The Linux Foundation and Google Cloud Launch Nephio to Enable and Simplify Cloud Native Automation of Telecom Network Functions

New Open Source Project at the LF brings Cloud, Telecom and Network functions providers together in a Kubernetes world

San Francisco—April 12, 2021 Today, the Linux Foundation, the nonprofit organization enabling mass innovation through open source, announced the formation of project Nephio in partnership with Google Cloud and leaders across the telecommunications industry. The Linux Foundation provides a venue for continued ecosystem, developer growth and diversity, as well as collaboration across the open source ecosystems.

www.nephio.org

- New Open Source Technical Project at the Linux Foundation
- Ecosystem aligning to simplify automation of Telecom Network Functions - the kubernetes way
- 25+ Major Service Providers, Services, Infrastructure & Network Function Vendors behind this global initiative.
- Complementary to major initiatives and projects in LF Networking, LF Edge and CNCF built on Kubernetes
The Linux Foundation and Google Cloud Launch Nephio to Enable and Simplify Cloud Native Automation of Telecom Network Functions

New Open Source Project at the LF brings Cloud, Telecom and Network functions providers together in a Kubernetes world

San Francisco—April 12, 2022  Today, the Linux Foundation, the nonprofit organization enabling mass innovation through open source, announced the formation of project Nephio in partnership with Google Cloud and leaders across the telecommunications industry. The Linux Foundation provides a venue for continued ecosystem, developer growth and diversity, as well as collaboration across the open source ecosystems.
Telco challenges to Cloud native evolution

**Telecom under several transitions**

- VNF to CNF
- Private to Public Cloud
- Centralized to highly distributed network

**Telecom in need of transformation**

- Zero-touch automation with human-free control-loop
- Reduce Opex and optimize scarce edge resources
- Addressing limitations of legacy out-of-band network automation
Nepho - comes from the Greek word néphos, meaning “a cloud, mass of clouds.”

Nephio is reflection of word Nepho to support **Cloud Native Network Automation**

Pronounced as “**neh**-fee-ow”
Mission Statement

Nephio’s goal is to deliver carrier-grade, simple, open, Kubernetes-based cloud-native intent automation and common automation templates that materially simplify the deployment and management of multi-vendor cloud infrastructure and network functions across large scale edge deployments. Nephio enables faster onboarding of network functions to production including provisioning of underlying cloud infrastructure with a true cloud native approach, and reduces costs of adoption of cloud and network infrastructure.
Lack of right technology: Challenges

**Kubernetes is not enhanced enough to support Network Function automation:**

- Kubernetes enables the telecom to run network functions (NFs) as containers.
- **Kubernetes only manages basic compute/network/storage whereas NFs have demands beyond standard Kubernetes** such as existing Pod extensions: Multus, SR-IOV support, DPDK, and Node configuration: such as VLAN membership, CPU isolation, hugepages, RT kernel.
- In addition NFs and cloud-infra have more complex lifecycle management requirements.
- Enterprises rely on Kubernetes based automation for workload but Telcos rely on traditional management functions (e.g. NFVO) outside of Kubernetes to manage workloads, which is not optimal for workload reconciliation.

**Infrastructure-as-Code falls short at many ends:**

- The existing automation to deploy network functions on top of K8s mostly uses **Infrastructure-as-Code (e.g Helm)**, which has many limitations, e.g.
  - Complex templates
  - Difficult to read and test
  - Limited re-use, end up with huge lists of values that need setting
  - Not composeable - cannot handle complex sets of applications built from reusable components
  - **Non-declarative and lacking vendor neutral templates**
# Transition to Cloud Native Automation

## Past 4 years

<table>
<thead>
<tr>
<th>MANO (NFVO/VNFM)</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Non-Container Era**

## Today

<table>
<thead>
<tr>
<th>MANO (NFVO/VNFM/CNFM)</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Traditional methods:** K8s with out-of-band automation

## Nephio

<table>
<thead>
<tr>
<th>K8s-based Cloud Native Automation with KRM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
<th>VM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CaaS/Infra Mgmt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Public**

**Private**

**Edge**

**True Cloud-Native:** K8s automation all the way down to Cloud infrastructure & NFs
Benefits of using K8s & KRM/CaD

Benefits of Kubernetes based automation

- **Kubernetes is a strong, proven framework for declarative management.** Operational team better understands the Kubernetes based automation than proprietary out-of-band automation.
- **Kubernetes has existing methodology such as CRDs to define fully declarative configuration and operators to support lifecycle of the workloads.**
- Kubernetes inherently provides a) clear declarative intent of the workload b) not fire and forget instead active reconciliation (continuously cares and manages the state) c) self-healing d) actively fixes configuration drift

Benefits of Kubernetes Resource Model (KRM) and Configuration as Data (CaD)

- Widely adopted in the enterprise world and telecom community can largely benefit from the same principles
- Replacement for Helm, reducing the complexity with declarative intent based schema
- Reusable, well-tested functions operate on configuration rather than embedding code inside the configuration
- **Automated changes, bulk operations, and human-initiated modifications co-exist peacefully**
- Automatic system validation of configuration before CRUD operations
- Not just API-driven, but API-centric
Three aspects for optimizing automation

**Intent-based automation**
- Simplified configuration to user e.g. Deploy 5G UPF with X capacity at Y location and do Z when this event occurs

**Declarative configuration**
- To address day 0, 1 and 2 configurations, rainy day scenarios, intelligent auto scaling control-loops, and full life-cycle support

**Non-complex: Cloud-Native automation**
- Simplified, unified cloud native management (Kubernetes) in every tier
- **Extend base Kubernetes with Infrastructure CRDs and Operators**
  - Declarative expression of ALL infrastructure requirements for NFs
  - Beyond the Pod, to Node
  - Beyond the Node, to ToR
- **Deploy a function anywhere**
  - No out-of-band infrastructure configuration
Different layers of configuration & existing standards & Nephio Focus

**Scope of Nephio community**

Nephio focuses on extending K8s to support unified automation with:

- K8s based CRDs and operators for each Public and private cloud Infrastructure automation.
- The workload cloud resource automation (i.e. CRUD operations of K8 Cluster, Network Functions deployment on top of the cloud, and NF infrastructure configuration such as SR-IOV)
- Workload configuration (i.e. NF level configuration)

Service Orchestration layer is not in scope for this community
Nephio Functional Building blocks

Orchestration Cluster

User Interaction Layer (APIs/GUI/CLI) with KRM / CaD Templates

Intent Design
- **Intent Design Studio**
  - Package Authoring, Publishing, Catalog
- **Intent Specialization**
  - Package Instantiation, Customization
- **Intent Validation**
  - Policy, Resource, Consistency Checks

Intent Deployment
- **Cluster Selection**
  - Package Cluster Placement, Specialization
- **Cross-Cluster Dependencies**
  - Package Delivery Order, Status
- **Intent Delivery**
  - Package Distribution, Progressive Rollout

Control Loop
- **Automated Intent Revisions**
- **Metric Aggregation**
- **Status Aggregation**

Cloud Mgmt Cluster

Intent Actuation
- **Local Policy Validation**
- **Package Ingestion**

Regional Clusters

Cross-Resource Dependencies

Edge Clusters

Intent Reconciliation

Resource Actuation

---

KRM - Kubernetes Resource model
CaD - Configuration as Data
Google’s software contribution to Nephio

![Diagram of Nephio's architecture]

- **Master Service Orchestration (MSO)/ETE Slice**
- **GUI Server**
- **Kubernetes API**
- **Controller operator(s)**
- **Package Mgmt API (Porch/kpt)**
- **Optional CI/CD (git)**
- **Artifact Registry (git)**
- **Package Ingestion (ConfigSync)**
- **Cloud K8s Operators**
- **Cloud API Service**
- **Any Cloud Infrastructure**
- **API Server**
- **Infra Control-loop controller**
- **Metrics Service**
- **Adapters**
- **Package Ingestion (ConfigSync)**
- **Kubernetes API**
- **K8s Resources, CRs**
- **Cluster Specific Indent K8s Resources, CRs**
- **CNF**
- **Open APIs**
- **High-Level User Indent OSS and Vendor CRDs**
- **Google Cloud CRDs & Samples 2H2022**
- **Google SW Modules Open Sourced**
- **Google Open Source (2H2022)**
- **Existing Open Source or Cloud**
- **Nephio community development***

*Reference architecture to evolve in the community
External: Service orchestration layer
- Accepts user requirements
- Composes functions and supports end-to-end n/w slicing configuration
- Pushes intent to domain orchestration layer
- PNF orchestration

Nephio: Domain orchestration layer
- Accepts service composition
- Calculates domain and cluster-specifics
- Pushes Kubernetes manifests

Nephio: Infrastructure orchestration layer
- Applies per-cluster Kubernetes manifests
- Actuates infrastructure resources
- Results in running network functions
Nephio: Proposed working structure

- **Linux Foundation**
- **Nephio Board (after 1st year)**
- **Nephio TSC**

**SIG 1: Network Architecture**
Specifications and Requirements

**SIG 2: Automation**
CRDs, Operators, and Related Tooling

**SIG 3: Release**
Reference Implementation, Packaging, Installation
Project resources

- Website - [https://nephio.org/](https://nephio.org/) , [https://nephio.org/about/](https://nephio.org/about/)
- Blog Postings - [https://nephio.org/blog/](https://nephio.org/blog/)
- Project Github - [https://github.com/nephio-project](https://github.com/nephio-project) (Please note “nephio-project is right one”)
- Project email distro
  - nephio-tsc@lists.nephio.org (for TSC members)
  - nephio-dev@lists.nephio.org (for all)

Create an Account:

Follow LF Documentation at: [https://docs.linuxfoundation.org/lfx/sso/create-an-account](https://docs.linuxfoundation.org/lfx/sso/create-an-account)

Please subscribe to the Nephio developers mailing list. You can do that by sending an email to this address: nephio-dev+subscribe@lists.nephio.org You will receive an auto reply requesting subscription validation.

To join as Supporter:

Again, there are no documents to sign or fees to join, just the form needs to be filled out with the requested information. You can fill out this Linux Foundation's [simple form](https://forms.gle/Q4pXJdTykYfgziaz9) - Contact mwoster@linuxfoundation.org for more information
Thank you!