ONAP
OPEN NETWORK AUTOMATION PLATFORM

Orchestration of the Containerized Network Functions

Seshu Kumar M (Huawei)

Lukasz Rajewski (Orange)

Last Updated: 9.09.2020

CNFO - Summary for the requirement subcommittee

Executive Summary - Provide CNF orchestration support through integration of K8s adapter in ONAP SO

- Support for provisioning CNFs using an external K8s Manager
- Support the Helm based orchestration
- leverage the existing functionality of Multi cloud in SO
- Bring in the advantages of the K8s orchestrator
- Set stage for the Cloud Native scenarios

REQ-341

Owners: Lukasz Rajewski (Orange), Seshu Kumar M (Huawei), Srini Addepalli (Intel)

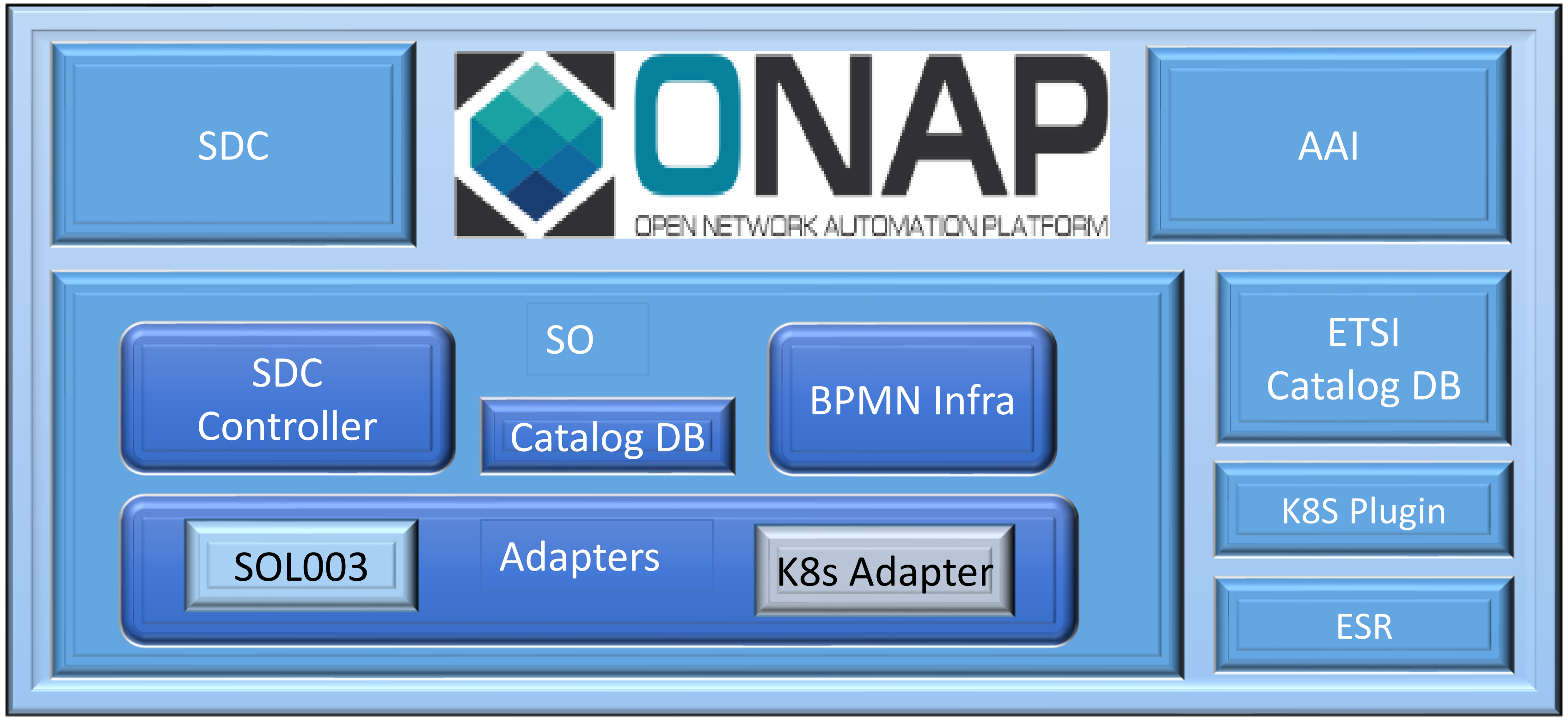
Business Impact - Enables operators and service providers to orchestrate CNFs based services along with the VNFs and PNFs

Business Markets - All operators and service providers that are intended to use the CNFs along with PNFs / VNFs

Funding/Financial Impacts - Reduction in the footprint of the ONAP for CNF support.

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

ONAP CNF Orchestration – Impacted Components - Guilin



Model to drive the flows:

- ✓ SDC to denote the flow of which VNFM should be used - similar to Orchestration Type
- ✓ Information model – optional
- ✓ Need to investigate the best place to have the meta data, we can perhaps use the existing fields

SDC:
On-board the helm and process it as an artifacts of the CSAR to be distributed

- ✓ On-board Helm Charts
- ✓ Resource model to include type
- ✓ Distribute them

SO:
Won't consume the Helm by itself but parse it and push it forward to other ONAP components

- ✓ Parse the CSAR, extract helm
- ✓ SOL003 adapter enhancements
- ✓ New K8s Adapter (Interface to K8s Plugin)

K8s Plugin:

- ✓ K8s Plugin without MSB
- ✓ Artifact Broker supports helm type artifact
- ✓ (Monitoring of status of k8s resources)

AAI:

- ✓ Persist the Service instance
- ✓ (Persist CNF as a resource)

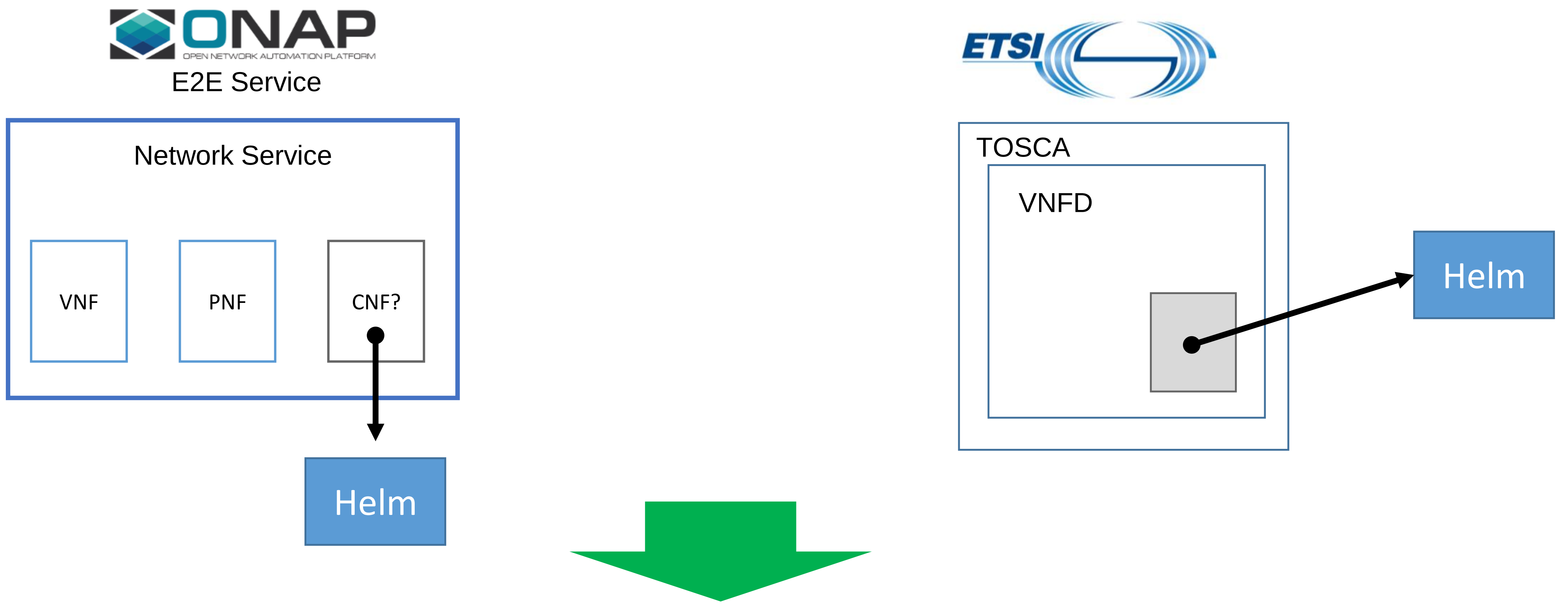
ESR:

- ✓ Assisted k8s cluster registration
- ✓ (K8s cluster auto-discovery)

ETSI Catalog DB:

- ✓ Persist the ETSi VNFM data (IFA-29 and IFA-40)
- ✓ Persist Images

ONAP - ETSI model Alignment

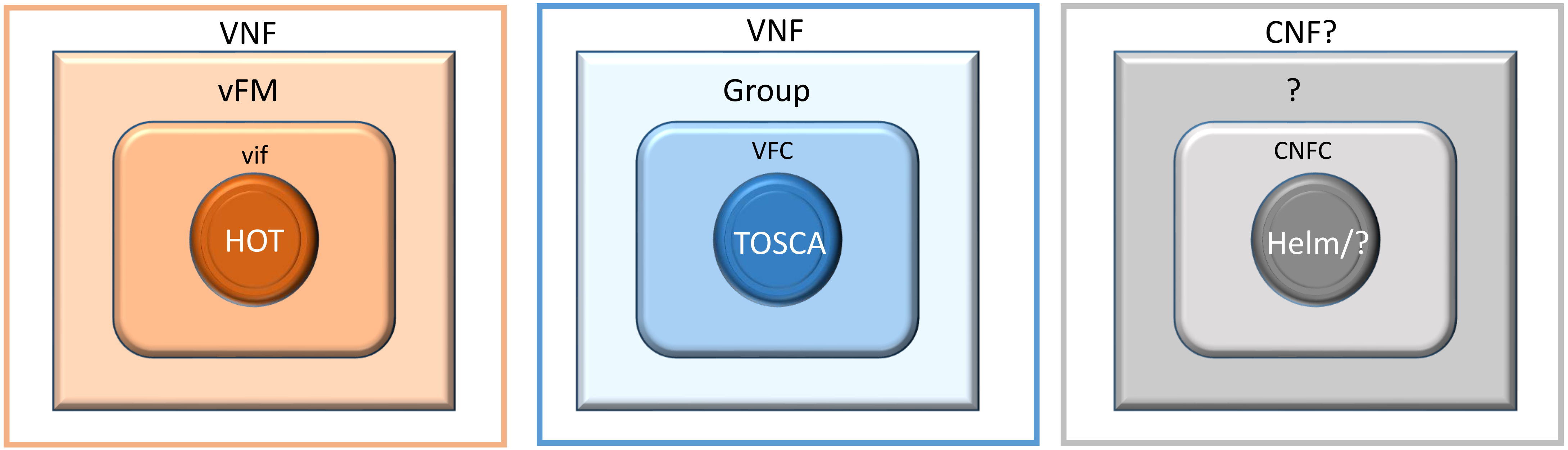


Integration of Native (K8s Adapter) with ETSi (SOL003 Adapter) paths in SO

→

Design/AAI CNF Model?

Approach 1 - Future



New model type into the SDC, AAI built over the helm charts as an input and would be distributed to the other ONAP components

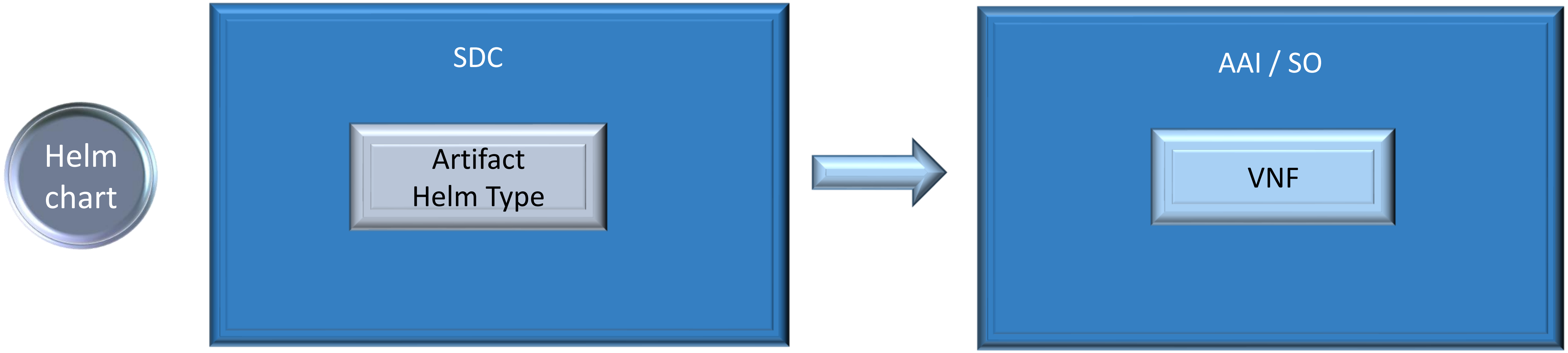
Pros

- ✓ Onboard a design template to the SDC and create a new resource from that
- ✓ Requires a new model to be introduced
- ✓ Will be inline to the existing models of the heat and TOSCA based VNFs
- ✓ Can also be extended to other formats for CNF modeling

Cons

- ✓ Initial analysis for understanding the standard model
- ✓ Requires more effort and may span across multiple ONAP releases
- ✓ The grouping model currently used in ONAP may pose a one-one mapping to the other standard formats

Approach 2 - Guilin



Helm Chart on-boarded as an artifact type to the SDC and distributed to ONAP, AAI would persist it as existing VNF
Helm chart would be stored as flat file and add it to the CSAR package to be distributed.

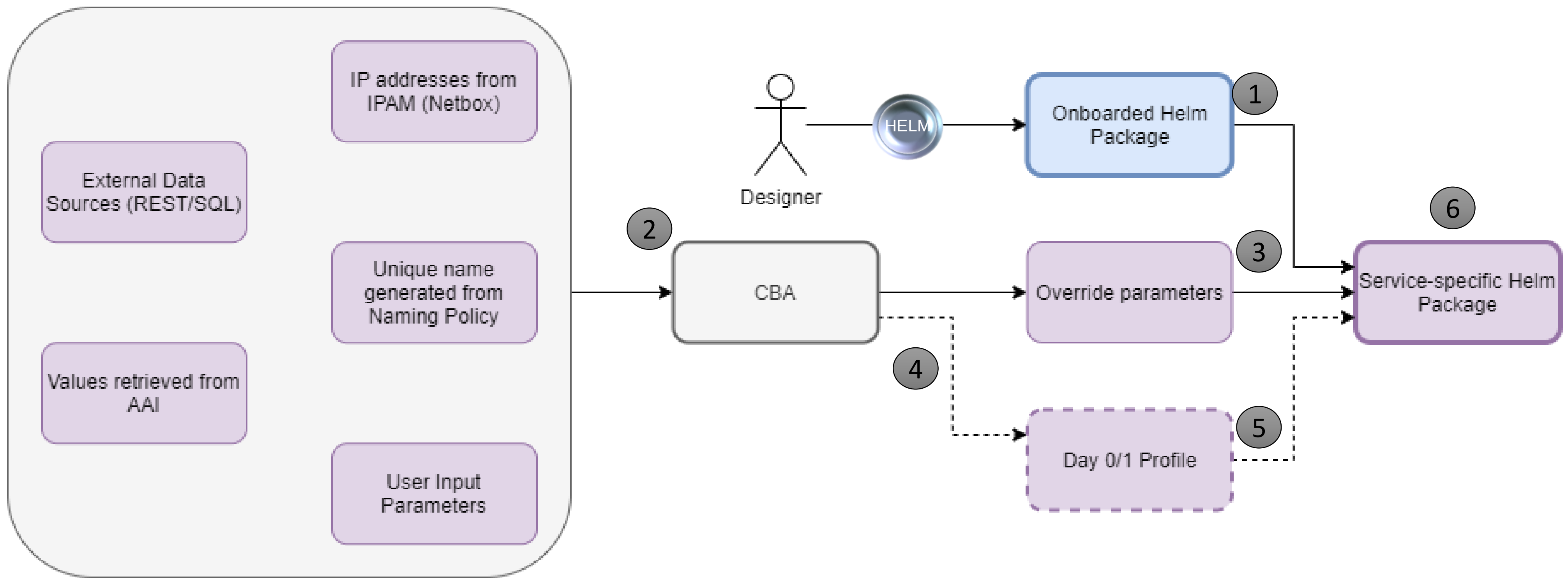
Pros

- ✓ Easy to develop comparing with Approach 1
- ✓ Better reuse of the existing functional code

Cons

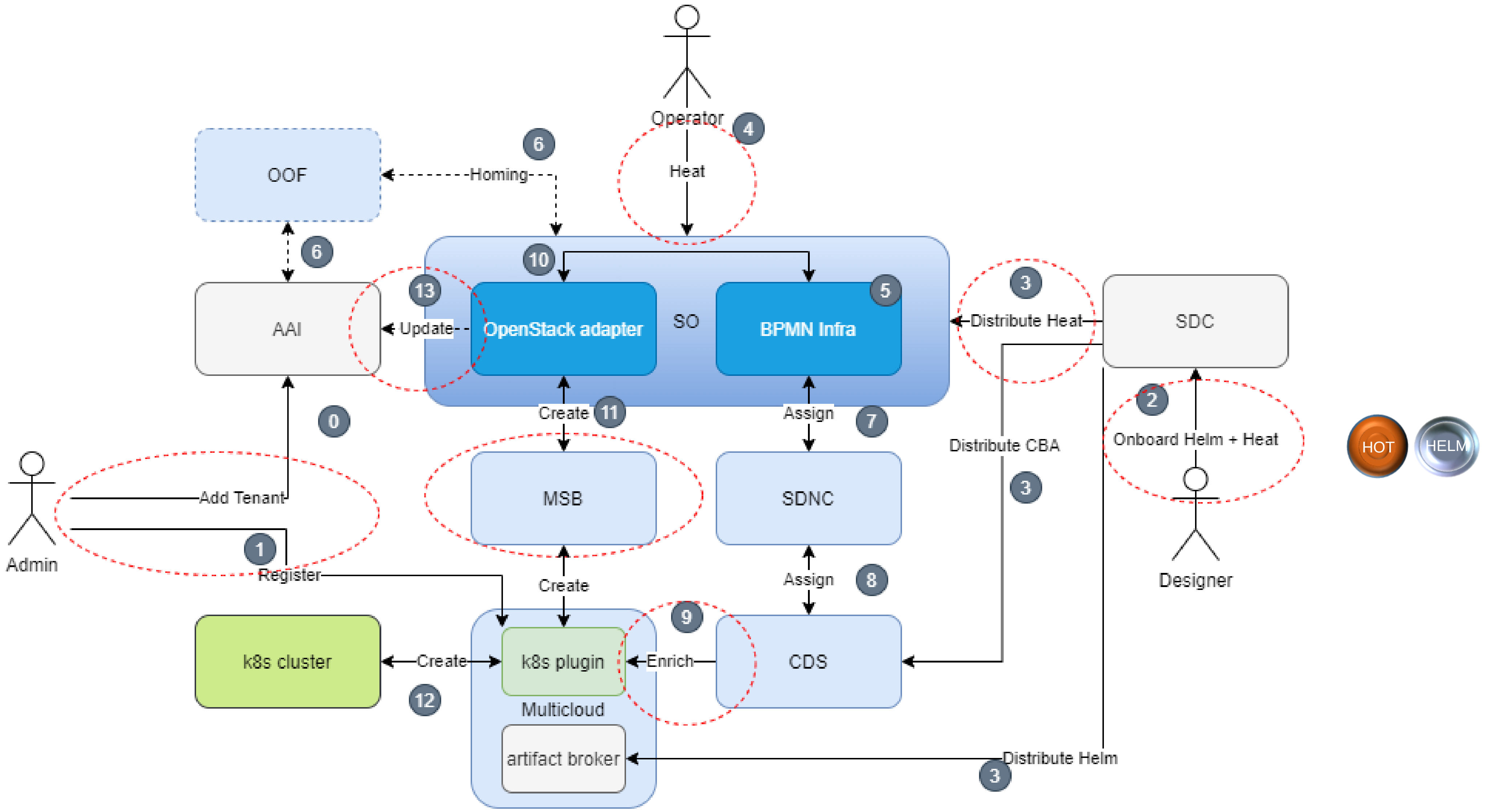
- ✓ Initial analysis for AAI persisting the CNF instance – Extend the VNF
- ✓ Very specific implementation and non extendable (non-helm)

Helm Package – From package in SDC to instantiation in K8s

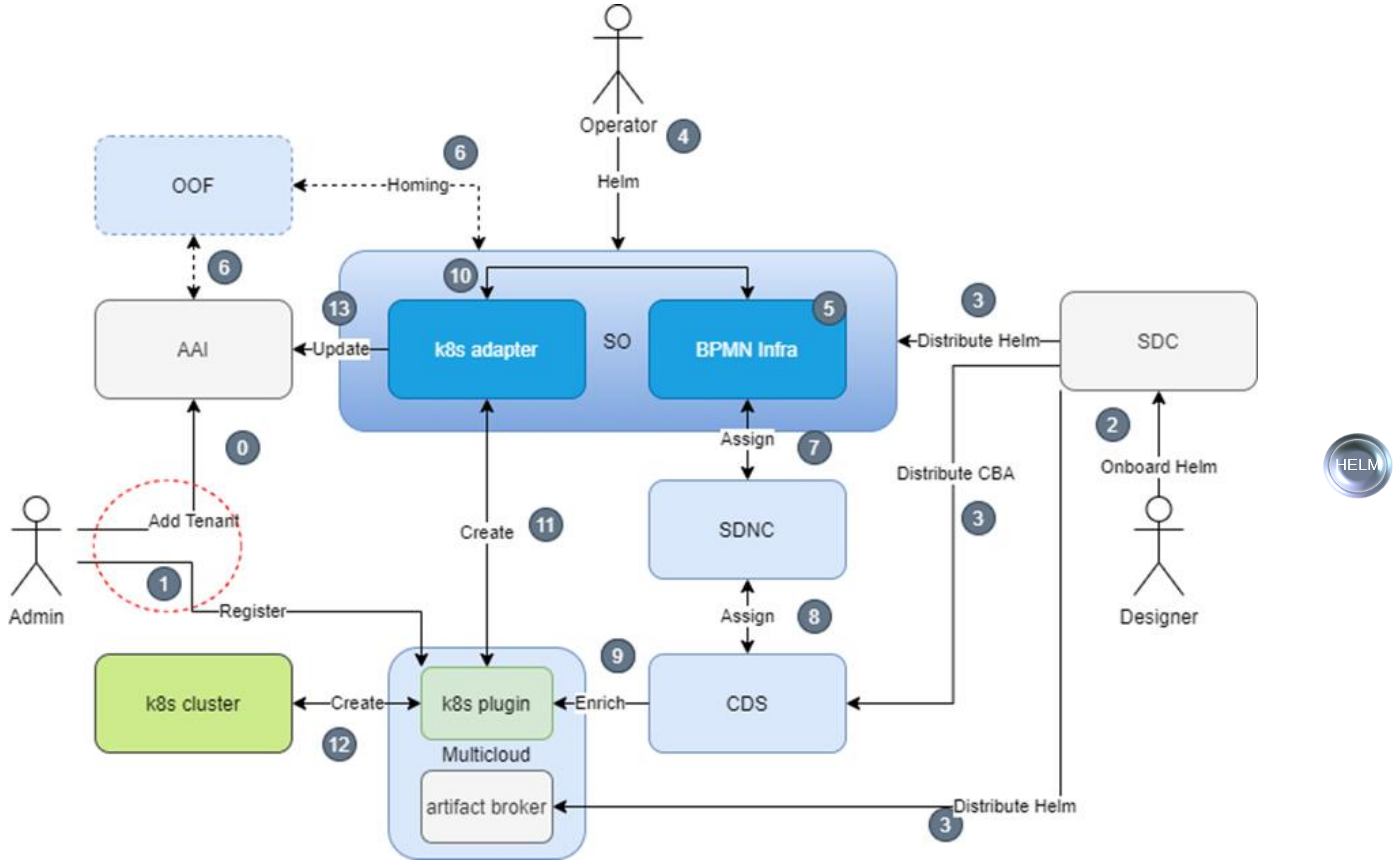


IP addresses from IPAM (Netbox)
External Data Sources (REST/SQL)
Unique name generated from Naming Policy
Values retrieved from AAI
User Input Parameters
Designer
HELM
Onboarded Helm Package
1
2
CBA
3
Override parameters
4
Day 0/1 Profile
5
6
Service-specific Helm Package

Frankfurt - OpenStack Adapter (Heat) Flow Day 0 + Day 1 Simplified



Guilin - K8s Adapter (Helm) Flow
Day 0 + Day 1 Simplified



Guilin – Design and Implementation Grounds

- ➢ Backward compatibility with CNF Macro Instantiation Workflow [Frankfurt] -> cvFW Example
- ➢ Instantiation of Helm Package with existing VNF model
- ➢ Status and synchronization of instantiated k8s resources
  - ✓ Helm Resource Artifact in SDC/SO
  - ✓ Selected k8s resources added to VNF model in AAI: i.e. Deployment/Stateful Set/Service etc.
  - ✓ Update of AAI Information by SO: vf-module + CNF specific sub-objects (stretch)
- ➢ K8s Plugin modifications
  - ✓ K8s Adapter in SO to interact directly with the K8s Plugin
  - ✓ Artifact Broker creates RB defnition from helm type artifact
  - ✓ Enhance it to support the functions like the monitoring resources and status update (stretch)
- ➢ Improvements in k8s cluster registration process (stretch)
- ➢ Validation through flows
  - ✓ cvFW Use Case
  - ✓ E2E (Core) Network Slicing Use Case (stretch)

REQ-341

CNF - Modeling and AAI SME work group (Future Enhancements)

- As part of extending the existing modeling to support the CNFO a separate group is formed with the SMEs from AAI, Modeling and existing function implementers.
- The key is to make sure we use the existing VNF model and adapt it to the CNFs to make them generic enough to handle most of the functional use cases
  - ✓ Generic VNF model to be re-used
  - ✓ Generic VF Module model to be extended to support CNFs?
- This group is discussing the details
  - ✓ How CNFs should be modeled
  - ✓ How to persist the available resources to be used by the CNFO
  - ✓ How to persist the CNF resource instances after they are validated
  - ✓ Who are the consumers
  - ✓ The intended format of the resource to be consumed by the Day 2 operations
  - ✓ Looking to other standardization bodies (CNCF, ETSi)
- Meeting is scheduled on every Friday 12:30 AM UTC

SDC

Helm Package – Structure v2 vs v3

Helm v2

```
wordpress/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license for the chart
  README.md           # OPTIONAL: A human-readable README file
  requirements.yaml   # OPTIONAL: A YAML file listing dependencies for the chart
  values.yaml         # The default configuration values for this chart
  charts/             # A directory containing any charts upon which this chart depends.
  templates/          # A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

Helm v3

```
wordpress/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license for the chart
  README.md           # OPTIONAL: A human-readable README file
  values.yaml         # The default configuration values for this chart
  values.schema.json  # OPTIONAL: A JSON Schema for imposing a structure on the values.yaml file
  charts/             # A directory containing any charts upon which this chart depends.
  crds/               # Custom Resource Definitions
  templates/          # A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

Helm Package – Chart descriptor

Helm v2

```
apiVersion: The chart API version, always "v1" (required)
name: The name of the chart (required)
version: A SemVer 2 version (required)
kubeVersion: A SemVer range of compatible Kubernetes versions (optional)
description: A single-sentence description of this project (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this project's home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
maintainers: # (optional)
  - name: The maintainer's name (required for each maintainer)
    email: The maintainer's email (optional for each maintainer)
    url: A URL for the maintainer (optional for each maintainer)
engine: gotpl # The name of the template engine (optional, defaults to gotpl)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
appVersion: The version of the app that this contains (optional). This needn't be SemVer.
deprecated: Whether this chart is deprecated (optional, boolean)
tillerVersion: The version of Tiller that this chart requires. This should be expressed as a SemVer range: "
```

Helm Package - Structure

```
|-Chart.yaml
|-templates
|   |-network_attachment_definition.yaml
|   |-onap-private-net.yaml
|   |-protected-private-net.yaml
|   |-unprotected-private-net.yaml
|-values.yaml
```

- Yaml syntax
- Complex Structure
  - ✓ Chart descriptor
  - ✓ Templates
  - ✓ Override values file
  - ✓ [Nested Helm charts]
- Override values
  - ✓ Full Yaml structure possible
  - ✓ Flat key - value map is only one option
  - ✓ Nested values.yaml

```
|-Chart.yaml
|-charts
|   |-packetgen
|   |   |-Chart.yaml
|   |   |-templates
|   |   |   |-deployment.yaml
|   |   |   |-service.yaml
|   |   |   |-_helpers.tpl
|   |   |-values.yaml
|   |-sink
|   |-Chart.yaml
|   |   |-templates
|   |   |   |-configmap.yaml
|   |   |   |-deployment.yaml
|   |   |   |-service.yaml
|   |   |   |-_helpers.tpl
|   |   |-values.yaml
|-templates
|   |-deployment.yaml
|   |-onap-private-net.yaml
|   |-protected-private-net.yaml
|   |-unprotected-private-net.yaml
|   |-_helpers.tpl
|-values.yaml
```

Helm Package – Values

```
title: "My WordPress Site" # Sent to the WordPress template

global:
  app: MyWordPress

mysql:
  global:
    app: MyWordPress
  max_connections: 100 # Sent to MySQL
  password: "secret"

apache:
  global:
    app: MyWordPress
  port: 8080 # Passed to Apache
```

- Complex Struture of parameters
- Keys like .mysql.global.app
- Arrays in key names

SDC - Onboarding Package with Helm

```
{
    "name": "virtualFirewall",
    "description": "",
    "data": [
        {
            "file": "base_template.yaml",
            "type": "HEAT",
            "isBase": "true",
            "data": [
                {
                    "file": "base_template.env",
                    "type": "HEAT_ENV"
                }
            ]
        },
        {
            "file": "base_template_cloudtech_k8s_charts.tgz",
            "type": "CLOUD_TECHNOLOGY_SPECIFIC_ARTIFACT"
        }

    ]
}
```

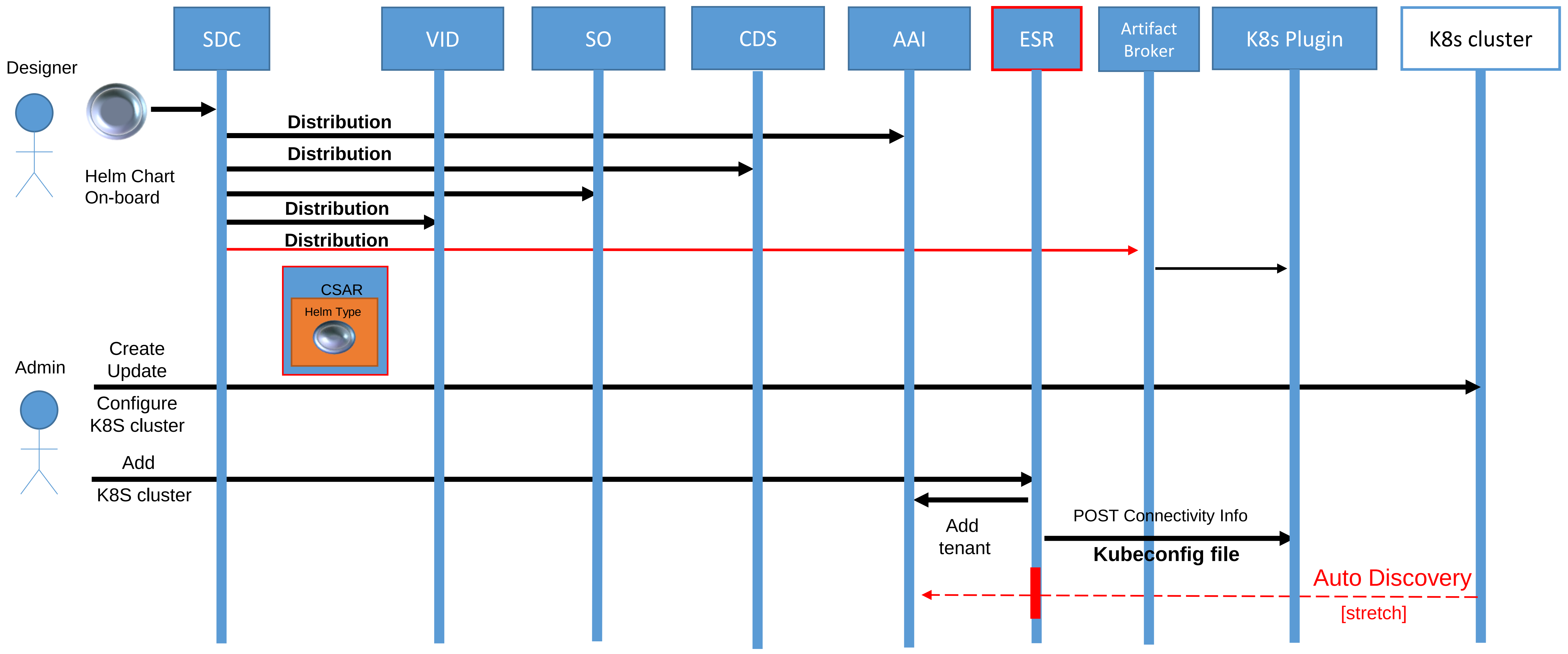
```
{
    "name": "virtualFirewall",
    "description": "",
    "data": [
        {
            "file": "base_template.tgz",
            "type": "HELM",
            "isBase": "true"
        }

    ]
}
```

- Validation of Helm package?
  - Name
  - Description
  - Version
- Basic Information read from Chart.yaml?
- Reading of values.yaml into VF properties

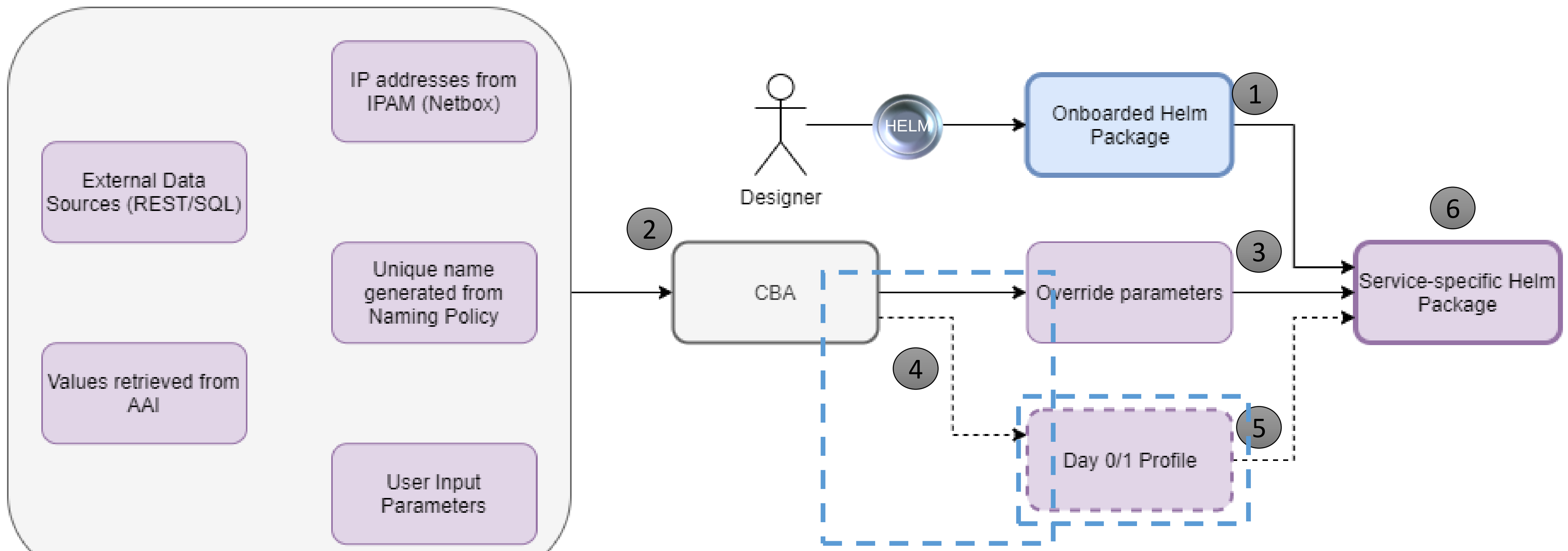
Stretch

Guilin - Design and Distribution of the Helm Chart - Day 0



SDNC & CDS

Helm Package – From package in SDC to instantiation in K8s


IP addresses from IPAM (Netbox)
External Data Sources (REST/SQL)
Unique name generated from Naming Policy
Values retrieved from AAI
User Input Parameters
Designer
HELM
Onboarded Helm Package
1
2
CBA
3
Override parameters
4
Day 0/1 Profile
5
6
Service-specific Helm Package

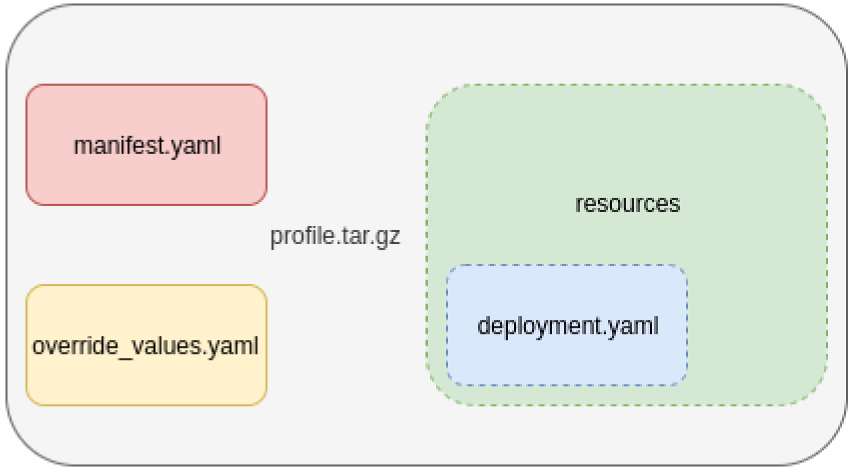
We want to improve creation of profile and its upload to k8s plugin

RB Profile – Helm Enrichment Package

Example of RB Profile's manifest

```
---
version: v1
type:
  values: override_values.yaml
  configresource:
    - filepath: resources/deployment.yaml
      chartpath: templates/deployment.yaml
```

Example of profile's structure



- K8s Plugin Requires profile to be archived as tar.gz file Complex Structure
- Profile contains Manifest + additional files
  - Files: override.yaml + optional extra resources
  - Extra resources: deployments.yaml, configmap.yaml etc.
  - Extra resources replace existing helm templates or add new ones
  - Values file gets merged with values file from helm package

Helm Package – Values

```
title: "My WordPress Site" # Sent to the WordPress template

global:
  app: MyWordPress

mysql:
  global:
    app: MyWordPress
  max_connections: 100 # Sent to MySQL
  password: "secret"

apache:
  global:
    app: MyWordPress
  port: 8080 # Passed to Apache
```

- Complex Structure of parameters
- Keys like .mysql.global.app
- Arrays in key names

- Too much for simple key-value input from SO to Plugin
- Good for Templating in CDS

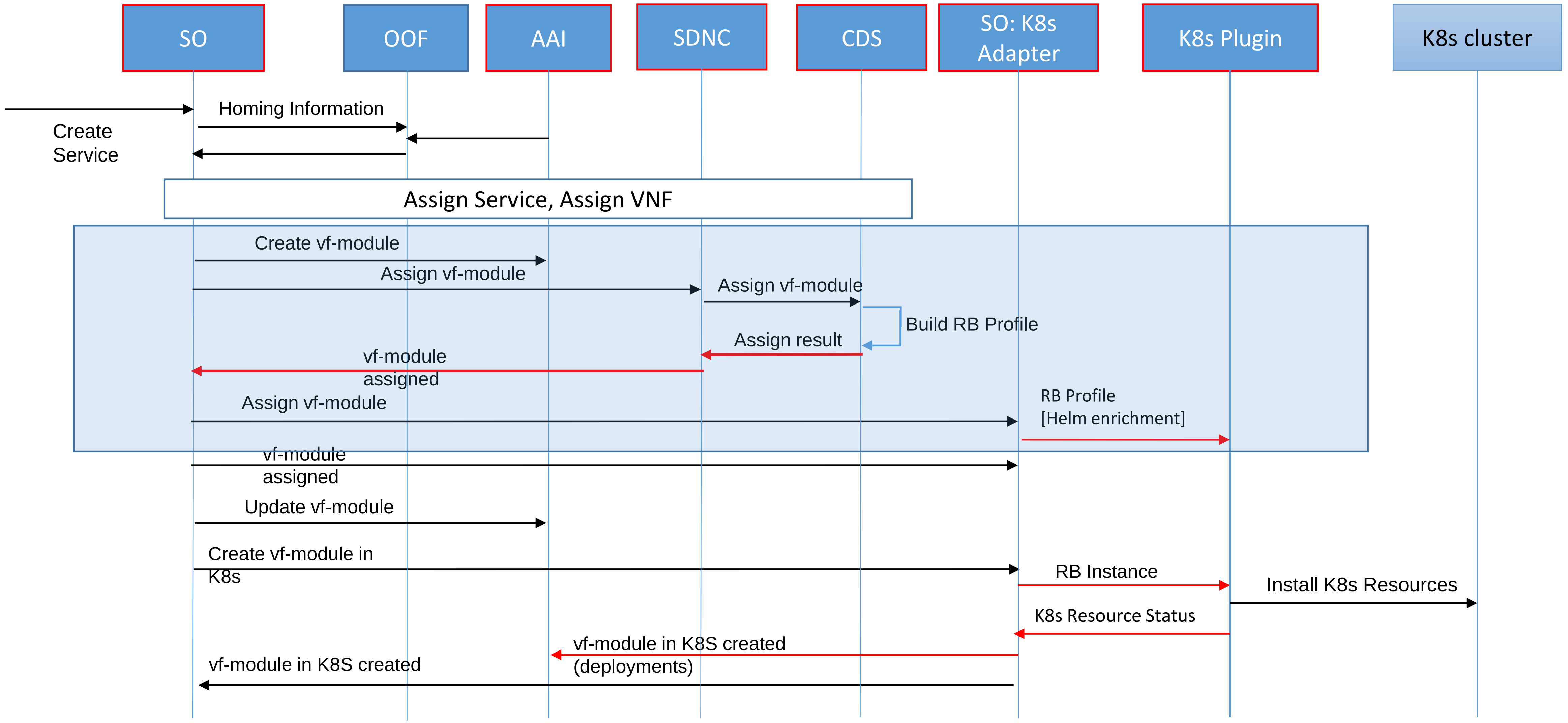
Helm Package - Structure

```
|-Chart.yaml
|-templates
|   |-network_attachment_definition.yaml
|   |-onap-private-net.yaml
|   |-protected-private-net.yaml
|   |-unprotected-private-net.yaml
|-values.yaml
```

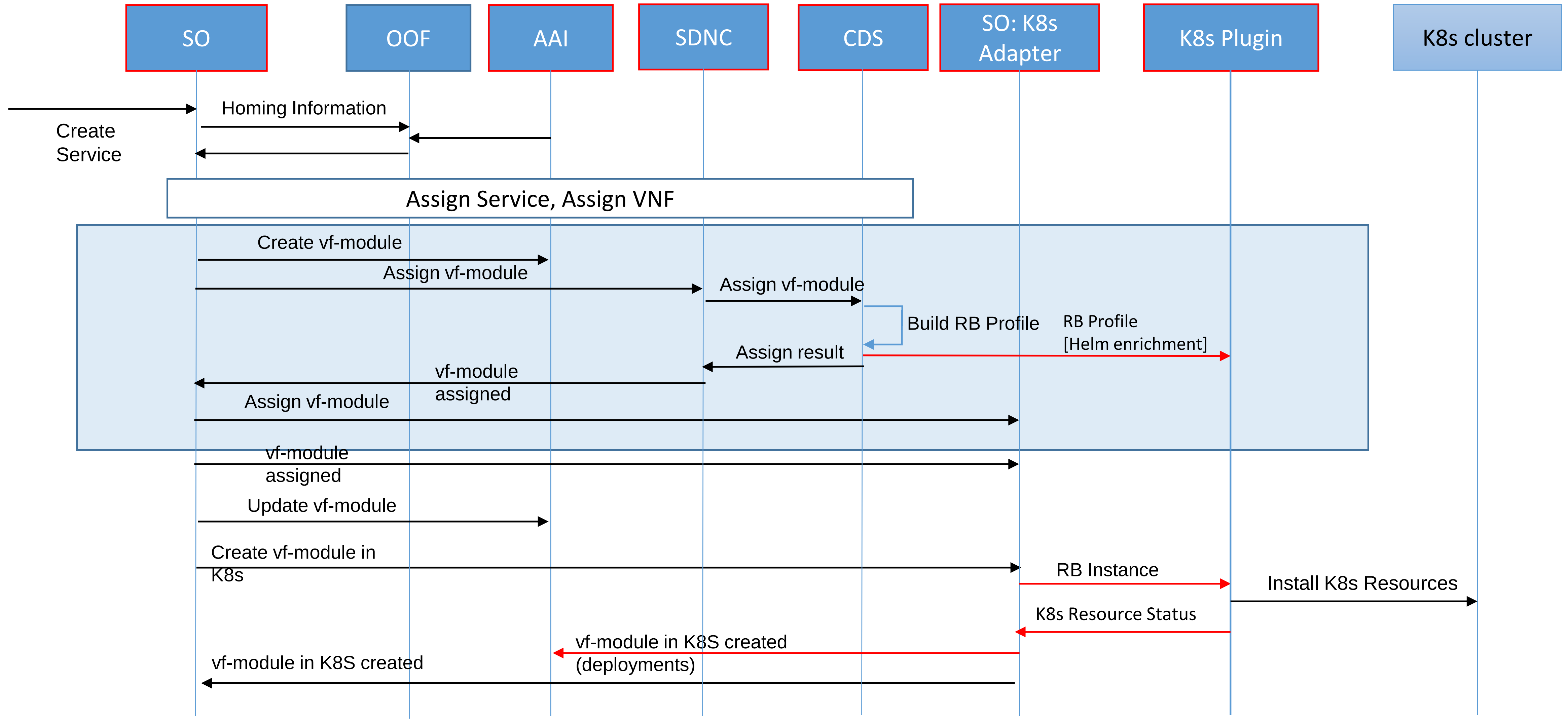
- Yaml syntax
- Complex Structure
  - ✓ Chart descriptor
  - ✓ Templates
  - ✓ Override values file
  - ✓ [Nested Helm charts]
- Override values
  - ✓ Full Yaml structure possible
  - ✓ Flat key - value map is only one option
  - ✓ Nested values.yaml

```
|-Chart.yaml
|-charts
|   |-packetgen
|   |   |-Chart.yaml
|   |   |-templates
|   |   |   |-deployment.yaml
|   |   |   |-service.yaml
|   |   |   |-_helpers.tpl
|   |   |-values.yaml
|   |-sink
|   |-Chart.yaml
|   |   |-templates
|   |   |   |-configmap.yaml
|   |   |   |-deployment.yaml
|   |   |   |-service.yaml
|   |   |   |-_helpers.tpl
|   |   |-values.yaml
|-templates
|   |-deployment.yaml
|   |-onap-private-net.yaml
|   |-protected-private-net.yaml
|   |-unprotected-private-net.yaml
|   |-_helpers.tpl
|-values.yaml
```

Guilin – Instantiation of the Helm Chart (Simplified- k8s Adapter Flow) - Day 1 – Option 1



Guilin – Instantiation of the Helm Chart (Simplified- k8s Adapter Flow) - Day 1 – Option 2



CDS Role in RB Profile creation

- For R6 we showcases the use of imperative workflows in CDS for RB profiling
- CDS builds and uploads RB Profile content in VF-module resource-assigment phase
- Two steps adds after standard resource-assignment phase
  1. Profile Modification – templating of profile – standard
     - ✓ Creation of override value.yaml from template
     - ✓ Creation of override value.yaml from vtl template for subcharts
     - ✓ On demand modification/creation of k8s helm templates from vtl templates
  2. Profile Upload to add profile creation step after standard resource assignment
     - ✓ We need to include this logic each time in dedicated script in CBA
- Option 1
  - ✓ CDS builds profile and returns extra binary content to SDNC
  - ✓ SDNC uploads profile to *sdnctl* DB, into dedicated table
  - ✓ SO K8s Adapter reads profile from *sdnctl* DB and uploads it to K8s Plugin
- Option 2 chosen for implementation
  - ✓ CDS builds profile
  - ✓ CDS uploads profile to K8s Plugin over built-in mechanism -> no need to implement it in each CBA file
  - ✓ Opportunity to build joint K8s interface in CDS for profiling + configuration templating (Day 0/1) + configuring (Day 2)

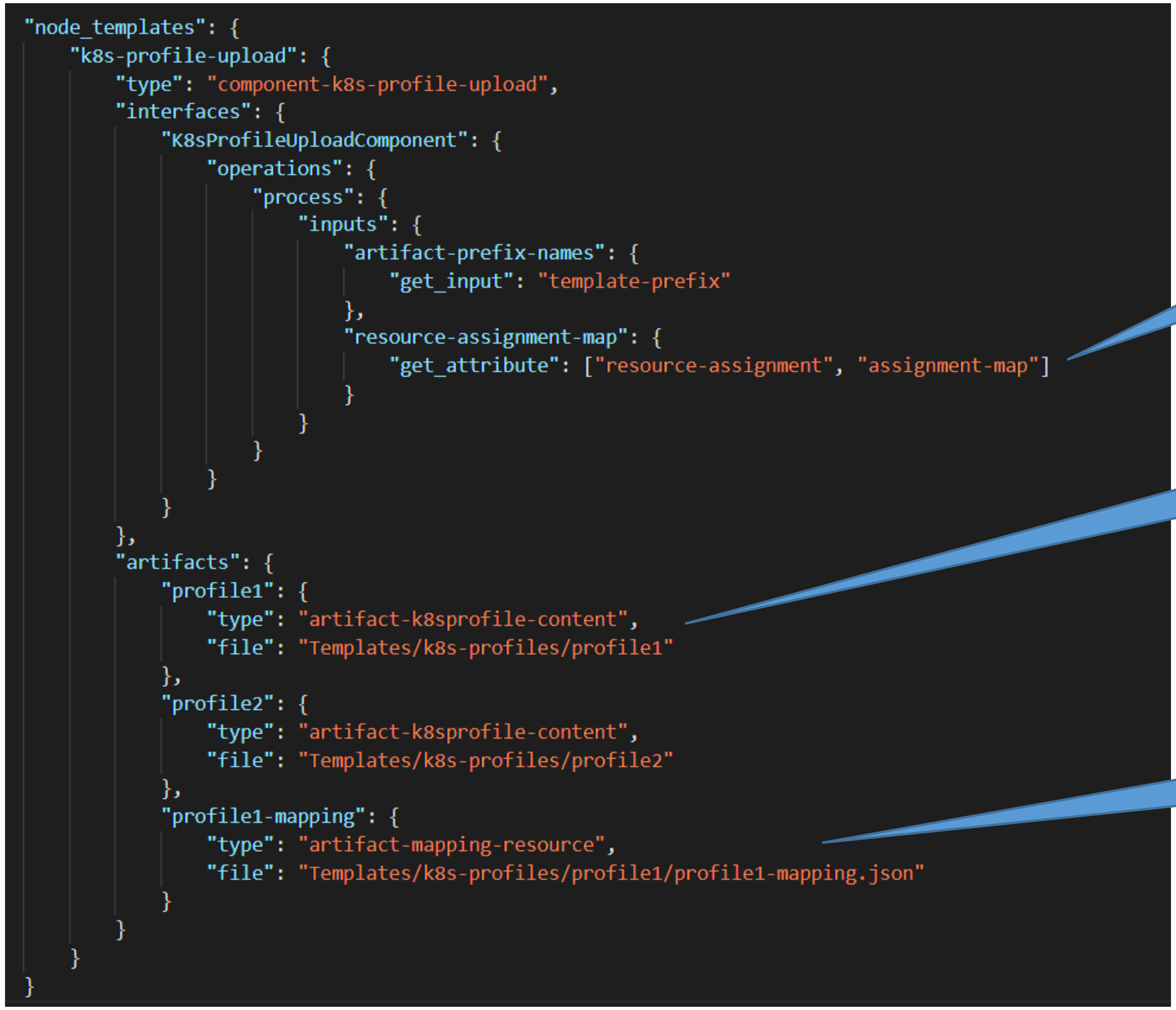
RB Profile Upload – Option 2 – Assumptions (1)

- We will utilize imperative workflows - **resource-assignment** workflow will have 2 steps: resources-assignment + **k8s-profile-upload**
- CBA may have dedicated subfolder like **k8s-profiles** in the Templates – it will be a source for k8s-profiles
- **k8s-profile-upload** step will have 7 inputs
  - **k8s-rb-profile-name –** the name of the profile under which it will be created in k8s plugin. Other parameters are required only when profile must be uploaded
  - k8s-rb-definition-name – the name under which RB definition was created
  - k8s-rb-definition-version – the version of created RB definition name
  - k8s-rb-profile-namespace – the k8s namespace name associated with profile being created
  - k8s-rb-profile-source – the source of profile content analyzed in the following order
    - If this is a file name with **tar.gz** or **.tgz** extensions i.e. **profile2**.tar.gz and is present in Template/k8s-profiles/**profile2.tar.gz**
    - If this is a folder i.e. **profile1** and is present in Template/k8s-profiles/**profile1** it means that profile must be generated
  - resource-assignment-map – result of the associated resource assignment step
  - artifact-prefix-names – the list of artifact prefixes like for **resource-assigment** step
- If any input required for profile upload is empty, profile upload should be skipped
- k8s-rb* inputs must be specified directly in the definition or can come from the **assignment-map** result of **resource-assignment** step which needs to have k8s-rb* inputs
- Profile upload will be skipped when **k8s-profile-upload** step is missing in resources-assignment workflow or **k8s-rb-profile-name** is default or **k8s-rb-profile-name** profile already exists or **k8s-rb-profile-name** is not defined

RB Profile Upload – Option 2 – Assumptions (2)

- Each subfolder under Templates/k8s-profiles will have all the files required for profile generation
- If instead of subfolder we have complete .tar.gz or .tgz file we just take it as a complete profile and we use it for upload into k8s plugin
- The complete profile manifest file must be present in the profile folder. Only files referenced in the manifest will be included in the final profile
- **K8s-profile-upload** needs to have **artifacts** sections specified with list of all profile sources and mappings for them
- If ***.vtl** files are present in profile-source they will be processed by CDS and will be templated
  - Dedicated component-resource-resolution process will be run once for selected profile-source
  - There is only one **{k8s-rb-profile-source}-mapping.json** file present for all the *.vtl templates of specific profile
  - There will be templating run for each *.vtl file present in the profile source.
  - **Only Velocity templating will be supported** as jinja templating conflicts with helm templates
  - The filepath from manifest must be the same like filepath in the profile source folder. After templating .vtl extension will be removed and such file must be referenced in the manifest

RB Profile Upload – Option 2 – Node Template in CBA



```
"node_templates": {
    "k8s-profile-upload": {
        "type": "component-k8s-profile-upload",
        "interfaces": {
            "K8sProfileUploadComponent": {
                "operations": {
                    "process": {
                        "inputs": {
                            "artifact-prefix-names": {
                                "get_input": "template-prefix"
                            },
                            "resource-assignment-map": {
                                "get_attribute": ["resource-assignment", "assignment-map"]
                            }
                        }
                    }
                }
            }
        },
        "artifacts": {
            "profile1": {
                "type": "artifact-k8sprofile-content",
                "file": "Templates/k8s-profiles/profile1"
            },
            "profile2": {
                "type": "artifact-k8sprofile-content",
                "file": "Templates/k8s-profiles/profile2"
            },
            "profile1-mapping": {
                "type": "artifact-mapping-resource",
                "file": "Templates/k8s-profiles/profile1/profile1-mapping.json"
            }
        }
    }
}
```

Here the k8s-rb* inputs will be taken from resource-assignment-map

Each profile source must be listed as artifact-k8sprofile-content artifact

If profile source is a folder it needs to have associated mapping file included

RB Profile Upload – Option 2 – Profile Manifest

```
-Templates
 └──k8s-profiles
     │    profile2.tar.gz
     │
     └────profile1
           │   manifest.yaml
           │   override_values.yaml.vtl
           │   profile1-mapping.json
           │
           └──resources
                   configmap.yaml.vtl
                   deployment.yaml
```

k8s-rb-profile-source: complete profile with tar.gz or tgz extension or folder with content

Templated file. Final file name without is .vtl extension

Definitions of values used for Velocity (vtl) templating

CBA profiling content

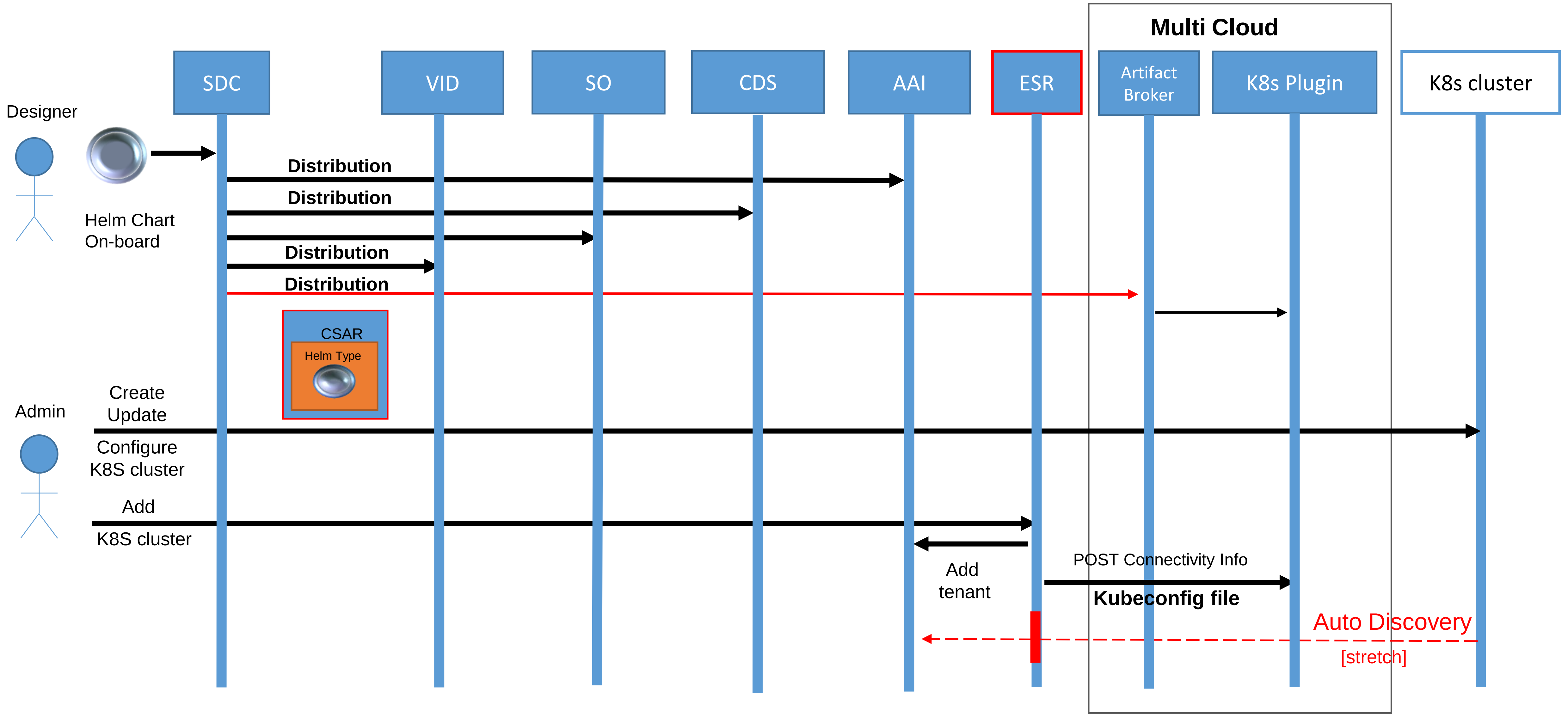
```
---
version: v1
type:
  values: override_values.yaml
  configresource:
   - filepath: resources/deployment.yaml
     chartpath: templates/deployment.yaml
  - filepath: resources/configmap.yaml
     chartpath: templates/deployment.yaml
```

Profile will contain only files mentioned in the manifest

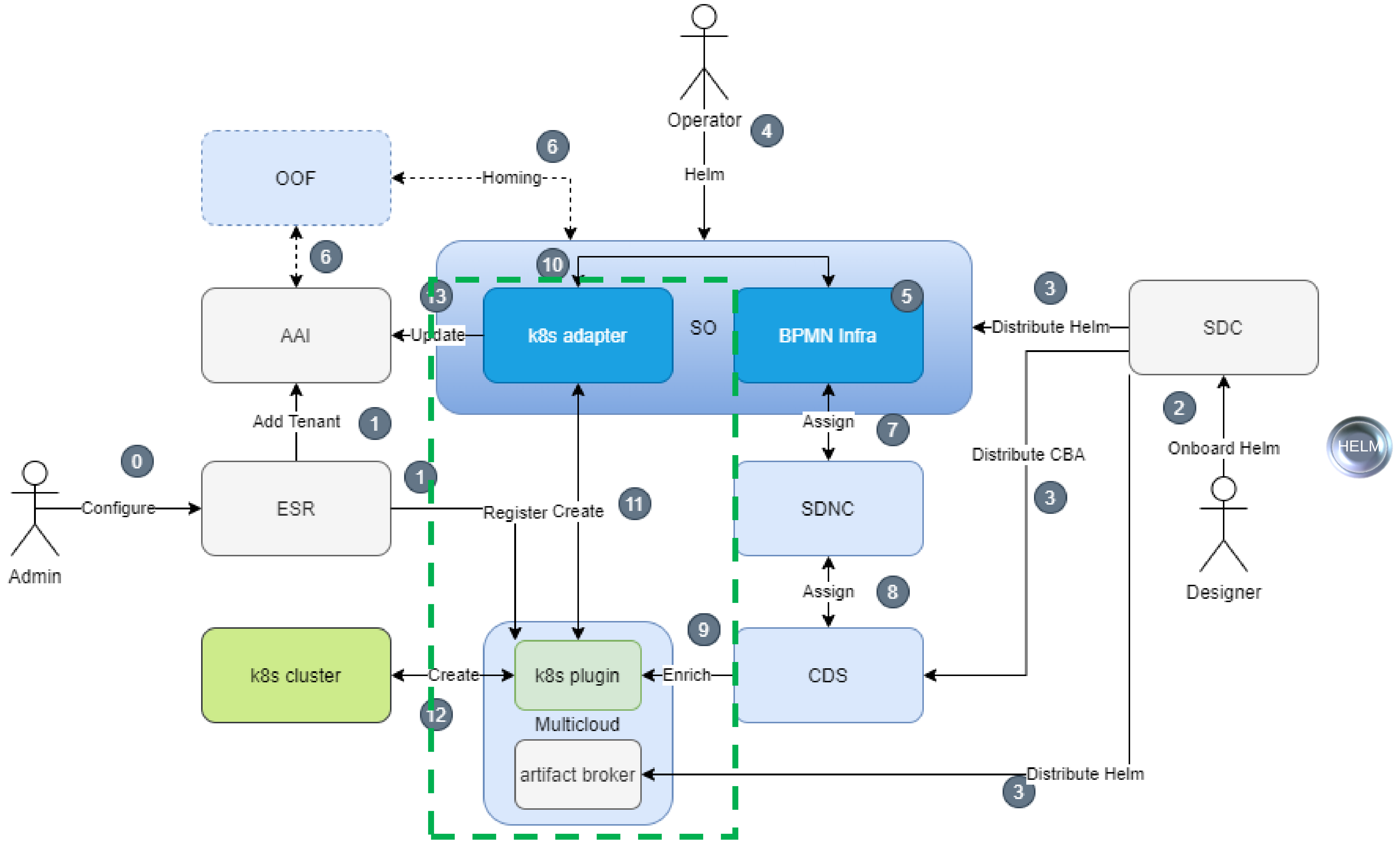
Manifest for profile1 source

MUTLICLOUD

Guilin - Design and Distribution of the Helm Chart - Day 0



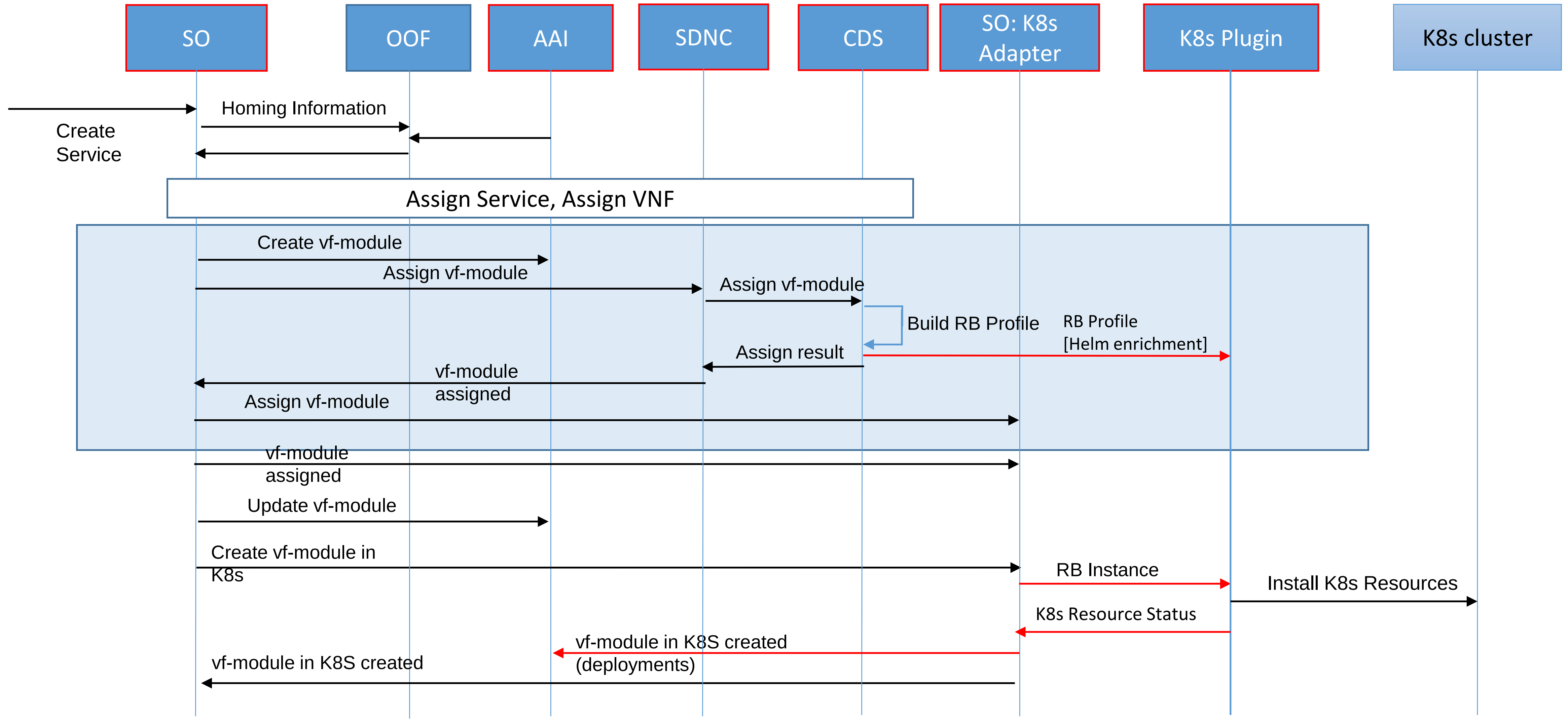
Guilin - K8s Adapter (Helm) Flow
Day 0 + Day 1 Simplified



Guilin – Multicloud Impacts

- K8s ArtifactBroker needs following minor changes
  - ArtifactBroker is responsible for reception of HELM_ARTIFACT (the new artifact type added in SDC)
  - RB Definition to be created from new HELM_ARTIFACT type artifacts distributed
  - Existing flow for reception of helm chart from CLOUD_TECHNLOGOY_ARTIFACT would be preserved
- CDS will interact with k8s plugin MS for profile enrichment
  - CDS will upload RB Profile before instantiation of CNF
- SO interacts with MC K8s Plugin MS for instantiation
  - There will be a new ‘k8s adapter’ MS in Service Orchestrator
  - It will interact directly (without MSB) with existing K8s plugin MS from Multicloud
- K8s resource status monitoring (stretch)
  - Potentially new API endpoint for k8s plugin v1
  - Related on implementation of Monitoring API in V2
  - Functionality from v2 to be ported into v1 – required analysis of feasibility

Guilin – Instantiation of the Helm Chart (Simplified- k8s Adapter Flow) - Day 1 – Option 2



v1 Instance API Change – CREATE instance req

POST http://multicloud-k8s:9015/v1/instance

BODY:
```
{
    "rb-name": "${rb_name}",
    "rb-version": "${rb_version}",
    "profile-name": "${profile_name}",
    "cloud-region": "${cloud_region_id}",
    "release-name": "${release_name}",
    "labels": {
        "custom-label-1": "abcdef"
    },
    "override-values": {
        "image.tag": "latest",
        "dcae_collector_ip": "1.2.3.4"
    }
}
```

"rb-name"	VF Module model invariant ID
"rb-version"	VF Module model version ID
"profile-name"	k8s-rb-profile-name (SDNC Directives)
"cloud-region"	SO decides
"release-name"	VF Module UUID
"custom-label-1"	SO decides, anything what later on might be used by k8s adapter in SO
"override-values"	SDNC Directives

v1 Instance API Change – CREATE instance resp

```
{
    "id": "frosty_kowalevski",
    "request": {
        "rb-name": "vfw",
        "rb-version": "plugin_test",
        "profile-name": "test_profile",
        "cloud-region": "kud",
        "labels": {
            "testCaseName": "plugin_fw.sh"
        },
        "override-values": {
            "global.onapPrivateNetworkName":
 "onap-private-net-test"
        }
    },
    "namespace": "plugin-tests-namespace",
    "resources": [
        {
            "GVK": {
                "Group": "",
                "Version": "v1",
                "Kind": "ConfigMap"
            },
            "Name": "sink-configmap"
        },
        {
            "GVK": {
                "Group": "",
                "Version": "v1",
                "Kind": "Service"
            },
            "Name": "packetgen-service"
        },
        {
            "GVK": {
                "Group": "apps",
                "Version": "v1",
                "Kind": "Deployment"
            },
            "Name": "test-release-packetgen"
        }
    ]
}
```

v1 Instance API Change – GET instance req

GET http://multicloud-k8s:9015/v1/instance/{instance-id}

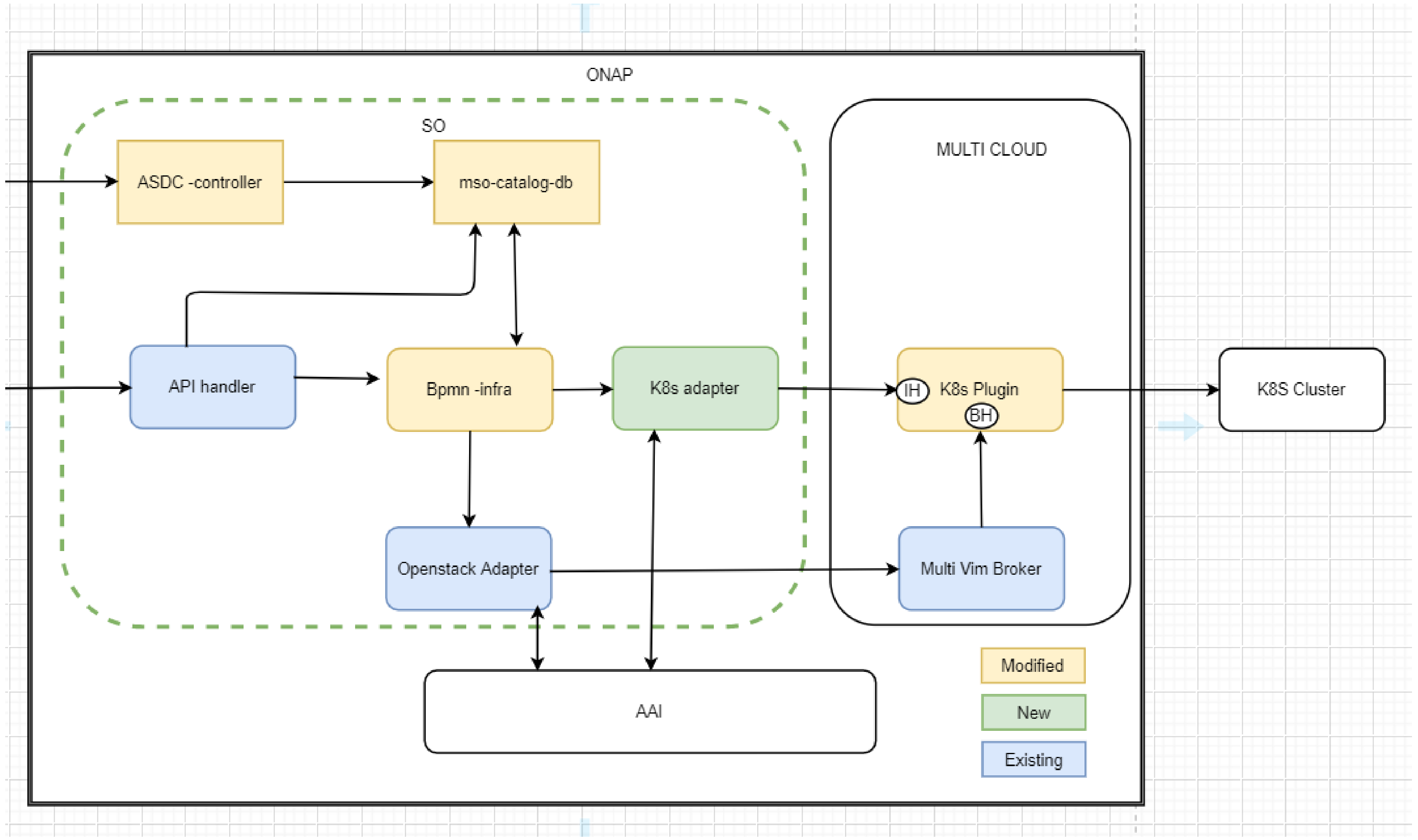
```
{
    "id": "frosty_kowalevski",
    "request": {
        "rb-name": "vfw",
        "rb-version": "plugin_test",
        "profile-name": "test_profile",
        "cloud-region": "kud",
        "labels": {
            "testCaseName": "plugin_fw.sh"
        },
        "override-values": {
            "global.onapPrivateNetworkName":
 "onap-private-net-test"
        }
    },
    "namespace": "plugin-tests-namespace",
    "resources": [
        {
            "GVK": {
                "Group": "",
                "Version": "v1",
                "Kind": "ConfigMap"
            },
            "Name": "sink-configmap"
        },
        {
            "GVK": {
                "Group": "",
                "Version": "v1",
                "Kind": "Service"
            },
            "Name": "packetgen-service"
        },
        {
            "GVK": {
                "Group": "apps",
                "Version": "v1",
                "Kind": "Deployment"
            },
            "Name": "test-release-packetgen"
        }
    ]
}
```

v1 Instance API Change – DELETE instance req

DELETE http://multicloud-k8s:9015/v1/instance/{instance-id}

S0

Guilin - SO Flow



Guilin – SO flow

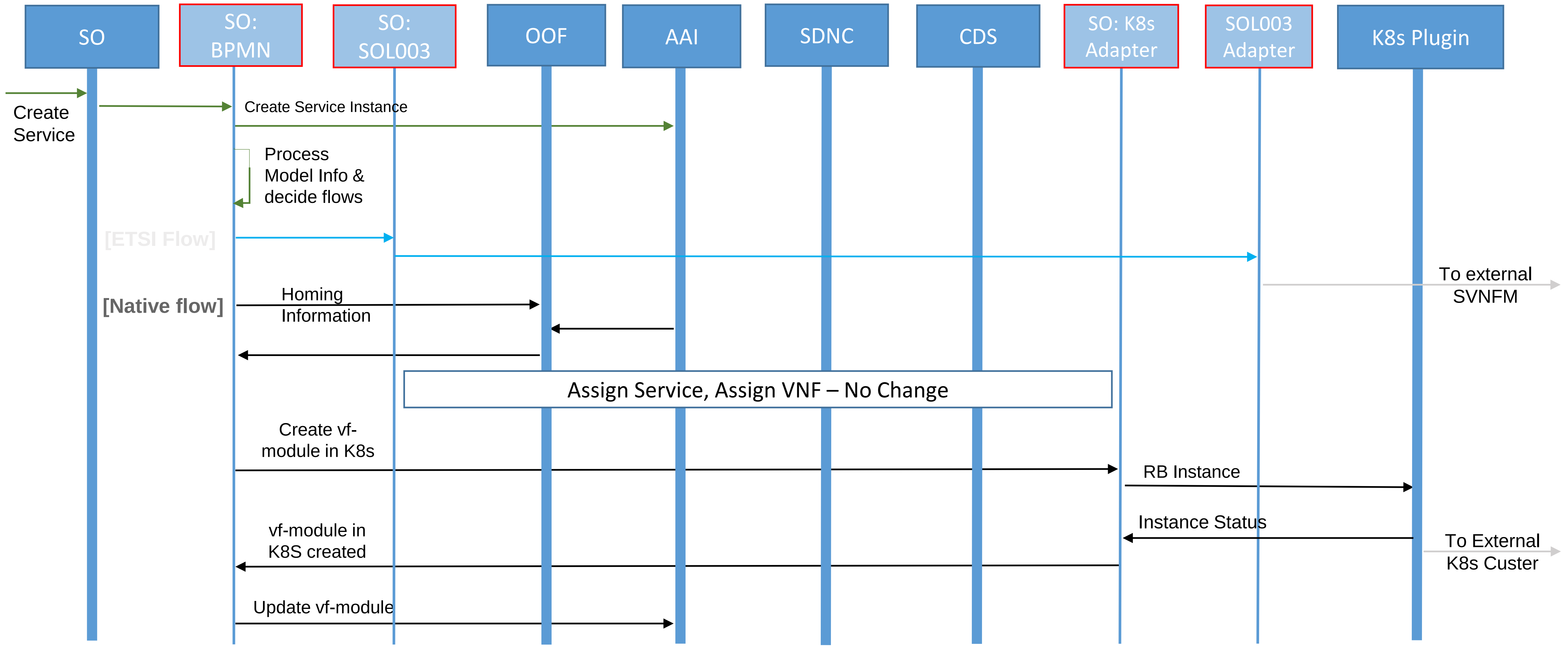
- 0 – asdc recieves callback notification to process csar file.
- 01- asdc saves the helm info in mso catalog db .
- 1- API handler recieves call from the operator.
- 2- API handler fetches required info from mso-catalog-db.
- 3-Bpmn infra is called by API handler.
- 4-Bpmn infra fetches required info from mso-catalog-db.
- 5- Bpmn infra calls the new K8s adaptor.
- 6-k8s adaptor calls the K8s plugin of multicloud
- 7-k8s adaptor and AAI updates each other .

ONAP

OPEN NETWORK AUTOMATION PLATFORM

Thank You!

Guilin – Instantiation of the Helm Chart – k8s Paths in SO - Day 1


SO
SO: BPMN
SO: SOL003
OOF
AAI
SDNC
CDS
SO: K8s Adapter
SOL003 Adapter
K8s Plugin
Create Service
Create Service Instance
Process Model Info & decide flows
[ETSI Flow]
To external SVNFM
[Native flow]
Homing Information
Assign Service, Assign VNF – No Change
Create vf-module in K8s
RB Instance
Instance Status
vf-module in K8S created
To External K8s Custer
Update vf-module