WORKSHOP GARR 2026

NET MAKERS

Orchestrazione della rete GARR

Matteo Colantonio

GARR

Due strumenti per la gestione a intenti di infra e servizi

workfloworchestrator / **orchestrator-core**

orchestrator-core Public





Production ready Orchestration Framework to manage product lifecycle and workflows. Easy to use, built on top of FastAPI and Pydantic

ConsortiumGARR / **yang2sdk**

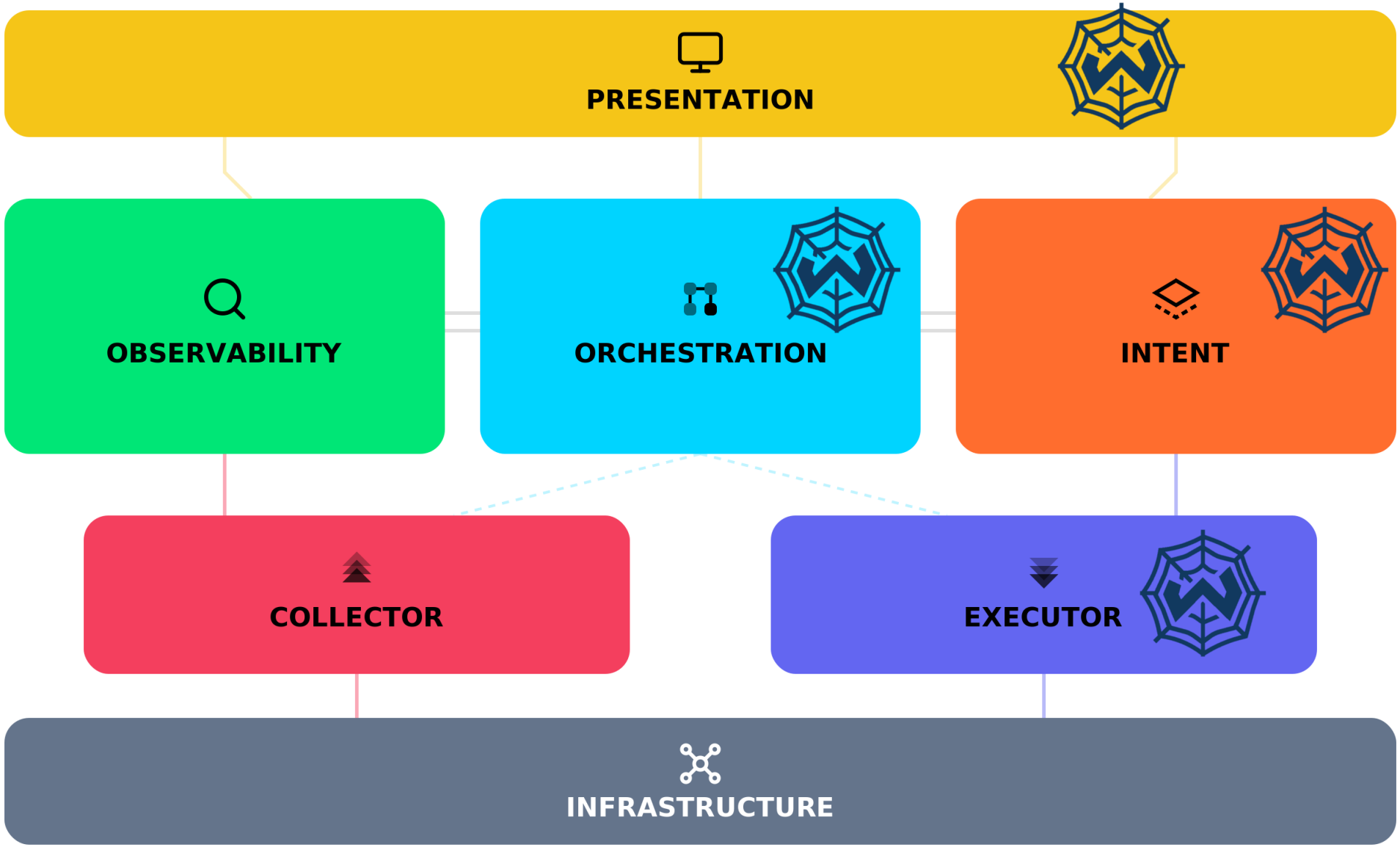
yang2sdk Public

From YANG to a typed Python SDK — without hand-written wrappers.

Point `yang2sdk` at a network device, confirm the modules you care about, and get a Pydantic v2 client that knows the device's real data model: strict models, typed navigators, RPCs, and both RESTCONF and NETCONF transports.

Network Automation Framework

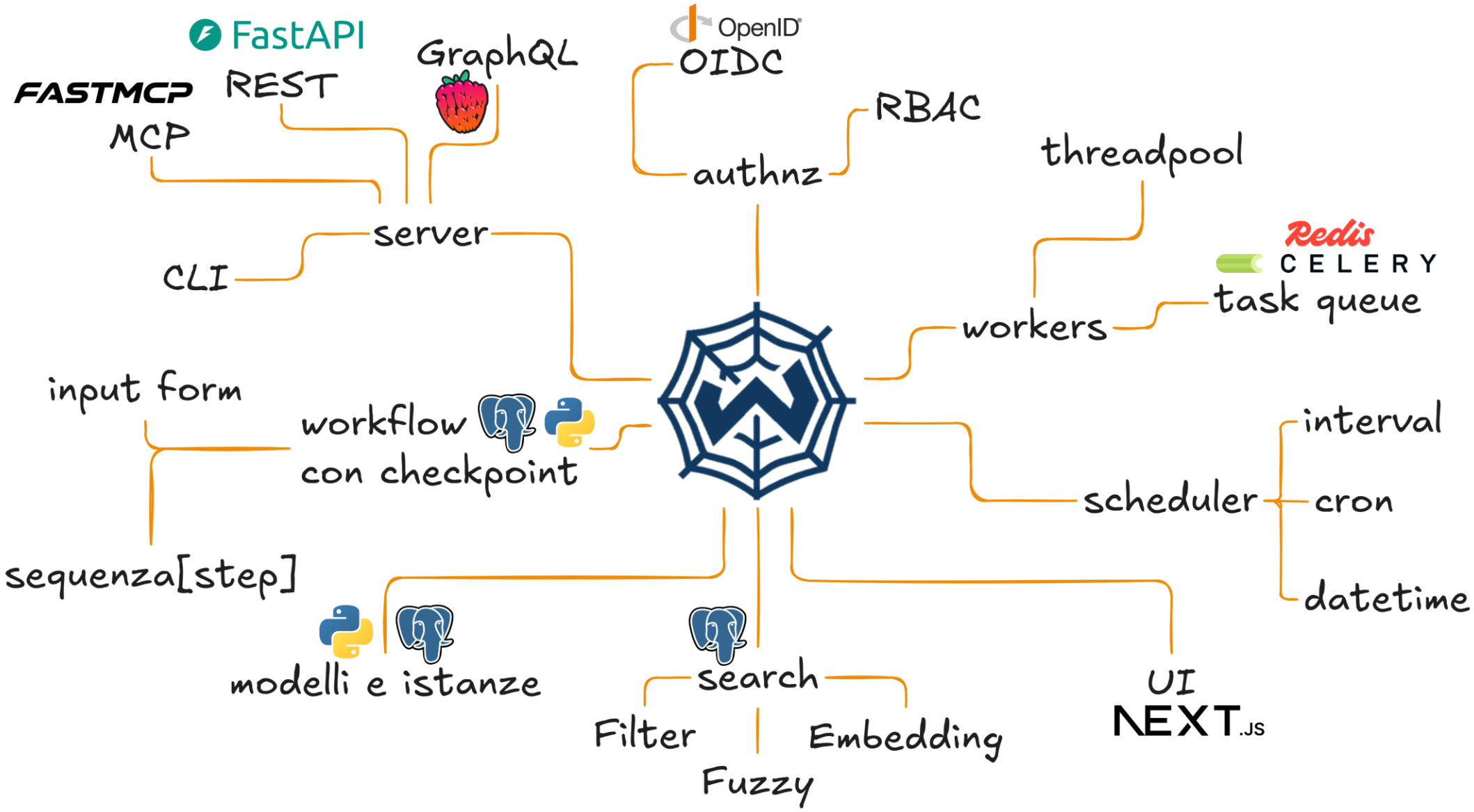
REFERENCE ARCHITECTURE



workfloworchestrator.org



workfloworchestrator.org



È un framework, sta a voi definire:

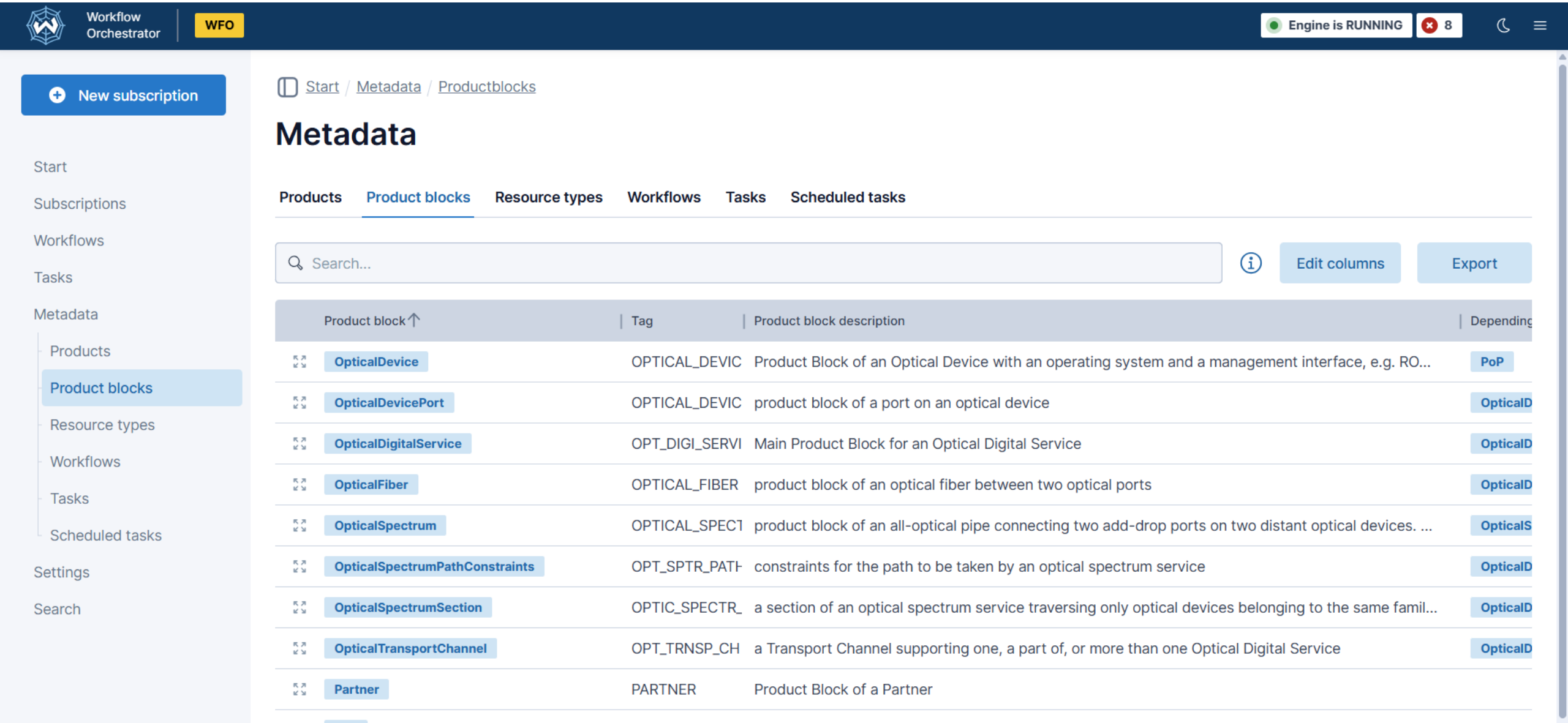
Modelli di dominio / Schema

- Ii controllate voi
- lifecycle: Inactive, Provisioning, Active
- istanze gestite **solo** via wf: create, modify, validate, reconcile, terminate

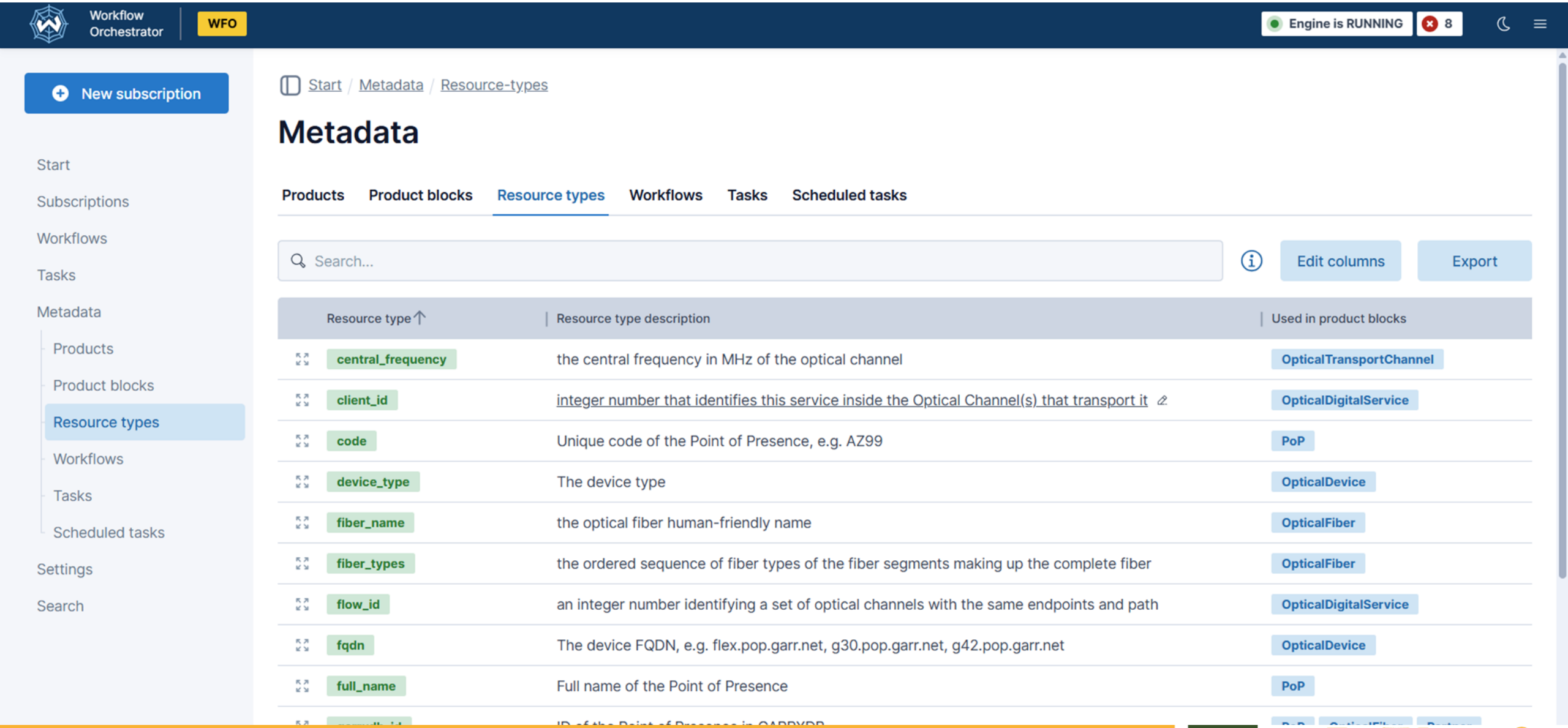
Workflow

- sequenza di step (funzioni Python)
- a cricchetto: se step va in errore, riprova da ultimo step concluso. Indietro mai.
- ‘task’ = wf non associati a ciclo di vita di un modello
- input forms = **API / MCP**

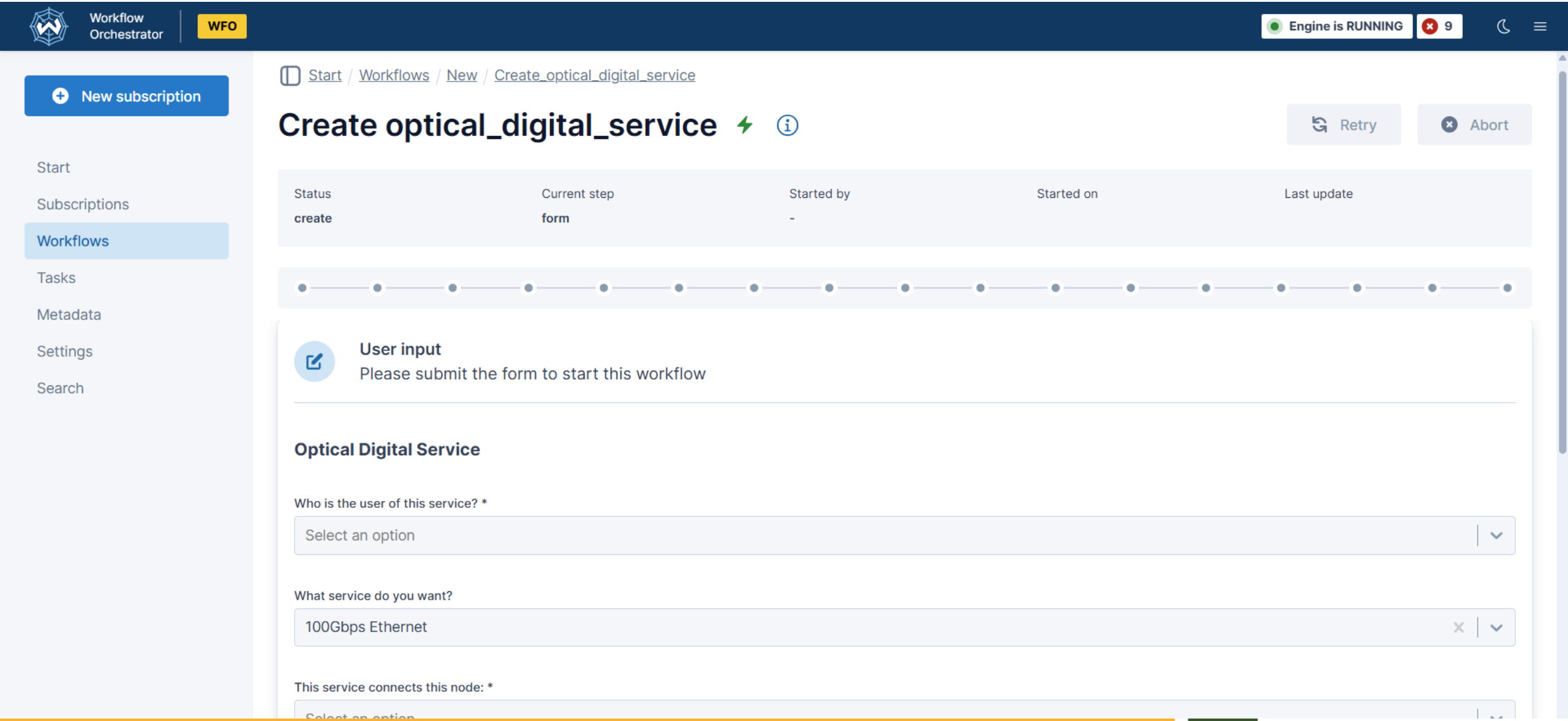
Modelli di dominio



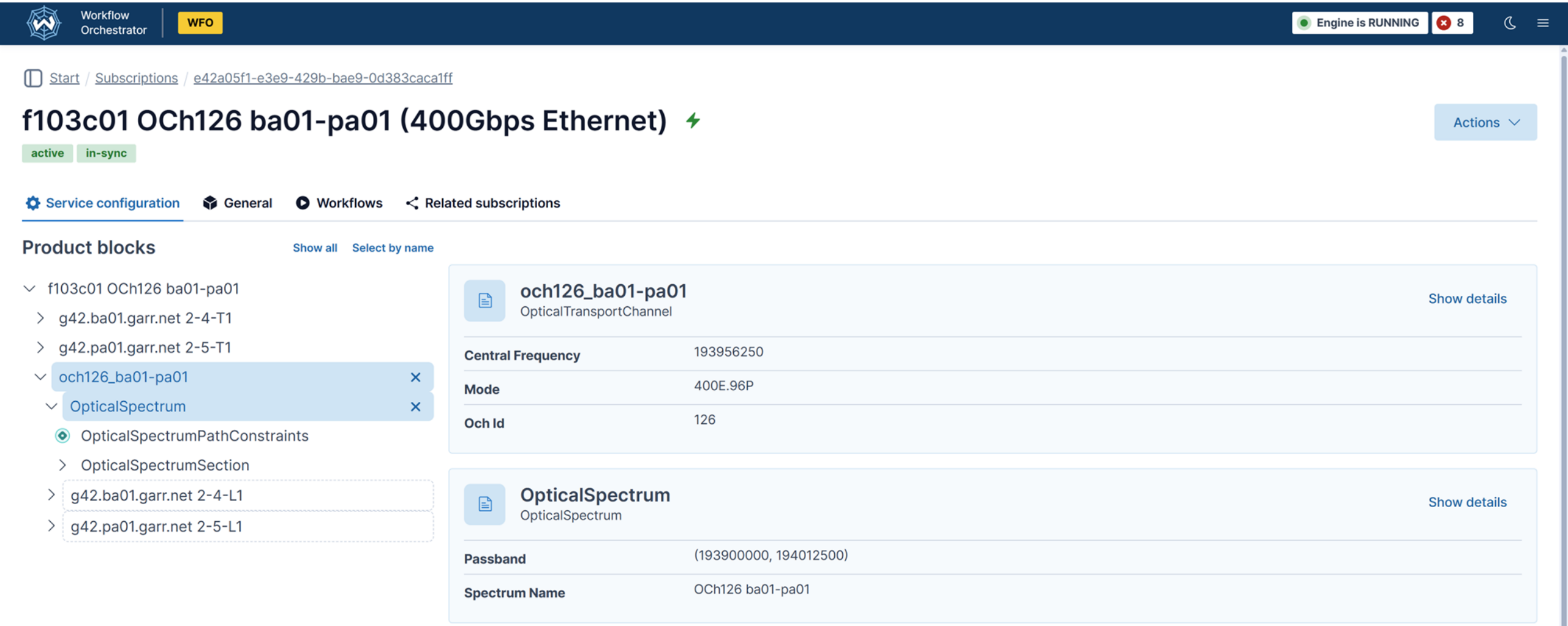
Modelli di dominio



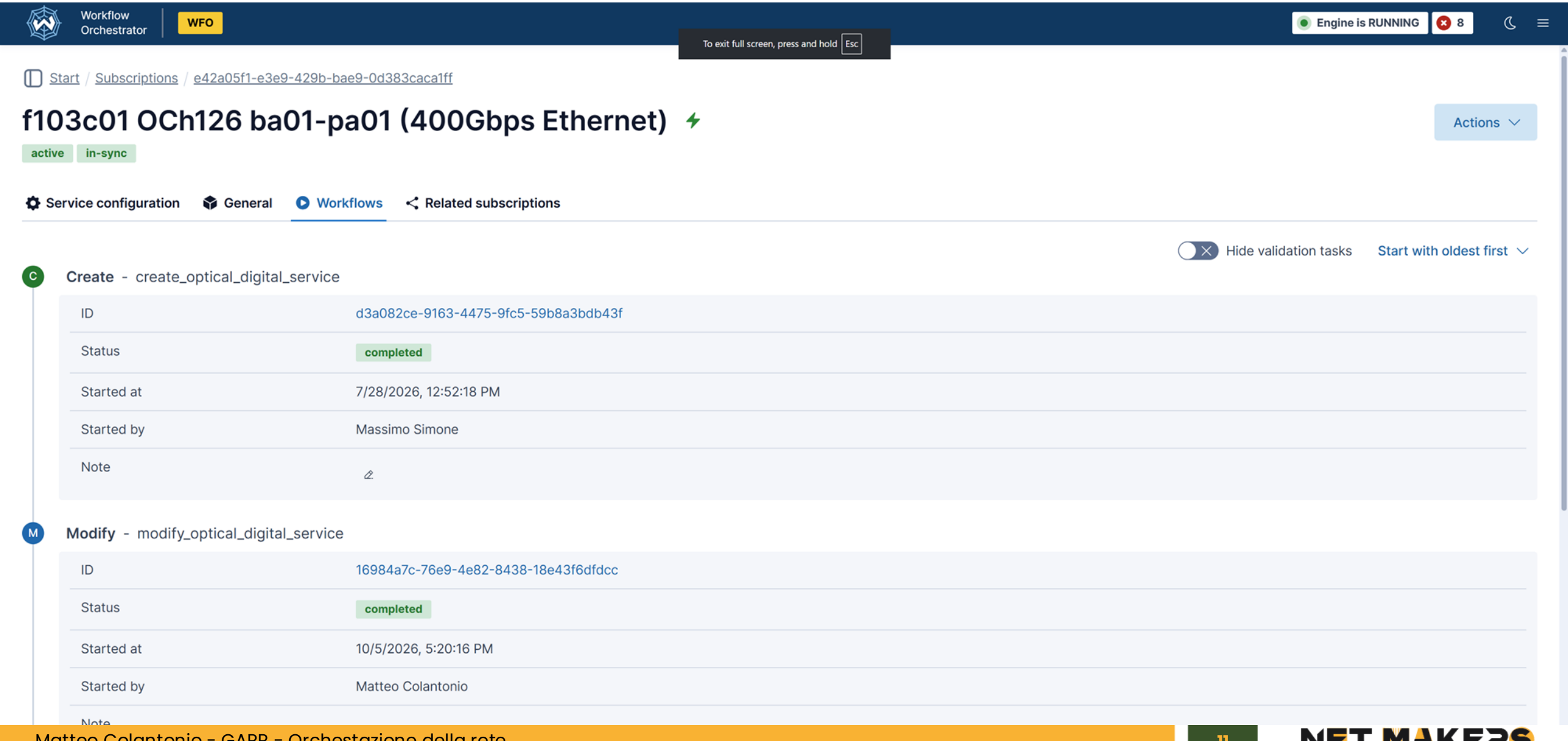
Input form (API workflow)



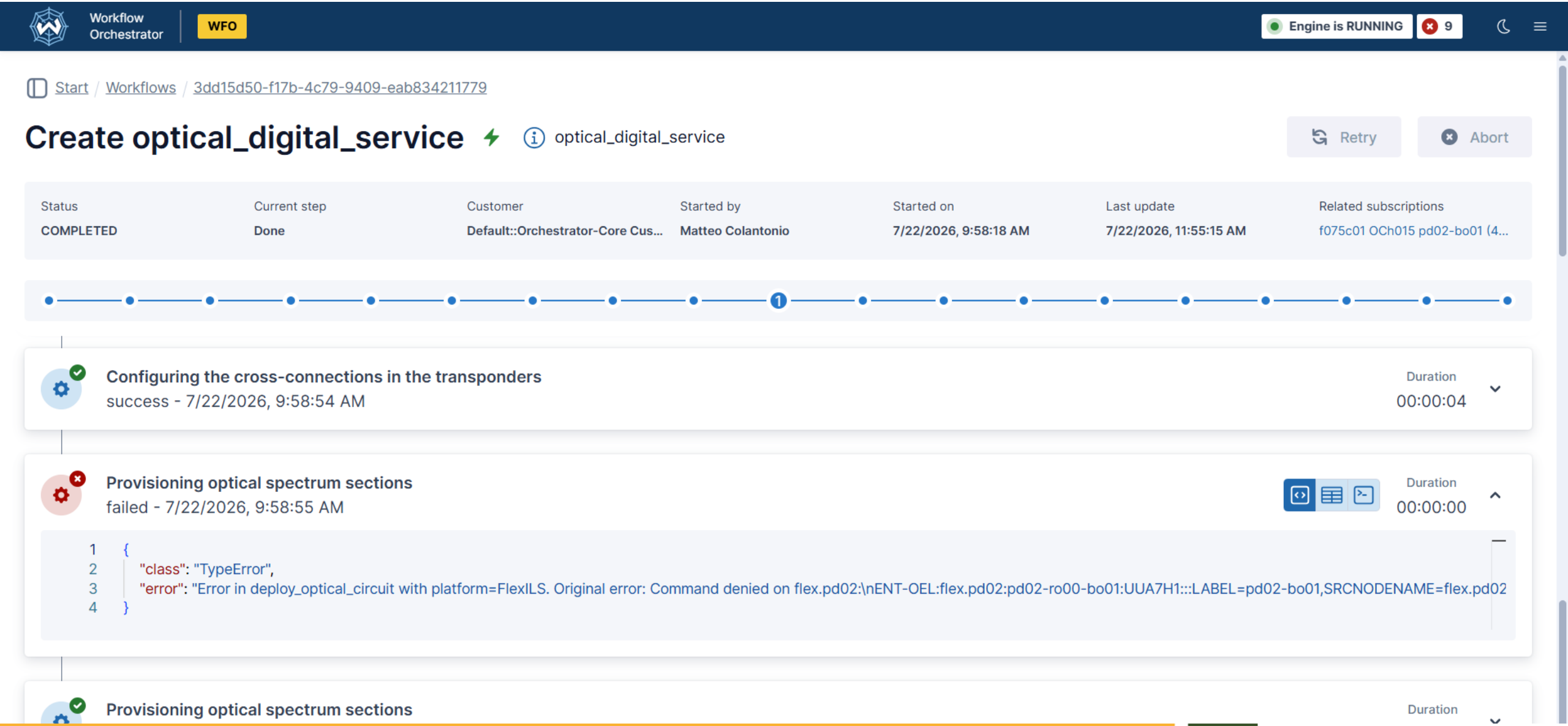
Istanza di modello



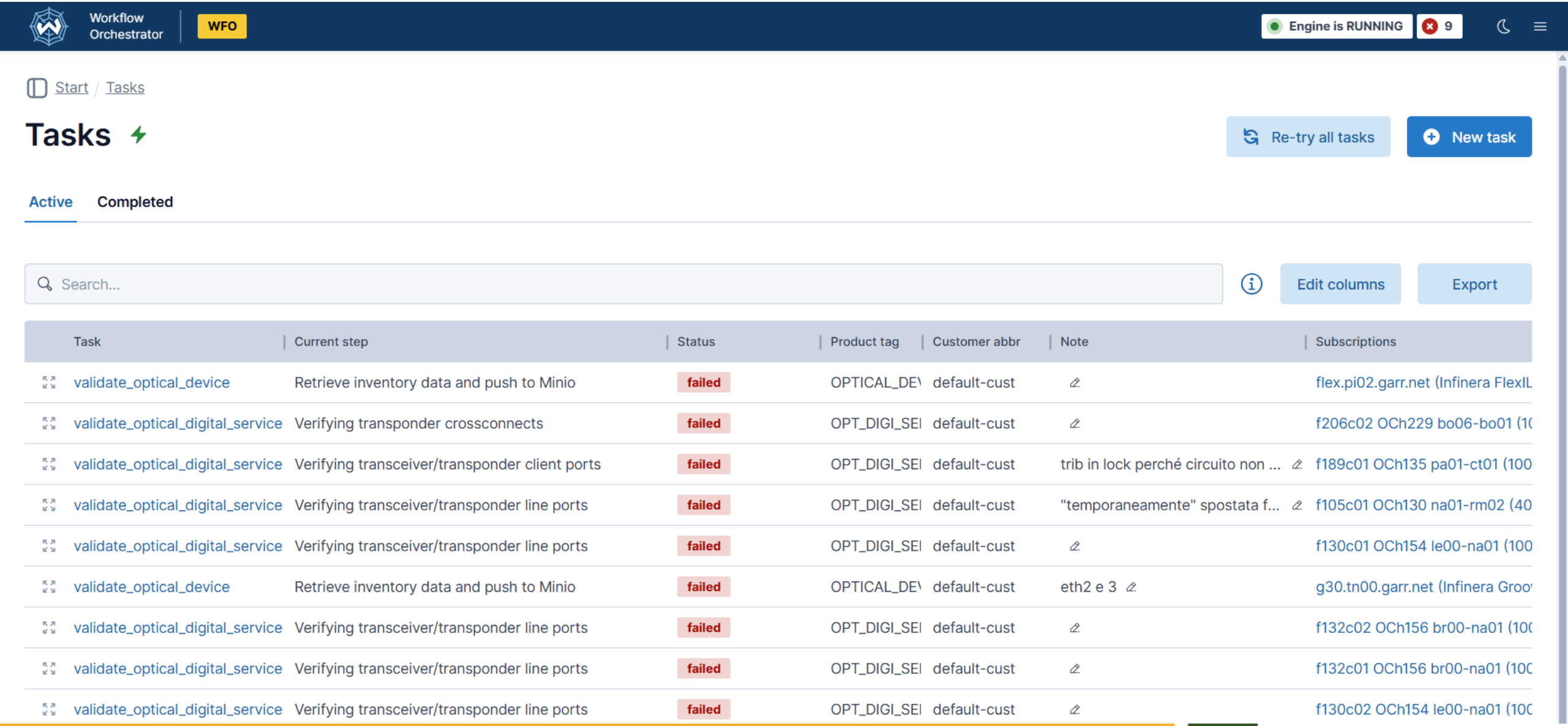
Audit trail dei workflow dell'istanza



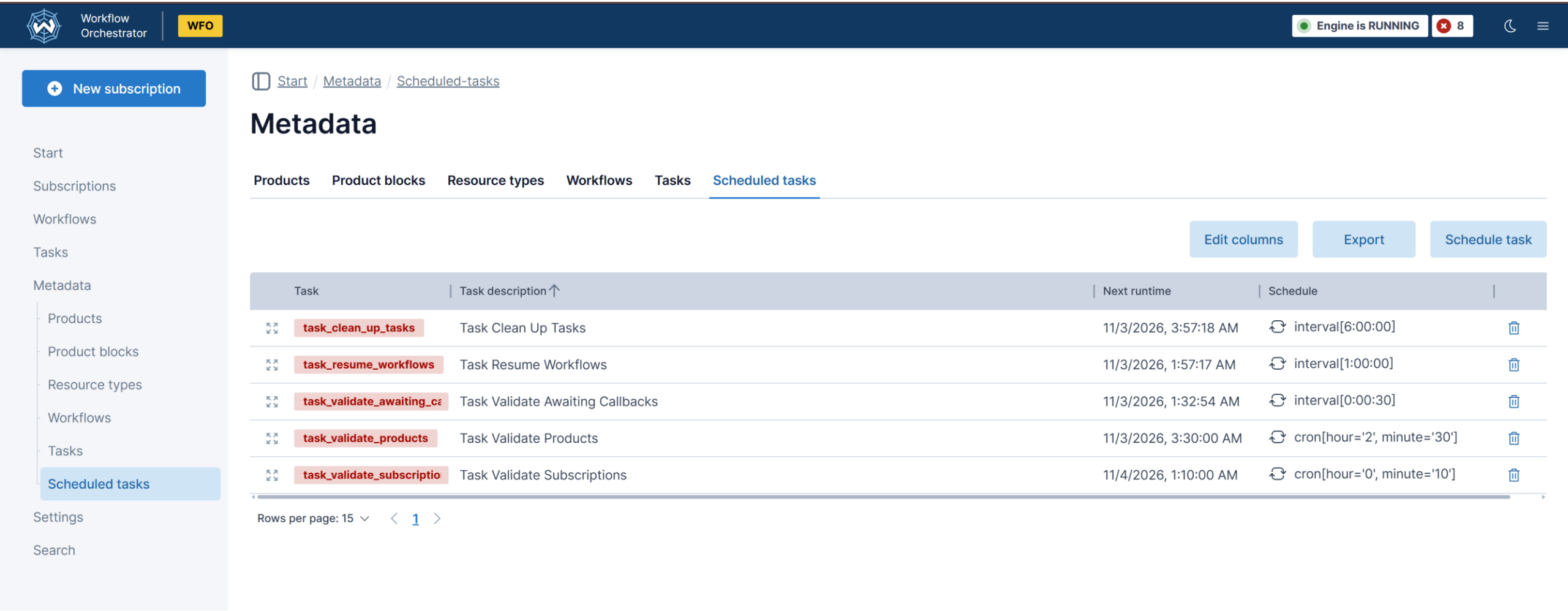
Audit trail di un workflow (esempio checkpoint)



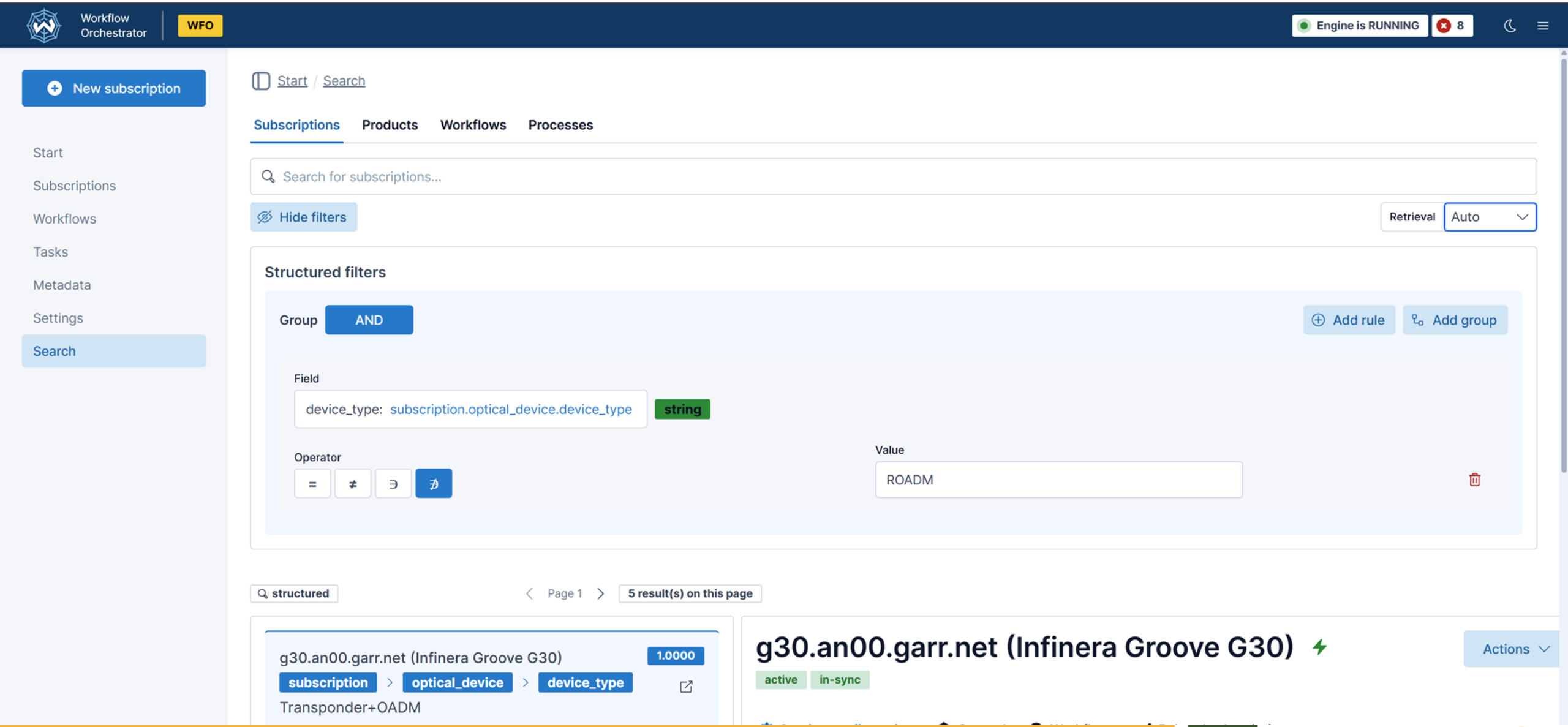
Validate periodici



Scheduler di workflow e task



Search: filter / fuzzy / embedding



Pro

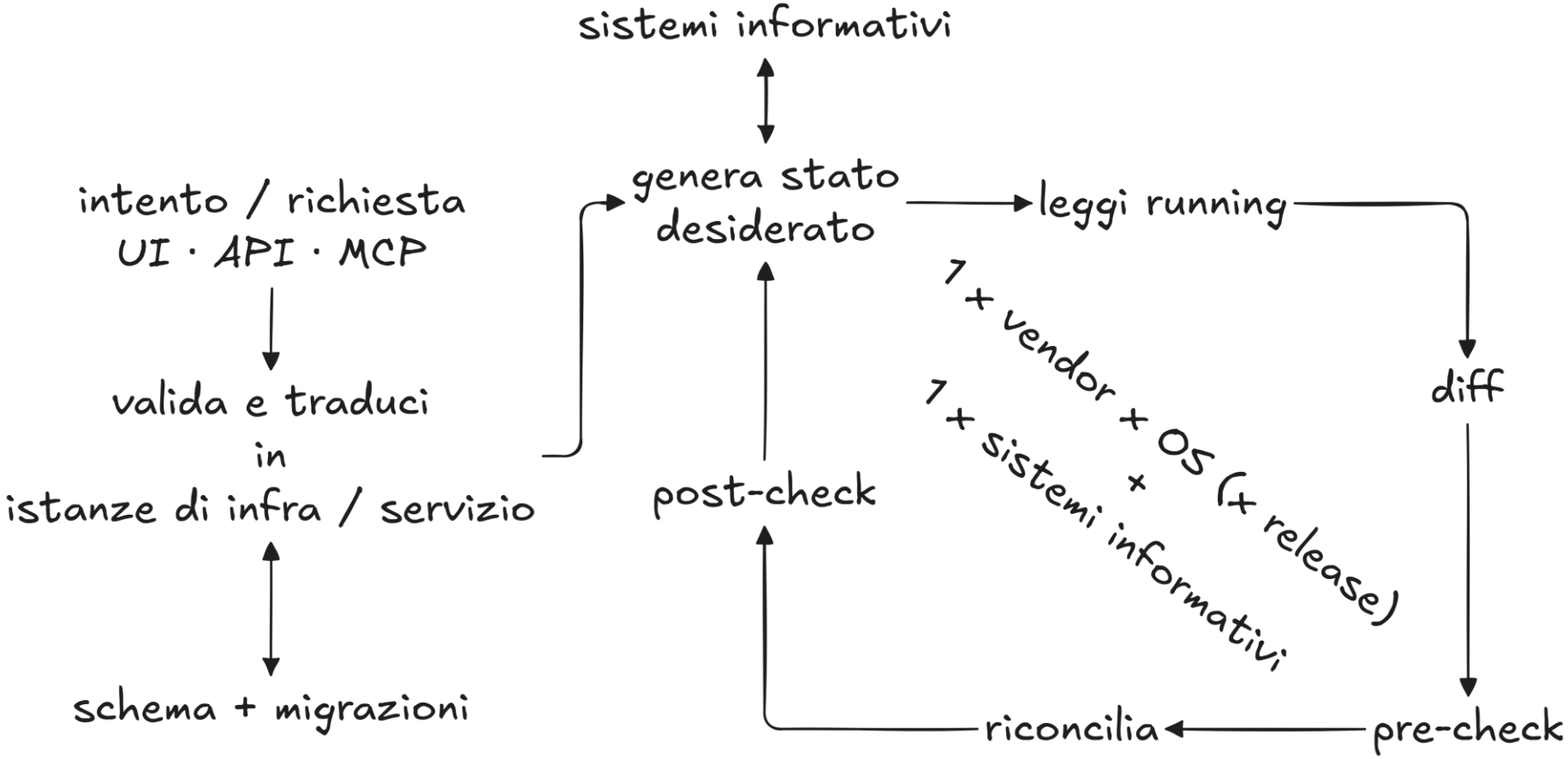
- Python: LLM, anche per input form (UI/API/MCP).
- Libertà: Gestisce qualsiasi procedura codificabile.
- API+MCP: self-service, integrabile, agentificabile.
- open-source: usato in prod da NREN (improbabile che diventi commerciale o venga abbandonato senza successore).
- in crescita: Maintainers molto attivi per bug, miglioramenti e funzionalità.

Contro

- DB: Evoluzione dei modelli e relativo codice → espandi v1+v2, task di migrazione da v1 a v2, termina v1.
- Docs: in miglioramento.
- Effetto LLM: inseguimento upstream.

La cosa più difficile riguarda qualsiasi orchestratore: trasformare le informazioni che oggi vivono nella testa delle persone, nelle config e nei fogli di calcolo e PDF in un sistema a intento eseguibile.

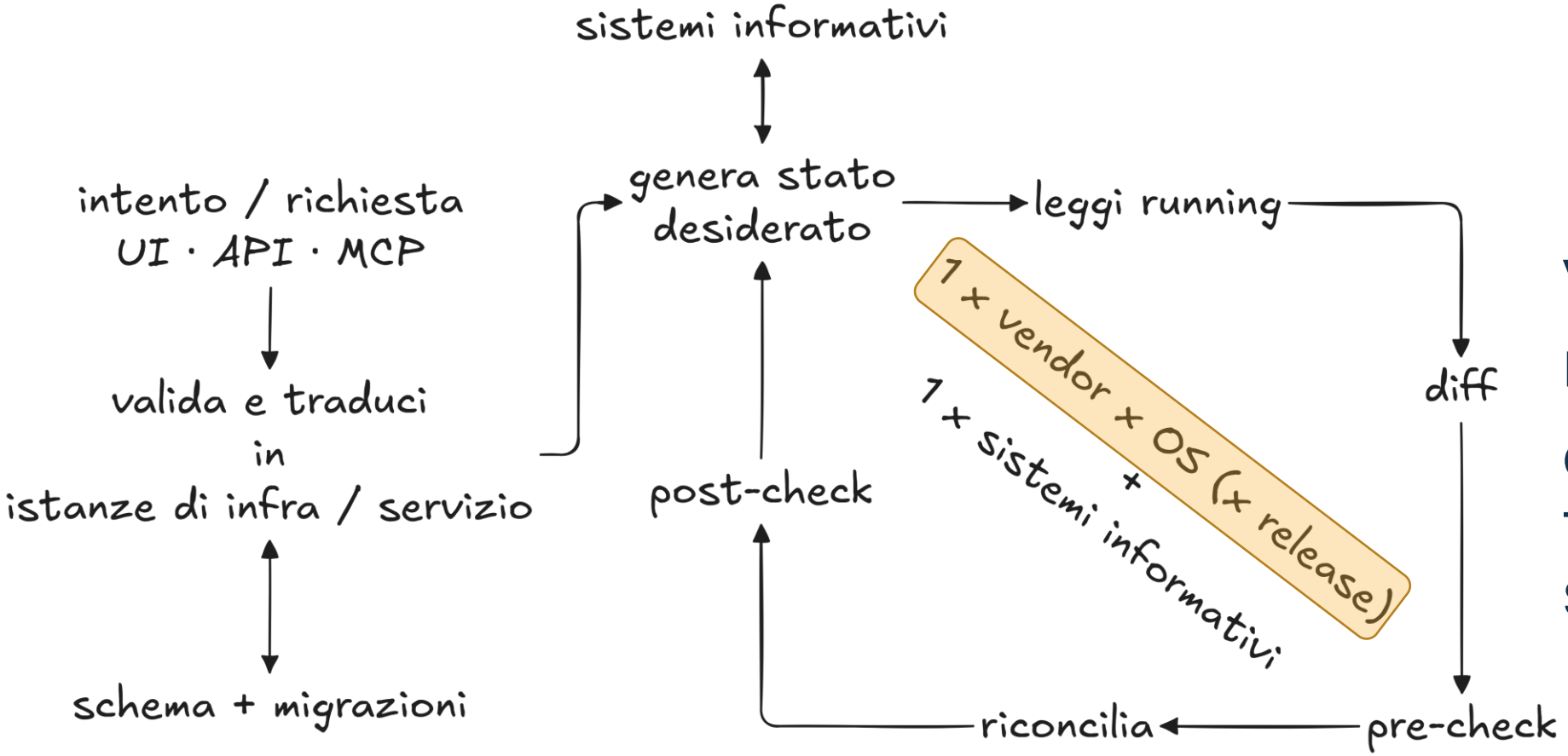
WFO o altro API+DB+worker, alla fine questo è quello che serve



validate → diff

reconcile → ciclo create / modify / terminate → scrivi su DB + reconcile

WFO o altri DB+worker, alla fine questo è quello che serve



validate → diff

reconcile → ciclo create / modify / terminate → scrivi su DB + reconcile

Software Development Kit direttamente dal dispositivo

yang2sdk

From YANG to a typed Python SDK — without hand-written wrappers.

Point `yang2sdk` at a network device, confirm the modules you care about, and get a Pydantic v2 client that knows the device's real data model: strict models, typed navigators, RPCs, and both RESTCONF and NETCONF transports.

```
uv run yang-downloader
uv run yang2sdk --protocol netconf   --device <device>
uv run yang2sdk --protocol restconf --device <device>
```

srl_nokia_interfaces.py

temp > netconf_clients > srl > data_models > srl_nokia_interfaces.py > ...

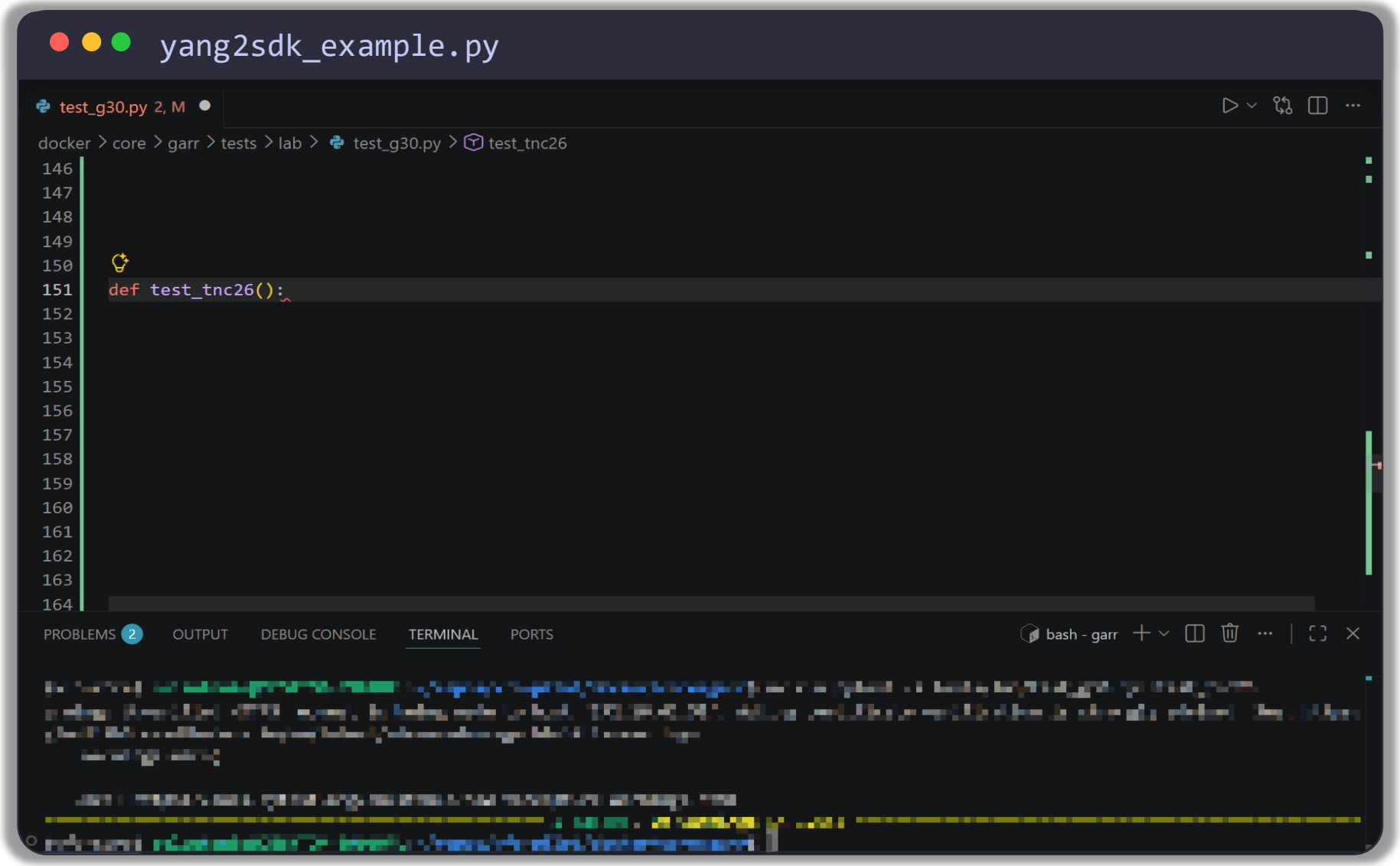
```
1264
1265    class InterfaceItem_1(NetconfBaseModel, tag="interface
1266        """The list of named interfaces on the device"""
1267        name: str = element(tag="name", description="The n
1268        description: str | None = element(tag="description
1269        admin_state: Literal["enable", "disable"] | None =
1270        breakout_mode: BreakoutMode | None = element(tag="
1271        mtu: int | None = element(tag="mtu", default=None,
```

srl_nokia_interfaces.py

temp > netconf_clients > srl > data_navigators > srl_nokia_interfaces.py > ...

```
9   class InterfaceItem_1Node(ItemNode):
10      def retrieve(self, *, source: str | None = None, co
11          from ..data_models.srl_nokia_interfaces import
12
13          xml_filter = build_filter_tree(self._path, Inte
14
15          xml_data = self._client.retrieve(source=source,
```

CRUD ops chirurgiche …



- Path autocomplete
- Type-safety & runtime validation. Fail locally.
- CRUD methods on datastore + RPC operations.

… o datastore completo (KV dichiarativo)

```python
config = client.data.retrieve(
    content="config",
    depth="unbounded",
    fields=_FIELDS,
)

instances = config.model_dump_instances("/", exclude_unset=True)

NetconfClient.diff_instances(running, load_intended(hostname))
```

```
{
  "+": {},
  "~": {
    "/ne:ne/shelf[shelf-id='1']/alias-name": ["shelf-1", "geronimo"]
  },
  "-": {}
}
```

```python
client.reconcile(diff)
```

È tutto! Domande?

deepwiki orchestrator-core

deepwiki yang2sdk



matteo.colantonio@garr.it

WORKSHOP GARR 2026

NET MAKERS