ARCHCOM: InfoFlow - RunTime Config DB Information Flow

1. Scope

DESCRIPTION: RunTime Config DB is a data lake repository for configuration and operational parameters. Run Time Config DB is a common service component that ONAP components can access write and read information to. The term "config" is used in the name for legacy purposes, but the use case is not limited to just configuration parameters and it is intended to be a repository for Operational parameters, and eventually policy information.

WHEN EXECUTED: This information flow is used during Run Time, when configuration or operational data is written to the Runtime Config DB.

- **Flow 1** (Write) - VES Configuration information coming from xNF via e.g. CM Notify
- **Flow 2** (Write) - New xNF is added or deleted from ONAP; A&AI notification xNF update.
- **Flow 3** (Write) - Micro-Service or ONAP component writing & updating operational information
- **Flow 4** (Read) - Data is read from RunTime Config DB

PURPOSE: RunTime Config DB serves as a data lake as a common service and data layer for ONAP components and micro-services.

INFORMATION PASSED: Configuration information (from CM Notify) or operational information (derived during ONAP operations).

ACTORS:

- RunTime Config DB
- Operations Specialist (ONAP user)
- Controller/ONAP Component, A&AI, VES collector/DMaaP

For more information and details you can visit the RunTime DB Use Case Wiki at: CONFIGURATION PERSISTENCE SERVICE R6

OVERVIEW RUNTIME CONFIG DB

The: ARC RunTime DB Component Description - R6 Frankfurt

wiki describes a more detailed figure and description of the component.

PURPOSE OF RUNTIME CONFIG DB:

- **REPOSITORY** - The types of data that is stored in the Run-Time data storage repository for:
  - (1) CONFIGURATION PARAMETERS used by xNFs in run time. For example 5G Network run-time instance configuration information. and
  - (2) OPERATIONAL PARAMETERS used by ONAP and xNFs. Exo-inventory information is information that doesn't belong in A&AI.
  - (3) POLICY INFORMATION - **FUTURE** - Policy, CLAMP Control Loops, Operational Views

- **DATA LAKE** - It is designed to be a common services data layer which can serve as a **data lake**.
- **SYNCING** - The RunTime Config DB enables the ability to sync data between ONAP & the xNFs. (The source of truth can be define).
- **CM FUNCTIONS** - Enables OSS configuration, optimization, and LCM operations. (FUTURE)
- **CM FUNCTIONS** - Enables future CM & Data management functions such as xNF Crash restoration, data restoration, data history management and auditing. (FUTURE)
- **CENTRAL/DISTRIBUTED** - Because it is a common service, it is part of an ONAP installation, so it could be deployed with either an Edge ONAP installation or a centralized ONAP installation. (FUTURE)
- **SCOPE** - The Run Time DB could also serve as the data storage to store for example ONAP Policy Rules, CLAMP Control Loop, Operational Views (FUTURE) and also accommodate other resources.

ACCESS TO RUNTIME DB (READ/WRITE):

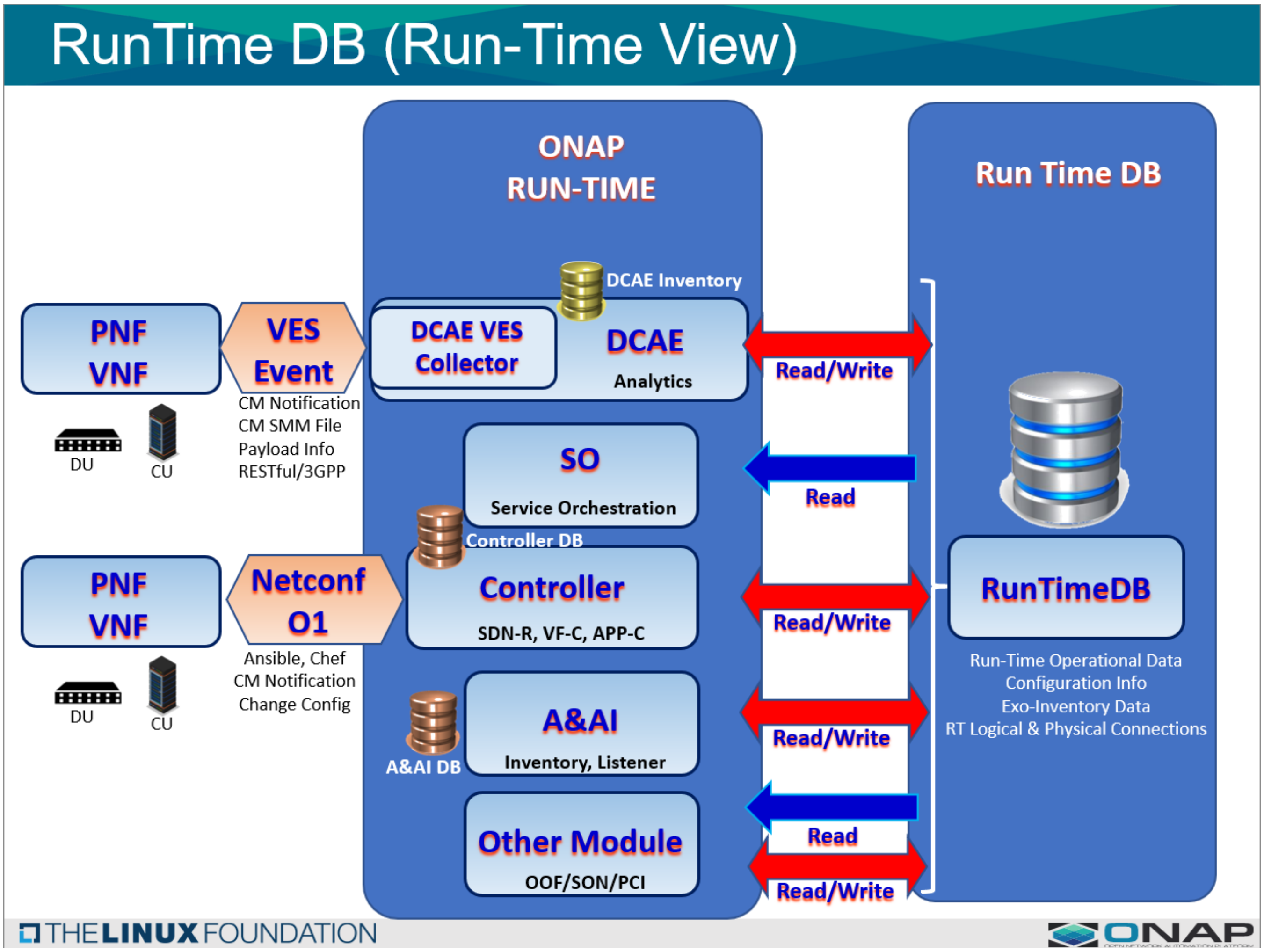
- **READ ONLY** - Run-Time parameters can be READ by any ONAP platform component and any ONAP plug-in. Examples of ONAP platform components are A&AI, SDC, SDNC etc.
- **READ/WRITE** - Parameters can be READ/WRITE from Controllers, DCAE (future), VES Collector/DMaaP, A&AI, Policy/CLAMP (future) and other components with permission settings.
- **DEFAULT** - SO (future), DCAE, A&AI (indirectly), Controllers (CDS, APPC, SDNC) will have default read/write access to RunTime DB
- **DEFINABLE** - Other components will have default read-only access to RunTime DB but can be given Read/Write access on a per record basis.

SYNCING NEW xNF ADDED or DELETED (A&AI):

- **ELEMENT SYNC** - Software keeps the A&AI elements with the elements in the RunTime Config DB in Sync. When the network first being established, a *GetAllPNFs* function from A&AI can be used on startup.
- **A&AI** - A&AI is still the master of valid entities in the network and provides a dynamic view of the assets (xNFs) available to ONAP
- **RUN TIME DB** - The RunTime DB is a master of the associate (exo-inventory) data associated with the entities.
- **DYNAMIC VIEW** - When a xNF appears or is removed from the system, RunTime DB records will be added/removed based on A&AI entries.
- **LOGIC** - When a xNF appears is removed there is logic to determine how and when something is to be updated. There is some intelligence to know what elements of update.

INDEXING:

- **INDEXING** - Data Records will be indexed by xNF (VNF, PNF, ANF). It would be an objective to have a similar indexing mechanism as A&AI. May also need an index to be a logical object ID.
- **RETRIEVAL** - How are data records retrieved efficiently. This relates how the records are indexed.



RunTime DB (Run-Time View)

The above diagram shows the usage of RunTime DB

It shows the four basic flows captured in the diagram.

- Writing information from a VES CM Notify event
- A&AI xNF addition/deletion
- Operational information written
- Information being read from the RunTimeDB

OVERVIEW RUNTIME CONFIG DB INFORMATION FLOW

Information Flows to Run Time DB or from RunTime DB during Run Time Operation of ONAP.

1 INFORMATION FLOW DATA WRITTEN TO RUNTIME Config DB:

Information Flows show data being written to the Runtime Config DB

New Information is written to RunTime Config DB from a Component or a Micro-service

The following three basic flows are described:

- **FLOW 1**: VES Event (CM Notify) Updates - Information flow
- **FLOW 2**: A&AI xNF (create/delete) updates - Information flow
- **FLOW 3**: micro-Service / Controller (SDN-R, SDN-C, APP-C, CC SDK) / other Component Updates for operational information - Information flow

2 INFORMATION FLOW DATA READ FROM RUNTIME Config DB:

Information Flow from RunTime DB

Other components are reading from RunTime DB.

Taking information from RunTime DB and using it to send to xNF components

- **FLOW 4**: Data is read from RunTime DB

Race Conditions - a hysteresis (a time difference) between writing information (from a Kafka broker) and a read request arriving before the writing has finished.

2. PRE CONDITIONS

ONAP is ready & running:

- **ONBOARDED ARTIFACTS** - (future) If dynamic setup is used, definition artifacts are onboarded and used to setup the RunTime DB structures
- **ONAP SOFTWARE** - There is an ONAP installation. Software images loaded in OpenStack installation, where instantiation will happen (since no S/W image repository). Need to be available in Target Cloud Instances.

RunTime Config DB is setup (Design Time):

- **RUN TIME DB SETUP** - RunTime DB has been setup properly and is ready to be used.
- **DESIGN TIME ACTIVITIES** - Design time activities have happened (SDC service creation)

2.1 RUNTIME CONFIG DB DATABASE & STRUCTURE

2.1.1 DATA STRUCTURE (ONBOARDING & DESIGN TIME)

A data structure which is common for all different vendor xNFs will be used for the RunTime Config DB.

Domain oriented components can be used where all of those components share common information.

Any micro-service or ONAP platform component can define information in the data structure.

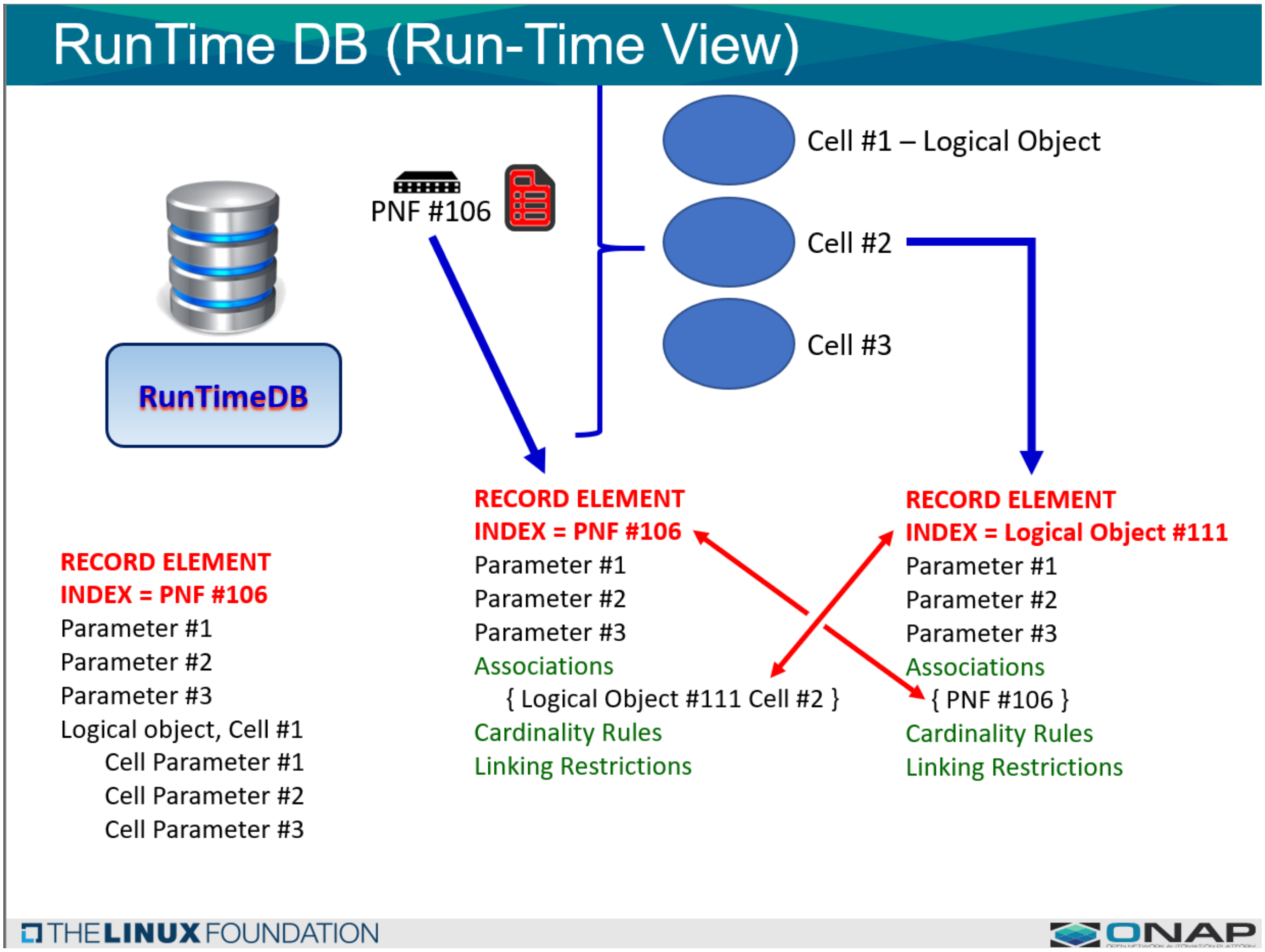
Before Run Time, the RunTime Config DB is setup with the appropriate data structures that it needs to work from SDC CSAR Service definition file package.

The Run Time Config is schema for the records RunTime Config DB are defined in advance. In the future (later releases) the RunTime Config DB schema may defined data structures on the fly.

Topology-type can be represented through the xNF associations in the schema

DESIGN TIME - The schema of data structure of RECORDS the run Time Config DB can support are created and defined in advance.

RUN TIME - During Run Time the data and associations are DYNAMICALLY Run Time config DB updated using the schema of the records.



RunTime DB (Run-Time View)

RECORD ELEMENT
INDEX = PNF #106
Parameter #1
Parameter #2
Parameter #3
Logical object, Cell #1
Cell Parameter #1
Cell Parameter #2
Cell Parameter #3

RECORD ELEMENT
INDEX = PNF #106
Parameter #1
Parameter #2
Parameter #3
Associations
{ Logical Object #111 Cell #2 }
Cardinality Rules
Linking Restrictions

RECORD ELEMENT
INDEX = Logical Object #111
Parameter #1
Parameter #2
Parameter #3
Associations
{ PNF #106 }
Cardinality Rules
Linking Restrictions

THE LINUX FOUNDATION

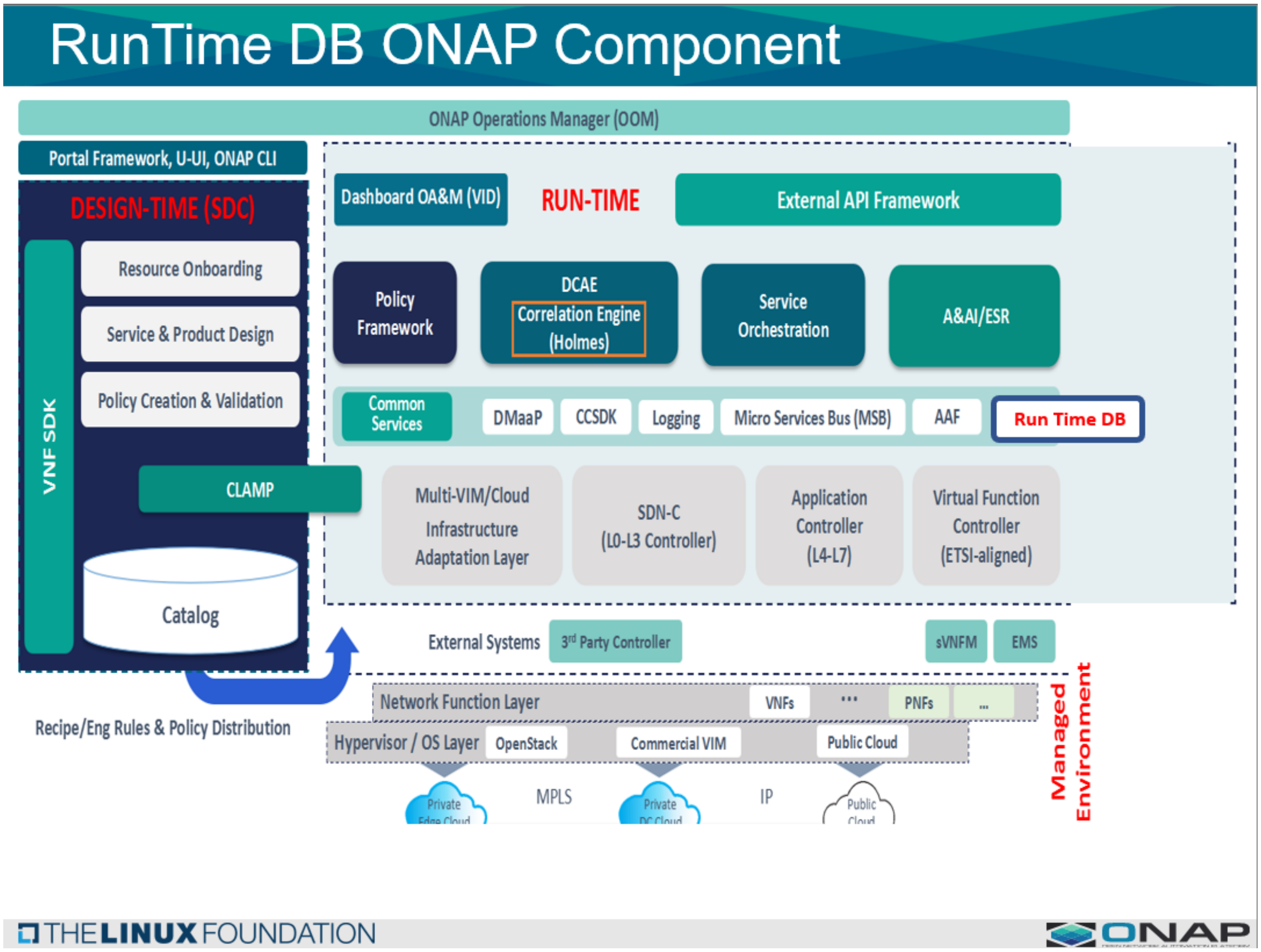
ONAP

2.1.2 RUNTIME DB DATA LAYER

The RunTime DB is a Data layer common service data lake.

There has been quite a bit of discussion related to how to architect the RunTime DB component.

In R6 is was determined, that it should be a common service as a data layer to other ONAP components.



RunTime DB ONAP Component

ONAP Operations Manager (OOM)

Portal Framework, U-UI, ONAP CLI

DESIGN-TIME (SDC)

VNF SDK

Resource Onboarding

Service & Product Design

Policy Creation & Validation

CLAMP

Catalog

Recipe/Eng Rules & Policy Distribution

Dashboard OA&M (VID)

RUN-TIME

External API Framework

Policy Framework

DCAE

Correlation Engine (Holmes)

Service Orchestration

A&AI/ESR

Common Services

DMaaP

CCSDK

Logging

Micro Services Bus (MSB)

AAF

Run Time DB

Multi-VIM/Cloud Infrastructure Adaptation Layer

SDN-C (L0-L3 Controller)

Application Controller (L4-L7)

Virtual Function Controller (ETSI-aligned)

External Systems

3rd Party Controller

sVNFM

EMS

Network Function Layer

VNFs

•••

PNFs

...

Hypervisor / OS Layer

OpenStack

Commercial VIM

Public Cloud

Private Edge Cloud

MPLS

Private DC Cloud

IP

Public Cloud

Managed Environment

THE LINUX FOUNDATION

ONAP

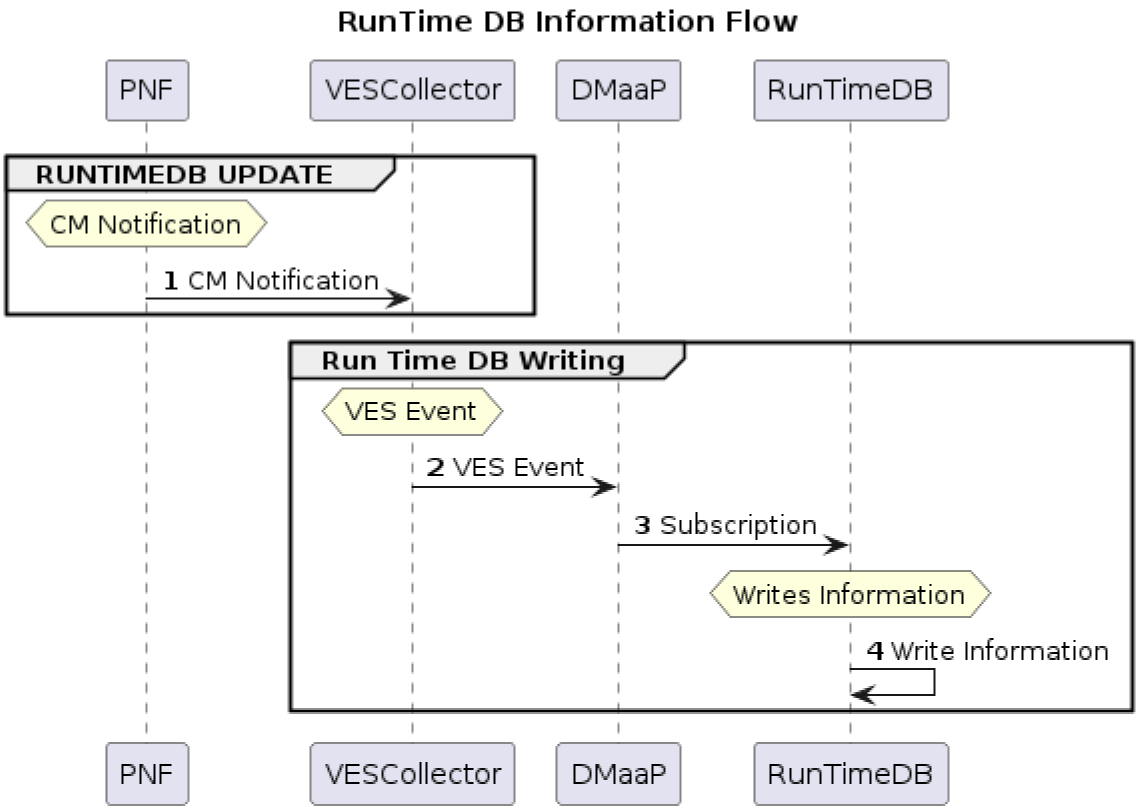
3. Information Flow

These four flows show the usage of RunTime DB

- **FLOW 1**: VES Event Updates (CM Notify) - Information flow
- **FLOW 2**: A&AI xNF (create/delete) updates - Information flow
- **FLOW 3**: micro-Service / Controller / Component Updates for operational information - Information flow
- **FLOW 4**: Data is read from RunTime DB

3.1 FLOW 1: VES Information Flow CM Notify - Writing to RunTime DB

The following UML diagram shows the Information Flow for RunTimeDB



3.3 Flow Description: mS Information Flow

1. **VES Event** – VES Event Received

A VES event (CM Notify) is received by the DCAE VES collector sent from the PNF.

A xNF determines that a configuration parameter needs to be updated, thus it triggers a CM Notify towards ONAP with the objective of updating the RunTime Config DB.

2. **Publish on DMaaP** – VES Collector publishes the event onto DMaAP

VES Collector Publishes on DMaaP with the CMNotify Topic

3. **Subscription on DMaaP** – RunTime Config DB gets the Notification

RunTime Config DB subscribes to the Event.

In the initial release (R6), the RunTime Config DB will use SDN-C's DMaaP listener capability; but the goal is that RunTime Config DB is a separate independent component so it would have its own DMaaP Listener.

4. **Updates DB** – The RunTime Config DB is updated with the information

The RunTime Config DB is updated with the information

3.1 FLOW 2: xNF Addition/Delete A&AI Update Flow - Updates to RunTime DB

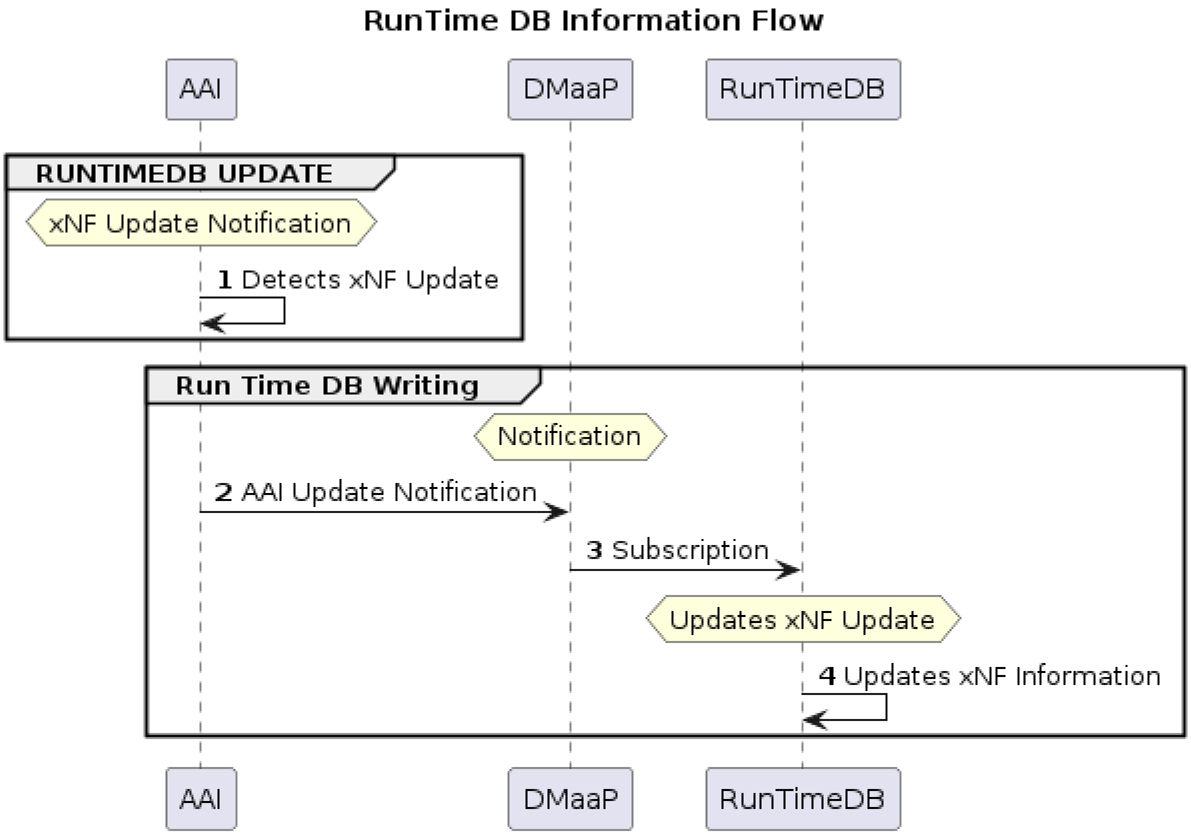
The following UML diagram shows the xNF Update flow from updates in A&AI for RunTimeDB

In this flow, A&AI has determined that a xNF (PNF or VNF) has been removed or added to the network.

And so downstream dependent components need to update their information that a xNF has been removed/added.

In the case of RunTime, there would be a record for that xNF and it would be need to be removed/added the basic mechanism of how this is done is reused (nothing new is introduced): A&AI publishes an notify event on DMaaP bus,

and RunTime (component) subscribes to that event and updates itself.



The describes the xNF information updates and keys based on xNF update.

3.3 Flow Description: AAI Update Information Flow

1. **AAI Determines update** – A xNF has been removed or added

AAI determines a xNF has been removed/added from the network. Thus, downstream ONAP components need to be aware of this change in the network.

2. **AAI publishes on DMaaP** – AAI publishes on DMaAP

AAI publishes on DMaaP A&AI Notify, that a xNF status has changed.

3. **Subscription on DMaaP** – DMaaP gets the Notification

RunTime Config DB subscribes to that event; and so, gets the AAI updates.

4. **Updating the RunTime Config DB** – The RunTime Config DB is update.

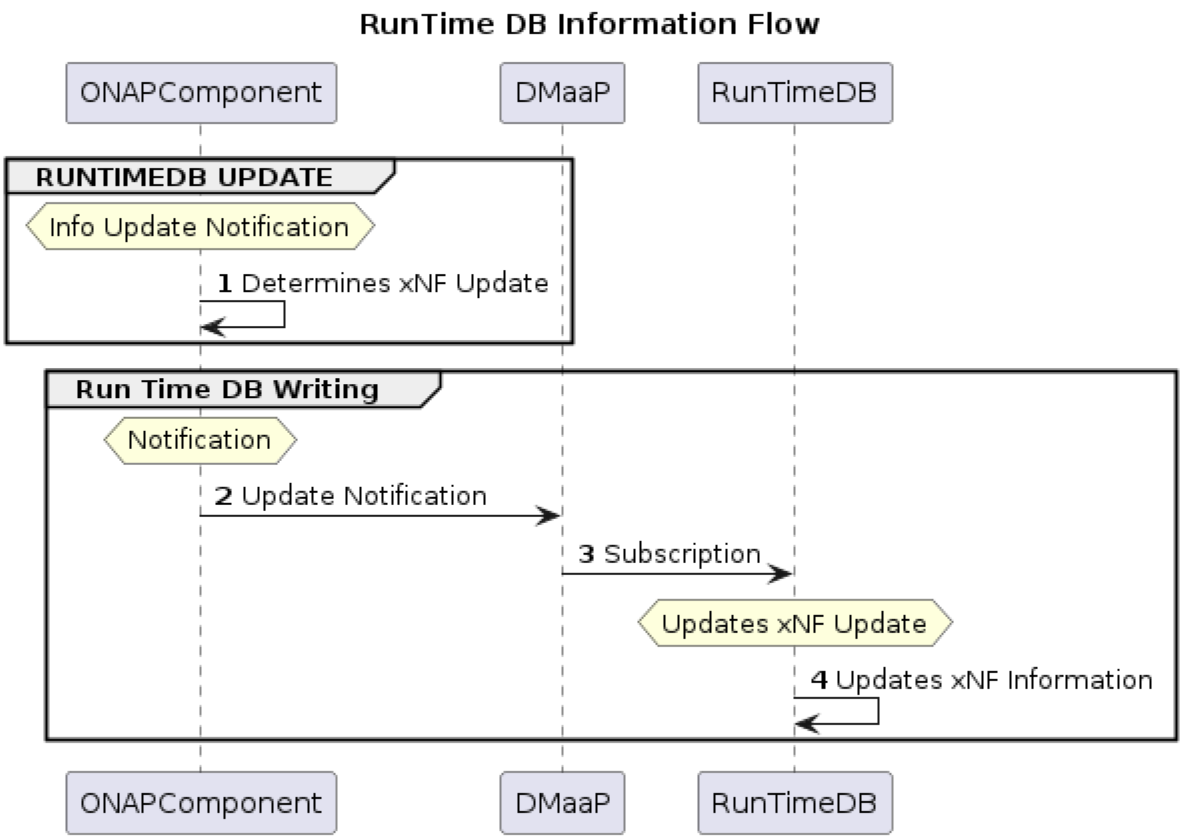
REMOVAL - RunTime DB removes the record for that xNF

ADDITION - If AAI determines that a new xNF is added to the network, then RunTime DB needs to setup a new record for that xNF with the default data structure and configuration. Default Configuration is setup. For a new xNF a pre-defined schema record is used for that xNF.

- (a) Some information may come from A&AI.
- (b) Other values will be discovered during operation.
- (c) Some information data may have default values defined in the schema.

3.1 FLOW 3: mS/Controller Operational Info Update Flow - Writing to RunTime DB

The following UML diagram shows Where another ONAP component or Micro-Service updates the RunTimeDB



The describes the xNF information updates and keys based on xNF update.

3.3 Flow 3 Description: ONAP Component Writing to RunTime Config DB Information Flow

1. **Micro Service Determines Update Needed** – The mS determines an update is necessary.

The micro service, (policy), controller, ONAP component during operation determines an update is needed and action needs to be taken.

When this happens, it will publish onto the DMaaP Bus. Composes policy guidance. When that request comes in the controller, makes a config change on the underlying xNF device, with a successful outcome, it then determines that an update to the Runtime Config DB needs to be updated.

(Controller/Micro-Service/ONAP component) determines that an update is needed > Controller TO xNF > NetConf Yang > Update RunTime ConfigDB

2. **mS publishes on DMaaP** – Publish onto DMaaP

The mS publishes an event on to the DMaaP Bus.

3. **Subscription to DMaaP** – Subscription from DMaaP

The RunTime Config DB Receives the event from the DMaaP Bus.

4. **Updates DB** – The Database is updated

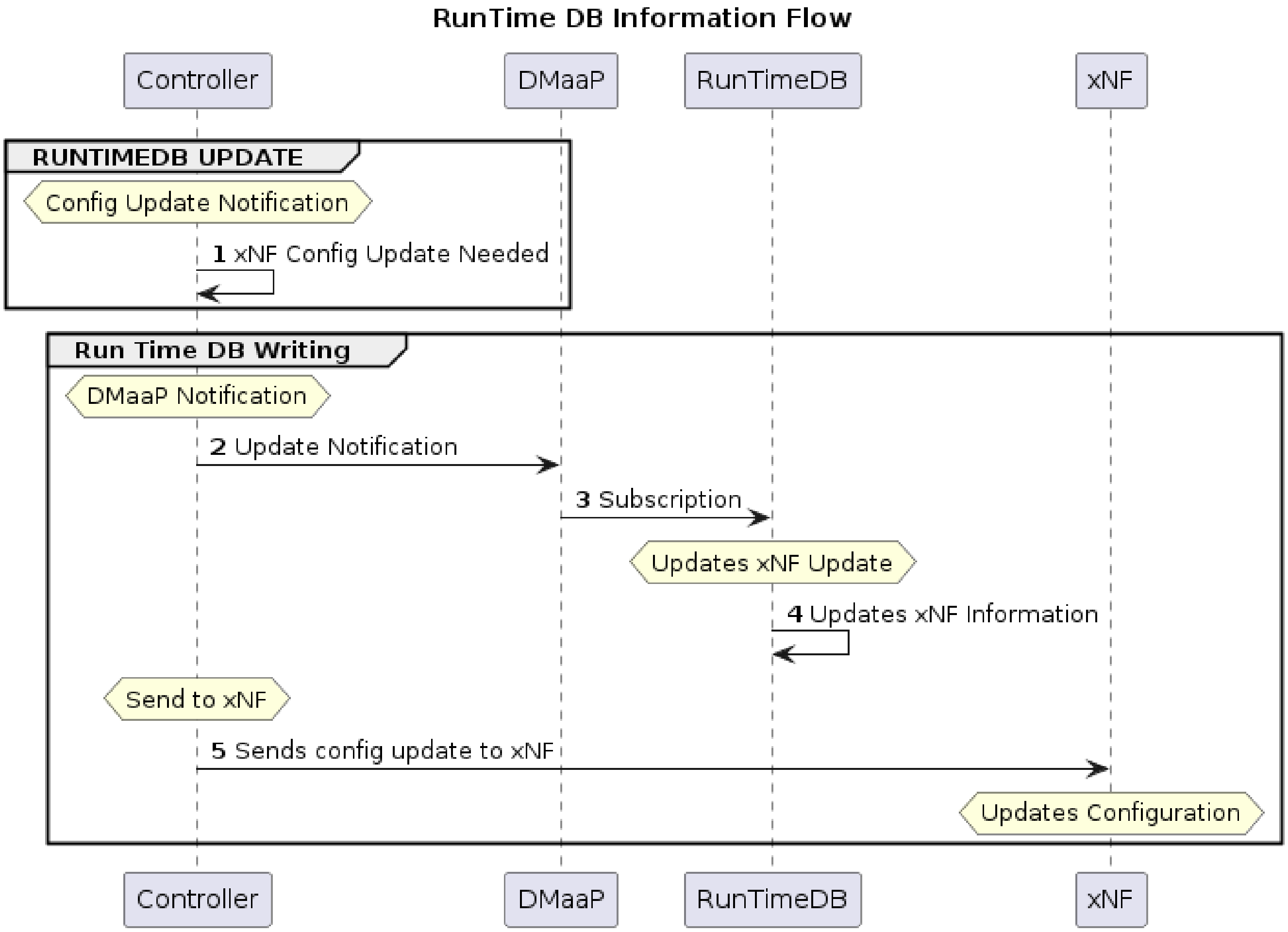
The RunTime Config DB is database is updated with the information that is needed.

3.1 FLOW 4: Controller Sending Configuration Info Update Flow - Writing to RunTime DB

The following UML diagram shows Where a Controller (SDN-C, SDN-R) is updates the RunTimeDB because of a configuration parameter update.

During this flow a configuration update is also sent to the PNF or VNF.

This flow is used when a Controller determines that a Config change is needed. The Controller sends a message to the xNF (via Ansible, or NetConf). The Controller would then send an update on DMaaP which will then update the RunTime Config DB.



3.3 Flow 4 Description: Controller writes to RunTime Config DB Information Flow

1. **Controller Determines Configuration Update Needed** – The Controller determines an update is necessary.

The controller determines a configuration (parameter) update is needed and action needs to be taken.

2. **Controller publishes on DMaaP** – Publish onto DMaaP

The Controller publishes an event on to the DMaaP Bus.

3. **Subscription to DMaaP** – Subscription from DMaaP

The RunTime Config DB Receives the event from the DMaaP Bus.

4. **Updates DB** – The Database is updated

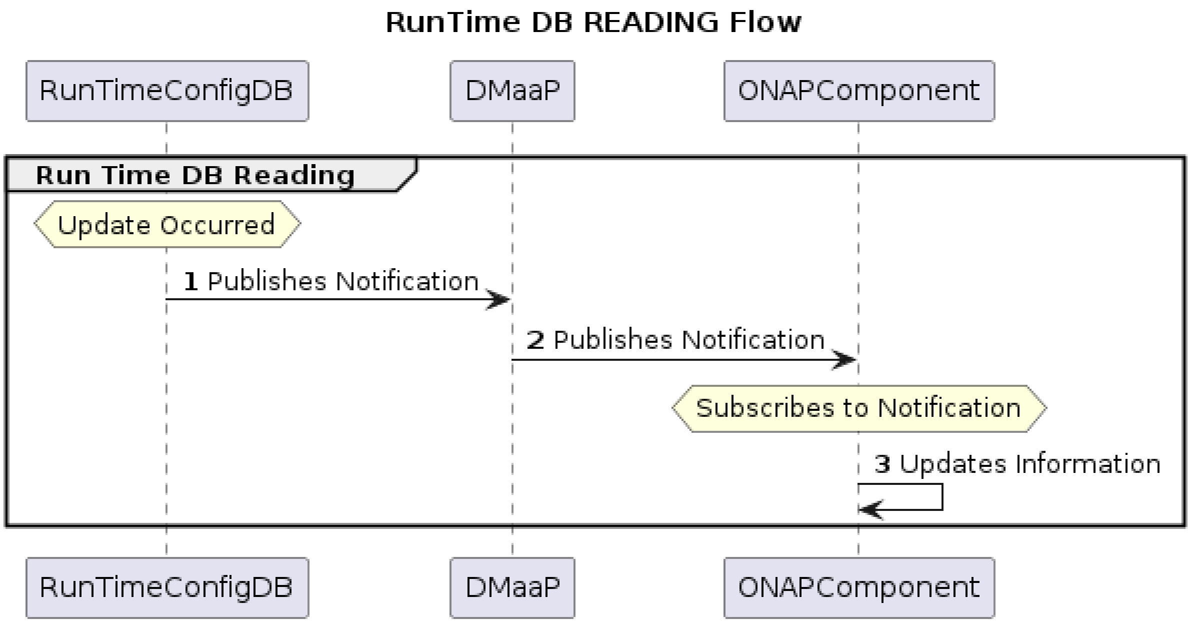
The RunTime Config DB is database is updated with the updated configuration parameter that is needed.

5. **Controller sends to xNF** – The Controller sends to xNF

The controller of the xNF updates the xNF with the configuration information. The xNF updates its internal storage.

3.1 FLOW 5: Reading from RunTime Config DB Info Flow

The following UML diagram shows reading information from the RunTime Config DB.



3.3 Flow 5 Description: Controller reads from RunTime Config DB Information Flow

1. **RunTime Config DB has Update Needed** – The Controller determines an update is necessary.

The RunTime Config DB determines a configuration (parameter) update is needed and action needs to be taken.

2. **RunTime Config DB publishes on DMaaP** – Publish onto DMaaP

The RunTime Config DB publishes an event on to the DMaaP Bus.

3. **Subscription to DMaaP** – The interested ONAP component Subscribes to updates from RunTime Config DB from DMaaP

The ONAP Component Receives the update event RunTime Config DB from the DMaaP Bus.

4. **ONAP Component updates info** – The ONAP Component is updated

The ONAP Component is database is updated with the updated configuration parameter that is needed.

4. Post Conditions

4a. Post Condition (Updated DB)

The post-conditions for the DB:

- **Database is Updated** - The RunTime Config DB has been successfully updated.

- **Acknowledgement** - A "ACK" (Acknowledgement) success or failure is returned to the requester. When a request from policy is updated, the DMaAP is posted that request is successfully completed which includes confirmation of NetConf parameters and runTime database updates. If this eventually becomes a database API, it will just be a simple success or failure message.

REFERENCES

Wiki Page for RunTime Db CONFIGURATION PERSISTENCE SERVICE R6

Architecture component Description ARC RunTime DB Component Description - R6 Frankfurt

SUPPORTING FILES & SLIDES

Files	File