Release I: rApp Manager

Introduction

The rApp Manager is a lifecycle management service for rApps. It gets the rApp as an ASD formatted package and lifecycle manages it based on its instance configuration. It uses ONAP ACM for lifecycle management operations and it integrates with other components for managing the rApp.

The ASD package contains the details required to create and integrate the required services/components. Each ASD package contains only one rApp and one rApp can have any number of rApp instances.

Source code repository: https://gerrit.o-ran-sc.org/r/gitweb?p=nonrtric%2Fplt%2Frappmanager.git;a=summary

Architecture
rApp Data Model
Integrations

The rApp Manager is integrated with the following components to support lifecycle managing the rApp.

ACM

Automation Composition Management (ACM) is a framework that supports Life Cycle Management of Automation Compositions. It supports deployment, monitoring, update and removal of Automation Compositions en-bloc, allowing users to manage their features, services, and capabilities as single logical units. More details about ACM can be found here.

ACM-R has the ability to support an unlimited number of participants and all the participants can be configured through the configuration in the rApp package.

List of participants used by rApp manager sample rApp.

- A1PMS Participant - It interacts with A1PMS of NONRTRIC. It is capable of lifecycle managing A1PMS service.
- Kubernetes Participant - It interacts with Helm/Kubernetes. It is capable of lifecycle managing Helm charts. It expects the helm charts to be available in the mentioned repository as it doesn't handle the helm chart creation in the chart repository.
- DME Participant - It interacts with DME(ICS) of NONRTRIC. It is capable of lifecycle managing DME entities.

ACM composition and instance details can be provided as part of the rApp package and the package structure can be found here.

DME

The DME(Information Coordination Service (ICS)) is a generic service that maintains data subscriptions. Its main purpose is to decouple data consumers and data producers in a multi vendor environment. A data consumer does not need to know anything about the producers of the data. More details about DME can be found here.

It is integrated with rApp manager to enable the rApp to produce/consume specific type of data (Information Type in DME terms).

Information type, and Data producer/consumer information can be provided as part of rApp package and the package structure can be found here.

SME
The CAPIF stands for Common API framework and it was developed by 3GPP to enable a unified Northbound API framework across 3GPP network functions, and to ensure that there is a single and harmonized approach for API development. More details about SME can be found here.

It is integrated with rApp manager to enable the rApp to expose/access/discover endpoints.

Service exposure/access related configurations can be provided as part of rApp package and the package structure can be found here.

State Transitions

rApp States

The rApp lifecycle contains 4 states. The state and transitions are as follows,

1. COMMISSIONED
   - rApp get created in this state and once the DEPRIMING is completed

2. PRIMING
   - This is a transition state. rApp will be in this state once the PRIMING requested for rApp

3. PRIMED
   - rApp will be in this state once the PRIMING is completed. In this state rApp instances can be created

4. DEPRIMING
   - This is a transition state. rApp will be in this state once the DEPRIMING requested for rApp

rApp Instance States

The rApp Instance lifecycle contains 4 states. The state and transitions are as follows,
1. UNDEPLOYED
   rApp instance gets created in this state and once the rApp Instance undeploy is completed

2. DEPLOYING
   This is the transition state. rApp instance will be in this state once DEPLOY is requested

3. DEPLOYED
   rApp instance will be in this state once the rapp instance deployment is completed.

4. UNDEPLOYING
   This is a transition state. rApp instance will be in this state once UNDEPLOY requested for rApp instance

Flows

rApp flow

<table>
<thead>
<tr>
<th>Create rApp</th>
<th></th>
</tr>
</thead>
</table>

| UNDEPLOYED
|---|
1. API user creates rApp by sending rApp package

2. rApp Manager validates the rApp

3. rApp Manager stores the rApp in the file system if the rApp is valid

4. API user provided with the status of rApp creation.

5. API user request to Prime the rApp

6. rApp Manager fetches the ACM composition from rApp package and creates the ACM composition in ACM-R

7. rApp Manager gets the ACM composition creation status from ACM-R

8. rApp Manager request ACM-R to prime the ACM composition

9. rApp Manager gets the ACM composition priming status from ACM-R

10. rApp Manager checks with DME for the unknown information type from rApp package

11. rApp Manager get the information type availability from DME

12. API user provided with the status of rApp priming

Delete rApp
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>API user request to Deprime rApp</td>
</tr>
<tr>
<td>2.</td>
<td>rApp Manager request ACM-R to deprime the ACM composition</td>
</tr>
<tr>
<td>3.</td>
<td>rApp Manager get the status of ACM composition depriming.</td>
</tr>
<tr>
<td>4.</td>
<td>rApp Manager requests ACM-R to delete the ACM composition</td>
</tr>
<tr>
<td>5.</td>
<td>rApp Manager gets the status of ACM composition deletion</td>
</tr>
<tr>
<td>6.</td>
<td>API user provided with the deprime rApp status.</td>
</tr>
<tr>
<td>7.</td>
<td>API user request to delete the rApp</td>
</tr>
<tr>
<td>8.</td>
<td>rApp Manager validates that the rApp is in COMMISSIONED state and there is no rApp Instances are available.</td>
</tr>
<tr>
<td>9.</td>
<td>API User provided with delete rApp status</td>
</tr>
</tbody>
</table>

rApp Instance flow

<table>
<thead>
<tr>
<th>Create rApp Instance</th>
<th></th>
</tr>
</thead>
</table>
1. API user request to create rApp Instance with the resource details from rApp package.

2. API user get the rApp instance creation response.

Deploy rApp Instance

1. API user request to deploy rApp instance.

2. rApp Manager fetches the rApp package from file system and fetches the resource details mentioned in the rApp instance.

3. rApp Manager request ACM-R with the necessary details from rApp package to Instantiate ACM instance.
4. rApp Manager gets instantiate ACM instance response
5. rApp Manager request ACM-R to deploy ACM instance
6. ACM-R deploys the A1PMS instance if it is configured in the ACM instance
7. ACM-R gets the A1PMS deployment status
8. ACM-R deploys the Kserve instance if it is configured in the ACM instance
9. ACM-R gets the Kserve deployment status
10. ACM-R deploys the Kubernetes instance if it is configured in the ACM instance
11. ACM-R gets the Kubernetes deployment status
12. ACM-R deploys the DME instance if it is configured in the ACM instance.

13. ACM-R gets the DME deployment status.

14. rApp Manager gets ACM deployment status.

15. rApp Manager request SME to create the entities in rApp instance.

16. SME creates the entities provided by rApp Manager.

17. rApp manager get the response of SME entities creation.
API user provided with the status of rApp instance deployment. rApp Manager combines the status of ACM deployment and SME deployment to provide the rApp instance status.

**Undeploy rApp Instance**

1. API user request to undeploy rApp instance
2. rApp Manager fetches the rApp and rApp instance details
3. rApp Manager request ACM-R to undeploy the ACM instance
4. ACM-R undeploy the AIPMS instance if it is already deployed
5. ACM-R gets the AIPMS undeploy status

6. ACM-R undeploy the Kserve instance if it is already deployed

7. ACM-R gets the Kserve undeploy status

8. ACM-R undeploy the Kubernetes instance if it is already deployed

9. ACM-R gets the Kubernetes undeploy status

10. ACM-R undeploy the DME instance if it is already deployed

11. ACM-R gets the DME undeploy status

12. rApp Manager gets ACM undeploy status

13. rApp Manager request ACM-R to delete the ACM instance
14. rApp Manager get response of ACM instance deletion

15. rApp Manager request SME to delete SME entities

16. SME deletes the entities created as part of rApp

17. rApp Manager get the response of SME entities deletion

18. API user get the status of undeploy rApp instance

**Delete rApp Instance**

1. API user request rApp Manager to delete rApp Instance

2. API user gets delete rApp instance response

Sample rApp package structure
The sample rApp package structure shown below and the location of relevant files for each integration is provided.

ACM (Files /Acm)
- Definition - Files/Acm/Definition (Only one file)
- Instances - Files/Acm/instances

DME (Files /Dme)
- Consumer Information Types - Files/Dme/consumerinfo
types
- Producer Information Types - Files/Dme/producerinfo
types
- Information Consumers - Files/Dme/infoconsumers
- Information Producers - Files/Dme/infoproducers

SME (Files /Sme)
- Providers Functions - Files/Sme/providers
- Service API - Files/Sme/serviceapis
- Invokers - Files/Sme/invokers

CSAR File Generation

This packaging structure is a prototype and subject to change
CSAR file generator is available in the rAppmanager repository here (i-release).

"rappmanager/csar-generator/resources" folder contains the resources to build the rApp package.

The contents of the "resources" directory can be modified as required and the package can be generated as shown below.

### Linux: Generate rApp package

```bash
> ./generate.sh
```

### Windows: Generate rApp package

```bash
> ./generate.bat
```

This will generate a package named "rapp.csar". It can be renamed as required.

This generated package can be used with rApp Manager to create rApp.

## Deployment Instructions

The scripts for the deployments of rApp Manager and its dependent components are available here (i-release).

These scripts are specifically designed for a fresh environment. Some tweaks may be required to run these in an environment where there are some existing installations.

### Pre-requisites

- Kubernetes Cluster (V1.24.6)
- GIT

### Environment setup

The installation scripts shown here installs all the required components.

ACM components should be configured with couple of other components for the participants to work.

In case some of the installation is already setup or not set by the installation scripts, the below environment variables can be used to set the configurations ACM through installation scripts.

Ignore the below variables if the entire environment is being setup by the following installation scripts

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHART_REPO_HOST</td>
<td>Address of the chart repository. It will be used by Kubernetes participant.</td>
<td>http://IP_ADDRESS:8879/charts IP_ADDRESS: IP of the host in which the installation scripts are running.</td>
</tr>
</tbody>
</table>

### Installation

All components can be installed using the command below,
>./install-all.sh

Individual components can be installed using the commands below,
To install the tools required for other installer scripts.

>./install-base.sh

To install the ACM, and it's related components.

>./install-acm.sh

To install the Kserve, and it's related components.

>./install-kserve.sh

To installs the NONRTRIC components.

>./install-nonrtric.sh

**Installed Components**

The below components should be up and running for the rApp Manager integrations to work properly.

**Istio components**

```bash
NAME                    TYPE   CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
pod/istiod-74657777d-69pce  1/1 Running  0 33m
NAME                     READY STATUS   AGE
service/istiod  1/1 1 33m
NAME                     TYPE     UP-TO-DATE AVAILABLE AGE
deployment.apps/istiod   1/1 1 33m
NAME                     DESIRED CURRENT READY AGE
replicaset.apps/istiod-74657777d  1 1 33m
NAME                     REFERENCE TARGETS MINPODS MAXPODS REPlicas AGE
horizontalpodautoscaler.autoscaling/istiod Deployment/istiod <unknown>/80% 1 5 1 33m
```

**Cert Manager components**
Kserve components

ACM Components
```bash
root@gest-selfservice-booking194-k8s-master-0:/home/infra/rappmanager/scripts/install# kubectl get all -n default

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>pod/mariadb-galera-0</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-clamp-ac-alpms-ppnt-5db94f40c89-4mdk4</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-clamp-ac-k8s-ppnt-587499d668-fsnln</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-clamp-ac-kservice-ppnt-6dd9d9906b-7qjqs</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-clamp-runtime-acm-845690d49b-9dfbn</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-galera-init-hrfbf</td>
<td>0/1</td>
<td>Completed</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>pod/policy-models-simulator-788fbd46df-wr7w</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>service/aai-sim</td>
<td>ClusterIP</td>
<td>10.233.10.82</td>
<td>&lt;none&gt;</td>
<td>6666/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/grpc-sim</td>
<td>ClusterIP</td>
<td>10.233.60.235</td>
<td>&lt;none&gt;</td>
<td>6668/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/kubernetes</td>
<td>ClusterIP</td>
<td>10.233.0.1</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>78d</td>
</tr>
<tr>
<td>service/mariadb-galera</td>
<td>ClusterIP</td>
<td>10.233.18.204</td>
<td>&lt;none&gt;</td>
<td>3306/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/mariadb-galera-headless</td>
<td>ClusterIP</td>
<td>10.233.18.204</td>
<td>&lt;none&gt;</td>
<td>4567/TCP,4568/TCP,4444/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/message-router</td>
<td>ClusterIP</td>
<td>10.233.47.23</td>
<td>&lt;none&gt;</td>
<td>3994/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/policy-clamp-ac-alpms-ppnt</td>
<td>ClusterIP</td>
<td>10.233.38.44</td>
<td>&lt;none&gt;</td>
<td>8086/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/policy-clamp-ac-k8s-ppnt</td>
<td>NodePort</td>
<td>10.233.20.192</td>
<td>&lt;none&gt;</td>
<td>8083,30063/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/policy-clamp-kservice-ppnt</td>
<td>ClusterIP</td>
<td>10.233.33.99</td>
<td>&lt;none&gt;</td>
<td>8087/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/policy-clamp-runtime-acm</td>
<td>NodePort</td>
<td>10.233.46.213</td>
<td>&lt;none&gt;</td>
<td>6969,30062/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/sdnc-sim</td>
<td>ClusterIP</td>
<td>10.233.38.76</td>
<td>&lt;none&gt;</td>
<td>6668/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/so-sim</td>
<td>ClusterIP</td>
<td>10.233.25.224</td>
<td>&lt;none&gt;</td>
<td>6669/TCP</td>
<td>35m</td>
</tr>
<tr>
<td>service/vf-c-sim</td>
<td>ClusterIP</td>
<td>10.233.36.42</td>
<td>&lt;none&gt;</td>
<td>6678/TCP</td>
<td>35m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>UP-TO-DATE</th>
<th>AVAILABLE</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>deployment.apps/policy-clamp-ac-alpms-ppnt</td>
<td>1/1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>deployment.apps/policy-clamp-ac-k8s-ppnt</td>
<td>1/1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>deployment.apps/policy-clamp-kservice-ppnt</td>
<td>1/1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>deployment.apps/policy-clamp-runtime-acm</td>
<td>1/1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>deployment.apps/policy-models-simulator</td>
<td>1/1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESIRED</th>
<th>CURRENT</th>
<th>READY</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>replicaset.apps/policy-clamp-ac-alpms-ppnt-5db94f40c89</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>replicaset.apps/policy-clamp-ac-k8s-ppnt-587499d668-fsnln</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>replicaset.apps/policy-clamp-kservice-ppnt-6dd9d9906b-7qjqs</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>replicaset.apps/policy-clamp-runtime-acm-845690d49b-9dfbn</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
<tr>
<td>replicaset.apps/policy-models-simulator-788fbd46df-wr7w</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>35m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>statefulset.apps/mariadb-galera</td>
<td>1/1</td>
<td>35m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>COMPLETIONS</th>
<th>DURATION</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>job.batch/policy-galera-config</td>
<td>1/1</td>
<td>2m25s</td>
<td>35m</td>
</tr>
<tr>
<td>job.batch/policy-galera-init</td>
<td>1/1</td>
<td>66s</td>
<td>35m</td>
</tr>
</tbody>
</table>

NONRTRIC Components
Troubleshooting

1. If Kserve installation failed or end up in the below state after "install-all.sh", Try run "patch-kservice.sh" script after "install-all.sh"

Uninstallation

To uninstall all the components

>./uninstall-all.sh