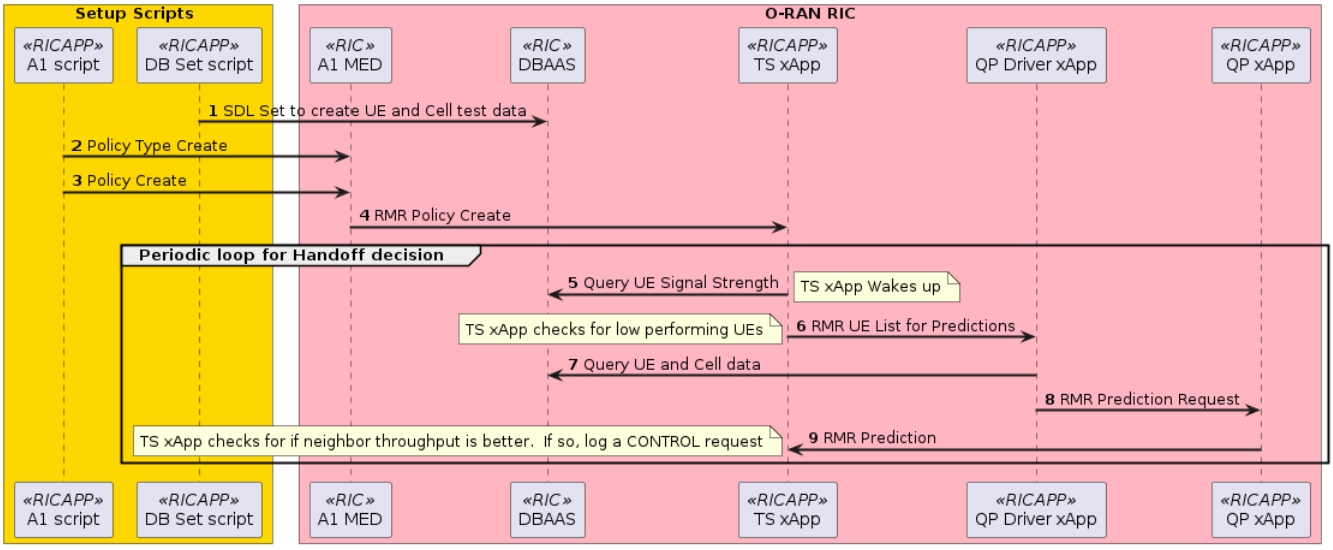
Traffic Steering Use Case

Call flow Diagram



Status

To be tested

Delivery Status

	OTF	OAM	NONRTRIC	RICP	RICAPP	O-DU	Test Result	Notes
1. SDL set up (RICP.DBaaS)								
2. Policy Type Create (RICP.A1)								
3. Policy Create (RICP.A1)								
4. RMR Policy Create (RICP.A1 RICAPP.TS)								
5. Query for UE Signal Strength (RICAPP.TS RICP.DBaaS)								
6. RMR UE List for Prediction (RICAPP.TS RICAPP.QPD)								
7. Query UE and Cell data (RICAPP.QPD DBaaS)								
8. RMR Prediction Request (RICAPP.QPD RICAPP.QP)								
9. RMR Prediction (RICAPP.QP RICAPP.TS)								

Test Manual

Pre-population of test data in DBAAS

In order to trigger the loop in the Traffic Steering use case, we must populate test data in DBAAS. This includes both cell and UE entries.

This is done through a script. Follow these directions to run the script:

git clone "https://gerrit.o-ran-sc.org/r/ric-app/ts"

cd ts/ts/test/populatedb

chmod a+x populate_db.sh

./populate_db.sh

This script will build a docker image locally and also install a helm chart in the kubernetes cluster to run the image.

The code that is run will write the necessary data to SDL

Onboarding and Deployment of xApps

The Use case involves three different xApps:

- Traffic Steering (TS) xApp
- QoE Prediction Driver (QPDriver) xApp
- QoE Prediction (QP) xApp

Each of them have a descriptor in their gerrit repo under the xapp-descriptor/ directory.

None of them have xapp specific controls and therefore no individual json schema

Here are the URLs for each which can be included in HTTP POST call to onboard tool

TS xApp:

https://gerrit.o-ran-sc.org/r/gitweb?p=ric-app/qp-driver.git;a=blob_plain;f=xapp-descriptor/config.json;hb=HEAD

QPDriver xApp:

https://gerrit.o-ran-sc.org/r/gitweb?p=ric-app/qp-driver.git;a=blob_plain;f=xapp-descriptor/config.json;hb=0a0dff549b933f049b05fc26ce2f0a13da853b78

QP xApp

https://gerrit.o-ran-sc.org/r/gitweb?p=ric-app/qp.git;a=blob_plain;f=xapp-descriptor/config.json;hb=HEAD

Here is the URL for the Xapp Onboarder in your environment. The values 'ingress_host', 'ingress_port_http' and 'xapp_onboarder_path' refer to the hostname and port for reaching Kong ingress controller, and the ingress path assigned to xapp onboarder (likely the path is set to 'onboard').

http://{{ingress_host}}:{{ingress_port_http}}/{{xapp_onboarder_path}}/api/v1/onboard/download

As an example, the message body for the TS xapp is below. This needs to be sent in a POST request with Content-Type equal to 'application/json'

```
{
      "config-file.json_url": "https://gerrit.o-ran-sc.org/r/gitweb?p=ric-app/qp-driver.git;a=blob_plain;f=xapp-descriptor/config.json;hb=HEAD"
}
```

To subsequently deploy any of these xapps, use the following command:

curl --location --request POST "http://{{ingress_host}}:{{ingress_port_http}}/{{xapp_onboarder_path}}/ric/v1/xapps" --header 'Content-Type: application/json' --data-raw '{"xappName": "trafficxapp"}'

(where xappName would be later set to 'qpdriver' and 'qp' to deploy the QP Driver and QP xapps respectively)

Creation of Traffic Steering Policy Type and Policy Instance

TS xApp consumes an A1 Policy

Below are the steps for providing it with a policy.

Policy Type Create

Copy the following into a file called create.json -

Policy Type Create

```
{"name" : "tsapolicy", "description" : "tsa parameters", "policy_type_id" : 20008, "create_schema" : {
"$schema" : "http://json-schema.org/draft-07/schema#", "type" : "object", "properties" : { "threshold" : {
"type" : "integer", "default" : 0 } }, "additionalProperties" : false } }
```

Then run:

curl -X PUT --header "Content-Type: application/json" --data-raw @create.json http://<Base URL for Kong>/a1mediator/a1-p/policytypes/20008

Policy Create

Run command:

curl -X PUT --header "Content-Type: application/json" --data "{\"threshold\" : 5}" http://<Base URL for Kong>/a1mediator/a1-p/policytypes/20008/policies/tsapolicy145