|  |
| --- |
| **12/15** |
| **See the comments and add more description in the highlighted parts** |

Performance Analysis with OAI and NS3

**Software Requirements Specification**

Version 1.0



**Group Id:** F200161379 (BC170402458)

**Supervisor Name :** Arif Husen

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date (dd/mm/yyyy)** | **Version** | **Description** | **Author** |
| 29/11/2020 | 1.0 | This project gets you familiarized with the 5G and basic principles to create test environment to measure general performance parameters such as maximum achievable throughput, latency, packet loss with different mobility patterns and speeds. | BC170402458 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. [Scope (of the project)](#gjdgxs)
2. [Functional Requirements Non Functional requirements](#30j0zll)
3. [Use Case Diagram](#1fob9te)
4. [Usage Scenarios](#3znysh7)
5. [Adopted Methodology](#2et92p0)
6. [Work Plan (Use MS Project to create Schedule/Work Plan)](#tyjcwt)

**SRS Document**

**Scope of Project:**

Technology is evolving at a very fast speed. It was just a few years ago when 4G was introduced and now we see 4G everywhere. But as the technology grows the need to communicate faster and efficiently increases, so now we need faster ways to communicate with less delay than 4G. This is where 5G comes into place.

We will research 5G , its architecture and its working. We will create a test environment to analyze 5G performance parameters such as maximum achievable throughput, latency, packet loss etc and show the real potential of 5G.

The testing environment will be built using ns3, OAI and a web server.

NS3 is a network simulator which will help us in creating various virtual nodes ,user devices and simulate communication between them using our chosen internet stack and protocols. We will use NS3 to set up mobility patterns.

OpenAirInterface (OAI) is an open-source LTE ecosystem. It offers a full protocol stack of 3GPP standard both in E-UTRAN and EPC.It also provides some built-in tools such as highly realistic emulation modes, soft monitoring and debugging tools, protocol analyzer and performance profiler. We will use OAI to create a core network of 5G. We will create 4 eNodesBs and one EPC and mobile users.

Mobile users will connect with eNodesBs which will connect to EPC and pass the traffic through the web server. This will be our simple connectivity scenario through which will analyze the performance of 5G.

**Functional and non-Functional Requirements:**

**Functional Requirements:**

1. A Presentation on 5G and OAI

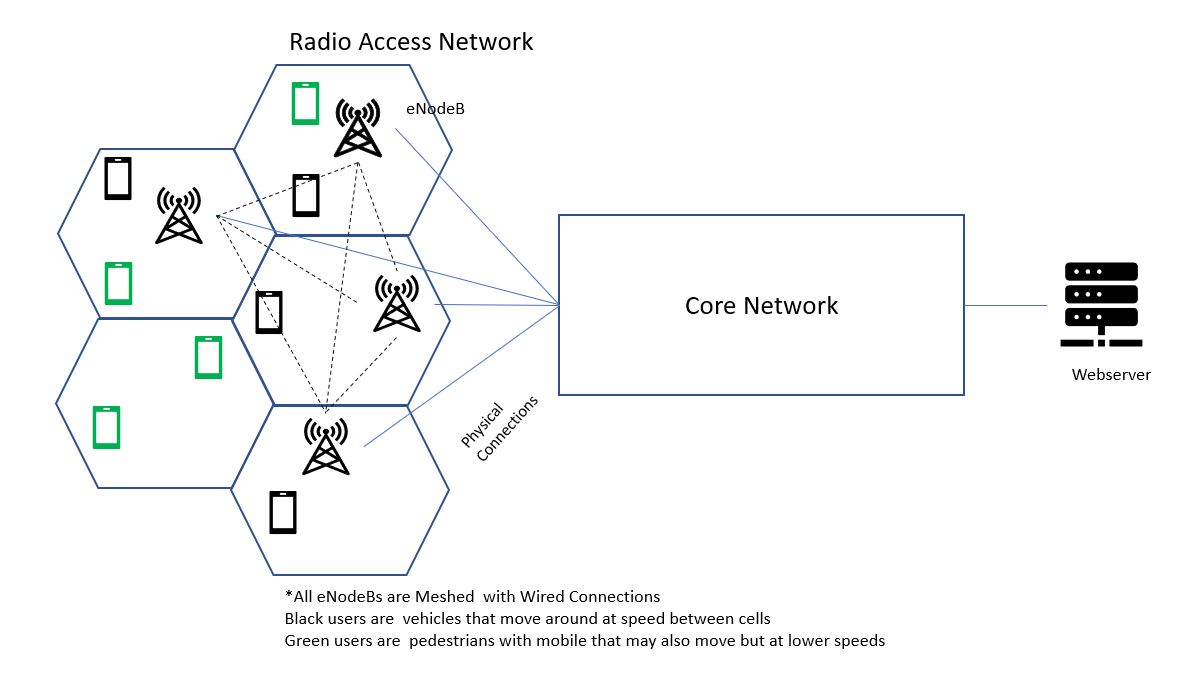
2. Setup suitable Linux environment with containers support

3. Install OAI stack and create 4 eNodeBs and one EPC

4. Create a set of mobile users (at least 8) and a webserver a. Connect Web Server with EPC b. Mobile users should associate with B

5. Setup Mobility Patterns using NS3

6. Create a simple connectivity scenario as shown in figure 1.



7. Measure Performance parameters (throughput, Latency, Jitter, Mobility)

**Non-Functional Requirements:**

**Performance Requirements:**

5G simulation may create an ideal scenario for measuring throughput, jitter, mobility and latency in real life there may be a very small variation due to some other elements.

**Space Requirements:**

There should be enough space in your system to install all softwares like ns3, OAI, dockers and a web server. Estimated space requirement is 40 GB.

**Reliability Requirements:**

The simulation environment should not produce incorrect results and the measured parameter output should be close to the output of real world scenario.

**Portability Requirements:**

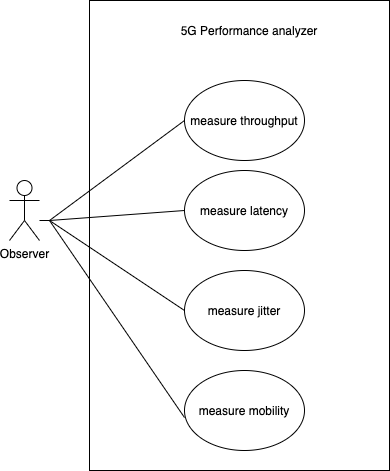
The simulation environment would only run on Linux and preferably ubuntu 18.0 or 14.0 .

**Implementation Requirements:**

The following softwares should be installed and integrated to make the simulation environment:

* Linux ubuntu
* Dockers
* NS3
* OAI (open air interface)

**Use Case Diagram(s):**



**Usage Scenarios:**

**Use Case 1:**

|  |  |
| --- | --- |
| **Use Case Title** | Measure Throughput |
| **Use Case ID** | UC-1 |
| **Actor** | Observer/Tester |
| **Description** | User will send data via 5G core network and get throughput in result |
| **Alternative Paths** | nil |
| **Pre-Conditions** | Nodes must be connected to each other properly and can communicate with each other. |
| **Post conditions** | Observer will get the results of throughput. |
| **Author** | Muhammad Usman |
| **Exceptions** | nil |

**Use Case 2:**

|  |  |
| --- | --- |
| **Use Case Title** | Measure latency |
| **Use Case ID** | UC-2 |
| **Actor** | Observer/Tester |
| **Description** | User will send data via 5G core network and get latency in result |
| **Alternative Paths** | nil |
| **Pre-Conditions** | Nodes must be connected to each other properly and can communicate with each other. |
| **Post conditions** | Observer will get the results of latency. |
| **Author** | Muhammad Usman |
| **Exceptions** | nil |

**Use Case 3:**

|  |  |
| --- | --- |
| **Use Case Title** | Measure Jitter |
| **Use Case ID** | UC-3 |
| **Actor** | Observer/Tester |
| **Description** | User will send data via 5G core network and get jitter in result |
| **Alternative Paths** | nil |
| **Pre-Conditions** | Nodes must be connected to each other properly and can communicate with each other. |
| **Post conditions** | Observer will get the results of jitter. |
| **Author** | Muhammad Usman |
| **Exceptions** | nil |

**Use Case 4 :**

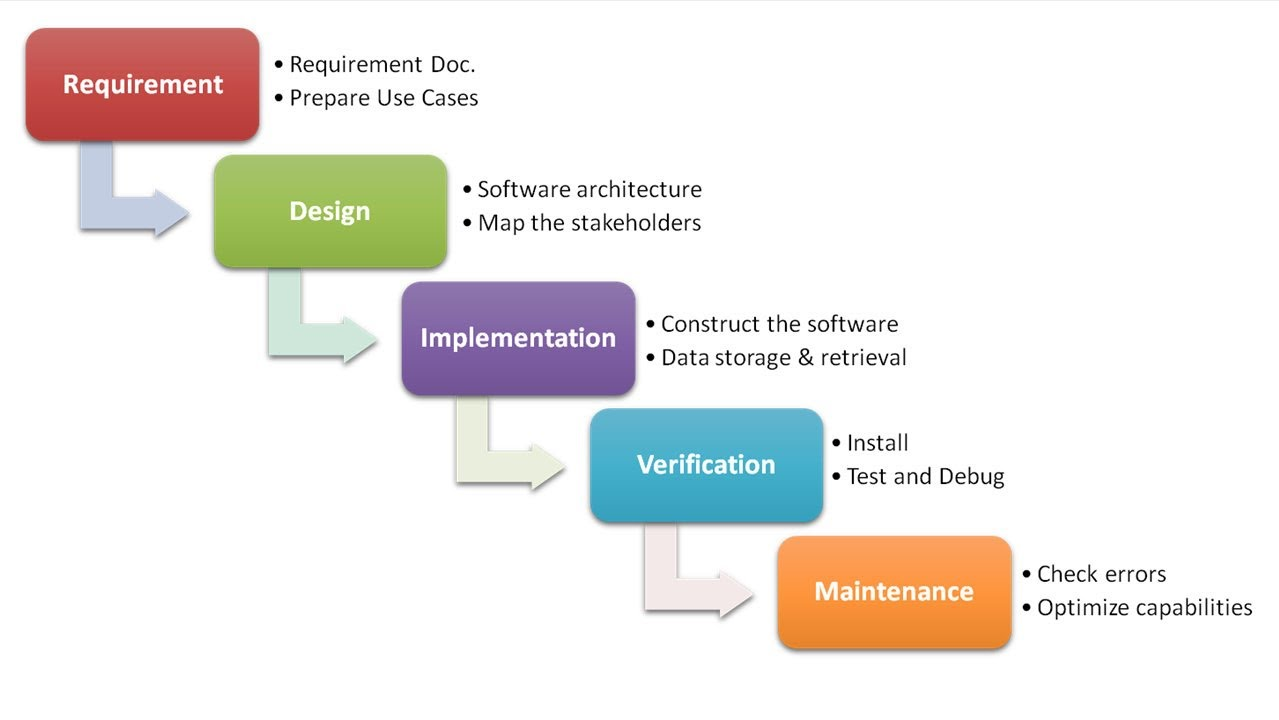
|  |  |
| --- | --- |
| **Use Case Title** | Measure Mobility |
| **Use Case ID** | UC-4 |
| **Actor** | Observer/Tester |
| **Description** | User will send data via 5G core network and get mobility in result |
| **Alternative Paths** | nil |
| **Pre-Conditions** | Nodes must be connected to each other properly and can communicate with each other. |
| **Post conditions** | Observer will get the results of mobility. |
| **Author** | Muhammad Usman |
| **Exceptions** | nil |

**Adopted Methodology**

The methodology that we have adopted for this Project is VU Process Model which is a combination of two very well-known and easy to use models i.e. Spiral Model and Waterfall Model. The idea of this model is to get the benefits of both these models. Essentially, Waterfall Model is a framework for software development in which development proceeds sequentially through a series of phases, starting with system requirements analysis and leading up to product release and maintenance, whereas the main key features of spiral model is risk management at regular stages in the entire software development cycle. The two models are explained below:

**Waterfall Model:**

As the Waterfall Model was the first Process Model to be introduced, it is very simple to use and understand. In this model the whole process of software development is divided into separate phases and all these phases are executed in a sequential manner i.e. each phase must be completed for the next phase to begin. In the waterfall model the progress of software development is flowing steadily downwards through the phases. These phases are described herein below:

 (Fig.0.3 Waterfall Model)

• Requirements: In the first phase system specification is developed by collecting and studying the functional and non-functional requirements, functions and purposes of the final product.

• System and Software Design: The gathered requirement specifications from the previous phase are studied here and system design is prepared in this phase. System design helps in specifying hardware and software systems and it establishes overall system architecture.

• Implementation and Unit Testing: In this phase the system is first developed in small programs called units with inputs from system design from the previous phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

• Integration and System Testing: All the units developed in the Implementation Phase are integrated into a system after testing of each unit. The designed software is put to constant software testing to find if there is any error and ensure that client does not face any problem during installation of the software.

• Operation and Maintenance: Finally, the system is installed and put into practical use. Maintenance involves correcting errors which were not discovered in the earlier stages or improving performance. Modifications are also made as and if new requirements are discovered.

**Spiral Model:**

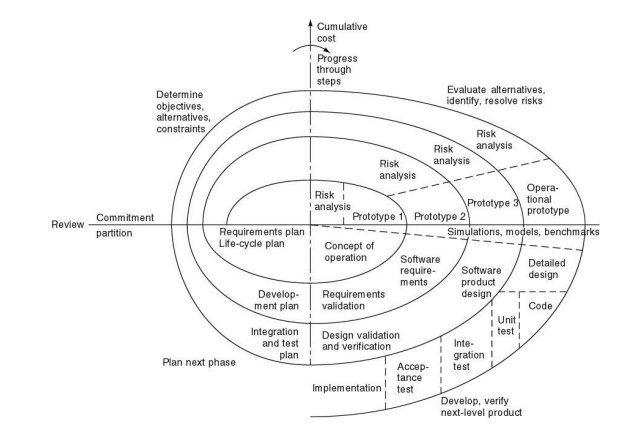
This model overcomes the cons of the waterfall model by adding a very high emphasis on risk analysis to its phase. Spiral Model contains 4 phases i.e. Planning, Risk Analysis, Engineering and Evaluation and these phases are iteratively followed by one after another in order to avoid maximum risk during software development. In other words, given tasks move through these four phases till the entire software is built.

Planning: Requirements are studied and gathered in this phase. It includes estimating the cost, schedule and resources for the spirals.

Risk Analysis: Risk Analysis includes identifying, estimating, and monitoring technical feasibility and management risks, such as schedule slippage and cost overrun. Once the risks are identified, risk mitigation strategy is planned and finalized.

Engineering: Actual development and testing of the software takes place in this phase. It includes testing, coding and deploying software at the customer site.

Evaluation: After testing the build, at the first phase, the customer evaluates the product and provides feedback.



(Fig.0.3 Spiral Model)

**Work Plan (Use MS Project to create Schedule/Work Plan)**

