O Uso de Padrões Abertos para Internet nas Aplicações Gráficas: Uma Aplicação Espacial

João Emile Louis
IAE - CTA
joaol@directnet.com.br

Nandamudi L. Vijaykumar
INPE
vijay@lac.inpe.br

Resumo

Um dos principais meios de difundir informações rapidamente e de maneira clara é fazer o uso de gráficos. Devido à sua importância, aplicações para WEB devem considerar os gráficos como ferramentas essenciais para publicação de documentos sem se esquecer da comunicação entre várias plataformas. A idéia deste artigo é mostrar que aplicações gráficas podem ser elaboradas fazendo uso de alguns padrões de interoperabilidade existentes para WEB (XML, SVG, DOM, etc), especialmente SVG (Scalable Vector Graphics). O artigo abordará um aplicativo que mostra a evolução da trajetória de um veículo sobre projeções cartográficas. Com isso pretende-se mostrar a capacidade de criar gráficos 2D dinâmicos e interativos usando SVG.

Abstract

Graphics play a major role in disseminating information to be grasped in a fast and a clear manner. Due to the increasing popularity of WEB, its applications have to consider graphic tools as essential without downlooking communication aspects among several platforms. This paper discusses that graphic applications can be formulated by using WEB interoperability standards such as XML, SVG, DOM, etc. Special attention is dedicated to SVG (Scalable Vector Graphics). The paper makes use of an example that shows the evolution of the trajectory of a vehicle on cartographic projections. This example will describe the capability of creating dynamic and interactive 2D graphics based on SVG.

1. Introdução

Por volta de 1994 surgiu um grupo chamado *World Wide Web Consortium* (W3C) com a finalidade, entre outras coisas, de desenvolver protocolos comuns na *Web* para promover a sua evolução e garantir sua interoperabilidade.

Uma variedade muito grande de especificações de novos padrões/tecnologias é objeto de estudo deste consórcio. Informações sobre todas as tecnologias podem ser verificadas no site da W3C [1]. Dentre as várias tecnologias, este artigo abordará as seguintes:

- HTML – *HyperText Markup Language*
- XML – *Extensible Markup Language*
- DOM – *Document Object Model*
- SVG – *Scalable Vector Graphics*
- CSS – *Cascading Style Sheets*

Dada a importância conquistada pela *web*, existem outras organizações desenvolvendo padrões para a *internet*. A W3C tenta manter contato com estas outras instituições e consórcios para sempre estar compatível e ser eficiente na resolução dos problemas atuais.

Um outro consórcio que vale mencionar é o *Web3d Consortium*, que é um fórum para criação de padrões abertos para representar os aspectos da tecnologia 3D na *internet*. Ainda existem outras companhias envolvidas com a padronização de processos no domínio de gráficos e internet, tais como, Corel, HP, Sun, Netscape e Adobe.

Em se tratando de visualização de gráficos vetoriais, deve ser considerada a necessidade de que a visualização desses gráficos seja feita dentro de um *browser*. Para isso existem vários visualizadores (*plug-in*) e nesta aplicação o Adobe SVG *Viewer* [2] é utilizado.

Uma aplicação gráfica tornar-se-ia ainda mais atraente se características de interatividade pudessem estar embutidas em suas próprias funcionalidades. Em conseqüência, o processo se torna dinâmico. A interatividade pode ser conseguida através do uso de tecnologias de Javascript (cliente) e PHP, JSP/Servlet (servidor).

2. Especificação das tecnologias

As tecnologias ou padrões visam, em geral, reduzir o custo e a complexidade do desenvolvimento de aplicações ligadas à *internet*, além de tentar manter sua interoperabilidade. A W3C, que é uma organização interessada em garantir a interoperabilidade, procura

desenvolver tecnologias com a capacidade de compartilhar e integrar informações independentemente da plataforma, sistema operacional ou linguagem de programação.

Os padrões que são utilizados nesta aplicação, ou seja, necessários para a visualização da evolução da trajetória de um veículo sobre projeções cartográficas, serão abordados a seguir.

- *HyperText Markup Language*, ou HTML, é chamada de *lingua franca* para divulgação de hipertextos na *web*. No entanto, uma nova especificação foi elaborada para ser a sucessora do HTML. Chamada de XHTML, *Extensible HyperText Markup Language*, ela consiste na família dos atuais e futuros tipos de documentos que podem reproduzir em HTML reformulados basicamente em XML. A aplicação final para este artigo já usa parte desta nova especificação e um dos próximos passos será validá-la conforme a especificação XHTML 1.1.
- *Cascading Style Sheets*, ou CSS, é o meio que autores e leitores de documentos em HTML possuem para definir novos estilos para os documentos, como por exemplo, fontes, cores e espaçamento.
- *Document Object Model*, ou DOM, é basicamente uma interface, usada por uma linguagem de programação, para acessar documentos XML. A grande ajuda aos desenvolvedores é que esta tecnologia, além de fornecer uma representação em forma de estrutura do documento, define a maneira como essa estrutura pode dinamicamente ser acessada por programas ou scripts para atualização de seu conteúdo, estilo ou de sua própria estrutura.
- *Extensible Markup Language*, ou XML, é um formato para organizar documentos e dados como uma estrutura na WEB. Esta especificação não faz nada com os dados, apenas descreve as informações para armazenamento, transmissão, ou processamento de dados por um programa, o qual realmente faz alguma coisa.
- *Scalable Vector Graphics*, ou SVG, é uma linguagem, baseada em XML, de elaboração gráfica bi-dimensional para a WEB. Usando os recursos desta tecnologia, as páginas WEB podem apresentar uma alta qualidade gráfica e as imagens geradas podem ser dinâmicas e interativas. O acesso a documentos XML pode ser feito com o uso do DOM, já que o SVG engloba os recursos de ambos. Qualquer objeto gráfico SVG pode ser manipulado a partir de uma grande variedade de eventos. Estes eventos agem sobre o objeto tornando-o interativo e, assim, o objeto pode ser controlado e modificado com o uso da linguagem Java ou por JavaScripts [3]. A seção 4 é dedicada aos detalhes da estrutura e da sintaxe SVG, assim como ao uso dessas tecnologias em uma aplicação espacial. Devido ao avanço de sua popularidade, visualizadores e editores nativos SVG, tais como Adobe SVG Viewer e W3C Amaya [4], estão sendo disponibilizados. Toda aplicação que faz uso desta tecnologia necessita ter, em seu navegador, um *software* que estenda suas capacidades para poder visualizar determinados efeitos de dentro do navegador. No caso desta aplicação será necessário instalar o *plug-in* Adobe SVG Viewer 3.0.

3. Visão Geral da Aplicação

A aplicação em questão serve para mostrar o uso das tecnologias apresentadas em um problema real. Aplicações envolvendo cálculo de trajetória de foguetes podem ter uma vantagem significativa se a visualização gráfica for considerada. Assim, a exata posição do veículo pode ser examinada fazendo-se a projeção da trajetória em mapas cartográficos [5].

Para se construir tais mapas, necessita-se, primeiro, conhecer como as informações estão armazenadas. Visando a utilização da tecnologia XML, pode-se definir a seguinte estrutura para armazenar os dados do mapa:

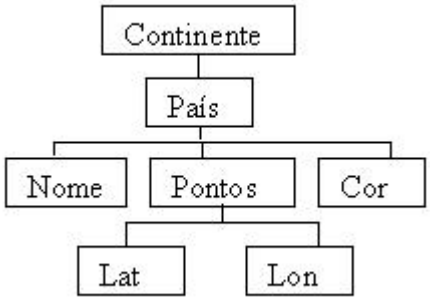

Figura 1. Estrutura XML para o mapa

Todas as coordenadas de contorno dos continentes são representadas por sua longitude e latitude geodésica, assim como o contorno dos países. A cada país, um nome e uma cor estão associados.

Então, com a estrutura definida, pode-se criar o arquivo XML que armazena as informações sobre o mapa:

```
<?xml version="1.0" encoding="ISSO-8859-1"?>

<Continente>
   <Pais>
      <Nome>Brasil</Nome>
      <Cor>rgb(0,200,0)<Cor>
      <Pontos>
         -32.897 -53.261
         -32.837 -53.244
         -32.805 -53.207
         ...
         ...
      </Pontos>
   </Pais>
</Continente>
```

Figura 2. Arquivo XML para o mapa

Vale ressaltar que esse exemplo faz parte da aplicação final proposta para este artigo. Portanto, pode ser considerado como um processo da elaboração do código. E, neste processo inicial, gerou-se um arquivo XML com as coordenadas do contorno dos países o qual será identificado por mapa.xml.

Existe também a necessidade de se conhecer as coordenadas da trajetória em questão. Na realidade, quando se fala em trajetória de um veículo, pode-se ter a trajetória calculada, ou nominal, e a trajetória que foi executada, ou real. Portanto, deve-se criar uma nova estrutura que atenda à necessidade de armazenar todos os dados da trajetória:

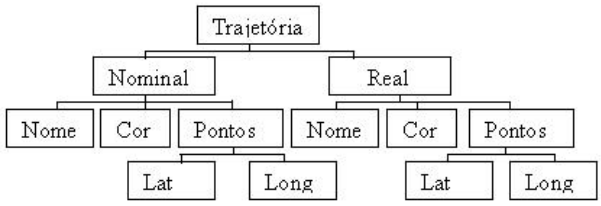

Figura 3. Estrutura XML para a trajetória

Como no caso dos contornos, as coordenadas para a trajetória também são representadas por sua longitude e latitude geodésica e cada trajetória será identificada por nome e cor.

Então o arquivo XML com os dados da trajetória, denominado trajetoria.xml, deve ter o seguinte formato:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<Trajetoria>
   <Nominal>
      <Nome>Sondagem Nominal</Nome>
      <Cor>rgb(200,0,0)</Cor>
      <Pontos>
         -2.44 -44.44
         ...
      </Pontos>
   </Nominal>
   <Real>
      <Nome>Sondagem Real</Nome>
      <Cor>rgb(0,200,0)</Cor>
      <Pontos>
         -2.43 -44.42
         ...
      </Pontos>
   </Real>
</Trajetoria>
```

Figura 4. Arquivo XML para a trajetória

Os dados armazenados no formato XML devem obedecer rigorosamente a estrutura hierárquica pré-definida. Esses arquivos devem passar por um processo de "validação" e assim se tornarem arquivos bem formados (XML well-formed). Esta validação consiste basicamente em confrontar a definição dos dados (DTD – *Data Type Definition*) com a sua especificação. Porém, este assunto foge ao escopo de estudo deste artigo.

Com isso pode-se dizer que a base de dados para a visualização está pronta e bem organizada.E como já foi dito anteriormente, esses arquivos XML são simplesmente descritivos e, portanto necessitam de um agente para processá-los, para depois finalmente desenhar o mapa.

A tecnologia DOM trata da manipulação das estruturas através de scripts, e para isso Javascript é utilizada Essa é uma linguagem interpretada a qual é executada no cliente, possibilitando a interação entre o usuário e o documento.

A tecnologia SVG, também com o auxílio da linguagem Javascript para acessar a estrutura de dados, fica responsável pelo desenho da aplicação na tela do computador, como será visto na seção 4. A visualização dessas imagens depende do *plug-in* SVG Viewer que deverá estar instalado no navegador.

O esquema abaixo mostra como as tecnologias se interagem para produzir o efeito desejado:

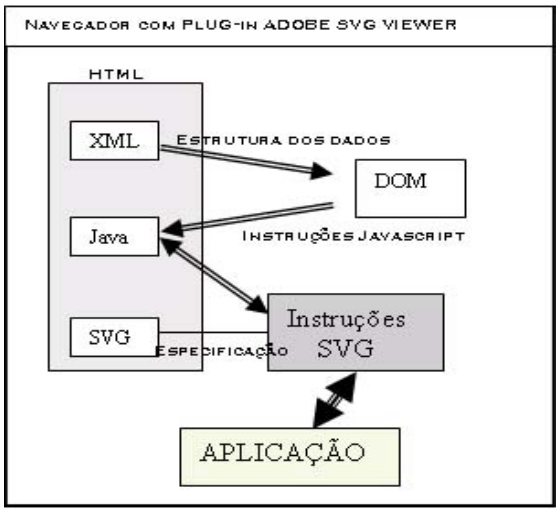

Figura 5. Esquema do uso das tecnologias

3.1. Plug-in SVG

As aplicações SVG são normalmente visualizadas dentro de navegadores como Netscape e Internet Explorer. Estes precisam de um plug-in específico para efetuar a elaboração de qualquer elemento SVG, tais como textos, formas e imagens, além de possibilitar animações e interações. Existem vários visualizadores no mercado, e um dos mais utilizados é o desenvolvido pela Adobe. Além de ser muito bom, ele não tem custo e pode ser instalado através do site da Adobe. Por esses motivos esta aplicação só poderá ser executada em navegadores que possuem o Adobe SVG Viewer.

3.2. Iniciando a aplicação

Esta aplicação é acessada invocando uma página HTML na qual os arquivos SVG estão embutidos e todas as referências aos arquivos XML, assim como aos arquivos Javascript, já estão determinadas, como mostra o script da figura 6.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
…
<script src="Aplic.js" language="javascript"></script>
…
<XML id="mapaXML" src="Mapa.xml"></XML>
<XML id="TrajetoriaXML" src="Trajetoria.xml"></XML>
…
<embed src="Aplic.svg" name="aplic" width="900"
 height="20"  type="image/svg+xml"
 pluginspage="http://www.adobe.com/svg/viewer/install/"/>
…
<p><a href="http://validator.w3.org/check/referer"><img
src="http://www.w3.org/Icons/valid-xhtml11"
alt="Valid XHTML 1.1!" height="31" width="88" /></a></p>
…
</html>
```

Figura 6. HTML da aplicação

Considerando que a W3C recomenda o uso da especificação XHTML 1.1, este arquivo HTML poderá sofrer alterações quando for realizada a validação para tal especificação.

A seguinte imagem será mostrada no navegador:

Figura 7. Imagem inicial

O arquivo XML com coordenadas dos contornos e das trajetórias é interpretado e é visualizado como mostra a figura 7. A próxima subseção descreve como o arquivo é interpretado para produzir a visualização no monitor.

3.3. Fazendo o Mapa

Partindo-se do princípio de que a estrutura dos dados dos arquivos XML é uma estrutura bem formada, a tarefa agora é fazer com que esses dados sejam acessados por meio de uma linguagem de programação para que possam ser manipulados e conseqüentemente projetados na tela.

Então, para se obter o mapa mostrado na figura 7, a tecnologia DOM manipula os dados XML como uma estrutura, e serve como a interface da qual a linguagem Javascript necessita para acessar a estrutura XML. Os comandos Javascript da figura 8 ilustram esta idéia, além de ressaltarem que uma linguagem de programação necessita que as coordenadas a serem projetadas estejam representadas em forma de variáveis.

```
  // mapaXML como definido no arquivo HTML
CoordMapa = mapaXML.XMLDocument;

  // Nomes, cores e pontos dos países
NomePais  = CoordMapa.getElementsByTagName("Nome");
CorPais   = CoordMapa.getElementsByTagName("Cor");
PtsPais   = CoordMapa.getElementsByTagName("Pontos");

  // TrajetoriaXML como definido no arquivo HTML
CoordTraj = TrajetoriaXML.XMLDocument;

  // Nomes, cores e pontos da trajetória
NomeTraj  = CoordTraj.getElementsByTagName("Nome");
CorTraj   = CoordTraj.getElementsByTagName("Cor");
PtsTraj   = CoordTraj.getElementsByTagName("Pontos");
```

Figura 8. Comandos Javascript

Neste ponto, vale ressaltar que todas as informações de nome, cor e coordenadas dos limites dos países e das trajetórias, já estão devidamente atribuídos a variáveis ou vetores. Portanto o próximo passo é fazer realmente a imagem. E, para isso, deve-se usar a tecnologia SVG. Porém, antes de mostrar a elaboração final da aplicação (na seção 5), a seção 4 descreve a tecnologia SVG.

4. Especificação SVG

A tecnologia SVG faz a descrição de gráficos vetoriais baseados em XML, preenchendo assim as necessidades dos desenvolvedores que procuram a mesma qualidade gráfica para as diferentes plataformas.

Algumas características dessa tecnologia assim como partes da sintaxe serão abordadas a seguir.

4.1. Estrutura do documento

Como todo arquivo XML, o arquivo SVG consiste de um cabeçalho <?xml ...>, um elemento raiz <svg> e uma seqüência de vários outros elementos. Esses elementos podem ser agrupados, estilizados, identificados e definidos para uso futuro. Existe um grande número de atributos para cada elemento. Normalmente, os elementos em um documento SVG são executados seqüencialmente, ou seja, aqueles que aparecem primeiro, são desenhados primeiro.

Linha, retângulo, círculo, elipse e polígono são alguns elementos do tipo formas geométricas que o SVG conhece. Esses elementos são portanto alguns objetos gráficos que podem ser transformados, animados, etc.

Um texto em SVG também é considerado um objeto gráfico.

A figura 9 mostra o script e o resultado do uso de alguns elementos. Com qualquer editor de textos, pode-se salvar este script com a extensão svg, e obter o resultado.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" >

<svg id="Hello" width="200" height="100" version="1.1"
xmlns="http://www.w3.org/2000/svg" >

<rect id="Retangulo" x="1" y="1" width="198" height="98"
style="stroke:blue;stroke-width:1.0;fill:yellow;" />

<text style="font-family:Times New Roman;font-size:25"
 id="Palavra1" x="5" y="50" >H E L L O</text>
<g id="World" >
   <circle id="Circulo" cx="150" cy="50" r="40"
   style="stroke:blue;fill:none" />
   <text transform="rotate(10) translate(0, 0) scale(1)"
   x="130" y="20" >W O R L D</text>
</g>
</svg>
```

Figura 9. Exemplo Hello World

4.2. Animação

Um elemento ou objeto gráfico é considerado animado quando um ou mais de seus atributos sofrem modificação depois que a imagem já foi desenhada. Quase todos os objetos em SVG podem ser animados. Portanto mudanças em "tempo real" de atributos como fonte, geometria, localização e cor podem dar efeitos bem complexos de animação. Vale ressaltar que as animações podem ser agrupadas e uma seqüência de animações podem ser desencadeada a partir de apenas uma ação do usuário, ou seja, uma nova animação se inicia com o término de uma outra.

No exemplo da figura 9, para fazer com que a palavra WORLD "passeie" pela tela, adiciona-se antes de </text>, a seguinte instrução:

```
<animate attributeName="x" attributeType="XML"
begin="0s" dur="9s" fill="freeze" from="0" to="130" />
```

4.3. Interatividade

Assim como a animação, a interatividade tem um papel importante na especificação SVG, considerando-se os atuais interesses por páginas dinâmicas na Internet. As ações do mouse ou teclado, por exemplo, podem desencadear a execução de uma animação ou de um script ou mesmo a chamada de uma página da Internet. Essas ações que na verdade disparam eventos podem estar associadas a qualquer objeto gráfico.

Voltando ao exemplo da figura 9, pode-se mostrar uma janela de alerta toda vez que a borda do círculo for clicada. Para isso, entre os elementos <svg > e <rect >, deve ser inserido o seguinte:

```
<script type="text/ecmascript" language="JavaScript"
><![CDATA[
function janela () { alert("O circulo foi clicado"); }
]]></script>
```

E para que a função janela seja acionada, basta colocar o atributo onclick="janela(evt)" no elemento <circle >.

5. A Aplicação Espacial

Como mencionado anteriormente, as tecnologias de interoperabilidade foram testadas para uma aplicação espacial onde é visualizada a trajetória de um foguete. Então, tem-se neste momento a necessidade de desenhar o mapa usando SVG. A primeira pergunta refere-se ao aspecto de como o SVG trata o sistema de coordenadas das imagens. O SVG assume que a origem do sistema está no canto superior esquerdo da área destinada para desenho, sendo que o eixo x caminha para a direita e o eixo y para baixo. Transformações específicas em um elemento, ou em um grupo de elementos, podem ser usadas para estabelecer um novo sistema de coordenadas.

Como a aplicação trata de mapas cartográficos, então o sistema de coordenadas deve abranger de -180 graus a +180 graus no eixo x e de -90 graus a +90 graus no eixo y. O atributo viewBox do elemento svg determina o sistema:

```
<svg ...viewBox="-180 -90 360 180" onload="init(evt)">
```

Assim, as coordenadas do canto superior esquerdo são (-180, -90). Porém o que a aplicação realmente necessita é que este ponto seja a coordenada (-180, 90). Então, uma rotação é necessária, mas o ideal é que esta transformação seja aplicada a todos os objetos de uma só vez. A solução para isso é considerar o agrupamento de objetos. Como exemplo, pode-se agrupar todos os países como um único objeto e então aplicar a operação de rotação em cima deste objeto. Assim, o script contendo o grupo formado pelos países, que serão representados por polígonos, poderá ter a seguinte forma:

```
<g id="AreaTrabalho" transform="rotate(-90)" >
...
  <polygon id="Paises" />
</g>
```

No script acima, foi definido um grupo identificado por AreaTrabalho. Todos os elementos dentro do grupo sofrerão uma rotação de -90 graus. Além disso, foi criado um polígono identificado por Paises. Nota-se que, as coordenadas dos países não constam no script. Isto se deve ao fato de que todas as coordenadas serão inseridas dinamicamente através de comandos Javascript, como se segue:

```
svgMapa  = document.aplic.getSVGDocument();
povoarPolygon = svgMapa.getElementById("Paises");
coordPais = PtsPais(0).text;
```

```
povoarPolygon.setAttribute("points", coordPais);
```

Com o método setAttribute, pode-se, a qualquer momento, atribuir ou modificar um atributo do objeto, ou país, que está associado a variável povoarPolygon. E usando o método createElement pode-se criar quantos objetos forem necessários para representar os diversos países.

```
coordPais[i] = svgMapa.createElement("polygon");
coordPais[i].setAttribute("points", PtsPais(i).text);
```

Neste ponto, a imagem da figura 7 já pode ser visualizada. Incorporando-se as trajetórias ao mapa, forma-se a imagem da figura 10. Por questões de segurança não foram usados dados reais para elaboração desta demonstração.

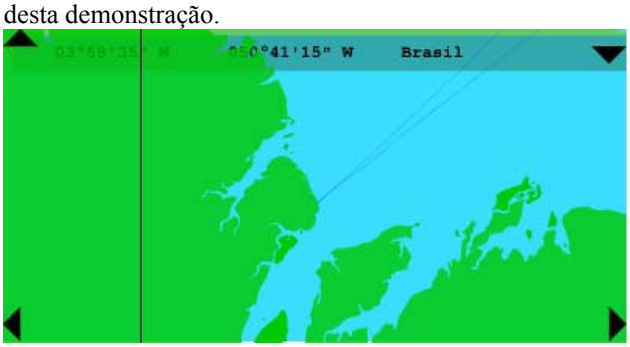

Figura 10. Imagem ampliada da região de Alcântara

5.1. Algumas funcionalidades

Uma das principais funções desta aplicação é a análise pré-vôo de um veículo. Com a funcionalidade de visualizar e conseqüentemente ajudar a definir a melhor trajetória para um veículo, considerando-se todas as restrições e condições impostas para uma determinada missão.

Na análise pós-vôo, esta aplicação também é de grande valia, já que se pode comparar o que foi calculado com o que foi executado, projetando-se a trajetória prevista e a real no mesmo mapa.

5.2. Ações do mouse

Eventos podem ser associados a qualquer objeto inserido no conteúdo SVG. Esses eventos podem ser disparados, por exemplo, por uma ação do usuário no teclado ou no *mouse*. Nesta aplicação não foi vinculado nenhum evento para ações ocorridas através do teclado.

Os eventos, nesta aplicação, estão associados a ações do *mouse*. Assim, qualquer ação do *mouse* pode disparar eventos, tais como *mousemove*, *mouseover*, *mouseout*, e *mouseclick*.

Um fato comum entre estes eventos é que todos estão associados à posição atual do *mouse*.

O evento *mousemove* converte a posição do *mouse*, dada em coordenadas da tela, para coordenadas do mapa e as mostra nas caixas referentes a latitude e longitude.

O evento *mouseover* detecta sobre qual objeto o *mouse* está posicionado e coloca o nome associado a este objeto na caixa referente ao nome do país.

O evento *mouseout* detecta que a atual posição do *mouse* não está sobre nenhum dos objetos inseridos no conteúdo SVG então deixa em branco a caixa referente ao nome do país.

O evento *mouseclick* faz uma aplicação no mapa, obedecendo a escalas pré-definidas.

5.3. Usuário final

As pessoas responsáveis por determinar a melhor trajetória para um dado veículo são, sem dúvida, os usuários em potencial da aplicação descrita neste artigo.

6. Conclusão

O trabalho mostrou como a especificação SVG pode ser uma ferramenta útil para resolver problemas de interoperabilidade, já que aplicações desenvolvidas em SVG podem ser executadas em várias plataformas ou sistemas operacionais. Futuras tecnologias serão incorporadas, como por exemplo, PHP pois poderá ser necessário fazer manipulação de configurações de veículo e ainda fazer controle de acesso à aplicação. Vale ressaltar que existe um esforço da comunidade internacional para a padronização de tais tecnologias, o que faz com que elas estejam em constante evolução, tentando-se assim manter a interoperabilidade.

Referências

[1] World Wide Web Consortium (W3C), Julho 2003. Disponível em http://www.w3.org

[2] Adobe SVG Viewer 3.0, Julho 2003. Disponível em http://www.adobe.com/svg/viewer/install/main.html

[3] Javascript, 30 Julho 2003. Disponível em http://devedge.netscape.com/central/javascript/

[4] Editor/Navegador W3C, 30 Julho 2003. Disponível em http://www.w3.org/Graphics/SVG/SVG-Implementations.html8#svgedit

[5] Louis, J. E.; Vijaykumar, N. L., "Monitoring and Analysing Rocket Trajectory using SVG". Disponível em http://www.svgopen.org/2003/proceedings.do