WFO Optical Module

Re-usable models, steps, and clients

Matteo Colantonio, Karel van Klink

Helsinki

June 2026

tnc26

GÉANT

Co-funded by the European Union

We have a dream!

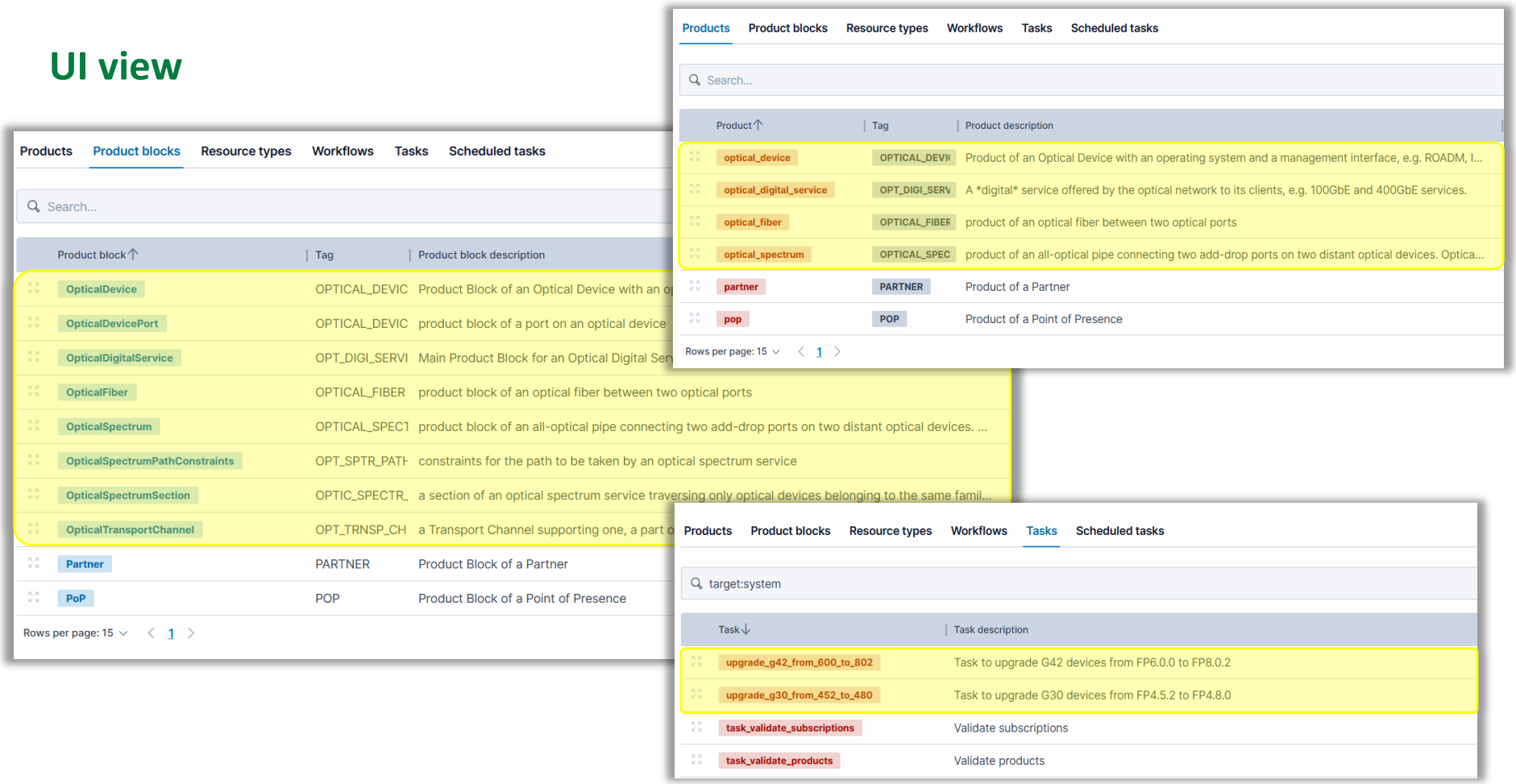
```
$ uv add orchestrator-optical
$ python main.py db migrate-domain-models ”Add optical module 🎉"
```

You’ll get:

1. **SubscriptionModels & ProductBlockModels** for optical hardware and services.
2. **Hardware-agnostic steps & workflows:** e.g. configure_optical_xcon(), `configure_transceiver()`, …
3. **Hardware-specific clients:** we can share code privately or use the same YANG-to-client compiler.*

**search “yang2sdk” on GitHub + lighting talk 2nd strike on Thursday*

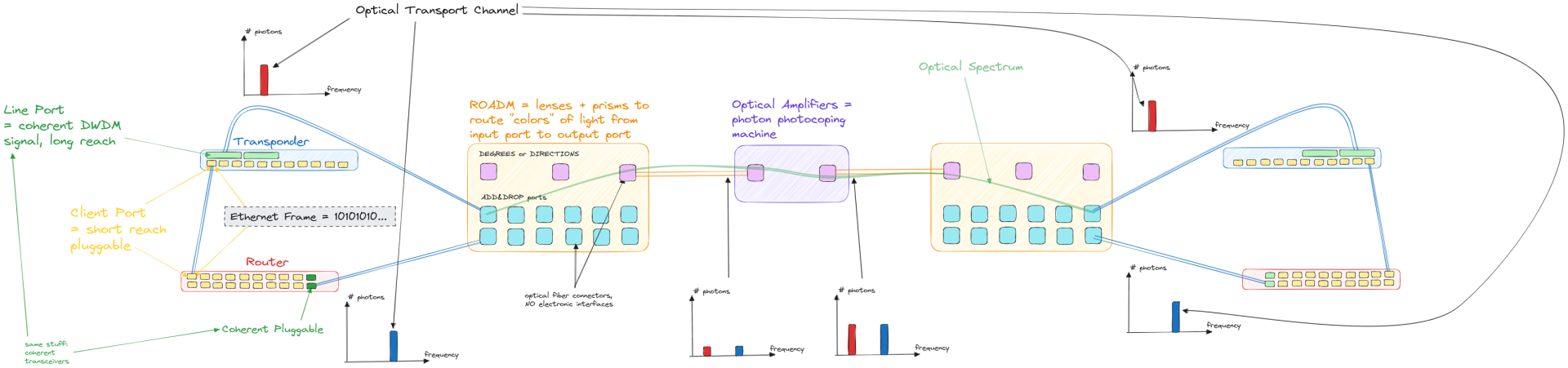
UI view


Products
Product blocks
Resource types
Workflows
Tasks
Scheduled tasks
Search...
Product block
Tag
Product block description
OpticalDevice
OPTICAL_DEVIC
Product Block of an Optical Device with an o
OpticalDevicePort
OPTICAL_DEVIC
product block of a port on an optical device
OpticalDigitalService
OPT_DIGI_SERVI
Main Product Block for an Optical Digital Serv
OpticalFiber
OPTICAL_FIBER
product block of an optical fiber between two optical ports
OpticalSpectrum
OPTICAL_SPECT
product block of an all-optical pipe connecting two add-drop ports on two distant optical devices. ...
OpticalSpectrumPathConstraints
OPT_SPTR_PATH
constraints for the path to be taken by an optical spectrum service
OpticalSpectrumSection
OPTIC_SPECTR_
a section of an optical spectrum service traversing only optical devices belonging to the same famil...
OpticalTransportChannel
OPT_TRNSP_CH
a Transport Channel supporting one, a part o
Partner
PARTNER
Product Block of a Partner
PoP
POP
Product Block of a Point of Presence
Rows per page: 15
1
Products
Product blocks
Resource types
Workflows
Tasks
Scheduled tasks
Search...
Product
Tag
Product description
optical_device
OPTICAL_DEVI
Product of an Optical Device with an operating system and a management interface, e.g. ROADM, I...
optical_digital_service
OPT_DIGI_SERV
A *digital* service offered by the optical network to its clients, e.g. 100GbE and 400GbE services.
optical_fiber
OPTICAL_FIBER
product of an optical fiber between two optical ports
optical_spectrum
OPTICAL_SPEC
product of an all-optical pipe connecting two add-drop ports on two distant optical devices. Optica...
partner
PARTNER
Product of a Partner
pop
POP
Product of a Point of Presence
Rows per page: 15
1
Products
Product blocks
Resource types
Workflows
Tasks
Scheduled tasks
target:system
Task
Task description
upgrade_g42_from_600_to_802
Task to upgrade G42 devices from FP6.0.0 to FP8.0.2
upgrade_g30_from_452_to_480
Task to upgrade G30 devices from FP4.5.2 to FP4.8.0
task_validate_subscriptions
Validate subscriptions
task_validate_products
Validate products

Modeling optical DWDM networks

https://excalidraw.com/#json=EkJcZlr4hRG9PDDQDKKZ9,DOXsGwnDbK1gDk-eBhf-LA

DWDM Optical Networks and Services



Building Blocks

Nodes:
- ROADMs
- Amplifiers
- Transponders
- Router (Packet Node)

Optical Ports/Interfaces:
- OLS ADD&DROP
- OLS LINE
- Transponder Line
- Transponder Client
- Coherent Pluggable

Optical Pipes: photons out = photons in
- Fiber Patches (meters)
- Fiber Spans (km)
- Leased Spectrum

Optical Services:
- Spectrum Service
- Transport Channel
- Ethernet Service
- 10101010... (more general: Digital Service)

Workflow steps are hardware-agnostic

```
@step("Configure transponders/transceivers")
def configure_trx_line_side(subscription: OpticalDigitalService) -> State:
    ...

    for t in terminations:
        configure_line_transceivers(t: OpticalPortBlock, frequency, mode)

    ...
```


Hardware abstraction layer

```
def configure_line_transceivers(port: OpticalPortBlock, frequency, mode):
    match port.host_node:
        case NokiaGrooveG30Block:
            g30_configure_line_transceivers(port, frequency, mode)
            ...
        case NokiaGxG42Block:
            G42Client()
            ...
        case NokiaSR7750Block:
            lso()
            ...
```


Hardware-specific operations

```
def g30_configure_line_transceivers(port, frequency, mode):
    client = G30Client(port.host_node.management_ips)
    ...
    port_endpoint.update(tx_frequency=frequency, modulation=mode, …)
    ...
```

- These RESTCONF clients 👆 are generated/compiled from YANG modules*.
- Old TL1 interface has already been taken care of as well.
- Private sharing if vendor's IP is a problem.

**search "yang2sdk" on GitHub + lighting talk 2nd strike on Thursday*

Plug-n-play optical orchestrator*

**if same building blocks*

- Today: GARR shares with GÉANT the code to manage its optical network (same hardware).
- Tomorrow: GÉANT builds on top, e.g. adds the code to manage coherent pluggables.
- In the future: someone adds Ciena, then somebody else can use it for free.

We can all take advantage from- and contribute back to a shared module.

Compoundable. Extensible. Maintainable.

Thank you

Any questions?

matteo.colantonio@garr.it