

Open-Source Prototyping of 5G Wireless Systems for UGV/UAV

Team: SDMAY20-36

Team Members: William Byers, Ibrica Tusic, Samuel Stanek,

Nathan Whitcome, Andrew Eschweiler, Nicholas Lorenz

Faculty Advisor/Client: Hongwei Zhang

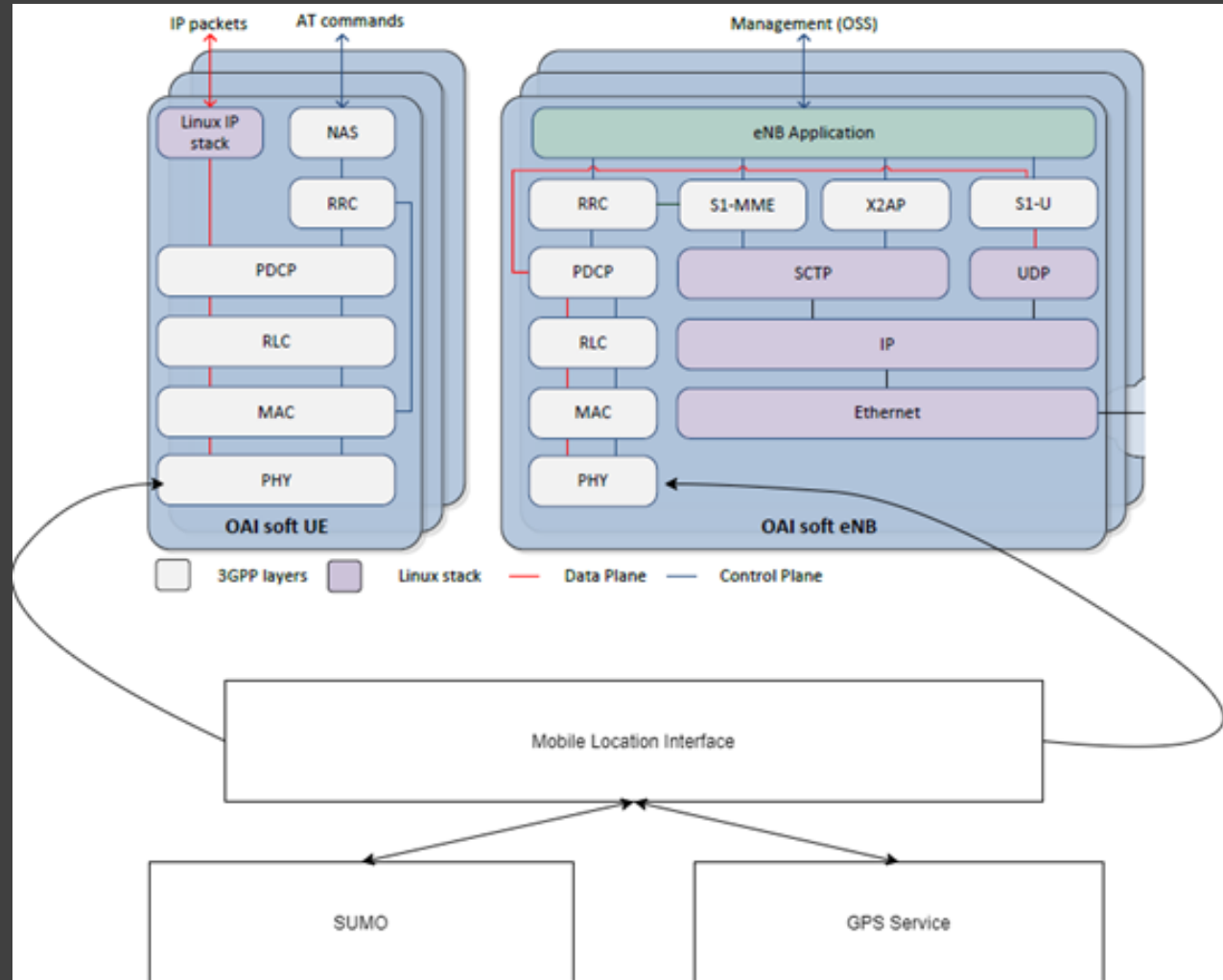
High Level Overview

- Create a 5G wireless solution that allows for large scale simulation of wireless networks
 - Low latency
 - High throughput
 - Extremely reliable network
- Tools
 - Open Air Interface (OAI)
 - Simulation of Urban Mobility (SUMO)
- Future uses
 - Connected autonomous transport
 - Smart agriculture

Problem Statement

- Need to modify OAI to simulate 5G networks in various traffic conditions
 - Need to simulate 100s of nodes
 - Ensure simulation output matches theory (no bugs)
- Utilize open source tools to solve the above problem

Conceptual Sketch



Functional Requirements

- Design a scalable system to test 5G scheduling algorithms
- Ensure the solution is:
 - Reliable
 - Has low latency
 - Has high throughput
- Utilize open source tools and follow good software practices
 - CI/CD
 - Jenkins
 - Cloud Deployments
 - OAI
 - SUMO

Technical/Other Constraints

- Limited in terms of hardware
 - Current server specs:
 - Intel Xeon 8 core processor (2.4 Ghz)
 - 8 GB of RAM
 - 120 GB disk
- Simulator is difficult to use
 - Requires multiple computers for various components
 - Very specific hardware requirements
 - Linux Low Latency Kernel
 - AVX512 Instruction Support
 - Turbo Decoding
 - Disable Processor Power Saving Features

Potential Risks & Mitigation

- Lack of knowledge of complex network simulations
 - LTE and 5G network architecture
 - Mitigated by learning as much as possible about how these systems work within reason
- Bugs in our code
 - Learn how to properly debug these kinds of systems
 - Ask for assistance from people who worked on the development team for the simulator
- Specific hardware requirements and vague default test case results
 - Mitigated by creating a build system to make it easier to track errors

Potential Risks & Mitigation

Risks

- Difficulty scaling networks to many UEs and eNBs
- New scheduling algorithm integration
- Difficulty recreating system configurations
 - Similar issues with cloud deployments
- Unknown simulator bugs
 - Large software project
 - Lots of legacy code

Mitigation

- Potentially deploy to cloud services
 - Cloudfab, ExoGENI
- Use CI/CD to verify old functionality isn't broken
- Use scripts where possible to create network layouts
 - Cloud formation templates
- Verify experiments match theory

Resource Cost/Estimate

- Hardware capable of running OAI and SUMO
 - A test bed specifically for OAI
 - Linux
- No financial requirement
 - OAI and SUMO are open source

Project Milestones

- Get OAI working
 - UE (like a cell phone)
 - ENB (a radio tower)
 - EPC (evolved packet core)
- Integrate OAI and SUMO
 - Get SUMO positional data
 - Find where in OAI to insert SUMO data
- Implement algorithm and check its validity
- Test system with network simulator and SUMO
 - Low Latency
 - High Throughput
 - High Reliability
 - Interoperability with current solutions

Project Schedule

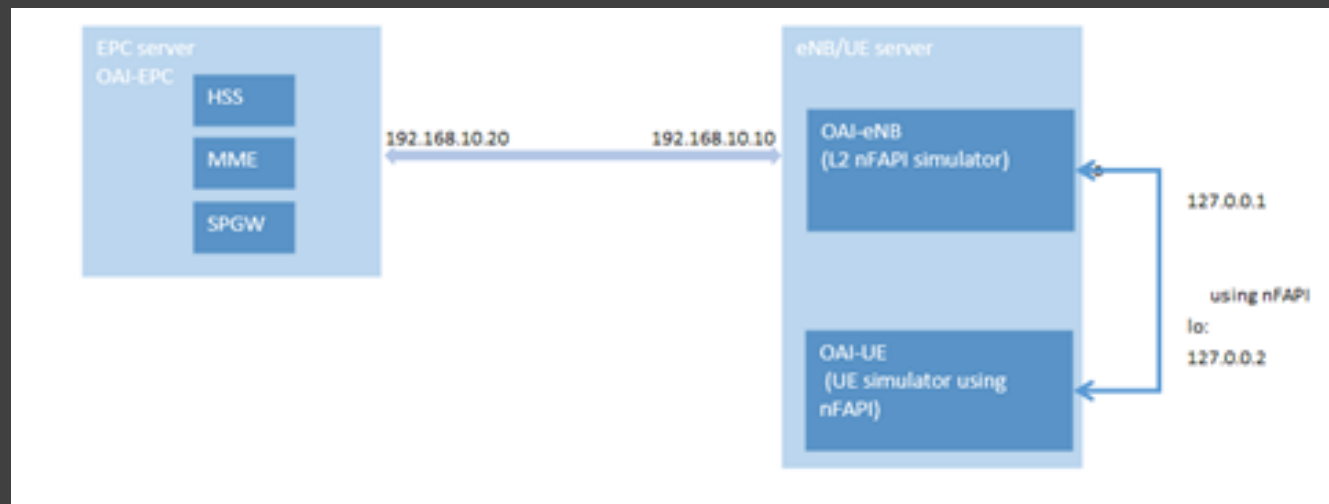
Assignment/Task	9/9-9/23	9/23-10/7	10/7-10/21	10/21-11/4	11/4-11/18	11/18-12/2	12/2-12/16	Winter Break	1/13-1/27	1/27-2/10	2/10-2/24	2/24-3/9	3/9-2/25	3/25-5/1
Read 4G/LTE Book	Green	Green	Green					Grey						
Research PRKS Paper	Green	Green	Green					Grey						
Research CPS Paper	Green	Green	Green					Grey						
Research SUMO	Green	Green	Green					Grey						
Research OAI and specific Install Steps				Green	Green	Green	Green	Grey						
Set Up Hardware/Operating Environment				Green	Green	Green	Green	Grey						
Install OAI					Green		Green	Grey						
Install SUMO				Green	Green	Green		Grey						
Verify/Run OAI on Server							Green	Grey	Green	Green	Green	Green	Green	
OAI/SUMO Integration							Yellow	Grey	Yellow	Yellow	Yellow	Yellow	Yellow	
Algorithm Development						Yellow	Yellow	Grey	Yellow	Yellow	Yellow	Yellow	Yellow	
Verify and Test Simulation								Grey						Red
Writing Report								Grey				Green	Green	Green
Compare Simulation Results to Control								Grey						Red
Hardware Deployment								Grey						Red
Finalize Report and Project								Grey						Green

Functional Decomposition

- SUMO Traffic Generation
 - SUMO Server to create traffic simulations
 - Update positions of nodes in OAI using SUMO data
 - Interface between OAI and SUMO
- Network Emulation
 - Various eNB and UE in configurations
 - 1 eNB, 1 UE
 - 1 eNB, Many UEs
 - Many eNBs, Many UEs
- Scheduling Algorithm (CPS-V2X)
 - Based on PKRS, CPS, and UCS
 - Adjusted for dynamic node configurations

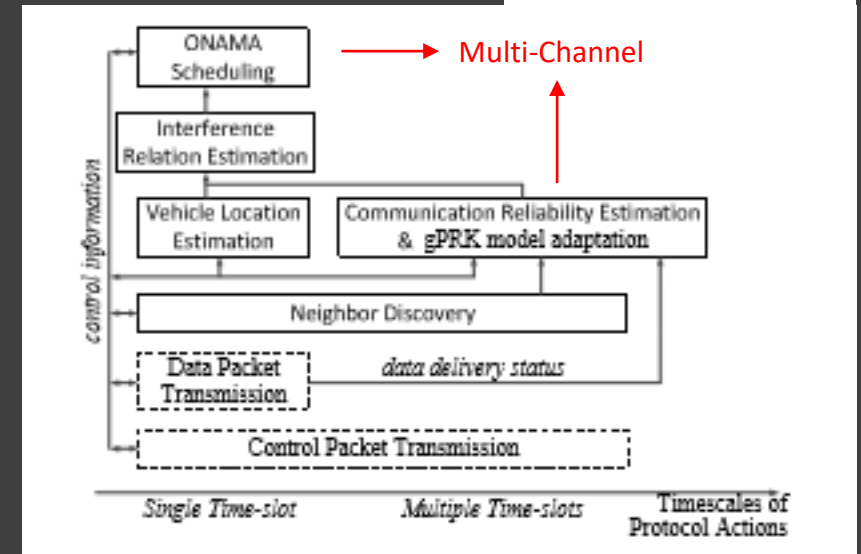
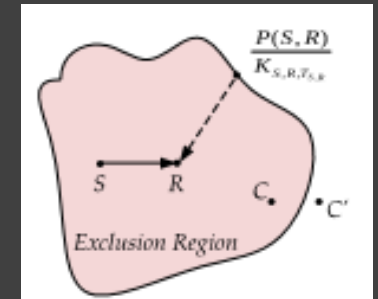
Detailed Design – OAI Simulation

- Configure and run UE and eNB on one server
 - UE – mobile phone, tablet, modem
 - eNB - Base stations connected to the network that communicate wirelessly with mobile handsets
- Configure and run EPC on another server
 - Network Access Controls, Packet Routing, Mobility Management, Security



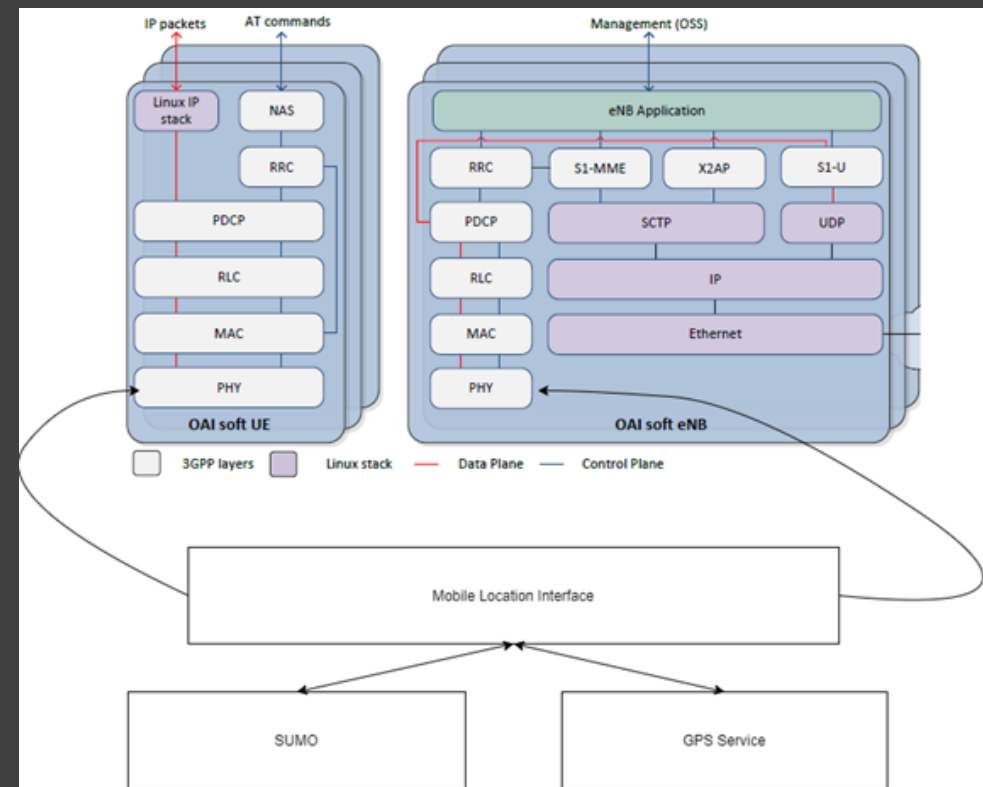
Detailed Design – CPS-V2X Algorithm

- Incorporates characteristics of three existing algorithms:
 - PRKS (Physical-Ratio-K Scheduling)
 - Interference control, High Reliability
 - CPS-V2V (Cyber-Physical Scheduling Vehicle to Vehicle)
 - Vehicle Mobility, Uses PRKS
 - UCS (Unified Cellular Scheduling)
 - Multi-Channel
- Will be implemented in MAC layer of OAI
 - Collision Detection, uses data from PHY layer



Detailed Design – OAI/SUMO Integration

- SUMO integrated with OAI at PHYS layer
- Mobile Location Interface between SUMO and OAI
 - Allows future integration of hardware-based GPS system instead of OAI
- PHYS layer returns location data to MAC layer through control plane



Detailed Design – Jenkins



HW/SW/Technology Platforms Used

- OAI
 - Open Air Interface
 - Open source 5g network simulator/emulator/test bench
 - Highly optimized, written in C
- SUMO
 - Simulation of Urban Mobility
 - Open source, highly customizable traffic simulator
 - Include an API to get vehicle positioning data
- Jenkins
 - CI/CD
 - Pipelined build jobs for regression testing

Test Plan

- Configure simulation test bed
 - 1 UE, 1 eNB to start with
 - Use nFAPI to facilitate communication
 - Network functional API
 - Connects Physical Network to Virtual Network (Layer 1 to Layer 2)
 - Modify number of eNBs/UEs
 - Incorporate SUMO traffic data
- Implement new scheduling algorithm
 - Does the simulation test bed break?
 - Unit tests using simple UE/eNB sets
 - Reliability, throughput, and latency measurements

Prototype Implementations

- Currently running 1 UE and 1 eNB with an EPC
 - Have configuration files for multiple UE/eNBs created
- Built a C client to support SUMO API
 - Used for traffic data
- Newest version of OAI running on server
 - Updated from v.5.2 -> 1.2.0 -> 1.2.1

Engineering Standards and Design Practices

- 3GPP
- E-UTRAN
- EURECOM
- IEEE
- Continuous Integration, Continuous Development (CI/CD)
- Coding Best-Practices

Conclusion

- Progress was slower than expected
 - Difficult learning curve with OAI
 - Switched to a newer version of OAI part of the way through
 - Able to run with one eNB and one UE
- Was able to get SUMO working
 - Created a program to grab information from SUMO and send it to a C program using sockets
 - Meant to be integrated with OAI to send data straight from SUMO to OAI

Conclusion

- The new scheduling algorithm has gotten to the point of carrying out the preliminary design
 - Not yet integrated into OAI
 - Have begun writing pseudocode and verifying functions
- Configured network simulation with one eNB/UE and EPC
 - eNB/UE run on the same server
 - EPC runs on separate server on same local network
 - Wrote documentation on how to recreate our configuration