Santhosh Kumar Kadari

Development, Implementation, and Testing of a GIS & GPS based Real-Time Vehicle Tracking System

Masterarbeit

zur Erlangung des akademischen Grades Diplom-Ingenieur

Studium Information Technology

Alpen-Adria-Universität Klagenfurt
Fakultät für Technische Wissenschaften

Begutachter: Univ.-Prof. Dr.-Ing. Kyandoghere Kyamakya

Institute for Smart System-Technologies
Transportation Informatics

Klagenfurt, im April 2010
Eidesstattliche Erklärung

Ich erkläre ehrenwörtlich, dass ich die vorliegende wissenschaftliche Arbeit selbstständig angefertigt und die mit ihr unmittelbar verbundenen Tätigkeiten selbst erbracht habe. Ich erkläre weiters, dass ich keine anderen als die angegebenen Hilfsmittel benutzt habe. Alle aus gedruckten, ungedruckten oder dem Internet im Wortlaut oder im wesentlichen Inhalt übernommenen Formulierungen und Konzepte sind gemäß den Regeln für wissenschaftliche Arbeiten zitiert und durch Fußnoten bzw. durch andere genaue Quellenangaben gekennzeichnet.

Die während des Arbeitsvorganges gewährte Unterstützung einschließlich signifikanter Betreuungshinweise ist vollständig angegeben.


Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

(Unterschrift)  

(Ort, Datum)
Abstract

In today’s wireless world, the word "communication" is everywhere, with everybody and at any time. We have come very close to this goal during the last few years. Along with the latest mobile communication technologies, people have been using two other technologies, including Global Positioning System (GPS) and Geographic Information System (GIS) for communication. The need for vehicle tracking and navigation has brought the emergence of GPS technology and the revolution in mapping procedures has brought GIS technology to the everyday life of common people.

The main objective of my thesis is to develop, implement and test a software which can be used in precision farming. Precision farming is a concept that denotes collecting field information using new technologies in agriculture. This information is used to estimate the quantity of fertilizers needed, to evaluate the optimum sowing density and to predict crop yields.

The steps to be followed in precision farming are:

- Gathering/Collecting of information on the sub-field
- Analyzing the gathered field information
- Considering/Taking decision steps based on analyzed information
- Implementation/Development of these decisions

The software to be developed will provide reliable communication between three parties, namely the User (Farmer), the Logistics Office (Central Office) and the Vehicle (Mobile Terminal) by using GIS and GPS technologies.

The working process of the software is as follows:

- First, the farmer sends his/her address, field and GIS information to the logistics office.
- Next, the logistics office sends the farmer field information to the vehicle.
- Finally, the vehicle finishes the farmer field work.

The vehicle sends its location information to the logistics office every 10 seconds via GPS. Based on this information, the logistics office relays the farmer’s field information to the vehicle closest to the field. This field information includes a GIS ID with the help of which the vehicle locates the farmer’s field.
In my thesis project, two different software languages have been used. These are Delphi (Delphi IDE) and J2ME (Netbeans IDE). Delphi is used to create the Graphical User Interface (GUI) form/window(s) at the position of farmer, central office and vehicle. J2ME allows the user to create an environment to extract the GPS information from the GPS receiver.

My thesis project aims at providing reliable communication between farmer, logistics office and vehicle and also reducing the idle times in waiting for vehicles.

The thesis contains eight chapters. The first chapter elaborates on the research background and motivation as well as the problem statement, research objectives, research questions and methodology. The second chapter gives a comprehensive summary of the related state-of-the-art of the following technologies: precision farming, mobile GIS, positioning technologies and location-based services. The third chapter explains the software development process and programming basics that are related to the software development life cycle, requirements engineering concepts of software requirements, designing concepts such as UML as well as database and programming design methodology.

The fourth chapter describes the development and implementation of the system design with the help of UML design, database design and software architecture. The fifth chapter describes the testing of the system design. A GIS and GPS based real-time vehicle tracking system and interfaces are established and a prototype is tested. The sixth chapter gives the conclusion and offers recommendations with respect to the research questions.

The seventh chapter presents the source code which has been used to implement the software. The eighth chapter gives the description of the hardware and software platforms that are related to the following technologies: GIS, DBMS, IDE’s, mobile terminals, servers and wireless communication system.

**Keywords:** Geographic Information System, Global Positioning System, Central Office, Mobile Terminal.
Acknowledgments

First and foremost I would like to thank Univ.-Prof. Dr.-Ing. Kyandoghere Kyamakya for accepting my request to work in the PROGIS company and for his constant encouragement and his constant support, advice throughout my work. I would like to thank C.E.O of the company Dip.-Ing. W.H.Mayer for giving me an opportunity to work in the company. It was a great learning experience working with the other colleagues in the company.

The informal support and encouragement of many friends has been indispensable, and I would like particularly to acknowledge the contribution of Adarsh Samrat for his constant encouragement and assistance throughout my master’s program.

Finally, I would like to express my special thanks of gratitude to my parents Kantha Rao and Indira Devi for their moral, emotional, financial and unconditional support. I would also like to thank my brother who helped me a lot during my studies. I am heartily thankful to my late grandfather Gopal Rao Neelagiri and late former chief minister of Andhra Pradesh Dr.Rajasekhara Reddy Yeduguri Sandinti who always inspired me to work hard and to be positive all times.
# Contents

## 1 Introduction

1.1 Background and Motivation .................................................. 1
1.2 Problem Statement, Objectives and Research questions .................. 2
1.3 Methodology to solve the problems ........................................ 4
  1.3.1 Methodology to reach objective 1 ..................................... 4
  1.3.2 Methodology to reach objective 2 ..................................... 4
  1.3.3 Methodology to reach objective 3 ..................................... 5
  1.3.4 Methodology to reach objective 4 ..................................... 5
  1.3.5 Methodology to reach objective 5 ..................................... 6
  1.3.6 Methodology to reach objective 6 ..................................... 7
  1.3.7 Methodology to reach objective 7 ..................................... 7
1.4 Outline of the thesis ......................................................... 8

## 2 A comprehensive summary of the related state-of-the-art ................. 10

2.1 Precision Farming .................................................................. 10
  2.1.1 Concept of Precision Farming .......................................... 10
  2.1.2 Goals of Precision Farming system .................................... 10
  2.1.3 Functionalities of Precision Farming system ....................... 11
  2.1.4 Basic Tasks of Precision Farming .................................... 11
  2.1.5 Basic required technologies in Precision Farming ................ 12
  2.1.6 Requirements of wireless technologies in Precision farming .. 12
  2.1.7 Communication system architecture ................................... 13
  2.1.8 Reasons for not considering GSM based positioning ............. 14
2.2 MOBILE GIS ....................................................................... 16
  2.2.1 Introduction to GIS ....................................................... 16
  2.2.2 Key Components of GIS ................................................ 16
  2.2.3 Key Functions of GIS .................................................... 17
  2.2.4 GIS Operations ............................................................ 17
  2.2.5 What is Mobile GIS ....................................................... 17
  2.2.6 Why Mobile GIS .......................................................... 18
  2.2.7 Tasks of Mobile GIS ...................................................... 18
  2.2.8 Benefits of Mobile GIS .................................................. 18
  2.2.9 Key components of Mobile GIS ....................................... 19
  2.2.10 Generic Architecture of Mobile GIS ............................... 19
2.3 Positioning Technologies ....................................................... 22
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.1 Definitions and terms</td>
<td>22</td>
</tr>
<tr>
<td>2.3.2 Positioning methods</td>
<td>22</td>
</tr>
<tr>
<td>2.3.3 Types of Positioning Systems in ITS</td>
<td>23</td>
</tr>
<tr>
<td>2.3.4 Introduction to GPS</td>
<td>24</td>
</tr>
<tr>
<td>2.3.5 GPS Segments</td>
<td>25</td>
</tr>
<tr>
<td>2.3.6 Working process of GPS Technology</td>
<td>26</td>
</tr>
<tr>
<td>2.3.7 The GPS Error Budget</td>
<td>28</td>
</tr>
<tr>
<td>2.3.8 DGPS Technology</td>
<td>29</td>
</tr>
<tr>
<td>2.3.9 RTK - GPS Technology</td>
<td>29</td>
</tr>
<tr>
<td>2.3.10 Components description of GPS receiver</td>
<td>30</td>
</tr>
<tr>
<td>2.3.11 Cellular Networks based positioning systems</td>
<td>31</td>
</tr>
<tr>
<td>2.3.12 GSM positioning versus GPS positioning</td>
<td>34</td>
</tr>
<tr>
<td>2.4 Mobile Applications and/or Location-Based Services (LBS)</td>
<td>35</td>
</tr>
<tr>
<td>2.4.1 Introduction to LBS</td>
<td>35</td>
</tr>
<tr>
<td>2.4.2 Concepts in LBSs</td>
<td>35</td>
</tr>
<tr>
<td>2.4.3 What are LBSs</td>
<td>36</td>
</tr>
<tr>
<td>2.4.4 LBS communication model</td>
<td>37</td>
</tr>
<tr>
<td>3 Software Development Process and Programming Basics</td>
<td>39</td>
</tr>
<tr>
<td>3.1 Basics of software development process</td>
<td>39</td>
</tr>
<tr>
<td>3.1.1 Requirements Analysis</td>
<td>39</td>
</tr>
<tr>
<td>3.1.2 Design</td>
<td>41</td>
</tr>
<tr>
<td>3.1.3 Implementation</td>
<td>41</td>
</tr>
<tr>
<td>3.1.4 Testing</td>
<td>41</td>
</tr>
<tr>
<td>3.1.5 Maintenance</td>
<td>41</td>
</tr>
<tr>
<td>3.2 Basics of the Requirement Engineering of Software systems</td>
<td>43</td>
</tr>
<tr>
<td>3.2.1 Introduction to Requirements Engineering</td>
<td>43</td>
</tr>
<tr>
<td>3.2.2 Definition to Requirements Engineering</td>
<td>43</td>
</tr>
<tr>
<td>3.2.3 Requirements Engineering Process</td>
<td>44</td>
</tr>
<tr>
<td>3.2.4 UML-Basics</td>
<td>45</td>
</tr>
<tr>
<td>3.2.5 Structural modeling</td>
<td>45</td>
</tr>
<tr>
<td>3.2.6 Behavioral modeling</td>
<td>46</td>
</tr>
<tr>
<td>3.3 SysML - Basics</td>
<td>48</td>
</tr>
<tr>
<td>3.3.1 Diagram Overview of SysML</td>
<td>48</td>
</tr>
<tr>
<td>3.4 Database Design basics/methodology</td>
<td>49</td>
</tr>
<tr>
<td>3.4.1 Elements of Relational Database</td>
<td>50</td>
</tr>
<tr>
<td>3.4.2 Table Relationships</td>
<td>50</td>
</tr>
<tr>
<td>3.4.3 Database Design Basics</td>
<td>50</td>
</tr>
<tr>
<td>3.5 Programming Design (basics/methodology)</td>
<td>51</td>
</tr>
<tr>
<td>3.5.1 Algorithms</td>
<td>51</td>
</tr>
<tr>
<td>3.5.2 Software Architecture</td>
<td>52</td>
</tr>
<tr>
<td>3.6 Software Testing</td>
<td>53</td>
</tr>
<tr>
<td>3.6.1 Software Testing Principles</td>
<td>54</td>
</tr>
<tr>
<td>3.6.2 Possible error types found by Software Testing</td>
<td>54</td>
</tr>
</tbody>
</table>
4 Software Design and Implementation Process 56
    4.1 UML design .............................................. 56
        4.1.1 Use-case diagram .................................... 56
        4.1.2 Class diagram ........................................ 57
        4.1.3 Object diagram ....................................... 58
        4.1.4 Component diagram ................................... 58
        4.1.5 Deployment diagram ................................... 59
        4.1.6 Sequence diagram ..................................... 59
        4.1.7 Collaboration diagram ................................ 59
        4.1.8 State diagram ......................................... 61
        4.1.9 Activity diagram ...................................... 61
    4.2 DB design .................................................. 63
    4.3 Software architecture ................................. 65
        4.3.1 Outflow of software architecture ...................... 65
        4.3.2 Internal software architecture at farmer side ...... 66
        4.3.3 Internal software architecture at Logistics Office 67
        4.3.4 Internal software architecture at Vehicle ........... 68
    4.4 Selection of the appropriate IDE’s .................... 69
    4.5 Source code for software implementation ............ 69

5 Testing and Interpretation of results 70
    5.1 Testing of the software implemented .................. 70
        5.1.1 Black-box testing: .................................. 71
        5.1.2 White-box testing: ................................... 71
        5.1.3 Interpretation of the results when Black-box testing is applied .................................. 71
        5.1.4 Interpretation of the results when White box testing is applied ................................ 72
    5.2 Comment on tests conducted ............................... 74
    5.3 Software implementation process ....................... 74
        5.3.1 Graphical User Interface of the software developed .................................................. 74

6 Conclusion .................................................. 84

7 Annex1 - Code implementation(Delphi, J2ME) ............. 86

8 Annex2 - Description of the Hardware and Software Platforms 113
    8.1 Geographic Information System .......................... 113
        8.1.1 WinGIS .................................................. 114
    8.2 DBMS (Microsoft Access, MySQL, PostgreSQL) ........ 115
        8.2.1 Database Types ......................................... 116
        8.2.2 Microsoft Access Database ............................ 117
        8.2.3 MySQL Database ......................................... 117
        8.2.4 PostgreSQL Database .................................... 119
    8.3 IDE’S (Delphi 6.0, Netbeans 6.7) ....................... 119
        8.3.1 Delphi IDE .............................................. 119
        8.3.2 Netbeans IDE ............................................ 124
    8.4 Mobile Terminals ......................................... 127
        8.4.1 E71 Mobile device ....................................... 128
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.4.2 N91 Mobile device:</td>
<td>129</td>
</tr>
<tr>
<td>8.5 Server</td>
<td>129</td>
</tr>
<tr>
<td>8.6 Wireless Communication System</td>
<td>130</td>
</tr>
<tr>
<td>8.6.1 Key benefits of GPRS</td>
<td>130</td>
</tr>
<tr>
<td>8.6.2 GPRS Network</td>
<td>131</td>
</tr>
</tbody>
</table>

**Bibliography**

132
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Flow Diagram of Methodology.</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>Wireless Communication techniques.</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>Communication System Architecture.</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>Generic architecture of mobile GIS.</td>
<td>19</td>
</tr>
<tr>
<td>2.4</td>
<td>Differential GPS System.</td>
<td>29</td>
</tr>
<tr>
<td>2.5</td>
<td>Real-Time Kinematic GPS.</td>
<td>30</td>
</tr>
<tr>
<td>2.6</td>
<td>The generic block diagram GPS Receiver.</td>
<td>31</td>
</tr>
<tr>
<td>2.7</td>
<td>Cells in the GSM network.</td>
<td>34</td>
</tr>
<tr>
<td>2.8</td>
<td>General LBS communication.</td>
<td>37</td>
</tr>
<tr>
<td>3.1</td>
<td>A simple view of software development.</td>
<td>40</td>
</tr>
<tr>
<td>3.2</td>
<td>The Waterfall model.</td>
<td>42</td>
</tr>
<tr>
<td>3.3</td>
<td>The process of design from the present through to future systems.</td>
<td>43</td>
</tr>
<tr>
<td>3.4</td>
<td>Activities within requirements.</td>
<td>44</td>
</tr>
<tr>
<td>3.5</td>
<td>UML vs SysML.</td>
<td>48</td>
</tr>
<tr>
<td>3.6</td>
<td>SysML Diagram Taxonomy.</td>
<td>49</td>
</tr>
<tr>
<td>4.1</td>
<td>UML use-case diagram.</td>
<td>56</td>
</tr>
<tr>
<td>4.2</td>
<td>UML class diagram.</td>
<td>57</td>
</tr>
<tr>
<td>4.3</td>
<td>UML object diagram.</td>
<td>58</td>
</tr>
<tr>
<td>4.4</td>
<td>UML component diagram.</td>
<td>59</td>
</tr>
<tr>
<td>4.5</td>
<td>UML deployment diagram.</td>
<td>60</td>
</tr>
<tr>
<td>4.6</td>
<td>UML sequence diagram.</td>
<td>60</td>
</tr>
<tr>
<td>4.7</td>
<td>UML collaboration diagram.</td>
<td>61</td>
</tr>
<tr>
<td>4.8</td>
<td>UML state diagram.</td>
<td>62</td>
</tr>
<tr>
<td>4.9</td>
<td>UML activity diagram.</td>
<td>62</td>
</tr>
<tr>
<td>4.10</td>
<td>Database Form.</td>
<td>63</td>
</tr>
<tr>
<td>4.11</td>
<td>Table designing view.</td>
<td>64</td>
</tr>
<tr>
<td>4.12</td>
<td>Store the data values in table.</td>
<td>64</td>
</tr>
<tr>
<td>4.13</td>
<td>Outflow of software architecture.</td>
<td>66</td>
</tr>
<tr>
<td>4.14</td>
<td>Internal software architecture at farmer.</td>
<td>67</td>
</tr>
<tr>
<td>4.15</td>
<td>Internal software architecture at logistics side.</td>
<td>68</td>
</tr>
<tr>
<td>4.16</td>
<td>Internal software architecture at vehicle side.</td>
<td>69</td>
</tr>
<tr>
<td>5.1</td>
<td>Test bed process.</td>
<td>70</td>
</tr>
<tr>
<td>5.2</td>
<td>Black-box testing approach.</td>
<td>72</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Comparison between wireless LAN, Bluetooth and ZigBee. . . . . . . . . . . 13
Chapter 1

Introduction

1.1 Background and Motivation

Development of precision farming was first started in Europe. Additionally with new technologies, precision farming was taken up in the USA and developed at great pace. Over the past several years, agricultural machinery has been developed to high technical standards in accordance with the enhancement of precision farming. The term 'Teileflächenwirtschaft' is the German term for precision farming. In summary the following points were achieved:

- New types of machines (tractors) have been developed whereby an operator can work all day in a comfortable environment.
- Application equipments such as spreaders and sprayers have developed to achieve uniform application of plant protection chemicals and fertilizers.

Key Components of a precision farming typically include:

- A yield monitor
- Intensive testing of plant issues or soils with in a field
- Equipment for locating a position within a field via the Global Positioning System (GPS).
- Variable-rate technologies for seed, pesticides, fertilizer, and irrigation water that are continually computer adjusted for various parts of the field.
- A computer to manipulate and store spatial data using some form of Geographic Information System (GIS) software.

The advancements in agriculture technology replaced the tools and the equipments used for farming resulting in higher yields. The basic concept of the precision farming is the collection of the data and making decisions using the collective data has been around for many years. One can work out on small fields very easily without this technology. As the size of the farming increases this becomes a problem that should be handled to increase the yields. Hence precision farming comes into play where the size of the farms matter. GIS is one of the first developed tools of the precision farming. It was used by the research
institutions at the start and was later commercialized. The maps designed in GIS can be uploaded into the GPS system and hence GPS system can be used as a source of the path for the automated guided vehicle systems. Thus, GPS played a major role in precision farming.

1.2 Problem Statement, Objectives and Research questions

The main focus of this thesis work is to develop a Tracking System application for supporting various processes involved in the real-time operations of a modern precision farming system. This tracking will basically involve two front-end types.

The first front-end is a mobile terminal concept which contains/involves/embed the following core technologies:

- An appropriate GUI.
- GSM and GPRS communication capabilities.
- The "mobile GIS" technology.
- GPS positioning capability.

This front-end will be located in diverse vehicles and tractors that are involved in different farming and/or transportation processes in the frame of precision farming.

The second front-end type is a desk-top based interface for the farmer, through which he should interact with the tracking system and the overall precision-farming system, both in real-time and in off-line mode. Through this interface the farmer can, amongst other things, either formulate his farming needs/requests or check the processing status of his requests by the precision farming system. Concerning the back-end, the mobile and tracking application to be developed should efficiently communicate with an already existing server, the so-called "Logistics Office Server" which is part of a complex precision farming IT system.

Therefore, this thesis is mainly addressing, in the core, the complete Systems Engineering of this tracking system involving a mobile application for the real-time support of precision farming operations. This results in the following specific objectives and/or research issues:

1. Summarize the basic principles and the state-of-the-art of precision farming and extract to formulate the related requirements (performance, availability, system architecture and so on) with regard to wireless communication technologies, Geographic Information Systems, and Global Positioning Systems. It should be discussed why GPS based positioning is not considered here.

2. Summarize the basic principles and the related state-of-the-art of the GIS technology with a focus on mobile GIS.
3. Summarize the basic principles and the related state-of-the-art of the Positioning
technologies with a focus on GPS and GSM based positioning.

4. Summarize the basic principles and the related state-of-the-art of the so-called mobile
applications (and/or location-based Services).

5. Summarize the basics of the software-related Systems Engineering.

6. Apply the Systems Engineering methods to the case "Tracking System". A compre-
hensive design is conducted by using the Systems Engineering instruments presented
in the objective 5.

7. Test the software system developed by applying the methods presented in the objec-
tive 6.

Figure 1.1: Flow Diagram of Methodology.
1.3 Methodology to solve the problems

1.3.1 Methodology to reach objective 1

Since 40 years, the development of precision farming has been enhancing using new technologies. Along with them, GIS, GPS and new types of wireless technologies are played major role in precision farming system. In this work, we have been reviewed the following literatures to reach this objective.

- **Data acquisition system:** A mobile field data acquisition system has been developed by Gomide et al. (2001) to collect data for crop management [80]. It consists of a data collection vehicle, a manager vehicle, a data acquisition system and a control system on farm machines. The system allows to conduct local survey and to collect data of soil water availability, soil fertility, soil compaction, leaf temperature, leaf chlorophyll content, local climate status, plant water status, grain yield, etc. The data collection vehicle retrieves data from machines via WLAN, analyzes, stores and transmits the data to the manager vehicle. The manager in the manager vehicle monitors the performance of the farm machines and data acquisition systems.

- **Yield mapping system:** A yield mapping system has been developed by Lee et al. (2002) [51]. This system consists of load cells, a moisture sensor, a GPS and a Bluetooth wireless communication module. The Bluetooth transmitter and the moisture sensor are installed on the chopper. Signal from the moisture sensor sends to a Bluetooth receiver on a host PC at a datarate of 115kbps.

- **Infrared thermometer system:** A wireless, infrared thermometer system has been developed by Mahan and Wanjura (2004) to collect data in the field [56]. This system includes infrared sensors, programmable Logic Controllers and low power radio transceivers to collect data in the field and transmits it to a remote receiver outside the field.

- **Automatic irrigation system:** A distributed, remotely controlled, automatic irrigation system has been developed and tested by Damas et al. (2001) to control a 1500 ha irrigated area in Spain [60]. The area was divided into seven sub-regions with a total of 1850 hydrants installed. Each sub-region was monitored and controlled by a control sector. The seven control sectors communicated to each other and with a central control through a WLAN network. Field tests showed water savage of 30-60 percentage.

- **Web server:** A web server has been developed by Jensen et al. (2000) provided information on pest and disease infestation and weather forecasts [7]. The information can be directly downloaded by the farmers via Internet and use them for operation scheduling.

1.3.2 Methodology to reach objective 2

Since many years, the GIS applications and its technological trends have been rapidly increasing which became so large due to the success of the World Wide Web. We have seen geoinformation technology emerging from mainframe computers to stand alone desktop
computer GIS through to local networking GIS, to the Web GIS and now mobile GIS where maps are processed on small mobile devices like PDAs and mobile phones. In this work, we have been reviewed the following literatures to reach this objective.

- Thomson (Thomson, 2002) focused on traffic data collection for travel time analysis by developing a prototype that integrated GIS and GPS technologies.
- Tsou (Tsou, 2004) also developed a mobile GIS prototype that allows multiple park rangers to access large-size, remotely sensed images and GIS layers from a web server mounted in a vehicle.
- In (Vivoni and Camilli, 2003), the development of a prototype mobile GIS environmental field data collection system for two way transfer and display of collected data between field site and remote location server has highlighted.
- Kang and Li (Kang and Li, 2005) presented a framework mechanism which deals with the maintenance of topological consistencies in updating a map data.

In the above listed works and many more, mobile GIS and data collection applications have been developed in various disciplines.

1.3.3 Methodology to reach objective

Since many years, different types of positioning technologies have been using to determine position of an object, mobile device, etc. These positioning technologies are divided into two groups. They are network-based and terminal-based positioning systems. In this work, we have been reviewed the following literatures to reach this objective.

A few research and commercial technologies are used for providing location-aware services in outdoor environments. These technologies are divided into network-based and handset-based ones (Burnham, 2002).

Giaglis et al. (2002) provides detailed analysis of network based positioning technologies. The most popular technologies of this category are Cell-Identification (Cell-ID), Enhanced Cell-Identification (E-Cell-ID), Time of Arrival (TOA), Observed Time Difference (OTD), Time Difference of Arrival (TDOA) and Enhanced-OTD (E-OTD).

Conversely, the handset-based technologies provide location identification information even in the absence of mobile network coverage. The prevalent information in this category is the Global Positioning System (GPS). GPS is the world wide satellite-based radio navigation system, consisting of 24 satellites, equally spaced in six orbital planes 20,200 Km above the earth, that transmit two specially coded carriers. Several applications are included i.e. one for civilian use and one for military and government use (Djuknic & Richton, 2001).

1.3.4 Methodology to reach objective

Since the early 90s the interest in Location Based Services (LBS) has increased with the help of mobile and wireless communication technologies. An increasing number of mobile
phones and Personal Device Assistants (PDA) allow people to access the Internet across the space. From the Internet they can obtain on one hand information on events and on the other hand information on places. In this work, we have been reviewed the following literatures to reach this objective.

- One of the first systems is to integrate "personalized" information to the user. The mobile city guide [21] for the city of Lancaster, UK.

- More recent projects like presenting a city guide for the city of Vienna, Austria to integrate multimedia in mobile city guides. Multimedia. The aim of the presented approach of these systems is not to address a dynamic creation of personalized [76].

- In the context of the OPERA project [10], when looking on the dynamic creation of personalized content, we find interesting research approaches with the Cuypers system [69]. Even though dealing with personalization, mobile devices are not in their focus.

The dynamic creation of mobile multimedia content can be found, e.g., with the research approaches, which use constraints, transformation rules, style sheets, and other to achieve the generation of personalized multimedia content. However, our observation is that these approaches are limited when it comes to more complex or very application specific personalization tasks [58] [52].

1.3.5 Methodology to reach objective 5

A software development life cycle represents different phases. In order to design the software requirements the software project goes through in and perform the program changes, test the program cycle to ensure that changes are accurate, and install the changes into the running system. In this project, we have been reviewed the following literatures to reach this objective.

- The software process model can be defined as a networked sequence of activities, objects, transformations, and events which includes strategies for accomplishing software evolution (Marciniak, J.J., 2001). Each stage of the software process is identified and the representation of the inherent activities associated within that stage is employed using a model. There are a few software process models that serve as an abstract representation of the software process. These include: waterfall model, spiral model, prototyping model and extreme programming [59].

- Objected-oriented analysis is relatively a young approach to requirements analysis. According to Coad and Yourdon (1991), it is based upon concepts that we learned in kindergarten: objects and attributes, wholes and parts, classes and members [15].

- According to Davis (1993), the primary motivation for objected orientation is that, as a system evolves, its functions tend to change but its objects remain unchanged. Thus, a system built using object-oriented techniques may be inherently more maintainable than one built using more conventional functional approaches [22].
Chapter 1. Introduction

1.3.6 Methodology to reach objective

The Unified Modeling Language (UML) is a standard language for modeling, visualizing, specifying, constructing and documenting the artifacts of software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems and it is a very important part of developing object-oriented software. From the study of state of the art, Three main approaches were identified which are followed for UML security specification in the software development process.

- The first approach is based on using the language artifacts provided by standard UML: Stereotypes, the Object Constraint Language (OCL) [15] and behavior diagrams.

- In the second approach, the UML meta-language is augmented by new language constructs allowing the specification of security requirements.

- The third approach, Includes in defining a new specification language to specify security requirements on UML diagrams.

In this project, we have been reviewed the following literatures to reach this objective.

- Gogolla and Henderson - Sellers [36] provide an analysis on UML stereotypes and some suggestions were proposed for the improvement of the definition and use of stereotypes within the UML meta-data.

- Schleicher and Westfechtel [84] discuss and evaluate the UML meta-language. A classification of stereotypes and a comparison of different approaches of extending the UML are also illustrated. Finally, the paper proposes various ways to extend the UML meta-model for better readability, expressiveness, and verifiability of the extensions.

- Regarding Object Constraint Language (OCL), [38] discusses a number of issues related to the syntax and semantics of OCL such as navigation, state models, object creation, etc. Additionally, the paper proposes some solutions for clarification and extension of the OCL.

The implementation and use of database management systems has become increasingly important over the past twenty years. According to (Toby J. Teorey and James P. Fry), [95] the design of a database is an important consideration in achieving effective database usability. There are different ways of approaching database design and its two phases - the design of a DBMS-processable logical database structure that describes the user’s view of the data, and the selection of a physical structure to represent the logical design.

1.3.7 Methodology to reach objective

Software Testing is the process of executing a program with the intent of finding errors. Software errors / bugs will almost always in any software module with moderate size, because the complexity of software is generally intractable and humans have only limited
to manage complexity. In this work, we reviewed the following literatures to reach this objective.

- The black-box approach is a testing method in which test data are derived from the specified functional requirements without regard to the final program structure \[74\]. It is also termed data-driven, input/output driven \[64\], or requirements based testing \[41\].

- Contrary to black-box testing, software is viewed as a white-box, as the structure and flow of the software under test are visible to the tester. Testing plans are made according to the details of the software implementation, such as programming language, logic, and styles. Test cases are derived from the program structure. White-box testing is also called glass-box testing, logic-driven testing \[64\], or design-based testing \[41\].

- According to Koutsoukos (2004), there are two types of service-oriented reengineering. There are: Black box (does not need knowledge of the code) and White box (source code is modified or changed) \[39\]. White box method can be seen as a more permanent solution than block box but extra time may be needed to understand the code if the system is not correctly documented \[39\]. Black box methods can be difficult to apply when business logic is tied up with presentation logic as software services require clean functional interfaces \[49\].

1.4 Outline of the thesis

The thesis contains eight chapters namely: introduction as chapter 1; a comprehensive summary of the related state-of-the-art as chapter 2; software development process and programming basics as chapter 3; software design and implementation process as chapter 4; testing and interpretation of results as chapter 5; Conclusion as chapter 6; source code implementation as Annex 1 and description of the hardware and software platforms as Annex 2.

- Chapter 2 gives a comprehensive summary of the related state-of-the-art of the following technologies: precision farming, mobile GIS, positioning technologies and location-based services.

- Chapter 3 explains the software development process and programming basics that are related to the software development life cycle, requirements engineering concepts of software requirements, designing concepts such as UML as well as database and programming design methodology.

- Chapter 4 describes the development and implementation of the system design with the help of UML design, database design and software architecture.

- Chapter 5 describes the testing of the system design. A GIS and GPS based real-time vehicle tracking system and interfaces are established and a prototype is tested.

- Chapter 6 gives the conclusion and offers recommendations with respect to the research questions.
• Chapter 7 presents the source code which has been used to implement the software.

• Chapter 8 gives the description of the hardware and software platforms that are related to the following technologies: GIS, GPS, DBMS, IDE’s, mobile terminals, servers and wireless communication system.
Chapter 2

A comprehensive summary of the related state-of-the-art

2.1 Precision Farming

This sub-chapter summarizes the basic principles and the related state-of-the-art of precision farming and formulates the required parameters including performance, availability, system architecture, precision, cost and device size that have with regard to wireless communication technologies, Geographic Information System, and positioning technologies.

2.1.1 Concept of Precision Farming

Precision farming is also called as precision agriculture. It is intensive cultivation technology of modern agriculture in order to obtain high grade, high production and high efficiency by using modern information technology [100].

In other words, the concept of precision farming can be defined as a management practice with the potential to increase profits by utilizing more precise information [35]. GPS technology makes it possible for farmers to measure yields and responses to management practices within their own fields. By using GPS, it is possible to acquire numerous measurement of environmental or yield conditions, and fuse them with computer-aided collection, and prepare maps of yield characteristics.

2.1.2 Goals of Precision Farming system

The main goals of precision farming are:

- Improving water quality by targeting areas for reduced nutrient and pesticide application.
- Enhancing soil quality through reduced erosion and soil compaction.
- Emerging conservation through accurate and efficient application of crop inputs.
2.1.3 Functionalities of Precision Farming system

Basic functionalities of precision farming system are [48]:

- Support for data collection by services organizations
- Support for data collection by farmers
- Data access for farmers
- Support for fertilization application by service organization
- Yield forecasting for farmers

2.1.4 Basic Tasks of Precision Farming

The basic principle of precision farming technology is an exact positional controlling of fertilization with accuracy up to a few meters [48]. Such that process requires a big amount of data to be collected that enables controlling of the whole process. The total process of precision farming can be divided into:

- Data capturing
- Data access
- Data analysis
- Variable applications of fertilizers
- Variable applications of pesticides
- Variable seeding

This process can be understood as a cyclic optimization process, with some control points. Quality of the process can be measured by using these control points. Different models of optimization are used [48]:

- Minimum cost
- Maximum economy
- Optimization of expenses
- Ecological effects
2.1.5 Basic required technologies in Precision Farming

Basic required technologies in precision farming system are:

- PC computers
- Hand held computers
- Notebooks
- Board computers in farm machines
- Net communication
- Internet
- Mobile phones
- GPS (Global Positioning system)
- GIS (Geographic Information System)

2.1.6 Requirements of wireless technologies in Precision farming

The use of wireless technologies is not a new concept. In the world around us, different types of wireless technologies were already developed and many industries/companies are developing new technologies like Mobile WiMAX and LTE by enhancing the data rate and the coverage area with reducing the cost. In agriculture, types of wireless technologies are being used in different ranges:

- IrDA (Infrared Data Association) uses infrared light for short range, point-to-point communications.
- WPAN (Wireless Personal Area Network) for short range, point-to-multi-point communications, such as Bluetooth and ZigBee.
- WLAN (Wireless Local Area Network) for mid-range, multi-hop.
- Long-distance cellular phone systems, such as GSM/GPRS and CDMA [67].

Among them, the standards for wireless LAN, IEEE 802.11b (WiFi) [42] and wireless PAN, IEEE 802.15.1 (Bluetooth) [43] and IEEE 802.15.4 (ZigBee) [44], are most widely used for measurement and automation application. Wireless communication standards are classified in figure 2.1. All these standards use radio bands, including the MHz bands of 868-870 MHz (Europe), 902-928 MHz (US), 433.05-434.79 (US and Europe) and 314-316 MHz (Japan) and the GHz bands of 2.400-2.4835 GHz (worldwide acceptable). Radio waves with a lower frequency allow a longer transmission range and easily absorbed by various materials, such as water and trees. Comparison of features between LAN, Bluetooth and ZigBee are as shown in Table 2.1 [67].

Figure 2.1: Wireless Communication techniques.

<table>
<thead>
<tr>
<th>Feature</th>
<th>WiFi (IEEE 802.11b)</th>
<th>Bluetooth (IEEE 802.15.1)</th>
<th>ZigBee (IEEE 802.15.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radio</td>
<td>DSSS</td>
<td>FHSS</td>
<td>DSSS</td>
</tr>
<tr>
<td>Data rate</td>
<td>11 Mbps</td>
<td>1 Mbps</td>
<td>250 kbps</td>
</tr>
<tr>
<td>Latency</td>
<td>Up to 3s</td>
<td>Up to 10s</td>
<td>30ms</td>
</tr>
<tr>
<td>Range (m)</td>
<td>100</td>
<td>10</td>
<td>70</td>
</tr>
<tr>
<td>Extendability</td>
<td>No roaming</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2.1: Comparison between wireless LAN, Bluetooth and ZigBee.

2.1.7 Communication system architecture

Communication system architecture is shown in figure 2.2. This architecture consists of PDA, Mobile PC, GSM/GPRS Provider, Central Computer, No Wireless Clients and Internet Provider. Communication between all of them is illustrated as follows:

- PDA (Personal Device Assistant) is also known as packet PC. This requires internet browser for accessing/updating information in the field.
- Mobile PC is any laptop. This requires Windows OS, Internet browser, GIS tools and GPS.
- Central Computer is used to store data. This is an Internet and database server of the organization. Requirements of central computer are as follows:
  - Windows NT/2000
  - Internet server Apache
  - Map server and Map script
  - Geomedia webmap applications
  - Mapserver applications
  - GIS data collection
- EUROTEL, OSKAR, PAEGAS are GSM/GPRS providers.
- Internet provider requires WAP applications and Mapserver applications.
• No wireless clients are Personal computers. Requirements of PCs are:
  - Internet browser
  - Windows OS
  - GIS tools
  - Network Card
  - Modem for Internet Access.

• For the wireless communication, the server (central computer) is connected to a GSM modem with the datarate of 9.6/14.4 kb/s.

• For the wired communication, the central computer of the organization is connected to an ISDN modem.

• Wireless communication requirements:
  - GSM Phone (IrDA Port or Serial cable)
    * Eurotel provides datarate of 14.4 kb/s
    * Paegas provides datarate of 9.6 kb/s
    * Oskar provides datarate of 9.6 kb/s
  - GPRS provides datarate of 20 to 115 kb/s

2.1.8 Reasons for not considering GSM based positioning

This section discusses or expresses the information about the reasons for not considering the GSM based positioning by taking or considering two types of GSM related positioning technologies such as self-positioning and remote-positioning individually.

GSM self-positioning:
• It is expensive to provide GSM remote coverage in remote areas, because GSM tends to operate with very few base stations in these areas.
• The accuracy is unlikely to be as good as Differential Global Positioning System (DGPS) [24].

GSM remote positioning:
• There is limit to the number of users that GSM remote positioning can support.
• Poor coverage in rural areas.
• The accuracy will not be as good as Differential Global Positioning System (DGPS) [24].

Summary: This sub-chapter has summarized the basic concept of precision farming and mainly concentrated on principles including tasks, architecture and functionalities. The GIS, GPS and wireless technologies plays major role in precision farming system for data collection and data processing. Finally, this sub-chapter summarizes that the GPS based positioning is more precise than GSM based positioning and works in areas which do not have GSM coverage as well.
Figure 2.2: Communication System Architecture.

2.2 MOBILE GIS

This sub-chapter summarizes the basic principles and the related state-of-the-art of the GIS technology with a specific focus on mobile GIS.

2.2.1 Introduction to GIS

Geographic Information System (GIS) can be looked at as information system that is used to store, analyze, capture and manipulate geographic information for decision making. GIS technology started in 1950s and is seen in many applications such as land surveying, remote sensing, aerial photography, health and country planning. Development of GIS has evolved into internet and now mobile GIS today [91].

Definition of GIS: A system of hardware, software, and procedures designed to support the capture, management, manipulation, analysis, modeling and display of spatially-referenced data for solving complex planning and management problems. (David Cowen, 1988) [19].

2.2.2 Key Components of GIS

Five major key components integrated by the GIS are:

- **Hardware:** GIS operates on wide range of hardware types, in which hardware is the computer where GIS is operated. The hierarchy of these hardware types range from centralized computers from servers to desktop PC’s which are used in network configuration [81].

- **Data:** The most important requirement of GIS is acquiring data. A GIS mostly uses the DBMS for manage and organize the data.

- **People:** The power of GIS technology is increased by the users who uses and manages it. Over the worldwide, GIS has wide range of users from technical specialists who design it to the end users.

- **Software:** GIS software provides or produces the tools and functions needed to store, display, and analyze geographic information. The following involved software components are:
  - A Database Management System (DBMS)
  - Tools for manipulation of geographic information
  - A Graphic User Interface (GUI) for easily accessing the tools.

- **Methods:** Methods include how the data will be retrieved, managed, input into the system, stored, transformed, analyzed, and finally presented in a final output. Each organization which operates successful GIS has well designed plan and business rules considering these like a model and operating practices which are unique to one’s organization [81].
2.2.3 Key Functions of GIS

The basic key functions of GIS characterize the steps that have to be taken to develop or implement a GIS. The following steps are:

- **Data Capture:** GIS uses the data that often come from many sources. Data sources are mainly obtained from paper maps, Manual Digitization and Scanning of aerial photographs, and existing digital data sets. GPS is the promising data input source for GIS. The process of converting paper maps into numerical digits is called as digitization. Digitizing simplifies map data into sets of lines, points, or cells that can be stored as red in the GIS computer [78].

- **Data Compilation:** After the process of conversion, the user is done with the compilation phase by relating all spatial features to the respective attributes. After the compilation phase, the digitally converted data will be cleaned up and correcting errors which results in the data conversion process.

- **Data Storage:** After the compilation phase, the converted digital map files are stored on digital or magnetic tapes. The conversion of map data into a digital form is based on a generic data model for the data storage. Vector and Raster are the two most common types of data models.

2.2.4 GIS Operations

There are three major types of operations in a GIS. The following involved operations are:

- **Access methods:** In general, user wants to access the data with certain properties and follow the rules before accessing the data, they are:
  - The requested data is available in the system will be checked.
  - How long it will take to find the data.
  - How users access the data (mostly used conventional query language SQL is used).

- **Analytical operations:** This allows user to combine data to produce new data. The overlay of different spatial data sets to produce a new data set is the most important operation in GIS.

- **Rendering:** Rendering operations show the data as some sort of map with the additional information displayed [70].

2.2.5 What is Mobile GIS

It is an extension of GIS technology from the office to field. Mobile GIS enables the field-based personnel to store, capture, manipulate, update, analyze, and display geographic information [28]. Mobile GIS combines or integrates one or more of the following technologies:

- Global Positioning System (GPS)
• Mobile devices
• Wireless communication for Internet GIS access

2.2.6 Why Mobile GIS

In general, the process of collecting field data and editing have been time consuming and error prone. The geographic information data is travelled in the form of paper maps into the field. Appending of fields were performed using notes and sketches on paper maps and forms. These field edits were comprehend and manually entered into the GIS database, once back in to the office [28]. A result in GIS database has often not been as up-to-date.

Recent developments in the mobile GIS have made many companies or industries to take the digital maps on mobile computers which provide access to enterprise geographic information to the field. Mobile GIS enables users or organizations to add real-time information to their database and applications, display, speeding up process or analysis, and decision making by using up-to-date.

2.2.7 Tasks of Mobile GIS

Surveyors, Firefighters, engineering crews, soldiers, field biologists, utility workers, Police officers, census workers, and others, use mobile GIS to finish the following tasks:

• Field Mapping: Create, edit, and GIS maps are used.
• Asset Maintenance: Update asset location and schedule maintenance and condition.
• Asset Inventories: Inventory of asset locations and attributes are created and maintained.
• Incident Reporting: The incidents and events for future actions is reported by the documenting the location.
• Inspections: The legal code compliance and ticketing is done by maintaining digital records and location of field records.
• GIS Analysis and Decision Making: Perform measuring, geo-processing, buffering, and other GIS analysis while in the field.

2.2.8 Benefits of Mobile GIS

Four major benefits rather than GIS are:

• Increased data accuracy
• More accurate decisions
• Better information accuracy
• Improved productivity
2.2.9 Key components of Mobile GIS

Key components of Mobile GIS are [93]:

- Mobile devices
- GIS editing tools
- Local maps and data
- Accurate location
- Spatial analysis
- Wireless connectivity

2.2.10 Generic Architecture of Mobile GIS

The architecture of mobile GIS is very similar to the wired-based Internet GIS and it extends the scope of server-based Internet GIS applications. Therefore, server-based applications have been developed with both wired and wireless clients.

The general architecture of mobile GIS is shown in figure 2.3 [72]. This architecture includes three types of elements.

- The presentation element is the user interface on an Internet-enabled hand-held device.
The business logic elements include a base transceiver station, a mobile switching center, a gateway service, a web server, a GIS server and other server applications.

The data element includes a location content server (data server) and database sources.

Working process of architecture is illustrated as follows:

- The user with an Internet-enabled mobile device is a wireless network (GSM, GPRS, etc.) access any information by sending a request through a URL.

- The mobile device first verifies whether it has already an open connection with the telecommunication service provider; if not, it dials up the modem attached to a dial-in server (Remote Access Service) in the base transceiver station [72].

- The Positioning-Determining Equipment (PDE) in the base transceiver station determines the location of the mobile device.

- The positioning information is processed and maintained by the mobile switching center. This dial-in server gives the mobile device access to the protocols it needs and assigns it an IP address. The request for the URL is sent to the gateway service.

- The gateway service maintains the wireless connection between the Internet-enabled mobile device and content provide servers. The presence of the gate-way service is a major difference between the wired Internet GIS and the mobile GIS. The gateway service acts as a translator between the mobile device and the web server, because it translates the requests from the mobile device to HTTP requests for the web server and also converts the response from the content provider and web server to a format that is readable by the mobile devices.

- There is a normal web server at the content service provider side. This server links with the GIS server (information process server), location content server and databases. Sometimes, the web server and the gateway service are combined as a "gateway server". The web server, depending on which type of browser the requests come from, sends out contents in different formats. For example, a browser inside the mobile device is called as microbrowser and a browser from a PC is called as web browser.

- When the web server returns the contents provided by the GIS server and/or content provider server, the gateway service has to compile them into the format that can be understood by the mobile device and can minimize the bandwidth usage. The transformed content is then passed back to the mobile device for display [72].

- Finally, the built-in microbrowser inside the Internet-enabled mobile device receives and reads the file and displays the contents on the mobile device for the user. This is the process that majority of mobile devices are connected to the web servers and GIS servers on the Internet.

**Summary:** This sub-chapter has summarized the basic principles of Geographic Information System (GIS) such as key components, key functions, and operations of GIS and
mainly concentrated on concepts of mobile GIS. Mobile GIS enables users to add real-time information to their database and applications, display, speeding up process or analysis, and decision making by using up-to-date. Finally, this sub-chapter has described the process of mobile devices are connected to the web servers and GIS servers on the Internet.
2.3 Positioning Technologies

This sub-chapter summarizes the basic principles and the related state-of-the-art of the positioning technologies with a particular focus on GPS and GSM based positioning.

2.3.1 Definitions and terms

- **Position**: A position is defined by a set of coordinates related to a well-defined coordinate reference frame. Every reference frame requires a convention on its origin and orientation of its coordinate axes. The process of determining a position is called position determination [9].

- **Location**: Location describes a position in terms of topological relations. The process of determining a location is called location determination [9].

- **Positioning System**: A positioning system is a mechanism for determining the location of an object.

- **Lateration**: Calculating the position of an object by measuring its distance from multiple reference positions.

- **Angulation**: Calculating the position of an object by measuring the angles with which it sees multiple reference positions.

- **Trilateration**: Calculating the position of an object by measuring its distance from three reference positions.

2.3.2 Positioning methods

This section describes the different positioning methods used to get the location of people, device, object and so on. A general classification of positioning methods can be done into two groups:

- **Network-based positioning**: In this positioning method, tracking and evolution of the user location is done by using the base station network. Therefore the mobile device sends either a signal or receives a signal which is sensed by the network.

- **Terminal-based positioning**: In this positioning method, the location is calculated by the user device itself using Global Positioning System (GPS).

The following possible techniques are mostly used for positioning in combination:

- **Cell of Origin (COO), location signature, location beacons**: The cell id is usually the identifier of the nearest base station, e.g. mobile phone antenna. With this technique, it is possible to get the known position in a defined cell or circle around the base stations known position. Beacons (e.g. infrared, ultrasound or RFID) are mostly used indoors [90]. They transmit their exact position to the mobile device by using identifier id.
• **Time of Arrival (TOA):** In general, electromagnetic signals are usually moved with light speed. Knowing the speed and the time difference between sending and receiving the distance can be measured. Light speed is very short and exact timers are needed. This same principle is applicable for slower signals like ultrasound.

• **Time Difference of Arrival (TDOA), Enhanced Observed Time Difference (E-OTD):** These two techniques are also used to compute the distance by measuring the runtime. But here the time difference between the signals of usually three different base stations has to be considered, such that they receive signals from different neighboring base stations to triangulated position. In case of TDOA, the position is determined by the network provider. In contrast with E-OTD the calculation of the position is done in the mobile device.

• **Angle of Arrival (AOA), Direction of Arrival (DOA):** The angle of arrival in the mobile device can be detected by using antennas with direction characteristics. This is exactly possible with many base stations with segment antennas which divide the circum-circle of the base station into segments of 90, 120 or 180 degrees.

### 2.3.3 Types of Positioning Systems in ITS

The positioning system can be classified based on its functions into two types self-positioning and remote positioning. In self-positioning, the object itself actively determines its own position. The GPS is a most important example of a self-positioning system. In remote positioning, the determination of position is the responsibility of a remote facility in which the object is not directly involved, for e.g. Radar for remote positioning. The basic classes of positioning system can be divided into three ways: signpost, wave-based, and dead reckoning.

**Signpost System:** Signpost system is the simplest form of positioning system. This system measures position by virtue of the fact that the vehicle is located close to a specific reference point, which is known as a signpost. Automated versions of these signpost systems are commonly referred to as proximity beacon systems. This system works on the basis of reception of light, sound, or radio waves. New South Wales government’s Roads and Traffic Authority’s Automatic Network Travel Time Measurement system (ANTTS) is an example of such systems. The vehicle - mounted "tag" and the roadside unit are the two most important elements of these systems.

The vehicle has a tag that picks up the signal from the beacon, in case of the system can be self-positioning. This signal will normally have some identification code. Therefore the vehicle knows to which beacon it is "in proximity".

The system can also be remote positioning, if the beacon senses the presence of the tag on the vehicle. In this case, some form of identification code can be emitted by the tag in order that the system knows which vehicle is being interrogated.
**Wave-based System:** This system uses the propagation properties of waves to determine position. Radar system is an example that uses the far-field planar wave front and the finite propagation time of electromagnetic waves. A wave-based positioning system will need one or more reference sites. Each reference site may have a transmitter or receiver, or both. Each vehicle may have a receiver, transmitter, reflective elements, or some combination of those.

For example, in the case of GPS, each satellite is a reference site and there is a transmitter on each satellite. A GPS receiver on the mobile vehicle picks up the signals from the satellites and uses time-of-arrival (TOA) information to calculate position. In a simple radar system, there is one reference site that fitted with both a transmitter and a receiver. The "targets" reflect the radio energy back to the reference site [24].

**Dead Reckoning System:** This is a system that rely on sensing the components of the vehicle’s velocity or acceleration information. This information can be then integrated to obtain the track of the vehicle. For example, consider a compass and odometer system. In order to perform the estimation of the distance traveled, the odometer integrates the angular velocity of the vehicle’s wheels. The direction of travel can be defined by the compass. The combined information can be used to track the course of the vehicle.

### 2.3.4 Introduction to GPS

GPS stands for Global Positioning System. In the early 1970s, the GPS was developed by the U.S Department of Defense (DOD) which is a satellite-based navigation system. Initially, it was used for military purposes, but in the year of 1980, U.S. government decided to make the system’s positioning data freely available to all over the world. Since then, most of the companies or industries have taken up the opportunity to access positioning data through GPS and now they use it to improve or enhance their services and products [45].

**Overview of GPS:** GPS is a constellation of 24 satellites and they are orbiting the earth at about 12000 miles. This constellation known as the initial operational capability (IOC) and that was finished in July 1993. The official announcement was made on December 8, 1993 [5]. Four GPS satellites are placed in each of six orbital planes to ensure continuous worldwide coverage. The GPS system was officially declared on July 17, 1995, to have achieved full operational capability to ensure the availability of at least 24 operational GPS satellites.

**Basic idea of GPS:** Each GPS satellite continuously transmits a signal composed of a navigation message, two digital codes, and two frequency carriers. In the switch on mode, the GPS receiver will pick up the signal through the receiver antenna. GPS receiver
processes the GPS signal using built-in software. The signal processing output consists of
the satellite coordinates through the navigation message and the distances to the satellites
through the digital codes. Three satellites must be used to uniquely determine the GPS’s
location relative to the satellite [47].

2.3.5 GPS Segments

GPS technology requires three segments. They are space segment, control segment and
user segment [99].

Space Segment: This segment consists of constellation of 24 satellites which orbit the
earth twice a day in a specific pattern. They travel at approximately 7,000 miles per hour
about 12,000 miles above the earth’s surface. These satellites are spaced so that a GPS
receiver anywhere in the world can receive signals from at least four of them. Each GPS
satellite constantly sends radio signal to the earth. These signals contain the following
information:

- Two digital codes
- Two sine waves (carrier frequencies)
- A navigation message

GPS receiver uses the digital codes and carrier frequencies to calculate the distance from
the user’s receiver to the GPS satellites. The GPS satellites are powered by solar energy.
If solar energy is unavailable, for example, when the satellite is in the earth’s shadow, the
satellites use backup batteries to continue running. Each GPS satellite is built to last
about 10 years. The Department of Defense monitors and replaces the satellites to ensure
that GPS technology continues to run smoothly for years to come [33].

Control Segment: This segment is responsible for constantly monitoring satellite health,
signal integrity, and orbital configuration from the ground. The control segment includes
the following sections:

Monitor Stations: At least six monitor stations are located around the world. Each
station constantly monitors and receives information from the GPS satellites and then
sends the orbital and clock information to the Master Control Station (MCS).

Master Control Station (MCS): Located in the U.S. at Colorado Springs, Colorado [20]. The MCS constantly receives GPS satellite orbital and clock information
from the monitor stations. The controllers in the MCS make precise corrections to the
data necessary and send that correct information to the GPS satellites using the ground
antennas [33].

Ground Antennas: Receive the corrected orbital and clock information from the MCS
(Master Control Station) and then send the correct information to the appropriate satel-
lites.
**User Segment:** This consists of GPS receivers that collects and processes signals from the GPS satellites those are in view and then uses that information to determine and display location, speed, time, and so forth. GPS receiver does transmit any information back to the satellites.

### 2.3.6 Working process of GPS Technology

As discussed already, each GPS satellite continuously transmits a signal. GPS receiver on the earth accurately determines its location (longitude, latitude, and altitude) in any weather, any where on the earth.

**Calculating position:** A GPS receiver needs to know the precise time to calculate its position. The receiver determines the signal transit time by comparing its internally generated code (at the receiver clock time of reception) with the code sent by the satellite (at the satellite clock time of transmission). With this time, the receiver calculates the distance to each satellite. GPS system applies trilateration principle to calculate the distance. Formula for calculating the distance can be considered as speed of light multiply with time of flight (distance = speed of light x time of flight).

The calculation of distance to each satellite from the receiver is very simple but in real life the range (distance) measurement is corrupted with a number of imperfections. The following relevant imperfections are:

- The receiver internal clock is not synchronized with GPS time. Adding a range (distance) measurement to a fourth satellite allows the user to solve for a receiver clock offset.
- The satellite internal clock is not synchronized with GPS time. The clock parameters in the navigation message allow calculation of this offset.
- The signal speed in the atmosphere is lower than the speed of light in a vacuum. Models for the tropospheric delay and ionospheric delay allow correction of these delays [97].

Incorporating the above effects, the range (distance) equation becomes:

$$PR_i^{(corr)} = \sqrt{(X_i - X_u)^2 + (Y_i - Y_u)^2 + (Z_i - Z_u)^2 - C_u}$$  \hspace{1cm} (2.1)

where $PR_i^{(corr)}$ is the corrected pseudorange measurement to satellite i. $X_i, Y_i, Z_i$ are the geocentric position coordinates of satellite i. $X_u, Y_u, Z_u$ are the geocentric position coordinates of the user(receiver). $C_u$ isthe receiver clock offset in meters.
The relation between the corrected and measured pseudorange is:

\[ PR^i_u (corr) = PR^i_u (measured) - C^i - I^i_u - T^i_u \]  \hspace{1cm} \text{(2.2)}

Where \( C^i \) is the clock offset of satellite 'i' in meters.
\( T^i_u \) is the tropospheric delay.
\( I^i_u \) is the ionospheric delay.

Equation 2.1 can not be solved directly. The equation must be linearized by developing them into Taylor series and taking the first order terms:

\[ PR^i_u = R^i_u (0) + \left( \frac{X^i - X_u (0)}{R^i_u (0)} \right) \Delta X + \left( \frac{Y^i - Y_u (0)}{R^i_u (0)} \right) \Delta Y + \left( \frac{Z^i - Z_u (0)}{R^i_u (0)} \right) \Delta Z - C_u \]  \hspace{1cm} \text{(2.3)}

Where \( \Delta X_u (0), \Delta Y_u (0), \Delta Z_u (0) \) are an initial guess for the coordinates of the user.
\( R^i_u \) is the range between the satellite i and the user based on the initial user coordinates.
\( \Delta X, \Delta Y, \Delta Z \) are the corrections to the initial user coordinates.

Equation 2.3 can be rearranged to give:

\[ \left( \frac{X^i - X_u (0)}{R^i_u (0)} \right) \Delta X + \left( \frac{Y^i - Y_u (0)}{R^i_u (0)} \right) \Delta Y + \left( \frac{Z^i - Z_u (0)}{R^i_u (0)} \right) \Delta Z - C_u = PR^i_u - R^i_u (0) \]  \hspace{1cm} \text{(2.4)}

The above relation in a more convenient matrix notation is:

\[ A \ast D = L \]  \hspace{1cm} \text{(2.5)}

Where \( A = \begin{bmatrix} A^i_1 & A^i_2 & A^i_3 & A^i_4 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ A^n_1 & A^n_2 & A^n_3 & A^n_4 \end{bmatrix} \)

\[ A^i_1 = \left( \frac{X^i - X_u (0)}{R^i_u (0)} \right) \]
\[ A^i_2 = \left( \frac{Y^i - Y_u (0)}{R^i_u (0)} \right) \]
\[ A^i_3 = \left( \frac{Z^i - Z_u (0)}{R^i_u (0)} \right) \]
\[ A^i_4 = -1 \]
\[ D = (\Delta X, \Delta Y, \Delta Z, C_u)^T \]

\[ L = (PR^1_u - R^1_u (0))^T, \ldots, (PR^n_u - R^n_u (0))^T \]
\( n = \text{number of satellites.} \)
'D' in Equation 2.5 can be solved using matrix algebra techniques and used to update the initial user position \(X_0, Y_0, Z_0\). 'A' is then recalculated and 'D' is solved again. This process is iterated until the position corrections 'D' is close to zero \([97]\). This provides the final estimates of the \(X_0, Y_0, Z_0\) user positions, and the \(C_u\) is receiver clock error.

2.3.7 The GPS Error Budget

The GPS system has been designed to be as nearly accurate as possible. However, there are still errors. These errors can cause a deviation of +/- 50 to 100 m from the actual GPS receiver position. There are several sources for these errors; some of them are discussed here:

**Atmospheric Conditions:** The ionosphere and troposphere both refract the GPS signals. This causes the speed of the GPS signal in the ionosphere and troposphere to be different from the speed of the GPS signal in space. Therefore, the distance calculated from "Signal Speed x Time" will be different for the portion of the GPS signal path that passes through the ionosphere and troposphere and for the portion that passes through space \([62]\).

- Ionosphere can introduce 0-30 meters of positional error
- Troposphere can introduce 0-30 meters of positional error

**Ephemeris Errors/Clock Drift/Measurement Noise:**

GPS signals contain information about ephemeris (orbital position) errors, and about the rate of clock drift for the broadcasting satellite. This data concerning ephemeris errors may not exactly model the true satellite motion or the exact rate of clock drift \([62]\). Distortion of the signal by measurement noise can further increase positional error.

- The disparity in ephemeris data can introduce 1-5 meters of positional error
- The clock disparity can introduce 0-1.5 meters of positional error
- The measurement noise can introduce 0-10 meters of positional error

**Selective Availability:**

This is the intentional alteration of the time and ephemeris signal by the Department of Defense.

- "Selective availability can introduce 0-70 meters of positional error

**Multipath:**

A GPS signal bouncing off a reflective surface prior to reaching the GPS receiver antenna is referred to as multipath. This multipath error is a serious concern to the GPS user, because it is difficult to completely correct it, even in high precision GPS units.

- "Multipath can introduce 0-1 meter of positional error.
2.3.8 DGPS Technology

The GPS accuracy was good in the early days of it to within 100 meters for a given location. This was due to the U.S government’s policy of "Selective Availability". But this selective availability introduced a timing error in the GPS signal. As a result the government has dropped this policy. Since then the accuracy of GPS units have improved tremendously. Most GPS units today are accurate to within 5-10 meters. This is good enough for most user applications such as road navigation, etc. For certain advanced applications, such as maritime navigation and aircraft navigation, even more accuracy is desirable [20]. For this a more accurate GPS system is required and one of the solutions is Differential GPS or DGPS.

**Working process of DGPS:** The working process of DGPS technology is as follows:

- The DGPS system works by placing a high-performance GPS receiver (reference station or base station) at a known location.

- Since the base station receiver knows its exact location, it can determine the errors in the satellite signals. It does this by measuring the ranges to each satellite using the signals received and comparing these measured ranges to the actual ranges calculated from its known position.

- The difference between the measured and calculated range is the total error. The error data for each tracked satellite is formatted into a correction message and transmitted to GPS users.

![Figure 2.4: Differential GPS System.](image)

2.3.9 RTK - GPS Technology

The DGPS accuracy is not enough for certain precision farming applications such as row crop bed preparation and planting and so forth. That’s where centimeter solutions using
RTK (Real-Time Kinematic) technology gives greater accuracy, as well as huge savings in time and money. This RTK receiver was designed by the Trimble Company and mainly built for agriculture applications [96].

**Principle of RTK - GPS:** RTK - GPS technology is used for quick survey. Traditionally user will setup one GPS receiver on control station with known coordinates as a reference station, and the rover receiver receive the same GPS signals at the same time. By using the continuous simultaneous observation and the reference station coordinate, we can solve the integer ambiguity of the carrier phase observations. GPS Reference stations can deliver traditional RTK data. User can connect to the Data Centre using PDA with GPRS connection, and obtain the required RTK data of the relevant reference stations on the PDA user interface [92].

![Figure 2.5: Real-Time Kinematic GPS.](image)

**2.3.10 Components description of GPS receiver**

This section interprets the generic block diagram of GPS receiver including components and briefly shows the information about components.

**Antenna and Preamplifier:** The antennas that are used for GPS receivers have broad beam characteristics. Like the traditional satellite TV receiving dishes there is no need of pointing the antennas to the signal sources. The antennas are smart and a variety of designs are also possible. A new trend is implemented to integrate the antenna assembly with the receiver electronics.

**Radio Frequency and computer processor:** The radio frequency section includes the signal processing electronics. To process the signal, different mechanisms were implemented using different receiver types. There is a microprocessor on-board not only to carry out
computations such as determining the elevation/azimuth of the satellites and so on, but also to control the tracking and measurement function within digital circuits [24].

**Control unit interface:** To interact with the microprocessor the operator is enabled by the control unit. The size and type of the control unit is dynamic. The dynamic property varies greatly for different receivers, ranging from a handheld unit to soft keys, which is surrounded by a LCD screen fixed to the receiver box.

**Recording device:** The process of surveying is mainly done by the GPS receivers where the measured data must be preserved such that it is useful for the future work like data processing. The large variety of ITS applications like logging of vehicle movement is done by these GPS receivers which record the derived coordinates and velocity. A variety of storage devices were used in the past, including tape records, computer tapes, floppy disks, and cassette, and so on, but in the earlier days almost all receivers utilize RAM or removable memory cards.

**Power supply:** The power consumption of the GPS receiver is very low because it consumes low voltage dc power and these receivers are also movable. Most GPS receivers operate from a number of power sources, including internal lithium batteries or NiCad and external batteries as wet cell car batteries.

### 2.3.11 Cellular Networks based positioning systems

These are the radio-based mobile communication systems. The term cellular defines that the systems use many base stations to receive or transmit the signals from the mobile
Advantages of a cellular telephone positioning system are:

- Cellular systems already have a spectrum.
- There are large numbers of installed user base.
- The cellular system provides a two-way communication link.

Disadvantages of a cellular telephone positioning system:

- The bandwidths tend to be narrower than is optimal for a positioning system designed for the higher accuracy ITS applications.
- Cellular systems tend to be designed so that it is only necessary for one base station to pick up the signal from a mobile.

Positioning in the GSM Network:

GSM network is composed of adjacent cells and each cell can cover an up to 20 kilometers, while all cells together can cover very large physical area. To distinguish cells from each other, the operator has given each cell a unique id number. Positioning methods in GSM network can be divided into three types:

- Network based method
- Network oriented method
- GPS based method

Network based methods include location estimation methods that are provided by standard equipment of GSM network infrastructure. All methods in this type are based on recognizing id number of GSM cell that is currently serving the user. If physical location of base station of the cell is known, the user's location can be estimated according to the cell size, if the cell size is known. Accuracy of the location estimation is based on cell size, which may vary from some hundred meters to tens of kilometers.

GSM network consists of data that can be used for providing more accurate estimates about the user's location in cell-id based positioning. The distance (range) between the GSM base station and the terminal device may vary a lot due to the user's mobility. The mobile terminal uses timing advance (TA) parameter to avoid radio signal overlapping of the multiple simultaneous transmitters in the time domain. Value of the timing advance corresponds to the approximation of microseconds the signal from mobile phone requires to travel to the base station. Respectively, value of TA parameter corresponds to the distance between the base station and the mobile terminal. Thus, timing advance can be utilized in
determining location of the mobile terminal more accurately than plain cell-id \cite{83}. This estimation can be further adjusted by using received radio signal level parameter.

In order to enable more accurate positioning methods, the GSM infrastructure can be updated with location measurement units (LMUs).

- The network oriented positioning methods include time of arrival (TOA), angle of arrival (AOA) and enhanced observed time difference (E-OTD).
- GPS based methods utilize GPS positioning and they require specialized hardware on the terminal device. GPS positioning can be done in stand-alone mode with external or integrated GPS receiver, which is connected to the mobile terminal.
- The problem with GPS positioning is that does not work in urban areas or inside the buildings. To avoid this problem, assisted GPS (A-GPS) can be used to provide positioning, if mobile operator network and mobile terminal supports it.

GSM network illustrates in figure \ref{fig:2.7} which is composed of several cells. Each cell has a base station, which acts as a network access point for the mobile terminals in the cell.

- As users move from one cell to another, a handoff takes place at the border of the two cells (marked as dots in Fig.2.7). If physical locations of base stations are known, it can be used to determine that user’s location is the same than location of the base station.
- However, this pure cell based positioning assumes that the mobile terminal is connected to the nearest base station, but it is reported to be true in only 57 percentage of the cases \cite{83}.
- This is due to fact that in GSM system, the handoff is determined by the network, not by the terminal device. For example, if some part of the network suffers high congestion, the network may force terminals to move to another cell.

According to figure \ref{fig:2.7} out of huge population, there are some users that carry GPS enabled mobile terminal. As the users move, every time when serving GSM cell changes, user’s mobile terminal connects to the network server and creates a log with following data: id of the cell where user was coming from, id of the cell where user arrived, coordinates of the handoff location (longitude, latitude). During that time there will be enough data available that the physical locations of each cell can be determined. Thus, when a user with no other positioning mechanism (GPS etc.) enters to the area, the outer bounds of geological region where the user is located can be calculated from data in the database \cite{83}.

One of the challenges in any cell based positioning methods is that they assume GSM network to be static over time. However, in real life the network can be modified in various ways.

- New cells may be included to the areas where amount of users is high. On the other hand, some cells may be combined or dropped out.
• Locations, where handoff under normal circumstances takes place may also be altered due to network optimization, which affects to the cell-based positioning performance.

• There is always a danger that changes and modifications in the network break the earlier calibration.

2.3.12 GSM positioning versus GPS positioning

GSM positioning:
• Utilize the GSM network
• Gives us a macro location
• Works inside buildings
• Works within cargo

GPS positioning:
• Uses satellites to find position
• Gives us good accuracy
• Does not work inside buildings
• Does not work in covered places

Summary: This sub-chapter has digested the basic principles of positioning technologies and classified the different ways to determining the position. In order to obtain the position, GPS has several advantages compare to GSM, because GSM provides poor coverage in rural areas.
2.4 Mobile Applications and/or Location-Based Services (LBS)

This sub-chapter summarizes the basic principles and the related state-of-the-art of the so-called mobile applications (and/or location-based services).

2.4.1 Introduction to LBS

A location-based service is any information that is available on mobile devices and makes use of geographical information. In other words, these are the information services accessible with mobile devices through the mobile network and utilizing the ability to make use of the location of the mobile devices. (Virrantaus et al. 2001) [90].

Normally, location information can be described or expressed in spatial terms or text descriptions. Spatial location information can be formulated or expressed in the widely used longitude-latITUDE-altitude coordinate system [57]. Longitude is formulated as 0-180 degrees east or west of the prime meridian, which passes through Greenwich, England, and latitude as 0-90 degrees north or south of the equator. Altitude is formulated in meters above sea level. Text description usually described as a street address, including postal code, city, and so on.

2.4.2 Concepts in LBSs

Position:

- Position appears to developers in the form of spatial coordinates.
- It can be represented as a single point in the Cartesian coordinate.

Location:

- Location is associated with a certain place in the world.
- If positioning delivers a spatial location, it will be mapped onto a descriptive location in order to be interpretable by the LBS user.

Location Service (LCS):

- Location service is distinguishable from location based service as it exclusively deals with the localization of target, and also makes the resulting location data available to external actors [87].
- It is responsible for the generation and delivery of location data.

Location Based Service (LBS):

- Location based service that adds value to target locations provided by location service. It uses knowledge of a mobile devices location to offer value to the mobile subscriber or to a third party [87].
2.4.3 What are LBSs

This section classifies the location-based services and describes major characteristics of them. The following location-based services are:

- The relation of GIS and LBS
- LBS components
- Push and Pull services

The relation of GIS and LBS:

GIS has been developed during several decades on the basis of professional geographic data applications, but LBS were born quite recently by the evolution of public mobile services. GIS can be observed as professional systems intended for experienced users with wide range collection of functionality and also require extensive computing resources [90]. In case of LBS, they are developed as limited services for large non-professional user groups. LBS applications are usually operating with the restrictions of mobile computing environment like small display or screen, low computational power, and small battery run time of the mobile device.

LBS Components: This sub-section characterizes the major key components of LBS that requires to the user to use a location based services. The following five major key components of LBSs are:

Mobile Device: Mobile device is a tool for the user to request the needed information. Typical devices are PDA’s (Personal Device Assistants), Mobile phones, Laptops, and so on [82] [63].

Communication Network: Transfers the user data and service request from the mobile device to the service provider and the acknowledged service sends to the user.

Positioning Component: Position of the user has to be obtained for the processing of a service. The user position can be determined either by using the GPS or by using the mobile communication network. Further possibilities to gain the position are radio beacons, WLAN stations [63].

Service and Application Provider: The main task of service provider is to provide or offer a variety number of different services to the user. Such services offer the computation of the position, finding a route and so on.

Data and Content Provider: Normally, all the information requested by the users will not be stored and maintained with the service providers. Therefore geographic data and location information data will be requested from the concerned authority.
**Push and Pull Services:** This sub-section distinguishes two kinds of location services that are considered by a user to know whether the information is delivered on user interaction or not:

**Pull Services:** Pull services deliver information directly requested from the user. An example for a directly requested service is to call a website in the Internet by giving its URL in the web browser [90].

**Push Services:** Push services deliver information indirectly requested from the user. An example for an indirectly requested service is a news subscription which contains event information with respect to the actual city directly from the service provider.

### 2.4.4 LBS communication model

Technologically, the realization of LBS can be described by a three-tier communication model. This communication model illustrates in figure 2.8 including a physical layer, a middleware layer, and an application layer [45].

![Diagram](image)

**Figure 2.8:** General LBS communication.

Process of work between all elements in the communication model is described as follows:

- The positioning layer is responsible for calculating the position of a mobile device or user with the help of position determination equipment (PDE) and geospatial data held in a geographic information system (GIS).

- While the PDE calculates where a device is in network terms, the GIS allow it to translate it to this raw information into geographic information (longitudes and latitudes).

- The end result of this calculation is then passed on via location gateway either directly to an application or to a middleware platform.
• The positioning layer would manage and send location information directly to an application that requests it for service delivery.

• The application layer comprises all of those services that request location data to integrate it into their offering. Due to the use of LBS applications increases, many network operators have put a middleware layer between the application layer and positioning layer [45].

• This middleware can significantly reduce the complexity of service integration because it is connected to the network and an operator’s service environment once and then mitigates and controls all location services added in the future.

Summary: This sub-chapter has digested the basic principles of location-based services and describes the basic concepts of it. Finally, this work has organized the communication model of LBS.
Chapter 3

Software Development Process and Programming Basics

This chapter deals with the basics of software development process and programming design concepts.

3.1 Basics of software development process

When building a house, at first, the builder analyzes and takes many factors into account such as requirements and possibilities. The architect considers these factors when designing the house. Only then, the construction of the house is started.

It is desirable to act in the same way when constructing or developing a software. First, analyze the problem to be solved and describe the requirements in a very precise manner. Based on these requirements, the design is formulated. Finally, the implementation process (programming of the solution) is started [95].

The basic workflow method for the software development process is shown in figure 3.1 and a brief description of each component is as follows:

3.1.1 Requirements Analysis

The main focus of requirements analysis is to get a complete description of the problem to be solved. The requirements analysis includes supporting software, hardware, and the number of users of the system to be developed. A description of the problem to be solved includes following things:

- Develop the functions of the software.
- Possible future extensions to the system.
- The amount and kind of documentation required.
- Response time and other performance requirements of the system.
Figure 3.1: A simple view of software development.
Feasible study is a part of the requirements analysis. The main purpose is to appraise if it is both technically and economically feasible. At the end of this requirement analysis, various people (perspective users, customer, programmers, and designers) have to collaborate intensively [98]. Communication between these people is not easy, because they are from different backgrounds. Requirement specification is the document, which shows the result of this activity.

### 3.1.2 Design

A model of the whole system is developed during the design phase. The design phase is crucial and the functions of these modules as well as the interfaces between them are specified in a very precise manner [98]. This phase is sometimes classified into two levels: high-level design and low-level design. The detailed interfaces between the various subsystems will be defined, during high-level design. At this level, the overall work can be divided into subprojects for the various teams. The low-level design involves the design of the various subsystems.

### 3.1.3 Implementation

This phase involves the development of various subsystems that are based on the earlier design documents. To produce a working system, various subsystems are integrated. The implementation process depends on the nature of the subsystem. If the project is based on hardware, there will be circuit board manufacturing and coding is used for software implementation.

### 3.1.4 Testing

This phase follows the implementation phase. Testing phase comes in two flavors. One is verification, which tests whether the transition from phase i to i+1 is correct or not. Another flavor is the validation, which checks the right track with regards to fulfilling user requirements. Measuring efficiency and optimization of software are also considered in this phase.

### 3.1.5 Maintenance

There are often errors that may come into picture, after the delivery of the software. Post delivery of the software, if there are errors/bugs which are realized while using it, then they have to rectify. In addition, changes and enhancements are requested by the actual system. All of these modifications are denoted by the rather unfortunate term maintenance. Maintenance concerns with all activities needed to keep the system operational after it has been delivered to the user [98].

There are possible life cycle models that are used for the phased development of software. Here, one famous model, namely waterfall model is to be discussed.

**The Waterfall Model:** This is the simplest form of software development life cycle model.
• First, the requirements analysis phase deals with specified problem along with the desired service objects (goals) and also for identification of constraints.

• Once the requirements analysis is finished, the design phase begins. In the design phase, the software engineer is concerned with software architecture, data structure, algorithms and interface representations.

• Implementation begins after the design phase is completed. Working with implementation phase is based on design phase. Because detailed documentation from the design phase can significantly reduce the coding effort.

• Once the implementation of programming is completed, the testing process begins. Testing software strictly follows the implementation phase and focuses on making sure that errors in the program are rectified.

• The system is installed, once finish the successful completion of testing. Then maintenance of the system takes place. Maintenance phase for software used to modify the objects to meet customer needs and enhancing the efficiency of the software.

Figure 3.2: The Waterfall model.

From the figure 3.2, V&V represents verification and validation. Verification represents that the system meets its requirements and thus tries to assess the correctness of the transition to the next phase. Validation represents the system meets the user’s requirements [98].
Advantages of Waterfall Life Cycle Model:

- Easy to explain to the user.
- Helps to plan and schedule the project.
- Activities and stages are well defined.
- Each stage of verification ensures early detection of errors.

3.2 Basics of the Requirement Engineering of Software systems

3.2.1 Introduction to Requirements Engineering

Requirements engineering is involved with what needs to be designed instead of how it is to be designed. It is also involved with some future situation. Consider an example as system design is shown in figure 3.3 which consists of the user’s present job and technological options as the inputs to the system and the future system is the output.

![Figure 3.3: The process of design from the present through to future systems.](image)

In this sense, we need the requirements for designing the system. There must be some knowledge of the future system included by the future situation, before the design can start. Requirements engineering is involved with finding out about the future situation and related change. It is involved with considering possible options and collecting information, and with identifying what should be designed in order to meet some perceived future need.

3.2.2 Definition to Requirements Engineering

"Requirements Engineering can be defined as the systematic process of developing requirements through an iterative co-operative process of analyzing the problem, documenting the
resulting observations in a variety of representation formats, and checking the accuracy of the understanding gained. (Pohl, 1993) [75]."

Pohl definition suggests that it may be rather easy to consider requirements engineering only in terms of a process which allows us to populate a requirements document.

3.2.3 Requirements Engineering Process

According to the Davis (1993) [22], two types of activities were described that occur during the requirements phase of a project. The first activity is the problem analysis, where by analysts spend their time brainstorming, interviewing people who have the most knowledge about the problem in hand, and identifying all the possible constraints on the problem’s solution [55].

Product description is the second activity, which allows time to take pen in hand and to make some difficult decisions and prepare a document that describes the expected external behavior of the product [55].

Davis recognizes that these two activities are represented in different ways. First activity is characterized by an expansion of information and knowledge. The second activity is characterized by resolving conflicting views, organization of ideas, and eliminating ambiguities and inconsistencies. General model of the requirements process is shown in figure 3.4.

![Figure 3.4: Activities within requirements.](image)

A trigger provided by the product concept starts the requirements process. The trigger might be used to improve the customer service. Problem analysis is involved with devel-
developing an understanding of the nature of the problem associated with the product concept. Possible solutions can be suggested, once the problem is clearly understood. Feasibility is involved with evaluating the costs and benefits of different solutions. Once the solution has been decided upon, modeling of the solution is the next step. In order to check the accuracy of the information collected, each activity should be followed by validation. Then the requirements process can be documented.

3.2.4 UML-Basics

UML stands for Unified Modeling Language. This is an Object-oriented system used for designing, modeling or analyzing the ideas in software systems before being implementing the project. UML diagrams are classified into two types such as structural modeling and behavioral modeling.

**Structural modeling consists of five types of diagrams:**

- Class Diagram
- Object Diagram
- Use-Case Diagram
- Component Diagram
- Deployment Diagram

**Behavioral modeling consists of four types of diagrams:**

- Sequence Diagram
- Collaboration Diagram
- State Diagram
- Activity Diagram

3.2.5 Structural modeling

Structural modeling helps in communicating and understanding the elements that make up a system and the functionality the system provides.

**Class diagram:** Class diagram represents the structure of a system. Class diagram consists of following elements.

- A class
- An association
- An attribute
- An operation
Object diagram:  
Object diagram represents the structure of a system at a particular point in time. Object diagram consists of following elements.

- An object
- A link
- An attribute value

Use-case diagram:

Use-case diagram represents the functionality of a system. Use-case diagram consists of following elements.

- An actor
- A use case
- A communicate association

Component diagram:

Component diagram is also known as implementation diagram that represents the implementation of a system. Component diagram consists of following elements.

- A component
- A dependency relationship

Deployment diagram:

Deployment diagram is also known as implementation diagram that represents the implementation environment of a system. Both deployment and component diagrams are specific types of implementation diagrams [6]. Deployment diagram consists of following elements.

- A node
- A communication association

3.2.6 Behavioral modeling

Behavioral modeling helps in communicating and understanding how elements collaborate and interact to provide the functionality of a system.

Sequence diagram:  Sequence diagram is also known as interaction diagram that represents how elements interact over time. A vertical axis represents the time proceeding down the page and a horizontal axis shows the elements involved in the interaction. Sequence diagram consists of following elements.

- Classes and objects
• A lifeline
• A communication

**Collaboration diagram:**
Collaboration diagram is also known as interaction diagram that represents how elements interact over time and how they are related. Both collaboration and sequence diagrams are specific types of interaction diagrams [6]. Collaboration diagram consists of following elements.

• Classes and objects
• Associations
• A communication

**State diagram:**
State diagram is also known as statechart diagram that represents the lifecycle of an element. State diagram consists of following elements.

• A state
• An event
• A transaction
• Initial state
• Final state

**Activity diagram:**
Activity diagram represents the responsibilities and activities of elements. Activity diagram consists of following elements.

• An action state
• A control-flow transition
• An initial action state
• A final action state
• An object-flow
• A swimlane

Clear description of each element in the behavioral and structural models is to be discussed in Chapter 4 with the help of proper diagrams.
3.3 SysML - Basics

SysML stands for Systems Modeling Language, which is a graphical modeling language for systems engineering developed by the OMG (Object Management Group), INCOSE (International Council on Systems Engineering’s) and ISO AP 233 [3] [11]. This modeling language supports the analysis, specification, design, verification and validation of systems that may include components for software, hardware, data, personnel, procedures, and facilities. SysML reuses a content of UML 2 and provides or gives additional extensions to satisfy the requirements or usage of the language. Relation between UML and SysML can be observed in figure 3.5.

![UML vs SysML](image)

Figure 3.5: UML vs SysML.

According to the figure 3.5, SysML is an extension of UML 2.0 and also reused a set of UML 2.0. SysML is meant to be compatible and consistent with the UML 2.0.

3.3.1 Diagram Overview of SysML

SysML diagrams comprise diagram elements, such as nodes connected by paths that represent model elements in the SysML model, such as associations, activities, and blocks.

Diagram taxonomy of SysML is shown in figure 3.6. SysML mostly reuses the UML diagrams. UML diagrams such as sequence, state machine, use case, and package diagrams are strictly reused in SysML. The other types of diagrams are modified and they are consistent with SysML extensions. SysML holds three modified diagrams from UML, such as activity, block definition, internal block diagrams [3]. The internal block diagram, and block definition diagram are similar to the UML composite structure diagram and class diagram respectively. The activity diagram modification has been also done via the activity extensions.

SysML does not use all of the UML diagrams such as the timing diagram, object diagram, interaction overview diagram, deployment diagram, and communication diagram. SysML has added two new diagram types including the parametric diagram and the requirement diagram [3].
A requirement diagram is one of the two SysML new diagram types that provide a modeling construct for text-based requirements, and the relationship between requirements and the other model elements that satisfy them. The parametric diagram is another SysML new type diagram that represents or describes the constraints among the properties associated with blocks [3]. With this diagram, integration can be done between structure and behavioral models with engineering analysis models such as reliability, performance, and mass property models.

In addition to represent a broad range of diagrams and diagram elements, SysML provides two mechanisms, such as the package diagram and the callout notation. With this package diagram, flexibility has been improved to organize the model in packages and views. A package diagram consists of a wide array of packageable elements. In order to represent relationships between model elements the callout notation appears on different diagram kinds.

### 3.4 Database Design basics/methodology

Database is a collection of data that one can search through in a systematic way to maintain and retrieve information or data. Database provides a mechanism for managing, storing and retrieving information with the help of tables. Database tables consist of columns and rows. Each row consists of single record and each column corresponds to different attribute. There are several database models were existed [5]. Here, let’s take a look at the elements of relational databases in order to understand database systems.
3.4.1 Elements of Relational Database

Relational database is based on the relational model, which is a group of rules set forth by E.F. Codd based on the relational algebra principles. This mathematical (relational algebra) principle defines how database management systems should function. The basic relational database model is a combination of elements such as tables, rows, columns, and keys.

- **Table**: Collection of logically related information treated as a unit. Table is made up of rows and columns.
- **Rows**: A single occurrence of the data contained in a table and each row is treated as a single unit.
- **Columns**: All rows in a table comprise the same set of columns.
- **Keys**: There are two types of keys namely primary and foreign. A primary key is a column whose value uniquely identifies each row in a table. A foreign key is a column value in one table that is required to match the column value of the primary key in another table [5].

3.4.2 Table Relationships

Tables can be related to one another by sharing a common column or columns in relational database. This sharing process can be done by using primary and foreign keys. There are three types of table relationships, such as one-to-one, one-to-many, and many-to-many. A one-to-one relationship exists when each row in one table has only row in a second table. A one-to-many relationship exists when each row in one table has one or many related rows in a second table. A many-to-many relationship exists when a row in one table has many related rows in a second table [3].

3.4.3 Database Design Basics:

Designing a database is an iterative process, which is mainly concerned with the following phases:

**Data Analysis**: Data analysis is the first step in the database designing. This is used to analyze and prepare input data in order to enter or append into the tables.

**Logical Database Design**: Creating logical database design is an information-gathering, iterative process. This includes following steps:

- Define the tables.
- Determine the relationships between the tables.
- Determine the columns of each table.
- Determine the primary keys and the column domain. A domain is the collection of set of valid values for each column.
Physical Database Design:
It is a refinement of the logical design and involves the following steps:

- Determine commonly used data.
- Prepare indexes for columns in the table based on data access.
- Improve the database performance.

Physical Implementation:
In this phase, after determining the physical design of database, check the software and hardware components of development and deployment environments, such as storage devices, disk drive access, operating systems and communication systems to maximize the application performance [5].

3.5 Programming Design (basics/methodology)

Programming design is the activity of progressing from a specification of some required program to a description of the program itself. Program design is a phase model recognized by the most phase models of software life cycle. A specification of what the program is required to do is the input to programming design phase. Decisions are made during this phase as to how the program will meet these requirements and description of the program is an output of the phase.

3.5.1 Algorithms

A simple definition for an algorithm is a set of instructions for solving a problem. An algorithm is deterministic process for performing a goal which, given an initial state, will terminate in a defined end state. The efficiency of implementation of the algorithm depends upon size, speed and resource consumption. Algorithms are usually either simulated by a program or implemented by a program [85].

Algorithms are very essential to the way process of computers information, because an algorithm tells the computer what specific steps to perform in order to finish the specified task. Information is read from an input source, written to an output sink, and stored for further processing, when an algorithm is typically associated with processing information [85]. Algorithms can be expressed in following ways:

- Pseudocode and flowcharts are formed ways to express algorithms that avoid the ambiguities.
- Natural language expressions of algorithms tend to be ambiguous and verbose and rarely used for technical or complex algorithms.
- Programming languages are primarily used for expressing algorithms in a form that can be executed by a computer
3.5.2 Software Architecture

In general, software architecture is the set of decisions the software architect makes. The following definition illustrates the software architecture in terms of structural elements and relationships.

"The software architecture of a computing system or program is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them [73] [53]."

Overview of Software Architecture: The computer science field has problems associated with complexity since its formation [86]. In the early days, complexity problems were solved by developers by selecting right algorithms, and data structures. Since the mid-1980s, the fundamental principles of the software architecture have been used by software engineers, even though the term "software architecture" is new to the industries or companies. Software architecture of a system was often characterized by a set of box-and-line diagrams to capture and explain the system imprecisely [32]. The idea of software architecture discipline in centered to reducing complexity through abstraction and separation of concerns. In the computer science programming, abstraction is used to control or reduce the data, and separation of concerns is the process of separating a computer program into distinct parts as little as possible.

Discussion of the selection of an appropriate IDE: IDE stands for integrated development environment, which is also known as integrated debugging environment or integrated design environment. IDE is a software application that provides comprehensive features to computer programmers for software development. Almost all IDE’s consists of following features:

- A compiler and/or an interpreter
- A source code editor
- Build automation tools
- A debugger

Many modern IDEs also have additional features such as an object inspector, a class hierarchy diagram, and a class browser for use with object-oriented software development [68]. IDE’s are typically dedicated to a specific programming language. There is also multiple-language IDEs are in use, such as Active State Komodo, Eclipse, Netbeans, WinDev, Visual Studio, and Xcode. Typically IDE presents a single program in which all development is done. This program provides many features for modifying, deploying, compiling and debugging software. Programmer can choose a specific IDE based on application development.

Use of an IDE: The basic purpose of using an IDE is to manage the application code. The size of code increases, as the requirements and thus application develops gradually. It is very important and necessary to manage this code well, therefore later debugging and
Modification of code can be carried out smoothly. An IDE helps to manage the application by breaking the application into projects, separating files and grouping the files into folders. Consider an example for developing an accounts management system that might be break into three projects:

- One for user interface application
- Other for accounts implementation
- The last one for information storage

User interface project could be further divide into three folders:

- One for user interface forms
- Other for classes to interact with the accounts
- The last one for classes to interact with the information storage project

An IDE also provides its support from the start of development to the creation of final installation.

**IDEs interesting for the thesis work:** There are several popular IDEs that are applicable or suitable for the thesis work. Apart from that, two IDEs are discussed here.

**Eclipse:** This is an IDE, which can be used to develop software in multiple languages. Eclipse was open sourced in November 2001. Eclipse is now controlled by the Eclipse Foundation, which is an independent non-profitable organization. Over worldwide, this IDE is now being used by thousands of developers. A system requirement is the main factor for all IDEs. Eclipse runs on today's most popular operating systems, including Windows XP, Mac OS X, and Linux.

**Netbeans:** This is an IDE, which can be used for developing software languages as PHP, C, C++, Java, JavaScript, Ruby, Python, and Groovy. Netbeans was first open sourced in November 2001 and it was brought by Sun Microsystems. Netbeans IDE is used to build very efficient applications and runs on today's most popular operating systems, including Windows, Linux, Mac OS X, and Solaris. Applications based on the Netbeans can be extended by third party developers.

### 3.6 Software Testing

Software testing is the process of identifying the completeness, correctness, quality, and security of developed software. It is also an empirical investigation conducted to provide stakeholders with information about the quality of the product. Software testing can also be stated as the process of validating and verifying that a software product:

- Works as expected.
- Meets the business and technical requirements that guided its design and development
- Can be implemented with the same characteristics.
3.6.1 Software Testing Principles

Software testing is an extremely creative and intellectually challenging task. The following mandatory software testing principles are [17]:

- Testing must be performed by the person that developed the software since they tend to defend the correctness of the program.
- Only the best personnel must be assigned to design, implement, analyze test cases, test data and test results, because testing requires high responsibility and creativity.
- Testing should not be planned under the tacit assumption that no errors will be found.
- The program or application generates correct results when the test is valid.
- Testing is the process of executing software with the intent of finding errors.
- The application or program must be static (not modified) during the implementation of the set of designed test cases.
- Document test cases and test results.

3.6.2 Possible error types found by Software Testing

Different software tests organize the methods for finding the possible different error types is as follows [94]:

- Syntax errors: These errors can be best found by compiler, lint. Examples for these errors are missing semicolons, values defined but not used, order of evolution disregarded.
- Data errors: These errors can be best found by software inspection and module tests. Examples for these errors are Overflow of variables at calculation, usage of inappropriate data types.
- Algorithm and logical errors: These errors can be best found by software inspection and module tests. Examples for these errors are wrong program flow, use of wrong formulas and calculations.
- Interface errors: These errors can be best found by software inspection, module tests and component tests. Examples for these errors are Overlapping ranges, range violation, unexpected inputs.
- Operating system errors, architecture and design errors: These errors can be best found by Design inspection, integration tests. Examples for these errors are disturbances by OS interruptions, timing problems, lifetime and duration problems.
- Integration errors: These errors can be best found by Integration tests and system tests. Examples for these errors are resource problems (run time, stack, registers, memory, etc).
CHAPTER 3. SOFTWARE DEVELOPMENT PROCESS AND PROGRAMMING BASICS

- **System errors**: These errors can be best found by System tests. Examples for these errors are wrong system behaviour and specification errors.

**Summary**: In this chapter, the following concepts were introduced.

- First, basics of software development process and the requirements engineering concepts for the software systems were discussed. There are several software development models. But, only the waterfall method has discussed, because it is simple method and also the basic principles of this method has helped to this thesis.

- Next, basics of modeling languages and database design methods were discussed. Elements in both structure and behavioral model have been showed and each element should be discussed in Chapter 5 diagrammatically.

- Finally, architecture, selection of integrated development environment and testing principles for software development were discussed.
4.1 UML design

Basic concepts of UML design has been discussed in section 3.3. This section shows practical UML diagrams.

4.1.1 Use-case diagram

UML use-case diagram is shown in figure 4.1. Brief description of each element can be explained as follows:

- **An actor**: Shown as a stick figure. This represents users and external elements with which the system we are discussing interacts.
• A use case: Shown as an ellipse. This represents a functional requirement that is described from the perspective of the users of a system.

• communication association: Shown as a solid-line path from an actor to a use case. This represents that the actor uses the use case.

4.1.2 Class diagram

UML class diagram is shown in figure 4.2. Brief description of each element can be explained as follows:

• A class: Shown as a solid-outline rectangle labeled with a name. This represents a general concept.

• An attribute: Shown as a text string in a class’s second compartment. This represents what objects of the class know.

• An operation: Shown as a text string in a class’s third compartment. This represents what objects of the class can do.

• An association: Shown as a solid-line path. This represents a relationship between classes.
4.1.3 Object diagram

UML object diagram is shown in figure 4.3. Brief description of each element can be explained as follows:

- **An Object**: Shown as a solid-line rectangle labeled with a name.
- **A link**: Shown as a solid-line path. This represents a specific relationship between objects.
- **An attribute**: Shown as a text string followed by an equal symbol and its value in an object’s second compartment. This represents what the object knows.

![UML Object Diagram](image)

Figure 4.3: UML object diagram.

4.1.4 Component diagram

UML component diagram is shown in figure 4.4. Brief description of each element can be explained as follows:

- **A component**: Shown as a rectangle with two small rectangles. This represents a part of the system that exists while the system is executing.
- **A dependency relationship**: Shown as an arrow. This represents that the client component uses the supplier component.
CHAPTER 4. SOFTWARE DESIGN AND IMPLEMENTATION PROCESS

4.1.5 Deployment diagram

UML deployment diagram is shown in figure 4.5. Brief description of each element can be explained as follows:

- A node: Shown as a three dimensional rectangle. This represents a resource that is available during execution time.

- A communication association: Shown as a solid-line path between nodes. This represents a communication path between the nodes.

4.1.6 Sequence diagram

UML sequence diagram is shown in figure 4.6. Brief description of each element can be explained as follows:

- Classes and objects: Classes and objects are shown much the same way as on class and object diagrams.

- A lifeline: Shown as a vertical dashed line from an element. This represents the existence of the element over time.

- A communication: This is shown as a horizontal solid arrow from the lifeline sender to the lifeline of the receiver.

4.1.7 Collaboration diagram

UML collaboration diagram is shown in figure 4.7. Brief description of each element can be explained as follows:
Figure 4.5: UML deployment diagram.

Figure 4.6: UML sequence diagram.
• **Classes and objects**: Classes and objects are shown much the same way as on class and object diagrams.

• **A communication**: This is arrow as an arrow attached to a relationship pointing from the sender toward receiver.

### 4.1.8 State diagram

UML state diagram is shown in figure [4.8](#). Brief description of each element can be explained as follows:

- **A state**: Shown as a rectangle with rounded corners. This represents a condition of an element.

- **Initial state**: Shown as a small solid, filled circle. When an element is created, it enters its initial state.

- **Final state**: Shown as a circle surrounding a small solid filled circle. This can be appeared, when an element enters its final state.

- **A transition**: Shown as a solid line from a source state to a target state labeled with an event.

- **An event**: This is an occurrence of receiving a message.

### 4.1.9 Activity diagram

UML activity diagram is shown in figure [4.9](#). Brief description of each element can be explained as follows:

- **An action state**: This represents processing. It is shown as a shape with straight top and bottom with convex arcs on the two sides.
Figure 4.8: UML state diagram.

Figure 4.9: UML activity diagram.
• **A control flow transition**: Shown as a solid line from a source action state to a target action state. This represents that the target action starts its processing, once the source action state completes its processing.

• **An initial action state**: Shown as a small solid filled circle. The control-flow transition originating from the initial state.

• **A final action state**: Shown as a circle surrounding a small filled circle. The control-flow transition ends with the final state.

### 4.2 DB design

This section describes the information about practical database design. In this work, Microsoft Access 2000 version has been used for the database design. DB form is shown in figure 4.10.

![Database Form](image)

Figure 4.10: Database Form.

New file in the DB form consists of objects such as tables, queries, forms, reports and form. All of these objects are used for designing a database. Table design view is shown in figure 4.11 which consists of several features such as create table in design view, create table by using wizard and create table by entering data. Entering and saving the data values are shown in figure 4.12.

This work has been used the Delphi programming for Graphic User Interface (GUI) that access the data from MS Access database. Source code for accessing and deleting the data from database is as follows:
Figure 4.11: Table designing view.

Figure 4.12: Store the data values in table.
adotable1.TableName:= SELECT OBJECT_ID FROM MRW_OBJECT WHERE SCHLAG_ID = '6';

adotable1.Open;
adotable1.Append;

adotable1byname ('JOB_ID').AsString:= 'OBJECT_ID';
adotable1.close;
adotable1.close;

4.3 Software architecture

4.3.1 Outflow of software architecture

The outflow architecture of software is shown in figure 4.13. This architecture illustrates the external objects of the overall work. There are total three objects that are played major role in this project work. They are farmer, logistics database, and vehicle. This work aims to crate a software that provides reliable communication between three objects.

Working method of software is as follows:

• First, the user sends his/her address, field and GIS information to the logistics office
or logistics database or logistics server, then logistics office transfers the user field information to the vehicle. Therefore, the vehicle locate the farmer field using GIS identification and finishes the user field work and that information can be send back to the farmer through the logistics office. Finally the farmer will finish the payment process.

- Based on the location information from the vehicle, the logistics office sends the user field information to a particular vehicle. In order to get the field information from the logistics office, the vehicle sends the GPS position information (longitude and latitude) to the logistics office for every 10 seconds. According to the position information from the different vehicles, logistics office knows which a vehicle is near to the field.

4.3.2 Internal software architecture at farmer side

Internal software architecture at farmer side is shown in figure 4.14. This allows a farmer to enter his/her address, field, and GIS information into the database. In this case, Graphic User Interface (GUI) provides following features:

- Ability to enter own address, field, and GIS information.
- View of all information as a PDF.
- Ability to draw the field information as a polygon on GIS window.
- Maintain the database of GIS and field information.
- Amendments of polygon and address information.
• Analyze, view, and send address, GIS and field information to the logistics office.

• Possible to send address, GIS and field information to the nearest vehicle.

![Diagram](image)

Figure 4.14: Internal software architecture at farmer.

4.3.3 Internal software architecture at Logistics Office

Internal software architecture at logistics office side is shown in figure 4.15. This allows farmer information for processing. In this case, Graphic User Interface (GUI) provides following features:

• Ability to enter new farmer address, field, and GIS information.

• View of farmer information as a PDF.

• Maintain the database of GIS and field information.

• Possible to send farmer address, GIS and field information to the nearest vehicle.

• Amendments of farmer polygon and address information.

• Designed to receive the address, GIS and field information from the nearest vehicle.

• Possible to filter the all types of users.

• Ability to see the farmers in the same route.
4.3.4 Internal software architecture at Vehicle

Internal software architecture at vehicle side is shown in figure 4.16. This allows farmer processing information from logistics office to finish the farmer field work. In this case, Graphic User Interface (GUI) provides following features:

- Ability to enter the new farmer address, field, GIS information.
- View of farmer information as a PDF.
- Maintain the database of GIS and field information.
- Possible to send farmer address, GIS and field information to the logistics office.
- Amendments of user polygon and address information.
- Designed to receive the address, GIS and field information from the logistics office.
- Possible to filter all types of users.
- Ability to see the users in the same route.
4.4 Selection of the appropriate IDE’s

This work has been used two different IDE’s for two different software languages. They are Delphi IDE for Delphi code implementation and Netbeans IDE for J2ME programming languages. Delphi IDE is used for implementing Graphic User Interface (GUI) at three different objects (farmer, logistics office, vehicle (mobile terminal)). Netbeans IDE is used for sending/receiving GPS information, save the GPS coordinates into the mobile device, and also possible to send these coordinates from mobile to mobile as a text message.

4.5 Source code for software implementation

Source code of this thesis work is to be shown in Chapter 7 (Annex 1).
Chapter 5

Testing and Interpretation of results

This chapter deals with the various types of software testing methods applied on the GUI.

5.1 Testing of the software implemented

The software can be tested using various types of testing methods. In this process, first, we have to test the test object which is the program. It is provided with input data before it is executed. The test object cannot be run alone, but must be embedded into the test bed to obtain an executable program. The test bed process is shown in figure 5.1.

![Figure 5.1: Test bed process.](image)

The test object (source code or program) will usually call different parts of the program (i.e. like libraries in java) through predefined interfaces. These parts of the program are substituted by stubs. The input/output behavior of the program can be simulated by stubs. The test driver simulates that part of the program. The test driver and test stub combine together to form the test bed.

During the program execution, the testing process should show the failures and those which do not satisfy the requirements. The following steps are necessary to execute the tests:
• Determine conditions and preconditions for the test and the goals that are to be achieved.

• Specify the individual test cases.

• Determine how to execute the tests.

There are mainly two methods to test a program. They are:

• Black-box testing

• White-box testing

5.1.1 Black-box testing:

Black-box test design treats the system as a "black-box", so it does not explicitly use knowledge of the internal structure. Black-box test design is usually described as focusing on testing functional requirements.

5.1.2 White-box testing:

White-box test design allows one to peek inside the "box", and it focuses specially on using internal knowledge of the software to guide the selection of test data.

5.1.3 Interpretation of the results when Black-box testing is applied

The test object can be seen as a black-box with the help of black-box testing. Test cases are derived from the specification of the test object. The behavior of the test object is observed from the specification of the test object. This is also called as PoO (Point of Observation is outside the test object). Other than choosing the adequate input test data, it is not possible to control the operating sequence of the object. This is also known as PoC (Point of Control is situated outside of the test).

The black-box testing technique is known as functional or behavioral testing technique, because of the observation of the input/output behavior. The functionality of the test object is the center of attention. This type of testing method is normally used after reaching higher levels (i.e. GUI in our case) even though it is reasonable to use in component tests. Black-box testing approach is shown in figure 5.2. The figure depicts that the PoC and PoO are outside the test object.

The inner structure and design of the object is unknown in case of black-box testing. A test with all possible input data combinations would be a complete test, but this is unrealistic considering the enormous number of combinations. Black-box testing consists of several techniques. State transition method is one of the techniques for software testing. This technique is useful for this thesis work.

In many cases, not only the current input, but also the history of execution, influences the outputs. State diagrams are used to illustrate the dependence on history. The system or test object starts from an initial state and can then come into different states. The
test of a Graphic User Interface (GUI) is an example for state transition method. The GUI consists of a set of screens and user controls, such as menus, dialog boxes, check box, text box, combo box and buttons. Between those, the user can switch back and forth. If screens and user controls are seen as states and input reactions as state transitions then the GUI can be modelled as a finite state machine. The GUI test is shown in figure 5.3.

5.1.4 Interpretation of the results when White box testing is applied

In white box testing, the user is aware of the test object with the help of which he builds a test design. The internal processing of the test object as well as the output is analyzed, while executing the test cases. This process is also known as the Point of Observation (PoO) which is inside the test object. This white box testing is also called as structural testing, because the test designer considers the structure, which includes component hierarchy, flow control, and data flow of the test object. This testing is mostly applied at the lower levels of the testing, i.e. component and integration test. White box testing approach is shown in figure 5.4. The figure depicts that the PoC and/or PoO are inside the test object.

This testing method is based on source code of the test object. Therefore, these techniques are often called as code-based testing techniques. The main idea of white box techniques is to execute every part of the test object at least once. Flow oriented test cases are identified, by analyzing the program logic and then executed. The basic white box test case design techniques are as follows:

- Statement Coverage
- Branch Coverage

Statement Coverage:

This method mainly focuses on each statement of the test object. The test cases shall
CHAPTER 5. TESTING AND INTERPRETATION OF RESULTS

Figure 5.3: The GUI testing.

Figure 5.4: White-box testing approach.
execute all statements of the test object.

- The first step is to translate the source code into a control flow graph.
- The graph makes it easier to specify in detail the control elements that must be covered.
- In the graph, the statements are represented as edges. If sequences of unconditional statements appear in the program, then they are illustrated as one single node, because execution of the first statement of the sequence guarantees that all following statements will be executed.
- During the execution, all statements (IF, CASE, WHILE, FOR) are covered.

**Branch Coverage:**

Branch coverage of the control flow graph is a more advanced criterion for white box testing. For example, the edges in the graph are the center of attraction. In this case, the execution of each statement is not considered, but rather the execution of decisions. The result of the decision determines which statement is executed next. Testing should make sure every decision is executed with both (TRUE and FALSE) possible outcomes.

### 5.2 Comment on tests conducted

As we have already discussed, there is a lot of difference between white box and black box software testing methods. Applying black box technique for software testing is an advantage for customer, because there is no need to check the source code. The customer simply checks the results of Graphic User Interface (GUI), such as text boxes, tool buttons and check boxes and so on. Therefore, the customer informs to the company/supplier who produces the product, if operation of product is not working properly. In other case, applying white box technique for software testing is an advantage for programmer, because it allows the programmer to peek and the check the program or source code. This technique is also convenient for programmer to debug/run the program for every statement/step.

### 5.3 Software implementation process

In this process, first the farmer sends his/her field and personal information to the central office (logistics office). With the help of this field information, the logistics office finds the unique GIS ID of that particular field. It then passes that information to the vehicle which ultimately finishes the field work.

#### 5.3.1 Graphical User Interface of the software developed

The following figure consists of various elements. They are:

- Basic data (Farmer information)
- Fields (Field information)
• Planning (Planned field information)
• Harvest data (Harvest information)
• Mobile terminal (Vehicle)
• JobManagement (Central Office)

Figure 5.5: GUI of the software.

We can access the central office GUI by pressing the JobManagement button as shown in above figure. The form at the central office looks as in figure 5.6. It consists of six different fields. They are:

• **Unplanned Tasks**: Consists of tasks that are to be planned according to the information given by the farmer.

• **Planned Tasks**: These are the tasks which have been planned from the information given.

• **Outbox Tasks**: Those tasks which do not reach the vehicle due to:
  
  – Information delays
  – Vehicle going offline
  – Server problem

• **Send Tasks**: These are the tasks which have been successfully sent from the central office to the vehicle.

• **Started Tasks**: Tasks which are at working stage.

• **Finished Tasks**: Tasks that have been completed.
By clicking on the 'Unplanned task' button in the JobManagement form, the 'Unplanned Task' form appears as shown in figure 5.7. In this form, the name and the farmer identification will be generated from the database. If you click the check box beside a farmer name, a pop-up appears which displays all the field related information as shown in figure 5.8.

Now select one of the field element names and click the update button as shown in figure 5.9.

The updated information will automatically appear in the unplanned tasks menu as shown in figure 5.10.

By clicking on the unplanned tasks as shown in above figure, one more new button will open at left bottom side of the window namely ‘planning a task’. The use of this button is to display the received tasks, activity, machine ID, and comment. And also observe a new window (Info-Fenster) at right bottom side which displays the information about planned tasks. The new button and the new window are to be shown in figure 5.11.

By selecting the unplanned tasks and pressing the planning a task button, a window will appear as shown in figure 5.12. Enter all fields in that form and update them. The updated information will appear in the planned tasks field as shown in figure 5.11 fig. 5.12. This information can be sent to the vehicle by pressing the export button.

We can ability to enter the new farmer information by clicking the new farmer button. With this button also possible to edit the new farmer information like draw the farmer field in the GIS window.

Filter window is shown in figure 5.16. This will match the same object names, such as fieldname, name, and machine.

By checking different items in the filter window in various sections there will open a button in the middle of the each two boxes, from that choose a format either AND or OR and click the update button. The common items are displayed in the JobManagement
### Figure 5.7: Unplanned Task Processed form.

![Unplanned Task Processed form](image)

### Figure 5.8: Unplanned Task Processed form after click the check box.

![Unplanned Task Processed form after click](image)
CHAPTER 5. TESTING AND INTERPRETATION OF RESULTS

Figure 5.9: Unplanned Task Processed form with updated the selected field task.

Figure 5.10: Updated unplanned task processed information.
Figure 5.11: Planning a Task and Info - Fenster information.

Figure 5.12: Planning a Task window.
CHAPTER 5. TESTING AND INTERPRETATION OF RESULTS

Figure 5.13: Seleklierte Schläge an Fahrzeuge Senden information.

Figure 5.14: New user information.
Figure 5.15: New user address information.

Figure 5.16: Filter window.
window.

VirtualEarth button is the last button at the central office and this is used for viewing the various maps, Images, Roads map and for finding the location.

Mobile terminal (vehicle) window will open by clicking the mobile terminal button as shown in figure 5.5. The mobile terminal window is shown in figure 5.17.

![Figure 5.17: Mobile terminal form.](image)

Vehicle should check the tasks in the tasks received and start the task by using the start/finished button. For finishing these tasks, the vehicle should check the same start/finished button. Start and finished task information at vehicle will also being possible to see at central office. And in this mobile terminal, vehicle also possible to plan the task like same as occurs at the central office by using 'plan a task' button.

Vehicle uses the mobile device to find GPS information and send that information to the logistics server through URL address. Along with GPS information, vehicle also sends ID, status and comment to the logistics server. The GUI functionalities at the mobile device are to be shown in figure 5.19. The vehicle uses the following steps:

- Vehicle finds the GPS information on the mobile device and sends those to Logistics server.
- Vehicle gets the processed job form logistics server.
- Vehicle finishes the task and sends status information back to the server.
CHAPTER 5. TESTING AND INTERPRETATION OF RESULTS

Figure 5.18: JobManagement window at mobile terminal.

Figure 5.19: GPS Information.
Chapter 6

Conclusion

This thesis in essence has researched on the following key objectives:

- The basic principles and the related state-of-the-art of precision farming and extract to formulate the related requirements (performance, availability, system architecture and so on) with regard to wireless communication technologies, Geographic Information Systems, and Global Positioning Systems.

- The basic principles and the related state-of-the-art of the GIS technology with a focus on mobile GIS.

- The basic principles and the related state-of-the-art of the Positioning technologies with a focus on GPS and GSM based positioning.

- The basic principles and the related state-of-the-art of the so-called mobile applications (and/or location-based Services).

- The basics of the software-related Systems Engineering.

- Apply the Systems Engineering methods to the case "Tracking System". A comprehensive design is conducted by using the Systems Engineering instruments presented in the objective 5.

- Test the software system developed by applying the methods presented in the objective 6.

The main contribution of this thesis is to develop, implement and test a software which can be used in precision farming. This software generates a reliable communication between three parties, such as farmer (user), central office (logistics office) and vehicle (mobile terminal). This work has been used the following software and hardware technologies.

- Integrated Development Environments (Delphi 6.0 and Netbeans 6.7).

- Programming Languages (Delphi and J2ME)

- Database (Microsoft Access)

- Geographic Information Systems Tool (WinGIS)
• Mobile Terminals (E71 and N95)
• Laptop (Acer)
• Wireless Internet Technologies (Wi-Fi and GPRS)
• Global Positioning Systems
• Geographic Information Systems
Chapter 7

Annex 1 - Code implementation (Delphi, J2ME)

/* Locating the coordinates (Longitude, Latitude, Altitude) */
****************************************************************
/*
/* To change this template, choose Tools or Templates
/*
/* santhosh1MIDlet.java
/*
/* and open the template in the editor.
/*
package GPSCoordinates;
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.location.*;
/**
/* @author Santhosh
/*
public class santhosh1MIDlet extends MIDlet implements CommandListener
private boolean midletPaused = false;
//<editor-fold defaultstate="collapsed" desc="Generated Fields ">
private Command exitCommand;
private Form form;
private StringItem stringItem;
//</editor-fold>
/**
/* The santhosh1MIDlet constructor.
/*
public QualifiedCoordinates Q;
public santhosh1MIDlet ()
{
}
private void initialize()
{
    // write pre-initialize user code here
    // write post-initialize user code here
}

public void startMIDlet()
{
    // write pre-action user code here
    switchDisplayable(null, getForm());
    // write post-action user code here
}

public void resumeMIDlet()
{
    // write pre-action user code here
    // write post-action user code here
}

public void switchDisplayable(Alert alert, Displayable nextDisplayable)
{
    // write pre-switch user code here
    Display display = getDisplay();
    if (alert == null)
    {
        display.setCurrent(nextDisplayable);
    }
    else
    {
        display.setCurrent(alert, nextDisplayable);
    }
}
// write post-switch user code here
}
}/*/editor-fold>
//<editor-fold defaultstate="collapsed" desc=" Generated Method: commandAction for Displayables ">
/**
/* @param command the Command that was invoked
/* @param displayable the Displayable where the command was invoked
*/
public void commandAction(Command command, Displayable displayable)
{
    // write pre-action user code here
    if (displayable == form)
    {
        if (command == exitCommand)
        {
            // write pre-action user code here
            exitMIDlet();
            // write post-action user code here
        }
    }
    // write post-action user code here
}/*/editor-fold>
//<editor-fold defaultstate="collapsed" desc=" Generated Getter: exitCommand ">
/**
/* Returns an initialized instance of exitCommand component.
/* @return the initialized component instance
*/
public Command getExitCommand()
{
    if (exitCommand == null)
    {
        // write pre-init user code here
        exitCommand = new Command("Exit", Command.EXIT, 0);
        // write post-init user code here
    }
    return exitCommand;
}/*/editor-fold>
//<editor-fold defaultstate="collapsed" desc=" Generated Getter: form ">
/**
/* Returns an initialized instance of form component.
/* @return the initialized component instance
*/
public Form getForm()
{  
    if (form == null)  
    {  
        // write pre-init user code here  
        form = new Form("form", new Item[] { getStringItem() });  
        form.addCommand(getExitCommand());  
        form.setCommandListener(this);  
        // write post-init user code here  
        form.append("Finding for Location...");  
        Display.getDisplay(this).setCurrent(form);  
        // boolean found = false;  
        // while (found == false)  
        try {  
            Criteria c = new Criteria();  
            c.setHorizontalAccuracy(5000);  
            c.setVerticalAccuracy(5000);  
            // c.setPreferredPowerConsumption(Criteria.POWER_USAGE_LOW);  
            LocationProvider lp = LocationProvider.getInstance(c);  
            Location loc = lp.getLocation(600);  
            Q = loc.getQualifiedCoordinates();  
            if (Q != null)  
            {  
                form.append("Altitude:" + Q.getAltitude() + ";");  
                form.append("Latitude:" + Q.getLatitude() + ";");  
                form.append("Longitude:" + Q.getLongitude() + ");");  
            }  
            else  
            {  
                form.append("Location API failed");  
            }  
            //found = true;  
            catch (Exception e)  
            form.append("Exception:" + e);  
            }  
        }  
        //} // while  
        
        return form;  
    }  
    } /* Returns an initialized instance of stringItem component. */
    /* @return the initialized component instance */
    public StringItem getStringItem() {  
        if (stringItem == null) {  
            // write pre-init user code here  
    }
stringItem = new StringItem("", "");
// write post-init user code here
}
return stringItem;
} /</editor-fold>
/**
/* Returns a display instance.
/* @return the display instance.
/*/ public Display getDisplay () {
return Display.getDisplay(this);
/**
/* Exits MIDlet.
/*/ public void exitMIDlet () {
switchDisplayable (null, null);
destroyApp(true);
notifyDestroyed();
/**
/* Called when MIDlet is started.
/*/ public void startApp () {
if (midletPaused)
resumeMIDlet ();
} else
initialize ();
startMIDlet ();
}
midletPaused = false;
}/**
/* Called when MIDlet is paused.
/*/ public void pauseApp ()
{
midletPaused = true;
}/**
/* Called to signal the MIDlet to terminate.
/*/ public void destroyApp (boolean unconditional)
{ }
}
public class santhosh2MIDlet extends MIDlet implements CommandListener,Runnable
{
    Command save, exit;
    TextBox text;
    Display display;
    public QualifiedCoordinates Q;
    public santhosh2MIDlet () {
        text = new TextBox("", "", 400, TextField.ANY);
        save = new Command("Save", Command.SCREEN, 1);
        exit = new Command("Exit", Command.EXIT, 1);
        text.addCommand(save);
        text.addCommand(exit);
        text.setCommandListener(this);
    }
    try
    {
        Criteria c=new Criteria();
        c.setHorizontalAccuracy(5000);
c.setVerticalAccuracy(5000);
LocationProvider lp = LocationProvider.getInstance(c);
Location loc = lp.getLocation(600);
Q = loc.getQualifiedCoordinates();
double Altitude;
double Latitude;
double Longitude;
if (Q != null)
{
    Altitude = Q.getAltitude();
    Latitude = Q.getLatitude();
    Longitude = Q.getLongitude();
text.setString("Altitude:" + Double.toString(Altitude) + "n"
    + "Latitude:" + Double.toString(Latitude) + "n"
    + "Longitude:" + Double.toString(Longitude) + "n");
}
else
{
text.setString("Location API failed");
}
} catch (Exception e)
{
text.setString("Exception:" + e);
}
protected void destroyApp(boolean arg0) throws MIDletStateChangeException
{
}
protected void pauseApp()
{
}
protected void startApp() throws MIDletStateChangeException
{
display = Display.getDisplay(this);
display.setCurrent(text);
}
public void commandAction(Command arg0, Displayable arg1)
{
if (arg0 == save)
{
    Thread t = new Thread(this);
    t.start();
}
else if (arg0 == exit)
{
    try
    {

destroyApp(true);
this.notifyDestroyed();
} catch (MIDletStateChangeException e)
{
    e.printStackTrace();
}
} }
public void run()
{
    saveFile();
}
void saveFile()
{
    try {
        FileConnection c = (FileConnection) Connector.open("file:///root1/photos/prog.txt",
Connector.READ_WRITE);
        if (c.exists())
        {
            System.out.println("exist");
        } else
        {
            System.out.println("no exist");
            c.create();
            OutputStream out = c.openOutputStream();
            String userText = text.getString();
            //System.out.println(userText);
            out.write(userText.getBytes());
            out.close();
            c.close();
        } catch (Exception e)
        {
            System.out.println(e.toString());
        }
    }
    /* Send coordinates(Longitude,Latitude,Altitudes)to the server */

package ServerCoordinates;
/*
 * HttpMidlet.java
 */
/*
 * Created on September 6, 2009, 12:20 PM
 */
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.location.*;
import javax.microedition.io.*;
import java.io.*;
public class HttpMidlet extends MIDlet implements CommandListener,Runnable {
    private static String defaultURL = "";
    private static String defaultID = "";
    private static String defaultStatus;
    private Display myDisplay = null;
    private Form requestScreen;
    private TextField requestField;
    private Form IDScreen;
    private TextField id;
    private TextField status;
    private List list;
    private String[] menuItems;
    private Form resultScreen;
    private StringItem resultField;
    private boolean bBreak;
    private int task;
    private Thread thread = null;
    // requestScreen Command
    Command UnextCommand;
    // Setting ON Commands
    Command SONbackCommand;
    Command SONnextCommand;
    // resultScreen Commands
    Command resultbackCommand;
    Command resultexitCommand;
    ChoiceGroup myChoiceGroup;
    public QualifiedCoordinates qc;
    public HttpMidlet() {
        // initialize the GUI components
        myDisplay = Display.getDisplay( this );
        // display the request URL
        UnextCommand = new Command("Next",Command.SCREEN,1);
        requestScreen = new Form( "Type in a URL:" );
        requestField = new TextField( null, defaultURL, 200, TextField.URL);
        requestScreen.append( requestField );
        requestScreen.addCommand(UnextCommand);
        requestScreen.setCommandListener( this );
        // settings ON Operation
        SONbackCommand = new Command("Back",Command.BACK,1);
        SONnextCommand = new Command("Next",Command.SCREEN,1);
        IDScreen = new Form( "Operation Settings" );
id = new TextField("ID[PhoneNumber] (*)", defaultID, 200, TextField.ANY);
IDScreen.append(id);
myChoiceGroup = new ChoiceGroup("Setting Status", Choice.EXCLUSIVE);
myChoiceGroup.append("First", null);
myChoiceGroup.append("Second", null);
myChoiceGroup.append("Third", null);
//myChoiceGroup.setSelectedIndex(1, true);
IDScreen.append(myChoiceGroup);
RadioButton();
status = new TextField("Status:", defaultStatus, 100, TextField.ANY);
IDScreen.append(status);
//Thread t1 = new Thread(this);
//t1.start();
// Display Longitude, Latitude, Altitude
try
{
Criteria c=new Criteria();
c.setHorizontalAccuracy(5000);
c.setVerticalAccuracy(5000);
LocationProvider lp=LocationProvider.getInstance(c);
Location loc=lp.getLocation(600);
qc=loc.getQualifiedCoordinates();
if (qc!=null)
{
IDScreen.append("Longitude:"+qc.getLongitude()+"\n");
IDScreen.append("Latitude:"+qc.getLatitude()+"\n");
IDScreen.append("Altitude:"+qc.getAltitude()+"\n");
}
else
{
IDScreen.append("Location API failed ");
}
} catch(Exception e)
{
IDScreen.append("Exception:"+e);
}
IDScreen.addCommand(SONbackCommand);
IDScreen.addCommand(SONnextCommand);
IDScreen.setCommandListener( this );
// select the HTTP request method desired
menuItems = new String[] "GET Request";
list = new List( "Select an HTTP method:" , List.IMPLICIT, menuItems, null );
list.setCommandListener( this );
// display the message received from server
resultbackCommand = new Command("Back", Command.BACK,1);
resultexitCommand = new Command("Exit", Command.EXIT,1);
resultScreen = new Form( "Server Response:" );
resultScreen.addCommand(resultbackCommand);
resultScreen.addCommand(resultexitCommand);
resultScreen.setCommandLister(this);
}
public void startApp() throws MIDletStateChangeException
{
myDisplay = Display.getDisplay(this);
myDisplay.setCurrent(requestScreen);
}
public void commandAction(Command com, Displayable disp)
{
if (com == UnextCommand)
{id.setString(defaultID);
status.setString(defaultStatus);
myDisplay.setCurrent(IDScreen);
task = 0;
bBreak = false;
thread = new Thread(this);
thread.start();
}
else if (com == resultbackCommand)
{
myDisplay.setCurrent(requestScreen);
}
else if (com == SONbackCommand)
{
myDisplay.setCurrent(requestScreen);
}
else if (com == SONnextCommand)
{
bBreak = true;
myDisplay.setCurrent(list);
}
else if (disp == list com == List.SELECT_COMMAND)
{
task = 1;
bBreak = false;
thread = new Thread(this);
thread.start();
}
else if (com == resultexitCommand)
{
try
{
destroyApp(true);

}
this.notifyDestroyed();
} catch (MIDletStateChangeException e)

e.printStackTrace();

}

public void RadioButton()
{
int i = myChoiceGroup.getSelectedIndex();
//String san = myChoiceGroup.getString(i);
for (i=0; i<3; i++)
{
if(i==0)
defaultStatus= "1";
else if(i==1)
defaultStatus= "2";
}
else if(i==2)
defaultStatus= "3";
}

public void run()
{
while(bBreak == false)

if (task == 0)

int index = myChoiceGroup.getSelectedIndex();
status.setString("n"+index);
else if (task == 1)

String result = sendHttpGet(requestField.getString()+ "?op=pos"+"&id="+
id.getString().trim()+"&x="+qc.getLongitude()+"&y="+qc.getLatitude()+"&status="+status.getString().trim());
resultField = new StringItem( null, result );
resultScreen.append(resultField);
myDisplay.setCurrent(result Screen);
bBreak = true;
}
private String sendHttpGet( String url )
{
    HttpConnection hcon = null;
    DataInputStream dis = null;
    StringBuffer responseMessage = new StringBuffer();
    try {
        // a standard HttpConnection with READ access
        hcon = ( HttpConnection )Connector.open( url );
        // obtain a DataInputStream from the HttpConnection
        dis = new DataInputStream( hcon.openInputStream() );
        // retrieve the response from the server
        int ch;
        while ( ( ch = dis.read() ) != -1 )
            responseMessage.append( (char)ch );
    } //end while ( ( ch = dis.read() ) != -1 )
    catch( Exception e )
    {
        e.printStackTrace();
        responseMessage.append( "ERROR" );
    } finally
    try
        if ( hcon != null ) hcon.close();
        if ( dis != null ) dis.close();
        catch ( IOException ioe )
            ioe.printStackTrace();
    } //end try/catch
    return responseMessage.toString();
} //end sendHttpGet( String )

public void pauseApp()
{
} //end pauseApp()

protected void destroyApp(boolean com) throws MIDletStateChangeException
{
}
import javax.wireless.messaging.TextMessage;  
//import com.sun.midp.lcdui.CommandAccess;
public class prosendSMS extends MIDlet implements CommandListener, Runnable {
    
private Command cmSend, cmExit;
private TextBox tbMessage;
public void pauseApp()
{
}

public void destroyApp(boolean arg0) throws MIDletStateChangeException
{
}
protected void startApp() throws MIDletStateChangeException
{
    tbMessage = new TextBox("Message", "", 160, TextField.ANY);
tbMessage.addCommand(cmSend = new Command("Send SMS", Command.OK, 1));
tbMessage.addCommand(cmExit = new Command("Exit", Command.EXIT, 1));
tbMessage.setCommandListener(this);
Display.getDisplay(this).setCurrent(tbMessage);
}
public void commandAction(Command c, Displayable d)
{
if(c == cmExit)
{
    notifyDestroyed();
}else //if(c == cmSend)
{
    new Thread(this).start();
}
}
public void run()
{
    boolean result = sendSms("", tbMessage.getString());
}
public boolean sendSms(String number, String message)
{
    boolean result = true;
    try
    {
        //set address to send message
        String addr = "sms://" + number;
        //opens connection
        MessageConnection conn = (MessageConnection) Connector.open(addr);
        //prepares text message
        TextMessage msg =
    }
(TextMessage) conn.newMessage(MessageConnection.TEXT_MESSAGE);
// set text
msg.setPayloadText(message);
// send message
conn.send(msg);
conn.close();
}
catch(SecurityException se)
{
result = false;
}
catch (Exception e)
{
result = false;
}
return result;
}
procedure ToolButton4Click(Sender: TObject);
procedure FormShow(Sender: TObject);
private
Private-Declaration
procedure OnWINGISSelected(Sender: TObject; var IDList: OleVariant);
function GetTableInformation(sTableName, sIDFieldName, sValue: string): string;
procedure showingis;
public
Public-Deklarationen
end;
var
frmvehicleinformation: TFormvehicleinformation;
implementation
R *.dfm
uses start, mobilegis;
function TFormvehicleinformation.GetTableInformation(sTableName, sIDFieldName, sValue: string): string;
var
i: integer;
begin
result:="";
adotable1.Close;
adotable1.TableName:=sTableName;
adotable1.Open;
if not adotable1.Locate(sIDFieldName, sValue,[]) then exit;
for i:=0 to adotable1.FieldCount-1 do
begin
result:=result+adotable1.Fields[i].FieldName+’;’+adotable1.Fields[i].AsString+’;’;
end;
end;
procedure TFormvehicleinformation.FormClose(Sender: TObject; var Action: TCloseAction);
begin
theGIS.COM.SetCanvas(frmstart.pan_gis_hide.Handle,false);
theGIS.COM.Core.OnMonitoring:=nil;
end;
procedure TFormvehicleinformation.showingis;
var
sl: TStringlist;
begin
if eltree1.Selected=nil then exit;
sl:=TStringlist.Create;
sl.Add(eltree1.Selected.subitems[1]);
theGIS.COM.SelectInGIS(sl,true,true,-1);
sl.Free;
eend;
procedure TFormVehiclesInformation.Eltree1ItemSelectedChange(Sender: TObject; Item: TElTreeItem);
begin
  showingis;
end;
procedure TFormVehiclesInformation.OnWINGISSelected(Sender: TObject; var IDList: OleVariant);
var i: integer;
s: string;
begin
  s:=IDList.item[0];
  for i:=0 to eltree1.Items.Count-1 do
  begin
    if eltree1.Items[i].SubItems[1]=s then
      begin
        eltree1.Items[i].Selected:=true;
        eltree1.Selected:=eltree1.Items[i];
        eltree1.EnsureVisible(eltree1.Selected);
        exit;
      end;
  end;
end;
procedure TFormVehiclesInformation.ToolButton2Click(Sender: TObject);
var
  t: TElTreeItem;
s: String;
begin
  //display the information of NEWJOBS into the eltree1
  theGIS.COM.SetCanvas(panel1.Handle,true);
  theGIS.COM.SwitchAutoMonitoringOnOff(true);
  theGIS.COM.Core.OnMonitoring:=OnWINGISSelected;
adotable1.Close;
adocollection1.Close;
adocollection1.ConnectionString=""; adotable1.TableName:='NEWJOBS';
adocollection1.Open;
adotable1.Open;
eltree1.items.clear;
progressbar1.Min:=1;
progressbar1.Max:=adotable1.RecordCount;
progressbar1.Position:=1;
progressbar1.Show;
while not adotable1.Eof do
  begin
    s:=";
t:=eltree1.Items.AddItem(nil);
s:=adotable1.FieldByName('Jobid').AsString;
t.Text:=s;
s:=adotable1.FieldByName('Fieldtype').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Fieldid').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Name').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Vorname').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('RUNDE').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Area').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('fruitid').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Begin time').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Fruit').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Object').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('eldid').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('Address').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ABFUHRPLAN_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ANBAUER_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('GIS_ID').AsString;
t.SubItems.Add(s);
adotable1.Next;
progressbar1.StepIt;
end;
progressbar1.Hide;
adotable1.Close;
adoconnection1.Close;
end;
procedure Tfrmvehiclenformation.ToolButton3Click(Sender: TObject);
var
t: TELTreeItem;
s: string;
res,i: integer;
begin
t:=eltree1.Selected;
listbox1.Clear;
s:=t.text;
s:='#JOB_ID#'+GetTableInformation('NEWJOBS','JOB_ID',eltree1.Selected.Text);
listbox1.Items.Add(s);
s:='#ACTIVITY#'+GetTableInformation('NEWJOBS','ACTIVITY',eltree1.Selected.SubItems.Strings[0]);
listbox1.Items.Add(s);
s:='#ACTIVITY_ID#'+GetTableInformation('NEWJOBS','ACTIVITY_ID',eltree1.Selected.SubItems.Strings[1]);
listbox1.Items.Add(s);
s:='#NAME#'+GetTableInformation('NEWJOBS','NAME',eltree1.Selected.SubItems.Strings[2]);
listbox1.Items.Add(s);
s:='#VORNAME#'+GetTableInformation('NEWJOBS','VORNAME',eltree1.Selected.SubItems.Strings[3]);
listbox1.Items.Add(s);
s:='#RUNDE#'+GetTableInformation('NEWJOBS','RUNDE',eltree1.Selected.SubItems.Strings[4]);
listbox1.Items.Add(s);
s:='#FLAECHE#'+GetTableInformation('NEWJOBS','FLAECHE',eltree1.Selected.SubItems.Strings[5]);
listbox1.Items.Add(s);
s:='#FRUCHTART#'+GetTableInformation('NEWJOBS','FRUCHTART',eltree1.Selected.SubItems.Strings[6]);
listbox1.Items.Add(s);
s:='#ERFASST#'+GetTableInformation('NEWJOBS','ERFASST',eltree1.Selected.SubItems.Strings[7]);
listbox1.Items.Add(s);
s:='#FRUCHT#'+GetTableInformation('NEWJOBS','FRUCHT',eltree1.Selected.SubItems.Strings[8]);
listbox1.Items.Add(s);
s:='#OBJECT_ID#'+GetTableInformation('NEWJOBS','OBJECT_ID',eltree1.Selected.SubItems.Strings[9]);
listbox1.Items.Add(s);
s:='#SCHLAG_ID#'+GetTableInformation('NEWJOBS','SCHLAG_ID',eltree1.Selected.SubItems.Strings[10]);
listbox1.Items.Add(s);
s:='#ADRESS_ID#'+GetTableInformation('NEWJOBS','ADRESS_ID',eltree1.Selected.SubItems.Strings[11]);
listbox1.Items.Add(s);
s:='#ABFUHRPLAN_ID#'+GetTableInformation('NEWJOBS','ABFUHRPLAN_ID',eltree1.Selected.SubItems.Strings[12]);
listbox1.Items.Add(s);
s:='#ANBAUER_ID#'+GetTableInformation('NEWJOBS','ANBAUER_ID',eltree1.Selected.SubItems.Strings[13]);
listbox1.Items.Add(s);
s:='#GIS_ID#'+GetTableInformation('NEWJOBS','GIS_ID',eltree1.Selected.SubItems.Strings[14]);
listbox1.Items.Add(s);

// message box show and ask the question "Is the job completed"
with Application do
begin
  NormalizeTopMosts;
  res:=MessageBox('Is the Job Completed', 'Look', MB_YESNO);
  RestoreTopMosts;
end;

// if yes append the job done information into the MRW_COMPLETEJOBS table
if res = IDYES then
begin
  adoconnection1.Close;
  adoconnection1.ConnectionString:='Provider=Microsoft.Jet.OLEDB.4.0;Data Source='+extractfilepath(application.ExeName)+';Persist Security Info=False';
adotable1.TableName:='COMPLETEJOBS';
adotable1.Open;
adoconnection1.Open;
begi
adotable1.Append;
if adotable1.findfield('JOB_ID')<> nil then
adotable1.fieldbyname('JOB_ID').AsString :=
GetTableInformation('NEWJOBS','JOB_ID',eltree1.Selected.Text);
if adotable1.findfield('ACTIVITY')<> nil then
adotable1.fieldbyname('ACTIVITY').AsString :=
GetTableInformation('NEWJOBS','ACTIVITY',eltree1.Selected.SubItems.Strings[0]) ;
if adotable1.findfield('ACTIVITY_ID')<> nil then
adotable1.fieldbyname('ACTIVITY_ID').AsString := GetTableInformation('NEWJOBS','ACTIVITY_ID',eltree1.Selected.SubItems.Strings[1]);
if adotable1.findfield('NAME')<> nil then
adotable1.fieldbyname('NAME').AsString :=
GetTableInformation('NEWJOBS','NAME',eltree1.Selected.SubItems.Strings[2]);
if adotable1.findfield('VORNAME')<> nil then
adotable1.fieldbyname('VORNAME').AsString :=
GetTableInformation('NEWJOBS','VORNAME',eltree1.Selected.SubItems.Strings[3]);
if adotable1.findfield('RUNDE')<> nil then
adotable1.fieldbyname('RUNDE').AsString :=
GetTableInformation('NEWJOBS','RUNDE',eltree1.Selected.SubItems.Strings[4]);
if adotable1.findfield('FLAECHE')<> nil then
adotable1.fieldbyname('FLAECHE').AsString :=
GetTableInformation('NEWJOBS','FLAECHE',eltree1.Selected.SubItems.Strings[5]);
if adotable1.findfield('FRUCHTART')<> nil then
adotable1.fieldbyname('FRUCHTART').AsString :=
GetTableInformation('NEWJOBS','FRUCHTART',eltree1.Selected.SubItems.Strings[6]);
if adotable1.findfield('ERFASST')<> nil then
adotable1.fieldbyname('ERFASST').AsString :=
GetTableInformation('NEWJOBS','ERFASST',eltree1.Selected.SubItems.Strings[7]);
if adotable1.findfield('FRUCHT')<> nil then
adotable1.fieldbyname('FRUCHT').AsString :=
GetTableInformation('NEWJOBS','FRUCHT',eltree1.Selected.SubItems.Strings[8]);
if adotable1.findfield('OBJECT_ID')<> nil then
adotable1.fieldbyname('OBJECT_ID').AsString :=
GetTableInformation('NEWJOBS','OBJECT_ID',eltree1.Selected.SubItems.Strings[9]);
if adotable1.findfield('SCHLAG_ID')<> nil then
adotable1.fieldbyname('SCHLAG_ID').AsString :=
GetTableInformation('NEWJOBS','SCHLAG_ID',eltree1.Selected.SubItems.Strings[10]);
if adotable1.findfield('ADRESS_ID')<> nil then
adotable1.fieldbyname('ADRESS_ID').AsString :=
GetTableInformation('NEWJOBS','ADRESS_ID',eltree1.Selected.SubItems.Strings[11]);
if adotable1.findfield('ABFUHRPLAN_ID')<> nil then
adotable1.fieldbyname('ABFUHRPLAN_ID').AsString := GetTableInformation('NEWJOBS','ABFUHRPLAN_ID',eltree1.Selected.SubItems.Strings[12]);
"
tion('NEWJOBS', 'ABFUHRPLAN_ID', eltree1.Selected.SubItems.Strings[12]);
if adotable1.fieldbyname('ANBAUER_ID').AsString <> nil then
    adotable1.fieldbyname('ANBAUER_ID').AsString := GetTableInformation('NEWJOBS', 'ANBAUER_ID', eltree1.Selected.SubItems.Strings[13]);
if adotable1.fieldbyname('GIS_ID').AsString <> nil then
    adotable1.fieldbyname('GIS_ID').AsString := GetTableInformation('NEWJOBS', 'GIS_ID', eltree1.Selected.SubItems.Strings[14]);
end;
adotable1.Post;
end;
adotable1.Close;
end;

// if yes delete the job done information from the MRW_NEWJOBS table
adoconnection1.Close;
adoconnection1.ConnectionString:='Provider=Microsoft.Jet.OLEDB.4.0;UserID=Admin;Data Source=' + extractfilepath(application.ExeName) + ';Mode=Share Deny None;Extended Properties="";Persist Security Info=False';
adotable1.TableName:='NEWJOBS';
adotable1.Open;
adotable1.Con.
adotable1.fieldbyname('FRUCHTART').Clear;
if adotable1.findfield('ERFASST').AsString =
GetTableInformation('NEWJOBS','ERFASST',eltree1.Selected.SubItems.Strings[7]) then
adotable1.fieldbyname('ERFASST').Clear;
if adotable1.findfield('FRUCHT').AsString =
GetTableInformation('NEWJOBS','FRUCHT',eltree1.Selected.SubItems.Strings[8]) then
adotable1.fieldbyname('FRUCHT').Clear;
if adotable1.findfield('OBJECT_ID').AsString =
GetTableInformation('NEWJOBS','OBJECT_ID',eltree1.Selected.SubItems.Strings[9]) then
adotable1.fieldbyname('OBJECT_ID').Clear;
if adotable1.findfield('SCHLAG_ID').AsString =
GetTableInformation('NEWJOBS','SCHLAG_ID',eltree1.Selected.SubItems.Strings[10]) then
adotable1.fieldbyname('SCHLAG_ID').Clear;
if adotable1.findfield('ADRESS_ID').AsString =
GetTableInformation('NEWJOBS','ADRESS_ID',eltree1.Selected.SubItems.Strings[11]) then
adotable1.fieldbyname('ADRESS_ID').Clear;
if adotable1.findfield('ABFUHRPLAN_ID').AsString =
GetTableInformation('NEWJOBS','ABFUHRPLAN_ID',eltree1.Selected.SubItems.Strings[12]) then
adotable1.fieldbyname('ABFUHRPLAN_ID').Clear;
if adotable1.findfield('ANBAUER_ID').AsString =
GetTableInformation('NEWJOBS','ANBAUER_ID',eltree1.Selected.SubItems.Strings[13]) then
adotable1.fieldbyname('ANBAUER_ID').Clear;
if adotable1.findfield('GIS_ID').AsString =
GetTableInformation('NEWJOBS','GIS_ID',eltree1.Selected.SubItems.Strings[14]) then
adotable1.fieldbyname('GIS_ID').Clear;
adotable1.Post;
end;

procedure Tfrmvehicleinformation.ToolButton4Click(Sender: TObject);
begin
Close;
end;

procedure Tfrmvehicleinformation.FormShow(Sender: TObject);
begin
ToolButton2.Click;
end;
end.

unit vehiclefinishedinfo;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, DB, ADODB, StdCtrls, ComCtrls, ExtCtrls, ETree, MMGISCom, ImgList, ToolWin;

type
Tfrmvehiclecompletedinformation = class(TForm)
ETree1: TETree;
Panel1: TPanel;
ProgressBar1: TProgressBar;
ADOTable1: TADOTable;
ADOConnection1: TADOConnection;
ToolBar1: TToolBar;
ToolButton1: TToolButton;
ToolButton2: TToolButton;
ToolButton3: TToolButton;
ToolButton4: TToolButton;
ToolButton5: TToolButton;
ToolButton6: TToolButton;
ImageList1: TImageList;
Procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure EITree1ItemSelectedChange(Sender: TObject; Item: TElTreeItem);
procedure ToolButton1Click(Sender: TObject);
procedure ToolButton2Click(Sender: TObject);
procedure ToolButton3Click(Sender: TObject);
procedure FormShow(Sender: TObject);
private
procedure OnWINGISSelected(Sender: TObject; var IDList: OleVariant);
procedure showingis;
function GetTableInformation(sTableName, sIDFieldName, sValue: string): string;
Public-Deklarationen
end;
implementation
R *.dfm
uses start,mobgis;
// Calling function for collecting the information from report tables
function TfrmVehiclecompletedinformation.GetTableInformation(sTableName, sIDFieldName, sValue: string): string;
var
i: integer;
begin
result:=""
adotable1.Close;
adotable1.TableName:=sTableName;
adotable1.Open;
if not adotable1.Locate(sIDFieldName,sValue,[]) then exit;
for i:=0 to adotable1.FieldCount-1 do
begin
result:=result+adotable1.Fields[i].FieldName+';'+adotable1.Fields[i].AsString+'';
en";
end;
procedure Tfrmvehiclecompletedinformation.FormClose(Sender: TObject; var Action: TCloseAction);
begin
theGIS.COM.SetCanvas(frmstart.pan_ishide.Handle, false);
theGIS.COM.Core.OnMonitoring:=nil;
end;

procedure Tfrmvehiclecompletedinformation.showingis;
var
sl: TStringlist;
begin
if eltree1.Selected=nil then exit;
sl:=TStringlist.Create;
sl.Add(eltree1.Selected.subitems[1]);
theGIS.COM.SelectInGIS(sl,true,true,-1);
sl.Free;
end;

procedure Tfrmvehiclecompletedinformation.eltree1ItemSelectedChange(Sender: TObject; Item: TElTreeItem);
begin
showingis;
end;

procedure Tfrmvehiclecompletedinformation.OnWINGISSelected(Sender: TObject; var IDList: OleVariant);
var
i: integer;
s: string;
begin
s:=IDList.item[0];
for i:=0 to eltree1.Items.Count-1 do
begin
if eltree1.Items[i].SubItems[1]=s then begin
eltree1.Items[i].Selected:=true;
eltree1.Selected:=eltree1.Items[i];
eltree1.EnsureVisible(eltree1.Selected);
exit;
end;
end;
end;

procedure Tfrmvehiclecompletedinformation.ToolButton1Click(Sender: TObject);
var
t: TELTreeItem;
s: string;
begin
Eltree1.CleanupInstance;
CHAPTER 7. ANNEX1 - CODE IMPLEMENTATION (DELPHI, J2ME) 110

theGIS.COM.SetCanvas(panel1.Handle,true);
theGIS.COM.SwitchAutoMonitoringOnOff(true);
theGIS.COM.Core.OnMonitoring:=OnWINGISSelected;
adotable1.close;
adotable1.Open;
adotable1.TableName:='MRW COMPLETEJOBS';
adotable1.open;
adotable1.First;
eltree1.items.clear;
progressbar1.Min:=1;
progressbar1.Max:=adotable1.RecordCount;
progressbar1.Position:=1;
progressbar1.Show;
while not adotable1.Eof do
begin
s="";
t:=eltree1.Items.AddItem(nil);
s:=adotable1.FieldByName('JOB_ID').AsString;
t.Text:=s;
s:=adotable1.FieldByName('ACTIVITY').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ACTIVITY_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('NAME').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('VORNAME').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('RUNDE').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('FLAECHE').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('FRUCHTART').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ERFASST').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('FRUCHT').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('OBJECT_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('SCHLAG_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ADRESS_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ABFUHRPLAN_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('ANBAUER_ID').AsString;
t.SubItems.Add(s);
s:=adotable1.FieldByName('GIS_ID').AsString;
t.SubItems.Add(s);
adotable1.Next;
progressbar1.StepIt;
end;
progressbar1.Hide;
adotable1.Close;
adoconnection1.Close;
end;

procedure Tfrmvehiclecompletedinformation.ToolButton2Click(Sender: TObject);
var
  t: TELTreeltem;
  s: string;
  i: integer;
begin
  t:=eltree1.Selected;
  s:=t.SubItems.Strings[14];
  s:=s+'#JOB_ID#'+GetTableInformation('MRW_COMPLETEJOBS','JOB_ID',eltree1.Selected.Text);
  s:=s+'#ACTIVITY_ID#'+GetTableInformation('MRW_COMPLETEJOBS','ACTIVITY_ID',eltree1.Selected.SubItems.Strings[0]);
  s:=s+'#NAME#'+GetTableInformation('MRW_COMPLETEJOBS','NAME',eltree1.Selected.SubItems.Strings[2]);
  s:=s+'#VORNAME#'+GetTableInformation('MRW_COMPLETEJOBS','VORNAME',eltree1.Selected.SubItems.Strings[3]);
  s:=s+'#RUNDE#'+GetTableInformation('MRW_COMPLETEJOBS','RUNDE',eltree1.Selected.SubItems.Strings[4]);
  s:=s+'#FLAECHE#'+GetTableInformation('MRW_COMPLETEJOBS','FLAECHE',eltree1.Selected.SubItems.Strings[5]);
  s:=s+'#FRUCHT_ART#'+GetTableInformation('MRW_COMPLETEJOBS','FRUCHT_ART',eltree1.Selected.SubItems.Strings[6]);
  s:=s+'#ERFASSST#'+GetTableInformation('MRW_COMPLETEJOBS','ERFASSST',eltree1.Selected.SubItems.Strings[7]);
  s:=s+'#FRUCHT#'+GetTableInformation('MRW_COMPLETEJOBS','FRUCHT',eltree1.Selected.SubItems.Strings[8]);
  s:=s+'#OBJECT_ID#'+GetTableInformation('MRW_COMPLETEJOBS','OBJECT_ID',eltree1.Selected.SubItems.Strings[9]);
  s:=s+'#SCHLAG_ID#'+GetTableInformation('MRW_COMPLETEJOBS','SCHLAG_ID',eltree1.Selected.SubItems.Strings[10]);
  s:=s+'#ADRESS_ID#'+GetTableInformation('MRW_COMPLETEJOBS','ADRESS_ID',eltree1.Selected.SubItems.Strings[11]);
  s:=s+'#ABFUHRPLAN_ID#'+GetTableInformation('MRW_COMPLETEJOBS','ABFUHRPLAN_ID',eltree1.Selected.SubItems.Strings[12]);
  s:=s+'#GIS_ID#'+GetTableInformation('MRW_COMPLETEJOBS','GIS_ID',eltree1.Selected.SubItems.Strings[13]);
  theMobGIS.SendText(cJOB,cNoACK,s,false,progressbar1,true,,true);
  ShowMessage('Export finished!');
end;

procedure Tfrmvehiclecompletedinformation.ToolButton3Click(Sender:TObject);
begin
close;
end;

procedure Tfrmvehiclecompletedinformation.FormShow(Sender: TObject);
begin
  ToolButton1.Click;
end;
end.
Chapter 8

Annex2 - Description of the Hardware and Software Platforms

8.1 Geographic Information System

A Geographic Information System (GIS) integrates hardware, software, and database for capturing, storing, updating, managing, analyzing, and displaying all forms of geographically referenced data. The following two types of geospatial data are [4]:

- **Vector Data**: A data structure that is used to represent linear geographic features that are made of ordered lists of x, y coordinates. These coordinates are represented by points, lines, or polygons. Attributes are associated with each feature.

- **Raster data**: A spatial data model which is combination of rows and columns of grid cells. Each cell contains an attribute value and location coordinates. Group of cells that share the same value to represent geographic features.

GIS can be viewed as three different ways that are database view, map view and model view.

- **Database View**: A geographic database namely "geodatabase" is a unique kind of database for GIS in the world. Geodatabase combines data repository (database) with spatial data (geo) to create a central data repository for storage of spatial data. Geodatabase allows you to store GIS data in central data repository for easy access and management and this stored information can be leveraged in all environments like server, mobile or desktop. Geodatabase offers you the ability to [27]:
  - Store collection of spatial data in a central data repository
  - Maintain advanced geospatial relational models (e.g. networks, topologies)
  - Apply sophisticated relationships and rules to the data
  - Define and maintain integrity of spatial data with an accurate database

- **Map View**: A GIS is a collection of intelligent maps and other views that show feature relationships on the earth surface. For supporting queries, analysis and editing
the information on GIS maps, maps of the geographic information can be constructed and used as windows into database.

- **Model View:** A GIS is a collection of information tools that creates geographic datasets from already existing datasets \[27\]. Geoprocessing functions take data from already existing datasets, apply analytical functions and write results into newly created datasets.

### 8.1.1 WinGIS

This is an object-oriented professional Geographic Information System developed by the PROGIS Software GmbH Company and used for the window’s environment. It offers full functionality of a GIS, such as the display of geo-referenced orthophotos, graphical objects using coordinates, import and export interfaces for all major GIS and CAD applications. The following points required for the WINGIS software tool are \[79\]:

- 80856 or PENTIUM or compatible processor
- 64 MB main memory (more is better)
- System software MS Windows 95 / NT 4.0 (or higher)
- Graphic card (800X600, 65536 colors)
- Monitor (17 inches or bigger)
- 120 MB disc space

Current graphic user interface (GUI) standard of this software offers consistent with other popular windows office applications. WinGIS has a powerful graphic and data exchange techniques that can be used to exchange the information in real time. WinGIS offered by the following requirements are:

- **Graphic Editor:**
  - Working process is easy with the graphic interface of MS windows.
  - A hybrid system based on vector diagram with embedded raster graphics.
  - Assign of attributes to each graphics object (e.g. lines and points) is also possible outside the database \[79\].
  - Changes in the graphics of an object will automatically update in the database.
  - Universal zoom functions (transparent, selective).

- **Special functions:**
  - Polygon overlay (with each overlaying polygons a new entry in the database is generated)
  - Automatic generation of grid points (e.g. as raster).
  - The raster graphics can be used for interactive digitization.
- Enclosed library for lines, symbols, hatchings and texts.
- Digitization: Snap minimizes errors; automatic snap allows corrections of already digitized maps.
- Plot of certain views, user defined regions, automatic cartographic grids, design of templates, plots on multiple pages.

**WinGIS - Database Features:**

- Graphic interface that guarantees simple working under MS windows.
- The PROGIS standard database is WinMaster 2000 or WinMonitor.
- Relational SQL standard, tabular structure of columns and rows.

**Functions of WinGIS database:**

- Database queries: each relationship and each nesting of queries possible.
- Each ODBC (Open Database Connectivity) database binding.
- Ability to open different database files, like e.g.: ACCESS.
- Statistic and mathematical operations and connections of data fields are possible.
- Interactive monitoring: Possible to see one or more objects of attribute data, while working in the graphics surface.
- Monitoring: Display data fields with connected objects [79].
- Edit / Paste the database.
- Label Objects: Look charts layout additional options.
- Diagram: Charts layout and additional options.

### 8.2 DBMS (Microsoft Access, MySQL, PostgreSQL)

Database provides a mechanism for managing, storing and retrieving information through the use of tables [15]. Since 30 years, the process of data has been changing dramatically. Data processing with conventional files fulfilled a need that existed during the 1960s. As users are exposed to data processing, they find the additional ways to use the services [89]. Today, different people can access the same data from different places with the help of DBMS.

DBMS stands for Database Management System. Processing with database management system offers several advantages. By using DBMS, one can store the data in one location and accessing the same data in many different systems. Storage, manipulate and retrieval of data can be organized by DBMS with the help of software and this software provides data integrity in database management system. There are always compromises, while using hardware and software requirements for installing or working with DBMS software [89]. Following advantages and disadvantages of DBMS are:
• Advantages:
  
  – Data independence
  – Data security
  – Data integrity
  – Consistent data
  – No unplanned redundant data

• Disadvantages:
  
  – Size - very large
  – High cost
  – Complex
  – High impact of failure
  – Increased hardware requirements

First, when you enter the data that can be stored in database as a physical sequence, after that entire file must be read and rewritten for performing the maintenance. This must be true even if you change one record. By using linked lists, data can be stored significantly and reduces the number of records with better performance of maintaining the records and tables. Since user views the data in many objectives, DBMS are usually classified into several models. Some of them are as follows:

8.2.1 Database Types

• Relational Databases: In this case, data can be represented as a two-dimensional table. Data stored by this model have the outward appearance of a flat file or a table. Each table has rows and columns of data. Select the data from the table by specifying the name assigned to the column and identifier of row (s) containing the desired data. In DBMS models, relational data manipulation is very powerful and easy to use. This database model is usually based on complex mathematical notation of relational calculus and relational algebra. Every database model uses query languages for accessing data. Relational database model uses SQL (Structured Query Language) for data manipulation.

• Hierarchical Databases: In this model, data relationships are viewed as a tree structure. This model represents the data as well as simple or complex networks. The application program views of the data maintain the framework of the hierarchical model. In hierarchical model, application program views of data represented as a tree structure. Establish the new databases with little or no change of existing programs as long as basic rules of the software are followed.

• Network Database Models: In this model, data relationships can be represented as 1: M relationship termed sets. It was common to determine the design of the system and requirements of the system in the starting days of data processing to check
the software and hardware requirements that are available. Actually doing business method has to be change to fit the data processing system, but data processing system has changed to user needs. In this model case, unable to represent the data relationships directly using tree structures (for example incase of child has more than one parent). In this network model case, data relationships exist under networks in complexity and it uses one to many relationships.

8.2.2 Microsoft Access Database

It is a relational database management system that offers full featured procedural programming language and features of Microsoft Access has extended to object oriented. From early days onwards, Microsoft has been supporting many features and offers several products. Microsoft Access package contains following elements:

- Access is a relational database management system, which supports both query languages as SQL (Structured Query Language) and QBE (Query By Example) for accessing database [12].
- Access offers full featured procedural programming language that expects you to view your application as commands to be sequentially executed.
- Access has extended the features to object oriented that expects you to view your application as objects.

Microsoft Access database file: In general database is a collection of relate data tables. In addition to the data tables, Microsoft access database file contains several database objects. Database objects in Microsoft access are [12]:

- Reports for printing results.
- Forms for interacting with data on screen.
- Saved queries for organizing data.
- Visual basic programs and macros for extending the database functionality applications.

All of the database objects are stored in a file named with <some filename>. mdb and when you run Microsoft Access, temporarily generate a file name with <some filename>. ldb.

8.2.3 MySQL Database

Microsoft Access is a powerful relational database management system that allows everyone to store information in tables and that manages directly from the local disk. Coming to my SQL database, which is a storage manager. Access can be ability to make ODBC connections to MySQL database servers over the network, if MyODBC driver has already installed in the system [23]. Still possible to see the contents of table information through the Access, but the tables themselves are hosted by the MySQL server. Microsoft Access
has its own strength such as an easy to use interface and at the same time has own limitations are generally used for single user application and managing limited amounts of data.

- **Reasons to Migrate from Microsoft Access to the MySQL database:** MySQL database offers several features rather than Access and it also supports Access features. Some of the features are discussed here.

  - **Deployment of information:** Contents of MySQL table information can be used in many ways and to access that information not only from the MySQL table at other location but also from Microsoft Access table. The MySQL database easily integrates with Web servers like Apache through a number of languages, such as PHP, Perl, Ruby, Python and Java [25].
  
  - **Multi-user access:** Access only support single user interface, but MySQL easily handles many simultaneous users.
  
  - **Management of large databases:** Ability to manage hundreds of megabytes of data by using MySQL database.
  
  - **Security:** There is no particular security features in Microsoft Access. So, anyone can easily access the data from the tables in the Access. MySQL server manages security, if store the data tables in MySQL. One can access the data tables in MySQL with proper username and password only.
  
  - **Cost:** Cost evolution of MySQL is freely obtainable, but Microsoft Access is not free.
  
  - **Hardware choices:** Microsoft Access works on a single platform but MySQL runs on several platforms.

- **Methods of Transferring Database from Microsoft Access to the MySQL database:** Before migrating information from Access to MySQL database, first copy the all contents of table information from Access to the MySQL database, then delete the table information stored in Access database. Establish an ODBC connection from Access to the MySQL server and recreating the table(s) links to the MySQL links.

- **Telling Microsoft Access to Export its own Tables:** Migrating data from Access to the MySQL can be done by using the export feature provided by Access itself to write out the contents of each table as a text file. Using a LOAD DATA statement, each file can be loaded into MySQL. MySQL program for importing the table file from Access to the MySQL database is shown like here [25]:

  ```
  C:> mysqlimport -local -fields-terminated-by=,
  -fields-enclosed-by='"'
  -lines-terminated-by='rn'
  mydb Tablename.txt
  ```

As discussed already, to create the username, hostname and password to the MySQL database of security purpose that commands are shown here:
8.2.4 PostgreSQL Database

PostgreSQL is an Object-Relational Database Management System (ORDBMS) developed by the computer science department members at the University of California and it offers several modern features. Modern features offered by PostgreSQL are [77]:

- **Triggers:** This is a specification function. In PostgreSQL, using with a trigger, database should execute a particular function automatically whenever a certain type of operation is performed. Triggers are generally organized or defined to functions either before or after any UPDATE, INSERT or DELETE operation, either once per modified row, or once per SQL statement. PostgreSQL offers two types of triggers such as per-statement triggers and per-row triggers. The trigger function of per-statement is invoked only once when an appropriate statement is executed in PostgreSQL database. The per-row trigger can be used once for each row. These triggers are sometimes called statement-level triggers and row-level triggers, respectively.

- **Views:** In PostgreSQL, views are implemented using the rule system. The view program used by PostgreSQL is shown here.

  ```sql
  CREATE TABLE myview (same attribute list as for mytab);
  CREATE RULE "_RETmyview" AS ON SELECT TO myview DO INSTEAD SELECT * FROM mytab;
  ```

- **Multi Version concurrency control:** MVCC removes old data level locking and replace it with a new locking system that is most superior to the commercial database system. To preventing reads by other users in a traditional system, each row that is modified is locked until committed [77]. Along with modern features, PostgreSQL also offers new data types, functions, operators, and procedural languages therefore user can be extend the PostgreSQL in many ways.

8.3 IDE'S (Delphi 6.0, Netbeans 6.7):

This section discusses about two different integrated development environments.

8.3.1 Delphi IDE

It is the premier development environment for software developers and database application developers who need to rapidly deliver high performance and easy to
maintain software applications. Compare to other versions, which provides three editions such as Enterprise, Professional, and personal. The personal edition is aimed lacks database support and other advanced features the professional version enables database support and other cross platform development but has no support for new XML and WEB services technologies.

Use the powerful Delphi IDE with its visual design surface, extensive component universe and powerful heterogeneous database framework to visually design and deliver applications in a fraction of the time. The powerful Delphi language and compiler deliver high performance and access to all the power and speed of native windows development. Delphi also provides a comprehensive set of editing, refactoring, and debugging tools in a complete solution focused on making development faster and easier. It has new features in the following areas [71]:

– **Data modules**: The title bar of a module displays the module’s name. "Data-ModuleN" is the default name for a data module where N is a number representing the lowest unused unit number in a project.

For example, when we start to implement new project, we have to open new project or new form. That the project name is Unit1, Unit2... etc. the name will be displayed except our presently using the project name. The following steps are needed to rename the project.

* Select the module and possible to edit the name property for the module in the Object Inspector.
* The new name for the module appears in the title bar when the title Name property in the Object Inspector no longer has focus.

![Figure 8.1: Delphi IDE.](image)

Figure 8.1: Delphi IDE.
By changing the name of a data module at design time that leads to change its variable name in the mode of interface section and it also changes any use of type name the procedure declarations. So, try to being change any references by manually to the data module in the code.

- **Object Tree View:** Object Tree View is located in the upper left-hand corner of the Delphi IDE and Tree View is a type of diagram that displays the logical relationships between non visual and visual components on a form, frame or data module. The Tree View is synchronized with the Object Inspector and Form Designer so, if we select an item and change the focus in any one of these three tools, the focus changes in the two tools. Functions of Object Tree View:
  
  * The Tree View is positioned above the Object Inspector and if not displayed, appears when you press Alt+Shift+F11 or Choose View Object Tree View.
  * The Tree View includes components on a form and a frame as well as a data module.
  * The Tree View includes visual as well as nonvisual components.
  * A new toolbar includes New Item, Delete, and Move Up/Down buttons.
  
  These buttons work for component properties.

For example, if we have added a dataset component, we can select the aggregates property and click the New Item button to add a field. If there is more than one type of thing that can be added, the New Item button will drop down a menu to pick from.

- **Code editor:** The code editor now supports surface designs, or package-loaded custom views. These views are accessible as tabs, or pages, located on the status bar. The only built-in view is the standard code page. There is a page on the code editor name as diagram that provides lines to display relationships schematically and visual tools for setting up a diagram of boxes and possible to add comments to the diagram page. Components do not appear on the diagram page until we drag them from the Object Tree View.

  * We can select the multiple items from the Object Tree View and drag them to the diagram page at one time.
  * The left side of the diagram pages an edit box where you can type a name and description for each diagram we create and a drop-down list box to find previously named diagrams.
  * The toolbar of buttons for connector relationships and comment blocks sits at the top of the diagram page.
  * Property connectors are labeled automatically. We can drag the labels elsewhere on the page.
  * We can create a diagram for each data module, form, or frame we have added to our project.

- **Object Inspector:** The Object Inspector is located below the Object Tree View in the Delphi Environment. **Instance List box:**

  * The instance drop-down list box is at the top of the Object Inspector.
* This instance list box displays the class name for each and every object in the list.
* It's possible to give component name and data module name as same. For example, we can add a button component to Form1 and rename the button as Form1; both names appear in the instance list.
* The instance list box can be hidden.

**Properties dialog box:**
* Speed Settings to customize the colors of the Object Inspector.
* Displaying or hiding the instance list, class names in the instance list, status bar, background grid, and read-only properties.
* Properties of component references can be expanded inline and displayed on both the properties and event pages.

**Expanded inline component references:**
* Properties that reference a second component are red and the properties of that second component are green, by default.
* Events that reference a second component are red and the events of that second component are green, by default.
* Properties that are interfaces can be referenced inline.

---

**Code Insight tools:** It is available while we are working in the code editor. This mainly consists of:

* **Code completion:** When we enter a class name followed by a period in the code editor, the list of properties, methods and events appropriate to the class or record is displayed and then it's possible to select the item and press enter to add it to our code. We can always invoke code completion using Ctrl+Space, even if the automatic feature is disabled.
* **Code parameters:** It is possible to view the syntax of a prototype method as we enter it into our code.
* **Tool tip expression evaluation:** When the compiler is stopped while debugging, we can view the value of a variable by pointing to it with our cursor.
* **Tool tip symbol insight:** Information can be displayed for any identifier by passing the mouse over it in the code editor.
* **Code templates:** In general, code templates consist of commonly used programming statements (such as while, if and for statements) that we can insert into our source code. While working in the code editor, press Ctrl+J to display the default code templates or any new ones that we define. If we want to add templates then click add, if we want to delete templates click delete, if we want to edit templates then click edit are possible in the code editor.

---

**Project Manager:** This is used to display the information about the status and file content of the currently open project. Project manager window displays information about all projects within the project group, if the project is part of a project group [71]. With the project manager, we can easily visualize how
all our project files are related and also possible to select any file displayed, right-click, and perform various project management tasks, such as removing, opening, adding compiling our projects.

- **File menu:** It creates the following items.
  - Application: Application creates a new project containing an empty form, a project file and a unit.
  - CLX Application: CLX application creates a new project containing an empty form, a project file for the Linux platform and a unit.
  - Data Module: Creates a new data module.
  - Form: Creates a blank form and adds it to the current project.
  - Frame: Creates a blank frame adds it to the current project.
  - Unit: Creates a new unit.

- **New items dialog box:** The new items dialog box (Object Repository), has three new pages with new wizards and data modules.
  - Web Services: These are the self-contained modular applications that can be invoked and published over a network and also provide well-defined interfaces that describe the services provided. Web services are mainly designed to allow a loose coupling between server and client. That is, clients do not require using in server implementations for a specific platform or programming language. Delphi’s support for web services is designed to work SOAP (Simple Object Access Protocol). SOAP is a standard lightweight protocol for exchanging information in a decentralized, distributed environment. It uses XML to encode remote procedure calls HTTP as a communications protocol.
  - WebSnap: It augments webBroker with new wizards, components and views for making it easier to build web applications that contain complex, data-driven web pages. Websnap’s support for multiple modules and for server-side scripts makes team development easier.
  - CORBA: We can use the CORBA IDL Files dialog box to easily generate a client or server application using an IDL-based file. The dialog box is the same as the CORBA Client Application and CORBA Server wizards that we access by choosing File/New/Other and clicking the CORBA tab.

- **Internet toolbar:** It offers easy access to components that allow us to create our own web-enabled applications.
  - New WebSnap Application: Displays the New websnap application dialog box. This is first step in creating web server application with websnap, or for converting an existing application written for one web server type to run on a different web server type.
  - New Web Snap Page Module: Displays the WebSnap Page Module dialog box. Use this option to create a content page. The page module contains a page producer that is responsible for generating the content of a page. The page can act as a default page when the path info is blank.
CHAPTER 8. ANNEX2 - DESCRIPTION OF THE HARDWARE AND SOFTWARE PLATFORMS

* **New WebSnap Module**: Displays the new WebSnap Data Module dialog box to create an empty data module for adding websnap internet components. For selecting this type of module does not create a content page and is only used as a container for components shared by other modules.

- **Component palette changes**: Component palette changes are used for building customized menus and toolbars, including TactionManager, TactionMainBar, and TactionToolBar [71].
  * For CLX applications, the Common controls page substitutes the Win32 page.
  * A new WebSnap page has been added with components to build a web server application with multiple modules.
  * The web service page has been added for writing multiplier applications over a network or the web.
  * There are many changes have been made on the database component palette pages.

8.3.2 Netbeans IDE

This is a type of IDE (Integrated Development Environment), mostly used for developing the java applications and also supports other software language applications such as PHP, Ruby, Python, Groovy, C and C++.

Netbeans IDE has written in java therefore it runs everywhere, if the JVM (Java Virtual Machine) is already installed in the system and this IDE supports several operating systems they are Windows, Linux, Mac OS, and Solaris.

In this thesis, Netbeans 6.7 version has been used for developing the mobile applications. Major features offered by Netbeans 6.7 are:

- XML Editor
- C/C++ Development
- Mobile Application Development (Java Micro Edition)
- Java ME CLDC Features (MIDP)
- UML Modeling
- New APIs (Application Program Interfaces)

Figure 8.2 represents the Netbeans 6.7 IDE, which is having several features that supports and runs many languages on this IDE. Main tools which we can see on this IDE are project files, Navigator, Code Editor and Mobile device Screen [65].

**Project Files**: Figure 8.3 represents the project files and that will maintain all projects files. Files include source code files and when tried to run the source code that will automatically create 'jar' files in the background. Jar files are enough to run the entire project on a mobile device.
CHAPTER 8. ANNEX2 - DESCRIPTION OF THE HARDWARE AND SOFTWARE PLATFORMS

Figure 8.2: Netbeans IDE.

Figure 8.3: Project File.
Navigator: Figure 8.4 corresponds to the Navigator and this will represent all class functions used in the source code.

![Navigator](image)

Figure 8.4: Navigator.

Code Editor: Figure 8.5 represents the code editor and that is available for many different languages in the Netbeans IDE.

![Code Editor](image)

Figure 8.5: Code Editor.

Mobile Device Screen: Figure 8.6 represents the Mobile device screen. Advantage of mobile device screen is to edit, change and update the properties (text width, height, color) of mobile device screen manually.

Flow of components: Figure 8.7 corresponds to the Components flow. The main advantage of components flow is to make some source code functions (Startup, Run,
CHAPTER 8. ANNEX2 - DESCRIPTION OF THE HARDWARE AND SOFTWARE PLATFORMS

8.4 Mobile Terminals

This section discusses about two types of mobile devices.
8.4.1 E71 Mobile device:

E71 mobile device is used for obtaining the GPS coordinates and sending them to a server.

Software and hardware specifications of Nokia E71:

Security features:
- Device lock
- Data encryption for both phone memory and microSD content
- Remote lock

Connectors:
- Micro USB connector
- 2.5mm Nokia AV connector

Memory:
- MicroSD memory card slot, hot swappable, max 8GB
- 110 MB internal dynamic memory

Operating Frequency:
- Quad-Band EGSM 850/900/1800/1900, WCDMA 900/2100 HSDPA

GPS and Navigation:
- Integrated A-GPS
- Nokia Maps application

Software platform and user interface:
- S60 3rd Edition
- Voice commands
- Symbian OS v9.2
- Two home screens with customizable active standby views

Data Network:
- GPRS class A, multislot class 32, maximum speed 100/60 kbps (DL/UL)
- EDGE class A, multislot class 32, maximum speed 296/177.6 kbps (DL/UL)
- WCDMA 900/2100 or 850/1900 or 850/2100, maximum speed 384/384 kbps (DL/UL)
- HSDPA class 6, maximum speed 3.6 Mbps/384 kbps (DL/UL)
- WLAN IEEE 802.11b/g

8.4.2 N91 Mobile device:

N91 mobile device is used for obtaining the GPS coordinates and sending them to a server.

Software and hardware specifications of Nokia N91:

**Operating System:**
- Symbian OS v9.2

**Frequency Band:**
- GSM 850
- GSM 900
- GSM 1800
- GSM 1900
- WCDMA 2100

**GPS and Navigation:**
- Integrated A-GPS
- Nokia Maps application

**Graphics Formats:**
- BMP, EXIF, GIF, JPEG, PNG

8.5 Server

This section gives the detailed information of working with MySQL server. First, use the login by choosing hostname, password and username that allows entering into the MySQL for security purpose.

```
# [mysql dir]/bin/mysql -h hostname -u root -p
```

Create a database on the sql server.

```
mysql> create database [DATABASENAME];
```
Show databases on the sql server.
mysql> show databases;
Switch to a database.
mysql> use [db name];
See all the tables in the db.
mysql> show tables;
See database’s field formats.
mysql> describe [table name];
To delete a db.
mysql> drop database [database name];
To delete a table.
mysql> drop table [MRW-OBJECT];
Show all data in a table.
mysql> SELECT * FROM [MRW-OBJECT];
The column information to the designed table.
mysql> show columns from [MRW-OBJECT];
Show the selected rows with the values.
mysql> SELECT * FROM [MRW-OBJECT] WHERE [field name] = "OBJECT-ID";

8.6 Wireless Communication System

This section discusses about the GPRS wireless communication system. GPRS stands for General Packet Radio Service which is a packet based wireless communication service for mobile terminals that allows data to be transmitted and received over a mobile telephone network [1]. Actually GPRS is a step towards third generation (3G) but often referred as 2.5G. Some key benefits of GPRS are to be discussed here.

8.6.1 Key benefits of GPRS

- **Connectivity**: GPRS supports always-on service and it makes services instantaneously available to a terminal.

- **Speed**: GPRS offers higher connection speeds at around 56 to 118 kbps and 20 to 50 kbps corresponds to lower connection speeds.

- **New and Better applications**: As we have already discussed, GPRS offers always-on connectivity and high speed connection that enables full internet applications and services as video conferencing for desktop and mobile devices [2].
8.6.2 GPRS Network

GPRS network is shown in figure ?? which consists of the following key elements:

- **SGSN (Serving GPRS Support Node):** With in the GSM infrastructure, this node sends and receives packet data to and from the mobile stations and keeps track of the mobiles within its service area. SGSNs send queries to Home Location Registers (HLRs) to obtain profile data of GPRS subcarriers and detects new GPRS mobile stations in a given service area. The SGSN performs several functions such as, tracking a mobile location, user verification, and collection of billing data [2].

- **GGSN (Gateway GPRS Support Node):** This node interfaces to external Public Data Networks (PDNs) such as the Internet and X.25 protocol. GGSNs maintain routing information that is necessary to tunnel the Protocol Data Units (PDUs) to the SGSNs. The GGSN performs several other functions include network and subcarrier screening and address mapping.

![GPRS Network Diagram](image)

Figure 8.9: GPRS Network.

- **GTP (GPRS Tunneling Protocol):** This is a specialized protocol that operates over the top of standard TCP/IP protocols to encapsulate IP packets.

- **Base Station System (BSS):** This must be enhanced to recognize and send user data to the SGSN that is serving the area.

- **Home Location Register (HLR):** This must be enhanced to register GPRS user profiles and respond to queries originating from SGSNs regarding these profiles.
Bibliography

[27] ESRI. The guide to geographic information system. 2003.


[42] 1999b. IEEE. Wireless medium access control (mac) and physical layer (phy) specifications: Higher-speed physical layer extension in the 2.4 ghz band. ieee standard 802.11b. The Institute of Electrical and Electronics Engineers Inc., 345 East 47th Street, New York, USA.

[43] 2002. IEEE. Wireless medium access control (mac) and physical layer (phy) specifications for wireless personal area networks (wpans). ieee standard 802.15.1. The Institute of Electrical and Electronics Engineers Inc., 345 East 47th Street, New York USA.

[44] 2003. IEEE. Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans). ieee standard 802.15.4. The Institute of Electrical and Electronics Engineers Inc., 345 East 47th Street, New York USA.


[78] GIS PRIMER. Functions of gis. *Designed and Developed by Remote Sensing & GIS Division, National Informatics Center(NIC), New Delhi*.


