Czech Technical University in Prague
Faculty of Civil Engineering

and

Helsinki University of Technology
Faculty of Engineering and Architecture

DIPLOMA THESIS

User interfaces of UMN MapServer in web mapping applications

Prague and Helsinki, 2009 **Bc. Markéta Havlíčková**

Declaration

I declare that I have worked on my diploma thesis on my own, using only the documents (literature, projects, software, etc.) listed in the attached list.

Prague, Helsinki _____ _____

signature

Acknowledgement

I would like to thank everyone who have helped me to write this thesis, especially my supervisor, Ing. Jiří Cajthaml Ph.D. from Czech Technical University in Prague and my consultants of thesis Paula Ahonen-Rainio, D.Sc (Tech.) and Prof. Kirsi Virrantaus from Helsinki University of Technology.

Abstrakt

Diplomová práce je zaměřena na open source webovou mapovou aplikaci UMN MapServer a jeho uživatelská rozhraní jako je např. OpenLayers, ka-Map! a Fusion. Cílem práce je analyzovat a porovnat tato rozhraní. Porovnání bylo provedeno z hlediska funkčnosti, designu, dokumentace a doby načítání aplikace. V následující části práce bylo vybrané rozhraní použito pro konkrétní webovou mapovou aplikaci.

Klíčová slova: map server, framework, webová mapová aplikace, open source

Abstract

The thesis focuses on UMN MapServer, which is an open source web mapping application, and its user interface frameworks, such as OpenLayers, ka-Map!, and Fusion. The aim of the thesis is to analyze and compare these frameworks. The comparison embraces many aspects of web mapping applications, such as functionality, layout, documentation and application loading time. In the following part the selected user interface framework is to be used for a particular web mapping application.

Key words: map server, framework, web mapping application, open source

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In our lives we encounter on-line maps almost every day on various occasions, such as while searching the nearest restaurant or planning Sunday's trip. Behind the fact that we can get the results of our search on the screen within a few seconds, there are many hours of development, programming and finding the best solutions.

I have chosen the topic of my thesis primarily because I was interested in how the web mapping applications, which we use every day, actually work and how difficult it is to create such a web mapping application.

The web mapping applications are based on the common client-server concept, which describes a relationship between two programs: a client program (e.g., web browser) that sends a request to a server program that then executes the request and returns the requested information. In the case of web mapping applications the server should be able to work with geographic data and that is why it is necessary to install a map server as well. It is possible to choose from several map servers. There are open source solutions, such as UMN MapServer and GeoServer, and commercial map servers, such as MapGuide, ArcGIS server, and GeoMedia WebMap.

My thesis focuses on the open source UMN MapServer and in Chapter 2 it is outlined how the UMN MapServer works. MapServer enables to create web mapping applications by means of the CGI program, or they can be programmed using MapScript in one of the supported languages (PHP, Perl, Python, ...), which can sometimes be quite difficult. However, there are open source frameworks for user interfaces, including advanced functions, such as map controls, and their usage is much easier than programming these components yourself. In most cases these frameworks for user interfaces are JavaScript libraries that can be used to create user interfaces for web mapping applications. Since there are many different frameworks, I wanted to compare some of them and choose the

one, which is the best according to the criteria specified in this thesis.

By means of simple examples of web mapping applications I demonstrated the basic possibilities of every framework for user interface studied in thesis. These sample web mapping applications are displaying the OpenStreetMap of Europe distributed via web map service (WMS), and districts of the Czech republic in a shapefile format. All these examples are available on `http://maps.fsv.cvut.cz/~havlima6`.

Those frameworks for user interfaces which I selected for study are described in Chapter 3.

The user interfaces can be evaluated from two different perspectives: from the perspectives of the user, and the application developer who creates the web mapping application.

1. Users are mainly interested in how quickly and how easily they can receive the required information from the web mapping application and if it is possible to choose their favorite tool to control the map. This depends mainly on the application developers, whether and how they can use the functionalities of the framework so that data access is fast and convenient for users.

   The other user's criteria is, e.g., layout, which means what the application looks like as a whole, whether it is clearly and logically organized.

2. From the perspective of the developer of a web mapping application it is important to have a quality documentation of the framework which can help a lot, mainly when beginning to design the web mapping application. The contact with developers of the framework or other application developers can be important as well. When the application developer gets into an impasse, the framework developers can help him to solve the problem.

   Other very important parts of designing a web mapping application are the functionalities and their possibilities. The frameworks contain different amount of functions and each of them can offer various ways of usage. E.g., zooming can be done in several ways, such as a dragged rectangle, by a double click on the map and by zooming buttons but not every framework supports these possibilities and it is up to the application developer, which framework is chosen and if the final web mapping application is user-friendly.

   The examples of sample applications are also important for application developers, these may reveal other functions of the framework and their usage and can serve as a template for the new web mapping application.

The frameworks for user interfaces of web mapping applications were evaluated according to the criteria resulting from two perspectives mentioned above. The selection of these criteria was based on the author's experience with using web mapping applications. These criteria can be divided into two groups: the criteria which can be compared objectively, such as map controls and functionalities connected with layers (e.g., panning, zooming, printing and layer tree, measuring), and those which cannot be evaluated in an exact way (spread of the user interfaces, application loading time and documentation).

Zooming and panning are the basic tools for controlling the map and there are different ways of how they can be provided. Zooming can be provided by a dragged rectangle, zooming buttons, a scrolling mouse wheel, a double click or by a click on the slider of scale range. Panning can be provided by a click and drag, navigation buttons, a double click or a click into the map. Panning can also be continuous without reloading the page or not. Comparison and evaluation panning and zooming options is subject of a research [9].

The other function that should be included in user interface is the layer tree, which describes the order of layers, their name, content and map symbols.

The spread of the user interface cannot be measured in an exact way, because the user interface frameworks studied in this thesis are open source and anybody can use them. The only way is to try to find out, how the framework is discussed or how many web pages focus on that framework and count the number of websites, which can be estimated through a web search engine by using a suitable search request.

The application loading time is also an important criterion as the user's satisfaction with the user interface depends on how fast he or she gets the answer for the request. It is not easy to measure the application loading time in an exact way and it is important to ensure the equal conditions for every tested web mapping application during the measuring.

The documentation of high quality is an important part of the framework, but the quality level of the documentation cannot be measured objectively either.

The evaluation of frameworks for user interfaces can be found in Chapter 4.

In evaluation MapFish was selected as the most satisfactory framework for user interface. This framework was used for a web mapping application and its parts were described in more detail in Chapter 5. The web mapping application consists of the geospatial data of Finland. There are a base map of Europe which is more detailed for Finland and two thematic maps (Structure of built up areas and Areas of Natura 2000), which covers whole Finland. In the final web mapping application there were used basic functions, such as zooming by scale range, zooming by dragged rectangle, layer tree, where the

order of map layers can be changed, overview map, drawing tool and shortcuts.

In Chapter 6 the conclusions are given.

Chapter 2

UMN MapServer

UMN MapServer is an open source platform which is used for publishing dynamic maps and spatial data on the web sites. It was developed in 1990's at the University of Minnesota in cooperation with NASA and Minnesota Department of Natural resources. Nowadays UMN MapServer [20] is a project of Open Source Geospatial Foundation [8] and is released under an MIT-style license [19].

It is written in C. MapServer runs on various operation systems such as Windows, Linux, Mac OS. It is able to render hundreds of raster and vector data formats thanks to the GDAL and OGR libraries, and supports a lot of scripting languages such as PHP, Perl and Python. The latest version 5.4.2 is available since 24 July 2009.

2.1 The Process in Web Mapping Applications

Web mapping applications are based on the client-server concept. The difference between a usual web application and a web mapping application is the necessity of a server which can work with spatial data. In this case it is UMN MapServer. The process of the client-server concept in web mapping applications is described in six following steps:

1. The MapServer receives a request from the client (web browser).
2. It reads the mapfile (a file, where the map layers are described).
3. It draws and saves the map according to the definitions in the mapfile.
4. It reads the HTML template and replaces the substitution strings with the current value.

5. It sends the rendered image map to the web server.

6. The web server sends data to the web browser of the end user.

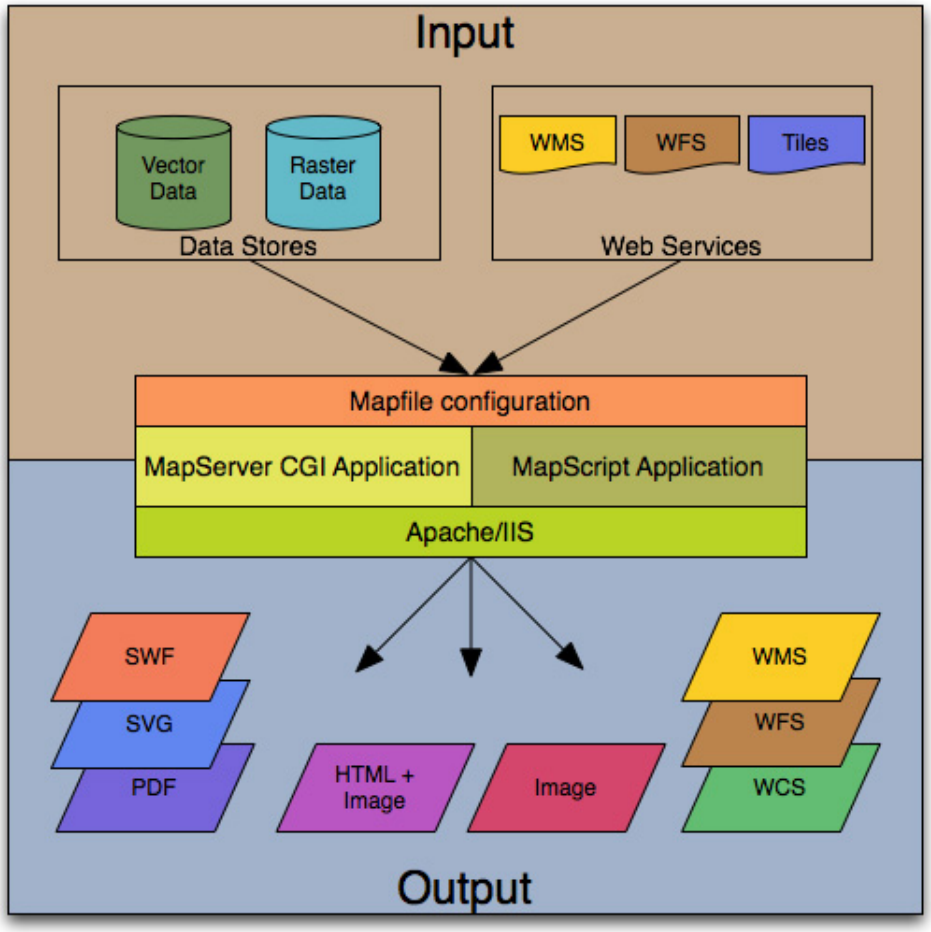

Figure 2.1: Architecture of MapServer application, Source: `http://mapserver.org/_images/architecture.png`

2.2 Components of Simple MapServer Applications

The simple web mapping application consists of components which are described below and are shown in Figure 2.1:

- **Geodata** - MapServer supports vector (e.g., shapefile, ArcSDE) or raster (e.g.,

GeoTIFF, GIF, JPEG) data, WMS, WFS. The default data format is shapefile from ESRI.

  - WMS - Web Map Service - a standard protocol for serving map images via the Internet. These images are generated by the map server using various data.
  - WFS - Web Feature Service - provides an interface allowing requests for geographic features via the Internet.

- **Mapfile** - XML-like structured configuration file, which ends with postfix .map and contains settings of the output design of resulting map image such as map extent, layers, symbology, projections, scale.
- **MapServer CGI** (Common Gateway Interface) - The program, usually called mapserv, receives the request from the web server and sends back the appropriate output image.
- **Web server** - Provides communication between the MapServer and the web browser, and stores HTML files.
- **HTML files**
  - **Initialization file** - Initializes the MapServer application when first invoked.
  - **HTML template** - Provides arrangement of the resulting map image and other information into the final form of the web pages.

2.2.1 Mapfile

Mapfile is a XML-like structured configuration file which describes a number of geographic objects and behavior of the map in the web browser. These objects include legends, maps colors, scale bars, types of symbol shapes, projections, map layers etc.

Mapfile contains special keywords which may take different values. The string values are usually quoted.

In the simple example below it is shown, what a mapfile looks like. Sake of the example I have chosen the geographic data of districts in the Czech Republic in a shapefile format. The attributes in the attribute table are the following:

- the name of the district

- the region of the district
- the area of the district

In this example I will use only one attribute - the name of the district.

```
MAP
NAME "Districts of the Czech Republic"
UNITS meters
SIZE 640 480
IMAGECOLOR 255 255 204
IMAGETYPE gif
SHAPEPATH "/home/havlima6/mapdata/okresy"
FONTSET "/home/havlima6/public_html/fonty/fontset.txt"
EXTENT -831337.0 -1184398.014085 -559195.0 -971397.985915
```

- MAP is the keyword which shows that the file is really a mapfile and it is closed by END at the end of the file.
- SIZE determines the size of the resulting map in pixels.
- IMAGECOLOR defines the background of the map image (in this case it is light yellow color).
- IMAGETYPE determines the format of the map image.
- SHAPEPATH shows the path to the directory, where data is stored at the server.
- FONTSET defines the path to the file, which defines the trutype fonts. Each line contains the name or path to the truetype font and its alias.
- EXTENT defines the rectangular geographic extent of the data by two pairs of coordinates (x and y coordinate of lower left corner and x and y of the upper left corner). In this case data is in the S-JTSK coordinate system.

```
WEB
   TEMPLATE "/home/havlima6/public_html/okresy/okresy.html"
   IMAGEPATH "/var/www/tmp/"
   IMAGEURL "/tmp/"
END
```

- WEB - contains keywords which are connected with publishing map images.
  - TEMPLATE the path to the HTML template, which is used for displaying the resulting map image as a web site.
  - IMAGEPATH defines the directory, where MapServer stores the created map images. Is is an absolute path.
  - IMAGEURL is a path to the image relative to the Web Document Root.

```
LAYER
    NAME "Districts"
    DATA "okresy"
    STATUS default
    TYPE polygon
    LABELITEM "okres"
```

- LAYER marks the definition of a layer and is closed by END keyword. The layers are displayed hierarchically as they are defined in the mapfile. The first layer is plotted first, which means that the other layers are plotted above this layer.
  - NAME defines the name of the layer
  - DATA is the name of the file, where data is stored. In this case there is no suffix in the name of the file, because it is the shapefile format which consists of three or more files with the same basic name and different suffixes.
  - STATUS determines if the layer will be rendered. It can take three values: default, on and off. The "default" value displays the layer every time, the "on" value renders the layer and it can be switched into the "off" status, when the layer is not rendered.
  - TYPE defines the type of the data and determines how the MapServer renders the data. It needn't correspond to the type of the shapefile data (e.g., the polygon shapefile can be rendered as a point), but it must be compatible with it. The keyword TYPE can take the following kinds of values:
    - point - It defines isolated objects that can be drawn by a point symbol, e.g., peaks of the mountains or ground elevation.
    - line - It defines the series of points representing continuous object, e.g., rivers or highways.

  - polygon - It defines an enclosed area. The first and the last vertices, which define the polygon, must be the same.
  - circle - It must be defined by two points of the bounding rectangle. These two points are the opposite corners of that box.
  - annotation - The label point is defined and the label of the object is rendered, but the map feature is not displayed.
  - raster - The layer is rendered as a georeferenced image, e.g., a satellite photograph.
  - query - It describes the layer which only can be queried but not drawn.
  - chart - It is possible to include the ability to render automatically pie or bar graphs with the version 5.0 of MapServer. The values for the graphs are taken from data attributes.

- LABELITEM defines the name of the attribute in attribute table which is used for labeling.

```
 CLASS
     NAME "District"
     STYLE
       COLOR 200 132 100
       OUTLINECOLOR 0 0 0
     END                  #end of the STYLE
     LABEL
      TYPE truetype
      FONT "arialuni"
      ENCODING "Windows-1250"
      OUTLINECOLOR 255 255 255
      COLOR 0 0 0
      SIZE 10
     END                   #end of the LABEL
   END                    #end of the CLASS
END                      #end of the LAYER
END                      #end of the MAP
```

- CLASS defines, where the class of an object starts. Each feature is tested for

each class. If the feature matches into the min or max scale limitations, than that class is used for the rendering the feature.

- STYLE defines the style of the class and is closed with the END keyword.
  - COLOR can be used at different places of the mapfile, as it can be seen above. In the first case the keyword COLOR defines the color of the object.
  - OUTLINECOLOR defines the color of the object borders. In this case it is the borderline of the district.
- LABEL defines labeling of the objects and it is closed by END.
  - TYPE defines the type of the font. It can take two values: bitmap (simple style, fontset is not requiered) or truetype (the font is scalable and available in variety of styles).
  - FONT is defined by alias, which is part of the fontset file.
  - ENCODING is useful in case that some characters are displayed in a wrong way.
  - OUTLINECOLOR defines the outlined color of the label features.
  - SIZE determines the font size of the label.

Finally the keyword END close the keywords CLASS, LAYER and MAP.

Of course this is not a complete list of keywords that can be used, but I only indicate the possibilities of objects definition using mapfile. A comprehensive list of keywords can be found on the MapServer website [20]. The complete code of the mapfile can be found in appendix A.

2.2.2 Initialization File

Initialization file is used to send the initializing query to the MapServer and web server. The example of the initialization file below shows, how it is used.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
   <head>
     <title>Districts of the Czech Republic</title>
   </head>
```

```
  <body>
    <form method="POST" action="/cgi-bin/mapserv">
      <input type="submit" value="Submit">
      <input type="hidden" name="program" value="/cgi-bin/mapserv">
      <input type="hidden" name="map" value="/home/havlima6/mapdata/
      okresy/okresy.map">
       <input type="hidden" name="layers" value="Districts">
    </form>
  </body>
</html>
```

As we can see, the HTML file contains the form, which sends information by POST method to the CGI executable program mapserv. The form contains hidden inputs. When the HTML template is read, the strings [program], [map], whose names correspond to the names of CGI variables, are substituted by the CGI variable values from the initialization file.

2.2.3 HTML Template

The HTML template arranges the resulting map, scale bars and other information on the web page. The HTML template contains the substitution strings, which are replaced by values from the initialization file or by the current values generated by MapServer.

```
<!-- MapServer Template -->
<html>
  <head>
    <title>Districts of the Czech Republic
    </title>
  </head>
  <body>
    <center>
      <form name="the_form" method="GET" action="[program]">
        <input name="img" type="image" src="[img]" width=640 height=480
        border=2>
        <br>
        <select name="zoom" size="1">
```

```
            <option value="2" [zoom_2_select]> Zoom in 2 times
            <option value="1"  [zoom_1_select]> Recenter Map
            <option value="-2" [zoom_-2_select]> Zoom out 2 times
          </select>
          <input type="hidden" name=imgxy value="320 240">
          <input type="hidden" name=imgext value="[mapext]">
          <input type="hidden" name=program value="[program]">
          <input type="hidden" name=map value="[map]">
          <input type="hidden" name="layers" value="[layers]">
        </form>
      </center>
   </body>
</html>
```

The first substitution string in this template is the [program], which is replaced by the value of the CGI variable from the initialization file. The next string is [img], which MapServer replaces with the name of the resulting map image and the URL path to it. The other part of the HTML template, describes zoom controls for the map. It is possible to select these options: zoom in, zoom out and recenter the map. If the zoom in option is selected, the MapServer checks it and replaces the substitution string [zoom_2_select] by string *selected*.

The string [mapext] is substituted by the coordinates of the lower-left and upper-right corner of the map bounding box i.e. extent. The MapServer has to remember the path to the mapfile and to the executable program. The variables are passed on to the MapServer by the hidden inputs. The HTML template after the substitution of the variables:

```
<!-- MapServer Template -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
   <head>
     <title>Districts of the Czech Republic</title>
   </head>
   <body>
     <center>
       <form name="the_form" method="GET" action="/cgi-bin/mapserv">
         <input name="img" type="image"
```

```
        src="/tmp/Districts of the Czech Republic125465214820945.gif"
        width=640 height=480 border=2>
        <br>
        <select name="zoom" size="1">
          <option value="2" > Zoom in 2 times
          <option value="1"  selected="selected"> Recenter Map
          <option value="-2" > Zoom out 2 times
        </select>
        <input type="hidden" name=imgxy value="320 240">
        <input type="hidden" name=imgext
        value="-843343.263843 -1170390.705343
        -559195.000793 -957390.677173">
        <input type="hidden" name=program value="/cgi-bin/mapserv">
        <input type="hidden" name=map">
        value="/home/havlima6/mapdata/okresy/okresy.map">
        <input type="hidden" name="layers" value="Districts">
      </form>
    </center>
  </body>
</html>
```

The complete simple application is available on the following website address: `http://maps.fsv.cvut.cz/~havlima6/okresy/okresy_i.html`. The sample is shown in the Figure 2.2.

2.3 Functionalities of Simple MapServer Applications

To creating a useful web mapping application for MapServer it is important to have some knowledge of HTML, CSS, JavaScript to be able to create map controls and other parts, which every web mapping application should have. In this section there is shown, that it is possible to design the web mapping application with tools of MapServer using JavaScript and HTML. The functionalities are demonstrated on the demo web mapping

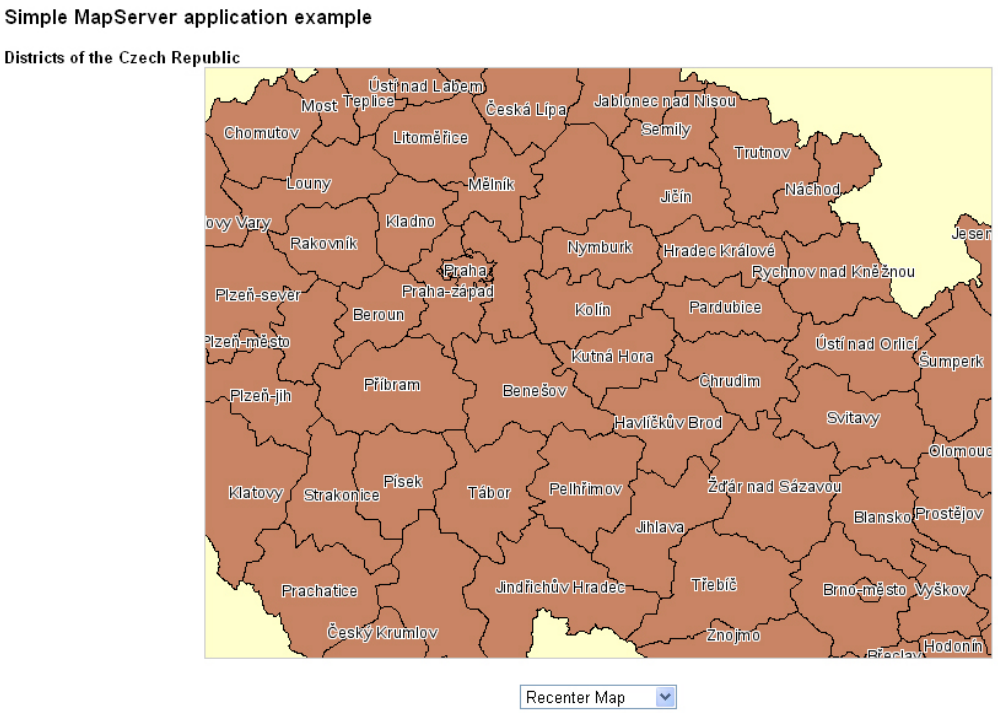

Figure 2.2: The resulting simple web mapping application, Source: `http://maps.fsv.cvut.cz/~havlima6/okresy/okresy_i.html`

application as it can be seen in Figure 2.3, which is available on MapServer website: `http://demo.mapserver.org/mapserver_demos/workshop-5.4/`

2.3.1 Map Controls

- Panning - The panning is performed by selecting this option and then clicking into the map or by buttons at the frame of the map image. The function of panning buttons was written in JavaScript.
- Zooming - The zooming option is performed by clicking into the map image.
- Overview map - It is possible to create an overview map image, which is generated by MapServer.

The scale bar image, the legend image, the map image and the overview map image are generated by MapServer according to the configuration in mapfile and saved. CGI variables are than replaced by paths to these images in the HTML template

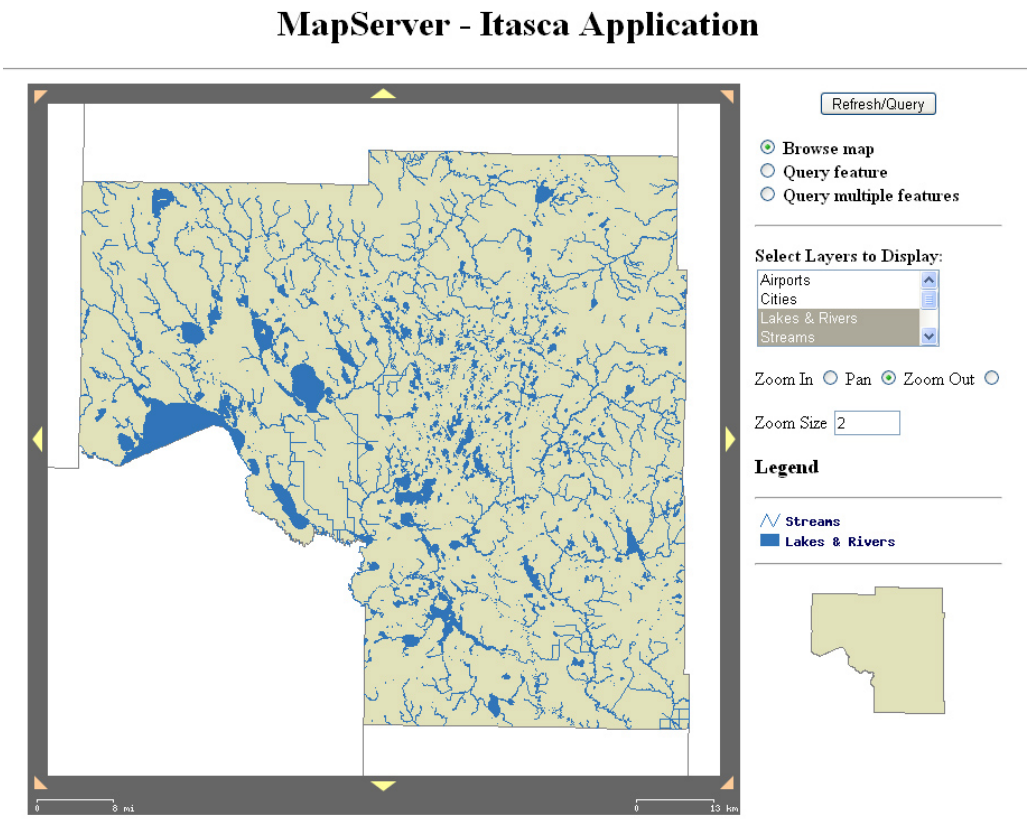

Figure 2.3: MapServer Web mapping application, Source: `http://demo.mapserver.org/mapserver_demos/workshop-5.4/`

2.3.2 Layers

MapServer supports a large amount of different data sources in vector or raster format such as WMS, WFS, shapefile, MapInfo, KML, GIF, GeoTIFF etc.

- Layer tree - There is a simple layer tree to switch the layers on/off. The legend image is generated by MapServer.
- Drawing - It is possible to program a drawing tool.
- Measuring - It is possible to program a measuring tool.

2.4 Experience of Application Developer

The user interface of this application is not very friendly for the final user. At first sight application does not have a very attractive design and the map control tools are not very user friendly either. The final user is accustomed to using different types of panning or zooming and these tools in this example may not be the best one for him. For the developer of the web mapping application it is quite difficult to program other types of map controls.

2.5 Conclusion

It is possible to program one's own tools, but it is very time-consuming. As it can be seen in the example, it is not easy to create functional map controls in a nice design on one's own and that is the reason why there are a lot of frameworks which focus on creating user interfaces for web mapping applications and which support a lot of widgets for map control and management of layers. In Chapter 3 I described functionalities and possibilities of these frameworks for user interfaces. I evaluated these frameworks in Chapter 4 and found the most satisfactory one, which I used to create the particular web mapping application described in Chapter 5.

Chapter 3

Frameworks for User Interfaces in Web Mapping Applications

User interface frameworks in web mapping applications are tools designed to help application developer make dynamic web map easily. These frameworks are usually JavaScript libraries for building web-based geographic application. In this chapter I describe selected frameworks by virtue of many aspects:

- General information
- Functionalities
  - Map Controls (zooming, panning, printing, overview map)
  - Layers (layer tree, measuring, drawing)
- Documentation
- Experience of application developer
- Layout

Though the examples of web mapping applications I show the possibilities of these frameworks.

These frameworks were chosen, because they are the most famous, and sophisticated enough to be compared with each other. Basically, they are the most available frameworks on the "market". The sequence of the frameworks in this thesis is not random. The first framework OpenLayers is the most famous, in my opinion, and the second MapFish is very close to OpenLayers, because it is based on OpenLayers. The other five frameworks are less known and their sequence is alphabetical. Described frameworks are the following:

- OpenLayers in Chapter 3.1
- MapFish in Chapter 3.2
- Fusion in Chapter 3.3
- GeoMoose in Chapter 3.4
- ka-Map! in Chapter 3.5
- msCross in Chapter 3.6
- p.mapper in Chapter 3.7

For every framework I created a simple example of the web mapping application which displays the same vector data of Districts of the Czech Republic as in the previous chapter and OpenStreetMap of Europe, which is distributed via WMS. The list of links for all of these applications is available at website address: `http://maps.fsv.cvut.cz/~havlima6/`.

3.1 OpenLayers

OpenLayers is a Free Software JavaScript library with JavaScript API. The initial part version of OpenLayers was developed by MetaCarta [17], which gave it to public for further use. Now the framework is developed by Open Source Geospatial Foundation [8] and is released under a BSD-style License [16].

3.1.1 Documentation

Website [7] of the OpenLayers framework contains lots of information about the framework, such as general information, wiki, where it is possible to find the documentation for users and developers, and advice how to start with OpenLayers, blog, FAQ, mailing list and a part where it is possible to download the current version, or older versions via SVN. There are also about 150 examples of usage, which are the best way to explore the basic functionalities. On the web sites it is possible to find the documentation of OpenLayers Library, where the functionalities and the parts of the web mapping applications

are described, such as layers, map controls etc. Some parts of this documentation are not finished yet.

3.1.2 Functionalities

3.1.2.1 Map Controls

- Panning - Map can be moved by visual buttons with arrows, by dragging the map by mouse, or by a dragged rectangle - the SHIFT button must be pressed. The panning is continuous without reloading the page.
- Zooming - It has four forms: the first one is just simple zoom in and zoom out marked by buttons plus and minus, the second one is the scale range which can be divided into an arbitrary amount of levels for any base layer, the third one is zooming by the scrolling wheel on the mouse and the last one is double click on the map. The first two forms are shown in Figures 3.2 and 3.3.
- Overview map - It can help users with orientation, when the main map is too zoomed. The overview map appears after clicking on the button on the right bottom side of Figure 3.2 with the plus icon.

3.1.2.2 Layers

The OpenLayers supports lots of layer formats, such as Google maps, WMS, WFS, KML, GeoRSS etc. There are two types of layers: the base layer and non-base layer. The raster layers are usually set up as a base layer and only one base layer can be displayed at any given time. The active base layer determines the projection and zoom levels. Non-base layers (or Overlays) are multiple enabled at a time and do not determine the projection or zoom levels. The usual source data format for overlay layers is a vector.

- Layer tree - The panel for switching layers is hidden, like the overview map. It appears after clicking on the "plus" button. It is not possible to move layers in the tree by drag and drop. The order of the layers in the tree must be changed in the code. See 3.2.
- Drawing - It is possible to edit map by three editing tools: point, path, polygon. See 3.3.

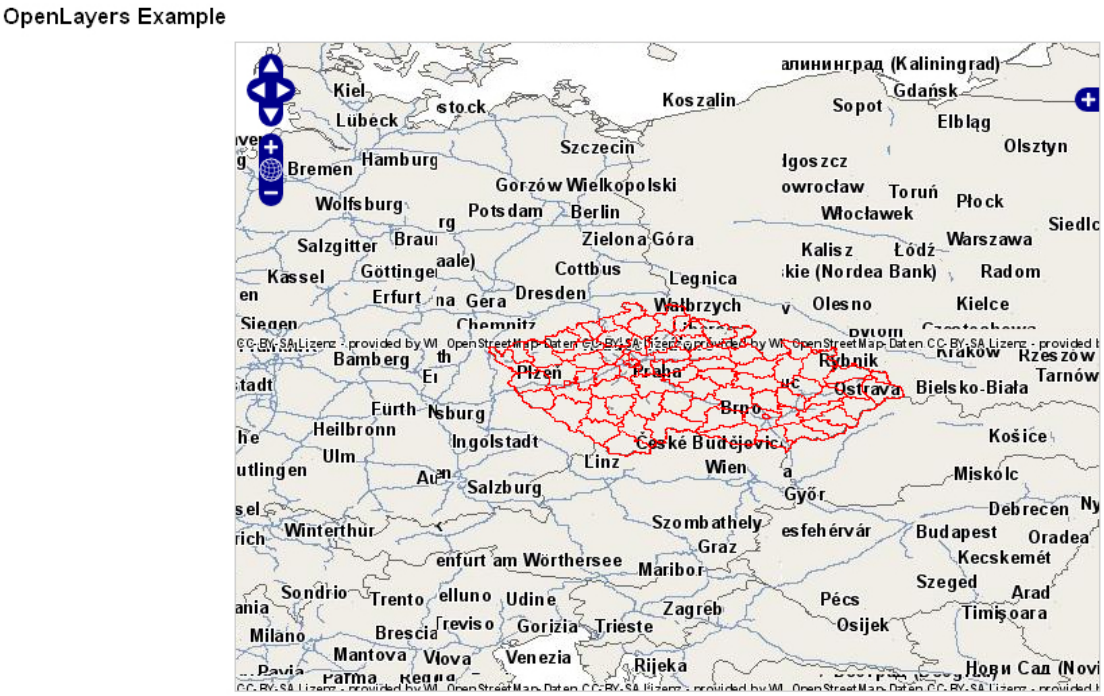

Figure 3.1: OpenLayers - example of usage,
Source: `http://maps.fsv.cvut.cz/~havlima6/openlayers`

Figure 3.2: Map Controls,
Source: `http://openlayers.org/dev/examples/controls.html`

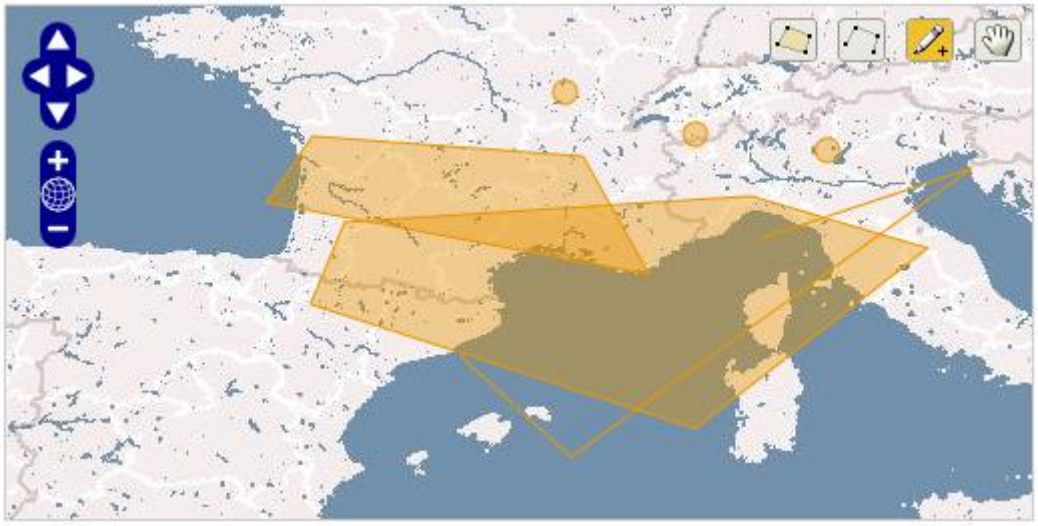

Figure 3.3: Edit tools,
Source: `http://openlayers.org/dev/examples/editingtoolbar.html`

- Measuring - It is possible to measure the distances and areas. The measurements of the areas are planar but if the map is in geographic projection, the measuring can be set as geodetic.
- Printing - The printing tool is not available.

3.1.3 Experience of Application Developer

The OpenLayers do not have to be downloaded and installed aon your server. The link for the JavaScript file OpenLayers.js, which is available at `http://www.openlayers.org/api/OpenLayers.js`, in the HTML template is enough and then the OpenLayers is ready to use.

Creating the first simple application in OpenLayers is quite easy. In the documentation of the framework there is clearly described how to begin and create the first map. Lot of other functions can be used from the examples, that are available on the framework web sites, and it can make first steps easier.

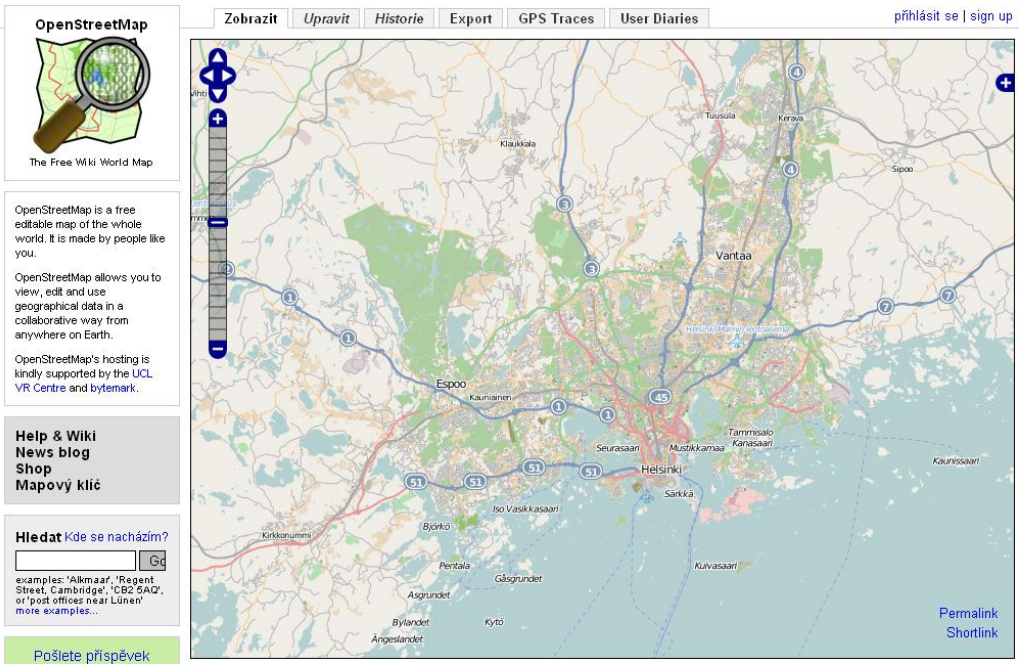

Figure 3.4: OpenStreetMap,
Source: `http://www.openstreetmap.org/`

3.1.4 Layout

The design of OpenLayers is visible at first sight. Its web map applications are characterized by the control panels colored dark blue with white signs. The scene around the map window is not the subject of the OpenLayers framework.

3.1.5 Examples of Usage

OpenLayers is a quite wide spread framework. One of the most famous usage is in the OpenStreetMap project (see Figure 3.4), whose goal is to create free available geographic data and its visualization to topographic maps. For geodata creating GPS or other digitized maps are usually used. It is based on the principles of Open Source and cooperation. Everyone can join the project and help with mapping. The framework has already several language versions and is available on web sites: `http://www.openstreetmap.org/`.

Another interesting web mapping application is on the web site of National Weather Service, where the data of radar measurements and the hazards such as thunderstorms and floods are displayed. The web mapping application is available on the web site: `http://radar.srh.noaa.gov/`.

3.2 MapFish

The latest version 1.2 of MapFish, which was released under the GPLv3 license [5] at the end of August 2009, is based on the Pylons Python web framework. MapFish JavaScript toolbox is based on OpenLayers, ExtJS and GeoExt.

3.2.1 Documentation

In that time, when I studied information about the MapFish, the web pages [22] of the framework were under the reconstruction, so functionality of the web pages was changing every day. A very interesting part is the one called "8 hours of introduction to MapFish framework", which explains all basic things around the framework. On other framework website I did not found such a comprehensive description of a framework.

On the website of the MapFish framework it is possible to find link to the MapFish blog, documentation, link to the twitter account and lot of sample web mapping applications, which shows functions and advantages of this framework. The web sites are very well processed.

3.2.2 Functionalities

The MapFish adopts the basic control panels, such as navigation (zooming, panning and overview map) and editing from the OpenLayers, and extends its possibilities.

3.2.2.1 Map Controls

The control panels (zooming, panning and overview map) can be the same as in the OpenLayers.

- Panning - This tool was adopted from OpenLayers. The panning is continuous without reloading the page.
- Zooming - This tool was adopted from OpenLayers. In contrast with OpenLayers there is one more possibility of zooming: zooming by using the dragged rectangle.
- Overview map - This tool was adopted from OpenLayers.
- Print widget - The print widget for report creation in PDF is available.

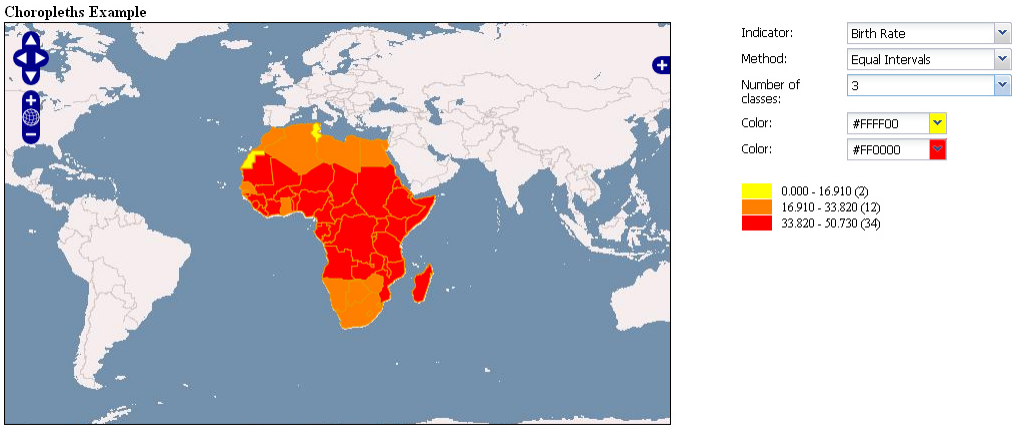

Figure 3.5: Statistic functions - choropleth,
Source: `http://demo.mapfish.org/mapfishsample/1.1/examples/geostat/choropleths.html`

3.2.2.2 Layers

- Layer tree - Layer tree in MapFish can be generated automatically from OpenLayers map. The layer tree widget is in MapFish framework on a high level. There are several possibilities how the layer tree can work. One interesting option is that final user can change the order in the layer tree by drag and drop on-line in the final web mapping application.
- Measuring - This tool was adopted from OpenLayers.
- Drawing - This tool was adopted from OpenLayers. Moreover it is possible to edit the drawings and their attributes.
- Geostatistics - One of the big advantages of MapFish is the possibility of creating a statistical outputs on-line in the web mapping application. There are two kinds of mapping methods of creating the thematic map: choropleth and proportional symbols. In case of choropleth it is possible to select parameter for indicator, the method of classification, the number of classes and the color of classes. In case of proportional symbols the indicator and maximum and minimum size of the symbols are set.

3.2.3 Experience of Application Developer

During my work on this thesis the latest version of MapFish was released. The installation of version 1.1 was very similar to OpenLayers - only the links to the JavaScript files were required in HTML template. Adding layers is the same as in OpenLayers framework. The installation of the latest version 1.2 is different. The first step is to download the python file (go-mapfish-framework-1.2.py). Then, according to the manual [2], thanks to the following command the virtual Python environment is activated and MapFish and its dependencies installed.

```
$ python go-mapfish-framework-1.2.py --no-site-packages env
```

By following the commands in the manual the sample example is generated.

3.2.4 Layout

The MapFish is based on OpenLayers, that is why the basic design is very similar. The main control buttons are in dark blue as well. It is possible to use the default layout, which consists of four basic sectors. The first sector is in the top part of the application and is used as a title. The second sector on the left or right side is usually used for displaying the layer tree and other functions, the third sector is used for the map image, and the last one is in the bottom and is usually used as a status bar as it is shown in Figure 3.6.

3.2.5 Examples of Usage

The MapFish is widely spread and is used in interesting projects, such as displaying of Switzerland mobility. In this project there are displayed the routes for many kinds of sports such as hiking, cycling, skating, mountainbiking and canoing in Switzerland. The web mapping application is available at: `http://map.veloland.ch`.

In summer semester 2009 I used the MapFish in teamwork for the course "Project of Informatics 2". The project focuses on introduction with MapFish and creating the web mapping application. The web mapping application displayed WMS layers (OpenStreetMap, Google maps and OpenAerialMap) and is connected to the PostGIS database. The PostGIS database layers contain the points, which represent the entrances into the metro stations in Prague, and lines, which represent railways and streets. This data

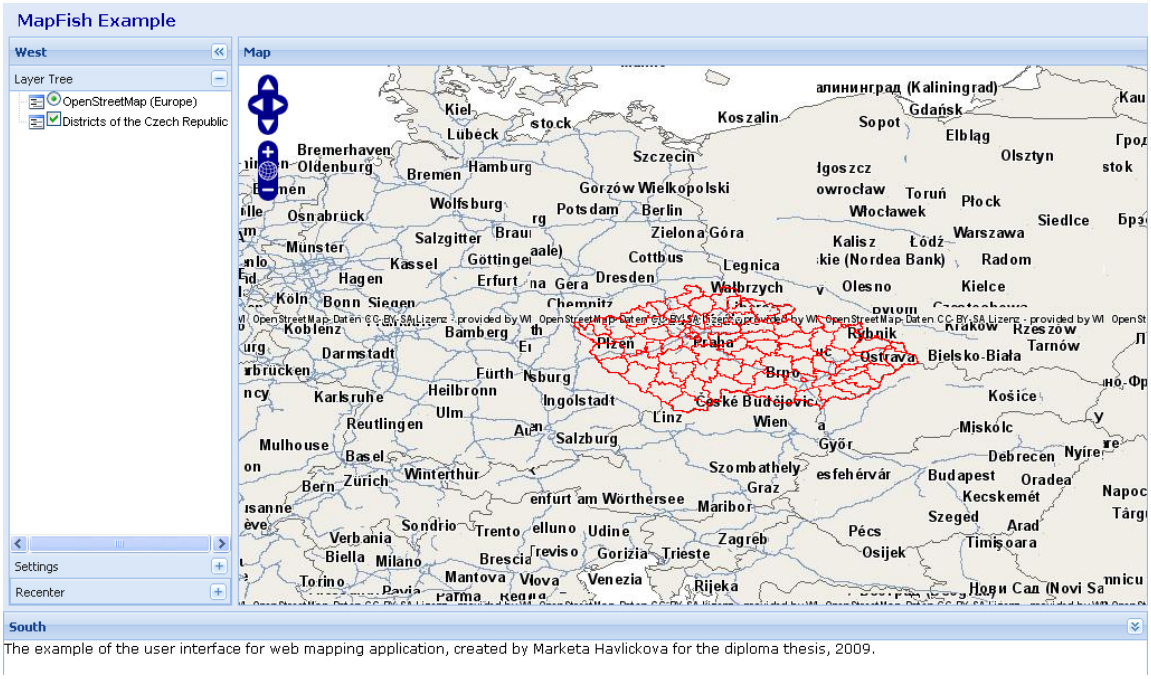

Figure 3.6: MapFish complex layout - example of usage,
Source: `http://maps.fsv.cvut.cz/~havlima6/mapfish/`

was mapped by other students teams using the GPS. The web sites of the project: `http://josef.fsv.cvut.cz/~pin2-2009/a/mapfish/`.

3.3 Fusion

Fusion is a development framework for creating web mapping applications. It is built in JavaScript. The latest version of Fusion 1.1.1 was released at the beginning of October 2009. Fusion is released under MIT license [19]. Fusion was originally developed for MapGuide [6] but it also supports MapServer for a short time.

3.3.1 Documentation

The websites of Fusion framework [21] are poor compared with other frameworks above. On the websites there can be found a mailing list, a bug tracker, FAQ, and, of course, documentation, but it is not complete. For a developer who is in touch with Fusion for a long time the information on the pages can be sufficient, but an application developer can

be confused at first. However, the biggest drawback is that there are no samples of web mapping applications on the website. After searching for a long time I found only one example, which is strongly customized and shown in Section 3.3.5. This can discourage new users, because it is impossible to get an idea about the functions and design of the final web mapping application at first sight.

3.3.2 Functionalities

3.3.2.1 Map Controls

- Panning - The panning is performed by dragging the map image or by recentering the map by means of a click into the map image. The continuous panning is not supported.
- Zooming - The zooming in is performed by a dragged rectangle or by a click on the map.
- Overview map - It is possible to create the overview map.

3.3.2.2 Layers

Fusion supports all data sources served by MapServer.

- Layer tree - The widget which supports creation of layer tree is available.
- Measuring - The measuring of the distances and areas is available.
- Drawing - Drawing is not available.
- Printing - Printing widget is available.

3.3.3 Experience of Application Developer

Fusion was originally developed for MapGuide but it also supports MapServer for a short time. It can be the reason, why the documentation of the Fusion framework is not complete and the application developer cannot get the specific idea about the framework and can not estimate which possibilities are covered in this framework at once.

The Fusion for MapServer must be downloaded and installed first and some changes must be done in the configuration file - config.json. The basic web mapping application

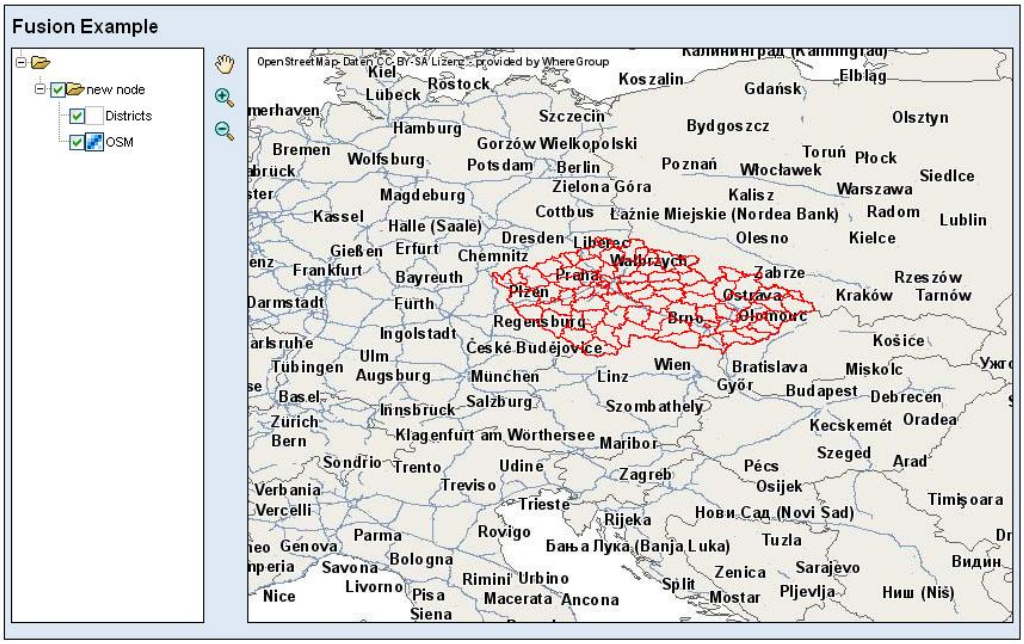

Figure 3.7: Fusion example, Source: `http://maps.fsv.cvut.cz/~havlima6/fusion-test/`

consists of ApplicationDefinition.xml file, HTML file and the mapfile. When installing Fusion and creating the first basic application I followed the manual step by step, nevertheless, the demo application did not work. After a long time I found the typing error in the example of ApplicationDefinition.xml at Fusion website.

3.3.4 Layout

There is no default layout for Fusion web mapping application. Only the standard icons can be used as it is shown in Figure 3.7. The layout is performed by application developer in HTML and CSS.

3.3.5 Examples of Usage

The Fusion for MapServer is not very spread. The incomplete documentation and an incorrect sample web mapping application made the work with Fusion difficult at the beginning. I found fewer examples of usage on the Internet than for the frameworks mentioned above. One of them which is strongly customized is shown in Figure 3.8

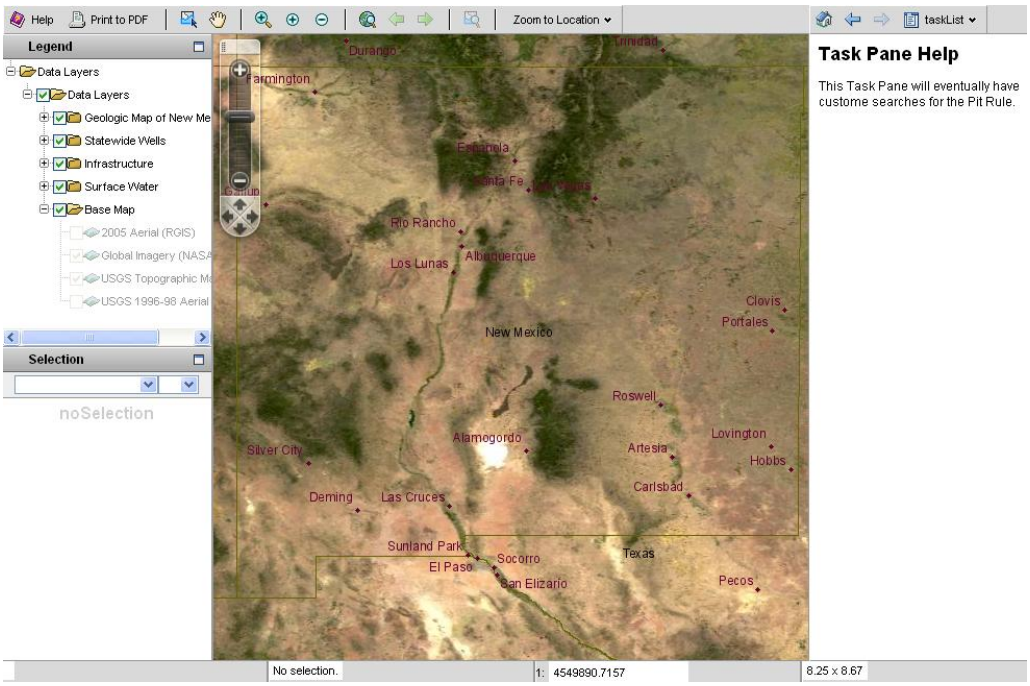

Figure 3.8: Fusion example of usage,
Source: `http://216.93.164.45/pitrule_fusion/`

3.4 GeoMoose

GeoMoose is a JavaScript framework for creating web mapping applications. The latest version 2.0.1 was released in the October 2009(but the mailing list announced that the latest version 2.2 would appear in November).

3.4.1 Documentation

The web pages of GeoMoose framework [13] are under reconstruction and these days they are divided into two parts. The new part of the web site informs about the newly released version 2.0.1 and contains documentation for this version, a section for downloading the code and a mailing list for users and developers. The old part contains the gallery of sample web mapping applications and also some older documentation.

3.4.2 Functionalities

3.4.2.1 Map Controls

- Panning - The map can be moved by visual buttons with arrows, by dragging the map by mouse or by recentering the map by means of a double click into the map image. Since the version 2.0 the panning is continuous without reloading the page.
- Zooming - Zooming is possible by clicking on the visual button plus/minus of the predefined scale range, by using a dragged rectangle, by clicking on the map or by selecting the scale from the selection menu with a predefined scale range.
- Overview map - Overview map is available. The position of the overview map can be changed on-line by selecting one of the options (top-left, top-right, bottom-left or bottom-right corner) in the menu. See Figure 3.9.
- Printing - It is possible to print the current view of the map and download it in PDF.

3.4.2.2 Layers

The GeoMoose supports WMS and WFS layers.

- Layer tree - The layer tree is available.
- Measuring - The measuring of distances and areas is available.
- Drawing - The drawing is available in the GeoMoose web mapping applications. The shapes of polygon, line or point can be drawn and than can be edited or their attributes can be changed.

3.4.3 Experience of Application Developer

For the installation it is necessary to download the source from GeoMoose web pages. Following the manual on the web pages is useful, but for the last steps it is necessary to have administrator's permits to server. However it is possible to avoid it. The better way for installation is the following:

1. Download the source and "untar".
2. Switch into the geomoose-[version] directory

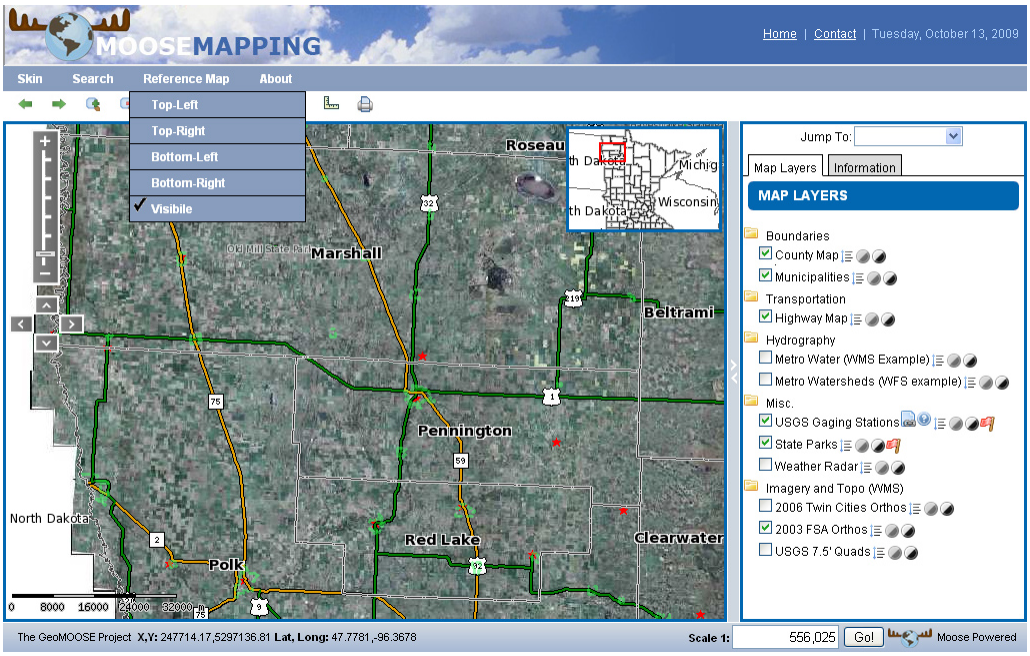

Figure 3.9: GeoMoose, Source: `http://geomoose.houstoneng.com/statedemo/statedemo.html`

3. Configure the paths: prefix is the path to your geomoose directory

```
./configure --prefix=/home/havlima6/GeoMoose/geomoose
--with-url-path=/~havlima6/geomoose
--with-server-name=maps.fsv.cvut.cz
```

4. Create symbolic link /htdocs/ directory, where the sample web mapping application is placed, into the web space.

```
ln -s /home/havlima6/GeoMoose/gemoose/htdocs/
/home/havlima6/public_html/geomoose/
```

5. The sample web mapping application is available at `http://maps.fsv.cvut.cz/~havlima6/geomoose/htdocs/index.html`. See Figure 3.10.

For customizing the web mapping application it is required to edit the /conf/mapbook.xml file and the HTML file /htdocs/geomoose.html. In mapbook.xml layers, mapfile, scales, search tools and other tools are setting up. The geomoose.html contains just layout settings of the final user interface.

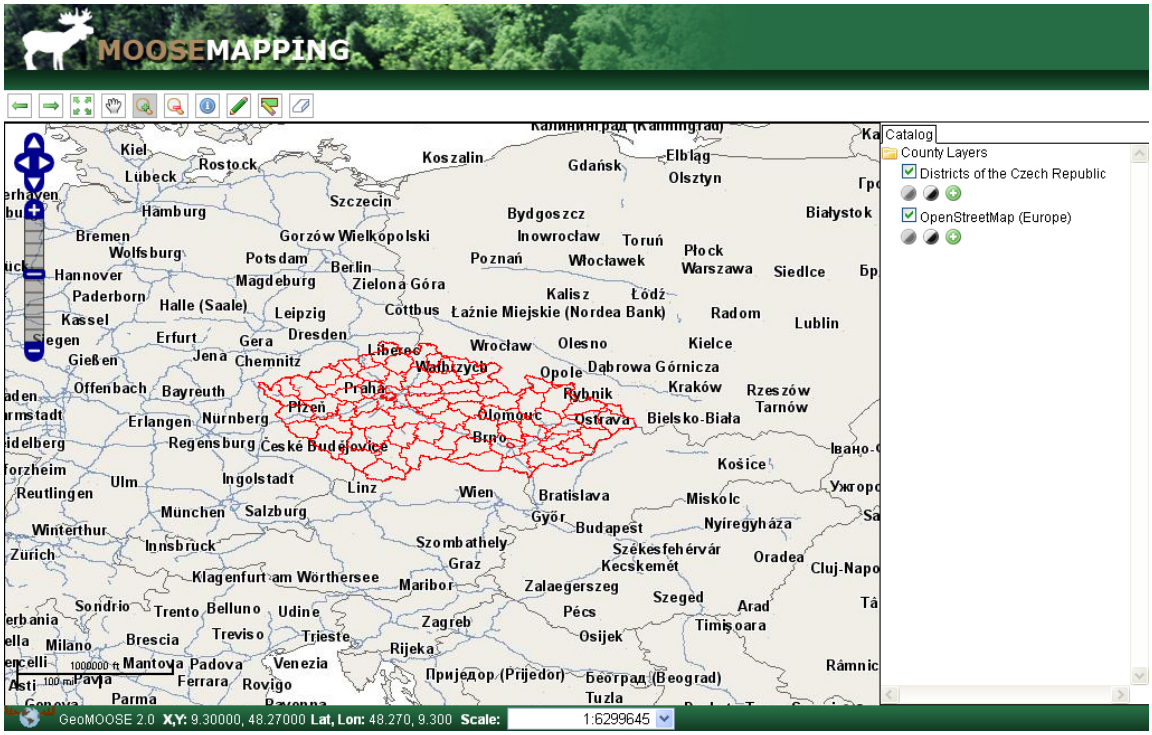

Figure 3.10: GeoMoose example of usage,
Source: `http://maps.fsv.cvut.cz/~havlima6/geomoose/htdocs/index.html`

Adding layers is configured in mapbook.xml. If it is necessary to use the projections, which have not been defined yet, it is important to define them into the separate JavaScript files.

3.4.4 Layout

The main design is available in three color hues - red, green and blue. The layout of the web mapping applications usually consists of three sectors. The first sector contains menu and tool bar buttons and is placed at the top of the application, the second sector is placed on the right side and contains other functions, such as a legend with the layer tree or, e.g., outputs of the queries. The third one is the sector of the map image, which usually contains a scale bar, a scale range and an overview map.

3.4.5 Examples of Usage

The GeoMoose is not very widespread throughout the world, but in the USA it is quite popular. Some US countries or cities use it. As an example I can metnion the web mapping application of the US city Blame, MN. The application shows the crime incidents from January to August 2009. See Figure 3.11. The web mapping application is available on website: `http://maps.ci.blaine.mn.us/geomoose/crimemapping.html`.

3.5 ka-Map!

ka-Map! is an open source framework with JavaScript API, used for creating web mapping applications. ka-Map! was originally developed by DM Solutions Group Inc. [11] in 2005. The latest version ka-Map! 1.0 was released in February 2007 under an open source license.

3.5.1 Documentation

The main web pages [14] contain the basic information about ka-Map! and a basic list of features. It is, of course, possible to subscribe to the mailing list, go through the old discussions in the archive or report new bugs. There are also wiki pages with lot

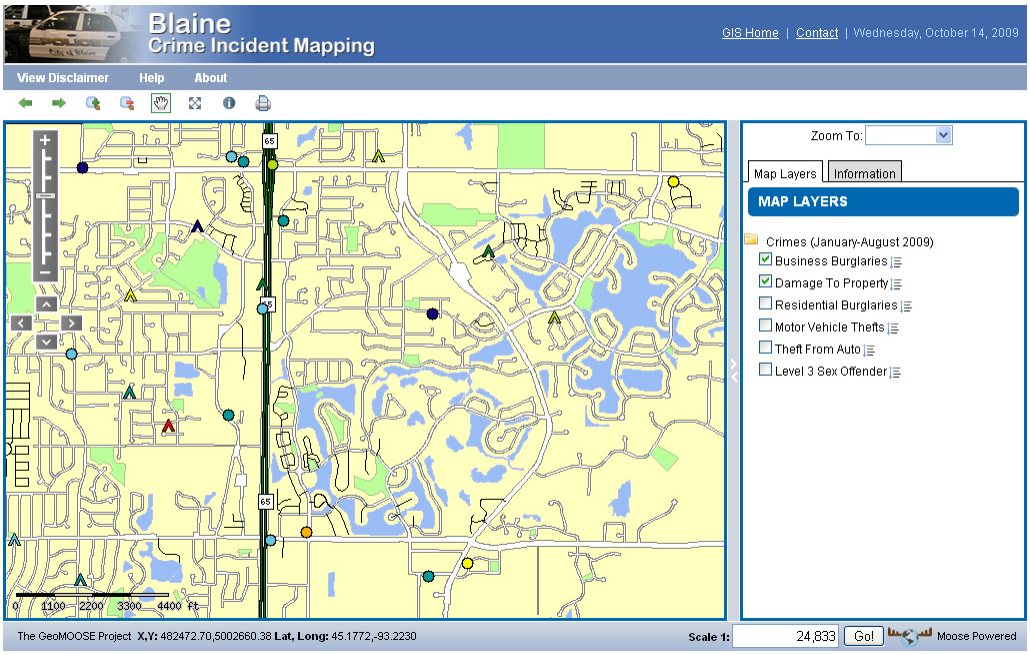

Figure 3.11: GeoMoose - The Crime Incident Mapping, Source: `http://maps.ci.blaine.mn.us/geomoose/crimemapping.html`

of examples of public applications created in different versions of ka-Map!. On the wiki pages there can be found a lot of user guides focused on different parts of ka-Map!, but one complete manual would be better and more comprehensive. Overall documentation is not processed so well as in the previous frameworks, such as OpenLayers or MapFish.

3.5.2 Functionalities

The ka-Map! framework contains mainly the basic functions such as zooming, panning etc., but printing is also available.

3.5.2.1 Map Controls

- Panning - Map image can be moved by mouse dragging, by clicking on the visual buttons in the shape of an arrow, or it can be recentered by a double click. The panning is continuous without reloading the page.
- Zooming - There are three basic possibilities of zooming: by a dragged rectangle, by clicking on the zoom in/zoom out button, or by selecting the predefined scale from the selection menu.

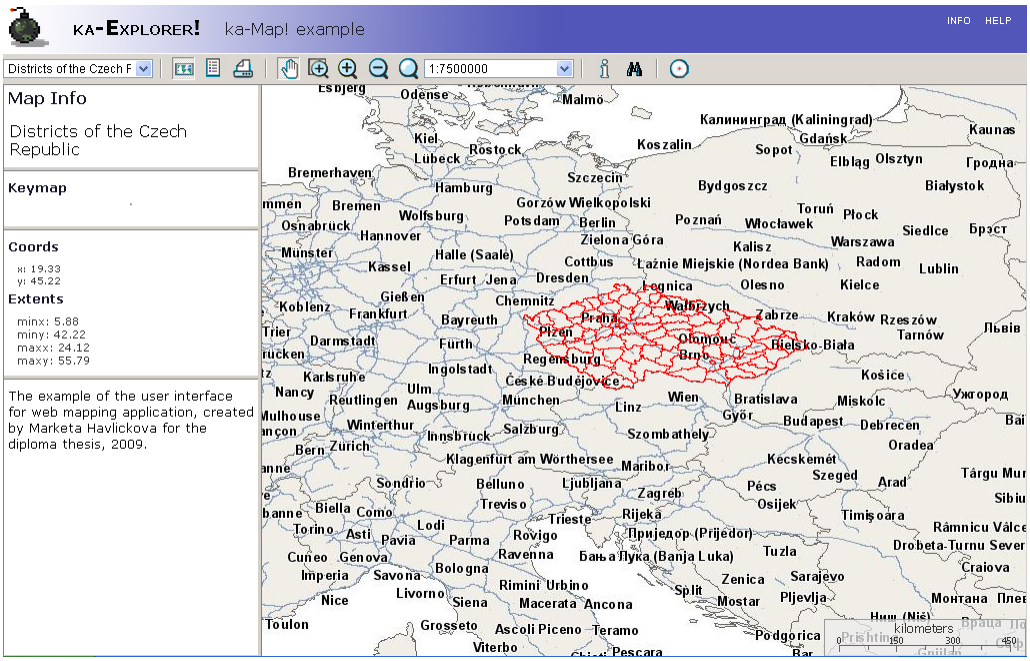

Figure 3.12: Default ka-Map! layout,
Source: `http://maps.fsv.cvut.cz/ka-map/index.html`

- Overview map - In the ka-Map! framework the overview map called keymap, is available.
- Printing - It is possible to print current view of the map and save it as GIF, JPG, PNG or PDF.

3.5.2.2 Layers

ka-Map! supports WMS and WFS layers.

- Layer tree - It is possible to use the function of changing layers order in the layer tree on-line.
- Measuring - Measuring is not available.
- Drawing - Drawing is not available.

3.5.3 Experience of Application Developer

For the installation it is necessary to download the latest version of the ka-Map! framework and follow the manual on the wiki pages "Preparing ka-Map!" [15]. For one of the

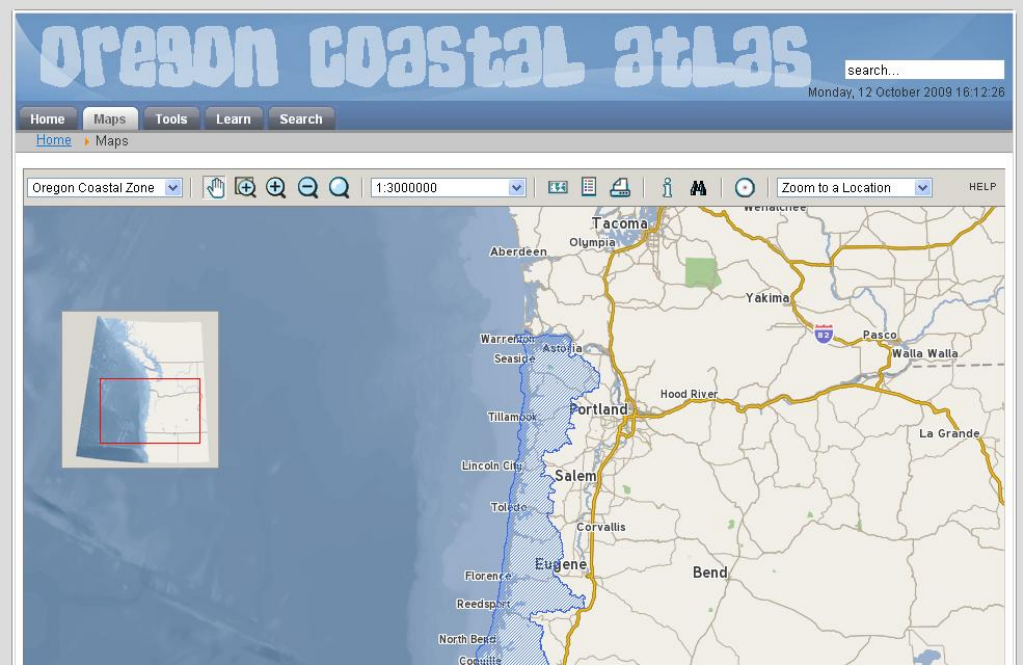

Figure 3.13: ka-Map! Oregon Coastal Atlas, Source: `http://www.coastalatlas.net/index.php?option=com_wrapper&Itemid=28`

steps, where the aliases are created, the administrator's permits for server are necessary, but this step is not so necessary and can be skipped. The different installation process is suggested bellow:

1. Download and unzip the package.

2. Create symbolic link of /htdocs/ directory, where the sample web mapping application is placed, into the web space.

   ```
   ln -s /home/havlima6/ka-map/htdocs/
   /home/havlima6/public_html/ka-map/
   ```

3. Rename /include/config.dist.php as /include/config.php and set up the path to the mapfile and scale range in config.php.

It is required to set up the config.php file, where it is possible to define the path to the mapfile and scale range.

In the installed ka-Map! there are various examples of sample user interfaces, which can be used as a default layout. I opted for the default layout called index.html. It is

difficult to customize this example, because styling is controlled in different files, such as startUp.js and three CSS files.

3.5.4 Layout

The default layout of the ka-Map! web mapping applications are typically in grey color. At the top of the user interface there is a tool bar with icons of map control, icons to turn on/off the legend (layer tree), map information and tools for searching and querying the map. On the left side there is a sector for a legend or map information and on the rest of the screen there is the sector for a map image. The design is very simple to intrude as little as possible on the application design. In my opinion, the ka-Map! framework is useful mainly for a basic map with dynamic point overlay layer. See 3.12.

3.5.5 Examples of Usage

On the wiki web sites of the framework there are a lot of links to web mapping applications using ka-Map! Some of them do not work anymore, some of them look very interesting such as Oregon Coastal Atlas. See Figure 3.13. The web mapping application is available: `http://www.coastalatlas.net/index.php?option=com_wrapper\&Itemid=28`.

3.6 msCross

The msCross is an open source JavaScript interface for UMN MapServer released under GPL license [5]. The latest version msCross 1.1.9 was released in February 2007.

3.6.1 Documentation

The msCross framework is very easy to use, that is why the documentation [4] does not have to be so extensive. The web pages contain general information about the framework and two examples of code of the web mapping application, and a description of API interface, which is not complete yet. It is possible to subscribe to the mailing list or take part in discussion.

3.6.2 Functionalities

msCross framework contains only the most basic map controls, such as zooming, panning and zoom to full map extent.

3.6.2.1 Map Controls

- Panning - Moving is simple by dragging the map image. The panning is not continuous, the page is reloaded every time.
- Zooming - In msCross framework zooming is possible by zoom in/zoom out buttons, or by a dragged rectangle.
- Overview map - The overview map is available.
- Printing - The printing in not available.

3.6.2.2 Layers

Supports WMS, WFS layers and overlay point layers.

- Layer tree - The layer tree is not available.
- Measuring - The measuring of distances and areas is not available.
- Drawing - The drawing is not available.

3.6.3 Experience of Application Developer

It is easy to start working with msCross. The installation is easy - the only JavaScript file must be downloaded and the link to this Javascript file must be placed in the HTML file. Adding layers is easy also, thanks to the examples of code, which are available at msCross web pages. The main problem is when the application should contain something more than only map image with navigation tools. The other parts such as layer tree must be programmed separately.

As mentioned at the beginning, it is possible to subscribe to the mailing list, or to take a part in discussions, but the last question in discussion was released a long time ago. Given the date of release of the latest version and the discussion, it seems that msCross framework is not alive anymore.

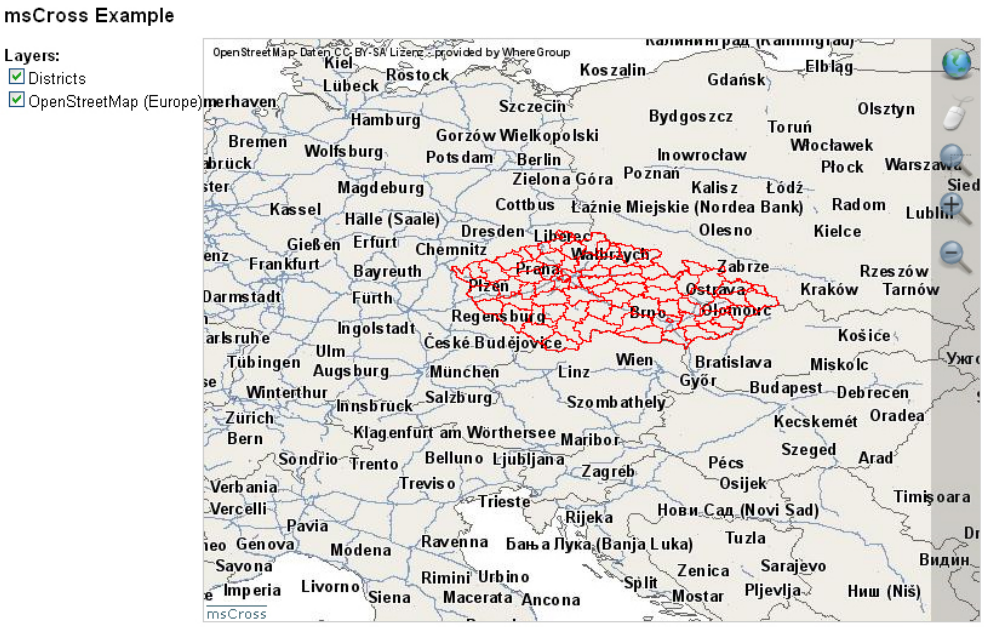

Figure 3.14: msCross,
Source: `http://maps.fsv.cvut.cz/~havlima6/mscross`

3.6.4 Layout

Design of the msCross framework is easy to describe. The only part of the design is the tool bar of map controls. The background of the tool bar is transparent, so it does not interfere with the appearance of the map. The tool bar can be placed in 5 positions: standard right (See Figure 3.14), standard left (placed on the left side of the map), standard up, standard corner left, right (the tool bar is angular and placed into the corner).

3.6.5 Examples of Usage

the web mapping application using msCross was created at the Czech Technical University in Prague, faculty of Civil Engineering, The Department of Mapping and Cartography. This application displayed the old Müller's maps of Bohemia and Moravia and was produced in cooperation with the Institute of History, Academy of Science of Czech Republic,v.v.i. and with the Central Archives of Surveying, Mapping and Cadastre. See Figure 3.15.

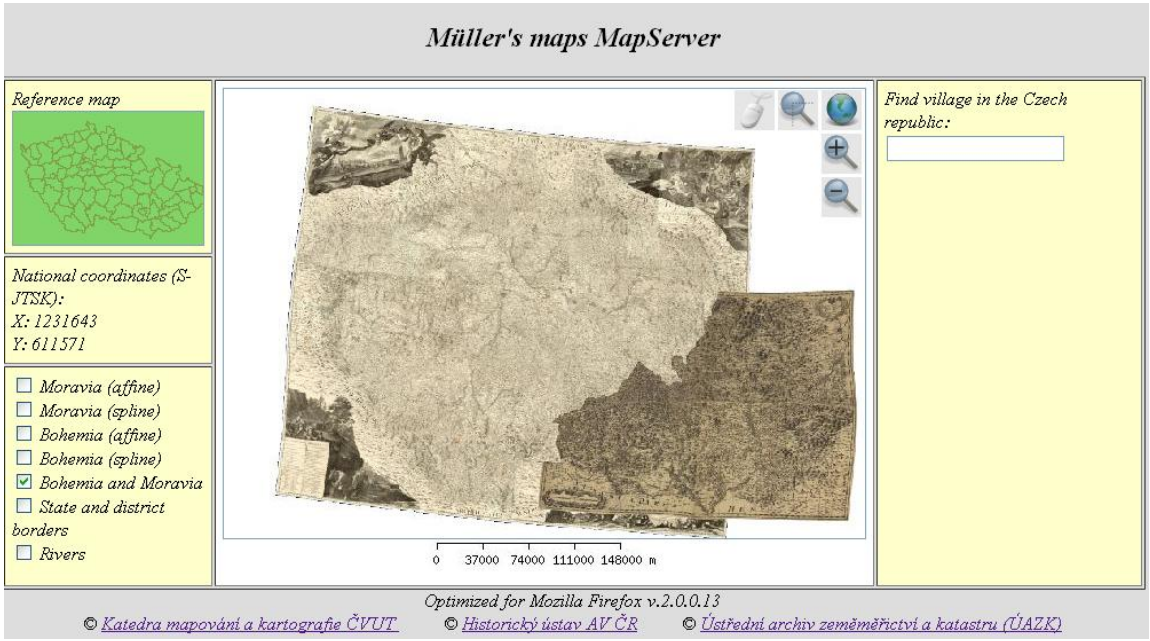

Figure 3.15: Müller's maps of Bohemia and Moravia,
Source: `http://maps.fsv.cvut.cz/muller/`

3.7 p.mapper

p.mapper is a framework based on UMN MapServer and PHP/MapScript. p.mapper is released under the GNU General Public License. The latest stable version 3 was released at the beginning of 2009 and 4.0 beta3 version at the end of September 2009.

3.7.1 Documentation

At the web pages of the framework [1] there are links for wiki pages, where the basic manuals are stored, it is possible to subscribe into the mailing list, or contact the commercial companies which are providing technical support for p.mapper. One of the advantages of p.mapper is that it is possible to download the various language files, which can localize the user interface. Unfortunately the Czech file was removed, because there are no maintainers available any more.

3.7.2 Functionalities

3.7.2.1 Map Controls

- Panning - Panning is performed by clicking into the map image, or by dragging the map image.
- Zooming - Zooming is possible by visual zooming buttons, by clicking into the map, by dragged rectangle.
- Printing - It is available to print into the PDF format, or save map image as a PNG format.
- Overview map - Overview map is available.

3.7.2.2 Layers

- Layer tree - The layer tree is available, the layers can be switched on and off.
- Measuring - The measuring of distances and areas is available.
- Drawing - It is not the typical drawing, such as in MapFish. It is possible to add the point of interest and name it.

3.7.3 Experience of Application Developer

For the installation of p.mapper just follow the installation manual at the p.mapper web pages. I found only one problem during the installation and configuration of the sample web mapping application. The access to the automatically generated images of legend was denied. This was easily corrected by setting permissions to these files.

The configuration files are stored in folder "../config/" and contain config_default.ini file (The format of file depends on version. It can be also XML file), where the layers, which should be displayed, are set and are contained also other custom settings. The other configuration files are in the "../config/default/" folder. There are two other configuration files: js_config.php, where the possibilities of layout, scale, scale bars, etc. can be set, and php_config.php file, where the categories of legend, toolbar buttons and their tool tips can be defined.

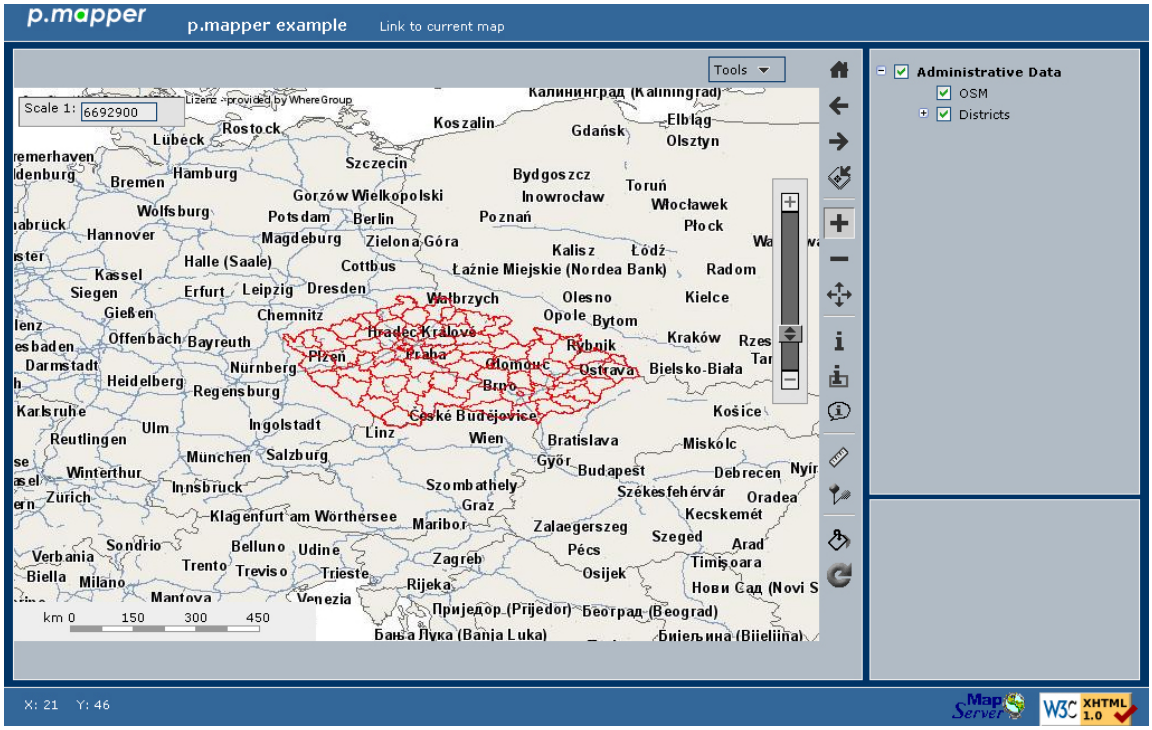

Figure 3.16: p.mapper example,
Source: `http://maps.fsv.cvut.cz/~havlima6/pmapper/pmapper-stable/`

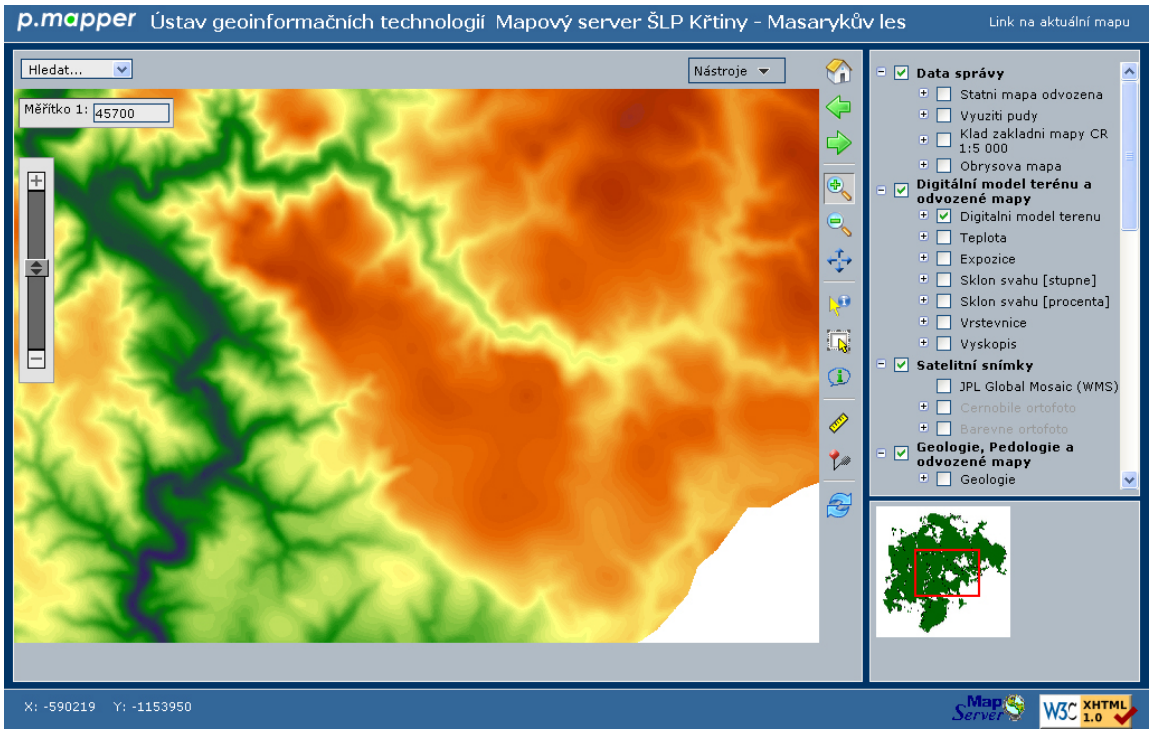

Figure 3.17: Example of p.mapper web mapping application, Source: `http://mapserver-slp.mendelu.cz//map.phtml`

3.7.4 Layout

The default p.mapper layout has a frame with layer tree at the right side of screen (Figure 3.16). The toolbars are at the right side of the map frame and also at the top. The background of the toolbars is in transparent gray color. It is possible to use other alternative layouts, which are, e.g., inspired by Google Maps layout.

3.7.5 Examples of Usage

The p.mapper is mainly spread in Europe. I found one Czech example which is from the Department of Geoinformation Technologies at the Mendel University of Agriculture and Forestry Brno. The web mapping application displays the region around Brno and it is possible to choose a lot of various layers from the branch of forestry and geology.

Chapter 4

Evaluation of User Interfaces

In this chapter I compare the user interfaces described in Chapter 3. The user interfaces were evaluated according to the criteria: functionalities, documentation, layout and application loading time.

The individual functionalities are summarized in Table 4.1. Individual functionalities are marked by "+" if the functionality is included in the framework, "-" if the functionality is missing, and "+/-" if the functionality varies from the others. These differences are described in Section 4.1.

The functionalities were divided into two groups: map controls, and functionalities connected with layers. These groups were evaluated by stars: *** (excellent), ** (average), * (unsatisfactory). This sort of evaluation helped me to take into account not only if the functionalities are included or not, but also the quality of the functionalities, and whether it is really necessary to include, e.g., all the panning functions into one framework.

This star classification is used also in Table 4.4. The meaning of the stars in the row, which evaluates experience with installation and configuration, is the following: *** (easy), ** (average), * (difficult).

			Framework						
Functionality			OpenLayers	MapFish	Fusion	GeoMoose	ka-Map!	msCross	p.mapper
Map Controls	Panning	dragging	+	+	+	+	+	+	+
		arrows	+	+	-	+	+	-	+
		double click	-	+	-	-	+	-	-
		click	-	-	-	+	-	-	+
		continuous	+	+	-	+	+	-	-
		Evaluation	***	***	*	**	**	*	**
	Zooming	buttons	+	+	+	+	+	+	+
		slider	+	+	-	+	+	-	+
		double click	+	+	+	+	-	-	-
		scrolling wheel	+	+	+	+/-	+	-	-
		dragged rectangle	+	+	+	+	+	+	+
		Evaluation	***	***	***	**	***	**	**
	Printing		-	+	+/-	+	+	-	+
	Overview map		+	+	+	+	+	+	+
Layers	Layer tree		+	+	+	+	+	-	+
	Dragging layers		-	+	-	-	+	-	-
	Drawing		+	+	+	+	-	-	+/-
	Edition of drawing		-	+	-	+	-	-	-
	Measuring		+	+	+	+	-	-	+
	Evaluation		**	***	**	***	*	-	**

Table 4.1: Overview of common functionalities of user interfaces

4.1 Functionalities

Table 4.1 contains an overview of common functionalities of the user interfaces. In this table there are not all the functionalities, but only the most common ones and the most used ones in web mapping applications, such as different types of zooming and panning. There is also mentioned if the framework contains tools for managing layers, such as a layer tree, and if it is possible to change order of these layers on-line. These functions mentioned above are the functions which are predefined in the frameworks. It is possible to program other functions on one's own. As it is shown in Table 4.1, the framework which contains the most functionalities is MapFish. Only MapFish also contains very interesting statistical functions, which allow creating thematic maps, such as choropleth maps and proportional symbol maps on-line.

The evaluation of panning and zooming functionalities is not represented only by the sum of "+" and "-", because it is not necessary to include all these different possibilities into one web mapping application, on the other hand more suitable possibilities for selecting various zooming/panning tools make the user more comfortable to choose the one that suits him or her the best.

According to [9] the most useful hybrid method, which covers zooming and panning tools, from functionalities mentioned above, is a dragged rectangle. It allows the user to drag and draw the rectangle directly into the map and get the new extent of the map. The user can save time and mouse movements better than while using two different tools for panning and zooming. The usability of this tool it is also shown in Table 4.1. All of the frameworks contain this tool.

The other useful zooming tool is scrolling the mouse wheel, it saves time and mouse movements, because the mouse pointer is still in the center of the map area and it is not necessary to move to tool bar area. This tool is available in OpenLayers, MapFish, Fusion, ka-Map! and also in GeoMoose, but in this user interface the panning must be turned on.

For panning the most favorite tool is dragging the map. This can be either passive (the functionality is always turned on as in the case of MapFish, OpenLayers, Fusion) or active (the user must turn the functionality on, as in the case of msCross, ka-Map!, GeoMoose, p.mapper). The passive method saves the user's time and also the mouse movements.

Another very useful function is continuous panning without reloading the page. Almost all of us know that feeling when, after only a small movement with the map, we

are impatiently waiting for loading the page, which is sometimes accompanied with an image of a sandglass or a text "Loading page". For the user's comfort it is useful, if the web mapping application can also load the image also outside the current map view and the small movements are not accompanied with waiting. I found the continuous panning without reloading the page in these frameworks: OpenLayers, MapFish, ka-Map!, GeoMoose.

As it is shown in Table 4.1, the tool which is available in all frameworks is the overview map. This map is one of the important functions. It helps the user to see the position of the current view relative to the whole.

Printing tool is used for printing the current view in different formats such as image formats or PDF. The +/- mark in the column of Fusion means, that in the latest version of Fusion, the print widget does not work well. There is a bug.

The other group of basic functions are the layer functions. Layer tree is available in all frameworks except msCross. Layer tree displays the names of layers and sometimes also serves as a legend with samples of symbols (e.g., p.mapper). In two frameworks (MapFish, ka-Map!) the layer reordering is enabled by drag and drop. It is useful, when one layer content covers the other and it is not visible.

The measuring of distances and areas is a function, which is ranked among the common functions in these days, but not every framework supports it. In this selection of frameworks OpenLayers, MapFish, Fusion, GeoMoose and p.mapper support this tool.

The drawing tool is the "cherry on the cake". This tool is not widely used, but sometimes it can be useful to draw a point or line. The drawing tool is available in five frameworks (OpenLayers, MapFish, Fusion, GeoMoose and p.mapper). In p.mapper the drawing tool is little different, it is possible to add a point of interest into the map and to name it. That is why there is +/- mark for this tool in Table 4.1. In two frameworks (MapFish and GeoMoose) it is possible to edit already finished drawing.

msCross was not evaluated in the group of layer functions, because it does not support any of these functionalities.

4.2 Documentation

The documentation is one of the most important parts for the application developer. Unfortunately framework developers sometimes forget that without quality documentation

interesting functionalities can become unused or fall into oblivion.

Documentation covers not only description of particular functionalities, manuals and installation instructions, which unfortunately usually are not comprehensive enough, but also the support from framework developers or other application developers, such as mailing lists and blogs. This kind of support can help to solve difficult problems, which are not described in manuals.

The examples are also a useful part of the documentation, because the application developer can imagine possibilities of the framework thanks to them.

In my opinion, the best documentation of all frameworks can be found on OpenLayers and MapFish web pages. It covers all aspects mentioned above.

4.3 Layout

The default layout of user interfaces can be divided into two groups according to the complexity. The first group includes the framework with default layout of the web mapping application, which can fulfill the whole web page, such as MapFish, GeoMoose, ka-Map! and p.mapper. These layouts are useful, if you do not need to add other frames or texts at the web pages.

The second group includes only default layout with control panels and some frames for a legend or an overview map, such as OpenLayers, Fusion and msCross. These user interfaces do not cover the whole page and that is why it is possible to use them as a part of a web page, which does not contain only the map area.

It is not easy to evaluate the layout, because it is very subjective. In my opinion,the MapFish and p.mapper web mapping applications have the most interesting layout in the first group have . The application as a whole looks modern, clear and legible, the icons are also well designed.

In the second group OpenLayers has the best default layout. The icons are well designed and the hiding and showing of a layer tree and legend is well solved and not disturbing the map area.

4.4 Spread

The spread of particular user interfaces is not possible to be measured exactly. It is not possible to contact definite group of users, because the software is open source and anybody can use it. The spread can be estimated, e.g., on the basis of a number of web pages found by a search engine, in this case Google.com. Google was chosen because nowadays it is the most widely used search engine in the world. In this case the spread of usage of user interface is not estimated exactly, but the result also contains the discussions and other web pages focused on the theme of a particular user interface.

It is not easy to find the right keyword for the search engine, because, e.g., the name of the framework "Fusion" can also be used as a common word. Finally, I decided to use the two words expression: "MapServer" "NameOfTheFramework". Nevertheless, the searching result of keyword "MapServer" "Fusion" was influenced by a project called Cold Fusion which also uses MapServer. That is why I searched the keyword "MapServer" and "Cold Fusion" and subtracted both results.

The spread is also influenced by other factors, such as a period of time during which the framework is available on the "market". In this point of view, OpenLayers and Fusion have an advantage, because both of them were published in 1990's. On cue the most frequently found framework was OpenLayers then Fusion, MapFish, p.mapper, ka-Map!, GeoMoose and msCross. See Table 4.4.

4.5 Application Loading Time

The presumption for measuring of the application loading time in an exact way are equal conditions for tested applications. I tried to ensure these conditions and that is why I created simple web mapping applications, which contain the same data, are stored on the same server, and the measuring was tested at night to avoid the influence of load of the server. The application loading time was measured three times on different days. The application loading time was measured by Firebug 1.4, which is integrated into the Firefox web browser. As it is shown in Table 4.2 there are large differences between measurements of application loading times. The slowest user interface is MapFish and the fastest is ka-Map!. The difference in arithmetic mean of application loading time between these two user interfaces is 10,3 s. The maximum application loading time of all measurements is 24,77 s (MapFish), and the minimum is 1,53 s (ka-Map!).

	Framework						
Details of measuring	OpenLayers	MapFish	Fusion	GeoMoose	ka-Map!	msCross	p.mapper
1.meas. [s] 3.12.09 23:15	4,50	14,43	10,94	5,18	3,81	5,10	5,14
2.meas. [s] 5.12.09 01:50	3,17	6,30	6,97	3,22	1,53	4,42	2,32
3.meas. [s] 6.12.09 00:05	7,96	24,77	5,62	9,85	2,39	5,06	6,04
Arithmetic mean	**5,21**	**15,16**	**7,84**	**6,08**	**2,24**	**4,86**	**4,50**

Table 4.2: Measuring of application loading time

rating	***	**	*
interval [s]	0 - 5	5,01 - 10	10,01 and more

Table 4.3: Evaluating of application loading time

	Framework						
	OpenLayers	MapFish	Fusion	GeoMoose	ka-Map!	msCross	p.mapper
Panning	***	***	*	**	**	*	**
Zooming	***	***	***	**	***	**	**
Layer functions	**	***	**	***	*	-	**
Documentation	***	***	**	**	**	**	**
Layout	***	***	*	***	**	*	***
Loading time	**	*	**	**	***	***	***
Sum of stars	**16**	**16**	**11**	**14**	**13**	**9**	**14**
Spread [no. of websites]	71 400	38 400	47 670	4 640	10 000	3 320	20 800

Table 4.4: Evaluation of using user interfaces

4.6 Summary

The comparison of the user interfaces was made according to different points of view: functionalities of zooming, panning and layer functions, documentation, layout, application loading time and spread, and appreciated by stars from *** to * from the highest to lowest correspondingly in first six categories. Then the number of stars was summed. The spread was not included into the evaluation, because it does not prove unambiguously the quality of the user interface, and it is mentioned only in order to have the full picture.

Table of evaluation 4.4 shows that OpenLayers and Mapfish gained the best rating. From these two frameworks I decided to select MapFish for further usage, because OpenLayers is the remarkable part of MapFish and MapFish has much more possibilities of customizing the functionalities and layout with other parts of MapFish (which are GeoExt and ExtJS), than OpenLayers itself.

Chapter 5

Application of Selected Framework for User Interface

In the previous Chapter 4 the framework were evaluated and on the basis of the evaluation the most satisfactory framework for user interface was selected. Two frameworks had the same number of points, OpenLayers and MapFish, and I decided to select MapFish, because OpenLayers is its remakable part and MapFish has more possibilities to customize the functionalities and layout. The web mapping application using MapFish was created and described in this Chapter.

5.1 Geographic Data Used in Web Mapping Application

The web mapping application consists of the geospatial data of Finland. There are a base map of Europe which is more detailed for Finland and two thematic maps (Structure of built up areas and Areas of Natura 2000), which cover whole Finland.

The shapefiles for the base map are listed in Table 5.1 and for the thematic map are listed in Table 5.2. The content of shapefiles for base map was reduced for needs of the map in Quantum GIS.

Content	**Format**	**Proj.**	**Downloading web pages**
Administrative boundaries of Finland	shapefile	WGS84	`http://www.diva-gis.org/gdatares`
Administrative boundaries of Europe	shapefile	WGS84	`http://sourceforge.net/projects/pmapper/files/`
Roads, railways, rivers, towns of Finland	shapefiles	WGS84	`http://download.geofabrik.de/osm/europe/`
Inland water of Finland	shapefile	WGS84	`http://www.diva-gis.org/gdatares`

Table 5.1: Shapefiles used for base map

Content	**Format**	**Projection**	**Downloading web pages**
Structure of built up areas of Finland	shapefiles	epsg 2393	`http://wwwp2.ymparisto.fi/scripts/oiva.asp`
Natura 2000, Finland	shapefiles	epsg 2393	`http://wwwp2.ymparisto.fi/scripts/oiva.asp`

Table 5.2: Shapefiles used for thematic maps

Finland is for its location projected in Finland uniform coordinate system (Finland zone 3), EPSG projection 2393 (Finland has moved to EUREF-FIN but many datasets are still available in the earlier system). Other projections, which are usually used in middle Europe, distorts the shape of Nordic areas and are not suitable. Because the unusual projection Finland zone 3 was used, it was not possible to find WMS, which provides the base map in this required projection. That was the reason why the base map was made from different shapefiles instead of WMS.

Definition of EPSG:2393 in MapServer mapfile:

```
PROJECTION
"proj=tmerc"
"lat_0=0"
"lon_0=27"
"k=1"
"x_0=3500000"
"y_0=0"
"ellps=intl"
"towgs84=-96.0617,-82.4278,-121.7435,4.80107,0.34543,-1.37646,1.4964"
"units=m"
"no_defs"
END
```

5.2 Mapfile

The mapfile is described in greater detail in Chapter 2.2.1. In the mapfile the design of the map and its layers were defined, such as projection of the map, projection of layers, definition of WMS, extent of the map, symbols of the layer, labels, the minimum and the maximum scale for rendering the layer and type of the layer. The basic mapfile was automatically generated by Quantum GIS and than modified and improved by hand.

5.3 Used Functions

The MapFish web mapping applications are based on OpenLayers, ExtJS and GeoExt JavaScript toolkits. In the final web mapping application there were used basic functions, such as zooming by scale range, zooming by dragged rectangle, layer tree, where the order of map layers can be changed, overview map, drawing tool and shortcuts. In this part I describe some functions of these toolkits, used in the web mapping application.

5.3.1 Creating New Map

To create a map image, the `OpenLayers.Map` constructor must be defined in every web mapping application:

```
var options = {
 controls: [new OpenLayers.Control.MousePosition(),
  new OpenLayers.Control.KeyboardDefaults(),
  new OpenLayers.Control.Navigation(),
  new OpenLayers.Control.PanZoomBar(),
  new OpenLayers.Control.Permalink('permalink'),
  new OpenLayers.Control.ScaleBar({abbreviateLabel: true}),
  ],
  maxExtent: new OpenLayers.Bounds(427919.335405, 5251463.16109,
  4151302.319441, 7778745.046409),
  maxResolution: 10000,
  units: 'meters',
  projection:"EPSG:2393",
  numZoomLevels:15
 };
var map = new OpenLayers.Map( $('center'), options);
```

`OpenLayers.Map` constructor is defined by properties of `div`, which points to HTML structure, where the map will be placed, and options, where the controls are added and there is also the basic definition of the map, such as maximum extent, units, projection and number of zoom levels.

5.3.2 Adding Controls

There are a lot of different map controls provided in OpenLayers. It is easy to add them into the map. As was mentioned above, map controls can be added straight into the options of `OpenLayers.Map` constructor and also can be added in this way:

```
map.addControl(new OpenLayers.Control.Navigation());
```

5.3.3 Adding Layers

The layers of the map were added as WMS. It is also possible to add layers as MapServer layer, Vector layer, WFS layer etc.

```
var bounderies = new OpenLayers.Layer.WMS("Base Map",
                 "http://maps.fsv.cvut.cz/cgi-bin/mapserv?map=/home/
                 havlima6/mapdata/final/map/finland2.map",
                 {layers: 'countries,FIN,rivers,inland_water,towns,cities,
                 rails,primary_roads,secondary_roads,tertiary_roads,
                 motorways',
                 projection: "EPSG:2393",
                 format:"image/png",
                 transparent: true},
                 {isBaseLayer: true} );
map.addLayer(bounderies);
```

The constructor `OpenLayers.Layer.WMS` properties are, e.g., name of the map, url of WMS, layers, which should be rendered in the map and projection of WMS. There should also be stated, whether the layer is base or not.

The layers of WMS are in this case defined in mapfile and also can be found in GetCapabilities request, which in this case looks like this:

`http://maps.fsv.cvut.cz/cgi-bin/mapserv?map=/home/havlima6/mapdata/final/map/finland2.map&VERSION=1.1.1&REQUEST=GetCapabilities&SERVICE=WMS`

5.3.4 Overview Map

The Overview map is created by `OpenLayers.Control.OverviewMap` constructor. The properties of this constructor are various, but the most important are `layers`. It is

possible to a use layer, which was already created, as a layer for the overview map. It is also possible to set the `div` parameter, where the map is placed on the web page. It is recommended to add `mapOptions`, where projection, boundaries and units are set. The overview map is added into the map in the same way like the other controls mentioned in Section 5.3.2, by `map.addControl()`.

```
var options={layers:[boundaries],
             div: $('overviewmap'),
             mapOptions:{units: 'meters',
             projection:"EPSG:2393",
             maxExtent: new OpenLayers.Bounds
             (479419.0, 5251463.0, 5539464.0, 7778745.0)}};
map.addControl(new OpenLayers.Control.OverviewMap(options));
```

5.3.5 Layer Tree

There are different ways, how to create the layer tree. It can be generated automatically, but the layer tree model can also be created. It is the way, of making the most customized layer tree. There can be defined the root nodes of layers corresponding to base map and groups of thematic maps and their children. There is also defined, whether the layer is rendered or not, by `checked: true`, and the path to the legend icon is defined.

The following code is shortened:

```
var model = [{
            text: "Base Map",
            leaf: false,
            checked: true,
            layerName: "Base Map"
            },{
            text: "Natura 2000",
            leaf: false,
            checked: false,
            expanded: false,
            children: [{
                text: "Natura SCI (Sites of Community Importance)",
                layerName: "Natura SCI",
```

```
            leaf: true,
            icon: "icons/natura1.PNG",
            checked: false
        },{
            text: "Natura SPA (Special Areas of Conservation)",
            layerName: "Natura SPA",
            leaf: true,
            icon: "icons/natura2.PNG",
            checked: false }] }];
```

There is also the contextual menu for each layer, which contains possibilities of changing opacity, removing the layer and zooming to extent of the layer, as it is shown in Figure 5.1. This function is enabled by plugin placed in `Ext.Viewport` constructor, which is mentioned in Section 5.3.7.

```
plugins:
 [mapfish.widgets.LayerTree.createContextualMenuPlugin
 (['opacitySlide','remove','zoomToExtent'])],
```

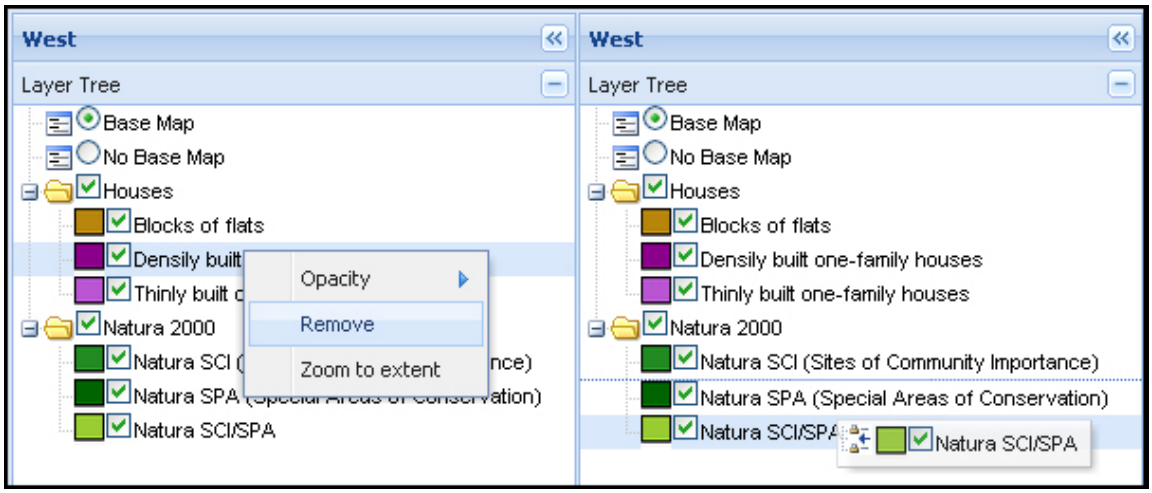

Figure 5.1: Layer tree - contextual menu, reordering layers

5.3.6 Shortcuts

The list of predefined locations to which the map can be zoomed and recentered is called shortcuts. In user interface it is displayed as a combo box, where the location can be

chosen. The shortcut store is created by `Ext.data.SimpleStore`, where shortcuts are defined by name and bounding box by `OpenLayers.Bounds()`. The shortcuts store is than added into the west panel.

```
function createShortcutsStore() {
    return new Ext.data.SimpleStore({
        fields: ["value", "text", "bbox"],
        data: [
        ["E", "East of Finland", new OpenLayers.Bounds(3402400,6806600,
                                                       3906200,7058500)],
        ["M", "Center of Finland", new OpenLayers.Bounds(3174900,6765300,
                                                       3677900,7017900)],
        ["N", "North of Finland", new OpenLayers.Bounds(3248700,7409100,
                                                       3752400,7657200)],
        ["S", "South of Finland", new OpenLayers.Bounds(3116200,6618500,
                                                       3620600,6870400)],
        ["W", "West of Finland", new OpenLayers.Bounds(3096200,6934100,
                                                       3601200,7183500)]
      ] });
}
```

5.3.7 Layout

Layout is provided by ExtJS `Ext.Viewport` constructor. Usually the layout, which suits web mapping applications the best, consists of four regions - north, south, west and center, where the map is placed. In the west region, there are usually widgets, such as layer tree, recentering, overview map and searching tools. In the north region there is usually title and in the south the status bar is placed.

The west region layout is called accordion `layout: 'accordion'`, which describes layout, where it is possible to hide or show particular parts of the panel. There are also some other layouts, such as table layout and border layout.

The property `collapsible: true` defines, that the panel is possible to hide. Some of the properties were not covered in basic ExtJS but were programmed for MapFish in addition, and are connected with ExtJS by property `xtype`, e.g., `xtype: 'layertree'`.

The property `enableDD: true` enables changing the layer order as it is shown in

Figure 5.1.

The following code is shortened and focuses only on the part of west panel:

```
var createViewport = function() {
       var viewport = new Ext.Viewport({
           layout:'border',
           items:[ {
                   region: 'west',
                   id: 'west-panel',
                   title: 'West',
                   split: true,
                   width: 200,
                   minSize: 175,
                   maxSize: 400,
                   collapsible: true,
                   margins: '0 0 0 5',
                   layout: 'accordion',
                   layoutConfig:{animate: true},
                   items: [{
                       contentEl: 'west',
                       title: 'Layer Tree',
                       border: false,
                       xtype: 'layertree',
                       model: model,
                       enableDD: true,
                       plugins:
                       [mapfish.widgets.LayerTree.createContextualMenuPlugin
                       (['opacitySlide','remove','zoomToExtent'])],
                       map: map
                   },{
                       title: 'Overview Map',
                       id:'overviewmap',
                       border: false
                   }] }] }); };
```

5.3.8 Toolbar

To create a toolbar, the GeoExt JavaScript toolkit is used. The toolbar contains zooming and panning options, drawing tools and the navigation history (next and previous view). Particular tools are construct by `GeoExt.Action`. One of the most important options is `toggleGroup`, which associates particular tools into the groups of tools with the same name, where the only one tool can be active. It protects to use, e.g., two drawing tools in one time.

The following code is shortened and focuses only on "zoom to full extent" tool:

```
var createToolbar = function() {
            var action;
            var createSeparator = function() {
               toolbarItems.push(" ");
               toolbarItems.push("-");
               toolbarItems.push(" ");
            };
            action = new GeoExt.Action({
                control: new OpenLayers.Control.ZoomToMaxExtent(),
                map: map,
                iconCls: 'zoomfull',
                toggleGroup: 'map',
                tooltip: 'Zoom to full extent'});
            toolbarItems.push(action);
            createSeparator();
            };
```

5.4 Summary

MapFish framework is a tool for designing user interfaces of web mapping applications. It is possible to customize layout thanks to GeoExt and ExtJS, and use functionalities of OpenLayers and MapFish at the same time.

The final web mapping application with one base map and two thematic maps was designed. It is shown in Figure 5.2, available on `http://maps.fsv.cvut.cz/~havlima6/app/` and its source codes are stored at the enclosed CD. The content of the CD is available in Appendix B.

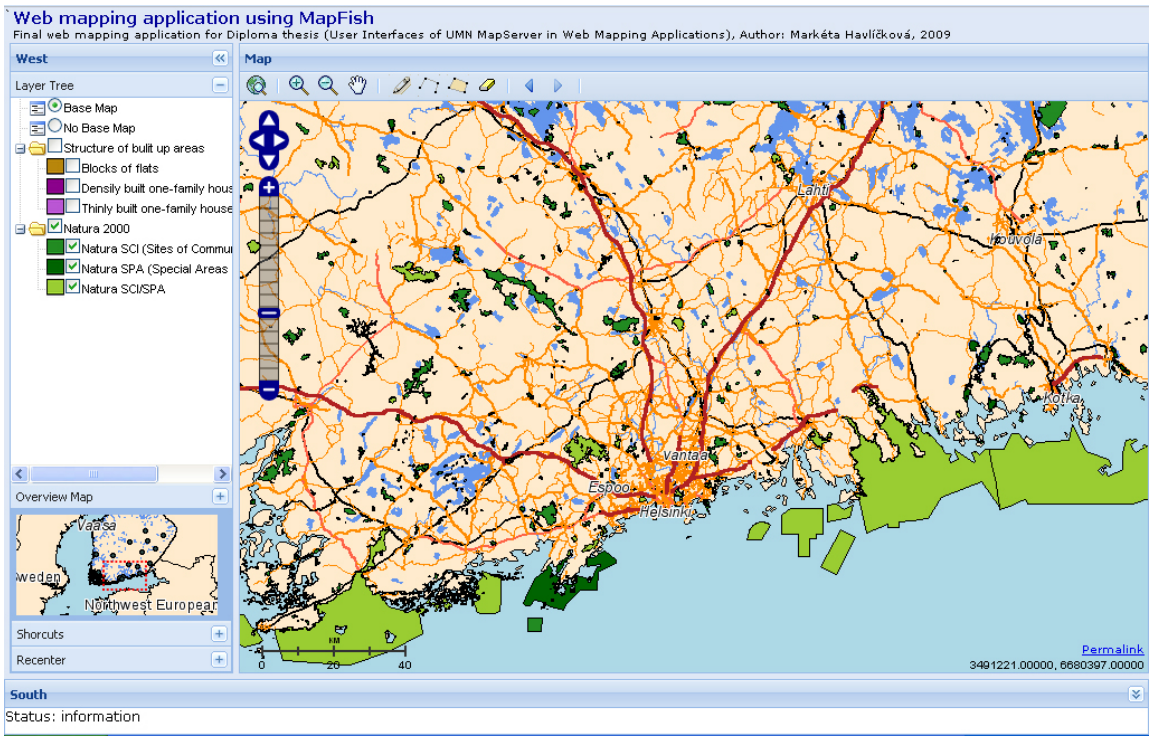

Figure 5.2: Final web mapping application,
Source: `http://maps.fsv.cvut.cz/~havlima6/app/`

Chapter 6

Conclusion

The aim of the thesis was to select the most satisfactory framework, used for designing user interfaces for UMN MapServer, from the frameworks studied in this thesis (OpenLayers, MapFish, Fusion, GeoMoose, ka-Map!, msCross, p.mapper). The selection was based on the description and evaluation of various criteria, such as functionalities, documentation, layout and application loading time. The author's experience with installation and configuration of the frameworks was also included in the evaluation.

The selected framework was MapFish. During the evaluation, MapFish got the best rating in the most of the criteria, only the application loading time was high in comparison with the other user interfaces.

Consequently, MapFish was used for creating a web mapping application, in which the data focuses on Finland. There is one general base map and two thematic maps focusing on the structure of built up areas and areas of Natura 2000.

The user interface contains a central map area, where the mouse position, permalink and graphic scale, toolbar with zooming and panning tools, drawing tools and navigation history are placed. In the left part of web mapping application there is a panel with a layer tree, an overview map, shortcuts and a tool for recentering the map by coordinates. The web mapping application is available on `http://maps.fsv.cvut.cz/~havlima6/`, where also links for the sample web mapping applications for every framework described in this thesis are published.

In the introduction I mentioned, that I would like to know, how difficult it is to create the web mapping application. Creating a basic web mapping application, which contains only WMS layers, following the sample applications and manuals is easy and it can take about one hour.

If you want to create application which differs from sample applications, then there

is a lot to learn, because MapFish is not one integral framework, but consists of other toolkits: GeoExt, ExtJS and Openlayers, and it is required to study GeoExt and ExtJS into greater detail. It is also important to know projections, how the MapServer works, how to create a mapfile. Sometimes it is required to edit the geographic data before using it in the application, such that it is good to know how to work with ArcGIS, or similar software, where the data can be displayed and edited. HTML, CSS and JavaScript are also important for every web mapping application, which uses MapFish.

I hope that this thesis can help users, who are novices in this theme. It can be used as an introduction to education, which focuses on UMN MapServer and designing web mapping applications, such as a course called Web Map Project at the Helsinki University of Technology, Faculty of Engineering and Architecture, Department of Surveying.

Using open source UMN MapServer and open source frameworks, which were described and evaluated in this thesis, is not the only option how to create the workable web mapping application, but for common people, who have knowledge mentioned above, it is definitely the cheapest way. There are also half commercial and commercial solutions.

The half commercial solution consists in the possibility of renting the specialized company (in the Czech Republic e.g., Help Service - Remote Sensing spol. s r.o.), which focuses on designing web mapping applications based on open source frameworks and map servers.

The commercial solutions are map servers, such as MapGuide, ArcGIS server and GeoMedia WebMap, which can be supplied with the functional user interface as a whole.

Bibliography

[1] Armin Burger. *p.mapper - a MapServer PHP/MapScript Framework*[online]. `http://www.pmapper.net/` (accessed Oct, 2009).

[2] Camptocamp. *Instalation - MapFish*[online]. `http://www.mapfish.org/doc/1.2/installation.html` (accessed Oct, 2009).

[3] Camptocamp. *mapfish-MapFish*[online]. `http://www.mapfish.org/` (accessed Oct, 2009).

[4] CRS4 S.r.l.-Center for Advanced Studies Research and Development in Sardinia. *WmsCross: AJAX (WEB 2.0) WEB GIS Client (was UMN Mapserver Javascript interface)*[online]. `http://datacrossing.crs4.it/en_Documentation_mscross.html` (accessed Oct, 2009).

[5] Free Software Foundation. *The GNU General Public License - GNU Project - Free Software Foundation (FSF)*[online]. `http://www.gnu.org/copyleft/gpl.html` (accessed Oct, 2009).

[6] Open Source Geospatial Foundation. *MapGuide Project Home — MapGuide Open Source*[online]. `http://mapguide.osgeo.org/` (accessed Oct, 2009).

[7] Open Source Geospatial Foundation. *OpenLayers:Free maps for web*[online]. `http://www.openlayers.org/` (accessed Oct, 2009).

[8] Open Source Geospatial Foundation. *OSGeo.org, Your Open Source Compass*[online]. `http://www.osgeo.org/` (accessed Oct, 2009).

[9] Mark Harrower and Benjamin Sheesley. Designing better map interfaces: A framework for panning and zooming. *Transaction in GIS*, 9(2):77–89, March 2005.

[10] Markéta Havlíčková. *Districts of the Czech Republic*[online]. `http://maps.fsv.cvut.cz/~havlima6/okresy/okresy_i.html`.

[11] DM Solutions Group Inc. *MapServer and OpenLayers Products and Services - DM Solutions Group Inc*[online]. `http://www.dmsolutions.ca/` (accessed Oct, 2009).

[12] Bill Kropla. *Beginning MapServer Open Source GIS Developement.* Apress, 2560 Ninth Street, Suite 219, Berkley, CA New York, NY 10013, 2005.

[13] Dan "Ducky" Little. *Welcome to GeoMOOSE - GeoMOOSE v2.0.1 documentation*[online]. `http://www.geomoose.org/moose/` (accessed Oct, 2009).

[14] MapTools.org. *ka-Map.MapTools.org*[online]. `http://ka-map.maptools.org/` (accessed Oct, 2009).

[15] MapTools.org. *Preparing ka-Map - ka-Map Wiki*[online]. `http://ka-map.ominiverdi.org/wiki/index.php/Preparing_ka-Map` (accessed Oct, 2009).

[16] Inc. MetaCarta. *License*[online]. `http://svn.openlayers.org/trunk/openlayers/license.txt` (accessed Oct, 2009).

[17] MetaCarta.Inc. *Geographic Search and Referencing Solutions - MetaCarta - At the Forefront of the GeoWeb*[online]. `http://www.metacarta.com/` (accessed Oct, 2009).

[18] Tyler Mitchtell. *Web Mapping.* O'Reilly Media, Inc., 1005 Gravestein Highway North, Sebastopol, CA 95472, 2005.

[19] Regents of the University of Minnesota. *License*[online]. `http://mapserver.org/copyright.html#license` (accessed Oct, 2009).

[20] Regents of the University of Minnesota. *Welcome to MapServer*[online]. `http://mapserver.org` (accessed Oct, 2009).

[21] Edgewall Software. *Fusion-trac*[online]. `http://trac.osgeo.org/fusion/` (accessed Oct, 2009).

[22] Edgewall Software. *mapfish-trac*[online]. `http://trac.mapfish.org/trac/mapfish/wiki` (accessed Oct, 2009).

Used Software

- *UMN MapServer*, version 5.2.4, released under MIT-License
- *Apache*, version 2.2.8, released under Apache License (`http://www.apache.org/licenses/LICENSE-2.0`)
- *PHP*, version 5.2.4., released under PHP License v3.01 (`http://www.php.net/license/3_01.txt`)
- *Mozilla Firefox*, version 3.5.5, Copyright©1998-2008 Contributors, released under the Mozilla Public License (MPL), the GNU General Public License (GPL) and the GNU Lesser General Public License (LGPL) (`http://www.mozilla.org/MPL/`)
- *Firebug*, version 1.4, released under BSD-License
- *PuTTY*, version 0.60, Copyright©1997-2007 Simon Tatham
- *WinSCP*, version 4.1.8, Copyright©2000-2008 Martin Prikryl
- *OpenLayers*, version 2.8, released under BSD-License
- *MapFish*, version 1.1, released under GPLv3 License
- *Fusion*, version 1.1.1, released under MIT-License
- *GeoMoose*, version 2.0.1
- *ka-Map!*, version 1.0, released under MIT-License
- *msCross*, version 1.1.9, released under GNU GPL License
- *p.mapper*, version 3.0, released under GNU GPL License
- *MiKTeX*, version 2.7, released under GNU GPL License

- *TeXnicCenter*, 1 Beta 7.01 - unstable, released under GNU GPL License
- *Quantum GIS*, version 0.11.0-Metis, released under GNU GPL License

Appendixes A

Code of Mapfile Exapmle

```
MAP
NAME "Districts of the Czech Republic"
UNITS meters
SIZE 640 480
IMAGECOLOR 255 255 204
IMAGETYPE jpg
SHAPEPATH "/home/havlima6/mapdata/okresy"
EXTENT -831337.0 -1184398.014085 -559195.0 -971397.985915

WEB
  TEMPLATE "/home/havlima6/public_html/okresy/okresy.html"
  IMAGEPATH "/var/www/tmp/"
  IMAGEURL "/tmp/"
END

LAYER
    NAME "Districts"
    DATA "okresy"
    STATUS default
    TYPE polygon
    LABELITEM "okres"
    CLASS
      NAME "District"
      STYLE
        COLOR 200 132 100
        OUTLINECOLOR 0 0 0
      END #end of the STYLE
      LABEL
        COLOR 0 0 0
        SIZE small
     END                 #end of the LABEL
    END                  #end of the CLASS
END                      #end of the LAYER
END                      #end of the MAP
```

Appendixes B

Content of Enclosed CD

CD is enclosed to this thesis. It contains text of diploma thesis, source codes of mapfile, web mapping application and MapFish and used geodata. In folders with shapefiles only the names with *.shp are stated, but folder contains also other files with posfixes: .prj, .dbf, .shx, etc.

Folder structure:

- Text_of_diploma_thesis
  - Havlickova_Marketa_DP_09.pdf
- Web_mapping_application
  - Data
    - Base_map
      - countries.shp,FIN_adm0.shp
      - rails.shp, motorways.shp, primary_roads.shp, secondary_roads.shp, tertiary_roads.shp
      - rivers.shp, inland_water.shp
    - Natura_2000
      - natura06_region_kohde.shp
      - Metadata in finnish (natura.gif, natura.doc, natura.htm)
    - Structure_of_built_up_areas
      - HarvaPientaloAlue05.shp, KerrostaloAlue05.shp, PientaloAlue05.shp
      - Metadata in finnish (asuinalue.jpg, asuinalueet_tekniset_tiedot.doc, asuinalueet_attr.jpg, asuinalueet.htm)

  - Mapfile
    - finland.map
  - MapFish-1.1 - contains MapFish framework, version 1.1

- index.html - index of final web mapping application
- list_of_links.html - list of links of sample and final web mapping applications
- style.css