O-RAN with Machine Learning in ns-3

WNS3 23 – June 2023

Wesley Garey[1], Tanguy Ropitault[2,3], Richard Rouil[1], Evan Black[1], Weichao Gao[2,4]

1 National Institute of Standards and Technology Gaithersburg, Maryland, USA

2 Associate, National Institute of Standards and Technology, Maryland, USA

3 Prometheus Computing LLC Bethesda, Maryland, USA

4 Dakota Consulting, Inc. Silver Spring, Maryland, USA

Disclaimer

Certain equipment, instruments, software, or materials are identified in this presentation in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement of any product or service by NIST, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

Context

Traditional Radio Access Network (RAN)

Proprietary and highly integrated system
⇒ limited flexibility, scalability, and vendor lock-in

Open-RAN (O-RAN)

Evolve the RAN to be *open*, *intelligent*, *interoperable* and *autonomous* *thanks to* ***disaggregation***, ***virtualization*** *and* ***open interfaces***

Usage of Artificial Intelligence/Machine Learning (AI/ML): Autonomous control using real-time data and statistics

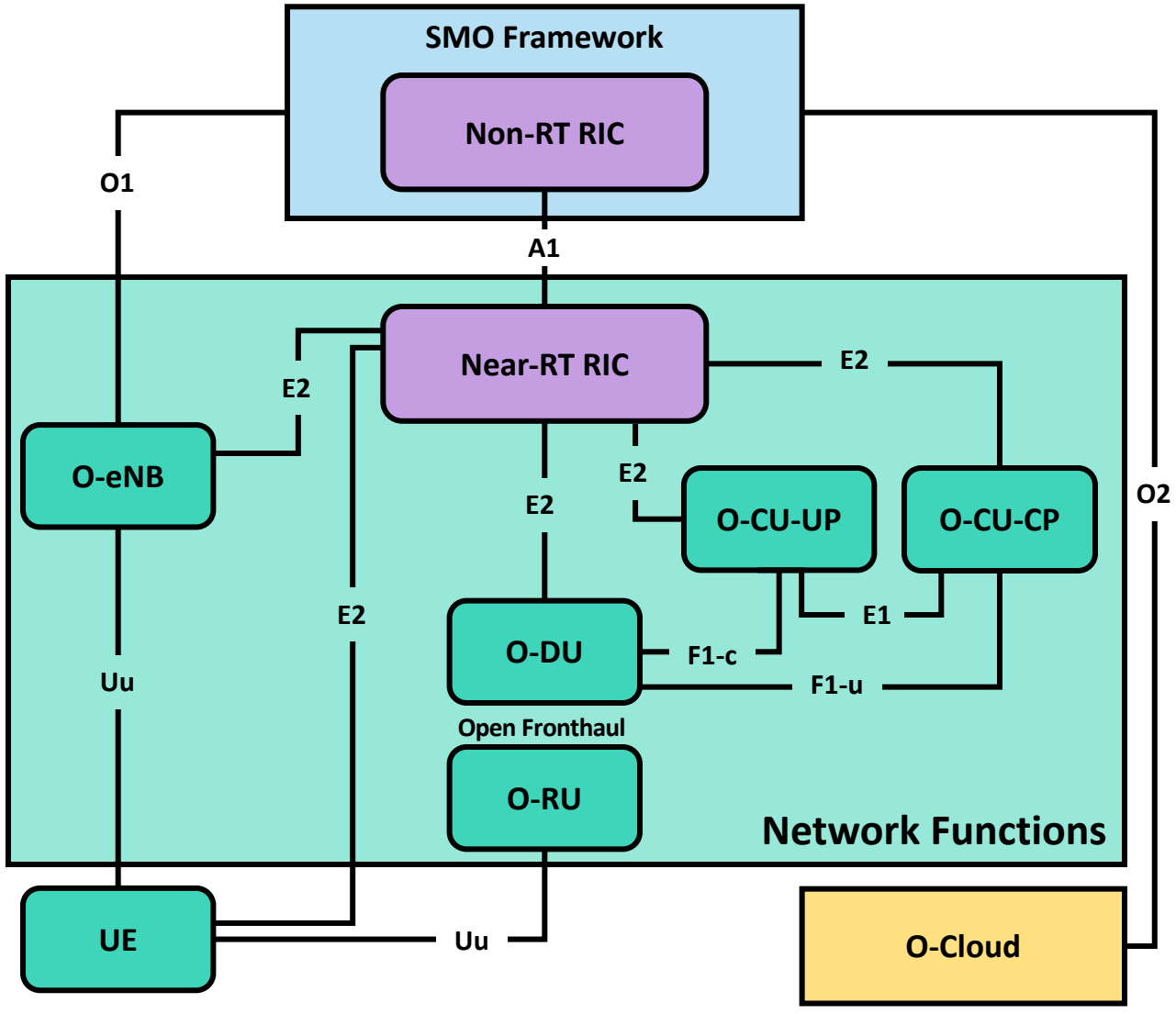
Evaluation: Complex

Understanding the interactions and the performance is a challenging task
⇒ **System-level simulations**

ns3-oran: https://github.com/usnistgov/ns3-oran

- Implementation of O-RAN architecture with a focus on ease of installation and extensibility. It includes:
  - Near-Real Time (RT) RAN Intelligent Controller (RIC) and E2 interface O-RAN components
  - Fast and efficient ML

About O-RAN
Architecture



Service Management and Orchestration (SMO) Framework

- Houses the non-Real Time (RT) Real-Time RIC: manages and orchestrates the top-level control of the network functions
- Ensures efficient operation through policy management and facilitates the deployment of AI/ML models

Network Functions

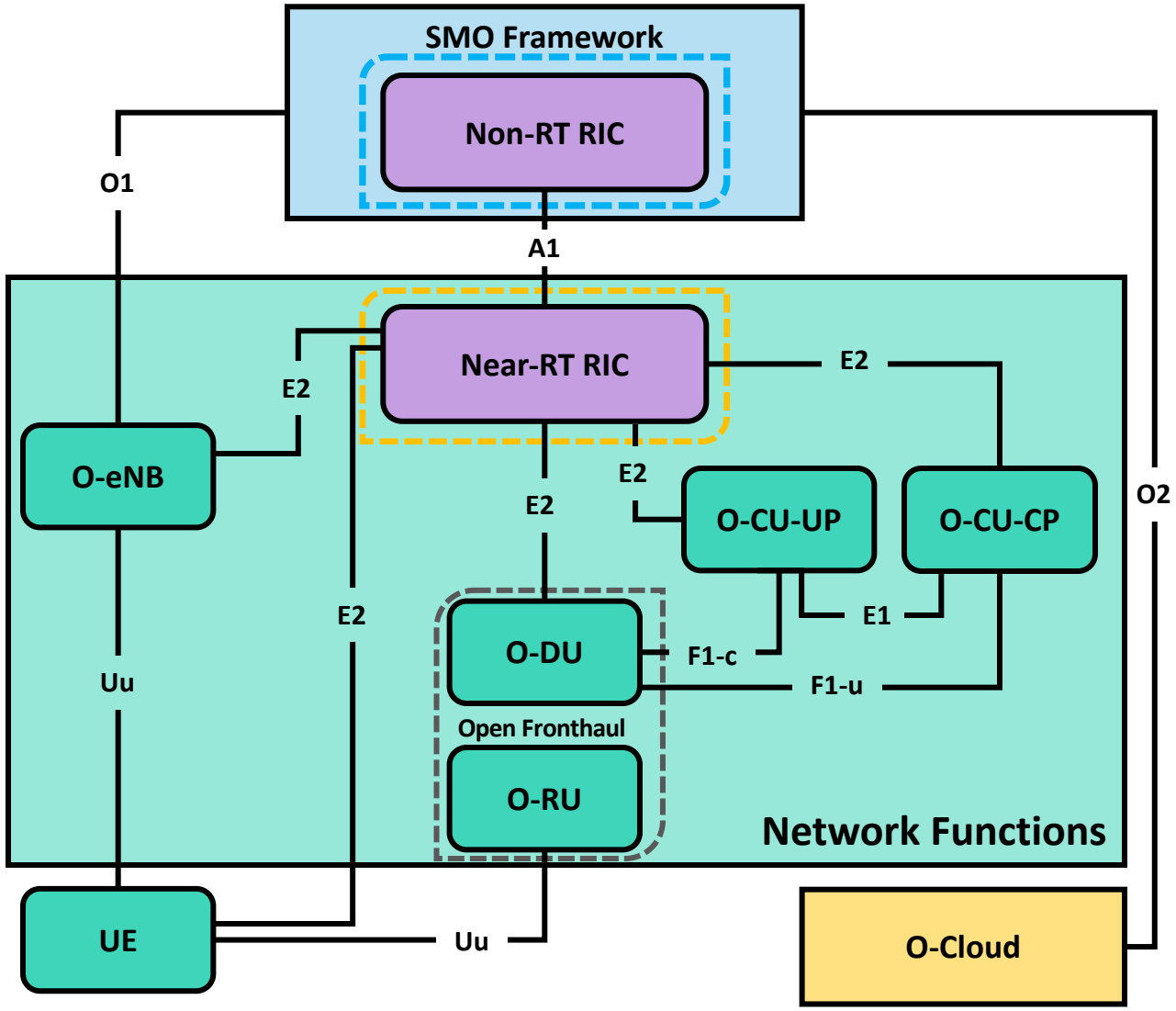
- Intermediary between the Non-RT RIC and the Radio Unit (RU) component
- Houses the Near-RT RIC:
  - Goal: optimizing network performance by dynamically adjusting network parameters based on real-time conditions and policies.
  - Connected to multiple components through the E2 interface to request O-RAN actions and/or provide information about network status and performance

O-Cloud

- Pool of physical servers hosting the Network Functions.
- Allows to host virtualized components
- Cloud-based: Dynamic scalability and agile deployment of services => Flexible RAN

About O-RAN

Architecture: Control Loop



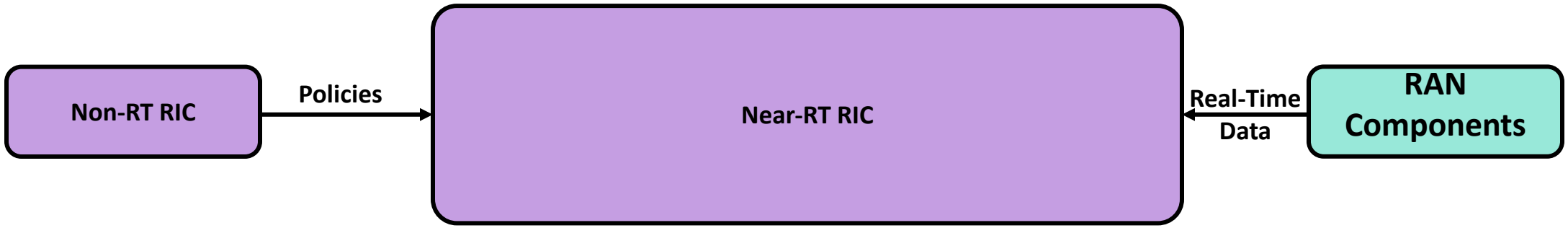
Non-RT RIC Control Loop > 1 s:
Managing less time-sensitive aspects of the radio access network, e.g., Network-wide optimization, longer-term radio resource management, policy creation and updates, or intelligent load balancing

Near RT-RIC >= 10 ms and < 1 s:
Fine-grained control and optimization of the RAN, e.g., interference management, handovers, load balancing, or spectrum sharing

O-DU and O-RU control loop*: < 1 ms:
Device management and optimization, e.g, scheduling, beam management

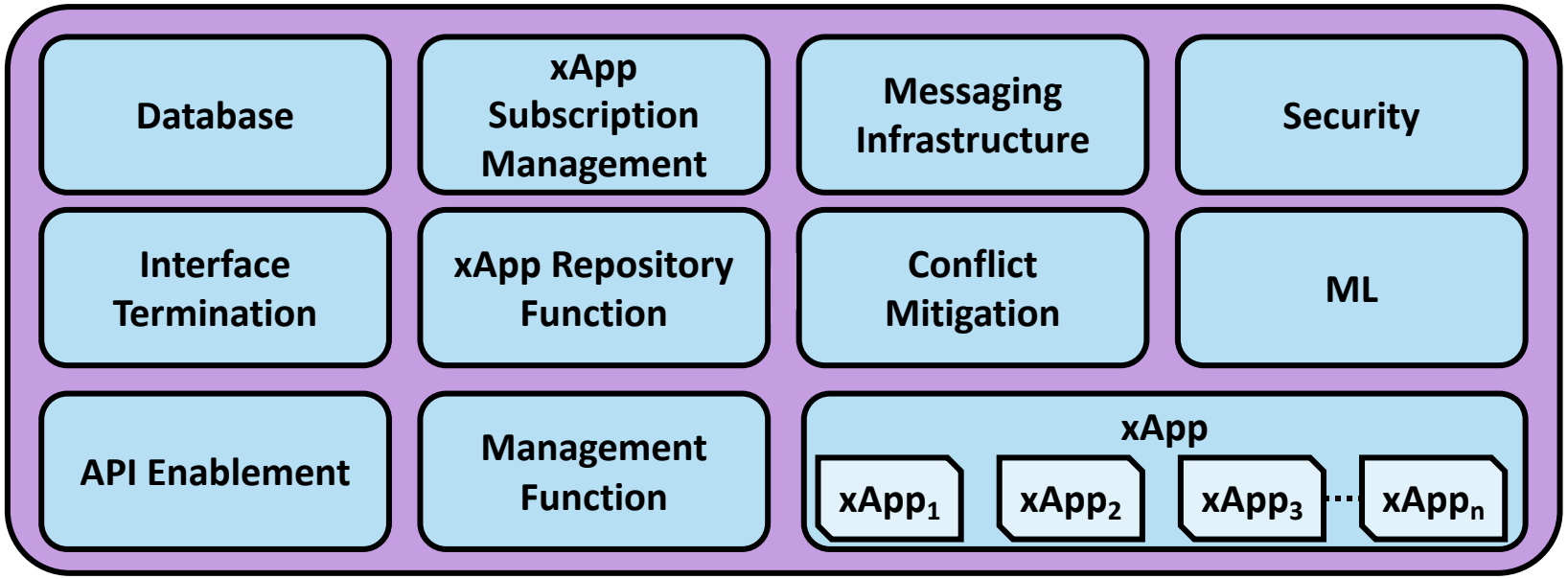
* Currently discussed

Focus on the near-RT RIC



- Collects real-time data from RAN components, performs analytics, and dynamically adjusts network parameters to align with the defined policies and guidelines, and this to improve performance
- The Non-RT RIC provides the policies

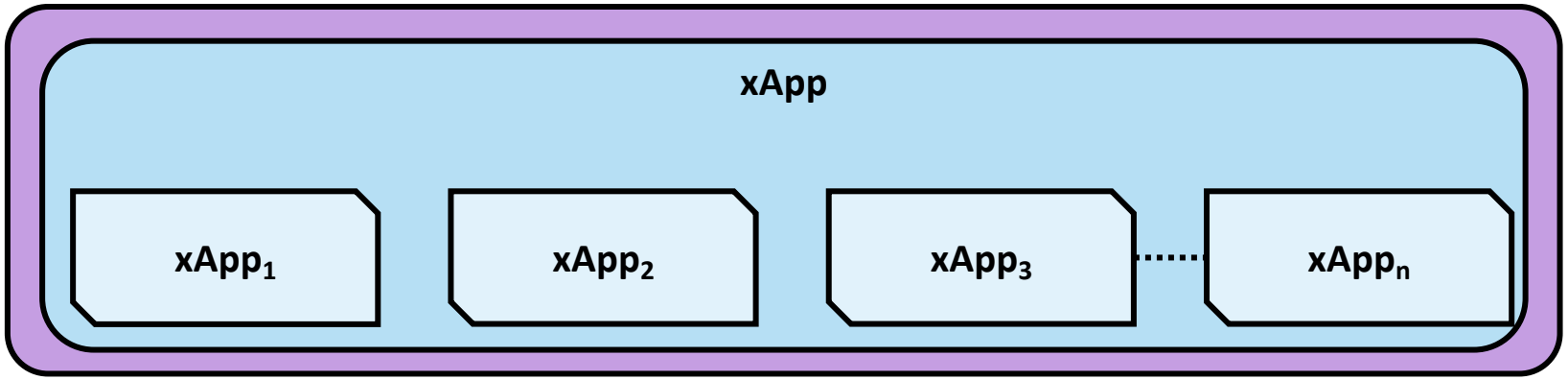
Focus on the near-RT RIC



- Collects real-time data from RAN components, performs analytics, and dynamically adjusts network parameters to align with the defined policies and guidelines, and this to improve performance
- The Non-RT RIC provides the policies
- **Multiple components**

Focus on the near-RT RIC

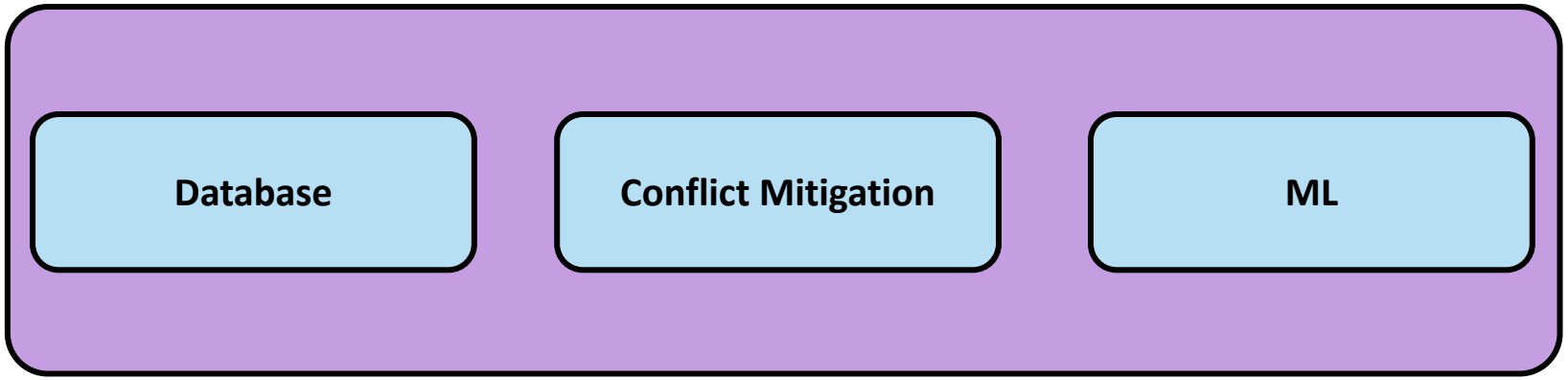
xApp



- xApp: Software applications running on the near-RT RIC designed to address specific use cases within the RAN domain (may be provided by any third party)
- xApp
  - Digests and realizes the policies pushed to the Near-RT RIC by the Non-RT RIC
  - Responsible for analyzing data stored at the Near-RT RIC to make useful decisions
- xApp examples
  - Spectrum sharing
  - Cell selection

Focus on the near-RT RIC

Database, Conflict Mitigation, and ML



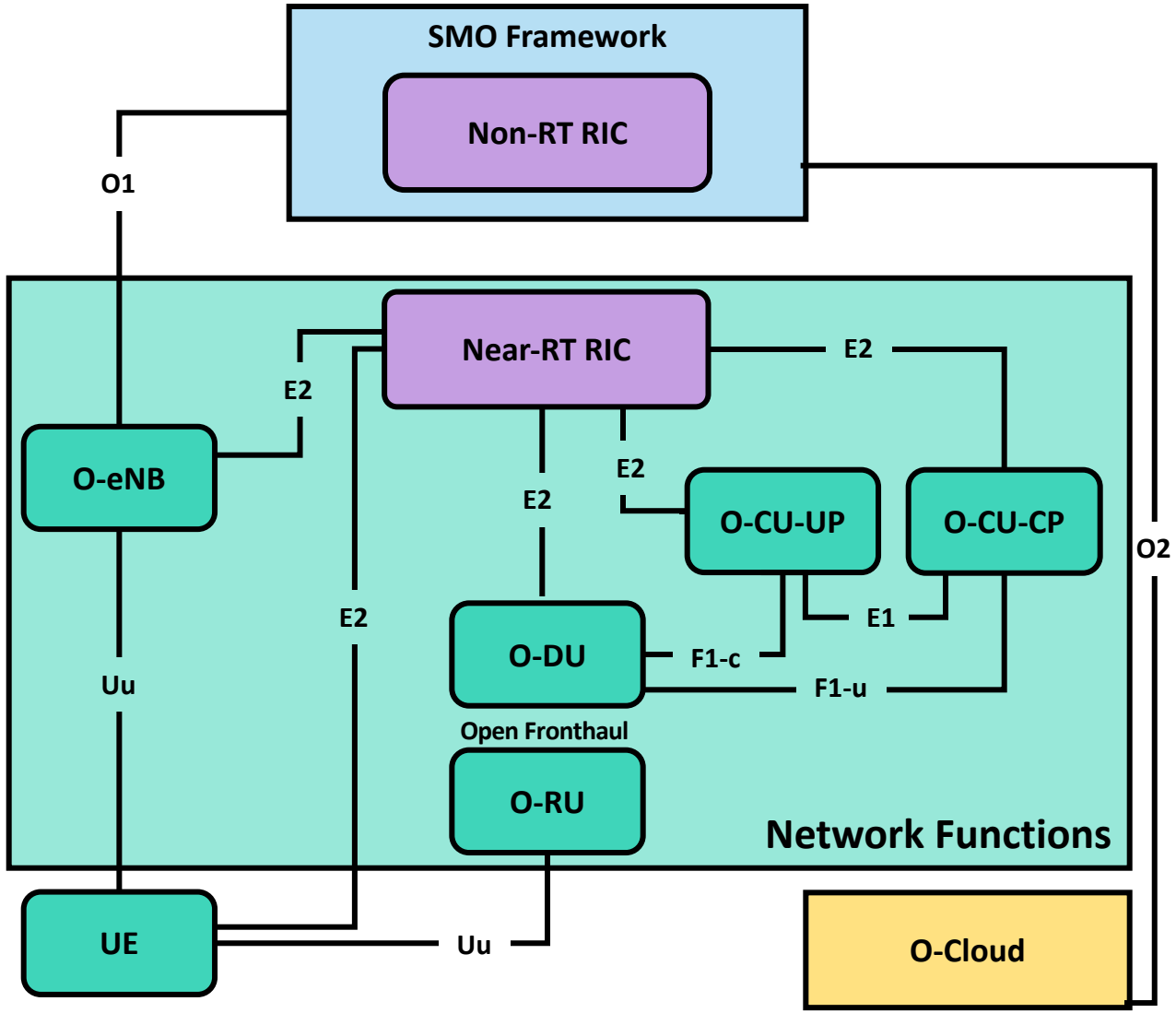
- **Database**: Where all of the data is stored that may need to be accessed by the Near-RT RIC and its components, such as the xApps

- **Conflict Mitigation**
  - Filter that processes all of the actions that xApps plan to carry out in the RAN to handle direct, indirect, and implicit conflicts
  - Needed as it is possible as different xApps goals might result in actions that affect each other performance
  - Responsibility of the CM to both discover these conflicts and form resolutions

- **ML**: The ML module provides ML support so that xApps can train models and use them to make inferences

Focus on the near-RT RIC

The E2 Interface



Serves as the logical link between E2 nodes and the Near-RT RIC (E2 node is any RAN component with an E2 interface between itself and the Near-RT RIC)

Two protocols:

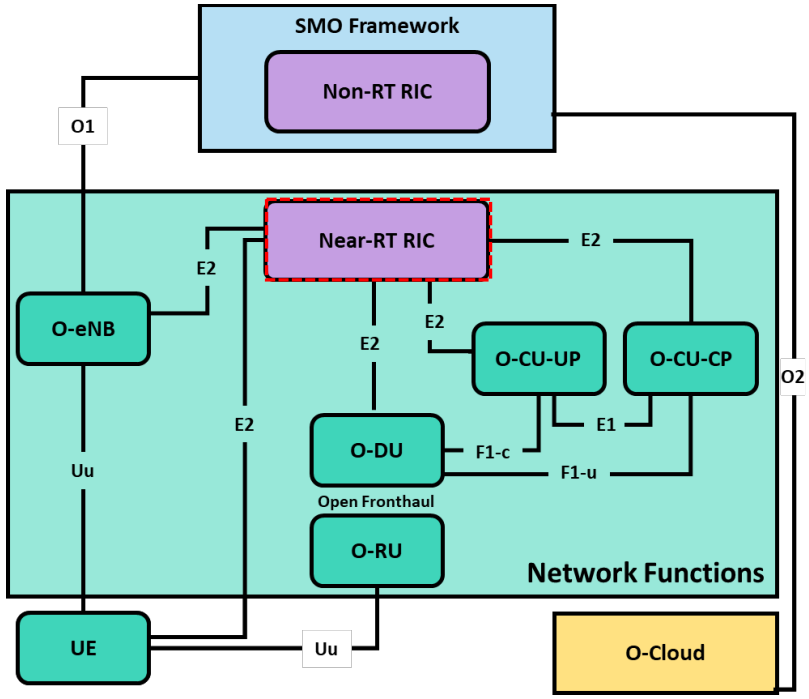
- *E2 Application Protocol (E2AP)*: helps E2 nodes establish a link with Near-RT RIC
- *E2 Service Model (E2SM)*: enables the E2 node to **report** network-related information to the Near-RT RIC and receive **control** and **policy** messages

Through E2AP and E2SM, E2 interface enables intelligent communication and policy enforcement in the network, enhancing the effectiveness of Near-RT RIC

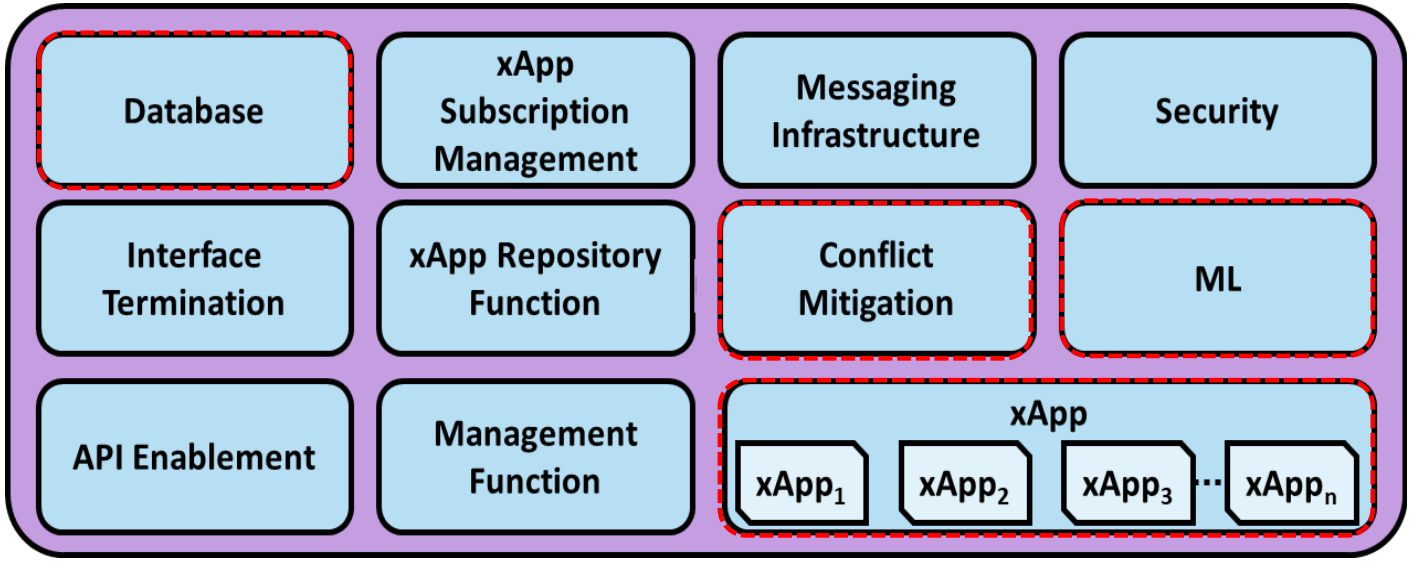
The O-RAN ns-3 module

Available at: https://github.com/usnistgov/ns3-oran

- Developed with **extensibility** and **ease of use** as primary considerations



Implement abstractions of the **Near-RT RIC** and virtual **E2 interface***



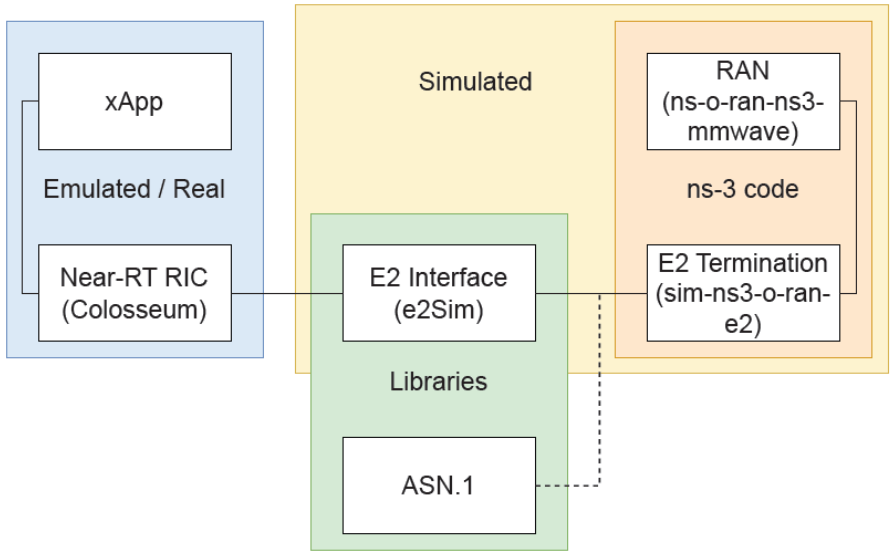
Implement the **xApp**, **Database**, **ML** and **Conflict Mitigation** module

NIST COMMUNICATIONS TECHNOLOGY LABORATORY

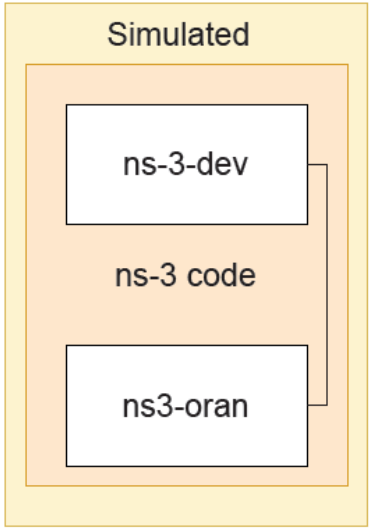
*No SMO or O-Cloud as policy-level design and RAN virtualization are out of the scope of our model.

The O-RAN ns-3 module

How does it compare to other existing ns-3 O-RAN Tools



- OpenRAN Gym [1]: **Very comprehensive O-RAN implementation**
- Requires the use of Docker containers **and/or** a true implementation of a Near-RT RIC, the E2 interface, and remaining O-RAN components, in combination with a specialized version of ns-3
- Attractive for vendors and operators who intend to implement and later deploy their solution

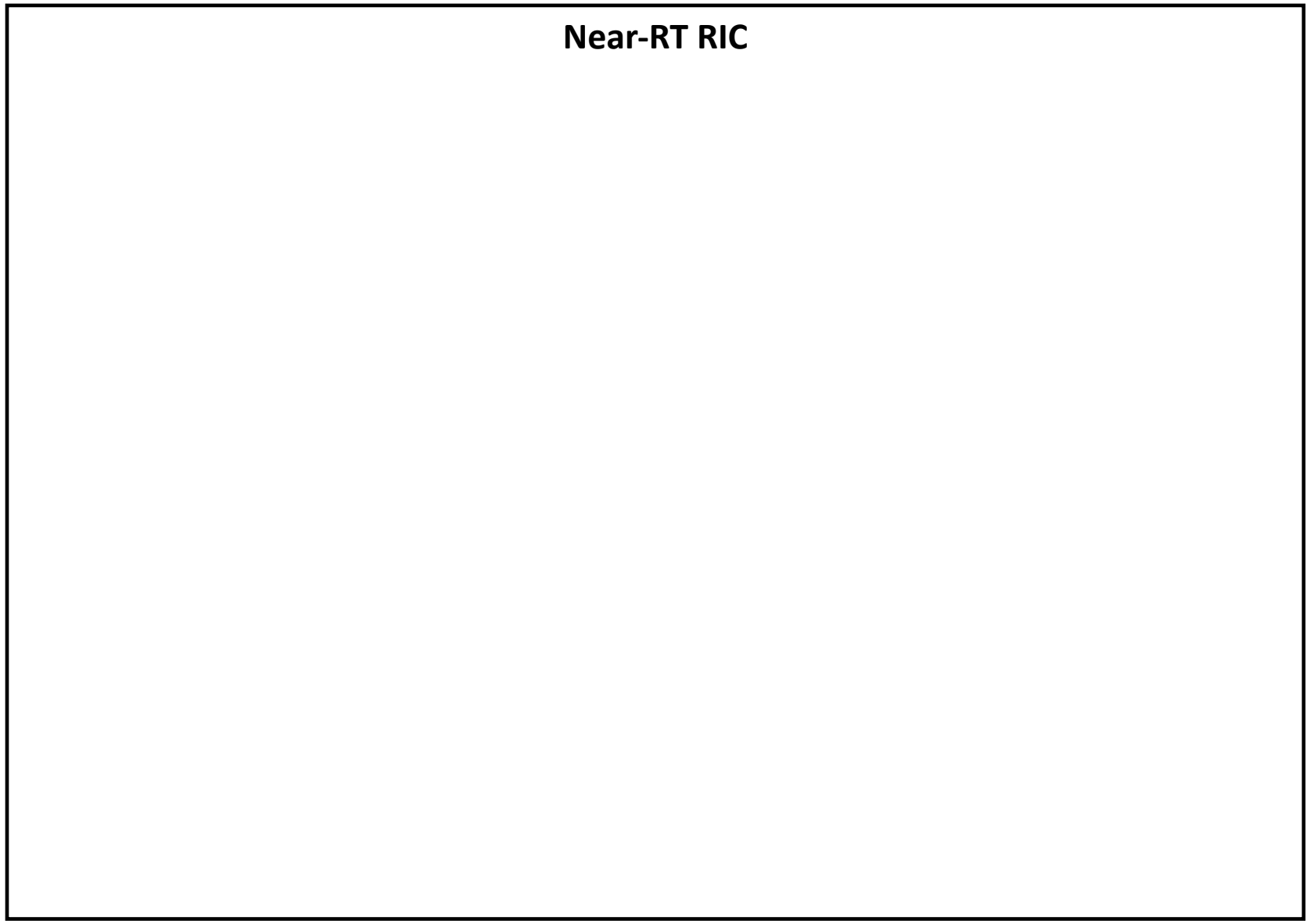


- **Simpler approach**: convenient for the researcher still in exploratory or design phase and with interest in the effect of O-RAN deployment rather than the deployment itself
- **Requires only the base version of ns-3 and the module that we provide**

[1] Leonardo Bonati, Michele Polese, Salvatore D'Oro, Stefano Basagni, and Tommaso Melodia. 2022. OpenRAN Gym: An Open Toolbox for Data Collection and Experimentation with AI in O-RAN. In 2022 IEEE Wireless Communications and Networking Conference (WCNC 2022).

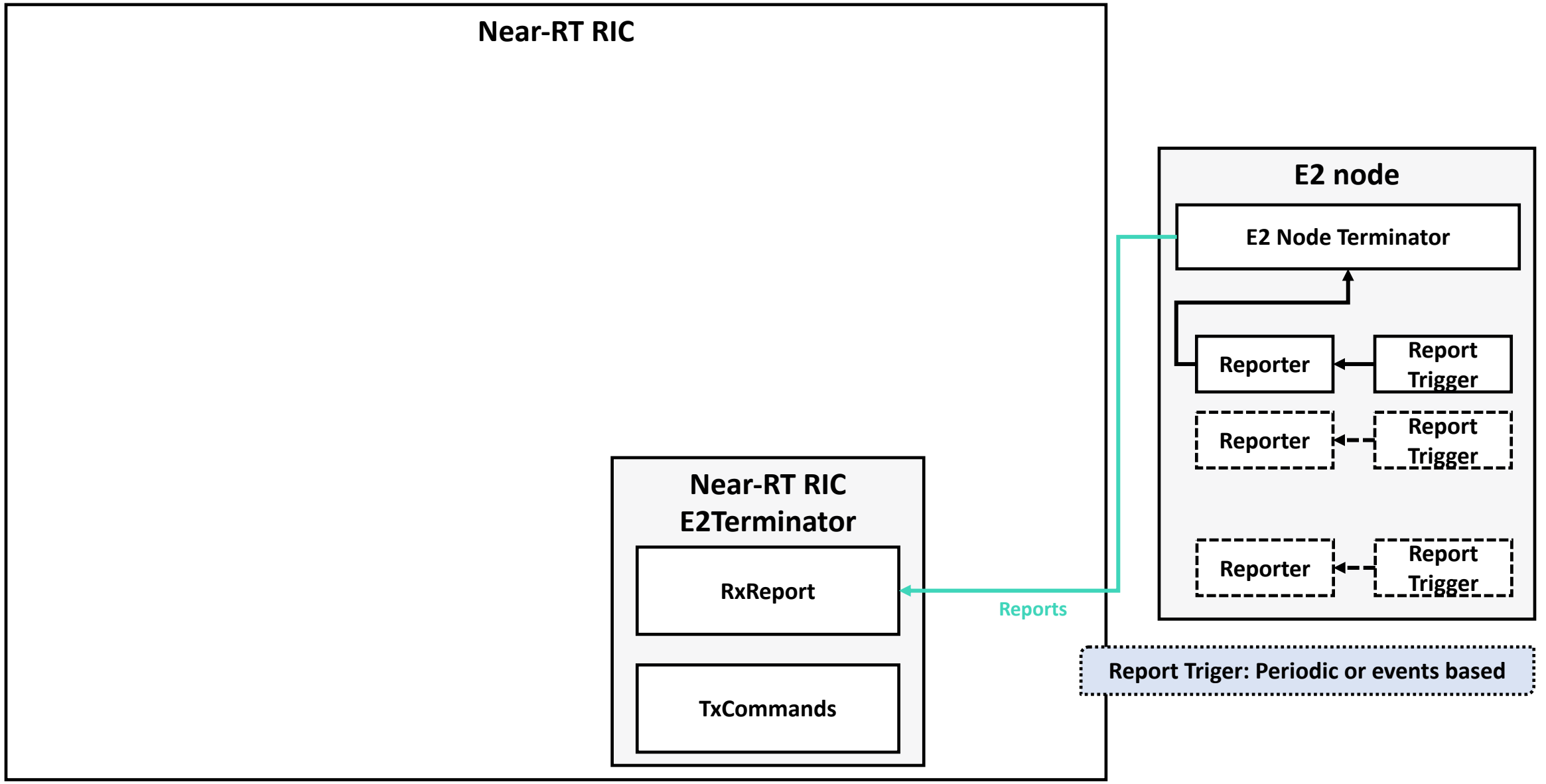
The O-RAN ns-3 module

Architecture



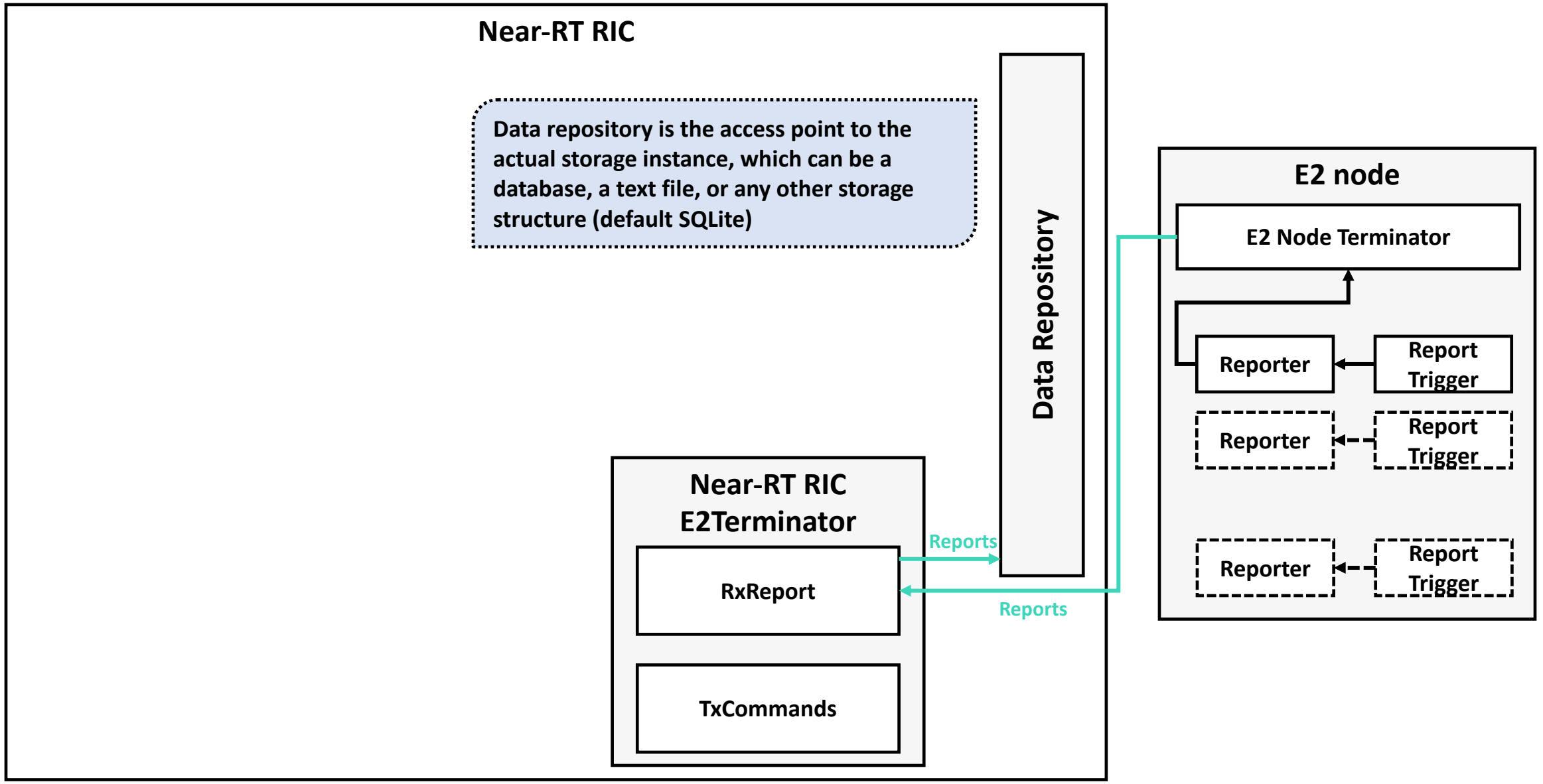
The O-RAN ns-3 module

Architecture



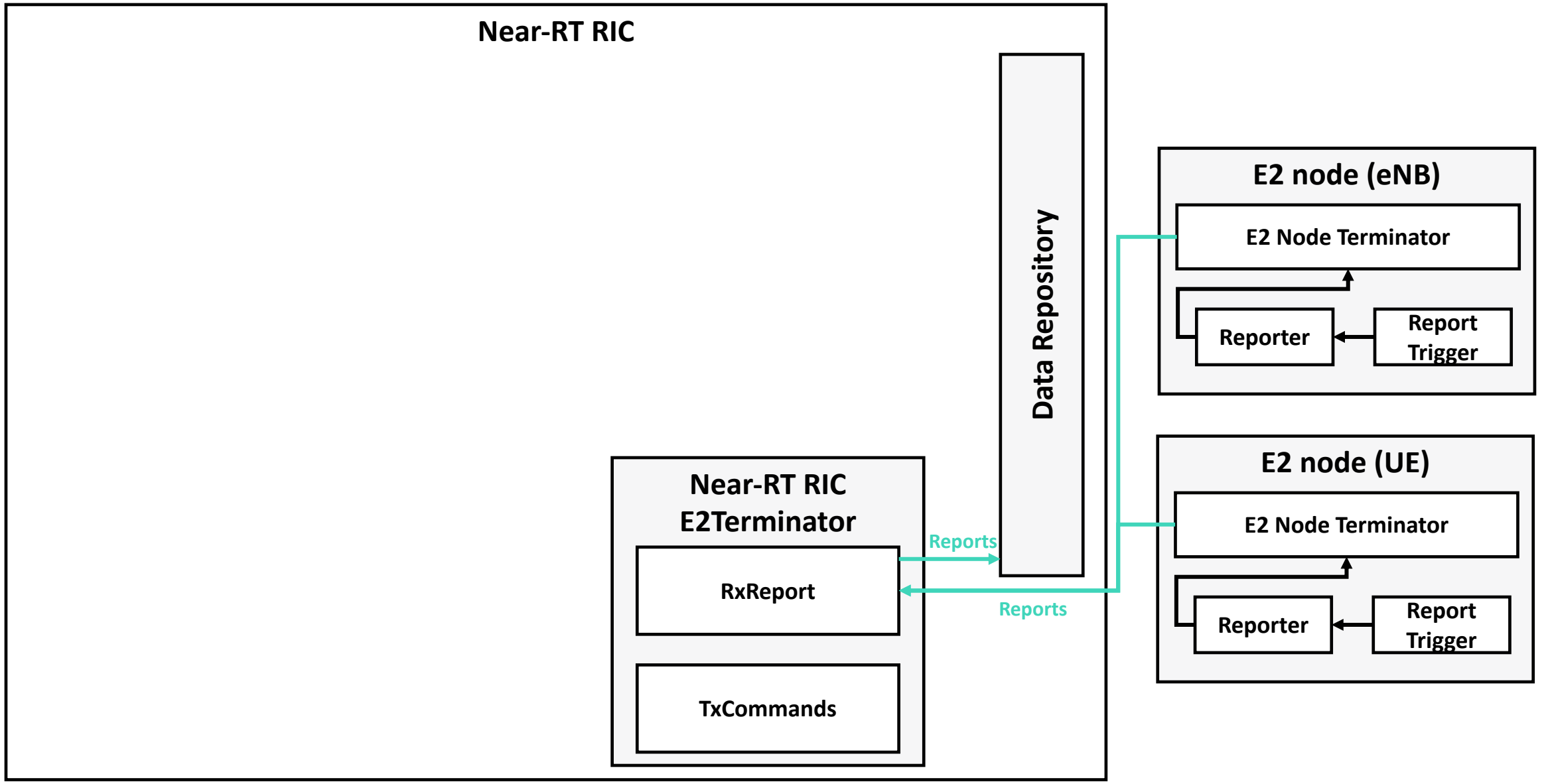
The O-RAN ns-3 module

Architecture

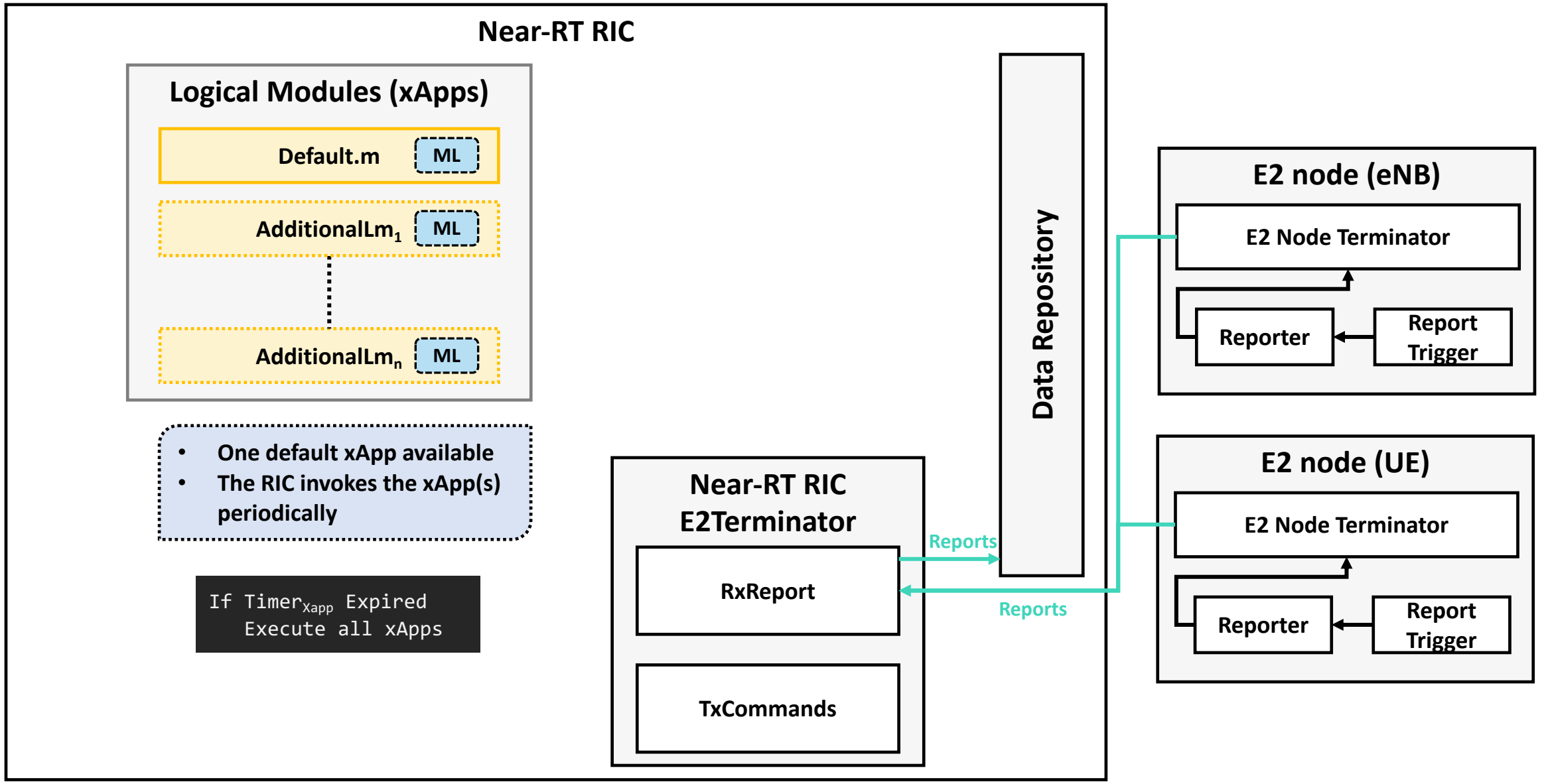


The O-RAN ns-3 module

Architecture

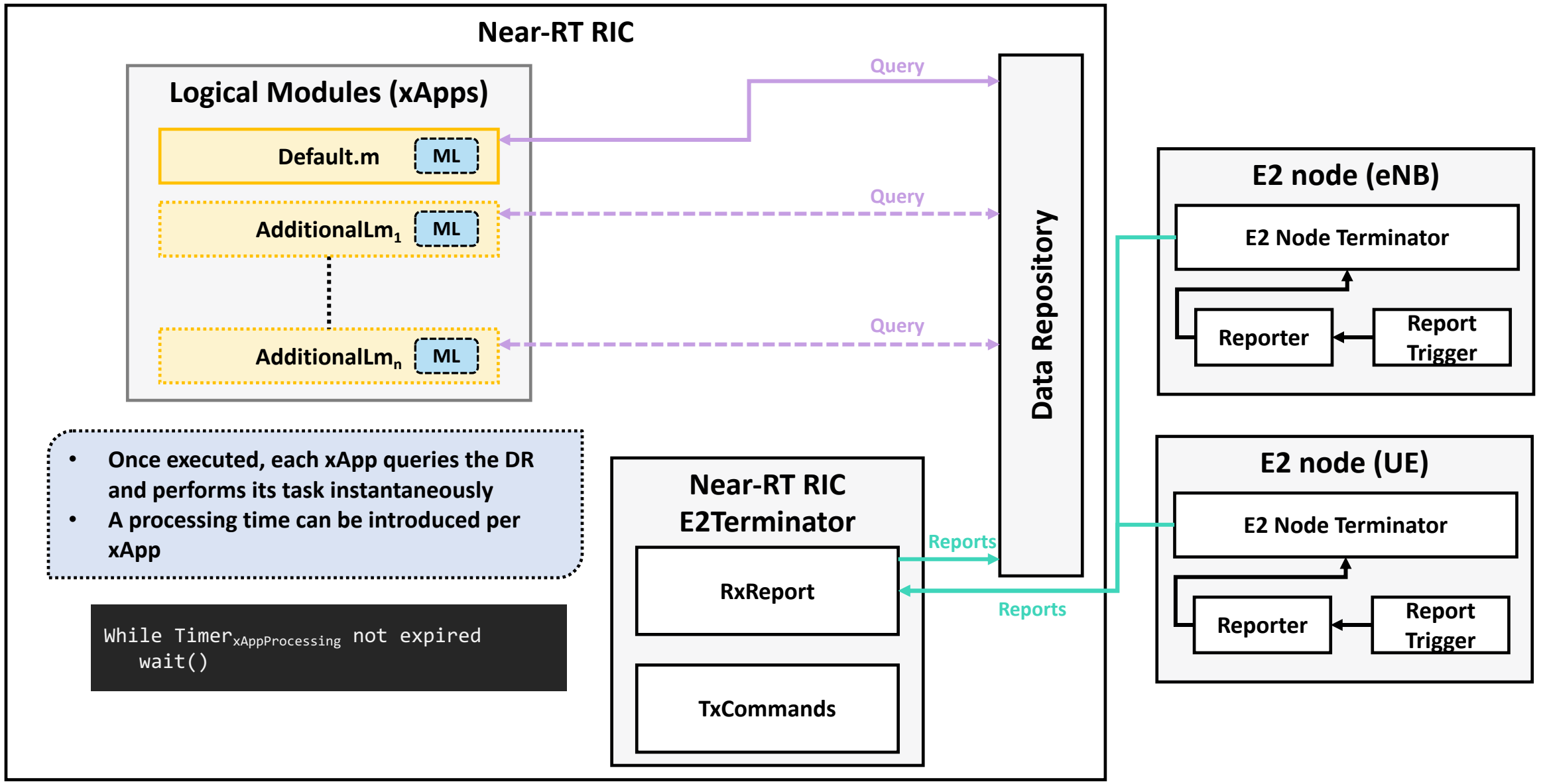


The O-RAN ns-3 module
Architecture



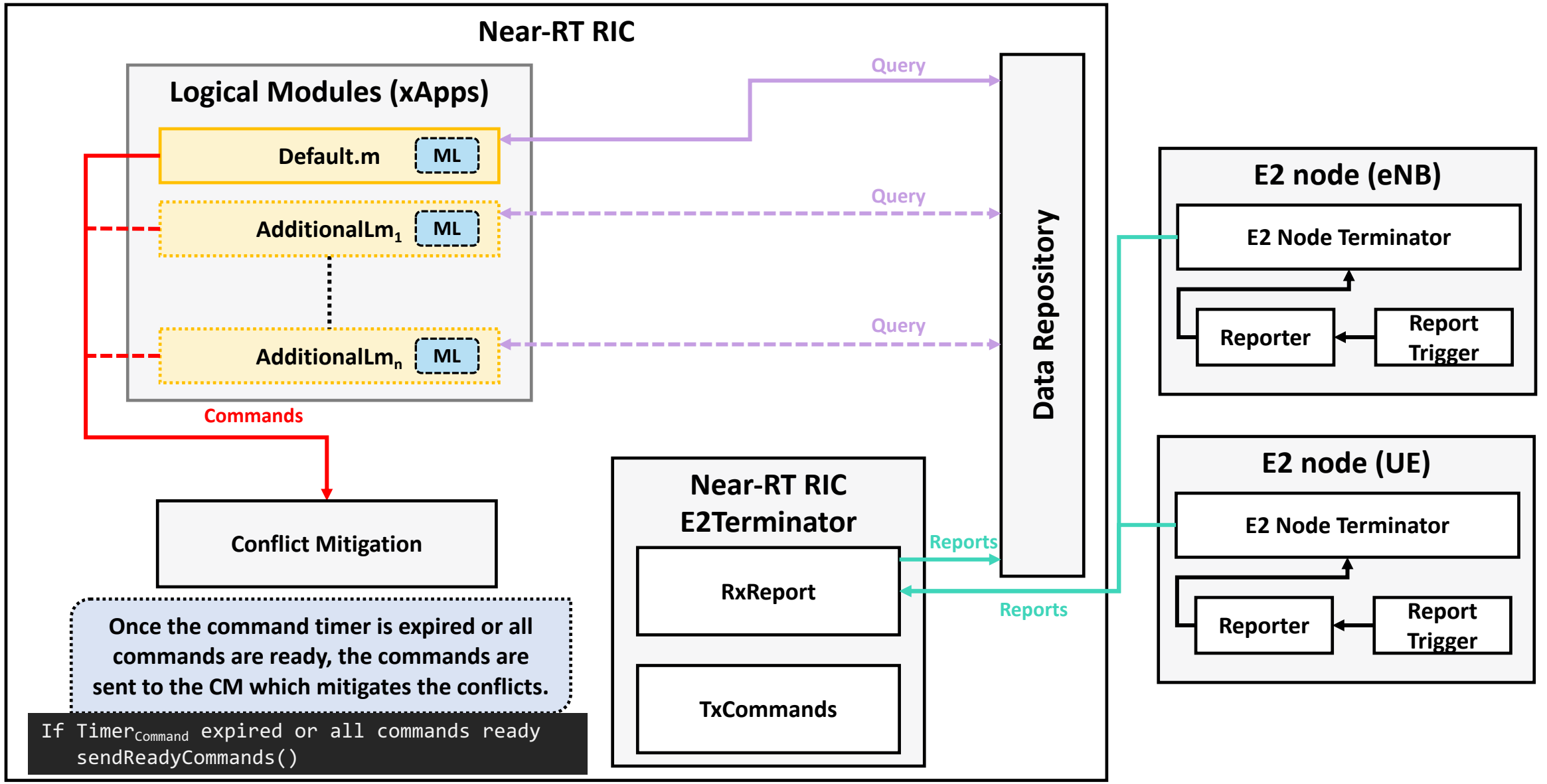
The O-RAN ns-3 module

Architecture



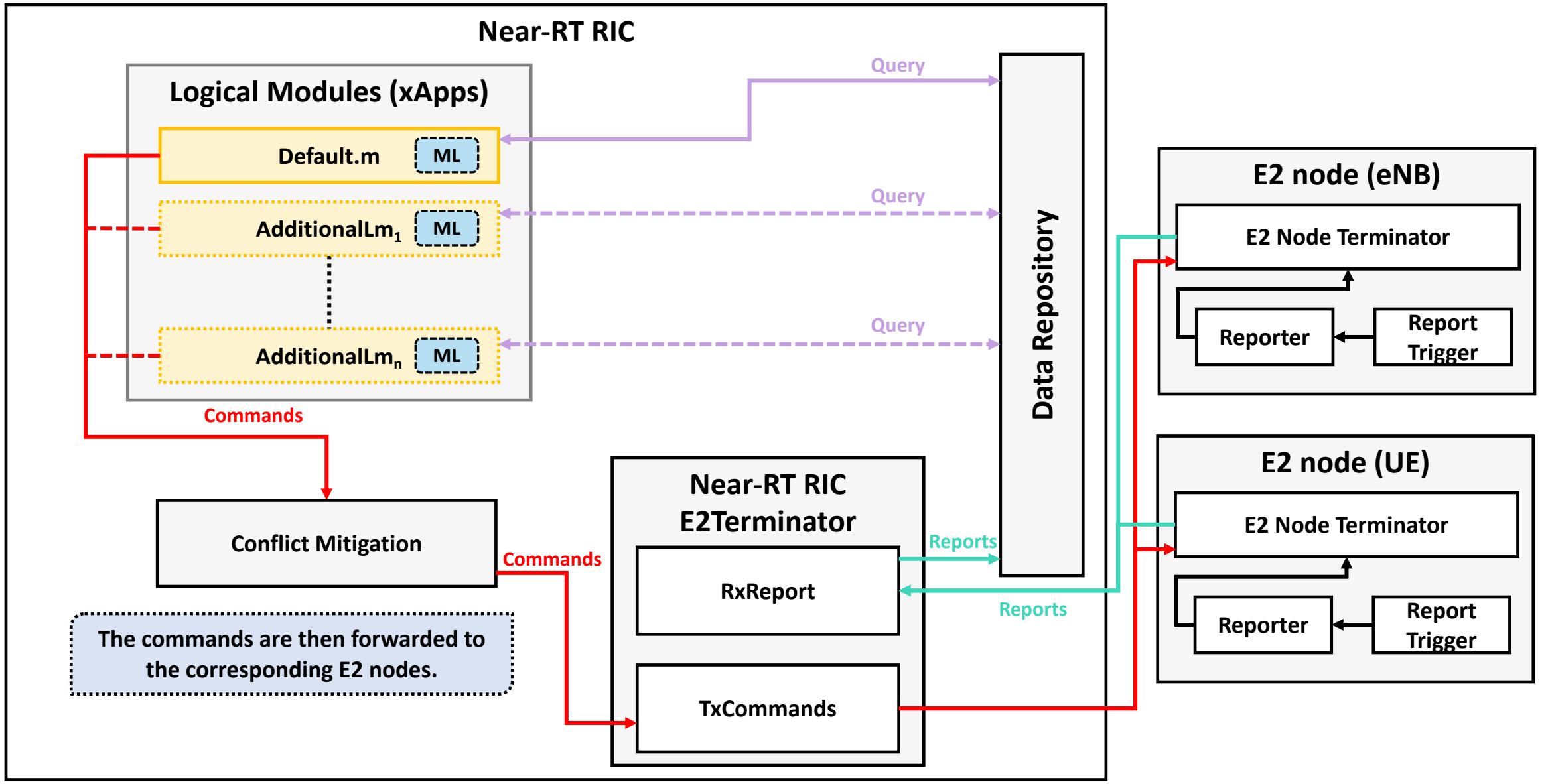
The O-RAN ns-3 module

Architecture



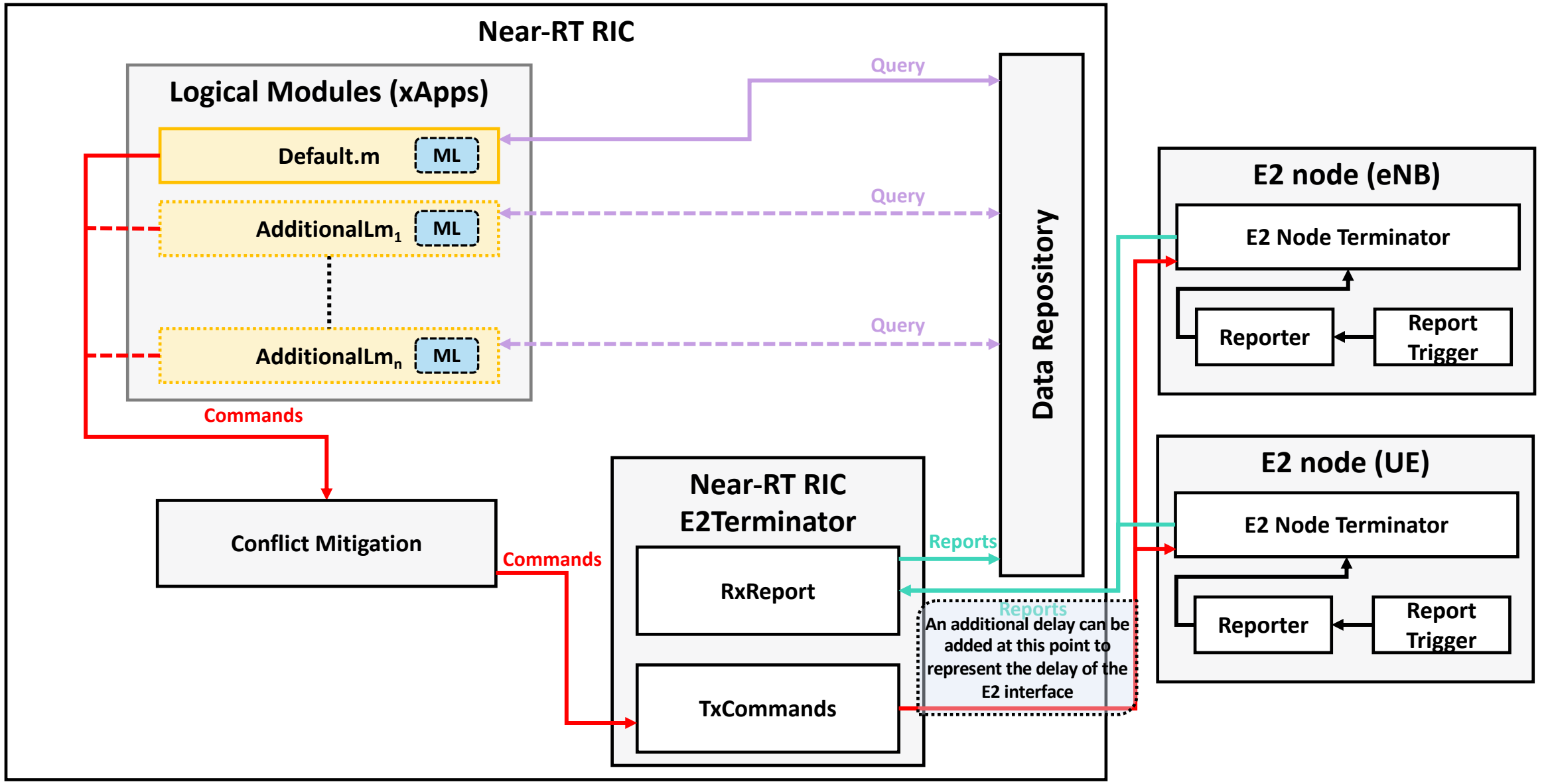
The O-RAN ns-3 module

Architecture



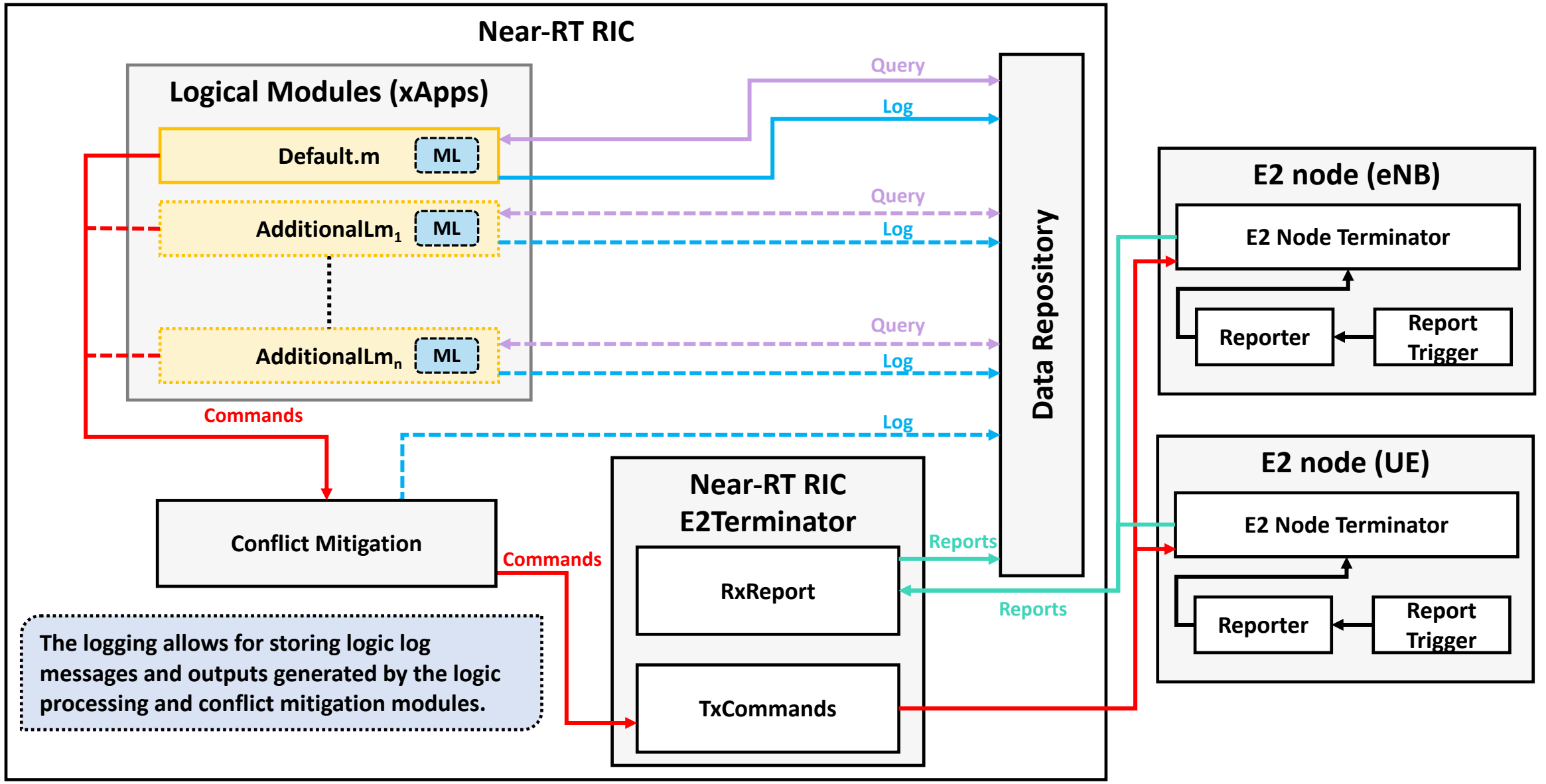
The O-RAN ns-3 module

Architecture



The O-RAN ns-3 module

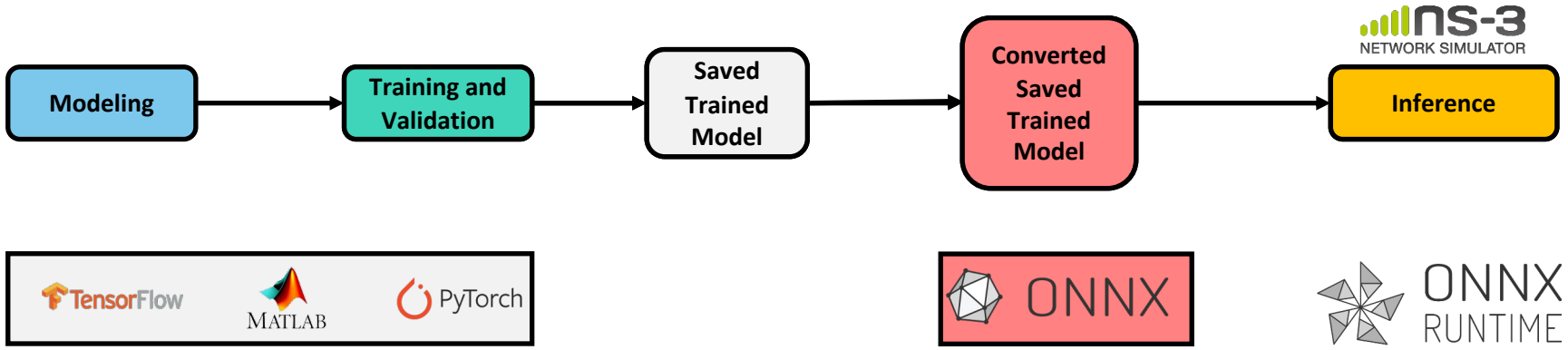
Architecture



The O-RAN ns-3 module
ML

ns3-oran uses Open Neural Network Exchange (ONNX) for inferences

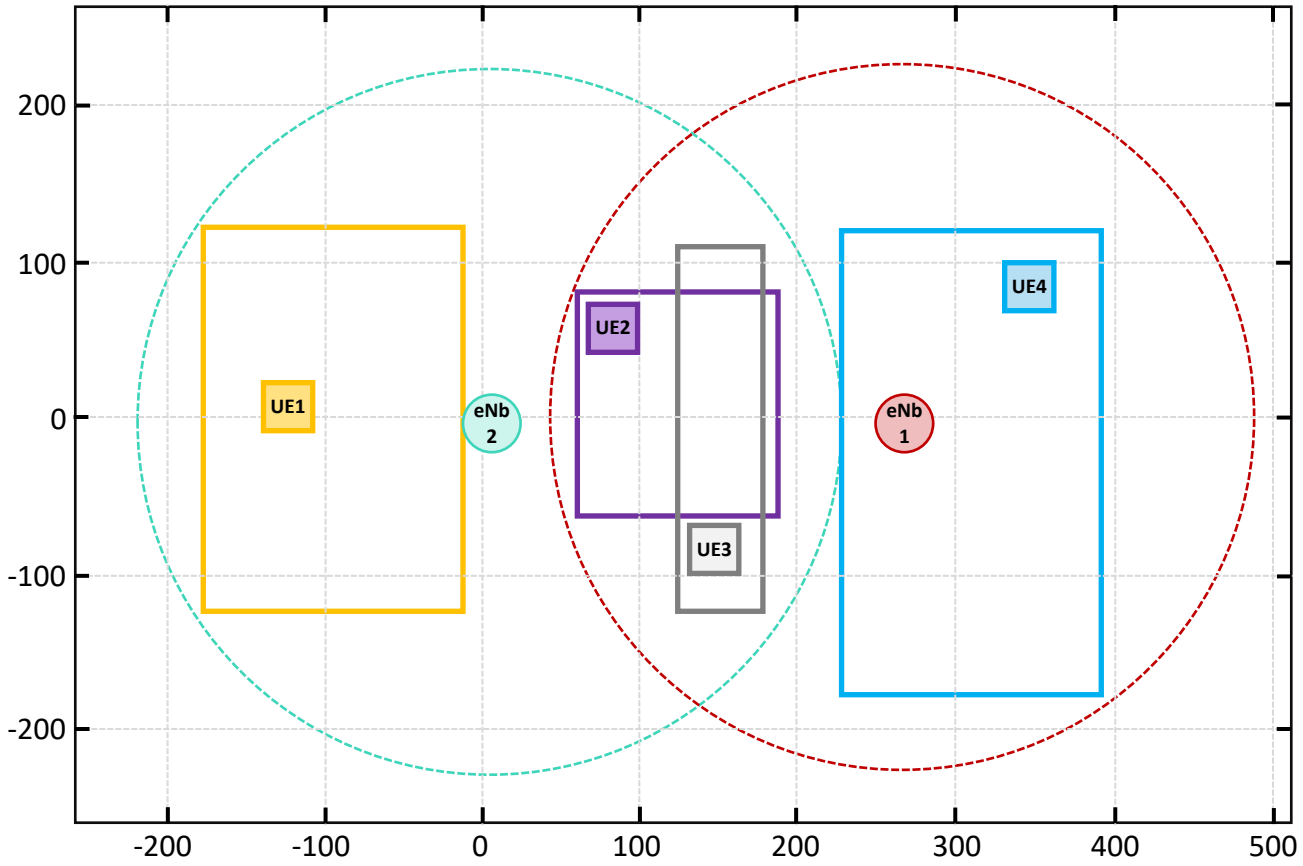
- ONNX defined to facilitate interoperability between various ML framework and tools.
- Both a file format and a set of operators
- Supported by PyTorch, MXNet but converters exist for TensorFlow and CoreML



Enabled by ONNX runtime (https://onnxruntime.ai/) **linked through the ns-3 build process**

- Allows access to ONNX models and query them directly via function calls as any other C++ Object => Fast compared to the usage of inter-process communication

Case study
Handovers: Scenario



eNb1 can serve a single UE without losing a single packet

eNb2 can serve a single UE without losing a single packet

- **2 Single-cell eNodeBs** (eNb 1, eNb 2)
- **4 mobile UEs**
  - UE1 and UE4: restricted to the coverage of a single eNodeB
  - UE2 and UE3: moves into an area covered by both cells
- The goal is to **trigger efficient handover for UE2 and UE3**
- **Data Traffic**: Each UE: 290 kbit/s video stream download for 200 s
- We compare the performances of **four approaches** using the following metrics
  - Packet Loss Rate for each UE
  - Aggregated Throughput observed at the UEs
  - Total Number of handovers

Case study
Handovers: the four approaches

Non O-RAN based

- ***Baseline***: No handover and UE in Configuration 1
- ***Heuristic***: Use the Reference Signal Received Power (RSRP) of the UE's serving cell and neighboring cells

  If $RSRP_{neighborCell} > RSRP_{servingCell}$
  Handover to neighbor cell

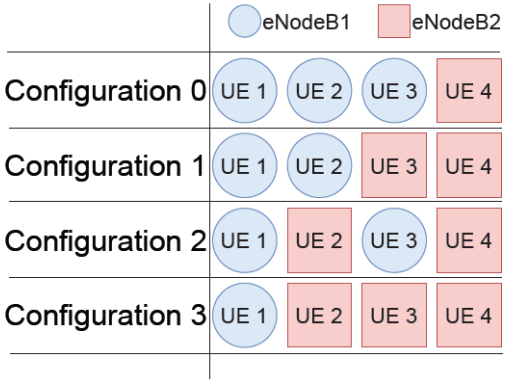


Table 1: Parameters of the Simulated Scenario

Parameter	*Value*	*Unit*
Simulated Time	210	s
Traffic	290	kbits/s
Pathloss Model	Cost231	NA
Max UE-to-eNodeB Distance	461	m
Scheduler	Round Robin	NA
O-RAN dependent parameters		
LM Invocation Interval	1	s
UE/eNodeB Report Interval	1	s
ML training parameters		
Epochs	3	NA
Training Set Size	666 667	NA
Verification Set Size	333 333	NA

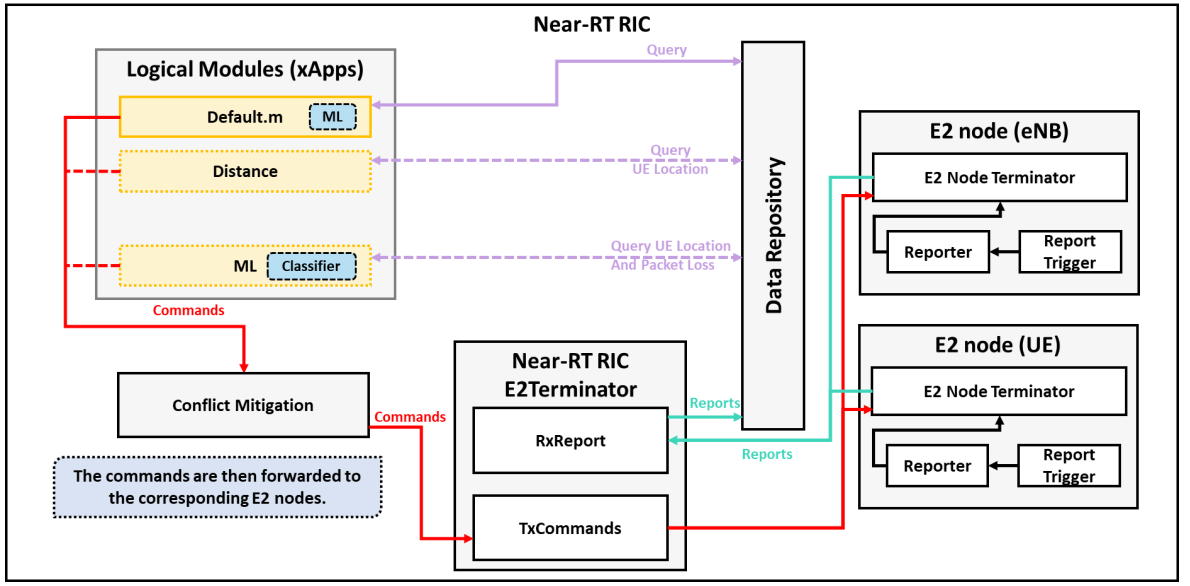
O-RAN based

- *Distance*: Use the distance from the UE's to the serving cell and neighboring cells

  If $distance_{neighborCell} < distance_{servingCell}$
  Handover to neighbor cell

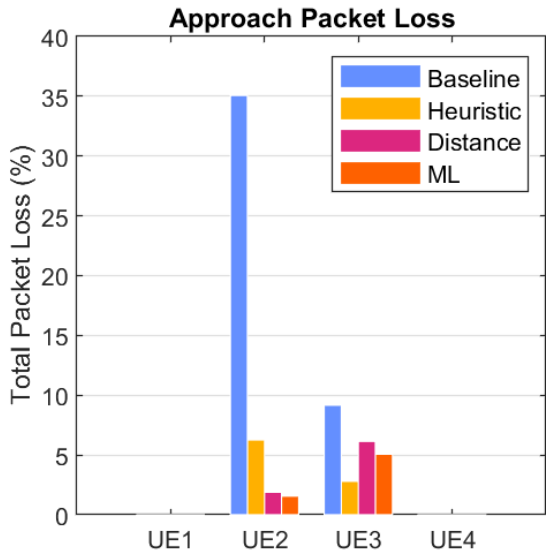
  - The xApp collects the location of each UE

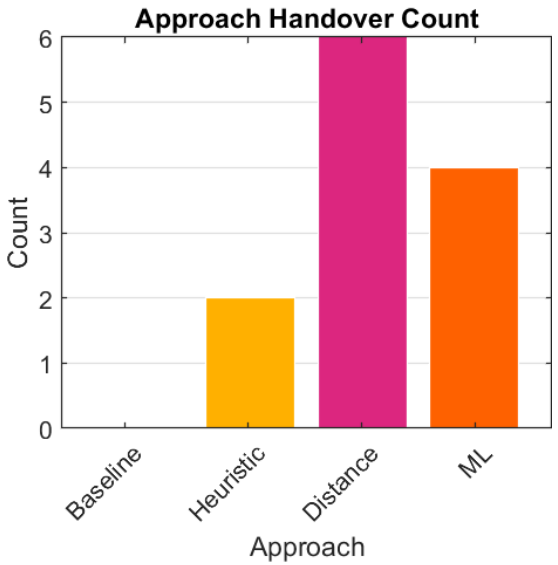
- *ML*: Deep learning Classification (4 configurations)
  - **Input**: distance of each UE to each eNodeB and the observed normalized packet loss of the UEs during the last 1 s
  - **Output**: Ranking of each configuration, which allows the selection of the one providing the lowest overall packet loss for the next 1 s
  - **Training**: One simulation for each configuration without any handover algorithms. Post-processing the data determines which configuration is best at any given time
  - The xApp collects the location of each UE as well as the packet loss and passes them to the classifier

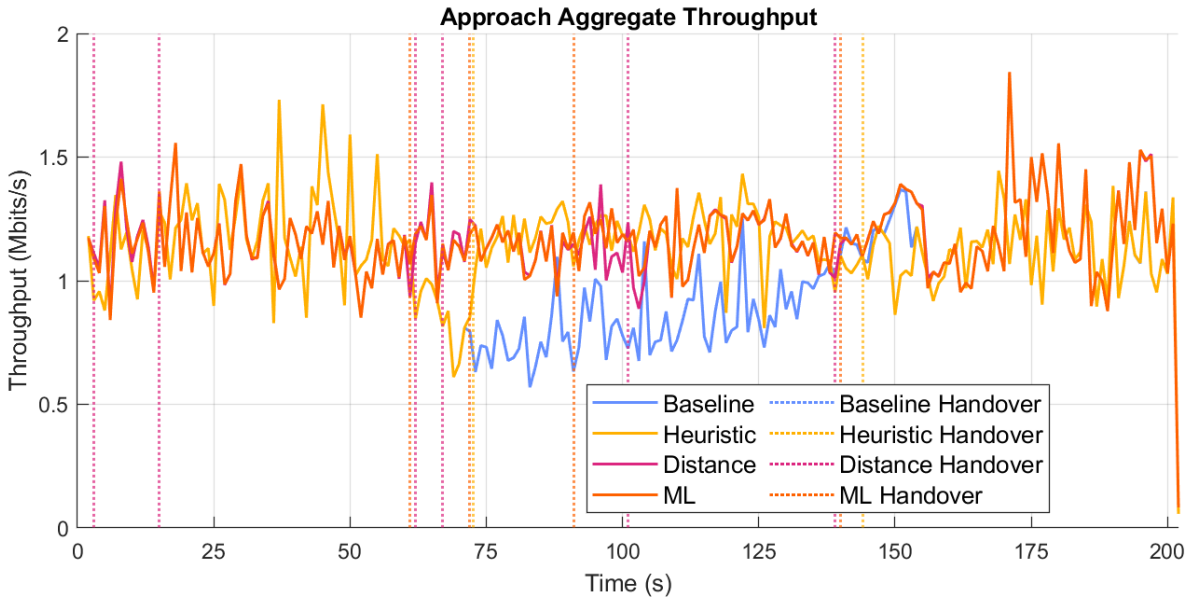


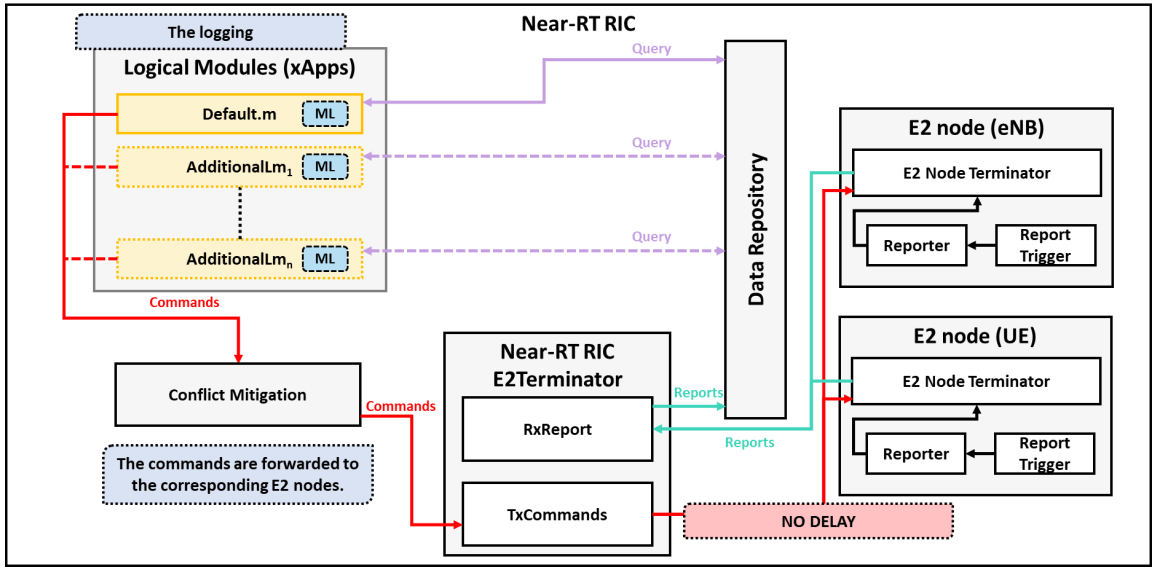
Case study

Results per approach



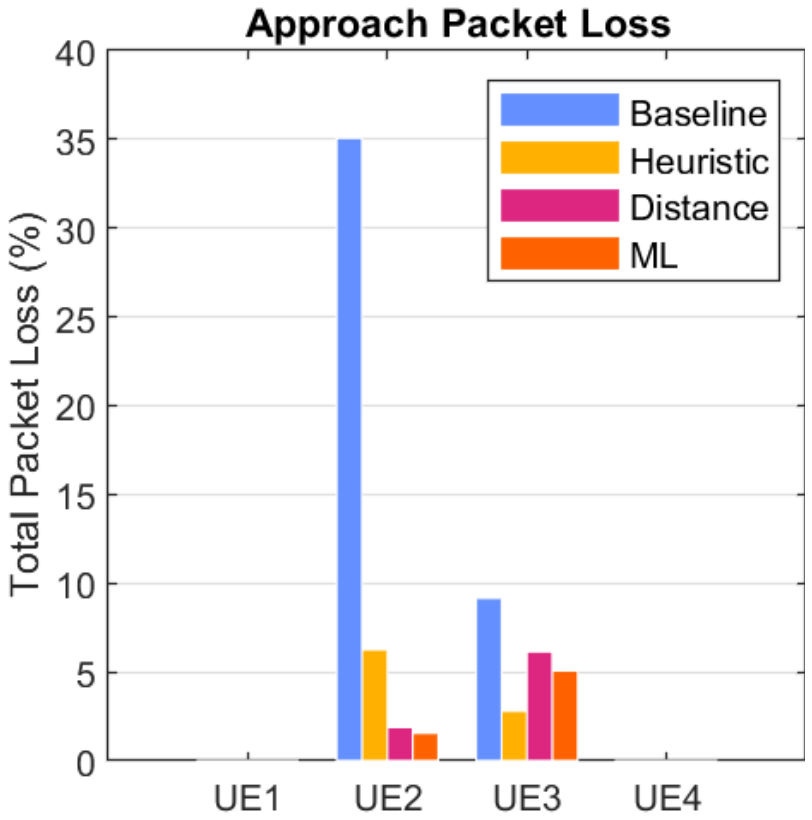


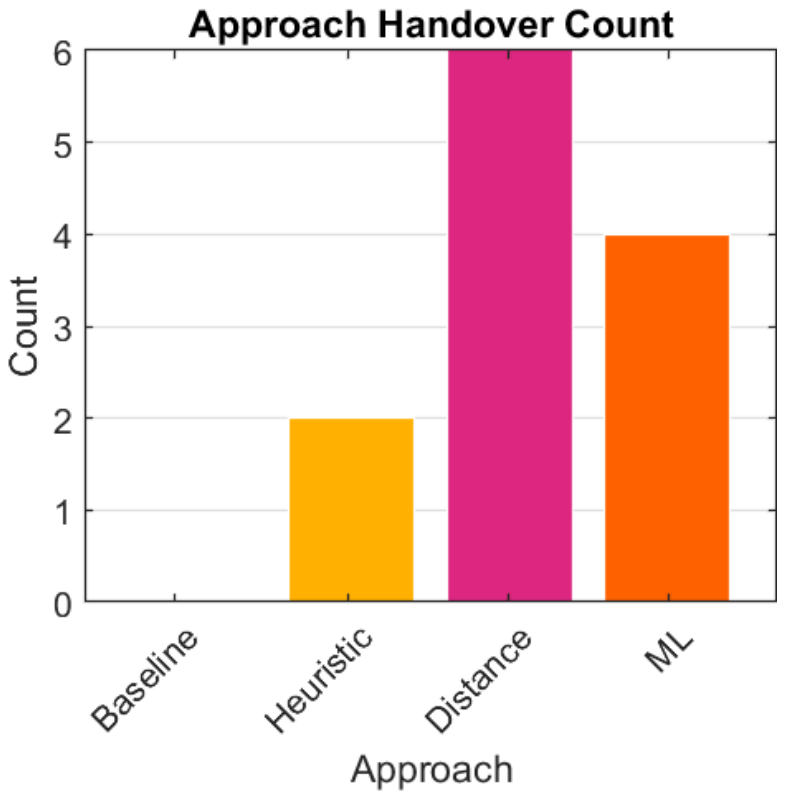




No delay added

Case study
Results per approach



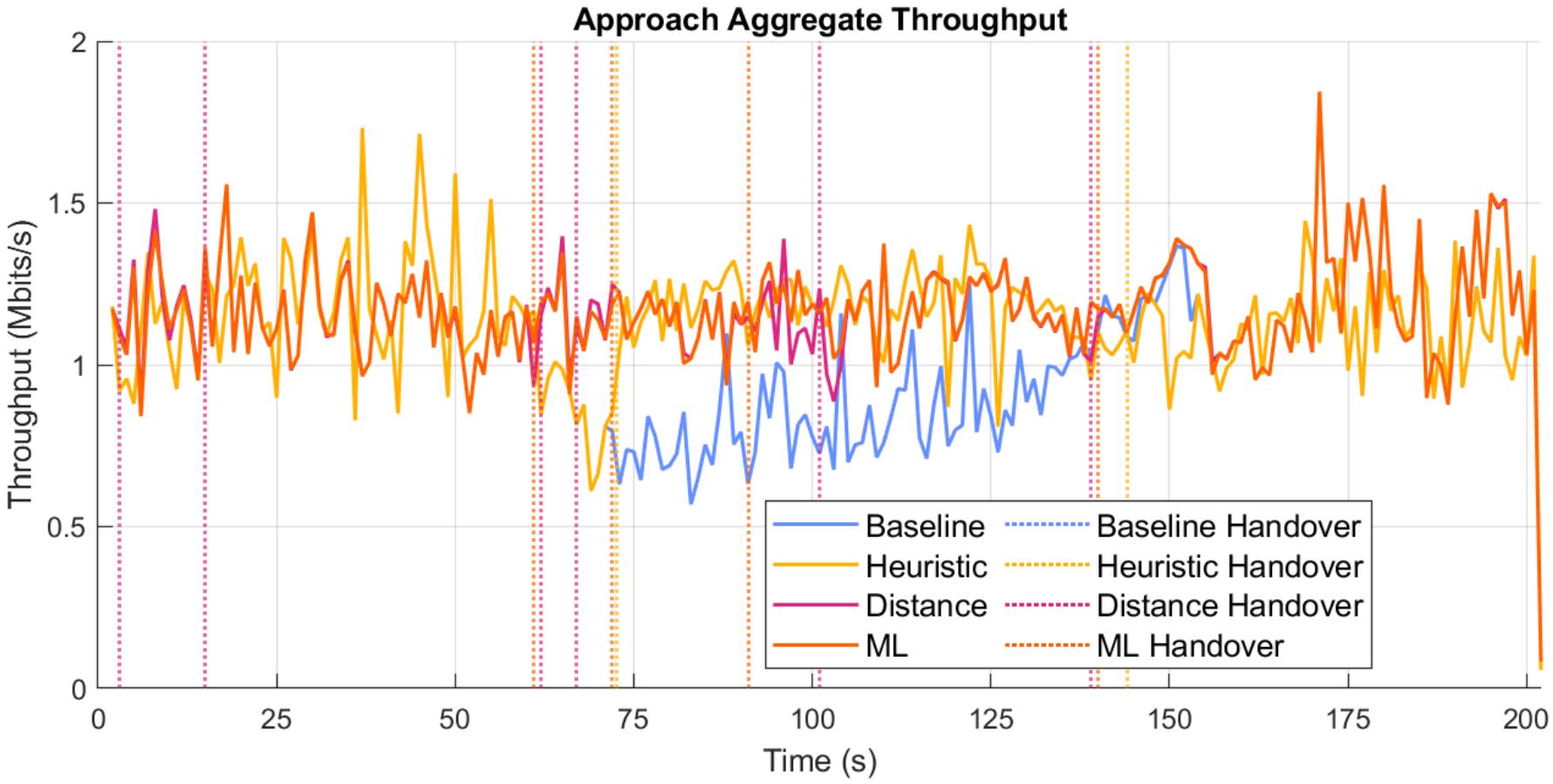


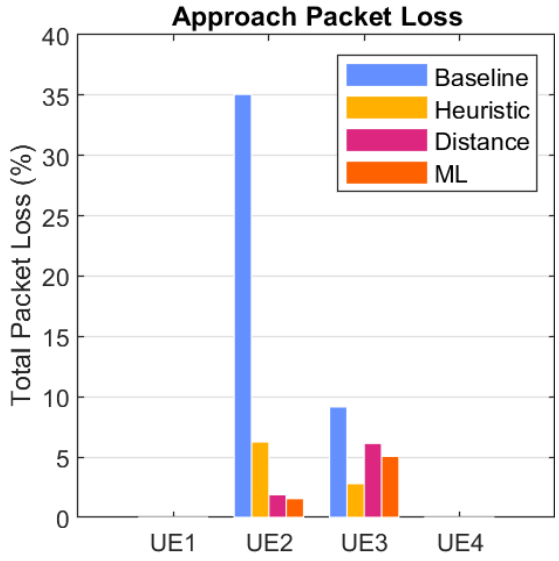
- Baseline: Higher Loss (Avg Packet Loss 22.1%)
- The greatest improvements come from the O-RAN-based approaches, with the ML approach providing the greatest reduction in packet loss (Avg Packet Loss 3.5%)

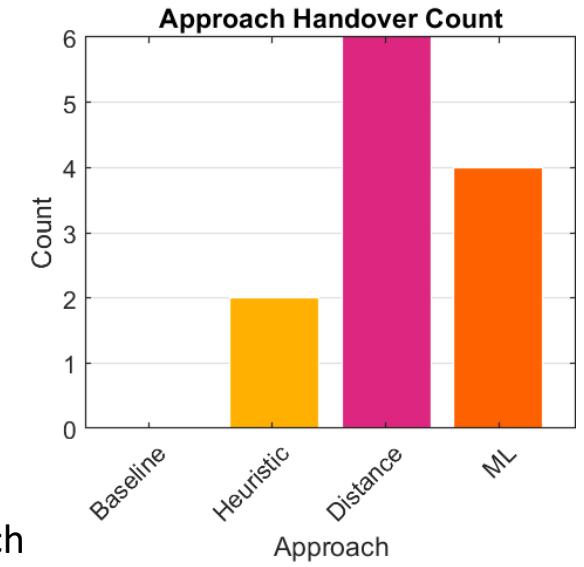
- O-RAN-based approaches require additional handovers

Case study

Results per approach: Aggregate throughput



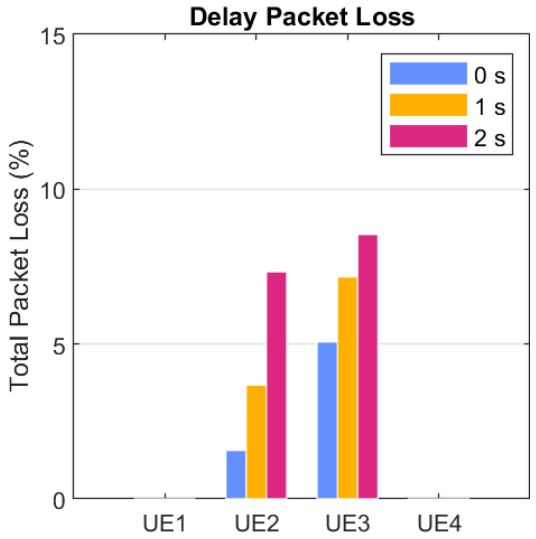


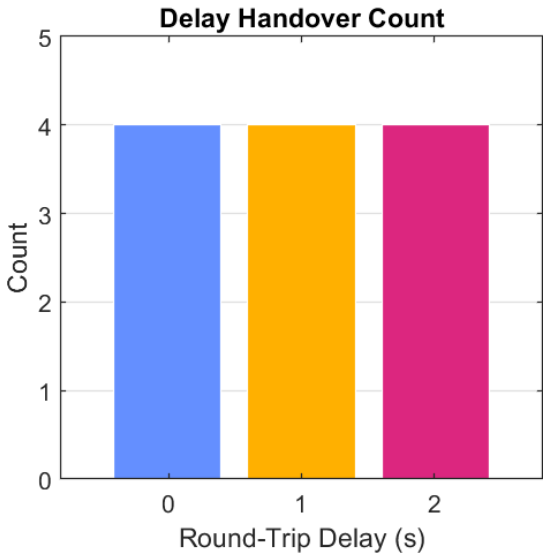


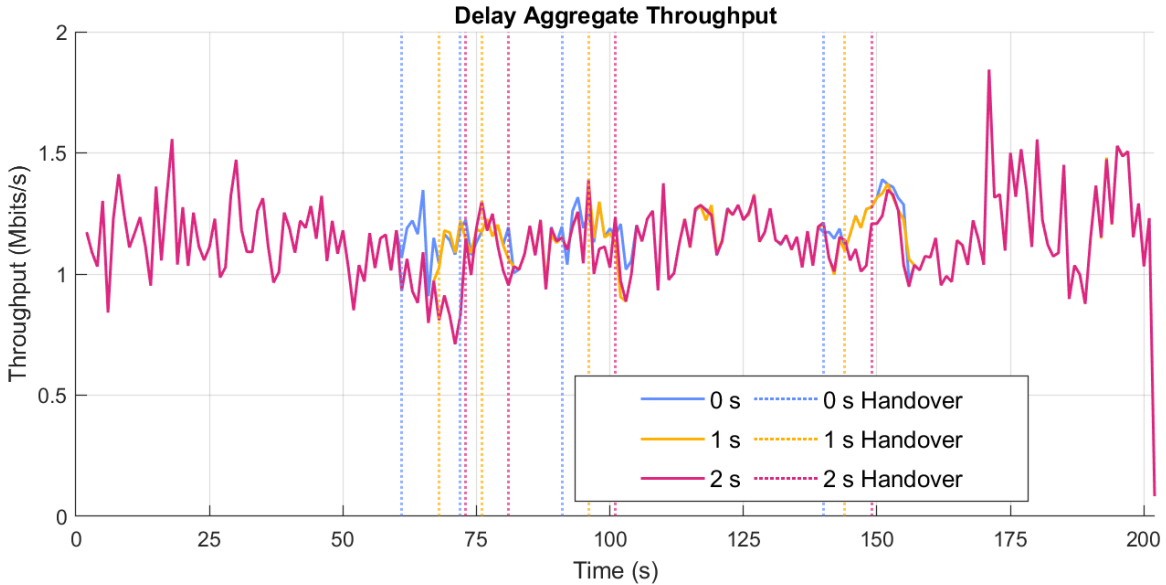
- Distance and ML approaches provide an overall higher throughput throughout the entire simulation.
- Two O-Ran based approaches (Distance and ML) throughput greater than 1Mbit/s
- Baseline and heuristic dips below 610 kbit/s.
- ML approach is able to provide increased performance but with fewer handovers than the distance approach

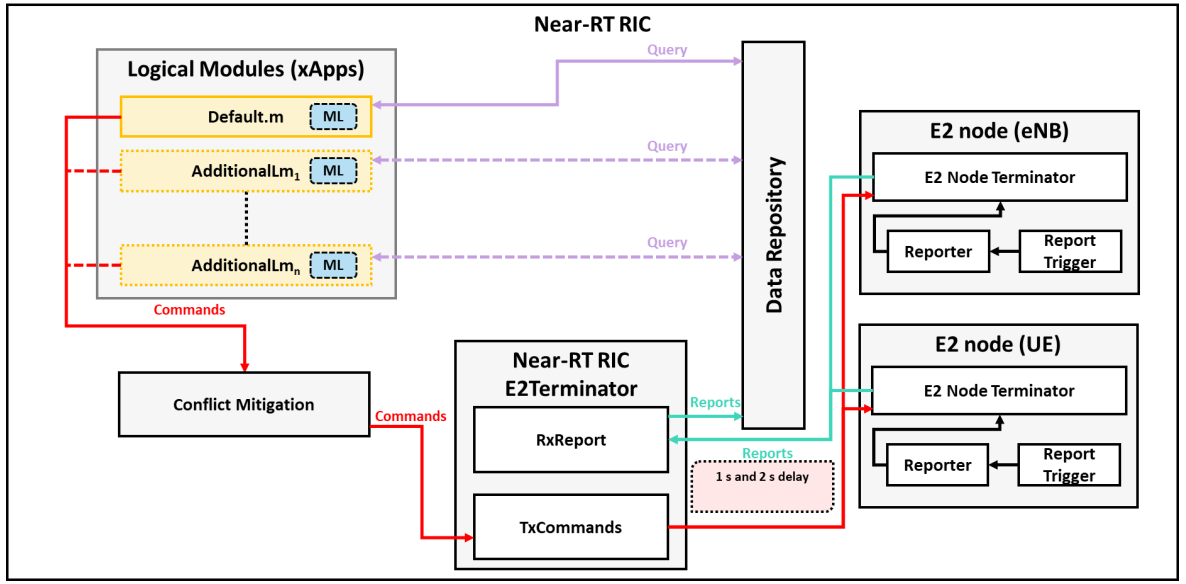
Case study

Results for ML approach: additional delay





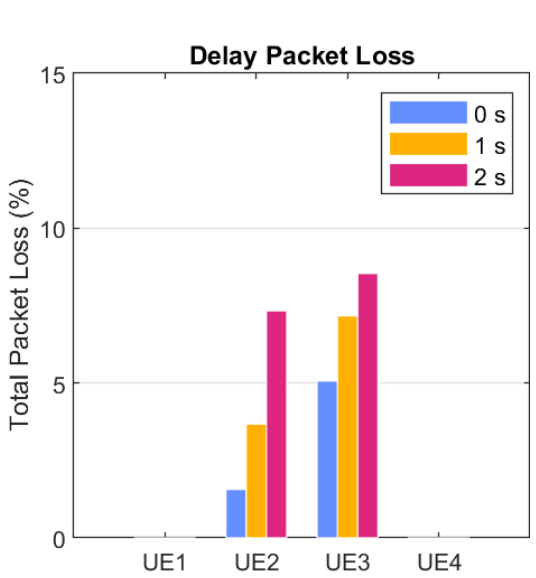


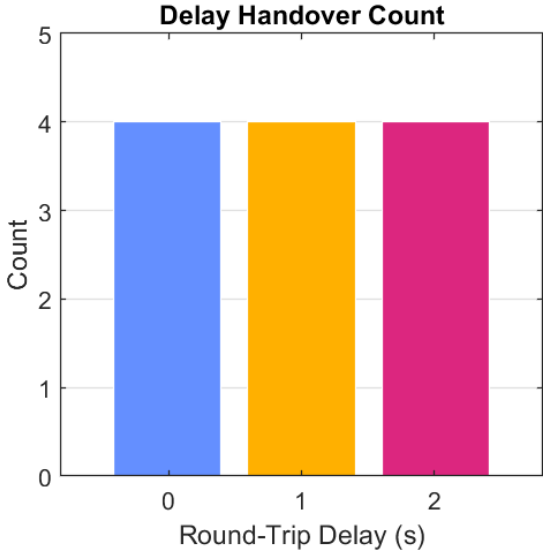


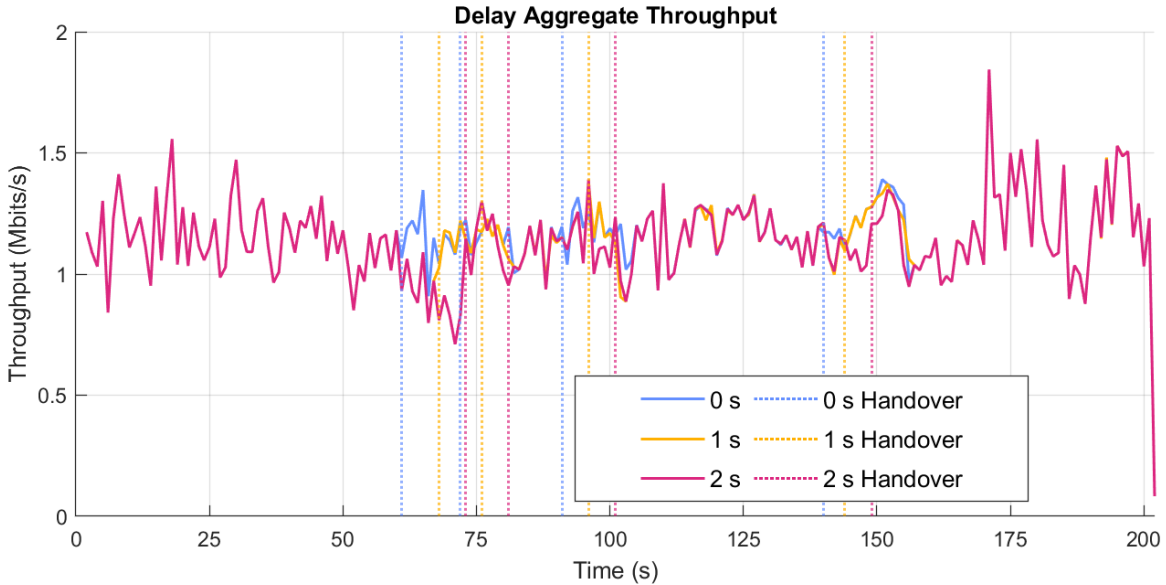
1s and 2s delay added

Case study

Results for ML approach: additional delay







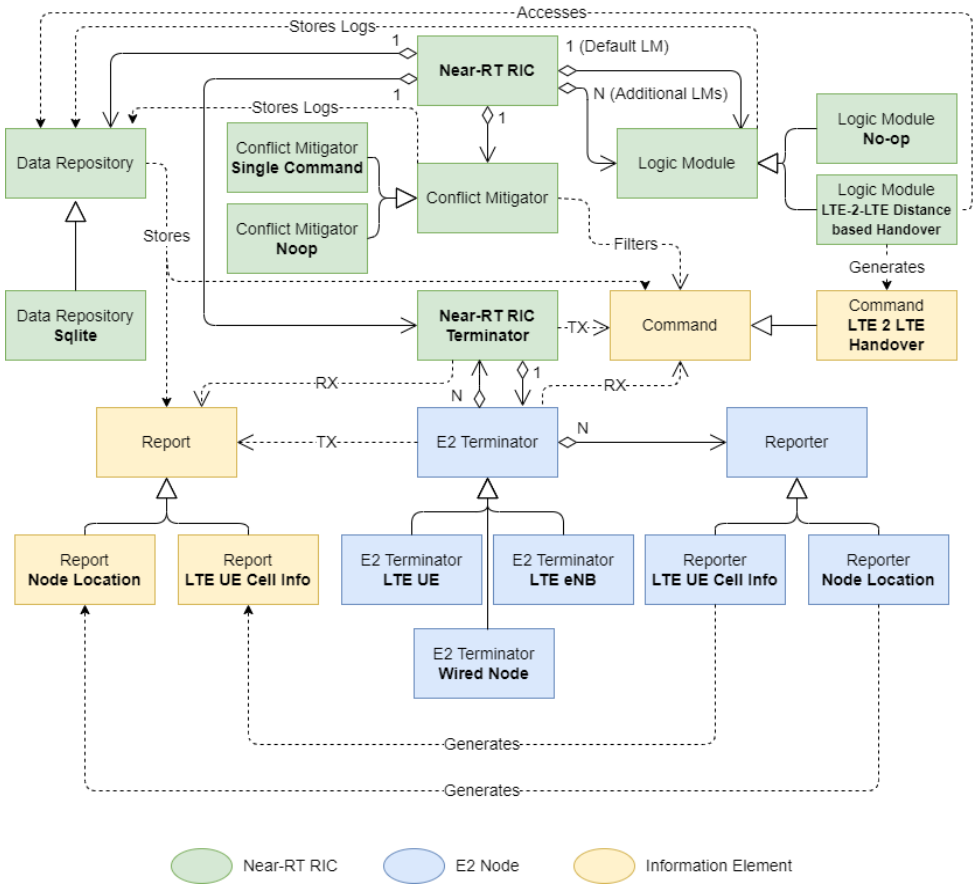
- The larger the delay, the larger the packet loss
- The number of handovers remains unchanged
- Larger delays hinder throughput performance

The O-RAN ns-3 module

Easy to extend

- All the components that contain logic modifiable by the end users have been modeled hierarchically, i.e., parent classes can take care of common actions and methods, and leave child models to focus on the logic itself.

O-RAN with O-RAN Distance Approach

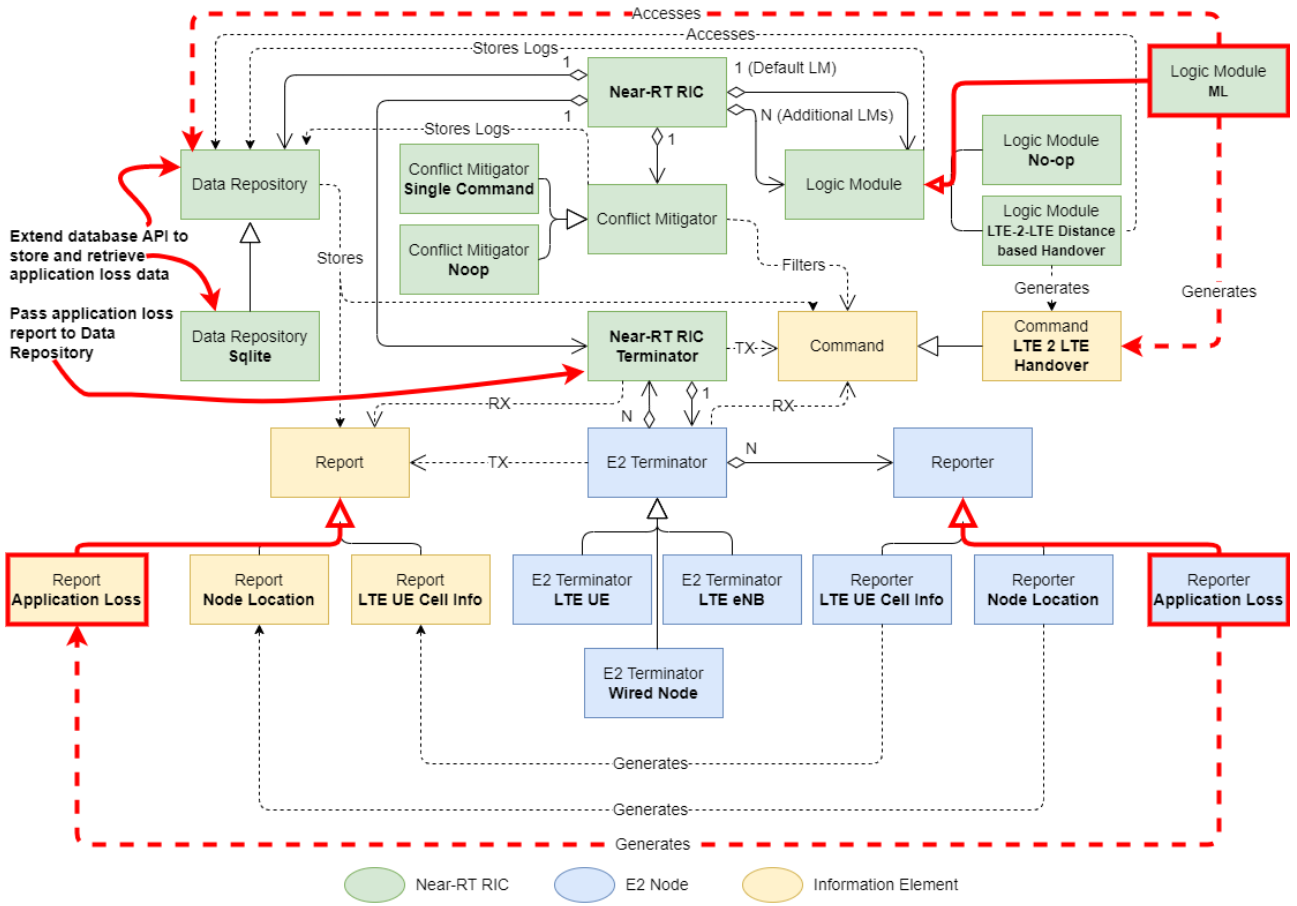


The O-RAN ns-3 module

Easy to extend

- All the components that contain logic that may be modified by end users have been modeled hierarchically, i.e., parent classes can take care of common actions and methods, and leave child models to focus on the logic itself.

To add O-RAN ML Approach



Conclusion and next steps

- ns3-oran makes it possible to quickly define, evaluate, and tune solutions and mechanisms for improving RAN performance based on O-RAN architecture

- ns3-oran is user-friendly and easily extendable

- Future work:
  - Implement O-RAN case studies
  - Use multiple xApps simultaneously
  - Extend ns3-oran to support NR Technology
  - Study the usage of ONNX to train models