

D release (Jul 2021)

Welcome to the D release page for the O-RAN Software community.

To download the source code for the D release, please check the section "D release source code, container images and deployment instructions" at the end of each subproject table below for the subproject that you are interested in. The same section - if applicable - also includes a reference to the container images that make up the D release and to deployment instructions.

[General D release container image list for all subprojects.](#)

- [Near-Real-time RIC X-APPs \(RICAPP\)](#)
- [Near-Real-time RAN Intelligent Controller Platform \(E2 Interface\) \(RICPLT\)](#)
- [Non-Real-time RIC \(A1 Interface\) \(NONRTTRIC\)](#)
- [Operation and Maintenance \(OAM\)](#)
- [O-RAN Central Unit \(OCU\)](#)
- [O-DU High](#)
- [O-DU Low](#)
- [Simulators \(SIM\)](#)
- [Infrastructure \(INF\)](#)
- [Integration and Test \(INT\)](#)
- [Documentation \(DOC\)](#)
- [Service Management and Orchestration \(SMO\)](#)

Near-Real-time RIC X-APPs (RICAPP)

Primary Goals: Expand the community working on open source xApps for O-RAN SC. Enhance the set of open source xApps in support of the R-SAC use cases (traffic steering, health check, life cycle management) as well new use cases. Update and enhance existing xApps to take advantage of the new features in xApp SDK (implemented by the xApp frameworks in C++, go, and python).

D release highlights (7-7-21):

- Expanded set of xApps from expanded community: D release includes xApps from HCL (AD, QP, Bouncer), Samsung (KPI, HW-python, HW-go), ChinaMobile (LP), Parana Technical University (TS), and AT&T (HW, MC).
- AD xApp implements true ML-based anomaly detection where the ML-model is trained using data from Viavi E2 simulator and then at runtime, AD reads time series data from the inFlux (DB new RIC platform component) and raises anomaly alarms.
- QP xApp implements for the first time true ML-based throughput prediction trained using data from Viavi simulator. QP receives prediction request for a set of UEs, determines the current UE metrics and current neighbor cells by reading the inFlux DB and provides prediction of the throughput prediction in the current serving cell as well as the neighboring cells. Note that the current QP includes the functionality previous embedded in the separate QP-driver xApp.
- TS xApp has been extended to receive anomaly detection messages and trigger predictions request based on these messages. It is also extended to make a call to Viavi simulator (REST based call while waiting for E2SM RC) to trigger UE handover.
- The AD, QP, and TS xApps have been integrated to implement an anomaly detection driven traffic steering use case.
- KPIMON xApp now implements E2SM KPM 2.0.3 - when an E2 simulator is available that implements that SM, the KPIMON flow can be tested (likely E Release) and integrated with the rest of the use case.
- The new LP xApp introduces a load prediction capability - this new xApp will be integrated with the traffic steering use case in E Release
- The new demo (hello world) xApps HW-python and HW-go introduce the initial versions of xApps to demonstrate how all RIC features can be utilized from xApps written in python and go.
- Bouncer xApp has now been properly introduced as an xApp in the RICAPP project - it will allow the performance benchmarking of the RIC platform (latency of INSERT-CONTROL loop, number of E2 Nodes supported).
- The HW (C++) and MC xApps include minor updates.
- Image list: [can be found here](#).
- Instructions for testing the xApps: [can be found here....](#)

PTL: [Matti Hiltunen](#)

Jira: Count of Epics, User Stories, Tasks, and Issues: [165 issues](#)

Status (5-25-21):

D Release status

- New xApps:
 - Bouncer xApp (HCL, C++): RIC performance measurement xApp - in conjunction with the appropriate E2 Sim, can test E2 control loop latency (INSERT-CONTROL) as well as the scalability of the RIC with regard to the number of E2 Nodes supported.
 - LP (Load Prediction, ChinaMobile, python): Initial version of a cell load predictor.
 - HW-P (Hello World - Python, Samsung): A python based demo xApp that demonstrates how an xApp can use the RIC platform features in python.
 - HW-G (Hello World - go, Samsung): A go-based demo xApp that demonstrates how an xApp can use the RIC platform features in go.
- Improved xApps:
 - AD (Anomaly Detection, HCL, python): A ML-based real-time anomaly detection using KPI data populated in inFlux DB.
 - KPIMON (Samsung, go): Improved version implements E2 SM KPM 2.0.3 version and stored collected data in time series DB (inFlux)
 - QP (QuE Predictor, HCL, python): A ML-based predictor of UE's throughput if it was handed over to a neighboring cell. The D release version finally uses a ML-trained prediction model and includes the functionality previously provided as a separate QP-driver xApp.
 - TS (Traffic Steering, UTFPR, C++): Extended version of the TS xApp that now receives anomaly detection messages, requests QoE prediction, and issue control operation to request a UE handover.
- Together, AD, QP, and TS xApps and Viavi E2 Tester, implement a use case where anomaly detection is combined with QoE prediction and traffic steering action to move the affected UEs to a different cell.

D release source code, container images and deployment instructions

Each repository has a branch named "dawn" that can be accessed using git. For example, the source code for the AD xApp can be retrieved using "git clone --branch dawn "https://gerrit.o-ran-sc.org/r/ric-app/ad". The other xApps in the D release can be found at ric-app/qp, ric-app/ts, ric-app/lp, ric-app/hw, ric-app/hw-go, ric-app/hw-python, ric-app/mc, ric-app/bouncer, and scp/ric-app/kpimon. Note that the other ric-app repos are obsolete.

Note that this branch is in maintenance and all new development is done in branch "master".

In order to deploy the D release xApps, you can re-use the pre-created container images as defined [here](#) and the instructions on testing the xApps can be found [here](#).

Near-Real-time RAN Intelligent Controller Platform (E2 Interface) (RICPLT)

Mission: Update to newer O-RAN specs (E2,A1,O2,O1) and related features.

Original primary goals: Update to E2APv1.1 (E2 Node configuration transfer in E2 Setup and E2 Configuration Update (even if likely changing again in E2APv2.0) and E2SM OID support in internal E2SM function query interfaces) // support for A1-EI (as per A1APv3.0) // support for O2 as per WG6 use cases // support for RIC-708 O1-CM to xApps // RIC-734 Include time series database into RIC platform (InfluxDB) for usage by xApps // RIC-421 O1 mediator graceful restart with O1 data being persisted over restarts // Concrete alarms from RIC platform (related to message overload): RIC-204, RIC-203 // SDK package, well documented interfaces to be used by xApps via xApp frameworks // Portability SDK (in xApp project) // REST interface for subscription management. 35 Epics planned: [link](#) and 11 items as stretch goals: [link](#)

Achieved D release highlights = high-level release notes (2021-06-28) below (note that the release image list is here: [Near-RT RIC \(D release\)](#))

- Some of the features are demoed here: [2021-06-08 Dawn](#). An extract of more fine-grained per component releases notes can be found in the attachment of [\[this page\]](#).
- REST interface for xApps towards E2 subscription manager. No need to encode E2AP subscription messages in the xApps anymore. The xApp framework for Go already supports/uses this.
- Support for A1-EI (Enhancement information) to xApps. The A1 container now uses Ubuntu base image (like all others) and not Alpine anymore.
- SDL: multiple groups of SDL standalone or sentinel instances.
- SDL-py: Pack all the events in a channel to one DB notification to be in line with SDL Golang
- A lot of extra load/scalability testing (using a new bouncer xApp) and functionality testing (E2, ...) was done under RIC-150 using a "bouncer xApp".
- Wider scope of the xapp framework for python related to SDL, xapp registration, RNIB and E2AP handling.
- We added InfluxDB as optional platform service time series database (RIC-734)
- Support for O2 as per WG6 use case "Deploy xAPP in Near-RT RIC" in O-RAN Orchestration Use Cases v2.0. This also includes a change in how xApps register as part of their startup.
- libe2ap (asn1c-based) can be re-used by components to encode/decode E2AP ASN.1 PDUs (Protocol Data Unit)
- E2 statistics are now visible as VES metrics events
- RMR raises alarms using the RIC alarm system in temporary overload situations
- The Near-RT RIC can be deployed on Kubernetes 1.18 and helm 3. For the first time, this and all robot framework based "end-to-end" tests have also been verified in the O-RAN SC lab.
- The Near-RT RIC project now achieved the CII (Core Infrastructure Initiative) badge "passing": ([link](#)).

For the D release of the near-RT RIC we did only limited integration testing: only the use case: deploy RIC, deploy xApp and make E2 connection were tested.

PTL: [Thoralf Czichy](#)

Status 2021-07-05: Work is completed for the following 25 (19 epics and 6 "others") items [link](#). All of these are already "done". The following 24 items (17 epics and 7 "others") we had to move out from Dawn content [link](#). None of the stretch goals ([link](#)) has been worked on. See release highlights (above) for what has been achieved. Most notable items that were dropped are support for the E2APv1.1 capabilities "config transfer" and "OID support" (i.e., we continue with E2APv1.0), RIC-708 O1-CM to xApps. Discussion on the portability SDK is still work in progress. We continue to support all the existing SDKs via the xapp Frameworks for C++, python and go.

Status 2021-03-03: Work started on many items. 31 Epics planned: [link](#) and 15 items as stretch goals: [link](#). Start of release snapshot (MS Excel): [link](#) For CII compliance ([link](#)) we now do some checks every two weeks in the status meeting and have started a [Release criteria checklist template](#) that we go through before releasing, Note that we update to E2APv1.1

D release source code, container images and deployment instructions

Each repository has a branch named "dawn" that can be accessed using git: "git clone --branch dawn "<https://gerrit.o-ran-sc.org/r/ric-plt/e2>". Make sure to replace the URL with correct repositories. Note that this branch is in maintenance and all new development is done in branch "master". The complete list of repositories belonging to the RIC platform is defined here: [Scope of the near-RT RIC platform and its components \(summary\)](#).

In order to deploy the D release of the near-RT RIC platform you can re-use the pre-created container images as defined [here](#). The same instructions as always apply, i.e., follow the general latest instructions: <https://docs.o-ran-sc.org/projects/o-ran-sc-ric-plt-ric-dep/en/latest/> Installing Near Realtime RIC in RIC Cluster, **but** make sure to use "git clone --branch dawn ..." instead of "git clone ..." when cloning it/dep and ric-plt/ric-dep.

Non-Real-time RIC (A1 Interface) (NONRTRIC)

Primary Goals:

- The primary goal of Non-RT RIC is to support intelligent RAN optimization by providing policy-based guidance, ML model management and enrichment information to the near-RT RIC function so that the RAN can optimize, e.g., RRM under certain conditions.
- It can also perform intelligent radio resource management function in non-real-time interval (i.e., greater than 1 second).
- Non-RT RIC can use data analytics and AI/ML training/inference to determine the RAN optimization actions for which it can leverage SMO services such as data collection and provisioning services of the O-RAN nodes.
- Non-RT-RIC will define and coordinate rApps (Non-RT-RIC applications) to perform Non-RT-RIC tasks.
- Non-RT-RIC will also host the new R1 interface (between rApps and SMO/NONRTRIC services)

D Feature Scope:

- NONRTRIC Functions: ([NONRTRIC Release D Wiki](#))
 - Integrated A1 Adapter from ONAP (controller – mediation)
 - Integrated A1 Policy Management Service from ONAP (controller – A1 policies)
 - OSC A1 Enrichment Information Coordinator (controller – A1 EI Job management)
 - OSC Non-RT-RIC Control Panel (GUI – for A1-P & A1-EI Job management)
 - OSC A1 Simulator (a stateful test stub to simulate near-RT-RIC end of A1 interface – A1-P & A1-EI)
 - OSC (initial) APP catalog (for registering/querying APPs)
 - Initial K8S Helm Chart LCM Manager - for APP μServices etc. (ONAP & OSC) (**new**)
 - Initial Service Exposure Function (**new**)
- In D Release: ([NONRTRIC Release D Wiki](#))
 - Improved A1-PMS NBI (REST & DMaaP) (Rest style alignment)
 - Runtime configuration API (REST) for A1 Policy Management Service (e.g. add/remove adapters, near-rt-rics, security certs, etc)
 - Deployment – Continued improvements for Docker & Kubernetes
 - Extended/Easier deployment options with OSC IT/DEP project (SMO/NONRTRIC deployment)
 - Improving CI/CD to support include A1 Policy controller dependencies from ONAP
 - Multi-version support (O-RAN A1-AP v1.1, v2.0, v2.1,v3.0 & OSC pre-spec A1)
 - Improved status monitoring/notification of A1-EI Jobs
 - Further improvement in security cert management (All interfaces can now be secured using TLS)
 - Re-architect & improve usability of Non-RT-RIC Control Panel (GUI)
 - Extend NONRTRIC Control Panel to edit/create A1 Enrichment Types/Jobs
 - Extend NONRTRIC Control Panel to configure A1 Policy Management Service
 - Configurable Service Exposure function – Extends/Replaces static exposure gateway in OSC Cherry
 - K8S Helm Chart LCM function for App μServices
 - Update NONRTRIC demo/test environment (one-click tests/use-cases, docker & single/multi-node K8s env)
 - OSC e2e integration use case – O-RU-FH>HelloWorld recovery
 - App to instigate O-RU-FH connection recovery after failure – via O-DU
 - Multiple implementation options – standalone μService and/or deployable ONAP-PF policy script
 - CII badging – Already achieved Bronze/Passing Grade

More detail:

- For documentation, guides and a full discussion of NONRTRIC D Release see [NONRTRIC: Release D](#)

PTL:

- [John Keeney](#)

Jira:

- Count of Epics ([20 issues](#)), User Stories, Tasks, and Issues: ([455 issues](#))

D release source code, container images and deployment instructions

- Source Code:
 - The source code can be found in the "dawn" branch **nonrtric**, **sim/a1-interface** and **portal/nonrtric-controlpanel** git repositories
 - nonrtric : <https://gerrit.o-ran-sc.org/r/admin/repos/nonrtric>
 - portal/nonrtric-controlpanel : <https://gerrit.o-ran-sc.org/r/admin/repos/portal/nonrtric-controlpanel>
 - sim/a1-interface : <https://gerrit.o-ran-sc.org/r/admin/repos/sim/a1-interface>
 - Detailed instructions to download and build the "D Release" NONRTRIC components can be found at [NONRTRIC - Release D - Build](#)
- Docker Images:
 - A full list of released Docker images can be found at [NONRTRIC - Release D - Run#Images](#)
- Deploy / Run:
 - Step-by-step guides to deploy and run the NONRTRIC components can be found at [NONRTRIC - Release D - Run](#)
 - Instructions to perform an end-to-end test of some of the NONRTRIC functions can be found at [NONRTRIC - Testing End to End call in Release D](#)
 - Alternative deployment options (Docker, Kubernetes, etc) can be found at [NONRTRIC - Operations and Management](#)
 - A full suite of function tests & demos, for all NONRTRIC versions, can be found at [NONRTRIC - Function Test suite](#)
- Documentation:
 - nonrtric : <https://docs.o-ran-sc.org/projects/o-ran-sc-nonrtric/en/dawn/>
 - portal/nonrtric-controlpanel : <https://docs.o-ran-sc.org/projects/o-ran-sc-portal-nonrtric-controlpanel>
 - sim/a1-interface : <https://docs.o-ran-sc.org/projects/o-ran-sc-sim-a1-interface>

Operation and Maintenance (OAM)

Primary Goals:

- updates of OAM related interface definitions based on
 - YANG from WG4 - **O-RAN Management Plane Specification - YANG Models 5.0 - November 2020** (with dependencies to IETF data models)
 - **status:** done
 - YANG from WG5 - **O-RAN O1 Interface for O-DU 1.0 - YANG Models - November 2020** (with dependencies to 3GPP data models)
 - **status:** after feedback of WG5, related merge request was abandoned - instead a pipeline will be established with O-RAN bitbuckets.
 - YANG from WG1 - O1-interface (November 2020 train)
 - **status:** not approved by O-RAN
 - VES from ONAP DCAE - VES 7.2
 - **status:** done
 - Notification syntax from 3GPP TS 28.532 V16.6.0
- support D-Release use case "O-RU recovery"
 - see Feature Scope below
- handover SMO artifacts to new SMO project

D Feature Scope:

- Update to OpenDaylight Silicon
- Support of Callhome via TLS
- CallHome to VES:pnfRegistration
- o-ran-fm.yang/alarm-notif to VES:fault

PTL: [Martin Skorupski](#)

D release source code, container images and deployment instructions (and status)

Jira: Count of Epics ([15 issues](#)), User Stories, Tasks, and Issues: [166 issues](#)

Source Code:

- The source code can be found in the "dawn" branch **oam**, **sim/a1-interface** and **portal/nonrtric-controlpanel** git repositories
 - oam : <https://gerrit.o-ran-sc.org/r/gitweb?p=oam.git;a=shortlog;h=refs%2Fheads%2Fdawn>
 - oam/nf-oam-adopter : <https://gerrit.o-ran-sc.org/r/gitweb?p=oam%2Fnf-oam-adopter.git;a=shortlog;h=refs%2Fheads%2Fdawn>
 - oam/tr069-adapter : <https://gerrit.o-ran-sc.org/r/gitweb?p=oam%2Ftr069-adapter.git;a=shortlog;h=refs%2Fheads%2Fdawn>

Integration:

- [D-Release Integration](#) (and its sub pages)
- [Test Topology](#)

Release notes:

- in DOCs: <https://docs.o-ran-sc.org/projects/o-ran-sc-oam/en/latest/release-notes.html>
- in JIRA: <https://jira.o-ran-sc.org/secure/ReleaseNote.jspa?projectId=10203&version=10500>

Please see: [Use Case Flow tests for D-Release](#)

O-RAN Central Unit (OCU)

Primary Goals:

- In the absence of O-CU, Radisys commercial CU image to be used for E2E testing

D Release Feature Scope:

- Radisys Commercial CU being used as a test fixture for E2E testing

PTL:

Status:

Radisys Commercial CU being used as a test fixture.

H/W and S/W requirements have been shared and awaiting the same to be configured.

D release source code, container images and deployment instructions

not applicable

O-DU High

Primary Goals:

- O-DU high integration in Radio mode
- Initial access and Attach procedure testing in Radio mode
- DL and UL data path in FDD, 20 MHZ with 256 QAM and 64 QAM respectively
- Static TDD mode support with Numerology =1
- O1 enhancement
- Closed Loop Automation Use case support

D Feature Scope:

1. Achieve UL and DL data flow using FDD mode on 20 MHz Bandwidth, Numerology = 0
2. Support for static TDD mode with pattern "DDDDDDDSUU" on 100 MHz Bandwidth, Numerology = 1
 - Evolve scheduler to support UL and DL scheduling of signaling and data messages on single spectrum in TDD mode
 - Expand scheduler to support Frame structure according to numerology = 1
 - Updates to cell broadcast for TDD and numerology = 1
3. Development activity for Closed Loop Automation use-case
 - Support for cell stop and restart within O-DU High layers
 - Support for cell stop and restart towards O-DU Low
 - F1AP Enhancements towards O-CU indicating cell stop and restart
4. Integration
 - Integration with O-DU Low in Radio mode
 - Integration with CU
5. End to end testing support (O-RU<->O-DU-LOW<->O-DU-HIGH<->RYS CU<->Viavi 5G Core)
6. O1 enhancements - by HCL
 - Re-structure O1 module to run as a thread in ODU-High
 - CM Support - IP and Port configuration for DU, CU stub and RIC stub via Netconf interface
 - Support for Closed Loop Automation use-case

PTL: [Sachin Srivastava](#)

Status:

Updated: 7th July 2021

JIRA: Epics Status below:

- <https://jira.o-ran-sc.org/browse/ODUHIGH-184> - Done
 - As an O-DU L2 developer, I want to implement single UE DL data path and bench-marking
- <https://jira.o-ran-sc.org/browse/ODUHIGH-185> - Done
 - As an O-DU L2 developer, I want to implement single UE UL data path and bench-marking
- <https://jira.o-ran-sc.org/browse/ODUHIGH-186> - Done
 - As an O-DU L2 developer, I want to add support for 64QAM modulation scheme in DL
 - Basic code changes complete. Testing in progress for data path
- <https://jira.o-ran-sc.org/browse/ODUHIGH-187> - Done
 - As an O-DU L2 developer, I want to add support for 16QAM modulation scheme in UL
 - Basic code changes complete. Testing to be done for data path
- <https://jira.o-ran-sc.org/browse/ODUHIGH-264> - Done
 - As an O-DU L2 developer, I want to add support for Mu1
 - Code changes at DU APP completed.
 - Resource allocation for SSB to msg5 completed
 - Code changes for UE registration flow in progress
 - Updates to k0, k1, k2 in progress
- <https://jira.o-ran-sc.org/browse/ODUHIGH-265> - Done
 - As an O-DU L2 developer, I want to add support for 100 MHz Bandwidth
 - Code changes at DU APP completed.
 - Resource allocation for SSB to msg5 completed
 - Code changes for UE registration flow in progress
 - Updates to k0, k1, k2 in progress will be continued in E release
- <https://jira.o-ran-sc.org/browse/ODUHIGH-266> - Done
 - As an O-DU L2 developer, I want to add support for TDD mode
 - Code changes at DU APP completed.
 - Resource allocation for SSB to msg5 completed
 - Code changes for UE registration flow in progress
 - Updates to k0, k1, k2 in progress will be continued in E release (Irrespective of FDD or TDD stack)
- <https://jira.o-ran-sc.org/browse/ODUHIGH-299> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case
 - Yang modules to be supported by O-DU to ensure the end-to-end functionality of the use case "Closed loop" is in progress. Basic configuration is agreed to support CLA use case.
 - Internal call flow/message sequence between O-CU and O-DU for cell activation and deactivation is clarified. Call flow updated at <https://wiki.o-ran-sc.org/display/RSAC/Closed+Loop+Automation+Call+Flow+-+O-DU+High+APIs>.
 - UE delete functionality complete
 - Cell delete functionality complete
 - Issue with mis-coordination between cell delete and DL RRC message, resolved.
 - Code changes for CU Interaction is completed
 - Code changes for Config update over F1 interface is completed
 - O1 Integration for O-DU for CLA is completed (Cell stop and Cell restart)
 - Blocker : code segmentation is observed, analysis is going on (code optimization is required to be scoped in E release)
- <https://jira.o-ran-sc.org/browse/ODUHIGH-267> - Done
 - As an O-DU L2 developer, I want to integrate O-DU High with O-DU Low in Radio Mode
 - SSB transmission successful
 - Debugging issue with Sib1 transmission , PDCCH is received but no PDSCH seen at O-DU low.
 - PDSCH for SIB1 is detected at L1 but L1 does not process it. Pointer is to check the PDSCH PDU parameters
 - Further debug sessions needed to close the ongoing issues.
 - There is no breakthrough even after several debug sessions with O-DU Low
 - SIB1 detection at L1 is successful. PHY.XML is updated with removing the hardware accelerator (<dpdkBasebandFecMode> from 1 to 0 to force the SW encoder)
 - For the CLA usecase, Cell stop request is received from O-DU high to low but O-DU low sends stop indication multiple times. This issue is fixed in L1 later binary 20.08. This binary update will happen in D-maintenance phase.
- <https://jira.o-ran-sc.org/browse/ODUHIGH-268> - Done
 - As an O-DU L2 developer, I want to integrate O-DU High with O-CU
 - Using Radisys commercial CU as a test fixture
 - New VM configured as per H/W and S/W requirements of Radisys CU
 - The Network interfaces and CentOS version needs to be revisited for the CU machine. This is achieved with limited OSC lab setup.
- <https://jira.o-ran-sc.org/browse/ODUHIGH-269> - Done
 - As an O-DU L2 developer, I want to support End to End testing scenarios
 - Testing of broadcast messages at O-RU emulator set to begin
 - Viavi confirmed receiving at O-RU. Needs verification from UE sim.
 - Debug session is planned on 23rd June to achieve SSB and SIB1 transmission till UE simulator and then follow with RACH procedure.
 - Latest issue: the eCPRI packets differentiation between control plane and user plane through vlan id is supported by Intel, however O-RU support the packet differentiation based on eCPRI packet type. hence the fronthaul transmission validation is blocked.
 - Intel shall update the L1 package supporting C/U plane differentiation using eCPRI packet type in the D-release maintenance phase.

Updates from HCL:

- <https://jira.o-ran-sc.org/browse/ODUHIGH-247> - Done
 - As an O-DU L2 developer, I want to Establish Netconf session for O1 interface for CM
- <https://jira.o-ran-sc.org/browse/ODUHIGH-297> - Done
 - As an O-DU L2 developer, I want to Establish Netconf session for O1 interface for CM
- <https://jira.o-ran-sc.org/browse/ODUHIGH-322> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case
- <https://jira.o-ran-sc.org/browse/ODUHIGH-327> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case
- <https://jira.o-ran-sc.org/browse/ODUHIGH-328> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case
- <https://jira.o-ran-sc.org/browse/ODUHIGH-347> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case
 - Tested locally with SMO both with IPv4 and IPv6. Able to mount and connect ODU.
 - Integration with SMO in working fine with IPv4 but with IPv6 could not tested as it is not enabled in OSC lab
 - Integration with O-DU code for cell down and up scenario is completed, validation is completed
 - Edit-config testing from SMO to ODU for cell down/up is in completed in OSC lab
- <https://jira.o-ran-sc.org/browse/ODUHIGH-349> - Done
 - As an O-DU L2 developer, I want to develop O-DU High Layers to support Closed Loop Automation Use-case

Dependency/Blockers:

- O1 configuration for day-1 shall need to be completed to start with CLA. However basic configuration e.g. cell state/operational state/admin state shall be supported initially. Use admin state as unlocked to validate the RU link failure.
- Server(VM) configuration (H/W and S/W) to mount Radisys CU as a test fixture.
- Unable to use valgrind with Intel libraries. Debugging must be carried out with Alternate methods.
- Intel/Viavi to confirm successful decoding of SSB/SIB1 at UE sim (TM500).

D release source code, container images and deployment instructions

[Release-Notes](#) — [o-du-l2 master documentation \(o-ran-sc.org\)](#)

O-DU Low

Primary Goals:

- Continue O-DU low and O-DU high pairwise test.
 - FAP P7 message integration -> Ongoing
- Continue O-DU Low and O-RU emulator test.
 - Further CU plane testing -> Ongoing
- Continue E2E test with UE simulator.
 - Support the UE attachment test
- Development activity for Closed Loop Automation use-case
 - Support and test for cell stop and restart within O-DU High layers

D Release Feature Scope:

PTL: @Zhimin Yuan

- Status

D release source code, container images and deployment instructions

TODO

Simulators (SIM)

Primary Goals:

- Support rapid prototyping by providing simulated interfaces

D Feature Scope:

- Enable "Closed Loop Use Case" demonstration by providing O1 interface Simulators for:
 - O-DU (containing o-du-hello-world YANG model)
 - O-RU (containing O-RAN-RU-FH November 2020 train YANG models)
- O1 Simulator improvements:
 - "Blank" simulator, which allows dynamically loading any YANG models of interest, for simulating a NETCONF/YANG interface

PTL: [Alex Stancu](#)

Jira: Count of Epics, User Stories, Tasks, and Issues: [5 issues](#)

Status:

- O-DU O1 Simulator docker image available in Nexus (version 1.3.3)
- O-RU O1 Simulator docker image available in Nexus (version 1.3.3)
 - Example docker-compose for deploying O-DU and O-RU simulators: "[D](#)" [Release - Closed Loop Use Case: Simulation of O-RU and O-DU](#)
- Blank O1 Simulator docker image available in Nexus (version 1.3.3)

D release source code, container images and deployment instructions

Docker container images are described [here](#).

Each repository has a branch named "dawn" that can be accessed using git: "git clone --branch dawn "https://gerrit.o-ran-sc.org/r/sim/o1-interface". Make sure to replace the URL with correct repositories. Note that this branch is in maintenance and all new development is done in branch "master". The complete list of repositories belonging to the SIM project is [here](#).

Deployment instructions [here](#).

Please see: [Use Case Flow tests for D-Release](#)

Infrastructure (INF)

Primary Goals:

- Implement the O-Cloud reference design, provide the real time performance to allow the O-DU and other components running on top of it.
- Provide interaction capabilities with other components.

D release Feature Scope:

- Enable the 2 AIO servers with additional worker nodes deployment scenario
- Major components upgrade
- Implement the O2 interface as the MVP (will defer to next release)

PTL: [Jackie Huang](#)

Jira: Count of Epics, User Stories, Tasks, and Issues:

Update at 05 Aug 2021

- [INF-172](#) - Getting issue details... [STATUS](#)
- [INF-214](#) - Getting issue details... [STATUS](#)

Test:

- [Getting Started /Sample test process](#)

Release Note:

- <https://docs.o-ran-sc.org/projects/o-ran-sc-pti-rtp/en/latest/release-notes.html#version-4-0-0-2021-06-29>

D release source code, container images and deployment instructions

- The repository has a branch named "dawn" that can be accessed using git: "git clone --branch dawn "<https://gerrit.o-ran-sc.org/r/pti/rtp.git>". Note that this branch is in maintenance and all new development is done in branch "master".
- No container image for INF project
- Deployment instruction: [Getting Start \(verification on Dell R740\) /Sample test process](#)

Integration and Test (INT)

Primary Goals: To support OSC project CI pipeline. To test and validate the components and use cases

Cherry Feature Scope:

- Automated CLM and SonarQube Scanning CI Jobs
- Improve CI for OSC projects
- Validate and and Test platform and use cases

PTL: [Zhe Huang](#)

Jira: Count of Epics, User Stories, Tasks, and Issues: [54 issues](#)

D release source code, container images and deployment instructions

not applicable

Documentation (DOC)

Primary Goals: TODO

Bronze Feature Scope: TODO

PTL: [weichen ni](#)

Jira: Count of Epics, User Stories, Tasks, and Issues:

D release source code, container images and deployment instructions

not applicable

Service Management and Orchestration (SMO)

Primary Goals: The primary goal of the SMO project is to integrate different software artifacts of existing open-source projects creating a fully functional open-source Service Management and Orchestration (SMO).

D Feature Scope:

- Support for O1 interface
 - Implementation of NETCONF client in SMO
 - Reference implementation of a NETCONF server that O-RAN Network Functions, e.g. Near-RT RIC, CU, DU and RU can use. The code can be found at <https://github.com/CESNET/netopeer2>
 - A minimal set of YANG models that demonstrate the capability of the O1 interface while satisfying the closed-loop automation use-case.
- Support for O1/VES interface
 - Demonstrate the capability to receive VES events, collect them in a dB, and display them in a dashboard.

PTL: [Mahesh Jethanandani](#)

Status:

- An implementation of the O1 interface has been checked into Gerrit. Check out [this](#) repo. It has been tested on Ubuntu Linux version 20.04. Feedback is appreciated on other versions and operating systems. Note, this commit is not feature compatible with the O1 interface in other implementations. Some of those features have been identified and marked as enhancements in either this or the next release.
- An implementation of the VES interface based on schema version 7.2.1, with backward compatibility to 7.0, has been submitted for Gerrit review, and review comments have been provided. Author has updated the commit based on the comments. Waiting on more reviews. Again, the commit is not feature compatible with VES interface in other implementations. Some of those features have been identified and will be added in this or the next release.

Jira: Count of Epics ([0 issues](#)), User Stories, Tasks, and Issues: [6 issues](#)

D release source code, container images and deployment instructions

Docker image and instruction on how to install SMO O1 interface can be found [here](#).

Docker image for instructions on how to install SMO O1/VES interface can be found [here](#).

The repository has a branch named "dawn" that can be accessed using git. For the O1 interface the repository can be found at "git clone --branch dawn " <https://gerrit.o-ran-sc.org/r/smo/o1mo/o1>", whereas the O1/VES repository can be had by "git clone --branch dawn " <https://gerrit.o-ran-sc.org/r/smo/vs>"