FUJITSU

shaping tomorrow with you

OpenAirInterface

Training of Continuous Integration bench: Jenkins, Scripting and Test Provisioning

20th Jun. 2018

FUJITSU Limited

SAWADA Kentaro (sawa-ken@jp.fujitsu.com)

Question Rule

FUJITSU

- If you have any questions, please don't hesitate to interrupt me.
  You can ask your questions anytime.

FUJITSU

Why CI?

CI for existing features

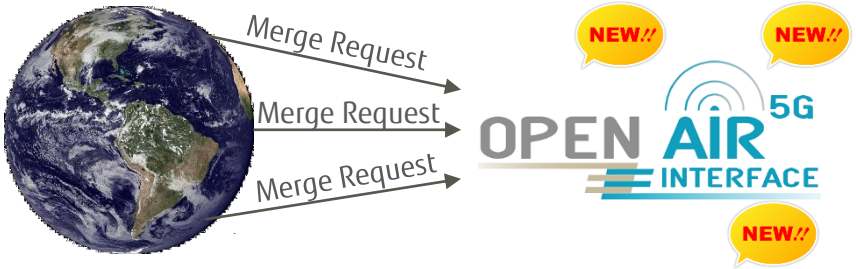
FUJITSU

CI = **C**ontinuous **I**ntegration

Question

Why do we need to **integrate** (test) OAI software **continuously** (all time)?

OAI = **O**pen **A**ir **I**nterface



What about existing features?

Who is responsible for it?


Developers

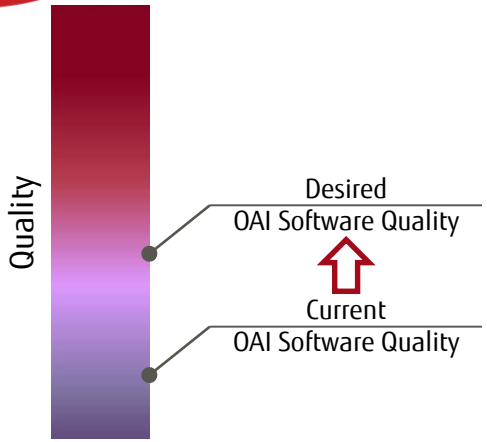

Users


Core Members

Positive Spiral

FUJITSU

Better Quality ▶ More Users ▶ More Attractive Features



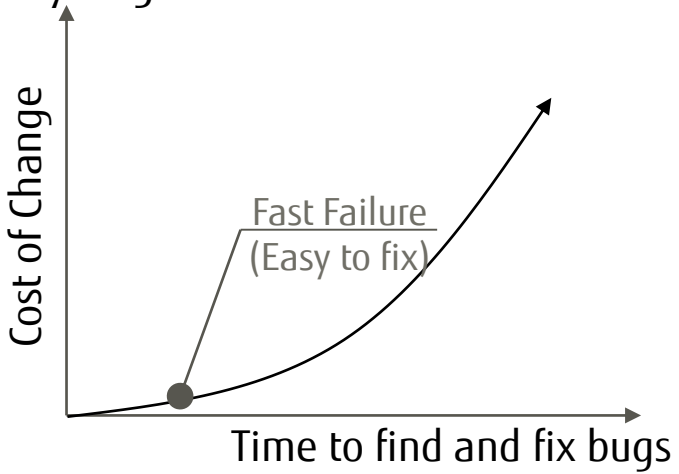
CI for new features

FUJITSU

- CI triggered immediately after Merge Request (Pull Request).
  - Automatically validate whether the compile/build is OK
  - Automatically validate whether or not we have any degradation.

To be discussed

- How to validate NEW features?
- How to merge tests for new features into CI test suite?



FUJITSU

Concept and Architecture

Concept

FUJITSU

- Everyone can use / modify / customize it easily!
  - Last December, we created CI environment with Jenkins only, but we decided to give it up.
  - Avoid Jenkins 職人 problem

SHOKUNIN
=Craftsman / Expert

Jenkins 職人 Problem

Previously configuring Jenkins was not well visible, difficult to be documented. The skills tended to be available only to the person in charge, which could be a single point of failure.

➡ Solved by introducing pipeline script

  - Easy to replicate

Infrastructure as Code

Architecture

FUJITSU

Jenkins (Declarative Pipeline)

Environments (Parameters)

eNB IP Address	eNB Username	eNB Password
eNB Repository	eNB Branch	eNB Commit ID
EPC IP Address	EPC Username	EPC Password
ADB IP Address	ADB Username	ADB Password

Functions

Trigger Control	Log Collection	Alarm Notification
Terminate Processes		

Jenkins

Call

Return

Python File

Basic Test Patterns

eNB Initialization	EPC Initialization	UE Initialization
eNB Termination	EPC Termination	UE Termination
UE Detach	UE Attach	ping
		iperf3

Functions

Log Collection	Time Stamp

python

Read

XML File

Various Test Cases

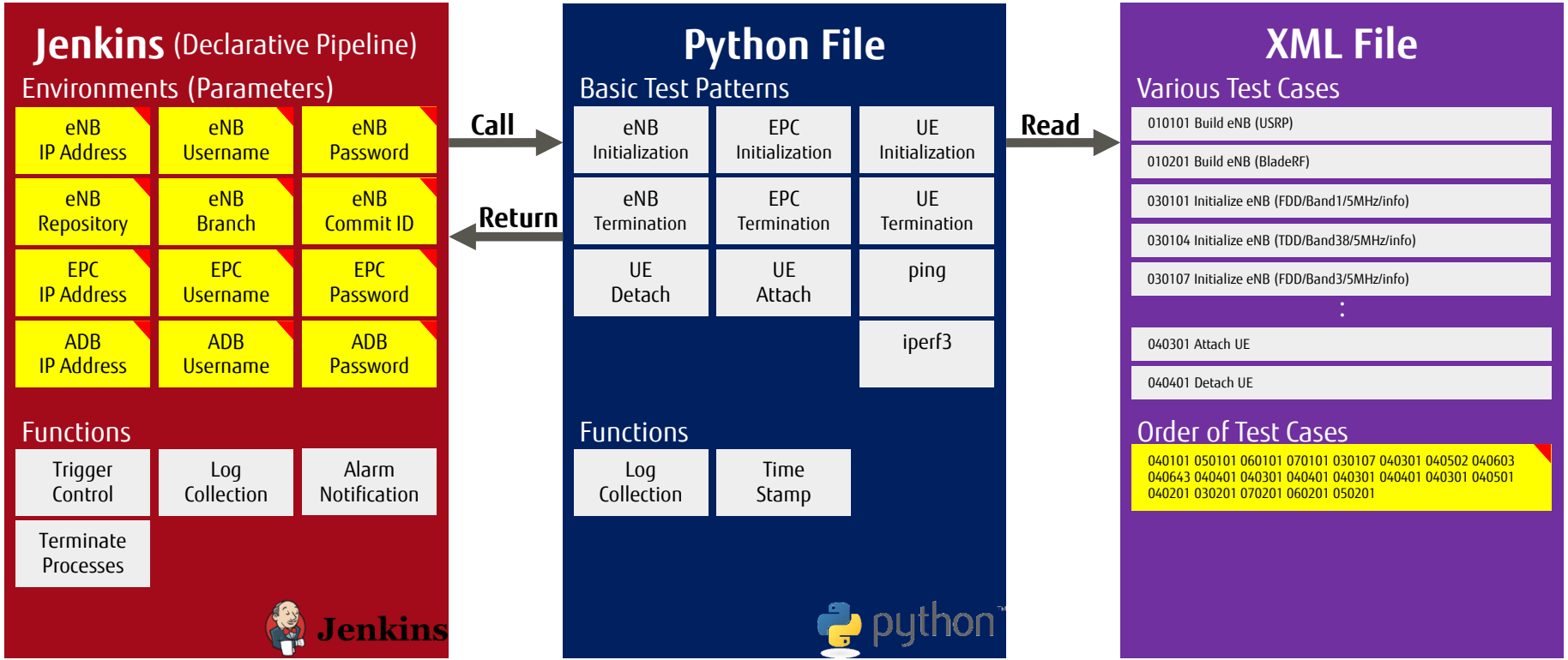
- 010101 Build eNB (USRP)
- 010201 Build eNB (BladeRF)
- 030101 Initialize eNB (FDD/Band1/5MHz/info)
- 030104 Initialize eNB (TDD/Band38/5MHz/info)
- 030107 Initialize eNB (FDD/Band3/5MHz/info)
- :
- 040301 Attach UE
- 040401 Detach UE

Order of Test Cases

040101 050101 060101 070101 030107 040301 040502 040603
040643 040401 040301 040401 040301 040401 040301 040501
040201 030201 070201 060201 050201

Architecture (Required to be changed when replicated)

FUJITSU



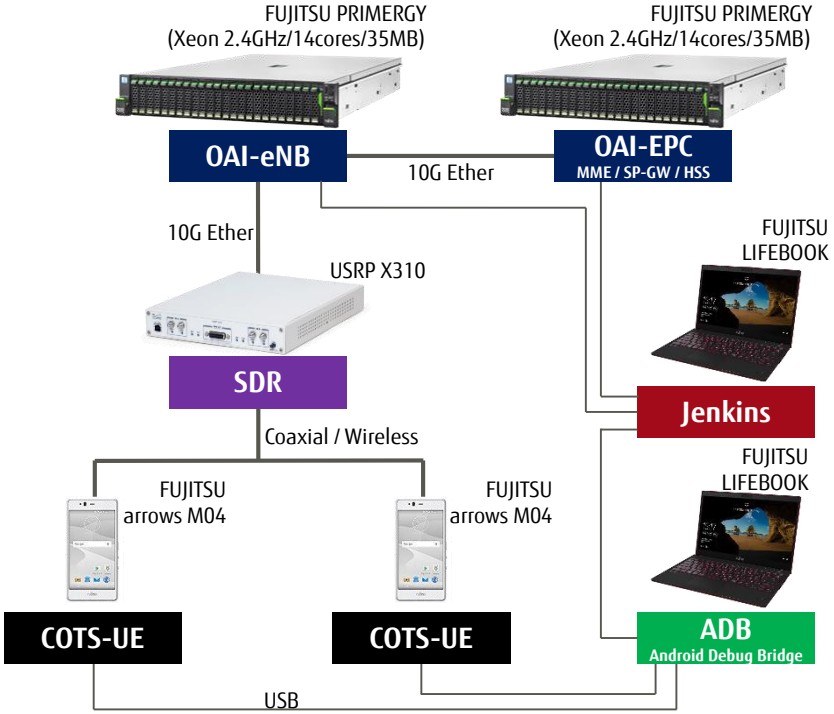
FUJITSU

Demonstration①

Demonstration①

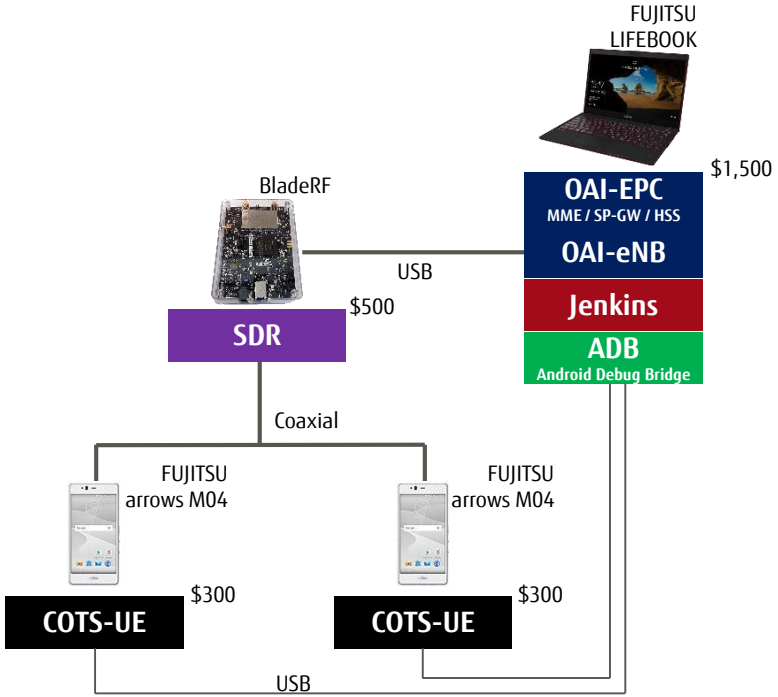
Configuration

Typical Configuration



Today's Configuration

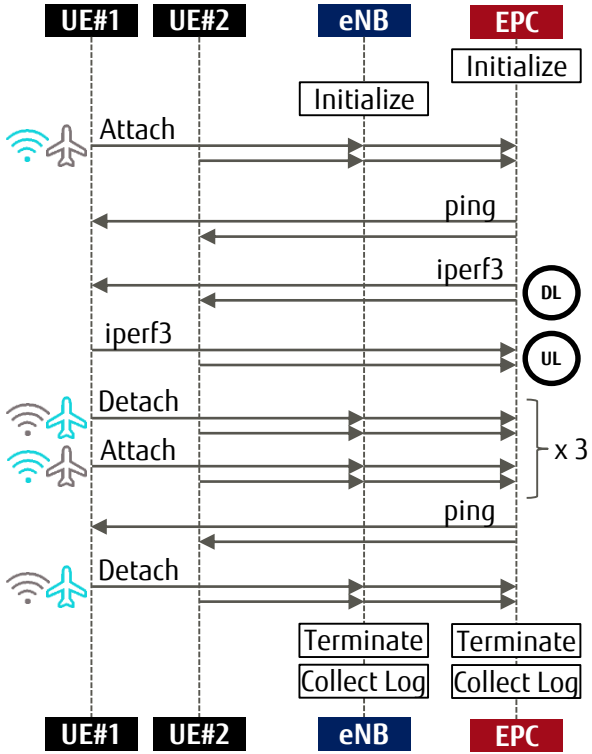
※ Amazon Price (2018/06/18)



Demonstration①

Sequence

1. Start Jenkins manually
2. Initialize 2UEs (Confirm they are in Airplane Mode ON)
3. Initialize EPC (HSS / MME / SP-GW)
4. Initialize eNB
5. Attach 2UEs (Airplane Mode OFF)
6. Ping to 2UEs (for 10sec)
7. iperf3 to 2UEs (DL for 10 sec)
8. iperf3 from 2UEs (UL for 10 sec)
9. Detach ➡ Attach × 3 times
10. Ping to 2UEs (for 10sec)
11. Terminate 2UEs (Flight Mode ON)
12. Terminate eNB / SP-GW / MME / HSS
13. Log Collection of eNB / SP-GW / MME / HSS / ping / iperf3



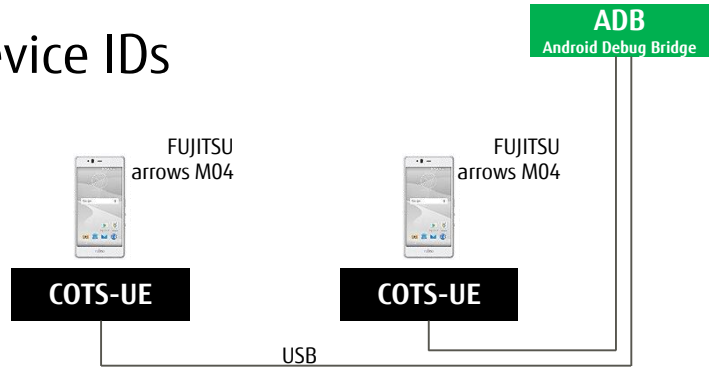
FUJITSU

Technical Design and Implementation in CI

ADB (Android Debug Bridge)

FUJITSU

Jenkins | **Python** | XML



- Count the number of UEs and retrieve UE's device IDs
  - adb devices
- Retrieve IP Addresses of UEs
  - adb -s DEVICEID shell ip addr show
- Airplane Mode OFF
  - adb -s DEVICEID settings put global airplane_mode_on 0
    adb -s DEVICEID am broadcast -a android.intent.action.AIRPLANE_MODE --ez state false
- Airplane Mode ON
  - adb -s DEVICEID settings put global airplane_mode_on 1
    adb -s DEVICEID am broadcast -a android.intent.action.AIRPLANE_MODE --ez state true

Calculation for suitable values

FUJITSU

Jenkins | **Python** | XML

- Timer duration
  - CI script is waiting for appearing of the expected strings when ping and iperf3, but the timeout duration value is automatically calculated from the value in ping/iperf3 arguments.
    ex. ping -c 10 ➡ T.O. value is 10 sec x 3 = 30 sec
- iperf3
  - iperf3's bitrate is automatically calculated (Given bitrate is going to be divided by the number of UEs).
    ex. 12Mbps system throughput ➡ iperf3 -b 6M (in case of 2 UEs)
    iperf3 –b 4M (in case of 3 UEs)

Colorful Console Output

FUJITSU

Jenkins Python XML

ping

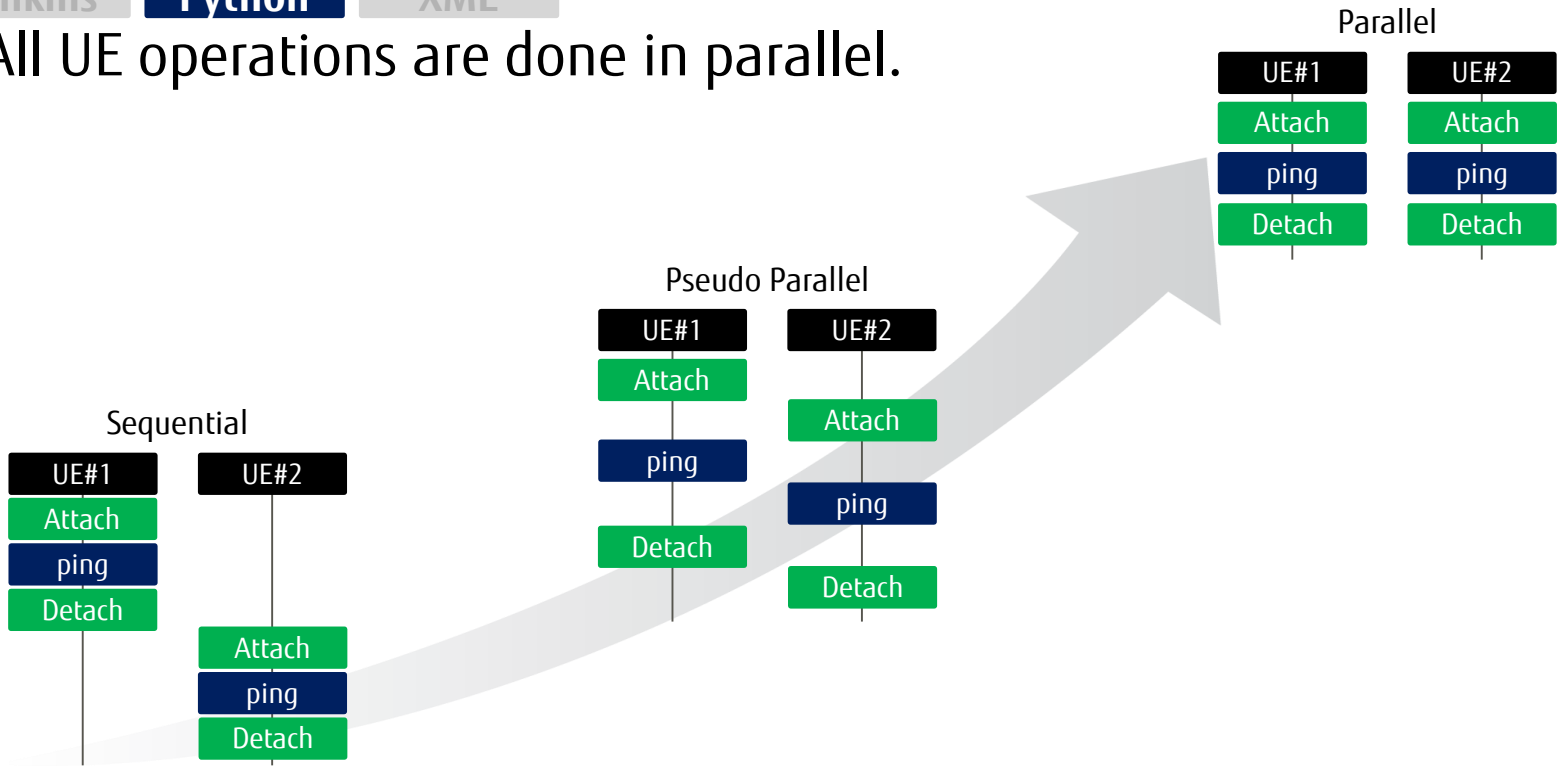
```
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG: ✓ Test OAI-eNB/COTS-UE/OAI-EPC
:
10:22:31 [2018-06-08 10:22:20,311] root:DEBUG: stdbuf -o0 ping -c 10 192.168.200.2 2>&1 stdbuf -o0 tee -a ping_040501_c0beb860.log
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG: Ping result (192.18.200.2)
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG:   Packet Loss : 5%
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG:   RTT(Min)    : 22.874 ms
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG:   RTT(Avg)    : 32.363 ms
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG:   RTT(Max)    : 101.017 ms
10:22:31 [2018-06-08 10:22:30,680] root:DEBUG: Packet Loss too high
```

Simultaneous Multiple UEs Operation

FUJITSU

Jenkins **Python** XML

- All UE operations are done in parallel.



Exclusive Lock while Log Output

FUJITSU

Jenkins | **Python** | XML

w/o Exclusive Lock

```
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  Ping result (192.18.200.2)
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  Ping result (192.18.200.3)
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  Packet Loss : 0%
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  Packet Loss : 0%
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Min)    : 22.874 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Min)    : 17.867 ms
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Avg)    : 32.363 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Avg)    : 31.129 ms
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Max)    : 101.017 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Max)    : 98.128 ms
```

w/ Exclusive Lock

```
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  Ping result (192.18.200.2)
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  Packet Loss : 0%
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Min)    : 22.874 ms
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Avg)    : 32.363 ms
18:06:08 [2018-05-24 18:05:59,906] root:DEBUG:  RTT(Max)    : 101.017 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  Ping result (192.18.200.3)
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  Packet Loss : 0%
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Min)    : 17.867 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Avg)    : 31.129 ms
18:06:08 [2018-05-24 18:05:59,918] root:DEBUG:  RTT(Max)    : 98.128 ms
```

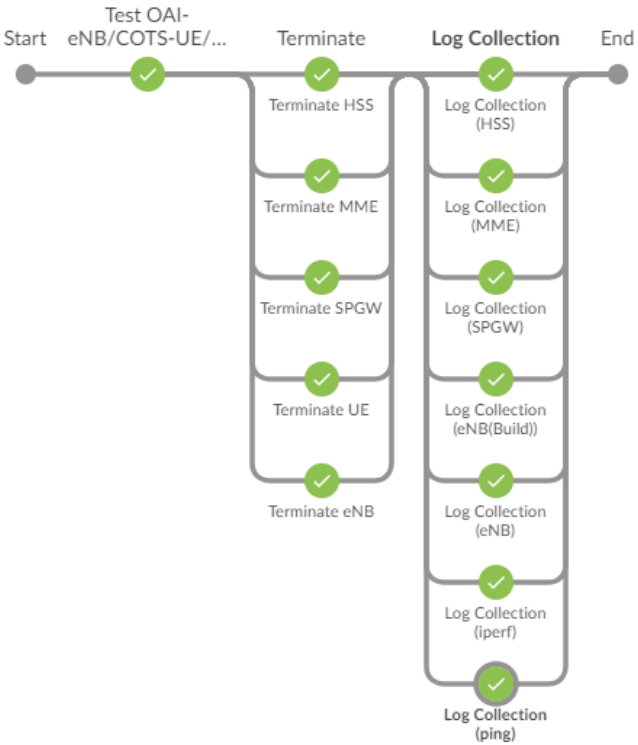
Simultaneous Termination and Log Collection

FUJITSU

Jenkins Python XML

- Termination processes and Log Collection processes should be executed after the test regardless of the test result.

- All Termination and Log Collection processes are now parallelized.



Speedup Jenkins Process

FUJITSU

Jenkins Python XML

- Jenkins Option Changed
  - (MAX_SURVIVABILITY ➡ PERFORMANCE_OPTIMIZED)

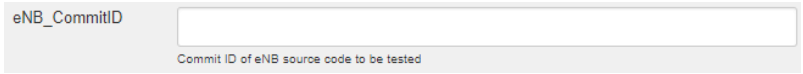
    Reduce disk I/O

Commit ID can be specified

FUJITSU

Jenkins Python XML

- You can specify the Commit ID for eNB to be tested.



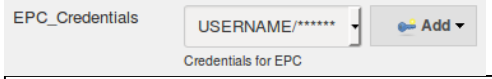
- This parameter is optional. If it's blank, the latest one will be tested.

password for ssh and sudo

FUJITSU

Jenkins Python XML

- How to make SSH connection without being asked about password?
  - sshpass –p **PASSWORD** ssh **USERNAME**@HOSTNAME
- How to "sudo" without being asked about password?
  - echo **PASSWORD** | sudo –S ./run_hss
- How to hide these Credential information in Jenkins log?




```
withCredentials([
  [$class: 'UsernamePasswordMultiBinding',
   credentialsId: "${params.eNB_Credentials}",
   usernameVariable: 'Username',
   passwordVariable: 'Password']
]) {
  sh "echo Password | sudo –S ./run_hss &"
}
```

```
10:22:01 [2018-06-08 10:22:01.856] root:DEBUG: echo PASSWORD | sudo –S ./run_hss &
10:22:03 [2018-06-08 10:22:02.940] root:DEBUG: echo PASSWORD | sudo –S ./run_mme &
10:22:04 [2018-06-08 10:22:04.920] root:DEBUG: echo PASSWORD | sudo –S ./run_spgw &
```

```
10:22:01 [2018-06-08 10:22:01.856] root:DEBUG: echo **** | sudo –S ./run_hss &
10:22:03 [2018-06-08 10:22:02.940] root:DEBUG: echo **** | sudo –S ./run_mme &
10:22:04 [2018-06-08 10:22:04.920] root:DEBUG: echo **** | sudo –S ./run_spgw &
```

iperf3

FUJITSU

Jenkins | **Python** | XML

- To be more precise, iperf is version 2 but it's already obsolete.
- The latest iperf is iperf3 (version 3.5).
  - Very easy to switch DL/UL traffic direction

iperf

```
ue$ iperf –s
epc$ iperf <UE_IP> -u -b 6M -t 60 -i 1

ue$ Ctrl-C
epc$ iperf -s
ue$ iperf <EPC_IP> -u –b 2M –t 60 –i 1
```

iperf3

```
ue$ iperf3 –s
epc$ iperf3 <UE_IP> -u -b 6M -t 60 -i 1

epc$ iperf3 <UE_IP> -u -b 2M -t 60 -i 1 -R
```

Timestamp

FUJITSU

Jenkins Python XML

- 2 timestamps added.
  - Jenkins Console Log Output (timestamp plugin)
  - eNB/EPC log files

```
./run_hss 2>&1 | awk '{ print strftime("[%Y/%m/%d %H:%M:%S] ",systime()) $0 }' | tee -a hss.log
./run_mme 2>&1 | awk '{ print strftime("[%Y/%m/%d %H:%M:%S] ",systime()) $0 }' | tee -a mme.log
./run_spgw 2>&1 | awk '{ print strftime("[%Y/%m/%d %H:%M:%S] ",systime()) $0 }' | tee -a spgw.log
./lte-softmodem 2>&1 | awk '{ print strftime("[%Y/%m/%d %H:%M:%S] ",systime()) $0 }' | tee -a enb.log
```

enb.log

```
[2018/05/28 17:21:15] [MAC][I][cancel_ra_proc] [eNB 0][RAPROC] CC_id 0 Frame 577 Canceled RA procedure for UE rnti 9ef5
[2018/05/28 17:21:16] [MAC][I][cancel_ra_proc] [eNB 0][RAPROC] CC_
[2018/05/28 17:21:16] Assertion (sched_ctl->round[CC_idP][harq_pid] < 8) failed!
[2018/05/28 17:21:16] In extract_harq() /home/fujitsu/smbshare/work_sawaken/CI/master/openair2/LAYER2/MAC/eNB_scheduler_primitives.c:4012
[2018/05/28 17:21:16] Got ACK/NAK for inactive harq_pid 0 for UE 0/bd5a
[2018/05/28 17:21:16]
[2018/05/28 17:21:16] Exiting execution
```

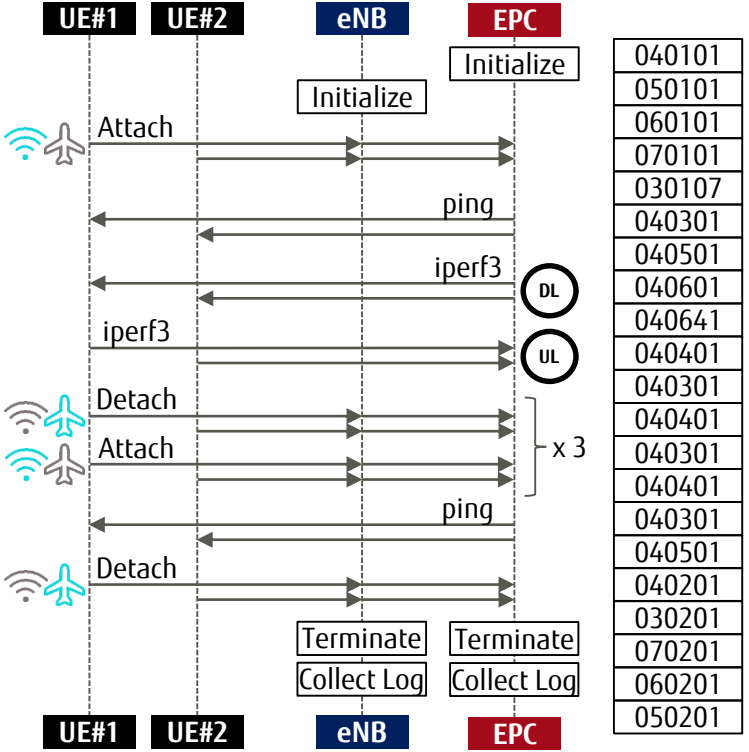
FUJITSU

Demonstration②

Demonstration②

Sequence

1. Get the latest scripts
2. Copy & Paste Jenkinsfile
3. Modify Jenkinsfile (IP Address, Username, Password etc.)
4. Run Jenkins one time (failed)
5. Copy & Paste Python File and XML File
6. **Modify XML File to create a new test suite.**
7. Run Jenkins
8. Check the Jenkins Output Console
9. Check the eNB/EPC log files



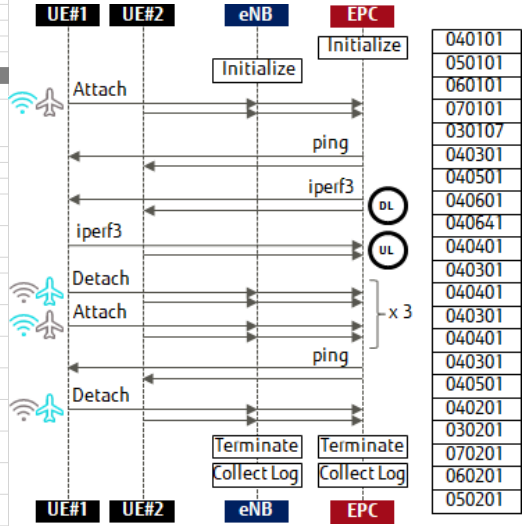
Demonstration②

Define Your Test Cases

FUJITSU

Test Case List

	testCase id	class	desc	Option	Note
03 eNB					
01 Initialization					
01	**030101**	Initialize_eNB	Initialize eNB (FDD/Band1/5MHz/info)	<Initialize_eNB_args>-O 5M.band1.FDD.info.conf</Initialize_eNB_args>	
02	**030102**	Initialize_eNB	Initialize eNB (FDD/Band1/10MHz/info)	<Initialize_eNB_args>-O 10M.band1.FDD.info.conf</Initialize_eNB_args>	
07	**030107**	Initialize_eNB	Initialize eNB (FDD/Band3/5MHz/info)	<Initialize_eNB_args>-O 5M.band3.FDD.info.conf</Initialize_eNB_args>	
08	**030108**	Initialize_eNB	Initialize eNB (FDD/Band3/10MHz/info)	<Initialize_eNB_args>-O 10M.band3.FDD.info.conf</Initialize_eNB_args>	
04 UE					
03 Attach					
01	**040301**	Attach_UE	Attach UE	#N/A	Flight Mode Off Wait for maximum 180 sec until mDataConnectionState becomes 2.
04 Detach					
01	**040401**	Detach_UE	Detach UE	#N/A	Flight Mode On
05 Ping					
01	**040501**	Ping	ping (20 sec)	<ping_args>-c 20</ping_args> <ping_packetloss_threshold>5</ping_pakcetloss_threshold>	ping is system command of Android
02	**040502**	Ping	ping (60 sec)	<ping_args>-c 60</ping_args> <ping_packetloss_threshold>2</ping_pakcetloss_threshold>	ping is system command of Android
06 Iperf					
01	**040601**	Iperf	iperf (DL/6Mbps/UDP)(60 sec)	<iperf_args>-u -b 6M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
02	**040602**	Iperf	iperf (DL/13Mbps/UDP)(60 sec)	<iperf_args>-u -b 13M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
03	**040603**	Iperf	iperf (DL/15Mbps/UDP)(60 sec)	<iperf_args>-u -b 15M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
04	**040604**	Iperf	iperf (DL/27Mbps/UDP)(60 sec)	<iperf_args>-u -b 27M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
05	**040605**	Iperf	iperf (DL/35Mbps/UDP)(60 sec)	<iperf_args>-u -b 35M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
06	**040606**	Iperf	iperf (DL/70Mbps/UDP)(60 sec)	<iperf_args>-u -b 70M -t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
41	**040641**	Iperf	iperf (UL/2Mbps/UDP)(60 sec)	<iperf_args>-u -b 2M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
42	**040642**	Iperf	iperf (UL/5Mbps/UDP)(60 sec)	<iperf_args>-u -b 5M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
43	**040643**	Iperf	iperf (UL/8Mbps/UDP)(60 sec)	<iperf_args>-u -b 8M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
44	**040644**	Iperf	iperf (UL/10Mbps/UDP)(60 sec)	<iperf_args>-u -b 10M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
45	**040645**	Iperf	iperf (UL/15Mbps/UDP)(60 sec)	<iperf_args>-u -b 15M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
46	**040646**	Iperf	iperf (UL/30Mbps/UDP)(60 sec)	<iperf_args>-u -b 30M -t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
90	**040690**	Iperf	iperf (DL/TCP)(60 sec)	<iperf_args>-t 60 -i 1</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end
91	**040691**	Iperf	iperf (UL/TCP)(60 sec)	<iperf_args>-t 60 -i 1 -R</iperf_args> <iperf_packetloss_threshold>50</iperf_pakcetloss_threshold>	"iperf3 -s" in UE before initializing iperf3 kill "iperf3 -s" in UE at the end



FUJITSU

ToDo List for CI

Upcoming Events

FUJITSU

- Update our workflow
  - Couple of test cases will be automatically executed immediately after Merge Request, which helps reviewers. **Update will be expected at the end of July**
  - Nightly tests will be executed to improve quality / stability of OAI.
- Integrate with various tools.
  - Slack / Mattermost
  - Code Coverage Tool
  - Code Static Analysis Tool
- Support Simulators
- Manual (Documentation)

FUJITSU

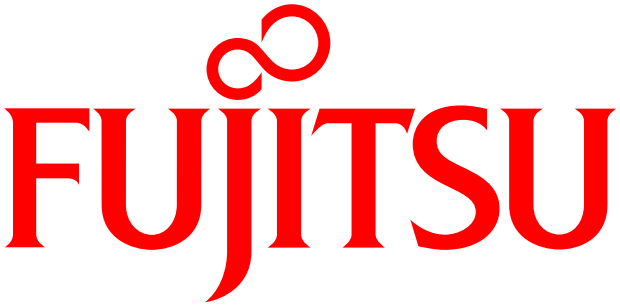
Summary

Summary

What you have learned today

FUJITSU

- The reason why we need CI
- CI's concept and architecture
- CI's technical advantages
- How CI works with Jenkins and Python file and XML file
- How you can setup CI
- How you can modify and customize CI
- Upcoming features



shaping tomorrow with you