Developer session
At the Open Source Conference 2021
5G, IoT and Edge computing

The O-RAN Alliance and the ORAN-SC Project

thoralf.czichy@nokia.com
PTL, near-RT RIC

Jul-03-2021
The O-RAN alliance was formed to increase competition
RAN openness, hardware vs software separation, programmability

**Key Objectives:**
- Bring Cloud Scale Economies to RAN
- Bring Agility to RAN

**Key Principles:**
- Lead the industry towards open, interoperable interfaces, RAN virtualization, and big data enabled RAN intelligence
- Maximize the use of common-off-the-shelf hardware and minimizing proprietary hardware
- Specify APIs and interfaces, driving standards to adopt them as appropriate, and exploring open source where appropriate

Source: https://www.o-ran.org/membership
The working groups of the O-RAN alliance

1. Non-real-time RIC and A1
2. Near-real-time RIC and E2AP plus E2SM (service models) for RC, KPM, NI
3. Open Fronthaul eCPRI 7.2x split
4. Open F1/W1/E1/X2/Xn (re-use of 3GPP specs)
5. Cloudification and Orchestration (O1 (3GPP reuse), O2)
6. Stack Reference Design
7. White-Box hardware
8. Open X-haul Transport
9. Use cases & overall architecture
O-RAN-SC – The O-RAN software community

Est. April 2019

- Project by the O-RAN Alliance and the Linux Foundation (LF)
- Open-source software aligned with the architecture specified by the O-RAN alliance
- Re-using Series of LF Projects, LLC (common “master LLC”, but separate divisions with limited liabilities).
- LF also provides project infrastructure

>20 companies

- Source code contributions from >20 companies
- 92% of the commits by the top 10 contributing companies
- License: Apache 2.0

4 releases

- A release every 6 months (in July and December):
  - Amber (Nov-2019)
  - Bronze (Jun-2020)
  - Cherry (Dec-2020)
  - Dawn (Jul-2021)
  - E ...
  - F ...
- [https://wiki.o-ran-sc.org/display/REL/Releases](https://wiki.o-ran-sc.org/display/REL/Releases)

Tue + Wed

- Day-to-day management via the TOC (Technical oversight committee)
- Weekly meetings on Wednesdays, 5:30pm IST
- [https://wiki.o-ran-sc.org/display/TOC](https://wiki.o-ran-sc.org/display/TOC)
- Subprojects have own meeting practice
- For example, near-RT RIC has fortnightly meetings on Tuesdays, 6:30pm IST
- [https://wiki.o-ran-sc.org/display/RICP/Project+meetings](https://wiki.o-ran-sc.org/display/RICP/Project+meetings)

SCP

- The O-RAN Specification Code Project has separate charter, but delegates administration to the O-RAN-SC TOC
- Shares infrastructure and meetings with O-RAN-SC
- License: O-RAN software license
### O-RAN-SC subprojects

<table>
<thead>
<tr>
<th>Project</th>
<th>PTL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RICAPP</td>
<td>Matti Hiltunen AT&amp;T</td>
<td>Near-RT RIC XApps</td>
</tr>
<tr>
<td>RIC</td>
<td>Thoralf Czichy Nokia</td>
<td>Near-RT RIC platform</td>
</tr>
<tr>
<td>OCU</td>
<td>Suzy Gu CMCC</td>
<td>Implementing L2, F1, MAC scheduler, RLC</td>
</tr>
<tr>
<td>ODUHIGH</td>
<td>Sachin Srivastava Radisys</td>
<td>Implementing L1, Intel FlexRAN binary used</td>
</tr>
<tr>
<td>ODULOW</td>
<td>Zhimin Yuan Intel</td>
<td>O-RAN Radio Unit</td>
</tr>
<tr>
<td>ORU</td>
<td>TBD NA</td>
<td>No contributions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project</th>
<th>PTL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OAM</td>
<td>Martin Skorupski HST</td>
<td>yang models, RIC dashboard, O1 reference impl. + O1 client</td>
</tr>
<tr>
<td>SIM</td>
<td>Alex Stancu HST</td>
<td>Simulators for testing O-RAN, e.g., E2AP</td>
</tr>
<tr>
<td>INF</td>
<td>Xiahua Zhang Windriver</td>
<td>Cloud infra for O-RAN (WR)</td>
</tr>
<tr>
<td>INT</td>
<td>&lt;open&gt;</td>
<td>Integration of O-RAN SC release</td>
</tr>
<tr>
<td>DOC</td>
<td>Weichen Ni CMCC</td>
<td>Documentation to readthedocs.io</td>
</tr>
<tr>
<td>NONRTRIC</td>
<td>John Keeney Ericsson</td>
<td>A1 policy mgr (used by rApps), r-app host, Enrichment data, (ML mgmt)</td>
</tr>
<tr>
<td>SMO</td>
<td>Mahesh Jethanandani Juniper</td>
<td>Integrated SMO &amp; deployment scenarios with ONAP</td>
</tr>
<tr>
<td>SMO</td>
<td>Mahesh Jethanandani Juniper</td>
<td>Integrated SMO &amp; deployment scenarios with ONAP</td>
</tr>
</tbody>
</table>

**Description**
- **Near-RT RIC XApps**: Implementations for SDAP and PDCP, eGTP-U as binary.
- **Near-RT RIC platform**: Implementing L2, F1, MAC scheduler, RLC.
- **O-RAN Radio Unit**: Implementing L1, Intel FlexRAN binary used.
- **O-RAN Radio Unit**: No contributions.
Licenses of O-RAN-SC and the Specification code project

- The Specification code project shares infrastructure and meetings with O-RAN-SC
- We give repositories within subprojects the choice between contributions under Apache 2 license or under ORAN Software license, also referred to as SCP (Specification code project)
- Documentation to be contributed under Creative Commons Attribution 4.0 (CC BY 4.0)
- The Apache license is a very liberal license in terms of being able to use the source code, e.g., it includes an explicit patent license
  - For example, the Near-RT RIC subproject uses this license.
- ORAN Software license, also referred to as SCP (Specification code project) license is used in the specification code project
  - Details of the license: https://www.o-ran.org/software
  - Used for some xApps and OCU repo. Repos always in folder ”scp” in gerrit
- No contributions without an online-signed contributor license agreement (CLA)
  - https://wiki.o-ran-sc.org/display/ORAN/Signing+Contributor+License+Agreement
- Minor related contribution to asn1c
  - asn1c fork: https://github.com/nokia/asn1c (BSD2/3) - minor adaptations to make it work with the O-RAN ASN.1 specifications.
# Committers

## Review
- Committers are the primary contact for a component, they review code contributions

## INFO.yaml
- Each repo has a set of committers. Check from the repo’s INFO.yaml file (base directory) of the repo.
- Example: https://gerrit.o-ran-sc.org/r/gitweb?p=ric-plt/e2.git;a/blob;f=INFO.yaml

## Merge
- Only committers can merge source code to master or maintenance branches
- Only committers can release

## New committer?
- Committers can change by majority vote (>=50%) of existing committers.
- Change is automatically notified post-fact to the O-RAN SC TOC (Technical oversight committee)
### Tools re-use from Linux Foundation infrastructure

**gerrit: All source code + review tool**
- [https://gerrit.o-ran-sc.org/r/#/admin/projects/](https://gerrit.o-ran-sc.org/r/#/admin/projects/)

**nexus3 as image repository. Release (port 10002) and Staging registry (10004)**
- Images in staging registry automatically deleted. Prefer use of release registry instead
- Also includes container base images: [https://wiki.o-ran-sc.org/display/ORAN/ORAN+Base+Docker+Images+for+CI+Builds](https://wiki.o-ran-sc.org/display/ORAN/ORAN+Base+Docker+Images+for+CI+Builds)

**packagecloud.io for binary artifacts, such as RPM and debian packages.**
- Master and staging used during development
- Packaging [https://wiki.o-ran-sc.org/display/ORAN/Package+Libraries+for+Linux+Distributions+with+CMake](https://wiki.o-ran-sc.org/display/ORAN/Package+Libraries+for+Linux+Distributions+with+CMake)
- Publishing: [https://wiki.o-ran-sc.org/display/ORAN/Binary+Repositories+at+PackageCloud.io](https://wiki.o-ran-sc.org/display/ORAN/Binary+Repositories+at+PackageCloud.io)

**NexusIQ (hosted by LF) for license checks**
- [https://nexus-iq.wl.linuxfoundation.org](https://nexus-iq.wl.linuxfoundation.org) (access limited) // Right now only A1 mediator. Checks done at end of release manually.

**sonarcloud.io: static code checking and code coverage (via tests)**
- All repos: [https://sonarcloud.io/organizations/o-ran-sc/projects](https://sonarcloud.io/organizations/o-ran-sc/projects)
- We aim for 80% code coverage

**readthedocs.io: automatically generated documentation**
- Results: [https://docs.o-ran-sc.org/en/latest/projects.html#near-realtime-ran-intelligent-controller-ric](https://docs.o-ran-sc.org/en/latest/projects.html#near-realtime-ran-intelligent-controller-ric)
- Instructions: [https://wiki.o-ran-sc.org/display/ORAN/Configure+Repo+for+Documentation](https://wiki.o-ran-sc.org/display/ORAN/Configure+Repo+for+Documentation)

**Testing**
- Robot framework used in test cases of E2 manager, routing manager and integration test
- Unit tests: cgreen, gtest, ...

**More details**
[https://wiki.o-ran-sc.org/display/ORAN/O-RAN+Developer%27s+Guide+to+CI+Resources+and+Processes+at+the+LF](https://wiki.o-ran-sc.org/display/ORAN/O-RAN+Developer%27s+Guide+to+CI+Resources+and+Processes+at+the+LF)
What is the near-RT RIC: architecture and key requirements

**RAN Intelligent Controller**

**Use cases**
- Network Intelligence
- Policy Enforcement
- Handover Management
- Resource Control
- Load Balancing
- Radio-Link Management
- Advanced SON
- RAN Slicing

**Network Intelligence**
- Possibly co-localized with CU
- Managing hundreds of DU’s
- Host 3rd Party apps
- Empowered with AI/ML
- Near-RT capabilities
- Role in network slicing.

** EMS functions**
- Orchestration
- Inventory
- Telemetry
- RAN analytics (non-RT RIC)
- Policy
- Control

**Near-RT RAN Intelligent Controller**
- Radio Connection Optimization (xApps)
- RRM optimization (xApps)
- Mobility optimization (xApps)
- Slicing optim. (xApps)
- 3rd party Applications

**Container platform (Akraino REC, WR)**

**Real-time**
- E2

**Multi-RAT**
- CU-CP
  - RRC
  - PDCP-C
- 3GPP E1
- CU-UP
  - SDAP
  - PDCP-U

**NVFI Platform: virtualization & COTS platform**

**ORAN eCPRI 7.2x**

**DU**
- RLC/MAC/PHY-High
- RRU
- PHY-Low/RF

**ORAN Alliance**

**Public**
E2AP functional procedures

**SUBSCRIPTION(REPORT)**
1. Detect Trigger
2. INDICATION (Subscription, Report)
3. Continue call processing

**INDICATION (optional)**
1. Detect EVENT
2. Perform Action
3. CONTROL (Event, State)
4. Resume or initiate call processing
5. CONTROL ACK

**SUBSCRIPTION(POLICY)**
1. Detect Trigger
2. Perform POLICY
3. Continue call processing

**SUBSCRIPTION(INSERT)**
1. Detect Trigger
2. INDICATION (Subscription, UE, Insert)
3. Halt or suspend call processing
The near-RT RIC platform components
E2 principles

• Amber implemented a pre-spec version of E2AP. Bronze, Cherry and Dawn implement E2AP version v01.00. E2AP version 2.0 is currently in the O-RAN approval cycle.

• The RIC E2AP (Application protocol) specification (ORAN-WG3.E2AP-v01.00.00) defines the general protocol by which the near-RT RIC and RAN (gNB, eNB, CU-{CP,UP}, DU) communicate.

• More detailed E2SM (Service model) specifications define the function-specific protocol that is implemented on top of the E2AP specification. Typical functions are X2AP, F1AP, E1AP, S1AP, NGAP interfaces and RAN internal functions UE, Cell, Node.
  • For example, while the E2 specification defines the concept of event triggers, it is the E2SM for NI that defines the specific triggers in the X2/F1/E1/… function based on matching X2/F1/E1/… AP message type, or X2/F1/E1/… IE.
  • E2SMs are an agreement between xApp and E2SM function on E2 node. To the RIC platform E2SMs are opaque.
  • The implementation of the E2 service model on gNB side requires explicit feature development on O-RAN CU/DU side.