Service Automation Framework

Prem Sankar Gopannan, Lumina Networks

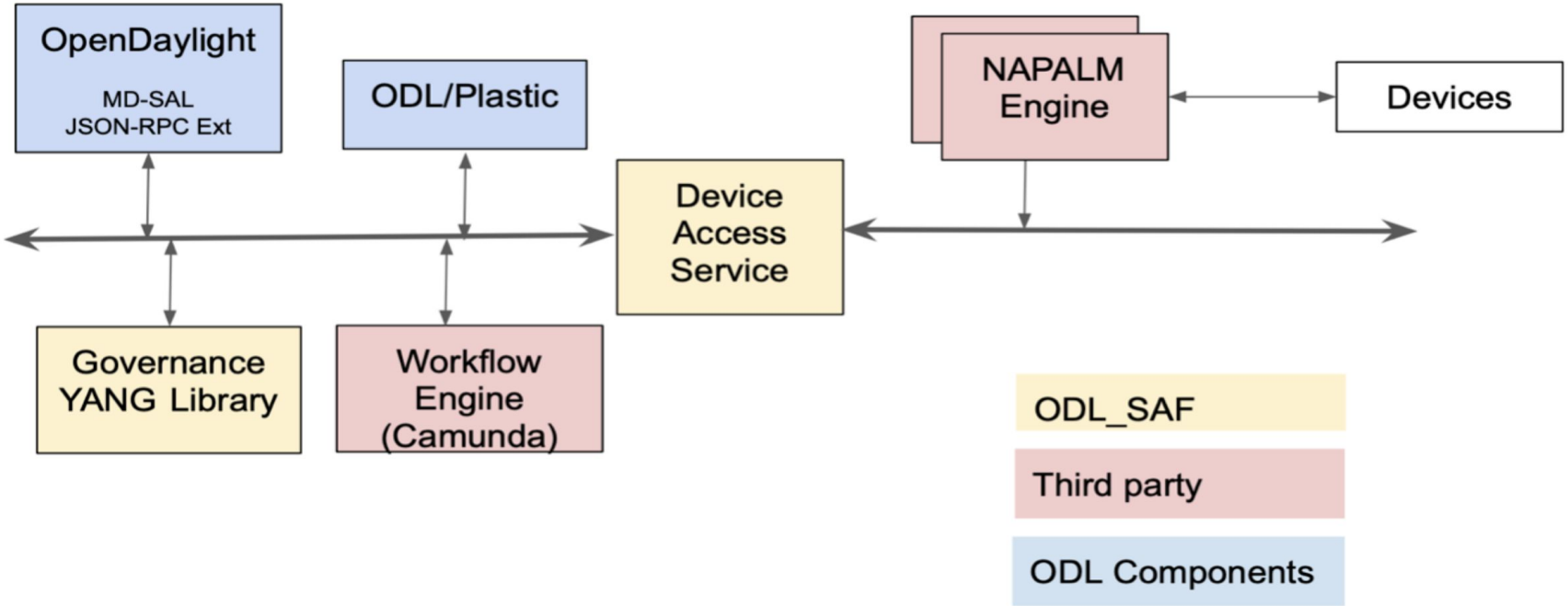
Agenda

- Problem definition
- SAF Architecture and modules
- Workflow Engine
  - APIs
- NAPALM
- Telemetry
- Demo

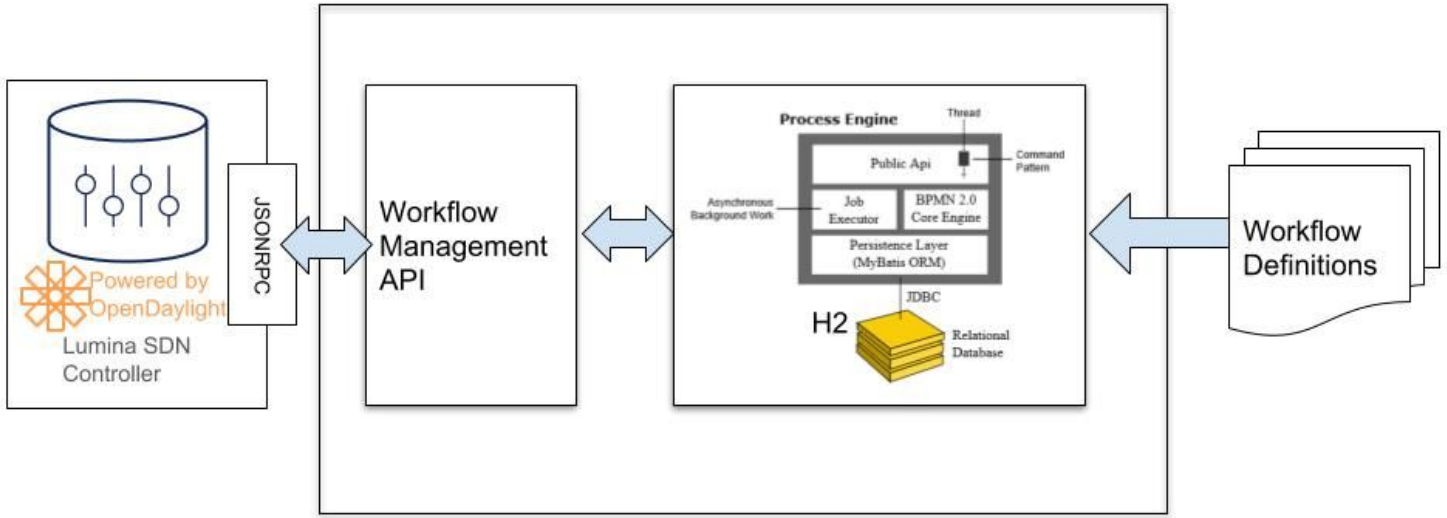
Problem definition

- Simplify Service provisioning
- Transaction Management capability for device/service provisioning
  - Pre-check, post-check, rollback,...
- Heterogenous device management
  - Vendor devices with varying CLIs

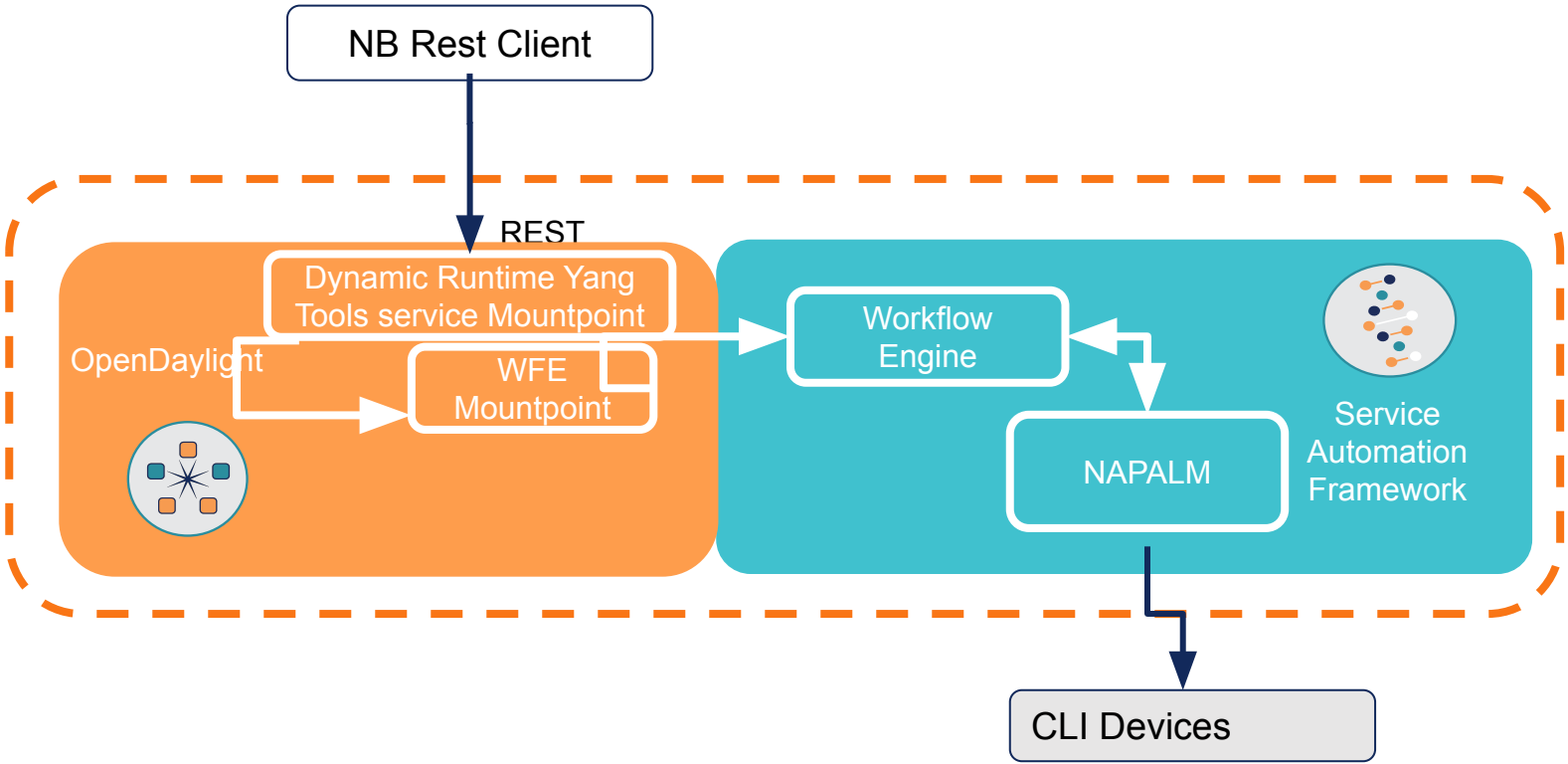
SAF Architecture



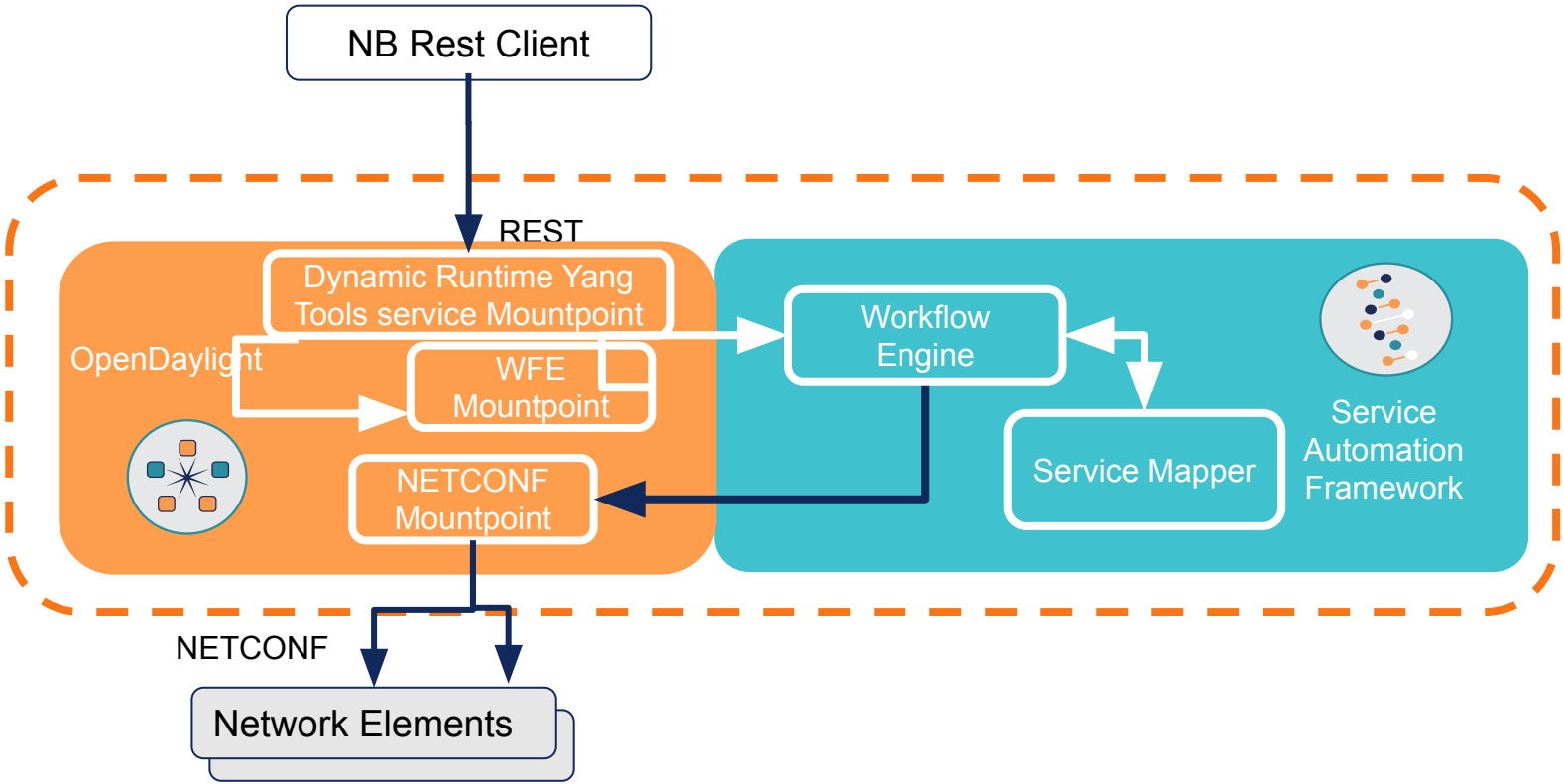
SAF Modules


Powered by OpenDaylight
Lumina SDN Controller
JSONRPC
Workflow Management API
Process Engine
Thread
Public Api
Command Pattern
Asynchronous Background Work
Job Executor
BPMN 2.0 Core Engine
Persistence Layer (MyBatis ORM)
JDBC
H2
Relational Database
Workflow Definitions

CLI Devices - SAF



Netconf - SAF


NB Rest Client
REST
Dynamic Runtime Yang Tools service Mountpoint
OpenDaylight
WFE Mountpoint
NETCONF Mountpoint
Workflow Engine
Service Mapper
Service Automation Framework
NETCONF
Network Elements

Overview of Workflow elements

Business Process Model and Notation (BPMN) is the global standard for workflow modeling.

Supported BPMN Constructs in SAF

- Start/Stop Event
- Tasks - Tasks allow modeling the actual work being performed in the process. Different types of tasks are supported.
- Gateways
  - **Data-based Exclusive Gateway (XOR)** - only one sequence flow is selected
  - **Parallel Gateway** - introduce concurrency in a process model is the Parallel Gateway, which allows forking into multiple paths of execution or joining multiple incoming paths of execution.

Workflow engine terminologies

Terminology	Description
Workflow Definition	A Workflow definition defines the structure of a Workflow
Workflow Instance	A Workflowinstance is an individual execution of a workflow definition.
Job ID	Workflow Instance UUID
Status	Output of workflow execution

NAPALM

- NAPALM (Network Automation and Programmability Abstraction Layer with Multivendor support) is a Python library that implements a set of functions to interact with different network device Operating Systems using a unified API.

Supported Network Operating Systems:

- Arista EOS
- Cisco IOS
- Cisco IOS-XR
- Cisco NX-OS
- Juniper JunOS

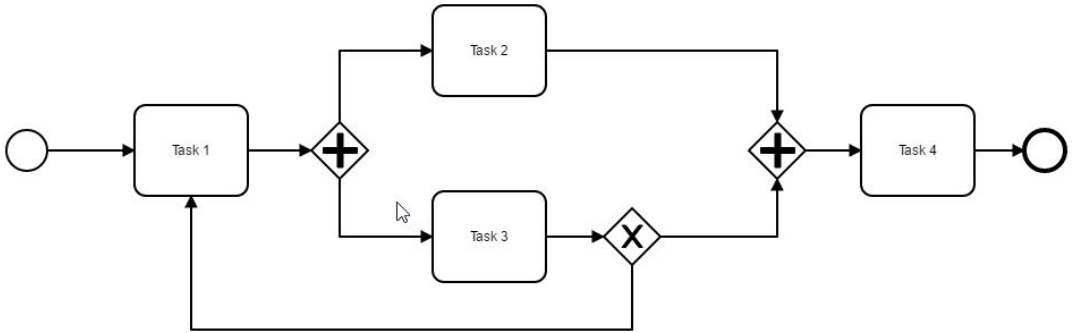
Extras

In addition to the core drivers napalm also supports community driven drivers. You can find more information about them here: Community Drivers

NAPALM as SAF service

- NAPALM has been packaged into a jsonrpc service in SAF
- Exposes only datastore methods. Custom rpc's are not supported
- Conversion of device config to/from standard yang model format can be achieved using either of the 2 different libraries:
  a. **napalm-yang**:
     - Works only with **openconfig models** (additional models can be added, but it is a tedious task. Also, the project has been abandoned by the owner in github)
     - Internally makes use of napalm in its source code, hence difficult to decouple
  b. **ntc-rosetta**:
     - Newer translation library, translations can be implemented for any yang model
     - No coupling with napalm, hence hosted as a separate microservice

SAF BPMN Workflow patterns



Pattern Name	Use-Case
Parallel Split	Service to Device
Synchronization	Aggregation of result
Exclusive Choice	Pre-Check, Post-Check

SAF Workflow API

Name	Description	Input	Output
Execute	Starts the workflow execution defined	Workflow Name + Workflow Specific Input(YangPath,device Id, etc)	Returns the Workflow Instance ID
Status	Status of the Workflow Instance and the Result if present	Job Id - Id of the Workflow Instance	Returns the Workflow Status and the Output JSON
List	Lists all Workflow Instances present in SAF	Optional filter for name where reg expression can be used	Workflow Instances Array
Export	Exports all Workflow definitions in a file		Name and the zip file will be exported in PVC location
Cancel	Active Workflow instance goes into terminated State	Workflow Instance Id	Workflow Status - Internally Terminated State

Execute

Input:

```
{
 "ldk-wfe:input":{
    "ldk-wfe:workflow-name":"GET",
    "ldk-wfe:workflow-input":[
       {
         "yang-path":"config/jsonrpc:config/configured-endpoints/nxos/yang-ext:mount/openconfig-interfaces:interfaces"
       }
    ]
 }
}
```

Output:

```
{
  "output": {
    "job-id": "2bb6cbc9-9197-11e9-b924-0242ac15000c"
  }
}
```

Status

Only for completed Workflows, history of workflow result is stored. When you call status RPC, transaction status will be one of these: "ACTIVE", "SUSPENDED", "COMPLETED", "EXTERNALLY_TERMINATED", "INTERNALLY_TERMINATED"

Input:

```
{
 "ldk-wfe:input": {
   "ldk-wfe:job-id": "af650f23-86bc-11e9-8939-0242ac13000a"
 }
}
```

Output:

```
{
    "output": {
        "workflow-state": "COMPLETED",
        "workflow-output": {
   ...
         }
    }
}
```

Delegates

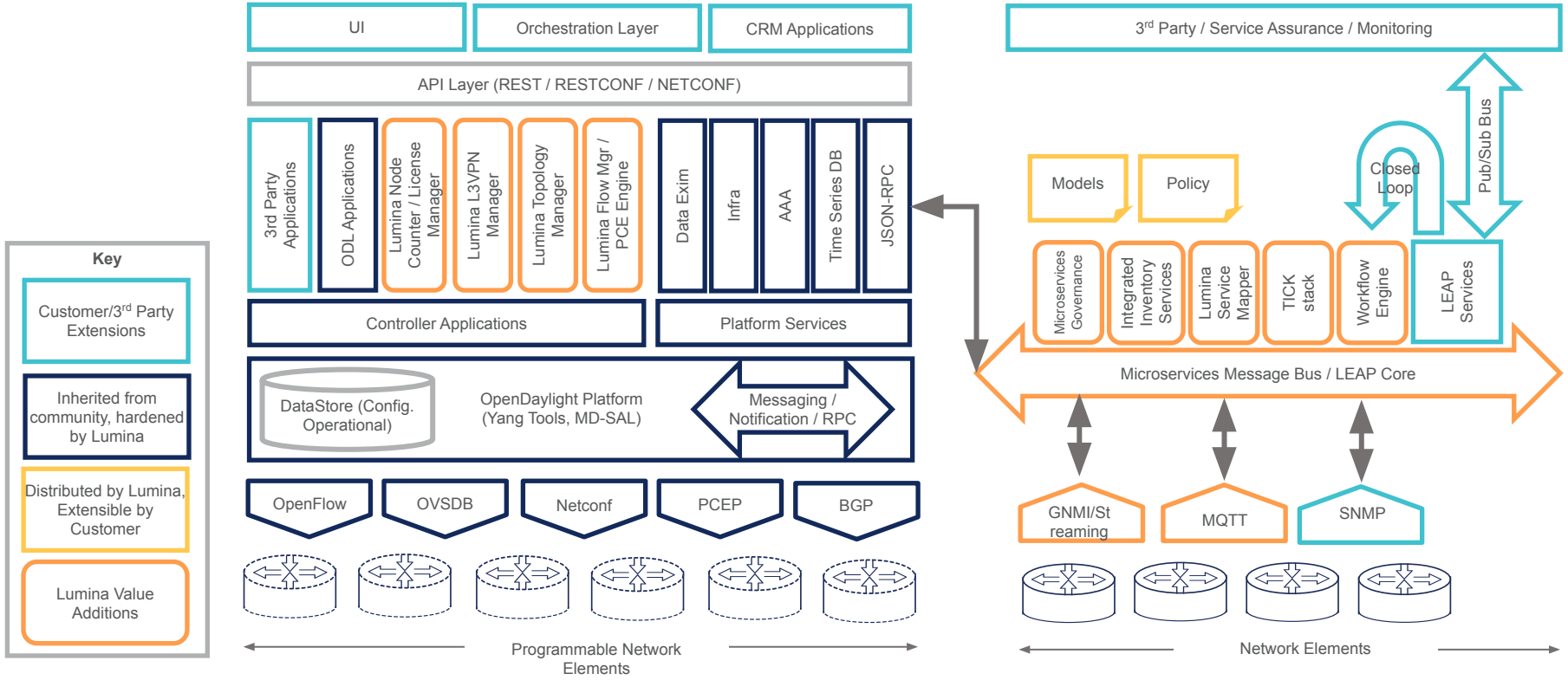
- Allows you to execute external Java code, scripts or evaluate expressions when certain events occur during process execution
- Can be attached to a BPMN task
- SAF's Delegates
  - **PutDelegate** - Modifies the device configuration(Interface/vlan) if exists by making rest API call to ODl
  - **GetDelegate** - Get the device information if exists by making rest API call to ODL
  - **RollbackDelegate** - Rolls back the device configuration using checkpoint file if exists on a device using Ldk Napalm Api RpcService
  - **TransformDelegate** - Transforms an input to an output based on the given schemas and the data given in workflow input. The input and output types do not need to be identical.
  - **BackupDelegate** - Backup data in a checkpoint file from device using Ldk Napalm Api RpcService if the device supports backup
  - **DeleteBackupDelegate** - Removes the backup checkpoint file created for a device if exists using Ldk Napalm Api RpcService
  - **DeleteDelegate** - Remove the device configuration(Interface/vlan) if exists by making rest API call to LSC
  - **PostcheckDelegate** - After device config modification such as PUT/DELETE, this delegates verifies the modification operation is successful
  - **PrecheckDelegate** - Before device config modification such as PUT/DELETE, this delegates verifies list of conditions(Example PathValidation)

Adding new Workflow

Add complete workflow related files at shared location

a. If we need to create new workflows and execute;
   There are shares locations needs to be created for sharing files between host on which SAF is running.
b. User will need to place all the files required for execution of the workflow i.e. bpmn files, task scripts etc.
c. Once all the files are placed WFE will add those workflows and ready for execution.
d. Shared locations are VPC created for K8S deployments which user must ensure are present on the host before installing SAF in system.

Lumina Extension & Adaptation Platform: Telemetry



Demo

The demo showcases configuring a vRouter CLI (JunOS) via Openconfig YANG model