VitaliiKlimenko

WEB CONTROL SYSTEM OF MOBILE PLATFORM

Part of the "Roboteh" project

Bachelor's Thesis
Information Technology

May 2012

MIKKELIN AMMATTIKORKEAKOULU
Mikkeli University of Applied Sciences

DESCRIPTION

MIKKELIN AMMATTIKORKEAKOULU Mikkeli University of Applied Sciences	**Date of the bachelor's thesis** May 2012
Author(s) VitaliiKlimenko	**Degree programme and option** Information Technology

Name of the bachelor's thesis

Web Control System of mobile platform

Abstract

The final thesis corresponds a development of web control system through which anybody can control the mobile robo-platform by using common device with a web browser and Internet connection. The system gives good mechanisms and tools for easy controlling of the mobile platform such as car and camera control elements, video transmission from camera, GPS navigation and mapping, visualization of environment around the car. The system has handy, flexible and configurable user interface. Itis fast and reliable. The main technologies which were used are JavaScript, JSP, Ajax, JSON, Java Applet, HTML, CSS, Google maps.

The results of the final thesis show the system can be used for normal control of mobile platform from everywhere through Internet. It works very fast with minimum delays about 20-30 milliseconds from user action to car reaction and video delay about 100-200 milliseconds. It is possible to reconfigure the system for any kind of robo-platform, improve it and implement addition functionality or used the same groundwork for realizing similar projects.

This development is a good example of modern web remote controlling system what can be popular in the future. In process of development of the system it was found out main possibilities and limitations, problems and their solutions which can help a lot for development similar projects in the future.

Subject headings, (keywords)

Web control system, Roboteh, remote control, mobile platform, video stream, mapping, auto navigation, visualization.

Pages	**Language**	**URN**
76	English	

Remarks, notes on appendices

Tutor	**Employer of the bachelor's thesis**
MattiKoivisto	Mikkeli University of Applied Sciences

CONTENTS

1 INTRODUCTION

In the age of high technologies and speeds it has become possible to develop a mobile surveillance system for different complicated missions which cannot be done by man. The most important thing is that this kind of system does not require any special expensive equipment and can be easy controlled through common devices as a personal computer or a mobile phone from anywhere.

“Roboteh” project is a striking example of up-to-date surveillance system. It is a completely mobile cross-platform complex which consists of a mobile platform which named “Decatrus”, server and client systems. The mobile platform is a car with an integrated video camera, sensors and a GPS module. The server performs functions of connecting link of system, manager and data collection centre. The client represents web control mechanisms and interfaces for an end user who controls the mobile platform. The same principles and structure can be realized for any platforms and for various tasks. It might be a stationary or mobile, surface or air system. The main function that can be performed by this complex is a remote surveillance of the environment.

The area of this final thesis is a developing of a web based system for remote control of the mobile platform with modern web oriented technologies and languages as JavaScript, JSP, Java Applet, Ajax, JSON, HTML, CSS, Google Maps andPostgreSQL database.

The aim of my work is to make a perfect web control service, to improve my knowledge in web technologies, to make ground works for features projects, to take a part in real interesting project and to work in a team.

My final thesis is a part of large project “Roboteh”. My role is a developing of the client part of the project which gives a user good interfaces and services of mobile platform controlling. Other project participants are Ivan Suvorov and StanislavShults.

A user interface must be easy-to-use and give a possibility of simplest control. This cross-platform system can beoperated from anywhere through Internet by using any user device with any operation system which has Internet access and a common web

browser. Also the client part must meet the requirements of reliability and very rapid interaction with the moving platform through the server. The most important things what will be considered in this thesis are developing of dynamic web system with using modern web languages and technologies, mobile module structure of user interface with animated control mechanisms and possibility of configuration for different users, real-time video transferring, rapid control data transferring, interaction this the server and the database, GPS navigation and control through Google Maps, visualization of the surrounding area by using data from sensors.

The structure of the final thesis is following. Chapter 2 describes modern web technologies which are used in this project. Chapter 3 contains description of Google Maps service and its potentials. Chapter 4 tells about the client part design. Chapter 5 relates to implementation of the system. Chapter 6 includestests of the system.And last chapter contains conclusions.

2 MODERN WEB TECHNOLOGIES

2.1 Web fundamentals

2.1.1 Common viewing a Web page

A web page is a document or information resource that is suitable for the World Wide Web and can be accessed through a web browser and displayed on a monitor or mobile device. This information is usually in HTML(HyperText Markup Language) or XHTML(eXtensibleHyperText Markup Language) format, and may provide navigation to other web pages via hypertext links. Web pages frequently subsume other resources such as style sheets, scripts and images into their final presentation.(Web page. [referred 20.01.2012])

With the help of Figure 1 Zambon&Sekler (2007, 3) describe what happens when we ask a browser to view a web page, either by typing a URL in the address field of the browser or by clicking on a hyperlink.

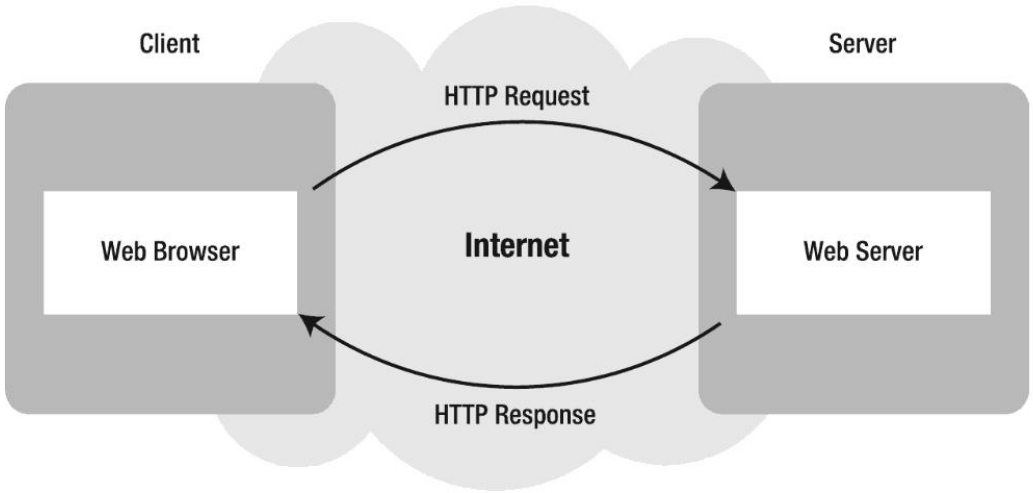

Figure 1. Viewing a plain HTML page (Zambon&Sekler, 2007, 3)

When we type the address into the address field the web browser resolves the domain name to an IP address through a domain name server. Then the web browser makes a HTTP request to the server by the founded IP address. In reply, the web server sends an HTTP response as a plain-text HTML page. At that time the web browser receives the server response, interprets the code of the page, requests referenced components from the server such as images, styles and scripts, and displays completed web page.

2.1.2 Modern Web interface

Any web application begins from development of web pages with using different web languages and technologies. This page can be browsed through Internet from any PC or mobile device which havea web browser without any special additional tools. It is comfortable, easy and available for everybody. This is a reason why web technologies are used more and more extent every day.

Usual web page corresponds a text file which contains a structured HTML code. HTML is a base language for developing of web documents. HTLM allowsto make a structure of the web page.

HTLM alone is static – its only responsibility is to display a text and images. That is, it cannot do anything that requires changes to the page after it is loaded, nor can it perform tasks such as adding two numbers together.(McIntire, 2008, 7)

Another technology allowsto make the web page more nice-looking. CSS (Cascading Style Sheets) is a standard which defines how to display HTML elements. CSS gives an opportunity to make the web page prettier with less weight. It also makes the page code easier and more understandable. CSS can be used for developing smallbut very bounded dynamic features.

Nowadays web applications and services become more and more dynamic. A web application can generate dynamic content, building web pages on the fly from sources of data that may include data supplied by user (Parsons, 2008, 2). Some of web application no longer can be imaged without dynamics. A lot of popular software moves to web.

The main problem of basic web languages as HTML, PHP, Perl and JSP is refreshing. The script is executed on the server and gives a user only text data which can be looked through a web browser. If the user wants to get more information or change something the web page will be generated fully anew by the server script and the user again has to wait while the page will be loaded by the web browser. Sometimes it takes too much time and it is so boring. And it is certainly inappropriate for applications which require regulardata transferring and page updating.

Following language is good solution of that problem. JavaScript is object oriented script language which allows to modifyan HTML page on the client device without request to the server and long waiting. JavaScript is a client-side language.

But very often applications require updating data from the server and inputting data to the server without refreshing the whole page. In that situation it is right to use Ajax(Asynchronous JavaScript and XML). Ajax technology allows to make a request to the server, get a reply from it and use these data for page updating through JavaScript. It is executed quite fast and almost imperceptibly for the user.

JavaScript coupled with Ajax gives a good opportunity for developing of modern dynamic web applications. This combination answers the requirements of cross-platform and cross-browser compatibility that is very important.

Also it is worth noting that any up-to-date web application processes huge amount of data which are stored usually in a database or special repositories as XML or JSON files. A possibility of storing those data gives great potentials for development modern dynamic web applications.

2.2 Web languages and technologies

2.2.1 JavaScript

JavaScript is a client side scripting language what provides a powerful extension to the HTML,it is interpreted language that means its code is embedded directly to HTML page as a part of the web document (Gandy &Stobart, 2005, 4). Thereby it can be run only within the web browser. When a user opens the HTML page the JavaScript code is downloaded and translated by the client machine. It is a lightweight language and usually does not require high network capacity. Syntax of JavaScript is very easy and it allows web developers who do not know the details of the language to make some easy but wonderful features.

One of the main and useful possibility of JavaScript is that it can read and write HTML elements and their styles, change and modify them, add new ones and generate

totally new HTML structure for the web page. JavaScript can react to different events. It is very usable and widely used for processing of user events. JavaScript very often is used to validate data before it will be transmitted to the server. Also it can work with cookies and detect the visitor's browser. Owing to these possibilities JavaScript includes to the web page more dynamic, usability and beauty.

JavaScript is object based language. But it has nothing in common with Java. They are completely two different languages. Like HTML and CSS the JavaScript does not require any special tools for writing a code. Any text editor is suitable for that.

JavaScript is a client-side scripting language. Everybody can see its code in downloaded page. Therefore it is not appropriate for developing secure web applications.

Gandy &Stobart (2005, 13-14) have described main JavaScript security implications:

- Scripts cannot make changes to the user's display preferences in the browser
- Scripts cannot access the history object of the browser in order to identify previous web pages you have visited.
- Access to the user's file system is restricted.
- You cannot issue a mail message or news item from within JavaScript code to prevent the user's email address being obtained without their knowledge.
- There are restrictions on what you are allowed to do with windows.
- JavaScript can respond to user events on the machine, but you cannot issue events that might run programs or perform tasks outside the web browser.
- Restrictions occur to prevent one script from accessing windows or documents downloaded from a different server. This prevents a script "spying" on other documents and windows.

JavaScript is a cross-platform and cross-browser language. But it might not work in old versions of some browsers, in browsers of mobile devices which cannot execute JavaScript or in browsers which have disabled JavaScript execution for security.

2.2.2 Ajax

Ajax is a modern Web technology. It is a group of interrelated web development techniques used on the client side to create interactive web applications. The applications employing Ajax technology are capable of retrieving data from the server asynchronously in the background, without having to interfere with the display and behavior of the currently loaded page.(Vakali, 2011, 91)

Comparison of a traditional client-server web communication model and Ajax model which is mentioned by Thomas(2008, 3-5) is described below.

Traditional Web application communication is shown in Figure 2. When a user opens a web page a web browser makes a request to the server. The server transmits data to the client side. Web browser receives these data and represents them to user. The user looks over the page and usually makes some event such as a clicking to a link. The web browser again sends request to the server. It gets a reply with new data and totally substitutes the page.

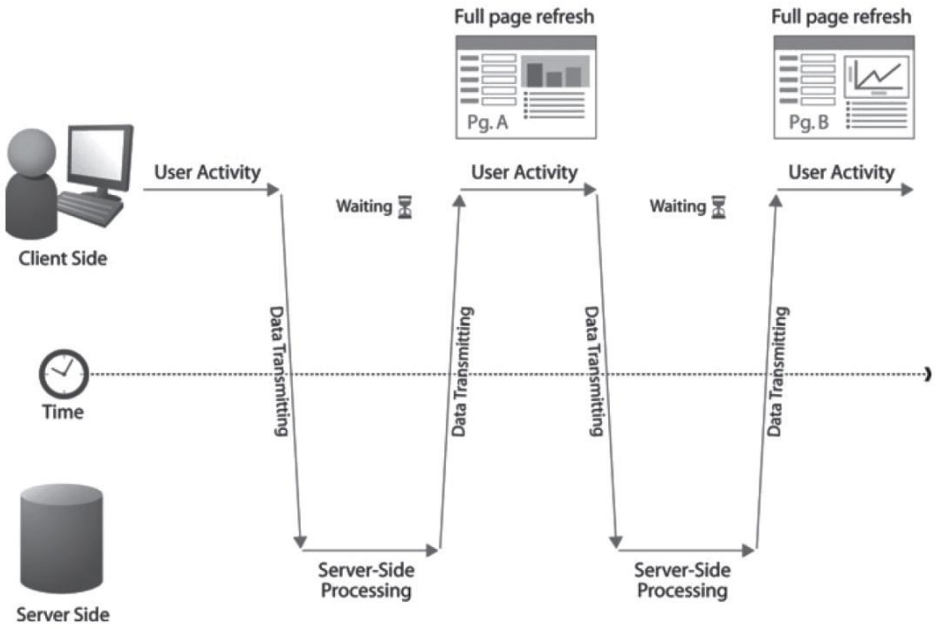

Figure 2. Traditional web application communication flow(Thomas, 2008, 4)

This process is always activatedby the user. The server every time sends full data to the client side even data is the same as previous ones.

Very often a part of data is repeated. It might be a code of website template particularly a header, menu, footer, sidebars, some content or a web application which changes only small part of data presented to user. Thereby a lot of unnecessary work is performed. It takes more network capacity. Also the server and the web browser process more data. It requires more time and a user has to wait.

Ajax applications use significantly different model of communication which is shown in Figure 3. This model means that a web page is loaded once and the user's events replace only that part of the page which should be updated.

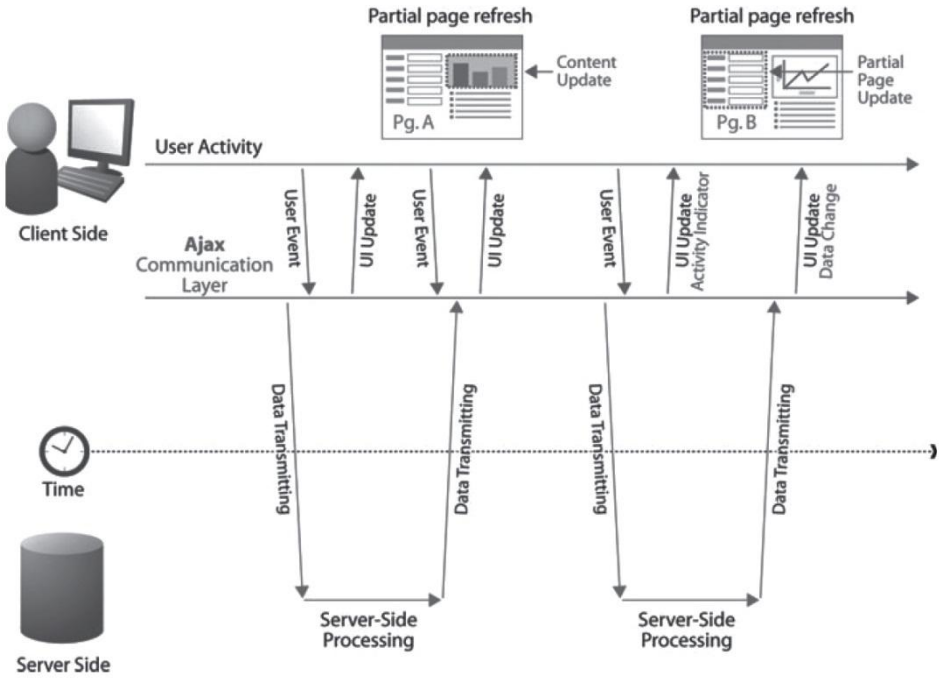

Figure 3. Ajax-style communication flow (Thomas, 2008, 4)

When a user makes an event the JavaScript makes a request to the server side. The server part might be a script which receives the request from the client side and processes it. The server side script should be PHP, Perl, JSP or another scripting language code which is located on the server. This server requests handler after receiving the data from the client, executes its own code with received parameters and returns the result. This result is received by the client side Ajax script and processed by JavaScript. When JavaScript receives data from the server side it can update HTML elements on thecurrent web page.

This process is asynchronous. It allows the user to perform other actions or make other events while one event is already processed. Ajax does not require anything special except an ability to perform JavaScript on the client side and an ability to perform the server side scripting file on the server.

Ajax has become more and more popular. A striking example of using Ajax in the Web World is Google. Services such as Gmail, Google Maps, Google Docs are very powerful web applications which represent a full dynamic and abilities of Ajax technology.

2.2.3 JSP

JSP (JavaServer Pages) is a Java technology that helps software developers to create dynamically generated web pages based on HTML, XML or other document types. JSP is a server-side programming language.(JavaServer Pages. [referred 25.01.2012])

A JSP page is an HTML web page that has been enhanced by the addition of special tags and Java code, both of which are used by the web server and the JavaServer Engine to create a page for a client's web browser. The JavaServer Engine strips out the special tags and Java source code, and uses the code to interpret and create the page that the web server sends out.No Java code is actually sent to the client. Thus, any web browser with or without Java installed can use a JSP application. (Cook, 2002, 4)

Bergsten (2004, 4) makes an example in Figure 4 how dynamic web page is generated from JSP script.

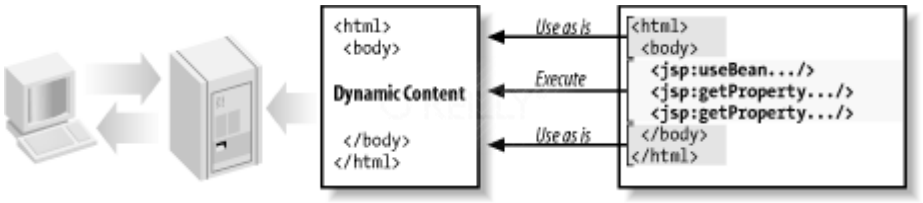

Figure 4. Generating dynamic content with JSP elements (Bergsten, 2004, 8)

A JSP contains standard HTML tags as a regular web page and special JSP elements that allow the server to insert dynamic content into the page. JSP elements can be used

for variety of purposes such as retrieving data from a database or registering user preferences. When a user asks for a JSP page the server leaves invariable HTML tags and executes JSP elements, merges result and sends this finished page to the browser.

With JSP the web page does not actually exist on the server. Zambon&Sekler(2007, 4-5) explain in Figure 5 how the server creates a JSP page.

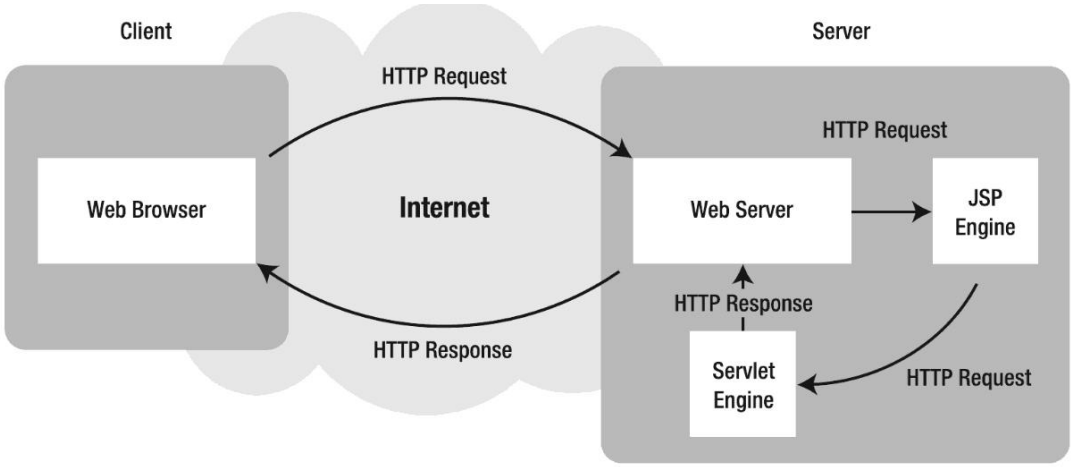

Figure 5. Viewing a JSP page(Zambon&Sekler,2007, 4)

A web browser sends an HTTP request to a web server. The web server has extensions necessary to identify and handle Java servlets. It recognizes that the HTTP request is for a JSP page and forwards it to a JSP engine. The JSP engine converts the JSP page into Java servlet. Then JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine. The JSP engine only converts the JSP page to Java and recompiles the servlet if it finds that the JSP page has changed since the last request. This makes the process more efficient than with other scripting languages and therefore faster.

Other appropriate scripting server-side languages which can be alternative instead of JSP for developing dynamic pages are PHP and ASP. But JSP has some advantages. The dynamic part is written in Java. It has full power of Java API. JSP is portable to other operating systems and Web servers. JSP avoids few troublesome areas of cross-platform Java development.

Thereby JSP is a powerful server-side scripting language which is suitable for different tasks especially for developing of complex dynamic web applications required to use power tools or Java libraries.

2.2.4 JSON

JSON (JavaScript Object Notation) is a data exchange format that is subset of the object literal notation in JavaScript (Holdener, 2008, 86). JSON is a lightweight alternative to XML.

These two technologies can be compared. Figures 6 and 7 show examples of JSON and XML models of data representation.

```
{ "classinfo" :
    {
        "students" : [
            {
                "name" : "Michael Smith",
                "average" : 99.5,
                "age" : 17,
                "graduating" : true
            },
            {
                "name" : "Steve Johnson",
                "average" : 34.87,
                "age" : 17,
                "graduating" : false
            },
        {
                "name" : "Rebecca Young",
                "average" : 89.6,
                "age" : 18,
                "graduating" : true
            }
        ]
    }
}
```

Figure 6. JSON model

```
<classinfo>
    <students>
        <student>
            <name>Michael Smith</name>
            <average>99.5</average>
            <age>17</age>
            <graduating>true</graduating>
        </student>
        <student>
            <name>Steve Johnson</name>
            <average>34.87</average>
            <age>17</age>
            <graduating>false</graduating>
        </student>
        <student>
            <name>Rebecca Young</name>
            <average>89.6</average>
            <age>18</age>
            <graduating>true</graduating>
        </student>
    </students>
</classinfo>
```

Figure 7. XML model

Advantages and disadvantages of using JSON instead of XML are following:

- Data format in JSON can be easy used without any modification by JavaScript like a JavaScript object in dot notation.
- It has the best compatibility with JavaScript.
- The weight of JSON file is much less then XML file.
- JSON data can be easy read and written by any web language as a simple text file.
- XML is much more understandable and readable for human than JSON.

JSON is a good simple exchange format which is suitable for any web scripting language especially for JavaScript because it has good abilities to parse JSON data quickly and easily.

2.2.5Java Applets

A Java applet is a special kind of Java program that is designed to be transmitted over the Internet and automatically executed by a Java-compatible web browser.

Furthermore, an applet is downloaded on demand without further interaction with the user. If the user clicks a link that contains an applet, the applet will be automatically downloaded and run in the web browser. (Schildt, 2011, 5)

Usually a Java applet is a small program which is used to display data from the server, to handle user input or to provide simple functions. Java applets run within a HTML page and can be viewed by browser which supports Java. Applets can be run on any computer, also on the smart-phones but it is not suitable for traditional mobile phones which do not support it.

Java applets are included in a web page by using special HTML tag <APPLET> which has a reference to Java applet on the server. When a web browser downloads a page it follows the reference and downloads the applet. The tag <APPLET> has possibility to change a size and a position of the applet on the page. In Figure 8 there is a common scheme of applet working process.

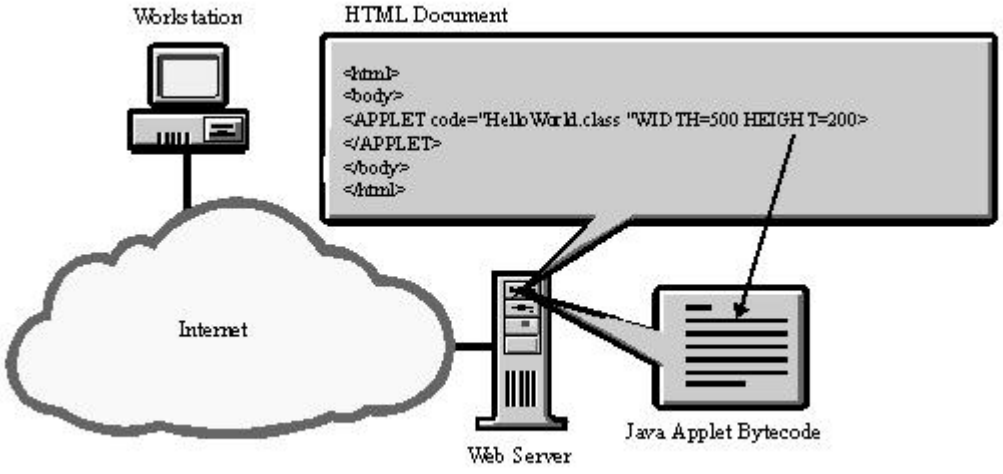

Figure 8. Applet working process

After downloading a web browser uses JVM plug-in to run the applet.JVM (Java Virtual Machine) is a software layer that is responsible for executing Java byte-codes (Debbabi, 2007, 18). But there is one inconvenience. JVM has to be up to date on the client.

Due to the fact that the applet is downloaded and runs automatically it must be prevented from doing harm. There is a risk that the applet might contain a harmful

code such as a virus or a Trojan horse. Therefore there are some restrictions. An applet is not able to read or write files on the client. Also it cannot communicate with a local server. But it has access to event handling and access to network.

Java applet can be run on the different devices with different operation systems. This makes itsuitable for cross-platform and cross-browser application.

2.3 Databases for Web

Databases are the heart of many information systems. They store data, basic facts that are processed to provide information, some useful combination of facts. (Prigmore, 2008, 1)

Figure 9 shows a process of interaction between a client, a web server and a database server.

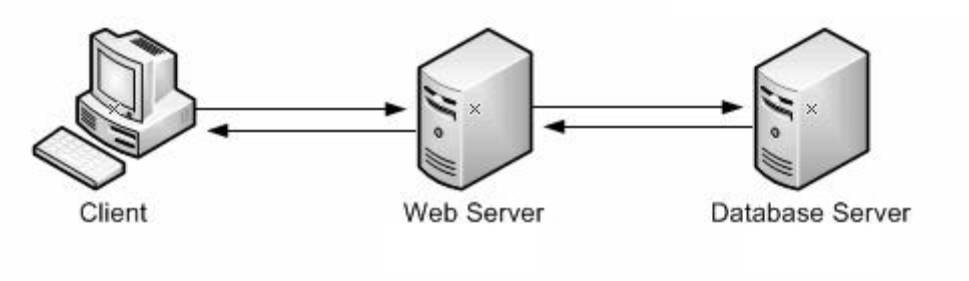

Figure 9. Tier of client with database

As is known when a user opens a web page the web browser makes a request to the web server. The web server gets the request, generates the web page if it is dynamic page or just takesa ready static page and sends it to the client. The client receives the required page and represents it through the web browser.

For web applications which use database for storing application data the web server has to read or write data from or to database. In process of a web page generation or process of saving, updating or deleting data in the database, theweb server makes one or more requests to the database server. The database server and the web server can be located on the same physical server or on different ones.

When the web server receives a request from the client to execute a server script such as PHP or JSP it has to make a connection to database, enquire, get reply,close the connection, process received data and finally create a web page on bases of these data. Sometime a creating one small page requires a lot of operation executions. For that reason it has to happen very fast. Speed is one of the main characteristics of a good database.

Most popular, fast and reliable database management systems for Web are MySQL and PostgreSQL.

3 GOOGLE MAPS

3.1 Google Maps service

Google Maps is a map service which can be viewed in a web browser. Depending on location, user can view basic or custom maps and local business information, including business locations, contact information and driving directions. (Welcome to Google Maps. [reffered 28.01.2012])

Google Maps can be viewed one of several ways:

- Visit maps.google.com.
- View a web page with an embedded Google Map.
- View Google Maps on a mobile device.
- View a private map created by Google Earth Enterprise products.

Google Maps has a simple and useful interface which is suitable for any device whether desktop or mobile device. The interface of this service is shown in Figure 10.

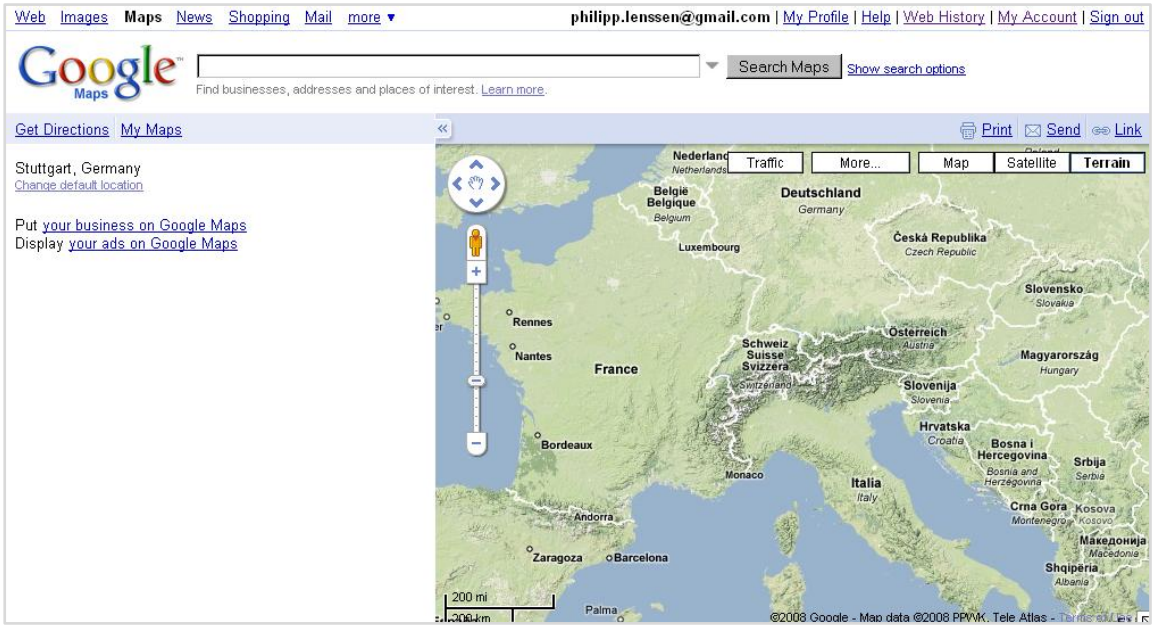

Figure 10. Google Maps interface

Google Maps gives following main features:

- Direction and route from one location to another by driving, walking, biking or public transportation.
- Own personal map and marked locations.
- Searching of different places, addresses, roads, geographical names.
- Adding layers over the map such as a photos, traffic, transit, terrain, weather, videos and etc.
- Printing or sending a map to somebody.
- Navigation around the map and zooming locations.
- Street-level imagery navigation and viewing street cyclorama in reality or its 3D projection.
- Toggling between Map, Satellite and Earth types of map.

Google Maps provides high resolution satellite images of most attended locations like cities or famous places. But most of the sparsely populated areas still have not good detailed view. Google Maps supports 20 levels of scale from 10000000 to 10 metres.

Google Maps uses JavaScript very extensively. When a user navigates the map, searches locations, makes markers and routes on the map the data transferring is

goingin the background the map is not reloaded. It is possible with using Ajax technology. For data transferringGoogle Maps uses a JSON format.

Also the Google Maps provides similar projects based on the Google Maps like Google Earth, Google Moon and Google Mars.

3.2 Google Map API

The Google Maps gives a possibility to include the maps with almost all its features into any web page by using open API(Application Program Interface). The Google API defines a set of JavaScript objects and methods that are able to use in own website or web application.

Google Maps has a wide array of APIs that let embed good functionality into a web page and overlay own data on top of it. The Maps API is a free service, available for any web site that is free to consumers. To get a permission to use Google Map API it is necessary to sign up for Google Maps API key and accept the terms of use. (Google Maps API Family. [referred 30.01.2012])

There are few lines of Google Maps development:

- Maps JavaScript API.
- Maps API for Flash.
- Google Earth API.
- Maps Image API.
- Web Services.

Maps JavaScript API is a solution for Maps application for desktop and mobile devices. Maps JavaScript API is based on the JavaScript and requires from developers to be familiar with JavaScript and object-oriented programming concepts. Maps JavaScript API is the most popular and easiest way to use Google Maps on the web page.

Maps API for Flash allowsto use Google Maps in the Flash applications. Similar with JavaScript version this API provides a number of utilities for manipulation and adding content to maps. Now this Flash version is deprecated.

With Google Earth API it is possible to add 3D Google Earth into a web page. It allows a 3D modelsapplying, markers and linesdrawing, images draping over the map and making sophisticated 3D application. Google Earth is the most perspective direction of Google Maps.

Maps Image API allows to add static images or street panoramas to the map on the web page very easy by constructing the URL of the page. Also it supports client styling, high image resolution, markers and geometry.

The Google Map API provides web services as an interface for requesting Maps API data from external services and using them within Maps applications. These services use passing URL parameters such as arguments for request and return data in JSON or XML formats.

For embedding Google maps through JavaScript API to the web page a developer has to include special code into the page. An example of Google Maps API is represented in Figure 11.

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
    <style type="text/css">
      html { height: 100% }
      body { height: 100%; margin: 0; padding: 0 }
      #map_canvas { height: 100% }
    </style>
    <script type="text/javascript"
      src="http://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sensor=SET_TO_TRUE_OR_FALSE">
    </script>
    <script type="text/javascript">
      function initialize() {
        var myOptions = {
          center: new google.maps.LatLng(-34.397, 150.644),
          zoom: 8,
          mapTypeId: google.maps.MapTypeId.ROADMAP
        };
        var map = new google.maps.Map(document.getElementById("map_canvas"),
            myOptions);
      }
    </script>
  </head>
  <body onload="initialize()">
    <div id="map_canvas" style="width:100%; height:100%"></div>
  </body>
</html>
```

Figure 11. Google Maps API code example

HTML page has to contain a canvas element which contains and presents the map. The main function is a function of map initialization. It creates a map object with default parameters like coordinates, zoomand type of map, and put it to the canvas object. Latitude and longitude are used for designation of coordinates.

Just adding Google Maps to the web page is not a difficult task. More complex functionality requires a knowledge and use of other more elaborate functions from Google API library. It involves functions of working with events, navigation control, styling, overlays (markers, icons, polylines, polygons, circles, rectangles and other shapes), windows, layers, directions, distances, geocoding, street view and etc.

3.3 Potentials of Google Map application

Google Maps is a powerful service which has huge potentials and perspectivesof using. Google Maps API allows todevelopa Google Maps based web application for variety purposes.

Nowadays there are navigation and tracking systems which are based on the Google Maps. Even small companies already use Google maps to control local transport traffic or social services. It is possible to use GPS coordinates and apply them in Google Maps. Also there are a lot of web services such as weather-cast or traffic informer,services which can help to find a location of a necessary place or object, a closest point and make a best route to it. That kind of systems can use Google Maps due to a possibility of adding own layers with variety information on the map.

Google Maps gives a good chance of Earth exploration and not only. Scientist can observe on the map a movement of under study objects. Companies can know where their units are and watch the process of them work. People can be informed though the map about many events and places in the World. The human life can be saved if a lifesaver would be able to observe him on the map and help him to save some valuable time.

The Google Maps might be used already in many applications to show a location of objects in different areas of the life. Also the Google already has 3D maps and 3D

projections of buildings, and continues to improve them. Maps become more and more detailed and rich. Thereby it gives more and more potentials of maps usage.

4 DESIGN OF WEB CONTROL SYSTEM

The design of the modern web car control system in this project corresponds development of a convenient interface with all necessary control mechanisms, elements of interaction with the server which is connecting link between the user interface and the mobile platform and other logic car control algorithms and tools.

The main requirements for development of this web control system are following:

- rapid control data transferring
- rapid video transferring
- stylish, easy-to-use and clear interface
- cross-platform
- fault tolerance

4.1 User interface design

A development of the modern user interface with strict requirements needs the use a modern approach of designing and knowledge ofcurrent technologies. The client interface has to be designed in the single style with modern dynamic elements of controlling. The control mechanisms have to be handy, clear and simple as much as possible. A user has to know what is going on with the car and get a response from the interface when he is controlling the car.

4.1.1 Module structure

The user interface of the system corresponds a set of modules. Each of them is separate panel with its own functionality. The module structure is a mobile structure which allows a user to dispose elements of the interface in accordance with the user's need.

Each module is a panel which has itsheader, content, resizing element. The header contains a title of the panel and introduces it. Also it allows to change a position of the module on the page by "drag and drop" functionality. A module size can be changed

through resizing block in low right corner. A scheme of the panel design is shown in Figure 12.

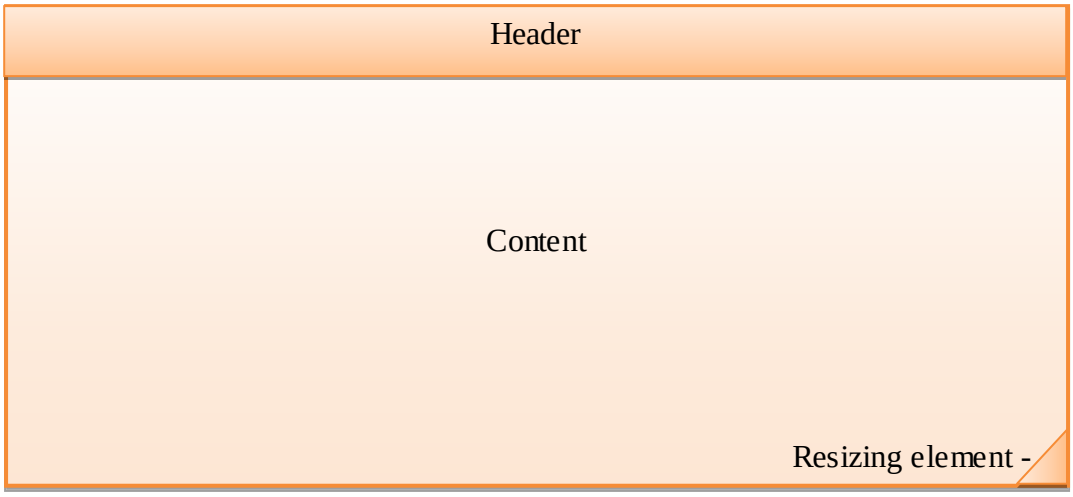

Figure 12. Common panel design

There are following panels in my design: Main, Server, Video, Map, GPS, Sensor, Car and Camera. All panels are designed in a single modernstylewith anappropriate color box.

4.1.2Panels functionality

The Main panel is a sole panel which has static position. It is located in the top of the page. A size of this module cannot be changed. A width of Main panel is the same as width of the browser's window. The module cannot be moved to another position because it has to be in full view. But it is possible to hide the module when it is unnecessary to use. When the Main panel is hidden it smoothly goes beyond the window. Especially it is useful for mobile devices with small screen.

The main function of the Main panel is a module and user management. There is a possibility to control visibility of modules. Each module has three states: visible, invisible and transparent. The invisible state allows a client to hide unnecessary panels. For example if we are not going to use a map we can just hide the Map panel and use free space for more usable disposition of other modules. The transparent state hides a background of the panel. Thereby one panel can be located over another one and does not obstruct visibility of a lower one. For example in mobile device we can

set a video to full screen and put a car control panel over the video in transparent mode.

One additional function of the Main panel is mode switching. Each mode is a special panels disposition for different sizes of screen. There are few default screen modes: mobile mode for size of screen 600*480, modes with size 1024*768 and size 1600*1024. The user can choose any default mode depending on the size of the screen just by clicking to the proper button. After that simultaneously all panels will smoothly move to intended positions and change visibility to proper one.

For example in mobile mode the Video panel goes to full screen and takes whole space. The Server, Car and Camera panels are located on the sides of the screen in transparent mode. Other modules become hidden.

The Server panel contains tools for management of the interaction with the server and shows information about that. User can turn on or off control command transfer to the server and video stream. There is a possibility to change a rate of sending control data and data from database. Also the panel displays information about successful or unsuccessful transmission and server state.

The Video panel contains Java applet which receives a video stream from theserver and plays it. The video can be turn off if a user does not want to use it.

The Map panel corresponds a map which is based on the Google Maps. The map shows a current position of the mobile object in the form of small icon.Also it can display the path the car has travelled. All routes can be saved and usedlater. Thereby through the map the user can monitor the car and its routes. Also the map can be turn off if it is unnecessary to use.

One more important thing is that the map allows user to add own routes on the map or set up stored routes. The route consists of points which are set by user. The routes are necessary for possibility to control the car by assigning a route. It is very useful just to set few points on the map, provide guidance to the car and it will automatically follow the instructions on the specified route. This functionality is one of the main andthe most difficult aims of the project development.

The GPS panel gives information about GPS coordinates and routes of the mobile platform. It correspondsan additional panel to work with a map. This panel displays information about points which are marked on the map, route lines of car motion and tools to work with them and manage the map. Also the panel shows subsidiary information such as a distance, amount of points and current GPS coordinates of the mobile object.

The Sensor panel is responsible for visualization. It contains the picture of the sketched car with sensors and shows information from the sensors about environment around the car.

The Car panel is a panel of car control. It contains motion control elements. The car can be directed to the front and back, left and right sides. When a user controls the car he is able to see which direction he is turning to.

Also there is a gear box. A gear state is shown on the Car panel because a user has to clearly know a current gear for a convenient car control. There is possibility to use auto gearbox.

Special animated design of car control mechanism was elaborated for easies and handiest using. It combines all control elements in a single control instrument.

The Camera panel contains elements of camera control. For a looking around the car the camera can be turned up, down, left, right and quickly turned back to default state. The default state is necessary because if the camera is not in default position it is so difficult to orientate by video from the camera and move the car to right direction. The camera can be turned to right or left direction up to 90° and 60° to up and down. Thereby almost half of environment can be viewed by the user from one car position.

An animated control mechanism is designed for a camera controllingin the single system style. It corresponds a simple and easy-to-use interface. Due to this the user always knows which direction he is turning the camera to.

4.1.3Screen variations

The user interface of this project is a dynamic alterable module interface. There are few screen modes of system which implies different dispositions of interface's panels.

When a user sign up and opens the main page the system loads configuration files with information about current user mode. In the process of page loading a web browser makes a request for configuration files to the server. The server findsnecessary files according to the specified parameters and sends these data back. After receiving the mode configurations the page can dispose all panels according to the configuration and finish its loading.

When a user turns a mode to another one the same scheme is performing. With new configuration all panels just smoothly change their positions. If the user changes a mode, moves panels in his own order, resizes them or changes visibility of panels then newly formed configuration is saved by the same way to the user configuration file on the server. Thereby the user always can configure his own interface in compliance with his wishes and abilities.

4.1.4 Configuration files

A configuration file (config) is a file which is located on the server and stores information about user's mode, panels positions, visibility and size of them. The file has extension .json. It contains the data in JSON notation.

For this purpose I could haveused XML files too. But because of some advantages in interaction between JSON, JavaScript and Ajax technologies the JSON was chosen for that.

There are several kinds of configs stored on the server. Default configs contain data for each default screen mode. Another type of them is a user config. Each user has own configuration file.

When a new user is added to the system he gets his own config with default parameters. When the user opens the pagethe system uses his configuration file. If the

user moves panels and changes their positionor sizehis new configuration is saved immediately into his configuration file.

Each config has common template and contains information about panels position, panels size, panels visibility and screen mode. These data are read or updated when a user sign in to the system and opens the control page or refresh it, when the user moves panels or changes them position, or switches the screen mode. Figure 13 describes a process of handling configuration files when some of these events occur.

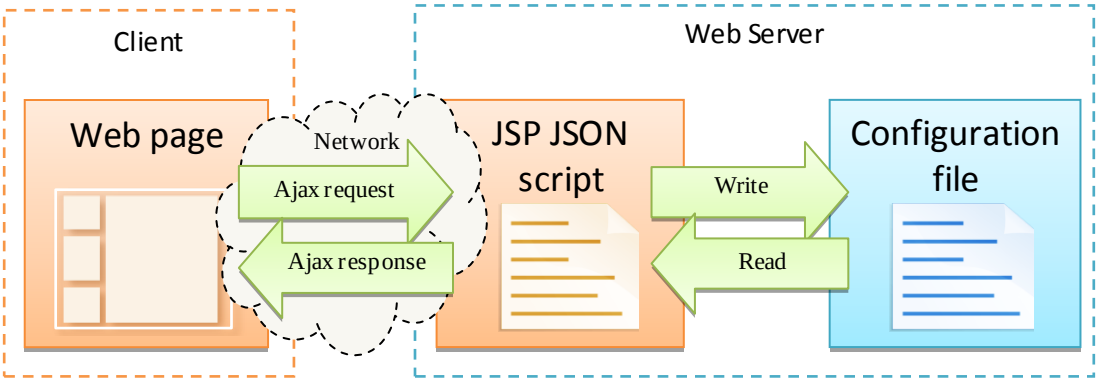

Figure 13. Handling of configuration file

The web page makes an Ajax request to the web server. Parameters of the Ajax request contain user name, type of action (set or get configuration data) and all configuration data if the request is going to save data into the config.

The JSP script processes the Ajax request. It writes new data to a proper configuration file if data should be saved. If the data should be gotten from the configthe JSP script reads the file and sends a response with all configuration data back. The web page processes the server response and applies a new configuration to the system.

4.2 Car control

The main aim of the user interface is to provide a user by handy, clear and multifunctional elements for car controlling. The basic devices for control and management of the system are a keyboard and a mouse (or touchscreen).

4.2.1 Control keys

The interface of the client side contains few elements for car controlling. A user can direct the car to four directions, change a gear and move car's camera by pressing proper keys on the keyboard. The control keys are described in Table 1.

Table 1. Control keys

Key	*Description*
↑	Go forward
↓	Go back
←	Go to the left
→	Go to the right
R	Increase the gear by one
F	Decrease the gear by one
Space	Hold the speed
W	Turn the camera up
S	Turn the camera down
A	Turn the camera to the left
D	Turn the camera to the right
Q	Reset camera to the default state

The main keys are arrow keys. These are common keys and they usually are used for motion control. A user can use his right hand for that. By another hand the user can control a gearbox with keys "R" and "F". Keys are located so that gear increasing is upper key and decreasing is lower key. A little bit to left on the keyboard there are keys for camera control. These keys are very often used for similar actions in different applications like simulators or games. They can be handled by the same left hand that is quite comfortable. Key "Q" is necessary for fast resetting camera position because the user sees a picture from camera when he is controlling the car. And it is very difficult when the camera is not in the right position.Key "Space" is used for holding a current speed of the car when auto gear mode is turned on.

4.2.2 Handling user events

When a user is controlling the mobile platform he is pressing control keys on the keyboard. One of the main problems of the project is to reduce a time between pressing a key by user and car's reaction to that. This time especially depends on processing time on the client part and data transfer time.

When a user event happens the system handles this event. It has to store values for each key. If the key is pressed down or released the system immediately saves the value of this key. Thereby the client side at any moment knows exactly which key is pressed and which is not.

Having information about pressed keys the system can use it for creating a special command packet for the server. This packet consists of commands which have to been passed to the mobile platform through the server and executed on the car.

4.2.3 Gear box

For a normal movement every usual car has some number of speeds. The car of this project has 200 levels of the speed which are described in Table 2.

Table 2. Speed levels

Speed number	*Description*
1..99	Backward speeds. 1 is maximum one. 99 is minimum one.
100	Neutral speed.
1..200	Forward speeds. 101 is minimum one. 200 is maximum one.

The neutral state of the car means that its speed is 100. 101 is a first forward speed. 99 is a first backward speed. Because of the large amount of speed levels differences between adjacent levels are very small. 200 levels are too much. It is not convenient for a user. Because of that the number of speeds is decreased to 11 levels. Correspondence between levels of the speed for a user and real levels are shown in Table 3.

Table 3.Correspondence between speed levels

User's speed	*Real car speed*	*Description*
-5	1	Maximum backward speed
-4	20	Backward speed
-3	40	Backward speed
-2	60	Backward speed
-1	80	Minimum backward speed
0	100	Neutral speed
1	120	Minimum forward speed
2	140	Forward speed
3	160	Forward speed
4	180	Forward speed
5	200	Maximum forward speed

Thereby a user can use 5 forwards speeds, 5 backward speeds and neutral one. He can rapidly switch the speed by pressing proper button on the keyboard.

To stop the car the user has to press a button of opposite direction. If he is going forward he has to press a backward speed key. The car uses a neutral speed and after replicated pressing it goes back. This is a reason why the car cannot go immediately to opposite direction. Fast pressing a key with opposite direction is good solution to stop it because the car has no brakes.

4.2.4 Auto gear

Auto gear gives a user a possibility to control the car without switching gears manually. When the user goes forward by pressing forward key the gear box is automatically changed to the first forward speed. While the user is continuing to press this button it smoothly is increasingthe speed of the car. The backward speed is also increased by long pressing of the backward speed button.

In auto gear mode a user can switch a speed to any one from 1 to 200. Each level is able for use. A special auto gear algorithm is designed. Each 30 milliseconds the system check whether any key is pressed or not. If no one of the keys is pressed the

speed is approach to neutral speed. If the forward speed button is pressed the speed is increased by one. If the backward speed button is pressed the speed is decreased by one. Thereby long pressing allows to increase the forward or backward speeds up for few seconds and short pressing allows to move the car this almost one speed. But it is quite hard to keep one speed. For that there is a special key which can keep the current speed. Then the user presses “Space” the current speed stays on the same level. With that user can slowly change the speed by one. It does not matter how long the user holds the button.

4.3Mapping

A mapping allows a user to observe the car wherever it is. Through the map the user can see a current car location, car routes, a distance of the routesand a direction of movement. Google Maps API gives good toolsto implement this.

4.3.1 GPS coordinates

The car contains a GPS module which sends GPS data to the server. The server handles these data and saves it to the database. The client can take any GPS data from the database. It allows to use this information to define the car location any time when car was working and GPS data was saved.

GPS data correspondinformation from satellite about location, time, direction and speed of the car. These data are sufficiently precisebut theycan have some inaccuracy especially inside a building. GPS data from the GPS module with description are shown in Table 4.

Table 4. GPS data

Data format	*Description*
HHMMSS.XXX	Time of GPS satellite in UTC. HH – hours. MM – minutes. SS – seconds. XXX – milliseconds.
DDMMYY	Date. DD – day. MM – month. YY – year.
A	Satellite status. "A" is active. "V" is invalid.
HHMM.M	Latitude. HH – hours. MM.M – minutes.
N	Latitude hemisphere. "N" is north. "S" is south.
HHHMM.M	Longitude. HHH – hours. MM.M – minutes.
W	Longitude hemisphere. "W" is west. "E" is east.
X.XX	Speed of the GPS module movement.
XXX.XX	Direction of the movement between 0 and 360 where 0 represents north, 90 - east, 180 - south and 270 - west.

The most important things in GPS data are latitude and longitude. These coordinates are used for positioning the car on the map.Time and date of the car location on the specified position allow us to know what time car is located in this position. The coordinates can be used only when the status is active i.e. car is fixed and satellite can calculate the car position. The speed and direction can be used for some algorithms like car controlling through the map.

Latitude and longitude from satellite and latitude and longitude in Google Maps use different formats. Thereby these coordinates have to be converted to one format. It is possible to convert it toGoogle Maps format by using following formulas.

$$Latitude = HH + MM.M/60$$
$$Longitude = HHH + MM.M/60$$

Google Maps API gives tools to put markers on the map. The marker is shown as an icon on the map. After conversion the coordinates can be used to put car marker to the map for designation of its location. When the car is moving the client gets new

information from database and changes the marker position. Due to this a user always can observe a current car location on the world map.

The database stores GPS coordinates for all times of car working. Thereby the client can take all this information or make selection by time or date and shows separate routes of the car. And a user has good possibility to observer a current route or previous ones.

Google Maps API allows to use polyline for drawing a route on the map. For that the client requests necessary data from database and locally stores these coordinates. These GPS coordinates corresponda set of points which are used for drawing a route. Also there is a possibility to calculate a distance of the route and display it on the screen.

4.3.2 Car control by map

In order to control the car by map it is necessary to draw a route on the map. Google Maps API allows to put markers on the mapby clicking and to draw a line between them. Each marker contains its coordinates. The client saves marked positions locally. The system makes a table of markers and gives the user a possibility to change the markers, put new ones or delete them. Also the user can move any marker by “drag and drop” function with a mouse. Thereby the user can make several clicks on the map and the route will be drawn. User’s route can be saved in database as a set of GPS coordinates. The user can make also own route or load previous saved one and use it iteratively.

The route following algorithm of the car requires availability of GPS coordinates, direction of car motion and permanent feedback with the car. The main idea is to move the car to the approximate direction then get new GPS coordinates and correct the direction. The car is considered as reached the point if it is located in a fixed radius of this point.

Controlling through the map imparts some autonomy to the car. By user instruction the car can follow determined route and perform stated tasks.

4.4 Data visualization

The car contains ultra-sonic sensors in front, right and left sides. These sensors calculate a distance to the object in front of them. Thereby these data can be used for visualization of environment around the car. Data from sensors are saved into database and can be selected from it by client. The client uses sensor data for visualization objects which can impede car movement. In order to the user knows about hindrances the visualization is designed.

The visualization is a picture of the car with schematically depicted sensors and sketched free space between sensors and objects. If the car is coming to hindrance the distance between sensor and the object is shorten on the picture. Thereby the user knows more about environment around the car and can easer control it.

Also this information can be helpful in algorithms of car movement to prevent the car from accidents with objects, to take a detour in algorithms when car is going by itselffollowing to user instruction.

4.5 Interactions with the server

4.5.1 Command packet

The command packet contains the main data which are used for car controlling. The data correspond a set of characters each of them has own destination. The command packet is shown in Figure 14.

Figure 14. Command packet

The packet consists of four cells each of them contains only one character. In Table 5 there are descriptions of the command packet cells with possible values of them.

Table 5.Description of command packet

Cell	*Possible values*	*Description*
1	L	Turn the car to the left
	R	Turn the car to the right
2	1..200 characters of ASCII table	Speed of the car. The cell contains only one character. From 1 to 99 are backward speeds, 100 is the neutral one, from 101 to 200 are front speeds.
3	U	Turn the camera up
	D	Turn the camera down
	B	Reset the camera to default state
4	L	Turn the camera to the left
	R	Turn the camera to the right

4.5.2Command packet transferring

For real time mobile platform controlling there was designed a special scheme of interaction between the client and the server. The client sends command data to the COM server which sends that data to the car. There are several points in the route from a user action to a car reaction. And each point takes time to process the commands. For that reason the main requirements for each point is to reduce the time of data transferring as much as possible. A dialogbetween the client and server sides has to be extremely fast. A possibility of comfortable car controlling depends heavily on delays. Communication between the client and the server is shown in Figure 15.

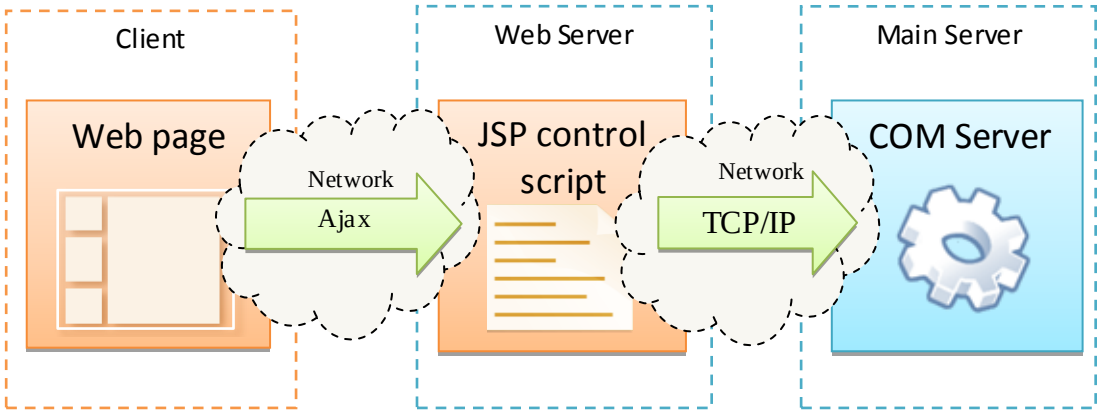

Figure 15. Scheme of a command packet transferring

The client sends the command packet through Ajax technology to the JSP script each several milliseconds. The JSP script is a server part of the client which is located on the Web server and it communicates with the COM server of the project. The COM server is a server which is interacting with a mobile platform. The Web server and main server can be located on the same hardware or different ones. When the JSP script receives data from the client it is executed and then it passes this command packet to the server through TCP/IPsockets.

Depending of connection rate the user can set an interval value. If the interval is wider there will be longerdelays between user action and car reaction. The default value is 50 milliseconds. Delay up to 100 milliseconds is acceptable for normal car controlling. Delay less than 20 milliseconds can be senseless.

4.5.3Database

The car is equipped with a GPS module and ultra-sonic sensors which can pass data to the client. It gives huge abilities to use these data for car coordination, mapping, making routes, evaluation of the environment around the car and creating some intelligent features.

The data from the GPS module and sensors go to the server which saves it into database and the client uses these data. Also the server saves client information such as commands, routes, some configuration and users data. The client has to send control commands as fast as possible without any delays but it is not necessary to get data from the car so fast. The user gets these data more seldom about one or two times per second. It is quite enough and is not loading the client PC and network. Figure 16 shows a scheme of interaction between the client and the database.

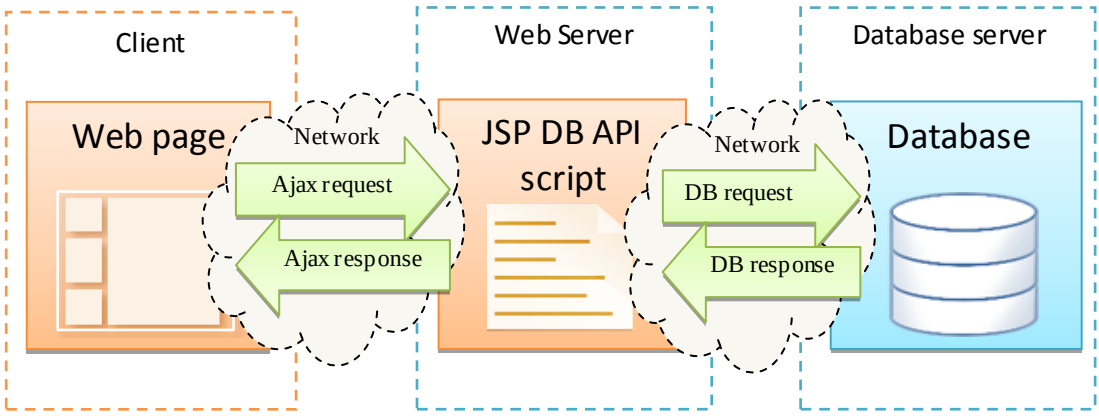

Figure 16. Interaction between the client and the database

If the client needs to get data from the database or write it to the database it makes an Ajax request to the JSP script with API which is handling database requests. This script processes the client request and makes a request to the database. Then the JSP script gets data from the database it arranges that and sends to the client in a suitable form.

4.5.4 Video transmission

As mentioned earlier the car has a movable camera. A user can view a video from car's camera. On basis of the picture from camera the user can orientate the car and control it. For real time car controlling it is necessary to get real time video. Video has to be without buffering and delays because when the user is moving the car his reaction depends from rate of video transferring.

Video server gets a video from the camera and transmits it to the client. To display video it was decided to use Java Applets. Java Applet can be loaded on the web page and it easily interacts with the video server through RTP (Real Time Protocol). Figure 17 describes a video stream transferring from the video server to the client part.

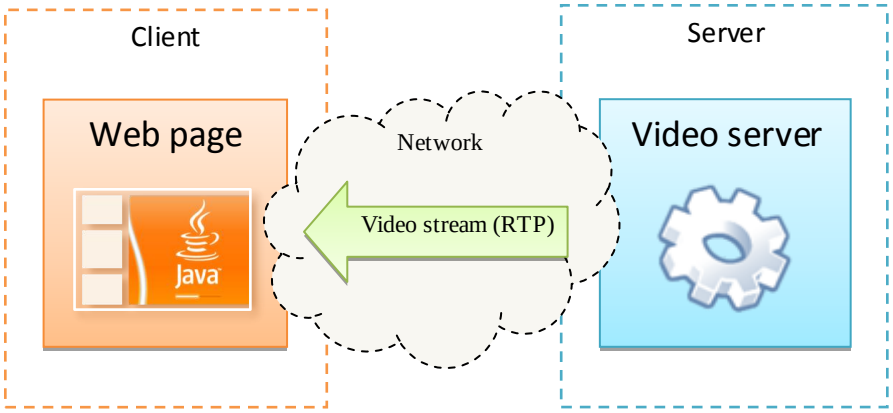

Figure 17. Video stream transferring

5 IMPLEMENTATION OF WEB CONTROL SYSTEM

5.1 User interface

5.1.1 Files structure

The project contains a lot of different types of files with variousextensions. Where are following types of files: html, css, js, json, jsp, jar, jpeg and png. The structure of the files is shown in Figure 18.

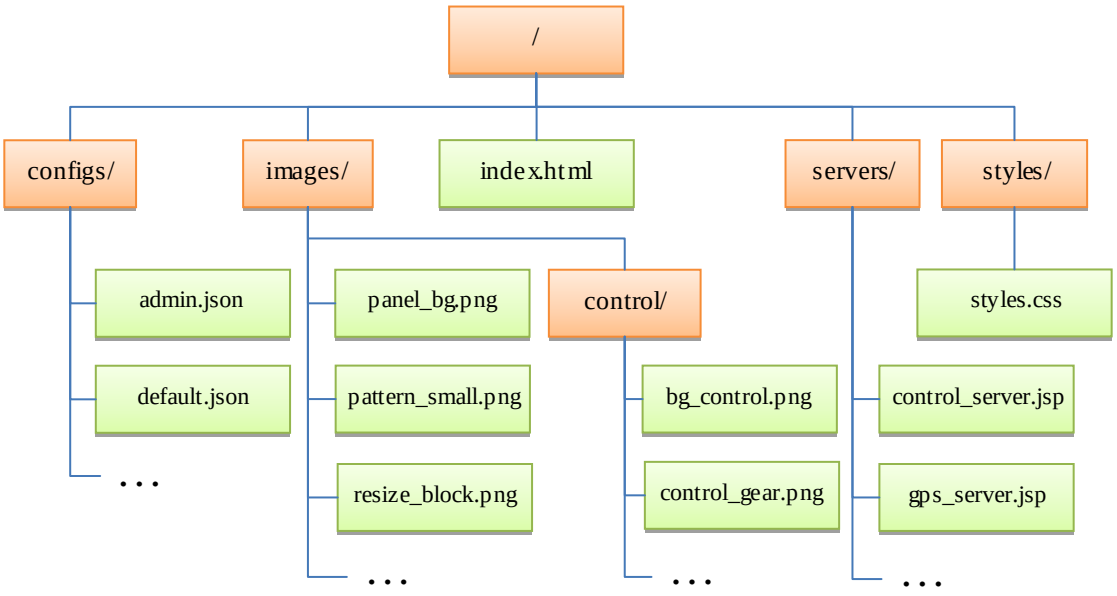

Figure18. File structure

The folder “configs” contains all configuration files with extensionjson. All images are grouped in folder “images”.All images for control elements are located in “control” folder. And folder “servers” contains jsp and jar files which communicate with a com server, video server and database server. The styles of the web system are in “styles” folder. And main page is index.html which is located in root.

5.1.2 Main page

The main page of the project is index.html. This is a common HTML document which has a simple structure.

```
<htmlxmlns="http://www.w3.org/1999/xhtml"dir="ltr"lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript"src="js/control.js"></script>
<script type="text/javascript"src="js/modules.js"></script>
<script type="text/javascript"src="js/gmaps.js"></script>
<script type="text/javascript"src="js/db.js"></script>
<linkhref="styles/styles.css"rel="stylesheet" type="text/css">
<scriptsrc="http://maps.google.com/maps?file=api&amp;v=2&amp;key=ABQ
IAAAAXZIALri8e1M1-
UX2igujHhRkPAwujHUUezZs1FKCnU7M913ZVhTYDm0oZ5g7zjMCy774n75ii2b7cg&se
nsor=true_or_false" type="text/javascript"></script>
</head>
<bodyonkeydown="keyDown(event);"onkeyup="keyUp(event);"onunload="GUn
load()">
<div id="body_panel"></div>
<div id="welcome_div"></div>
</body>
<script>modulesInicialization1Step();</script>
</html>
```

The character encoding of the main page is utf-8. The head includes JavaScript files, Google maps API and styles. The body of the page contains only couple of div tags. Thebody_panel is used as a main container for panels of user interface which will be generated by executing function *modulesInicialization1Step()*. There is a

welcome_div which contains the login form and welcomes a user until the page is generated and all pictures and files are loaded.

5.1.3 Panel generation

When the main page is loaded the function *modulesInicialization1Step()* is executed. This function starts the process of thepanels generation. It makes a welcoming image, shows it and calls the function *getJson()*to load configuration for user interface.

```
function modulesInicialization1Step()
{
varwin_width=getWindowWidth();
varwin_height=getWindowHeight();
document.getElementById('welcome_div').style.width=win_width;
document.getElementById('welcome_div').style.height=win_height;
document.getElementById('welcome_div').innerHTML=
"<div id='center_div_welcome_img'><imgsrc='images/welcome_img.png'
/><div>";
getJson("");
}
```

Function *getJson()* make a request to json_server.jsp to get a user's configuration data from a proper config on the server.

```
functiongetJson(mode)
{
varparams="action=getconfig&username="+user_name+"&mode="+mode;
json_request.open("POST","json_server.jsp",true);
json_request.setRequestHeader("Content-type","application/x-www-
form-urlencoded");
json_request.setRequestHeader("Content-length",params.length);
json_request.setRequestHeader("Connection","close");
json_request.onreadystatechange=jsonReply;
json_request.send(params);
}
```

The reply from the server is gotten by *jsonReply()* function. It receives data in text format from the server. Function *eval()* converts this received text to the json object. And then function *modulesInicialization2Step()* is called.

```
function jsonReply()
{
if(json_request.readyState==4)
{
if(json_request.status==200)
{
var text =json_request.responseText;
json_object=eval("("+text+")");
if(initialization)updateAllPanelsPositions();
else modulesInicialization2Step();
}
}
}
```

Second initialization function *modulesInicialization2Step()*calls *createNewPanel()* for each panel. For example it calls *createNewPanel()* for Video panelasit is shown below.

```
createNewPanel('Video',json_object.modules.Video.width,json_object.m
odules.Video.height);
```

Then created panel has to be positioned on the page.In order to do this a *setPanelPosition()* function is called.

```
setPanelPosition('Video',json_object.modules.Video.x,json_object.mod
ules.Video.y);
```

After that it applies a visibility for the panel by calling function *setPanelVisibility()*.

```
setPanelVisibility('Video',json_object.modules.Video.visibility);
```

When all panels are created in concordance with a configuration it sets a mode of the Main panel. Then the function putsproper content to each panel as it is shown for the Video panel.

```
html_string="<div id='video_content'>\n\
<div id='video_applet'></div>\n\
</div>";
putHTMLtoPanel('Video',html_string);
```

When all panels are filled by content and everything is in a default state, the initialization is finished and the welcome picture slowly vanishes from sight. Figure 19 present the user interface of the system after it has been loaded.

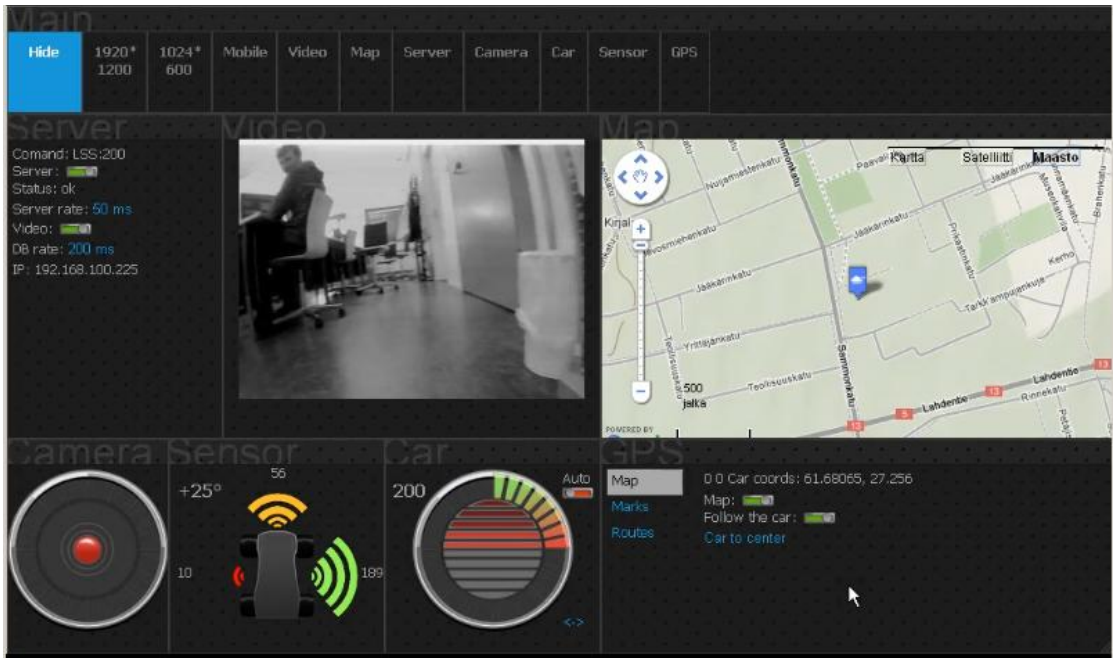

Figure 19. User interface of the system

5.1.4 Panel movement and resizing

The user interface gives a user a possibility to change panel's positions and sizes. It makes the system much more flexible and configurable.

To move a panel from one place to another the user has to click on the head of the panel by a mouse, hold the key, move themouse and release the key. It common

dragand drop action. To handle this process the system listens*onmousedown*, *onmouseup* and *onmousemove* events. If the mouse down event is happened on the panel's header the system takes the id of the panel, releases the panel from current position and makes it as a dragging object.

Then the user moves the mouse and the system processes every motion and changes a position of the dragging object. Thereby the user can see that the panel is moving. If the user releases the button the system catches this event, calculates a suitable position, fixes the panel and call the function *setXYToJson()* which saves the new configuration to user'sconfig.

```
varshiftLeft=parseInt(drag_object.offsetWidth/2)-20;
varshiftTop=-10;
drag_object.style.left=parseInt((e.pageX-shiftLeft)/50)*50+"px";
drag_object.style.top=parseInt((e.pageY-shiftTop)/50)*50+"px";
setXYToJson(drag_object.id,drag_object.style.left,drag_object.style
.top);
drag_object=null;
```

There is invisible marking of the page. It means that panels can be moved according to it. A distance between adjacent possible positions is 50 pixels. Also the size of the panel is multiply of 50. Thereby the user does not need to move panel or change its size pixel by pixel with the accuracy up to a pixel. The system by itself findsa necessary position and size for the panel.

The similar way is used for resizing a panel. When the user clicks on the resizing block the system setsthe panel as a resizing object. The user holds the pressed key and moves the mouse. The system refreshes a size of the panel. When the user releases the button the system fits the panel's size, free resizing object, fixes it and call function *updateSizeOfElements()* which updates size of the page's content. Thereby the moving and resizing of panels gives the user a possibility to configure and the interface of the system in accordance with his needs. Example of the freely configurable user interface is shown in Figure 20.

Figure 20. Freely configurable user interface

5.1.5 Panel visibility

For more comfortable configuration of the user interface the system has an instrument which changes visibility of any panel. It allows to hide the panel or to put one panel over another. There are three states of visibility of the panel: visible, invisible and transparent. Figure 21 shows all possiblepanel's visibilities.

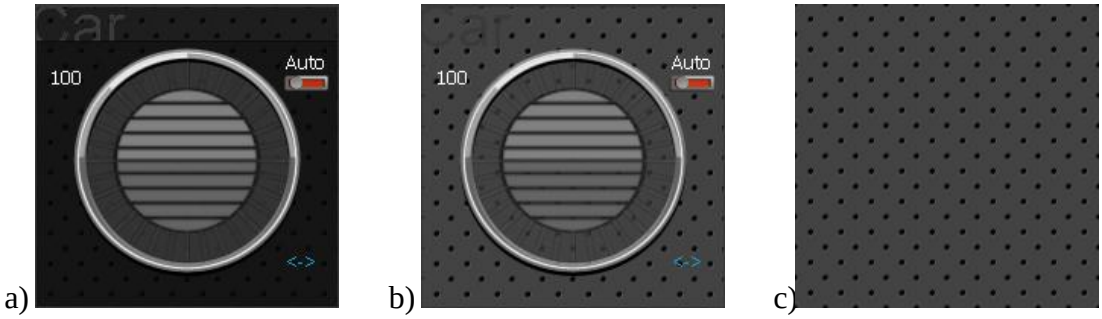

Figure 21. Three states of panel visibility

To change visibility the system calls function *changeVisibility()* when a user clicks to proper button on the Main panel. This function determinesstate of visibility and calls a function *setPanelVisibility()* to set it. If the panel is visible it becomes transparent, if it

is transparent it becomes invisible, if it is invisible it becomes visible again. Also the function change *z-index* of panel that makes the panel position over other. It is necessary because one panel can obstruct another one. The function *setPanelVisibility()*sets specified styles for panel.

```
function setPanelVisibility(id,visibl)
{
if(visibl=="full")
{
document.getElementById(id).style.display="block";
document.getElementById("content_"+id).style.backgroundImage="url('
../images/panel_bg.png')";
document.getElementById("title_"+id).style.backgroundImage="url('..
/images/panel_bg.png')";
}elseif(visibl=="transparent")
{
document.getElementById("content_"+id).style.backgroundImage="none"
;
document.getElementById("title_"+id).style.backgroundImage="none";
}elseif(visibl=="none")
{
document.getElementById(id).style.display="none";
}
}
```

After this procedure is completed the function *setVisibilityToJson()* is called to save new configuration to user'sconfig.

5.1.6 Mode switching

The system gives a user a possibility to change a screen mode of the system by clicking the proper mode button on the Main panel. There are few default modes. Mode with resolution 1900*1080 is shown in Figure 22.

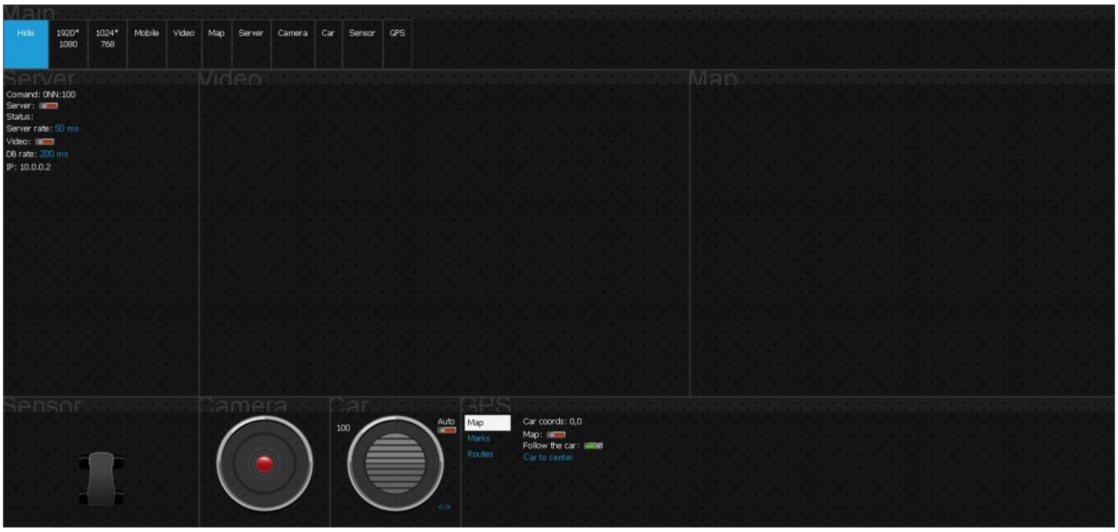

Figure 22. Screen mode 1900*1080

For more common monitors there is a mode with resolution 1024*768 it is presented in Figure 23.

Figure 23. Screen mode 1024*768

For mobile devices there is a mode which is oriented on the car control without using a map. It is shown in Figure 24.

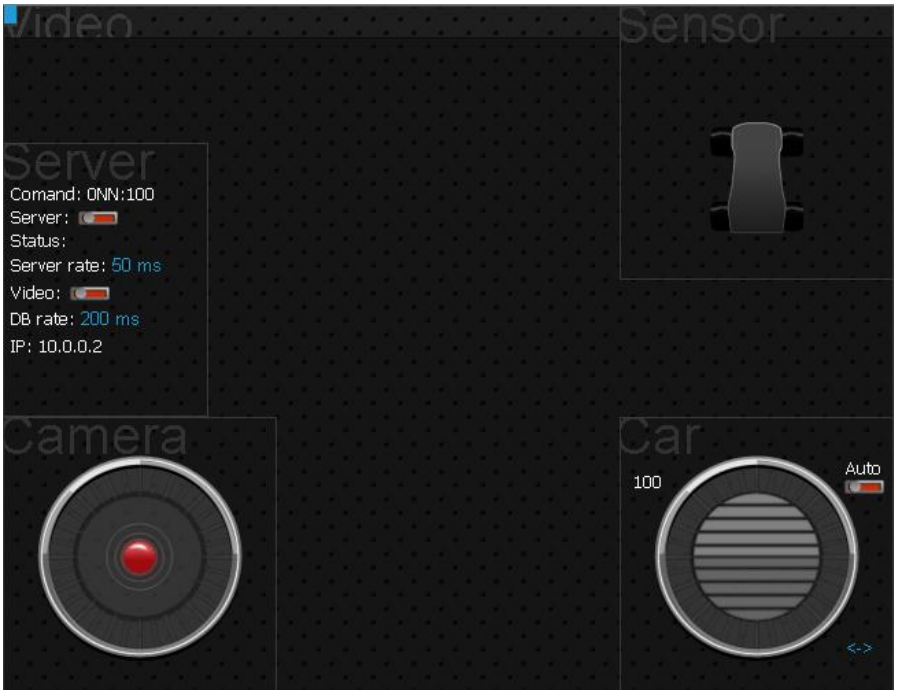

Figure 24. Screen mode Mobile

When a user chooses a mode from default ones the function *goToMode()* is called. It checks whether a similar process has been already started up. If it is not so it starts to change the mode and calls the function *getJson()* which loads the configuration of new mode. Then the configuration is loaded and the function *updatePanelPosition()* is called for each panel. This function changes a panel position slowly by using special algorithm. A part of algorithm for horizontal moving of a panel is show below.

```
if(document.getElementById(Id).offsetLeft!=X)
{
if(document.getElementById(Id).offsetLeft>X)
{
if((document.getElementById(Id).offsetLeft-
X)>50)document.getElementById(Id).style.left=document.getElementByI
d(Id).offsetLeft-30;
elseif((document.getElementById(Id).offsetLeft-
X)>10)document.getElementById(Id).style.left=document.getElementByI
d(Id).offsetLeft-10;
elsedocument.getElementById(Id).style.left=document.getElementById(
Id).offsetLeft-1;
}
else
{
if((X-
document.getElementById(Id).offsetLeft)>50)document.getElementById(
Id).style.left=document.getElementById(Id).offsetLeft+30;
elseif((X-
document.getElementById(Id).offsetLeft)>10)document.getElementById(
Id).style.left=document.getElementById(Id).offsetLeft+10;
elsedocument.getElementById(Id).style.left=document.getElementById(
Id).offsetLeft+1;
}
}
```

Similar code is executed for vertical movement at the same time. When all panels are on the right places the function *setPanelVisibility()* sets a new visibility to each panel.

5.1.7Control elements

For the car and camera controls there are special designed mechanisms or dynamic user interface elements. They give a user a possibility to observe a reaction of control interface to user events. When a user is controlling the car and pressing on the keys the interface elements are moving.

The car control element consists of52 images. Each of them represents one state of the control mechanism. All images are loaded before they are used in process of page

loadingas it is shown below.Otherwise they would be loaded in process of controlling and it would make delays.

```
varcontrol_scale_up=newImage();
control_scale_up.src="images/control/control_scale_up.png";
```

There is a function *refreshControlPanel()* which repeats each 30 milliseconds. It checks pressed keys and refreshes control elements. This function is started by timer when the page is loaded.

```
control_timer=setInterval("refreshControlPanel();",30);
```

For a control element there is a *control_counter* which is increased if the right arrow key is pressed and decreased if the left one is. If neither rightor left keys are pressed then the counter goes to 0 i.e to thedefault stateand the control element is in default position as it is shown in Figure 25 a. Also Figure 25 shows all nine states of control interface for right turn.

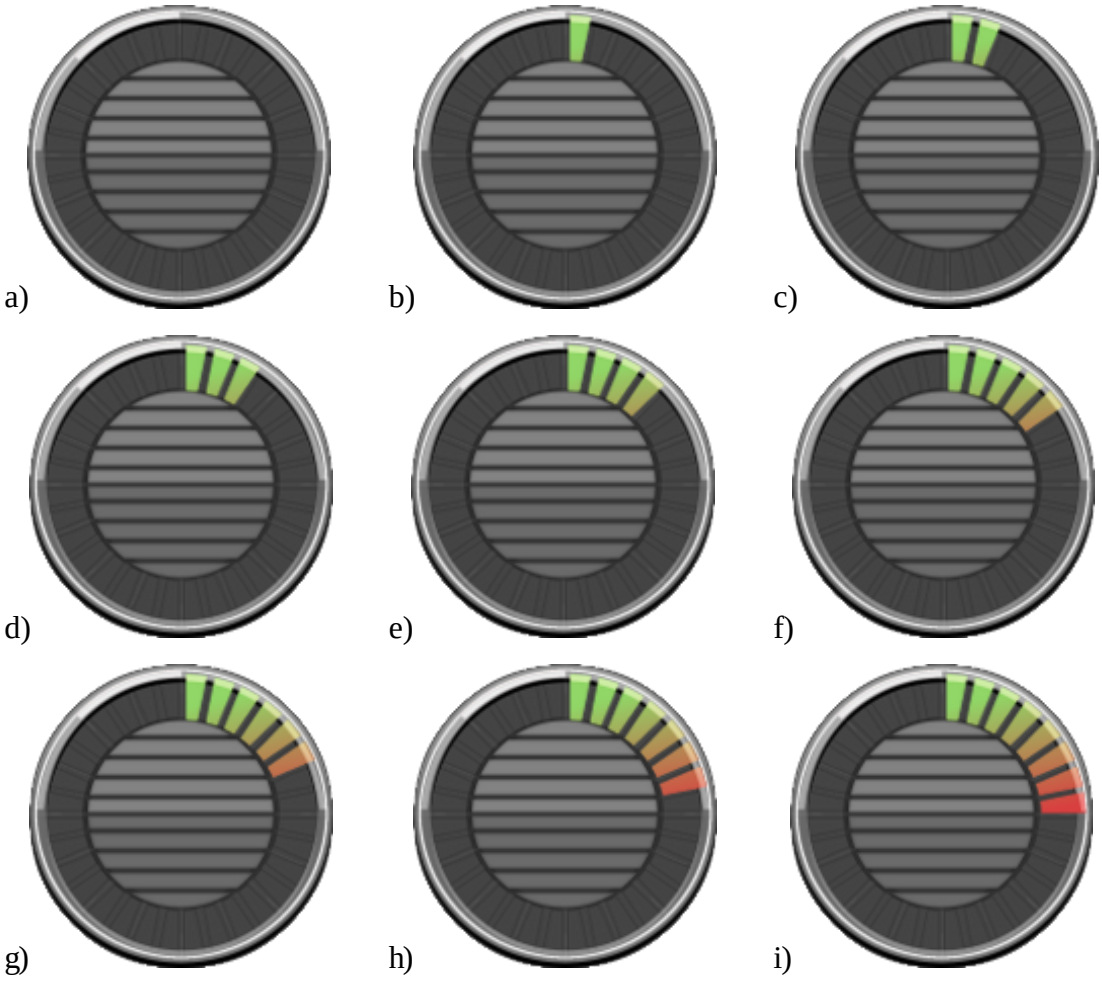

Figure 25. Right turn of car control states

The same states there are for left turn. When the user is moving the car back the control element shows similar right or left turns in lower part of control element.

The function checks value of the counter and loads the proper image. Background is the same for all sates therefore only green scale is changed. For example for image h in Figure 25the changing code is shown below.

```
document.getElementById("car_control_left_right").style.backgroundI
mage="url("+control_scale_right_7.src+")";
```

The same way the function checks a gear and loads a proper image. It changes only red scale. All gear states are shown in Figure 26.

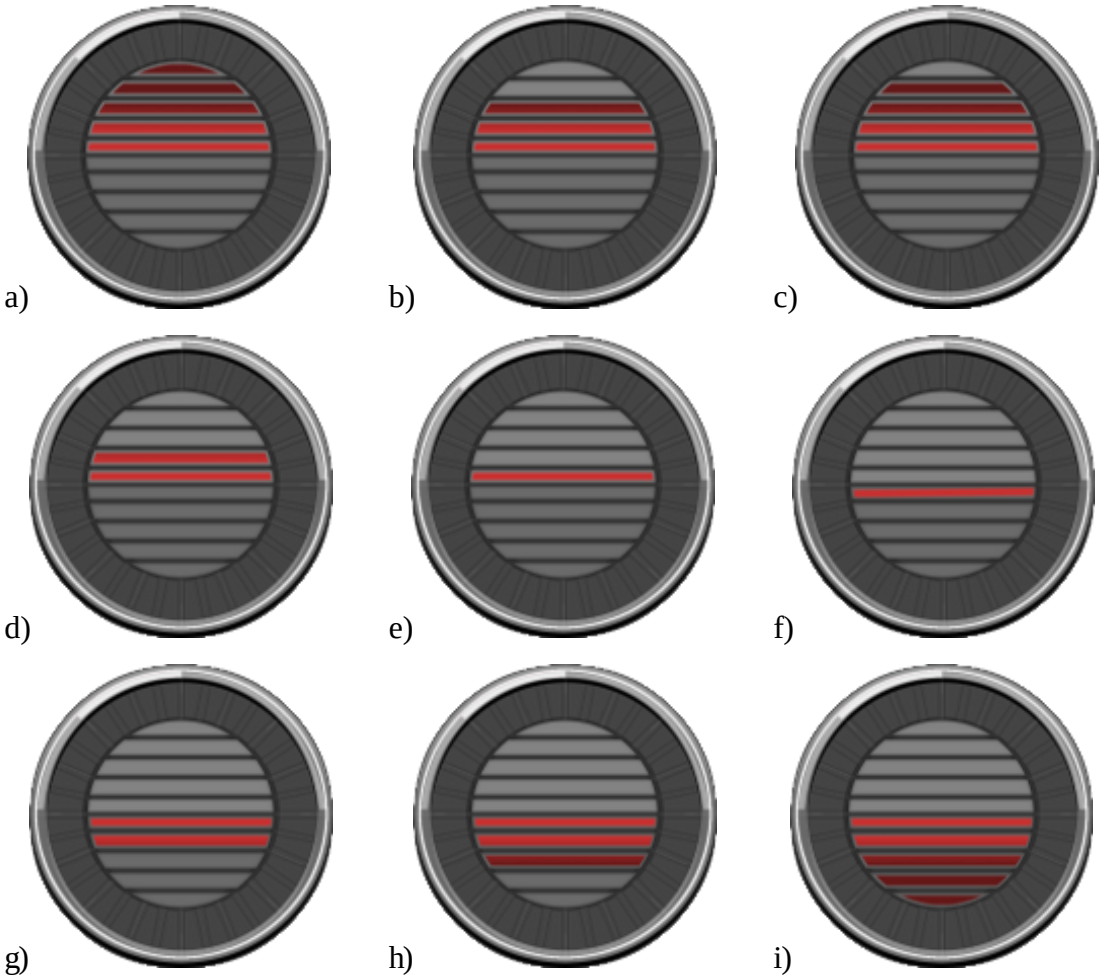

Figure 26. Gear control states

There are five forwards gears and five backward gears. In Figure 26picture ais a themaximum forward gear, e is the first forward gear, f is the first backward gear and i is the maximum backward gear.

Auto gear possibility allowsto control the car without switching on the keyboard and it gives a user all 200 possible speeds. If auto gear is turn on *auto_counter* is changed as it is shown below.

```
if(up&auto_counter<0&!bs)auto_counter=0;
if(up&auto_counter<25&!bs)auto_counter=25;
if(dp&auto_counter>0&!bs)auto_counter=0;
if(!bs)
{
if(up&auto_counter<100)auto_counter++;
elseif(dp&auto_counter>-98)auto_counter--;
elseif(auto_counter>0)auto_counter--;
elseif(auto_counter<0)auto_counter++;
gear=auto_counter+100;
speed=gear;
}else
{
if(up&auto_counter<100)auto_counter++;
elseif(dp&auto_counter>-98)auto_counter--;
}
gear=autocount+100;
```

The function increases the counter if the user holds a pressed up arrow key and decreases it if down arrow key. If the user does not press up or down keys the counter goes to 0.

The user can hold one speed or change it smoothly by holding “Space” key. If the key is pressed the function increases or decreases the counter by one only once. To increase or decrease more the user has to releaseup or down button and press it again. If no one from up or down keys is pressed the function holds current gear. It allows a user to choose easily necessary speed and move the car with this permanent speed more smoothly.

The camera control element is managed the same function. If the user presses camera movement keys the function detects this action by similar way and changes the image of the camera control element. All states of camera control element are shown in Figure 27.

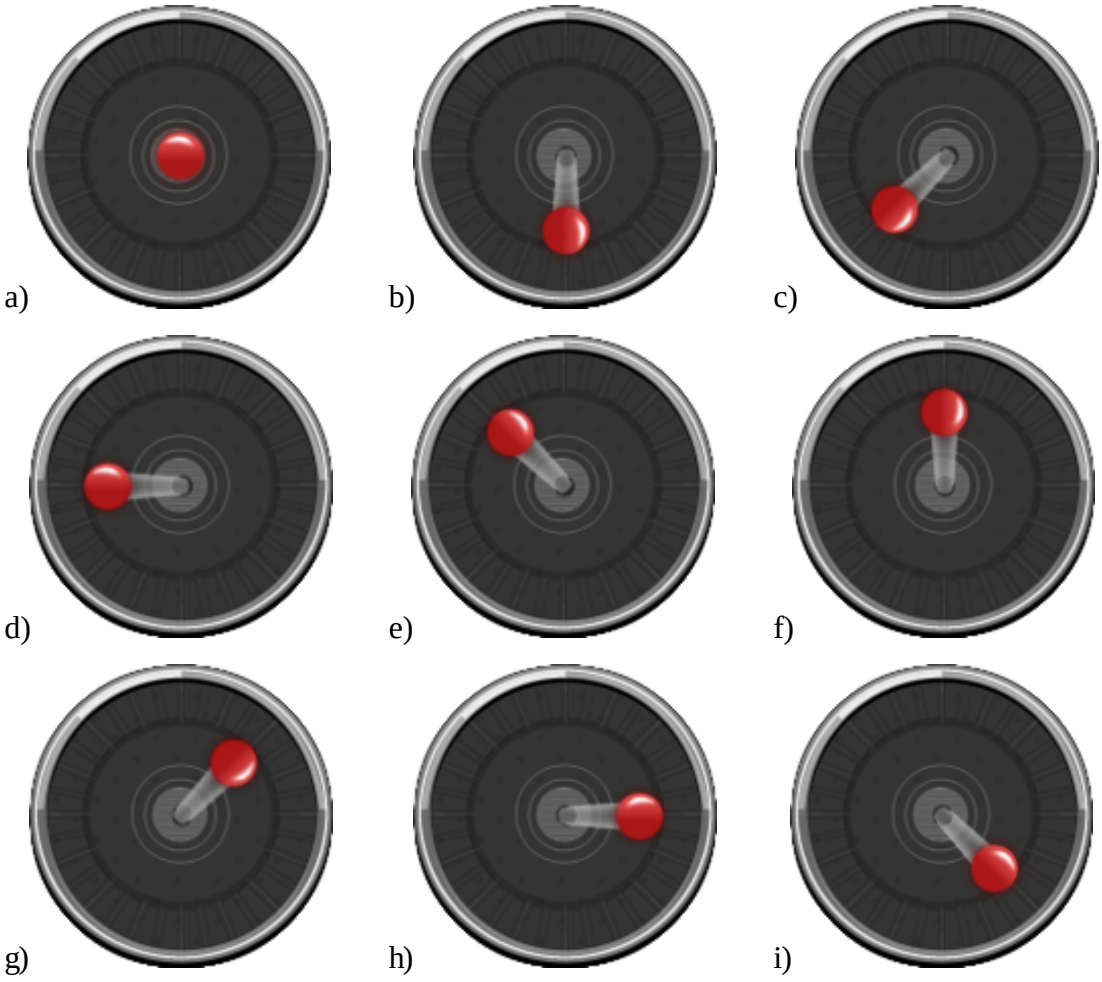

Figure 27. Camera control element states

This function *refreshControlPanel()* is repeated quite often thereby the user cannot detect this delay. In the end the function calls*makeCommand()* which is responsible for creating a command to be sent to the car. This function checks current values of gear, pressed buttons and generate the control command for transferring to the server.

5.2Data transferring algorithms

5.2.1 Command packet transferring

There are few stages of the command packet transferring. The command which is sent to the server is stored on the client. A function *makeCommand()* is responsible for creating this command. This function is called each several milliseconds by function *refreshControlPanel()* or when some user event has happened. Thereby the command is refreshed more often than the client sends it to the server.

The system has a value which defines a rate of the command packet transferring. Default value of it is 50 milliseconds and it can be changed by a user. Thereby each several milliseconds function *sendCommand()*is executed. This function makes an Ajax request to the server part of the system.

```
function sendCommand()
{
url="servers/control_server.jsp?st="+encodeURIComponent(comm_to_ser
ver)+"&ip="+encodeURIComponent(server_ip);
xhr.open("GET",url,true);
xhr.onreadystatechange=updateStatus;
xhr.send(null);
}
```

The *xhr* is a*XMLHttpRequest* object which is created when the page is loadedas it is shown below.

```
try
{
xhr=newXMLHttpRequest();
}
catch(trymicrosoft)
{
try
{
xhr=newActiveXObject("Msxml2.XMLHTTP");
}
catch(othermicrosoft)
{
try
{
xhr=newActiveXObject("Microsoft.XMLHTTP");
}
catch(failed)
{
xhr=false;
}
}
}
if(xhr==false)
{
alert("Ajax Object cannot be created! Please, try another
browser.");
}
```

The function *sendCommand()* makes a request and sends it to the script file *control_server.jsp* which is located on the web server. The request contains only two parameters. The first one is the command packet and the second one is IP address of the server. The server can be changed. For that reason IP address cannot be hard-coded into jsp script, it cannot store IP locally and the script cannot spend time to get IP from file or other outside resources. Thereby the system has to send IP every time with command packet. The script gets the request and handles it. The code of this script is shown below.

```
<%@pageimport="java.io.IOException"%>
<%@pagecontentType="text/html"pageEncoding="UTF-
8"import="java.net.Socket,java.io.OutputStream,java.io.InputStream"
%>
<%
Stringcom_pack=request.getParameter("st");
Stringip_add=request.getParameter("ip");
try
{
Socketsock=newSocket(ip_add,17777);
OutputStreamostream=sock.getOutputStream();
ostream.write(com_pack.getBytes());
ostream.flush();
sock.close();
out.println("ok");
}
catch(IOExceptionioe)
{
out.println(er);
}
```

Whenjsp script is executed it makesa connection througha TCP\IP socket and sends the command packet to the Com server with the received IP address. The script detects a state of the connection and sends a response back to the client. The function *updateStatus()* receives the response from the server and shows a user a successful result or error of that transferring. Errorscan occur. But if it happens so often it means that the server or network cannot handle each request and user has to reduce the transferring rate.

```
functionupdateStatus()
{
if(x.readyState==4)
{
vartext=x.responseText;
document.getElementById("server_status").innerHTML=text;
if(x.status==200)
{}
}
}
```

As an alternative of TCP connection it was made UDP connection. Comparing the TCP with the UDP connection, I found out that UDP performs the command transferring faster but it does not send the result of this process. And the user does not know that transferring was successful or not.

```
<%@page import="java.net"%>
<%@pagecontentType="text/html"pageEncoding="UTF-8"%>
<%
Stringcom_pack=request.getParameter("st");
Stringip_add=request.getParameter("ip");
DatagramSocketclientSocket=newDatagramSocket();
InetAddressIPAddress=InetAddress.getByName(ip_add);
DatagramPacketsendPacket=newDatagramPacket(com_pack.getBytes(),
com_pack.getBytes().length, IPAddress, 34444);
clientSocket.send(sendPacket);
out.println("ok");
clientSocket.close();
%>
```

5.2.2 Writing and reading of configuration files

When a user loads a control user interface or changes its configuration the system reads an existent configuration from the config file or writes a new one there. For reading the configuration files there is function *getJson()*.

```
functiongetJson(mode)
{
varparams="action=getconfig&username="+user_name+"&mode="+mode;
json_request.open("POST","servers/json_server.jsp",true);
json_request.setRequestHeader("Content-type","application/x-www-
form-urlencoded");
json_request.setRequestHeader("Content-length",params.length);
json_request.setRequestHeader("Connection","close");
json_request.onreadystatechange=jsonReply;
json_request.send(params);
}
```

This function makes anAjax request with action, username and modeparameters. Action contains an instruction what to do for the server script. The user name is necessary because the system can be used by several users.The mode defines which screen mode has to be used. Then the request is sent to the server *json_server.jsp*script executes it.

```
Stringuser_name=request.getParameter("username");
Stringmode=request.getParameter("mode");
try
{
Stringfile_name="";
if(mode=="")
{
file_name=request.getRealPath("configs/"+user_name+".json");
}elseif(mode.equals("1024"))
{
file_name=request.getRealPath("configs/json_default_1024.json");
}elseif(mode.equals("800"))
{
file_name=request.getRealPath("configs/json_default_mobile.json");
}elseif(mode.equals("1800"))
{
file_name=request.getRealPath("configs/json_default.json");
}
FileInputStreamfis=newFileInputStream(file_name);
BufferedInputStreambuf=newBufferedInputStream(fis);
intch;
while((ch=buf.read())!=-1)
{
out.print((char)ch);
}
fis.close();
}
catch(Exceptione)
{
out.println("error:"+e.getMessage());
}
```

This code contains the Ajax request. At first it checks delivered parameters and detects necessary name of the configuration file. Then it reads the file through file stream and displays whole data from it.

Function *jsonReply()*receives the reply from the server i.e. configuration data in string format and transforms it by function eval() from text to JSON object as it is shown below.

```
vartext=json_request.responseText;
json_object=eval("("+text+")");
```

Then this JSON object can be used to set a configuration of the user interface. The same scheme is used to save configuration to config. For that Ajax request contains the current configuration as it is shown below.

```
varparams="action=setconfig&username=admin&jsonobj="+encodeURICompo
nent(jsonString);
```

Server *json_server.jsp*script defines a file name and writes configuration data to that configuration file by similar way.

```
Stringfile_name=request.getRealPath("configs/"+user_name+".json");
FileOutputStreamfos=newFileOutputStream(file_name);
PrintWriterpw=newPrintWriter(fos);
pw.println(json_data);
pw.close();
fos.close();
```

Thereby Ajax and JSON technologies allowto implement a using of configuration files for different users.

5.2.3 Video applet

To add the applet on the page there is function which executes the following code.

```
document.getElementById('video_applet').innerHTML="<applet
code='source.robo_pla' archive='pla_apl.jar,jmf.jar' width='640'
height='480'/>";
```

The *source.robo_pla* is a main class which describes the video applet. This class is located in archive *pla_apl.jar*which is loaded from the web server. Also *jmf.jar* archive is loaded. It contains necessary libraries for the video applet. The main class is *robo_pla*. It has the main method *init()* which initializes the applet.

```
publicvoidinit()
{
JPanelmain_win=newJPanel(newGridLayout(0,3));
main_win.setPreferredSize(new Dimension(320,240));
main_win.setLayout(newFlowLayout());
        Component comp_video= null;
MediaLocator ml =newMediaLocator("rtp://localhost:3232/video/1");
try
{
robo_player=Manager.createRealizedPlayer(ml);
}
        . . . //catch different exceptions
if((comp_video=robo_player.getVisualComponent())!= null)
{
main_win.add(comp_video);
}
main_win.setOpaque(true);
setContentPane(main_win);
main_win.setVisible(true);
robo_player.start();
}
```

5.2.4 Realization of interaction with database

The client system uses the database to store information from sensors, GPS coordinates and routes data. For interaction with database there were made special API functions which are located on the server part.

Script *db_getdata_api_v1.jsp* is responsible for getting data from the database. The request to database looks like it is shown below.

```
varurl="http://"+db_ip+"/esms/db_getdata_api_v1.jsp?db_data=route&v
alue="+id;
```

The*db_data* is a parameter of the request which defines the action or type of necessary data. The *value* is additional parameter which contains data as a descriptor. In Table 6 there are described possible values of*db_data* parameter.

Table 6. Database API parameters

Value	*Description*	*Addition parameters*
sensors_last	To get the last value of data from sensors and GPS coordinates.	
route_list	To get whole list of stored routes.	
route	To get coordinates of route points by route's id.	*value* – route's id

To send the request it is necessary to create a script element with a formed url and put it to the head of the page. When the element is added it immediately sends the request to the API function.

```
var script =document.createElement('script');
script.setAttribute('src',url);
document.getElementsByTagName('head')[0].appendChild(script);
document.getElementsByTagName('head')[0].removeChild(script);
```

The process of data request happens few times per second and each time a new script element is added to head of the page. To prevent overflow of head the script element is removed after adding.

The server sends data back after processing of the client request. The reply contains a name of the function with a parameter in json format. When the client receives the data from the server it starts to execute function which is contained in the reply and which is described on the client. For getting sensors data the replay contains function *jsonDATA()*.

```
function jsonDATA(JSON)
{
json_data= JSON;
showSensorData();
updateCarPosition();
if(autoFollowingTheRoute)goNextPoint();
}
```

The function *showSensorData()* execute visualization of the sensors. The function *updateCarPosition()*updatesthe car position on the map with new GPS data. And *goNextPoint()* is a function of auto control algorithm.

For getting alist ofthe routes the replay sends *jsonROUTE()* function with type *route_list*. And for getting a route with coordinate points the replay sends also *jsonROUTE()* function with type*route_table*.

```
function jsonROUTE(JSON)
{
if(JSON.table.table_info.type=="route_list")
{
makeRouteTable(JSON);
}elseif(JSON.table.table_info.type=="route_table")
{
json_route= JSON;
setRouteIntoMap();
}
}
```

The function *makeRouteTable()* generates a table of the saved routes. An function *setRouteIntoMap()* shows the selected route on the map.

5.3 Mapping

5.3.1Map initialisation

To set up the map on the page there is a following sequence of actions. The map is an object which is created by using function *GMap2()*. The map is put into*map_canvas* container which is located on the Map panel.It sets default parameters on the map as a position, type and a scale.

```
map=new GMap2(document.getElementById("map_canvas"));
map.setCenter(getCarPosition(),13);
map.setUIToDefault();
```

Function *getCarPosition()* defines the current car position. It takes latitude and longitude from json_data object which stores last coordinates of the car.

```
car_position=new GLatLng(convertLatCoord(json_data.table.rows[0].lat
),convertLngCoord(json_data.table.rows[0].lng));
```

So as the GPS coordinates and Google Maps use different formats of coordinates there is a function*convertLatLngCoord()*to convert coordinate to appropriate format for using on the map. The function convertsGPS data as it is shown below.

```
str=coords.toString();
dot_index=str.indexOf(".",0);
h=str.substr(0,dot_index-2);
m=str.substr(dot_index-2,str.length-dot_index+2);
rez=parseFloat(h)+parseFloat(m/60);
```

The map has an event listener which detects user's clicks and puts markers on the map. Thereby when a user clicks on the map he sets a new marker. The even listener is described below.

```
GEvent.addListener(map,"click",function(overlay,latlng)
{
if(latlng)
{
if(allow_put_marks)
{
varpoint=newGLatLng(latlng.lat(),latlng.lng());
map.addOverlay(createMarker(point,++map_objs.counter));
if(route==null)
{
route=newGPolyline([point],"#1F9CD6",3);
map.addOverlay(route);
drawLine(0);
}
drawLine(map_objs.counter);
}
}
});
```

This function also makes the main car marker through function *createMarker()*. The car marker displays the current car position on the map.

5.3.2 Markers

To show car position and route points on the map the Google map uses markers. To create and put a marker on the map there is function *createMarker()* which is shown below.

```
vardot_icon=newGIcon(G_DEFAULT_ICON);
varmarkerOptions=null;
if(number==0)
{
dot_icon.image="http://google-maps-
icons.googlecode.com/files/military.png";
markerOptions={icon:dot_icon,draggable:false};
}else
{
varicon_number="00";
if(number<10)icon_number="0"+number;
elseicon_number=number;
dot_icon.image="http://google-maps-
icons.googlecode.com/files/red"+icon_number+".png";
markerOptions={icon:dot_icon,draggable:true};
}
```

The function makes a marker object by using number of the marker and its coordinates. It chooses an icon for a marker. For thecar there is a special marker. For route points the system uses default markers with numberof the point from 1 to 100 which are shown in Figure 28.

Figure 28. Icons for markers

The marker is made by function *GMarker()*.

```
GEvent.addListener(marker,"click",function()
{
updateMarkersTable(number);
});
GEvent.addListener(marker,"dragstart",function()
{});
GEvent.addListener(marker,"dragend",function()
{
reDrawRouterPoint(marker.value);
});
```

Each marker except the car marker can be moved or clicked. Proper events are defined. Information about marker objects is saved locally.

5.3.3 Routes

A user can make a route from one point to another. The route is a sequence of markers connected by lines. To draw the line between two point there is a function *drawLine()*. This function inserts vertexes to the route object and puts it on the map through *addOverlay()* function.

```
function drawLine(id)
{
if(id==0)
{
route.insertVertex(0,getCarPosition());
map.addOverlay(route);
}else
{
var lat=map_objs.markers[id-1].getPoint().lat();
var lng=map_objs.markers[id-1].getPoint().lng();
var point=new GLatLng(lat,lng);
route.insertVertex(id,point);
map.addOverlay(route);
route.deleteVertex(id+1);
}
}
```

Information about markers of the route which is drawn on the map is displayed on the GPS panel in the form of a table. The markers table is shown in Figure 29.

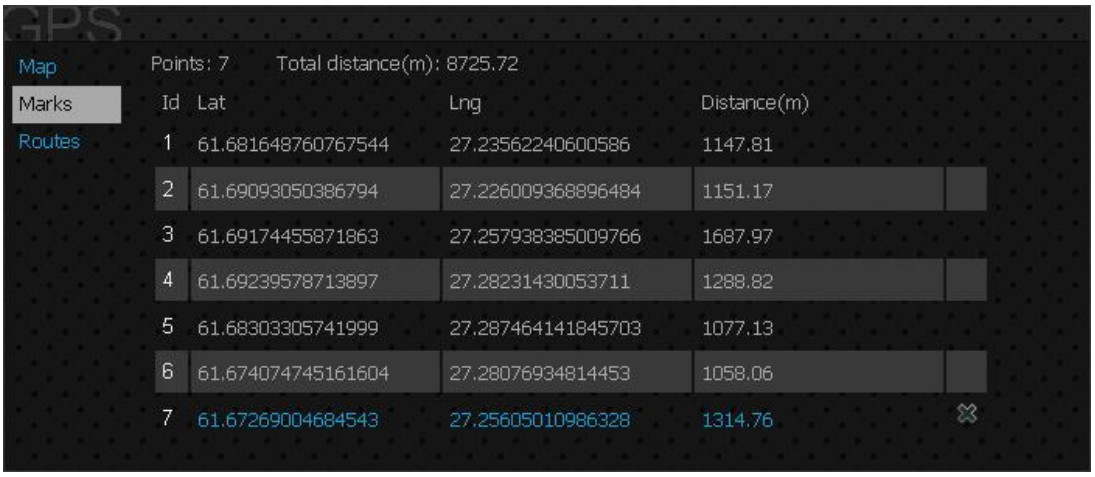

Figure 29. Markers table

A user can see all points in the table and their coordinates. Also there is a possibility to move any marker to another position and accordingly to change the route. When the marker is moved bythe user the line of the route is redrawn through the function *reDrawRouterPoint()*. Figure 30 shows how the user can make a route.

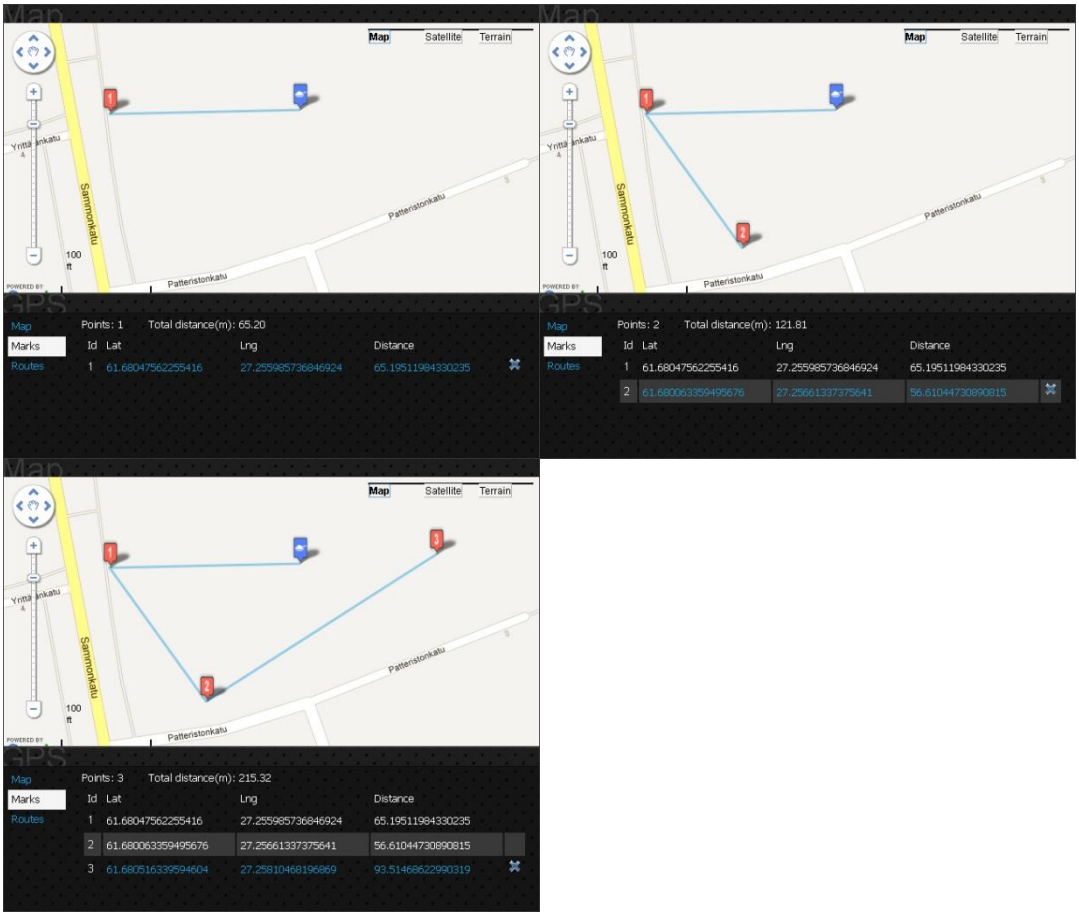

Figure 30. Drawing the route

The user can choose any marker except the car marker and remove it. When a marker is deleted by function *removeMarker()* the route is redrawn. Also the user can see distance of the route. To get distance between two points there is a function distanceFrom(). Thereby distance of the route is calculated by the system every time when it is changed.

```
vardstc=point.distanceFrom(map_objs.markers[number-2].getPoint());
```

Any user's route can be saved to the database on the server. When the user saves the route he enters route's name and calls function *saveCurrentRoute()* by clicking a proper button. After that the route is appeared in table of routes.

```
functioninsertRoute(route_str,dist,points,name)
{
varurl="http://"+db_ip+"/esms/db_moddata_api_v1.jsp?db_insert=route
&value="+route_str+";"+dist+","+points+","+name;
var script =document.createElement('script');
script.setAttribute('src',url);
document.getElementsByTagName('head')[0].appendChild(script);
document.getElementsByTagName('head')[0].removeChild(script);
window.setTimeout("getRouteTable();",1000);
}
```

The table of routes contains all user's routes which have been saved. From this table the user can see route's name, amount of points, a total distance of the route and date of the route creation. The table of routes is made the same style such as the table of markers.

The user can easily choose and load on the map any route by clicking a proper row in the table. For loading a route the system makes a request to the server by following way.

```
functiongetRouteData(id)
{
varurl="http://"+db_ip+"/esms/db_getdata_api_v1.jsp?db_data=route&v
alue="+id;
varscript=document.createElement('script');
script.setAttribute('src',url);
document.getElementsByTagName('head')[0].appendChild(script);
document.getElementsByTagName('head')[0].removeChild(script);
}
```

When the client gets data about route it cleans the map and creates new route by function *setRouteIntoMap()*.To clean the map there is a function *clearOverlays()*.

```
functionsetRouteIntoMap()
{
if(map!=null)
{
map.clearOverlays();
map_objs.markers=[];
map_objs.distance=[];
map_objs.counter=0;
route=null;
map.addOverlay(createMarker(getCarPosition(),0));
for(vari=1;I <=json_route.table.table_info.len;i++)
{
varpoint=newGLatLng(json_route.table.rows[i-
1].LAT,json_route.table.rows[i-1].LONG);
map_objs.counter=i;
map.addOverlay(createMarker(point,map_objs.counter));
if(route==null)
{
route=newGPolyline([point],"#1F9CD6",3);
map.addOverlay(route);
drawLine(0);
}
drawLine(map_objs.counter);
}
}
}
```

5.3.4 Control the car by map

When a user has fixed a route or loaded it from the database and switched on the autonomy mode the system starts to execute this function. Each several milliseconds (time interval can be set by a user) the system gets GPS data from the car and executesa function *goNextPoint()*.

The function determines a current car position and calculates the direction which the car has to follow. To calculate direction there is a function*getBearing()*.

```
function getBearing(lat2,lon2,lat1,lon1)
{
var dLon =(lon2 - lon1);
var y = Math.sin(dLon)* Math.cos(lat2);
var x = Math.cos(lat1)* Math.sin(lat2)-
Math.sin(lat1)*Math.cos(lat2)* Math.cos(dLon);
return(Math.atan2(y, x)*180/Math.PI)+180;
}
```

The system compares the current direction and calculated one and makes a decision to turn the car or not. Then it emulates user actions and generates the control command for the car to move it.

The function determines the hit. Ifthe current coordinates satisfy thecondition the function marks current point as a hit. If it was last point the system stop the car and turn off the auto navigation. If there is next point of route the system takes that one and repeat calculation between current and next points before it will not be achieved.

5.4 Data visualization

Visualization allows a user to analyze environment around the car through ultra-sonic sensors. They are located on the front, left and right sides of the car. Thereby data from sensors give a possibility to know the distance between the car and obstacle from three direction. These data are written every 150 milliseconds to database by the server. And theclient can see updated data from sensors almost every 150 milliseconds.

The visualization corresponds a sketched picture of the car with sensors. Distance from each sensor to obstacle is shown as a wave which is long if the distance is long and short if the distance is short.The wave has 5 type of length from 1 to 5. Description of each type is presented in Table 7.

Table 7. Wave types

Wave length	*Distance to* obstacle	*Color of wave*
1	<10 cm	Red
2	10-20 cm	Red
3	20-50 cm	Yellow
4	50-100 cm	Yellow
5	>100 cm	Green

In the user interface each type of the wave has a certain number of lines from 1 to 5 and proper color. Figure 31 shows an example ofvisualisation with three different types of the wave.

Figure 31. Car visualization

By this visualisation a user can analyze environment around the car. Also in front of each sensors user can see a digit value of the distance in centimetres.

When the system gets GPS data from the database it also gets data from the sensors. To update the visualization with new data there is a function *showSensorData()*. This function checks received values and analyses the presence of the errors. If data are correct the function implements these data for each sensor as it is shown for top sensor below.

```
if(json_data.table.rows[0].ULTRA_F!=0&&json_data.table.rows[0].ULTR
A_F!=777)sensor_top=json_data.table.rows[0].ULTRA_F;
if(sensor_top<10)document.getElementById("top_sensor_wave").style.b
ackgroundImage="url("+sensor_top_1.src+")";
elseif(sensor_top<20)document.getElementById("top_sensor_wave").sty
le.backgroundImage="url("+sensor_top_2.src+")";
elseif(sensor_top<50)document.getElementById("top_sensor_wave").sty
le.backgroundImage="url("+sensor_top_3.src+")";
elseif(sensor_top<100)document.getElementById("top_sensor_wave").st
yle.backgroundImage="url("+sensor_top_4.src+")";
elsedocument.getElementById("top_sensor_wave").style.backgroundImag
e="url("+sensor_top_5.src+")";
document.getElementById("front_sensor_num").innerHTML=sensor_top;
```

If data from sensors do not contain 0 or 777 (error) then these data are shown on the page. To visualize the data the system changes a picture of the sensor wave and digital value near that. The same way is used for each sensor. All images of the wave states are preloaded when the system is loaded.

6 TESTING

For checking any system there are common tests. For this web control system it is necessary to check the user interface with different platforms, connections with the database, video and com servers, control algorithms and work of the system as a whole.

6.1 Test of user interface and control elements

At first it was made common test of the user interface. The test checks a possibility to load the websystem, working of panels, their functionalities and also possibility of the car control.This test was done on the common PCwith different web browsers. Web browsers were used the most popular ones.Results of the test are shown in Table 8.

Table 8. User interface test

Web browser	*Loading*	*Design*	*Control possibility*	*All functions*
Firefox 10.0.2	ok	ok	ok	ok
Chrome 19.0.1061	ok	ok	ok	ok
Opera 11.61	ok	ok	ok	ok
IE 9	fail	-	-	-

The system works in all most popular web browsers very good without any bugs and errors except in Internet Explorer. It does not work in Internet Explorer because this browser interprets some JavaScript functions not correct. Thereby this browser is not recommended to use.

The user interface looks very modern. It is dynamic and flexible. It can be configured in concordance with user’s needs for different screen resolutions. The interface has an appropriate color box which does not innervate a user and user’s attention is directed at the control elements.

Elements of the car and camera control are modern, dynamic and informative. They allowa user to understand control mechanismsvery clearly.With this user interface it is easy to orientate and control the car.

6.2 Speed test of data transferring

For this kind of system it is important to have suitable rate of data transfer. Car's reactionsand convenience of control dependon delays anddata rate. In the speed test the control data transferring rate from the client to the server and back from the server to the client is measured.Also time of user's action handling on the client is analysed.

The test was made with different types of devices. First device is a quite powerful modern PC with high performance. Second one is anetbook which can be used for internet surfing. It is not so powerful and quite old. And third one is a common mobile phone which also supports a web browser with JavaScript. Testing took place inside one network because the delay of data transferring through the local network is insignificant about 1-2 milliseconds. It may not be taken into account. For a wide network the delay can be much higher dependingon the user location and transmission channels. It was made about 100test iterations for each device. Results of testing are presented in Table 9.

Table 9. Data transferring test

Device	*Time of user's action handling (ms)*	*Rate of control data transferring (ms)*	*Rate of database data transferring*
Powerful PC	20	2	20
Netbook	30	2	25
Mobile phone	30	2	25

With the powerful PC it is possible to send control data to the server every 20 milliseconds.It means that each 20 milliseconds system can determine user's action, update the command for the car and send it. This handling of user actions is so fast that the user cannot recognize this delay. Alsoittakesabout 2 milliseconds for command data transferring through the network. The rate of the system meets all requirements.

The netbook and the mobile phone need a little bit more time to handle user's action about 30 milliseconds because JavaScript is executed longer on the less power devices.But still this time is quite short and it also meets all requirements.

The user can control the car from any kind of device even not so powerful and from any place through the Internet. With ping up to 50-70 milliseconds it is still possible to control the car normally. Also to reduce load on the device there is a possibility to change data rate or turn it off, and hide panels with graphic elements which are refreshed every 50 milliseconds and take the power.

The rate of data transferring from database for each device meets the requirement. It is more than enough to get data from database every 150 milliseconds because sensors make new measurements with 150 milliseconds intervals.

Also it was made a test of the video stream transfer. In result of the test it was found out that the video stream transferring from the camera to the user interface takes about 100-200 milliseconds with video resolution 320*240 and wide bandwidth of network. It is good result. With higher resolution it takes much more resources and delay might be more than 200 milliseconds. Also with narrow bandwidth of the network the video can have bigger delays.

6.3 GPS and map test

The first test of the GPS corresponds checking the accuracy of the GPS data and possibility to use it in different environments.The results of test are shown in Table 10.

Table 10. GPS test

Measurement	*Amount(m)*
Maximum inaccuracy in good weather	5
Average inaccuracy in good weather	2
Maximum inaccuracy in bad weather	30
Average inaccuracy in bad weather	10
Maximum inaccuracy inside the building	500
Average inaccuracy inside the building	50

In the result of the test it was found out that the GPS module has some limitations. It works incorrect inside the building and it can have inaccuracy up to few hundred metres or does not work at all. Average inaccuracy of the GPS data outside in good weather is about 2 meters. It is enough to detect the car on the city map and perform the missions with long distances which do not require accuracy less than a couple of meters.

The main idea of second test is checking the representation of the GPS data. The test showed that the system gets right GPS coordinates and represents them correctly on the map. Google Maps and coordinate converterworkedperfectly.

7 CONCLUSIONS

The web control system of mobile platform as a part of "Roboteh" project has shown that nowadays with modern technologies it is possible to develop a modern system to control the mobile platform from everywhere through Internet and observer the environment around it with a real time video, sensors visualisation and GPS navigation.

The aim of the project was development of web control system through which anybody can control the mobile platform by using common computer or mobile device with web browser and Internet connection. The aimwas reached. The web control system has a modern dynamic user interface with useful control mechanisms and possibilities. The user interface is cross-platform and configurable for different users and different devices. The system meets all speed requirements. It can handle user's events, transfer control data, process GPS data and sensors information from database very fast on the various kinds of devices with various web browsers. The system realizes a real-time video with minimum delay. It implements GPS navigation and auto navigation through Google Maps, visualization of the surrounding area with using data from sensors.

In process of developing of the project a lot of web technologies and technics were used. The main technologies are JavaScript, Ajax, JSON, JSP and Java Applet. They were chose because they give dynamics and very rapid data transferring what are main requirements for real time controlling through the web.

This project is good example of modern web control system. On base of the project it is possible to make similar system for any kind of remote controls. It may be car control, aircraft control or control system of static object as a house or company's office. And the user of the system can observe and manage the object from any were just by using web browser.

The project can be improved and extended. With using other sensors it is possible to realize other functionalities as more accurate mapping, a 2D or 3D scanning of environment or auto control with AI elements.

BIBLIOGRAPHY

Books

Bergsten, Hans 2004. JavaServer Pages.Third Edition. Sebastopol, CA. O'Reilly Media.

Cook, Todd 2002. Mastering JSP. Alameda, CA. Sybex.

Debbabi, Mourad 2007. Embedded Java Security: Security for Mobile Devices. London. Springer.

Gandy, Elizabeth – Stobart, Simon 2005. JavaScript: Creating Dynamic Web Pages. Colchester.LexdenPunlishing Limited.

Holdener, Anthony 2008. Ajax: The Definitive Guide. Sebastopol, CA. O'Reilly Media.

McIntire, Penny 2008. Visual Design for the modern Web. Berkeley, CA. New Riders.

Parsons, David 2008. Dynamic Web Application Development using XML and Java. London. Cengage Learning EMEA.

Progmore, Martyn 2008. An Introduction to Databases with Web Applications. Harlow. Pearson Education Limited.

Schildt, Herbert 2011. Java: A Beginner's Guide. 5 edition. McGraw-Hill Prof Med/Tech.

Thomas, A. Powell 2008. Ajax: The Complete Reference. McGraw-Hill Prof Med/Tech.

Vakali, Athena 2011. New Directions in Web Data management 1.Springer.

Zambon,Giulio – Sekler, Michael 2007. Beginning JSP, JSF, and Tomcat Web Development: from Novice to Professional. New York. Apress.

Electronic sources

Google Maps API Family. [referred 30.01.2012].Available in www-format:
<URL:http://code.google.com/intl/th-TH/apis/maps/index.html>.

JavaServer Pages. [referred 25.01.2012]. Available in www-format:
<URL:http://en.wikipedia.org/wiki/JavaServer_Pages>.

Welcome to Google Maps. [reffered 28.01.2012]. Available in www-format:
<URL:http://support.google.com/maps/bin/answer.py?hl=en&answer=144352>.

Web page.[referred 20.01.2012]. Available in www-format: <URL: http://en.wikipedia.org/wiki/Web_page>.