Kubernetes as a Service for everyone

Michal Jura
Linux Cloud/HA Developer
mjura@suse.com

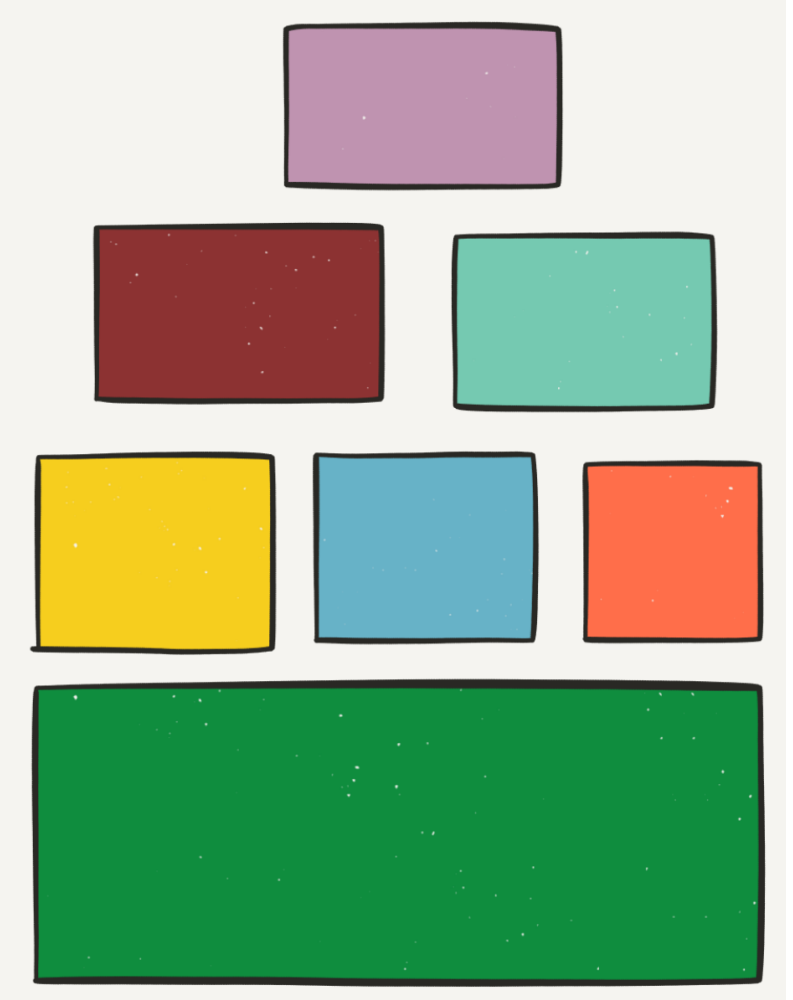
New challenges

- New age of internet and mobile applications.
- Application super portability.
- Higher complexity of cloud environments.
- Different cloud providers.
- Adoption of micro services architectures.
- Stay agile.

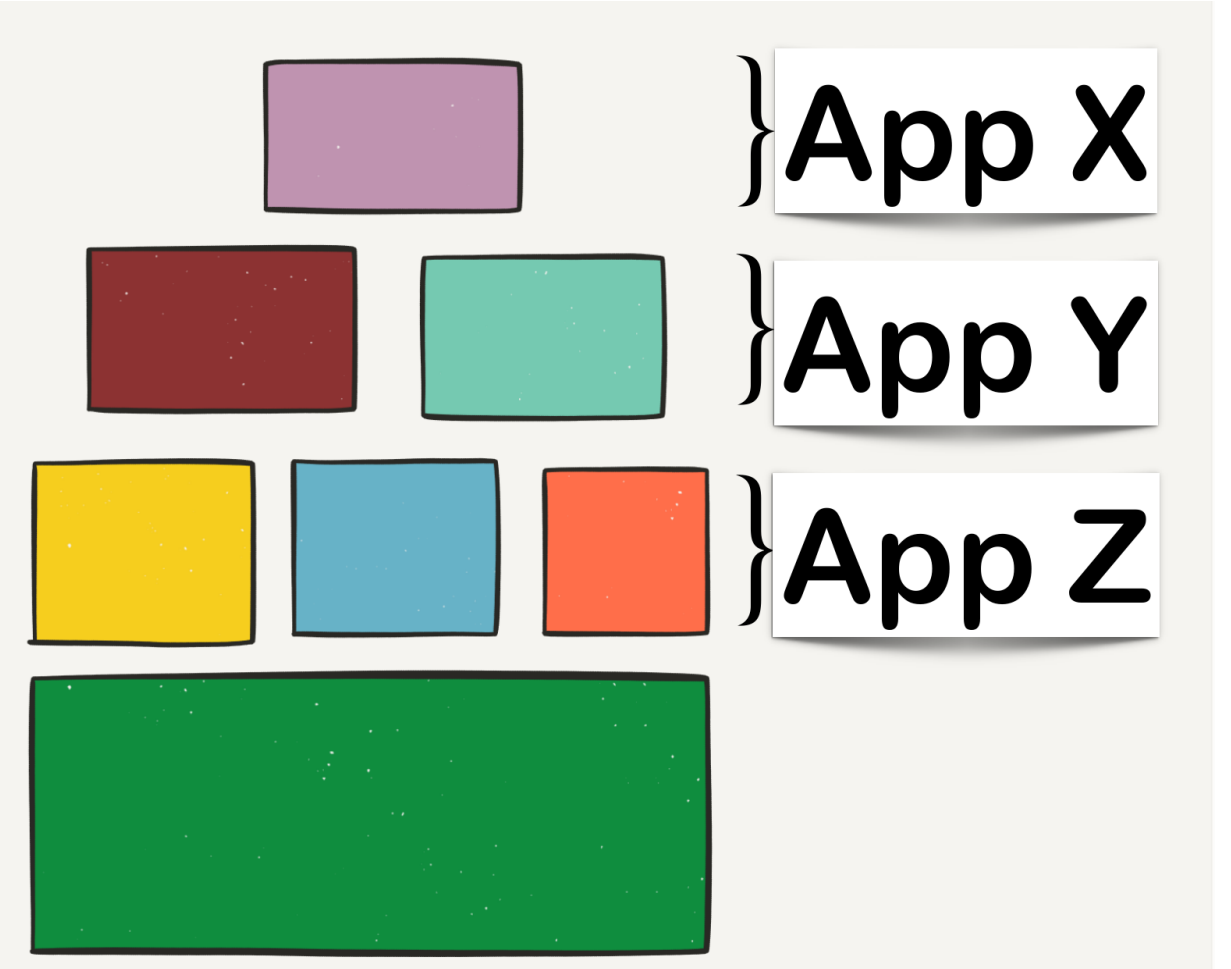
Manage applications, not machines

Let’s talk about application containers

Lightweight

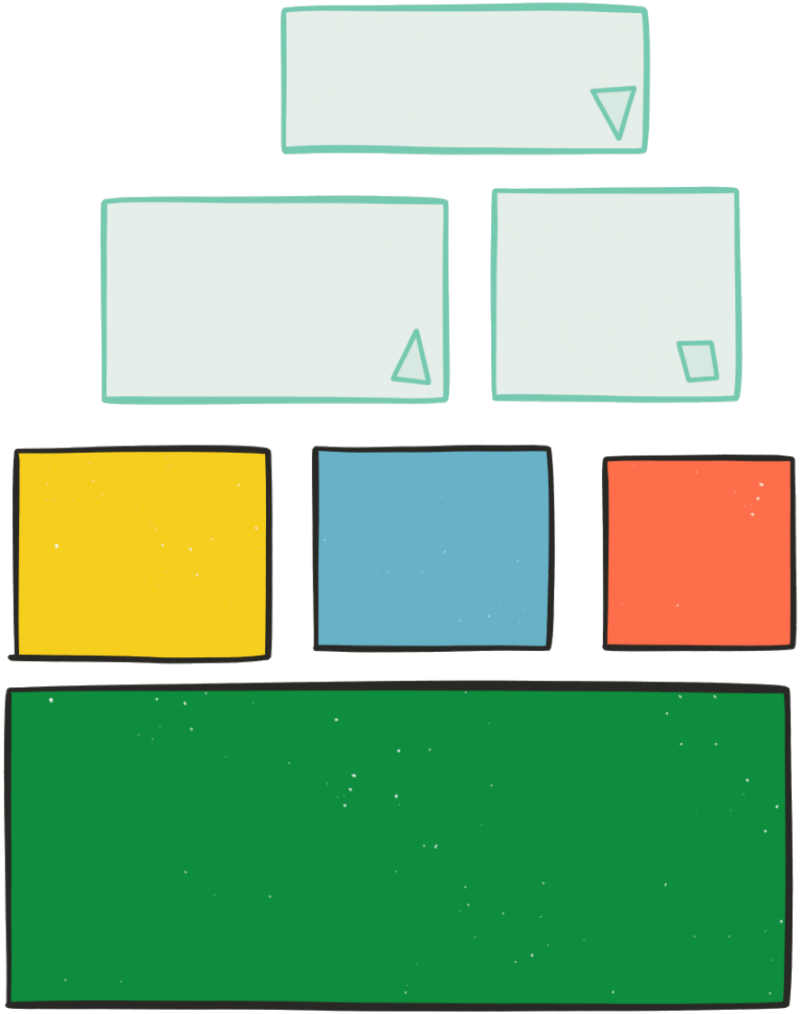


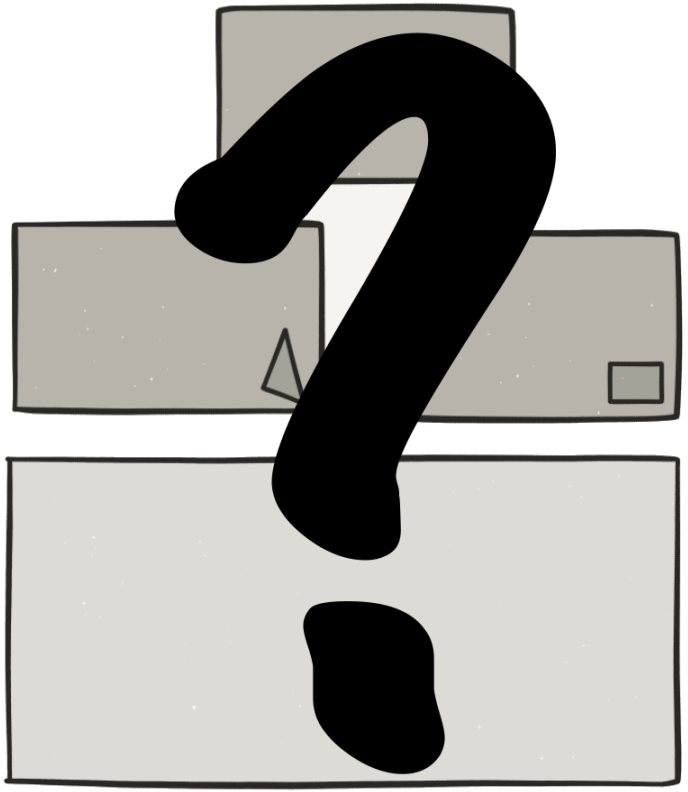
Composable



Getting serious ...

Multiple hosts deployments





Some challenges

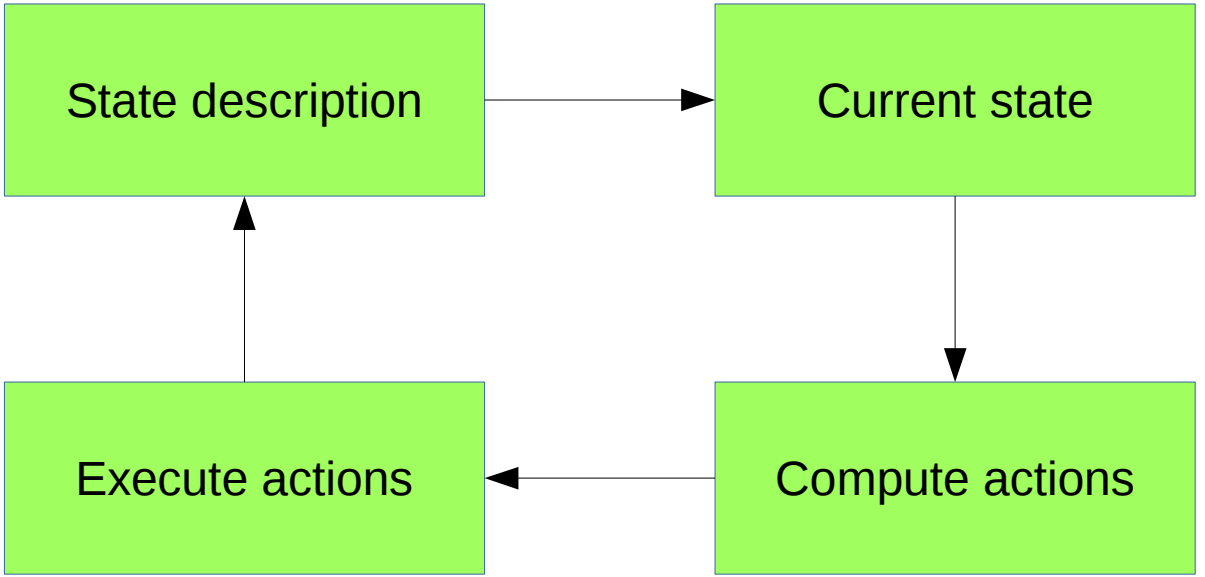
- Decide where to run each container
- Monitor nodes and containers
- Recover from failures
- Service discovery
- Expose services to external consumers
- Handle secrets (eg: credentials, certificates, keys,…)
- Handle data persistence

Containers orchestration

How can they help us?

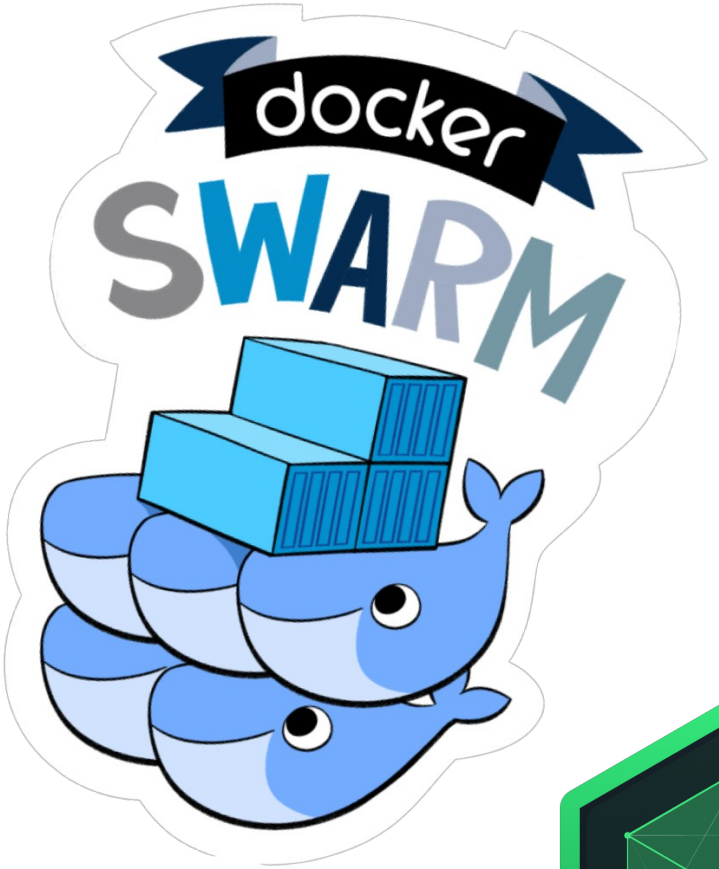
- We don't have to find the right placement for each container
- We don't have to recover from failures manually
- We just declare a desired state

Desired state reconciliation



Which one?

kubernetes



Nomad

MESOS

Kubernetes, of course!!!

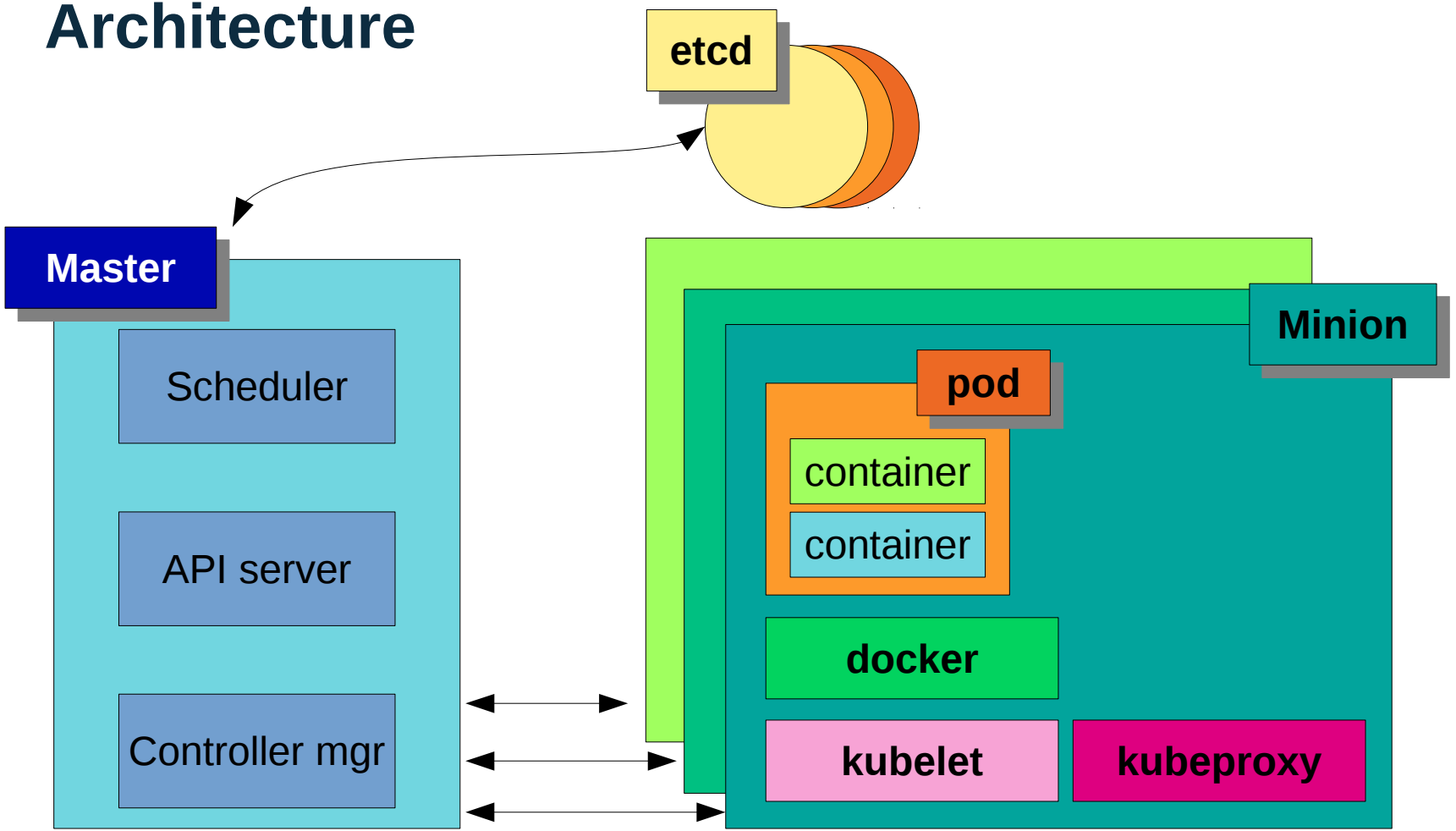
Kubernetes (aka k8s)

- Created by Google, now part of CNCF
- Solid design
- Big and active community
- An established “brand”
- Opinionated solution, has an answer to most questions

Kubernetes advantages

- Workload portability: doesn't enforce its directives to the application.
- Friendly with legacy applications: smooth migration path.
- Avoid vendor lock-in.
- Self healing.
- Auto-scaling.
- Has a solution for many problems:
  - Persistent storage.
  - Secrets management.
  - Blue-Green deployments.
- Flexible: plug-in architecture

Architecture



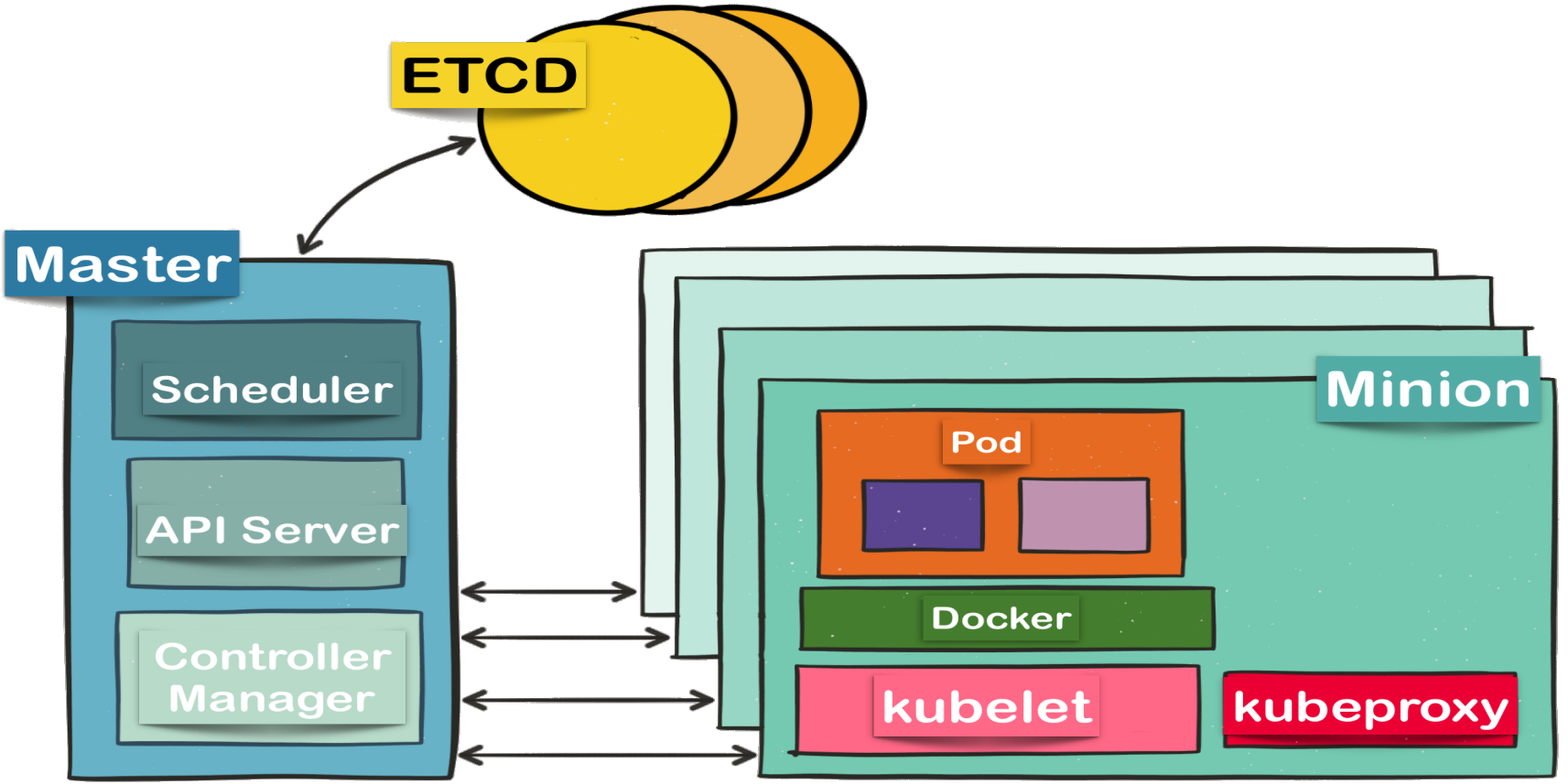
Kubernetes concepts

- Containers
- Pods
- Replication controllers
- Services
- Namespaces
- Labels
- Volumes

Kubernetes deployment

- Requires an *etcd* cluster.
- Requires one or more *master* nodes.
- Requires one or more *worker* nodes.
- Requires a SDN network joining all the *worker* nodes.
- Requires a load balancer to expose internal applications.
- Lots of patience to link all these components together.

Architecture



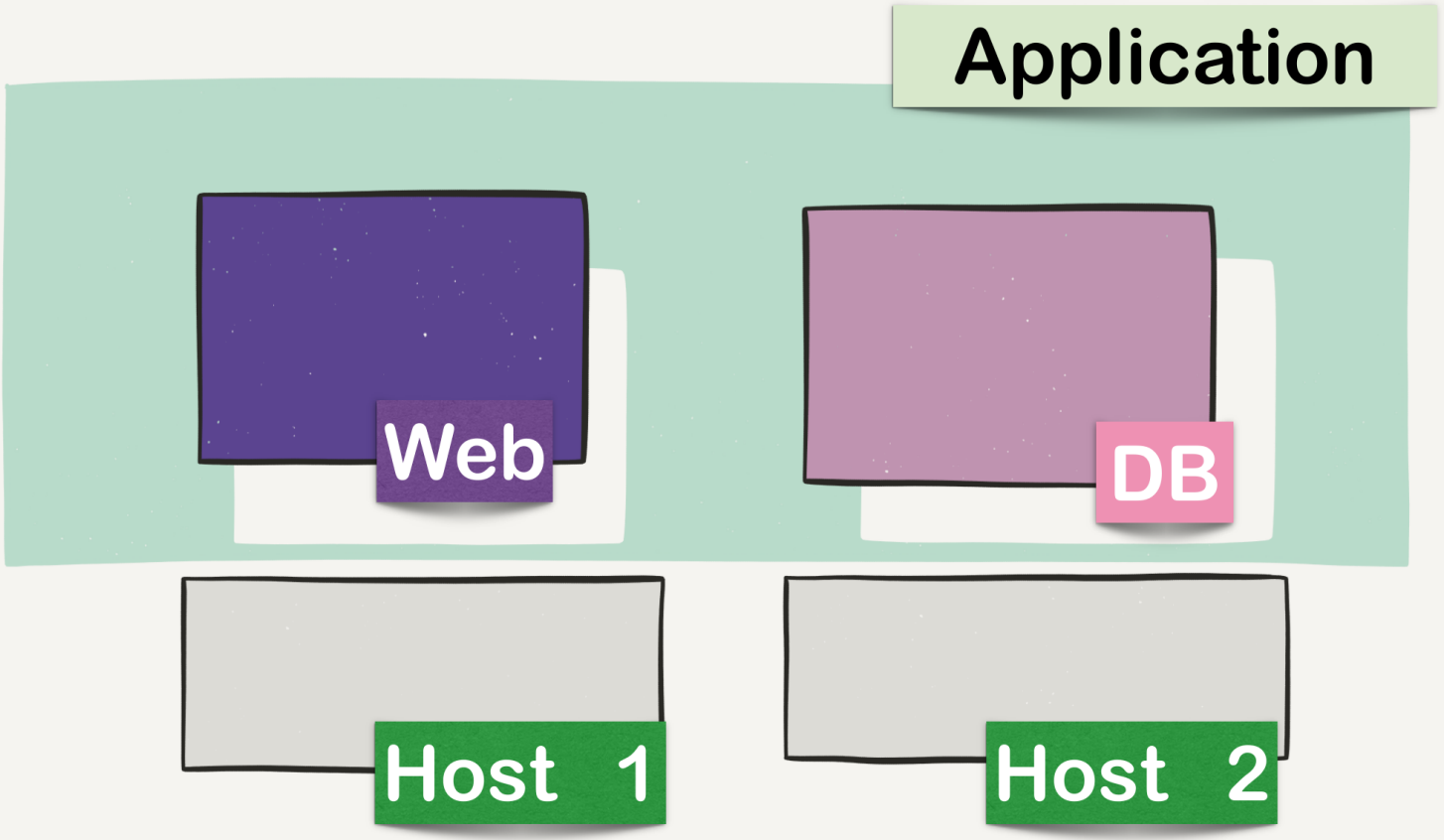
Kubernetes status

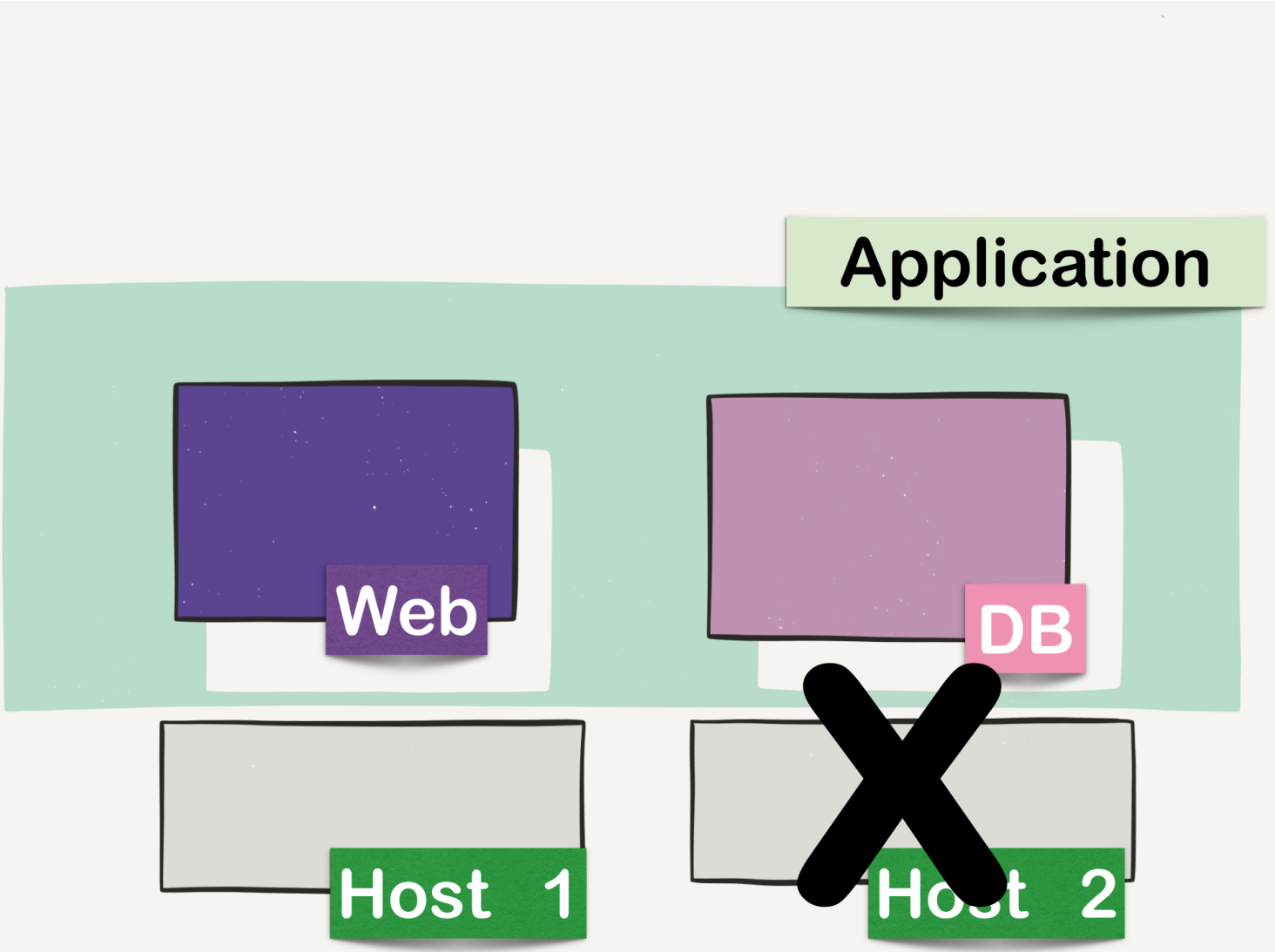
- It’s a pleasure to use as developer deploying your application.
- It’s a pleasure to administer as an operator.
- It’s a pain to deploy.

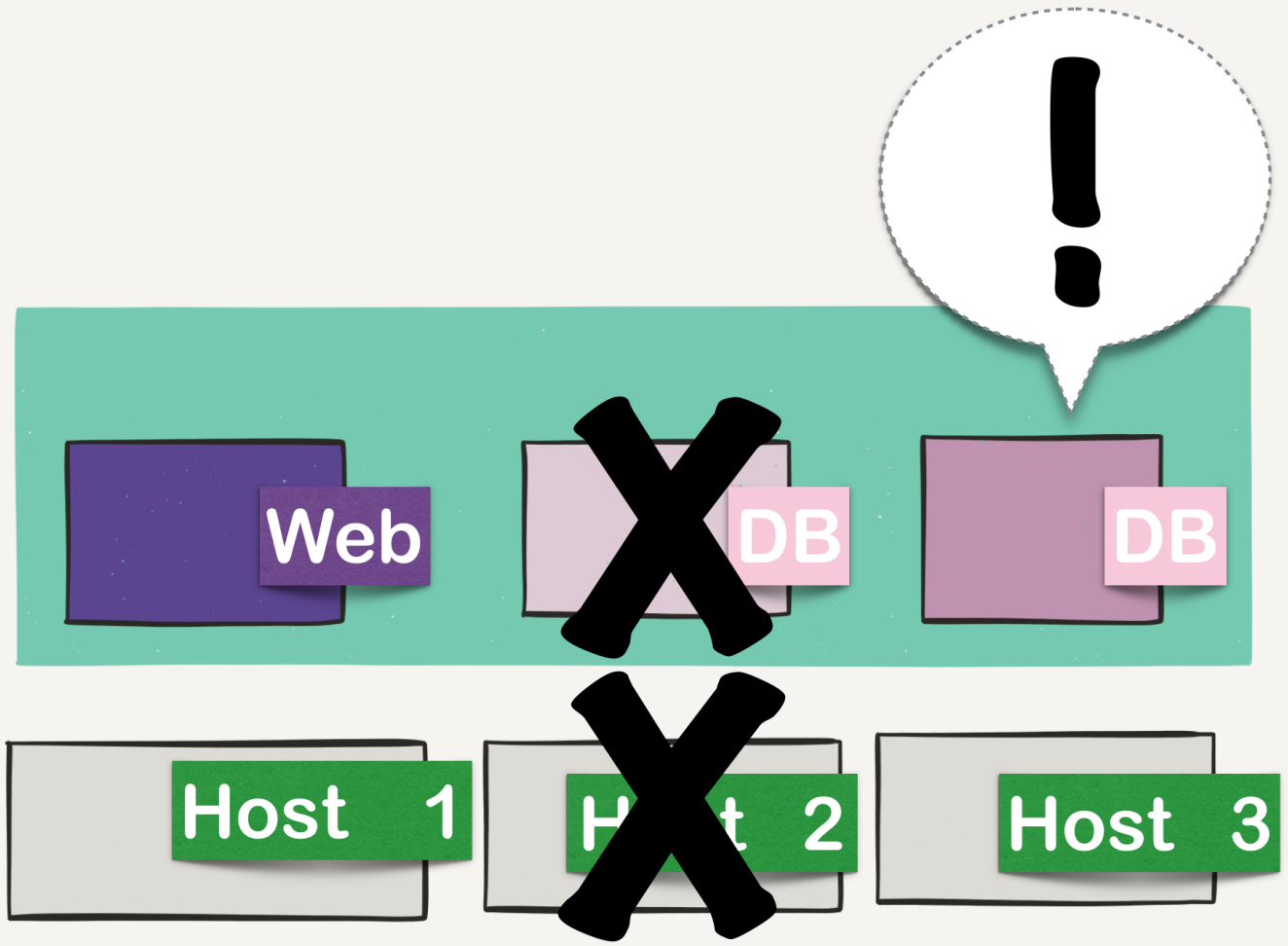
Things are changing

- **kubeadm**: upstream tool for kubernetes deployment.
- Kubernetes deployed with containers, by kubernetes itself.
- Introduced with the 1.4 release, still alpha.
- External orchestrators
  - OpenStack Magnum
  - Salt
  - Terraform

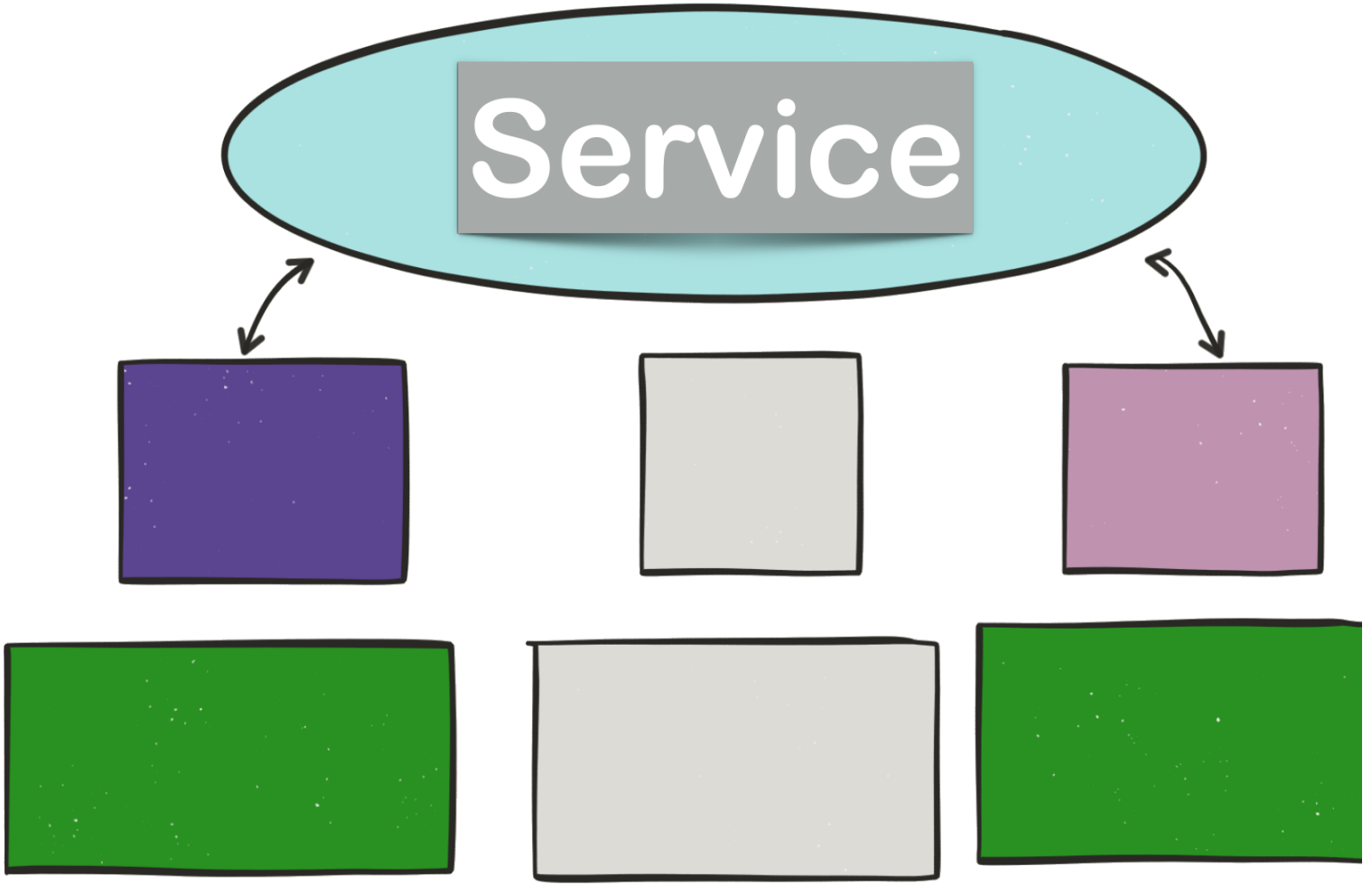
Replication Controller


Application
Web
DB
Host 1
Host 2

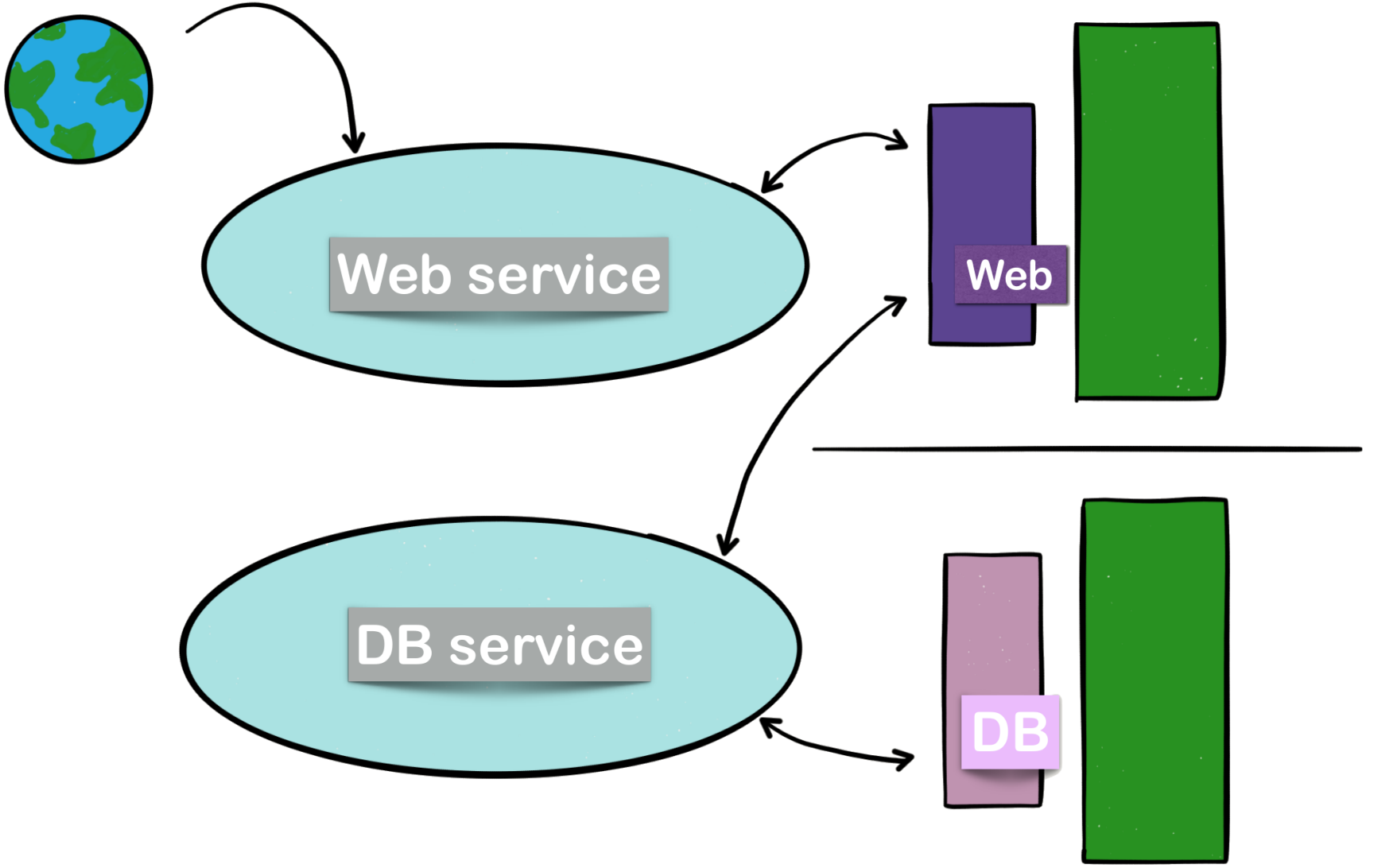

Application
Web
DB
Host 1
Host 2


!
Web
DB
DB
Host 1
Host 2
Host 3

Kubernetes Services


Service

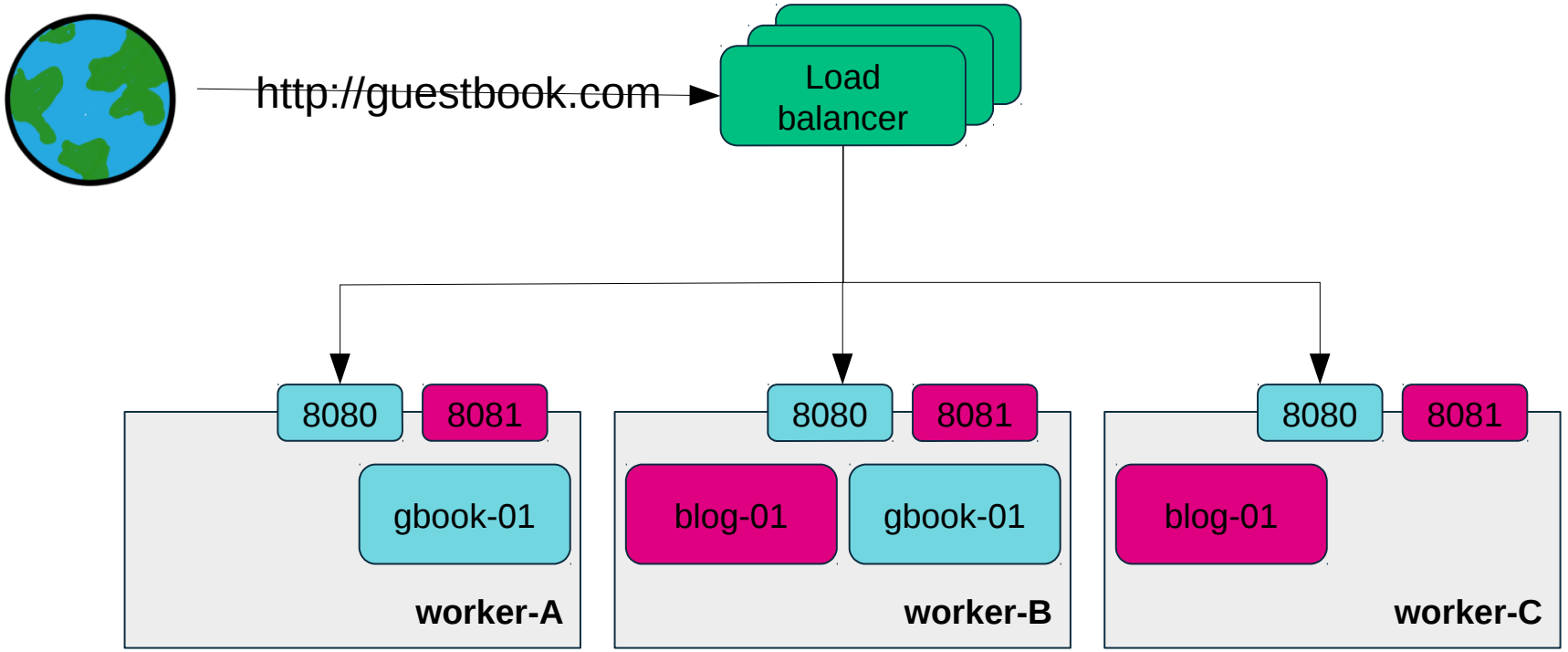
Basically


Web service
Web
DB service
DB

Implementation details

- Internal service:
  - Virtual IP: defined at creation time, doesn't change
  - Internal load balancer: kube-proxy
  - An internal DNS can point to the Virtual IP
- Published service: like an internal one, but...
  - Each minion exposes the internal service on a port
  - Can use cloud providers to have a load balancer

Architecture



What about data?

Data persistence

- Not all applications can offload data somewhere else
- Data should persist when the POD is moved to another host

Kubernetes Volumes

- Built into Kubernetes
- They are like other resources → scheduler awareness
- Support different backends via dedicated drivers:
  - Ceph
  - Cinder
  - NFS
  - GlusterFS
  - …..

Demo example

Questions ?

Michal Jura
Linux Cloud/HA Developer
mjura@suse.com