

ITU ML5G Build-a-thon

Demonstrate POC done as part of
Activity-4 –ORAN Control Loop
Instantiation

Team -RAN-RIC-xApp
Deena Mukundan
Divyani Achari

GOALS

Goal-1 On reception of Emergency Intent According to the given requirements fetch model from Acumos.

Goal-2 Deploy the model as xApp in ORAN. A pre-trained model might be used for this purpose -

Build-a-thon Activity-4

Achievements

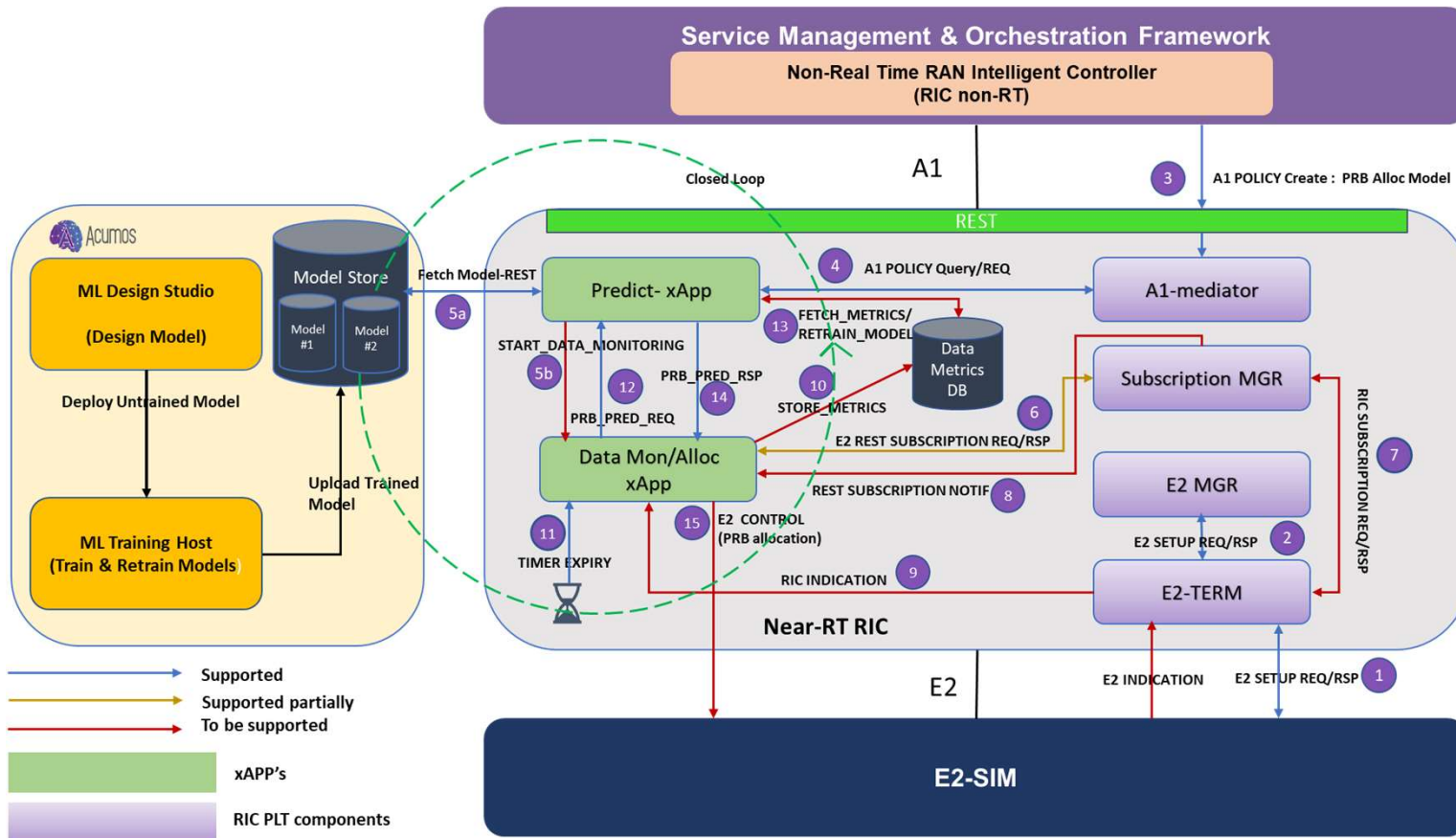
Brought up ORAN-RIC platform with Dawn release content and attained hands-on knowledge on RIC Platform

Implemented & Deployed 2 custom xApps in RIC platform that interacts with RIC platform components and amongst each other

Implemented capability to dynamically fetch model from a remote model store based on A1 policy configured

Brought up E2 SIM process and registered with RIC E2 components

Workflow- Network Resource Allocation



- RIC is UP and Running, RAN(E2-SIM in this case) is registered/associated with RIC
- RIC receives policy update from A1 for triggering closed loop PRB Allocation
- Fetch model based on the A1 Policy details
- Based on RAN data monitored, predict PRB utilisation [test data was used for POC instead of actual data from E2]
- Compute the PRB to be allocated and send E2 control message. PRB's are always reserved for Emergency Slide and additionally resources can be reallocated based on situational considerations
- Continuously monitor, evaluate and improve decision

Note: E2 Indication is for future reference, currently data is not received via RIC Indication.
In future, based on subscription E2 interface will receive data via RIC Indication.

Credit Note: The pre-trained model, model specific implementation and PRB allocation ALG1 developed by Team "AUTOMATO" as part of this build-a-thon is re-used for this POC

WorkFlow

Points 1&2 show E2 SIM is up and association with RIC is setup

Point 3 nRT RIC receives A1 policy update to trigger closed loop monitoring

Point 4 A1-mediator sends A1 Policy REQ to prbpred xApp

Point 5a, Point 5b show the model is fetched from model store as per policy guidelines and prbPred instructs DataMon/Alloc xApp to start monitoring the data

Point 6,7,8 shows the messaging done for subscribing to E2 for data

Point 9 shows Data reception from E2 node. The received metrics are stored in metrics DB as in Point 10

Upon timer expiry as in Point 11, request for Prediction is sent to prbpred xApp as in Point 12.

prbPred uses ML model to predict the future utilisation. Retraining may be done and sends predicted values to Datamon/Alloc xApp as in Points 13,14

DataMon/Alloc xApp computes the PRB to be allocated and sends E2 control message towards E2 as in Point 15

xApp POC Implementation Details

Prbpred- xApp is developed as Reactive xApp

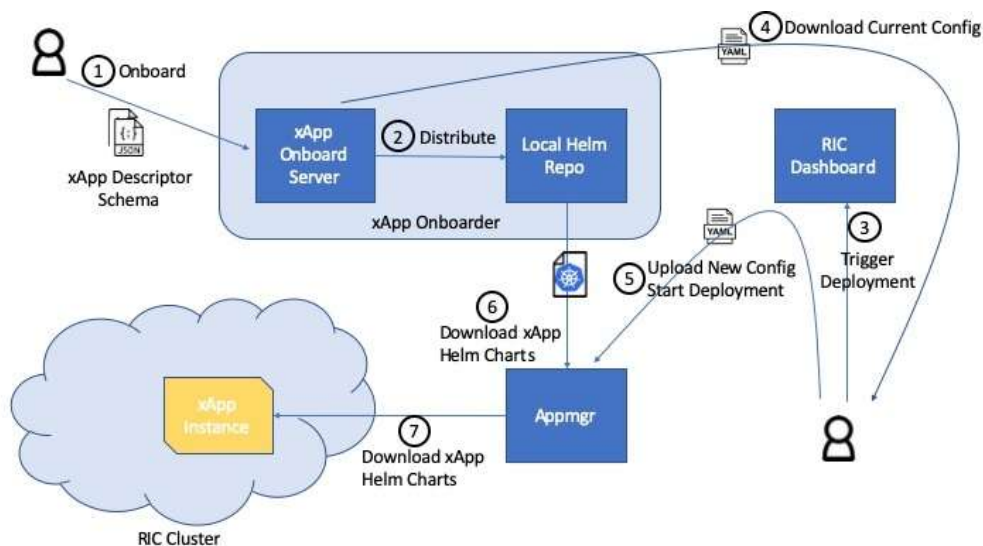
- i. Upon Initialization does following :-
 - i. Registers for PRB_PRED_REQ (PRB Prediction Request) and A1_POLICY_REQ (A1 Policy Request)
 - ii. xApp provides details of the policy supported in xApp-descriptor and registers handler function to receive A1_POLICY_REQ
 - iii. Queries A1-mediator and gets the Policy details
 - iv. As part of step 4,5a in the sequence diagram mentioned in Reference [4]. A specific policy was created which gives info on model and model version info to be used.
- ii. Based on the policy details xApp fetches the model from remote model store by constructing the URL based on the information received and stores it locally
- iii. Upon reception of PRB_PRED_REQ, based on the model fetched performs prediction of PRB utilisation for each slice and sends response to Alloc xApp
- iv. Following are the message types handled by this xApp
 - i. Outgoing Message types: - "A1_POLICY_RESP", "PRB_PRED_RSP", "A1_POLICY_QUERY"
 - ii. Incoming Message Types - "PRB_PRED_REQ", "A1_POLICY_REQ"

alloc xApp – is developed as proactive xApp

- i. Is as Proactive xApp
- ii. Upon Initialization does following:-
 - i. registers with subscription mgr. for E2 information
 - ii. And starts timer to trigger PRB_PRED_REQ periodically.
- iii. Based on predicted future PRB utilisation computes PRB to be allocated for emergency slice
- iv. Emergency slice has reserved PRB's, in addition the remaining unutilised PRB's are allocated
- v. Alloc xApp sends the E2 control message to allocate the available PRB's from the computation
- vi. Following are the message types handled by this xApp
 - i. Outgoing Message types: - "PRB_PRED_REQ", "RIC_HEALTH_CHECK_RESP"
 - ii. Incoming Message Types - "PRB_PRED_RESP", "SUBSCRIPTION_REQ", "RIC_HEALTH_CHECK_REQ"

In the Dawn release, creation of A1 policy instance doesn't trigger A1 Policy create message towards xApp. This was confirmed by ORAN-RIC team <https://wiki.o-ran-sc.org/display/IAT/Traffic+Steering+Flows?focusedCommentId=41456537#comment-41456537>. Hence the workflow was modified to send timer-based event from alloc XApp to trigger PRB prediction.

On-boarding and Deploying xApps



xApp Onboarding Instructions through DMS CLI

```
docker run --rm -u 0 -it -d -p 9090:8080 -e DEBUG=1 -e STORAGE=local -e
STORAGE_LOCAL_ROOTDIR=/charts -v $(pwd)/charts:/charts chartmuseum/chartmuseum:latest
export CHART_REPO_URL=http://0.0.0.0:9090
dms_cli onboard --config_file_path=config.json --
shcema_file_path=/root/appmgr/xapp_orchestrator/dev/docs/xapp_onboarder/guide/embedded-
schema.json
dms_cli install --xapp_chart_name=prbpredxapp --version=0.0.2 --namespace=ricxapp
dms_cli install --xapp_chart_name=alloc --version=0.0.2 --namespace=ricxapp
```

Reference : <https://wiki.o-ran-sc.org/display/RICA/On-boarding+and+Deploying+xApps>

- 1.To onboard an xApp, the xApp descriptor and its schema will be submitted to the xApp onboarder. (ADD LINK TO API DOC)
- 2.xApp onboarder generates helm charts and distributes them to the local helm repo in the RIC platform instance
- 3.Operator triggers xApp deployment
- 4.(OPTIONAL)Through RIC dashboard, download an values.yaml file that contains the default xApp configuration parameters
- 5.(OPTIONAL) Modify the xApp configuration parameters, upload the new configuration to appmgr
- 6.Appmgr combines the xApp helm charts from local helm repo and the new configuration
- 7.Appmgr creates an xApp instance

Prbpred-xAppDescriptor

```
{
  "xapp_name": "prbpredxapp",
  "version": "0.0.2",
  "containers": [
    {
      "name": "prbpredxapp",
      "image": {
        "registry": "nexus3.o-ran-sc.org:10002",
        "name": "o-ran-sc/ric-app-prbpred",
        "tag": "0.0.2"
      }
    }
  ],
  "messaging": {
    "ports": [
      {
        "name": "http",
        "container": "prbpredxapp",
        "port": 10003,
        "description": "http service"
      },
      {
        "name": "rmr-data",
        "container": "prbpredxapp",
        "port": 4560,
        "rxMessages": [
          "A1_POLICY_REQ",
          "PRB_PRED_REQ"
        ],
        "txMessages": [
          "A1_POLICY_RESP", "PRB_PRED_RSP", "A1_POLICY_QUERY"
        ],
        "policies": [20008],
        "description": "rmr receive data port "
      },
      {
        "name": "rmr-route",
        "container": "prbpredxapp",
        "port": 4561,
        "description": "rmr route port "
      }
    ]
  },
  "rmr": {
    "protPort": "tcp:4560",
    "maxSize": 2072,
    "numWorkers": 1,
    "txMessages": [
      "PRB_PRED_RSP",
      "A1_POLICY_RESP",
      "A1_POLICY_QUERY"
    ],
    "rxMessages": [
      "PRB_PRED_REQ",
      "A1_POLICY_REQ"
    ],
    "policies": [20008]
  }
}
```

Container Info

Services port

RMR Messages

Policy Info

alloc-xAppDescriptor

```
{
  "xapp_name": "alloc",
  "version": "0.0.2",
  "containers": [
    {
      "name": "alloc",
      "image": {
        "registry": "nexus3.o-ran-sc.org:10002",
        "name": "o-ran-sc/ric-app-alloc",
        "tag": "0.0.2"
      }
    }
  ],
  "messaging": {
    "ports": [
      {
        "name": "http",
        "container": "alloc",
        "port": 10005,
        "description": "http service"
      },
      {
        "name": "rmr-data",
        "container": "alloc",
        "port": 4560,
        "txMessages": ["PRB_PRED_REQ", "RIC_HEALTH_CHECK_RESP"],
        "rxMessages": ["PRB_PRED_RSP", "SUBSCRIPTION_REQ", "RIC_HEALTH_CHECK_REQ"],
        "policies": [],
        "description": "rmr receive data port for alloc"
      },
      {
        "name": "rmr-route",
        "container": "alloc",
        "port": 4561,
        "description": "rmr route port for alloc"
      }
    ]
  },
  "rmr": {
    "protPort": "tcp:4560",
    "maxSize": 2072,
    "numWorkers": 1,
    "rxMessages": ["PRB_PRED_RESP"],
    "txMessages": ["PRB_PRED_REQ"],
    "policies": []
  },
  "controls": {
    "fileStorage": false
  },
  "db": {
    "waitForSdl": false
  }
}
```

Metadata

Internal
configuration of
xApp

Interactions With RIC Components

As part of this POC, direct/in-direct interactions with below mentioned RIC Platform Components was explored

A1-Mediator

This component listens for policy type and policy instance requests sent via HTTP (the "northbound" interface) and publishes those requests to running xApps via RMR messages (the "southbound" interface).

E2 manager

The E2 manager controls E2 connection establishment and provides REST APIs to manage these connections.

E2 Term

The E2 termination component establishes E2 SCTP connections and routes messages received/sent over E2 to/from RMR.

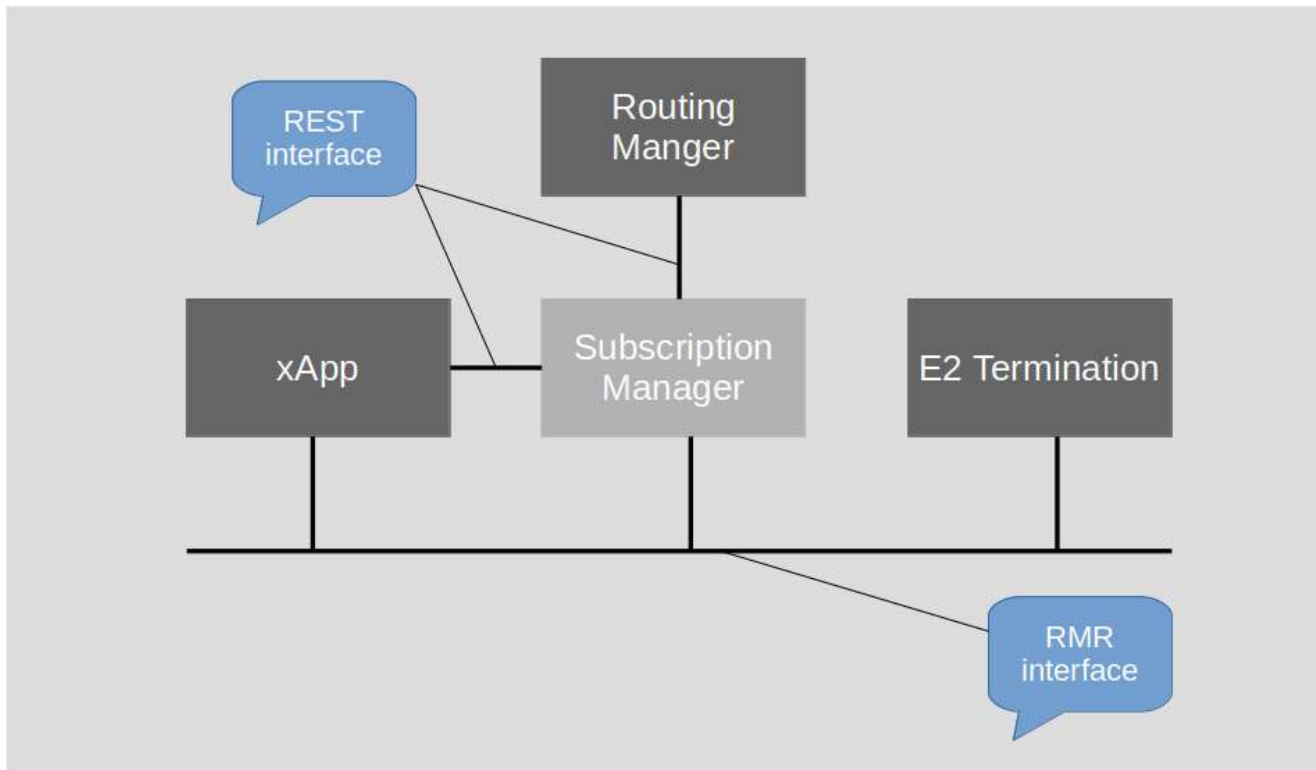
Subscription Manager

Subscription Manager is responsible for managing E2 subscriptions from xApps to the E2 Node (eNodeB or gNodeB). xApp can make subscriptions to E2 Node through Subscription Manager. xApp can subscribe REPORT, INSERT, CONTROL and POLICY type services from E2 Node.

Routing Manager

Routing Manager is responsible for distributing routing policies among the other platform components and xApps.

Messaging Interface



xApp can make subscriptions to E2 Node through Subscription Manager

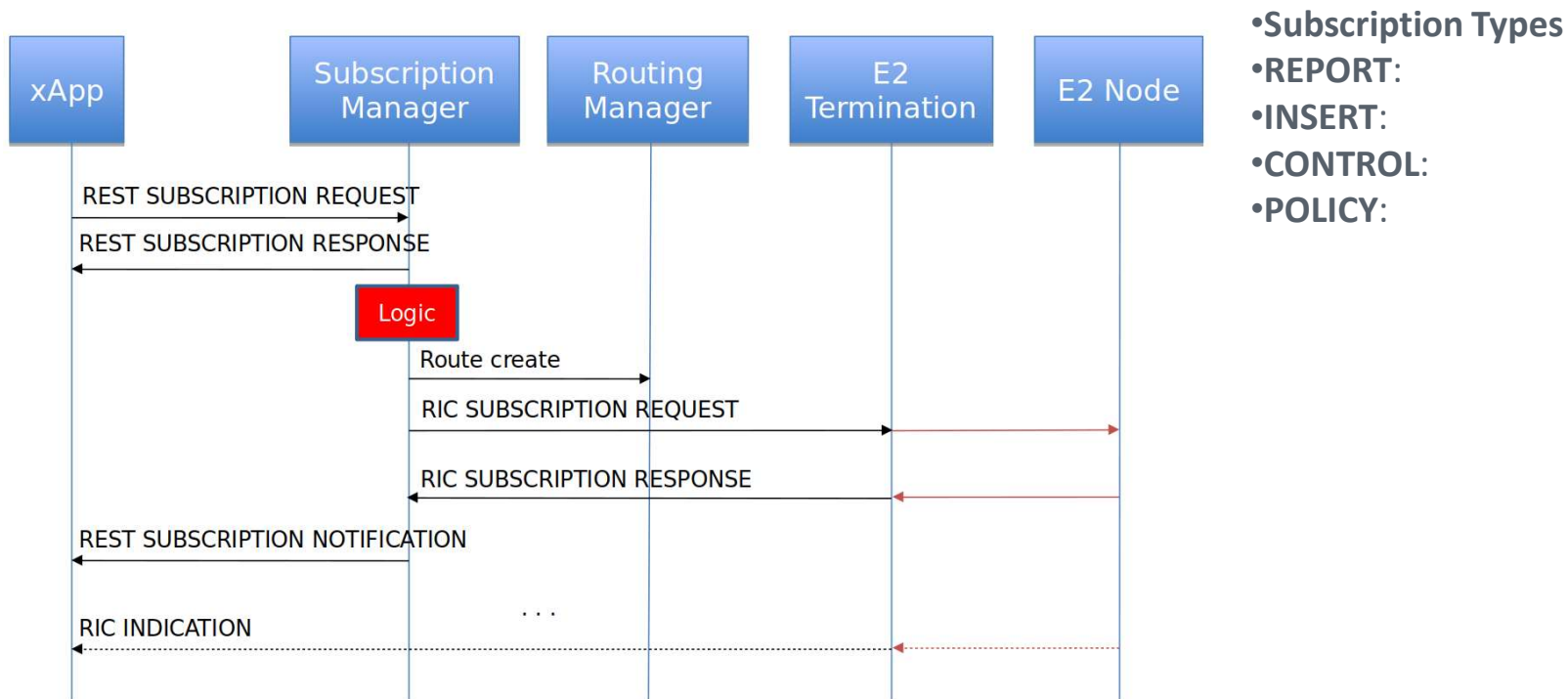
xApp can subscribe REPORT, INSERT, CONTROL and POLICY type services from E2 Node

Subscribed messages from E2 Node are transported to RIC inside RIC Indication message. RIC Indication message is transported to xApp inside RMR message in Payload field of the RMR message.

Subscription Manager allocates unique E2 instance id for every E2 subscription during subscription procedure.

Subscribed messages are routed to xApps based on InstanceId in E2 Indication message.

xApp- E2 Subscription Flow



Screen shots from demo Setup

1. RIC Platform Snapshot

NAME	READY	STATUS	RESTARTS	AGE
deployment-ricplt-almediator-b4576889d-dqs2b	1/1	Running	60	41d
deployment-ricplt-alarmanager-f59846448-76ts1	1/1	Running	36	41d
deployment-ricplt-appmgr-7cfbff4f7d-8gkmh	1/1	Running	36	41d
deployment-ricplt-e2mgr-556748b66f-9tgpv	1/1	Running	6	6d2h
deployment-ricplt-e2term-alpha-7dbd577c8d-dhcb4	1/1	Running	31	24d
deployment-ricplt-jaegeradapter-76ddbf9c9-r464v	1/1	Running	41	41d
deployment-ricplt-olmediator-f7dd5fcc8-dt9kq	1/1	Running	36	41d
deployment-ricplt-rtmgi-7455599f43-v9tfs	1/1	Running	43	41d
deployment-ricplt-submgr-6cd6775cd6-x8z74	1/1	Running	36	41d
deployment-ricplt-vespamgr-757b6cc5dc-4vtzn	1/1	Running	36	41d
deployment-ricplt-xapp-onboarder-5958856fc8-p8bjl	2/2	Running	72	41d
r4-infrastructure-kong-7995f4679b-n65qm	2/2	Running	99	41d
r4-infrastructure-prometheus-alertmanager-5798b78f48-xks4r	2/2	Running	72	41d
r4-infrastructure-prometheus-server-c8ddcdf5-55tf8	1/1	Running	36	41d
ricplt-influxdb-meta-0	0/1	Pending	0	41d
statefulset-ricplt-dbaas-server-0	1/1	Running	36	41d

2. E2 SIM Process Snapshot

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
f8dfa3da6cae	e2simul:1.0.0	"/bin/sh -c 'kpm_sim_...	5 seconds ago	Up 4 seconds

3. E2 Setup Procedure :Setup request from E2 SIM

```
<E2setupRequestIEs>
  <id>3</id>
  <criticality><reject/></criticality>
  <value>
    <GlobalE2node-ID>
      <gNB>
        <global-gNB-ID>
          <plmn-id>37 34 37</plmn-id>
          <gNB-ID>
            10110101110001100111011110001
          </gNB-ID>
        </global-gNB-ID>
      </gNB>
    </GlobalE2node-ID>
  </value>
</E2setupRequestIEs>
<E2setupRequestIEs>
  <id>10</id>
  <criticality><reject/></criticality>
  <value>
    <RANfunctions-List>
      <ProtocolIE-SingleContainer>
        <id>8</id>
        <criticality><reject/></criticality>
        <value>
          <RANfunction-Item>
            <ranFunctionID>0</ranFunctionID>
            <ranFunctionDefinition>
              30 00 00 00 05 4F 49 44 31 32 33 05 00 4B 50 4D
              20 6D 6F 6E 69 74 6F 72 01 01 60 00 01 01 07 00
              50 65 72 69 6F 64 69 63 20 72 65 70 6F 72 74 01
              05 14 01 01 1D 00 4F 2D 44 55 20 4D 65 61 73 75
              20 66 6F 72 20 74 68 65 20 35 47 43 20 63 6F 6E
              6E 65 63 74 65 64 20 64 65 70 6C 6F 79 6D 65 6E
              74 01 01 01 01 00 01 02 1D 00 4F 2D 44 55 20 4D
              65 61 73 75 72 65 6D 65 6E 74 20 43 6F 6E 74 61
              69 6E 65 72 20 66 6F 72 20 74 68 65 20 45 50 43
              20 63 6F 6E 6E 65 63 74 65 64 20 64 65 70 6C 6F
              79 6D 65 6E 74 01 01 01 01 01 01 03 1E 80 4F 2D
              43 55 2D 43 50 20 4D 65 61 73 75 72 65 6D 65 6E
              74 20 43 6F 6E 74 61 69 6E 65 72 20 66 6F 72 20
              74 68 65 20 35 47 43 20 63 6F 6E 6E 65 63 74 65
              64 20 64 65 70 6C 6F 79 6D 65 6E 74 01 01 01 01
              00 01 04 1E 80 4F 2D 43 55 2D 43 50 20 4D 65 61
              73 75 72 65 6D 65 6E 74 20 43 6F 6E 74 61 69 6E
              65 72 20 66 6F 72 20 74 68 65 20 45 50 43 20 63
              6F 6E 6E 65 63 74 65 64 20 64 65 70 6C 6F 79 6D
              65 6E 74 01 01 01 01 01 00 01 05 1E 80 4F 2D 43
              55 2D 43 50 20 4D 65 61 73 75 72 65 6D 65 6E 74
              43 6F 6E 74 61 69 6E 65 72 20 66 6F 72 20 74 68
              65 20 35 47 43 20 63 6F 6E 6E 65 63 74 65 64 20
              64 65 70 6C 6F 79 6D 65 6E 74 01 01 01 01 00 01
              06 1E 80 4F 2D 43 55 2D 55 50 20 4D 65 61 73 75
              72 65 6D 65 6E 74 20 43 6F 6E 74 61 69 6E 65 72
              20 66 6F 72 20 74 68 65 20 45 50 43 20 63 6F 6E
              6E 65 63 74 65 64 20 64 65 70 6C 6F 79 6D 65 6E
              74 01 01 01 01
            </ranFunctionDefinition>
          </ranFunctionItem>
          <ranFunctionRevision>2</ranFunctionRevision>
        </value>
      </ProtocolIE-SingleContainer>
    </RANfunctions-List>
  </value>
</E2setupRequestIEs>
</E2setupRequest>
```

E2 Mgr Setup Procedure

- ### 5. E2 Mgr changing the state as connected and sending response E2 setup response

State change to
CONNECTED

4. E2 Mgr receiving E2 Setup Request

RIC E2 SETUP RESP

```

{"crit": "INFO", "ts": "1635747069356", "id": "E2Manager", "msg": "#RanNbDataService.Instance.AddNodeIdentity", "ranName": "gmb_734 733 b5c67788 - Successfully added node identity", "mdc": {"time": "2021-11-01 06:11:09.356"}}, {"crit": "INFO", "ts": "1635747069356", "id": "E2Manager", "msg": "#E2TAssociationManager.AssociateRan - Associating RAN gmb_734 733 b5c67788 to E2T instance address: 10.110.226.182:38000", "mdc": {"time": "2021-11-01 06:11:09.356"}}, {"crit": "INFO", "ts": "1635747069356", "id": "E2Manager", "msg": "[E2 Manager -> Routing Manager] #RoutingManagerClient.sendMessage - POST url: http://service-ricltp-rtmgr-http:3800/ric/v1/handles/associate-ran-to-e2t, request body: [{\"E2Address\": \"10.110.226.182:38000\", \"RanNameList\": [{\"gmb_734 733 b5c67788\"}], \"mdc\": {\"time\": \"2021-11-01 06:11:09.356\"}}], {"crit": "INFO", "ts": "1635747069358", "id": "E2Manager", "msg": "[Routing Manager -> E2 Manager] #RoutingManagerClient.sendMessage - success, http status code: 201\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.358\"}}], {"crit": "INFO", "ts": "1635747069358", "id": "E2Manager", "msg": "#RanConnectStatusChangeManager.ChangeStatus - RAN name: gmb_734 733 b5c67788, currentStatus: UNKNOWN CONNECTION STATUS, nextStatus: CONNECTED\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.358\"}}], {"crit": "INFO", "ts": "1635747069358", "id": "E2Manager", "msg": "#RanConnectStatusChangeManager.setEvent - Connectivity Event for RAN gmb_734 733 b5c67788 is: gmb_734 733 b5c67788 CONNECTED\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.358\"}}], {"crit": "INFO", "ts": "1635747069358", "id": "E2Manager", "msg": "#RanNbDataService.UpdateNodeInfoOnConnectionStatusInversion - event: gmb_734 733 b5c67788 CONNECTED, nodeInfo: ran name: \\gmb_734 733 b5c67788\\ connection status: CONNECTED global nb id: {plmn id: \\\"373437\\\" nb id: \\\"1011010111000110011101110001\\\"} node type: GNB gnb: {ran functions: {ran function definition: \\\"300000000054f943431323005404B504D206D6F69674E672010600001101070056547296F646930272675072740105140101110004F2D4455204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D65674010101010010012D1D004F2D4455204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D65674010101010001041E804F2D4355204350204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D65674010101010001041E804F2D4355204350204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D656740101010001061804F2D4355204350204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D65674010101001061804F2D4355204350204D656173757265D6567420436F6E7416196E6572206667F220746852035474320636F6EE6563746564206465706C6F796D6567401010101\\\"} ran function revision: 2) gnb type: GNB) associated e2t_instance_address: \\\"10.110.226.182:38000\\\" setup from network: true\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.358\"}}], {"crit": "INFO", "ts": "1635747069359", "id": "E2Manager", "msg": "#RanConnectStatusChangeManager.updateNodeInfoOnConnectionStatusInversion - RAN names: gmb_734 733 b5c67788 Successfully updated. \\nb id: \\\"1011010111000110011101110001\\\"} associated e2t_instance_address: \\\"10.110.226.182:38000\\\" setup from network: true\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.359\"}}], {"crit": "INFO", "ts": "1635747069359", "id": "E2Manager", "msg": "#RanNbDataService.GetE2TInstance - E2T instance address: 10.110.226.182:38000, state: ACTIVE, associated RANs count: 0, keep Alive ts: 1635747046708612521\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.359\"}}], {"crit": "INFO", "ts": "1635747069359", "id": "E2Manager", "msg": "#RanNbDataService.SaveE2TInstance - E2T instance address: 10.110.226.182:38000, podName: e2tterm, state: ACTIVE, associated RANs count: 1, keep Alive ts: 1635747046708612521\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.359\"}}], {"crit": "INFO", "ts": "1635747069359", "id": "E2Manager", "msg": "#E2TInstancesManager.AddToRanToInstance - RAN (gmb_734 733 b5c67788) were added successfully to E2T 10.110.226.182:38000\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.359\"}}], {"crit": "INFO", "ts": "1635747069359", "id": "E2Manager", "msg": "#E2TAssociationManager.AssociateRan - successfully associated RAN gmb_734 733 b5c67788 with E2T 10.110.226.182:38000\", \"mdc\": {\"time\": \"2021-11-01 06:11:09.359\"}}], {"crit": "INFO", "ts": "1635747069360", "id": "E2Manager", "msg": "#E2SetupRequestNotificationHandler.handleSuccessfulResponse - payload: <E2AP-PDU>{successfulOutcome}<procedureCode>{criticality}<reject>{criticality}<value>{GlobalRanId}<E2setupResponse>{protocolIEs}<E2setupResponseIEs>{<id>4/<id>criticality}<reject>{criticality}<value>{GlobalRanId}<plmn-identity>{131014/<plmn-identity>{<ric-ID>1010101010011001110/<ric-ID>{GlobalRAN-ID}<value>{<E2setupResponseIEs>{<E2setupResponseIEs>{<id>9/<id>criticality}<reject>{criticality}<value>{RanFunctionsID-list}<SingleContainer>{<id>6/<id>criticality}<ignore>{criticality}<value>{RanFunctionID-Item}<RanFunctionID>{<ranFunctionID-Item>{<ranFunctionRevision>{<ranFunctionRevision>{<RanFunctionID-Item>{<value>{<ProtocolIE-SingleContainer>{<RanFunctionsID-list>{<value>{<E2setupResponseIEs>{<protocolIEs>{<E2setupResponse>{<value>{successfulOutcome}<E2AP-PDU>\"}, \"mdc\": {\"time\": \"2021-11-01 06:11:09.360\"}}}, {"crit": "INFO", "ts": "1635747069360", "id": "E2Manager", "msg": "#E2SetupRequestNotificationHandler.handleSuccessfulResponse - RAN name: gmb_734 733 b5c67788 - R5-C2 SETUP RESP message has been built successfully. Message: {<2ee2 676e2 027333543f733335f623563367373838 - 4153241502d5044553e3c7375636365f737366756c4f7574636f6d653e3c70276f636554f75726543}"}

```


E2 Setup details can be confirmed with response from below CURL command

6. E2 Setup Response at E2 SIM end

E2 SETUP RESPONSE- ricid

```

<E2setupResponse>
  <protocolIEs>
    <E2setupResponseIEs>
      <id>4</id>
      <criticality><reject></criticality>
      <value>
        <GlobalRRC-ID>
          <plmn-Identity>13 10 0</plmn-Identity>
          <rrc-ID>
            10101010110011001110
          </rrc-ID>
          </GlobalRRC-ID>
        </value>
      </E2setupResponseIEs>
    <E2setupResponseIEs>
      <id>9</id>
      <criticality><reject></criticality>
      <value>
        <RANfunctionsID-List>
          <ProtocolIE-SingleContainer>
            <id>6</id>
            <criticality><ignore></criticality>
            <value>
              <RANfunctionID-Item>
                <ranFunctionID>0</ranFunctionID>
                <ranFunctionRevision>2</ranFunctionRevision>
              </RANfunctionID-Item>
            </value>
          </ProtocolIE-SingleContainer>
        </RANfunctionsID-List>
      </value>
    </E2setupResponseIEs>
  </protocolIEs>
</E2setupResponse>

```

```
7. curl -v --location --request GET "http://<E2Mgr ip>/v1/e2t/list" --header 'Content-Type: application/json'
```

Result displays the e2Term instance and RAN name associated

```
root@instance-2:~# curl -v --location --request GET "http://10.244.0.128:38000/v1/e2t/list" --header 'Content-Type: application/json'
Note: Unnecessary use of -X or --request, GET is already inferred.
* Trying 10.244.0.128...
* TCP_NODELAY set
* Connected to 10.244.0.128 (10.244.0.128) port 38000 (#0)
> GET /v1/e2t/list HTTP/1.1
> Host: 10.244.0.128:3800
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json

< HTTP/1.1 200 OK
< Content-Type: application/json
< Date: Mon, 01 Nov 2021 06:12:57 GMT
< Content-Length: 75

Connection #0 to host 10.244.0.128 left intact
{"e2tAddress":"10.110.226.182:38000","ranNames":["qmb 734 733 b5c67788"]}|root@instance-2:~#
```

8. Fetch the gNB details associated on trigger of GET request from E2mgr

```
curl -v --location --request GET "http://<E2Mgr
ip>/v1/nodeb/gnb_734_733_b5c67788" --header 'Content-Type:
application/json'
```

[illegible]

A1 Policy and Policy Instances

•9 A1- Policy and Policy Instances are created by executing following curl commands ..Indicating an emergency and defines the model to be used here.

•A1-Policy

- curl -X PUT --header "Content-Type: application/json" -d @create1.json http://<a1med_ip>: <a1med_port>:/a1-p/policytypes/20008

A1 Policy Instance creation

- curl -X PUT --header "Content-Type: application/json" --data '{"modelVersion": "1.0.0", "modelname": "prb_pred_model.pkl", "modelstoreUrl": "http:// <modelstore IP >/model_store"}' http://<a1med_ip>: <a1med_port>/a1-p/policytypes/20008/policies/tsapolicy145

Below snapshot shows the policy instance is created with the values supplied in the curl command

```
root@instance-2:~# curl --header "Content-Type: application/json" http://10.244.0.87:10000/a1-p/policytypes/20008/policies/tsapolicy145
{"modelVersion": "1.0.0",
 "modelname": "prb_pred_model.pkl",
 "modelstoreUrl": "http://34.72.49.222:10001/model_store"
}
```

Policy Schema Used

```
{
  "name": "tsapolicy",
  "description": "tsa parameters",
  "policy_type_id": 20008,
  "create_schema": {
    "$schema": "http://json-schema.org/draft-07/schema#",
    "type": "object",
    "properties": {
      "modelname": {
        "type": "string"
      },
      "modelVersion": {
        "type": "string",
        "default": 0.0
      },
      "modelstoreUrl": {
        "type": "string"
      }
    },
    "additionalProperties": false
  }
}
```

xApp Screenshots & Model store

10. Alloc xApp is deployed

```
root@instance-2:~# kubectl get pods -n ricxapp
```

NAME	READY	STATUS	RESTARTS	AGE
ricxapp-alloc-b9f994b84-x7zxr	1/1	Running	0	8h
ricxapp-prbpredxapp-66bfd5bc55-jhrmn	1/1	Running	0	8h

11. Emulated Acumos Model Store listening on 10001 port, receives REST REQ for fetching model. Acumos functionality was emulated considering HW requirements in bringing up Acumos.

Model Fetch Request

```
(my_env) root@instance-2:~/customxapp python modelstore.py
```

```
db      .g8""bgd `7MME' `7MF'`7MMM. ,MMF' .g8""8g. .M""bgd
;MM:    .dp'   `M   MM   M   NMMb dPMM .dp'   `YM. ,MI   "Y
,V`MM.  dm'    `   MM   M   M YM ,M MM dm'   `MM  NMMb.
,M `MM   MM     MM   M   Mb M' MM MM   MM   `YMMNg.
AbmmmmMA MM.    MM   M   YM.P' MM MM.   ,MP.   `MM
A"      VML  `Mb.   ,`  YM.   ,M   `YM' MM  `Mb.  ,dp' Mb   dm
.AMA.   .AMMA.  `Lmmmd'  `bmmmmmd" .JML.  ``.JMML.  ``Emmd"  P"YEmmd"
```

```
Starting model store 2021-10-31 09:58:03.722292
* Serving Flask app 'modelstore' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://10.128.0.4:10001/ (Press CTRL+C to quit)
Received req2021-10-31 10:02:05.442191
34.72.49.222 - - [31/Oct/2021 10:02:05] "GET /model_store/1.0.0/prb_pred_model.pkl HTTP/1.1" 200 -
[FC:AT:E8:27:04:DF]
```

12. Below Snapshot shows REST Req being sent to model store with the details received from A1 policy request. Model store URL being constructed based on the A1 policy request. Model is successfully downloaded to prbpred container.



```

31-Oct-21 10:02:04 - pred xapp created@@@
31-Oct-21 10:02:04 - A1PolicyInterface
31-Oct-21 10:02:04 - send a1 policy query (A1_POLICY_QUERY)= {"policy_type_id": "20008"}
{"ts": 1635674525435, "crit": "DEBUG", "id": "ricxappframe.xapp_frame", "mdc": {}, "msg": "run: invoking msg handle
on type 20010"}
31-Oct-21 10:02:05 - request handler.resp handler:: Handler processing A1_POLICY_REQ request
31-Oct-21 10:02:05 - request handler.resp handler:: Handler verified policy
type id: '20008', 'policy_instance_id': 'tsapolicy145', 'payload': {'modelVersion': '1.0.0', 'modelName': 'prb
pred_model.pkl', 'modelstoreUrl': 'http://34.72.49.222:10001/model_store/1.0.0/'}, 'handler_id': 'prbpred', 'status': 'ok'}
31-Oct-21 10:02:05 - A1PolicyInterface::request handler received payload {'modelVersion': '1.0.0', 'modelName': 'prb
pred_model.pkl', 'modelstoreUrl': 'http://34.72.49.222:10001/model_store/1.0.0/'}, 'handler_id': 'prbpred', 'status': 'ok'}
31-Oct-21 10:02:05 - store model info: Fetch model from model store http://34.72.49.222:10001/model_store/1.0.0/
31-Oct-21 10:02:05 - pull_model::Sent Download request to model store http://34.72.49.222:10001/model_store/1.0.0/
31-Oct-21 10:02:05 - pull_model::Successfully Downloaded model to ./prbpred/prb_pred_model.pkl
1635674525 1/RMR [INFO] sends: ts=1635674525 src=service-ricxapp-prbpredxapp-rmr.ricxapp:4560 target=service-ricxap
p-alloc-rmr:4560 open=0 succ=0 fail=0 (hard=0 soft=0)
1635674525 1/RMR [INFO] sends: ts=1635674525 src=service-ricxapp-prbpredxapp-rmr.ricxapp:4560 target=service-ricpl
-almediator-rmr.ricplt:4562 open=1 succ=1 fail=0 (hard=0 soft=0)
/opt/conda/lib/python3.7/site-packages/sklearn/base.py:333: UserWarning: Trying to unpickle estimator GaussianProce
ssRegressor from version 0.24.1 when using version 1.0.1. This might lead to breaking code or invalid results. Use
at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning:
31-Oct-21 10:02:05 - store_model info::Saved Model info
31-Oct-21 10:02:05 - request handler.resp handler:: A1_POLICY_RESP Response sent {"policy_type_id": "20008", "poli
cy_instance_id": "tsapolicy145", "payload": {"modelVersion": "1.0.0", "modelName": "prb_pred_model.pkl", "modelstor
eUrl": "http://34.72.49.222:10001/model_store/1.0.0/"}, "handler_id": "prbpred", "status": "ok"}
{"ts": 1635674553708, "crit": "DEBUG", "id": "ricxappframe.xapp_frame", "mdc": {}, "msg": "run: invoking msg handle
on type 20003"}
31-Oct-21 10:02:33 - predict handler received payload b'1'
31-Oct-21 10:02:33 - Predictor::predict()
/opt/conda/lib/python3.7/site-packages/sklearn/base.py:333: UserWarning: Trying to unpickle estimator GaussianProce
ssRegressor from version 0.24.1 when using version 1.0.1. This might lead to breaking code or invalid results. Use
at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning:
31-Oct-21 10:02:33 - Predicted value for Slice 1&2 : b'{"prediction": [71.7, 70.6]}'
31-Oct-21 10:02:33 - Sending message to alloc xApp : b'{"prediction": [71.7, 70.6]}'
31-Oct-21 10:02:33 - predict handler: sent message successfully

```

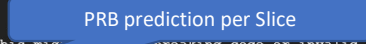



13. On Timer expiry Alloc xApp sent PRB_PRED_REQ to prbpred xApp and prbpred xApp receives PRB_PRED_REQ, performs prediction of the PRB's and sends message to Alloc XAPP

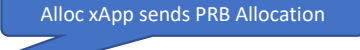
```

31-Oct-21 10:03:33 - predict handler received payload b'1'
31-Oct-21 10:03:33 - Predictor::predict()
/opt/conda/lib/python3.7/site-packages/sklearn/base.py:333: UserWarning: Trying to unpickle estimator GaussianProce
ssRegressor from version 0.24.1 when using version 1.0.1. This might lead to breaking code or invalid results. Use
at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning:
31-Oct-21 10:03:33 - Predictor::predict()
/opt/conda/lib/python3.7/site-packages/sklearn/base.py:333: UserWarning: Trying to unpickle estimator GaussianProce
ssRegressor from version 0.24.1 when using version 1.0.1. This might lead to breaking code or invalid results. Use
at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
UserWarning:
31-Oct-21 10:03:33 - Predicted value for Slice 1&2 : b'{"prediction": [71.7, 70.6]}'
31-Oct-21 10:03:33 - Sending message to alloc xApp : b'{"prediction": [71.7, 70.6]}'
31-Oct-21 10:03:33 - predict handler: sent message successfully
1635674618 1/RMR [INFO] sends: ts=1635674618 src=service-ricxapp-prbpredxapp-rmr.ricxapp:4560 target=service-ricxap
p-alloc-rmr:4560 open=0 succ=0 fail=0 (hard=0 soft=0)
1635674618 1/RMR [INFO] sends: ts=1635674618 src=service-ricxapp-prbpredxapp-rmr.ricxapp:4560 target=service-ricpl
-almediator-rmr.ricplt:4562 open=1 succ=2 fail=0 (hard=0 soft=0)
1635674618 1/RMR [INFO] sends: ts=1635674618 src=service-ricxapp-prbpredxapp-rmr.ricxapp:4560 target=service-ricxap
p-alloc-rmr.ricxapp:4560 open=1 succ=1 fail=0 (hard=0 soft=0)

```



14. Alloc xApp receives PRB_PRED_RSP from prbpred xApp, computes the PRB to be allocated and sends control message to E2



```

31-Oct-21 10:03:33 - [msg to pred]
31-Oct-21 10:03:33 - [INFO] Message to pred : message sent successfully
31-Oct-21 10:03:33 - [INFO] Received acknowledgement from pred (PRB_PRED_RSP): {'payload': b'{"prediction": [71.7, 70.6]}'
, 'payload length': 28, 'message type': 20002, 'subscription id': -1, 'transaction id': b'aaec35103a311ec832
headad170f84a', 'message state': 0, 'message status': 'RMR_OK', 'payload max size': 3136, 'meid': b'', 'message sou
rce': 'service-ricxapp-prbpredxapp-rmr.ricxapp:4560', 'errno': 0}
31-Oct-21 10:03:33 - Estimated PRB usage of Slice 1:25
31-Oct-21 10:03:33 - Estimated PRB usage of Slice 2:25
31-Oct-21 10:03:33 - PRB allocated to Emergency SLice :50

```


Proposed Future Work

- Build a multivariant timeseries model with monitored data and arrive at proper inference.
- It is recommended that gNode/E2 interface has reserved resources for Emergency situations. Additionally, based on the situation resource reallocation from lower QOS based services should be explored.
- Develop a user friendly webapp to onboard xApp's & trigger policy towards near-RT RIC and support visualisations
- Extend the solution to self learning Closed Loops with following capabilities :-
 - Continuously perform Collection, Analytics, Decision and Actuation
 - Detect model performance and trigger a switch-over to another better performing model
- Analyse and trigger different set of data/measurements for data analysis
- Points for future study from FGAN-O-013 :=
 - 1) How ML pipelines can be synchronized/managed across the edge and emergency responder devices ?
 - 2) The split of inference tasks/model functionalities between edge and emergency responder devices

Open Issues

- a. ricplt-influxdb-meta-0 pod is in pending state in RIC platform. Tried all the suggestion as mentioned in RIC wiki but couldn't succeed
- b. Not receiving Subscription response from Subscription Manager
- c. Behaviour on A1 mediator sending A1 POLICY REQ when policy instance is CREATED/UPDATED to xApps is suppressed in Dawn release

Special Thanks

Thoralf Czichy - Nokia

Abukar Mohamed - Nokia

REFERENCES

1. A. Dandekar, J.Schulz-Zander, H.Wissing, Fraunhofer HHI, "Use case and requirements for orchestration of AI/ML based closed loops to enable autonomous networks", Fraunhofer HHI, Apr. 2021.
2. [Build-a-thon FG AN] ITU-T FG AN-I-146 "Proposal for a "Build-a-thon" for ITU AI/ML in 5G Challenge (second edition, 2021), aligned with FGAN WG3" <https://extranet.itu.int/sites/itu-t/focusgroups/an/input/FGAN-I-114-R1.docx>
3. [Build-a-thon Challenge] ITU-T AI/ML in 5G Challenge problem statement "ITU-ML5G-PS-014: Build-a-thon (PoC) Network resource allocation for emergency management based on closed loop analysis" <https://challenge.aiforgood.itu.int/match/matchitem/45>
4. https://github.com/ITU-build-a-thon/challenge-resources/blob/main/intro_tutorial.pdf
5. FGAN-153 "Team AUTOMATO" https://extranet.itu.int/sites/itu-t/focusgroups/an/_layouts/15/WopiFrame.aspx?sourcedoc=%7B85757552-DFBE-479A-A816-003AE91C2B22%7D&file=FGAN-I-155.docx&action=default
6. Pre-trained model and repository https://github.com/krcmehmet/ITUChallenge_BuildaThon_Activity4
7. Near Realtime RIC <https://wiki.o-ran-sc.org/display/GS/Near+Realtime+RIC+Installation>
8. <https://wiki.o-ran-sc.org/display/ORANSdk/App+Writing+Guide>
9. <https://github.com/o-ran-sc/code-repository>
10. <https://lists.o-ran-sc.org/g/main/topics>
11. https://docbox.etsi.org/ISG/ZSM/Open/Drafts/009-3ed111_Cla_AdvTop/ZSM-009-3_Cla_AdvTopv010