THE DEVELOPMENT OF SOFTWARE FOR A SATELLITE GROUND STATION

by

NG KIAN KIONG

Thesis submitted in fulfilment of the requirements for the Bachelor Degree of Engineering (Honours) (Aerospace Engineering)

June 2019

ENDORSEMENT

I, Ng Kian Kiong hereby declare that all corrections and comments made by the supervisor and examiner have been taken consideration and rectified accordingly.

(Signature of Student)

Date:

(Signature of Supervisor)

Name:
Date:

(Signature of Examiner)

Name:
Date:

DECLARATION

This thesis is the result of my own investigation, except where otherwise stated and has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any other degree.

(Signature of Student)

Date:

ACKNOWLEDGMENT

First and foremost, I must thank my final year project supervisor, Dr. Aiffah Mohd Ali. Without her assistance and dedicated involvement in every step throughout the process, this thesis paper would have never been accomplished. I would like to thank you very much for your support and understanding throughout the project period.

I would also like to show gratitude to my final year project examiner, Dr. Norilmi Amilia Ismail for her encouragement, insightful comments, and challenging questions. Her suggestions have aided me in approaching this final year project.

Additionally, I would like to thank my friends in Universiti Sains Malaysia: Teo Chen Lung, Lim Yew Hao, Leong Yeong Chee, Lam Kuek Shen, Tan Chun Khuen, Low Jian Yan and Yap Kai Wen.

Most importantly, none of this is possible without my family. My parents, who offered their encouragement through phone calls as often as they could, has become my moral support over the last several years. My siblings who always show their moral support throughout my educational years. They always cheer me up during time when I'm feeling down, and I am forever grateful. This thesis stands as a testament to their unconditional love and encouragement.

ABSTRACT

MYSat (Malaysian Youth Satellite) is a CubeSat aimed to undergo the mission of measuring the electron density in the ionosphere. However, a satellite's mission success depends on different aspects, and one of them is the satellite's ground station. Ground station acts as the mission control of the satellite, which is responsible for enabling the communication between the satellite and the ground station personnel. A strong communication is required to allow data tracking and receiving from the satellite for data analysis purposes. With the ground station successfully fabricated, a software responsible of command uplink and downlink of data is required to be developed for the ground station to function as desired. Therefore, the purpose of this project is to develop a software acting as the ground station control which command uplink and downlink for MYSat Ground Station. The software developed is fitted with GUI (Graphical User Interface), which is meant to be user-friendly and easily accessible and understandable. Microsoft Visual Studio with C# language is used as the software developing environment to design and develop the GUI software. A simulation of the hardware connection with the computer software which includes Arduino Nano, radio telemetries and sensors, is constructed to understand, develop and verify the functionality of software viability, so that errors could be reduced when developing the finalized software of ground station while further modification could be made easily to accommodate the satellite's mission requirement. Database design is required for the satellite's mission, for storing the data throughout the mission period. The database allows the user to easily access the data from the mission, which is categorized based on different group such as date and time. This allows the data to be available for future references, analysis or comparison of mission data.

ABSTRAK

MYSat ialah sebuah CubeSat yang bertujuan untuk mengukur ketumpatan elektron di ionosfera. Tetapi, kejayaan misi satelit beergantung pada beberapa aspek, dan salah satunya ialah stesen darat satelit. Stesen darat setelit bertujuan sebagai misi kawalan satelit, untuk komunikasi antara satelit dan pengendali stesen darat. Komunikasi yang kuat diperlukan dalam penjejakan dan penerimaan data untuk data analisis. Setelah stesen darat satelit berjaya dibina, perisian yang memberi perintah uplink dan downlink diperlukan untuk stesen darat yang siap bina berfungsi seperti yang dikehendaki. Oleh itu, tujuan projek ini adalah untuk membina perisian yang bertujuan memberi perintah uplink dan downlink daripada stesen darat satelit yang sudah dibina. Perisian tersebut telah dibina dengan GUI, yang bertujuan untuk mesra pengguna dan senang diakses. Microsoft Visual Studio dengan bahasa C# diguna untuk membina perisian tersebut. Simulasi antara perisian computer dan perkakasan termasuk Arduino Nano, radio telemetri dan sensor telah dibina untuk memahami dan mengesahkan fungsi perisian yang telah dibina. Dengan ini, kesilapan dapat dikurangkan semasa membina perisian yang muktamad serta pengubahsuaian dapat diberi dengan senang. Pangkalan data juga diperlukan untuk misi satelit. Pangkalan data memberi pengguna kesenangan mengakses data yang telah dikategorikan berdasarkan tarikh dan masa. Pangkalan data juga membenarkan akses untuk rujukan masa depan serta perbandingan misi data.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF ABBREVIATIONS

MYSat	Malaysian Youth Satellite
LEO	Low Earth Orbit
USSL	USM Space System Lab
GUI	Graphical User Interface
CLI	Command Line Interface
OBC	On-Board Computer
TNC	Terminal Node Controller
ERS	Enhanced and Redesign Scripting
IDE	Integrated Development Environment
DBMS	Database Management System
RDBMS	Relational Database Management System
SQL	Structured Query Language
API	Application Programming Interface

CHAPTER 1

INTRODUCTION

1.1 Research Background

CubeSat is a comparatively small satellite which has a lower cost of development and has been a favourable option of satellite nowadays considering the absurd cost of the large satellite (Dalgleish *et al.*, 2007). CubeSat was introduced with a standardized specification of usage while simplifying the launching option (THAHER, 2017). Many parties such as universities and companies are interested in CubeSat development nowadays, due to its high success rate of deployment with relatively low investment cost (Chin *et al.*, 2012), which becomes the driving aspect for Universiti Sains Malaysia to develop its own CubeSat named MYSat (Malaysian Youth Satellite). MYSat's mission is expected to measure the electron density in the ionosphere layer, more specifically the ionosphere E-layer. Events or natural disasters such as earthquakes, volcano eruption and the launching of a rocket are noticed to excite the atmospheric wave, which is desirable if known beforehand for countermeasures or taking safety measures against them.

1.1.1 Ground Control Station

Ground stations play a very important role in facilitating the mission of the CubeSat. This is because a strong communication system between the CubeSat and the ground station is important to allow data tracking and receiving from the CubeSat for data analysis purposes (Harris, 2014). Nanosatellites orbiting at Low Earth Orbit (LEO) are only visible for a short period of time from the perspective of one location on Earth (Du, 2005). Therefore, it is important for the satellites' data to be dumped to the ground station within the time period when it passes by the ground station area. The satellites'

mission will be irrelevant if important data such as payload data fails to be received by the ground station at a consistent pace or designated period. In order to support MYSat and any future nanosatellites missions developed by USM Space System Lab (USSL), a ground station is required to facilitate the communication between the operator on Earth and the satellite.

1.1.2 Ground Control Station Software

Complete fabrication of MYSat's ground control station only covers portion of the work to be done before the ground control station becomes fully functional for MYSat's mission. For a ground control station to be functional, a dedicated software needs to be developed to compliment the hardware, which is the fabricated ground control station. Ground control station software plays the major role in commanding uplink and downlink data, sending off signals of commands and instructions through the fabricated ground control station. A functioning ground control station software is expected to facilitate the communication between the fabricated ground control station and the satellite through the input of text command or command line. Therefore, the development of ground control station software is closely associated with a designated computer language to forge the connection between ground control station and satellite.

1.1.3 Graphical User Interface (GUI)

Graphical User Interface (GUI) is part of the ground control station software, where it acts as the direct interaction between the users with the developed software and hardware. The ground control station software is more of a command-line interface (CLI) and text-based user interface, where certain computer language or text command is

written to execute certain tasks of the developed software. Therefore, having either of the two interfaces is not particularly user-friendly because of the visually unappealing interface, complicated computer language and steep learning curve. On the other hand, the implementation of GUI provides a clear visual presentation in its interface through the usage of graphical icons and visual indicators. GUI is often presented with a clear direction of usage for the specific software, ensuring that the users understand what everything displayed in the interface does, without having to learn through a whole series of programming language. Therefore, a GUI is often necessary for complimenting the developed software, providing an interface which is easily understood for majority of users.

1.1.4 Satellite Database

The amount satellite's data is usually massive, ranging from mission data to housekeeping data, therefore MYSat's mission is expected to house a tremendous amount of data throughout its mission lifespan as well. However, the organizing of satellite's data can be hideous after an intensive amount of data is dumped to the ground control station through several cycles of mission. Therefore, a database is often required to countermeasure the inconvenience brought upon by continuous accumulation of satellite's data.

A satellite database acts as a storage for the housing of satellite's data which is meant to be easily accessible, managed and updated throughout MYSat's mission lifespan. The data stored in the database is organized in the formats of rows, columns and tables, allowing users to browse through relevant data or information easily. Besides, certain relationship can be forged between different tables, further defining the relation

between each types of data as well as data categorization. Therefore, developing a database for MYSat is necessary for the mission to be successful and manageable in long run.

1.2 Problem Statement

Ground Station plays a major role for a satellite mission, even though satellite is the actual hardware executing the mission objectives. However, ground station is the one responsible for sending a signal of commands to the satellite to execute the instructions desired, while acting as the communication system for the satellite mission. USM Space System Lab (USSL) has successfully designed and fabricated its own ground station, acting as the hardware of the satellite mission's communication system, while the software responsible for the control of ground station is still currently absent. The ground station is practically non-functional without the presence of a proper functional software responsible for the ground station's control. This project is to develop the software acting as ground station control which commands the data uplink and downlink for MYSat Ground Station.

1.3 Objective of Research

There is a total of three objectives derived from the problems stated, which is expected to be completed in this project. The objectives are as stated below:

(i) To analyze and design the software requirement for the ground station.

(ii) To create a database responsible of storing satellite's data

(iii) To develop the software responsible for the command uplink and downlink of the ground station.

(iv) To integrate and test the designed ground station software with the available hardware of MYSat.

1.4 Thesis Layout

This thesis is comprised of 5 chapters. Chapter 1 gives a general overview of MYSat's mission and Ground Control Station. Additionally, the ground station software and Graphical User Interface (GUI) are briefly discussed, along with the concept of satellite's database. Finally, the problems of software implementation in ground control station is discussed, followed by the objective of this research.

Chapter 2 reviews all literatures related to this project, focusing on the satellite communication system, ground station software and database. Chapter 3 describes the procedures in designing and developing the ground station software whereas Chapter 4 discuss the finalized product of the developed software along with its performance. The final chapter conclude the research done in this paper with suggestion for future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Satellite Communication System

Satellite communication system is the communication between the satellite and the users or ground station. It is the system where data transfer exists between the satellite and ground station. Satellite tends to transmit data such as telemetry and payload to the ground station, whereas ground station sends off signals of command and satellite tracking data.

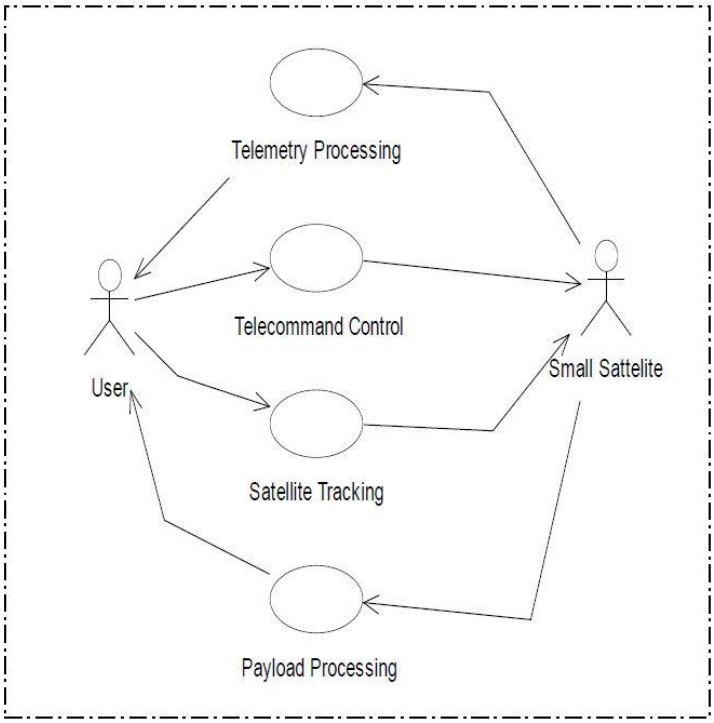

Figure 2.1 : Satellite Communication System Network

(Dinesha, Bhagavan and Agrawal, 2013)

Telemetry between satellite and ground station is often divided into three phases, namely pre-pass, real-time and post-pass. In pre-pass, the trajectory of satellite should be predicted, so that operator can program the telecommands to be sent. In real-time, the

antennas' auto tracking is activated for reception from satellite, with telemetry-on demand data received. Finally, each of the received parameters is evaluated in relation to their operating ranges in order to determine the state of the satellite (González, Cabrera and Calderón, 2016).

On-board Computer (OBC) plays a major role in telemetry of satellite mission. The main responsibility of the OBC is to monitor the health of the satellite system, as well as taking necessary actions whenever situations demand for it. The health of satellite is usually monitored by periodically requesting health packages from respective software instances and sensors for different modules. Additionally, OBC also acts as a gateway between the satellite bus, and the radio link to ground station (Normann and Birkeland, 2016).

Commonly, satellite communication system exists between the satellite and the respective ground station. However, relaying communications signals between one or more user terminals and a ground Station has been proposed for satellite communication system as well. The user terminals may be located at various places with multiple terminals located with a common field of view of one specific satellite. These satellites typically include antenna arrays that define spot beams or footprints on the Earth's Surface (Fan, 2006).

2.2 Ground Station

Ground station is generally divided into 4 segments, which are hardware, software, people and operation. Regarding the hardware for ground station, it usually includes antenna, rotors, peripherals, transceivers and computers. Software aspects of ground station generally involves the command uplink and downlink of data. The people

involved in a ground station operation is responsible for project management, operation shift staff, hardware staff, software staff, data and engineering support stuff, administration, etc. Finally, operation brings the three other aspects together, which is fundamental of integrating the software, hardware and the people involved into an effective routine process (Wickramanayake, 2007).

Most ground station is fitted with a Terminal Node Controller (TNC). TNC is part of the ground station which is responsible for the control of major ground station hardware and equipment such as radio, antenna and rotor. It also performs certain spacecraft orbital calculation for predicting the position of spacecraft, while processing the data transfer between spacecraft and ground station (Tuli, Orr and Zee, 2006).

2.2.1 Ground Station Software

Computers are closely tied to the operation of space vehicles, especially in earlier days before Windows existed. Those type of computers are typically run with FORTRAN or C programs which utilized text-based user interfaces. However, the advent of graphical user interface (GUI) caused many software program to be redesigned or revised to implement GUI (Davis and Parkway, 2000).

GUI plays a dominant role over software program, especially with its user interfaces which are welcoming for all types of user. GUI's layout is fitted with graphical indicators and interactive interfaces such as buttons, combo box and display screen. Even for operations regarding space vehicles, GUI implemented software program provides ground control station certain level of convenience. A direct example is the data display, since GUI implemented software usually displays values of data as "string" format, rather than unprocessed computer language (Aydemir, Dursun and Pehlevan, 2013).

Scripting language is a common way of implementation of command and control in a spacecraft's ground control operations. Enhanced and Redesign Scripting (ERS) which is closely related to GUI implementation, make use of graphical and logical richness of a programming language while providing ease of control in scripting language. ERS is commonly used for integrating spacecraft command mnemonics, telemetry measurements, command and telemetry control procedures into a standard programming language. Script-style user control becomes available during procedure execution through robust graphical user input and output feature (Ritter and Pedoto, 2012).

Great example of ground station software is Mercury developed by SSDL (Cutler and Kitts, 2008). Mercury provides operational efficiency by providing a centralized software interface which can control all ground station equipment, software routines to automate station operation with an Internet gateway for accessing the centralized interface.

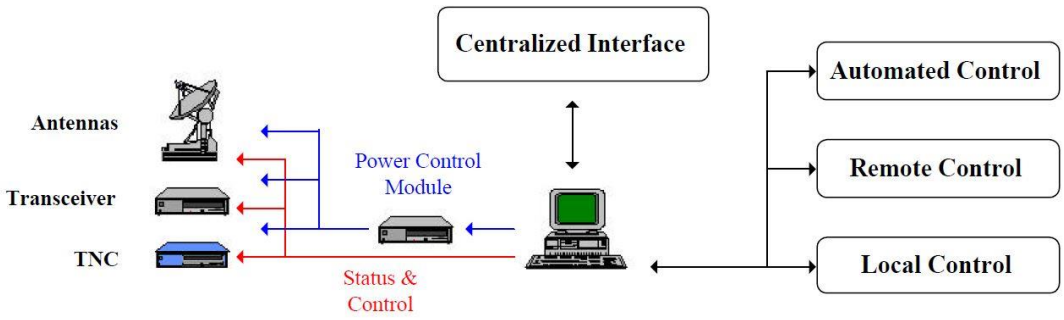

Figure 2.2 : Block Diagram of Mercury System

(Cutler and Kitts, 2008)

2.2.2 Database

Database design or data modelling is one aspect of software engineering as well. Modelling a data is the first design step towards using a database in an application. The structure of a database has to be defined, especially for a Relational Database Management System (RDBMS), which the structure includes details like defining attributes, tables and specifying rules for ensuring the integrity of tables (Rosenmill and Österman, 2000).

Structured Query Language (SQL) is one of the more common and higher-level programming languages for RDBMS. SQL statements can modify the structure of databases using Data Definition Language statements, and manipulate the contents of databases using Data Manipulation Language statements (Anley, 2002).

An active database system should support constraints as well as event-driven application logic, where database events can be easily triggered through the happening of certain logic such as pushing a button (Cochrane, Pirahesh and Mattos, 1996).

CHAPTER 3

METHODOLOGY

3.1 Project's Flow Chart

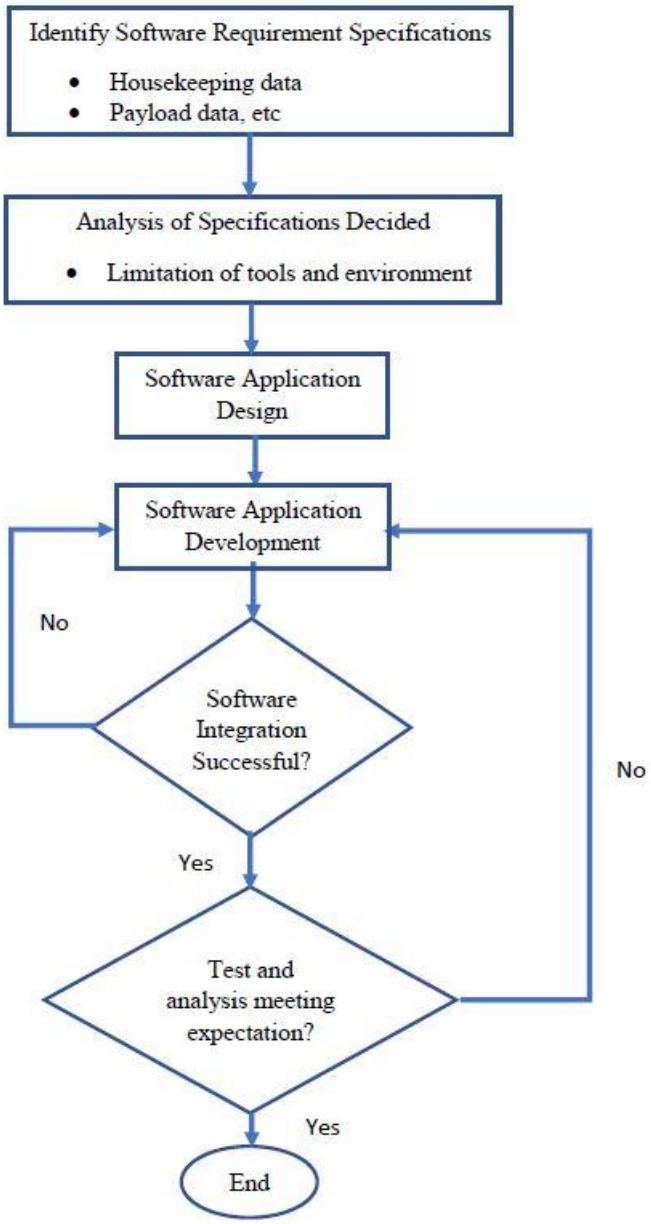

Figure 3.1 : Project's Flow Chart

The project starts off with the identification of software requirement specifications, which include the parameters to be recorded and displayed in the developed software. These specifications are then analysed based on the existing

limitations such as the capability of development tools and hardware setup available, before deciding the acceptance of those specifications in the software development later. The software is then designed and developed based on the decided specifications while providing further improvements to the developed software until finalization. Next phase involves the integration of the developed software with the hardware, for identifying whether the developed software can be successfully implemented on a working hardware. This is a continuous phase until the developed software fully meets the expectation as well as functioning during the integration with hardware.

3.2 Hardware Configuration

Certain configuration of hardware is required for the software development. A computer is necessary to the design and development of software. The developed software also requires a functioning computer so that it can be operated.

On the other hand, a setup of hardware is required to simulate the connection between the software developed and fabricated hardware. Therefore, a simple configuration of hardware which is available from USSL is used for the simulation of connection. Figure 3.2 shows a set of hardware components which are Arduino Nano, a pair of radio telemetry acting as transmitter and receiver, and sensors.

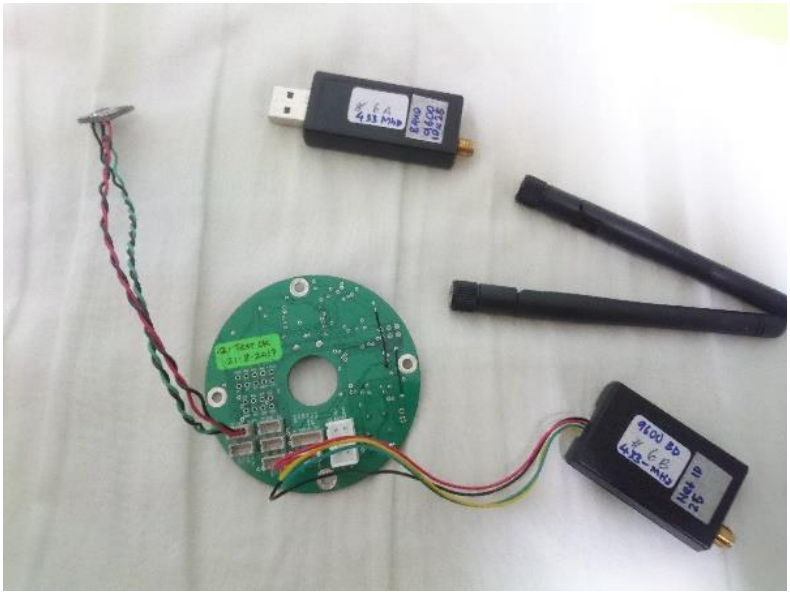

Figure 3.2 : Arduino Nano, Radio telemetry, Sensors

Figure 3.3 : Computer connected to the hardware

Figure 3.3 shows the complete setup of hardware components along with the computer which is used for operating the developed software. This setup acts as a simulated connection between a developed software with a functioning hardware.

3.3 Software Architecture

Software architecture plays the most important part in this project, since the main objective of this project is to develop a functioning software to compliment the fabricated hardware. The software architecture is generally classified into few sections, which are definition of parameters, development of database and development of GUI.

3.3.1 Parameter Definition

Defining parameters to be recorded and displayed in the developed software is the very first step of developing the ground control station software, since defining parameters beforehand provides a clear vision for the design and development of both database and the GUI. However, the parameters accepted for this project is not definitive, because the developed software is meant to be improved over time. Therefore, new additions or changes towards the parameters may be required before the software is finalized while satisfying the expectations of the users. Besides, the parameters are still very limited by the available hardware such as sensors, since there will not be any other means to detect the specified parameters' data for the display in GUI.

3.3.2 Database Development

Microsoft SQL Server is the choice of tool used for development of database, complimented with Microsoft SQL Server Management Studio as the developing environment. Microsoft SQL Server is a type of Relational Database Management System (RDBMS) software which is built on top of SQL, a standardized programming language used among database administrator. There are also other RDBMS software available such as Oracle, MySQL, SQLite. However, Microsoft SQL Server is chosen

for the development of software because it comes with it is an open source software which has its very own accessible database engine. Besides, Microsoft SQL Server also compliments well with the GUI development software, Microsoft Visual Studio which will be discussed in the next section.

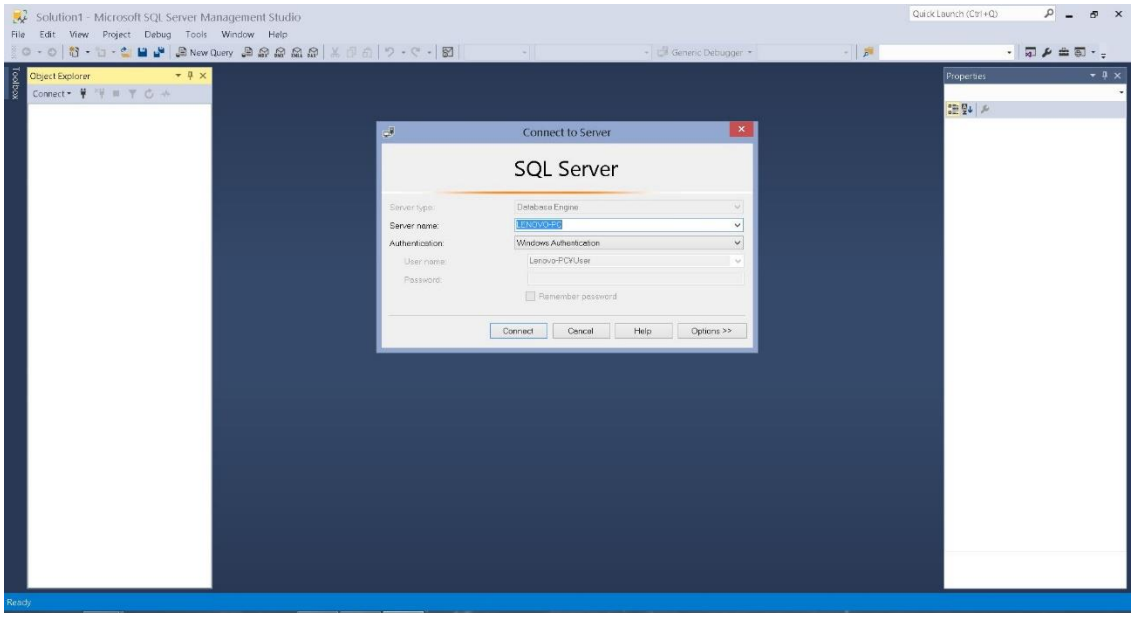

Figure 3.4 : General Interface of Microsoft SQL Server Management Studio

Figure 3.4 shows the general interface of Microsoft SQL Server Management Studio with the connection of preferred database engine. Database engine comes in various forms such as cloud-based database engine and local storage-based database engine. Before any further decision is made for the future of MYSat, this project uses a local storage-based database engine, which means the computer itself acts as the storage for data.

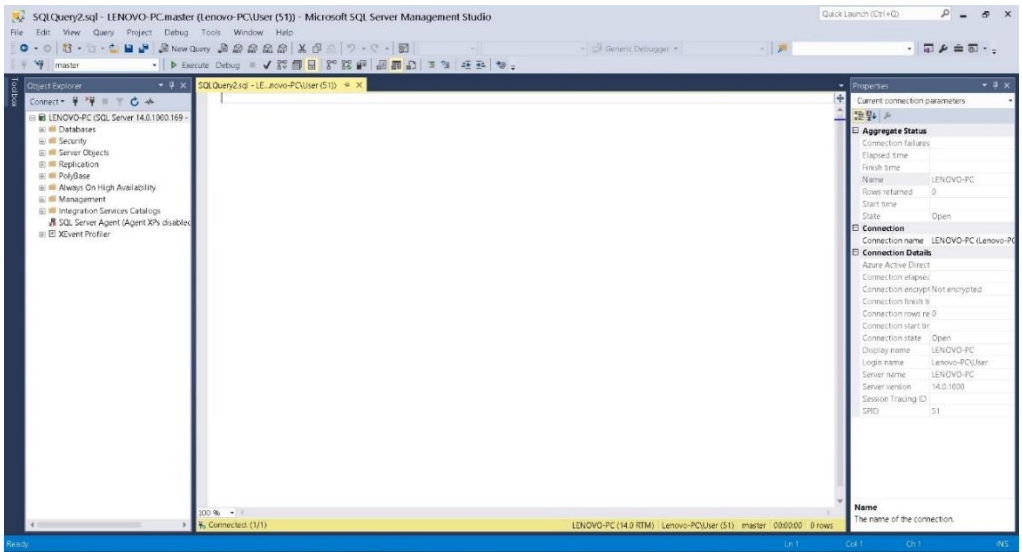

Figure 3.5 : Microsoft SQL Server Application Programming Interface

Figure 3.5 shows the interface after connecting to a database engine, along with Microsoft SQL Server's own application programming interface (API) also known as query. Query allows the user to interact and manage the software's database engine as desired, such as updating data and retrieving data from database through usage of SQL programming language.

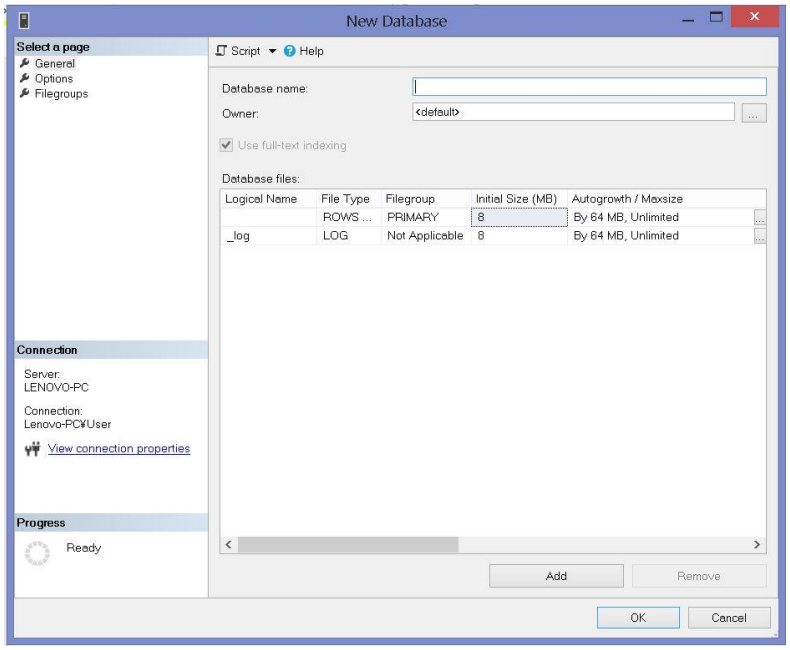

Figure 3.6 : Creating Database in Microsoft SQL Server

A database can be easily created using Microsoft SQL Server Management Studio as shown in Figure 3.6 after defining certain properties for the database. Data can then be stored in a created database in tabular form.

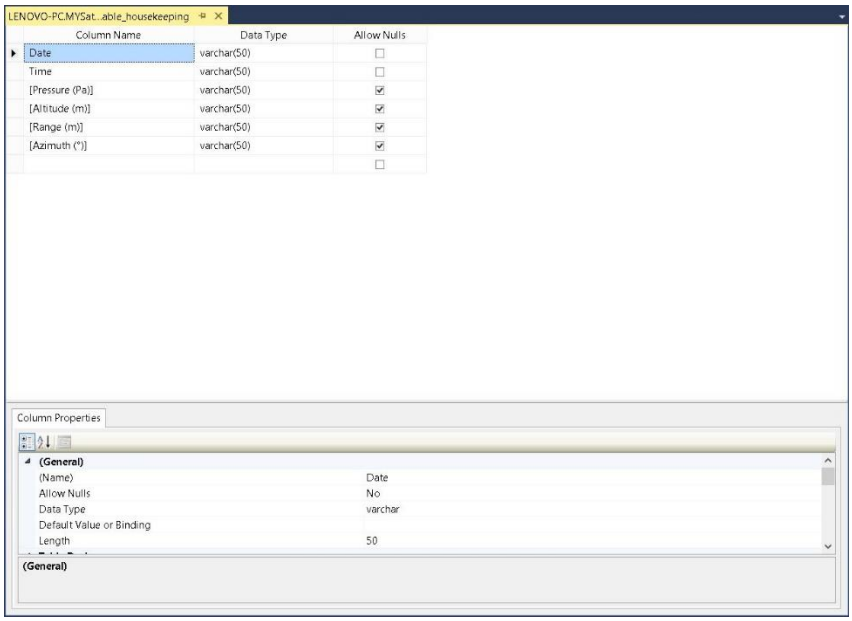

Figure 3.7 : Creating Tables in Database

Figure 3.7 shows the interface for the design of tables in an existing database. Each column within a table needs to be defined properly before the table is available for the storage and organizing of data. The column contains properties such as column names and data type. Both properties are crucial for the development of GUI later, since column names need to be used in development of software while data type affects the types of data which can be stored inside the column such as numbers or characters.

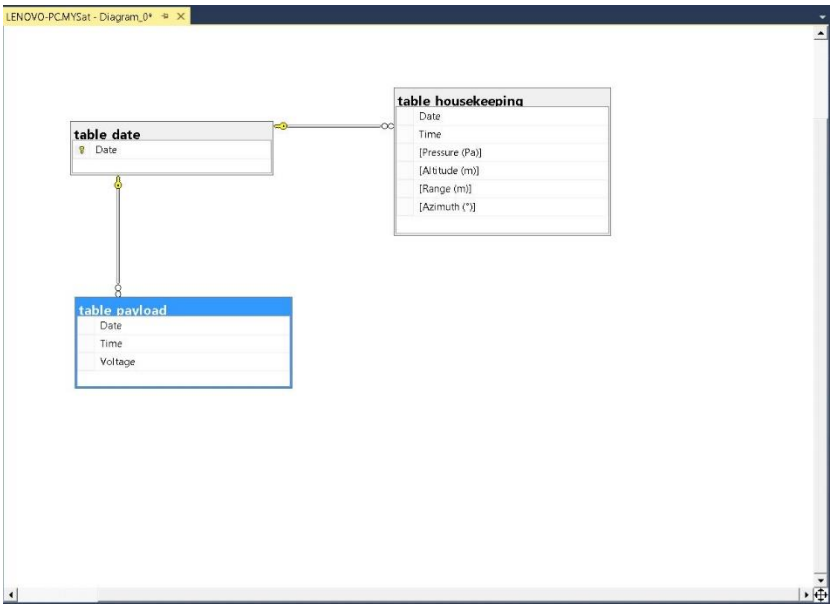

Figure 3.8 : Creating Relationship between Multiple Tables

One thing that differs RDBMS and Database Management System (DBMS), is the storage of data in tabular form. Data stored in tabular form allows the user to form a relation between different tables by defining certain columns as PRIMARY key column and FOREIGN key column. This type of relational database provides a more organized way of data storage, while providing various ways for relational data to be extracted and viewed.

3.3.3 GUI Development

Microsoft Visual Studio is used as the GUI development environment for this project because it is an open source software. It provides an integrated development environment which is user-friendly, especially with features such as IntelliSense which supports code completion and syntax highlighting.

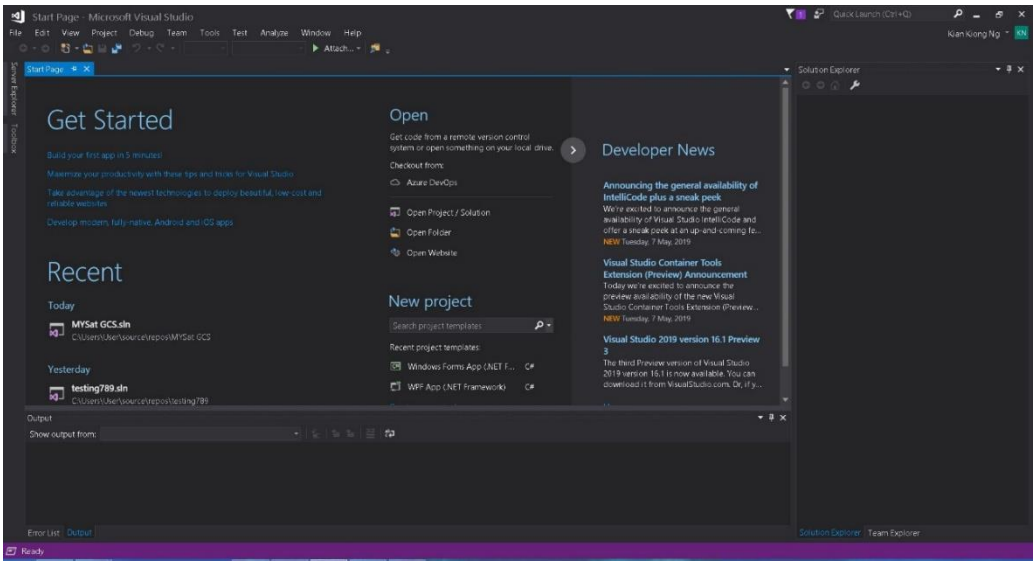

Figure 3.9 : General Interface of Microsoft Visual Studio

Microsoft Visual Studio supports different kind of programming language such as Visual Basic (VB), C++ and C#. It also allows development of various form of applications such as Windows Form Application, Web Application and Console Application. However, this project is developing a Windows Form Application based GUI in C# programming language, since the ground control station software to be developed is mainly used with computer.

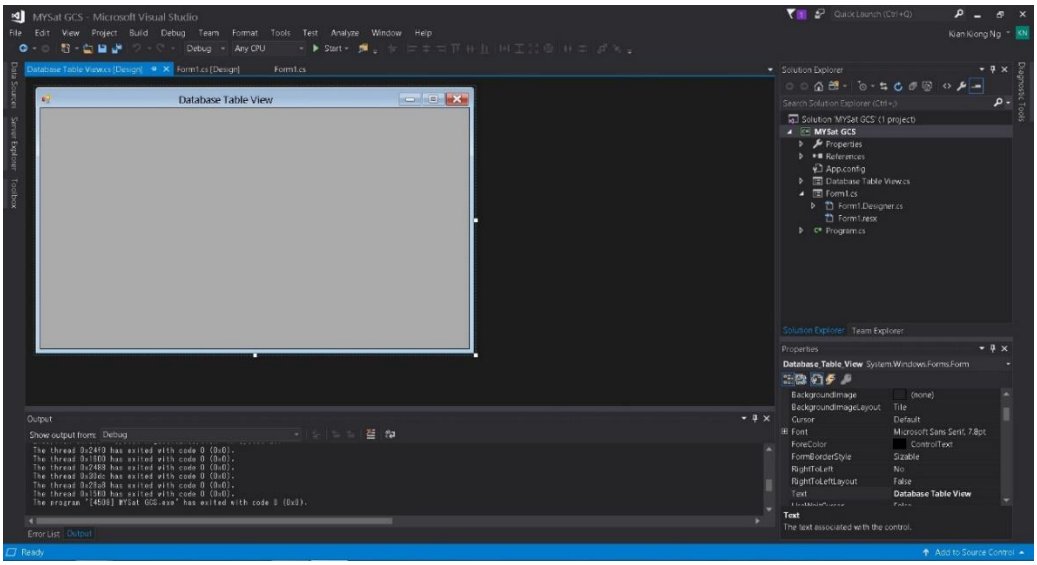

Figure 3.10 : Design Interface of Windows Form Application

Figure 3.10 shows the interface for the design of Windows Form Application in Microsoft Visual Studio. This panel is where the GUI is designed based on personal preferences and requirement, in order bring out a GUI that meets the expectation of being user-friendly and interactive.

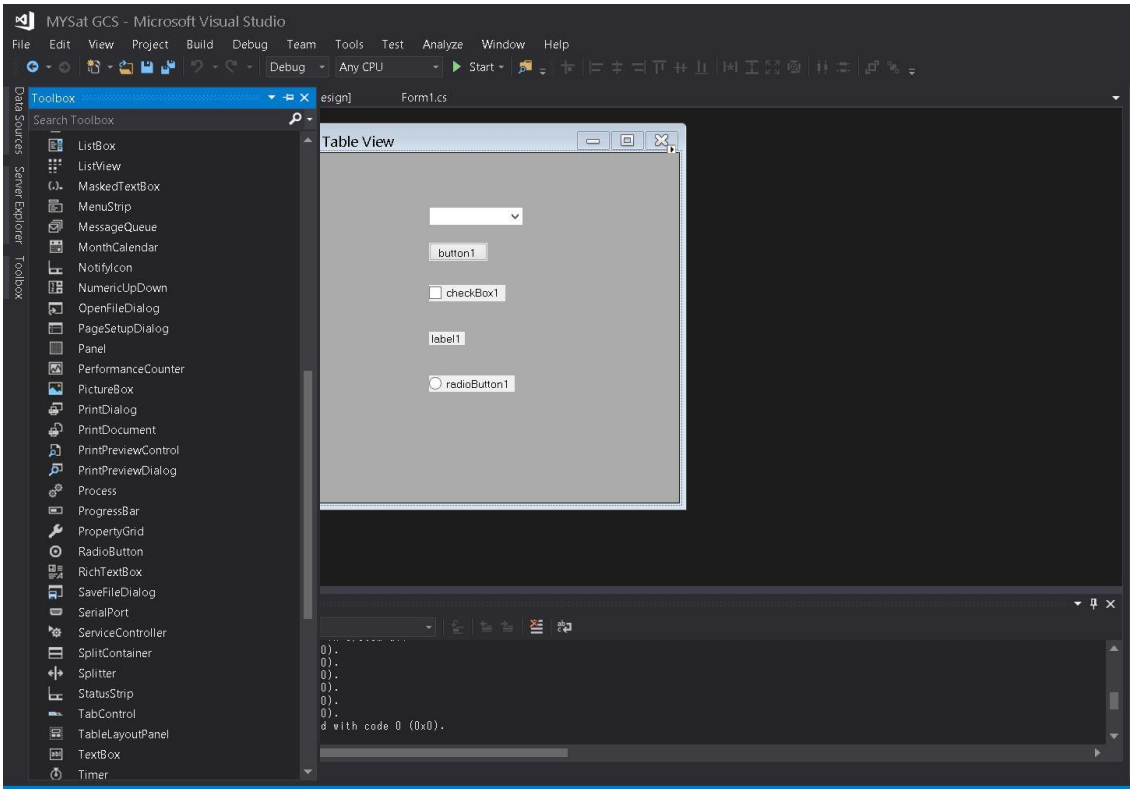

Figure 3.11 : Toolbox Panel from Design of Windows Form Application

Figure 3.11 shows the toolbox panel where interactive interface features such as buttons, checkboxes and labels reside. These tools can then be added into the design interface based on each design's requirement and functionality.

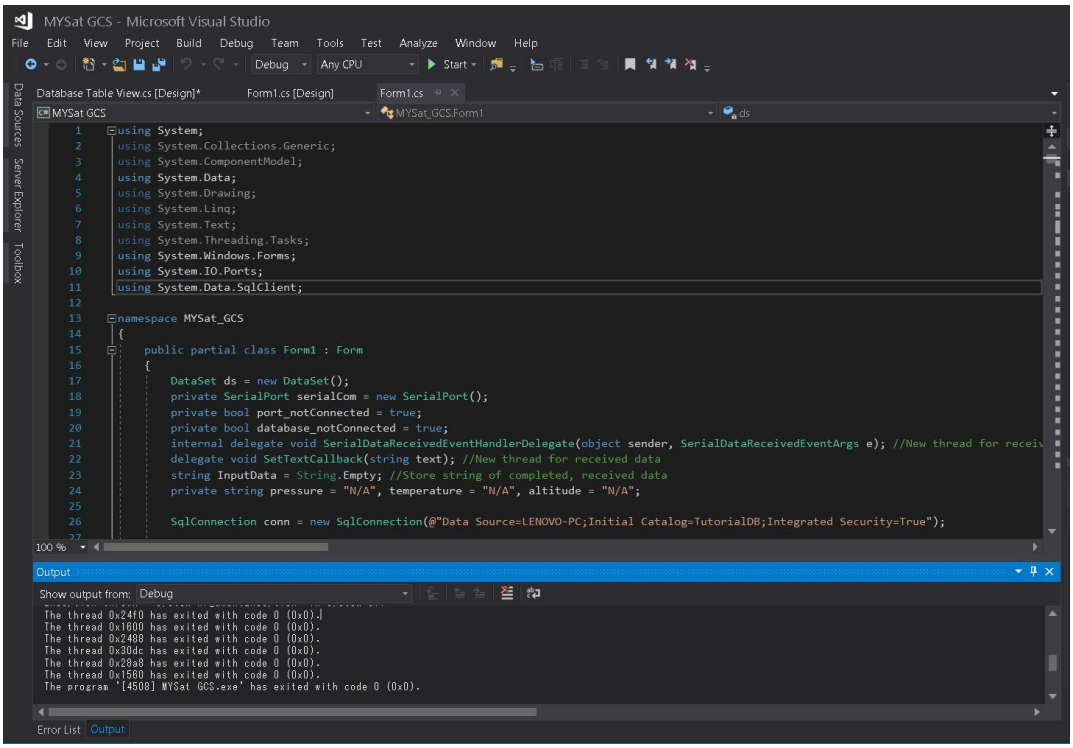

Figure 3.12 : Programming Panel in Microsoft Visual Studio

Figure 3.12 shows the programming panel of Microsoft Visual Studio. Programming panel is where coding is written in order to make the GUI functional. Each interactive interface requires their very own coding written in the programming panel for them to be functional in performing certain tasks.

Microsoft Visual Studio supports various extensions as well, including the support of Microsoft SQL Server. Therefore, connection between the created database and developed GUI can be easily made within the programming panel. Certain properties of the database need to be specified for ensuring a successful connection between two different software, such as database engine to connect and table to be updated. Therefore, even minor properties like table column names are crucial in ensuring a smooth connection between two software.

3.3.4 Data Flow

There is a certain flow of data before the collected data is successfully displayed in the developed GUI and recorded in the created database.

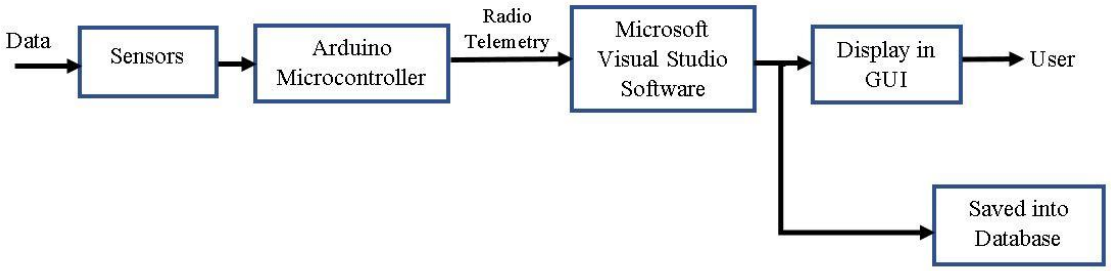

Figure 3.13 : General Data Flow

Figure 3.13 shows the general data flow after collected from the sensors. Data is first detected by sensors and processed in the Arduino Microcontroller. It is then sent out through radio telemetry signal to the designated computer device. The developed software in the computer will then processed the signal into its very own desired format before finally being displayed in the developed GUI. The developed software also processes the data into a format which can be saved into the database created.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Database

A database named “MYSat” is created for this project, along with 6 different tables namely “table_date”, “table_housekeeping”, “table_temperature”, “table_power”, “table_solar”, “table_payload”.

Figure 4.1 : Created Database and Tables

LENOVO-PC.MYSat - dbo.table_date

Column Name	Data Type	Allow Nulls
Date	varchar(50)	☐
		☐

Figure 4.2 : "table_date" properties

LENOVO-PC.MYSat...able_housekeeping

Column Name	Data Type	Allow Nulls
Date	varchar(50)	☐
Time	varchar(50)	☐
[Pressure (Pa)]	varchar(50)	☑
[Altitude (m)]	varchar(50)	☑
[Range (m)]	varchar(50)	☑
[Azimuth (°)]	varchar(50)	☑

Figure 4.3 : "table_housekeeping" properties

LENOVO-PC.MYSat -...table_temperature

Column Name	Data Type	Allow Nulls
Date	varchar(50)	☐
Time	varchar(50)	☐
[Satellite Temperature (°C)]	varchar(50)	☑
[Battery Temperature (°C)]	varchar(50)	☑
[Transmitter Temperature (°C)]	varchar(50)	☑
[Mission Board Temperature (°C)]	varchar(50)	☑
[Solar Panel X Temperature (°C)]	varchar(50)	☑
[Solar Panel Y Temperature (°C)]	varchar(50)	☑
[Solar Panel Z Temperature (°C)]	varchar(50)	☑

Figure 4.4 : "table_temperature" properties