OpenAirInterface
Continuous Integration Training

Raphael Defosseux
December 11th, 2018

5G software alliance for democratising wireless innovation
A little bit about me

• 20+ years of experience in developing HW and SW code
• 13 years in real SW dev as
  • Coder in C/C++/Java/Android/any scripting language
  • Integrator and maintainer for industrial products
  • Negotiator of features/improvements with customers

• Joined OpenAirInterface Software Alliance on April 3\textsuperscript{rd} 2018
  • As “Software Manager” \(\rightarrow\) Continuous Integration and Methodology expert
  • Still new to the Open Source world as a contributor
About this training session

• This session is an extension of the training done during the Beijing Workshop in June by Kentaro Sawada from FUJITSU
  • His presentation is available on the OAI web-site:
    • [http://www.openairinterface.org/?page_id=2816](http://www.openairinterface.org/?page_id=2816)

• What was presented in June is now a building brick of the whole CI process
  • Improvements have been made to the different scripts and scenarios

• Continuous Integration is a living animal like the OAI code
  • Some points in this presentation might quickly become obsolete
Introduction to CI

CI = Continuous Integration
CD = Continuous Deployment
OpenAirInterface Software Alliance Goals

• Provide a clean, working software base
  • Functional with a feature list
  • Verified with test-cases and coverage

• Welcome new contributors
  • Ease of use

• Make life easier for
  • Contributors
  • Integrators
Why do we do Continuous Integration?

• Making sure existing features are always tested
  • That their performances are not degraded
• Making sure new feature introductions do not break existing features
  • In addition new tests have to be added to keep testing these new features
• Making sure new contributions meet quality standards
• Making integrator life easier
  • Focus is now on the code review and not validating the added functionality
Code Quality Definition

• Code shall be:
  • “Beautiful”  ➔ Does code follow coding guidelines?
  • “Correct”  ➔ Does code compile and won’t crash?
  • “Functional”  ➔ Is code tested against known pattern?
  • “Fully Tested”  ➔ Are all code lines reached?

• Code Quality Indicators shall be
  • Measured by automated tools such as Continuous Integration
  • Logged as proof -- Traceability
Testing

• Test pattern / scripts written by the feature developer is nice
• But it does not prevent misunderstanding on the feature implementation

• 3rd party tool / equipment is better used to test part of OAI code
OAI Code vs 3rd party

- LTEBOX
- OAI CN
- RAN Emulator
- OAI eNB / gNB
- OAI FLEXRAN CTL
- OAI CN CI
- OAI RAN CI
- COTS-UE
- OAI UE
In the Future: OAI Continuous Deployment

ORCHESTRATION

OAI CN

OAI eNB / gNB

OAI FLEXRAN CTL

COTS-UE

OAI UE
RAN vs CN CI Status

• Fully integrated to GitLab Merge-Request process
• CI scripts embedded in repo
• Formatting Rule Check
• Static Code Analysis
• Variant Builds on several OS
• Testing w/ simulator and RF-board
• Feedback to contributor by email

• Integration with GitHub done
  • But not with Pull Requests
• CI scripts in a private separate repo
• Build and Configuration of vNF
  • On separate Virtual Machines
• Minimal feedback to GitHub
• No test integration w/ NG4T RAN emulator yet

• Will be the focus during Q1 2019
Focus on the RAN CI

It is currently the most advanced CI process

But the principles are the same for the CN CI
Where does it start? The GITLAB repository

All CI scripts / files are located here

CI Status Notification

To contribute, reach us at contact@openairinterface.org
A Push vs a Merge Request

- eNB RAN CI jobs will be **automatically** triggered when:
  - A push to **develop** branch occurs
    - Power users like myself (on the previous slide) do a straight push
    - When a (several) merge request(s) is (are) merged into **develop** branch
  - A Merge Request to **develop** branch is opened, edited, updated
    - At creation or after edition (for example, assigning someone)
    - Mainly when you are pushing again to the source branch
WIP or not WIP? That is the question

• **WIP**: Work In Progress

• When creating a Merge Request, you can tag it as **WIP**
  - WIP Merge Requests **WILL NOT** trigger CI jobs

• **For CI to process, the Merge Request has to be** **MERGE-ABLE**.
  - As a matter of fact, it is the first check performed by CI pipeline script
  - At least the MR should be merge-able at Creation **(without the WIP tag)**
    - If the MR review is taking a long time, it might become **un-merge-able** with conflicts due to other MR merges

Don’t forget to select box here in order to remove the source branch after the merge!
GitLab CI Notifications

Click on Any to access directly the Jenkins job URL

The Pipeline page opens with all the intermediate notifications
GitLab CI Notifications on a MR (1)

CI ran 9 times on this MR
GitLab CI Notifications on a MR (2)

CI is using my credentials

Contributor is making a new push
-> CI is starting again

I took another currently opened MR (#432)

OAI Formatting Check

Compilation Warnings are located on the files modified by the MR
You may or may not be responsible for them

Click Here to access Jenkins master job web-page
Notifications on Slack Channel

You can DM me

I made a few mistakes developing CI scripts

Join us at https://openairinterface.slack.com/
Email to the committer

I was the committer

Correction to make: if MR author is not the committer, put author in cc

Job Summary -- Job: eNb-CI -- Build-ID: 503

Was it a Push or a Merge Request?

All HTML reports
eNB CI Jobs Architecture

Web-hooks are triggered by
• Merge-Requests
• Pushes to develop branch

Visible from the World Wide Web

Currently we are mainly testing the eNB

Web-hooks are triggered by master job
Triggers will carry some parameters

TDD Monolithic Slave Job
FDD Monolithic Slave Job
FDD IF4p5 Slave Job

Open5glab.eurecom.fr

OAI Continuous Integration Training
Master eNB CI Job

Where the CI starts
OS / Platform Flavors

• Currently most of our CI build / testing is done under
  • x86 – Ubuntu 16.04 LTS (xenial)
  • We are building a single variant on a remote CentOS 7.4 server

• I would like to have dedicated eNB master pipeline jobs for
  • x86 – Ubuntu 16.04 LTS (xenial)
  • x86 – Ubuntu 18.04 LTS (bionic)
  • x86 – CentOS 7.4 or 7.6
Clean Containers

• The purpose is to always from a clean sheet

• Currently we are creating “quickly” little Virtual Machines
  • We are using uvt-kvm package for ease of use

• Each Virtual Machine (VM) has to reinstall tools for the variant build
  • Checking if configuration is not missing from the GITLAB repository

• We (CI team) are investigating using Docker instances to speed up process
Deported Processing

Web-hooks are triggered by
- Merge-Requests
- Pushes to develop branch

Visible from the World Wide Web

Node is a Jenkins terminology == worker

Pipeline executes on

nodea.eurecom.fr

open5glab.eurecom.fr

eNb-CI Master Job

Guideline checks
- Static Code Analysis (cppcheck)

All common variants are built
Tests without any RF HW
- Unit-level simulators
- Full stack simulators
- Ethernet transport

Eurecom private network

OAI Continuous Integration Training
eNB Master Job Pipeline

- **Stage where formatting is checked**
  - Currently non blocking

- **Stages where VM are created and builds started**
  - -> It is sequential so host server won't crash

- **Build Stage on a remote CENTOS Server (not a VM)**

- **If Any of Build Stages fails, no Testing performed**

- **Not every stage is a GitLab intermediate notification**

- **Slave Jobs**

Still the eNB CI job #503!
## Variants Builds

<table>
<thead>
<tr>
<th>Variant Name</th>
<th>Build Options (Default 3GPP-Release is Release 14)</th>
</tr>
</thead>
<tbody>
<tr>
<td>enb-usrp</td>
<td>build_oai --eNB -w USRP --mu</td>
</tr>
<tr>
<td>basic-sim</td>
<td>build_oai --basic-simulator</td>
</tr>
<tr>
<td>phy-sim</td>
<td>build_oai --phy_simulators</td>
</tr>
<tr>
<td>enb-ethernet</td>
<td>build_oai --eNB -t ETHERNET --noS1</td>
</tr>
<tr>
<td>ue-ethernet</td>
<td>build_oai --UE -t ETHERNET --noS1</td>
</tr>
<tr>
<td>cppcheck</td>
<td>cppcheck --enable=warning --force --xml --xml-version=2</td>
</tr>
</tbody>
</table>

- I know “build_oai” has a lot of options
- But we cannot check every option

- If you think a variant is missing and is important to developers / users, let us know
  - We could add a variant in the pipeline
What is tested in eNB Master Job?

• Here we are performing **RF-less** testing

• The “physical simulators” unitary tests are testing L1 physical layer

• The “basic simulator” allows to connect a eNB and a single UE through a tunnel
Basic Simulator Testing

Once the connection between all components is complete
• ping UE from EPC VM
• iperf DL from EPC VM to UE
• iperf UL from UE to EPC VM

If the FLEXRAN Controller installation is complete
• Status queries
• Configuration changes
HTML Reporting on builds

Job Summary -- Job: eNb-CI -- Build-ID: 505

Build Summary

OAI Coding / Formatting Guidelines Check
- 55 modified files in Merge-Request DO NOT follow OAI rules. 

Ubuntu 16.04 LTS -- Summary

OAI Static Code Analysis with CPPCHECK
- CPPCHECK found 193 errors and 744 warnings.

OAI Build eNB -- USRP option
- BUILD was SUCCESSFUL.

OAI Build UE -- ETHERNET transport option
- BUILD was SUCCESSFUL.

Red Hat (CentOS Linux release 7.4.1708) -- Summary
- Red Hat -- OAI Build eNB -- USRP option
- BUILD was SUCCESSFUL.

More details

Click Here to access the Jenkins Job webpage

Details on the job

Click to have the list of files to format

Details on the SCA errors and warnings

Each variant summary for each library

More details
HTML Reporting on tests

Click Here to access the Jenkins Job webpage

Click Here for more details

Click Here for more details

Click Here for more details
Let see more details on our failing test

Here is the issue

Click here to expand the table
How to get to build log files

Job Summary -- Job: eNb-CI -- Build-ID: 503

Click Here to access the Jenkins Job webpage from the HTML build or test report

Click on the Artifacts tab

Download build or test logs

HTML Reports from Master and Slave Jobs are archived also

Download All

Click Here to access the Jenkins Job webpage from the HTML build or test report
VM build and test log files
How you can contribute to CI Master Job

• Only a single script is pulled from the develop branch
  • `ci-scripts/Jenkinsfile-gitlab`
  • It means a modification on this file during a Merge-Request won’t be taken into account unless merged to develop branch
    • Adding / Removing a stage
    • Changing actions within a stage

• All the rest is opened to any contributor:
  • Adding tests to basic-simulator → see `ci-scripts/runTestOnVM.sh`
  • Adding tests to physical-simulators → see `cmake_targets/autotests/test_case_list.xml`
  • Adding FLEXRAN commands/queries → see `ci-scripts/runTestOnVM.sh`
  • Later on we will introduce full L1 and L2 simulator stages
Side Branches Testing

• We can have independent CI for a feature branch

• Currently gNB-CI master job is automatically triggered by
  • push or merge request to develop-nr branch

• It has extra-stages on build (gNB and 5G-NR UE)
• Additional Testing on physical simulators
• It is calling the same eNB slave jobs for legacy testing

• It SHALL be temporary and SHALL be removed once feature branch is merged into develop
Slave eNB CI Jobs

A lot has been already covered by Kentaro SAWADA’s June training session

I will present the improvements
Modularity

• Currently we have 4 slave jobs
  • Monolithic FDD : Band 7 : 5MHz, 10MHz and 20MHz
  • Monolithic TDD : Band 40 : 5MHz and 10 MHz
  • IF4p5 FDD : Band 7 : 5MHz, 10MHz and 20MHz
  • IF4p5 TDD : Band 40 : 5MHz and 10 MHz

• Based on the same pipeline and the same python script
  • Only the scenario changes

• More can be added
  • LTE-M, NR, .....
Deported Processing

Pipeline executes on:
- **nodeb.eurecom.fr**
  - Common Python
  - Common Pipeline
  - Test XML is a parameter on slave job

- **enbz.eurecom.fr**
  - Build eNB with USRP option

- **epc4ci.eurecom.fr**
  - Nokia LTEBOX EPC
  - OpenAirCN EPC

- **adbx.eurecom.fr**
  - Control UEs

Visible from the World Wide Web

- open5glab.eurecom.fr
- Any Slave Job

Eurecom private network

Faraday Cage

USRP board
Pipeline Parameters

- Parameters are now part of the slave job configuration
- No more inside the Jenkins pipeline file
- Reason: modularity
  - Same pipeline re-used for several Slave Jobs
Added a Stage in Pipeline

Slave job to master job #503
Lockable Resources

- All slave jobs are launched in parallel
- But they may share HW / SW resources

- Jenkins plugin allows locking a “resource” for the whole pipeline
  - Other pipelines have to wait until unlocked
- Currently a single resource: our CI bench
- Could be split
  - By UE
  - By RF board
  - By EPC
One “Action Class” in Python Script: iperf

- Iperf original implementation was using iperf3
- At Eurecom we are mainly using iperf (2.0)
  - We support UDP in Downlink and Uplink
  - We support TCP in Downlink and Uplink
- We introduced “profiles”: single-ue, balanced, un-balanced
- Checks are performed against the requested throughput in UDP mode
HTML Report Live Generation

• At the end of each test-case inside the XML scenario file
  • Python script appends a row to an HTML report file

• If any test-case in scenario fails after eNB sync
  • Automatic detachment / termination of UE
  • Automatic termination of eNB(s)
    • Analysis of the eNB log file to potentially spot issue

• If successful, during termination of eNB(s)
  • Automatic analysis of eNB log file
Slave Job HTML Report Example

Job Summary -- Job: eNB-CH-IF4p5-FDD-Band7-B210 -- Build-ID: 149

2 UE(s) is(are) connected to ADB bench server
1 CAT-M UE(s) is(are) connected to bench server

LTE-M support is currently experimental

Click here to get the Slave job web-page

Click here to get there
Tool Validation

- Recently we have introduced T Tracer support in XML scenario and python script
  - Recording of a RAW file
  - Replaying the extracted messages from the RAW file
  - Tshark recording of the packets between eNB and EPC

<table>
<thead>
<tr>
<th>Time</th>
<th>Event Description</th>
<th>Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:100</td>
<td>Initialize eNB (FOO Band 7/5 MHz) with TTracer on T-UDFWK 0</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>04:0001</td>
<td>Attach single UE</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>04:0002</td>
<td>UE (22000235F0C99522) Attached completed in 2 seconds.</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>04:0003</td>
<td>Detach UE</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>04:0004</td>
<td>Terminate UE</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>04:201</td>
<td>Terminate eNB</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

eNB log file analysis is made from the replayed file.
How to get to the log files

Click Here to access the Jenkins Job webpage from the HTML test report

Click on the Artifacts tab

The most interesting one contains all eNB log files

HTML Report is also archived here
How you can contribute to CI Slave Jobs (1)

• 3 scripts/files are pulled from the develop branch
  • ci-scripts/Jenkinsfile-tmp-ran, ci-scripts/main.py and one of the XML scenarios under ci-scripts/xml_files
  • It means a modification on these files during a Merge-Request won’t be taken into account unless merged to develop branch
    • Adding / Removing a stage on the pipeline
    • Changing actions within a stage
    • Adding / Removing test-case(s) inside the XML scenario file(s)
    • Changing the parameters of a test-case inside the XML scenario file(s)
    • Changing the behavior of an “action class” in the python script

• Nonetheless development on these files can be done and check with a template CI job and a push to develop-ci branch
  • https://open5glab.eurecom.fr:8083/jenkins/view/eNB/job/CI-Sandbox-Ran/
How you can contribute to CI Slave Jobs (2)

• All the configuration files used by CI are located under
  • `ci-scripts/conf_files` folder

• CAUTION: there are 4 keywords that are automatically replaced with correct values by CI scripts
  • `CI_ENB_IP_ADDR`
  • `CI_UE_IP_ADDR`
  • `CI_MME_IP_ADDR`
  • `CI_FLEXRAN_CTL_IP_ADDR`
Statistics since June 2018

- eNb-CI Master job: started June 12th
  - Ran 506 times
- gNb-CI Master job: started June 25th
  - Ran 247 times
- eNB-CI-FDD-Band7-B210: started July 30th
  - Ran 419 times
- eNB-CI-TDD-Band40-B210: started September 3rd
  - Ran 312 times
- eNB-CI-IF4p5-FDD/TDD: started October 16th
  - Ran each 152 times

As of December 7th, 2018
Time Statistics

The difference is some jobs did have compilation issues.

Slave Jobs are triggered by both eNB-CI and gNB-CI master jobs.

The graph shows the time distribution for various jobs, with peaks in September and October for eNB-CI and gNB-CI. The compilation issues are noted in the text overlay on the graph.
eNB-CI Master Job Statistics

PASS / FAIL RATIO

<table>
<thead>
<tr>
<th>MERGE</th>
<th>PASS</th>
<th>FAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>74</td>
<td>302</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PUSH</th>
<th>PASS</th>
<th>FAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>34</td>
<td>80</td>
</tr>
</tbody>
</table>

We have a lot of False Negatives due to:
- Code underperformances
- Tool instability

IT IMPLIES A MANUAL INSPECTION OF
- HTML REPORTS
- LOG FILES
TO SEE IF IT IS REALLY A FAILURE

We have serviced 115 Merge Requests since June
It includes Merge Requests that were trashed
It includes Weekly Integration Merge Requests
Why the eNb-CI Master Job Failed?

<table>
<thead>
<tr>
<th>Failure Reason</th>
<th>Fail</th>
<th>Run</th>
<th>Failure Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI script debugging</td>
<td>4</td>
<td>376</td>
<td>1.1%</td>
</tr>
<tr>
<td>Merge Conflicts</td>
<td>12</td>
<td>376</td>
<td>3.2%</td>
</tr>
<tr>
<td>Compile/Build Issues</td>
<td>34</td>
<td>376</td>
<td>9.0%</td>
</tr>
<tr>
<td>Physical Simulators</td>
<td>6</td>
<td>214</td>
<td>2.8%</td>
</tr>
<tr>
<td>Basic Simulator</td>
<td>96</td>
<td>214</td>
<td>44.9%</td>
</tr>
<tr>
<td>Monolithic-FDD Slave Job</td>
<td>133</td>
<td>266</td>
<td>50.0%</td>
</tr>
<tr>
<td>Monolithic-TDD Slave Job</td>
<td>50</td>
<td>206</td>
<td>24.3%</td>
</tr>
<tr>
<td>IF4p5-FDD Slave Job</td>
<td>15</td>
<td>55</td>
<td>27.3%</td>
</tr>
<tr>
<td>IF4p5-TDD Slave Job</td>
<td>8</td>
<td>62</td>
<td>12.9%</td>
</tr>
</tbody>
</table>

THIS IS ONLY ON THE 376 MERGE REQUESTS’ RUN
CI ALSO EVOLVED OVER TIME

IF THE CI BENCH IS FREE, THE WHOLE PROCESS (MASTER + SLAVES) TAKES ABOUT 90 MINUTES
Issues

• Basic Simulator was never created/meant to be a reliable way of testing
  • It often fails for no reason
  • We need a reliable true simulation tool (RF-less)
    • eNB / gNB
    • Simulated Channel
    • UE

• UL in any transmission mode is underperforming
  • Root of many fail runs with real UE and real RF boards

• TDD is also underperforming
Roadmap for CI and for OAI Code base

• Building and Testing on a ARM platform
• Full over-the-air deployment integrated to the CI
• LTE-M, nbIOT, 5G NR tested by CI

• All C/C++ files in openairinterface5g repository are meeting OAI coding guidelines (formatting, naming convention, logging...)
• No error in static code analysis
• Better Performance
Thanks for listening

More details on the implementation can be found on the Wiki

https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/ci/continuous-integration-home