



Run Time Config Database (DB) Component For Data Persistency of Run-Time Information For R7 GuiLin Release

- Requirements call presentation – Mar 16, 2020 version 3

Ben Cheung (Nokia)
Marge Hillis (Nokia)
Joanne Liu-Rudel (AT&T)
Shankar N K (AT&T)

R7 Data Persistency Service / RTCDB

Executive Summary - The RunTime Configuration Database / Data Persistency Service is a new platform component that is designed to serve as a data repository for Run-time data that needs to be persistent. As a stand-alone ONAP component, this project provides data layer services to other ONAP platform components and use cases that require persistent configuration or operational data. The R6 development will be enhanced as well.

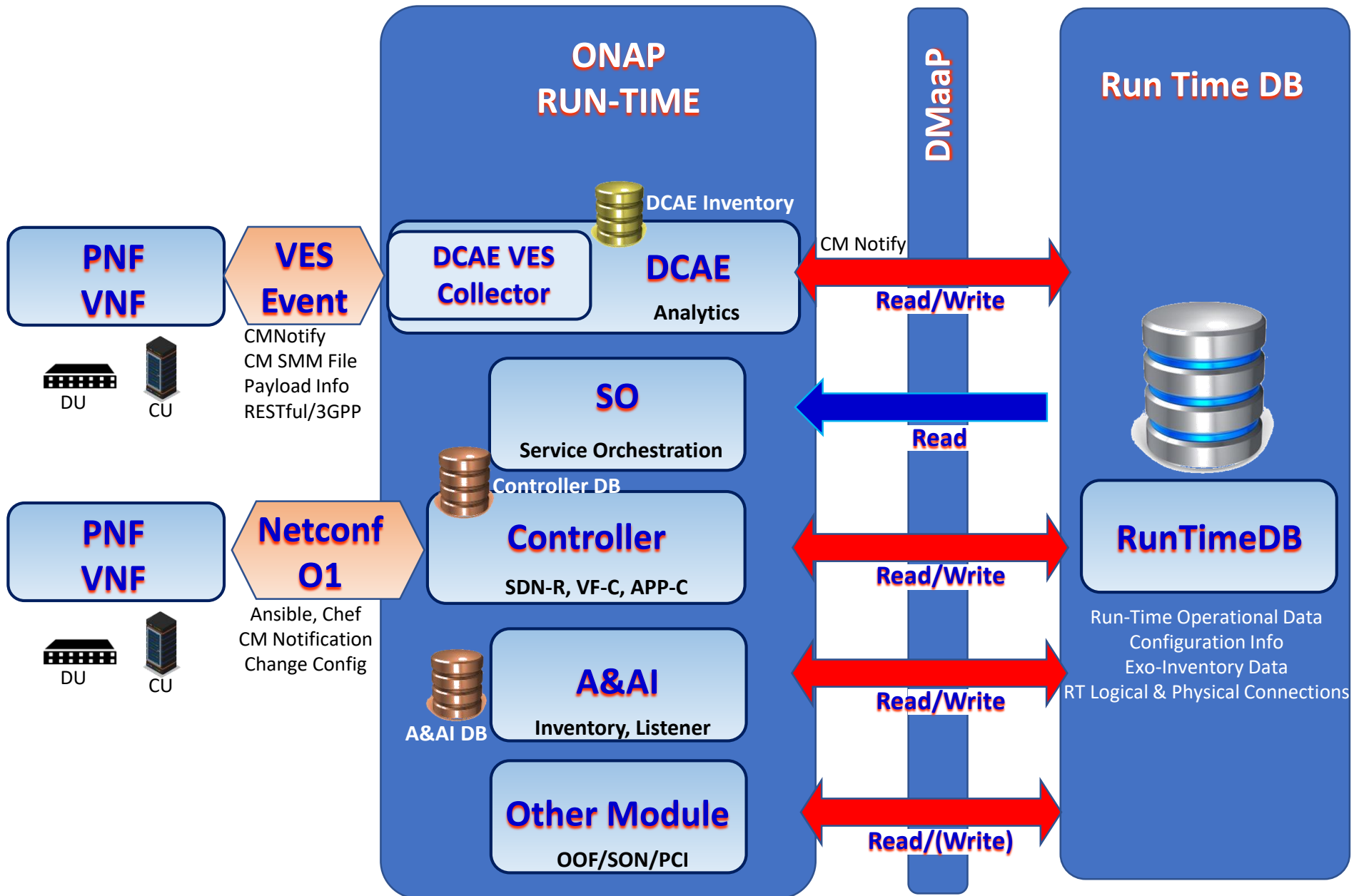
Business Impact - The ability for service operators to visualize and manage data in a RAN network (PNFs, VNFs, and logical constructs) with ONAP is a critical business function because they are key Life Cycle Management (LCM) and OA&M operations. The project has business impacts to enhance the operation of data-handling within ONAP by providing efficient data layer services.

Business Markets - This project applies to any domain (wireless, transport, optical, and wireline) that ONAP may manage. It is not a market or geographical specific capability. It is expected that scaled ONAP installations such as Edge & Core ONAP deployments will also deploy the database across each installation.

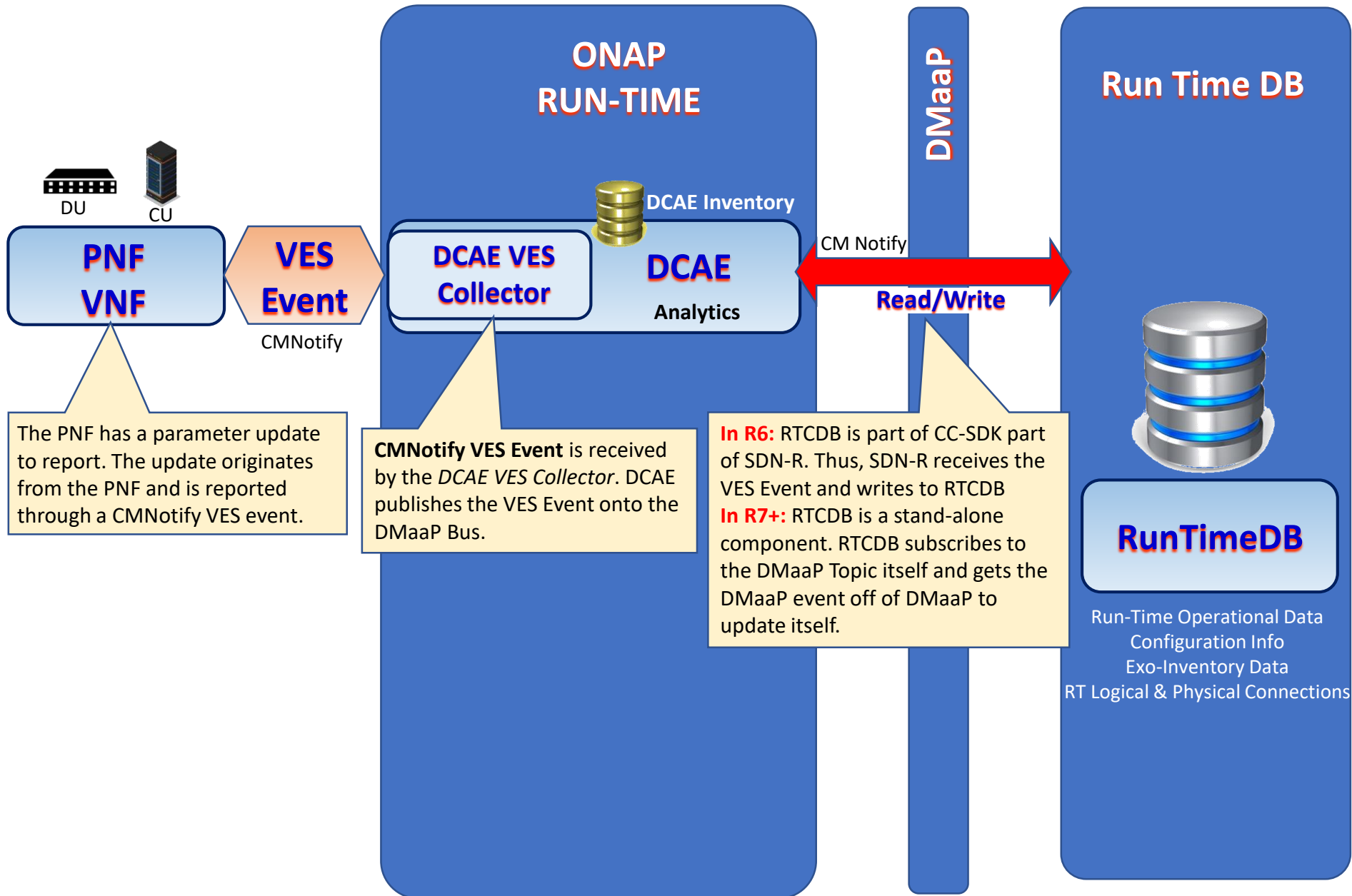
Funding/Financial Impacts - This project represents a large potential Operating Expense (OPEX) savings for operators because of the ability to configure networks saving time and expenses.

Organization Mgmt, Sales Strategies - There is no additional organizational management or sales strategies for this use case outside of a service providers "normal" ONAP deployment and its attendant organizational resources from a service provider.

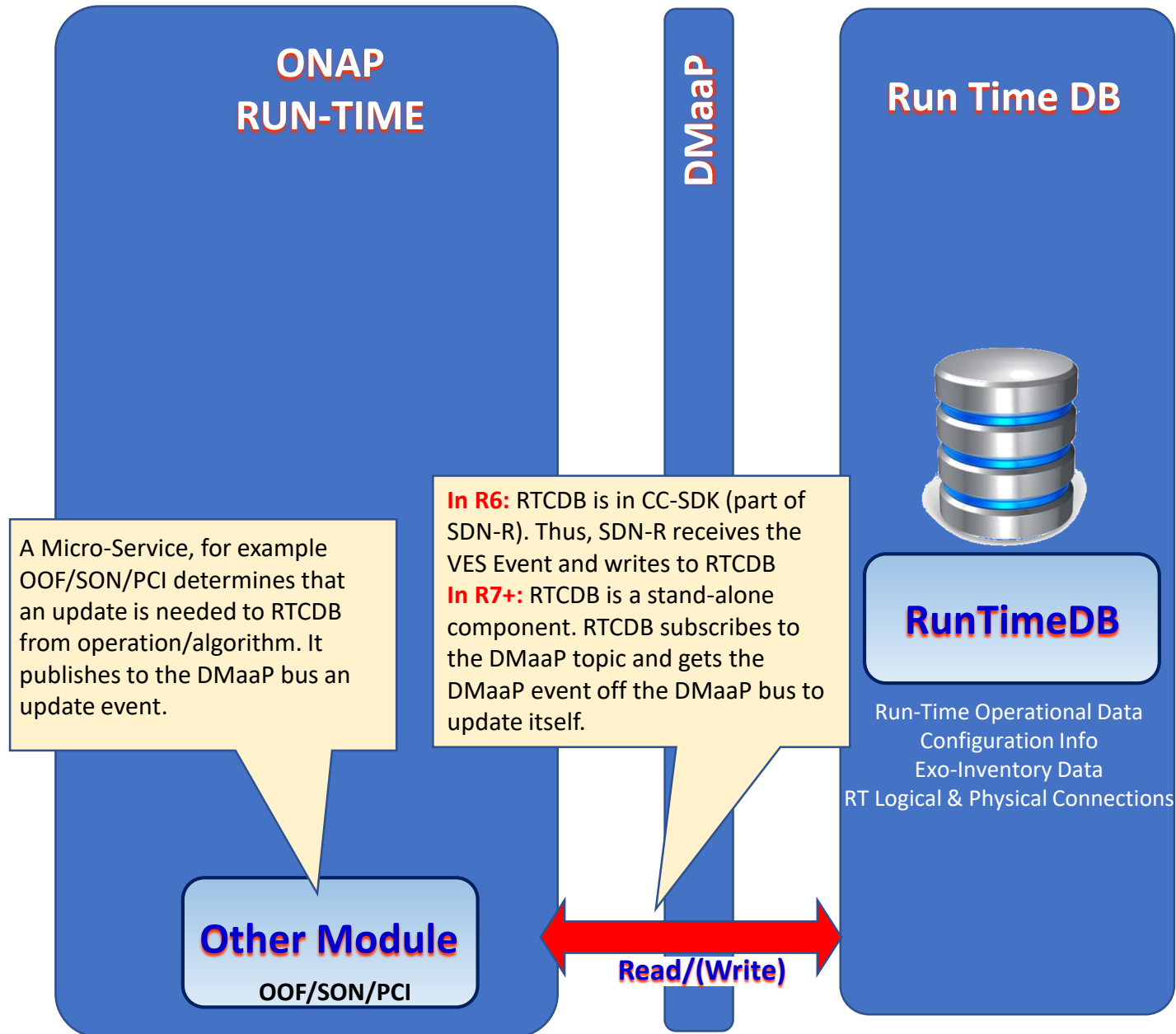
Data Persistency Service (Run-Time View)



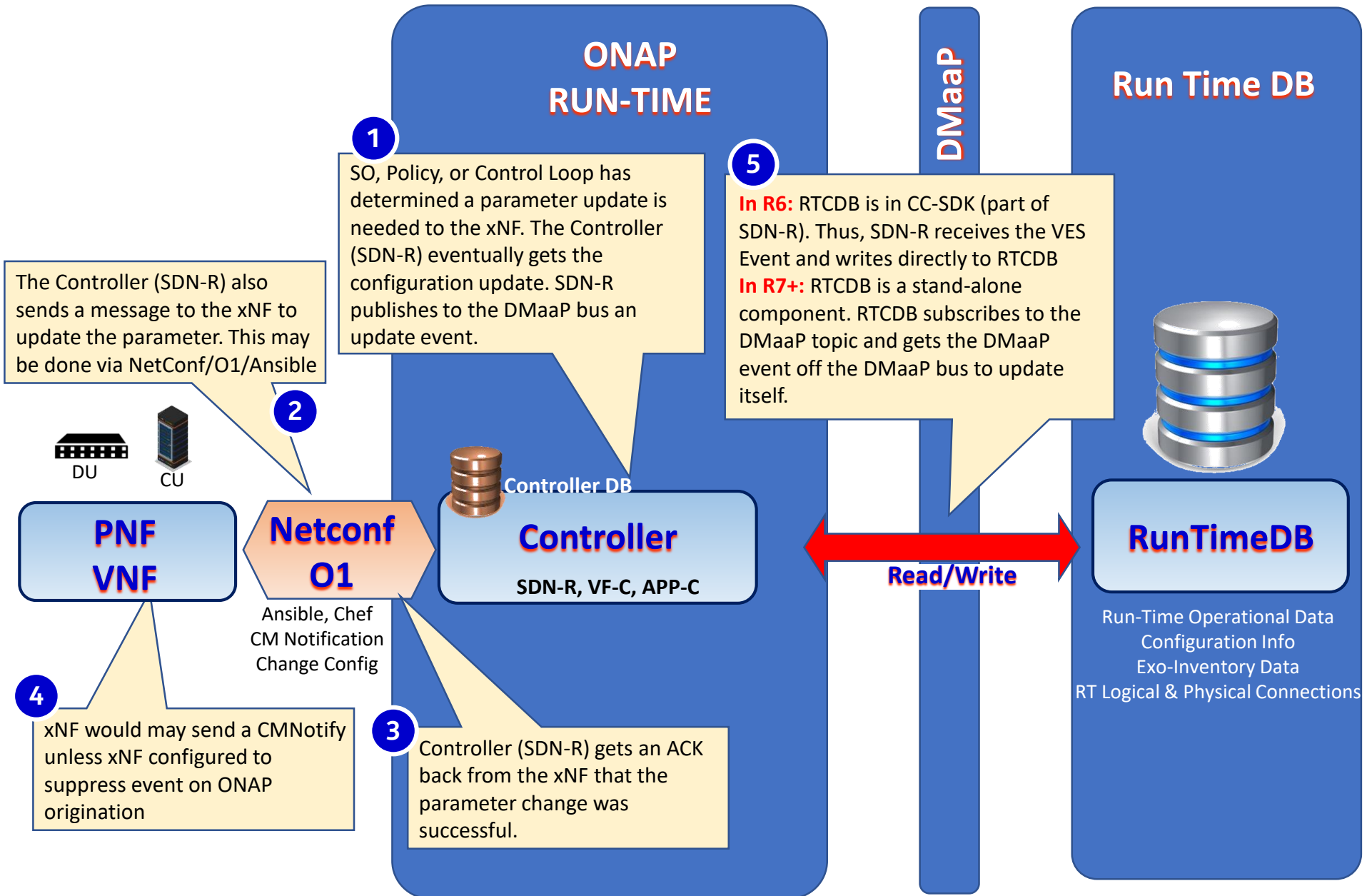
RTCDB READING: PNF Reports CMNotify



RTCDB WRITING: Micro Service Update



RTCDB WRITING: From Controller SDN-R



R7 Data Persistence Service Business Driver

PURPOSE: RunTime Config DB new (ONAP platform component) would be dedicated to storing run-time configuration and operational information that doesn't belong to A&AI inventory (exo-inventory data). A common services data layer architecture

USAGE: It will enable Use Cases, Change Management, facilitate real-time operational LCM, and allow for OSS management.

KEY FUNCTIONS:

- Provides for a run-time repository of data such as configuration data, operational data

- Stores exo-inventory information

- Sync data (future)

- Allow for ONAP platform components and plug-in to access data

- Allow for controllers and A&AI to update RunTimeDB

- Allows for a "golden" parameter templating (future)

- Enables OSS configuration, Optimization, and LCM operations (e.g. RF Optimization)

- NF recovery from crashes (restoration), history, NF audits (auditing) (future)

- A&AI xNF presence Sync.

Purpose of Runtime Config DB

Component Description (Architecture Wiki):

<https://wiki.onap.org/display/DW/ARC+RunTime+DB+Component+Description+--+R6+Frankfurt>

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, Syncing, Indexing Runtime Config DB

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 Objectives in R7 GuiLin

R6 Objectives:

EPIC #1 - Partner with Use Cases in R6 that would need this capability (e.g. 5G OOF/SON/PCI) ... a solution was implemented in R4 (ConfigDB/MariaDB)

EPIC #2 - Provide a real-time instance of a 5G RAN network configuration data (example use case)

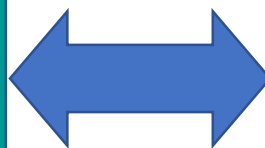
EPIC #3 - Create “base” RunTime Config DB component

EPIC #4 - Develop the Netconf/yang/O1 interfaces pathway to update RunTimeDB
Develop VES event notification/VES Collector (DCAE)/VES Config Change Event Notification pathway to update RunTimeDB

Dependencies vs Scope

DEPENDENCIES – need to operate

SDC Yang Model (to load schema)
ability to process & translate yang models into schemas
AAF (intra-ONAP security)
Database implementation for Data Persistency
(for example MariaDB)



DEPENDENCIES – value added

DMaaP (some use cases to work / indirect dependency)



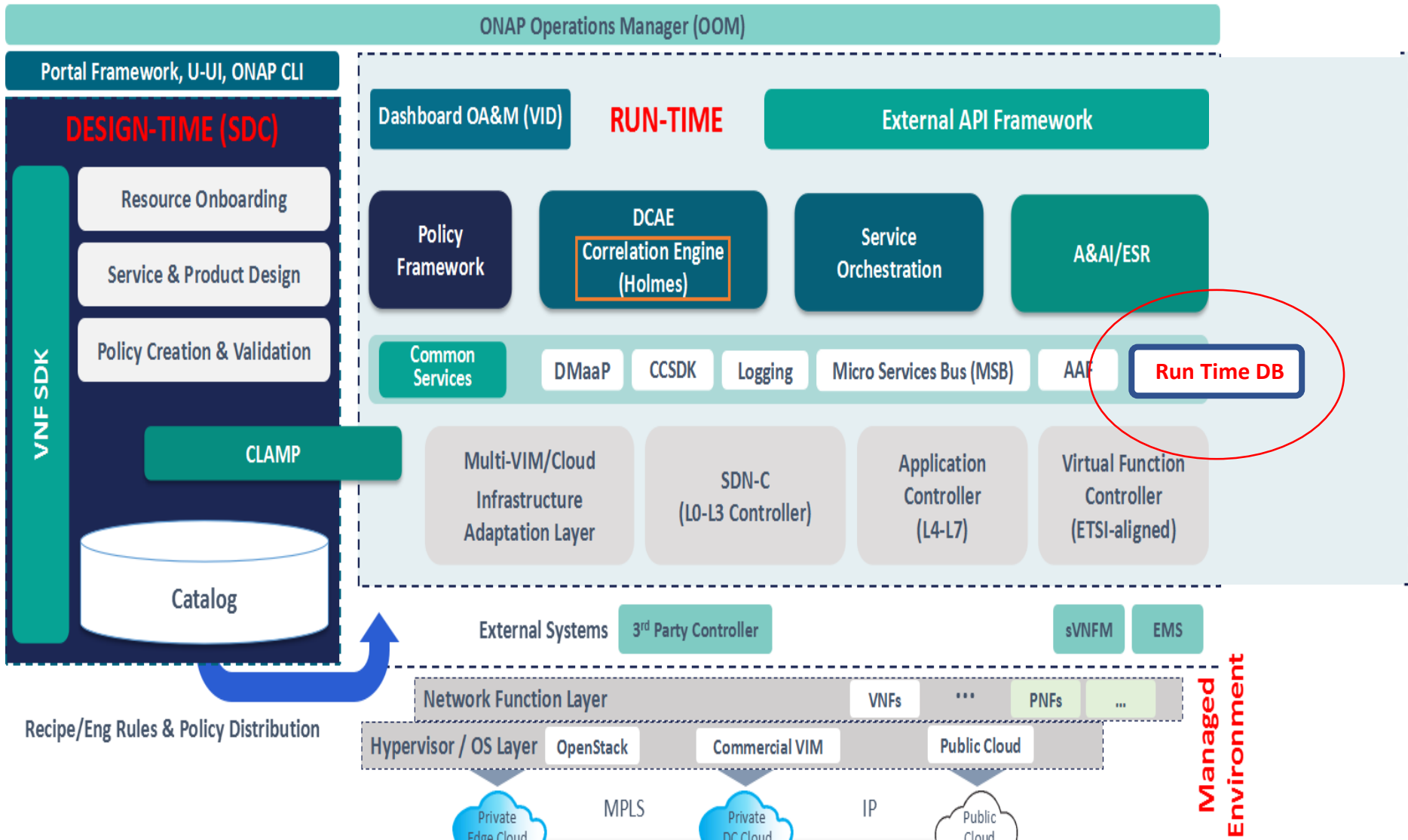
SCOPE



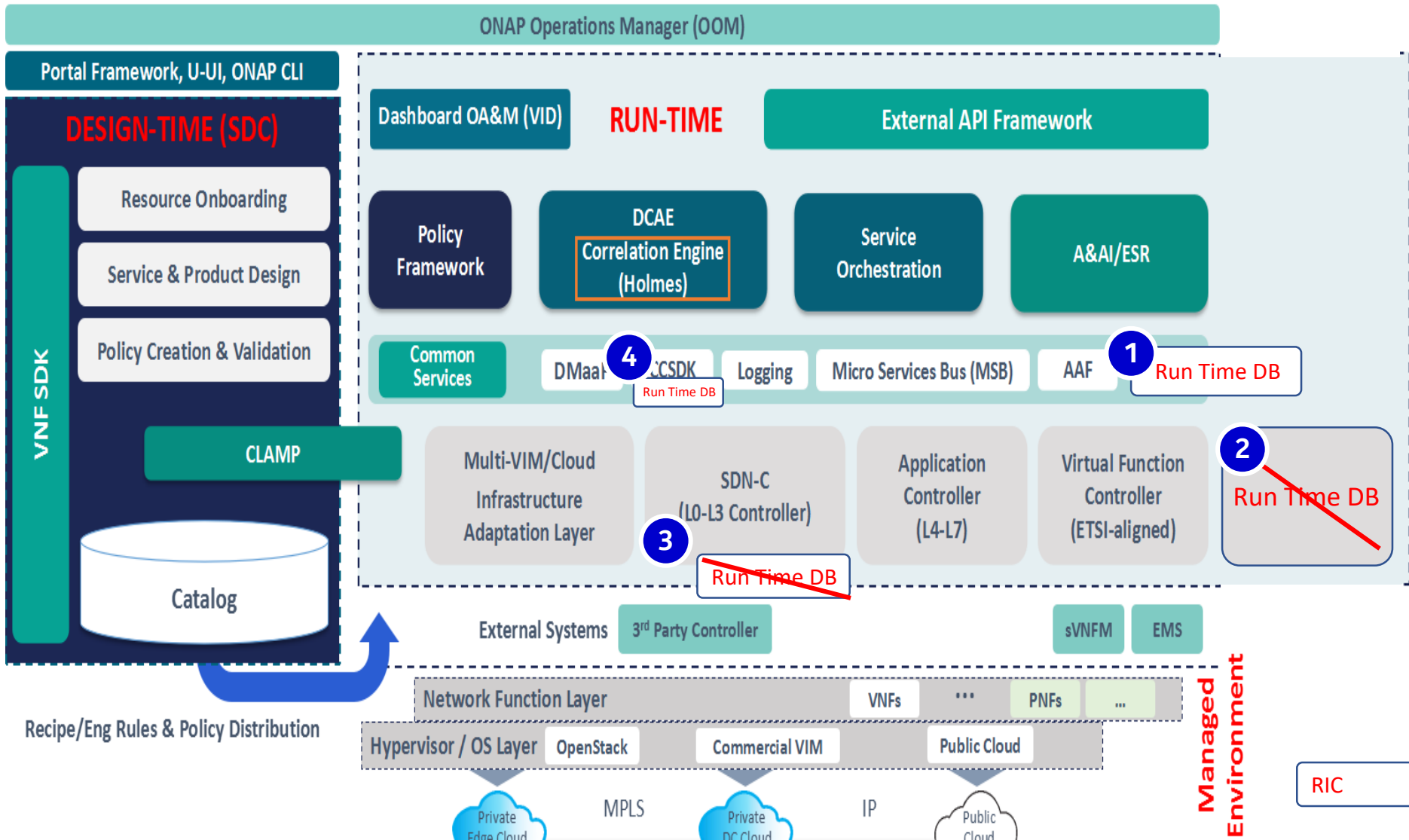
**RunTime
Configure DB**

RECEIVE INFORMATION
WRITE INFORMATION
PUBLISH CHANGES
REFERENTIAL INTEGRITY
INGEST PACKAGES
LOGICAL OBJECTS
ASSOCIATIONS
CARDINALITY RULES
LINKING RESTRICTIONS
SYNCHRONIZATION
DATA INTEGRITY & RECOVERY

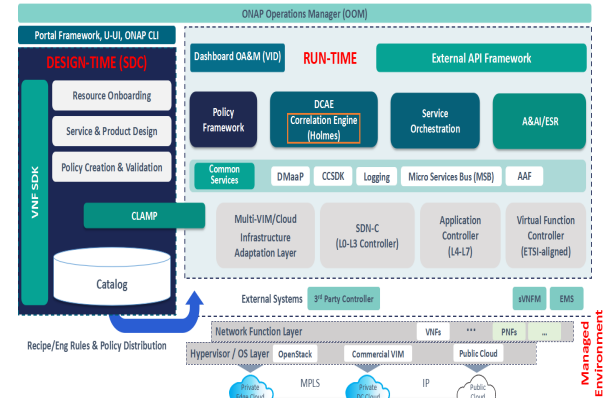
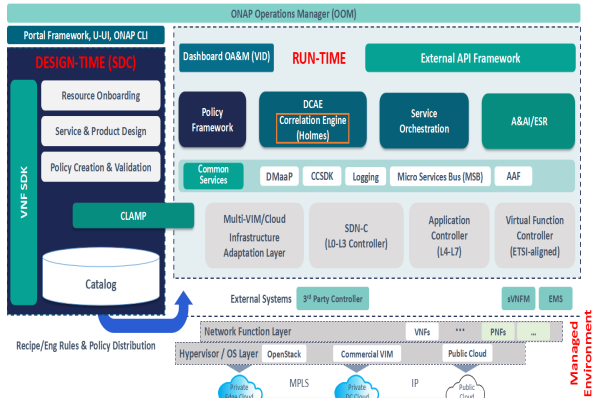
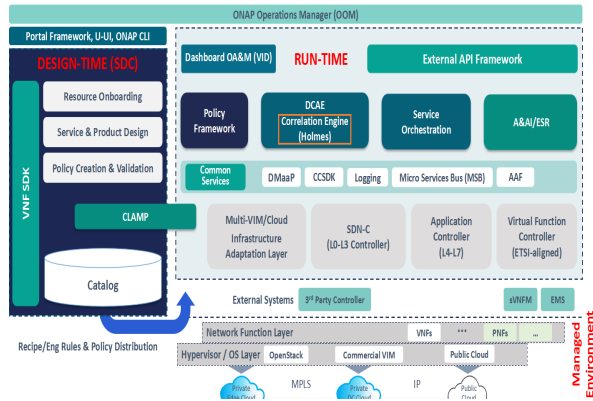
RunTime DB ONAP Component



RunTime DB ONAP Component

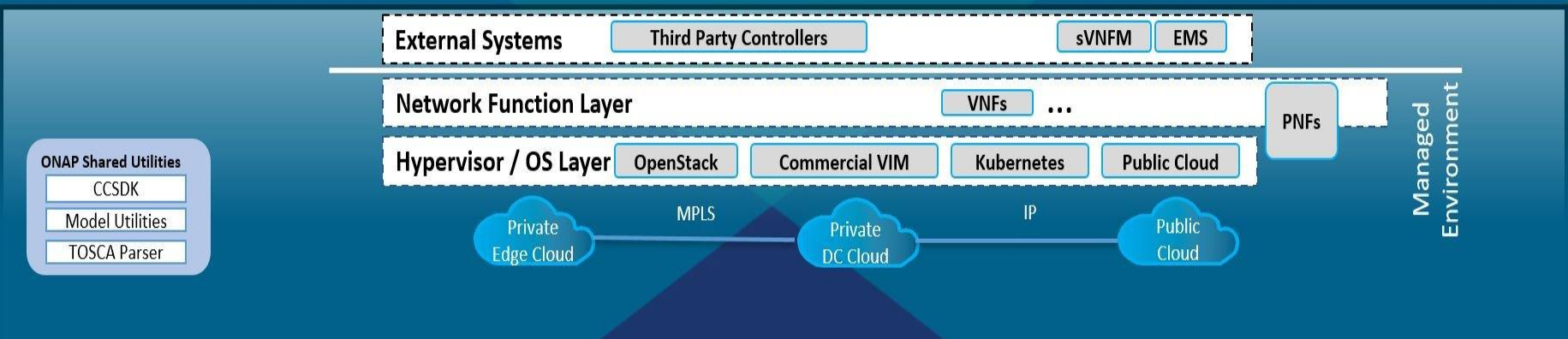
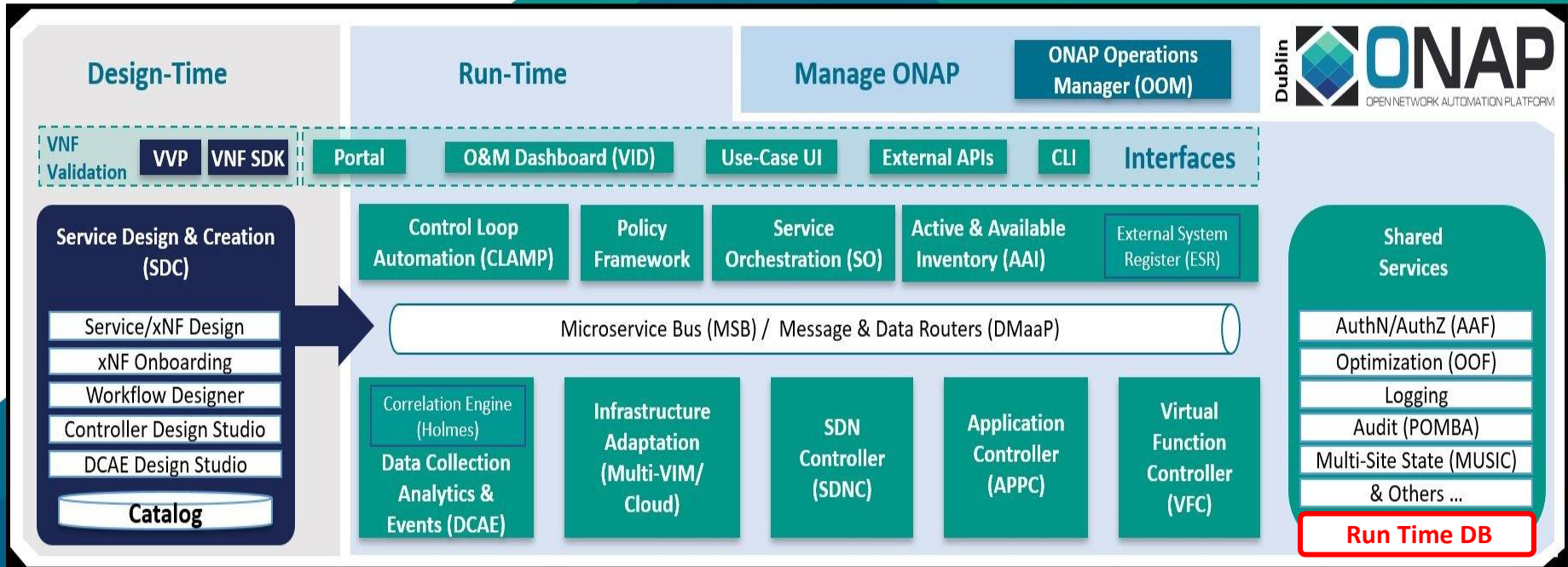


Edge ONAP Deployments

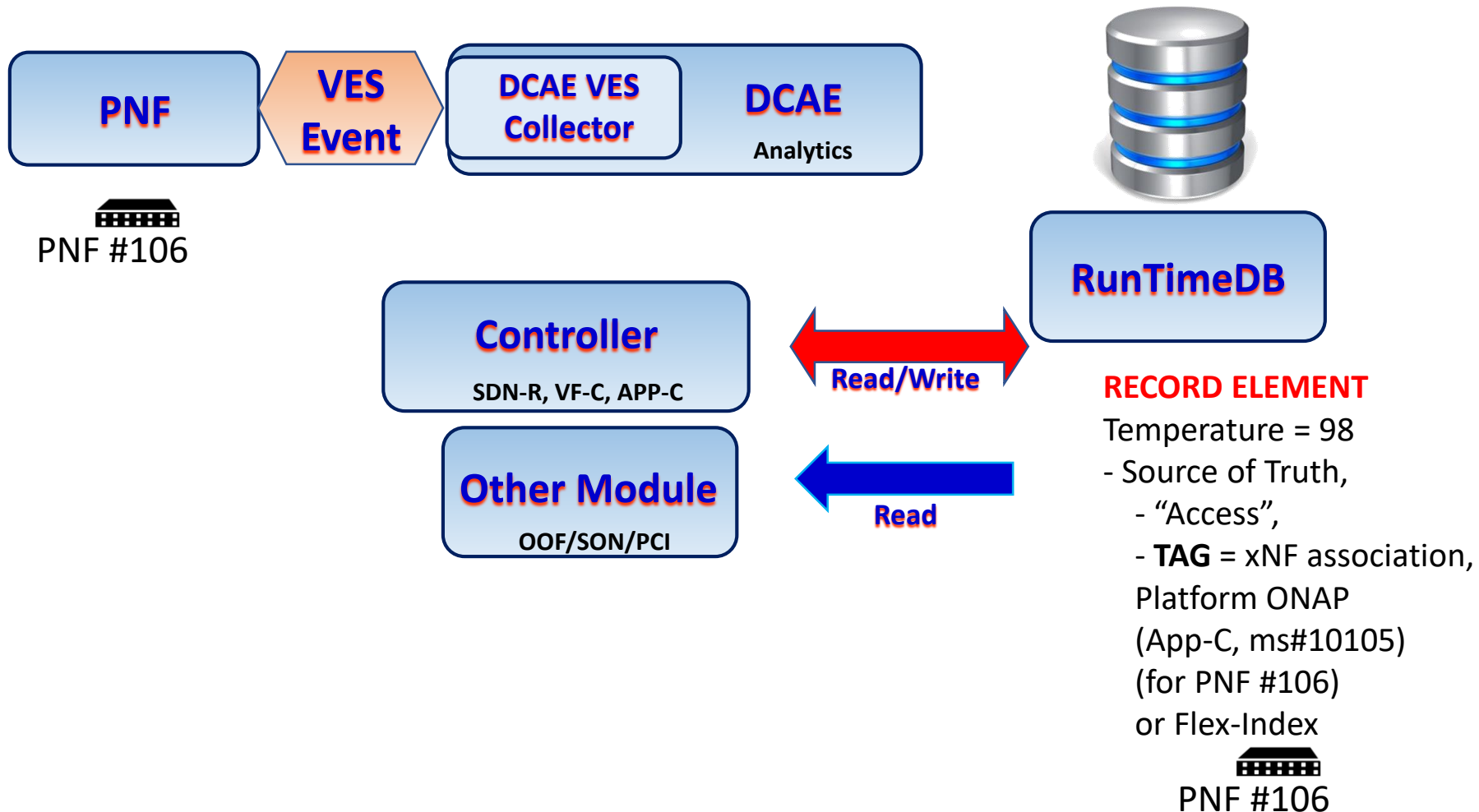




OSS / BSS / Other

Legend **Design** **Orchestration & Management** **Operations**











RunTime DB (Run-Time View)









-  PNF #101
-  PNF #102
-  PNF #103
-  PNF #104



-  PNF #101 
-  PNF #102 
-  PNF #103 
-  PNF #104 

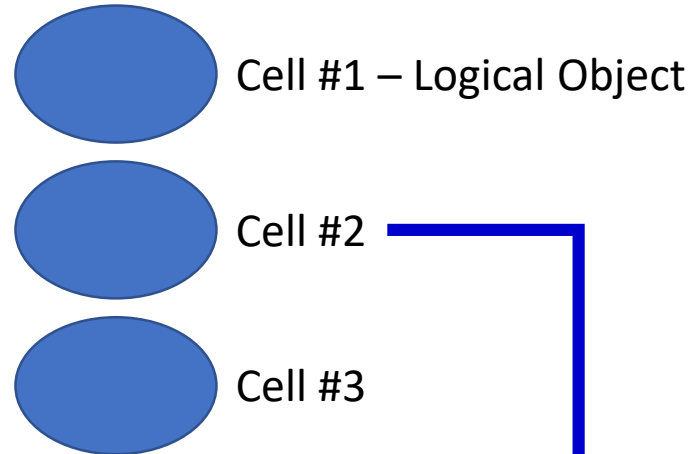


-  PNF #101 
-  PNF #102 
-  PNF #103 
-  PNF #104 

A&AI correlated/Index to RunTimeDB
Publish changes in A&AI, notification on DMaaP

Indices into RunTime DB may also use Flex-Index (such as CellID)

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

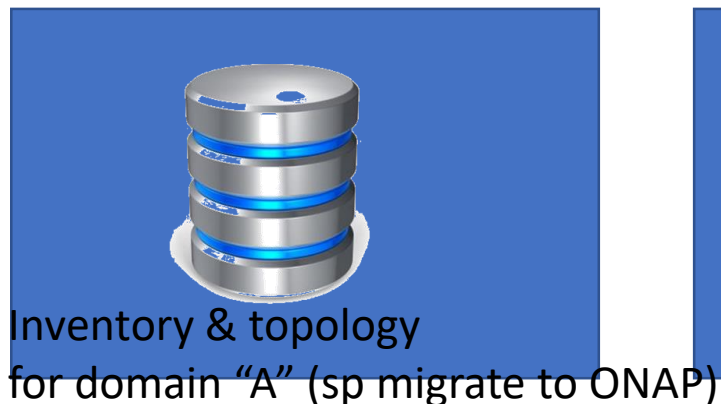
RECORD ELEMENT
INDEX = Logical Object #111
Parameter #1
Parameter #2
Parameter #3
Associations
{ PNF #106 }
Cardinality Rules
Linking Restrictions

Migration Brown-Field Deployment

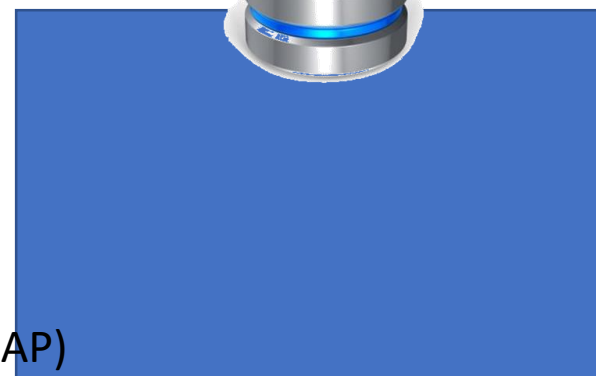
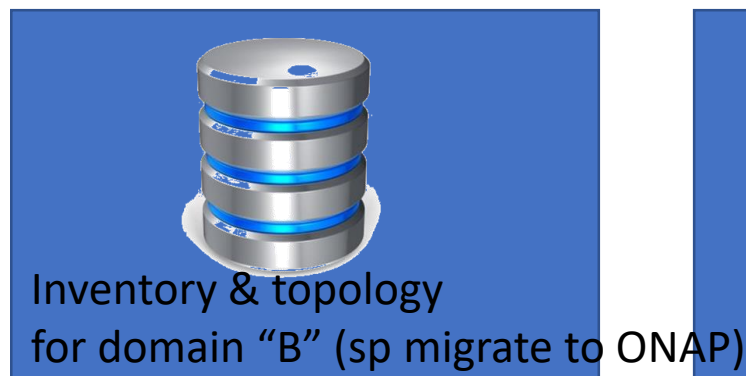
INVENTORY

NON-INVENTORY

Example Wireless
Domain "A"



Example Non-Wireless
Domain "A"



Not-ONAP Within carriers multiple Database -> or moving toward ONAP
If we define RT DB assist in the unification of multiple
RunTime

INVENTORY

NON-INVENTORY

Example Wireless
Domain "A"

ONAP - A&AI

ONAP RunTime Configure DB

POMBA

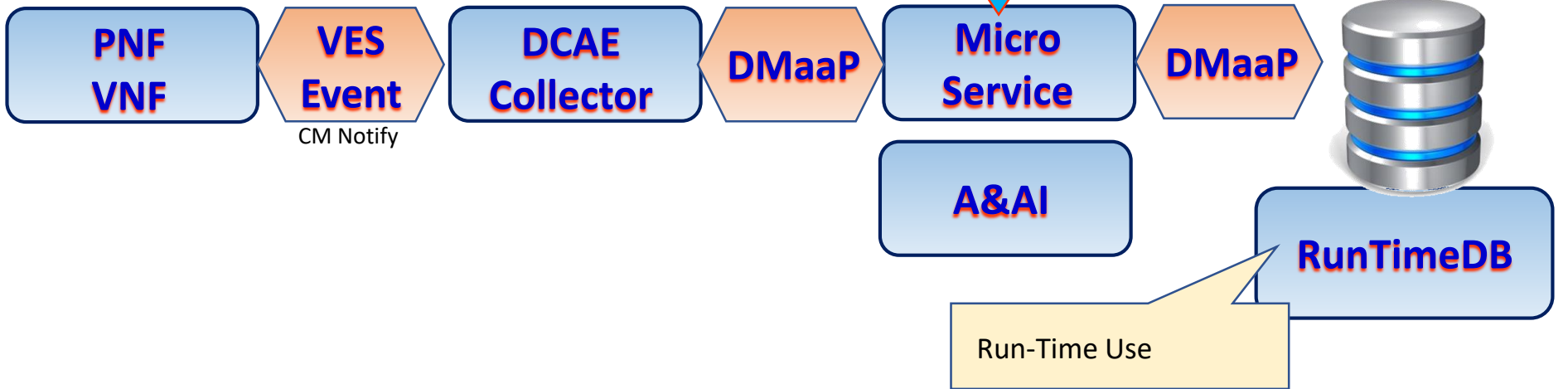
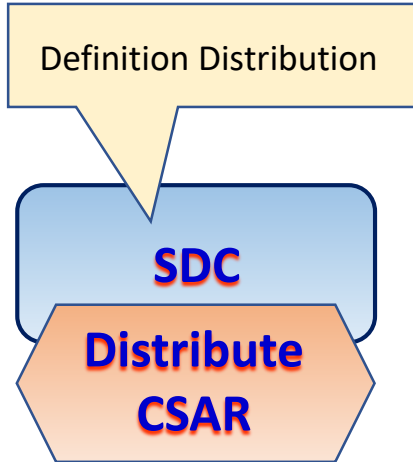
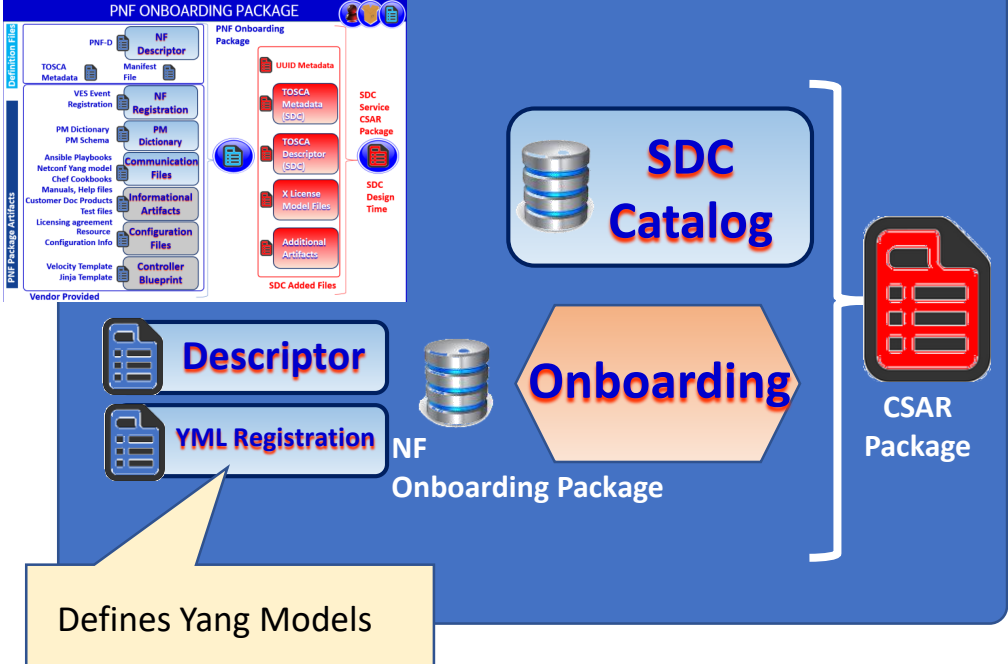


Example Non-Wireless
Domain "A"

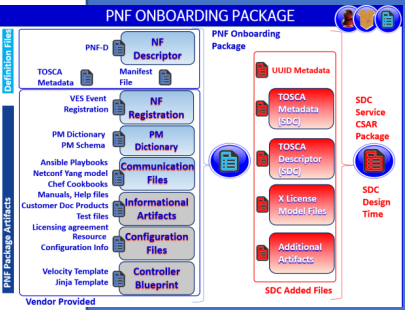
ONAP - A&AI

Not-ONAP Within carriers multiple Database -> or moving toward ONAP
If we define RT DB assist in the unification of multiple
RunTime

DESIGN-TIME (SDC)



DESIGN-TIME (SDC)



Platform Info Model



SDC Catalog



Onboarding



NF Onboarding Package

Descriptor

YML Registration

CSAR Package

DMaaP

xyz
x

RunTime DB Info Model



RunTimeDB

ORAN Yang model

Elec-Tilt-Sector 1
Mech-Tilt-Sector 1

Analyze the Yang Model Properties
MySQL MariaDB data model
Converter from Yang Model > M Group container relationships gr

```

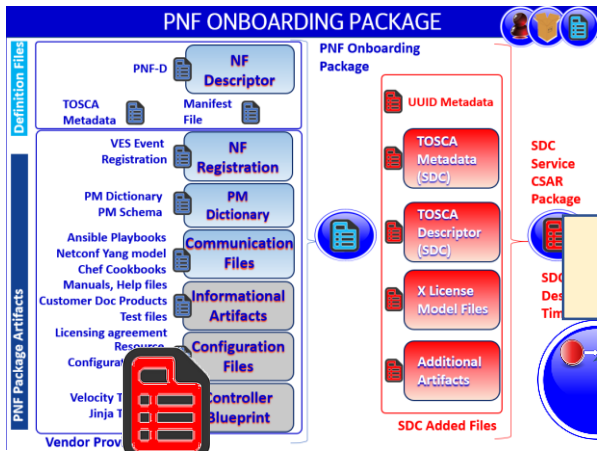
E
N
module example-5GNodeBParameters {
  namespace "http://example.com/example-runtimeDB";
  prefix 5GNodeB;
  import ietf-yang-types { prefix yang; }

  typedef product {
    type string;
    description
      "RunTimeDB ONAP 5G NodeB Parameters";
  }

  container RunTimeDBOnap5GNodeBParameters {
    config true;

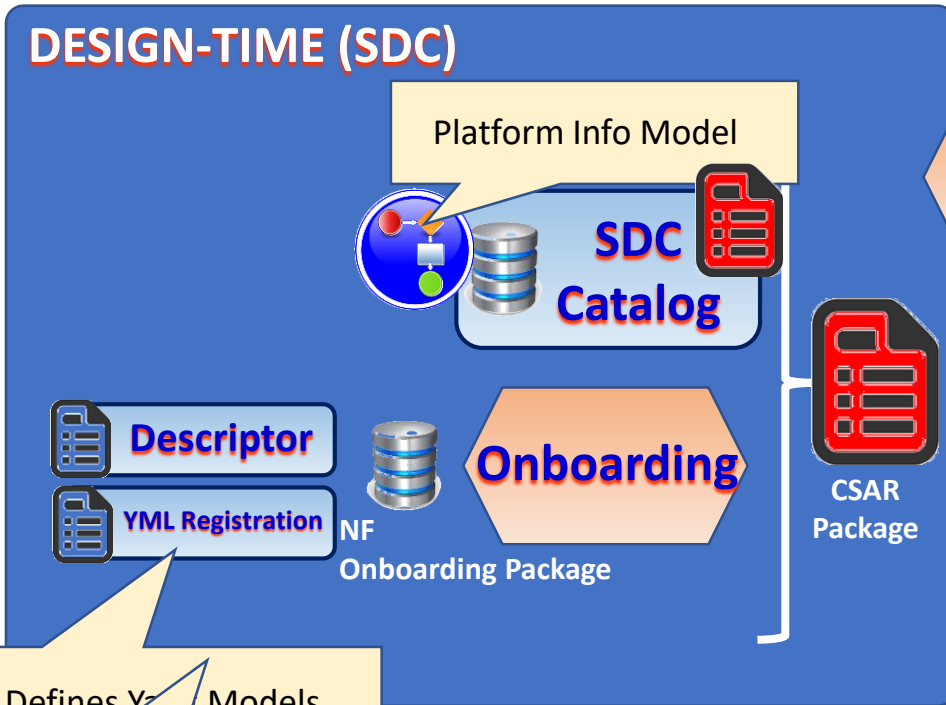
    list parameter {
      key parameterkey;
      leaf networkID { type string; }
      leaf CellId { type string; }
      leaf pciValue { type uint64; }
      leaf lastModifiedTimeStamp { type yang:date-and-time; mandatory true; }
      leaf pnfId { type string; mandatory true; }
    }

    list NeighborList {
      key parameterkey;
      leaf CellId { type string; }
      leaf TargetCellId { type string; }
      leaf Handover { type string; }
    }
  }
}
    
```



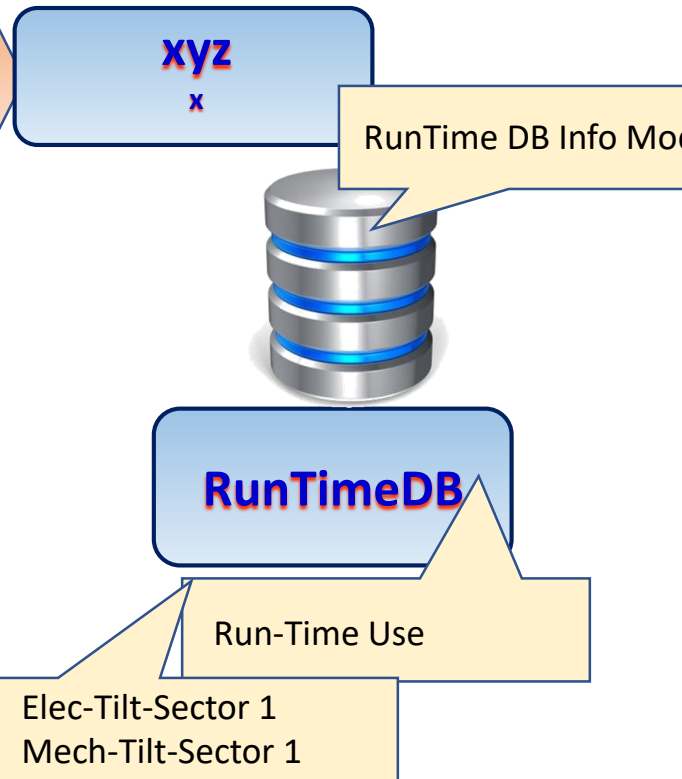
PNF-D (Vendor Model)

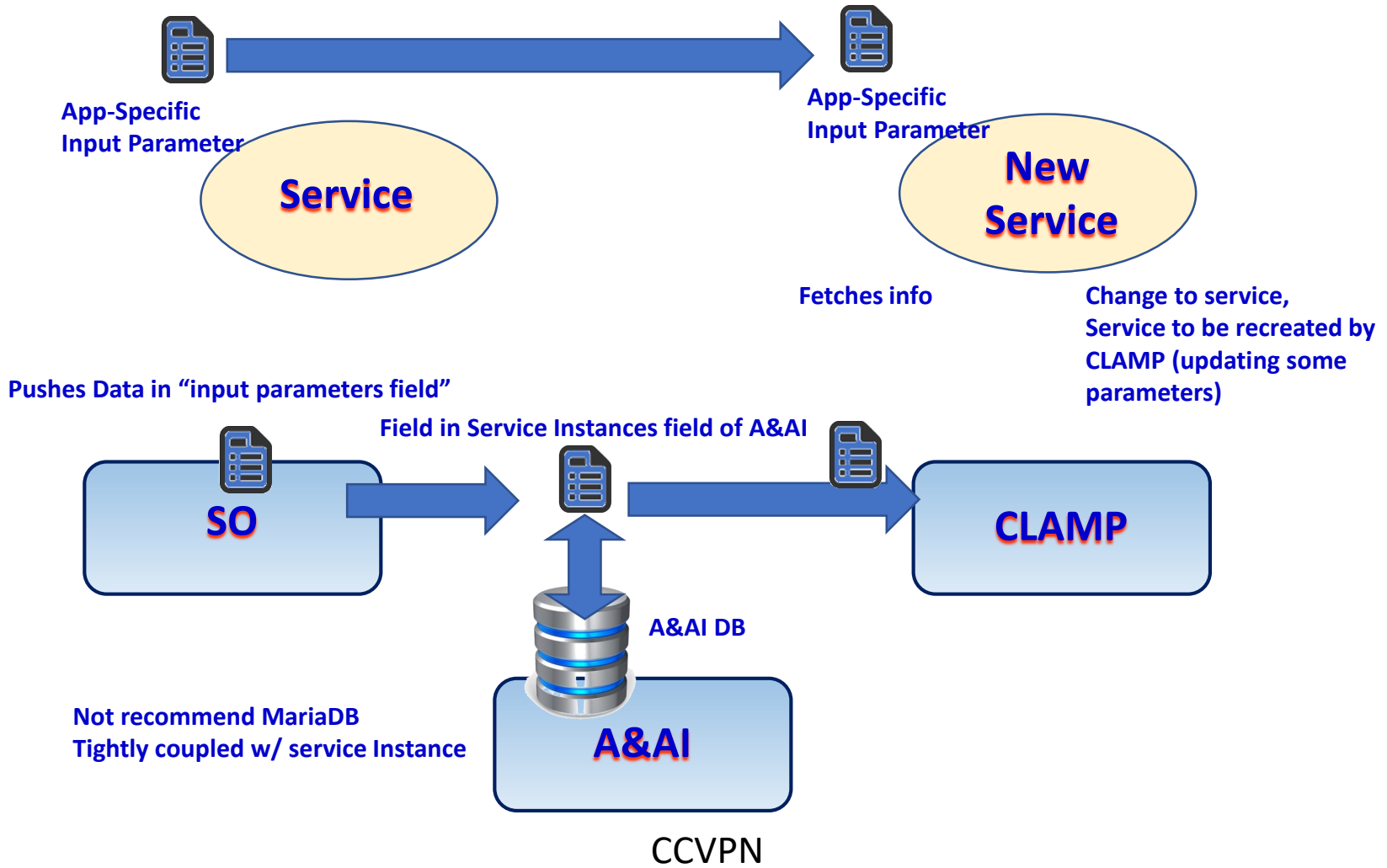
Vendor package
 HEAT template
 SDC attribute in internal CSAR data service model
 Structure > table in SDC design
 Internal data structure



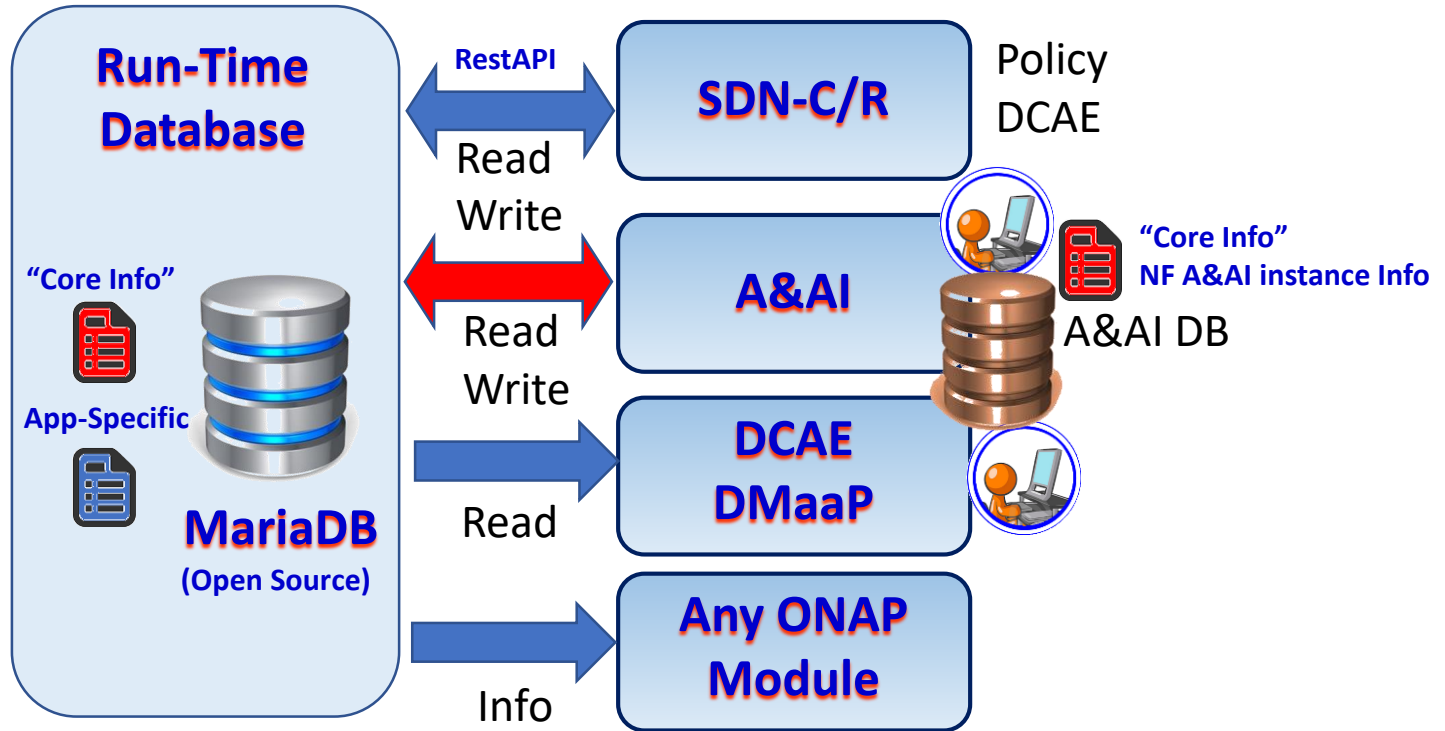
Defines YML Models

Elec-Tilt-Sector 1
 Mech-Tilt-Sector 1





- TOPICS:
- Syncing
 - Run Time
 - Alert/Updates
 - Race Conditions
 - New NFs
 - Ground Truth
 - State tracking



Query Interface (used by any module)

Post-gress database
Relational Database, Scaled

Sandeep Shah / SDN-C Jira-431

ACTION – Architecture Present Proposal

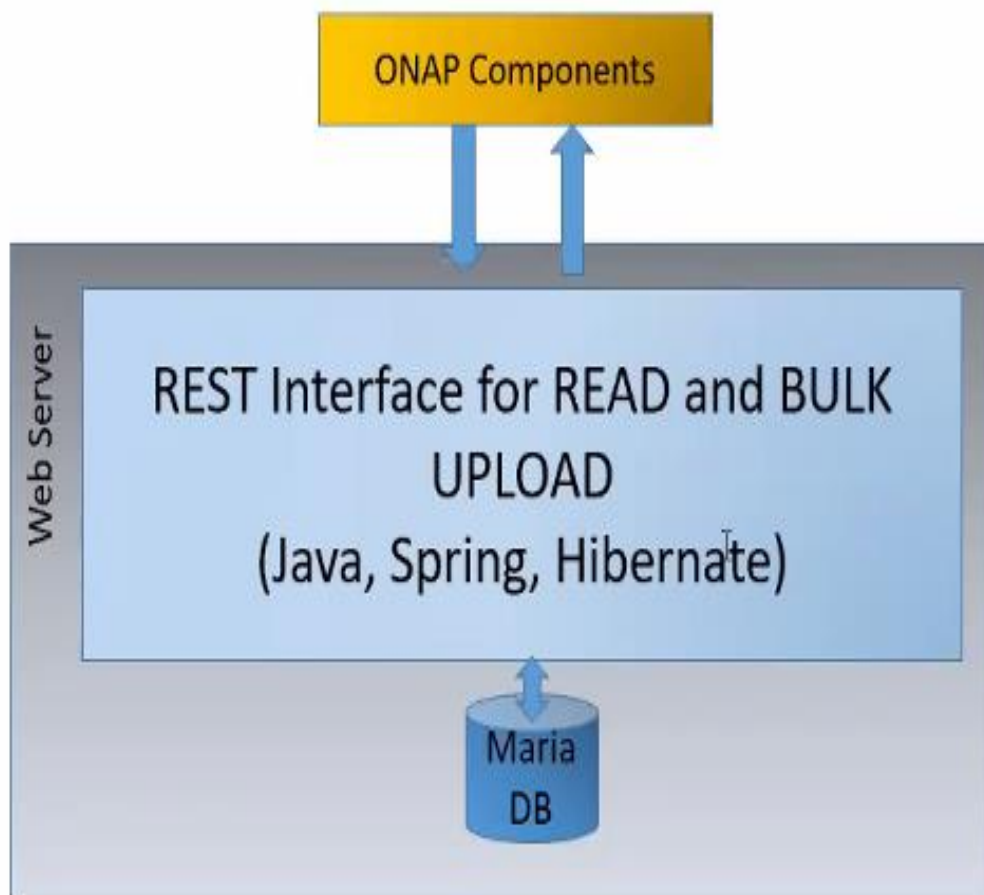
ACTION – Topics

ACTION – 3 Use cases ... change S/W

USE CASES	DESCRIPTION	ITEMS STORED
CCVPN U/C	Producer Consumer Pair	Input Parameters field Service Instance Info
OOF & PCI U/C	Maria DB	Config Values, PCI values
BBS U/C	(Independent solution)	ONT PON UNI (Reregistration), PNF resource

ConfigDB System Architecture

- Implemented to maintain configuration information of objects (e.g. cell) associated with the RAN network
- Initial version in Casablanca; will support all CRUD operations in Dublin (including PUT, PATCH)



Swagger API JSON spec (Casablanca)

GET API's

- GET /SDNCConfigDBAPI/getCellList/{networkId}/{ts}
- GET /SDNCConfigDBAPI/getPCI/{cellId}/{ts}
- GET /SDNCConfigDBAPI/getNbrList/{cellId}/{ts}
- GET /SDNCConfigDBAPI/getPnfName/{cellId}/{ts}

Bulk Upload API

- PUT /SDNCConfigDBAPI/insertData

Config-DB Data Model and APIs

- Config DB (MariaDB) used by PCI-H-MS (step 4b) and OOF (step 7)
- Query API (swagger JSON spec) exposed to other ONAP modules
- cellId needs to be globally unique (assumed eCGI) and align with ONAP YANG model, ORAN, 3GPP
- pnf-name indicates netconf server to be used for interactions regarding cells
- Pnf object (pnf-name, pnf-id) to be aligned with A&AI (A&AI/ConfigDB interaction to be finalized in Dublin release)

Cell (Object)

Attribute	Format
networkId	string
cellId	string
pciValue	uint64
nbrList	list of cellId
lastModifiedTS	timestamp
pnf-name	string

pnf (Object)

Attribute	Format
pnf-name	String
cells	List of cellID's
lastModifiedTS	timestamp

ConfigDB API

API	Input	Output
GET cellList	networkId, ts	List of cellIds
GET PCI	cellId, ts	PCI Value
GET nbrList	cellId, ts	List of cellIds and their PCI values
GET pnf-name	cellID, ts	pnf-name



MEETING NOTES

Benjamin Cheung

Sept 10

Potential Use Case / Usage

ORAN / If ONAP is host the non-RT RIC function / analytics ML processes that can based on

Data determine that ONAP should send policies to a policy aware NF in the RIC across

A1 then a way to model that would be the things in ONAP the mS would have access to a data store. mS not aware of each other, put policies to RIC, each individually writing policies to a repo scoped to a RIC instance. Something in ONAP taking policies sync across A1 into RICs. Infrastructure layer svc to mSs. If syncing info that is policy, a policy repository vs a controller repo. A neutral place. Federated close to the RAN couldn't be central. Regional / edge.

Variety of Info; part of Shared svcs; used by policy- controller – DCAE.

The DB should store being used for 5G RAN (1st application OOF/SON/PCI U/C). Posts data to DMaaP, nonRT RIC gets info from DMaaP.

Introducing a means to extend I/F between components in an uncontrolled way. Who owns the model; who can read the models.

Modeling availability. Govern the RunTime DB. Many U/C share data some sharing, others reading. What are the models behind the end-points. Between model component architecture. If buried in a controller this would be controller-specific would be difficult to manage across the components in the ONAP family. If the usage is only for the controller; but if the usage is broader then it shouldn't be part of the controller. Conway's law, if you stick something into a component; that component's I/F are favored. Pros/cons

Analytics MS can change : configuration. Not only thing that can change configuration controller would set initial default config; operations can change config. Set limits – controller understands what is the config which parts are sensitive. Controller terminates I/F. POLICY – the opposite; the controller is the conduit; it doesn't understand the policies. The ML written policy to this table > RIC > VNF over A1.

Start: see definition; capabilities, interfaces.

Controller of Svc, controlling resources. Each controller does its part to restore NF. Svc controller; svc mgmt. function of ONAP is distributed. Svc management database. Mapp svc view resource view map (should) svc lvl req mapped to resource level data (audit bss). Should be someplace accessible by a controller, which instance would you put it in? Controller database instance.

Derived Values. Instance values. May need to be remembers

Service levels. Config/info not part of inventory. FW rules put in by WebURL. Read. State info of VNF. Information rcvs persistent to perform job incontext of svc. (1) ONAP needs to store run-time data. (2) should not be in A&AI. Should be globally accessible.

Data lake - State topo PM FM analytics RIC -

Sept 10

Architecture recommendation is as a Common Service

Would we want to make it mandatory – unless svc not available does it come up any svc can get instantiated w/ default values.
Can a svc “run” w/ wo/ the RunTime DB

Who has responsibility for creating? Could be a separate project or part of SDN-C (for example) ; project scope vs architecture.

(Yuriy) in CC-SDK moving to mS implementation for controller disag & scalable. Model design intent we want to provision push config on different NFs .idea of runtime DB can live wi/ CC-SDK as a mS. In CCSDK that are CDS related dealing w/ scope of config. Mgmt that is shareable entity in library that any controller persona can access. Parallel between data. Config

CCSDK design-time tool for deriving instance values.

Run-time config of values into the NFs.

CCSDK library set of repos performs for provisioning and configuration management domain. Naming.

CDS tool > model config svc. > model svc context design tool mS inside CCSDK.

ONAP personas. Edge ONAP. What components/ mS should be in there.

nonRT RIC – deploy Edge ONAP.

RIC is not working on it in ONAP community. nonRT RIC is part of ONAP.

Operational view.

(Chaker) CCSDK as a template/framework to instantiate to control an application to control a layer3 network wi/ that framework you have a database maintains information state/ wi. Controller.

NEXT STEPS: Draw a functional description; as a component release-F repository; who are users.

Controller – continues to use xyz does this replace controller database

Define Epics -

<https://wiki.onap.org/display/DW/ARC+AAI+Component+Description+-+Dublin>

Flows

Database as a service

(collecting all model platform database)

Share Casadrana shared maria-db

DATA BACKUP & SYNC & GEO-REDUNDANCY

- Sync – can the system continue to function if runtimeDB goes away.
- (body data) xyz = 5>10 [crashes] ->
- (sync operation) -> ONAP “rebuilding” DB.
- who is the source of truth?
 - A1 ONAP Policy, CLAMP (reassemble KPI); A2 xNF (poll/query)
- geo-redundancy (ONAP multiple installations)
- similar solution with A&AI, and other ONAP components.
- BACKUP function ... f(BackupRTDB)

ROLL-BACK & HISTORY

- View of config data, roll-back, keep track of config & historical view.

ACCESS & CONTROL

General purpose DB, specific service each component, integration between ONAP component

- component A > xyz =10
- component B > xyz = 7
- how does component A know about “xyz”? A: onboarding, define design time.
- defining xyz > xNF, ONAP, micro/component “access”? Model (xNFD)
- Administration control, access rights

SMARTS “INTELLIGENT” DATABASE

Does the RTDB “understand” the data or is it just a data storage. E.g. A&AI stores data but also “understands” the data. “outsource” who has control of that data. Yang model/Design time/Run time gets CSAR definition/artifacts.

Logical connection of data.

OBJECT MODEL

“cookie” structure every S/C can interpret the blob independent of other S/C, to track where you were during previous execution. – vs – well structured DB that every component would have to follow the object model.

“Parameter” object dynamic – Design time package. Design time model. PNF package onboard.



APPENDIX – A&AI BACKUP SLIDES

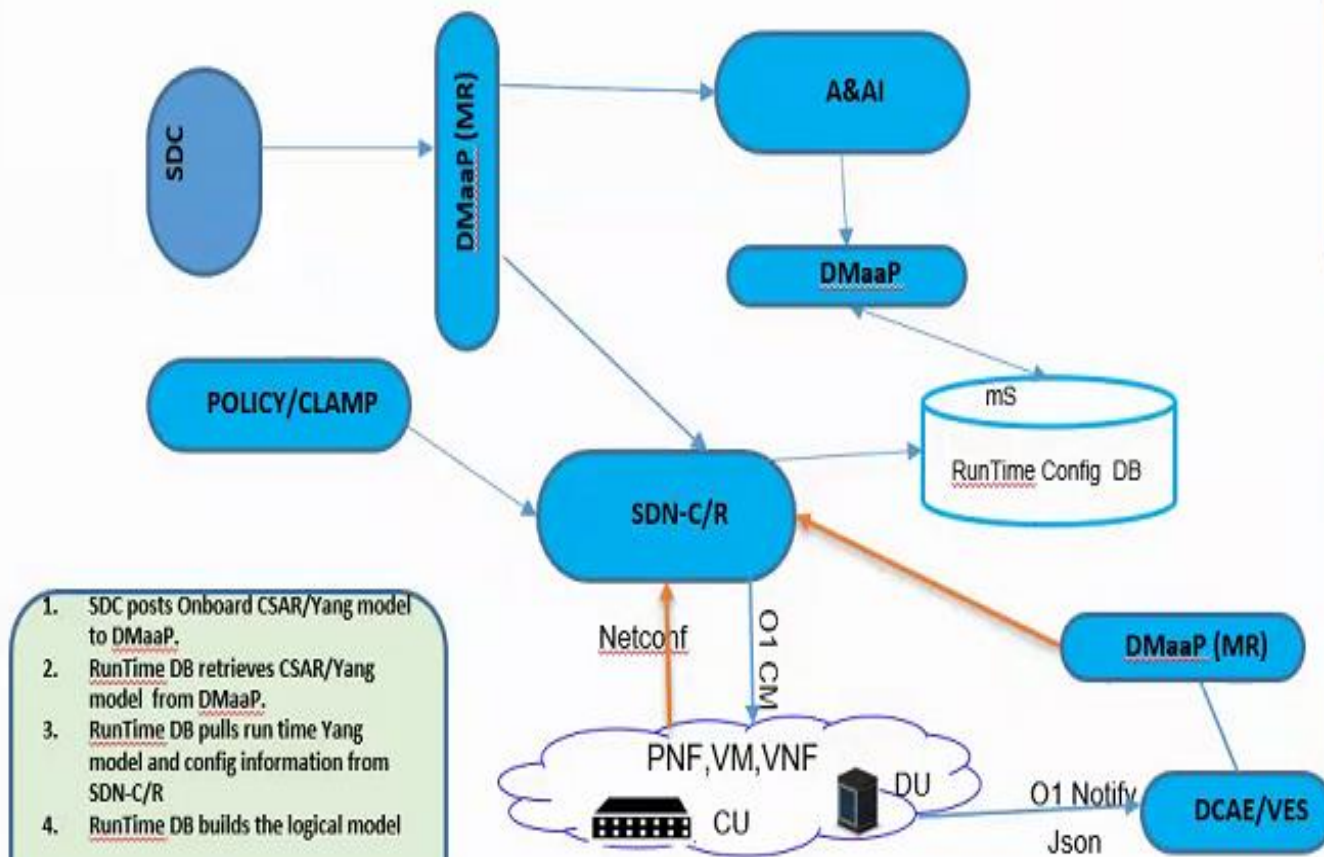


So
Closed Loop
Recreate svc instance object
"input Parameters"

Producer-Consumer Pairs

	Producer	Consumer	AAI Representation	Scenario and References
1	SO	Closed Loop	the "input-parameters" attribute of "service-instance" object described as: "String capturing request parameters from SO to pass to Closed Loop."	<p>Closed loop scenario:</p> <ul style="list-style-type: none">• SO will create "service-instance" object in AAI• SO will store "customer-request" string on service-instance object in AAI• When Closed Loop call recreates the "service-instance", it will query "service-instance" information first, to get the "customer-request" <p>References</p> <ul style="list-style-type: none">■ reference id AAI-1353-2 in AAI-CCVPN Schema Proposal for Casablanca Release■ AAI discussion on 2018-11-28 ExtAPI Meeting notes

RunTime Cofig DB



1. SDC posts Onboard CSAR/Yang model to DMaaP.
2. RunTime DB retrieves CSAR/Yang model from DMaaP.
3. RunTime DB pulls run time Yang model and config information from SDN-C/R
4. RunTime DB builds the logical model

- RunTime DB(Config DB)**
1. Real time config data
 2. Realtime logical connection
 3. EXO-Inventory
 4. Netconf/O 1 CM and VES CM event Change notification
 5. VES Collector and VES Config change event notification

- CM Notificaiton**
1. Device can send a CM notify event to DCAE VES Collector
 2. DCAE VES Collector receives CM notification and publish the CM event to Dmaap MR
 3. RunTime DB shall subscribe this event for config data and store Config DB based on Scheme in Config DB.

Courtesy to valuable inputs from David Kinsey & Alok Gupta on CM notify

Webex Meeting Reminder

Yang model to ?

Run Time Database (DB) Component

Data	Deployment	Access	Other
<p>Types:</p> <ul style="list-style-type: none"> - Related to services and network functions, but is not inventory - Configuration data - Historical - Golden templates - Normalized? 	<p>Central. A single centralized deployment of the runtime DB</p>	<p>CRUD Query REST (Swagger/OpenAPI)</p>	<p>Mastership. Configurable master (DB/External) with support for reconciliation</p>
<p>Structure:</p> <ul style="list-style-type: none"> - Hierarchal - Tabular - Connected 	<p>Distributed</p> <ul style="list-style-type: none"> - Multiple federated? - Partitioned? - Hierarchal? Instances - Read replicas? 	<p>Access control</p> <ul style="list-style-type: none"> - Object/attribute - Target - Role/application 	<p>Sync:</p> <ul style="list-style-type: none"> - Bulk (Netconf) - Attribute Value Change (VES) - AAI (ONAP REST)
<p>Dynamic:</p> <ul style="list-style-type: none"> - Schema - Data sources (AAI; xNF; Controllers; ...) 	<p>Dynamic deployment of models/data – flexible partitioning</p>	<p>Model driven</p>	<p>Autonomy</p> <ul style="list-style-type: none"> - Remove data related to removed inventory - Automate reconciliation

ONAP RAN Database (Runtime Config DB)

- Objectives:
 - Build upon 5G SON RAN Database started in Dublin & Frankfurt
 - Collaborate with (new) Rel 7 Runtime Config DB project to make this a shared ONAP functionality
- Dependency
 - Availability of O-RAN yang models and VES specification
 - Runtime Config DB, DCAE

	Frankfurt (<u>Rel 6</u>)	Guilin (<u>Rel 7</u>)
RAN database	ConfigDB (part of SDN-C CCSDK)	Runtime Config DB (RCDB) (New <u>Rel 7</u> function)
Config (CM) notification	Pre-standard VES message	ORAN standard VES CM Notify message
CM Notify processing	<u>RANSim</u> -> DMaaP -> SDNR	<u>RANSim</u> -> DCAE -> DMaaP -> SDNR
CM update	SDNR -> ConfigDB	SDNR -> RCDB

