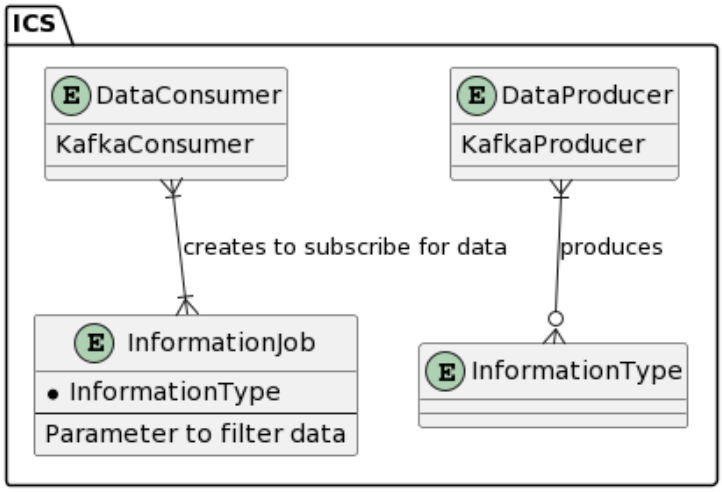
[WIP] Simple guide how to use ICS - Information coordinator service

See also: JIRA link: NONRTRIC-965 - Simple sample rApps using nonrtric/sample/* services IN PROGRESS

what is it (Data management and exposure) Service that manages data subscriptions. It separates data consumers from data producers (for different vendor). Data consumer doesn't need to be aware of where the data source.
where is it: https://github.com/o-ran-sc/nonrtric-plt-informationcoordinatorservice mirror of https://gerrit.o-ran-sc.org/r/nonrtric/plt/informationcoordinatorservice
historical names: Information Coordinator Service (ICS), Enrichment Information Coordinator.

Terminology:

- Information *Type*: Represents the types of data that can be produced by data producers and consumed by data consumers.
- Information *Job*: Represents an active data subscription by a data consumer, specifying the type of data to be produced and additional parameters for filtering.
- Data Consumer: Represents entities that consume data and manage data subscription jobs.
- Data Producer: Represents entities that produce data.



API offered in ICS:

- **Data producer API**: Information *Type* and Information *Producer*
  - Producer *CALLBACKS*: GET *healthcheck* (supervision); Information *Job* Creation/Modification/Delete.
- **Data consumer API**: Information *Type Subscription* Creation/Modification/Delete (REGISTERED/UNREGISTERED); Information *Job* (Creation/Modification/Delete) and GET Information *Type*
  - Consumer *CALLBACKS*: POST Information *Type Status*: REGISTERED/UNREGISTERED invoked when a Information type status has been changed
- **Service status API**: Returns statistics such as Number of Producers Types and Jobs

ICS Docker Image:

1. Building the docker image from source and run it on port 8083 http

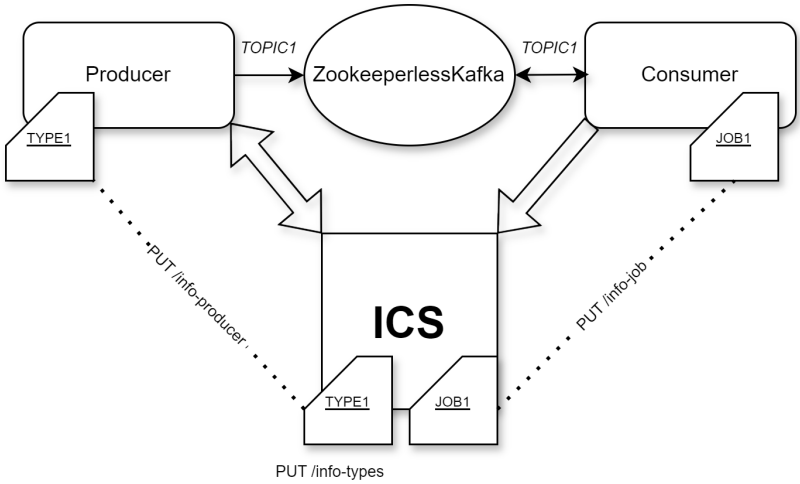
```
git clone "https://gerrit.o-ran-sc.org/r/nonrtric/plt/informationcoordinatorservice"
cd informationcoordinatorservice
mvn clean install
docker run -d -p 8083:8083 o-ran-sc/nonrtric-plt-informationcoordinatorservice:latest
```

Or use the pre-built image

```
docker run -d -p 8083:8083 nexus3.o-ran-sc.org:10001/o-ran-sc/nonrtric-plt-informationcoordinatorservice:1.6.0
```

2. Import the swagger.json in Postman (informationcoordinatorservice/api/ics-api.json) as an OpenAPI3.0
3. Replace the baseUrl with http://localhost:8083 (in the Data management and exposure variables), and change accordingly {{infoTypeId}} from : infoTypeId
Other variables will be :{{infoJobId }}/{{infoProducerId}}/{{infoTypeId}}/{{subscriptionId}} etc

4. ICS flow:
a) Create a type (PUT /info-types)
b) Create a producer (PUT /info-producers) {supports type for filtering}
c) Create a job (PUT /info-jobs) {consumer subscription}



a) ICS type: Stores the data schema of what's sent between the producer and consumer.

PUT {{baseUrl}}/data-producer/v1/info-types/{{infoTypeId}}
Body:

```
{
    "info_job_data_schema": {
        "topicName": "example_topic",
        "key": "example_key",
        "message": "example_message"
    },
    "info_type_information": {}
}
```

b) Onboarding a producer in ICS:

PUT {{baseUrl}}/data-producer/v1/info-producers/{{infoProducerId}}
Body:

```
{
  "supported_info_types": ["example_info_type_id"],
  "info_job_callback_url": "http://example.com/job_callback", //POST JobCallbackUrl() + "/" + infoJob.getId();
  "info_producer_supervision_callback_url": "http://example.com/producer_supervision_callback"
}
```

"jobCallbackUrl" and "producerSupervisionCallbackUrl" are used for communication between a service and external producers in the context of the Information Control Service (ICS).

- *jobCallbackUrl*: This URL serves as a callback endpoint for the producer. When the service needs to communicate or interact with the producer regarding a specific job, it sends requests to this URL. In the stopInfoJob() method, the service constructs a URL by appending the job ID to the jobCallbackUrl of the producer. Then it sends a **DELETE** request to this URL, effectively stopping the job. In the startInfoJob() method, the service sends a **POST** request to the jobCallbackUrl to start a job in the producer.
- *producerSupervisionCallbackUrl*: This URL is used for health checks or supervision purposes. The service can send requests to this URL to check the health or status of the producer. In the healthCheck() method, the service sends a **GET** request to the producerSupervisionCallbackUrl to check the health of the producer.

In summary, both URLs facilitate communication between the service and external producers, enabling actions like starting and stopping jobs, as well as monitoring the health and status of the producers.

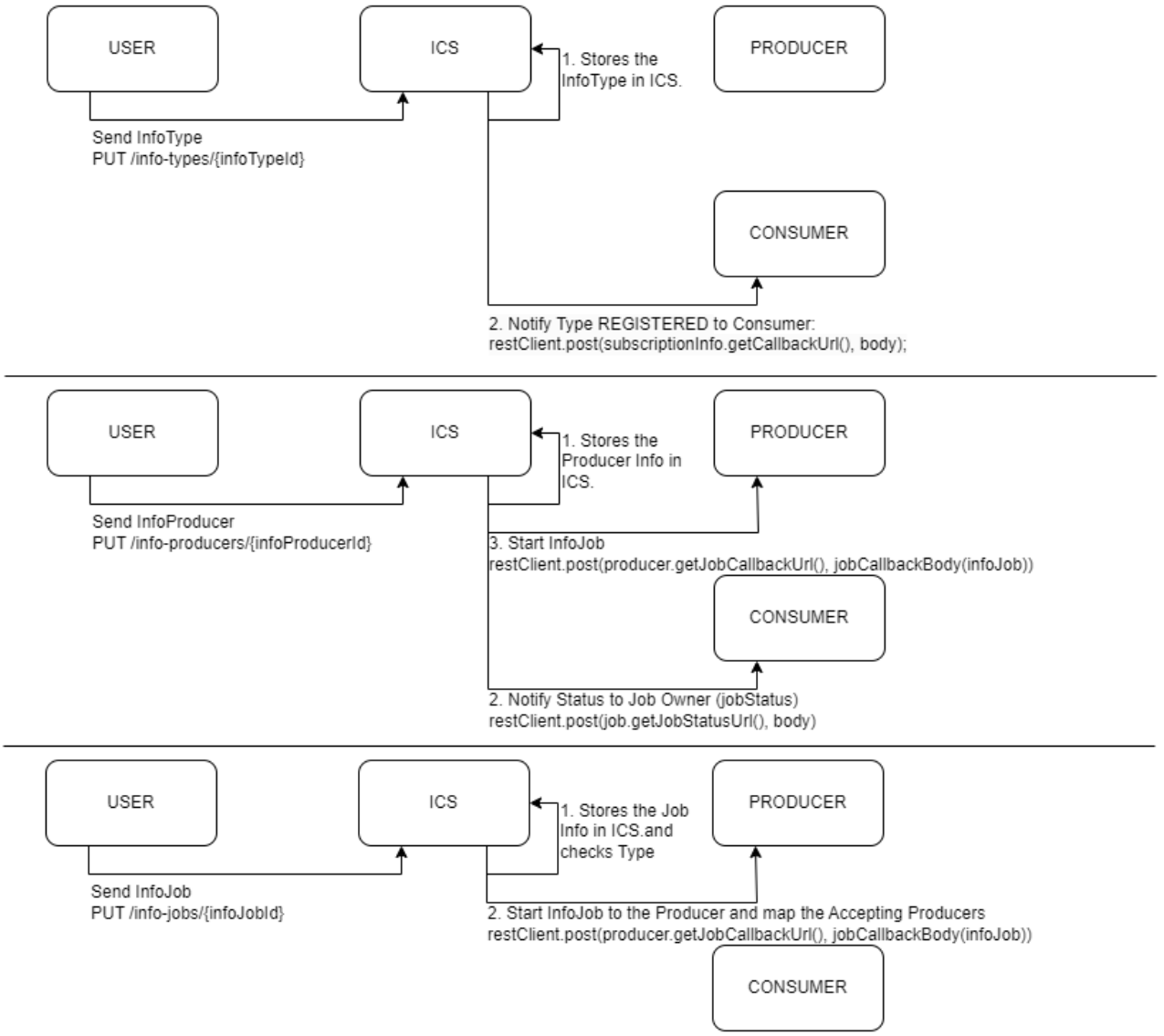
c) Giving the consumer a job definition:

PUT {{baseUrl}}/data-consumer/v1/info-jobs/{{infoJobId}}
Body:

```
{
  "info_type_id": "example_info_type_id",
  "job_owner": "example_owner",
  "job_definition": {
    "example_key1": "example_value1",
    "example_key2": "example_value2"
  },
  "job_result_uri": "http://example.com/job_result",
  "status_notification_uri": "http://example.com/status_notification"
}
```

1. Authorization check: POST to the Authentication Agent (from the starting config *config/application.yaml*)
2. Validation: The URLs seem to be used only for URI validation (?)
3. Consumer starts a job on the Producer POST producerCallbacks.startInfoSubscriptionJob->restClient.post(producer.getJobCallbackUrl(), jobCallbackBody(infoJob))

ICS Callbacks Flow



Demo Application - Java Producer and Consumer

WIP application: https://gerrit.nordix.org/c/local/oransc/nonrtric-prototyping/+/20750
Script for the demo: https://gerrit.nordix.org/gitweb?p=local%2Foransc%2Fnonrtric-prototyping.git;hb=refs%2Fchanges%2F50%2F20750%2F15;f=kafka-demo-app%2Fdemo2.sh

Running the script will check the requirements and start 3 containers: DemoApp(localhost:8080), Kafka(localhost:9092), ICS(localhost:8083)

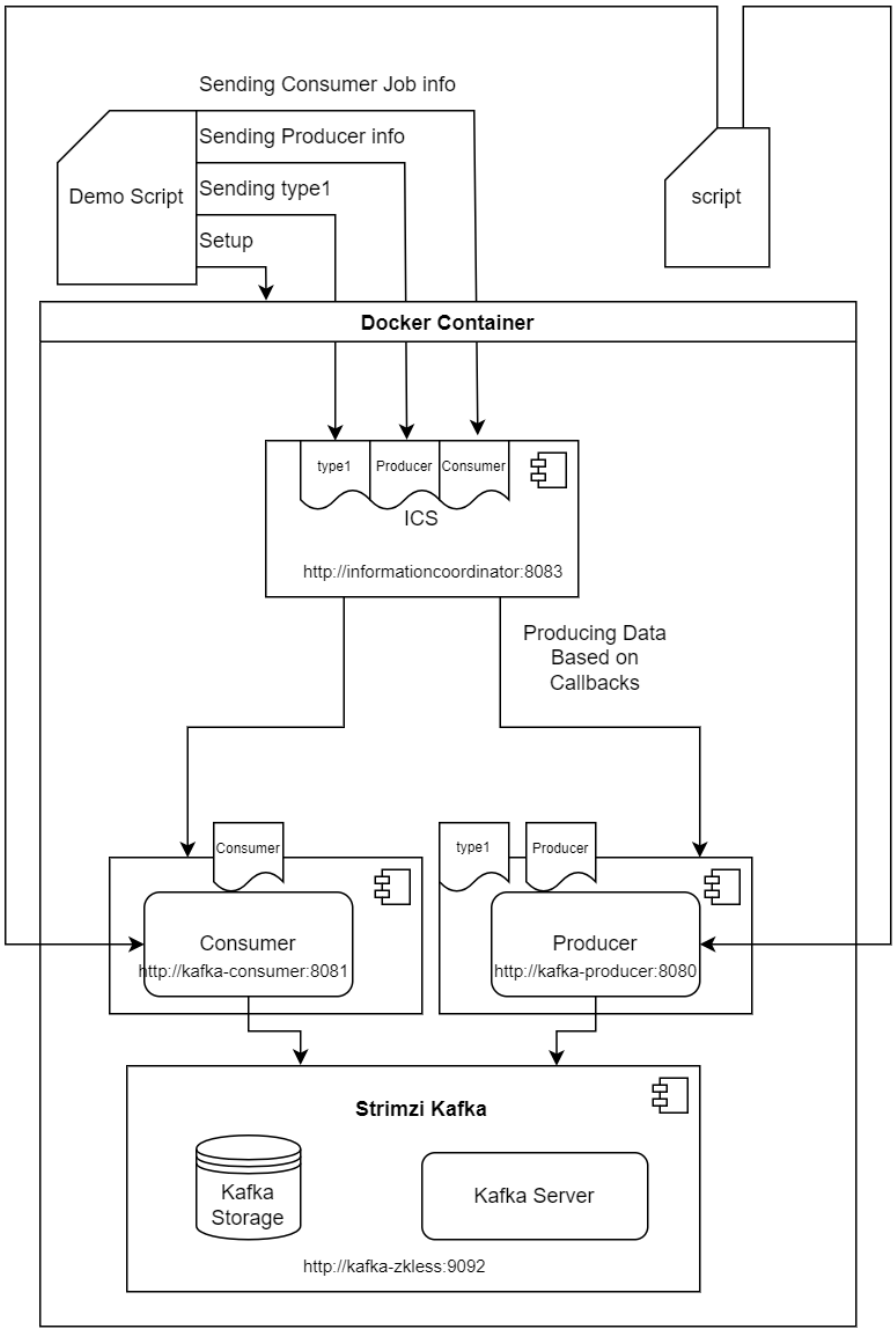
The demo application must implement these callbacks in order to work with ICS:

1. GET *SUPERVISION_URL Return 200*
2. DELETE *JOB_URL + "/{infoJobId}" Return 200*
3, GET *JOB_URL Return 200 and a collection of JOB*
4. POST JOB_URL Return 200 and send in body a JOB

This also assumes that the Demo Application has a definition of a TYPE and a JOB on that type.

Run the demo:
The demo.sh script will:

1. Check the system for dependencies such as Maven, Java, Docker and docker-compose
2. Package the demo application for a producer and a consumer and build the docker images
3. Start the docker container in the same docker network with docker-compose
4. After Strimzi kafka is up and running the user can run manually ./runproducer.sh and ./runconsumer.sh in different shells or use demo.sh to start the producer and consumer
5. The script will send type1 to ICS, which is already predefined in the demo application
6. The script will send the producer info to ICS
7. The script will send the consumer job info to ICS
8. ICS will trigger the demo application based on its callbacks
9. Data is produced on the demo application
10. The script sends docker logs of the Producer Callback function of ICS
11. The script sends docker logs of the Demo applications


Sending Consumer Job info
Sending Producer info
Sending type1
Setup
Demo Script
script
Docker Container
type1
Producer
Consumer
ICS
http://informationcoordinator:8083
Producing Data
Based on
Callbacks
Consumer
Consumer
http://kafka-consumer:8081
type1
Producer
Producer
http://kafka-producer:8080
Strimzi Kafka
Kafka
Storage
Kafka Server
http://kafka-zkless:9092

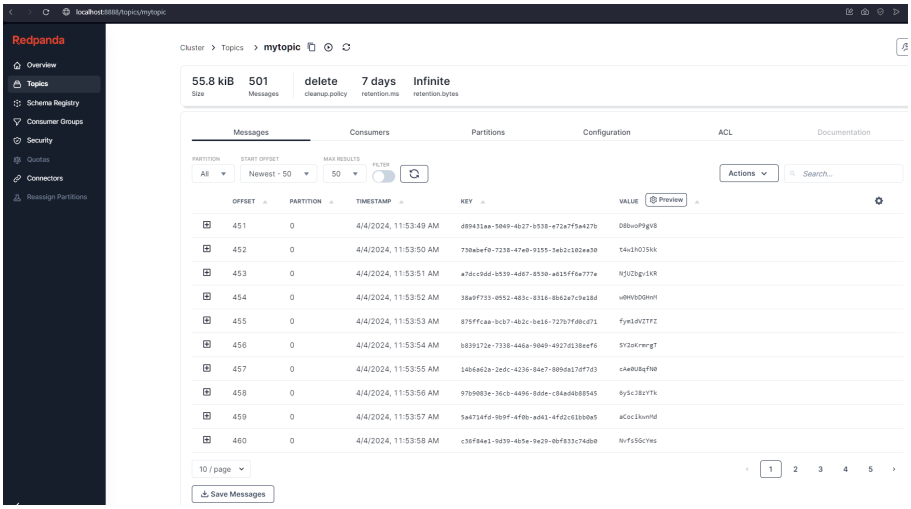
```
Demo Producer Docker logs

2024-04-02 12:48:05 INFO  c.d.p.p.SimpleProducer:141 - {"bootstrapServers":"kafka-zkless:9092","topic":"
mytopic","source":"com.demo.producer.producer.SimpleProducer","message":"ygHwxXSIxW","key":"f8f1a7a7-a78e-4c7d-
9b8d-108bb0cc9e2c"}
2024-04-02 12:48:06 INFO  c.d.p.p.SimpleProducer:141 - {"bootstrapServers":"kafka-zkless:9092","topic":"
mytopic","source":"com.demo.producer.producer.SimpleProducer","message":"KNIbP10zfN","key":"b058d00f-bbcd-4d2c-
936b-6327847d4c2a"}
2024-04-02 12:48:07 INFO  c.d.p.p.SimpleProducer:141 - {"bootstrapServers":"kafka-zkless:9092","topic":"
mytopic","source":"com.demo.producer.producer.SimpleProducer","message":"V6fH1NkdeH","key":"ae1a83a3-d8a7-40c8-
9d98-529230f8b585"}
2024-04-02 12:48:08 INFO  c.d.p.p.SimpleProducer:141 - {"bootstrapServers":"kafka-zkless:9092","topic":"
mytopic","source":"com.demo.producer.producer.SimpleProducer","message":"m76qvRFh6f","key":"abccde52-fa72-4fd4-
99ab-5bc21514d825"}
2024-04-02 12:48:09 INFO  c.d.p.p.SimpleProducer:141 - {"bootstrapServers":"kafka-zkless:9092","topic":"
mytopic","source":"com.demo.producer.producer.SimpleProducer","message":"t7FJYnFr43","key":"0602239e-34e9-45a6-
a04a-3c67b4c7d9e4"}

++++++++++++++++++++++++++++++++++++++++++++++++++++++

Demo Consumer Docker logs

2024-04-02 12:48:05 INFO  c.d.c.c.SimpleConsumer:158 - {"message":"Topic: mytopicMessage: ygHwxXSIxW"}
2024-04-02 12:48:06 INFO  c.d.c.c.SimpleConsumer:158 - {"message":"Topic: mytopicMessage: KNIbP10zfN"}
2024-04-02 12:48:07 INFO  c.d.c.c.SimpleConsumer:158 - {"message":"Topic: mytopicMessage: V6fH1NkdeH"}
2024-04-02 12:48:08 INFO  c.d.c.c.SimpleConsumer:158 - {"message":"Topic: mytopicMessage: m76qvRFh6f"}
2024-04-02 12:48:09 INFO  c.d.c.c.SimpleConsumer:158 - {"message":"Topic: mytopicMessage: t7FJYnFr43"}

++++++++++++++++++++++++++++++++++++++++++++++++++++++

ICS logs

2024-04-02T12:48:05.615Z DEBUG 1 --- [or-http-epoll-2] o.o.i.c.r1producer.ProducerCallbacks     : Job
subscription 1 started OK 1
2024-04-02T12:48:05.820Z DEBUG 1 --- [io-8083-exec-10] o.o.i.repository.InfoTypeSubscriptions   : Added type
status subscription 1
```

Redpanda Console:

After kafka is up and running

docker-compose -f docker-composeRedPanda.yaml up -d

Redpanda console available at: http://localhost:8888



NONRTRIC-controlpanel:

git clone "https://gerrit.o-ran-sc.org/r/portal/nonrtric-controlpanel"

Changed the configuration files as shown here:

```
/nonrtric-controlpanel$ git diff ./docker-compose/
diff --git a/docker-compose/.env b/docker-compose/.env
index 69fdd12..257ac73 100644
--- a/docker-compose/.env
+++ b/docker-compose/.env
@@ -16,9 +16,9 @@
 #

 #CONTROL_PANEL
-CONTROL_PANEL_IMAGE_BASE="nexus3.o-ran-sc.org:10004/o-ran-sc/nonrtric-controlpanel"
+CONTROL_PANEL_IMAGE_BASE="nexus3.o-ran-sc.org:10002/o-ran-sc/nonrtric-controlpanel"
 CONTROL_PANEL_IMAGE_TAG="2.5.0"

 #NONRTRIC_GATEWAY
-NONRTRIC_GATEWAY_IMAGE_BASE="nexus3.o-ran-sc.org:10004/o-ran-sc/nonrtric-gateway"
+NONRTRIC_GATEWAY_IMAGE_BASE="nexus3.o-ran-sc.org:10002/o-ran-sc/nonrtric-gateway"
 NONRTRIC_GATEWAY_IMAGE_TAG="1.2.0"
\ No newline at end of file
diff --git a/docker-compose/control-panel/docker-compose.yaml b/docker-compose/control-panel/docker-compose.yaml
index 2716ed8..a42b7a8 100644
--- a/docker-compose/control-panel/docker-compose.yaml
+++ b/docker-compose/control-panel/docker-compose.yaml
@@ -16,18 +16,17 @@
 version: '3.5'

 networks:
-  default:
-    driver: bridge
-    name: nonrtric-docker-net
+  kafka:
:...skipping...
diff --git a/docker-compose/.env b/docker-compose/.env
index 69fdd12..257ac73 100644
--- a/docker-compose/.env
+++ b/docker-compose/.env
@@ -16,9 +16,9 @@
 #

 #CONTROL_PANEL
-CONTROL_PANEL_IMAGE_BASE="nexus3.o-ran-sc.org:10004/o-ran-sc/nonrtric-controlpanel"
+CONTROL_PANEL_IMAGE_BASE="nexus3.o-ran-sc.org:10002/o-ran-sc/nonrtric-controlpanel"
 CONTROL_PANEL_IMAGE_TAG="2.5.0"

 #NONRTRIC_GATEWAY
-NONRTRIC_GATEWAY_IMAGE_BASE="nexus3.o-ran-sc.org:10004/o-ran-sc/nonrtric-gateway"
+NONRTRIC_GATEWAY_IMAGE_BASE="nexus3.o-ran-sc.org:10002/o-ran-sc/nonrtric-gateway"
 NONRTRIC_GATEWAY_IMAGE_TAG="1.2.0"
\ No newline at end of file
diff --git a/docker-compose/control-panel/docker-compose.yaml b/docker-compose/control-panel/docker-compose.yaml
index 2716ed8..a42b7a8 100644
--- a/docker-compose/control-panel/docker-compose.yaml
+++ b/docker-compose/control-panel/docker-compose.yaml
@@ -16,18 +16,17 @@
 version: '3.5'

 networks:
-  default:
-    driver: bridge
-    name: nonrtric-docker-net
+  kafka:
+    external: true

 services:
   policy-control-panel:
     image: "${CONTROL_PANEL_IMAGE_BASE}:${CONTROL_PANEL_IMAGE_TAG}"
```

```
     container_name: policy-control-panel
     networks:
-      - default
+      - kafka
     ports:
-      - 8080:8080
-      - 8082:8082
+      - 8181:8080
+      - 8282:8082
     volumes:
       - ./control-panel/config/nginx.conf:/etc/nginx/nginx.conf:ro
diff --git a/docker-compose/docker-compose.yaml b/docker-compose/docker-compose.yaml
index 64bcbb9..c30aa2c 100644
--- a/docker-compose/docker-compose.yaml
+++ b/docker-compose/docker-compose.yaml
@@ -16,6 +16,5 @@
 version: '3.5'

 networks:
-  default:
-    driver: bridge
-    name: nonrtric-docker-net
\ No newline at end of file
+  kafka:
+    external: true
diff --git a/docker-compose/nonrtric-gateway/config/application-nonrtricgateway.yaml b/docker-compose/nonrtric-
gateway/config/application-nonrtricgateway.yaml
index 7230175..c620b03 100644
--- a/docker-compose/nonrtric-gateway/config/application-nonrtricgateway.yaml
+++ b/docker-compose/nonrtric-gateway/config/application-nonrtricgateway.yaml
@@ -26,12 +26,8 @@ spring:
       httpserver:
         wiretap: true
       routes:
-      - id: A1-Policy
-        uri: https://policy-agent:8433
-        predicates:
-        - Path=/a1-policy/**
       - id: A1-EI
-        uri: https://ics:8434
+        uri: http://informationcoordinatorservice:8083
         predicates:
         - Path=/data-producer/**,/data-consumer/**
 management:
diff --git a/docker-compose/nonrtric-gateway/docker-compose.yaml b/docker-compose/nonrtric-gateway/docker-
compose.yaml
index fbf3d9b..138229d 100644
--- a/docker-compose/nonrtric-gateway/docker-compose.yaml
+++ b/docker-compose/nonrtric-gateway/docker-compose.yaml
@@ -16,16 +16,15 @@
 version: '3.5'

 networks:
-  default:
-    driver: bridge
-    name: nonrtric-docker-net
+  kafka:
+    external: true

 services:
   nonrtric-gateway:
     image: "${NONRTRIC_GATEWAY_IMAGE_BASE}:${NONRTRIC_GATEWAY_IMAGE_TAG}"
     container_name: nonrtric-gateway
     networks:
-      default:
+      kafka:
         aliases:
           - nonrtric-gateway-container
     ports:
(END)
```

```
docker-compose -f ./nonrtric-controlpanel/docker-compose/docker-compose.yaml \
        -f ./nonrtric-controlpanel/docker-compose/control-panel/docker-compose.yaml \
        -f ./nonrtric-controlpanel/docker-compose/nonrtric-gateway/docker-compose.yaml up -d
```

ICS informations displayed here: http://localhost:8181