VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

A PROJECT REPORT

Submitted by

K. PAVAN KUMAR	**(315132910015)**
T. CHARISHMAA	**(315132910038)**
P. DIVYA	**(315132910027)**
V. SANGAM NAIDU	**(315132910039)**

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Under the esteemed guidance of

CH. VENKATESWARA RAO, M.Tech

Assistant Professor,

Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SANKETIKA VIDYA PARISHAD ENGINEERING COLLEGE

(Affiliated to Andhra University)

P.M. Palem, Visakhapatnam – 530041

March 2019

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

A PROJECT REPORT

Submitted by

K. PAVAN KUMAR **(315132910015)**

T. CHARISHMAA **(315132910038)**

P. DIVYA **(315132910027)**

V. SANGAM NAIDU **(315132910039)**

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Under the esteemed guidance of

CH. VENKATESWARA RAO, M.Tech

Assistant Professor,

Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SANKETIKA VIDYA PARISHAD ENGINEERING COLLEGE

(Affiliated to Andhra University)

P.M. Palem, Visakhapatnam – 530041

March 2019

SANKETIKA VIDYA PARISHAD ENGINEERING COLLEGE

Affiliated to Andhra University, P. M. Palem, Visakhapatnam-530041

BONAFIDE CERTIFICATE

This is to certify that the project work entitled **"VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM"** is a bonafide work done by **K. PAVAN KUMAR(315132910015), T. CHARISHMAA(315132910038), P. DIVYA(315132910027), V. SANGAM NAIDU(315132910039)** as a part of IV/IV B.Tech 2nd Semester of Computer Science and Engineering during the year 2015-19.

Project Guide

CH.VENKATESWARA RAO, M.Tech
Assistant Professor,
Department of CSE,
SVP Engineering College

Head of Department

A. SURAJ KUMAR, M.Tech
Head of Department,
Department of CSE,
SVP Engineering College

External Examiner

DECLARATION

We hereby declare that the project work entitled **"VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM"** is the original work done by us under the guidance of **CH. VENKATESWARA RAO** (Assistant Professor, Department of CSE) and submitted to the Department of Computer Science and Engineering, **SANKETIKA VIDYA PARISHAD ENGINEERING COLLEGE (Affiliated to Andhra University),** for the partial fulfilment of the requirement for the award of **B.Tech degree** and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

K. PAVAN KUMAR

T. CHARISHMAA

P. DIVYA

V. SANGAM NAIDU

ACKNOWLEDGEMENT

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavour to a successful result. We extend our sincere and heartful thanks to our esteemed guide, **CH. VENKATESWARA RAO**, for providing us with the right guidance and advice at the crucial junctures and for showing us the right way and coordinate our project, especially in writing this report.

We extend our sincere thanks to our respected Head of the Department, **A. SURAJ KUMAR,** who invested his full effort in guiding the team in achieving the goal.

We are also thankful to our honourable Principal, **Dr. M. RAMJEE,** for his continuous support and encouragement during the development of this project.

We have to appreciate the guidance given by other faculty members as well as the panels especially in our project presentation that has improved our presentation skills, thanks to their comment and advices.

We would like to thank the other non-teaching staff on this occasion. Last but not the least, we would like to thank our friends and family for the support and encouragement they have given us during the course of this work.

K. PAVAN KUMAR

T. CHARISHMAA

P. DIVYA

V. SANGAM NAIDU

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

CONTENTS

ABSTRACT

This system designs the prototype of accident alert system which is mainly based on Arduino, GPS, GSM and MEMS Accelerometer.When an accident is occurred it is identified immediately and its location is transmitted to one or more subscribers of the system via GSM service.Then the location information can be displayed on the google map by the subscriber, in order to reach the scene of the accident faster and provide necessary medical services required.The communication between the web server and the hardware device is established via GSM shield and the location is traced via GPS shield.

List of Figures

List of Abbreviations

GSM	Global System for Mobile Communications
GPS	Global Positioning System
MEMS	Micro-Electro-Mechanical Systems
SRS	Software Requirement Specification
LED	Light Emitting Diode
LCD	Liquid Crystal Display

List of Tables

CHAPTER 1
PROJECT INTRODUCTION

1.1 Introduction

Figuring most brief ways effectively in a dynamic chart environment additionally discovers its application in a spatial database framework. In such a framework, it is fundamental to give the usefulness of finding an ideal course in a system. A chart in a course question framework by and large is of a self-assertive size and is excessively colossal, making it impossible to be primary memory occupant. In the previous decade, a famous way to deal with taking care of the versatility issue depends on chart apportion in. The entire chart is initially apportioned into littler estimated parts [1][2], each of which can fit into the primary memory. Since the span of a chart could be subjectively extensive, to accelerate the inquiry procedure and to minimize the I/O movement, a typical strategy is to emerge, in every section, the (nearby) briefest separation data between the alleged fringe vertices (those common by more than one piece).

In a continuous movement data framework, an edge weight in a section could be overhauled powerfully, the briefest separation data between fringe vertices must be re-processed quickly for it to be valuable in a course inquiry assessment. This can be expert by emerging, for every fringe vertex, a Shortest Path Tree (SPT) to all other fringe hubs in a section; and re-figuring each SPT at whatever point some edge weights in the piece have been changed. Consider an application in which there are various dispersion bases that are scattered on a metropolitan zone, and it is valuable to know the minimum cost activity courses from every area to all major crossing points. Taking crossing points as vertices, squares between two convergences as edges, and movement latencies as edge weights, the city activity guide is a digraph with non-negative edge weights [3].

The minimum cost course inquiry between two convergences is to locate a most limited way between two vertices in the comparing diagram. Since the activity

condition changes quickly, minimum cost courses may not be right a couple of minutes after they are figured. One could apply Dijkstra's calculation, more than once to process the briefest ways. Be that as it may, this very much contemplated static calculation may get to be ineffectual when just a little number of the city streets experience inertness changes. Consequently, specialists have been considering incremental calculations to minimize briefest ways re-calculation time.

1.2 Spatial Database

A spatial database is a database that is optimized to store and query data that represents objects defined in a geometric space. Most spatial databases allow representing simple geometric objects such as points, lines and polygons. Some spatial databases handle more complex structures such as 3D objects, topological coverages, linear networks, and TINs. While typical databases have developed to manage various numeric and character types of data, such databases require additional functionality to process spatial data types efficiently, and developers have often added geometry or feature data types. The Open Geospatial Consortium developed the Simple Features specification (first released in 1997)[1] and sets standards for adding spatial functionality to database systems.[2] The SQL/MM Spatial ISO/EIC standard is a part the SQL/MM multimedia standard and extends the Simple Features standard with data types that support circular interpolations[4][5].

1.3 Scope of the Project

For experimental purpose we considered some set of nodes with their latitude and longitude values. Entities and feature set are considered with respect to configured nodes in the application. User receives the search results from the available spatial database. We implemented our clustering-based model with in the configured nodes and we will consider the nodes which has entities and feature set of the nodes.

1.4 Problem Statement

Various traditional approaches proposed by various authors from years of research, every approach has its own advantages and disadvantages. Performance and time complexity are the major factors while community searches. Nodes should be grouped based on the weights and edges existence between the nodes. Traditional community-based approaches is more complex to group from the source node and there is no further practical search implementation. Identification of neighbor with simple edge does not retrieve optimality. The major drawbacks with traditional model are More time complexity if route API computes path with all available nodes for every request , less performance and additional overhead to location-based service and User may receive irrelevant results if response is slow.

CHAPTER 2

LITERATURE SURVEY

2.1 Overview

Consider an application in which there are a number of distribution centers that are scattered around a metropolitan area, and it is useful to know the least-cost traffic routes from each location to all major intersections. Taking intersections as vertices, blocks between two intersections as edges, and traffic latencies as edge weights, the city traffic map is a digraph with non-negative edge weights. The least-cost route query between two intersections is to find a shortest path between two vertices in the corresponding graph. Since the traffic condition changes rapidly, least-cost routes may not be correct a few minutes after they are computed. One could apply Dijkstra's algorithm repeatedly to compute the shortest paths. However, this well-studied static algorithm may become ineffective when only a small number of the city roads experience 1 latency changes. Therefore, researchers have been studying incremental algorithms to minimize shortest paths re-computation time. Computing shortest paths efficiently in a dynamic graph environment also finds its application in a spatial database system. In such a system, it is essential to provide the functionality of finding an optimal route in a network. A graph in a route query system in general is of an arbitrary size and is too huge to be main-memory resident. In the past decade, a popular approach to solve the scalability problem is based on graph partitioning, the whole graph is first partitioned into smaller sized fragments, each of which can fit into the main-memory. Because the size of a graph could be arbitrarily large, to speed up the search process and to minimize the I/O activity, a common technique is to materialize, in each fragment, the (local) shortest-distance information between the so-called border vertices (those shared by more than one fragment). In a real-time traffic information system, an edge weight in a fragment could be updated dynamically, the shortest-distance information between border vertices has to be re-computed fast for it to be useful in

a route query evaluation. This can be accomplished by materializing, for each border vertex, a Shortest Path Tree (SPT) to all other border nodes in a fragment; and re-computing each SPT whenever some edge weights in the fragment have been changed.

We call the problem of re-computing SPTs in a dynamic environment the DSP problem. Let G = (V,E,w) be a simple digraph, in which all edge weights are non-negative real numbers. Let G = (V,E,w) be obtained from G by the application of a set of edge weight updates (increases and/or decreases) to G. Let s ∈ V ; let Ts and T s be SPTs rooted at s in G and G , respectively. The DSP problem is to compute T s from Ts. For the DSP problem, the input edge weight changes could come in three forms: increases only, decreases only, and a mixture of both. We denote an algorithm as semi-dynamic if the input is either a set of edge weight increases or a set of edge weight decreases, but not both. An algorithm is said to be fully-dynamic if the input can be a set of mixed edge weight changes. We shall investigate the performance of both semiand fully-dynamic algorithms in this work. An intelligent approach to solve the DSP problem is proposed in [28]. We denote their semi-dynamic algorithms as BallString since they are based on a ball-and-string model. Unfortunately, the semi-dynamic algorithm for edge weight increases case is incorrect. We amend BallString by proposing MBallStringInc that updates SPTs correctly in the case of multiple edge weight increases. We propose a dynamic version of Dijkstra, which we call DynDijkstra. DynDijkstra are two semi-dynamic algorithms that can handle multiple edge weight increases and decreases, respectively. A fully-dynamic algorithm called DynamicSWSF-FP is proposed in . However, a problem with DynamicSWSF-FP is that some of its computation is inefficient. We modify DynamicSWSF-FP by applying some optimizations on re-

computing the so-called rhs values and adding SPT tree structure maintenance. We call the resulting more efficient algorithm MFP

The clustering algorithms in wireless sensor networks there are cluster heads which consists of clusters. In this there are some parameters there are number of clusters, intra-cluster communication, nodes types and roles, cluster head selection and mobility. Coming to number of clusters that is cluster count is the number of clusters. The groups of cluster heads are described and the number of clusters is usually a complex parameters which considers the efficiency of the routing protocol. In clustering methods, the communication between a sensor and its cluster heads are considered as single hop communication.

In many techniques already proposed to increase network time span wireless sensor networks. The clustering techniques are used in wireless ado networks, mobile ado networks with sensor networks. Clustering is a method is hosted in sensor networks are set into clusters. The sensor node is responsible for base station in the communication in a cluster. The sensor node is also called as cluster head and the remaining nodes in the cluster are known as followers. There are various clustering methods are established for wireless sensor networks. The techniques cannot used in wireless sensor networks because of the moderate the need energy efficient than mobile networks and there are very differences in energy of the sensor nodes.

In previous traditional algorithms the routing algorithm merges hierarchical routing that is performed in greedy habitats. The method of data packet sending from the source peers and the destination location to the base station have two phases such as inter cluster routing and intra cluster routing.

2.2 Related Work

An area-based administration (LBS) is a product level administration that utilizations area information to control highlights. In that capacity, LBS is a data benefit and has various utilizations in long range informal communication today as data, in amusement or security, which is available with cell phones through the portable system and which utilizes data on the geological position of the versatile device.[6].LBS can be utilized as a part of an assortment of connections, for example, wellbeing, indoor item search entertainment, work, individual life, etc.

LBS is basic to numerous organizations and in addition government associations to drive genuine understanding from information fixing to a particular area where exercises happen. The spatial examples that area related information and administrations can give are situated within one on its most intense and valuable perspective where the area is a shared factor in these exercises and can be utilized to better comprehend examples and connections [7]. LBS incorporate administrations to distinguish a location [8] of a man or question, for example, finding the closest saving money machine (ATM) or the whereabouts of a companion or worker. LBS incorporate bundle following and vehicle following administrations. LBS can incorporate versatile trade when appearing as coupons or promote coordinated at clients in light of their present area. They incorporate customized climate benefits and even area based diversions. They are a case of telecom meeting.

Database frameworks use files to rapidly gaze upward values and the way that most databases list information is not ideal for spatial questions. Rather, spatial databases utilize a spatial file to accelerate database operations. Notwithstanding common SQL inquiries, for example, SELECT explanations, spatial databases can play out a wide assortment of spatial operations. The accompanying operations and numerous more are determined by the Open Geospatial Consortium standard: [8]

Spatial Measurements: Computes line length, polygon region, the separation between geometries, and so forth.

Spatial Functions: Modify existing elements to make new ones, for instance by giving a support around them, crossing highlights, and so on. [9]

Spatial Predicates: Allows genuine/false inquiries about spatial connections between geometries. Illustrations incorporate "do two polygons cover" or 'is there a home situated inside a mile of the region.

Geometry Constructors: Creates new geometries, normally by determining the vertices (focuses or hubs) which characterize the shape.

Spectator Functions: Queries which return particular data around an element, for example, the area of the focal point of a circle.

A few databases bolster just rearranged or altered arrangements of these operations, particularly in instances of NoSQL frameworks like MongoDB and CouchDB.

A quadtree is a tree data structure in which each internal node has exactly four children. Quadtrees are most often used to partition a two-dimensional space by recursively subdividing it into four quadrants or regions. The regions may be square or rectangular, or may have arbitrary shapes. This data structure was named a quadtree by Raphael Finkel and J.L. Bentley in 1974. A similar partitioning is also known as a Q-tree. All forms of quadtrees share some common features: They decompose space into adaptable cellshas a maximum capacity. When maximum capacity is reached, the bucket splits. The tree directory follows the spatial decomposition of the quadtree[10].

2.3 Existing System

Various traditional approaches proposed by various authors from years of research, every approach has its own advantages and disadvantages. Performance and time complexity are the major factors while community searches. Nodes should be grouped based on the weights and edges existence between the nodes. Traditional community based approaches is more complex to group from the source node and there is no further practical search implementation. Identification of neighbor with simple edge does not retrieve optimality.

2.3.1 Drawbacks of Existing System

1. More time complexity if route API computes path with all available nodes for every request.
2. Less performance and additional overhead to location-based service.
3. User may receive irrelevant results if response if slow.

2.3.2 Method Used in Existing System

KNN Query Algorithm

In this section, we extend our Route-Saver algorithm for processing KNN queries. We will also examine suitable orderings for processing candidates. Unlike range queries, KNN queries do not have a (fixed) travel time limit T for obtaining a small candidate set. Instead, we first compute a (temporary) result set R so that it contains K candidates with the smallest p:tþ G or p:tG. Recall that we can obtain these bounds/values for all candidates efficiently by two Dijkstra traversal on G. Let g be the largest p:tþ G or p:tG in R. Having this value g, we can prune each candidate p that satisfies p:t > g, as it cannot become the result. Algorithm 2 is the pseudo-code

of our KNN algorithm. First, we initialize the candidate set C with the data set P, insert K dummy pairs (with 1 travel time) into the result set R, and set g to the largest travel time in R. The algorithm consists of three phases. In the first phase, it obtains g by using the idea discussed above. In the second phase, it prunes candidates whose lower bounds or exact times are larger than g. In the third phase, it examines the candidates according to a certain order and issues route requests for them. The algorithm terminates when the candidate set contains exactly K objects, and then reports them as query results.

1.

Algorithm 2: Route-Saver Algorithm for KNN Queries

function Route-Saver-KNN (Query (q, K), Data set $\mathcal{P}$)
▷*system parameters*: time-tagged graph G, route log $\mathcal{L}$, expiry time δ

Remove from the log $\mathcal{L}$ any route ψ_t with $t < t_{now} - \delta$
Create a candidate set $C \leftarrow \mathcal{P}$
Create a result set R with K pairs $\langle NULL, \infty \rangle$
$\gamma \leftarrow$ the largest travel time in R
Run Dijkstra for q on G using $\omega^+(e)$ and $\omega^-(e)$ to retrieve $p.\tau_G^+, p.\tau_G^-, p.\tau_G$ ▷Phase 1: obtain the threshold γ
for each $p \in C$ **do**
    Update R, γ by p with $p.\tau_G^+$ or $p.\tau_G$
for each $p \in C$ **do** ▷Phase 2: prune objects
    if $p.\tau_G > \gamma$ or $p.\tau_G^- > \gamma$ **then**
        Remove p from C
    if $\exists$ route $\psi \in \mathcal{L}$ such that ψ contains p and q **then**
        Compute $p.\tau_{\mathcal{L}}$ ▷optimal subpath property [15], [31]
        Update R, γ by p with $p.\tau_{\mathcal{L}}$
    Compute $p.\tau^-$ as $max\{p.\tau_G^-, p.\tau_I^-\}$
    if $p.\tau^- > \gamma$ or ($p.\tau_{\mathcal{L}}$ is known and $p.\tau_{\mathcal{L}} > \gamma$) **then**
        Remove p from C
while $|C| > K$ **do** ▷Phase 3: Issue route requests
    Pick an object $p \in C$ with minimum $p.\tau^-$ ▷ordering
    Route $\psi_{t_{now}} \leftarrow$ **RouteRequest**(q, p) ▷call external API
    Insert $\psi_{t_{now}}$ into $\mathcal{L}$; Update $\omega(e), \mu(e)$ in G for $e \in \psi_{t_{now}}$
    Update $p.\tau_{\mathcal{L}}$ for all p on $\psi_{t_{now}}$ ▷optimal subpath property [15]
    Run incremental Dijkstra to update all $p.\tau^-$ ▷By [14]
    for each $p' \in C$ **do**
        if $p'.\tau^- > \gamma$ or $p'.\tau_{\mathcal{L}} > \gamma$ **then**
            Remove p' from C
        if $p'.\tau_{\mathcal{L}} < \gamma$ **then**
            Update R by p' with $p'.\tau_{\mathcal{L}}$
Return R

2. Consider the KNN query with K ¼ 3 in Fig. We illustrate the running steps of Route-Saver in Table 2. Entries without values are marked as '/'. In the first phase, we derive the upper bounds p:tþ G; p:tG; p:t G using the time-tagged road graph G, which are shown in the first three columns . Since p1:tG, p4:tG and p5:tþ G are the smallest three travel times, we insert them into R and update g ¼ 40. In the second phase, first we prune candidates p2; p5; p6 since their p:t G are larger than g. Then, we calculate the lower-bound travel time for p7 using L: p7:t I ¼ 41 > g, so p7 is pruned. We skip the third phase as the candidate set contains exactly K ¼ 3 objects, the same as the result set R. Thus, the algorithm returns R ¼ fp1; p4; p5g as the query result. Route-Saver issues 0 route request in this example. On the other hand, SMashQ incurs seven route requests when solving this query.

2.4 Proposed System

We proposing an efficient Cluster based route implementation and search implementation, nodes can be grouped with clustering approach based on weights and edges, in terms of graphical nodes and edges format. A node attracts its neighboring individuals to be a part of its path computation. Those that find enough connectivity may choose to stay. The communities then expand further as the process is iterated by the newly added members. Search implementation can be done that nearest neighbors, if does not get the required limit, moves to next search community. The main advantages here, Clustering process reduces the number of nodes while computation of paths and irrelevant paths can be ignored and less time complexity, user can receive query results in optimal time

2.4.1 Advantages

1. Clustering process reduces the number of nodes while computation of paths.
2. Irrelevant paths can be ignored.
3. Less time complexity, user can receive query results in optimal time.

Path generator receives a request from the mobile user through the intermediate server and gives response as optimal path to the requested user. Path can be computed with clustering mechanism, most of the routing algorithms compute all possible paths from the source to destination, it is very time complexity implementation, here cluster-based implementation minimizes the time to generate clusters for path.

Generally, nodes are placed in various zones or nodes can be clustered base on the location and time duration between the nodes, these nodes are iteratively clustered, now source and destination available nodes only can be used in path computation, from these clusters we can eliminate unnecessary nodes during path computation and computes optimal path base on time duration between nodes and forwards to intermediate server.

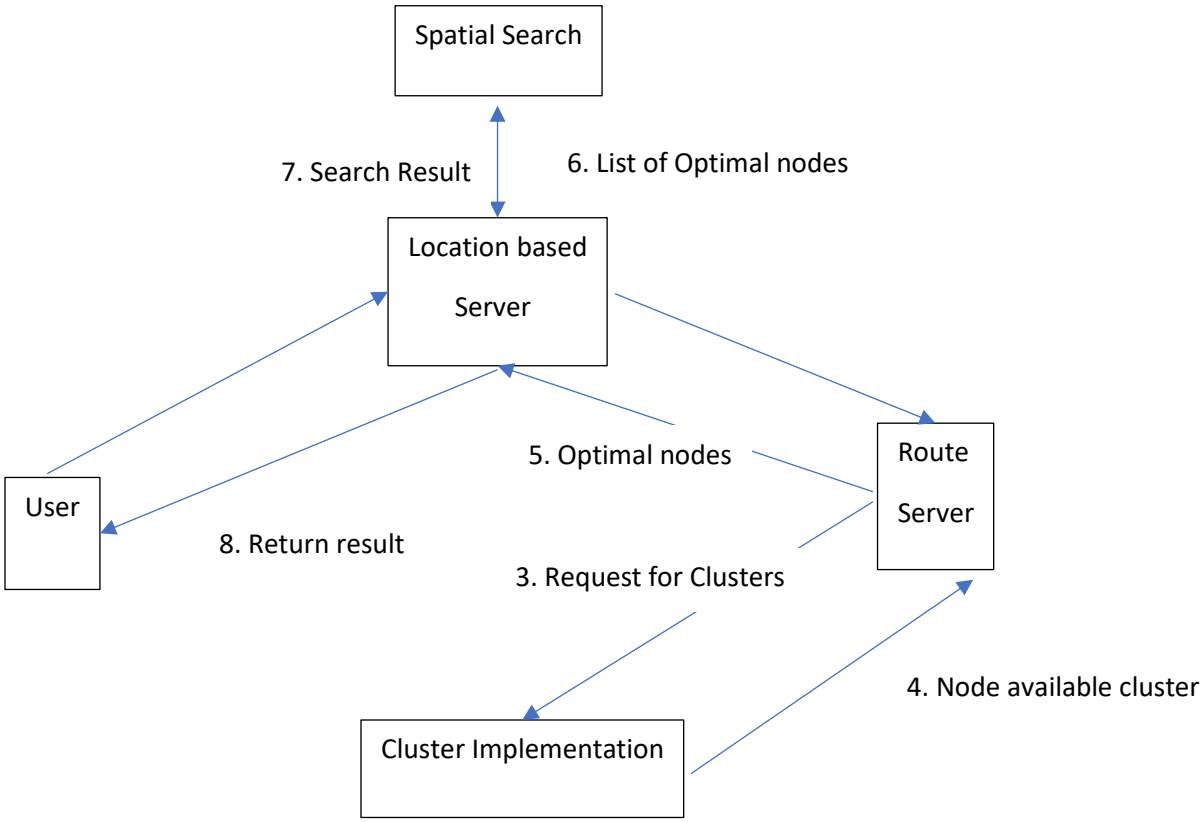


Fig 1: Data Flow Diagram

2.4.2 Algorithms

K Means Cluster Implementation:

Step1: Consider K number of points as prior centroids for first iteration.

Step 2: Continue the process until it meets the user specified maximum number of iterations.

Step 3: Compute Euclidean distance (ED) between random data point and centroid.

Step 4: Assign each data point to its nearest centroid point by computing the Euclidean distance to form K number of clusters.

Step 5: Re generate new centroids within individual clusters.

Step 6: .Continue the process or steps from 2 to 5

User makes a request to the location-based service, in turn it returns the query based results with respect to features of the object. User receives user interesting results based on the features of the requested query. Cache can be maintained if user makes the same request with minimum time duration and latitude and longitude parameters.

Location Query Search implementation:

Input: Qi—Input Spatial Query, DO_{list}(Total Data objects)

Output: R_{list} (result set)

1. User provides the spatial query which involves the spatial object and feature
2. Load DO_{list} from database(LBS)
3. For i=0;i<List_Nodes ;i++

    For each Object O in DO_{list}

    If(O== Q_i.objectname)

        Add to Object_List

    Next

    For each object O in Object_List

        If (O.attribute==Q_i.attribute)

    Add 'O' to R_{list}

    Next

Next

4. Sort the Result set
5. Return R_{list}

LBS receive the request from the user and forwards the geocoding's of the user to route generator and receives the shortest or optimal path and retrieve the object-based results from the nodes by using road network, it maintains the objects with features. LBS retrieve the results from the nodes and check the required threshold, if it does not meet check for next immediate neighbor, until it meets threshold value.

2.5 Feasibility Study

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time.

There are aspects in the feasibility study portion of the preliminary investigation.

- Economical Feasibility
- Technical Feasibility
- Operational Feasibility

2.5.1 Economical Feasibility

As system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the

development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies java SDK 1.6 open source, There is nominal expenditure and economic feasibility for certain.

2.5.2 Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following-

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

2.5.3 Technical Feasibility

Here we used one-dimensional array and a two-dimensional array to speed up the process of the first and the second pass correspondingly. The support counting phase runs very fast by using an array, since no searching is needed. When a large number of two item sets are present in the used data, the Prune and Recovery methods help a lot in reducing the subsets to participate in the further passes, which accomplishes the task in a fast and efficient manner and also I am implementing in Java which is a platform independent language with a strong set of collections, which contain many utility classes. Java is simple, secure, portable, object-oriented, robust, multithreaded, architecture neutral, interpreted, high performance, distributed and dynamic. Java is easy to learn which makes it simple.

2.6 Software Requirement Specification

A Software Requirements Specification (SRS) is a complete description of the behaviour of the system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. Use cases are also known as functional requirements. In addition to use cases, the SRS also contains non-functional (or supplementary) requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

2.6.1 Functional Requirements:

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the

cases where the system uses the functional requirements are captured in use cases. Functional requirements are supported by non-functional requirements (also known as quality requirements), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). How a system implements functional requirements is detailed in the system design. In some cases a requirements analyst generates use cases after gathering and validating a set of functional requirements. Each use case illustrates behavioral scenarios through one or more functional requirements. Often, though, an analyst will begin by eliciting a set of use cases, from which the analyst can derive the functional requirements that must be implemented to allow a user to perform each use case.

Functional requirements of this project are as shown below:

1. Initiate all nodes with x axis and y axis
2. Forward source node location and nearest neighbor location to location based server
3. LBS intern forwards to route server
4. Route server computes the path and forwards to LBS
5. LBS searches for input query and forwards the result
6. Results can be forwarded to end user to view

2.6.2 Non-Functional Requirements:

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behavior or functions In general, functional requirements define what a system is supposed to do whereas non-functional requirements define how a system is supposed to be. Non-functional

requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals" and "quality of service requirements," and "non-behavioral requirements."[1] Qualities, that is, non-functional requirements, can be divided into two main categories:

1. Execution qualities, such as security and usability, which are observable at run time.
2. Evolution qualities, such as testability, maintainability, extensibility and scalability, which are embodied in the static structure of the software system.

2.6.3 Hardware and Software Requirements

Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Hardware Requirements for Present Project:

1. VDU: Monitor/ LCD TFT / Projector
2. Input Devices: Keyboard and Mouse
3. RAM: 512 MB
4. Processor: P4 or above
5. Storage: 10 to 100 MB of HDD space.

Software Requirements

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Software Requirements for Present Project:

1. Operating System: Windows XP or above
2. IDE: visual studio 2010
3. Language : C#

CHAPTER 3

SYSTEM DESIGN

3.1 Introduction

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development[3], then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing a system to satisfy specified requirements of the user. Until the 1990s systems design had a crucial and respected role in the data processing industry. In the 1990s standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering. Object-oriented analysis and design methods are becoming the most widely used methods for computer system design [citation needed]. The UML has become the standard language used in Object-oriented analysis and design [citation needed]. It is widely used for modeling software systems and is increasingly used for high designing non-software systems and organizations.

Fig 2 : Factors contributing for SRS of a project

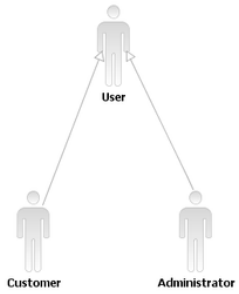
3.2 Unified Modelling Language (UML) Diagrams

3.2.1 Use Case Diagram

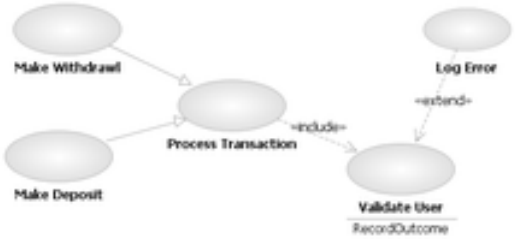
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

Use Case diagrams are formally included in two modeling languages defined by the OMG: the Unified Modeling Language (UML) and the Systems Modeling Language (SysML).

Diagram building blocks



Actor inheritance



Use case relationships

Interaction among actors is not shown on the use case diagram. If this interaction is essential to a coherent description of the desired behavior, perhaps the system or use case boundaries should be re-examined. Alternatively, interaction among actors can be part of the assumptions used in the use case.

Actor Generalization

One popular relationship between Actors is Generalization/Specialization. This is useful in defining overlapping roles between actors. The notation is a solid line ending in a hollow triangle drawn from the specialized to the more general actor.[1][2][3]

Use Case Relationships

Three relationships among use cases are used often in practice.

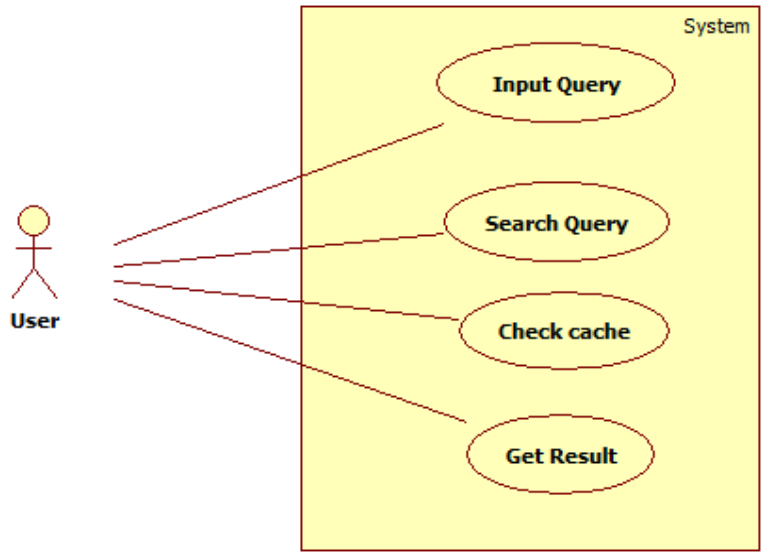


Fig 3: User Use Case Diagram

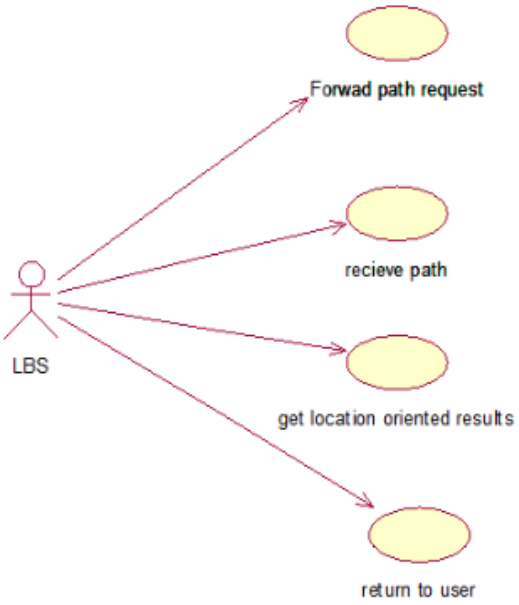


Fig 4: LBS Use Case Diagram

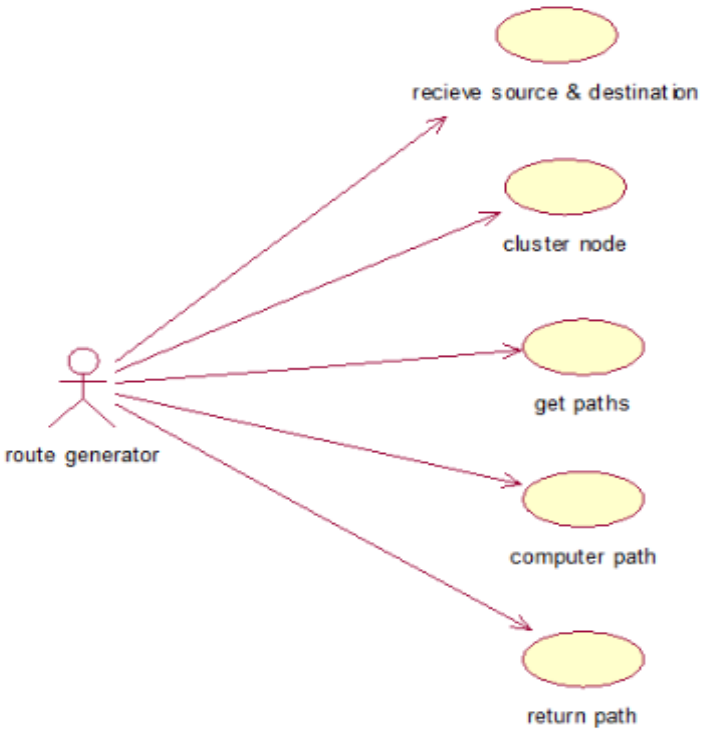


Fig 5 : Route Generator Use Case Diagram

3.2.2 Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

For instance, the UML 1.x diagram on the right describes the sequences of messages of a (simple) restaurant system. This diagram represents a Patron ordering food and wine, drinking wine then eating the food, and finally paying for the food. The dotted lines extending downwards indicate the timeline. Time flows from top to bottom. The arrows represent messages (stimuli) from an actor or object to other objects. For example, the Patron sends message 'pay' to the Cashier. Half arrows indicate asynchronous method calls. The UML 2.0 Sequence Diagram supports similar notation to the UML 1.x Sequence Diagram with added support for modeling variations to the standard flow of events.

Diagram building blocks

If the lifeline is that of an object, it demonstrates a role. In order to display interaction, messages are used. These are horizontal arrows with the message name written above them. Solid arrows with full heads are synchronous calls, solid arrows with stick heads are asynchronous calls and dashed arrows with stick heads are return messages. This definition is true as of UML 2, considerably different from UML 1.x.

Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message (Execution Specifications in UML).Objects calling methods on themselves use messages and add new activation boxes on top of any others to indicate a further level of processing. When an object is destroyed (removed from memory), an X is drawn on top of the lifeline, and the dashed line ceases to be drawn below it (this is not the case in the first example though). It should be the result of a message, either from the object itself, or another. A message sent from outside the diagram can be represented by a message originating from a filled-in circle (found message in UML) or from a border of sequence diagram (gate in UML).

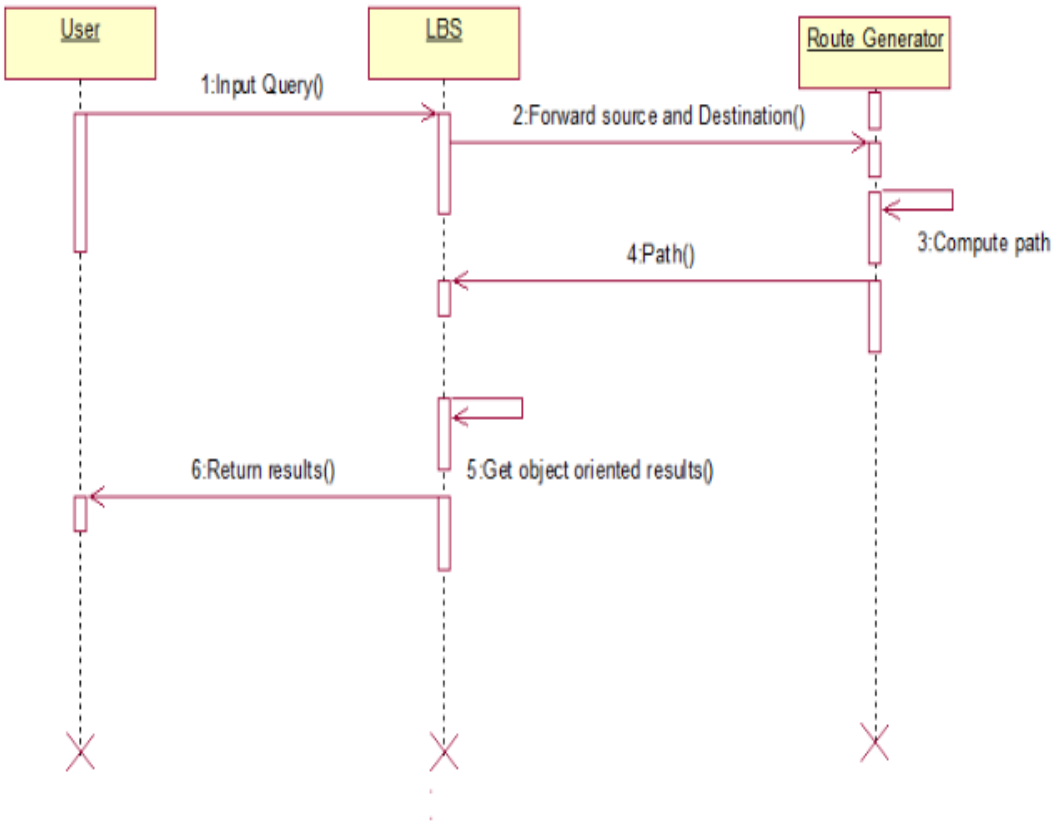


Fig 6 : Sequence Diagram

3.2.3 Collaboration Diagram

A Collaboration diagram is very similar to a Sequence diagram in the purpose it achieves; in other words, it shows the dynamic interaction of the objects in a system. A distinguishing feature of a Collaboration diagram is that it shows the objects and their association with other objects in the system apart from how they interact with each other. The association between objects is not represented in a Sequence diagram. A Collaboration diagram is easily represented by modeling objects in a system and representing the associations between the objects as links.

The interaction between the objects is denoted by arrows. To identify the sequence of invocation of these objects, a number is placed next to each of these

Elements of a Collaboration diagram

A Collaboration diagram consists of the following elements:

Element and its description	Symbol
Object: The objects interacting with each other in the system. Depicted by a rectangle with the name of the object in it, preceded by a colon and underlined.	:ObjectName
Relation/Association: A link connecting the associated objects. Qualifiers can be placed on either end of the association to depict cardinality.	0..* 1..*
Messages: An arrow pointing from the commencing object to the destination object shows the interaction between the objects. The number represents the order/sequence of this interaction.	1:Function ()

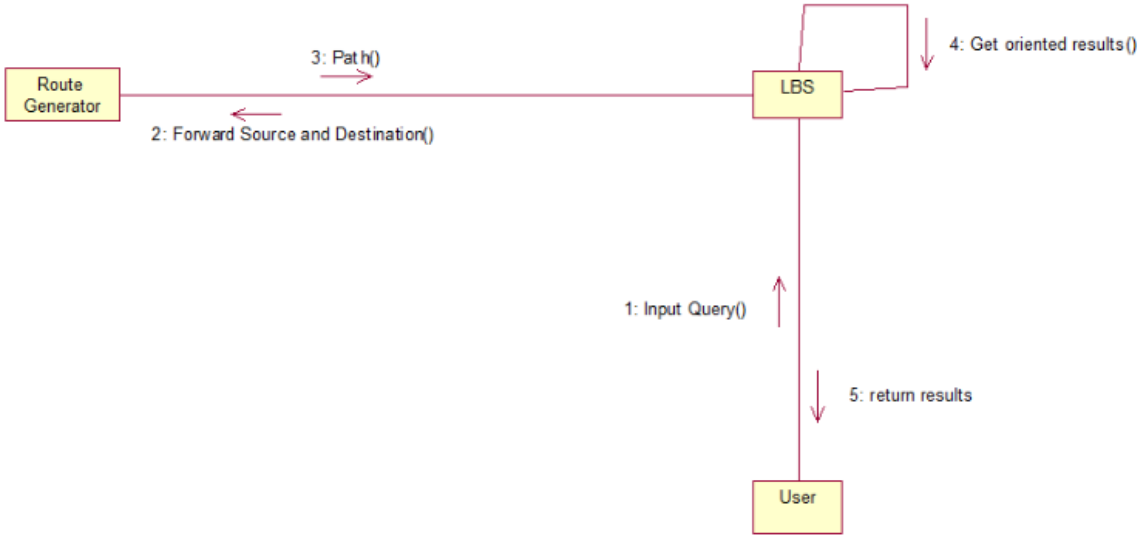


Fig 7: Collaboration Diagram

3.2.4 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. [1] In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows. The most important shape types:

* Rounded rectangles represent activities;

* Diamonds represent decisions;

* Bars represent the start (split) or end (join) of concurrent activities;

* A black circle represents the start (initial state) of the workflow;

* An encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen. Hence, they can be regarded as a form of flowchart. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops. While in UML 1.x, activity diagrams were a specialized form of state diagrams, in UML 2.x, the activity diagrams were renormalized to be based on Petri net-like semantics, increasing the scope of situations that can be modeled using activity diagrams. These changes cause many UML 1.x activity diagrams to be interpreted differently in UML 2.x

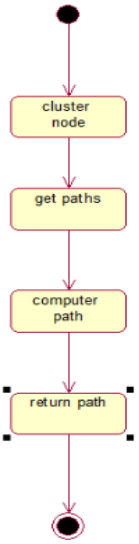


Fig 8: Activity Diagram

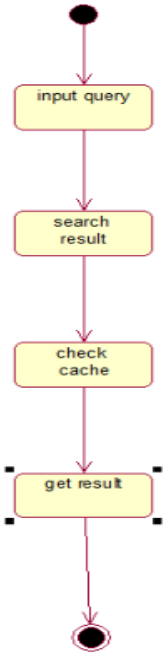


Fig 9: Activity Diagram

3.2.5 State Chart Diagram

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. There are many forms of state diagrams, which differ slightly and have different semantic state diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented in series of events that could occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and track the different states of its objects through the system".

State diagrams can be used to graphically represent finite state machines. This was introduced by Taylor Booth in his 1967 book "Sequential Machines and Automata Theory". Another possible representation is the State transition table.

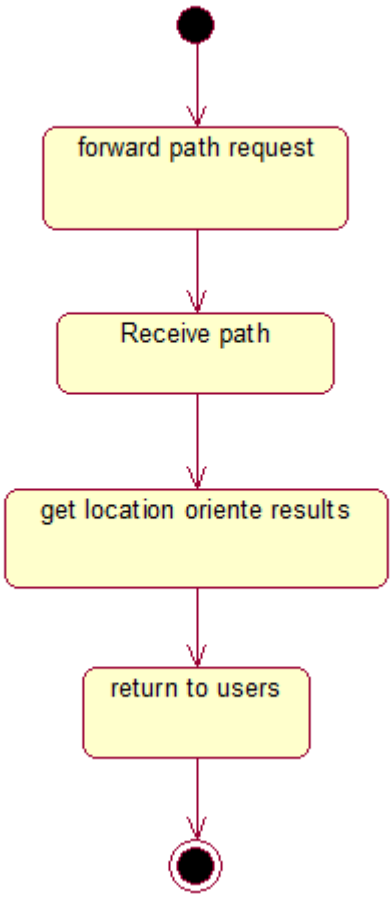


Fig 10 : State Diagram

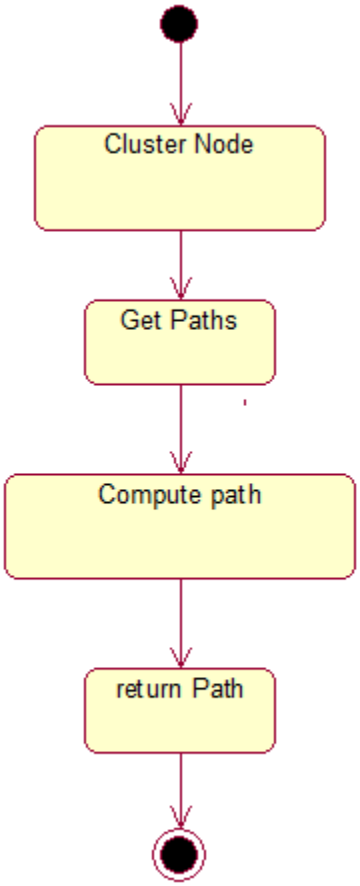


Fig 11 : State Diagram

3.2.6 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. The class diagram is the main building block in object oriented modelling. They are being used both for general conceptual modelling of the systematics of the application, and for detailed modelling translating the models into programming code. The classes in a class diagram represent both the main objects and or

interactions in the application and the objects to be programmed. In the class diagram these classes are represented with boxes which contain three parts:

A class with three sections.

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

In the system design of a system, a number of classes are identified and grouped together in a class diagram which helps to determine the statical relations between those objects. With detailed modeling, the classes of the conceptual design are often split in a number of subclasses.

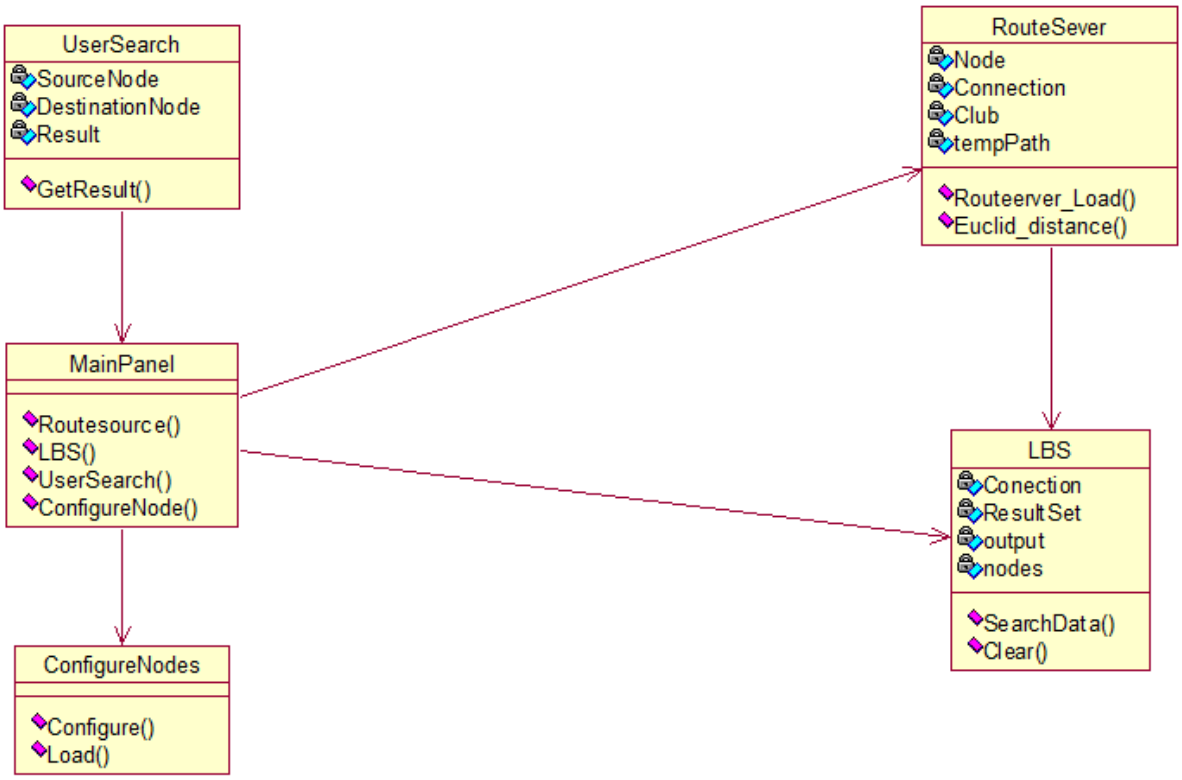


Fig 12: Class Diagram

CHAPTER 4

TESTING

4.1 Introduction

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software testing can also be stated as the process of validating and verifying that a software program/application/product:

1. meets the business and technical requirements that guided its design and development;

2. Works as expected; and

3. Can be implemented with the same characteristics.

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers. In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

Testing can never completely identify all the defects within software. Instead, it furnishes a criticism or comparison that compares the state and behavior of the product against oracles—principles or mechanisms by which someone might recognize a problem. These oracles may include (but are not limited to) specifications, contracts,[2] comparable products, past versions of the same product, inferences about intended or expected purpose, user or customer expectations, relevant standards, applicable laws, or other criteria.

Every software product has a target audience. For example, the audience for video game software is completely different from banking software. Therefore, when an organization develops or otherwise invests in a software product, it can assess whether the software product will be acceptable to its end users, its target audience, its purchasers, and other stakeholders. Software testing is the process of attempting to make this assessment.

A study conducted by NIST in 2002 reports that software bugs cost the U.S. economy $59.5 billion annually. More than a third of this cost could be avoided if better software testing was performed.

4.2 Testing Methods

The box approach

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

4.2.1 White Box Testing

White box testing is when the tester has access to the internal data structures and algorithms including the code that implement these.

Types Of White Box Testing

The following types of white box testing exist:

* API testing (application programming interface) - testing of the application using public and private APIs

* Code coverage - creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)

* Fault injection methods - improving the coverage of a test by introducing faults to test code paths

* Mutation testing methods

* Static testing - White box testing includes all static testing

Test coverage

White box testing methods can also be used to evaluate the completeness of a test suite that was created with black box testing methods. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. [21]

Two common forms of code coverage are:

* Function coverage, which reports on functions executed

* Statement coverage, which reports on the number of lines executed to complete the test

4.2.2 Black Box Testing

Black box testing treats the software as a "black box"—without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing and specification-based testing.

Specification-based testing: Specification-based testing aims to test the functionality of software according to the applicable requirements. [22] Thus, the tester inputs data into, and only sees the output from, the test object. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either "is" or "is not" the same as the expected value specified in the test case.

Specification-based testing is necessary, but it is insufficient to guard against certain risks.

Advantages and disadvantages: The black box tester has no "bonds" with the code, and a tester's perception is very simple: a code must have bugs. Using the principle, "Ask and you shall receive," black box testers find bugs where programmers do not. But, on the other hand, black box testing has been said to be "like a walk in a dark labyrinth without a flashlight," because the tester doesn't know how the software being tested was actually constructed. As a result, there are situations when (1) a tester writes many test cases to check something that could have been tested by only one test case, and/or (2) some parts of the back-end are not tested at all.

Therefore, black box testing has the advantage of "an unaffiliated opinion," on the one hand, and the disadvantage of "blind exploring," on the other. [24]

CHAPTER 5

SAMPLE CODE

Cluster.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Intrusiondetection
{
   public partial class Cluster : Form
   {
      SqlConnection con = new SqlConnection(Common.conn);
      List<DataItem> items = new List<DataItem>();
      DataTable dt;
      public Cluster()
      {
         InitializeComponent();
      }

      private void Cluster_Load(object sender, EventArgs e)
      {
         dt = new DataTable();
```

```
    con.Open();
    new SqlDataAdapter("select * from Nodelocations", con).Fill(dt);
    con.Close();
    dataGridView1.DataSource = dt;
    for (int i = 0; i < dt.Rows.Count; i++)
    {
       DataItem item = new DataItem();

       item.Nodeid = dt.Rows[i][0].ToString();
       item.Lat =Convert.ToInt32( dt.Rows[i][1].ToString());
       item.Lon = Convert.ToInt32(dt.Rows[i][2].ToString());
       items.Add(item);

    }

}

private void button2_Click(object sender, EventArgs e)
{

    List<DataItem> centroids = new List<DataItem>();
    Random r=new Random();
    List<int> ids = new List<int>();
    for (int i = 0; i < 3; i++)
    {
     int temp=r.Next(items.Count);
     if (!ids.Contains(temp))
     {
        ids.Add(temp);
        DataItem item = items.ElementAt(temp);
        centroids.Add(item);
     }
     else
        i = i - 1;
    }
    // compute distance between items and centroids
```

```
con.Open();
  new SqlCommand("delete from zones", con).ExecuteNonQuery();
con.Close();
double mincost = 0;
double cost = 0;
int minid = 0;
int clusterid=0;
for (int i = 0; i < items.Count; i++)
{
   mincost=Common.Euclideandist(items.ElementAt(i),centroids.ElementAt(0));
   minid = i;
   clusterid = 0;
   for(int j=1;j<centroids.Count;j++)
   {
   cost=Common.Euclideandist(items.ElementAt(i),centroids.ElementAt(j));
      if(cost<mincost)
      {
         mincost=cost;
         minid = i;
         clusterid=j;
      }
   }
   string tempzone="";
   tempzone="Zone"+clusterid;
   con.Open();
   new SqlCommand("insert into zones values('" + tempzone + "','" +
items.ElementAt(minid).Nodeid + "')", con).ExecuteNonQuery();
   con.Close();
}
```

```
            con.Open();

            DataTable dt2 = new DataTable();
            new SqlDataAdapter("select * from Zones order by zone", con).Fill(dt2);
            con.Close();
            dataGridView2.DataSource = dt2;
            con.Open();
            SqlDataReader dr2;
            dr2 = new SqlCommand("select distinct Zone from Zones", con).ExecuteReader();
            while (dr2.Read())
                comboBox1.Items.Add(dr2[0].ToString());
            con.Close();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            Common.Zoneid = comboBox1.SelectedItem.ToString();
            Classification_Module cm = new Classification_Module();
            cm.ShowDialog();
        }
    }
}
```

MasterNode.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
using System.Data.SqlClient;

using System.Threading;
using System.Collections;
using System.Security.Cryptography;
namespace Intrusiondetection
{
   public partial class MasterNode : Form
   {
      SqlConnection con = new SqlConnection(Common.conn);
      System.Windows.Forms.Timer timer = new System.Windows.Forms.Timer();
      SqlDataReader dr;
      ArrayList nodelist = new ArrayList();
      ArrayList keylist = new ArrayList();
      Boolean status = false;
      public static int usercount = 0;
      public static Boolean refresh = false;
      public static int secretNUmber = 0;
      public static string secretShare = null;
      public static int sig = 0;
      public MasterNode()
      {
         InitializeComponent();
         timer.Tick += new EventHandler(timer_Tick); // Everytime timer ticks, timer_Tick will be called
         timer.Interval = (1000) * (4);            // Timer will tick evert 10 seconds
         timer.Enabled = true;                    // Enable the timer
         timer.Start();
      }
      private void MasterNode_Load(object sender, EventArgs e)
      {
```

```
  // InitializeComponent();
   SqlConnection con1 = new SqlConnection(Common.conn);
   con1.Open();
   SqlCommand cmd2 = new SqlCommand("delete from NSignature", con1);
   cmd2.ExecuteNonQuery();
   con1.Close();
}
void timer_Tick(object sender, EventArgs e)
{
   String messagedetails = "";
   String temp = "";
   string key = "";
      string tempSign = "";
      SqlConnection con1 = new SqlConnection(Common.conn);
      con1.Open();
      dr = new SqlCommand("select * from Nodes", con1).ExecuteReader();
    while (dr.Read())
      {
         usercount = usercount + 1;
        temp = dr[0].ToString();
        if (!nodelist.Contains(temp))
        {
           nodelist.Add(temp);
           Common.node_list.Add(temp);
           key = dr[1].ToString();
           keylist.Add(key);
           textBox1.AppendText(temp + "\n");
           textBox2.AppendText("Secret seed  value  from " + temp + " is " + key +"\n");
        }
      }
```

```
        dr.Close();


            con1.Close();
            Common.n = nodelist.Count;
            if (usercount != nodelist.Count)
            {
                nodelist.Clear();
                keylist.Clear();
                Common.node_list.Clear();
                refresh = true;
            }
            usercount = 0;
            SqlConnection con2 = new SqlConnection(Common.conn);
            con2.Open();
           // SqlDataReader dr3;
            if (sig != 0)
            {
                SqlCommand cmd1 = new SqlCommand("select NodeId,SecretKey,Nodesignature from
NSignature", con2);
               dr= cmd1.ExecuteReader();
                while(dr.Read())
                {
                    if(dr[0]!=DBNull.Value)
                    textBox2.AppendText("Node " + dr[0].ToString() + " signature is: " + dr[2].ToString()+"\n");
                }
            }
            con2.Close();
            dr.Close();
            sig = 0;
        //}
```

```
   // // End try
   //catch (Exception ex)
   //{
   //    label1.Text = ex.ToString();
   //}// Alert the user
}
private void button1_Click(object sender, EventArgs e)
{
    string node = "";
    int keyval = 1 ;
   for (int j = 0; j < keylist.Count; j++)
   {
      keyval = keyval * Convert.ToInt32(keylist[j]);
   }
       if (refresh == true)
       {
          con.Open();
          new SqlCommand("delete from  keydetails", con).ExecuteNonQuery();
          con.Close();
          refresh = false;
          textBox2.AppendText("----New Key Generation------ \n");
       }
       con.Open();
       new SqlCommand("delete from  keydetails", con).ExecuteNonQuery();
       con.Close();
   for (int i = 0; i < nodelist.Count; i++)
   {
```

```
            //textBox2.AppendText("Key Generated at  " + nodelist[i] + " is" + keyval + "\n");
            con.Open();
            new SqlCommand("insert into  keydetails values('" + nodelist[i] + "' ,'" + keyval + "')",
con).ExecuteNonQuery();
            con.Close();
        }
    }
    private void button2_Click(object sender, EventArgs e)
    {
        Random rnd = new Random();
        int randomNumber = rnd.Next(1,100);
        secretNUmber = randomNumber;
        MD5 md5 = MD5.Create();
        byte[] hashData = md5.ComputeHash(Encoding.Default.GetBytes(randomNumber.ToString()));
        StringBuilder returnValue = new StringBuilder();
        for (int i = 0; i < hashData.Length; i++)
        {
            returnValue.Append(hashData[i].ToString());
        }
        secretShare = returnValue.ToString();
        textBox3.Text = randomNumber.ToString();
        button3.Visible = true;
    }
    private void button3_Click(object sender, EventArgs e)
    {
        Node.secretshare = Convert.ToInt32(textBox3.Text);
        button4.Visible = true;
    }
```

```
        private void button4_Click(object sender, EventArgs e)
        {
            SqlDataReader dr1;
            int status = 0;
            con.Open();
            SqlCommand cmd = new SqlCommand("select NodeId,NodeSignature from NSignature",con);
            dr1 = cmd.ExecuteReader();
            while(dr1.Read())
            {
                if (secretShare == dr1[1].ToString())
                    status =status+ 1;
                else
                    status = -1;
            }
            if (status > 0)
                MessageBox.Show("Verified Successfully");
            else if(status < 0)
                MessageBox.Show("Verified UnSuccessfull");
            con.Close();
            button1.Visible = true;
        }
    }
}
```

Node.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Security.Cryptography;
using System.IO;
using System.Threading;
using System.Collections ;
namespace Intrusiondetection
{
   public partial class Node : Form
   {
      string inputFile = "", inFile = "";
      int count = 0;
      string tempFile = "";
      string sign = "";
      int fLen = 0;
      string path = "";
      SqlConnection con = new SqlConnection(Common.conn);
      System.Windows.Forms.Timer timer = new System.Windows.Forms.Timer();
      SqlDataReader dr;
      string srcid = "", destid = "", protocol = "", packets = "", msg = "";
      Boolean status = false;
      ArrayList authorized = new ArrayList();
     public  static string cipher = "",signature="";
     public static string remVector = "",qntVector="";
     public static int secretshare = 0,finalkey=0;
      public Node()
      {
```

```
        InitializeComponent();
        timer.Tick += new EventHandler(timer_Tick); // Everytime timer ticks, timer_Tick will be called
        timer.Interval = (1000) * (4);            // Timer will tick evert 10 seconds
        timer.Enabled = true;                     // Enable the timer
        timer.Start();                            // Start the timer
    }
    void timer_Tick(object sender, EventArgs e)
    {
        if(secretshare!=0)
            label10.Text = secretshare.ToString();
        String key = "";
        //  MessageBox.Show(Adduser.myuserid);
        con.Open();
        dr = new SqlCommand("select finalkey from keydetails where node='" + label2.Text + "'",
con).ExecuteReader();
        if (dr.Read())
        {
            textBox6.Text = dr[0].ToString();
          //  textBox3 .AppendText ("Generated Key "+textBox6.Text );
        }
        dr.Close();
        con.Close();
        button2.Enabled = true;
        con.Open();
        string node="";
        dr = new SqlCommand("select reminderVector,QntVector,Node,type from message where
node='"+label2 .Text +"'", con).ExecuteReader();
        if (dr.Read())
        {
            node=dr[2].ToString();
            if (node == label2.Text)
```

```
            textBox4.Text = "Reminder Vector: "+dr[0].ToString()+" ,Quotient Vector"+dr[1].ToString();
           //  string msg=Decrypt (dr[0].ToString(),textBox6 .Text );
          //  textBox3.AppendText("Msg from  " + node + " is " +msg  +"\n" );
        }
        con.Close();
     //   Thread.Sleep(1000);
        con.Open();
        new SqlCommand("delete from message where node='" + label2 .Text + "'",
con).ExecuteNonQuery();
        con.Close();
    }
    private void Node_Load(object sender, EventArgs e)
    {
        textBox5.Text = Form1.key;
        label2.Text = Form1.nodeid;
        textBox3.AppendText("Secret seed at Node " + label2.Text + " is " + textBox5.Text +"\n");
        con.Open();
        new SqlCommand("insert into Nodes values('" + label2.Text + "','"+textBox5 .Text +"','')",
con).ExecuteNonQuery();
        //   new SqlCommand("update Nodes set key='" + textBox1.Text + "' where node='" + label2.Text
+ "'", con).ExecuteNonQuery();
        con.Close();
    }
    private void button4_Click(object sender, EventArgs e)
    {
        int x = MasterNode.secretNUmber;
        MD5 md5 = MD5.Create();
        byte[] hashData = md5.ComputeHash(Encoding.Default.GetBytes(x.ToString()));
        StringBuilder returnValue = new StringBuilder();
        for (int i = 0; i < hashData.Length; i++)
        {
```

```
            returnValue.Append(hashData[i].ToString());
        }
        textBox8.Text = returnValue.ToString();
        con.Open();
        SqlCommand cm = new SqlCommand("INSERT INTO NSignature(NodeId, SecretKey,
NodeSignature) values ('"+label2 .Text+"','"+x+"','"+textBox8.Text+"')",con);
        cm.ExecuteNonQuery();
        con.Close();
        //label10.Text = computekey(x, Common.n, Common.N).ToString() ;
        //con.Open();
        ////  new SqlCommand("insert into Nodes values('" + label2.Text + "','')",
con).ExecuteNonQuery();
        //new SqlCommand("update Nodes set subkey='" + label10 .Text  + "' where node='" + label2.Text
+ "'", con).ExecuteNonQuery();
        //con.Close();
        //label6.Text = "Forwarded sucessfully";
    }
    int res = 0;
    public int computekey(int x, int n, int N)
    {
        if (n == 0)
            return 1;
        else if (n == 1)
            return x % N;
        else
            res = (2 * x * computekey(x, n - 1, N)) - (computekey(x, n - 2, N) % N);
        return res;
    }
    private void groupBox1_Enter(object sender, EventArgs e)
    {
    }
```

```
    private void button3_Click(object sender, EventArgs e)
    {
    }
    private void button5_Click(object sender, EventArgs e)
    {
       con.Open();
      new SqlCommand("delete from nodes where node='" + label2.Text + "'", con).ExecuteNonQuery();
       new SqlCommand("delete from keydetails where node='" + label2.Text + "'",
con).ExecuteNonQuery();
       con.Close();
      this.Close();
    }
    private void button2_Click(object sender, EventArgs e)
    {
     //  byte[] buffer = Encrypt(textBox2.Text, textBox6.Text);
       con.Open();
       SqlCommand cmd1 = new SqlCommand("Delete from Message", con);
       cmd1.ExecuteNonQuery();
       con.Close();
      con.Open();
       for (int i = 0; i < Common.node_list.Count; i++)
       {
          if(!Common.node_list [i].ToString ().Equals(label2 .Text))
            new SqlCommand("insert into Message values('" + Common.node_list[i] + "','" + remVector +
"','"+qntVector+"','"+label2 .Text +"')", con).ExecuteNonQuery();
       }
       con.Close();
      // textBox4.Text = "Reminder Vector: " + remVector + " ,Quotient Vector: " + qntVector;
    }
    public  void Encrypt(string msg, int key)
    {
```

```
    remVector = "";
    qntVector = "";
    try
    {
        char[] temp = msg.ToCharArray();
        if (key > 65)
            key = key % 65;
        for (int i = 0; i < temp.Length; i++)
        {
            int ascichar = Convert.ToInt32(temp[i]);
            int rem = ascichar / key;
            int qnt = ascichar % key;
            if (rem == 0)
                rem = 1;
            remVector += rem + ",";
            qntVector += qnt + ",";
        }
        remVector = remVector.Substring(0,remVector.Length-1);
        qntVector = qntVector.Substring(0,qntVector.Length-1);
        textBox1.Text = "Quotient Vector: " + remVector + "\n" + " Reminder Vector: " + qntVector;
    }
    catch (Exception)
    {
    }
}
/// <summary>
/// Decrypt the given string using the specified key.
/// </summary>
/// <param name="strEncrypted">The string to be decrypted.</param>
/// <param name="strKey">The decryption key.</param>
```

```
/// <returns>The decrypted string.</returns>
public string Decrypt(string rvec,string qvec, int strKey)
{
   string plain = "";
   //try
   //{
      //strKey = strKey % 65;
      string[] remVec = rvec.Split(',');
      string[] qntVec = qvec.Split(',');
      if (strKey > 65)
         strKey = strKey % 65;
      for(int i=0;i<remVec.Length;i++)
      {
         int echar = (strKey * Convert.ToInt32(remVec[i])) + Convert.ToInt32(qntVec[i]);
         char pln = Convert.ToChar(echar);
         plain += pln;
      }
      return plain;
      remVector = "";
      qntVector = "";
   //}
   //catch (Exception )
   //{
   //}
}
private void button6_Click(object sender, EventArgs e)
{
   textBox1.Text = "";
   textBox3.AppendText("Me : " + textBox2.Text + "\n");
   Encrypt(textBox2.Text,Convert.ToInt32(textBox6.Text));
```

```
    textBox2.Text = "";
    byte[] buffer = Encryption(textBox2.Text, textBox6.Text);
    string b = Convert.ToBase64String(buffer);
    textBox9.Text = b;
  }
private void button7_Click(object sender, EventArgs e)
{
    textBox7.Text = "";
    msg =Decrypt(remVector, qntVector, Convert.ToInt32(textBox6.Text));
     textBox7.Text = msg;
     textBox4.Text = "";
    textBox3.AppendText("Msg from destination node  " +msg  +"\n" );
}
private void label1_Click(object sender, EventArgs e)
{
}
private void label2_Click(object sender, EventArgs e)
{
}
private void label5_Click(object sender, EventArgs e)
{
}
private void textBox2_TextChanged(object sender, EventArgs e)
{
}
private void label7_Click(object sender, EventArgs e)
{
}
private void textBox1_TextChanged(object sender, EventArgs e)
{
```

```
}
private void label3_Click(object sender, EventArgs e)
{
}
private void textBox6_TextChanged(object sender, EventArgs e)
{
}
private void label11_Click(object sender, EventArgs e)
{
}
private void label10_Click(object sender, EventArgs e)
{
}
private void label9_Click(object sender, EventArgs e)
{
}
private void label6_Click(object sender, EventArgs e)
{
}
private void textBox5_TextChanged(object sender, EventArgs e)
{
}
private void label4_Click(object sender, EventArgs e)
{
}
private void label8_Click(object sender, EventArgs e)
{
}
private void textBox4_TextChanged(object sender, EventArgs e)
{
```

```
 }
 private void label12_Click(object sender, EventArgs e)
 {
}
 private void groupBox2_Enter(object sender, EventArgs e)
 {
 }
 private void textBox7_TextChanged(object sender, EventArgs e)
 {
 }
 private void label13_Click(object sender, EventArgs e)
 {
 }
 private void textBox3_TextChanged(object sender, EventArgs e)
 {
 }
 private void button1_Click(object sender, EventArgs e)
 {
 }
private void button8_Click(object sender, EventArgs e)
 {
    MasterNode.sig = 1;
    button6.Enabled = true;
    button2.Enabled = true;
    button7.Enabled = true;
 }
 //private void Encrypt_Click(object sender, EventArgs e)
 //{
 //    byte[] buffer = Encryption(textBox1.Text, txtKey.Text);
 //    string b = Convert.ToBase64String(buffer);
```

```
    //    textBox2.Text = b;
    //}
    public static byte[] Encryption(string PlainText, string key)
    {
        TripleDES des = CreateDES(key);
        ICryptoTransform ct = des.CreateEncryptor();
        byte[] input = Encoding.Unicode.GetBytes(PlainText);
        return ct.TransformFinalBlock(input, 0, input.Length);
    }
    public static string Decryption(string CypherText, string key)
    {
        byte[] b = Convert.FromBase64String(CypherText);
        TripleDES des = CreateDES(key);
        ICryptoTransform ct = des.CreateDecryptor();
        byte[] output = ct.TransformFinalBlock(b, 0, b.Length);
        return Encoding.Unicode.GetString(output);
    }
    //private void Decrypt_Click(object sender, EventArgs e)
    //{
    //    textBox3.Text = Decryption(textBox2.Text, txtKey.Text);
    //}
    static TripleDES CreateDES(string key)
    {
        MD5 md5 = new MD5CryptoServiceProvider();
        TripleDES des = new TripleDESCryptoServiceProvider();
        des.Key = md5.ComputeHash(Encoding.Unicode.GetBytes(key));
        des.IV = new byte[des.BlockSize / 8];
        return des;
    }
} }
```

CHAPTER 6
SCREENS

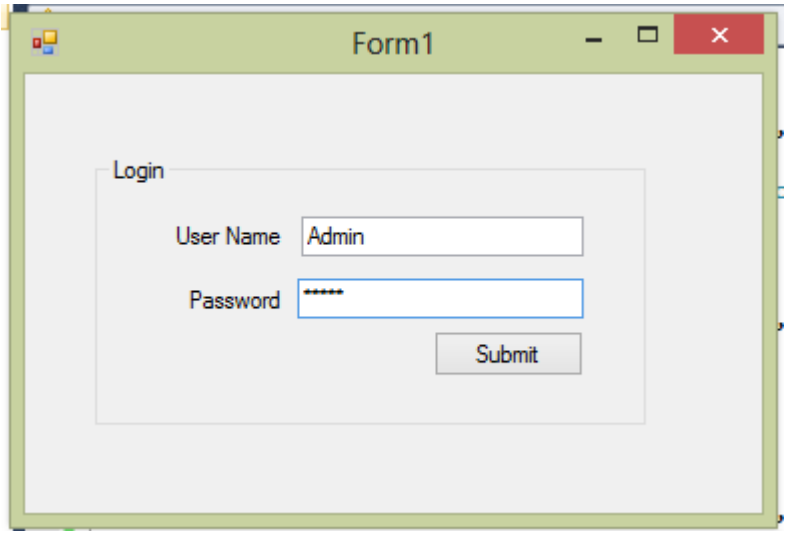


Fig 13: Login Page

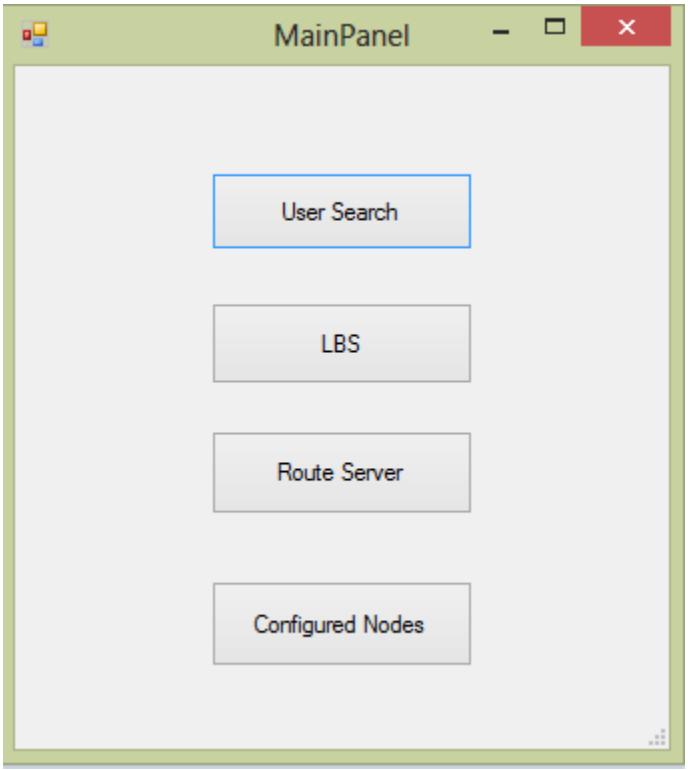


Fig 14 : Main Panel

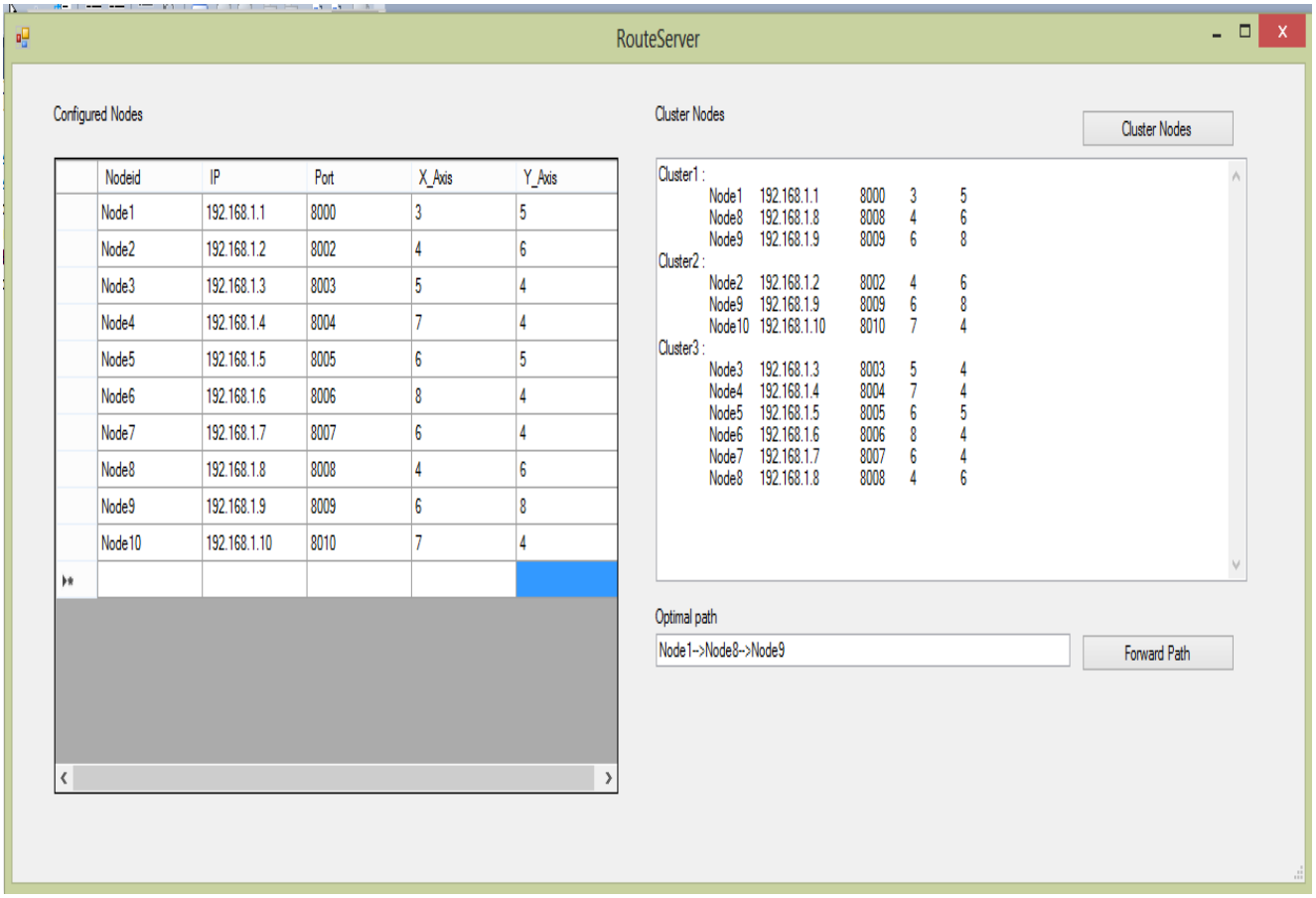


Fig 15 : Route Server

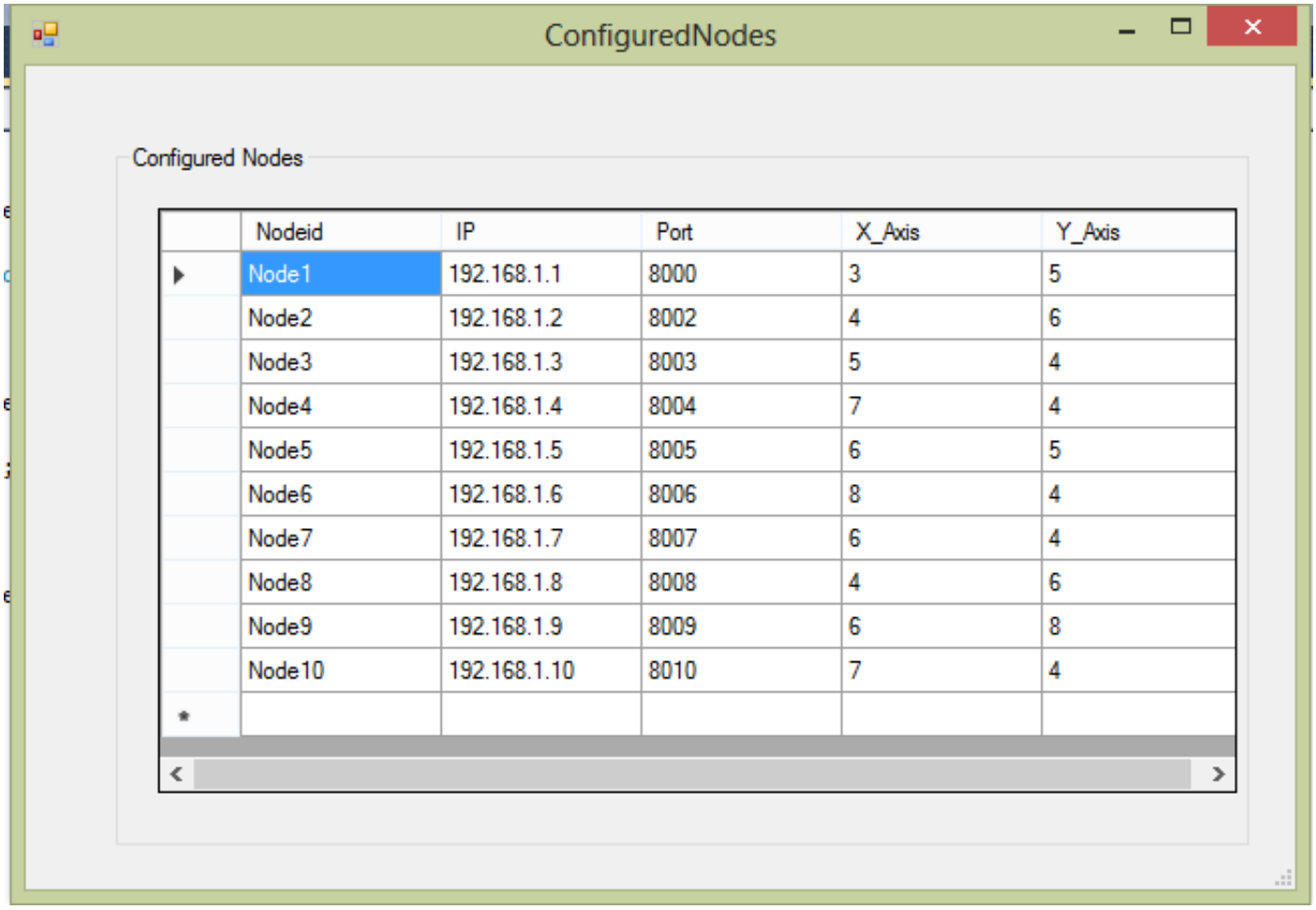


Fig 16 : Configure Nodes

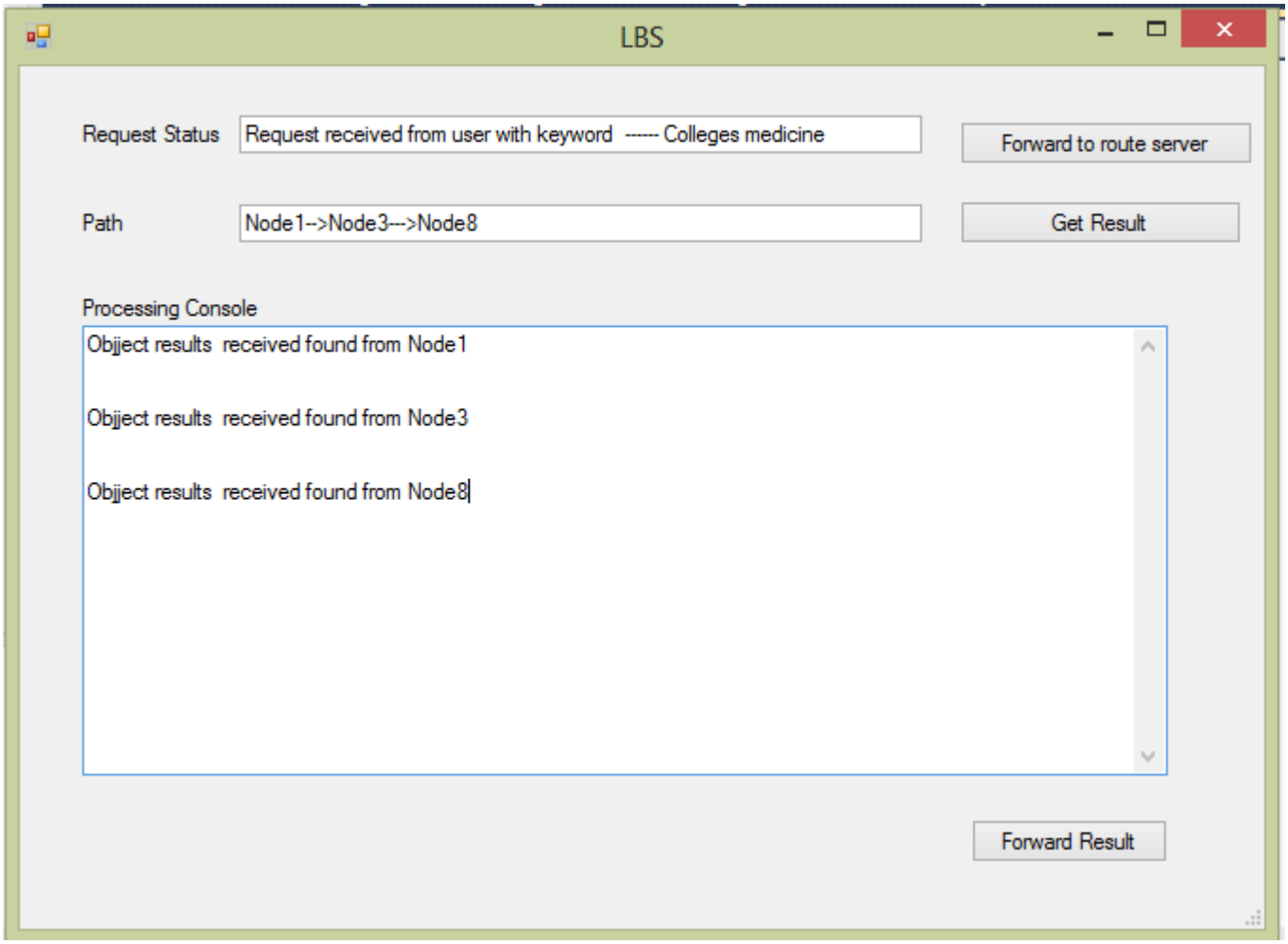


Fig 17 : Location Based Server

FUTURE SCOPE

We can improve our current cluster-based approach with dynamic cluster implementation, in our current approach we implemented node clustering based on the geo locations with static number of clusters but in real time application data should be grouped or clusters based on data dynamically and if it can support multi-dimensional data then we can improve performance.

CONCLUSION

We have been concluding our current research work with efficient route server-based implementation, user query can be forwarded to router server for cluster based path implementation and then query based results at location based system. Data can be retrieved based on the compared feature of object and attribute of feature and they are integrated and forwarded. This proposed model gives more efficient results than traditional models.

REFERENCES

[1] Statistics of Usage. (2013). [Online]. Available: http://www. quantcast.com

[2] US Maps from Government. (2013). [Online]. Available: http:// www.usgs.gov/pubprod/

[3] N. Bruno, S. Chaudhuri, and L. Gravano, "STHoles: A multidimensional workload-aware histogram," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2001, pp. 211–222.

[4] E. P. F. Chan and Y. Yang, "Shortest path tree computation in dynamic graphs," IEEE Trans. Comput., vol. 58, no. 4, pp. 541–557, Apr. 2009.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms. Cambridge, MA, USA: MIT Press, 2009.

[6] U. Demiryurek, F. B. Kashani, C. Shahabi, and A. Ranganathan, "Online computation of fastest path in time-dependent spatial networks," in Proc. 12th Int. Symp. Adv. Spatial Temporal Databases, 2011, pp. 92–111.

[7] A. Dingle and T. Partl, "Web cache coherence," Comput.Netw., vol. 28, pp. 907–920, 1996.

[8] M. Drozdowski, Scheduling for Parallel Processing, 1st ed. New York, NY, USA: Springer, 2009.

[9] H. Hu, D. L. Lee, and V. C. S. Lee, "Distance indexing on road networks," in Proc. 32nd Int. Conf. Very Large Data Bases, 2006, pp. 894–905.

[10] S. Jung and S. Pramanik, "An efficient path computation model for hierarchically structured topographical road maps," IEEE Trans. Knowl. Data Eng., vol. 14, no, 5, pp. 1029–10.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Now a days accident are occurring frequently and it is very difficult to help them in critical conditions. We need an automatic system to monitor and inform whenever accidents occur while travelling. With the problem of global population aging increasing, lack of medical facility at the location of accident, inadequate accident sense system performance and other related issues have become increasingly prominent. The population of our country has been increasing rapidly which indirectly has increased the vehicle density and lead to many road accidents.

The main causes of accidents include drunk driving, use of mobile phones, collision of vehicle with obstacles, over speeding etc. A lot of accidents are happening now-a-days because of increased vehicle density, violating rules and carelessness.

The aim of this project is to minimize the road accidents occurring due to drunk driving which cause the loss of invaluable human life and detecting accidents and thus tracking the accidental vehicle to serve emergency medical services to the victim.

In many such situations neither the family members nor the ambulance nor the police authority is informed in time. This results in delay of help reached to the person suffering from accident. So we came up with a project called “Vehicle Accident Location Identifying Alert System “. Here , whenever an accident occurs, with the help of GSM (Global System for Mobile Communications) and GPS (Global Positioning System) we are able to locate where the vehicle has been prone to accident and send an SMS regarding the vehicle state and location to the subscriber.

This Project “Vehicle Accident Location Identifying Alerting System” which is Based on IOT is designed to avoid such situations by

detecting the exact location by tracing the accident spot and send the location immediately to the family members.

1.2 Vehicle Tracking Features

It is mainly benefit for the companies which are based on transport system. Since it can show the position of all vehicles in real time, so that they can create the expected data accordingly. These tracking systems can store the whole data where the vehicle had gone, where did it stop, how much time it take at every stop and can create whole data analysis. It is also used in buses and trains, to estimate how far are they, how much time it takes for them to come to a particular stop. These systems are used to data capture, data storage, data analysis and finally data transfer.

1.3 Accident Alert System Features

This system is based on new technology, its main purpose is to detect an accident and alert to the subscriber, so the victim can find some help. It can detect accidents the intensity of the accident without any visual contact. If this system is inserted in every vehicle then it is easy to understand how many vehicles are involved in a particular accident and how intense is it. So that the help will be according to the need. The present board designed has both vehicle tracking and accident alert systems, which make it more valuable and useful. This board alerts us from theft and on accident detection also. This device detects fire accidents also by placing fire detector in one of the interrupt pins.

1.4 Usage of tracking in India

Tracking in India is mainly used by transport systems, taxi companies, traffic operators. Taxi operators use this to estimate how far the

vehicle is from a particular area and send this information to call centers and they can inform general public about the distance of the taxi location and time it takes tom come to them. Another use is for traffic police if this system is located in every vehicle they can estimate the traffic by looking on the map and if any accident is detected then they can route the traffic in to another way. This is how tracking is useful because India is one of busy traffic countries and this system can control many of the traffic problems.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing System

When an accident occurs, the information is only sent through GSM but there is no possibility to locate the spot where accident has occurred. Due to this reason many number of lives are being lost as required medical attention cannot be given to the needy person at the right time. So far, we are able communicate using GSM but unfortunately there is no innovation to evaluate on this basic problem. The only way of communication is done by contacting to the digital cellular connection. So this leads to uncertain number of accidents due to lack of exact location.

The main role of GSM in existing system is to send an message through the digital cellular connection whenever an accident had occurred manually. The existing system only applicable in the presence of the digital cellular connection and can't be operated and cannot be extended to the utmost level.

2.1.1 Drawbacks For Existing System

- Existing system can only be accessed through GSM but there is no scope to navigate the accident spot.
- The accident information can only be transmitted manually.
- Due to this many accidents are unnoticed and paying huge human loss.

2.2 Proposed System

The proposed system is designed in order to minimize the limitations of previous works and gives better flexibility in terms of its working, more secured and more comfortable.

By taking the drawbacks of the existing system we proposed a new system with an elevation. This system presents an automatic accident detector and location identifier using MEMS Accelerometer, GPS and GSM

where the accelerometer is used to detect the sudden change in the axis of the vehicle and this sudden change indicates that an accident has occurred while the buzzer gets activated with alerting sounds where the GPS tracks the location and GSM transmits the location information through an SMS. With the help of this system we can help immediately at least to the person who has been met with an accident by tracing the location of the accident spot instantly.

If the victim hadn't been harmed even after the accident the system can be set back to normal by pressing the reset switch provided in the system. Which in turn terminates from sending the alert message to the subscriber.

When the vehicle is in motion and undergoes an accident, there might be chances where the power to the system might get disconnected and, in such cases, the secondary power source is activated which has been installed and power is supplied to the system instantly. So that the system runs continuously without any interruption.

2.2.1 Advantages of Proposed System

- The sensor used is petite in size hence it can be kept in a secured place as it is not massive.
- The accident spot can be located via GPS without any delay.
- An immediate medication will be provided to the accident victims in the remote areas.
- As the project is based on IOT, it is unexceptional within itself and thus can be implemented as a safety system.

2.3 Feasibility Analysis

A feasibility analysis is a preliminary study undertaken to determine the viability of a project. The objective of such a study is to ensure a project is technically feasible and economically justifiable. It tells whether the project is worth investment. It also decides whether an order processing can be carried out by a new system more efficiently than the previous one. The five areas of feasibility study for this project are:

- Technical feasibility
- Economic feasibility
- Legal feasibility
- Operational feasibility
- Scheduling feasibility

2.3.1 Technical Feasibility

This assessment is based on an outline design of system requirements, to determine whether the company has the technical expertise to handle completion of the project. It is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. It is an evaluation of the hardware and software and how it meets the need of the proposed system. The proposed system is developed in Arduino, and some technical support has been taken from Google Maps. All required software and hardware are available with the organization hence the project is technically feasible.

2.3.2 Economic Feasibility

The economic feasibility helps the organizations to assess the viability, cost, and benefits associated with projects before financial

resources are allocated. It also serves as an independent project assessment, and enhances project credibility, as a result, it helps decisionmakers determine the positive economic benefits to the organization that the proposed system will provide, and helps quantify them. This assessment typically involves a cost/ benefits analysis of the project. The proposed solution is economically feasible. This firm has fully equipped hardware, and full-fledged software, so no need to spend money on these things.

2.3.3 Legal Feasibility

The legal feasibility investigates if the proposed system conflicts with legal requirements like data protection acts or social media laws.

This project is developed such that it doesn't violate any of the legal copyrights of the technology being implemented. Moreover, it follows all the rules required for the development process. Hence the project is legally feasible.

2.3.4 Operational Feasibility

This involves undertaking a study to analyse and determine whether the problem statement needs can be fulfilled by using the proposed solution. It also measures how well the proposed system solves problems and takes advantage of the opportunities identified during scope definition. To ensure operational feasibility, the proposed system is developed so as to include design-dependent parameters such as reliability, maintainability, supportability, usability, disposability, sustainability, affordability, and others.

2.3.5 Scheduling Feasibility

Scheduling feasibility is the most important criteria for a project success. A project will fail if not completed on time. It is estimated

here that how much time the project will take to complete, and with our technical skills we need to estimate the period to complete the project using various methods of estimation. Moreover, the priority is gives as high for this project, so as to make it feasible as per schedule.

2.4 Software Requirement Specification (SRS)

Software requirements specifications are typically developed during the first stages of "Requirements Development," which is the initial product development phase in which information is gathered about what requirements are needed–and not. This information-gathering stage can include onsite visits, questionnaires, surveys, interviews, and perhaps a return-on-investment (ROI) analysis or needs analysis of the customer or client's current business environment. The actual specification, then, is written after the requirements have been gathered and analysed. This Software Requirements Specification (SRS) provides a description of all the functions, specifications, external behaviours, design constraints, requirements (function and non-functional) and other factors necessary to provide a complete and comprehensive description of the proposed system.

Fig 2.4.1 – Factors contributing for SRS of a project

2.4.1 Functional Requirements

A functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behaviour, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define *what* a system is supposed to accomplish. Based on functional requirements, an engineer determines the behaviour (output) that a device or software is expected to exhibit in the case of a certain input. A system design is an early form of a functional requirement. A functional requirement specification may be in the form of a document which may include the following:

- Descriptions of data to be entered into the system
- Descriptions of work-flows performed by the system
- Who can enter the data into the system ?

In this project, the functional requirement specification can be given as:

- Detects vehicle accident using Accelerometer.
- Tracks the exact vehicle accident location using GPS.
- Alerts the subscriber of the system through GSM.

2.4.2 Non-Functional Requirements

Non-functional requirements describe how the system works**.** For example, attributes such as performance, security, usability, compatibility aren't a "feature" of the system, but are a required characteristic. We can't write a specific line of code to implement them, rather they are "emergent" properties that arise from the entire solution. For example, the following are the important non-functional requirements:

- Usability- it means where and how the product can be used.
- Reliability- how the users can trust the system?
- Performance- to what extent the system gives the correct results?

- Supportability- the system compatibility with different platforms.

2.4.3 Hardware and Software Requirements

As the vehicle accident location identifying alert system is integrated with Arduino, we have the following hardware and software requirements for the development and execution of the project-

Hardware requirements

- Arduino Uno R3
- GPS Modem
- LCD
- GSM Modem
- Piezo Buzzer
- MEMS Accelerometer ADXL335

Software requirements

- Arduino Compiler
- Embedded C

2.5 Software Environment

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

2.5.1 Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The

message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Verify

 Checks your code for errors compiling it.

Upload

 Compiles your code and uploads it to the configured board. See uploading below for details.

New

 Creates a new sketch

Open

 Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Save

 Saves your sketch.

Serial Monitor

 Opens the serial monitor.

2.5.2 Uploading

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The

bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

2.5.3 Libraries

Libraries provide extra functionality for use in sketches. There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

2.5.4 Third – Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory.

2.5.5 Serial Monitor

This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac or Linux the board will reset when you connect with the serial monitor.

CHAPTER 3

SYSTEM DESIGN

3.1 System Overview

There are four main parts in our project:

- Arduino
- GPS
- GSM
- MEMS Accelerometer

3.1.1 Arduino

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can communicate with software running on your computer (e.g. Flash, Processing, MaxMSP.) The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

Why Arduino ?

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handy board, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package.

Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms.
- **Cross-platform** - The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- **Simple, clear programming environment** - The Arduino programming environment is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with the look and feel of Arduino
- **Open source and extensible software**- The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
- **Open source and extensible hardware** - The Arduino is based on Atmel's ATMEGA8 and ATMEGA168/ATMEGA2560 microcontrollers. The plans for the modules are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

3.1.2 GPS

GPS (Global Positioning System) technology is used to find the location of any object or vehicle to monitor a child continuously using satellite signals. Three satellite signals are necessary to locate the receiver in 3D space and fourth satellite is used for time accuracy. GPS will give the information of parameters like longitude, latitude and attitude. With the help of these parameters one can easily locate the position of any object. In this GPS technology, the communication takes place between GPS transceiver and GPS satellite.

3.1.3 GSM

GSM (Global System for Mobile communications) is the technology that underpins most of the world's mobile phone networks. The GSM platform is a hugely successful wireless technology and an unprecedented story of global achievement and cooperation. GSM has become the world's fastest growing communications technology of all time and the leading global mobile standard, spanning 218 countries. GSM is an open, digital cellular technology used for transmitting mobile voice and data services. GSM operates in the 900MHz and 1.8GHz bands GSM supports data transfer speeds of up to 9.6 kbps, allowing the transmission of basic data services such as SMS.

3.1.4 MEMS Accelerometer

An accelerometer is a device that measures non-gravitational accelerations.

The former provides information on taps and other handset motions allowing the development of 'gesture' user interfaces while the latter provides information on the accelerometer orientation.

Accelerometers are used in tablet computers and digital cameras so that images on screens are always displayed upright. Also used in aero planes.

3.2 Unified Modeling Language (UML)

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

- UML stands for **Unified Modeling Language**.
- UML is different from the other common programming languages such as C++, Java, COBOL, etc.
- UML is a pictorial language used to make software blueprints.
- UML can be described as a general purpose visual modeling language to visualize, specify, construct, and document software system.
- Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well*. For example, the process flow in a manufacturing unit, etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. After some standardization, UML has become an OMG standard.

3.2.1 Types of UML Diagrams

The current UML standards call for 13 different types of diagrams: class, activity, object, use case, sequence, package, state, component, communication, composite structure, interaction overview, timing, and deployment. These diagrams are organized into two distinct groups: structural diagrams and behavioral or interaction diagrams.

A *diagram* is the graphical presentation of a set of elements, most often rendered as a connected graph of vertices (things) and arcs (relationships). You draw diagrams to visualize a system from different perspectives, so a diagram is a projection into a system. For all but the most trivial systems, a diagram represents an elided view of the elements that make up a system. The same element may appear in all diagrams, only a few diagrams (the most common case), or in no diagrams at all (a very rare case). In theory, a diagram may contain any combination of things and relationships. In practice, however, a small number of common combinations arise, which are consistent with the five most useful views that comprise the architecture of a software-intensive system. For this reason, the UML includes nine such diagrams:

1. Class diagram
2. Object diagram
3. Use case diagram
4. Sequence diagram
5. Collaboration diagram
6. State chart diagram
7. Activity diagram
8. Component diagram
9. Deployment diagram

- A *class diagram* shows a set of classes, interfaces, and collaborations and their relationships. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagrams address the static design view of a system. Class diagrams that include active classes address the static process view of a system.
- An *object diagram* shows a set of objects and their relationships. Object diagrams represent static snapshots of instances of the things found in class

diagrams. These diagrams address the static design view or static process view of a system as do class diagrams, but from the perspective of real or prototypical cases.

- A *use case diagram* shows a set of use cases and actors (a special kind of class) and their relationships. Use case diagrams address the static use case view of a system. These diagrams are especially important in organizing and modeling the behaviours of a system.
- Both sequence diagrams and collaboration diagrams are kinds of interaction diagrams. An shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams address the dynamic view of a system. A *sequence diagram* is an interaction diagram that emphasizes the time-ordering of messages; a *collaboration diagram* is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.
- A *state chart diagram* shows a state machine, consisting of states, transitions, events, and activities. State chart diagrams address the dynamic view of a system. They are especially important in modeling the behaviour of an interface, class, or collaboration and emphasize the event ordered behaviour of an object, which is especially useful in modeling reactive systems.
- An *activity diagram* is a special kind of a state chart diagram that shows the flow from activity to activity within a system. Activity diagrams address the dynamic view of a system. They are especially important in modeling the function of a system and emphasize the flow of control among objects.

- A *component diagram* shows the organizations and dependencies among a set of components. Component diagrams address the static implementation view of a system. They are related to class diagrams in that a component typically maps to one or more classes, interfaces, or collaborations.
- A *deployment diagram* shows the configuration of run-time processing nodes and the components that live on them. Deployment diagrams address the static deployment view of an architecture. They are related to component diagrams in that a node typically encloses one or more components.

3.2.2 UML Diagrams for Vehicle Accident Location Identifying Alert System

- Use case Diagram : Use case diagram for Vehicle Accident Location Identifying Alert System can be drawn as follows-

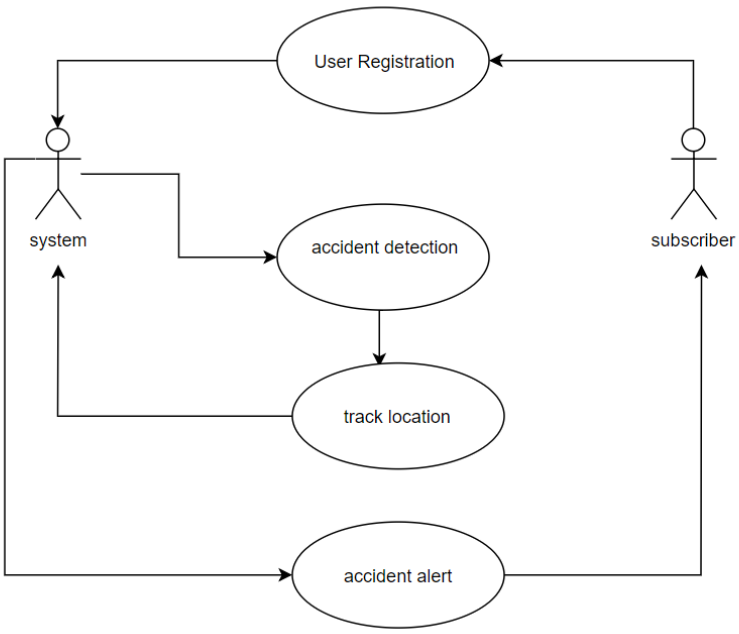


Fig 3.2.2.1 - Use Case Diagram

- Activity Diagram : Activity diagram for Vehicle Accident Location Identifying Alert System can be drawn as follows-

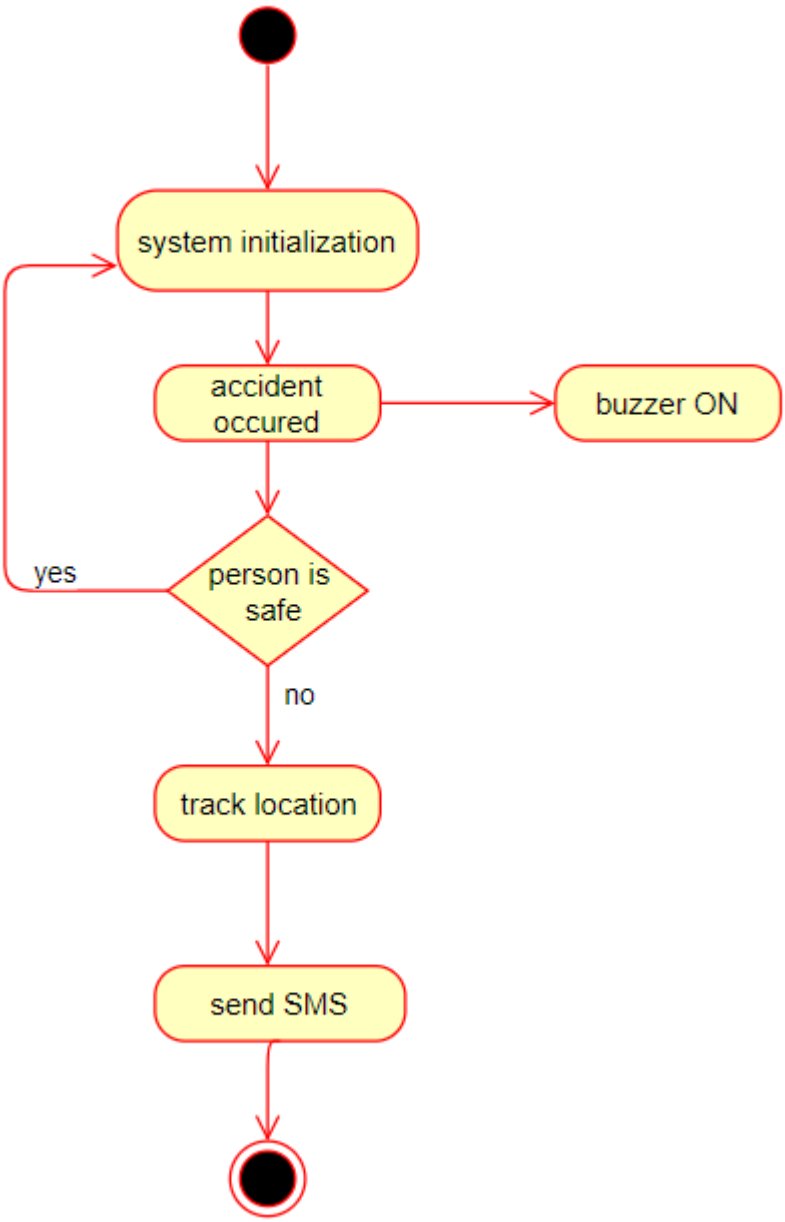


Fig 3.2.2.2 - Activity Diagram

CHAPTER 4

PROJECT IMPLEMENTATION

PROJECT DESCRIPTION:

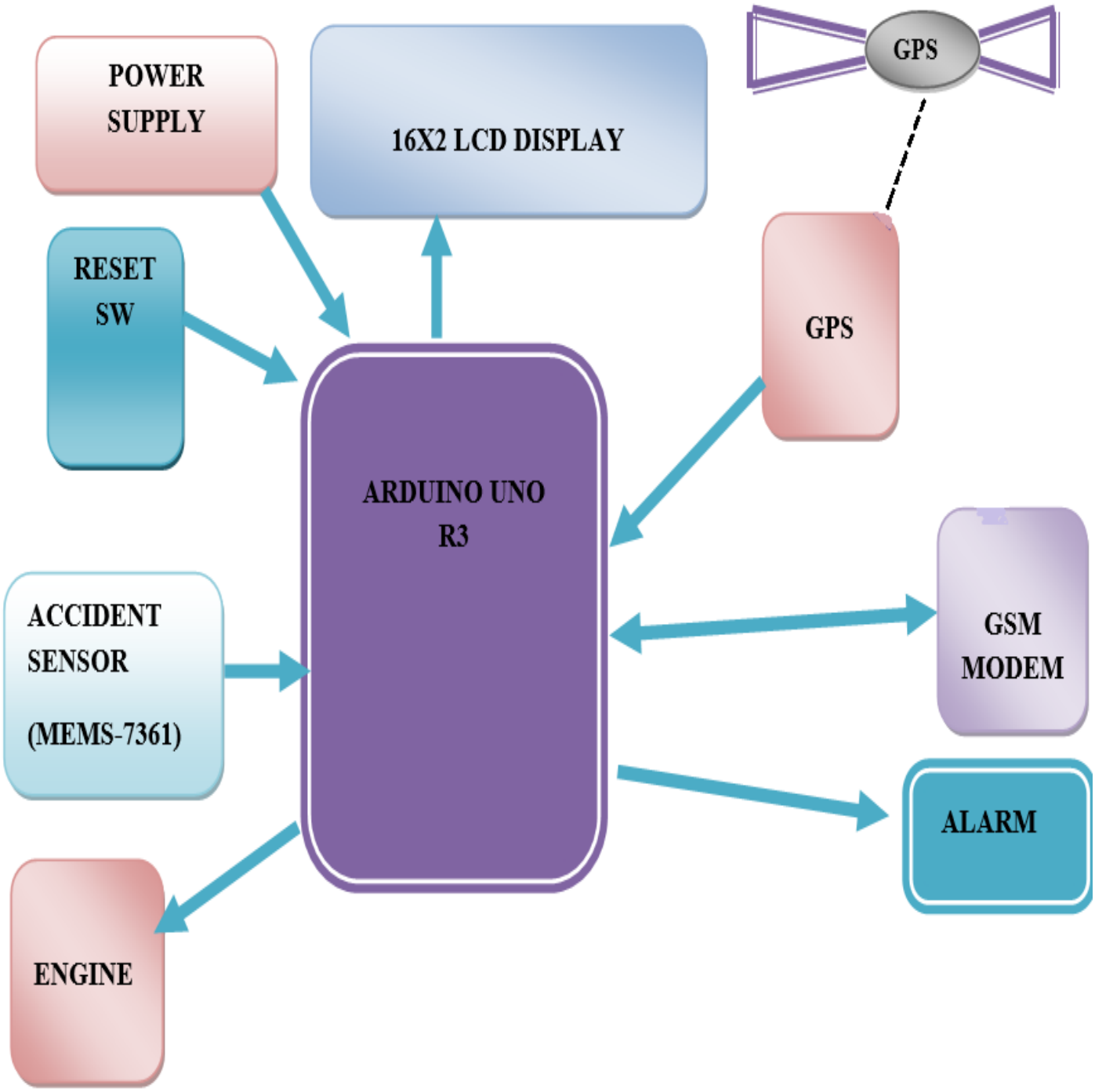


Fig 4.1 – Block Diagram of Vehicle Accident Location Identifying Alert System

4.1 Arduino

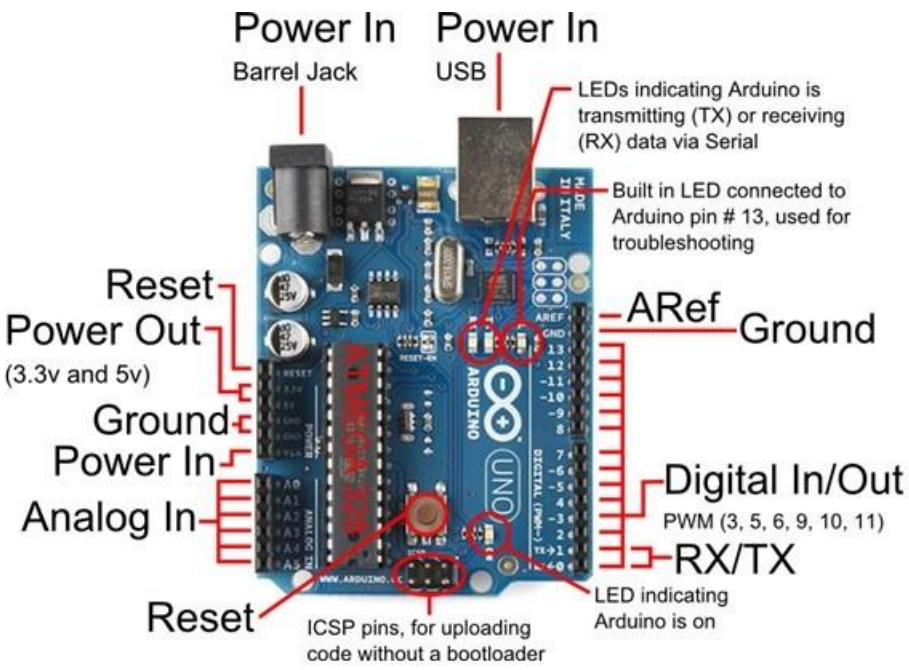


Fig 4.1.1 – Arduino Board

The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Table 4.1.1 - Technical specifications of Arduino Board

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V

Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

4.1.1 Basic Arduino Code Definitions

setup( ): A function present in every Arduino sketch. Run once before the loop( ) function. Often used to set pinmode to input or output. The setup( ) function looks like:

```
void setup( ){
    //code goes here
    }
```

loop( ): A function present in every single Arduino sketch. This code happens over and over again. The loop( ) is where (almost) everything happens. The one exception to this is setup( ) and variable declaration. ModKit uses another type of loop called "forever( )" which executes over Serial. The loop( ) function looks like:

```
void loop( ) {
    //code goes here
    }
```

input: A pin mode that intakes information.

output: A pin mode that sends information.

HIGH: Electrical signal present (5V for Uno). Also, ON or True in Boolean logic.

LOW: No electrical signal present (0V). Also OFF or False in boolean logic.

digitalRead: Get a HIGH or LOW reading from a pin already declared as an input.

digitalWrite: Assign a HIGH or LOW value to a pin already declared as an output.

analogRead: Get a value between or including 0 (LOW) and 1023 (HIGH). This allows you to get readings from analog sensors or interfaces that have more than two states.

analogWrite: Assign a value between or including 0 (LOW) and 255 (HIGH). This allows you to set output to a PWM value instead of just HIGH or LOW.

PWM: Stands for Pulse-Width Modulation, a method of emulating an analog signal through a digital pin. A value between or including 0 and 255. Used with analogWrite.

4.2 MEMS Accelerometer

An accelerometer is an electromechanical device that will measure acceleration force. It shows acceleration, only due to cause of gravity i.e. g force. It measures acceleration in g unit.

On the earth, 1g means acceleration of 9.8 m/s2 is present. On moon, it is 1/6th of earth and on mars it is 1/3rd of earth.

Accelerometer can be used for tilt-sensing applications as well as dynamic acceleration resulting from motion, shock, or vibration.

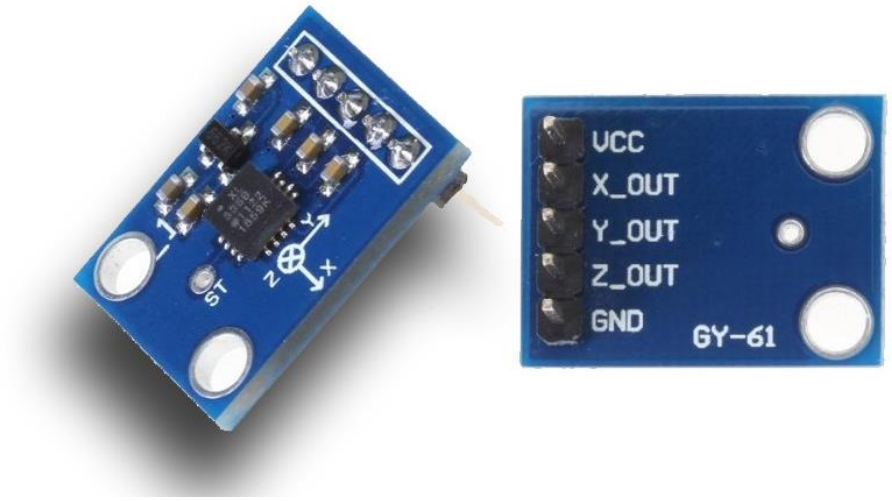


Fig 4.2.1 – MEMS Accelerometer (ADXL335)

4.2.1 ADXL335 Module

- The ADXL335 gives complete 3-axis acceleration measurement.
- This module measures acceleration within range ±3 g in the x, y and z axis.
- The output signals of this module are analog voltages that are proportional to the acceleration.
- It contains a polysilicon surface-micro machined sensor and signal conditioning circuitry.
- Angle of inclination means by how much angle the device is tilted from its plane of surface.

Table 4.2.1.1 – Pin Description of ADXl335 Accelerometer

Pin	Description
VCC	Power supply pin i.e. connect 5V here
X_OUT	X axis analog output
Y_OUT	Y axis analog output
Z_OUT	Z axis analog output
GND	Ground pin i.e. connect ground here

ADXL335 accelerometer provides analog voltage at the output X, Y, Z pins; which is proportional to the acceleration in respective directions i.e. X, Y, Z.

4.3 GSM

A GSM modem is a specialized type of modem which accepts a SIM card, and operates over a subscription to a mobile operator, just like a mobile phone. From the mobile operator perspective, a GSM modem looks

just like a mobile phone. When a GSM modem is connected to a computer, this allows the computer to use the GSM modem to communicate over the mobile network. While these GSM modems are most frequently used to provide mobile internet connectivity, many of them can also be used for sending and receiving SMS and MMS messages. A GSM modem can be a dedicated modem device with a serial, USB or Bluetooth connection, or it can be a mobile phone that provides GSM modem capabilities.

A GSM modem exposes an interface that allows applications such as Now SMS to send and receive messages over the modem interface. The mobile operator charges for this message sending and receiving as if it was performed directly on a mobile phone. To perform these tasks, a GSM modem must support an "extended AT command set" for sending/receiving SMS messages.

GSM modems can be a quick and efficient way to get started with SMS, because a special subscription to an SMS service provider is not required. In most parts of the world, GSM modems are a cost effective solution for receiving SMS messages, because the sender is paying for the message delivery.

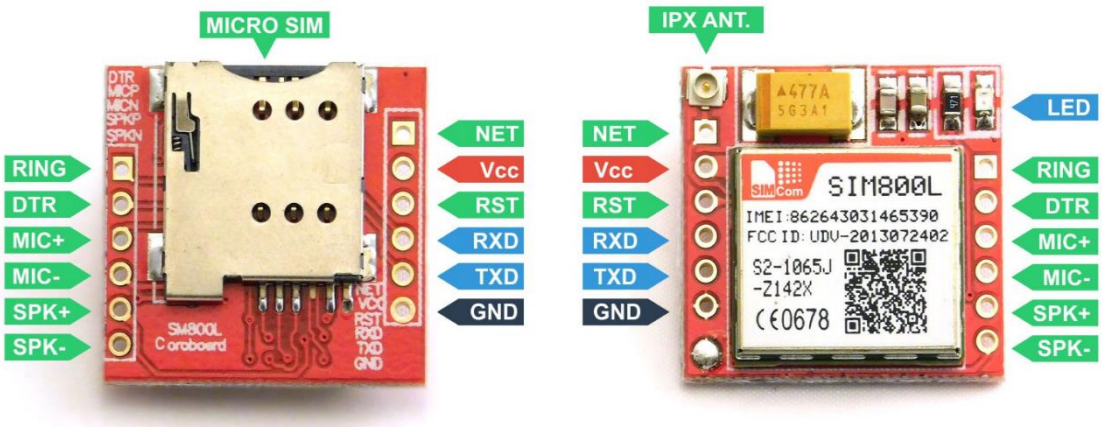


Fig 4.3.1 – GSM Modem (SIM800L)

4.3.1 SIM800L

SIM800L is a miniature cellular module which allows for GPRS transmission, sending and receiving SMS and making and receiving voice calls. Low cost and small footprint and quad band frequency support make this module perfect solution for any project that require long range connectivity. After connecting power module boots up, searches for cellular network and login automatically. On board LED displays connection state (no network coverage - fast blinking, logged in - slow blinking).

Table 4.3.1.1 – Pin Description of SIM800L GSM Modem

Pin	Description
RING	LOW state while receiving call
DTR	sleep mode. Default in HIGH state After setting it LOW will wake up.
MICP, MICN	microphone (P + / N -)
SPKP, SPKN	speaker (P + / N -)
NET	antenna
VCC	supply voltage
RESET	reset
RXD	serial communication
TXD	serial communication
GND	ground

4.3.2 Architecture of GSM Network

A GSM network is composed of several functional entities, whose functions and interfaces are specified. Figure 1 shows the layout of a generic GSM network. The GSM network can be divided into three broad parts. The Mobile Station is carried by the subscriber. The Base Station Subsystem

controls the radio link with the Mobile Station. The Network Subsystem, the main part of which is the Mobile services Switching Center (MSC), performs the switching of calls between the mobile users, and between mobile and fixed network users. The MSC also handles the mobility management operations. Not shown are the Operations

i. Mobile Station

The mobile station (MS) consists of the mobile equipment (the terminal) and a smart card called the Subscriber Identity Module (SIM). The SIM provides personal mobility, so that the user can have access to subscribed services irrespective of a specific terminal. By inserting the SIM card into another GSM terminal, the user is able to receive calls at that terminal, make calls from that terminal, and receive other subscribed services.

ii. Base Station Subsystem

The Base Station Subsystem is composed of two parts, the Base Transceiver Station (BTS) and the Base Station Controller (BSC). These communicate across the standardized Abis interface, allowing (as in the rest of the system) operation between components made by different suppliers. The Base Transceiver Station houses the radio transceivers that define a cell and handles the radio-link protocols with the Mobile Station.

***iii.* Network Subsystem**

The central component of the Network Subsystem is the Mobile services Switching Center (MSC). It acts like a normal switching node of the PSTN or ISDN, and additionally provides all the functionality needed to handle a mobile subscriber, such as registration, authentication, location updating, handovers, and call routing to a roaming subscriber.

4.4 GPS

The Global Positioning System (GPS) is the most significant recent advance in navigation and positioning technology. In the past, the stars were used for navigation. Today's world requires greater accuracy. The new constellation of with radius equal to the distance to the satellite. If two satellites are used, then the receiver must be on the surface of both spheres, which is the intersection of the two spheres or the perimeter of a circle. If a third satellite is used, then the location of the user is narrowed down to the two points where the three spheres intersect. Three measurements are enough for land receivers since the lower of the two points would be selected. But when in the air or space, four satellites are needed; the intersection of all four spheres will be the receiver's location. When more than four satellites are used, greater accuracy can be achieved.

Global Positioning System satellites transmit signals to equipment on the ground, GPS receivers passively receive satellite signals; they do not transmit. GPS receivers require unobstructed views of the sky, so they are used only outdoors and they often do not perform well within forested areas or near tall buildings. GPS operations depend on a very accurate time reference.

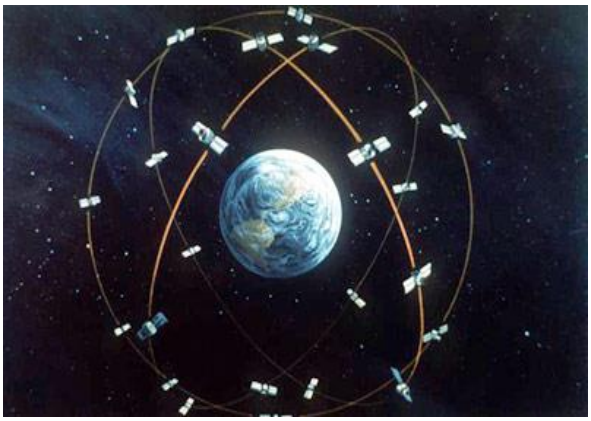


Fig 4.4.1 – Satellites Orbiting the Earth

4.4.1 Satellites in Space

The complete GPS space system includes 24 satellites, 11,000 nautical miles above the Earth, which take 12 hours each to go around the Earth once (one orbit). Satellites are equipped with very precise clocks that keep accurate time to within three nanoseconds - that's 0.000000003, or three billionths, of a second. This precision timing is important because the receiver must determine exactly how long it takes for signals to travel from each GPS satellite. The receiver uses this information to calculate its position.

4.4.2 NEO-6M

The NEO-6M GPS module is shown in the figure below. It comes with an external antenna, and doesn't come with header pins.

- This module has an external antenna and built-in EEPROM.
- Interface: RS232 TTL
- Power supply: 3V to 5V
- Default baud rate: 9600 bps
- Works with standard NMEA sentences

Fig 4.4.2.1 – GPS Modem (NEO-6M)

Table 4.4.2.1 – Pin Description of NEO-6M GPS Modem

Pin	Description
VCC	5V
RX	TX pin defined in the software serial
TX	RX pin defined in the software serial
GND	GND

4.5 Liquid Crystal Display (LCD)

Liquid crystal display is a type of display used in digital watches and many portable computers.

Fig 4.5.1 – 2 x 16 LCD Display

LCD displays utilize two sheets of polarizing material with a liquid crystal solution between them. An electric current passed through the liquid causes the crystals to align so that light cannot pass through them. Each crystal, therefore, is like a shutter, either allowing light to pass through or blocking the light. The liquid crystals can be manipulated through an applied electric voltage so that light is allowed to pass or is blocked.

A back light provides LCD monitor's brightness. Other advances have allowed LCD's to greatly reduce liquid crystal cell response times.

Response time is basically the amount of time it takes for a pixel to "change colours". In reality response time is the amount of time it takes a liquid crystal cell to go from being active to inactive. Here the LCD is used at both the Transmitter as well as the receiver side.

The input which we give to the microcontroller is displayed on the LCD of the transmitter side and the message sent is received at the receiver side which displays at the receiver end of the LCD and the corresponding operation is performed.

They make complicated equipment easier to operate. LCDs come in many shapes and sizes but the most common is the 16-character x 4-line display with no backlight.

It requires only 11 connections – eight bits for data (which can be reduced to four if necessary) and three control lines (we have only used two here). It runs off a 5V DC supply and only needs about 1mA of current.

The display contrast can be varied by changing the voltage into pin 3 of the display.

Table 4.5.1 – Pin Description of 2 x 16 LCD Display

Pin No.	**Pin Names**	**Description**
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.

4	Register Select	Connected to Microcontroller to shit between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

4.6 Light Emitting Diode (LED)

A light-emitting diode (LED) is a semiconductor diode that emits incoherent narrow spectrum light when electrically biased in the forward

direction of the pn-junction, as in the common LED circuit. This effect is a form of electroluminescence

While sending a message in the form of bits such as 1, the data is sent to the receiver side correspondingly the LED glows representing the data is being received simultaneously when we send 8 as a data the LED gets off.

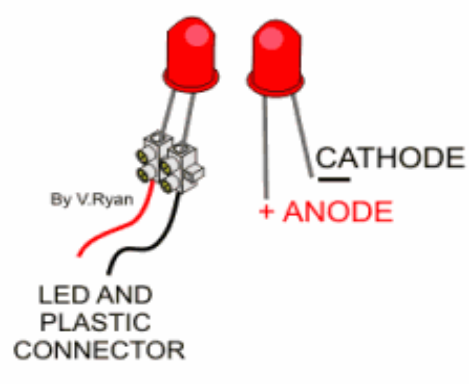


Fig 4.6.1 – LED

4.7 DC Motor

Fig 4.7.1 – DC Motor

DC motors are configured in many types and sizes, including brush less, servo, and gear motor types. A motor consists of a rotor and a permanent magnetic field stator. The magnetic field is maintained using

either permanent magnets or electromagnetic windings. DC motors are most commonly used in variable speed and torque.

Motors are the devices that provide the actual speed and torque in a drive system. This family includes AC motor types (single and multiphase motors, universal, servo motors, induction, synchronous, and gear motor) and DC motors (brush less, servo motor, and gear motor) as well as linear, stepper and air motors, and motor contactors and starters.

In any electric motor, operation is based on simple electromagnetism. A current-carrying conductor generates a magnetic field; when this is then placed in an external magnetic field, it will experience a force proportional to the current in the conductor, and to the strength of the external magnetic field. As you are well aware of from playing with magnets as a kid, opposite (North and South) polarities attract, while like polarities (North and North, South and South) repel. The internal configuration of a DC motor is designed to harness the magnetic interaction between a current-carrying conductor and an external magnetic field to generate rotational motion.

4.8 Piezo Buzzer

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Fig 4.8.1 – Piezo Buzzer

The **piezo buzzer** produces sound based on reverse of the piezoelectric effect. The generation of pressure variation or strain by the application of electric potential across a piezoelectric material is the underlying principle. These buzzers can be used alert a user of an event corresponding to a switching action, counter signal or sensor input. They are also used in alarm circuits.

The buzzer produces a same noisy sound irrespective of the voltage variation applied to it. It consists of piezo crystals between two conductors. When a potential is applied across these crystals, they push on one conductor and pull on the other. This, push and pull action, results in a sound wave. Most buzzers produce sound in the range of 2 to 4 kHz.

4.9 Power Supply

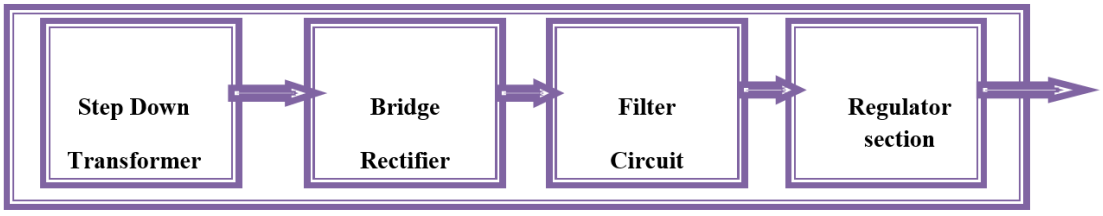


Fig 4.9.1 – Block Diagram of Power Supply

In our project we used a 9 volts transformer for continuous power supply Otherwise If we use a battery sometimes the total currents will loss so that's why we are using A.C Transformer. A.C transformer is giving the input to Bridge Rectifier. Bridge Rectifier converts A.C to D.C. After that we are using one filter capacitor 1000uf/25v electrolytic capacitor. We are connecting this capacitor in parallel section. The main purpose of this capacitor is if there are any alternate peaks we need to reduce that peaks. Nothing but a filtering that repull's. After that we are using LM7805 Regulator Most digital logic circuits and processors need a 5-volt power supply. To use these parts, we need to build a regulated 5-volt source. We make a 5-volt power supply, The LM7805 is simple to use. First connect the positive lead of our unregulated DC power supply Input pin, connect the negative lead to the Common pin and then when we turn on the power, we get a 5-volt supply from the Output pin. Here we are using one red colour led to indicate the power.

Steps to be followed during execution

- Step – 1 : Supply the power to Arduino and GSM Module using AC power supply or using 9 volt batteries which are the secondary source of power using the provided power cables.
- Step – 2 : Once the system gets the required power the system gets automatically initialized and checks the connectivity and working condition of the connected components is displayed on the 16x2 LCD.
- Step – 3 : The Arduino continuously reads values that are extracted from the GPS modem and checks if any changes have been occurred in the axis.
- Step – 4 : Now whenever there is an accident, the vehicle gets tilt and accelerometer changes its axis values.
- Step – 5 : Once it detects any sudden change in axis of the vehicle, the Arduino reads coordinates by extracting from GPS module data and sends SMS to the subscriber with the location coordinates of accident place.
- Step – 6 : The message contains a Google Map link to the accident location, so that location can be easily tracked. And upon clicking on the link the subscriber can see the exact location of the vehicle.
- Step – 7 : The RESET button provided in the Arduino stops from sending an unwanted alert message to the subscriber and can be clicked when the user doesn't want the SMS to be delivered

CHAPTER 5

TESTING

5.1 Introduction

Testing is the process of evaluating a system with the intent to find that whether it satisfies the specified requirements or not. In simple words testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirement

5.2 Testing Types

a) ***Unit Testing***

Unit testing is the testing of individual unit or group of related units. It is often done by programmer to test that the unit has implemented is producing expected output for given input. A unit is the smallest testable part of software. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

Limitations of Unit Testing

- Testing cannot catch each and every bug in an application.
- It is impossible to evaluate every execution path in every software application. The same is the case with unit testing.

b) ***Integration Testing***

The testing of combined parts of an application to determine if they function correctly together is Integration testing. There are two methods of doing Integration Testing :

- Bottom-up Integration testing – it begins with unit testing, followed by tests of progressively high-level combinations of units called modules.
- Top Down Integration testing- the highest-level modules are tested first and progressively lower-level modules are tested after that. In a

comprehensive software development environment, bottom-up testing is usually done first followed by top-down testing.

c) ***Functional Testing***

This is a type of black box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional Testing of the software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. There are five steps that are involved when testing an application for functionality.

- Step I - The determination of the functionality that the intended application is meant to perform.
- Step II - The creation of test data based on the specifications of the application.
- Step III - The output based on the test data and the specifications of the application.
- Step IV - The writing of Test Scenarios and the execution of test cases.
- Steps V - The comparison of actual and expected results based on the executed test case

d) System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., operating system) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

e) Black Box Testing

Black box testing is a software testing technique in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications. This method attempts to find errors in the following categories:

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external database access.
- Behaviour or performance errors.
- Initialization and termination errors.

f) *White Box Testing*

It is also known as Clear Box testing , Open Box Testing. It is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the approximate outputs. Programming know-how and the implementation knowledge is essential. White box testing is a testing beyond the user interface and into the ninty-gritty of a system.

CHAPTER 6

SAMPLE CODE

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 10, 11, 12, 13);
#include <SoftwareSerial.h>
SoftwareSerial gps(4, 5); // RX, TX
int i=0;
int  gps_status=0;
float latitude=0;
float logitude=0;
String gpsString="";
char *test="$GPRMC";
////////////////////////////////////////////////////////////////////////////////////////////////////////
int fan = 7;                          //Connect LED 1 To Pin #A4          //motor
int BUZZ = 6;                         //Connect LED 2 To Pin #7           //buzzer
////////////////////////////////////////////////////////////////////////////////////////////////////////
unsigned int MEMSX;
unsigned int MEMSY;
////////////////////////////////////////////////////////////////////////////////////////////////////////
void initModule(String cmd, char *res, int t)
{
while(1)
{
Serial.println(cmd);
delay(100);
while(Serial.available()>0)
{
if(Serial.find(res))
{
Serial.println(res);
delay(t);
```

```
return;
}
else
{
Serial.println("Error");
}
}
delay(t);
}
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void setup()
{
lcd.begin(16, 2);
Serial.begin(9600);
gps.begin(9600);

pinMode(fan, OUTPUT);
pinMode(BUZZ, OUTPUT);

digitalWrite(fan,LOW);
digitalWrite(BUZZ,LOW);
lcd.begin(16,2);
lcd.setCursor(0,0);

lcd.print("Vehicle Accident");
lcd.setCursor(0,1);
lcd.print("Detection Using");
delay (5000);
```

```
lcd.clear();
lcd.setCursor(0,0);
lcd.print("GPS MODEM &");
lcd.setCursor(0,1);
lcd.print("GSM MODEM ");
delay (5000);
lcd.clear();
lcd.print("Waiting For GPS");
lcd.setCursor(0,1);
lcd.print(" Signal    ");
get_gps();
show_coordinate();
delay(3000);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("GPS is OK");
delay(1000);
lcd.clear();
lcd.print("Initializing");
lcd.setCursor(0,1);
lcd.print("GSM MODEM");
delay(1000);
initModule("AT","OK",1000);
initModule("ATE1","OK",1000);
initModule("AT+CPIN?","READY",1000);
initModule("AT+CMGF=1","OK",1000);
initModule("AT+CNMI=2,2,0,0,0","OK",1000);
lcd.clear();
lcd.print("Initialized");
```

```
lcd.setCursor(0,1);
lcd.print("Successfully");
delay(2000);
lcd.clear();
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void loop()
{st:
lcd.clear() ;
MEMSX = analogRead(0);
MEMSX=MEMSX/2;
lcd.setCursor(0,0);
lcd.print("X:");
lcd.setCursor(3,0);
lcd.print(MEMSX);
delay(1000);
MEMSY = analogRead(1);
MEMSY=MEMSY/2;
lcd.setCursor(0,1);
lcd.print("Y: ");
lcd.setCursor(3,1);
lcd.print(MEMSY);
delay(1000);
////////////////////////////////////////////////////////////////////////////////////////////////////////////
if(((MEMSX >= 190) & (MEMSX <= 200)) &  ( (MEMSY >= 190) &
(MEMSY <= 200 )) )
{
lcd.setCursor(8,0);
lcd.print("NORMAL  ");
```

```
delay(500);
digitalWrite(fan,HIGH);
digitalWrite(BUZZ,LOW);
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
if(((MEMSX >= 130) & (MEMSX <= 180)) &  ( (MEMSY >= 190) &
(MEMSY <= 200 )) )
{
lcd.setCursor(8,0);
lcd.print("RIGHT   ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
digitalWrite(fan,LOW);
digitalWrite(BUZZ,HIGH);
delay(2000);
lcd.clear();
lcd.setCursor(6,0);
lcd.print("PRESS RST");
lcd.setCursor(6,1);
lcd.print("SWITCH");
delay(5000);
delay(5000);
lcd.clear();
lcd.setCursor(8,0);
lcd.print("RIGHT   ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
delay(2000);
lcd.clear();
```

```
lcd.print("Sending SMS ");
delay(2000);
Send1();
delay(2000);
goto st;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
if(((MEMSX >= 210) & (MEMSX <= 230)) &  ( (MEMSY >= 190) &
(MEMSY <= 200 )) )
{
lcd.setCursor(8,0);
lcd.print("LEFT    ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
digitalWrite(fan,LOW);
 digitalWrite(BUZZ,HIGH);
delay(2000);
lcd.clear();
lcd.setCursor(6,0);
lcd.print("PRESS RST");
lcd.setCursor(6,1);
lcd.print("SWITCH");
delay(5000);
delay(5000);
lcd.clear();
lcd.setCursor(8,0);
lcd.print("LEFT    ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
```

```
delay(2000);
lcd.clear();
lcd.print("Sending SMS ");
delay(2000);
Send2();
delay(2000);
goto st;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
if(((MEMSX >= 190) & (MEMSX <= 200)) &  ( (MEMSY >= 210) &
(MEMSY <= 220 )) )
{
lcd.setCursor(8,0);
lcd.print("FRONT  ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
digitalWrite(fan,LOW);
 digitalWrite(BUZZ,HIGH);
delay(2000);
lcd.clear();
lcd.setCursor(6,0);
lcd.print("PRESS RST");
lcd.setCursor(6,1);
lcd.print("SWITCH");
delay(5000);
delay(5000);
lcd.clear();
lcd.setCursor(8,0);
lcd.print("FRONT    ");
```

```
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
delay(2000);
lcd.clear();
lcd.print("Sending SMS ");
delay(2000);
Send3();
delay(2000);
goto st;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
if(((MEMSX >= 190) & (MEMSX <= 200)) &  ( (MEMSY >= 130) &
(MEMSY <= 180 )) )
{
lcd.setCursor(8,0);
lcd.print("BACK  ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
digitalWrite(fan,LOW);
 digitalWrite(BUZZ,HIGH);
delay(2000);
lcd.clear();
lcd.setCursor(6,0);
lcd.print("PRESS RST");
lcd.setCursor(6,1);
lcd.print("SWITCH");
delay(5000);
delay(5000);
lcd.clear();
```

```
lcd.setCursor(8,0);
lcd.print("BACK    ");
lcd.setCursor(8,1);
lcd.print("ACCIDENT ");
delay(2000);
lcd.clear();
lcd.print("Sending SMS ");
delay(2000);
Send4();
delay(2000);
goto st;
}
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void gpsEvent()
{
gpsString="";
while(1)
{
while (gps.available()>0)  //Serial incoming data from GPS
{
char inChar = (char)gps.read();
gpsString+= inChar;  //store incoming data from GPS to temparary string str[]
i++;
if (i < 7)
{
if(gpsString[i-1] != test[i-1])  //check for right string
{
i=0;
```

```
gpsString="";
}
}
if(inChar=='\r')
{
if(i>65)
{
gps_status=1;
break;
}
else
{
i=0;
}
}
}
if(gps_status)
break;
}
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void get_gps()
{
lcd.clear();
lcd.print("Getting GPS Data");
lcd.setCursor(0,1);
lcd.print("Please Wait.....");
gps_status=0;
int x=0;
```

```
while(gps_status==0)
{
gpsEvent();
int str_lenth=i;
coordinate2dec();
i=0;x=0;
str_lenth=0;
}
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void show_coordinate()
{
lcd.clear();
lcd.print("Lat:");
lcd.print(latitude);
lcd.setCursor(0,1);
lcd.print("Log:");
lcd.print(logitude);
delay(2000);
lcd.clear();
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
$GPRMC,053508.00,A,1725.64574,N,07835.11697,E,0.041,,121217,,,D*79
void coordinate2dec()
{
String lat_degree="";
for(i=19;i<=20;i++)
lat_degree+=gpsString[i];
String lat_minut="";
```

```
for(i=21;i<=27;i++)
lat_minut+=gpsString[i];
String log_degree="";
for(i=32;i<=34;i++)
log_degree+=gpsString[i];
String log_minut="";
for(i=35;i<=41;i++)
log_minut+=gpsString[i];
float minut= lat_minut.toFloat();
minut=minut/60;
float degree=lat_degree.toFloat();
latitude=degree+minut;
minut= log_minut.toFloat();
minut=minut/60;
degree=log_degree.toFloat();
logitude=degree+minut;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void init_sms1()
{
Serial.println("AT+CMGF=1");delay(400);
Serial.println("AT+CMGS=\"9985710567\"");   // use your 10 digit cell no.
here
delay(400);

}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void init_sms2()
{
```

```
Serial.println("AT+CMGF=1");delay(400);
Serial.println("AT+CMGS=\"9885564114\"");   // use your 10 digit cell no.
here
delay(400);
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void send_data(String message)
{ Serial.print(message);delay(200);}
//////////////////////////////////////////////////////////////////////////
void send_sms()
{ Serial.write(26);}
//////////////////////////////////////////////////////////////////////////
void lcd_status()
{lcd.clear();lcd.print("Message Sent");  delay(3000);  return;}
//////////////////////////////////////////////////////////////////////////
void send_sms_msg()
{
Serial.println("Vehicle Location Place");delay(500);
Serial.print("Latitude:");Serial.print(latitude,6);send_data(",N\n");
Serial.print("Longitude:");Serial.print(latitude,6);send_data(",E\n");
Serial.println("Plz Rescue ");
Serial.print("https://www.google.com/maps/place/");
Serial.print(latitude,6);Serial.print(",");Serial.print(logitude,6);Serial.write(26
);delay(2000);
send_sms();
delay(2000);
lcd_status();
delay(500);
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void Send1()
{
init_sms1();Serial.println("RIGHT
ACCIDENT");delay(500);send_sms_msg();
init_sms2();Serial.println("RIGHT
ACCIDENT");delay(500);send_sms_msg();
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void Send2()
{
init_sms1();Serial.println("LEFT ACCIDENT");delay(500);send_sms_msg();
init_sms2();Serial.println("LEFT ACCIDENT");delay(500);send_sms_msg();
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void Send3()
{
init_sms1();Serial.println("FRONT
ACCIDENT");delay(500);send_sms_msg();
init_sms2();Serial.println("FRONT
ACCIDENT");delay(500);send_sms_msg();
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void Send4()
{
init_sms1();Serial.println("BACK
ACCIDENT");delay(500);send_sms_msg();
init_sms2();Serial.println("BACK
ACCIDENT");delay(500);send_sms_msg();    }
```

CHAPTER 7

SCREENS

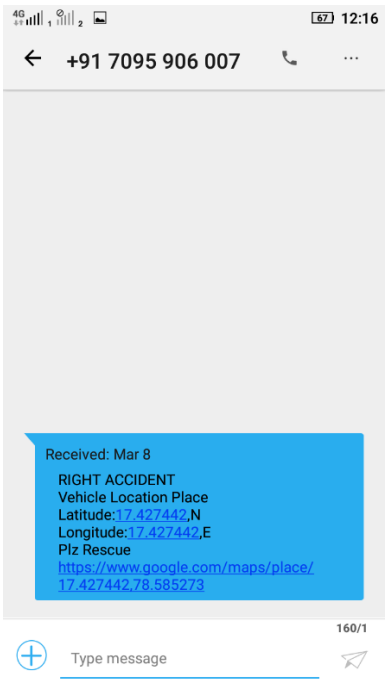


Fig 7.1 - Alert SMS received by subscriber

Fig 7.2 - Pinned location of accident displayed in google maps

FUTURE SCOPE

In some places where there is no provision of GSM network it is difficult for communication. The research work is going on for tracking the position of the vehicle even in dark clumsy areas where there is no network for receiving the signals.

CONCLUSION

The advent of this project platform is based mainly on the GSM and GPS services along with MEMS Accelerometer. This system can overcome the problems that lack in existing system by detecting the exact location when an accident occurs and alerting the subscriber automatically. Main motto of the Vehicle Accident Alert System is to recognize the location of the accident spot without any delay.

REFERENCES

1. P.Jyothi, G. Harish. *Design and Implementation of Real time vehicle Monitering, Tracking and Controlling System*, IEEE
2. Seok Lee, G. Tewolde, Jaerock Kwon, "*Design and Implementation of vehicle tracking system using GPS, GSM technology and smart phone application*, IoT 2014 IEEE
3. *https://www.youtube.com/watch?v=sbLSdP5GxPc*
4. *https://circuitdigest.com/microcontroller-projects/arduino-based-accident-alert-system-using-gps-gsm-accelerometer*
5. *Building Arduino Projects for the Internet of Things: Experiments with Real-World Applications* 1st ed. Edition by Adeel Javed (Author)
6. *Cognitive Sensors and IoT: Architecture, Deployment, and Data Delivery* by Fadi Al-Turjman (Author)
7. Introduction to Internet of Things Lecture Course from NPTEL

International Journal of Technical Innovation in Modern Engineering & Science (IJTIMES)
Impact Factor: 5.22 (SJIF-2017), e-ISSN: 2455-2585
Volume 5, Issue 03, March-2019

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Ch. Venkateswara Rao[1], K. Pavan Kumar[2], T. Charishmaa[3], P. Divya[4], V. Sangam Naidu[5]

Assistant Professor, Dept. of Computer Science and Engineering, Sanketika Vidya Parishad Engineering College, Visakhapatnam, India [1]

B.Tech(IV/IV) Students, Dept. of Computer Science and Engineering, Sanketika Vidya Parishad Engineering College, Visakhapatnam, India[2,3,4,5]

ABSTRACT :
This system designs the prototype of accident alert system which is mainly based on Arduino, GPS, GSM and MEMS Accelerometer.When an accident is occurred it is identified immediately and its location is transmitted to one or more subscribers of the system via GSM service.Then the location information can be displayed on the google map by the subscriber, in order to reach the scene of the accident faster and provide necessary medical services required.The communication between the web server and the hardware device is established via GSM shield and the location is traced via GPS shield.

Keywords : GSM , GPS , MEMS Accelerometer, Arduino

I. INTRODUCTION :

Now a-days there are a lot of accidents occurring on roads due to increased traffic and rash driving of the drivers. It is gloom to say that we are not even knowing where the accidents are occurring.

In many such situations neither the family members nor the ambulance nor the police authority is informed in time.

This results in delay of help reached to the person suffering from accident. So we came up with a project called "Vehicle Accident Location Identifying Alert System ".

Here, whenever an accident occurs, with the help of GSM (Global System For Mobile Communications) and GPS (Global Positioning System) we are able to locate where the vehicle has been prone to accident and send an SMS regarding the vehicle state and location to the subscriber.

This Project "Vehicle Accident Location Identifying Alerting System" which is Based on IOT is designed to avoid such situations by detecting the exact location by tracing the accident spot and send the location immediately to the family members.

II. SCOPE AND OBJECTIVE :

In present scenario, we see many cases where people die on roads due to unnoticed road accidents and due to help not offered in time. This usually happens a lot in the night. This system overcomes the present scenario.When an accident occurs the accelerometer present in this system detects the sudden change in the vehicle movement and gets activated automatically and the buzzer tends to give alerting sounds. This system then locates the accident spot by tracking both the latitude and longitude using the Global Positioning System(GPS) and it will share the location and time to the subscriber through Global System For Mobile Communication (GSM).

It tracks the exact location and will immediately send an alerting message to one or more subscribers of the system through the GSM.All the process in this system takes place automatically without human intervention.

If the victim hadn't been harmed even after the accident the system can be set back to normal by pressing the reset switch provided in the system. Which in turn terminates from sending the alert message to the subscriber.

III. EXISTING SYSTEM :

When an accident occurs the information is only sent through GSM but there is no possibility to locate the spot where accident has occured. Due to this reason many number of lives are being lost as required medical attention cannot be given to the needy person at the right time. So far we are able communicate using GSM but unfortunately there is no innovation to evaluate on this basic problem. The only way of communication is done by contacting to the digital cellular connection. So this leads to uncertain number of accidents due to lack of exact location.

The main role of GSM in existing system is to send an message through the digital cellular connection whenever an accident had occurred manually. The existing system only applicable in the presence of the digital cellular connection and cant be operated and cannot be extended to the utmost level.

A. DRAWBACKS FOR EXISTING SYSTEM :

- Existing system can only be accessed through GSM but there is no scope to navigate the accident spot.

- The accident information can only be transmitted manually.
- Due to this many accidents are unnoticed and paying huge human loss.

IV. GSM :

GSM is a digital cellular communication system.Global System For Mobile Communications (GSM) is a technology used to establish wireless cellular connection between two or more devices.
It is used for transmitting mobile voice and data services.
GSM operates in the 900MHz and 1.8GHz bands GSM supports data transfer speeds of up to 9.6 kbps, allowing the transmission of basic data services such as SMS.

V. PROPOSED SYSTEM :

By taking the drawbacks of the existing system we proposed a new system with an elevation. This system presents an automatic accident detector and location identifier using MEMS Accelerometer, GPS and GSM where the accelerometer is used to detect the sudden change in the axis of the vehicle and this sudden change indicates that an accident has occurred while the buzzer gets activated with alerting sounds where the GPS tracks the location and GSM transmits the location information through an SMS. With the help of this system we can help immediately atleast to the person who has been met with an accident by tracing the location of the accident spot instantly.
If the victim hadn't been harmed even after the accident the system can be set back to normal by pressing the reset switch provided in the system. Which in turn terminates from sending the alert message to the subscriber.
When the vehicle is in motion and undergoes an accident, there might be chances where the power to the system might get disconnected and in such cases the secondary power source is activated which has been installed and power is supplied to the system instantly. So that the system runs continuously without any interruption.

A. ADVANTAGES:

- The sensor used is petite in size hence it can be kept in a secured place as it is not massive.
- The accident spot can be located via GPS without any delay.
- An immediate medication will be provided to the accident victims in the remote areas.
- As the project is based on IOT, it is unexceptional within itself and thus can be implemented as a safety system.

VI. GPS :

Global Positioning System (GSM) technology is used to trace the location of any object and monitor them continuously using satellite signals.
Three satellite signals are necessary to locate the receiver in 3D space and fourth satellite is used for time accuracy. GPS will give the information of parameters like longitude, latitude and altitude.
The communication takes place between GPS transceiver and GPS satellite.

VII. ACCELEROMETER :

An accelerometer is a device that measures non-gravitational accelerations.
The former provides information on taps and other handset motions allowing the development of 'gesture' user interfaces while the latter provides information on the accelerometer orientation.
Accelerometers are used in tablet computers and digital cameras so that images on screens are always displayed upright.
Also used in aero planes.
Using this we can measure tilt.

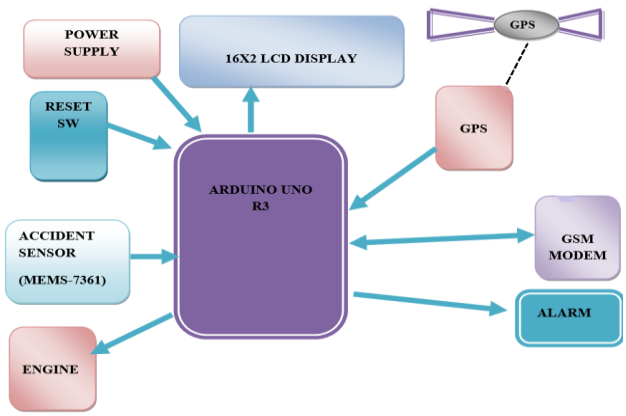


Fig 1: Block diagram

VIII. RESULT :

When the system detects an accident it sends an alerting SMS through GSM module. The message received by the subscriber is displayed in Google Maps which shows the exact location of accident and its details.

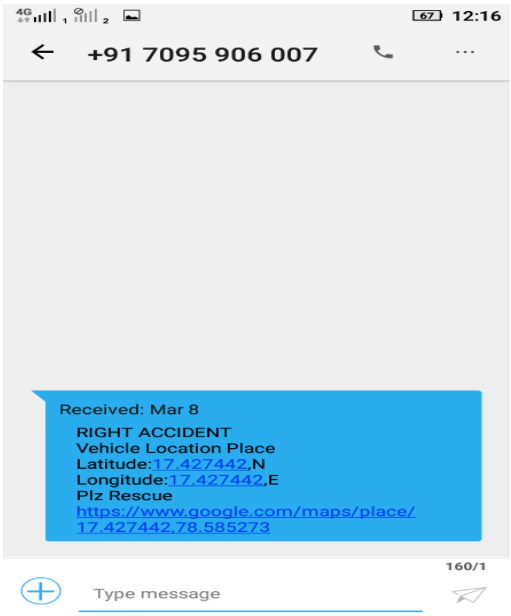


Fig 2: Alert SMS received by subscriber

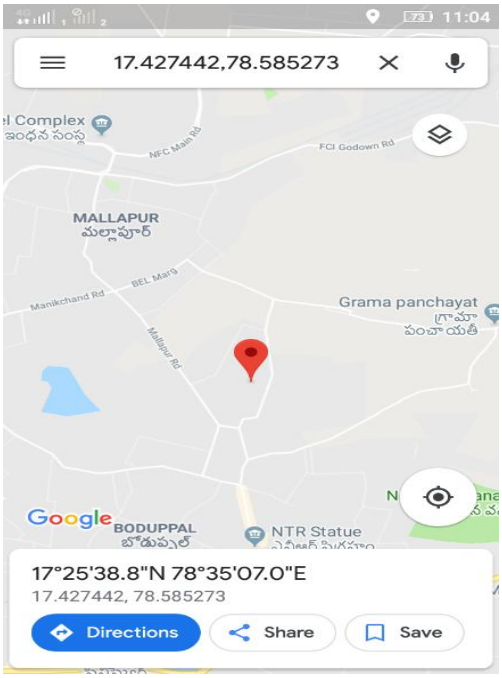


Fig 3 : Pinned location of accident displayed in google maps

IX. FUTURE SCOPE :

In some places where there is no provision of GSM network it is difficult for communication. The research work is going on for tracking the position of the vehicle even in dark clumsy areas where there is no network for receiving the signals.

X. CONCLUSION :

The advent of this project platform is based mainly on the GSM and GPS services along with MEMS Accelerometer. This system can overcome the problems that lack in existing system by detecting the exact location when an accident occurs and alerting the subscriber automatically. Main motto of the Vehicle Accident Alert System is to recognize the location of the accident spot without any delay.

REFERENCES :

1. P.Jyothi, G. Harish. *Design and Implementation of Real time vehicle Monitering, Tracking and Controlling System*, IEEE
2. Seok Lee, G. Tewolde, Jaerock Kwon, "*Design and Implementation of vehicle tracking system using GPS, GSM technology and smart phone application*, IoT 2014 IEEE
3. ***https://www.youtube.com/watch?v=sbLSdP5GxPc***
4. ***https://circuitdigest.com/microcontroller-projects/arduino-based-accident-alert-system-using-gps-gsm-accelerometer***
5. *Building Arduino Projects for the Internet of Things: Experiments with Real-World Applications* 1st ed. Edition by Adeel Javed (Author)
6. *Cognitive Sensors and IoT: Architecture, Deployment, and Data Delivery* by Fadi Al-Turjman (Author)
7. *Introduction to Internet of Things* Lecture Course from NPTEL

AUTHORS PROFILE :

Ch. Venkateswara Rao currently working as assistant professor from department of Computer Science and Engineering at Sanketika Vidhya Parishad Engineering College

K. Pavan Kumar pursuing B.Tech from department of Computer Science and Engineering at Sanketika Vidhya Parishad Engineering College

T. Charishmaa pursuing B.Tech from department of Computer Science and Engineering at Sanketika Vidhya Parishad Engineering College

P. Divya pursuing B.Tech from department of Computer Science and Engineering at Sanketika Vidhya Parishad Engineering College

V. Sangam Naidu pursuing B.Tech from department of Computer Science and Engineering at Sanketika Vidhya Parishad Engineering College

IJTIMES Impact Factor: 5.22 e-ISSN: (2455-2585)
INTERNATIONAL JOURNAL OF TECHNICAL INNOVATION IN MODERN ENGINEERING & SCIENCE

Certificate

This is to Certify that

CH VENKATESWARA RAO

is one of the author or sole author of the paper titled

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Paper ID#:150313161105

Published in IJTIMES, Volume 05 Issue 03, March-2019

Editorial Board

of

International Journal of Technical Innovation in Modern Engineering & Science(IJTIMES)

ISSN (Online): 2455-2585, UGC Approved Journal

http://www.ijtimes.com/

Managing Director

E-mail (s): editor.ijtimes@gmail.com

IJTIMES Impact Factor: 5.22 e-ISSN: (2455-2585)
INTERNATIONAL JOURNAL OF TECHNICAL INNOVATION IN MODERN ENGINEERING & SCIENCE

Certificate

This is to Certify that

K. PAVAN KUMAR

is one of the author or sole author of the paper titled

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Paper ID#:150313161105

Published in IJTIMES, Volume 05 Issue 03, March-2019

Editorial Board

of

International Journal of Technical Innovation in Modern Engineering & Science(IJTIMES)

ISSN (Online): 2455-2585, UGC Approved Journal

http://www.ijtimes.com/

Managing Director

E-mail (s): editor.ijtimes@gmail.com

IJTIMES Impact Factor: 5.22 e-ISSN: (2455-2585)
INTERNATIONAL JOURNAL OF TECHNICAL INNOVATION IN MODERN ENGINEERING & SCIENCE

Certificate

This is to Certify that

T. CHARISHMAA

is one of the author or sole author of the paper titled

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Paper ID#:150313161105

Published in IJTIMES, Volume 05 Issue 03, March-2019

Editorial Board

of

International Journal of Technical Innovation in Modern Engineering & Science(IJTIMES)

ISSN (Online): 2455-2585, UGC Approved Journal

http://www.ijtimes.com/

Managing Director

E-mail (s): editor.ijtimes@gmail.com

IJTIMES Impact Factor: 5.22 e-ISSN: (2455-2585)
INTERNATIONAL JOURNAL OF TECHNICAL INNOVATION IN MODERN ENGINEERING & SCIENCE

Certificate

This is to Certify that

P. DIVYA

is one of the author or sole author of the paper titled

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Paper ID#:150313161105

Published in IJTIMES, Volume 05 Issue 03, March-2019

Editorial Board

of

International Journal of Technical Innovation in Modern Engineering & Science(IJTIMES)

ISSN (Online): 2455-2585, UGC Approved Journal

http://www.ijtimes.com/

Managing Director

E-mail (s): editor.ijtimes@gmail.com

IJTIMES Impact Factor: 5.22 e-ISSN: (2455-2585)
INTERNATIONAL JOURNAL OF TECHNICAL INNOVATION IN MODERN ENGINEERING & SCIENCE

Certificate

This is to Certify that

V. SANGAM NAIDU

is one of the author or sole author of the paper titled

VEHICLE ACCIDENT LOCATION IDENTIFYING ALERT SYSTEM

Paper ID#:150313161105

Published in IJTIMES, Volume 05 Issue 03, March-2019

Editorial Board

of

International Journal of Technical Innovation in Modern Engineering & Science(IJTIMES)

ISSN (Online): 2455-2585, UGC Approved Journal

http://www.ijtimes.com/

Managing Director

E-mail (s): editor.ijtimes@gmail.com