UDC 7.05:007
ORIGINAL SCIENTIFIC PAPER
RECEIVED: 28-11-2011
ACCEPTED: 12-04-2012

ACTA GRAPHICA 204

The Alternative Way of Creating Infographics Using SVG Technology

Authors

Sandra Pavazza[1]*, Klaudio Pap[2]

[1]*Zagreb, Croatia*
**E-mail: s.pavazza@gmail.com*

[2]*Faculty of Graphic Arts, University of Zagreb, Croatia*

Abstract:

The article develops new ways of creating and using interactive SVG infographics. The emphasis lies on the compatibility of SVG standard with other web standards, like XML, XSL, CSS, SMIL and ECMAScript, the advantages that it brings are particularly explored. There is a XSLT template developed which transforms XML data into SVG infographic, and the way of achieving complete control over data and data visualization is tested. This enabled the achieving of dynamic control of content and its presentation, and contributed to the results in reduced developing cost and time, better flexibility and reliability of the organizational system. The paper also studied the possibility to convey infographic message by adding interactivity, and explored technologies by means of which this can be achieved. The aspects of establishing a more efficient communication with end users, such as searchability and accessibility are also considered. SVG infographics are compared with other approaches for creating infographics in raster and vector techniques.

Keywords:

Infographic, Interactivity, Content management, SVG, XML, XSLT, SMIL, ECMAScript

1. Introduction

Information has certain power. However, that same information can, e.g. in media context, be both uninteresting and unimaginative. By means of information visualization and the creation of infographic it can be turned into a more interesting and clear graphical representation of content that will draw reader's attention more than a text paragraph on the web page. Infographic is a simplified and easily understandable visual representation of complex information, data and knowledge by which the process of content understanding in a viewer is accelerated and facilitated (*Eastman, 1980, Owen, 1992, Steele & Iliinski, 2010*).

Visualization of certain content alone can increase its interest and degree of understanding. However, that infographic can be made interactive so users are able to communicate, choose and explore its certain concepts, and then it becomes powerful and effective tool in conveying information and knowledge (*Sorapure, 2010, Moreno & de Oliveira, 2009*).

The use of infographics is very common on the web, and SVG technology is ideal for creating interactive infographics. SVG (Scalable Vector Graphic) is an open standard, based on XML (eXtensible Markup Language) technology that describes two-dimensional vector graphic. It is possible to animate infographics and make them interactive through SMIL technology that is built into the SVG standard. With ECMAScript (standardized version of JavaScript) even a more flexible way of managing content can be achieved (*Duce, Herman & Hopgood, 2002, Moreno & de Oliveira, 2009, Dahlström, Ferraiolo & Grasso, 2011*).

The content of SVG infographics can be dynamically generated by storing data in the XML document and then transforming it through XSLT (eXtensible Stylesheet Language for Transformations) template into a SVG graphic. This is how the separation of content and its presentation is achieved, which means that they can be edited separately, and this has numerous advantages. SVG standard supports CSS (Cascading StyleSheet) specification, that by defining styles, provides additional separation of structure and presentation of XML or SVG content (*Harold & Means, 2002, ***, 2011a*).

2. Theoretical Part

A SVG was developed by W3C consortium and it is compatible with other W3C standards like DOM, CSS, XML, XSLT, SMIL, HTML and XHTML. A SVG file is in fact an XML document and that means that every SVG document begins with an XML declaration (*Dahlström, Ferraiolo & Grasso, 2011*):

```
<?xml version="1.0"?>
<svg version="1.1"
xmlns="http://www.w3.org/2000/
svg"
 xmlns:xlink="http://
www.w3.org/1999/xlink"
width="1200px" height="800px">
   <!-- Drawing comes here. -->
   <circle cx="100" cy="200"
r="50" style=" fill:red;
stroke:blue; stroke-width:2"/>
</svg>
```

An XML declaration includes a recommendation according to which the document is created. A SVG document begins with `<svg>` start tag. SVG code is located between start and end `<svg>` tag, and that element is the root element of the document. Inside `<svg>` element canvas dimensions, a SVG specification that is used in the document and namespace are defined (*Eisenberg, 2002, Dahlström, Ferraiolo & Grasso, 2011*).

The SVG standard includes a certain number of predefined shapes that can be used and manipulated (rectangle, circle, ellipse, line, polyline, polygon, and path). Geometric shapes defined by the SVG standard are very carefully selected and are expected to cover a wide area of application. They are created by corresponding tags (`<circle>`), inside which the attributes defining the structure of this shapes are specified (`cx`, `cy`, `r`), as well as the CSS properties that define visual presentation of this shapes (`fill`, `stroke`, `stroke-width`). Instead of CSS styles, XML presentational attributes, whose syntax is same as syntax of other XML elements, can be used. However, the best content and presentation separation, and thus the most effective manipulation of both, is achieved by means of creating a separate CSS file that is referenced in the head of the SVG document. SVG infographics can include different special effects like transparency, filter effects, gradients, patterns and embedded text (*Eisenberg, 2002, Dahlström, Ferraiolo & Grasso, 2011*).

SVG content can be interconnected with the XML data base through a XSLT template. XML

is primary used to store data and information, which means that it is not concerned with the presentation of certain content, but with the way of a structural and hierarchical organization of data in order to achieve the most effective and easiest manipulation. Such organization is accomplished by XML tags and attributes that describe what particular data represent, as well as their mutual relationship (*Harold & Means, 2002*):

```
<?xml version="1.0"?>
<element attribute="value">
  <subelement>data1</
subelement>
  <subelement>data2</
subelement>
  <subelement>data3</
subelement>
</element>
```

The base of dynamic content generation are XSLT templates. It is necessary to reference the XSLT file, according to which the XML content is transformed, inside the XML document (*Harold & Means, 2002*):

```
<?xml-stylesheet
href="template.xsl" type="text/
xsl"?>
```

The XSLT template specifies the rules for XML content transformation. XSLT processor takes the data from the XML document, processes it according to the instructions from the XSLT document and, as a result, generates a document that is similar to the template, but that has dynamic content (*Tidwell, 2008, ***, 2011a*).

All instructions for XML transformation are contained inside `<xsl:stylesheet>` element (*Tidwell, 2008*):

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.
w3.org/1999/XSL/Transform">
  <xsl:output method="xml"
version="1.0"/>
  <xsl:template match="/">
      <xsl:apply-templates
select="subelement"/>
  </xsl:template>
</xsl:stylesheet>
```

The element `<xsl:output>` specifies the format of the output document as a value of the `method` attribute. A SVG is an XML document so that the value of that attribute is `xml`. The templates are defined inside `<xsl:template>` element. It is assigned to the root element ("`/`" is an xPath expression that signifies the root of the document and, as it contains the whole document, the whole XML document). For applying the template `<xsl:apply-templates>` element is used, and the template will be applied on the XML element whose name is specified inside the `select` attribute of that element (*Tidwell, 2008*).

With SMIL (Synchronized Multimedia Integration Language) specification interactivity can be added to the SVG infographic. SMIL is a standard built into the SVG standard and it includes `animate`, `animateColor`, `set`, `animateTransform`, `animateMotion` animation elements. SMIL constructions usually consist of a trigger (that is defined temporally or in the event context, for example `click`, `mouseover`), target element, changing attribute, starting and ending value of the changing attribute, and duration of the animation (*Eisenberg, 2002, Dahlström, Ferraiolo & Grasso, 2011*).

Much more interactivity options are provided by the ECMAScript technology. To include ECMAScript code into SVG page, `<script>` element is used (*Eisenberg, 2002, Flanagan, 2006*):

```
<script type="text/ecmascript">
<![CDATA[
   //Here comes ECMAScript code
]]></script>
```

Attribute `type` of the `<script>` element specifies the script language used. Expression `<![CDATA[` is an XML command that tells the browser to stop viewing the content situated

inside that command as XML (or SVG) and start viewing it as sign data. This prevents the interpretation of certain special signs (like <, & and similar) that are a part of the ECMAScript code, as a part of the XML syntax. ECMAScript scripts can be specified in the external file and then referenced inside the SVG document. Interactivity of the ECMAScript specification is achieved by event triggers that are almost the same as the ones defined by SMIL specification – the difference is in prefix `on` before the ECMAScript event (`onclick`, `onmouseover`) *Eisenberg, 2002, Flanagan, 2006*).

3. Experimental Part

Visualized information has a much greater impact on the message recipient than that same message in textual form. That is particularly true for statistical data that, only when the data is visualized into graphs and tables, can successfully convey and demonstrate a mutual relationship of particular data values and the difference between them. By creating practical examples, the advantages are explored that the SVG format brings, such as dynamic data control and interactivity with end user (*Tomes, Oates & Armstrong, 1998, Eastman, 1980*).

Infographic was created on the basis of a hypothetical research with the goal of determining the share of each eye colour in the test groups. The research was conducted in four groups, and gathered data was organized into one XML document (*Harold & Means, 2002*):

```
<share>
  <title>SHARE OF EACH EYE
COLOR:</title>
  <color id="AMBER">
    <group id="A">1</group>
    <group id="B">2</group>
    <group id="C">1</group>
    <group id="D">0</group>
  </color>
  <color id="BLUE">
    <group id="A">10</group>
    <group id="B">19</group>
    <group id="C">38</group>
    <group id="D">24</group>
  </color>
    ...
</share>
```

The goal is to represent visually a mutual relationship of this data in bar graph and to calculate the percentages of each eye colour share in the whole test sample. If we assume that the research is conducted many times with different data gathered, graph generation can be automated with XML data base creation. New data are updated inside the XML document and the XSLT processor, based on XSLT template's instructions, generates new SVG graph with new data, while no change in code is necessary. If the graph is created only in a SVG document, for every change in data, new percentage values must be calculated and every SVG element's values must be manually changed.

XSLT template contains all elements of the ultimate SVG graphic, but it lacks real data that define how the graph should look. Data are needed to calculate the height of each bar, share in percentages, to preview each unit's name and the name of infographic. The template includes references on needed data, as well as instructions how these data should be included for the XSLT processor (*Tidwell, 2008, Harold & Means, 2002*):

```
  <xsl:variable name="index"
select="(position() mod 8)"/>
  <xsl:variable name="color">
     <xsl:choose>
        <xsl:when test="$index
= 1">#C99159</xsl:when>
        ...
     </xsl:choose>
  </xsl:variable>
```

To variable `index` numerical value, that corresponds to the position (the order of) each element inside XML document, is assigned, and that is accomplished with `position()` function. Element `when` assigns the specified colour based on that order (*Tidwell, 2008*):

```
  <xsl:variable name="x-off-
set" select="10 + (position() *
80)"/>
  <xsl:variable name="y-offset"
select="400"/>
  <xsl:variable name="y"
select="$y-offset -
(((sum(group)) div 4))*8"/>
  <path style="stroke-width:1;
stroke:black; fill: {$colour}"
   d="M {$x-offset - 30} {$y-
offset} L {$x-offset - 30} {$y}
L {$x-offset + 30} {$y} L {$x-
offset + 30} {$y-offset} Z"/>
```

The values of `x-offset` and `y-offset` variables, which represent the coordinates of right bottom point of the bar (`x-offset` is defined as an equation that calculates the values of x coordinate based on XML data), are defined. Variable represents the height of each bar and it is calculated based on data gathered in the XML database. In the end bars are drawn (`<path>` element) /11/.

The same procedure is executed for every `colour` element - that code is situated inside the element `<xsl:for-each select="colour">` which applies the template on every XML element under that name. For direct preview of each element's value, `<xsl:value-of>` element is used (*Tidwell, 2008*):

```
<text style="font-size:25;
font-family:Calibri;
font-weight:bold;
 font-style:italic;
fill:#705B55;" x="25" y="35">
    <xsl:value-of
select="title"/>
</text>
```

After the XSLT processor processes the XSL template based on XML data, the SVG document is generated (*Figures 1 and 2*) (*Tidwell, 2008, ***, 2011a*).

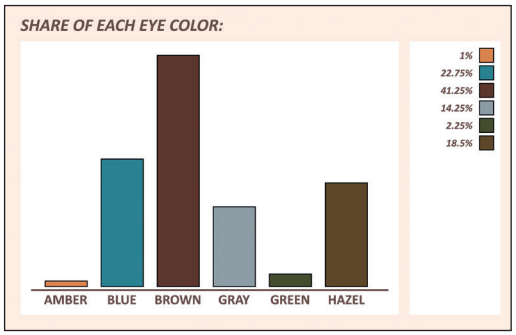

Figure 1. Infographic created based on XML data

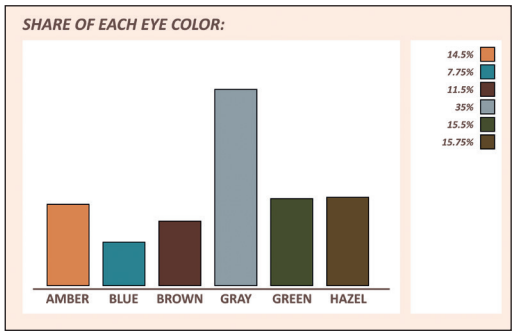

Figure 2. The same SVG infographic created on updated XML data and with same XSLT template

The advantages of interactive transmission of the visual information are also analyzed. When users communicate with infographic, they stop being passive viewers and become active participants in the process of information transmission.

Infographic is created by using exclusively the basic SVG shapes. Interactivity is added with SMIL and ECMAScript standards (*Figure 3*). ECMAScript function `change(evt)` causes the colour change of the SVG object. It assigns, with the `getElementById("S1")` method, new value to the attribute `fill` of the specified element (with the method `setAttribute`) (*Flanagan, 2006*):

```
function change(evt) {
   var j1 = document.
getElementById("S1");
   R = 117; G = 57; B = 17;
   j1.setAttribute("fill", "rgb("
+ R + "," + G + "," + B + ")");
}
```

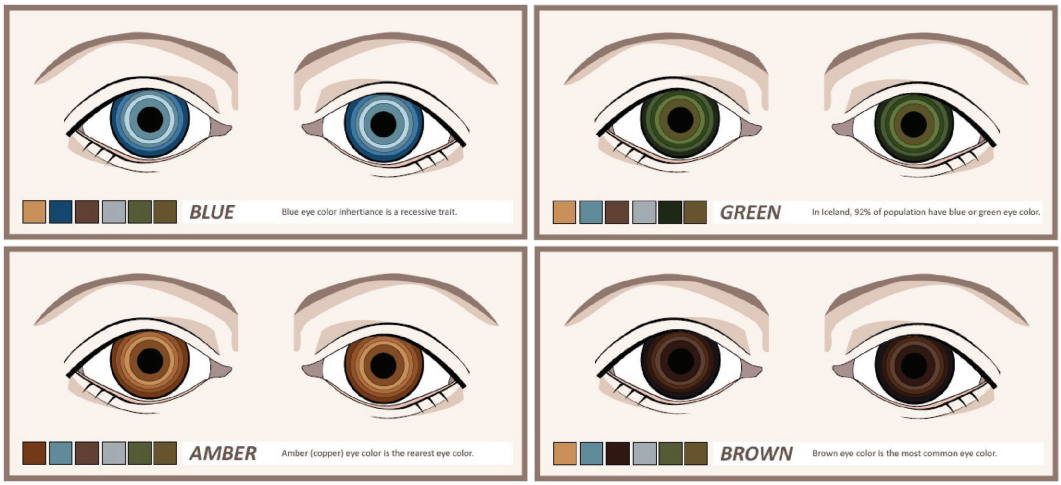

Figure 3. Four different stages of interactive infographic.

That function is invoked inside the SVG element as the attribute value of the event that activates the script (`onclick`). SMIL animation element `set` that causes colour change of that object indicating its *clickability*, is applied on the same SVG element (*Eisenberg, 2002, Flanagan, 2006*):

```
<rect id="linkAMBER"
x="47.5" y="267.5"
fill="#C99159" stroke="#000000"
width="28" height="28"
onclick="change(evt)">
<set attributeName="fill"
attributeType="XML"
from="#C99159" to="#753911"
begin="mouseover"
end="mouseout"/>
</rect>
```

If the web browsers' support of particular standards is taken into account, ECMAScript has an advantage over SMIL specification. Web browser Mozilla Firefox has a built-in SVG support while other web browsers require the installation of SVG content viewing program (the most common is Adobe Viewer). That means that Mozilla is aware of the SVG document and it loads it without any problems. However, animation and interactivity defined by SMIL specification will not work inside Mozilla while ECMAScript will (***, *2011a*).

For infographic to be successful, it needs to be much more than sole information transferor (*Figure 4*). When it provides a new, innovative view on certain data that results in a new level of understanding in the viewers, infographic becomes a superior communicator. Its main purpose is to be informative and that means that the primary goal of infographic's visual design is its ability to convey information. Infographic must be efficient and it has to convey the message clearly. It should not contain too much data that is not strictly connected to the key information it relays. If too much data is included, it will result in the degradation of infographic's organizational structure and content overcrowdings that the viewer will have to examine longer in order to get the key information. That problem can be solved by adding interactivity - then the user decides if she or he will go deeper into the represented issues or not, and in that way the main message is clearer, visually simpler and easier to understand and remember. And finally, infographic should be aesthetically pleasing. Its design should primarily be in function of information presentation. Every part of the design that does not help the process of message conveying is a potential barrier to the successful communication. It lowers the infographic's efficiency and its overall successfulness (*Tomes, Oates & Armstrong, 1998, Owen, 1992, Steele & Iliinski, 2010*).

Figure 4. Example of good and not so good infographic - Alternative ways of SVG infographic's publication.

In figure 4, the first infographic is visually more interesting and it will draw more attention than the second one which is a sole information transferor. Infographic that uses graphic elements has the content and the message repeated on more than one level. SVG is not only written in words but it is visually represented as a format and as a vector graphic format - that information is lost in the second infographic where SVG is only written in words. Also, in the first infographic the same SVG graphic appears on a portable device, on a computer and in printed media amplifying the main message – that the same SVG graphic can be used in different media. This dimension is lost in the second infographic. Although the same structure and information organization is used, the second infographic loses much in information transmission.

4. Discussion

Data and information visualization through the SVG technology provides many advantages. SVG is a vector format and that fact results in a smaller file size, which is in web context a great advantage (faster loading). That is so because SVG graphics do not carry data about every pixel (which is the case with bitmap images), but all image information is stored inside vector mathematical shapes that describe their creation (most commonly Bezier). In that context, vector graphics can be seen as a set of commands for drawing shapes in the output coordinate space that is defined by the output resolution. For the same reason, the size change of an SVG image can be made without lowering the preview quality, no matter what the degree of magnification is. By enlarging the bitmap, defined by a final number

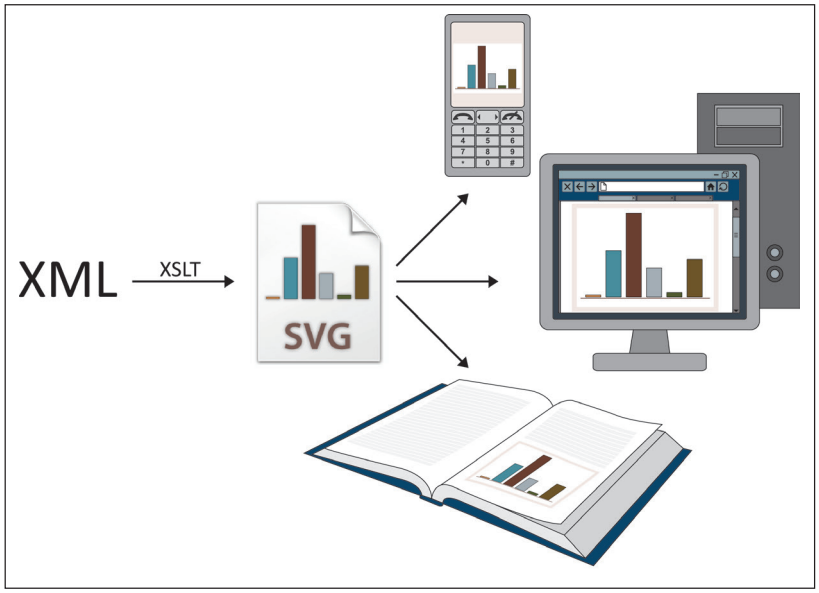

Figure 5. Alternative ways of SVG infographic's publication.

of pixels, the dimensions of pixels are also unavoidably enlarged, and that has a negative visual effect, especially when it comes to the sharpness of the image. The possibility of changing the size of the image and at the same time preserving the quality of its presentation is very important when it comes to the infographics that are often complex and require a closer examination in the sense of readability of the represented information. SVG infographics give more creativity possibilities to its creator like the micro and macro information that are prepared beforehand for certain viewing level. When a user wants to see micro information, SVG infograpic can be magnified to a desired readability level (*Duce, Herman & Hoopgood, 2002, Moreno & de Oliveira, 2009, Eisenberg, 2002, ***, 2011a*).

SVG infographics are resolution-independent and that means that they retain the same preview quality independent of the operating system and device support. Same SVG infographic can be used for web, in print and on the portable devices and still have the same high quality (*Figure 5*). Bitmaps intended for web cannot be used for high quality printing and for every different use it is necessary to create a new file (*Tidwell, 2008, ***, 2011a*).

A SVG is based on the XML technology and it provides the possibility of dynamic interactivity that responds to the user's actions. For the same reason the text in SVG graphic is selectable and searchable. Other technologies' textual information is rasterized, which means that it cannot be searchable. XML's code property of SVG format brings one more big advantage and that is the accessibility to the individuals with impaired eyesight or blind people via special equipment like convertors of text into speech or into Braille. Without it, visual information would be completely inaccessible to them (*Herman & Dardailler, 2002, Duce, Herman & Hopgood, 2002*).

The fact that SVG is a part of the XML world is a big advantage because it opens up many possibilities of dynamic content management: XML data base can, via XML Schema and corresponding XSLT templates, be connected to the generated SVG document (*Figure 6*). A direct consequence of that is the cost reduction of system maintenance: SVG enables a dynamic change of the graphic attributes, which eliminates the need for multiple image files. It also has a positive effect on the reliability of presented data that are changed only on one location, and the development time of the infographic is

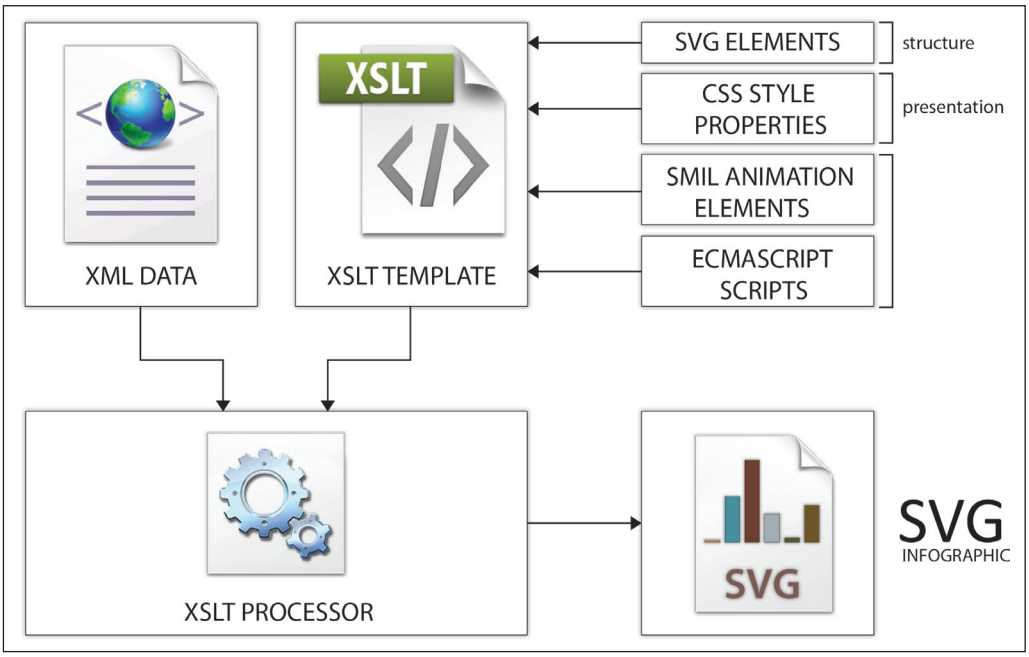

Figure 6. SVG file creation based on XML data base via XSL templates.

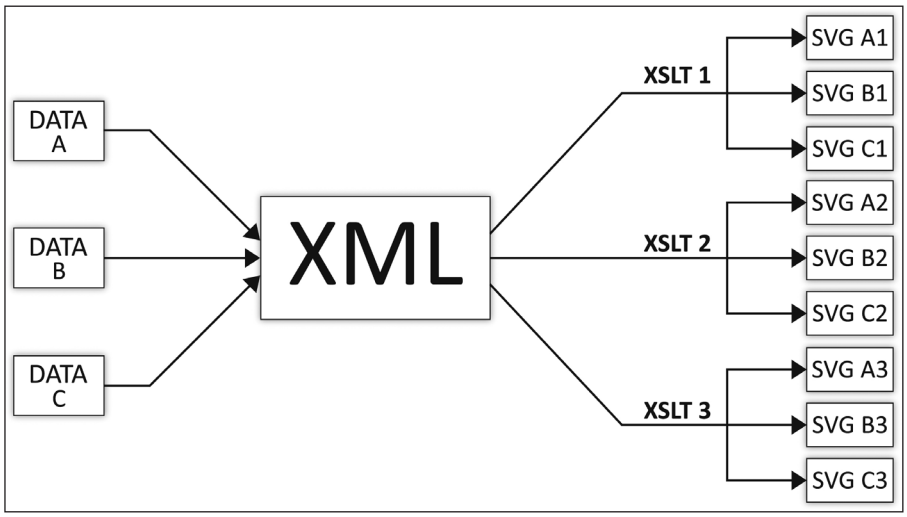

Figure 7. Alternative ways of creating SVG infographics.

reduced. SVG separates the three elements of the traditional content for the web creation workflow: content (data), presentation (graphics) and application logic (scripts). That approach enables the modular way of infographic creation according to different organizational scenarios (*Figure 7*). To change the content of the raster image, a completely new file must be created. And finally, SVG separates the design from the content and each one can be easily edited (*Herman & Dardailler, 2002, Eisenberg, 2002, ***, 2011a*).

SVG infographics can be personalized or visually customized for different demographic groups, cultures, aesthetical preferences or users with special needs. SVG content can be dynamically generated based on the existing information and the end user can interactively choose how that information will be shown. For that purpose few different style templates are created, which determine visual and style characteristics of the infographic's content presentation. Visual impression and the feel of the infographic can be drastically altered with the use of multiple style templates (*Eisenberg, 2002, ***, 2011a*).

For information visualization on the web, Shockwave Flash (SWF) format is more commonly used than the SVG. However, compared to the SWF, SVG has many advantages. SVG is an open-source format, which means that for its usage no expensive software packages are needed. For editing SVG code simple text editors are more than sufficient. XML format of SVG infographic can easily be viewed, read and edited. SWF is a format owned by the Adobe Corporation - its code is hidden from the public and inaccessible to other display technologies. For creating and editing Flash file it is necessary to own Flash software, and even then user would not be able to see what happens in the background of the program (*Duce, Herman & Hopgood, 2002, Cerri & Fuggetta, 2007, Eisenberg, 2002*).

SVG is a real web standard that is deeply integrated with other web standard while this is not the case with Flash (*Figure 8*). SVG content can be created with programs like Adobe Illustrator, CorelDraw i Xara X, although for its creation a simple text editor is sufficient. However, there are many WYSIWYG editors created exclusively for generating SVG graphic like DrawPlus, Beez, WebDwarf and Inkscape that support many advanced SVG possibilities and have very simple and well organized graphic interface (*Figure 10*) (****, 2011a, ***, 2011b*).

Figure 8. SVG's integrity with other web standards.

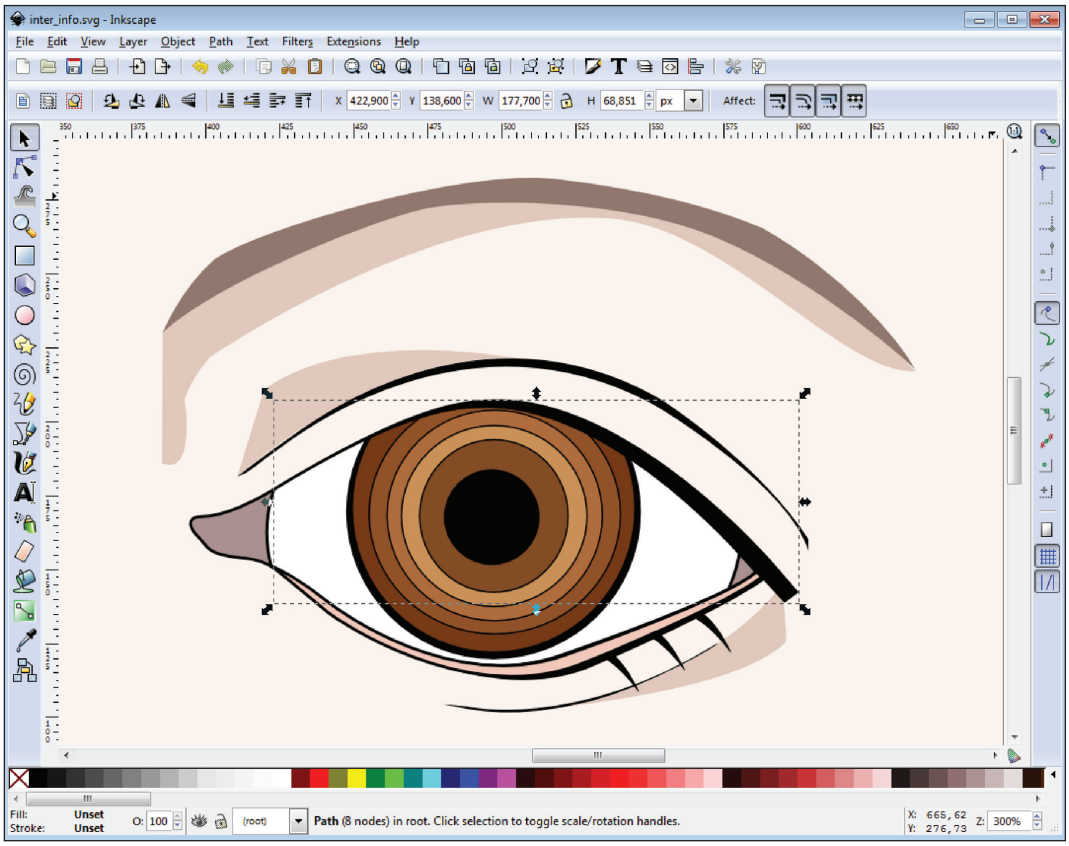

Figure 9. Example o SVG editor's graphic interface – Inkscape

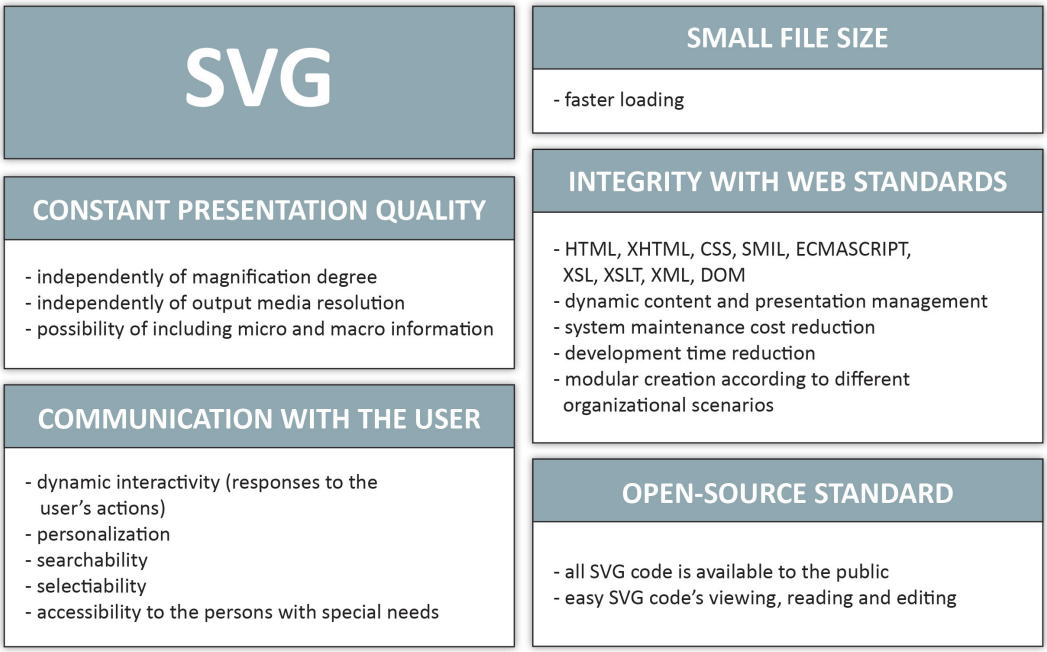

Figure 10. Advantages of infographics created with SVG technology.

Flash has a considerable advantage when it comes to the more complex animations, applications or web pages because some effects that they require are difficult to achieve with SVG. Moreover, a support issue is taken as a shortcoming of the SVG technology. Most web browsers require the installation of an additional program for viewing the SVG content. Although these programs are free of charge, this can pose a problem when it comes to a wider population of users with different computer literacy. However, data and information visualization is usually a part of communication in the computer literate population and should not be a problem. In addition, SVG support grows constantly and it is a matter of time when it will be successfully integrated in all leading web browsers (****, 2011a*).

When it comes to the infographics, advantages of SVG format prevail over all other possible formats (*Figure 10*).

5. Conclusion

Visualization makes information more interesting and noticeable to the viewer. It also analyzes, decomposes and interconnects complex information, thus making it more understandable, concise and integrated. With usage of different graphical elements, colors and images, information can be made visually impressive and appealing. By adding interactivity elements to that kind of infographic, the viewer is actively included in the process of information conveying, she or he participates, uses the information and in that way takes it in easier and remembers longer (*Sorapure, 2010, Eastman, 1980, Steele & Iliinski, 2010*).

Raster infographics are static in size and in content. SVG infographics, whose content is specified as an array of instructions that produce an image after they are evaluated, are dynamic in size (presentation quality is independent of the magnification degree and resolution) and in communication with the user. The text inside the SVG infographic is selectable and searchable, and accessible to the users with special needs. A text that is a part of the raster infographic is no longer a text - it is pixilated, which means that the content of this infographic will neither be searchable nor accessible (*Herman & Dardailler, 2002, Duce, Herman & Hopgood, 2002*).

SVG is compatible with other web standards like HTML, XHTML, CSS, SMIL, XML, XSLT and ECM AScript. It is an open standard that doesn't require the usage of propriety languages and tools like Flash. It is a textual format that is open to the public so it can be acquired and the knowledge can be upgraded by studying the code of other, more complicated SVG files (*Duce, Herman & Hopgood, 2002, Eisenberg, 2002, ***, 2011a*).

SVG separates the design and the application logic from the content: XML data base contains data and information that are via XSLT templates (CSS styles, SMIL and ECMAScript interactive elements) transformed into SVG infographic. A direct consequence of that dynamic content management is the reduction to the system maintenance cost and to the infographic's developing time, increased reliability of the system and bigger flexibility of the organizational scenarios of the infographic creation (*Eisenberg, 2002, ***, 2011a*).

When it comes to the graphic content of the web today, SVG is not as widespread as the raster and the Shockwave Flash formats are. SVG is most commonly used for depicting icons, clip arts and simple illustrations where its full potential (animation, interactivity, searchable text, zoomability, etc.) is rarely reached. On the web today SVG is still mostly present in the academic context and this opens up a whole new area of creative application and usage of SVG graphics in a more practical way.

Due to the number of advantages it provides the future of SVG as a leading format for interactive vector graphic is very promising, and its integration and usage develops and increases daily. Infographics based on the SVG technology are efficient, simple to develop and maintain, and they provide many possibilities of visual design, data organization, animation and interaction with the user.

References

Cerri, D. & Fuggetta, A. (2007) "Open standards, open formats, and open source", Journal of Systems and Software, vol. 80, no. 11,

Dahlström, E., Ferraiolo, J., Grasso, A. et al. (2011) "Scalable Vector Graphic (SVG) 1.1 (2nd Edition)", available at: http://www.w3.org/TR/SVG11/ [accessed: 22nd August 2011],

Duce, D., Herman, I. & Hopgood, B. (2002) "Web 2D Graphics File Formats", Computer Graphics Forum, vol. 21, no. 1,

Eastman, C.M. (1980) "Information and databases in design: A survey of uses and issues in building design", Design Studies, vol. 1, no. 3,

Eisenberg, J. D. (2002) "SVG Essentials", O'Reilly,

Flanagan, D. (2006) "JavaScript: The Definite Guide, 5th Edition", O'Reilly,

Harold, E.R. & Means, S.W. (2002) "XML in a Nutshell, 2nd Edition", O'Reilly,

Herman, I. & Dardailler, D. (2002) "SVG Linearization and Accessibility", Computer Graphics Forum, vol. 21, no. 4,

Moreno, E. D. & De Oliveira, J. I. F. (2009) "Architectural impact of the SVG-based graphical components in web applications", Computer Standards & Interfaces, vol. 31, no. 6,

Owen, C.L. (1992) "Context for creativity", Design Studies, vol. 13, no. 3,

Sorapure, M. (2010) "Information Visualization, Web 2.0, and the Teaching of Writing", Computers and Composition, vol. 27, no. 1,

Steele, J. & Iliinski, N. (2010) "Beautiful Visualization", O'Reilly,

Tidwell, D. (2008) "XSLT, Second Edition". O'Reilly,

Tomes, A., Oates, C. & Armstrong, P. (1998) "Talking design: negotiating the verbal-visual translation", Design Studies, vol. 19, no. 2,

*** (2011a) "What is SVG", available at: http://www.adobe.com/svg/overview/svg.html, [accessed: 21st August 2011],

*** (2011b) "About Inkscape, available at: http://inkscape.org/, [accessed: 24th August 2011].