Controlling Your Environments using Infrastructure As Code

ConFoo Montréal - February 2016

Steve Mercier

Who am I?

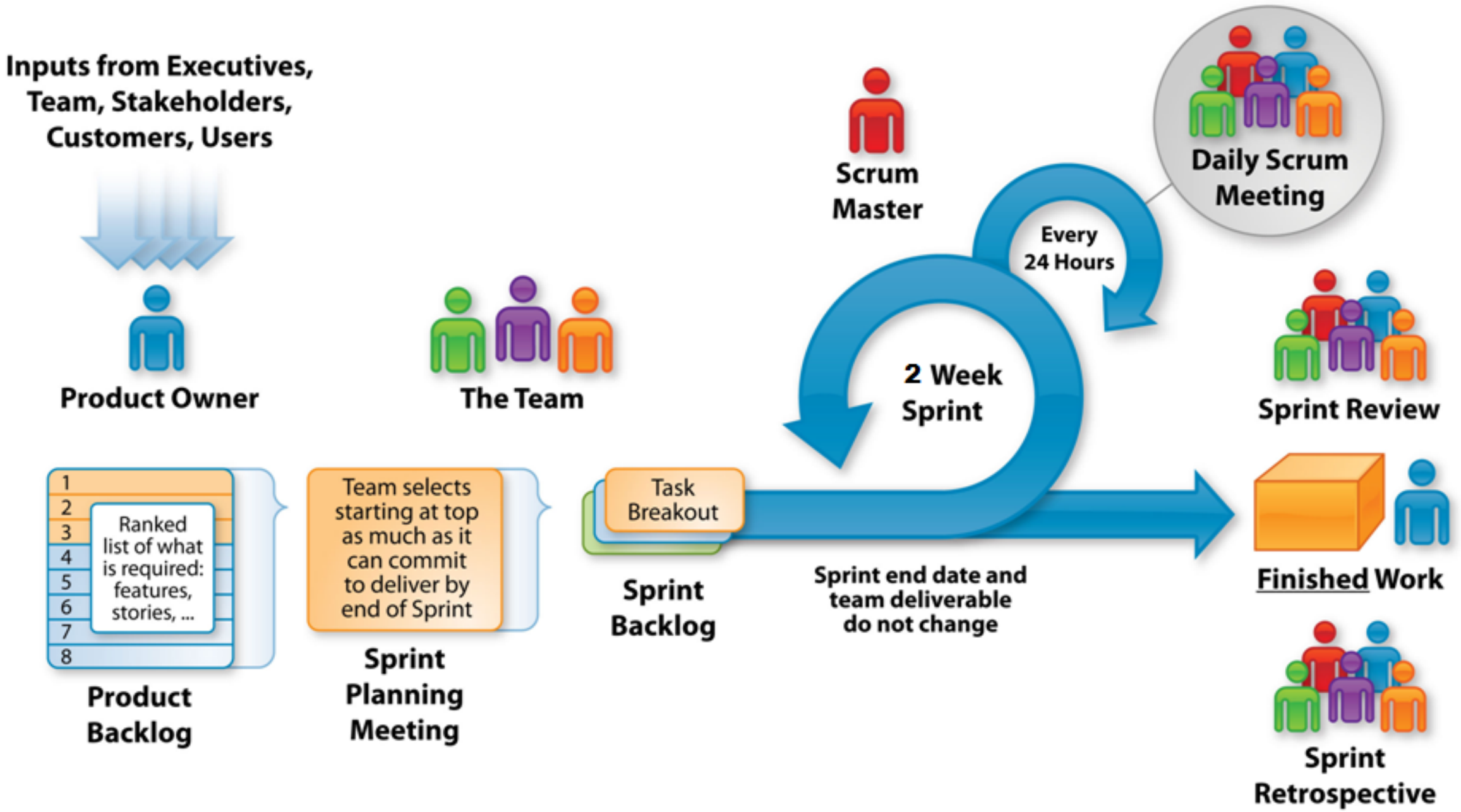
I participated in multiple software development projects:

- from very small (<10 person) to large (~400 persons)
- from a long time ago (>20 years) up to now
- from different angles/roles: Developer, Architect, PM, DevOps/ BuildMaster, Software Release Manager
- I have seen software methods/processes come and go: Waterfall, RUP, OpenUP, XP, Scrum, Scaled Agile, etc.

But the essential remains:
I believe that to produce good software, it takes good people + resources AND discipline/professionalism/focus!

Are You Agile?

Agile is not easy… and is not only about ceremonies + tools!



Scrum Overview

Nothing on the previous slide concerning software context, where is the code delivered, running, performing…

Agile typically deals with this with the Definition Of Done concept, sadly overlooked way too often…

What would be a good measure of Agility anyway?

Suggestion #1:

The capability to release
(aka)
The frequency at which you are delivering software updates that add business value to your clients without breaking any previously delivered business value that you want to retain

Suggestion #2:

The time to release

(aka)

The time it takes you to deliver the smallest change/fix to your software in production

Given that:

What prevents true agility?

What prevents you to **release**?

What prevents you to release **fast**?

From my point of view:

Lack of environments' management

True for all types of environments:
DEV, QA, Staging, Production

Platform dependencies?
Do you manage them?

ex: Frameworks dependencies, external libraries dependencies, etc. Are they either never updated or are they continuously creating problem when you upgrade them?

OS dependencies?
Do you manage them?

How is the OS changed on your platform? Does it break your applications sometimes? Any logs/traces of those changes? Can they be rolled back in case of problems?

Hardware/VM specs dependencies?
Do you manage them?

What happens if available RAM/Disk/Network gets below what your applications need? Do you know what they need?

If you manage your environments, do you manage them manually?

Everything not automated reduces your Agility

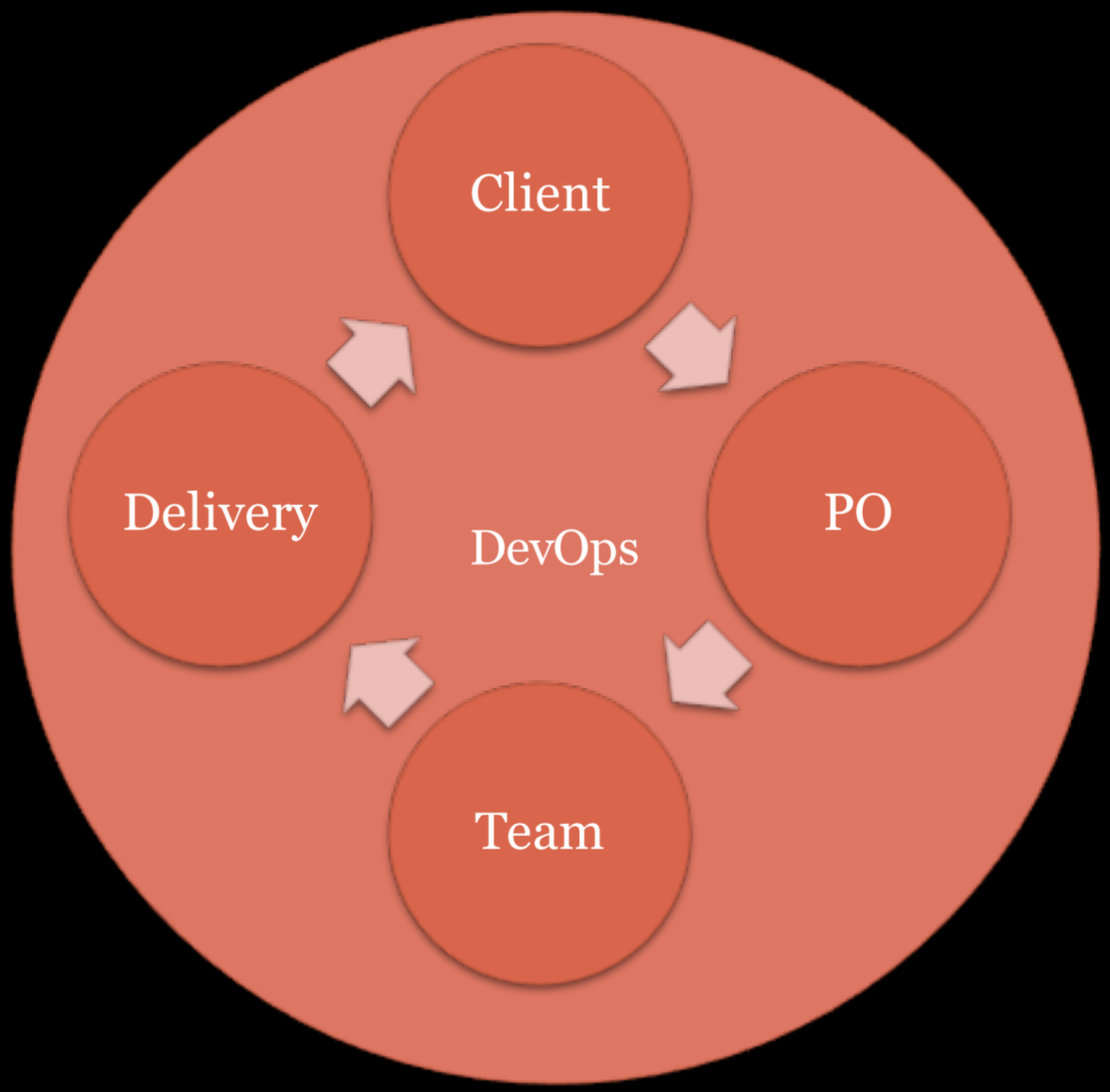
Manual triggers can take a long time

Manual = Time++;

As your manager might say

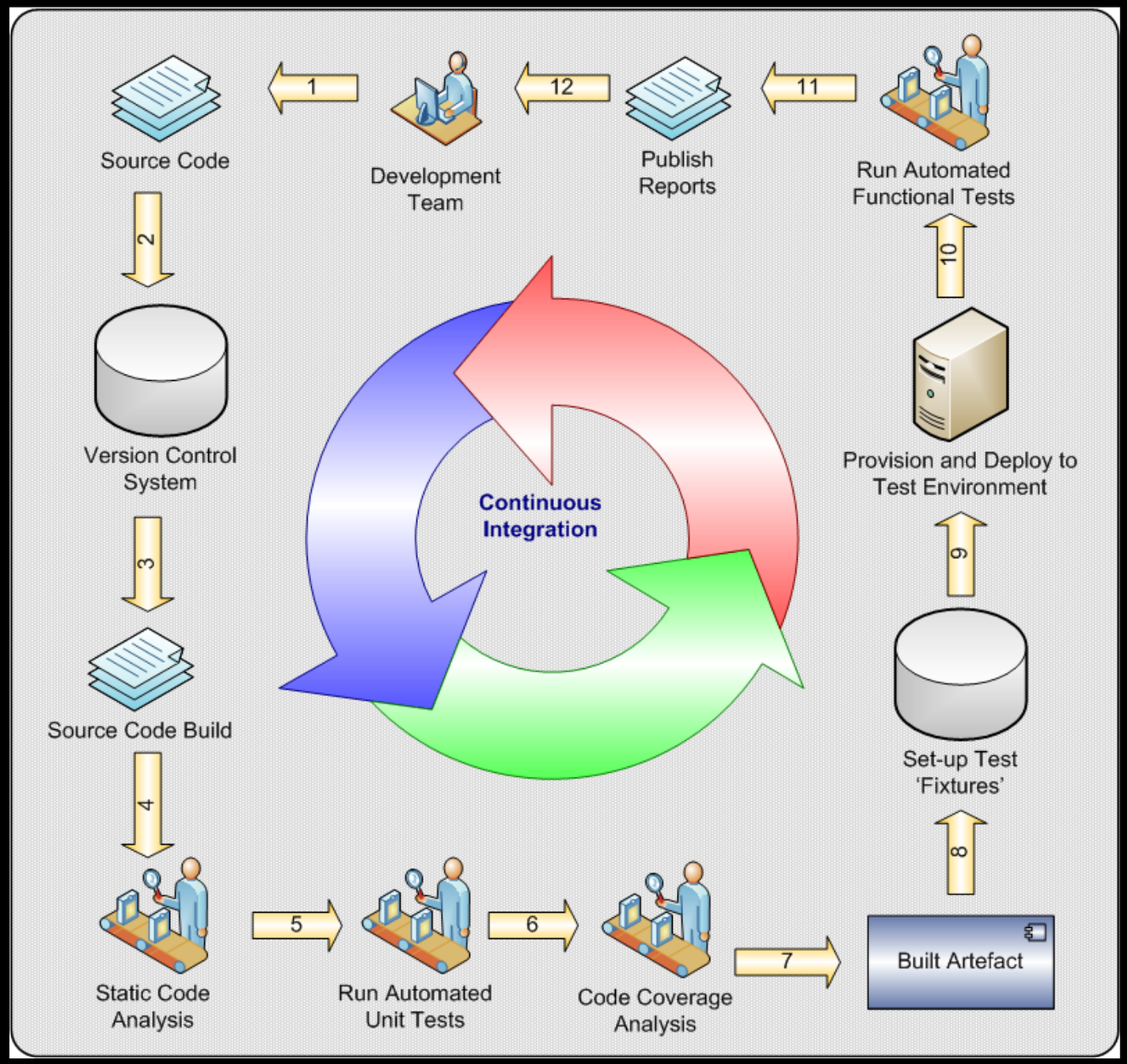
Time == Money();

What techniques can reduce manual steps?



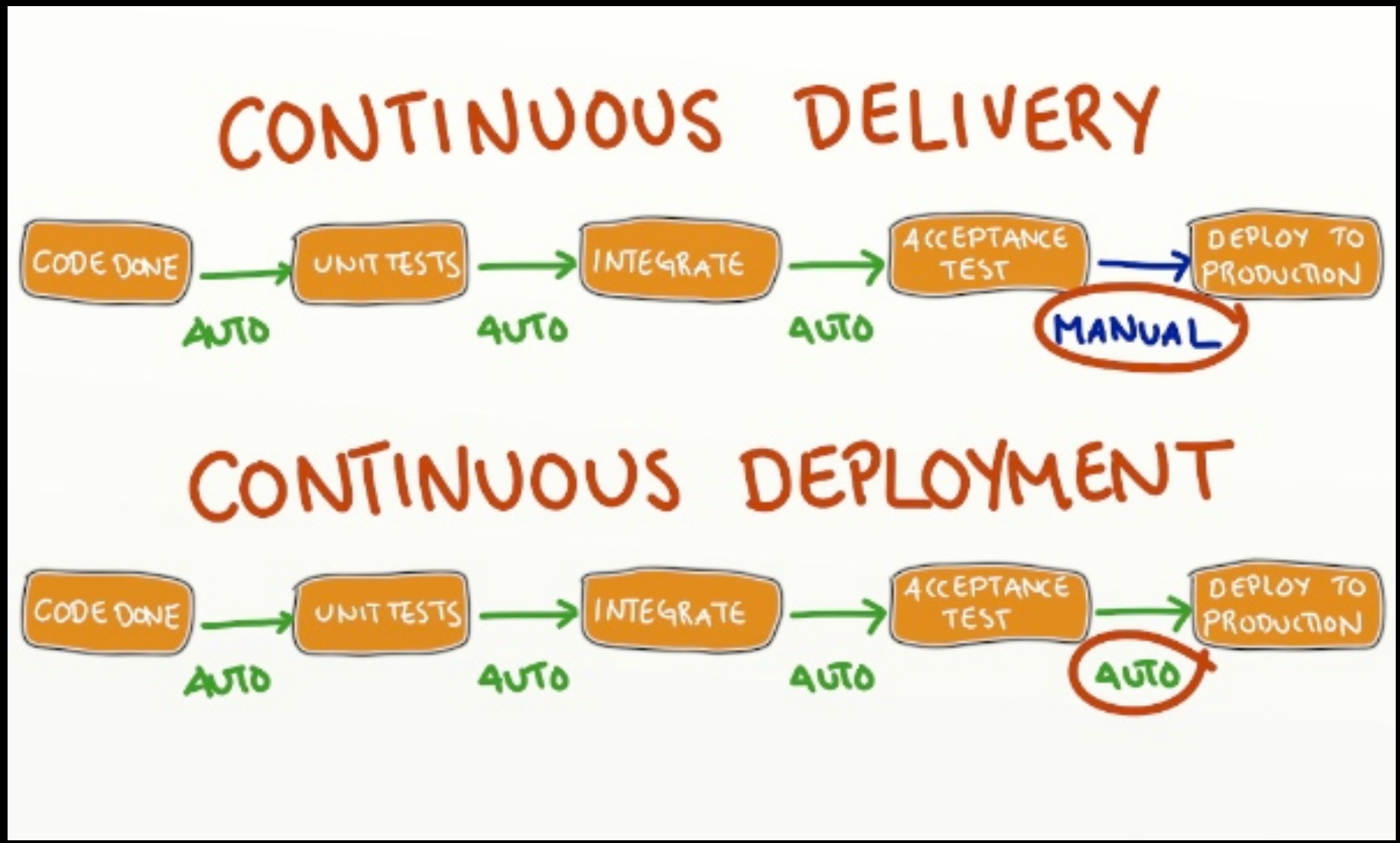
You could consider using DevOps:
CI server, CD server, Infrastructure as Code to automate as mush as possible your release process

What techniques can reduce manual steps?



Continuous Integration

What techniques can reduce manual steps?



Continuous Delivery/Deployment

What techniques can reduce manual steps?

Puppet

```
node 'codecamp.ro' {
  package { 'ruby'
    ensure => 'latest'
  }
}
```

CODECAMP 2012 @hurrycane

Saturday, March 10, 12

Infrastructure As Code

Are your Agile Demos done from unreleased software

(aka Works at my desk)

Your software must have it!

Demos from unreleased software / Difficulty to release in production

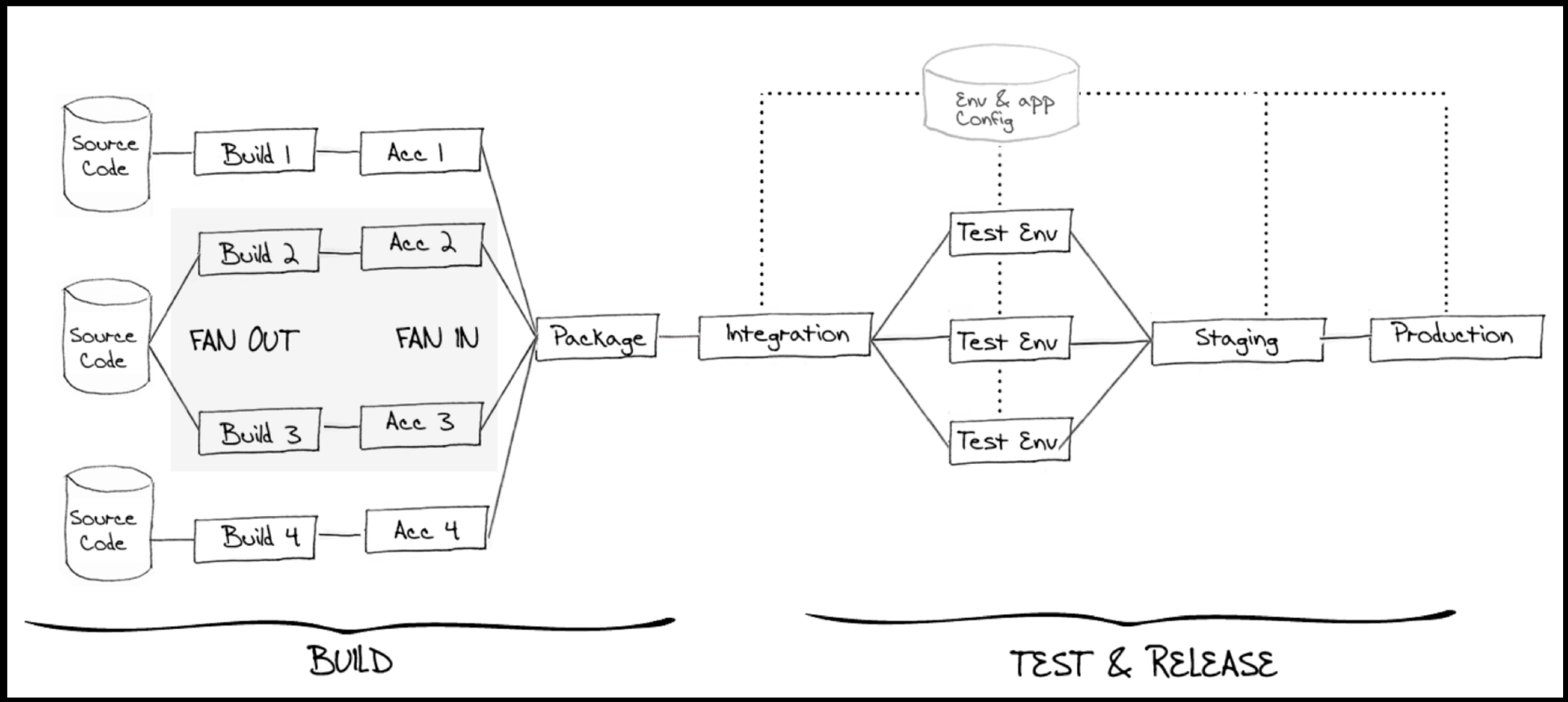
Involve Ops people in sprints
Releasing in prod should be doable by anyone, anytime, using a single click
(rollback is obviously a feature you will want!)

CD–Continuous Delivery/Deployment

Continuous Delivery/Deployment

Always have a shippable version available for your customers

Ex: GO CD (from Thoughtworks - now Open Source)
You can integrate your CI servers (Jenkins) with a CD server



Continuous Delivery/Deployment

A good practice to deploy gradually using cascaded environments:

- (Development)

- Test
  - Less resources, used mostly to validate business logic

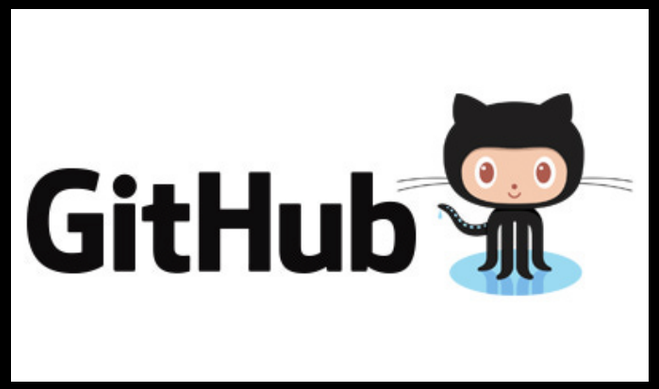
- Staging/Pre-Production
  - More representative of the production environment. Can be used for load/performance testing. Typically uses a data set that is a copy of the Production data set.

- Production

IaC–Infrastructure as Code

Infrastructure as Code

Your code is under CM, but your infrastructure is typically not! It also needs to be versioned, tracked and automated!



Infrastructure as Code

There are so many tools available. But essentially, keep **ALL** under source control, including what it takes to reproduce your production environment from **ZERO**

- How to restart a hardware environment/virtual machines
- How to install the platform on the machines
- How to install the applications on the platforms
- How to configure the whole stack
- The databases schemas and content
- **Everything!**

Infrastructure as Code

If you are not convinced, think about your disaster recovery plan (you have one, right?)

What if your server room is destroyed by water/ fire? (or the one from your cheap cloud provider…)

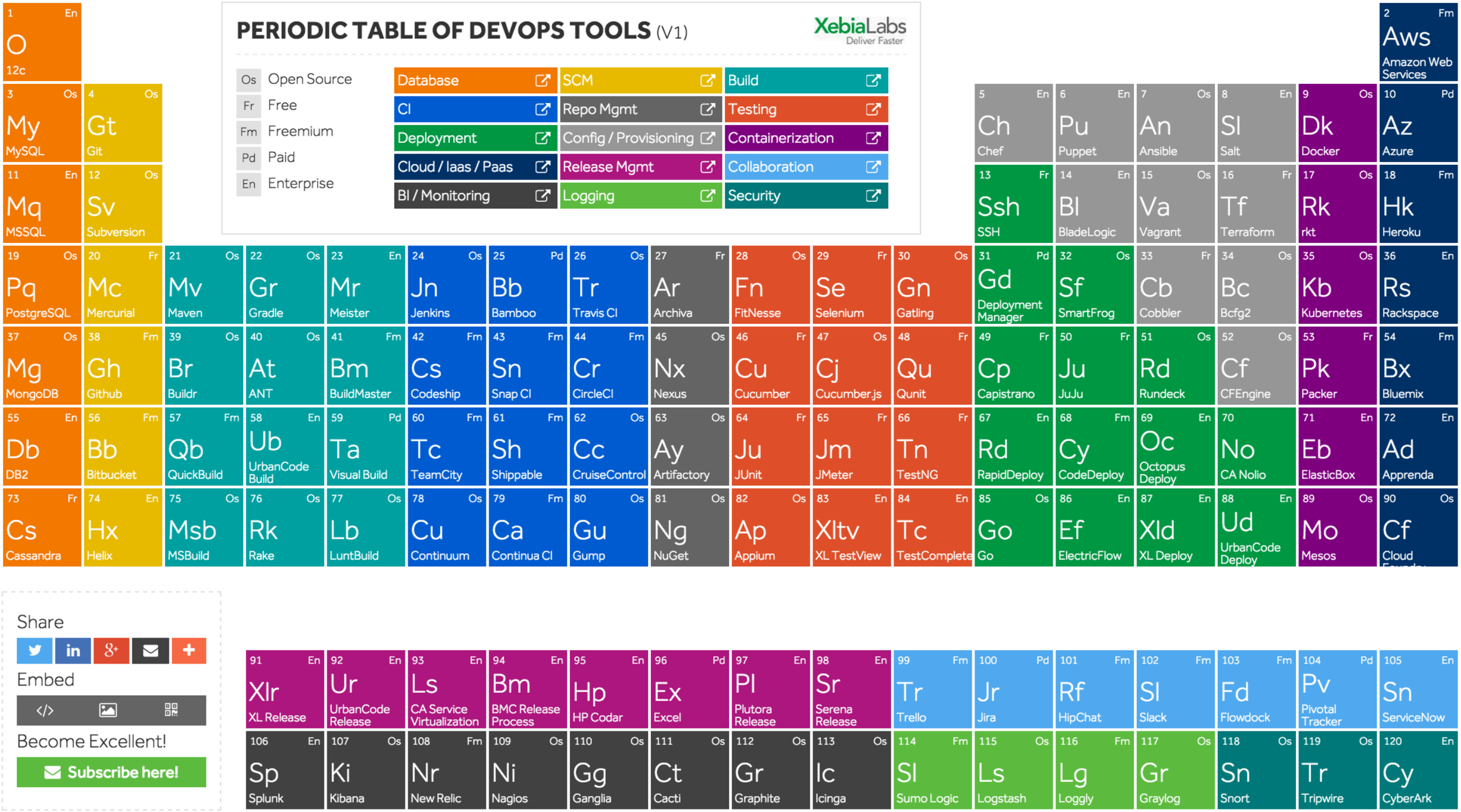
With IaC at least, the software part is covered in case of disaster

DevOps

DevOps

- A Good app without an infrastructure to run it is useless…
- A superb, scalable infrastructure without an app adding business value to a customer is also useless…
- You need both!
- DevOps should not be a separate team! It serves to bridge the gap between development and operations teams.
- And if you are not convinced that DevOps cannot typically be a single person’s responsibility…

DevOps related tools



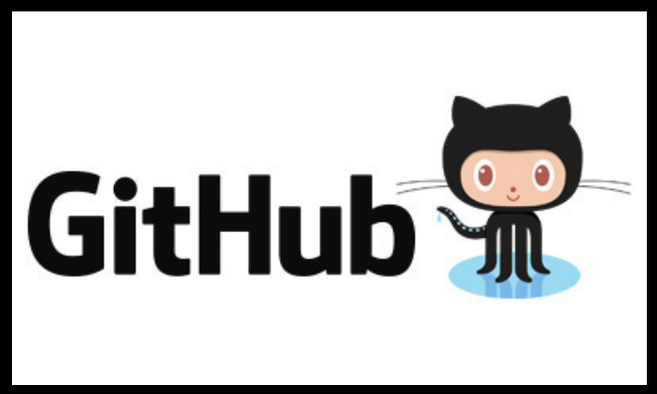
What could be potential solutions to deliver and faster?

1- Testing (TDD, BDD)
2- CI
3- CD
4- IaC

DEMO

Demo content

- GO CD presentation
  - Application build pipeline
    - connected to GitHub for app code
    - running unit tests
  - Triggering Staging pipeline on success
    - using Vagrant + VirtualBox + Ansible to provision production like environment for system tests
      - connected to GitHub for IaC code
- Triggering Deployment pipeline on success
  - using Ansible (Tower) to provision non VM production multiple environments



Application Build/Unit tests

Continuous Integration pipeline
Triggered on code changes

Staging pipeline

For deployment + system testing
Triggered on new application integration OR new IaC code

Deployment pipelines

Used to actually deploy the application into production
triggered on successful staging pipelines OR manually when needed

Discussion

How do you do it?

Agile Values vs.

Agile Practices

CRAFTSMANSHIP MANIFESTO

http://manifesto.softwarecraftsmanship.org/

NOT ONLY WORKING SOFTWARE,
BUT ALSO **WELL-CRAFTED SOFTWARE**
NOT ONLY RESPONDING TO CHANGE,
BUT ALSO **STEADILY ADDING VALUE**
NOT ONLY INDIVIDUALS AND INTERACTIONS,
BUT ALSO A **COMMUNITY OF PROFESSIONALS**
NOT ONLY CUSTOMER COLLABORATION,
BUT ALSO **PRODUCTIVE PARTNERSHIPS**

Agility is more than Agile values and ceremonies

It is acting as per the Agile values and producing software with related best practices

But where to start?

Every Agile cycle, try to improve on those issues, trying to automate everything you can, while having the conversation with your key stakeholders

Questions or comments?

Thanks!

softwarethatmattersdoneright.com

+

http://ca.linkedin.com/in/stevemercier

Backup

Install provisioning tools
Ex: Vagrant + VirtualBox

- vagrant init hashicorp/precise64
- vagrant up
- vagrant ssh
- vagrant destroy

Ex: See how using the Vagrant file, we can provision the platform

config.vm.box = “hashicorp/precise64"
config.vm.network :forwarded_port, guest: 80, host: 4567
config.vm.provision :shell, path: “bootstrap.sh"

#!/usr/bin/env bash

apt-get update
apt-get install -y apache2
if ! [-L /var/www]; then
rm -rf /var/www
ln -fs /vagrant /var/www
fi
apt-get install -y python3

How to install the applications on the platforms

Ex: We can simply reuse the GIT clone using a shared folder between host and guest VM or use similar platform provisioning technique to perform a git clone in the VM after provisioning