

Testing End to End call in Release E

This page describes how to demonstrate the entire flow for Information Job/Producer/Consumer management, and A1 Policy Management.

- [Overview of the End to End call](#)
 - Preparation
- [End to end call from Policy Management](#)
 - [End to End call from Policy Control GUI to Near-RT RIC simulator](#)
 - [Perform Policy Operations via Control Panel](#)
- [End to end call from Information Coordinator Service](#)
 - [Type](#)
 - [Create Type](#)
 - [Delete Type](#)
 - [Producers](#)
 - [Create Producer](#)
 - [Delete Producer](#)
 - [Jobs](#)
 - [Create Job](#)
 - [Delete Job](#)

Overview of the End to End call

Details of the architecture can be found from the following link: [Release E](#)

Preparation

It is recommended to use [it/dep repo](#) as it has ready to use helm charts and test data to create and populate data into nonrttrc k8s deployment.

Instruction can be found in [Deploy NONRTRIC in Kubernetes](#).

End to end call from Policy Management



End to End call from Policy Control GUI to Near-RT RIC simulator

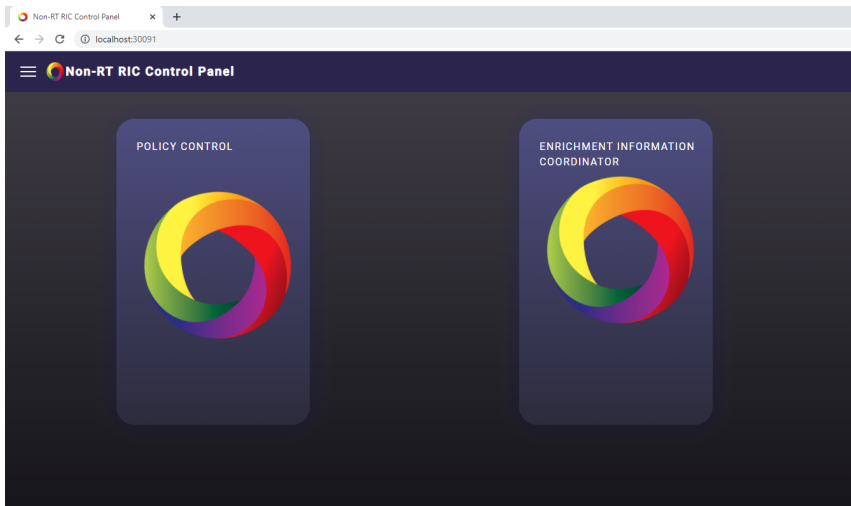
Perform Policy Operations via Control Panel

- Open Control Panel in browser. To check which port your control panel is running, the following command can be used:

```
kubectl get svc -n nonrttrc controlpanel
```

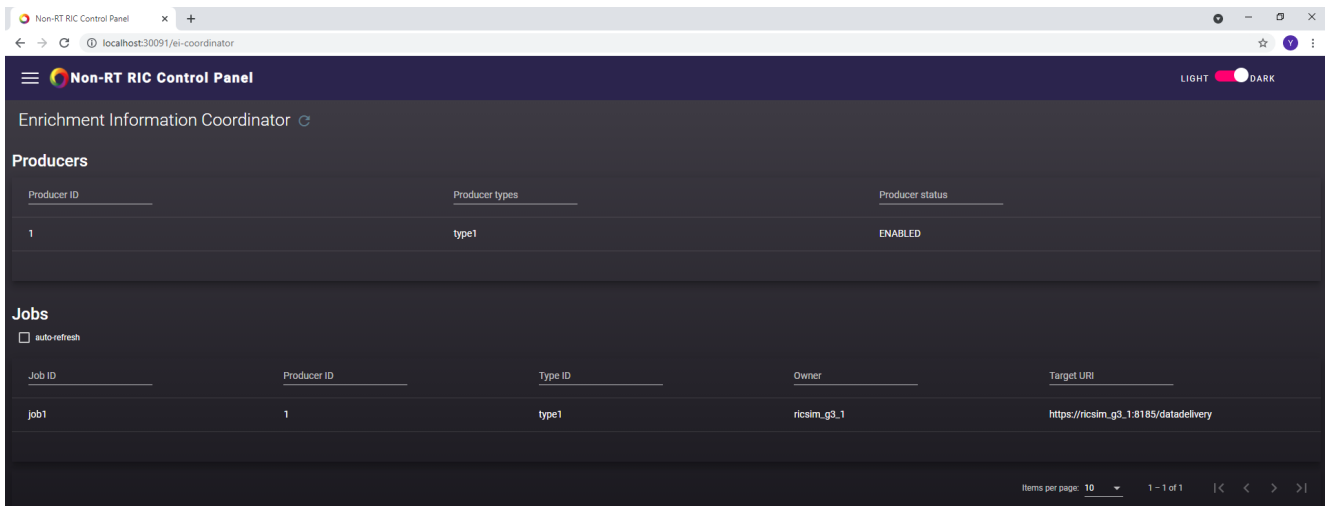
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
controlpanel	NodePort	10.111.91.199	<none>	8080:30091/TCP, 8081:30092/TCP	168m

In this case port for HTTP connection would be 30091 and for HTTPS 30092.



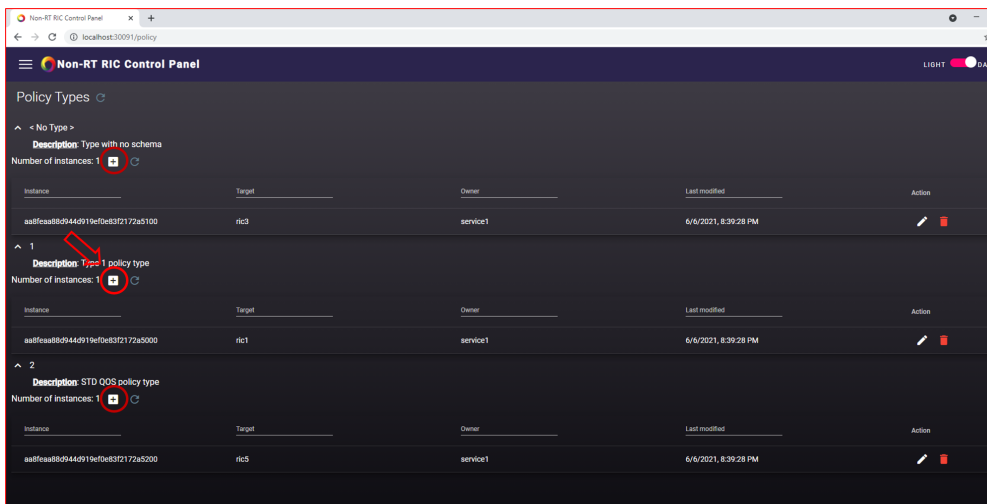
Pic is out of date!

- Click on Policy Control and the screen below with policy types should open

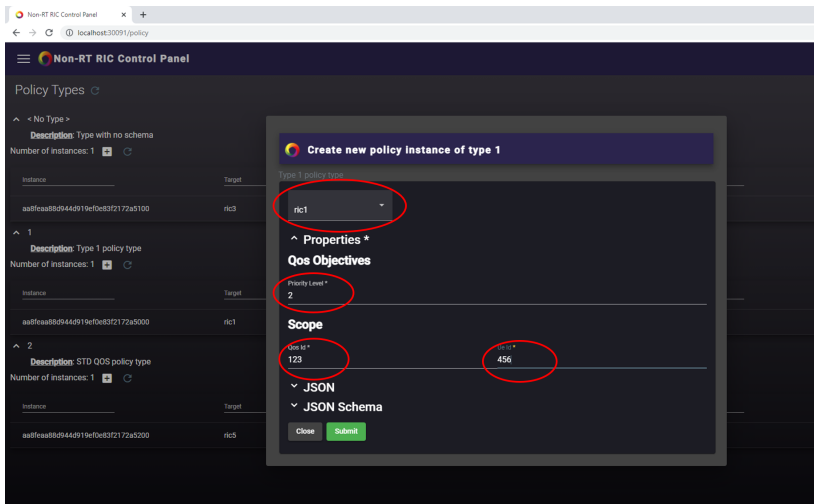


Wrong Pic?!

- For each policy type there is a '+' icon to create a new policy of that specific type. Click on '+' icon to create new policy



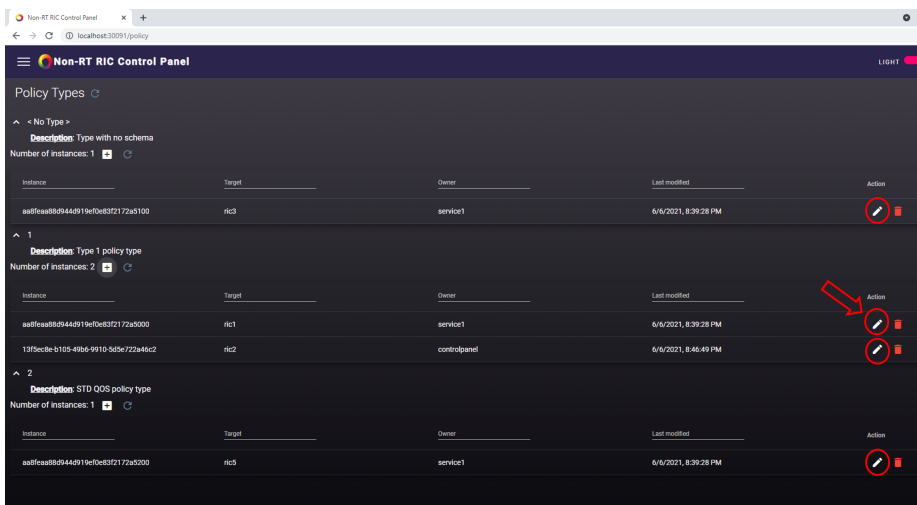
- Choose your *ric* from drop-down menu and provide any arbitrary values for *QoS-ID*, *UE-ID* and *Priority*



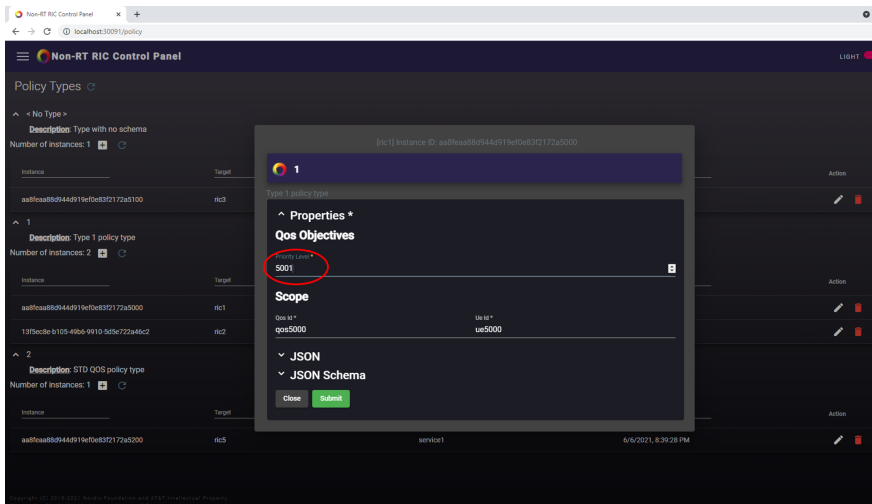
- Submit the policy
- An entry to the chosen Near-RT RIC simulator console/log should appear (see, for example, `kubectll logs -n nonrttric a1-sim-osc-0` in case ric1 policy was created):

```
127.0.0.1 - - [08/Jun/2021 11:35:20] "PUT /a1-p/policytypes/1/policies/fd5f8848-575a-4d1c-887e-b16ad24b22b1 HTTP/1.0" 202 -
```

- Click the pen icon to modify any policy

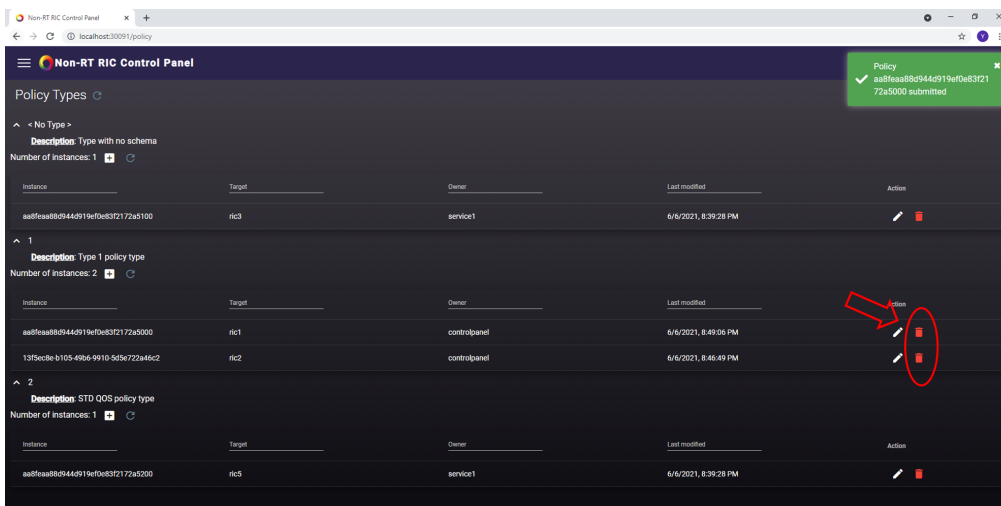


- Modifications are submitted in the same way as in creation, and the resulting simulator log should also be same as the one we saw in creation

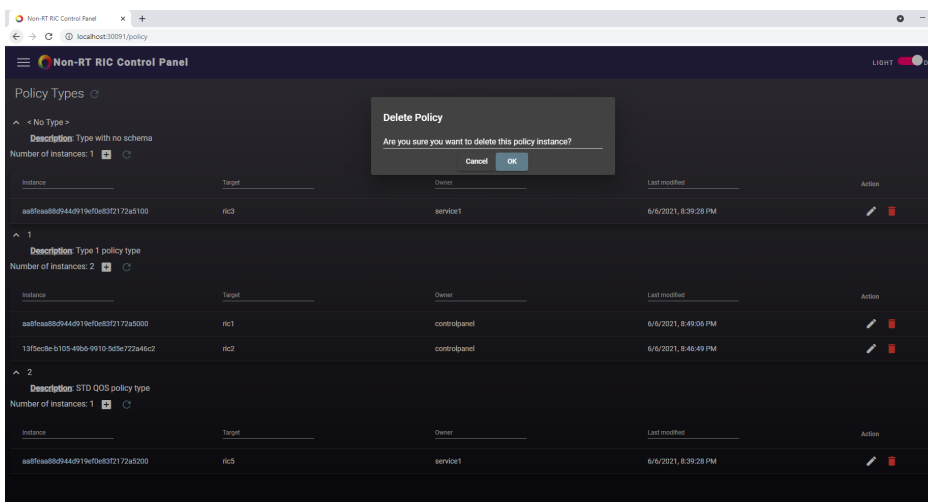


127.0.0.1 - - [08/Jun/2021 11:39:38] "PUT /a1-p/policytypes/1/policies/aa8feaa88d944d919ef0e83f2172a5000 HTTP/1.0" 202 -

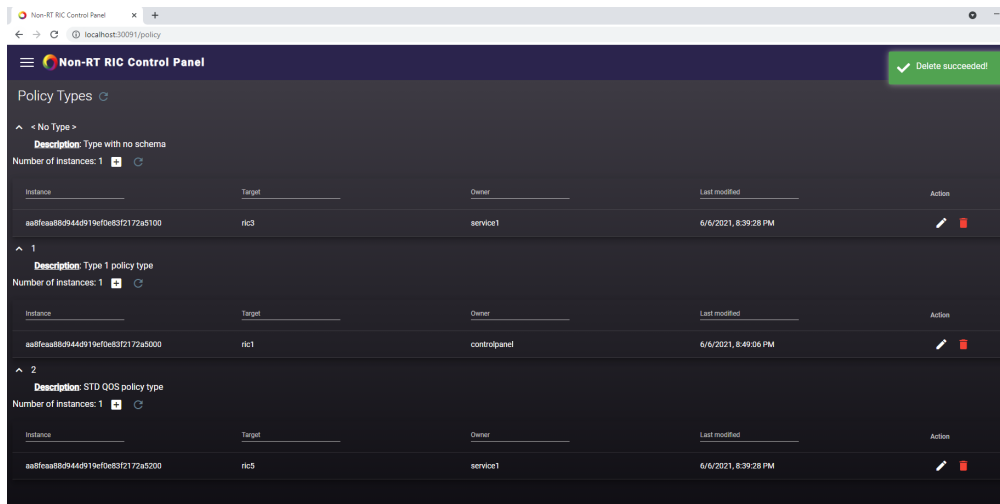
- Delete any policy by clicking the red trashcan icon



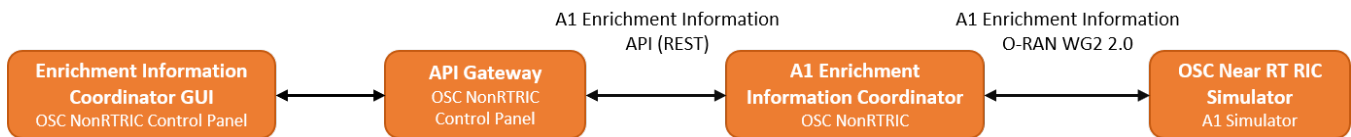
- Confirm deletion and verify the log on simulator side:



```
127.0.0.1 - - [08/Jun/2021 11:43:42] "DELETE /a1-p/policytypes/1/policies/aa8feaa88d944d919ef0e83f2172a5000 HTTP/1.0" 202 -
```

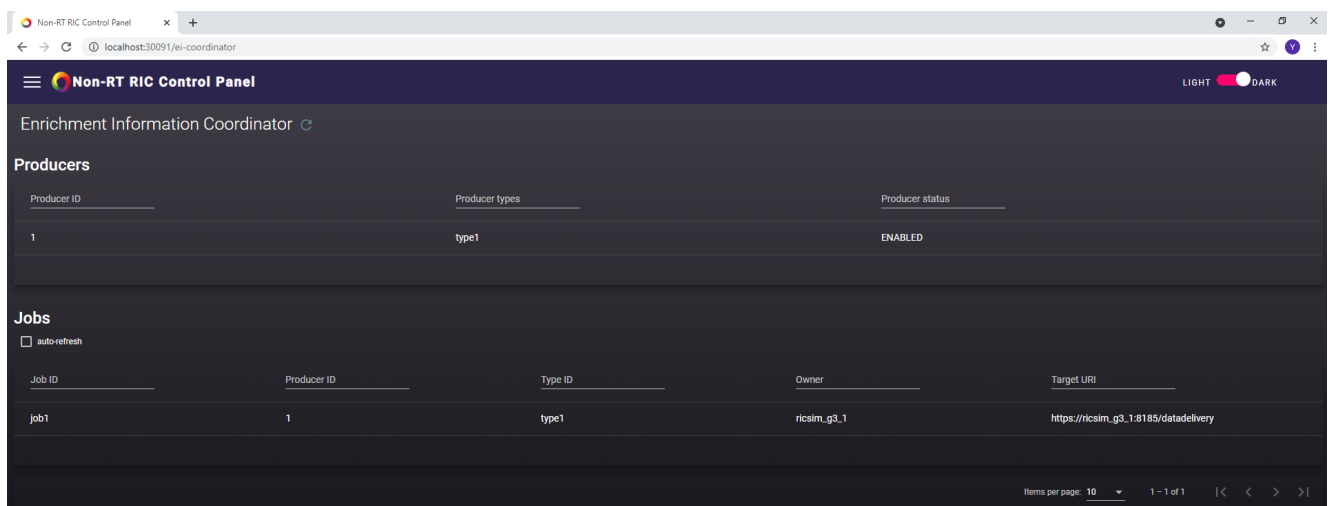


End to end call from Information Coordinator Service



Pic is out of date!

From the Control Panel, Producer and Jobs can only be listed.



Pic is out of date!

The Information Coordination Service (ICS) is not exposed outside the k8s cluster, hence the commands for populating Information Jobs/Consumers /Producers need to be run from within the cluster.

In order to execute other actions such as create, modify and delete, curl can be use to interact with ICS API using NodePort from control panel.

In this example, test data is been used, so there is no real producer connected.

Type

Create Type

```
curl -X 'PUT' \
  'http://localhost:30091/data-producer/v1/info-types/type2' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "info_job_data_schema": {
      "$schema": "http://json-schema.org/draft-07/schema#",
      "title": "STD_Type1_1.0.0",
      "description": "EI-Type 1",
      "type": "object"
    }
  }'
```

Response could be both 200 and 201 depending if the command was used to create a new type or to modify an existing one.

Response:

- 200 Type updated
- 201 Type created

Delete Type

```
curl -X 'DELETE' \
  'http://localhost:30091/data-producer/v1/info-types/type2' \
  -H 'accept: application/json'
```

Response:

- 204 Type deleted
- 404 Type not found

Producers

Create Producer

```
curl -X 'PUT' \
  'http://localhost:30091/data-producer/v1/info-producers/2' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "info_producer_supervision_callback_url": "https://producer-stub:8093/callbacks/supervision/prod-a",
    "supported_info_types": [
      "type2"
    ],
    "info_job_callback_url": "https://producer-stub:8093/callbacks/job/prod-a"
  }'
```

Response could be both 200 and 201 depending if the command was used to create a new producer or to modify an existing one.

Response:

- 200 Producer updated
- 201 Producer created

Delete Producer

```
curl -X 'DELETE' \
'http://localhost:30091/data-producer/v1/info-producers/2' \
-H 'accept: application/json'
```

Response:

- 204 Producer deleted
- 404 Producer not found

Jobs

Create Job

```
curl -X 'PUT' \
'http://localhost:30091/data-consumer/v1/info-jobs/2?typeCheck=false' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
  "info_type_id": "type2",
  "job_result_uri": "https://ricsim_g3_1:8185/datadelivery",
  "job_owner": "ricsim_g3_1",
  "job_definition": {},
  "status_notification_uri": "http://producer:80/"
}'
```

Response could be both 200 and 201 depending if the command was used to create a new job or to modify an existing one.

Response:

- 200 Job updated
- 201 Job created

Delete Job

```
curl -X 'DELETE' \
'http://localhost:30091/data-consumer/v1/info-jobs/1' \
-H 'accept: application/json'
```

Response:

- 204 Job deleted
- 404 Job not found