Open Source Netphony Suite: Enabling Multi-layer Network Programmability

V. López, R. Jimenez, O. Gonzalez de Dios, L.M. Contreras, J.P. Fernandez Palacios

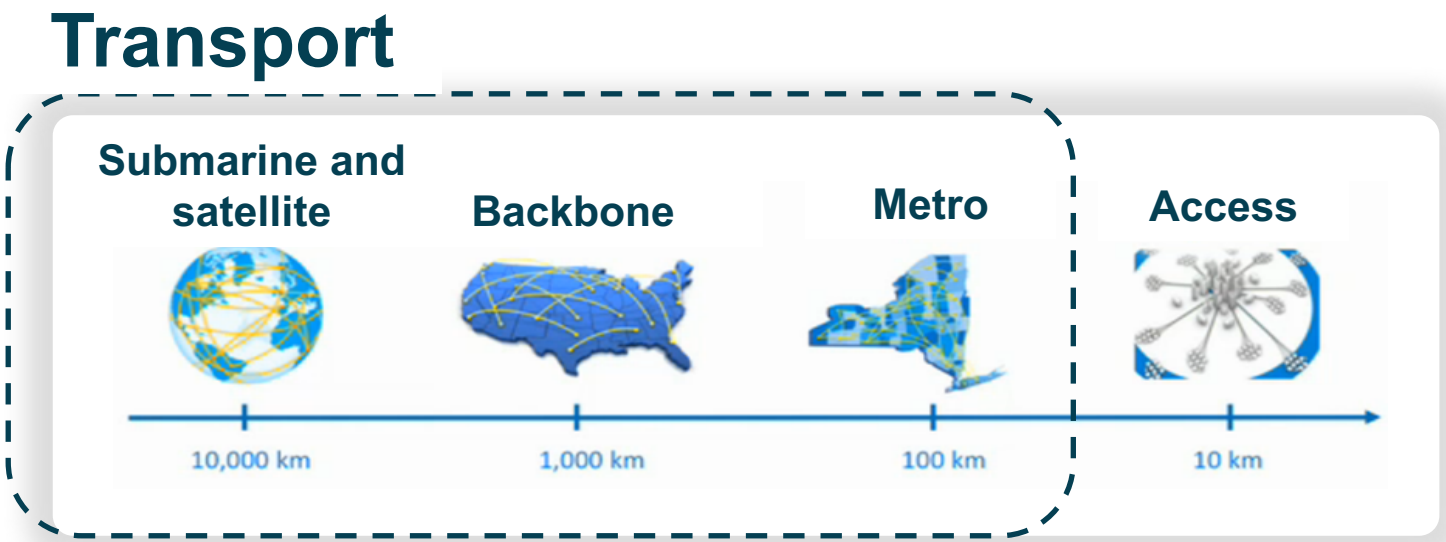
ONDM 2017
Telefonica I+D GCTO Office
16.05.2017

WE CHOOSE IT ALL_

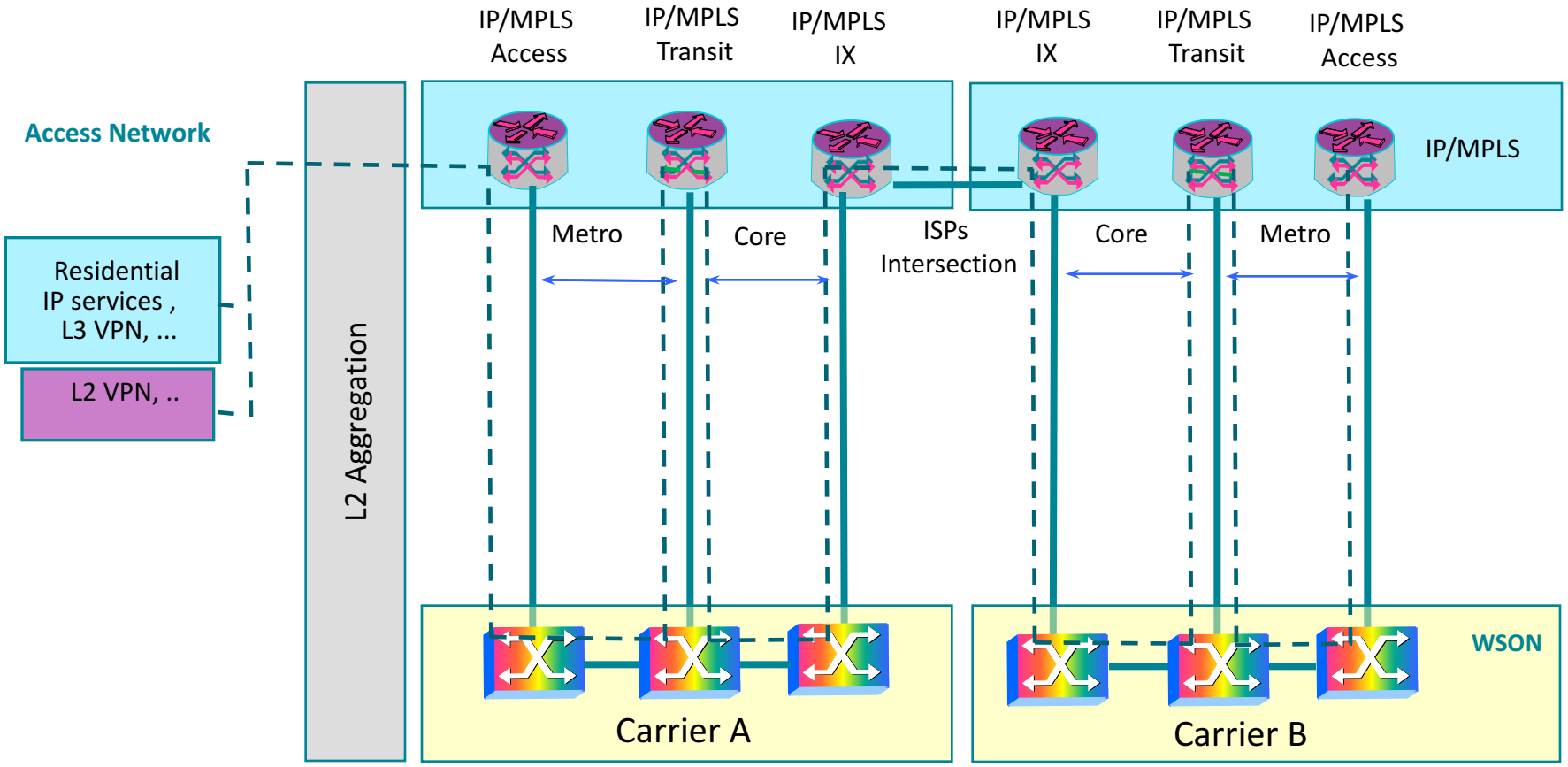
01

Rationale for multi-layer programmability

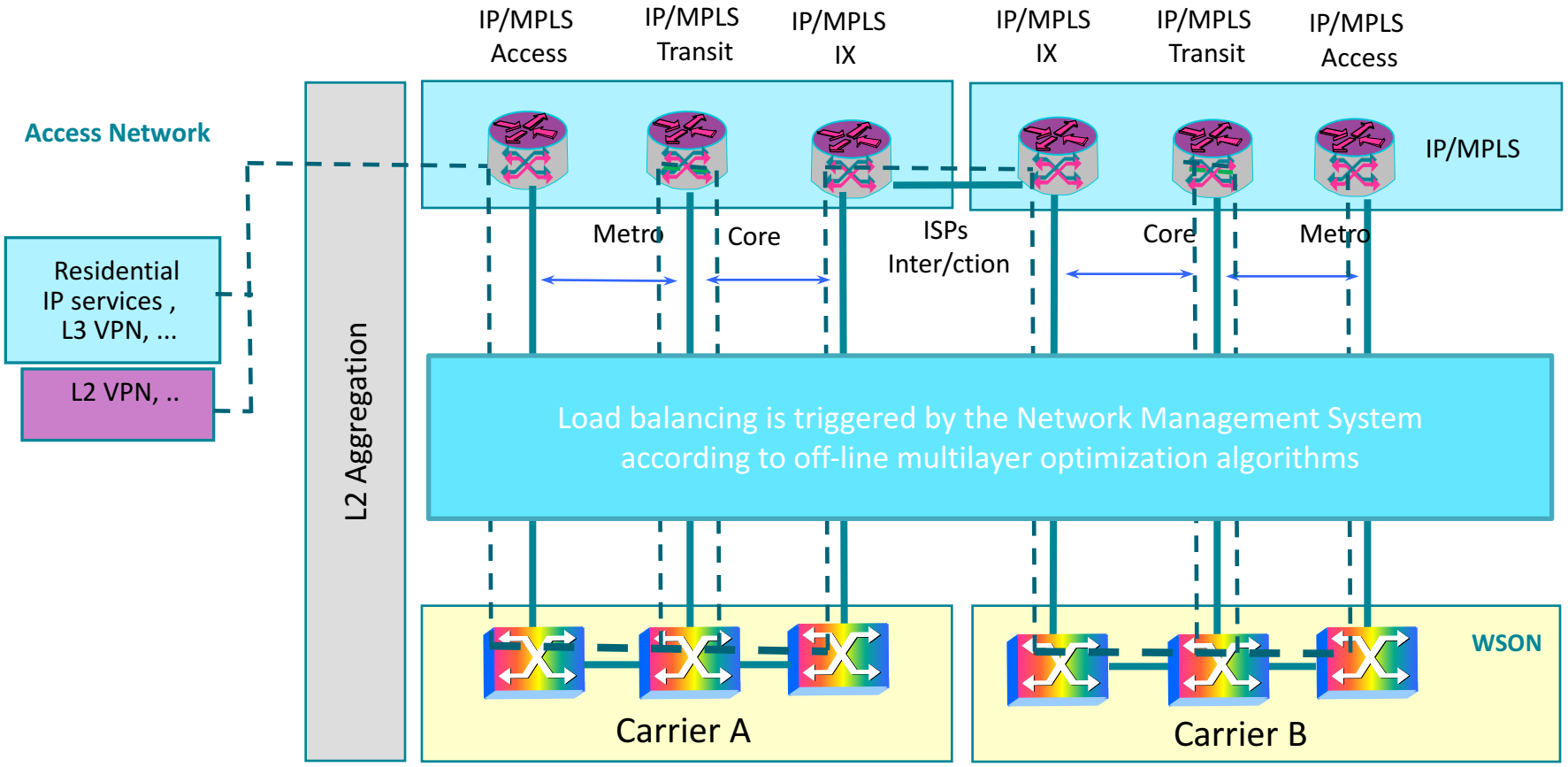
Traditional **transport network** operation is very **complex and expensive**



Independent Layers



Current Multilayer Coordination



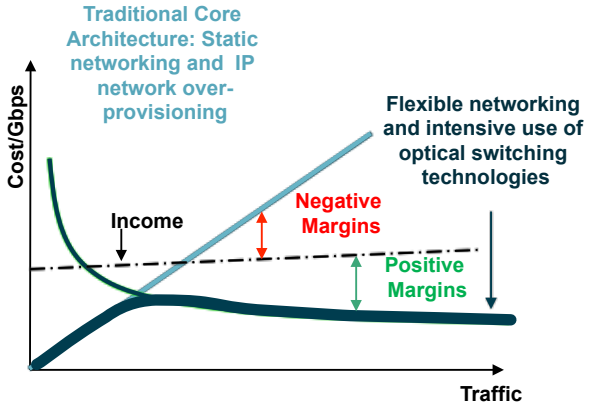
Load balancing between IP/MPLS and optical. High capacity flows (e.g between IP access and IP interconnection) are directly transported over optical

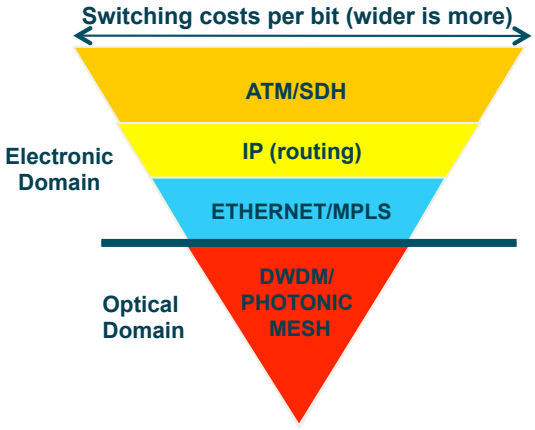
Multi-layer optimization

Why is it interesting to work on this?

- **CAPEX savings:** Core IP and optical transport resources minimization; Reduction of investments in NMS
- **OPEX savings:** Minimization of manual configuration actions; Simplified NMS operation and maintenance
- **Competitiveness gain:** Increased survivability

Cost per bit minimization for increasing traffic demands (e.g. HD Video OTT could multiply by five the traffic of core transport networks)





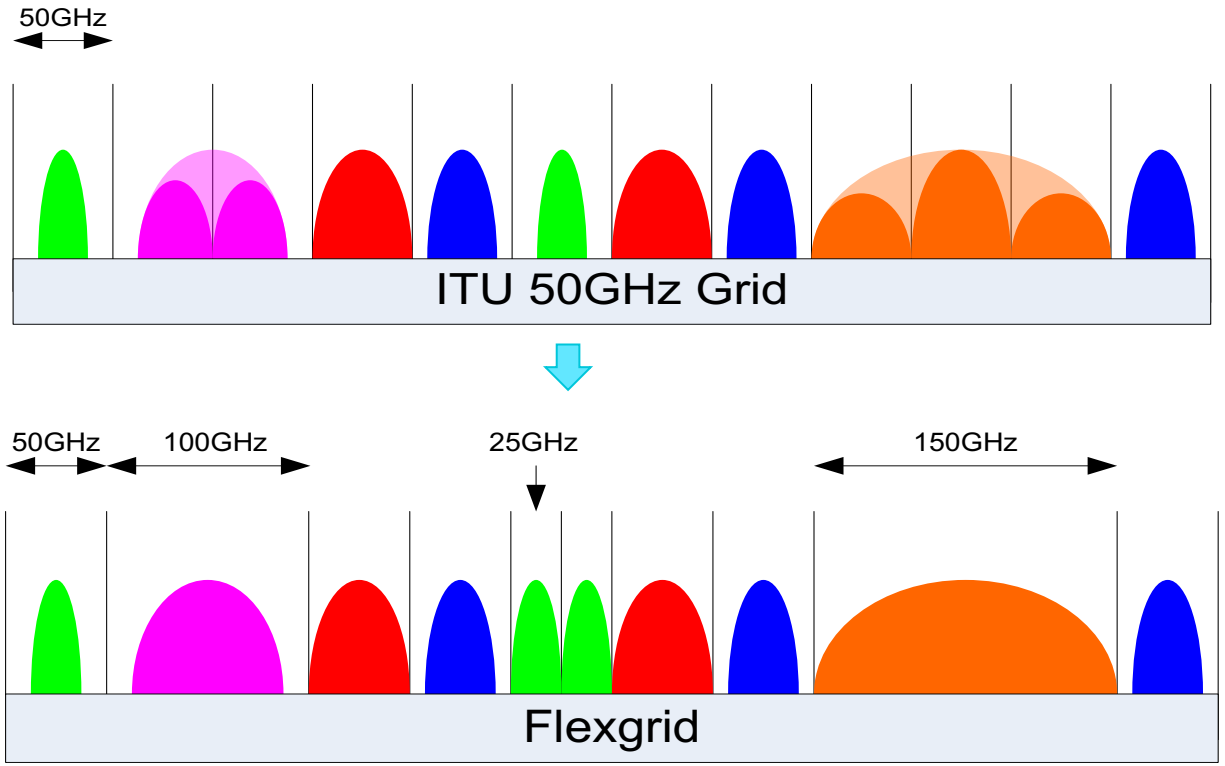
02

Optical Layer Evolution

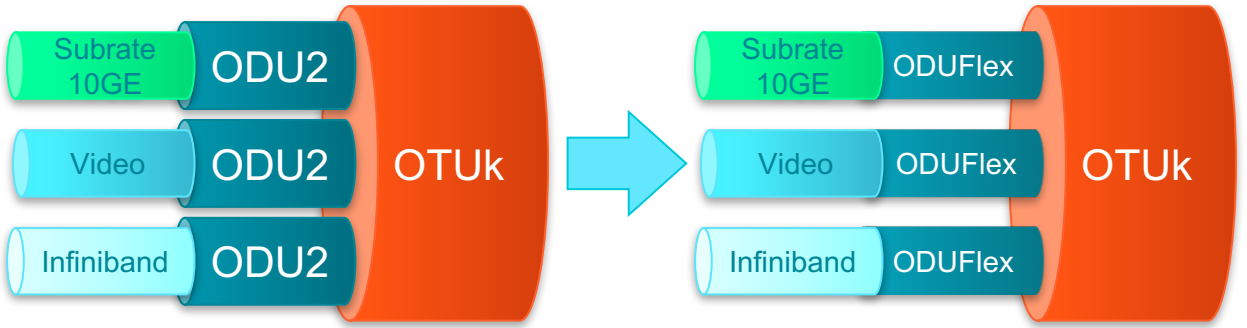
Optical transmission is becoming flexible

. . .

From Static Spectrum to Elastic Spectrum Allocation

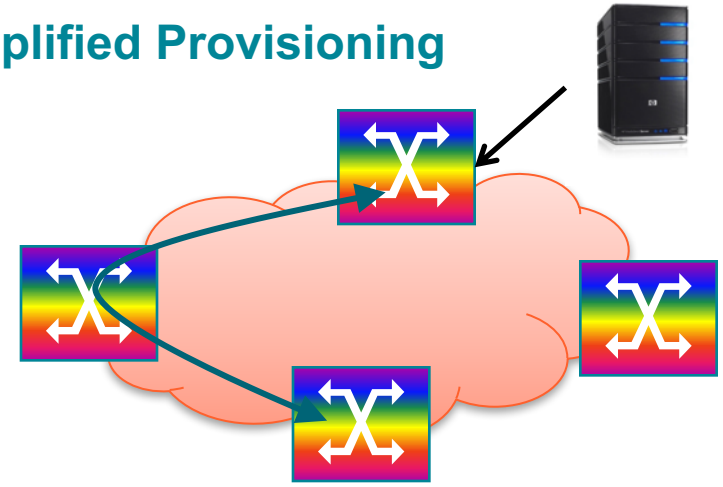


From Static ODU to ODUFlex

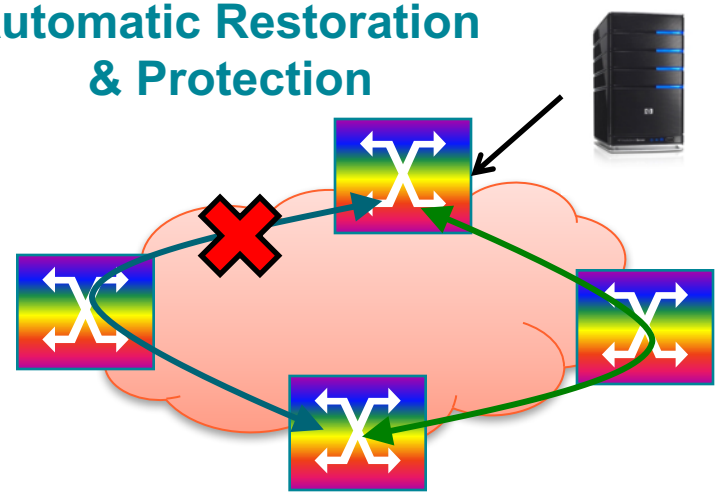


… and a Software based control provides unprecedent flexibility

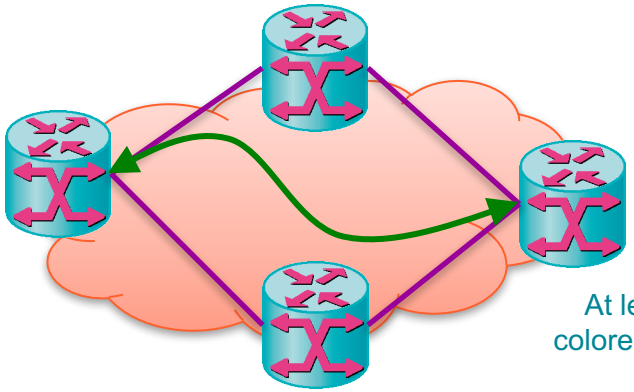
Simplified Provisioning



Automatic Restoration & Protection



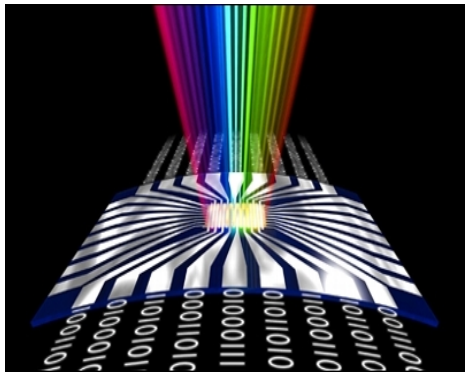
Dynamic topology creation



At least without colored interfaces…

Elastic Optical Networks

Efficient use of available spectrum and **flexibility** (change modulation, reconfigure by software and slicing)



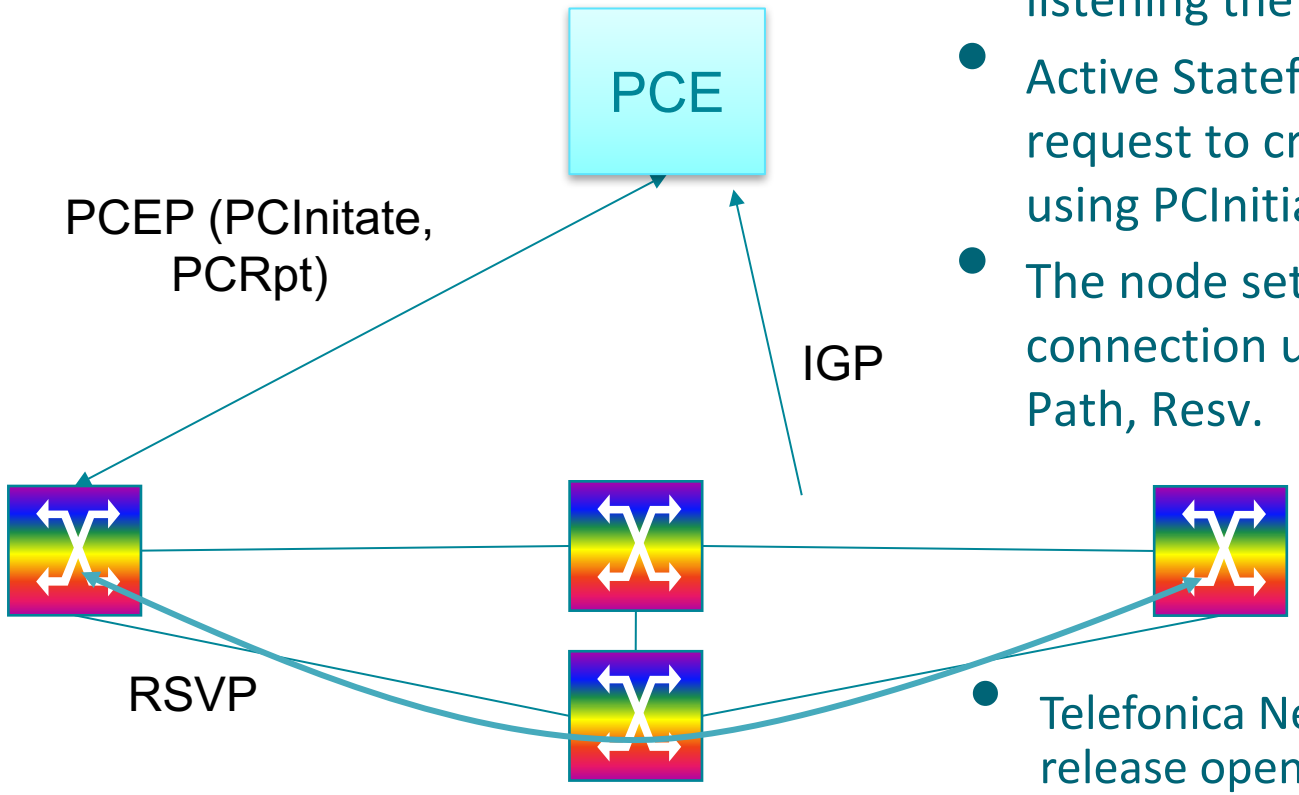
Full potential enabled by a control based on a combination of **signaling and SDN** concepts

03

Control Architecture: GMPLS + Path Computation Element

Path Computation Element

GMPLS is complemented with a logically centralized element, the PCE

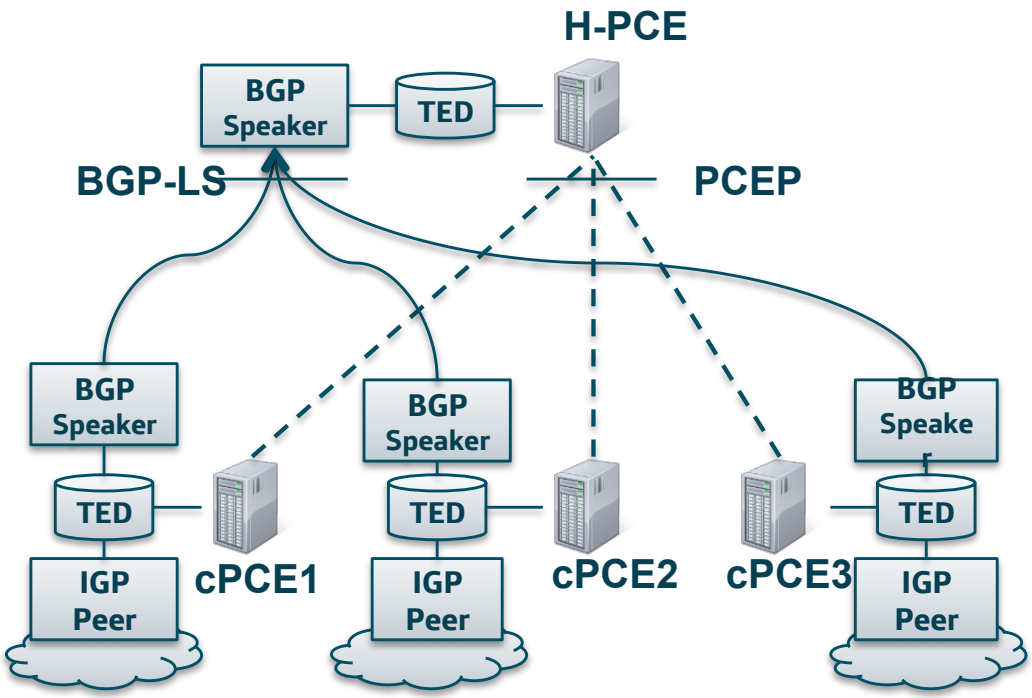


- PCE learns the TE DB listening the IGP.
- Active Stateful PCE can request to create a path using PCInitiate.
- The node set-up the connection using RSVP Path, Resv.

- Telefonica Netphony release open source PCE implementation and GMPLS control plane.

Hierarchical Path Computation Element for Multi-domain Flexi-grid networks

- Hierarchical Path Computation Element extends PCE concept for multi-domain scenarios with multiple Flexi-grid networks.



- Each domain learns the topology as shown in previous slide.
- Parent PCE learns the TE DB via BGP-LS.
- Parent Active Stateful PCE can ask for path computation (PCEReq) and request to create a path (PCInitiate).
- Each PCE configures its domain.

Hierarchical Path Computation Element

- New protocol to export the optical topology between the domains.
  - BGP-LS.

- Technical feasibility demonstrated
  - Multi-partner demonstration for EON

F. Paoluci et al. Experimenting Hierarchical PCE Architecture in a Distributed Multi-Platform Control Plane Testbed , in OFC 2012

M. Cuaresma, et al "*Experimental Demonstration of H-PCE with BGP-LS in elastic optical networks"*, in ECOC 2013.

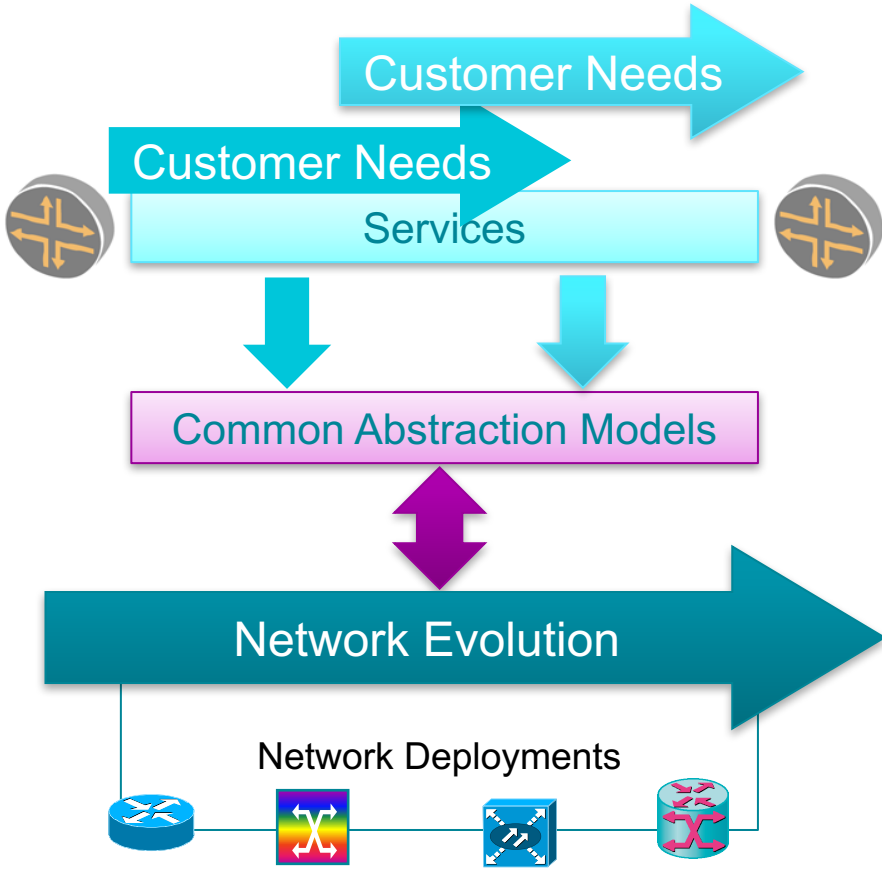
O. Gonzalez de Dios, et al. "*First Demonstration of Multi-vendor and Multi-domain EON with S-BVT and Control Interoperability over Pan-European Testbed"*, ECOC 2015

04

Control Architecture: SDN and Transport API

Common abstraction model

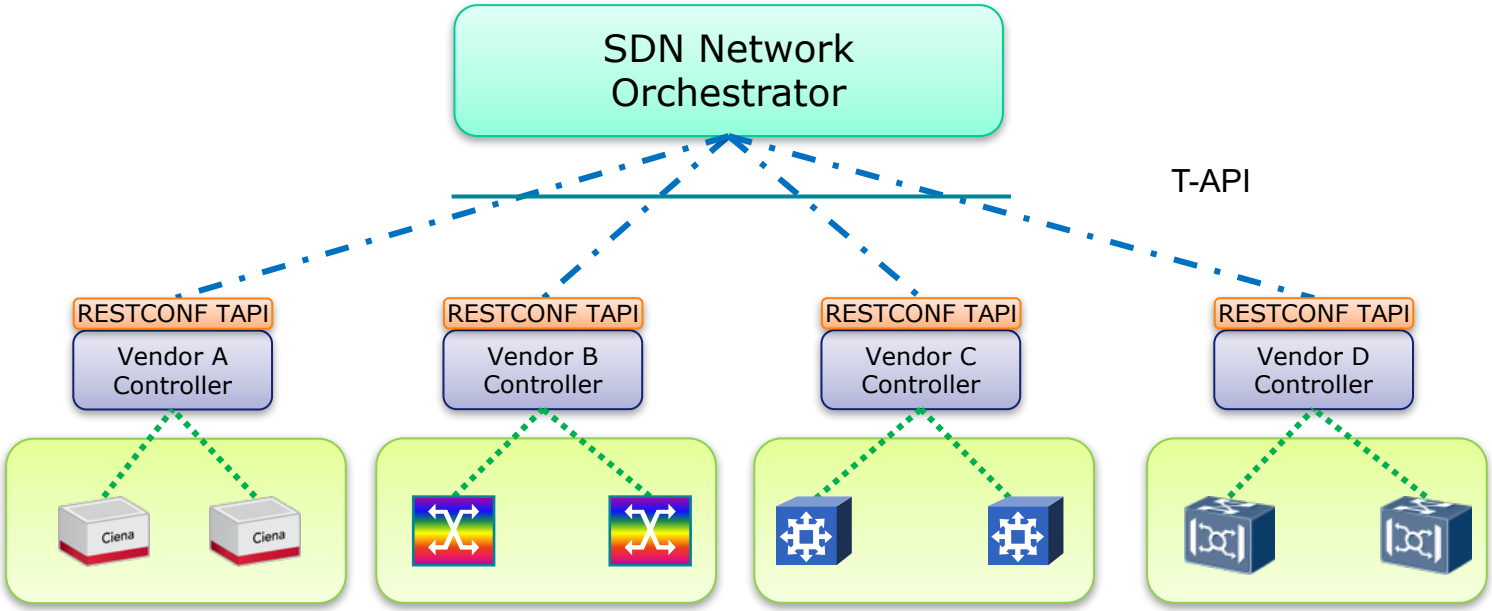
- Program networks with software is not new for transport.
- SDN must impact on the deployment of Standard Interfaces.



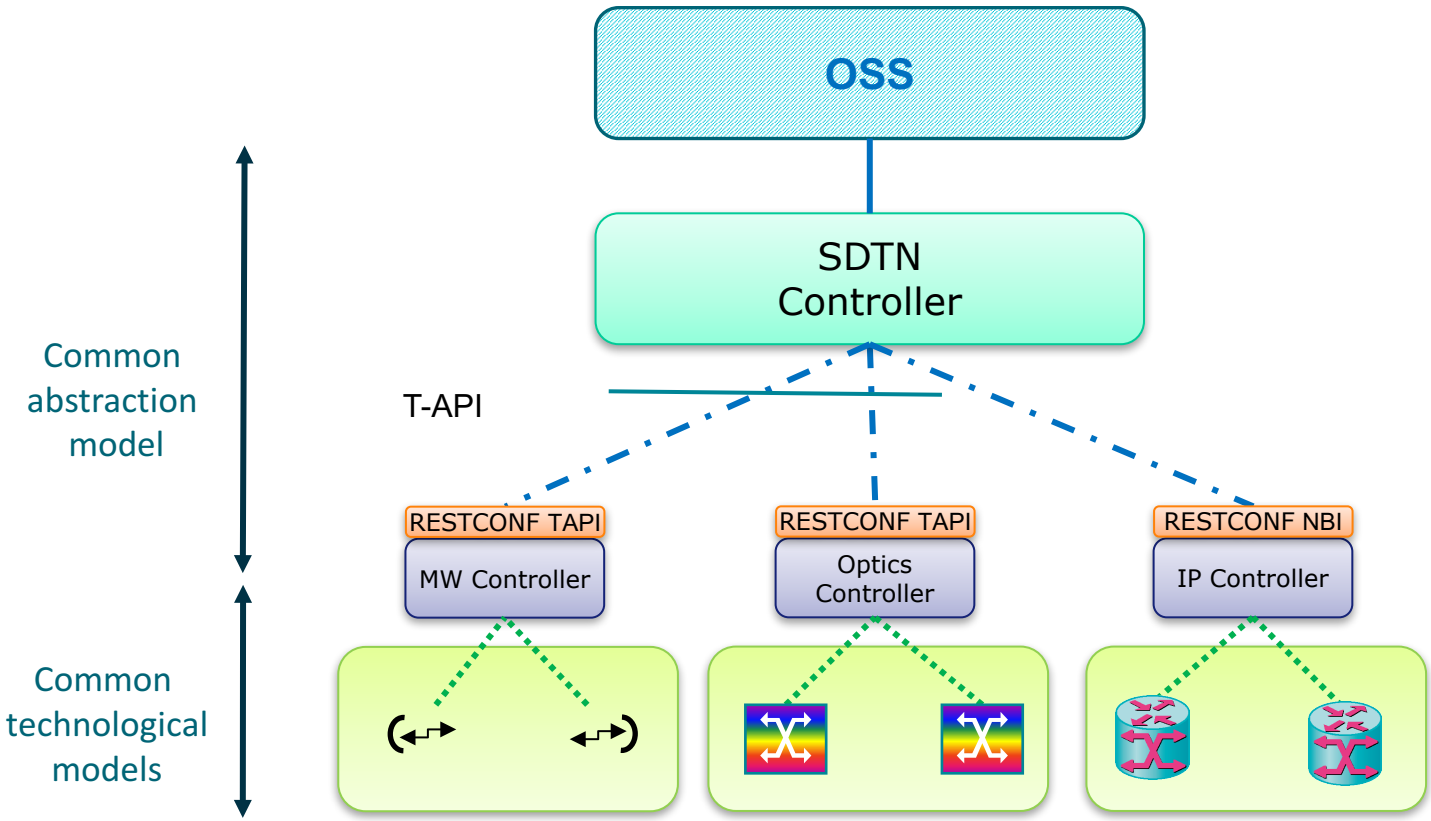
Transport API

OPEN NETWORKING FOUNDATION

- The Transport API (T-API) abstracts a set of control plane functions used by an SDN Controller, allowing the interworking of heterogeneous control plane paradigms (i.e., OpenFlow, GMPLS/PCE).



Software Defined Transport Network



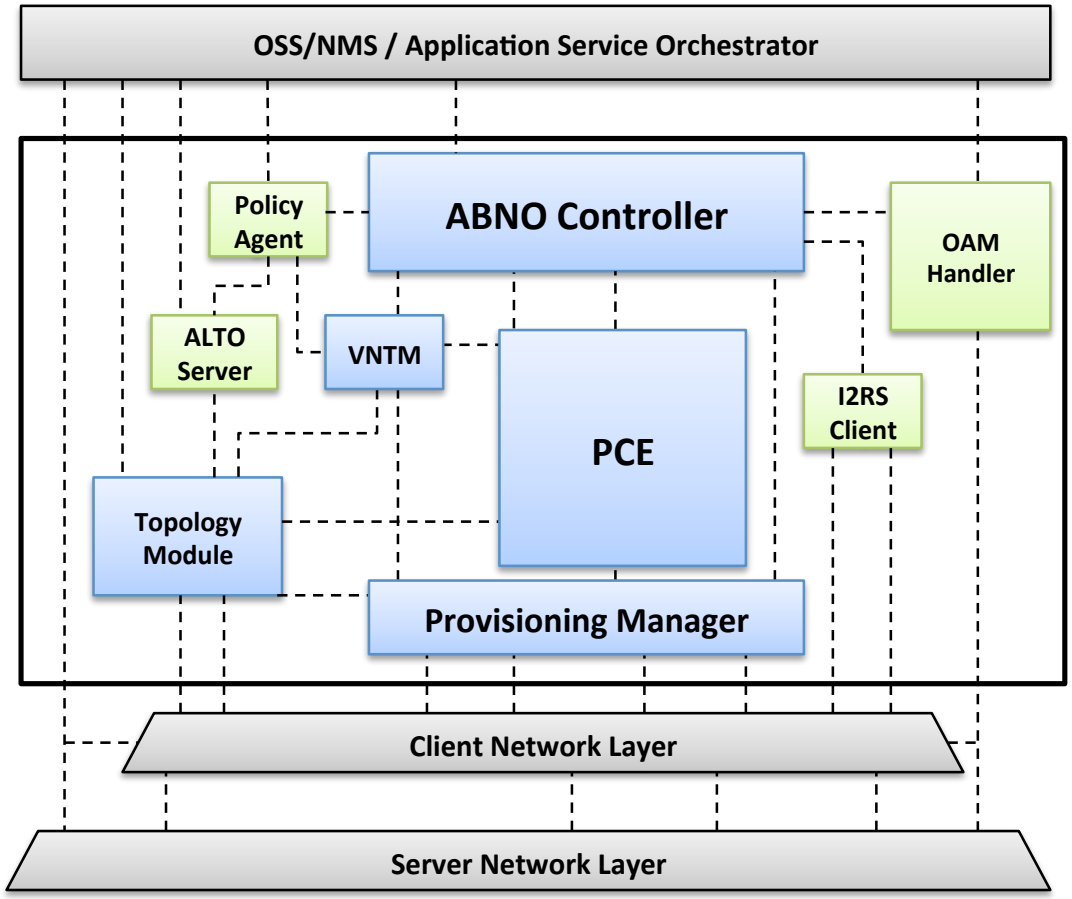
05

Netphony Orchestration suite

Netphony orchestration suite

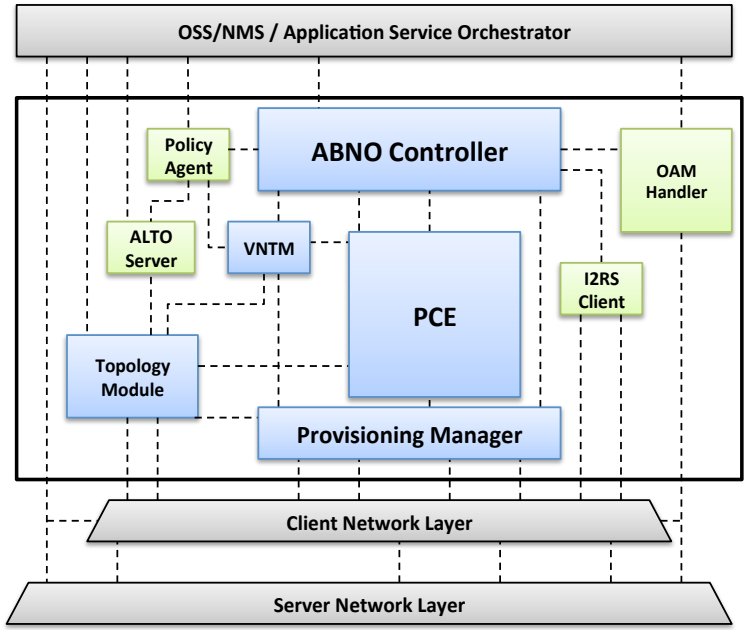
- **Netphony suite** is a set of **Java** based components that can be used either as stand-alone appliances or as libraries embedded in third-party software.
- The source code is publicly available in a set of **github** repositories
- Also, each component can be built using **maven**, which includes the detail of the dependencies for easier management with external third party libraries.
- The components are also distributed as artifacts, with the jar file and javadoc, in the maven central repository.
- Implements components of the **ABNO Architecture**

ABNO Architecture (functional components)



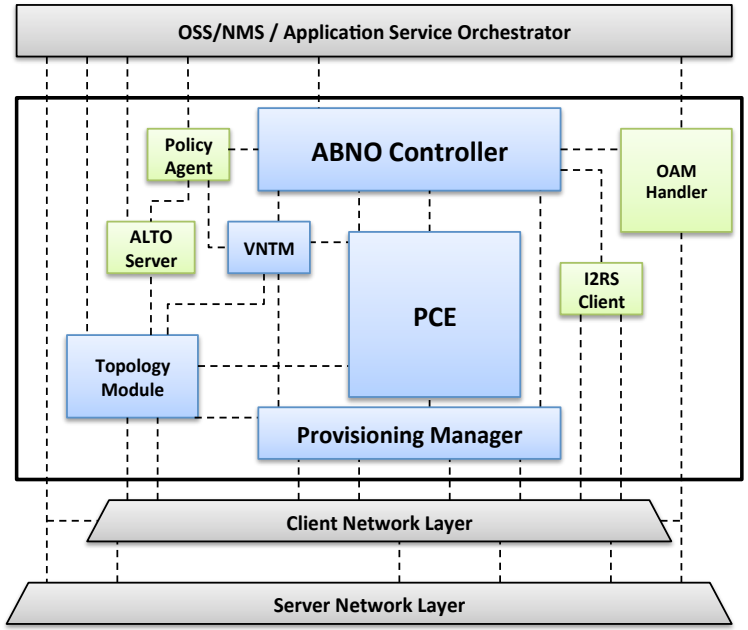
ABNO building blocks

- **ABNO Controller**
  - Main component of the architecture
  - Responsible of orchestrating
  - Request from the NMS/OSS and selects the appropriate workflow to follow in order to satisfy each request.
- **Policy Agent**
  - Stores the restrictions and policies
  - Responsible for propagating those
- **Path Computation Element**
  - Path computation across the network graph.
  - Coordination between multiple PCEs for multi-domain (for example, inter-AS) or multi-layer networks.
  - Instantiation capabilities.



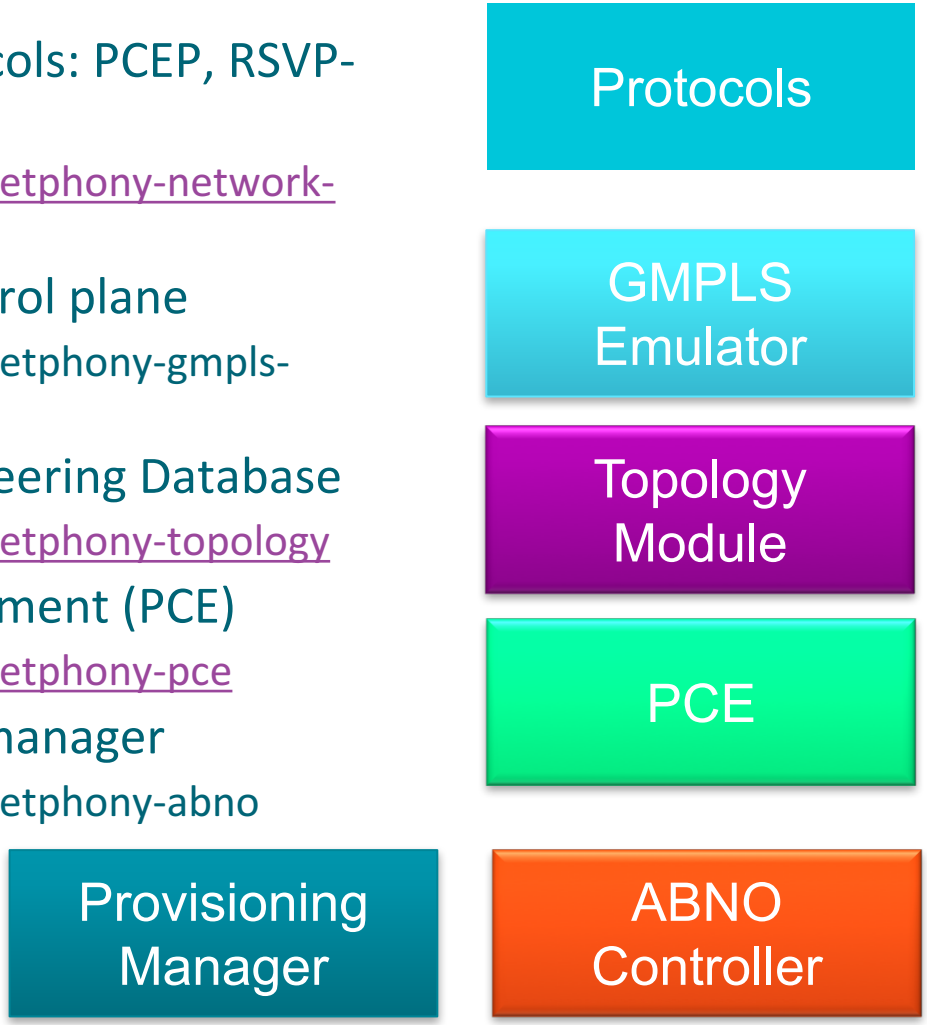
ABNO building blocks

- **Virtual Network Topology Manager (VNTM)**
  - Maintaining the topology of the upper layer by connections in the lower layer.
  - Simplifies the upper-layer routing and traffic engineering decisions.
- **Topology Module.**
  - Retrieve and provide network topology information, both per-layer topologies as well as inter-layer topology.
- **Provisioning Manager.**
  - In charge of configuring the network elements so the LSP can be established.
  - There are several protocols that allow the configuration of specific network resources such as Openflow, Netconf, CLI and PCEP.



Netphony components

- Java Library of Networking Protocols: PCEP, RSVP-TE, OSPF, BGP-LS:
  - https://github.com/telefonicaid/netphony-network-protocols
- Network emulator & GMPLS control plane
  - https://github.com/telefonicaid/netphony-gmpls-emulator
- BGP-LS Speaker and Traffic Engineering Database
  - https://github.com/telefonicaid/netphony-topology
- Java Based Path Computation Element (PCE)
  - https://github.com/telefonicaid/netphony-pce
- ABNO Controller & provisioning manager
  - https://github.com/telefonicaid/netphony-abno



Networking Protocol Library

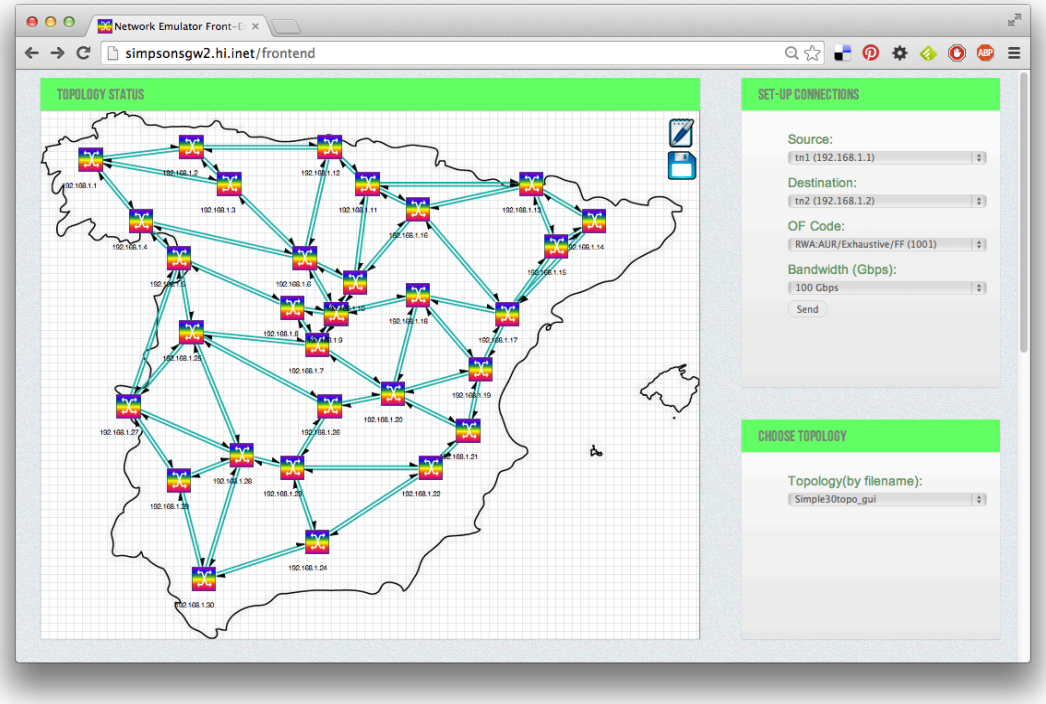
Protocols

- PCEP and BGP-LS protocols
- Provides encoding/decoding of the protocols
- Easy to extend
- Extensions to cover multi-domain networks
- Extensions to carry also resource information
- API available in http://telefonicaid.github.io/netphony-network-protocols/api/

GMPLS emulated domain

GMPLS Emulator

- Telefonica I+D control plane test bed is composed by 30 GMPLS nodes.
- Each GMPLS controller is a VM and all are running is a server with two processor Intel Xeon E5-2630 2.30GHz, 6 cores each, and 192 GB RAM.



BGP-LS speaker and Topology

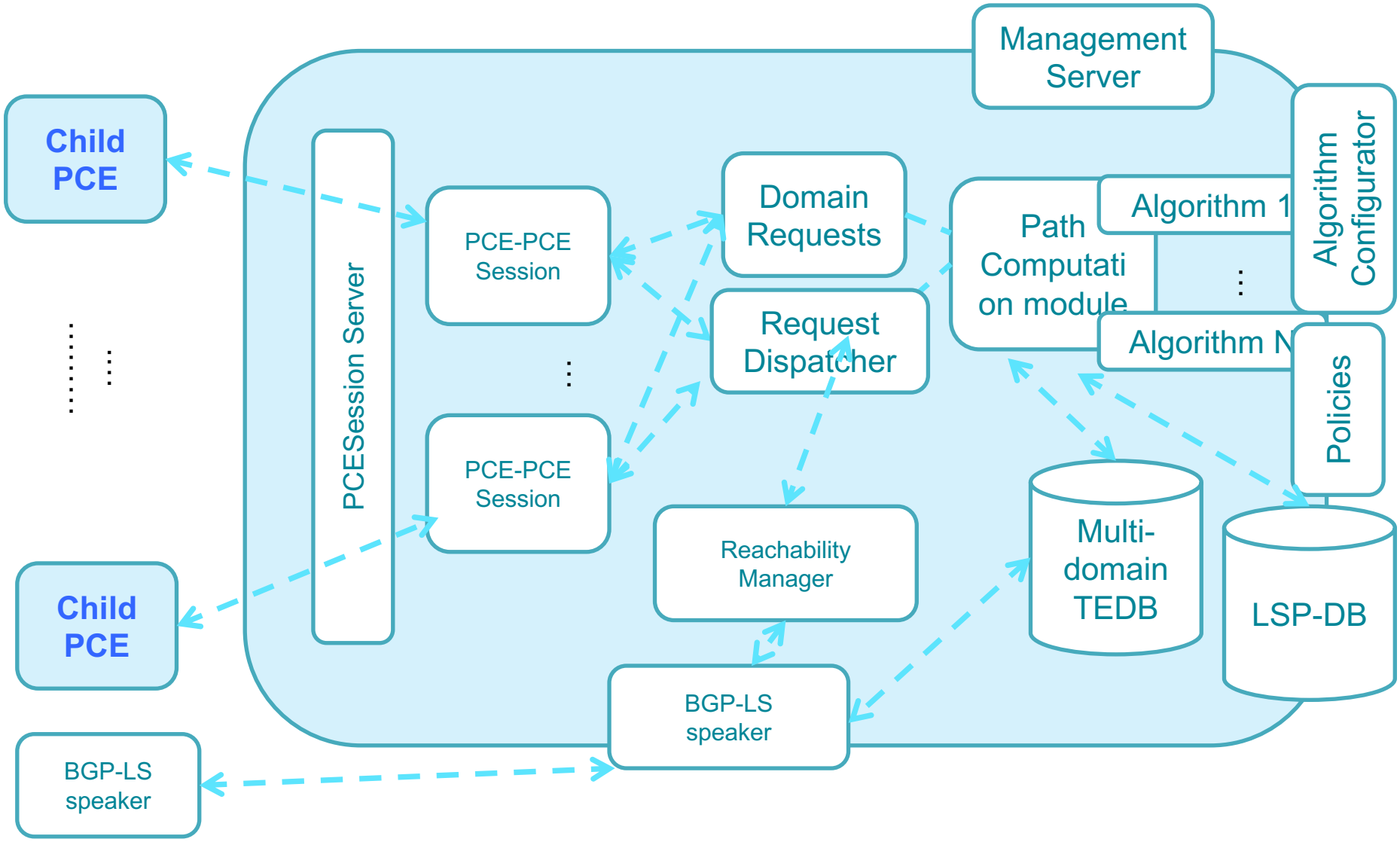
Topology Module

- Used for Topology acquisition
- Full BGP-LS Speaker
- TEDB stores in memory a graph with nodes, links and their traffic engineering attributes.
- Recently added: performance metrics (e.g. delay)
- BGP-LS speaker maintains a collection of TEDBs sorted by domain_id and a multi-domain TEDB
- Work in progress:
  - Integration of ONF Transport-API
  - Integration of APIs with other controllers

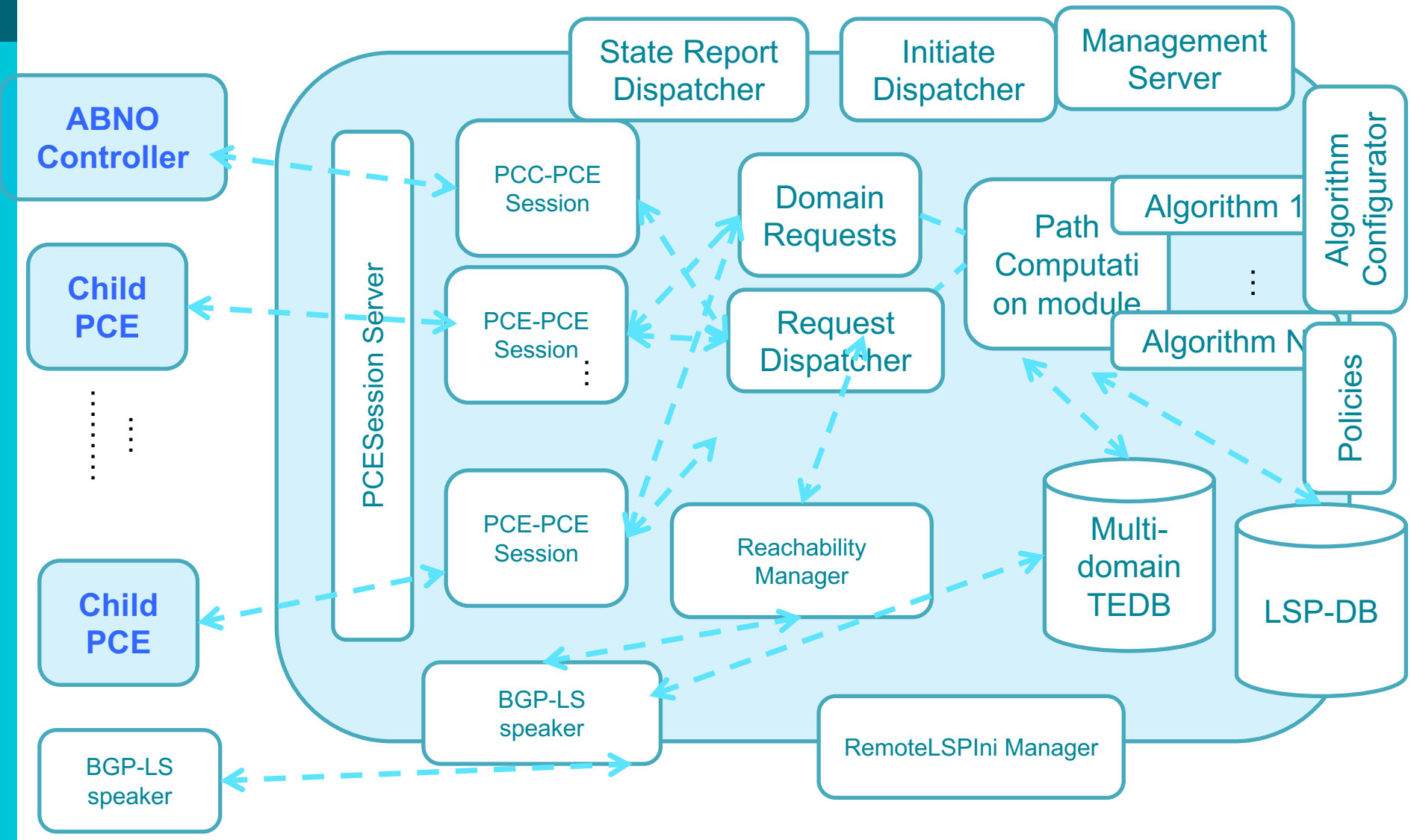
Path Computation Element

- Implementation of a PCE (Path Computation Element)
- Single domain and multi-domain Scenario
- Collection of algorithms available
- Easy to extend to add new algorithms and consider other kind of graphs (e.g. service graphs)
- Extended to consider network and computing resources.
- Hierarchical PCE for multi-domain orchestration
- Path Computation client available
- Domain PCE, parent PCE. Passive and Active.
- Multi-technology (depends on network graph and service requested)

Parent PCE


Management Server
Child PCE
PCESession Server
PCE-PCE Session
PCE-PCE Session
Domain Requests
Request Dispatcher
Path Computation module
Algorithm 1
Algorithm N
Algorithm Configurator
Policies
Reachability Manager
Multi-domain TEDB
LSP-DB
Child PCE
BGP-LS speaker
BGP-LS speaker

Active Stateful Parent PCE



ABNO Controller and provisioning manager

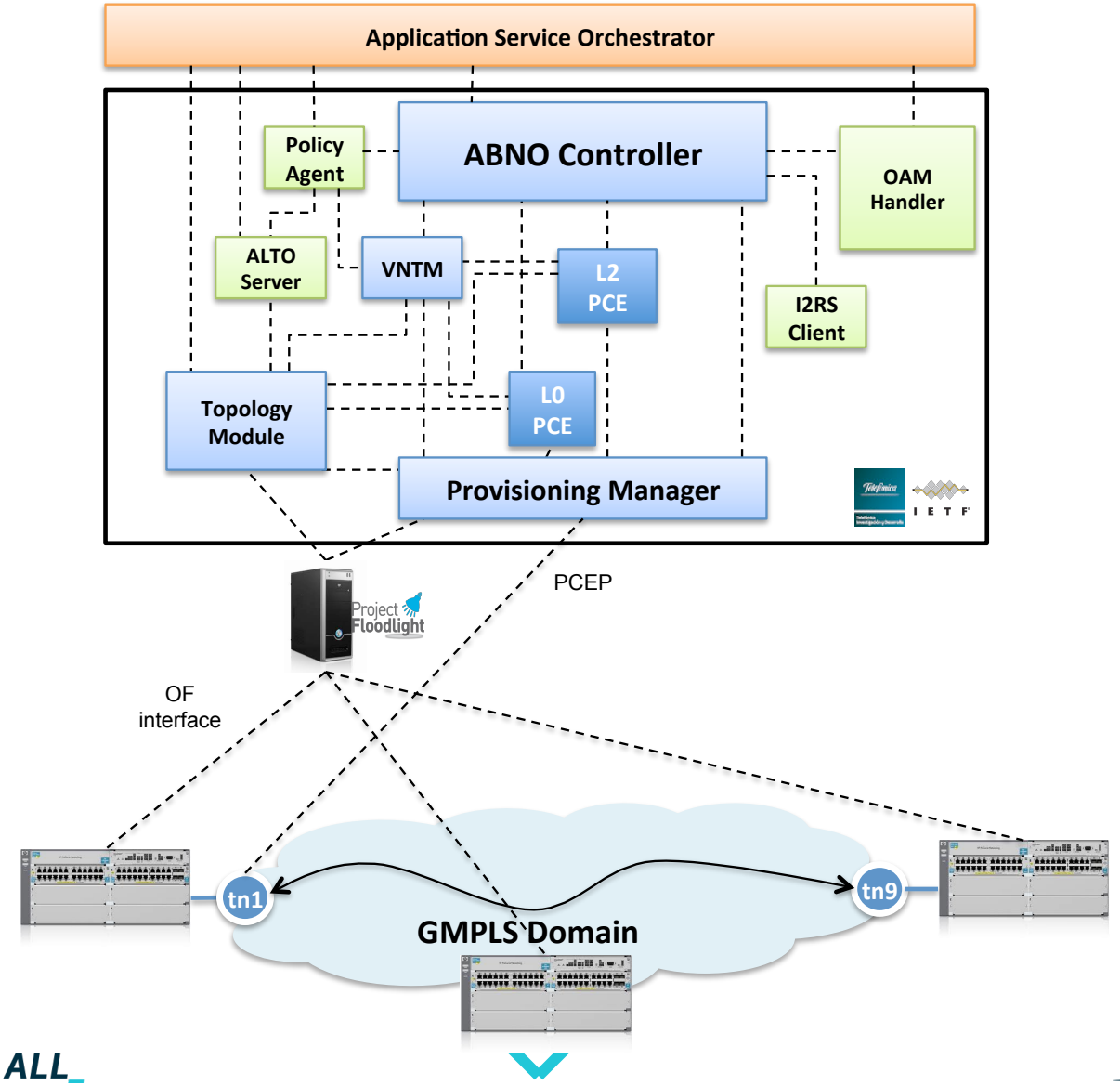
ABNO Controller

- The ABNO Controller is the main component of the architecture and is responsible of orchestrating, and invokes the necessary components in the right order
  - External interface :
    - REST API
    - COP (Common Orchestration Protocol)
  - Interfaces to other components:
    - PCEP
  - Performs Operations with other components (PCE, provisioning manager..)
  - Integrated with Openstack neutron
- The provisioning manager interfaces the network elements using the required interface to configure them.
  - PCEP
  - CLI, Netconf/YANG
  - Rest/APIs

Provisioning Manager

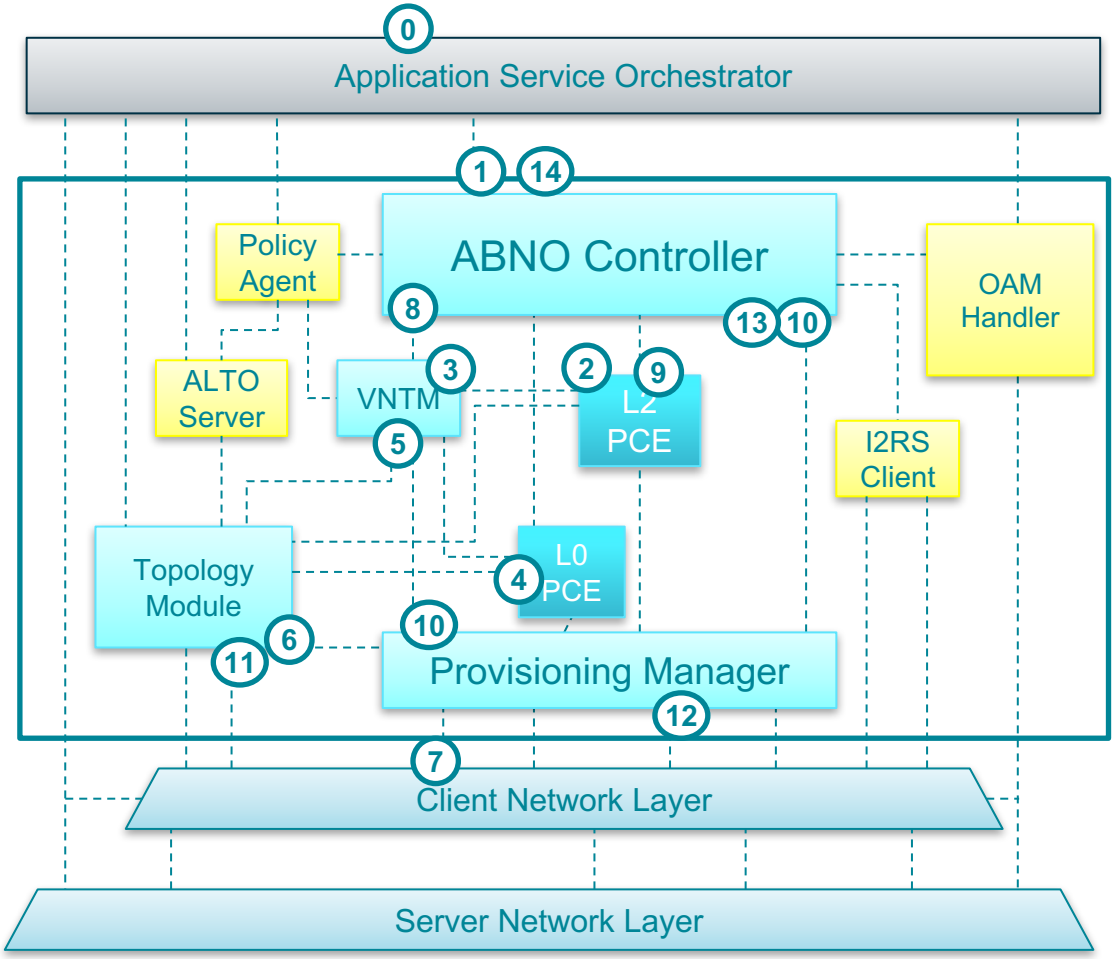
06 E2E orchestration

E2E orchestration

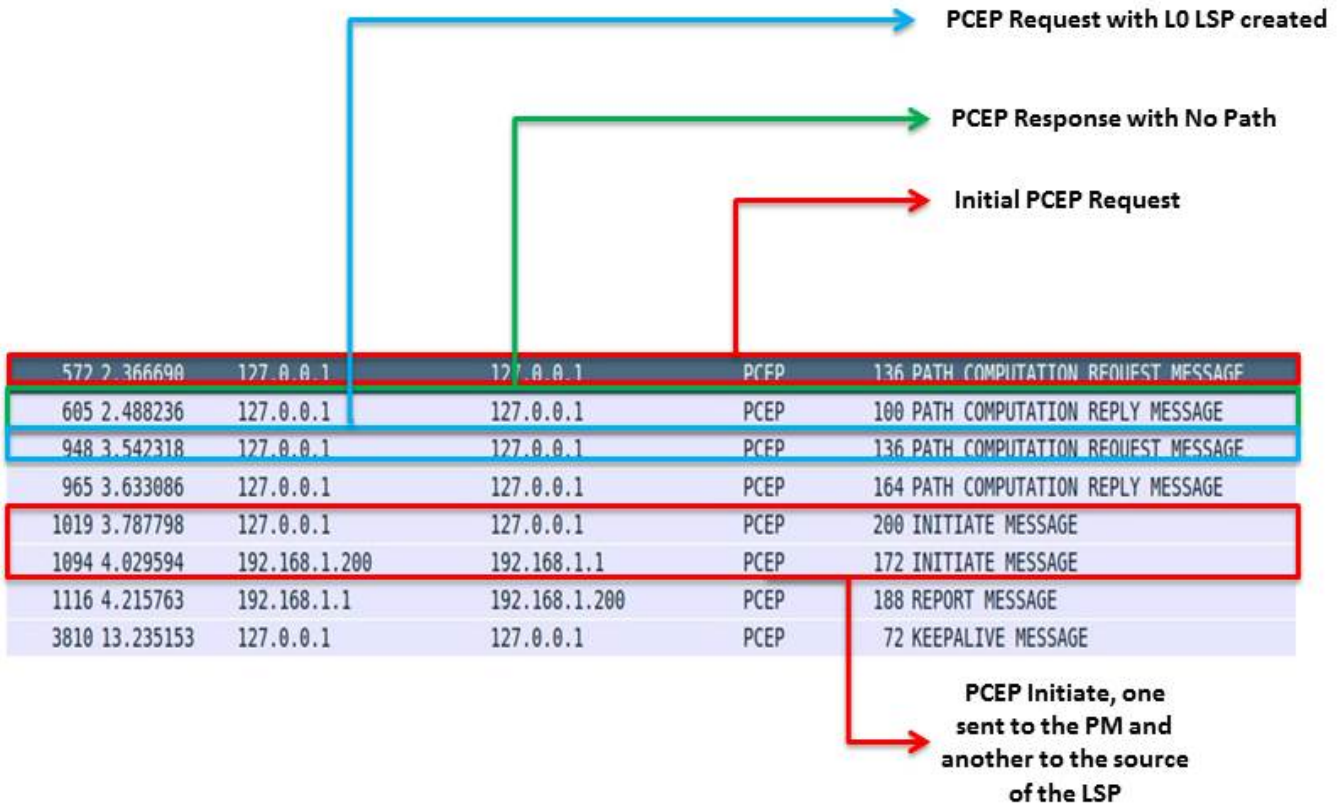


Telefónica

Inter Data Center Connection - Workflow



Inter Data Center Connection - Traces



07

Conclusions

Conclusions

- The SDN architecture proposed here enables automated and simplified network service provisioning through different vendors, network segments (metro, core, data center…) and technologies (IP/MPLS, optical, OpenFlow…)
- Such automation and simplification could be achieved by applying two complementary measures:
  - Network configuration points minimization by transferring multidomain and multilayer provisioning functionalities from NMS to the control plane.
  - PCEP as main SBI.
  - Transport API as NBI.
- Encourage optical community to push for an interface and avoid multiple standards.

Thank you!!

The work on this paper is partially funded by the European Commission within the H2020 Research and Innovation program under the ACINO project

Telefónica

WE CHOOSE IT ALL_