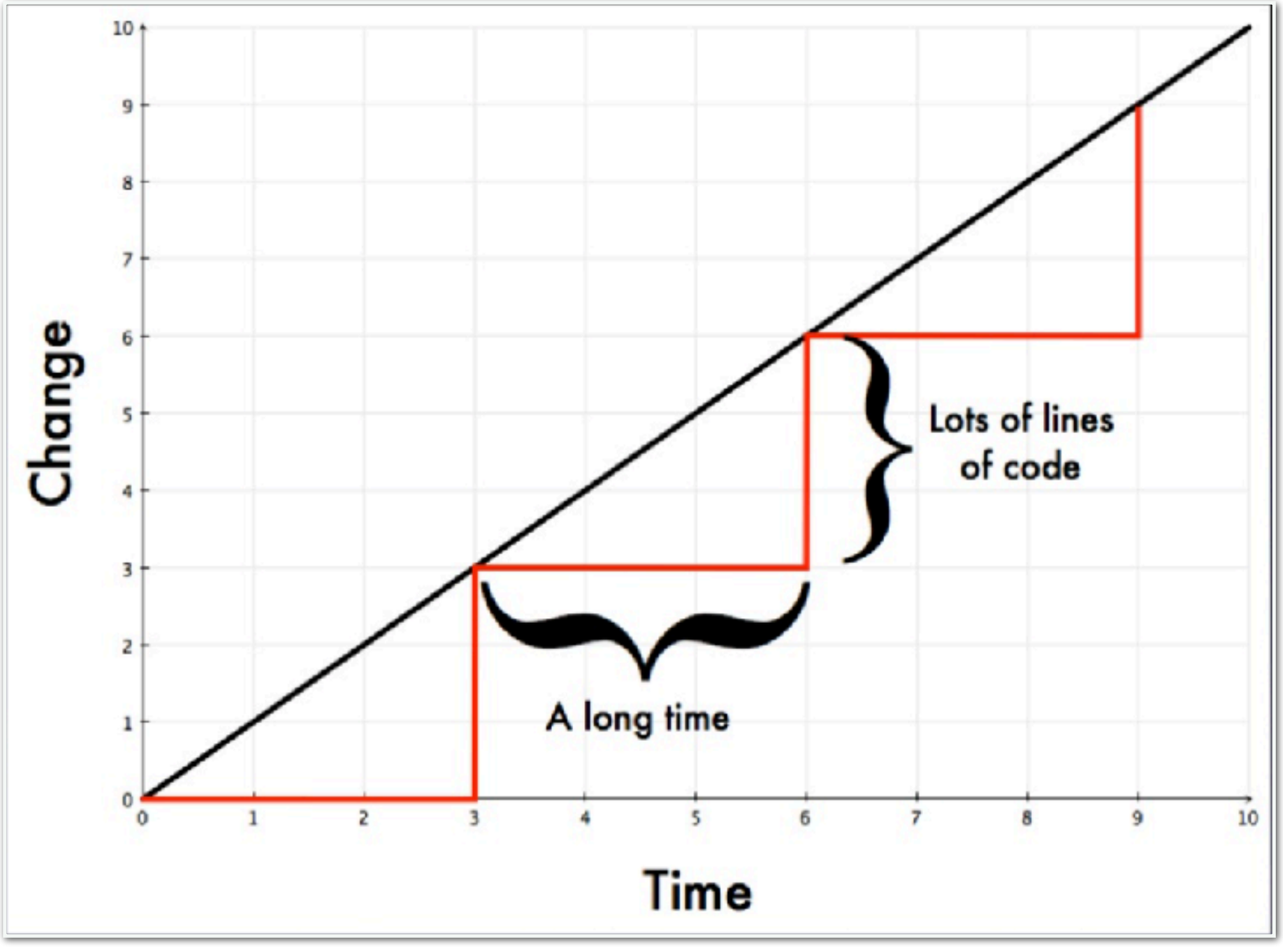
@samnewman

Treat every check-in as a release candidate

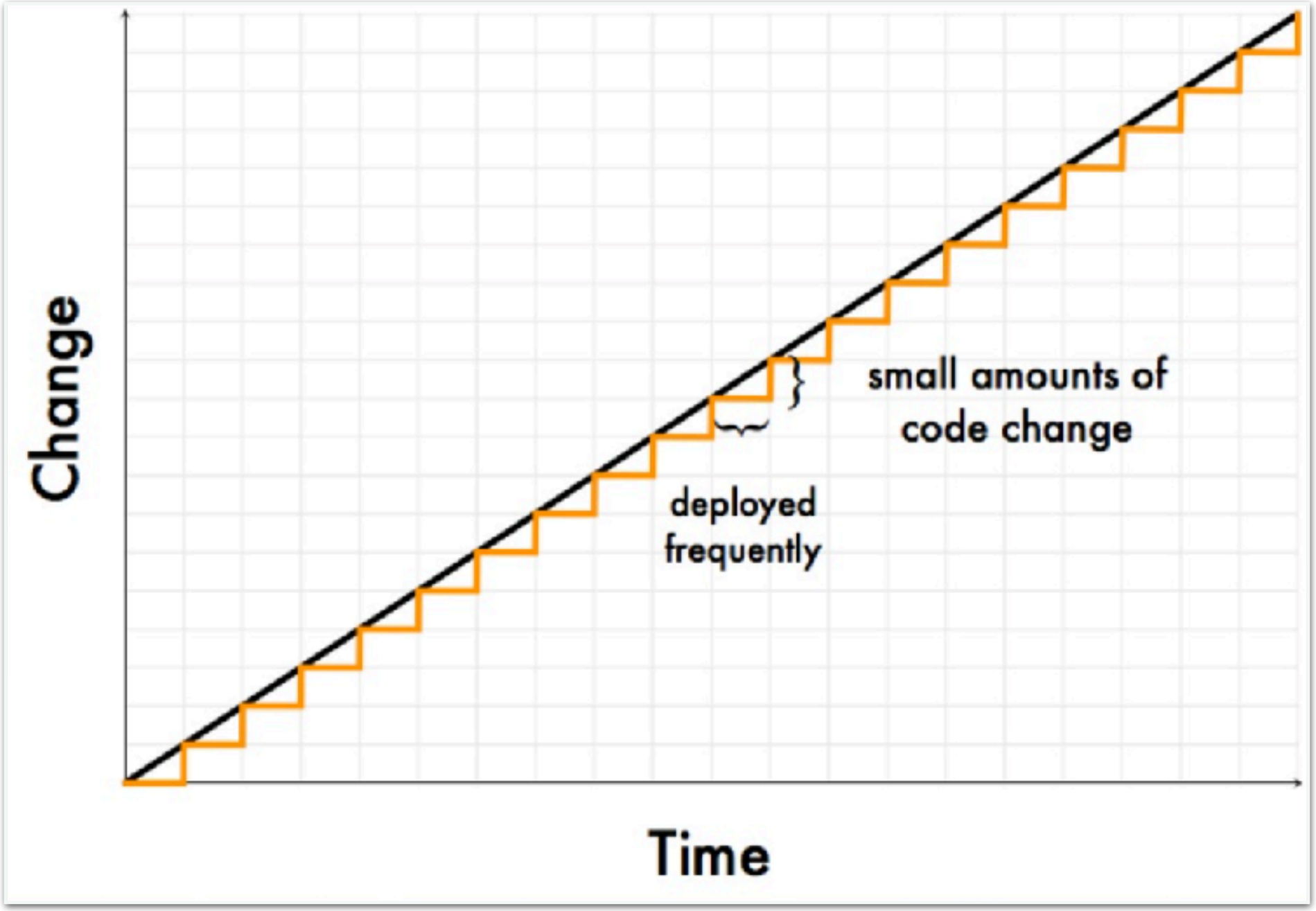


https://www.slideshare.net/jallspaw/ops-metametrics-the-currency-you-pay-for-change-4608108

@gotochi @samnewman



Ops Meta-Metrics - John Allspaw



Ops Meta-Metrics - John Allspaw

Integrate often

@gotochi

@samnewman

Integrate often

Keep batch size small

Integrate often

Keep batch size small

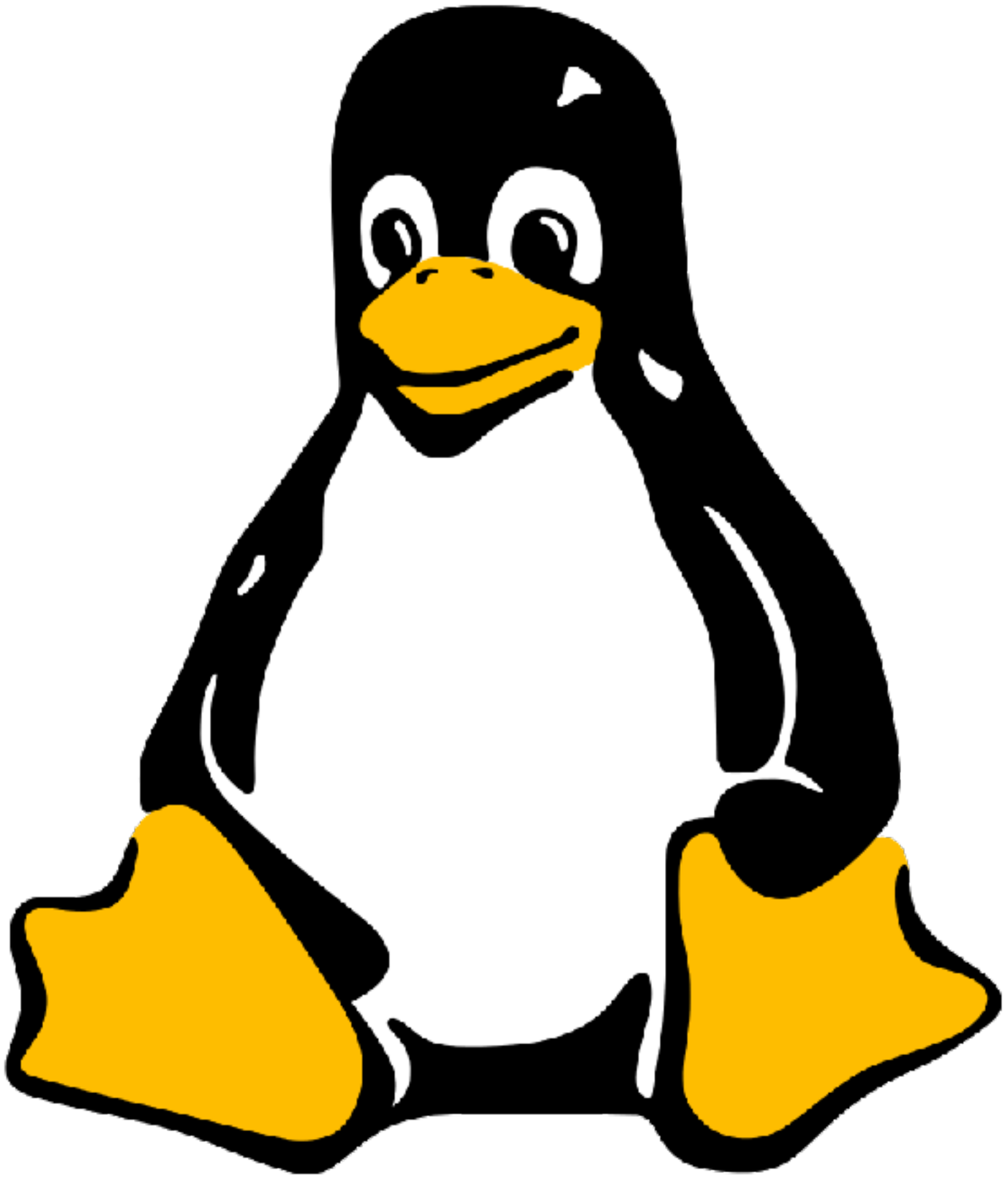
Ship often

@gotochi

@samnewman



And there there was Git



@gotochi

@samnewman

Goal

@gotochi

@samnewman

Goal

Merge a patch in less than 3 seconds

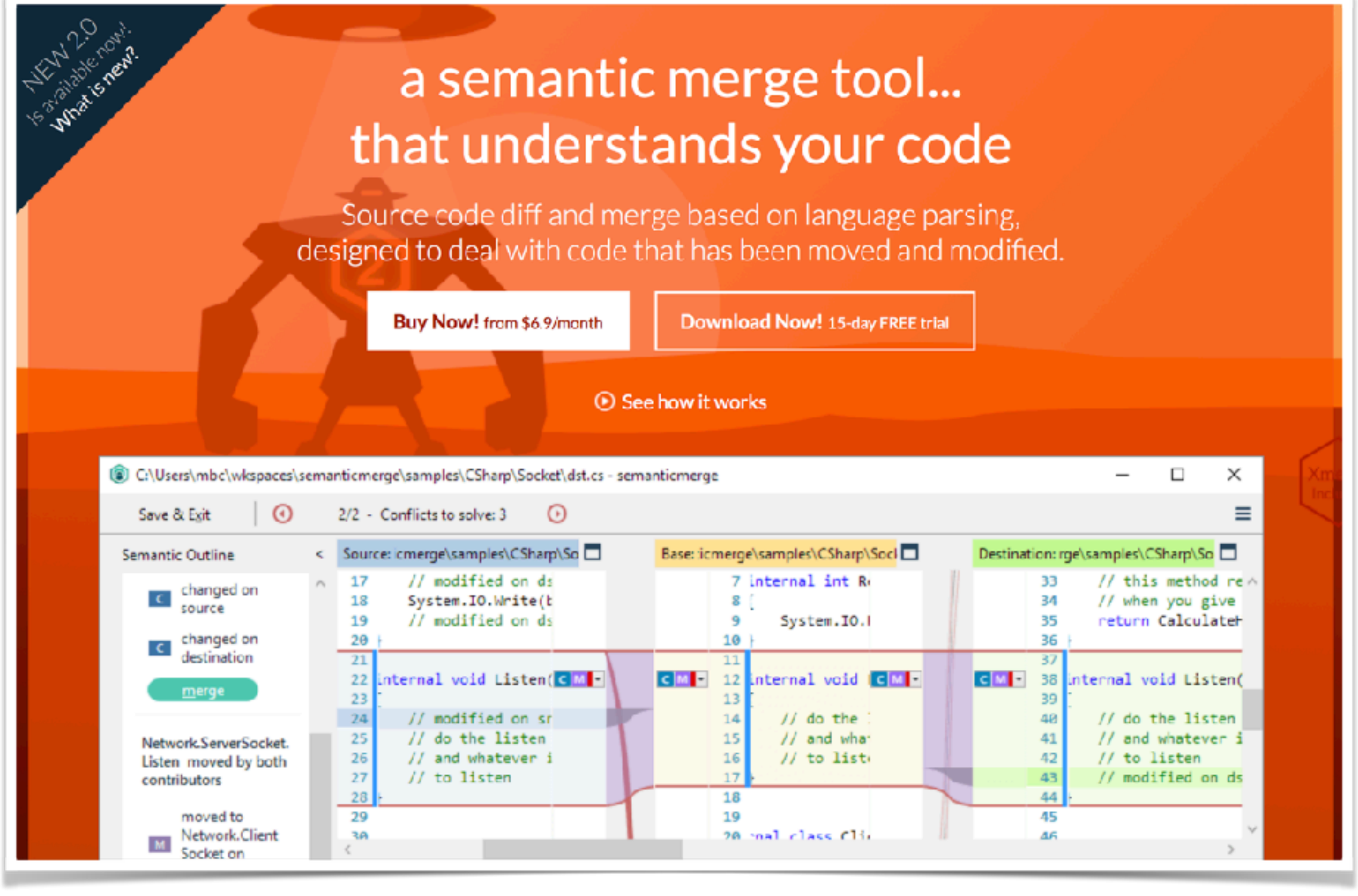
@gotochi

@samnewman

Branches much more lightweight

Full local copy of source history

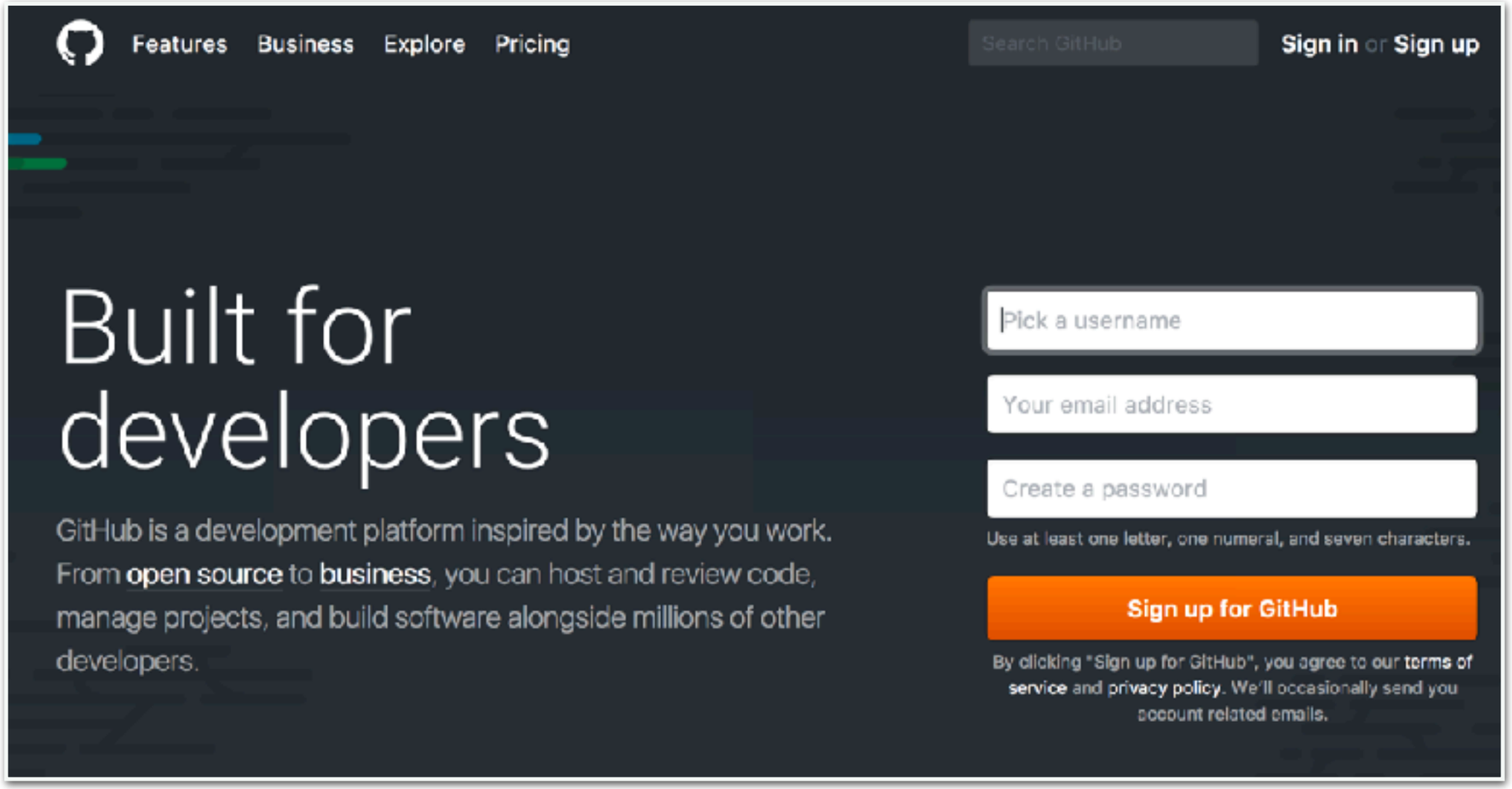
Merging of text is better but...



@gotochi

@samnewman

And then there was GitHub...


Features
Business
Explore
Pricing
Search GitHub
Sign in or Sign up
Built for developers
GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside millions of other developers.
Pick a username
Your email address
Create a password
Use at least one letter, one numeral, and seven characters.
Sign up for GitHub
By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We'll occasionally send you account related emails.

2008

Oh yeah, there's pull requests now

February 23, 2008 defunkt New Features

Last night I pushed out a feature Tom and I have been talking about since day one: pull requests. That's the short walkthrough.



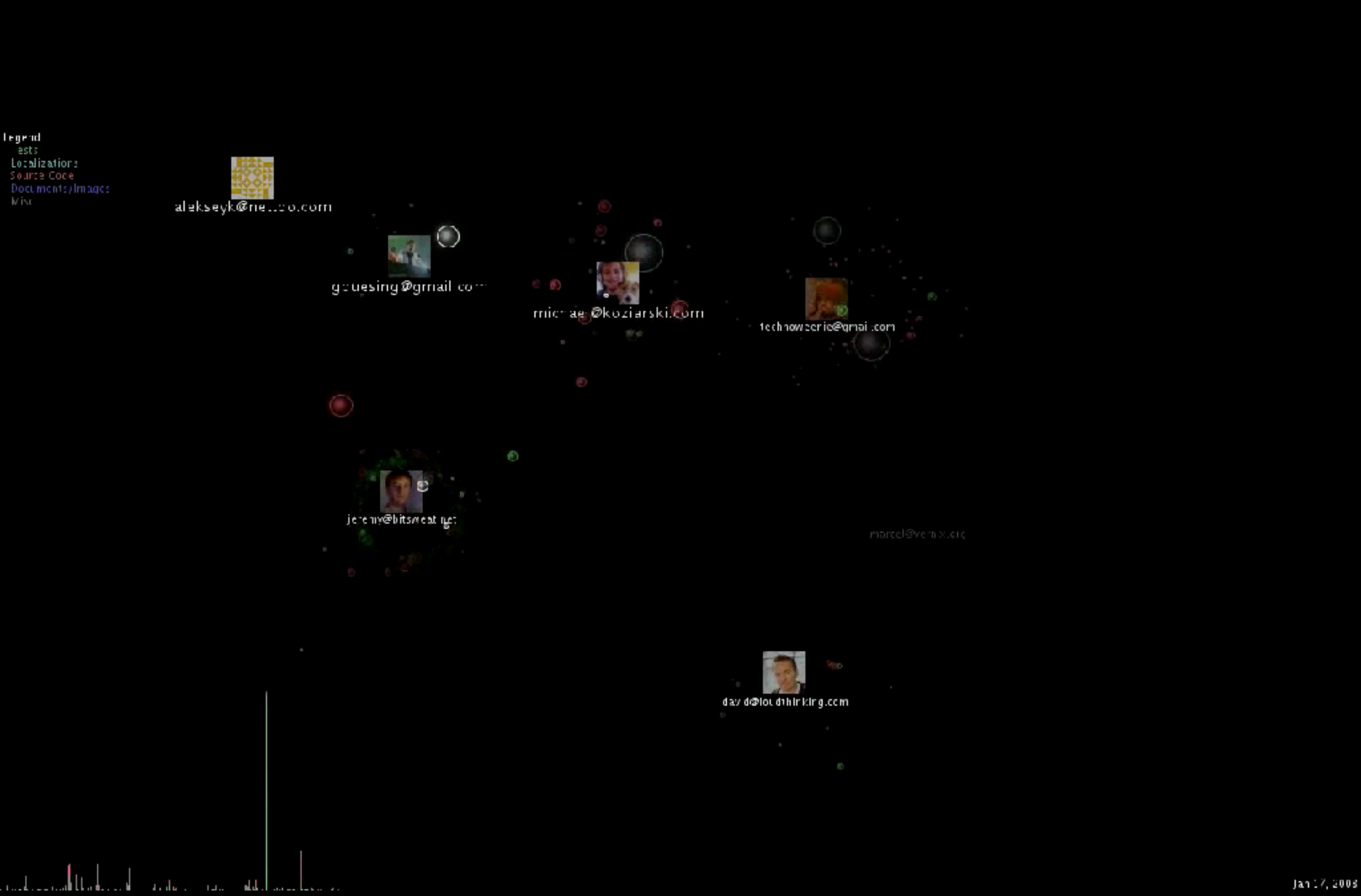
You can use it to tell people who forked from you they need to pull, or they can use it to ask you to pull. It's also great for letting someone else know you have a cool feature pushed to some non-master branch.

Have feedback on this post? Let @github know on Twitter.

Need help or found a bug? Contact us.

https://github.com/blog/3-oh-yeah-there-s-pull-requests-now

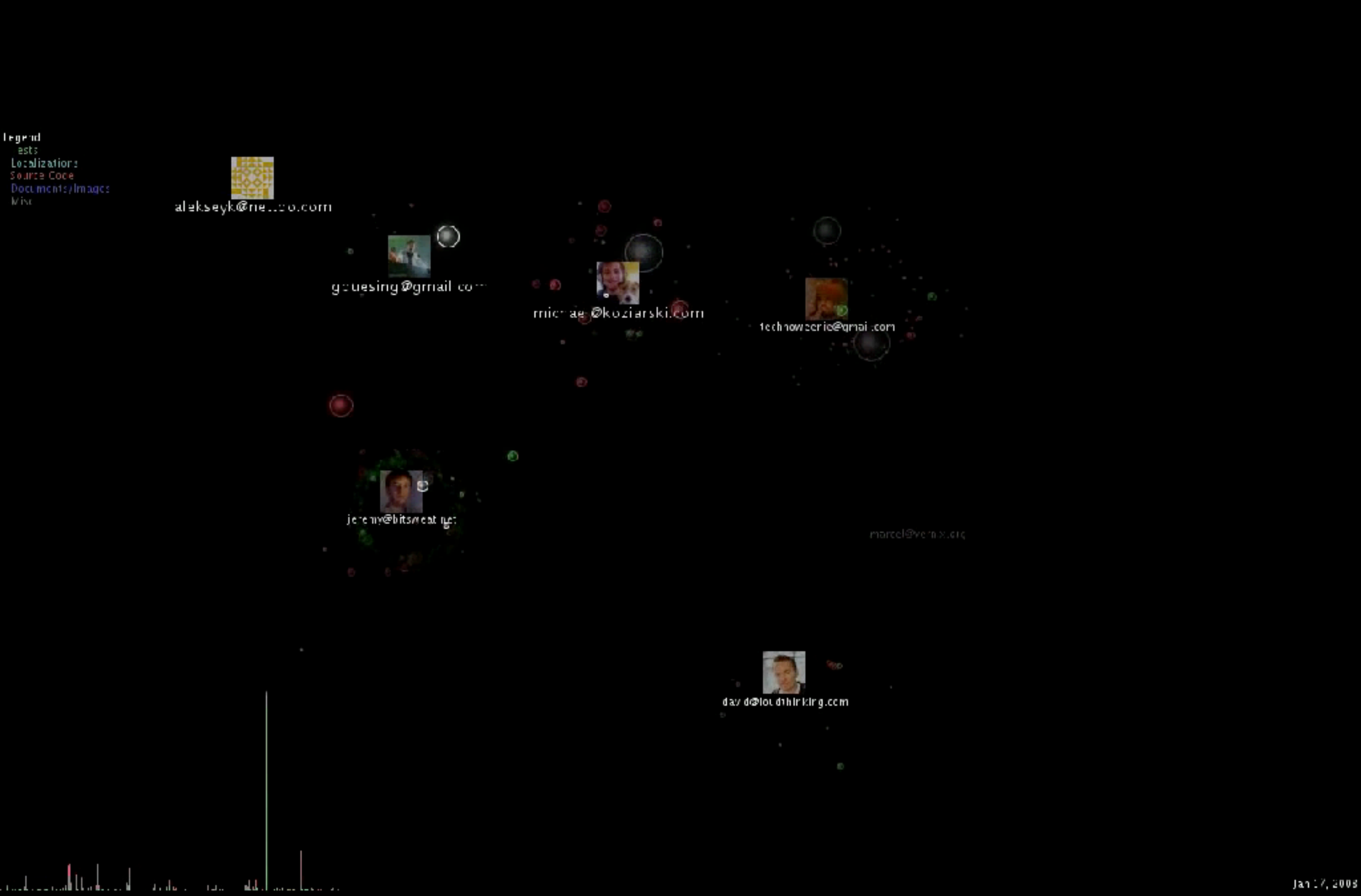
@gotochi @samnewman



@gotochi

https://vimeo.com/2979844

@samnewman



@gotochi

https://vimeo.com/2979844

@samnewman

rwLABS
The Internet of Things Accelerator
SAN FRANCISCO | HONG KONG | SHENZEN
BUILDING AN IOT STARTUP?
Apply Now

Github Has Surpassed Sourceforge and Google Code in Popularity

Posted on June 2, 2011 in HACK



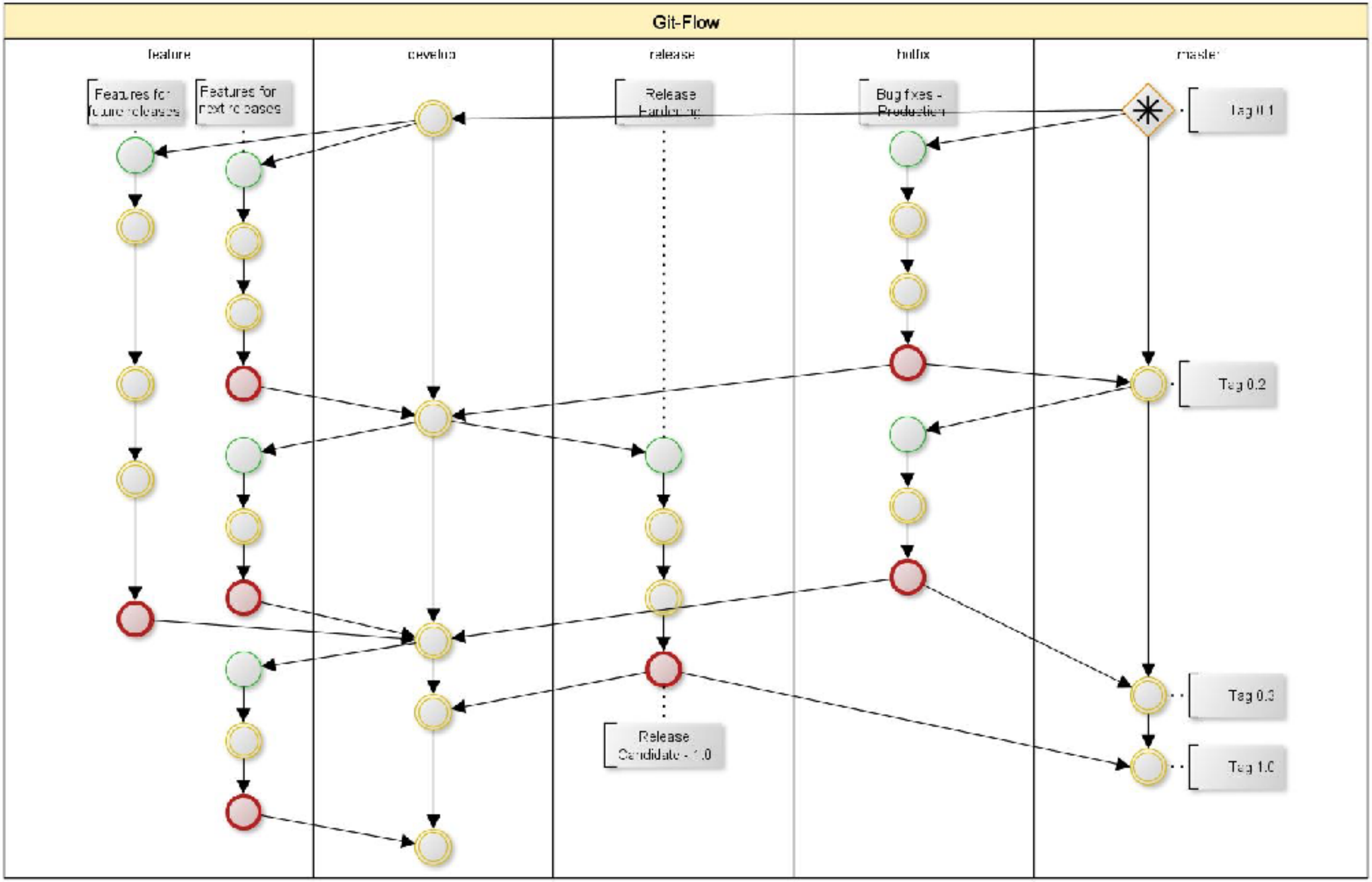
KLINT FINLEY
Contributing Writer

6
Shares

http://readwrite.com/2011/06/02/github-has-passed-sourceforge/

But PRs use branching!

And then there was gitflow…

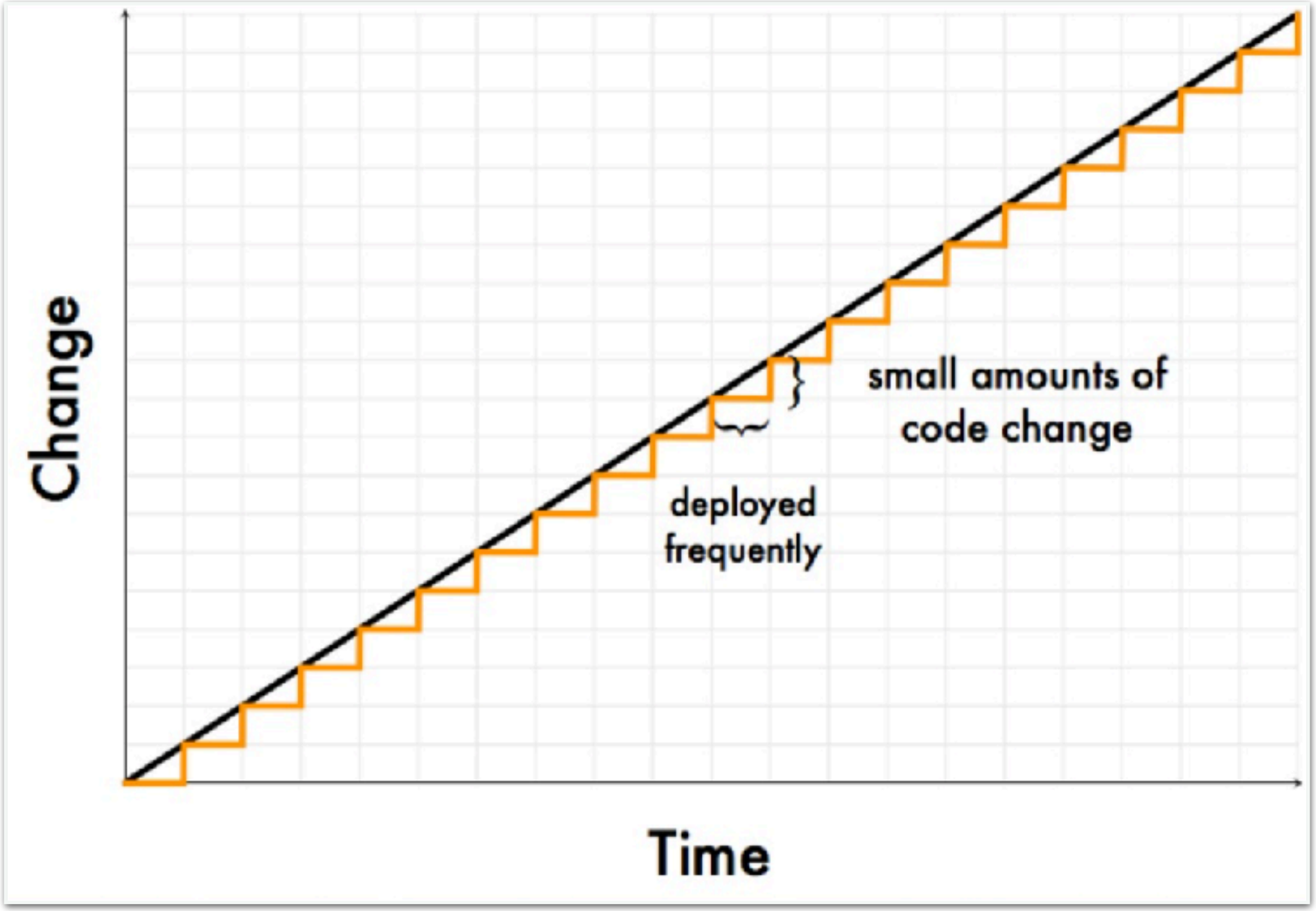


@gotochi

@samnewman



https://www.flickr.com/photos/hackny/8675049276/



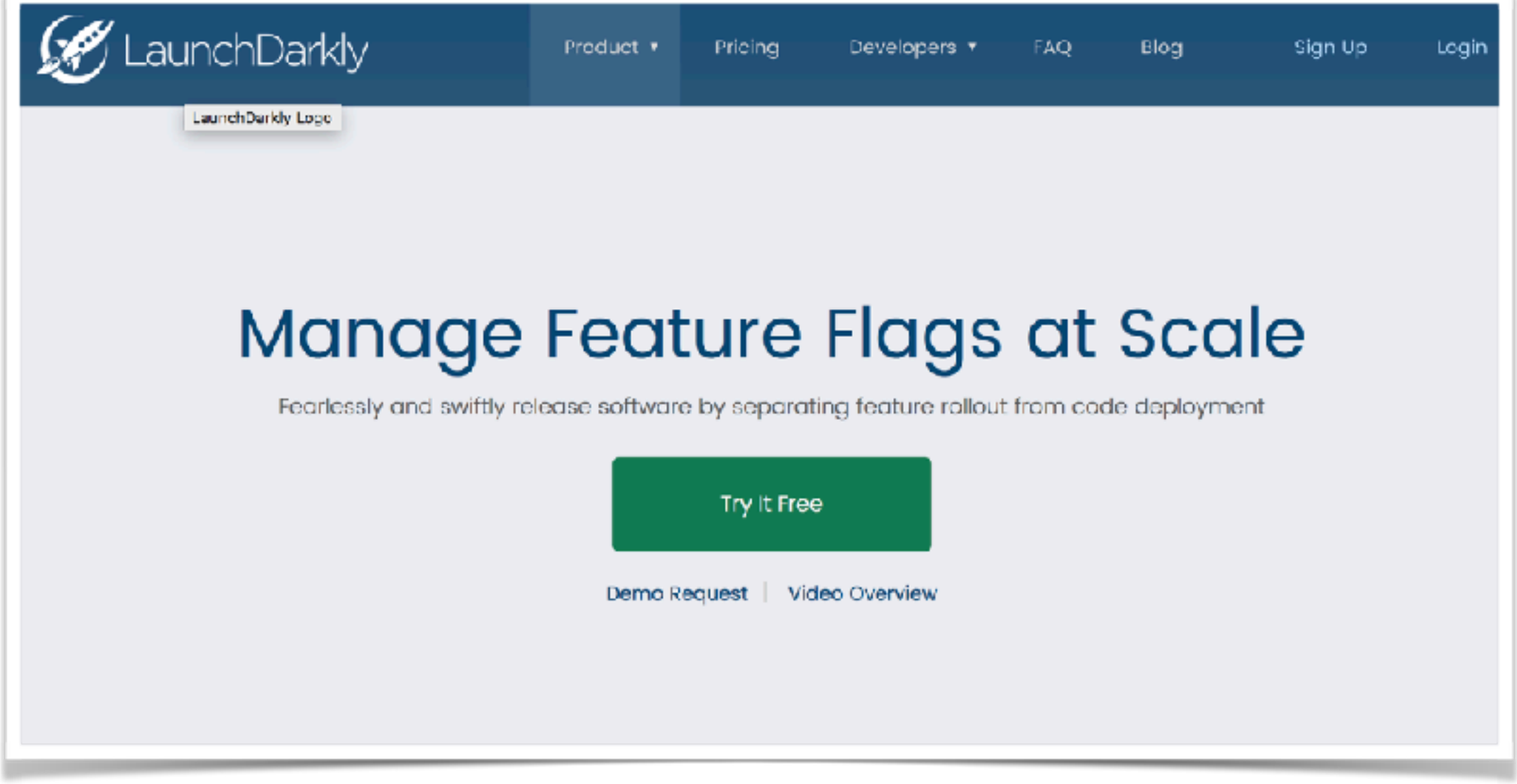
Ops Meta-Metrics - John Allspaw

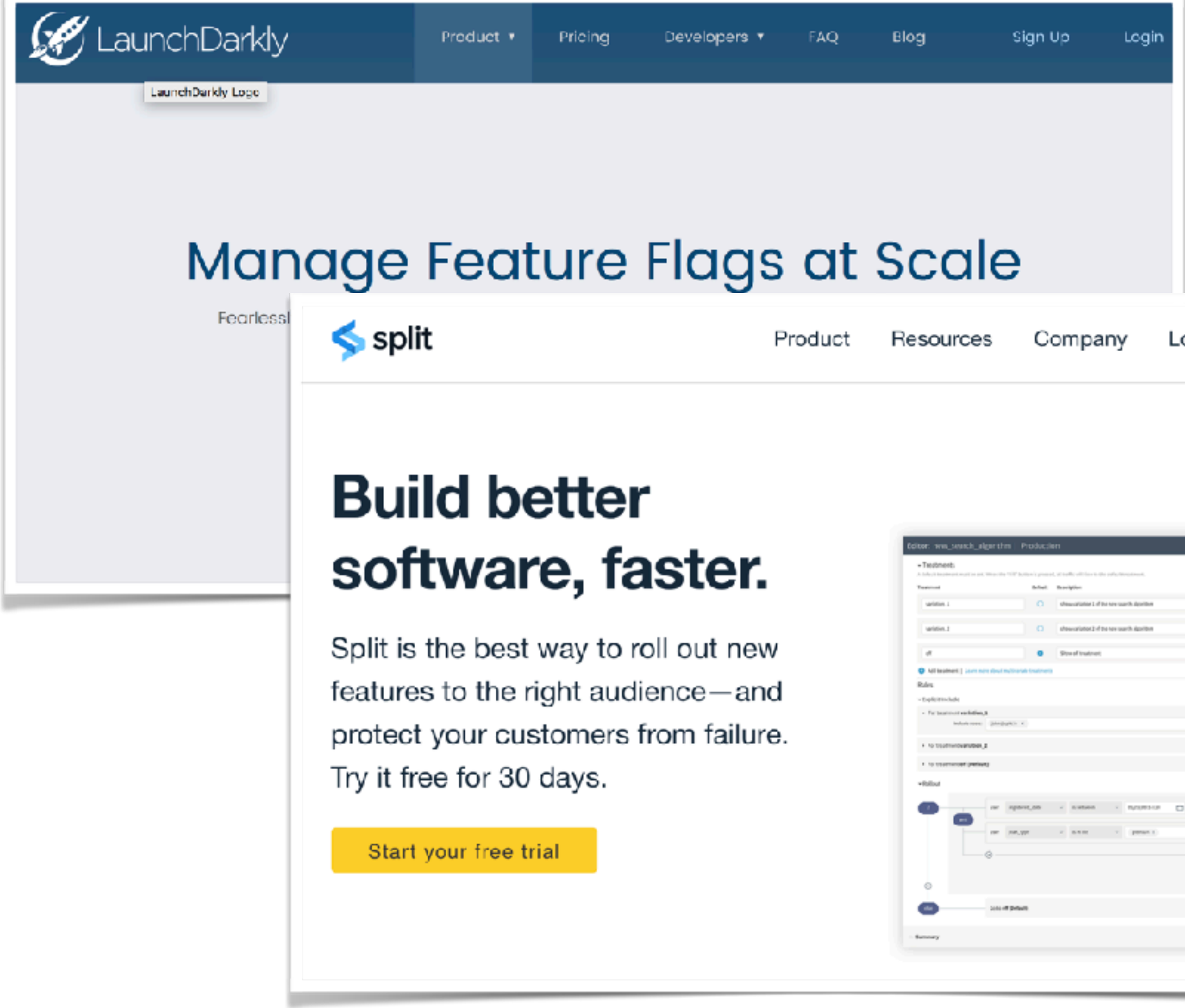
But if I merge in frequently, why do I need GitFlow?

split.io, launch darkly

@gotochi

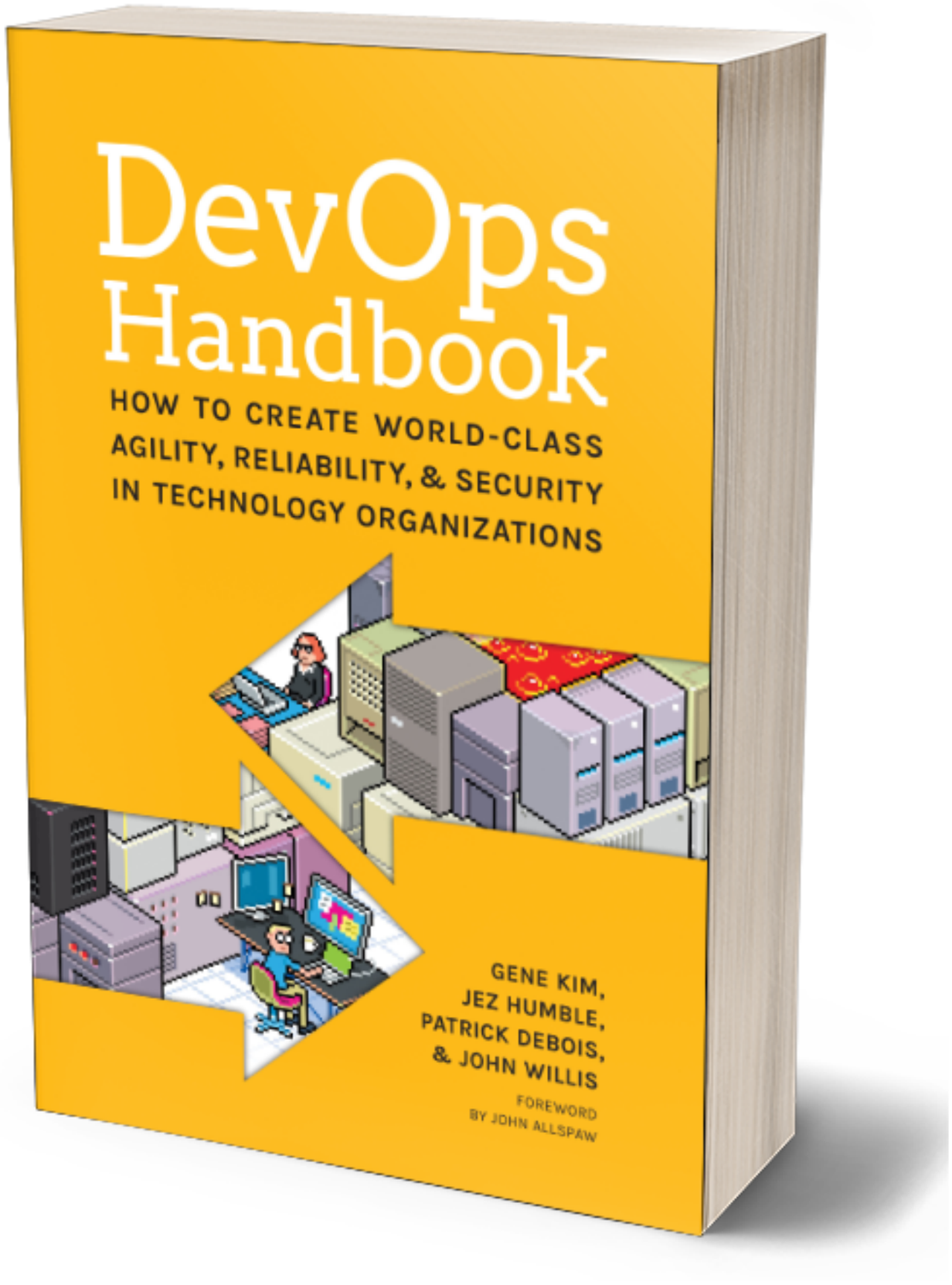
@samnewman


LaunchDarkly
Product
Pricing
Developers
FAQ
Blog
Sign Up
Login
LaunchDarkly Logo
Manage Feature Flags at Scale
Fearlessly and swiftly release software by separating feature rollout from code deployment
Try It Free
Demo Request
Video Overview



@gotochi

@samnewman



@gotochi

@samnewman

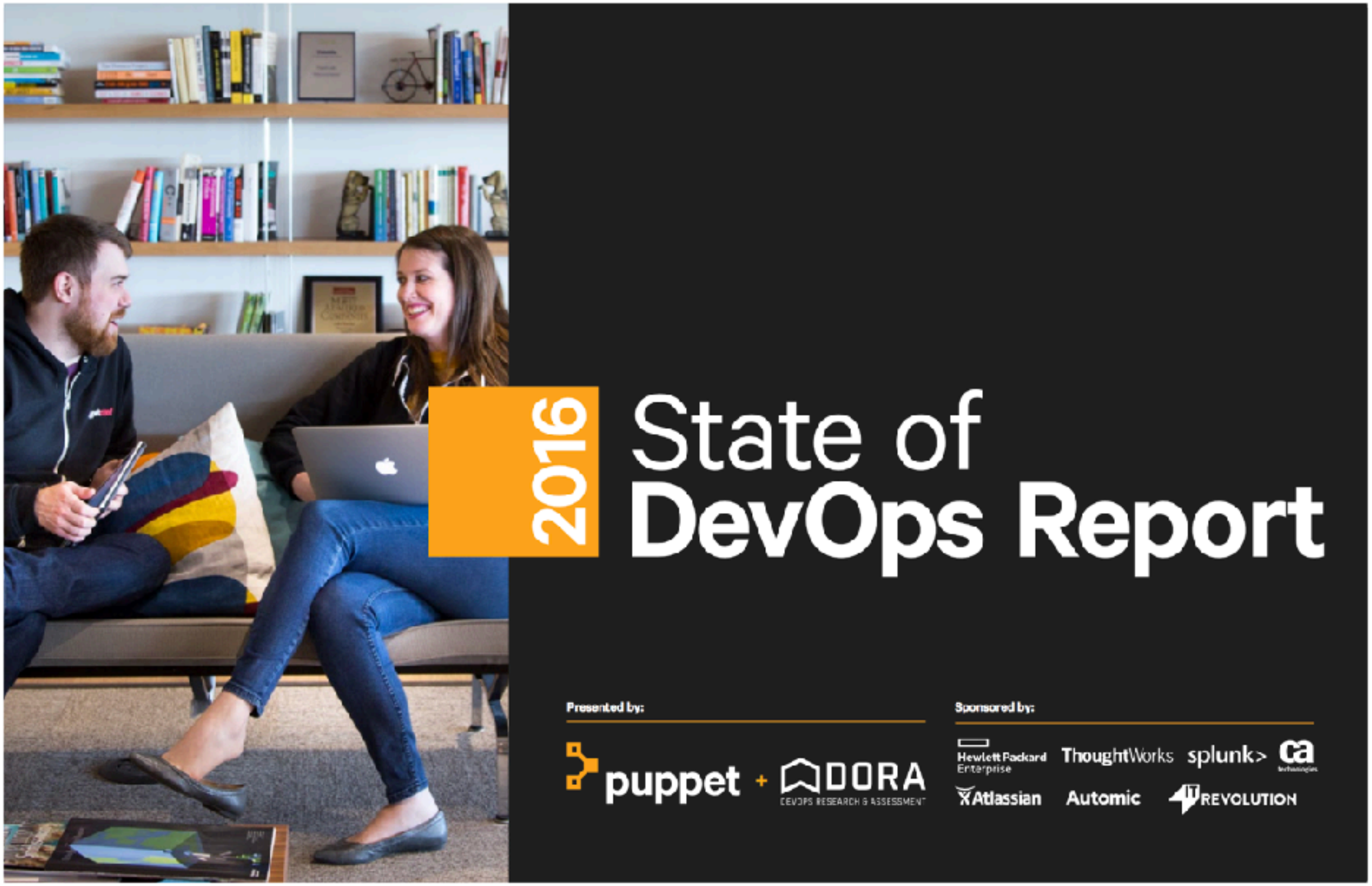
“Trunk-based development is likely the most controversial practice discussed in this book

"Trunk-based development is likely the most controversial practice discussed in this book

[…]

However, the data from Puppet Labs' *2015 State of DevOps Report* is clear: trunk-based development predicts higher throughput and better stability, and even higher job satisfaction and lower rates of burnout."

- Gene Kim, Jez Humble, Patrick Debois & John Willis



@gotochi

@samnewman

“We found that having branches or forks with very short lifetimes (less than a day) before being merged into trunk, and less than three active branches in total, are important aspects of continuous delivery, and all contribute to higher performance. So does merging code into trunk or master on a daily basis. “

- State Of Devops Report, 2016

So are branches evil?

Integrate often

@gotochi

@samnewman

Integrate often

Keep batch size small

Integrate often

Keep batch size small

Ship often

@gotochi

@samnewman

Thank You!

http://samnewman.io

@samnewman

Also thanks to Don Clark for the people icons
https://commons.wikimedia.org/wiki/File:Gender_neutral.svg CC BY-SA 4.0