



VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Duy Nguyen

TOUCH SCREEN OPERATED DATA WAREHOUSE APPLICATION

Technology and Communication
2010

TIIVISTELMÄ

Tekijä	Duy Nguyen
Opinnäytetyön nimi	Kosketusnäyttö toimintainen tietovarasto sovellutus
Vuosi	2010
Kieli	Englanti
Sivumäärä	71 + 20 liitettä
Ohjaaja	Smail Menani

Opinnäytetyö tarkastelee reaaliaikaista seurantajärjestelmää. Vaikka on ollut paljon nykyisiä reaaliaikaisen seurannan järjestelmiä, ne eivät ole riittävän täsmällisiä, jotta vaatimukset tässä logistiikan seurantajärjestelmä. On mahdollista yhdistää näiden järjestelmien välillä, mutta se ei ole kustannustehokasta hankkeen kannalta.

Tämän projektin tarkoituksena on luoda vuorovaikutteisia multi-touch-ominaisuuden tietoja toteutus-sovellus, jota käytetään jäljittämään paketteja ja hankkeita eri puolilta maailmaa GPS ja GPRS-tekniikalla. Yksinkertaisesti, tämä projekti on otettu käyttöön, jotta se vastaisi kysymykseen "missä on minun paketti". Siinä on yhdistetty kaksi seurantajärjestelmät: AIS-järjestelmä ja GPS / GPRS-järjestelmä, mikä parantaa luotettavuutta ja tehokkuutta.

Tutkimalla GPS / GPRS- ja AIS viestintää, hankkeessa kuvataan keskeiset ominaisuudet mahdollisia menetelmiä, joita käytetään seurannan järjestelmät, integrointi näihin järjestelmiin ja joitakin mahdollisia ratkaisuja akun ongelma, joka on ollut olemassa useimmissa seurantajärjestelmissä johtuen suuresta tehonkulutuksesta GPS / GPRS-laitteissa.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Programme

ABSTRACT

Author	Duy Nguyen
Title	Touch Screen Operated Data Warehouse Application
Year	2010
Language	English
Pages	71 + 20 Appendices
Name of Supervisor	Smail Menani

This thesis was about real time tracking system. Although there have been many existing real-time tracking systems, they are not specific enough for the demands of this particular logistics tracking system. It is possible to integrate between these systems but it is not cost effective for the purpose of the project.

The idea of this project is to create an interactive multi-touch based data warehouse application which is used to trace packages and projects from all over the world using GPS and GPRS technology. Simply, this project is used to answer the question “where is my package”. It is combined by two tracking systems: AIS system and GPS/GPRS system to increase reliability and efficiency.

Through research into GPS/GPRS and AIS communications, the project describes the relevant characteristics of possible methods which are used in tracking systems, the integration of those systems and some possible solutions for battery problem which have occurred in most tracking systems because of high power consumption in the GPS/GPRS devices.

Keywords GPS/GPRS, Tracking System, XAML, Google Earth, AIS

PREFACE

It has been two years since I started my study at the Vaasa University of Applied Sciences, Finland. During those years, I have achieved lots of useful knowledge such as designing, programming and testing software, or working with other people in projects. Now, in the last period of my study, I can use the skills which I have been taught to work on this project as my thesis.

This thesis is very interesting for me because I have learnt a lot of things after finishing it. This report is the final product of my thesis which I have been working on from August 2010 to November 2010. The project was appointed by Wärtsilä and took place in the Faculty of Technology of University of Vaasa. It was supervised by Prof. Petri Helo and Dr. Smail Menani.

Its idea is to create a logistics tracking system which gives Wärtsilä employees an access to the management of logistic processes by using future IT solutions such as various real time tracking technologies. Of course, it would not be possible to implement such a huge system in just half a year. That is the reason why I would only build the base which can be built further. This basic framework is the touch screen operated data warehouse application, written in C# and XAML language, which makes it possible to control logistic processes in the same way of known programs, like Marine Traffic from <http://www.marinetraffic.com/ais/>

I can understand that the readers themselves wonder why I need to build another system instead of using the existing systems. The reason is very simple: they are not specific enough for the requirements and demands of Wärtsilä. Another reason is that it would be very expensive to purchase that kind of system and we need to pay for monthly service although they can be adjusted for our own purpose.

Since the system was built from scratch, the project focuses mainly on the general design and the implementation of a basic working system, which then will be expanded in future projects. Therefore, it would probably take a couple of years before the system is reaching a mature state, but the final result will be very rewarding.

ABBREVIATION

LogTrack	Logistics Tracking System
XAML	Extensible Application Markup Language
IMO	International Maritime Organization
MMSI	Maritime Mobile Service Identity
AIS	Automatic Identification System
NMEA	National Marine Electronics Association
JSON	JavaScript Object Notation
XML	Extensible Markup Language
KML	Keyhole Markup Language
GPS	Global Positioning System
GPRS	General Packet Radio Service
WPF	Windows Presentation Foundation
VHF	Very High Frequency
TCP/IP	Transmission Control Protocol and Internet Protocol
SVN	Apache Subversion
RFID	Radio Frequency Identification
GTIN	Global Trade Item Number
IT	Information Technology
GUI	Graphical User Interface

CONTENTS

TIIVISTELMÄ

ABSTRACT

PREFACE 4

ABBREVIATION 5

1 INTRODUCTION..... 9

1.1 An introduction about Wärtsilä..... 9

1.1.1 Ship power..... 9

1.1.2 Power plants 9

1.1.3 Services 9

1.2 The situation of existing tracking systems 10

1.3 Statements of problem..... 10

1.4 Scope and limitation..... 11

2 THE STARTING POINT 13

2.1 Description of the situation..... 13

2.1.1 The current system (Marine Traffic system)..... 14

2.1.2 The desired system 15

2.2 Description of technologies used in this project..... 17

2.2.1 Global Positioning System..... 17

2.2.2 Automatic Identification System..... 17

2.2.3 Extensible Application Markup Language..... 18

2.2.4 Google Earth Plug-in and KML 18

2.2.5 Android Operating System..... 19

2.2.6 Windows Presentation Foundation 19

3 PROJECT DEVELOPMENT PROCESS..... 21

3.1 4.1 Requirements and resources: 22

3.1.1 Requirements 22

3.1.2 Resources 23

3.2 Solution 24

3.3 Plan of action 26

3.4 The approach 27

3.5 Conclusions 29

4	TECHNICAL ARCHITECTURE.....	31
4.1	The database	31
4.2	The architecture of the system.....	31
4.2.1	About LogTrack system.....	31
4.2.2	About process	31
4.2.3	LogTrack Integration Technology.....	33
4.2.4	Android phone as GPS/GPRS device	35
5	THE IMPLEMENTATION	40
5.1	The server	40
5.1.1	AIVDM protocol (used in AIS communication).....	42
5.1.2	AND_UPDT protocol (used in Android phones).....	46
5.1.3	GPS/GPRS protocol (used in Chinese device).....	47
5.2	The Android application.....	47
5.2.1	Configuration for Android phones.....	47
5.2.2	Implementation of the Android application	51
5.2.3	Implementation of Arduino application.....	53
5.3	The client.....	55
5.3.1	Configuration for the client	55
5.3.2	General layout of this client	56
5.3.3	Description about components and searching methods.....	58
5.3.4	Communication between Google Earth and .NET framework ...	63
6	PROBLEMS ENCOUNTERED	64
6.1	Battery life problem	64
7	DISCUSSION AND CONCLUSION	67
	REFERENCES	69
	APPENDICES.....	72
	Appendix A – LogTrack Database.....	72
	A.1 LogTrack Main Schema.....	72
	A.2 LogTrack Device Schema.....	74
	A.3 LogTrack Data Schema	75
	Appendix B – AIVDM protocol	76
	Appendix C – Data Format for Chinese device	81

C.1 GPRS Data Format 81

C.2 NMEA Data Format 82

Appendix D - A description about KML class 85

1 INTRODUCTION

This chapter will give some introduction about the project and its owner, Wärtsilä, what the situation of existing tracking systems is and why this project is needed.

1.1 An introduction about Wärtsilä

“Wärtsilä is a global leader in complete lifecycle power solutions for the marine and energy markets [1]”. By emphasizing technological innovation and total efficiency, Wärtsilä maximizes the environmental and economic performance of its customers’ the vessels and power plants.” In 2009, Wärtsilä’s net sales totaled EUR 5.3 billion with 18000 employees. The company has operations in 160 locations in 70 countries around the world in 2010. Their main products include ship power, power plants and services.

1.1.1 Ship power

Wärtsilä is the leader of ship power solutions including engines, generating sets, reduction gears, propulsion equipment, automation and power distribution systems as well as sealing solutions for the marine industry. Their customers are global or local companies within the merchant, offshore, cruise and ferry, navy and special vessel segments. They are in a strong position of a supplier of highly rated ship machinery and systems in all main marine segments.

1.1.2 Power plants

Wärtsilä is a leading supplier of flexible power plants for the decentralized power generation market. This company also offers solutions for power generation, industrial self-generation as well as for the oil and gas industry.

1.1.3 Services

Wärtsilä provides services to support its customers throughout the lifecycle of its installations by optimizing efficiency and performance. It also provides the broadest portfolio and best services in the industry for both ship power and power plants.

1.2 The situation of existing tracking systems

It has been mentioned in the beginning of this section that Wärtsilä is a global company and it has a plenty of transportations which are done mostly by vessels. These vessels can be tracked by Marine Traffic based on the information which suppliers provide for such as IMO number, MMSI number or call sign. Nonetheless, the problem is that the location of vessels is the only information we can know from this system.

In addition, there are several tracking systems available through GPS, GTIN [2], and RFID [3], Barcode etc; nonetheless, all these systems are not fully compatible for industry. “Most of the available tracking and tracing systems utilize proprietary tracking numbers defined by the individual companies operating systems and are based on information architecture, where the tracking information is centralized to the provider of the tracking service” [4]. The contents within a box cannot be identified by existing tracking systems for example, if the box is open or the contents are lost or stolen etc. Moreover, they cannot sense the surrounding environment and collect environmental data such as temperature, humidity, rust sources, vibration/shock etc. This information will help them prevent rust from engines.

Without proper and suitable tracking and tracing system, efficient co-ordination of logistic flow would be difficult to acquire [5]. Through the implementation of this system, it is possible to detect and react to any uneven situations in the logistics chain and where needed significant problems can be resolved or at least the damage can be minimized [6, 7]. However, why is this kind of tracking system needed? This question will be answered in the next part.

1.3 Statements of problem

Tracking and tracing of shipments are considered highly important to manufacturing firms in terms of customer service and essential for managing logistics networks efficiently. “Global industries are facing problems both from tracking and tracing in their logistics supply networks, which create huge coordination prob-

lems in the overall product development sites [4]”. This problem loses the track among production, delivery and distribution in the complete logistics chain from source to destination, which is responsible for opportunity cost through customers’ dissatisfaction. Tracking system helps to identify the position of the shipment and informs the customer well in advance. Without tracking system it is almost impossible to find out delivered items and these items are often considered to be lost or stolen, which causes business loss [8, 9]. In fact, many projects have been in threat of cancellation because of delayed packages or lost packages. And it, in the end, would result in lost revenue, lost business or further damage to company’s reputation.

This system is designed to solve those problems. It was appointed by Wärttilä and its idea is to trace packages and projects around the world through GPS/GPRS devices and central stations. By integrating between AIS and GPS/GPRS system and using new features of XAML and Google Earth, the application gives access to packages which are shipped from countries to countries, tracks them, manages them and maintains them.

1.4 Scope and limitation

This project provides a possible solution to track packages and maintain them. By analyzing, updating and showing data received from GPS/GPRS devices on Graphical User Interface (GUI), the current situation of packages and projects can be known. The implementation of the program that is directed by those purposes will be discussed, described and analyzed. This application will only focus on AIS and GPS/GPRS tracking system. Other tracking systems will be ignored. Moreover, the reasons of using XAML, Google Earth and touch screen PCs will be explained in the scope of this thesis.

However, like other tracking systems, it also has its own limitations. Battery problem is one of the limitations of GPS/GPRS devices. This application provides some possible approaches which are used to optimize this problem. But to provide a complete solution for this battery problem is out of scope of this project. In addition, this application supports only some of specific protocols such as “AIVDM”

protocol (raw AIS data) or “AND_UPDT” protocol (which is defined in this project, using for Android phones). Other protocols will not be accepted.

2 THE STARTING POINT

This chapter will describe the situation as it was before starting the project and technologies used in this project.

2.1 Description of the situation

As mentioned in the beginning of chapter 1, Wärtsilä is a global manufacturer and it has lots of transportations, as illustrated in figure 1.



Figure 1. Global delivery network. [28]

All the shipments are usually shipped to the consolidation port. If the shipments are located in Europe, they will be generally shipped to Hamburg, Germany. After that, they will be stored in the warehouse until all shipments come. Then, they will be assembled here and delivered to customer (usually shipyard).

2.1.1 The current system (Marine Traffic system)

Marine Traffic system is based on UAIS, known as AIS, which is a civilian information system that makes it possible to exchange data between ships and land based stations [10]. The system has been introduced by the International Maritime Organization (IMO) and it is basically an anti-collision system for vessels at sea. The system is also largely suitable for monitoring the ship traffic in coastal areas. A ship equipped with AIS transponder transmits information, including ship's name, position, course, speed, draft, type of vessel etc, via radio communication to other ships equipped with AIS as well as land based AIS stations. The land based stations can send short messages to AIS equipped vessels within in a given area, thus providing marine traffic with important traffic and safety related data.

Marine Traffic uses this information to plot the real time position of marine vessels on Google Map. From this system, we can know the real time position of all marine vessels in the coverage map because The International Maritime Organization required all vessels over 299GT to carry an AIS transponder class A on board. The vessel's position is shown on map in shape of boat with color. There are many types of vessel and each of them has its own color. Clicking on these tags will display the information of vessel, its current destination and the route which it has moved.

This system is being expanded all the time because anyone can install an AIS receiver, a VHF antenna and share the data receiving from AIS transponders. However, there are still some problems with this system. From the specification of this system, it is clear to see that it does track only vessels and we cannot collect environmental data from this system. Therefore, they need another solution for their production logistic network because they want to track packages/projects all the time and all the way.

Figure 2 shows the coverage map of Marine Traffic system until now and figure 3 displays the main page from <http://www.marinetraffic.com/ais>.

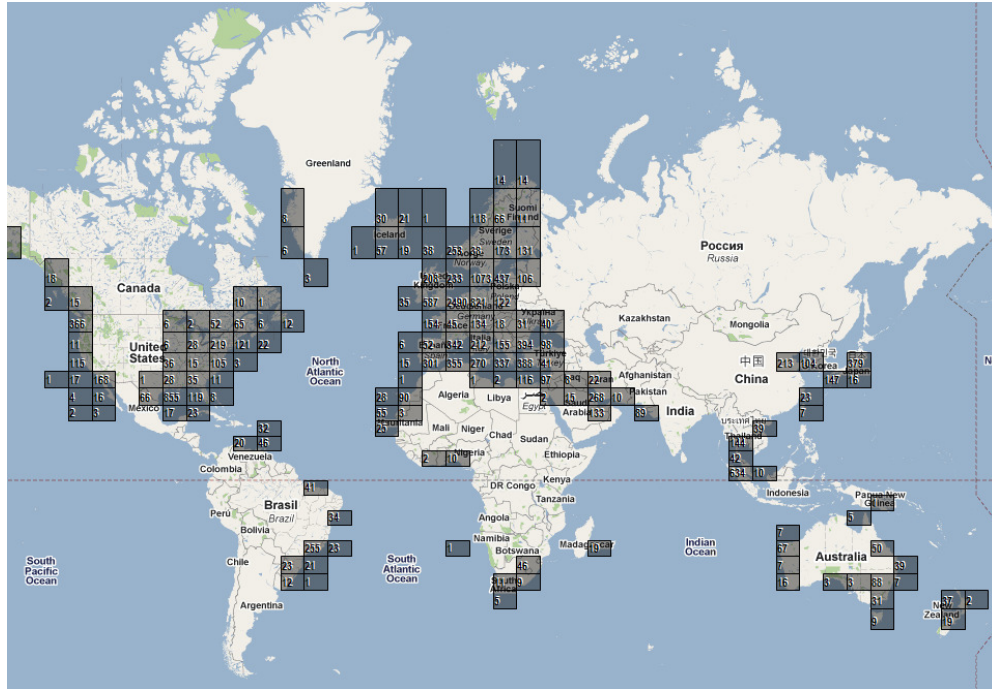


Figure 2. The coverage map of Marine Traffic. [10]

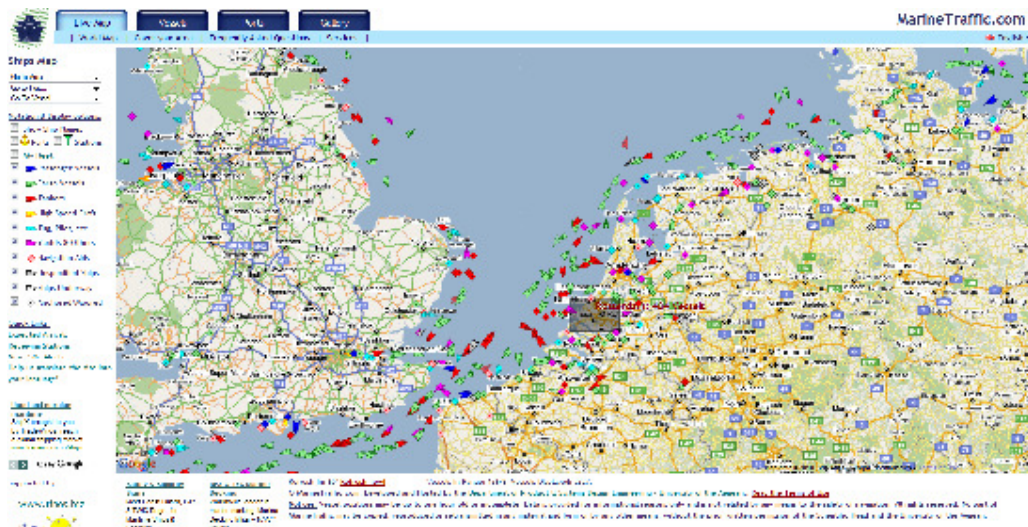


Figure 3. Marine Traffic main page. [10]

2.1.2 The desired system

Project's target is to integrate this LogTrack system with the existing Marine Traffic system. In addition, there will be other requirements and specifications for the

desired system because project's objective is to create a future model which is able to manage the Wärtsilä production network logistic systems by using future information technology system solutions as a formidable tool by building a pragmatic feasibility study and then following through as such. It should be possible to track shipments all the time and all the way from factories, suppliers through consolidation points and then, finally delivery to customer, even when the packages are on land or at sea. The system must ensure that shipments could be tracked in real time and tracking data is easily accessible. The tracking route of desired system is shown in figure 4.

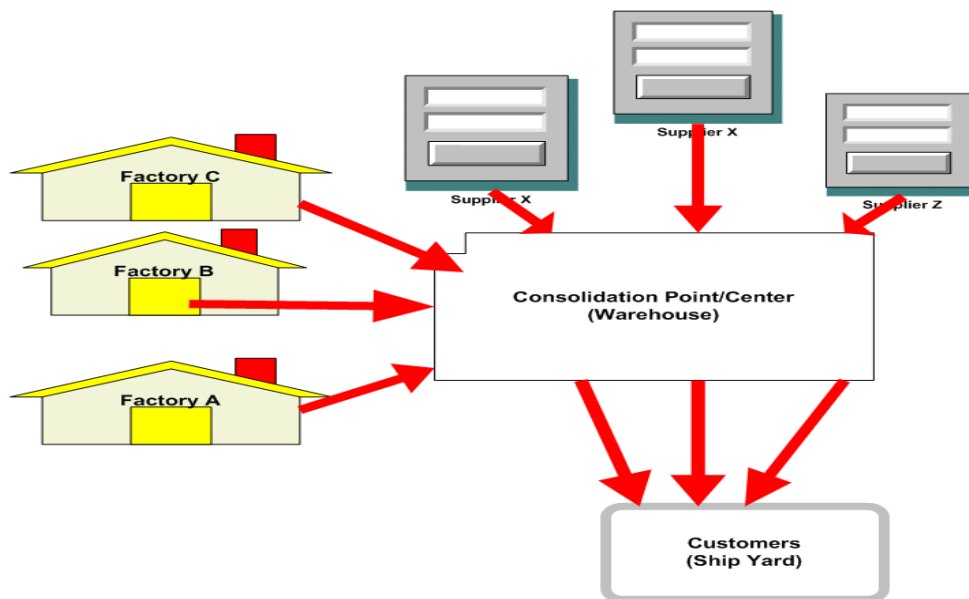



Figure 4. Tracking route for the desired system. [30]

 Red arrows in the diagram illustration denote shipment / transport routes that could be tracked for information.

Moreover, the system must be capable of collecting data on humidity, temperature, rust sources and vibration/shock and the battery life for GPS tracking devices should last more than 2 weeks because it is the minimum time for the whole transportation.

2.2 Description of technologies used in this project

This section will give a short description about technologies used in this project and how to take advantages of them.

2.2.1 Global Positioning System

Global Positioning System (GPS) is a global navigation satellite system which provides location and time information of any objects which implement GPS technology at all time and anywhere on the earth. This new technology has been established in the end of 20th century and brought back many benefits, especially in tracking solutions. Nowadays, there are more and more devices which support GPS technology such as GPS navigation softwares which use some algorithms to calculate direction or route from point A to point B and draw it on a real-time map. TomTom Navigator is an example in this case. It is quite popular and is used in most of car navigation systems.

However, similarly to other tracking systems, GPS tracking systems still have some limitations. Although it is very accurate in tracking objects, the startup performance of a GPS satellite-based positioning system is quite slow and the signal is sometimes unstable, especially in very poor signal conditions such as in the bad weather condition or in the middle of city where the GPS signal is blocked by high buildings or weakened by passing walls and trees. Due to these disadvantages, the US accelerated the development of new system, Assisted GPS. Assisted GPS (A-GPS) is an extended GPS system. It is used extensively in cellular phones to help to improve startup performance of GPS system by using data available from a network. Therefore, in this application, A-GPS was used because of its more compelling characteristics.

2.2.2 Automatic Identification System

According to Wikipedia, Automatic Identification System (AIS) is an automatic tracking system used on ships and Vessel Traffic Services (VTS) to identify and locate vessels by exchanging data with nearby vessels and VTS station. It uses VHF (very high frequency) to transmit and receive data.

Information provided by AIS equipment, such as name of vessel, its position, speed, heading degree, course, IMO (International Marine Organization) number..., can be displayed on screen. This information is used in collision avoidance, aids to navigation, search and secure. The basic idea of AIS systems is very simple. It uses AIS receivers to transmit data receiving from AIS transmitters to central server. By analyzing data received from AIS transmitters and receivers, we can know where our packages are. This process has been completely implemented with help from <http://aprs.fi>.

2.2.3 Extensible Application Markup Language

Extensible Application Markup Language (XAML) is a declarative markup language which was developed by Microsoft and used extensively in .NET Framework 3.0, 3.5 and 4.0. In Window Presentation Foundation (WPF), it is used as a user interface language to define UI elements, data bindings, and other features. It is separated with run-time logic by code behind files. In other words, when developing in .NET Framework, it is possible to create visible UI elements in XAML mark up files at first, and then apply logic later.

Cool features of XAML language have been used to develop this application. The target operating system was Windows 7 and the target hardware was touch screen PCs. These new features will help increase visual effect in creating an interactive multi-touch application.

2.2.4 Google Earth Plug-in and KML

Google Earth is a free service and available for any applications or web sites which are free to consumers. It is a powerful feature which helps embed a true 3D globe into our application and supported by Google. In addition, its API is easy to use and supports KML language.

KML is an open-standard markup language for the display of geographic data in geo-browsers. Google Earth can import KML in different ways to draw lines, place marks, 2D/3D objects or display features, views or tours. There are many web sites which provide KML library to help developers develop their application

using Google Earth feature. However, in this application, I implemented a new KML class which is used only for this touch screen operated data warehouse application. The purpose of this KML class is to translate C# objects, structs and classes into KML string so that we could import them into Google Earth. More detail about this class will be described in section 6.3.5 and Appendix D.

2.2.5 Android Operating System

Android is a mobile phone operating system built on the Linux kernel. It was developed by Google and later positioned to Open Handset Alliance, an association of several companies. It is programmed in Java language and supported many third party libraries. By taking good points of this new operating system, I developed a new Android application. This application will be installed on an Android phone which is attached to each package as a client. This virtual client worked as a fully functional GPS/GPRS device, used in tracking, managing and maintaining packages. For example, it can send information including location, time, temperature, vibration, speed, and destination of transportation to the central station through cellular or WIFI network. At central station, the information will be processed, updated and shown on user interface.

2.2.6 Windows Presentation Foundation

Generally, this project was written in WPF language. But why is this language used? At first, WPF is a powerful, easy to use programming language running on Windows operating system, which is supported by Microsoft. It is a graphical subsystem for rendering user interfaces in Windows-based applications [11]. WPF uses XAML, a declarative of XML, to define UI elements, data binding and events. The most important thing is that it is easy to render 3D objects in WPF, using Direct3D engine which is a part of Microsoft's DirectX API, and interactive 2D elements can be drawn on 3D surfaces. This will help to improve visual effect of the application.

Moreover, our target hardware is multi-touch screen PCs and specific drivers are needed for this kind of new technology. Windows 7 Multi-Touch operating sys-

tem including .NET Framework 4.0, which supports touch screen PCs running Windows operating system, has been released in beginning of this year, 2010. This feature meets our requirement and obviously, compared to other solutions, this definitely is the best one.

3 PROJECT DEVELOPMENT PROCESS

The situation and the direction of this project have been clearly defined. Thus, it is time to give a description of what will be built in this project.

As mentioned before, this thesis is merely a small part of the desired system described in chapter 2.1.2. However, it still has its own objectives which need to be done during the processing of the project. This chapter will give a description about project development process, as illustrated in table 1.

Task No.	Description	Priority
1	Fully implement the LogTrack application which is used to display visual data and process measurement and analysis of ecological footprint of network logistics.	1
2	Conduct a feasibility analysis for real-time tracking technologies, to recognize potential partners in delivering IT-solutions.	2
3	Develop methods to analyze the ecological footprint of network logistics and manage logistic processes by using future IT-solutions such as various real time tracking technologies.	3
4	Implement a basic framework for the central station .which is used to receive and process logistic data.	1
5	Test the application as a complete system.	1

Table 1. Quality Function Deployment (QFD) of the project.

3.1 4.1 Requirements and resources:

In order to complete the project successfully, it is important to know what the requirements of the project are and what kind of resources it can use.

3.1.1 Requirements

The purpose of using tracking devices is to avoid delayed and lost packages and prevent rust on main engines and generating sets. To reach this desired goal, they should be implemented with the following functionalities:

- GPS and GPRS modules should be installed on trackers so that they can send information about location in real-time.
- Temperature sensor, humidity sensor and acceleration sensor should be implemented on tracking devices to sense the surrounding environment.
- The battery life for tracking devices should last at least 2 weeks
- Magnetic sensor should be mounted on trackers in case if someone opens the box, it will send alarm to the central station.

In addition, the server and the client also have their own requirements:

- The server and application must be written in C#.
- The application should use cool features of XAML language as much as possible.
- It should have 3D zoomable map to display information of vessels and projects.
- The server can handle multiple trackers at the same time.
- Windows 7 multi-touch operating system should be used to create an interactive application.

3.1.2 Resources

During this thesis, the following software applications need to be installed in order to work in a convenient way:

- Microsoft Visual studio 2010 which is used to program C# and WPF language.
- MySQL database with MySQL Query Browser to access to the database. This database is used to store the information for the whole system.
- An editor like Notepad++ to edit KML files and XML files.
- A web browser, where I can store the HTML page which is used to display Google Earth plug-in.
- Eclipse and Android IDE to program Android phones.
- Arduino IDE to program Arduino micro-controller.
- A terminal like TeraTerm software to configure Bluetooth module.
- Windows 7 multi-touch operating system.

The above list of software applications contains basic software applications which are used to program this system and to extend the project in the future.

In addition, there are also software packages which make it easier to work on this project:

- A version control system like SVN in order to maintain versions of the project files.
- A hibernate core library like NHibernate in order to make connection between system and database.
- .NET framework 4.0 in order to compile the software.

- A Bluetooth library such as Amarino to connect Arduino micro-controller and Android mobile phones.
- MySQL connector for .NET environment.
- WPFToolkit library in order to render 2D elements on 3D surface.

Besides software, there are also necessary hardware components which are used in this project:

- An Arduino micro-controller.
- Temperature sensor and humidity sensor.
- A Bluetooth module such as BlueSMiRF 2.0.
- An Android phone with GPS and acceleration sensor such as HTC Legend.
- A touch screen computer with 22 inch screen. In case touch screen computer is not available, a normal computer can be used because the application works on both of them.

3.2 Solution

During the thesis there was only one major choice which had to be made between different solutions. This part will go through this choice and explain what its benefits are and why it was chosen.

The major choice which had to be made is the selection between normal GPS/GPRS devices and an Android phone. Should the Android phone be programmed as a GPS tracker or is buying a GPS/GPRS device better because they are available on the market and the prices are declining now? Obviously, building a new GPS/GPRS device from scratch will be better but it will be too much for only one person working on the project in 4 months. And of course, this system is about not only tracking devices but also a server and an interactive application. That is the reason why the Android phone was chosen because it is cheap, power-

ful, programmable, full of functionalities, good battery and easy to integrate with a micro-controller for collecting environmental data.

Figure 5 illustrates the basic structure of LogTrack system which I did come up after this selection:

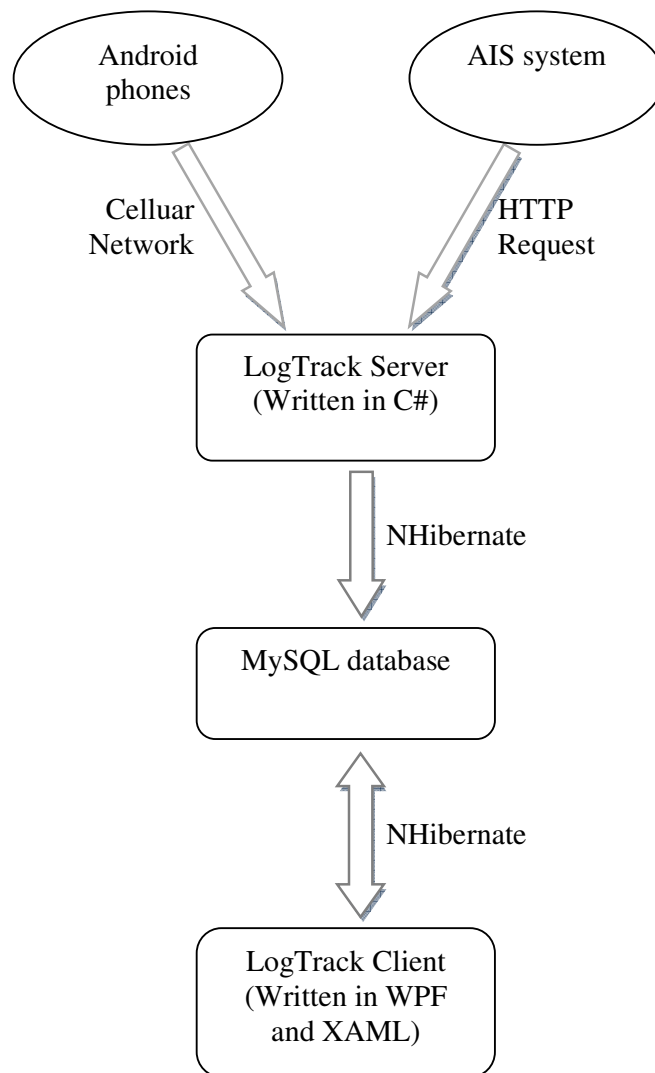


Figure 5. Basic structure of the LogTrack system.

Basically, this project is divided into two separated parts: client and server. Each of them has its own functionalities.

The server was written in C# language and runs as a background service. It was connected to MySQL database using hibernate technology. Data received from Android phones and AIS system were processed by this server, and later saved into database.

The client was written in WPF and XAML language. One of the good points of WPF is that it can be deployed as a standalone desktop program or hosted as an embedded application in a web site. To create an interactive multi-touch application, cool features of XAML and Google Earth have been used. The client was also connected to MySQL database using hibernate technology.

In addition, there are also Android phones. These Android devices communicate with the central station through cellular network such as 3G, EDGE or GPRS. It will periodically send data including GPS location, temperature, humidity, vibration, and speed ... to this central station. This information is collected through a micro-controller by Bluetooth communication.

3.3 Plan of action

Here is the plan which was created in the beginning of the project:

Date	Action
August 2010	<ol style="list-style-type: none"> 1. Investigate on GPS/GPRS devices on the market and Android phones. 2. Define the requirements for the project based on the information received from Wärtsilä. 3. Design and implement a framework based on the requirements mentioned above.
September 2010	<ol style="list-style-type: none"> 1. Implement the application. Learning about WPF and XAML at the same time.
October 2010	<ol style="list-style-type: none"> 1. Implement the Android phones and the server 2. Investigate on Arduino micro-controller and Blue-

	tooth communication. 3. Implement the Arduino micro-controller.
November 2010	1. Test the system and prepare the documentation.

In addition, it is essential to know the milestones of the project. The project milestones which were established in the first meeting with Wärtsilä are described in table 2:

	Date	Project Milestones
G1	01/09/2010	Project Planning
G2	30/09/2010	Project Kick-Off
G3	31/10/2011	Handover
G4	31/12/2011	Project Closing
G5	31/05/2012	Handover Implementation and Closing

Table 2. Project milestones.

3.4 The approach

An essential subject of one project is to know how team members tackle the problems during that project. It is, of course, possible to jump into programming, but probably after few weeks, you are completely stuck. Hence, it is very important to know which software development methodology is used during the project development process.

During the time working at Wapice as a software developer, I did learn many software development methodologies. Each of them has its own advantage. However, spiral model has been used in this project because of its more compelling strengths. The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine ad-

vantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model (or spiral development), it is a systems development method (SDM) used in information technology (IT) [12]. The diagram of this model is illustrated in figure 6.

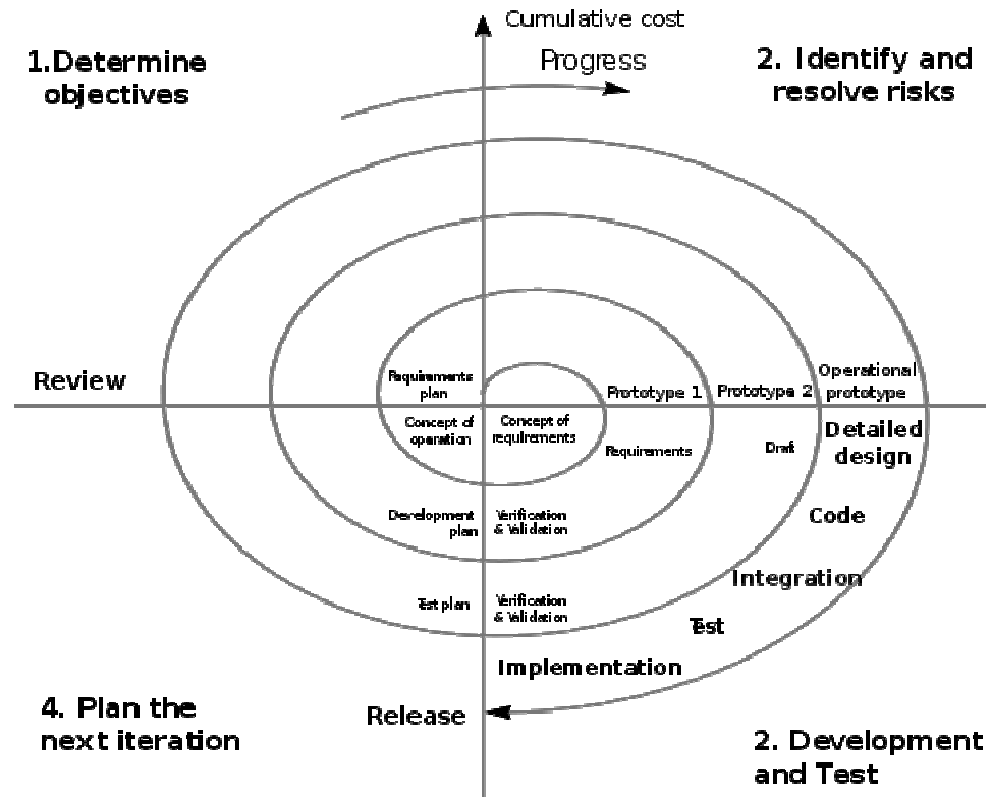


Figure 6. Spiral model (Boehm, 1988). [12]

In other words, it is the combination of many short cycles of designing, implementing and testing. The project will be designed at the first stage and divided into many small parts. Each part will be designed and implemented separately. Finally, it will be tested and if everything works properly, it will be integrated into the main application, where it will be tested again as a whole. In this way programming errors are found quickly and it is easier to fix these errors because if you first build half of the application according to the design, everything seems to be alright, but when the testing fails, you have to investigate thousand lines of code to find such a small error that can be easily overlooked.

The steps in the spiral model iteration can be generalized as follows: [12]

- The system requirements are defined in as much detail as possible.
- A preliminary design is created for the whole system. This phase is the most important part of "Spiral Model". In this phase all possible (and available) alternatives, which can help develop a cost effective project are analyzed and strategies to use them are decided. This phase has been added specially in order to identify and resolve all the possible risks in the project development. If risks indicate any kind of ambiguity in requirements, prototyping may be used to proceed with the available data and find out possible solution in order to deal with the potential changes in the requirements.
- A first prototype of the system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
- A second prototype is evolved by a fourfold procedure:
 - Evaluating the first prototype in terms of its strengths, weaknesses, and risks.
 - Defining the requirements of the second prototype.
 - Planning and designing the second prototype.
 - Constructing and testing the second prototype.

3.5 Conclusions

Finally, after working on this project for 4 months, the following conclusions can be drawn:

- During the project, I have learned a lot about programming in C# and Windows Presentation Foundation, especially using binding, dependency objects, database and socket programming.

- When it comes to project planning, I need more experience. It is very difficult to make a plan for a project, especially when building from a scratch. Lots of things need to be investigated and several choices have to be made.
- I need to learn a lot about electronics and embedded system, in this case the micro-controller and the Android phone. This was very nice experience for me because I am quite interested in hardware stuffs. I learnt the use of an Android phone as a tracking device and something I have never learnt before: communication between micro-controller and Android phone using Bluetooth technology.

The meetings were held every week in Wärtsilä office to process the status of this project. I got many useful experiences from these kinds of meeting such as how to manage a big project, how to deal with business problems or how to solve problems logistically etc.

4 TECHNICAL ARCHITECTURE

4.1 The database

The database is a very important part of the Logistics Tracking system because all information used throughout the application is stored here. A detail description about the database and its tables is explained in appendix A.

4.2 The architecture of the system

4.2.1 About LogTrack system

Name LogTrack comes from “Logistics Tracking System”. This system is considered as the link between the information systems and the physical reality (the material flow) in the logistics network. Its main purpose is to track packages and store their information into database for maintaining purpose. It does not manage packages and projects by itself. Instead of that, it provides visual information and necessary data for users via integration systems. Users can use these data for other systems or just for their purpose. In addition, users can search projects and packages by many methods. These methods will be explained in detail in chapter 5.

4.2.2 About process

The whole process of LogTrack system is defined in detail in below sections, but just to give an idea, the process is explained here (figure 7):

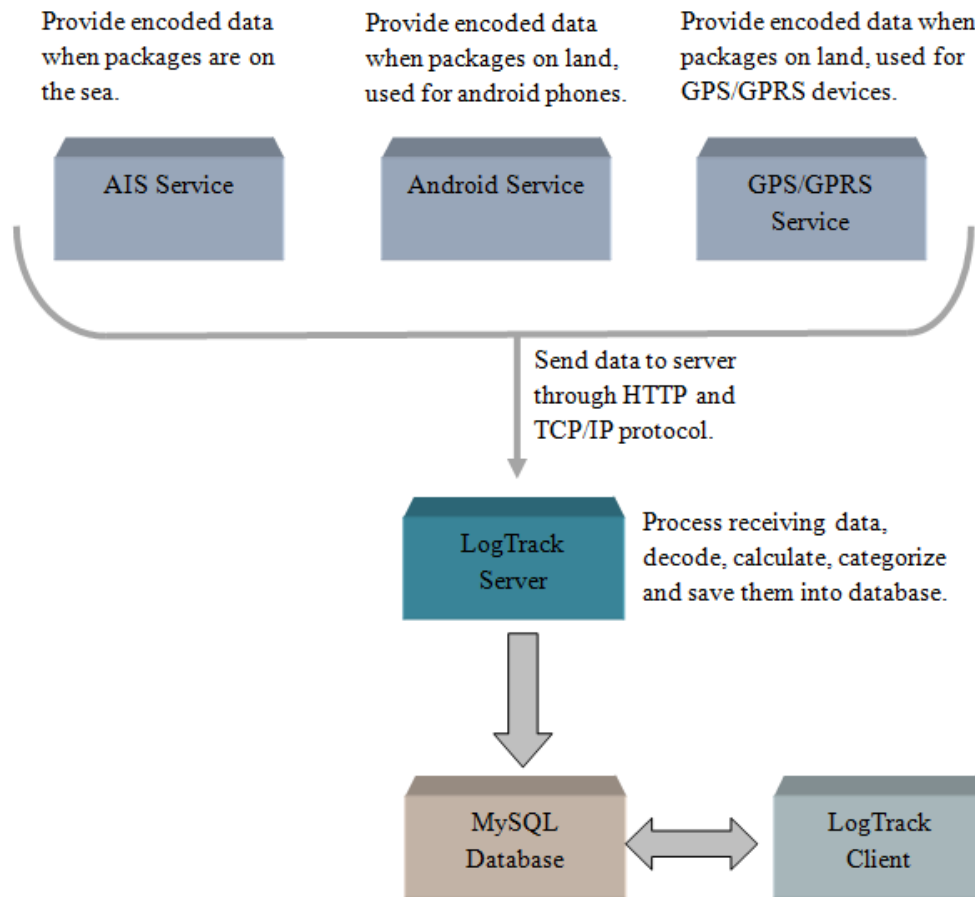


Figure 7. The process of Logistics Tracking system.

Data will be encoded by these services before sending to central station. At the central station, data will be decoded and save into database. The information of devices such as battery level, signal strength, status and so on... will be checked periodically and if something happens (low battery or loss signal), server will send notifications to services and client. On the client side, data will be retrieved from database and updated on GUI. The notifications will be shown on GUI if they are available.

4.2.3 LogTrack Integration Technology

4.2.3.1 About AIS system

AIS or Automatic Identification System is an automated tracking system used on ships and by Vessel Traffic Services (VTS) for identifying and locating Vessels by electronically exchanging data with other nearby ships and VTS stations. AIS information supplements marine radar, which continues to be the primary method of collision avoidance for water transport [13]. Information provided by AIS equipment, such as name of vessel, its position, speed, heading degree, course, IMO (International Marine Organization) number..., will be sent to AIS stations. From these data, we will know where our packages are.

The following picture (figure 8) illustrates the basic structure of an AIS system:

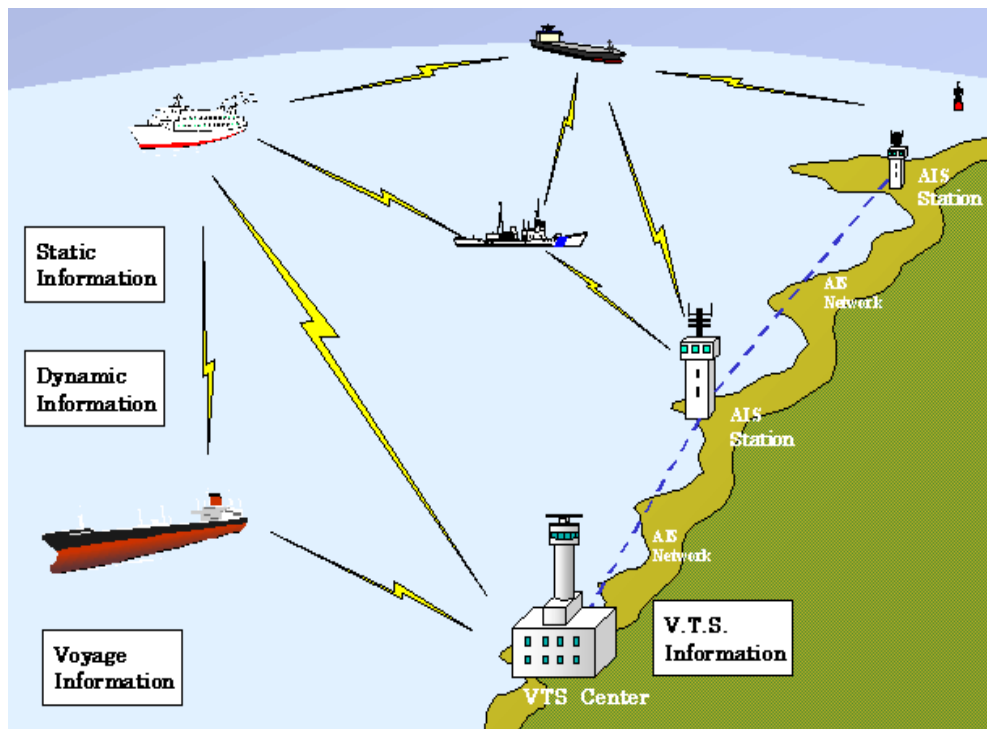
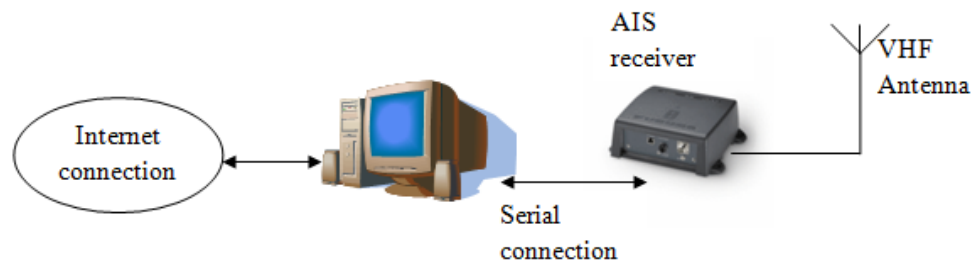


Figure 8. Basic structure of AIS system. [29]

4.2.3.2 How to integrate AIS into LogTrack system

To build an AIS station, we need at least an AIS receiver, a VHF antenna and a computer with internet connection:



Specific software (such as AIS Logger) needs to be installed on this computer to process AIS data which is received from AIS receiver and send it to a central server. This process is quite complicated and cost lots of money because we need many AIS stations to cover a big area.

As a result, integrating the whole AIS system into this application is quite complex and out of scope of this thesis. However, there is another solution and it is the API from <http://aprs.fi>. APRS or Automatic Package Reporting System is not a vehicle tracking system. It is a two-way tactical real-time digital communications system between all assets in a network sharing information about everything going on in the local area [14]. The system from this website has provided an API which allows developers to query aprs.fi database and retrieve AIS data from other existing AIS stations through HTTP protocol. Data is returned in either JSON or XML format (JSON has been used as a default format because of its smaller overhead when compared to XML). This API is used for only non-commercial purpose.

4.2.3.3 Limitations of AIS stations and Solutions:

Although AIS is very useful in navigation and tracking, it still has some limitations such as range limitation. AIS transponders broadcast information, including

their position, speed, name of vessel, navigation status, at regular interval via VHF transmitters built in on ships. Those data can be displayed on the screen of AIS system of vessel or progressed in other ways by AIS station.

However, VHF transmitters have their own limitation. VHF is the radio frequency with range from 30 to 300MHz. It is ideal for short distance terrestrial communication. VHF transmission range is a function of transmitter power, receiver sensitivity and distance to the horizon [15]. But normally, VHF signals propagate under normal conditions as a line-of-sight phenomenon:

Distance in kilometers = SQUARE_ROOT (12.746 * Am), with Am is the height of antenna in meters.

Assuming that our AIS station has a 50-m antenna, the line-of-sight range of our AIS receiver is approximately 25.245 km

Nevertheless, by exchanging data between vessels and between AIS stations, we can expand the coverage area.

4.2.4 Android phone as GPS/GPRS device

Another issue concerning to the process of this LogTrack system is about communication between Android devices and central station. Just like other GPS/GPRS devices, these Android phones will connect to central station through cellular network. TCP/IP protocol has been used in this communication because it is fast and easy to establish the connection. It is two way communication and used in most of client-server application.

4.2.4.1 About Android phones

Android phones have been growing quickly recently. It has a large community of developers writing application programs that extend functionality of devices. It is the second most popular mobile development environment [16]. The application programs are written in Java language and submitted on Android Market which is hosted by Google. Another feature is that most of android phones support GPS sensor. It helps us in developing a tracking application because if we attach this

mobile phone to packages, we will know where our packages are by retrieving GPS data. In addition, it is a low power consumption device. This feature is very important because GPS/GPRS devices generally drain energy very fast.

Based on these advantages, an Android application has been developed. This application was installed on an Android phone and runs as a background service. When it runs, it will send data including GPS location, temperature, vibration, speed and so on to the central station, using AND_UPDT protocol. The description of this protocol will be explained in detail on section 2.4.4.

4.2.4.2 About Arduino micro-controller

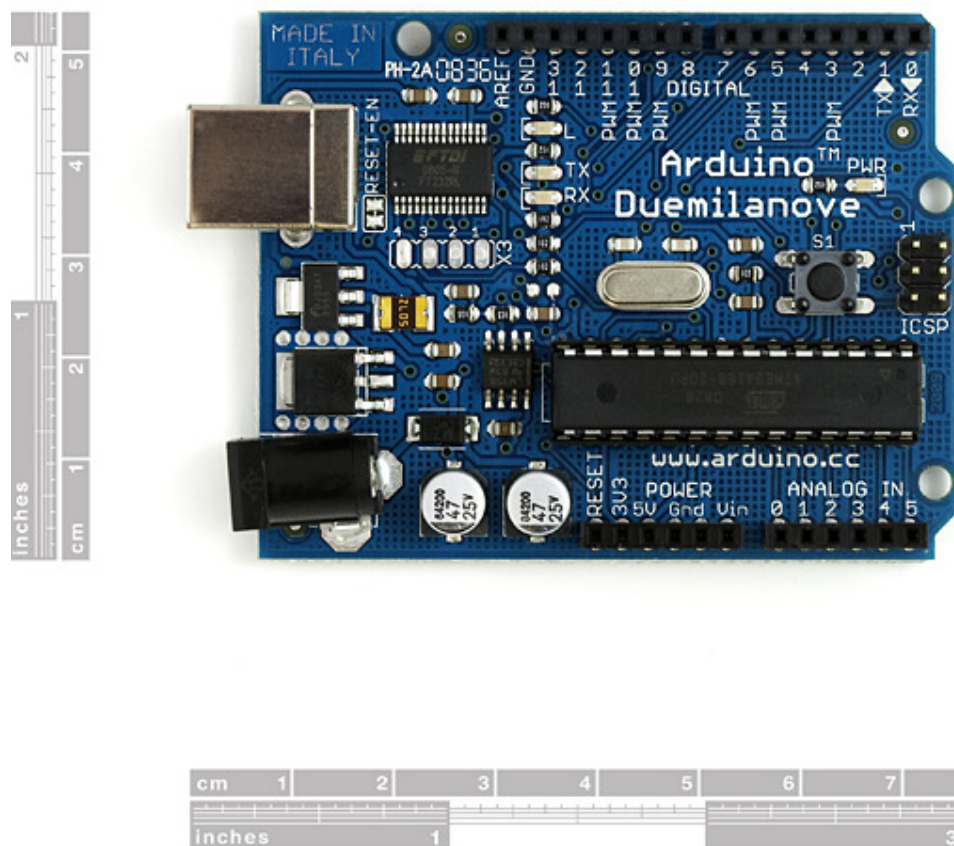


Figure 9. Arduino Duemilanove main board (from sparkfun.com). [27]

Our purpose is not only to trace packages but also to manage and maintain them. Through Android phones, we will know where our packages are. But to manage

and maintain them, more information will be needed. This is the reason why an Arduino micro-controller was used in this system (figure 9).

Arduino is a single board micro-controller. It is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software [17]. It can sense the environment by receiving inputs from a variety of sensors and send this information to an Android phone, using Bluetooth communication. Thereafter, this data will be sent to central station for processing. Following are some features of this new micro controller:

- ATmega328 microcontroller
- Input voltage - 7-12V
- 14 Digital I/O Pins (6 PWM outputs)
- 6 Analog Inputs
- 32k Flash Memory
- 16Mhz Clock Speed

4.2.4.3 Communication between Android and Arduino

Bluetooth is a wireless communication standard for exchanging data over short distance (10-100m). It is considered as an alternative of RS-232 data cable. It is primarily designed for low power consumption, short distance communication between fixed and mobile devices using frequency-hopping spread spectrum technology. It can connect up to 7 devices at one time in a mini network. Moreover, because devices are connected through a radio (broadcast) communication system, they do not have to be in the line of sight of each other and especially, it is considered as a low cost type of networking thanks to cheap price when attaching a Bluetooth module to a device (figure 10).

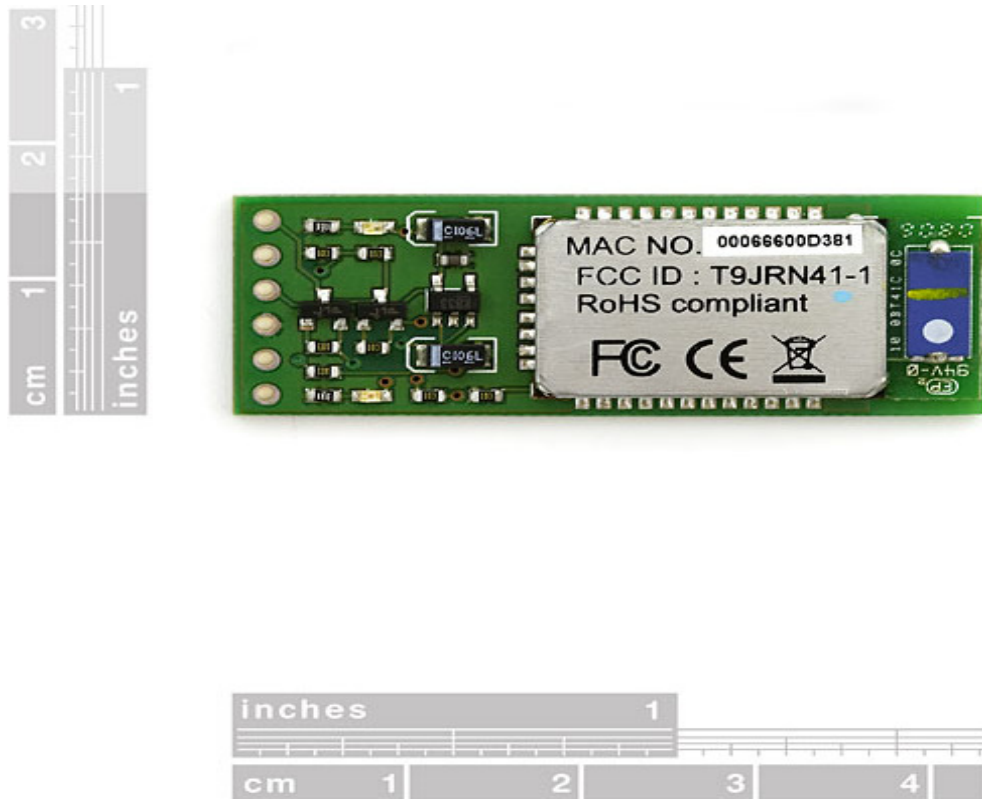


Figure 10. Bluetooth Module - BlueSMiRF Gold (from sparkfun.com). [27]

More details about the configuration of this Bluetooth module will be explained in chapter 5.2.1.

Figure 11 illustrates the architecture diagram of this system.

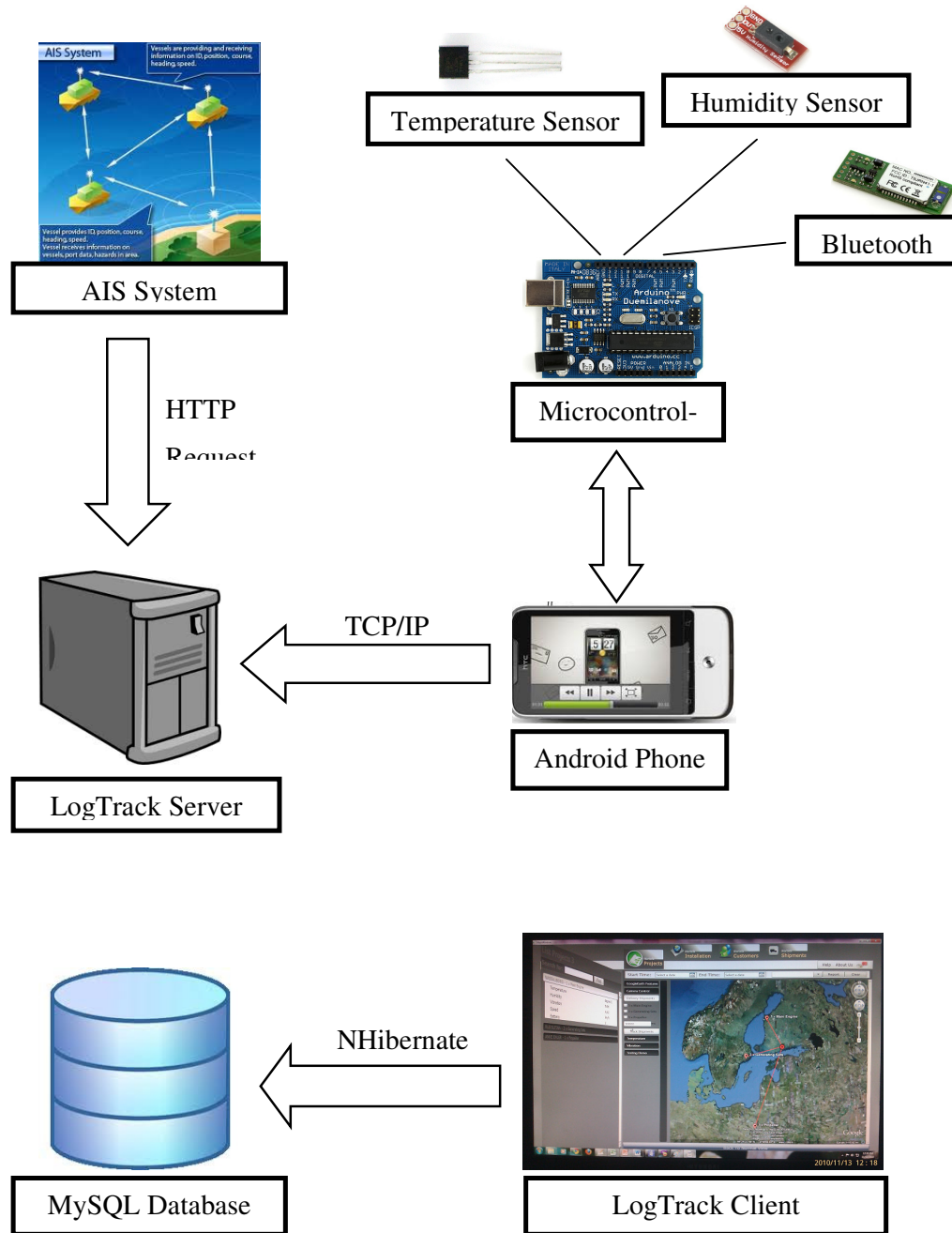


Figure 11. Architecture design of the system.

5 THE IMPLEMENTATION

In this chapter, the application will be discussed: how it works and how it is implemented. The discussion starts with a detailed description of LogTrack server: how does it work and how is it implemented? Then, the Android application and communication between Android and Arduino, which sends and receives messages to and from LogTrack server, will be explained. Last but not least, the client, which is the main part of the application and is the closest to users, will be described.

5.1 The server

The server object is used to start or stop the communication between tracking devices and LogTrack system. In other words, it will process all information which is received from trackers and save them into MySQL database. This information will be used to display visual data in the client.

Following is the class diagram of the LogTrack server (figure 12):

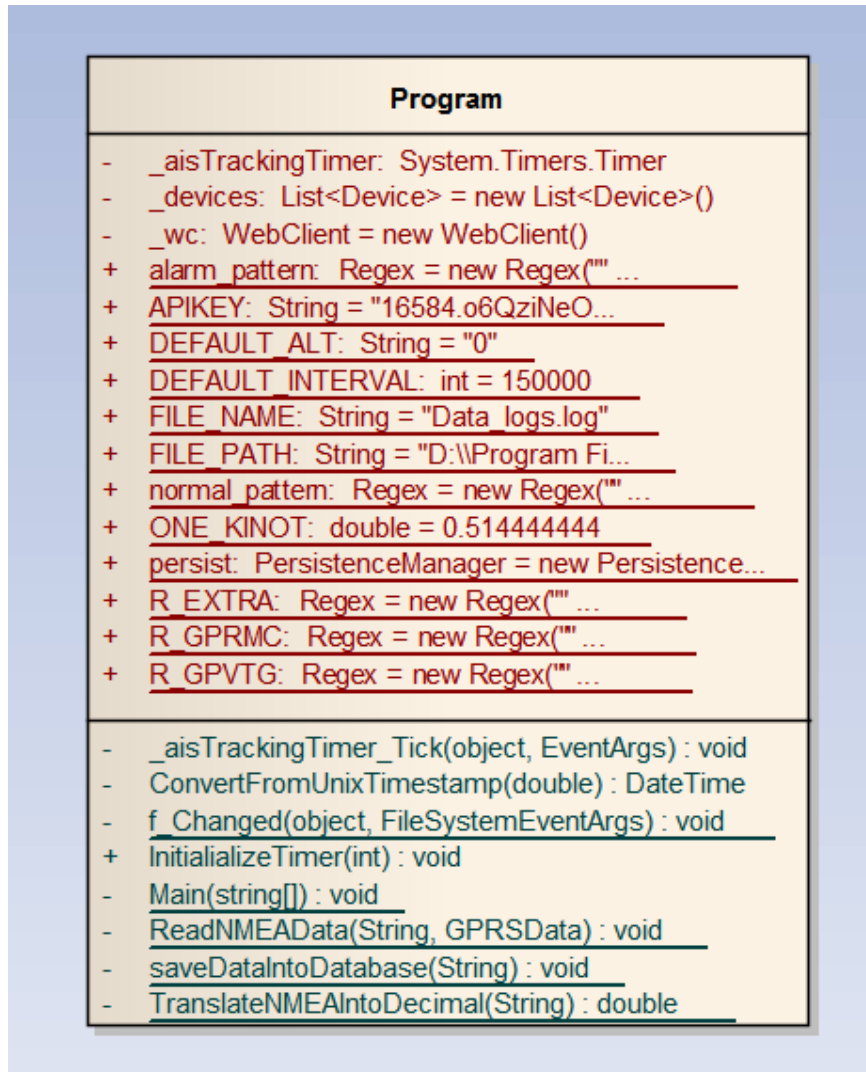
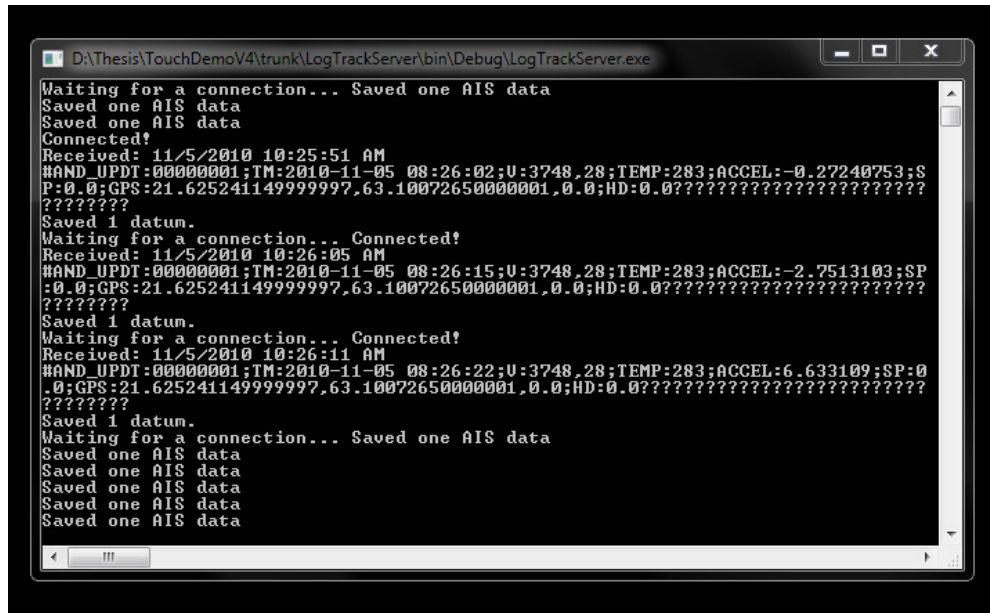


Figure 12. Class diagram of LogTrack server.

The constructor of the LogTrackServer class creates all the components, including a new connection to MySQL database. Then, the window shown in figure 13 will be opened:



```

D:\Thesis\TouchDemoV4\trunk\LogTrackServer\bin\Debug\LogTrackServer.exe
Waiting for a connection... Saved one AIS data
Saved one AIS data
Saved one AIS data
Connected!
Received: 11/5/2010 10:25:51 AM
#AND_UPDT:00000001;TM:2010-11-05 08:26:02;U:3748,28;TEMP:283;ACCEL:-0.27240753;S
P:0.0;GPS:21.625241149999997,63.10072650000001,0.0;HD:0.0????????????????????
????????
Saved 1 datum.
Waiting for a connection... Connected!
Received: 11/5/2010 10:26:05 AM
#AND_UPDT:00000001;TM:2010-11-05 08:26:15;U:3748,28;TEMP:283;ACCEL:-2.7513103;SP
:0.0;GPS:21.625241149999997,63.10072650000001,0.0;HD:0.0????????????????????
????????
Saved 1 datum.
Waiting for a connection... Connected!
Received: 11/5/2010 10:26:11 AM
#AND_UPDT:00000001;TM:2010-11-05 08:26:22;U:3748,28;TEMP:283;ACCEL:6.633109;SP:0
.0;GPS:21.625241149999997,63.10072650000001,0.0;HD:0.0????????????????????
????????
Saved 1 datum.
Waiting for a connection... Saved one AIS data
Saved one AIS data
Saved one AIS data
Saved one AIS data
Saved one AIS data
Saved one AIS data

```

Figure 13. The GUI of LogTrack server.

A new `TcpListener` object which connects to an IP address and a port number will be created. After calling “connect()” method on this `TcpListener` object, new connection will be established and the server is ready to receive data from tracking devices.

There are 3 types of message which the server can recognize. Each of them will be processed in a different way before saved into database.

5.1.1 AIVDM protocol (used in AIS communication)

A detailed explanation of this protocol can be found on appendix B.

This protocol has been decoded by `aprs.fi` API (Application Programming Interface). This API allows application developers to query the `aprs.fi` database from their own applications. In other words, it is possible to retrieve AIS data which have been decoded from this database. The implementation and usage of this API in the LogTrack application are described in next paragraphs.

First of all, it is important to know what kind of data format this API returns. It returns data in either JSON or XML format. In my situation, I use JSON because

it has much smaller overhead when compared to XML. JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language [18]. Figure 14 illustrates one example of JSON as an object:

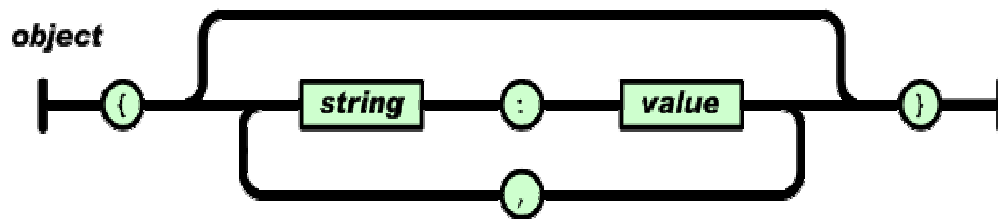


Figure 14. An object form of JSON. [18]

There are many open source library for JSON in .NET environment. This application used [Json.NET library](#). It is a free open source library which makes working with JSON formatted data in .NET simple. Key features include a flexible JSON serializer to for quickly converting .NET classes to JSON and vice versa and LINQ to JSON for reading and writing JSON.

Secondly, after knowing the structure of data, the AISObject class which inherited from CommonObject class was created by using this open source library. Class diagram and its relationship are shown in the following diagram (figure 15):

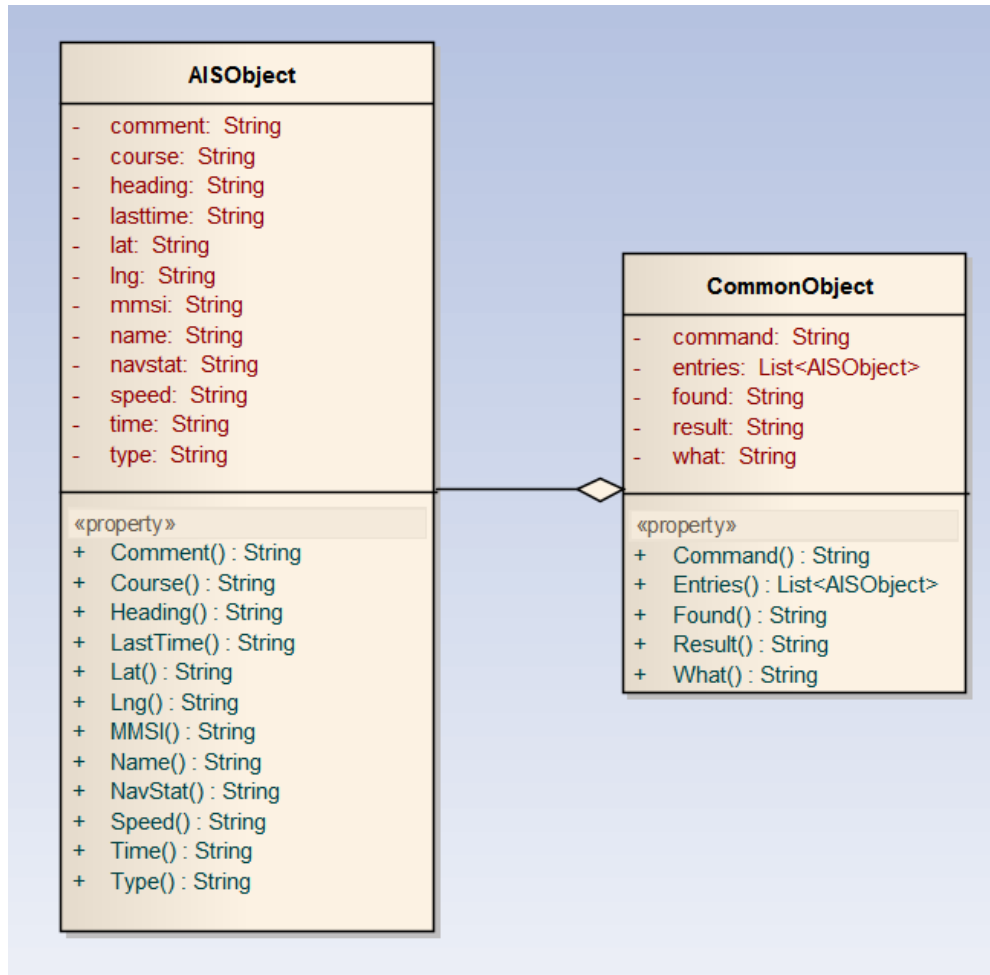


Figure 15. Class diagrams for AISObject and CommonObject class.

Description of common fields in CommonObject class:

- command - the API command which was called
- what - what was being queried
- result - the result of the query, either ok or fail
- found - the number of entries returned

Description of location record fields in AISObject class:

- name - name of station, object, item or vessel

- type - type of target: a for AIS, l for APRS station, i for APRS item, o for APRS object, w for weather station
- time - the time when the target first reported this (current) position (the time of arrival at current coordinates) (Timestamps are returned in the Unix time format)
- lasttime - the time when the target last reported this (current) position
- lat - latitude in decimal degrees, north is positive
- lng - longitude in decimal degrees, east is positive
- course - Course over ground / COG, in degrees
- speed - Speed, in kilometers per hour
- comment - APRS comment or AIS destination and estimated time of arrival
- mmsi - AIS vessel MMSI number
- navstat - AIS navigational status code
- heading – Heading degree

Then, the AIS data of shipments can be achieved in real time by using a WebClient object to download data from the following URL:

```
"http://aprs.fi/api/get?name=" + ship.Code + "&what=loc&apikey=" + APIKEY
+ "&format=json"
```

A sample code to retrieve data from aprs.fi database is illustrated below:

```
byte[] jsonData = _wc.DownloadData(
    "http://aprs.fi/api/get?name=" + ship.Code + "&what=loc&apikey=" + APIKEY + "&format=json");
String strData = Encoding.ASCII.GetString(jsonData);

CommonObject commonData = JsonConvert.DeserializeObject<CommonObject>(strData);
```

5.1.2 AND_UPDT protocol (used in Android phones)

To communicate between central station and Android devices, a new protocol called AND_UPDT was defined. This name comes from “Android Update” phrase. It is often or rarely sent to server based on the level of battery. For example, in high level mode (battery level is more than 80%), data will be sent every 5 or 10 minutes depending on moving distance. But if battery level is lower than 30%, data will be sent every 45 minutes.

Here is a sample of AND_UPDT protocol:

```
#AND_UPDT:00000001;TM:2010-08-20
23:01:00;V:04700,90;TEMP:00247;ACCEL:0000.200;SP:023.5;GPS:123.454000
00,062.00214500,000.00000000;HD:30.5????????????????????
```

It is total of 168 bits and occupying in one sentence.

Fields	Description	Unit
AND_UPDT	Device Id	String, total of 8 bits
TM	Data time	UTC, format “yyyy-mm-dd hh:MM:ss”
V	Battery level and voltage	See below (1)
TEMP	Temperature	See below (2)
ACCEL	Accelerometer	See below (3)
SP	Speed	Km/h, total of 5 bits
GPS	GPS location	See below (4)
HD	Heading degree	Degree

(1) (V:04700,90) the first part (before ‘,’ character) is value of battery level measured in V. It is given in $1*100$ unit; divide by 100 to obtain the real value. The second part (after ‘,’) is percentage of battery level. Maximum value is 100% and minimum value is 0%

(2) (TEMP:00247) The value is measured in degree Celsius and given in $1*10$ unit; divide by 10 to achieve the real value.

(3) (ACCEL:0000.200) Value is measured in m/s^2 . It is the average value and measured in 3 directions: X, Y and Z

(4) (GPS:123.45400000,062.00214500,000.00000000) It measures longitude, latitude and altitude respectively.

5.1.3 GPS/GPRS protocol (used in Chinese device)

Please refer to appendix C to have a detail description of this protocol.

5.2 The Android application


5.2.1 Configuration for Android phones

Before connecting Android phone to Arduino micro-controller, Bluetooth modem needs to be configured. In this LogTrack system, BlueSMiRF was used as a Bluetooth module for the micro-controller. Specifications of this module are shown below [19]:

- FCC Approved Class 1 Bluetooth Radio Modem
- Extremely small radio - 0.15x0.6x1.9"
- Very robust link both in integrity and transmission distance (100m)
- Low power consumption : 30mA connected, < 10mA sniff mode

- Operate in harsh RF environments like WiFi, 802.11g, and Zigbee
- Encrypted connection
- Frequency: 2.4~2.524 GHz
- Operating Voltage: 3.3V-6V
- Serial communications: 2400-115200bps
- Operating Temperature: -40 ~ +70C
- Built-in antenna

At first, the baud rate of Bluetooth module has to be changed to 57600 baud. Normally, the default value will be 115200 baud, but only ArduinoBT board works with 115200 baud. Instructions on how to set baud rate for this Bluetooth module are described below: [20]

- Power up the Bluetooth device.
- Use Bluetooth to connect your device with your laptop:
 - Right click on Bluetooth icon  and select “Add a device”.
 - We are using Bluetooth BlueSMiRF module here, so select FireFly
 - Click “Next” and enter pair code. The default one is 1234.
 - Then, open Bluetooth devices, right click on FireFly and select Properties. Select “Services” tab, the COM port for this device will be displayed.
- Use a terminal emulator to connect to this COM port. In this project, Ter-aTerm which is a free software terminal emulator (communication program) for MS-Windows has been used.

- Remember that to enter the command mode remotely over Bluetooth , we need to make connection with the device within “configuration time” after power up. This can be modified; the default “configuration time” is 60s.
- Type “\$\$\$” to enter command mode. You should see “CMD” returned. When finishing configuring, make sure that the device is reset or type “---” command, which will exit configuration mode and allow data pass normally.
- To make sure that you are in command mode, type “D” or “E” and see the result.
- If everything is correct, you should see something as the below picture:

```
$$$CMD
D
***Settings***
BTAd=000666050382
BTName=FireFly-0382
Baudrt=57.6
Parity=None
Mode =Slav
Authen=0
Encryp=0
PinCod=1234
Bonded=0
Rem=NONE SET
E
***ADVANCED Settings***
SrvName= SPP
SrvClass=0000
DevClass=1F00
InqWindw=0100
PagWindw=0100
CfgTimer=60
StatuStr=NULL
```

- Next, type “SU,<rate>” to change the baud rate. For example, type “SU,57600” to set the Bluetooth module to 57600 baud. You can type “D” to check if the baud rate changes or not.
- Finally, type “---” to exit the configuration mode. More information about AT command set of this Bluetooth device can be found on [20].

Then, this Bluetooth device needs to be paired with the Android phone. The default PIN code is 1234.

Now you can try to connect to this Bluetooth module. If everything is correct, “Connect” LED on Bluetooth board will turn on and the Android phone will be ready to receive data from micro-controller.

In addition, the Android phone also has to be configured. The Android application was written on Android 2.1 operating system, so the software should be installed on an Android 2.1 phone which supports GPS sensor and accelerometer sensor. The configuration screen is shown in figure 16.



Figure 16. Android emulator’s screen.

Description of fields in this screen:

- Device Id: the identification of device, which is unique for each device
- IP address: IP address of server
- Port number: port number which is used in server (default is 9090)
- Interval time: the android application will send data to server periodically
- Max Temp: maximum value of temperature (in degree Celsius)
- Min Temp: minimum value of temperature (in degree Celsius)
- Max Accel: maximum value of acceleration (in m/s^2)
- Min Accel: minimum value of acceleration (in m/s^2)
- Max Humi: maximum value of humidity (in %)
- Min Humi: minimum value of humidity (in %)

5.2.2 Implementation of the Android application

The Android platform uses a hierarchy of View and ViewGroup nodes to define user interface of an Activity, as shown in the figure 17:

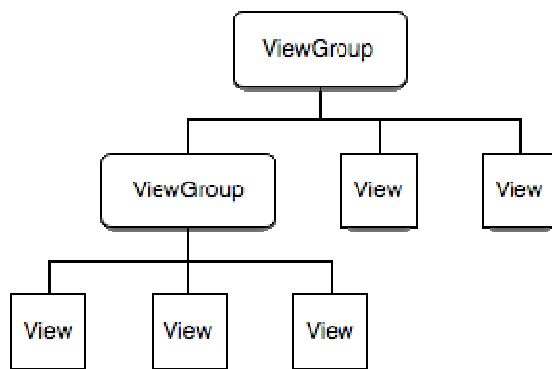


Figure 17. View and ViewGroup hierarchy in Android platform [21].

The Activity class must call the `setContentView()` method in `onCreate()` method in order to attach the view hierarchy tree to the screen for rendering:

```
“setContentView(R.layout.main)”
```

Each Android application has its own layout. There are many pre-defined view groups (called layouts) offered by Android, including `LinearLayout`, `RelativeLayout`, `TableLayout`, `GridLayout` and others. Our own layout can be defined in a variety of ways. The most common way to define a layout is using an XML layout file to express the view hierarchy as it has been done in this Android application with `TableLayout`.

After creating the UI (user interface) for the Android application, `OnClickListener` events are attached to “Start” and “Stop” button. When these events are triggered by user’s action, they will call `startService()` and `stopService()` methods respectively.

In order to send GPS data to the central station periodically, `Timer` class was used. This `Timer` class was used to schedule jobs for execution in a background process. In other words, a single thread was used for the scheduling and this thread has the option of being a daemon thread. The `Timer` and its association thread can be stopped or terminated by calling `cancel()` method. Tasks for repeated fixed-delay execution after a specific delay can be scheduled by calling `schedule (TimerTask task, long delay, long period)` method.

The diagram below illustrates the class diagram for `MyService` class (figure 18):

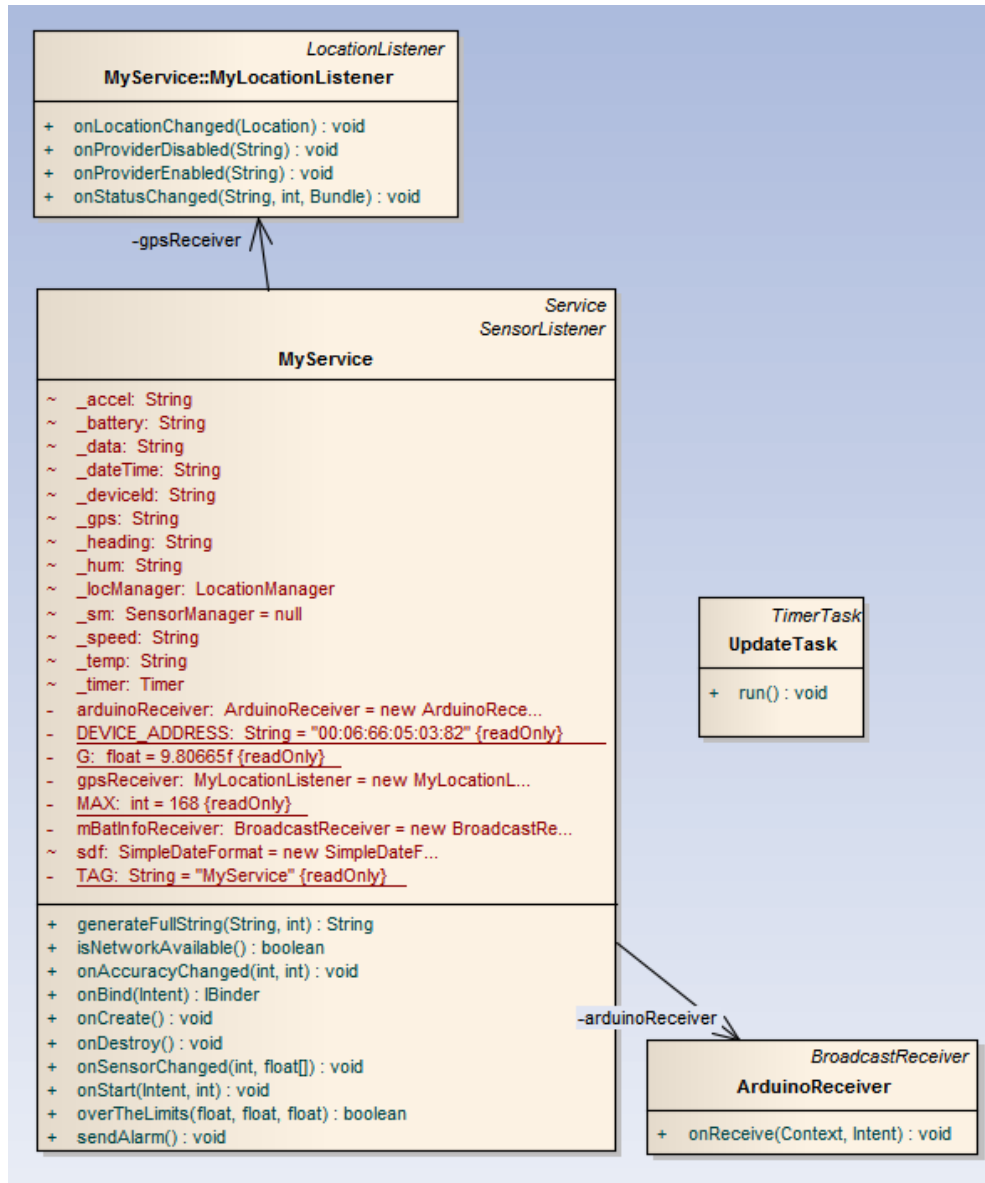


Figure 18. Class diagram of MyService class.

5.2.3 Implementation of Arduino application

The BlueSMiRF module attached with Arduino provides Bluetooth communication with computers, phones and other Bluetooth devices. This module is connected to Arduino via serial (shared with the RX and TX pins on the board). It is configured for 57600 baud communication. The output of temperature sensor is connected to analog input pin 5, 4 of micro-controller and the output of humidity sen-

is connected to analog input pin 3. The diagram below illustrates the schematic of this micro-controller (figure 19):

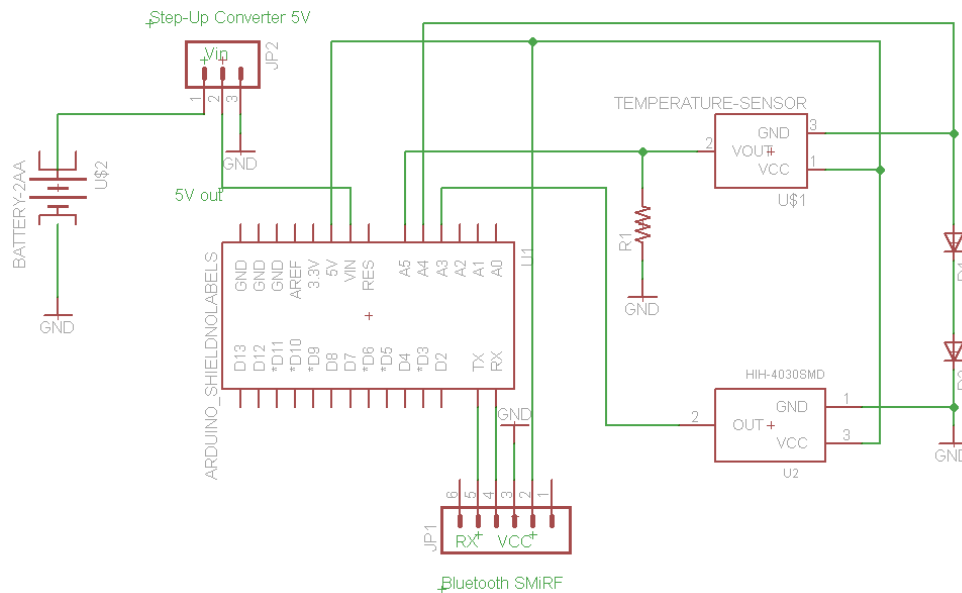


Figure 19. Schematic of Arduino micro-controller.

In default, the Arduino will send data including temperature and humidity to Android phone every 10 seconds. In order to receive this information, MyService class must register a BroadcastReceiver to be run in the main activity thread. The receiver will be called with any broadcast Intent that matches filter, in the main application thread. Following is the code which is used to register ArduinoReceiver class:

```
// in order to receive broadcasted intents,
// we need to register our receiver
registerReceiver(arduinoReceiver,
    new IntentFilter(AmarinoIntent.ACTION_RECEIVED));
```

5.3 The client

5.3.1 Configuration for the client

This application is running on Windows PC with full internet connection. Google earth Plug-in must be installed on this computer. This application is more compatible with Windows 7, yet it can run on any kinds of Windows operating system. The application was written in WPF language, so .NET framework 3.5 or later need to be installed. The source code can be edited by Visual Studio or Visual C# Express Edition.

MySQL database has been used in this system. This is a popular open source database and is supported by a large community and enthusiasts. The port number (3306) need to be reserved for this database. Server and client are connected to database by NHibernate. NHibernate is a port of Hibernate Core for Java to the .NET Framework. It handles persisting plain .NET objects to and from an underlying relational database. Given an XML description of your entities and relationships, NHibernate automatically generates SQL for loading and storing the objects. Optionally, you can describe your mapping metadata with attributes in your source code [22]. The purpose and advantages of using this framework will be explained in detail in next section. The configuration file needs to be changed before connecting to MySQL database, as shown in figure 20:

```
<?xml version="1.0" encoding="utf-8" ?>
<hibernate-configuration xmlns="urn:hibernate-configuration-2.2" >
  <session-factory name="NHibernate.Test">
    <property name="connection.driver_class">NHibernate.Driver.MySqlDataDriver</property>
    <property name="connection.connection_string">
      Database=touchdemo;Data Source=oly.mine.nu;User Id=touchdemo;Password=touchdemo
    </property>
    <property name="dialect">NHibernate.Dialect.MySQLDialect</property>
    <property name="proxyfactory.factory_class">NHibernate.ByteCode.Castle.ProxyFactoryFactory, NHibernate.ByteCode.Castle</property>
  </session-factory>
</hibernate-configuration>
```

Figure 20. Hibernate configuration file (hibernate.config.xml).

The structure of this configuration file is described as follows:

- A XML header

- A hibernate configuration namespace declaration
- A driver class for database: we are using MySQL database
- A connection string:
 - Database: name of database which we used in MySQL server
 - Data Source: DNS of server or server name or IP address
 - User Id: username which was used to connect to the above database
 - Password: password of the above user
- A dialect of database
- A factory class which is used in NHibernate

5.3.2 General layout of this client

The diagram below (figure 21) illustrates the general layout of this client:

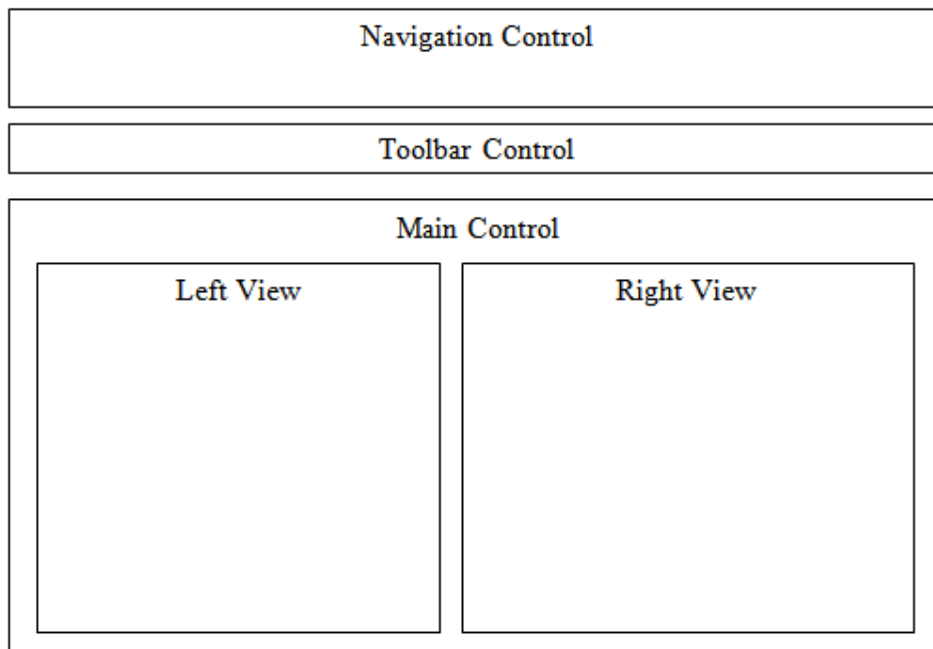


Figure 21. Layout of the application .

The navigation control is on top of the application. It composes of 4 buttons which have indicator glass button style. Each of them represents one method for finding a specific project. In main control, list of projects is shown on the left side and detail information for these projects is shown on the right side. These components are displayed in a composite way so that all the components are bound to each other by using binding technology in WPF. For instance, when clicking on a project on the left side, the information about this project will be displayed automatically on the right side (figure 23). This will improve the application's efficiency. By simply clicking on "On Earth" button, Google Earth view is activated as shown in figure 22.

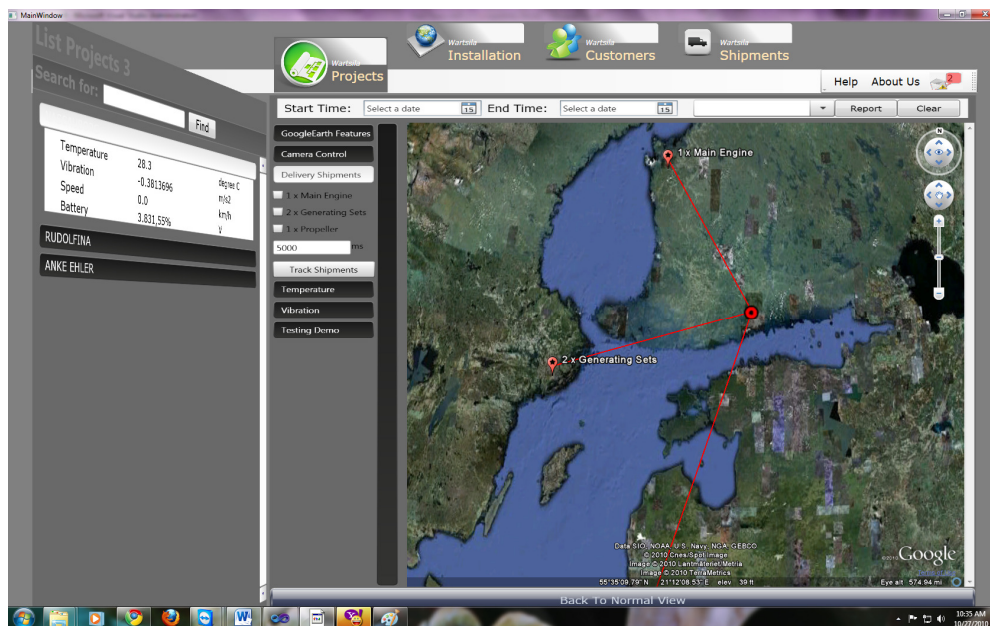


Figure 22. Google Earth view of the application.

In this view, the visual information of project and its shipments will be displayed on Google Earth in real time. Shipments or packages can be tracked and the environmental data recorded in the shipment can be observed in real time in this view.

The visual display of normal view is shown in figure 23.

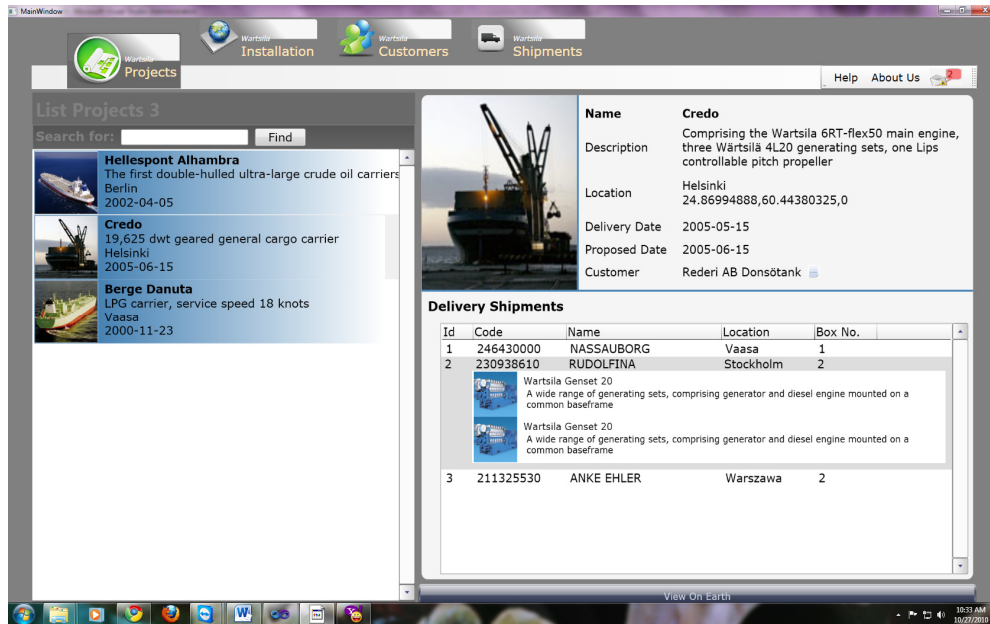


Figure 23. Normal view of the application.

5.3.3 Description about components and searching methods

The diagram below (figure 24) illustrates the class hierarchy of MainWindow class, which is the main component of this application.

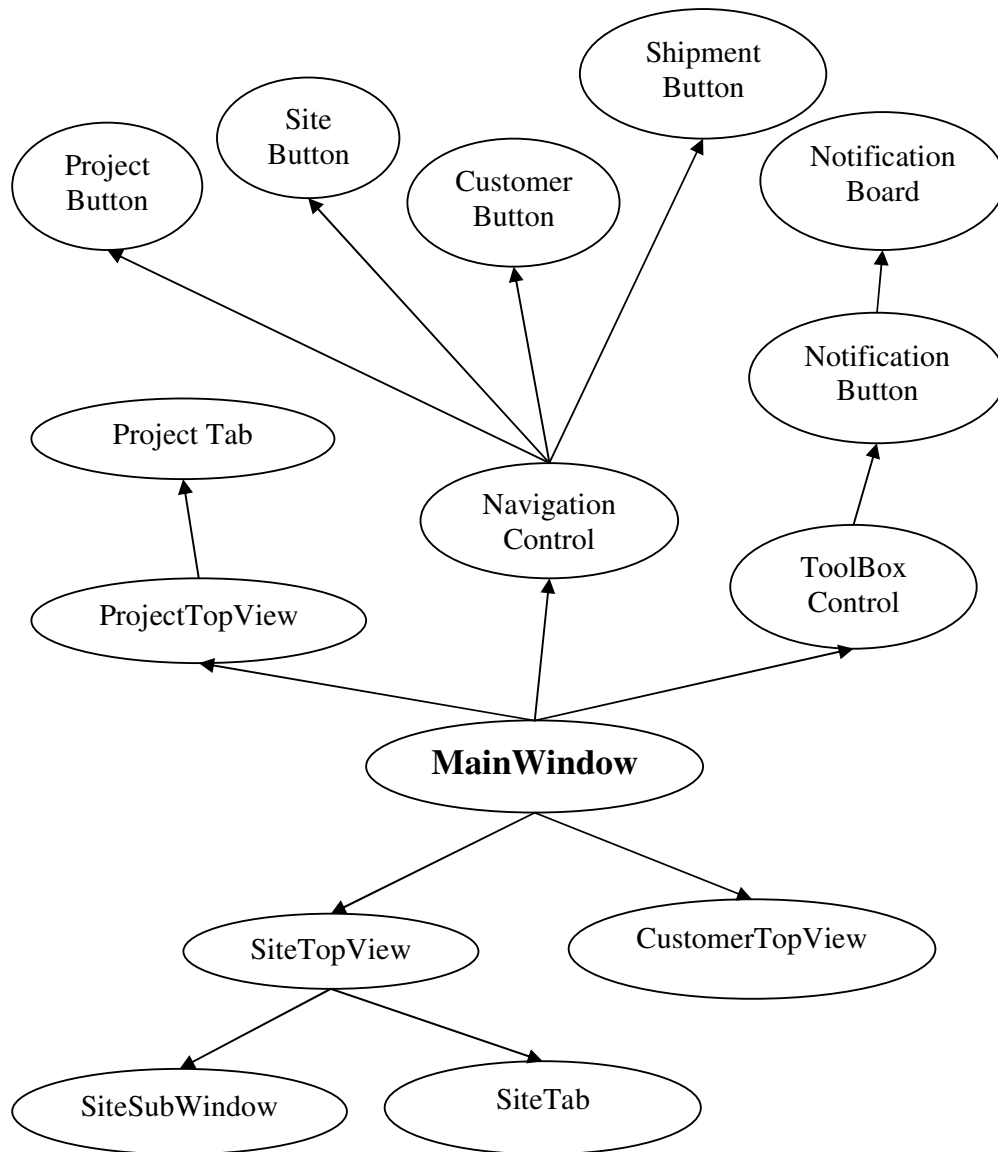


Figure 24. Class hierarchy of components inside the application.

As previously stated, there are 4 approaches to find a specific project. First of all, users can search for a specific project by its name (figure 25).

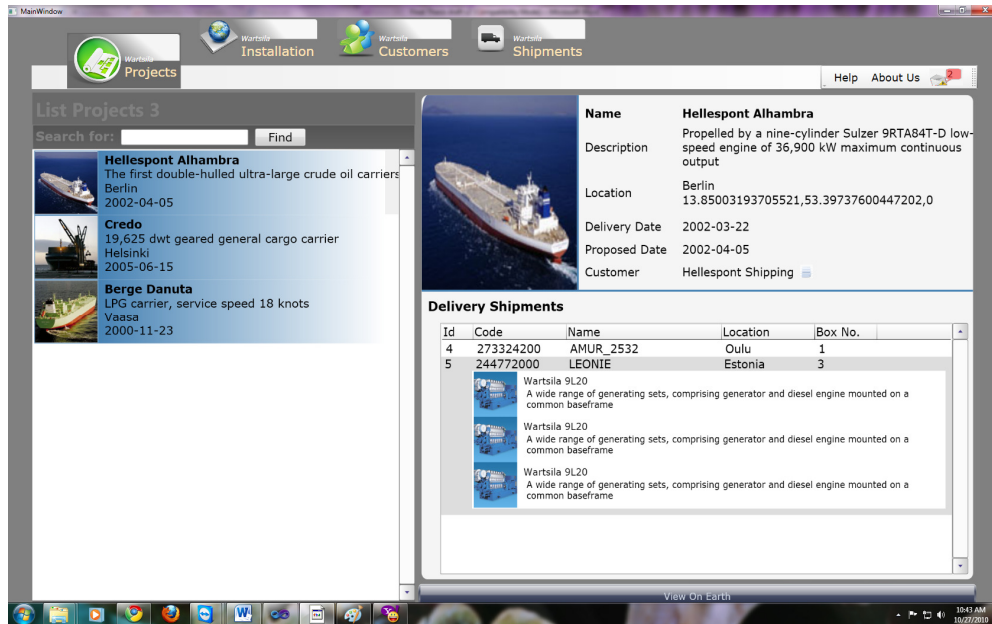


Figure 25. Project view.

Secondly, a project can also be searched for by its location. This method is written in SiteTopView class and SiteSubWindow class. In this view, list of all locations (from SITES table) are displayed randomly on the screen. To choose one location, user simply double-clicks on it (figure 26).

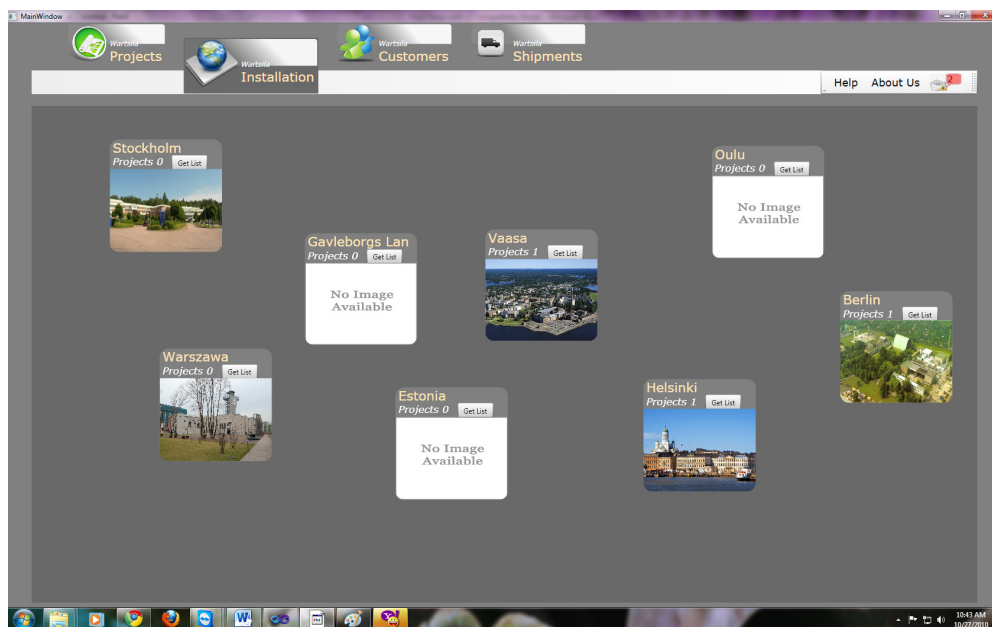


Figure 26. Site view.

Next, the project can be searched by its customer. A list of customers (from CUSTOMERS table) with first name, last name, location, phone number, email etc is displayed in this customer view (figure 27). Choosing one customer will give us a list of projects and their information. This is a composite component and written in CustomerTopView class.

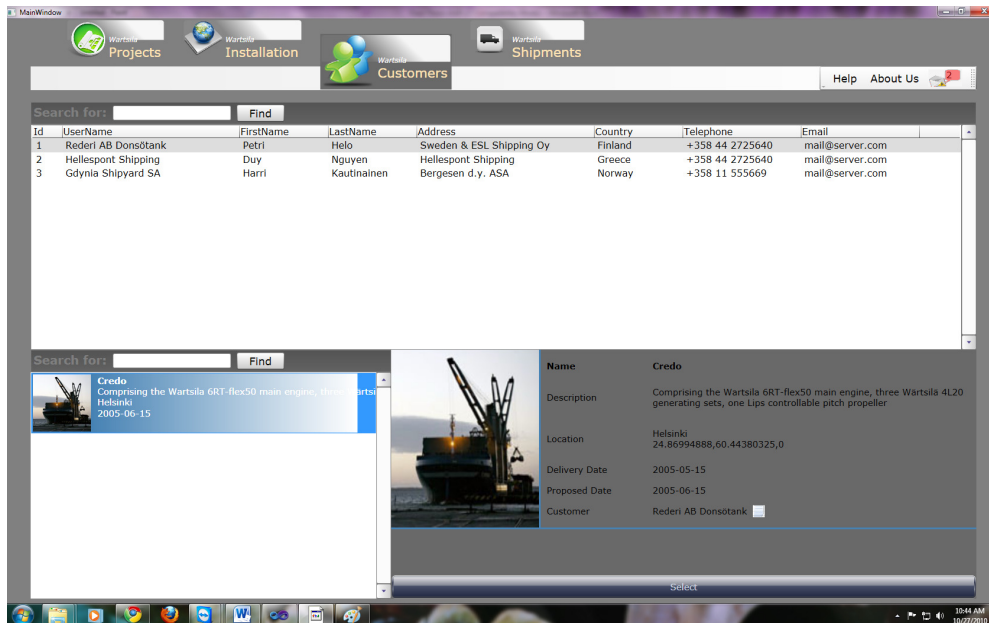


Figure 27. Customer view.

Finally, we can search for a specific project by its shipments. The Google Earth plug-in was implemented and embedded into this shipment view to improve the application's interactivity (figure 28). A list of all shipments (from SHIPMENTS table) is shown on the left side and a web browser component which stores the Google Earth plug-in is shown on the right side. Double-clicking on one shipment will navigate us to its location on Google Earth. More information about the implementation of these views and components can be found on the Annex A.

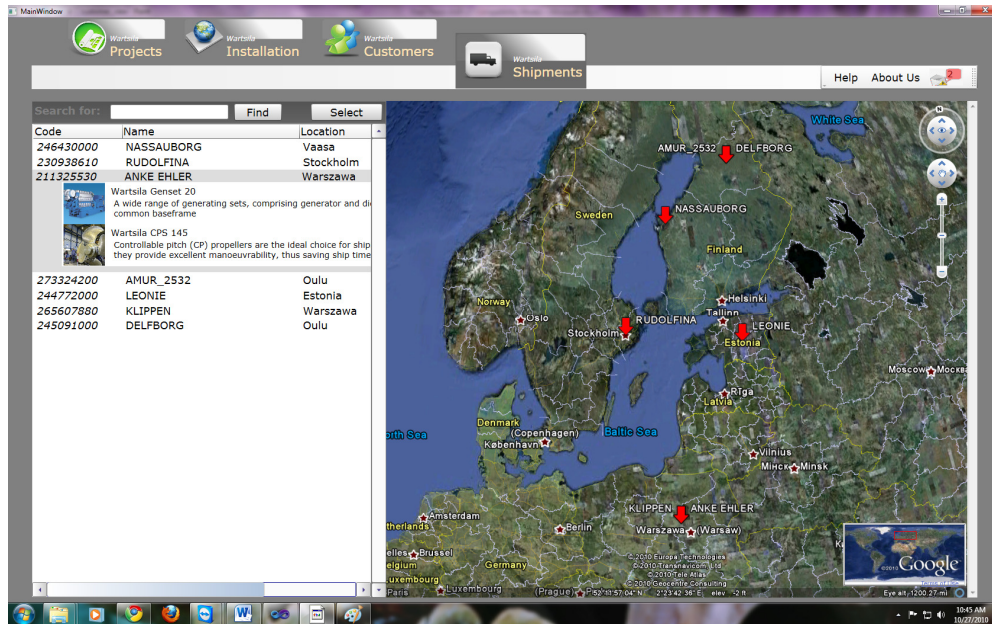


Figure 28. Shipment view.

The following diagram illustrates the use case diagram for this application (figure 29):

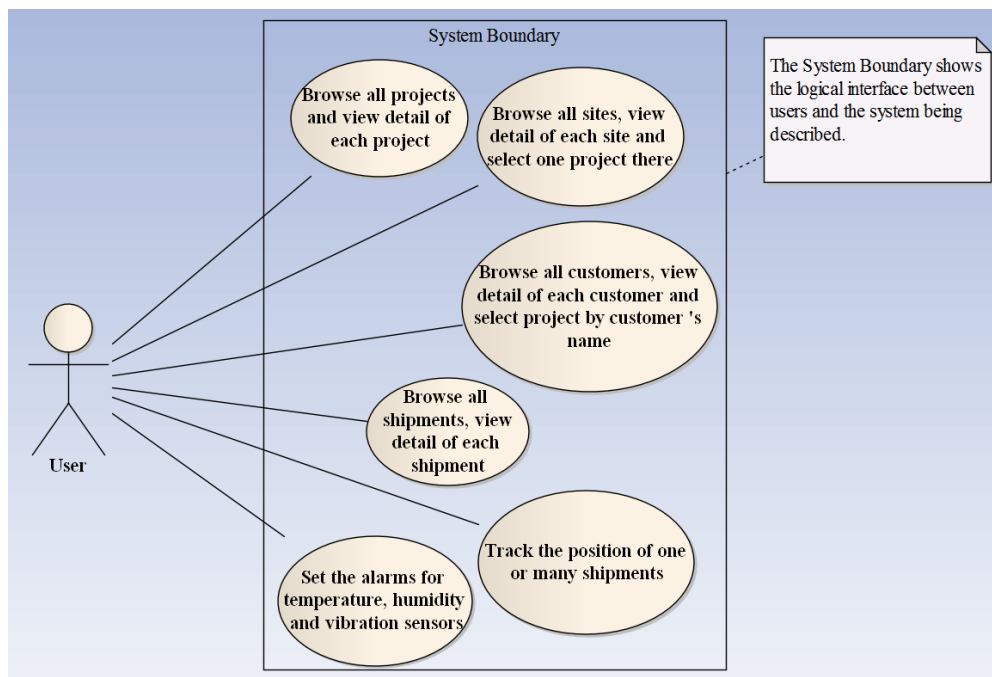


Figure 29. Use case diagram for this application.

5.3.4 Communication between Google Earth and .NET framework

KML class is used to display geographic data in Google Earth. More information about this KML class and how to use it can be found on appendix D.

The class diagram of this KML class is shown in figure 30.

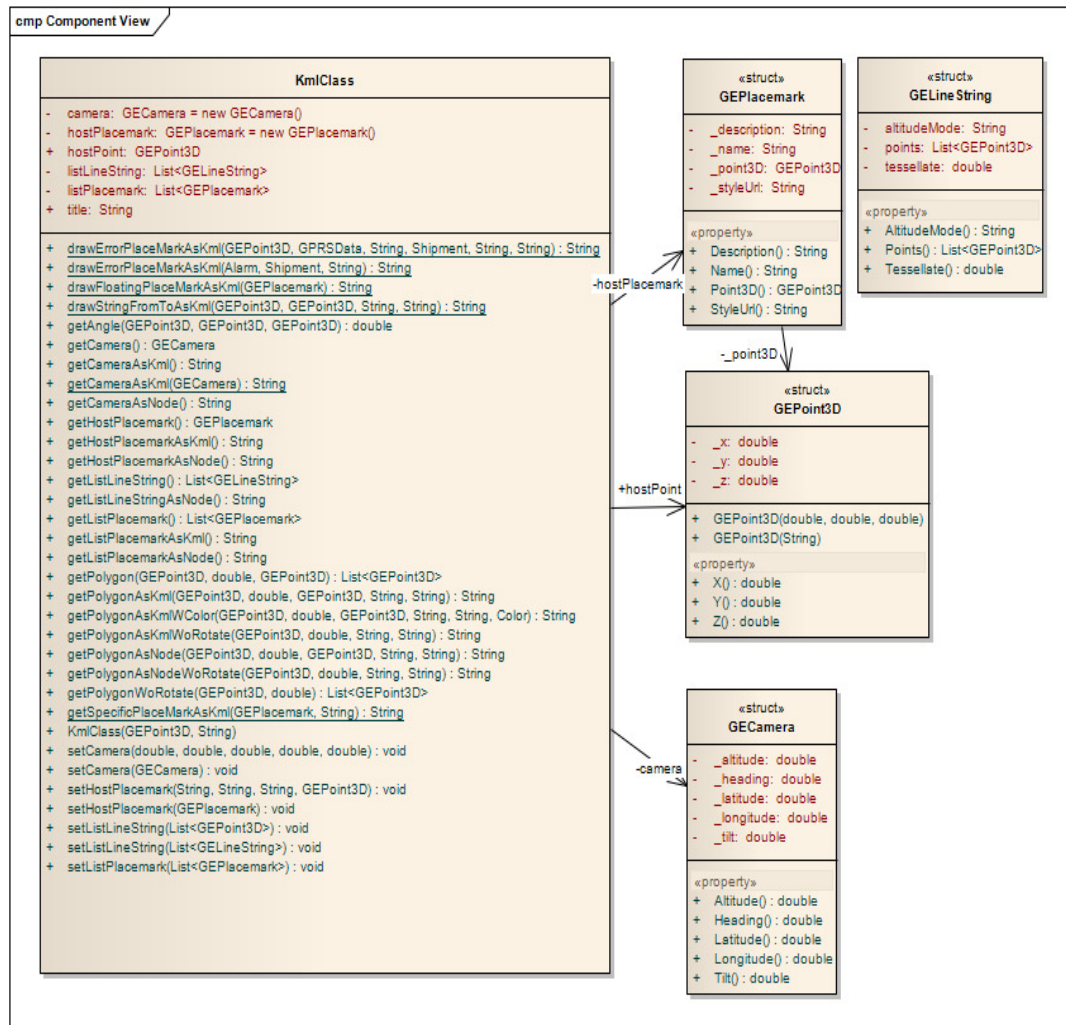


Figure 30. Class diagram of this KML class .

6 PROBLEMS ENCOUNTERED

This chapter will describe the problems this project encounters and the approaches used to tackle these problems.

6.1 Battery life problem

It is obvious that most of GPS/GPRS devices have battery problem because they drain energy very fast. This will cause serious problems in our tracking purpose due to the fact that we need to trace them in such a long period of time, approximately 2 weeks. So, what will happen when the package is in the middle of the sea and our device is out of battery? Of course, we can get the GPS location of shipments through AIS systems. But when it is on land, we will lose its signal. As a result, we cannot manage and maintain them. This can lead to some severe cases which are in threat of cancellation because of delayed packages or lost packages.

There are some possible solutions which can be used to extend the battery life for GPS devices.

Firstly, an external battery can be used. Normally, an internal battery whose capacity is 1350mAh is integrated into Android phones. For the typical usage (42mA average), it can last up to 32 hours. We can integrate this internal battery with an external one to extend the battery life for our Android phones. The illustration of external battery which was used in this application is shown in figure 31:



Figure 31. Lead-acid battery 6V 12Ah.

With this 6V 12Ah battery, it can extend the battery life up to 12 days (for the typical usage, 42mA average). This sounds very promising because at least it meets our minimum requirements.

Secondly, we can extend the battery life for these tracking devices by optimizing the programming code.

Firstly, as is has been mentioned in chapter 4.2, these Android phone use cellular network to send data to central station. It would cost more energy when the device lost internet connection and tried to find an available network. Therefore, it would be better if GPS data are only sent when 3G or EDGE is connected:

```

// Skip if no connection or background data is disable.
// Only update if 3G or EDGE is connected
public boolean isNetworkAvailable() {

    ConnectivityManager mConnectivity =
        (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);

    NetworkInfo info = mConnectivity.getActiveNetworkInfo();
    if (info == null || !mConnectivity.getBackgroundDataSetting()) {
        return false;
    }

    int netType = info.getType();
    int netSubType = info.getSubtype();

    if (netType == ConnectivityManager.TYPE_WIFI) {
        return false;
    } else if (netType == ConnectivityManager.TYPE_MOBILE
        && (netSubType == TelephonyManager.NETWORK_TYPE_GPRS
            || netSubType == TelephonyManager.NETWORK_TYPE_UMTS)) {
        return info.isConnected();
    } else {
        return false;
    }
}

```

Another issue which must be taken into account is the usage of GPS sensor. It takes almost 25 seconds to retrieve a location by using GPS sensor, but it takes only 2 seconds when using network:

- GPS: 25 seconds * 140mA = 1mAh
- Network: 2 seconds * 180mA = 0.1mAh

As a consequence, using coarse network location is much cheaper. In addition location updates can continue after onPause(), so it must be unregistered when not in use, especially when the device is at sea and cellular network is not available. [23]

7 DISCUSSION AND CONCLUSION

At the end of each project, it is crucial to carry out an assessment of what has happened and what has been done during the last four months. Many things have occurred during this period of time. This chapter will therefore evaluate both achievements and limitations of the system.

First of all, understanding the system's restrictions is important. As mentioned in chapter 1, due to shortage of time and resources, this project focuses mainly on the design and implementation of the basic working system which can be built further. So, the first limitation of this LogTrack system is about tracking devices. It is limited to Android phones and the Chinese device which I bought from the market. In the future, we should build our own tracking device which has the same features with the previous ones.

Furthermore, the whole system has not been tested in the real case. The real case mentioned here is mounting the tracking device on the engine which is really shipped from Wärtsilä to its customer. This testing procedure must be done in the next stage of the project, but not in this stage because it is very complicated. For example, informing the customers and the customs about these tracking devices should be done before sending them with the pilot. It is also necessary to prepare a cost analysis for the shipment. In my case, to test the GPS device, I had to put the tracking device on a car and drove it around the city.

Comparing my starting point and the outcome now, lots of things have certainly changed. New tracking devices have been added to the system. New components have been created and made to work. The existing system has been extended with many new functionalities and features.

However, there are still several problems/drawbacks in the system. Battery life is one. It needs to last at least 2 weeks based on the requirement from Wärtsilä people but I did not have enough time to make a testing done although I have proposed some solutions in chapter 6. The main reason for this is partly due to the fact that I needed to make a testing plan for the tracking devices and it took a lot

of time to wait for the response from Wärtsilä. I contacted them in the beginning of the project to get information about requirement and specifications of the project and it took almost 4 weeks to receive their response. Another reason is that I needed to buy some electronic components from the market during the project and it was very time-consuming to ship these components.

But the good point that the whole system is working properly. The response time from the client and the Android phone have been improved a great deal compared to the first time when the system was tested.

In conclusion, a structure of the basic architecture for the tracking and tracing network along with the generic technology behind the tracking and tracing network is defined and discussed. Regarding the implementation structure of the tracking technology, future research will continue to describe the practical business cases in detail and assess the sophistication of the technology applied [4].

REFERENCES

- 1 About Wärtsilä. Available on the Internet: <URL: <http://www.Wärtsilä.com/en/aboutus,,,,,htm>>.
- 2 EAN International, 2001, "Global Trade Item Numbers (GTIN), Application Guideline," EAN International, Online at: <http://www.ean-int.org/>.
- 3 ISO/IEC, 2000, "Automatic Identification – Radio Frequency Identification for Item Management – Working Draft," International Organization for Standardization, ISO/IEC 18000-1-v8.
- 4 Real-time Tracking and Tracing System: Potentials for the Logistics Network, AHM Shamsuzzoha and Petri T Helo, 1-2.
- 5 Harris, E., 1999, "Project Risk Assessment: A European Field Study," British Accounting Review, 31, 347-371.
- 6 Willesdorf, R.G., 1991, "Adding Value Through Logistics Management," International Journal of Physical Distribution & Logistics Management, 21(4), 6-8.
- 7 Kärkkäinen, M., Holmström, J., Främpling, K., and Artto, K., 2003, "Intelligent Products: A Step towards A More Effective Project Delivery Chain," Computers in Industry, 50(2), 141-51.
- 8 Helo, P., and Szekely, B., 2005, "Logistics Information Systems: An Analysis of Software Solutions for Supply Chain Co-Ordination," Industrial Management and Data Systems, 105(1), 5-18.
- 9 Helo, P., 2006, "Agile Production Management – An Analysis of Capacity Decisions and Order-Fulfilment Time," International Journal of Agile Management Systems, 1(1), 2-10.
- 10 Marine Traffic AIS Vessel Tracking. Available on the Internet: <URL: <http://www.marinetraffic.org/>>.

- 11 Windows Presentation Foundation. Available on the Internet: <URL: http://en.wikipedia.org/wiki/Windows_Presentation_Foundation>.
- 12 Spiral model. Available on the Internet: <URL: http://en.wikipedia.org/wiki/Spiral_model>.
- 13 Automatic Identification System (AIS). Available on the Internet: <URL: http://en.wikipedia.org/wiki/Automatic_Identification_System>.
- 14 APRS, Automatic Package Reporting System. Available on the Internet: <URL: <http://www.aprs.org/>>.
- 15 Very high frequency. Available on the Internet: <URL: http://en.wikipedia.org/wiki/Very_high_frequency>.
- 16 Android (operating system). Available on the Internet: <URL: [http://en.wikipedia.org/wiki/Android_\(operating_system\)](http://en.wikipedia.org/wiki/Android_(operating_system))>.
- 17 Arduino. Available on the Internet: <URL: <http://www.arduino.cc/>>.
- 18 JSON. Available on the Internet: <URL: <http://www.json.org/>>.
- 19 Roving Network Bluetooth module datasheet. Available on the Internet: <URL: <http://www.rovingnetworks.com/documents/RN-41.pdf>>.
- 20 RN-41 AT Command Set. Available on the Internet: <URL: <http://www.sparkfun.com/datasheets/Wireless/Bluetooth/rn-bluetooth-um.pdf>>.
- 21 Android developers. Available on the Internet: <URL: <http://developer.android.com/guide/topics/ui/index.html>>.
- 22 NHibernate for .NET. Available on the Internet: <URL: <http://community.jboss.org/wiki/NHibernateforNET>>.
- 23 Google I/O – Coding for Life – Battery Life, That Is. Available on the Internet: <URL: <http://www.google.com/events/io/2009/sessions/CodingLifeBatteryLife.html>>

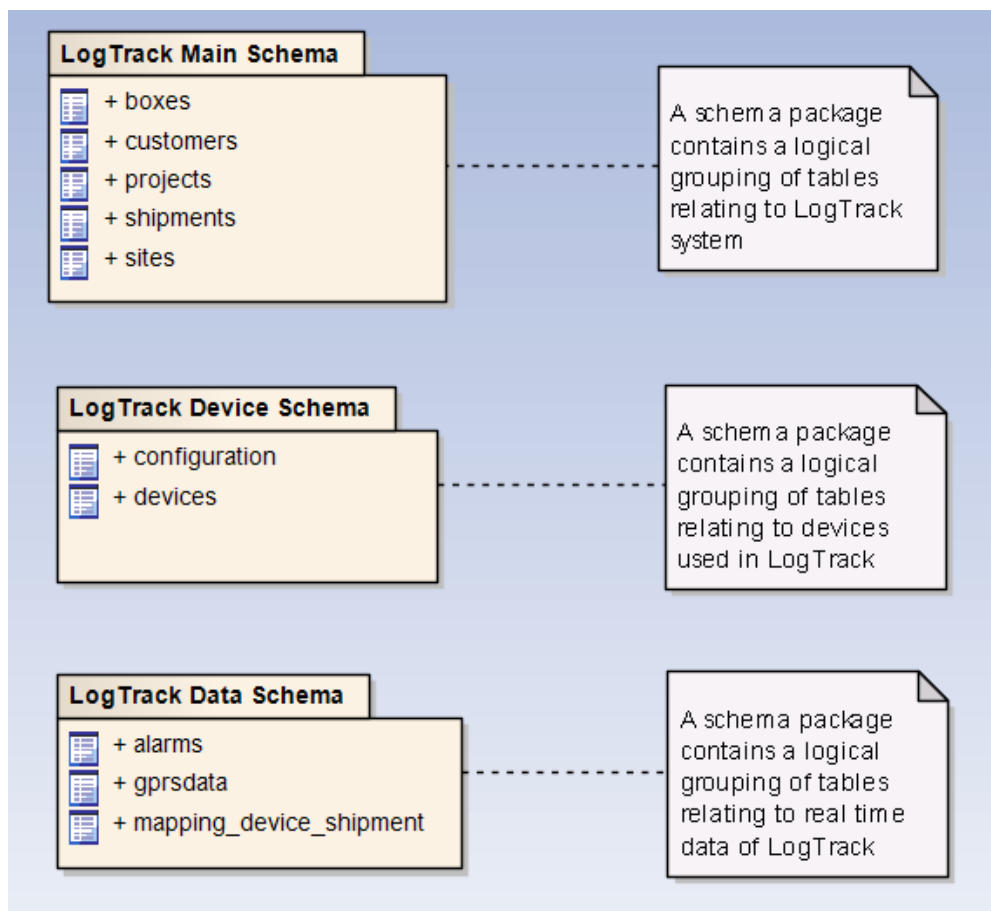
- 24 NMEA data. Available on the Internet: <URL:
<http://www.gpsinformation.org/dale/nmea.htm>>.
- 25 KML | OGC. Available on the Internet: <URL:
<http://www.opengeospatial.org/standards/kml/>>.
- 26 KML reference. Available on the Internet: <URL:
<http://code.google.com/apis/kml/documentation/kmlreference.html>>.
- 27 SparkFun Electronics. Available on the Internet: <URL:
<http://www.sparkfun.com/>>.
- 28 Wartsila LogTrack PowerPoint, University of Vaasa, p2.
- 29 The operation beginning of AIS. Available on the Internet: <URL:
http://www6.kaiho.mlit.go.jp/kanmon/eng/mg_2.htm>.
- 30 Wärtsilä LogTrack Project documentation PowerPoint, University of Vaasa,
p4.

APPENDICES

Appendix A – LogTrack Database

The database is a very essential part of the application due to the fact that all data that will be used throughout the application are stored here. In this section, I will describe structure of the LogTrack database and its tables.

The database schema is shown in the following figure:



A.1 LogTrack Main Schema

The diagram below (figure 32) illustrates tables which are available in this schema:

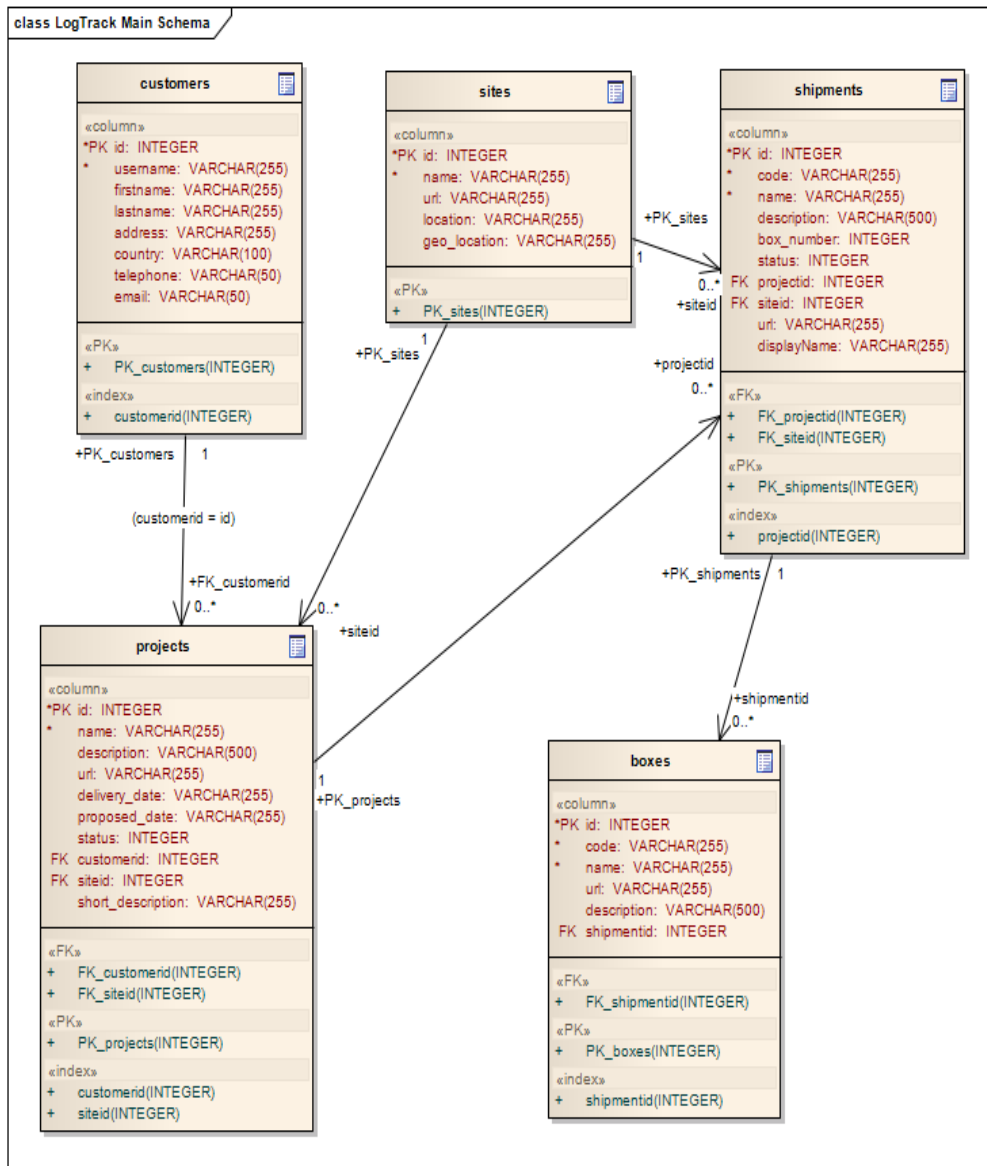


Figure 32. Table diagram for LogTrack main schema.

This schema contains tables which are used to store information about projects and their related information. This information is integrated with SAP system of Wärtsilä.

Description about tables:

- Table **PROJECTS**: store information about projects including name, description, delivery date, proposed date and status.

- Table SITES: store information about installations where projects or shipments locate, including name, location, coordinate of location.
- Table SHIPMENTS: store information about shipments or pilots including name, MMSI, description, amount of boxes and status.
- Table BOXES: store information about packages/boxes which are shipped by shipments. Normally, they are main engines, generating sets, propeller etc.
- Table CUSTOMERS: store information about customers of Wärtsilä, including name, address, country, email and phone number.

A.2 LogTrack Device Schema

Table diagram for this schema is described in figure 33

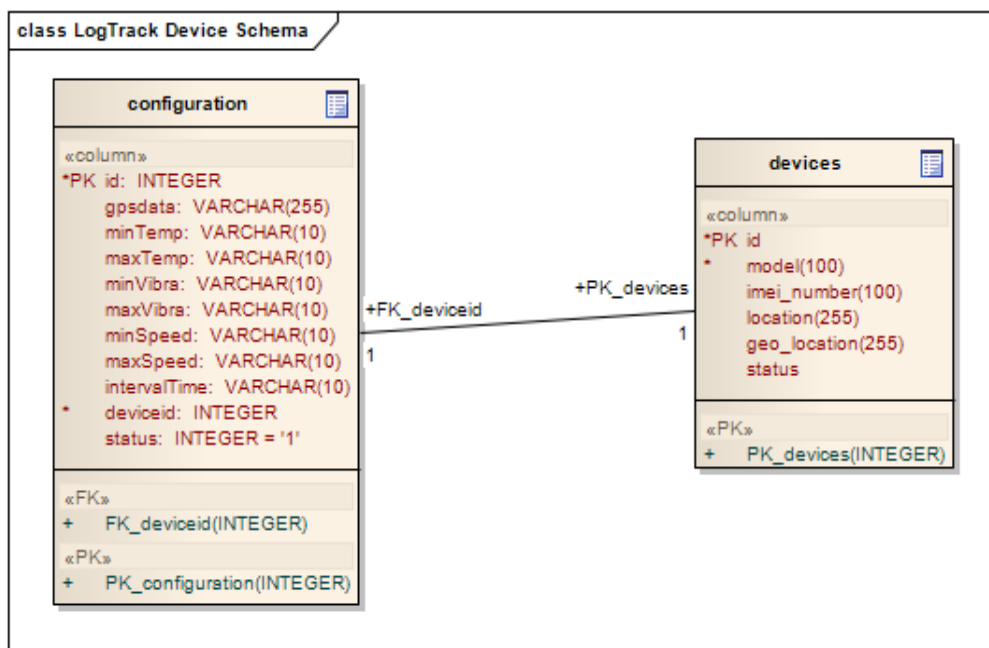


Figure 33. Table diagram for this device schema.

This schema contains tables which are used for tracking devices.

Description of tables:

- Table DEVICES: store information about GPS devices including id, model number, IMEI number, location and status.

- Table CONFIGURATION: store information which is used to configure tracking devices, including GPS data, minimum temperature, maximum temperature, minimum vibration, maximum vibration, minimum speed, maximum speed, interval time and status.

A.3 LogTrack Data Schema

The following diagram (figure 34) illustrates tables in LogTrack data schema

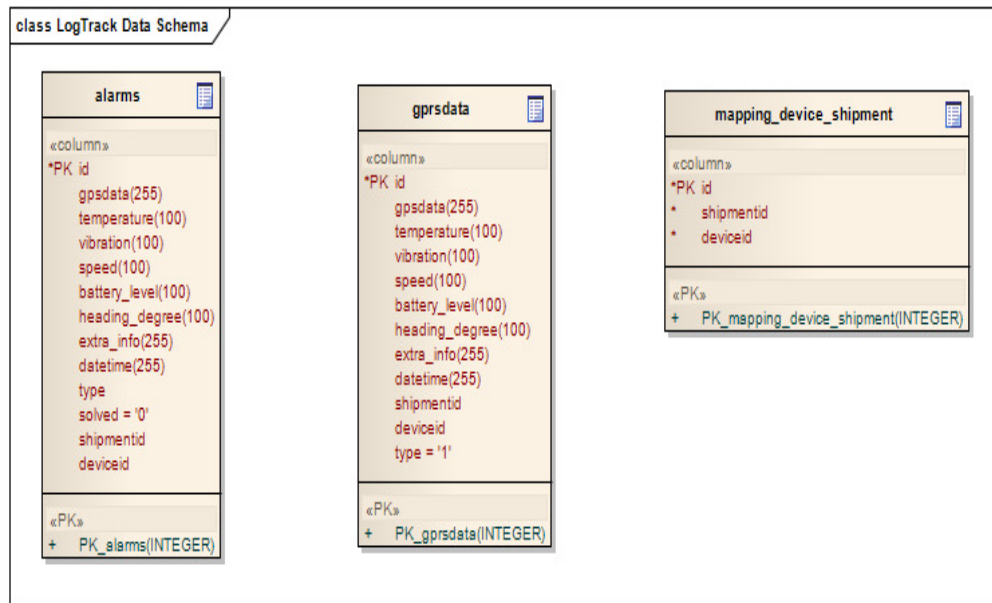


Figure 34. Table diagram for LogTrack data schema.

This schema contains tables which are used to store data from tracking devices.

Description of tables:

- Table GPRSDATA: store information which is received from tracking devices, including GPS data, temperature, vibration, speed, batter level, heading degree, extra information, time, and data type.
- Table ALARMS: store error data or warning information from tracking devices
- Table MAPPING_DEVICE_SHIPMENT: this table is used to map devices with shipments.

Appendix B – AIVDM protocol

There are 26 different types of messages which are transmitted by AIS transponders. These messages specify individual product types.

AIS message types are primarily classified into 2 classes: class A and class B:

- Class A – Vessel mounted AIS transceiver (transmit & receive) which operates using SOTDMA. Default transmit rate is from 2 to 10 seconds, depending on vessel's speed or 3 minutes while a vessel is at anchor. AIS Class A type compliant devices receive all types of AIS messages.
- Class B – Vessel mounted AIS transceiver (transmit & receive) which operates using CSTDMA. Class B's can be connected to most display systems which the received messages will be displayed in lists or overlaid on charts. Default transmit rate is normally every 30 seconds, but this can be varied according to vessel speed or instructions from base stations. The Class B type standard requires integrated GPS and certain LED indicators. Class B does receive all types of AIS messages.

AIVDM sentences are emitted by AIS receivers. These receivers are fitted on vessels or base station. It reports to a computer by USB port or serial port, using NMEA 0183 format and physical network standard.

AIVDM is a two-layer protocol. The outer layer is a variant of NMEA 0183, the ancient standard for data interchange in marine navigation systems.

Following here is the typical AIVDM packet:

```
!AIVDM,1,1,,B,177KQJ5000G?tO`K>RAIwUbN0TKH,0*5C
```

1st field: (“!AIVDM”) identify this is AIVDM packet.

2nd field: (“1”) identify the count of fragment of this accumulating message.

3rd field: (“1”) identify the fragment number of this sentence.

4th field: (“”) is a sequential message ID for multi-sentence messages.

5th field: (“B”) identify radio channel code.

6th field: (“177KQJ5000G?tO`K>RAIwUbN0TKH”) is the data payload.

7th field: (“0*5C”) is the checksum for this sentence.

The data payload is an ASCII-encoded bit vector. Each character represents six bits of data. The first six bits identifies the message type of packet (in this example, it is 1=000001=1 in decimal). So, firstly, we need to convert this data into binary string. Then, based on each type of message we will have the following tables.

We focus mainly on message type 1, 2, 3 (class A), 4 (base station report) and 18 (class B) of AIS message types. The others are unusual or rare; even many AIS transmitters never emit them.

Type 1, 2 and 3 (Position Report Class A): These message types are highly popular to decoding software. It is broadcast every 2 to 10 seconds, total of 168 bits, occupying in one AIVDM sentence.

Field	Len	Description	Member	Units
0-5	6	Message type	type	Unsigned integer: 1-3
8-37	30	MMSI	mmsi	Unsigned integer: 9 digits
50-59	10	Speed Over Ground	Speed	Unsigned integer (1)
61-88	28	Longitude	lon	Minutes/10000 (2)
89-115	27	Latitude	lat	Minutes/10000 (3)

116-127	12	Course Over Ground	course	Relative to true north, to 0.1 degree precision (4)
128-136	9	True heading	heading	0 to 359 degrees
137-142	6	Time stamp	second	Second of UTC timestamp (5)

Type 4 (Base station report): periodically report its position and time reference to station, total of 168 bits, occupying one AIVDM sentence.

Field	Len	Description	Member	Unit
0-5	6	Message type	type	Unsigned integer: 4
8-37	30	MMSI	mmsi	Unsigned integer: 9
38-51	14	Year	year	UTC, 1-999, 0 = N/A
52-55	4	Month	month	1-12; 0 = N/A
56-60	5	Day	day	1-31; 0 = N/A
61-65	5	Hour	hour	0-23; 24 = N/A
66-71	6	Minute	minute	0-59, 60 = N/A
72-77	6	Second	second	0-59, 60 = N/A
79-106	28	Longitude	lon	(see above)

107-133	27	Latitude	lat	(see above)
---------	----	----------	-----	-------------

Type 18 (Standard Class B CS Position Report): A position report which is used by Class B transmitters, total 168 bits, occupying in one AIVDM sentence. Fields are encoded as in common navigation block.

Fields	Len	Description	Member	Unit
0-5	6	Message type	type	Unsigned integer: 18
8-37	30	MMSI	mmsi	Unsigned integer: 9 digits
46-55	10	Speed Over Ground	speed	(See above)
57-84	28	Longitude	lon	(See above)
85-111	27	Latitude	lat	(See above)
112-123	12	Course Over Ground	course	(See above)
124-132	9	True heading	heading	0 to 359 degrees, 511 = N/A
133-138	6	Time stamp	second	Second of UTC timestamp

- (1) Speed over ground is in 0.1 –knot resolution from 0 to 102 knots. Value 1023 indicates speed is not available.
- (2) Longitude is given in 1/10000 min; divide by 600000.0 to obtain degrees. Value 181 indicates that longitude is not available and is the default.

- (3) Latitude is given in 1/10000 min; divide by 600000.0 to obtain degrees.
Value 91 indicates latitude is not available and is the default.
- (4) Course over ground will be 3600 if this data is not available.
- (5) Seconds in UTC timestamp should be 0-59, except for these special values:
- 60: timestamp is not available
 - 61: positioning system is in manual input mode
 - 62: electronic position fixing system operates in estimated mode
 - 63: the positioning system is inoperative

Appendix C – Data Format for Chinese device

C.1 GPRS Data Format

#STA:xxxxxxxxx,ccc;TM:10/08/2008,08:15,V:xxxx;A1:nnnnn;A2:zzzzz;K:yyy
yy,SD:sssss,y;TH:abcdefg;%NMEA%

Command: PWD:1234,STATUS%

Reply Message:

#	Start Character		
Header	STA	Normal Message	
	STD	Historical Data	
	STR	Over speed Alarm Message	
	STF	Position Lock Alarm Message	
	STI	Fence In Alarm Message	
	STO	Fence Out Alarm Message	
	STP	Power Source Low Level Alarm Message	
	STM	Analog Input Channel 2 Triggered Message	
	STN	Analog Input Channel 1 Triggered Message	
xxxxxx,ccc		Station ID (10 digits), Hardware version (3 digits)	
TM:		Date Time	
V:		Power Source Voltage Level	
A1:	nnnnn	Analog Channel 1	e.g. A1:34.52
A2:	zzzzz	Analog Channel 2	e.g. A2:8.294
K:	yyyyy	1 st ~ 4 th digits: 4 alarm inputs, 0: open, 1: close	
		5 th digit: panic button, 0: open, 1: close	
SD:	sssss	current speed	y 0: mile per hour 1: km per hour 2: nautical mile per hour
TH:		Alarm Enable/Disable Status	
	a	Fence In Alarm	1: enable 0: disable
	b	Fence Out Alarm	1: enable 0: disable
	c	Alarm Input	1: enable 0: disable
	d	Position Lock Alarm	1: enable 0: disable

e	Idle Time Alarm	1: enable	0: disable
f	Analog Input 1 Alarm	1: enable	0: disable
g	Analog Input 2 Alarm	1: enable	0: disable
#	End Character		
%	NMEA Start Character		
NMEA:	Global Position System Protocol Message		
%	End Character		

C.2 NMEA Data Format

The National Marine Electronics Association (NMEA) has developed a specification that defines the interface between various pieces of marine electronic equipment. The standard permits marine electronics to send information to computers and to other marine equipment. GPS receiver communication is defined within this specification. Most computer programs that provide real time position information understand and expect data to be in NMEA format. This data includes the complete PVT (position, velocity, time) solution computed by the GPS receiver. The idea of NMEA is to send a line of data called a sentence that is totally self-contained and independent from other sentences [24].

There are many sentences in the NMEA standard for all kinds of devices that may be used in a Marine environment. This Chinese device used GPRMC (recommended minimum data for GPS) and GPVTG (Vector track an Speed over the Ground) as standard NMEA protocols when sending out message.

First of all, we will take a look into the first NMEA protocol: GPRMC

RMC - NMEA has its own version of essential gps pvt (position, velocity, time) data. It is called RMC, The Recommended Minimum, which will look similar to:

```
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A
```

Where:

RMC Recommended Minimum sentence C

123519 Fix taken at 12:35:19 UTC

A Status A=active or V=Void.

4807.038,N Latitude 48 deg 07.038' N

01131.000,E Longitude 11 deg 31.000' E

022.4 Speed over the ground in knots

084.4 Track angle in degrees True

230394 Date - 23rd of March 1994

003.1,W Magnetic Variation

*6A The checksum data, always begins with *

It is essential to know how to calculate Google Map coordinates from NMEA format. Following here is the simple instruction how to calculate this.

NMEA format LA:3324.4382,N

At first, we need to change this NMEA format into Degree-Minute-Second (D⁰M'S'') format:

$$\begin{aligned}
 3324.4382 &= 33^0 + 24.4382' = 33^0 + 24' + (60 * 0.4382)'' \\
 &= 33^0 + 24' + 26.292''
 \end{aligned}$$

Secondly, from this format, we change it into Decimal Degree format (Google Map format):

$$\begin{aligned}
 33^0 24' 26.29'' &= 33 + 24/60 + 26.29/(60*60) \\
 &= 33.4073027777
 \end{aligned}$$

The final result (in double format) is LA:33.407303,N

Then, let take a look on GPVTG protocol:

VTG - Velocity made good. The gps receiver may use the LC prefix instead of GP if it is emulating Loran output.

\$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K*48

Where:

VTG	Track made good and ground speed
054.7,T	True track made good (degrees)
034.4,M	Magnetic track made good
005.5,N	Ground speed, knots
010.2,K	Ground speed, Kilometers per hour
*48	Checksum

Heading degree: 054.7^0

Appendix D - A description about KML class

Keyhole Markup Language (KML) is an open standard officially named the OpenGIS® KML Encoding Standard (OGC KML) [25]. It is an international standard language for expressing geographic annotation and visualization on mobile maps and earth browsers [26]. It was developed by Google and maintained by Open Geospatial Consortium (OGC). It is a XML language concentrated on geographic visualization, including annotation of maps and images. In other words, it includes not only the presentation of graphical data but also the control of user's navigation in the sense of where to go, where to look and how to look. In this part, I will give some descriptions of the structure and specification of this standard and how to use it in LogTrack system.

KML is a XML language used in displaying geographical data on mobile maps and earth browser such as Google Map and Google Earth. It uses a tag-based structure with nested attributes and elements. Figure 35 shows the class tree of KML elements:

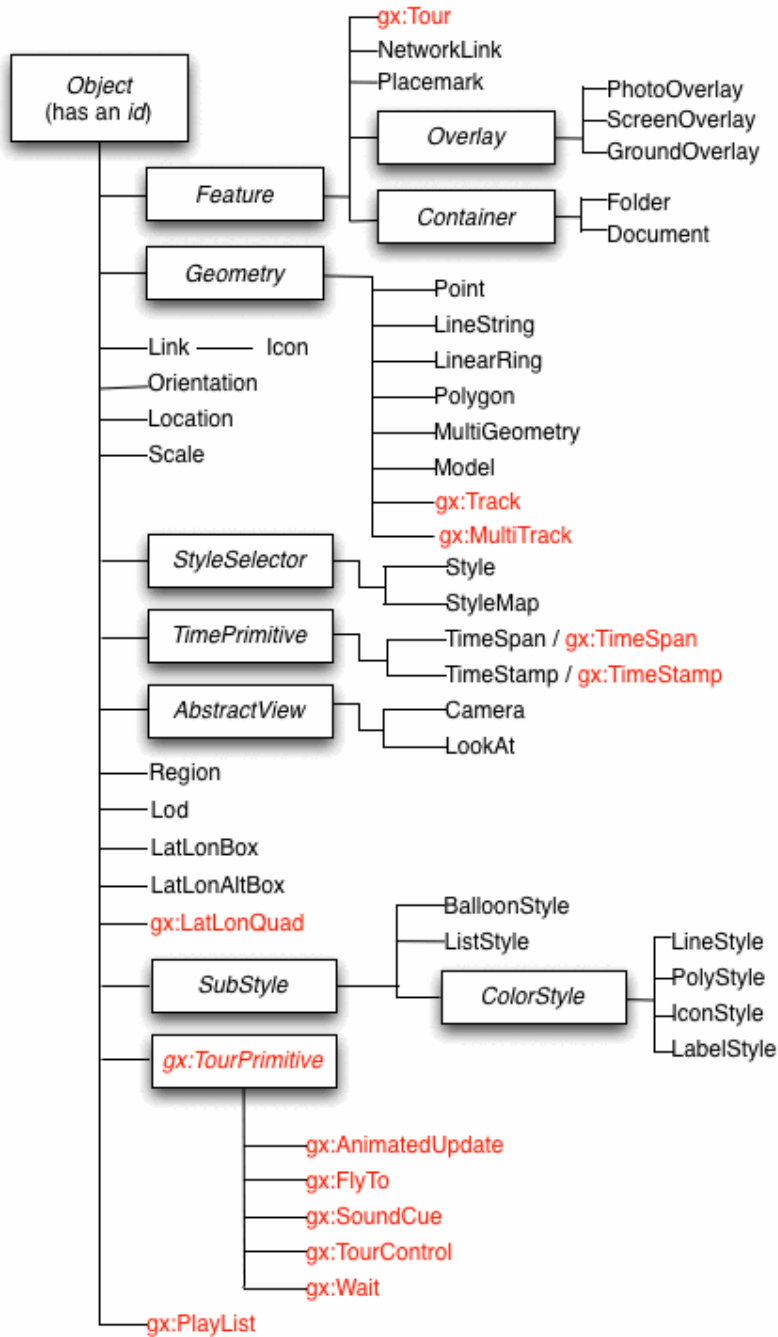


Figure 35. Class tree of KML elements [26].

The simplest KML document can be created directly by Google Earth. A sample of this basic KML document is shown in figure 36.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Placemark>
    <name>Simple placemark</name>
    <description>Attached to the ground. Intelligently places itself
      at the height of the underlying terrain.</description>
    <Point>
      <coordinates>-122.0822035425683,37.42228990140251,0</coordinates>
    </Point>
  </Placemark>
</kml>
```

Figure 36. A sample of basic KML document. [26]

The structure of this XML document is described as follows:

- An XML header
- A KML namespace declaration
- A Placemark object which contains following elements:
 - Name: name of place mark.
 - Description: a description of place mark which will appear in the “balloon” attached to the Placemark.
 - Point: specifies the coordinate of place mark on Earth’s surface (including longitude, latitude and optional altitude).

It is easy to attach HTML elements in this KML document by using CDATA element (figure 37):

```

<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <Placemark>
      <name>CDATA example</name>
      <description>
        <![CDATA[
          <h1>CDATA Tags are useful!</h1>
          <p><font color="red">Text is <i>more readable</i> and
            <b>easier to write</b> when you can avoid using entity
            references.</font></p>
        ]]>
      </description>
      <Point>
        <coordinates>102.595626,14.996729</coordinates>
      </Point>
    </Placemark>
  </Document>
</kml>

```

Figure 37. An example of how to use descriptive HTML in Placemark. [26]

It is one of the most benefits of KML language. It helps to display geographic visualization on Google Earth more efficiently.

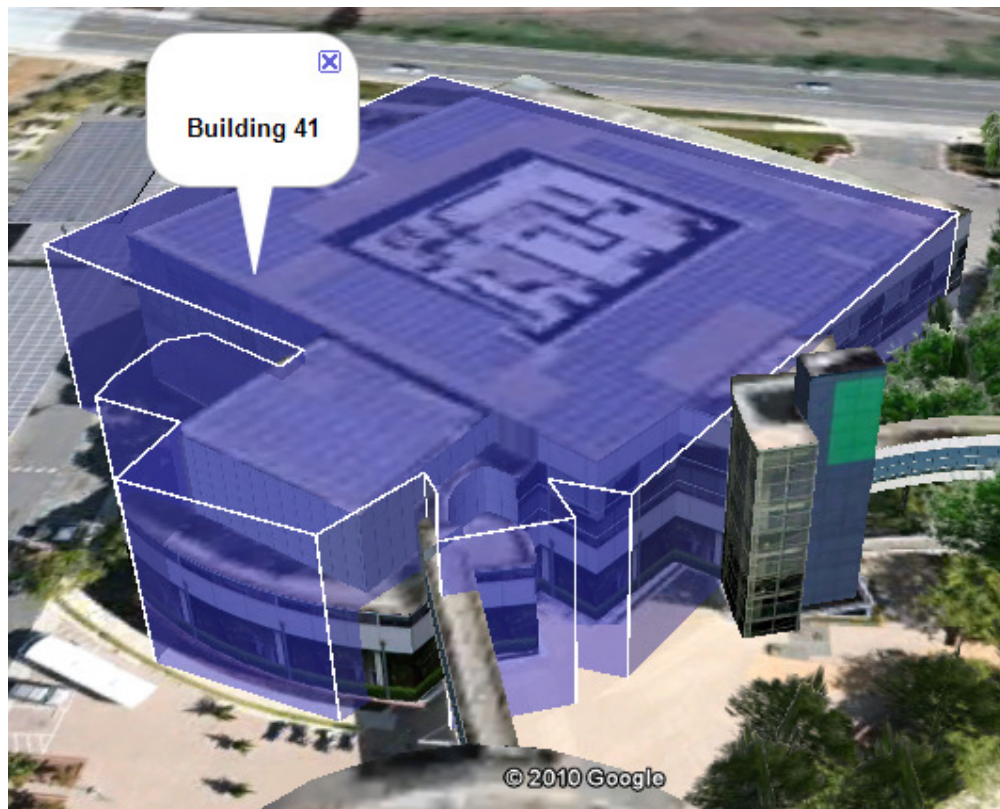
In advanced KML documents, it is possible to define styles for geometry. Style is an important part of KML document in the sense of how data is displayed. An example of style is shown below:


```

<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <Style id="transBluePoly">
      <LineStyle>
        <width>1.5</width>
      </LineStyle>
      <PolyStyle>
        <color>7dff0000</color>
      </PolyStyle>
    </Style>
    <Placemark>
      <name>Building 41</name>
      <styleUrl>#transBluePoly</styleUrl>
      <Polygon>
        <extrude>1</extrude>
        <altitudeMode>relativeToGround</altitudeMode>
        <outerBoundaryIs>
          <LinearRing>
            <coordinates> -122.0857412771483,37.42227033155257,17
              -122.0858169768481,37.42231408832346,17
              -122.085852582875,37.42230337469744,17
              -122.0858799945639,37.42225686138789,17
              -122.0858860101409,37.4222311076138,17
              -122.0858069157288,37.42220250173855,17
              -122.0858379542653,37.42214027058678,17
              -122.0856732640519,37.42208690214408,17
              -122.0856022926407,37.42214885429042,17
              -122.0855902778436,37.422128290487,17
              -122.0855841672237,37.42208171967246,17
              -122.0854852065741,37.42210455874995,17
              -122.0855067264352,37.42214267949824,17
              -122.0854430712915,37.42212783846172,17
              -122.0850990714904,37.42251282407603,17
              -122.0856769818632,37.42281815323651,17
              -122.0860162273783,37.42244918858722,17
              -122.0857260327004,37.42229239604253,17
              -122.0857412771483,37.42227033155257,17
            </coordinates>
          </LinearRing>
        </outerBoundaryIs>
      </Polygon>
    </Placemark>
  </Document>

```

Here is the visualization on Google Earth:



Google Earth Plug-in provides methods for importing KML documents into Google Earth by using Web Browser object and javascript. Here is an example of how to use those methods (written in JavaScript):

```
//Function: parse KML string to GE.
function JSParseKml(kmlString) {
    try {
        var kmlObject = ge.parseKml(kmlString);
        ge.getFeatures().appendChild(kmlObject);
    }
    catch (ex) {
        alert('Parse error: Invalid KML file');
    }

    //fly to view: get camera from KML file and setAbstractView
    if (kmlObject.getAbstractView())
        ge.getView().setAbstractView(kmlObject.getAbstractView());
}
```

