Configuration & Persistence Service

Component Overview, Developments & Demos

Agenda

• Component Overview
• Recent Developments
• Demo’s
  • CPS-Core: Store & Query Yang Modelled Data
  • NCMP: Async Passthrough Requests
  • NCMP: Module Synchronization Watchdog
  • NCMP: Metadata Retrieval and Queries
• Upcoming Developments
Component Overview

CPS Project

TBDMT

NCMP

DMI-Plugin

SDNC

CPS

Postgres

CPS Temporal

Postgres Timescale DB
Responsibilities CPS

• Configuration Persistence Service
  – Cloud native, YANG native, persistence
    • Handles any YANG Modelled data
    • Adds new YANG modules & data at run time
    • Validation, CRUD and Query support
    • Data Separation using Dataspaces (user, application) and Anchors (instance model) separation
  – CPS-’Core’ Beating heart of NCMP
Responsibilities NCMP

- **Network Configuration Model Proxy**
  - Proxy to harmonize access to network CM data
  - Hides the complexity of addressing
  - Provide Yang Data (optional cache of network data – in development)
  - Provides device models (YANG modules)
  - Respect Access Control (TBD)
  - Leverages CPS-Core for its own and CM Data
Responsibilities CPS-Temporal

- Provides an historical view for operational network data
  - Time oriented perspective for all CPS data
  - Features to store and retrieve sequences of configurations and states with associated time when they have been observed
  - For use cases in which analytic apps need to know about variations or trends over time (e.g. BGP flapping, thresholding, …)
Responsibilities DMI Plug-in

• Data Model Inventory Plug-in
  – DMI provides abstract view of:
    • Data – The CM data as published by the network function – always YANG
    • Models – The YANG models that describe the network functions
    • Inventory – The network functions that will be exposed by NCMP
  – Enables integration with:
    • Multi-vendor EMS
    • Network functions (e.g. ORAN O1 functions)
Responsibilities TBDMT

- Template Based Data Model Transformer
  - Generic (pre-defined) queries to CPS or NCMP
  - Templates encapsulate 'mappings' from an application model to one or more network models
  - Helps with Multi-vendor and Multi-version issues
- Hosted by CPS-Project, Development by Wipro
Recent Developments

- Query NCMP CM Handles on Modules (J) and Properties (K)
- Multi Instance Support (J & K)
- Support Passthrough-Running write use cases (J)
- Async Requests Handling using Kafka (K)
- CM Handle States for Module Sync, Data Sync (K)
1. CPS-Core: Store & Query Yang Modelled Data
2. NCMP: Metadata Retrieval and Queries
3. NCMP: Async Passthrough Requests
4. NCMP: Module Synchronization Watchdog
CPS Core Concepts

- **Dataspace**: An application defines a dataspace (name) and is responsible for maintaining the models in it.

- **SchemaSet**: 1 or more Yang Modules in 1 or more files describing the data-tree for an Anchor

- **Anchor**: a reference to the Schema Set and placeholder for top element of the data-tree

More details in: [CPS Concepts PPT](#)
Prerequisites: CPS-Core (incl. Postgres backend)

- Create
  - Dataspace
  - SchemaSet
  - Anchor
- Store Bookstore Data
- Get Bookstore Data (with/without descendants)
- Query: CPS-Path Queries (ancestor option)
- Patch Data
Demo #2 NCMP Metadata

- CM Handle Registration Process
- Get CM Handle Data (incl. module references)
- Query on Module Name
- Query on Properties
- Combinations
For ‘Advised’ CM Handle Models:
- Wake up every 30 seconds (configurable)
- Get any CM Handle where state='ADVISED'
- execute model-sync (on watch dog thread)
  - If OK set state to 'READY'
  - If model sync fails, cm handle state = "LOCKED"
- Go back to step 2 and repeat
NCMP will Asynchronously forward request to DMI and returns a Request ID to client.

DMI handles Asynchronous request and publishes response to Kafka.

Kafka consumed by NCMP and forwarded to be consumed by client consumer.

Request ID will match Correlation ID from Kafka message created by DMI.
1. Client makes async REST call with \texttt{?topic=client-topic}.
2a. NCMP assigns id and sends 200 OK to client (ACK).
2b. NCMP sends request to DMI with id \\texttt{?topic=client-topic}.
3a. DMI retrieves information and sends 200 OK to NCMP (ACK).
3b. DMI produces Kafka message with information.
4a. NCMP consumes response.
4b. NCMP produces Kafka message for client.
5. Client consumes response.

\textbf{Note:} The diagram illustrates two separate Kafka instances whereas in reality it will be the same instance with multiple topics.
Upcoming Developments Short Term

- Data Sync NCMP-Operational Datastore ‘Cache’ (K)
- State Change Notifications (K)
- Yang Resource Retrieval (K)
- CM Data Change Notification (K)
- Update NCMP-Operational Datastore based on Notifications (K)
Upcoming Developments Long Term

- Additional Data Stores
  (MD-SAL inspired ODL sample Presentation ODL)

- Multiple ‘Top Level’ Elements per Anchor/CM Handle

- Fine Grained Cache Control

- Support Dynamic Inventory Changes

- Extend CPS Path Query Capabilities
CPS Quality

- Open SSF Best Practices Gold (K?)
Thank You

for more information see CPS Developers Page
Questions

for more information see [CPS Developers Page](#)