**Chapter 1: Introduction**

* 1. introduction to the main area of the project

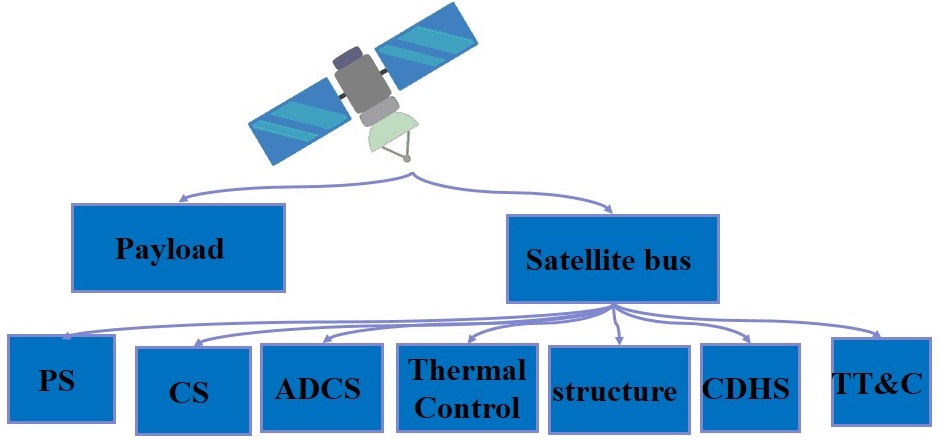
Space industry is our domain as its one of the most important industries in the modern age and used to measure the advancement of nations in the world. Egypt will launch the first satellite created and manufactured by Egyptian hands. In this project, a computer program will be developed to decrypt the data sent from the satellite to the ground unit. This program will unpack the decrypted telemetry packets sent by the satellite in each session and show the data sent in readable form to the responsible engineers. The Graphical User Interface will show these sensor reads and make some charts and graphs that demonstrate the health of the satellite through the sensors

**Overview**

* Satellite Architecture

A satellite consists of a payload, which is the mission-specific equipment, and a collection of subsystems is called bus. Satellite bus is a group of components that support a common function. There is a difference between the payload and the rest of the satellite bus, because the payload is typically unique for a given mission, whereas the bus may be able to support different missions.

A bus typically consists of Power Subsystem (PS), Communication Subsystem (CS), Attitude Determination and Control Subsystem (ADCS), Thermal Control Subsystem, Structure Subsystem, Command and Data Handling Subsystem (CDHS), Telemetry, Tracking and Command Subsystem (TT&C).



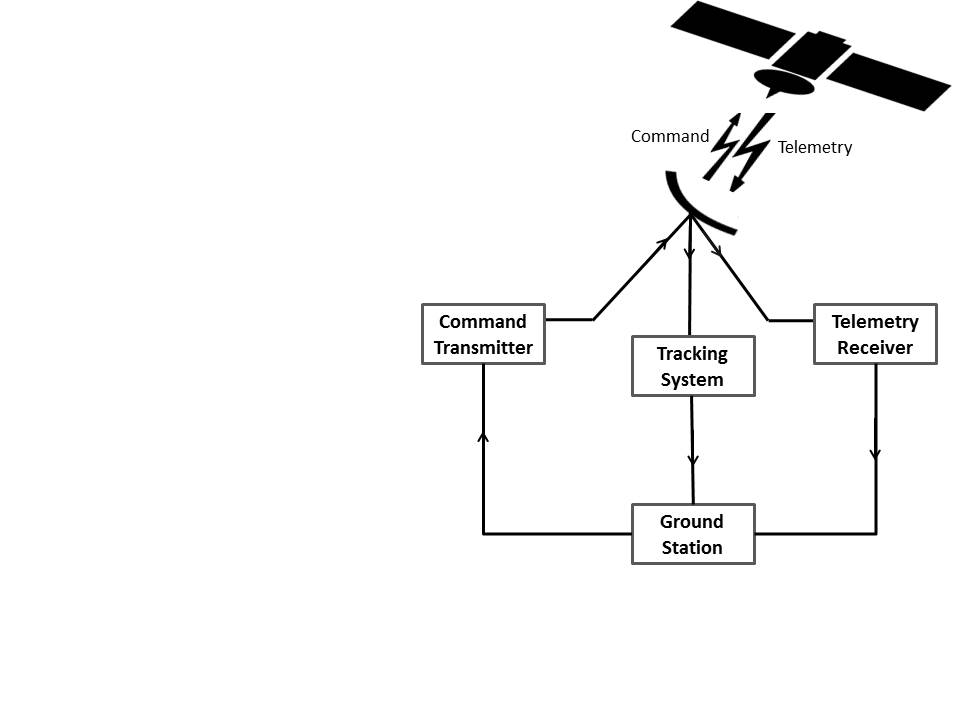
**Fig. 1.** Satellite Architecture

The satellite consists of two main components which are the payload and satellite bus. The payload is the main mission of the satellite. The satellite bus consists of many systems which are power source System, communication System, Attitude determination and control System (ADCS), Telemetry System, Thermal System, Structure System, On Board Computer System (OBC) and Propulsion System.

1. The power system is responsible for the management of the power generated by the solar cells and stored in the batteries in the satellite to certain levels to maintain the availability of power when the satellite needs it.
2. The communication system is responsible for the communication with the ground station on earth.
3. The Attitude determination and control System (ADCS) is one of the most important systems in the satellite as it responsible for many tasks:
   1. In the first launch it stabilizes the satellite by damping the angular velocity and initializing the construction of satellite attitude.
   2. Changing the attitude and orientation needed to capture image.
   3. Keeps the orientation to stay in touch with the command with low accuracy and low power consumption.
   4. The execution of the ground station order of the desired attitude and orientation. The determination of the position of the satellite is done in the ADCS system using Magnetometer, Sun Sensor and Star Sensor. In addition to this the changing in the attitude or the orientation of the satellite is done in the ADCS by the Magneto Torque (MT) and The Reaction Wheel. There several operational modes of this system which are IAA (DE tumble mode) (Damping mode), SB (Stand By mode), PTM (Imaging mode), HAAC and EM (Emergency mode).
4. The Telemetry system is responsible for combining all the reads of all the system into the form of the packet sent from the satellite to the ground station.
5. The Structure system is responsible for measuring the health of the components of the satellite.
6. The On Board Computer System (OBC) is the brain of the satellite which manages all the systems above and it controls the satellite by the orders sent from the ground station.
7. The propulsion system is responsible for the control of the satellite by the external thrust generated for the satellite movement to change speed or to maintain certain condition.

* Telemetry, Tracking and Command (TT&C) Subsystem

A communication between the satellite and the ground station is needed. The primary goal of it is to provide a link to transfer the health status information to the ground station then returns back to the satellite with a proper command or task from ground station. These functions are performed by Telemetry, Tracking and Control (TT&C), which is the brain subsystem of satellite provides a connection between the satellite itself and the facilities on the ground. The TT&C subsystem is required for all satellites regardless of the application to ensure the satellite performs correctly.



**Fig. 2.** Main Functions of Telemetry, Tracking and Control Subsystem

* Telemetry Data

During the artificial satellite operational lifetime the ground station is receiving the telemetry data, is non-stationary time series dataset contains thousands of sensor measurements from various subsystems, which contains the wealth information related to the health and status of the entire satellite and all its subsystems which reflect the operational status and payload of satellites. The health and status measurements of the satellite include the status of resources, the health and mode of operation for each subsystem and environmental data like values of sun and radiation or like star trackers. The telemetry data is analyzed in the ground control station for the health monitoring purposes such as failure diagnostic or prognostic, and anomaly detection.

There are two type of telemetry data:

* Housekeeping Data: Instrument health, safely and state parameters, are transmitted to ground in periodic telemetry data.
* The data for which the satellite was launched, such as photographs of some of the areas that were claimed, mineral, etc.
* Ground Station:

The earth station is the ground part of the satellite system. Its main functions are transmitting and receiving traffic signals to and from satellites. Note that transmission from an earth station to the satellite is called uplink (Command), while transmission from the satellite to the earth station is called downlink (Telemetry).

After reception of the earth station for satellite data in the period of communication between the station and the satellite, which can be more than once in some industrial satellites and at most once in others, and this data is a frame, frame from a number of packets. Each satellite has its own protocol to connect to the earth station and to send data to the ground to be protected. This data has its own structure even if it is received from anyone who is not responsible cannot benefit from it and know that information.

The problem facing the ground station is that it wants to decode the structure of each packet and convert it to the concept information of the space engineers and draw some on the charts.

* 1. Motivation

Satellites are expensive to develop in both time and money. They cost millions of dollars and years of time. That made them monopolized by rich countries. Also, universities were far from this important industry. By introducing small satellites known as CubeSats, this monopolization was broken and universities and developing countries could participate in space industry.

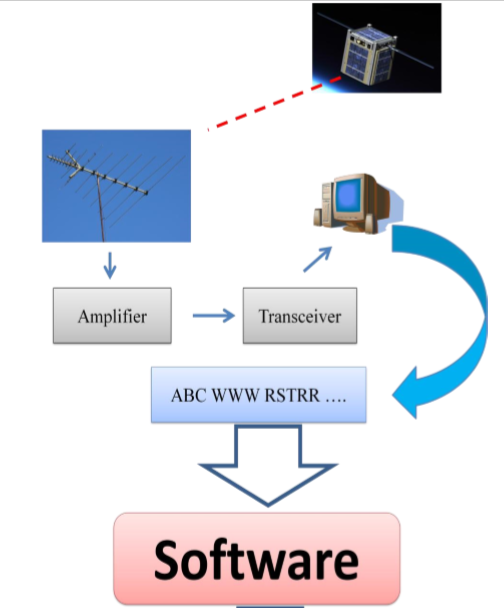
In the past space missions never used up to date technology in their spacecraft due to the long development time; the used technology becomes old when the satellite is launched,

Any spacecraft must communicate with its ground station. It sends the data of the mission that it was made to do and data about its health. This data is usually sent encoded. The data is decoded at the ground station using decoder software. Such a decoder was built in this project.

* 1. Problem definition

The main problem is the unpacking and the understanding of the telemetry data sent by the satellite and processing it into readable and understandable information.

The satellite contacts the ground station on average 2 times a day. On contact the satellite send enormous amount of packets which represent a huge amount of data which is impossible to be monitored by eye the led to the need of processing and accurate unpacking and storing in the database and fast retrieving too.



* 1. Project Objective (suggested solution)

. We solved this problem by developing a software that unpacks the packet received from the satellite and a database that we store the unpacked data in. The database is designed that we can retrieve the desired sensor reads using the session number and the specific time. Finally the GUI (graphical user interface) that shows the responsible engineer the sensors’ reads, charts and graphs to facilitate the satellite monitoring.

In the case of EGYSAT1 the Ukrainian model was a solution but on the other hand it was a black box to us. The advantage was that it was compatible with the EGYSAT1 and it had a lot of options. The disadvantage was that we couldn't do more that was permitted by the Ukraine.

It’s not secure to have a satellite software designed by other country and it’s a black box to us.

* 1. Gantt chart of project time plan
  2. Project development methodology

Waterfall

* 1. The used tools in the project (SW and HW)

We are using java for the internal implementation as its code can run on all platforms that support Java without the need for recompilation using Net Beans IDE 8.2. We used also MYSQL for the creating the database developing it using EasyPHP Deserver , JavaFx also used for User interface design

* 1. Report Organization ( summary of the rest of the report)

**Chapter 2: Related work**

(The nearest examples of the project and the main differences between them and your project (if found) take care about references)

1. **FUNcube-1 (AO73)**

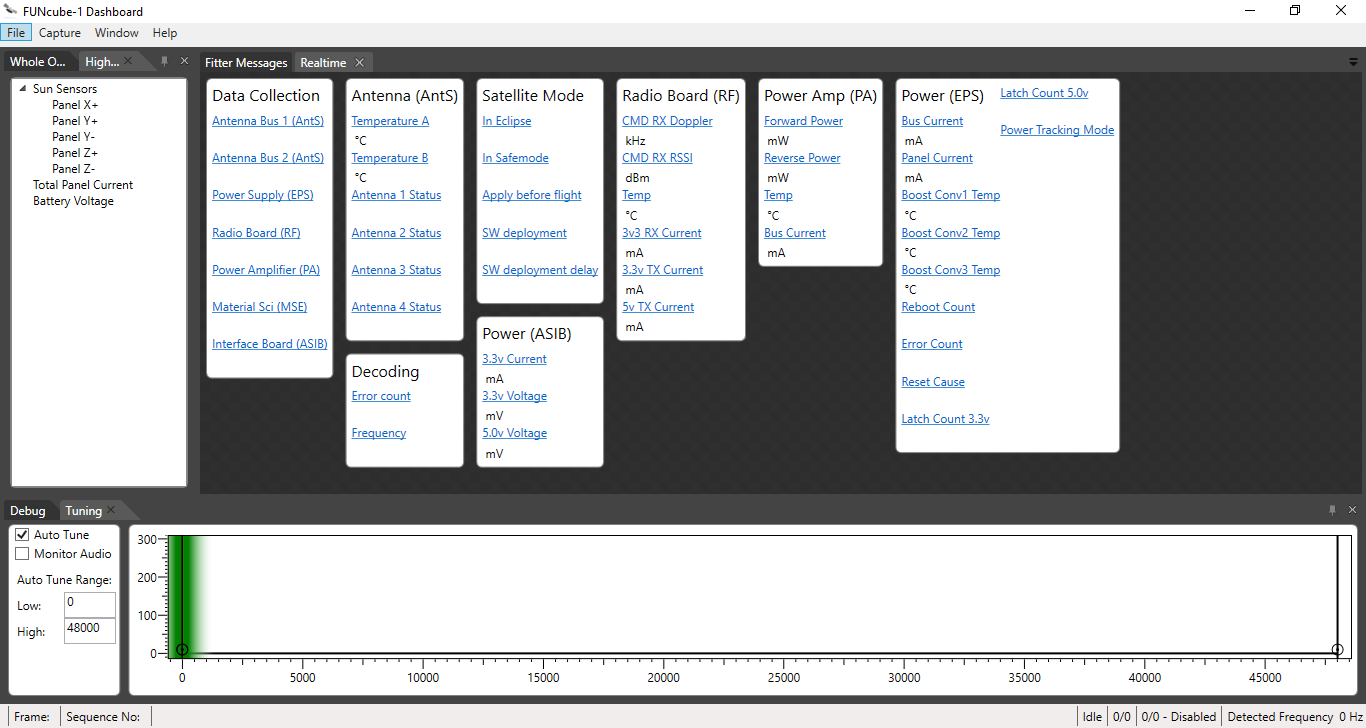
The FUNcube-1 Dashboard user interface enables the display of telemetry, debug data and Fitter messages from the FUNcube-1 (FC1) spacecraft. It can also upload the received data to the FUNcube Data Warehouse over the internet.

The Dashboard can accept live inputs from a directly connected FUNcube or by audio from another SSB radio fed into the computer soundcard. It can also display previously recorded data from IQ “WAV” files or from the “.funcubebin” recordings that it can create during operation.

FUNcube-1 Dashboard

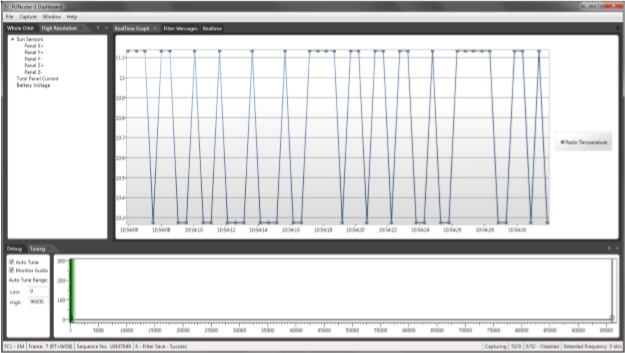
The first screen you will see is shown in Figure 1. There are eleven separate sets of telemetry data displayed on the initial Dashboard screen along with a summary set (“Telemetry Decoding”) which gives details on how successful the data decoding is progressing. The last field to appear is usually the “Power Tracking Mode” field at the bottom of the EPS frame on the right hand side or may be the Telemetry Decoding details. This will depend upon your screen resolution etc. Please ensure you resize the Dashboard so that all the data is visible on your screen.

The initial display shown in Figure 1 is the “RealTime” data. Each box of data represents different segments of the telemetry. Do not be concerned by the erroneous numbers that initially appear in some fields before any real data has been received.



WOD/HR Panel : Whole Orbit tab

WOD contains specific channels of data, sampled on the spacecraft at one min intervals. These telemetry channels can be displayed in graphical form by selecting one of the entries in the list. In flight, the spacecraft will store 104 minutes of data. Figure below shows an example of some WOD displayed as a graph.



Normal Operation of the FUNcube Dashboard

The FUNcube Dashboard is designed to require very little interaction to receive, record and decode the telemetry from the FUNcube Satellites.

The key steps to making the Dashboard operate and decode data are:-

1. Ensure that the Audio Configuration is setup as described in this document.

2. If you have registered at the FUNcube Warehouse, select the “Stream to Warehouse” option in the Warehouse settings according to whether you wish to submit data to the Warehouse.

3. From the “Capture” menu, select the “Write capture data to disk” (so you can save the data for replay at a later date).

4. From the “Capture” menu, select the source for your audio.

**The main different** is the Funcube Dashboard is working on static subsystems and sensors as user can’t change anything in the number of subsystems or sensors

And also can’t change sensors information or add more sensors in subsystem

But Funcube Dashboard has more functionality like debug and tuning and too many related graphs

Their website : <https://funcube.org.uk/>

Another Examples:

1. PolySat

In 1999, a team designed, constructed, and tested CP1, Cal Poly‟s first satellite . CP1 communicated on amateur radio frequencies using a combination of Morse Code and Dual Tone Multi-Frequency (DTMF) to encode data. Morse Code is used to identify transmissions while allowing operators to tune to the correct frequency, and DTMF data is sent at 15 characters per second. Despite its relative simplicity, the CP1 communication system was highly efficient .

1. AAU1 CubeSat

It was developed by students of Aalborg University in Denmark. The system used a 9600 baud rate for communications. This satellite used a Mobitex packet encoding scheme underneath standard AX.25. These packets contained telemetry data, but could not be decoded by regular amateur radio operators due to the Mobitex packet encoding. This satellite beaconed every two minutes if the on-board computer was not functioning, and every four minutes in a low battery situation. Ground stations only received about 1 kB of data .

1. CUTE-1

CUTE-1 is a Japanese CubeSat developed by sixteen graduate and undergraduate students at Tokyo Institute of Technology Laboratory for Space Systems. One of the primary mission goals was to test two different implementations of downlink communication protocols.

3.1 Project specification

3.1.1. Functional requirement

1. The software is to decode this text file to its corresponding values.
2. After decoding and processing data they are to be displayed to the user using GUI.
3. Sensors values are to be displayed using charts.
4. Data saved into the DB so that user can view them anytime.
5. If the data decoded indicated that the satellite may be in danger, the software is to alarm the user.
6. Initialize section enable the user to initialize the sensors in every subsystems
7. Displaying values of the sensors of all subsystems
8. Design window to display specific sensors
9. Show data mining result

3.1.2. Non-functional requirement

Usability:

UI enable the user to see all sensors values in the required packet in specific subsystem or in chosen sensors needed

Can also make charts of that values to make it more readable and make decisions on it

Modifiability:

System can be generic for more than one satellite

As all needed parameter will entered by the user

Security:

Using 3 Tier architecture makes User doesn’t have direct access to the database and logic layer

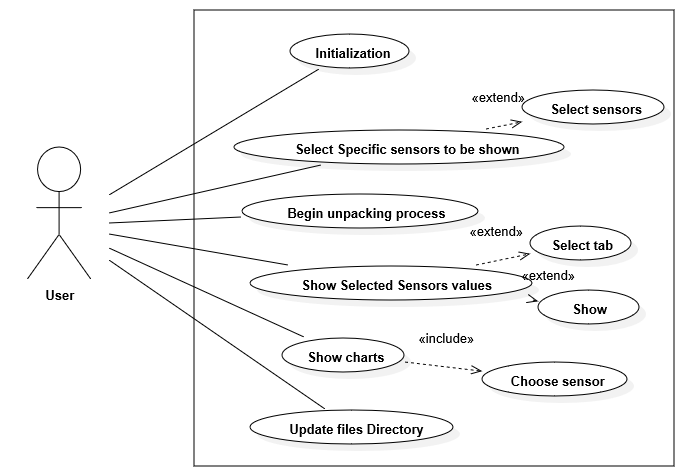
Performance:

Performance involves things such as throughput of information through the system, system response time (which also relates to usability), recovery time, and start-up time.

Robustness:

The ability of a system to maintain a function. Even if the user enters a wrong input, the program is not crash and handle this situations. Even if there are changes in the environment.

3.2. Use case Diagrams



|  |  |  |
| --- | --- | --- |
| Use Case ID | 1 | |
| Use Case Name: | initialization | |
| Actors: | User | |
| Brief Description | This use case allow the user to enter sensors information | |
| Pre-conditions: | Opening the software | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1- Press initialization button |  |
|  | 2- Ask user to fill the required initialization filed |
| 3- fill the required initialization filed for every sensor  4 – press button show |  |
|  | 5 – Show the entered data for every sensor in the table |
| 6 – Press save |  |
|  | 7- Save sensors information into DB |
| Includes: |  | |
| Notes and Issues: | 1. User can remove sensor after showing it the table 2. Initialization process has limitation on the sensors number if user exceed it warning notification will appear | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 2 | |
| Use Case Name: | Select Specific sensors to be shown | |
| Actors: | User | |
| Brief Description | This use case allow the user to select specific sensors to be shown in the unpacking process | |
| Pre-conditions: | Opening the software | |
| Post-conditions: | Make new tab | |
| Flow of events: | **User Action** | **System Action** |
| 1. Press select sensors key |  |
|  | 1. Ask user to fill required filed |
| 1. Choose sensors 2. Write tab name 3. Press save key |  |
|  | 1. Save tab name and its sensors in DB |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 3 | |
| Use Case Name: | Begin unpacking process | |
| Actors: | User | |
| Brief Description | This use case show the value of each sensor in the subsystem for all the packets in the selected file | |
| Pre-conditions: | Update files Directory | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1- Press Begin unpacking process key |  |
|  | 2 – Show the equivalent value for every sensor in each subsystem |
| Includes: |  | |
| Notes and Issues: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 4 | |
| Use Case Name: | Show Selected Sensors values | |
| Actors: | User | |
| Brief Description | This use case allow the user to show specific sensor that had been selected | |
| Pre-conditions: | Select Specific sensors to be shown | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1. Press Show Selected Sensors values key 2. Choose tab from the table 3. Press show |  |
|  | 1. Begin unpacking process for only selected tab sensors |
| Includes: |  | |
| Notes and Issues: |  | |

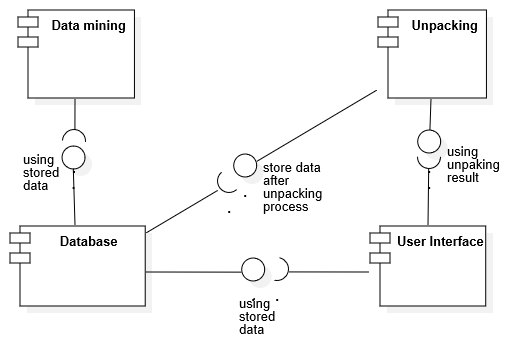
|  |  |  |
| --- | --- | --- |
| Use Case ID: | 5 | |
| Use Case Name: | Show charts | |
| Actors: | User | |
| Brief Description | This use case allow the user to show chart for specific sensor | |
| Pre-conditions: | Begin unpacking process or Show Selected Sensors values | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1. Press chart button 2. Choose sensor 3. Press show button |  |
|  | 1. Show chart for selected sensor |
| Notes and Issues: |  | |
| Includes: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 6 | |
| Use Case Name: | Update files Directory | |
| Actors: | User | |
| Brief Description | This use case allow the user to enter the file directory for files that have the packets | |
| Pre-conditions: |  | |
| Post-conditions: |  | |
| Flow of events: | **User Action** | **System Action** |
| 1. Press Update files Directory button 2. Press browse key 3. Choose file 4. Press save button |  |
|  | 1. Save file Directory |
| Includes: |  | |
| Notes and Issues: |  | |

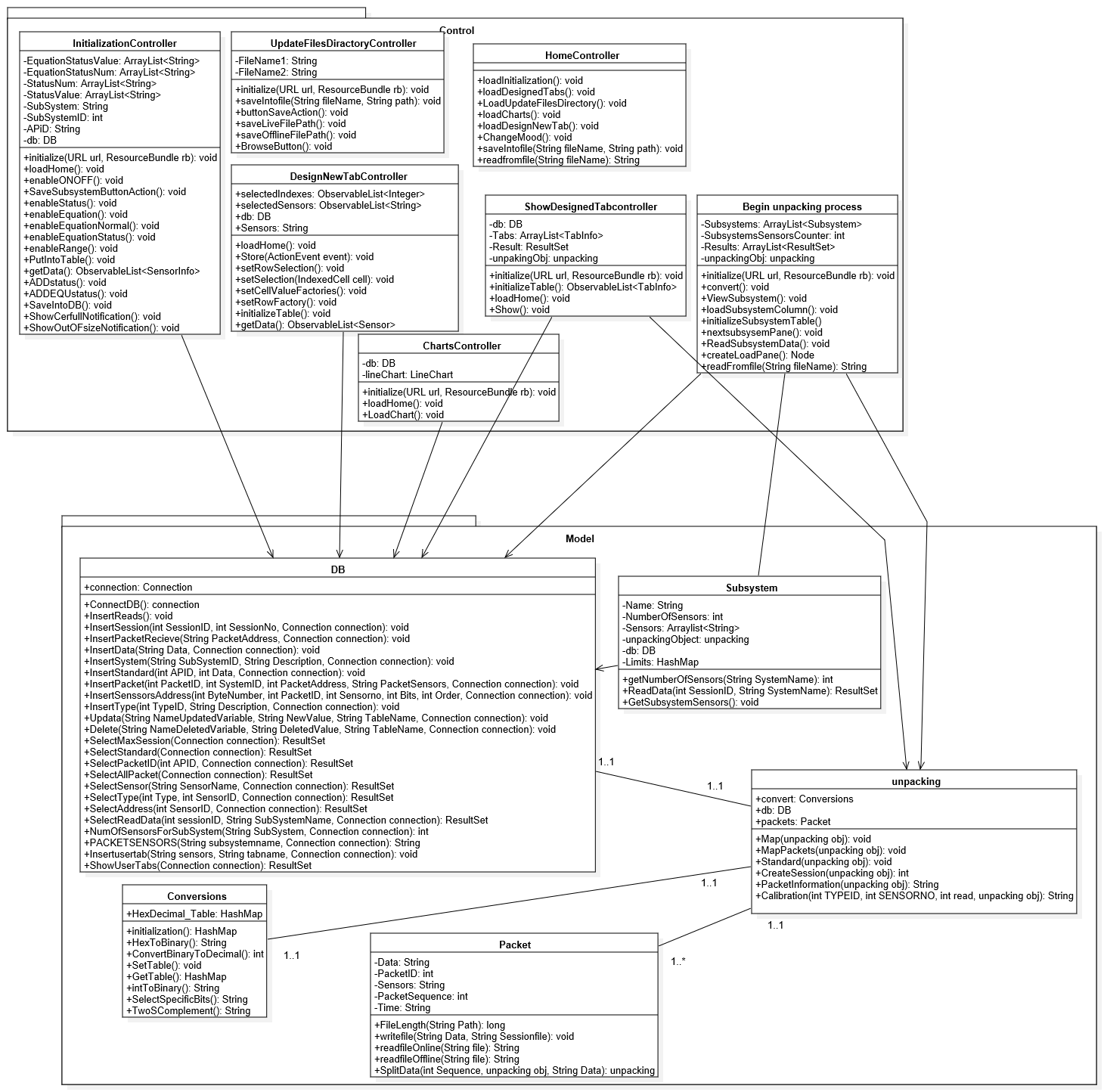
3.4. System test cases.

**Chapter 4: System Design**

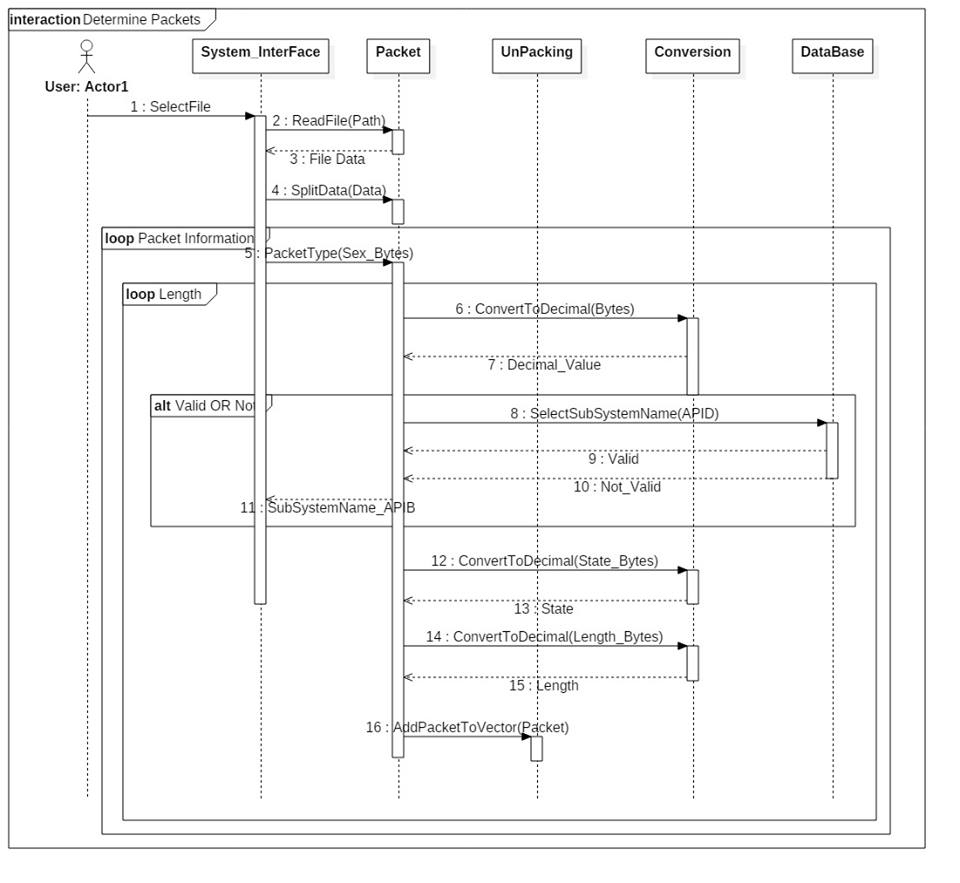
1. System Component Diagram

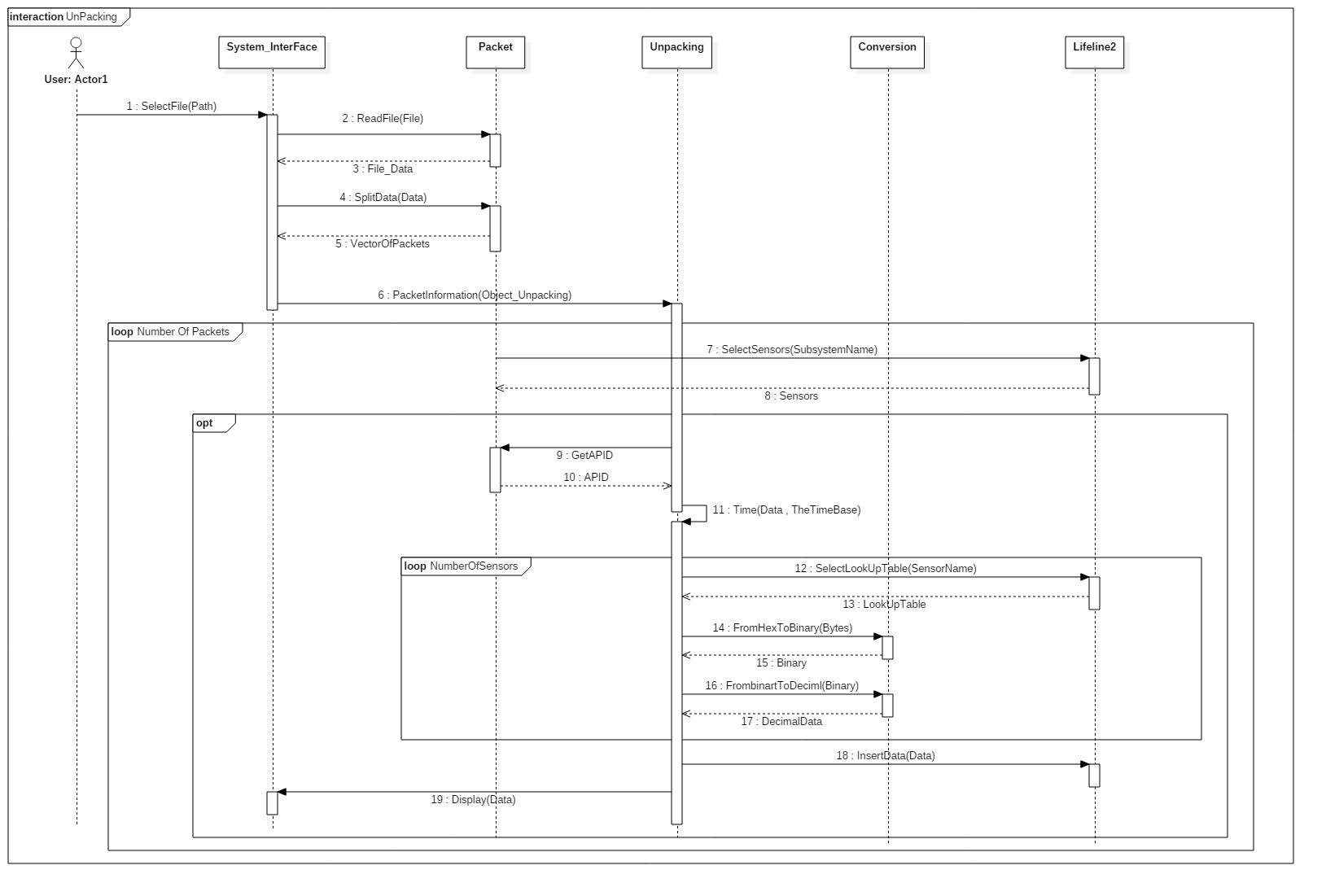


System Class Diagrams,



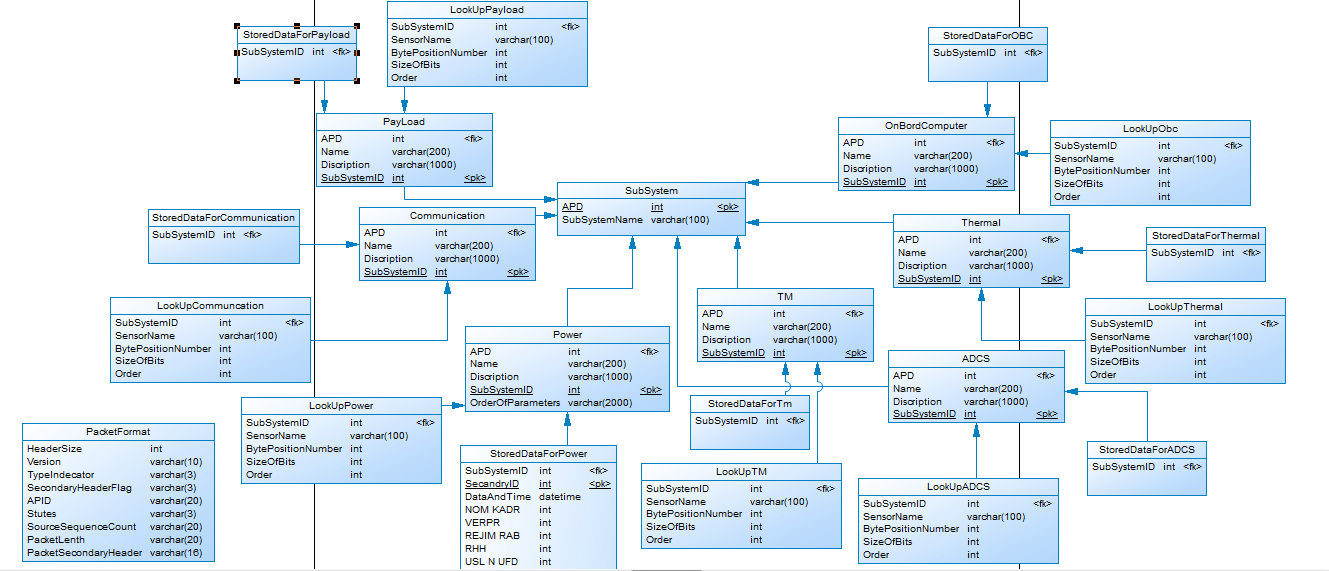
Sequence Diagrams





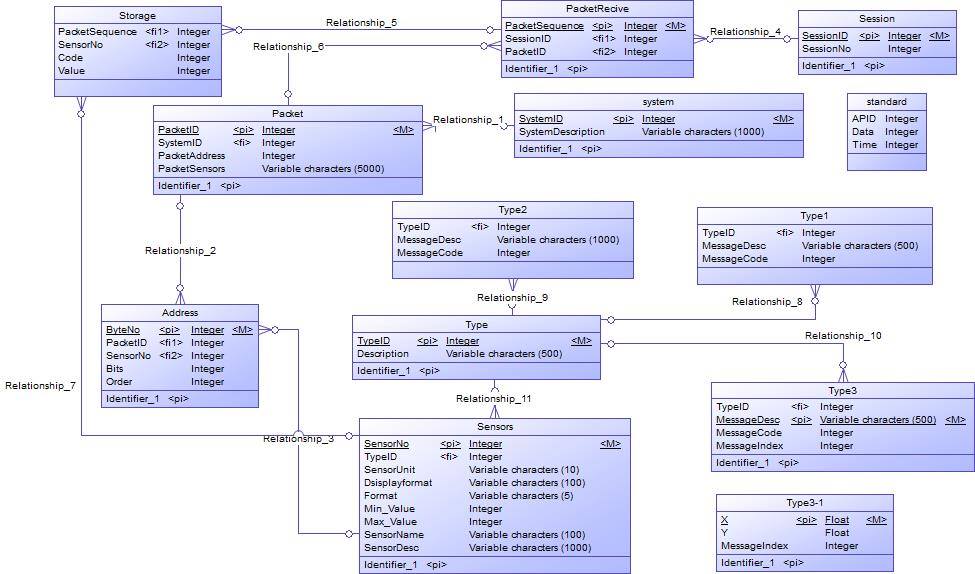
Project ERD

For module 1 (EGYSAT-1 )



For module 2

Generic module



1. System GUI Design

The code of the software was written based on the design shown in chapter 3. Since the code is huge, part of it is shown in appendix C and the complete code is attached in the CD with the thesis.

The system developed was Telemetry unpacking software that has a UI. The UI consists of:

1. Home page :

Is the first screen you will see is shown in Figure 5. There are seven separate buttons displayed on the initial screen as the menu of the program

Mood ON/OFF button

Update files directory button

Charts button

Show selected sensors button

Begin unpacking process button

Select sensors to be shown button

Initialization button



##### **Menu**:

##### Initialization

##### As the last module is a dynamic model initialization section is momentous in our software to enable the user to enter the subsystems information’s and its sensors

##### Select Sensors to be shown

##### Sometimes user don’t need to watch all the sensors values during the unpacking process so that section enable the user to choose specific sensors only to be shown and save that sensors in the database under chosen name

##### Begin Unpacking Process

##### After selecting the files directory software begin to unpack the packets in the chosen file and begin to display it to the user

##### Show selected sensors

##### In this section user can watch only the selected sensors by choosing the specific tab

##### Charts

##### User can select sensor and the time of received packets from satellite then software will show the chart of that values

##### Update files Directory

There are two type of unpacking online or offline, every type has a deferent file directory so user can enter the files directory

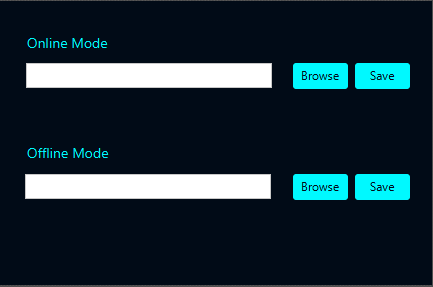
**Software functionality :**

##### **Update file directory**

##### It gives the user the ability to make an online session with the satellite or to unpack a previous stored session by choosing which file would be used

Choose file from the machine

Save it in the software



##### **Initialization**

##### UI enable the user to initialize the sensors for every subsystem by entering their information

Filed to initialize subsystem which every sensor belong to

##### C:\Users\norhan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2.png

Fields needed to initialize sensors

##### 

Table to show the all sensors before store into db

* **Select Sensors to be shown**

Also user can choose specific sensors to watch only their values during unpacking process

##### C:\Users\norhan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\4.png

List view display selected sensors

Write tab name

Table to display all sensors