



Programmable and Cloud-Native RAN: Challenges, Solutions, enabling Technologies and Tools

Navid Nikaein, Raymond Knopp, and Adlen Ksentini

Communication System, Eurecom



The 5G Infrastructure Public Private Partnership



21th May, 2017

Outline

- **Principles (20')**
- **Technologies (50')**
- **Challenges (50')**
- **Testbeds and Field Trials (50')**
- **Conclusion(10')**

Part I - Principles



Principles

Many use-cases are considered in 5G

High traffic

Office

Residential area



High density

Stadium

Festival



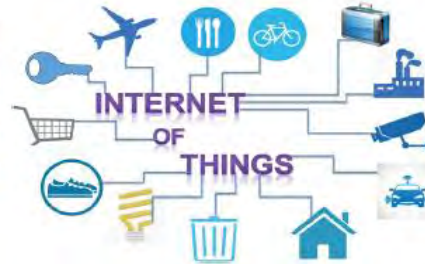
High mobility



High speed train

Freeway

Internet of Things (IoT)



Home automation



Intelligent Transport Sys. (ITS)



Smart city



Mobile Applications



UHD/4K video

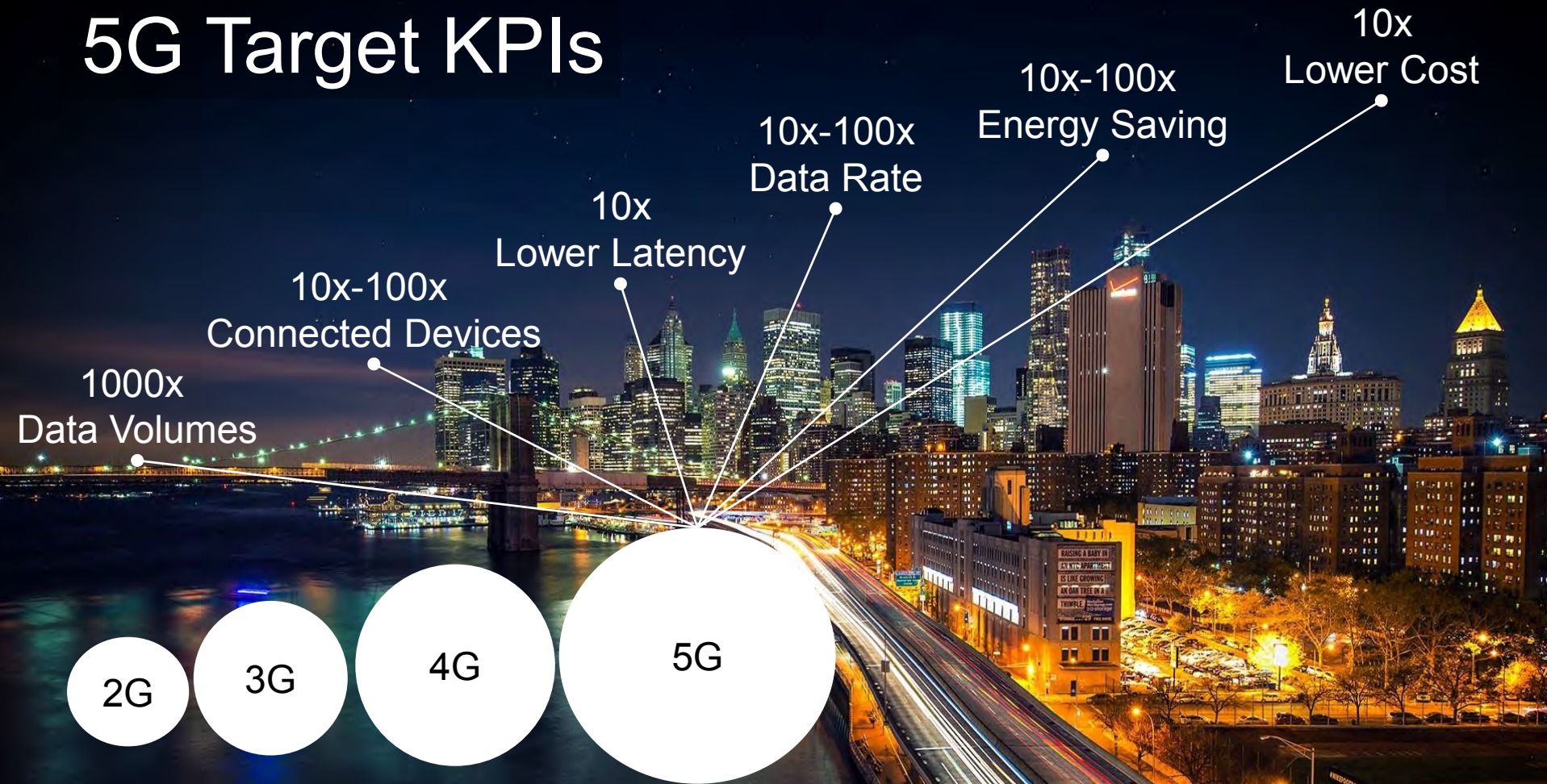
Augmented Reality

What will 5G be?

NGMN high level view





- **5G is an end-to-end ecosystem to enable a fully mobile and connected society**
 - empowers **value creation** towards customers and partners, through existing and emerging **use cases**,
 - delivered with **consistent experience**, and enabled by **sustainable business models**.

5G Target KPIs







Not all of these Requirements need to be satisfied simultaneously

5G Use cases: NGMN Perspective

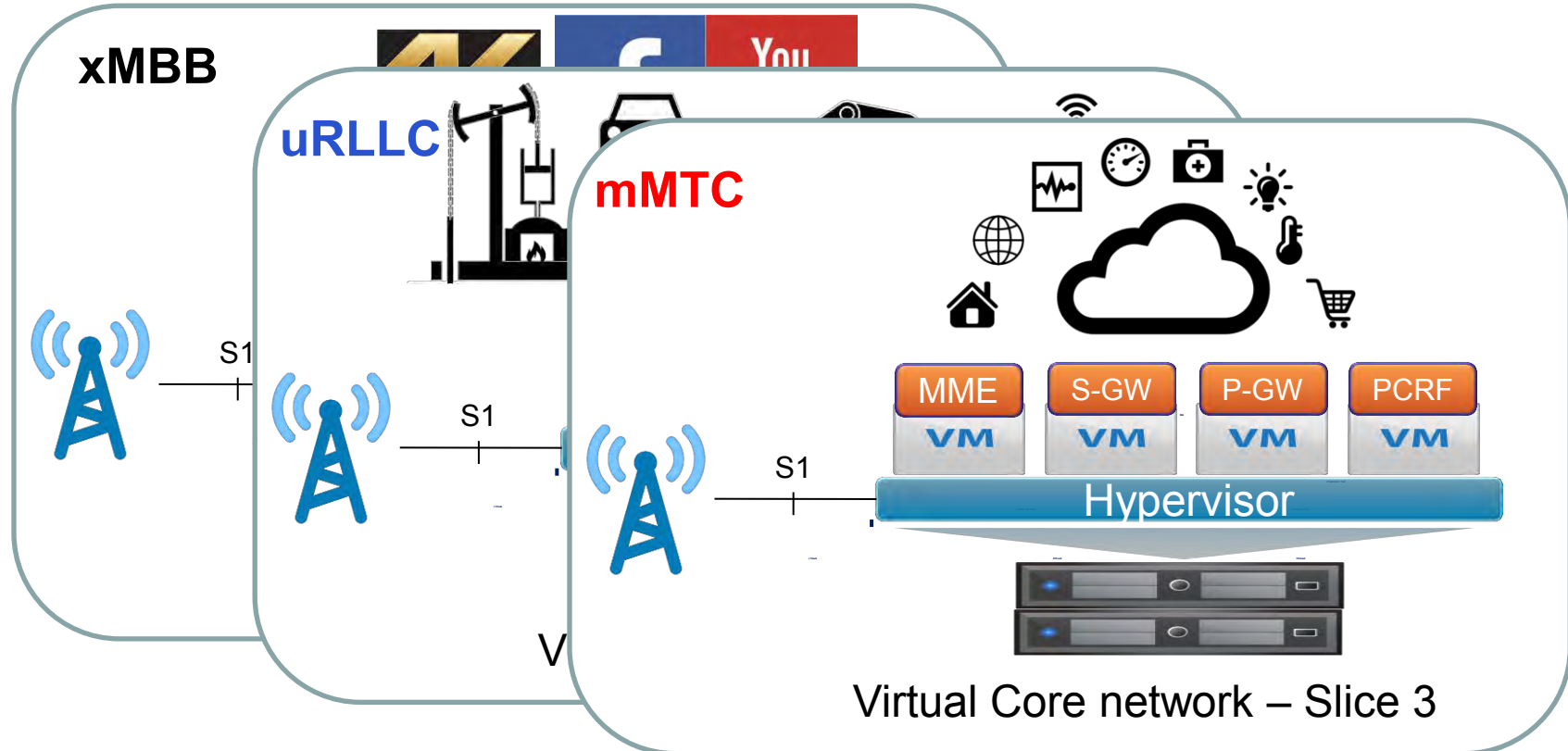
	<p>Broadband access in dense areas</p> <p>PERVASIVE VIDEO</p> 	<p>Broadband access everywhere</p> <p>50+ MBPS EVERYWHERE</p> 	<p>Higher user mobility</p> <p>HIGH SPEED TRAIN</p> 	<p>Massive Internet of Things</p> <p>SENSOR NETWORKS</p> 
Challenge	Data Rate	UBIQUITOUS COVERAGE QUALITY	Mobility	Connectivity Density
Other Usecase	Cloud Service Smart Office HD video/photo sharing	Ultra-low Cost networks	Moving HotSpots Remote Computing	Smart wearables Smart Grid Mobile video surveillance

5G Use cases: NGMN Perspective

	Extreme real-time communications TACTILE INTERNET 	Lifeline communications NATURAL DISASTER 	Ultra-reliable communications E-HEALTH SERVICES 	Broadcast-like services BROADCAST SERVICES 
Challenge	Latency Reliability	Availability	Reliability Latency	Reachability Connectivity
Other Usecase	Industry automation	Earthquakes	Automated driving Collaborative robots Remote operation Public safety	News / information Reginal and national services

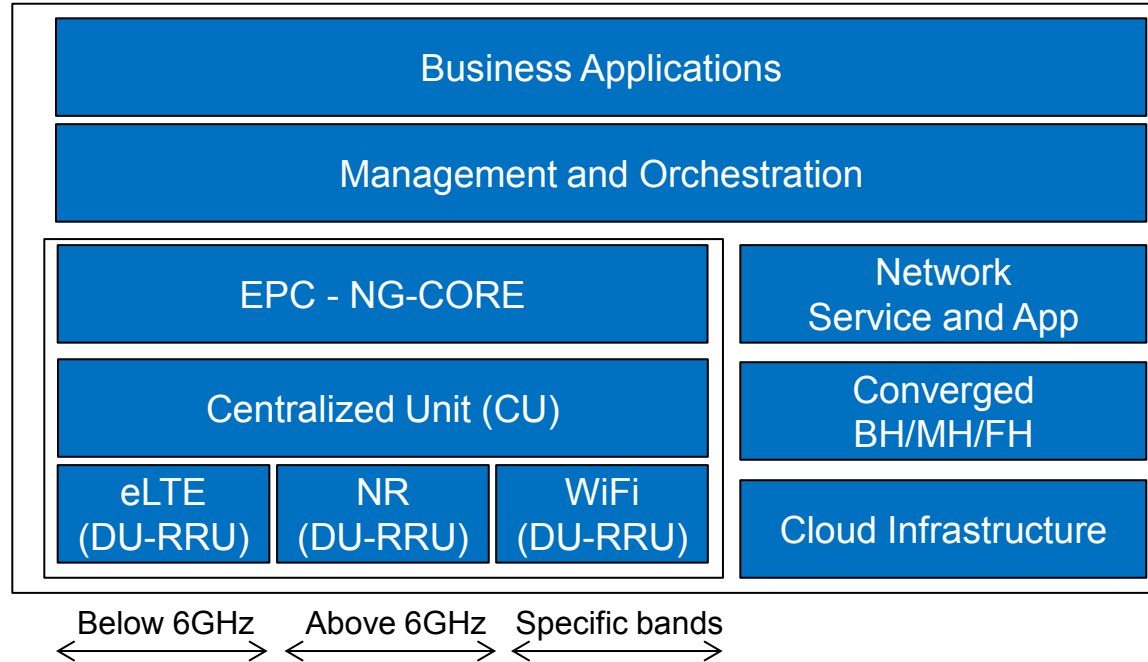
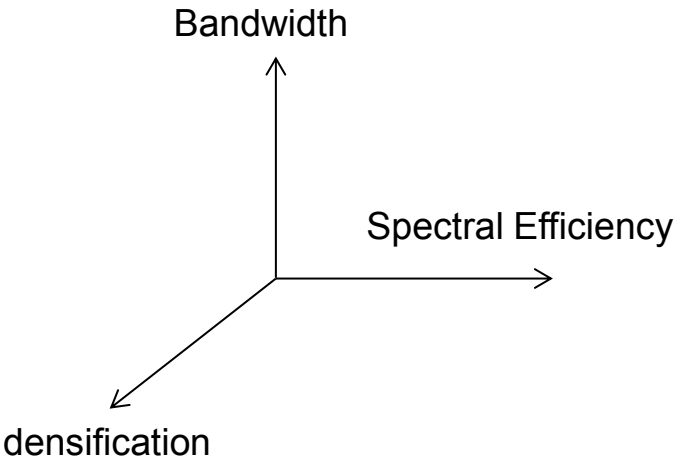
Network Slicing in 5G

Multiple dedicated network for each service on a shared infrastructure



5G will be a paradigm shift

Overall 5G Solution

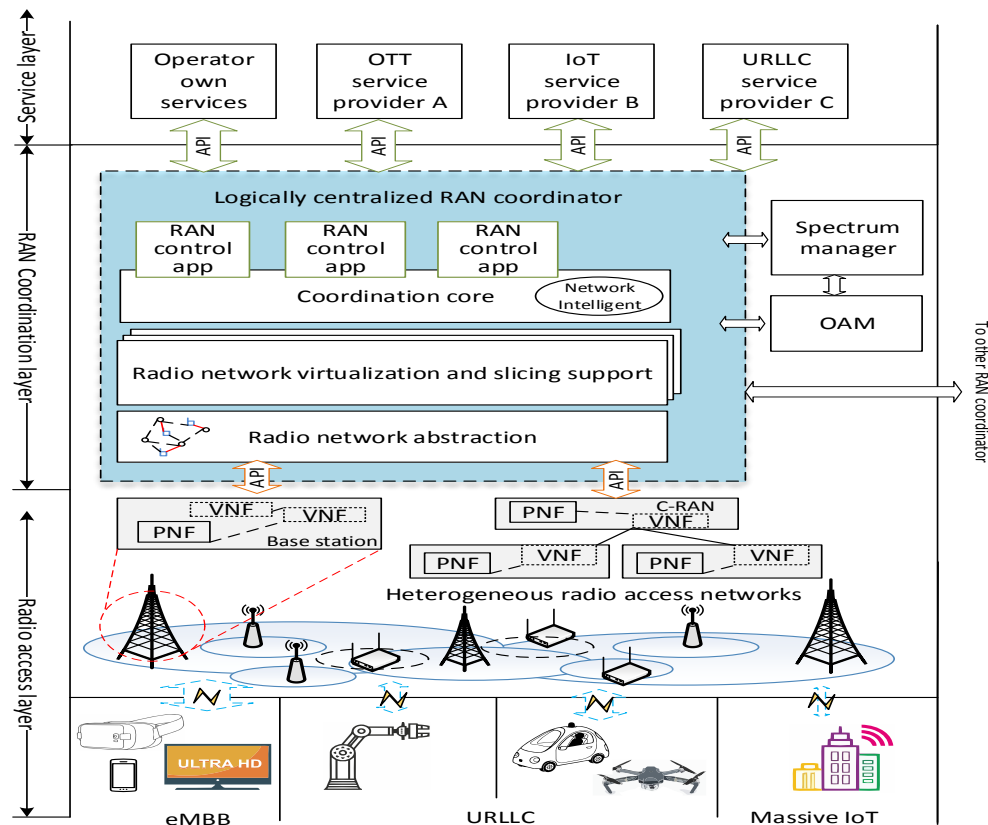
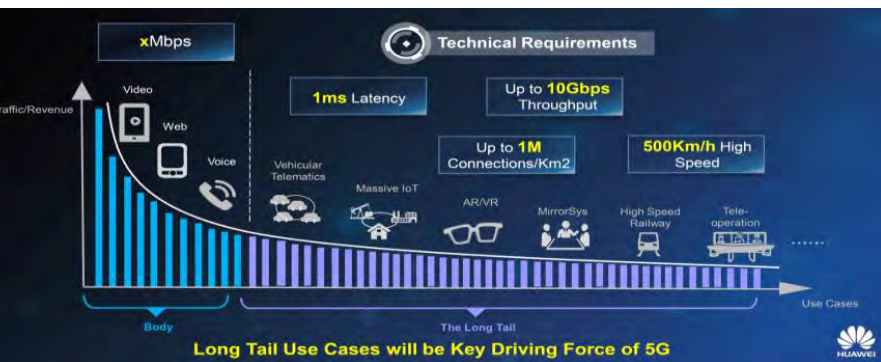


- 5G is not just a new radio/spectrum, but also a new architecture and business helper
- 3-tier Architectue

5G will be a paradigm shift

Long tail use cases to support verticals

- Support of **verticals** will be key driving forces of 5G business to empower value creation
- Key capabilities
 - eMBB, URLLC, mMTC

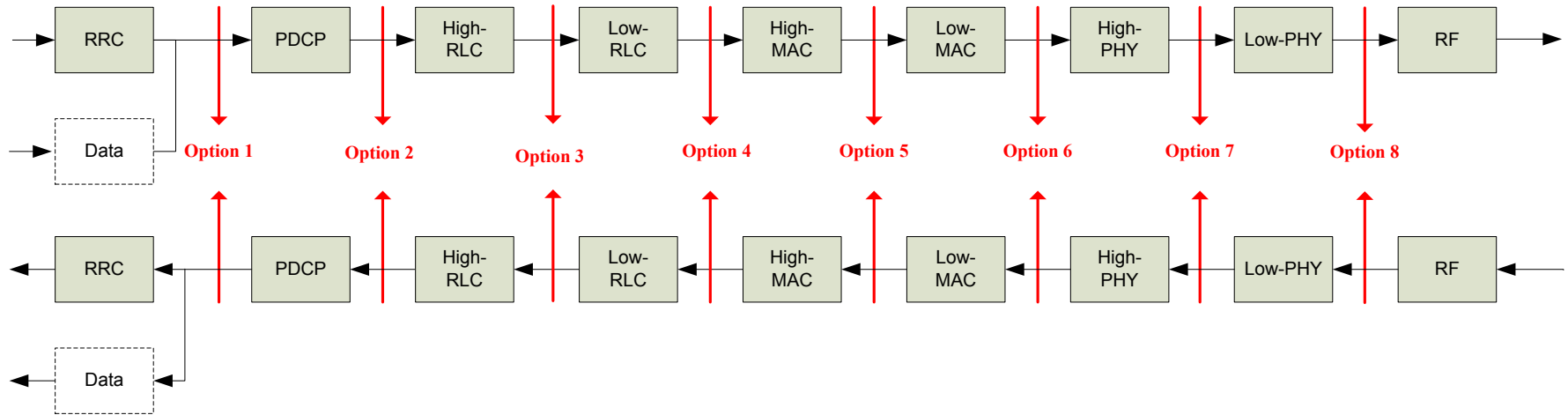


Source: Coherent Project

3GPP re-architects RAN and CN

- **RAN: a 3 tier architecture (CU0 → DU[0-n] → RRU[0-m])**
 - Functional split between CU and DU
 - Functions split between DU and RRU
 - Functional split between c-plane and d-plane
- **CN: a service-centric architecture (1:N, N:1)**
 - Network Service catalog and discovery
 - Functional split between c-plane and d-plane

3GPP RAN Functional Split



■ 3 tier Architecture:

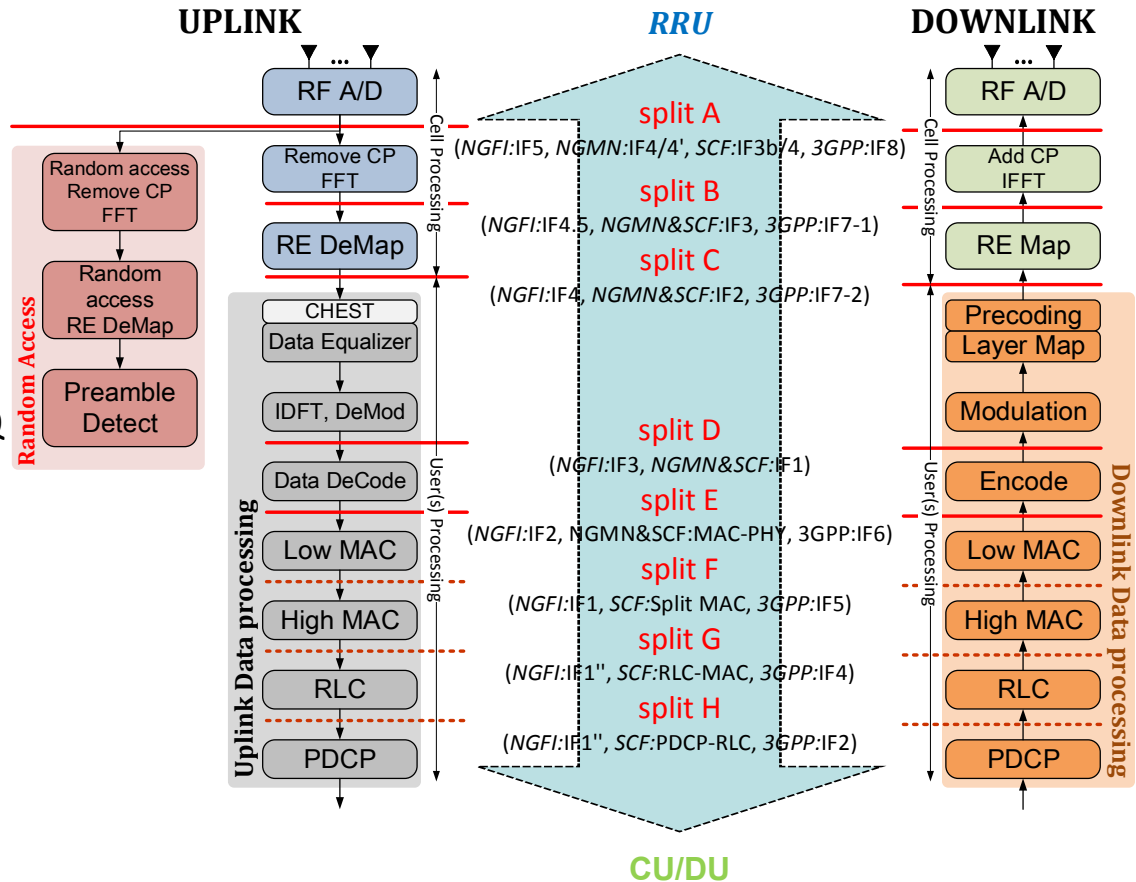
- Centralized unit (CU): coverage area of 100-200 KM radius
- Distributed Unit (DU): coverage area of 10-20 KM radius
- Remote Radio Unit (RRU): coverage area of < 2KM radius

■ CU → DU → RRU

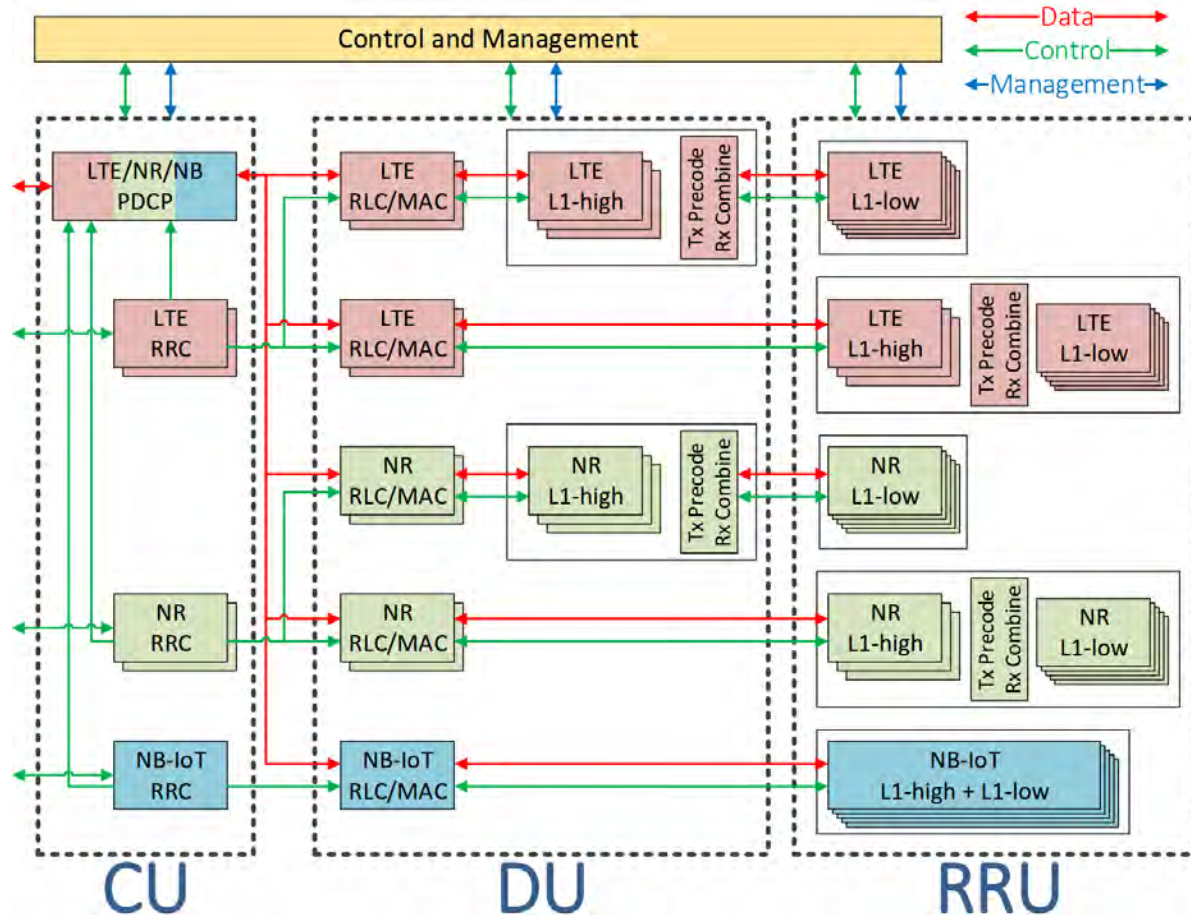
RAN functional Split

■ RRU-BBU functional split

- FH capacity relaxation
- NGFI, NGMN, 3GPP, SCF compliant
- Transport intermediate samples
 - ☞ Split A: Time-domain I/Q
 - ☞ Split B: Freq-domain I/Q
 - ☞ Split C: Allocated freq-domain I/Q
 - ☞ Split D: Log-likelihood ratio of each bit
 - ☞ Split E: Decoded bit



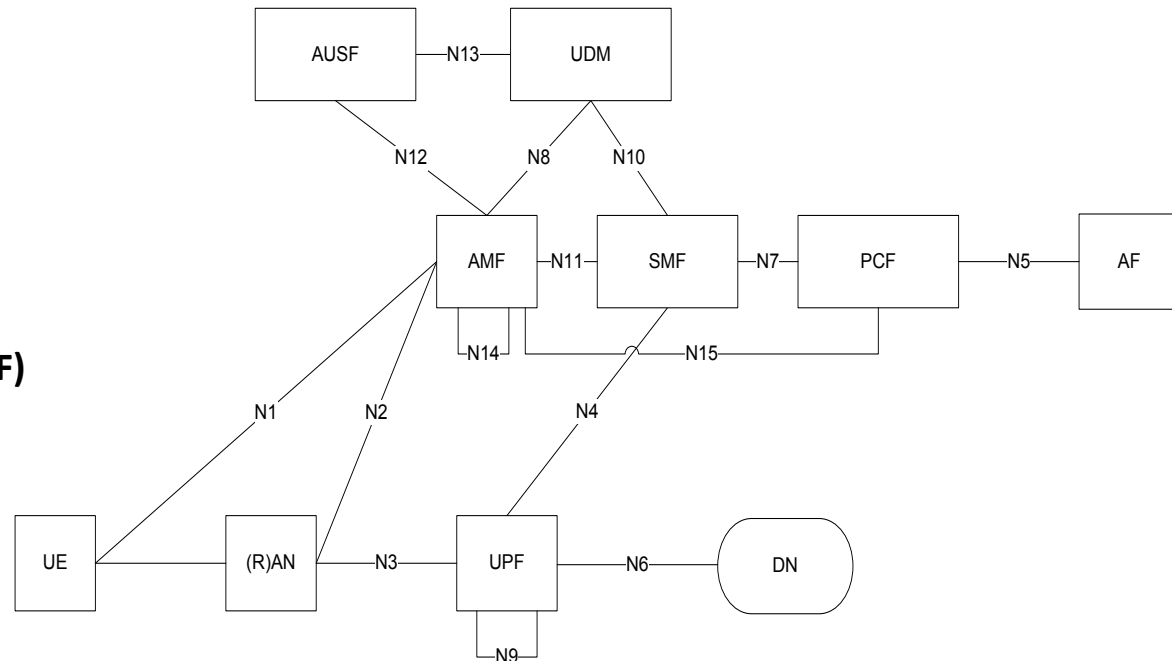
RAN Functional Split



Next Generation Core Architecture 1

- **Authentication Server Function (AUSF)**
- **Core Access and Mobility Management Function (AMF)**
- **Data network (DN)**
 - e.g. operator services, Internet access or 3rd party services
- **Policy Control function (PCF)**
- **Session Management Function (SMF)**
- **Unified Data Management (UDM)**
- **User plane Function (UPF)**
- **Application Function (AF)**
- **User Equipment (UE)**
- **(Radio) Access Network ((R)AN)**

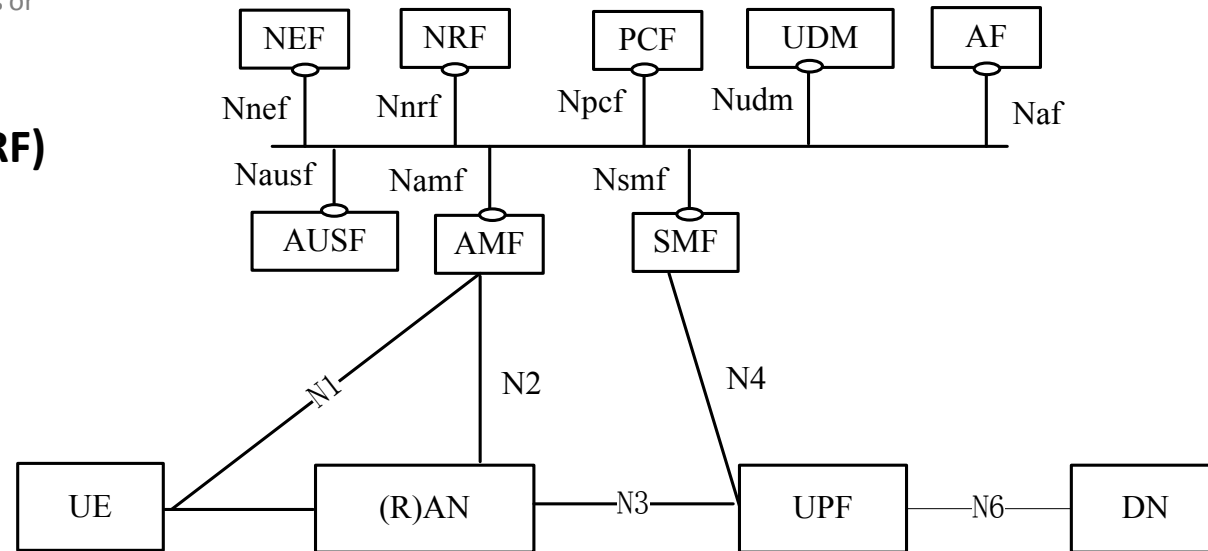
Closer to traditional CN with functions and IF



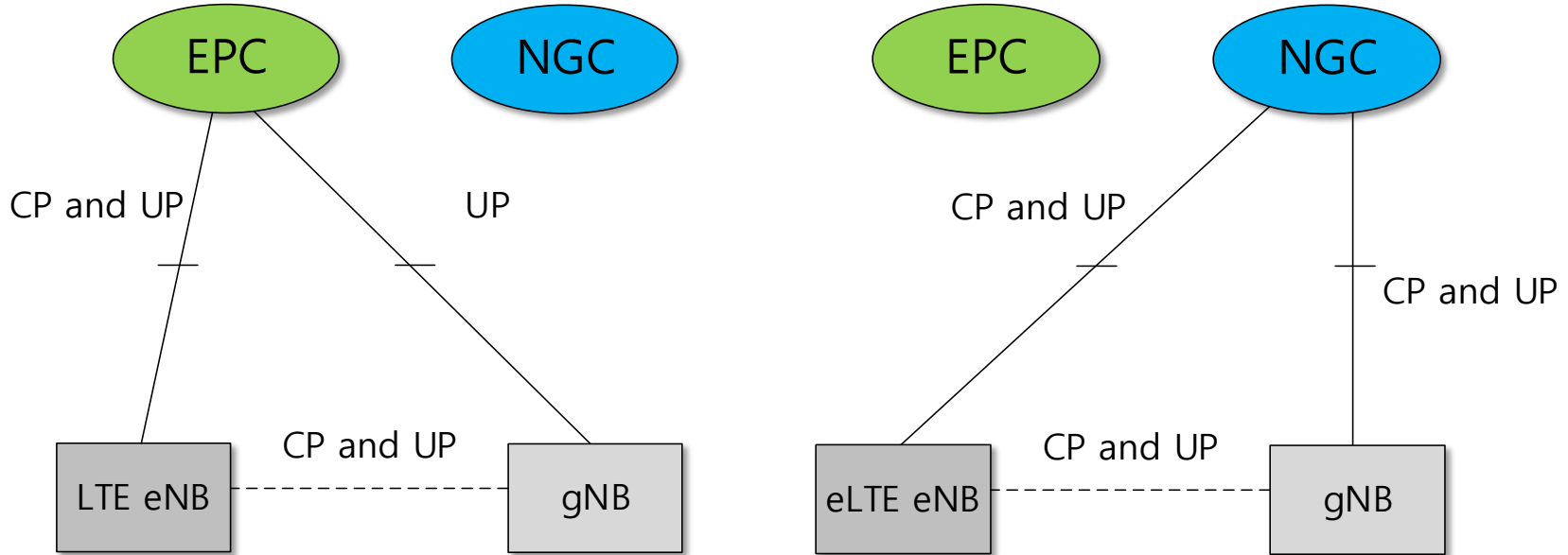
Next Generation Core Architecture 2

- Authentication Server Function (AUSF)
- Core Access and Mobility Management Function (AMF)
- Data network (DN)
 - e.g. operator services, Internet access or 3rd party services
- Network Exposure Function (NEF)
- **NF Repository Function (NRF)**
 - VNF catalog / store
- Policy Control function (PCF)
- Session Management Function (SMF)
- Unified Data Management (UDM)
- User plane Function (UPF)
- Application Function (AF)
- User Equipment (UE)
- (Radio) Access Network ((R)AN)

Service-centric control plane

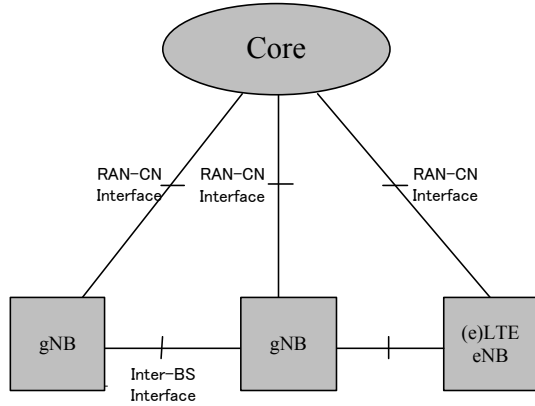


RAN-CN interface

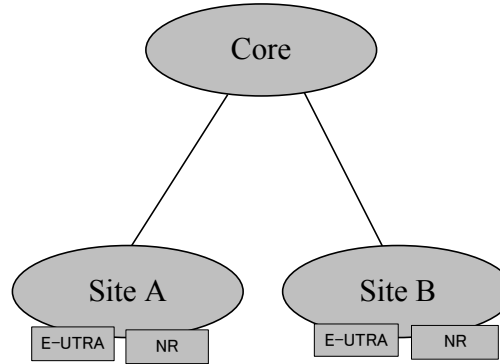


3GPP Envisaged Deployment Scenarios

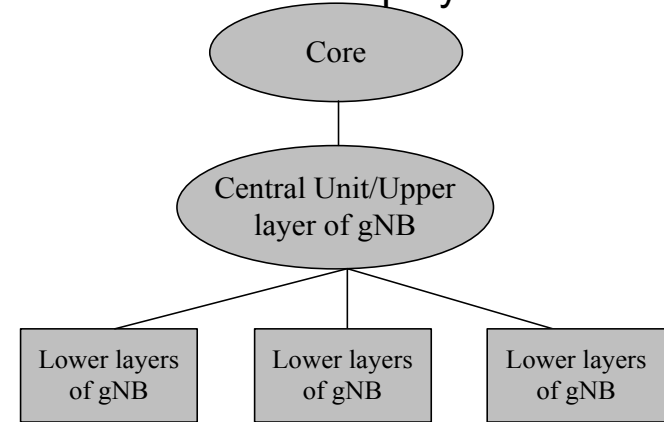
Non-centralised deployment



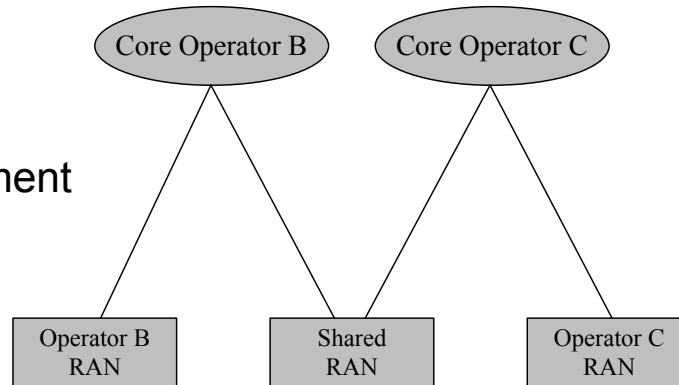
Co-sited deployment



Centralized deployment



Shared RAN deployment



Consideration of Functional Split

- Two type of xhaul

- Low latency
- High latency

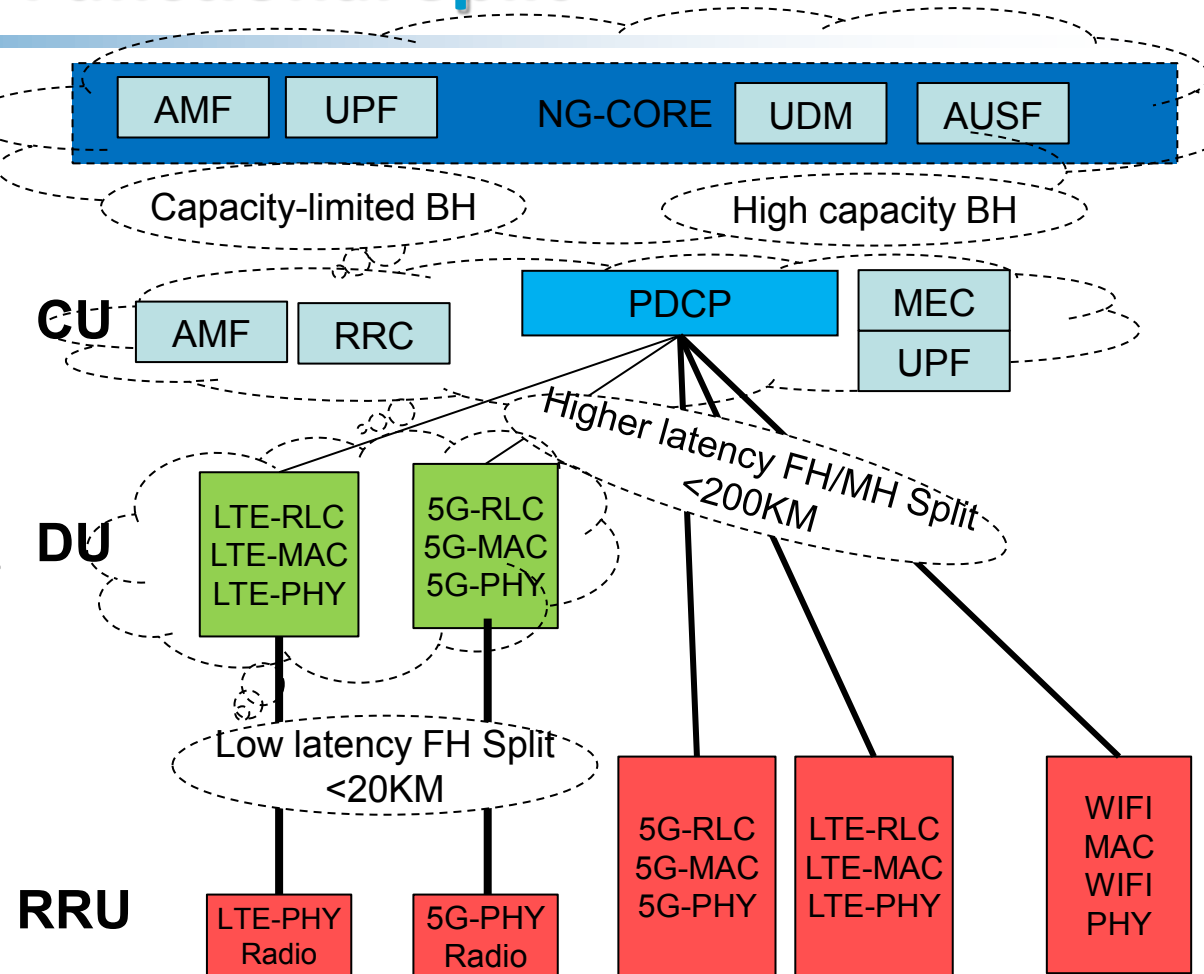
- Various topologies

- multi-tier – flat
- Mesh – tree

- Switching vs routing

- Aggregation
- Distribution

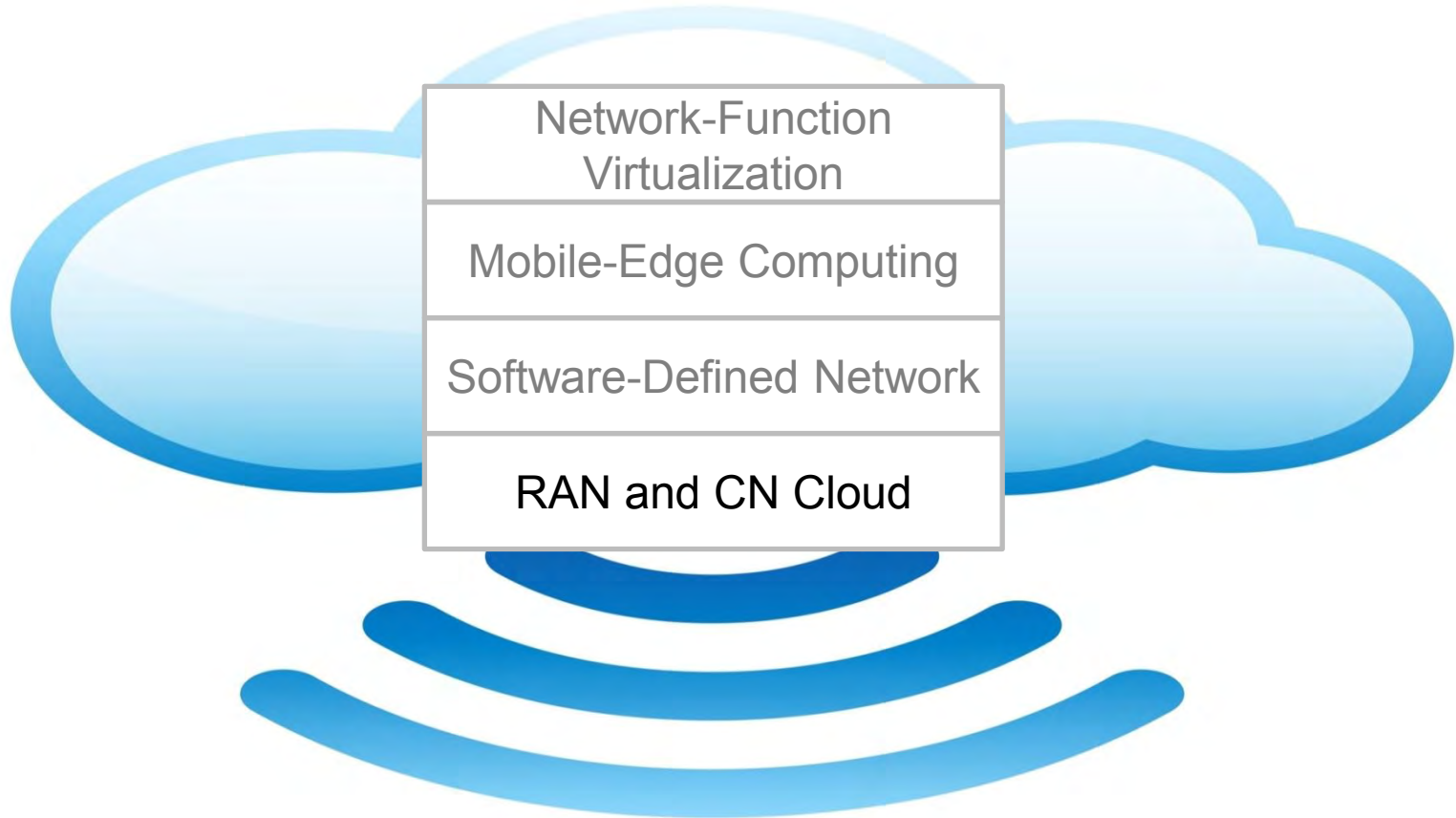
- Data-plane accelerations



Economics of mobile are changing

- **Softwarization and Commoditization**
 - Software implementation of network functions on top of GPP with no or little dependency on a dedicated hardware
 - ☞ Full GPP vs. accelerated vs. system-on-chip
 - Programmable RF
- **Virtualization and Cloudification**
 - Execution of network functions on top of virtualized computing, storage, and networking resources controlled by a cloud OS.
 - Share I/O resources among multiple guests
- **Abstraction and programmability**
 - Unified control-plane framework
 - Logical resources and network graphs
 - Data models, APIs and standardized I/F
- **Cognitive network control**
 - Logics for programmability in RAN and CN
 - ☞ Data plane and control-plane

Part II - Technologies



Technologies

Network-Function Virtualization

Mobile-Edge Computing

Software-Defined Network

RAN and CN Cloud

Cloud Computing

- **Cloud Computing disrupts IT consumption and operations**
 - on-demand, self-service, elastic, pay-as-you-go, metered service,
 - Automated management, remote access, multi-tenancy, rapid deployment and service provisioning, load-balancing
- **New business models based on sharing**
 - Public, private, community, and hybrid clouds
- **Promising business potentials (CAPEX/OPEX)**
 - Start small and grow on-demand

Software as a service Virtual desktop, games, analytics, ...
Platform as a service Data base, web service, ...
Infrastructure as a service VM, storage, network, load balancer

Cloudification
Virtualization
Server Consolidation

Cloud RAN Primer

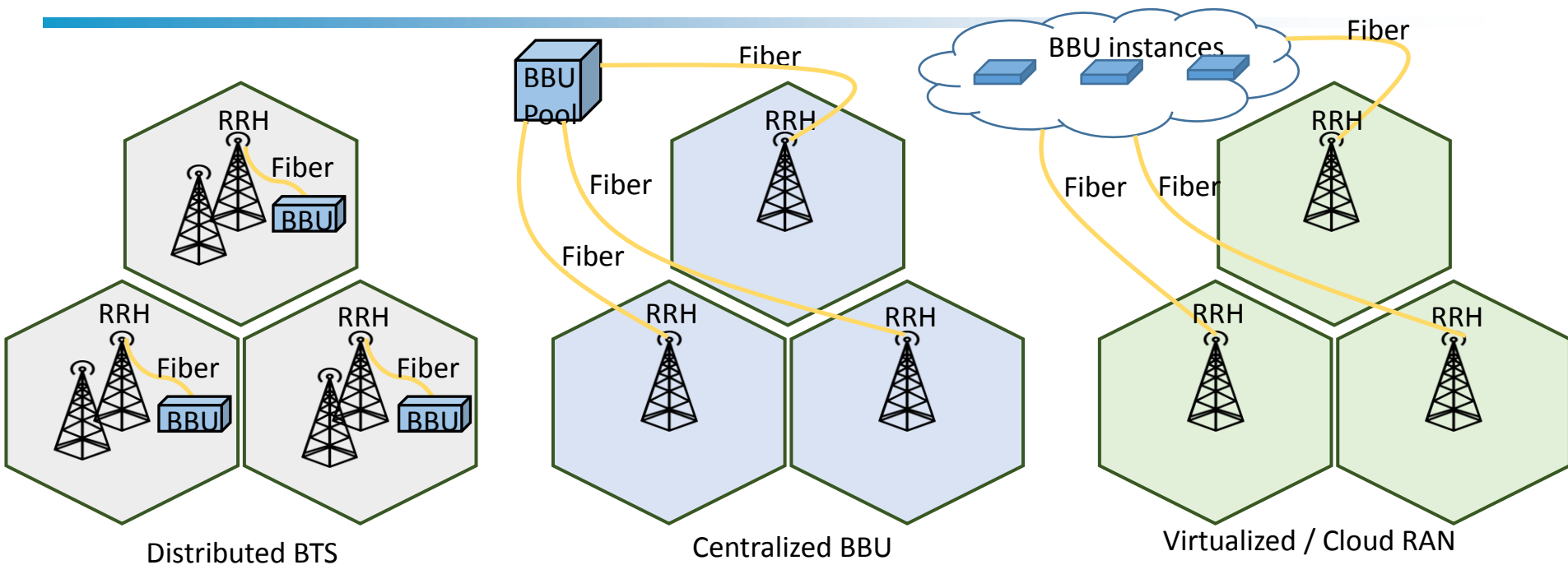
- **Main idea:**

- Decouple the base station processing from the radio unit
- Perform the processing at the high performance cloud infrastructure
- Transport the data through a high speed medium

- **Components**

- **Remote radio head (RRH):** lightweight (passive) radio element with power amplifier and antennas
- **Base band Unit (BBU):** a centralized pool of virtualized base station covering a large set of cells (10 – 100)
- **Fronthaul (FH):** data distribution channel between the BBU pool and RRHs

Cloudification of RAN



- + Split of BBU from RRH
- + Lower power consumptions
- + Better placement of RRH
- Underutilized resources
- Lower spectral efficiency

- + Network of RRHs and better network densification
- + Better placement of BBU and RRH
- + Lower the number of BBU
- Higher complexity
- Fronthaul Capacity requirement (non commodity)

Benefit of a Cloudified RAN

■ Cooperation

- Coordinated signal processing
- Joint scheduling
- Interference management through channel feedbacks

■ Interconnections

- Maximize statistical multiplexing gain
- Load balancing

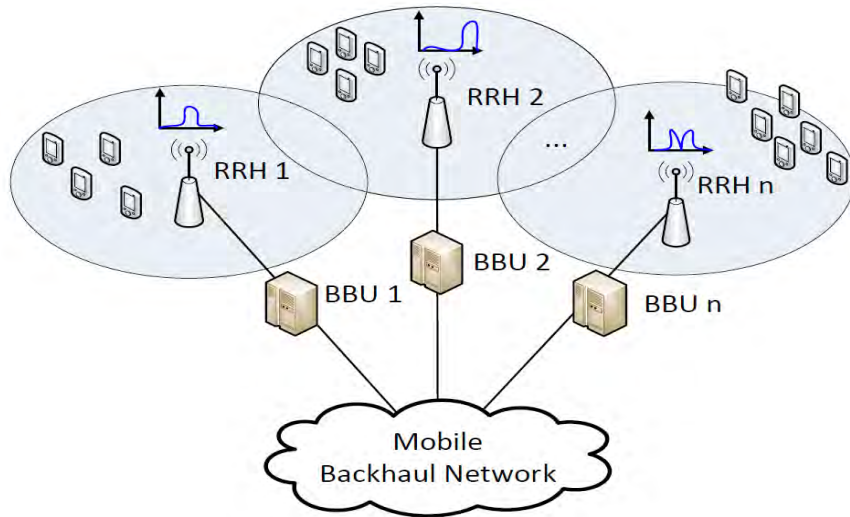
■ Clustering

- RRH aggregation and assignment to BBU pools
- Reduce the number of BBUS to save energy

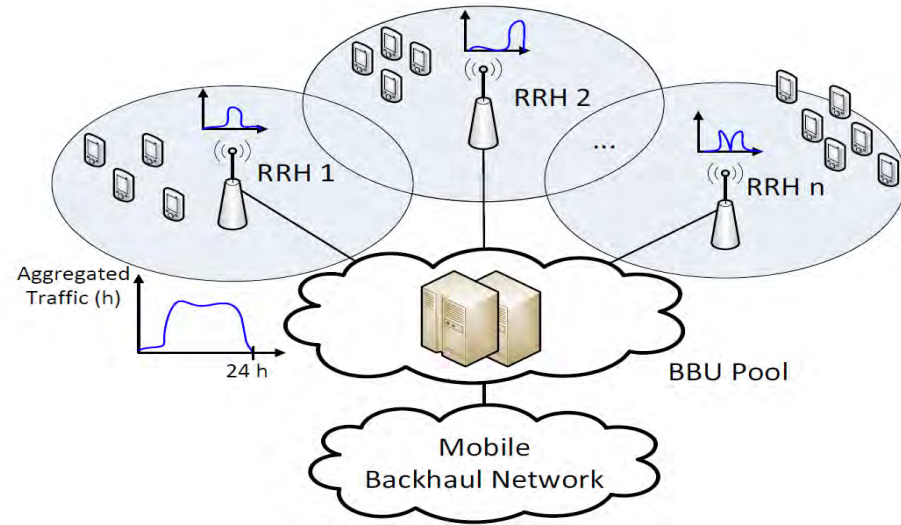
Benefit of a Cloudified RAN

Adapt to spatio-temporal traffic fluctuation

Distributed BS



C-RAN



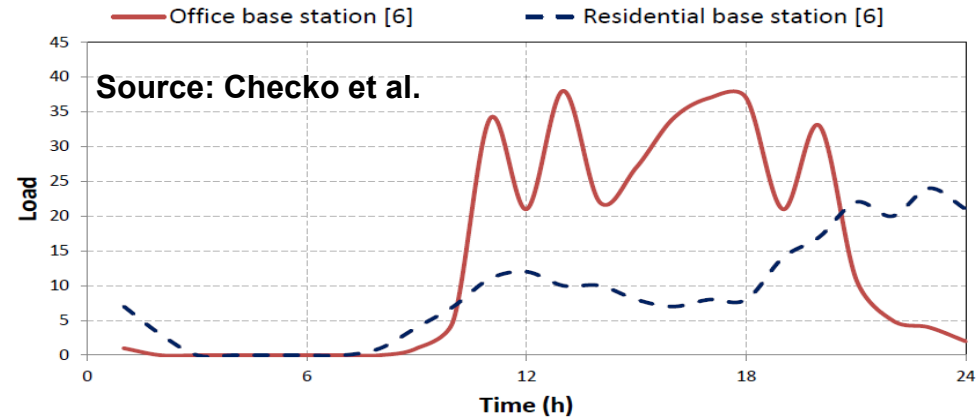
- **Statistical Multiplexing Gain**
- **Scalability**
- **Workload sharing**

Source: Checko et al.

Cloudified RAN Benefit

Exploit workload variations through the statistical multiplexing gain among multiple RAN

- BS are often dimensioned for the peak traffic load!
- Peak traffic load \rightarrow 10x off-the-peak hours
- Exception: load-aware BS

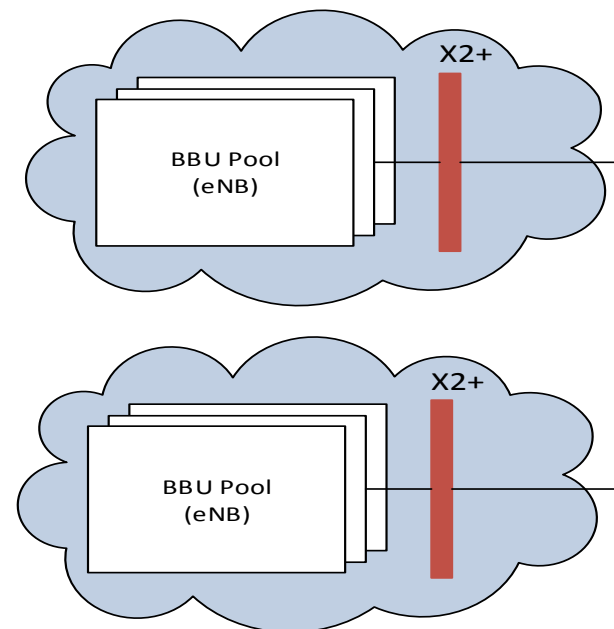


- **Observation:** Centralized BBUs' processing $<$ Σ of BSs' processing
- **Statistical multiplexing gain** = $\frac{\Sigma \text{ of BSs' processing}}{\text{Centralized BBUs' processing}}$
- **Gain:** depends on traffic pattern, BBU to RRH mapping, BBU load balancing

Benefit of a Cloudified RAN

Improve of spectral efficiency (throughput, latency)

- **Centralization of BBU pool in C-RAN facilitates the inter BBU cooperation**
 - **Joint scheduling**
 - ☞ minimize inter cell interference (e.g. eICIC)
 - **Joint and coordinated signal processing**
 - ☞ utilize interference paths constructively (e.g. CoMP, MU-MIMO)
 - **Shared Context**
 - ☞ reduce control plane signaling delay (e.g. handover via X2+)



GP-Cloud computing vs C-RAN applications

	GP-Cloud Computing	C-RAN
Data rate	Mbps, bursty,	Gbps, stream
Latency / Jitter	Tens of ms	< 1, jitter in ns
Lifetime of data	Long	Extremely short
Number of clients	Millions	Thousands – Millions
Scalability	High	Low
Reliability	Redundancy, load balancing	Redundancy, over-provisioning, Offloading / load balancing
Placement	Depends on the cost and performance	Constraints, multi-objectives, NF dependency
Time scale (operation, recovery)	Non-realtime	Realtime and low-latency

Technologies

Network-Function Virtualization

Mobile-Edge Computing

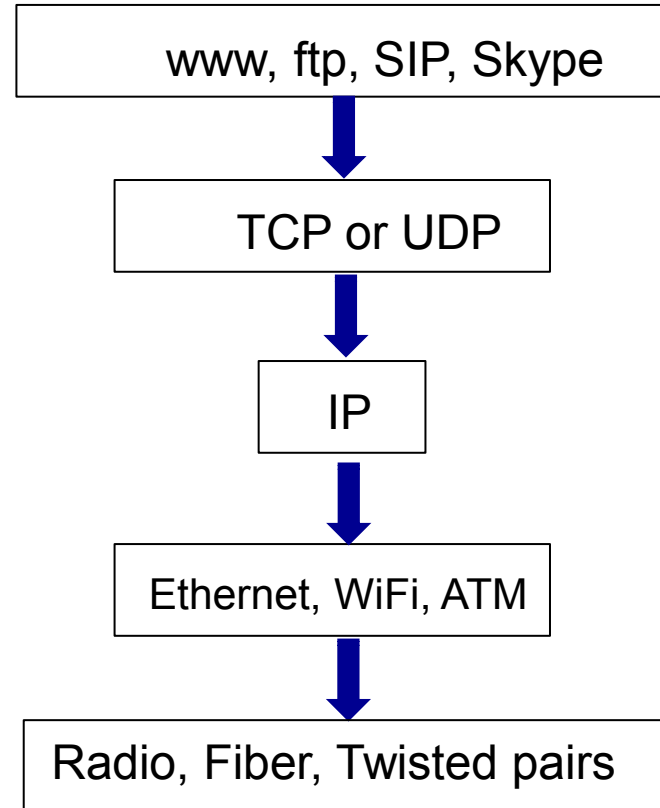
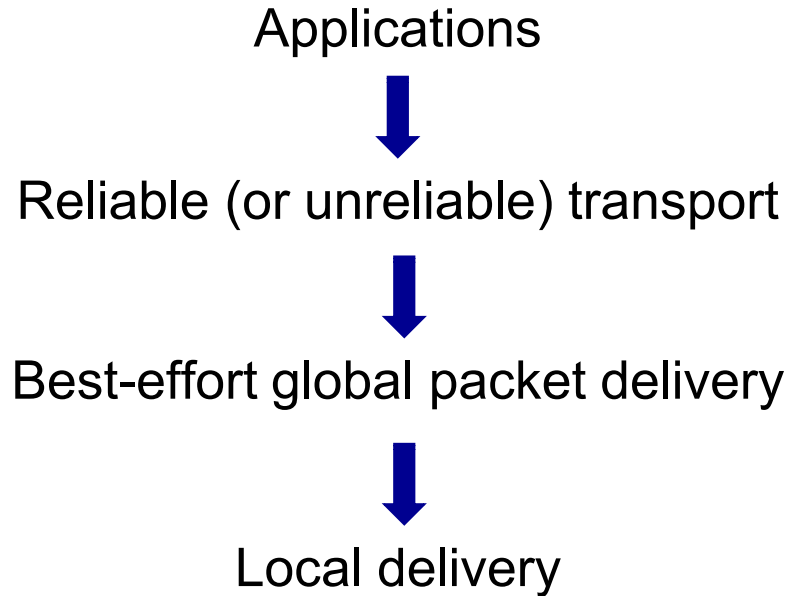
Software-Defined Network

RAN and CN Cloud

Networking planes

- **Data plane** processing and delivery of packets with local forwarding state
 - Forwarding state + packet header = forwarding decision
 - Nanosecond time scale, local
 - **Control plane** computes the state in routers (forwarding state)
 - Determine how and where packets are forwarded
 - Traffic Engineering (TE), firewall state
 - Distributed protocols, manual configuration
 - Millisecond/second time scale, global
- => These different planes require different abstraction**

Data plane abstraction: layers



Control plane abstraction?

- **Several control plan mechanisms:**
 - Routing: distributed routing algorithms
 - Isolation: ACLs, VLANs, Firewalls,
 - Traffic Engineering: MPLS, VPN
- **No modularity, limited functionality**
- **Control plane: mechanism without abstraction**
 - No modularity, no reusable mechanism

SDN: Two control plane abstractions

- **Abstraction: global network view**
 - Provides information about the current network
 - Implementation: “Network Operating System”
 - ☞ Runs on servers in network
- **Abstraction: forwarding model**
 - Provides standard way of defining forwarding state
 - Southbound API (ex. OpenFlow)

SDN Abstraction

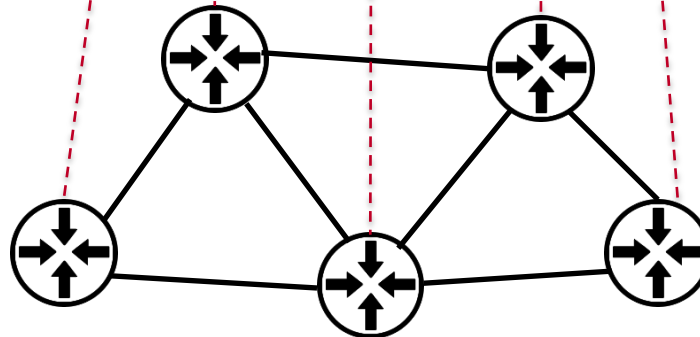
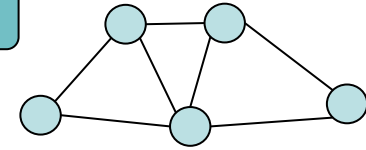
Control mechanism
expresses desired behaviour

Routing, ACLs, Load Balancing

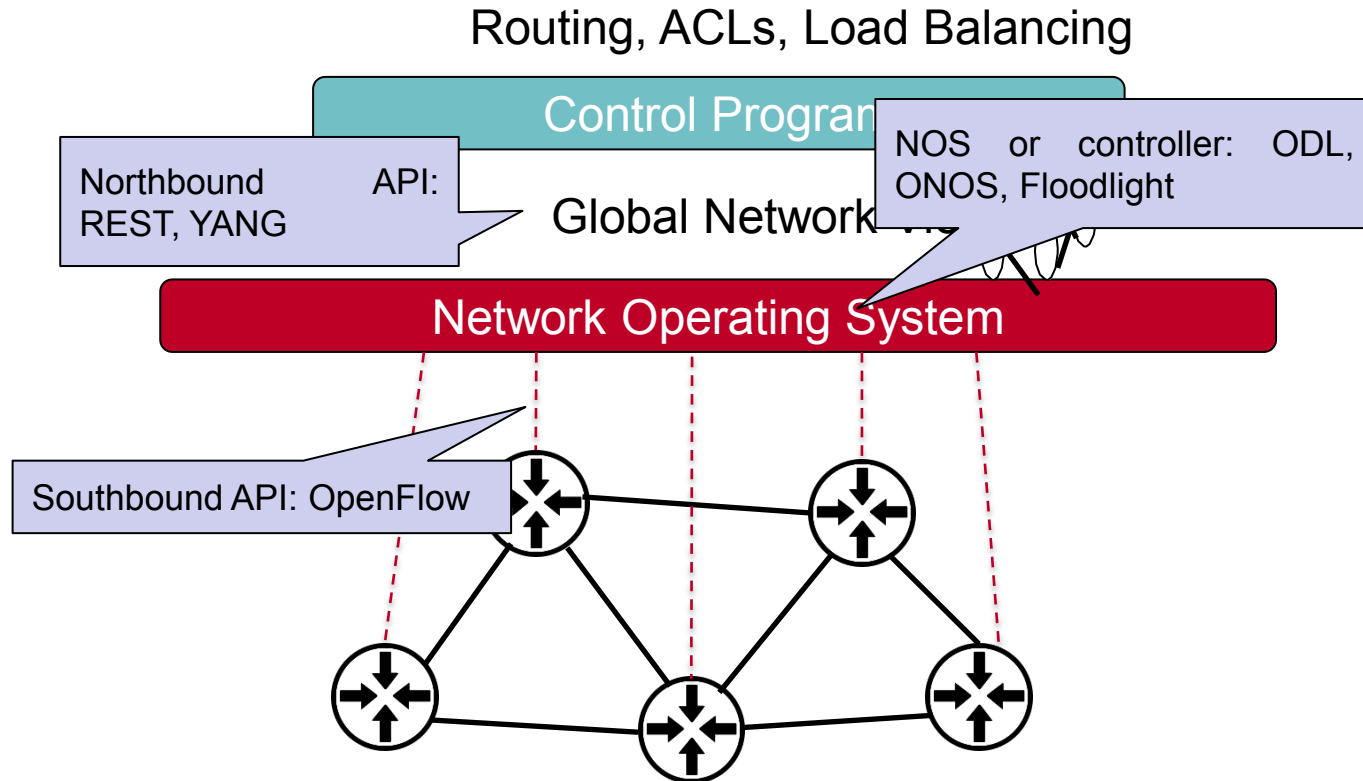
Control Program

Global Network View

Network Operating System

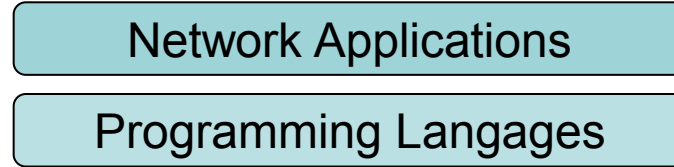
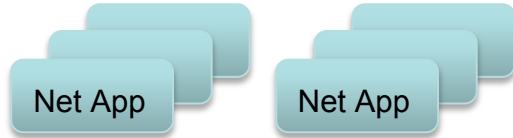


SDN Abstraction

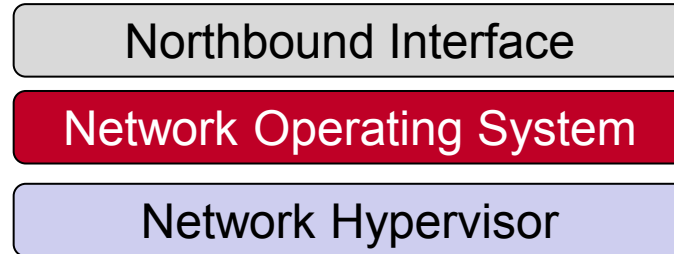


Layered SDN architecture

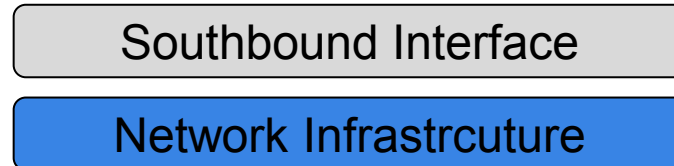
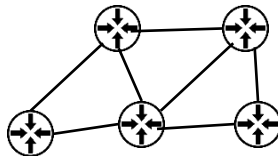
Management Plan



Control Plan



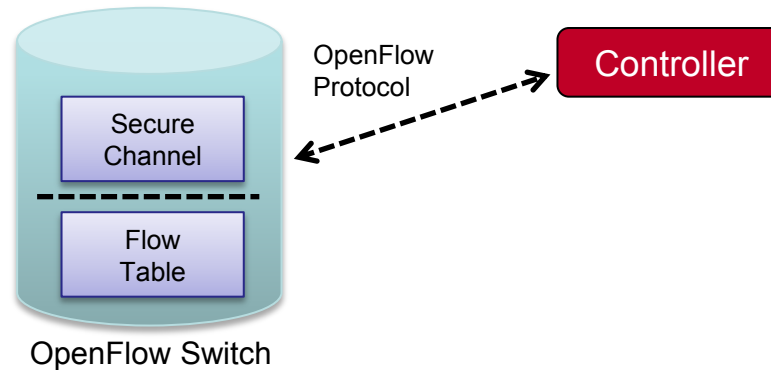
Data Plan



Southbound interfaces

- **Southbound interfaces (or southbound API) connect the control and forwarding elements**
- **OpenFlow is the most widely accepted and deployed open southbound standard for SDN**
- **But others exist:**
 - ForCES, OVSDB, POF, OpFlex, Netconf

- **Defined by Open Networking Foundation**
 - Major actors (Cisco, IBM, NEC, HP, Alcatel-Lucent, VMWare,...)
- **Principle: an OpenFlow controller communicates over a secure channel**
 - OpenFlow protocol defines messages format
 - Purpose of control channel: update flow table
 - Logic is executed at the controller



Programmable data plane: why?

OpenFlow

- Limited set of fields to match on (3.6 times more fields in 1.5 than 1.0)
- What about new protocols? And custom protocols
- What about statistics other than Packet, Byte count, Flow duration?
- What about statefull matching or forwarding logic?

Programmable data plane: why?

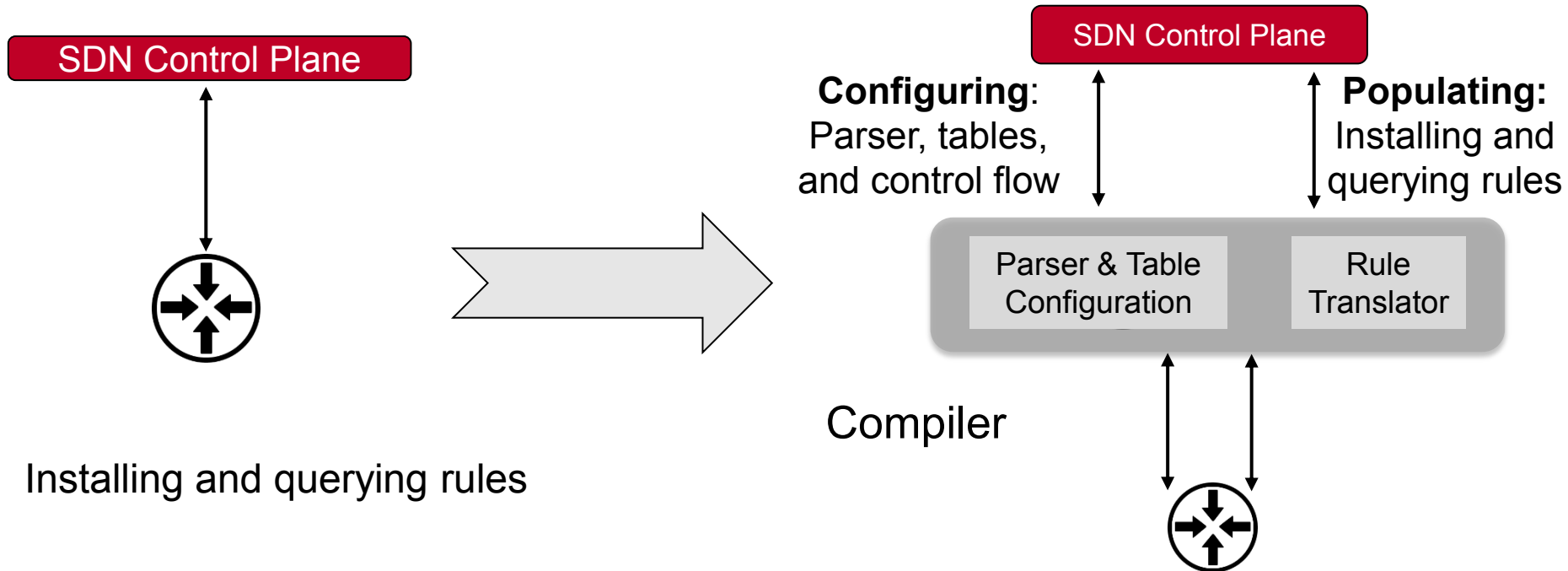
Version	Date	# Headers
OF 1.0	Dec 2009	12
OF 1.1	Feb 2011	15
OF 1.2	Dec 2011	36
OF 1.3	Jun 2012	40
OF 1.4	Oct 2013	41
OF 1.5	Dec 2014	44

Still not enough. New protocols are appearing!!

Data plane programmability

- **Program the data plane to achieve arbitrary matching and processing**
- **Protocol independence**
 - No knowledge of Ethernet, TCP, UDP,
 - Work with existing and future protocols
- **Reconfigure the switches on the fly**
 - No specific target hardware
- **Cautious : Do not try to support every existing protocol header fields**
 - Provide an instruction set suitable to match arbitrary protocols and fields
 - Execution of the instruction set is an implementation detail

OpenFlow v2 or programmable Data Plane



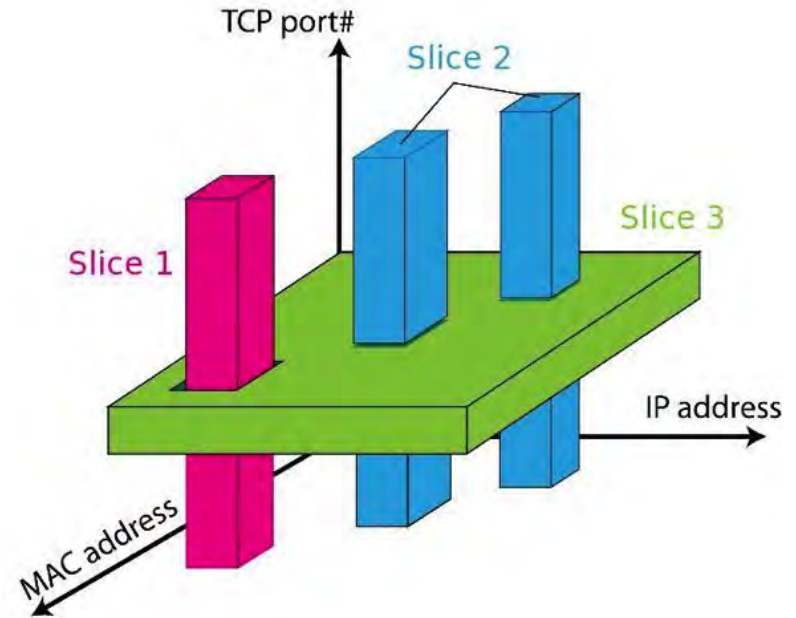
- **NETCONF is an IETF configuration management protocol designed to support management of configuration**
 - Distinction between configuration and state data
 - Configuration change validations
 - Configuration change transactions
 - Remote procedure call mechanism
- **In response to SNMP shortcoming for managing configuration**
- **YANG high level language designed to write data models for NETCONF**

FlowVisor: Virtualizing SDN Control

- An OpenFlow controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers
 - OpenVirteX (OVX)
- **Network Slice: divide the network into logical slices**
 - Each slice controls its own packet forwarding
 - Users chooses which slice controls their traffic

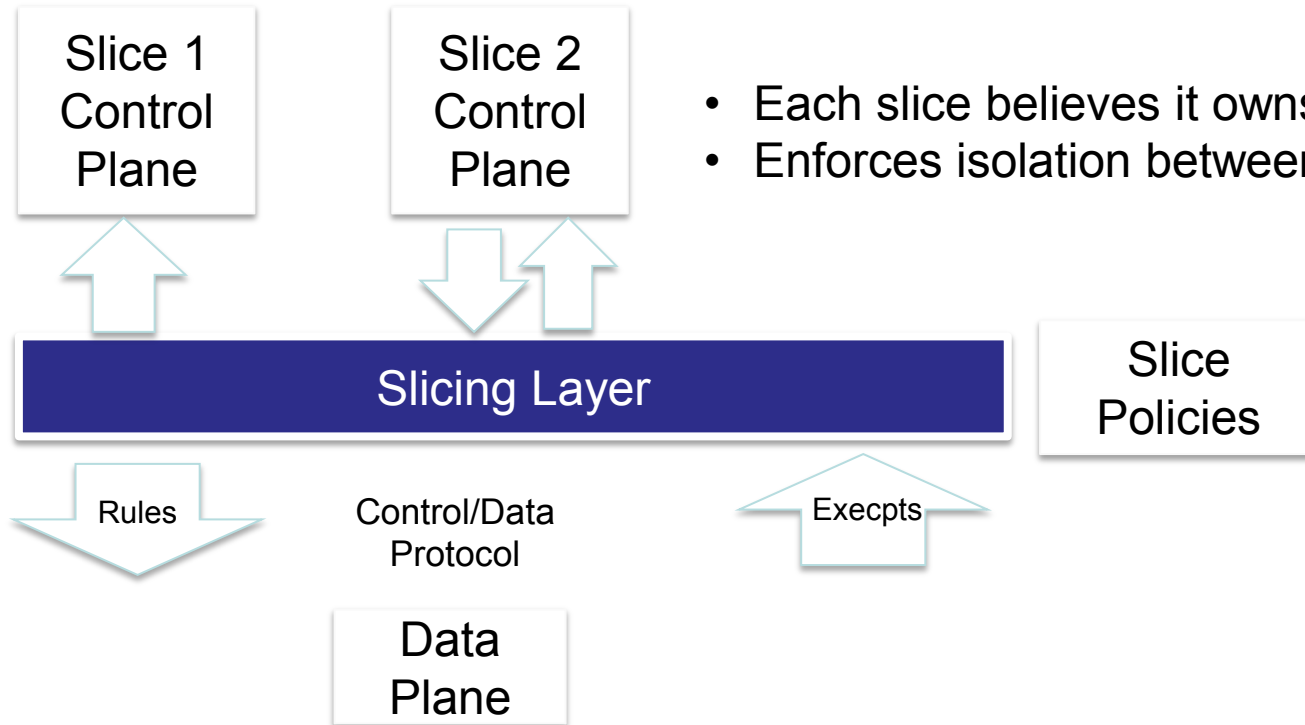
FlowVisor slices

- **Slices - any combination of**
 - switch ports (layer 1),
 - src/dst Ethernet addresses (layer 2),
 - src/dst IP addresses or type (layer 3)
 - src/dst TCP/UDP port of ICMP code/type (layer 4)



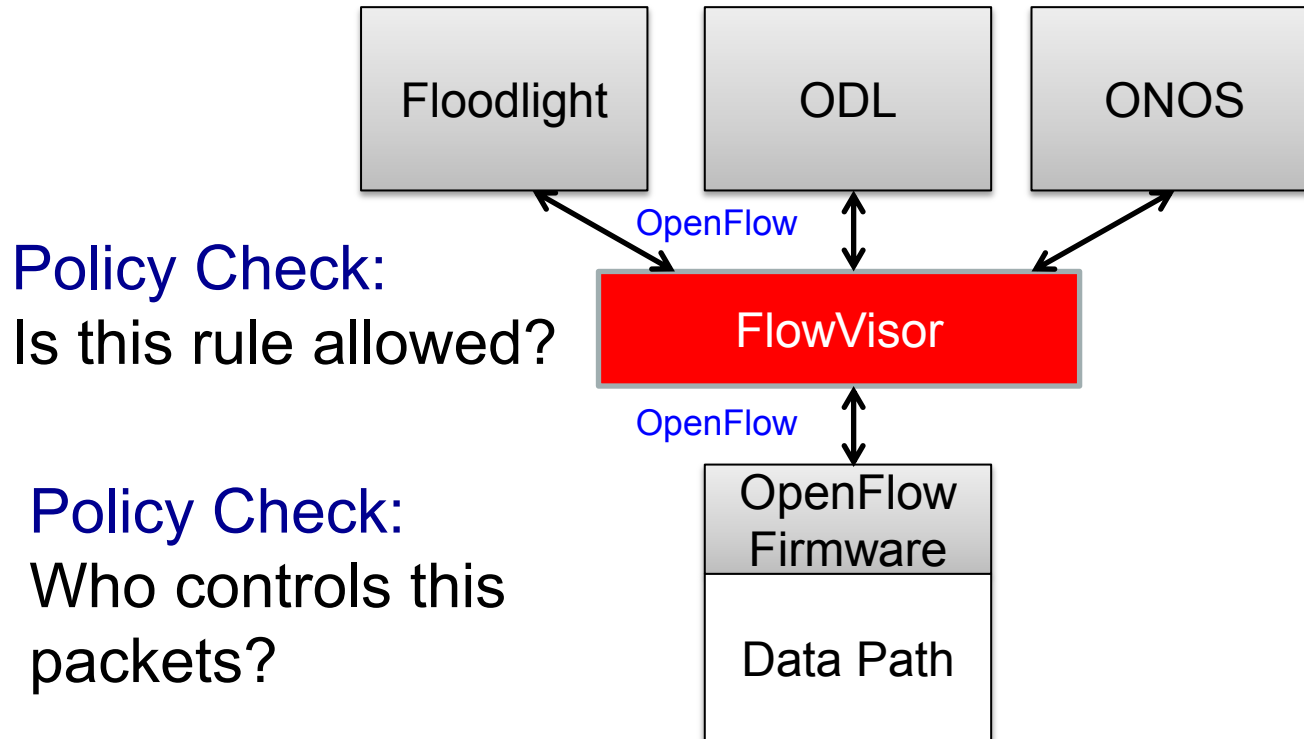
<https://openflow.stanford.edu/display/DOCS/Flowvisor>

FlowVisor slices



- Each slice believes it owns the data path
- Enforces isolation between slices

FlowVisor on OpenFlow



Network Operating System / Controller

- **Traditional operating systems provide**
 - abstractions (high-level programming API) to access low level devices, manage concurrent access to the underlying resources (e.g. hard drive, CPU, network adapter, memory)
- **In contrast, networks have so far been managed and configured using**
 - Low level, device-specific instruction sets and most closed proprietary network operating system
 - e.g. Cisco IOS, Juniper JunOS
- **NOS aims to provide abstraction and common API to developers.**
- **Generic functions and services**
 - Network state, Network topology information, device discovery, distribution of network configuration

- **NOS (or controller) is a critical element in an SDN architecture**
 - Support the control logic (applications) to generate the network configuration
 - ☞ Based on policies defined by Network Operator
- **Similar to the traditional OS, the control platform abstracts the low level details**
 - Connecting and interacting with forwarding devices

SDN Controllers

- **Several controllers and NOS exist**
- **Many considerations**
 - Base Network services
 - ☞ Topology, stats, switch manager, ...
 - Southbound API
 - ☞ OpenFlow, OVSDB, OpenFlex, other.
 - Northbound API
 - ☞ REST, JSON, etc.
 - Integration Plug-ins
 - ☞ OpenStack
 - Management interfaces
 - ☞ GUI/CLI

SDN Controllers – A comparison

- Many SDN controller POX, NOX, OpenDayLight, ONOS, ARIA

Use-Cases \ Controllers	Trema	Nox/Pox	RYU	Floodlight	ODL	ONOS***
Network Virtualization by Virtual Overlays	YES	YES	YES	PARTIAL	YES	NO
Hop-by-hop Network Virtualization	NO	NO	NO	YES	YES	YES
OpenStack Neutron Support	NO	NO	YES	YES	YES	NO
Legacy Network Interoperability	NO	NO	NO	NO	YES	PARTIAL
Service Insertion and Chaining	NO	NO	PARTIAL	NO	YES	PARTIAL
Network Monitoring	PARTIAL	PARTIAL	YES	YES	YES	YES
Policy Enforcement	NO	NO	NO	PARTIAL	YES	PARTIAL
Load Balancing	NO	NO	NO	NO	YES	NO
Traffic Engineering	PARTIAL	PARTIAL	PARTIAL	PARTIAL	YES	PARTIAL
Dynamic Network Taps	NO	NO	YES	YES	YES	NO
Multi-Layer Network Optimization	NO	NO	NO	NO	PARTIAL	PARTIAL
Transport Networks - NV, Traffic-Rerouting, Interconnecting DCs, etc.	NO	NO	PARTIAL	NO	PARTIAL	PARTIAL
Campus Networks	PARTIAL	PARTIAL	PARTIAL	PARTIAL	PARTIAL	NO
Routing	YES	NO	YES	YES	YES	YES

Source : thenewstack.io

Northbound API

- **North- and southbound interfaces are two keys abstraction of the SDN ecosystem**
- **Southbound interface has already a widely accepted proposal (OpenFlow)**
- **Common Northbound API is still an open issue**
- **What is Northbound API?**
 - API allowing application and orchestration systems to program the network

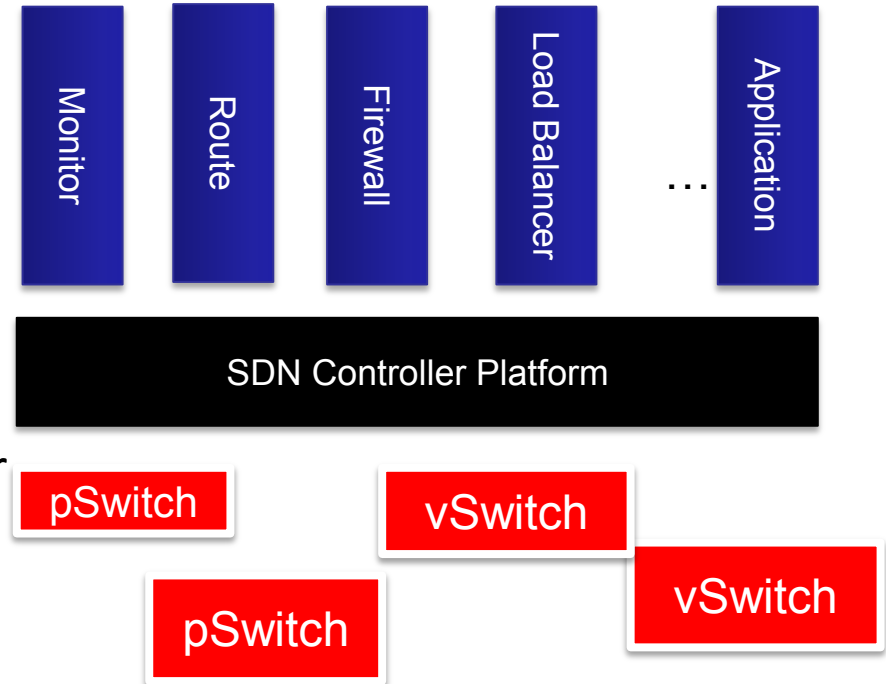
Northbound API

■ Uses for Northbound API

- Path computation
- Loop avoidance
- Routing
- Security
- Statistics/monitoring

■ Used by:

- Sophisticated network operator
- Service providers
- Vendor
- Anyone who want to develop on top of southband API



Northbound API: REST

- **Most of the NOS uses REST (REpresentation State Transfer) or RESTful API as Northbound interface**
- **Initially for web services**
 - REST describes the Web as a distributed hypermedia application whose linked resources communicate by exchanging representations of resource state
- **The aim of NOS is to provide a REST API that should translate high level rules to OpenFlow rules**

Northbound API: example

JSON description of adding a flow through a REST API

```
0b={
  "flow": {
    "priority": 30000,
    "idle_timeout": 60,
    "match": [
      {"eth_type": "ipv4"},
      {"ipv4_src": "10.0.0.1"},
      {"ipv4_src": "10.0.0.22"}
    ],
    {"ip_proto": "tcp"},
    {"tcp_dst": "80"}
  ],
  "actions": [{"output": 1}]
}
```

Push (curl tool) to the Controller through a web HTTP POST method

```
curl -sk -H "X-Auth-Token:$token" -X POST -H 'Content-Type:application/json' -d $ @0b https://192.168.56.7:8443/sdn/v2.0/of/datapaths/00:00:00:00:00:0b/flows
```

Programming Network: Pyretic

- **Network program language and Runtime**

Language

Provide Programming API

Runtime

Compile Policy to OpenFlow Rules

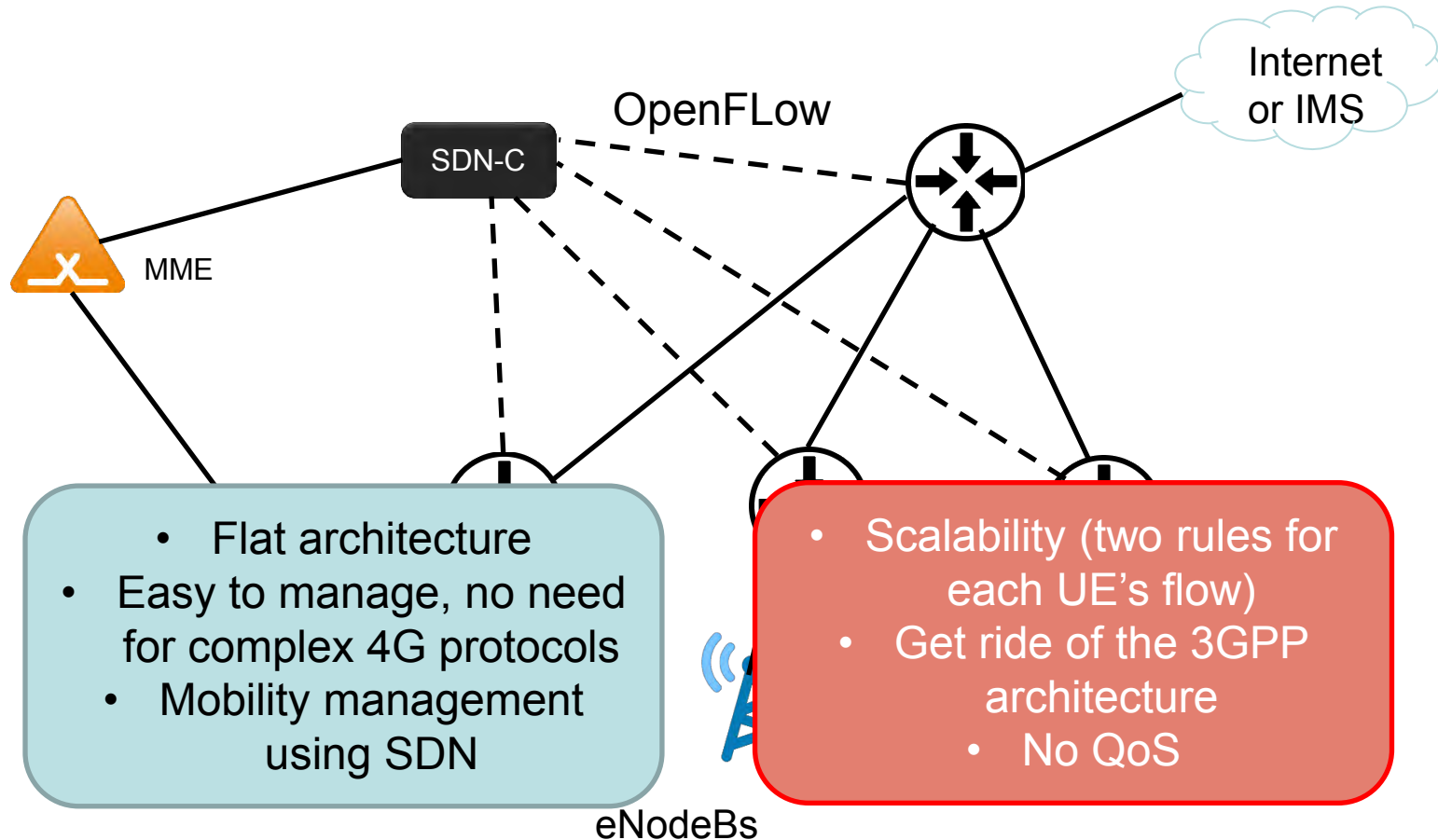
- **Enable programmers to specify policies on located packets**
 - Packet and their location (switch and port)

Pyretic: policy composition

- **Ex. `match(switch=s1,inport=2,srcip='1.2.3.4')[flood]`**
- **Sequential composition: perform one operation, then the next (e.g. firewall then route)**
`match(dstip=2.2.2.8) >> fwd(1)`
- **Parallel composition: perform both operation simultaneously (e.g. counting, route)**
`match(dstip=2.2.2.8) >> fwd(1) | count()`

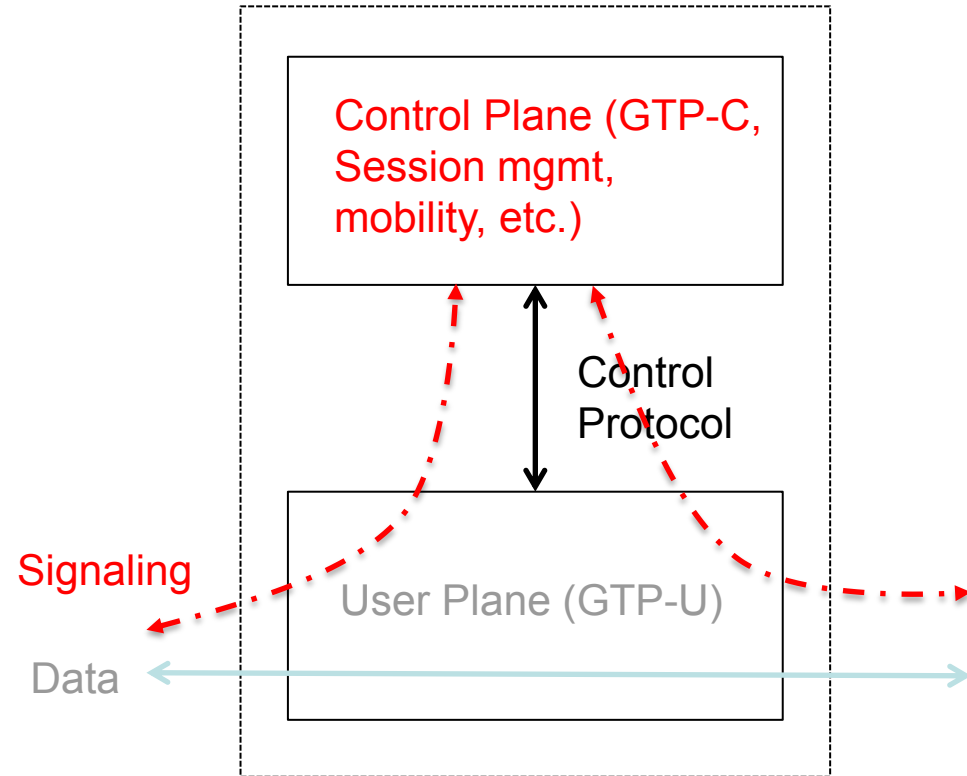
Deployment Example: EPC

New vision



Compatible with the 3GPP architecture

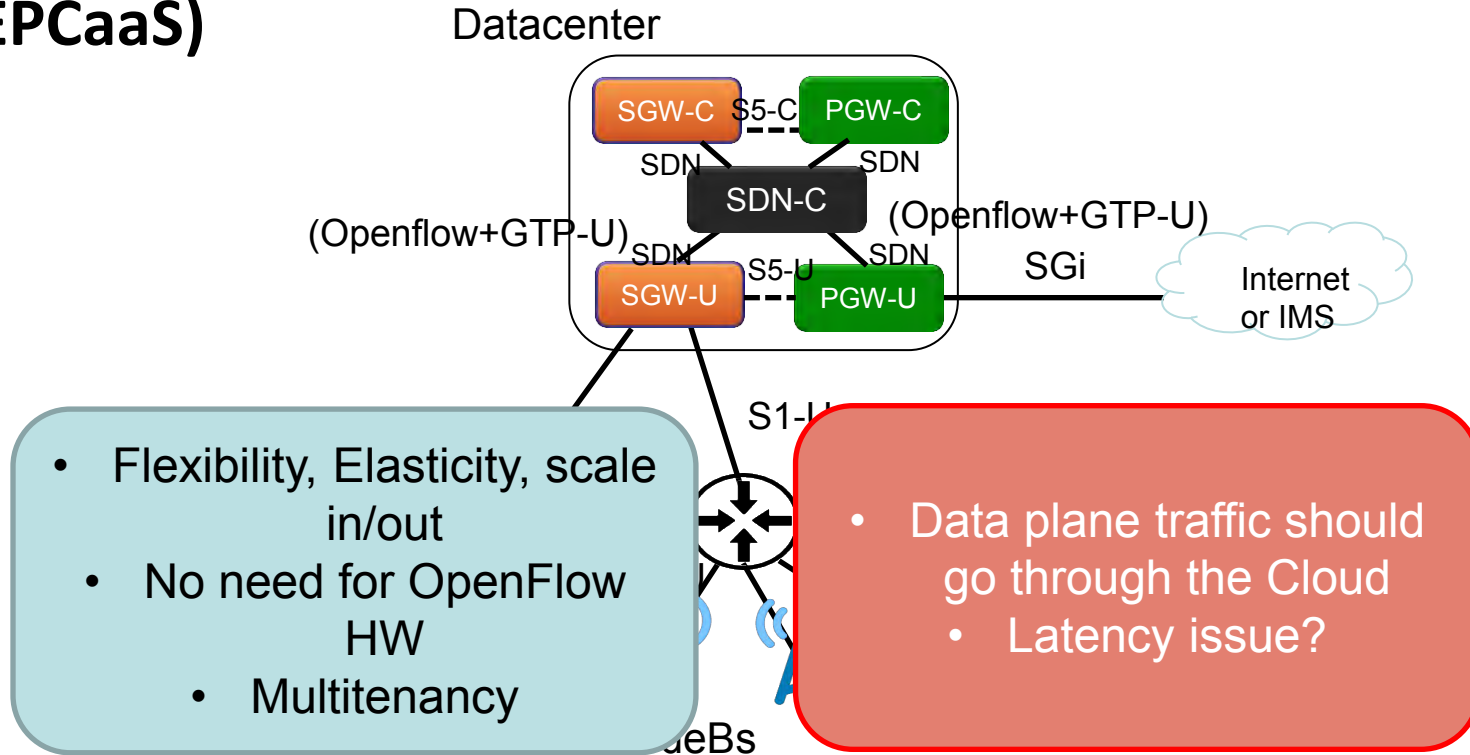
- **SDN Separation of the control and user plane => separate functions of the gateways (SGW and PGW)**
 - S/P-GW-C: manage control plane functions (i.e. session management, mobility, GTP-C)
 - S/P-GW-U: manage user plane functions (mainly forwarding traffic, GTP-U)
 - ☞ 3GPP CUPS group: working on the gateway functions separation issues



62

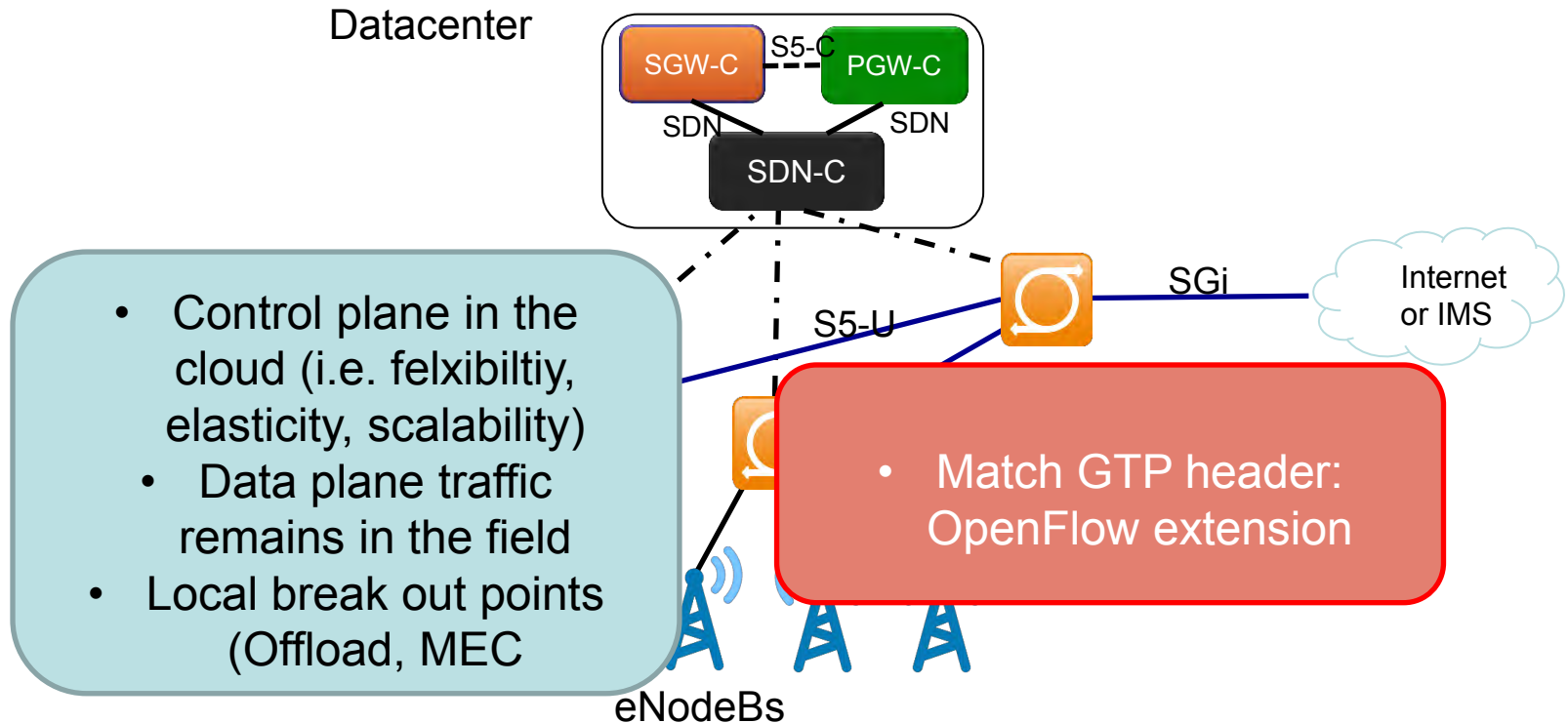
Scenario n°1

- All functions are virtualized and hosted in the Cloud (EPCaaS)



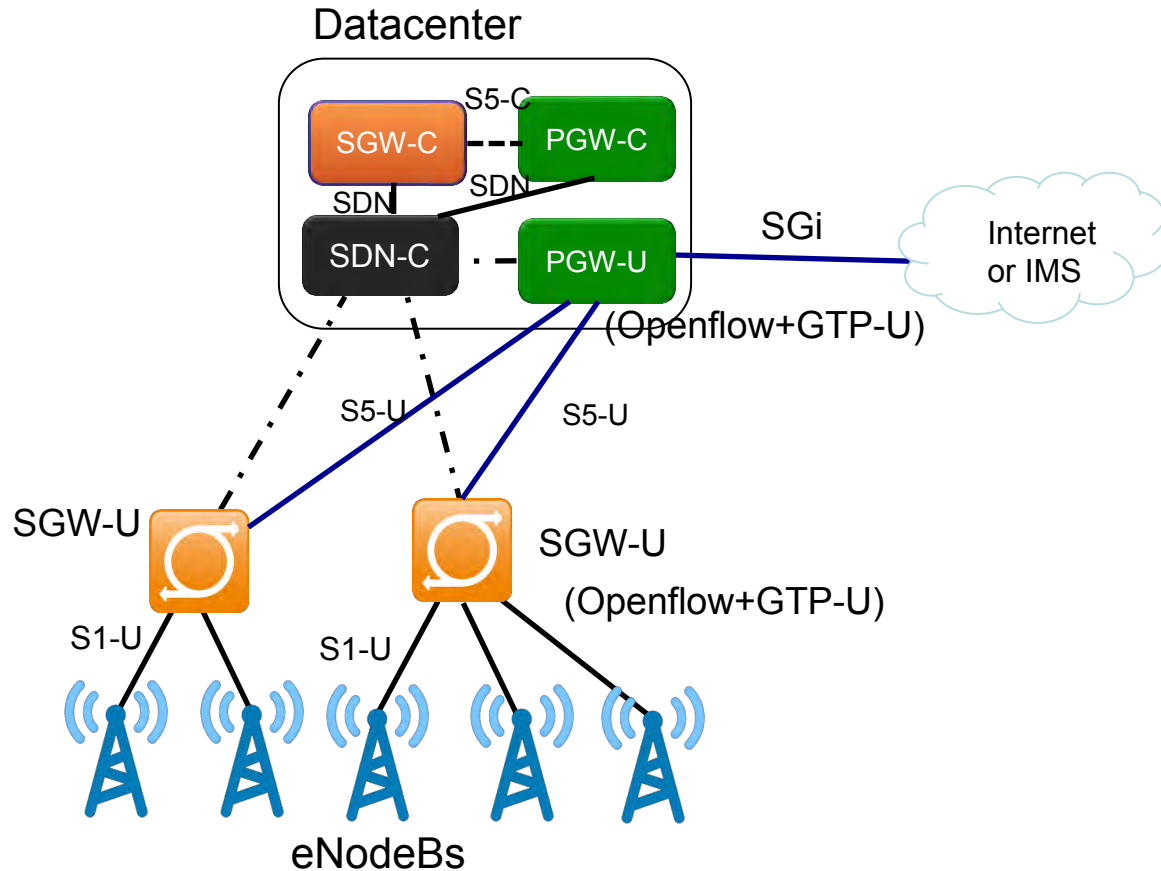
Scenario 2.

■ Control plane functions are virtualized



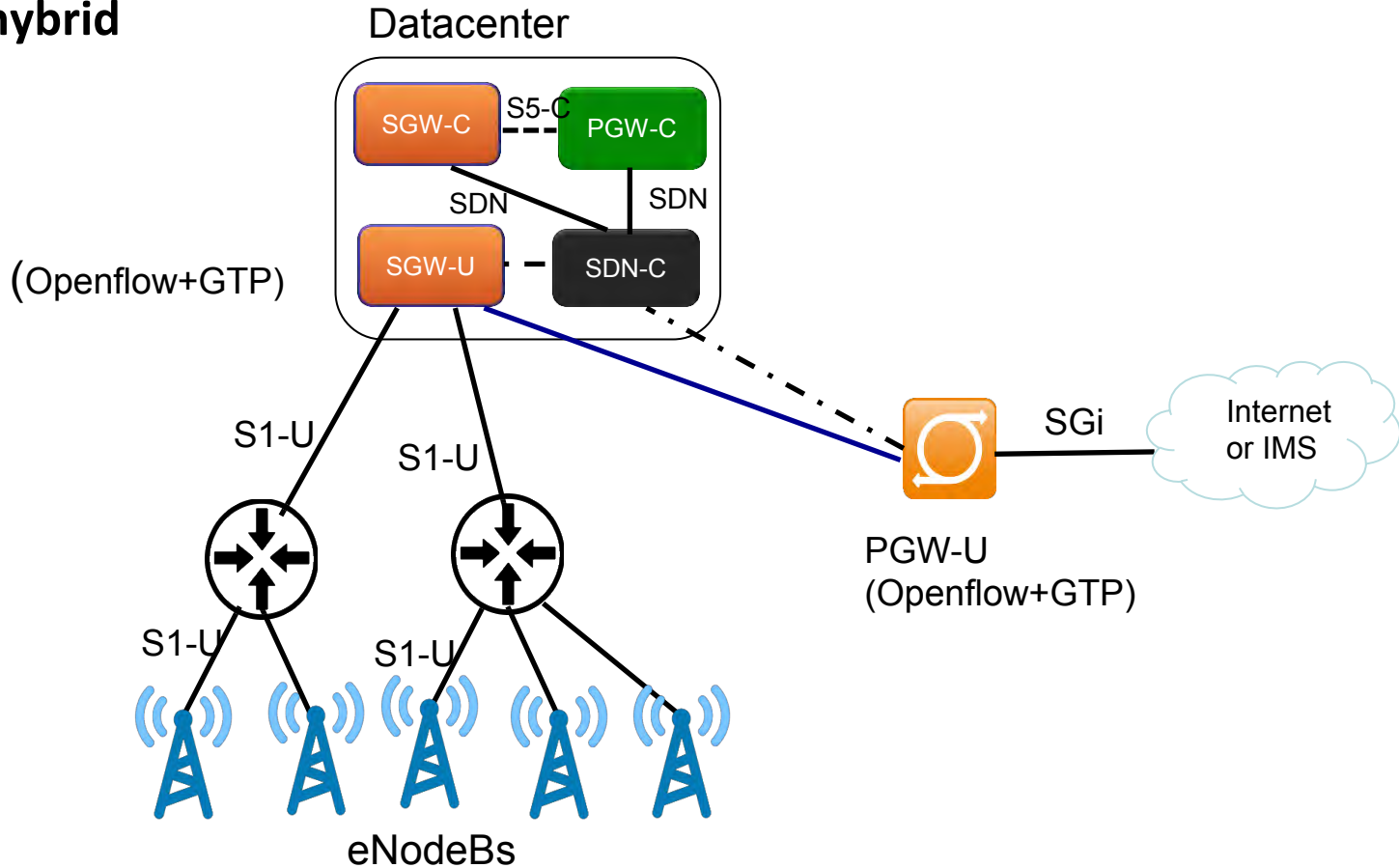
Scenario. 3

- Mix or hybrid



Scenario. 3

- Mix or hybrid



Technologies

Network-Function Virtualization

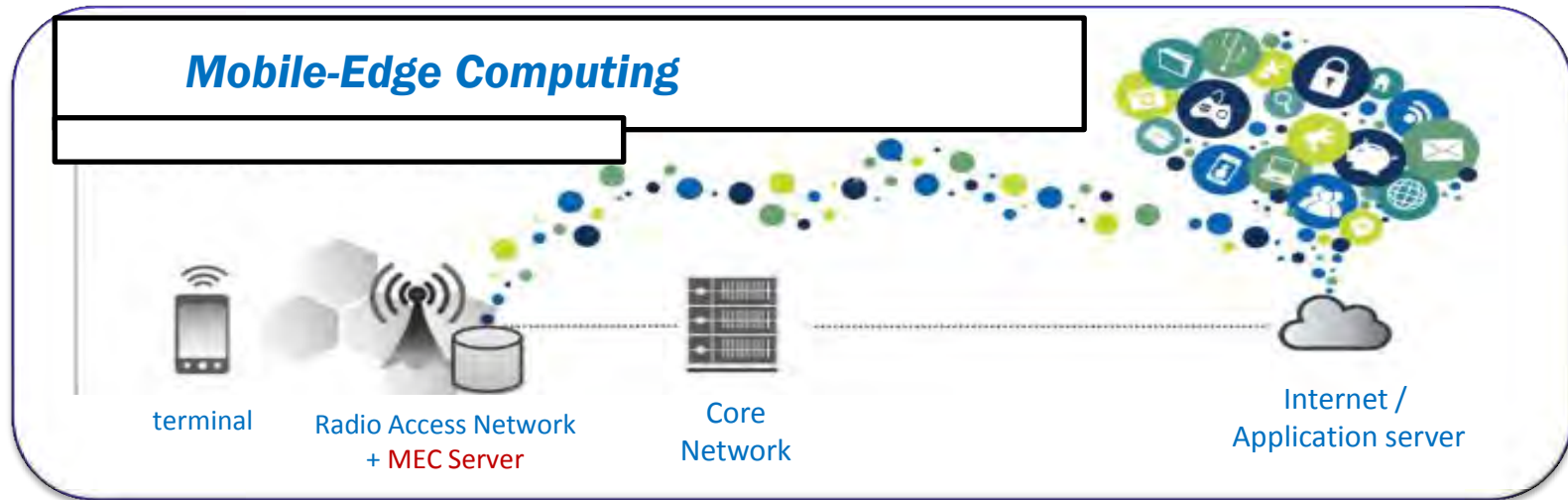
Mobile-Edge Computing

Software-Defined Network

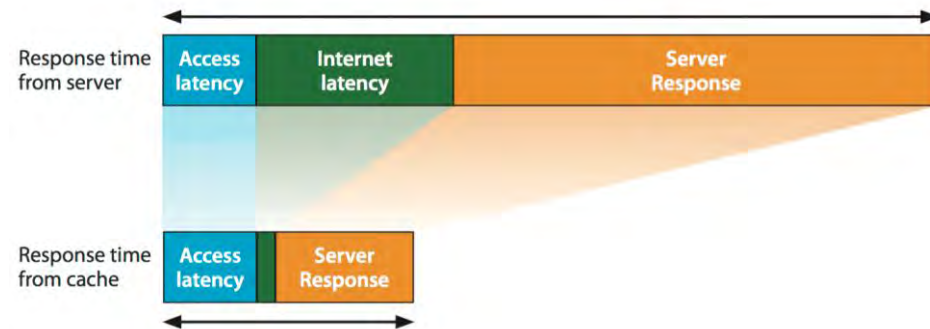
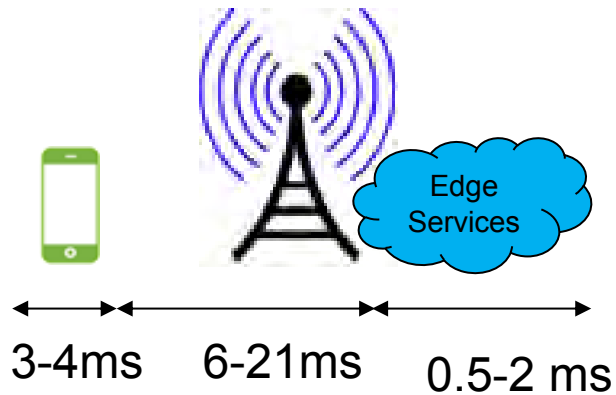
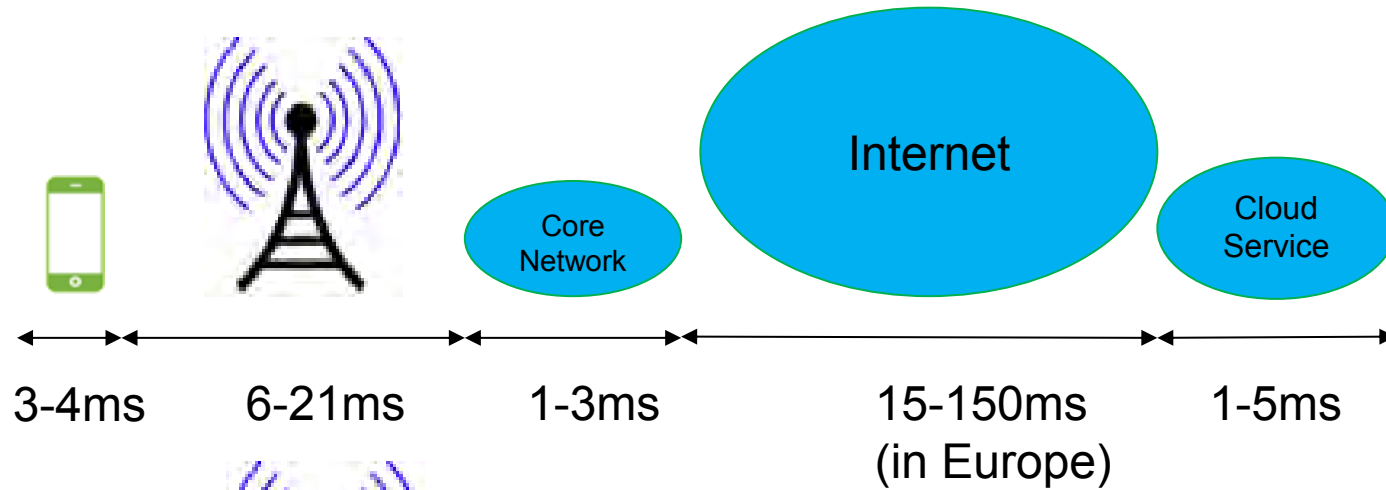
RAN and CN Cloud

Mobile Edge Computing

- **Bringing application, service and content closer to the user**
 - low-latency and high-bandwidth service deployed at the network edge
 - Direct access to **real-time radio and context information** (e.g. radio status, statistics, terminal info)
- **Local cloud-computing capabilities and an IT service environment at the edge of the mobile network**
 - Open the RAN edge to authorized third-parties, allowing them to flexibly and rapidly deploy innovative applications and services towards mobile subscribers, enterprises and vertical segments.



Mobile Edge Computing



Cloud Computing vs Edge Computing

- **Edge Computing extends the cloud computing paradigm to the edge of the network**
 - Shares many of the Cloud mechanisms and attributes: [Virtualization](#), [multi-tenancy](#)
- **Edge Computing addresses applications that require low latency access to servers**

Characteristics	Cloud Computing	Edge Computing
Major applications	Most of current cloud-involved applications	IoT, VR, Smart Home, Smart Cities, smart Vehicle
Availability	A small number of large sized data centers	A large number of small-sized datacenters
Proximity	Far from users	At the edge close to users
End-to-end latency	High due to the distance between DC and edge	Low
Network bandwidth	High	Low

➤ **Technologies: FoG (Cisco) and MEC (Telco. ETSI)**

Mobile Edge Computing: Real-Time Radio Network

Radio Access

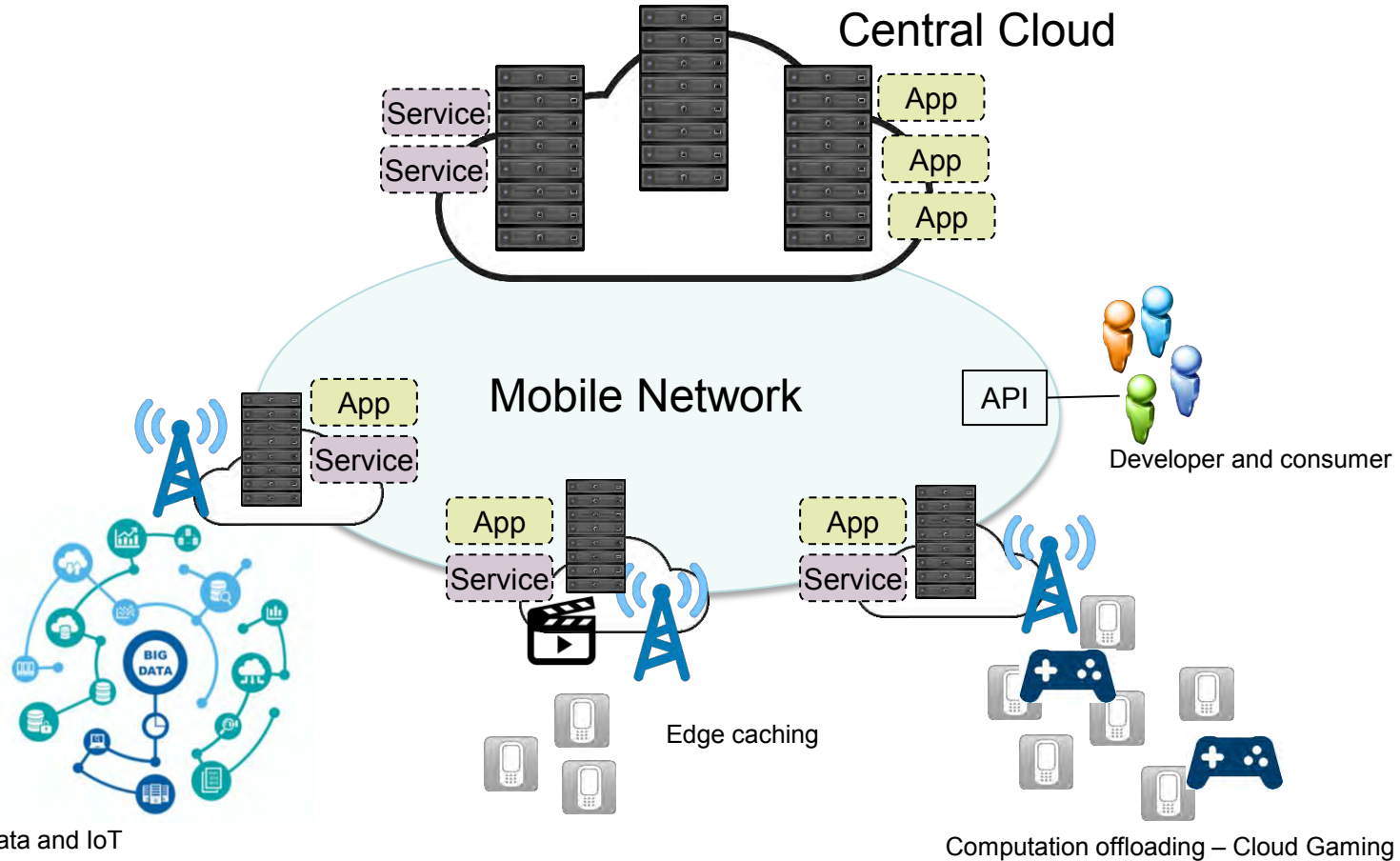


Operators can control the network allowing them to

Radio API – OAI MEC

- **UE configuration information:** PLMN ID, C-RNTI, Downlink (DL)/Uplink (UL) bandwidth, DL/UL carrier frequency, UE category.
- **UE status information:** Global Navigation Satellite System (GNSS) info (timing/phase), Angle of Arrival (AoA), System Frame Number (SFN), Buffer Status Report (BSR), Timing Advanced, Serving cell RSRP/RSRQ/RSSI, Non-serving cell DL RSRP/RSRQ/RSSI, Device-to-Device (D2D) transmission/reception frequency, D2D Destination info List.
- **eNB configuration information:** DL/UL radio bearers configuration, Aggregated DL/UL bandwidth, Tracking Area Code (TAC), DL/UL carrier frequency, PLMN identity.
- **eNB status information:** Global Navigation Satellite System (GNSS) info (timing/phase), Angle of arrival (AoA), System Frame Number (SFN), transmit power, DL/UL scheduling info, wideband and subband channel quality, buffer occupancy, Timing Advanced, Number of active UEs, UL interference level.

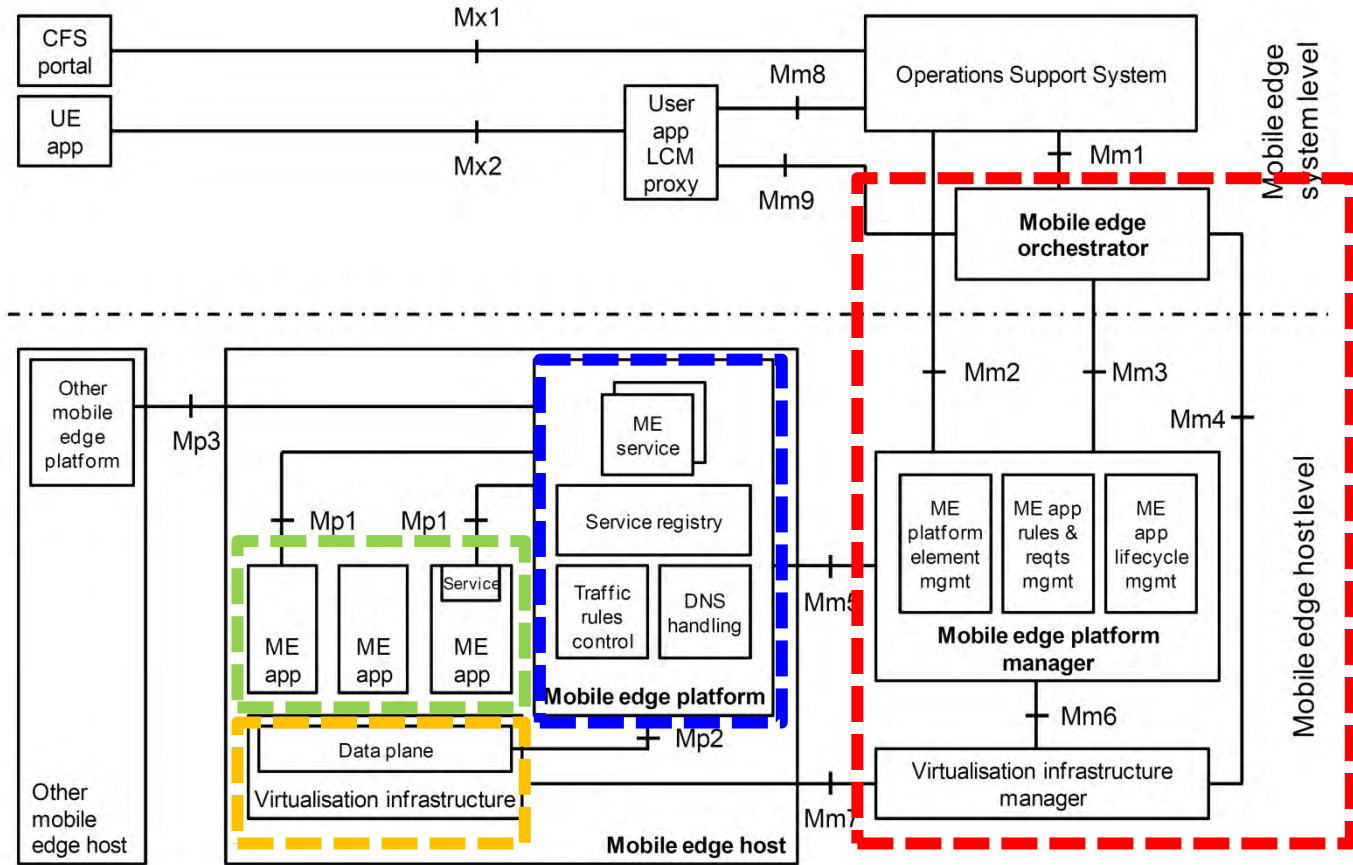
MEC – global picture



Big data and IoT

Computation offloading – Cloud Gaming

ETSI MEC Reference Architecture



Technologies

Network-Function Virtualization

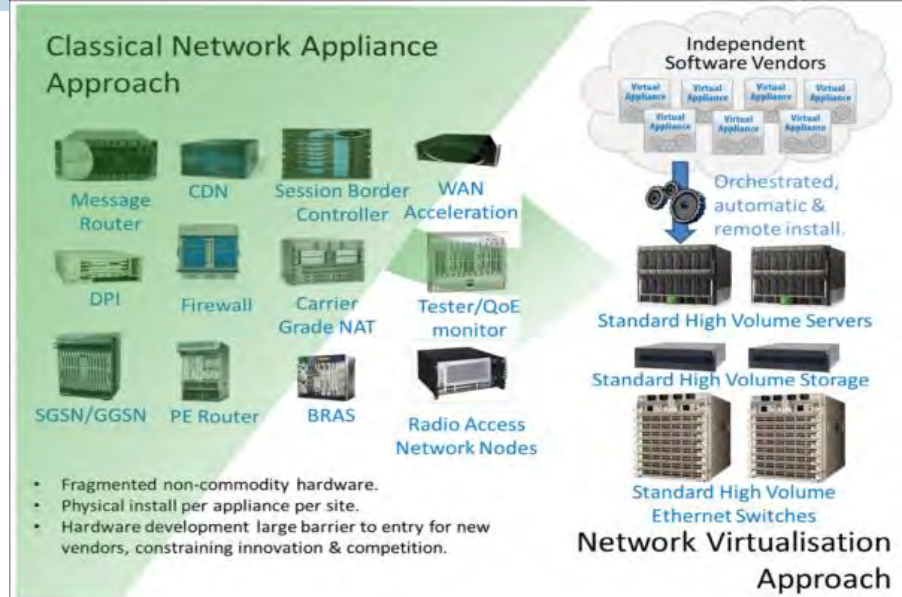
Mobile-Edge Computing

Software-Defined Network

RAN and CN Cloud

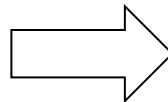
Network Function Virtualization

- Virtualization is the act of creating and running a software on a virtualized execution environment
- NFV breaks a network node into a set of atomic virtualized network functions (VNF) that can be composed and chained together to deliver an E2E service



Single service single
tenant network node

Dedicated HW platform
and SW control



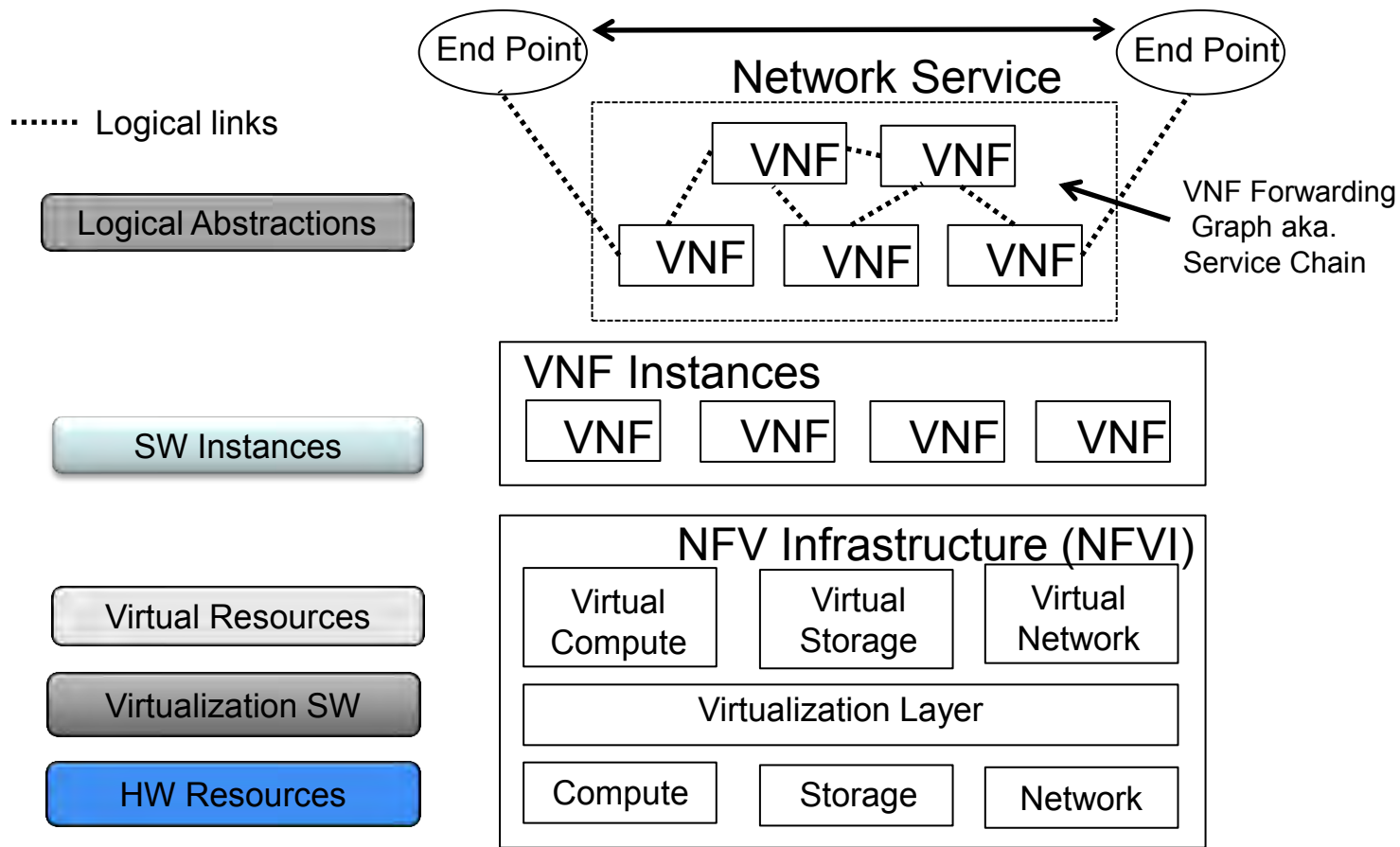
multi-service multi-tenant
network node

General purpose HW

ETSI NFV Architecture

- **Network Function (NF):**
 - Functional building block with a well defined interfaces and well defined functional behavior
- **Virtualized Network Function (VNF):**
 - Software implementation of NF that can be deployed in a virtualized infrastructure
- **VNF Forwarding Graph:**
 - Service chain when network connectivity order is important, e.g. firewall, NAT, load balancer
- **NFV Infrastructure (NFVI):**
 - Hardware and software required to deploy, manage and execute VNFs including computation, networking and storage

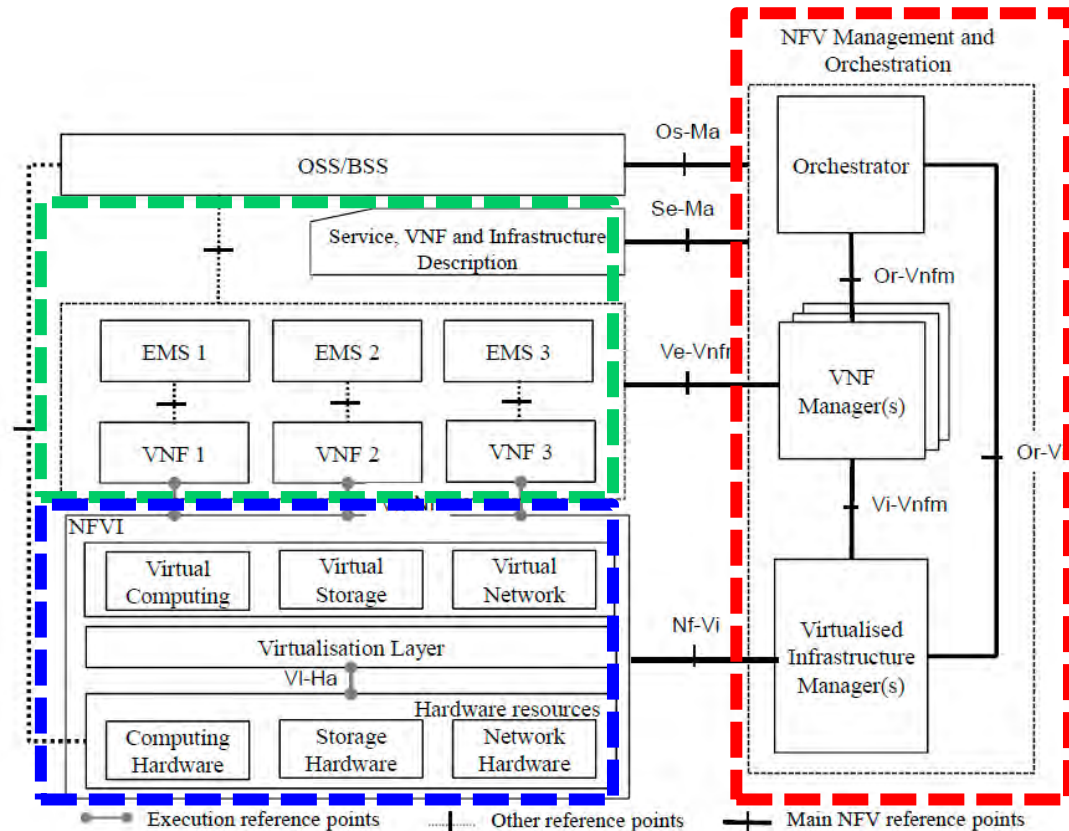
NFV Components



ETSI NFV Architecture

NFV management and Orchestration

- **NFV Orchestrator:**
(a) E2E network service and lifecycle management, (b) global resource management, and (c) policy management
- **VNF Manager :**
lifecycle management of VNF instances and overall coordination and adaptation role for configuration and event reporting
- **Virtual Infra. management (VIM)**
(a) controlling and managing the compute, storage and network resources, within one domain, and (b) collection and forwarding of performance measurements and events

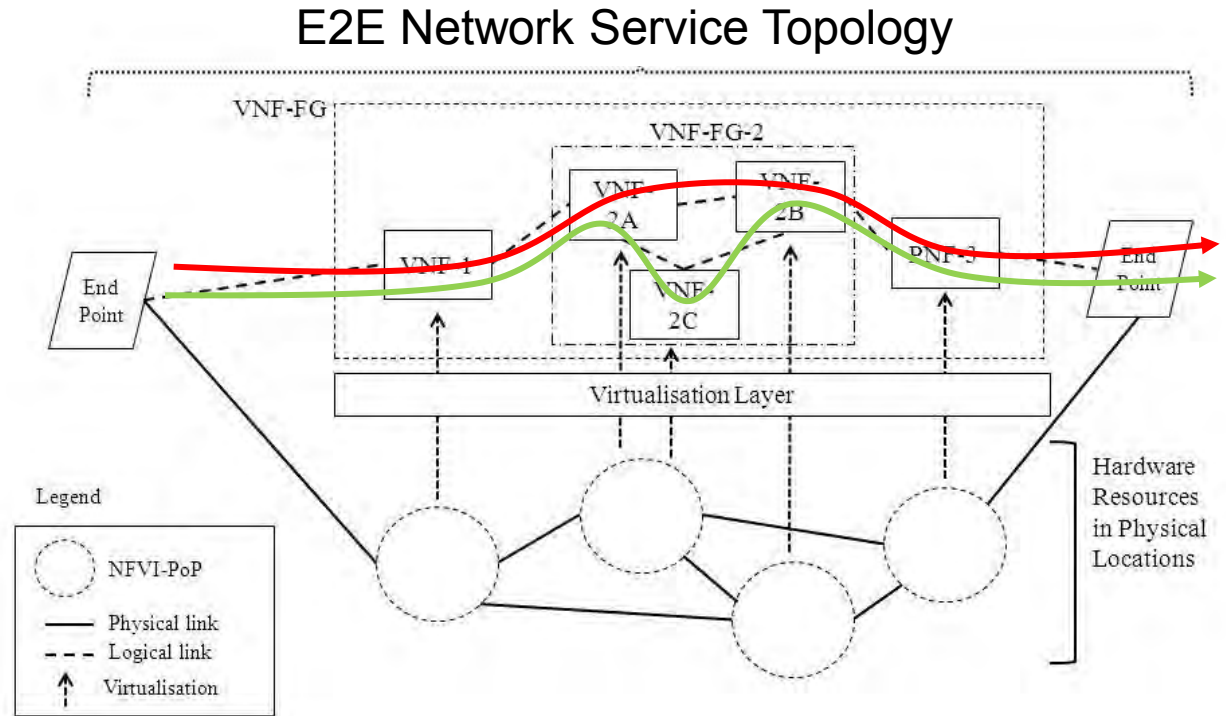


NFV and SDN

- **NFV: re-definition of network equipment architecture**
- **NFV was born to meet Service Provider (SP) needs:**
 - Lower CAPEX by reducing/eliminating proprietary hardware
 - Consolidate multiple network functions onto industry standard platforms
- **SDN: re-definition of network architecture**
- **SDN comes from the IT world:**
 - Separate the data and control layers, while centralizing the control
 - Deliver the ability to program network behavior using well-defined interfaces

Service Function Chaining (SFC)

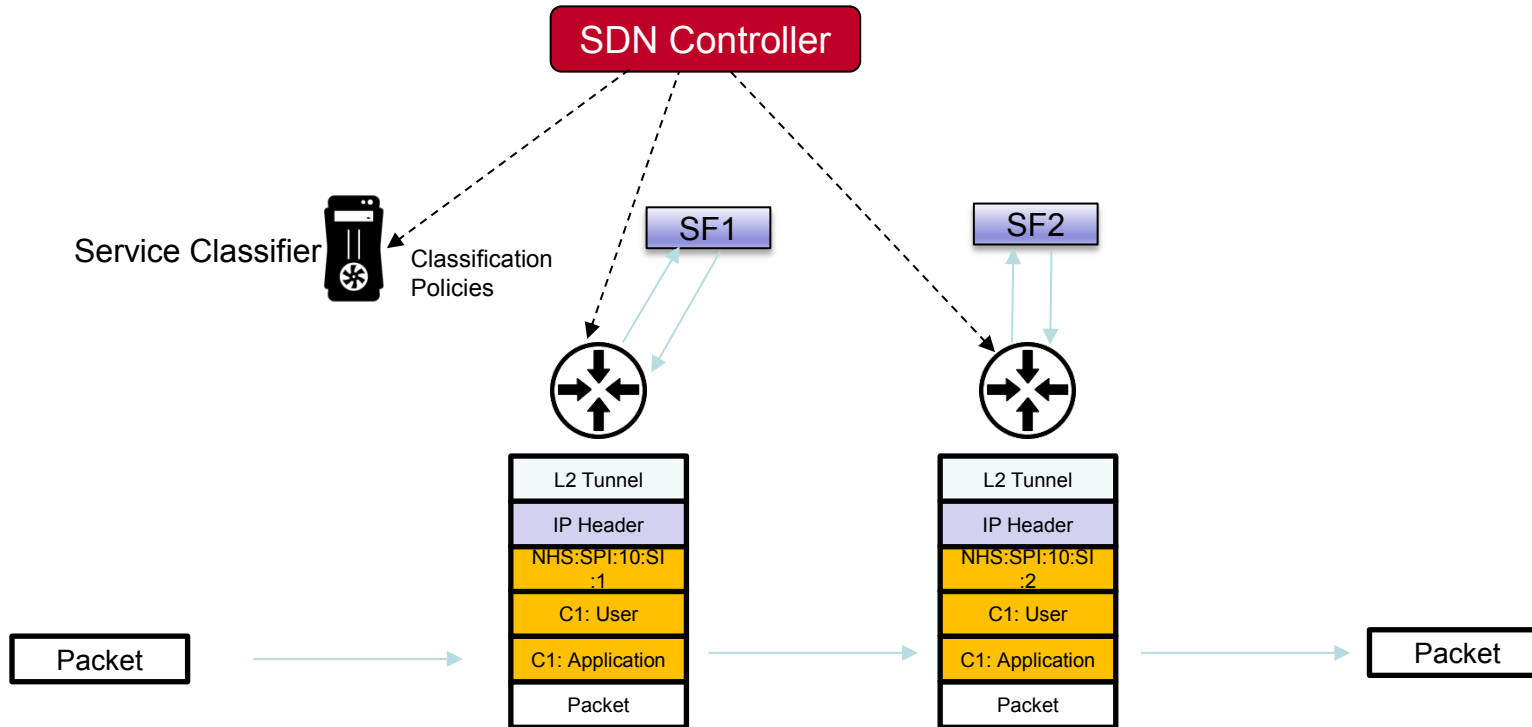
- An end-to-end service may include nested forwarding graphs



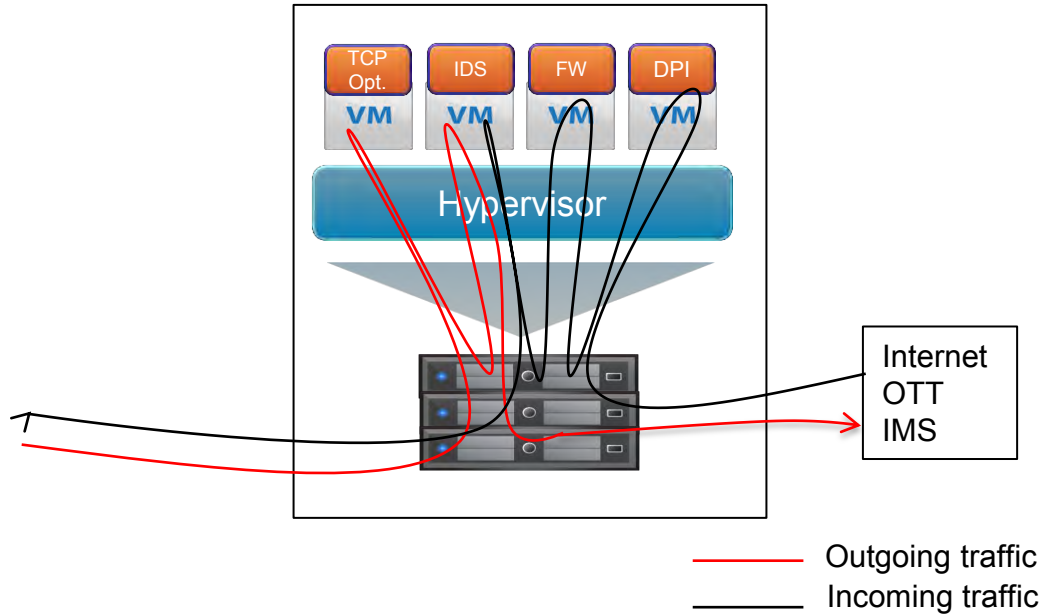
SFC in the wide

- **Build an overlay network between DCs**
- **Create Classification policies to steer application traffic throughout the Service Functions (VNF)**
- **Integrate a Network Service Header (NSH) and SDN to dynamically steer specific traffic to the appropriate VNFs**
 - Service Path Identifier: to specify an overlay path
 - Service Index: represents the index

SFC in the wide

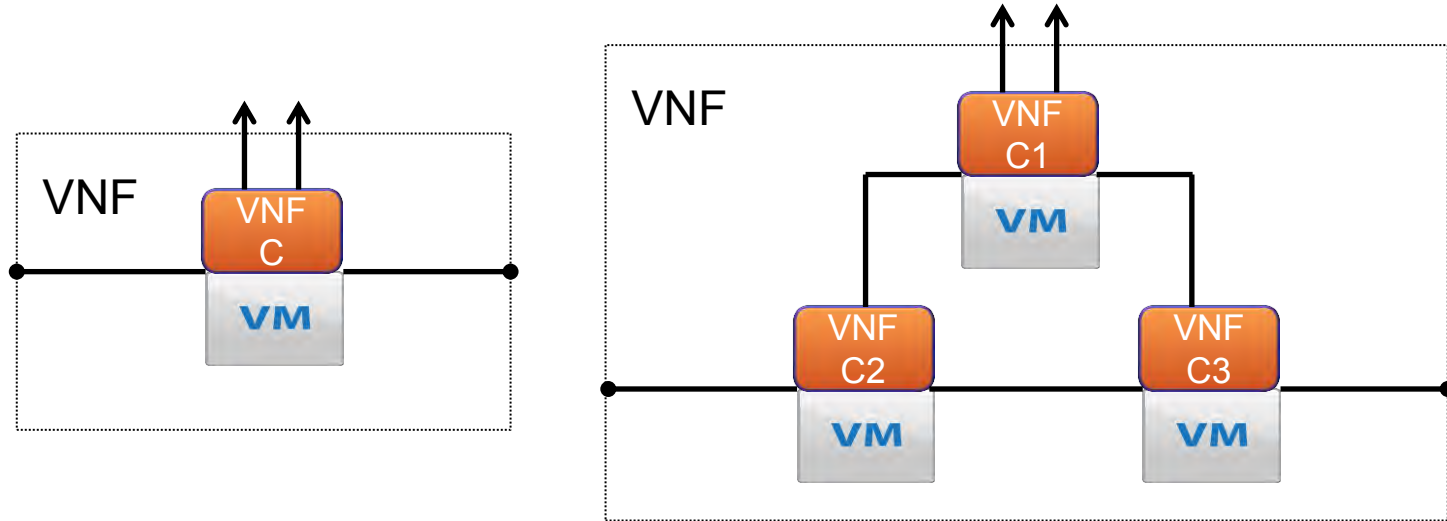


Network Forwarding graph



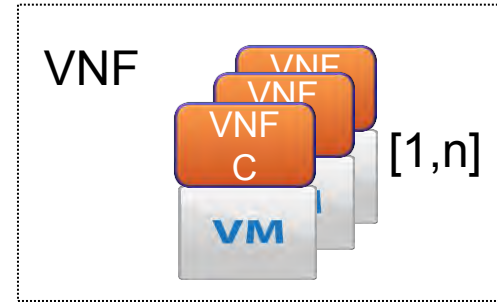
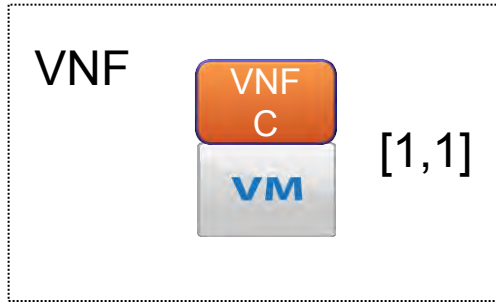
Ex. Service Chaining SGI-LAN 3GPP

Software implementation: VNF Decomposition



VNF structure: single vs. multiple components

Software implementation: reliability



VNF instantiation: non-parallel vs. parallel VNFCs

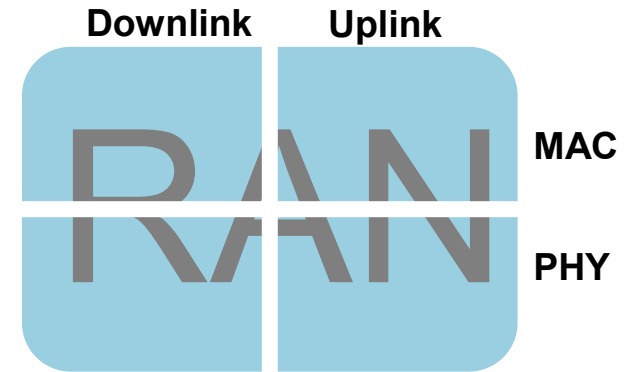
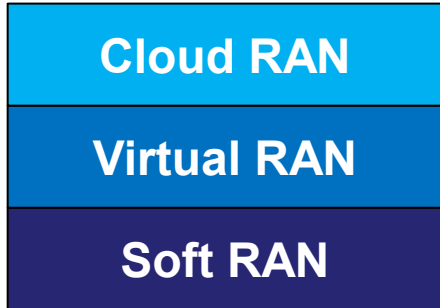
Open Source initiative: MANO

- **Open Source MANO (OSM): An open-source project hosted by ETSI**
 - Supported by many companies
 - Deliver an open source MANO stack following ETSI NFV model
 - Legacy: OpenMANO, Juju Charms VNFM
- **OPNFV initiative**
 - Open Source tool for the NFV eco-system: MANO, VNFM, etc.
- **Open-O**
 - Opens source hosted by linux foundation
 - Deliver end-to-end services across NFV and SDN: NFVO, NFVM and SDN
 - Just merged with eCOMP (at&t)
- **OpenBaton**
 - NFV eco-system: NFVO

Part III – Challenges



C-RAN Challenges



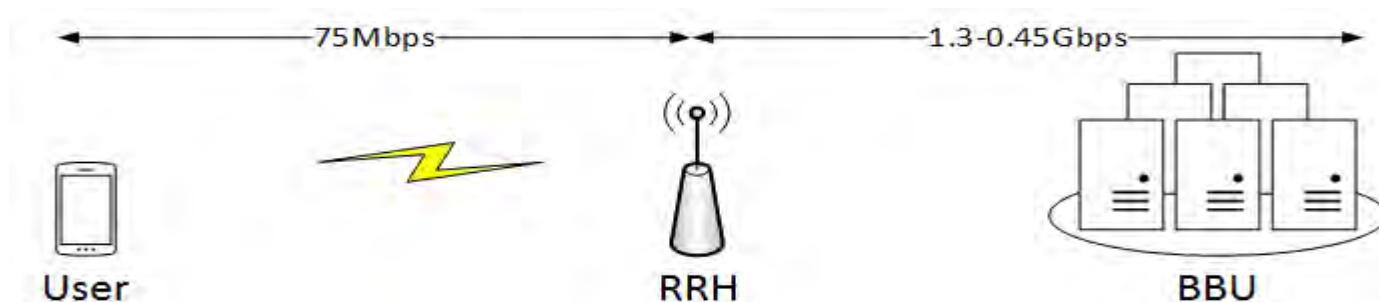
- Capacity, latency, and jitter requirements for fronthaul
- Flexible Functions Split
- BBU processing budget and protocol deadlines
- Realtime, virtualization environment and BBU performance

Capacity, latency, and jitter requirements for fronthaul

- **Transport Network between RRH and BBU**
 - Dark fiber
 - WDM/OTN: Wavelength-division multiplexing (WDM)/Optical Transport Network (OTN)
 - Unified Fixed and Mobile access (microwave)
 - Carrier Ethernet
- **Protocols**
 - Common Public Radio Interface (CPRI)
 - Open Base Station Architecture Initiative (OBSAI)
 - Open Radio equipment Interface (ETSI-ORI)
 - Radio over Ethernet (RoE)
- **Key requirements**
 - Supported Topology (star, ring, mesh), reliability, distance, multiplexing, capacity, scalability

Capacity, latency, and jitter requirements for fronthaul

- **Latency required by the HARQ RRT deadline**
 - 250 us maximum one-way latency adopted by NGMN, limiting the length of BBU-RRH within 20-40 Km
 - ☞ speed of light in fiber is approximately 200×10^6 m/s
- **Jitter required by advanced CoMP schemes**
 - <65 ns(MIMO, 36.104) timing accuracy in collaboration between base stations, which is the tightest constraint.
- **Frequency error < 50 ppb (macro BS)**
- **BER < 10e-12**
- **20MHz channel BW, SISO, 75 Mbps for users**
 - 2.6Gbps on Fronthaul without compression (0,87Gbps with 1/3)



Capacity, latency, and jitter requirements for fronthaul

$$c = 2 \cdot N_{Antenna} \cdot M_{Sector} \cdot F_{Sampling} \cdot W_{I/Q} \cdot C_{carriers} \cdot O_{coding+proto} \cdot K_{comp}$$

Bandwidth	$N_{antenna}$	M_{sector}	$F_{sampling}$	$W_{I/Q}$	$O_{coding+proto}$	$C_{Carriers}$	K	Data Rate
1.4MHz	1x1	1	1.92	32	1.33	1	1	163Mb/s
5MHz	1x1	1	7.68	32	1.33	1	1	650Mb/s
5MHz	2x2	1	7.68	32	1.33	1	1	1.3Mb/s
10MHz	4x4	1	15.36	32	1.33	1	1/2	2.6Gb/s
20MHz	1x1	1	30.72	32	1.33	1	1	2.6Gb/s
20MHz	4x4	1	30.72	32	1.33	1	1/3	3.4Gb/s
20MHz	4x4	1	30.72	32	1.33	1	1	10.4Gb/s

■ Costs

- Tens of BS over long distance → 100 Gbps

■ Savings

- Equipment's
- Energy

Capacity, latency, and jitter requirements for fronthaul

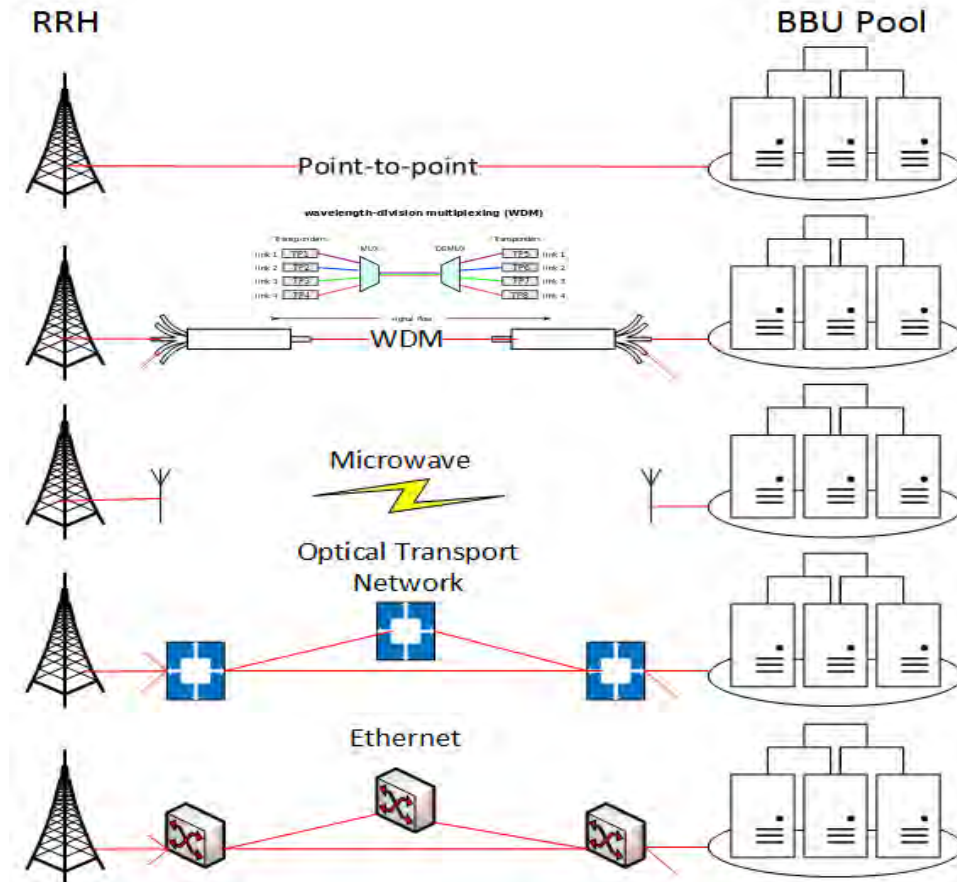
Medium	Bit rate	Distance	Remark
Fiber	100Gbps	~20Km	OTN: expensive
Copper	10Gpbs	100m	Low cost, SYNC
Wireless	1Gbps	2-15Km	LoS, high latency

■ Synchronization

- Frequency of transmission
- Frame
- Handover, coding

■ Solution

- GPS
- PHY layer clock, SyncEth
- Packet-based sync (IEEE 1588v2)



Capacity, latency, and jitter requirements for fronthaul

Asynchronous Ethernet

- Reduce the fronthaul capacity
- I/Q transport over Ethernet
- Some DSP in RRH to reduce transport speed/cost (split)
 - ☞ Decoupling of user-processing and cell-processing (iFFT/FFT)

Advantages

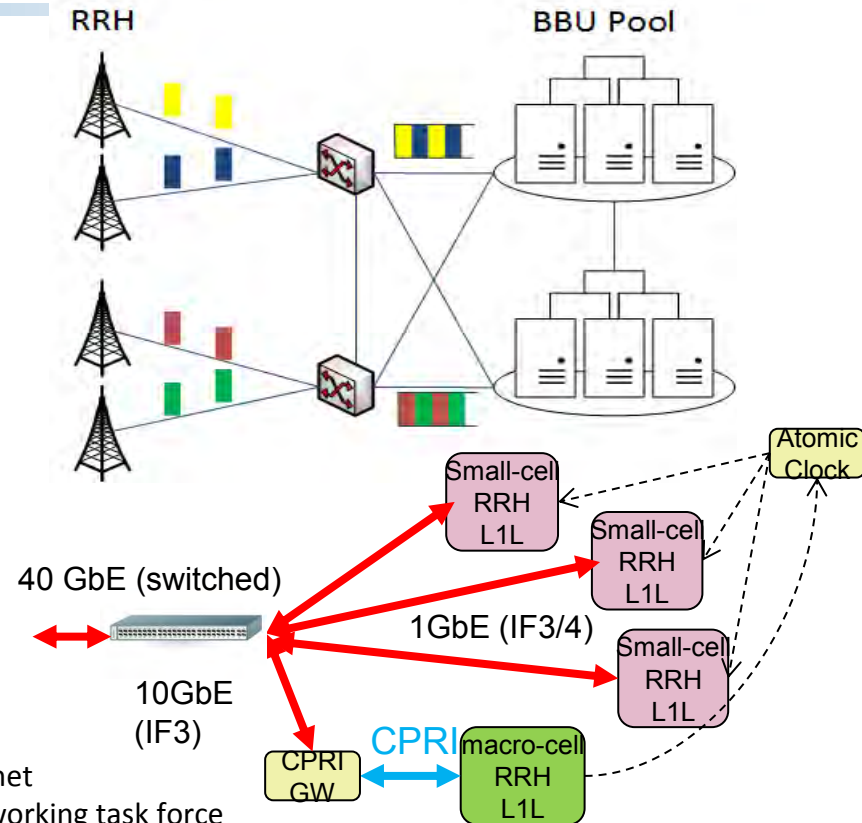
- Cost saving (reuse, commodity hardware)
- Switching (packet-based)
- Multiplexing / load balancing
- Flexible topology (mesh)

Challenges

- Distributed computation
- Cheap synchronization ((GPS, 1588v2)
- Loosening of control/user-plane coupling
- Real-time I/Q over Eth links (copper, low-cost fiber)

Hot topics

- IEEE 1904.3 - encapsulation and mapping of IQ data over Ethernet
- IEEE 802.1 – CPRI fronthaul discussion with Time Sensitive Networking task force
- CPRI → CPRI2?
- 3GPP - proposal on a study item on variable rate multi-point to multi-point packet-based fronthaul interface supporting load balancing



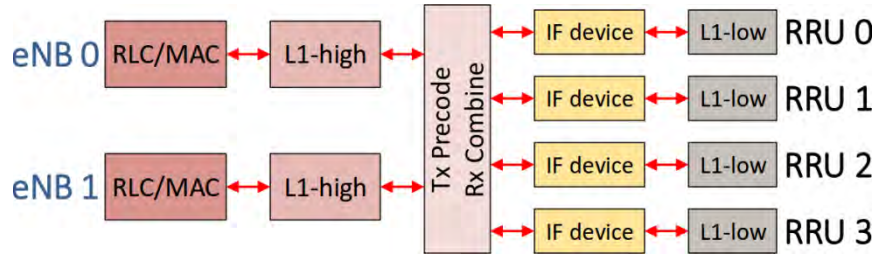
Reduce FH Capacity, Latency, Jitter?

- **Functional split**
 - DU and RRU
 - CU and DU
- **Compression**
- **Clustering : association of DU and RRU (1:N)**
- **Routing, switching**
- **Packetization**
- **Packet scheduling**

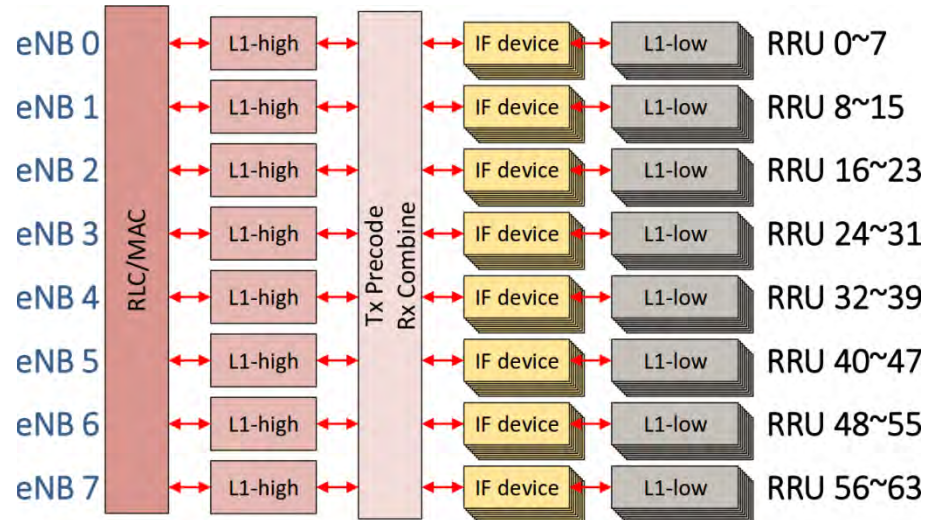
Flexible functional Split

Use-cases

DAS deployment example



Massive MIMO Deployment example



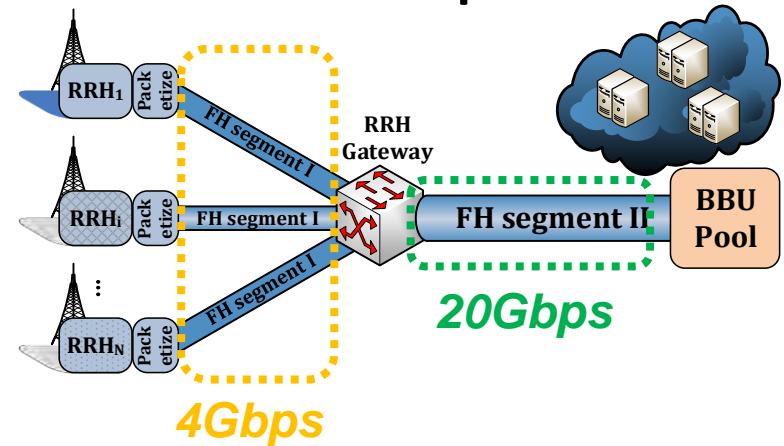
Flexible Functional Split

- **Different levels of split exist**
 - **PHY split** : trade-off FH capacity and coordinated signal processing
 - **Protocol split**: flexibility in deployment, and support of Multi RAT
 - **Control and data plane**: programmability, and support SDN
- **Split Type :**
 - **Split-specific deployment**: Only deploys the necessary functions at RRU to save the expenditures but with the loss in terms of fewer flexibility of RRU-DU splits.
 - **Flexible-split deployment**: Deploys all baseband functionalities at RRU (i.e., similar to legacy BS) to change the RRU-DU split.

Flexible Functional Split

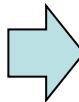
- Derive maximum supported RRHs based on achievable peak-rate

Scenario	1	2	3
Bandwidth	20 MHz		
Oversampling Ratio	1		
Rx Antennas	4		
Cyclic prefix length	Normal		
MIMO	4 Layer		
PUCCH RB	4		
SRS BW Config	7		
SRS SF Config	9		
Control Overhead	4.3%		
RA Config	0		
RA Overhead	0.3%		
Modulation	64 QAM	16 QAM	QPSK
TBS index	26	16	9
Time sample bitwidth	16		
Frequency sample bitwidth	16		
LLR bitwidth	8		



Based on achievable peak-rate on all RRHs

Scenario	1	2	3
Split A	5		
Split B	8		
Split C	9		
Split D	7	11	22
Split E	66	161	313



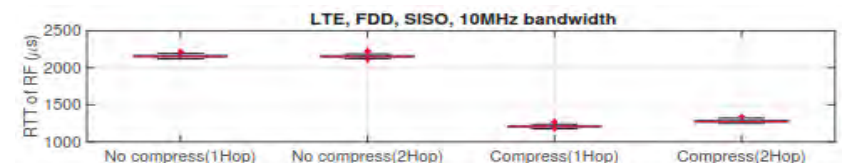
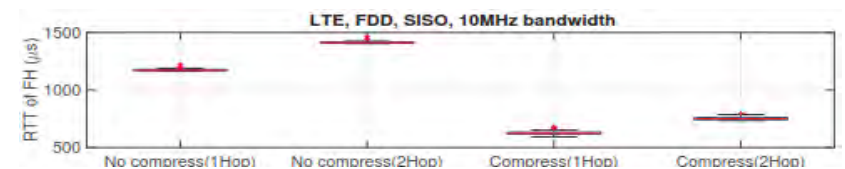
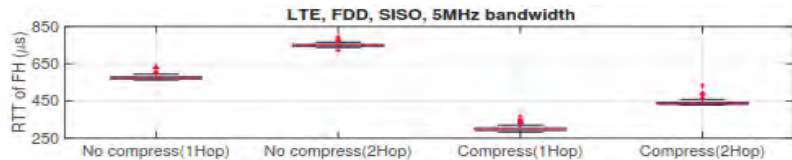
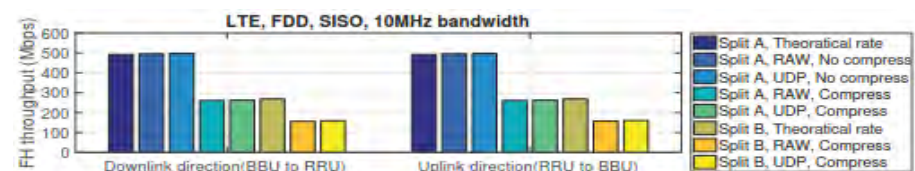
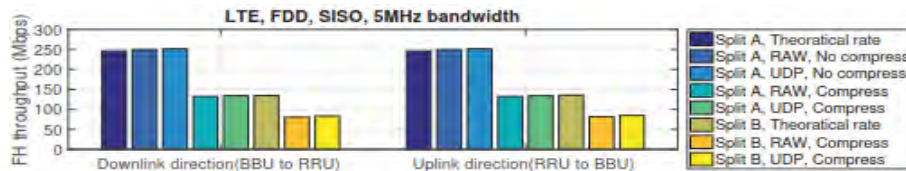
Peak data rate : Scenario 1
3.93 Gbps
2.15 Gbps
2.14 Gbps
2.63 Gbps
300.8 Mbps

Flexible functional split

Compression

■ FH-related KPIs

- FH link throughput 50% reduction : (a) A-law compression, (b) Split A → B
- RTT of FH and RF front-end
 - ☞ RTT of FH reduction due to A-law compression
 - ☞ RTT of RF front-end is doubled due to excess RTT of FH w/o compression

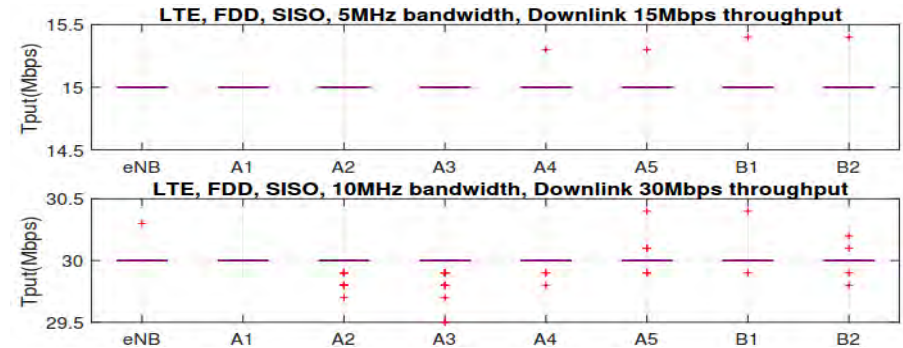
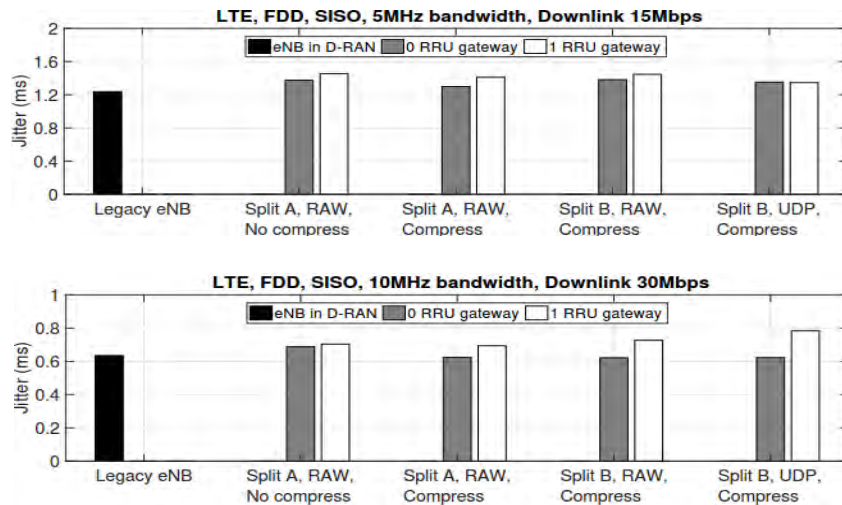


Flexible Functional Split

KPI

Data-plane KPIs

- Apply 15Mbps/30Mbps downlink user traffic rate in 5MHz/10MHz BW
- Larger data-plane jitter and larger average RTT due to extra route
 - ☞ Less Tx/Rx processing time in order to provide extra FH transportation time
- Different C-RAN deployments show almost same good-put as D-RAN

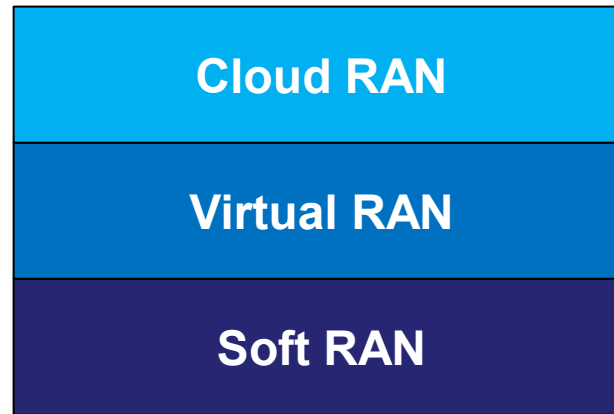


Mode	Split	Protocol	Compression	Hop count
A1	A	RAW	No	1
A2	A	RAW	No	2
A3	A	UDP	No	2
A4	A	RAW	Yes	2
A5	A	UDP	Yes	2
B1	B	RAW	Yes	2
B2	B	UDP	Yes	2

BBU processing budget

Soft RAN

- 4G Feasible on General Purpose Processors (x86)
- An eNB is approximately **1-2 x86 cores on Gen 3 Xeon silicon**
 - Perhaps more power efficient solutions from TI, Freescale or Qualcomm
 - But: lose commodity software environment and common HW platform to high-layer protocols and cloud



eNB Rx stats (1 subframe)

- OFDM demod : 109.695927 us
- ULSCH demod: 198.603526 us
- ULSCH Decoding : 624.602407 us

→ 931 us (<1 core)

eNB Tx stats (1 subframe)

- OFDM mod : 108.308182 us
- DL SCH mod : 176.487999 us
- DL SCH scrambling : 123.744984 us
- DL SCH encoding : 323.395231 us

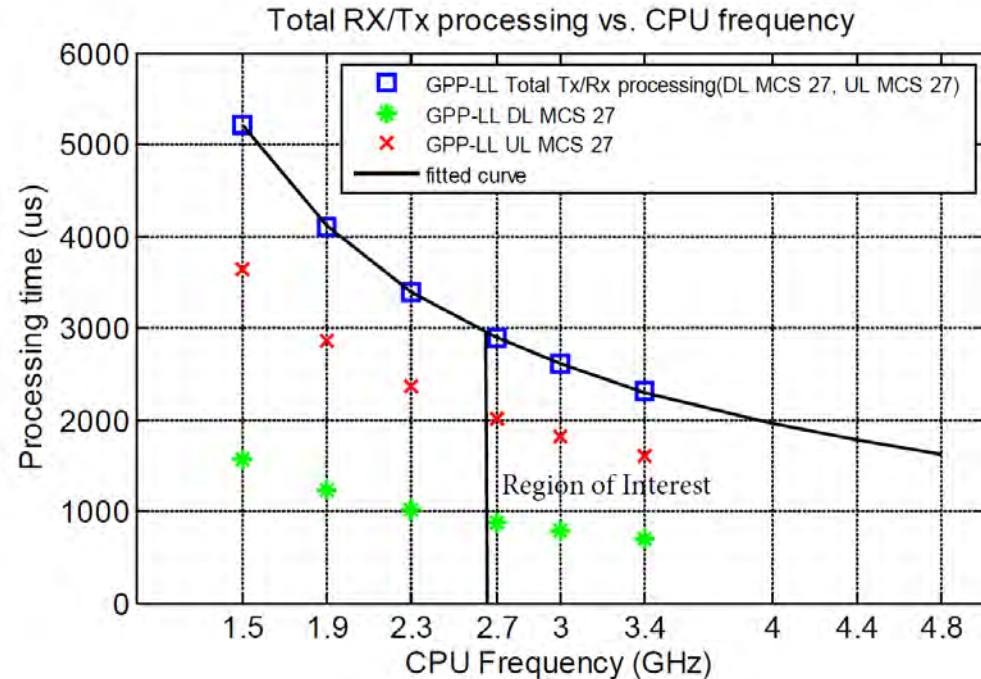
→ 730 us (< 1core)

- Efficient base band unit is challenging
- With AVX2 (256-bit SIMD), turbo decoding and FFT processing will be exactly twice as fast
 - <1 core per eNB
 - .4 core per eNB without TC ← can this be exploited efficiently with HW acceleration?
(Solution adopted in China Mobile CRAN project, offload of TC on Altera FPGA)
- Configuration
 - gcc 4.7.3, x86-64 (3 GHz Xeon E5-2690),
 - 20 MHz bandwidth (UL mcs16 – 16QAM, DL mcs 27 – 64QAM, transmission mode 1 - SISO)
 - 1000 frames, AWGN channel

Processing Budget for Peak Rate

Soft RAN

- FDD LTE HARQ requires a round trip time (RTT) of 8ms
 - $T_x + R_x \leq T_{harq}/2 - (acquisition + transport + offset) \approx 3ms$
 - ~2ms RX and 1ms TX (can't be fully parallelized)
- Processing time reduces with the increase of CPU Freq.
- min CPU Freq is 2.7GHz
 - HARQ deadline
- $T_{subframe} = \alpha / x,$
 - $\alpha = 8000$
 - x is the CPU freq GHZ



Processing Budget for Peak Rate

Soft RAN Considerations

- **Key Consideration to meet the deadlines (SF, protocol)**
 - **Real-time/low-latency OS** (linux with deadline scheduler) and optimized BIOS
 - ☞ Problem: **OS scheduler latency** (kernel is not pre-emptible)
 - Real-time data acquisition to PC
 - SIMD optimized integer DSP (SSE4, AVX2)
 - Parallelism (SMP)
 - x86-64
 - ☞ more efficient for Turbo decoding because of the number of available registers is doubled

- **Remove bottlenecks with**
 - hardware accelerators or hybrid CPUs
 - ☞ Turbo decoders (easily offloaded to FPGA-based accelerators), FFT, PDCP (de)encryption
 - GPUs or Xeon PHY-type devices
 - ☞ Perhaps interesting for Turbo-decoders and encoders
 - ☞ Applicable to FFTs, but may not be worth it
 - Main issue in both FPGA/GPU offloading
 - ☞ High-speed low-latency bus between CPU memory and external processing units

Realtime, virtualization environment and BBU performance

RTOS issues

- **Low-latency radio applications for PHY (e.g. 802.11x,LTE) should run under an RTOS**
 - Meet strict hard deadline to maintain the frame/subframe and protocol timing
 - efficient/elastic computational resources (e.g. CPU, memory, network)
- **Example OS**
 - eCos/MutexH for generic GNU environment
 - RTAI for x86
 - VXWorks (\$\$\$)
- **Example: RTAI / RT-PREEMPT kernel can achieve worst-case latencies below 30 μ s on a loaded-PC. More than good enough for LTE, but not 802.11x because of MAC timing.**
- **Should make use of POSIX multithreading for SMP**
 - Rich open-source tool chains for such environments (Linux, BSD, etc.)
 - Simple to simulate on GNU-based systems for validation in user-space
 - Allow each radio instance to use multiple threads on common HW

Processing Budget for Peak Rate

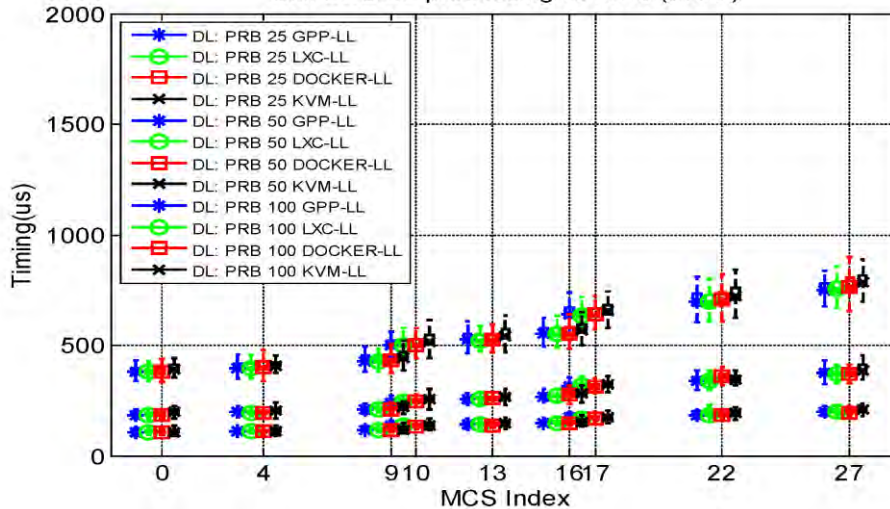
Virtual-RAN



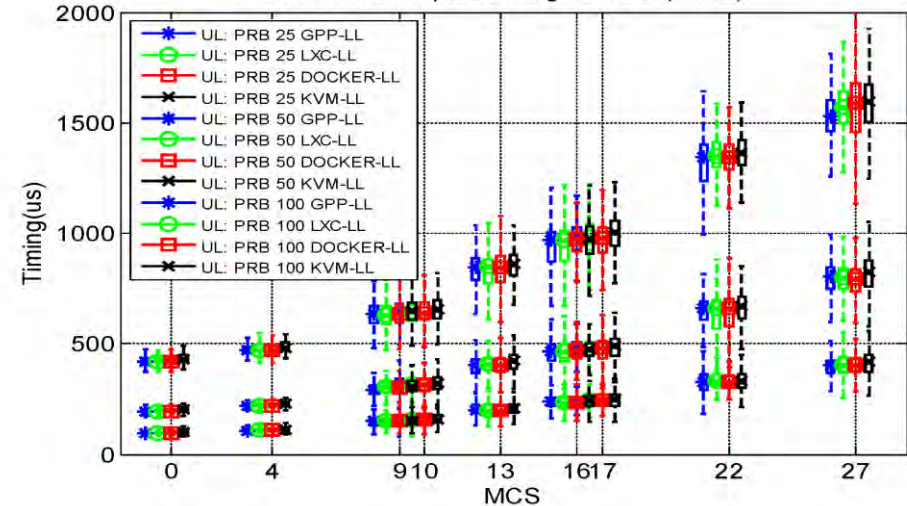
- DL and UL BBU processing load for various MCS, PRB, and virtualization flavor

➤ Comparable BBU Processing time

OAI BBU DL processing vs MCS (SISO)



OAI BBU UL processing vs MCS (SISO)



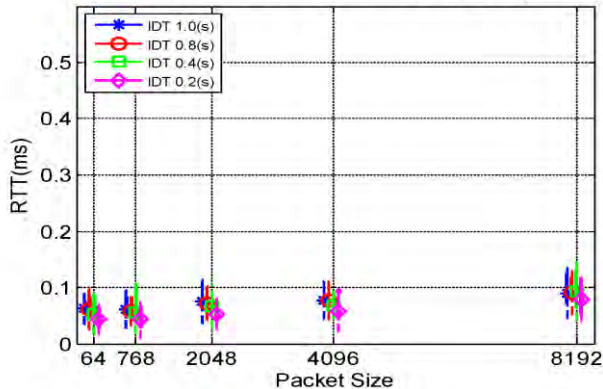
Additional Consideration

Virtual-RAN

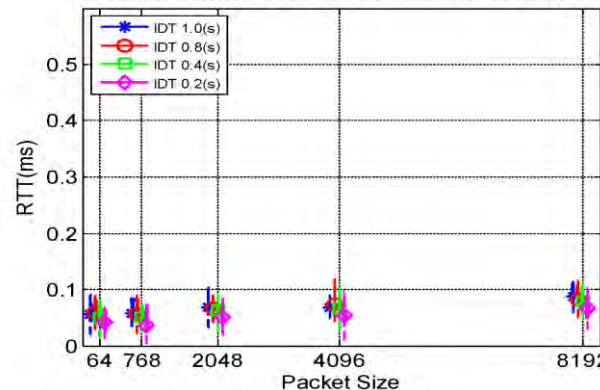


- **I/O access delay**
 - RF, ETH, and HW accelerator
 - RF Passthrough vs Hardware virtualization (and sharing)
 - Delay and jitter requirement on the fronthaul network
- **Limitation of the guest-only network data rate**

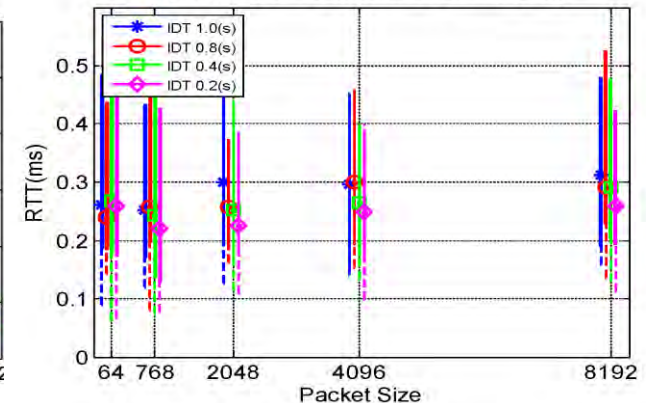
Guest LXC to Host communication delay



Guest DOCKER to Host communication delay



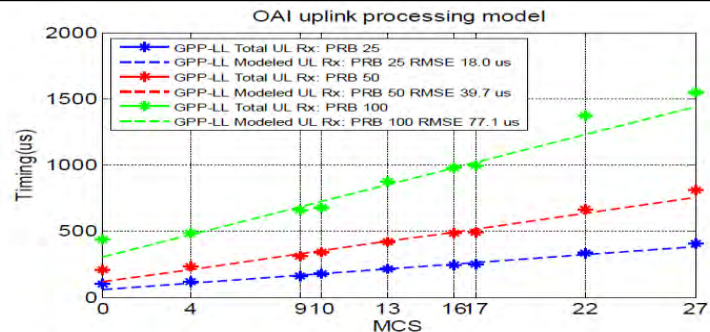
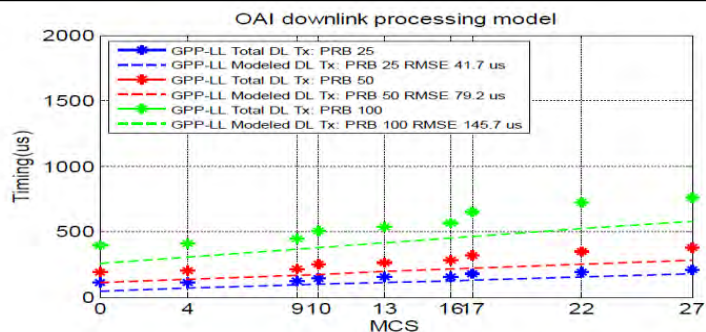
Guest KVM to Host communication delay



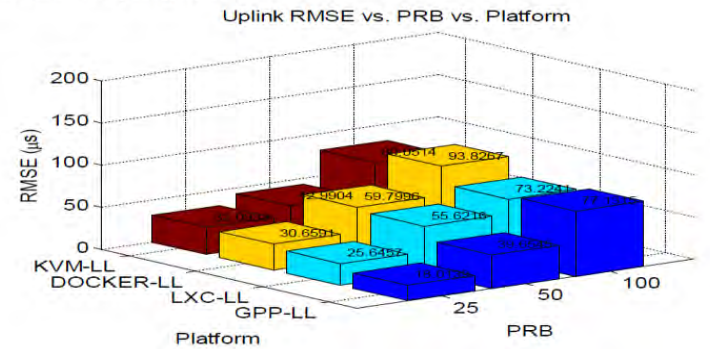
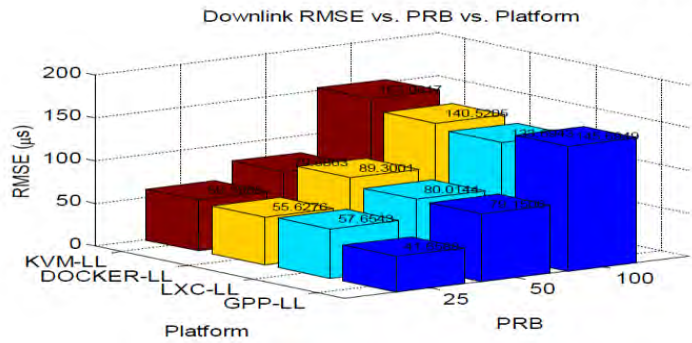
Additional Consideration

Soft and Virtual RAN Modelling

- $T_{subframe} = c[x] + p[w] + u_s(x, y) + u_r(x), \text{ where}$
- $u_s(x, y) = a(x) \cdot y + b(x), \text{ and } x \in PRB, y \in MCS, w \in VE$



(a) Fitted curves for GPP-LL platform



Additional Consideration

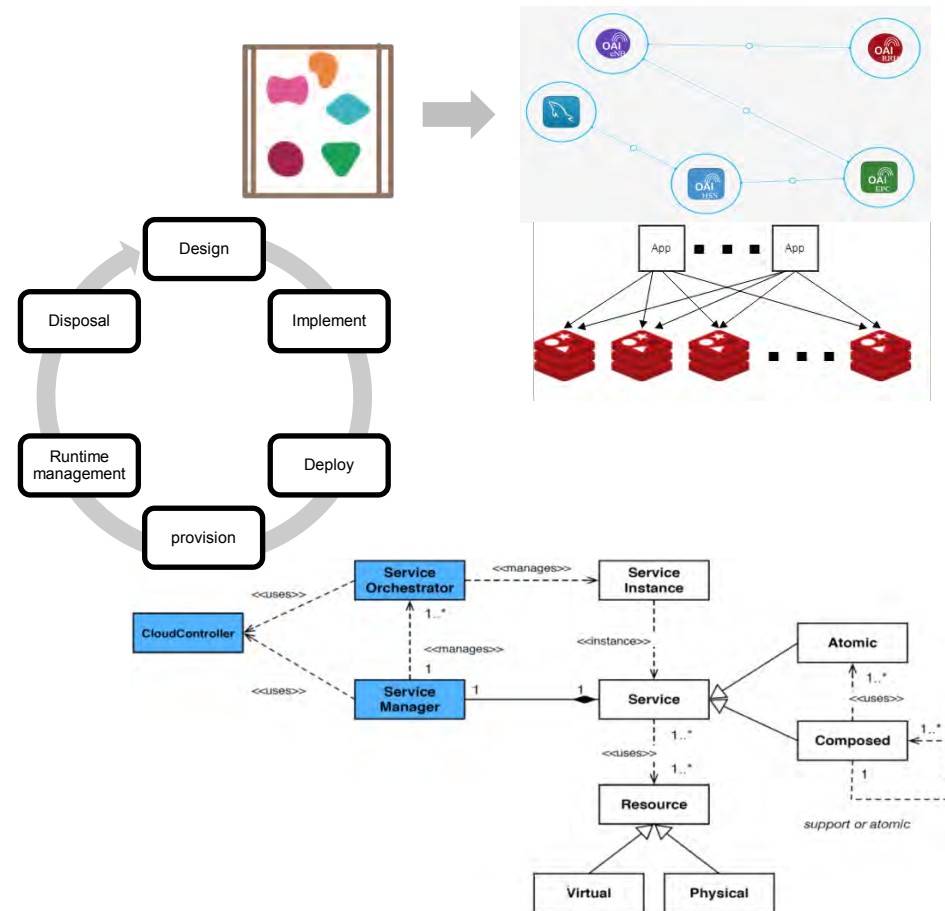
Cloud-Native RAN

Cloud RAN

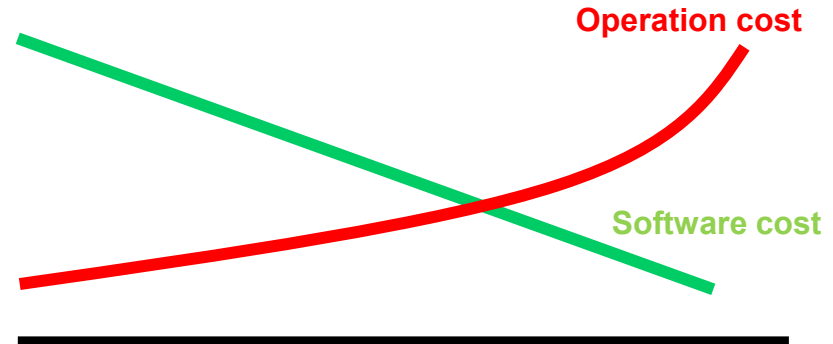
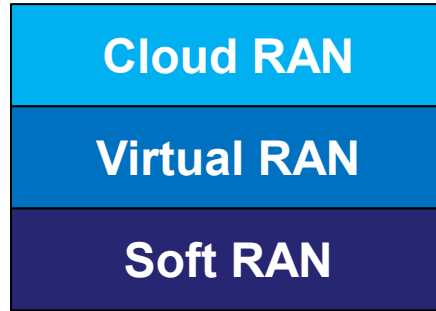
Virtual RAN

Soft RAN

- **Microservice Architecture along with NFV**
 - Flexible Functional split
 - Move from monolithic to a composed and metered service
 - Stateless, composable, reusable
- **Discovery and Automation**
 - E2E Lifecycle management
- **Scalability**
 - Scale in and out, pay-as-you-go
- **Reliability**
 - Redundancy and stateless
 - Availability and service assurance
- **Multitenancy**
 - Share the resources
 - (spectrum, radio, and infrastructure)
- **Placement**
 - Optimize the cost and performance
 - Supported Hardware, in particular for RAN
- **Realtime edge services**
 - Direct access to the radio information



NFV Challenges



- **E2E Service modelling and template definition**
- **NFV Service manager and orchestrator**
 - Multi-domain
 - Automated and/or cognitive logic
 - Network service manager, fault manager, autoscaling
 - Negotiation, policy, and charging
 - Network function and network application store
- **From NF virtualization to cloudification**
- **Placement of time-critical VNF chain**

E2E Service modelling and template definition

■ Requirements for modeling

- Design an abstract network slice for a particular use-case
- Identify the data models and interfaces across the network functions
- Standardize reference network slice templates
 - ☞ Define network service chain as a reference model

■ Service layer encapsulates

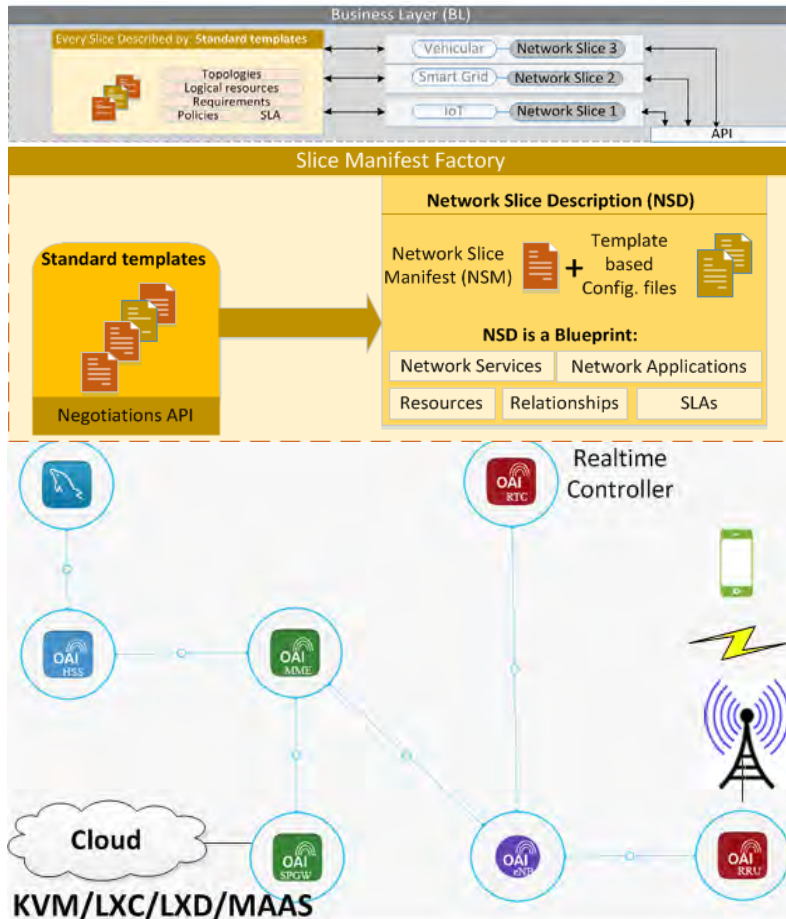
- VNF image and descriptor
- Configuration
- Connection points
- Two distinct lifecycles
 - Service
 - Relationships
- Health and monitoring parameters
- Resources and constraints
- Upgrade

■ Service template defines

- Service descriptor
- Input Parameters
- Configuration primitives
- Relationships, dependencies
- Resources and constraints
- Units (number of instances)
- Machine (physical or virtual)

E2E Service modelling and template definition

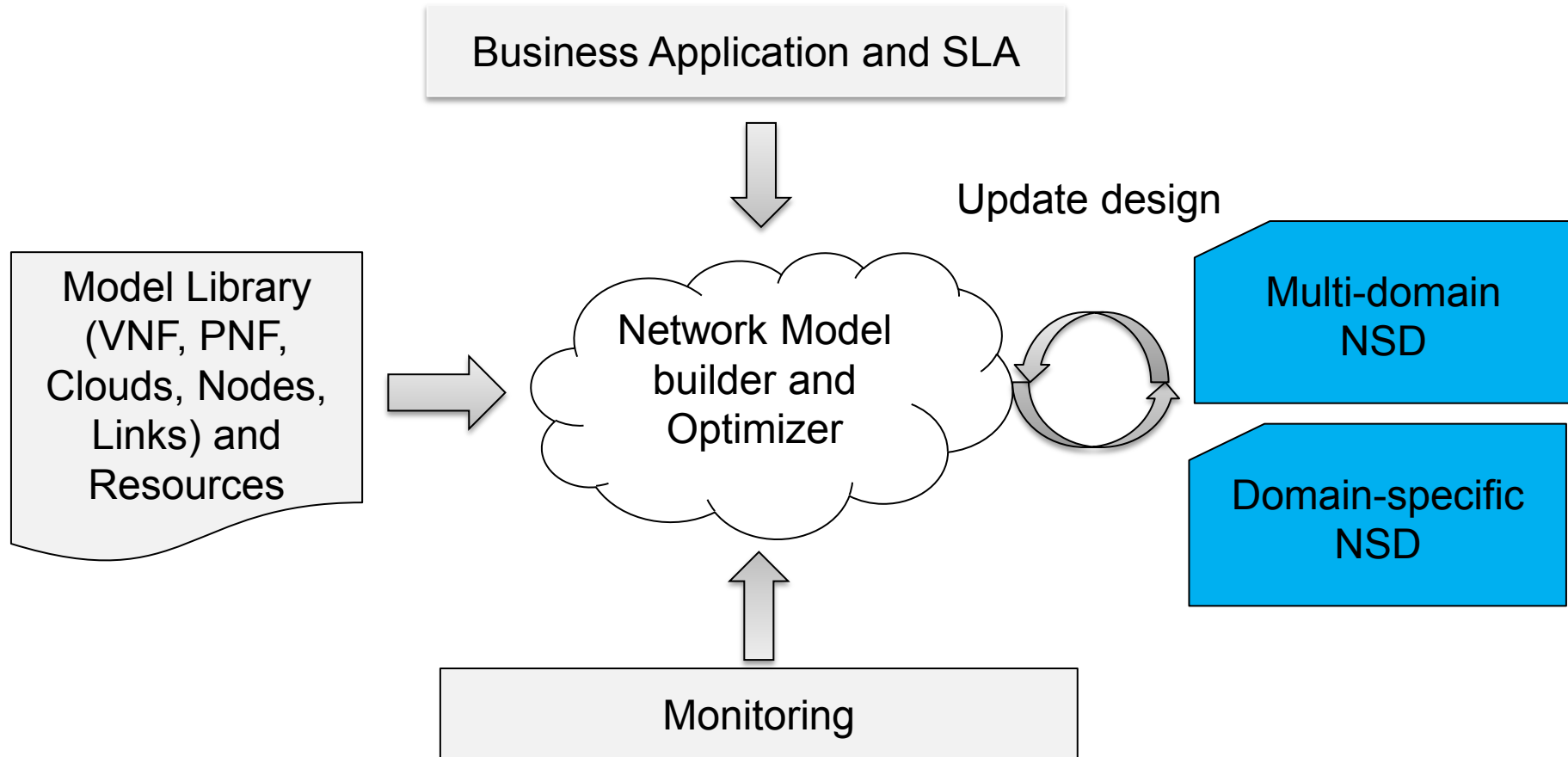
Blueprint of CRAN NFV Service Template



```

series: xenial
services:
  "oai-enb":
    charm: "cs:oai-enb"
    num_units: 1
    options:
      N_RB_DL: 50
      downlink_frequency: 2680000000L
      eutra_band: 7
      rrh_active: "yes"
      uplink_frequency_offset: "-120000000"
    to:
      - "0"
  "oai-epc":
    charm: "cs:oai-epc"
    num_units: 1
    annotations:
      "gui-x": "353"
      "gui-y": "267"
    to:
      - "kvm:oai-dnb/0"
relations:
  -- "oai-enb:epc"
  -- "oai-epc:epc"
  -- "oai-hss:db"
  -- "mysql:db"
  -- "oai-epc:hss"
  -- "oai-hss:hss"
machines:
  "0":
    series: trusty
    constraints: "arch=amd64 cpu-cores=4
    mem=15951 root-disk=8192"
  
```

E2E Service modelling and template definition

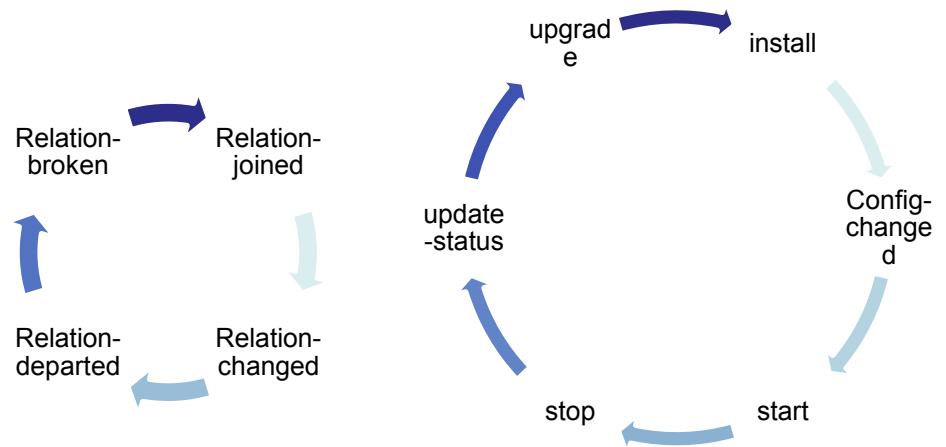
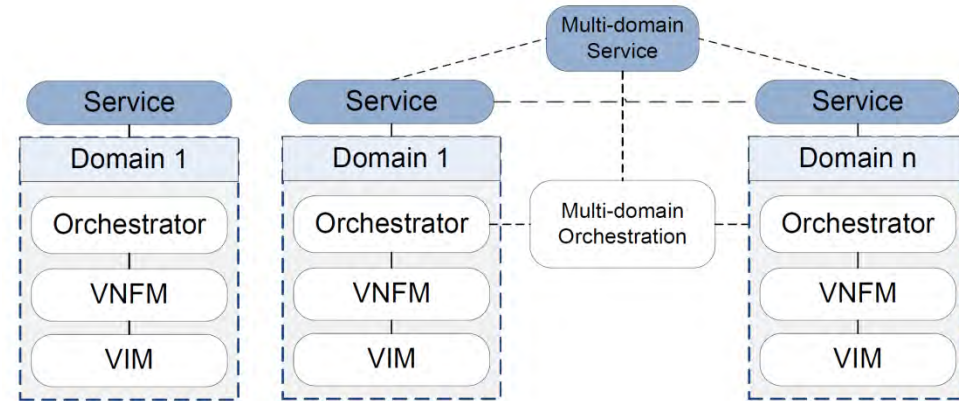


NFV Service manager and orchestrator

Multi-domain Orchestrator

■ Multidomain orchestrator

- Domain boundaries
 - ☞ Administrative, operators, service providers/vertical
- Design choices
 - ☞ centralized vs distributed
 - ☞ best-effort vs SLA driven
 - ☞ manual vs automatic vs cognitive
- Interfaces for inter and intra-orchestrator communication
 - ☞ Data models
 - ☞ Control plane interfaces



NFV Service manager and orchestrator

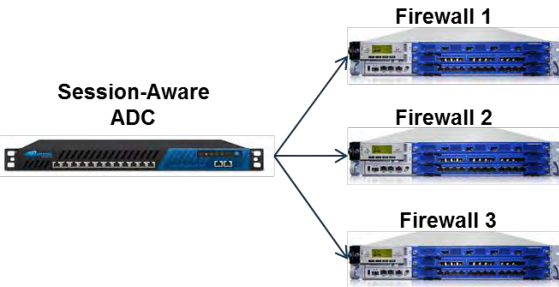
Multi-domain Orchestrator

- **Extend the scope of lifecycle management**
 - Distinguish common and specific actions
 - ☞ Service, configuration, scale, relationships, monitoring
 - ☞ Micro-services in a nested chain
 - Tight coupling with the VNF primitives and APIs
 - Time scale separation in managing and orchestrating services and interface with controller
- **Support for multi-service multi-tenant networks**
 - Nested service chaining
 - NFV and micro-service architecture
- **Resource elasticity beyond the infrastructure resources**
 - Radio frontend, Radio resources, and spectrum
- **Network stores, VNF IoT and benchmarking methodology**
- **Orchestration logic**
 - Resource allocation policy
 - E2E Service assurance, availability , reliability and scalability
- **Gap between static and cognitive management and orchestration**
 - Exploit machine learning and data mining techniques
 - ☞ E.g. monitoring

From NF virtualization to cloudification

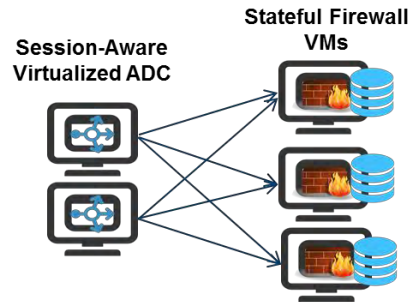
Pre-Virtualization

- Complex Appliances
- Hard to scale
- Hard to recover from box failure
- Over provisioning and waste of resources



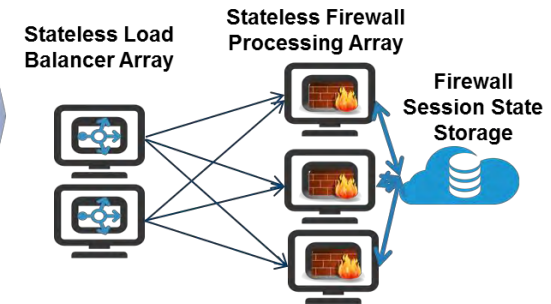
Post-Virtualization

- Complex stateful software
- Hard to scale
- Hard to recover from VM failure
- Automated provisioning possible
- If you virtualize complex system, you get virtualized complex system



Future – Cloud Native VNF

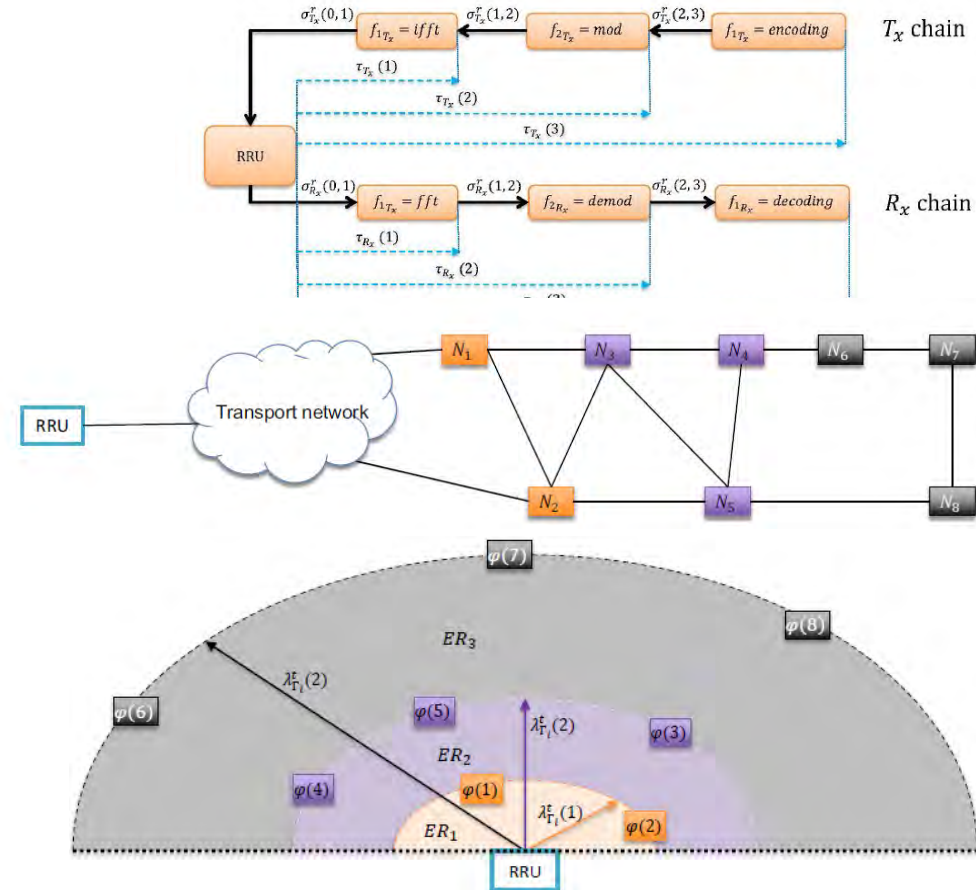
- Simple virtual appliances with stateless transaction processing coupled with state storage access
- Scale almost infinitely
- Fast recovery from VM failure
- On-demand provisioning and consolidation



<http://www.mellanox.com/blog/2015/03/from-network-function-virtualization-to-network-function-cloudification-secrets-to-vnf-elasticity/>

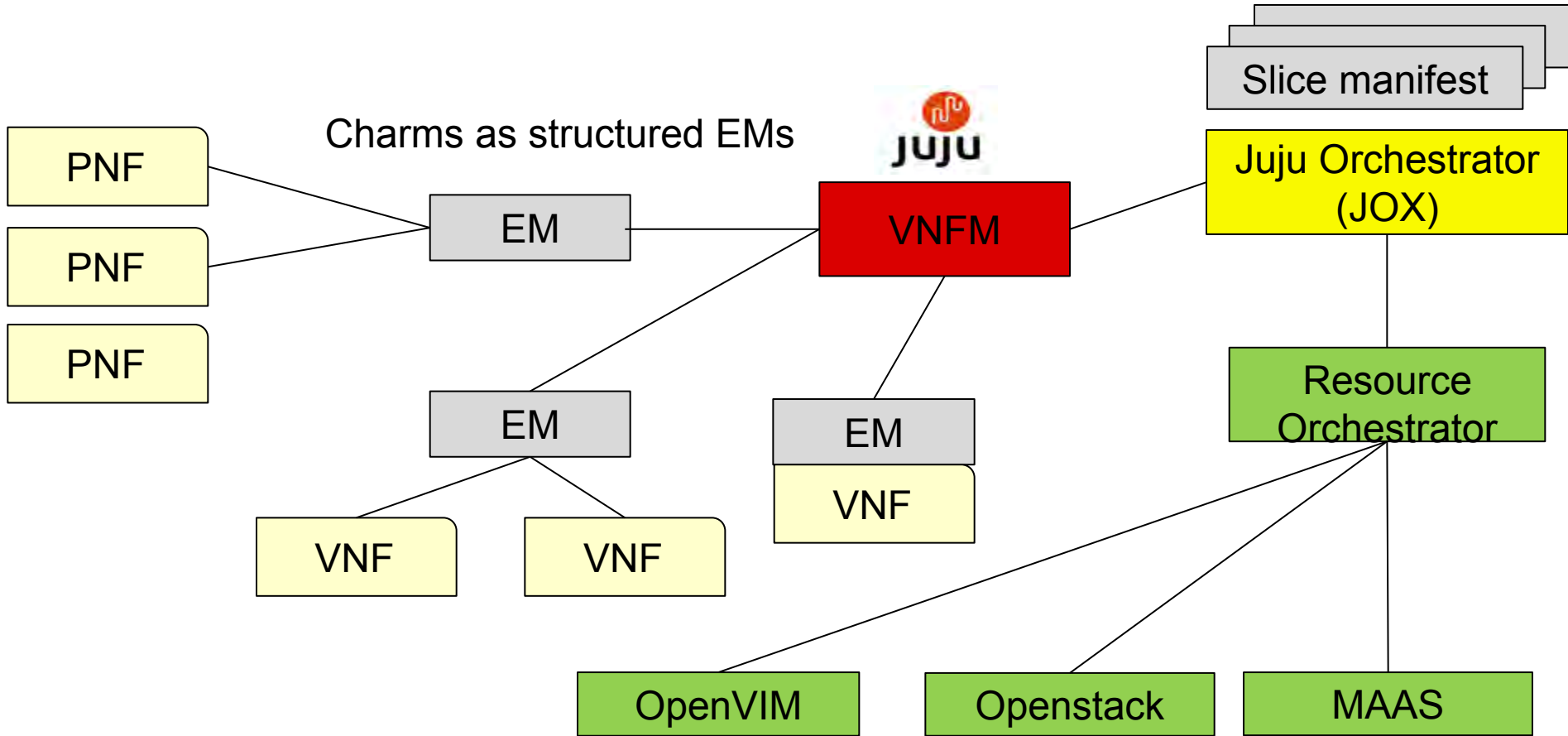
Placement of time-critical VNF chain

- **RAN functions must be efficiently placed in the cloud**
 - Respecting the timing of each function and the chain
 - Respecting the constraints (regions, resources)
- **Placement must**
 - Maximize the total number of chains
 - Minimize the embedding time
 - Load balancing
 - Dynamic



Deployment Example

NFV and OSM

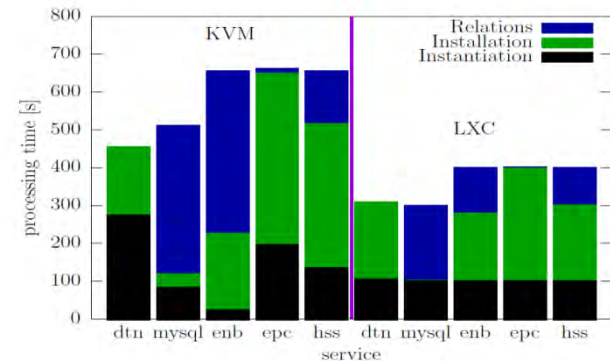
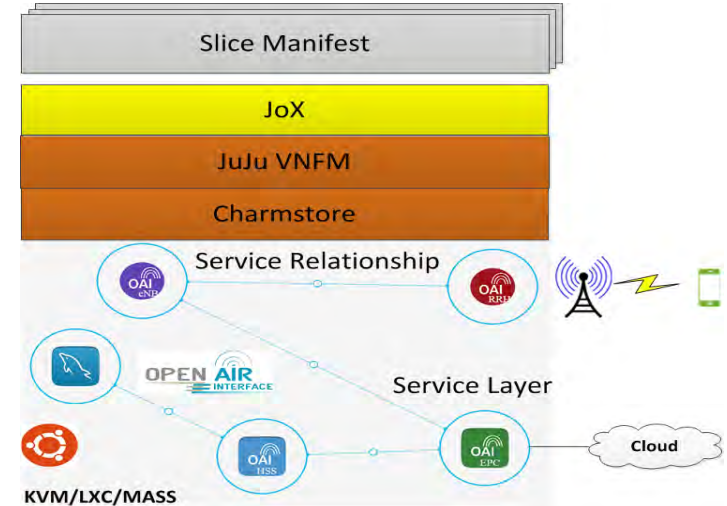


Deployment Example: LTEaaS

MOSAIC-5G JoX Platform



- Slice manifest defining the service as a whole**
 - Service is a composition of VNFs spanning across a set of domains and machines
 - E.g. two units of this app with their respective configuration file
- JoX orchestrates the E2E service lifecycle according to the slice manifest**
- Juju manages the services over the infrastructure**
- Charm acts a structured EM driven by juju**
 - Lifecycle
 - Install, update, and upgrade
 - configuration
 - Scale and elasticity
 - Integration
 - Relationship and interfaces, peers



Deployment Example: LTEaaS MOSAIC-5G JoX Platform



- **Event-driven Juju Orchestrator Core**

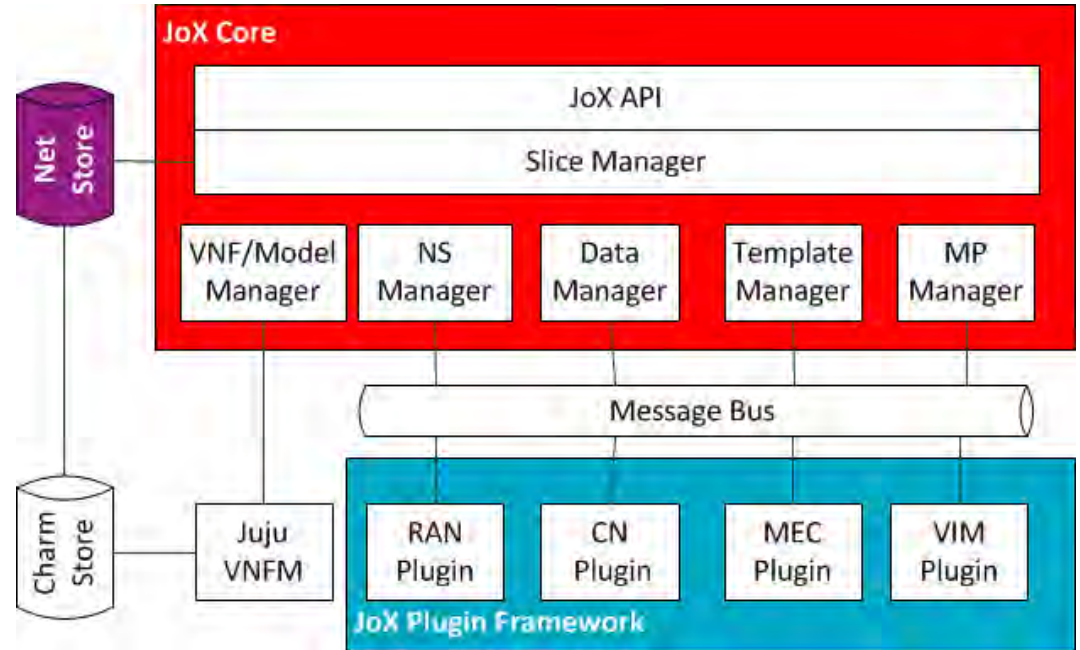
- Reusable
- Message bus
- ETSI Compatible

- **Plugin architecture**

- Juju passthru

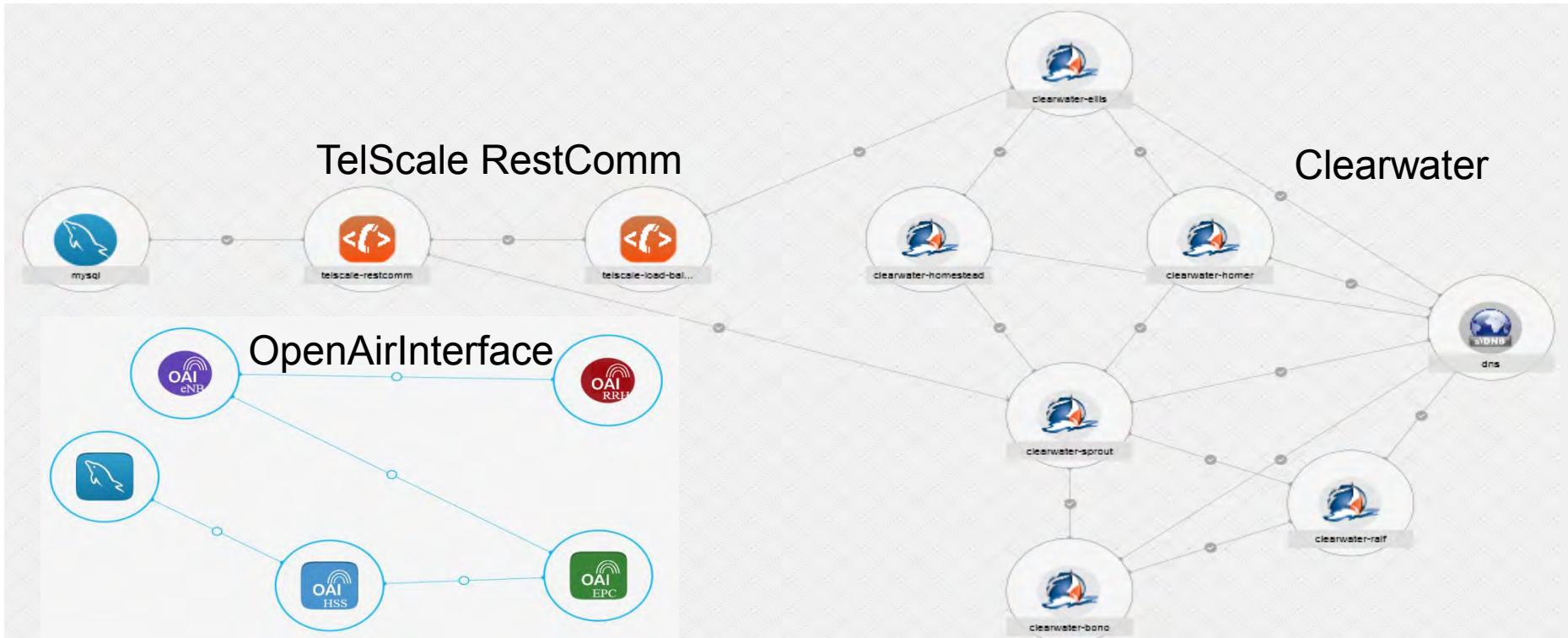
- **Support of network slicing**

- **Dedicated network store**



Deployment Example

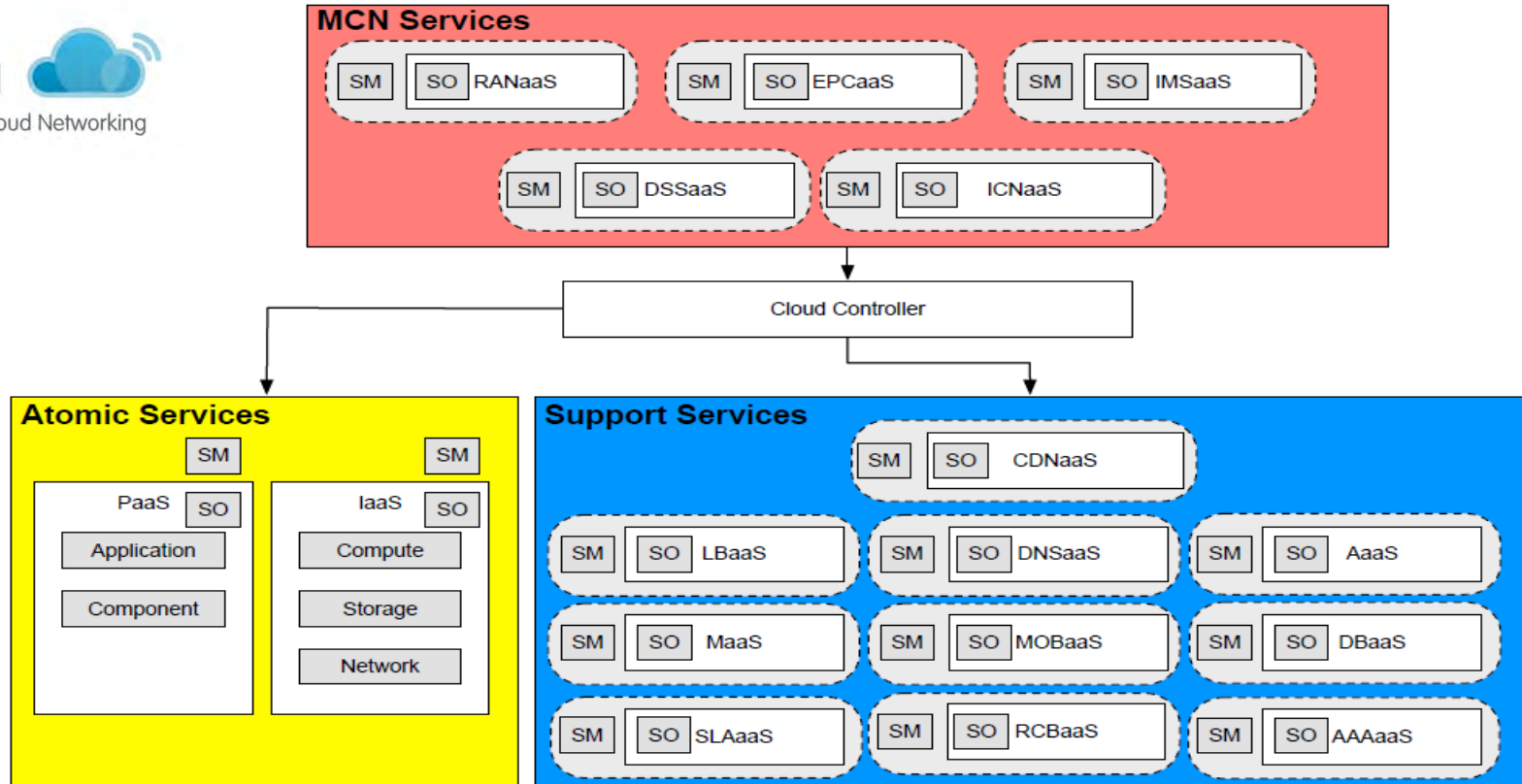
Juju and JoX



- **Rapidly build voice, video, WebRTC, USSD, SMS, fax and rich messaging applications over LTE**

Deployment Example: IMSaaS

OS, HEAT, OPS

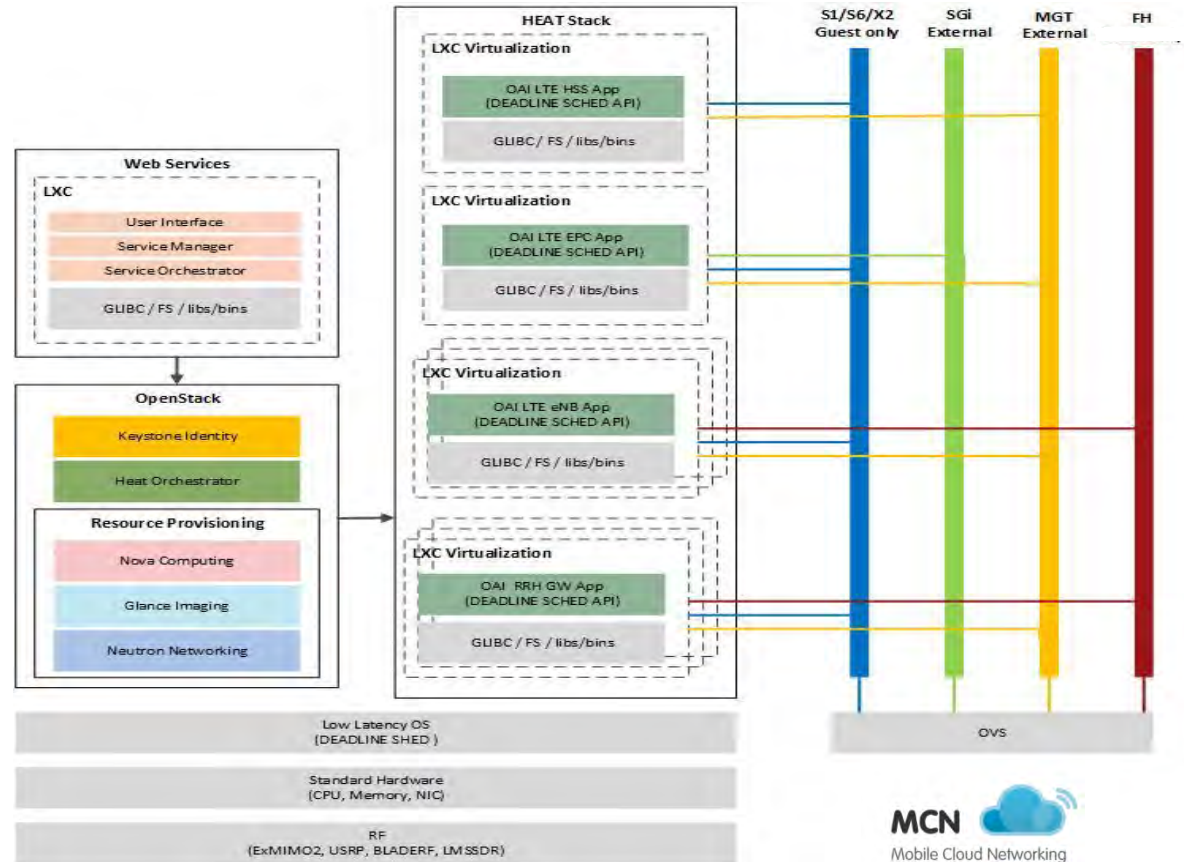


Deployment Example: IMSaaS

OS, HEAT, OPS



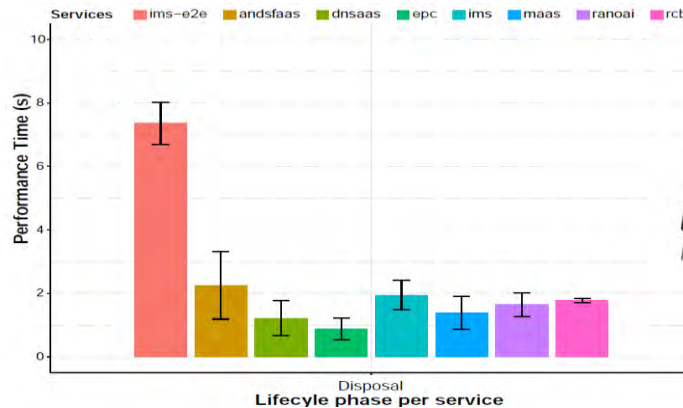
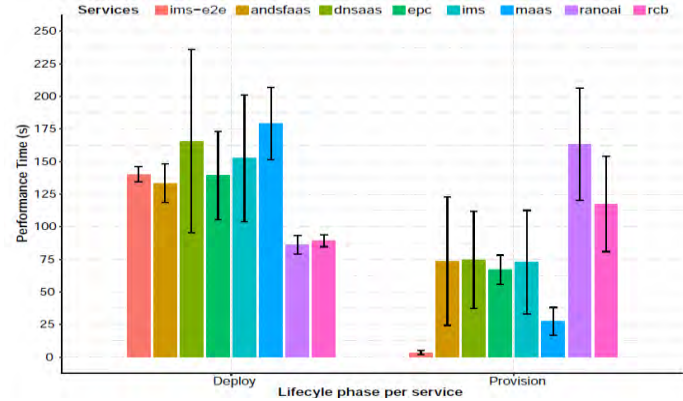
- **Three components**
 - web service
 - OpenStack
 - Heat stack
- **Heat Template describes the virtual network deployment**
 - Deployment Lifecycle
- **Linux Container**
- **Open vSwitch**
- **Low latency kernel**
- **RF frontend HW**



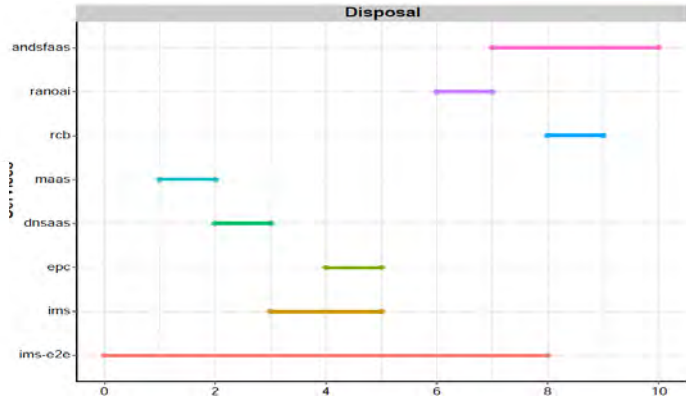
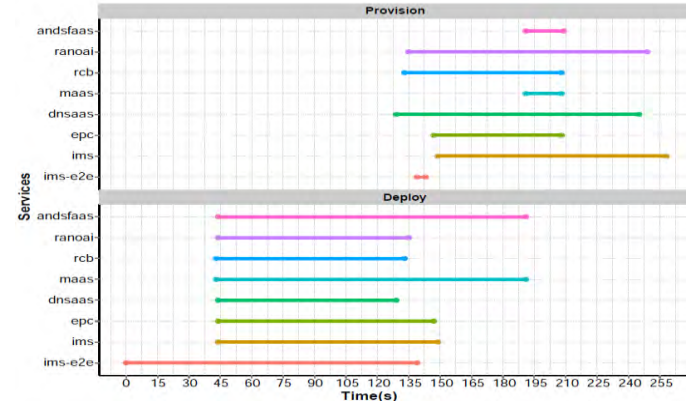
Deployment Example: IMSaaS OS, HEAT, OPS



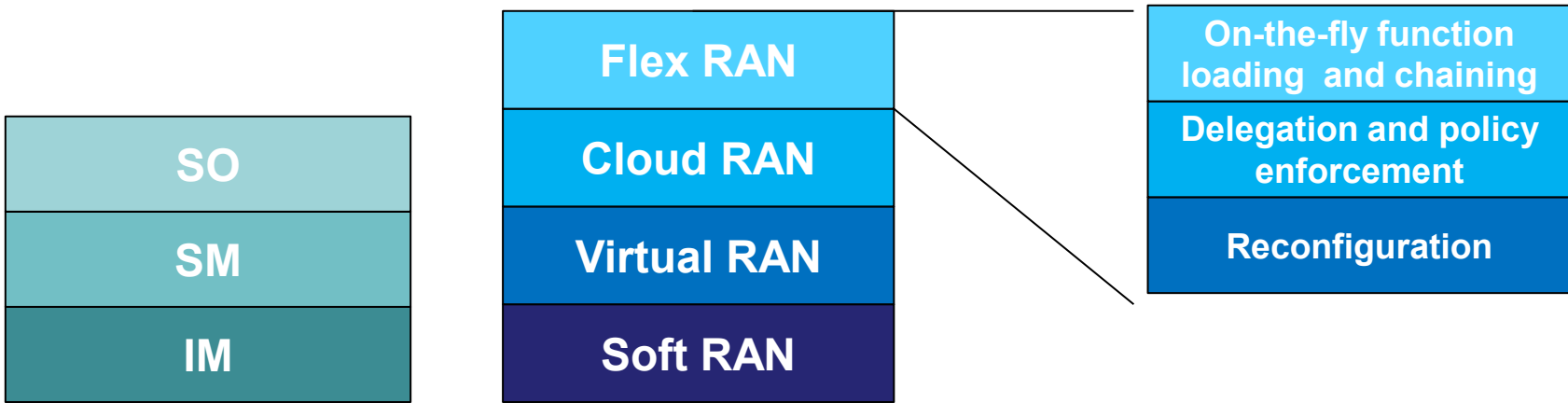
Lifecycle Time



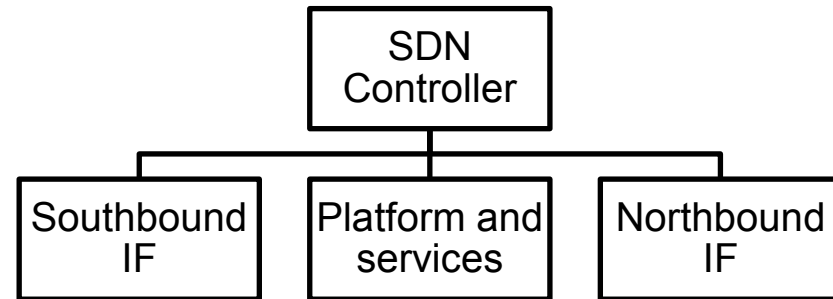
Lifecycle Sequence



SDN and MEC Challenges



- Radio and core API and Southbound Protocol
- Network Abstraction and graphs
- Scalability and Control delegation mechanisms
- Realtime control
- Low latency edge packet services
- Cognitive management, self-adaptive, and learning methods
- Northbound Application programming interface



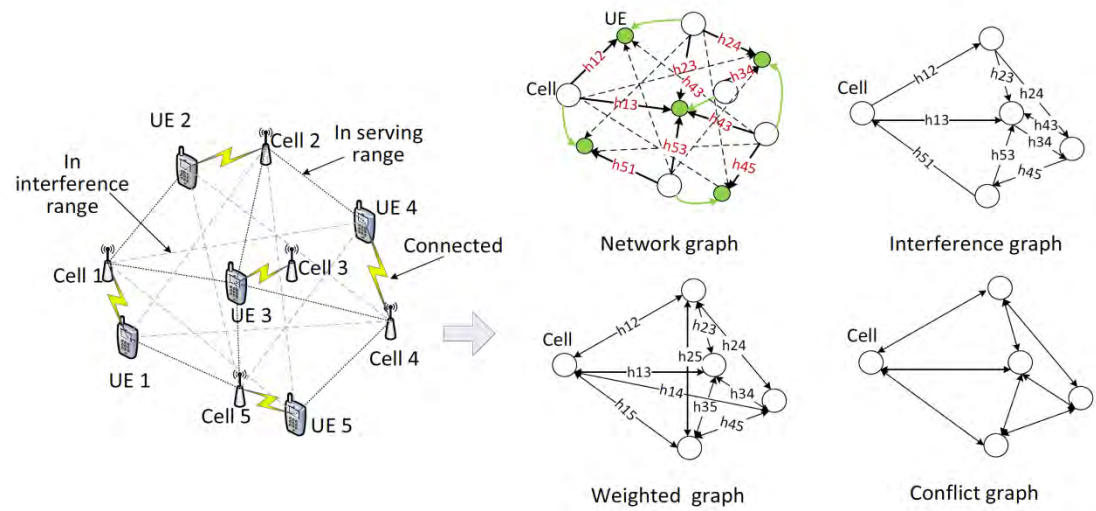
Radio and core API and Southbound Protocol

- **Control plane APIs allowing fine grain radio and core control and monitoring**
- **Platform- neutral and extendable protocol message service**
 - Language agnostic
- **Optimize message footprint**
 - Aggregation
 - (de)serialization
- **Asynchronous control channel**
 - Queue
 - Pubsub communication model
- **Supported network topologies**
 - P2P, P2MP, and possibly others

Message	Field	Usage
Configuration request	Configuration type	Type of configuration, either set or get
	Cell configuration flag	Bit map of the requested cell configuration
	Cell configuration list	List of cells (in IDs) to request configuration
	UE configuration flag	Bit map of the requested UE configuration
	UE configuration list	List of UEs (in IDs) to request configuration
Configuration reply	Cell configuration	Requested cell configuration report
	UE configuration	Requested UE configuration report
Status request	Status type	Can be periodical, one-shot, event-driven
	Status period	Period in Transmission Time Interval (TTI)
	Cell status flag	Bit map for the requested cell status
	Cell list	List of cells (in IDs) to request the status
	UE status flag	Bit map for the requested UE status
	UE status list	List of UEs (in IDs) to request the status
Status reply	Cell status	List of cell including the statistic reports
	UE status	List of UE including the statistic reports

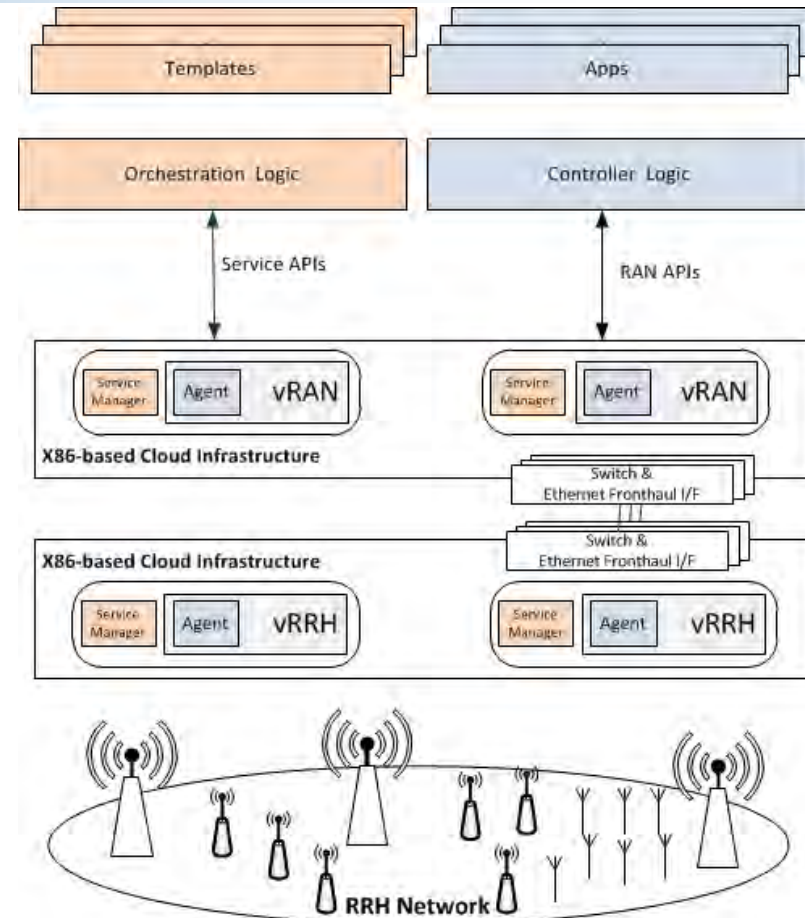
Network Abstraction and graph

- **Effective representation of the network state at different network levels allowing**
 - fine-grained programmability, coordination and management of atomic or composed services across different domains/regions via
- **Network graphs can be separated based on**
 - Region, operator, cell, ...
- **Encompass data models**
 - Time-frequency status and resources
 - Spatial capabilities
 - Key performance indicators



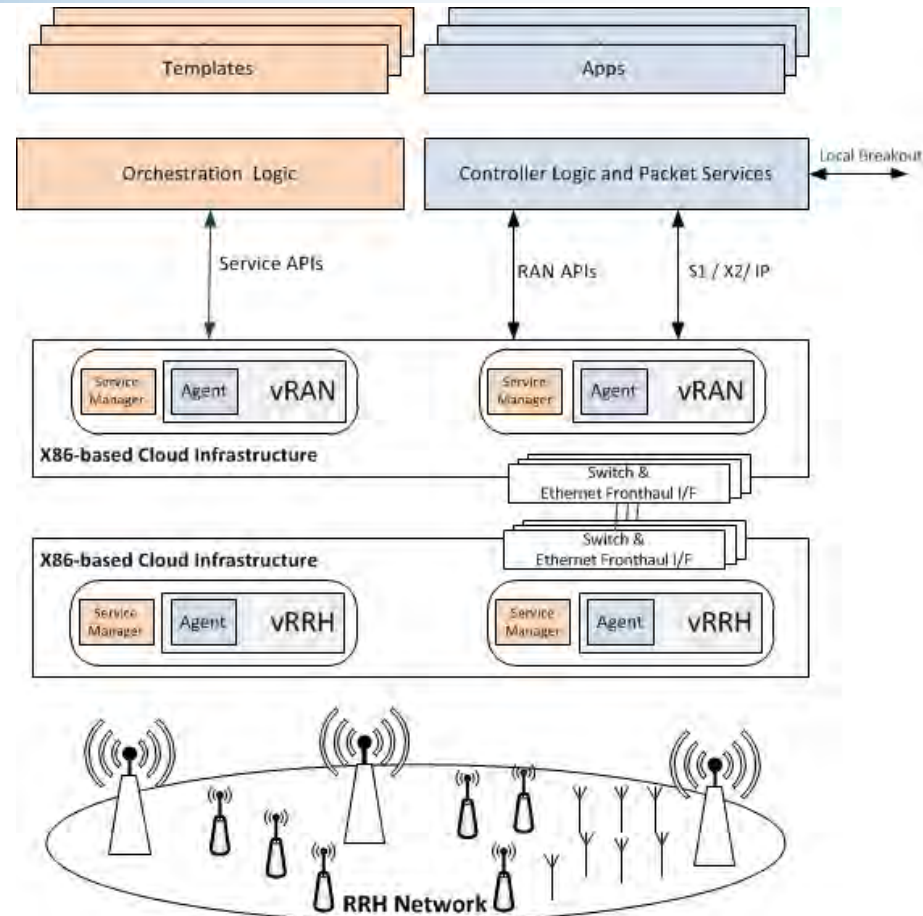
Scalability, Control delegation mechanisms, and Realtime Control

- **Guarantee a (quasi-) deterministic reaction time of a control command triggered by the controller**
- **Hierarchical controller logic**
 - non-time critical → centralized entity
 - time critical → edge entity
 - May offloaded time critical operation to an agent acting as a local controller
- **Interplay with the orchestrator**



Low Latency Edge Packet Service

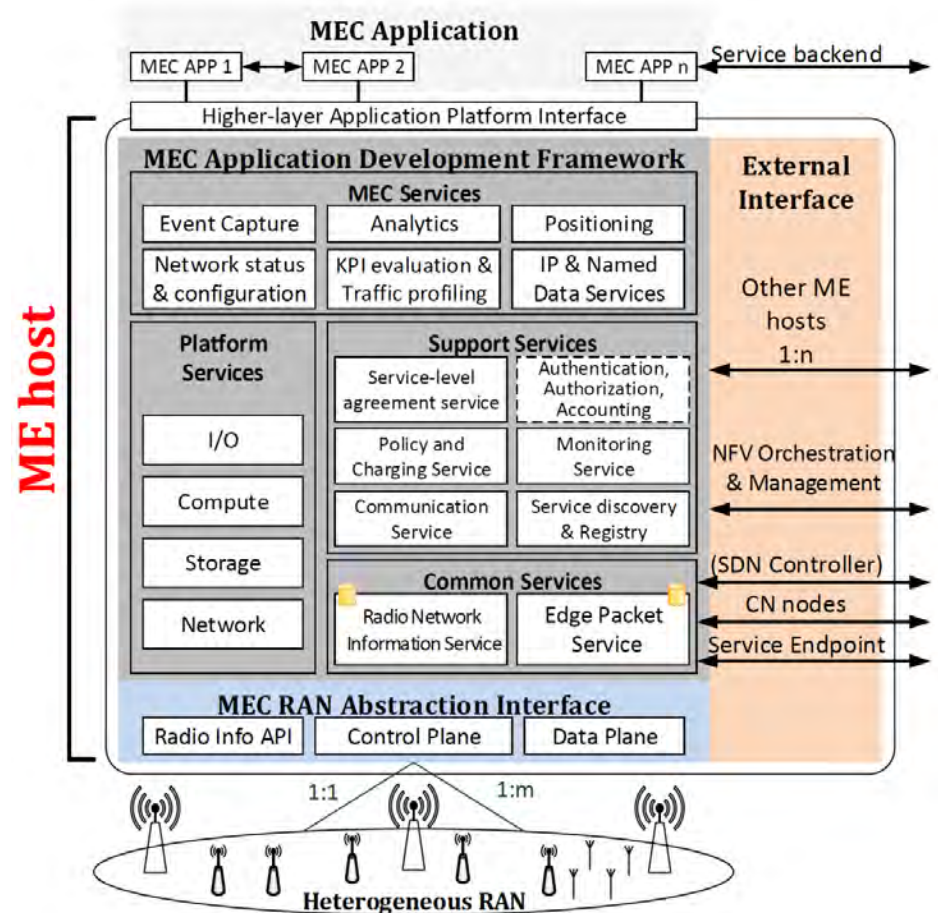
- **Coupling with the SDN controller**
 - Manages the SDN-enabled switches
- **Local breakout**
 - Traffic redirection
 - Maintain state information
- **Interplay with the S-GW**
- **network operator oriented-VNF at the edge**
 - LightEPC function to handle specific traffic (MTC)
 - Security functions to analyze and react locally to malicious traffic
- **Follow Me Edge**
 - Edge service following mobile users
- **Native IP, and CDN/ICN support**



Low Latency Edge Packet Service

MEC Design Principles

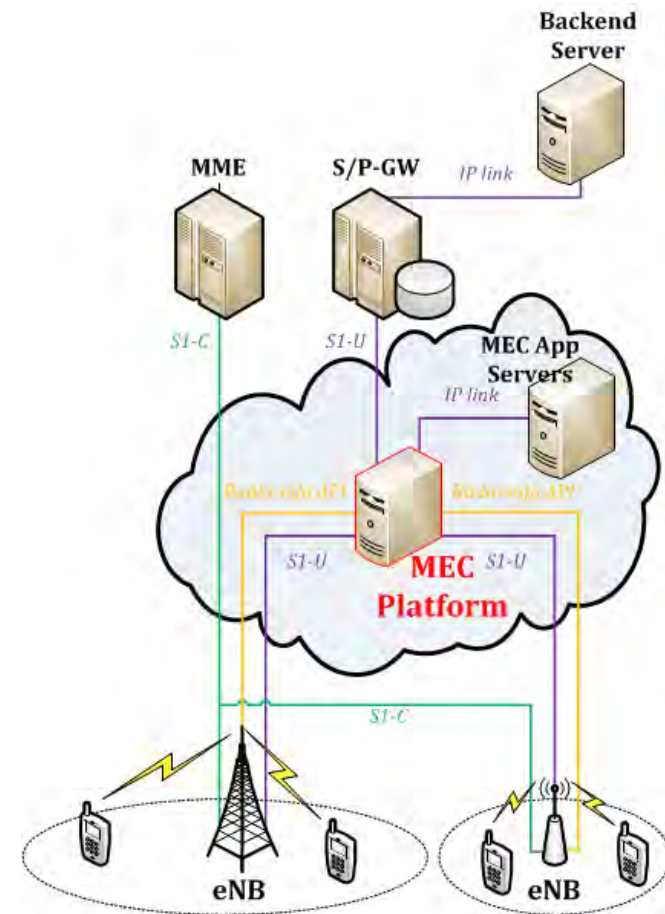
- **Design principles**
 - Tight coupling between MEC and SDN
 - Exploit the interplay between control-plane and user-plane
- **Realtime/low-latency SDN+MEC controller**
- **RAN functional split implication and MEC**
 - CU and/or DU in MEC
 - Native IP service endpoint at eNB and GTP (de)encapsulation in MEC
- **Abstract communication interface**
 - Message queue, RestAPI
- **Agnostic to the underlying RAN technologies**
 - Support of RNIS (eLTE, and NR)
 - Interaction with EPC and NG-CORE
- **Edge packet service**
 - Interplay with S-GW and proxy EPC
 - ☞ Placement of packet switching
 - Lawful interception
 - Policy and charging
 - Traffic shaping



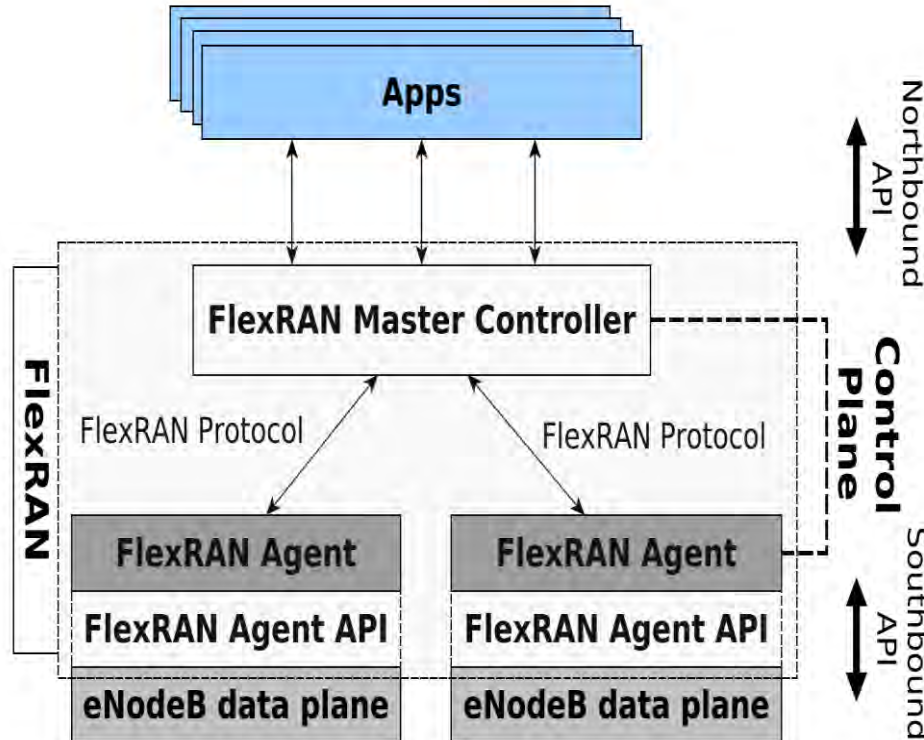
Low Latency Edge Packet Service

MEC Design Principles

- **Design principles**
 - Tight coupling between MEC and SDN
 - Exploit the interplay between control-plane and user-plane
- **Realtime/low-latency SDN+MEC controller**
- **RAN functional split implication and MEC**
 - CU and/or DU in MEC
 - Native IP service endpoint at eNB and GTP (de)encapsulation in MEC
- **Abstract communication interface**
 - Message queue, RestAPI
- **Agnostic to the underlying RAN technologies**
 - Support of RNIS (eLTE, and NR)
 - Interaction with EPC and NG-CORE
- **Edge packet service**
 - Interplay with S-GW and proxy EPC
 - ☞ Placement of packet switching
 - Lawful interception
 - Policy and charging
 - Traffic shaping



Deployment Example: SD-RAN MOSAIC-5G FlexRAN Platform



▶ FlexRAN Master Controller

- ▶ Top level controller

▶ FlexRAN Agent

- ▶ Local controller
 - ▶ Limited network view
 - ▶ Delegated control

▶ FlexRAN Protocol

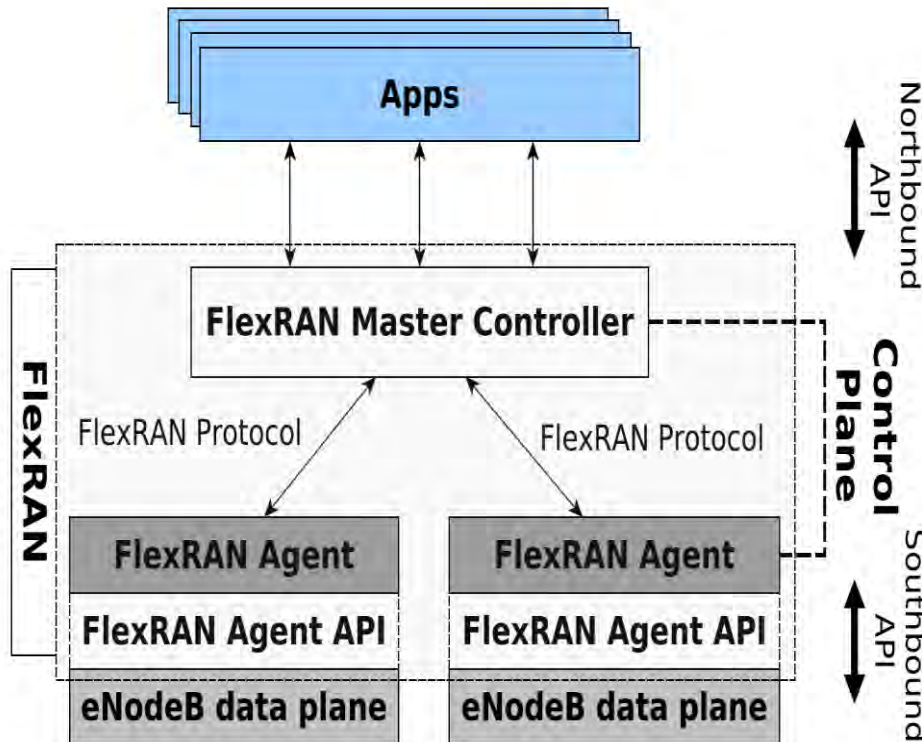
▶ Agent-to-master

- ▶ Statistics
- ▶ Configurations
- ▶ Events

▶ Master-to-agent

- ▶ Control commands

Deployment Example: SD-RAN MOSAIC-5G FlexRAN Platform



Salient Features

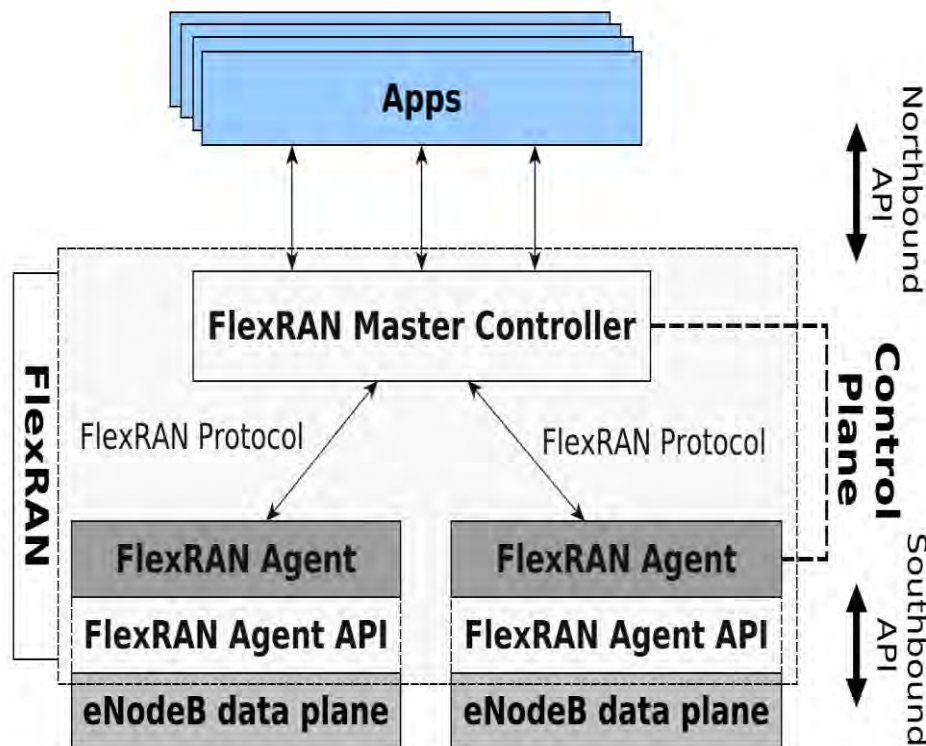
Control & Data Plane Separation

Real-time Control

Virtualized Control Functions

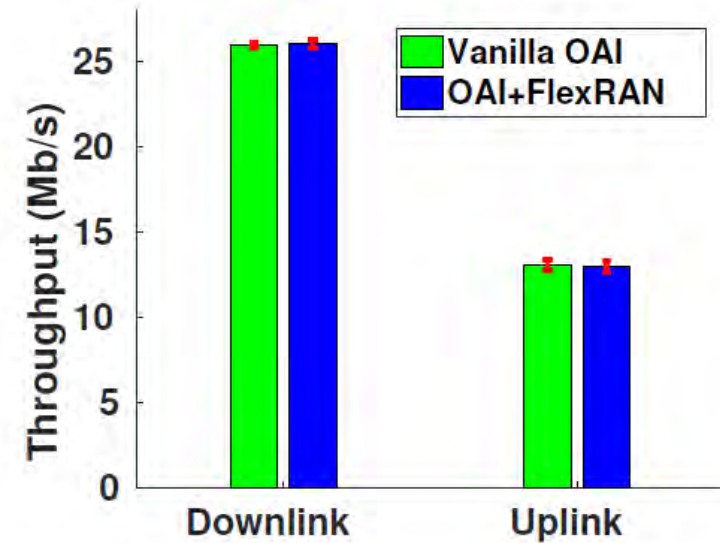
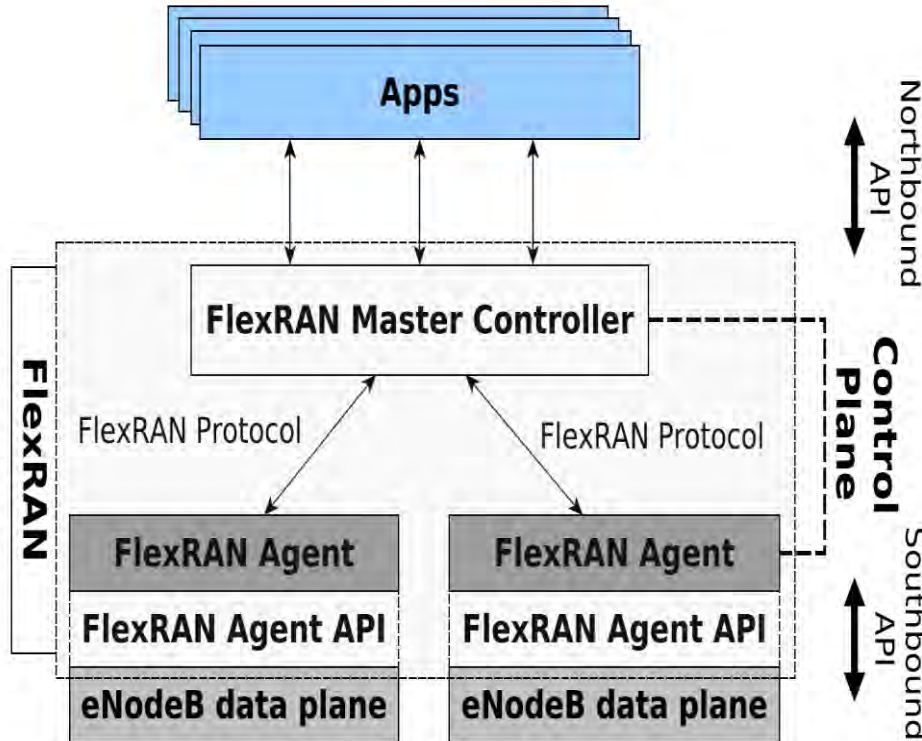
Control Delegation

Deployment Example: SD-RAN MOSAIC-5G FlexRAN Platform



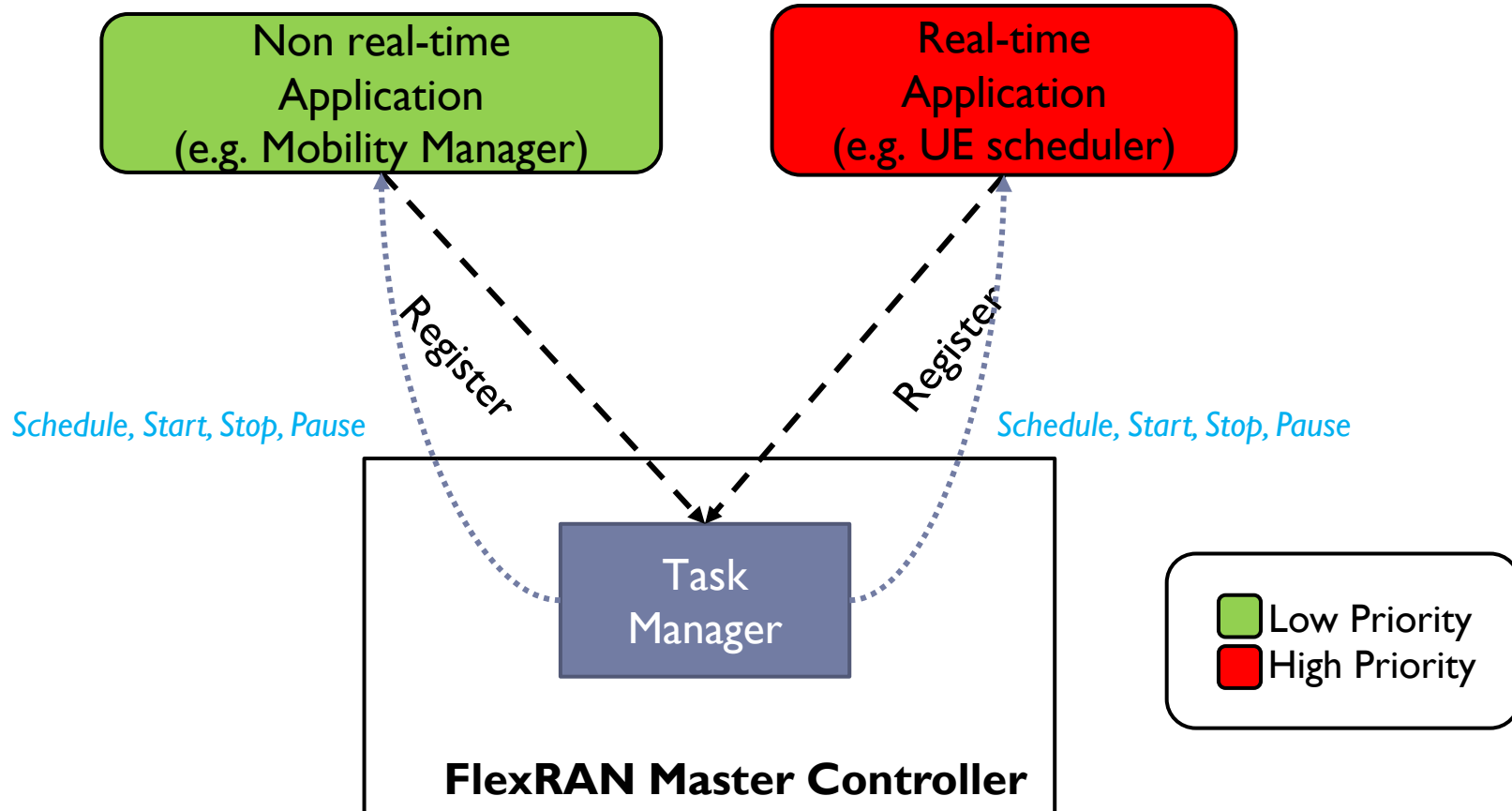
API Call Type Description	Example
Get/Set configurations	UL/DL cell bandwidth
Statistics	UE Tx queue sizes
Issue commands	Scheduling decisions
Event-triggers	New transmission time interval notification
Control Delegation	Perform scheduling centrally or locally

Deployment Example: SD-RAN MOSAIC-5G FlexRAN Platform



Deployment Example: SD-RAN

FlexRAN Application Type

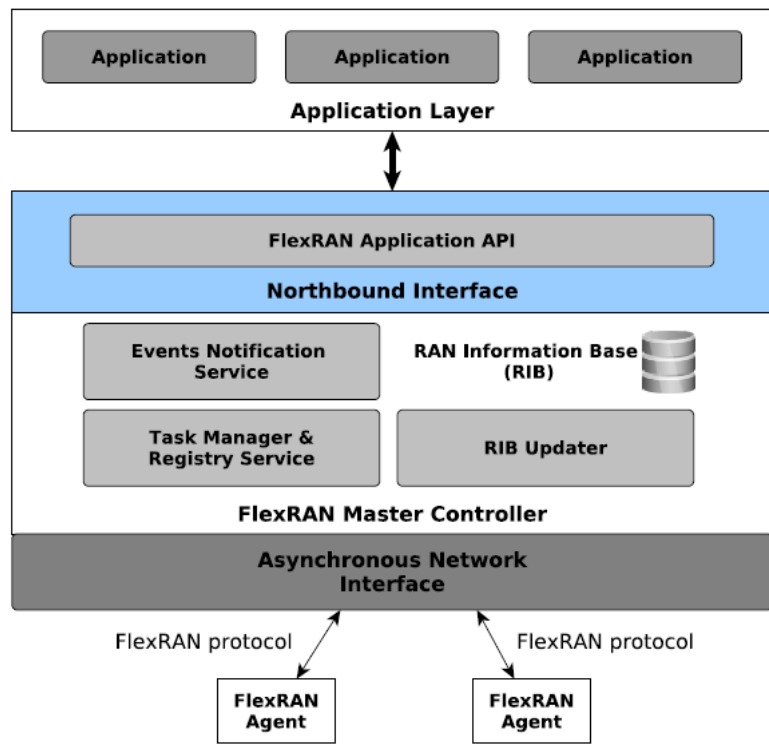
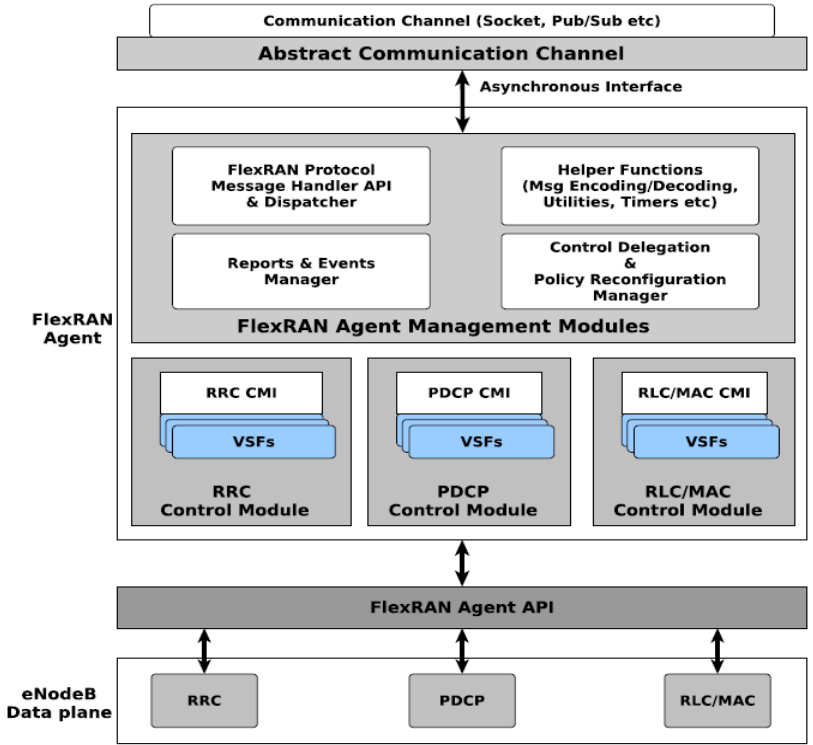


Deployment Example: SD-RAN FlexRAN Architecture



Agent

Controller

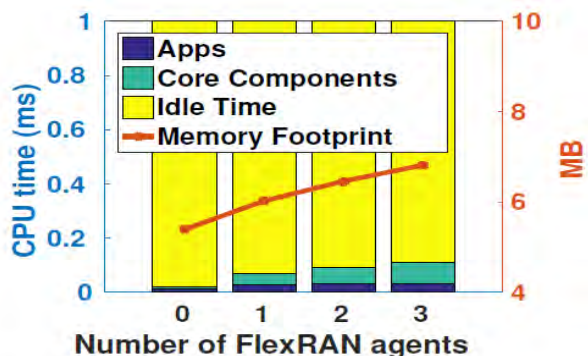
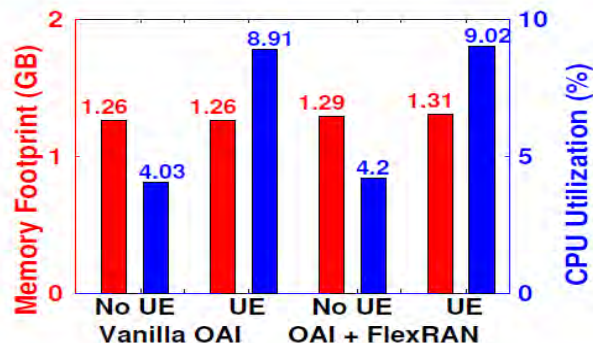


Deployment Example: SD-RAN

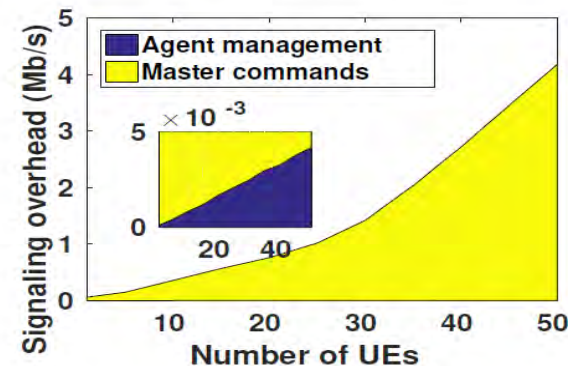
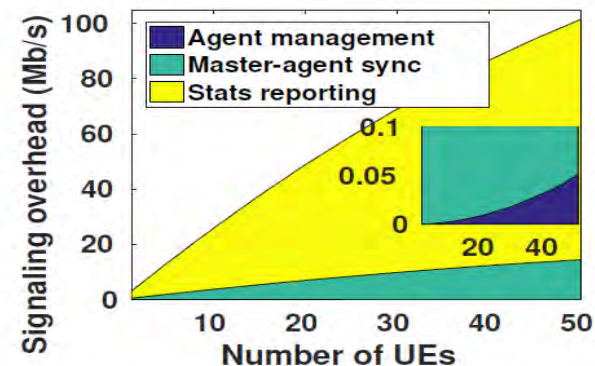
FlexRAN Scalability



CPU Utilization and memory footprint



Agent-to-controller Controller-to-agent



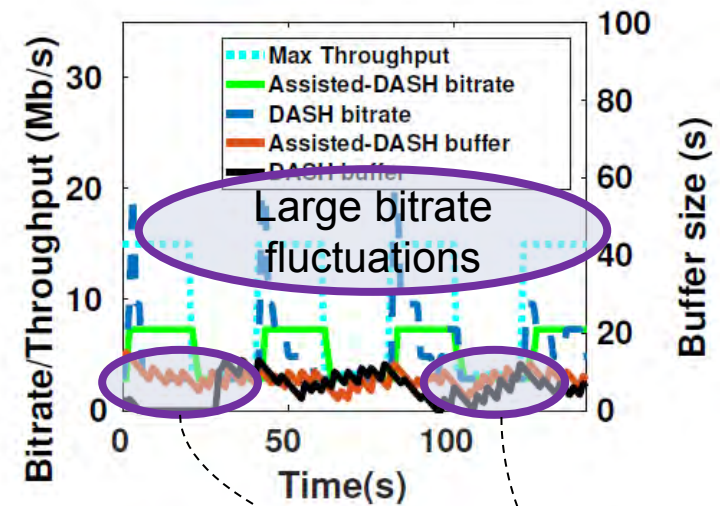
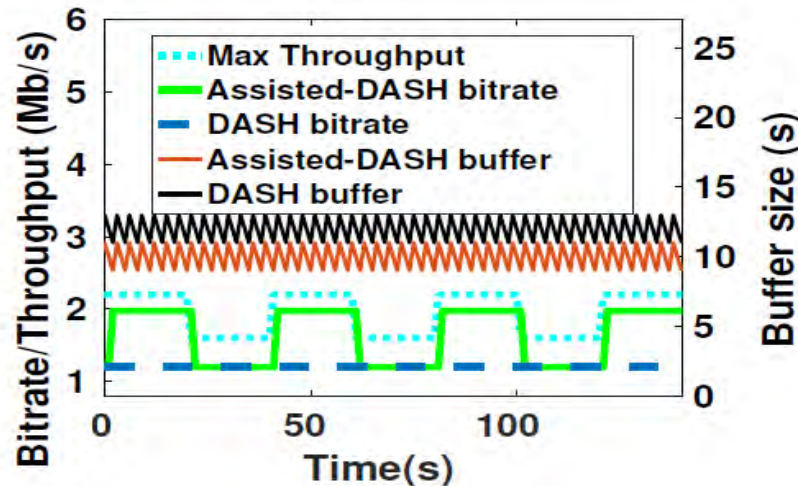
Deployment Example: SD-RAN

FLEXRAN: DASH Rate Adaptation



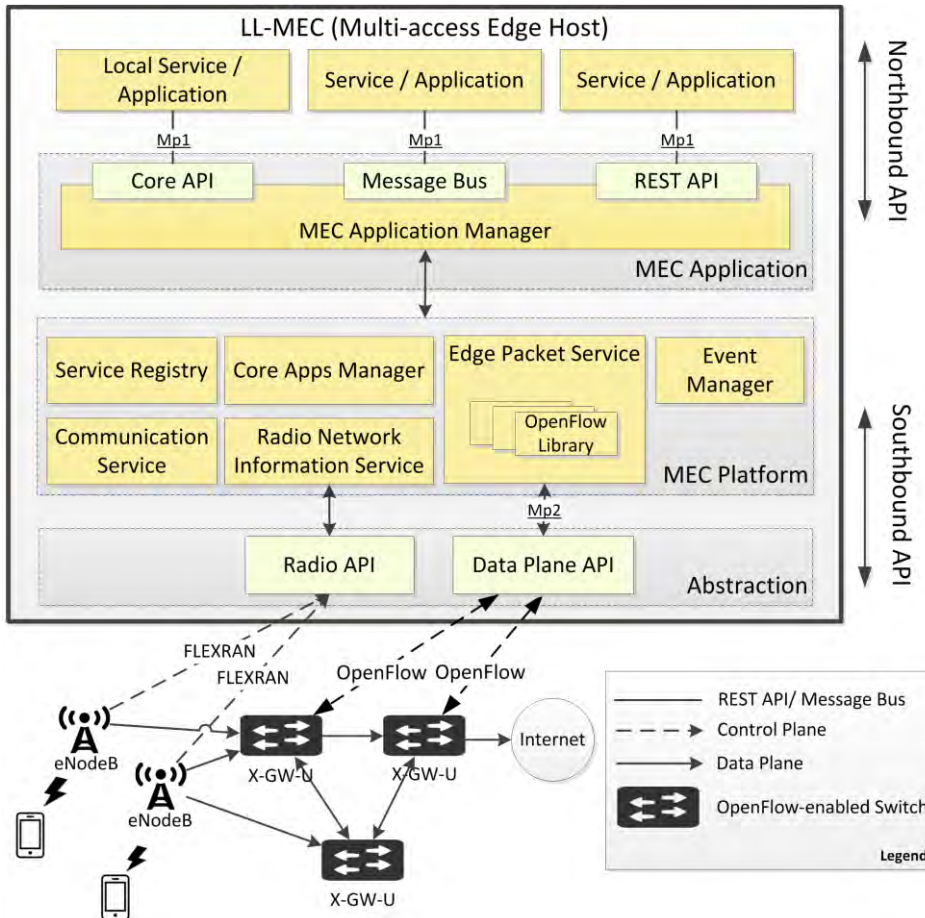
Low Variability

High Variability



CQI	TCP Throughput (Mb/s)	Max sustainable bitrate (Mb/s)
2	1.63	1.4
3	2.2	2
4	3.3	2.9
10	15	7.3

Deployment Example: MEC MOSAIC-5G LL-MEC Platform



■ Application

- Limitless applications to be developed
- No need to know the detailed knowledge for underlying network

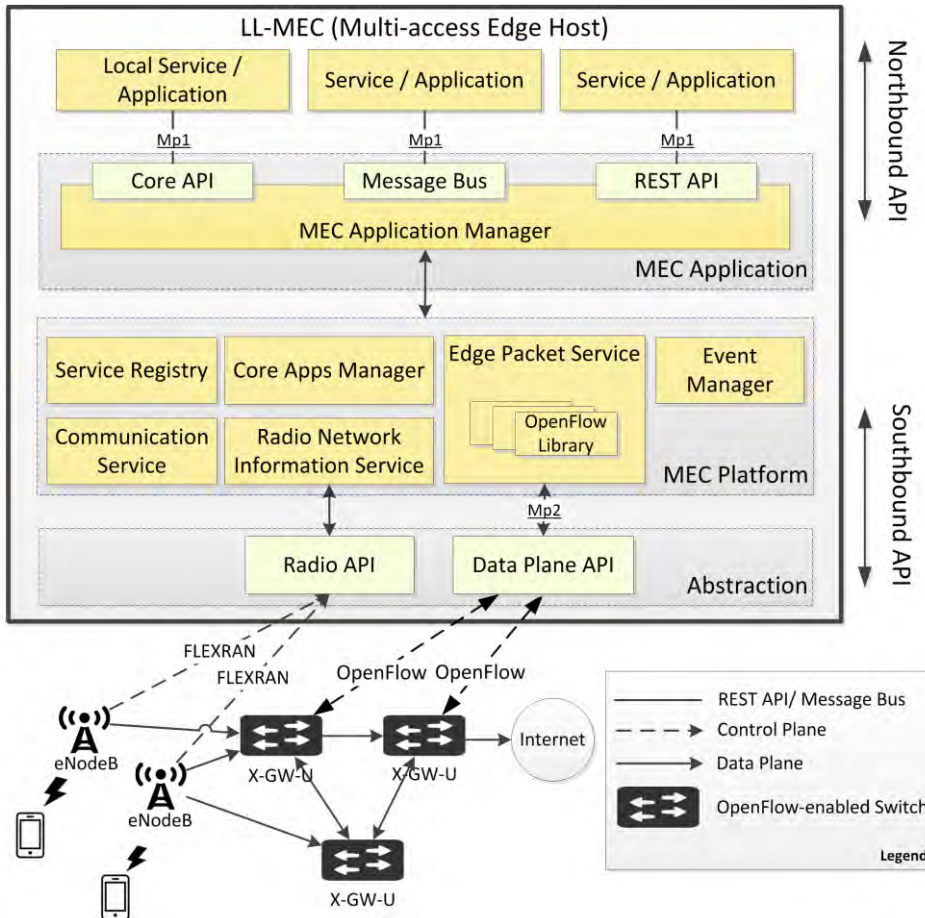
■ Platform

- Where core functionality is located
- Provide the necessary functionality for application developer to focus on their specific purpose rather than how to interact with network

■ Abstraction

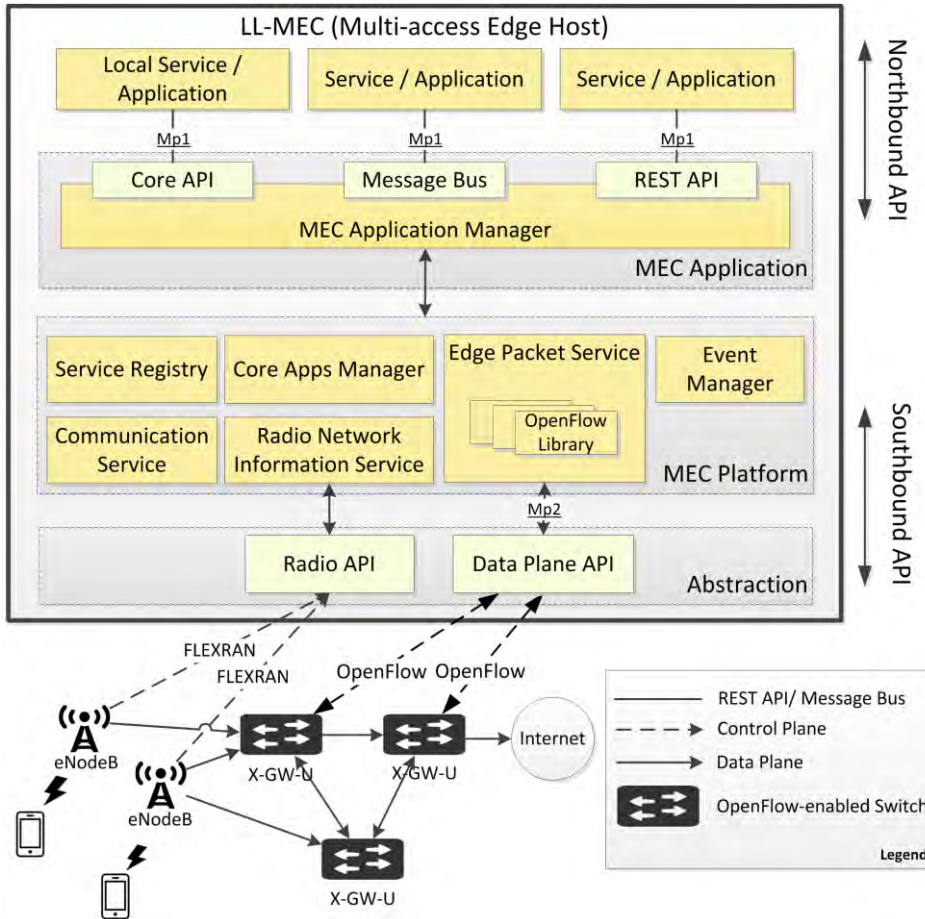
- Functions on eNodeB and OpenFlow switch, physical or software
- Provides well-defined and uniform API, where SDN involves

Deployment Example: MEC MOSAIC-5G LL-MEC Platform



- **Provide low-latency and high-bandwidth application/service environment at the edge of network**
 - Two type of apps: low-latency and elastic
 - Able to control and monitor RAN in realtime
- **Retain the compliance with 3GPP spec as well as the functionalities specified by ETSI MEC**
- **Default / Dedicated core network deployed at edge**
 - Control and data plane separation in core network
- **SDN MEC controller**
 - Mobile network control and monitoring in support of programmability
- **Verticalization of core network at network edge**
 - Network can be sliced and adapted per service

Deployment Example: MEC MOSAIC-5G LL-MEC Platform



Salient Features

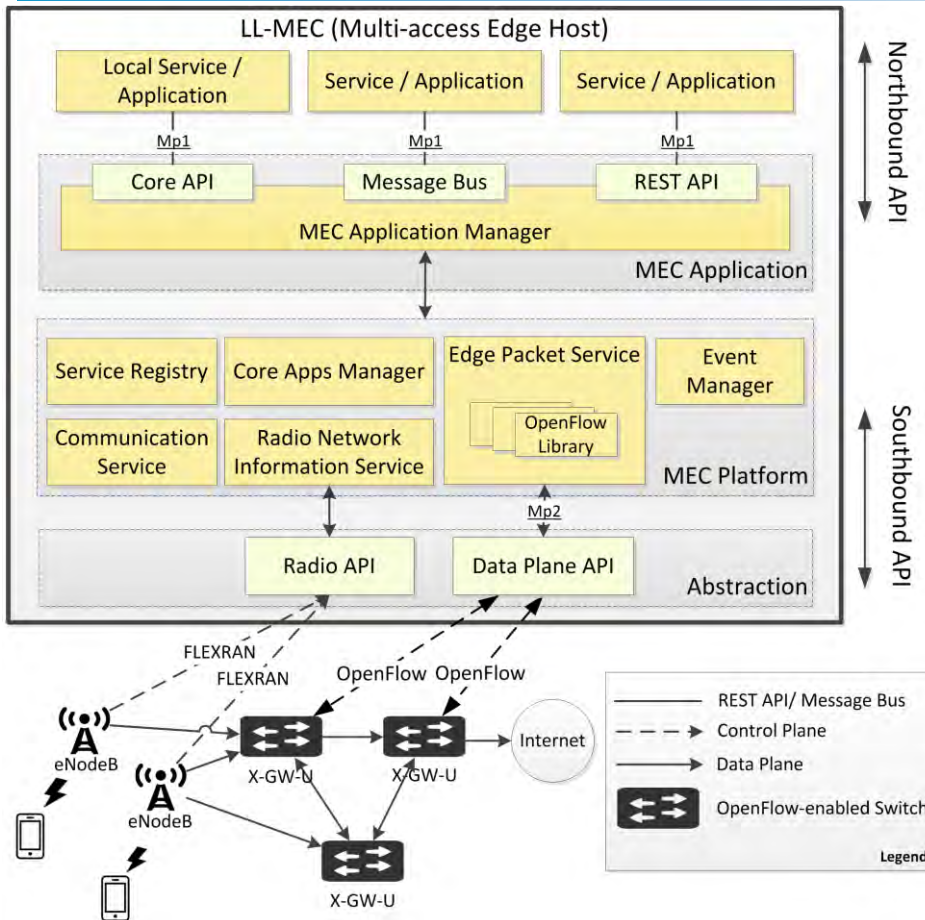
Joint Data and Control plane programmability

Low latency service access

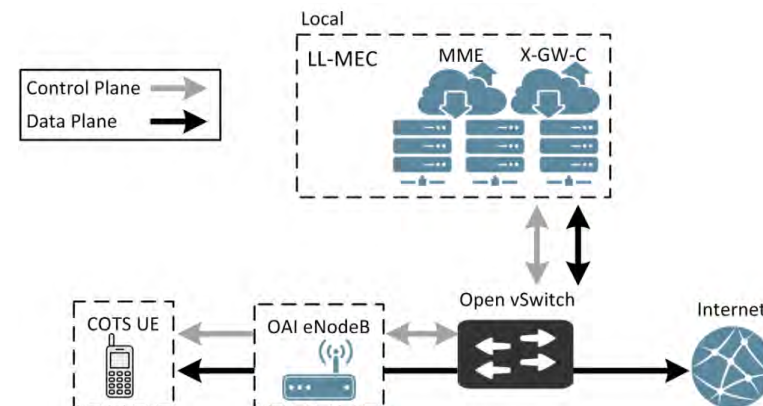
MEC application manager

OF and OVS

Deployment Example: MEC MOSAIC-5G LL-MEC Platform



	Setup	Mean (MB/s)
Downlink	Legacy LTE	15.691
	LL-MEC	15.112
Uplink	Legacy LTE	8.214
	LL-MEC	8.197



Deployment Example: MEC

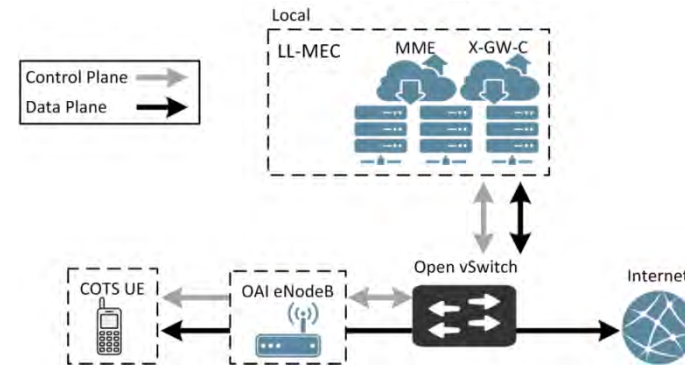
LL-MEC Measured RTT



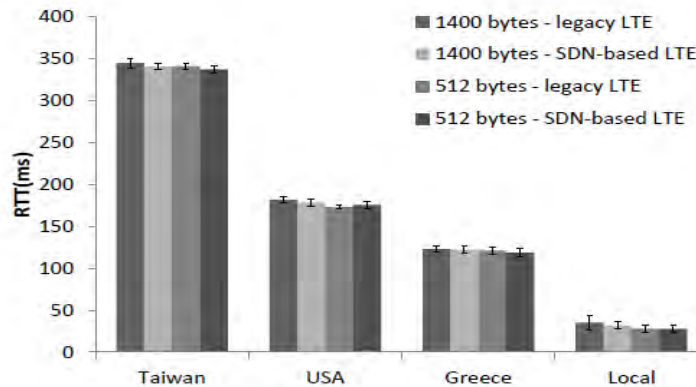
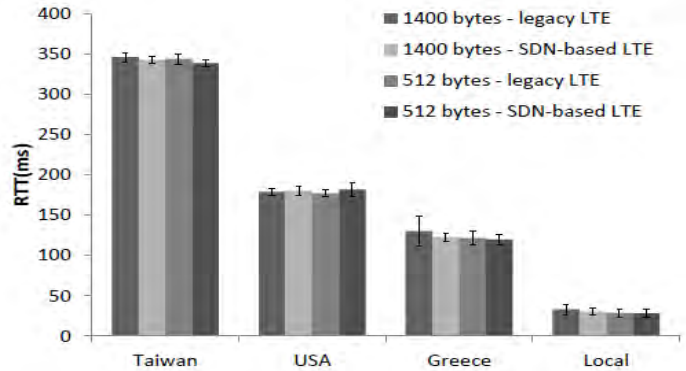
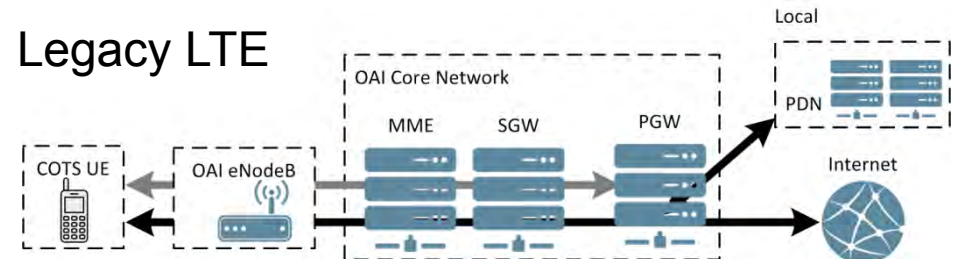
IDT=1, 0.2s

Setup

MEC framework and SDN-based LTE



Legacy LTE



Deployment Example: MEC

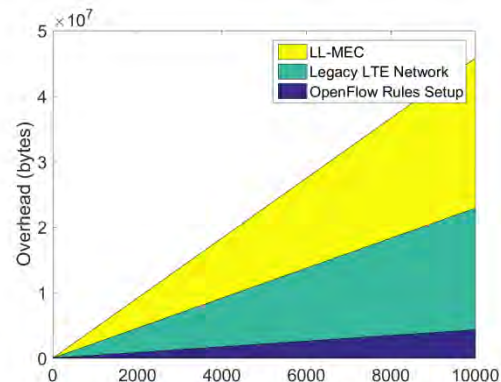
LL-MEC Scalability



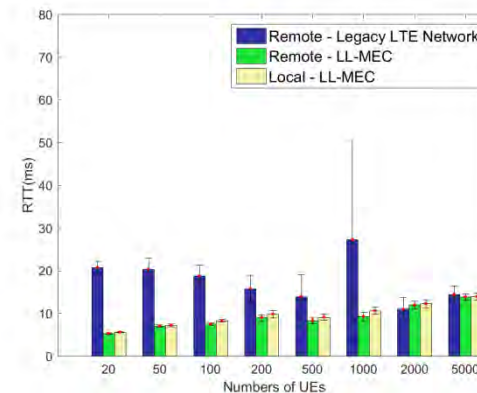
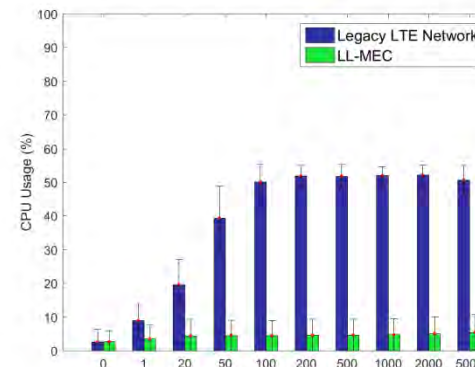
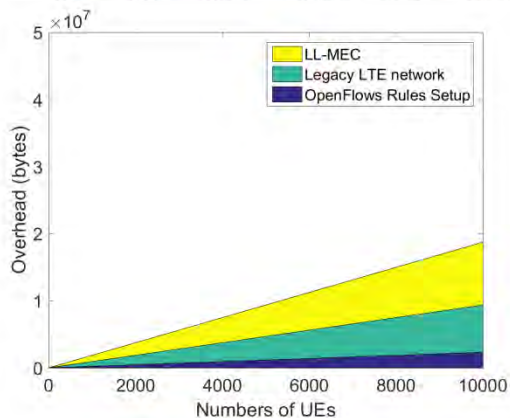
Control-Plane Signaling

CPU Usage and RTT

Default Bearer

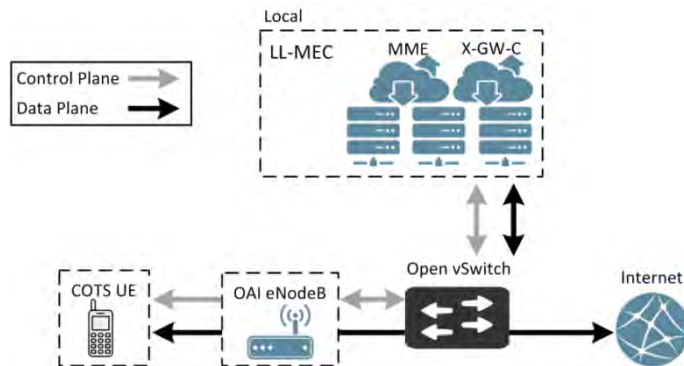


Dedicated Bearer

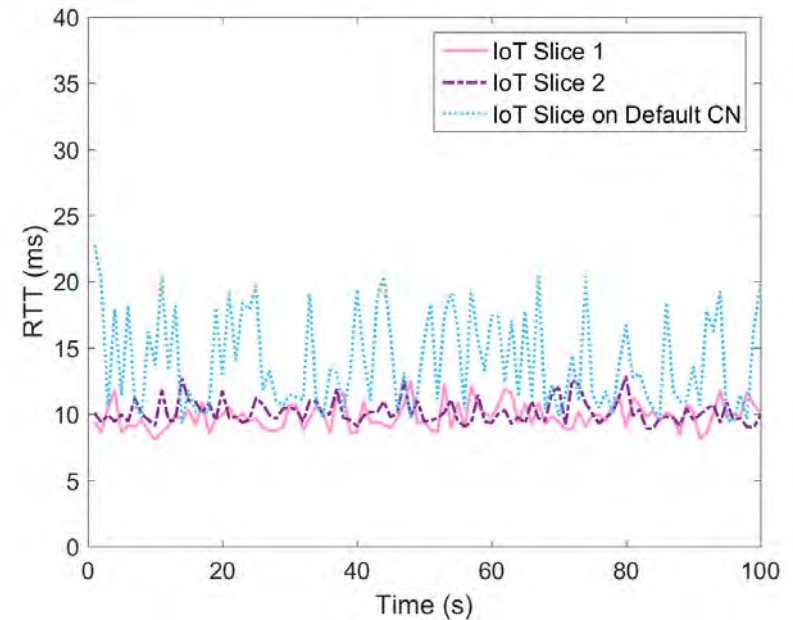


IoT gateway

- Dedicated data plane slice
- Consider 1000 UEs for each slice



Latency of isolated / non-isolated slices



Network Slicing

Network Slicing

Network slice and store concepts

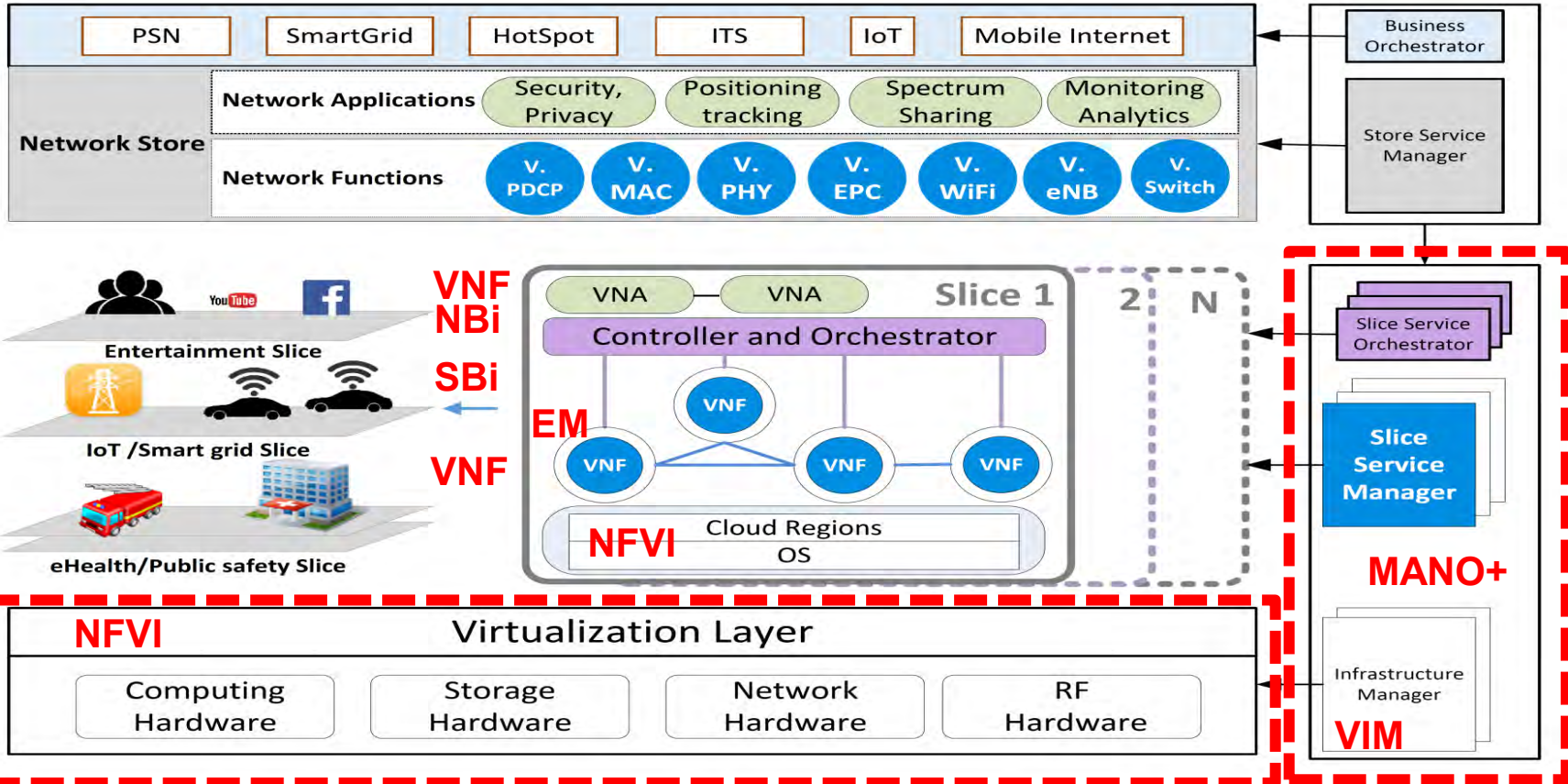
- **Network slice is**
 - A set of independent virtual networks
 - A set of slice-specific SLA and QoS requirements
 - Running on the same infrastructure resources

Network slicing

- **Slice manifest describes the business application across three planes**
 - Business, Service, Infrastructure
- **Network store allows creation of a slice for each virtual network through digital distribution platforms**
 - Network functions and network applications
- **Network slice is an E2E virtual network with a set of SLA/QoS that is instantiated on a common shared infrastructure**
 - Chain and compose adequately configured network functions, network applications, and underlying cloud infrastructures
 - Map and place them onto the infrastructure resources and assign target performance metrics
 - Program and scale them according to a particular business application scenario

Network Slicing

Network slice and store concept



Network slicing

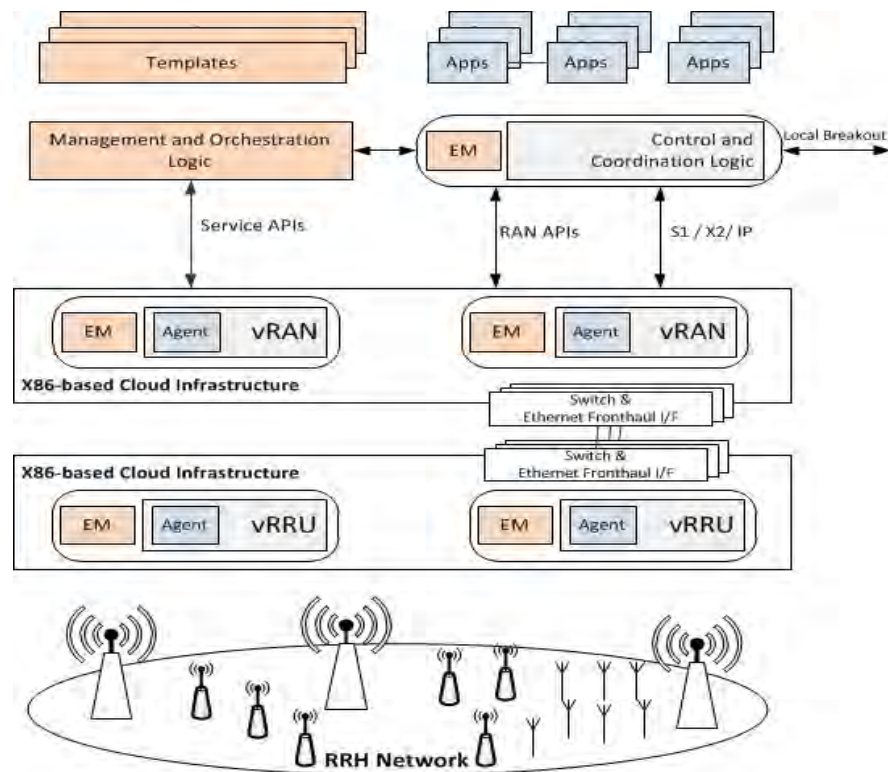
NFV-SDN-MEC Interplay

- **Scaling capacity and managing a dense and potentially time-varying network require a tight coordination and programmability**
 - Obj: Decouple the control plane from the data plane
- **Flexibility to change the network service definition on-the-fly to deal with spatiotemporal network and traffic diversity**
 - Obj: abstraction and programmability of network functions
 - ☞ Control plane and data plane
- **Multi-service multi-tenant networks**
 - Obj: dynamic network service composition from reusable network functions
 - ☞ Nested chaining following micro-service design pattern
 - Obj: resource sharing (infra, radio, and spectrum)

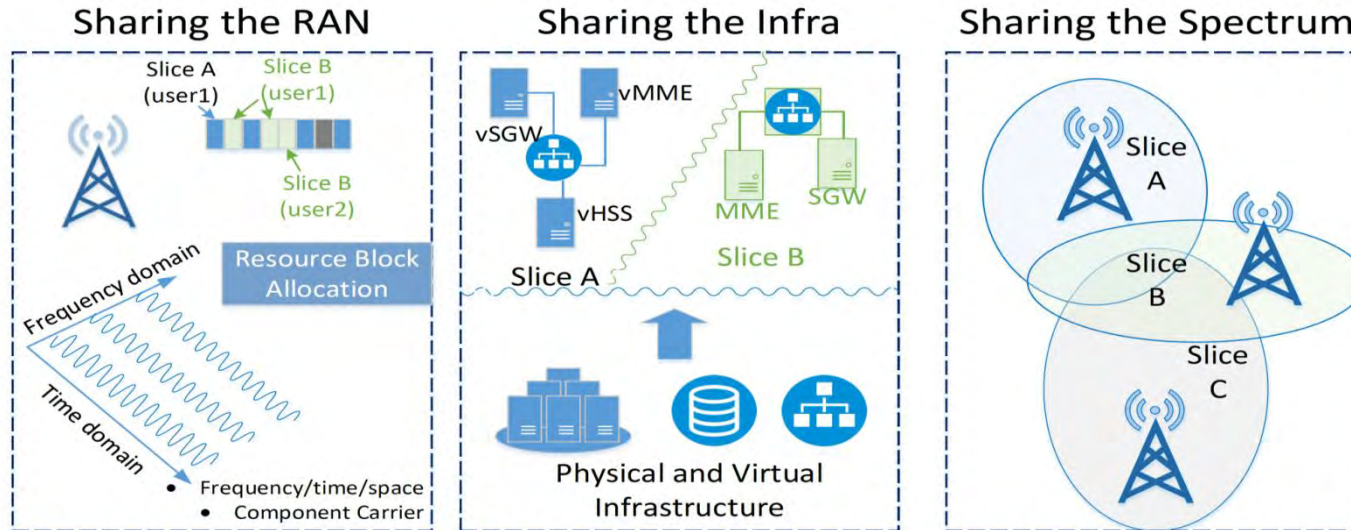
Network slicing

NFV-SDN-MEC Interplay

- **Automated Multi-domain orchestrator**
 - On-the-fly changes in network service definition
 - Native support of network slicing
 - Network store
- **Coordinated data-plane and control plane programmability**
 - Vertical APIs
 - Network intelligent
 - Hierarchical controller logic managed by the orchestrator
- **Micro-service architecture for VNF and PNF**



Network slicing



- **Sliceable elementary resources**
 - [RRU/Antenna, Fronthaul, CU, DU, Backhaul, CN]
 - [CPU/MEM/NET, Radio resources, spectrum]
 - [configuration, chain, placement]
- **Resource abstraction and network programmability** is a key to achieve the required flexibility in slicing

Network slicing

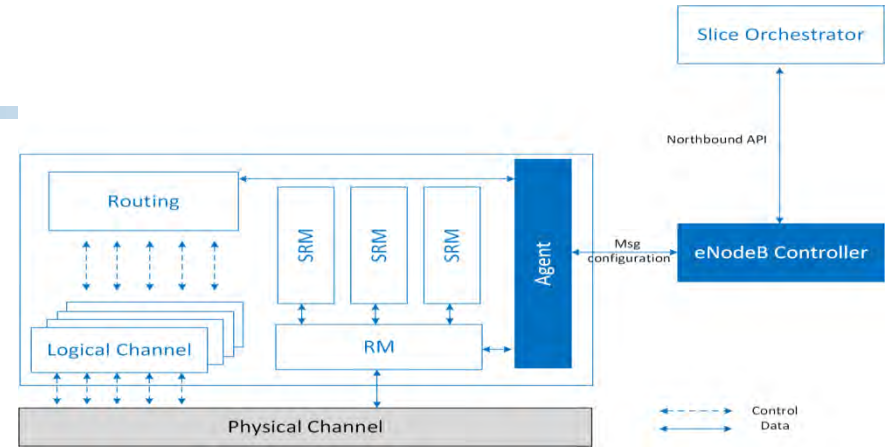
- **Slice strategy (two extremes)**
 - Isolation
 - + Dedicate elementary resources to the slice
 - ☞ Reduce slices elasticity/scalability
 - Resource Sharing:
 - + Exploit the **statistical multiplexing gain**
 - ☞ No hard performance guarantee
- **Tradeoff between slice isolation and resource sharing**
 - Dedicated and shared core networks
 - ☞ Dedicated control-plane and shared data plane
 - ☞ Dedicated data plane and shared control plane
 - ☞ Dedicated Control-plane and data plane (3GPPP eDECORE)
 - Dedicated and shared CU and DU and even RRU
 - ☞ Protocol stack isolation (data plane and/or control-plane) → contained CU and DU
 - ☞ Radio resource and spectrum isolation → different time-scale
 - ☞ Performance isolation (Slice-based KPI) → smart scheduling (multi-dimensional)
- **Multi-service network function chaining**
 - change the network service definition on the fly on per slice basis

Network slicing

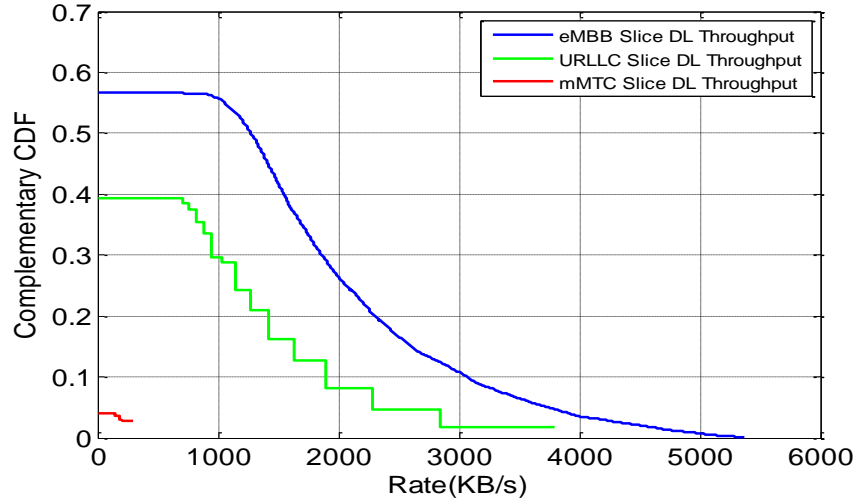
RAN

3 slices

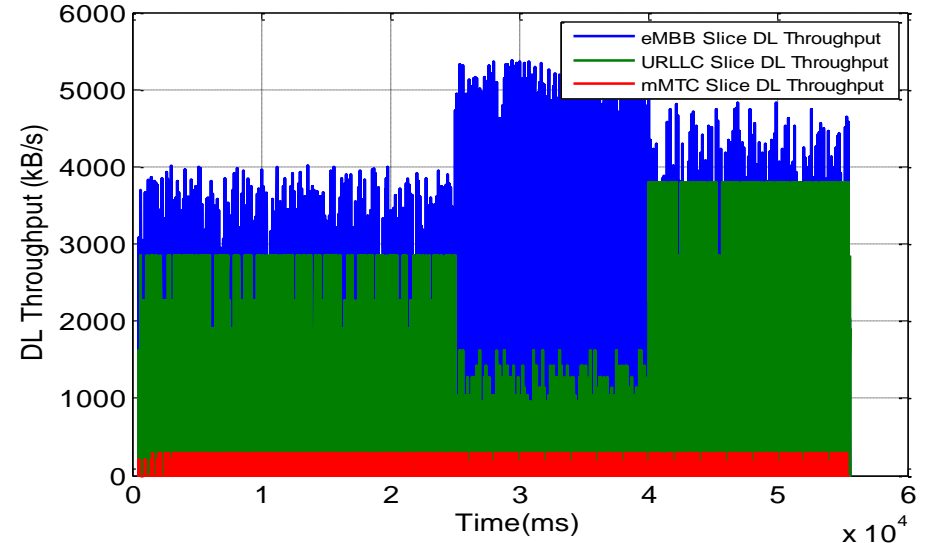
- Slice-specific scheduling
- Dynamic Slice RM
 - 👉 Enforce policy over time



DL Throughput



DL Rate

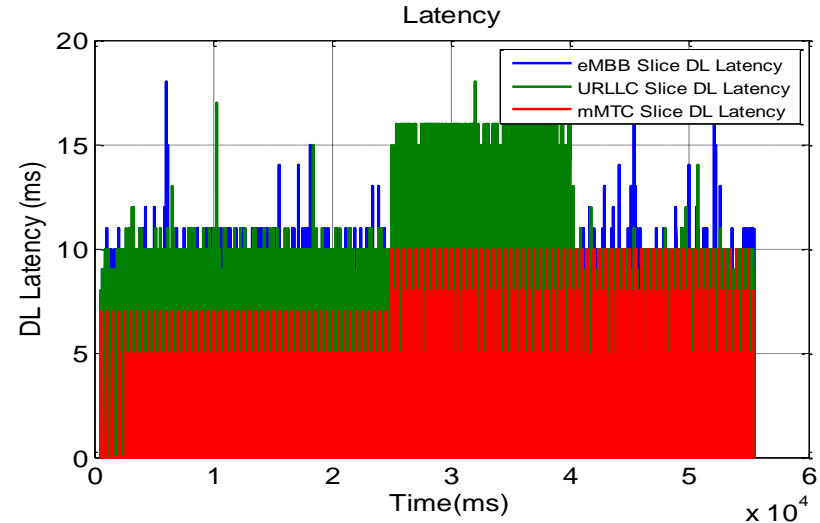
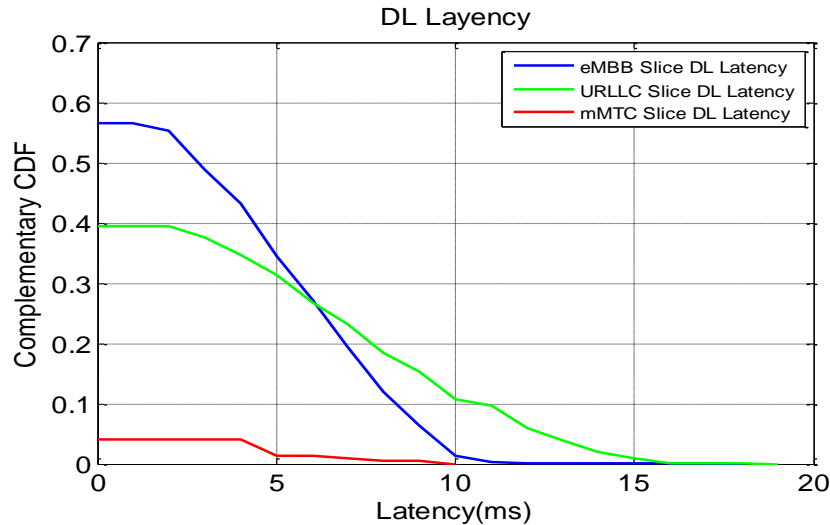
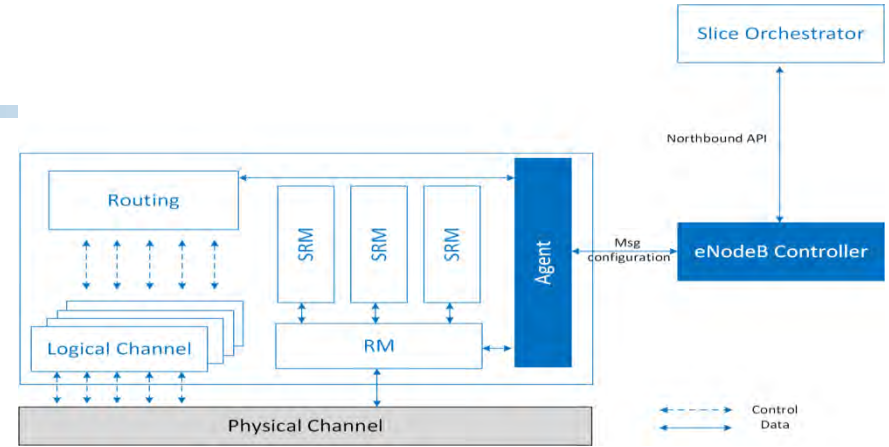


Network slicing

RAN

3 slices

- Slice-specific scheduling
- Dynamic Slice RM
 - 👉 Enforce policy over time

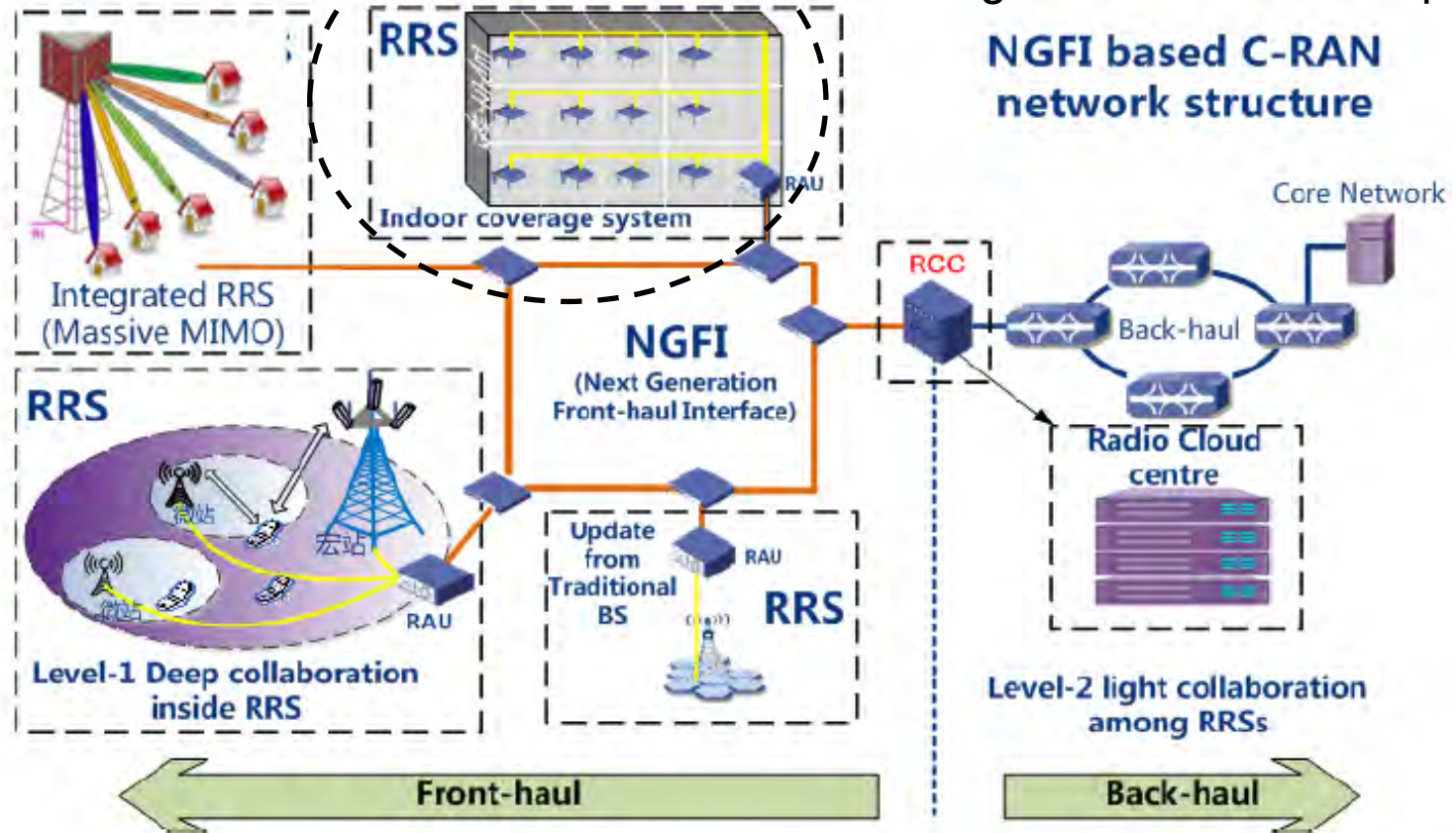


Part IV – Testbeds and Field Trials



NGFI (IEEE 1914) – Fronthaul Vision

Main target of EURECOM deployment



Background

- **3GPP access network is evolving towards a 3-tier network**
 - CU => DU => RRU
 - CU is envisaged to cover an area of 100-200 km radius
 - DU is envisaged to cover an area of 10-20 km radius
 - RRU covers < 2km radius
- **DU will be a data center for centralized radio signal processing**
- **Indoor networks (DAS)**

Considered RAN Splits in 3GPP evolution

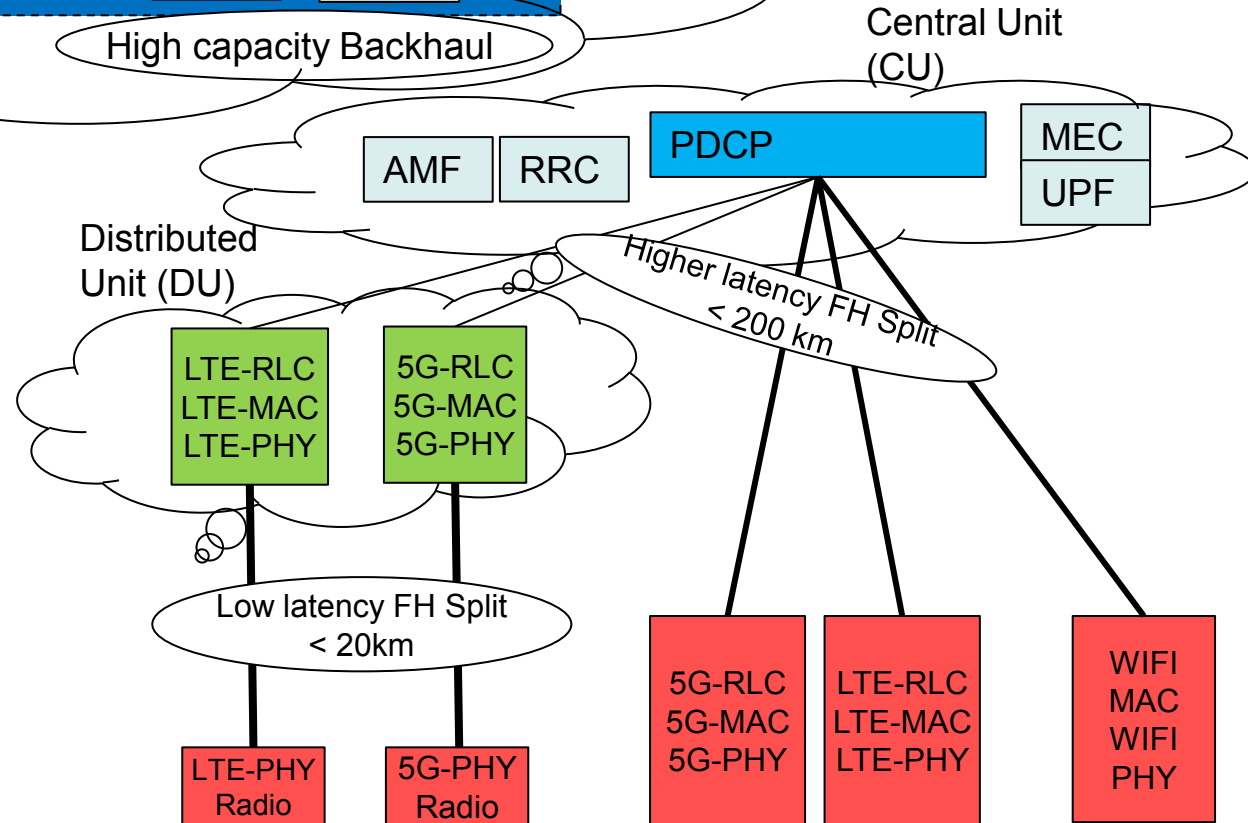


Capacity-Limited Backhaul

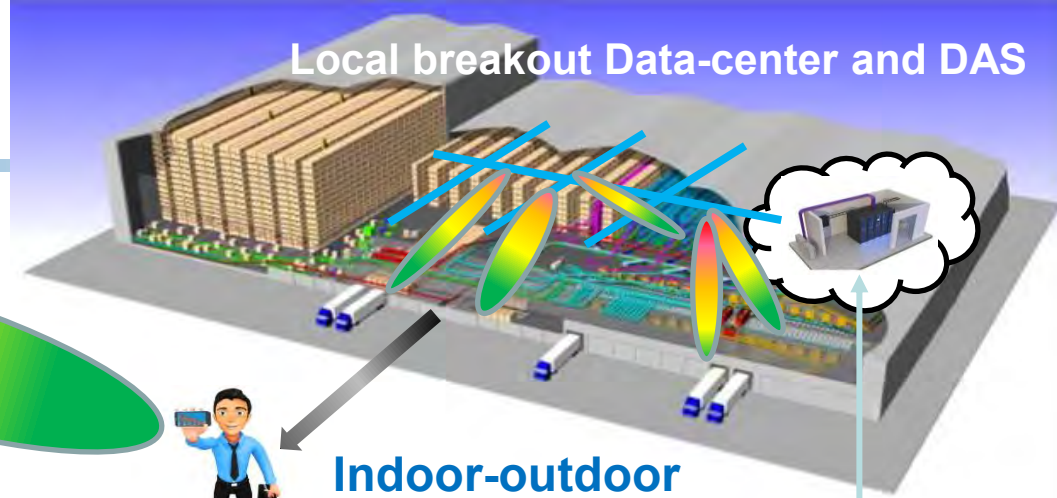
High capacity Backhaul

■ Fronthaul/Midhaul

- Lowlatency FH
- Highlatency FH



Deployment Example



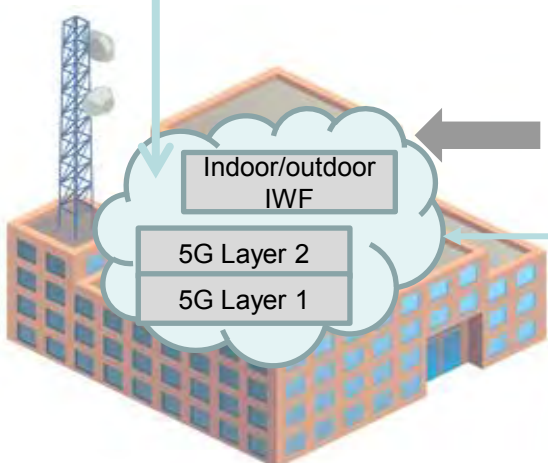
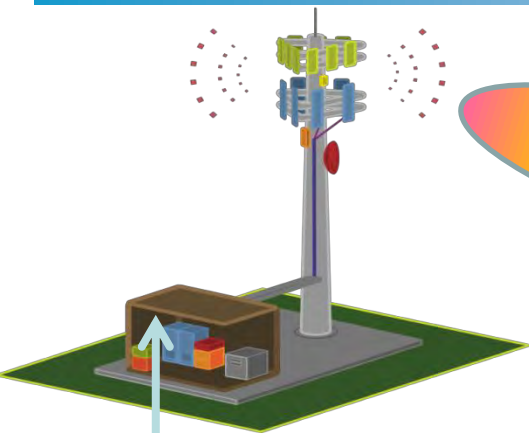
Local breakout Data-center and DAS

Indoor-outdoor mobility

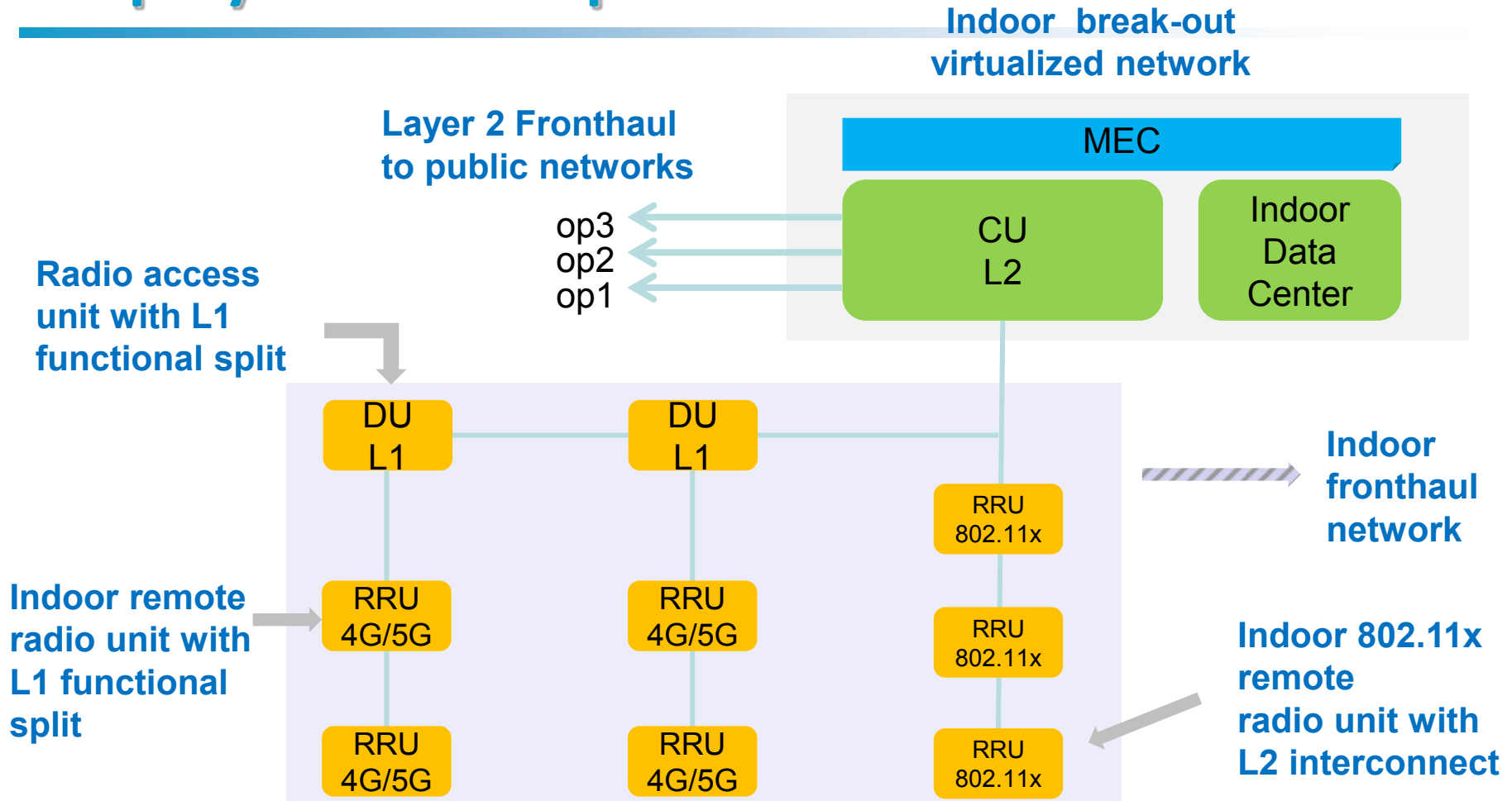
Indoor Layer 2 fronthaul
Public network
(medium speed, medium latency)

Outdoor Layer 1 fronthaul
(high-speed, low latency)

Radio cloud
Central office with Indoor/outdoor IWF



Deployment Example

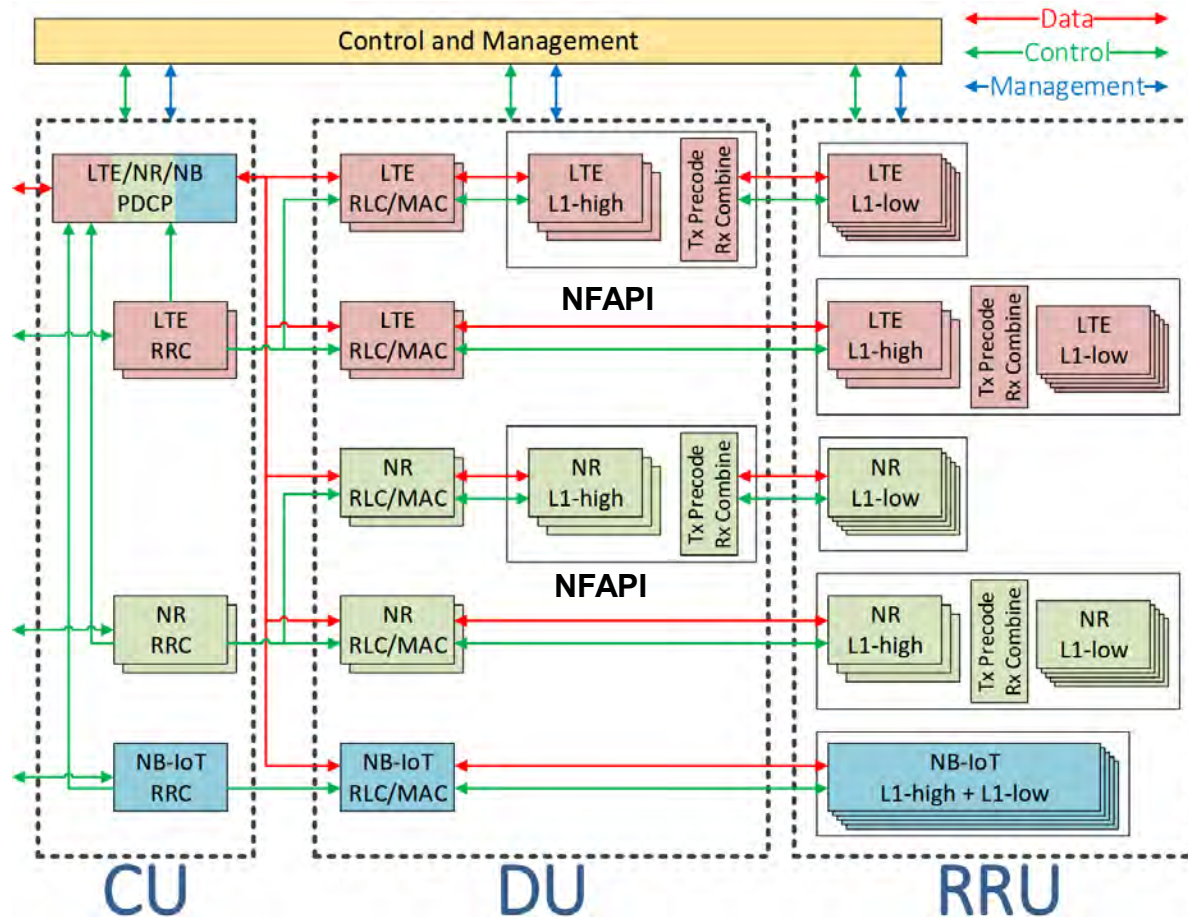


OPENAIRINTERFACE RAN USE-CASE

Commoditization of 3GPP Radio Systems and Open-Source : OAI Software Alliance

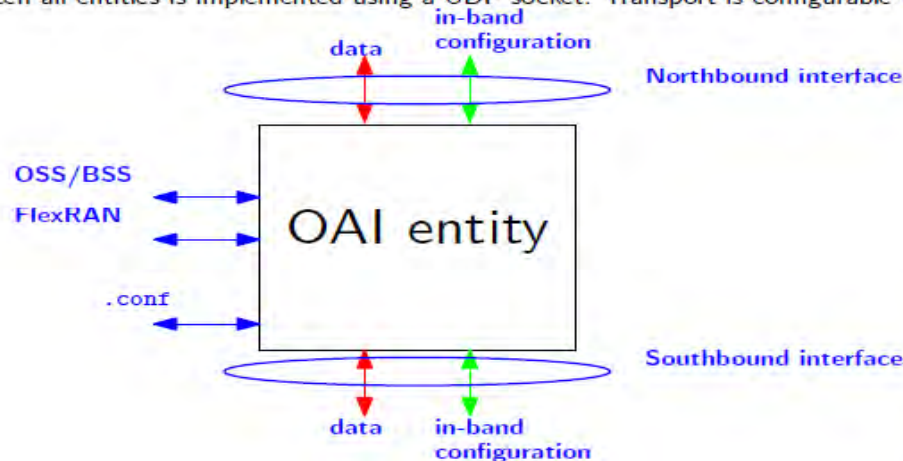
- **Today it is feasible to put a fully-compliant 4G eNodeB and EPC in a commodity x86 or ARM-based computer (or data center for a pool of eNodeBs)**
 - Emergence of “radio”-hackers in addition to commercial vendors
 - OAI Alliance
 - ☞ launched in 2014 as a “Fonds de Dotation”
 - ☞ 3GPP strategic members in 2015-2017 (Orange, TCL-Alcatel, Ercom, Nokia)
 - ☞ Many associate members (Cisco, B-COM, INRIA, IMT, TNO, III, Rutgers WINLAB, U. Washington, BUPT, etc.)
- **Coupling this with an open-source community makes for a very disruptive technology for the onset of 5G**
 - What we’re building
 - ☞ Community of individual developers, academics and major industrials embracing open-source for 5G
 - What we hope to become
 - ☞ A strong voice and maybe a game-changer in the 3GPP world
 - Real impact from “the little guys” on 3GPP systems

Current vRAN Roadmap in OAI



Current Functional Entities

- OAI currently implements the following entities in openairinterface5g
 - LTE-MODEM (eNB 36.211 OFDM modulation/demodulation)
 - LTE-L1 (eNB 36.211/212/213)
 - LTE-MACRLC (eNB 36.321/322)
 - LTE-PDCP (eNB PDCP/GTPU 36.323)
 - LTE-RRC (eNB RRC/SCTP 36.331)
- Each entity comprises
 - a northbound interface (backhaul/midhaul/fronthaul and configuration)
 - a southbound interface (midhaul/fronthaul and configuration)
 - one or two management interfaces
 - Three computing nodes
 - * **Radio Cloud Center (RCC)** : multiple RRC/PDCP entities
 - * **Radio-Access Unit (RAU)**: multiple MACRLC entities with medium-latency midhaul and L1 entities with low-latency fronthaul.
 - * **Remote Radio-Unit (RRU)**: Equipment at radio site. Varying degrees of processing elements depending on fronthaul/midhaul interface.
- Each entity has a configuration which is a local file or received via the management interface
- default interface between all entities is implemented using a UDP socket. Transport is configurable via a dynamically-loadable networking device



NGFI fronthaul splits today in OAI

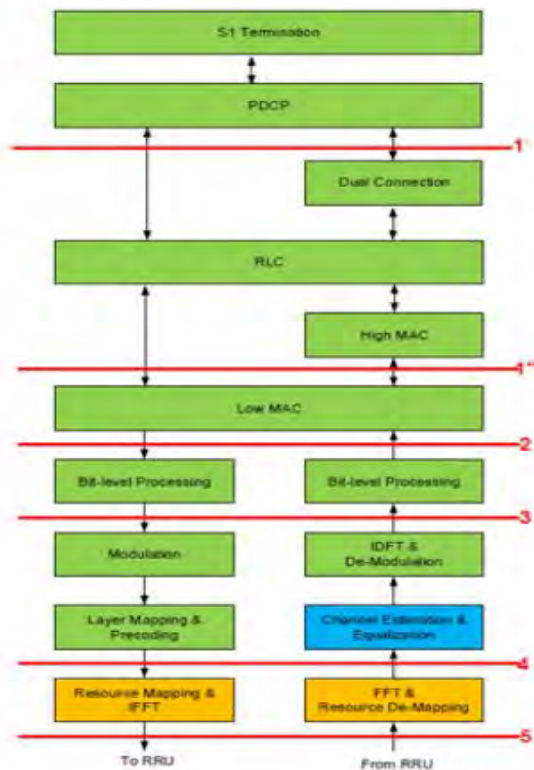
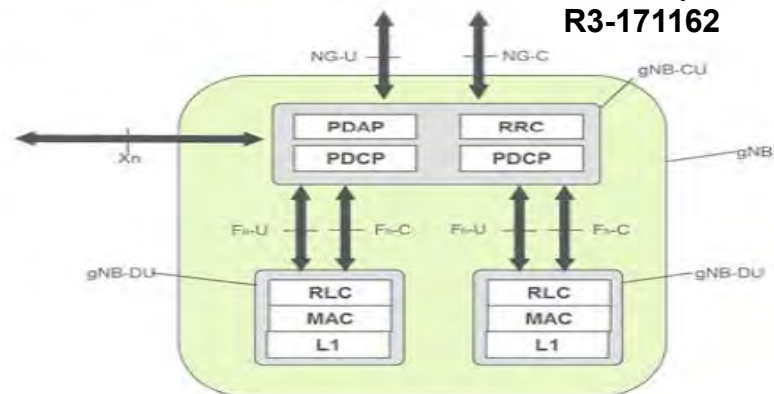


Figure 3-1: Division Plans for the RCC-RRS Interface

- Current OAI implementation (RRU/RCC) supports either
 - IF5 time-domain fronthaul (> 1 GbE required)
 - IF4.5 split (FFT) (280 Mbit/s/antenna port fronthaul 20 MHz carrier) per carrier/sector
 - Soon IF2 (NF-API)
 - IF1 for PDCP/RRC soon (3GPP Fh-C/Fh-U)

NR study item
R3-171162

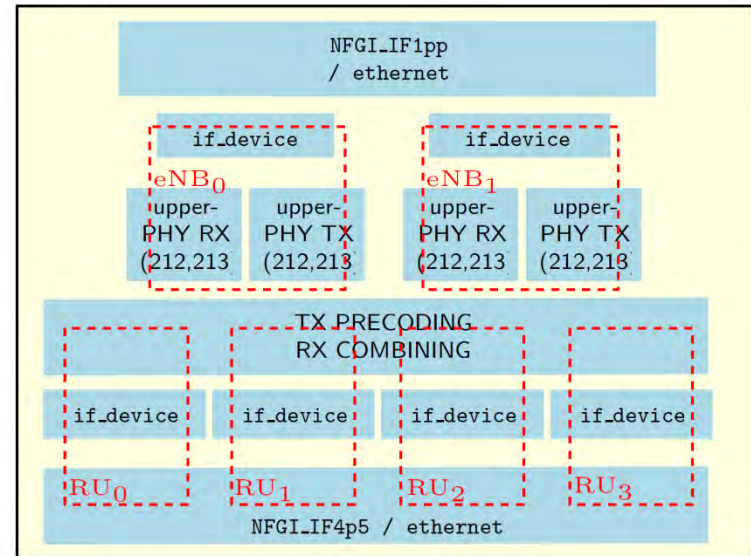


- **OAI current lower-layer functional split**

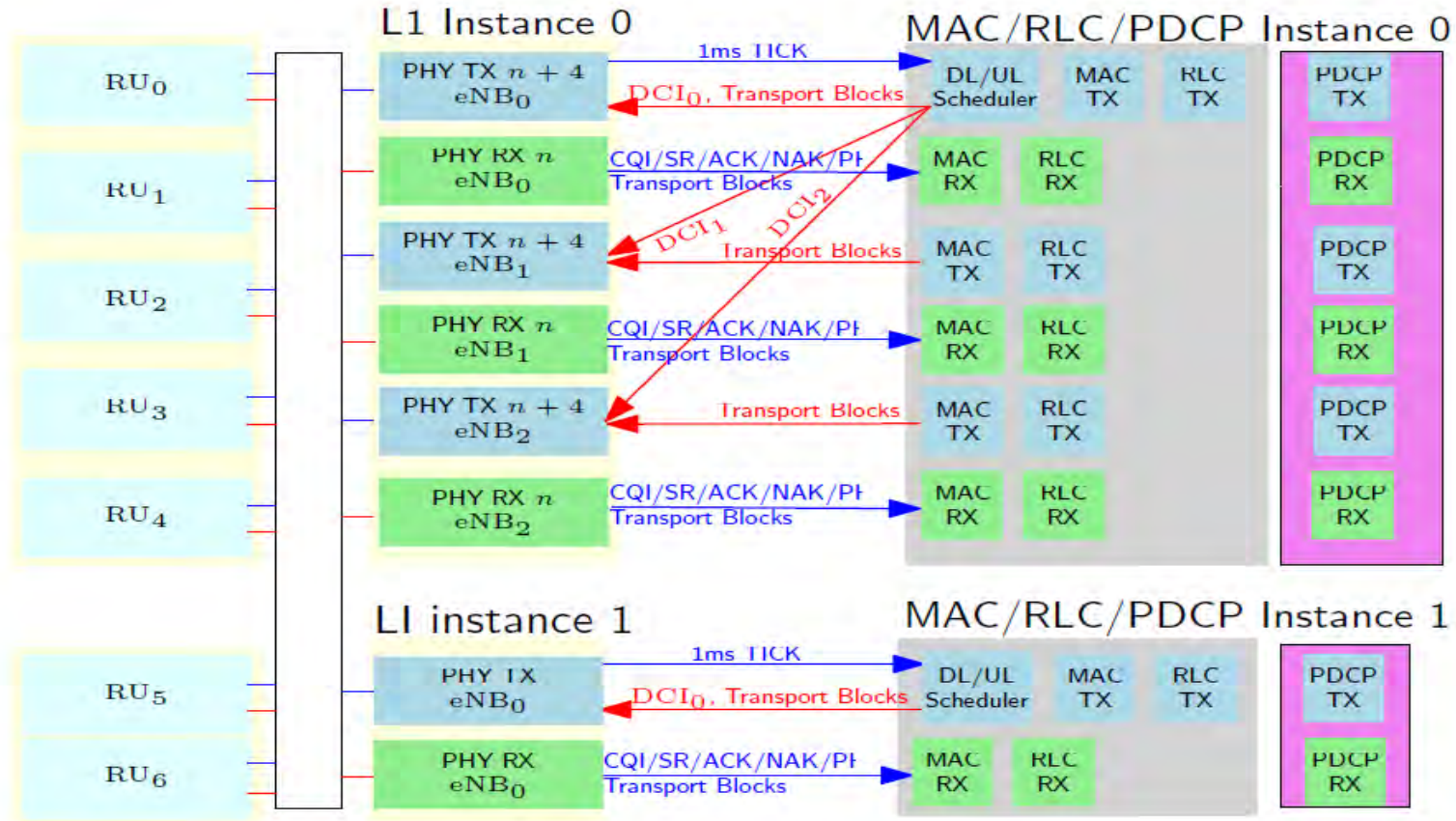
- a network of radio units (L1-low)
- a precoding function and switching function
- Regular (virtualized) eNB functions

- **Example:**

- RAU with NGFI IF1pp xhaul (MAC/PHY split) northbound,
- NGFI IF4p5 fronthaul southbound, 2 vCell logical interfaces (2 L1/L2) instances, or 1 L2 instance and 2 CCs), 4 RRU with NGFI IF4p5



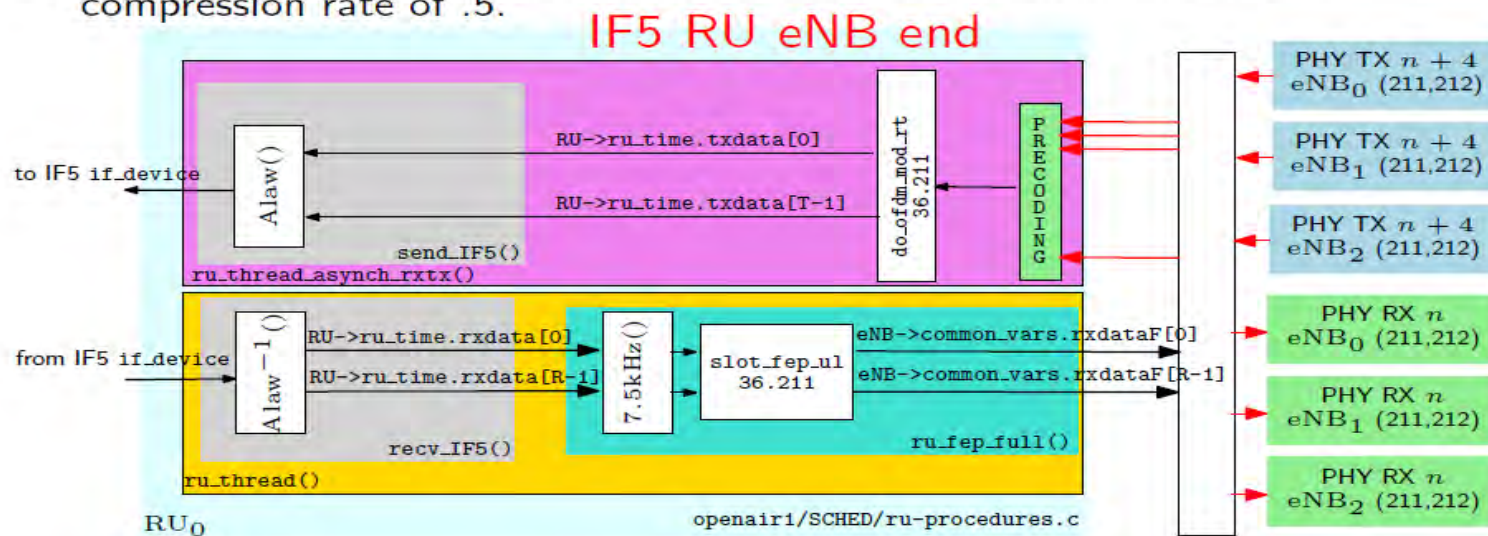
OAI RAN (CU-DU) Architecture



RADIO SEGMENT (RU)

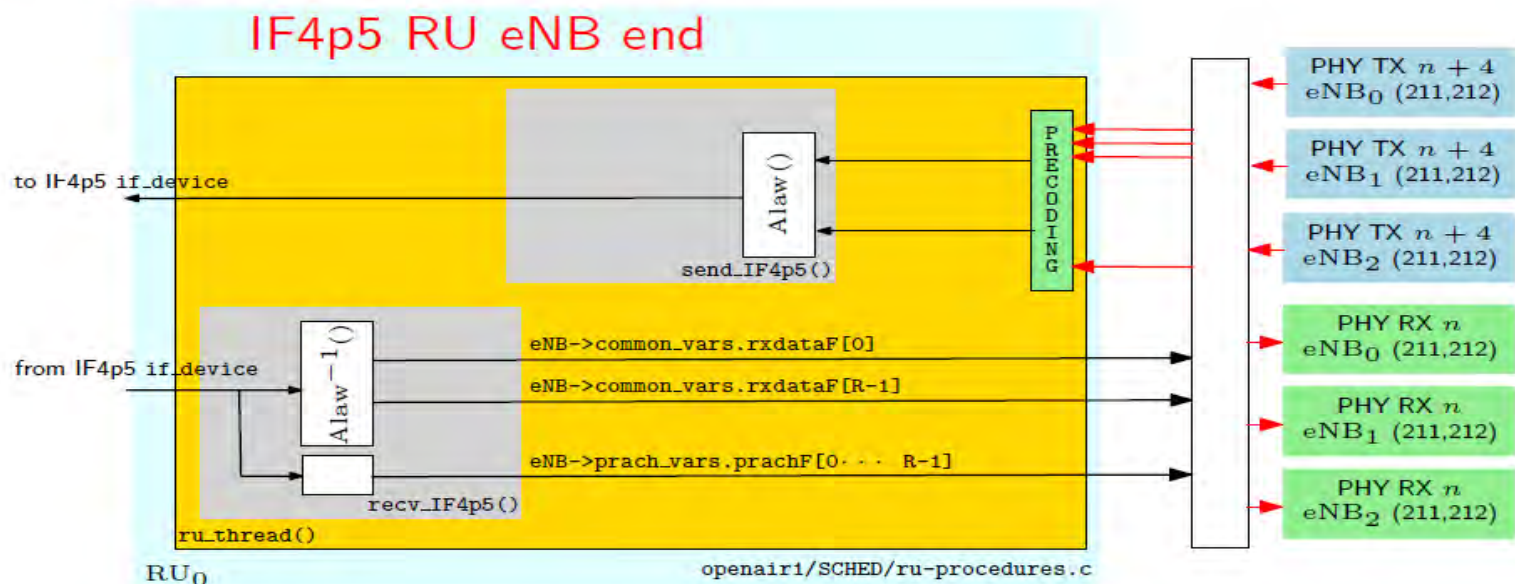
IF5 Radio Unit “Agent”

- IF5 transports packets of size equal to a subframe and corresponding to a 1ms chunk of signal in the time-domain. This is done via the functions `send_if5` and `recv_if5`, in the layer1 transport procedures (`openair1/PHY/LTE-TRANSPORT/if5_tools.c`). A timestamp is given along with the samples, corresponding to the time (in samples) of the first sample of the packet.
- each block can be compressed with A-law compression, yielding a compression rate of .5.



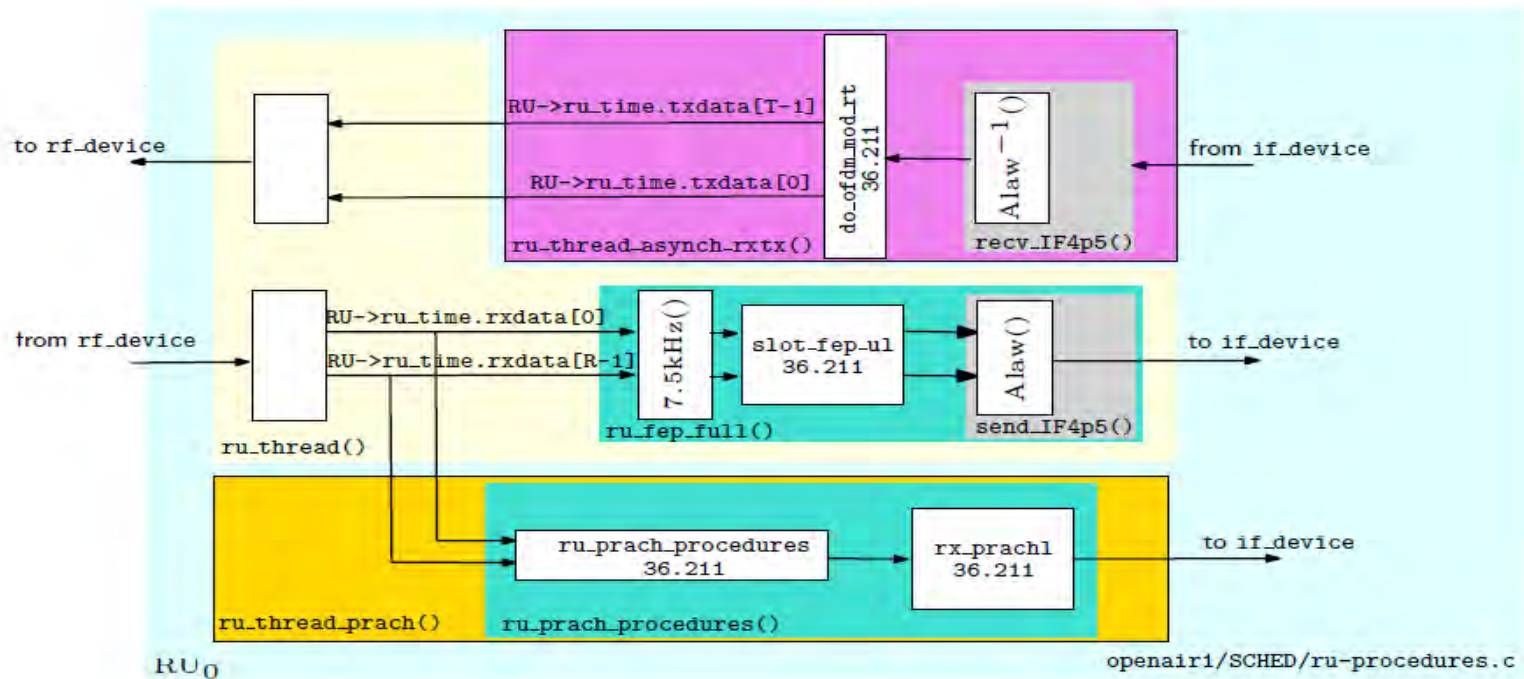
IF4p5 Radio Unit “Agent”

- IF4p5 transports packets of size equal to an OFDM symbol (for DLRE and ULRE) indexed by the symbol, subframe and frame number. This is done via the functions `send_if4p5` and `recv_if4p5`, in the layer1 transport procedures (`openair1/PHY/LTE_TRANSPORT/if4_tools.c`).
- each block are compressed with A-law compression, yielding a compression rate of .5.



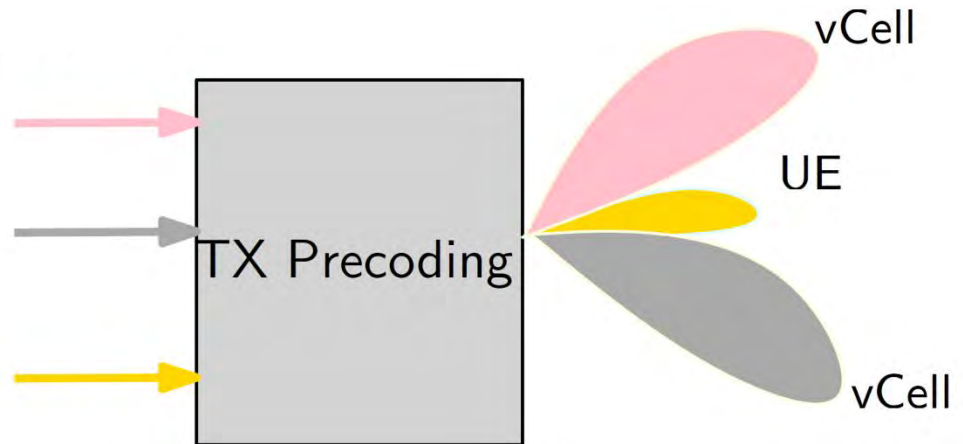
IF4p5 Remote Radio Unit

IF4p5 RU remote-end



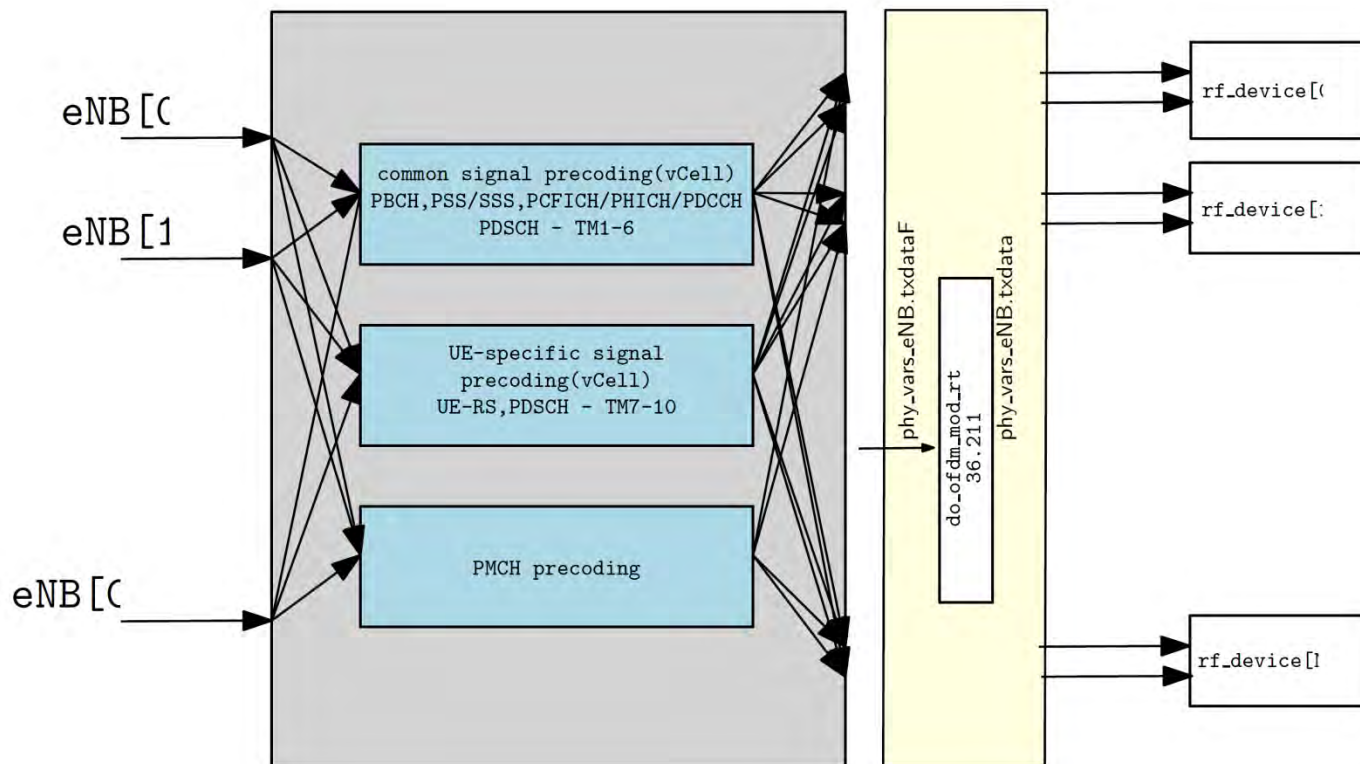
TX Precoding

- **Spatio-temporal filtering for multi-cell (vCell) and multi-user transmission.**
 - Input and output are frequency-domain signals.
- **Applicable to Rel-10/11/12/13 physical channels and Rel-8 common channels**
 - UE-specific precoding (TM7-10)
 - vCell-specific precoding (PDCCH + TM1-6) for groups of Ues
 - PMCH vCells
- **Precoding applicable to**
 - indoor DAS
 - outdoor co-localized arrays
 - ☞ Massive-MIMO
 - outdoor CoMP



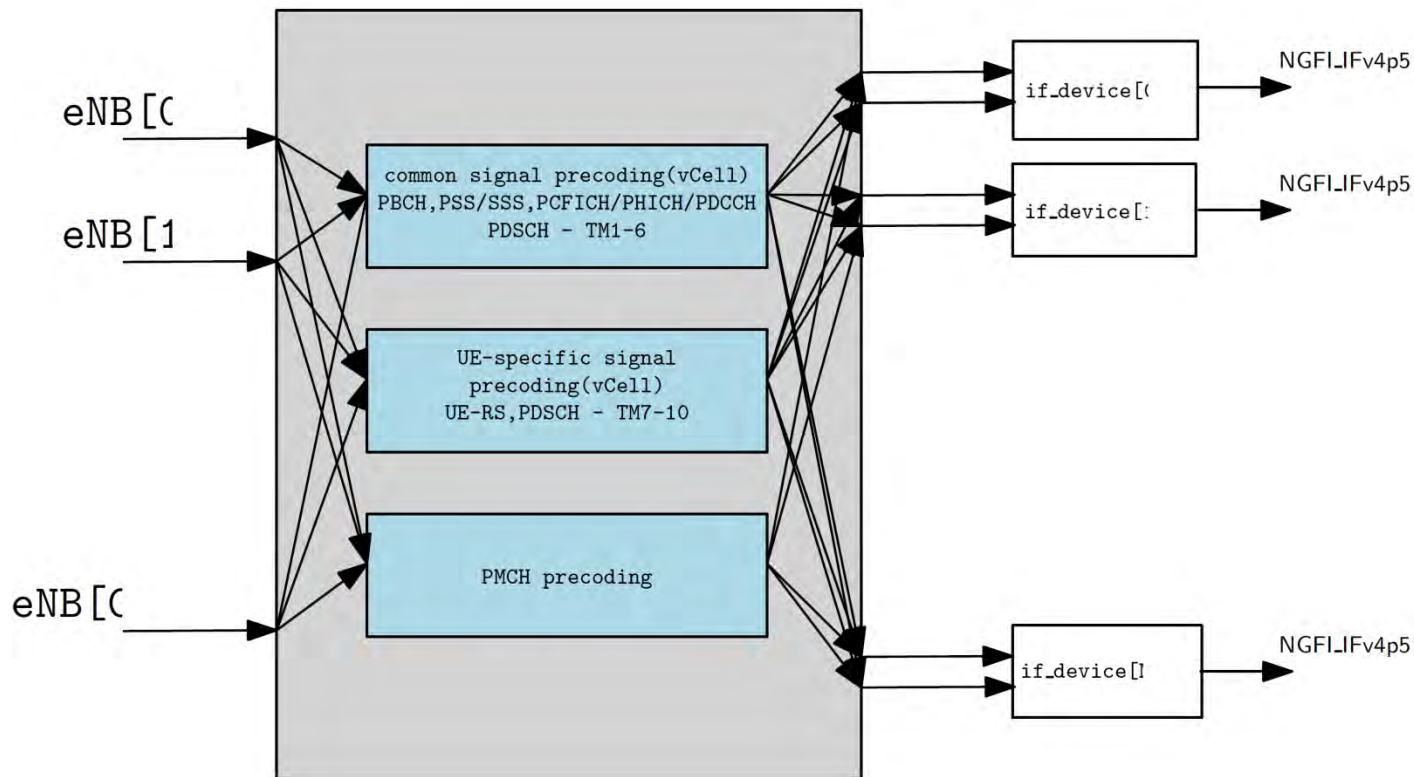
TX Precoding (to RF device)

- Use case: Massive MIMO RRS



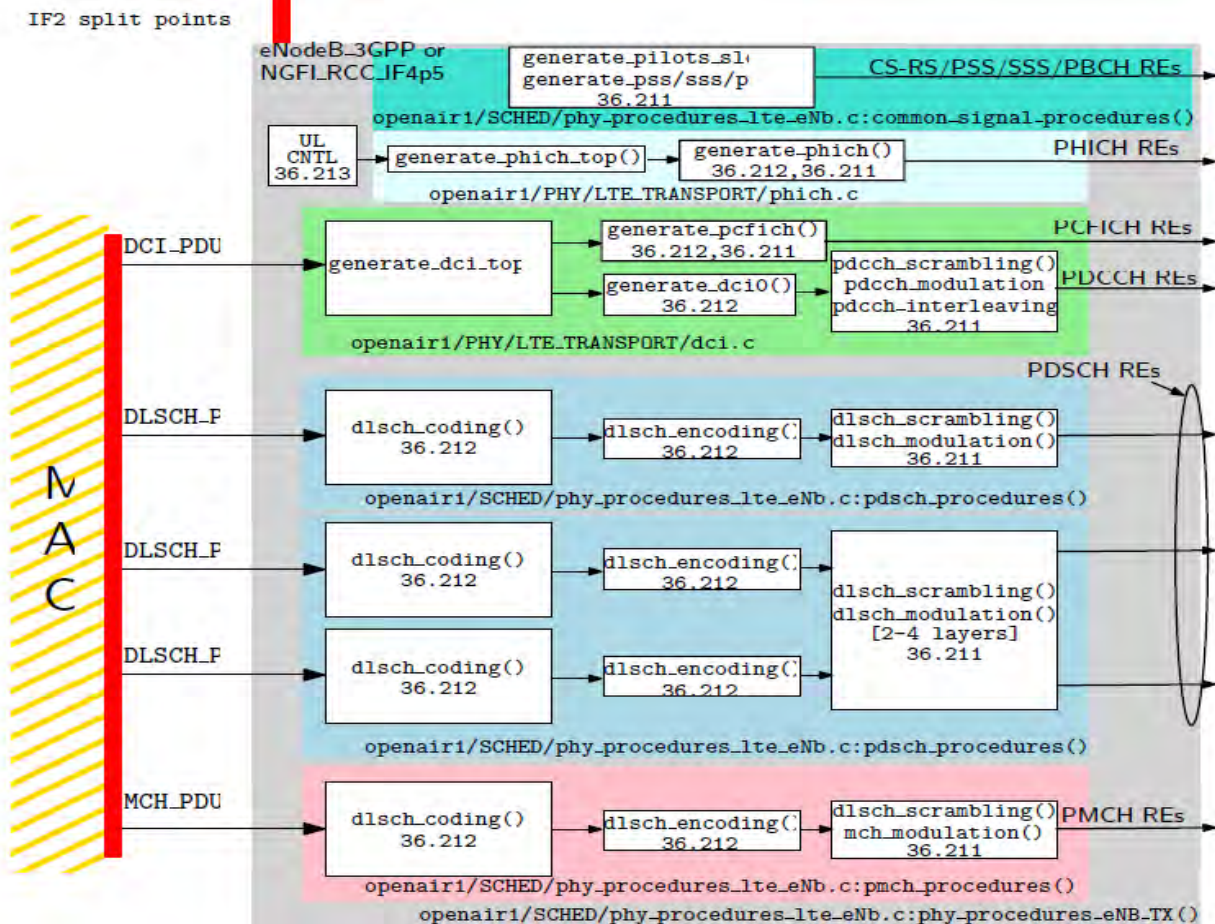
TX Precoding (to RF device, NGFI_IFv4p5)

- Use case: Indoor DAS RRS

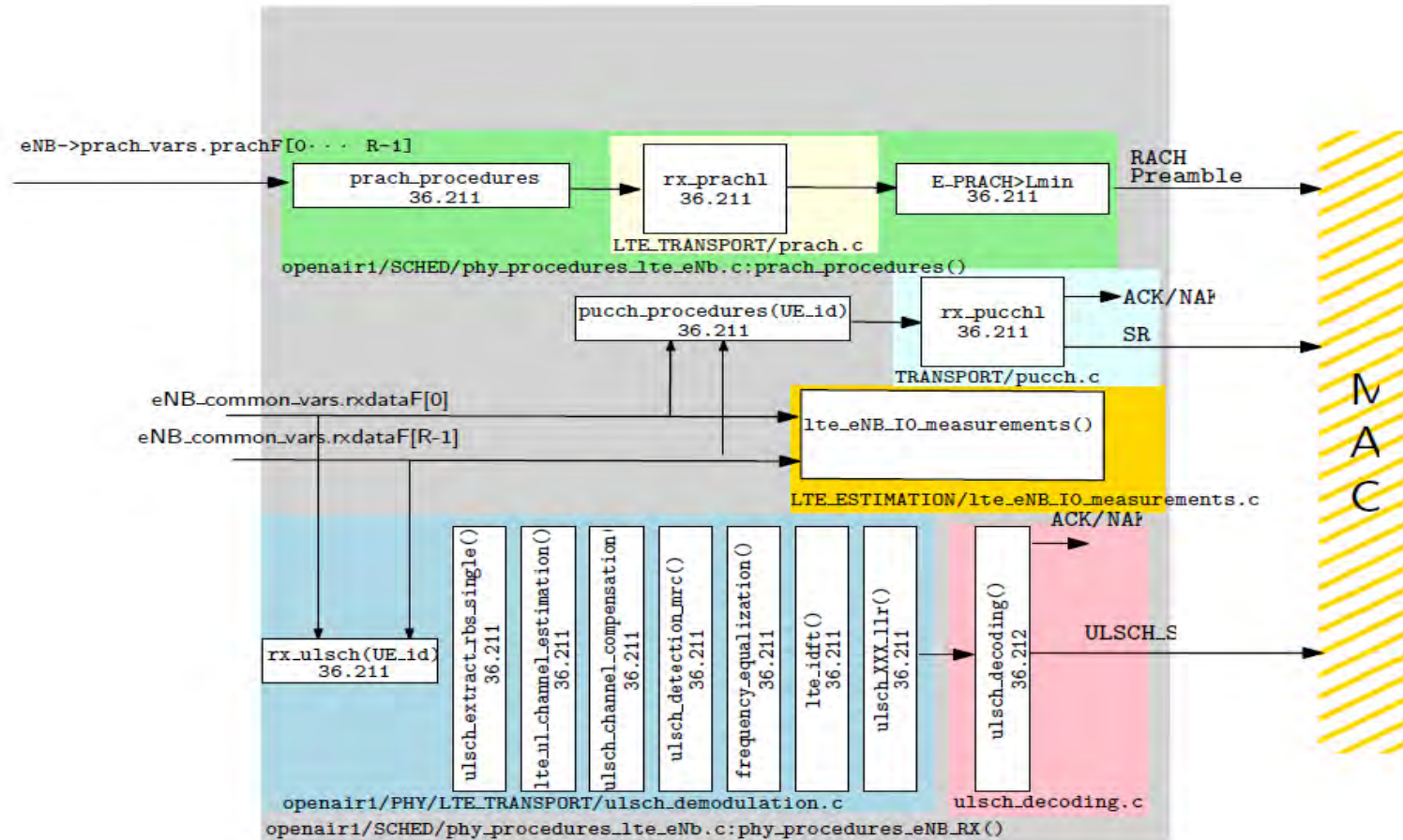


LAYER 1 HIGH SEGMENT (ENB/GNB)

Layer 1 TX per eNB instance

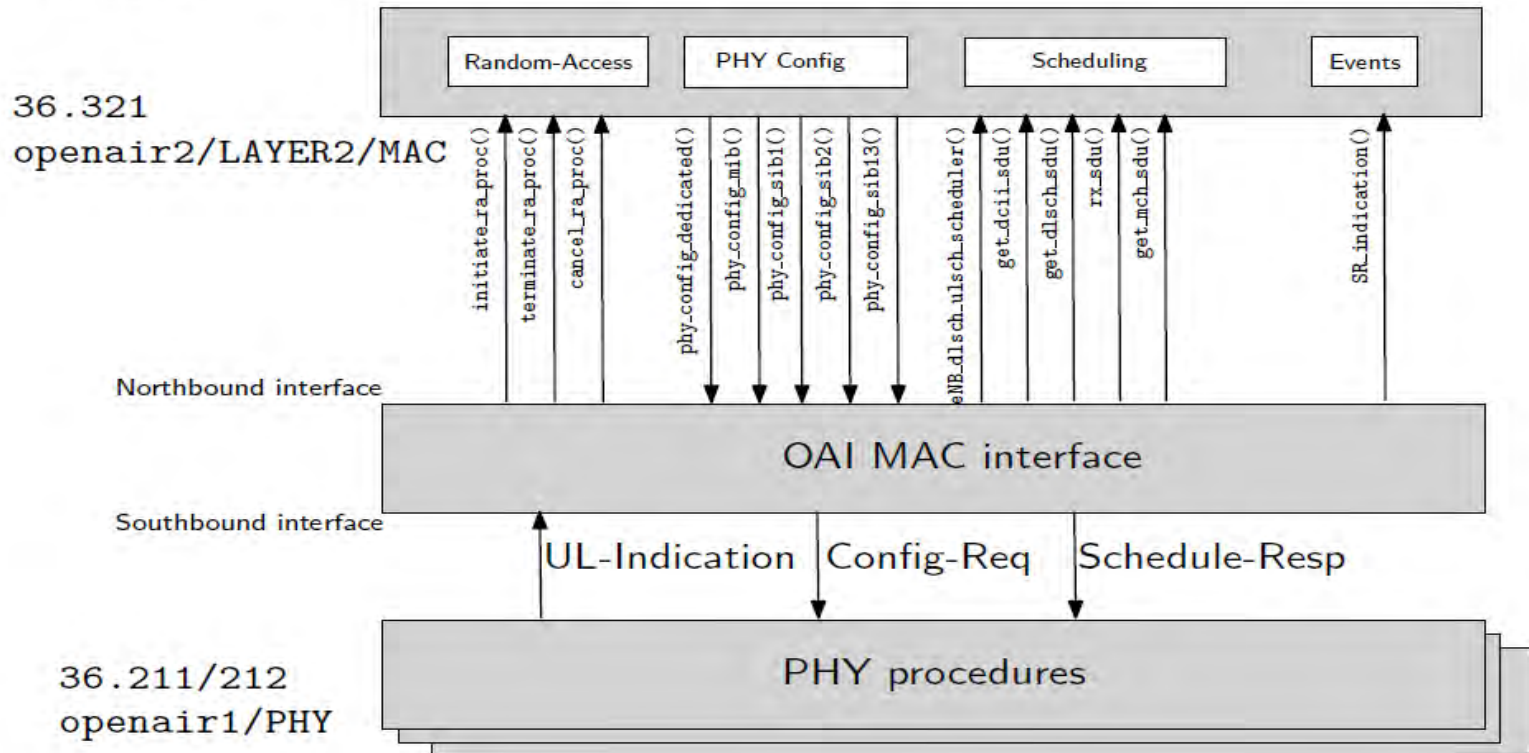


Layer 1 RX per eNB instance



OAI IF1" – MAC/PHY

- OAI IF1" is the interface between the 36.321 Medium-Access (MAC) Layer Procedures and the 36.213 Physical Layer Procedures. It links several PHY instances to one MAC instance.
- It is a configurable (dynamically loadable) module which can implement an (N)FAPI P5/P7 or a simpler interface.

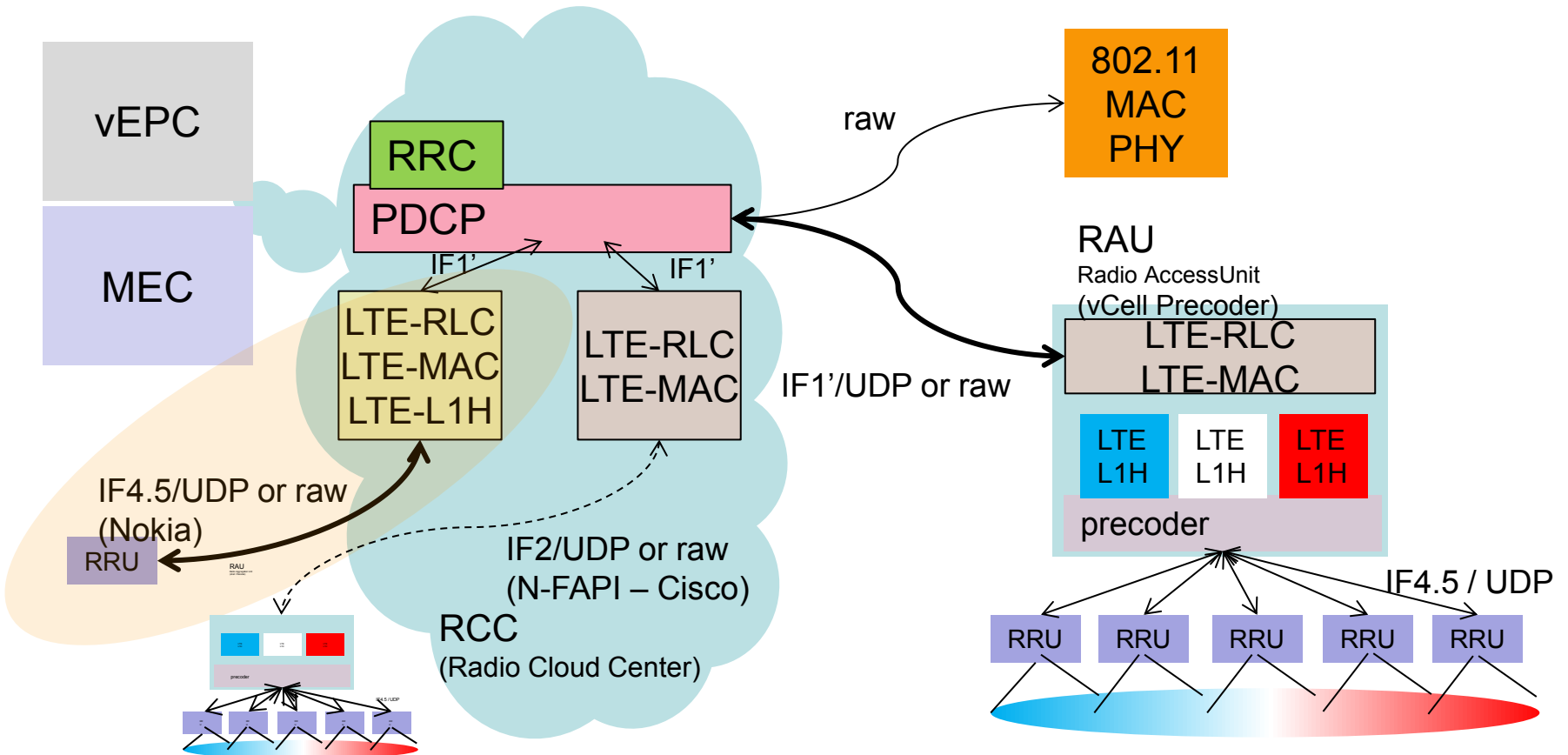


OAI IF1” – MAC/PHY

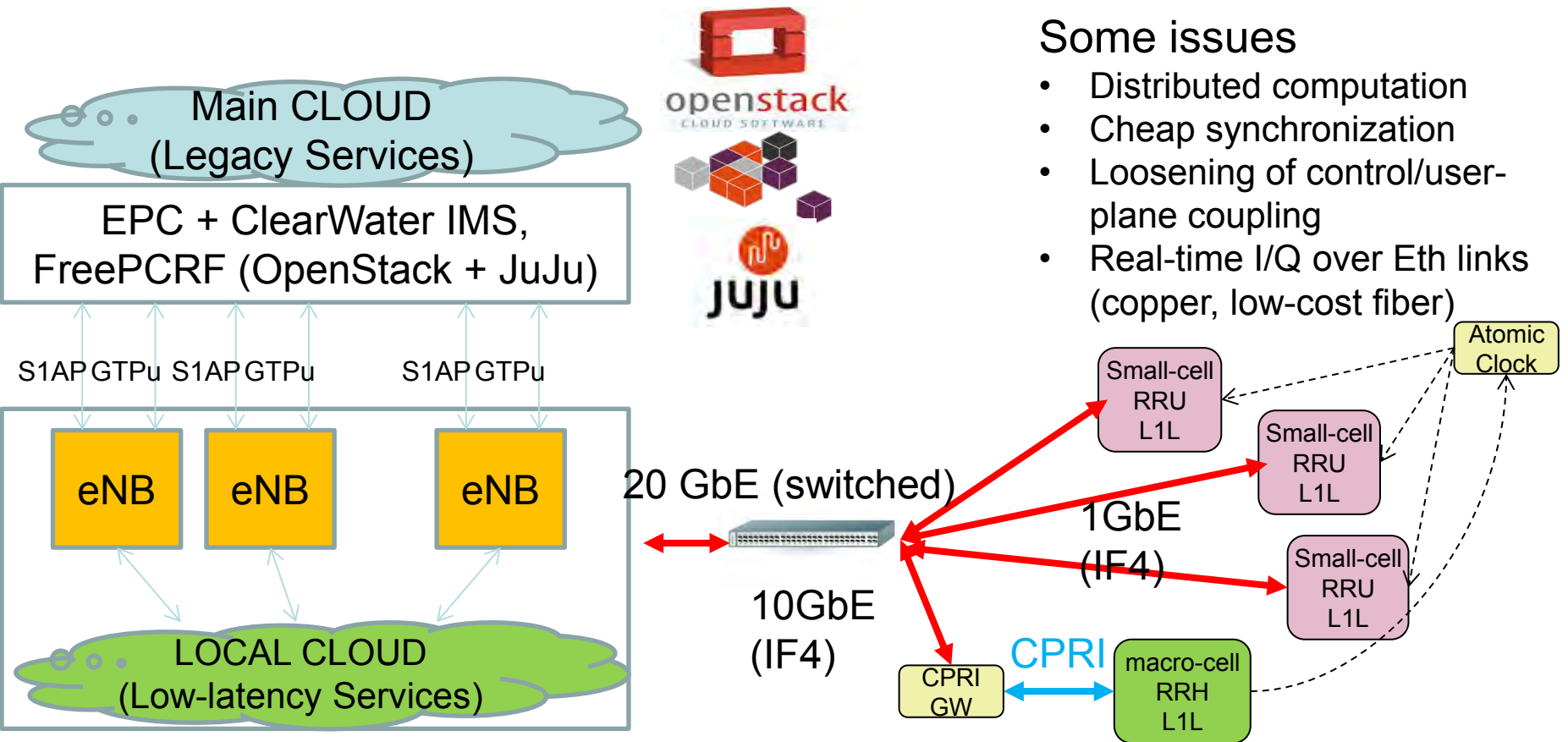
- The PHY end uses three basic messages
 - CONFIG_REQ: this provides the cell configuration and UE-specific configuration to the PHY instances. This comprises the following FAPI P5/P7 messages
 1. CONFIG.request
 2. UE_CONFIG.request
 - UL_INDICATION This is an uplink indication that sends all UL information received in one TTI, including PRACH, if available. It also provides the subframe indication for the DL scheduler. It maps to the following FAPI P7 messages
 1. SUBFRAME.indication
 2. HARQ.indication
 3. CRC.indication
 4. RX_ULSCH.indication
 5. RX_SR.indication
 6. RX_CQI.indication
 7. RACH.indication
 8. SRS.indication
 - SCHEDULE_REQUEST This message contains the scheduling response information and comprises the following FAPI P7 messages
 1. DL_CONFIG.request
 2. UL_CONFIG.request
 3. TX.request
 4. HI_DCIO.request
- The module is registered both by PHY and MAC and can implement different types of transport (NFAPI, function call, FAPI over UDP, etc.). During registration, function pointers for the different messages are provided for the module to interact with either PHY or MAC or both if they are executing in the same machine. Note that for a networked implementation (e.g. NFAPI), there are north and south components running in different machines.

EXPERIMENTAL REAL-TIME DEPLOYMENT

Splits under construction in OAI Community



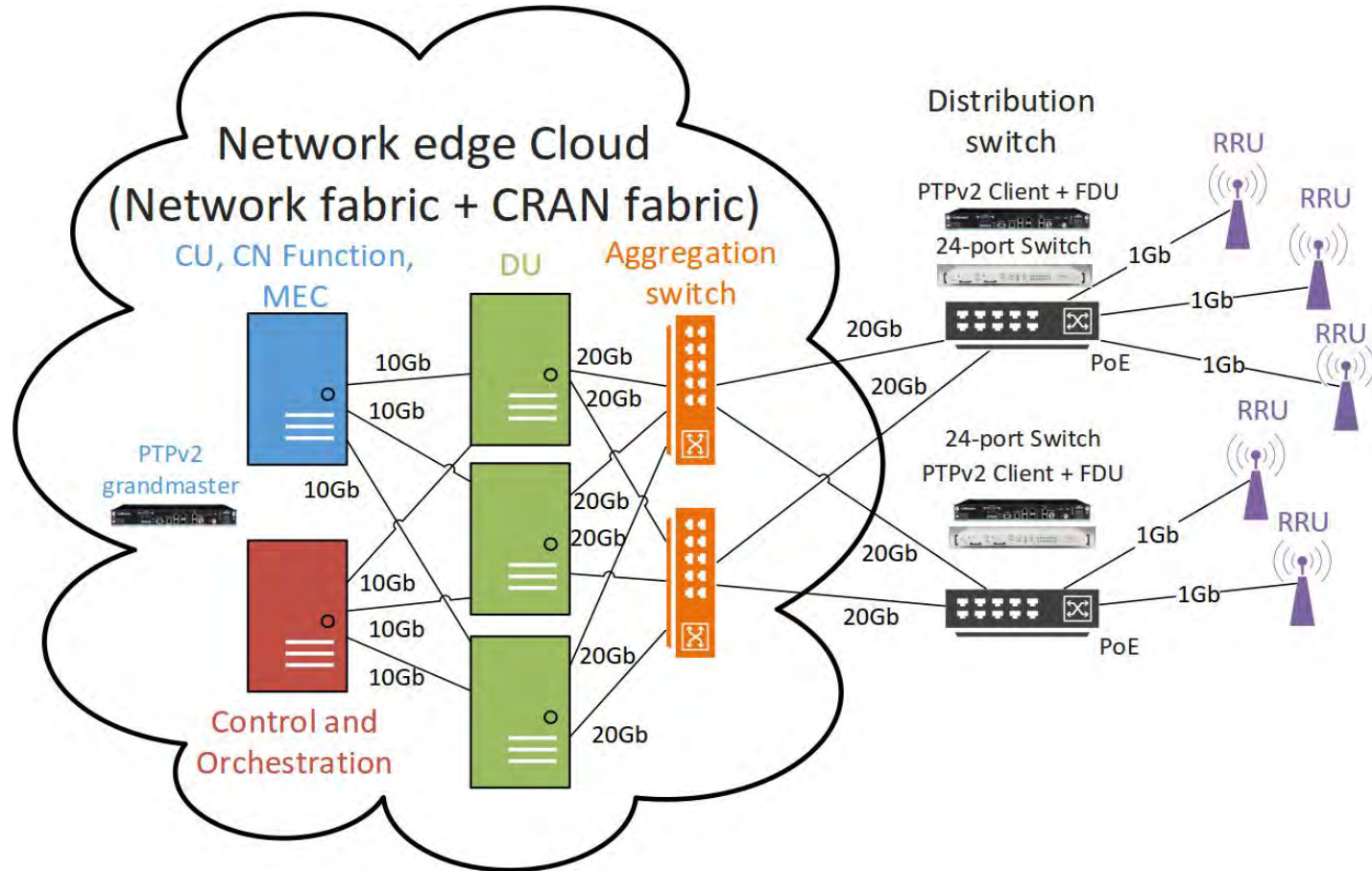
OAI Playground @ EURECOM



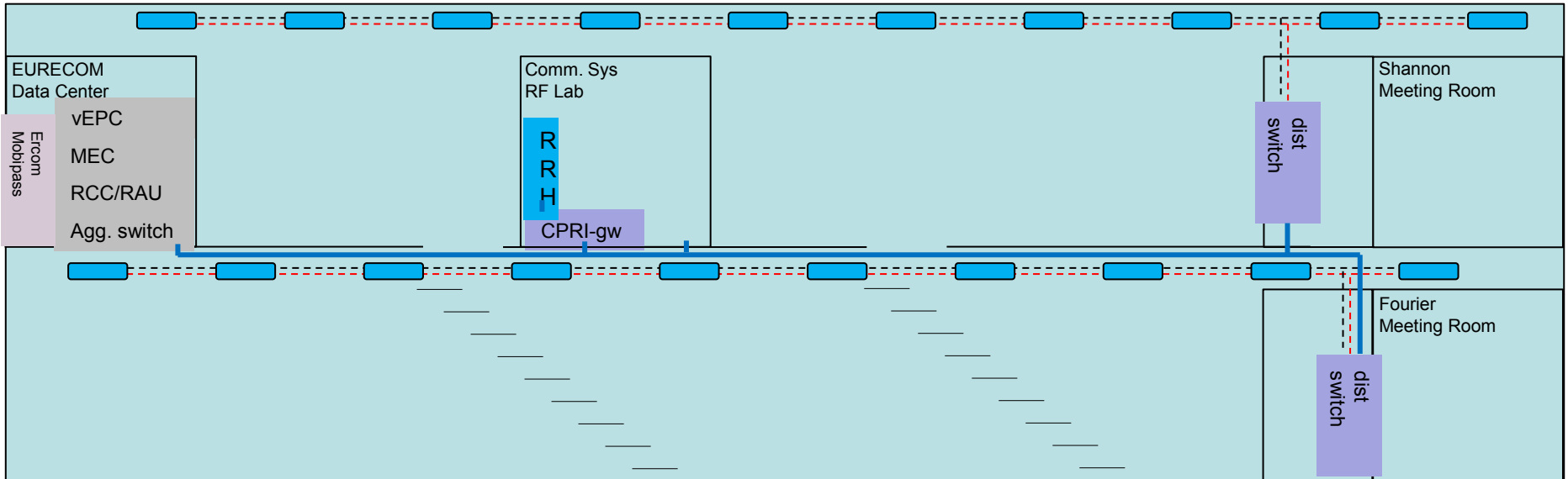
Some issues

- Distributed computation
- Cheap synchronization
- Loosening of control/user-plane coupling
- Real-time I/Q over Eth links (copper, low-cost fiber)

OAI Playground: Components



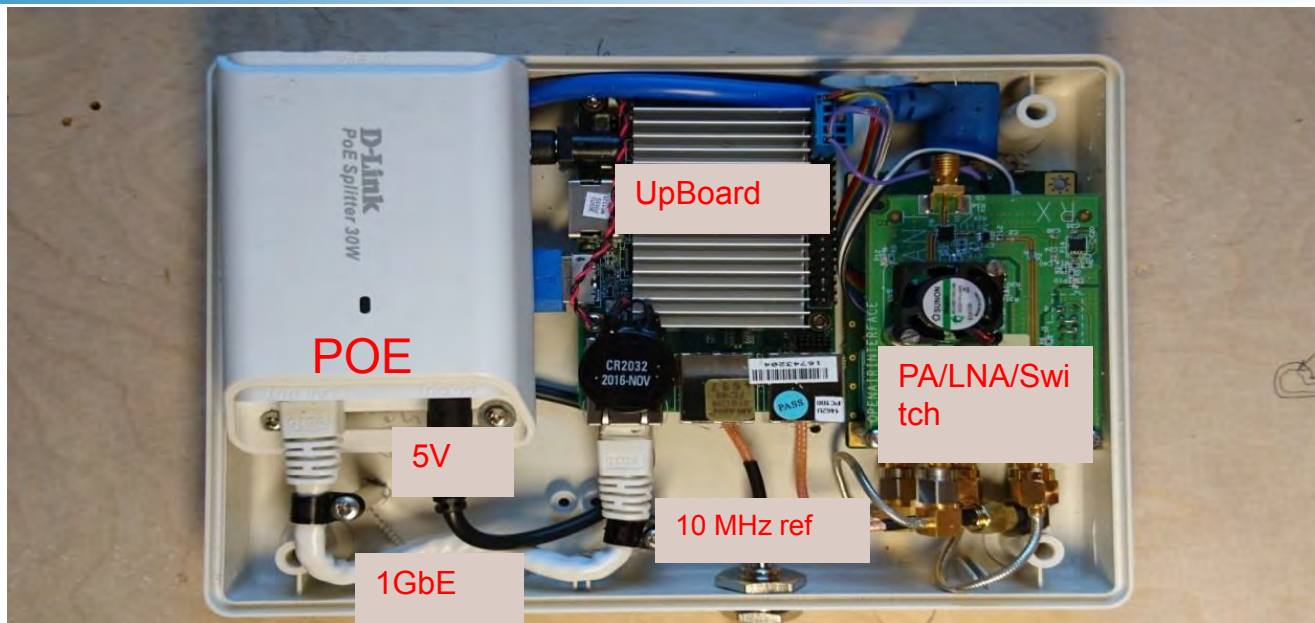
OAI Playground: Indoor Segment



OAI Playground: synchronization

- **PTPv2**
 - grandmaster in RAU
 - clients in distribution switches to regenerate 10 MHz (PPS if needed) from GPS source behind RAU
- **Distribution of 10 MHz/PPS to RRU**
- **Frame synch**
 - Over-the-air between RRU (TDD)
 - via PPS distribution
- **Note: in a commercial solution, PTPv2 would go all the way to the RRU and clock synchronization would be rederived in each RRU**
 - No off-the-shelf solution for everyday users

OAI Playground: RRU – Bill of Materials



■ SISO (20 MHz, 1GbE Fronthaul)

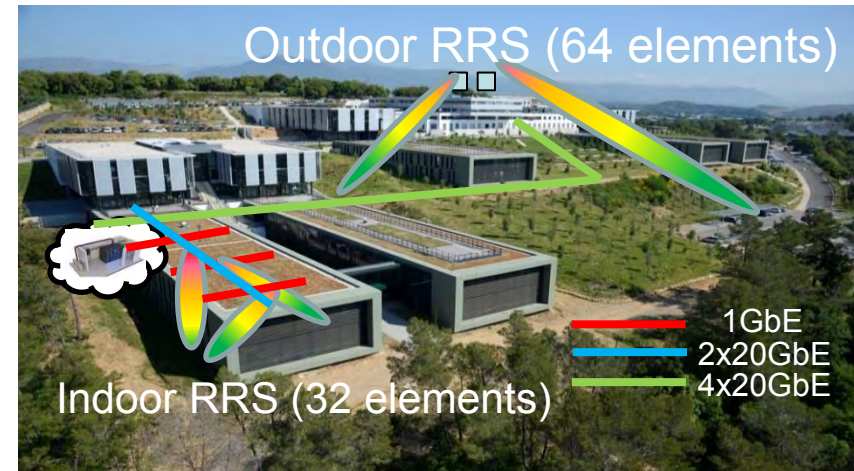
- UPBoard (100\$)
- USRPB200-mini (500\$ in quantities)
- PA/LNA/Switch (100\$)
- PoE+ module (50\$) => 750\$

■ 2x2 (20 MHz, 2 GbE Fronthaul)

- UpBoard2 (200\$)
- LimeSDR (300\$ in quantities)
- PA/LNA/Switch (200\$)
- PoE+ modules (100\$) => 800\$

Extension on SophiaTech Campus Deployment

- **Collaboration around OAI technologies on SophiaTech Campus**
 - EURECOM (Open5GLab) + INRIA (R2Lab)
- **Showcase**
 - Centralized RAN and MEC
 - ☞ Antenna processing
 - ☞ RAN slicing
 - ☞ Indoor-outdoor coordination
 - ☞ RAN/CN Orchestration
 - ☞ Multi-tenancy
 - Optical/wireless technologies
 - 2.6/3.5 GHz TDD
 - eMBB and eMTC
 - 802.11 convergence



Part V – Perspectives and Conclusion

3G

4G

5G

GSM

Data Center Technologies for RAN

- **5G and beyond is not only New Radio and verticals**
- **It is also evolution in computing for wireless networks**
 - Central offices becoming data-centers (see CORD / M-CORD projects in USA as an example)
 - High-performance fronthaul networks for distributed computing
 - ☞ Fixed network to support advanced radio
 - Centralized computing and storage using
 - ☞ More general-purpose equipment (Intel servers)
 - ☞ More and more software technologies from cloud-computing (NFV,SDN,MEC, etc.) jointly with radio signal processing
 - Applicable to lesser extent for existing and evolving 4G radio
- Fusion of Information and Cellular technologies

Emergence of Open-Source

- **Increased interest in understanding (managing?) the role of open-source communities by**
 - **ITU Focus Group on IMT-2020 (SG13) : mandate completed**
 - ☞ the standardization expert group responsible for future networks, cloud computing and network aspects of mobile communications. new mandate includes the exploration of demonstrations or prototyping with other groups, notably the open-source community
 - Extension for ITU-T Focus Group on network aspects of 5G
 - Group instructed to work with open source communities
 - “Influencing and taking advantage of their work” part of its mandate
 - **ETSI (ETSI Summit on Standardization and Open Source, Nov. 17 2015)**
 - **NGMN (Joint ITU-NGMN Alliance Workshop “Open Source and Standards for 5G”, May 26th 2016)**
- **Main issue: patent-pool licensing**

New Players in Telecom

- **Facebook and Google are quickly entering the datacenter Telco space**
 - Value-chain of Telecom is under siege and may become very different because of this.
 - <https://telecominfraproject.com/>
 - ☞ Low-cost equipment for rural areas
 - ☞ Federating open-source developers
- **Google**
 - implication in openCORD

NFV/SDN/MEC

■ Benefits of NFV/SDN/MEC

- Making the Telecom Network an interconnection of many “Apps”
- Tailored deployments for vertical industries
- Commodity servers

■ Challenges for NFV

- Real-time processing in access-layer
 - ☞ 1ms for 4G
 - ☞ 250μs for 5G, but possibilities for higher latency too
- Software licenses

■ Challenges for SDN/MEC

- Diving deeper into the access-layer using SDN/slicing concepts for fine-grain control of radio network
 - ☞ MAC scheduling
 - ☞ Dynamic radio network topology (virtual cells)
 - ☞ Multi-RAT on mutualized hardware (4G, 5G, WIFI, LWA)
 - ☞ Multi-tenancy on mutualized hardware
 - ☞ Private/public traffic on mutualized hardware
- Joint control-plane and data-plane programmability and support of vertical
- managing time-scales (latency critical vs. delay-tolerant)

A note on real-time for SDN/NFV

- **4G L1/L2 is possible in real-time on pure x86-64 now (Xeon v3/v4)**
 - Even low-latency kernel is ok, 100 μ s jitter
 - Deadline scheduler ($\geq 3.14.10$ kernel) better, tens of μ s
 - 4G L1 feasible even on kvm/docker, but to be avoided, better as PNF with RTOS
 - 4G L2 no real issue, can be virtualised easily
- **5G will be more challenging**
 - Process scheduling requiring an order of magnitude better determinism
 - Control of HW accelerators coupled with x86/ARM GPP in real-time processing chain
 - Data plane physical networking (10-100 Gbit/s), CRAN fabric (interprocessor communications)
- **Real-time controllers will also be necessary for MAC scheduling and more advanced PNF control (e.g. virtual cell formation, CoMP)**

Targets for Academic Experimentation

- **Modern Software Technologies are new to telecommunications**
 - Different design methodologies
 - Open-source vs. standards approach
 - Teaching “modern” telecommunications
 - ☞ CS people learning signal processing
 - ☞ EE people learning Software Development Methodologies
- **Use open-source communities coupled with open hardware (e.g. USRPs, LimeSDR, etc.)**
 - Cross community research and teaching
 - Large-scale experimental facilities to influence 3GPP evolution
 - ☞ Parallel to 3GPP process, communities including universities and research centers

Q&A