IPPON

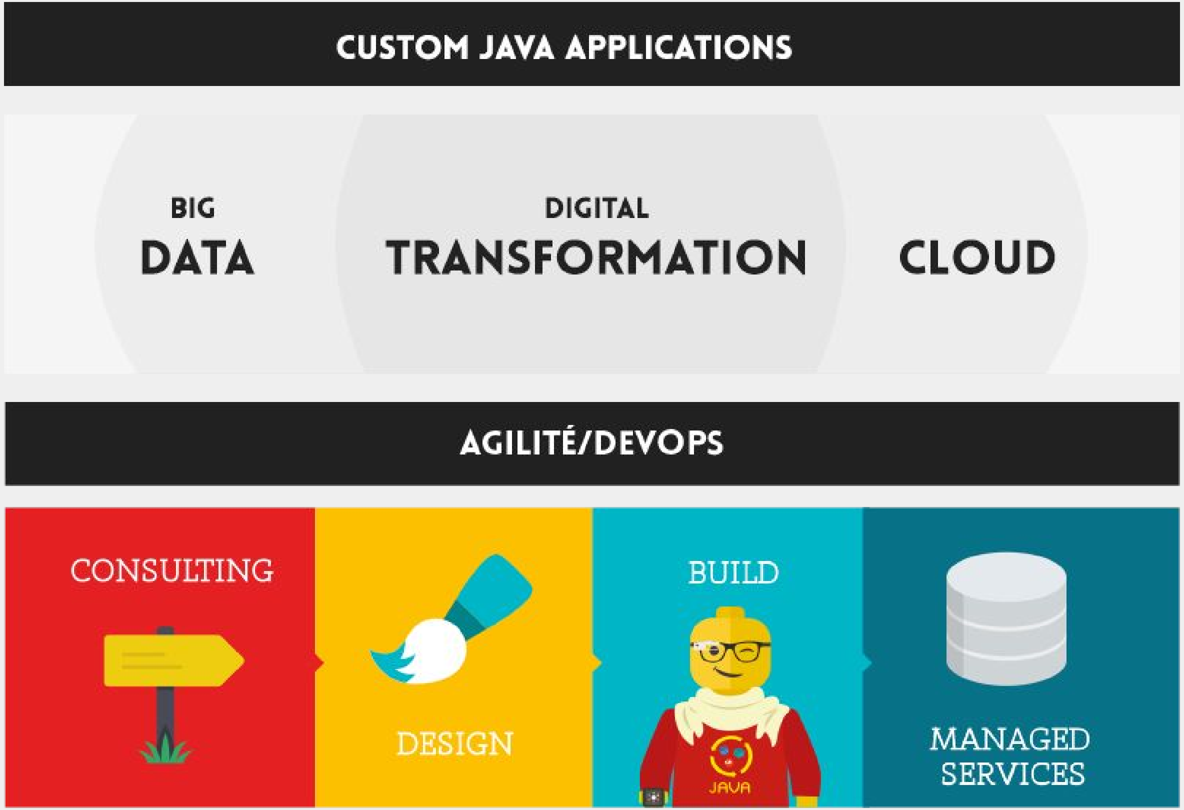
Digital . Technologies . Hosting

Dynamic platform, where are our µservices ?

#docker #dns #scalability...

About Ippon

- **250**
  salariés
- **20,5M€**
  CA 2015
- **24M€**
  prévisionnel 2016
- **6**
  agences
  · PARIS
  · BORDEAUX
  · NANTES
  · RICHMOND,VA
  · WASHINGTON
  · MELBOURNE



CUSTOM JAVA APPLICATIONS

BIG DATA

DIGITAL TRANSFORMATION

CLOUD

AGILITÉ/DEVOPS

CONSULTING

DESIGN

BUILD

MANAGED SERVICES

Us

Florian GARCIA

@Im_flog

#java #microservices #cloud

Jeremie MONSINJON

@jmonsinjon

#java #microservices

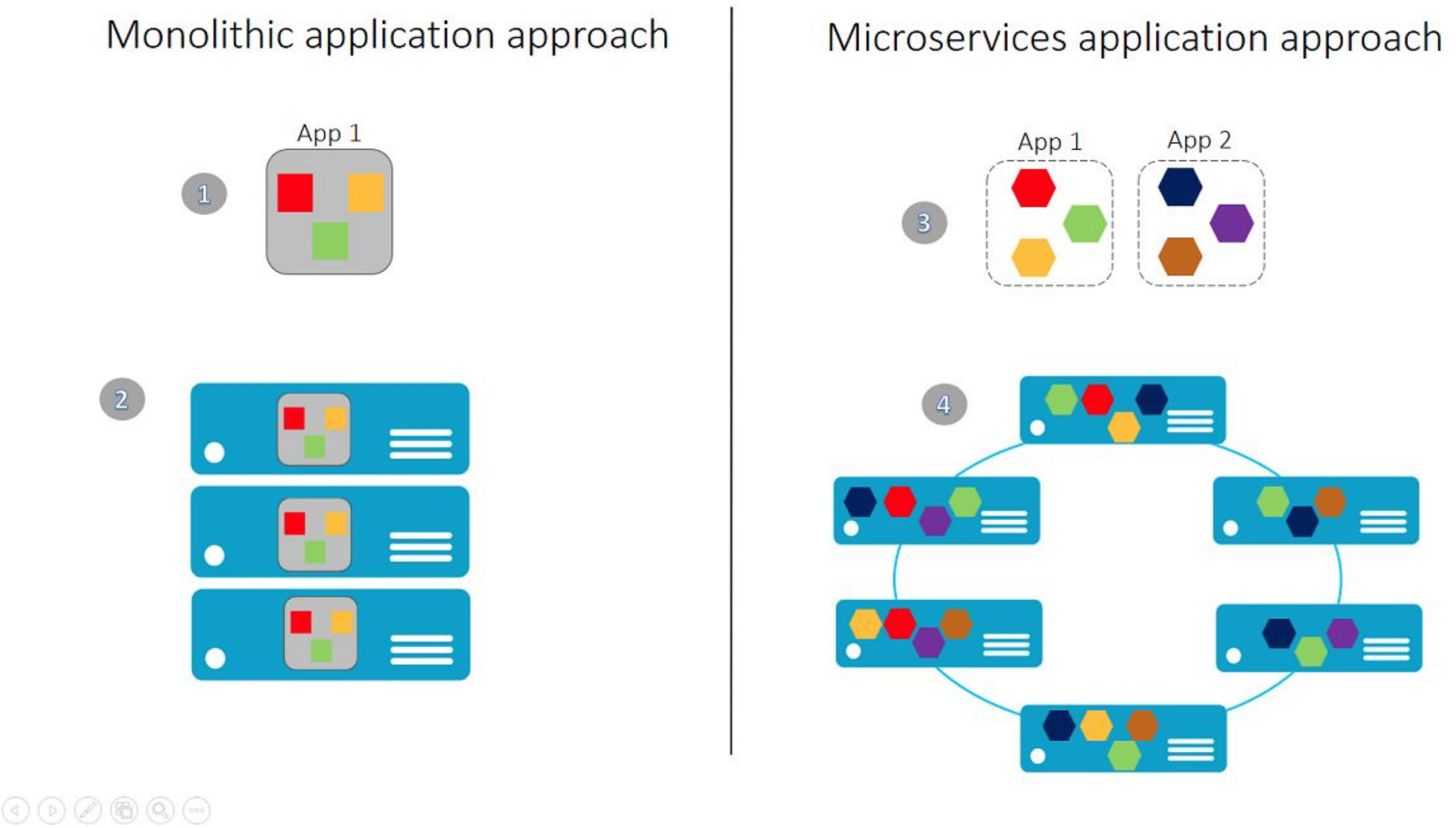
#elastic #docker

Plan

1. Microservices? Discovery? Why?
2. Java Discovery
3. Docker basics
4. Docker network
5. Docker Compose
6. Docker Swarm
7. Bonus: Experimental

Microservices

Where the complexity appears



Monolithic application approach

Microservices application approach

Network management

Monolith	Microservices
● Internal librairies calls	● Http calls
○ Easy to use	○ Introduce latency
○ Easy to trace	○ Introduce network failures
● No internal network to manage	● Hard to trace

Dynamic platform

- We need Scalability !
- We don’t care about the infra (cloud)
- We don’t care where are our applications
- We just want to know how many are “alive”

- But the applications need to know each other’s location !

The disco(very)

- Use of a “register” to store address and port for each microservice
- Client side register
- Common implementations:
  - Before each call, get an available service host and port from the registry
  - DNS for each service
- Central concept in a microservices architecture

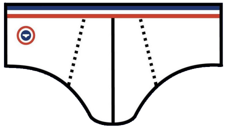
Disco(very) in Java

- Shared Key-Value Store
  - Zookeeper, Consul, etcd ...
- Custom mechanisms
  - Netflix Eureka, Serf ...
- DNS management
  - Spotify Apollo
  - Kong

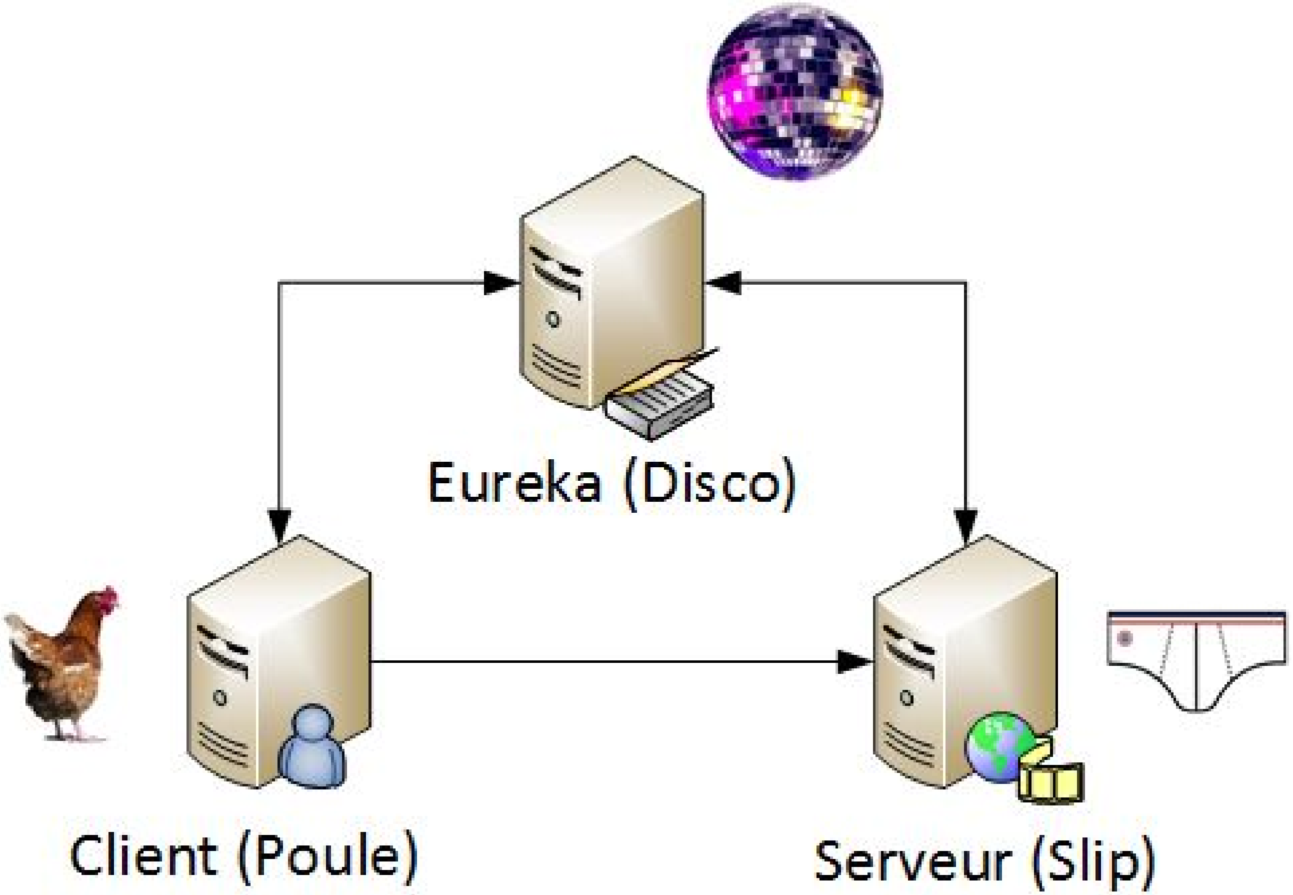
We will use Eureka, because it is very easy !

Demo context

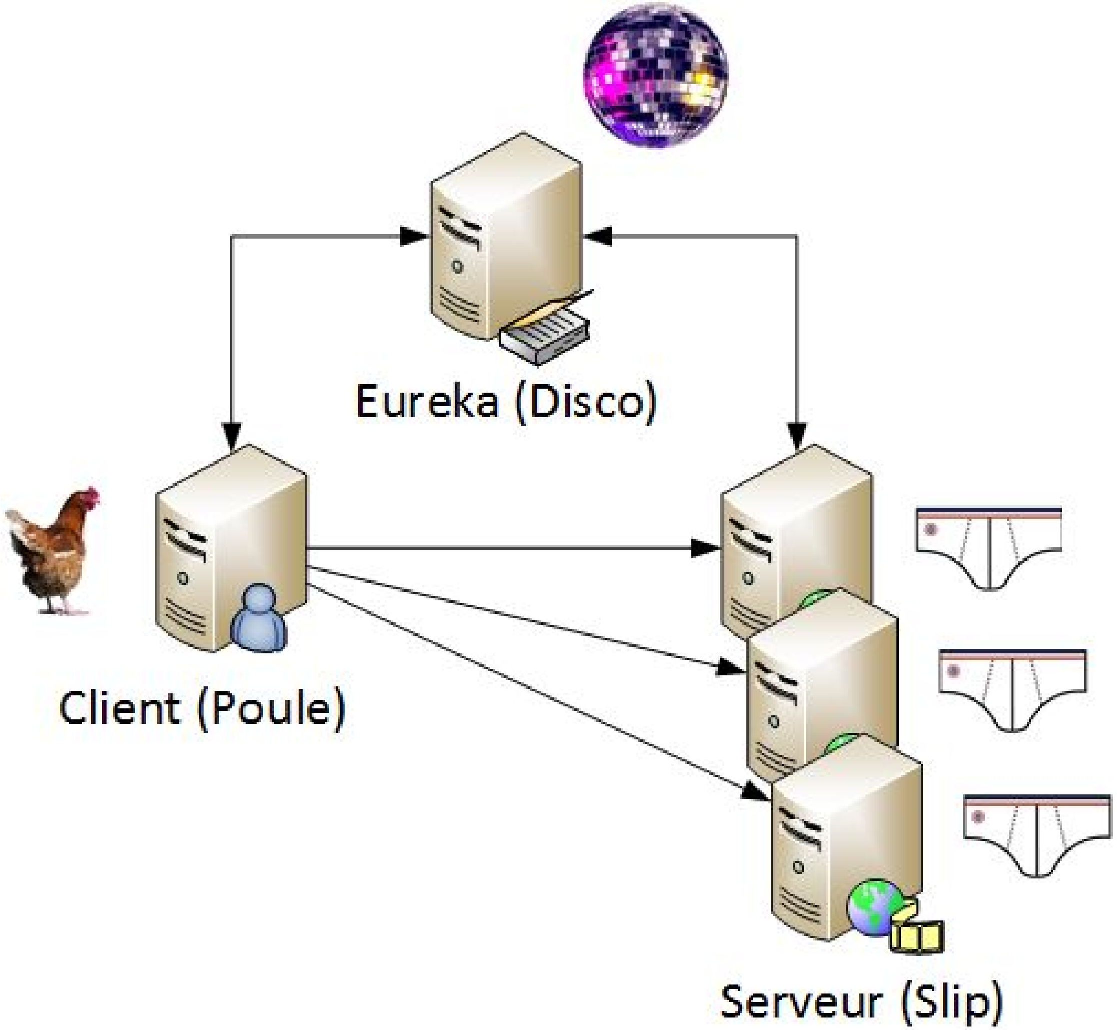
- Java / SpringBoot
- 3 services
  - **Slip**: listen http requests and for each one, wait 100ms before answering `$HOSTNAME`
  - **Poule**: Parallelize 10 calls to slip permanently
  - **Disco**: Manage the services discovery with Eureka



The app

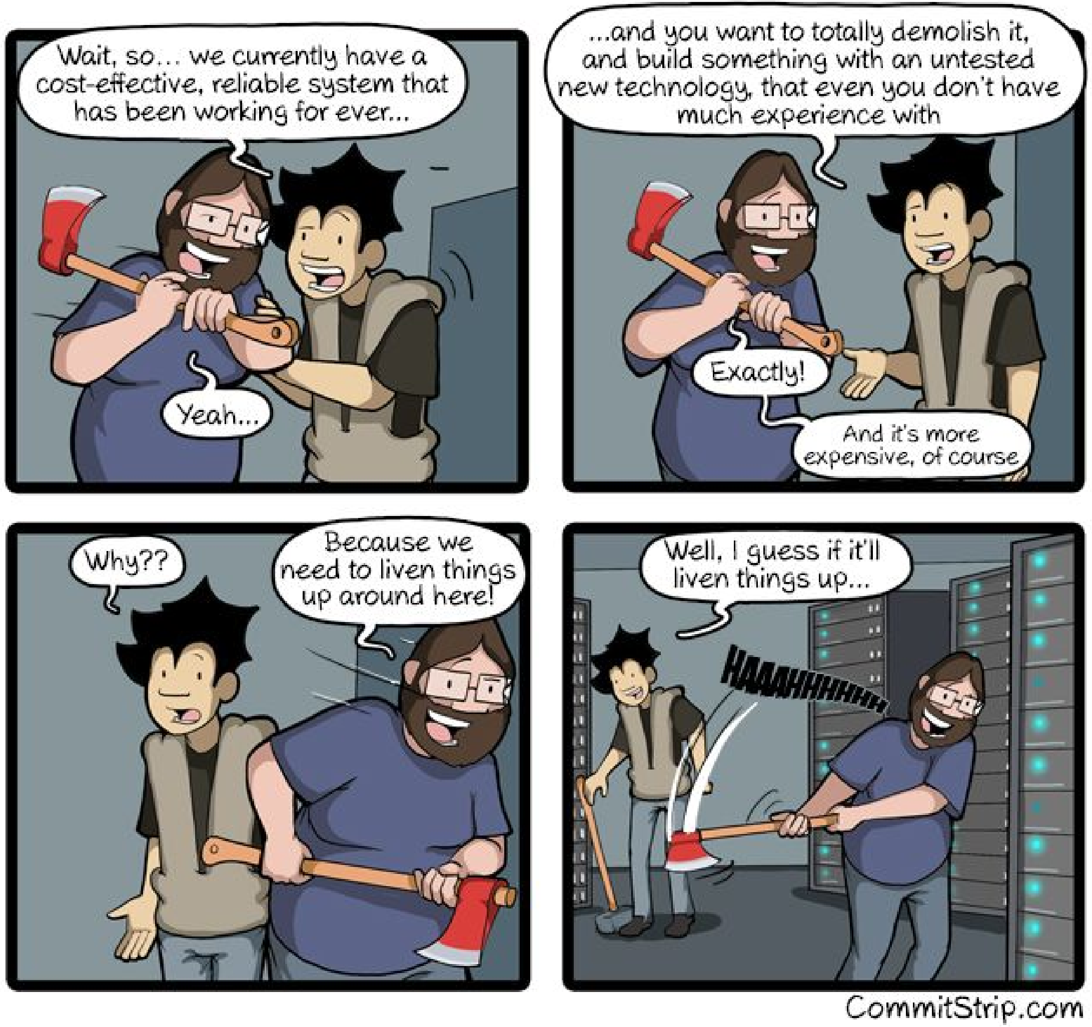


Scaling



Java discovery demo

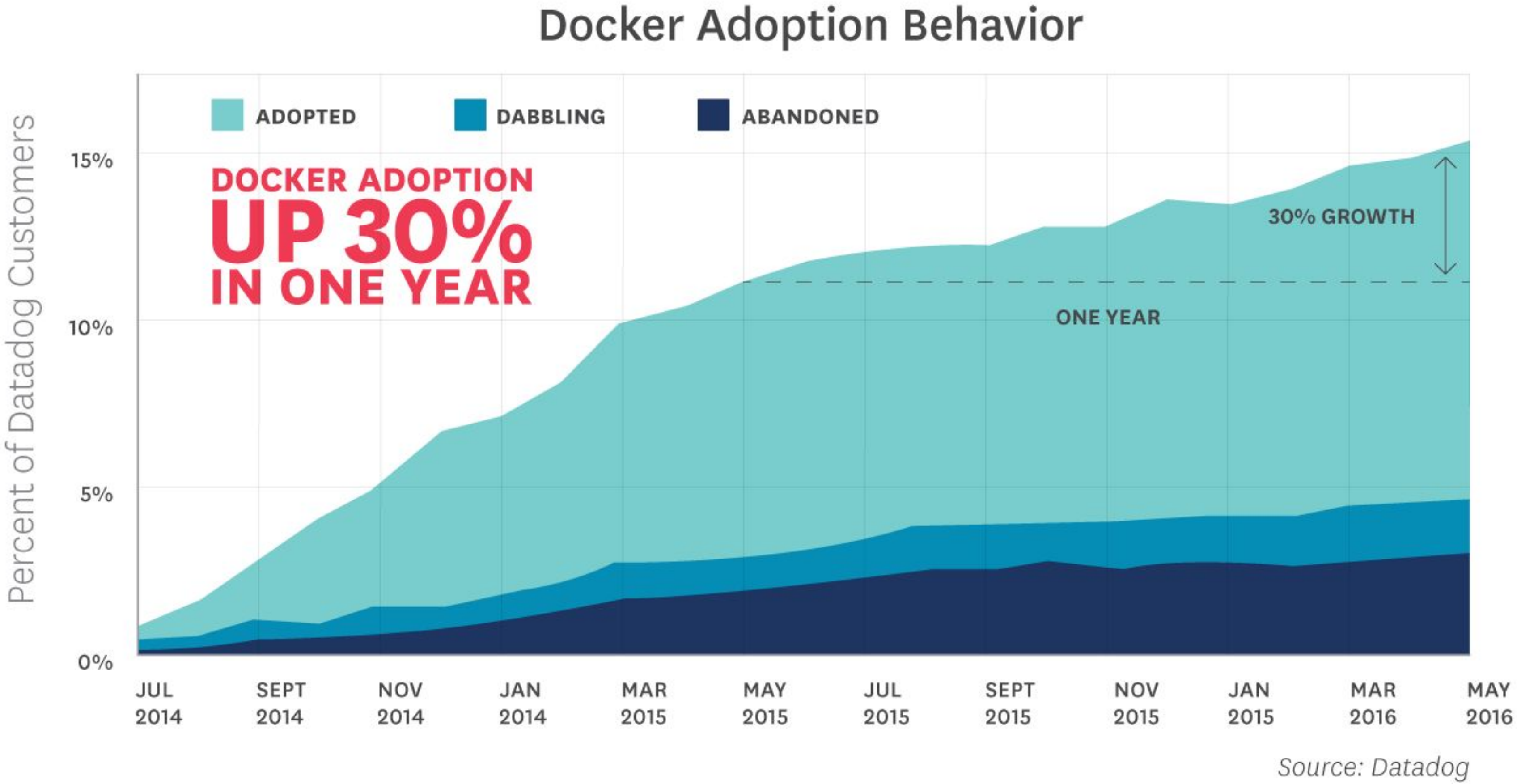
So you heard Docker was great !!!

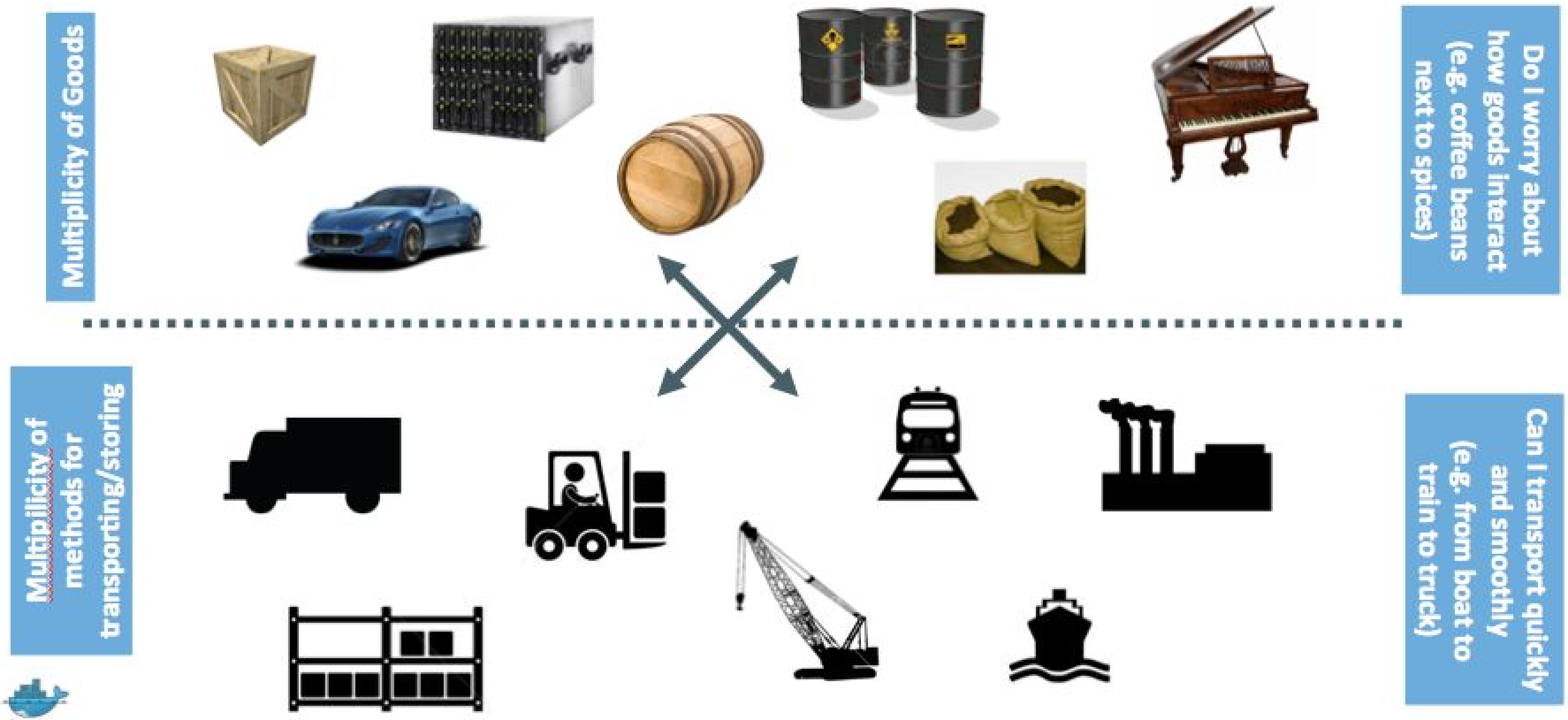

Wait, so… we currently have a cost-effective, reliable system that has been working for ever...
Yeah...
...and you want to totally demolish it, and build something with an untested new technology, that even you don't have much experience with
Exactly!
And it's more expensive, of course
Why??
Because we need to liven things up around here!
Well, I guess if it'll liven things up...
HAAAHHHHHHH
CommitStrip.com

Docker basics

Docker history

- Developed mostly by Solomon Hykes at dotCloud, a PAAS company. Distributed as an open source project since mars 2013.
- 29 july 2016, docker 1.12 release, includes built-in orchestration tools like Swarm.
- Currently 37 000 stars on GitHub, more than 10 000 forks and 1 500 contributors.
- Docker 1.13 => Soon


Docker Adoption Behavior
ADOPTED
DABBLING
ABANDONED
DOCKER ADOPTION
UP 30%
IN ONE YEAR
30% GROWTH
ONE YEAR
Percent of Datadog Customers
15%
10%
5%
0%
JUL 2014
SEPT 2014
NOV 2014
JAN 2014
MAR 2015
MAY 2015
JUL 2015
SEPT 2015
NOV 2015
JAN 2015
MAR 2016
MAY 2016
Source: Datadog

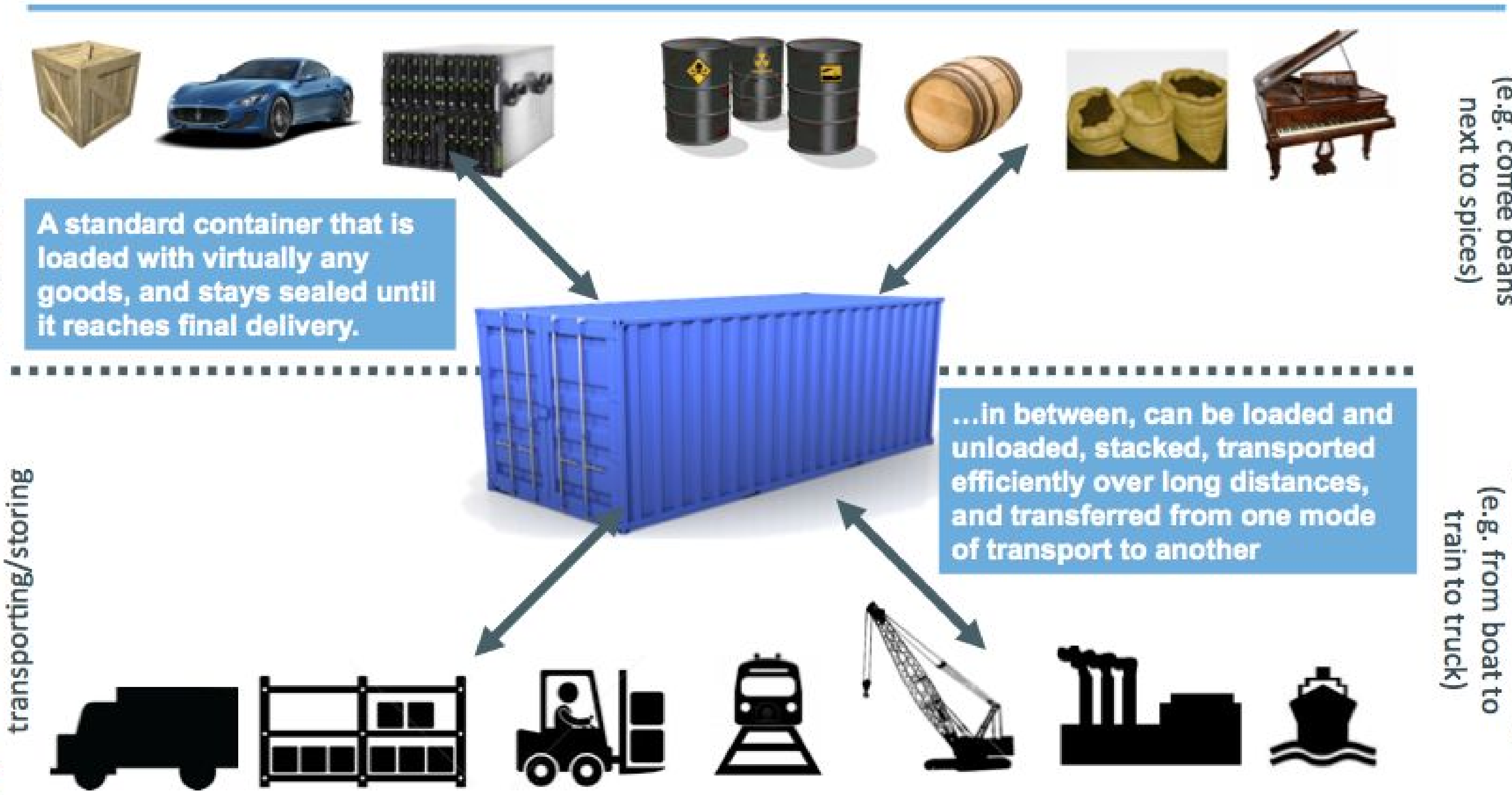


Multiplicity of Goods

Do I worry about how goods interact (e.g. coffee beans next to spices)

Multipilicity of methods for transporting/storing

Can I transport quickly and smoothly (e.g. from boat to train to truck)



Multiplicity of Goods

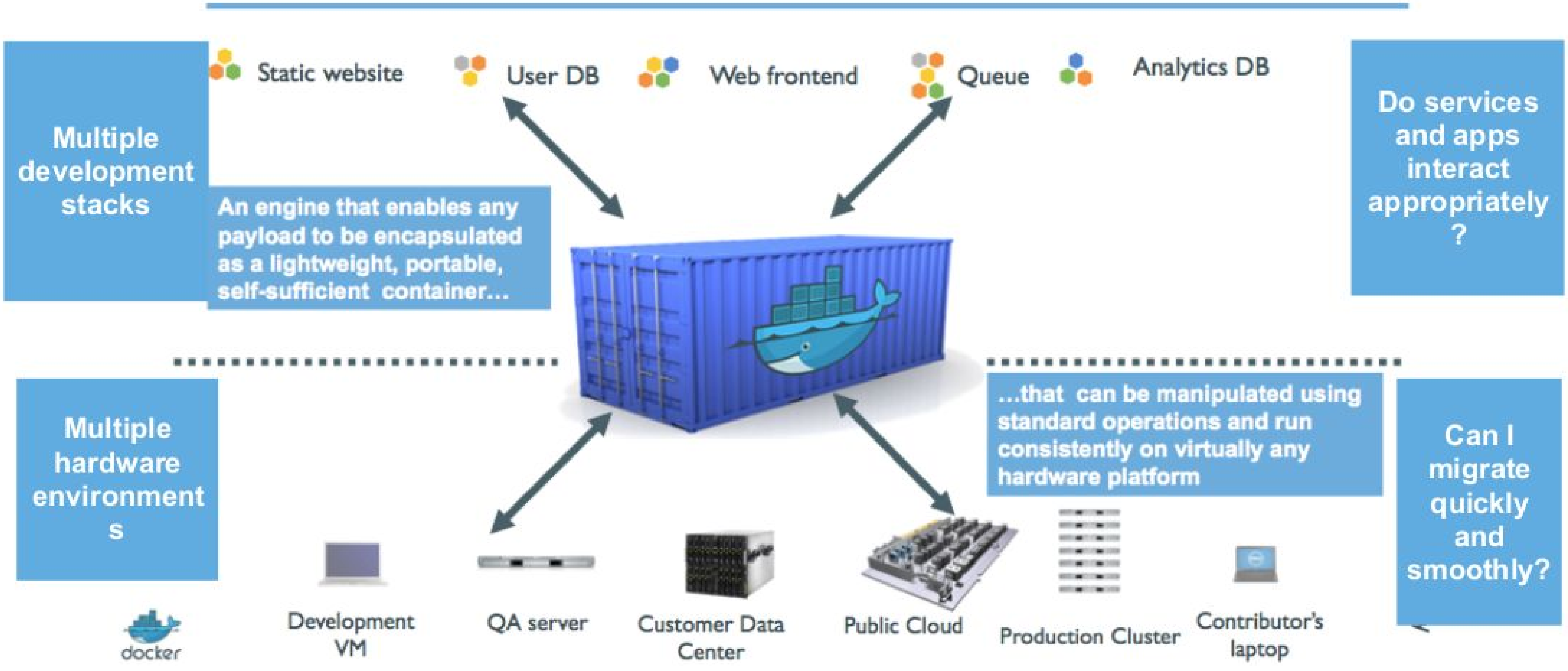
A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.

Do I worry about how goods interact (e.g. coffee beans next to spices)

Multiplicity of methods for transporting/storing

…in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another

Can I transport quickly and smoothly (e.g. from boat to train to truck)



Multiple development stacks

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container…

Do services and apps interact appropriately ?

Multiple hardware environments

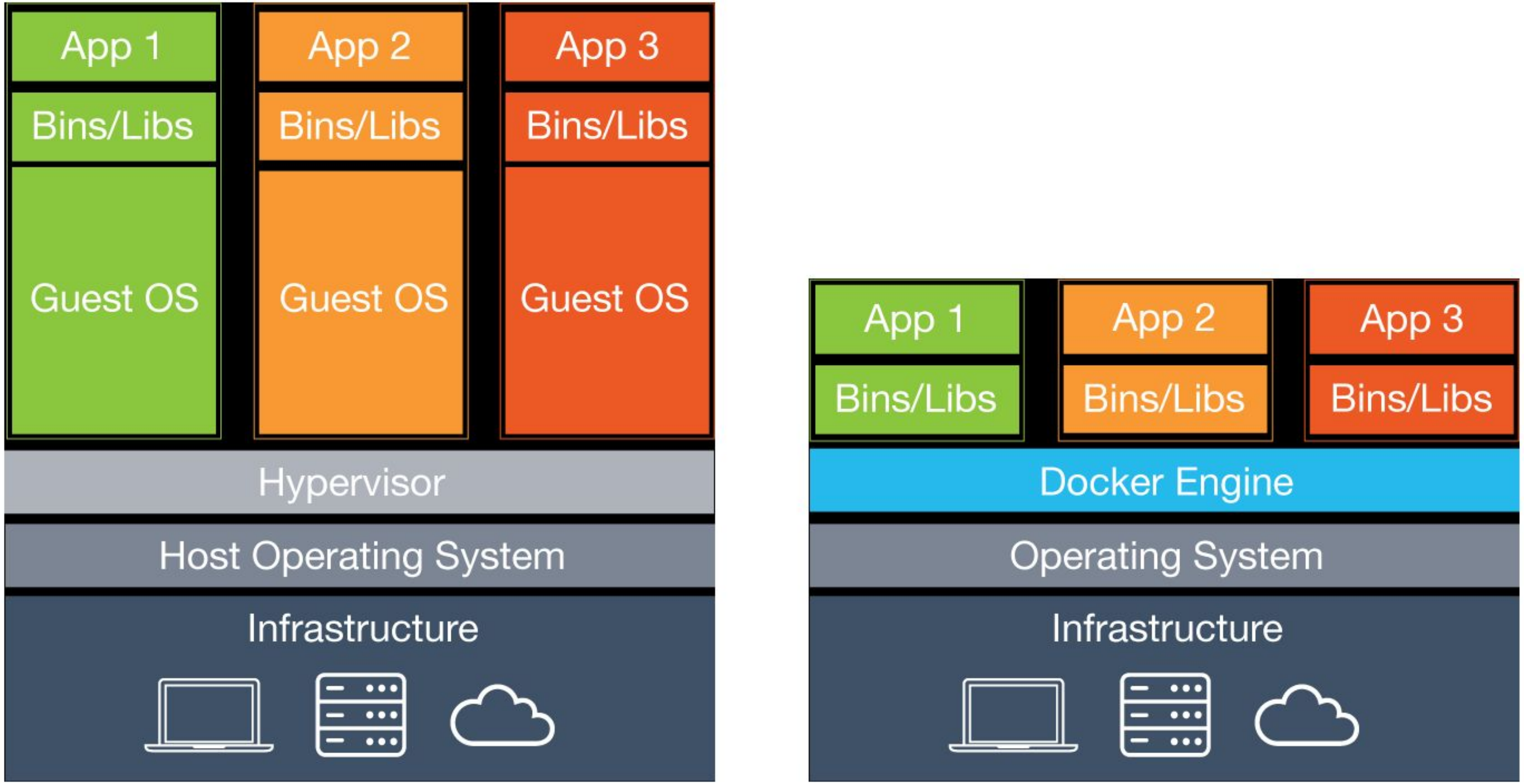
…that can be manipulated using standard operations and run consistently on virtually any hardware platform

Can I migrate quickly and smoothly?

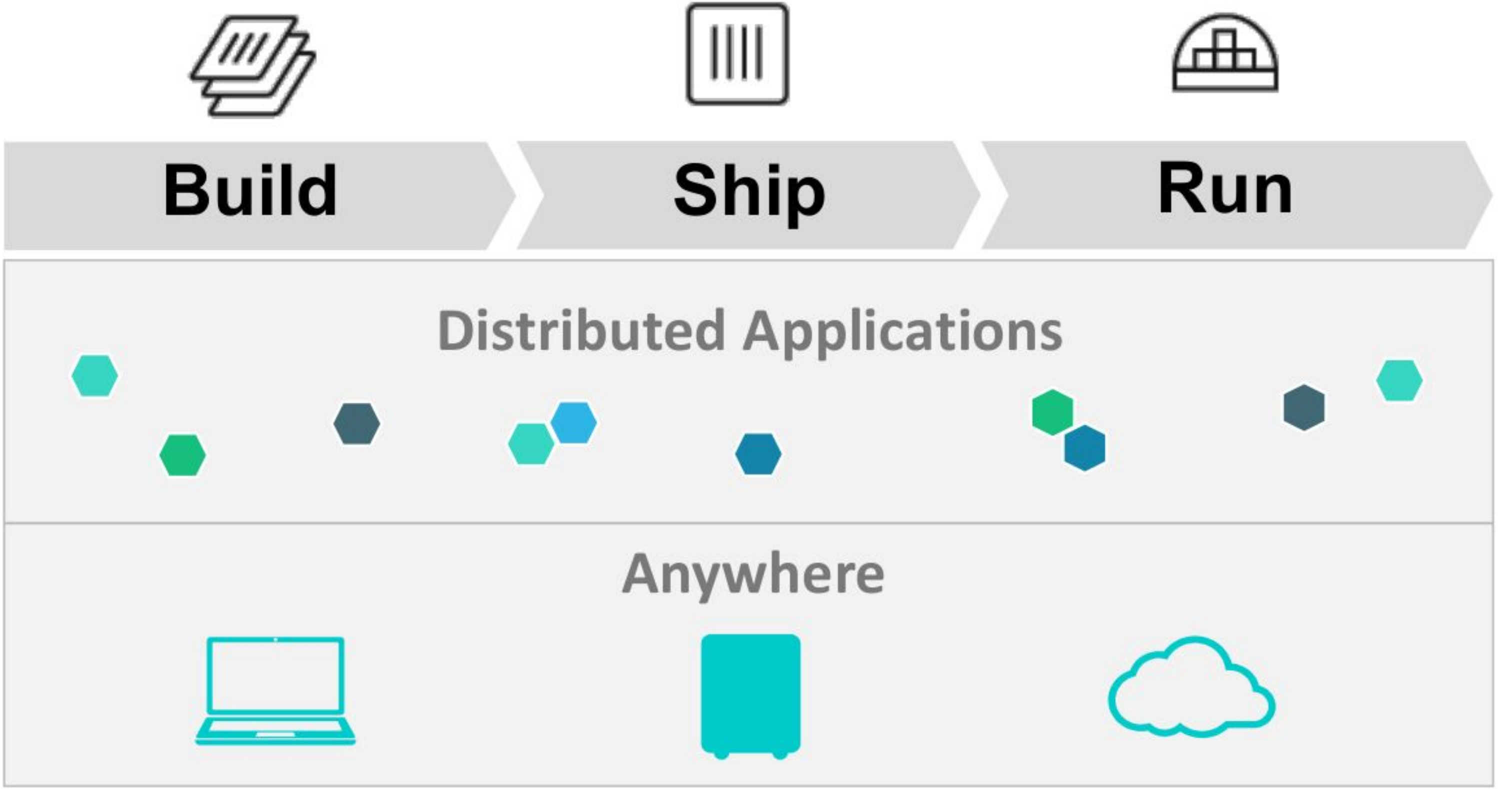
Hey guys, you're kind of quiet. Anyone want to troll us to wake us up?
Ah yeah. I've got an idea, but it's not really trolling...
Try anyway
Do you agree that Docker is just a kind of light-weight VM?
?!
NO
Pfff
You're crazy!
!
Docker isn't a VIRTUAL MAC
You can do "ONE CONTAINER per PROCESS" or "ONE CONTAINER per APP"
Just look at it for a second, God dammit!
It's a
for A SINGLE PROCESS ONLY!
Of course!
How could you SAY that?!

CommitStrip.com

Virtual machine VS Docker



The Docker mission



Build

Ship

Run

Distributed Applications

Anywhere

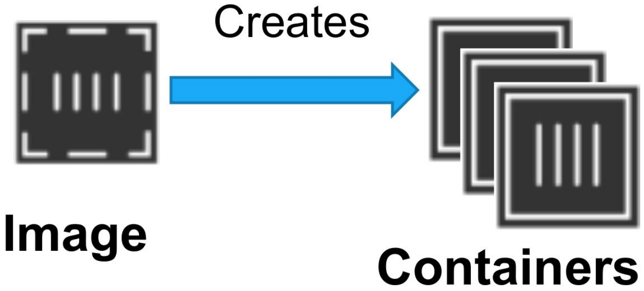
#Docker ?

Why would we change ?

● For the hype ?	Nope
● Standardize deployments ?	Yes
● Reproducibility ?	Yes
● Isolation ?	Yes
● Security ?	Kinda
● To scale ?	No
● First steps to the cloud ?	Kinda

Base concepts

- Cgroups & Namespaces isolation
- Images & Containers



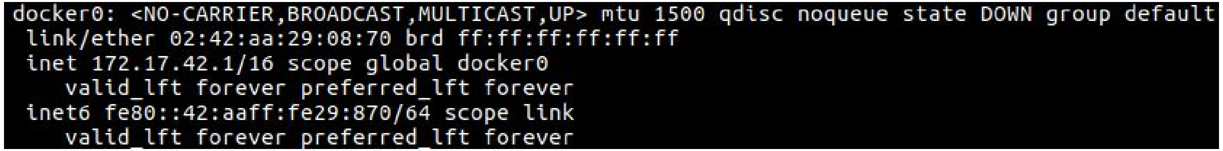
- Container = single process
- Container is “ephemeral”

Building docker images demo

Docker network

Network with Docker

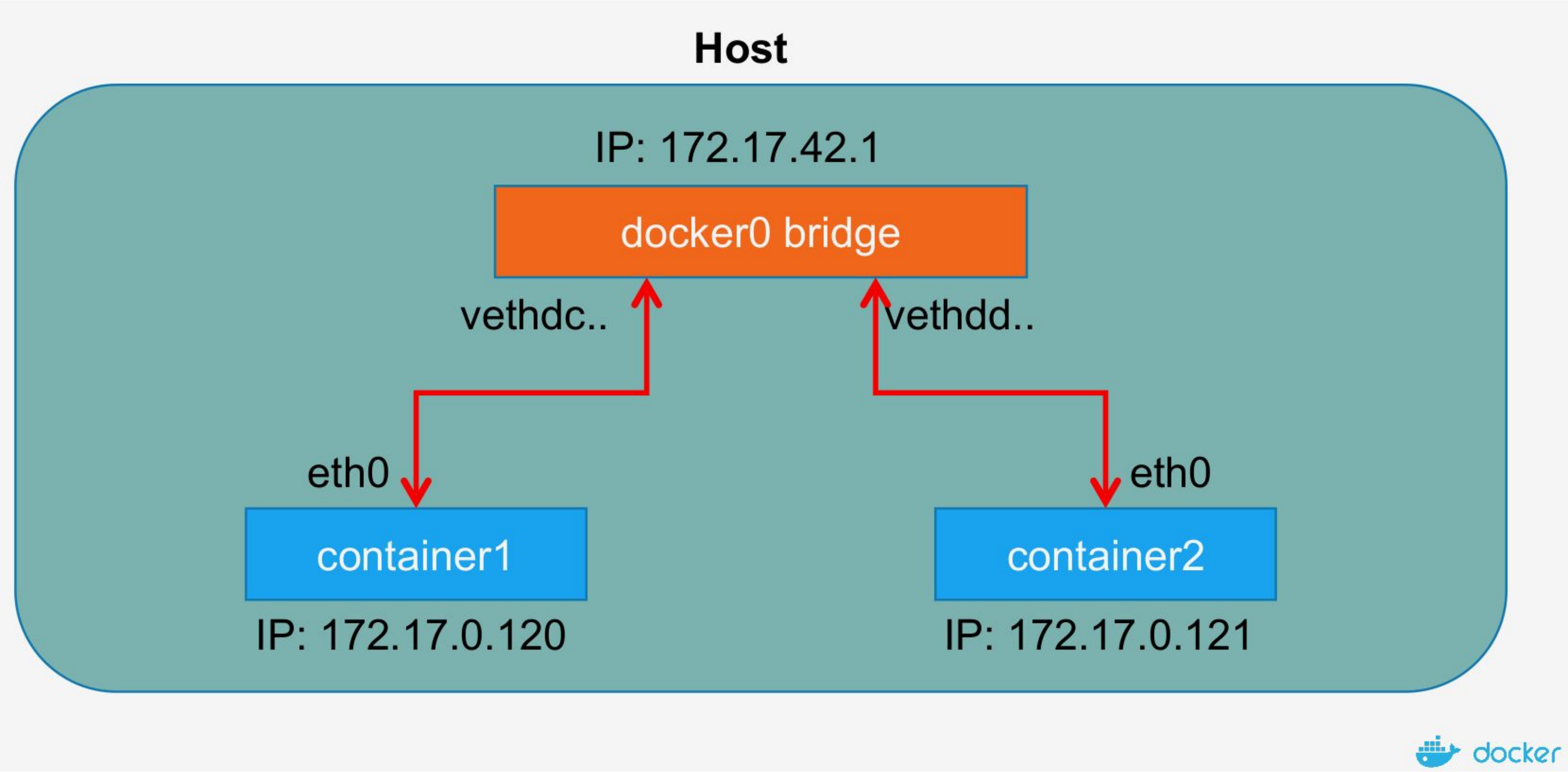
- **3 network mods available**
  - host: use directly the host network
  - bridge: use a virtual Ethernet bridge interface



```
docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
 link/ether 02:42:aa:29:08:70 brd ff:ff:ff:ff:ff:ff
 inet 172.17.42.1/16 scope global docker0
    valid_lft forever preferred_lft forever
 inet6 fe80::42:aaff:fe29:870/64 scope link
    valid_lft forever preferred_lft forever
```

  - none: can’t access to host network
- **1 Docker bridge network = 1 host network interface**
- **Can create custom bridge sub-networks**

Bridge network



Docker link / bridge demo

Remember

- link option adds an entry in /etc/hosts
- link option is deprecated, use `docker network`
- DNS is not activated in the default bridge network
- But activated by default in a custom bridge network

Docker Compose

Docker-Compose

- External Docker tool
- Describes in a single file all services that make up an “application”
  - ➔ Network
  - ➔ Volumes
  - ➔ Environment variables
  - ➔ etc.
- Simple YML syntax
- Bridge network contains DNS

Docker compose demo

Remember

- Convenient way to structure image linking and launching
- First approach of scaling / failover
- Single host only

Docker Swarm

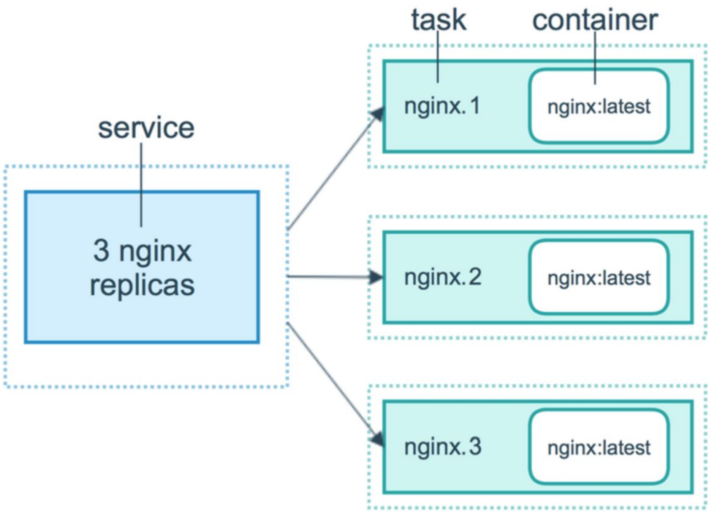
Orchestrators

- Manage a cluster of servers
- Distribute containers according to constraints
- Ensures that active services match an expected state

- **Main actors**
  - Mesos (*Apache*) / Marathon (*Mesosphere*)
  - Kubernetes (*Cloud Native Computing Foundation*)
  - Swarm Mode (*Docker*)

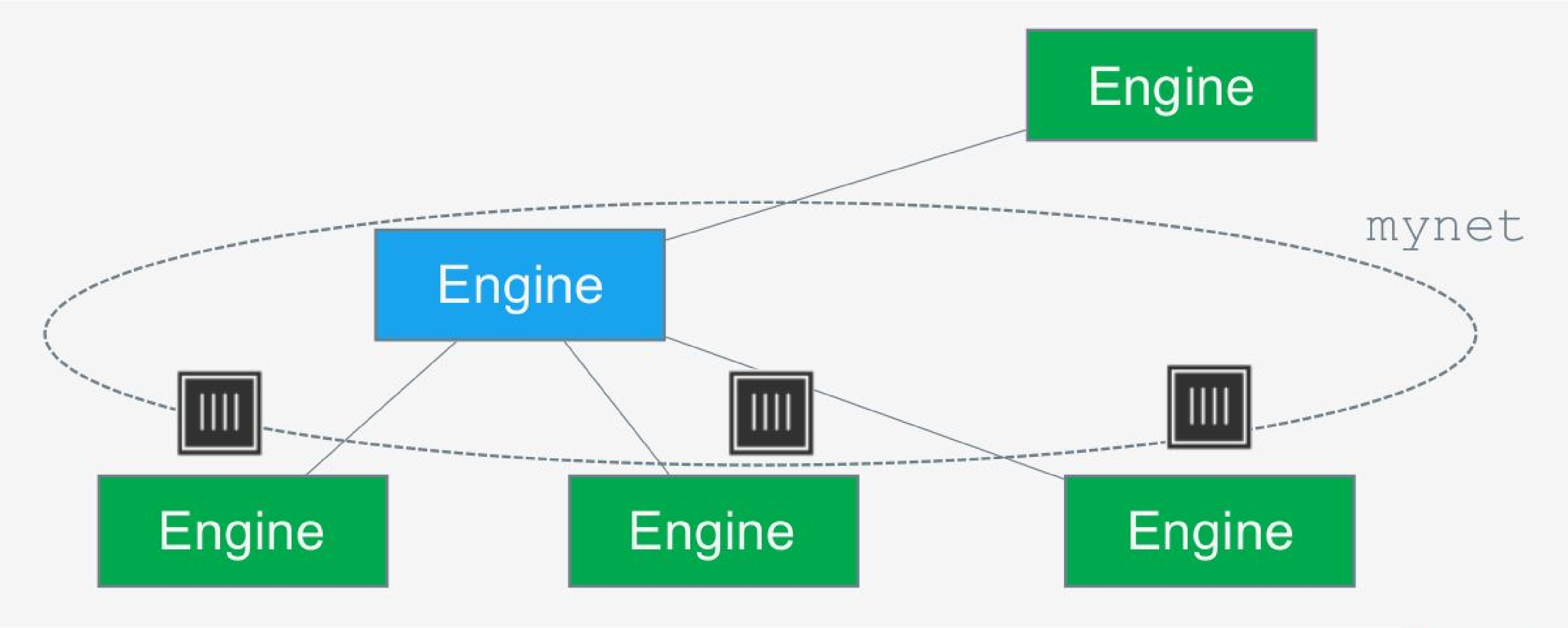
Docker swarm to the rescue !

- The concept of “service”

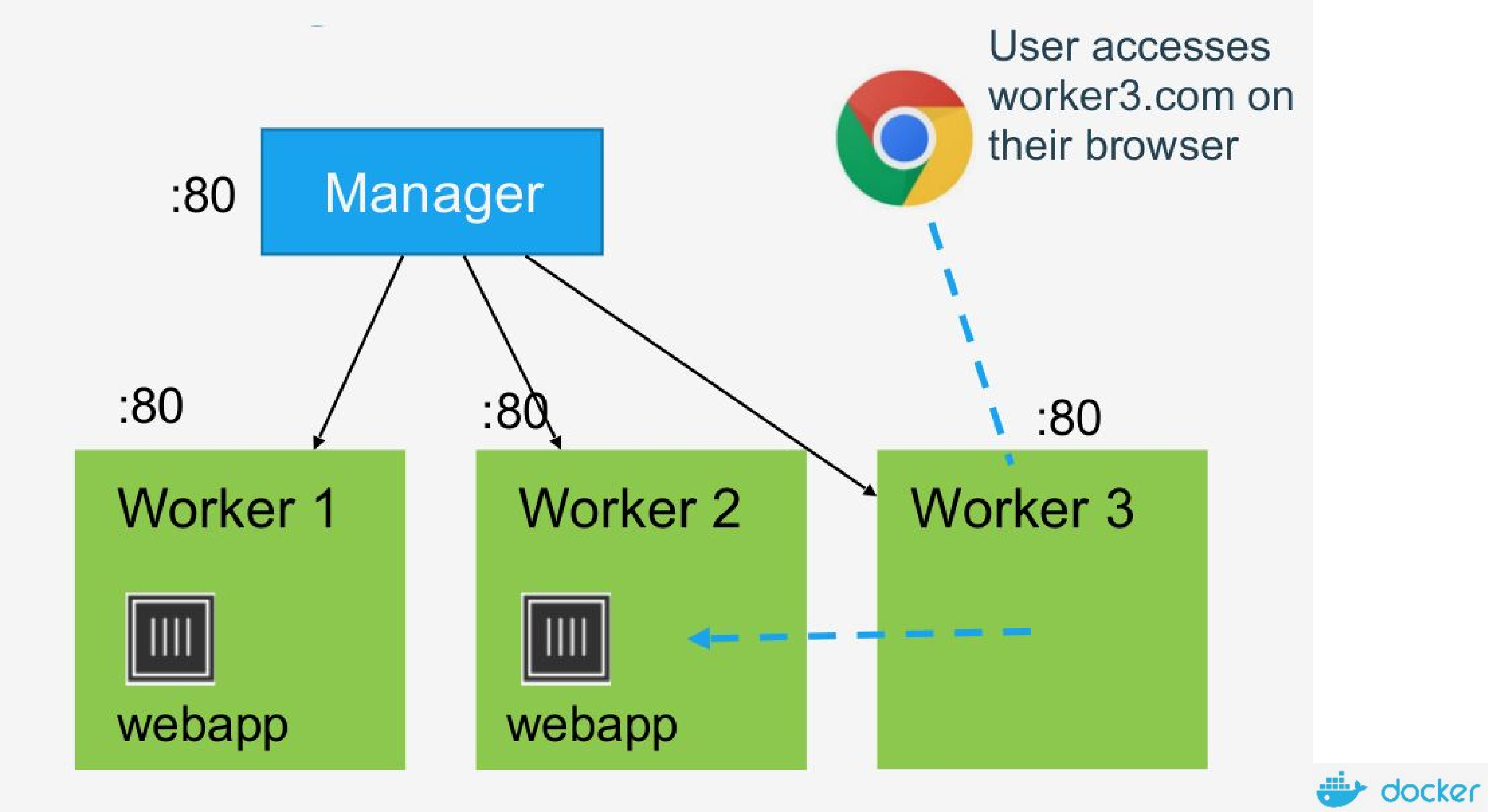


- Embedded discovery
- Load balancing
- Backups and disaster recovery
- Rolling updates
- ...

The overlay network



Routing Mesh



Docker swarm demo

Remember

- One of the easiest orchestrator to set up
- Included in Docker daemon since docker 1.12
- Routing Mesh
- VIP / DNSRR

“Experimental”

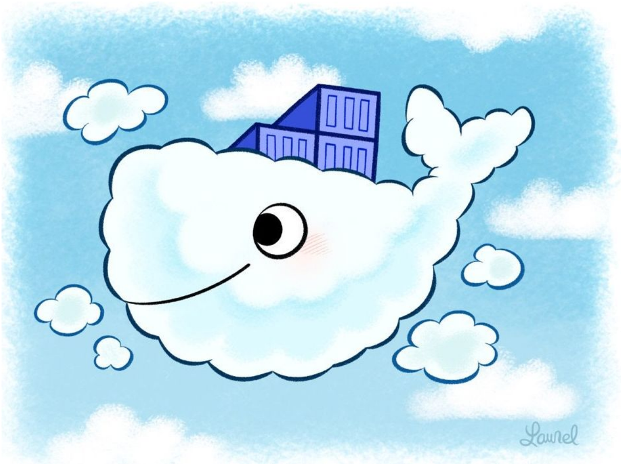
Bundle and stack deploy

- Deploy a list of services on multiple servers
- From Compose file to Bundle file (.DAB)
- Create a “stack” on a Swarm cluster from DAB file
- Automatic overlay network instead of bridge network
- Use of specific “stack” commands

Bundle demo

Happy whale

- Docker ecosystem is rich
- Easy to use
- Powerful
- Moves fast
- The hype



Sad whale

- Not completely production ready
- Moves fast
- Docker is not the only answer
    - RKT
    - Kubernetes, Mesos/Marathon, Rancher, ...

https://github.com/ImFlog/docker-discovery

IPPON

Digital . Technologies . Hosting

PARIS
BORDEAUX
NANTES
LYON
MARRAKECH
WASHINGTON DC
NEW-YORK
RICHMOND
MELBOURNE

contact@ippon.fr
www.ippon.fr - www.ippon-hosting.com - www.ippon-digital.fr
@ippontech
-
01 46 12 48 48