Analýza webových technologii pro tvorbu grafických výstupů energetických podniků

Analysis of graphical web technologies for purposes of online data visualisation in IS Energis
Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

Sample pictures of IS Energis component are property of Instar ITS Company

In Ostrava, May 7th 2010 .................................

I declare that I have wrote this thesis alone. I document all the literary sources and publications, from which I gathered.

In Ostrava, May 7th 2010 .................................
Thanks to Instar ITS company, which gave me the possibility to write this work about their great product, especially thanks to Carl Kuzelka (Go Carl!). Thanks to my parents and my brother, who gave me precious notes and thanks to my Love Silvia. Thanks also to my thesis tutor Ing. Martin Němec, Ph.D. for his patience and advices.
Abstrakt
Tato práce se zabývá možnostmi vizualizace dat v online prostředí webových prohlížečů. Obecně popisuje technologii a principy webových prohlížečů, analyzuje některé W3C standardy a zkoumá možnosti vizualizace dat v prostředí průmyslových podniků. Práce dále konkrétně analyzuje technologie vizualizace dat v informačním systému Energis, popisuje dřívější technologie zobrazování dat. Na základě porovnání vizualizačních technologií jsou implementovány ukázkové komponenty, porovnány jejich možnosti a vyvozeny požadované závěry.

Klíčová slova: vizualizace dat, grafy, JavaScript, FLASH, Java, SilverLight, SVG, HTML, Canvas, XML, AJAX

Abstract
This thesis deals with web browser based data visualization possibilities. It describes technology and common principles of web browser, analyses some web W3C standards and examines data visualization possibilities for purposes of industrial companies. Thesis is focused on data visualization of information system Energis and describes older solution. Work also analyses possible new visualization technologies and reconsider its capabilities and provides result.

Keywords: Data visualization, chart, JavaScript, FLASH, Java, SilverLight, SVG, HTML, Canvas, XML, AJAX
List of abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJAX</td>
<td>Asynchronous JavaScript and XML</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AWT</td>
<td>Abstract Window Toolkit</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>DLL</td>
<td>Dynamic-link library</td>
</tr>
<tr>
<td>DOM</td>
<td>Document Object Model</td>
</tr>
<tr>
<td>GNU</td>
<td>GNU’s Not Unix!</td>
</tr>
<tr>
<td>GPL</td>
<td>General Public License</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>GWT</td>
<td>Google Web Toolkit</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>Hypertext Transfer Protocol Secure</td>
</tr>
<tr>
<td>IE</td>
<td>Internet Explorer</td>
</tr>
<tr>
<td>IS</td>
<td>Information System</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>JRE</td>
<td>Java Runtime Environment</td>
</tr>
<tr>
<td>JS</td>
<td>JavaScript</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>LGPL</td>
<td>Lesser General Public License</td>
</tr>
<tr>
<td>MSJVM</td>
<td>Microsoft Java Virtual Machine</td>
</tr>
<tr>
<td>PNG</td>
<td>Portable Network Graphics</td>
</tr>
<tr>
<td>RGB</td>
<td>Red Green Blue</td>
</tr>
<tr>
<td>SVG</td>
<td>Scalable Vector Graphics</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>VML</td>
<td>Vector Markup Language</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition</td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction 5
  1.1 Thesis assignment ........................................ 5

2 Current solution of data visualization 7
  2.1 Architecture of IS Energis ................................ 7
  2.2 Background and History of development .................. 7
  2.3 Used applets overview .................................... 8

3 Analysis and Requirements 11
  3.1 Analysis of the environment ................................ 11
  3.2 Data analysis .............................................. 12
  3.3 New data structure definition ............................. 14
  3.4 Old Visualization components analysis .................... 16
  3.5 Requirements ............................................... 17

4 Web browsers 19
  4.1 Definition .................................................. 19
  4.2 Browsers overview ......................................... 20
  4.3 JavaScript and AJAX ....................................... 21
  4.4 Web browser graphical possibilities ....................... 23
  4.5 Browsers plug-in system ................................... 26
  4.6 Selected Technologies ..................................... 27

5 HTML5 canvas element & exCanvas library 29
  5.1 Technology overview and main features .................... 29
  5.2 Example graph component .................................. 31
  5.3 Summary ..................................................... 33

6 FLASH 37
  6.1 Technology overview and main features .................... 37
  6.2 Example graph component .................................. 38
  6.3 Summary ..................................................... 41

7 Java Applets 43
  7.1 Platform overview and main features ....................... 43
  7.2 Example graph component .................................. 44
  7.3 Summary ..................................................... 47

8 Silver Light .NET 51
  8.1 Platform overview and main features ....................... 51
  8.2 Example graph component .................................. 52
  8.3 Summary ..................................................... 54
9 Online demo Site
  9.1 Technology .............................................. 57
  9.2 Content .................................................. 57
  9.3 Data source for sample components ..................... 57

10 Results ...................................................... 59
  10.1 Tested features .......................................... 59
  10.2 JavaScript with HTML5 Canvas or (SVG/VML) ........... 59
  10.3 Flash ....................................................... 60
  10.4 Java ......................................................... 61
  10.5 Silverlight .................................................. 61
  10.6 Final result ............................................... 62

11 Conclusion ................................................... 65

12 Reference ..................................................... 67
List of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Energis functions</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>Trend component</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>Gas Consumption component</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>State diagram - summarize possible component states</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Electricity consumption component</td>
<td>17</td>
</tr>
<tr>
<td>6</td>
<td>Simple JavaScript &amp; DOM elements Bar graph</td>
<td>25</td>
</tr>
<tr>
<td>7</td>
<td>Browser plug-in component diagram</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>Canvas sample component - electricity consumption (kWH) - line chart</td>
<td>33</td>
</tr>
<tr>
<td>9</td>
<td>Canvas sample component - electricity consumption (kWH) - bar graph</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>Flash sample component - electricity consumption (kWH) - line chart</td>
<td>41</td>
</tr>
<tr>
<td>11</td>
<td>Flash sample component - electricity consumption (kWH) - bar graph</td>
<td>42</td>
</tr>
<tr>
<td>12</td>
<td>Java applet sample component - electricity consumption (kWH) - bar chart</td>
<td>48</td>
</tr>
<tr>
<td>13</td>
<td>Java applet sample component - electricity consumption (kWH) - line chart</td>
<td>49</td>
</tr>
<tr>
<td>14</td>
<td>Silverlight sample component - electricity consumption (kWH) - line chart</td>
<td>55</td>
</tr>
<tr>
<td>15</td>
<td>Silverlight sample component - electricity consumption (kWH) - bar graph</td>
<td>56</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Thesis assignment

Analysis of graphical web technologies for purposes of on-line data visualization in IS Energis.

An important part of an IT solution of energetics concerns is measured data visualization (graph, charts, etc.) The purpose of this thesis is to find out the requirements for the visualization task, analyse the possibility of web graphics technologies and to design optimal solution.

- Analyse the current data visualization solution of IS Energis and consider advantages and disadvantages.
- Choose relevant web technologies and examine their capabilities for on-line data visualization.
- Using chosen technology implement sample graphic components and reconsider its possibilities.

This thesis is dedicated to the Instar ITS company, which develops information system called Energis. IS ENERGIS is a comprehensive SW tool for enterprise business process control (production balances, planning, billing, cost allocation, technical controlling, production management, business process modeling). IS ENERGIS is useful especially in power generating, utilities and industrial enterprises, the management of which is interested in the maximum efficiency of production, distribution and sales. IS ENERGIS supports all levels of company management. It has been created using the up-to-date technologies so that it can meet demanding requirements of users on quality of provided information, system openness and flexibility.[1]

ENERGIS main features:

- Effective management of the energy production and distribution
- Objective calculation of energy consumption costs within the facility (or company-wide)
- Minimize the loss of energy during distribution by identifying leakage points
- Optimization of resource planning function
- Ensure an integrated global data-acquisition environment
- Centralize maintenance dispatching for the entire company
- Effective control of distribution network (steam, hot water, electricity, water, gas, etc)
Obrázek 1: Energis functions

- Maximize machinery usage / up-time
- Meet environmental regulations

[1]
2 Current solution of data visualization

2.1 Architecture of IS Energis

Energis is effective tool for energy management solution. Energis collects and stores data for various purposes of decision support and industry controlling systems. Depending on user it provides different views, data sets and data visualization components focused on relevant data needed by certain user case.

System has classical 3-layer architecture: database, logical layer and visualization layer. This thesis is focused on visualization layer. Visualization is based on HTML and XML languages and HTTPS protocol. The most common part of data visualization like text information or different tables or forms are provided by HTML. More complex visualization components supports Java applets technology. Complex charts, graphs, data tables and their combination creates multi-purposes tool for creating user or user cases visualization.

2.2 Background and History of development

User friendly data visualization is one of many Energis purposes which Instar ITS developed for many years. Development in every computer science or computer related branches is generally very rapid and every year getting even faster. But when we speak about Internet technologies and on-line data visualization techniques, we can say that the tempo of revolutionary changes and different kinds of fascinating evolutions is even greater. From the beginning one of the main goal of IS Energis was to create set of flexible easy-use components, aimed for rapid user pages creating. Since beginning, visualization technology of Energis was HTML language combined with Java applets. Those days Internet consist only of static text, forms, few pictures and other graphical-poor features. Flash technology was too young and barely known, immature, with unclear future. Other nowadays visualization technology e.g. Microsoft’s .Net platform was distant future and Microsoft had no means of their own graphical on line visualization technology suitable HTML and so they support Java platform. From the beginning of Java applet technology Microsoft’s IE support this revolutionary technology and soon create its own, controversial, but little bit faster clone of Java virtual machine implementation. Java applets spread fast through web. On the basis of these facts Java technology was right choice.

From the beginning Java has following advantages:

• Java was designed to be easy to use and is therefore easy to write, compile, debug, and to learn.

• Java is object-oriented Which allows us create modular programs and reusable code.

• Java is platform-independent. Platform indecency and its ability to move easily from one computer system to another is one of the most significant advantages of Java. This ability to run the same sources on many different systems is crucial
to World Wide Web software. Java succeeds also by being platform-independent at both, on the source and binary level.

Because of Java’s robustness, ease of use, cross-platform capabilities and security features, it has become a language chosen for providing worldwide Internet solutions.

Strategic decision to choose Java platform for visualization layer of Energis was logical consequence of shown facts and advantages. Nevertheless there was still two version of Java to choose from – Microsoft VM and Sun Microsystems Java virtual machine. Microsoft took opportunity and implement its own very fast virtual machine and include it to its browser Internet Explorer 3 by the Sun Specifications. Thou next version of Internet Explorer 4 did not implement Sun’s most recent set of Java specifications and Java API despite they was obligated to do so, which cause that Sun Microsystems, the developer and licensor of Java Technology, suit Microsoft. USA justice issue that it is considered as attempt to fragment the standardized application environment, and break with cross platform compatibility. Final decision was made on the background of these court wars for Microsoft VM, because in its golden age MS VM was faster than Sun Virtual machine. Visualization components based on class Applet and AWT was compiled using 1.3 Java version. Over the years of development it became obsolete and finally Microsoft issued a statement that they are going to support the MSJVM until December 31, 2007 when the MSJVM reached the end of its life cycle and it will be no longer available for distribution from Microsoft and there will be no enhancements to the MSJVM. From the year 2008 all the visualization components used in Energis are slightly redesigned and recompiled in up to date version of Java. But historical reasons and backward compatibility with older Java versions demands its workaround and could be drawback for future evolution.

2.3 Used applets overview

Visualization is based certain visualization components. To know the needs of visualization components we have to analyze current graph solution and find out requirements and conditions for future development. Each component consist of chart (bar graph, line chart, etc.), data table and other control and information elements.

2.3.1 Electricity consumption component

First graph component displays one or more plots or curves for selected time span. Graph is useful to data examination and data comparison. Whole component is shown at figure 5 on page 17

2.3.2 Trend component

Second component displays one or more plots or curves in real time for selected Date and time. The data are periodically updated and displayed. Whole component is shown at figure 2 on page 9
2.3.3 Gas consumption component

Third component displays energy day schedule of take out of different kind. It shows the plan and real take out and issue recommendations. Whole component is shown at figure 3 on page 10.
Obrázek 3: Gas Consumption component
3 Analysis and Requirements

To understand the main goal of this work is is necessary to describe environment in which components are integrated and how does the interfaces looks like.

3.1 Analysis of the environment

The main goal is to find out best technology suitable for designing new set of visualization components, which should replace old on line data visualization solution. Therefore chosen technology or group of compatible web technologies and protocols must be capable of visualizing data within web browser user friendly and modern way. New solution also must be capable of integration to current data interface structure.

Visualization components of IS Energis offers complex, volatile and scalable extensions. Among default and others possibility every user or user group also have possibility to create its own visualization page displaying data from IS Energis its own way to satisfied different approaches and needs. There are some templates and common use case scenarios, but usually it is not "out of the box" solution. It is possible to say that IS Energis provides platform for creating simple HTML pages or complex web applications.

As we mentioned above IS Energis is classical 3 layer architecture Information system. Visualization layer offers both statical data sets and overview and custom made user visualization pages.

At the first sight, architecture of on line application is divided into two main parts: Server and client. Amount of dependency of these two parts may vary in large scale. On one hand, there are technologies, capable of creating complex web applications, where client side is strongly dependent on server side e.g. ASPx.NET or Google Web Toolkit (GWT). These server centered systems produces and compile almost all part of on line application on the server side, send it to client side and waits for user interaction. When the page is changed whole page is routed back to server where the changes are processed. ASP.NET technology use term life cycle of the web page or user components. Disadvantages of this approach are high traffic between server and client and absolute dependency on the server side and its technology.

On the other hand different approach consist of relatively independent client side using different stand alone components and other web technologies which glue client side together. In this scenario server side serves as storage of components which are deployed to the client side and as data source for the component. User actions are processed on client side and server is requested only for structured data. This component architecture bring more freedom to custom web site creation.

New IS Energis visualization technology must be able to create stand alone reusable component with possibility of integrating to any web pages, therefor web components must be independent to server side.

Since users can access IS Energis through Internet form different computers with different operational system and also web browser, visualization technology should be penetrated as much as possible and easily reachable. This feature is very important because in some cases customer has its own network set up and secured by corporate security po-
licy and any additional software have to be permitted and also installed. Speaking about big customers it is a lot of time and money.

3.2 Data analysis

3.2.1 Old data format

Old Java applet components use different data format e.g. plain text, XML and their combinations. Also each component has its own and different way of communication. Communication scheme has changed during development many times Early stages of data transport consist of socket communication and HTTP protocol which transport plant text data. Later components supports HTTP protocol transferring structured data using XML documents and older components communication was also rewritten. Current transportation content uses XML but it is not defined in any way nor anyhow restricted, data are not very systematically placed into XML nodes attributes and to the nodes it self. XML format was developed ad-hoc only by immediate needs.

On load the visualization components requests and load titles and description strings, translation strings depending on selected language and other statistical data needed for its functions. After the component start up and initialization start periodical data request. Components are refreshed different way, but typically it gets measured values with time stamps and other value describing its validity status given by measuring device. Each validity status describe value certain predefined way. For State diagram See figure 4 on page 12.
**Component - server communication example**  Example of visualization component communication with server

1. Request access control
   Request is send using HTTPS Get with parameters: session=123, user=alsh

   ```xml
   <prava>
     <u100 access='1' />
   </prava>
   ```

2. Requests translation
   Request is send using HTTPS Get with parameters: text=id.;id.;storno ...

   ```xml
   <data>
   <setting>
     <colorset>red|green|blue</colorset>
     <startDate>200808261215</startDate>
     <samplesCount>96</samplesCount>
   </setting>
   <preklad>
     <texty lang="cz">
       OK
       Storno
       Datum
       Období
       Hodnota
     </texty>
   </preklad>
   </data>
   ```

3. Requests Values
   Request is send using HTTPS Get with parameters: merak=123, od = 101020090000000, do = 10102009234500 Component gets data describing electricity compsumtion for interval of one day with granularity one quarter hour. XML Document consist of root element data, which contains others nodes named by id of the electricity measure device which supply measured value.

   ```xml
   <data>
     <u100 date='10102009000000' value='235.4' status='0' />
     ... and other 94 similar nodes which differs by value status and time stamp
     <u100 date='10102009234500' value='235.4' status='0' />
   </data>
   ```
3.2.2 Amount and frequency of data requests

Data refresh period is dependent on user choice. Visualization components are able to display selected time period span, real time values or are able to combine first two possibilities. This selected serves for comparison of consumption of electricity. User choose time period which want to see and chose granularity of samples. For example for examining electricity consumption it is useful select granularity of X axis as quarter hour. Visualization component displays on or more plots eventually show other limit e.g. hour limit. In this case amount of data for this component for one day and one plot is sum of 96 value, time and status records. It means 96 elements of above mentioned XML for one page reload. When this sample component is set up for real time value displaying, periodical refresh contains only new value samples.

3.3 New data structure definition

For purposes of this thesis I have created randomly generated data source, but with strictly defined data structure and its restriction. Creating random data source is described in 9. New XML data format consist of root element `data` with count, max and min attributes. Root element contains number of `sample` elements specified by count data attribute. Sample element represent one point or bar of the visualized data. Sample element contains its id and order attributes. Id attribute specifies sample source and order define order of the sample values. Each sample node has exactly three child nodes. Node value contains Y-axis value of the graph, color child node specifies color as an RGB triplet in hexadecimal format and info node, which carries information of data description.

Here is an example of data sample. Each set of samples is enveloped by root element `data`. Data element consist of randomly chosen samples nodes. Data attribute count specifies count of the samples, max attribute specifies max sample value and min specifies minimal value of all samples.

```xml
<?xml version="1.0" encoding="utf-8"?>
<!--XML document version and encoding must be on first line -->
<!--root node with count, max and min attributes-->
<data count="6" max="87" min="24">
  <!--sample node with id and order attributes-->
  <sample id="0" order="0">
    <value>86</value> <!-- sample value -->
    <color>#4F8B5C</color> <!-- sample color -->
    <info>info No0</info> <!-- additonal information -->
  </sample>
  ... <!-- others sample nodes -->
</data>
```

New example is more precisely described by following XSD:

```xml
<?xml version="1.0" encoding="utf-8"?>
```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<!-- definition of the root element data -->
<xsd:element name="data">
  <!-- data node is defined as complex type and could contains other child nodes -->
  <xsd:complexType>
    <!-- complex type can contain sequece of followed elements -->
    <xsd:sequence>
      <!-- definition of the sample element. Element must be presented at least once, but most 1024 times -->
      <xsd:element name="sample" minOccurs="1" maxOccurs="1024/">
        <xsd:complexType>
          <xsd:sequence>
            <!-- define the value element as a string type. -->
            <xsd:element name="value" minOccurs="1" maxOccurs="1" type="xsd:string"/>
            <!-- define the color element as a string type -->
            <xsd:element name="color" minOccurs="1" maxOccurs="1" type="xsd:string"/>
            <!-- define the info element as a string type. -->
            <xsd:element name="info" minOccurs="1" maxOccurs="1" type="xsd:string"/>
          </xsd:sequence>
          <!-- define sample attribute id as an integer type -->
          <xsd:attribute name="id" type="xsd:integer"/>
          <!-- define sample attribute order as an integer type -->
          <xsd:attribute name="order" type="xsd:integer"/>
        </xsd:complexType>
      </xsd:element>
      <!-- count attribute is integer type of the data node -->
      <xsd:attribute name="count" type="xsd:positiveInteger"/>
      <!-- max attribute is integer type of the data node-->
      <xsd:attribute name="max" type="xsd:positiveInteger"/>
      <!-- min attribute is integer type of the data node-->
      <xsd:attribute name="min" type="xsd:positiveInteger"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
</xsd:schema>
Data description using XSD is a precise way how to define XML documents. Visualization components are designed on the basis of data structure, which must be clearly defined. Describing data exchange using native language would be difficult and exhausting, and also confusing and imprecise. There are more ways how to strictly define data structure, but XSD definition is very convenient, standardized and easy to use. Advantage is also fact that it is written in XML which provide wide scale of powerful checking and validating tools. Data itself and data description have the same format. XSD description is very useful not only for describing the data structure, type and their restrictions, but it provide also easy way how to check data integrity and data validation. Validation of XML documents demands XML validity at the first place. This means that document consist of the only one root element, all elements must be properly nested and all entities have to be well-formed. Satisfying this conditions exchanged XML documents can be validated against selected XSD scheme.

3.4 Old Visualization components analysis

IS Energis offers variety of visualization components. It utilize simple and small applets, which simply displays demanded values or images, but also complex and sophisticated gadgets and interactive tools. Most of them shares some common functionalities e.g. data refreshing and handling system, text styling and formatting or others reusable features. In general we can say the more complex component is the more visualization features it demands. For detail examination of the components requirements, lets focus on the most complex one of them. Specifying the needs of component with the most features should picture desired criteria for the new technology.

Lets focus on one of the most feature-rich component at figure 5 on page 17. This component consist of two main parts. First is the chart displaying few plot lines and table containing data. We can also see labels, buttons, pictures and other common GUI parts. Additional functionality provides tab panel and its control section located at the top left corner, which switches tabs between data table and chart. By default setting chart tab displays line chart, but user is allowed to change chart type. Component displays line, bar, or point chart. Chosen component part are marked by numbers in purple circles. Here is the description of each part:

1. Tab containing chart
2. Tab containing data table
3. Axis Y description
4. Chart line
5. Background grid for better user experience.
6. Value of selected sample.

7. Scroll Bar - scrolls samples

8. Data description

9. Axis X

10. Date selection

11. Displayed data description

12. Chart type: bar/line

13. Setup button

14. Axis X description

15. Graph label

3.5 Requirements

Criteria for chosen technology have to fulfill following criteria:
3.5.1 Technology criteria

Must be web browser based technology. The most important condition is that Web browser provides runtime environment for selected technology or exist some trustworthy and easily reachable plug-in system for all the most used browsers.

Must be capable of sufficient fast graphical operations. Very important is possibility of drawing basic shapes primitives as point, line, circle, polygon and others. Technology should be able of real-time data displaying or handling larger data set in short time.

Must combine drawing functions and GUI parts together. To create useful and user friendly components technology have to add text description of the chart axes or show tool tip on plot line.

Must be able of user interaction handling. Technology have to react properly to GUI parts as well as should handle user interaction with drawable area.

Necessity is also implementation of HTTP/HTTPS protocols. Each component have to communicate with server it self or using browser functionality.

Must support Standardized data exchange formats like XML, JSON or its clones to easy and seamless integration component to current system interfaces.

Must support play sound. For better user user experience components must play sound for purposes of different alarms and other warning.
4 Web browsers

IS Energis visualization is based on web browser. To choose proper new technology it is necessary to describe browsers features and visualization possibilities.

4.1 Definition

A web browser is a software application for retrieving, presenting, and traversing information resources on the World Wide Web. An information resource is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video, or other piece of content. Hyper links present in resources enable users to easily navigate their browsers to related resources. Although browsers are primarily intended to access the World Wide Web, they can also be used to access information provided by Web servers in private networks or files in file systems. [2].

4.1.1 Function

A Web browser is a software application which enables a user to display and interact with text, images, videos, music, games and other information typically located on a Web page at a Web site on the World Wide Web or a local area network. Text and images on a Web page can contain hyper links to other Web pages at the same or different Web site. Web browsers allow a user to quickly and easily access information provided on many Web pages at many Web sites by traversing these links. Web browsers format HTML information for display, so the appearance of a Web page may differ between browsers. Web browsers are the most-commonly-used type of HTTP user agent. Although browsers are typically used to access the World Wide Web, they can also be used to access information provided by Web servers in private networks or content in file systems. [2].

From certain point of view modern web browsers provides common framework and application programmer interface for other technologies, which use browser as lower layer to build web applications by providing all basic functions e.g. data transmitting, images and other multimedia formats handling, security and others. Modern browser are also extensible by various plug ins or modules, which provide specified functionality. Core of the browser responsible for page lay-outing called layout engine, decide how to layout objects on the HTML page. Modern approach to developing web browsers use web browser cores developed, easily separable from the other parts of browser. Overall behavior of the web browser is dependent on combination of browser core JavaScript engine and other parts.

Discussed issue of today’s web technologies is different interpretation of defined web standards. Nowadays, there is a lot of interesting new technologies, usually based on XML language or other standards, which usability is being hold because of different implementation and behavior more or less in each browser. Standardizing organizations and work groups issues various recommendations, but real implementation depends on each browsers implementors. To depict the situation, list of main browsers and used lay-out engines follows.
4.2 Browsers overview

**Internet Explorer**  Windows Internet Explorer is the primary graphical web browser developed by Microsoft for the Windows Operating System. The most recent release of IE is Internet Explorer 8, released in March of 2009. IE family is the most used web browser because of its integration in Windows OS. IE is based on Trident layout system.

**Mozilla Firefox**  Firefox is a Web browser created by Mozilla Corporation. It is the second most used browser on the market. Firefox open source system is based on Gecko Lay out engine and SpiderMonkey JavaScript interpret.

**Google Chrome**  Google Chrome is a web browser developed by Google that uses the WebKit layout engine and application framework and V8 JavaScript engine. Chrome is developed as an open source project.

**Safari**  Safari is a web browser developed by Apple. It uses Apple’s WebKit for both rendering web pages and running JavaScript.

**Opera**  Opera 10 is the latest edition of the Opera web browser. Opera uses Presto layout engine and JavaScript Futhark or Core engines.

**Layouts Engines and JavaScript Interprets**

**WebKit**  WebKit was originally created as a fork of KHTML as the layout engine for Apple’s Safari; it is portable to many other computing platforms. WebKit’s WebCore and JavaScriptCore components are available under the GNU Lesser General Public License, and the rest of WebKit is available under a BSD-style license.

**Trident**  Trident (also known as MSHTML) is the name of the layout engine for the Microsoft Windows version of Internet Explorer. In latest versions of Internet Explorer, Microsoft made significant changes to the Trident layout engine to improve compliance with web standards and add support for new technologies, but there is still a lot of issues. With version 5.0 of Trident, Microsoft intends to comply with many modern web standards, and also intends to significantly update the layout engine to be more competitive and modern compared to other current layout engines.

**Gecko**  Gecko is a layout engine currently developed by Mozilla Corporation, known as the layout engine of the Firefox web browser and Mozilla Thunderbird. It is designed to support open Internet standards, and is used by applications such as Mozilla Firefox, Gecko offers a rich programming API that makes it suitable for a wide variety of roles in Internet-enabled applications, such as web browsers, content presentation, and client/server. Gecko is written in C++ and is cross-platform. Licensed by a tri-license of
the Mozilla Public License (MPL), GNU General Public License (GPL) and GNU Lesser General Public License (LGPL).

**Presto** Presto is a layout engine for the Opera web browser developed by Opera Software. After several public betas and technical previews, it was released on January 28, 2003 in Opera 7.0 for Windows; it is the browser’s current layout engine. Presto is dynamic: the page or parts of it can be re-rendered in response to DOM and script events. Presto is available only as a part of Opera browser or related products. The source or binary (DLL) forms of the engine are not publicly available.

4.2.0.1 **SpiderMonkey** SpiderMonkey JavaScript engine written in C and contains a compiler, interpreter, decompiler, garbage collector, and standard classes. It does not itself provide host environments such as Document Object Model (DOM). SpiderMonkey and its sister engine Rhino have implemented support for the ECMAScript for XML (E4X) standard. [13]

4.3 **JavaScript and AJAX**

4.3.1 **JavaScript**

The name JavaScript is protected trademark of the Sun Microsystems company, that is why standardization organization use therm ECMAStandard. For many people it become synonymous names and so for the purposes of this work, will be used like this. JavaScript, despite the name, is essentially unrelated to the Java programming language, although both have the common C syntax, and JavaScript copies many Java names and naming conventions. The language’s name is the result of a co-marketing deal between Netscape and Sun, in exchange for Netscape bundling Sun’s Java runtime with their dominant browser. The key design principles within JavaScript are inherited from the Self and Scheme programming languages. Latest JavaScript is a superset of the ECMA-262 Edition 3 standard (ECMA3Script) issued by Standardizing Information and Communication Systems. It is prototype based, dynamic scripting language, which was developed by Netscape company. JavaScript is scripting language, which needs an scripts commands interpreter. The most common host environment for JavaScript is web browsers. Web browsers typically use the public API to create ‘host objects’ responsible for reflecting the DOM into JavaScript etc. There is no common commands interpreter and each JavaScript engine implements its own. Behavior of certain script could more or less differ in each one of them.

Since its beginning, many people consider JavaScript as useless language for web. It is true that, if we compare it with many others programming language JavaScript still look very simple and has some issues, but despite that, JavaScript proof it self worthy. Modern trends as WEB 2.0 and fancy graphical user effects raised the interest about server scripting side enormously. Thou with rapid Internet and development on-line application, its important grow even more. Last widely used version of JavaScript was defined in 1999 as ECMA3Script 3. It was not designed for today purposes of on line applications nor for
complex JS frameworks. But as Js become interesting also for big players of on line world, we can see more and more effort in evolving JS to much mature programming language concepts and frameworks, which meets web application programmers needs.

**JS frameworks** As JavaScript became more popular, many new JavaScript frameworks raised. Framework is a bundle of programming code designed for specific problem domain. Helping programmer solving common and repeating problems by set of specified API or abstract methods. Basic feature of each JavaScript toolkit is to encapsulate the differences of ambiguous browsers environment. Many of them offers additional functionality such as charting and advanced lay out handling.

- Prototype
- jQuery
- Dojo Toolkit
- YUI - The Yahoo! User Interface
- GWT - Google Web Toolkit

**Acid test** Acid test is set of web compatibility test. Its curious name comes from way of testing valuable metals from the age of golden rush, where quality of gold or other rare metal was rated using acid. Acid tests are based on W3C recommendations, but they do not tests each one of the specified features. Acid are not completely exhausting tests. Each Acid generation just choose the main specification browser failures from most of contemporary browsers. In the time of browser issue, browser rating is low. This approach should help all browser developers to focus on greatest specification implementation errors and render browser more interpretable. On the other side perfect match in Acid test is not proof that browser implements specification completely. Tests which examine the W3C specification exhaustingly have to check one features by another. Such test is e.g. CSS Selector test. The Acid tests are standard tests to rate the performance of a browser. Upon each test’s release, they are designed so that no existing browser can pass without further development. In order for a browser to pass any Acid test, the latest public release of the browser (not an alpha, beta, release candidate, or other version under development or testing procedures) must meet the requirements shown below. In addition, the browser should be tested upon completion of installation, with no add-ons installed (some browsers make this easy by providing a “safe mode” option) and all the factory settings (no options have been changed from their defaults).

- Acid1 Final rendering looks exactly like the rendering provided by the Acid tests website. Text can be highlighted and radio buttons can be selected.
- Acid2 Final rendering looks exactly like the rendering provided by the Acid tests website. Smiley’s nose turns blue when hovered over.
• Acid3 Final score of 100/100. No error messages on final rendering. Render-in-progress loads smoothly (no pausing). Final rendering looks exactly like the rendering provided by the Acid tests website.

4.3.2 AJAX

Examining the abbreviation AJAX we can say it is just combination of JavaScript and XML, but on its way to became very popular, AJAX expression was extended by second, more important, meaning. As well as genuine meaning it is also certain approach to the dynamic web development. Nowadays AJAX is the synonymous for development of interactive web applications, which refresh and changes itself without prolonging reloading of whole page. This new model is more user friendly and the AJAX based website is more desktop like, but it is also more demanding for programmer skill and time of development as well as it need new skills and modern technologies.

Technologies used by Ajax:

• HTML (XHTML) and CSS for data formatting and visualization
• DOM and JavaScript dynamically creating and editing web page, handling and processing data
• XMLHttpRequest object responsible for synonymous or asynchronous web server calls. Basically supports XML document, JSON, simple text or others, but real support may vary depending on browser implementation.
• XML Standardized representation of data structures.

4.4 Web browser graphical possibilities

Each browser is possible to create graphical outputs by its basic native functions, but depending on the browser it could be challenging and still not very usable.

4.4.1 JS charting possibilities

For purpose of this thesis lets focus on basics charting. It is possible to draw a chart using certain HTML elements. In case we want to draw a bar graph, the HTML div combined with img element it is sufficient enough for this purposes. Basic component could be div with desired width, height and style, which is parent element for others graph parts. Certain bars could be than created by others well styled divs elements as shows figure 6 on page 25. First step is to find out the size of chart area. This dimension could be specified by user or inherited from parent HTML node. To measure the parent dimensions we can use the scrollWidth and scrollHeight attributes of HTML tags. It is necessary to set overflow attribute of measured parent element to hidden and because of Internet Explorer compatibility, set fontSize attribute to 0px size. Than depending on count of samples and space to draw in, we need to examine bar width and spare some space to axis and description tags. Important part is also to specified the DTD for which the graph will be
used for. It is well known that different browsers renders many tags different way and interpret various attributes not by standard. Advantages of pure JavaScript technologies for charting is its absolute compatibility with native browser language and environment. Graph is easily accessible from embedded HTML page and could be changed and styled using CSS styles and JavaScript properties. For simple charts advantages:

- Simple bar graphs are easy to create.
- Using combination of HTML and JavaScript creates powerful tool for creating basics bar graphs. Graph components are created by very short source code and effective way.
- Excellent Integration Integration to embedded HTML site
- Possibility of graphs styling using CSS styles

Disadvantage:

- This techniques are not capable of creating bigger and complex graphs or charts.
- Possibilities of drawing lines or curves are strongly restricted. Only way how to draw curves is to imitate lines using very small DIV elements, which is possible but extremely slow.
- Strong dependency on used browser or HTML interpreter.
- Browsers incompatibility, especially Internet Explorer with HTML standards.

4.4.2 SVG

SVG is a language for describing two-dimensional graphics in XML. SVG allows for three types of graphic objects: vector graphic shapes (e.g., paths consisting of straight lines and curves), images and text. Graphical objects can be grouped, styled, transformed and composited into previously rendered objects. The feature set includes nested transformations, clipping paths, alpha masks, filter effects and template objects. SVG drawings can be interactive and dynamic. Animations can be defined and triggered either declaratively (i.e., by embedding SVG animation elements in SVG content) or via scripting. Sophisticated applications of SVG are possible by use of a supplemental scripting language which accesses SVG Document Object Model (DOM), which provides complete access to all elements, attributes and properties. A rich set of event handlers such as onmouseover and onclick can be assigned to any SVG graphical object. Because of its compatibility and leveraging of other Web standards, features like scripting can be done on XHTML and SVG elements simultaneously within the same Web page. SVG is a language for rich graphical content. For accessibility reasons, if there is an original source document containing higher-level structure and semantics, it is recommended that the higher-level information be made available somehow, either by making the original source document
available, or making an alternative version available in an alternative format which conveys the higher-level information, or by using SVG’s facilities to include the higher-level information within the SVG content. For suggested techniques in achieving greater accessibility, see Accessibility.[7]

main shapes and functions: Paths Simple or compound shape outlines drawn with curved or straight lines can be filled in or outlined (or used as a clipping path) and are expressed in a highly compact coding in which, for example, M precedes the initial numeric X and Y coordinates and L will precede a subsequent point to which a line should be drawn. Basic Shapes Straight-line paths or paths made up of a series of connected straight-line segments (polylines), as well as closed polygons, circles and ellipses can be drawn. Rectangles and round-cornered “rectangles” are other standard elements. Text Unicode character text included in an SVG file is expressed as XML character data. Many visual effects are possible, and the SVG specification automatically handles bidirectional text (as when composing a combination of English and Arabic text, for example), vertical text (as Chinese was historically written) and characters along a curved path (such as the text around the edges of the Great Seal of the United States). Painting SVG shapes can be filled and/or outlined (painted with a color, a gradient or a pattern). Fills can be opaque or have various degrees of transparency. “Markers” are end-of-line features, such as arrowheads, or symbols which can appear at the vertices of a polygon. Color Colors can be applied to all visible SVG elements, either directly or via the ‘fill’, ‘stroke’ and other properties. Colors are specified in the same way as in CSS2, i.e. using names like black or blue, in hexadecimal such as #2f0 or #22ff00, in decimal like rgb(255,255,127) or as percen-
Gradients and Patterns

SVG shapes can be filled or outlined with solid colors as above, or with color gradients or with repeating patterns. Color gradients can be linear or radial (circular), and can involve any number of colors as well as repeats. Opacity gradients can also be specified. Patterns are based on predefined raster or vector graphic objects, which can be repeated in x and/or y directions. Gradients and patterns can be animated and scripted. Clipping, Masking and Compositing Graphic elements, including text, paths, basic shapes and combinations of these, can be used as outlines to define both ‘inside’ and ‘outside’ regions that can be painted (with colors, gradients and patterns) independently. Fully opaque clipping paths and semi-transparent masks are composited together to calculate the color and opacity of every pixel of the final image, using simple alpha blending.[8]

4.4.3 VML

The Vector Markup Language (VML) supports the markup of vector graphic information in the same way that HTML supports the markup of textual information. Within VML the content is composed of paths described using connected lines and curves. The markup gives semantic and presentation information for the paths. VML is written using the syntax of XML just as HTML is written using the syntax of SGML (the Standard Generalized Markup Language, [ISO 8879]) - XML is a restricted form of SGML. VML uses Cascading Style Sheets, Level 2 in the same way as HTML to determine the layout of the vector graphics which it contains. [5]

4.5 Browsers plug-in system

Browser plug-in is an add-on, that expand the capabilities or functionality of the web browser somehow. In the beginning of Energis data visualization extends browsers by any plug-in was clear way how to do it. Browser provides plug-in system as can be seen on figure 7 on page 26. Summary of possible graphical browser plug-able software follows:
• The Java Plug-in software is a component of the Java Runtime Environment (JRE). The JRE allows applets written in the Java programming language to run inside various browsers. The Java Plug-in is cross-platform available.

• SilverLight plug-in is based on Windows Presentation Foundation (WPF) and .NET Framework 3.0 technology.

• Adobe Flash plug-in is a multimedia platform for playing compiled Flash, actionScript applications and swf or flv format.

4.6 Selected Technologies

Technologies chosen for further examination, which fulfill basic criteria described in 3.5.1 on page 18 follows:

• AJAX technology + HTML5 canvas & exCanvas Library Future technology of the web. HTML5 offers enough features even for graphical rich systems.

• Java Applet Java platform offers features rich platform for Browsers.

• .NET Silverlight Modern technology by Microsoft.

• Adobe Flash Very popular graphical web technology.
5 HTML5 canvas element & exCanvas library

HTML5 is not in the time of writing this thesis standardized yet, but despite this fact all modern browsers implements canvas element or its parts. Microsoft trying to win the fight of future web pushing its Silverlight to the client side of web, but HTML5 canvas element among others, like video and audio, are clear sign of the future HTML and web technologies evolution. Genuine idea comes from Apple company who first implemented it in Safari web browser.

Using combination of this technologies for creating new set of IS Energis components has important advantage. This technology does not need any additional plug-ins or addons. Only condition for this environment is relatively modern browser.

5.1 Technology overview and main features

The canvas element provides scripts with a resolution-dependent bitmap canvas, which can be used for rendering graphs, game graphics, or other visual images on the fly. [3] Canvas is block HTML element which can be added to the DOM structure and become part of HTML page. Canvas Element provides basic drawing functions accessible through JavaScript Object Context. Canvas specification provides relatively simple interface defining browser implementation. It consist of canvas element width and height, toDataURL function which returns URL containing a representation of the image as a PNG or other file(depending on parameter). And function similar to other canvas technologies (e.g. Java) getContext(), which returns object that exposes an API for drawing on the canvas. GetContext(id) function has one parameter which identify type of drawing context. W3C defines only 2d context, but other specifications may define their own contexts, which would return different objects. Vendors may also define experimental contexts by using the syntax vendor-name-context, for example, moz-3d.

Canvas specification interface:

```javascript
interface HTMLCanvasElement : HTMLElement {
  attribute unsigned long width;
  attribute unsigned long height;

  DOMString toDataURL(in optional DOMString type, in any... args);

  object getContext(in DOMString contextId);
}
```

Canvas element has all needed capabilities for charts drawing, but what about other criteria for our new technology? Disadvantage is that user interaction is not supported for following reason. Canvas is accessible only through its drawing context. Points, lines nor shapes drawn on canvas have no references. Canvas only draw defined pixels. In case we need to change canvas content, we have to redraw specified points. On the other hand it makes Canvas technology very fast. Browser access directly to drawing routines by natively supported routines.
Because canvas, except Microsoft IE, need no plug-in components installation what is the difference between canvas and SVG/VML technology? There are two great differences between canvas and SVG/VML. The first great difference is in DOM representation. SVG/VML represents all its elements as a XML/DOM nodes, where Canvas is in DOM structure only one complex element. This implies great difference, in our case, which is handling charts parts. Where in SVG/VML every shape can be manipulated separately, providing event hook to connect handler script and all typical DOM functionality. Canvas is possible to change only by setting Canvas DOM node properties. This first great difference means, that handling user events is completely different. For example displaying tool tip information of each chart part in SVG/VML is done by simply connecting event handler on certain SVG/VML element. Handling events of Canvas is way more complicated. Because there is no mean to connect event handler to the part of canvas. Nevertheless solution could provide other block element, for example DIV, which covers all canvas size for events catching. Connected handler have to calculate position of event and execute desired changes, by redrawing canvas.

5.1.1 ExCanvas

Microsoft Internet Explorer does not support canvas technology. Maybe in hope they force users to install their proprietary solutions. However there are ways how to make HTML5 Canvas really cross-browser. Modern browsers like Firefox, Safari, Chrome and Opera support the HTML5 canvas tag natively to allow 2D command-based drawing. ExCanvas library brings the same functionality to Internet Explorer. ExplorerCanvas is library distributed under the terms of the Apache License V2.0. is Google library for transformation of the canvas commands to VML technology. If we need use canvas like real multi platform graphical feature we needs to include ExplorerCanvas library. Great advantage is that this approach needs no plug-in installation. The excanvas.js file must be included in the page before any occurrences of canvas elements in the markup. This is due to limitations in IE and we need to do our magic before IE sees any instance of canvas in the markup. It is recommended to put it in the head. following way:

```html
<head>
  <!--[if IE]><script src="excanvas.js"><script/><![endif]--/>
</head>
```

Canvas 2D context

The 2D context represents a flat Cartesian surface whose origin (0,0) is at the top left corner, with the coordinate space having x values increasing when going right, and y values increasing when going down. Complete interface is defined at W3C HTML Canvas 2D Context specification [4]

Most important functions for this work follows

- fillRect - Draws a filled rectangle
• StrokeRect(x, y, width, height) - Draws a rectangular outline
• clearRect(x, y, width, height) - Clears the specified area and makes it fully transparent
• beginPath() - starts the list of sub paths
• closePath() - This method tries to close the shape by drawing a straight line from the current point to the start.
• stroke() - draw a line to the canvas
• fill() - draw the shape to the canvas
• moveTo(x, y) - changes current position of drawing context
•.lineTo(x, y) - draws a line from current position to specified x, y point.
• arc(x, y, radius, startAngle, endAngle, anticlockwise) - draws arcs or circles
• quadraticCurveTo(cp1x, cp1y, x, y) - draws Quadratic Curve
• bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y) - draws Bezier curve

5.2 Example graph component
Sample component basically consist of canvas chart, data table, legend tab label and controls elements. GUI is create using HTML language and CSS styles. All elements of line chart, bar chart and axis is drawn on canvas element. Bars, lines and other graphical primitives are drawn loop through data samples using canvas fillRect, moveTo(), lineTo() and Path functions. Line styles are defined using stroke style, bar style define fill style.

5.2.1 Code snippets
In this section, lets see interesting parts of the chart life cycle.
First step is adding the canvas element to the body of the web page:

```html
<div id='jsGraph' class='JSgraph'>
  <canvas id='cGraph'/>
</div>
```

Then we call constructor of the canvasGraph object with setup object: defining id of the canvas DOM id, graph type (line/bar or both), width and height of the canvas element.

```javascript
cGraph = new canvasGraph({
  canvasId: 'cGraph', showBars: false, showLines: true,
  w: 500, h: 330,
});
```
data request:
this.getData = function(){
this.AJAX.onreadystatechange = this.getFnRef(this, 'showData');
this.AJAX.open('GET', this.dataSource + '?id=' + this.dataId, true);
this.AJAX.send(null);
}

and function show data and draw graph:
this.showData = function(){
  if (this.AJAX.readyState == 4) {
    if (this.AJAX.status == 200 || true) {

      this.xmlData = this.AJAX.responseXML;
      this.root = this.xmlData.getElementsByTagName('data');
      this.xmlData.getElementsByTagName('sample');

      this.processData(
        this.xmlData.getElementsByTagName('sample'));
      this.adjustCanvas();
      this.drawChart();

      this.bottomTableTab.appendChild(this.createBottomTable());
      this.createDataTable(this.values[0]);
    }
  }
}

Drawing lines and bars in loop over data samples
this.ctx.strokeStyle = colour;
this.ctx.lineWidth = 3;
this.ctx.lineTo(x1,y1,x2,y2);
this.ctx.stroke();

Also interesting part of graph component is handling user interaction. Event are catches by the invisible div covering all canvas visible area. When event is cached component calculate its position and displays tool-tip information at right position. Chart displays either statitical data or can be triggered to periodical data refreshing.
this.refresh = function(timeout){
  this.timer = setInterval(dojo.hitch(this, 'getData'), 5000);
}

Each time data are refreshed component calls adjustCanvas which resize drawable area and move scroll bar to the most right value.
Obrázek 8: Canvas sample component - electricity consumption (kWH) - line chart

this.adjustCanvas = function(){
    var width = ((this.samplesCount) * (this.step) + this.border);
    this.canvasElm.width = this.border + (this.samplesCount * this.step);
    dojo.style(this.eventCatcher, 'width', this.canvasElm.width + 'px');
    dojo.style(this.XAxis, 'width', this.canvasElm.width + 'px');
    dojo.byId('jsGraph').scrollLeft = this.canvasElm.scrollWidth;
}

5.3 Summary

Using native possibilities of modern web browser provides sufficient means to create selected sample component or event more complex web application. HTML and JavaScript environment is nowadays rapidly evolving and despite some serious issues this platform becomes self-sufficient. HTML5 is very young platform, but interest of big players like Google (gmail, Docs, maps), Facebook, Apple and also Microsoft proof, that evolution will not stops here. Serious problem is different implementation of W3C standards. However big number specialized frameworks and and toolkits help to cover differences of browsers.
Developer support provides Eclipse based integrated development environment for HTML5 and JavaScript called Aptana. Excellent debugging tool is Firebug browser plug-in or Chrome build in debugger.

This sample component use Dojo Toolkit and ExCanvas library to provide seamless cross-browser application and provide proof of concept for this technology. Components are displayed at figure 9 and 8 on pages 33, 34.

**Chart features** Sample component works in two basic modes.

- First displays real-time data. It means that new values are added to the older data samples and chart area grows with time.
- Second mode enable samples refreshing. New coming samples are overwritten and whole component is refreshed.
- Line chart could be instantly switched to bar graph and back any time or both of them could be visible at once.
- Detailed data are displayed in data table.
- Move over the chart objects display information label.

Improvement for our sample component could bring interesting possibilities of JavaScript charting library called FLOT. It is JavaScript library providing easy and user
friendly graphs and chart. FLOT uses same approach to creating complex and scalable chart using HTML5 canvas technology combined with exCanvas library.
6 FLASH

Adobe Flash (formerly Macromedia Flash) is a multimedia platform used to add animation, video, and interactivity to web pages. Flash is frequently used for advertisements and games. Flash manipulates vector and raster graphics to provide animation of text, drawings, and still images. It supports bidirectional streaming of audio and video, and it can capture user input via mouse, keyboard, microphone, and camera. Flash contains an Object-oriented language called ActionScript. Flash content may be displayed on various computer systems and devices, using Adobe Flash Player, which is available free for common web browsers, some mobile phones and a few other electronic devices (using Flash Lite).[12]

6.1 Technology overview and main features

Flash players uses the SWF file format. SWF delivers vector graphics, text, video, and sound over the Internet. The files are compressed to be small and support incremental rendering through streaming. The SWF file format is a binary format and is not human readable like HTML. The SWF file format uses techniques such as bit-packing and structures with optional fields to minimize file size.

format includes tags that provide sequences of byte codes to be interpreted by a stack machine. The byte codes support the ActionScript language. Flash Player provides a runtime ActionScript object model that allows interaction with drawing primitives, servers, and features of Flash Player. SWF files have the extension .swf and a MIME type of application/x-shockwave-flash.

Flash platform offers two run-time environment AIR and Flash Player.

Flash was developed as tool for graphic designers which is extended by ActionScript to provide limited possibilities of programming. On the other hand other way how to create SWF file is Flex SDK. It consist of Compiler debugger and Flax Framework. Flex is more oriented towards developers for creating business-type applications. Flex provides an application “framework” which provides the components we need to create complex applications. Flax framework consist of Flax libraries and

MXML and ActionScript. MXML is an XML markup language that you use to lay out user interface components. You also use MXML to declaratively define nonvisual aspects of an application, such as access to data sources on the server and data bindings between user interface components and data sources on the server. Like HTML, MXML provides tags that define user interfaces. MXML will seem very familiar if you have worked with HTML. However, MXML is more structured than HTML, and it provides a much richer tag set. For example, MXML includes tags for visual components such as data grids, trees, tab navigators, accordions, and menus, as well as nonvisual components that provide web service connections, data binding, and animation effects. You can also extend MXML with custom components that you reference as MXML tags. One of the biggest differences between MXML and HTML is that MXML-defined applications are compiled into SWF files and rendered by Adobe Flash Player or Adobe AIR, which provides a richer and more dynamic user interface than page-based HTML applications. You can
write an MXML application in a single file or in multiple files. MXML also supports custom components written in MXML and ActionScript files.[9]

6.1.1 ActionScript

Adobe ActionScript is the programming language of the Adobe Flash Platform. Originally developed as a way for developers to program interactivity, ActionScript enables efficient programming of Adobe Flash Platform applications for everything from simple animations to complex, data-rich, interactive application interfaces. [8]

Initially focused on animation, early versions of Flash content offered few interactivity features and thus had very limited scripting capability.

6.1.2 Flash Builder 4

Adobe creates very powerful Integrated Development environment for Flash and Flax called Flash Builder 4. Flash Builder provide feature rich environment for Flash / Flax developers.

Embedding Flash into HTML pages  Flash components could be embedded in HTML pages two ways. First and more supported choice is embed tag, which uses attributes width and height, that specifies the width or height of the Flash component in either pixels or percentage of browser window. Necessary is also src attribute, that specifies the location (URL) of flash source to be loaded. Pluginspage attribute, pointing to flash plug-in download web site is useful for users, who does not have flash plug-in installed.

Object is second - W3C standardized possibility. Object accepts attributes width and height, that specifies the width or height of the embedding component in either pixels or percentage of browser window. Child nodes of object tag are param elements containing key-value pairs. Most important params are movie, which specifies the location (URL) of the loaded movie, classid - ActiveX control identifier, codebase - location of the Flash Player ActiveX control and also id and name parameters.

6.2 Example graph component

Flash Sample component consist of main Application element containing Tab Navigator element with Panel elements. Panel elements contains Graph component and data table.

6.2.1 Code snippets

As a first step we have to embedded flash object into the web page. Because of the differences of implementation of W3C standards it is recommended to use both possibilities of component embedding. Minimal construction of embedding element follows:

```xml
<object id='graph' width='550' height='450' >
  <param name='movie' value='http:// ... ' />
  <param name='dataSource' value='sajtne.com/data.php'/>
</object>
```
Data are requested using HTTP service which defines functions for data and error handling.

```xml
<mx:HTTPService id="xmlRPC" result="handleXML(event);">
  fault="handleFault(event);" resultFormat="object"
  url="" showBusyCursor="true" useProxy="false" />
```

Useful feature is user friendly data handling. XML data are easily parsed to ArrayCollection which is direct source for chart, data table and other possible components.

```java
public function handleXML(event:ResultEvent):void
{
  result = event.result.data.sample as ArrayCollection;
}
```

mx:Model tag produces an Object, that deserializes the XML into an object. The benefit of this element is that you can reference members of XML with dot notation and XML nodes can be navigates to the array of customer objects representing XML Nodes.

```xml
<mx:Model id="samples" source="dataF.xml"/>
<mx:ArrayCollection id="dataset"
  source="{
    ArrayUtil.toArray(samples.sample)}"
/>
```

Create line chart with specified parameters and data source.

```xml
<mx:LineChart chromeColor="119900" width="100%" height="100%"
  id="myChart" dataProvider="{dataset}" showDataTips="true">
  <mx:backgroundElements>
    <mx:GridLines>
      <mx:verticalFill>
        <mx:SolidColor color="haloBlue" alpha="0.2"/>
      </mx:verticalFill>
      <mx:verticalAlternateFill>
        <mx:SolidColor color="haloSilver" alpha="0.2"/>
      </mx:verticalAlternateFill>
    </mx:GridLines>
  </mx:backgroundElements>
```
Create column chart with specified parameters and data source.

```xml
<mx:ColumnChart id="myChart" dataProvider="{dataset}" showDataTips="true">
    <mx:backgroundElements>
        <mx:GridLines>
            <mx:verticalFill>
                <mx:SolidColor color="haloBlue" alpha="0.2" />
            </mx:verticalFill>
            <mx:verticalAlternateFill>
                <mx:SolidColor color="haloSilver" alpha="0.2" />
            </mx:verticalAlternateFill>
        </mx:GridLines>
    </mx:backgroundElements>
    <mx:horizontalAxis>
        <mx:CategoryAxis categoryField="order" />
    </mx:horizontalAxis>
    <mx:series>
        <mx:ColumnSeries yField="value" displayName="Data set NO. 1" />
    </mx:series>
</mx:ColumnChart>
```

Create data table with specified data source.

```xml
<mx:DataGrid id="tabula" x="0" y="0" width="548" height="100%"
    dataProvider="{dataset}" >
    <mx:columns>
        <mx:DataGridColumn headerText="Hodnoty" dataField="value"/>
        <mx:DataGridColumn headerText="Typ dat" dataField="color"/>
        <mx:DataGridColumn headerText="Detail info" dataField="info"/>
    </mx:columns>
</mx:DataGrid>
```
Obrázek 10: Flash sample component - electricity consumption (kWH) - line chart

6.3 Summary

Flash provides all features needed to implement selected graph component. Native Flash libraries and class are designed for easy graphical components development which makes it very easy to create even more complicated components. Flash Builder 4 based on Eclipse is powerful tool for building and programming flash component. It provides user friendly tools and functions, which makes development simple and fast.

Chart features

- Graph displays data sets requested from web server as either line chart or bar graph
- Graph and chart could be switched by user interaction.
- Data table displays detail information in table form.
- Move over the chart objects interactivity displays values.
Obrázek 11: Flash sample component - electricity consumption (kWH) - bar graph
7 Java Applets

Java applet technology is well established platform in the IT world. It has long development history. From its beginning, when Java was developed it was revolutionary and ambitious technology. Java is a programming language originally developed by Sun Microsystems and released in 1995 as a core component of Sun Microsystems’ Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. This fact makes it easy to integrate to any web browser, that runs Operational system capable of running Java runtime environment. During Java development Sun made available most of their Java technologies as free software under the GNU - General Public License. On April 20, 2009 Sun Microsystems was bought by Oracle.

Java applet technology was originally used for data visualization in IS Energis, but as Java evolve this solution become obsolete, as described in section 2 on page 7

7.1 Platform overview and main features

One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any supported hardware/operating-system platform. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to platform-specific machine code. Java bytecode instructions are analogous to machine code, but are intended to be interpreted by a virtual machine (VM) written specifically for the host hardware. End-users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a Web browser for Java applets. [10]

7.1.1 Java applet

A Java applet is an applet delivered to the users in the form of Java bytecode. Java applets can run in a Web browser using a Java Virtual Machine (JVM), or in Sun’s AppletViewer, a stand-alone tool for testing applets. Java applets were introduced in the first version of the Java language in 1995. Java applets are usually written in the Java programming language.

Java applets are executed in a sandbox by most web browsers, preventing them from accessing local data like clipboard or file system. The code of the applet is downloaded from a web server and the browser either embeds the applet into a web page or opens a new window showing the applet’s user interface. A Java applet extends the class java.applet.Applet, or in the case of a Swing applet, javax.swing.JApplet.

The class must override methods from the applet class to set up a user interface inside itself (Applet is a descendant of Panel which is a descendant of Container. As applet inherits from container, it has largely the same user interface possibilities as an ordinary Java application, including regions with user specific visualization. The domain from where
the applet executable has been downloaded is the only domain to which the usual (un-
signed) applet is allowed to communicate. This domain can be different from the domain
where the surrounding HTML document is hosted.
Java system libraries and runtimes are backwards compatible, allowing to write code
that runs both on current and on future versions of the Java virtual machine. [11]

**Embedding Java Applet into web page** Java applets are usually included into the
HTML pages using, nowadays obsolete, applet tag element or W3C standardized object
element which is not implemented by most of browsers. Applet element consist of co-
debase and code attributes specifying path to the main class and attributes for applet
dimension setup width and height attributes. Applet element contains PARAM elements
child nodes, that contains key-value pairs NAME and VALUE.

### 7.2 Example graph component

Sample Java applet component is inherited from JApplet and at the top level consist
of tabbedPane, control buttons and data table displaying selected data details. Tabbed-
pane switches Graph tab and data table tab. Graph tab contains drawable area - object
inherited from JPanel overriding method paintComponent, which paint selected graph
or chart. Object inherited from JPanel override also mouse event handlers, which crea-
tes user interactive tool tip. Applet requests and handles data from HTTP server using

#### 7.2.1 Code snippets

before applet initialization, it parse applet parameters

```java
try {
    this.dataSource = getParameter("dataSource");
    System.out.println("dataSource: " + this.dataSource);
} catch (Exception e) {
    this.dataSource =
        "http://sajtna.freehostia.com/thesis/data.php";
    System.out.println("Default dataSource: " +
        this.dataSource);
}
```

On initComponent start data request and handling XML document. First we create
DocumentBuilderFactory and request data from data source

```java
DocumentBuilderFactory dbf =
    DocumentBuilderFactory.newInstance();
DocumentBuilder db = null;
try {
    db = dbf.newDocumentBuilder();
```
Then we start to parse XML Response

doc.getDocumentElement().normalize();

NodeList nodeLst = doc.getElementsByTagName("sample");
this.data = new Sample[nodeLst.getLength()];

for (int s = 0; s < nodeLst.getLength(); s++) {
    Node fstNode = nodeLst.item(s);
    if (fstNode.getNodeType() == Node.ELEMENT_NODE) {
        Element fstElmnt = (Element) fstNode;
        Element fstNmElmnt = (Element) (fstElmnt.getElementsByTagName("value")).item(0);
        int val = Integer.parseInt(((Node) (fstNmElmnt.getChildNodes()).item(0)).getNodeValue());
        ...
    }
}

Getting drawing context of JPanel, where graphs are displayed

super.paintComponent(g);
    if (this.samples == null || this.samples.length == 0) {
        return;
g2 = (Graphics2D) g;
g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
    RenderingHints.VALUE_ANTIALIAS_ON);
g2.setFont(font);
frc = g2.getFontRenderContext();

Let's draw all bars using this code
for (int j = 0; j < this.count; j++) {

g.setColor(this.samples[j].color);
g.fillRect(x2Points[j], (this.axisHeight-this.samples[j].percetVal), stepX, this.samples[j].percetVal);
g.setColor(Color.black);
g.drawRect(x2Points[j], (this.axisHeight - this.samples[j].percetVal), stepX, this.samples[j].percetVal);
}

And draw also line and points
for (int j = 0; j < this.count; j++) {
    polyline.lineTo(x2Points[this.samples[j].order] +
                (stepX / 2), (this.axisHeight - this.samples[j].percetVal));
}
g.drawLine(polyline);

for (int j = 0; j < this.count; j++) { // kreslim kolecku
    g.setColor(this.samples[j].color);
g.fillRect(x2Points[this.samples[j].order] + (stepX / 2) -
5, this.axisHeight - this.samples[j].percetVal -
5, 10, 10, 0, 360);
}

And handle user interaction by overriding mouseMoved event handler
public void mouseMoved(MouseEvent e) {
    int x = e.getX();
    int y = e.getY();

    this.Tooltip = new JLabel();
    this.Tooltip.setLocation(x,y);
}
this.Tooltip.setText(this.samples[xPosition].info);
this.paint(g2);
this.repaint();
}

Parsed data are stored in helper objects of Sample class

class Sample {
    float value;
    Color color;
    String info;
    int order;
    int id;
    float percentVal;
}

7.3 summary

Chart features Java applet is fully capable of all needed features. Current visualization components of IS Energis are based on AWT Graphical Package. Reworked Java applet component uses modern SWING graphical components package, which makes programming faster and more comfortable. Useful tool for Java applet component is NetBeans IDE, which provides useful tool for graphical user interface creation and other helper functions.

Chart features

- Displays data in from web server either as line chart or bar graph
- Type of graph could be switched
- Detailed data are displayed in data table
- Move over the chart objects display information label.

Components are displayed at figure 12 and 13 on pages 49, 48.
Obrázek 12: Java applet sample component - electricity consumption (kWH) - bar chart
Obrázek 13: Java applet sample component - electricity consumption (kWH) - line chart
8 Silver Light .NET

SilverLight is a subset of WPF - modern presentation technology based on Microsoft .NET platform.

8.1 Platform overview and main features

8.1.1 .NET

The .NET Framework is an integral Windows component that supports building and running the applications and XML Web services. The .NET Framework is designed to fulfill the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.

- To provide a code-execution environment that minimizes software deployment and versioning conflicts.

- To provide a code-execution environment that promotes safe execution of code, including code created by an unknown or semi-trusted third party.

- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.

- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.

- To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services. The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and
unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts. For example, ASP.NET hosts the runtime to provide a scalable, server-side environment for managed code. ASP.NET works directly with the runtime to enable ASP.NET applications and XML Web services, both of which are discussed later in this topic. Internet Explorer is an example of an unmanaged application that hosts the runtime (in the form of a MIME type extension). Using Internet Explorer to host the runtime enables you to embed managed components or Windows Forms controls in HTML documents. Hosting the runtime in this way makes managed mobile code (similar to Microsoft® ActiveX® controls) possible, but with significant improvements that only managed code can offer, such as semi-trusted execution and isolated file storage. [6]

8.1.2 WPF

The core of WPF is a resolution-independent and vector-based rendering engine that is built to take advantage of modern graphics hardware. WPF extends the core with a comprehensive set of application-development features that include Extensible Application Markup Language (XAML), controls, data binding, layout, 2-D and 3-D graphics, animation, styles, templates, documents, media, text, and typography. WPF is included in the Microsoft .NET Framework, so you can build applications that incorporate other elements of the .NET Framework class library. [7]

8.2 Example graph component

Sample Silvrlight component is create of SilverLight Application Object, which contain User control object, that provides all desired functionality.

8.2.1 Code snippets

Basic layout provide Grid element containing other parts of sample component. Main container contains Labels canvas DataGrid and buttons elements

```xml
<Grid x:Name="LayoutRoot" Background="White" ShowGridLines="True">
    <dataInput:Label x:Name="Info_Label" Grid.Column="0"
        Content="Nadpis grafu" Width="100" ToolTipService.ToolTip="Nadpis grafu" VerticalAlignment="Top" Height="20"/>
    <Canvas x:Name="Canvas_Canvas" Grid.Row="0" Grid.Column="0"
        Background="BurlyWood" Margin="0,20,0,0"/>
    <data:DataGrid x:Name="GrafData_DataGrid" Grid.Row="1" Grid.Column="0"
        Background="Azure" AutoGenerateColumns="True" Margin="0" />
    <dataInput:Label x:Name="GraphTitle_Label"/>
    <Button HorizontalAlignment="Right"/>
```
Silver Light component consist of class inherited from Application, which uses User Control Graph_UC as a RootVisual

```csharp
public partial class App : Application{
    private void Application_Startup(object sender, StartupEventArgs e){
        String DataSource = "";
        try{
            DataSource = e.InitParams["DataSource"];}
        catch (Exception ex) { DataSource = "No datasource!!!";}
        MainPage mp = new MainPage(GrafType, DataSource);
        this.RootVisual = mp;
    }
}
```

Data transport and handling

```csharp
void ParseResponse(object sender,
                    DownloadStringCompletedEventArgs e){
    if (e.Error == null){
        XDocument DataDoc = XDocument.Parse(e.Result);
        LoadAndParseXML(DataDoc);
    }
}
```

```csharp
private void LoadAndParseXML(XDocument DataDoc){
    this.GraphTitle_Label.Content = "Graph description";
    this.dataXMLInfo = (from data in DataDoc.Descendants("data")
                        select new DataInfo{
                            Min = data.Attribute("min").Value,
                            Max = data.Attribute("max").Value,
                            Count = data.Attribute("count").Value
                        }).ToList();
    this.min = Int32.Parse(dataXMLInfo.ElementAt(0).Min);
    this.max = Int32.Parse(dataXMLInfo.ElementAt(0).Max);
    this.count = Int32.Parse(dataXMLInfo.ElementAt(0).Count);
    this.dataXML = (from data in DataDoc.Descendants("sample")
                    select new Sample{
                        Value = data.Element("value").Value,
                    });
```
Draw one plot line of graph component.

```csharp
this.ChartPoints = new PointCollection();

for (int i = 0; i < this.count; i++){
    if (LineDataSet.ElementAt(i) != null){
        double x = i * (int)(this.stepX) + this.zero.X;
        double y = (this.height) - ((((Double.Parse(LineDataSet.ElementAt(i).Value.ToString())) / this.max) * (this.axisH)) - this.zero.Y);
        ChartPoints.Add(new System.Windows.Point(x + (this.stepX / 2), y));

        Ellipse joint = new Ellipse();
        joint.Width = 14;
        joint.Height = 14;
        joint.Fill = new SolidColorBrush(GetColorFromHex(point));
        joint.StrokeThickness = 2;
        joint.Stroke = new SolidColorBrush(GetColorFromHex(circle));
        Canvas.SetLeft(joint, x + (this.stepX / 2) - 7);
        Canvas.SetTop(joint, y - 7);
        this.Canvas_Canvas.Children.Add(joint);
    }
}
chart.Points = this.ChartPoints;
this.Canvas_Canvas.Children.Add(chart);
```

8.3 summary

SilverLigth and .Net platform is capable of any graphical data visualization. It is very powerful tool for creating user friendly components and Microsoft Corporation keep
Obrázek 14: Silverlight sample component - electricity consumption (kWH) - line chart

evolving this platform. SilverLight also provides Chart and plots basic elements as a part of platform package, which can make development better. SilverLight sample component uses common UI elements and transformations from the framework and uses built-in functions to load and process an XML Document. Microsoft Visual Studio offers great platform for SilverLight application development, GUI creating and debugging.

Chart features

- Displays data from web server either as line chart or bar graph
- Type of graph can be switched
- Detailed data are displayed in a data table
- Move over the chart objects display information label.

Components are displayed at figure 14 and 14 on pages 55, 55.
Obrázek 15: Silverlight sample component - electricity consumption (kWH) - bar graph
9 Online demo Site

9.1 Technology

As part of this thesis is also included demo testing site, displaying sample components and other data visualisation possibilities. Demo site provides overview of each used technology in separated pages. Demo site has home page which navigate user to sub section dedicated to certain technology or its solution. Most important part of the web site is data source. Data source was created by specification described in the beginig of this work at section 3 on page 11.

Demo site is designed in PHP technology with aid of Dojo Toolkit and CSS. Web is hosted on realy slow free hosting service sajtna.freehostia.com/thesis.

9.2 Content

Demo site contains information of selected technologies and platforms as well as sample components ad useful links. Each technology has its own dedicated sub section with saple component and charts created by differnet charting technology. Main goal of this website is to ease comparation and provide complex overview. Basically demo site consist of two main part. On the left side is navigation menu with following sections:

- Javascript
- SVG/VML
- Flash
- HTML5 canvas
- SilverLight

Each section contains three subsections Information - short decriptions and overview, Chart - contains chart examples and subsection Links provideing list of interesting charting sites.

9.3 Data source for saple components

In general, first step of data visualisation is to get specified data for visualisation purposes. Speaking about online visualization, we assume some kind of web server or web service as a data source. This implies use of HTTP/HTTPS communication protocol for data transportation. Using standardized communicating protocol brings many advantages. Beside encapsulation functionality of transportation issues developed for long time by leading IT companies, it provides well designed interface which is implemented by all web browsers, others utilities and also by software of modern displaying devices. There are some data formats suitable for client server architecture such e.g. JSON format, which saves data in form of JavaScript Object notation or YAML human-readable data serialization format, but best practice of online data exchange is using HTTP/HTTPS protocol.
combined with XML data format. Concerning compatibility it is the best way how to share various data among many clients. XML and XML based languages provides flawless standards and tools for clear and easy to use data handling. XML also provides possibility of defining strict data structure. XML structure can be described using DTD definition or its successor XSL.

Each chart could visualise data or set of data from independent source, thus receiving have to know transported data structure. Data description is crucial for checking integrity of fetched data and also helpful for data parsing. Data description is also good reference for possible developers or other interested persons, which needs to know data structure, their types and restrictions. Easy data validation against DTD or XSD scheme could prevent possible data corruption. More at subsection 3.3 on page 14. Data can be requested from testing web server at the http://sajtna.freehostia.com/thesis/data.php.

PHP script generates random data following way:

```php
$dom = new DOMDocument("1.0");

// create root element
$root = $dom->createElement("data");
$dom->appendChild($root);

$samplesCount = $dom->createAttribute("count");
$root->appendChild($samplesCount);
$samplesCount->appendChild($dom->createTextNode($count));
...

for ($i = 0; $i < $count; $i++) {

    $sample = $dom->createElement("sample");
    $root->appendChild($sample);
    $id = $dom->createAttribute("id");
    $id->appendChild($dom->createTextNode($i));

    $sample->appendChild($id);
    $id->appendChild($dom->createTextNode($i));

    $value = $dom->createElement("value");
    $sample->appendChild($value);
    $vText = $dom->createTextNode(rand($min,$max));
    $value->appendChild($vText);
    ...
}

// save and display tree
echo $dom->saveXML();
```
10 Results

10.1 Tested features

10.1.0.1 Performance  Evaluate speed of code execution. Based on different test of some basics mathematical operations, String handling, sorting and other computing operations. Performance score influence also time needed to start up the component inside the web browser and length of initial phase.

10.1.0.2 Platform Independency  Rate how much is the selected technology dependent on certain architecture, operational system or hardware.

10.1.0.3 Support / community  Rate how big company or community is behind the project, who finance and develop it.

10.1.0.4 API / developers friendly  Maturity of the code, API and tools for programmer.

10.1.0.5 Documentation  Rate the documentation of each technology, compare its quality and quantity.

10.1.0.6 Licensing  Show licensing of language it self and it’s IDEs or other tools for developers.

10.2 JavaScript with HTML5 Canvas or (SVG/VML)

Seem to be excellent choice. Simple Javascript it self is not powerful enough to provide sufficient possibilities especially to draw graphical parts of graphs, charts and e.g. Simple Javascript is able to create simple bar graph using coloured html tags. However JavaScript, when combined with vector graphics format of any kind or HTML5 canvas element, rapidly raised its potential of chart drawing or other even more complex graphical objects. Excellent example is combination of HTML5 or SVG format, Javascript. Javascript supply graph with data and implements its logic and basic layout in the HTML document. SVG or Canvas provide drawing complex graphs or curves. HTML and Javascript evolve rapidly based of interest of the strogest companies in IT world, its openes and true platform independecy makes this fascinating technology very potent and flexible.

10.2.0.7 Advantages  Javascript is implemented in all the modern web browsers a thus using JavaScript graphs eliminates need for plug ins of any kind. For solution, where downloading any new plug in is difficult, for example users with restricted Internet access or other security reason is this solution very useful. SVG as well as Javascript - EcmaScript are standardized and open web technology, which prove its self( for example:
gmail or facebook Js written GUI and AJAX based architecture) as a reliable language for web development. As Javascript become popular and standardized many new frameworks was created. For writing set of various graphical components there is wide selection of well supported JavaScript projects. Many of them use the SVG graphical format or Canvas elements or its suplements for drawing complex graphical shapes. Some of the frameworks has also international leading companies on their side e.g. Google’s GWT or Yahoo’s UI. Microsoft in collaboration with Nokia lately adopt jQuery to their ASP.NET AJAX Control Tool kit, as well as to implement new Ajax server-side helper methods for ASP.NET MVC.

10.2.0.8 Disadvantages comparing to Java or .NET Silverlight, JavaScript is still only scripting interpretation language which means that Javascript is remarkably slower, than its compiled competitors. However vibrant development of Javascript bring many new scripting languages optimization, the executing speed cannot be faster or even the same as compiled language’s. On the other side, for graphical user interface or some basic mathematical calculation speed is not so important the issue. As long as Javascript is language for web design, AJAX, DOM manipulation and layout tool, futures JS performance is not excellent, but sufficient. Other disadvantage is different JS implementation. Each browser (JavaScript engine) has its own JavaScript behaviour or even worst - a bit different syntax. This problem could be solved by using JavaScript framework, encapsulating different Javascript interpretation into standardized unite API. Despite of the fact that SVG is standard for long time (since 2003) Microsoft reject to implement this standard even in their Internet Explorer 8. Latest preview of IE9 says SVG will be supported. Microsoft developed its own web graphics called VML, but again, frameworks works as interfaces between Javascript code and VML/SVG commands.

10.3 Flash

Flash is fully able to implement all variety of graphical output as charts, plots or complex graphs. It support well designed features and protocols for communicating with server as well as GUI and graphical features providing drawing of different 2D and 3D objects.

10.3.0.9 Advantages: Flash is developed for long time and became very popular among both webdesigners and users too. It has gone long way from its beginning and its maturity level is high. Because of its popularity, Flash has wide base of skilled programmer and also sophisticated tools and purpose based IDE of different kind. Its ActionScript is more advanced than Javascript. Adobe took advantage of fight between big companies about Javascript standard and implement revolutionary scripting language instead of evolutionary Javascript. Because of its popularity, Flash player (Flash runtime environment browser plug in) is well penetrated to the browser of the Internet population. Its penetration is about 90%. Strong part of Flash is also quick loading of flash programs, vector graphics. Flash is backward, forward and platform compatible. Flash ability of chart drawing is huge. Native support of charts components e.g. line, bars and axis are included.
in Flex API. This makes Flax and ActionScript technology most advanced API of selected technologies.

10.3.0.10 Disadvantages: Flash needs its runtime environment, which in web browsers is provided by the flash player plug in. Plug in have to be downloaded from the Internet or distributed other way. This disadvantage is basically about need of downloading and updating flash player. Because Flash platform is proprietary it has way smaller community. Flash player itself is also distribute for open platforms, but its implementation e.g. for Linux is lame and its development is is delayed. Worst thing is that original Adobe's IDE is proprietary and windows only. Also for other reasons Flash is strongly platform dependent. Flash is also offently blamed to cause stability issues and browser crashes as well as high operational memory and processor demands.

10.4 Java

Nowadays Java more than programming language is platform supporting almost all capable computers and devices. Java as a programming language is fully capable of creating the most complex graphical output, calculate the difficult tasks or use well designed packages for communication with server side. Java is can solve difficult problems using big numbers open project or existing web services.

10.4.0.11 Advantages: Java is a very powerful platform, supported by giant companies with formerly SUN and Oracle in first place as well as open-source community is perfect choice for many kinds of use. Web browsers allows Java applet integration seamlessly. Java runtime environment is implemented for most thinkable platforms. Java is on very high maturity level. It have been intensively developed and optimized over fifteen year. Java also provides many commercial and opens solutions of charting a provides vide base of skilled programmers

10.4.0.12 Disadvantages: Despite all the maturity of the Java platform there are still some flaws. As as a little disadvantage we can see difficulties in communicating between applet element hosted in HTML document and the DOM model of the HTML document. The thing is, that it is difficult to call for example javascript function from Java applet. The weight of this disadvantage depends on the type of the project. However, this disadvantage could be removed by using existing package, supporting javascript call. Other disadvantage could be the size of the Java applet or java runtime environment. For smaller projects it could be heavy tool and that is why the loading time of applet could takes more time, than its competitors.

10.5 Silverlight

Microsoft Silverlight is a programmable web browser plugin that enables features such as animation, vector graphics and audio-video playback that characterise rich Internet
applications. Latest version, brings additional interactivity features and support for .NET languages and development tools.

10.5.0.13 Advantages: Silverlight is also fully capable of building complex and functional solution. Silverlight Visual Studio is really powerful tool for developing all kind of solutions. When server side and client side both runs .NET platform it allows fast and effective development. Silverlight is based on .NET platform, which is developed and supported by Microsoft. This brings all known advantages - great documentation, hordes of skilled programmers, professional development tool and easy accessible support. Some demo application show that web application written in SilverLight are real eye candy experience.

10.5.0.14 Disadvantages: The real flaw is the platform dependency. Silverlight runtime environment is officially available only for Windows Vista, 7, XP-SP2 and Mac OS X only. This flaw reduces the usability to only three, but the most wide spread operational system. Linux project Moonlight is in progress, but it is way form being satisfying late Silverlight environment. Silverlight is focuset only on two Microsoft OS and one Mac OS, which is while compared with Java small number of all possible platforms. After years of development SilverLight has not been able to conquer much share of browser plugin based market.

10.6 Final result

Choose system architecture, platform and language is crucial task in each project analysis and design. Every case is different in many ways and should be reasonably examined and discussed. Also in our case, choosing visualizing technology, each one of them has its advantages and disadvantages. After all facts that have been mentioned, it is not possible to simply decide which one of them is the best in general. Each technology have to be reconsidered in context of its client and server side, users, developers, and also network infrastructure. Important are also other points of view like numbers of users, who will request service, size of the visualized data and databases, a computer system infrastructure where system is deployed, security, and last but not least users needs.

Considering performance SilverLight and Java Applet followed by Flash technology provides sufficient power to solving non trivial tasks and problems. JavaScript as a scripting language, despite of its lately radical performance improvements and computer improvements in general, is not the right choice for computational very difficult tasks. Performance could be influenced also by numbers of visual elements included in web page. Start up of client side component, based on plug-in technology, could be negatively influenced by its initial sequence and in this case start up JavaScript based component are generally faster.

Platform dependency does not matter much when all the client computers are chosen and installed by the system requirements. But this question important when our system should be accesible from as much as possible computer devices. When new system
should be deployed in existing, for example large and complex computer infrastructure, it could be difficult to ask client to change its corporate policy or rules. In case that we need to visualize data in environment with small possibilities of change current infrastructure or install any new software, JavaScript combined with Canvas and SVG/VML graphics is excellent choice. Java is also platform independent, but requires at least JRE browser plug-in. Flash and SilverLight runs on limited numbers of platform. As well as Flash, Silverlight is possible of very nice graphical effects with minimal effort. For typical Internet users SilverLight is unknown technology. Its penetration is very low. Optimistic research said it is about 30% of all the browsers users. Flash penetration is about 95% and is designed for fancy graphical user Interfaces and stunning graphical effects.

Technology support is important part of decision. Technology have to be updated and maintained. Technology must have company or community capable of further evolution, documenting and supporting selected technology. Each selected visualization technology has its home company strong enough to keep technology going for long time. Java, Flash and SilverLight is supported by strong IT companies and JavaScript and HTML5 canvas should be implemented by standards issued by W3C Consorciun which committees consist of strong IT companies.

Documentation and API is matter of developers. Traditionally, Microsoft corporation has traditional excellent documentation and support. Java and Flash provides also precise and user friendly documentation and tutorials. Javascript as becoming more and more popular provides for its developers all needed support.

<table>
<thead>
<tr>
<th>Technology</th>
<th>JavaScript</th>
<th>FLASH</th>
<th>Java</th>
<th>Silverlight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Platform Independent</td>
<td>++</td>
<td>-</td>
<td>++</td>
<td>-</td>
</tr>
<tr>
<td>Support / community</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>API / developers friendly</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Documentation</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>++</td>
</tr>
<tr>
<td>Licensing</td>
<td>Open</td>
<td>Proprietary EULA</td>
<td>Open</td>
<td>MS-EULA, with MS-PL components</td>
</tr>
</tbody>
</table>

Tabulka 1: Table of results
11 Conclusion

Thesis describes background environment of IS Energis, part of its history and possibilities of on line data visualization. In more details I examines temporary visualization components, that are pluggable to custom users web pages and summarize technologies, capable of the same or even better visualization features. Java applet, Flash, Microsoft Silverlight and HTML5 canvas are analyzed an described. Thesis summarize possibilities and features of each mentioned platforms or technology and focused on defined criteria I reconsider its advantages and disadvantages. Using chosen technologies i implemented proof of concept of each one of them. In each technology, sample visualizing component consist of graphical user interface, line plot, bar chart, data tables elements and system of handling user interaction as well as data handling an parsing XML documents function. I also created new XML data scheme and describe possible XML format using XDL language.

Part of this thesis is also testing demo site, which could be used as source of graphical possibilities demonstration as well as technology study support materials. Demo site is created in PHP language and Dojo Toolkit JavaScript framework. Demo site also serves as source of XML data for sample component.

I would like to remark that I wrote this thesis in English language, which is not my native language and this work was also used as support decision data source in real-life, commercial company.

Aleš Fuks
12 Reference


