O DESENVOLVIMENTO DE UM PADRÃO GRÁFICO GKS

João Ricardo de Freitas Oliveira
Fernando Yutaka Yamaguchi

DEPARTAMENTO DE PROCESSAMENTO DE IMAGENS
INSTITUTO DE PESQUISAS ESPACIAIS
Caixa Postal 515
12.201 - São José dos Campos - SP

SUMÁRIO

Este trabalho descreve as motivações, requisitos, técnicas, e decisões que nortearam a implementação do padrão gráfico internacional GKS, resultando no GKS-INPE.

ABSTRACT

This work describes the motivations, requirements, techniques, and decisions that guided the implementation of the international graphics standard GKS, resulting in GKS-INPE.

1. INTRODUÇÃO

Este trabalho descreve a implementação de um pacote gráfico padrão GKS no âmbito do Instituto de Pesquisas Espaciais - INPE. O GKS-INPE, em sua implementação, satisfaz a várias características de um produto deste tipo (eficiência, mínimo uso de memória, controladores de dispositivo residentes).

O objetivo maior do projeto foi permitir que os sistemas desenvolvidos no INPE - tanto em hardware como em software - pudessem se beneficiar de todas as vantagens de um pacote gráfico padrão: portabilidade de programas entre diferentes instalações (equipamento e sistema operacional), independência de dispositivos físicos, cultura e terminologia comuns (portabilidade de programador). A flexibilidade decorrente do acesso ao código-fonte também foi requerida.

2. O PADRÃO GRÁFICO GKS

O GKS (ISO 7942-1985 e ANSI X3.124-1985) é um pacote gráfico padrão (Enderle et alii, 1987; Cunha et alii, 1987) que trata dos problemas específicos de apresentação gráfica.

O conceito de "estação de trabalho gráfica" é utilizado pelo padrão GKS, com o objetivo de formalizar a noção de dispositivo gráfico.

O GKS trabalha com três sistemas de coordenadas: coordenadas globais (WC), coordenadas normalizadas de dispositivo (NDC), e coordenadas de dispositivo (DC). O aplicativo trabalha com coordenadas globais e define dois tipos de transformações: a transformação de normalização, que mapeia as coordenadas de WC para NDC, e a transformação de estação, de NDC para DC. Cada transformação é especificada através de uma janela e um viuporte.

As primitivas de saída gráfica do GKS são: polilinha, polimarca, área (preechimento), texto, e matriz de células. Para o traçado de outras primitivas básicas tais como a circunferência, o GKS provê a "primitiva genérica" (GDP), definida de forma dependente da implementação. As primitivas de saída possuem atributos (cor, tipo de linha, direção, etc.) que podem ser ajustados pelo programa aplicativo.

O GKS controla a entrada de dados de forma independente de dispositivos, através da definição de dispositivos lógicos de entrada: localizador (par de coordenadas), vetorizador (vetor de pares de coordenadas), opção (número inteiro), sequência (cadeia alfanumérica), avaliador (valor real), e seleção (identificador de segmento).

O GKS permite a manipulação de primitivas gráficas de forma conjunta, através de segmentos, que podem ser exibidos em uma superfície de visualização, transformados, copiados, armazenados, e apagados.

Para permitir o armazenamento a longo prazo de figuras, o GKS possui a capacidade de gerar e interpretar meta-arquivos. Através dessa facilidade é possível a transferência de figuras para outros sistemas, aumentando de forma considerável a flexibilidade do núcleo gráfico GKS.

O GKS mantém listas e tabelas contendo informações estáticas e dinâmicas acerca do sistema. Através de funções de consulta estas informações podem ser recuperadas e utilizadas pelo aplicativo.

O padrão gráfico GKS pode ser implementado em nove níveis diferentes, balanceando um compromisso entre a quantidade de funções e o tamanho (memória ocupada) do padrão GKS. O nível mais baixo (0a) inclui apenas um conjunto mínimo de funções de saída, enquanto que o nível mais alto (2c) inclui todas as funções do GKS. A vantagem de utilizar um nível baixo, além da economia de memória, é o menor tempo de execução do aplicativo.

Com o objetivo de prover recursos tridimensionais, a definição de funções 3D pela proposta de padrão internacional ("draft international standard") ISO/DIS 8805-1986 extende o padrão atual com a inclusão de novos níveis.

3. MOTIVAÇÕES PARA A IMPLEMENTAÇÃO DO GKS-INPE

O INPE está envolvido com a implementação de programas aplicativos, tendo como requisito de projeto a utilização do padrão gráfico GKS. Exemplos desses aplicativos são:

- o SGI - Sistema de Informações Geográficas (Alves,D. et alii), que é um banco de dados geográficos com capacidade para adquirir, armazenar, combinar, analisar, e recuperar informações codificadas espacialmente;

- o MICRO-MAGICS (Camara,G. et alii), desenvolvido juntamente com o Centro Europeu de Previsão do Tempo de Médio Prazo (ECMWF), para a apresentação gráfica de dados meteorológicos, como campos de vento, temperatura e umidade.

A decisão tomada pelo Departamento de Processamento de Imagens (DPI) do INPE para que fosse desenvolvida uma implementação própria do GKS baseou-se nas seguintes constatações, decorrentes da posse do programa fonte:

- Habilidade de escrever controladores de dispositivos para periféricos de interesse (CGA, EGA, SITIM-150, plotadoras, ratinho, mesas digitalizadoras, e outros). Vale ressaltar que o DPI desenvolve equipamentos específicos para processamento de imagens e computação gráfica, destacando-se o uso de processadores gráficos e numéricos avançados, como os da linha TI 340x0 e TI 320xx.

- Possibilidade de prover os níveis desejados conforme as necessidades de cada aplicativo.

- Capacidade de prover características (fontes de texto, tipos de marca, gdp, etc.) específicas requeridas pelos aplicativos desenvolvidos pelo DPI.

- Capacidade de otimização dos algoritmos do GKS.

- Possibilidade de desenvolver implementações do GKS para vários ambientes computacionais.

- Oportunidade para aprofundar os conhecimentos a respeito do padrão GKS.

- Possibilidade de integração a redes locais.

- Possibilidade de geração e utilização de uma quantidade irrestrita de cópias do pacote GKS, sem ônus adicional.

A contrapartida dessa decisão foi a alocação de 2 pessoas-ano para a implementação do nível 0a.

4. REQUISITOS DE PROJETO

A partir da identificação das características desejadas para o projeto GKS-INPE, os seguintes requisitos foram delineados:

- O ambiente-alvo de desenvolvimento foi a linha de micro-computadores PC-compatíveis, sob sistema MS-DOS versão 2.x ou mais recente. Futuras extensões para supermicros com ambientes UNIX (ou equivalentes) também são requeridas.

- O espaço de memória livre deve permitir o desenvolvimento de aplicativos com grande nível de complexidade, como o MICRO-MAGICS.

- Interfaces ("binding") para as linguagens de programação C e FORTRAN.

- Desempenho compatível com as melhores implementações existentes (como o pacote GKSGRAL da empresa alemã GTS-GRAL).

- Maximização da possibilidade de sucesso.

- Código e dados estruturados, permitindo fácil compreensão, manutenção, e evolução para outros níveis.

- Aproveitamento da capacidade de processamento local em dispositivos gráficos avançados.

- Possibilidade de inserção de novos controladores de dispositivos sem a necessidade de recompilar o GKS.

5. CONSIDERAÇÕES DE IMPLEMENTAÇÃO

5.1 USO DE TIPOS ABSTRATOS DE DADOS (TAD) NO GKS-INPE

A decisão de utilizar TAD no desenvolvimento do GKS-INPE decorreu das vantagens conhecidas que a técnica oferece, que são:

- Compreensibilidade, que permite mais fácil depuração, modificação e evolução;

- Capacidade de modificações do código fonte sem afetar de modo sensível o restante do programa;

- Habilidade de reutilizar em outros programas os TAD que já tenham sido definidos e testados. A confecção de uma biblioteca de TAD deverá ter um impacto positivo na produtividade da equipe nos projetos subsequentes.

Adicionalmente, os TADs se adequam de forma natural às aplicações de Computação Gráfica, que lidam com objetos pictóricos e seus atributos (Câmara et alii, 1988).

Como a linguagem utilizada ("C") não possui mecanismos e facilidades específicas para o suporte direto do uso de TAD, é necessário observar uma postura de programação que permita o desenvolvimento da técnica adotada. O acesso às estruturas dos TADs deve ser feito somente através das funções pertinentes a cada abstração.

Exemplificando a utilização de TAD na construção do GKS, é mostrada parte da rotina que implementa a primitiva polilinha (gpl em FORTRAN). Esta rotina foi desenvolvida tomando por base, entre outros, três TAD e suas operações:

a) **CAIXA**

Descrição:

A abstração **CAIXA** é utilizada para representar um retângulo, definido através das coordenadas de seu vértice inferior esquerdo e de seu vértice superior direito. Utilizações típicas desta abstração incluem retângulos que definem janelas, viuportes e recortes.

Estrutura:

```
typedef struct
{
     float     x_inf,
               y_inf,
               x_sup,
               y_sup;
}CAIXA_TIPO, *CAIXA;
```

b) **TRANSF**

As transformações de normalização e de estação são definidas a partir de duas abstrações **CAIXA**: uma janela e um viuporte.

```
typedef struct
{
     CAIXA     janela;
     CAIXA     viuporte;
}TRANSF_TIPO, *TRANSF;
```

g_transf_cria (TRANSF transf)

modifica: transf.
efeito: Carrega em transf todas as informações necessárias que definem uma transformação (de normalização ou de estação).

c) **PONTOS**

A abstração **PONTOS** é utilizada para representar uma sequência de pontos, através de dois vetores, x e y, contendo as coordenadas dos pontos a serem traçados, e de um escalar num_pontos, representando o número de pontos dados.

```
typedef struct
{
     int       num_pontos;
     float     *x,
               *y;
}PONTOS_TIPO, *PONTOS;
```

g_pontos_aloca (int num_pontos)

devolve: PONTOS.
efeito: Inicialmente aloca espaço para abstração PONTOS, em seguida, baseado no valor num_pontos, aloca memória para x e y.

g_pontos_cria (PONTOS pontos, float x[], float y[])

modifica:pontos.
efeito: Carrega os vetores de entrada x e y na abstração pontos.

g_pontos_trf_nrm_exec (PONTOS pontos, TRANSF transf)

modifica:pontos.
efeito: Executa a transformação de normalização nas coordenadas pertencentes a pontos.

A parte relevante da rotina que implementa a polilinha, em linguagem C, é mostrada a seguir:

```
gpl (int num_pontos, float x[], float y[])
{ /* número de pontos, vetor x, vetor y */
    PONTOS    pontos_poli;
    TRANSF    transf_n, transf_e;
...
        /* ALOCA ESPAÇO PARA PONTOS */
    pontos_poli = g_pontos_aloca (num_pontos);
        /* COLOCA X E Y EM PONTOS */
    g_pontos_cria (pontos_poli, x, y);
        /* CRIA TRANSF. DE NORMALIZAÇÃO */
    g_transf_cria (transf_n);
        /* EXECUTA TRANF. DE NORMALIZAÇÃO */
    g_pontos_trf_nrm_exec (pontos_poli, transf_n);
        /* CRIA TRANSF. DE ESTAÇÃO */
    g_transf_cria (transf_e);
        /* EXECUTA TRANSFORMAÇÃO DE ESTAÇÃO */
    g_pontos_trf_est_exec (pontos_poli, transf_e);
        /* CHAMA O CONTROLADOR DE DISPOSITIVOS */
    g_driver_pontos (num_est, pontos_poli);
...
}
```

5.2. USO DE CONTROLADORES RESIDENTES DE DISPOSITIVOS

Qualquer implementação do GKS pode ser dividida em duas partes: dependente e independente de dispositivos. A parte dependente de dispositivos é normalmente desenvolvida como controladores de dispositivos ("device drivers").

A determinação de utilizar controladores residentes de dispositivos ("resident device drivers") decorreu naturalmente dos requisitos de projeto, pois apresenta a vantagem de mínimo impacto em termos de tamanho de código objeto. No caso da implementação do GKS com controladores embutidos no código-fonte, o código-objeto gerado cresce na proporção dos dispositivos suportados.

O GKS-INPE permite a instalação de vários controladores simultaneamente residentes em memória. Assim, controladores novos poderão ser escritos e instalados sem a necessidade de recompilar a parte independente de dispositivos do GKS.

A decisão crítica do projeto foi a escolha do ponto de separação das partes independente e dependente de dispositivos. A partir da constatação de que, no ambiente IBM-PC / MS-DOS, os aplicativos que utilizam o GKS fazem uso (tipicamente) de uma, ou no máximo duas estações simultâneas, decidiu-se manter a porção independente de dispositivos o mais simples possível, deixando-se para o código do controlador (ou para as facilidades de um dispositivo "inteligente") as tarefas que, como o preenchimento de área, poderão eventualmente ser executadas de uma forma muito mais eficiente em dispositivos adequados.

Um dispositivo com capacidade de traçado de linhas, tipos de linha, preenchimento de área, fontes de texto, recorte ("clipping"), poderá ter todos estes seus recursos utilizados pelo GKS, a partir da escrita apropriada de um controlador para este fim. Quanto mais recursos o dispositivo possuir, mais simples será o código do controlador.

6. ESTADO ATUAL DO GKS-INPE E SUA EVOLUÇÃO

O nível 0a do GKS-INPE foi desenvolvido pelos autores no período de abril a novembro de 1989, incluindo-se a entrada e saída de meta-arquivos. A figura a seguir foi obtida em uma plotadora Digicon TDD43, com penas de 0.7mm, com a utilização do aplicativo Micro-MAGICS, que faz uso do GKS-INPE.

Estima-se que até o fim de abril de 1989 estará disponível a versão 1b, uma vez que algumas características deste nível já estão implementadas. A partir do mês de novembro de 1989, duas pessoas foram adicionadas à equipe de projeto (Missae Yamamoto e Sérgio Rosim).

Será feito um esforço para alcançar o nível 1c nesse mesmo prazo, com a inclusão dos modos de operação de entrada amostra ("sample") e evento ("event"). O maior desafio reside na implementação do modo evento, que requer detecção de entrada assíncrona e manutenção de uma fila de eventos, no ambiente IBM-PC / MS-DOS.

Na versão atual, o espaço de memória utilizado pelo GKS-INPE ("driver" residente e código objeto) em um programa de aplicação típico é da ordem de 100 k bytes, o que permite a execução de programas gráficos complexos.

No futuro, planeja-se o desenvolvimento do GKS-3D, para a geração de figuras no espaço. A expectativa é que, em decorrência do uso de TAD, esta evolução aconteça de modo a não interferir significantemente na porção já acabada da versão bidimensional.

7. AGRADECIMENTOS

Gostaríamos de agradecer a todos aqueles que, de uma forma ou outra, nos ajudaram nessa tarefa. Em especial, a Carlos Alberto Felgueiras pelo trabalho pioneiro no desenvolvimento do GKS no DPI; a Flávio Roberto Dias Velasco, nosso expert em programação orientada por objeto e tipos abstratos de dados; a André Battaiola e Elisa Nishimura, pelas sugestões e discussões; a Gilberto Camara, pelas discussões técnicas, decisões, apoio, e sugestões; e a Ricardo Cartaxo Modesto de Souza, pela sua experiência, espírito crítico, e participação, indicou com precisão o caminho a ser trilhado, propiciando o sucesso deste projeto.

9. BIBLIOGRAFIA

ALVES,D.S.; ERTHAL,G.J.; CAMARA,G.; FELGUEIRAS,C.A.; PAIVA,J.A.C.; OLIVEIRA,E.A.; DIAS,L.A.V.; GODOY JR.,M.; ABRAHÃO,A. "Sistemas de Informação Geográfica". Anais do XXI Congresso Nacional de Informática, Rio de Janeiro, agosto de 1988.

CAMARA,G.; NISHIMURA,E.; BATTAOILA,A.; NING,C.; MASSA,L.; VIJAYKUMAR,N.; DAABECK,J. "Micro-MAGICS: Meteorological Graphics on Microcomputers". Submetido para apresentação no II Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, 1989.

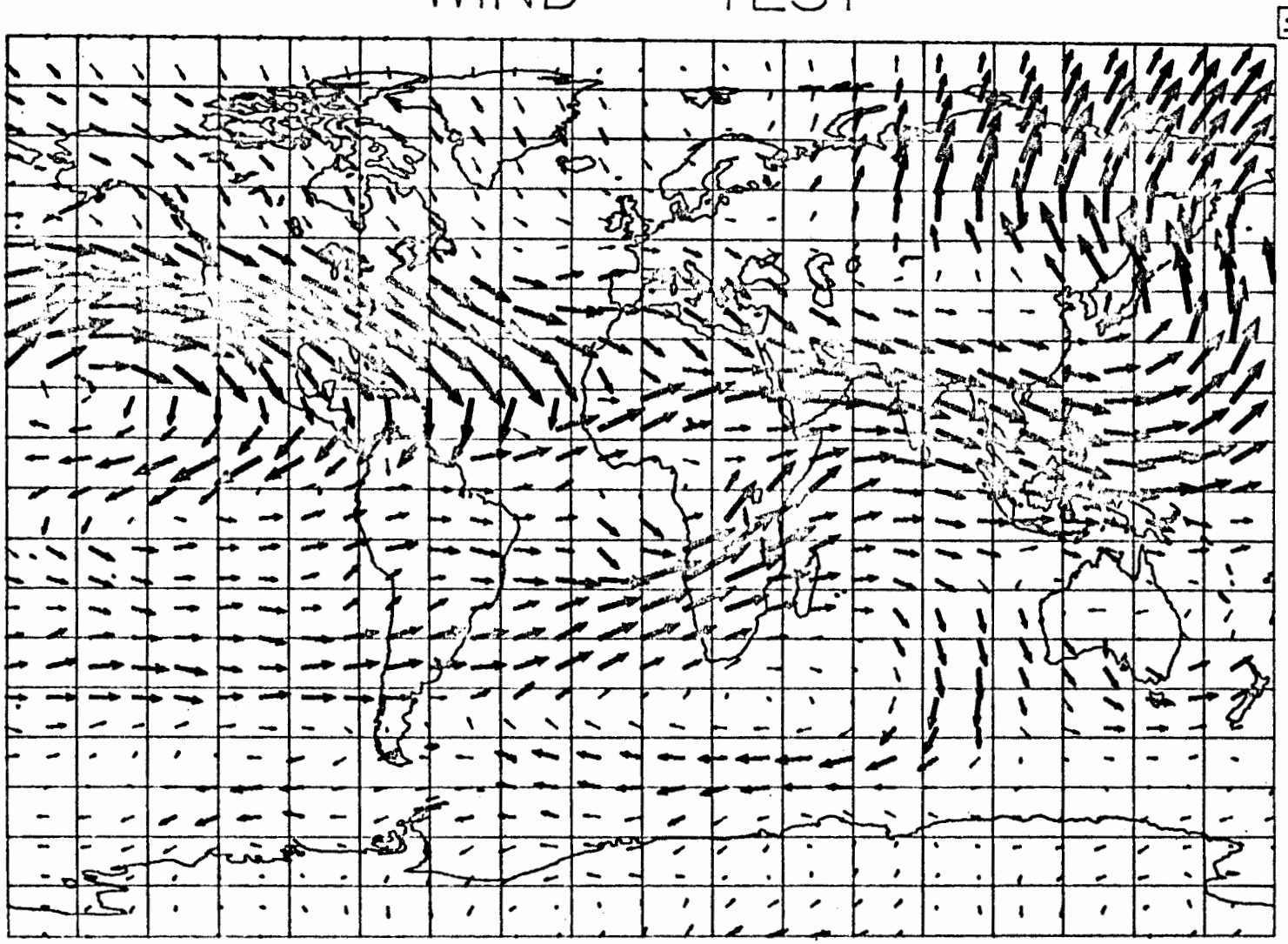
CAMARA, G.; VELASCO, F.; OLIVEIRA, J.R.F.; YAMAGUCHI, F.Y.; SOUZA, R.C.M. "Tipos Abstratos de Dados em Computação Gráfica e Processamento de Imagens". II Simpósio Brasileiro de Engenharia de Software , Canela , Outubro 1988.

CUNHA,G.; PIERY, F.; BERALDO, A; BATTAIOLA, A. "O Padrão GKS". São Paulo, ATLAS, 1987.

ENDERLE,G.; KANSY,K.; PFAFF,G. Computer Graphics Programming: GKS - The Graphics Standard. Berlin, Springer-Verlag, 1987.

VELOSO,P.A.S. Estruturação e Verificação de Programas com Tipos de Dados. São Paulo, Edgard Blücher, 1987.

WIND – TEST



MAGICS 2.0 PC/DOS – MICRO–MAGICS 1 August 1988 16: 8:12 – LOGO – TEST