LTE Networks

An overview on LTE and on ns-3 LTE simulations

Michele Polese, Andrea Zanella

Department of Information Engineering, University of Padova, Italy

e-mail: {polesemi, zanella}@dei.unipd.it

Telecommunication Networks 2016/2017

November 9th, 2016

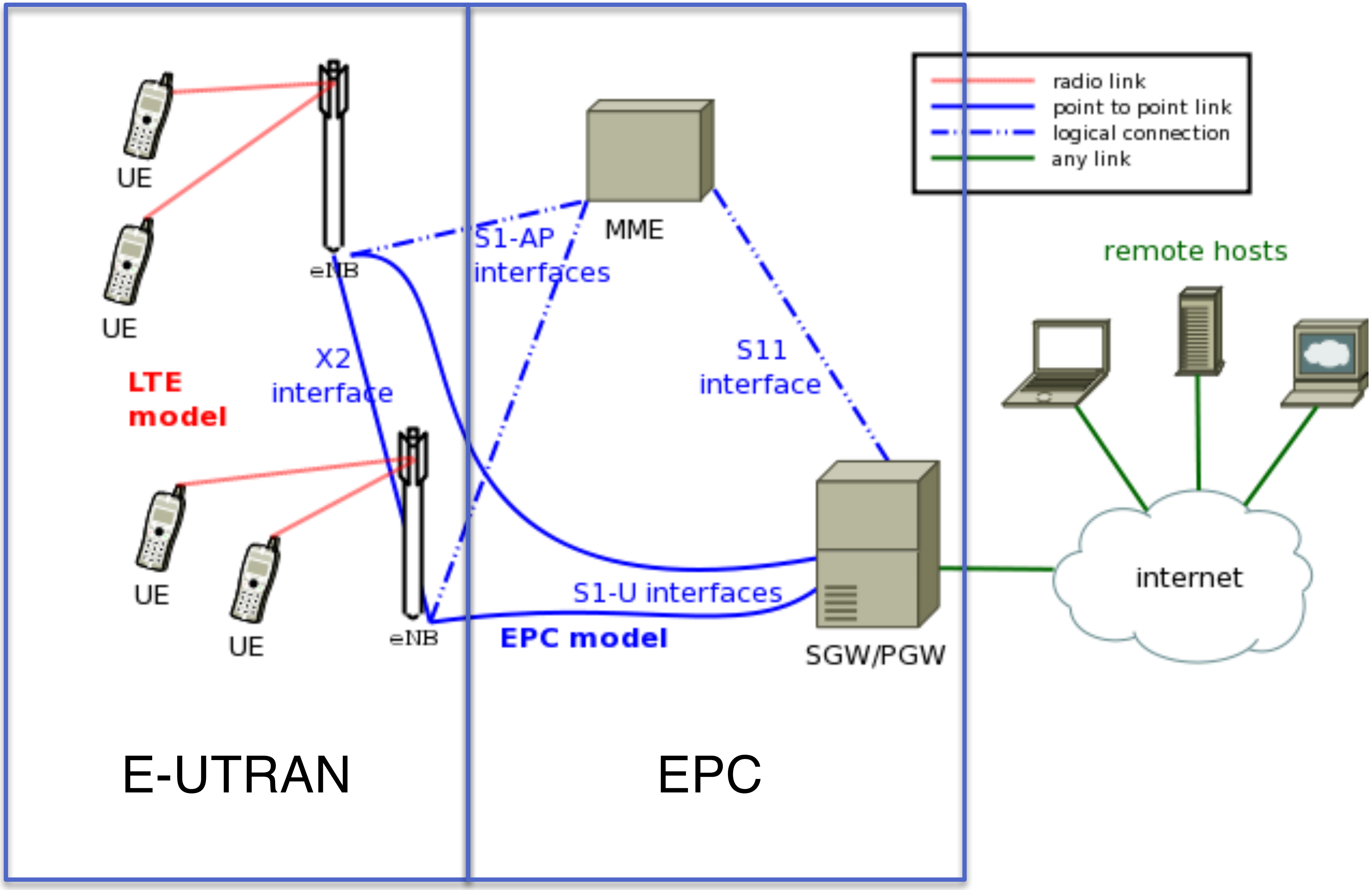
DEPARTMENT OF INFORMATION ENGINEERING

UNIVERSITY OF PADOVA

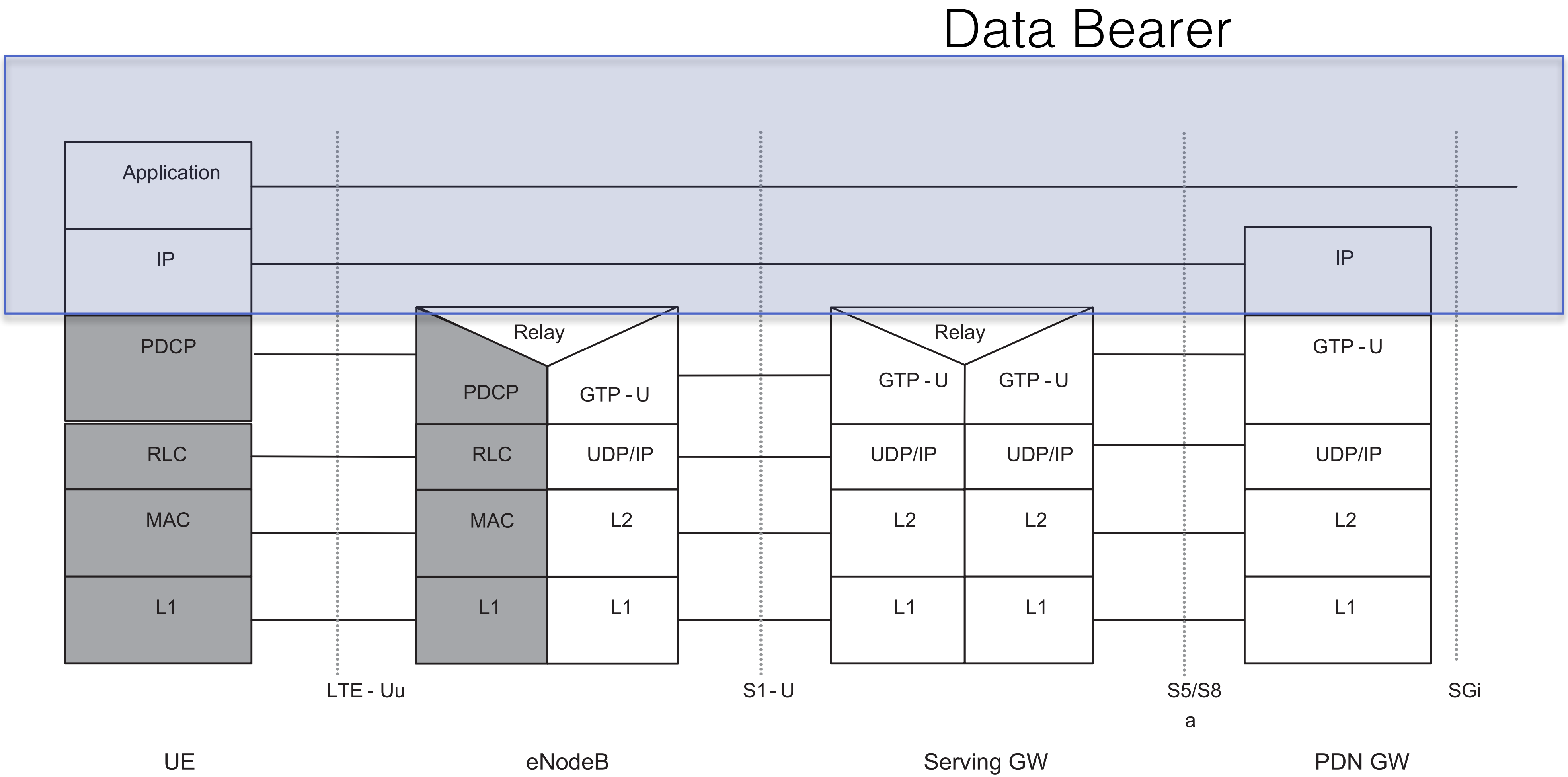
Outline

- LTE Networks: an overview
  - RAN
  - Core Network
- ns-3 LTE module
- Random Access in LTE
- LENA+ and RACH modeling

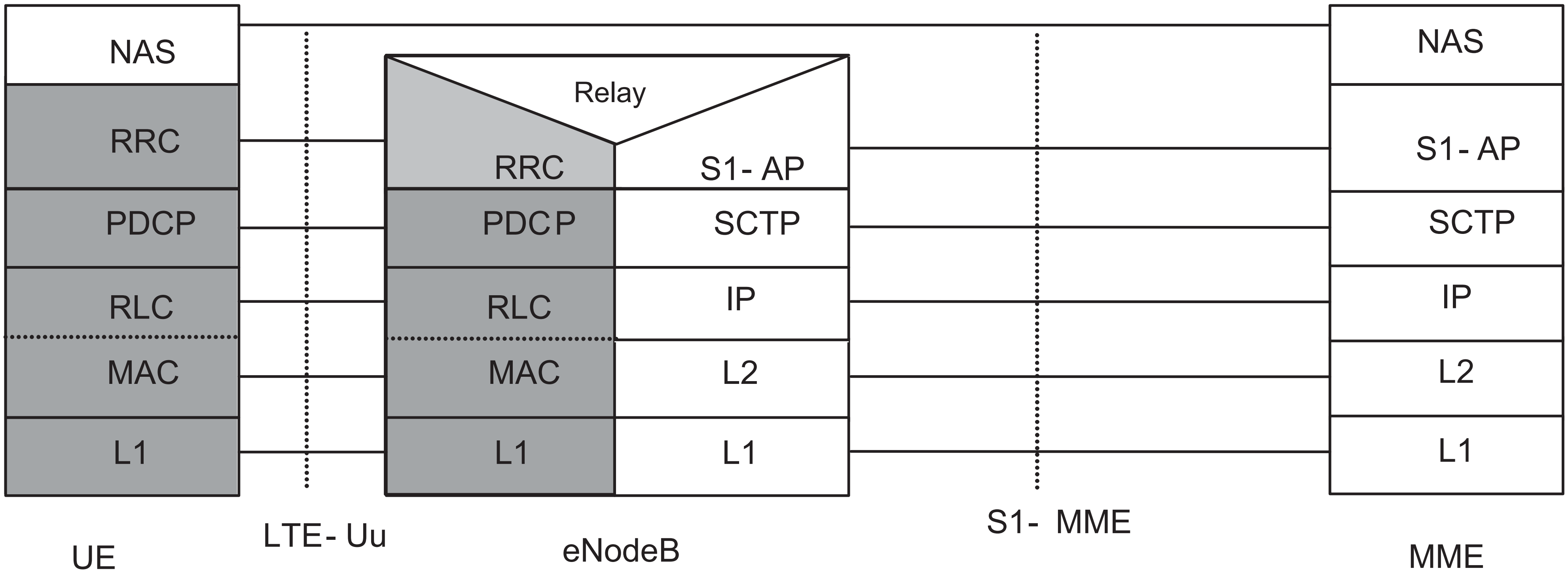
LTE Networks



LTE Protocol Stack: User Plane



LTE Protocol Stack: Control Plane

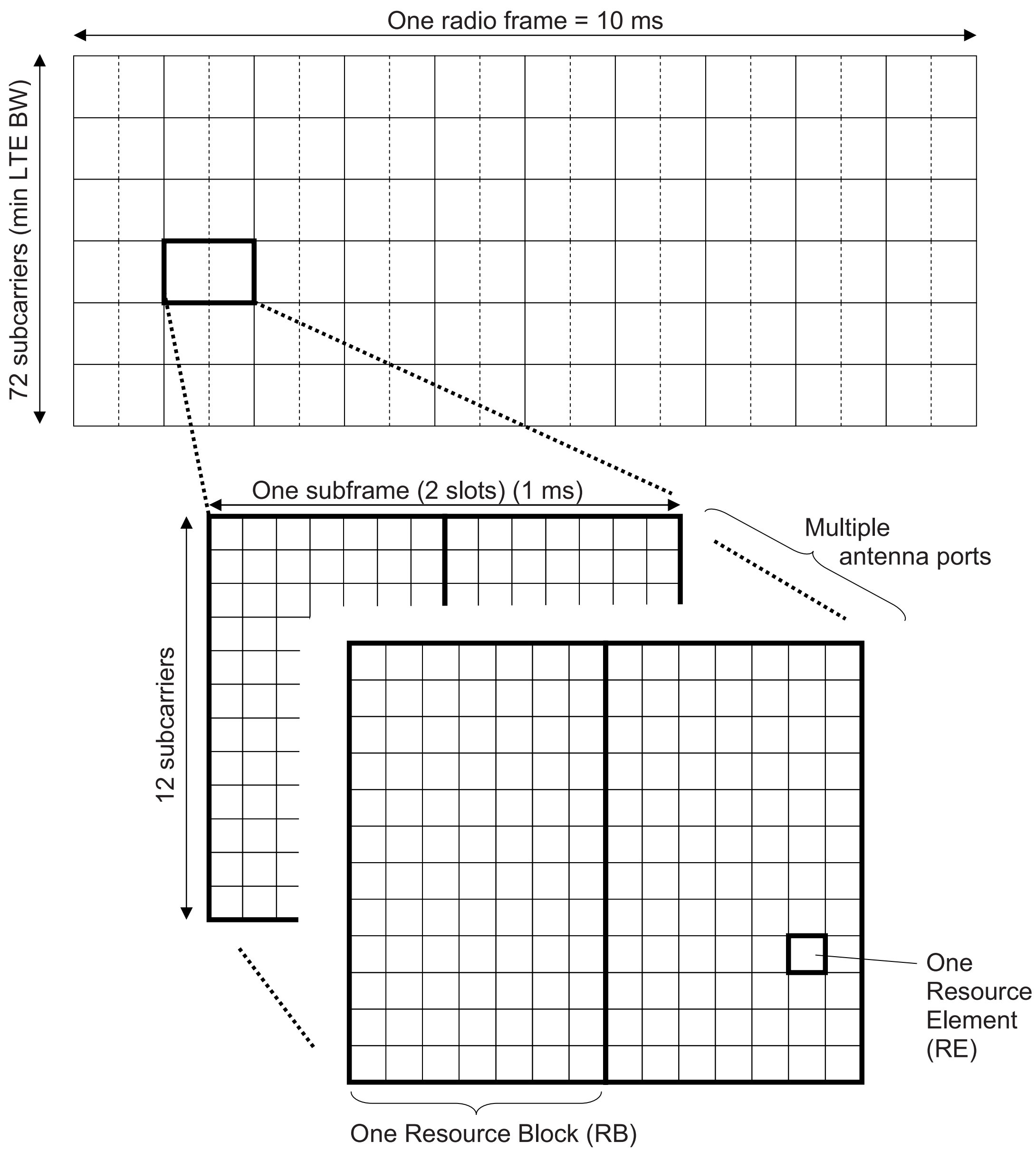


Physical Layer

- FDD
  - 2 different frequencies for DL and UL
  - Radio frame of 10 ms, with subframes (TTI) of 1 ms
- TDD
  - Same freq for DL and UL
  - TTI are either UL or DL

Physical Layer

FDD frame



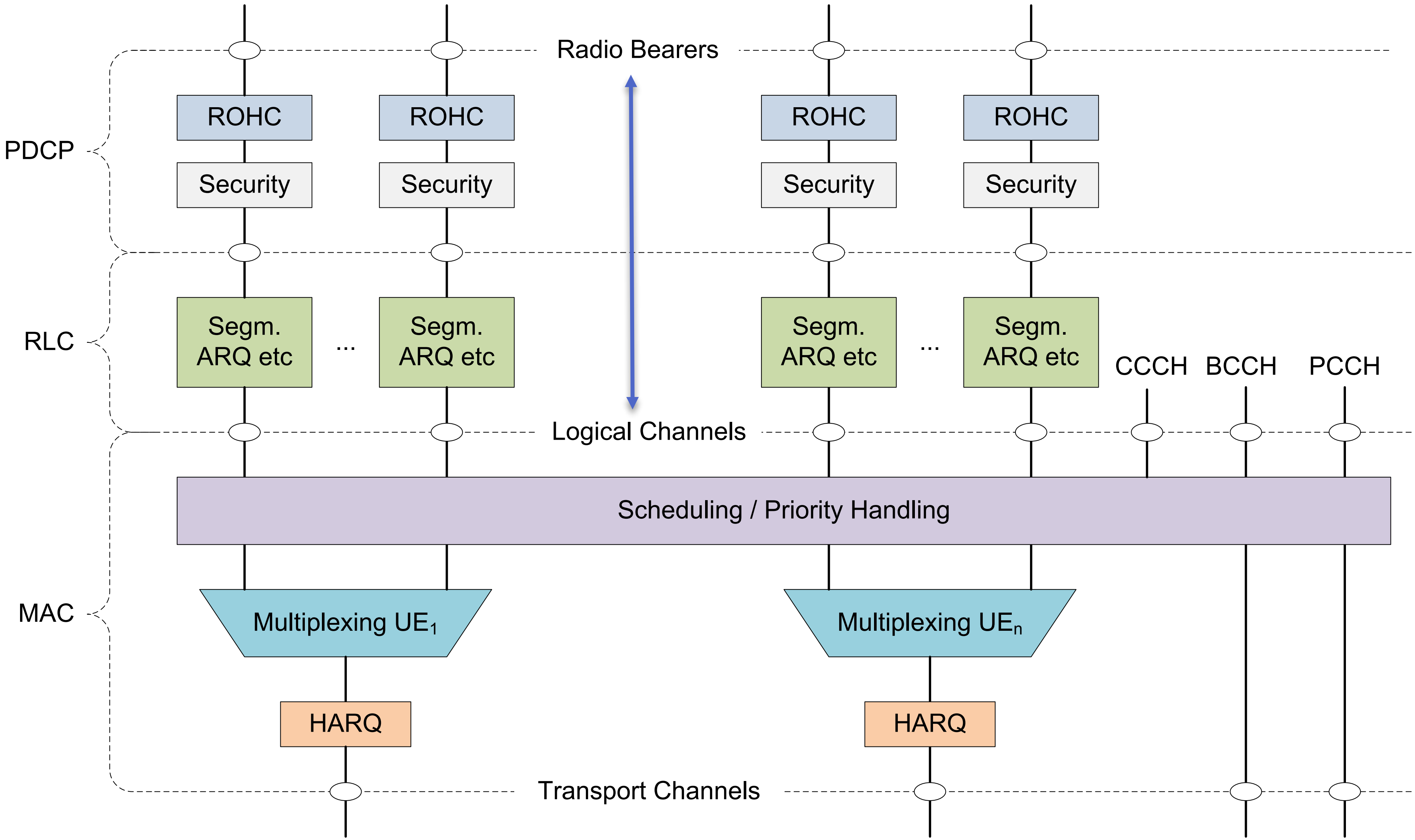
Physical Layer

- Scalable bandwidth (up to 20 MHz, 40 MHz in LTE-A)
- Different carrier frequencies
  - In Italy, operators use 900 MHz, 1.8 GHz and 2.4 GHz
- Peak spectral efficiency of 16.3 bit/s/Hz (30 for LTE-A)

MAC Layer

- Dynamically scheduled transmissions
- Hybrid ARQ
- Adaptive Modulation and Coding
- Random Access (more on this later)
- Mapping between Logical Channels and Transport Channels

MAC Layer - Channels



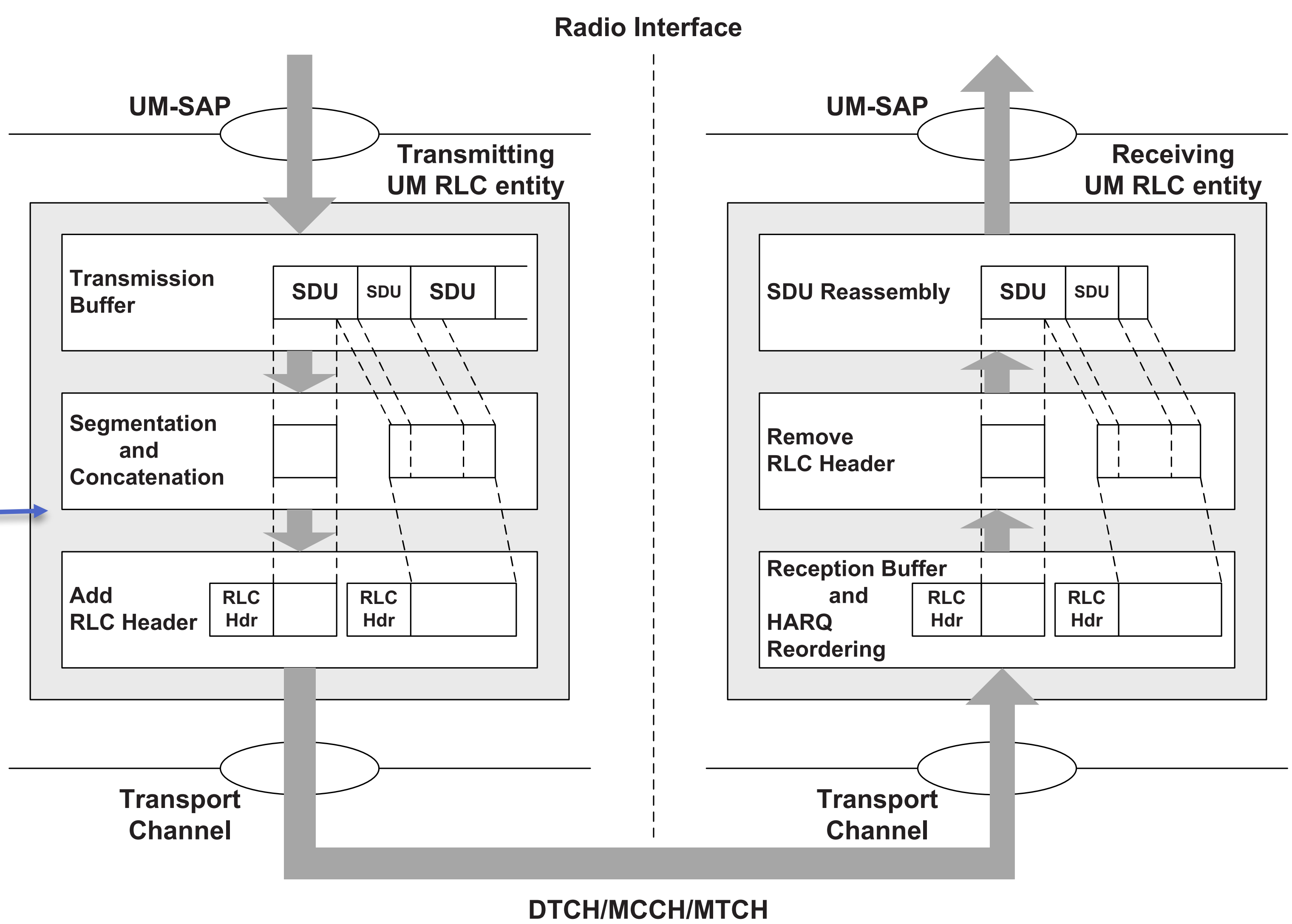
RLC Layer

- One RLC instance for each radio bearer
- It receives *Transmission Opportunities* from MAC
- It performs segmentation/concatenation of PDCP PDUS
- At receiver it performs reordering (due to variable delays @ MAC)
- 3 modes
  - Transparent Mode (TM)
  - Unacknowledged Mode (UM)
  - Acknowledged Mode (AM)

RLC Layer – TM mode

- Maps RLC SDUs into RLC PDUs
- Transparent: nothing changes
- Used for control signalling (before the connection enters the active state)

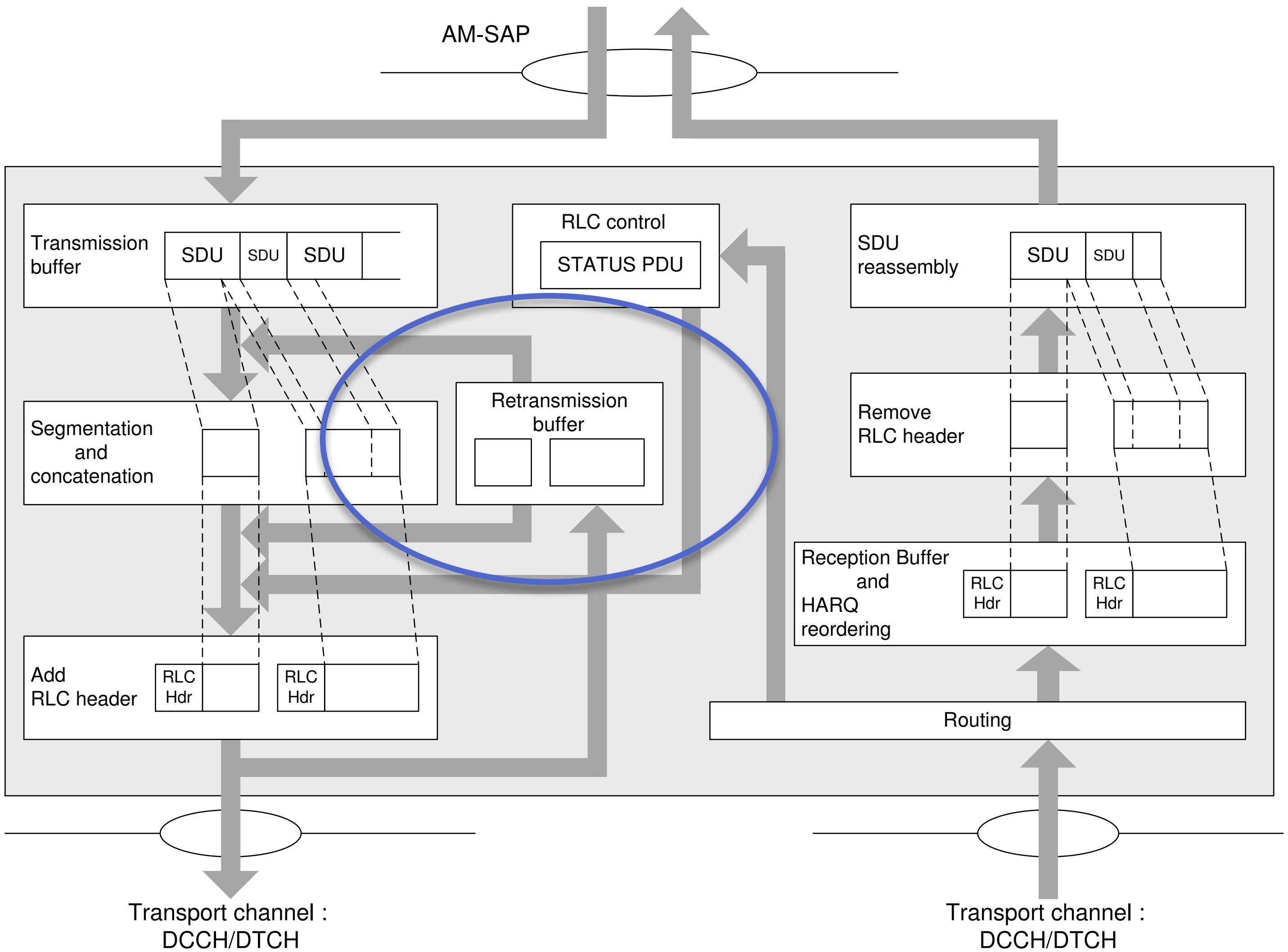
RLC Layer – UM mode



TX opp

Delay-sensitive & error-tolerant applications

RLC Layer – AM mode

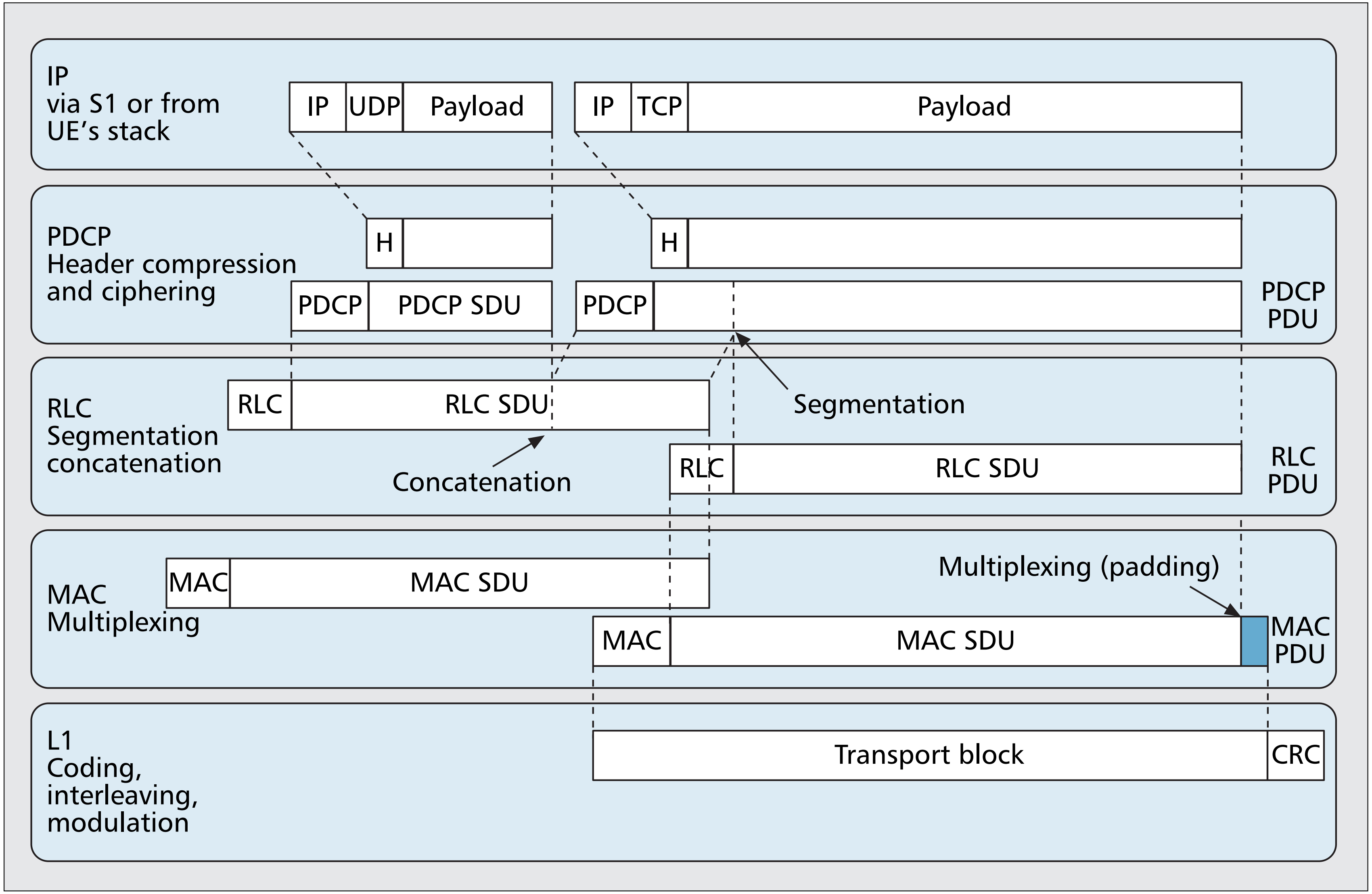


Delay-tolerant applications – RRC packets

PDCP Layer

- One PDCP instance for each radio bearer
- It receives IP packets + RRC control PDUs
- It performs
  - *Header compression*
  - *Data encryption/decryption*
  - *Reordering during handover events*

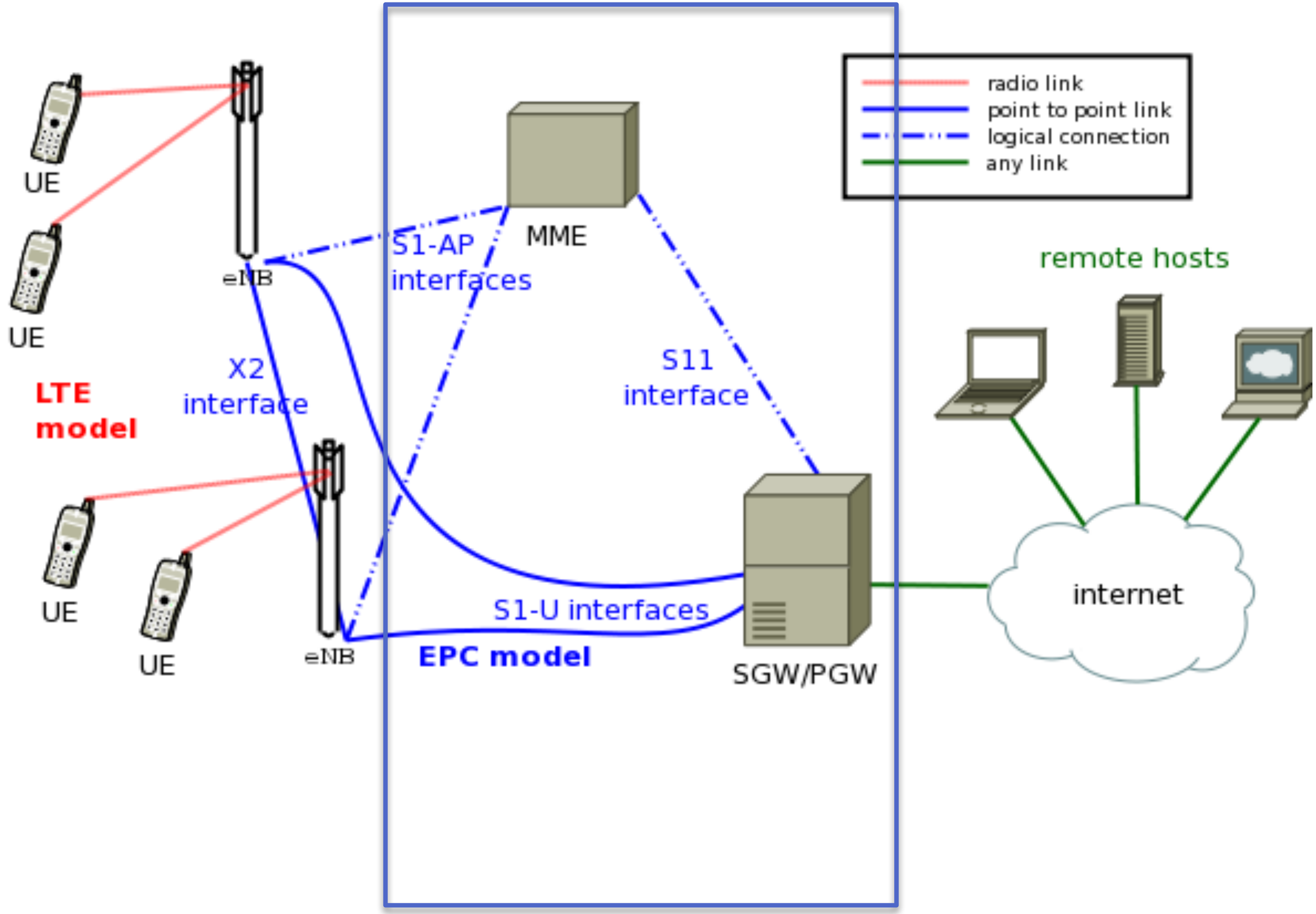
LTE User Plane Flow



RRC Layer

- Broadcast of System Information messages
- Management of
  - RRC Connection (RRC states: RRC idle, RRC connected)
  - Signalling & Data Radio Bearers (-> RLC, PDCP instances)
- Mobility (cell selection, handover)
- UE measurement reporting

EPC (LTE Core Network)



MME

- CN node: it interacts with the RRC layer for
  - Bearer management
  - Connection management
  - Mobility
- A LTE network may have more than one MME, all connected to the HSS (server that manages UEs' subscriptions)

P-GW + S-GW

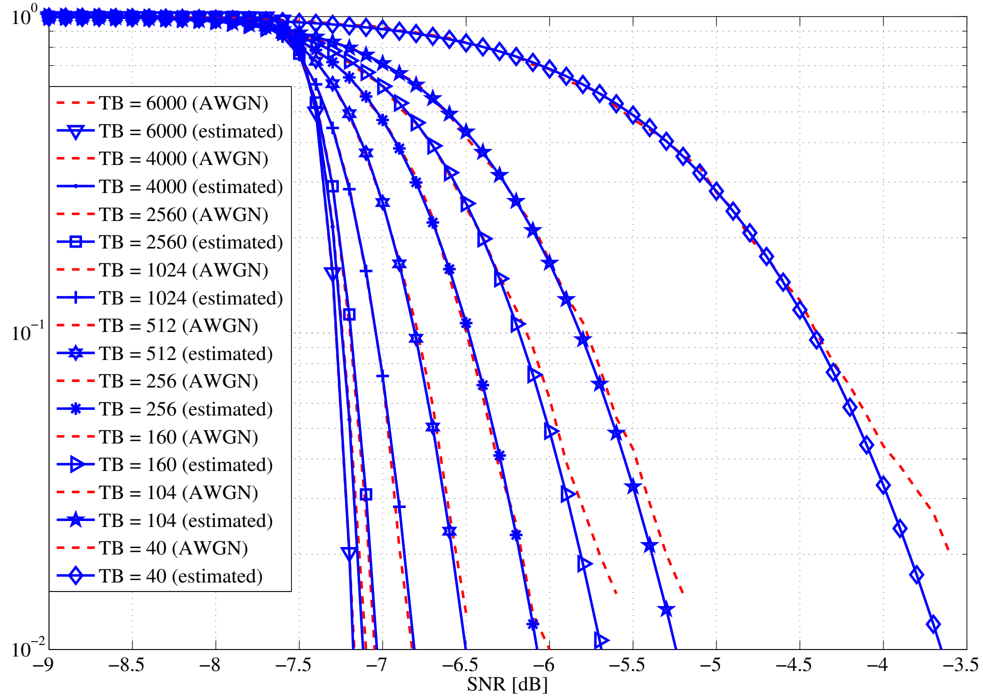
- P-GW is the CN node connected to the internet. It assigns IP packets to the respective data bearers
- S-GW receives IP packets from P-GW and
  - It is the mobility anchor for the UE
  - It interacts with the MME to track the UE position in the network
  - Connected to the eNB with the S1 interface

ns-3 LTE module (LENA)

- System Level end-to-end simulator

The link is abstracted by the SINR, and mapped to an error probability for a Transport Block

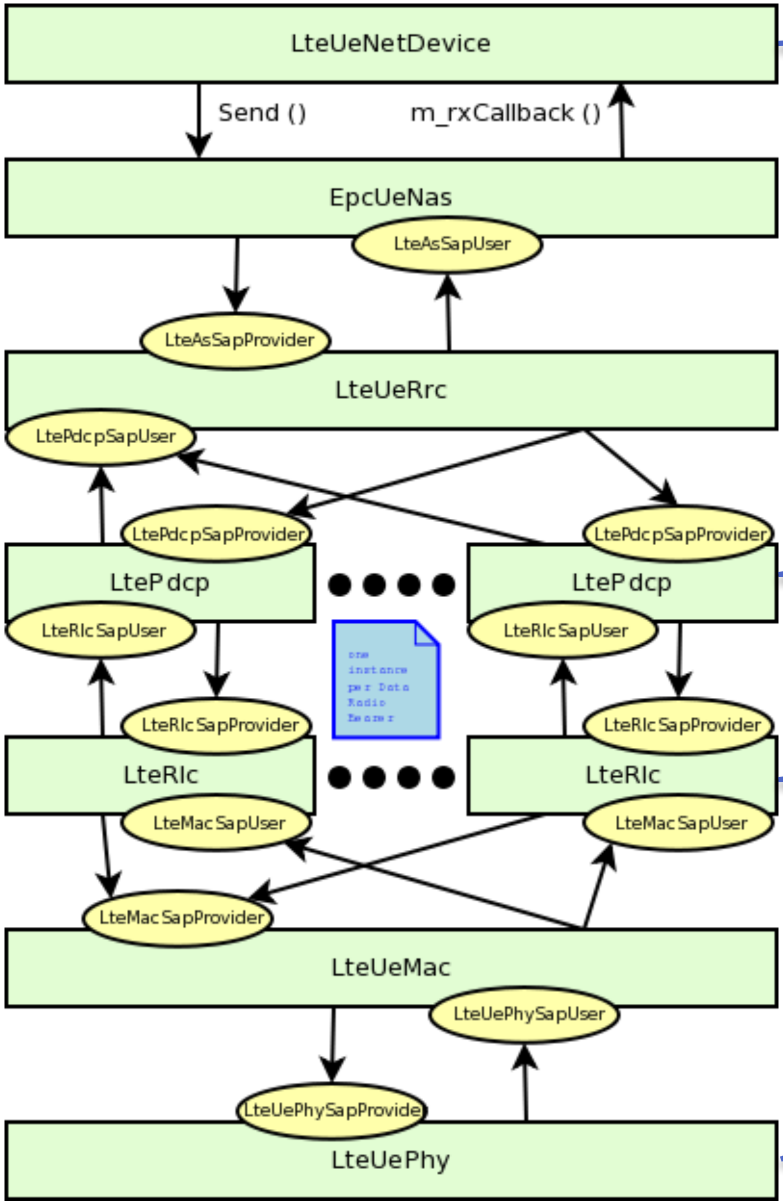
Detailed implementation of LTE layers + TCP/IP stack



ns3 LTE module - assumptions

- LTE model
  - Focus on RRC Connected state
  - It supports 10s of eNBs (also femtocells), 100s of UEs
  - Complete LTE stack (PHY, MAC, RLC, PDCP, RRC)
- EPC model
  - PGW and SGW are the same node
  - UDP instead of GTP
  - MME is an *ideal* node

ns3 LTE module – UE model



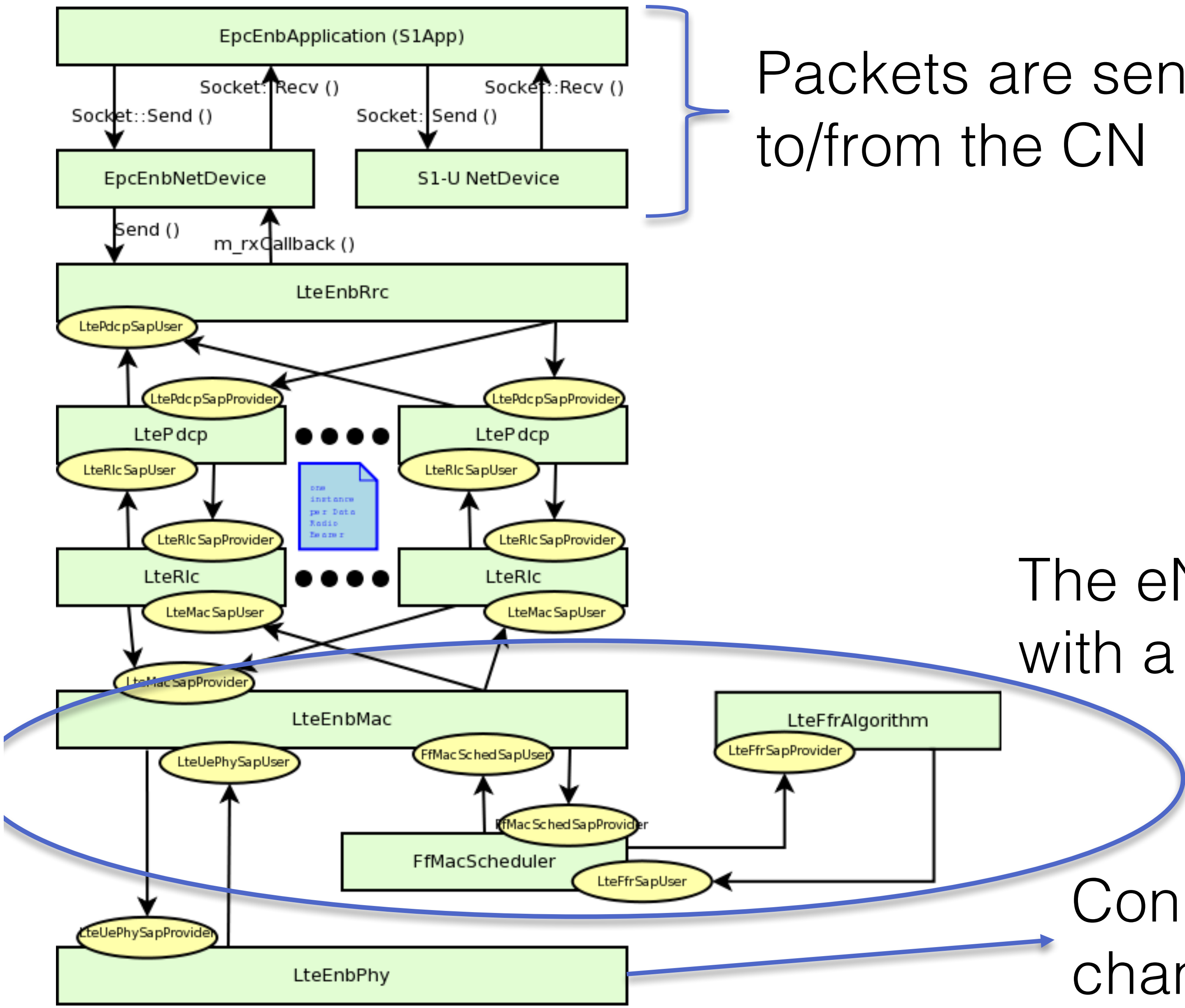
It receives IP packets

- No ROHC
- No packet reordering

All 3 modes (AM, UM, TM)

Pluggable to different channel models

ns3 LTE module – eNB model

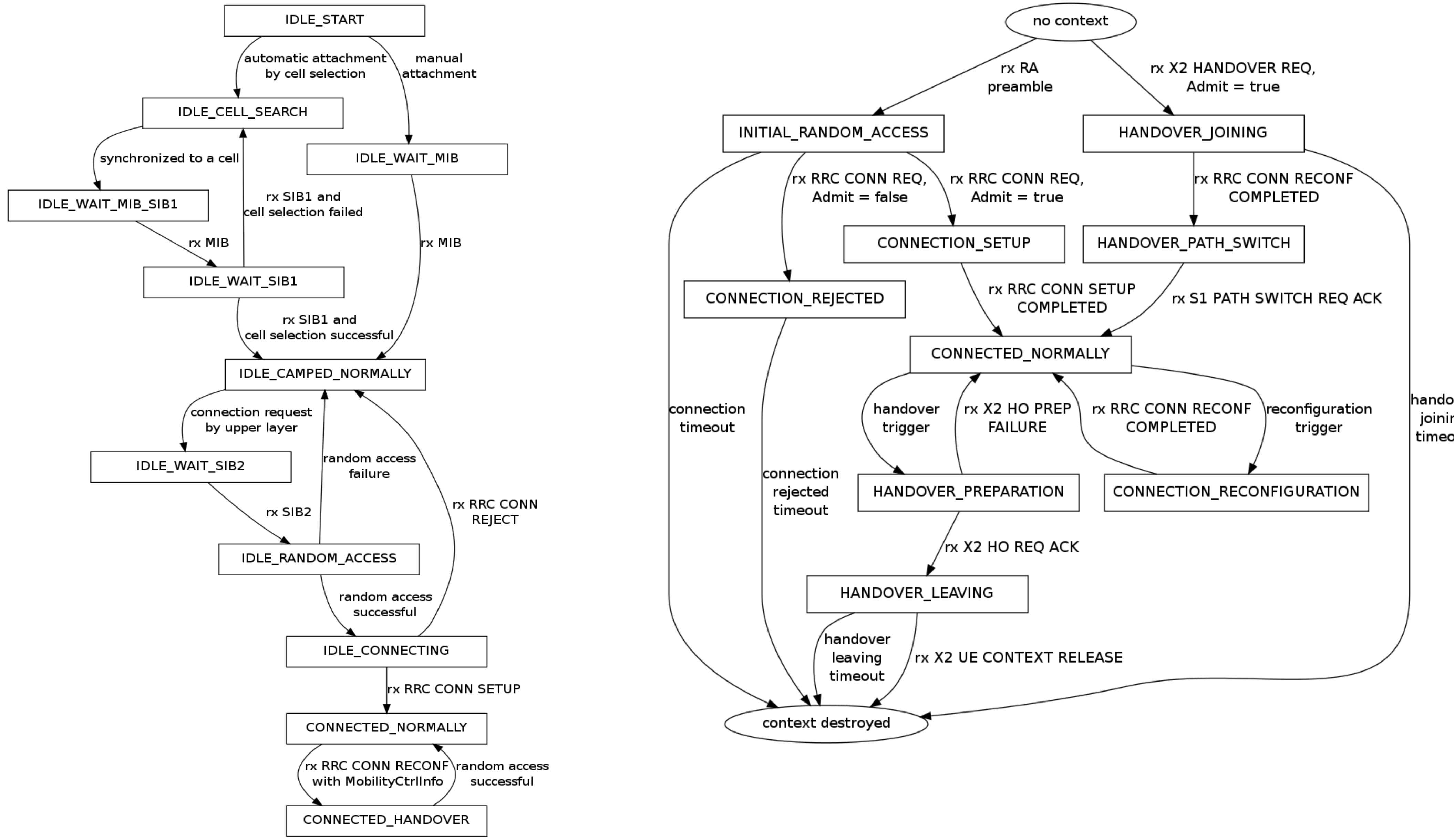


Packets are sent/received to/from the CN

The eNB MAC is interfaced with a scheduler (PF, RR…)

Connected to the same channel model of the UE

ns3 LTE module – RRC state machine



ns3 LTE module: lena-simple-epc.cc

#include instructions

```
#include "ns3/lte-helper.h"
#include "ns3/epc-helper.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/lte-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-helper.h"
#include "ns3/config-store.h"
//#include "ns3/gtk-config-store.h"

using namespace ns3;
```

ns3 LTE module: lena-simple-epc.cc


```

NS_LOG_COMPONENT_DEFINE ("EpcFirstExample");

int
main (int argc, char *argv[])
{

  uint16_t numberOfNodes = 2;
  double simTime = 1.1;
  double distance = 60.0;
  double interPacketInterval = 100;

  // Command line arguments
  CommandLine cmd;
  cmd.AddValue("numberOfNodes", "Number of eNodeBs + UE pairs", numberOfNodes);
  cmd.AddValue("simTime", "Total duration of the simulation [s])", simTime);
  cmd.AddValue("distance", "Distance between eNBs [m]", distance);
  cmd.AddValue("interPacketInterval", "Inter packet interval [ms])", interPacketInterval);
  cmd.Parse(argc, argv);
```

ns3 LTE module: lena-simple-epc.cc


```

Ptr<LteHelper> lteHelper = CreateObject<LteHelper> ();
Ptr<PointToPointEpcHelper>  epcHelper = CreateObject<PointToPointEpcHelper> ();
lteHelper->SetEpcHelper (epcHelper);

Ptr<Node> pgw = epcHelper->GetPgwNode ();
```

ns3 LTE module: lena-simple-epc.cc


```
  // Create a single RemoteHost
  NodeContainer remoteHostContainer;
  remoteHostContainer.Create (1);
  Ptr<Node> remoteHost = remoteHostContainer.Get (0);
  InternetStackHelper internet;
  internet.Install (remoteHostContainer);

  // Create the Internet
  PointToPointHelper p2ph;
  p2ph.SetDeviceAttribute ("DataRate", DataRateValue (DataRate ("100Gb/s")));
  p2ph.SetDeviceAttribute ("Mtu", UintegerValue (1500));
  p2ph.SetChannelAttribute ("Delay", TimeValue (Seconds (0.010)));
  NetDeviceContainer internetDevices = p2ph.Install (pgw, remoteHost);
  Ipv4AddressHelper ipv4h;
  ipv4h.SetBase ("1.0.0.0", "255.0.0.0");
  Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign (internetDevices);
  // interface 0 is localhost, 1 is the p2p device
  Ipv4Address remoteHostAddr = internetIpIfaces.GetAddress (1);

  Ipv4StaticRoutingHelper ipv4RoutingHelper;
  Ptr<Ipv4StaticRouting> remoteHostStaticRouting = ipv4RoutingHelper.GetStaticRouting (remoteHost->GetObjec
  remoteHostStaticRouting->AddNetworkRouteTo (Ipv4Address ("7.0.0.0"), Ipv4Mask ("255.0.0.0"), 1);
```

ns3 LTE module: lena-simple-epc.cc


```
NodeContainer ueNodes;
NodeContainer enbNodes;
enbNodes.Create(numberOfNodes);
ueNodes.Create(numberOfNodes);

// Install Mobility Model
Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
for (uint16_t i = 0; i < numberOfNodes; i++)
  {
    positionAlloc->Add (Vector(distance * i, 0, 0));
  }
MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.SetPositionAllocator(positionAlloc);
mobility.Install(enbNodes);
mobility.Install(ueNodes);

// Install LTE Devices to the nodes
NetDeviceContainer enbLteDevs = lteHelper->InstallEnbDevice (enbNodes);
NetDeviceContainer ueLteDevs = lteHelper->InstallUeDevice (ueNodes);
```

ns3 LTE module: hex grid

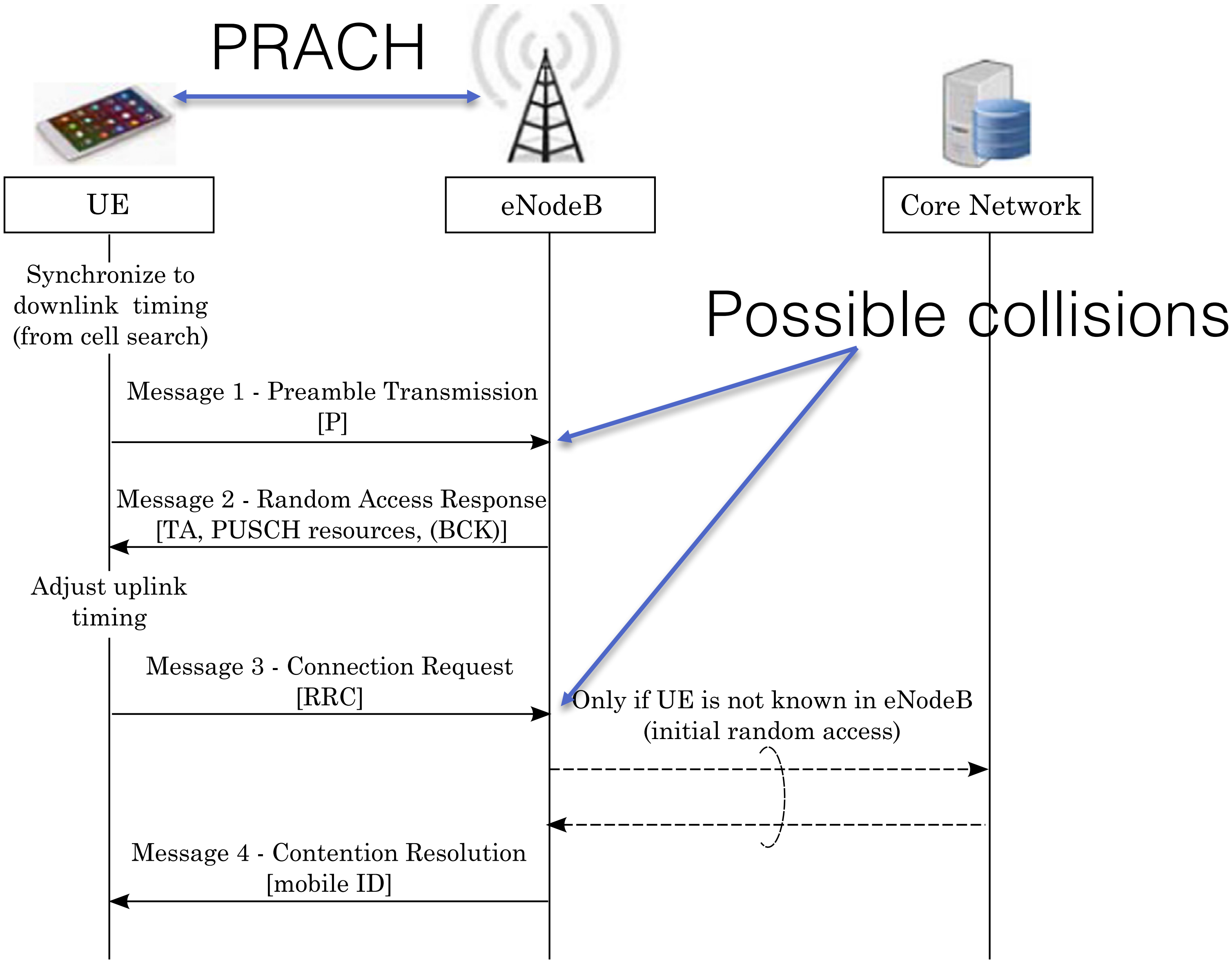

```
BuildingsHelper::Install (macroEnbs);
Ptr<LteHexGridEnbTopologyHelper> lteHexGridEnbTopologyHelper = CreateObject<LteHexGridEnbTopologyHelper> ();
lteHexGridEnbTopologyHelper->SetLteHelper (lteHelper);
lteHexGridEnbTopologyHelper->SetAttribute ("InterSiteDistance", DoubleValue (interSiteDistance));
lteHexGridEnbTopologyHelper->SetAttribute ("MinX", DoubleValue (interSiteDistance/2));
lteHexGridEnbTopologyHelper->SetAttribute ("GridWidth", UintegerValue (nMacroEnbSitesX));
Config::SetDefault ("ns3::LteEnbPhy::TxPower", DoubleValue (macroEnbTxPowerDbm));
lteHelper->SetEnbAntennaModelType ("ns3::ParabolicAntennaModel");
lteHelper->SetEnbAntennaModelAttribute ("Beamwidth",   DoubleValue (70));
lteHelper->SetEnbAntennaModelAttribute ("MaxAttenuation",     DoubleValue (20.0));
lteHelper->SetEnbDeviceAttribute ("DlEarfcn", UintegerValue (macroEnbDlEarfcn));
lteHelper->SetEnbDeviceAttribute ("UlEarfcn", UintegerValue (macroEnbDlEarfcn + 18000));
lteHelper->SetEnbDeviceAttribute ("DlBandwidth", UintegerValue (macroEnbBandwidth));
lteHelper->SetEnbDeviceAttribute ("UlBandwidth", UintegerValue (macroEnbBandwidth));
NetDeviceContainer macroEnbDevs = lteHexGridEnbTopologyHelper->SetPositionAndInstallEnbDevice (macroEnbs);
```

ns3 LTE module: limitations

- Simple CN model
- No transition from RRC Connected to RRC Idle
- ~~No realistic Random Access~~

LENA+ extension

LTE Random Access



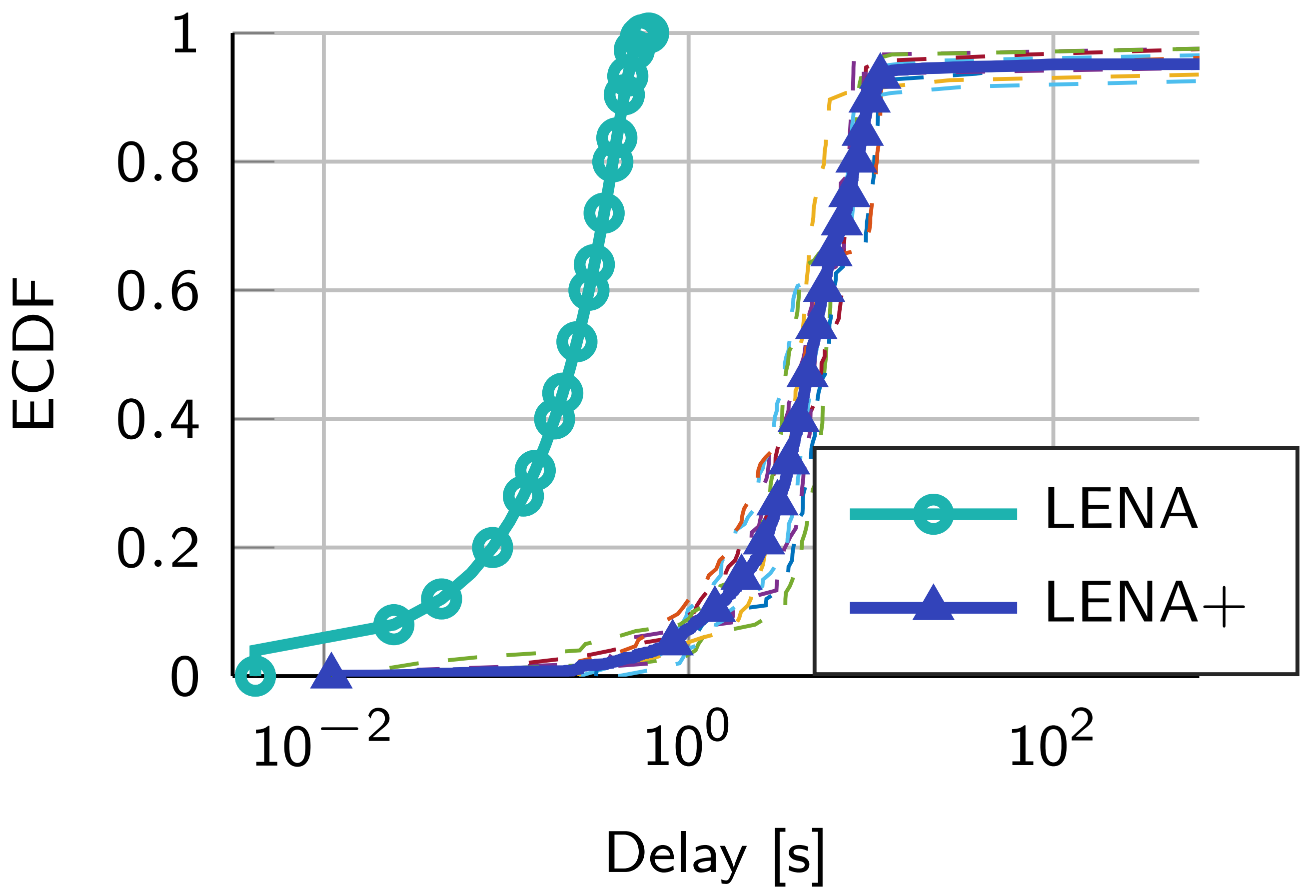
LTE Random Access in ns-3

- PRACH is ideal
  - No propagation, every UE has the same SINR
  - Preamble is always detected
- Msg 3 and Contention Resolution are not modeled
  - Collision is always solved at the first step
- Unrealistic modelling of the performance with massive access

ns-3 LENA+

- PRACH is modelled as a real channel
  - With noise, radio propagation and miss detection
- Collisions may
  - be detected or not at the first step
  - If not, they are detected at the third step
- Added PHY and MAC details in compliance with 3GPP requirements
- https://github.com/signetlabdei/lena-plus

ns-3 LENA+: example



References

- LTE
  - Sesia, LTE – The UMTS Long Term Evolution
  - 3GPP website with Technical Reports and Specs
  - A. Larmo, M. Lindström, M. Meyer, G. Pelletier, J. Torsner and H. Wiemann, "The LTE link-layer design," in IEEE Communications Magazine, vol. 47, no. 4, pp. 52-59, April 2009
- LTE in ns-3
  - https://www.nsnam.org/docs/models/html/lte-design.html (with many other references)
  - https://www.nsnam.org/docs/models/html/lte-user.html (ns-3 LTE user manual)
  - M. Polese, M. Centenaro, A. Zanella and M. Zorzi, "M2M massive access in LTE: RACH performance evaluation in a Smart City scenario," 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016

ns-3 NYU mmWave module