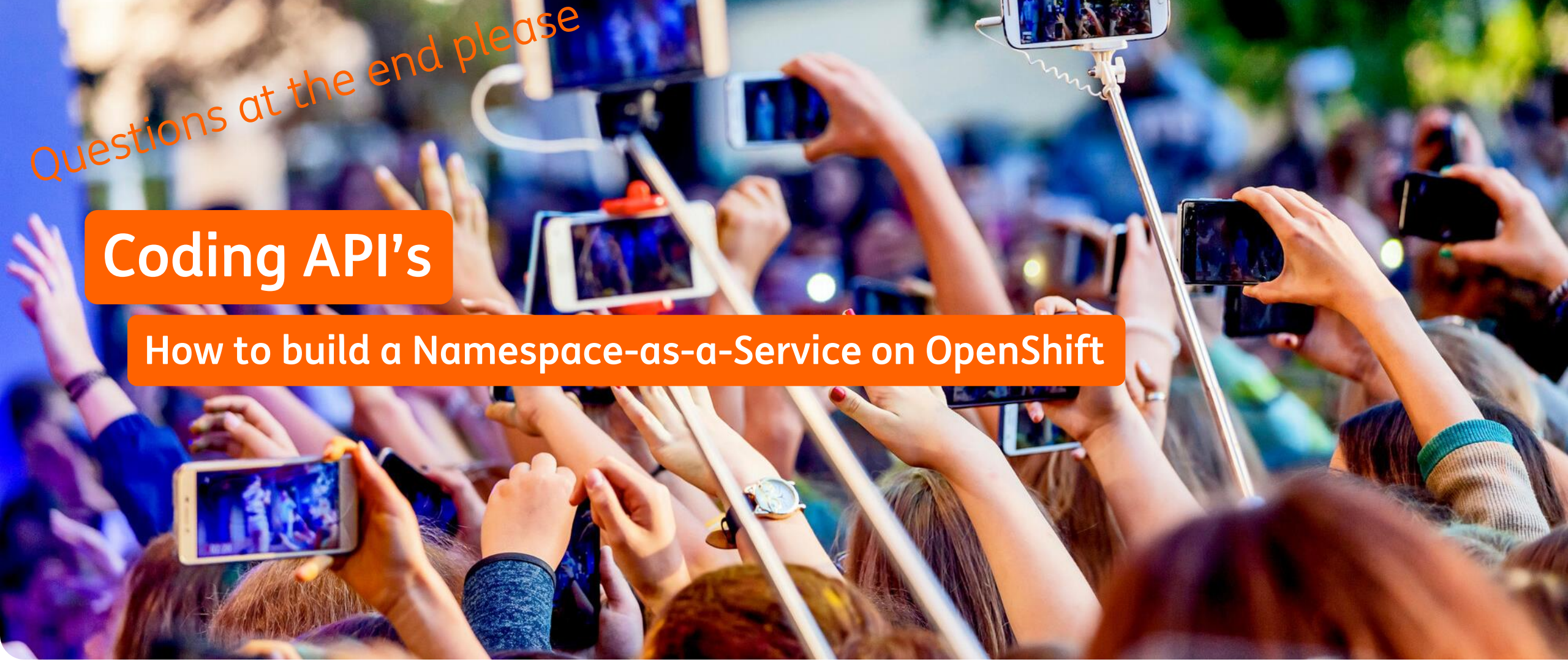


Coding API's

How to build a Namespace-as-a-Service on OpenShift

ING Tech Infra&Engineering

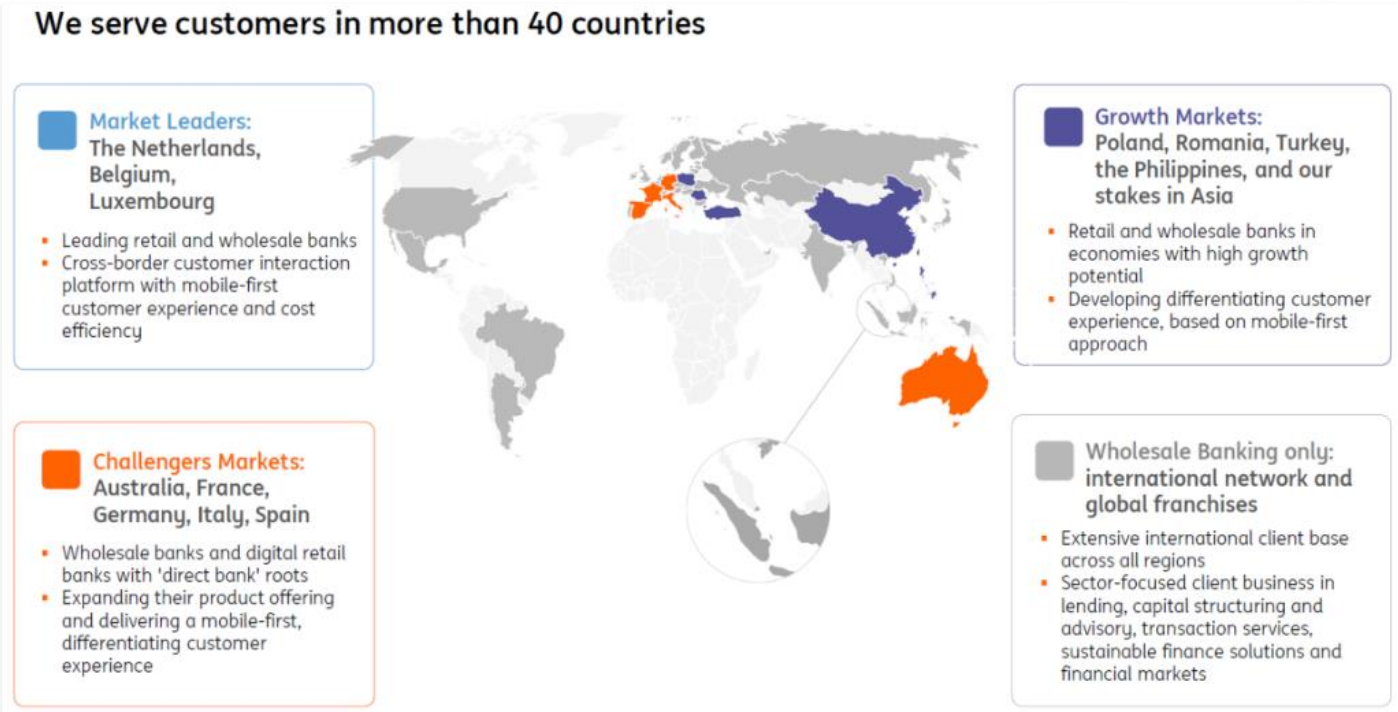
October 2022

ING

Introductions

Arno Vonk
Product Owner ING Container Hosting Platform & Namespace-as-a-Service since 2018

Jan-Willem Bijma
Engineering ING Container Hosting Platform



ING is a global bank with a strong European base. Our more than 57,000 employees serve around 38 million customers, corporate clients and financial institutions in over 40 countries. Our purpose is to empower people to stay a step ahead in life and in business.

* ING is undergoing a transition to close our Wholesale Banking offices in Argentina, Brazil and Kazakhstan as announced on 5 November 2020. ING has exited the Austrian and Czech retail banking markets as of the end of 2021. We announced in December 2021 that we will leave the retail banking market in France after a strategic review of our retail banking operations there. We announced June 24th 2022 we will leave the Philippines retail banking market as of then end of 2022.

For ING's position regarding Russia and the Ukraine see our Feb 28 2022 statement

Agenda

- Why Namespace-aaS ?
- NaaS in a Private Cloud
- Building a NaaS on top of OCP 4.10 IPI
- NaaS Dependencies overview
- GitOps@ICHP / Demo 1





- Project Controller
- ICHP API
- Quota AutoScaler
- Demo 2
- Surprise





ING

Why ICHP only offers a “Namespace-as-a-Service”

ING aims to make our DevOps teams autonomous (As explained in ING’s Way-of-Working in 2019).
We also want to enable those DevOps teams to deliver maximum value to the business, by not bothering them with IT-Infrastructure concerns, nor bothering them with having to deliver compliancy evidence for the hosting platform.

Hence offering a Namespace-as-a-Service is for us the sweet spot:

- Clear demarcation between (Infra)Provider and Consumer
  - Enabling hand-over of compliance evidence
  - Enabling multi-tenancy (hence fast time-to-market/self-service consumption, and the potential for efficient utilization of resources)
- The DevOps team can assume responsibility for (almost) the full stack (only the kernel stays shared). They have liberty/responsibility to choose/maintain their (versions of) base image, runtime engine, libraries, etc. *(within the boundaries set by the ING corporate risk&compliancy rules !)*

Namespace-aaS in a Private Cloud

“oc adm new-project myproject” right ?

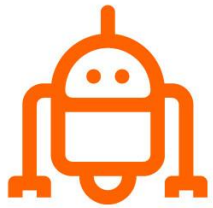
Not really….

- How to identify this namespace in your private cloud ecosystem ?
- How to charge the resources used for this namespace ?
- How to request a subnet and IP address for external firewall rules / egress IP ?
- How to create a trust relationship with the workload deployment pipeline ?
- How to make sure an identical namespace is created in DC2 (for a dual DC setup) ?

And do all of the above (and more) in an Automated (“Zero-Touch”) way ?

ING

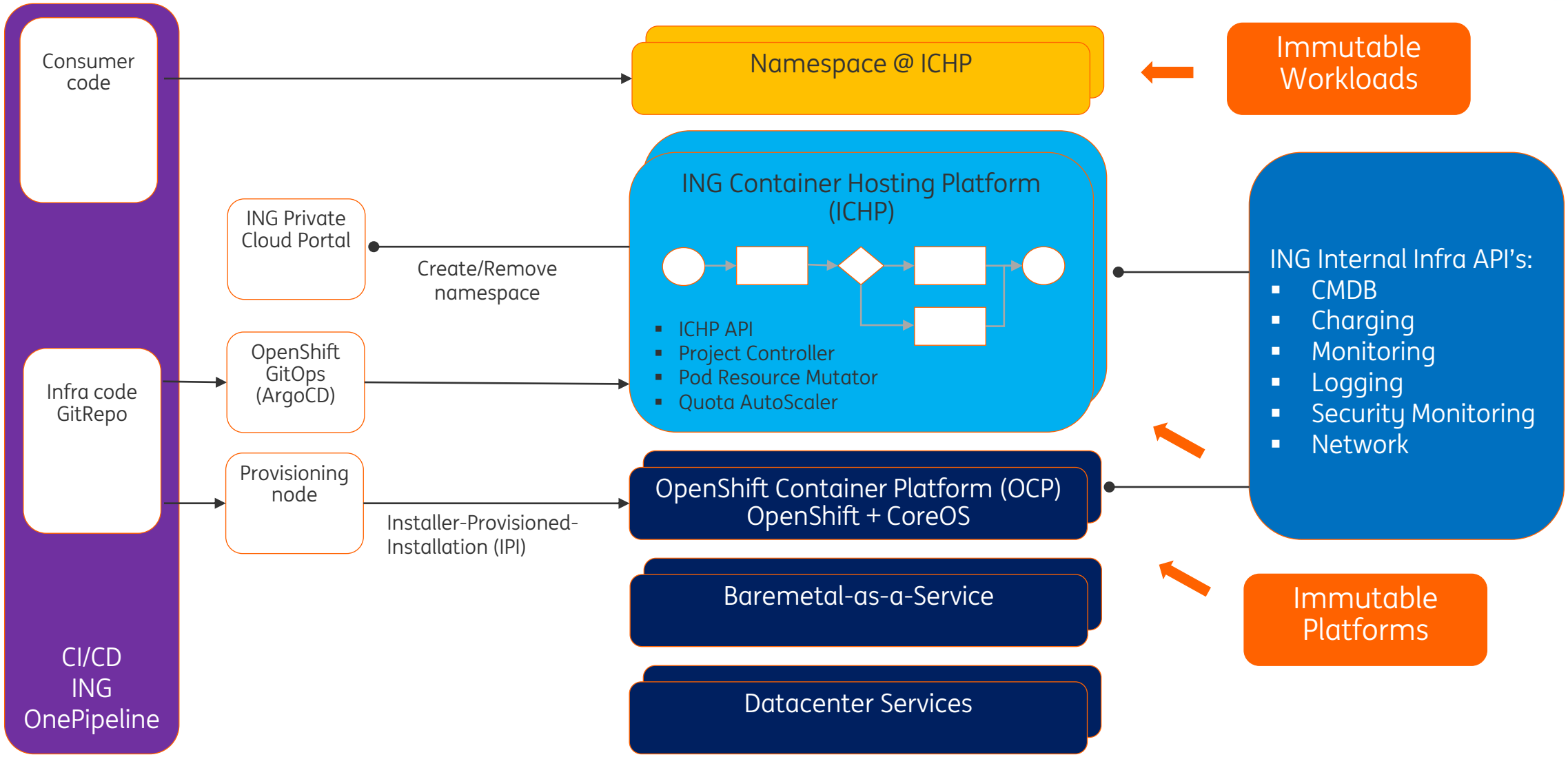
Building a NaaS on top of OpenShift 4.10 IPI stack



ING proprietary work: NaaS, which consists 36 packages, here are some of the most important

- **ICHP API**
  Orchestration over Clusters, interfacing with Private Cloud Orchestrator
- **Project Controller**
  Cluster based project creation workflow
- **Auth Delegator**
  Create cluster rolebindings based upon namespaced custom resources
- **CDaaS Controller**
  CI/CD integration
- **Image Reporter**
  Querying of ICHP hosted container-images metadata for CI/CD & Security partners
- **Pod Resource Mutator**
  To change consumer pod resources without giving namespace privileges to provider admins
- **Quota AutoScaler**
  Sets namespace quota according to deployment limits

Namespace-as-a-Service in Private Cloud interactions overview

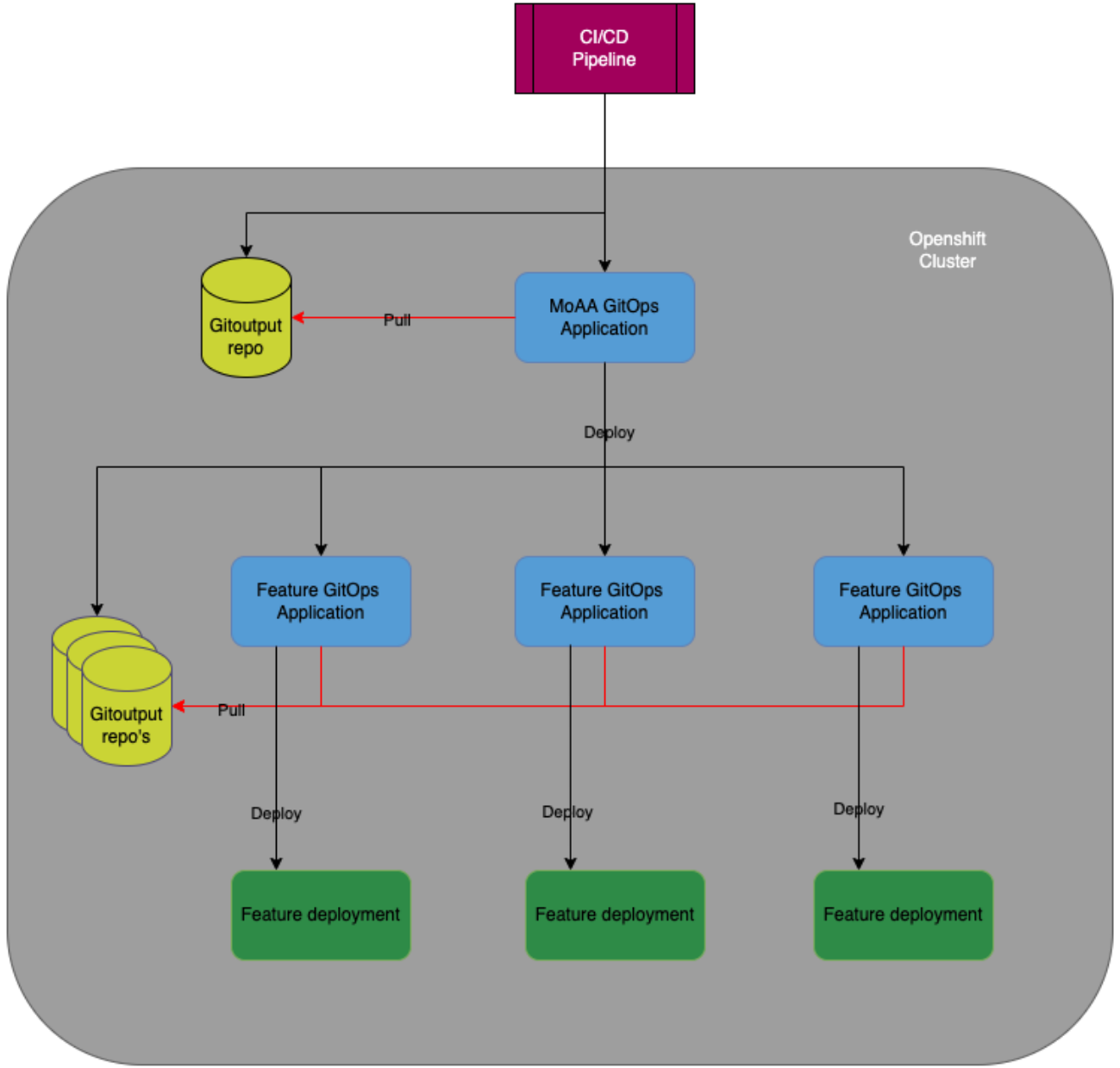


GitOps@ICHP

Standardization is essential to decrease the workload on the operations team. To prevent configuration drift we started working with openshift-gitops (ArgoCD).

1. We select the commits we wish to have in a specific release
2. An outputrepo image is build with these repositories and pushed to the local registry
3. The outputrepo is deployed within the cluster (pointing to the newer image)
4. A manifest pointing to the outputrepo in the cluster will trigger the full update

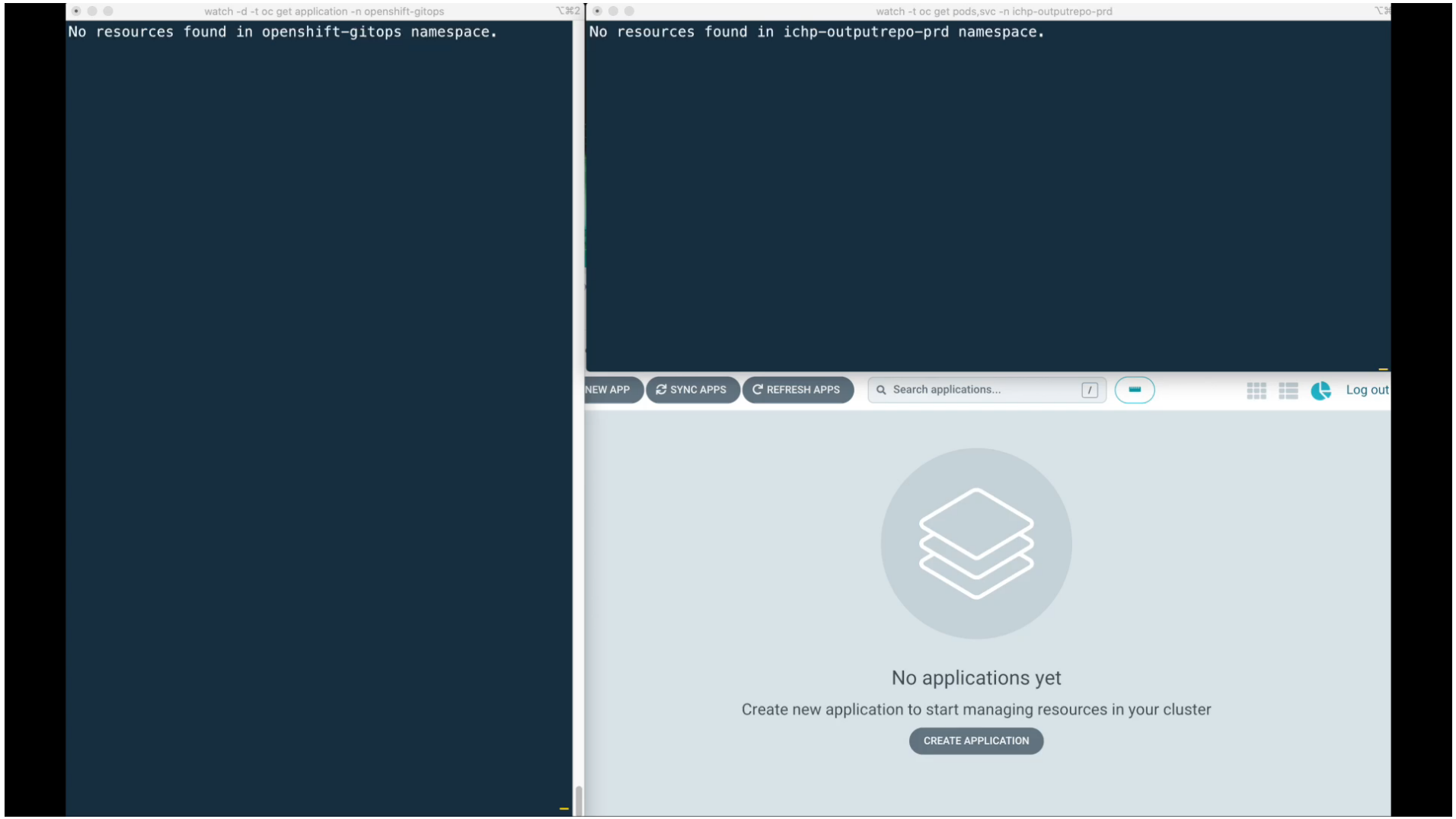
GitOps@ICHP



1. A specific version of our configuration is pushed through the CI/CD Pipeline to a cluster
2. The “cloned” outputrepo image starts and the MoAA gitops application is loaded into Openshift-Gitops
3. Openshift Gitops sees features missing and starts deploying them

GitOps@ICHP – Demo 1

- We'll show you an (accelerated) video of the Namespace-aaS being created using GitOps
- This includes all stages after the IPI install has completed until a fully operational NaaS environment



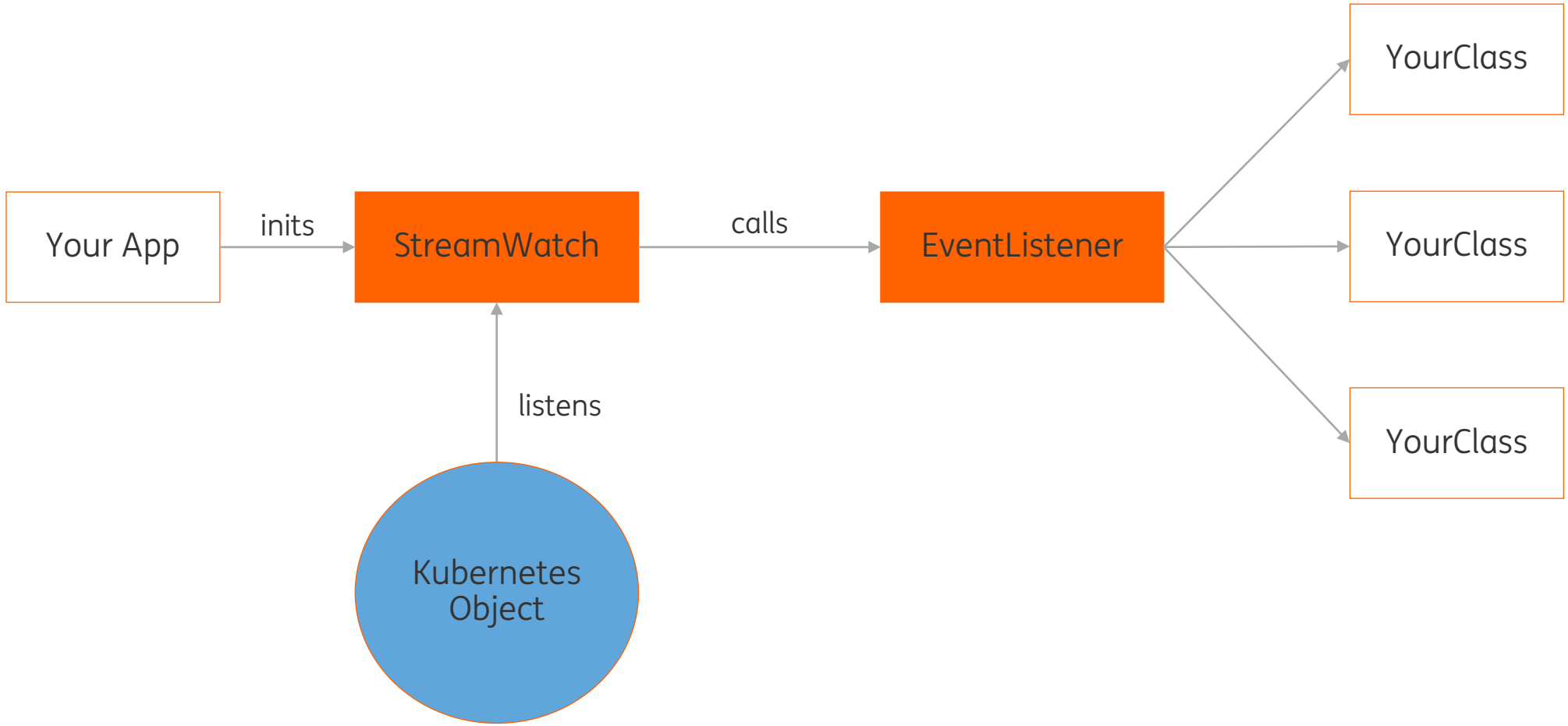
Project Controller : Namespace configuration management

The ING Project Controller is built in such a way to enable rollback and be performant

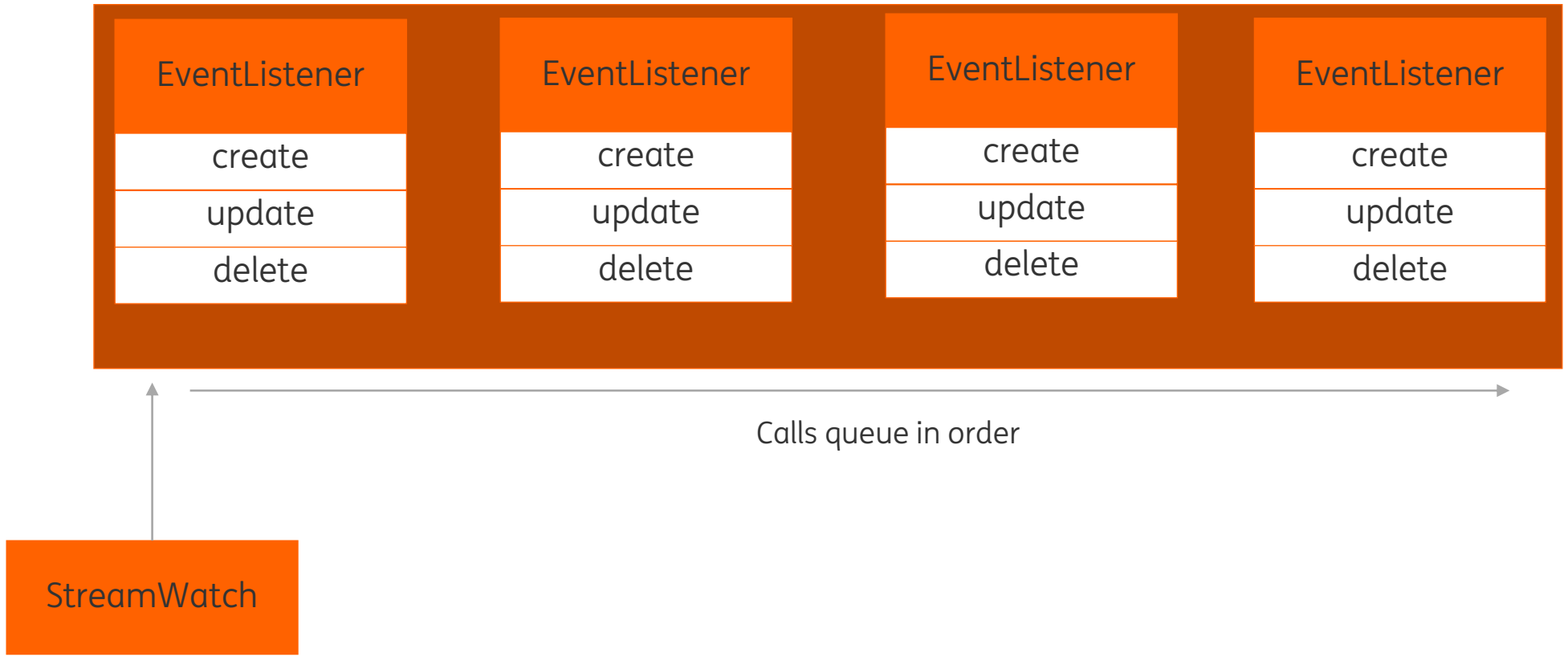
- Manages the namespaces on a cluster in a standardized way
  - Connects namespace to consumers AD Group
  - Hardens the namespace
  - Creates service connection between CI/CD (“One Pipeline”) and Cluster

- Uses ING’s Skafos Framework
  - Framework for operators in Python
  - Supports rollback incase of failure

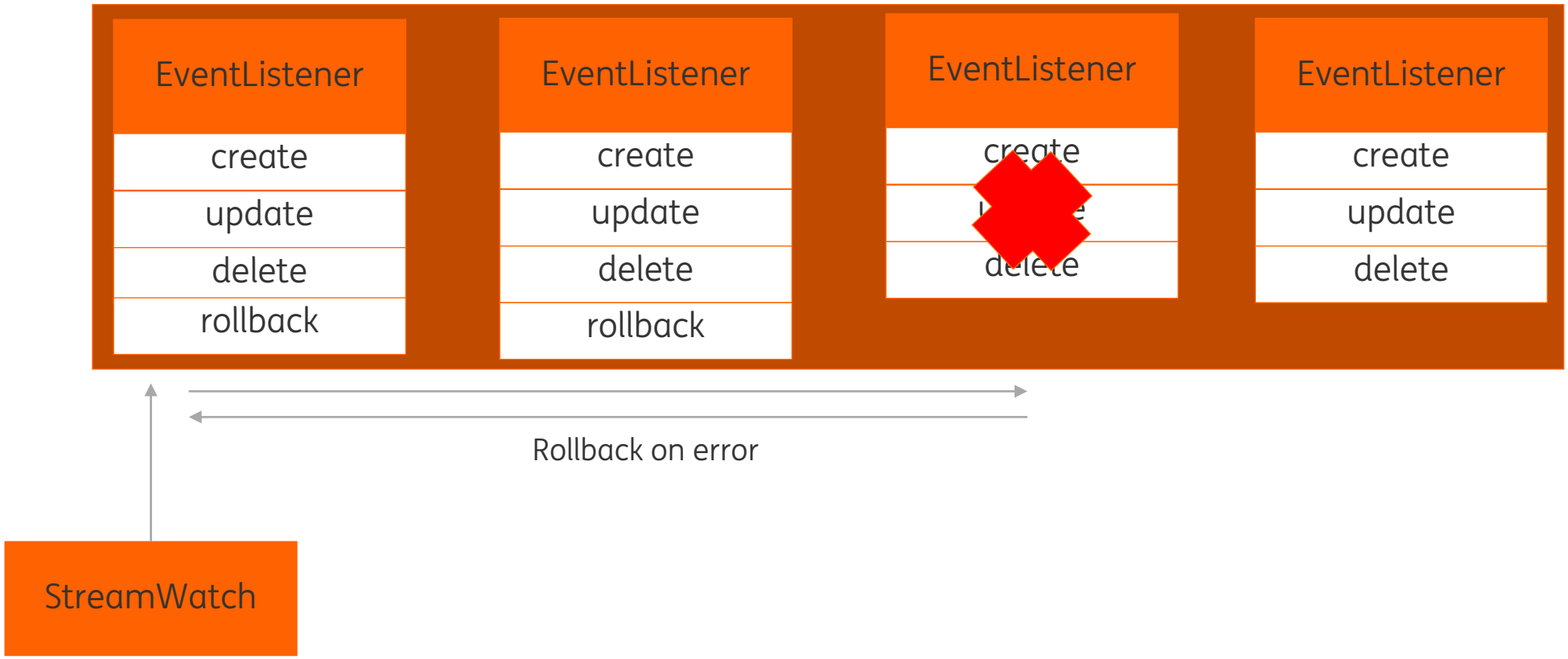
Skafos structure



Skafos structure



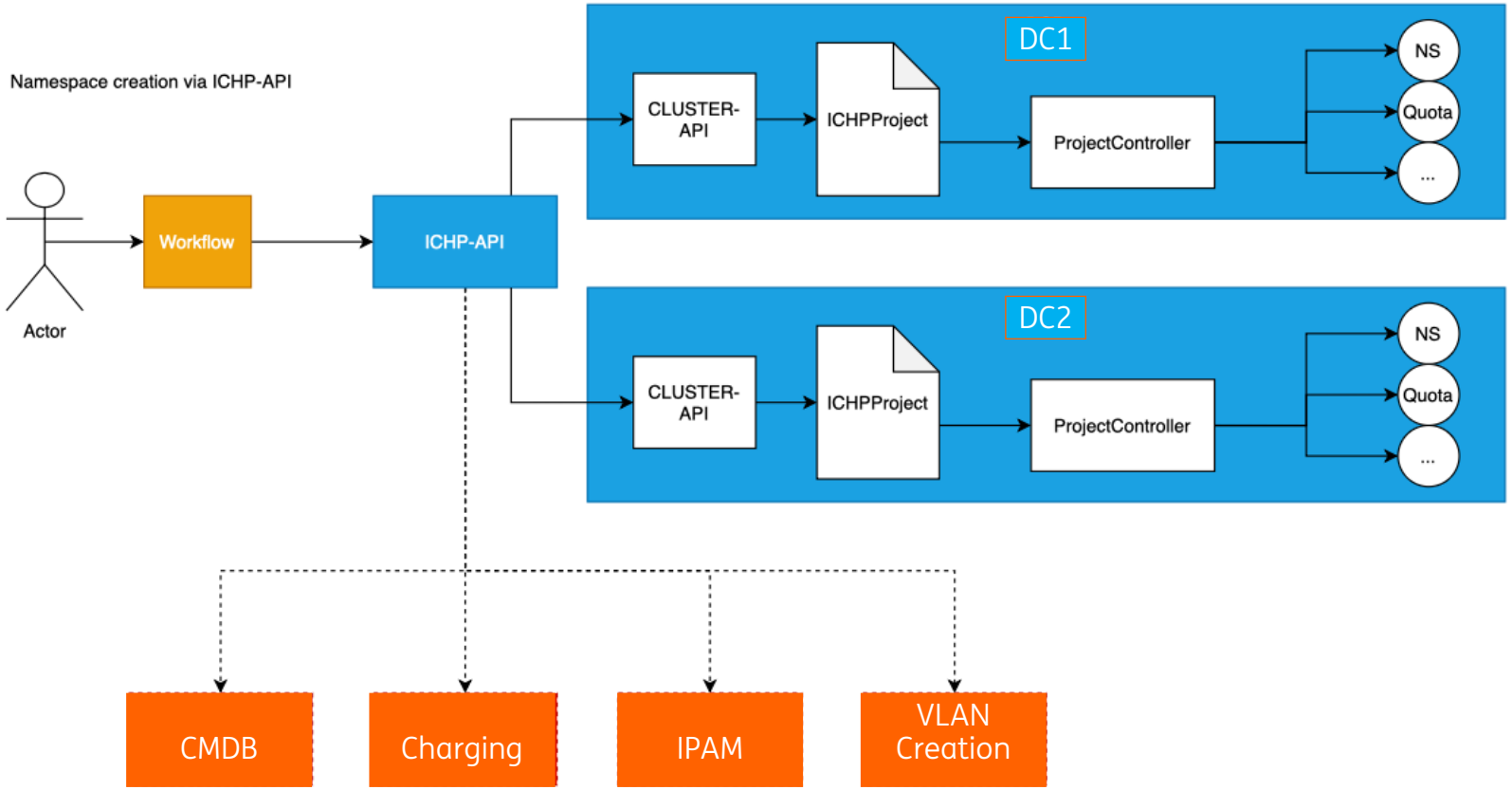
Skafos structure



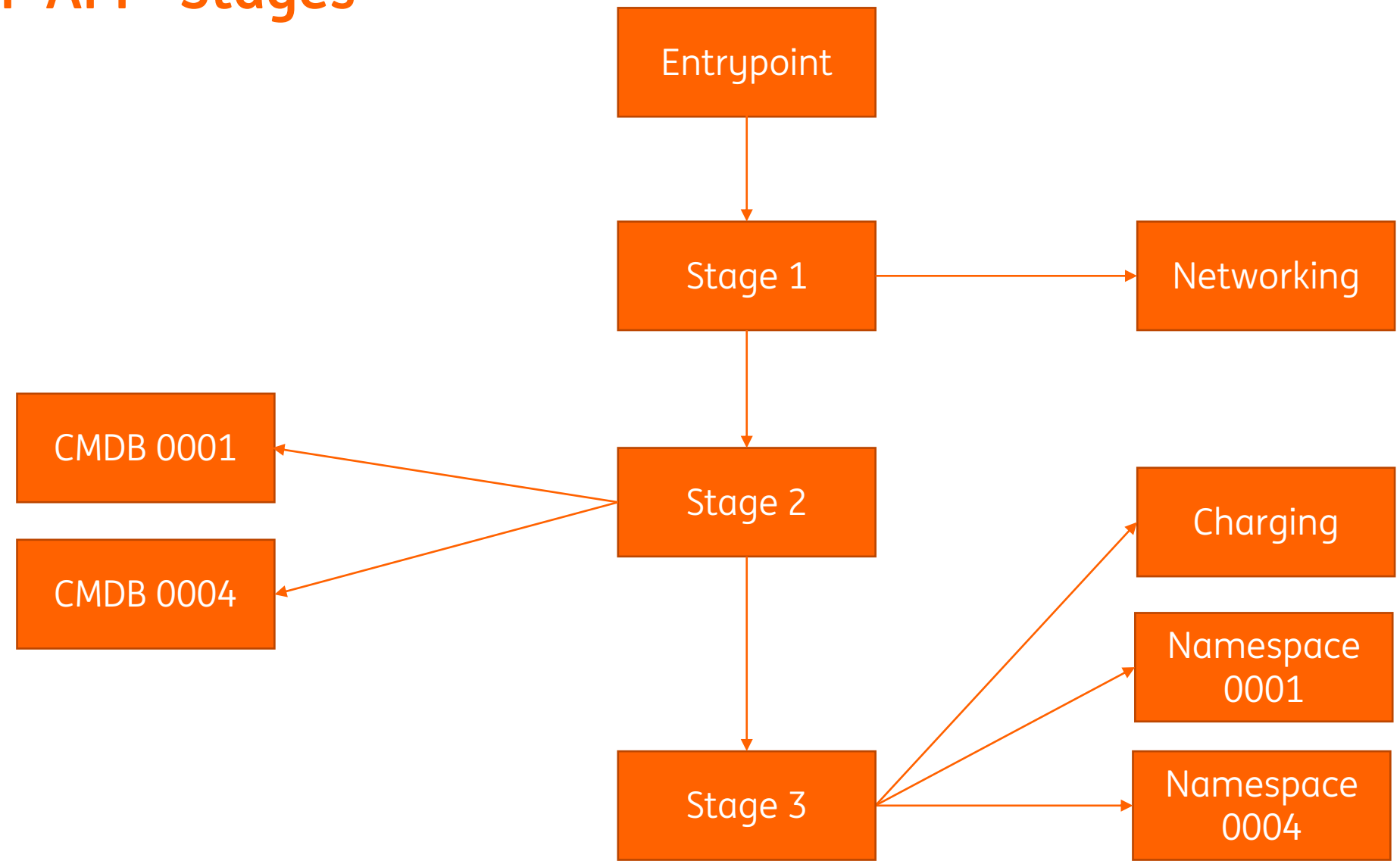
ICHP-API

Our ICHP-API allows us to manage namespaces over multiple clusters and connects to other back-end systems.

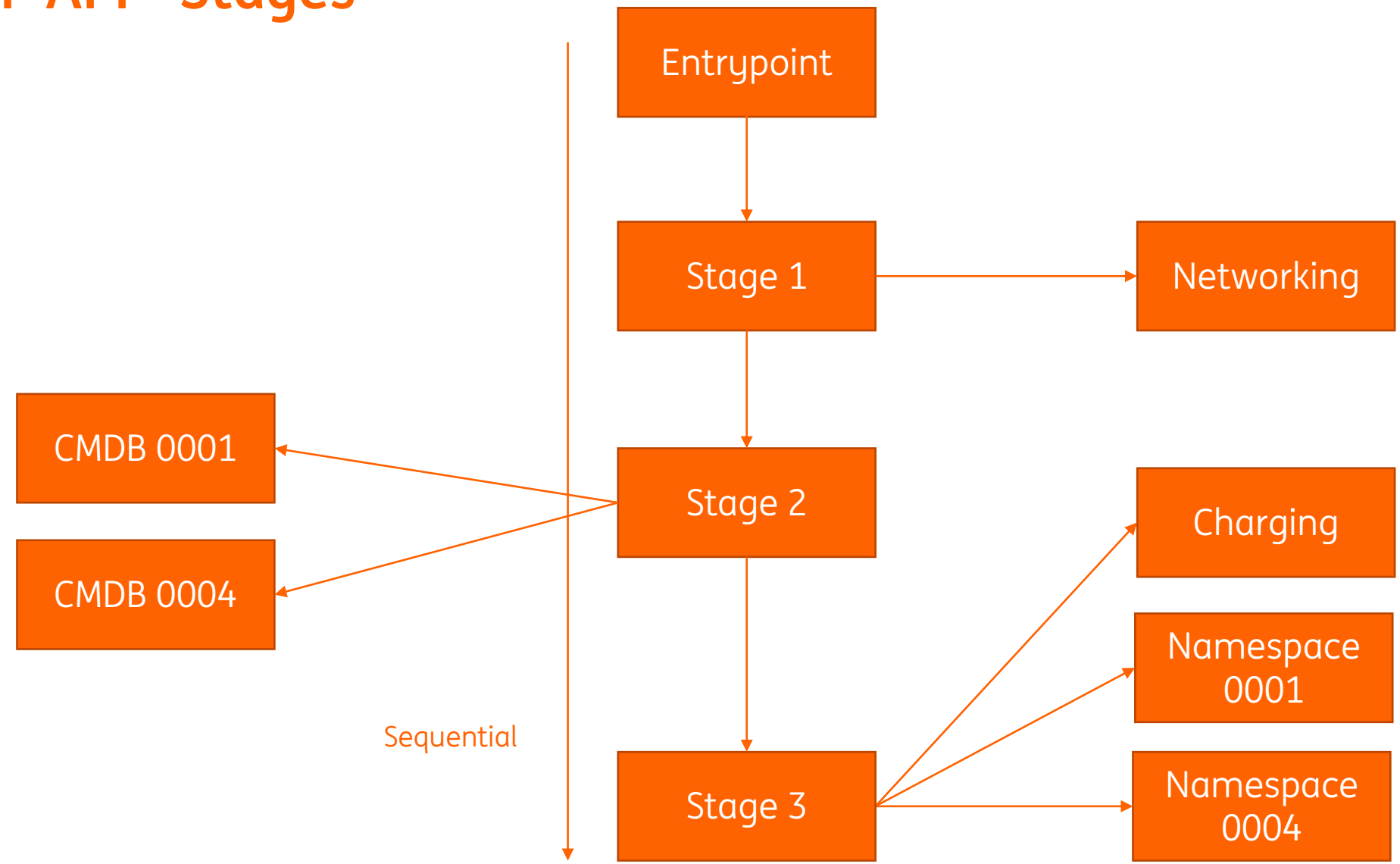
A (potential) failure will result in a full rollback of already performed actions.



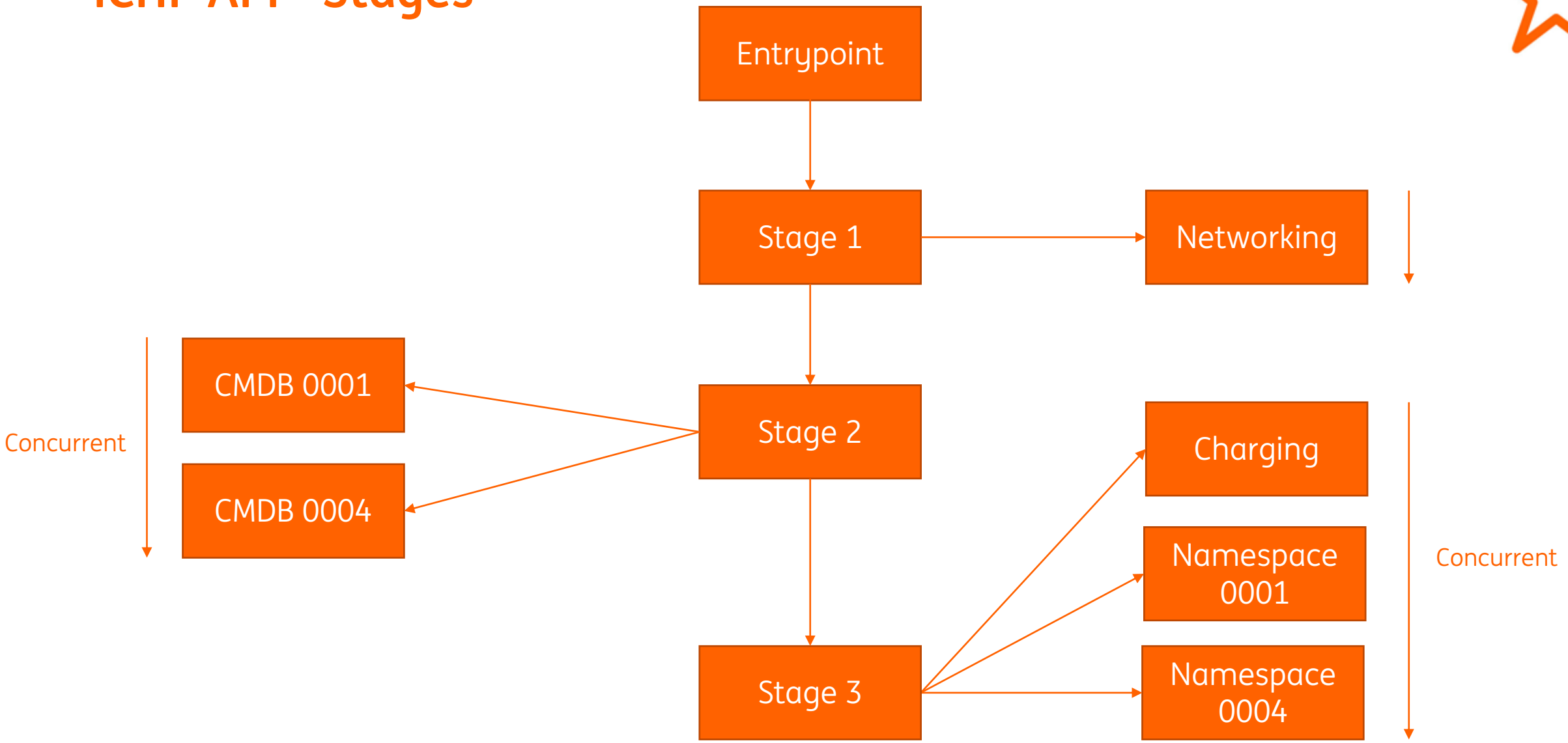
ICHP API - Stages



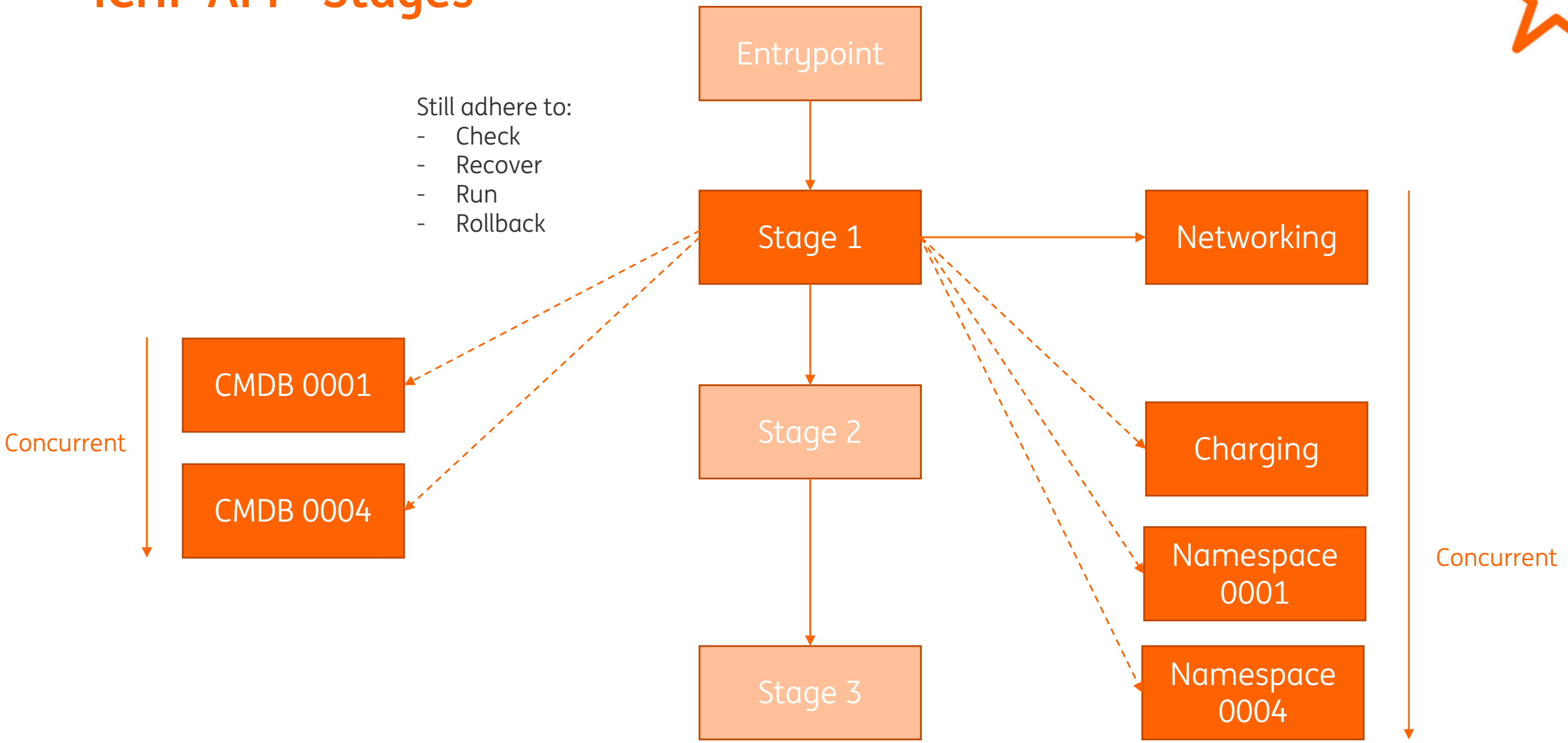
ICHP API - Stages



ICHP API - Stages

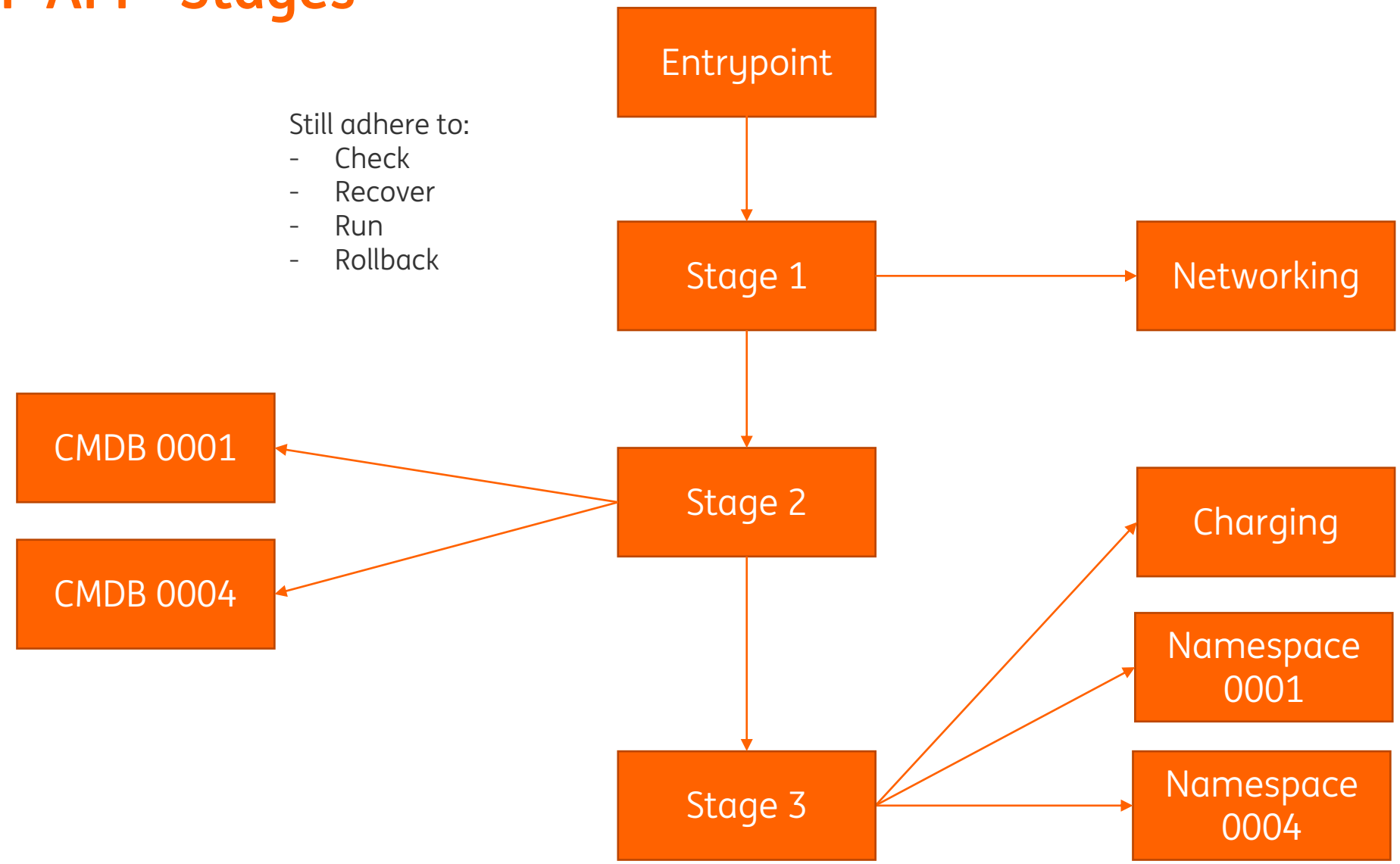


ICHP API - Stages



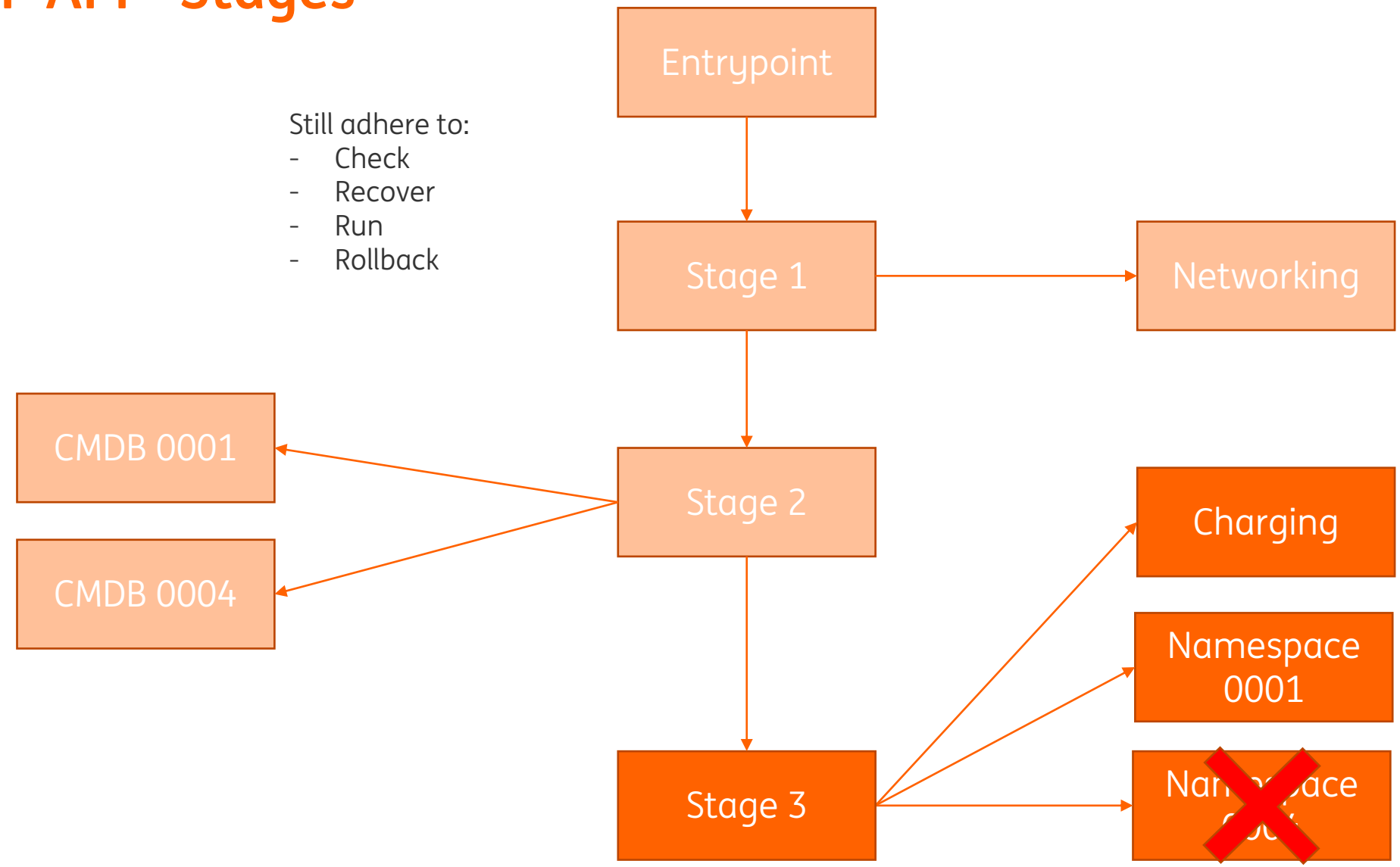
ICHP API - Stages

Still adhere to:
- Check
- Recover
- Run
- Rollback



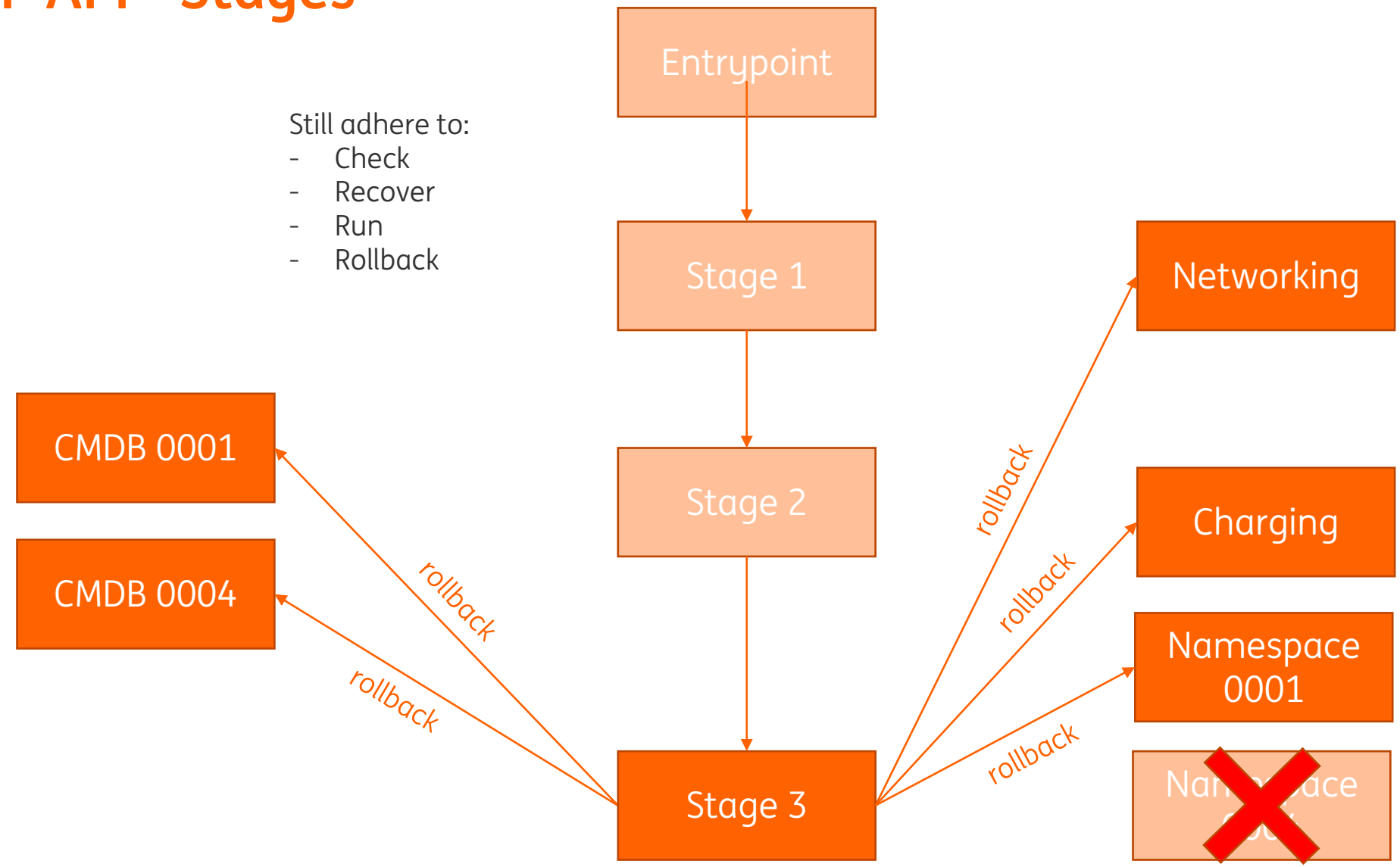
ICHP API - Stages

Still adhere to:
- Check
- Recover
- Run
- Rollback



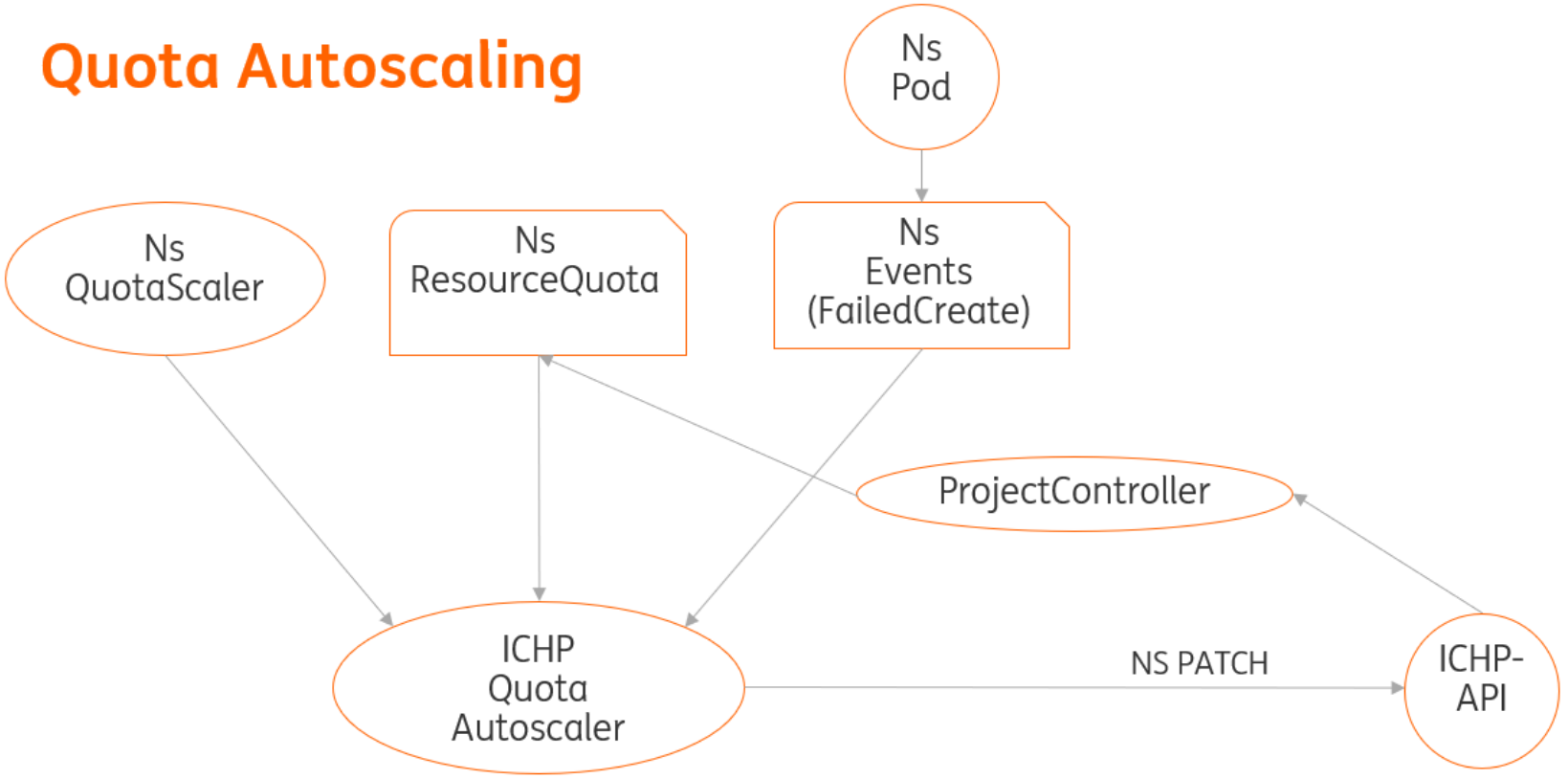
ICHP API - Stages

Still adhere to:
- Check
- Recover
- Run
- Rollback



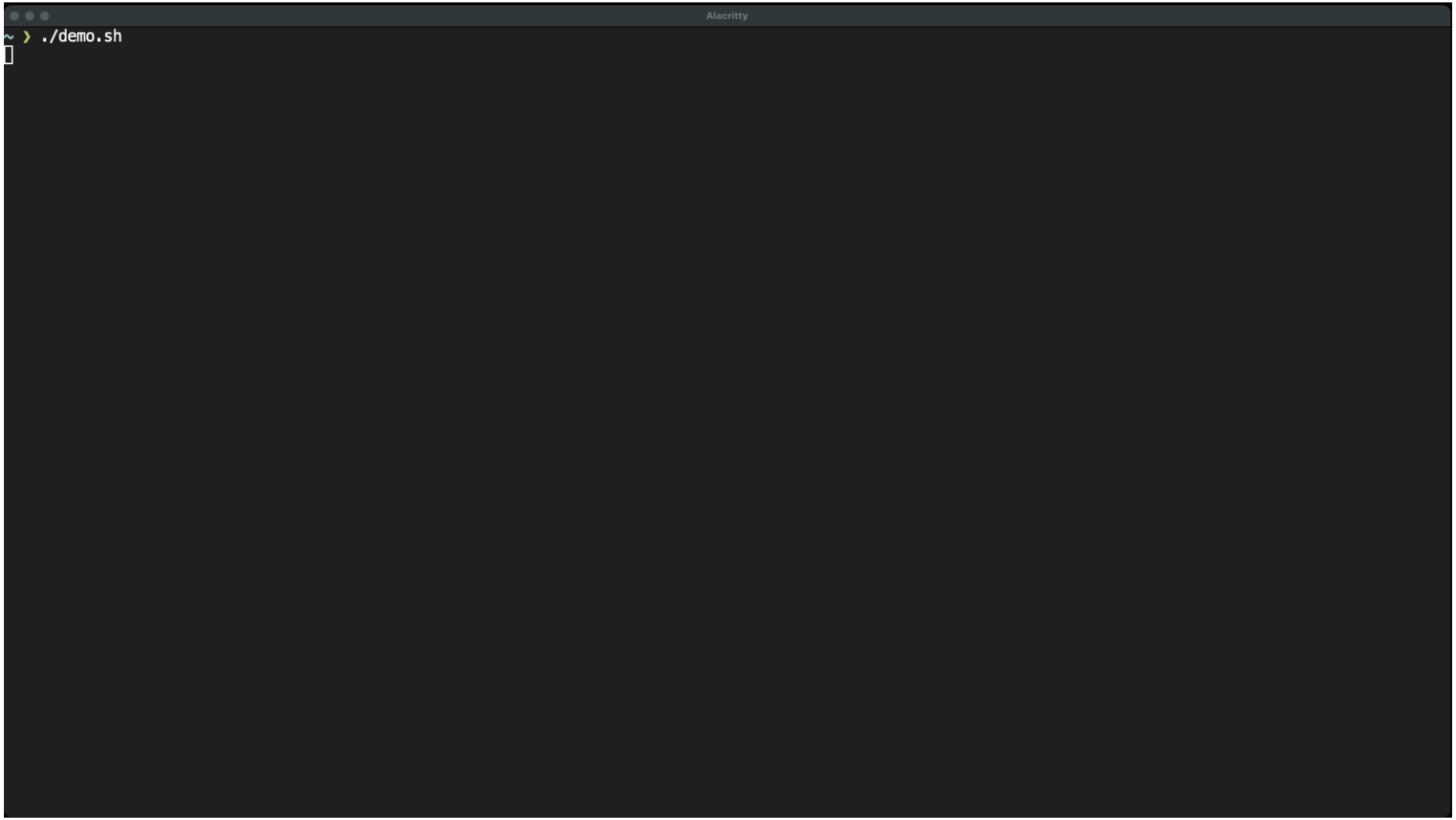
Quota Autoscaler (Typical Dev's request more then they consume...)

- Adjust resource reservation based on consumption
  - Higher density and usage on nodes and clusters
  - Flexible scaling to support temporary increased resource requirements (Upgrading)
  - Lower running cost for infra and consumer
- Managed by consumers through CR's in Openshift



Demo 2

- We'll show you an (accelerated) video of a namespace being created with all it's dependencies managed
- The result is a namespace the consumer can deploy it's workload into
- For the purpose of this demo we also will trigger the quota autoscaler



ING

Surprise

Question : Can I have the NaaS code ?

Answer : ICHP is in the (ING internal) process of Open Sourcing (under MIT licence) our first components (those we think are interesting enough) on https://github.com/ing-bank :

- ICHP Orchestration package
- Skafos : Operator Framework for Python based upon observer design pattern
- Quota AutoScaler

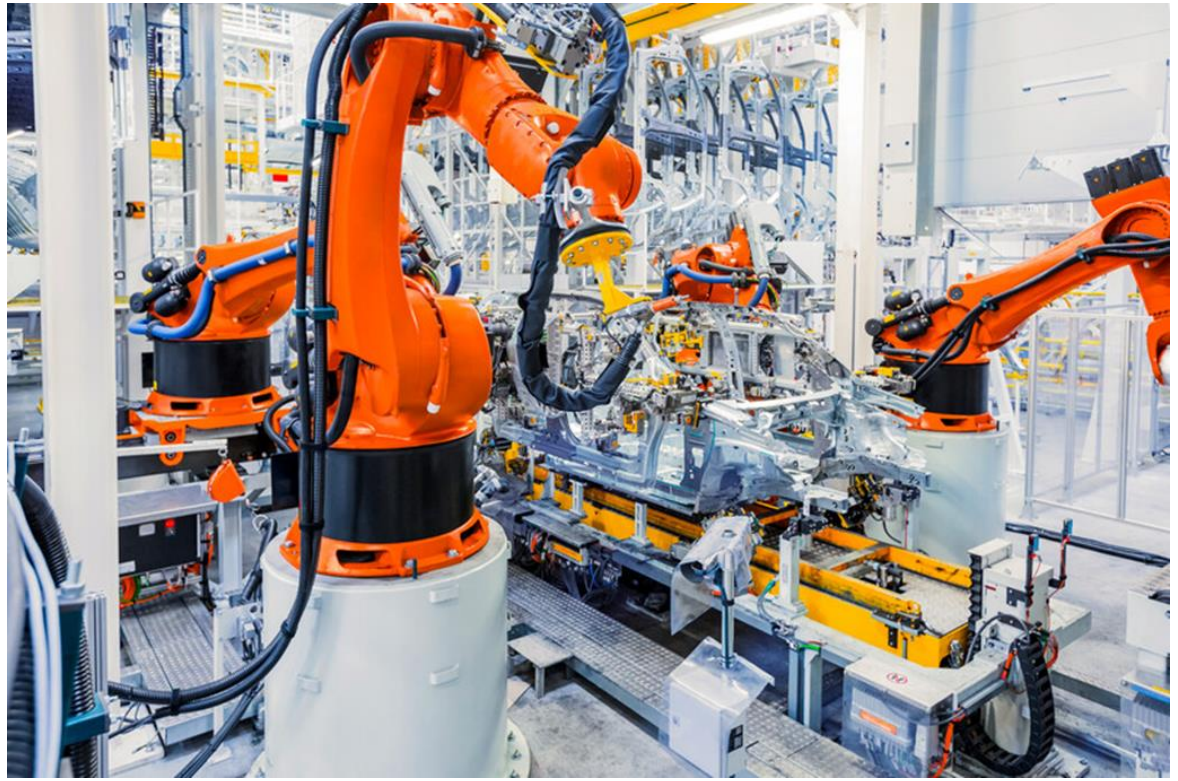
See previous slides for more details on these components.

We were aiming to complete this process before today, but those of you who are interested will just have to show a bit more patience… Stay tuned !

Why do be do this ?

- We believe in the power of Open Source, and us contributing code back is the next step in our Journey. In particular it will give our engineers the option to make a difference even outside the bank and participate in a bigger community. We do realize it will not be a free ride, and people will comment our contributions on their value (at least we hope so…)

Thank you / Questions



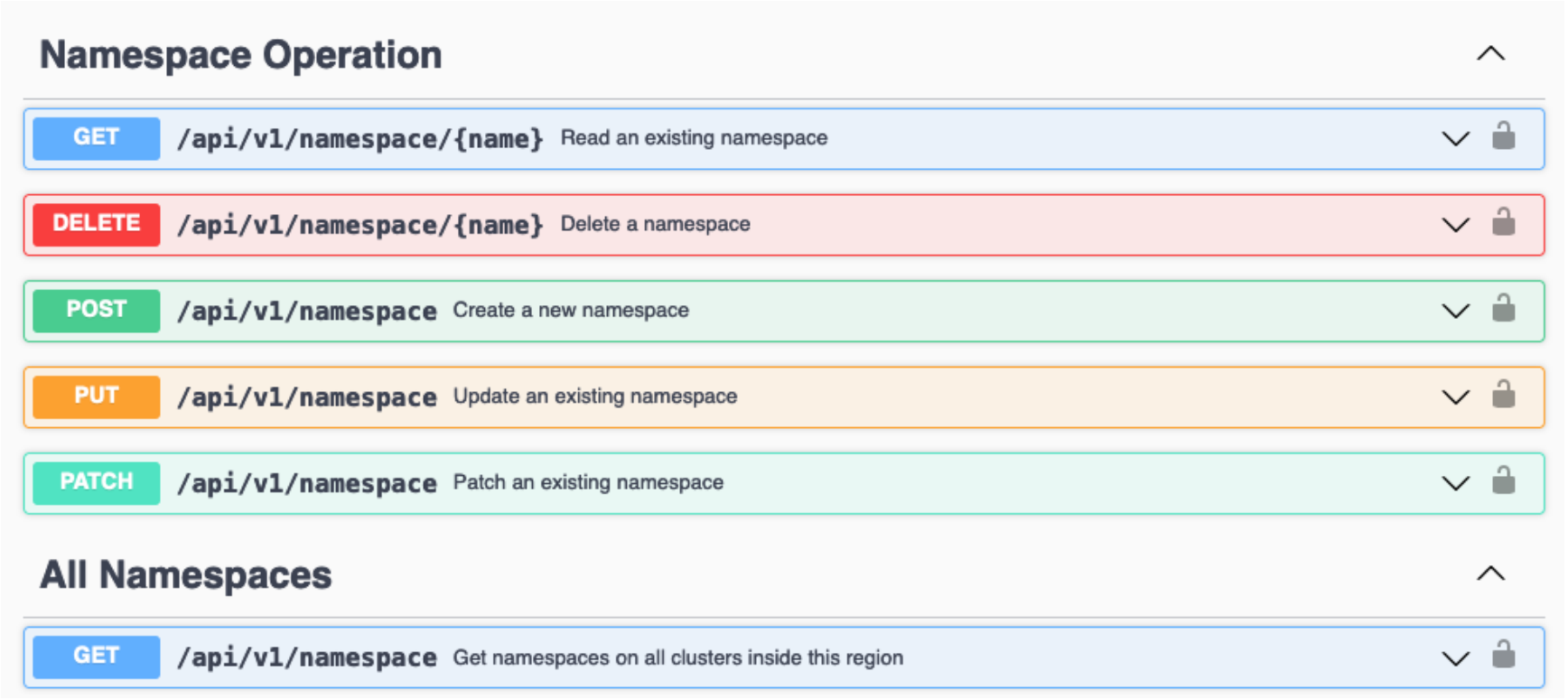
Of course we're hiring ;-)
https://www.ing.jobs/tech

ING Open Sourced code will become available at
https://github.com/ing-bank


Appendices
Additional materials for your understanding

What exactly is the ICHP-API?

- HTTP REST API that allows CRUD for namespaces



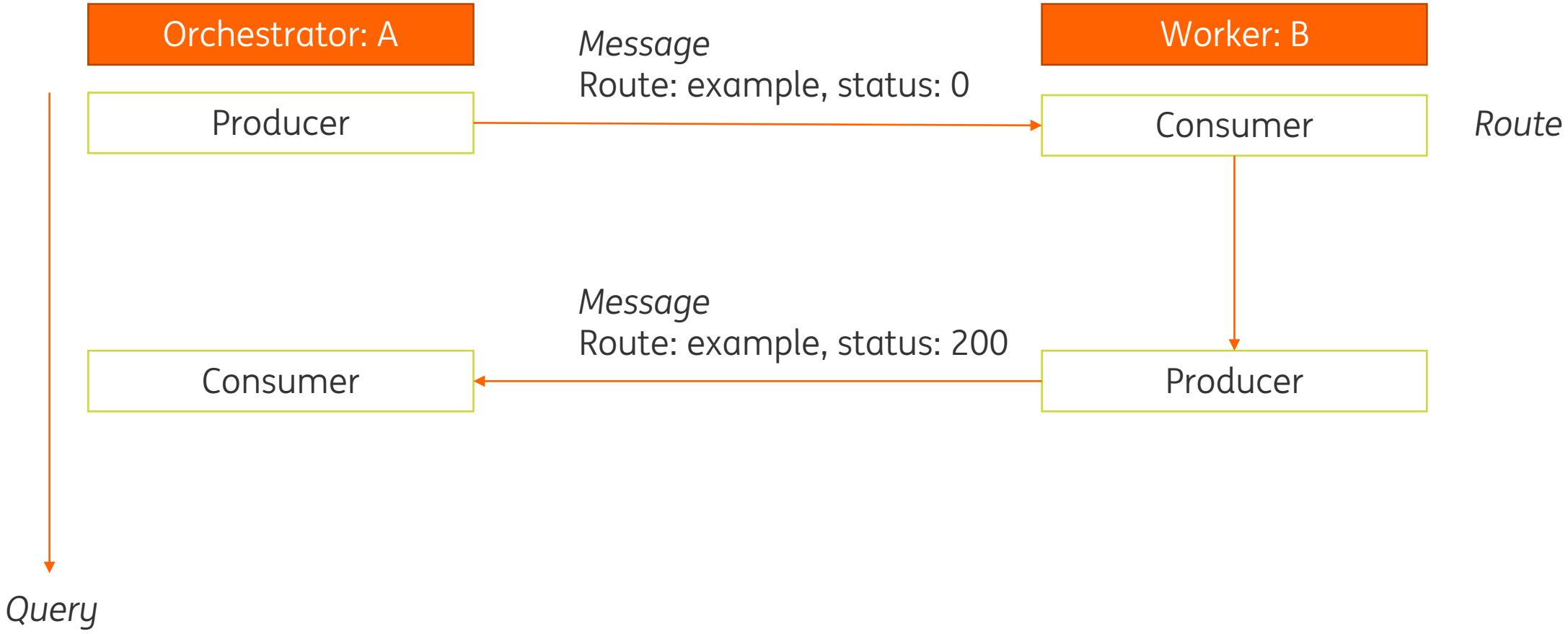
ING

ICHP API - Stages in code

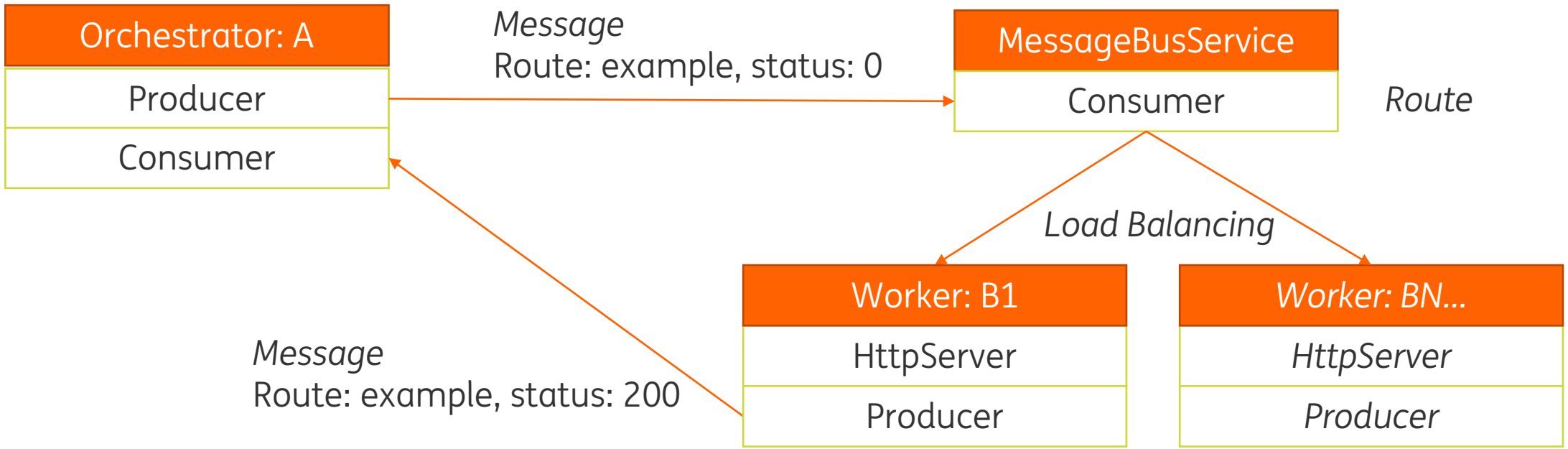
```
95      units := helpers.ClusterActionStages{
96          {
97              {Factory: MakeNetworkCreate},
98              {Factory: helpers.MakeDryRun(MakeNamespaceCreate), AllClusters: true},
99              {Factory: helpers.MakeDryRun(MakeNamespaceCmdbCreate), AllClusters: true, Skip: !shared.GlobalCo
100             {Factory: helpers.MakeDryRun(MakeNamespaceChargingCreate), Skip: !shared.GlobalConfig.Charging.E
101         },
102         {
103             {Factory: MakeNamespaceCmdbCreate, AllClusters: true, RequireClusterSpecific: true, Skip: !share
104         },
105         {
106             {Factory: MakeNamespaceCreate, AllClusters: true, RequireClusterSpecific: true},
107             {Factory: MakeNamespaceChargingCreate, Skip: !shared.GlobalConfig.Charging.Enabled},
108         },
109     }
110
111     return helpers.RunStagedClusterActionsAndReply(units, namespace, msg: "Namespace Create")
112 }
```

ING

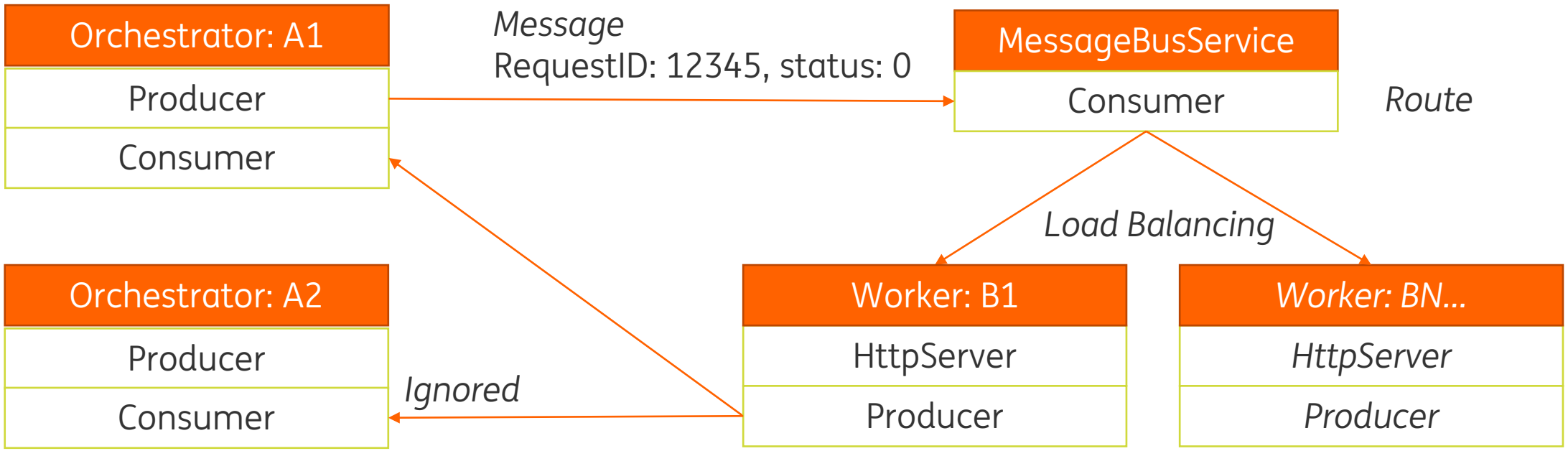
ICHP-API Messaging



ICHP-API Messaging



ICHP-API Messaging



Follow us

ing.com

@ING_News

LinkedIn.com/company/ING

SlideShare.net/ING

Medium.com/ing-blog

ingwb.com

Facebook.com/ING

YouTube.com/ING

Flickr.com/INGGroup

do your thing