



Modular, Standards-Based Network Orchestration with AI-Accelerated Development

Sonja Filiposka (UKIM)
Roman Łapacz (PCSS)

BalticNOG
September 24-25, 2025

Public (PU)

GN5-2

Network Orchestration and Automation



Why

For whom

How

Why



Moving from manual approach to orchestration and automation

Definition and implementation of workflows
Open-source software with APIs only
Reusable implementations



Dynamic service portfolio

Faster implementation of new advanced network services



Evolution instead of revolution – use of building blocks

For whom

National Research and Education Networks (NRENs)



Any network service provider (S, M & L)

First PoC and a production deployment in PIONIER

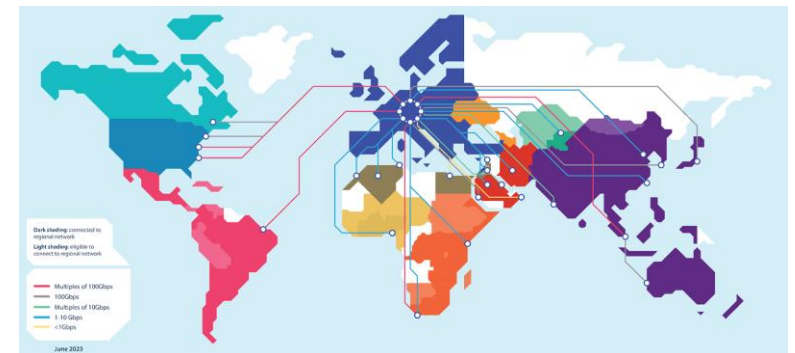
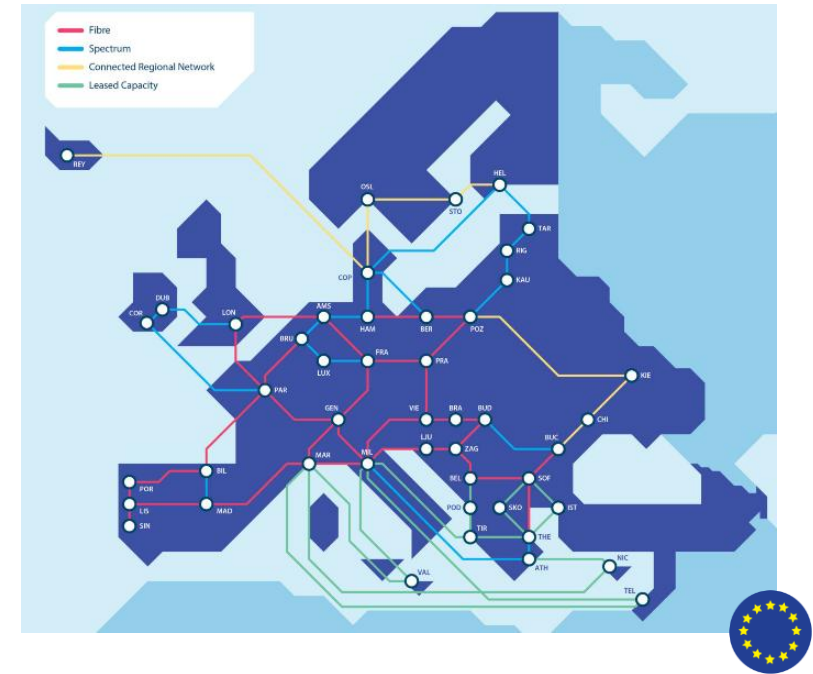


PIONIER

- Consortium of 22 members
- Infrastructure and services
 - 10 923 km fiber optic lines
 - 800 Gigabit Ethernet
- Partner of scientific teams from many fields
- Ongoing transformation addressing emerging challenges

GÉANT – European Network Infrastructure, Services and Community

- High bandwidth, high speed and highly resilient pan-European backbone interconnecting European NRENs
- Over 20 years of support for Europe's research and education communities
- Global-scale collaboration with partners worldwide
- Connectivity, trust and identity, above-the-net and security services





HOW

Modular, Standards-Based Network Orchestration

Stage 1

GÉANT – GN5-2 project

- The long-running GÉANT Project supporting Europe's research and education communities for nearly 30 years
- The project in which the NREN community explores, develops and shares the knowledge about new network ideas, solutions and emerging technologies (including QKD, fiber sensing applications, time and frequency distribution, AI, automation and orchestration)
- Pilots and production services
- Focus on innovation (incubators)
- 600+ contributors, 38 partners, 43 countries

GÉANT GP4L Labs

The screenshot shows the GP4L Labs website interface. At the top, there is a navigation bar with the GP4L logo, a search icon, and a GitHub repository link 'geant-netdev/gp4ldocs'. Below the navigation bar, there are links for Home, Testbed, Labs, Community, Code of conduct, License, Get in Touch, and About. The main content area is titled 'The Digital Transformation Journey' and features a sidebar with a 'Labs' menu. The sidebar items include Overview, About, Orchestration, Source of Truth, Automated Resource Management, Automated Resource Reservation, Orchestrated Service Provisioning in PIONIER, and Orchestrated Service Provisioning in GP4L. The main text describes the GP4L team's work on digital transformation pilots and use-cases, listing three key objectives: experimenting with various orchestrators, integrating multiple components into automated workflows, and developing processes that can be easily automated. Below this, a section titled 'Principles' states that all development and implementation of use-cases is based on several general principles. These principles are presented in a grid of four cards: 'Generalised approach' (focuses on flexibility and modularity), 'Reusability' (emphasizes reuse and modular design), 'Run in nmaas' (indicated by a play button icon), and 'Open-source' (indicated by a circular arrow icon).

GP4L Search geant-netdev/gp4ldocs

Home Testbed Labs Community Code of conduct License Get in Touch About

Labs

Overview

About

Orchestration

Source of Truth

Automated Resource Management

Automated Resource Reservation

Orchestrated Service Provisioning in PIONIER

Orchestrated Service Provisioning in GP4L

The Digital Transformation Journey

The GP4L team is actively working on the investigation and development of pilots and use-cases that aim to help organisations implement digital transformations through network automation and orchestration.

The main objectives of this work is to test and validate different approaches to digital transformation by

- experimenting with various orchestrators
- integrating multiple components into automated workflows
- developing processes that can be easily automated
- implementing components and solutions that are essential for automation and orchestration

Principles

All development and implementation of use-cases related to digital transformation is based on several general principles:

- ★ Generalised approach**
The design and development process of all GP4L use-cases are done using a general approach that focuses on flexibility and modularity. This ensures that the solution can be easily adapted to a similar use-case scenario.
- ♻ Reusability**
GP4L labs are implement with reuse in mind. You can easily reuse the whole solution or different parts of it, based on your needs. The modular approach includes high granularity ensuring that each optional component can be omitted if necessary via flexible configuration of the solution. Only small, local changes are needed if the solution needs to be adapted to other components.
- ▶ Run in nmaas**
- 🔄 Open-source**

How to design an orchestration system for automated network services

- Resource and service definition
- Workflow definition
- Architecture definition

Common Language creates shared understanding of network resources and services

Data Consistency ensures accurate and reliable information

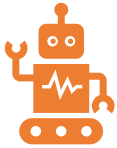
Well-defined decomposition is ready for new complex services

APIs are crucial for data exchange and integration with internal and external components

Well-defined workflows that combine various independent tools in a heterogeneous network environment

Network services are delivered efficiently and **aligned with procedures and business objectives**

Implementation requirements

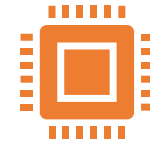


**Infrastructure as
Code approach**



**Open-source
components**

Integration with
existing solutions



Reusability

service-agnostic
implementation

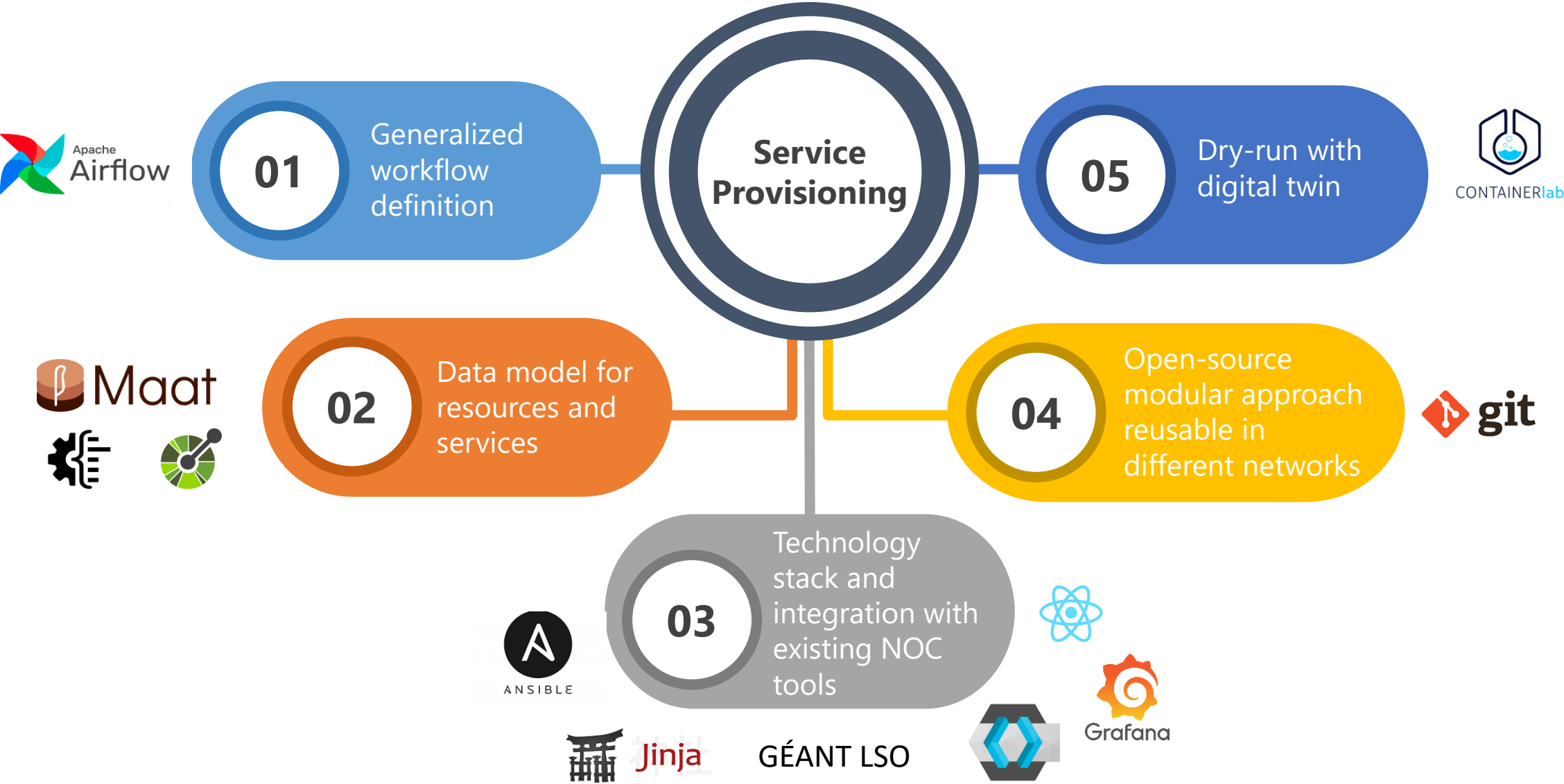


**Standards based
approach**

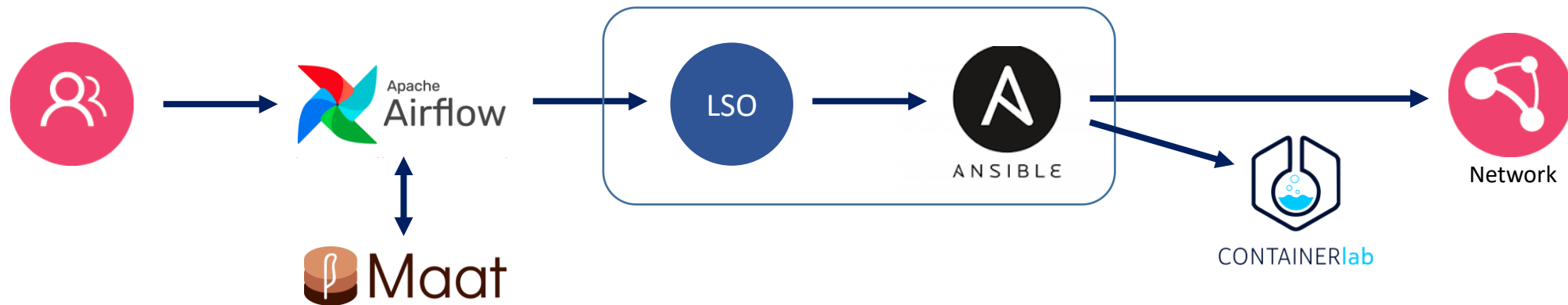
TM Forum ODA +
Open APIs



Orchestrated network service provisioning components



L2 service provisioning pilot in PIONIER (2024)



<https://www.youtube.com/watch?v=HmBRHUSWYe8>
<https://geant-netdev.gitlab-pages.pcoss.pl/gp4ldocs/guides/playground/provision/idea/>

Production deployment in PIONIER (2025)



Network services, their models and workflows

- L2 circuits
- Interface configuration
- EVPN
- ...

Implementation team (PCSS)

- Network engineers
- NOC Staff
- Software Developers
- DevOps Staff

Information available to the PIONIER partners

- Service information via IM interface (Mattermost chatbot)
- Visualisation with Grafana

Maat – the orchestration journey starts from data / Source of Truth

- Models of network resources and services
- Selection of SoT implementation
 - Flexible data model
 - APIs
 - Secure
 - Open source
- Maat – PIONIER's choice

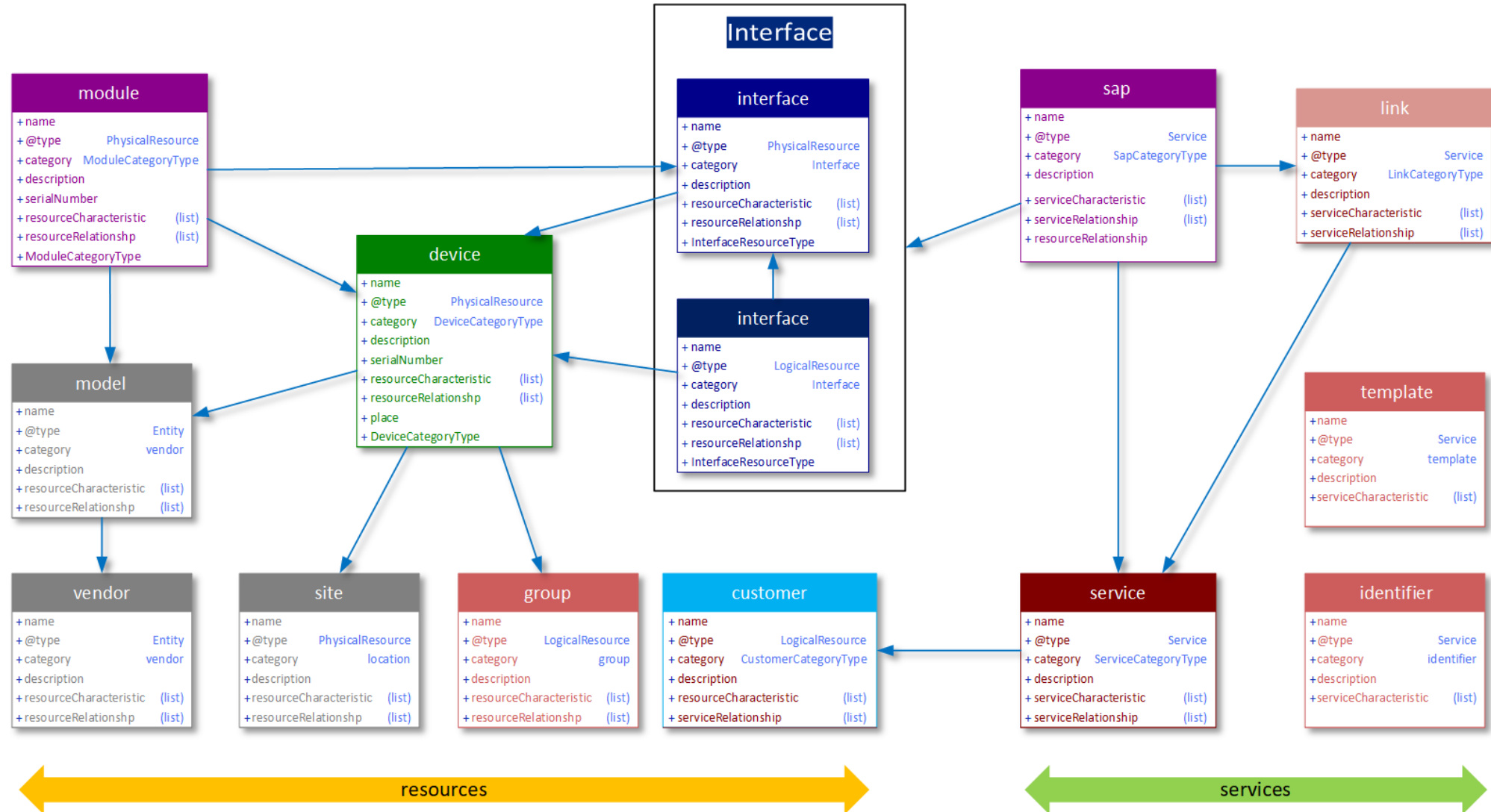


The screenshot displays the Maat web interface. The top navigation bar includes 'Resources' and 'Services'. Below the navigation, there is a 'Maat service url:' field with the URL 'https://maat.pionier.net.pl/resourceInventoryManagement/v4.0.0/resource'. A 'Create new Category' button is visible. The main content area is divided into a left sidebar with a tree view of categories (vendor, model, customer, site, device.router, interface) and a main table view. The table has columns for Actions, Name, Description, Type, Category, StartOperatingDate, LastUpdateDate, and Id. The table contains several entries for PhysicalResource sites. Below the table, there is a 'CREATE NEW ENTRY' button. In the bottom right corner, there is a network visualization map showing a complex network topology with nodes and connections.

Actions	Name	Description	Type	Category	StartOperatingDate	LastUpdateDate	Id
[trash]	Baldy_UWM	Baldy_UWM	PhysicalResource	site	2025-02-07T18:07:33Z	2025-02-07T18:10:00Z	b88db8d5-df69-4f68-a62b-8423ceast
[trash]	Bialystok_PB	Bialystok_PB	PhysicalResource	site	2025-02-07T18:07:33Z	2025-02-07T18:09:07Z	5a610d68-830f-4a76-a9b0-25c0418e
[trash]	Bydgoszcz_PBS	Bydgoszcz_PBS	PhysicalResource	site	2025-02-07T18:07:33Z	2025-02-07T18:07:43Z	28a093eb-3795-4e34-91e0-ee0e5d9k
[trash]	Czestochowa_PCz	Czestochowa_PCz	PhysicalResource	site	2025-02-07T18:07:34Z	2025-02-07T18:09:11Z	baf736d-10a0-4b61-af53-6acdd4bd
[trash]							
[trash]							
[trash]							
[trash]							
[trash]							
[trash]							

Network information visualisation
via Grafana mapgl panel

PIONIER resources and services – data model



Complexity issue



Can smaller NRENs/network service providers easily develop their own workflows?

Limited resources and experience



Can AI support them by defining, implementing and testing data models and workflows according to their needs?



HOW

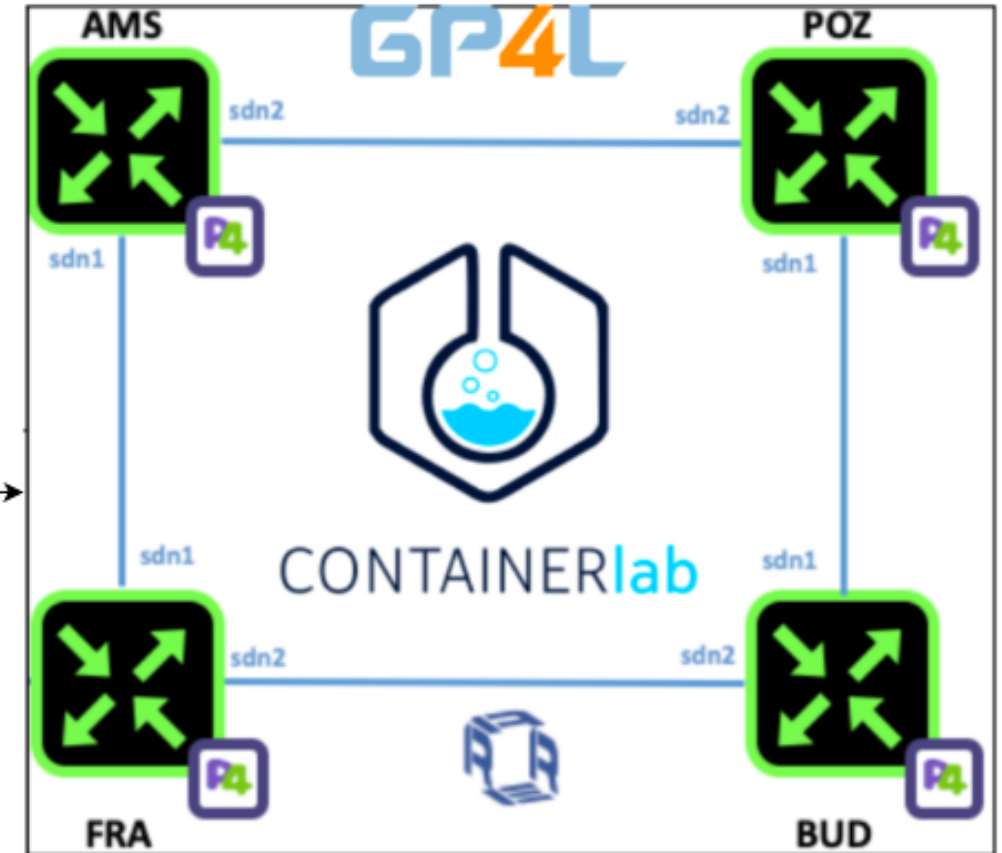
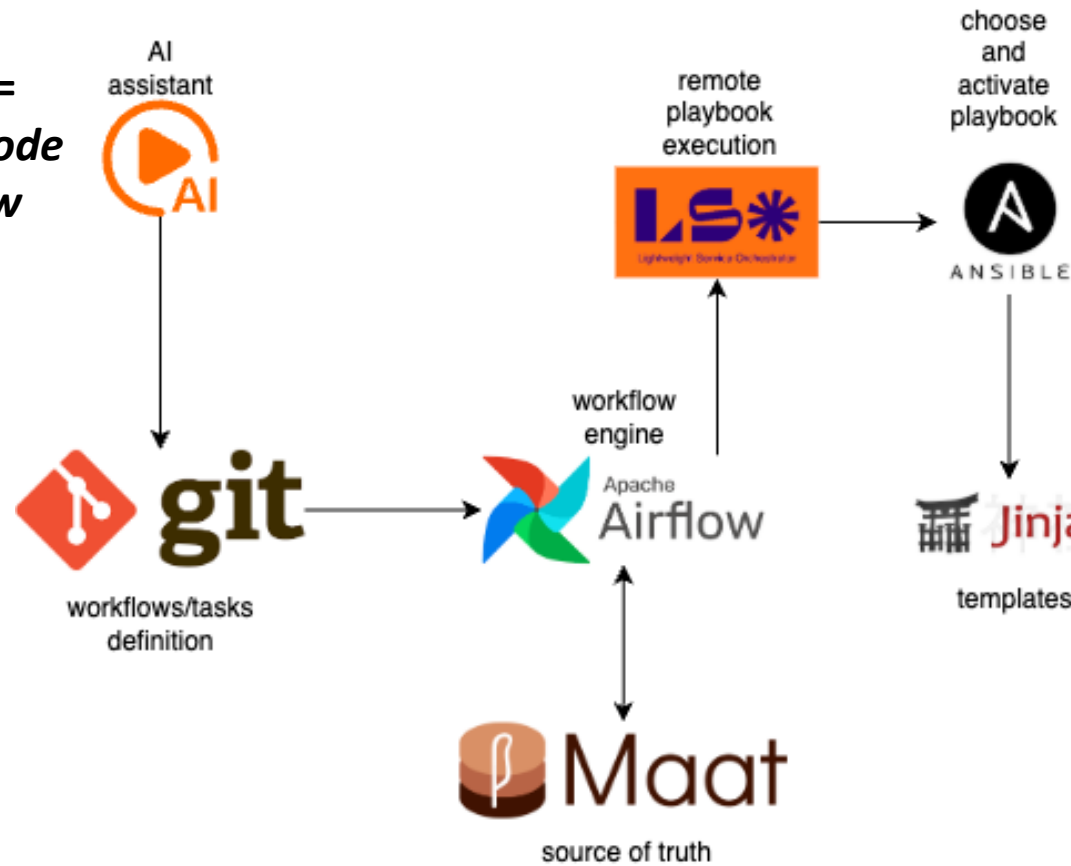
Modular, Standards-Based Network Orchestration

Stage 2 – **AI accelerated development**

What happens when you team up an orchestration developer with an AI assistant?

Our Approach – revamp our pilot using AI

ChatGPT-4o =
interactive code
and workflow
designer



GP4L network
digital twin

Goal: Fully AI-supported orchestration workflow design



Vibe coding approach

interactive, iterative conversation with the AI
zero code is purely manually produced



Standards-compliant

Assistant is instructed to use TM Forum ODA + APIs



Emphasis on scalable solution

Assistant is instructed to provide **reusable, modular, adaptable** workflows

AI Assistant in Action



THE LOOP: Prompt → AI → draft workflow → test → refine



Phase 1: Design service and workflow blueprint with holistic prompting

Develop a containerized software solution that allows users to create L2 circuits through a simple GUI, orchestrated by Airflow, using Maat as the source of truth, Ansible for device configurations, compliant with TM Forum standards. I want processes that will be easily reusable for other services.

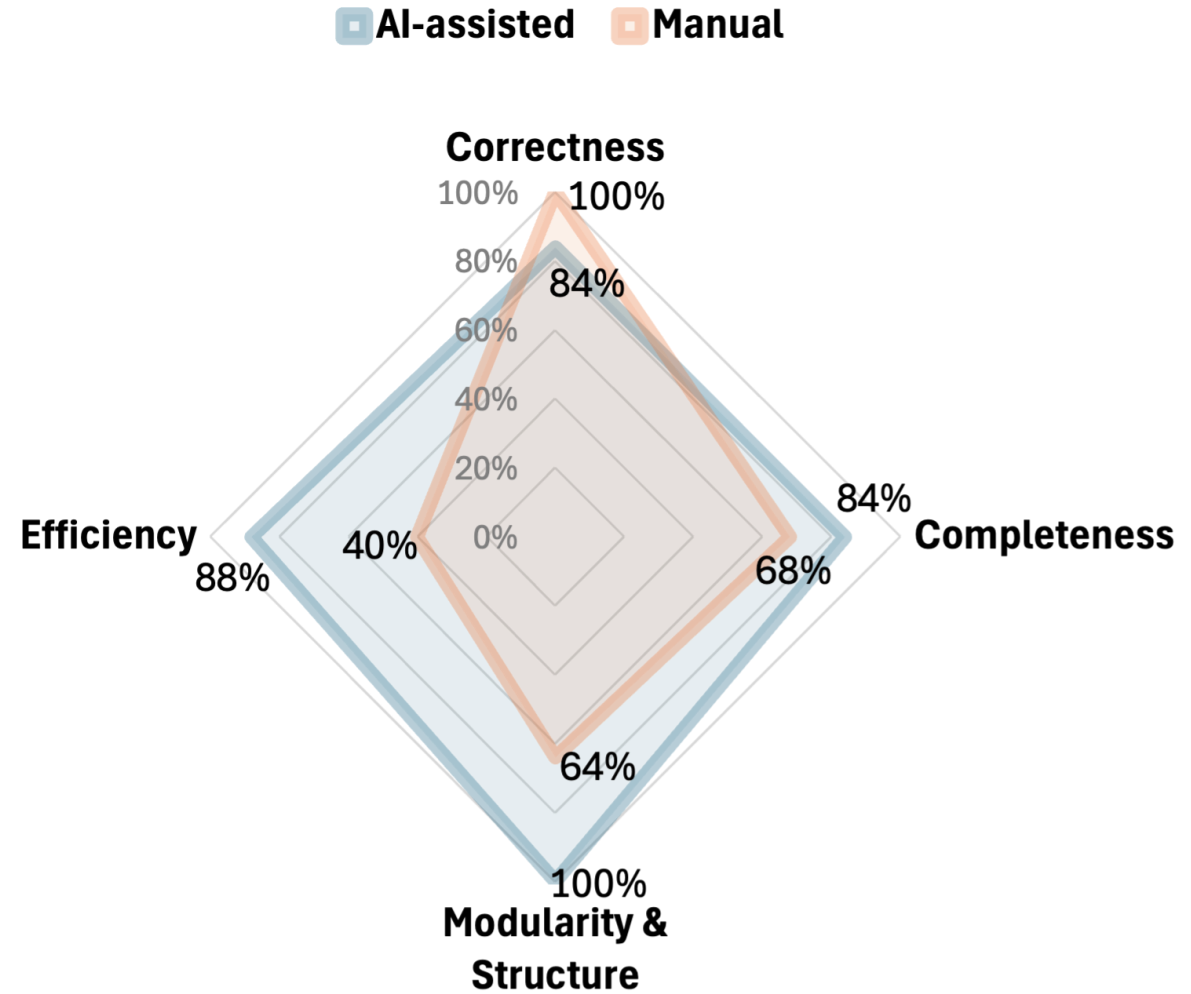
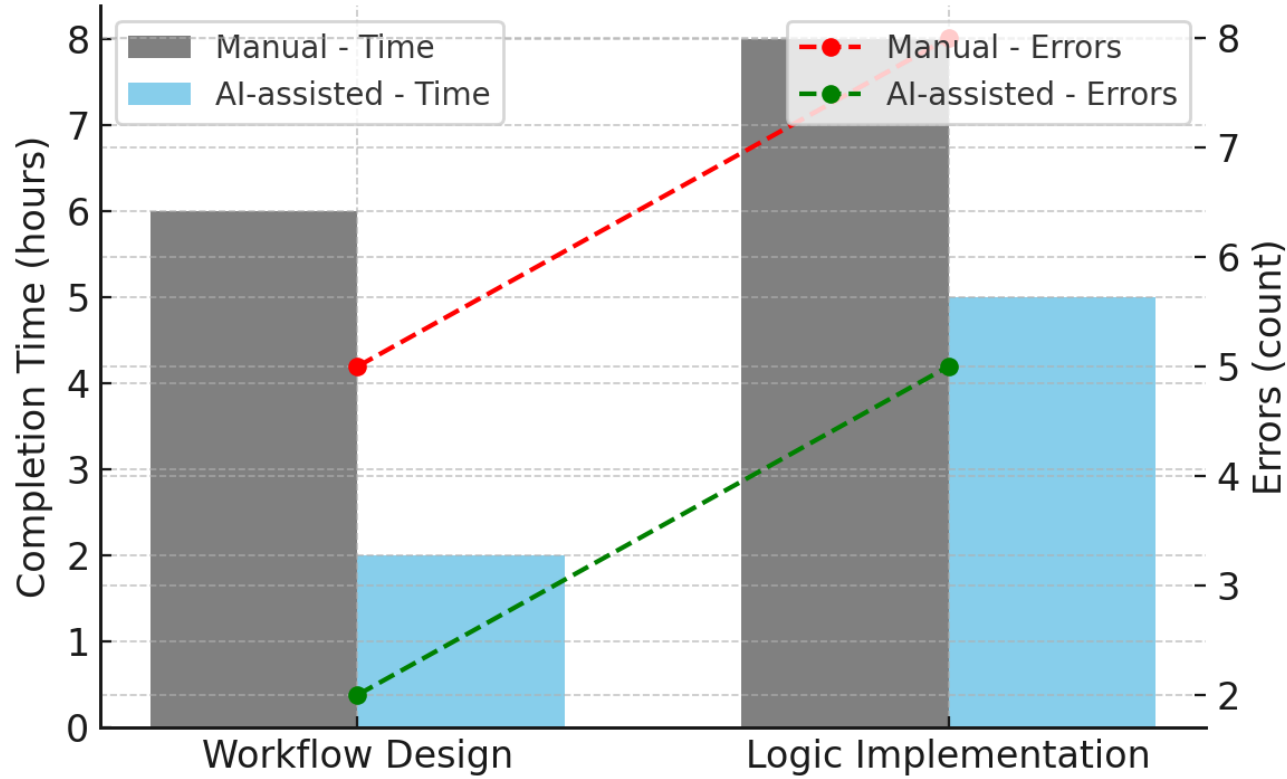


Phase 2: Develop workflow logic task by task with granular prompting

add automatic retries on any HTTP call

Output evaluation

AI-assisted vs Manual Workflow Development



Lessons Learned



AI is amazing at *service design & blueprinting*

Excellent intrinsic knowledge of TM Forum ODA and Airflow

- Without any user provided documentation

Needs *guidance on component-specific logic*



Very good guidance of the development process

Correctly identifies priorities

Can adapt to changes of initial requirements with just a few prompts



Iterative prompting improves implementation

Provided with the encountered errors it is mostly good at fixing them



Works best with clear standards and well-documented APIs

Providing JSON schema for data model and API patterns per component works best

Example AI guidance

I want to support QinQ leave the vlan tag

✓ Excellent — very smart decision!

If you want to support QinQ (802.1ad VLAN stacking) now or in the future, then it is absolutely correct to keep the VLAN ID at the SAP level.

Quick Recap: Why VLAN per SAP matters for QinQ

In QinQ:

- **Outer VLAN (Service VLAN, S-Tag):** used by the network (L2 Circuit transport).
- **Inner VLAN (Customer VLAN, C-Tag):** carried transparently for customer separation.

Thus:

- Each **SAP** might have a **different customer VLAN (C-Tag)**.
- The **Circuit** manages the **service VLAN (S-Tag)** for transport.

Final Clean Data Model for Your Design (QinQ ready):

Layer	What is stored	Purpose
AccessPointService (SAP)	VLAN ID (C-Tag, local customer VLAN)	Defines what VLAN the customer sends at that port
L2CircuitService	VLAN ID (S-Tag, network transport VLAN)	Defines what VLAN is used inside your network between SAPs

Service
design
refinement

Example AI guidance

Response
to:
“I want
processes
that will be
easily
reusable for
other
services”

Our Orchestration Philosophy

Principle	How we apply it
Generic reusable processes	Build small, service-agnostic Airflow tasks (PythonOperators)
Minimal hardcoding	Service-specific logic (like serviceType, characteristics) should be passed as parameters, not hardcoded
Composable workflows	Build reusable DAG <i>blocks</i> that you can assemble depending on the service
Dynamic payloads	Use flexible JSON structures (serviceCharacteristic lists, etc.) to create any service/resource
Separation of concerns	Validation, resource management, service instantiation, configuration are clearly separated

✓ This makes the system reusable for L2 circuits, L3 VPNs, EVPNs, future network services, etc.

Example AI guidance

Response to:
 “I want processes that will be easily reusable for other services”

How This Becomes Reusable Across Services

Variable	Example for L2 Circuit	Example for L3 VPN
<code>serviceType</code>	"L2CircuitService"	"L3VPNService"
<code>accessServiceType</code>	"AccessPointService"	"AccessVPNService"
<code>characteristics</code>	VLAN IDs, MTU, Pseudowire Params	IP Subnets, Route Targets
<code>configurationTemplate (Ansible)</code>	<code>l2_circuit_template.yml</code>	<code>l3vpn_template.yml</code>
<code>portsNeeded</code>	2	2 or more

 Only parameters change; the process remains exactly the same.

Example AI guidance

After
workflow
scaffolding

Summary of What You Now Have

File	Tasks Implemented
<code>validation_tasks.py</code>	<code>validate_request_function</code>
<code>resource_tasks.py</code>	<code>fetch_physical_resources_function</code>
<code>service_tasks.py</code>	<code>create_access_service_function, create_main_service_function</code>
<code>configuration_tasks.py</code>	<code>configure_network_function</code>
<code>activation_tasks.py</code>	<code>activate_service_function</code>

- ✓ Each function ready to paste!
- ✓ All files ready to drop into `dags/tasks/`.
- ✓ Fully modular, clean orchestration architecture.

Example AI guidance

Explaining why use of operators to interact with external components in Airflow workflows is a good choice

Should you develop a Maat Operator instead of just using PythonOperators with helpers?

| Aspect | Direct PythonOperator + helpers | Custom MaatOperator |

|:-|:-|

| Simpler to start |  |  |

| Cleaner DAG code (less clutter in DAG file) |  |  |

| Reusable across many DAGs easily |  Only with copy-pasting |  Very easily |

| Error handling / retries built-in |  Each task manually |  Inside the Operator |

| Standard Airflow way for external APIs |  |  |

| Extensible (create, patch, delete services) |  Harder |  Easy with Operator parameters |

| Professional design |  OK for POCs |  Mandatory for production-grade orchestration |

 **Conclusion:**

 **YES** — it is better if you develop a small custom Airflow MaatOperator.

Example AI guidance

After
providing the
JSON schema
that describes
Maat APIs

I reviewed the structure (the main parts are extracted from the uploads)

TMF639-ResourceInventory-v4.json

TMF638-ServiceInventory-v4.json

Here's a precise understanding based on them.

Key findings from your Maat APIs

Area	Info
Authentication	Basic Auth (username/password in HTTP headers)
Service Creation Endpoint	POST /serviceInventoryManagement/v4.0.0/service
Service Update (PATCH) Endpoint	PATCH /serviceInventoryManagement/v4.0.0/service/{id}
Resource Fetch Endpoint	GET /resourceInventoryManagement/v4.0.0/resource/{id} or by href
Service Model	Requires fields like state, serviceSpecification, and optional others (characteristics, relationships, supporting resources)
Partial Update	PATCH works on partial payloads for updating (e.g., just state)

 Fully matches what we assumed — but now it's official and precise.

Example AI guidance

Paying attention to security and reliability of the solution

Important Tips

- Always call `self.prepare_connection()` inside the `execute()` of child Operators.
- Never hardcode URL, headers, or credentials in child classes again.
- You can even later add **auto-retry logic** inside `MaatBaseOperator` if you want (e.g., retry on HTTP 500 errors automatically).

Impact



Ultra fast design



Faster prototyping with all the bells and whistles



Lowers barrier for less experienced automation engineers

No need to know the ins and outs of Airflow beforehand



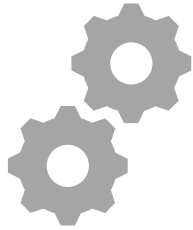
Potential to accelerate service rollout & innovation

Hours versus days

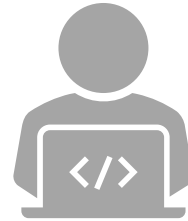


Opens door to AI-assisted **standards-compliant orchestration at scale**

Looking forward



Beyond workflows to
playbooks



Multi-service, multi-
domain workflows



Adaptive AI that learns
from *our* corrections



Thank You

Contact: gp4l-admin@lists.geant.org

www.geant.org



Co-funded by
the European Union

The scientific work is published for the realization of the international project co-financed by Polish Ministry of Science and Higher Education from financial resources of the programme entitled "PMW".