POLITECHNIKA POZNAŃSKA

WYDZIAŁ ELEKTRYCZNY

INSTYTUT INFORMATYKI

ALGORYTMY WYZNACZAJĄCE OPTYMALNE TRASY PRZEJAZDU W MIEJSKIEJ SIECI ULIC, DZIAŁAJĄCE W OPARCIU O BAZĘ DANYCH PRZESTRZENNYCH

Praca dyplomowa magisterska

Krystian Nowak

promotor: dr inż. Grzegorz Waligóra

koreferent: prof. dr hab. inż. Jan Węglarz

Polaris 2 – An integrated vehicle navigation system

operating in changeable urban traffic situation.

1. Introduction

We all sorely complain of the shortness of time, and yet have much more than we know what to do with. Our lives are either spent in doing nothing at all, or in doing nothing to the purpose, or in doing nothing that we ought to do. We are always complaining that our days are few, and acting as though there would be no end of them.

-- *Lucius Annaeus Seneca 3 B.C. - 65 A.D.*

Time is a very significant part of human life. It was considered to be crucial by many great people of their times, and almost nothing has changed since that time except for the rush in our lives. On one hand we want to have as much spare time as we can, not to waste it, which entails a need of ability to moving fast from one place to another. On the other hand we still can notice all the problems connected with transport, its imperfectness, which makes an impression of powerlessness due to spending almost half of our lives in traffic jams. If we also make ourselves aware of the fact that a human life can be dependent on time, our expectations for the technology take on the form of a request for improvement of the current situation.

1.1. Motivation

The technology available nowadays is able to improve transport communication to such an extent that the impact of heavy burdens of traffic difficulties can be lessened. It is crucial, especially in the case of emergency services on which human health or even life is dependent. The problems to be solved to improve transport communication: are traffic in big cities and unknown location of destination. The location of destination problem is dependent on application domain and, in most situations, it is simple to solve. But as far as changing per minute urban traffic situation is concerned, an integrated system is required to solve the problem. The system must be capable of handling many important issues, operating at different levels of detail, and applying to different, outwardly loosely-coupled domains.

An efficient operation of a comprehensive system solving problems of traffic communication in urban network needs up-to-date knowledge of urban traffic situation, accurate location of a vehicle, one or more defined destinations, and availability of determining their location. Although there are systems which collect and store information of urban traffic situation, there is a need of their tight integration with vehicle navigation systems. A location of a vehicle, called positioning, can be accomplished using a satellite positioning system, e.g. GPS (Global Positioning System). Nevertheless, the system has to be robust to inaccuracy and temporary unavailability of the service. A solution for the problem stated above is to join information from standard devices, e.g. GPS receiver and other sources, e.g. electronic magnetic compass, odometer (called also electronic log), and a vector street map of a city the system is operating in. The route calculation has to be done in a dynamic manner, taking into consideration varied user preferences and information of urban traffic situation obtained from systems mentioned above. There are many databases containing information about places (e.g. companies, institutions, or other objects) which could be important for a client of the system and chosen as a destination of a trip. Cooperation between the system and information databases is of great importance for many applications.

The universality of the system is also an important goal to be achieved. There are a few systems available on the market. Nevertheless, they are dedicated to a specific domain, e.g. a navigation system for police cars. We aim to design a system to be universal to the largest possible extent. The system could be used in a public sector (every-day drivers, public transport), as well as in emergency and special services (ambulances, police, fire brigades, security agencies). Although the system users will be divided among many profiles, the core of the system will be responsible for the same task – an efficient passing through a labyrinth of streets in an urban network.

1.2. The goal and scope of work

The goal of our work is to design and make a prototype of an integrated vehicle navigation system. To compromise the goal we pointed out the following tasks:

- A study of recent achievements of science and industry in navigation in urban networks, specifically:
    - Spatial data storage techniques;
    - Graph theory algorithms in navigational applications;
    - Available mobile data transmission techniques – radio networks (direct communication/trunking) and GSM networks (SMS, CSD, HSCDS, WAP, and GPRS techniques);
    - Reliable and scalable server techniques (application servers);

- A design of the system which consists of three steps:
    - Data representation design – including spatial data (streets, intersections, objects, etc.) and traffic information;
    - Communication layer design. The task includes following sub-tasks:
        - between mobile unit and central unit; including transmission technique selection, data transmission security analysis and transmission costs optimization;
        - between central unit and traffic centre;
        - between central unit and management consoles;

    - Central unit design, including system configuration design preceded by system load analysis, future extension availability analysis, reliability analysis, traffic control centre communication software design, traffic data acquisition and analysis software, communication software with mobile units (in vehicles), navigational computations software, vehicle route tracking software and system management software;

- System modules implementation in the central unit, especially selected algorithms adaptation and software implementation in selected server technology, and integration with traffic control centre;

- System field studies which consist of:
    - An audit of traffic information data transformations into travel times in urban network;
    - An audit of shortest path computation in different urban traffic situations;
    - Correctness and efficiency test of designed communication protocols.

1.3. Thesis structure

The remaining part of the thesis is composed of six chapters. In chapter 2 the state of current technology used in the research will be shown. We will describe various storage methods designed for spatial data. Additionally we will show the positioning techniques available nowadays. Next we will look at various communication techniques for mobile applications.

In chapter 3 we will introduce an overall architecture of our system. In detail we will show central unit architecture and its functionality. We will also describe communication with mobile units, traffic control centre, and management consoles.

Fourth chapter deals with some implementation issues of the system prototype. It will show the implementation of central unit software and spatial database.

Fifth chapter shows system testing phase. It will show us how our system deals with real-world problems in testing field. We will describe problems connected with navigation efficiency and various aspects of cost-effectiveness and reliability of communication.

In chapter 6 we will draw conclusions after an evaluation of our system. We will summarize the progress in research and we will also point out some features of future work.

1.4. Typographic conventions

To make the reading of the thesis easier a number of typographic conventions are used in the text:

- Terms in definitions are typed in italics;
- Important text areas are typed in bold face;
- Reference numbers are composed of the number of the chapter and the number of the figure in the chapter, e.g. the third figure in the fifth chapter will be noted as Figure 5.3;
- Citations and references to the literature listed in bibliography at the end of the thesis are marked in square brackets.

2. State of the Art

2.1. Positioning techniques

2.1.1. Measurement devices for positioning purposes

In positioning the basic parameter needed to obtain is, of course, position. Position, in two-dimensional space, consists of two parameters: longitude and latitude. Both parameters can be obtained from GPS (Global Positioning System) receiver.

The Global Positioning System is a space-based navigation and positioning system that was designed by the U.S. Department of Defence to allow autonomously determining position to within 10 to 20 meters of truth.

The Global Positioning System consists of three major segments: the Space Segment, the Control Segment, and the User Segment. The space and control segments are operated by the United States Military and administered by the U.S. Space Command of the U.S. Air Force. Basically, the control segment maintains the integrity of both the satellites and the data that they transmit. The space segment is composed of the constellation of satellites as a whole that are currently in orbit, including operational, backup and inoperable units. The user segment is simply all of the end users who have purchased any of a variety of commercially available GPS receivers. The user segment includes military and civilian users.

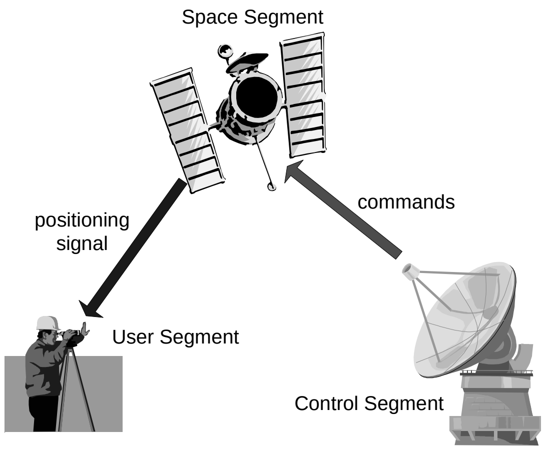


Figure 2.1 GPS system segments

The control segment of the Global Positioning System consists of one Master Control Station (MCS) located at Falcon Air Force Base in Colorado Springs, Colorado, and five unmanned monitor stations located strategically around the world. In addition, the Air Force maintains three primary ground antennas, located more or less equidistant around the equator. In the event of some catastrophic failure, there are also two backup Master Control Stations, one located in Sunnyvale, California, and the other in Rockville, Maryland. The unmanned monitor stations passively track all GPS satellites visible to them at any given moment, collecting signal (ranging) data from each. This information is then passed on to the Master Control Station at Colorado Springs via the secure DSCS (Defense Satellite Communication System) where the satellite position ("ephemeris") and clock-timing data (more about these later) are estimated and predicted. The Master Control Station then periodically sends the corrected position and clock-timing data to the appropriate ground antennas which then upload those data to each of the satellites. Finally, the satellites use that corrected information in their data transmissions down to the end user.

The space segment consists of the complete constellation of orbiting Navstar GPS satellites.

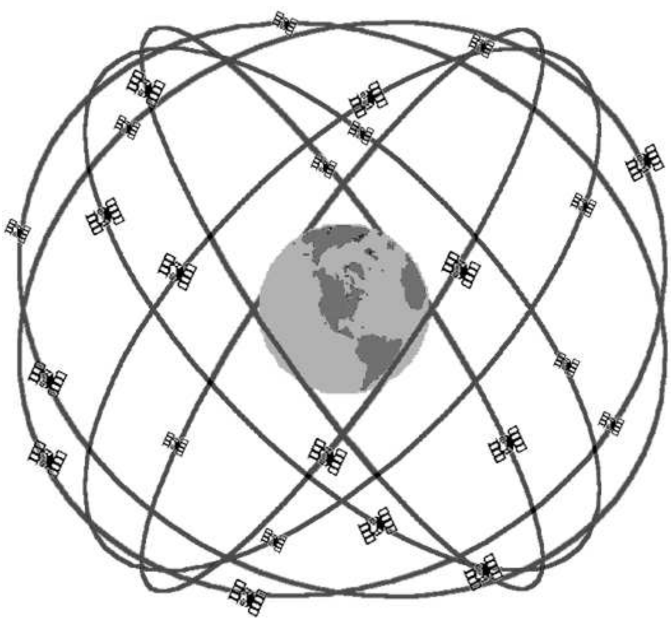


Figure 2.1 Navstar satellites orbits

Each satellite weighs approximately 900 kilograms and is about five meters wide with the solar panels fully extended. The base size of the constellation includes 21 operational satellites with three orbiting backups, for a total of 24. They are located in six orbits at approximately 20,200 kilometres altitude. Each of the six orbits is inclined 55 degrees up from the equator, and is spaced 60 degrees apart, with four satellites located in each orbit. The orbital period is 12 hours, meaning that each satellite completes two full orbits each 24-hour day.

The GPS is a distance (ranging) system, which means that the only thing that the user is trying to do is determine how far they are from any given satellite. In essence, the GPS operates on the principle of trilateration.

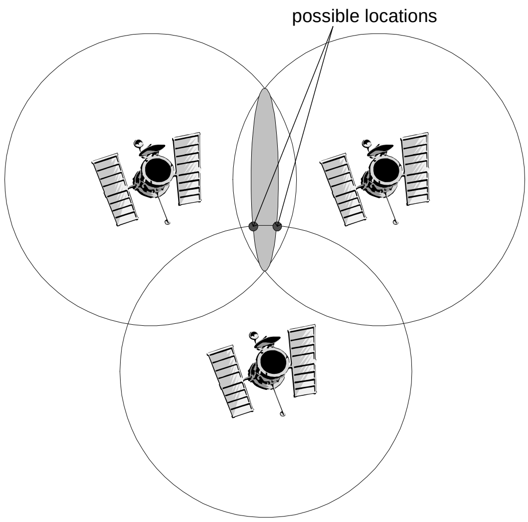


Figure 2.2 The principle of trilateration

In trilateration, the position of an unknown point is determined by measuring the lengths of the sides of a triangle between the unknown point and two or more known points (i.e. the satellites). This is opposed to the more commonly understood triangulation, where a position is determined by taking angular bearings from two points a known distance apart and computing the unknown point’s position from the resultant triangle.

To compute a location in three-dimensional space we have to solve a system of equations with four unknowns: x, y and z coefficient and time. The number of equations equals the number of visible satellites. Having an atomic clock in GPS receiver we eliminate time unknown so there is enough to get signal from at least three satellites to obtain latitude and longitude on earth surface. Since civilian receivers have to be inexpensive they are equipped with quartz clocks which are inaccurate to calculate valid position. The solution is to get signal from at least four satellites. Then we have a system of four equations with four unknowns and the result is obvious.

GPS resolution for civil purposes is degraded by a selective availability. *Selective Availability is the intentional degradation of the GPS signal by either dithering the clocks or the orbital information to produce incorrect satellite positions and, thereby, provide incorrect receiver positions.* The purpose is to limit accuracy for civilian users to a 95% probability of 100 meters or less.

Additionally there are many other features limiting the signal availability, especially in urban areas, such as:

- Multi-path – the satellite signal reaches the receivers from many angles since it is reflected by glasses of large buildings;
- Canyons – the access to satellite signal in narrow streets and between high buildings is very limited;
- Tunnels and underground parking lots – the signal is unavailable.

2.1.2. “Dead-reckoning” method

Another way to compute current position on earth surface is to use so called “dead-reckoning” method.

Dead-reckoning is a calculation of a position from the record of the course changes and distances travelled as shown by compass and log.

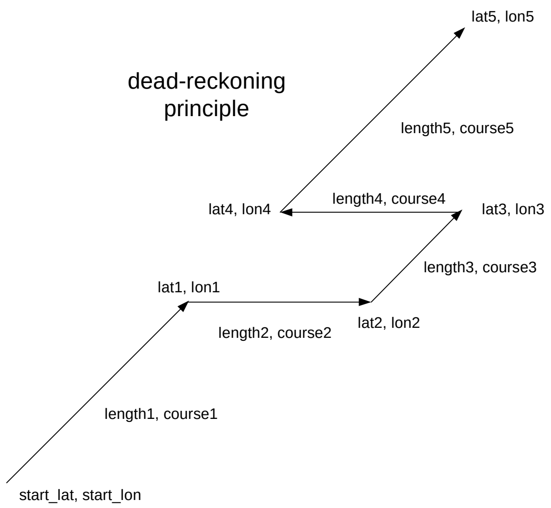


Figure 2.3 The principle of “dead-reckoning”

The coordinates of each point are computed as follows:

$lat_n = f_1(lat_{n-1}, length_n, course_n)$

$lon_n = f_2(lon_{n-1}, length_n, course_n)$

Nowadays electronic magnetic compasses and odometers (electronic logs) are used to implement dead-reckoning in urban car navigation.

2.2. Spatial data storage techniques

Spatial data provides information on the locations of various types of geographic features. This includes: transportation features such as roads, airports, and rail lines; administrative features such as counties, states, and political districts; social, economic, cultural, and demographic features such as retail centres, workplaces, residences, land use patterns and environmental features such as rivers and streams. It also includes features constructed for analytical purposes, such as transportation analysis zones.

Spatial data typically represents geographic features as points, lines, and polygons. Points, for instance, might represent the locations of airports and retail centres. Lines might represent roads and rivers. And polygons might represent counties or transportation analysis zones. The type of symbol used to represent features can vary with geographic scale. For instance, cities and airports might be best portrayed as points on a map of a country and as polygons on county map.

To describe a location of a feature on the earth surface longitude and latitude attributes are used. Since relational databases are the most widely used in industry, we have to map various spatial data types to relational model. Mapping location of a point feature is quite simple since only two attributes (single longitude and single latitude) are needed. Worst situation is in case of multi-point features.

First option is to map a feature as two tables: the first one represents the given feature (holding its id, name and so on) and the second one is a list of nodes and holds its latitudes, longitudes and id of the feature as foreign key. This modelling is quite straightforward and simple to implement in relational databases but is inefficient in queries because of the const of joining those two tables.

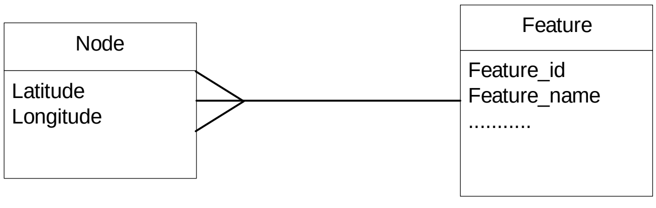


Figure 2.4 Modelling using two relations

Second option to map multi-point features in relational database is to hold a list of multiple points in special attribute in relation. Since it breaks the first normal form of relation design, database systems with spatial data storage option (thus, implementing spatial types) must be used.

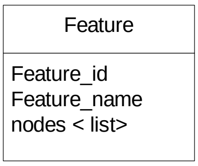


Figure 2.5 Modelling using non-1NF relation

In case of using object databases as a storage system for spatial data in most cases one of the two methods described above is used transparently.

2.3. Mobile devices

To handle remote management for any where any time there is a need to use a mobile computing unit.

Mobile computing unit is a hardware and software unit able to handle computing tasks in mobile environment.

The most ergonomically adapted to mobile management usage is to use an integrated device composing colour graphical display, touch-screen, keyboard (implemented as handwriting recognition or virtual keyboard) and voice output is to use handheld computer. Cost of such a device is in range from 200 € (palm-series) to 1200 € (iPAQ-series). This one can be utilized also in every-day life as a standard personal digital assistant.

2.4. Mobile communication

2.4.1. Radio networks

2.4.1.1. Structure of radio systems

In conventional radio system a group of radio users share one fixed channel or frequency as shown on figure below:

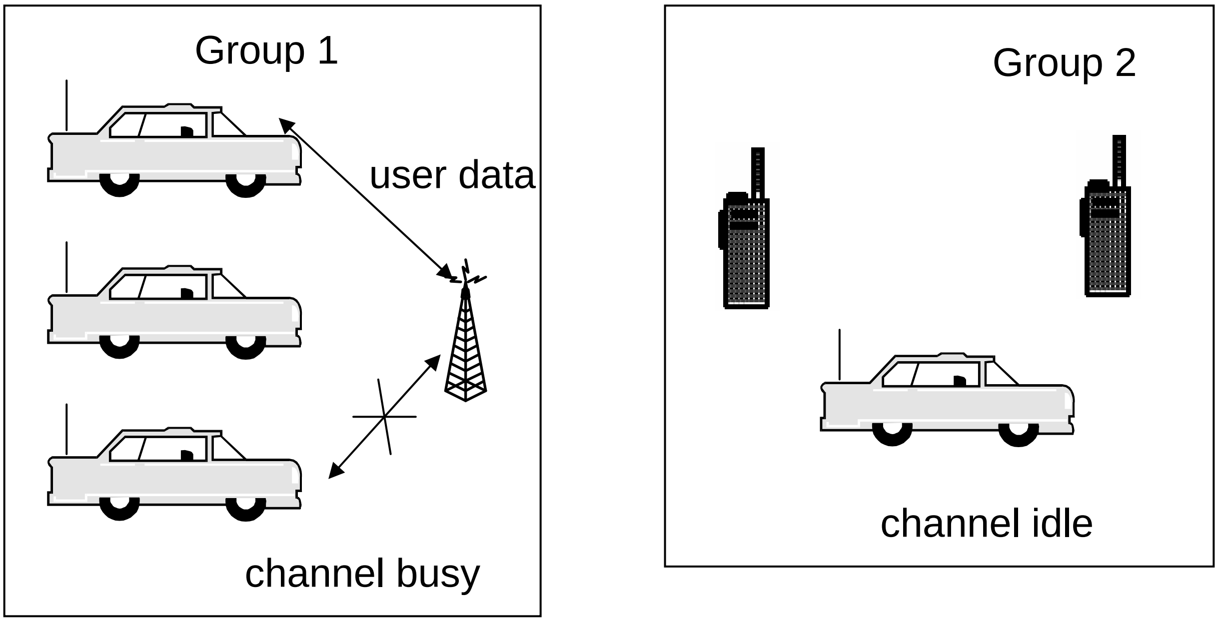


Figure 2.6. Conventional radio architecture

If the communication channel is in use by some users in the workgroup, service is not available to others. Often the radio channel is idle, which is a poor utilization of this valuable resource.

In a trunked network, user groups share multiple channels, with queuing and channel assignment being handled dynamically by the system infrastructure.

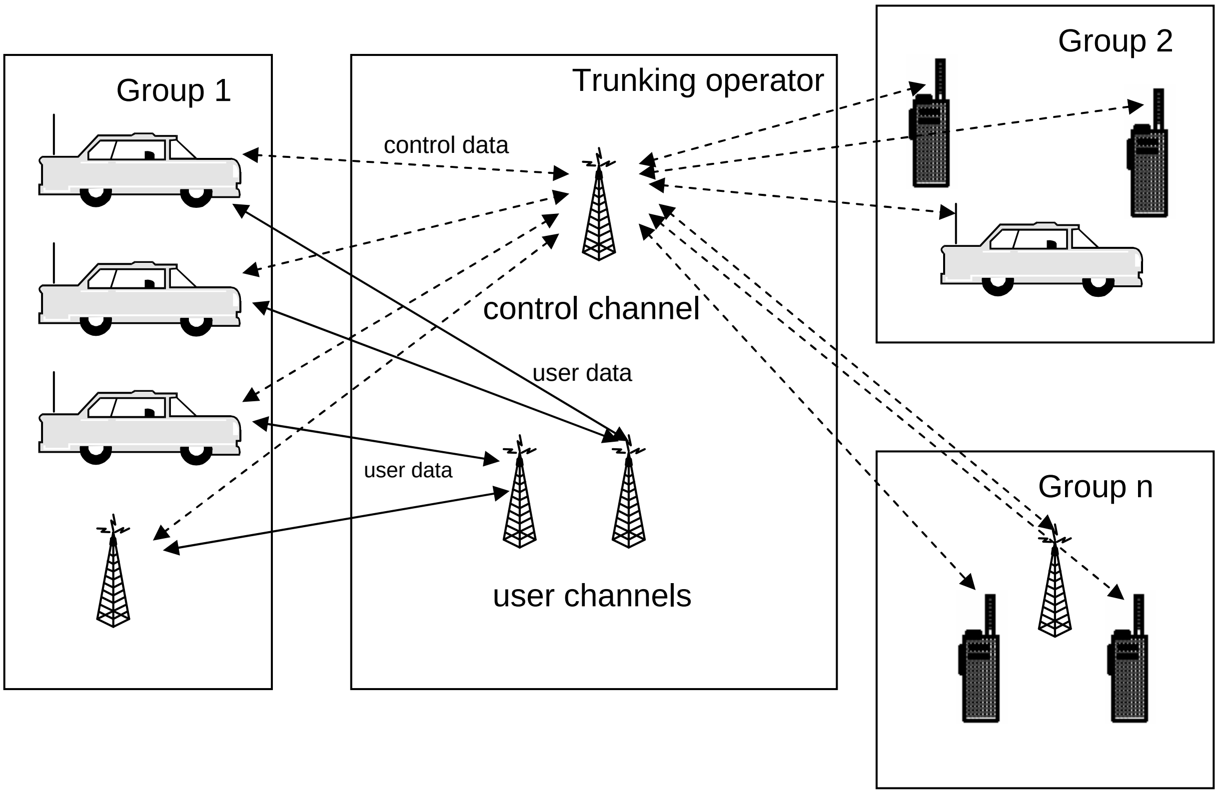


Figure 2.7. Trunking radio architecture

Trunking permits a large number of users to share a relatively small number of communication channels (trunks). Trunking is a wireless version of commercial telephone communication (PTSN). When the radio user places a call, the trunking radio system automatically allocates an available free channel. Channel selections and other decisions, which normally made by the radio user, are made by the central controller using control channel. Channel reassignments are automatic and completely transparent to individual users.
Because of its flexibility a trunked system can be expanded easily to accommodate a growing number of users and restructuring as needed.

2.4.1.2. Data transmission techniques – Packet Radio.

Packet radio is digital transmission method available for use via radio transceivers. Packet radio has its roots in the 1960s, in research conducted organizations such as the Rand Corporation and DARPA. The best known project was ALOHANET, a research network linking computers in different parts of the Hawaiian Islands by radio links. This work eventually attracted the interest of amateur radio operators. In 1978, regulatory changes in Canada paved the way for amateur operators to begin experimenting with packet radio techniques, leading to the first design for a TNC (Terminal Node Controller, a device for interconnecting a computer or data terminal to a radio). Interest soon spread to the USA, and in the early 80s several new TNC designs appeared, accompanied by work on standardizing packet radio protocols.

The boom in amateur packet radio began in earnest in 1985, when a group called Tucson Amateur Packet Radio (TAPR) introduced a new TNC called the TNC-2, and made it available as a kit. The TNC-2, and other designs which followed, includes a built-in 1200 bps modem. Although slow by computer networking standards, this bit rate was chosen because inexpensive modem chips were available, and because the modem can be interfaced to virtually any voice radio transceiver. Transmitting at higher bit rates using radios which were designed for voice transmission is much less straightforward. One of the more popular protocols used for amateur packet radio is known as AX.25. This is similar to the commercial X.25 standard.

2.4.2. GSM networks

2.4.2.1. History of the GSM networks

The Global System for Mobile communications (GSM) is a digital cellular communications system. It was developed in order to create a common European mobile telephone standard but it has been rapidly accepted worldwide.

The idea of cell-based mobile radio systems appeared at Bell Laboratories (in USA) in the early 1970s. However, mobile cellular systems were not introduced for commercial use until the 1980s. During the early 1980s, analog cellular telephone systems experienced a very rapid growth in Europe, particularly in Scandinavia and the United Kingdom. Today cellular systems still represent one of the fastest growing telecommunications systems.

But in the beginnings of cellular systems, each country developed its own system, which was an undesirable situation for the following reasons:

- The equipment was limited to operate only within the boundaries of each country.
- The market for each mobile equipment was limited.

In order to overcome these problems, the Conference of European Posts and Telecommunications (CEPT) formed, in 1982, the Groupe Spécial Mobile (GSM) in order to develop a pan-European mobile cellular radio system (the GSM acronym became later the acronym for Global System for Mobile communications). The standardized system had to meet certain criteria:

- Spectrum efficiency;
- International roaming;
- Low mobile and base stations costs;
- Good subjective voice quality;
- Compatibility with other systems such as ISDN (Integrated Services Digital Network);
- Ability to support new services.

Unlike the existing cellular systems, which were developed using an analog technology, the GSM system was developed using a digital technology.

In 1989 the responsibility for the GSM specifications passed from the CEPT to the European Telecommunications Standards Institute (ETSI). The aim of the GSM specifications is to describe the functionality and the interface for each component of the system, and to provide guidance on the design of the system. These specifications will then standardize the system in order to guarantee the compatibility between the different elements of the GSM system. In 1990, the phase 1 of the GSM specifications were published but the commercial use of GSM did not start until mid-1991. Since 1992 the GSM system has evolved rapidly:

- 1992 - Coverage of larger cities and airports;
- 1993 - Coverage of main roads GSM services start outside Europe;
- 1995 - Phase 2 of the GSM specifications Coverage of rural areas.

From the evolution of GSM, it is clear that GSM is not anymore only a European standard. GSM networks are operational or planned in over 80 countries around the world. The rapid and increasing acceptance of the GSM system is illustrated with the following figures:

- 1.3 million GSM subscribers worldwide in the beginning of 1994;
- Over 5 million GSM subscribers worldwide in the beginning of 1995;
- Over 10 million GSM subscribers only in Europe by December 1995.

2.4.2.2. The cellular structure

In a cellular system, the covering area of an operator is divided into cells. A cell corresponds to the covering area of one transmitter or a small collection of transmitters. The size of a cell is determined by the transmitter's power.

The concept of cellular systems is the use of low power transmitters in order to enable the efficient reuse of the frequencies. In fact, if the transmitters used are very powerful, the frequencies can not be reused for hundred of kilometres as they are limited to the covering area of the transmitter.

The frequency band allocated to a cellular mobile radio system is distributed over a group of cells and this distribution is repeated in all the covering area of an operator. The whole number of radio channels available can then be used in each group of cells that form the covering area of an operator. Frequencies used in a cell will be reused several cells away. The distance between the cells using the same frequency must be sufficient to avoid interference. The frequency reuse will increase considerably the capacity in number of users.

In order to work properly, a cellular system must verify the following two main conditions:

- The power level of a transmitter within a single cell must be limited in order to reduce the interference with the transmitters of neighbouring cells.

- Neighbouring cells can not share the same channels. In order to reduce the interference, the frequencies must be reused only within a certain pattern.

In order to exchange the information needed to maintain the communication links within the cellular network, several radio channels are reserved for the signalling information.

2.4.2.3. Architecture of GSM network

The GSM technical specifications define the different entities that form the GSM network by defining their functions and interface requirements.

The GSM network can be divided into four main parts:

- The Mobile Station (MS);
- The Base Station Subsystem (BSS);
- The Network and Switching Subsystem (NSS);
- The Operation and Support Subsystem (OSS).

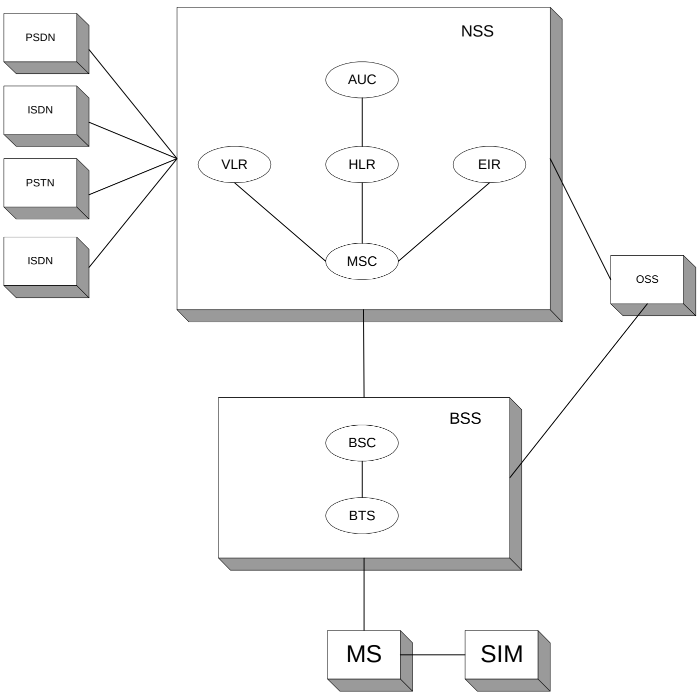


Figure 2.8 GSM network structure

A Mobile Station (MS) consists of two main elements:

- The mobile equipment or terminal (fixed, portable or handheld);
- The Subscriber Identity Module (SIM) - a smart card that identifies the terminal. By inserting the SIM card into the terminal, the user can have access to all the subscribed services.

The Base Station Subsystem (BSS) connects the Mobile Station and the NSS (Network and Switching Subsystem). It is in charge of the transmission and reception. The BSS can be divided into two parts:

- The Base Transceiver Station (BTS) or Base Station;
- The Base Station Controller (BSC).

The BTS corresponds to the transceivers and antennas used in each cell of the network. Each BTS has between one and sixteen transceivers depending on the density of users in the cell. The BSC controls a group of BTS and manages their radio resources. A BSC is principally in charge of handovers, frequency hopping, exchange functions and control of the radio frequency power levels of the BTSs.

Its main role of the Network and Switching System (NSS) is to manage the communications between the mobile users and other users, such as mobile users, ISDN users, fixed telephony users, etc. It also includes data bases needed in order to store information about the subscribers and to manage their mobility.

The central component of the NSS is The Mobile services Switching Centre (MSC). The MSC performs the switching functions of the network. It also provides connection to other networks.

The Gateway Mobile services Switching System (GMSC) is a node interconnecting two networks. The GMSC is the interface between the mobile cellular network and the PSTN. It is in charge of routing calls from the fixed network towards a GSM user.

The Home Location Register (HLR) is a very important database that stores information of the subscribers belonging to the covering area of a MSC. It also stores the current location of these subscribers and the services to which they have access.

The Visitors Location Registers (VLR) contains information from a subscriber's HLR necessary in order to provide the subscribed services to visiting users. When a subscriber enters the covering area of a new MSC, the VLR associated to this MSC will request information about the new subscriber to its corresponding HLR. The VLR will then have enough information in order to assure the subscribed services without needing to ask the HLR each time a communication is established.

The Authentication Centre (AuC) register is used for security purposes. It provides the parameters needed for authentication and encryption functions.

The Equipment Identity Register (EIR) is a register containing information about the mobile equipments. More particularly, it contains a list of all valid terminals. A terminal is identified by its International Mobile Equipment Identity (IMEI). The EIR allows then to forbid calls from stolen or unauthorized terminals (e.g., a terminal which does not respect the specifications concerning the output RF power).

The GSM Internetworking Unit (GIWU) corresponds to an interface to various networks for data communications. During these communications, the transmission of speech and data can be alternated.

The Operation and Support Subsystem (OSS) is connected to the different components of the NSS and to the BSC, in order to control and monitor the GSM system. It is also in charge of controlling the traffic load of the BSS.

2.4.2.4. GSM services

Three categories of services can be distinguished:

- Teleservices: telephony, facsimile , emergency calls, Short Message Service (SMS);
- Bearer services: asynchronous and synchronous user data;
- Supplementary Services: call forwarding, call barring, call waiting, advice of charge, call hold, closed user group, calling line identification presentation, calling line identification restrictions, connected line identification presentation, connected line identification restrictions.

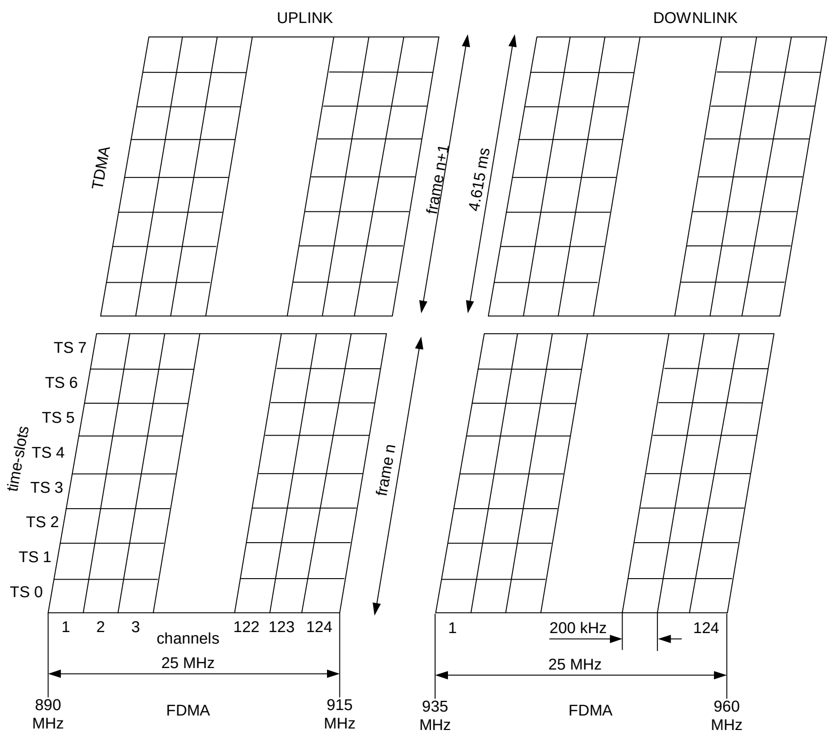
2.4.2.5. GSM data services

GSM offers low speed communication services. The services are divided into two levels as determined in the Open System Interconnection (OSI) reference model: services to users and teleservices: Services to GSM users and Teleservices.

- Services to GSM users provide users with the possibility of transmitting signals between interfaces on the same network or on different networks. The two transparent or non-transparent modes are supported.
- Teleservices are communication protocols used for voice or data transmissions between two terminals.

GSM technology incorporates various platforms that have evolved throughout the years: Circuit Switched Data (CSD), High Speed Circuit Switched Data (HSCSD), General Packet Radio Service (GPRS), and Enhanced Data GSM Environment (EDGE). These platforms operate throughout the large amount of GSM networks that are operating and serve to transfer data through GSM phones.

The way in which the GSM technology works with data transfer is fairly straight-forward. In order to increase the efficiency of the communication, each frequency is further divided into so-called time-slots (TS) – Time Division Multiple Access (TDMA). The duration of a time-slot is approximately 0.05 second, and there is eight timeslots. In GSM-900 frequency band is divided into 124 channels – Frequency Division Multiple Access (FDMA). Each channel is divided into 8 timeslots.



As time-slot number 0 is usually used to communicate control information, this gives the possibility to have 868 calls (voice or data) at the same time.

One channel transfers 114 information bits per timeslot. One channel occupies 24 timeslots in 120ms. One channel has a raw data rate 22.8kbits per second.

Data speeds are determined by the speed of an individual time slot, and the number of time slots used.

CSD

CSD offers 9.6 Kbps data transmission over a single time slot. Since this platform is the most basic of the GSM platforms, most GSM phones will support this base. CSD however, has very few users today. Problems with the CSD technology was the connection time, sometimes taking up to 30 seconds.

HSCSD

HSCSD is the high speed version of CSD and can offer data transmission speeds of 9.6 or 14.4 Kbps per slots. Multiple time slots can be combined to create higher speeds, up to 38.4 and 57.6 Kbps respectively. Regardless of the increased speed, there are still issues with efficiency with this platform. In an actual network, users are typically allotted between 2 and 4 time slots with a HSCSD network.

GPRS

GPRS is designed for irregular data traffic. Packet switched data is transferred only on timeslots not used for voice connections. GSM voice traffic is prioritized and GPRS connection stays connected without wasting radio resources. With GPRS, each time slot can be 9.6 to 21.4 kbps. Multiple time slots can be combined for higher data speeds, up to 171.2 Kbps if incorporating all 8 time slots. With technology hindrances in place, current phones are limited to lower data speeds. Though boasting speeds that incorporate 8 time slots, this is realistically unpractical. In actuality, a GPRS platform will perform a data connection of 4 downstream slots coupled with 1 upstream slot.

EDGE

The EDGE technology is the cutting edge of the GSM platform capabilities. Just as HSCSD was an extension of CSD, EDGE is an extension of the GPRS platform. EDGE increases maximum time slot speed to 48 kbps, pushing maximum combined data speed up to 384 kbps if using all 8 time slots. The EDGE technology is not yet widely deployed, as this will be the base for the upcoming generation of GSM platforms. The EDGE platform will assist in the delivery advanced mobile services.

SMS

SMS is an exceptional method of transmitting data because the messages are supported by signal networks meaning that a message can be sent or received during a call (voice or data). In this way, SMS can reach users even if they are not available at the time, or if the line is busy. The phone may be turned off or may be out of range.

SMS provides for store and forward, which means the text messages are stored on the SMS Center (SMSC) until the mobile subscriber is available to accept delivery of the message.

An SMS message is limited to 140 bytes of data (160 7 bit characters) and the messages are sent between mobile operator SMSCs using the global SS7 telecommunications network.

2.4.3. Mobile communication techniques comparison

In order to choose the best communication method we should take into account costs of the equipment needed (i.e. transceivers, antennas, mobile phones, modems), costs of getting access to the network, costs of subscriptions to network access (per month), and costs of data transmissions. To calculate the transmission costs we have made the assumption that the amount of transmitted data per month equals to average size of a vector map of an average city (i.e. 10MB for a 1 million citizens city).

	Conventional Radio	**Trunking Radio**	**SMS**	**CSD**	**HSCSD**	**GPRS**
Equipment cost	300,00 €	400,00 €	1,00 €	1,00 €	1,00 €	1,00 €
Network access cost	500,00 €	1,00 €	1,00 €	1,00 €	1,00 €	1,00 €
Subscription per month	1,00 €	50,00 €	10,00 €	10,00 €	10,00 €	10,00 €
Data unit	packet	packet	message	Minute	minute	byte
Data size per unit [kB]	1	1	0,1367188	70	280	100
Data units transmitted	10240	10240	74898	146	37	102
Unit cost	0,00 €	0,00 €	0,08 €	0,25 €	1,00 €	0,25 €
Transmission cost	0,00 €	0,00 €	5 991,86 €	36,57 €	36,57 €	25,60 €
Transmission speed kB/s	0,15	0,15	0,02	1,17	4,68	14,06
Transmission time [min]	1138	1138	8533	146	36	12
Total cost per year	812,00 €	1 001,00 €	72 024,35 €	560,86 €	560,86 €	429,20 €

Criteria used by selection are: effectiveness (transmission time) and total costs per year. In every respect the GPRS platform is the most attractive option.

2.5. Enterprise Java Beans as a server-side architecture

The **Enterprise JavaBeans** (EJB) standard is a component architecture for deployable server-side components in written in Java programming language [Roman et al., 02]. It is an agreement between components and application servers that enable any component to run in any application server. EJB components (called enterprise beans) are deployable, and can be imported and loaded into an application server, which hosts those components.

The top three values of EJB are as follows:

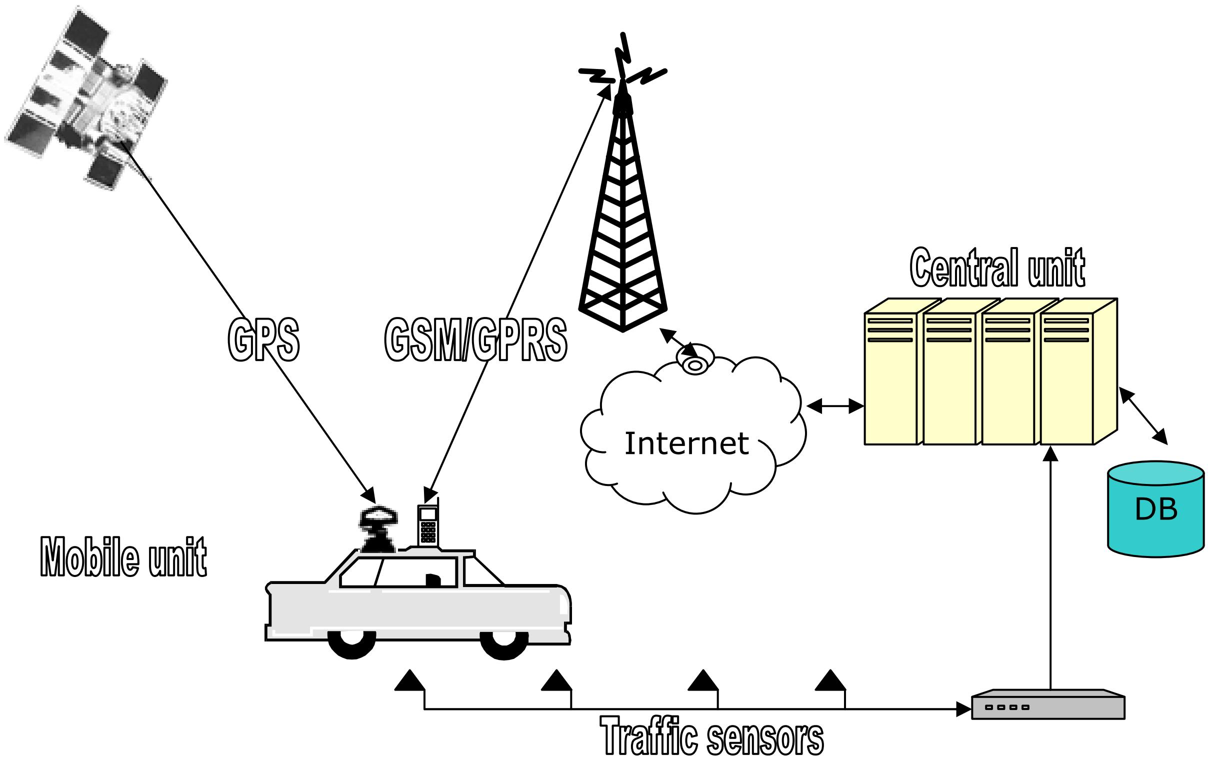
- It is agreed upon by the industry. Those who use EJB will benefit from its widespread use. Because everyone will be on the same page, in the future it will be easier to hire employees who understand this class of systems (since they may have prior EJB experience), learn best practices to improve the system, partner with businesses (since technology will be compatible), and sell software (since customers will accept a well known solution). The concept of “train once, code anywhere” applies.
- Portability is easier. The EJB specification is published and available freely to all. Since EJB is a standard, one do not need to gamble on a single, proprietary vendor’s architecture. And although portability will never be free, it is cheaper than without a standard.
- Rapid application development. An application can be constructed faster because a developer gets middleware from the application server. There’s also less of a mess to maintain.

Nowadays, there are many vendors who offer EJB compliant application servers e.g. Oracle Corporation (Oracle 10g Application Server), IronFlare AB (Orion Application Server), BEA (WebLogic Server 8.1), IBM (WebSphere Application Server) or even an open source JBoss created by the JBoss Group.

3. System design

3.1. Overall system architecture

The system consists of two interconnected parts: mobile unit and central unit. Mobile unit will be mounted in a vehicle and will communicate with central unit through GSM/GPRS network. The main function of mobile unit is to determine vehicle position and to assist the driver by helping in navigation to chosen destination, whereas the main function of the central unit is to acquire current traffic situation from traffic sensors, store it in database and compute most efficient paths in urban network for mobile units.



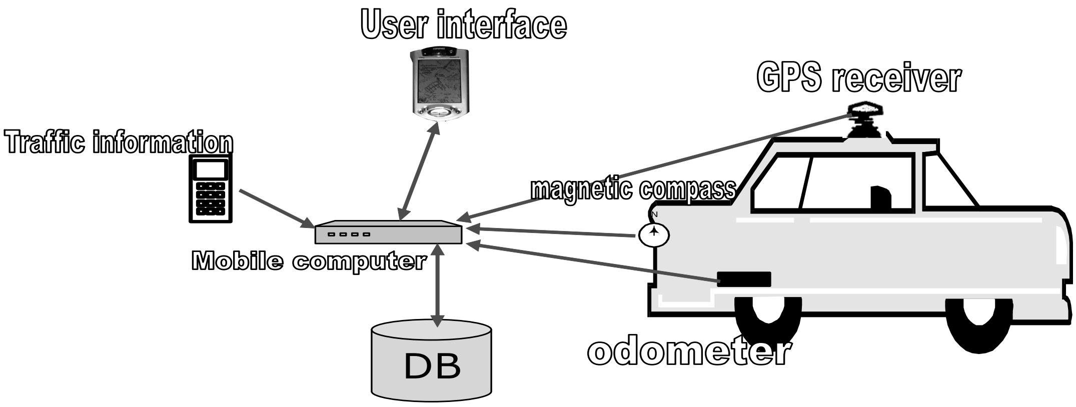
To determine vehicle position mobile unit gathers measurements from GPS receiver, electronic magnetic compass, odometer (electronic log). In case of GPS signal being temporarily inaccessible, a current vehicle position can be computed using remaining measurement devices and “dead-reckoning” method.

3.2. Mobile unit architecture

The architecture of mobile unit is fully described in a thesis written by Tomasz Pużak [Pużak, 03].

The mobile unit consists of seven parts:

- GPS receiver;
- electronic magnetic compass;
- odometer (electronic log);
- mobile computer;
- mobile database;
- personal digital assistant as a user interface;
- GSM/GPRS communication device for traffic information retrieval.



3.3. Central unit architecture

The central unit consists of three main parts:

- traffic data integration module;
- mobile unit services module;
- management module.

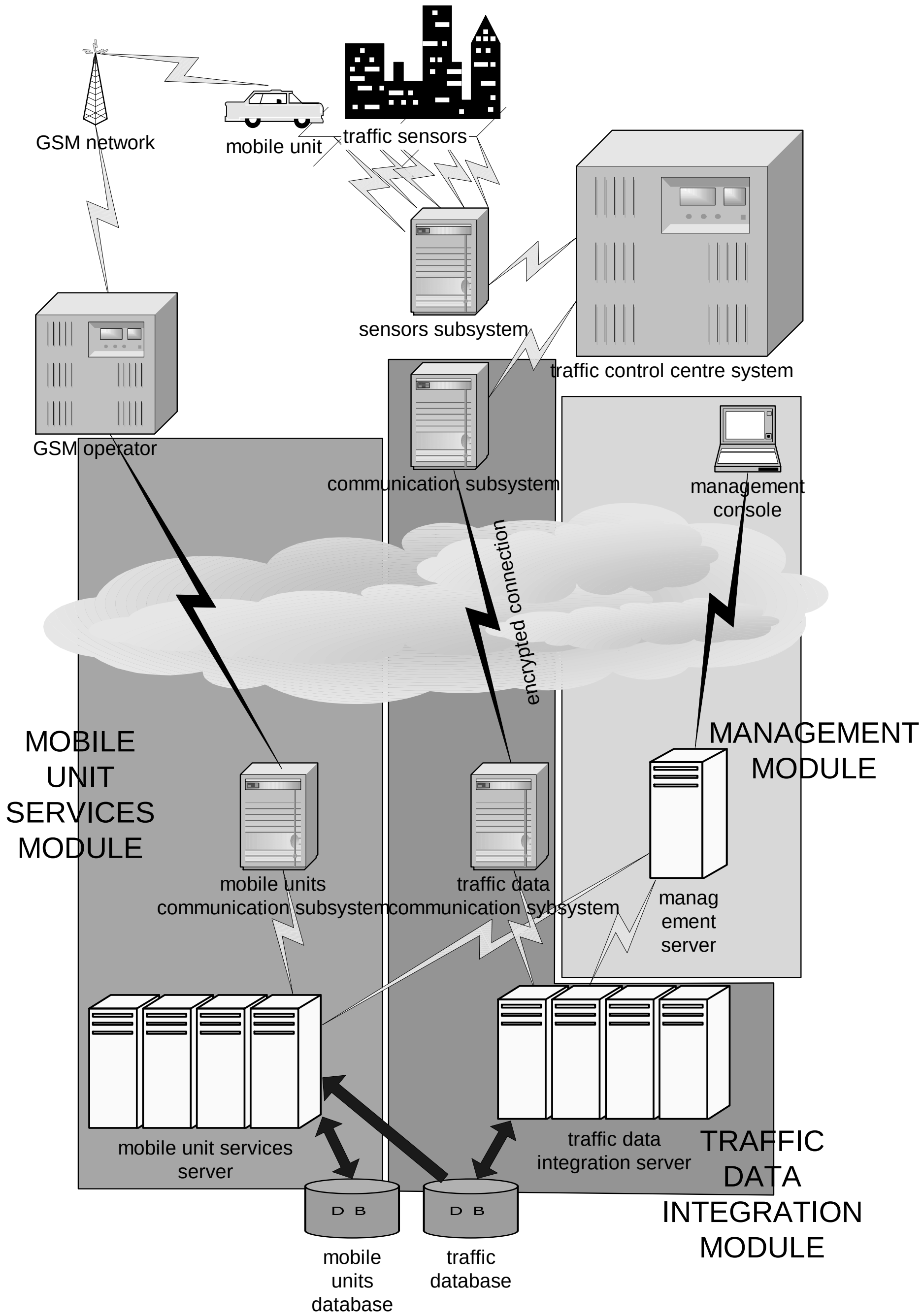


Figure 3.1 Central unit architecture

The traffic data integration module is responsible for acquiring current traffic situation data from traffic control centre, process it and store in traffic database. Two communication subsystems, being a part of the module, convert the traffic data from an internal traffic centre native form to a more universal one, send it from traffic control centre side to traffic data integration server side and then pass it to traffic data integration server. Since the data transferred from traffic control centre is very valuable, an encrypted connection between communication modules have to be used. The traffic data integration server calculates the traffic data given as the number of cars passed a traffic sensor into form of a travel time on an arc in urban network graph. Then the server stores the results in traffic database updating travel times of each arc.

The mobile unit services module is responsible for handling mobile units' requests by finding locations of destination objects, computing shortest paths from mobile unit's location to a location of the destination and send it back to given mobile unit. To perform the task the module uses the data obtained from traffic database and mobile unit database. The mobile units communication subsystem handles network communication with mobile units, authorises the communication sessions and encrypts the data. The mobile unit services server's main task is to compute shortest path from a given source to a given destination. The location of the source is known since this is a location of mobile unit sent in the request, whereas the location of the destination has to be obtained from the traffic database where the objects of various types are stored along with their locations. Also a session data of a request for each mobile unit is handled by the module for accounting purposes.

The management module main function is to control system parameters and to supervise traffic data integration module and mobile unit services module operation. Also a various types of reports can be produced as a summary for auditing purposes. The module consists of management server which connects to two controlled modules and management console presenting data to the operator and prompting for commands.

3.4. Spatial data model

The urban network is represented in a form of direct graph with a set of nodes and arcs. Streets are represented as arcs in a graph. Intersections are represented as nodes.

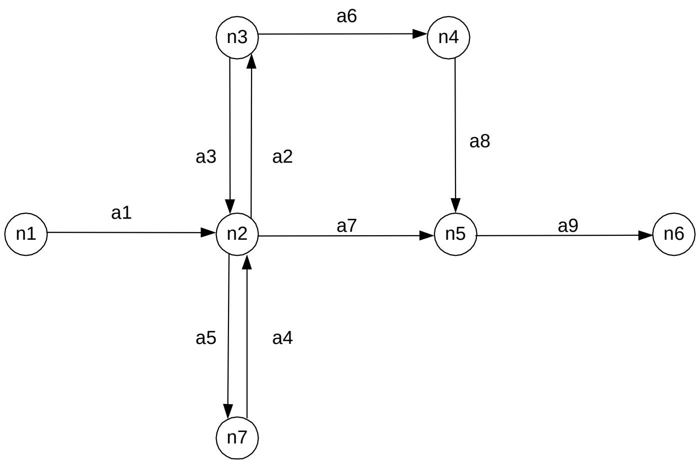


Figure 3.2 Urban network represented as a graph

Also urban objects (e.g. buildings, parking lots, gates etc.) are stored in a database. An object position is represented by a pair of an identifier of an arc the object is located at and an offset from a beginning of the arc.

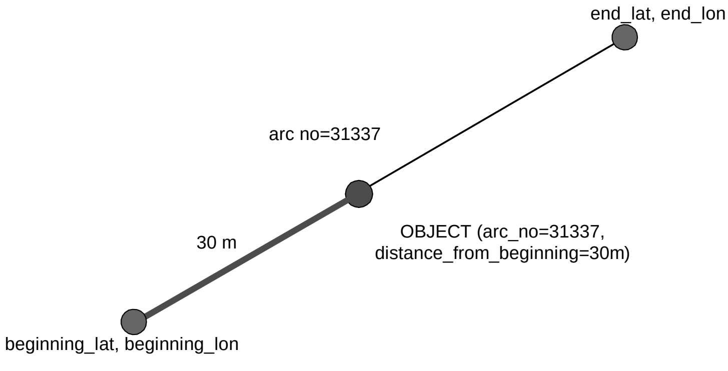


Figure 3.3 Offset object representation

That representation can be easily transformed to longitude and latitude representation because we always know the latitude and longitude of the beginning of the arc:

$object\_lat = f_{lat}(arc\_beginning\_lat, object\_distance\_from\_beginning)$
$object\_lon = f_{lon}(arc\_beginning\_lon, object\_distance\_from\_beginning)$

To represent the ability or inability to turn left, right or to go straight (generally to make a change from one arc to another) a relation of allowed moves has been introduced. For example to represent a situation that we can go from arc no. 1 to no. 2 but we cannot go from arc no. 1 to arc no. 3 we define a relation of allowed moves from 1 to 2.

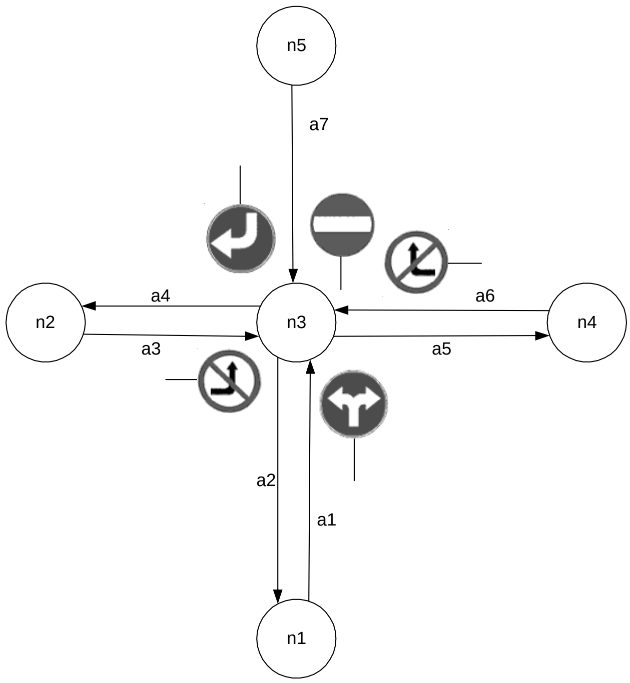


Figure 3.4 An example of allowed moves relation

In the example above we have following allowed moves relations defined:

- from arc a1 to arcs: a4 (turn left), a5 (turn right), a2 (turn back);
- from arc a3 to arcs: a2 (turn right), a5 (go straight) (turning left and also back is not allowed);
- from arc a7 to arc a4 (turn right);
- from arc a6 to arcs: a4 (go straight), a2 (turn left), a5 (turn back).

The data model to store all the information described above has been designed using OMT methodology is shown in figure below:

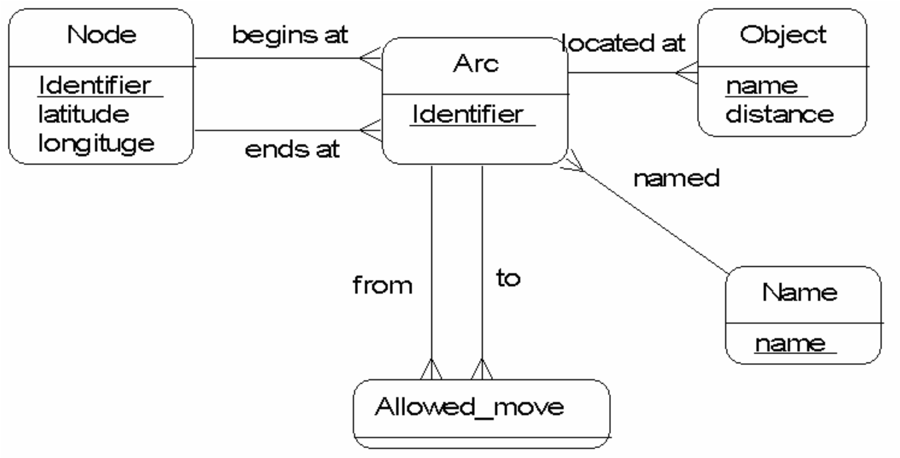


Figure 3.5 A model of spatial data for urban networks

According to the principles of database design [Ullman et al, 02] a data model should meet at least the requirements of third normal form.

*A relation is in **First Normal Form** (1NF) if and only if each attribute of the relation is atomic.*

*A relation is said to be in **Second Normal Form** if it fulfils the requirements of First Normal Form and each non-key attribute in the relation is functionally dependent upon the primary key.*

*A relation is said to be in **Third Normal Form** if it fulfils the requirements of Second Normal Form and all its attributes are functionally dependent upon the primary key.*

The model of spatial data shown in Figure 3.5 meets the requirements of third normal form.

The urban data available on the market offer much more information. It could be used in application for users of mobile units. The additional knowledge is represented by following attributes:

- node's level – the ordinal number of a node multi-level intersection or multi-level intersection;
- node's type – a type of a node e.g. road continuation, curve, intersection, round about;
- arc's level – the ordinal number of an arc on flyover or multi-level intersection;
- arc's type – a type of a road represented by the arc e.g. dirt road, access road, side road or main road;
- object's type – a type of an object e.g. a bank, a shop, a parking lot;
- arc twin relation - a connection between a pair of arcs if there is one.

To represent a fact that there is an arc in the opposite direction a unary relationship named "twin" has been introduced. Additionally there is a need to store information of travel time along the arc. The system will compute and store there the information to calculate the shortest path when requested by the user.

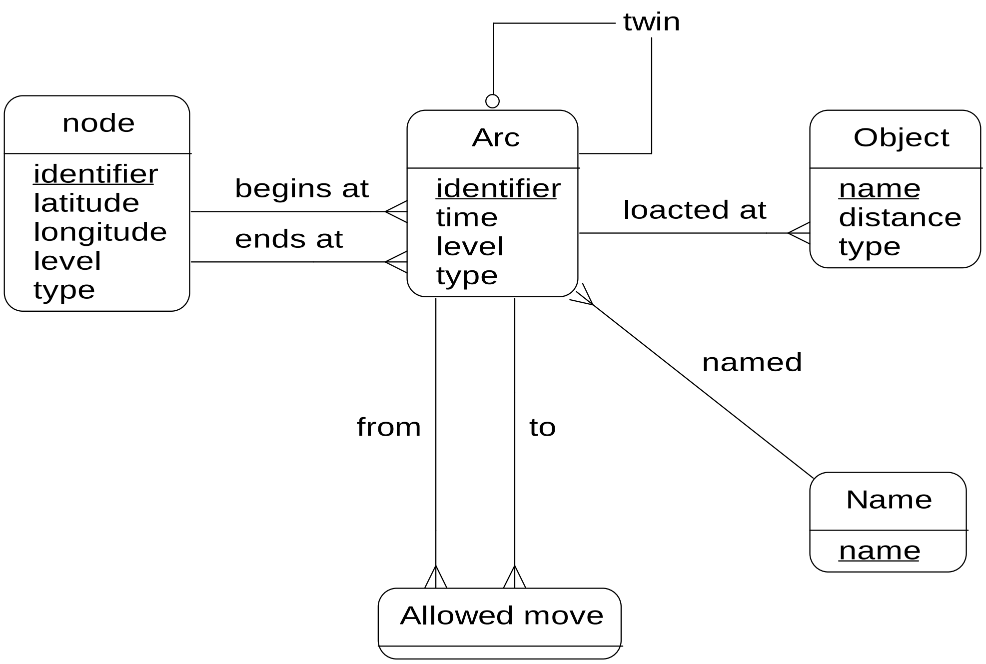


Figure 3.6. A model of spatial data including additional information

In the figure below a storage version of data model is presented showing foreign keys as attributes in relations.

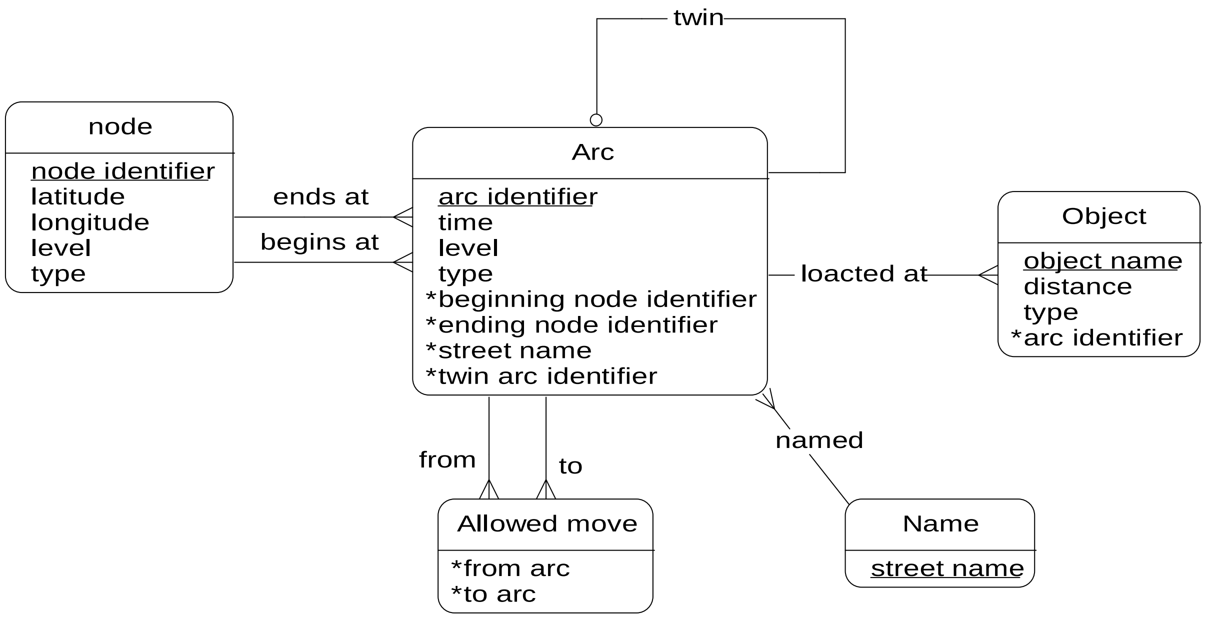


Figure 3.7 A model of spatial data including foreign keys

The most frequent database operation in navigational and positioning algorithms is joining arc and node tables. It is generally known that the join operation in database systems is the most resource expensive one. As a result of the fact we have to de-normalize the relation by copying the latitude and longitude of beginning and ending node to a relation representing an arc. To speed-up the access along foreign keys two artificial keys has been introduced: an object identifier and a street identifier.

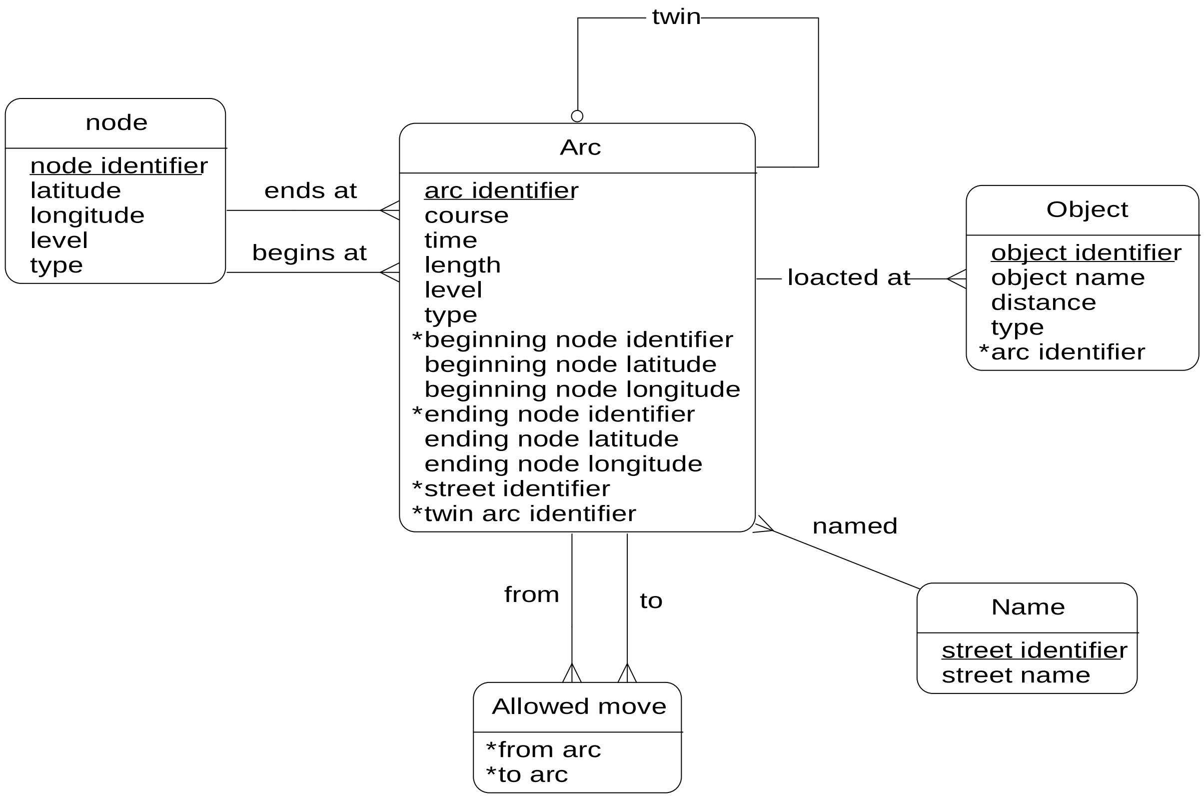


Figure 3.8 A model of spatial data after denormalization and adding artificial keys

After the de-normalization the data model the model of spatial data meets only the requirements of first normal form. Since the data is not expected to be changed no update and delete anomalies are supposed to occur. The model in a form shown in Figure 3.8 is sufficient for the system described in the thesis.

3.5. Navigation algorithm

The most important assumption in urban network navigation is that the road map of the network is represented as a directed graph. It is directly connected with the spatial data model previously described. Having determined current position of a mobile unit we can also establish a starting point for navigation algorithm. The location of a destination for the algorithm is acquired by mobile unit's query for the destination object. Now the problem is to find the shortest path from the starting point to the destination. Here shortest means having the smallest time of travel. Since the traffic information from traffic control centre is transformed into travel times of each arc in traffic database, the travel times are easy to acquire.

3.5.1. Shortest path algorithm

A directed graph (N, A) is a set of nodes $N = \{1, 2, 3, \dots, n\}$ *and a set of arcs* $A \subseteq \{ (i, j)$ *where* $i, j \in N$ *and* $i \neq j \}$

A path is a the sequence of nodes $(i_1, i_2, i_3, \dots, i_k)$ *such that* (i_m, i_{m+1}) *is an arc for all* $m=1, \dots, k-1$*. The last node of the path is called a terminal node.*

A length of a path is a sum of lengths of its arcs.

The shortest path problem is to find a path between two given nodes which length is the minimal one among all paths between these two nodes.

We also assume that all arcs have positive lengths. This is true if a length of a path is its travel time since it must be a positive value. Second assumption is that there is at most one arc between two nodes in each direction.

The Dijkstra’s method – the Johnson’s variant is described below:

Let **V** is a set of nodes.
Let **s** is the beginning (source) node of the path.
Let **t** is the ending (terminal) node of the path.
Let $\mathbf{distance_v}$ is the distance from starting node to v’th node.
Let $\mathbf{final_v}$ is the predicate if the node’s $\mathbf{distance_v}$ value is a final one.
Let $\mathbf{predecessor_v}$ is the v’s preceding node on the path from beginning node to the terminal node.
Let **current** is the node currently the algorithm is considering.
Let $\mathbf{w_{i,j}}$ is the weight (length) of the arc from node i to node j.

Step 1 – Initialization:

For each node **v** in **V** but **s**:

$\mathbf{distance_v} = \infty$

$\mathbf{final_v} =$ false

$\mathbf{predecessor_v}$ = UNDEFINED

$\mathbf{distance_s} = 0$ // the distance from source to source equals 0
$\mathbf{final_s}$ = true // and the distance is final
current = s // we start the algorithm from the source node

Step 2 – Iteration:

While $\mathbf{final_t}$ is false do:

For each **current**’s direct successor **v** if $\mathbf{final_v}$ is false do:

new_distance = $\mathbf{distance_{current}} + \mathbf{w_{current,v}}$

if **new_distance** < $\mathbf{distance_v}$ do

$\mathbf{distance_v}$ = new_distance

$\mathbf{predecessor_v}$ = **current**

Find a node **m** in **V** where $\mathbf{final_m}$ is false and $\mathbf{distance_m}$ is minimal.

If $\mathbf{distance_m}$ equals ∞ there is no path from **s** to **t**.

Else do:

final$_m$ = true

current = **m**

When the algorithm finishes and the path from **s** to **t** exists, its length equals **distance$_t$** and the path itself can be retrieved from **predecessor$_v$** as shown below:

current = **t**

route = **empty**

Add **current** at the beginning of **route**

While **current** not equals **s** do:

current = **predecessor$_{current}$**

Add **current** at the beginning of **route**

And then the **route** is the shortest route from **s** to **t**.

Example (according to [Syslo et al, 83]):

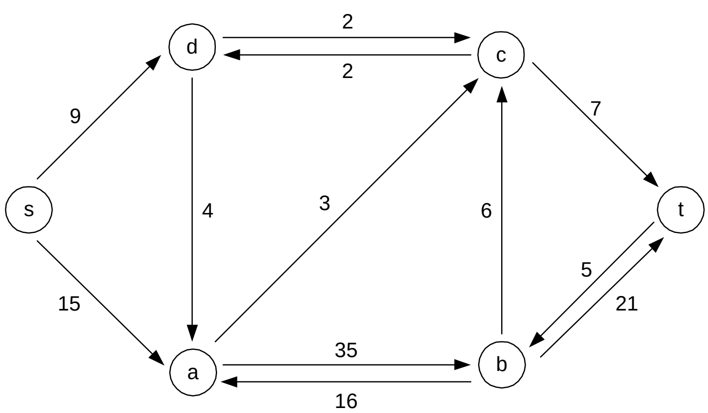


Figure 3.9 An example of directed graph

Now we will show a table which describes every step of Dijkstra’s algorithm for the graph shown in Figure 3.9.

	s	a	b	c	d	t
initialization	**0**	∞	∞	∞	∞	∞
1st iteration	**0** **0**	15 15	∞ ∞	∞ ∞	9 **9**	∞ ∞
2nd iteration	**0** **0**	13 13	∞ ∞	11 **11**	**9** **9**	∞ ∞
3rd iteration	**0** **0**	13 **13**	∞ ∞	**11** **11**	**9** **9**	18 18
4th iteration	**0** **0**	**13** **13**	**48** **48**	**11** **11**	**9** **9**	18 **18**

In the table we show a content of **distance**$_{v}$ and the value in bold face means it is a final one - **final**$_{v}$ is true.

3.5.2. Computational complexity considerations.

Let n equals the number of nodes in **V**. Then the initialization step's complexity results from iterations for each node in **V** and is O(n). In the iteration step the external loop is executed at most n-1 times. A lookup through successors of **current** and updating the **distance**$_v$ and **predecessor**$_v$ takes at most time proportional to n. To find the shortest distance in i^{th} iteration we have to execute at most n-i-1 times the comparison. So the iteration step's complexity is $O(n^2)$. Finally, the computational complexity of this algorithm equals $O(n^2)$.

Taking into consideration the fact that the graph of the urban network is a sparse one (the number of arcs is much less than n^2), the real time of computations can be decreased by using linked list of neighbours as a graph representation. An example of a linked list of neighbours for a graph from Figure 3.9 is shown below:

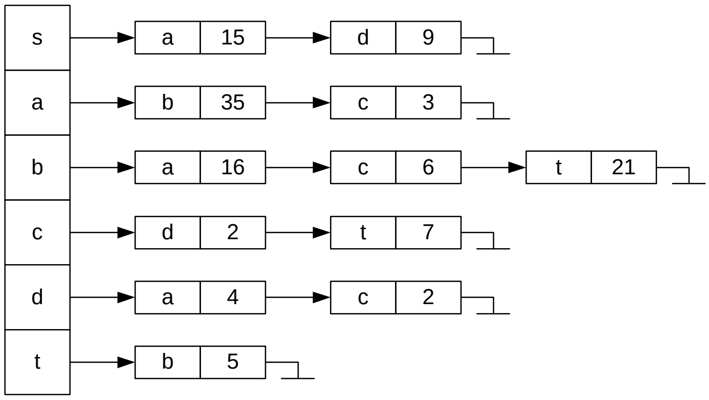


Figure 3.10 An example of linked list of neighbours

In Figure 3.10 there is a list associated with each node. The list contains node's neighbours and a weight (distance) between each of them and the node. In a case of urban network this representation of interconnections is very suitable, because each node is connected only with few others. For example one street is connected at intersection usually with at most three others. Bearing in mind that each street has to be modelled as two arcs (if it's a two-way one), on the list of neighbours we have at most four others – to turn right, go straight, turn left and turn back.

3.5.3. Spatial data model as an input data.

In [Spatial data model] we have described the urban network mainly using three relations – nodes, arcs and allowed moves. The use of the algorithm without allowed moves relation is obvious, but to run it taking into consideration which moves from arc to arc are allowed we have to transform graph G of arcs and nodes into graph G'. There are two possible transformations:

- Creating artificial nodes and adding arcs with travel times (weights) equal to zero;
- Changing allowed moves into arcs and arcs into nodes.

In the first transformation each node has to be divided into a number of new nodes equal a number of incoming and outgoing arcs. There are also new arcs between the nodes created in previous step. Their number equals the number of allowed moves between arcs adjacent to given node. For divided node **k** each new arc $\mathbf{arc_{k,i,j}}$ in **G'** connects new nodes $\mathbf{node_{k,i}}$ and $\mathbf{node_{k,j}}$ if there exists an allowed move between arc **i** and arc **j**, arc **i** ends in **k** node in **G** and arc **j** starts in **k** node in **G**. For example graph given in Figure 3.4 will be transformed to a following graph:

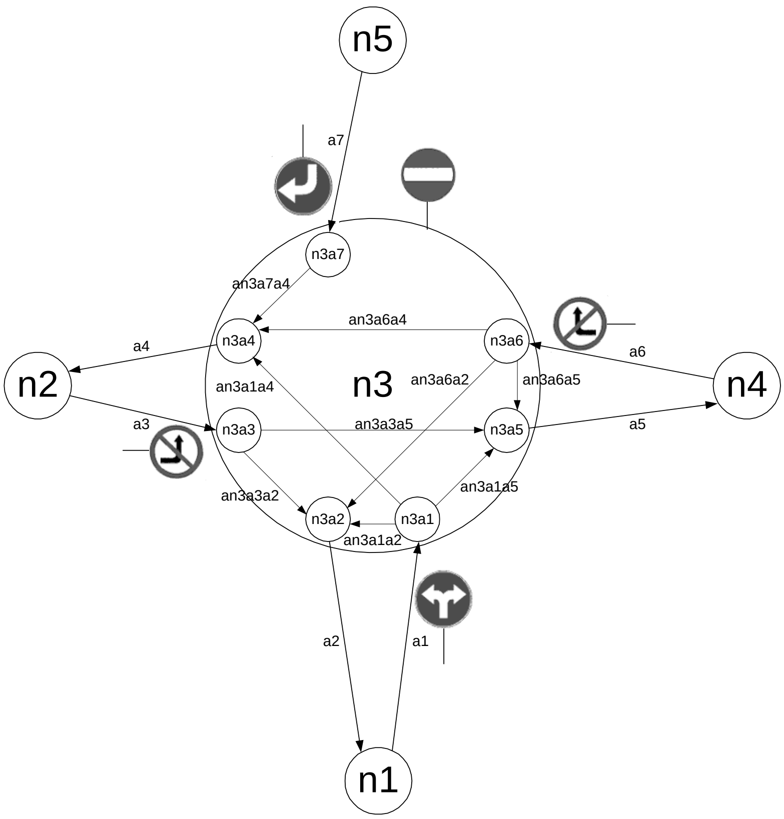


Figure 3.11 An example of allowed moves transformation no. 1

In the graph above the node n3 is transformed into seven nodes n3ax for each incoming and outgoing arc ax and into arcs an3axay for each allowed move from arc ax to arc ay. In this transformation weights on old arcs stay the same whereas weights on newly created arcs (from allowed moves) are equal to zero.

In the second transformation allowed moves are changed into arcs and arcs into nodes. Each arc **i** in **G** is transformed into node **i** in **G'**. If there is an allowed move from arc **i** to **j** in **G** an arc $\mathbf{allowedmove_{i,j}}$ is created in **G'**. For example graph given in Figure 3.4 will be transformed to a following graph:

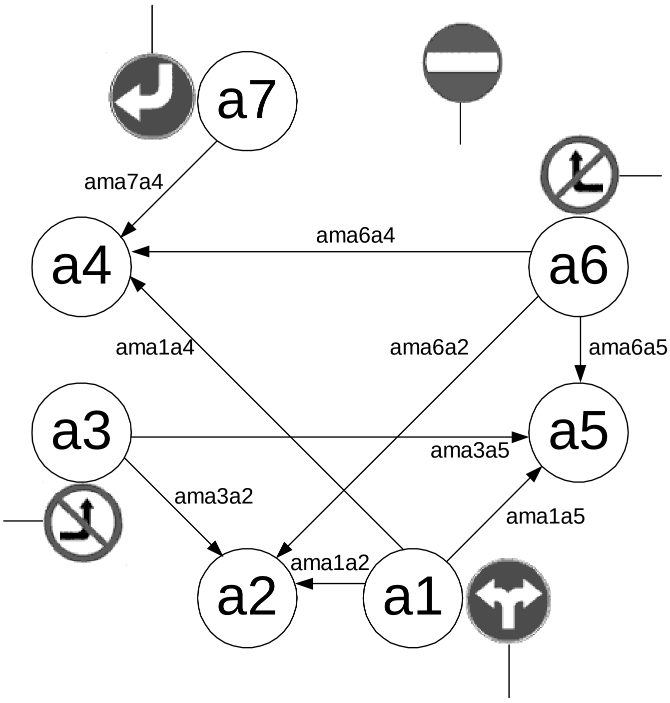


Figure 3.12 An example of allowed moves transformation no. 2

In the graph above each arc ax is transformed into node ax. And if there is an allowed move from arc ax to arc ay it is transformed into an arc amaxay from node ax to node ay.

In this transformation there is a problem where to store information about travel times (weights from graph G). The only suitable places are weights of allowed moves arcs. We can do it in two ways – store travel time of an arc from which the move is allowed or to which the move is allowed. At first glance it may seem as an inaccuracy issue. Now we will show that the inaccuracy is not very significant for long paths containing many arcs.

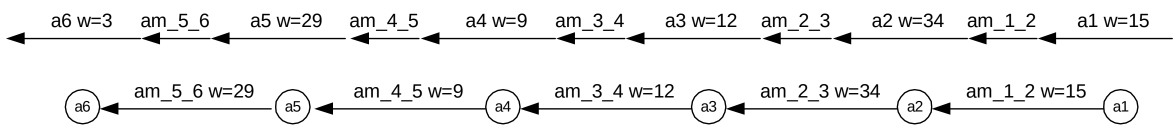


Figure 3.13 An example of weight assignment in allowed moves transformation no. 2

In the example above we have used the first way of weight assignment from arcs to allowed moves. We have to bear in mind that while computing the shortest path the only inaccuracy we will encounter is the omission of one weight of the last arc. But if we state the problem in a different way – find the shortest changes of arcs to get from beginning arc to the destination arc, which is still true in urban network; we will get the correct path.

3.5.4. Execution time increase after spatial data transformation

In the table below there are the number of arcs, nodes and allowed moves in a urban network graph for an example city having about 600 000 inhabitants.

Arcs	Nodes	Allowed moves
ca. 36 000	ca. 18 000	ca. 90 000

We can assume that the proportions will be similar for other cities of this size.

Now we will draw a few coefficients from those numbers:

- arcs per node = 36 / 18 = 2;
- allowed moves per node = 90 / 18 = 5;
- allowed moves per arc = 90 / 36 = 2.5.

In the first transformation if in the graph **G** there were **n** nodes then in the graph **G'** each node can be transformed into **six** new nodes on average.

new nodes count = old nodes count * allowed moves per node * allowed moves per arc / 2 = old nodes count * 6.25

Then the time of execution can increase up to $6.25^2 \approx 40$ times.

In the second transformation if in the graph **G** there were **n** nodes then in the graph **G'** each arc will be transformed into node.

new nodes count = old nodes count * arcs per node = old nodes count * 2

Then the time of execution can increase up to $2^2 = 4$ times.

Summing up the conclusions drawn we can see that the only reasonable transformation of urban network spatial data into a graph for shortest path algorithm is the second one. It is accurate enough for the application of the system described in the thesis.

3.6. Traffic data integration module

3.6.1. Acquiring traffic data from control centre

The traffic data in the traffic control centre system is available upon a request in a form of a log file. This file contains a list of measurements from traffic sensors placed mainly at intersections in various locations in a city. Each measurement is a number of cars that have passed the sensor in a given period of time. An example cutting of the file describing traffic situation at an example intersection is shown below:

```
COUNTING RESULTS INTERSECTION 0213
=============================

Counter group 0050
```

m-d	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26	11-26
h:m	18:00	18:15	18:30	18:45	19:00	19:15	19:30	19:45	20:00	20:15	20:30	20:45	21:00
FC02	135	127	120	138	103	123	114	91	102	127	123	129	
FC03	9	10	9	16	10	9	8	5	6	5	5	7	
FC04	7	9	16	9	12	9	8	8	5	10	7	5	
FC05	164	132	103	127	117	86	87	93	82	76	61	70	
FC07	95	96	90	67	88	55	56	59	49	39	51	32	
FC08	243	253	230	197	232	202	170	156	197	167	147	165	
FC11	58	54	56	54	24	31	31	21	25	19	21	10	
FC72	5	3	7	3	10	8	1	2	5	1	1	1	
FC81	1	3	6	3	2	1	4	4	1	1	0	0	
Tot.	717	687	637	614	598	524	479	439	472	445	416	419	

Figure 3.14 An example of a log file

In the example there is a situation for an intersection number 0213 on 26th of November between 18:00 and 21:00 in 15-minutes precision. Each column in the table represents a 15-minute long period of time whereas each row represents a data for given sensor. The sensors are marked with FCxx symbols. The sensors marked between FC01 and FC12 show how many cars have passed the intersection in a following manner:

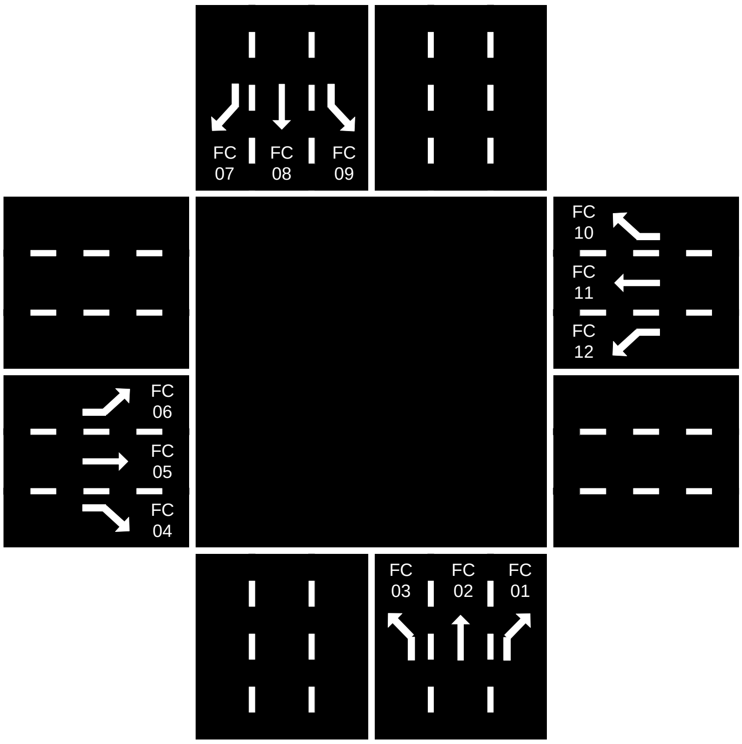


Figure 3.15 The assignment of sensor symbols at an intersection

For example the FC01 sensor shows how many vehicles has driven from south to east through the intersection, whereas FC02 sensor shows how many has driven to north and respectively FC03 to west. The information of the traffic incoming from the remaining drives onto the intersection is assigned in analogous way to the described. We assume that the measurements are already pre-processed to sum a number of cars on lanes in the same direction.

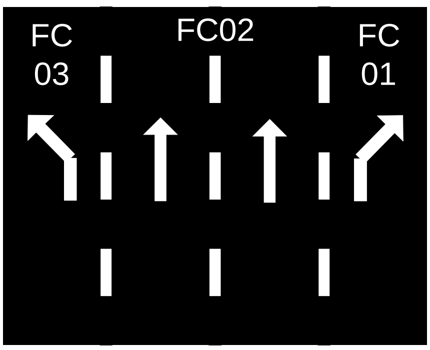


Figure 3.16 An example of lanes in the same direction

The problem is only with lanes from which a vehicle can go in more than one direction. Then statistical approach has to be used for example we have to assume that 60% of vehicles will go straight and 40% will turn right.

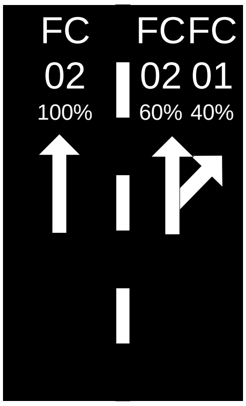


Figure 3.17 An example of a lane in two directions

The information from traffic control centre is in a pre-processed form and nothing can be done to acquire more accurate data. But for the purpose of computing shortest path in urban network the accuracy of the data is more than enough.

The sensors using numbers above 12 describe traffic on tram rails and so they do not contain any important information for the designed system.

Since the information containing the traffic data is saved in textual form into the file an appropriate parser has been designed and is a part of this module. The parser reads the log file and transforms each measurement into a form of a message. The message structure is described below:

date	time	timespan	intersection	sensor	value

Figure 3.18 The traffic situation message

The fields in this message have the following meaning:

- date – the date of the measurement
- time – the time of a start of the measurement
- timespan – the time period between a start and an end of the measurement (currently fixed to 15 minutes)
- intersection – an identifier of the intersection the measurement has taken a place
- sensor – an identifier of the sensor the data come from
- value – a value of this measurement, a number of vehicles in a period of time

To give an example of the traffic message we will create one for the first row and first column from Figure 3.14.

date	time	timespan	intersection	sensor	value
2002-11-26	18:00:00	0:15:00	0213	FC02	135

Figure 3.19 An example of a traffic situation message

Traffic information in the message form designed above is more useful to handle in message-driven systems and to store in database for processing purposes. The form is universal enough that any changes in the measurement timespan can be done anytime.

3.6.2. Communication with the centre

The connection between the central unit and traffic control centre is made through and encrypted connection over a serial link since that is the requirement of the centre security policy.

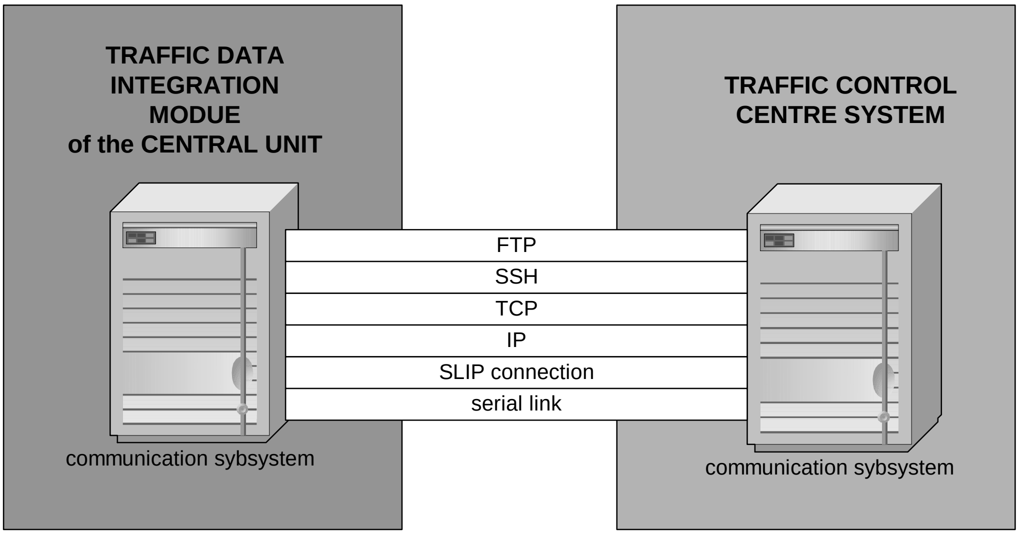


Figure 3.20 The two communication subsystem and their interconnection

On the required serial link a SLIP (Serial Line Internet Protocol) has been set and the IP and TCP protocols are made available. Then the subsystem's applications can create a SSH connection and FTP connection tunnelled by the first one.

3.6.2.1. The communication subsystem on the centre side

The subsystem is responsible for querying the traffic control centre system to produce log files and when a new log file is created to send the traffic data to the central unit side. Since it requires an access to a log file directory appropriate access rights have to be granted for communication subsystem process. Since the system in the centre is a UNIX-like one, only a simple administrative operation is required. Then the subsystem connects to its counterpart on the central unit side and establishes an encrypted connection. Since the operation between those modules is a simple file transfer a standard FTP-like protocol is used but it is tunnelled through an encrypted SSH connection. When the file is successfully transmitted the connection is closed and the subsystem queries and waits for next file. The connection through SSH is secured enough not to be tapped. A private key for the SSH connection is stored by the subsystem in a file. The file is placed on a storage is encrypted to prevent falling into the wrong hands.

3.6.2.2. The communication subsystem on the central unit side

In the traffic data communication subsystem on the central unit side an SSH server is operating by listening for connection from its counterpart on the traffic control centre side. When the connection is established and the other side finishes sending a log file the connection is closed. Then the file is stored in an archive for auditing purposes. The file is parsed and the traffic information is transformed by the parser described in 0. The information transformed into messages is sent to traffic data integration server to be processed.

3.6.2.3. Robustness and efficiency issues

The connection is reliable in a way that if on of the layers (serial link, SLIP, IP, TCP, SSH or FTP) will fail the subsystem on the centre side will restart the communication (log file transfer) and only after successful ending of communication its counterpart on the central unit side it will query the traffic control centre system for new log files. Since the transferred data is in a form of a textual file the transfer time on a dedicated serial link is better than required. A file containing information about one intersection with ten sensors takes approx. 3kB long. The time to send 3kB of data on standard serial link at 115200 baud rate takes approx. 0.3 s. We have to bear into mind that the SSH protocol allows to compress the data transferred and that textual files usually compress the best. Then we can see that in a time period of 15 minutes information about more than 3000 of intersections can be sent which is more than enough for a city of 1 000 000 citizens. If the parsing took place on the centre side the communication issues would not be the case. But since there is an assumption that all the resource demanding operations are performed on the central unit side and on the centre side there is only file transfer software this is the best solution for the case.

3.6.3. Traffic data transformation

The main task of the traffic data integration server is to update the times of travel for each arc in urban traffic database. A time of travel on a section is defined as follows:

$$\textit{time of travel [s]} = \frac{3.6 * length\_of\_a\_section[m]}{coef_1 - e^{coef_2 * number\_of\_cars\_per\_hour}}$$

if time of travel < 0 then time of travel = MAX_VALUE (e.g. 3600 s)

In the formula above the coefficients $coef_1$ and $coef_2$ depend strongly on a type of the road (as defined in data model in 3.4.

Road type	$coef_1$	$coef_2$
1	71	0.00124
2	61	0.00404
3	51	0.00382
4	21	0.00924
5	71	0.00924

Figure 3.21 Coefficients assignment for various road types

The coefficients where selected based upon a statistical observation of traffic in urban network. If we will use the coefficient for an example road which length equals 10 000 metres we can see the following chart of travel times in function of cars passing the road in a period of one hour.

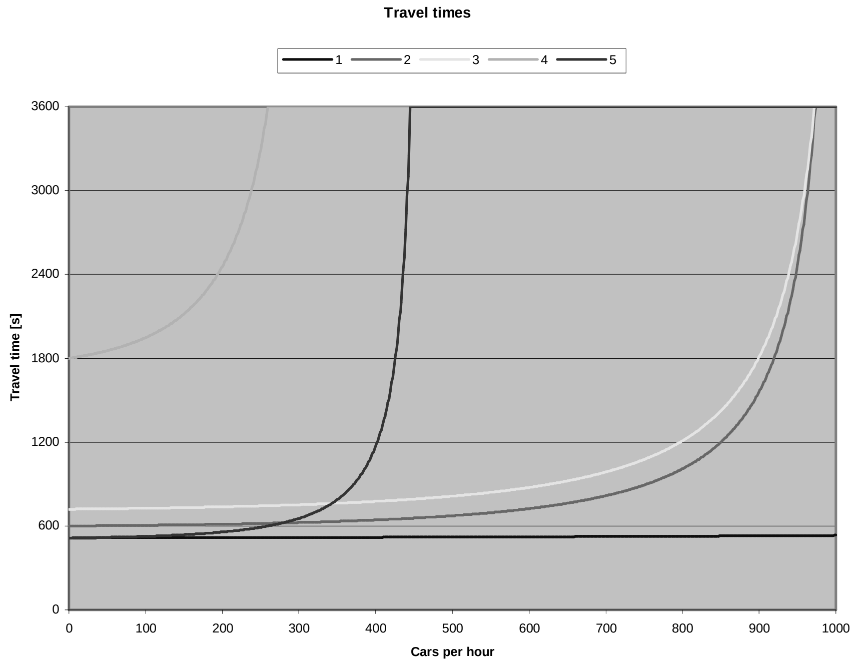


Figure 3.22 Travel time in function of cars passing per hour at a 10 km road

Now what we have to do is only to sum all the outgoing flows from an intersection, analogous to the First Kirhoff Rule, to be able to update the travel time for an outgoing arc. This operation is done by the traffic data integration server each time a traffic message will come from the communication subsystem.

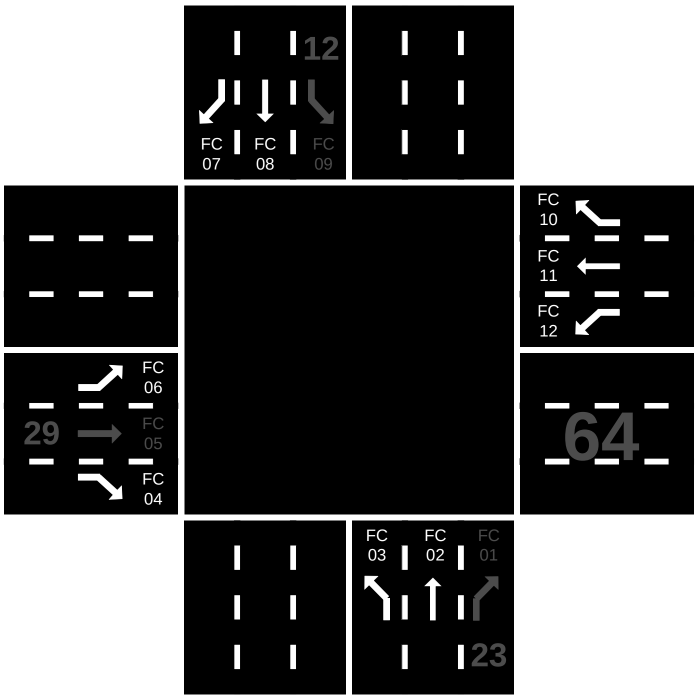


Figure 3.23 Outgoing flows summing example

In the example above within an hour 12 vehicles have crossed the FC09 sensor, 29 the FC05 sensor and 23 the FC01 sensor, so 64 vehicles have left the intersection using the eastern exit. If we know the length and a road type of the arc outgoing to the east we can now calculate its travel time. Assuming that the arc is 100 m long and the road it represents is of type 1 the travel time equals

$$\frac{3.6*100}{71-e^{0.00124*64}} = 5{,}148932 \text{ s.}$$

In such way all arcs in traffic database can be updated.

3.7. Mobile unit services module

Since the most important tasks of the central unit are connected with mobile unit a module handling mobile unit service has been separated.

3.7.1. Handling connection with mobile unit

Each mobile unit is able to connect to the central unit using a standard TCP/IP connection used in the Internet. In most cases the Internet will be the medium that will occur between those units. It seems that it does not really matter if there is a GSM/GPRS network as an underlying layer. But it really does. Not only is the throughput and burst nature of the GPRS communication a concern. There are also accounting issues of the GSM network operator. We have to bear in mind that the most frequent case is that its worth not to disconnect the session between a mobile unit and the central unit since it’s very probable that a new request will be made after one has been completed. Because a payment in such networks is based on a gross data transferred including the headers of all the protocols at various network layers, a permanent connection seems to be the most cost-efficient one. Also a use of UDP protocol is concerned as a better solution for short messages e.g. an update of traffic information for one arc whereas a use of TCP protocol is justified in situations entailing a bulk transmission of data e.g. a computed path or an update of a city map. The unified session token is the best solution for handling a session between a mobile unit and a central unit.

session ID	client ID	issued	cryptographic signature

Figure 3.24 The structure of a session token

The structure of a session token is as follows:

- session ID – a unique identifier of the session used to determine a session in subsequent request in a session;
- client ID – a unique identifier of a mobile unit;
- issued – a date of an issue of the session token;
- cryptographic signature – used to proof the authenticity of the token.

The exchange of messages shown below describes the process of acquisition, usage and invalidation of the token.

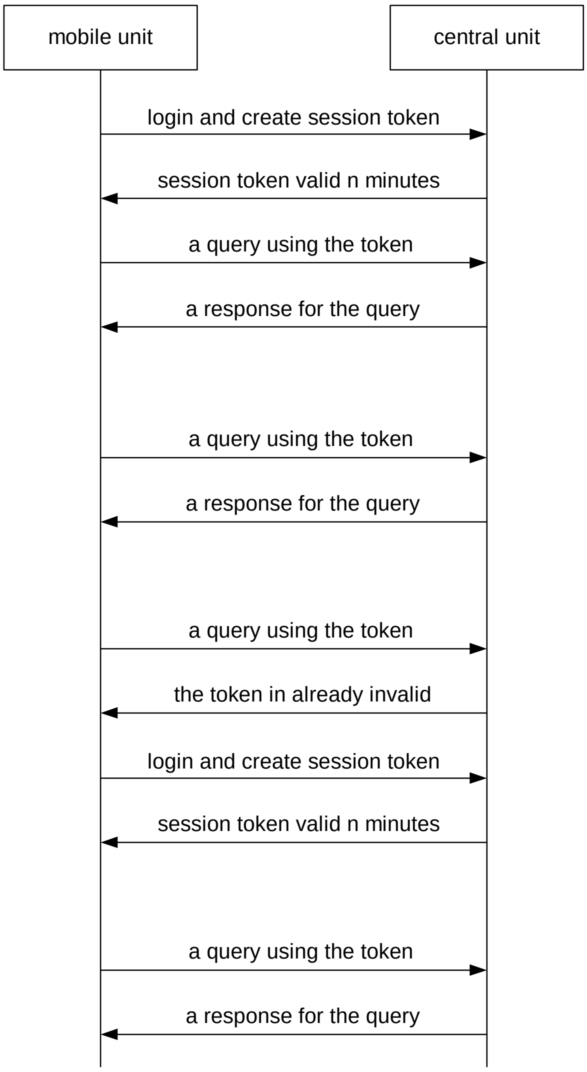


Figure 3.25 A token usage scenario

As we can see from the figure above when token has lost its validity a new one has to be acquired using all the credentials needed for logging authentication purposes.

Since the communication between a mobile unit and the central unit is based on a client-server model, a client (the mobile unit) has to connect to the server (the central unit) each time it wants to create a connection. For this reason a central unit has to listen for incoming connection on the IP network layer. For performance purposes a pool of connections is created in the server and a load balancing issues has been considered. In an advanced configuration a special network routing techniques might be used to balance the load of incoming connections and divide it in a farm of servers.

Since the communication between mobile units and the central unit is asymmetric in the volume of the data sent through the connection link, more specific protocols available in GSM network can be used. For example many GSM operators rate WAP protocol services in a more preference way to the data transmission services which can be a cheaper option. In some cases a subscription for less than 3€ per month is the only cost. On the mobile unit side a simple dummy WAP browser has to be implemented whereas on the central unit side a standard HTTP server handling the requests.

3.7.2. Authorization and encryption of the connection

A session connection is authorized using a scrapbook method. The method is an extension of the connection token idea described above. Here each client stores a private key to decrypt a bunch of session tokens which is called a scrapbook. Each time a client log into the server and the server accepts the credentials of the client a scrapbook is created, encrypted and sent to the client side.

SCRAPBOOK

1th SCRAP	
2nd SCRAP	
3rd SCRAP	
	n-1 th SCRAP
	n th SCRAP

SCRAP

KEY (16B)	VALUE (16B)

Figure 3.26 A scrapbook structure

In the figure above we can see that a scrapbook is a list of scraps. A scrap is a pair of key and value numbers. The whole content of the scrapbook is generated randomly on the server side, one copy is stored there and one is sent via the connection to the mobile unit as a request for a successful login. Each time a client wants to send a request it uses a scrap to sign the request and next scrap to decode a response. Since the server has the same copy of a scrapbook it can use it in a similar way.

In the option of using WAP protocol as an underlying layer the whole request can be encoded in a HTTP request in a form of a hexadecimal string. The hexadecimal string is an uppercase textual representation of binary data. For example a hexadecimal data 0x021f153 is represented as a ‘021F153’ string. In a form of HTTP request it would look like:

http://wap.polaris2.pl/request?r=021F153

The response in WAP protocol can be encapsulated in a dummy resource e.g. a wireless bitmap (WBMP). We have to bear in mind that the sizes of a WAP request and response are usually limited by the GSM operator and in most cases equals adequately 256 bytes and 2 kilobytes. Those two values can be considered as WAP request MTU (maximal transfer unit) and WAP response MTU.

In the example of using the WAP protocol as the underlying layer for communication with mobile units the following schematic communication layer architecture is used:

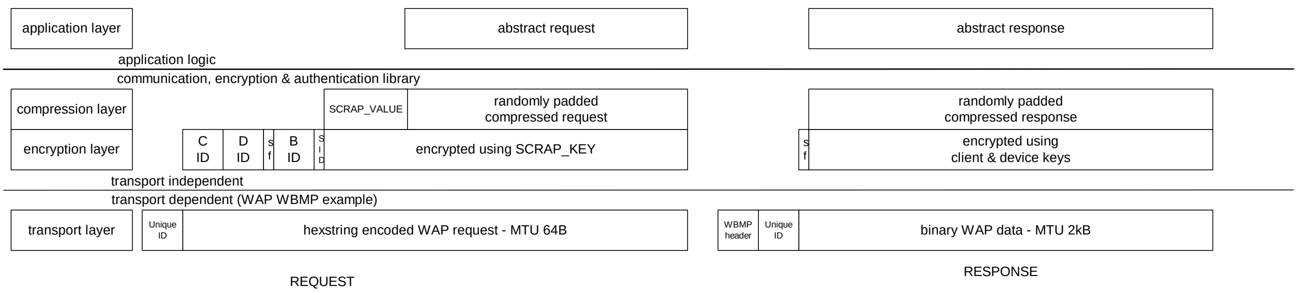


Figure 3.27 Mobile unit communication layers

Description :

- sf – status flags (1B);
- CID – client identifier (4B);
- DID – device identifier (4B);
- SID – scrap identifier (1B);
- BID – scrapbook identifier (4B);
- Unique ID – request unique identifier for given client and device, e.g. a timestamp of a request (4B).

The status flags byte is described in a table below:

bit	meaning
0^{th}	SCRAPBOOK STALE - 0-all OK, 1 in REQ - request for new scrapbook, 1 in RESP - indicates invalid request
1^{st}	COMPRESSION - 0-off, 1-on
2^{nd}	Reserved
3^{rd}	Reserved
4^{th}	Reserved
5^{th}	Reserved
6^{th}	Reserved
7^{th}	Reserved

As we can see in the figure above there is also a compression option in the protocol. The compression can be particularly useful for sending textual data since this kind of data can be compressed most effectively. In case of not using other protocols e.g. TCP a maximal transfer unit should be omitted and the architecture of the communication layers would be still valid.

The client is identified by a client identifier and a device identifier. The device identifier is stored securely into a mobile unit device and is fixed during manufacturing process. The client identifier is stored on an identification card which the driver of a vehicle inserts into a special reader in the mobile device. Also the private key used for the session encryption is divided into two parts: the user part and the device part. The key used to decrypt the scrapbook obtained from a central unit is a concatenation of its two parts. The user part is also stored on a driver's card but it is also protected with a pin code the driver must enter before starting the mobile unit system.

3.7.3. A lookup of a destination object

To obtain a location of a destination object the mobile unit queries the central unit which then looks up the object database which is stored in more general – traffic database. In a query a vehicle location is also stored for the purpose of sorting the distance to each object meeting the conditions of the query. There are various types of a destination object query:

- an exact query – the name and type of the object is given

vehicle longitude	vehicle latitude	object name	object type

Example:

vehicle longitude	vehicle latitude	object name	object type
17 12.230'E	52 49.030'N	Reiffeisen Bank	BANK

- a partial name query – the name of the object is given in a partial form

vehicle longitude	vehicle latitude	partial name

Example:

vehicle longitude	vehicle latitude	partial name
17 12.230'E	52 49.030'N	Reiffe...

- a type query – only the type of the object is given

vehicle longitude	vehicle latitude	object type

Example:

vehicle longitude	vehicle latitude	object type
17 12.230'E	52 49.030'N	BANK

As a result of the query a list of objects meeting its conditions is returned. In the worst case it is an empty list if there is no such an object that would satisfy the query. The list is ordered by a distance from a location of the vehicle to a location of the object. A content of an item of the list is described below:

object longitude	object latitude	object identifier	object name	object type	arc identifier	distance

Example:

object longitude	object latitude	object identifier	object name	object type	arc identifier	distance
17 13.120'E	52 50.010'N	32561	Reiffeisen Bank	BANK	52743	32 m
17 11.070'E	52 49.540'N	3321	WBK Bank	BANK	87213	728 m

After retrieving the list of object the mobile unit can update its traffic database containing objects information if necessary and query the central unit to compute the shortest path from its location to selected destination.

3.7.4. Shortest path service

The service is the most obvious in its assumptions of all the services in the module. The mobile unit sends a request to the central unit to compute the shortest path from its location to a location of a chosen object. More precisely it requests to compute the path from starting arc to ending arc using the information of current traffic situation. The request for shortest path looks as shown below:

from arc	to arc

Example:

from arc	to arc
32776	98127

Then the mobile unit services server using the navigation algorithm described in 3.5 and traffic information contained in traffic database computes the shortest path from the source arc to the destination arc and returns the result to the mobile unit. The result is in a form of a list as described below:

arc identifier	time of travel	TTL

Example:

arc identifier	time of travel	TTL
3872	16 seconds	15 minutes
3873	23 seconds	15 minutes
3877	9 seconds	30 minutes

The list is ordered and each item shows one step on the path from source to destination. The TTL information is so called "time-to-live" and is used to signify the time span the "time of travel" information is valid. The information can be used by the mobile unit also to update its internal traffic database. When the traffic information in its internal database is no longer valid it will query the central unit for an updated traffic data.

3.7.5. An update of current traffic situation data

The mobile unit can always query for a current traffic data. The query is realized by sending a query message for each arc in the traffic database. The form of the query is very simple:

arc id

Example:

arc id
32776

A response for such a query is as follows:

time of travel	TTL

Example:

time of travel	TTL
16 seconds	15 minutes
23 seconds	15 minutes
9 seconds	30 minutes

And here the TTL means the same „time-to-live” as in 3.6.4.

Concerning the cost-effectiveness issues a bunch queries are also handled by the module. They are in form of a list of arc identifiers and the response has the same form as the response shown in 3.6.4. The last types of traffic information update request are range ones:

- Distance from a point range – the mobile unit requests the traffic information of arcs in a circle surrounding a point (circle diameter, point latitude and longitude are given);
- Latitude and longitude range – the mobile unit requests the traffic information of arc between minimal and maximal latitude and between minimal and maximal longitude.

As a result of the request a response having the same form as shown in 3.6.4 is sent back to the mobile unit.

3.7.6. The urban network map update

The mobile unit can query for a map update. It can be useful if a vehicle having no map preloaded login into system and queries for a path or an object. The request has the form of a range request described previously. The response is a list of nodes, arcs, objects, allowed moves and names matching the range criterion. The items transferred to the client have the form identical to shown in spatial data model in Figure 3.7. The map update can be done periodically based on user preferences or on a request of the mobile unit (e.g. a path that leads through an arc that is not in local traffic database).

3.7.7. Mobile unit location supervision service.

There is a service responsible for supervising a location of a mobile unit. It could be used for security purposes i.e. guarding a car against a theft or for logistics purposes e.g. a flotilla management. If the service is available to a given vehicle its mobile unit has to report its position periodically. Then it is stored in the central unit and can be acquired via the management module.

3.7.8. Mobile unit database

Each mobile unit data is stored in a mobile unit database. The database contains the private keys for each client (e.g. a driver) and each device (e.g. a mobile unit itself). Also a current scrapbook for authenticating and encryption purposes is stored there. Also a history of vehicle locations and mobile unit queries can be retrieved from the database.

3.7.9. Accounting issues

For accounting purposes the data from the mobile unit database can be used. The service is responsible for generation of various reports and to cooperate with automatic accounting systems. Accounting can be done for a driver, a vehicle, a group of drivers or a group of vehicles to show the efficiency and usage of the system.

3.8. Management module

The module is responsible for controlling system parameters and supervising the operation of the traffic data integration module and the mobile unit services module. Also a various types of reports are produced by the module.

3.8.1. Management server

The management server is interconnected to the two controlled modules – to the mobile unit services module and to the traffic data integration module. It also enables the connection with management consoles which are used by the managing operators.

3.8.2. Management consoles

The management consoles connect to the management server in various ways. In all cases a standard TCP/IP protocol is used.

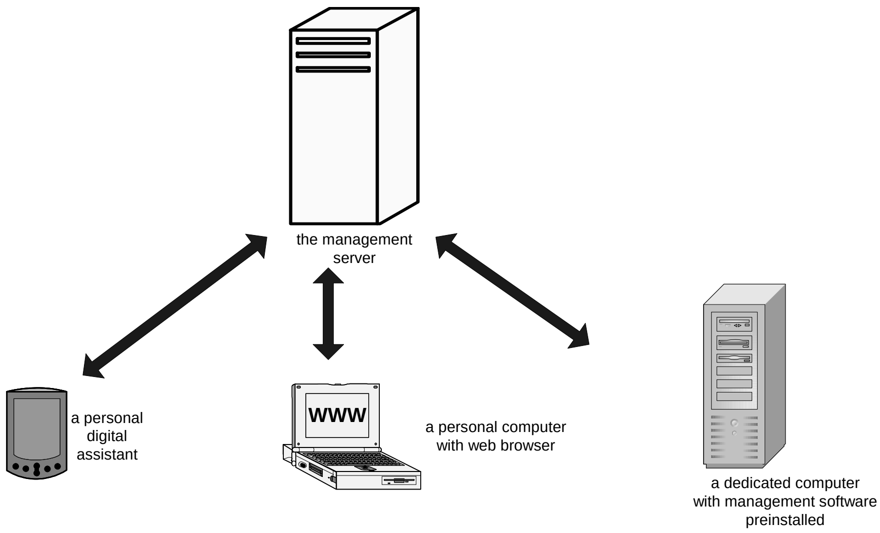


Figure 3.28 An interconnection of management consoles

3.8.2.1. Application-based consoles

This is an application installed on a desktop computer. This kind of console is the most powerful and flexible one since the desktop computer's abilities are usually more than enough for reporting and management purposes.

3.8.2.2. PDA-based consoles

This is an option where personal digital assistant (PDA) equipment acts as a mobile management console. It was designed to increase the mobility of the management operators. The option is the most mobile guided, but since the processing possibilities in PDA devices are limited, only a part of management functionality can be provided.

3.8.2.3. Web-based consoles

This option is a balance between functionality and mobility. Using web-based consoles the management functionality is available anywhere where there is a network access and a web browser. An operator using web browser connects to the management server which acts as a web server. Then the operator is authenticated and requests and responses between a web browser and the web server are encrypted through secured connection e.g. using SSL (Secure Socket Layer) protocol.

4. Implementation issues

4.1. Central unit software implementation

Most of the central unit software has been implemented in Java language to assure the portability of the central unit. Thanks to the requirement stated above the core of the system may run on almost every operating system (e.g. Solaris Operating Environment, Linux, HP-UX, AIX, Tru64 Unix, Windows, MacOS X) and hardware platforms (both CSIC and RISC architectures). As the core of the central unit an EJB (Enterprise Java Beans) application has been chosen to run on an application server. Despite the fact that there are many application servers (as listed in 2.5, the JBoss application server has been chosen. The main reason is not only that it is an open source product but also because of its modular architecture. This helps to tailor the server to contain only the necessary modules and to save valuable resources.

The remaining software which resides on the traffic control centre side of the traffic data integration module has to be written natively since the operating system of the traffic control centre system is predefined and there is a strong requirement to use as little system resources as possible. The software described has been written in C++ language and compiled using standard C/C++ compiler provided for the platform.

There is also visualization software written for an embedded Microsoft Pocket PC platform for PDA-based management console. It is written in C++ language using MFC (Microsoft Foundation Classes) library for visual components. The software also uses some low level Windows CE functions for communication purposes.

4.2. Database implementation

There are many databases systems available nowadays. The most popular and reliable ones are, among the others, Oracle Database 10g, IBM DB2, Microsoft SQL Server 2000 or an open source PostgreSQL 7.3. The last one has been chosen as the database for our system. Since the PostgreSQL database system is portable to many operating systems, the same database can be applied to the central unit as well as to the mobile unit to store the traffic information. This is the most efficient well known open source database and this is the main reason it has been chosen.

The model of spatial data shown in Figure 3.8 has been implemented in following database tables:

- An “arc” relation is implemented in “arc” table

```
                      Table "arc"
     Column         |  Type   | Modifiers
--------------------+---------+-----------
 id                 | integer | not null
 course             | integer |
 time               | integer |
 length             | integer |
 level              | integer |
 type               | integer |
 beginning_node_id  | integer |
 beginning_node_lat | integer |
 beginning_node_lon | integer |
 ending_node_id     | integer |
 ending_node_lat    | integer |
 ending_node_lon    | integer |
 twin_node_id       | integer |
Indexes: arc_beginning_node_id_idx,
         arc_beginning_node_lat_idx,
         arc_beginning_node_lon_idx,
         arc_course_idx,
         arc_ending_node_id_idx,
         arc_ending_node_lat_idx,
         arc_ending_node_lon_idx,
         arc_id_idx,
         arc_name_idx
Primary key: arc_pkey
```

- A “node” relation is implemented in “node” table

```
        Table "node"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer | not null
 lat    | integer |
 lon    | integer |
 level  | integer |
 type   | integer |
Indexes: node_id_idx,
         node_lat_idx,
         node_lon_idx
Primary key: node_pkey
```

- An “allowed move” relation is implemented in “allowed move” table

```
     Table "allowed_move"
  Column  |  Type   | Modifiers
----------+---------+-----------
 from_arc | integer | not null
 to_arc   | integer | not null
Indexes: allowed_move_from_arc_idx,
         allowed_move_to_arc_idx
Primary key: allowed_move_pkey
```

- A “name” relation is implemented in “name” table

```
                Table "name"
 Column |          Type          | Modifiers
--------+------------------------+-----------
 id     | integer                | not null
 name   | character varying(100) |
Indexes: name_id,
         name_name
Primary key: name_pkey
```

- An “object” relation is implemented in “object” table

```
                Table "object"
 Column   |          Type          | Modifiers
----------+------------------------+-----------
 id       | integer                | not null
 name     | character varying(100) |
 distance | integer                |
 type     | integer                |
 arc_id   | integer                |
Indexes: object_id,
         object_name
         object_arc_id
Primary key: object_pkey
```

As we can see from definitions of the tables shown above various indexes have been created mainly to speedup the join and search operations. The indexes are B-Tree based in all cases.

5. System testing

The testing phase consists of three test cases:

- traffic information data transformation;
- shortest path computation;
- correctness and efficiency of a communication link with PDA management console (visual operations).

5.1. Traffic information data transformation test case

Given:

- measurements from traffic sensors FC01 = 3, FC05 = 7 and FC09 = 13 done within 15 minutes time span;
- a eastwardly outgoing arc on a road type = 3 and length 153 m.

Calculating the travel time for the arc – execution screen:

```
FC01 – 3
FC02 – n/a
FC03 – n/a
FC04 – n/a
FC05 – 7
FC06 – n/a
FC07 – n/a
FC08 – n/a
FC09 – 13
FC10 – n/a
FC11 – n/a
FC12 – n/a

Number of cars per hour
- south – n/a
- west  – n/a
- north – n/a
- east  - (3 + 7 + 13) * 4 = 92

The arc id is 3122 type 3 length 153
Coef1 equals 51
Coef2 equals 0.00382
11.10956755 seconds
```

The result obtained by the software equals the theoretical value. The module works correctly.

5.2. Shortest path computation test case

Given:

- the map of Poznań city in a spatial database;
- the starting street – Zbąszyńska;
- the ending street – Szamarzewskiego.

A shortest path algorithm execution screen:

```
Database connected
36454 rows selected
adding names in progress:
names added

Database connected
84506 rows selected
adding arcs in progress:
arcs added

finding shortest path from 32064[19409] to 27345[8668] in progress:
shortest path found - length 88
Zbąszyńska[32064] Marcelińska[12678] Wolsztyńska[31045] Opalenicka[15939] Opalenicka[15941]
Szamotulska[27394] Szamotulska[27396] Szamotulska[27398] [34191] Grodziska[6400]
Szamarzewskiego[27345]
```

The algorithm has found a following path:

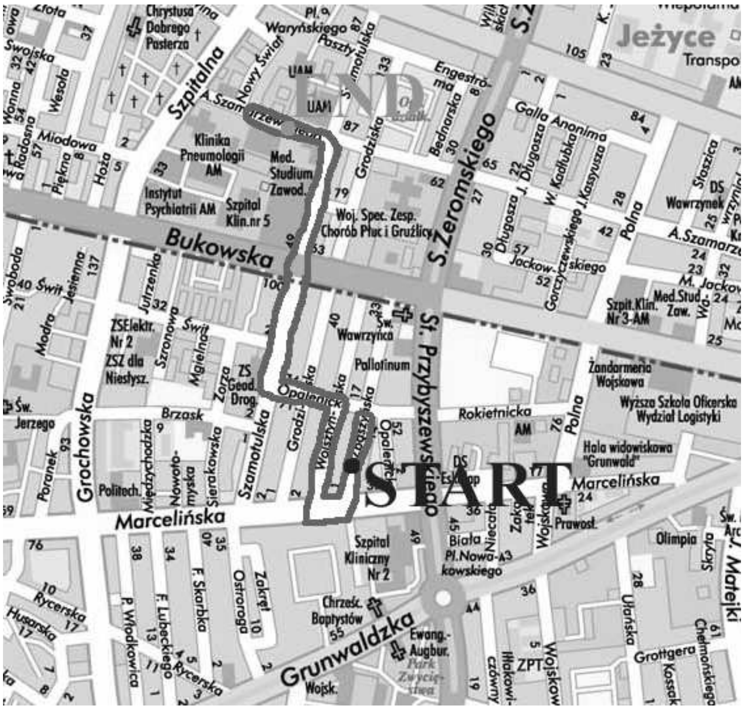


Figure 5.1 A navigation algorithm test case

As we can see the path is correct (Zbąszyńska and Wolsztyńska are one way streets).

5.3. Communication link test case

Given:

- the PDA as a visual management console;
- the management server as a source of a graphics displayed at the PDA screen;
- an interconnection between PDA and management server.

The test case checks a correctness and efficiency of visual operations on PDA management console. The test will display a map and a location of a tracked vehicle (mobile unit) and textual information sent to the central unit.

Figure 5.2 A graphical PDA operation test case

As we can see in the figure above on the display the map is displayed correctly and also textual information is presented. The image transfer and display time was measured and is given in a chart below:

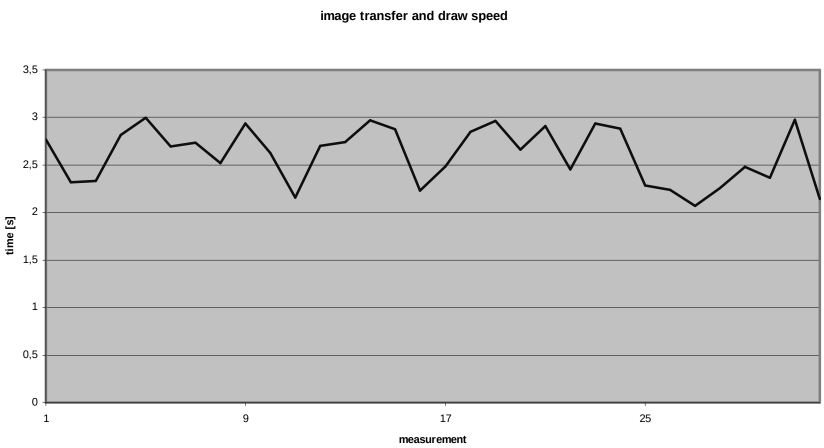


Figure 5.3 Image transfer and draw speed

As we can see from the figure above the time of whole the operation oscillates around 2.5 seconds per draw. This is enough for management and auditing purposes.

6. Summary

The main aim of this thesis was to design an integrated system solving many problems connected with traffic in today urban networks. The system, which consists of a mobile unit in a vehicle and a central unit, has been designed and appropriate prototypes of its modules have been created and their functionality tested.

In this thesis we have been focused mainly on the central unit and its connection with mobile unit. In the central unit at first an overall architecture has been designed. After an in-depth analysis of data available from traffic control centre an appropriate module for parsing data files and sending it through an encrypted connection has been designed, implemented and tested in detail. A research into sophisticated transformation of traffic data from its raw form into a suitable one for a traffic information database format has been conducted. It resulted in two ways. The first one is the transformation method itself. The second one is a design of a database for spatial data containing traffic information for urban networks. Many issues connected with the database design and implementation have been considered, especially efficiency requirements. This resulted in various transformations of the database schema, which is described step-by-step in the thesis. The shortest-path algorithm has been introduced as a mean to calculate the most efficient path in urban network. A transformation of urban network data to act as an input of the algorithm has also been described. A wireless connection using the GSM/GPRS network with mobile units in client's vehicles has been shown taking into consideration various aspects of encryption, session management and client services. Finally a management module of the system has been described and the results from its testing case have been presented.

The thesis introduces a system of promise which is able to help the drivers in their every-day life. It will make the inconvenience of traffic jams less harmful and will raise an overall efficiency of communication in urban networks. To lend wings to the system a closer integration with business databases should be proposed as a way of looking up the information about various objects fulfilling sophisticated criteria. Also a tight integration with logistics systems could add value especially when a fleet of trucks is concerned. There are many other applications that can be used with the system described in the thesis.

We believe that the system described in this thesis is worth to be developed much further beyond the design described. It is a new approach to communication in urban networks and it can leave its valuable stamp on our every-day life.

7. Bibliography

[Cormen et al., 94] T. H. Cormen, C.E. Leiseron, R. L. Rivest, *Introduction to Algorithms*, MIT, 1994. ISBN 0-262-530-910

[French, 96] Gregory T. French, *Understanding the GPS. An Introduction to the Global Positioning System*, GeoResearch, Inc., 1996. ISBN 0-9655723-0-7

[Hofmann-Wellenhof et al., 01] B. Hofmann-Wellenhof, H. Lichtenegger, J. Collins, *Global Positioning System: Theory and Practice*, Springer-Verlag, 2001. ISBN: 3-211-83534-2

[Mouly and Pautet, 92] Mouly and Pautet, *The GSM System for Mobile communications,* Cell & Sys., 1992. ISBN 2-9507190-0-7

[Pużak, 03] T. Pużak, *System nawigacyjny dla pojazdu z radiową aktualizacją danych o ruchu*, praca magisterska niepublikowana, Politechnika Poznańska, Poznań, 2003

[Roman et al., 02] E. Roman, S. Ambler, T. Jewell, *Mastering Enterprise Java Beans*, John Wiley & Sons, Inc., 2002. ISBN 0-471-41711-4

[Redl at al., 95] Redl, Weber and Oliphant, *An introduction to GSM*, Artech House, 1995. ISBN 0-89006-785-6.

[Rumbaugh et al, 91] J. R. Rumbaugh, M. R. Blaha, W. Lorensen, F. Eddy, W. Premerlani, *Object-Oriented Modeling and Design*, Prentice-Hall, 1991

[Simon and Walczyk, 02] Simon and Walczyk, *Sieci komórkowe GSM/GPRS: usługi i bezpieczeństwo*, Sylab, 2002. ISBN 83-917724-0-1

[Sobczak, 00] M.Sobczak, *Positioning of moving vehicle in urban networks - method and platform*, No. 2/2000, pages 115-126, Annual of Navigation Polish Academy of Sciences, Polish Academy of Sciences, Gdynia, 2000

[Syslo et al, 83] M. M. Syslo, N. Deo, J. S. Kowalik, *Discrete optimization algorithms with Pascal programs*, Prentice-Hall, 1983. ISBN 0-13-215509-5

[Ullman et al, 02] H. Garcia-Molina, J. D. Ullman, J. D. Widom, *Database Systems: The Complete Book*, Prentice Hall, 2002

Table of contents