ONAP Use Case – Third Party Operational Domain Manager

Telstra

July, 2019

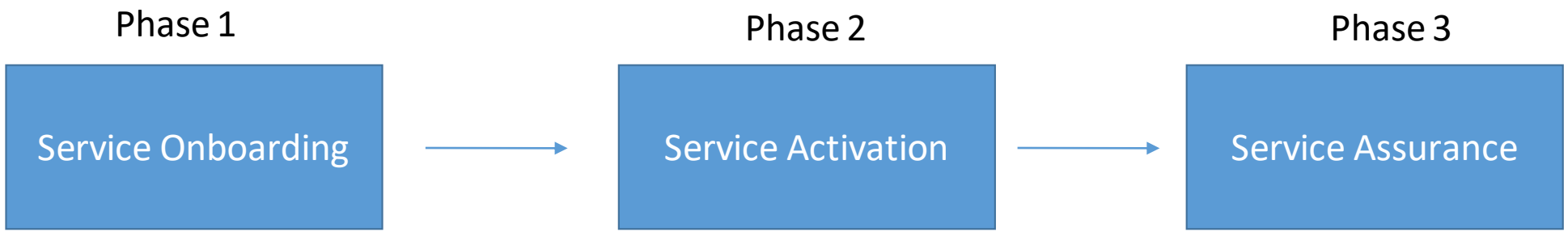
Telstra ONAP Use Cases

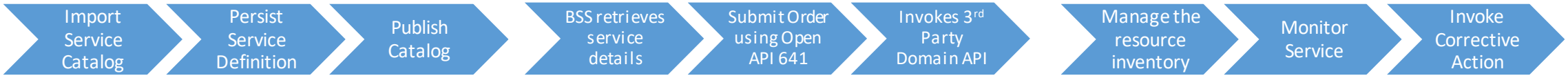
Telstra has multiple domains present under NaaS. These domains include external 3rd party domains and internal network domains. We are proposing to leverage capabilities of ONAP as an operational domain manager in order to provide services managed by these domains.

Below matrix depicts the plan for the use cases and different phases within the individual use case

Use Cases for ONAP as:	Phase 1	Phase 2	Phase 3
3rd party Operational Domain Manager	Service On-boarding (Planned for ONAP F Release)	Service Activation	Service Assurance
VNF Operational Domain Manager	VNF Inventory Management (Planned for future ONAP Release)	VNF LCM	VNF LCM contd...
Cross Domain Inventory Manager	Resource On-boarding (Planned for future ONAP Release)	Resource Catalog Management	ONAP as Master Catalog Manager

Use Case 1 – Phased Approach





Use Case #1 ONAP as “Third Party” Operational Domain Manager

Business Drivers

Executive Summary - In case of tier 1 / brownfield operators, it’s more likely that ONAP might need to interface with existing orchestration platforms for specific domains This use case will provide ONAP capability to be operational domain manager for third party services. Service providers will be able to use ONAP to provide end to end automation for composite or white labelled services which could be provided and managed by third parties. This use case will enable federated catalog and orchestration management.

Business Impact - The use case will provide capability in ONAP for seamlessly on-boarding services from partner (or specific) domain catalog. Lack of this capability today leads to manual creation of partner services in ONAP which is time consuming and error prone. With introduction of this capability, ONAP will be able to consume domain specific service definitions via Open APIs and publish the same to run time components. Next phase of this use case will extend the operational domain manager capabilities to support complete service operations value chain for “Third Party” or Domain specific services via federation.

Business Markets

- Potential candidates for Third Party Domains which can be supported by ONAP in this context:
  - Fixed Broadband Service from Last Mile Connectivity Provider
  - Managed Network Service from other service providers (Telco Peers)
  - Hybrid cloud ecosystem of private and public clouds from multiple cloud service providers
  - Special case could be composite services which include service components managed by a existing domain manager

•This use case is also relevant to service provider environment where all services are managed by single ONAP environment (e.g. If there is need to move catalog from dev / test to production)

•This will be very relevant for automation of digital services delivered via diverse 5G Ecosystem (B2B2X Models) for vertical industry solutions

Funding/Financial Impacts -

•This use case, once developed, can be used by any service provider deploying and using ONAP.

•ONAP as “Third Party” Domain manger will play a significant role in on-boarding partner domains in a uniform manner.

•Service definition from “Third Party” will be made available to service provider in few hours, consumable via an abstraction layer (optional) (NaaS in Telstra context).

•Once catalog is on-boarded ONAP can publish the service definition to other design time and run time components such that ONAP can support complete life cycle management of the service via federation

All this will essentially bring down time to market for partner services. Telstra is committed to drive the implementation of these capabilities in ONAP across next few releases.

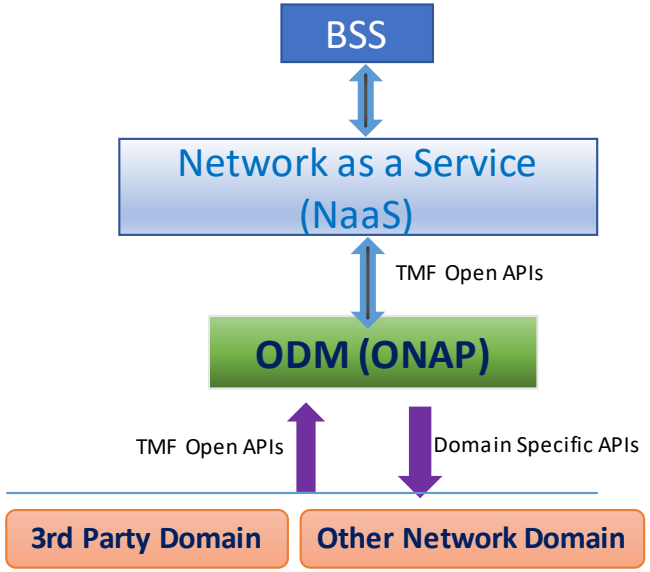
Organization Mgmt, Sales Strategies - There is no additional organizational management or sales strategies for this use case outside of a service provider’s "normal" ONAP deployment and its attendant organizational resources from a service provider.

Usecase#1: ONAP as "Third Party" Operational Domain Manager

Detailed View

Use Case Overview

- A standards-based approach that allows a service provider to have a network automation platform for composite or white labelled services managed by specific/ Third Party domain managers
- ONAP provides Operations Domain Management (ODM) and other complementary capabilities to ensure full automation of the E2E lifecycle management of the service via federation
- Services are exposed and consumed via Network as a Service (NaaS) which is an abstraction layer above the operational domains and exposes the services to BSS
- Consistent way of consuming 3rd party services for service providers like Telstra
- ONAP will facilitate service operations value chain for third party domain via federation
- Substitutes multiple handovers between parties/teams and applications to enable zero touch automation



An abstraction layer above the operational domains which acts as a gateway ; exposes the services to BSS

Operational Domain Manager to support Third Party Services via federation

Partner Domain – which can be another SP or another ONAP or non ONAP instance

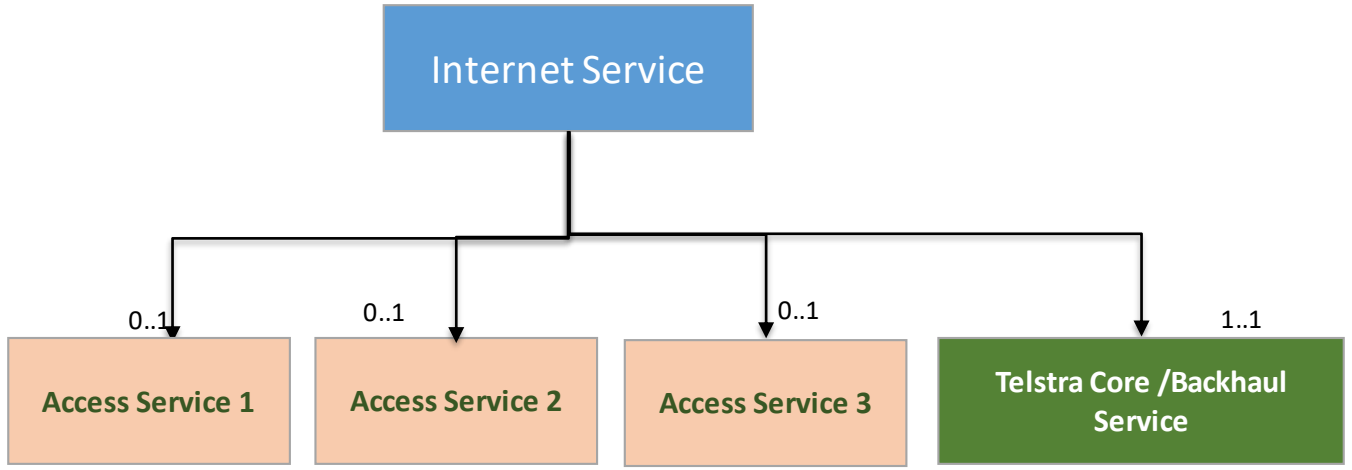
- **This use case implementation will be phased out across release F and beyond**
- **Initial Impact Analysis**
  - Service Provisioning
    - **SDC – Import 3rd party service definition, Publish catalog to run time ONAP components**
    - Any potential SO enhancements for orchestration
    - A&AI – add references on 3rd party domain inventory
    - Enhancements to Ext APIs (POST Operation for Catalog API)
  - Service life cycle management action associated with O2A and T2R must be supported via federation
    - CM: change management (capacity increase to meet scaling demands);
    - IM: Support incident management (problem identification and fix); and
    - VM: Any support (workflow/notification if any) for services associated with VNFs managed by "Third Party")
  - Remediation actions are triggered automatically using ONAP as ODM
  - Necessary references to service provided by "Third Party" domains will be made available to support C2M/P2O processes

Changes mentioned in Grey are proposed for subsequent ONAP Releases beyond Frankfurt

Beneficiary: 3rd party providers and Service Providers like Telstra

Sample Internet Service - Service Specifications

Mapping of Service Specifications of the sample Internet Service with the respective domain managers



TMF SID Framework Terminology
CFS – Customer Facing Service like Internet Access
CFSS – Customer Facing Service Specification
RFS – Resource Facing Service like Copper / Fiber Access
RFSS – Resource Facing Service Specification

CFS/Public Service
Managed by ONAP ODM

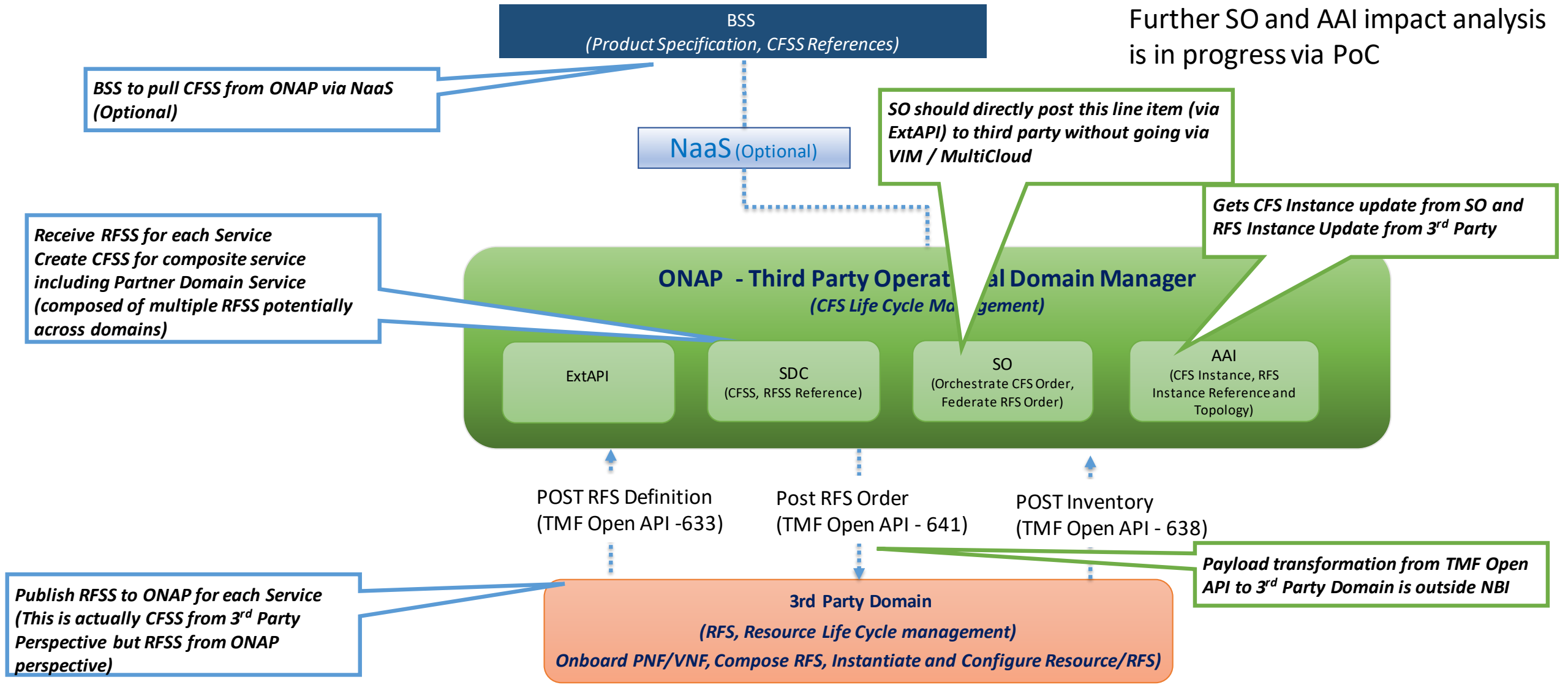
RFS/ Private Service
Managed Service from third party suppliers ;
Could be of different access types also like fibre / copper etc

RFS/ Private Service
Managed by Telstra's Domain

Guiding Principles Followed for this Use Case

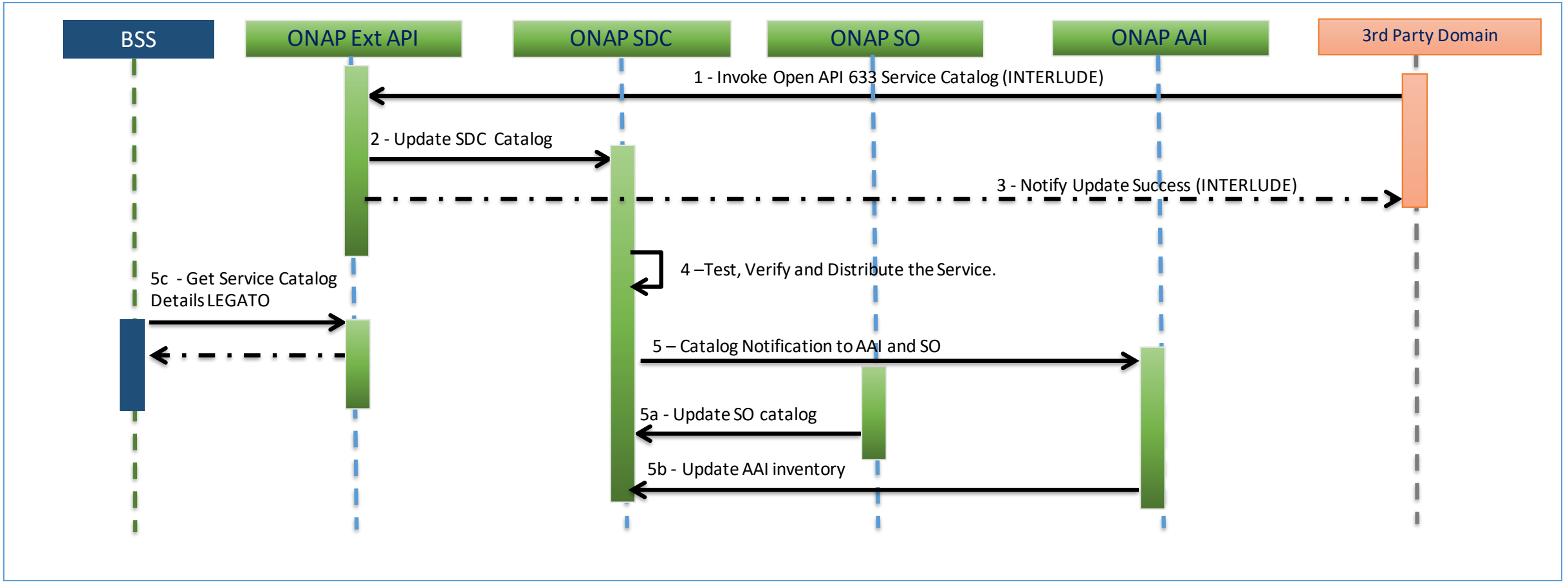
- **Minimize impact to existing ONAP Information Model**. (No impact to existing SDC model is foreseen based on the analysis done so far)
- **All communication from external application with ONAP must be via ExtAPI.** This is available today for Northbound Integration for Catalog/Order/Inventory. We would propose to extend this guidance for southbound integration as well.
- **Southbound Payload Translation** : Any order payload translation towards 3rd Party Domain Manager to stay outside ExtAPI.
- **Exposure of third party domain** : ONAP will communicate with third party domain and this will not be directly exposed to BSS.
- **Controlled access to ONAP SDC Catalog** – Only consumers defined in ONAP will have access to post service specification
- **Separation of Concerns** : Third Party payload for service definition will not have resource level deployment artifacts since resource management is responsibility of third party

Design Time and Run Time View



Further SO and AAI impact analysis is in progress via PoC

Flow Diagram for 3rd Party Catalog Sync– Design Time

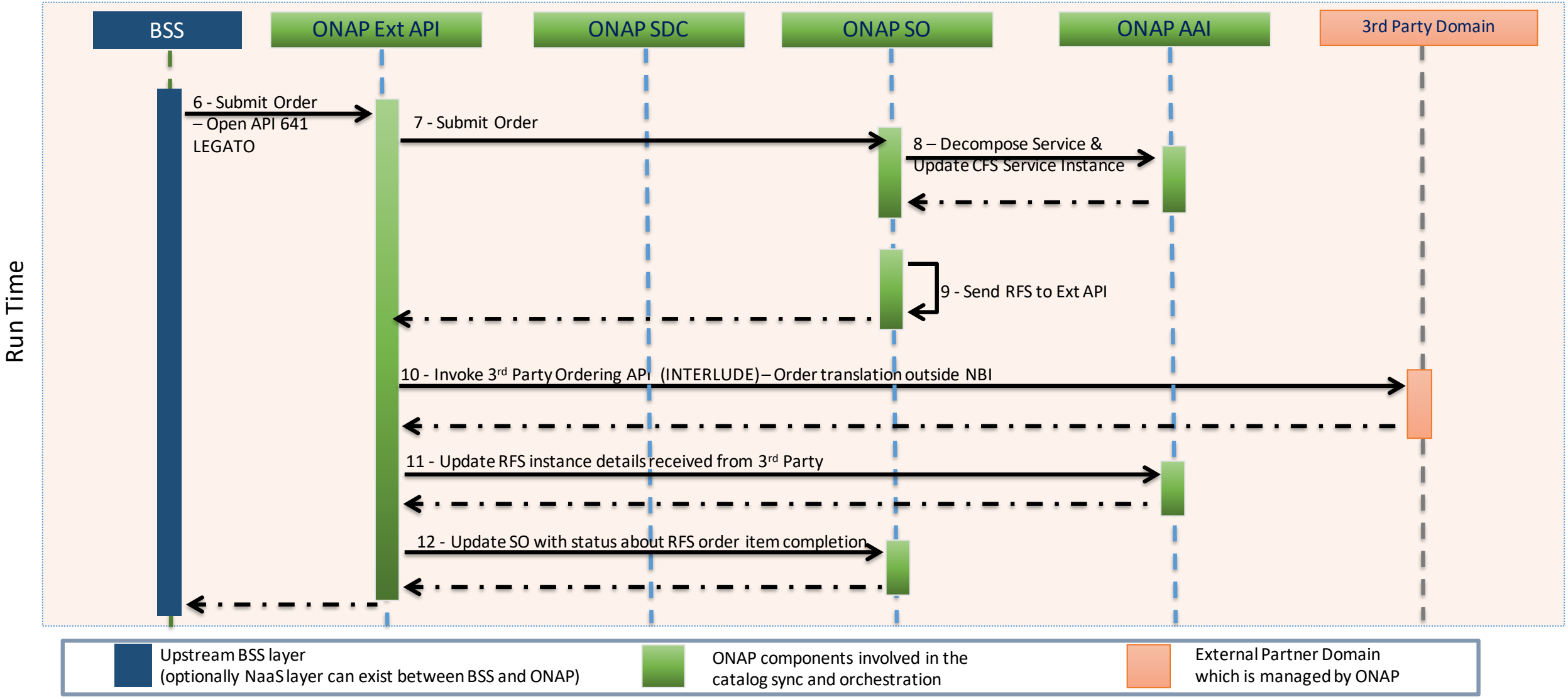


The flow steps (1 – 5c)

Catalog Sync Summary

1 – External Third party domain exports it service catalog details to Telstra. Telstra orchestrator ONAP exposes TMF Open API 633 Service Catalog API via ONAP Ext API component. Third Party Domain leverages the API 633 to POST the Service Catalog payload.
POST nbi/api/v2/serviceSpecification
Request body – TMF 633 Service Catalog compatible payload
Payload contents:
RFSS for Partner Domain Service
2 – ONAP Ext API updates SDC catalog by invoking internal SDC API
POST sdc/v1/catalog/services
3 – Ext API notifies Third party after successful update within ONAP
4 – Service Definition Updates / Creation of Composite Service happen in SDC UI (any manual updates to the received service definition)
Test, Verify and Distribute the Service definition. SDC updates other ONAP components (which have registered with SDC DMaaP) with catalog details
5a – SO pulls SDC catalog details
5b – AAI pulls inventory details
Ext API notifies northbound systems (BSS/NaaS) after successful import of the service catalog into ONAP.
5c - BSS retrieves catalog information from ONAP

Flow Diagram for 3rd Party Order Activation – Run Time



The flow steps (6 – 12)

Order Activation Summary

6 – BSS submits order using TMF 641 Service Ordering API, that is exposed by ONAP Ext API
7 – ONAP Ext API submits the request to ONAP SO
8 – ONAP SO decomposes the service, updates AAI with Service instance details
9 – Ext API submits the request by invoking Ext API (This is similar to what is being proposed for CCVPN use case as well. This maintains that only Ext API interacts with outside world and other ONAP components do not) [Note - There can be multiple 3rd Party domains, After SO decomposes the CFS into multiple RFSs, Ext API will send the request to the corresponding 3rd party domain]
10 – ONAP Ext API invokes Ordering API, order translation to 3rd Party format happens outside Ext API, translated order gets submitted to 3rd Party domain
11 – Ext API updates AAI with the RFS instance details received from 3rd party response. AAI topology gets synced with the Service instance details to the level of the RFS instance.
12 - Ext API updates SO with the order item status for the RFS order item. Once SO has received responses for all the RFS order items in the order, it sends a response to Ext API which then responds to BSS with order update.

Impact Analysis So Far for 3rd Party Catalog Sync

SDC

- Expose POST functionality of SDC Onboarding API as an external API within ONAP
- Reuse sdc-dao to update the Cassandra database and store the new service in SDC catalog
- Reuse SDC distribution functionality to distribute the new service to registered ONAP components (no change)
- Existing UUID creation logic will be used
- Last mile access service from 3rd party will be used for detailed analysis and reference implementation
- TOSCA based onboarding in work is progress in SDC, it supports heat based onboarding only. The TOSCA based work is ongoing separately in Modeling project. This dependency on Modeling project need to be looked into.

Ext API / NBI

- Introduce POST for TMF API 633 – Service Catalog API
- Realization of POST operation in Ext API will depend on decisions taken during SDC implementation.

- Ext API changes to be planned for future release

Possible Approaches for 3rd Party Catalog Sync

Legend : Pros , Cons, Neutral

Entity Option 1: Resource

- Onboard the resource in ONAP SDC as a VSP, will require updates to VSP onboarding API

Entity Option 2: Service (Proposed)

- Onboard the service in ONAP SDC as a Service, will require updates to Service onboarding API

Payload Option 1: JSON (Proposed)

- Leveraging existing approach for Ext API / NBI
- Ext API will expose POST for TMF633 Service Catalog
- 3rd Party will send the payload in TMF633 format
- ExtAPI / NBI will send the JSON in SDC compatible format for its Consumption in v1/catalog/services

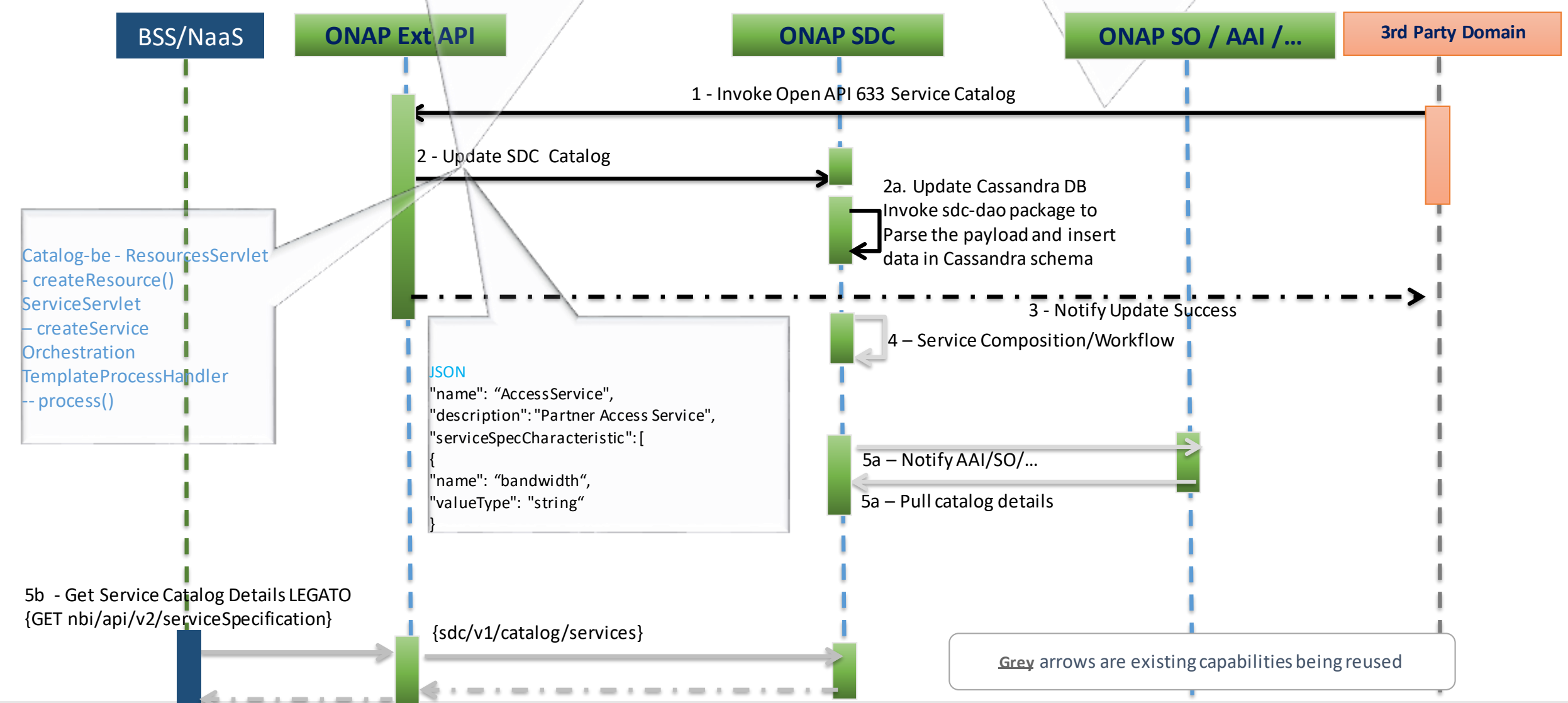
Payload Option 2: CSAR

- Potential reuse from TOSCA onboarding Project in SDC
- This might alter existing Ext API / NBI approach
- There would be additional implementation at Third Party end to generate higher level TOSCA

Flow Diagram for 3rd PARTY SDC Catalog Sync

Invoke onboarding API {POST sdc/v1/catalog/services} JSON contents:
RFSS for partner service to be created as a Service in ONAP

Publish Resource or RFSS to ONAP using JSON
(INTERLUDE) {POST nbi/api/v2/serviceSpecification}



Steps of Flow Diagram SDC Catalog sync

Steps 1 to 3 on previous slide - which are part of new functionality - are explained here.
Steps 4 onward depict existing functionality reused

1 – Invoke TMF 633 Service Catalog API
3rd Party Domain's Payload to be submitted as a JSON –
Expected format - Service Specification payload specified by TMF 633

2- Ext API to invoke SDC onboarding API to updated ONAP SDC catalog
Invoke ServiceServlet - createService() – JSON payload
Currently on-boarding API is invoked when Create Service button is clicked in SDC UI
Ext API needs to be added as a consumer of the API
Existing logic to be reused:
- UUID creation in validateServiceBeforeCreate
- Logic to add default TOSCA components

2a – Persist the service in SDC database

3-Ext API will Notify 3rd Party after SDC catalog update
- Register for Distribution: Ext API will register itself with SDC.
- Ext API will receive distribution notification from SDC after service catalog creation in SDC
- Ext API notifies 3rd Party Domain

Payload structure of input to ONAP from Third Party


```
{
  "id": "2944ce7c-a7ce-4816-b08c-d51b8bbb2830",
  "name": "partner Access",
  "description": "Partner Access",
  "version": "v1.0.0",
  "lifecycleStatus": "Active",
  "serviceSpecCharacteristic": [
    {
      "name": "serviceDetails",
      "description": "Service details",
      "valueType": "object",
      "@type": "ServiceSpecCharacteristic",
      "minCardinality": 1,
      "maxCardinality": 1,
      "access": [
        "Create",
        "Read",
        "Update"
      ],
     "serviceSpecCharacteristicAttributes":[…],
      "configurable": false,
      "isUnique": false,
      "extensible": false
    },
    {  "name": "order"…},
    {  "name": "access"…},
],
  "@type": "NetworkServiceSpecification",
  "isBundle": false,
  "lastUpdate": "2019-05-17T06:37:31.911Z"
}
```

Payload structure of input to SDC Service creation API-with sample

```
{
"contactId": "cs0008"
"categories": [{}]
"name": "ExtService",
"tags": ["ExtService"],
"componentType": "SERVICE",
"projectCode": "010203",
"properties": [{}]
"inputs": [{}]
"ecompGeneratedNaming": true,
"serviceApiArtifacts": {},
"instantiationType": "A-la-carte",
"environmentContext": "General_Revenue-Bear
}
```

```
  "name": "Partner",
  "normalizedName": "Partner",
  "uniqueId":
"serviceNewCategory.Partner",
  "icons": ["Partner"],
  "subcategories": null,
  "version": null,
  "ownerId": null,
  "empty": false,
  "type": null
```

```
{
   "uniqueId": "",
   "type": "integer",
   "required": false,
   "definition": false,
   "description": "size",
   "password": false,
   "name": "addressId",
   "hidden": false,
   "immutable": false,
   "parentUniqueId": "",
   "isDeclaredListInput": false,
   "schemaType": "",
   "schemaProperty": {
      "type": "",
      "required": false,
      "definition": true,
      "password": false,
      "hidden": false,
      "immutable": false,
      "isDeclaredListInput": false,
      "getInputProperty": false,
      "empty": false
   },
   "getInputProperty": false,
   "ownerId": "",
   "empty": false
```

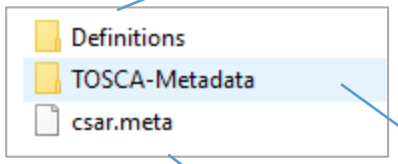
D:\Telstra\
teService-access1.j

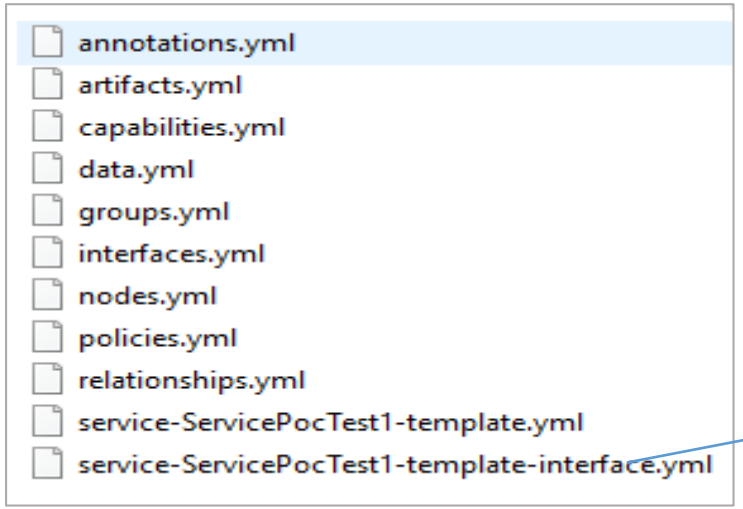
Placeholder for attributes needed for instantiation

Structure of SDC generated TOSCA CSAR

Below is the expanded view of the TOSCA CSAR generated by ONAP SDC.
Definitions – contains the interface yaml file which contains the metadata definition of the properties defined in the payload (detailed in previous slide)

No **Artifacts** folder as there are no deployment artifacts for the partner service

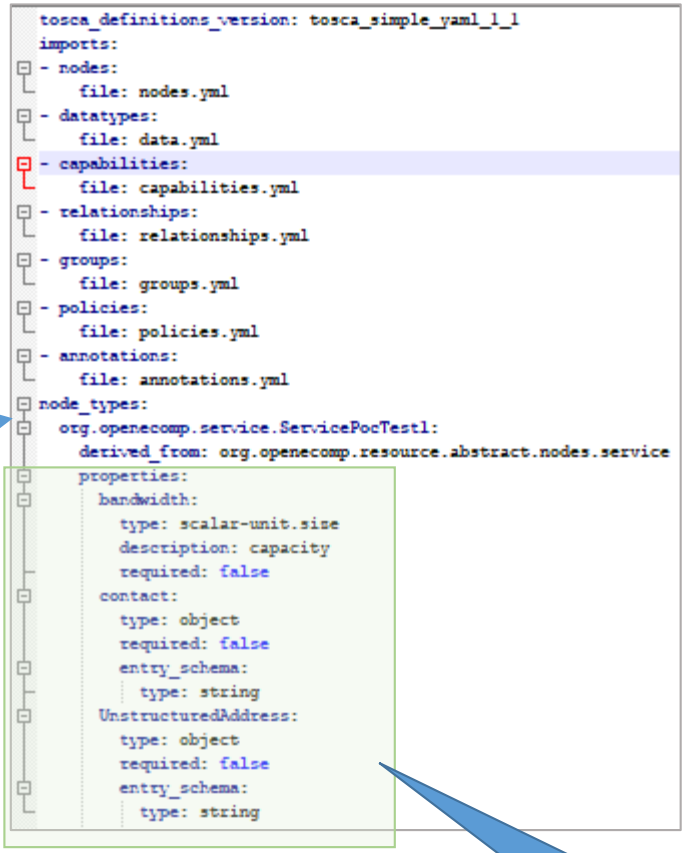




TOSCA-Meta-File-Version: 1.0
CSAR-Version: 1.1
Created-By: Carlos Santana
Entry-Definitions: Definitions/service-ServicePoc2-template.yml
Name: csar.meta
Content-Type: text/plain

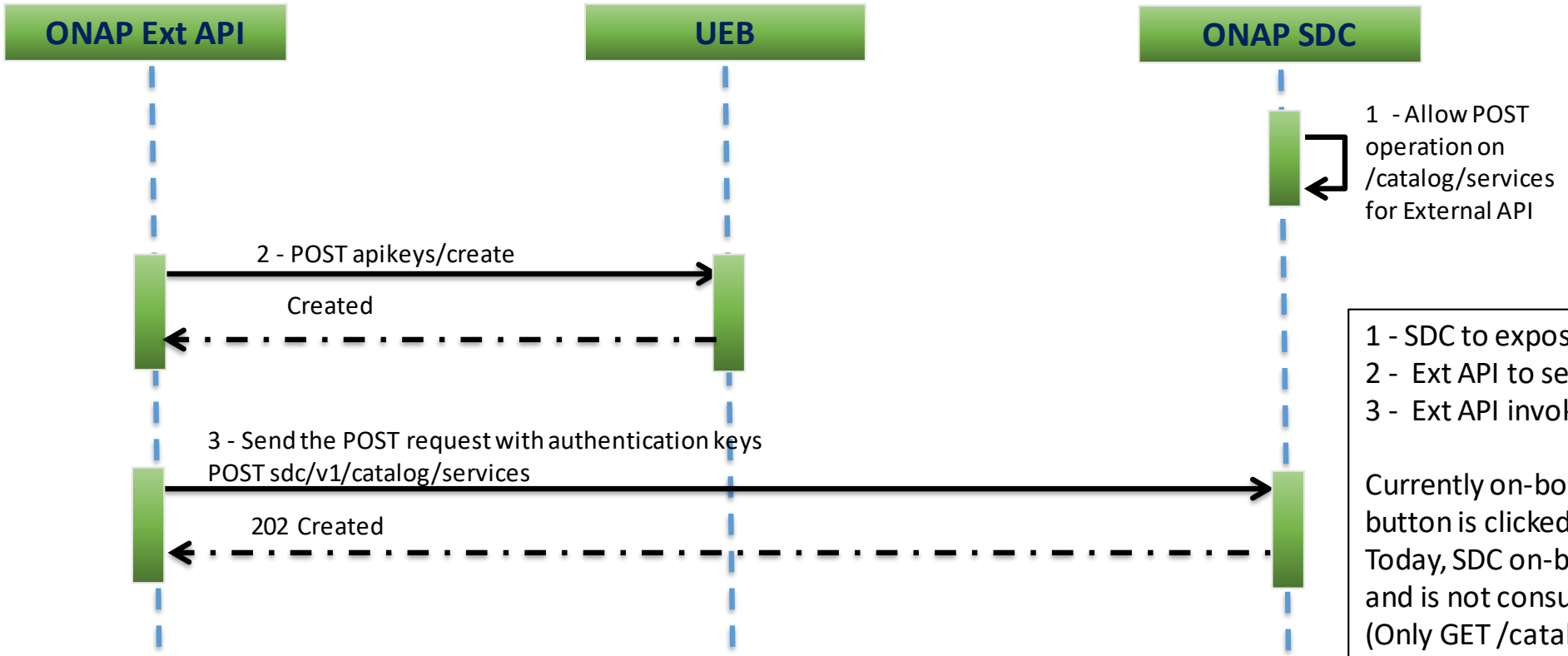
SDC-TOSCA-Meta-File-Version: 1.0
SDC-TOSCA-Definitions-Version: 9.0

service-ServicePocTest1-csar.zip



Service spec characteristics get added here

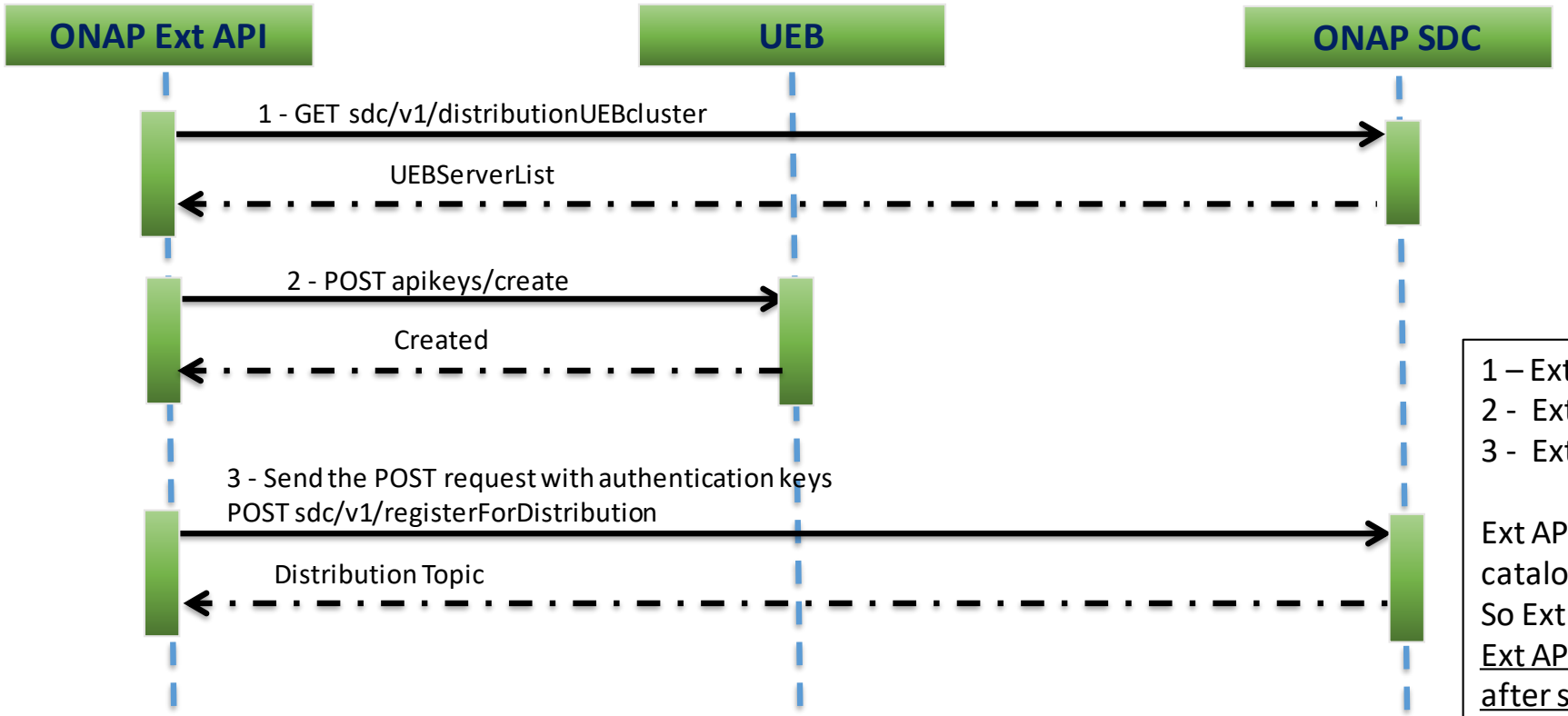
Flow Diagram for Ext API to consume SDC On-Boarding API



1 - SDC to expose on-boarding API to Ext API
2 - Ext API to send apikeys for the onboarding request
3 - Ext API invoke SDC onboarding API

Currently on-boarding API is invoked when Create Service button is clicked in SDC UI
Today, SDC on-boarding API is part of SDC Internal API and is not consumed by any ONAP component.
(Only GET /catalog/services is exposed)
Changes -
SDC on-boarding API needs to be exposed (within ONAP) similar to SDC Distribution APIs
Ext API needs to be added as a consumer of the API for POST operation.

Flow Diagram for Ext API to register for SDC Service Creation Notification



1 – Ext API to get server list
2 - Ext API creates API keys to authenticate
3 - Ext API registers with SDC

Ext API needs to Notify 3rd Party after successful Service catalog update on ONAP/SDC
So Ext API needs to register for notification:
Ext API will receive distribution notification from SDC after service catalog creation in SDC
Ext API notifies 3rd Party Domain

As part of the current phase of the use case, we will Concentrate on SDC changes and stub out Ext API function

Activities in Scope for SDC

- Impact Analysis
- Created epic https://jira.onap.org/browse/SDC-2378
  - Stories added so-far under the epic
  - https://jira.onap.org/browse/SDC-2382
  - https://jira.onap.org/browse/SDC-2383
  - https://jira.onap.org/browse/SDC-2385
- Sprint Planning
- Design
- Implementation
- Architecture Documentation
- Integration ??
- Testing Based on ROBOT Framework
- Defect Management

SDC-2382 - Introduce a new category for the 3rd party Service

S. No.	Change
1	We need to introduce new service category for 3rd Party Services
2	The dashboard has to be customized for the new service category, as certain functionalities need to be shown to the designer for the new kind of service
3	'Composition' tab need not be shown to the catalog user, as the 3rd Party service will not have VNF components to be added to it
4	3rd party service will only be created using an API. So manual creation of the service will be disabled

Projects	Files
catalog-be	sdc\catalog-be\src\main\resources\import\tosca\categories\categoryTypes.yml
catalog-ui	• menu.js • workspace-view-model.ts

SDC-2383 - Expose the API for service creation as an External API

Changes
SDC exposes GET on Service catalog to Ext API
Same auth can be used by Ext API to consume POST as well
SDC needs to introduce the method for service creation
The exposed method will reuse the logic available
Project – catalog-be Files – CrudExternalServlet Add method createServiceExternal – it will invoke existing ServiceBusinessLogic.createService

SDC-2385 - Introduce property mapping rules to define parent-child mapping for properties added in service definition

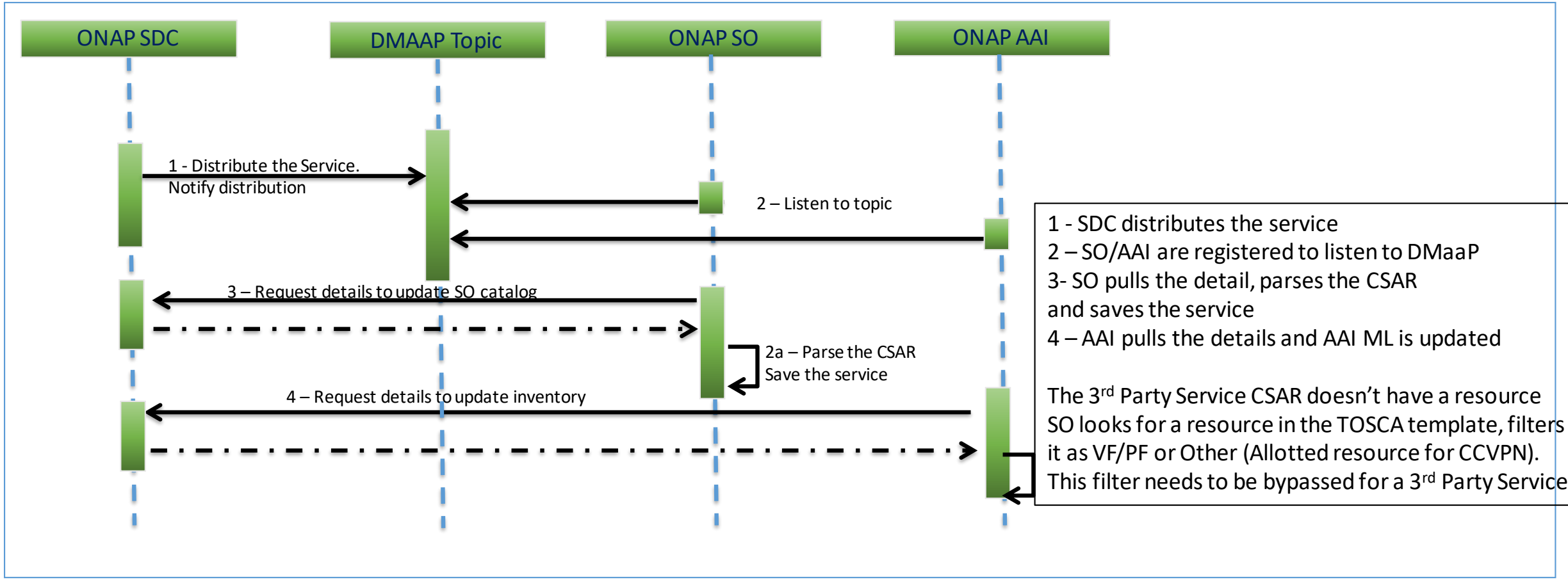
Changes
We need to define service characteristics for 3rd party services and many of the characteristics are of object type with child nodes for child attributes. e.g. bandwidth attribute can have child nodes upstream_speed, downstream_speed, unit and those child nodes can be of type enum will multiple values. see attached file In SDC payload, we define the service specification in a single hierarchy. We would need a logic to map the child nodes to the parent node

POC – Service on-boarding using a 3rd Party payload

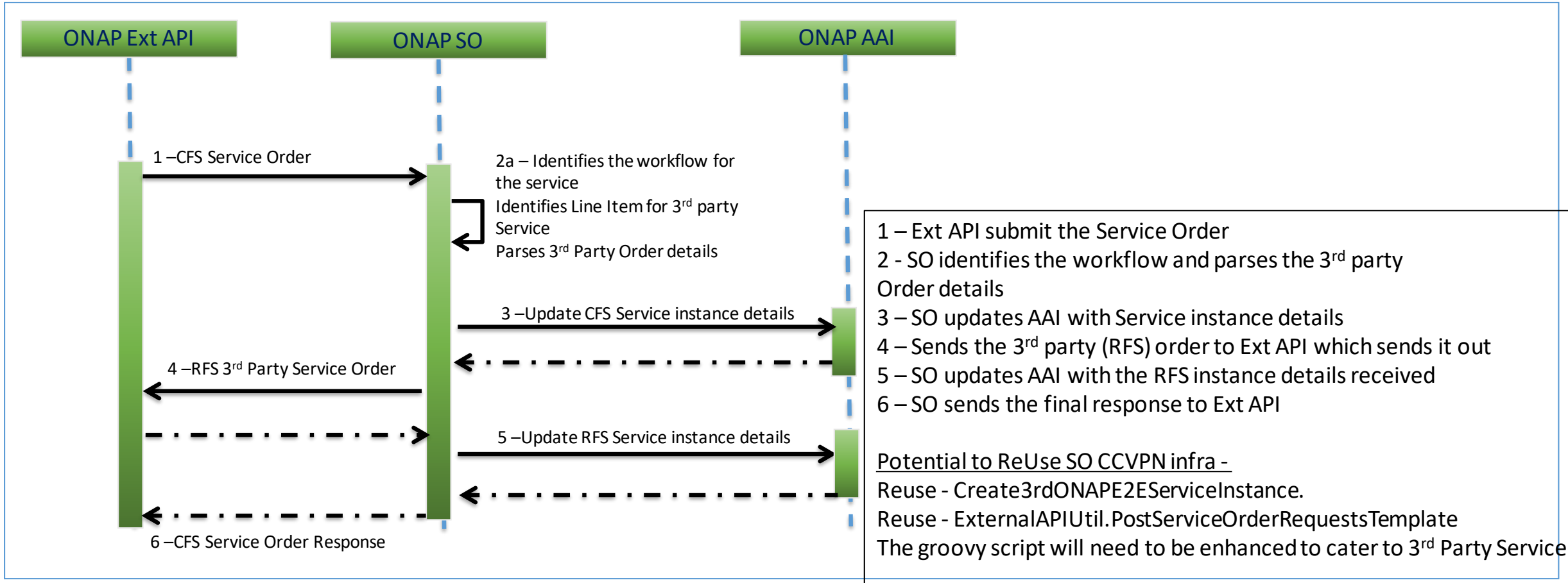
A POC has been done on local SDC dev instance to on-board the service using a sample payload for a 3rd party access service
Attached video takes us through the service creation journey, from creation to distribution-approved,
in SDC with the help of the API and SDC UI

It is also uploaded on the use case page at https://wiki.onap.org/display/DW/Third-party+Operational+Domain+Manager

Service Distribution – Design Time – SO Impact



Service Instantiation – Run Time – SO Impact



1 – Ext API submit the Service Order
2 - SO identifies the workflow and parses the 3rd party Order details
3 – SO updates AAI with Service instance details
4 – Sends the 3rd party (RFS) order to Ext API which sends it out
5 – SO updates AAI with the RFS instance details received
6 – SO sends the final response to Ext API

<u>Potential to ReUse SO CCVPN infra -</u>
Reuse - Create3rdONAPE2EServiceInstance.
Reuse - ExternalAPIUtil.PostServiceOrderRequestsTemplate
The groovy script will need to be enhanced to cater to 3rd Party Service

saveSPPartnerInAAI can be reused
[Analysis ongoing for SPPartner model in AAI]

Thank you