devops jungle of tools

Infrastructure v/s **Deployment** automation

@rantav

CTO @ Social Studios TV

What is Social Studios?

What is Social Studios?

What is Social Studios?

What is Social Studios?

facebook

Jan 25th, 2013
Ran Tavory

CREATIVE DIRECTOR

Yaeli Avnery

COPYWRITERS

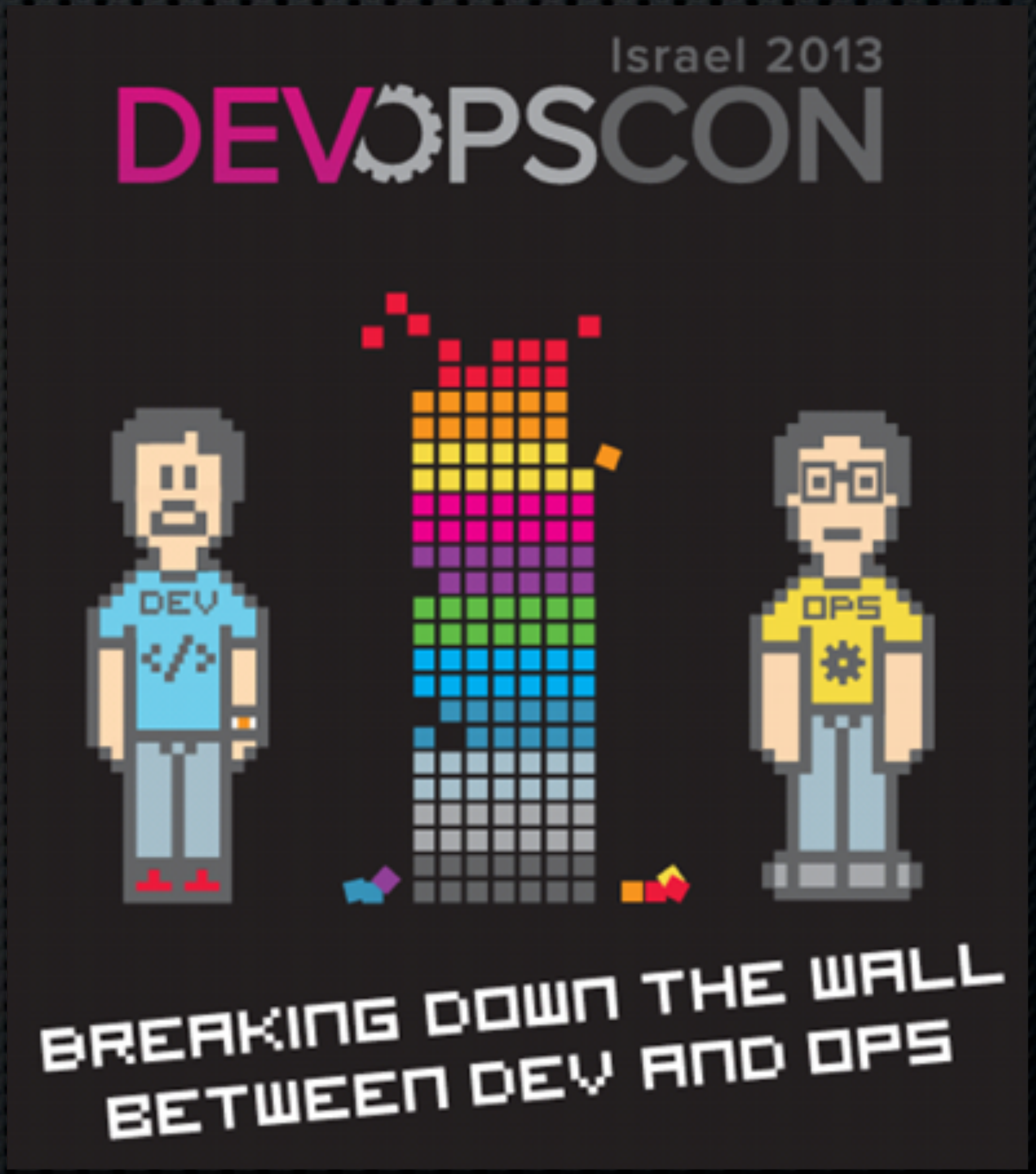
Yossi Gendler

Dan Barak

Guy Tavor

David Schwartz

devops
jungle of
tools



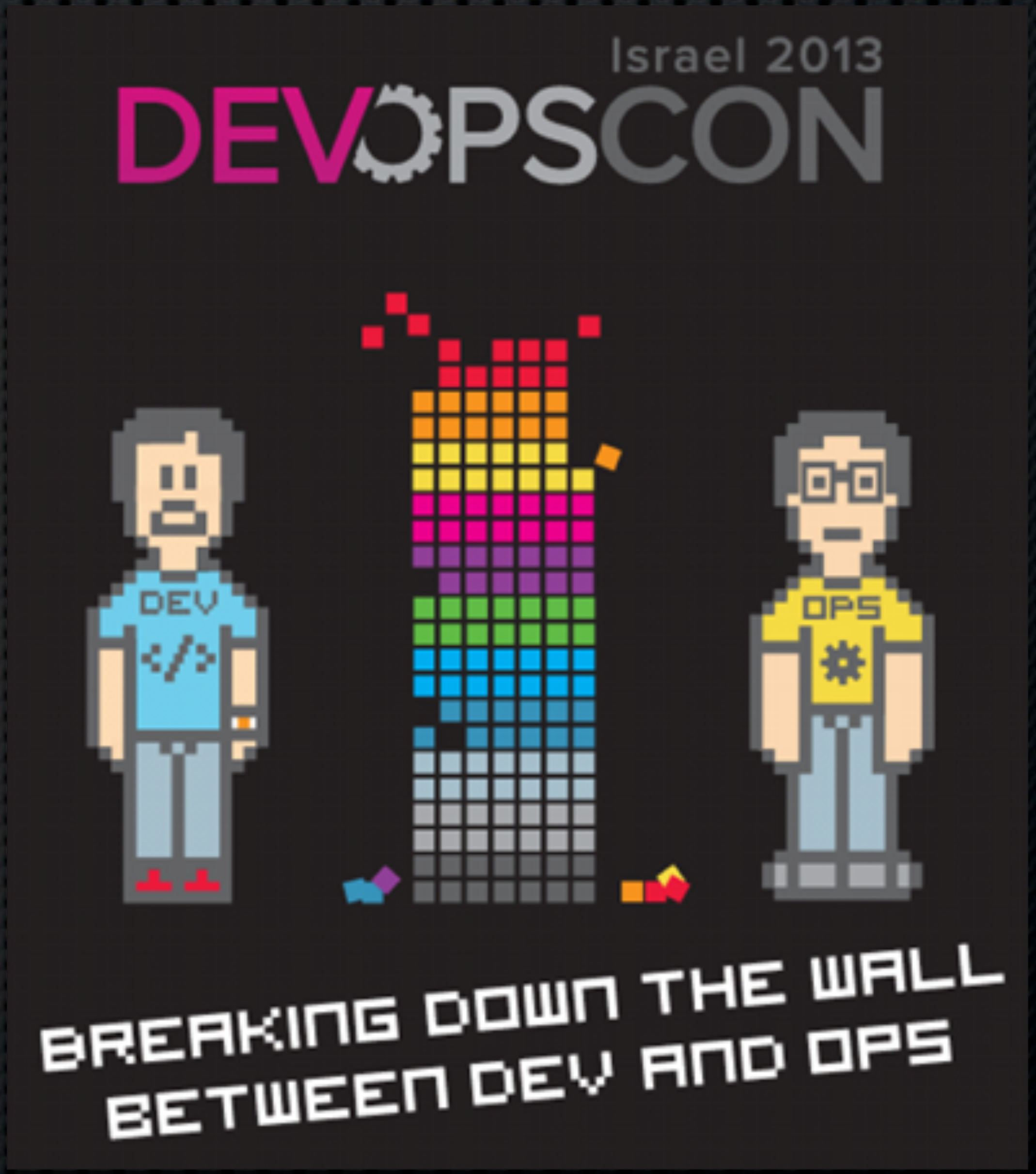
As learnt by - Myself at

Outbrain

Start with a personal story

My goal was to:

- ✓ Deploy apps
- ✓ Maintain infra



Start with a personal story

That's me 3 years ago.

Confused

Chef

Puppet

Control Tier

CFEngine

...

???

What should I do?

What should I choose?

I have to maintain infrastructure

But I also have to deploy my apps

Are these the same?

Aha !!!

A h a ! ! !

I need a tool that:

Aha !!!

I need a tool that:

- Maintains Infrastructure

Aha !!!

I need a tool that:

Maintains Infrastructure

AND

Deploy my apps!

Maintain Infrastructure

AND

Deploy my apps ???

I didn’t find it

Maintain Infrastructure

AND

Deploy my apps ???

???

Maintain Infrastructure

AND

Deploy my apps ???

But Why ???

???

Maintain Infrastructure

AND

Deploy my apps ???

They are not the same!

Infrastructure

≠

App Deployment

Oh...

They are not the same!

Infrastructure

≠

App Deployment

Why are they different?

And what do they have in common?

And what did I end up using?

What did I use?

- ***Case 1:***
  - ***Chef***
  - ***glu***

What does chef do?

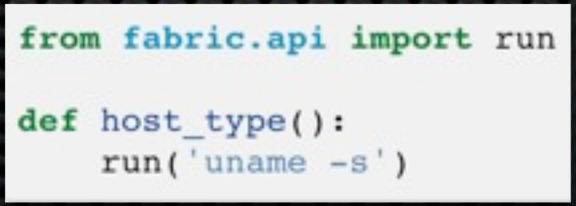
- Installs infrastructure
  - java
  - databases
  - etc...
- Installs Glu
  - server and agents

What does glu do?

- Deploy our apps
  - Tomcat based apps

What did I use?

- **Case 2:**
  - **Chef**
  - **Fabric**



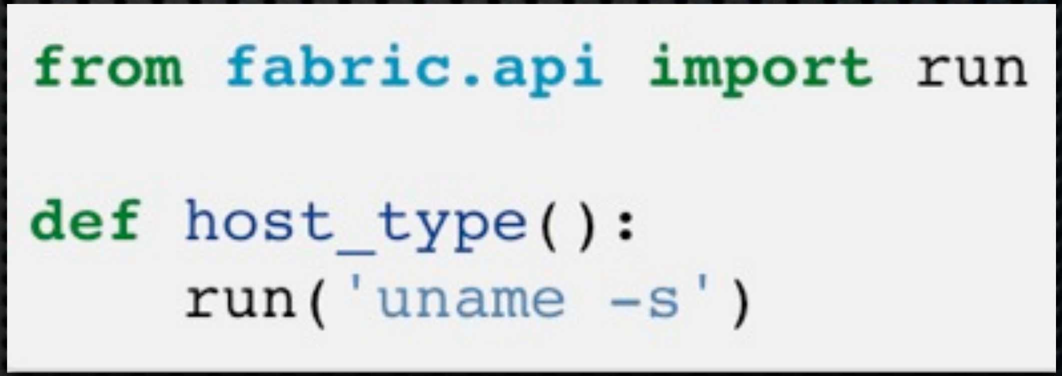
```
from fabric.api import run

def host_type():
    run('uname -s')
```

What does chef do?

- Provision servers
- Install Java, Tomcat
- Install DBs
- Set up users, keys
- logstash
- nagios, etc...

What does Fabric do?

- Deploys apps
  - Tomcat based
  - or jetty
  - Notifications
  - tests



```
from fabric.api import run

def host_type():
    run('uname -s')
```

Isn’t it easier to use just one tool?

Yes!

So why am I using two tools?

- Because they are different

Let's take a look at the tools now

Infrastructure

≠

App Deployment !!!

Chef

Recipes, Resources,

Convergence.

Puppet

Define Desired State

Enforce

Monitor

CFEngine

- Desired State
- Self Healing
- Monitor

Control Tier

Command Dispatcher

ControlTier

Deployment Automation

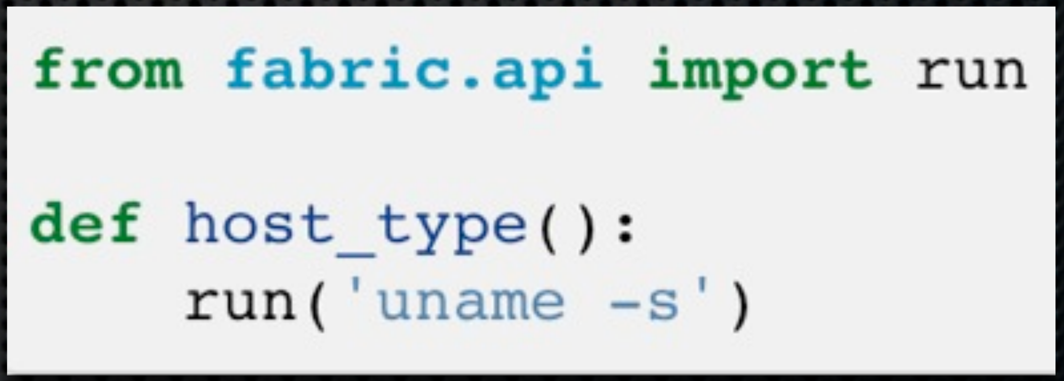
glu

Deployment Automation and Monitoring

fabric

Deployment

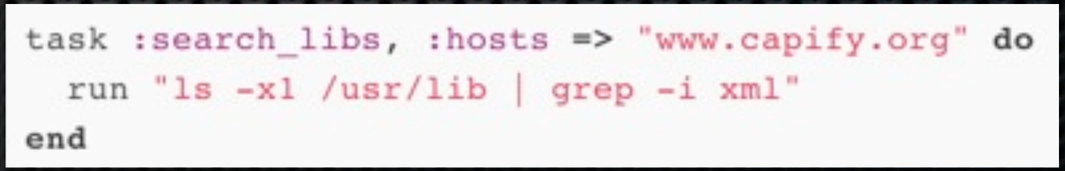
and administration



```
from fabric.api import run

def host_type():
    run('uname -s')
```

Capistrano

Remote server automation

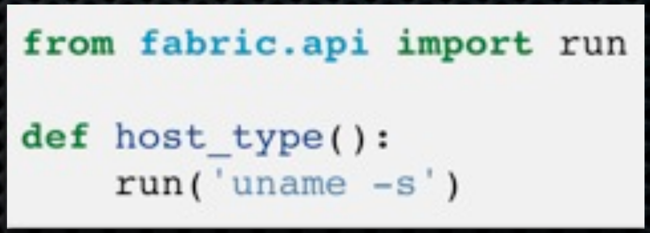


```
task :search_libs, :hosts => "www.capify.org" do
  run "ls -x1 /usr/lib | grep -i xml"
end
```

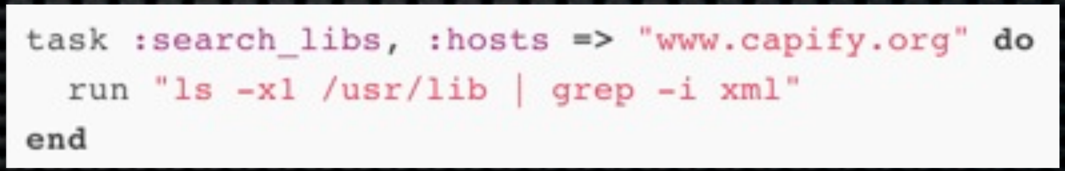
Hybrid?

- So - Chef, Puppet, CFEngine **maintain infra**
  - But - can they also **deploy applications?**
  - They could...
  - But - it’s awkward ☹
  - Example:
    - Use shef for ad-hoc tasks.

Hybrid?



```python
from fabric.api import run

def host_type():
    run('uname -s')
```



```ruby
task :search_libs, :hosts => "www.capify.org" do
  run "ls -xl /usr/lib | grep -i xml"
end
```

- So - glu, fab, cap, CTier can **deploy**
  - But - can they also **maintain infra**?
  - They could...
  - But - it's awkward ☹
  - Example: Use fabric to deploy mysql.

Why awkward?

Because **Infrastructure** ≠ **Deployment** automation

How is it different?

Let’s see...

How is it different?

Infrastructure and application are different in a few ways

Confidence

Confidence

Different level of Confidence

- conf(linux || mysql) > conf(in-house apps)

Confidence

≠

Different level of Confidence

- conf(linux || mysql) > conf(in-house apps)

Widely used systems
(linux, mysql)

Confidence

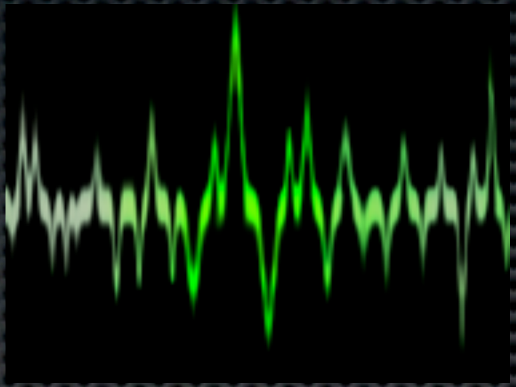
Different level of Confidence

- conf(linux || mysql) > conf(in-house apps)

Widely used systems (linux, mysql)

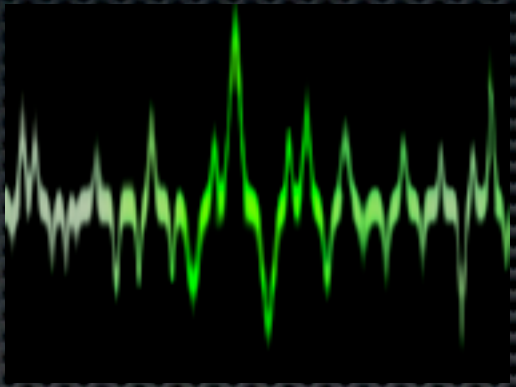
vs in-house apps, limited testing

Frequency



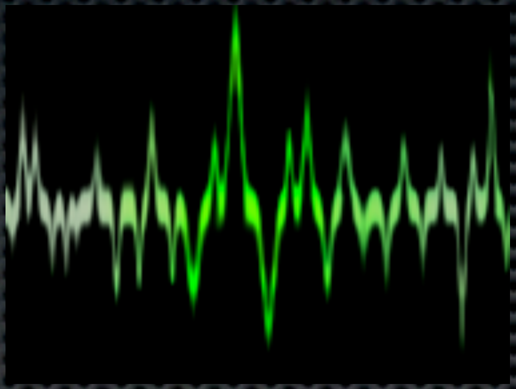
≠

Frequency



Frequency of change

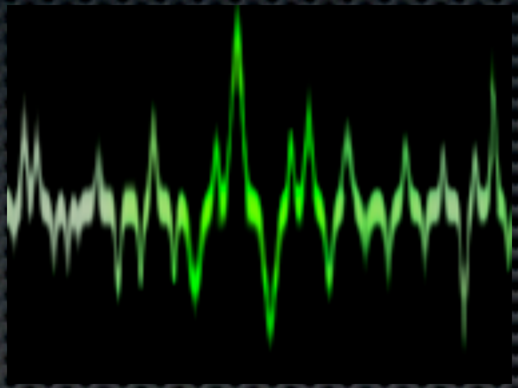
Frequency



Frequency of change

- freq(deploy database) ≪ freq(deploy new version)

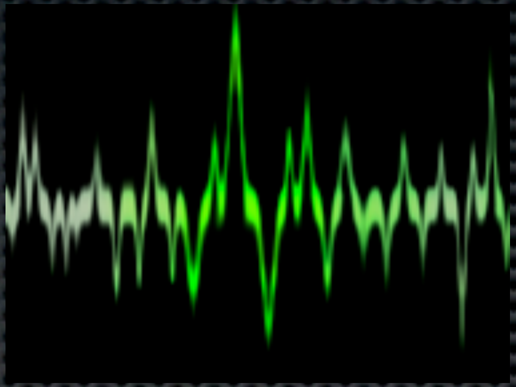
Frequency



Frequency of change

- freq(deploy database) ≪ freq(deploy new version)
- How often do you deploy a new DB?
  - every couple of months / years

Frequency



Frequency of change

- freq(deploy database) ≪ freq(deploy new version)
- How often do you deploy a new DB?
  - every couple of months / years
- How often do you deploy new apps?
  - Dozens a day

Control

≠

Control

Control over the actual process

Control

Control over the actual process

- Deployments:

Control

Control over the actual process

- Deployments:
  - Control exactly when they happen

Control

Control over the actual process

- Deployments:
  - Control exactly when they happen
  - Notify ppl, monitoring systems, with progress

Control

Control over the actual process

- Deployments:
  - Control exactly when they happen
  - Notify ppl, monitoring systems, with progress
  - Gradual, controlled and cautious deployments

Control

Control over the actual process

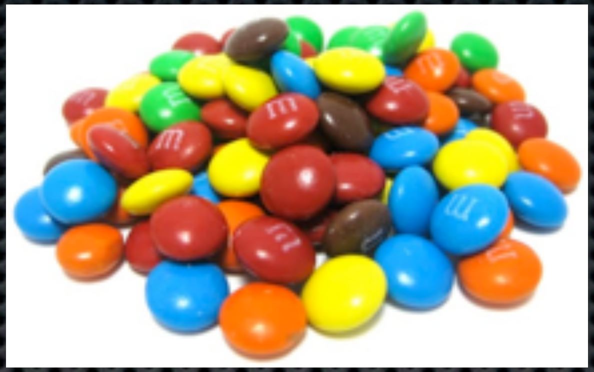
- Deployments:
  - Control exactly when they happen
  - Notify ppl, monitoring systems, with progress
  - Gradual, controlled and cautious deployments
  - Test as you go

Control

Control over the actual process

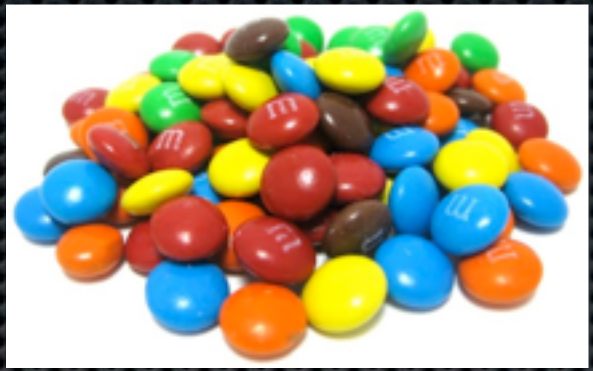
- Deployments:
  - Control exactly when they happen
  - Notify ppl, monitoring systems, with progress
  - Gradual, controlled and cautious deployments
  - Test as you go
  - Maybe rollback

Heterogenous
Homogenous



≠

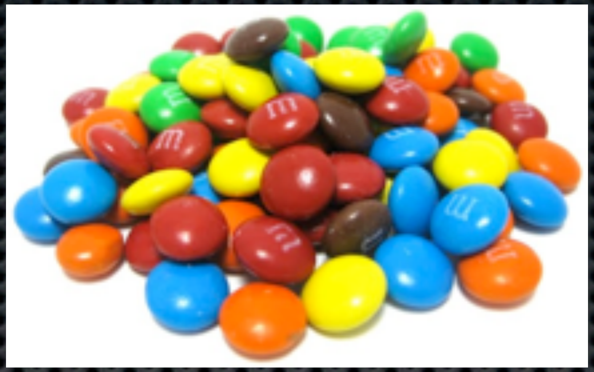
Heterogenous
Homogenous



≠

Heterogeneous v/s Homogenous

Heterogenous Homogenous

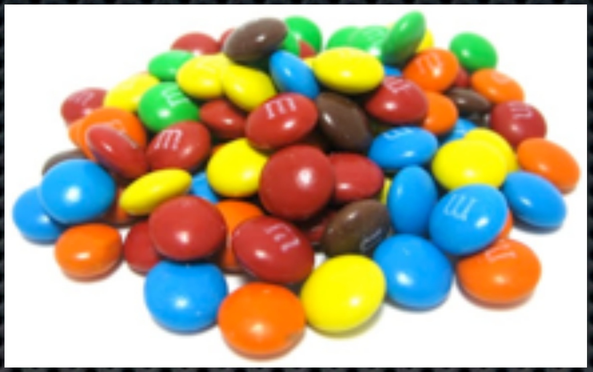


≠

Heterogeneous v/s Homogenous

- **Infrastructure** lives in **Heterogeneous** environments
  - Example: install mysql on ubuntu, centos, osx, win

Heterogenous Homogenous



≠

Heterogeneous v/s Homogenous

- **Infrastructure** lives in **Heterogeneous** environments
  - Example: install mysql on ubuntu, centos, osx, win
- **Apps** live in **Homogenous** environments.
  - Example: Rails apps only need a Rack server
  - Example: Java apps need only a JVM

Who’s code is it?

Who’s code is it?

- When it’s your code you can:
  - Instrument it (healthcheck, deployment hooks)

Who's code is it?

- When it's your code you can:
  - Instrument it (healthcheck, deployment hooks)
- If it's not your code, you have less control
  - hope to get lucky
  - or hack around it...

Take Chef and Glu

- So, for example...

Where does Chef stand out?

- **Recipes** for almost anything
  - Databases, App Servers, Languages...

Where does Chef stand out?

~~Recipes for almost anything~~

~~Databases, App Servers, Languages...~~

Your code ⇒ no recipes

Chef

Where does Chef stand out?

- **Heterogeneous** environments
  - Any linux, windows, osx (resource providers)

Where does Chef stand out?

- ~~Heterogeneous environments~~

Deployment environments are Homogenous

ix, windows, osx (resource providers)

Where does Chef stand out?

- Runs **unattended**
  - to assure state

Where does Chef stand out?

Your want to monitor it

Where does Glu stand out?

- Fine control over the deployment process

Where does Glu stand out?

Packaged code ⇒ Not needed

Where does Chef stand out?

- Status update and monitoring during deployment

Where does Chef stand out?

Infrastructure update - usually taken offline

deployment automation platform glu

Where does Chef stand out?

- High frequency model change

Where does Chef stand out?

- ~~Low Frequency~~ model change

Compare

	Infrastructure	Deployment
Chef	✓	✓̸ (shef)
Puppet	✓	?
Glu	×	✓
Fabric	✓̸	✓

To sum up

- Chef ⇒ Infrastructure
- Glu / Fabric / Capistrano ⇒ Applications
- Yes - it’s more tools
- But - **Use the right tool for the job**...

To sum up

- Chef ⇒ Infrastructure
- Glu / Fabric / Capistrano ⇒ Applications
- Yes - it's more tools
- But - **Use the right tool for the job**...

What does the future hold?

- Immutable Servers?
  - aka Phoenix Servers
  - vs Snowflake Servers
- Pallet?

This presentation

Is here:

https://speakerdeck.com/rantav/devops-jungle-of-tools