TeraFlow SDN by ETSI

Use cases and architecture for TeraFlowSDN

Oscar Gonzalez de Dios (Telefonica)

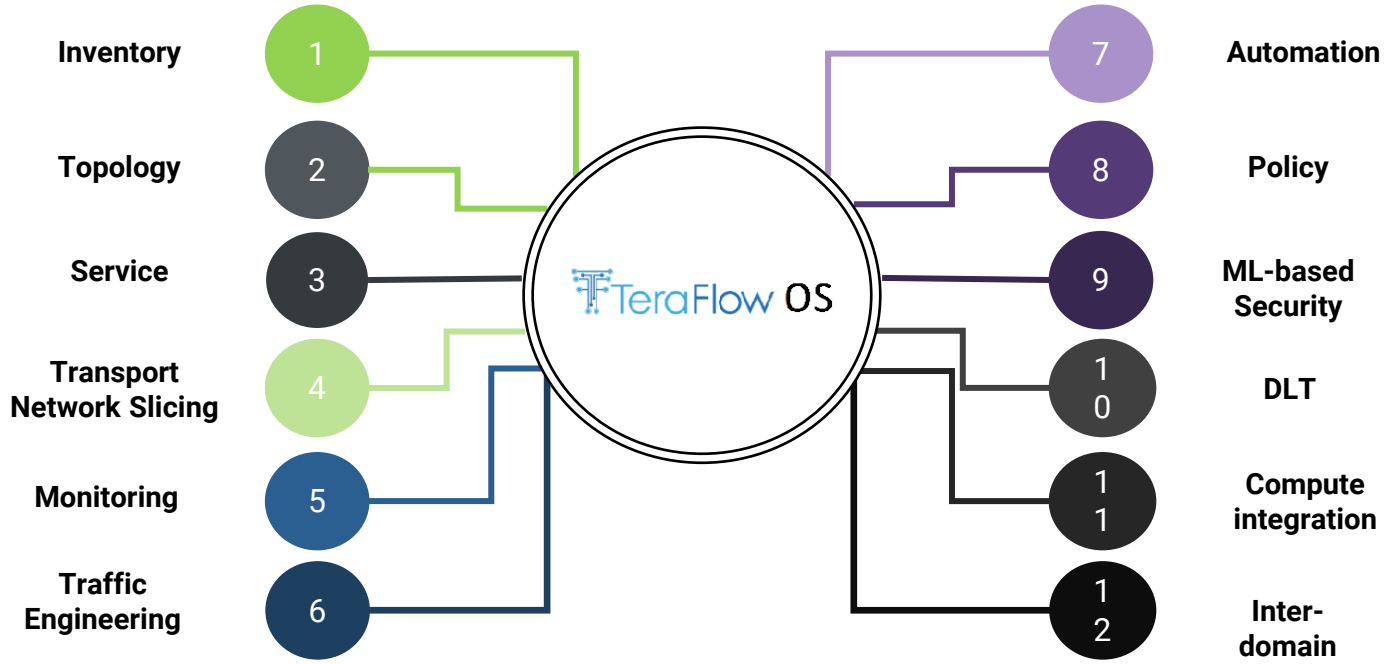
NGON & DCI World 2022

Which set of use cases is Teraflow covering?

These use cases refer to **multiple network technologies** covering: IP, **Optical**, and Microwave.

The use cases are aimed at **supporting** the network operation.

The use cases are aligned with **Telecom Infra Project MUST initiative**



Scenarios

Autonomous network B5G

Carrier grade. Industry validated

Inventory

Topology

Service

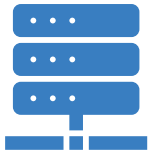
Transport network slicing

Traffic Engineering

Automation

Policy

Compute



Inter-domain

Cloud-scale, Multi-domain

Inventory

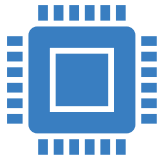
Topology

Service

Transport network slicing

DLT

Inter-domain



CyberSecurity

Security, attack detection/mitigation

Topology

Service

Monitoring

ML-based security

Standard Based Aproach

TeraFlow

Impact in Standards:

TELECOM INFRA PROJECT

- TIP MUST (focus on interfaces)
  - MUST IP – SDN Controller SBI / Router NBI Technical Requirements V1
  - MUST IP SDN Controller NBI Technical Requirements
  - MUST IP SDN Controller SBI Requirements
  - MUST Optical SDN Controller NBI Requirements
- ONF
  - Transport API 2.3
- IETF/IRTF
  - A Layer 2/3 VPN Common YANG Model
  - A Layer 3 VPN Network YANG Model
  - IETF Network Slices
- Openconfig

ONF Open Networking Foundation

IETF

ETSI

OPENCONFIG

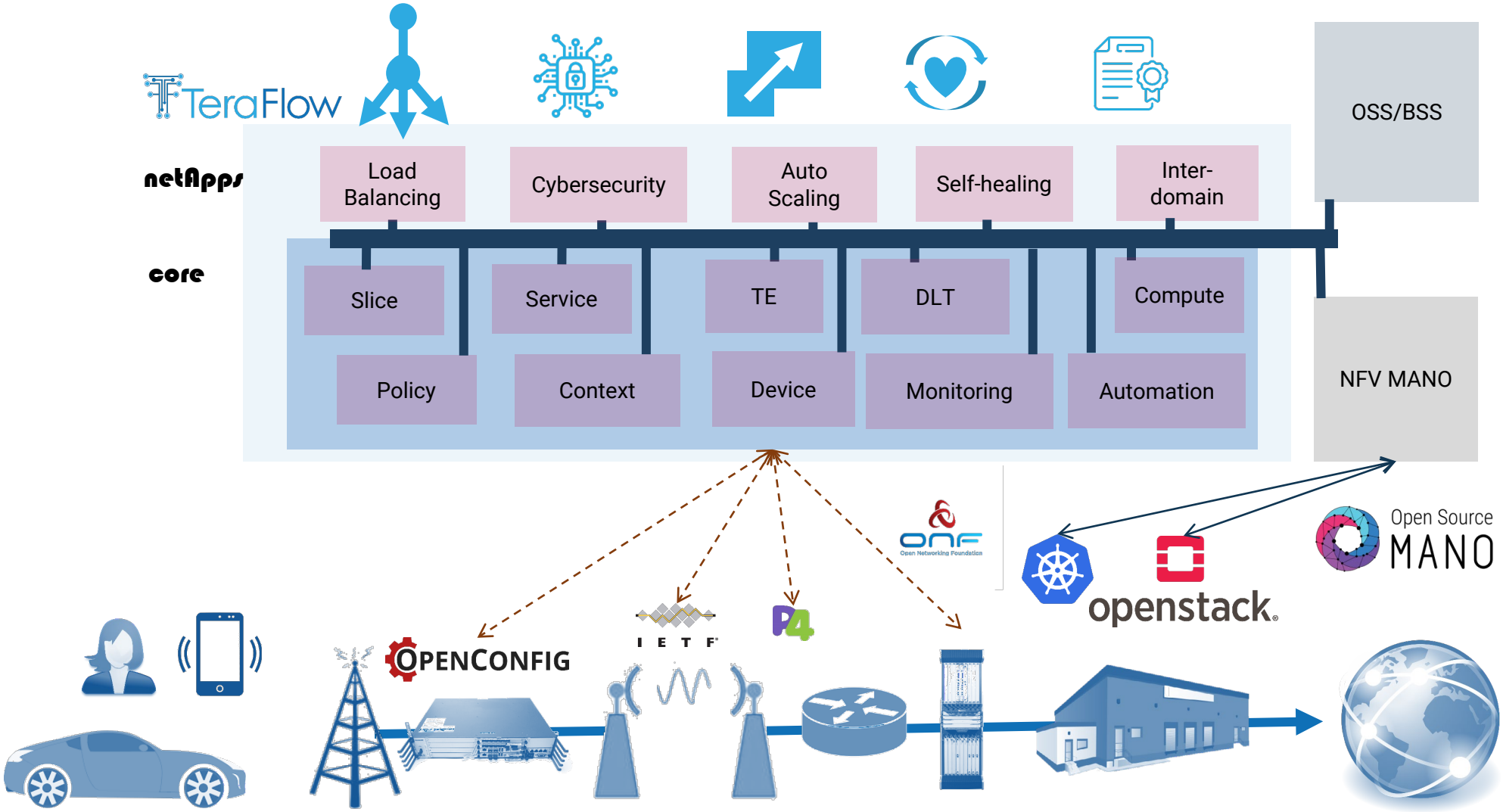
Teraflow OS Architecture

The TeraFlow OS is a cloud native SDN controller that is composed of multiple **micro-services**. Micro-services interact with each other using a common integration fabric.

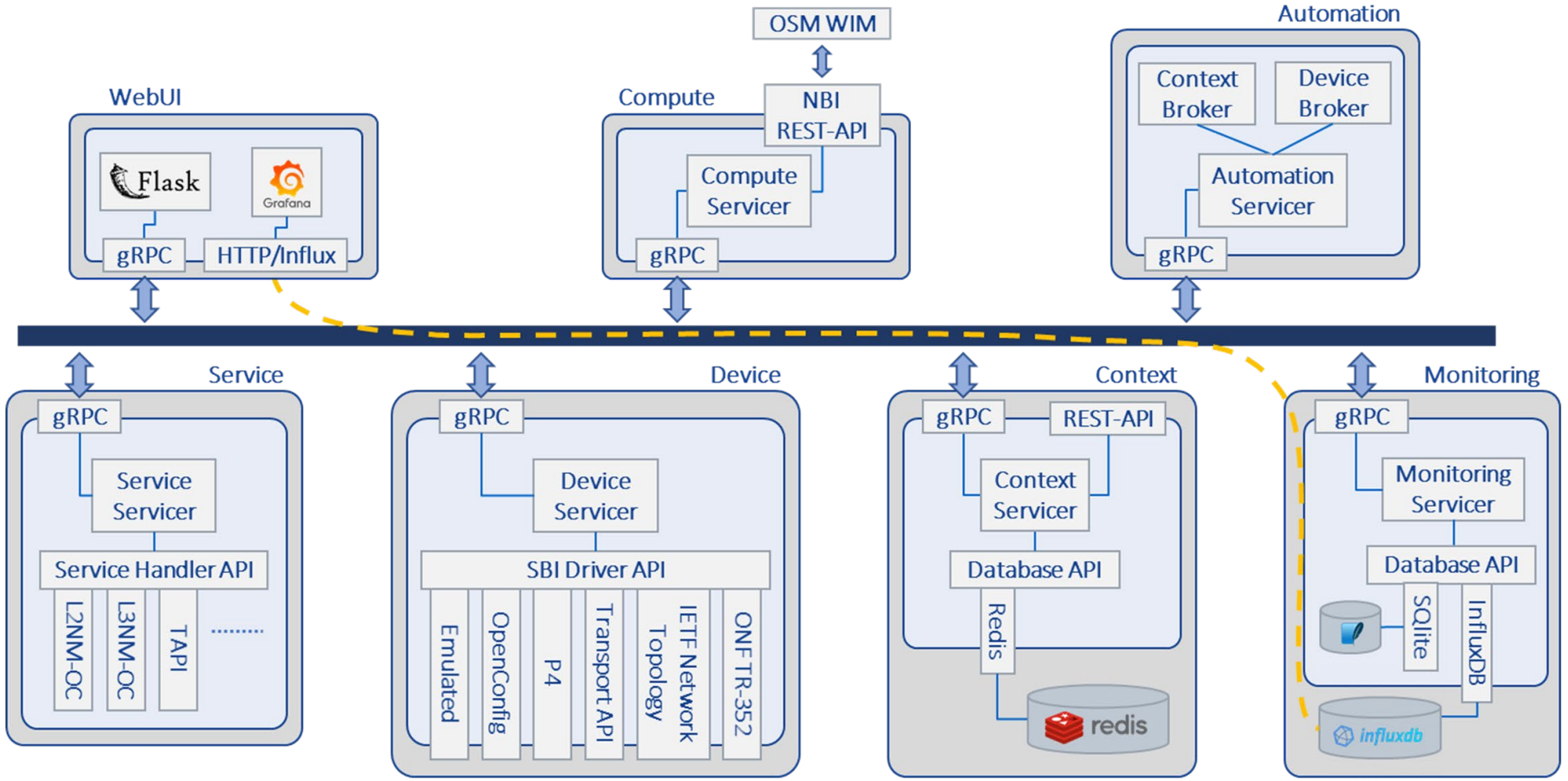
TeraFlow **core micro-services** are **tightly interrelated and collaborate** to provide a complete smart connectivity service.

TeraFlow **netApps consume TeraFlow core micro-services**. The TeraFlow netApps provide the necessary carrier-grade features with a dedicated focus on: load-balancing, cybersecurity, auto-scaling, self-healing, and inter-domain smart connectivity services.

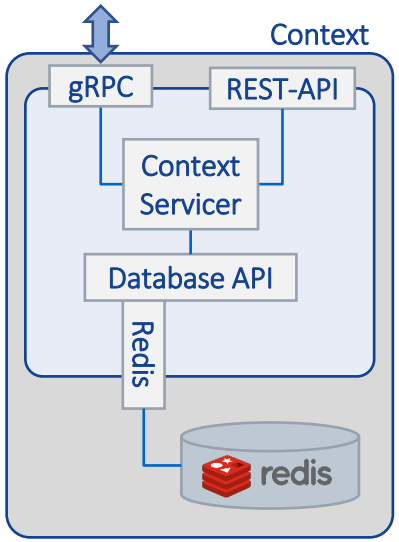
Teraflow OS Architecture



Internal Architecture (first release)



Context Component



<u>Objective:</u> Store the configurations and attributes (active context, topologies, devices, links, and services) of the different network elements managed by the TeraFlow OS

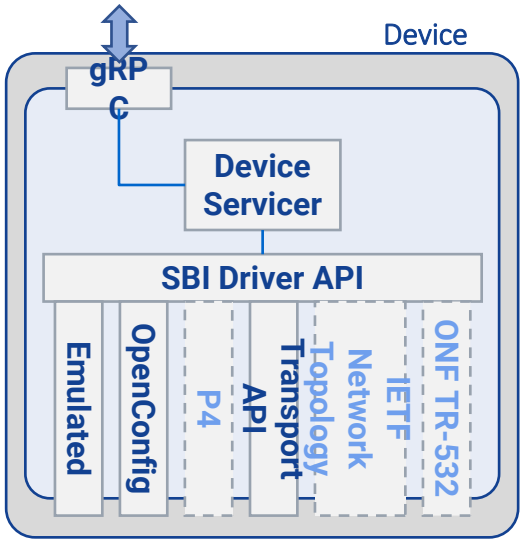
Key features

- Context Client exposed to the rest of TeraFlow OS components
- Context Server exposed through gRPC (read/write) and REST API (read-only)
- Database API with capacity for pluggable Database Backends
- Event streaming through gRPC of entities changed in the database

Achievements

- Fully implemented in a microservice-based approach (Docker over K8s)
- Results shown in OFC 2022 demonstration

Device Component



<u>Objective:</u> Interact with the underlying network equipment using protocols and data models they support. A Driver API enables developers to integrate new drivers into the TeraFlow OS

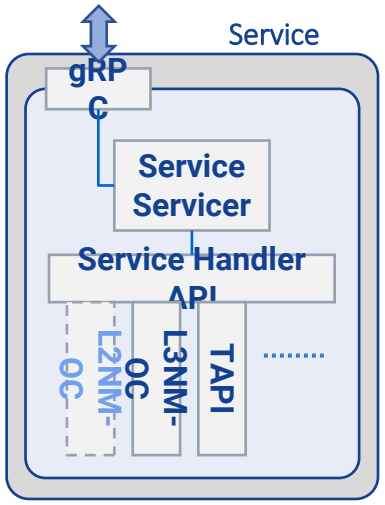
Key features

- Device Client exposed to the rest of TeraFlow OS components
- Device Server exposed through gRPC to the rest of components
- Driver API enables extension with new device drivers
- Preliminary functional versions of "OpenConfig" and "Transport API" drivers
- "Emulated" driver for testing purposes

Achievements

- Fully implemented in a microservice-based approach (Docker over K8s)
- Results shown in OFC 2022 demonstration

Service Component

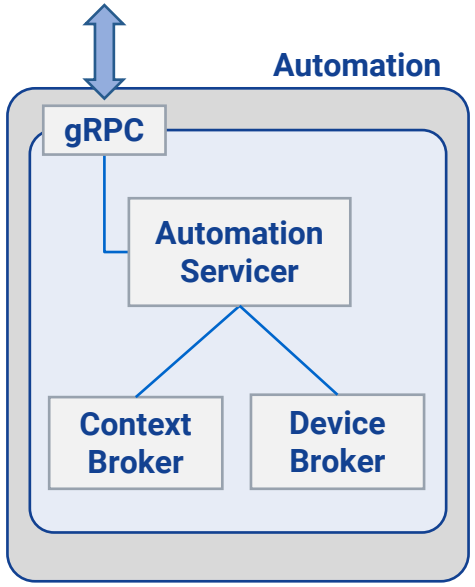


<u>Objective:</u> Manage the life-cycle of the connectivity services established in the network. A Service Handler API enables to add support for different service types, configuration protocols, and data models into the TeraFlow OS

Key features

- Service Client exposed to the rest of TeraFlow OS components
- Service Server exposed through gRPC to the rest of components
- Service Handler API enables adding support for new service types, protocols, and data models
- Preliminary functional versions of “L3NM-OpenConfig” and “Transport API” service handlers

Automation Component



<u>Objective:</u> Zero-touch device (i) onboarding, (ii) reconfiguration, and (iii) deletion as a service to TeraFlow OS and/or external systems

Key features

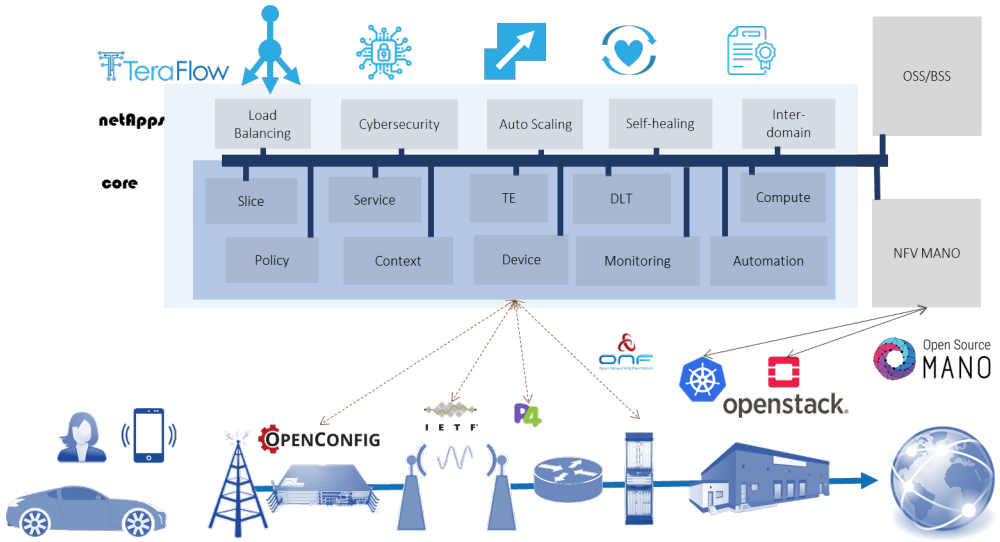
- Automation gRPC server-client
- Context broker to receive device info and device events (which trigger automation)
- Device broker to configure a device
- Zero-touch device provisioning fully-implemented

TeraFlow OS Data Models

Selected data models to be used as both external (including Northbound and Southbound) and internal interfaces.

NorthBound Interfaces

- IETF L2 network model – Available
- IETF L3 network model – On-roadmap
- IETF Transport Network Slice – On-roadmap



SouthBound Interfaces

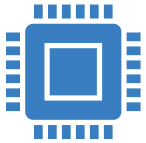
ONF Transport API - Available

ONF TR-532 Microwave - On-roadmap

ONF P4- On-roadmap

OpenConfig – Available On-roadmap extensions

Internal TeraFlow OS Data Models



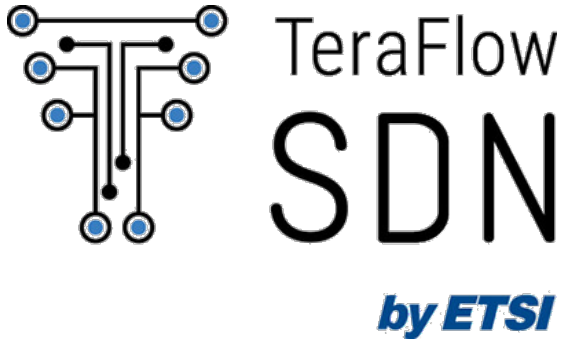
Google Remote Procedure Calls (gRPC) is a protocol based on HTTP/2 as a transport protocol and it uses protocol buffer encodings for transported messages and data models.

As it is based on HTTP/2 and uses byte-oriented encoding, it introduces low latency.

gRPC has been used in highly scalable and distributed systems.

It has been decided to **use gRPC as the internal protocol in the TeraFlow SDN Controller.**

All defined internal protocol buffers are part of the TeraFlow SDN source code, and they are available at https://gitlab.com/teraflow-h2020/controller/-/tree/develop/proto



Thank you!

TFSsupport@etsi.org