VGLib2D - Biblioteca de classes para a visualização gráfica de grandes volumes de dados em 2D

Sofia Gameiro Luís Almeida
Centro de Computação Gráfica
Departamento de Coimbra
Centro de Empresas de Taveiro
Estrada de Condeixa
3040 – 912 Coimbra | PORTUGAL
{ Sofia.Gameiro, Luis.Almeida}@coimbra.ccg.pt

Adérito Marcos
CCG / Universidade do Minho
Escola de Engenharia
Departamento de Sistemas de Informação
Campus de Azurém
4800-058 Guimarães, Portugal
marcos@dsi.uminho.pt

Resumo

Este artigo apresenta a implementação de uma biblioteca de classes, em linguagem C#, recorrendo às recentes tecnologias .Net, destinada à criação de documentos/imagens em formato SVG (Scalable Vector Graphics), para a representação, visualização e impressão de grandes volumes de dados em duas dimensões (2D).

O objectivo é tirar partido das vantagens do SVG para efeitos de implementação de aplicações, por exemplo baseadas em Web, onde seja necessária a representação gráfica de grandes volumes de informação, de forma dinâmica e em tempo real, bem como a sua disponibilização imediata para visualização, impressão e download.

Palavras-chave

SVG (Scalable Vector Graphics), XML (Extensible Markup Language), Visualização de Informação 2D, C#, .Net

1. INTRODUÇÃO

A *World Wide Web* constitui, sem sombra de dúvida, o maior repositório e veículo de circulação de informação da actualidade.

A facilidade de acesso, o baixo custo de utilização e a facilidade da navegação tornaram-na no instrumento de pesquisa de informação mais usado em todo o mundo.

Contribui fortemente para este fenómeno a constante evolução das formas de apresentação e disponibilização de informação de todo o tipo e formato. Tal evolução abrange aspectos importantes como o *design* e aparência da apresentação da informação, a facilidade de navegação entre conteúdos, a possibilidade de animação e interacção com a mesma, a compactação e rapidez de *download,* entre outros. Paralelamente, há ainda que garantir a possibilidade de pesquisar essa informação, seja qual for o tipo de formato em que esta se encontra.

No que respeita a informação textual, tanto a sua visualização como pesquisa estão bastante optimizadas. A informação gráfica, no entanto, não é tão facilmente pesquisável ou visualizável. Tal deve-se a algumas limitações ainda existentes na actual tecnologia *Web*, no que respeita à apresentação de imagens 2D, não animada ou vídeo, que consistem, por exemplo, no facto de a maior parte do material gráfico ser apresentado num dos três formatos de varrimento seguintes: GIF, JPEG, PNG, os quais, para além de distorcerem qualquer factor de escala diferente de 1:1, não são tão facilmente pesquisáveis [Probets,01].

O SVG é uma tecnologia XML que veio contornar esta situação: para além de permitir a representação de informação gráfica em formato compacto, genérico e portátil [Eisenberg, 02], a sua natureza textual (implícita à sua representação XML) facilita a pesquisa da informação representada [Probets,01].

Assim sendo, a biblioteca implementada - VGLib2D - visa a criação de documentos e imagens SVG de forma apropriada à representação e visualização de grandes volumes de dados em 2D, uma vez que se considerou que o SVG seria, por estas e outras razões explicitadas mais adiante, o formato mais adequado para a representação de informação gráfica com forte componente vectorial ou representada por meio de vectores.

Nas restantes secções deste artigo faz-se uma breve referência às tecnologias usadas, abordando-se de seguida a tecnologia SVG – características e potencialidades - e alguns aspectos técnicos no que respeita à implementação da biblioteca. Finalmente apresentar-se-á um contexto de aplicação da biblioteca implementada e far-se-á referência a trabalho futuro.

2. TECNOLOGIAS UTILIZADAS

Como plataforma de desenvolvimento optou-se pela plataforma .Net [Microsoft, 03], um conjunto de tecnologias que pretende simplificar o desenvolvimento de aplicações *Web* e para plataforma Microsoft Windows [Duthie, 02]. Tecnologicamente actuais, componenciais, e, ao basear-se num conjunto de *XML Web Services* (pequenas aplicações reusáveis escritas em XML, uma linguagem dirigida para a unificação da representação e circulação de informação), estas tecnologias revelaram-se altamente integráveis.

Neste âmbito, a biblioteca de classes foi implementada em C#, uma linguagem de programação da Microsoft®, relativamente recente, desenhada especialmente para a plataforma .Net. Baseada no paradigma de programação para objectos, a linguagem C# é derivada do C e do C++,

apresentando grande semelhança com o C++ e a Java. Para além disso, o C# baseia-se numa vasta biblioteca de classes presente no *.Net Framework*, promovendo a robustez, consistência e eficiência do *software*.[Drayton, 02]

Apesar da plataforma .Net não ter *namespeaces* (conjunto de classes específicas) para trabalhar o SVG, foi possível recorrer ao suporte disponível para XML, o qual garante a criação de potencialmente qualquer tipo de imagens usando SVG, a partir de qualquer tipo de informação disponível [Wahlin, 02].

Toda a programação foi efectuada recorrendo ao *Visual Studio .Net* da Microsoft®.

3. A TECNOLOGIA SVG

3.1. Introdução

Uma das mais excitantes tecnologias XML actuais é o *Scalable Vector Graphics* (SVG), que, fornecendo uma sintaxe baseada em XML, pode ser usada para a criação e visualização de imagens, desde figuras a *reports* e gráficos, numa variedade de dispositivos [Wahlin, 02].

O SVG permite o uso de elementos e atributos específicos, definidos no *SVG Document Type Definition (DTD)*, publicado no site *da World Wide Web Consortium* - W3C ([W3C, 00a]), para a criação de imagens. Apesar de haver uma vasta lista de elementos e atributos a aprender, é relativamente fácil trabalhar com SVG, e editores visuais de companhias tais como a Adobe® e a JASC podem exportar imagens em formato SVG [Wahlin, 02]. Para a visualização num *browser* é necessário um *plug-in* próprio, sendo o mais difundido o Adobe® SVG Viewer ([Adobe, 01]) .

No texto 1, e correspondente figura 1, é apresentado um exemplo dos elementos básicos usados para desenhar formas e texto, criando uma imagem simples [Wahlin, 02]:

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C// DTD SVG 1.0//EN"
"http://www.w3.org/TR/2001/REC-SVG-
20010904/DTD/svg10.dtd">
<svg width="300" height="150">
<rect x="31" y="36" width="240" height="59"
style="fill:#02027a;stroke:#000000;stroke-
width:1"/>
<text x="56px" y="67px" style="fill:#ffffff;
font-family:Arial; font-size:18"> My First SVG
Document </text>
</svg>
```

Texto 1: Exemplo de um documento SVG

Figura 1 : Visualização gráfica do documento usando o *Adobe SVG Viewer*

3.2. Principais características do SVG

Há quem afirme que a forma de representação gráfica na Internet está a ser revolucionada pelo formato SVG. O sucesso deste formato deve-se a algumas das suas características [Probets,01] [Eisenberg, 02] [Mouza, 02] [Marriott, 02], as quais foram decisivas para a sua eleição no que respeita à implementação da VGLib2D, nomeadamente:

- **Formato vectorial**;
- **Sintaxe XML;**
- **Ficheiro de texto**, permitindo que motores de busca indexem ficheiros SVG ou componentes SVG embebidos em ficheiros HTML (*HyperText Markup Language*);
- **Independente de plataforma e dispositivo;**
- **Permite interacção**, ao possuir um número de procedimentos orientados ao evento, que permitem um certo grau de interacção com o utilizador;
- **Integração com CSS (*Cascade Style Sheets*) e XSL(*Extensible Stylesheet Language*)**;
- **Flexibilidade**, no que respeita aos atributos dos elementos, permitindo uma separação entre a apresentação gráfica e a estrutura geométrica dos elementos;
- ***Interface* DOM (*Document Object Model*)**, graças à qual se pode modificar o documento do lado do servidor. Quando isto acontece o *browser* SVG transmite de imediato a alteração para o *display* gráfico;
- **Permite animação.**

Apesar de tudo o SVG apresenta também algumas limitações que é preciso ter em conta:

- **Não suporta adaptação do lado do cliente de acordo com diferentes condições de visualização.** Esta é uma desvantagem importante se tivermos em consideração que o uso de diferentes tipos de dispositivos é cada vez mais frequente, indo desde pequenos ecrãs de *PDA*s (*Personal Data Assistants*) e telefones móveis até aos grandes dispositivos de salas de conferências;
- **Formato textual do ficheiro pode, em alguns casos, resultar em ficheiros de grandes dimensões**. Tal problema pode ser ultrapassado desde que se tenham em conta alguns cuidados a nível da implementação que visam garantir a optimização dos ficheiros. Este aspecto será abordado mais adiante.

3.3. Áreas aplicacionais do SVG

O impacto do SVG está a ser de tal forma marcante, aos mais variados níveis, que são várias, e muito díspares, as áreas de conhecimento a recorrerem a esta nova tecnologia para efeitos de representação e visualização gráfica de dados.

Referenciam-se aqui apenas quatro exemplos, seleccionados entre os muitos disponíveis, que se consideram significativos e ilustradores da flexibilidade de aplicação deste formato, bem como do seu sucesso e difusão a nível da computação gráfica.

Pretende-se com esta abordagem fazer uma aproximação ao estado da arte da tecnologia SVG, bem como apresentar de forma mais concreta, recorrendo a exemplos reais, as potencialidades deste formato.

3.3.1. SVG e cartografia

Uma das áreas na qual o impacto do formato SVG está a ser mais marcante é a da cartografia. De facto a cartografia representa uma aplicação perfeita para o SVG: os mapas são, por definição, representações vectoriais de áreas 2D, distribuídas por camadas. A especificação SVG contém esse mesmo conceito de camadas ou agrupamentos, que é essencial para a representação de informação usando os Sistemas de Informação Geográfica (SIG).

Os mapas GIF ou JPEG usados até muito recentemente, apesar de não apresentarem problemas de compatibilidade com as ferramentas *Web* e ambientes de trabalho, não oferecem potencialidades de interacção, *zoom*, controlo de *layers*, entre outras. O SVG veio resolver todos esses problemas: não sendo um substituto para um Sistema de Informação Geográfica completo, o SVG personifica uma nova forma de representar informação geográfica de qualidade na *Web* [Seff, 02].

Em [Custom, 03] pode ser encontrado um exemplo de aplicação (ver figura 2):

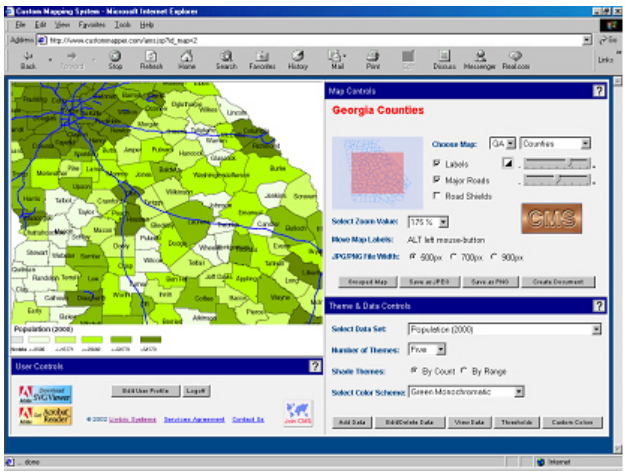

Figura 2 : Custom Mapping System v1.1, desenvolvido pela Limbic Systems, Inc

3.3.2. Chemical Markup Language (CML™)

A *Chemical Markup Language* (*CML*™) [CML, 03] é uma nova aproximação para a gestão de informação molecular que tira partido das tecnologias XML. Sendo capaz de conter estruturas de informação extremamente complexas, actua desta forma como um mecanismo de intercâmbio ou de arquivo.

Em [Adobe, 03] pode encontrar-se o seguinte exemplo, no qual informação armazenada em CML™ é manipulada via uma XSL (*Extensible Stylesheet Language*) de forma a criar um SVG passível de visualização e interacção (ver figura 3):

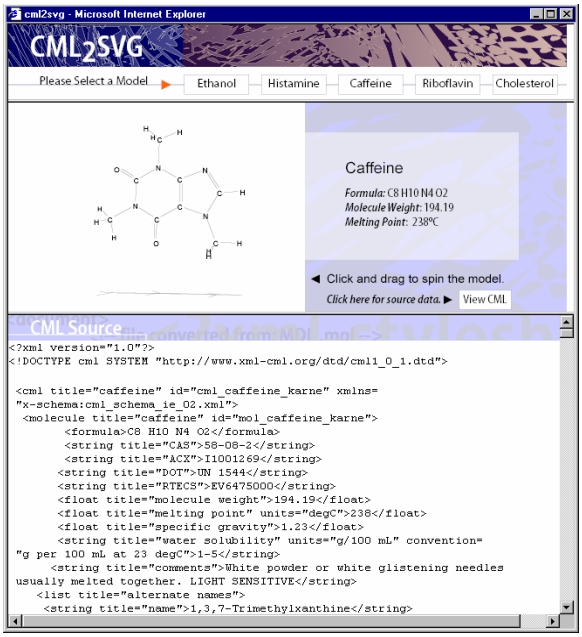

Figura 3 : Visualização 3D de uma molécula de cafeína em formato SVG, obtida a partir do formato CML™

3.3.3. SVG e Dispositivos Móveis

Dispositivos móveis tais como os *PDAs*, *PalmPilots*, *SmartPhones*, entre outros, ajudam na gestão e optimização de sequências de actividades a realizar pelos seus utilizadores.

Neste âmbito, recorreu-se ao formato SVG para a representação e visualização de toda a informação gráfica associada a essas actividades e tarefas, cuja representação fiel compreende a interacção de uma série de técnicas de visualização tais como a representação espacial do local onde ocorrem as actividades, representação temporal das tarefas, representação textual das mesmas através de listas, texto contínuo e hierarquias, representação de prioridades através do uso de cores, entre outros [Bieber, 03].

3.3.4. Bloco de notas de laboratório

O último exemplo apresentado será talvez o mais surpreendente no que respeita à demonstração da aplicabilidade do formato SVG.

Trata-se de um novo conceito, o "*augmented laboratory notebook*": um espaço de trabalho que pretende fazer a ligação entre informação em formato papel e digital, em contexto laboratorial.

Sem se pretender aprofundar os aspectos técnicos inerentes aos protótipos implementados, importa referir que cada bloco de notas consiste num conjunto de folhas e que cada folha contém várias camadas distintas de informação. O SVG foi o formato seleccionado para toda a representação e posterior visualização da informação gráfica inerente ao uso deste dispositivo, que os autores acreditam poder vir a atingir as proporções de um

autêntico conjunto de ferramentas de realidade aumentada para trabalho sobre papel [Mackay, 02] (ver figura 4).

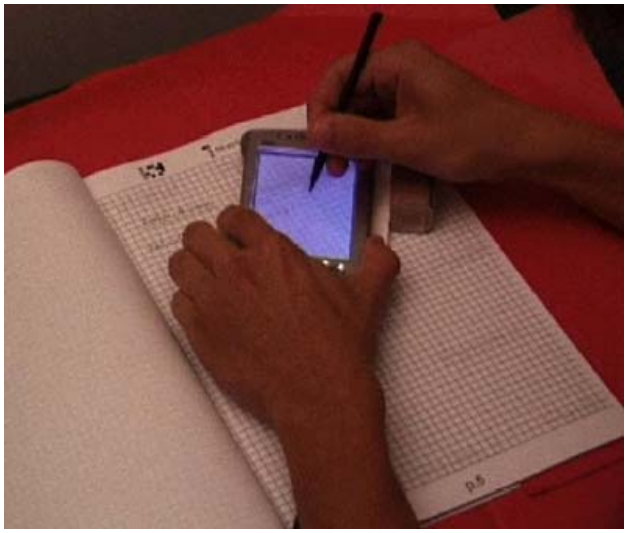

Figura 4 : Um dos protótipos implementados, o *a-book* [Mackay, 02].

4. A BIBLIOTECA

4.1. Introdução

Como já foi anteriormente referido, a plataforma .Net não contem *namespeaces* para trabalhar o SVG. Esta é, de momento, uma limitação inerente, também, às restantes plataformas e linguagens de desenvolvimento. Tal significa que toda a implementação respeitante à geração de ficheiros em formato SVG é ainda trabalhada a muito baixo nível, o que é de certa forma preocupante, se se tiver em conta o estado da arte deste formato, bem como a quantidade de aplicações actualmente desenvolvidas com suporte SVG.

Foi precisamente com o intuito de colmatar esta lacuna que se optou pela implementação da VGLib2D: uma biblioteca de classes que permite a criação e escrita de ficheiros SVG. Mais do que isso, a VGLib2D pretende tirar partido das potencialidades do SVG, atrás citadas, de forma a possibilitar a representação gráfica, o *download* e impressão de grandes volumes de dados, com a rapidez e qualidade desejadas.

A VGLib2D compreende classes de mais baixo nível para a criação e definição de propriedades de elementos básicos do SVG, tais como *SVGCircle*, para a criação e definição de um círculo, *SVGLine*, para a criação e definição de uma linha, etc., bem como classes de mais alto nível, que funcionam como *interface* para a criação de documentos SVG: escrita de cabeçalho (*header*), escrita de elementos, de comentários, etc., e que se baseiam nas classes de mais baixo nível referidas (ver figura 5). Pretende-se com esta divisão a separação visível do núcleo relativo ao SVG propriamente dito (elementos e atributos SVG), da *interface* da biblioteca, acessível ao programador e dirigida para a criação e gestão de documentos que contêm a definição de todas as entidades gráficas a representar.

A VGLib2D foi implementada para funcionar de forma equiparada aos *namespaces* do *.Net Framework* e garantindo a interoperabilidade e comunicação com os mesmos.

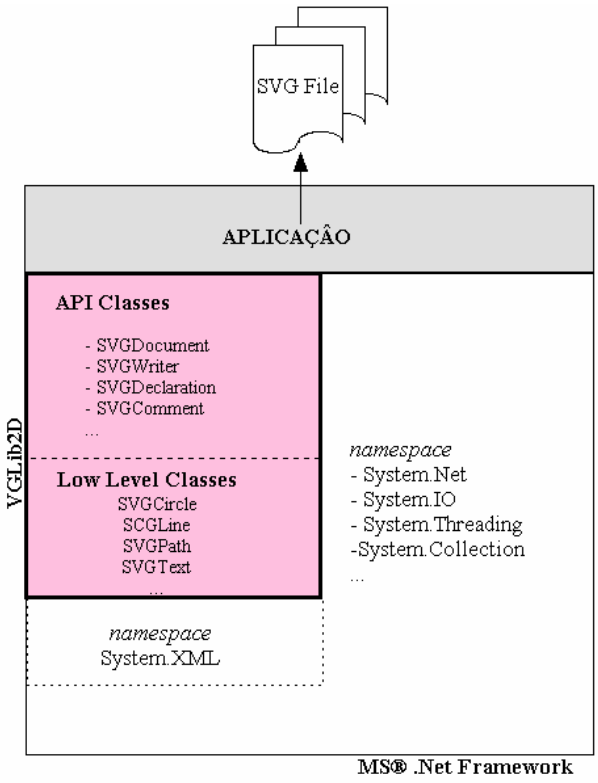

Figura 5 : VGLib2D – Arquitectura e Comunicação com o *.Net Framework*

4.2. Estratégia de implementação

Para efeito de implementação partiu-se do *namespace* System.Xml presente no *.Net Framework Software Development Kit*, e incluído na biblioteca de classes FCL (*Framework Class Library*) da Microsoft®.

Este *namespace* fornece suporte para a gestão de documentos XML, de acordo com um conjunto de standards definidos pelo *World Wide Web Consortium* (3WC). As suas classes implementam objectos que respeitam a especificação XML1.0([W3C, 98]) e o DOM Core Level 1 e Core Level 2 ([W3C, 00b]) [Drayton, 02]. Pretendeu-se com isto garantir a validação do formato XML dos ficheiros e a criação de um DOM coerente e estruturado que permita edições posteriores, com vista à obtenção de interactividade, entre outros.

Para validação dos ficheiros SVG criados utilizou-se o software XMLSpy®[XMLSpy, 03].

Uma vez que um ficheiro SVG mais não é do que uma variação de um ficheiro XML, e tendo em conta que partimos do *namespace* System.Xml, considerou-se que a arquitectura da VGLib2D deveria reflectir, até certo ponto, a arquitectura do *namespace* referido. Assim sendo, as classes de *interface* tais como *SVGDocument*, *SVGDeclaration*, *SVGWriter*, etc.… apresentam os mesmos métodos que se podem encontrar nas classes paralelas pertencentes ao *namespace* System.Xml,

acrescidas de outros métodos específicos dirigidos ao formato SVG e suas particularidades. É o caso de métodos tais como *SetWidth*, *SetHeight*, *SetUnits*, *DrawLine*, *DrawCircle*, entre muitos outros. Pretende-se assim estender a filosofia de utilização inerente aos *namespaces* do *.Net Framework* à VGLib2D, assegurando desta forma a uniformidade da *interface*: um programador experiente na plataforma .Net não deverá ter quaisquer dificuldades no uso da biblioteca implementada.

A biblioteca implementada visou conseguir o máximo grau de abstracção e uma hierarquia consistente, de forma a garantir a sua generalização a aplicações diversificadas.

Para a criação dos ficheiros SVG existem algumas questões de optimização que se devem ter em conta, tais como o uso preferencial de alguns atributos, como sejam *path* em substituição de múltiplas utilizações do atributo *line* ou *polyline,* o *H* e o *V* para linhas horizontais e verticais, entre outros. Considerações semelhantes aplicam-se a curvas[Probets,01]. Após algum debate, decidiu-se que caberia ao utilizador a gestão das questões de optimização dos ficheiros, através da selecção correcta dos métodos referentes aos diferentes atributos a utilizar. No entanto a VGLib2D possui os mecanismos que proporcionam essa optimização.

4.3. Características

A VGLib2D possibilita a criação de documentos SVG com as mais variadas dimensões, definidas em todas as unidades métricas consideradas válidas pela especificação deste formato.

Para além de permitir a associação de folhas de estilo e a definição de estilos gerais para o documento, estilos estes automaticamente adoptados pelos elementos criados que os referenciem, a VGLib2D permite também a definição de elementos gráficos com o seu estilo próprio, independente dos estilos gerais do documento.

Todos os elementos ou entidades que fazem parte da especificação do formato SVG são considerados na VGLib2D. Sobre estes elementos é possível a definição e edição dos seus atributos e a aplicação de operações geométricas tais como translações, rotações e escalamentos.

Sem se pretender entrar em grande detalhe, exemplos de elementos suportados e respectivos atributos são o caso das linhas, com atributos como a espessura, cor, opacidade e estilo (contínuo, interrompido,…); do texto, com atributos como o tipo, tamanho, espessura e estilo de letra, alinhamento, espaçamento entre letras e palavras, orientação (vertical, horizontal, inclinada segundo determinado ângulo, com leitura da direita para a esquerda, de baixo para cima, etc.); entre tantos outros cuja enumeração exaustiva não é viável no contexto deste artigo.

A *interface* para a definição dos elementos gráficos é intuitiva e devidamente documentada, obedecendo todas as funções a uma sintaxe simples e evidente, como é o caso do seguinte exemplo, que permite a definição de uma recta/linha contínua:

- *DrawLine (SVGWriter filePointer, string id, double x1, double y1, double x2, double y2, int lineWidth, string color);*

Neste caso temos que *filePointer* é o ponteiro para o ficheiro SVG criado, *id* é o identificador da linha criada, *x1*, *y1*, *x2* e *y2* são as coordenadas da linha e *lineWidth* e *color* são, respectivamente, a espessura e a cor da linha.

Caso se pretendesse aplicar, na criação da linha, um estilo pré definido, invocaríamos antes a seguinte função, em que *style* é o identificador desse estilo:

- *DrawLine (SVGWriter filePointer, string id, double x1, double y1, double x2, double y2, string style);*

Finalmente a VGLib2D possibilita ainda a inserção de elementos em formato de varrimento, a definição de padrões e gradientes, de curvas de Bézier, e o uso de filtros e a definição de áreas geométricas com ou sem preenchimento.

Até ao momento não foram ainda contempladas classes ou funcionalidades destinadas a animação ou interacção com o utilizador em tempo real. A biblioteca foi, no entanto, arquitectada de forma a possibilitar esta implementação futura.

4.4. Contextos de aplicação

A VGLib2D permite o desenvolvimento de todo e qualquer tipo de aplicações que considerem o recurso ao SVG para a visualização de informação, desde que esta seja passível de representação/"tradução" gráfica 2D. Tal significa que pode ser utilizada no desenvolvimento de aplicações SVG em todas as áreas de aplicação atrás referidas, com a virtude de transportar, desta forma, toda a programação referente à geração de documentos SVG para um mais alto nível, que se optimizará progressivamente à medida que novas funcionalidades/classes forem implementadas.

A VGLib2D foi testada e refinada quando da sua utilização no desenvolvimento de uma aplicação *Web* para a gestão de Horários Técnicos de Comboios para a Refer,E.P., na qual se revelou crucial a nível da implementação da componente de *output* gráfico.

Um dos componentes requeridos para a aplicação implicava a visualização de Gráficos de Circulação, mapas técnicos onde aparece representada a circulação de um dado número de comboios ao longo de um intervalo horário e de uma sequência de estações. Estes mapas devem respeitar os horários em vigor, o período de validade dos comboios (período temporal de circulação destes comboios) e o seu regime de frequência (dias da semana em que circulam), entre muitos outros aspectos. Foi especificamente nesta componente que se aplicou a VGLib2D, com o objectivo concreto da geração dos Gráficos de Circulação em formato SVG.

Esta aplicação, acessível ao utilizador através de um simples *Web browser*, apresenta uma arquitectura Cliente-Servidor baseada em tecnologia Asp.Net e numa base de dados relacional SQL (*Structured Query*

Language), como se pode observar, de forma esquemática, na figura 6.

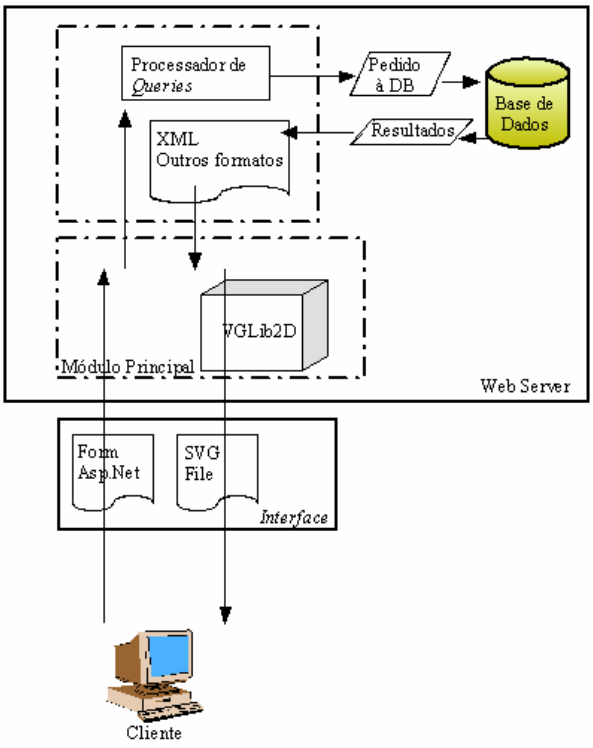

Figura 6 : Arquitectura do módulo referente aos Gráficos de Circulação da aplicação

A *interface* com o utilizador consiste assim num Formulário de Pesquisa (*form* Asp.Net) onde devem ser seleccionadas as linhas e estações que se pretendam consultar, intervalo horário, período de validade dos comboios e regime de frequência (ver figura 7).

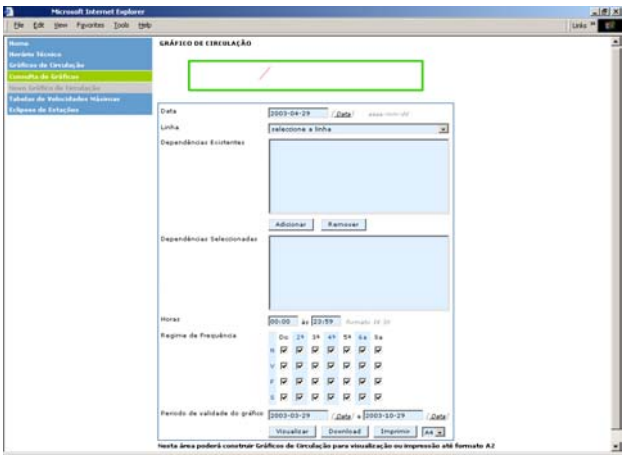

Figura 7 : Formulário de pesquisa

A aplicação acede de seguida à base de dados através de *Data Queries* correspondentes aos dados introduzidos pelo utilizador, devolvendo resultados correspondentes aos comboios que circulam nos intervalos definidos pelo utilizador. Cada comboio é definido por um conjunto de informação genérica (primeiro número de identificação, segundo número de identificação, data de inicio de circulação, data de final de circulação, data de inserção do comboio na base de dados, descrição, identificador operacional, identificador comercial, etc.…), e um conjunto de nós ou estações correspondentes ao percurso efectuado (número de identificação da estação, número de identificação da linha, hora de chegada, hora de partida, tempo de paragem, etc.….). Tanto a informação genérica como a relativa aos nós é bastante complexa, podendo o número de comboios ascender aos dois mil, nos casos em que o intervalo de pesquisa é extenso (por volta de um ano).

Toda a informação resultante é submetida a pré processamento, sendo depois "traduzida" e representada graficamente, de forma dinâmica, através do recurso à VGLib2D, ficando o ficheiro SVG resultante automaticamente disponível para visualização, impressão e *download*.

Um comboio, na sua representação gráfica, mais não é do que uma linha que percorre determinadas estações (eixo vertical) em determinado período temporal (eixo horizontal). Desta forma, recorreu-se a funções da VGLib2D para a criação de *popylines* e *paths*, para representar as trajectórias dos comboios ao longo das várias estações. Objectos adicionais de texto foram usados para as legendas, uma vez que todos os comboios têm de ter associada a informação genérica mencionada atrás. O uso de cores é também obrigatório na definição dos comboios, uma vez que esta funciona como elemento classificativo de diferentes tipos de comboios.

Convém salientar, no entanto, que o gráfico apresentado recorre apenas a uma parcela muito reduzida de elementos do SVG e usufrui de uma margem reduzidíssima das suas potencialidades.

Os Gráficos de Circulação, para além da visualização, estão especialmente orientados para a impressão. Este era, efectivamente, um dos requisitos primordiais a cumprir, uma vez que, devido à complexidade e ao elevado montante de informação muitas vezes contido em cada gráfico, a sua visualização e análise no ecrã mostra-se insatisfatória. Foi assim preciso garantir a possibilidade de criação de ficheiros SVG de grandes dimensões, que podem ir desde o formato A4 (14,8cmx21cm) até ao A0 (84,1cmx118,9cm), passíveis de impressão em plotters específicas.

Apesar de ter sido possível a criação de ficheiros SVG com estas dimensões, a sua impressão só foi possível até ao formato A2(42cmx59,4cm) devido a limitações do *plug-in*, que obrigaram a que esta impressão fosse efectuada via *browser*. Prevê-se que esta limitação seja ultrapassada com a nova versão do *plug-in*, que promete disponibilizar funcionalidades de impressão.

Factores relevantes para o uso da VGLib2D foram o facto da aplicação desejada ser uma aplicação Web e de ter sido indicada como plataforma de desenvolvimento a plataforma .Net.

Houve, no entanto, uma outra intenção na escolha do SVG como formato a adoptar para a representação gráfica da informação: por um lado, com o SVG, e

recorrendo à utilização de grupos, é possível permitir ao utilizador opções de visualização interactiva da

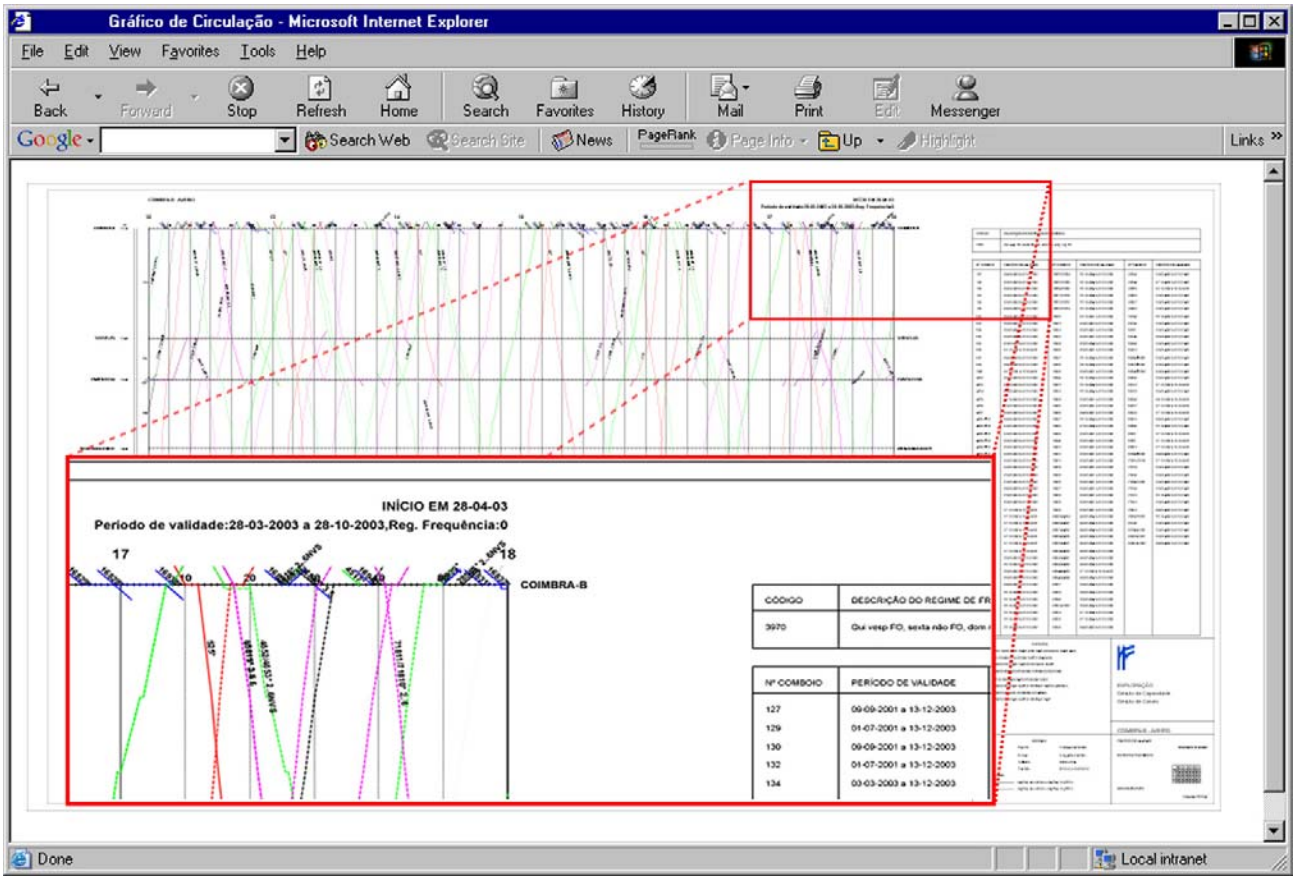

Figura 8 : Exemplo de um Gráfico de Circulação

informação tais como ocultar, evidenciar ou visualizar apenas determinados subgrupos de informação, do total que está contido no ficheiro SVG. Este tipo de funcionalidade pode ser equiparado ao uso de *layers*, técnica de visualização amplamente difundida em todo o tipo de *software* que trabalha com informação gráfica, e que permite a repartição da informação geométrica disponível em camadas isoladas, facilitando a sua leitura e interpretação. Por outro lado, ao possibilitar animação, o formato SVG permite que se venha a implementar, num futuro próximo, Gráficos de Circulação dinâmicos, em que o utilizador observa em tempo real a circulação dos comboios. Esta aproximação recai numa abordagem recente em que a noção tradicional de documentos como sendo maioritariamente compostos por entidades estáticas se tem vindo a modificar em direcção a uma nova perspectiva em que se envolvem constantemente artefactos informativos. São os chamados *Live Documents* : com as tecnologias emergentes, como é o caso do SVG, podem-se enriquecer os documentos informativos com componentes multimédia que possibilitam o acesso a informação adicional, interactividade e trabalho cooperativo [Weber, 02].

Um outro aspecto que levou ao uso da VGLib2D foi o facto de se estar a trabalhar com grandes volumes de informação e respectiva visualização gráfica: como já foi referido, alguns Gráficos de Circulação chegam a apresentar informação referente a cerca de dois mil comboios. Tal significa ficheiros com um DOM bastante complexo e sobre os quais é necessário incidir todo um conjunto de medidas de optimização atrás referidas. Assim mesmo, em termos comparativos e a título de exemplo, importa salientar que o documento da figura 8, em dimensões consideradas legíveis (em mapas tão complexos e abarcando intervalos horários não inferiores a 12 horas, considera-se que um formato de papel inferior ao A2 não é legível) e com qualidade de impressão mínima (150dpi), em ficheiros de formatos de varrimento JPEG e TIFF apresenta, respectivamente, 1674KB e 41769KB, enquanto que em formato SVG apresenta 210KB. A escolha do SVG como o formato de *output* da VGLib2D mostrou ser, também neste aspecto, crucial.

De referir ainda que os ficheiros SVG permitem compactação no formato SVGZ recorrendo à ferramenta Gzip [Gzip, 03], mantendo-se passíveis de visualização no *plug-in.*

5. CONCLUSÃO E TRABALHO FUTURO

A VGLib2D revelou-se de grande utilidade para a implementação de aplicações baseadas em SVG e orientadas para a representação gráfica 2D de grandes volumes de informação.

Veio também colmatar a lacuna presente no *.Net Framework*, no que respeita à disponibilização de um *interface* de mais alto nível para a geração de documentos SVG. O contexto de aplicação apresentado veio reforçar esta ideia, bem como evidenciar, mais uma vez, a grande flexibilidade do formato SVG e, consequentemente, cimentar a certeza de que este era o formato de *output* mais indicado a adoptar pela VGLib2D.

No que respeita a trabalho futuro, um dos aspectos de maior importância a considerar é o de dotar a VGLib2D de funcionalidades que suportem uma extensão ao SVG denominada de *Constraint Scalable Vector Graphics* (*CSVG*), que permite maior flexibilidade na descrição dos elementos gráficos. Com o CSVG, uma imagem pode conter objectos cujas posições e outras propriedades são especificadas em relação a outros objectos e não em termos absolutos. O CSVG permite retardar o *layout* dos objectos definidos no documento até ser efectuado o *rendering* pela aplicação cliente, o que resulta numa maior flexibilidade quanto ao uso de diferentes dispositivos de visualização[Badros, 01]. Ultrapassar-se-á, desta forma, um dos principais inconvenientes do SVG.

A mais longo prazo prevê-se a implementação de novas classes destinadas às funcionalidades de animação e interacção em tempo real.

6. AGRADECIMENTOS

Uma referência especial deve ser feita à Rede Ferroviária Nacional - REFER EP, à Meticube, Lda, e à noLimits consulting, S.A., com as quais o Centro de Computação Gráfica colaborou, implementando entre outras componentes a VGLib2D, para o desenvolvimento da plataforma de gestão de informação de material circulante actualmente em uso na REFER EP.

7. REFERÊNCIAS

[Adobe, 01] Adobe® SVG Viewer 3, 2001. URL: http://www.adobe.com/svg

[Adobe, 03] CML2SVG. URL: http://www.adobe.com/svg/demos/cml2svg/html/index.html

[Bieber, 03] Gerald Bieber, Volker Leck. Scalable Vector Graphics for SaiMotion. *Computer Graphik Topics – Reports on Computer Graphics, 1/2003 Vol.15.~* http://www.inigraphics.net/publications/topics/2003/issue1/1_03a08.pdf

[Badros, 01] Greg J. Badros , Jojada J. Tirtowidjojo , Kim Marriott , Bernd Meyer , Will Portnoy , Alan Borning. A constraint extension to scalable vector graphics. *Proceedings of the tenth international conference on World Wide Web.* April 2001, 489-498.

[CML, 03] Chemical Markup Language (CML™). URL: http://www.xml-cml.org/

[Custom, 03] Custom Mapping System V1.1. Limbic Systems, Inc. URL: http://www.custommapper.com/index.jsp

[Drayton, 02] Peter Drayton, Ben Albahari, Ted Neward. C# in a nutshell – A desktop quick reference. O'Reilly & Associates, Inc. 2002.

[Duthie, 02] G. Andrew Duthie, Matthew MacDonald. Asp.Net in a nutshell – A desktop quick reference. O'Reilly & Associates, Inc. 2002.

[Eisenberg, 02] J. David Eisenberg. SVG Essentials. O'Reilly & Associates, Inc. 2002.

[Gzip, 03] The gzip home page. http://www.gzip.org/

[Mackay, 02] Wendy E. Mackay , Guillaume Pothier , Catherine Letondal , Kaare Bøegh , Hans Erik Sørensen. Interaction in the real world: The missing link: augmenting biology laboratory notebooks. *Proceedings of the 15th annual ACM symposium on User interface software and technology.* October 2002, 41-50.

[Marriott, 02] Kim Marriott , Bernd Meyer , Laurent Tardif. XML Applications: Fast and efficient client-side adaptivity for SVG. *Proceedings of the eleventh international conference on World Wide Web.* May 2002, 496-507.

[Microsoft, 03] Microsoft® .Net™ http://www.microsoft.com/net/

[Mouza, 02] Cédric du Mouza , Philippe Rigaux. GIS and the internet: Web architectures for scalable moving object servers. *Proceedings of the tenth ACM international symposium on Advances in geographic information systems.* November 2002, 17-22.

[Probets,01] Steve Probets, Julius Mong, David Evans, David Brailsford. Hypermedia and Graphics 2: Vector graphics: from PostScript and Flash to SVG. *Proceedings of the 2001 ACM Symposium on Document engineering.* November 2001, 135-143.

[Seff, 02] George Seff. SVG and GIS. 2002. URL: http://www.directionsmag.com/article.php?article_id=198

[W3C, 00a] Wordl Wide Web Consorcium. DTD - W3C SVG 1.0 Specification - Candidate Recommendation 20001102. URL: http://www.w3.org/TR/SVG/svgdtd.html

[W3C, 00b] Wordl Wide Web Consorcium. The Document Object Model. 2000 URL: http://www.w3.org/DOM

[W3C, 98] Wordl Wide Web Consorcium. Extensible Markup Language (XML) 1.0 – W3C Recommendation 10 Feb 98. URL: http://www.w3.org/TR/1998/REC-xml-19980210

[Wahlin, 02] Dan Wahlin. SVG: .Net Graphics Courtesy of XML. *XML & Web Services Magazine.* August/September, 2002.

[WebDraw, 03] Jasc® WebDraw™. URL: http://www.jasc.com/products/webdraw/

[Weber, 02] Anke Weber , Holger M. Kienle , Hausi A. Müller. Live documents with contextual, data-driven information components. *Proceedings of the 20th annual international conference on Computer documentation.* October 2002, 236-247.

[XMLSpy, 03] XMLSpy http://www.xmlspy.com/