# openWNS

## Open Source Wireless Network Simulator

*27. Treffen der VDE/ITG-Fachgruppe 5.2.4*

*Mobilität in IP-basierten Netzen*

*Simulating Mobile Networks*

**Chair of Communication Networks**

RWTH Aachen University

*Maciej Mühleisen, Daniel Bültmann*

*{mue,dbn}@comnets.rwth-aachen.de*

# Overview

- Introduction
- WNS Structure
  - Simulator Core
  - Layer Development Kit Framework
  - Channel Model Framework
- Roadmap to Open Source

# Introduction: What it is and what it's not

## What *open*WNS is

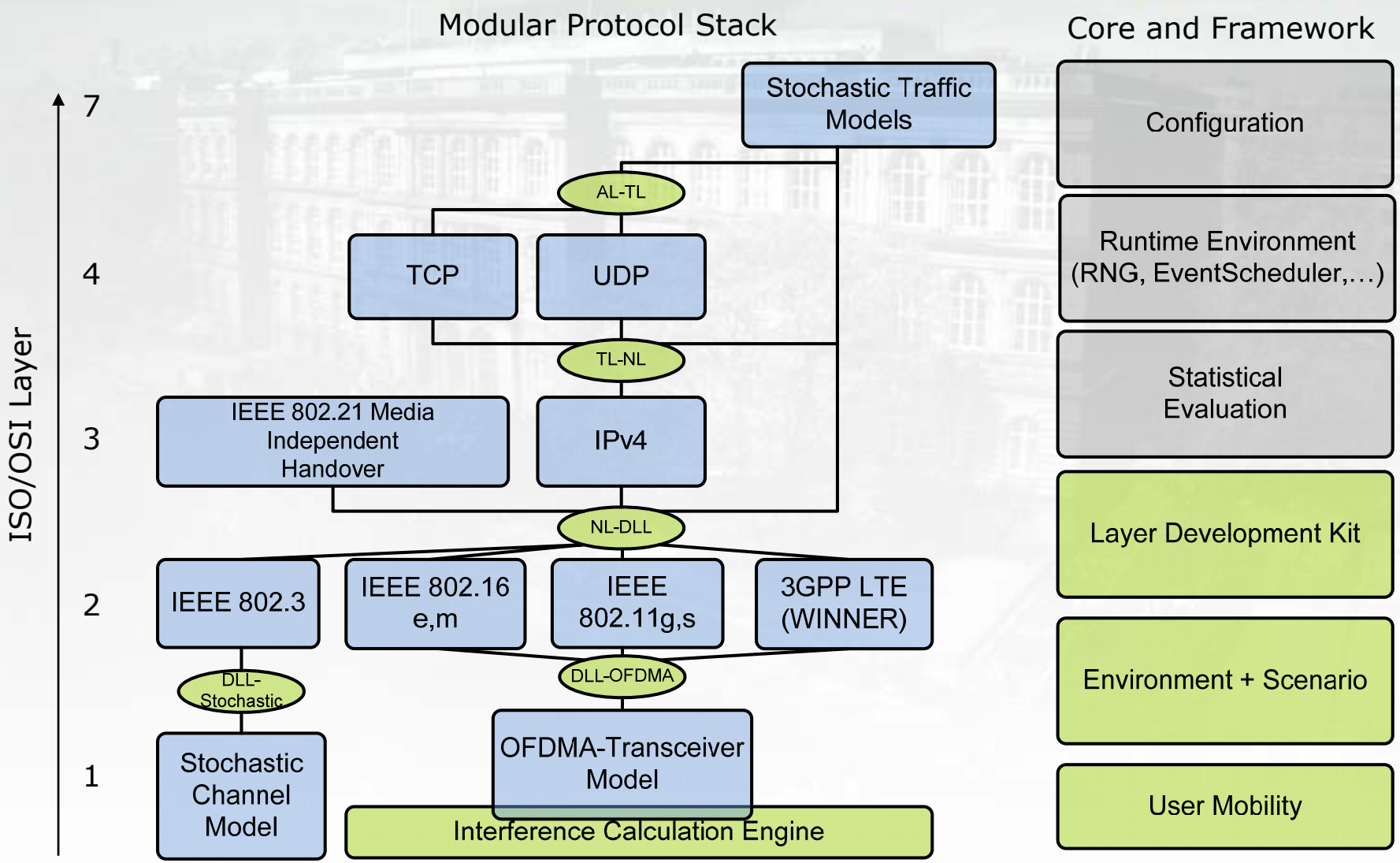
- Dynamic Event Driven System Level Simulation Platform
  - Investigations of dynamic protocol behavior
  - Cross-layer effects
  - Online calculation of intra- and inter-cell interference
- Full fledged protocol stacks
  - Used for investigation of
    - **IEEE 802.16 e, j, m**
    - **3GPP-LTE (WINNER)**
    - **IEEE 802.11 g,s**
- Typical Results
  - Protocol level results
    - E2E Packet Delay, Throughput
    - Buffer Fill Levels
    - Retransmissions
    - BER, PER, FER
  - Physical layer results
    - SINR distributions (over area, per terminal, per cell)

## What *open*WNS is **NOT**

- Radio planning tool with ray tracing capabilities covering large scenarios of several 100 km²
- Tool to design and run protocol stacks on an FPGA
- Monte-Carlo Simulator

→ But since it is open source, you never know …

# openWNS Structure



# openWNS Core - Configuration

- Configuration uses **Python** programming language
  - Interpreter language (no compiling)
  - Object oriented
  - Easy to use
  - Well documented

- Object oriented programming tries to imitate the real world:
  - "An access point is a special kind of WLAN device"
    ⇒ Deriving
  - "An access point has an IP protocol implementation. It also has an antenna"
    ⇒ Composition
  - "Create 10 WLAN devices. Place them in a line with 10m distance"
    ⇒ Control structures: for-loop, if…then…else …

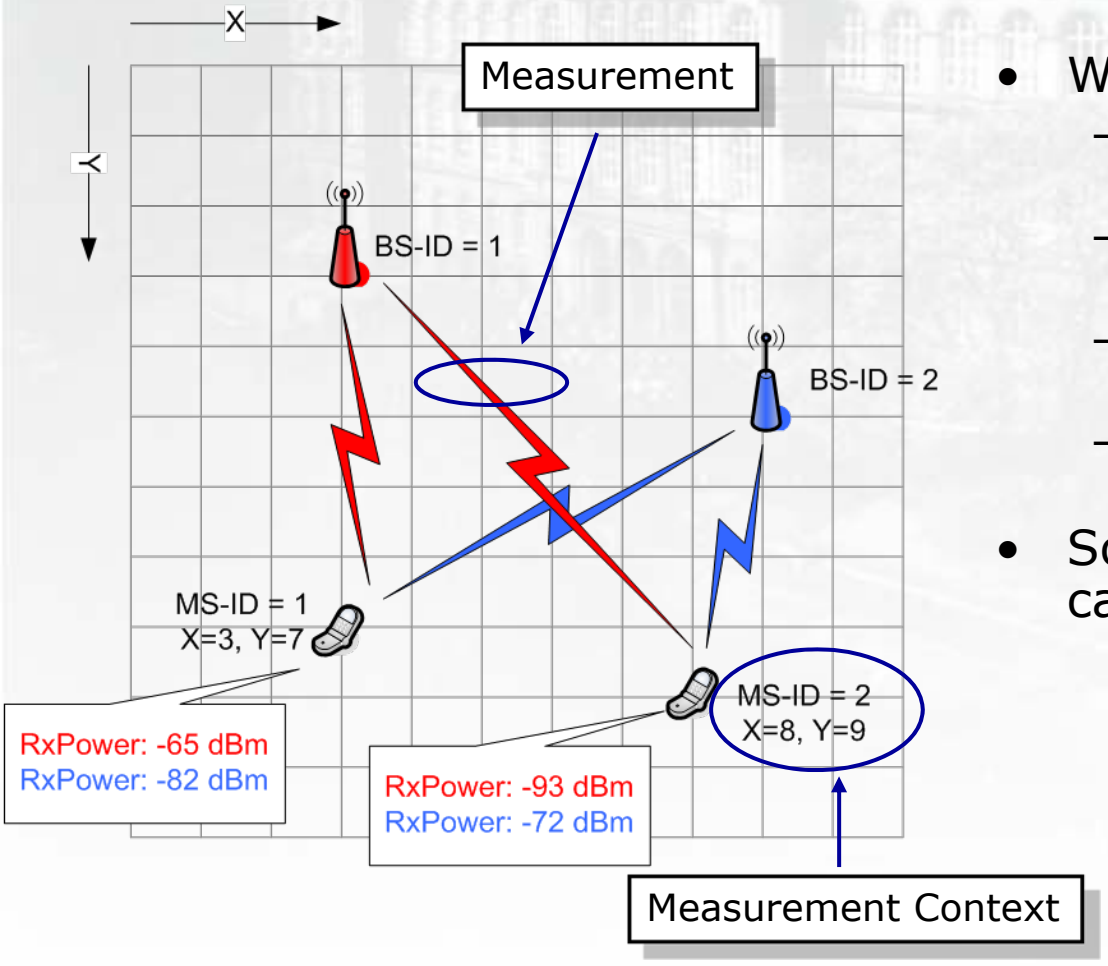
# openWNS Core – Runtime Environment

- Uses boost C++ libraries[1]
- Libraries are candidates to become part of standard C++[2]
- Boost is available on multiple platforms

- Event scheduler:
  - Events can be scheduled and canceled with O(1) (FastList)
- Boost random number generator:
  - Default uniform RNG is Mersenne Twister MT19937 [1]
  - Negligible correlation
  - Period: $2^{19937} - 1$
  - Fast
  - Many distributions available: Uniform, Triangle, Bernoulli, Cauchy, Exponential, Normal, Log-Normal, Uniform on n-dimensional sphere

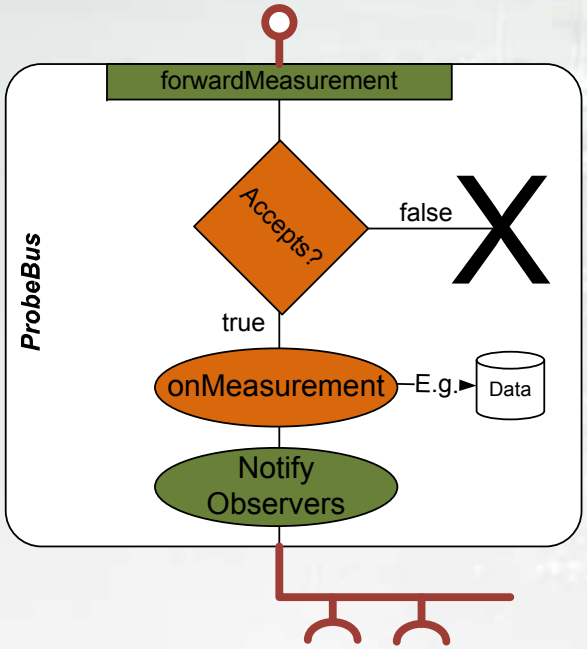
[1] http://www.boost.org [2]Proposed Draft Technical Report on C++ Library Extension

# openWNS Core - Evaluation Sub-System

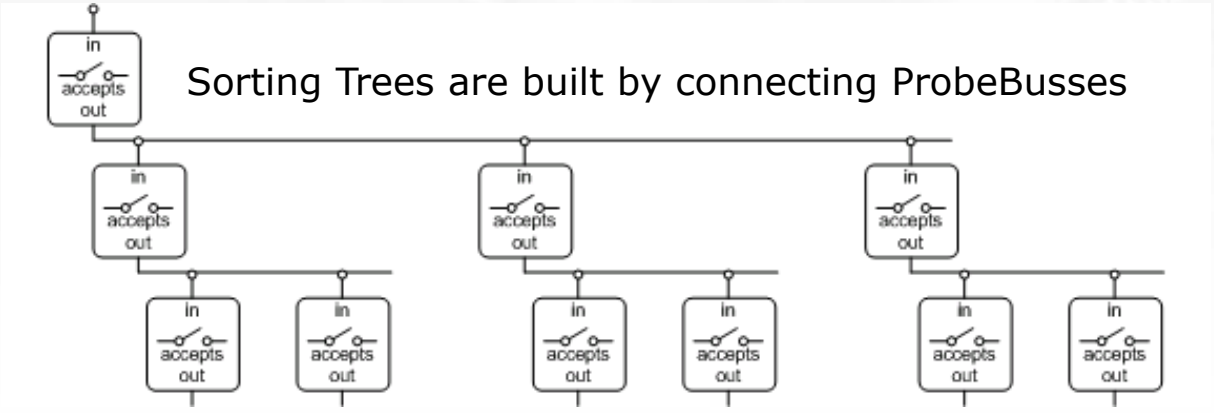



- What you want:
  - What are the RxPower statistics of BS 1 and BS 2 ?
  - What is the RxPower statistics that MS 1 sees from BS 2 ?
  - What is the average RxPower measured at Position X,Y ?
  - What is the RxPower measured from BS1, BS2 at Position X,Y ?
- Sort measurements before calculating statistics!

# openWNS Core - Evaluation Sub-System

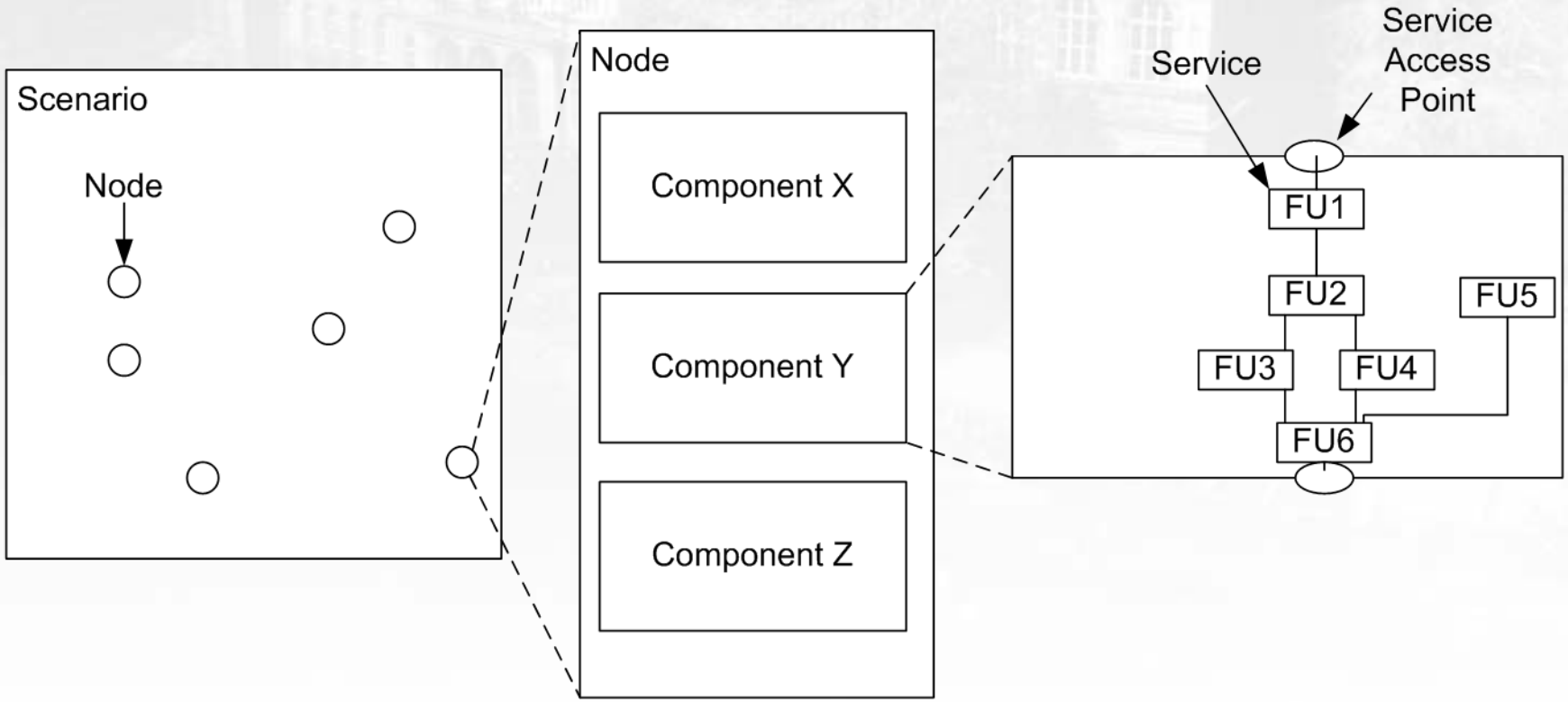



- Basic building block follows Subject/Observer pattern
- ProbeBus
  - Processing Aspect
  - Forwarding Aspect
  - Connecting Aspect
- Behaviour Extension Points
  - Accepts
  - onMeasurement
- Behaviour can be implemented in your configuration!



# openWNS Framework – Layer Development Kit

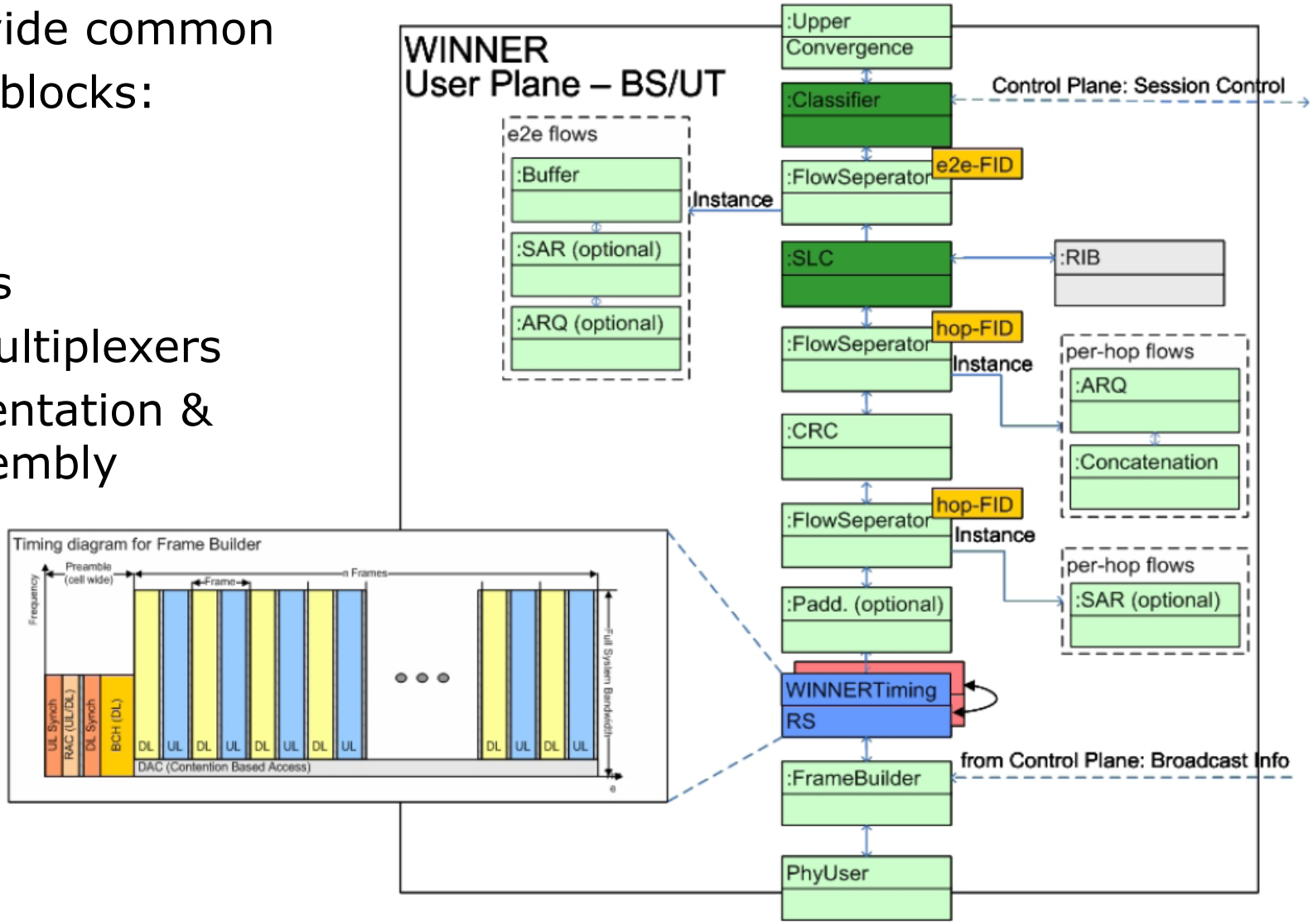
- Scenario is built on startup: Node / Component model
- Components (layers)
  - Communicate through services
  - Can be built using **reusable** "Functional Units" (FUs)



# openWNS Framework – Layer Development Kit

FUs provide common building blocks:

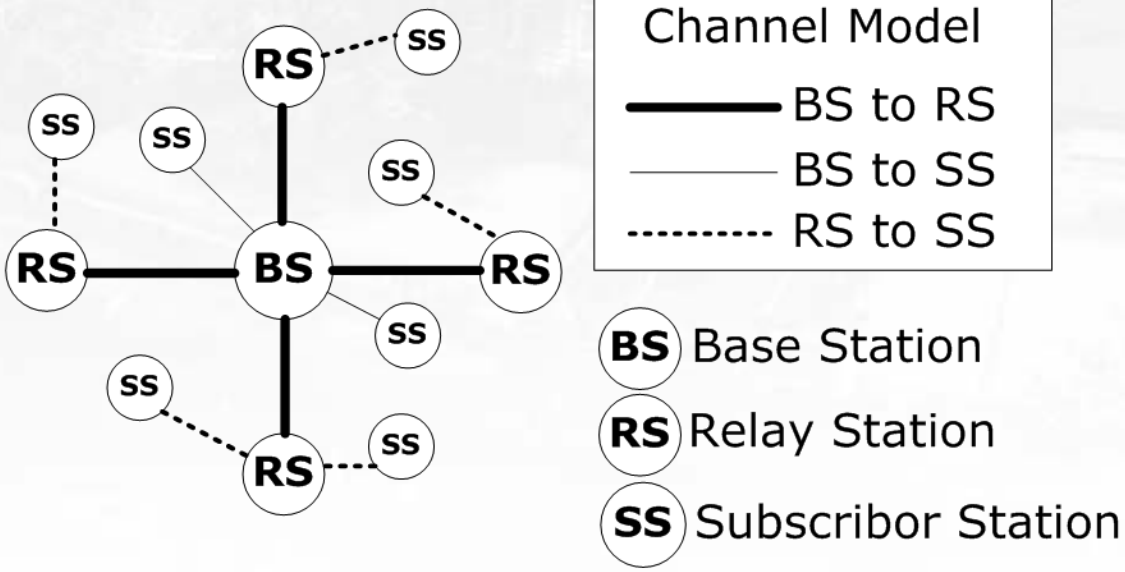
- CRC
- ARQ
- Buffers
- (De)Multiplexers
- Segmentation & Reassembly
- …



# openWNS Framework – Channel Model

- Channel and PHY layer modeling framework
- Received radio power calculation
  - Antenna gain
  - Path loss
  - Shadowing
  - Fading

  } Choose per receiver / transmitter type pair
- Interference calculation
- Mobility



# openWNS Framework – Channel Model

Path loss models:

- Distance and frequency dependent
- Different attenuation factors for different distances (Multi Slope)
- Optional: Implement path loss function in configuration
  - No recompilation required

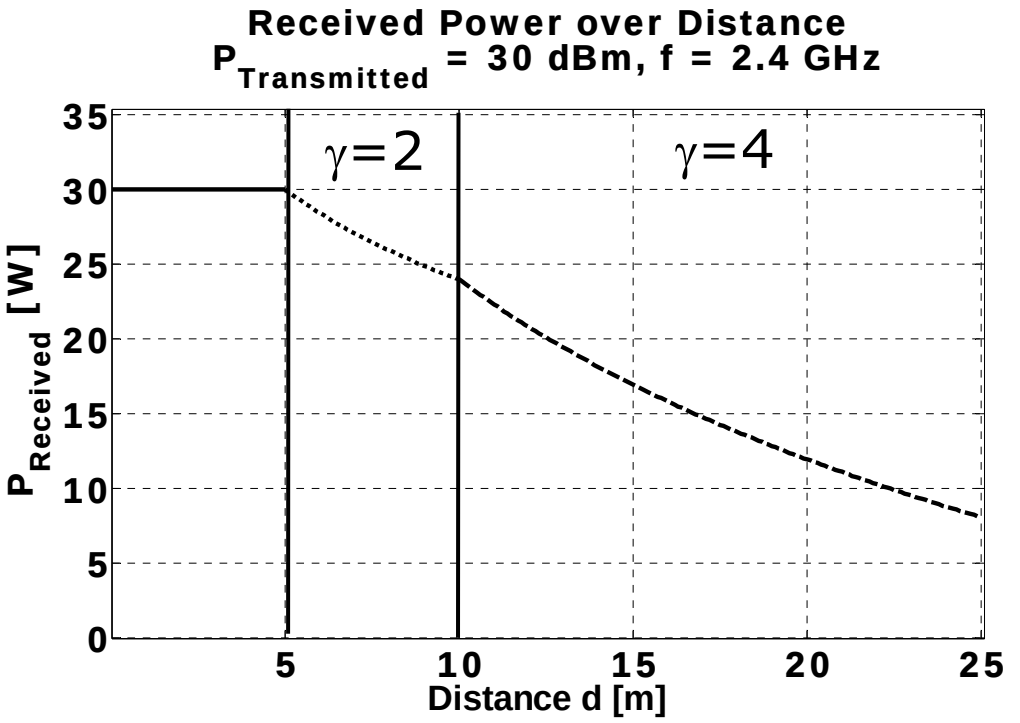
$$L[\mathrm{dB}] = 10\gamma \log_{10}\left(\frac{\lambda}{4\pi d}\right)$$

$L$ : Path loss [dB]

$d$ : Distance [m]

$\gamma$ : Propagation factor (Slope)

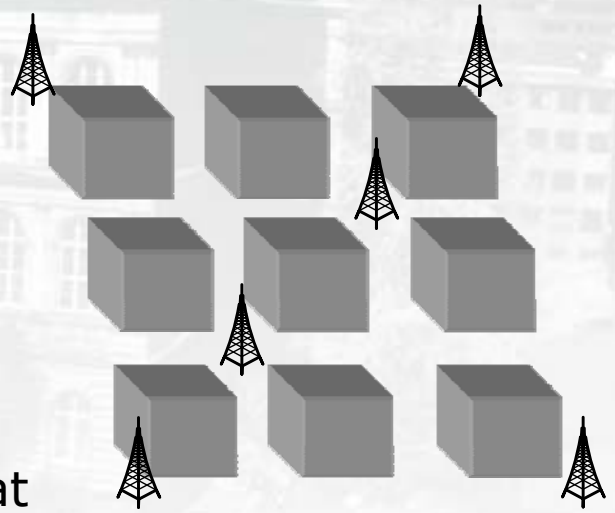
$\lambda$ : Wavelength [m]

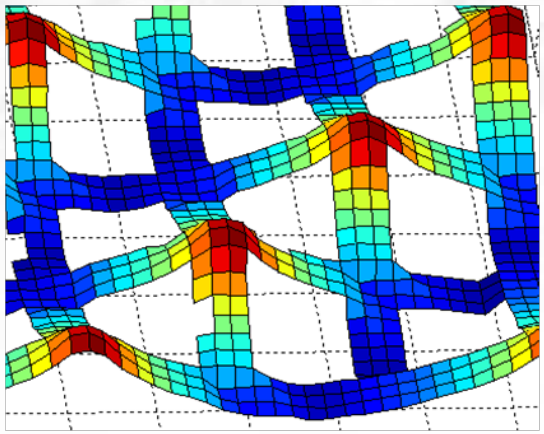


# openWNS Framework – Channel Model

Shadowing models:

- Attenuation from solid objects (buildings, walls, …)
- Different models / implementations:
  - Object based
    - Distribute objects on scenario with given attenuation
    - Calculate penetrated objects between transmitter and receiver
    - Accumulate attenuation
  - Map file based
    - Calculate received power for each BS/AP at different positions
    - Interpolate in between
    - Assume symmetric channel
  - Stochastic: 2D spatial correlated log-normal[1]



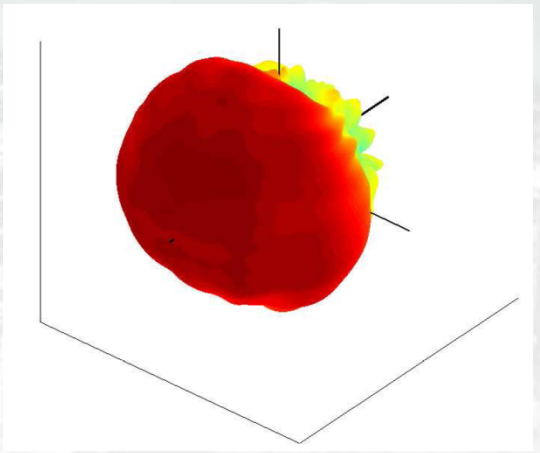


[1]Cai, Xiaodong and Giannakis, Georgios B.: "A Two-Dimensional Channel Simulation Model for Shadowing Processes"

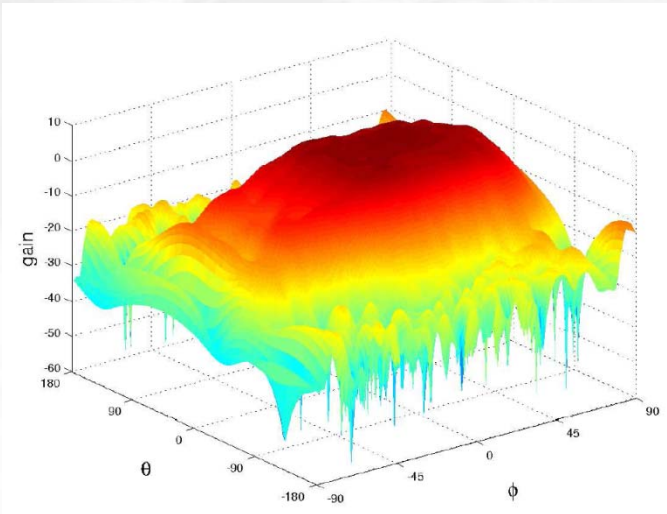
# openWNS Framework – Channel Model

Antennas:

- Calculate signal gain depending on direction
- Static
  - Isotropic
  - 2D pattern
  - 3D pattern

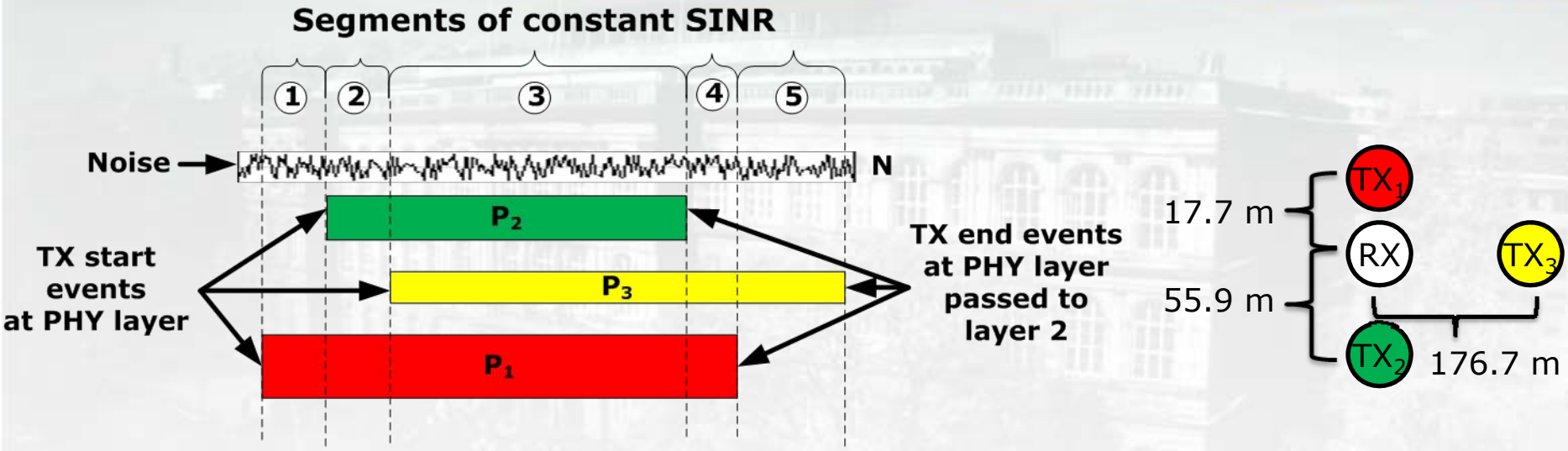


- Dynamic (Beamforming)
  - Optimal beamformer[1]
  - Linear or circular segment alignment
  - Configurable number of antenna segments
  - Used for SDMA




[1]L.C. Godara: "Application of antenna arrays to mobile communications, Part II: Beamforming and direction of arrival considerations"

# openWNS Framework – Channel Model




SINR calculation:

- Identify segments of constant SINR
- Calculate SINR (time weighted average / no averaging)
- Let layer 2 decide successful reception (link-level mapping)
- Optional optimization: Layer 1 addressing
  - Only calculate interference in target receiver

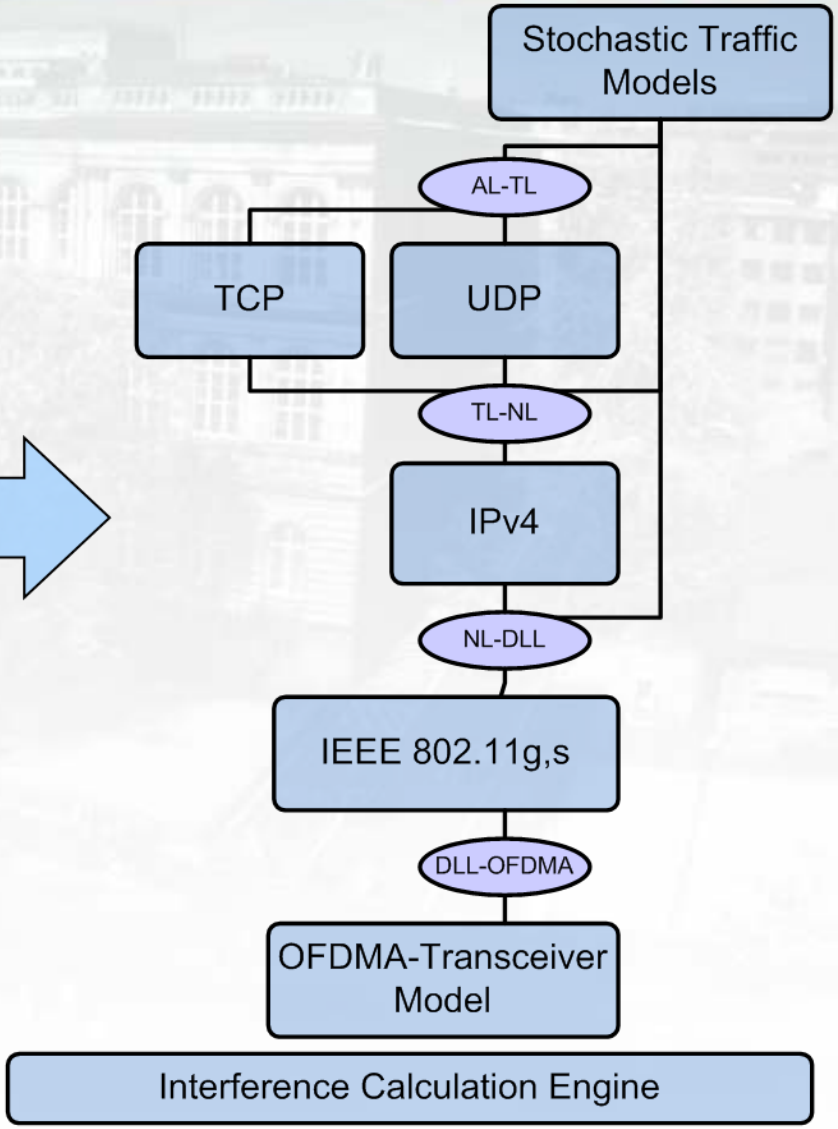
# Roadmap to Open Source

- Available on Launchpad
  - www.openwns.org
  - www.launchpad.net/openwns-sdk
- Lesser General Public License (LGPL)
- The Core is already published
- Published by the end of 2008:
  - Full IEEE 802.11g,s simulator
- IEEE 802.3, IEEE 802.16, IEEE 802.21 & IST-WINNER will follow later
- Can't wait?

  ⇒ **Become a "Friendly User"**

  **&**

  ⇒ **Get the 802.11 protocol stack**



# Thank you for your attention!

Contact:
mue@comnets.rwth-aachen.de
info@openwns.org