GPS Monitoring Station Data Management

Héctor Manuel Maestro Alegre

Dissertação para obtenção do Grau de Mestre em
Engenharia Electrotécnica e de Computadores

Júri

Presidente: José Manuel Bioucas Dias
Orientador: José Eduardo Charters Ribeiro da Cunha Sanguino
Co-Orientador: António José Castelo Branco Rodrigues
Vogal: Francisco António Bucho Cercas

Dezembro 2010
GPS is nowadays the most important GNSS and the only one fully operational. Nowadays, this navigation system is becoming omnipresent in our society. Cars, vessels and airplanes include a GPS receiver to travel. The GPS receivers are also found on cellular phones and laptop computers.

The IT GPS monitoring station is continuously receiving data from GPS system. All this received data is acquired with the aim of being used to support the research and development of navigation algorithms. This data includes, among others, the orbital parameters (ephemerides), the GPS almanac and the satellite status. However, the monitoring station does not store permanently this data.

The aim of this project is to implement a database application to store all these values. This database uses the MySQL technology. Firstly, a reader programs are create to read these parameters and send it to the database. Users can access to the database using a website. Using it, users can create and send queries to the database. The results of these queries are shown on the website or it can be downloaded on a file. Furthermore, the application supports the real-time upload of the database as well as the download of data. A backup system is implemented to protect the data. This system creates each week a file that contains all received data.

The project involves the integration of several navigation, communication and computing equipment. The first part of this report includes a study of GPS to understand this navigation system and the parameters that are used. After this study, all the parts that compose this project are presented. These parts are: the data acquisition, the database and the website.

Keywords: GPS, database, MySQL, web programming, C, php.
List of Figures

Figure 2.1: GPS orbital inclination ................................................................. 3
Figure 2.2: Control Stations ................................................................. 4
Figure 2.3: WGS 84 system ................................................................. 5
Figure 2.4: Range determination from single source ..................................... 6
Figure 2.5: Ambiguity resulting from measurements to two sources ................. 7
Figure 2.6: Position ambiguity removal by additional measurement ................. 7
Figure 2.7: 3-D trilateration ........................................................................ 7
Figure 3.1: Scenario ................................................................................... 14
Figure 4.1: Azimuth example ..................................................................... 19
Figure 4.2: Satellite Azimuth and Elevation in degrees .................................. 20
Figure 4.3: Data acquisition from shared memory ......................................... 22
Figure 5.2: Database structure ................................................................... 27
Figure 5.1: Table creation .......................................................................... 26
Figure 5.3: insert command ........................................................................ 27
Figure 5.4: Backup script ........................................................................... 29
Figure 6.1: Laboratory network ................................................................... 31
Figure 6.2: Model-View-Controller .............................................................. 33
Figure 6.3: Structure of the website ............................................................. 34
Figure 6.4: Date selectors .......................................................................... 35
Figure 6.5: Example of minimum number of satellites ................................. 35
Figure 6.6: Observation checkboxes ............................................................ 36
Figure 6.7: Elevation & Azimuth page ......................................................... 37
Figure 6.8: Navigation page ........................................................................ 38
Figure 6.9: Navigation option .................................................................... 38
Figure 6.10: Observation page .................................................................... 39
Figure 6.11: Observation option .................................................................. 39
Figure 6.12: Results buttons ....................................................................... 40
Figure 6.13: Results page .......................................................................... 40
Figure 6.14: Download option .................................................................... 41
### List of Abbreviation

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>DoD</td>
<td>Department of Defense</td>
</tr>
<tr>
<td>ECEF</td>
<td>Earth-Centered, Earth-Fixed</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>ICRF</td>
<td>International Celestial Reference Frame</td>
</tr>
<tr>
<td>IERS</td>
<td>International Earth Rotation Service</td>
</tr>
<tr>
<td>IRM</td>
<td>IERS Reference Meridian</td>
</tr>
<tr>
<td>IRP</td>
<td>IERS Reference Pole</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MCS</td>
<td>Master Control Station</td>
</tr>
<tr>
<td>MVC</td>
<td>Model-View-Controller</td>
</tr>
<tr>
<td>NAVSTAR</td>
<td>Navigation System Time and Ranging</td>
</tr>
<tr>
<td>PHP</td>
<td>PHP Hypertext Pre-processor</td>
</tr>
<tr>
<td>RINEX</td>
<td>Receiver Independent Exchange</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal-to-Noise Ratio</td>
</tr>
<tr>
<td>SQL</td>
<td>Structure Query Language</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>UTC</td>
<td>Universal Time Coordinated</td>
</tr>
<tr>
<td>WGS-84</td>
<td>World Geodesic System 1984</td>
</tr>
</tbody>
</table>
1 Introduction

Nowadays, Global Position System (GPS) is used for many different applications. A GPS receiver provides us information about the position on any point of the Earth. This system is free and available 24 hours a day. To know this point, GPS satellites send to the receiver much information to calculate its position. All this data can be used to research purposes.

The IT GPS monitoring station has a set of GPS receivers that are continuously receiving data from the GPS system. This data is being acquired through several serial interfaces with the goal of being used to support the research and development of navigation algorithms. Among other parameters this data contains the satellite ephemerides (orbital parameters), the GPS almanac, the pseudoranges and the satellite status.

The aim of this project is the development of a database application that supports the access to the data that is continuously being acquired. A website is created to access to database easily. Users can create queries to consult the database. Website provides visual forms to create the queries. Also, the application supports real-time download of the database as well as the upload data. A backup system is created to protect the data stored on database. This system creates each week a file that contains all the received data.

The project development involved the integration of several navigation, communications and computing equipment. This integration was done by software and involved among others the following tools: C programming, databases and network and web programming.

Organization

After this introduction, the report is structured as follows.

The second chapter contains the background study. It is focused into GPS, its principles and its messages. This chapter also contains a brief explanation about the technologies used.

Chapter 3 presents the different GPS data stored on the database and how it is acquired.

On chapter 4, the database is explained. This chapter explains the different database tables where values are stored. The backup system is explained also on this chapter.

On chapter 5, the website of the project is detailed. This chapter includes all web pages created and how users can consult the database.

The last two chapters are the conclusions of this project and the bibliography used on it.
2 Background

2.1 GPS

From immemorial times, people have travelled around the world looking for new lands, transporting, trading products with neighbour communities, or, simply, travelling for fun. These are the origins of the navigation, the art of moving between two points following a known path.

Recent progress in space technologies has prompted a revolution in the navigation methods. It is hoped that before 2020 the GNSS (Global Navigation Satellite Systems) will replace the majority of the current navigation systems. Nowadays, GPS (Global Positioning System) is the most important GNSS and the only one fully operational, providing different sorts of service to civilian and military users.

2.1.1 History

At the beginning of 60s, American Department of Defence (DoD) was interested on the development of a navigation satellite system. These satellites could determine the position of one point on Earth surface. The main objectives of DoD were a global coverage, a continuous operation and high precision.

The first system of global positioning created was Transit, and it was operational on 1964. Despite of some applications used it, this system did not fulfil the objectives. But Transit helped to investigators to acquire useful knowledge to develop GPS.

To fulfil the same objectives of Transit, in 1970 DoD started to develop a new system named NAVSTAR (Navigation Satellite with Time and Ranging) – GPS.

Officially GPS was declared completely operational in 1995. This means that a minimum of 24 satellites are in orbit and work correctly.

The GPS System was initially designed to give service to the US army. But, since first of May of 2000, civil users were allowed to access to the error-free signal of GPS. [1]

2.1.2 General Description

The Global Positioning System (GPS), also called Navstar, is a space-based global navigation satellite system that provides reliable location and time information in all weather and at all
times and anywhere on or near the Earth. It is maintained by the United States government and is freely accessible by anyone with a GPS receiver.

The system can be divided in three segments: space, control and user.

- **Space segment**

  Space segment is composed of the orbiting GPS satellites. This segment consists of 24 satellites in six equally spaced orbit planes. Each satellite circles the Earth twice a day with period of 11h 58' (half sidereal day), approximately 20 200 km above the Earth's surface. The orbits are tilted to the earth's equator by 55º, and the orbital planes are spaced 60º apart to give coverage to the whole world. Figure 2.1 shows the orbital inclination of GPS satellites.

![Figure 2.1: GPS orbital inclination](image)

- **Control segment**

  The control segment of the GPS system consists of a worldwide network of tracking stations, with a master control station (MCS) located in the United States at Colorado Springs. The primary task of the operational control segment is tracking the GPS satellites in order to determine and predict satellite locations, system integrity, the satellite almanac, and other considerations. Then, this information is packed and it is uploaded into the GPS satellites.
• **User segment**

The users segment includes all military and civilian users. With GPS receiver connected to a GPS antenna, a user can receive the GPS signals to determine his or her position. GPS is currently available to all users worldwide at no direct charge. [1]

### 2.1.3 Principles of GPS

On this chapter the principles of GPS are explained to understand how it works. The main features of this system and the GPS standards are presented. With these parameters, a receiver can calculate its position.

#### 2.1.3.1 World Geodetic System

Firstly, the standard of the coordinated system that GPS used is explained. This standard does not model the Earth as a sphere. Instead, GPS models the Earth as an ellipsoid. The standard ellipsoid model used is the DoD’s World Geodetic System 1984 (WGS-84). [1]

WGS-84 is a right-handed, Earth-fixed orthogonal coordinate system and it is shown in Figure 2.3.
WGS-84 specifications:

- Origin fixed at the Earth’s centre of mass.

- Z-Axis = The direction of the IERS Reference Pole (IRP). This direction corresponds to the direction of the Conventional Terrestrial Pole (CTP).

- X-Axis = Intersection of the IERS Reference Meridian (IRM) and the plane passing through the origin and normal to the Z-axis.

- Y-Axis = Completes a right-handed, Earth-Centered Earth-Fixed (ECEF) orthogonal coordinate system.

### 2.1.3.2 GPS Time

While most clocks are synchronized to Coordinated Universal Time (UTC), the atomic clocks on the satellites are set to GPS time. The difference is that GPS time is not corrected to match the rotation of the Earth, so it does not contain leap seconds or other corrections that are periodically added to UTC. GPS time was set to match UTC in 1980, but has since diverged. GPS week zero started at 00:00:00 UTC on January 6, 1980.

As opposed to the year, month, and day format of the Gregorian calendar, the GPS date is expressed as a week number and a seconds-into-week number. Week number is the number of weeks since 1980, and seconds-into-week is the number of seconds starting on Sunday each week. [1]
2.1.3.3 How does it work?

Summarizing, GPS utilizes the concept of time-of-arrival ranging to determine user position. This means that the only thing that the users are trying to do is determine how far they are from any given satellite.

A receiver on the ground passively receives each visible satellite's radio signal. The receiver rebuilds the message. Then, the receiver measures the time that it takes for the signal to travel. This time-of-arrival is then multiplied by the speed of the signal (in this case, the speed of light) to obtain the emitter-to-receiver distance.

In essence, GPS operates on the principle of trilateration. In trilateration, the position of an unknown point is determined by measuring the lengths of the sides of a triangle between the unknown point and two or more known points (in this case, the satellites). This way, we are able to pose the problem geometrically.

Firstly, an ideal example of two-dimensional positioning is provided. Imagine that we are in the middle of the sea (without height the problem becomes 2D) and we want to determine our position from a foghorn. Assume that our clock is perfectly synchronized with the foghorn time base and the foghorn position is known.

![Figure 2.4: Range determination from single source](image)

With only one measurement, we know that we are somewhere on a circle with radius R1 centred about the Foghorn 1.

But, if we are able to simultaneously measure the range from a second foghorn, therefore, our position relative to the foghorns is one of the intersections of the range circles (Figure 2.5).
To solve this ambiguity, additional information sources must be added. For example, a third foghorn marking a new ranges measurement, as shown in Figure 2.6. With this third foghorn, position ambiguity is removed.

Our 3-D trilateration problem is basically the same idea as 2-D trilateration. But in this case we need to imagine 3 or more spheres instead of 3 circles, as shown in Figure 2.7.

In this case, we still have ambiguity between two points; however, it can be observed that the candidate locations are mirror images of one another with respect to the plane of the satellites.
But, we also have a 4th sphere handy: the Earth itself. Only one of the two intersecting points we have just identified will actually be on or near the Earth’s surface. Then, assuming that we are not floating around somewhere in space, we could know exactly where we are.

However, GPS receivers normally use at least 4 satellites to improve accuracy and, also, to solve another ambiguity caused by the clocks.

### 2.1.3.4 Pseudorange

The pseudorange is a measure of the range, or distance, between the GPS receiver and the GPS satellite.

The procedure of the GPS range determination can be described as follows. If satellite and receiver clocks are perfectly synchronized, multiplying the travel time by the speed of light gives the range between the satellite and the receiver.

Regrettably, in the real world, things are not as easy as explained in the former example. In order to keep very accurate time, each satellite carries on board two rubidium and two caesium atomic clocks. This is certainly accurate, but practically impossible to implement for ground-based receivers. Receivers on Earth usually have quartz clocks with much lower accuracy; therefore, an error is occurred.

Supposing that the satellites’ clocks are perfectly synchronized, we have another unknown which is the offset time of our receiver clock regarding the satellites’ clock. Our problem has become a problem with 4 unknowns’ problem: X, Y, Z and “receiver clock offset”. That explains why in GPS at least four satellites are required to allow the positioning (if every satellite represents an equation, the problem is an equation system formed by four unknowns and four or more equations, one per available satellite).

### 2.1.3.5 GPS signal

Each satellite transmits a microwave radio signal composed of two carrier frequencies modulated by two digital codes and a message. The two carrier frequencies are generated at 1,575.42 MHz (L1) and 1,227.60 MHz (L2). All of the GPS satellites transmit the same L1 and L2 carrier frequencies. However, the code modulation is different for each satellite, which significantly minimizes the signal interference.

The two GPS codes are called coarse acquisition (C/A-code) and precision (P-code). Each code consists of a stream of binary digits, generated using a mathematical algorithm. Presently,
the C/A-code is modulated onto the L1 carrier only, while the P-code is modulated onto both the L1 and the L2 carriers.

The C/A-code is a stream of 1,023 binary digits that repeats itself every millisecond. Each satellite is assigned a unique C/A code, which enables the GPS receivers to identify which satellite is transmitting a particular code. The C/A-code range measurement is relatively less precise compared with that of the P-code. However, it is less complex and is available to all users.

The P-code is a very long sequence of binary digits that repeats itself after 266 days. The code is divided into 38 segments; each is one week long. Of these, 32 segments are assigned to the various GPS satellites. That is, each satellite transmits a unique 1-week segment of the P-code, which is initialized every Saturday/Sunday midnight crossing. The remaining six segments are reserved for other uses. The P-code is designed primarily for military purposes. It was available to all users until 1994. At that time, the P-code was encrypted by adding to it an unknown code [2].

2.1.4 GPS Message

The GPS message is a continuous 50 bits/second data stream added to both the L1 and the L2 carriers. The data is transmitted in logical units called frames. A frame is 1500 bits long, so takes 30 seconds to be transmitted. Every satellite begins to transmit a frame precisely on the minute and half minute, according to its own clock. Each frame is divided into five subframes, each 300 bits long. [3]

Each subframe starts with two specific words: Telemetry (TLM) and HandOver Word (HOW).

- **TLM**: it is a preamble of 8 bits that helps to find the beginning of each subframe. After these bits, there are 16 bits more that are reserved to users and the control segment.
- **HOW**: Following TLM, HOW is formed by 17 bits that contains the time start of the next subframe and the time inside the week.

Each subframe contains a different data depending on its number.

- **Subframe 1** contains second degree polynomial coefficients used to calculate the satellite clock offset.
- **Subframe 2 and 3**: contain the ephemerides of the satellite.
Subframe 4: contains the almanac data which include information about orbit parameters of all satellites, their technical status and actual configuration, identification number and so on. This subframe contains data for the satellites number 25 – 32, ionospheric correction data, special information and UTC time information.

Subframe 5: contains almanac data for the satellites 1 – 24 as well as time and the number of the GPS week.

This project is based on two types of data: ephemerides and almanac. These parameters are on the subframes 2 to 5.

Ephemeris data is a set of parameters that can be used to accurately calculate the location of a GPS satellite at a particular point in time. It describes the path that the satellite is following as it orbits Earth. In this project, this data is also called Navigation data. All these parameters are explained on section 3.1.

Almanac data is a set of data that every GPS satellite transmits, and it includes information about the state (health) of the entire GPS satellite constellation, and coarse data on every satellite’s orbit. In this project, this data is also called Observation data. These parameters are explained on 3.2.

These values are stored on database, and users can read it on the project website.

2.2 Software tools

2.2.1 Language C

Language C is chosen to implement this project. It is a general-purpose computer programming language, it is adaptable to any project and it is independent of the system. Language C was created in 1972, so there is a lot of documentation about it. It has a lot of characteristics: code portability, efficiency, and low runtime demand on system resources. Other reason that language C was chosen is that the previous project [5] is also developed in C, and the access to specific libraries is easy.

2.2.2 MySQL

SQL was created in 1981 by IBM and since then it is considered the standard for the databases. MySQL is a relational database management system that fulfils the SQL standard without loose speed and usability. MySQL run as a server, providing multi-user access to a number of
databases. This characteristic is important, because on the website of this project can access many users.

MySQL is very popular; it is used on web applications, platforms (such Linux or Windows) and error detection programs. When it is used on web applications, the language PHP is associated with MySQL.

MySQL includes other program to do backups, this program is MySQLdump client. This program allows creating a backup of a database or loading a backup to restore the database. This is useful to save a database or load a database of one year ago, for example.

MySQL code uses C, the same language of this project, so it is easy to connect the project programs with the database. MySQL has a library developed in C that allows connecting and consulting databases. Including in the project the library mysql.h, it is possible to connect one program with a specific database. When the connection is established the program can send queries like a user. This library allows to opening a connection between program and database. This connection uses TCP (Transmission Control Protocol), and it does not finish since the program close it. This connection does not use much computer memory, and only is used when new data is sent.

2.2.3 HTML and Apache

HTML (Hypertext Mark-up Language) is chosen to create the project website. This language is easy and is based on mark-ups. With HTML, it is possible to insert images and create different appearances using CSS (Cascading Style Sheets). To create the website, the same CSS of monitoring station website is used.

The web server chosen to maintain the website is Apache HTTP Server. It is a modular server, and it is the server most popular. This software is used by the project [5] and it was not necessary to install. Apache Server keeps the website always available.

2.2.4 PHP

PHP (PHP Hypertext Pre-processor), created on 1994, is an interpreted language used mainly in web applications. Normally it is specially suited to server-side web development, where
PHP runs on a web server but firstly it was designed to create dynamic web pages. PHP code in a requested file is executed by the PHP runtime, and it can be deployed on most web servers, many operating systems and platforms.

PHP is executed on server-side, due to this; web browsers not need support it. This is an important difference between PHP and Java, which requires install it on the user system. A simple web browser can connect to website to consulting the GPS database.

This language is chosen to connect the web page to database. PHP can be inserted on a HTML code easily, only using simple mark-ups. PHP has a library that allows connecting to database, such the C library. PHP code and database are in the same computer, so the connection between them is faster.

### 2.2.5 JavaScript and jQuery

JavaScript is an object-oriented scripting language that is dynamic and weakly typed. It is also considered a functional programming language because it has closures and supports higher-order functions. JavaScript uses syntax influenced by that of C syntax.

JavaScript is primarily used in the form of client-side JavaScript, implemented as part of a web browser in order to provide enhanced user interfaces and dynamic websites. JavaScript can create functions to use by HTML document. Because JavaScript code can run locally in a user's browser (rather than on a remote server), the browser can respond to user actions quickly, making an application more responsive. All current browsers supports JavaScript, so, there are not problems of compatibility.

jQuery is also used on this project. jQuery is a JavaScript library designed to simplify the client-side scripting of HTML. It is free and open source software. jQuery's syntax is designed to make it easier to navigate a document, select different elements, create animations, handle events, and develop other applications. It can be included within a web page by linking to a local copy, or to one of the many copies available from public servers. The website project uses the library stored in a Google server.

On this project, JavaScript and jQuery are used on the website to add more functionality. For example, when user chooses the dates, user can writes manually the date or opens a calendar and marks the day.
2.2.6 RINEX

Receiver Independent Exchange Format (RINEX) is a data format for raw satellite navigation system data. It was created in 1989 to standardizing the received data from different receivers.

This format is defined in ASCII files (American Standard Code for Information Interchange). Although an ASCII file needs more memory than a binary file, it has more flexibility.

There are different versions of RINEX, the last version is 3.00. This version is used to do this project. The standard specifies three different files: observation data, navigation data and meteorological data. In this project, RINEX files are used to read the observation data. Navigation data is obtained directly from shared memory.

The name of a RINEX file has a defined format. This format is “ssssdddf.yyt”, where “ssss” is the station name, “ddd” is a sequence of characters between “a” and “x” if files are generated each hour or 0 if the files are created daily. At last, “yy” is the year and “t” identify if the file is an observation file (with an “O”) or a navigation file (with an “N”).

RINEX files are based in two parts. The first part is a header that contains the station name, information about GPS antenna, the station coordinates, the number of observations, among others. Header finishes with the sequence of characters: “END OF HEADER”. The second part is divided in epochs, which starts always with a time indicator in the GPS scale time and the number of visible satellites. For each satellite, observation file contains the received values of pseudorange, carrier phase, Doppler and SNR. [6]

2.3 Scenario

The scenario is located on the 12th floor of North building of the IST on campus Alameda. On this laboratory, there are installed two GPS receivers that are always receiving the signals of the satellites that are visible.

By default, this project only uses one receiver. The distance between both receivers is only a few meters, therefore the ephemeris and the almanac are equal for two receivers. However, the pseudoranges are different. The database only stores data of one receiver, but modifying a number of the reader program, both pseudoranges can be obtained. This modification is explained on the appendix.

Three different computers are used to do this project. The first computer is named IT GPS monitoring station. The IT GPS monitoring station is a dedicated computer that received the data, but this station does not store it. This station is part of the project [5] of other student of
The operating system installed on the monitoring station is openSuse 11.1, a Linux distribution. To do this project, the monitoring station is a "black box" where the different programs obtain the GPS data. The programs that read the memory are running on the monitoring station. These programs acquire the GPS data and send this data to database. This project only uses the monitoring station to obtain the data.

The database is installed on another computer. This computer uses a Windows 7, but there are not problems between Windows and UNIX because programs connect directly to MySQL Server, using a MySQL connector. MySQL connector provides security, because to start the communication is mandatory to register with the administrator password.

Finally, Apache web server is installed on a third computer to not overload the used computers. This computer is used also by other projects to store other websites. The operating system used is also OpenSuse 11.1. On this computer there are all of HTML and PHP files. Users connect to this server to create database queries. Moreover, this computer shows the results to users.

All computers of this project are on the same LAN. Due to this, all connections between these computers are faster. All these connections use the MySQL connector. The laboratory LAN is formed by a router that connects all computers. This router also protects the computer of any foreign attack. All ports are closed, except the http port. Users can use this port to connect to the website. All other foreign connections are denied by the router.

The next figure shows the final scenario.
2.4 User queries

This section presents the different options that the users can choose to create the queries. The objective of this project is to allow users to create the query that they want. The creation has to be easy and flexible. The most important parameters to create the queries are studied. After this study, some options were chosen. Next these options are explained.

Firstly, the main option is to define a period. The users could want to obtain data of a specific day or period. This period limits the results of the query.

In addition, other time option is added. Maybe, the users would be interested to obtain a period that its data contains a minimum number of satellites. This option is also available, users can select the number and the results will be limited by this sub-period. All the data of this sub-period includes that minimum number of satellites.

Other option that was considered was to obtain the values of one specific satellite. If a user selects a satellite number, the results will only show the data of that satellite.

Probably, the users are interested only on the results if a parameter is bigger than a specific value. With this database application, the users can select a parameter, a value and compare them using a comparison operator. These operators are: “more than”, “less than” and “equal to”. So, a user can select a parameter, a value and an operator. The results will only show the data that fulfills the comparison.

Finally, the users can select which parameters will be shown on the results. If a user is only interested on one parameter, he can select it and the result will only show that parameter. The users can select the parameters that they want.

All these option can be combined to create the desired query. For example a user could create a query to know a specific parameter of one satellite if it fulfills a comparison. Or a user could compare the pseudoranges of the satellites that its phase is lower than a specific value on a particular day.

The creation of queries is based on website forms (explained on chapter 5). These forms are designed to allow users to create the queries with all these options.

2.5 State of the art

On this chapter, the different Global Navigation Satellite Systems are explained. GPS is the only operational GNSS, but there are more systems in development.
**GLONASS**

The Global Navigation Satellite System (GLONASS) is the satellite system developed by the Union Soviet. Now it is operated for the Russian government by the Russian Space Forces. It is an alternative to GPS and it began on 1976 with a goal of global coverage by 1991.

The GLONASS constellation consists of 24 satellites in a three orbital planes, eight satellites in each plane. Currently, GLONASS is slightly less accurate than GPS. With specific equipment, it is possible to receive GPS and GLONASS signal at the same time. [7]

**Galileo**

Galileo is a GNSS currently being built by the European Union and the European Space Agency. It is an alternative to GPS and GLONASS, and it is expected to be operational on 2014. Galileo is intended to provide more precise measurements than GPS or GLONASS. This system consists of 30 satellites, and the receivers will be able to combine the signals from both Galileo and GPS satellites to greatly increase the accuracy.

The political aim is to provide an independent positioning system upon which European nations can rely even in times of war or political disagreement, since Russia or USA could disable use of their national systems by others. Like GPS, use of basic Galileo services will be free and open to everyone. However, the high-accuracy capabilities will be restricted to military use and paying commercial users. [8]

**COMPASS**

The COMPASS system (also known as Beidou-2) is a project by China to develop an independent GNSS. This new system will be a constellation of 35 satellites that will offer complete coverage of the globe. Similar to the others GNSS there will be two levels of positioning service: open and restricted for military purposes. [9]

**Current status**

Nowadays, there are developed a lot of applications that use GPS and it is a technology used in many markets. GPS is implemented on many devices used by the world population. It can be found on mobile phones, cars, military systems among others.

When all the currently planned GNSS are deployed, the users will benefit from the use of a total constellation of 75 satellites, which will significantly improve all the aspects of positioning.
3 GPS Data Management

This chapter presents the different GPS data used on this project. All of this data is stored on the database, and it is accessible using the website. There are three types of data: the navigation data, the observation data and the elevation and azimuth. On this chapter the data is explained in detail. The methodology used to obtain these parameters is also explained.

3.1 Navigation Data

This data consists on the ephemeris values. These parameters describe the satellite orbit. Each satellite broadcasts only its own ephemeris data. These parameters are explained on this section. [4]

\textbf{SV\_Nb}: indicates the satellite number. This value can be a number between 2 to 32, without 25.

\textbf{Week Number}: this value indicates the number of week, starting on 1980.

\textbf{URA}: The User Range Accuracy (URA) is a statistical indicator of the ranging accuracies obtainable with a specific satellite. It includes all errors for which the Space and Control Segments are responsible. It does not include any errors introduced in the user set or the transmission media.

\textbf{Health Satellite}: is a number that indicates if data is correct or not. If this value is equal to 0, all navigation data is OK. If this number is different to 0, some or all navigation data are bad.

\textbf{Issue of Data Clock (IODC)}: indicates the issue number of the data set and thereby provides the user with a convenient means of detecting any change in the correction parameters.

\textbf{Issue of Data Ephemeris (IODE)}: this value is useful to detecting any change in the ephemeris representation parameters. This value changes when ephemeris changes.

\textbf{Timing Group Delay (TGD)}: is the bias difference between each GPS satellites P-code transmissions at the L1 and L2 frequencies.

\textbf{Time of Ephemeris (toe)}: Ephemeris data reference time of week in seconds.

\textbf{Time of Clock (toc)}: Clock data reference time of week in seconds.
**Curve Fit Interval:** is a flag which indicates the interval used by the control segment in determining the ephemeris parameters. If this value is 0, the interval is equal to 4 hours. If it is equal to 1, the interval is greater than 4 hours.

\( M_0 \): is the mean anomaly at ephemeris reference time (toe).

\( e \): Eccentricity of the elliptic orbit.

\( A (\sqrt{A}) \): Square root of semi-major axis.

\( \Delta n \): is the correction of the satellite’s angular velocity in its orbital plane.

\( \Omega_0 \): Longitude of ascending node of orbit plane at weekly epoch.

\( i_0 \): Inclination angle respect to equator at ephemeris reference time.

\( \omega \): argument of perigee.

\( \Omega \): Rate of Right Ascension.

\( IDOT \): rate of inclination angle.

\( af0, af1, af2 \): coefficients used to correct the satellite’s clock.

\( C_{rc}, C_{rs} \): Amplitude of the cosine and sine harmonic correction term to the orbit radius.

\( C_{uc}, C_{us} \): Amplitude of the cosine and sine harmonic correction term to the argument of latitude.

\( C_{ic}, C_{is} \): Amplitude of the cosine and sine harmonic correction term to the angle of inclination.

### 3.2 Observation Data

The observation data is composed by four different values: pseudorange, carrier phase, doppler and SNR. These parameters are read from the RINEX files created by the monitoring station.

**Pseudorange:** (Explained on 2.1.3.4) is the distance from the receiver antenna to the satellite antenna including receiver and satellite clock offsets (and other biases, such as atmospheric delays). This value is measured in meters.
**Phase:** The phase is the carrier-phase measured in whole cycles.

**Doppler:** The sign of the Doppler shift as additional observable is defined as usual: Positive for approaching satellites.

**SNR:** Signal to noise ratio.

### 3.3 Elevation and Azimuth

In addition to navigation and observation data, elevation and azimuth are also stored on the database. These parameters are calculated by the monitoring station using the received values. Elevation and azimuth allow to situate the visible satellites. These parameters are measured in degrees.

Azimuth is defined as a horizontal angle measured clockwise from the North as shown the Figure 3.1. The values range from 0 to 359.

![Figure 3.1: Azimuth example](image)

Elevation is the angle between the horizon and the line of sight to a satellite. Elevation values oscillate between 0 and 90. The value is equal to 0 when satellite is on the horizon and 90 when it is above the antenna.

On Figure 3.2, there is an example that shows the visible satellites with its elevation and azimuth. This figure is extracted from the monitoring station. Elevation is the radius between the centre and the satellite position, and azimuth is the angle formed between North and satellite position.
3.4 Data Acquisition

On this subchapter data acquisition is explained. This acquisition is not from the message sent by GPS satellite. As it was explained on chapter 2.3, the IT GPS monitoring station is the responsible to obtain GPS data from the received message. The monitoring station saves data on its memory. This memory is shared and other programs can read it. For this project, monitoring station is a “black box”. Programs only read the shared memory to obtain the GPS data.

Navigation data, elevation and azimuth are obtained directly from a partition of the shared memory. However, observation data is not saved on the same partition. Due to this, IT GPS monitoring station does not provide the necessary libraries to read these values. To solve this problem, it is necessary create new libraries and structures to store all these parameters. This solution involves modifying the monitoring station, created on the project [5].

Finally, other solution was chosen, without changing the monitoring station. The monitoring station does not include the required libraries to acquire directly the observation parameters. However, it uses these values to create the RINEX files, one each hour. A RINEX reader program was created to obtain the observation data. This program reads all values from RINEX files. All values are from the last hour, because only when RINEX file is created, other programs can read it. Then, the reader sends all values to database.
Summarizing, there are two ways to obtain the different parameters. The first one reads directly from the shared memory, used to obtain the navigation data, elevation and azimuth. The second one reads from the RINEX files to obtain the observation data. Both ways use reader programs that are implemented in language C. Two reader programs read from shared memory and the other program reads from the RINEX files.

All reader programs are executed on monitoring station to read all GPS data. These programs are executed only when is necessary to not overload the monitoring station. To do this task, Cron from UNIX is used. Cron is a time-based job scheduler in Unix-like computer operating systems. It is included in all UNIX distributions, so, it is also included on openSuse used by the monitoring station. It enables users to schedule jobs (commands or scripts) to run periodically at certain times or dates. [10]

The readers are executed by Cron. Each program is executed with a different period, but this period can be changed, modifying the Cron configuration file. By default, the periods are:

- Elevation and Azimuth: these both parameters are read every 10 seconds. With this short time, visible satellites can be monitored constantly. This information is useful also to create complex queries.

- Navigation data: this period is equal to 15 min. This data contains more values than elevation and azimuth and it does not change quickly, so, period can be longer.

- Observation data: this data is not read from shared memory, and it is read from RINEX file. This file is created every hour, and the period to send it to database is also 1 hour. Every hour is sending 4 epochs of the last hour. Only it is possible to read the RINEX file when the program that is writing on it finishes. So data that is sent to database has a delay of 1 hour.

Using Cron, programs do not need run all the time. They only are executed when the clock of the system calls them. Due to this, if computer shut-downs or crashes, in the next boot are not necessary run any program to start the acquisition data. It is an autonomous system.

3.4.1 Data reading

Navigation, observation data and elevation and azimuth are stored on the database. All these parameters are obtained from the IT GPS monitoring station. The methodology to read all data is similar. Firstly, the data is read by reader programs. A reader was implemented for each type of data. These readers can read the values from the shared memory or the RINEX files. The readers obtain the values and store them on structures. There is one structure for each type of
data. These structures organize the parameters before send them to the database. A structure contains one field for each parameter. For example, the navigation structure contains one field for each ephemeris. Other field is created to store the date. When the read data is sent, other new data overwrites the structure. These structures organize the data and it is easily to send the parameters to the database.

There are two different ways to read data. One way is read the data directly from shared memory, used to read ephemerides, elevation and azimuth. The other way to read the data is from a RINEX file, used to read the observation data. Both ways to read data use structures to store the values.

### 3.4.1.1 Reading from shared memory

Ephemerides and elevation and azimuth are stored in the shared memory of the monitoring station. This data refreshes every time that a receiver receives new parameters. The new data overwrites the last received data.

![Diagram](image)

**Figure 3.3: Data acquisition from shared memory**

The shared memory can be read by programs to obtain the received GPS data. The monitoring station includes semaphores to access to this memory. These semaphores control the access to prevent possible conflicts between reading and writing. The semaphores are implemented on monitoring station, and they were created on the project [5]. The reader programs do not perceive these semaphores, they only read directly from the shared memory.
Thanks to monitoring station obtain the data is easy; only it is necessary include the library that contains the memory address of each parameter. All libraries are in C, the same language used on this project. There is one library for each type of data, and the readers include them on the code.

Firstly, a reader stores the read values into a structure. When all values of one type of data are read, the program opens the MySQL connector and sends these parameters to database. This connector opens a communication directly to database. Using the connector and the “insert query” command (explained on 4.2), readers can insert parameters to database easily. All values are processed by the database that stores them on the specific table. When all data is sent, program closes the connector.

3.4.1.2 Reading from RINEX file

On the other hand, observation data cannot be read directly from the shared memory of the monitoring station. This data is stored on a different partition of memory and monitoring station does not include the required library to read it. However, the observation data is used to create a RINEX file each hour. So, observation data is acquired reading these files. The RINEX files include the values received the last hour. A RINEX reader was created to read these files. This reader also stores the values on a structure and sends it to the database.

The RINEX files are created automatically every hour by the monitoring station. Each RINEX file is stored on a different subfolder, depending on the day. All data of a day is on a folder that contains all the data of this day. Furthermore, all data of each hour is stored on a different subfolder. So, the first job for the reader is to convert the current date into a path to find the correct RINEX file.

When the reader finds the RINEX file, this file is opened. Firstly, the header of the file is read. The header includes the order of the observation parameters, and it is read to save the values correctly on the structure. The header also includes more parameters, like RINEX version and other information about RINEX, which are not used on this project.

Following the header, all data is included. There are all parameters received in one hour, with a period of 10 seconds. The reader always opens a RINEX file that contains the values from the last hour. These values do not change much in one hour, they are very similar. To not store duplicated data, it is only sent to database four epochs (every 15 min; at o’clock, 15’, 30’ and 45’). The reader program “jumps” to these specific epochs. By default, this period is 15 minutes, but it can be changed easily, modifying the Cron configuration file. Each read epoch is stored on the specific structure, and it is sent to database.
Parameters are sent in the same way than parameters read from shared memory. The reader program creates and fills the structure with the read data. Then, the reader opens a connection using MySQL connector, and it sends the values to database. When the reader reads and sends the entire file, closes MySQL connector and the RINEX file.
4 Database

This project is based on a database to store different parameters. The readers send constantly all GPS data to the database. So, the main requirement of the database is to be stable and functional 24 hours a day. Other important requirement is a fast response; users do not like waiting for the results.

MySQL was chosen to create the database. MySQL fulfil these requirements and it is the most popular system to create databases. It uses SQL language that is very powerful and it has high flexibility. This language allows creating complex queries to obtain the desired results.

Additionally, MySQL includes two useful tools: MySQL connector and mysqldump. The first allows opening connections between a program and database. The mysqldump is useful to create and load backups. This tool can create backup file of a remote database using MySQL connector. The database creates backup files to protect the received data using this tool. These files can be used to restore the database if a problem occurs. In addition, mysqldump and MySQL connector can be combined to create backups on remote computers. This feature is used to download the data by the users.

4.1 Database structure

Database is deployed on another computer, to not overload the monitoring station. This computer is dedicated only to store data. Also, it processes the user queries and sends the results to web server. The MySQL server is installed, including the mysql dump to create backups.

A specific port is opened to enable the connection from other computers. This port is open on Windows firewall. The opening of this port is not a security problem, because the laboratory router has rules to protect all foreign connections. The connection between database, monitoring station and web server are on the same network. Router accepts these connections and refuses connections of other networks. A foreign computer cannot connect to database.

Once installed, the database was created. This database is called “GPS” and all data is stored on it. This database includes all the tables of this project. Creating all the tables on the same database, the queries can consult faster all the tables of database.

Once the database is implemented, the different tables are created to organize the received values. There are three types of data to store: navigation data, observation data and elevation and azimuth. For each type of data, a table was created. These tables include the different fields for each data. All tables include the date and the satellite number.
The next figure shows the command that was used to create the “elevation and azimuth” table.

```
CREATE TABLE `ele_azi` (  
`date` datetime,  
`SV_Nb` int(2),  
`Ele` int(2),  
`Azi` int(3)  
) ENGINE=InnoDB
```

Figure 4.1: Table creation

The table created on Figure 4.1 is named “ele_azi”. The first line defines the name of the table. Following this statement, the different fields and its type are included between brackets.

The first field of this example is the date. This field is included on all tables of this project, and all values are sorted by it. The type of this parameter is “datetime”, a specific type of MySQL to define dates. The next field is “SV_Nb” and its type is an integer. This parameter is the satellite number. The next parameters are elevation and azimuth and its types that are also integers.

The number between brackets after each type is the number of characters that can be stored. For example, on azimuth, this number is equal to 3 because azimuth values range from 0 to 359, and it does not need more memory. On the other hand, elevation values range from 0 to 90, so it only need two numbers.

Finally, SQL engine is chosen, all tables of this project has the same engine, InnoDB. It is a storage engine for MySQL. The other engine that it is possible to use is MyISAM, but finally InnoDB was chosen. InnoDB is faster and it recovers from a crash by replaying its logs.

For each type of data was created one table with all parameters. The GPS database is based on three tables. Elevation and azimuth table is created as is shown on Figure 4.1. Navigation table includes one field for each ephemeris, and observation table includes all parameters that are read from RINEX file. Figure 4.2 shows database structure.
4.2 Insertion of values

Readers obtain the data of the shared memory or RINEX files. This data has to send to the database. When a program read data, it stores the values on structures before sends them to the database. When all values are stored, the reader opens a connection to the database server. This connection is used to send the data to the database. Reader programs of the monitoring station have to include the library mysql.h to use this connector.

Firstly, a message is sent to initialize the connection. This first message includes the username, the password and the name of the database. If these parameters are correct, the connection between monitoring station and database will be created successful. If one parameter is incorrect, MySQL server will return an error message, and the connection will be closed.

When the connection is initialized, programs can execute any command as it was using the MySQL command line client. All MySQL statements are accepted. The readers only use the command to insert the read values. The main command to insert data to a database is “insert into”. There are other commands to insert values but they are not used. Following lines shows an example of this command.

```sql
insert into sats (Date, SV_Nb, pseudo, phase, doppler, SNR) values (date_buf, data.SV_Nb, data.pseudo, data.phase, data.doppler, data.SNR);
```

Figure 4.3: insert command
Firstly, the command is defined with the statement “insert into”. After this statement the name of the database is included. On this example, database is named “sats”. Following, between brackets, the names of data is defined. These names are the names of each parameter. This example shows the insertion of the observation data, and the types are: date, satellite number, pseudorange, phase, Doppler and SNR. After the brackets, the word “values” indicates the different values that will be sent. All the values are between brackets, and they must be on the order defined before. On this example, these values are stored in a structure named data, and all values are read from it.

The date value is different due to GPS time. This time is different to UTC, as it was explained on chapter 2.1.3.2. The values obtained by the reader are two: the week number since 1980 and the number of seconds inside the current week. Before sending, these values are converted into a specific format. This format is “YYYY-MM-DD hh:mm:ss” and it is used by MySQL.

All MySQL commands are sent using the MySQL connector, and if an error occurs, a message mysql_error is returned which contains the error type. Other possible error is when database is off, and a user sends a message. On this case, the error message is notified to user. All errors are stored on the MySQL log files.

4.3 Backups

The database stores all information about GPS satellites. This information is the core of this project. If some value is lost, the queries of users will not find the correct result. A backup system was created to protect all received data.

A backup file or a dump is a file created with all the content of database. This file contains SQL statements to create the table, populate it, or both. The file is readable by any SQL server. On this project, all backup files include statements to create and populate the tables. All tables of this project (navigation, observation and elevation and azimuth) are chosen to create automatically the backup file.

The backup file is created using mysqldump. This program allows to create and restore backup files. Mysqldump has many options to create dumps. Some options are selected to protect all values of the database.

The main option is not delete the previous database. This option does not let delete any database or table of the MySQL server. However, this option allows to create a table if it does not exist on the database. Other option included on the backup files is not to overwrite a value if it is on database. A backup file cannot modify the previous values. With these options, the stored data cannot be deleted restoring the database with a dump.
Mysqldump is also used to restore the database using a file. The backup files contain all SQL statements to restore it. This program only can be executed on the database computer.

However, on this project, restoration for users is available only on website, but, for protection reasons, they cannot modify any value. The users only can add new values. If a user needs to change any parameter, he must connect directly with MySQL server, using the administrator password.

An autonomous system was developed to save all the data. The main objective of this system is the creation of one backup each week. Weekly on Sunday, a backup of all tables is created. This file is stored on the database computer.

The database computer uses Windows as operating system, so Windows’ tools are used. Firstly a script was created using batch language. This script contains commands to create a backup file. These commands are interpreted by the command interpreter. The batch files are executed by shell programs, in this case, cmd.exe. Figure 4.4 shows the script.

```plaintext
set DATE= %date%
set DATE =% DATE:/=%
set DATE =% DATE: =%
set DATE =% DATE::=%
set DATE =% DATE:,=%
set NAME=C:/database/backups/backup_%DATE%.sql
mysqldump -p"password" -u root gps > %NAME%
```

Figure 4.4: Backup script

The first five lines remove all separators of the date to create the name. The next line creates the final name of the backup file and its path. The path is “C:/database/backups”. All files created by this system are stored on this folder. The format of the name is “backup_XXXddMMYYYY.sql”. The first three characters are the three first letters of the day (i.e. Mon on Monday). Following these letters, the numerical date is included without separators and, finally, the extension of SQL files.

The last line of the Figure 4.4 is the mysqldump command. This command contains the root password and it executes mysqldump with root privileges. Finally, database gps is chosen to create the backup with the selected name. This selection creates a backup file of all tables.
Every time that this script is executed, a backup file will be created. A scheduler executes the script. On UNIX systems, Cron can be used to schedule tasks, but the database computer uses Windows. This operating system also has a scheduler named Microsoft Windows Task Scheduler. It is easy to use; only it is necessary select the script and when it has to be executed. Default options are chosen and the script is executed every Sunday. If there is a problem, scheduler will try to execute it every 5 minutes since 5 times. After 5 times, the scheduler does not execute more the script that week.

With this backup system, all data has a copy and it is more difficult to lose it. To restore a dump file, only it is necessary use the mysqldump with root privileges or the MySQL command line on the same computer. Other way to restore a database with a file, it is using the website and upload the backup file. This way is explained on the chapter 0. However, the stored values cannot be modified using the website.
5 Website

All values of GPS satellites are stored on the database. These values have to be available to users; this is the aim of this project. To do this, a website is created. Using the website, users can create queries, view the results and download it. Users can create different types of queries and they can select the values visible on the results.

Users cannot connect directly to database using the MySQL port. The laboratory router denies all foreign connections, but it accepts http (Hypertext Transfer Protocol) connections. Thus, the only way to connect to database is using the website.

Website can connect to database because it does not need across the laboratory router, as is shown on Figure 5.1. So, users can connect to website to create their queries and send it to database.

![Laboratory network diagram](image)

Figure 5.1: Laboratory network

Furthermore, website is more “user friendly” than MySQL command line client. Website allows to create queries and to show the results more visual than a command line, like MySQL client. Using the website, users can create simple and complex queries. These queries are sent by the website to database. Finally, database executes the queries and it sends the results to the website.
5.1 Architecture

Firstly, an http server was chosen. Apache server is used to maintain the website. This server is very stable and guarantees that website is functional all the time. The configuration of this server is on an xml (eXtensible Mark-up Language) files. To realize this project, the default configuration is used, only the path of the new pages is added.

Website uses html and php, two different languages to do different functionalities. Also MySQL server is used. To not mix all, the Model-View-Controller (MVC) is used.

Model-View-Controller is a software architecture that divides the view, the control logic, and the data. This architecture creates three differential parts, due to this, each part can be independent developed or tested. With this pattern, the different “behaviours” of the web application are isolated, creating modularity. There are three different modules: the model, the view and the controller. [11]

Model

The model manages all the data of the application. This module offers the required data when the controller needs it. This module does not have knowing about the controller, it only stores and give the data. In this project, the model is the database because it stores all the information. Queries are sent by the controller to obtain the results.

View

The view manages the display of information. This is the module that users view on their browsers and they can interact with it. On website, the view is the html page including the CSS.

Controller

The controller processes the user inputs, consults the model and informs to the view to show the results. The controller creates the relation between the view (the users) and the model (the data). In this project, the controller is the php pages. These pages create the user queries to send it to the model (database), and return the results to the view (the html page).

Figure 5.2 explains how this architecture works. Firstly, on the view module, the user chooses the options to search on the database. This module only collects the selected options, like date, satellite number or the parameters that the user wants to view on the results. When the user selects all the options, an http request is sent to the controller. Then, the controller receives all options and a MySQL query is created with all this parameters. This query is sent by the
controller to the model. The model (MySQL server, on another computer) executes the query and returns the results to the controller. The controller modifies the view showing the results on a table.

![Diagram of Model-View-Controller](image)

Figure 5.2: Model-View-Controller

The main advantages of this model are the separation between the user's view and the data. With this pattern, it is possible to add different views, only it is necessary to connect the new view with the controller. Furthermore, with this modularity, it is easy to repair a problem that occurred in a module, without changing other modules. On the other hand, with MVC, it is necessary to create more files than a monolithic architecture.

All web pages created on this project use this pattern. There are four web pages, one for each table (elevation and azimuth, navigation and observation) and one more to download or upload the data. For each one, a controller is implemented to connect it to the corresponding table.

The website is located inside the laboratory domain. This domain is [www.satlab.it.pt](http://www.satlab.it.pt). On the main page of this domain there are three different links: two are created for other projects, and the third link is where the GPS Database website is.
Figure 5.3 shows the organization of the GPS Database website. The website is structured on four parts. Each part is one table on database. All these parts have the same Model-View-Controller pattern explained on the previous chapter.

On the next chapters these parts are explained. All of these parts consist on an http page and a php file that connects it with database. The download/upload part is more different than the others because different files and programs are involved.

All pages have the same structure: it contains a form with all options to do the queries. These pages have two buttons at bottom, users can choose how the results are shown: on the browser or in a file and download it. These buttons send the user inputs to the php page.

### 5.2 Basic form

On this subchapter the basic form of the website is explained. This form is used by the pages that connect with the different tables of the database. These pages are “Elevation and Azimuth”, “Navigation” and “Observation”. This form includes the basic tools to create the queries. All these pages use the same form structure, but each one has some changes.

Firstly, users can select a time period, specifying the first and the last date using two selection boxes. The subset of results is related to this period. These both dates are mandatory and it must fulfil the format “YYYY-MM-DD”. If any of these fields is empty or not fulfils the specific format, when a user presses the “Search” or the “Download” button, a message of error will be returned. A calendar is implemented to make easier this selection. This calendar is developed
using JavaScript and it is supported by all current browsers. These selectors are shown on Figure 5.4. If this period is long, the database will process more data, so the processing time will be longer.

![Date selectors](image)

**Figure 5.4: Date selectors**

Following the date selector, all advanced options are presented. The first option is the selection of one satellite. Filling this box with a number different to zero, the result only will show the data of this satellite. The satellite number is between 2 to 32, without the number 25. If this box fills with a zero, the query will show all the satellites.

The second option that users can choose is the minimum number of the satellites. This option is only activated if the checkbox is marked and a number is written on the box. With this option, a sub-period will be created inside the selected period. This sub-period is formed by the longest period that more satellites are visible. All data of this period has, at least, a number of satellites equal to the selected number. Figure 5.5 shows an example.

![Select the minimum number of satellites](image)

**Figure 5.5: Example of minimum number of satellites**
On this example a different values are shown, the table has two columns: one is the date and other indicates the number of visible satellites on this date. A user selects 10 on the box of minimum number of satellites. The created algorithm searches the longest period that has 10 or more visible satellites. This algorithm is based on a vector that uses indexes and counters to find this period. On this example the first date of the sub-period is 15:40:00 and the last date is 15:41:00, all values between both dates are shown on results page. This is useful when a researcher needs values of a period with a minimum number of visible satellites.

Below this option, each form has other options that are explained on the next subchapters. After these different options, all forms include the type of data that will be shown on the results. To do this, a group of checkboxes is created, one checkbox for each option. Figure 5.6 shows this option.

![Observation checkboxes](image)

Figure 5.6 Observation checkboxes

Moreover, this option includes two checkboxes more to select or unselect automatically all options. The first checkbox named “All data” also selects all option. To do this automatically, JavaScript functions are used.

After this, all forms include two buttons at bottom. These buttons allow choosing the results format. One shows the results obtained in a web and the other in a file to download. These both options are explained on chapter 5.6.

### 5.3 Elevation and Azimuth page

On this page, users can create queries and send it to “Elevation and Azimuth” table on the database. The page is the basic form where users can fill different fields to create the desired query. Other options and information are included into the basic form. The html page is shown on Figure 5.7.
Firstly, this form informs about the position of the antenna. This is the position of the laboratory on the 12th floor of North building. These coordinates are in WGS84, the standard used by GPS, and it includes the latitude, the longitude and the altitude. These parameters are necessary to interpret the values of elevation and azimuth. The coordinates places the antenna in a point of the Earth. Elevation and azimuth situate the GPS satellites respect this point. Without these coordinates, it is impossible to situate the satellites.

This page contains other advanced options. These options allows to users choosing a minimum and maximum value of elevation and azimuth. All these parameters are in degrees, elevation range from 0 to 90 and azimuth range from 0 to 359. If zero is filled in a box, the minimum or maximum value will be chosen to create the query.
5.4 Navigation page

This page allows users to create queries and send it to navigation table on the database. The page is a basic form with other advanced options. This page is shown on Figure 5.8.

![Navigation Data](image)

Figure 5.8: Navigation page

The added option of this page is the possibility to choose a parameter and compare it with a value, using comparison operators. These operators are: “more than”, “less than” and “equal to”. The parameter is selected using a list that contains all navigation parameters. The query will include all these comparisons. This option is shown on Figure 5.9.

![Navigation option](image)

Figure 5.9: Navigation option

Moreover, users can add different comparisons. Users can press the right button named “Add more...” to add a new comparison. Up to 10 comparisons can be added using this option. Each time that the button is pressed, a new row will be added. To do this, jQuery is used. This library changes dynamically html page without any load.
5.5 Observation page

On this page users can create queries to send to observation table. The page is shown on Figure 5.10.

[Image: Observation page]

Figure 5.10: Observation page

This page uses also a basic form, and one advanced option is added. This option is very similar to the navigation comparison option. However, observation has less parameter than navigation (only 4), so all parameters can be shown without any button. This option includes a checkbox to activate the option, the parameter, the comparison operator and a box that users can write the value to compare. Figure 5.11 shows his option.

[Image: Observation option]

Figure 5.11: Observation option
5.6 Results page

If a user creates a correct query, this query will be sent to database by the website. All these data is sent using the channel created by MySQL connector. Before the query is sent, users can select the format of the response. Users can choose between two buttons that are shown on Figure 5.12.

![Results buttons](image)

Figure 5.12: Results buttons

The first button allows showing the results on the website. A new html page is created dynamically by the web server. An example of this page is shown on Figure 5.13.

![Results page](image)

Figure 5.13: Results page

The response is received by the PHP file. This file creates the results page, using MySQL library of PHP. This library is useful to create visual tables. Using this library is possible to distinguish the different fields and values received. With this information the table is created. The first row of the table defines the different parameters. Then, each value is put on the correct cell using loops. When table is finished, the final result will be shown to users.

The second button that users can choose to send the query is “Download in a file”. With this button, it is not necessary create an html page to show the results. Instead of this, a temporary file is created on the web server. This file includes the same table of the Figure 5.13, but without lines. Table lines are changed by tabs and, as the previous case, the first row defines the parameters. Tabs are used because users can open the files with a spreadsheet application (i.e. Microsoft Excel) that provides to users more versatility. User can define tabs as separators, and the application will open correctly the file. When all values are on the temporary file, file is
completed and it is sent to user using the http connection. Then, the temporary file is deleted, to not overload the computer.

5.7 Download/Upload page

Other option is the possibility to download the data. This data can be used to restore or copy the database. This database can be used in any SQL servers. Moreover, this page allows user to load other backup files, if these files does not override the data. All these options use also MySQL connector but without the administrator password. Without this password, users cannot delete any existing data. Both options are on the same html page.

5.7.1 Download

Users can download the values of the database; they can select between the different tables or download all data. The html part is very simple; it only includes one selection list to select the data and a button to download it. This option is shown on Figure 5.14.

![Select table](image)

Figure 5.14: Download option

This page, as all of pages, includes a PHP file; other pages create a query to be sent to MySQL server. But on this case, the PHP file does not create a query. This file executes the mysqldump with some options to create a backup file with the required data. These options include that this backup cannot overwrite any value.

This option is very similar to “Download the results in a file”. Firstly, PHP file uses mysqldump to create a temporary file on the web server. This file is a remote backup file. Mysqldump of the web server connects to MySQL server, and it receives all the data. When this backup is finished, web server will send the file to the user using http connection. Finally, when user receives the entire file, the temporary file will be deleted.
5.7.2 Upload

Users can load to database other SQL files. These files cannot overwrite or delete any existing value. Users can upload backup files or new file of other database, if this database is equal to the project database. The main objective of this option is the upload of backups that are not on the database. An example can be the upload of ancient values.

The files must fulfil the SQL statements. All files based on SQL language are accepted. Furthermore, the load statements cannot include the creation of new tables, and it must be adapted to the tables of the database. If one of these rules is not fulfilled, the MySQL server will not load the file.

The html page includes one file selector. Users select the file to upload using this selector. Firstly, user selects a file of his hard disk and presses the “Upload” button. Download and upload shares the same html page, but not the php file. The “Upload” button executes other php file. This file uses the http connection to upload the selected file. The server stores temporarily a copy of this file on the /temp/ folder. This folder is used by some programs to store temporary files.

When the file is uploaded on the web server, php file executes mysqldump. Mysqldump also includes an option to load a file on database. Web server sends the uploaded file to MySQL server that processes it. On this case, the administrator password is not included. Thus, uploaded files cannot override any value. Without this password all existing data is protected. When mysqldump finishes, if all is correct, web server will delete the temporary file.
6 Conclusions

This project is based on the IT GPS monitoring station. This station is continuously receiving data from the GPS system. However, this data is stored for a few seconds and it is overwritten by the new data. This project solves this problem with the creation of a database. Furthermore, users can access to the database using the website.

The project was divided on four parts. Firstly, the Global Positioning system was studied to understand how it works and the different parameters that it uses. With this knowledge, the database was designed to store all the received data.

The second part of this project is the data acquisition, based on reader programs. The basic idea of these programs is to obtain all the GPS data from the shared memory of the monitoring station. However, the observation data cannot be obtained in this way because the required libraries are missing. The solution adopted is the reading of the RINEX files that contains the observation data. This solution does not change the monitoring station but it introduces a 1 hour delay on this data. This delay is not very important, because the system is running 24 hours a day. So, there are a lot of hours stored on the database.

The third part of this project is the development of the database. This stage includes also the implementation of a backup system to protect the received data. Each week, the database creates a backup file with all the data. This file can be used to restore the database if it fails, but also it can be used to copy the database on another computer.

The last part is the access to this data. All GPS data is accessible for the users through a website. This website is friendly and users can create easily queries using the forms. The architecture used to create the website, the Model-View-Controller, provides modularity. With this feature, new web pages can be added easily or if one page crashes, the website will not crash. In addition, Users can create and download a backup file with all or a part of the received data.

During this project, the following topics have been successfully achieved:

- **Database**: the database can store all data received by the IT GPS monitoring station. Before the creation of this database, users cannot access easily to the data. Database stores and organizes all GPS data.

- **Data Acquisition**: the reader programs acquire the data from shared memory. These programs read and store on structures the data. Then, the readers send the data to the database, but they can be used to send the data to another computer if it is necessary.

- **RINEX reader**: the RINEX reader created to read the observation data could be used to read other RINEX files.
• **Website:** the created website is user-friendly. Users can create queries, download the results or upload new GPS data.

• **Backup system:** this feature provides data protection. It is autonomous and all backup files are created automatically.

• **Fast response and security:** users want a fast response of his queries, and the website achieves it. All equipment used on this project is located on the same LAN. So, the connection between computers is very fast. Also, the Laboratory router has rules to protect all these connections and computers from external intruders.

• **Autonomous and stable:** Automatically, all readers are executed and they send the data to database. The database also creates the backup file using a script; hence all the created system is autonomous. In addition, it is also stable, because the created programs cannot crashes all the system if they fails.

• **24 hours a day:** the system can run 24 hours a day. The programs are executed only when it is necessary, so they do not overload the computer memory. The web server and database server are chosen to run all the time without problems.

6.1 **Future works**

The aim of this project is to store the received GPS data. Now, users can access to this data easily, without programming knowledge. The website provides all data received by the IT GPS monitoring station. Users can view the results online or download it. With all this results, many applications can be developed, such as:

• Develop navigation algorithms using real data.

• Plot graphically the GPS satellite constellation, selecting a specific date.

• Study the received values and the possible errors.

• Implement this system on others stations to increase the database and compare values.
References


APPENDIX A. User’s guide

This user’s guide describes those technicalities that are required to know in order to use the system. This guide is divided on three parts: data acquisition, database and website. This system is now installed on three different computers, one for each part. However, all these parts can be installed on the same computer, but it is possible to overload it.

Data acquisition

The reader programs acquire the data received by the IT GPS monitoring station. All these programs are installed on the monitoring station. The IP address of the database is defined in the first lines of each file.

The ephemerides reader is located on the folder /home/it/Desktop/it/Eph_download, where the required libraries are. This folder contains all required files to read the ephemerides. The reader program is named read_eph. The reader and the libraries must be located on the same folder.

The elevation and azimuth reader is also located where the required libraries are. This folder is /home/it/Desktop/it/GPGSV_download. The reader is named read_num and it is mandatory to install on this folder.

Finally, the RINEX reader is located on /home/it/Desktop/it/Eph_download, and it is named read_rinex. This file does not have any specific folder. At the beginning, the reader finds the folder to read the data.

All readers read the data of one receiver, the receiver number 0. The distance between two receivers is only a few meters, so the ephemeris and almanac of each receiver are very similar. However, the pseudoranges are different, and database only stores the value of the receiver 0 by default. The reader can be modified to read both receivers. The line 53 contains the folder where the RINEX file is. This folder contains only the data of the receiver 0. Modifying this line, the user can select the receiver.

Receiver 0:

"/usr/local/apache2/htdocs/Rinex/Receiver0/%Y/%B/%d"

Receiver 1:

"/usr/local/apache2/htdocs/Rinex/Receiver1/%Y/%B/%d"

All these programs are executed automatically by Cron, the job scheduler. Cron is driven by a crontab file, a configuration file that specifies shell commands to run periodically on a given
schedule. This configuration is located on `/etc/crontab`. In this file the period of the data acquisition is defined and it can be changed by the root.

**Database**

Database server is developed in another computer. On this computer the default installation of MySQL server is chosen. This software is free and it can be downloaded on the project website, [www.mysql.com](http://www.mysql.com). One stage of installation is the selection of the port number. MySQL uses port 3306 by default. This is important because all connections to database use this port. Other option is to enable external connections, if this option is not activated, any program can connect to database.

MySQL server is always running, and it is executed every time that Windows starts. The main folder of the database is located on `C:\Database`. This folder contains scripts to restore the database, but it removes all the stored data. This script is used to start the database for first time. This folder also includes a subfolder where the created backups are.

Commands:

To connect to database using the MySQL command line:

- `Mysql --u root --p[password]`

To create a backup file using the MySQL command line:

- `Mysqldump -opt --u root --p[password] [dbname] > [backupfile.sql]`

To restore the database using a backupfile:

- `Mysql --u root --p[password] [db_to_restore] < [backupfile.sql]`

**Website**

Firstly, the Apache web server has to be installed. This software can be downloaded from the Apache website, [www.apache.org](http://www.apache.org). This server maintains the project website on [www.satlab.it.pt](http://www.satlab.it.pt). The following commands are used to start and stop the server:

- **Start**: `/usr/local/apache2/bin/apachectl -k start`
- **Stop**: `/usr/local/apache2/bin/apachectl -k stop`

All website files are inside `/usr/local/apache2/`. This is the root path of Apache. On this folder, there are all configuration and log files. Inside the subfolder `htdocs`, the http and php files are
located. The errors occurred on the website are written automatically by the server on a file named error_log.

PHP has a configuration file to configure some parameters. This file is located on /usr/local/lib/, and it is named php.ini. To connect to MySQL server, a parameter is activated, this parameter loads the module of php to create the connection. All other options are the same to default configuration.

The website is composed by the following files. Each page has an html file and a php file. The html file contains the form, and the php file sends the queries to database.

**Elevation and azimuth page:**
- num_form.html
- num_cons.php

**Navigation page:**
- nav_form.html
- nav_cons.php

**Observation page:**
- obs_form.html
- obs_cons.html

**Download/Upload page:**
- down_form.html
- down_cons.php