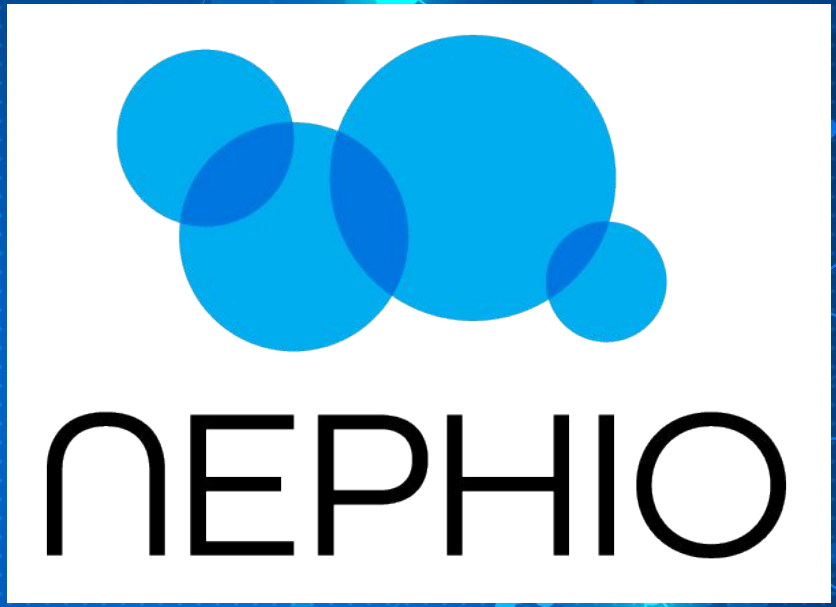
The Nephio Approach

Nephio R1 Concepts and Tutorials
Episode 3
August 2023

Prerequisites:

- Episode 1 - Series Introduction
- Episode 2 - Why Nephio?



https://nephio.org/learn

John Belamaric, Sr Staff Software Engineer, Google
Nephio SIG Automation Chair
Kubernetes SIG Architecture Co-chair

THE LINUX FOUNDATION

What do we do? Where do we start?

THE LINUX FOUNDATION

NEPHIO

Reduce Complexity

- Consolidate on a **single, unified platform for automation**
  - Across infrastructure, workloads, workload configs, vendors and deployment tiers.
- **Declarative configuration with active reconciliation** to support days one and two.
  - And distribute state (intent) across geography for resilience
- Configuration that can be **cooperatively managed** by machines and humans.
  - Machine-manipulable configuration is fundamental to automation.

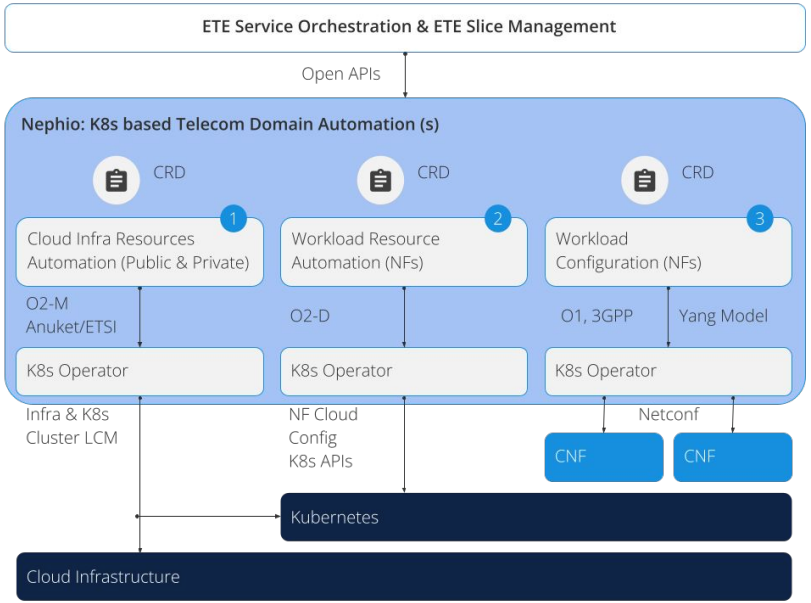
Uniform Platform for Automation

"Swimlanes"

1. Infrastructure
2. Workload (network function)
3. Workload configuration

R1 Demonstrates Some of Each

1. Cluster provisioning
2. Network function provisioning
3. NF config file generation in operator



Declarative Management

THE LINUX FOUNDATION

NEPHIO

Express intent...

I'm thirsty...
I want soda in my blue cup.

Sure! Thanks for trusting me to figure out how to get you a soda.

Declarative: User Friendly

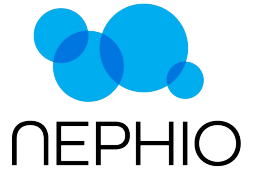
...without the toil of "how"

I'm thirsty...
Go to the kitchen
Open the fridge
Reach in the back to get the soda
Close the fridge
Grab a blue cup from the left cupboard
Pour out the soda in the cup
Bring the cup back to me

Ummm... okay thanks for telling me exactly what to do...

Imperative : Hardship to user

Kubernetes Everywhere



- **Uniform automation tooling** for topology, infra, workloads, and workload config
  - No out-of-band changes!
- Foundation for **intent-based, declarative management** with active reconciliation
- Widely adopted and understood
- Existing K8s-based point solutions in many of the layers
- Support for custom schemas and controllers
- Strong extensibility
- Rich ecosystem

What about the third principle?

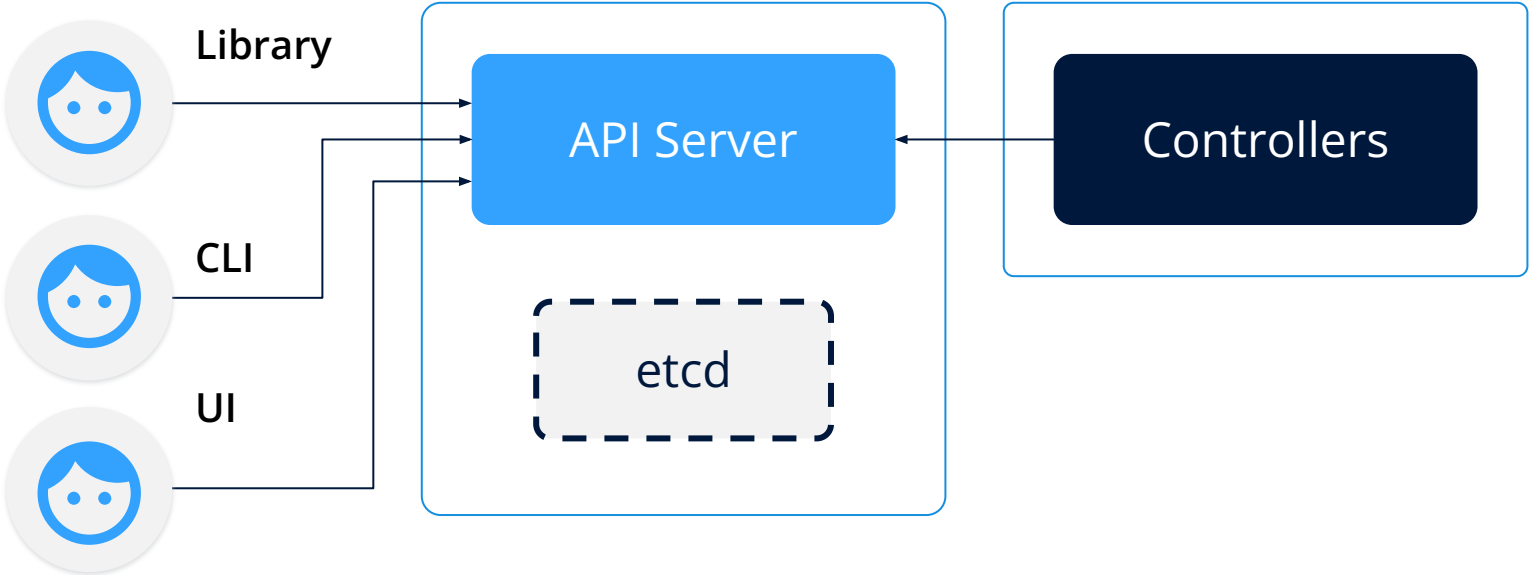
THE LINUX FOUNDATION

NEPHIO

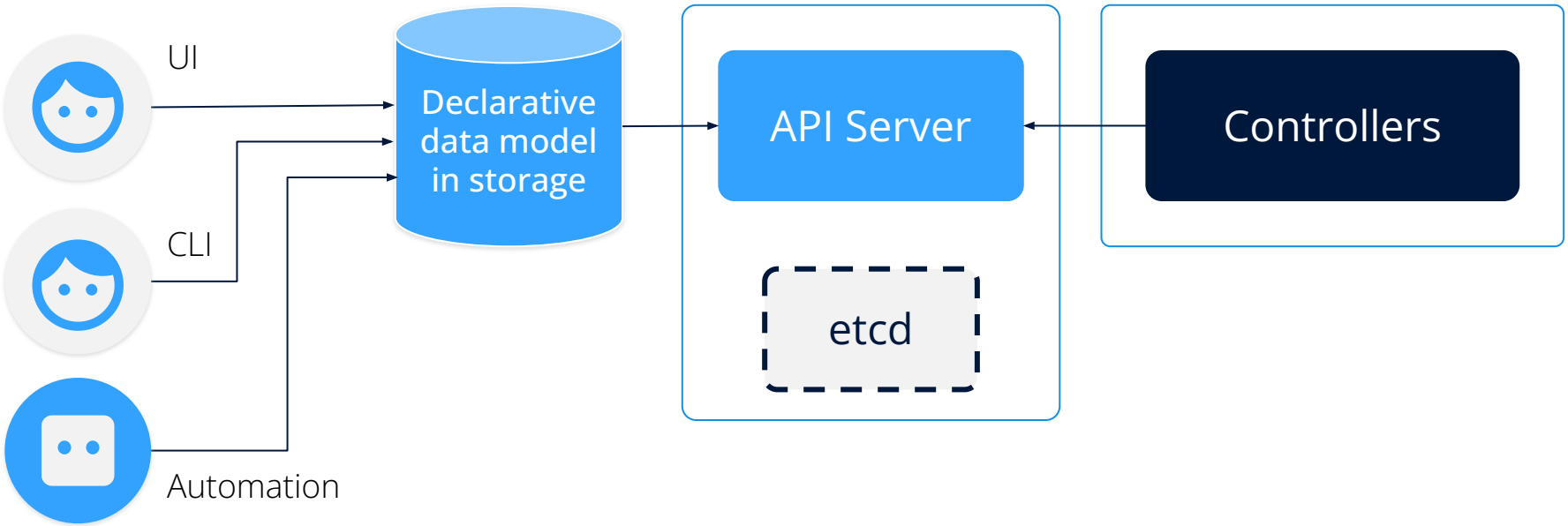
Reduce Complexity

- Consolidate on a **single, unified platform for automation**
  - Across infrastructure, workloads, workload configs, vendors and deployment tiers.
- **Declarative configuration with active reconciliation** to support days one and two.
  - And distribute state (intent) across geography for resilience
- Configuration that can be **cooperatively managed** by machines and humans.
  - Machine-manipulable configuration is fundamental to automation.

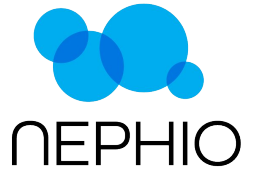
Kubernetes Desired State is Managed Through its API



What if we could Operate on Configuration in Storage?



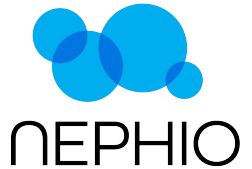
What is Configuration as Data?



Simple core principles:

1. Makes configuration data in versioned storage (git) the source of truth
2. Uses a uniform, serializable data model (KRM) to represent configuration
3. Separates code that acts on the configuration from the data
4. Clients manipulating configuration data don’t need to directly interact with storage, they operate on data via APIs

Benefits of Config-as-Data



- Machine manageable configurations
- Enables iterative, multi-actor workflows to operate and validate configurations
- Automated changes, bulk operations, and human-initiated modifications co-exist peacefully
- Automatic system validation of configuration before applying to live state
- Reusable, well-tested functions operate on configuration rather than embedding code inside the configuration