
KubeCon
CloudNativeCon
Europe 2024

Why Kubernetes Is Inappropriate for Platforms
and How to Make It Better.


Stefan Schimanski, Upbound, Senior Principal Engineer

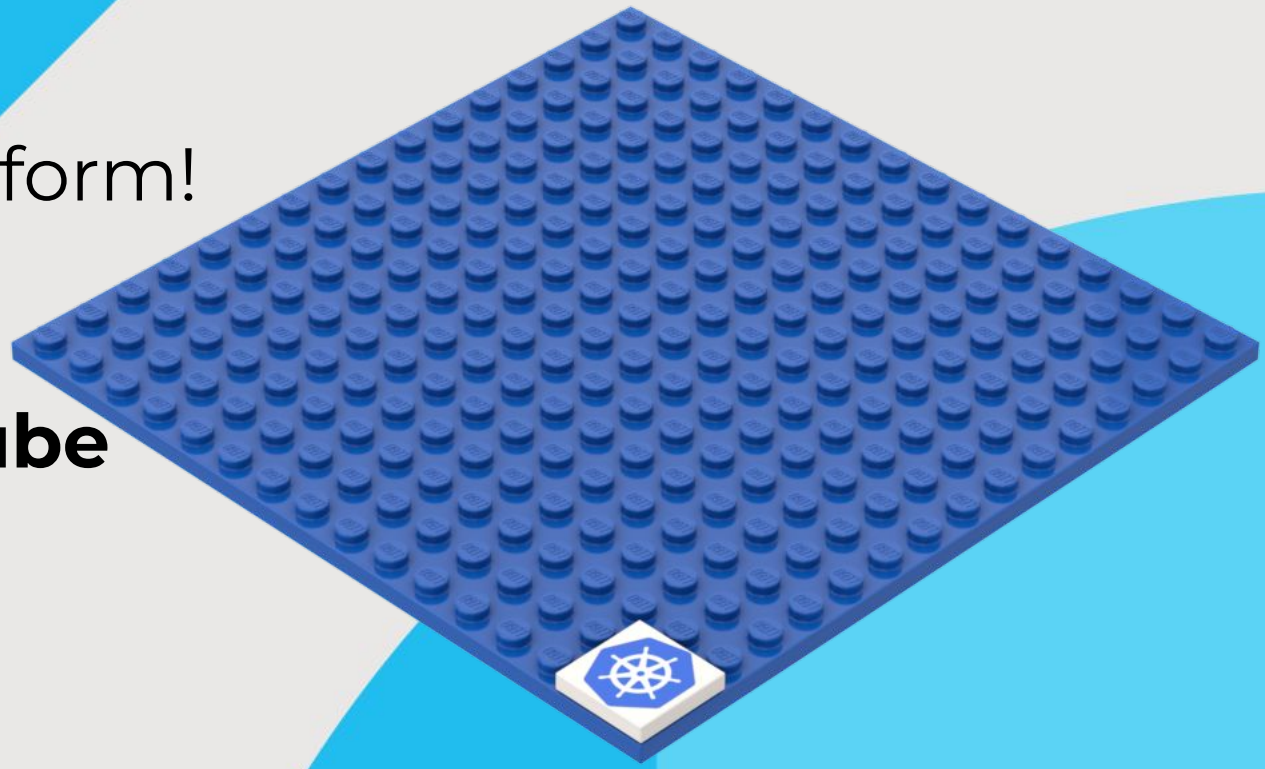

Mangirdas Judeikis, CAST AI, Staff Engineer


Sebastian Scheele, Kubermatic - CEO

Thought experiment for today

Let's build an internal AI/ML platform!

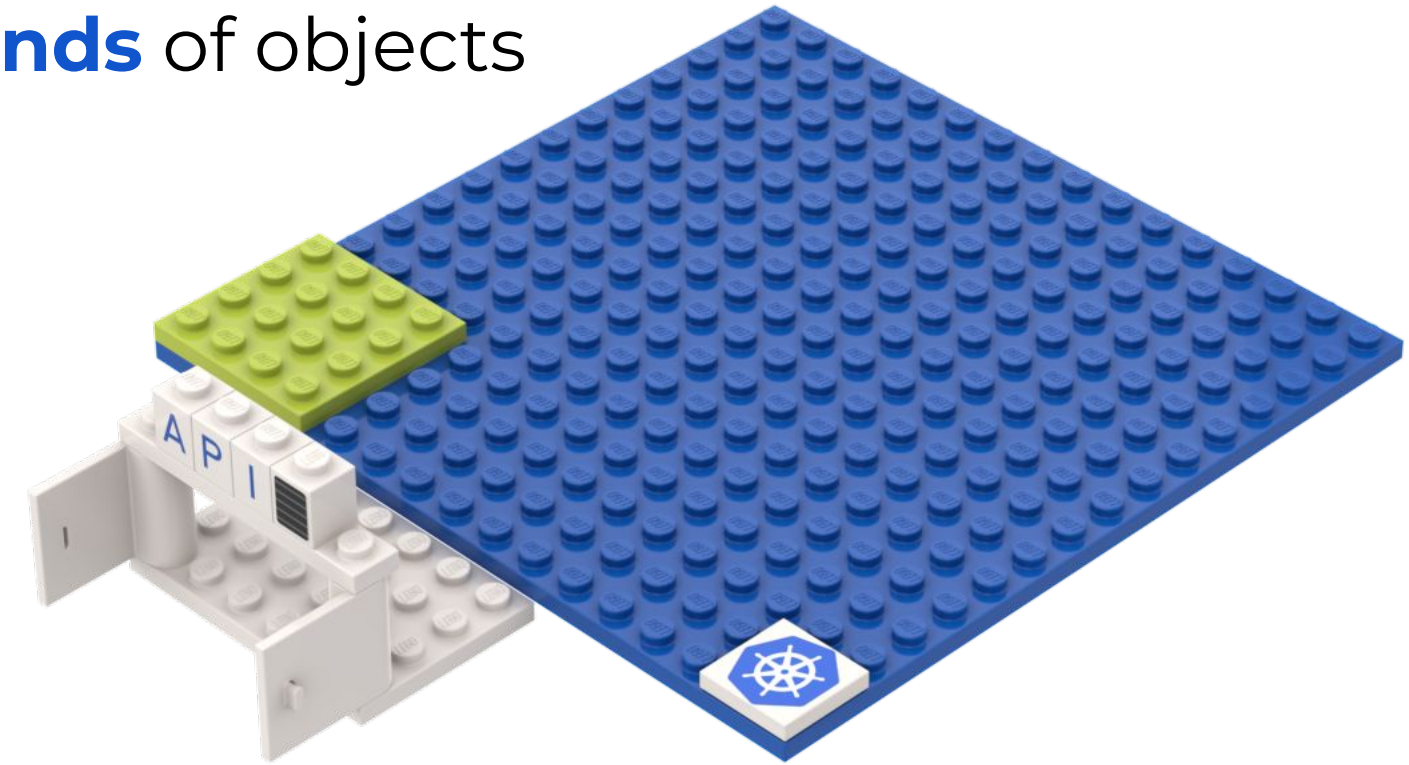
... and remind us **why we like kube**
... and **what we dislike**.



1. An Object Universe


```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat
  namespace: team1
spec:
  model: Llama2
  nProcPerNod: 1
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 6
```

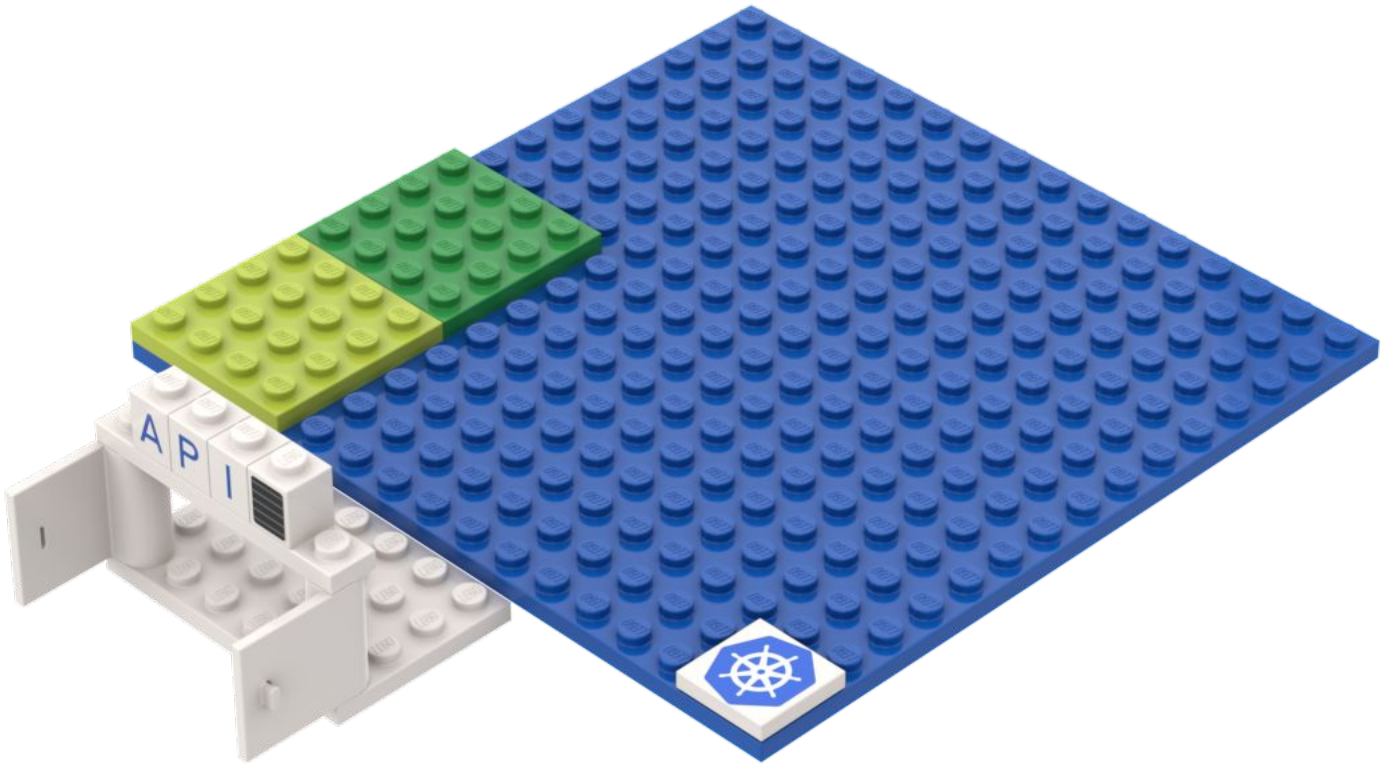
Different **kinds** of objects



1. An Object Universe


```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat
  namespace: team1
spec:
  model: Llama2
  nProcPerNod: 1
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 6
```

```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat
  namespace: team2
spec:
  model: Llama2
  nProcPerNod: 2
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 6
```



Living in **namespaces**.

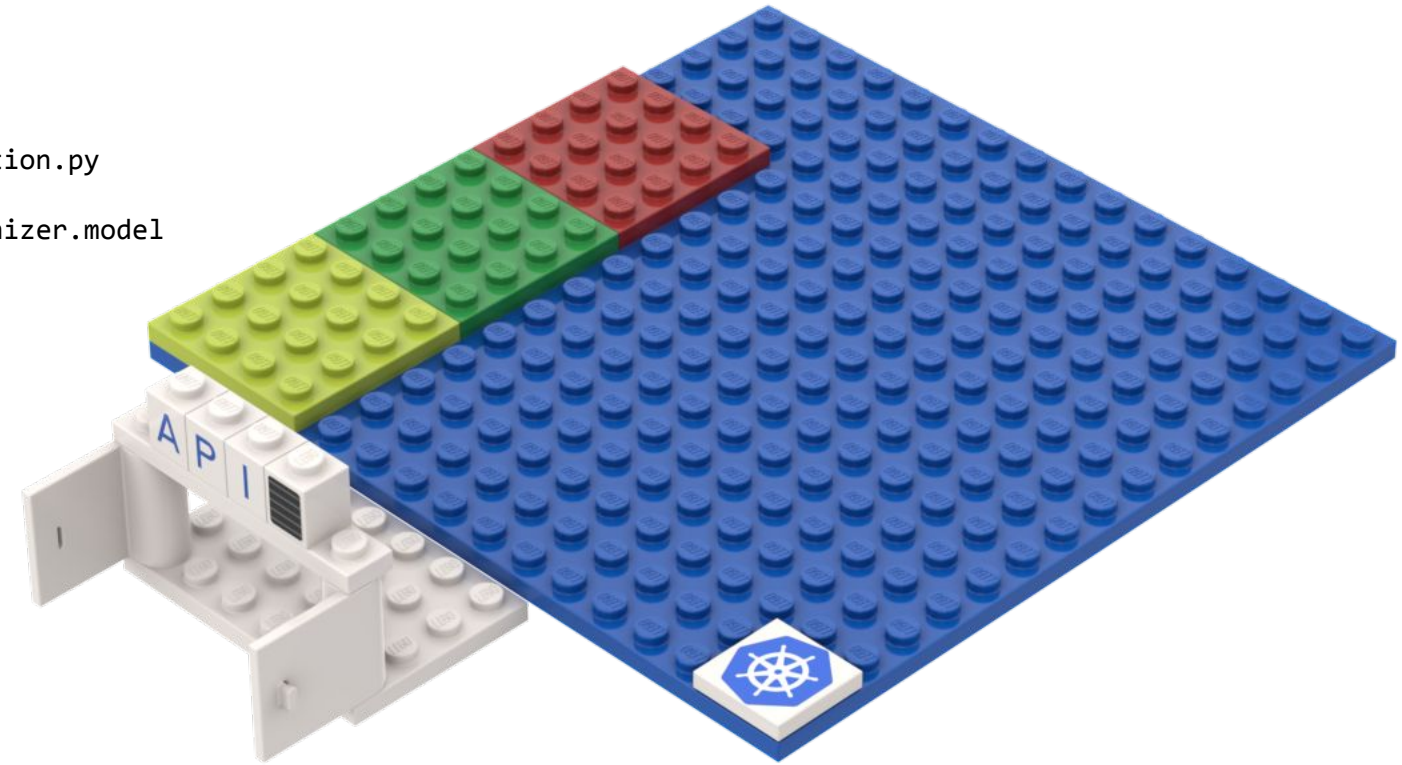
1. An Object Universe


```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat
  namespace: team1
spec:
  model: Llama2
  nProcPerNod: 1
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 6
```

```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat-internal
  namespace: team3
spec:
  model: Llama2
  nProcPerNod: 2
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 3
```

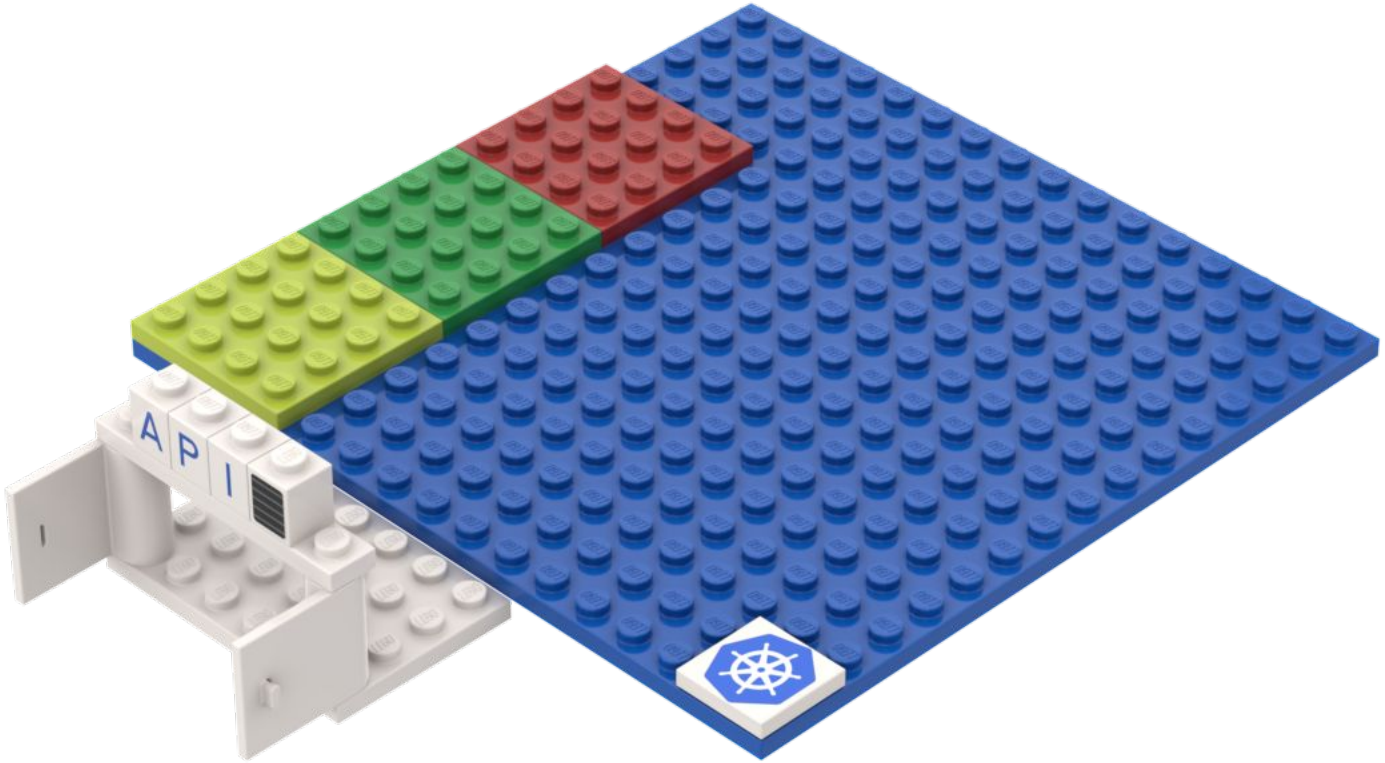
```yaml
kind: Model
apiVersion: training.faros.sh/v1alpha1
metadata:
  name: my-amazing-chat
  namespace: team2
spec:
  model: Llama2
  nProcPerNod: 2
  script: chat_completion.py
  ckptDir: /pvc/data
  tokenizerPath: tokenizer.model
  maxSeqLen: 512
  maxBatchSize: 6
```

with a **uniform API**.



Se

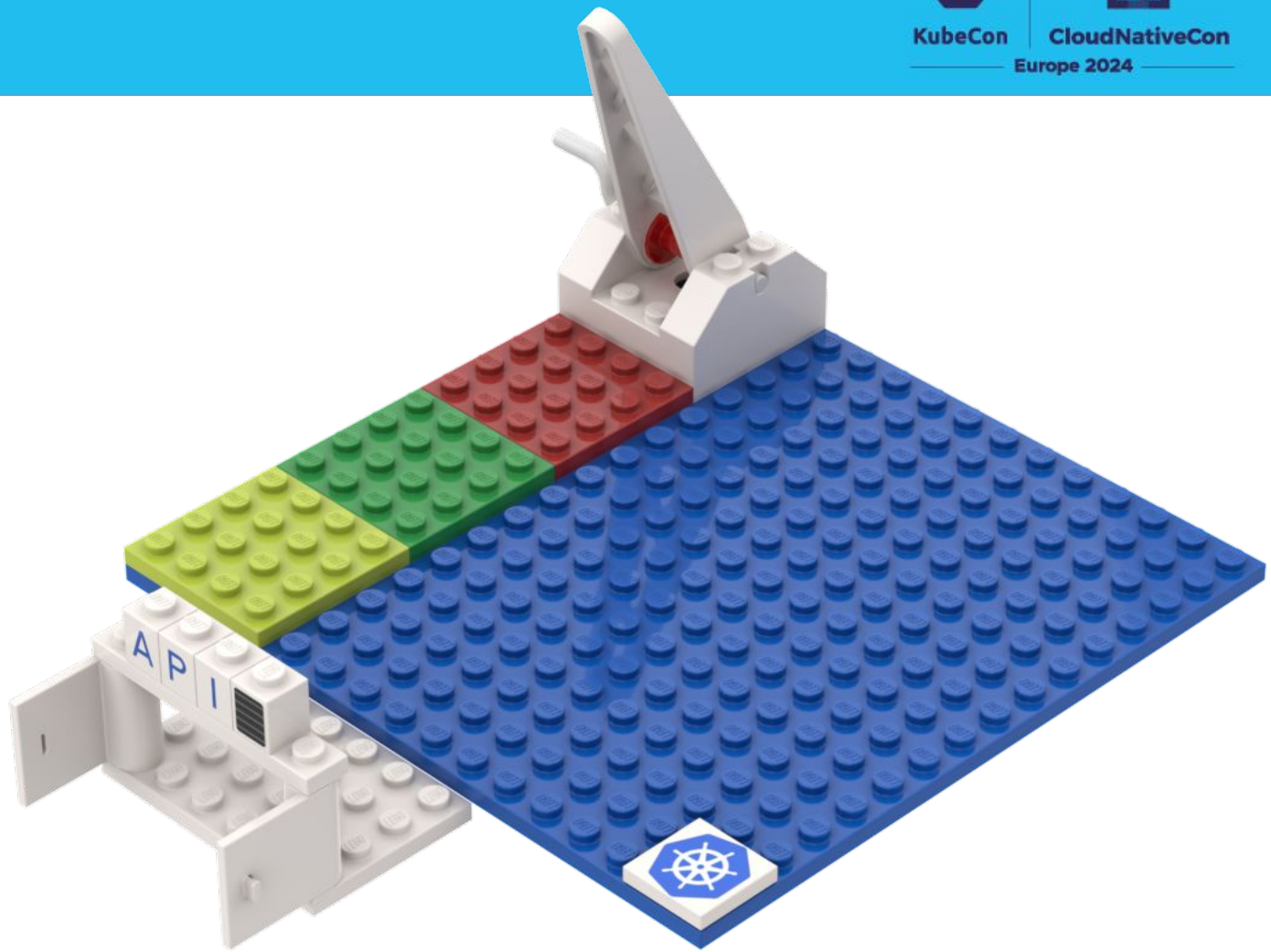
2. The Reconciler Pattern



2. The Reconciler Pattern

KubeCon CloudNativeCon Europe 2024

Controller runtime

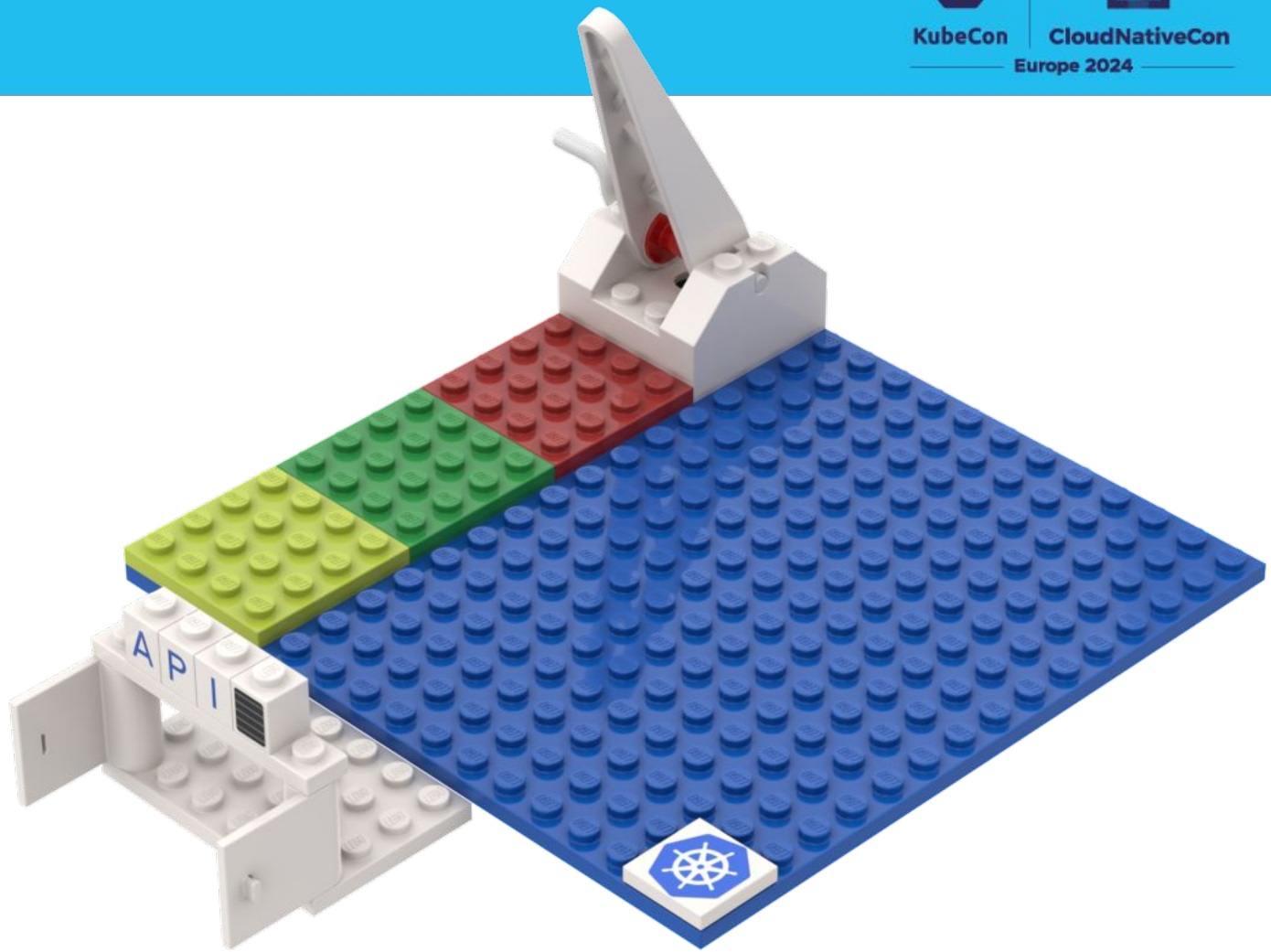


2. The Reconciler Pattern

KubeCon CloudNativeCon Europe 2024

Controller runtime

kubebuilder



2. The Reconciler Pattern

Controller runtime

kubebuilder

metacontroller

2. The Reconciler Pattern

Controller runtime

kubebuilder

metacontroller

client-go

2. The Reconciler Pattern

Controller runtime

kubebuilder

metacontroller

client-go

kube-rs, python, java, ...

Big ecosystem of tooling

3. Multi-Tenancy?

3. Multi-Tenancy?

KubeCon CloudNativeCon Europe 2024

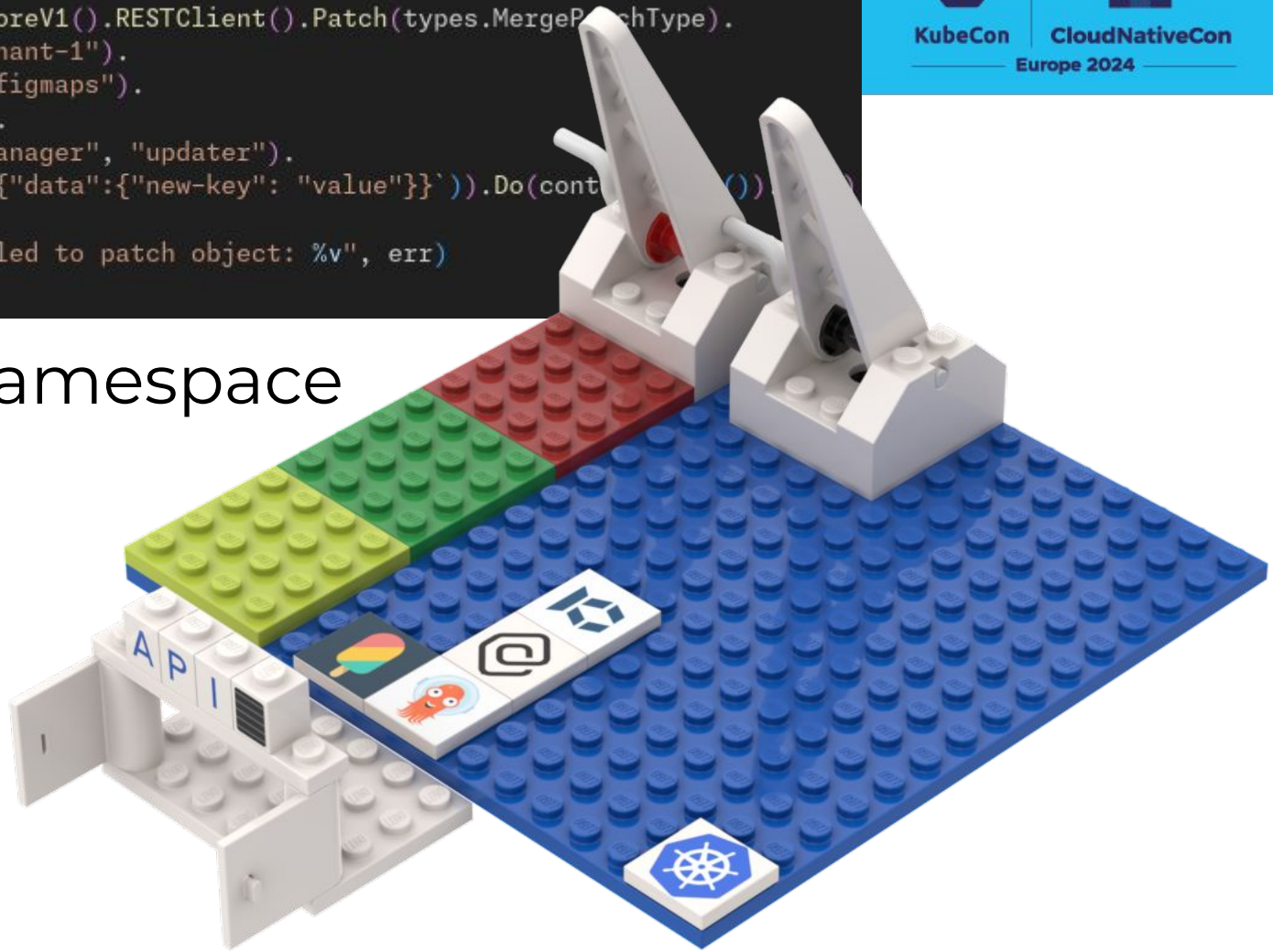
Namespaces

3. Multi-Tenancy?


```
_, err = client.CoreV1().RESTClient().Patch(types.MergePatchType).
    Namespace("tenant-1").
    Resource("configmaps").
    Name("my-cm").
    Param("fieldManager", "updater").
    Body([]byte(`{"data":{"new-key": "value"}}`)).Do(cont
if err != nil {
    t.Fatalf("Failed to patch object: %v", err)
}
```

Namespaces

Clients access all or one namespace



3. Multi-Tenancy?

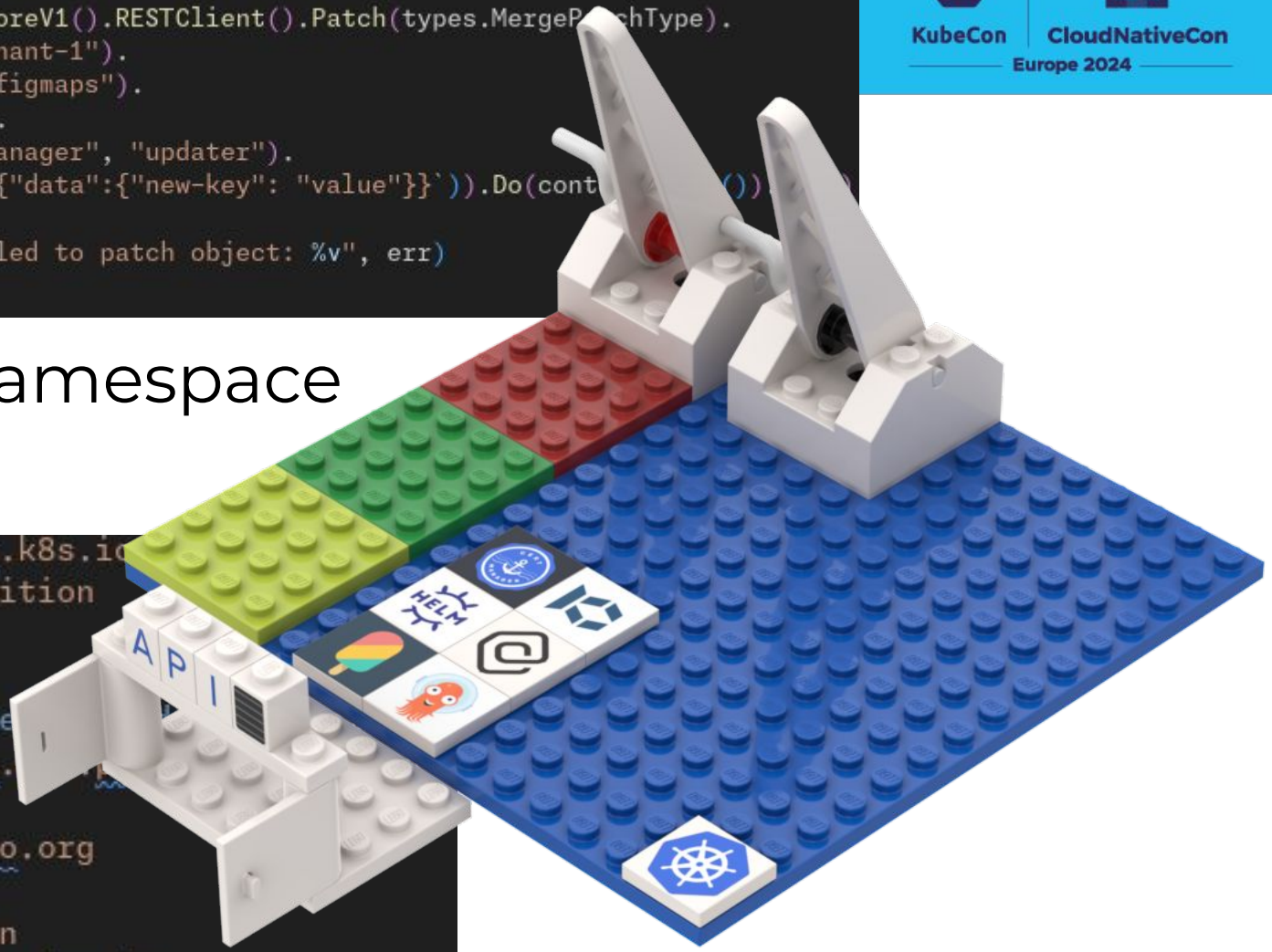
Namespaces

```
_, err = client.CoreV1().RESTClient().Patch(types.MergePatchType).
    Namespace("tenant-1").
    Resource("configmaps").
    Name("my-cm").
    Param("fieldManager", "updater").
    Body([]byte(`{"data":{"new-key": "value"}}`)).Do(cont
if err != nil {
    t.Fatalf("Failed to patch object: %v", err)
}
```

Clients access all or one namespace

CRDs

```
apiVersion: apiextensions.k8s.io
kind: CustomResourceDefinition
metadata:
  labels:
    addonmanager.kubernete
  name: bgpconfigurations.
spec:
  group: crd.projectcalico.org
  names:
    kind: BGPConfiguration
    listKind: BGPConfigurationList
    plural: bgpconfigurations
    singular: bgpconfiguration
  scope: Cluster
  versions:
  - name: v1
```



3. Multi-Tenancy?

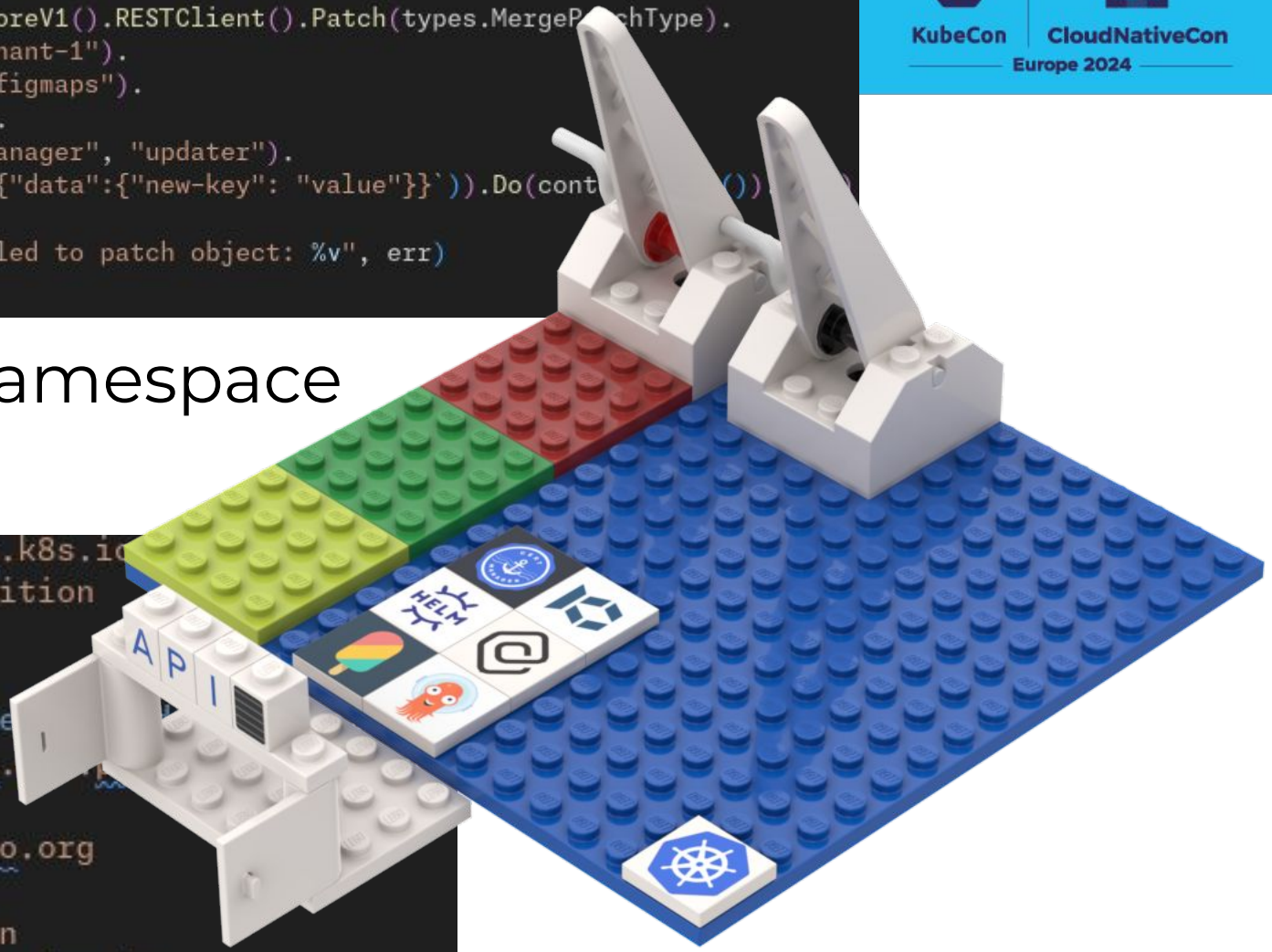
Namespaces

```
_, err = client.CoreV1().RESTClient().Patch(types.MergePatchType).
    Namespace("tenant-1").
    Resource("configmaps").
    Name("my-cm").
    Param("fieldManager", "updater").
    Body([]byte(`{"data":{"new-key": "value"}}`)).Do(cont
if err != nil {
    t.Fatalf("Failed to patch object: %v", err)
}
```

Clients access all or one namespace

CRDs

```
apiVersion: apiextensions.k8s.io
kind: CustomResourceDefinition
metadata:
  labels:
    addonmanager.kubernete
  name: bgpconfigurations.
spec:
  group: crd.projectcalico.org
  names:
    kind: BGPConfiguration
    listKind: BGPConfigurationList
    plural: bgpconfigurations
    singular: bgpconfiguration
  scope: Cluster
  versions:
  - name: v1
```



Short Status Check

This looks ok’ish.
Cracks already visible.

But we have learned to
live with our pains.

Let's continue building!

This picture is very 2017

Single region

Multi-tenancy via namespaces

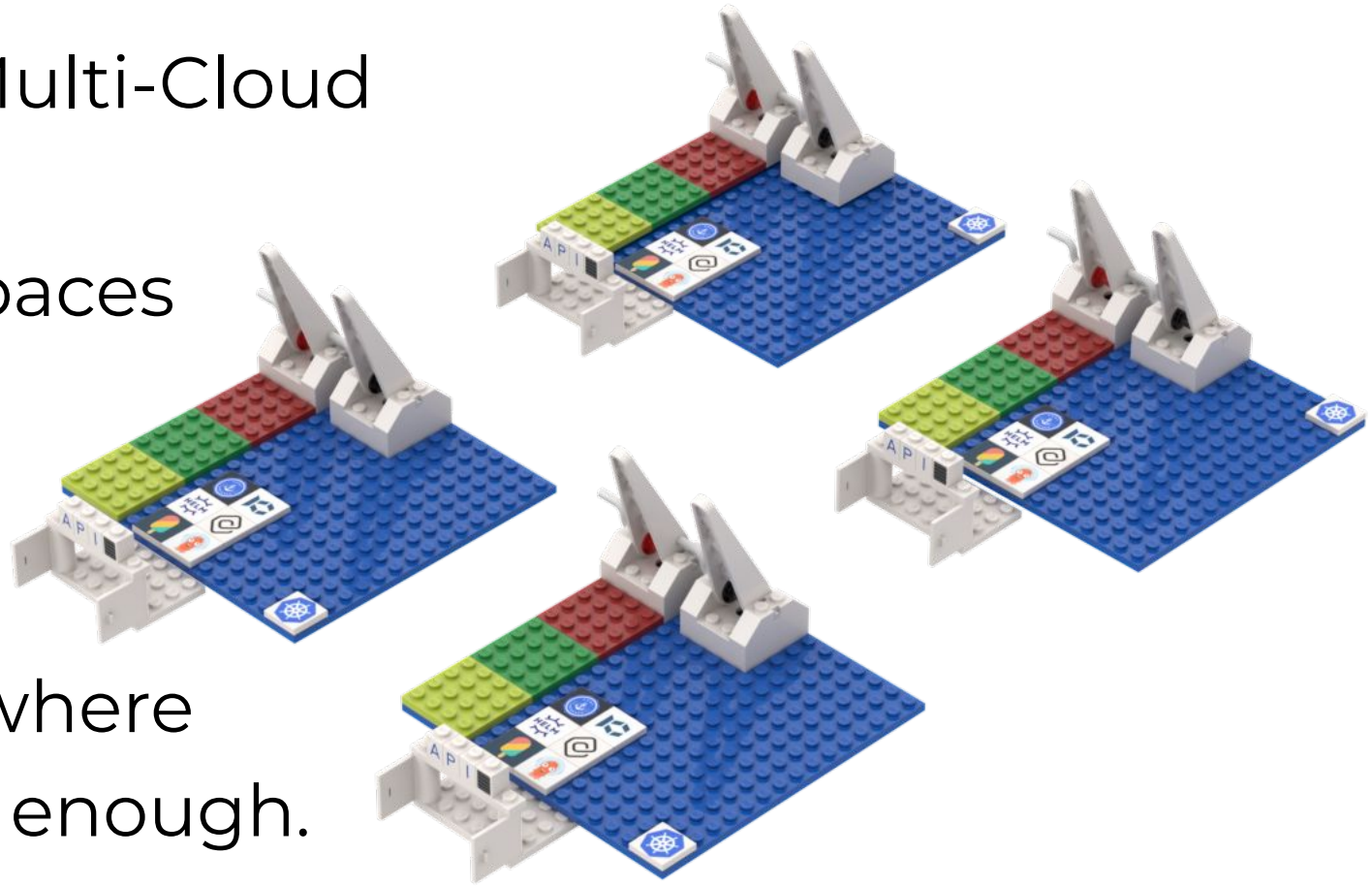
Single source of truth

Reality 2024

~~Single region~~ Multi-region Multi-Cloud

Multi-tenancy via namespaces

~~Single source of truth~~



Even more clusters where namespaces are not enough.

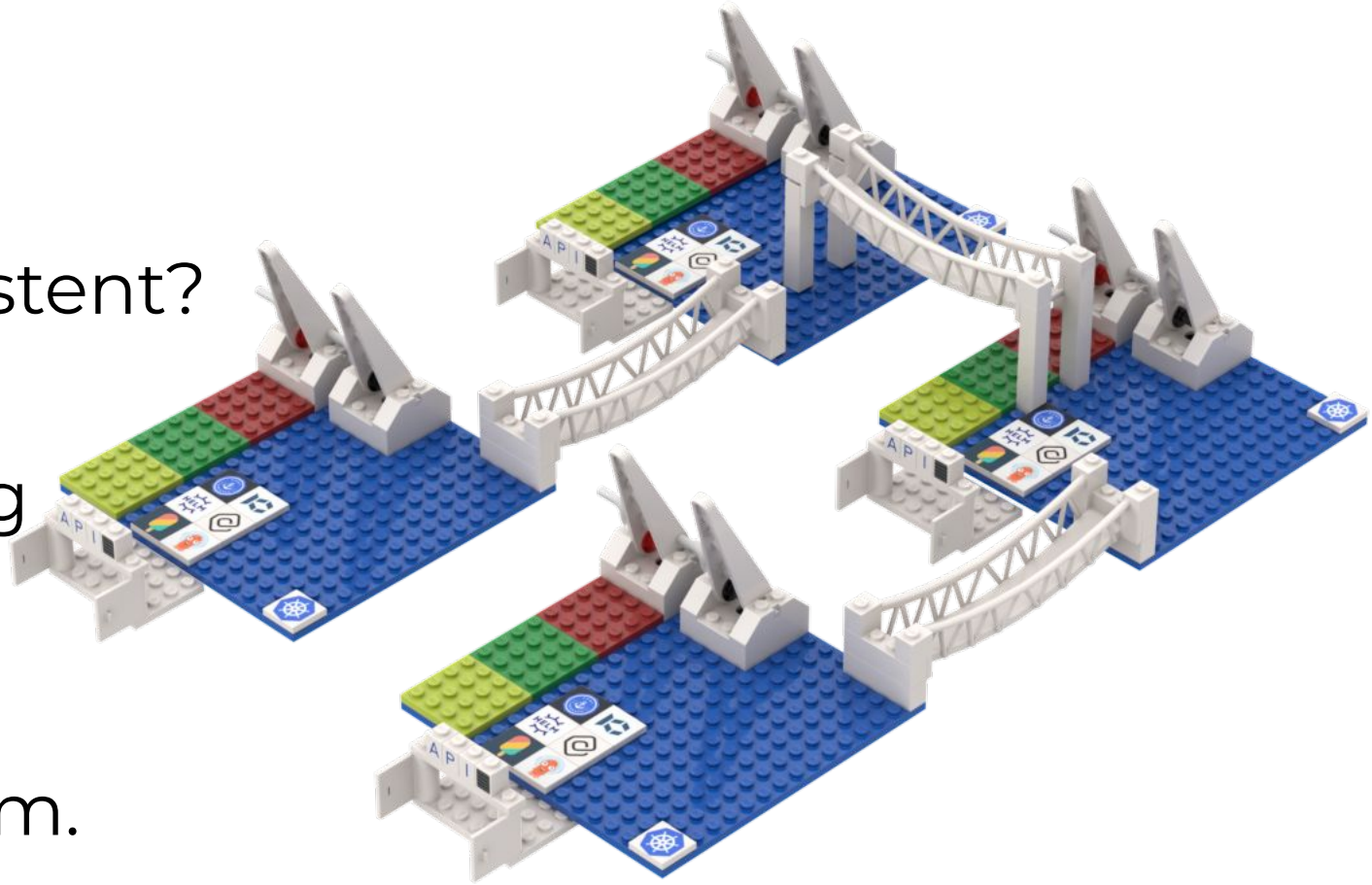
Sprawl of clusters.

… with that more challenges.

How to share data?

How to keep config consistent?

How to keep everything compliant?



We need bridges, lots of them.

We have multi-cluster tools – many to create them

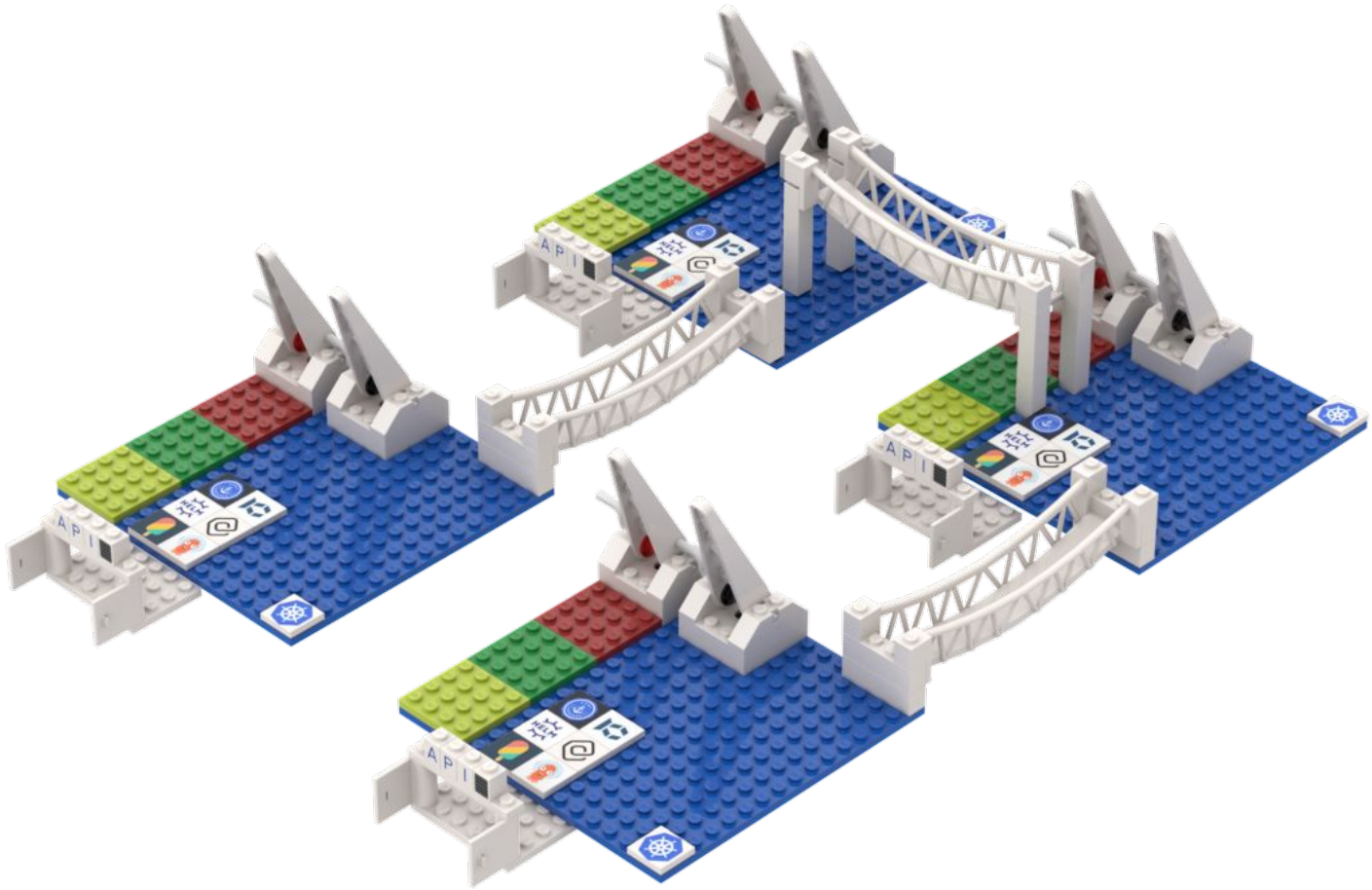
vclusters

Kamaji

KubeVirt

Karmada

Cluster API



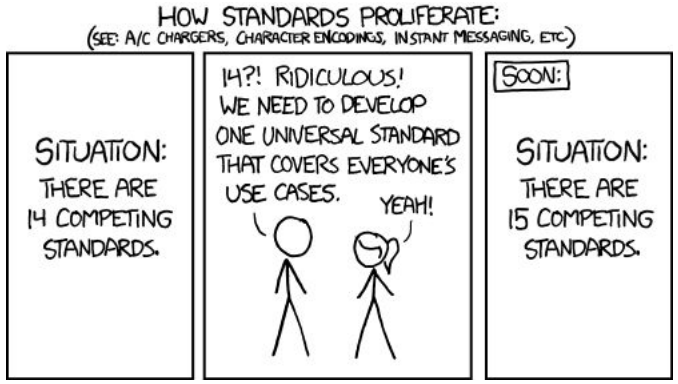
We have multi-cluster tools – many to tame them

ArgoCD

Crossplane

Cluster Federation

OpenClusterManager



But technology and tools are not the only problem.

In platforms, we have different personas.

And every single tool dictates their view on these.

Roles & responsibilities - permissions

Roles & responsibilities - permissions

Typical Roles & responsibilities:

- platform owner
- service provider
- user

Everyone with partial responsibility.

And finally 3rd-parties.

Everybody can build a helm chart for one cluster.

How to package something for this setup?

Reality Check

Complexity explodes

hard to support

hard to extend

hard to integrate into

not a good experience
for any of the roles.

Reality Check

Kube was built to run containers.

Kube was never built for platforms.

Reality Check

Kube was built to run containers.
Kube was never built for platforms.

Remember why we like Kube.

Reality Check

Kube was built to run containers.
Kube was never built for platforms.

Remember why we like Kube.

What would be a Kube built for platforms?

Towards a Kubernetes for Platforms

We know our final target. We did it before. But this time:

Ambitions are different.
Personas changed.
Kube was never the endgame.

A necessary shift in mindset

Traditional

lots of fun, great for creativity, but everything built out of these will look differently. Generally too low level.

A more Grown up Approach

Everything lives in one environment. We want off-the-shelf components. We want one environment where we can collaborate. Where solution from one party fit to setup of somebody else.

The platform challenge

Three Personas – Three Views

The platform owner

the ones with the keys
for everything,
the ones connecting and aligning
everything

The service provider

AI/ML service team,
DB team, PKI team,
CI/CD service, 3rd-parties
service owner

The users

Developer,
Data scientist,
application owner

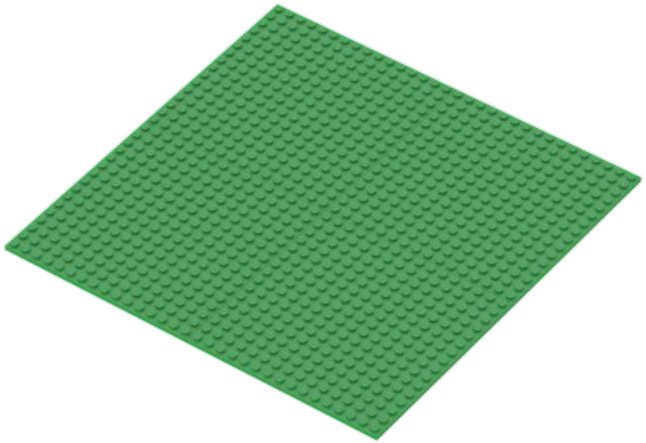
Se

The platform owner

the ones with the keys
for everything,
the ones connecting and aligning
everything

A Kubernetes for Platforms – the Platform Owner

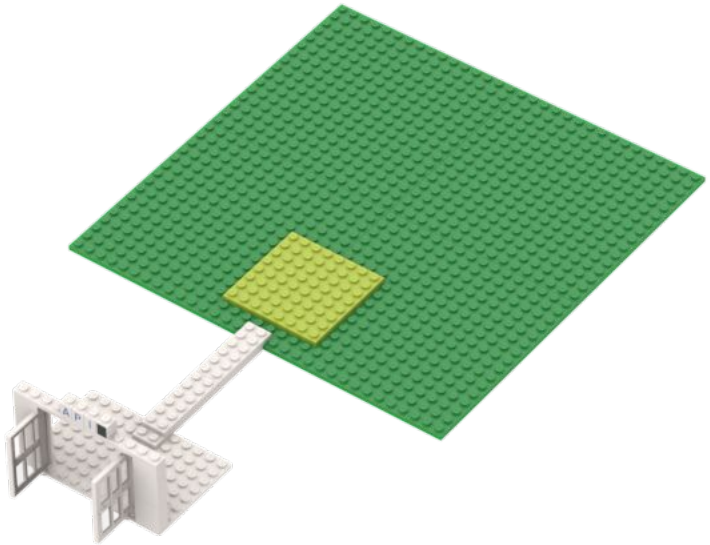
As a platform owner - **you are an enabler!**



A Kubernetes for Platforms – the Platform Owner

As a platform owner - you are an enabler!

Give user a **well defined**,
flexible platform
to ease their job!

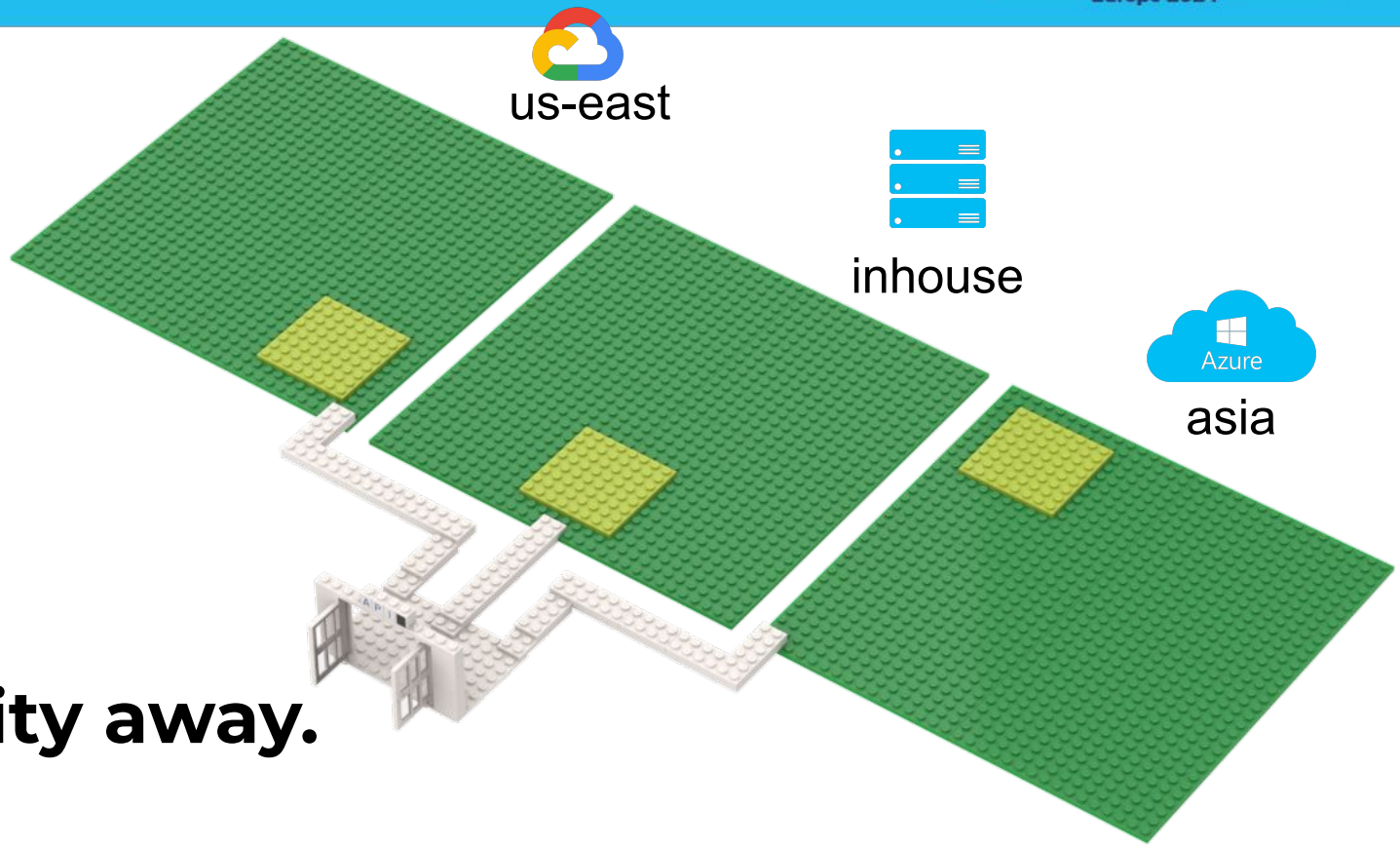


A Kubernetes for Platforms – the Platform Owner

As a platform owner

Give user a well defined,
flexible platform
to ease their job!

Abstract the complexity away.



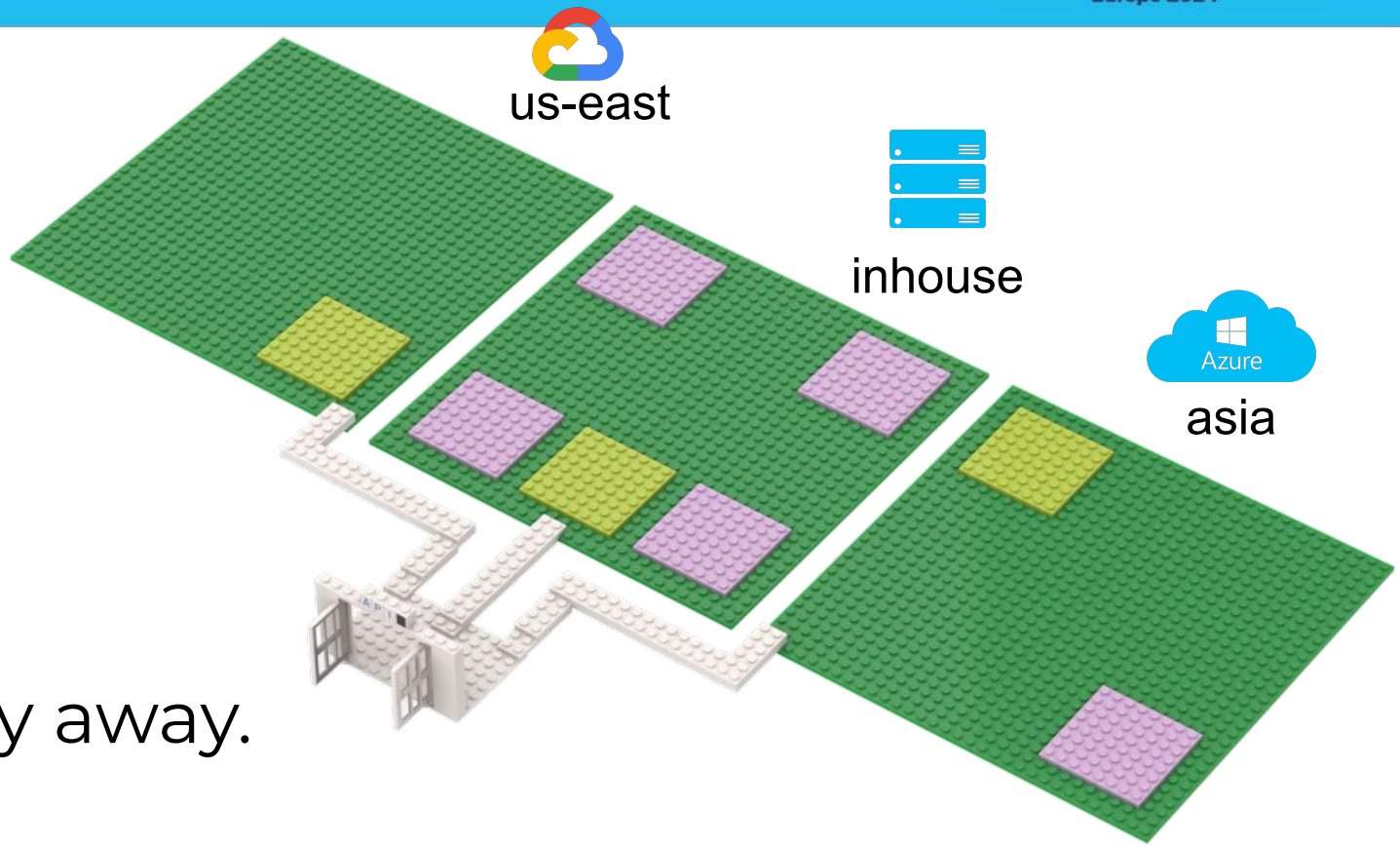
A Kubernetes for Platforms – the Platform Owner

As a platform owner

Give user a well defined, documented platform to ease their job!

Abstract the complexity away.

And scale it horizontally.



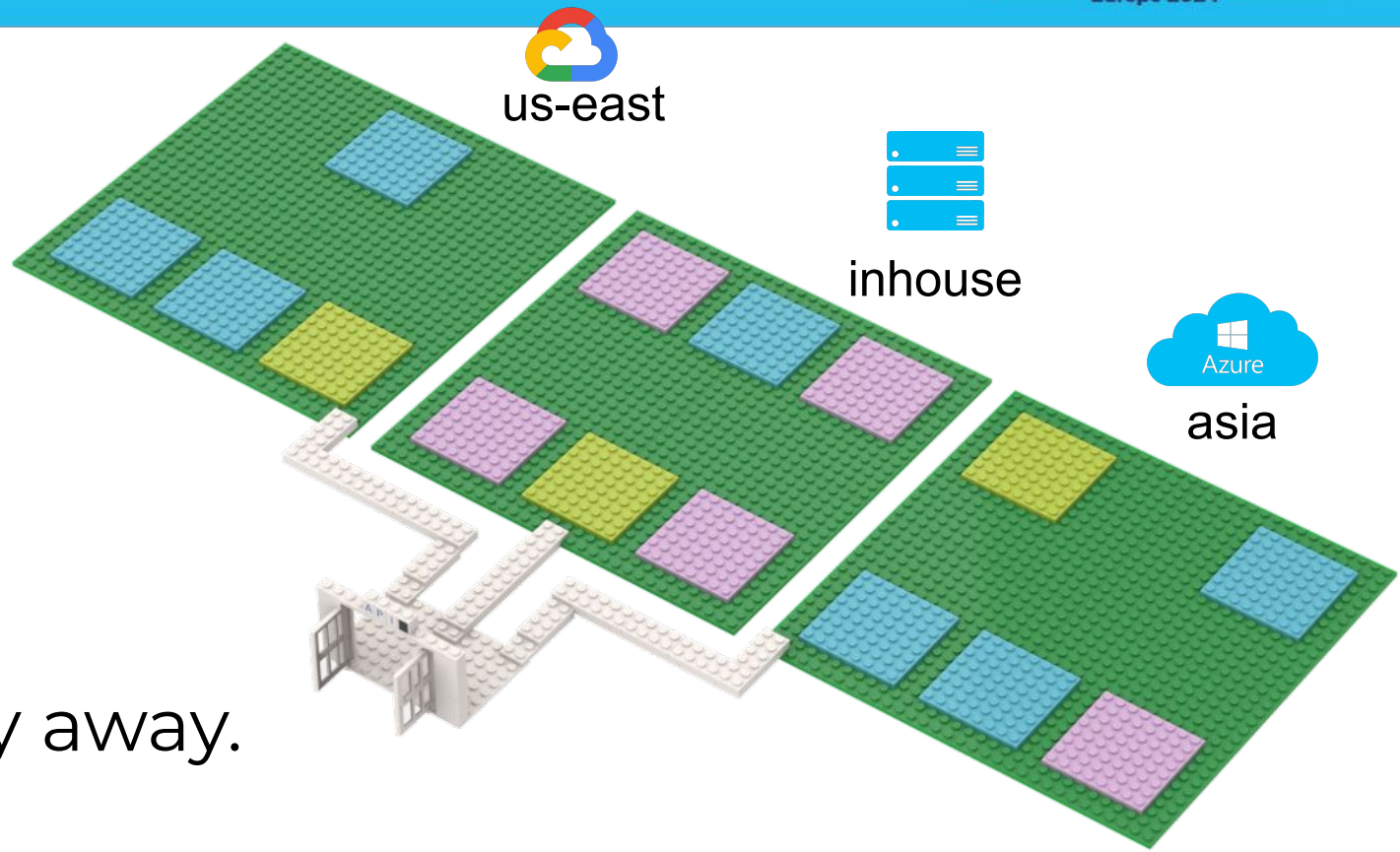
A Kubernetes for Platforms – the Platform Owner

As a platform owner

Give user a well defined,
documented platform
to ease their job!

Abstract the complexity away.

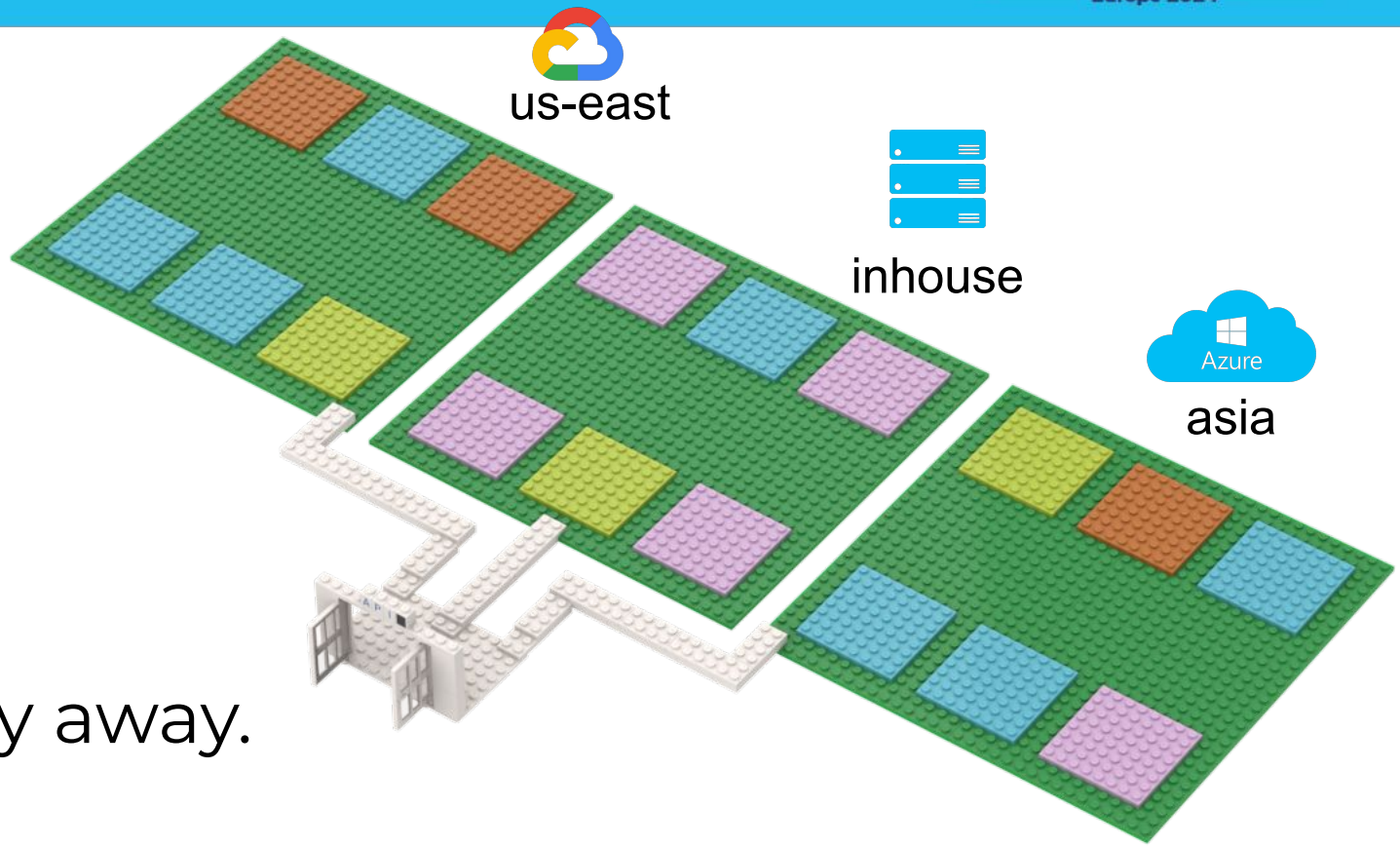
And scale it horizontally
without reinventing the wheel.



A Kubernetes for Platforms – the Platform Owner

As a platform owner

Give user a well defined,
documented platform
to ease their job!



Abstract the complexity away.

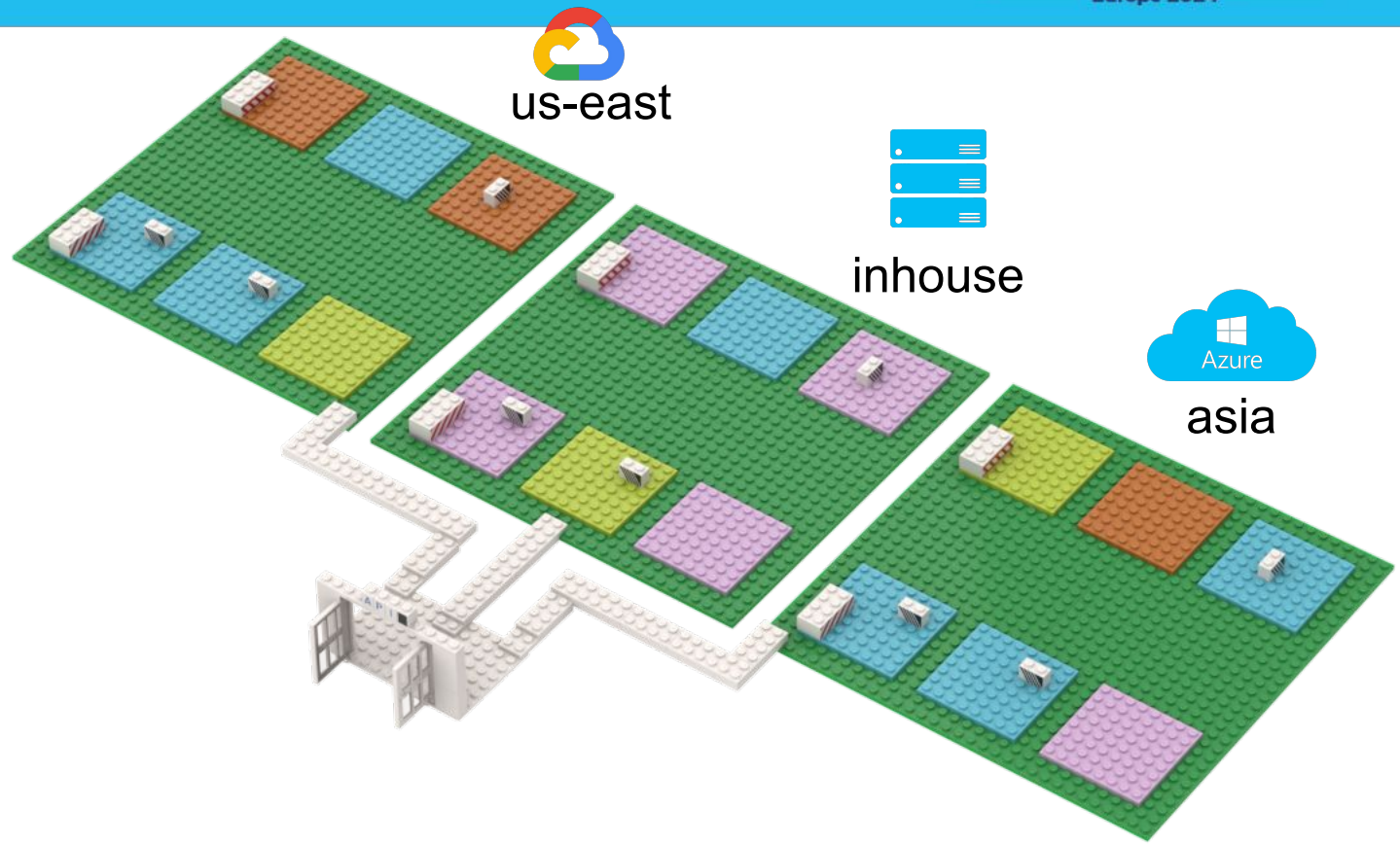
And scale it horizontally
without reinventing the wheel.

Keep it homogenous and simple.

A Kubernetes for Platforms – the Platform Owner

KubeCon CloudNativeCon Europe 2024

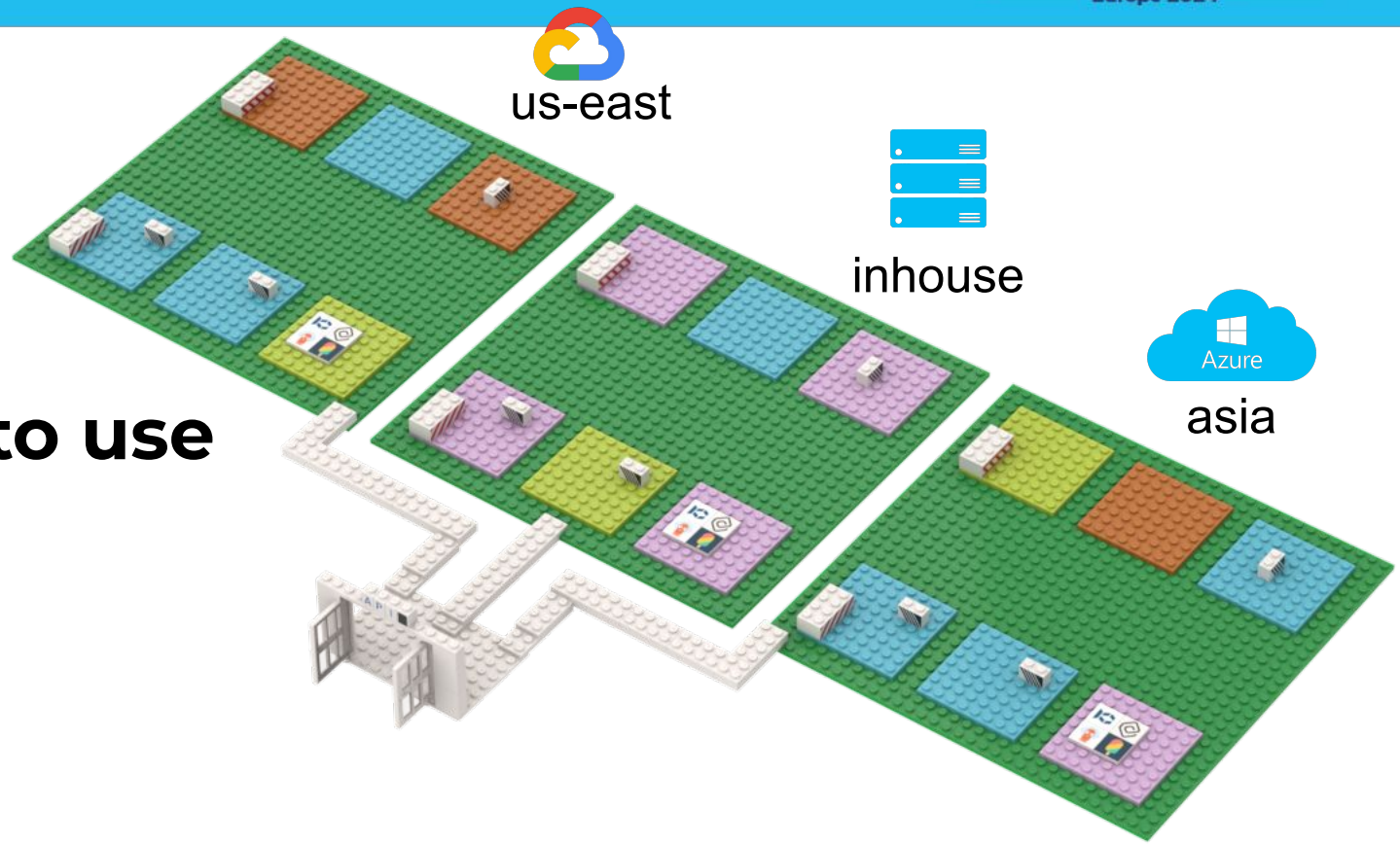
Let your users build.



A Kubernetes for Platforms – the Platform Owner

Let your users build

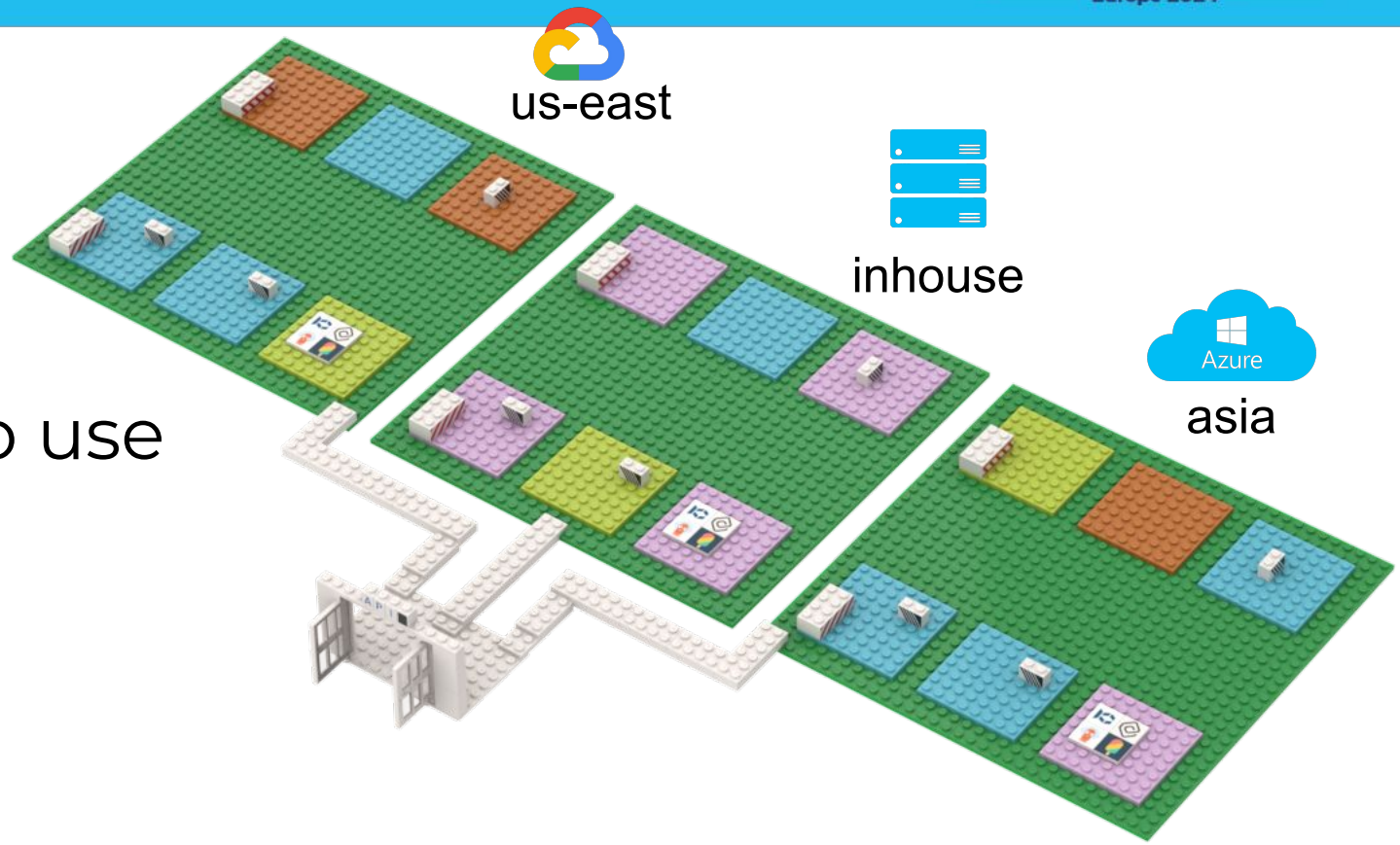
You should not have opinions on them what to use and what not.



A Kubernetes for Platforms – the Platform Owner

Let your users build

You should not have opinions on them what to use and what not.



Give them services to consume.

If they give value, your users will use them.

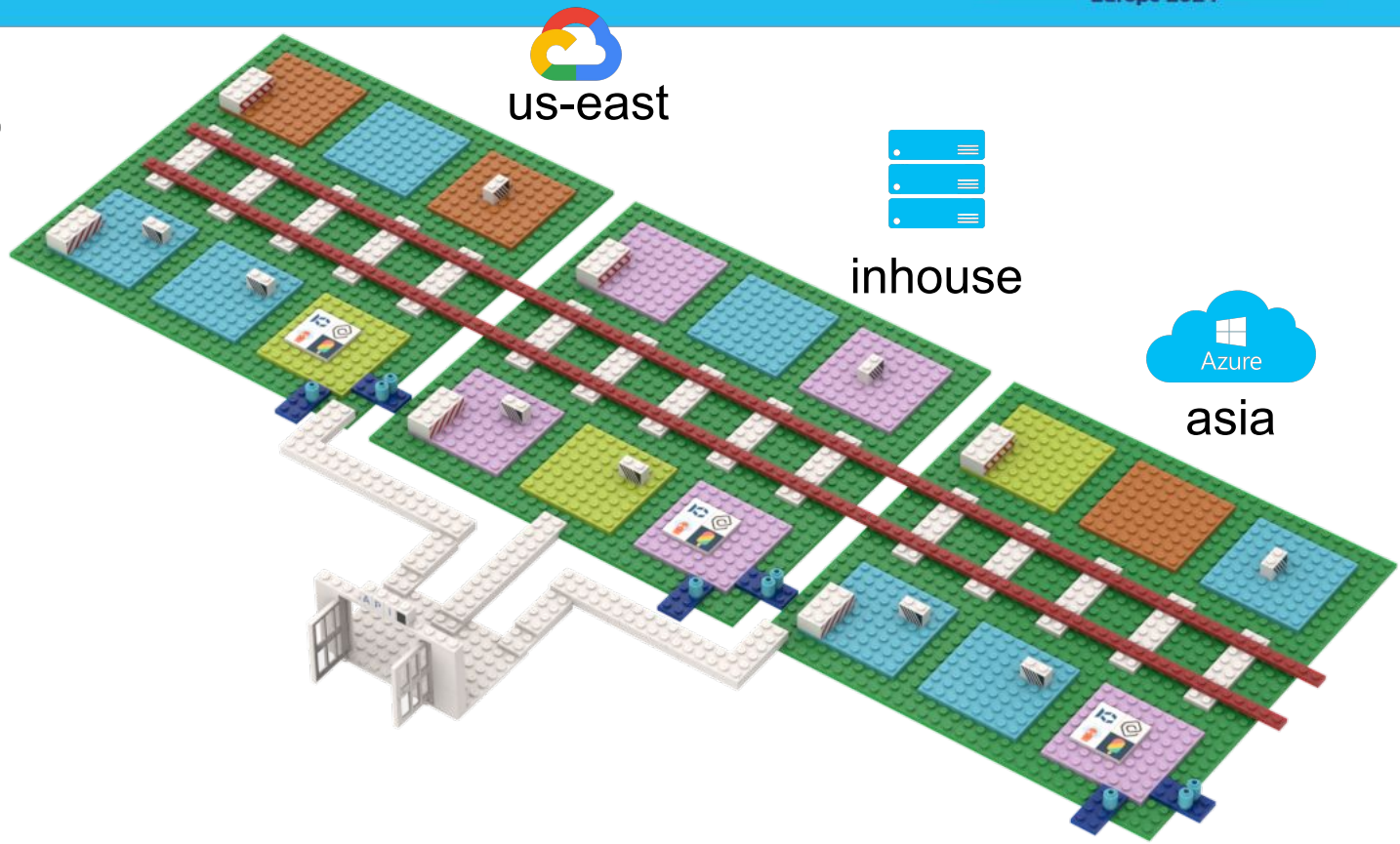
The service provider

DB team, PKI team,
CI/CD service, 3rd-parties
service owner

A Kubernetes for Platforms – the Service Provider

As an integrator you want to

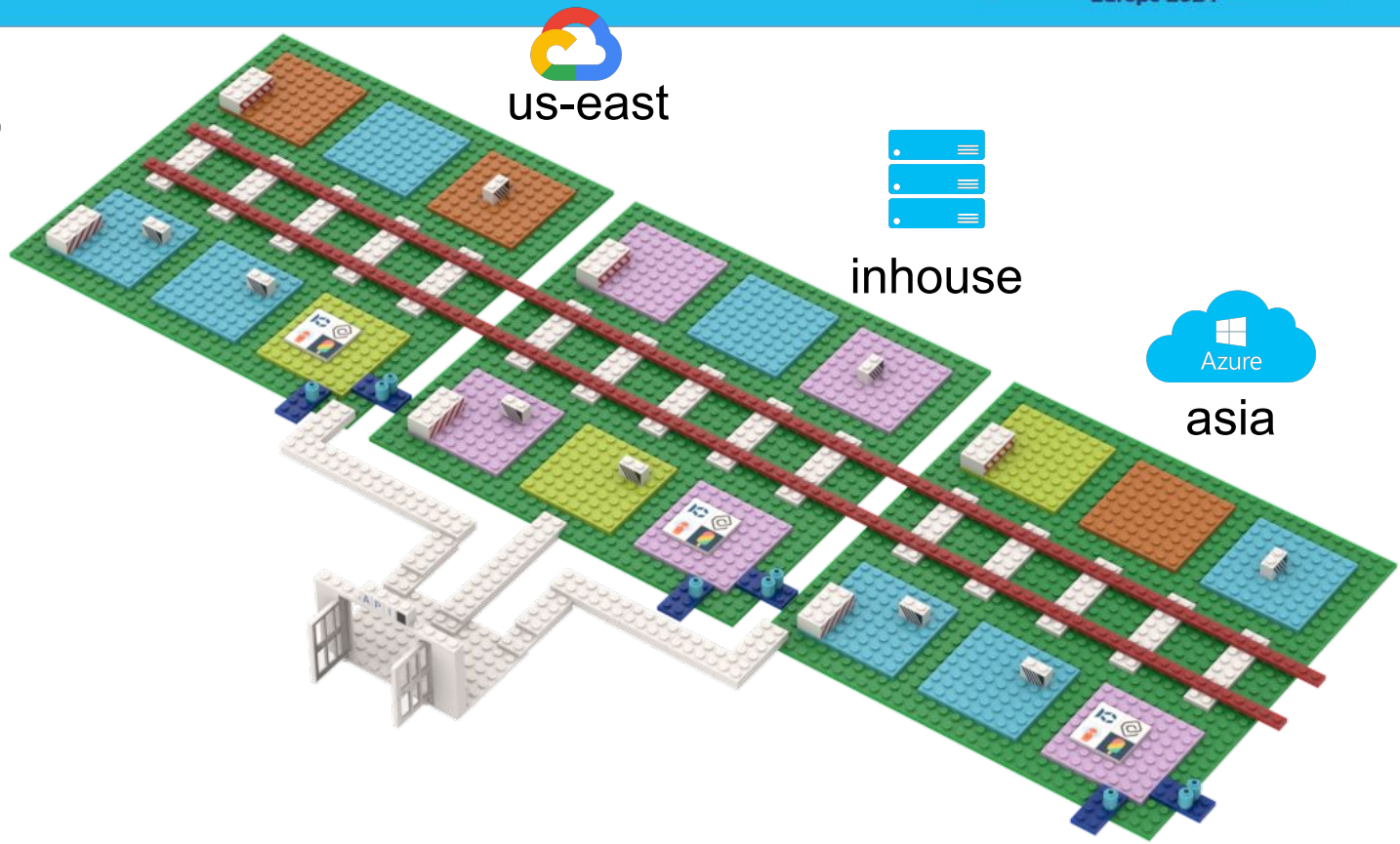
offer a service.



A Kubernetes for Platforms – the Service Provider

As an integrator you want to offer a service.

Consistently.
Efficiently.
Safely.



St

A Kubernetes for Platforms – the Service Provider

As an integrator you want to
offer a service.

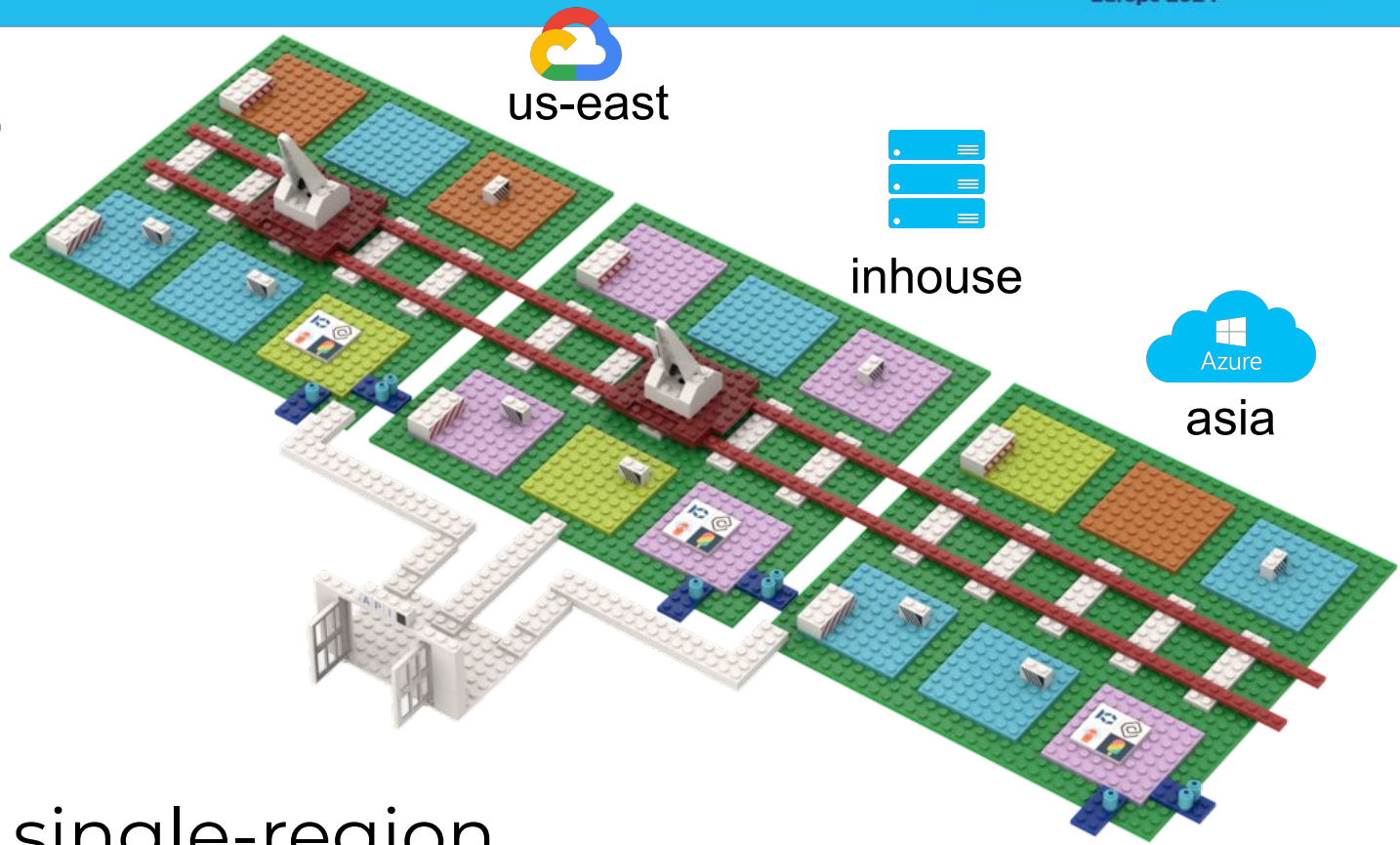
Consistently.
Efficiently.
Safely.

Same set of tools for single-region.

A Kubernetes for Platforms – the Service Provider

As an integrator you want to
offer a service.

Consistently.
Efficiently.
Safely.

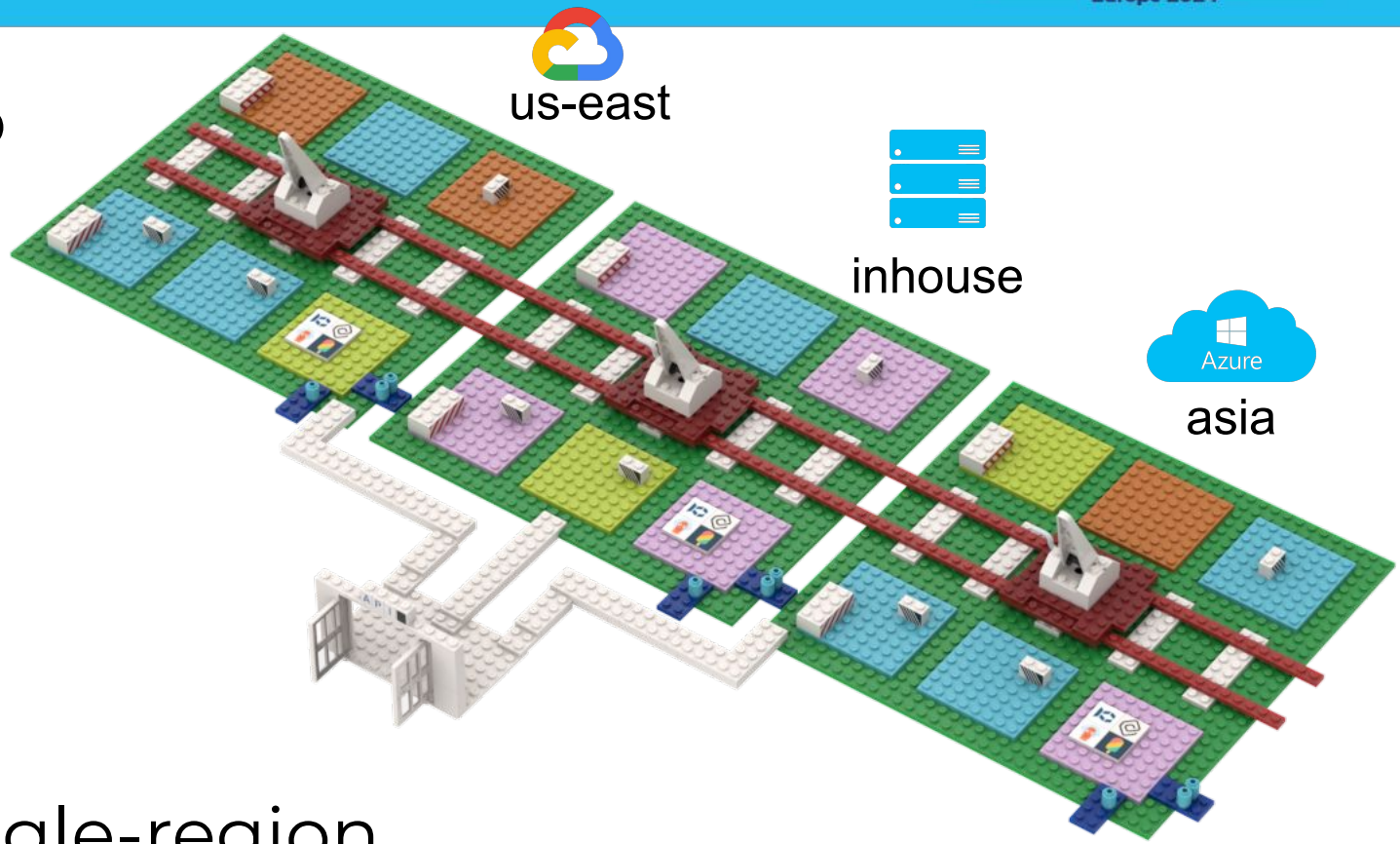


Same set of tools for single-region
or multi-region.

A Kubernetes for Platforms – the Service Provider

As an integrator you want to
offer a service.

Consistently.
Efficiently.
Safely.

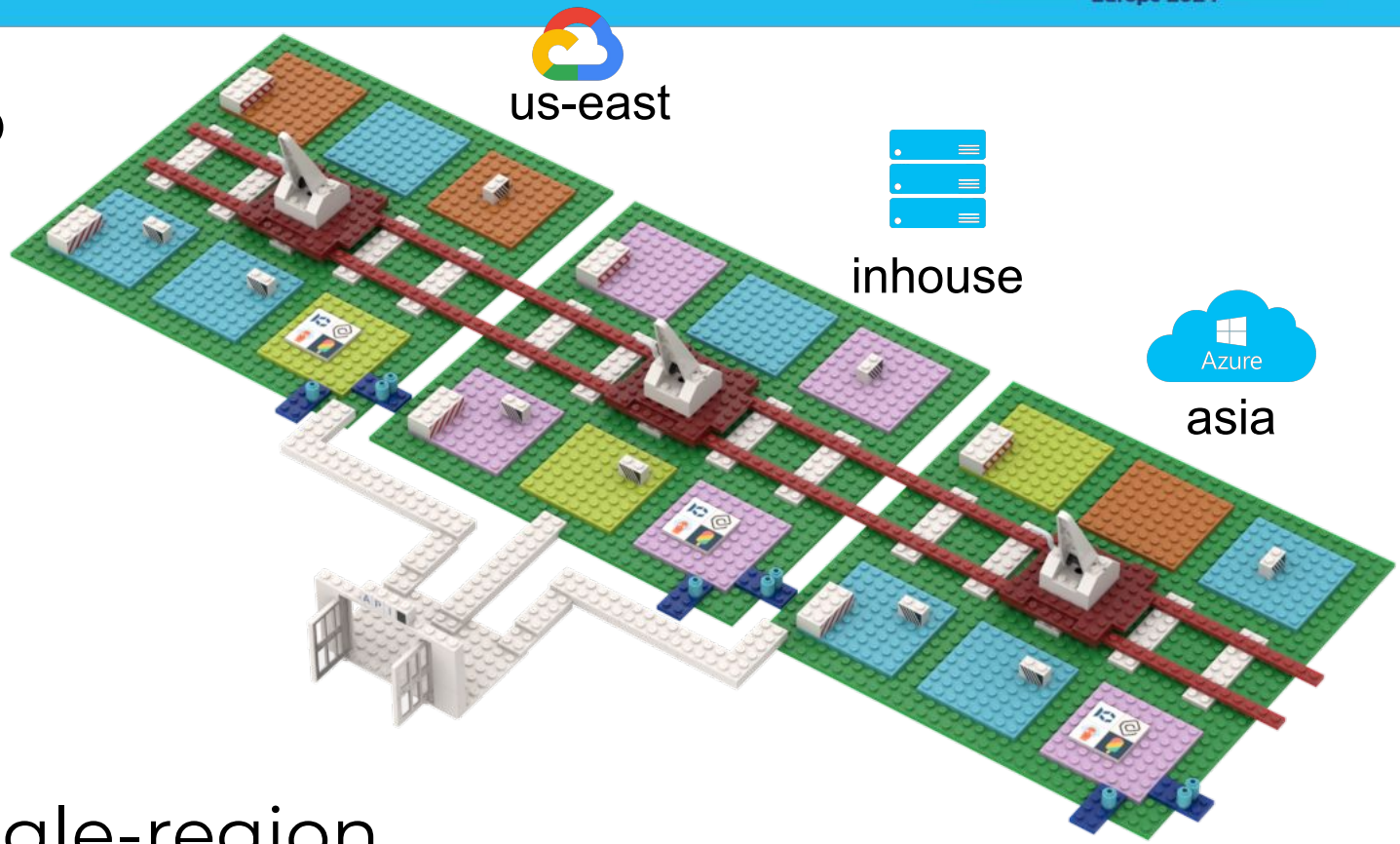


Same set of tools single-region
or multi-region **or multi-cloud.**

A Kubernetes for Platforms – the Service Provider

As an integrator you want to
offer a service.

Consistently.
Efficiently.
Safely.



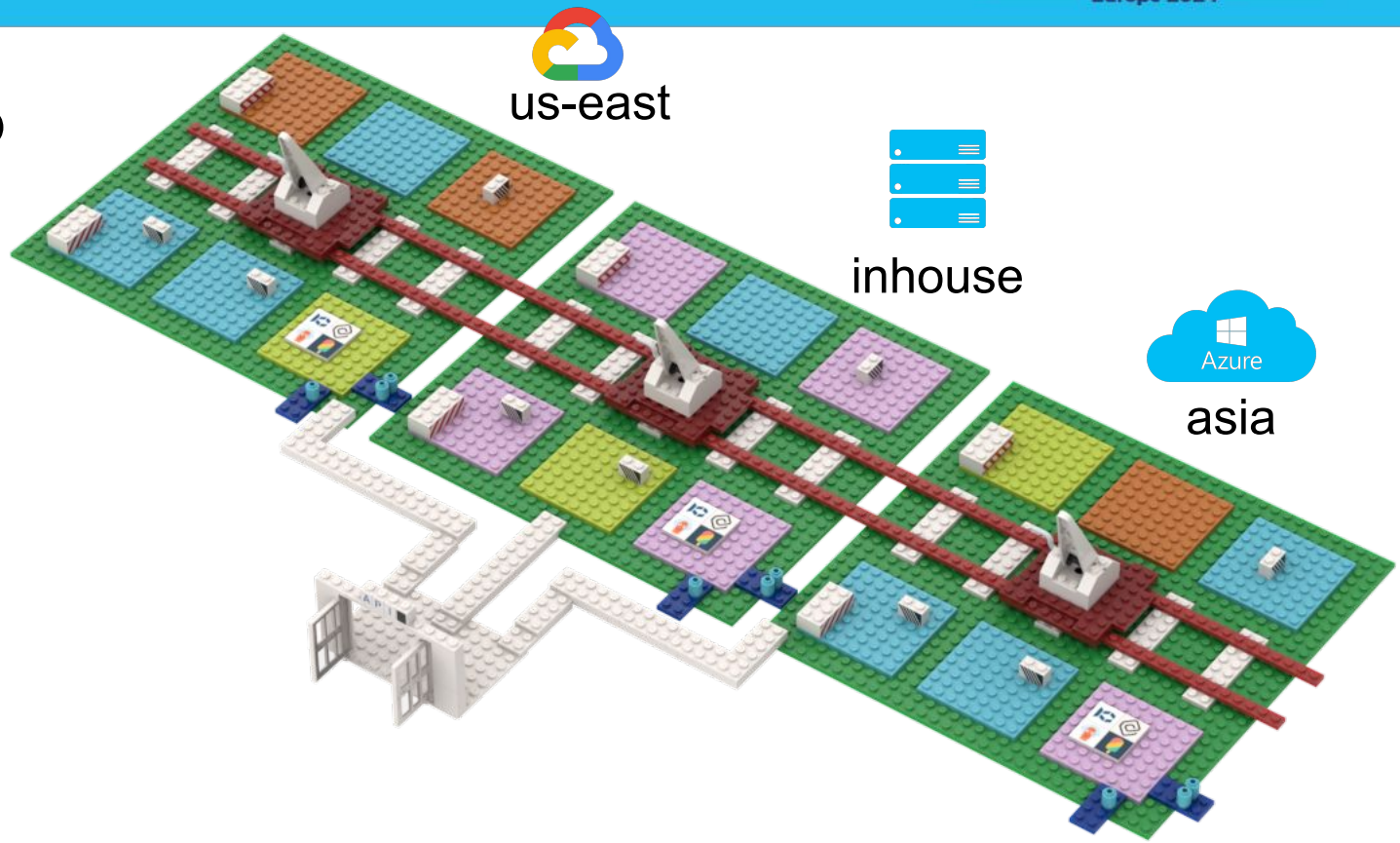
Same set of tools single-region
or multi-region or multi-cloud.

Towards a Global service as a Product.

A Kubernetes for Platforms – the Service Provider

As an integrator you want to offer a service.

Consistently.
Efficiently.
Safely.
Securely.



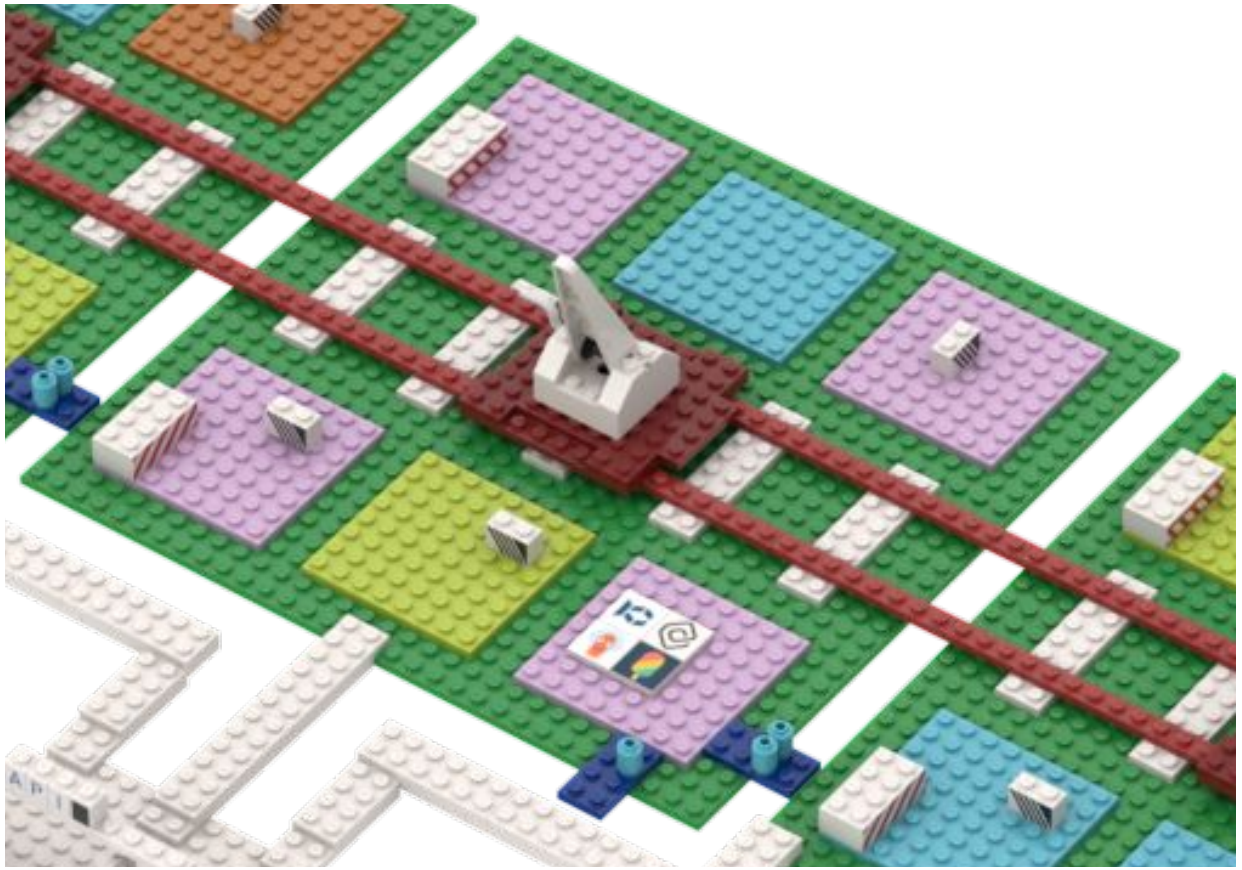
Same set of tools single-region or multi-region or multi-cloud.

Towards a Global service as a Product.

A Kubernetes for Platforms – the Service Provider

KubeCon CloudNativeCon Europe 2024

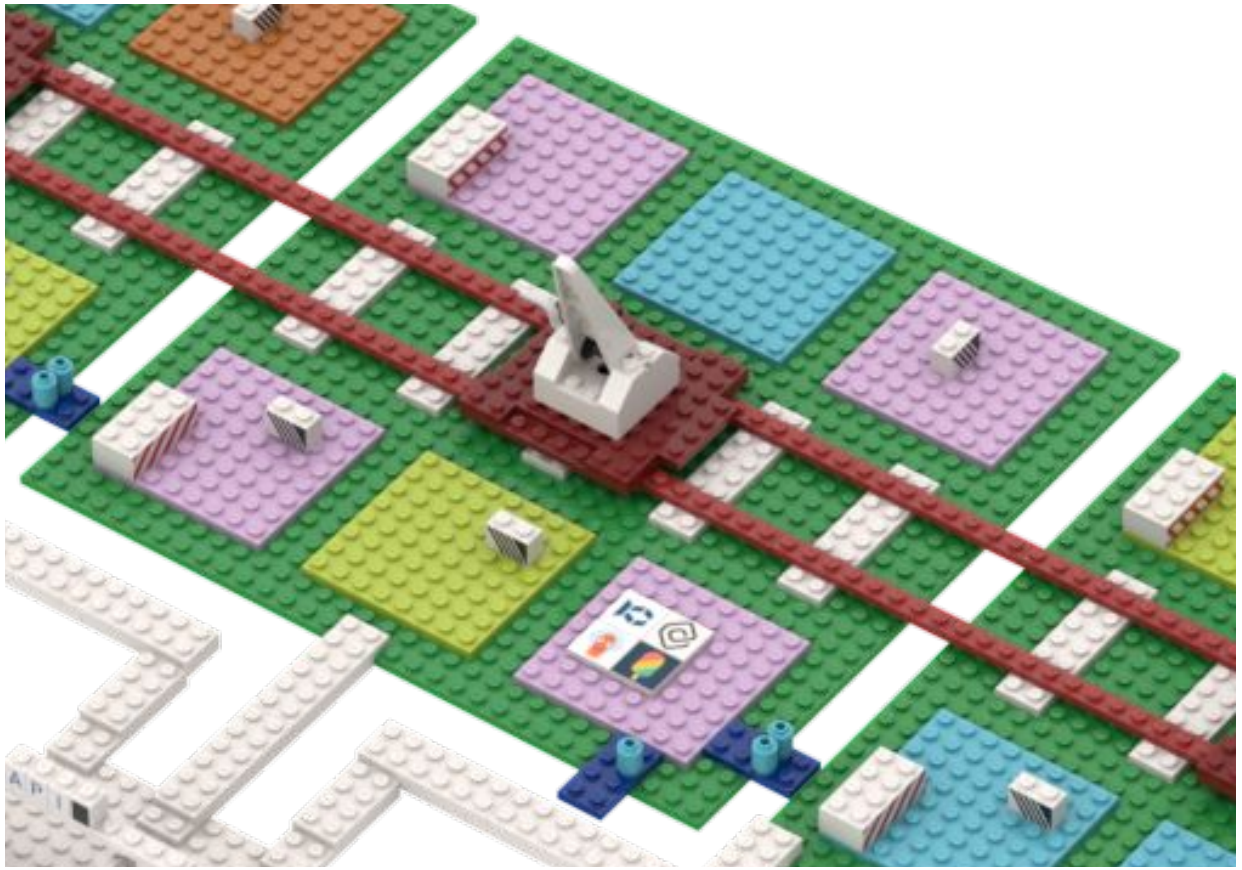
Which primitives does this need ?



A Kubernetes for Platforms – the Service Provider

Which primitives
does this need ?

Are CRDs enough ?

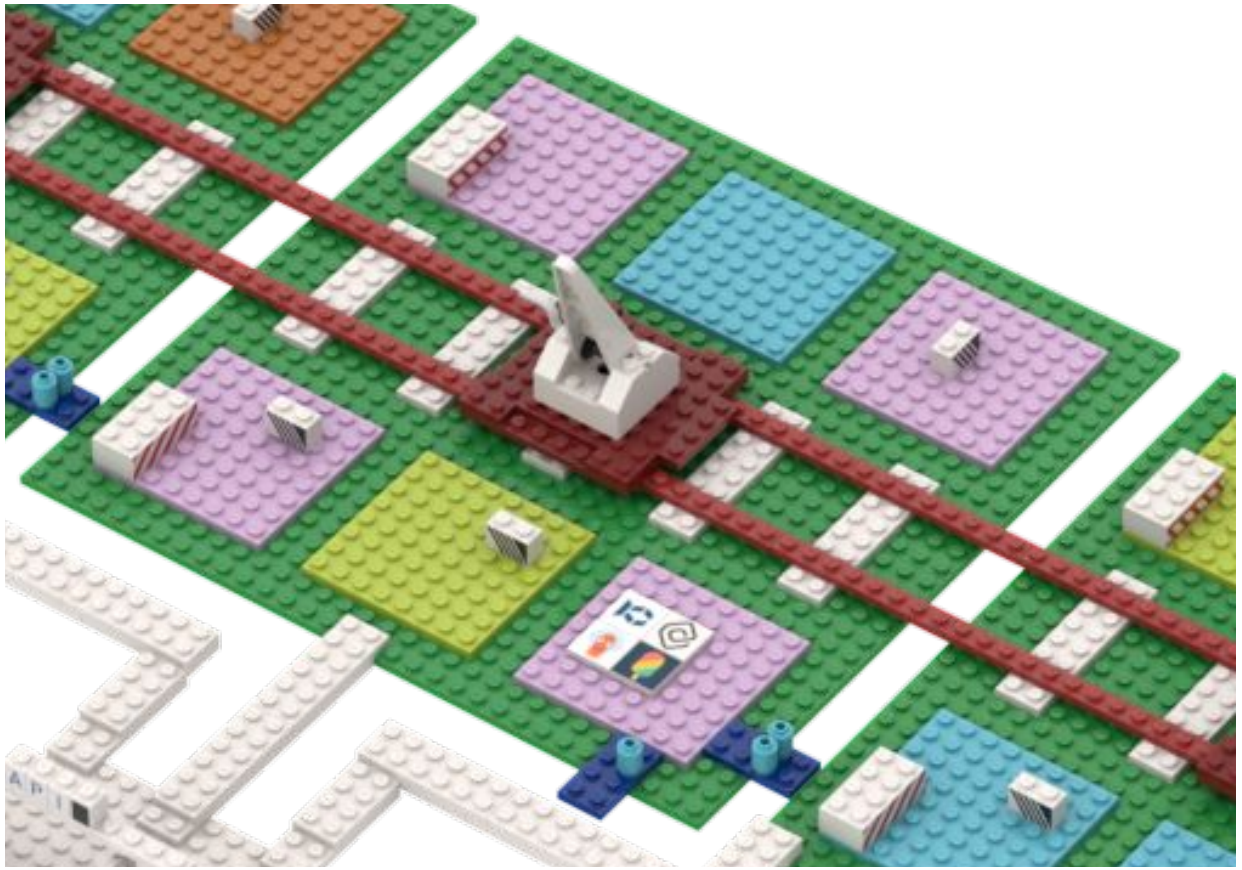


A Kubernetes for Platforms – the Service Provider

Which primitives
does this need ?

Are CRDs enough ?

Tools like Helm ?



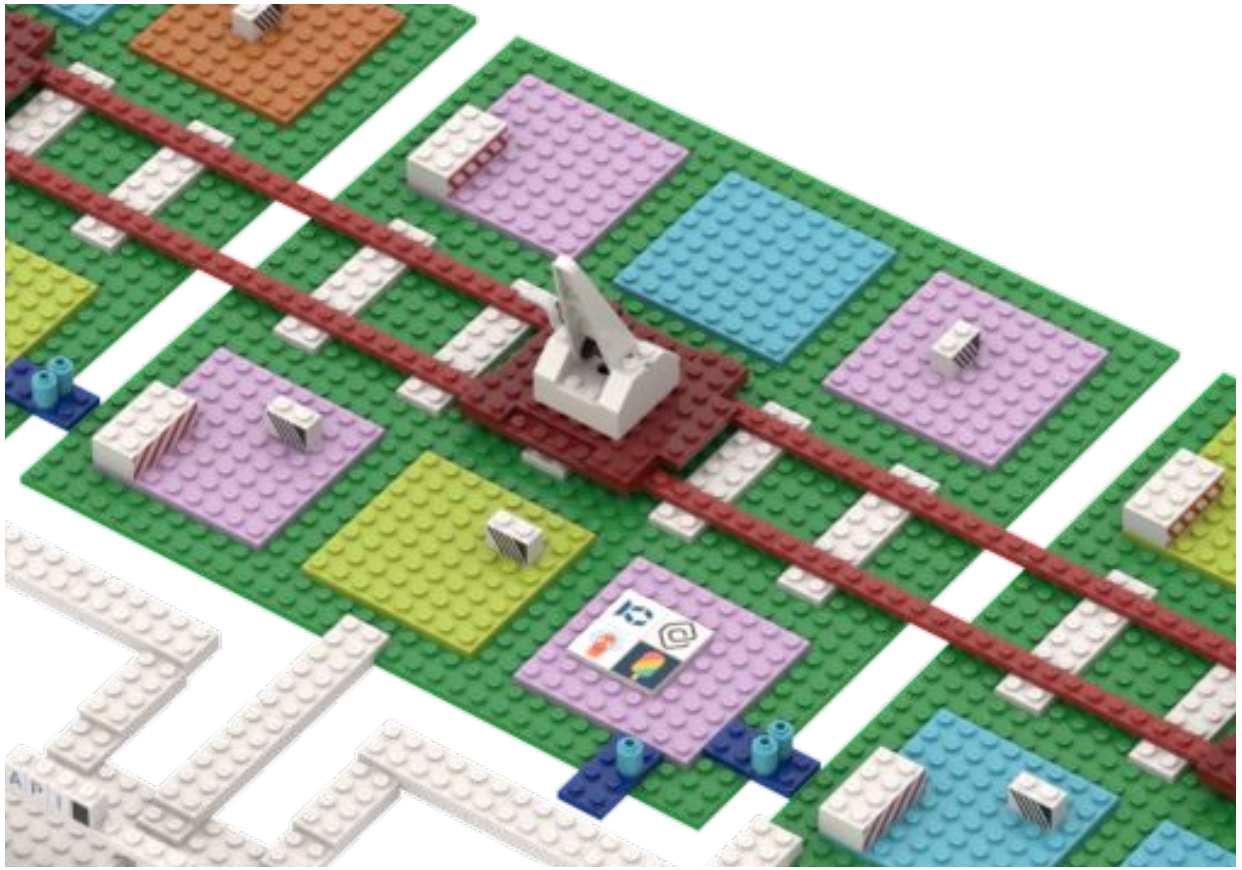
A Kubernetes for Platforms – the Service Provider

Which primitives
does this need ?

Are CRDs enough ?

Tools like Helm ?

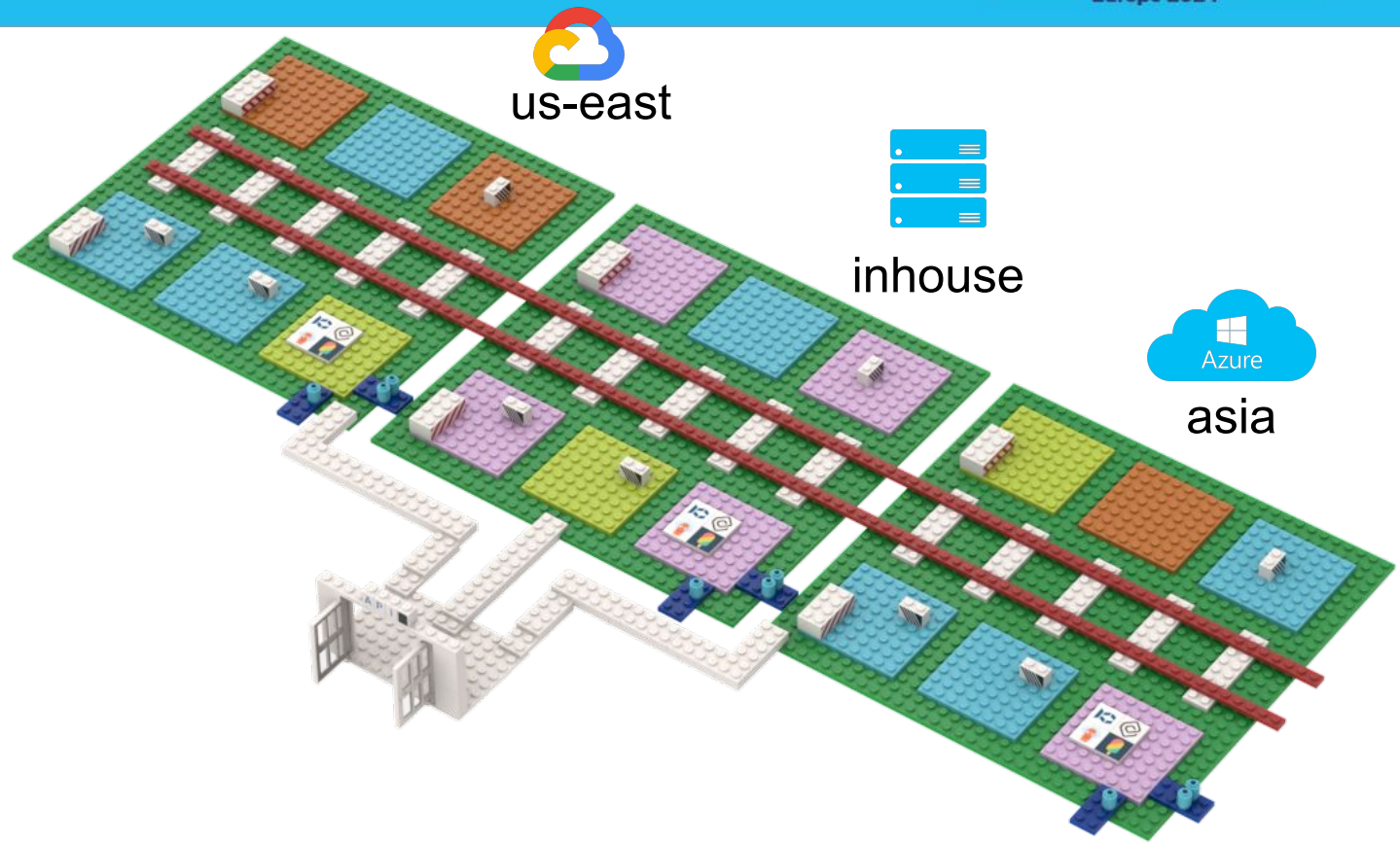
Kube RBAC ?



A Kubernetes for Platforms – the Service Provider

KubeCon CloudNativeCon Europe 2024

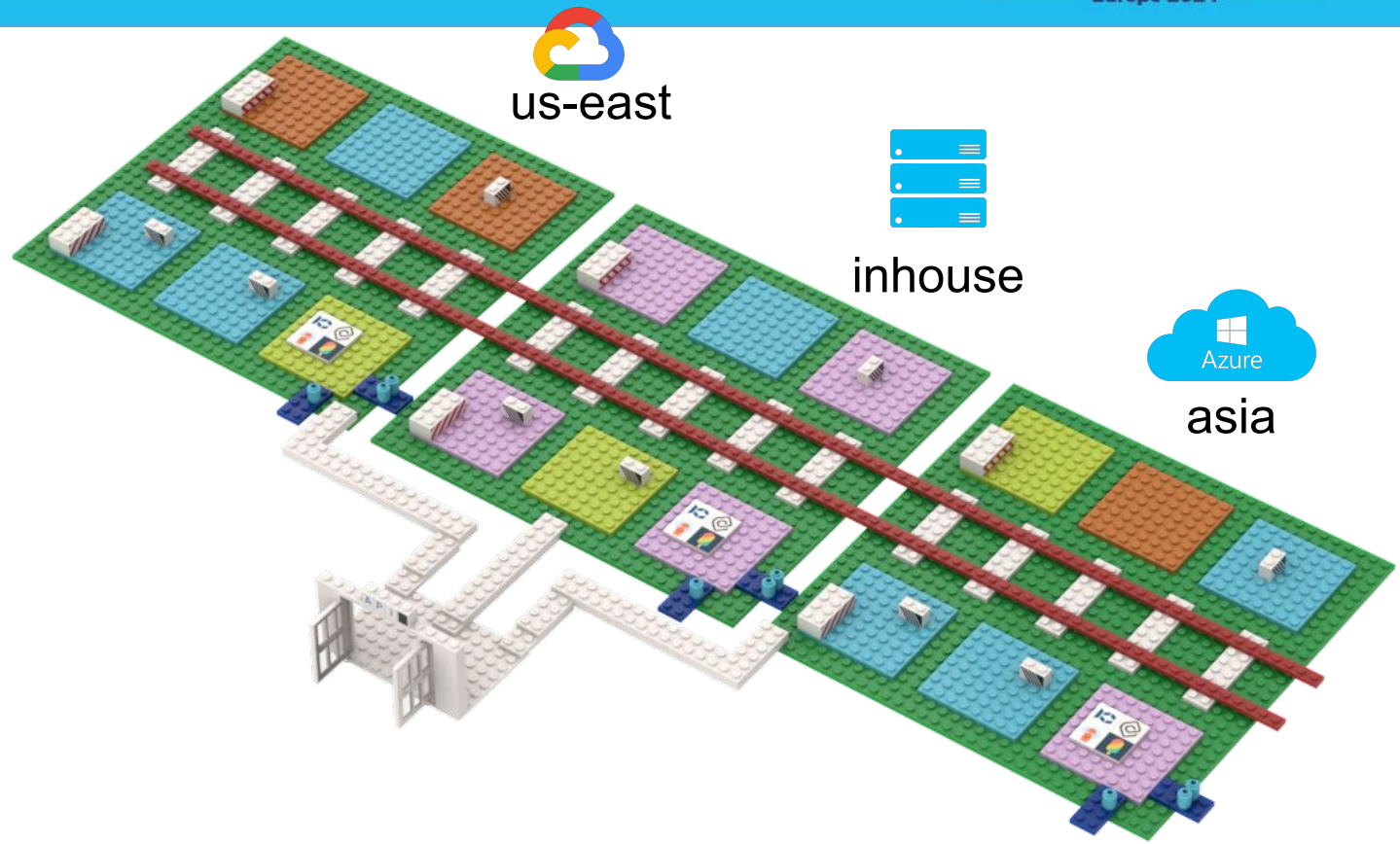
But what about compute?



A Kubernetes for Platforms – the Service Provider

But what about compute?

Compute is just another API.



A Kubernetes for Platforms – the Service Provider

But what about compute?

Compute is just another API.
“Utility Computing”.

A Kubernetes for Platforms – the Service Provider

But what about compute?

Compute is just another API.
“Utility Computing”.

We know that already with ArgoCD or Flux or Crossplane.

A Kubernetes for Platforms – the Service Provider

But what about compute?

Compute is just another API.
Utility Computing.

We know that already with
ArgoCD or Flux or Crossplane.

Many flavors, different APIs:
vclusters, KubeVirt, Karmada, Cluster API

A Kubernetes for Platforms – the Service Provider

But what about compute?

Compute is just another API.
Utility Computing.

We know that already with
ArgoCD or Flux or Crossplane.

Many flavors, different APIs:
vclusters, KubeVirt, Karmada, Cluster API

Compute is attached. Compute is not the center.

St

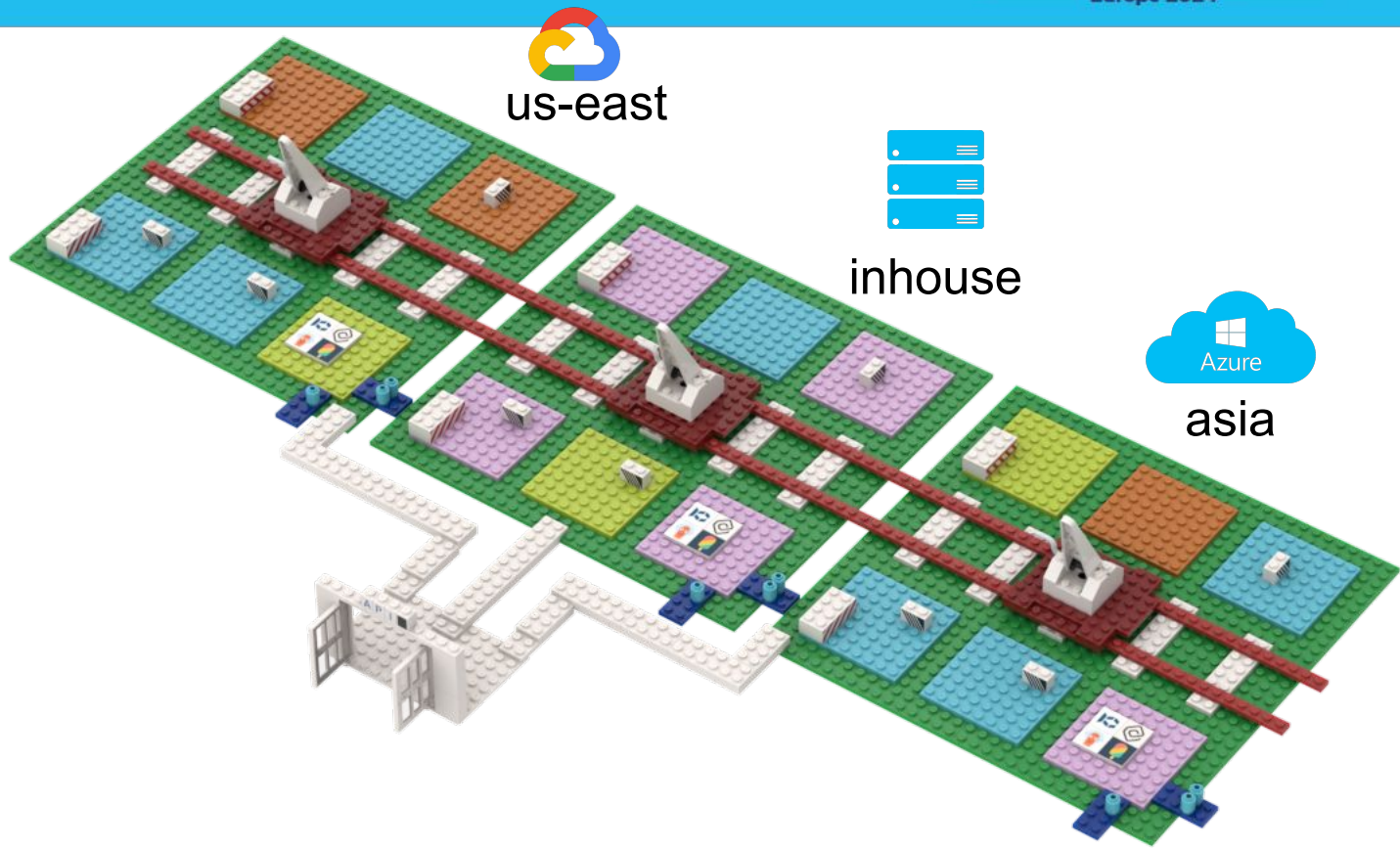
The users

Developer, application owner

A Kubernetes for Platforms – the User

Decoupled, multi-tenant, distributed control planes

“Workspaces”

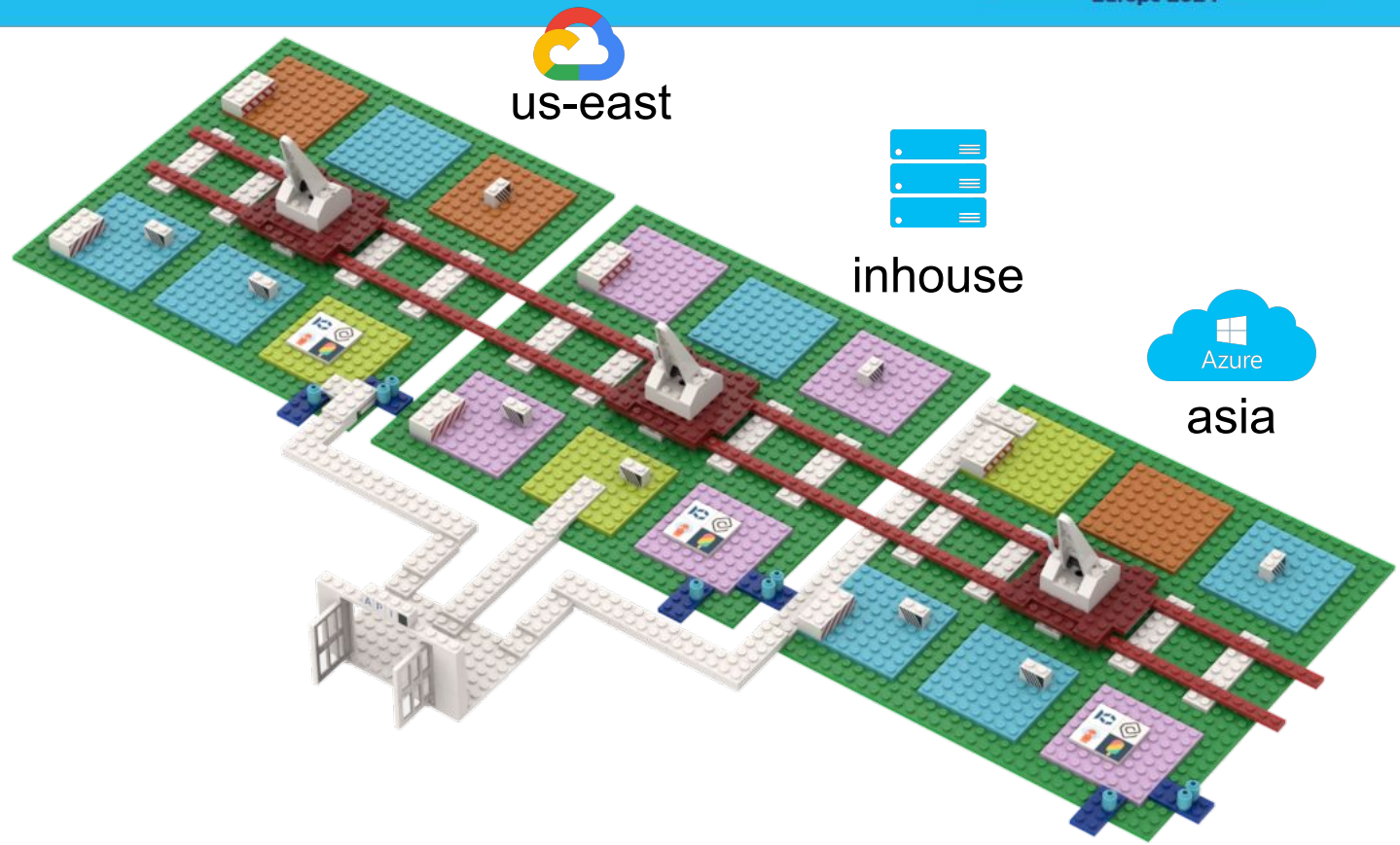


A Kubernetes for Platforms – the User

Decoupled, multi-tenant, distributed control planes

"Workspaces"

all logically **connected**.



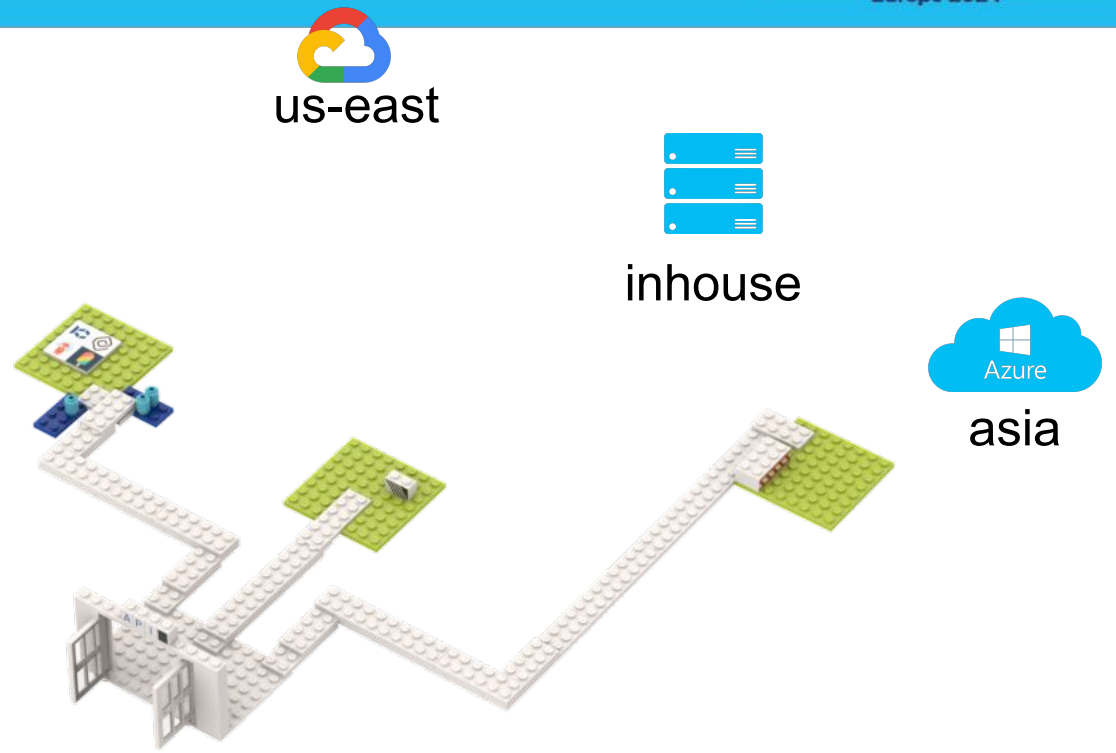
A Kubernetes for Platforms – the User

Decoupled, multi-tenant, distributed control planes

"Workspaces"

all logically connected.

Easy to navigate.



A Kubernetes for Platforms – the User

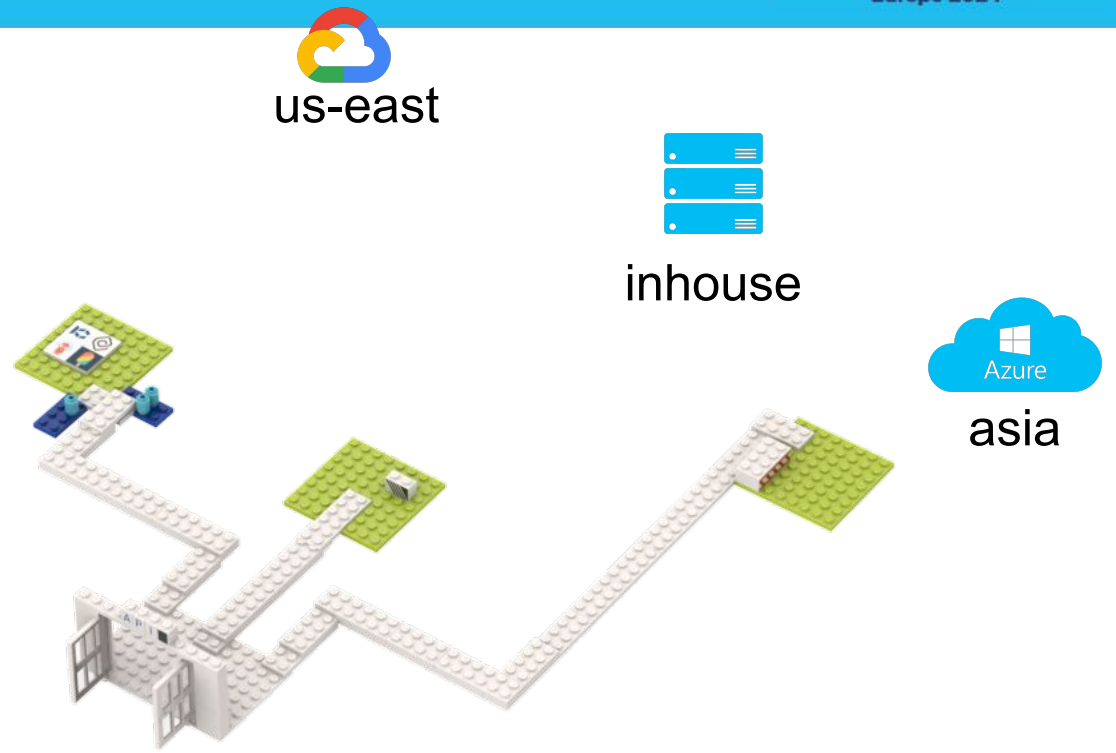
Decoupled, multi-tenant, distributed control planes

"Workspaces"

all logically connected.

Easy to navigate.

from app to app
from region to region
from cloud to cloud.



What we described is not Kubernetes.

You cannot build it just with more clusters.

We need new primitives.

New primitives for a world that is not just containers
for a world that is inherently multi-tenant/region/cloud.

New primitives to regain speaking the same language.

New primitives to collaborate on together.

KCP - Kubernetes-like Control Planes

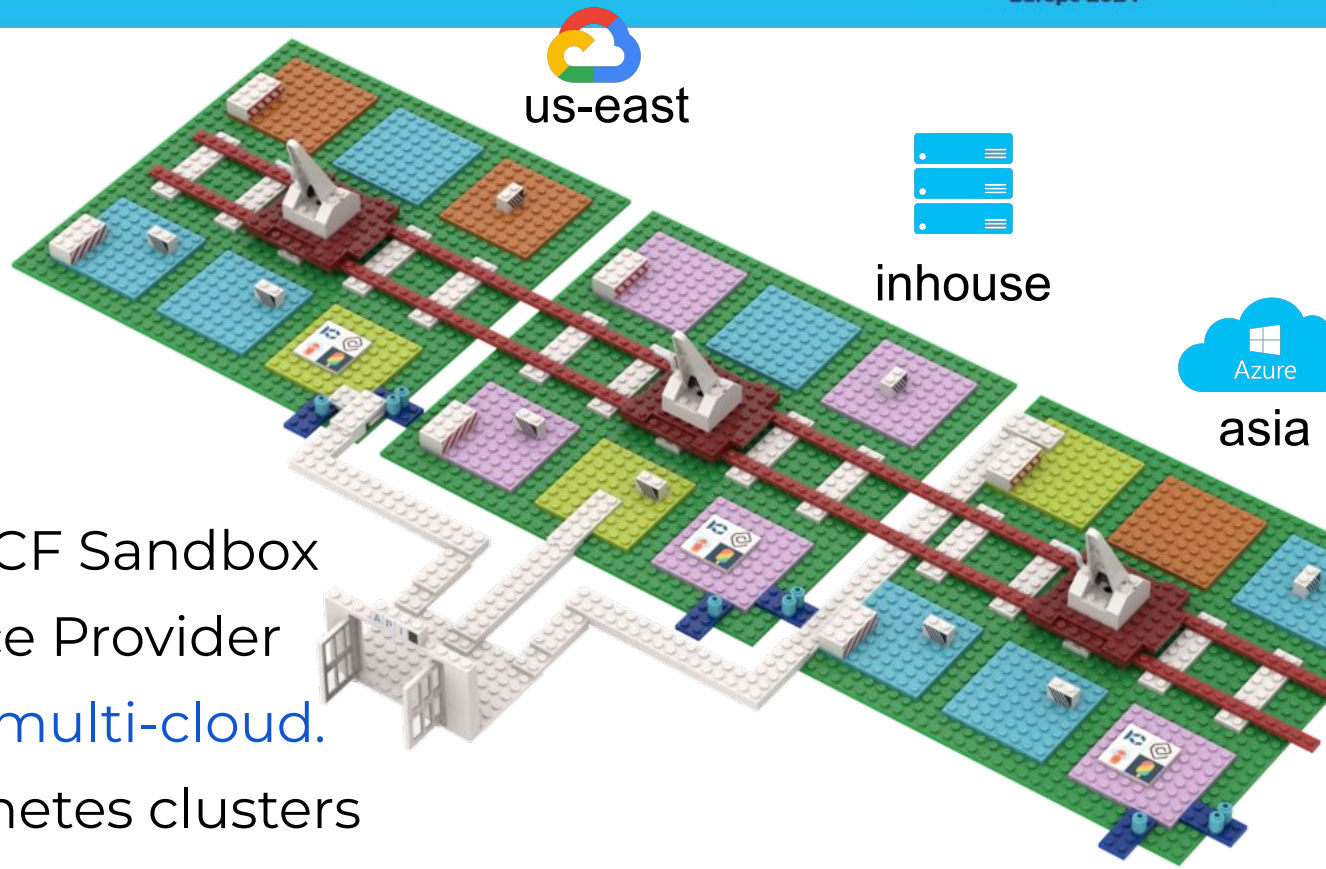
- CNCF Sandbox project
- Framework to bootstrap Kubernetes-like SaaS/Platforms
- Work in progress to move “generic controlplanes” to upstream

Demo

```
# region:name:root = Europe
# region:name:beta = US
$
```

KCP – a framework exploring these ideas

kcp



- Kubernetes like APIs
- Based on Kubernetes source code, CNCF Sandbox
- Platform Personas: Owner, User, Service Provider
- Inherently multi-tenant, multi-region, multi-cloud.
- Workspaces, each like (generic) Kubernetes clusters
- All under one API endpoint.
- Horizontally scalable (sharding)
- Extended API management for service providers

Talk to us - look for logo! We have stickers!

Find us at #kcp-dev kubernetes slack

github.com/kcp-dev/kcp

@kcp @the_sttts @sscheele @mangirdas

Find us at one of these booths!

G14 L11 E7

Slack channel QR

Feedback

Thanks