Project on:

6G Diagnostic Monitor

-by Nirzari

A REPORT

ON

"6G DIAGNOSTIC MONITOR"

BY

Nirzari Kalpesh Shah **2019A8PS0576G**

At

SAMSUNG

Samsung R & D Institute India – Bangalore
(Beyond 5G Team)

A Practice School-II Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

(December, 2022)

A REPORT

ON

"6G DIAGNOSTIC MONITOR"

BY

Nirzari Kalpesh Shah **2019A8PS0576G**

Prepared in partial fulfillment of the
Practice School-II

At

SAMSUNG

Samsung R & D Institute India – Bangalore
(Beyond 5G Team)

A Practice School-II Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

(December, 2022)

Acknowledgements ...

I am highly indebted to my reporting manager Mr. Ramanathan Palaniappan, Architect, Beyond 5G Team, SRI Bangalore for providing me with the opportunity to work on this project. I am grateful to Dr. Lucy J. Gudino, Associate Professor, BITS Pilani, for her constant guidance and support throughout PS II program. I owe my sincere gratitude to them for their constant guidance, invaluable suggestions and constructive feedback.

I would also like to express my sincere gratitude to the Practice School Division for giving us the opportunity to undergo an internship in such a reputed company and also to Samsung R&D Institute - Bangalore for taking me under its fold.

My sincere gratitude to Shruti Joshi, Professional, Talent Acquisition, SRI-Bangalore, for smoothly coordinating the PS-II internship program and helping me with all organization related matters.

In preparation of our project, I was helped and guided by my peers and mentors, among the others who utilized their time and energy and I would like to expand my gratitude to all those who have guided me in this project.

Abstract Sheet ...

BIRLA INSTITUTE OF TECHNOLOGY AND SCENCE PILANI

(RAJASTHAN)

Practice School Division

Station: Samsung R and D, Modem **Centre:** Bangalore

Duration: 6 months **Date of Start:** 13th July, 2022

Date of Submission: 5th December, 2022

Title of the Project: 6G Diagnostic Monitor

Student ID/ Name/ Discipline:

2019A8PS0576G - Nirzari Kalpesh Shah -Electronics and Instrumentation

Expert's Name/ Designation: Mr. Ramanathan Palaniappan - Architect, Beyond 5G, SRIB

PS Faculty's Name: Dr. Lucy J. Gudino - Associate Professor, Computer Science and Information Systems group at Bits Pilani

Key Words: Python, PythonQt/PyQt, Qt Designer, Anaconda, Spyder, GUI

Project Areas: 6G Network, Software Development

Abstract: The project aims to provide users GUI named 6G Diagnostic Monitor. The report contains detailed information on the tools, technologies and methodologies involved in the successful completion of this project and gives the reader an in-depth understanding of all its various aspects, from design to execution to future scope.

Signature of Student *Signature of PS Faculty*

Date: 5th December,2022 *Date: 5th December,2022*

Response Option Sheet ...

BIRLA INSTITUTE OF TECHNOLOGY AND SCENCE PILANI

Practice School Division

RESPONSE OPTION SHEET

Station: Samsung R and D, Modem **Centre:** Bangalore

ID No & Name: 2019A8PS0576G – Nirzari Kalpesh Shah

Title of the Project: 6G Diagnostic Monitor

The usefulness of the project to the on-campus courses of study in various disciplines. Project should be scrutinized keeping in view the following response options.

Refer Bulletin for Course No. and Course Name.

Code No.	Response Option	Course No.(s) & Name
1.	A new course can be designed out of this project.	-
2.	The project can help modification of the course content of some of the existing Courses	-
3.	The project can be used directly in some of the existing Compulsory Discipline Courses (CDC)/ Discipline Courses Other than Compulsory (DCOC)/ Emerging Area (EA), etc. Courses	-
4.	The project can be used in preparatory courses like Analysis and Application Oriented Courses (AAOC)/ Engineering Science (ES)/ Technical Art (TA) and Core Courses.	-
5.	This project cannot come under any of the above-mentioned options as it relates to the professional work of the host organization.	-

Signature of Student *Signature of PS Faculty*

Date: 5th December,2022 *Date: 5th December,2022*

Table of Contents ...

Company Profile ...

1.1 Foundation

Samsung was founded as a grocery trading store on March 1, 1938, by Lee Byung-Chull. He started his business in Taegu, Korea, trading noodles and other goods produced in and around the city and exporting them to China and its provinces. (The company name, Samsung, came from the Korean for "three stars.") After the Korean War, Lee expanded his business into textiles and opened the largest woolen mill in Korea. He focused heavily on industrialization with the goal of helping his country redevelop itself after the war. During that period his business benefited from the new protectionist policies adopted by the Korean government, whose aim was to help large domestic conglomerates (chaebol) by shielding them from competition and providing them easy financing. In the late 1950s the company acquired three of Korea's largest commercial banks as well as an insurance company and firms that made cement and fertilizer. Samsung in the 1960s acquired more insurance companies as well as an oil refinery, a nylon company, and a department store.

During the 1970s the company expanded its textile-manufacturing processes to cover the full line of production—from raw materials all the way to the end product—to better compete in the textile industry. New subsidiaries such as Samsung Heavy Industries, Samsung Shipbuilding, and Samsung Precision Company (Samsung Techwin) were established. Also, during the same period, the company started to invest in the heavy, chemical, and petrochemical industries, providing the company a promising growth path.

Figure 1.1 Company Logo[1]

Samsung first entered the electronics industry in 1969 with several electronics-focused divisions. Their first products were black-and-white televisions. During the 1970s the company began to export home electronics products overseas.

1.2 Growth in Last Years

The 2000s witnessed the birth of Samsung's Galaxy smartphone series, which quickly not only became the company's most-praised products but also were among the best-selling smartphones in the world. Samsung also supplied the microprocessors for Apple's earliest iPhone models and was one of the largest microprocessor manufacturers in the world in the late 20th and early 21st centuries. Since 2006 the company has been the top-selling global manufacturer of televisions. Beginning in 2010, the Galaxy series expanded to tablet computers with the introduction of the Galaxy Tab and in 2013 to smartwatches with the introduction of the Galaxy Gear. Samsung introduced a foldable smartphone, the Galaxy Fold, in 2019.

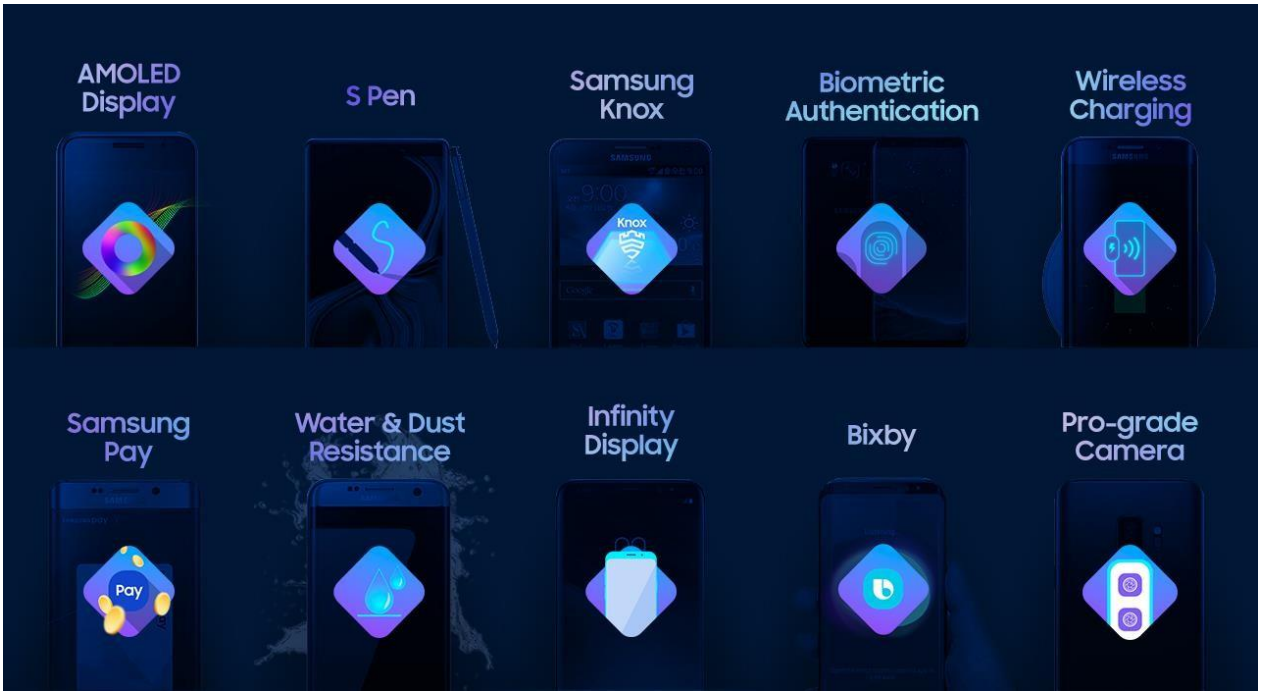


Figure 1.2. Innovations Done in Last Decade[2]

1.3 Acquisitions

Samsung has made 38 acquisitions and 51 investments. The company has spent over $ 17.53B for the acquisitions. Samsung has invested in multiple sectors such as Display Technology, Energy Efficiency Tech, Imaging Diagnostics and more. Notable acquisitions by Samsung are Harman International for 8 Billion USD, SmartThings for 200 Million USD, LoopPay (now Samsung Pay) for approximate 250 Million USD and Novaled for 347 Million USD.

1.4 Business

- Brand Value: $91.3 billion
- Global Revenue: $218.17 billion
- Global Smartphone Shipments: 292.3 million
- LCD TV unit shipments: 40.8 million
- Number of employees: 320671
- Samsung's share in the European smartphone market: 40.6%
- Samsung's share in the US smartphone market: 26%
- Research and Development contribution: $14 billion
- Number of countries in which Samsung has employees: 73
- 1st Samsung app to reach 1 billion downloads: Samsung Push Service
- The word Samsung means "three stars." The word "three" represents "big, numerous and powerful."

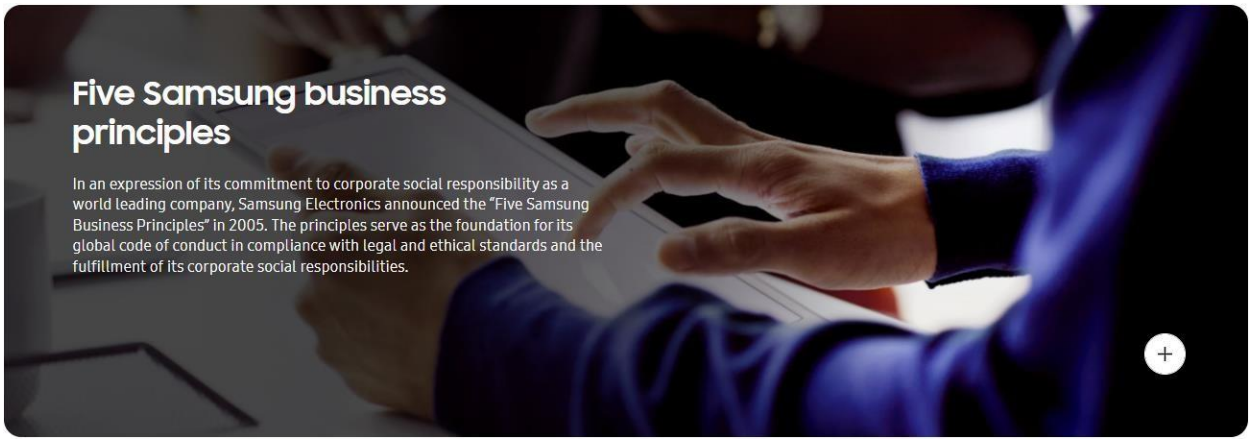


Figure 1.3. Business Principles of Samsung[3]

In August 2019, the firm has a 31.47% market share as a mobile vendor. In comparison to its competitors Apple (22.17%), Huawei (9.02%), Xiaomi (8.38%), Oppo (4.73%) and unknown (3.69%), the company is succeeding well.

1.5 Products of Samsung

The business model of Samsung is such that it is the highest contributor to the GDP (Gross Domestic Product) of South Korea. The organization provides goods and services which help the common man in South Korea to take all important decisions.

The commodities by Samsung are as follows:

- Electronics
- Automotives
- Apparels
- Chemicals
- Home appliances
- Medical equipment
- Consumer electronics

The company is diverse in the products and services they administer to. Samsung is famous for its smartphones. But, the reach of the company is beyond just a phone. Although it is the most vital part, the business model of Samsung is highly dependent on the various businesses it indulges in.

It includes the following services

- High calibre healthcare
- Life insurance
- Purchasing medicines
- Pursuing studies in universities
- Vacations
- Electronics
- Engineering

Industrial affiliates of Samsung include-

- Samsung Electronics which is the largest IT Company, consumer electronics maker and chipmaker measured in the world
- Samsung Heavy Industries which is 2nd largest shipbuilder in the world
- Samsung Engineering and Samsung C&T which are respective 13th and 36th largest construction companies in the world
- Samsung Life Insurance which is the 14th largest life insurance company in the world
- Samsung Everland which is the operator of Everland Resort that is also the oldest theme park in South Korea
- Cheil Worldwide which is known as the 15th largest advertising agency in the

world Other products associated with the Samsung Business Model are-

- Consumer electronics
- Electronic components
- Semiconductors
- Solid-state drives
- DRAM
- Telecommunications equipment

1.6 More about the Company

1.6.1 Samsung's Mission and Approach

i. Supporting people to be their best

On the basis of human resources & technologies

a. Make the extension of human resource development and technical superiority with management principles.

b. Increase the synergy effect of the whole management system through human resources and technologies.

ii. Our internal goals

Create the best products and services

a. Create the products and services that give customers the best satisfaction.

b. Retain the 1st position in the world in the same line of business

iii. Beyond Samsung

And contribute to society

a. Contribute for common interests and a rich life.

b. Perform the mission statement by a member of the community.

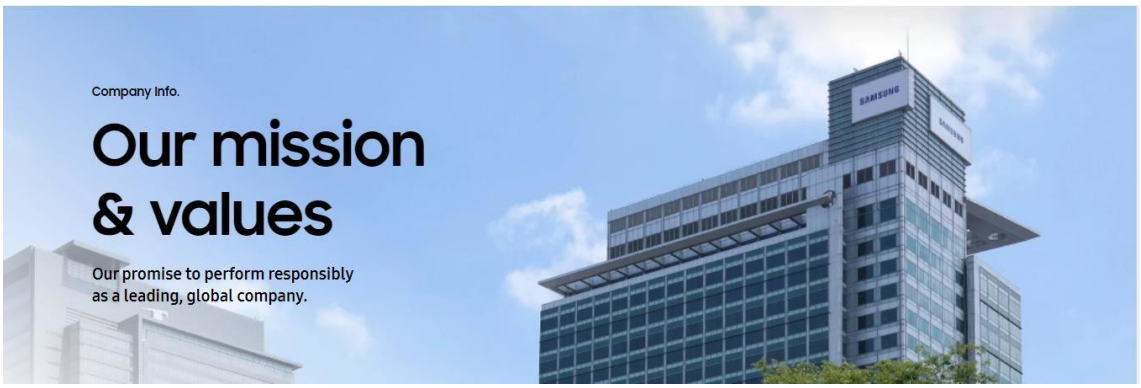


Figure 1.4. Company's Mission & Values [3]

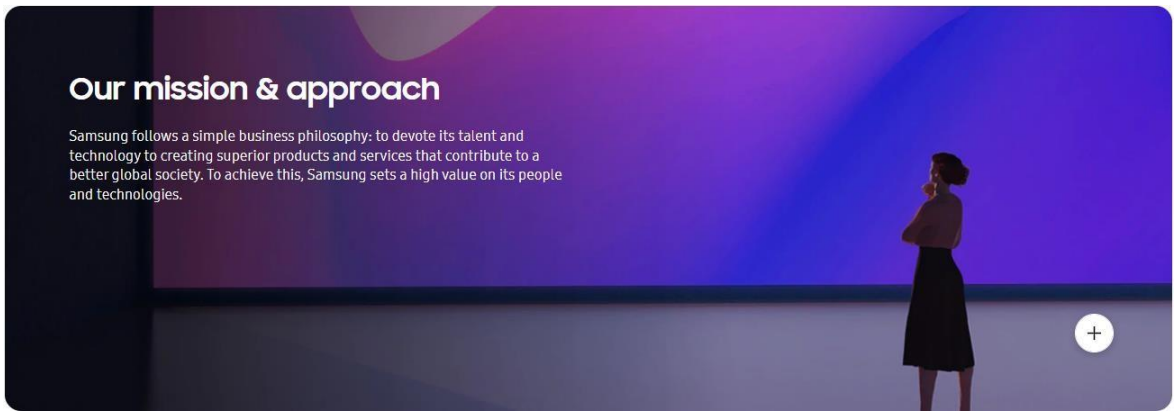


Figure 1.5. Company's Mission & Approach [3]

1.6.2 About CEO's

Han, Jong Hee

- Vice Chairman & CEO [2022~Present]
- Vice Chairman & Head, Device eXperience [2021~Present]
- President & Head, Visual Display Business [2017~2021]
- Head, R&D Team, Visual Display Business [2015~2017]
- Head, R&D Office, Visual Display Business [2013~2015]
- Head, Product R&D Team, Visual Display Business [2011~2013]

Kyung, Kye Hyun

- President & CEO [2022~Present]
- President & Head, Device Solutions [2021~Present]
- CEO, Samsung Electro-Mechanics Co., Ltd. [2020~2021]
- Head, Solution Product & Development, Memory Business [2018~2020]
- Head, Flash Product & Development, Memory Business [2015~2018]
- Head, Flash Design Team, Memory Business [2011~2015]

Figure 1.6. Company's CEO's [3]

1.6.3 Company's Ethics

As our business spans across numerous countries around the globe, we recognise and analyse differences in laws, regulations, and practices in respective countries while conducting business in lawful and ethical manner. We also develop and implement global personal data security policies to respect the privacy of our customers and employees and to protect their personal data. Our endeavours to systematically manage compliance and ethical risks are driven by the Samsung Global Code of Conduct and the Business Conduct Guidelines that guide all our employees in taking action and making value judgment. Our

Compliance Team, previously under the Legal Office, now reports directly to the CEO and the head of Compliance Team attends all board meetings to support important decisions made by the board. We operate dedicated organizations at each business division and overseas regional offices to manage compliance issues pertaining to each business and region.

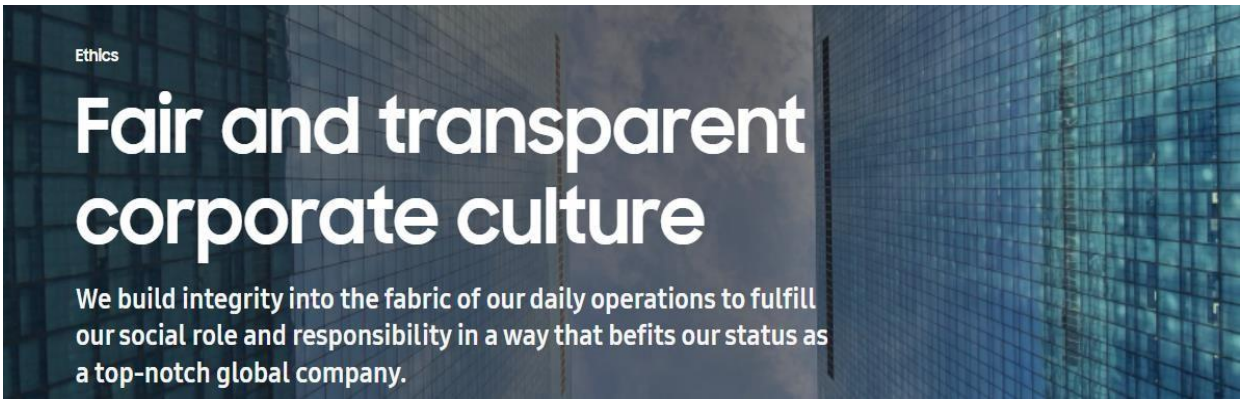


Figure 1.7. Company's Ethics [4]

We disclose our Global Code of Conduct ('Samsung Business Principles') to our suppliers, customers, and other external stakeholders as well as to our employees through our ethics management website, and provide a channel to report on any violation of ethical standards. Furthermore, the 'Employee Business Conduct Guidelines' that serve as the ethical standards for our employees are translated and available in a total of 15 languages (including Korean) and uploaded on our in-house intranet. Relevant details are disseminated and shared among all our employees around the globe through collective, online, and audio/visual training offered at least annually if not more. Separate 'Business Guidelines' are also provided to our suppliers in order to establish transparent business practices.

Figure 1.8. Compliance Management Process [4]

1.7 About SRIB Campus

Samsung R&D Institute India-Bangalore (SRI-B) is the largest R&D Centre outside of South Korea and a key innovation hub in the Samsung group. With the best of talent from India and overseas, our focus is on creating cutting edge technologies across multiple areas of Samsung's business, that transform experiences of users both globally, as well as in local markets. The specific purpose of SRI-B in the Samsung family is twofold: to create USPs for global flagship devices with by creating significant advancements in Modem, Multimedia, AI, Internet of Things, and to make for India by catering to the specific needs of Indian consumers. At SRI-B, employees not only have an opportunity to work on areas involving pure research but also carry it forward as proof of concept towards Advanced Development, by seeing the research being implemented.

The Samsung R&D Institute In Bangalore (SRI-B) has established five Centers of Excellence (CoEs) with the focus areas of Communication, Camera and Multimedia, On-Device AI, IoT and Services. SRI-B has experience executing projects from the research to market stage in each of these areas, and makes impactful contributions to Samsung

product lines on the backs of these CoEs every year.

At the Communication CoE, SRI-B has dedicated teams working on mobile terminals, network RAN/core development and wireless standards. Strong synergy between these teams has resulted in the establishment of end-to-end domain expertise. In addition to this, we have recently seeded advanced communication research in a bid to make impactful contributions to Beyond 5G and 6G evolution.

Figure 1.9. SRIB Campus [5]

Introduction ...

2.1 About Project

Name of the Project – 6G Diagnosis Tool

6G (sixth-generation wireless) is the successor to 5G cellular technology. 6G networks will be able to use higher frequencies than 5G networks and provide substantially higher capacity and much lower latency. One of the goals of the 6G internet is to support one microsecond latency communications. This is 1,000 times faster – or 1/1000th the latency – than one millisecond throughput.

The 6G technology market is expected to facilitate large improvements in the areas of imaging, presence technology and location awareness. Working in conjunction with artificial intelligence (AI), the 6G computational infrastructure will be able to identify the best place for computing to occur; this includes decisions about data storage, processing and sharing.

It is important to note that 6G is not yet a functioning technology. While some vendors are investing in the next-generation wireless standard, industry specifications for 6G-enabled network products remain years away.

In this project, we are developing a **Diagnosis Tool**. It is a desktop widget mainly developed to collect and display metrics (like Rx throughput, Tx throughput, BLER) remotely. It is being developed to test **6G**. It is basically a GUI which will display various parameters to test 6G performance. Two mode is being developed: Engineering View and Demo View.

We are using PyQt5 for developing this GUI because PyQt5 is one of the most used modules in building GUI apps in Python, and that's due to its simplicity as you will see. Another great feature that encourages developers to use PyQt5 is the PyQt5 designer, which makes it so easy to develop complex GUI apps in a short time. You just drag your widgets to build your form.

Engineering View:

- Displays logs from TBNB and TBUE
- Graphically displays L1/L2/L3 Modem metrics.
- Displays Running Log Traces, decode RRC control messages.

Demo View:

- Graphical Critical/Important metrics display
- Speed, BLER, Tx/Rx Power, Beam-pairs, etc.
- Offline Analysis of Log Traces
- Provides offline tool to analyse the logs.
- Provides Graphical and Statistical Analysis.

Technology Used: PyQt5

- PyQt5 is a blend of Python programming language and the Qt library.
- Qt is an open source and cross platform. Uses the system resources to draw windows, controls, etc., so the application will get a native look.
- Python and Qt has a rich set of APIs. [Needed for Graphs, Charts, Sockets, Animations, etc.]
- Python Bindings allow us to call functions and pass data from Python to C or C++, letting us to take advantage of the strengths of both languages.

DM Client Application Requirements:

- Shall Display Graphically: L1, L2, and L3 modem metrics from both TBNB and TBUE
- Shall provide option to select/de-select(filter) metrics for graphical display.

Log Traces:

- Shall display online traces from logs generated by TBNB and TBUE.
- Shall provide option to select/de-select(filter) logs based on TBNB/TBUE modules [module level filter]
- Shall provide option to select/de-select(filter) logs based on Verbosity [Log debug level filter].

RRC Message Decoding:

- Shall display online traces from RRC message logs from TBNB and TBUE.
- When any RRC message selected in GUI, shall decode the RRC control message dumps and display as per 'ASN'.

Remote Debug when Crash/Core-Dump:

- Shall provide debug interface remotely to debug the crash scenario.
- Offline analysis of logs [Analyse the logs statically after a session of 6G].

2.1.1 Different Components in the GUI

2.1.1.1 Add/Delete Buttons

These buttons are used to add or delete Ues to the main UI. Whenever either of these button is pressed, a pop window appears which have three blank rows i.e., UE Name, IP Address and Port Number and then that UE will be added in the main window and different parameters of that UE are shown.

2.1.1.2 Settings Button

Settings button is used to set buffer value for the graph and to clear all profiles (Ues) from main window.

Figure 2.1. Add, Delete and Settings Button Outline (Made on Figma)

2.1.1.3 Animation Widget

Animation widget is used to show the number of UEs connected to a NB at any point of time. It also depicts about the connection status of UE i.e., Connecting, Connected, Disconnected, and Server Down.

2.1.1.4 Metrics Widget

Metrics Widget is the most important component of the GUI as it contains all the widgets that shows different parameters. It includes Throughput Widget, Throughput Graph Widget, BLER Percentage Widget, BLER Graph Widget and Constellation Widget.

2.1.1.5 Speedometer

Speedometer shows the sum of Throughput of all the UEs added in the main GUI.

Figure 2.2. Speedometer

2.1.1.6 Constellation Matrix

A constellation diagram is a representation of a signal modulated by a digital modulation scheme such as quadrature amplitude modulation or phase-shift keying. It displays the signal as a two-dimensional *xy*-plane scatter diagram in the complex plane at symbol sampling instants. The angle of a point, measured counter clockwise from the horizontal axis, represents the phase shift of the carrier wave from a reference phase. The distance of a point from the origin represents a measure of the amplitude or power of the signal.

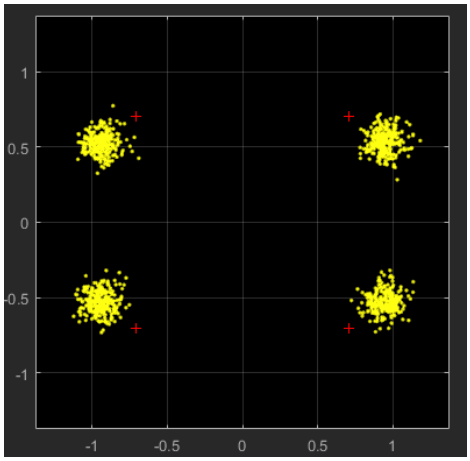


Figure 2.3. Constellation

2.1.2 Enhancements done on the Project

2.1.2.1 SysTray Implementation

When the close button on the GUI is pushed, it is minimized in SysTray, where it continues to gather metrics, and when we maximize it, all previously inserted UEs should remain in the GUI. This improves the user experience because, as a Diagnostic Tool, it will be minimized the majority of the time.

2.1.2.2 Pop Up Window fo Constellation Matrix

Because the space available for displaying the Constellation Matrix is limited, a new feature has been added. When we click on a Constellation Matrix, a pop-up window with an enlarged image of the Constellation Matrix appears, making it simpler to see the many dots in the Constellation Matrix.

2.1.2.3 Spline Interpolation

Because the metrics are not consistent, the graph is uneven and has sharp edges. To smooth the graph curve, we 13tilized Spline Interpolation.

2.1.2.4 Creation of our own Python Library for Testing

We had to manually input a number of values to the Constellation Matrix for testing, which made the process uneven, therefore we designed our own python library that will automatically deliver different values to the constellation matrix at time intervals.

2.1.2.5 Addition of Floating Widget in the main UI

Previously Add/Delete and Settings button are always shown on the main UI, but it is only required when we want to add or delete UE or we want to change some setting. So we have a implemented Floating Window, there is a arrow button when it is clicked then only the add/delete and settings button are shown. We have used QAnimation and QParallelAnimationGroup.

2.1.2.6 Backend Part of the Project

We used QThread for getting data from the server and pass it to the main thread. PyQt graphical user interface (GUI) applications have a **main thread** of execution that runs the event loop and GUI. If we launch a **long-running task** in this thread, then our GUI will freeze until the task terminates. During that time, the user won't be able to interact with the application, resulting in a bad user experience. Luckily, PyQt's QThread class allows you to work around this issue.

The application's GUI freezes as a result of a blocked main thread. The main thread is busy processing a long-running task and doesn't immediately respond to the user's actions. This is an annoying behaviour because the user doesn't know for sure if the application is working correctly or if it's crashed.
Fortunately, there are some techniques you can use to work around this issue. A commonly used solution is to run your long-running task outside of the application's main thread using a **worker thread**.

2.2 About Team

Name of the Team – Beyond 5G

It is one of the most important team of SRIB (Samsung Bangalore). Beyond 5G works on technology which enhances the 5G experience for Samsung customers.

Firstly, this team specializes in radio, data networking protocols and embedded modem system software. Beyond 5G craft the 5G radio experience for different markets around the world. Our team is engaged in the product development of 5G mobile terminals for a range of world-wide markets.
Secondly, this team is engaged in advanced research and development surrounding communication protocols. Some of this work makes it into Samsung products as differentiating features and solutions. The rest of this team's work is aimed at creating standards and implementation IP (intellectual property) pertaining to Beyond 5G and 6G systems.

Background ...

Various concepts of Python and C++ are used for ground work. PyQt5 is used for development of GUI. PyQt is a blend of Python programming language and the Qt library. Qt is set of cross- platform C++ libraries that implement high-level APIs for accessing many aspects of modern desktop and mobile systems. These include location and positioning services, multimedia, NFC and Bluetooth connectivity, a Chromium based web browser, as well as traditional UI development.

PyQt5 is a comprehensive set of Python bindings for Qt v5. It is implemented as more than 35 extension modules and enables Python to be used as an alternative application development language to C++ on all supported platforms including iOS and Android.

PyQt5 may also be embedded in C++ based applications to allow users of those applications to configure or enhance the functionality of those applications.

3.1 Why PyQt?

PyQt brings together the Qt C++ cross-platform application framework and the cross-platform interpreted language Python.

Qt is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets.

Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create re-usable software components.

Qt also includes Qt Designer, a graphical user interface designer. PyQt is able to generate Python code from Qt Designer. It is also possible to add new GUI controls written in Python to Qt Designer.

Python is a simple but powerful object-orientated language. Its simplicity makes it easy to learn, but its power means that large and complex applications can be created. Its interpreted nature means that Python programmers are very productive because there is no edit/compile/link/run development cycle.

Much of Python's power comes from its comprehensive set of extension modules providing a wide variety of functions including HTTP servers, XML parsers, database access, data compression tools and, of course, graphical user interfaces. Extension modules are usually implemented in either Python, C or C++. Using tools such as SIP it is relatively straight forward to create an extension module that encapsulates an existing C or C++ library. Used in this way, Python can then become the glue to create new applications from established libraries.

PyQt combines all the advantages of Qt and Python. A programmer has all the power of Qt, but is able to exploit it with the simplicity of Python.

3.2 PyQt – Introduction

PyQt is a GUI widgets toolkit. It is a Python interface for **Qt**, one of the most powerful, and popular cross-platform GUI libraries. PyQt was developed by RiverBank Computing Ltd. The latest version of PyQt can be downloaded from its official website – riverbankcomputing.com

PyQt API is a set of modules containing a large number of classes and functions. While **QtCore** module contains non-GUI functionality for working with file and directory etc., **QtGui** module contains all the graphical controls. In addition, there are modules for working with XML **(QtXml)**, SVG **(QtSvg)**, and SQL **(QtSql)**, etc.

A list of frequently used modules is given below –

QtCore – Core non-GUI classes used by other modules

QtGui – Graphical user interface components

QtMultimedia – Classes for low-level multimedia programming

QtNetwork – Classes for network programming

QtOpenGL – OpenGL support classes

Q**tScript** – Classes for evaluating Qt Scripts

QtSql – Classes for database integration using

SQL

QtSvg – Classes for displaying the contents of

SVG files

QtWebKit – Classes for rendering and editing

HTML

QtXml – Classes for handling XML

QtWidgets – Classes for creating classic desktop-style Uis

QtDesigner – Classes for extending Qt Designer

PyQt is compatible with all the popular operating systems including Windows, Linux, and Mac OS. It is dual licensed, available under GPL as well as commercial license. The latest stable version is **PyQt5-5.13.2.**

Windows

Wheels for 32-bit or 64-bit architecture are provided that are compatible with Python version 3.5 or later. The recommended way to install is using **PIP** utility –

pip3 install PyQt5

To install development tools such as Qt Designer to support PyQt5 wheels, following is the command –

pip3 install pyqt5-tools

Figure 3.1. PyQt Logo[6]

3.3 PyQt5 – Qt Designer

Qt Designer is the Qt tool for designing and building graphical user interfaces (GUIs) with Qt Widgets. You can compose and customize your windows or dialogs in a what-you-see-is-what- you-get (WYSIWYG) manner, and test them using different styles and resolutions.

Widgets and forms created with Qt Designer integrate seamlessly with programmed code, using Qt's signals and slots mechanism, so that you can easily assign behaviour to graphical elements. All properties set in Qt Designer can be changed dynamically within the code. Furthermore, features like widget promotion and custom plugins allow you to use your own components with Qt Designer.

File generated by Qt Designer has .ui extension. This ui file contains XML representation of widgets and their properties in the design. This design is translated into Python equivalent by using pyuic5 command line utility. This utility is a wrapper for uic module of Qt toolkit. The usage of pyuic5 is as follows − *pyuic5 -x file_name.ui -o file_name.py.* In the above command, -x switch adds a small amount of additional code to the generated Python script (from XML) so that it becomes a self-executable standalone application.

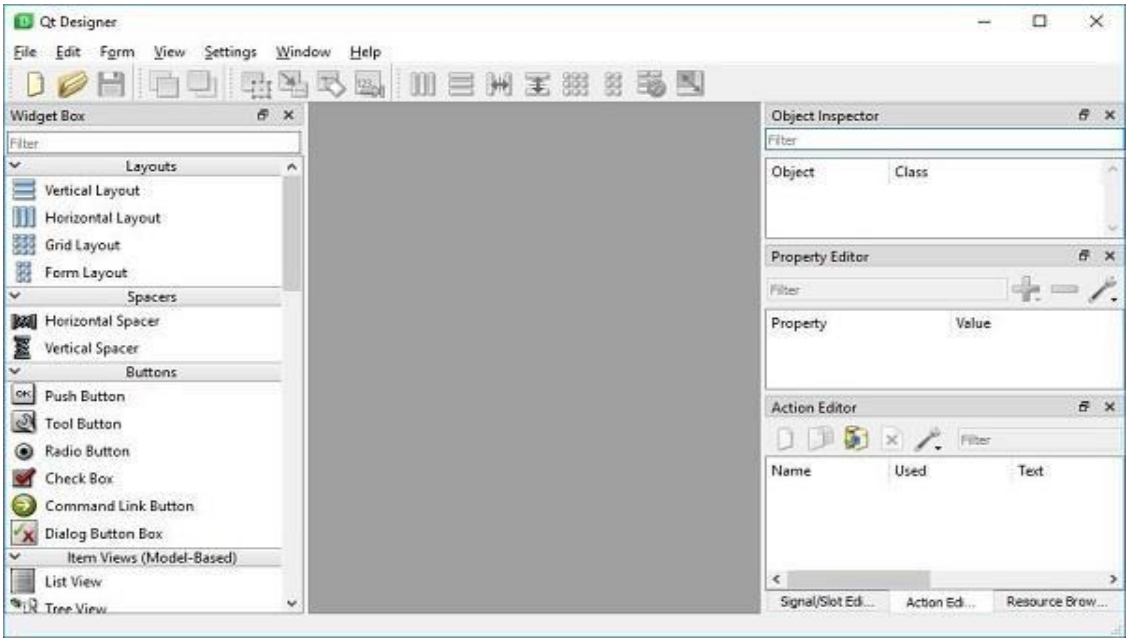


Figure 3.2. Qt Designer Main Window [7]

3.4 PyQt5 – Layout Management

PyQt's layout managers provide a user-friendly and productive way of arranging graphical components, or widgets, on a GUI. Laying out widgets properly will make your GUI applications look polished and professional. Learning to do so efficiently and effectively is a fundamental skill for you to get up and running with GUI application development using Python and PyQt.

In PyQt, widgets are graphical components that you use as building blocks for your GUI applications. When you place a bunch of widgets on a window to create a GUI, you need to give them some order. You need to set the widgets' size and position on the window, and you also need to define their behavior for when the user resizes the underlying window.

To arrange the widgets on windows or forms in PyQt, you can use the following techniques:

- Use .resize() and .move() on your widgets to provide an absolute size and position.
- Reimplement .resizeEvent() and calculate your widgets' size and position dynamically.
- Use layout managers and let them do all the calculations and hard work for you.

These techniques generally correspond to the three different approaches for laying out a GUI that you saw in the previous section.

Again, calculating the size and position dynamically might be a good approach, but most of the time you'll be better off using layout managers. In PyQt, layout managers are classes that provide the required functionality to automatically manage the size, position, and resizing behaviour of the widgets in the layout.

With layout managers, you can automatically arrange **child** widgets within any **parent**, or container, widget. Using layout managers will ensure that you make good use of the available space on your GUI and also that your application remains usable when the user resizes the window.

Layout managers work as containers for both widgets and other layouts. To add widgets to a layout manager, you call .addWidget() on the layout at hand. To add a layout to another layout, you call .addLayout() on the layout at hand. You'll dive deeper into

nesting layouts in the section Nesting Layouts to Build Complex GUIs.

Once you've added all the required widgets to a layout manager, you set the layout manager on a given widget using .setLayout(). You can set a layout manager on any subclasses of QWidget, including windows or forms.

All the widgets in a layout are automatically set as children of the widget on which you install the layout, not of the layout itself. That's because widgets can have only other widgets, not layouts, as their parent.

PyQt's layout managers provide some cool features that make your life a lot easier when it comes to creating good-looking GUI applications:

- Handling the **size** and **position** of widgets without the need for any calculation
- Handling the **resizing** and **repositioning** of widget when the user resizes the underlying window
- Resizing labels to better support **internationalization**
- Providing a native window layout for **multiplatform** applications

Using layout managers will also dramatically increase your productivity and improve your code's maintainability in the long term.

PyQt provides four general-purpose layout manager classes:

1. QHBoxLayout arranges widgets in a horizontal box.
2. QVBoxLayout arranges widgets in a vertical box.
3. QGridLayout arranges widgets in a grid.
4. QFormLayout arranges widgets in two columns

QHBoxLayout is one of the two available box layouts in PyQt. This layout manager allows you to arrange widgets **horizontally**, one next to the other. The widgets are added to the layout from left to right. This means that the widget that you add first in your code will be the left- most widget in the layout.

QVBoxLayout arranges widgets **vertically**, one below the other. You can use this class to create vertical layouts and arrange your widgets from top to bottom. Since QVBoxLayout is another box layout, its .addWidget() method works the same as in

QHBoxLayout.

Sr.No.	Methods & Description
1	**addWidget()** Add a widget to the BoxLayout
2	**addStretch()** Creates empty stretchable box
3	**addLayout()** Add another nested layout

Table 3.1 QHBoxLayout and QVBoxLayout Functions

QGridLayout takes the available space on its parent, divides it into rows and columns, and places each widget into its own cell or box. QGridLayout automatically figures out how many rows and columns the final layout will have depending on the number of widgets and their coordinates.

QFormLayout is a convenient way to create two column form, where each row consists of an input field associated with a label. As a convention, the left column contains the label and the right column contains an input field. Mainly three overloads of addRow () method addLayout() are commonly used.

Sr.No.	Methods & Description
1	**addRow(QLabel, QWidget)** Adds a row containing label and input field
2	**addRow(QLabel, QLayout)** Adds a child layout in the second column
3	**addRow(QWidget)** Adds a widget spanning both columns

Table 3.2 QFormLayout Functions

3.5 PyQt5 – Signals and Slots

Unlike a console mode application, which is executed in a sequential manner, a GUI based application is event driven. Functions or methods are executed in response to user's actions like clicking on a button, selecting an item from a collection or a mouse click etc., called **events**.

Widgets used to build the GUI interface act as the source of such events. Each PyQt widget, which is derived from QObject class, is designed to emit **'signal'** in response to one or more events. The signal on its own does not perform any action. Instead, it is 'connected' to a **'slot'**. The slot can be any **callable Python function**.

Signals are notifications emitted by widgets when something happens. That something can be any number of things, from pressing a button, to the text of an input box changing, to the text of the window changing. Many signals are initiated by user action, but this is not a rule.

In addition to notifying about something happening, signals can also send data to provide additional context about what happened.

Slots is the name Qt uses for the receivers of signals. In Python any function (or method) in your application can be used as a slot – simply by connecting the signal to it. If the signal sends data, then the receiving function will receive that data too. Many Qt widgets also have their own built-in slots, meaning you can hook Qt widgets together directly.

3.6 PyQt5 – Basic Widgets

PyQt has number of Widgets. Important ones are listed here –

Sr.No	Widgets & Description
1	QLabel A QLabel object acts as a placeholder to display non-editable text or image, or a movie of animated GIF. It can also be used as a mnemonic key for other widgets.
2	QLineEdit QLineEdit object is the most commonly used input field. It provides a box in which one line of text can be entered. In order to enter multi-line text, QTextEdit object is required.
3	QPushButton In PyQt API, the QPushButton class object presents a button which when clicked can be programmed to invoke a certain function.
4	QRadioButton A QRadioButton class object presents a selectable button with a text label. The user can select one of many options presented on the form. This class is derived from QAbstractButton class.

5	QCheckBox A rectangular box before the text label appears when a QCheckBox object is added to the parent window. Just as QRadioButton, it is also a selectable button.
6	QComboBox A QComboBox object presents a dropdown list of items to select from. It takes minimum screen space on the form required to display only the currently selected item.
7	QSpinBox A QSpinBox object presents the user with a textbox which displays an integer with up/down button on its right.
8	QSlider Widget & Signal QSlider class object presents the user with a groove over which a handle can be moved. It is a classic widget to control a bounded value.
9	QMenuBar, QMenu & QAction A horizontal QMenuBar just below the title bar of a QMainWindow object is reserved for displaying QMenu objects.
10	QStacked Functioning of QStackedWidget is similar to QTabWidget. It also helps in the efficient use of window's client area.

Table 3.3 PyQt Basic Widgets

3.7 PyQt5 – QPainter

We used QPainter for developing **Constellation Matrix.** The QPainter class performs low- level painting on widgets and other paint devices. QPainter provides highly optimized

functions to do most of the drawing GUI programs require. It can draw everything from simple lines to complex shapes like pies and chords. It can also draw aligned text and pixmaps. Normally, it draws in a “natural” coordinate system, but it can also do view and world transformation. QPainter can operate on any object that inherits the QPaintDevice class.

The common use of QPainter is inside a widget's paint event: Construct and customize (e.g., set the pen or the brush) the painter. Then draw.

The core functionality of QPainter is drawing, but the class also provide several functions that allows you to customize QPainter’s settings and its rendering quality, and others that enable clipping. In addition, you can control how different shapes are merged together by specifying the painter’s composition mode.

The isActive() function indicates whether the painter is active. A painter is activated by the begin() function and the constructor that takes a QPaintDevice argument. The end() function, and the destructor, deactivates it.

Together with the QPaintDevice and QPaintEngine classes, QPainter form the basis for Qt’s paint system. QPainter is the class used to perform drawing operations. QPaintDevice represents a device that can be painted on using a QPainter. QPaintEngine provides the interface that the painter uses to draw onto different types of devices. If the painter is active, device() returns the paint device on which the painter paints, and paintEngine() returns the paint engine that the painter is currently operating on.

Sometimes it is desirable to make someone else paint on an unusual QPaintDevice. QPainter supports a static function to do this, setRedirected().

3.8 PyQt5 – QGraphicsItem Class

We used this class for developing the **Animation** Widget. The QGraphicsItem class is the base class for all graphical items in a QGraphicsScene.

It provides a light-weight foundation for writing your own custom items. This includes defining the item’s geometry, collision detection, its painting implementation and item interaction through its event handlers. QGraphicsItem is part of the Graphics View Framework.

QGraphicsItem receives events from QGraphicsScene through the virtual function

sceneEvent(). This function distributes the most common events to a set of convenience event handlers. Basically, it is used to design widgets according to one's need and making dynamic widgets.

3.9 PyQt5 – QPropertyAnimation and QParallelAnimationGroup

We used these in the development of the Floating Widget for Add/Delete and Settings Button. QPropertyAnimation allows you to change the value of an attribute of an object from a startValue to an endValue over a certain amount of time, and optionally following a custom easing Curve.

QParallelAnimationGroup –a container for animations –starts all its animations when it is started itself, i.e., runs all animations in parallel. The animation group finishes when the longest lasting animation has finished.

3.10 QThread

We used this to develop the backend of our GUI. A QThread object manages one thread of control within the program. QThreads begin executing in run(). By default, run() starts the event loop by calling exec() and runs a Qt event loop inside the thread.

PyQt graphical user interface (GUI) applications have a main thread of execution that runs the event loop and GUI. If you launch a long-running task in this thread, then your GUI will freeze until the task terminates. During that time, the user won't be able to interact with the application, resulting in a bad user experience. Fortunately, PyQt's QThread class allows you to work around this issue.

Long-running tasks occupying the main thread of a GUI application and causing the application to freeze is a common issue in GUI programming that almost always results in a bad user experience.

The application's GUI freezes as a result of a blocked main thread. The main thread is busy processing a long-running task and doesn't immediately respond to the user's actions. This is an annoying behaviour because the user doesn't know for sure if the application is working correctly or if it's crashed.\

Fortunately, there are some techniques you can use to work around this issue. A

commonly used solution is to run your long-running task outside of the application's main thread using a worker thread.

Methodology ...

4.1 Software Development Life Cycles Methodologies (SDLC)

Software Development Life Cycle (SDLC) is a method used by the software industries to create, design, develop, and check high quality software. The SDLC aims to create high-quality software that reaches or exceeds customer expectations and completion is done within time and cost as estimated. SDLC means Software Development Life Cycle. It is also known as the Software Development Process. SDLC is a framework explaining tasks performed at each time in the software development process. ISO/IEC 12207 is an international standard for the software life- cycle processes. It aims to be the standard that explains all the tasks needed for developing and maintaining software.

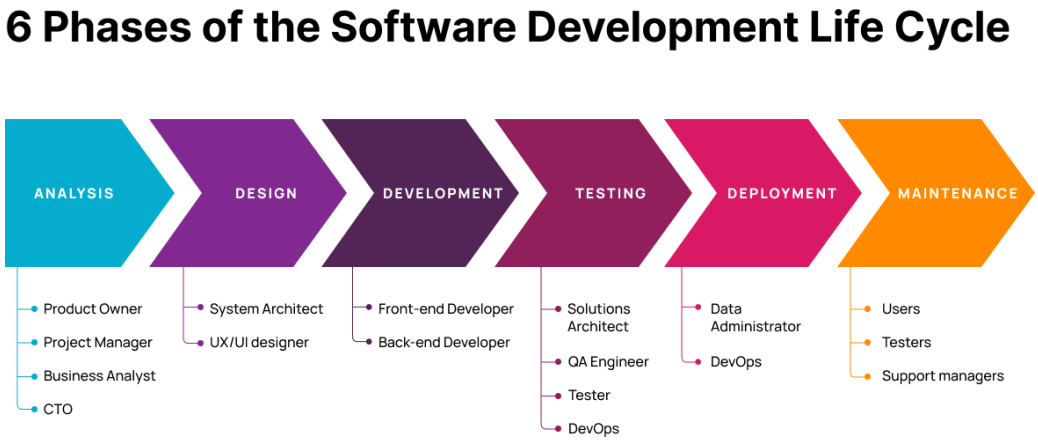


Figure 4.1. SDLC [8]

4.2 Agile Methodology

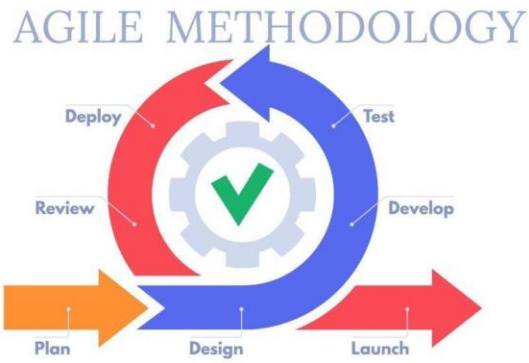


Figure 4.2. Agile Methodology [9]

4.3 Scrum

Scrum is an agile framework created for teams of three to nine members who divide their work into actions that can be done within certain time limits, known as sprints, not more than one month and most commonly two weeks, then record the progress and re-plan and re-think in 15- minute time-boxed stand-up meetings, known as daily scrums. Now, while working on this project, we had a daily scrum meeting of around 1 hour for a project in which we used to discuss what we have done, what will be done today and if facing any issues, then everyone tries to find a solution for that.

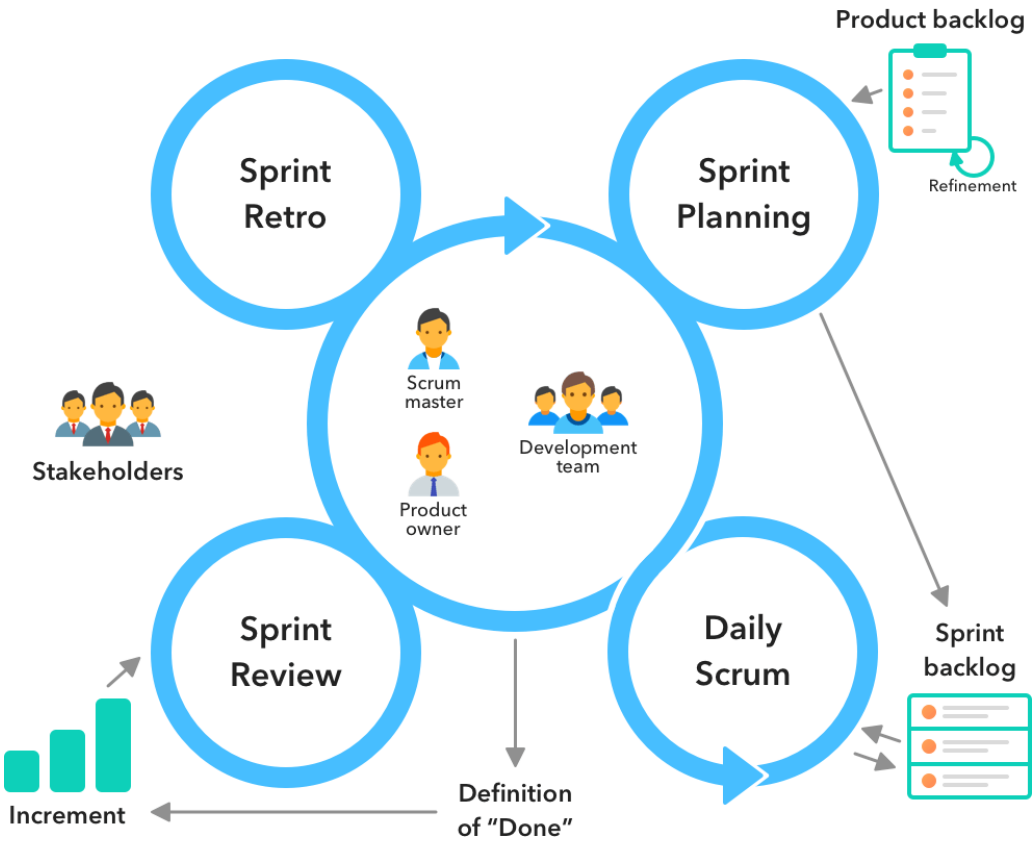


Figure 4.3. Scrum Methodology [10]

4.4 Development Phases

The different phases of the development process – collecting requirements, building or coding, Report Design, Linking, testing ability required in the development process.

4.5 SDLC Activities

Planning-Requirement collecting or requirement analysis, Scope of the project is founded and documented.

4.5.1 Implementation

Building the code according to the client's requirements and needs.

4.5.2 Testing

Method of finding bugs and errors/defects.

4.5.3 Documentation

Series of Steps in the project are written for future reference

4.5.4 Deployment

Software is deployed after the software has been approved.

4.6 Delivery and Deployment Process

4.6.1 DOU Preparation

DOU (Document of understanding) is a high-level requirement document. This document is checked and signed-off by the client and is used as the baseline document for making detailed requirements.

4.6.2 FRD Prepaation

DOU (Document of understanding) is a high-level requirement document. This document is checked and signed-off by the client and is used as the baseline document for making detailed requirements.

4.6.3 Development

This phase starts when the FRD is decided and finalized. Developers create and perform unit testing on the Dev server. Post verification, the build is deployed on a QA sever and released to the Quality Analyst team for verification.

4.6.4 Quality Testing

Quality Analysts checks the build against the test cases created by them. Post completion of QA, bugs are written in Jira and assigned for resolution and tracking to the development team.

4.6.5 Review Test

Product Analysts review test cases created by the quality analysts. It performs functional testing of the build.

Observations and Findings ...

5.1 Review Test

The developed GUI was able to fulfil all the criteria provided. Unit testing was done for each component. Separate stories were created to look after each component in detail without depending on other components. Each component was designed separately with separate style files so that no two components depend on each other.

5.2 All the functionalities were achieved

Firstly, basic Frontend (GUI) and Backend (QThread Work) is developed separately then they are merged. After merging, continuous testing and debugging is done. GUI is showing all the metrics that are desired correctly and GUI is also looking good having all the features required. Enhancements on the GUI are done later.

5.3 Best Practices

During the development of the GUI, best practises were effectively followed. When creating the GUI, best practises such as accessibility, reusability, user experience, and responsiveness were considered. The GUI employs a pleasing font type, appropriate font size, and a pleasing colour scheme.

5.4 Code Formatting

- It helps in making our code more readable and maintainable.
- Layout, Indentation, Whitespaces, Line breaks
- Naming conventions, spelling, typos, camelCase/UPPERCASE/lowercase
- Proper use of comments for e.g., explaining the working of a function, layout structure, working of widget, etc.
- Creating and following a style guide specific to your project.

5.5 Tools Usage

Taking advantage of tools – IDE (VSCode, Spyder), Online linters and formatters, etc. to help in code formatting, debugging errors, optimizing our code and much more.

Conclusion ...

Learnings:

Through the time of the internship, I was able to do work along with groups to enhance the base product. Aligning with the GUI development team I was able to assist in implementing and helped in creating the new technology for Samsung Beyond 5G team capabilities and creating the functional capabilities. Able to document all the corner cases. Along with all individual component's unit testing was performed. Every component was having a separate testing file so it can be tested without affecting and similarly each component was having its own style file. New tech stacks that I learned are PyQt, Python and I also learnt how to develop installer version for python code.

Work Done:

During my internship, I was assigned to this project and was able to study documents including needs and specifications, assess requirements, and map them to mock-ups. I was able to decipher new tech stack documents and map it to required duties. I was able to write requirements-driven tales. I knew what it was like to work in a fast-paced atmosphere. I started by documenting stories or tasks that would take a day to complete, then did them and concluded the story.

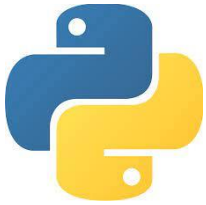
Achieved Best Practices:

Finally, we conclude that our GUI provides a user-friendly, device-compatible, adaptable, and high-quality interface that will provide a time-saving approach for performing 6G testing. We created an installer for this that makes it incredibly simple to install on any computer. During the development of this GUI, the accessibility feature was considered. During the development of the GUI, best practises were effectively followed. When designing the interface, best practises such as accessibility, reusability, user experience, and responsiveness were taken into consideration. The GUI employs a pleasing font type, appropriate font size, and a pleasing colour scheme.

Appendices ...

Appendix 1: Details of the Software Packages used in the project

- **Python:** Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation.



- **PythonQt/PyQt:** PyQt is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in. PyQt is free software developed by the British firm Riverbank Computing.

- **Spyder:** Spyder is an open-source cross-platform integrated development environment for scientific programming in the Python language.

Appendix 2: List of Figures

Appendix 3: List of Tables

Appendix 4: Acronyms and Abbreviations

Abbreviation	Full Form
RRC	Radio Resource Control
SRIB	Samsung Research Institute Bangalore
TBNB	Test Bed eNodeB
TBUE	Test Bed User Equipment
VS CODE	Visual Studio Code
API	Application programming interface
ASN	Autonomous System Number
IP	Internet Protocol
GUI	Graphical User Interface
BLER	Block Error Rate
UI	User Interface
SDLC	Software Development Life Cycle

References ...

- https://www.samsung.com/in/about-us/brand-identity/logo
- https://news.samsung.com/za/10-for-10-highlights-from-a-decade-of-galaxy-innovation
- https://www.samsung.com/in/about-us/ethics/
- https://research.samsung.com/sri-b
- https://en.wikipedia.org/wiki/PyQt
- https://doc.qt.io/qt-6/designer-to-know.html
- https://brocoders.com/blog/agile-software-development-life-cycle/
- https://mobile-jon.com/2021/04/05/applying-the-agile-methodology-to-the-modern-workplace/
- https://startinfinity.com/project-management-methodologies/scrum
- https://riverbankcomputing.com/software/pyqt/
- https://www.samsung.com/us/about-us/leadership-and-mission/
- https://doc.qt.io/
- https://www.tutorialspoint.com/pyqt5/index.htm
- https://pypi.org/project/PyQt5/
- https://www.geeksforgeeks.org/python-introduction-to-pyqt5/
- https://www.pythonguis.com/tutorials/pyqt-signals-slots-events/
- https://realpython.com/python-pyqt-qthread/
- https://wiki.qt.io/How_to_use_QPainter
- https://het.as.utexas.edu/HET/Software/html/qgraphicsscene.html
- https://www.pythonguis.com/tutorials/creating-your-own-custom-widgets/
- https://www.britannica.com/topic/Samsung-Electronics
- https://news.samsung.com/global/into-the-future-with-samsung-research-5-samsung-rd-institute-india-bangalore-advanced-communication-networks-innovate-the-daily-life-of-the-future
- https://build-system.fman.io/pyqt5-tutorial

Glossary ...

UI (User Interface) **:** User interfaces are the access points where users interact with designs. They come in three formats: Graphical user interfaces (GUIs)—Users interact with visual representations on digital control panels. A computer's desktop is a GUI. Voice-controlled interfaces (VUIs)—Users interact with these through their voices.

API **:** API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.

Responsiveness **:** Responsive web design is the approach that suggests that design and development should respond to the user's behavior and environment based on screen size, platform and orientation.

END

Thank You